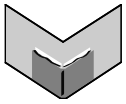
The Journal of Machine Learning Research
Volume 17
Print-Archive Edition

Issues 1–120



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 17
Print-Archive Edition

The Journal of Machine Learning Research (JMLR) is an open access journal. All articles published in JMLR are freely available via electronic distribution. This Print-Archive Edition is published annually as a means of archiving the contents of the journal in perpetuity. The contents of this volume are articles published electronically in JMLR in 2016.

JMLR is abstracted in ACM Computing Reviews, INSPEC, and Psychological Abstracts/PsycINFO.

JMLR is a publication of Journal of Machine Learning Research, Inc. For further information regarding JMLR, including open access to articles, visit http://www.jmlr.org/.

JMLR Print-Archive Edition is a publication of Microtome Publishing under agreement with Journal of Machine Learning Research, Inc. For further information regarding the Print-Archive Edition, including subscription and distribution information and background on open-access print archiving, visit Microtome Publishing at http://www.mtome.com/.

ISSN 1532-4435 (print)
ISSN 1533-7928 (online)

JMLR Editorial Board

Journal of Machine Learning Research

Volume 17, 2017

Part A

Part B

17(239):1–41

On the properties of variational approximations of Gibbs posteriors

Pierre Alquier, James Ridgway, Nicolas Chopin

Journal of Machine Learning Research 17 (2016) 1-42 Submitted 7/14; Revised 2/15; Published 1/16

On the Complexity of Best-Arm Identification in Multi-Armed Bandit Models

Emilie Kaufmann EMILIE.KAUFMANN@TELECOM-PARISTECH.FR
LTCI, CNRS, Télécom ParisTech
46, rue Barrault, 75013 Paris

Olivier Cappé OLIVIER.CAPPE@TELECOM-PARISTECH.FR
LTCI, CNRS, Télécom ParisTech
46, rue Barrault, 75013 Paris

Aurélien Garivier AURELIEN.GARIVIER@MATH.UNIV-TOULOUSE.FR
Institut de Mathématiques de Toulouse ; UMR5219
Université de Toulouse ; CNRS
UPS IMT, F-31062 Toulouse Cedex 9

Editor: Gábor Lugosi

Abstract

The stochastic multi-armed bandit model is a simple abstraction that has proven useful in many different contexts in statistics and machine learning. Whereas the achievable limit in terms of regret minimization is now well known, our aim is to contribute to a better understanding of the performance in terms of identifying the m best arms. We introduce generic notions of complexity for the two dominant frameworks considered in the literature: fixed-budget and fixed-confidence settings. In the fixed-confidence setting, we provide the first known distribution-dependent lower bound on the complexity that involves information-theoretic quantities and holds when $m \geq 1$ under general assumptions. In the specific case of two armed-bandits, we derive refined lower bounds in both the fixed-confidence and fixed-budget settings, along with matching algorithms for Gaussian and Bernoulli bandit models. These results show in particular that the complexity of the fixed-budget setting may be smaller than the complexity of the fixed-confidence setting, contradicting the familiar behavior observed when testing fully specified alternatives. In addition, we also provide improved sequential stopping rules that have guaranteed error probabilities and shorter average running times. The proofs rely on two technical results that are of independent interest : a deviation lemma for self-normalized sums (Lemma 7) and a novel change of measure inequality for bandit models (Lemma 1).

Keywords: multi-armed bandit, best-arm identification, pure exploration, information-theoretic divergences, sequential testing

1. Introduction

We investigate in this paper the complexity of finding the m best arms in a stochastic multi-armed bandit model. A bandit model ν is a collection of K arms, where each arm ν_a $(1 \leq a \leq K)$ is a probability distribution on $\mathbb{R}$ with expectation μ_a. At each time $t = 1, 2, \ldots$, an agent chooses an option $A_t \in \{1, \ldots, K\}$ and receives an independent draw Z_t from the corresponding arm ν_{A_t}. We denote by $\mathbb{P}_\nu$ (resp. $\mathbb{E}_\nu$) the probability law (resp. expectation) of the process (Z_t). The agent's goal is to identify the m best arms, that is, the set $\mathcal{S}_m^*$ of indices of the m arms with highest expectation. Letting $(\mu_{[1]}, \ldots, \mu_{[K]})$ be the K-tuple of expectations $(\mu_1, \ldots, \mu_K)$ sorted in decreasing order, we assume that the bandit model ν belongs to a class $\mathcal{M}_m$ such that for every $\nu \in \mathcal{M}_m$, $\mu_{[m]} > \mu_{[m+1]}$, so that $\mathcal{S}_m^*$ is unambiguously defined.

In order to identify $\mathcal{S}_m^*$, the agent must use a strategy defining which arms to sample from, when to stop sampling, and which set $\hat{S}_m$ to choose. More precisely, its strategy consists in a triple $\mathcal{A} = ((A_t), \tau, \hat{S}_m)$ in which :

- the *sampling rule* determines, based on past observations, which arm A_t is chosen at time t; in other words, A_t is $\mathcal{F}_{t-1}$-measurable, with $\mathcal{F}_t = \sigma(A_1, Z_1, \ldots, A_t, Z_t)$;
- the *stopping rule* τ controls the end of the data acquisition phase and is a stopping time with respect to $(\mathcal{F}_t)_{t\in\mathbb{N}}$ satisfying $\mathbb{P}(\tau < +\infty) = 1$;
- the *recommendation rule* provides the arm selection and is a $\mathcal{F}_\tau$-measurable random subset $\hat{S}_m$ of $\{1, \ldots, K\}$ of size m.

In the bandit literature, two different settings have been considered. In the *fixed-confidence setting*, a risk parameter δ is fixed, and a strategy $\mathcal{A}(\delta)$ is called *δ-PAC* if, for every choice of $\nu \in \mathcal{M}_m$, $\mathbb{P}_\nu(\hat{S}_m = \mathcal{S}_m^*) \geq 1 - \delta$. The goal is to obtain δ-PAC strategies that require a number of draws τ_δ that is as small as possible. More precisely, we focus on strategies minimizing the expected number of draws $\mathbb{E}_\nu[\tau_\delta]$, which is also called the *sample complexity*. The subscript δ in τ_δ will be omitted when there is no ambiguity. We call a family of strategies $A = (\mathcal{A}(\delta))_{\delta\in(0,1)}$ *PAC* if for every δ, $\mathcal{A}(\delta)$ is δ-PAC.

Alternatively, in the *fixed-budget setting*, the number of draws τ is fixed in advance to some value $t \in \mathbb{N}$ and for this budget t, the goal is to choose the sampling and recommendation rules of a strategy $\mathcal{A}(t)$ so as to minimize the failure probability $p_t(\nu) := \mathbb{P}_\nu(\hat{S}_m \neq S_m^*)$. In the fixed-budget setting, a family of strategies A$=(\mathcal{A}(t))_{t\in\mathbb{N}^*}$ is called *consistent* if, for every choice of $\nu \in \mathcal{M}_m$, $p_t(\nu)$ tends to zero when t increases to infinity.

In order to unify and compare these approaches, we define the *complexity* $\kappa_C(\nu)$ (resp. $\kappa_B(\nu)$) of best-arm identification in the fixed-confidence (resp. fixed-budget) setting as follows:

$$\kappa_{\mathrm{C}}(\nu) = \inf_{A \text{ PAC}} \limsup_{\delta\to 0} \frac{\mathbb{E}_\nu[\tau_\delta]}{\log \frac{1}{\delta}}, \qquad \kappa_{\mathrm{B}}(\nu) = \inf_{A \text{ consistent}} \left(\limsup_{t\to\infty} -\frac{1}{t}\log p_t(\nu)\right)^{-1}. \quad (1)$$

Heuristically, on the one hand for a given bandit model ν, and a small value of δ, a fixed-confidence optimal strategy needs an average number of samples of order $\kappa_C(\nu)\log\frac{1}{\delta}$ to identify the m best arms with probability at least $1-\delta$. On the other hand, for large values of t the probability of error of a fixed-budget optimal strategy is of order $\exp(-\kappa_B(\nu)t)$, which means that a budget of approximately $t = \kappa_B(\nu)\log\frac{1}{\delta}$ draws is required to ensure a probability of error of order δ. Most of the existing performance bounds for the fixed confidence and fixed budget settings can indeed be expressed using these complexity measures.

In this paper, we aim at evaluating and comparing these two complexities. To achieve this, two ingredients are needed: a lower bound on the sample complexity of any δ-PAC algorithm (resp. on the failure probability of any consistent algorithm) and a δ-PAC (resp.

consistent) strategy whose sample complexity (resp. failure probability) attains the lower bound (often referred to as a 'matching' strategy). We present below new lower bounds on $\kappa_C(\nu)$ and $\kappa_B(\nu)$ that feature information-theoretic quantities as well as strategies that match these lower bounds in two-armed bandit models.

A particular class of algorithms will be considered in the following: those using a *uniform sampling strategy*, that sample the arms in a round-robin fashion. Whereas it is well known that when $K > 2$ uniform sampling is not desirable, it will prove efficient in some examples of two-armed bandits. This specific setting, relevant in practical applications discussed in Section 3, is studied in greater details along the paper. In this case, an algorithm using uniform sampling can be regarded as a statistical test of the hypothesis $H_0 : (\mu_1 \leq \mu_2)$ against $H_1 : (\mu_1 > \mu_2)$ based on paired samples (X_s, Y_s) of ν_1, ν_2; namely a test based on a fixed number of samples in the fixed-budget setting, and, a *sequential test* in the fixed-confidence setting, in which a (random) stopping rule determines when the experiment is to be terminated.

Classical sequential testing theory provides a first element of comparison between the fixed-budget and fixed-confidence settings, in the simpler case of fully specified alternatives. Consider for instance the case where ν_1 and ν_2 are Gaussian laws with the same known variance σ^2, the means μ_1 and μ_2 being known up to a permutation. Denoting by P the joint distribution of the paired samples (X_s, Y_s), one must choose between the hypotheses $H_0 : P = \mathcal{N}\left(\mu_1, \sigma^2\right) \otimes \mathcal{N}\left(\mu_2, \sigma^2\right)$ and $H_1 : P = \mathcal{N}\left(\mu_2, \sigma^2\right) \otimes \mathcal{N}\left(\mu_1, \sigma^2\right)$. It is known since Wald (1945) that among the sequential tests such that type I and type II error probabilities are both smaller than δ, the Sequential Probability Ratio Test (SPRT) minimizes the expected number of required samples, and is such that $\mathbb{E}_\nu[\tau] \simeq 2\sigma^2/(\mu_1 - \mu_2)^2 \log(1/\delta)$. However, the batch test that minimizes both probabilities of error is the Likelihood Ratio test; it can be shown to require a sample size of order $8\sigma^2/(\mu_1 - \mu_2)^2 \log(1/\delta)$ in order to ensure that both type I and type II error probabilities are smaller than δ. Thus, when the sampling strategy is uniform and the parameters are known, there is a clear gain in using randomized stopping strategies. We will show below that this conclusion is not valid anymore when the values of μ_1 and μ_2 are not assumed to be known. Indeed, for two-armed Gaussian bandit models we show that $\kappa_B(\nu) = \kappa_C(\nu)$ and for two-armed Bernoulli bandit models we show that $\kappa_C(\nu) > \kappa_B(\nu)$.

1.1 Related Works

Bandit models have received a considerable interest since their introduction by Thompson (1933) in the context of medical trials. An important focus was set on a different perspective, in which each observation is considered as a reward: the agent aims at maximizing its cumulative rewards. Equivalently, his goal is to minimize the expected *regret* up to horizon $t \geq 1$ defined as $R_t(\nu) = t\mu_{[1]} - \mathbb{E}_\nu\left[\sum_{s=1}^t Z_s\right]$. Regret minimization, which is paradigmatic of the so-called *exploration versus exploitation dilemma*, was introduced by Robbins (1952) and its complexity is well understood for simple families of parametric bandits. In generic one-parameter models, Lai and Robbins (1985) prove that, with a proper notion of consistency adapted to regret minimization,

$$\inf_{A \text{ consistent}} \liminf_{t\to\infty} \frac{R_t(\nu)}{\log t} \geq \sum_{a:\mu_a<\mu_{[1]}} \frac{(\mu_{[1]} - \mu_a)}{\mathrm{KL}(\nu_a, \nu_{[1]})} ,$$

where $\mathrm{KL}(\nu_i, \nu_j)$ denotes the Kullback-Leibler divergence between distributions ν_i and ν_j. This bound was later generalized by Burnetas and Katehakis (1996) to distributions that depend on several parameters. Since then, non-asymptotic analyses of efficient algorithms matching this bound have been proposed. Optimal algorithms include the KL-UCB algorithm of Cappé et al. (2013)—a variant of UCB1 (Auer et al., 2002) using informational upper bounds, Thompson Sampling (Kaufmann et al., 2012b; Agrawal and Goyal, 2013), the DMED algorithm (Honda and Takemura, 2011) and Bayes-UCB (Kaufmann et al., 2012a). This paper is a contribution towards similarly characterizing the complexity of *pure exploration*, where the goal is to determine the best arms without trying to maximize the cumulative observations.

Bubeck et al. (2011) show that in the fixed-budget setting, when $m = 1$, any sampling strategy designed to minimize regret performs poorly with respect to the *simple regret* $r_t := \mu^* - \mu_{\hat{S}_1}$, which is closely related to the probability $p_t(\nu)$ of recommending the wrong arm. Therefore, good strategies for best-arm identification need to be quite different from regret-minimizing strategies. We will show below that the complexities $\kappa_B(\nu)$ and $\kappa_C(\nu)$ of best-arm identification also involve information terms, but these are different from the Kullback-Leibler divergence featured in Lai and Robbins' lower bound on the regret.

The problem of best-arm identification has been studied since the 1950s under the name 'ranking and identification problems'. The first advances on this topic are summarized in the monograph by Bechhofer et al. (1968) who consider the fixed-confidence setting and strategies based on uniform sampling. In the fixed confidence setting, Paulson (1964) first introduces a sampling strategy based on eliminations for single best arm identification: the arms are successively discarded, the remaining arms being sampled uniformly. This idea was later used for example by Jennison et al. (1982); Maron and Moore (1997) and by Even-Dar et al. (2006) in the context of bounded bandit models, in which each arm ν_a is a probability distribution on $[0, 1]$. m best arms identification with $m > 1$ was considered for example by Heidrich-Meisner and Igel (2009), in the context of reinforcement learning. Kalyanakrishnan et al. (2012) later proposed an algorithm that is no longer based on eliminations, called LUCB (for Lower and Upper Confidence Bounds) and still designed for bounded bandit models. Bounded distributions are in fact particular examples of distributions with subgaussian tails, to which the proposed algorithms can be easily generalized. A relevant quantity introduced in the analysis of algorithms for bounded (or subgaussian) bandit models is the 'complexity term'

$$H(\nu) = \sum_{a\in\{1,2,\dots K\}} \frac{1}{\Delta_a^2} \quad \text{with} \quad \Delta_a = \begin{cases} \mu_a - \mu_{[m+1]} & \text{for } a \in \mathcal{S}_m^*, \\ \mu_{[m]} - \mu_a & \text{for } a \in (\mathcal{S}_m^*)^c. \end{cases} \tag{2}$$

The upper bound on the sample complexity of the LUCB algorithm of Kalyanakrishnan et al. (2012) implies in particular that $\kappa_C(\nu) \leq 292H(\nu)$. Some of the existing works on the fixed-confidence setting do not bound τ in expectation but rather show that $\mathbb{P}_\nu(\hat{\mathcal{S}}_m = \mathcal{S}_m^*, \tau = O\left(H(\nu)\right)) \geq 1 - \delta$. These results are not directly comparable with the complexity $\kappa_C(\nu)$, although no significant gap is to be observed yet.

Recent works have focused on obtaining upper bounds on the number of samples whose dependency in terms of the squared-gaps Δ_a (for subgaussian arms) is optimal when the Δ_a's go to zero, and δ remains fixed. Karnin et al. (2013) and Jamieson et al. (2014) exhibit

δ-PAC algorithms for which there exists a constant C such that, with high probability, the number of samples used satisfies

$$\tau \leq C_0 \sum_{a \neq a^*} \frac{1}{\Delta_a^2} \log\left(\frac{1}{\delta} \log \frac{1}{\Delta_a}\right),$$

and Jamieson et al. (2014) show that the dependency in $\Delta_a^{-2} \log(\log(\Delta_a^{-1}))$ is optimal when Δ_a goes to zero. However, the constant C_0 is large and does not lead to improved upper bounds on the complexity term $\kappa_C(\nu)$.

For $m = 1$, the work of Mannor and Tsitsiklis (2004) provides a lower bound on $\kappa_C(\nu)$ in the case of Bernoulli bandit models, under the following ϵ-relaxation sometimes considered in the literature. For some tolerance parameter $\epsilon \geq 0$ the agent has to ensure that $\hat{S}_m$ is included in the set of (ϵ, m) optimal arms $\mathcal{S}^*_{m,\epsilon} = \{a : \mu_a \geq \mu_{[m]} - \epsilon\}$ with probability at least $1 - \delta$. This relaxation has to be considered, for example, when $\mu_{[m]} = \mu_{[m+1]}$, but has never been considered in the literature for the fixed-budget setting. In this paper, we focus on the case $\epsilon = 0$ that allows for a comparison between the fixed-confidence and fixed-budget settings. Mannor and Tsitsiklis (2004) show that if an algorithm is δ-PAC, then in the bandit model $\nu = (\mathcal{B}(\mu_1), \dots, \mathcal{B}(\mu_K))$ such that $\forall a$, $\mu_a \in [0, \alpha]$ for some $\alpha \in (0, 1)$, there exists two sets $\mathcal{G}_\alpha(\nu) \subset \mathcal{S}_1^*$ and $\mathcal{H}_\alpha(\nu) \subset \{1, \dots, K\} \backslash \mathcal{S}_1^*$ and a positive constant C_α such that

$$\mathbb{E}_\nu[\tau] \geq C_\alpha \left(\sum_{a \in \mathcal{G}_\alpha(\nu)} \frac{1}{\epsilon^2} + \sum_{a \in \mathcal{H}_\alpha(\nu)} \frac{1}{(\mu_{[1]} - \mu_a)^2} \right) \log\left(\frac{1}{8\delta}\right).$$

This bound is non asymptotic (as emphasized by the authors), although not completely explicit. In particular, the subset $\mathcal{G}_\alpha$ and $\mathcal{H}_\alpha$ do not always form a partition of the arms (it can happen that $\mathcal{G}_\alpha \cup \mathcal{H}_\alpha \neq \{1, \dots, K\}$), hence the complexity term does not involve a sum over all the arms. For $m > 1$, the only lower bound available in the literature is the worst-case result of Kalyanakrishnan et al. (2012). It states that for every δ-PAC algorithm *there exists* a bandit model ν such that $\mathbb{E}_\nu[\tau] \geq K/(18375\epsilon^2) \log(m/8\delta)$. This result, however, does not provide a lower bound on the complexity $\kappa_C(\nu)$.

The fixed-budget setting has been studied by Audibert et al. (2010); Bubeck et al. (2011) for single best-arm identification in bounded bandit models. For multiple arm identification ($m > 1$), still in bounded bandit models, Bubeck et al. (2013b) introduce the SAR (for Successive Accepts and Rejects) algorithm. An upper bound on the failure probability of the SAR algorithm yields $\kappa_B(\nu) \leq 8 \log(K) H(\nu)$.

For $m = 1$, Audibert et al. (2010) prove an asymptotic lower bound on the probability of error for Bernoulli bandit models. They state that for every algorithm and every bandit problem ν such that $\forall a$, $\mu_1 \in [\alpha, 1 - \alpha]$, there exists a permutation of the arms ν' such that

$$p_t(\nu') \geq \exp(-t/C_\alpha H_2(\nu'))), \quad \text{with} \quad H_2(\nu) = \max_{i; \mu_{[i]} < \mu_{[1]}} \frac{i}{\left(\mu_{[1]} - \mu_{[i]}\right)^2} \quad \text{and} \quad C_\alpha = \frac{\alpha(1-\alpha)}{5 + o(1)}.$$

Gabillon et al. (2012) propose the UGapE algorithm for m best-arm identification for $m > 1$. By changing only one parameter in some confidence regions, this algorithm can be adapted either to the fixed-budget or to the fixed-confidence setting. However, a careful inspection shows that UGapE cannot be used in the fixed-budget setting without the

knowledge of the complexity term $H(\nu)$. This drawback is shared by other algorithms designed for the fixed-budget setting, like the UCB-E algorithm of Audibert et al. (2010) or the KL-LUCB-E algorithm of Kaufmann and Kalyanakrishnan (2013).

1.2 Content of the Paper

The gap between lower and upper bounds known so far does not permit to identify exactly the complexity terms $\kappa_B(\nu)$ and $\kappa_C(\nu)$ defined in (1). Not only do they involve imprecise multiplicative constants but by analogy with the Lai and Robbins' bound for the expected regret, the quantities $H(\nu), H_2(\nu)$ presented above are only expected to be relevant in the Gaussian case.

The improvements of this paper mainly concern the fixed-confidence setting, which will be considered in the next three Sections. We first propose in Section 2 a distribution-dependent lower bound on $\kappa_C(\nu)$ that holds for $m > 1$ and for general classes of bandit models (Theorem 4). This information-theoretic lower bound permits to interpret the quantity $H(\nu)$ defined in (2) as a subgaussian approximation.

Theorem 6 in Section 3 proposes a tighter lower bound on $\kappa_C(\nu)$ for general classes of two-armed bandit models, as well as a lower bound on the sample complexity of δ-PAC algorithms using uniform sampling. In Section 4 we propose, for Gaussian bandits with known—but possibly different—variances, an algorithm exactly matching this bound. We also consider the case of Bernoulli distributed arms, for which we show that uniform sampling is nearly optimal in most cases. We propose a new algorithm using uniform sampling and a non-trivial stopping strategy that is close to matching the lower bound.

Section 5 gathers our contributions to the fixed-budget setting. For two-armed bandits, Theorem 12 provides a lower bound on $\kappa_B(\nu)$ that is in general different from the lower bound obtained for $\kappa_C(\nu)$ in the fixed-confidence setting. Then we propose matching algorithms for the fixed-budget setting that allow for a comparison between the two settings. For Gaussian bandits, we show that $\kappa_C(\nu) = \kappa_B(\nu)$, whereas for Bernoulli bandits $\kappa_C(\nu) > \kappa_B(\nu)$, proving that the two complexities are not necessarily equal. As a first step towards a lower bound on $\kappa_B(\nu)$ when $m > 1$, we also give in Section 5 new lower bounds on the probability of error $p_t(\nu)$ of any consistent algorithm, for Gaussian bandit models.

Section 6 contains numerical experiments that illustrate the performance of matching algorithms for Gaussian and Bernoulli two-armed bandits, comparing the fixed-confidence and fixed-budget settings.

Our contributions follow from two main mathematical results of more general interest. Lemma 1 provides a general relation between the expected number of draws and Kullback-Leibler divergences of the arms' distributions, which is the key element to derive the lower bounds (it also permits, for example, to derive Lai and Robbin's lower bound on the regret in a few lines). Lemma 7 is a tight deviation inequality for martingales with sub-Gaussian increments, in the spirit of the Law of Iterated Logarithm, that permits here to derive efficient matching algorithms for two-armed bandits.

2. Generic Lower Bound in the Fixed-Confidence Setting

Introducing the Kullback-Leibler divergence of any two probability distributions p and q:

$$\mathrm{KL}(p,q) = \begin{cases} \int \log\left[\frac{dp}{dq}(x)\right] dp(x) \text{ if } q \ll p, \\ +\infty \text{ otherwise,} \end{cases}$$

we make the assumption that there exists a set $\mathcal{P}$ of probability measures such that for all $\nu = (\nu_1, \dots, \nu_K) \in \mathcal{M}_m$, for $a \in \{1, \dots, K\}, \nu_a \in \mathcal{P}$ and that $\mathcal{P}$ satisfies

$$\forall p, q \in \mathcal{P},\ p \neq q \ \Rightarrow\ 0 < \mathrm{KL}(p,q) < +\infty.$$

A class $\mathcal{M}_m$ of bandit models satisfying this property is called *identifiable*.

All the distribution-dependent lower bounds derived in the bandit literature (e.g., Lai and Robbins, 1985; Mannor and Tsitsiklis, 2004; Audibert et al., 2010) rely on *changes of distribution*, and so do ours. A change of distribution relates the probabilities of the same event under two different bandit models ν and ν'. The following lemma provides a new, synthetic, inequality from which lower bounds are directly derived. This result, proved in Appendix A, encapsulates the technical aspects of the change of distribution. The main ingredient in its proof is a lower bound on the *expected log-likelihood ratio* of the observations under two different bandit models which is of interest on its own and is stated as Lemma 19 in Appendix A. To illustrate the interest of Lemma 1 even beyond the pure exploration framework, we give in Appendix B a new, simple proof of Burnetas and Katehakis (1996)'s generalization of Lai and Robbins' lower bound in the regret minimization framework based on Lemma 1.

Let $N_a(t) = \sum_{s=1}^{t} \mathbb{1}_{\{A_s = a\}}$ be the number of draws of arm a between the instants 1 and t and $N_a = N_a(\tau)$ be the total number of draws of arm a by some algorithm $\mathcal{A} = ((A_t), \tau, \hat{S}_m)$.

Lemma 1 *Let ν and ν' be two bandit models with K arms such that for all a, the distributions ν_a and ν'_a are mutually absolutely continuous. For any almost-surely finite stopping time σ with respect to $(\mathcal{F}_t)$,*

$$\sum_{a=1}^{K} \mathbb{E}_\nu[N_a(\sigma)]\mathrm{KL}(\nu_a, \nu'_a) \geq \sup_{\mathcal{E} \in \mathcal{F}_\sigma} d(\mathbb{P}_\nu(\mathcal{E}), \mathbb{P}_{\nu'}(\mathcal{E})),$$

where $d(x,y) := x\log(x/y) + (1-x)\log((1-x)/(1-y))$ is the binary relative entropy, with the convention that $d(0,0) = d(1,1) = 0$.

Remark 2 *This result can be considered as a generalization of Pinsker's inequality to bandit models: in combination with the inequality $d(p,q) \geq 2(p-q)^2$, it yields:*

$$\sup_{\mathcal{E} \in \mathcal{F}_\sigma} |\mathbb{P}_\nu(\mathcal{E}) - \mathbb{P}_{\nu'}(\mathcal{E})| \leq \sqrt{\frac{\sum_{a=1}^{K} \mathbb{E}_\nu[N_a(\sigma)]\mathrm{KL}(\nu_a, \nu'_a)}{2}}\ .$$

However, it is important in this paper not to use this weaker form *of the statement, as we will consider events $\mathcal{E}$ of probability very close to 0 or 1. In this regime, we will make use of the following inequality:*

$$\forall\ x \in [0,1], \qquad d(x, 1-x) \geq \log\frac{1}{2.4x}\ , \tag{3}$$

which can be checked easily.

2.1 Lower Bound on the Sample Complexity of a δ-PAC Algorithm

We now propose a non-asymptotic lower bound on the expected number of samples needed to identify the m best arms in the fixed confidence setting, which straightforwardly yields a lower bound on $\kappa_C(\nu)$.

Theorem 4 holds for an identifiable class of bandit models of the form:

$$\mathcal{M}_m = \{\nu = (\nu_1, \dots, \nu_K) : \nu_i \in \mathcal{P}, \mu_{[m]} > \mu_{[m+1]}\} \tag{4}$$

such that the set of probability measures $\mathcal{P}$ satisfies Assumption 3 below.

Assumption 3 *For all $p, q \in \mathcal{P}^2$ such that $p \neq q$, for all $\alpha > 0$,*
there exists $q_1 \in \mathcal{P}$: $\mathrm{KL}(p,q) < \mathrm{KL}(p,q_1) < \mathrm{KL}(p,q) + \alpha$ and $\mathbb{E}_{X\sim q_1}[X] > \mathbb{E}_{X\sim q}[X]$,
there exists $q_2 \in \mathcal{P}$: $\mathrm{KL}(p,q) < \mathrm{KL}(p,q_2) < \mathrm{KL}(p,q) + \alpha$ and $\mathbb{E}_{X\sim q_2}[X] < \mathbb{E}_{X\sim q}[X]$.

These continuity conditions are reminiscent of the assumptions of Lai and Robbins (1985); they include families of parametric bandits continuously parameterized by their means (e.g., Bernoulli, Poisson, exponential distributions).

Theorem 4 *Let $\nu \in \mathcal{M}_m$, where $\mathcal{M}_m$ is defined by (4), and assume that $\mathcal{P}$ satisfies Assumption 3; any algorithm that is δ-PAC on $\mathcal{M}_m$ satisfies, for $\delta \leq 0.15$,*

$$\mathbb{E}_\nu[\tau] \geq \left[\sum_{a \in \mathcal{S}_m^*} \frac{1}{\mathrm{KL}(\nu_a, \nu_{[m+1]})} + \sum_{a \notin \mathcal{S}_m^*} \frac{1}{\mathrm{KL}(\nu_a, \nu_{[m]})}\right] \log\left(\frac{1}{2.4\delta}\right).$$

Proof. Without loss of generality, one may assume that the arms are ordered such that $\mu_1 \geq \cdots \geq \mu_K$. Thus $\mathcal{S}_m^* = \{1, ..., m\}$. Let $\mathcal{A} = ((A_t), \tau, \hat{S}_m)$ be a δ-PAC algorithm and fix $\alpha > 0$. For all $a \in \{1, \dots, K\}$, from Assumption 3 there exists an alternative model

$$\nu' = (\nu_1, \dots, \nu_{a-1}, \nu'_a, \nu_{a+1}, \dots, \nu_K)$$

in which the only arm modified is arm a, and ν'_a is such that:

- $\mathrm{KL}(\nu_a, \nu_{m+1}) < \mathrm{KL}(\nu_a, \nu'_a) < \mathrm{KL}(\nu_a, \nu_{m+1}) + \alpha$ and $\mu'_a < \mu_{m+1}$ if $a \in \{1, \dots, m\}$,
- $\mathrm{KL}(\nu_a, \nu_m) < \mathrm{KL}(\nu_a, \nu'_a) < \mathrm{KL}(\nu_a, \nu_m) + \alpha$ and $\mu'_a > \mu_m$ if $a \in \{m+1, \dots, K\}$.

In particular, on the bandit model ν' the set of optimal arms is no longer $\{1, \dots, m\}$. Thus, introducing the event $\mathcal{E} = (\hat{\mathcal{S}}_m = \{1, \dots, m\}) \in \mathcal{F}_\tau$, any δ-PAC algorithm satisfies $\mathbb{P}_\nu(\mathcal{E}) \geq 1 - \delta$ and $\mathbb{P}_{\nu'}(\mathcal{E}) \leq \delta$. Lemma 1 applied to the stopping time τ (such that $N_a(\tau) = N_a$ is the total number of draws of arm a) and the monotonicity properties of $d(x,y)$ ($x \mapsto d(x,y)$ is increasing when $x > y$ and decreasing when $x < y$) yield

$$\mathrm{KL}(\nu_a, \nu'_a)\mathbb{E}_\nu[N_a] \geq d(1-\delta, \delta) \geq \log(1/2.4\delta),$$

where the last inequality follows from (3). From the definition of the alternative model, one obtains for $a \in \{1, \dots, m\}$ or $b \in \{m+1, \dots, K\}$ respectively, for every $\alpha > 0$,

$$\mathbb{E}_\nu[N_a] \geq \frac{\log(1/2.4\delta)}{\mathrm{KL}(\nu_a, \nu_{m+1}) + \alpha} \quad \text{and} \quad \mathbb{E}_\nu[N_b] \geq \frac{\log(1/2.4\delta)}{\mathrm{KL}(\nu_b, \nu_m) + \alpha}.$$

Letting α tend to zero and summing over the arms yields the bound on $\mathbb{E}_\nu[\tau] = \sum_{a=1}^{K} \mathbb{E}_\nu[N_a]$.

Remark 5 *This inequality can be made tighter for values of δ that are sufficiently close to zero, for which the right-hand-side can then be made arbitrarily close to* $\log(1/\delta)$.

Lemma 1 can also be used to improve the result of Mannor and Tsitsiklis (2004) that holds for $m = 1$ under the ϵ-relaxation described before. Combining the changes of distribution of this paper with Lemma 1 yields, for every $\epsilon > 0$ and $\delta \leq 0.15$,

$$\mathbb{E}_\nu[\tau] \geq \left(\frac{|\{a : \mu_a \geq \mu_{[1]} - \epsilon\}| - 1}{\mathrm{KL}\left(\mathcal{B}(\mu_{[1]}), \mathcal{B}(\mu_{[1]} - \epsilon)\right)} + \sum_{a:\mu_a \leq \mu_{[1]} - \epsilon} \frac{1}{\mathrm{KL}\left(\mathcal{B}(\mu_a), \mathcal{B}(\mu_{[1]} + \epsilon)\right)} \right) \log \frac{1}{2.4\delta},$$

where $|\mathcal{X}|$ denotes the cardinal of the set $\mathcal{X}$ and $\mathcal{B}(\mu)$ the Bernoulli distribution of mean μ.

2.2 Bounds on the Complexity for Exponential Bandit Models

Theorem 4 yields the following lower bound on the complexity term:

$$\kappa_C(\nu) \geq \sum_{a \in \mathcal{S}_m^*} \frac{1}{\mathrm{KL}(\nu_a, \nu_{[m+1]})} + \sum_{a \notin \mathcal{S}_m^*} \frac{1}{\mathrm{KL}(\nu_a, \nu_{[m]})}.$$

Thus, one may want to obtain strategies whose sample complexity can be proved to be of the same magnitude. The only algorithm that has been analyzed so far with an information-theoretic perspective is the KL-LUCB algorithm of Kaufmann and Kalyanakrishnan (2013), designed for *exponential bandit models*: that is

$$\mathcal{M}_m = \left\{ \nu = (\nu_{\theta_1}, \dots, \nu_{\theta_K}) : (\theta_1, \dots, \theta_K) \in \Theta^K, \theta_{[m]} > \theta_{[m+1]} \right\},$$

where ν_θ belongs to a *canonical one-parameter exponential family*. This means that there exists a twice differentiable strictly convex function b such that ν_θ has a density with respect to some reference measure given by

$$f_\theta(x) = \exp(\theta x - b(\theta)), \quad \text{for } \theta \in \Theta \subset \mathbb{R}. \tag{5}$$

Distributions from a canonical one-parameter exponential family can be parameterized either by their natural parameter θ or by their mean. Indeed $\dot{b}(\theta) = \mu(\theta)$, the mean of the distribution ν_θ and $\ddot{b}(\theta) = \mathrm{Var}[\nu_\theta] > 0$. The mapping $\theta \mapsto \mu(\theta)$ is strictly increasing, and the means are ordered in the same way as the natural parameters. Exponential families include in particular Bernoulli distributions, or Gaussian distributions with common variances (see Cappé et al. (2013) for more details about exponential families).

We introduce the following shorthand to denote the Kullback-Leibler divergence in exponential families: $\mathrm{K}(\theta, \theta') = \mathrm{KL}(\nu_\theta, \nu_{\theta'})$ for $(\theta, \theta') \in \Theta^2$. Combining the upper bound on the sample complexity of the KL-LUCB algorithm obtained by Kaufmann and Kalyanakrishnan (2013) and the lower bound of Theorem 4, the complexity $\kappa_C(\nu)$ can be bounded as

$$\sum_{a \in \mathcal{S}_m^*} \frac{1}{\mathrm{K}(\theta_a, \theta_{[m+1]})} + \sum_{a \notin \mathcal{S}_m^*} \frac{1}{\mathrm{K}(\theta_a, \theta_{[m]})} \leq \kappa_C(\nu) \leq 24 \min_{\theta \in [\theta_{[m+1]}, \theta_{[m]}]} \sum_{a=1}^K \frac{1}{\mathrm{K}^*(\theta_a, \theta)}, \tag{6}$$

where $\mathrm{K}^*(\theta, \theta')$ is the Chernoff information between the distributions ν_θ and $\nu_{\theta'}$ (see Cover and Thomas (2006) and Kaufmann and Kalyanakrishnan (2013) for earlier notice of the

relevance of this quantity in the best-arm selection problem). Chernoff information is defined as follows and illustrated in Figure 1:

$$\mathrm{K}^*(\theta, \theta') = \mathrm{K}(\theta^*, \theta), \text{ where } \theta^* \text{ is such that } \mathrm{K}(\theta^*, \theta) = \mathrm{K}(\theta^*, \theta').$$

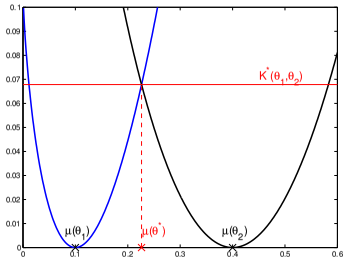


Figure 1: For Bernoulli distributions, the blue and black curves represent respectively $\mathrm{KL}(\mathcal{B}(\mu), \mathcal{B}(\mu_1))$ and $\mathrm{KL}(\mathcal{B}(\mu), \mathcal{B}(\mu_2))$ as a function of μ. Their intersection gives the value of the Chernoff information between $\mathcal{B}(\mu_1)$ and $\mathcal{B}(\mu_2)$, two distributions alternatively parameterized by their natural parameter θ_1 and θ_2.

3. Improved Lower Bounds for Two-Armed Bandits

Two armed-bandits are of particular interest as they offer a theoretical framework for sequential A/B Testing. A/B Testing is a popular procedure used, for instance, for website optimization: two versions of a web page, say A and B, are empirically compared by being presented to users. Each user is shown only one version $A_t \in \{1, 2\}$ and provides a real-valued index of the quality of the page, Z_t, which is modeled as a sample of a probability distribution ν_1 or ν_2. For example, a standard objective is to determine which web page has the highest conversion rate (probability that a user actually becomes a customer) by receiving binary feedback from the users. In standard A/B Testing algorithms, the two versions are presented equally often. It is thus of particular interest to investigate whether uniform sampling is optimal or not.

Even for two-armed bandits, the upper and lower bounds on the complexity $\kappa_C(\nu)$ given in (6) do not match. We propose in this section a refined lower bound on $\kappa_C(\nu)$ based on a different change of distribution. This lower bound features a quantity reminiscent of Chernoff information, and we will exhibit algorithms matching (or approximately matching) this new bound in Section 4. Theorem 6 provides a non-asymptotic lower bound on the sample complexity $\mathbb{E}_\nu[\tau]$ of any δ-PAC algorithm. It also provides a lower bound on the performance of algorithms using a uniform sampling strategy, which will turn out to be efficient in some cases.

Theorem 6 *Let $\mathcal{M}$ be an identifiable class of two-armed bandit models and let $\nu = (\nu_1, \nu_2) \in \mathcal{M}$ be such that $\mu_1 > \mu_2$. Any algorithm that is δ-PAC on $\mathcal{M}$ satisfies, for all $\delta \in]0,1]$,*

$$\mathbb{E}_\nu[\tau] \geq \frac{1}{c_*(\nu)} \log\left(\frac{1}{2.4\delta}\right), \quad \textit{where} \quad c_*(\nu) := \inf_{(\nu_1',\nu_2') \in \mathcal{M}: \mu_1' < \mu_2'} \max\left\{\mathrm{KL}(\nu_1,\nu_1'), \mathrm{KL}(\nu_2,\nu_2')\right\}.$$

Moreover, any δ-PAC algorithm using a uniform sampling strategy satisfies,

$$\mathbb{E}_\nu[\tau] \geq \frac{1}{I_*(\nu)} \log\left(\frac{1}{2.4\delta}\right), \ \textit{where} \ I_*(\nu) := \inf_{(\nu_1',\nu_2') \in \mathcal{M}: \mu_1' < \mu_2'} \frac{\mathrm{KL}\left(\nu_1,\nu_1'\right) + \mathrm{KL}\left(\nu_2,\nu_2'\right)}{2}. \tag{7}$$

Obviously, one has $I_*(\nu) \leq c_*(\nu)$. Theorem 6 implies in particular that $\kappa_C(\nu) \geq 1/c_*(\nu)$. It is possible to give explicit expressions for the quantities $c_*(\nu)$ and $I_*(\nu)$ for important classes of parametric bandit models that will be considered in the next section.

The class of Gaussian bandits with known variances σ_1^2 and σ_2^2, further considered in Section 4.1, is

$$\mathcal{M} = \{\nu = \left(\mathcal{N}\left(\mu_1,\sigma_1^2\right), \mathcal{N}\left(\mu_2,\sigma_2^2\right)\right) : (\mu_1,\mu_2) \in \mathbb{R}^2, \mu_1 \neq \mu_2\}. \tag{8}$$

For this class,

$$\mathrm{KL}(\mathcal{N}\left(\mu_1,\sigma_1\right), \mathcal{N}\left(\mu_2,\sigma_2\right)) = \frac{(\mu_1-\mu_2)^2}{2\sigma_2^2} + \frac{1}{2}\left[\frac{\sigma_1^2}{\sigma_2^2} - 1 - \log\frac{\sigma_1^2}{\sigma_2^2}\right] \tag{9}$$

and direct computations yield

$$c_*(\nu) = \frac{(\mu_1-\mu_2)^2}{2(\sigma_1+\sigma_2)^2} \quad \text{and} \quad I_*(\nu) = \frac{(\mu_1-\mu_2)^2}{4(\sigma_1^2+\sigma_2^2)}.$$

The observation that, when the variances are different $c_*(\nu) > I_*(\nu)$, will be shown to imply that strategies based on uniform sampling are sub-optimal (by a factor $1 \leq 2(\sigma_1^2+\sigma_2^2)/(\sigma_1+\sigma_2)^2 \leq 2$).

The more general class of two-armed exponential bandit models, further considered in Section 4.2, is

$$\mathcal{M} = \{\nu = (\nu_{\theta_1}, \nu_{\theta_2}) : (\theta_1,\theta_2) \in \Theta^2, \theta_1 \neq \theta_2\}$$

where ν_{θ_a} has density f_{θ_a} given by (5). There

$$c_*(\nu) = \inf_{\theta \in \Theta} \max\left(\mathrm{K}(\theta_1,\theta), \mathrm{K}(\theta_2,\theta)\right) = \mathrm{K}_*(\theta_1,\theta_2),$$

where $\mathrm{K}_*(\theta_1,\theta_2) = \mathrm{K}(\theta_1,\theta_*)$, with θ_* is defined by $\mathrm{K}(\theta_1,\theta_*) = \mathrm{K}(\theta_2,\theta_*)$. This quantity is analogous to the Chernoff information $\mathrm{K}^*(\theta_1,\theta_2)$ introduced in Section 2 but with 'reversed' roles for the arguments. $I_*(\nu)$ may also be expressed more explicitly as

$$I_*(\nu) = \frac{\mathrm{K}\left(\theta_1,\overline{\theta}\right) + \mathrm{K}\left(\theta_2,\overline{\theta}\right)}{2}, \quad \text{where} \quad \mu(\overline{\theta}) = \frac{\mu_1+\mu_2}{2}.$$

Appendix C provides further useful properties of these quantities and in particular Figure 7 illustrates the property that for two-armed exponential bandit models, the lower bound on $\kappa_C(\nu)$ provided by Theorem 6,

$$\kappa_C(\nu) \geq \left(\frac{1}{\mathrm{K}_*(\theta_1,\theta_2)}\right), \tag{10}$$

is indeed always tighter than the lower bound of Theorem 4,

$$\kappa_C(\nu) \geq \left(\frac{1}{\mathrm{K}(\theta_1,\theta_2)} + \frac{1}{\mathrm{K}(\theta_2,\theta_1)}\right). \tag{11}$$

Interestingly, the changes of distribution used to derive the two results are not the same. On the one hand, for inequality (11), the changes of distribution involved modify a single arm at a time: one of the arms is moved just below (or just above) the other (see Figure 2, left). This is the idea also used, for example, to obtain the lower bound of Lai and Robbins (1985) on the cumulative regret. On the other hand, for inequality (10), both arms are modified at the same time: they are moved close to the common intermediate value θ_* but with a reversed ordering (see Figure 2, right).

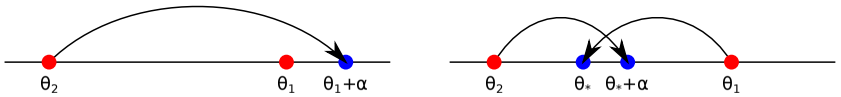


Figure 2: Alternative bandit models considered to obtain the lower bounds of Theorem 4 (left) and Theorem 6 (right).

We now give the proof of Theorem 6, in order to show how easily it follows from Lemma 1.

Proof of Theorem 6. Without loss of generality, one may assume that the bandit model $\nu = (\nu_1,\nu_2)$ is such that the best arm is $a^* = 1$. Consider any alternative bandit model $\nu' = (\nu_1',\nu_2')$ in which $a^* = 2$. Let $\mathcal{E}$ be the event $\mathcal{E} = (\hat{S}_1 = 1)$, which belongs to $\mathcal{F}_\tau$.

Let $\mathcal{A} = ((A_t),\tau,\hat{S}_1)$ be a δ-PAC algorithm: by assumptions, $\mathbb{P}_\nu(\mathcal{E}) \geq 1-\delta$ and $\mathbb{P}_{\nu'}(\mathcal{E}) \leq \delta$. Applying Lemma 1 (with the stopping time τ) and using again the monotonicity properties of $d(x,y)$ and inequality (3)

$$\mathbb{E}_\nu[N_1]\mathrm{KL}(\nu_1,\nu_1') + \mathbb{E}_\nu[N_2]\mathrm{KL}(\nu_2,\nu_2') \geq \log(1/(2.4\delta)). \tag{12}$$

Using moreover that $\tau = N_1 + N_2$, one has

$$\mathbb{E}_\nu[\tau] \geq \frac{\log\left(\frac{1}{2.4\delta}\right)}{\max_{a=1,2}\mathrm{KL}(\nu_a,\nu_a')}. \tag{13}$$

The result follows by optimizing over the possible model ν' satisfying $\mu_1' < \mu_2'$ to make the right hand side of the inequality as large as possible. More precisely, for every $\alpha > 0$, from the definition of $c_*(\nu)$, there exists $\nu_\alpha' = (\nu_1',\nu_2')$ for which

$$\max_{a=1,2}\mathrm{KL}(\nu_a,\nu_a') < c_*(\nu) + \alpha.$$

Inequality (13) for the particular choice $\nu' = \nu_\alpha'$ yields $\mathbb{E}_\nu[\tau] \geq (c_*(\nu)+\alpha)^{-1}\log(1/(2.4\delta))$, and the first statement of Theorem 6 follows by letting α go to zero. In the particular case of exponential bandit models, the alternative model consists in choosing $\nu_1' = \nu_{\theta_*}$ and $\nu_2' = \nu_{\theta_*+\epsilon}$ for some ϵ, as illustrated on Figure 2, so that $\max_{a=1,2}\mathrm{KL}(\nu_a,\nu_a')$ is of order $\mathrm{K}_*(\theta_1,\theta_2)$.

When $\mathcal{A}$ uses uniform sampling, using the fact that $\mathbb{E}_\nu[N_1] = \mathbb{E}[N_2] = \mathbb{E}[\tau]/2$ in Equation (12) similarly gives the second statement of Theorem 6.

4. Matching Algorithms for Two-Armed Bandits

For specific instances of two-armed bandit models, we now present algorithms with performance guarantees that closely match the lower bounds of Theorem 6. For Gaussian bandits with known (and possibly different) variances, we describe in Section 4.1 an algorithm termed α-Elimination that is optimal and thus makes it possible to determine the complexity $\kappa_C(\nu)$. For Bernoulli bandit models, we present in Section 4.2 the SGLRT algorithm that uses uniform sampling and is close to optimal.

4.1 Gaussian Bandit Models

We focus here on the class of two-armed Gaussian bandit models with known variances presented in (8), where σ_1 and σ_2 are fixed. We prove that

$$\kappa_C(\nu) = \frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}$$

by exhibiting a strategy that reaches the performance bound of Theorem 6. This strategy uses non-uniform sampling in case where σ_1 and σ_2 differ. When $\sigma_1 = \sigma_2$, we provide in Theorem 8 an improved stopping rule that is δ-PAC and results in a significant reduction of the expected number of samples used.

The α-Elimination algorithm introduced in this Section can also be used in more general two-armed bandit models, where the distribution ν_a is σ_a^2-subgaussian. This means that the probability distribution ν_a satisfies

$$\forall \lambda \in \mathbb{R}, \quad \mathbb{E}_{X\sim\nu_a}\left[e^{\lambda X}\right] \leq \frac{\lambda^2\sigma_a^2}{2}.$$

This covers in particular the cases of bounded distributions with support in $[0,1]$ (that are 1/4-subgaussian). In these more general cases, the algorithm enjoys the same theoretical properties: it is δ-PAC and its sample complexity is bounded as in Theorem 9 below.

4.1.1 Equal Variances

We start with the simpler case $\sigma_1 = \sigma_2 = \sigma$. Thus, the quantity $I_*(\nu)$ introduced in Theorem 6 coincides with $c_*(\nu)$, which suggests that uniform sampling could be optimal. A uniform sampling strategy equivalently collects paired samples (X_s, Y_s) from both arms. The difference $X_s - Y_s$ is normally distributed with mean $\mu = \mu_1 - \mu_2$ and a δ-PAC algorithm is equivalent to a sequential test of $H_0 : (\mu < 0)$ versus $H_1 : (\mu > 0)$ such that both type I and type II error probabilities are bounded by δ. Robbins (1970) proposes the stopping rule

$$\tau = \inf\left\{ t \in 2\mathbb{N}^* : \Big|\sum_{s=1}^{t/2}(X_s - Y_s)\Big| > \sqrt{2\sigma^2 t\beta(t,\delta)} \right\}, \text{with } \beta(t,\delta) = \frac{t+1}{t}\log\left(\frac{t+1}{2\delta}\right). \quad (14)$$

The recommendation rule chooses the empirically best arm at time τ. This procedure can be seen as an *elimination strategy*, in the sense of Jennison et al. (1982). The authors of this paper derive a lower bound on the sample complexity of any δ-PAC *elimination* strategy

(whereas our lower bound applies to *any* δ-PAC algorithm) which is matched by Robbins' algorithm: the above stopping rule τ satisfies

$$\lim_{\delta\to 0}\frac{\mathbb{E}_\nu[\tau]}{\log(1/\delta)} = \frac{8\sigma^2}{(\mu_1-\mu_2)^2}.$$

This value coincide with the lower bound on $\kappa_C(\nu)$ of Theorem 6 in the case of two-armed Gaussian distributions with similar known variance σ^2. This proves that in this case, Robbins' rule (14) is not only optimal among the class of elimination strategies, but also among the class of δ-PAC algorithm.

Any δ-PAC elimination strategy that uses a threshold function (or *exploration rate*) $\beta(t,\delta)$ smaller than Robbins' also matches our asymptotic lower bound, while stopping earlier than the latter. From a practical point of view, it is therefore interesting to exhibit smaller exploration rates that preserve the δ-PAC property. The failure probability of such an algorithm is upper bounded, for example when $\mu_1 < \mu_2$, by

$$\mathbb{P}_\nu\left(\exists k\in\mathbb{N} : \sum_{s=1}^{k}\frac{X_s - Y_s - (\mu_1-\mu_2)}{\sqrt{2\sigma^2}} > \sqrt{2k\beta(2k,\delta)}\right) = \mathbb{P}\left(\exists k\in\mathbb{N} : S_k > \sqrt{2k\beta(2k,\delta)}\right) \quad (15)$$

where S_k is a sum of k i.i.d. variables of distribution $\mathcal{N}(0,1)$. Robbins (1970) obtains a non-explicit confidence region of risk at most δ by choosing $\beta(2k,\delta) = \log(\log(k)/\delta) + o(\log\log(k))$. The dependency in k is in some sense optimal, because the Law of Iterated Logarithm (LIL) states that $\limsup_{k\to\infty} S_k/\sqrt{2k\log\log(k)} = 1$ almost surely. In this paper, we propose a new deviation inequality for a martingale with sub-Gaussian increments, stated as Lemma 7, that permits to build an explicit confidence region reminiscent of the LIL. A related result was recently derived independently by Jamieson et al. (2014).

Lemma 7 *Let $\zeta(u) = \sum_{k\geq 1} k^{-u}$. Let $X_1, X_2, \ldots$ be independent random variables such that, for all $\lambda \in \mathbb{R}$, $\phi(\lambda) := \log\mathbb{E}[\exp(\lambda X_1)] \leq \lambda^2\sigma^2/2$. For every positive integer t let $S_t = X_1 + \cdots + X_t$. Then, for all $\eta > 1$ and $x \geq \frac{8}{(e-1)^2}$,*

$$\mathbb{P}\left(\exists t\in\mathbb{N}^* : S_t > \sqrt{2\sigma^2 t(x+\eta\log\log(et))}\right) \leq \sqrt{e}\,\zeta\left(\eta\left(1-\frac{1}{2x}\right)\right)\left(\frac{\sqrt{x}}{2\sqrt{2}}+1\right)^\eta \exp(-x).$$

Lemma 7 allows to prove Theorem 8 below, as detailed in Appendix E, where we also provide a proof of Lemma 7.

Theorem 8 *For $\delta \leq 0.1$, with*

$$\beta(t,\delta) = \log(1/\delta) + 3\log\log(1/\delta) + (3/2)\log(\log(et/2)), \quad (16)$$

the elimination strategy is δ-PAC.

We refer to Section 6 for numerical simulations that illustrate the significant savings (in the average number of samples needed to reach a decision) resulting from the use of the less conservative exploration rate allowed by Theorem 8.

4.1.2 Mismatched Variances

In the case where $\sigma_1 \neq \sigma_2$, we rely on the α-Elimination strategy, described in Algorithm 1 below. For $a = 1, 2$, $\hat{\mu}_a(t)$ denotes the empirical mean of the samples gathered from arm a up to time t. The algorithm is based on a non-uniform sampling strategy governed by the parameter $\alpha \in (0, 1)$, that maintains the proportion of draws of arm 1 close to α. At the end of every round t, $N_1(t) = \lceil \alpha t \rceil$, $N_2(t) = t - \lceil \alpha t \rceil$ and $\hat{\mu}_1(t) - \hat{\mu}_2(t) \sim \mathcal{N}\left(\mu_1 - \mu_2, \sigma_t^2(\alpha)\right)$ (where $\sigma_t^2(\alpha)$ is defined at line 6 of Algorithm 1). The sampling schedule used here is thus deterministic.

Algorithm 1 α-Elimination

Require: Exploration function $\beta(t, \delta)$, parameter α.

1: *Initialization*: $\hat{\mu}_1(0) = \hat{\mu}_2(0) = 0$, $\sigma_0^2(\alpha) = 1$, $t = 0$
2: **while** $|\hat{\mu}_1(t) - \hat{\mu}_2(t)| \leq \sqrt{2\sigma_t^2(\alpha)\beta(t, \delta)}$ **do**
3: $t \leftarrow t + 1$.
4: If $\lceil \alpha t \rceil = \lceil \alpha(t-1) \rceil$, $A_t \leftarrow 2$, else $A_t \leftarrow 1$
5: Observe $Z_t \sim \nu_{A_t}$ and compute the empirical means $\hat{\mu}_1(t)$ and $\hat{\mu}_2(t)$
6: Compute $\sigma_t^2(\alpha) = \sigma_1^2/\lceil \alpha t \rceil + \sigma_2^2/(t - \lceil \alpha t \rceil)$
7: **end while**
8: **return** $\underset{a=1,2}{\operatorname{argmax}}\ \hat{\mu}_a(t)$

Theorem 9 shows that an optimal allocation of samples between the two arms consists in maintaining the proportion of draws of arm 1 close to $\sigma_1/(\sigma_1+\sigma_2)$ (which is also the case in the fixed-budget setting, see Section 5.1). Indeed, for $\alpha = \sigma_1/(\sigma_1+\sigma_2)$, the α-elimination algorithm is δ-PAC with a suitable exploration rate and (almost) matches the lower bound on $\mathbb{E}_\nu[\tau]$, at least asymptotically when $\delta \to 0$. Its proof can be found in Appendix D.

Theorem 9 *If $\alpha = \sigma_1/(\sigma_1 + \sigma_2)$, the α-elimination strategy using the exploration rate $\beta(t, \delta) = \log \frac{t}{\delta} + 2 \log\log(6t)$ is δ-PAC on $\mathcal{M}$ and satisfies, for every $\nu \in \mathcal{M}$, for every $\epsilon > 0$,*

$$\mathbb{E}_\nu[\tau] \leq (1+\epsilon)\frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}\log\left(\frac{1}{\delta}\right) + \underset{\delta\to 0}{o_\epsilon}\left(\log\left(\frac{1}{\delta}\right)\right).$$

Remark 10 *When $\sigma_1 = \sigma_2$, 1/2-elimination reduces, up to rounding effects, to the elimination procedure described in Section 4.1.1, for which Theorem 8 suggests an exploration rate of order $\log(\log(t)/\delta)$. As the feasibility of this exploration rate when $\sigma_1 \neq \sigma_2$ is yet to be established, we focus on Gaussian bandits with equal variances in the numerical experiments of Section 6.*

4.2 Bernoulli Bandit Models

We consider in this section the class of Bernoulli bandit models

$$\mathcal{M} = \{\nu = (\mathcal{B}(\mu_1), \mathcal{B}(\mu_2)) : (\mu_1, \mu_2) \in (0; 1)^2, \mu_1 \neq \mu_2\},$$

where each arm can be alternatively parameterized by the natural parameter of the exponential family, $\theta_a = \log(\mu_a/(1-\mu_a))$. Observing that in this particular case little can

be gained by departing from uniform sampling, we consider the SGLRT algorithm (to be defined below) that uses uniform sampling together with a stopping rule that is not based on the mere difference of the empirical means.

For Bernoulli bandit models, the quantities $I_*(\nu)$ and $c_*(\nu)$ introduced in Theorem 6 happen to be practically very close (see Figure 3 in Section 5 below). There is thus a strong incentive to use uniform sampling and in the rest of this section we consider algorithms that aim at matching the bound (7) of Theorem 6—that is, $\mathbb{E}_\nu[\tau] \leq \log(1/\delta)/I_*(\nu)$, at least for small values of δ—, which provides an upper bound on $\kappa_C(\nu)$ that is very close to $1/c_*(\nu)$. For simplicity, as $I_*(\nu)$ is here a function of the means of the arms only, we will denote $I_*(\nu)$ by $I_*(\mu_1, \mu_2)$.

When the arms are sampled uniformly, finding an algorithm that matches the bound of (7) boils down to determining a proper stopping rule. In all the algorithms studied so far, the stopping rule was based on the difference of the empirical means of the arms. For Bernoulli arms the 1/2-Elimination procedure described in Algorithm 1 can be used, as each distribution ν_a is bounded and therefore 1/4-subgaussian. More precisely, with $\beta(t, \delta)$ as in Theorem 8, the algorithm stopping at the first time t such that

$$\hat{\mu}_1(t) - \hat{\mu}_2(t) > \sqrt{2\beta(t,\delta)/t}$$

has its sample complexity bounded by $2/(\mu_1 - \mu_2)^2 \log(1/\delta) + o(\log(1/\delta))$. Yet, Pinsker's inequality implies that $I_*(\mu_1, \mu_2) > (\mu_1 - \mu_2)^2/2$ and this algorithm is thus not optimal with respect to the bound (7) of Theorem 6. The approximation $I_*(\mu_1, \mu_2) = (\mu_1 - \mu_2)^2/(8\mu_1(1-\mu_1)) + o\left((\mu_1 - \mu_2)^2\right)$ suggests that the loss with respect to the optimal error exponent is particularly significant when both means are close to 0 or 1.

To circumvent this drawback, we propose the SGLRT (for Sequential Generalized Likelihood Ratio Test) stopping rule, described in Algorithm 2. The appearance of I_* in the stopping criterion of Algorithm 2 is a consequence of the observation that it is related to the generalized likelihood ratio statistic for testing the equality of two Bernoulli proportions. To test $H_0 : (\mu_1 = \mu_2)$ against $H_1 : (\mu_1 \neq \mu_2)$ based on $t/2$ paired samples of the arms $W_s = (X_s, Y_s)$, the Generalized Likelihood Ratio Test (GLRT) rejects H_0 when

$$\frac{\max_{\mu_1,\mu_2:\mu_1=\mu_2} L(W_1, \ldots, W_{t/2}; \mu_1, \mu_2)}{\max_{\mu_1,\mu_2} L(W_1, \ldots, W_{t/2}; \mu_1, \mu_2)} < z_\delta,$$

where $L(W_1, \ldots, W_{t/2}; \mu_1, \mu_2)$ denotes the likelihood of the observations given parameters μ_1 and μ_2. It can be checked that the ratio that appears in the last display is equal to

Algorithm 2 Sequential Generalized Likelihood Ratio Test (SGLRT)

Require: Exploration function $\beta(t, \delta)$.

1: *Initialization*: $\hat{\mu}_1(0) = \hat{\mu}_2(0) = 0$. $t = 0$.
2: **while** $(tI_*(\hat{\mu}_1(t), \hat{\mu}_2(t)) \leq \beta(t, \delta)) \bigcup (t = 1\ (mod.\ 2))$ **do**
3: $t = t + 1$. $A_t = t\ (mod.\ 2)$.
4: Observe $Z_t \sim \nu_{A_t}$ and compute the empirical means $\hat{\mu}_1(t)$ and $\hat{\mu}_2(t)$.
5: **end while**
6: **return** $a = \underset{a=1,2}{\operatorname{argmax}}\ \hat{\mu}_a(t)$.

$\exp(-tI_*(\hat{\mu}_{1,t/2}, \hat{\mu}_{2,t/2}))$. This equality is a consequence of the rewriting

$$I_*(x,y) \quad = \quad H\left(\frac{x+y}{2}\right) - \frac{1}{2}\left[H(x) + H(y)\right],$$

where $H(x) = -x\log(x) - (1-x)\log(1-x)$ denotes the binary entropy function. Hence, Algorithm (2) can be interpreted as a sequential version of the GLRT with (varying) threshold $z_{t,\delta} = \exp(-\beta(t,\delta))$.

Elements of analysis of the SGLRT. The SGLRT algorithm is also related to the KL-LUCB algorithm of Kaufmann and Kalyanakrishnan (2013). A closer examination of the KL-LUCB stopping criterion reveals that, in the specific case of two-armed bandits, it is equivalent to stopping when $t\mathrm{KL}_*(\mathcal{B}(\hat{\mu}_1(t)), \mathcal{B}(\hat{\mu}_2(t)))$ gets larger than some threshold. We also mentioned the fact that $\mathrm{KL}_*(\mathcal{B}(x), \mathcal{B}(y))$ and $I_*(x,y)$ are very close (see Figure 3). Using results from Kaufmann and Kalyanakrishnan (2013), one can thus prove (see Appendix F) the following lemma.

Lemma 11 *With the exploration rate*

$$\beta(t,\delta) = 2\log\left(\frac{t(\log(3t))^2}{\delta}\right)$$

the SGLRT algorithm is δ-PAC.

For this exploration rate, we were able to obtain the following asymptotic guarantee on the stopping time τ of Algorithm 2:

$$\forall \epsilon > 0, \quad \limsup_{\delta \to 0} \frac{\tau}{\log(1/\delta)} \leq \frac{2(1+\epsilon)}{I_*(\mu_1,\mu_2)} \quad a.s.$$

(see Lemma 26 in Appendix F for the proof of this result). By analogy with the result of Theorem 8 we conjecture that the analysis of Kaufmann and Kalyanakrishnan (2013)—on which the result of Lemma 11 is based—is too conservative and that the use of an exploration rate of order $\log(\log(t)/\delta)$ should also lead to a δ-PAC algorithm. This conjecture is supported by the numerical experiments reported in Section 6 below. Besides, for this choice of exploration rate, Lemma 26 also shows that

$$\forall \epsilon > 0, \quad \limsup_{\delta \to 0} \frac{\tau}{\log(1/\delta)} \leq \frac{(1+\epsilon)}{I_*(\mu_1,\mu_2)} \quad a.s..$$

5. The Fixed-Budget Setting

In this section, we focus on the fixed-budget setting and we provide new upper and lower bounds on the complexity term $\kappa_B(\nu)$.

For two-armed bandits, we obtain in Theorem 12 lower bounds analogous to those of Theorem 6 in the fixed-confidence setting. We present matching algorithms for Gaussian and Bernoulli bandits. This allows for a comparison between the fixed-budget and fixed-confidence setting in these specific cases. More specifically, we show that $\kappa_B(\nu) = \kappa_C(\nu)$ for Gaussian bandit models, whereas $\kappa_C(\nu) > \kappa_B(\nu)$ for Bernoulli bandit models.

When $K > 2$ and $m \geq 1$, we present a first step towards obtaining more general results, by providing lower bounds on the probability of error $p_t(\nu)$ for Gaussian bandits with equal variances.

5.1 Comparison of the Complexities for Two-Armed Bandits

We present here an asymptotic lower bound on $p_t(\nu)$ that directly yields a lower bound on $\kappa_B(\nu)$. Moreover, we provide a lower bound on the failure probability of consistent algorithms using uniform sampling. The proof of Theorem 12 bears similarities with that of Theorem 6, and we provide it in Appendix G.1. However, it is important to note that the informational quantities $c^*(\nu)$ and $I^*(\nu)$ defined in Theorem 12 are in general different from the quantities $c_*(\nu)$ and $I_*(\nu)$ previously defined for the fixed-confidence setting (see Theorem 6). Appendix C contains a few additional elements of comparison between these quantities in the case of one-parameter exponential families of distributions.

Theorem 12 *Let $\nu = (\nu_1, \nu_2)$ be a two-armed bandit model such that $\mu_1 > \mu_2$. In the fixed-budget setting, any consistent algorithm satisfies*

$$\limsup_{t\to\infty} -\frac{1}{t}\log p_t(\nu) \leq c^*(\nu), \quad \textit{where} \quad c^*(\nu) := \inf_{(\nu_1',\nu_2')\in\mathcal{M}:\mu_1'<\mu_2'} \max\left\{\mathrm{KL}(\nu_1',\nu_1), \mathrm{KL}(\nu_2',\nu_2)\right\}.$$

Moreover, any consistent algorithm using a uniform sampling strategy satisfies

$$\limsup_{t\to\infty} -\frac{1}{t}\log p_t(\nu) \leq I^*(\nu), \quad \textit{where} \quad I^*(\nu) := \inf_{(\nu_1',\nu_2')\in\mathcal{M}:\mu_1'<\mu_2'} \frac{\mathrm{KL}(\nu_1',\nu_1) + \mathrm{KL}(\nu_2',\nu_2)}{2}. \tag{17}$$

Gaussian distributions. As the Kullback-Leibler divergence between two Gaussian distributions—(9)—is symmetric with respect to the means when the variances are held fixed, it holds that $c^*(\nu) = c_*(\nu)$. To find a matching algorithm, we introduce the simple family of *static strategies* that draw n_1 samples from arm 1 followed by $n_2 = t - n_1$ samples of arm 2, and then choose arm 1 if $\hat{\mu}_{1,n_1} > \hat{\mu}_{2,n_2}$, where $\hat{\mu}_{i,n_i}$ denotes the empirical mean of the n_i samples from arm i. Assume for instance that $\mu_1 > \mu_2$. Since $\hat{\mu}_{1,n_1} - \hat{\mu}_{2,n_2} - \mu_1 + \mu_2 \sim \mathcal{N}\left(0, \sigma_1^2/n_1 + \sigma_2^2/n_2\right)$, the probability of error of such a strategy is upper bounded by

$$\mathbb{P}\left(\hat{\mu}_{1,n_1} < \hat{\mu}_{2,n_2}\right) \quad \leq \quad \exp\left(-\left(\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}\right)^{-1} \frac{(\mu_1-\mu_2)^2}{2}\right).$$

The right hand side is minimized when $n_1/(n_1+n_2) = \sigma_1/(\sigma_1+\sigma_2)$, and the static strategy drawing $n_1 = \lceil \sigma_1 t/(\sigma_1+\sigma_2)\rceil$ times arm 1 is such that

$$\liminf_{t\to\infty} -\frac{1}{t}\log p_t(\nu) \geq \frac{(\mu_1-\mu_2)^2}{2(\sigma_1+\sigma_2)^2} = c^*(\nu)\,.$$

This shows in particular that for Gaussian distributions the two complexities are equal:

$$\kappa_B(\nu) = \kappa_C(\nu) = \frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}.$$

Exponential families. For exponential family bandit models, it can be observed that

$$c^*(\nu) = \inf_{\theta\in\Theta} \max\left(\mathrm{K}(\theta,\theta_1), \mathrm{K}(\theta,\theta_2)\right) = \mathrm{K}^*(\theta_1,\theta_2),$$

where $\mathrm{K}^*(\theta_1,\theta_2)$ is the Chernoff information between the distributions ν_{θ_1} and ν_{θ_2}. We recall that $\mathrm{K}^*(\theta_1,\theta_2) = \mathrm{K}(\theta^*,\theta_1)$, where θ^* is defined by $\mathrm{K}(\theta^*,\theta_1) = \mathrm{K}(\theta^*,\theta_2)$. Moreover, one has

$$I^*(\nu) = \frac{\mathrm{K}\left(\frac{\theta_1+\theta_2}{2},\theta_1\right) + \mathrm{K}\left(\frac{\theta_1+\theta_2}{2},\theta_2\right)}{2}.$$

In particular, the quantity $c^*(\nu) = \mathrm{K}^*(\theta_1,\theta_2)$ does not always coincide with the quantity $c_*(\nu) = \mathrm{K}_*(\theta_1,\theta_2)$ defined in Theorem 6. More precisely, $c_*(\nu)$ and $c^*(\nu)$ are equal when the log-partition function $b(\theta)$ is (Fenchel) self-conjugate, which is the case for Gaussian and exponential variables (see Appendix C). However, for Bernoulli distributions, it can be checked that $c^*(\nu) > c_*(\nu)$. By exhibiting a matching strategy in the fixed-budget setting (Theorem 13), we show that this implies that $\kappa_C(\nu) > \kappa_B(\nu)$ in the Bernoulli case (Theorem 14). We also show that in this case, only little can be gained by departing from uniform sampling.

Theorem 13 *Consider a two-armed exponential bandit model and $\alpha(\theta_1,\theta_2)$ be defined by*

$$\alpha(\theta_1,\theta_2) = \frac{\theta^*-\theta_1}{\theta_2-\theta_1} \quad \textit{where} \quad \mathrm{K}(\theta^*,\theta_1) = \mathrm{K}(\theta^*,\theta_2).$$

For all t, the static strategy that allocates $\lceil \alpha(\theta_1,\theta_2)t \rceil$ samples to arm 1, and recommends the empirical best arm, satisfies $p_t(\nu) \leq \exp(-tK^*(\theta_1,\theta_2))$.

Theorem 13, whose proof can be found in Appendix G.2, shows in particular that for every exponential family bandit model there exists a consistent static strategy such that

$$\liminf_{t\to\infty} -\frac{1}{t}\log p_t \geq \mathrm{K}^*(\theta_1,\theta_2), \quad \text{and hence that} \quad \kappa_B(\nu) = \frac{1}{\mathrm{K}^*(\theta_1,\theta_2)}.$$

By combining this observation with Theorem 6 and the fact that, $\mathrm{K}_*(\theta_1,\theta_2) < \mathrm{K}^*(\theta_1,\theta_2)$ for Bernoulli distributions, one obtains the following inequality.

Theorem 14 *For two-armed Bernoulli bandit models, $\kappa_C(\nu) > \kappa_B(\nu)$.*

Note that we have determined the complexity of the fixed-budget setting by exhibiting an algorithm (leading to an upper bound on κ_B) that is of limited practical interest for Bernoulli bandit models. Indeed, the optimal static strategy defined in Theorem 13 requires the knowledge of the quantity $\alpha(\theta_1,\theta_2)$, that depends on the unknown means of the arms. So far, it is not known whether there exists a *universal* strategy, that would satisfy $p_t(\nu) \leq \exp(-\mathrm{K}^*(\theta_1,\theta_2)t)$ on *every* Bernoulli bandit model.

However, Lemma 27 shows that the strategy that uses uniform sampling and recommends the empirical best-arm satisfies $p_t(\nu) \leq \exp(-I^*(\nu)t)$, and matches the bound (17) of Theorem 12 (see Remark 28 in Appendix G.2). The fact that, just as in the fixed-confidence setting $I^*(\nu)$ is very close to $c^*(\nu)$ shows that the problem-dependent optimal strategy described above can be approximated by a very simple, universal algorithm that samples the arms uniformly. Figure 3 represents the different informational functions c_*, I_*, c^* and I^* when the mean μ_1 varies, for two fixed values of μ_2. It can be observed that $c^*(\nu)$ and $c_*(\nu)$ are almost indistinguishable from $I^*(\nu)$ and $I_*(\nu)$, respectively, while there is a gap between $c^*(\nu)$ and $c_*(\nu)$.

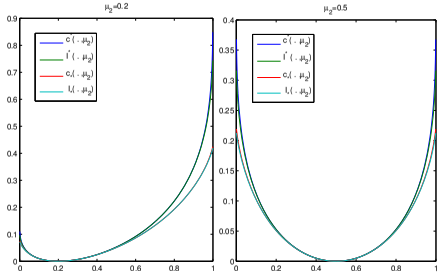


Figure 3: Comparison of different informational quantities for Bernoulli bandit models.

5.2 Lower Bound on $p_t(\nu)$ in More General Cases

Theorem 12 provides a direct counterpart to Theorem 6, allowing for a complete comparison between the fixed confidence and fixed budget settings in the case of two-armed bandits. However, we were not able to obtain a general lower bound for K-armed bandit that would be directly comparable to that of Theorem 4 in the fixed budget setting. Using Lemma 15 stated below (a variant of Lemma 1 proved in Appendix A.2), we were nonetheless able to derive tighter, non-asymptotic, lower bounds on $p_t(\nu)$ in the particular case of Gaussian bandit models with equal known variance, $\mathcal{M}_m = \{\nu = (\nu_1,\dots,\nu_K) : \nu_a = \mathcal{N}\left(\mu_a,\sigma^2\right), \mu_a \in \mathbb{R}, \mu_{[m]} \neq \mu_{[m+1]}\}$.

Lemma 15 *Let ν and ν' be two bandit models such that $\mathcal{S}^*_m(\nu) \neq \mathcal{S}^*_m(\nu')$. Then*

$$\max\left(\mathbb{P}_\nu(\mathcal{S} \neq \mathcal{S}^*_m(\nu)), \mathbb{P}_{\nu'}(\mathcal{S} \neq \mathcal{S}^*_m(\nu'))\right) \geq \frac{1}{4}\exp\left(-\sum_{a=1}^K \mathbb{E}_\nu[N_a]\mathrm{KL}(\nu_a,\nu'_a)\right).$$

Theorem 16 *Let ν be a Gaussian bandit model such that $\mu_1 > \mu_2 \geq \cdots \geq \mu_K$ and let*

$$H'(\nu) = \sum_{a=2}^K \frac{2\sigma^2}{(\mu_1-\mu_a)^2}.$$

There exists a bandit model $\nu^{[a]}$, $a \in \{2,\dots,K\}$, (see Figure 4) which satisfies $H'(\nu^{[a]}) \leq H'(\nu)$ and is such that

$$\max\left(p_t(\nu), p_t(\nu^{[a]})\right) \geq \exp\left(-\frac{4t}{H'(\nu)}\right).$$

This result is to be compared to the lower bound of Audibert et al. (2010). While Theorem 16 does not really provide a lower bound on $\kappa_B(\nu)$, the complexity term $H(\nu)$ is close to the quantity that appears in Theorem 4 for the fixed-confidence setting (in the Gaussian case), which improves over the term $H_2(\nu) = \max_{i:\mu_{[i]}<\mu_{[1]}} i(\mu_{[1]}-\mu_{[i]})^{-2}$ featured in Theorem 4 of Audibert et al. (2010).

For $m > 1$, building on the same ideas, Theorem 17 provides a first lower bound, which we believe leaves room for improvement.

Theorem 17 *Let ν be such that $\mu_1 > \dots \mu_m > \mu_{m+1} > \dots > \mu_K$ and let*

$$H^+(\nu) = \sum_{a=1}^{m} \frac{2\sigma^2}{(\mu_a - \mu_{m+1})^2}, \quad H^-(\nu) = \sum_{a=m+1}^{K} \frac{2\sigma^2}{(\mu_m - \mu_a)^2}, \quad \text{and} \quad H(\nu) = H^+(\nu) + H^-(\nu).$$

There exists $a \in \{1, \dots, m\}$ and $b \in \{m+1, \dots K\}$ such that the bandit model $\nu^{[a,b]}$ described on Figure 4 satisfies $H(\nu^{[a,b]}) < H(\nu)$ and is such that

$$\max\left(p_t(\nu), p_t(\nu^{[a,b]})\right) \geq \frac{1}{4}\exp\left(-\frac{4t}{\tilde{H}(\nu)}\right), \quad \text{where} \quad \tilde{H}(\nu) = \frac{H(\nu)\min(H^+(\nu), H^-(\nu))}{H(\nu) + \min(H^+(\nu), H^-(\nu))}.$$

The proofs of Theorem 16 and Theorem 17 are very similar. For this reason, we provide in Appendix G.3 only the latter. Introducing the gaps Δ_a defined in (2), the precise definition of the modified problems $\nu^{[a]}$ and $\nu^{[a,b]}$ in the statement of the two results is:

$$\nu^{[a]}: \left\{ \begin{array}{rcl} \mu'_k &=& \mu_k \text{ for all } k \neq a \\ \mu'_a &=& \mu_a + 2\Delta_a \end{array} \right. \quad \text{and} \quad \nu^{[a,b]}: \left\{ \begin{array}{rcl} \mu'_k &=& \mu_k \text{ for all } k \notin \{a,b\} \\ \mu'_a &=& \mu_a - 2\Delta_b \\ \mu'_b &=& \mu_b + 2\Delta_a \end{array} \right. .$$

6. Numerical Experiments

In this section, we focus on two-armed models and provide experimental experiments designed to compare the fixed-budget and fixed-confidence settings (in the Gaussian and Bernoulli cases) and to illustrate the improvement resulting from the adoption of the reduced exploration rate of Theorem 8.

In Figure 5, we consider two Gaussian bandit models with known common variance: the 'easy' one is $\{\mathcal{N}(0.5, 0.25), \mathcal{N}(0, 0.25)\}$, corresponding to $\kappa_C = \kappa_B = \kappa = 8$, on the left; and the 'difficult' one is $\{\mathcal{N}(0.01, 0.25), \mathcal{N}(0, 0.25)\}$, that is $\kappa = 2\times 10^4$, on the right. In the fixed-budget setting, stars ('*') report the probability of error $p_n(\nu)$ as a function of n. In the fixed-confidence setting, we plot both the empirical probability of error by circles ('O') and the specified maximal error probability δ by crosses ('X') as a function of the empirical average of the running times. Note the logarithmic scale used for the probabilities on the y-axis. All results are averaged over $N = 10^6$ independent Monte Carlo replications. For comparison purposes, a plain line represents the theoretical rate $t \mapsto \exp(-t(1/\kappa))$ which is a straight line on the log scale.

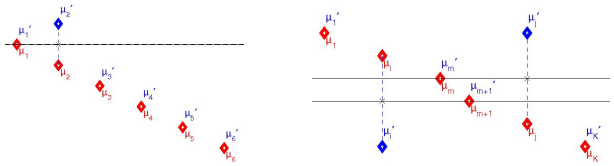


Figure 4: Left: bandit models ν, in red, and $\nu^{[2]}$, in blue (Theorem 16). Right: bandit models ν, in red, and $\nu^{[i,j]}$, in blue (Theorem 17).

In the fixed-confidence setting, we report results for elimination algorithms of the form (14) for three different exploration rates $\beta(t,\delta)$. The exploration rate we consider are: the provably-PAC rate of Robbins' algorithm $\log(t/\delta)$ (large blue symbols), the conjectured optimal exploration rate $\log((\log(t)+1)/\delta)$, almost provably δ-PAC according to Theorem 8 (bold green symbols), and the rate $\log(1/\delta)$, which would be appropriate if we were to perform the stopping test only at a single pre-specified time (orange symbols). For each algorithm, the log probability of error is approximately a linear function of the number of samples, with a slope close to $-1/\kappa$, where κ is the complexity. A first observation is that the 'traditional' rate of $\log(t/\delta)$ is much too conservative, with running times for the difficult problem (right plot) which are about three times longer than those of other methods for comparable error rates. As expected, the rate $\log((\log(t)+1)/\delta)$ significantly reduces the running times while maintaining proper control of the probability of failure, with empirical error rates ('O' symbols) below the corresponding confidence parameters δ (represented by 'X' symbols). Conversely, the use of the non-sequential testing threshold $\log(1/\delta)$ seems too risky, as one can observe that the empirical probability of error may be larger than δ on difficult problems. To illustrate the gain in sample complexity resulting from the knowledge of the means, we also represented in red the performance of the SPRT algorithm mentioned in the introduction of Section 5 along with the theoretical relation between the probability of error and the expected number of samples, materialized as a dashed line. The SPRT stops for t such that $|(\mu_1 - \mu_2)(S_{1,t/2} - S_{2,t/2})| > \log(1/\delta)$.

Robbins' algorithm is δ-PAC and matches the complexity (which is illustrated by the slope of the measures), though in practice the use of the exploration rate $\log((\log(t)+1)/\delta)$ leads to huge gain in terms of number of samples used. It is important to keep in mind that running times play the same role as error exponents and hence the threefold increase of average running times observed on the rightmost plot of Figure 5 when using $\beta(t,\delta) = \log(t/\delta)$ is really prohibitive.

On Figure 6, we compare on two Bernoulli bandit models the performance of the SGLRT algorithm described in Section 4.2 (Algorithm 2) using two different exploration rates, $\log(1/\delta)$ and $\log((\log(t)+1)/\delta)$, to the 1/2-elimination stopping rule (Algorithm 1) that stops when the difference of empirical means exceeds the threshold $\sqrt{2\beta(t,\delta)/t}$ (for the

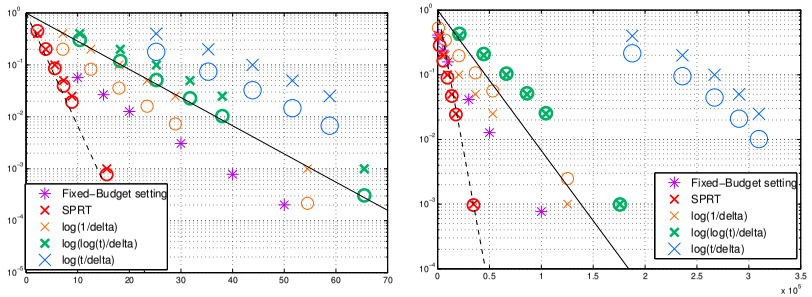


Figure 5: Experimental results for Gaussian bandit models

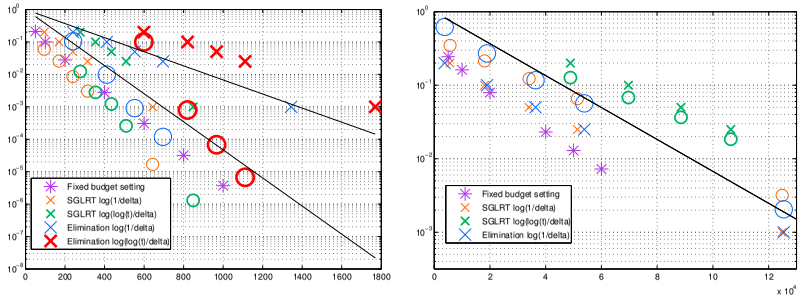


Figure 6: Results for Bernoulli bandit models: $0.2 - 0.1$ (left) and $0.51 - 0.5$ (right).

same exploration rates). Plain lines also materialize the theoretical optimal rate $t \mapsto \exp(-t/\kappa_C(\nu))$ and the rate attained by the 1/2-Elimination algorithm $t \mapsto \exp(-t/\kappa')$, where $\kappa' = 2/(\mu_1 - \mu_2)^2$. On the bandit model $0.51 - 0.5$ (right) these two rates are very close and SGLRT mostly coincides with Elimination, but on the bandit model $0.2 - 0.1$ (left) the practical gain of the use of a more sophisticated stopping strategy is well illustrated. Besides, our experiments show that SGLRT using $\log((\log(t)+1)/\delta)$ is δ-PAC on both the (relatively) easy and difficult problems we consider, unlike the other algorithms considered.

If one compares the results for the fixed-budget setting (in purple) to those for the best δ-PAC algorithm (or conjectured δ-PAC for SGLRT in the Bernoulli case), in green, one can observe that to obtain the same probability of error, the fixed-confidence algorithm usually needs an average number of samples that is about twice larger than the deterministic number of samples required by the fixed-budget setting algorithm. This remark should be related to the fact that a δ-PAC algorithm is designed to be uniformly good across all problems, whereas consistency is a weak requirement in the fixed-budget setting: any strategy that draws both arm infinitely often and recommends the empirical best is consistent. Figure 5 also shows that when the values of μ_1 and μ_2 are unknown, the sequential version of the test is no more preferable to its batch counterpart and can even become much worse if the exploration rate $\beta(t,\delta)$ is chosen too conservatively. This observation should be mitigated by the fact that the sequential (or fixed-confidence) approach is adaptive with respect to the difficulty of the problem whereas it is impossible to predict the efficiency of a batch (or fixed-budget) experiment without some prior knowledge regarding the difficulty of the problem under consideration.

7. Conclusion

Our aim with this paper has been to provide a framework for evaluating, in a principled way, the performance of fixed-confidence and fixed-budget algorithms designed to identify the best arm(s) in stochastic environments.

For two-armed bandits, we obtained rather complete results, identifying the complexity of both settings in important parametric families of distributions. In doing so, we observed that standard testing strategies based on uniform sampling are optimal or close to optimal for Gaussian distributions with matched variance or Bernoulli distributions but can be

improved (by non-uniform sampling) for Gaussian distributions with distinct variances. This latter observation can certainly be generalized to other models, starting with the case of Gaussian distributions whose variances are a priori unknown. In the case of Bernoulli distributions, we have also shown that fixed-confidence algorithms that use the difference of the empirical means as a stopping criterion are bound to be sub-optimal. Finally, we have shown, through the comparison of the complexities $\kappa_C(\nu)$ and $\kappa_B(\nu)$, that the behavior observed when testing fully specified alternatives where fixed confidence (or sequential) algorithms may be 'faster on average' than the fixed budget (or batch) ones is not true anymore when the parameters of the models are unknown.

For models with more than two arms, we obtained the first generic (i.e. not based on the sub-Gaussian tail assumption) distribution-dependent lower bound on the complexity of m best-arms identification in the fixed-confidence setting (Theorem 4). Currently available performance bounds for algorithms performing m best-arms identification—those of Kaufmann and Kalyanakrishnan (2013) notably—show a small gap with this result and it is certainly of interest to investigate whether those analyses and/or the bound of Theorem 4 may be improved to bridge the gap. For the fixed-budget setting we made only a small step towards the understanding of the complexity of m best-arms identification and our results can certainly be greatly improved.

Acknowledgments

We thank Sébastien Bubeck for fruitful discussions during the visit of the first author at Princeton University. This work has been supported by the ANR-2010-COSI-002 and ANR-13-BS01-0005 grants of the French National Research Agency.

Appendix A. Changes of Distributions

Let ν and ν' be two bandit models such that for all $a \in \{1, K\}$ the distributions ν_a and ν'_a are mutually absolutely continuous. For each a, there exists a measure λ_a such that ν_a and ν'_a have a density f_a and f'_a respectively with respect to λ_a. One can introduce the log-likelihood ratio of the observations up to time t under an algorithm $\mathcal{A}$:

$$L_t = L_t(A_1, \ldots, A_t, Z_1, \ldots, Z_t) := \sum_{a=1}^{K} \sum_{s=1}^{t} \mathbb{1}_{(A_s=a)} \log\left(\frac{f_a(Z_s)}{f'_a(Z_s)}\right).$$

The key element in a change of distribution is the following classical lemma that relates the probabilities of an event under $\mathbb{P}_\nu$ and $\mathbb{P}_{\nu'}$ through the log-likelihood ratio of the observations. Such a result has often been used in the bandit literature for ν and ν' that differ just from one arm, for which the expression of the log-likelihood ratio is simpler. In this paper, we consider more general changes of distributions, and we therefore provide a full proof of Lemma 18 in Appendix A.3.

Lemma 18 *Let σ be any stopping time with respect to $\mathcal{F}_t$. For every event $\mathcal{E} \in \mathcal{F}_\sigma$ (i.e., $\mathcal{E}$ such that $\mathcal{E} \cap (\sigma = t) \in \mathcal{F}_t$),*

$$\mathbb{P}_{\nu'}(\mathcal{E}) = \mathbb{E}_\nu[\mathbb{1}_\mathcal{E} \exp(-L_\sigma)]$$

A.1 Proof of Lemma 1

To prove Lemma 1, we state a first inequality on the expected log-likelihood ratio in Lemma 19, which is of independent interest.

Lemma 19 *Let σ be any almost surely finite stopping time with respect to $\mathcal{F}_t$. For every event $\mathcal{E} \in \mathcal{F}_\sigma$,*

$$\mathbb{E}_\nu[L_\sigma] \geq d(\mathbb{P}_\nu(\mathcal{E}), \mathbb{P}_{\nu'}(\mathcal{E})).$$

Lemma 1 easily follows: introducing $(Y_{a,s})$, the sequence of i.i.d. samples successively observed from arm a, the log-likelihood ratio L_t can be rewritten

$$L_t = \sum_{a=1}^{K} \sum_{s=1}^{N_a(t)} \log\left(\frac{f_a(Y_{a,s})}{f'_a(Y_{a,s})}\right); \quad \text{and} \quad \mathbb{E}_\nu\left[\log\left(\frac{f_a(Y_{a,s})}{f'_a(Y_{a,s})}\right)\right] = \mathrm{KL}(\nu_a, \nu'_a).$$

Wald's Lemma (see e.g., Siegmund (1985)) applied to $L_\sigma = \sum_{a=1}^{K} \sum_{s=1}^{N_a(\sigma)} \log\left(\frac{f_a(Y_{a,s})}{f'_a(Y_{a,s})}\right)$ yields

$$\mathbb{E}_\nu[L_\sigma] = \sum_{a=1}^{K} \mathbb{E}_\nu[N_a(\sigma)]\mathrm{KL}(\nu_a, \nu'_a). \tag{18}$$

Combining this equality with the inequality in Lemma 19 completes the proof.

Proof of Lemma 19. Let σ be a stopping time with respect to $(\mathcal{F}_t)$.

We start by showing that for all $\mathcal{E} \in \mathcal{F}_\sigma$, $\mathbb{P}_\nu(\mathcal{E}) = 0 \Leftrightarrow \mathbb{P}_{\nu'}(\mathcal{E}) = 0$. This proves Lemma 19 for events $\mathcal{E}$ such that $\mathbb{P}_\nu(\mathcal{E}) = 0$ or 1, for which the quantity $d(\mathbb{P}_\nu(\mathcal{E}), \mathbb{P}_{\nu'}(\mathcal{E})) = d(0,0)$ or $d(1,1)$ is equal to zero by convention, and the inequality thus holds since the left-hand side is non-negative (which is clear from the rewriting (18)). Let $\mathcal{E} \in \mathcal{F}_\sigma$. Lemma 18 yields $\mathbb{P}_{\nu'}(\mathcal{E}) = \mathbb{E}_\nu[\mathbb{1}_\mathcal{E} \exp(-L_\sigma)]$. Thus $\mathbb{P}_{\nu'}(\mathcal{E}) = 0$ implies $\mathbb{1}_\mathcal{E} \exp(-L_\sigma) = 0$ $\mathbb{P}_\nu - a.s.$ As $\mathbb{P}_\nu(\sigma < +\infty) = 1$, $\mathbb{P}_\nu(\exp(L_\sigma) > 0) = 1$ and $\mathbb{P}_{\nu'}(\mathcal{E}) = 0 \Rightarrow \mathbb{P}_\nu(\mathcal{E}) = 0$. A similar reasoning yields $\mathbb{P}_\nu(\mathcal{E}) = 0 \Rightarrow \mathbb{P}_{\nu'}(\mathcal{E}) = 0$.

Let $\mathcal{E} \in \mathcal{F}_\sigma$ be such that $0 < \mathbb{P}_\nu(\mathcal{E}) < 1$ (then $0 < \mathbb{P}_{\nu'}(\mathcal{E}) < 1$). Lemma 18 and the conditional Jensen inequality lead to

$$\begin{aligned}
\mathbb{P}_{\nu'}(\mathcal{E}) &= \mathbb{E}_\nu[\exp(-L_\sigma)\mathbb{1}_\mathcal{E}] = \mathbb{E}_\nu[\mathbb{E}_\nu[\exp(-L_\sigma)|\mathbb{1}_\mathcal{E}]\mathbb{1}_\mathcal{E}] \\
&\geq \mathbb{E}_\nu[\exp\left(-\mathbb{E}_\nu[L_\sigma|\mathbb{1}_\mathcal{E}]\right)\mathbb{1}_\mathcal{E}] = \mathbb{E}_\nu[\exp\left(-\mathbb{E}_\nu[L_\sigma|\mathbb{1}_\mathcal{E}]\mathbb{1}_\mathcal{E}\right)\mathbb{1}_\mathcal{E}] \\
&= \mathbb{E}_\nu[\exp\left(-\mathbb{E}_\nu[L_\sigma|\mathcal{E}]\mathbb{1}_\mathcal{E}\right)\mathbb{1}_\mathcal{E}] = \mathbb{E}_\nu[\exp\left(-\mathbb{E}_\nu[L_\sigma|\mathcal{E}]\right)\mathbb{1}_\mathcal{E}] \\
&= \exp\left(-\mathbb{E}_\nu[L_\sigma|\mathcal{E}]\right)\mathbb{P}_\nu(\mathcal{E}).
\end{aligned}$$

Writing the same for the event $\overline{\mathcal{E}}$ yields $\mathbb{P}_{\nu'}(\overline{\mathcal{E}}) \geq \exp\left(-\mathbb{E}_\nu[L_\sigma|\overline{\mathcal{E}}]\right)\mathbb{P}_\nu(\overline{\mathcal{E}})$, hence

$$\mathbb{E}_\nu[L_\sigma|\mathcal{E}] \geq \log\frac{\mathbb{P}_\nu(\mathcal{E})}{\mathbb{P}_{\nu'}(\mathcal{E})} \quad \text{and} \quad \mathbb{E}_\nu[L_\sigma|\overline{\mathcal{E}}] \geq \log\frac{\mathbb{P}_\nu(\overline{\mathcal{E}})}{\mathbb{P}_{\nu'}(\overline{\mathcal{E}})}. \tag{19}$$

Therefore one can write

$$\begin{aligned}
\mathbb{E}_\nu[L_\sigma] &= \mathbb{E}_\nu[L_\sigma|\mathcal{E}]\mathbb{P}_\nu(\mathcal{E}) + \mathbb{E}_\nu[L_\sigma|\overline{\mathcal{E}}]\mathbb{P}_\nu(\overline{\mathcal{E}}) \\
&\geq \mathbb{P}_\nu(\mathcal{E})\log\frac{\mathbb{P}_\nu(\mathcal{E})}{\mathbb{P}_{\nu'}(\mathcal{E})} + \mathbb{P}_\nu(\overline{\mathcal{E}})\log\frac{\mathbb{P}_\nu(\overline{\mathcal{E}})}{\mathbb{P}_{\nu'}(\overline{\mathcal{E}})} = d(\mathbb{P}_\nu(\mathcal{E}), \mathbb{P}_{\nu'}(\mathcal{E})),
\end{aligned}$$

which concludes the proof.

A.2 Proof of Lemma 15

The proof bears strong similarities with that of Lemma 1, but an extra ingredient is needed: Lemma 4 of Bubeck et al. (2013a), that provides a lower bound on the sum of type I and type II probabilities of error in a statistical test.

Lemma 20 *Let ρ_0,ρ_1 be two probability distributions supported on some set $\mathcal{X}$, with ρ_1 absolutely continuous with respect to ρ_0. Then for any measurable function $\phi : \mathcal{X} \to \{0,1\}$, one has*

$$\mathbb{P}_{X\sim\rho_0}(\phi(X) = 1) + \mathbb{P}_{X\sim\rho_1}(\phi(X) = 0) \geq \frac{1}{2}\exp(-\mathrm{KL}(\rho_0, \rho_1)).$$

Let ν and ν' be two bandit models that do not have the same set of optimal arms. We denote by $\mathcal{S}_1, \ldots, \mathcal{S}_M$ the $M = \binom{K}{m}$ subsets of m, ordered so that $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) is the set of m best arms in problem ν (resp. ν'). One has

$$\begin{aligned}
\max\left(\mathbb{P}_\nu(\hat{\mathcal{S}}_m \neq \mathcal{S}_1), \mathbb{P}_{\nu'}(\hat{\mathcal{S}}_m \neq \mathcal{S}_2)\right) &\geq \frac{1}{2}\left(\mathbb{P}_\nu(\hat{\mathcal{S}}_m \neq \mathcal{S}_1) + \mathbb{P}_{\nu'}(\hat{\mathcal{S}}_m \neq \mathcal{S}_2)\right) \\
&\geq \frac{1}{2}\left(\mathbb{P}_\nu(\hat{\mathcal{S}}_m \neq \mathcal{S}_1) + \mathbb{P}_{\nu'}(\hat{\mathcal{S}}_m = \mathcal{S}_1)\right).
\end{aligned}$$

Let $\rho_0 = \mathcal{L}(\hat{\mathcal{S}}_m)$ and $\rho_1 = \mathcal{L}'(\hat{\mathcal{S}}_m)$ be the distribution of $\hat{\mathcal{S}}_m$ for algorithm $\mathcal{A}$ under problems ν and ν' respectively. ρ_1 is absolutely continuous with respect to ρ_0, since as mentioned above, for any event in $\mathcal{F}_t$, $\mathbb{P}_\nu(A) = 0 \Leftrightarrow \mathbb{P}_{\nu'}(A) = 0$. Therefore one can apply Lemma 20 with ρ_0,ρ_1 and $\phi(x) = \mathbb{1}_{(x\neq\mathcal{S}_1)}$ and write

$$\max\left(\mathbb{P}_\nu(\hat{\mathcal{S}}_m \neq \mathcal{S}_1), \mathbb{P}_{\nu'}(\hat{\mathcal{S}}_m \neq \mathcal{S}_2)\right) \geq \frac{1}{4}\exp\left(-\mathrm{KL}(\mathcal{L}(\hat{\mathcal{S}}_m), \mathcal{L}'(\hat{\mathcal{S}}_m))\right).$$

To conclude the proof, it remains to show that $\mathrm{KL}(\mathcal{L}(\hat{\mathcal{S}}_m), \mathcal{L}'(\hat{\mathcal{S}}_m))$ is upper bounded by $\sum_{a=1}^{K}\mathbb{E}_\nu[N_a(t)]\mathrm{KL}(\nu_a, \nu'_a)$, which is equal to $\mathbb{E}_\nu[L_t]$, as shown above (equation (18)).

The rest of the proof boils down to prove a lower bound on $\mathbb{E}_\nu[L_t]$ slightly different from the one used to obtain Lemma 1. For $k \in \{1, \ldots, M\}$, applying inequality (19) to $(\hat{\mathcal{S}}_m = \mathcal{S}_k) \in \mathcal{F}_\tau$ yields

$$\mathbb{E}_\nu[L_t|\hat{\mathcal{S}}_m = \mathcal{S}_k] \geq \log\left(\frac{\mathbb{P}_\nu(\hat{\mathcal{S}}_m = \mathcal{S}_k)}{\mathbb{P}_{\nu'}(\hat{\mathcal{S}}_m = \mathcal{S}_k)}\right).$$

Thus one can write, letting $\mathcal{I} = \{k \in \{1, \ldots, M\} : \mathbb{P}_\nu(\hat{\mathcal{S}}_m = \mathcal{S}_k) \neq 0\}$,

$$\begin{aligned}
\mathbb{E}_\nu[L_t] &= \sum_{k\in\mathcal{I}} \mathbb{E}_\nu[L_t|\hat{\mathcal{S}}_m = \mathcal{S}_k]\mathbb{P}(\hat{\mathcal{S}}_m = \mathcal{S}_k) \\
&\geq \sum_{k\in\mathcal{I}} \log\left(\frac{\mathbb{P}_\nu(\hat{\mathcal{S}}_m = \mathcal{S}_k)}{\mathbb{P}_{\nu'}(\hat{\mathcal{S}}_m = \mathcal{S}_k)}\right)\mathbb{P}_\nu(\hat{\mathcal{S}}_m = \mathcal{S}_k) = \mathrm{KL}(\mathcal{L}(\hat{\mathcal{S}}_m), \mathcal{L}'(\hat{\mathcal{S}}_m)),
\end{aligned}$$

which concludes the proof.

A.3 Proof of Lemma 18

Recall that for all $a \in \{1,\dots,K\}$ there exists a measure λ_a such that ν_a (resp. ν'_a) has density f_a (resp. f'_a) with respect to λ_a. For all $a \in \{1,\dots,K\}$, let $(Y_{a,t})_{t\in\mathbb{N}}$ be an i.i.d. sequence such that if $A_t = a$, $Z_t = Y_{a,t}$.

We start by showing by induction that for all $n \in \mathbb{N}$ the following statement is true: for every function $g : \mathbb{R}^n \to \mathbb{R}$ measurable,

$$\mathbb{E}_{\nu'}[g(Z_1,\dots,Z_n)] = \mathbb{E}_{\nu}\left[g(Z_1,\dots,Z_n)\exp(-L_n(Z_1,\dots,Z_n))\right].$$

The result for $n = 1$ follows from the following calculation:

$$\begin{aligned}
\mathbb{E}_{\nu'}[g(Z_1)] &= \mathbb{E}_{\nu'}\left[\sum_{a=1}^K \mathbb{1}_{(A_1=a)} g(Y_{a,1})\right] = \sum_{a=1}^K \mathbb{E}_{\nu'}\left[\mathbb{1}_{(A_1=a)}\mathbb{E}_{\nu'}[g(Y_{a,1})|\mathcal{F}_0]\right] \\
&= \sum_{a=1}^K \mathbb{P}_{\nu'}(A_1=a)\mathbb{E}_{\nu'}[g(Y_{a,1})] = \sum_{a=1}^K \mathbb{P}_{\nu}(A_1=a)\mathbb{E}_{\nu}\left[g(Y_{a,1})\frac{f'_a(Y_{a,1})}{f_a(Y_{a,1})}\right] \\
&= \mathbb{E}_{\nu}\left[\sum_{a=1}^K \mathbb{1}_{(A_1=a)} g(Y_{a,1})\frac{f'_a(Y_{a,1})}{f_a(Y_{a,1})}\right] \\
&= \mathbb{E}_{\nu}\left[g(Z_1)\sum_{a=1}^K \mathbb{1}_{(A_1=a)}\exp\left(-\log\frac{f'_a(Z_1)}{f_a(Z_1)}\right)\right] \\
&= \mathbb{E}_{\nu}\left[g(Z_1)\exp\left(-\sum_{a=1}^K \mathbb{1}_{(A_1=a)}\log\frac{f'_a(Z_1)}{f_a(Z_1)}\right)\right] \\
&= \mathbb{E}_{\nu}\left[g(Z_1)\exp(-L_1(Z_1))\right].
\end{aligned}$$

We use that the initial choice of action satisfies $\mathbb{P}_\nu(A_1 = a) = \mathbb{P}_{\nu'}(A_1 = a)$.

We now assume that the statement holds for some integer n, and show it holds for $n+1$. Let $g : \mathbb{R}^{n+1} \to \mathbb{R}$ be a measurable function.

$$\begin{aligned}
&\mathbb{E}_{\nu'}[g(Z_1,\dots,Z_n,Z_{n+1})] = \mathbb{E}_{\nu'}[\mathbb{E}_{\nu'}[g(Z_1,\dots,Z_n,Z_{n+1})|\mathcal{F}_n]] \\
&\quad\overset{(*)}{=} \mathbb{E}_{\nu}\left[\mathbb{E}_{\nu'}[g(Z_1,\dots,Z_n,Z_{n+1})|\mathcal{F}_n]\exp\left(-L_n(Z_1,\dots,Z_n)\right)\right] \\
&\quad= \mathbb{E}_{\nu}\left[\sum_{a=1}^K \mathbb{1}_{A_{n+1}=a}\mathbb{E}_{\nu'}[g(Z_1,\dots,Z_n,Y_{a,n+1})|\mathcal{F}_n]\exp\left(-L_n(Z_1,\dots,Z_n)\right)\right] \\
&\quad= \mathbb{E}_{\nu}\left[\sum_{a=1}^K \mathbb{1}_{A_{n+1}=a}\int g(Z_1,\dots,Z_n,z)\frac{f'_a(z)}{f_a(z)}f_a(z)d\lambda_a(z)\exp\left(-L_n(Z_1,\dots,Z_n)\right)\right].
\end{aligned}$$

Observing that on the event $(A_{n+1} = a)$, $L_{n+1}(Z_1,\dots,Z_n,z) = L_n(Z_1,\dots,Z_n) + \log\frac{f_a(z)}{f'_a(z)}$ leads to:

$$\begin{aligned}
&\mathbb{E}_{\nu'}[g(Z_1,\dots,Z_n,Z_{n+1})] \\
&\quad= \mathbb{E}_{\nu}\left[\sum_{a=1}^K \mathbb{1}_{A_{n+1}=a}\int g(Z_1,\dots,Z_n,z)\exp(-L_{n+1}(Z_1,\dots,Z_n,z))f_a(z)d\lambda_a(z)\right] \\
&\quad= \mathbb{E}_{\nu}\left[\sum_{a=1}^K \mathbb{1}_{A_{n+1}=a}\mathbb{E}_{\nu}\left[g(Z_1,\dots,Z_n,Y_{a,n+1})\exp(-L_{n+1}(Z_1,\dots,Z_n,Y_{a,n+1}))|\mathcal{F}_n\right]\right] \\
&\quad= \mathbb{E}_{\nu}\left[\mathbb{E}_{\nu}\left[g(Z_1,\dots,Z_n,Z_{n+1})\exp(-L_{n+1}(Z_1,\dots,Z_n,Z_{n+1}))|\mathcal{F}_n\right]\right] \\
&\quad= \mathbb{E}_{\nu}\left[g(Z_1,\dots,Z_n,Z_{n+1})\exp(-L_{n+1}(Z_1,\dots,Z_n,Z_{n+1}))\right].
\end{aligned}$$

Hence, the statement is true for all n, and we have shown that for every $\mathcal{E} \in \mathcal{F}_n$,

$$\mathbb{P}_{\nu'}(\mathcal{E}) = \mathbb{E}_{\nu}[\mathbb{1}_{\mathcal{E}}\exp(-L_n)].$$

Let σ be a stopping time w.r.t. $(\mathcal{F}_n)$ and $\mathcal{E} \in \mathcal{F}_\sigma$.

$$\mathbb{P}_{\nu'}(\mathcal{E}) = \mathbb{E}_{\nu'}[\mathbb{1}_{\mathcal{E}}] = \sum_{n=0}^\infty \mathbb{E}_{\nu'}[\underbrace{\mathbb{1}_{\mathcal{E}}\mathbb{1}_{(\sigma=n)}}_{\in\mathcal{F}_n}] = \sum_{n=0}^\infty \mathbb{E}_{\nu}[\mathbb{1}_{\mathcal{E}}\mathbb{1}_{(\sigma=n)}\exp(-L_n)] = \mathbb{E}_{\nu}[\mathbb{1}_{\mathcal{E}}\exp(-L_\sigma)].$$

Appendix B. A Short Proof of Burnetas and Katehakis' Lower Bound on the Regret

In the regret minimization framework, briefly described in the Introduction, a bandit algorithm only consists in a sampling rule (there is no stopping rule nor recommendation rule). The arms must be chosen sequentially so as to minimize the regret, that is strongly related to the number of draws of the sub-optimal arms (using the notation $\mu^* = \mu_{[1]}$):

$$R_T(\nu) = \mu^* T - \mathbb{E}_\nu\left[\sum_{t=1}^T Z_t\right] = \sum_{a:\mu_a<\mu^*}(\mu^* - \mu_a)\,\mathbb{E}_\nu\big[N_a(T)\big] \tag{20}$$

The lower bound given by Lai and Robbins (1985) on the regret holds for families of distributions parameterized by a (single) real parameter. Their result has been generalized by Burnetas and Katehakis (1996) to larger classes of parametric distributions. The version we give here deals with identifiable classes of the form $\mathcal{M} = (\mathcal{P})^K$, where $\mathcal{P}$ is a set of probability measures satisfying

$$\forall \nu_a,\nu_b \in \mathcal{P},\ \nu_a \neq \nu_b \ \Rightarrow\ 0 < \mathrm{KL}(\nu_a,\nu_b) < +\infty.$$

Theorem 21 *Let $\mathcal{M}$ be an identifiable class of bandit models. Consider a bandit algorithm such that for all $\nu \in \mathcal{M}$ having a unique optimal arm, for all $\alpha \in (0,1]$, $R_T(\nu) = o(T^\alpha)$. Then, for all $\nu \in \mathcal{M}$,*

$$\mu_a < \mu^* \ \Rightarrow\ \liminf_{T\to\infty}\frac{\mathbb{E}_\nu[N_a(T)]}{\log(T)} \geq \frac{1}{\mathcal{K}_{inf}(\nu_a;\mu^*)}, \tag{21}$$

where $\mathcal{K}_{inf}(p;\mu) = \inf\{\mathrm{KL}(p,q) : q \in \mathcal{P} \text{ and } \mathbb{E}_{X\sim q}[X] > \mu\}$.

Proof. Let $\nu = (\nu_1, \dots, \nu_K)$ be a bandit model such that arm 1 is the unique optimal arm. Without loss of generality, we show that inequality (21) holds for the sub-optimal arm $a = 2$. Consider the alternative bandit model ν' such that $\nu'_a = \nu_a$ for all $a \neq 2$ and $\nu'_2 \in \mathcal{P}$ is such that $\mathbb{E}_{X\sim\nu'_2}[X] > \mu_1$. Arm 1 is thus the unique optimal arm under the model ν, whereas arm 2 is the unique optimal arm under the model ν'. For every integer T, let $\mathcal{E}_T$ be the event defined by

$$\mathcal{E}_T = \left(N_1(T) \leq T - \sqrt{T}\right).$$

Clearly, $\mathcal{E}_T \in \mathcal{F}_T$. From Lemma 1, applied to the stopping time $\sigma = T$ a.s.,

$$\mathbb{E}_\nu[N_2(T)]\mathrm{KL}(\nu_2, \nu'_2) \geq d(\mathbb{P}_\nu(\mathcal{E}_T), \mathbb{P}_{\nu'}(\mathcal{E}_T)). \tag{22}$$

The event $\mathcal{E}_T$ is not very likely to hold under the model ν, in which the optimal arm should be drawn of order $T - C\log(T)$ times, whereas it is very likely to happen under ν', in which arm 1 is sub-optimal and thus only drawn little. More precisely, Markov inequality yields

$$\begin{aligned}
\mathbb{P}_\nu(\mathcal{E}_T) &= \mathbb{P}_\nu(T - N_1(T) \geq \sqrt{T}) \leq \frac{\sum_{a\neq 1}\mathbb{E}_\nu[N_a(T)]}{\sqrt{T}} \\
\mathbb{P}_{\nu'}(\mathcal{E}_T^c) &= \mathbb{P}_{\nu'}(N_1(T) \geq T - \sqrt{T}) \leq \frac{\mathbb{E}_{\nu'}[N_1(T)]}{T - \sqrt{T}} \leq \frac{\sum_{a\neq 2}\mathbb{E}_{\nu'}[N_a(T)]}{T - \sqrt{T}}
\end{aligned}$$

From the formulation (20), every algorithm that is uniformly efficient in the above sense satisfies

$$\sum_{a\neq 1}\mathbb{E}_\nu[N_a(T)] = o(T^\alpha) \quad \text{and} \quad \sum_{a\neq 2}\mathbb{E}_{\nu'}[N_a(T)] = o(T^\alpha)$$

for all $\alpha \in (0, 1]$. Hence $\mathbb{P}_\nu(\mathcal{E}_T) \underset{n\to\infty}{\to} 0$ and $\mathbb{P}_{\nu'}(\mathcal{E}_T) \underset{n\to\infty}{\to} 1$. Therefore, we get

$$\frac{d(\mathbb{P}_\nu(\mathcal{E}_T), \mathbb{P}_{\nu'}(\mathcal{E}_T))}{\log(T)} \underset{T\to\infty}{\sim} \frac{1}{\log(T)}\log\left(\frac{1}{\mathbb{P}_{\nu'}(\mathcal{E}_T^c)}\right) \geq \frac{1}{\log(T)}\log\left(\frac{T - \sqrt{T}}{\sum_{a\neq 2}\mathbb{E}_{\nu'}[N_a(T)]}\right).$$

The right hand side rewrites

$$1 + \frac{\log\left(1 - \frac{1}{\sqrt{T}}\right)}{\log(T)} - \frac{\log\left(\sum_{a\neq 2}\mathbb{E}_{\nu'}[N_a(T)]\right)}{\log(T)} \underset{T\to\infty}{\to} 1$$

using the fact that $\sum_{a\neq 2}\mathbb{E}_{\nu'}[N_a(T)] = o(T^\alpha)$ for all $\alpha \in (0, 1]$. Finally, for every $\nu'_2 \in \mathcal{P}$ such that $\mathbb{E}_{X\sim\nu'_2}[X] > \mu_1$ on obtains, using inequality (22)

$$\liminf_{T\to\infty} \frac{\mathbb{E}[N_2(T)]}{\log(T)} \geq \frac{1}{\mathrm{KL}(\nu_2, \nu'_2)}.$$

For all $\epsilon \in (0, 1)$, ν'_2 can then be chosen such that $\mathrm{KL}(\nu_2, \nu'_2) \leq \mathcal{K}_{\inf}(\nu_2, \mu_1)/(1 - \epsilon)$, and the conclusion follows when ϵ goes to zero.

Appendix C. Properties of K_* and K^* in Exponential Families

In this section, we review properties of K_* defined in Section 3 as well as those of K^* defined in Section 5 in the case of one-parameter exponential family distributions. We recall that $\mathrm{K}_*(\theta_1, \theta_2) = \mathrm{K}(\theta_1, \theta_*)$ where θ_* is defined by $\mathrm{K}(\theta_1, \theta_*) = \mathrm{K}(\theta_2, \theta_*)$ and that $\mathrm{K}^*(\theta_1, \theta_2) = \mathrm{K}(\theta^*, \theta_1)$ where θ^* is defined by $\mathrm{K}(\theta^*, \theta_1) = \mathrm{K}(\theta^*, \theta_2)$.

Figure 7 displays the geometric constructions corresponding to the complexity terms of Theorems 4 and 6, respectively. As seen on the picture, the convexity of the function $\theta \mapsto K(\theta_i, \theta)$, for any value of θ_i, implies that

$$\frac{1}{\mathrm{K}_*(\theta_1, \theta_2)} \geq \frac{1}{\mathrm{K}(\theta_1, \theta_2)} + \frac{1}{\mathrm{K}(\theta_2, \theta_1)}.$$

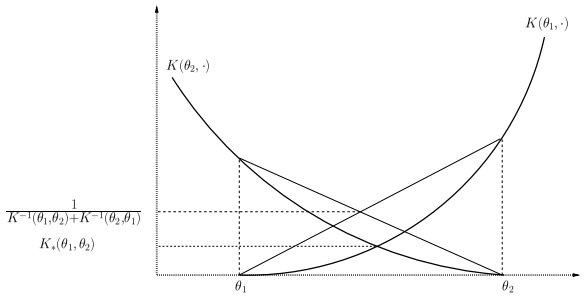


Figure 7: Comparison of the complexity terms featured in Theorems 4 and 6.

It is well known that in exponential families, the Kullback-Leibler divergence between distributions parameterized by their natural parameter, θ, may be related to the Bregman divergence associated with the log-partition function b:

$$\mathrm{K}(\theta_1, \theta_2) = b(\theta_2) - b(\theta_1) - \dot{b}(\theta_1)(\theta_2 - \theta_1) = \mathrm{Bregman}_b(\theta_2, \theta_1).$$

From this representation, its is straightforward to show that

- $\mathrm{K}(\theta_1, \theta_2)$ is a twice differentiable strictly convex function of its second argument,
- θ^* corresponds to the dual parameter $\mu^* := \dot{b}(\theta^*) = (b(\theta_2) - b(\theta_1))/(\theta_2 - \theta_1)$,
- $\mathrm{K}^*(\theta_1, \theta_2)$ admits the following variational representation

$$\mathrm{K}^*(\theta_1, \theta_2) = \max_{\theta\in(\theta_1,\theta_2)} \left\{ b(\theta_1) + \frac{b(\theta_2) - b(\theta_1)}{\theta_2 - \theta_1}(\theta - \theta_1) - b(\theta), \right\},$$

  corresponding to the maximal gap shown on Figure 8 (achieved in θ^* for which $\dot{b}(\theta^*) = \mu^*$). The quantity $I^*(\theta_1, \theta_2)$ related to the use of uniform sampling, is equal to the value of the gap in $\theta = (\theta_1 + \theta_2)/2$, which confirms that it is indeed smaller than $\mathrm{K}^*(\theta_1, \theta_2)$.

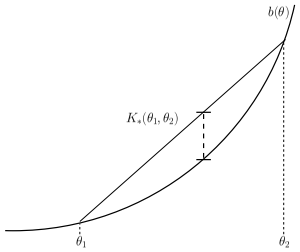


Figure 8: Interpretation of $\mathrm{K}^*(\theta_1,\theta_2)$.

Indexing the distributions in the exponential family by their mean $\mu = \dot{b}(\theta)$ rather than their natural parameter θ and using the dual representation

$$\mathrm{K}(\mu_1,\mu_2) = b^\star(\mu_1) - b^\star(\mu_2) - \dot{b}^\star(\mu_2)(\mu_1-\mu_2) = \mathrm{Bregman}_{b^\star}(\mu_1,\mu_2),$$

where $b^\star(\mu) := \sup_\theta(\theta\mu - b(\theta))$ is the Fenchel conjugate of b, similarly yields

- $\mathrm{K}(\mu_1,\mu_2)$ is a twice differentiable strictly convex function of its first argument,
- $\theta_* = \dot{b}^\star(\mu_*) = (b^\star(\mu_2) - b^\star(\mu_1))/(\mu_2-\mu_1)$;
- $\mathrm{K}_*(\theta_1,\theta_2)$ is defined by

$$\mathrm{K}_*(\theta_1,\theta_2) = \max_{\mu\in(\mu_1,\mu_2)} \left\{ b^\star(\mu_1) + \frac{b^\star(\mu_2)-b^\star(\mu_1)}{\mu_2-\mu_1}(\mu-\mu_1) - b^\star(\mu), \right\}.$$

From what precedes, equality between K_* and K^* for all values of the parameters is only achievable when the log-partition function b is self-conjugate.

Appendix D. Proof of Theorem 9

Let $\alpha = \sigma_1/(\sigma_1+\sigma_2)$. We first prove that with the exploration rate $\beta(t,\delta) = \log(t/\delta) + 2\log\log(6t)$ the algorithm is δ-PAC. Assume that $\mu_1 > \mu_2$ and recall $\tau = \inf\{t\in\mathbb{N} : |d_t| > \sqrt{2\sigma_t^2(\alpha)\beta(t,\delta)}\}$, where $d_t := \hat{\mu}_1(t) - \hat{\mu}_2(t)$. The probability of error of the α-elimination strategy is upper bounded by

$$\begin{aligned}
\mathbb{P}_\nu\left(d_\tau \le -\sqrt{2\sigma_\tau^2(\alpha)\beta(\tau,\delta)}\right) &\le \mathbb{P}_\nu\left(d_\tau - (\mu_1-\mu_2) \le -\sqrt{2\sigma_\tau^2(\alpha)\beta(\tau,\delta)}\right)\\
&\le \mathbb{P}_\nu\left(\exists t\in\mathbb{N}^* : d_t - (\mu_1-\mu_2) < -\sqrt{2\sigma_t^2(\alpha)\beta(t,\delta)}\right)\\
&\le \sum_{t=1}^{\infty} \exp\left(-\beta(t,\delta)\right),
\end{aligned}$$

by an union bound and Chernoff bound applied to $d_t - (\mu_1-\mu_2) \sim \mathcal{N}\left(0,\sigma_t^2(\alpha)\right)$. The choice of $\beta(t,\delta)$ mentioned above ensures that the series in the right hand side is upper bounded

by δ, which shows the algorithm is δ-PAC:

$$\begin{aligned}
\sum_{t=1}^{\infty} e^{-\beta(t,\delta)} &\le \delta\sum_{t=1}^{\infty}\frac{1}{t(\log(6t))^2} \le \delta\left(\frac{1}{(\log 6)^2} + \int_1^\infty \frac{dt}{t(\log(6t))^2}\right)\\
&= \delta\left(\frac{1}{(\log 6)^2} + \frac{1}{\log(6)}\right) \le \delta.
\end{aligned}$$

To upper bound the expected sample complexity, we start by upper bounding the probability that τ exceeds some deterministic time T:

$$\begin{aligned}
\mathbb{P}_\nu(\tau\ge T) &\le \mathbb{P}_\nu\left(\forall t = 1\dots T,\ d_t \le \sqrt{2\sigma_t^2(\alpha)\beta(t,\delta)}\right) \le \mathbb{P}_\nu\left(d_T \le \sqrt{2\sigma_T^2(\alpha)\beta(T,\delta)}\right)\\
&= \mathbb{P}_\nu\left(d_T - (\mu_1-\mu_2) \le -\left[(\mu_1-\mu_2) - \sqrt{2\sigma_T^2(\alpha)\beta(T,\delta)}\right]\right)\\
&\le \exp\left(-\frac{1}{2\sigma_T^2(\alpha)}\left[(\mu_1-\mu_2) - \sqrt{2\sigma_T^2(\alpha)\beta(T,\delta)}\right]^2\right).
\end{aligned}$$

The last inequality follows from Chernoff bound and holds for T such that $(\mu_1-\mu_2) > \sqrt{2\sigma_T^2(\alpha)\beta(T,\delta)}$. Now, for $\gamma\in(0,1)$ we introduce

$$T_\gamma^* := \inf\left\{ t_0\in\mathbb{N} : \forall t\ge t_0, (\mu_1-\mu_2) - \sqrt{2\sigma_t^2(\alpha)\beta(t,\delta)} > \gamma(\mu_1-\mu_2)\right\}.$$

This quantity is well defined as $\sigma_t^2(\alpha)\beta(t,\delta)$ goes to zero when t goes to infinity. Then,

$$\begin{aligned}
\mathbb{E}_\nu[\tau] &\le T_\gamma^* + \sum_{T=T_\gamma^*+1} \mathbb{P}\left(\tau\ge T\right)\\
&\le T_\gamma^* + \sum_{T=T_\gamma^*+1} \exp\left(-\frac{1}{2\sigma_T^2(\alpha)}\left[(\mu_1-\mu_2) - \sqrt{2\sigma_T^2(\alpha)\beta(T,\delta)}\right]^2\right)\\
&\le T_\gamma^* + \sum_{T=T_\gamma^*+1}^{\infty} \exp\left(-\frac{1}{2\sigma_T^2(\alpha)}\gamma^2(\mu_1-\mu_2)^2\right).
\end{aligned}$$

For all $t\in\mathbb{N}^*$, it is easy to show that the following upper bound on $\sigma_t^2(\alpha)$ holds:

$$\forall t\in\mathbb{N},\ \sigma_t^2(\alpha) \le \frac{(\sigma_1+\sigma_2)^2}{t}\times\frac{t-\frac{\sigma_1}{\sigma_2}}{t-\frac{\sigma_1}{\sigma_2}-1}. \tag{23}$$

Using the bound (23), one has

$$\begin{aligned}
\mathbb{E}_\nu[\tau] &\le T_\gamma^* + \int_0^\infty \exp\left(-\frac{t}{2(\sigma_1+\sigma_2)^2}\frac{t-\frac{\sigma_1}{\sigma_2}-1}{t-\frac{\sigma_1}{\sigma_2}}\gamma^2(\mu_1-\mu_2)^2\right)dt\\
&\le T_\gamma^* + \frac{2(\sigma_1+\sigma_2)^2}{\gamma^2(\mu_1-\mu_2)^2}\exp\left(\frac{\gamma^2(\mu_1-\mu_2)^2}{2(\sigma_1+\sigma_2)^2}\right).
\end{aligned}$$

We now give an upper bound on T^*_γ. Let $r \in [0, e/2-1]$. There exists $N_0(r)$ such that for $t \geq N_0(r)$, $\beta(t,\delta) \leq \log(t^{1+r}/\delta)$. Using also (23), one gets $T^*_\gamma = \max(N_0(t), \tilde{T}_\gamma)$, where

$$\tilde{T}_\gamma = \inf\left\{ t_0 \in \mathbb{N} : \forall t \geq t_0, \frac{(\mu_1-\mu_2)^2}{2(\sigma_1+\sigma_2)^2}(1-\gamma)^2 t > \frac{t - \frac{\sigma_1}{\sigma_2} - 1}{t - \frac{\sigma_1}{\sigma_2}} \log \frac{t^{1+r}}{\delta} \right\}.$$

If $t > (1+\gamma\frac{\sigma_1}{\sigma_2})/\gamma$ one has $(t-\frac{\sigma_1}{\sigma_2}-1)/(t-\frac{\sigma_1}{\sigma_2}) \leq (1-\gamma)^{-1}$. Thus $\tilde{T}_\gamma = \max((1+\gamma\frac{\sigma_1}{\sigma_2})/\gamma, T'_\gamma)$, with

$$T'_\gamma = \inf\left\{ t_0 \in \mathbb{N} : \forall t \geq t_0, \exp\left(\frac{(\mu_1-\mu_2)^2}{2(\sigma_1+\sigma_2)^2}(1-\gamma)^3 t\right) \geq \frac{t^{1+r}}{\delta} \right\}.$$

The following Lemma, whose proof can be found below, helps us bound this last quantity.

Lemma 22 *For every $\beta, \eta > 0$ and $s \in [1, e/2]$, the following implication is true:*

$$x_0 = \frac{s}{\beta}\log\left(\frac{e\log\left(1/(\beta^s\eta)\right)}{\beta^s\eta}\right) \quad \Rightarrow \quad \forall x \geq x_0, \quad e^{\beta x} \geq \frac{x^s}{\eta}.$$

Applying Lemma 22 with $\eta = \delta$, $s = 1+r$ and $\beta = (1-\gamma)^3(\mu_1-\mu_2)^2/(2(\sigma_1+\sigma_2)^2)$ leads to

$$T'_\gamma \leq \frac{(1+r)}{(1-\gamma)^3} \times \frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}\left[\log\frac{1}{\delta} + \log\log\frac{1}{\delta}\right] + R(\mu_1,\mu_2,\sigma_1,\sigma_2,\gamma,r),$$

with

$$R(\mu_1,\mu_2,\sigma_1,\sigma_2,\gamma,r) = \frac{1+r}{(1-\gamma)^3}\frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}\left[1 + (1+r)\log\left(\frac{2(\sigma_1+\sigma_2)^2}{(1-\gamma)^3(\mu_1-\mu_2)^2}\right)\right].$$

Now for $\epsilon > 0$ fixed, choosing r and γ small enough leads to

$$\mathbb{E}_\nu[\tau] \leq (1+\epsilon)\frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}\left[\log\frac{1}{\delta} + \log\log\frac{1}{\delta}\right] + \mathcal{C}(\mu_1,\mu_2,\sigma_1,\sigma_2,\epsilon),$$

where $\mathcal{C}$ is a constant independent of δ. It can be noted that $\mathcal{C}(\mu_1,\mu_2,\sigma_1,\sigma_2,\epsilon)$ goes to infinity when ϵ goes to zero, but for a fixed $\epsilon > 0$,

$$(1+\epsilon)\frac{2(\sigma_1+\sigma_2)^2}{(\mu_1-\mu_2)^2}\log\log\frac{1}{\delta} + \mathcal{C}(\mu_1,\mu_2,\sigma_1,\sigma_2,\epsilon) = \underset{\delta\to 0}{o_\epsilon}\left(\log\frac{1}{\delta}\right),$$

which concludes the proof.

Proof of Lemma 22. Lemma 22 easily follows from the fact that for $\eta > 0$ and $s \in [1, e/2]$,

$$x_0 = s\log\left(\frac{e\log\left(\frac{1}{\eta}\right)}{\eta}\right) \quad \Rightarrow \quad \forall x \geq x_0, \quad e^x \geq \frac{x^s}{\eta}$$

Indeed, it suffices to apply this statement to $x = x\beta$ and $\eta = \eta\beta^s$. The mapping $x \mapsto e^x - x^s/\eta$ is increasing when $x \geq s$. As $x_0 \geq s$, it suffices to prove that x_0 defined above

satisfies $e^{x_0} \geq x_0^s/\eta$.

$$\begin{aligned}
\log\left(\frac{x_0^s}{\eta}\right) &= s\log\left(s\log\left(\frac{e\log\frac{1}{\eta}}{\eta}\right)\right) + \log\frac{1}{\eta} \\
&= s\left(\log(s) + \log\left[\log\frac{1}{\eta} + \log\left(e\log\frac{1}{\eta}\right)\right]\right) + \log\frac{1}{\eta} \\
&\leq s\left(\log(s) + \log\left[2\log\frac{1}{\eta}\right]\right) + \log\frac{1}{\eta}
\end{aligned}$$

where we use that for all y, $\log(y) \leq \frac{1}{e}y$. Then, using that $s \geq 1$,

$$\log\left(\frac{x_0^s}{\eta}\right) \leq s\left(\log(s) + \log(2) + \log\log\frac{1}{\eta} + \log\frac{1}{\eta}\right).$$

For $s \leq \frac{e}{2}$, $\log(s) + \log(2) \leq 1$, hence

$$\log\left(\frac{x_0^s}{\eta}\right) \leq s\left(1 + \log\log\frac{1}{\eta} + \log\frac{1}{\eta}\right) = s\log\left(\frac{e\log\left(\frac{1}{\eta}\right)}{\eta}\right) = x_0,$$

which is equivalent to $e^{x_0} \geq \frac{x_0^s}{\eta}$ and concludes the proof.

Appendix E. A Refined Exploration Rate for α-Elimination

E.1 Proof of Theorem 8

According to (15), to prove Theorem 8 it is enough to show that for

$$\beta(t,\delta) = \log(1/\delta) + 3\log\log(1/\delta) + (3/2)\log\log(et),$$

if $S_t = \sum_{s=1}^t X_s$ is a sum of i.i.d $\mathcal{N}(0,1)$ random variables, one has

$$\mathbb{P}(\exists t \in \mathbb{N}^* : S_t > \sqrt{2t\beta(t,\delta)}) \leq \delta. \tag{24}$$

Let $z = \log(1/\delta)$. Using Lemma 7, one can write, choosing $x = z + 3\log z$ and $\beta = 3/2$,

$$\mathbb{P}\Big(\exists t \in \mathbb{N} : S_t > \sqrt{2t\beta(t,\delta)}\Big) \leq \frac{\sqrt{e}}{8}\zeta\Big(\frac{3}{2} - \frac{3}{4(z+3\log z)}\Big)\frac{(\sqrt{z+3\log z}+\sqrt{8})^{3/2}}{z^3}\delta.$$

It can be shown numerically that for $z \geq 2.03$,

$$\frac{\sqrt{e}}{8}\zeta\Big(\frac{3}{2} - \frac{3}{4(z+3\log z)}\Big)\frac{(\sqrt{z+3\log z}+\sqrt{8})^{3/2}}{z^3} \leq 1.$$

Thus for $\delta \leq \exp(-2.03) \leq 0.1$, inequality (24) holds.

E.2 Proof of Lemma 7.

We start by stating three technical lemmas, whose proofs are partly omitted.

Lemma 23 *For every $\eta > 0$, every positive integer k, and every integer t such that $(1+\eta)^{k-1} \leq t \leq (1+\eta)^k$,*

$$\sqrt{\frac{(1+\eta)^{k-1/2}}{t}} + \sqrt{\frac{t}{(1+\eta)^{k-1/2}}} \leq (1+\eta)^{1/4} + (1+\eta)^{-1/4} .$$

Lemma 24 *For every $\eta > 0$,*

$$A(\eta) := \frac{4}{\left((1+\eta)^{1/4} + (1+\eta)^{-1/4}\right)^2} \geq 1 - \frac{\eta^2}{16}.$$

Lemma 25 *Let t be such that $(1+\eta)^{k-1} \leq t \leq (1+\eta)^k$. Then,*

$$\sigma\sqrt{2z} \geq \frac{A(\eta)z}{\lambda\sqrt{t}} + \frac{\lambda\sigma^2\sqrt{t}}{2}, \quad \textit{with} \quad \lambda = \sigma^{-1}\sqrt{2zA(\eta)/(1+\eta)^{k-1/2}}.$$

Proof of Lemma 25.

$$\frac{A(\eta)z}{\lambda\sqrt{t}} + \frac{\lambda\sigma^2\sqrt{t}}{2} = \frac{\sigma\sqrt{2zA(\eta)}}{2}\left(\sqrt{\frac{(1+\eta)^{k-1/2}}{t}} + \sqrt{\frac{t}{(1+\eta)^{k-1/2}}}\right) \leq \sigma\sqrt{2z}$$

according to Lemma 23.

□

An important fact is that for every $\lambda \in \mathbb{R}$, because the X_i are σ-subgaussian, $W_t = \exp(\lambda S_t - t\frac{\lambda^2\sigma^2}{2}))$ is a super-martingale, and thus, for every positive u,

$$\mathbb{P}\left(\bigcup_{t\geq 1}\left\{\lambda S_t - t\frac{\lambda^2\sigma^2}{2} > u\right\}\right) \leq \exp(-u). \tag{25}$$

Let $\eta \in (0, e-1]$ to be defined later, and let $T_k = \mathbb{N} \cap \left[(1+\eta)^{k-1}, (1+\eta)^k\right[$.

$$\mathbb{P}\left(\bigcup_{t\geq 1}\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{x+\beta\log\log(et)}\right\}\right) \leq \sum_{k=1}^{\infty}\mathbb{P}\left(\bigcup_{t\in T_k}\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{x+\beta\log\log(et)}\right\}\right)$$
$$\leq \sum_{k=1}^{\infty}\mathbb{P}\left(\bigcup_{t\in T_k}\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{x+\beta\log\left(k\log(1+\eta)\right)}\right\}\right) .$$

We use that $\eta \leq e-1$ to obtain the last inequality since this condition implies

$$\log(\log(e(1+\eta)^{k-1}) \geq \log(k\log(1+\eta)).$$

For $k \geq 1$, let $z_k = x + \beta\log\left(k\log(1+\eta)\right)$ and $\lambda_k = \sigma^{-1}\sqrt{2z_kA(\eta)/(1+\eta)^{k-1/2}}$. Lemma 25 shows that for every $t \in T_k$,

$$\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{z_k}\right\} \subset \left\{\frac{S_t}{\sqrt{t}} > \frac{A(\eta)z_k}{\lambda_k\sqrt{t}} + \frac{\sigma^2\lambda_k\sqrt{t}}{2}\right\} .$$

Thus, by inequality (25),

$$\mathbb{P}\left(\bigcup_{t\in T_k}\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{z_k}\right\}\right) \leq \mathbb{P}\left(\bigcup_{t\in T_k}\left\{\frac{S_t}{\sqrt{t}} > \frac{A(\eta)z_k}{\lambda_k\sqrt{t}} + \frac{\sigma^2\lambda_k\sqrt{t}}{2}\right\}\right)$$
$$= \mathbb{P}\left(\bigcup_{t\in T_k}\left\{\lambda_kS_t - \frac{\sigma^2\lambda_k^2t}{2} > A(\eta)z_k\right\}\right)$$
$$\leq \exp\left(-A(\eta)z_k\right) = \frac{\exp(-A(\eta)x)}{(k\log(1+\eta))^{\beta A(\eta)}} .$$

One chooses $\eta^2 = 8/x$ for x such that $x \geq \frac{8}{(e-1)^2}$ (which ensures $\eta \leq e-1$). Using Lemma 24, one obtains that $\exp(-A(\eta)x) \leq \sqrt{e}\exp(-x)$. Moreover,

$$\frac{1}{\log(1+\eta)} \leq \frac{1+\eta}{\eta} = \frac{\sqrt{x}}{2\sqrt{2}} + 1 .$$

Thus,

$$\mathbb{P}\left(\bigcup_{t\in T_k}\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{z_k}\right\}\right) \leq \frac{\sqrt{e}}{k^{\beta A(\eta)}}\left(\frac{\sqrt{x}}{2\sqrt{2}} + 1\right)^{\beta A(\eta)}\exp(-x)$$
$$\leq \frac{\sqrt{e}}{k^{\beta A(\eta)}}\left(\frac{\sqrt{x}}{2\sqrt{2}} + 1\right)^{\beta}\exp(-x).$$

Hence,

$$\mathbb{P}\left(\bigcup_{t\geq 1}\left\{\frac{S_t}{\sigma\sqrt{2t}} > \sqrt{x+\beta\log\log(et)}\right\}\right) \leq \sqrt{e}\zeta\left(\beta A(\eta)\right)\left(\frac{\sqrt{x}}{2\sqrt{2}} + 1\right)^{\beta A(\eta)}\exp\left(-x\right)$$
$$\leq \sqrt{e}\zeta\left(\beta\left(1 - \frac{1}{2x}\right)\right)\left(\frac{\sqrt{x}}{2\sqrt{2}} + 1\right)^{\beta}\exp\left(-x\right),$$

using the lower bound on $A(\eta)$ given in Lemma 24 and the fact that $A(\eta)$ is upper bounded by 1.

Appendix F. Bernoulli Bandit Models

F.1 Proof of Lemma 11

Assume that $\mu_1 < \mu_2$. Recall the KL-LUCB algorithm of Kaufmann and Kalyanakrishnan (2013). For two-armed bandit models, this algorithm samples the arms uniformly and builds

for both arms a confidence interval based on KL-divergence $\mathcal{I}_a(t) = [l_{a,t/2}, u_{a,t/2}]$, with

$$\begin{aligned} u_{a,s} &= \sup\{q > \hat{\mu}_{a,s} : sd(\hat{\mu}_{a,s}, q) \leq \tilde{\beta}(s,\delta)\}, \quad \text{where} \quad d(x,y) = \mathrm{KL}(\mathcal{B}(x), \mathcal{B}(y)) \\ l_{a,s} &= \inf\{q < \hat{\mu}_{a,s} : sd(\hat{\mu}_{a,s}, q) \leq \tilde{\beta}(s,\delta)\}, \end{aligned}$$

for some exploration rate that we denote by $\tilde{\beta}(t,\delta)$. The algorithm stops when the confidence intervals are separated; that is either $l_{1,t/2} > u_{2,t/2}$ or $l_{2,t/2} > u_{1,t/2}$, and recommends the empirical best arm. A picture helps to convince oneself that

$$(l_{1,s} > u_{2,s}) \;\Leftrightarrow\; (\hat{\mu}_{1,s} > \hat{\mu}_{2,s}) \cap (sd_*(\hat{\mu}_{1,s}, \hat{\mu}_{2,s}) > \beta(s,\delta)) \tag{26}$$

Additionally, as mentioned before, $I_*(x,y)$ is very close to the quantity $d_*(x,y)$ and one has more precisely $I_*(x,y) < d_*(x,y)$. Using all this, we can upper bound the probability of error of Algorithm 2 in the following way.

$$\begin{aligned} &\mathbb{P}_\nu\left(\exists t \in 2\mathbb{N}^* : \hat{\mu}_{1,t/2} > \hat{\mu}_{2,t/2}, tI_*(\hat{\mu}_{1,t/2}, \hat{\mu}_{2,t/2}) > \beta(t,\delta)\right) \\ &\quad \leq \mathbb{P}_\nu\left(\exists t \in 2\mathbb{N}^* : \hat{\mu}_{1,t/2} > \hat{\mu}_{2,t/2}, (t/2)d_*(\hat{\mu}_{1,t/2}, \hat{\mu}_{2,t/2}) > (\beta(t,\delta)/2)\right) \\ &\quad = \mathbb{P}_\nu\left(\exists s \in \mathbb{N}^* : \hat{\mu}_{1,s} > \hat{\mu}_{2,s}, sd_*(\hat{\mu}_{1,s}, \hat{\mu}_{2,s}) > (\beta(2s,\delta)/2)\right) \\ &\quad = \mathbb{P}_\nu(\exists s \in \mathbb{N}^* : l_{1,s} > u_{2,s}) \leq \mathbb{P}_\nu(\exists s \in \mathbb{N}^* : (\mu_1 < l_{1,s}) \cup (\mu_2 > u_{2,s})) \\ &\quad \leq 2\sum_{s=1}^{\infty} \exp(-\beta(2s,\delta)/2) \end{aligned}$$

where the last inequality follows from an union bound and for example Lemma 4 of Kaufmann and Kalyanakrishnan (2013). Note that the indices $l_{1,s}$ and $u_{2,s}$ involved here use the exploration rate $\tilde{\beta}(s,\delta) = \beta(2s,\delta)/2$. The choice $\beta(t,\delta)$ in the statement of the Lemma shows the last series is upper bounded by δ, which concludes the proofs.

F.2 An Asymptotic Bound for the Stopping Time

Lemma 26 *Consider a strategy that uses uniform sampling and a stopping rule of the form*

$$\tau = \inf\left\{t \in 2\mathbb{N}^* : tf(\hat{\mu}_{1,t/2}, \hat{\mu}_{2,t/2}) \geq \log\left(\frac{g(t)}{\delta}\right)\right\}$$

where f is a continuous function such that $f(\mu_1,\mu_2) \neq 0$ and $g(t) = o(t^r)$ for all $r > 0$. Then for all $\epsilon > 0$,

$$\mathbb{P}_\nu\left(\limsup_{\delta\to 0} \frac{\tau}{\log(1/\delta)} \leq \frac{1+\epsilon}{f(\mu_1,\mu_2)}\right) = 1.$$

Proof. We fix $\epsilon > 0$ and introduce

$$\sigma = \max\left\{t \in 2\mathbb{N}^* : f(\hat{\mu}_{1,t/2}, \hat{\mu}_{2,t/2}) \leq \frac{f(\mu_1,\mu_2)}{1+\epsilon/2}\right\}.$$

By the law of large numbers, $\mathbb{P}(\sigma < +\infty) = 1$. Hence, $\lim_{n\to\infty} \mathbb{P}(\sigma \leq n) = 1$ and for every $\alpha \in (0,1)$ there exists $N(\epsilon,\alpha,\mu_1,\mu_2)$ such that $\mathbb{P}(\sigma \leq N(\epsilon,\alpha,\mu_1,\mu_2)) \geq 1-\alpha$. Therefore, introducing the event

$$E_\alpha = \left(\forall t \geq N(\epsilon,\alpha,\mu_1,\mu_2), f(\hat{\mu}_{1,t/2}, \hat{\mu}_{2,t/2}) > \frac{f(\mu_1,\mu_2)}{1+\epsilon/2}\right), \quad \text{one has} \quad \mathbb{P}(E_\alpha) \geq 1-\alpha.$$

On the event E_α,

$$\begin{aligned} \tau &\leq \max\left(N(\epsilon,\alpha,\mu_1,\mu_2); \inf\left\{t \in \mathbb{N} : t\frac{f(\mu_1,\mu_2)}{1+\epsilon/2} \geq \log\left(\frac{g(t)}{\delta}\right)\right\}\right) \\ \tau &\leq N(\epsilon,\alpha,\mu_1,\mu_2) + \inf\left\{t \in \mathbb{N} : t\frac{f(\mu_1,\mu_2)}{1+\epsilon/2} \geq \log\left(\frac{g(t)}{\delta}\right)\right\} \end{aligned}$$

We can use Lemma 22 to bound the right term in the right hand side, which shows that there exists a constant $C(\epsilon,\mu_1,\mu_2)$ independent of δ such that

$$\tau \leq N(\epsilon,\alpha,\mu_1,\mu_2) + \frac{1+\epsilon}{f(\mu_1,\mu_2)}\left[\log\frac{1}{\delta} + \log\log\frac{1}{\delta}\right] + C(\epsilon,\mu_1,\mu_2)$$

Thus we proved that for all $\alpha > 0$,

$$\mathbb{P}\left(\limsup_{\delta\to 0} \frac{\tau}{\log(1/\delta)} \leq \frac{1+\epsilon}{f(\mu_1,\mu_2)}\right) \geq 1-\alpha.$$

This concludes the proof.

Appendix G. Upper and Lower Bounds in the Fixed-Budget Setting

G.1 Proof of Theorem 12

Without loss of generality, assume that the bandit model $\nu = (\nu_1,\nu_2)$ is such that $a^* = 1$. Consider any alternative bandit model $\nu' = (\nu_1',\nu_2')$ in which $a^* = 2$. Let $\mathcal{A}$ be a consistent algorithm such that $\tau = t$ and consider the event $A = (\hat{\mathcal{S}}_1 = 1)$. Clearly $A \in \mathcal{F}_t = \mathcal{F}_\tau$.

Lemma 1 applied to the stopping time $\sigma = t$ a.s. and the event A gives

$$\mathbb{E}_{\nu'}[N_1(t)]\mathrm{KL}(\nu_1',\nu_1) + \mathbb{E}_{\nu'}[N_2(t)]\mathrm{KL}(\nu_2',\nu_2) \geq d(\mathbb{P}_{\nu'}(A), \mathbb{P}_\nu(A)).$$

Note that $p_t(\nu) = 1 - \mathbb{P}_\nu(A)$ and $p_t(\nu') = \mathbb{P}_{\nu'}(A)$. As algorithm $\mathcal{A}$ is correct on both ν and ν', for every $\epsilon > 0$ there exists $t_0(\epsilon)$ such that for all $t \geq t_0(\epsilon)$, $\mathbb{P}_{\nu'}(A) \leq \epsilon \leq \mathbb{P}_\nu(A)$. For $t \geq t_0(\epsilon)$,

$$\mathbb{E}_{\nu'}[N_1(t)]\mathrm{KL}(\nu_1',\nu_1) + \mathbb{E}_{\nu'}[N_2(t)]\mathrm{KL}(\nu_2',\nu_2) \geq d(\epsilon, 1-p_t(\nu)) \geq (1-\epsilon)\log\frac{1-\epsilon}{p_t(\nu)} + \epsilon\log\epsilon.$$

Taking the limsup and letting ϵ go to zero, one can show that

$$\limsup_{t\to\infty} -\frac{1}{t}\log p_t(\nu) \leq \limsup_{t\to\infty} \sum_{a=1}^{2} \frac{\mathbb{E}_{\nu'}[N_a(t)]}{t}\mathrm{KL}(\nu_a',\nu_a) \leq \max_{a=1,2} \mathrm{KL}(\nu_a',\nu_a).$$

Optimizing over the possible model ν' satisfying $\mu_1' < \mu_2'$ to make the right hand side of the inequality as small as possible gives the result.

For algorithms using uniform sampling, $\limsup -\frac{1}{t}\log p_t(\nu)$ is upper bounded by the quantity $(\mathrm{KL}(\nu_1',\nu_1) + \mathrm{KL}(\nu_2',\nu_2))/2$, which yields the second statement of the Theorem.

G.2 An Optimal Static Strategy for Exponential Families

Bounding the probability of error of a static strategy using n_1 samples from arm 1 and n_2 samples from arm 2 relies on the following lemma.

Lemma 27 *Let $(X_{1,t})_{t\in\mathbb{N}}$ and $(X_{2,t})_{t\in\mathbb{N}}$ be two independent i.i.d sequences, such that $X_{1,1}\sim\nu_{\theta_1}$ and $X_{2,1}\sim\nu_{\theta_2}$ belong to an exponential family. Assume that $\mu(\theta_1)>\mu(\theta_2)$. Then*

$$\mathbb{P}\left(\frac{1}{n_1}\sum_{t=1}^{n_1}X_{1,t}<\frac{1}{n_2}\sum_{t=1}^{n_2}X_{2,t}\right)\leq\exp(-(n_1+n_2)g_\alpha(\theta_1,\theta_2)), \tag{27}$$

where $\alpha=\frac{n_1}{n_1+n_2}$ and $g_\alpha(\theta_1,\theta_2):=\alpha\mathrm{K}(\alpha\theta_1+(1-\alpha)\theta_2,\theta_1)+(1-\alpha)\mathrm{K}(\alpha\theta_1+(1-\alpha)\theta_2,\theta_2)$.

The function $\alpha\mapsto g_\alpha(\theta_1,\theta_2)$, can be maximized analytically, and the value α^* that realizes the maximum is given by

$$\begin{aligned}\mathrm{K}(\alpha^*\theta_1+(1-\alpha^*)\theta_2,\theta_1) &= \mathrm{K}(\alpha^*\theta_1+(1-\alpha^*)\theta_2,\theta_2)\\ \alpha^*\theta_1+(1-\alpha^*)\theta_1 &= \theta^*\\ \alpha^* &= \frac{\theta^*-\theta_2}{\theta_1-\theta_2}\end{aligned}$$

where θ^* is defined by $\mathrm{K}(\theta^*,\theta_1)=\mathrm{K}(\theta^*,\theta_2)=\mathrm{K}^*(\theta_1,\theta_2)$. More interestingly, the associated rate is such that

$$g_{\alpha^*}(\theta_1,\theta_2)=\alpha^*\mathrm{K}(\theta^*,\theta_1)+(1-\alpha^*)\mathrm{K}(\theta^*,\theta_2)=\mathrm{K}^*(\theta_1,\theta_2),$$

which leads to Theorem 13.

Remark 28 *When $\mu_1>\mu_2$, applying Lemma 27 with $n_1=n_2=t/2$ yields*

$$\mathbb{P}\left(\hat{\mu}_{1,t/2}<\mu_{2,t/2}\right)\leq\exp\left(-\frac{\mathrm{K}\left(\theta_1,\frac{\theta_1+\theta_2}{2}\right)+\mathrm{K}\left(\theta_2,\frac{\theta_1+\theta_2}{2}\right)}{2}t\right)=\exp\big(-I_*(\nu)t\big),$$

which shows that the strategy using uniform sampling and recommending the empirical best arm matches the lower bound (17) in Theorem 12.

Proof of Lemma 27. The i.i.d. sequences $(X_{1,t})_{t\in\mathbb{N}}$ and $(X_{2,t})_{t\in\mathbb{N}}$ have respective densities f_{θ_1} and f_{θ_2} where $f_\theta(x)=\exp(\theta x-b(\theta))$ and $\mu(\theta_1)=\mu_1,\mu(\theta_2)=\mu_2$. α is such that $n_1=\alpha n$ and $n_2=(1-\alpha)n$. One can write

$$\mathbb{P}\left(\frac{1}{n_1}\sum_{t=1}^{n_1}X_{1,t}-\frac{1}{n_2}\sum_{t=1}^{n_2}X_{2,t}<0\right)=\mathbb{P}\left(\alpha\sum_{t=1}^{n_2}X_{2,t}-(1-\alpha)\sum_{t=1}^{n_1}X_{1,t}\geq 0\right).$$

For every $\lambda>0$, multiplying by λ, taking the exponential of the two sides and using Markov's inequality (this technique is often referred to as Chernoff's method), one gets

$$\begin{aligned}\mathbb{P}\left(\frac{1}{n_1}\sum_{t=1}^{n_1}X_{1,t}-\frac{1}{n_2}\sum_{t=1}^{n_2}X_{2,t}<0\right) &\leq \left(\mathbb{E}_\nu[e^{\lambda\alpha X_{2,1}}]\right)^{(1-\alpha)n}\left(\mathbb{E}_\nu[e^{\lambda(1-\alpha)X_{1,1}}]\right)^{\alpha n}\\ &= \exp\Bigg(n\underbrace{\left[(1-\alpha)\phi_{X_{2,1}}(\lambda\alpha)+\alpha\phi_{X_{1,1}}(-(1-\alpha)\lambda)\right]}_{G_\alpha(\lambda)}\Bigg)\end{aligned}$$

with $\phi_X(\lambda)=\log\mathbb{E}_\nu[e^{\lambda X}]$ for any random variable X. If $X\sim f_\theta$ a direct computation gives $\phi_X(\lambda)=b(\lambda+\theta)-b(\theta)$. Therefore the function $G_\alpha(\lambda)$ introduced above rewrites

$$G_\alpha(\lambda)=(1-\alpha)(b(\lambda\alpha+\theta_2)-b(\theta_2))+\alpha(b(\theta_1-(1-\alpha)\lambda)-b(\theta_1)).$$

Using that $b'(x)=\mu(x)$, we can compute the derivative of G and see that this function as a unique minimum in λ^* given by

$$\mu(\theta_1-(1-\alpha)\lambda^*)=\mu(\theta_2+\alpha\lambda^*)\quad\Leftrightarrow\quad\theta_1-(1-\alpha)\lambda^*=\theta_2+\alpha\lambda^*\quad\Leftrightarrow\quad\lambda^*=\theta_1-\theta_2,$$

using that $\theta\mapsto\mu(\theta)$ is one-to-one. One can also show that

$$G(\lambda^*) \;=\; (1-\alpha)[b(\alpha\theta_1+(1-\alpha)\theta_2)-b(\theta_2)]+\alpha[b(\alpha\theta_1+(1-\alpha)\theta_2)-b(\theta_1)].$$

Using the expression of the KL-divergence between ν_{θ_1} and ν_{θ_2} as a function of the natural parameters: $\mathrm{K}(\theta_1,\theta_2)=\mu(\theta_1)(\theta_1-\theta_2)-b(\theta_1)+b(\theta_2)$, one can also show that

$$\begin{aligned}&\alpha\mathrm{K}(\alpha\theta_1+(1-\alpha)\theta_2,\theta_1)\\ &\qquad=-\alpha(1-\alpha)\mu(\alpha\theta_1+(1-\alpha)\theta_2)(\theta_1-\theta_2)+\alpha[-b(\alpha\theta_1+(1-\alpha)\theta_2)+b(\theta_1)]\\ &(1-\alpha)\mathrm{K}(\alpha\theta_1+(1-\alpha)\theta_2,\theta_2)\\ &\qquad=\alpha(1-\alpha)\mu(\alpha\theta_1+(1-\alpha)\theta_2)(\theta_1-\theta_2)+(1-\alpha)[-b(\alpha\theta_1+(1-\alpha)\theta_2)+b(\theta_2)]\end{aligned}$$

Summing these two equalities leads to

$$G(\lambda^*)=-\left[\alpha\mathrm{K}(\alpha\theta_1+(1-\alpha)\theta_2,\theta_1)+(1-\alpha)\mathrm{K}(\alpha\theta_1+(1-\alpha)\theta_2,\theta_2)\right]=-g_\alpha(\theta_1,\theta_2).$$

Hence the inequality $\mathbb{P}\left(\frac{1}{n_1}\sum_{t=1}^{n_1}X_{1,t}<\frac{1}{n_2}\sum_{t=1}^{n_2}X_{2,t}\right)\leq\exp(nG(\lambda^*))$ concludes the proof.

G.3 Proof of Theorem 17

First, with Δ_a as defined in the introduction, there exists one arm $a\in\{1,\dots,K\}$ such that $\mathbb{E}_\nu[N_a(t)]\leq 2\sigma^2t/(H(\nu)\Delta_a^2)$. Otherwise, a contradiction is easily obtained.

Case 1 If $a\in\{1,\dots m\}$ there exists $b\in\{m+1,\dots K\}$ such that $\mathbb{E}_\nu[N_b(t)]\leq\frac{2\sigma^2t}{H^-(\nu)\Delta_b^2}$.

Case 2 If $a\in\{m+1,\dots K\}$ there exists $b\in\{1,\dots,m\}$ such that $\mathbb{E}_\nu[N_b(t)]\leq\frac{2\sigma^2t}{H^+(\nu)\Delta_b^2}$.

These two cases are very similar, and the idea is to propose an easier alternative model in which we change only arm a and b, the arms that are less drawn among the set of good and the set of bad arms. Assume that we are in Case 1. We introduce $\nu^{[a,b]}$ a Gaussian bandit model such that:

$$\begin{cases}\mu'_k &= \mu_k \text{ for all } k\notin\{a,b\}\\ \mu'_a &= \mu_a-2\Delta_b\\ \mu'_b &= \mu_b+2\Delta_a\end{cases}$$

In $\nu^{[a,b]}$ good arm a becomes a bad arm and bad arm b becomes a good arm. One can easily check (or convince oneself with Figure 4) that $H(\nu^{[a,b]})\leq H(\nu)$ and as already explained, ν and $\nu^{[a,b]}$ do not share their optimal arms. Thus Lemma 15 yields

$$\begin{aligned}\max\left(p_t(\nu),p_t(\nu^{[a,b]})\right) &\geq \frac{1}{4}\exp\left(-\left[\mathbb{E}_\nu[N_a(t)]\mathrm{KL}(\nu_a,\nu'_a))+\mathbb{E}_\nu[N_b(t)]\mathrm{KL}(\nu_b,\nu'_b)\right]\right)\\ &= \frac{1}{4}\exp\left(-\left[\mathbb{E}_\nu[N_a(t)]\frac{(2\Delta_a)^2}{2\sigma^2}+\mathbb{E}_\nu[N_b(t)]\frac{(2\Delta_b)^2}{2\sigma^2}\right]\right),\end{aligned}$$

thus

$$\begin{aligned}\max\left(p_t(\nu), p_t(\nu^{[a,b]})\right) &\geq \frac{1}{4}\exp\left(-\left[\frac{2\sigma^2 t}{H\Delta_a^2}\frac{4\Delta_a^2}{2\sigma^2}+\frac{2\sigma^2 t}{H^-\Delta_b^2}\frac{4\Delta_b^2}{2\sigma^2}\right]\right)\\ &= \frac{1}{4}\exp\left(-\frac{4t}{\tilde{H}}\right) \quad \text{with} \quad \tilde{H}=\frac{HH^-}{H+H^-}.\end{aligned}$$

References

S. Agrawal and N. Goyal. Further optimal regret bounds for Thompson Sampling. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013.

J-Y. Audibert, S. Bubeck, and R. Munos. Best arm identification in multi-armed bandits. In *Conference on Learning Theory (COLT)*, 2010.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, 2002.

Robert Bechhofer, Jack Kiefer, and Milton Sobel. *Sequential identification and ranking procedures*. The university of Chicago Press, 1968.

S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in finitely armed and continuous armed bandits. *Theoretical Computer Science*, 412:1832–1852, 2011.

S. Bubeck, V. Perchet, and P. Rigollet. Bounded regret in stochastic multi-armed bandits. In *Conference On Learning Theory (COLT)*, 2013a.

S. Bubeck, T. Wang, and N. Viswanathan. Multiple identifications in multi-armed bandits. In *International Conference on Machine Learning (ICML)*, 2013b.

A.N Burnetas and M. Katehakis. Optimal adaptive policies for sequential allocation problems. *Advances in Applied Mathematics*, 17(2):122–142, 1996.

O. Cappé, A. Garivier, O-A. Maillard, R. Munos, and G. Stoltz. Kullback-Leibler upper confidence bounds for optimal sequential allocation. *Annals of Statistics*, 41(3):1516–1541, 2013.

T. Cover and J. Thomas. *Elements of information theory (2nd Edition)*. Wiley, 2006.

E. Even-Dar, S. Mannor, and Y. Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7:1079–1105, 2006.

V. Gabillon, M. Ghavamzadeh, and A. Lazaric. Best arm identification: a unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

V. Heidrich-Meisner and C. Igel. Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In *International Conference on Machine Learning (ICML)*, 2009.

J. Honda and A. Takemura. An asymptotically optimal policy for finite support models in the multiarmed bandit problem. *Machine Learning*, 85(3):361–391, 2011.

K. Jamieson, M. Malloy, R. Nowak, and S. Bubeck. lil'UCB: an optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory (COLT)*, 2014.

C. Jennison, I.M. Johnstone, and B.W. Turnbull. Asymptotically optimal procedures for sequential adaptive selection of the best of several normal means. *Statistical Decision Theory and Related Topics III*, 2:55–86, 1982.

S. Kalyanakrishnan, A. Tewari, P. Auer, and P. Stone. PAC subset selection in stochastic multi-armed bandits. In *International Conference on Machine Learning (ICML)*, 2012.

Z. Karnin, T. Koren, and O. Somekh. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning (ICML)*, 2013.

E. Kaufmann and S. Kalyanakrishnan. Information complexity in bandit subset selection. In *Conference On Learning Theory (COLT)*, 2013.

E. Kaufmann, A. Garivier, and O. Cappé. On Bayesian upper-confidence bounds for bandit problems. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012a.

E. Kaufmann, N. Korda, and R. Munos. Thompson Sampling : an asymptotically optimal finite-time analysis. In *Algorithmic Learning Theory (ALT)*, 2012b.

T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

S. Mannor and J. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, pages 623–648, 2004.

O. Maron and A. Moore. The Racing algorithm: Model selection for lazy learners. *Artificial Intelligence Review*, 11(1-5):113–131, 1997.

E. Paulson. A sequential procedure for selecting the population with the largest mean from k normal populations. *Annals of Mathematical Statistics*, 35:174–180, 1964.

H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

H. Robbins. Statistical methods related to the law of the iterated logarithm. *Annals of Mathematical Statistics*, 41(5):1397–1409, 1970.

D. Siegmund. *Sequential analysis*. Springer-Verlag, 1985.

W.R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285–294, 1933.

A. Wald. Sequential tests of statistical hypotheses. *Annals of Mathematical Statistics*, 16(2): 117–186, 1945.

Journal of Machine Learning Research 17 (2016) 1-51 Submitted 5/14; Published 2/16

Multiscale Dictionary Learning: Non-Asymptotic Bounds and Robustness

Mauro Maggioni MAURO@MATH.DUKE.EDU
Departments of Mathematics, Electrical and Computer Engineering, and Computer Science
Duke University
Durham, NC 27708, USA

Stanislav Minsker MINSKER@USC.EDU
Department of Mathematics
University of Southern California
Los Angeles, CA 90089, USA

Nate Strawn NATE.STRAWN@GEORGETOWN.EDU
Department of Mathematics and Statistics
Georgetown University
Washington D.C., 20057, USA

Editor: Benjamin Recht

Abstract

High-dimensional datasets are well-approximated by low-dimensional structures. Over the past decade, this empirical observation motivated the investigation of detection, measurement, and modeling techniques to exploit these low-dimensional intrinsic structures, yielding numerous implications for high-dimensional statistics, machine learning, and signal processing. Manifold learning (where the low-dimensional structure is a manifold) and dictionary learning (where the low-dimensional structure is the set of sparse linear combinations of vectors from a finite dictionary) are two prominent theoretical and computational frameworks in this area. Despite their ostensible distinction, the recently-introduced Geometric Multi-Resolution Analysis (GMRA) provides a robust, computationally efficient, multiscale procedure for simultaneously learning manifolds and dictionaries.

In this work, we prove non-asymptotic probabilistic bounds on the approximation error of GMRA for a rich class of data-generating statistical models that includes "noisy" manifolds, thereby establishing the theoretical robustness of the procedure and confirming empirical observations. In particular, if a dataset aggregates near a low-dimensional manifold, our results show that the approximation error of the GMRA is completely independent of the ambient dimension. Our work therefore establishes GMRA as a provably fast algorithm for dictionary learning with approximation and sparsity guarantees. We include several numerical experiments confirming these theoretical results, and our theoretical framework provides new tools for assessing the behavior of manifold learning and dictionary learning procedures on a large class of interesting models.

Keywords: dictionary learning, multi-resolution analysis, manifold learning, robustness, sparsity

1. Introduction

In many high-dimensional data analysis problems, existence of *efficient data representations* can dramatically boost the statistical performance and the computational efficiency of learning algorithms. Inversely, in the absence of efficient representations, the curse of dimensionality implies that required sample sizes must grow exponentially with the ambient dimension, which ostensibly renders many statistical learning tasks completely untenable. Parametric statistical modeling seeks to resolve this difficulty by restricting the family of candidate distributions for the data to a collection of probability measures indexed by a finite-dimensional parameter. By contrast, nonparametric statistical models are more flexible and oftentimes more precise, but usually require data samples of large sizes unless the data exhibits some simple latent structure (e.g., some form of sparsity). Such structural considerations are essential for establishing convergence rates, and oftentimes these structural considerations are geometric in nature.

One classical geometric assumption asserts that the data, modeled as a set of points in $\mathbb{R}^D$, in fact lies on (or perhaps very close to) *a single d-dimensional affine subspace* $V \in \mathbb{R}^D$ where $d \ll D$. Tools such as PCA (see Hotelling, 1933, 1936; Pearson, 1901) estimate V in a stable fashion under suitable assumptions. Generalizing this model, one may assert that the data lies on a union of several low-dimensional affine subspaces instead of just one, and in this case the estimation of the *multiple affine subspaces* from data samples already inspired intensive research due to its subtle complexity (e.g., see Chen and Lerman, 2009; Chen and Maggioni, 2011; Elhamifar and Vidal, 2009; Fischler and Bolles, 1981; Ho et al., 2003; Liu et al., 2010; Ma et al., 2007, 2008; Sugaya and Kanatani, 2004; Tipping and Bishop, 1999; Vidal et al., 2005; Yan and Pollefeys, 2006; Zhang et al., 2010). A widely used form of this model is that of k-sparse data, where there exists a dictionary (i.e., a collection of vectors) $\Phi = \{\varphi_i\}_{i=1}^m \subset \mathbb{R}^D$ such that each observed data point $x \in \mathbb{R}^d$ may be expressed as a linear combination of at most $k \ll D$ elements of Φ. These *sparse representations* offer great convenience and expressivity for signal processing tasks (such as in Peyré, 2009; Protter and Elad, 2007), compressive sensing, statistical estimation, and learning (e.g., see Aharon et al., 2006; Candes and Tao, 2007; Chen et al., 1998; Donoho, 2006; Kreutz-Delgado et al., 2003; Lewicki et al., 1998; Maurer and Pontil, 2010, among others), and even exhibits connections with representations in the visual cortex (see Olshausen and Field, 1997). In geometric terminology, such sparse representations are generally attainable when the local *intrinsic dimension* of the observations is small. For these applications, the dictionary is usually assumed to be known a priori, instead of being learned from the data, but it has been recognized in the past decade that data-dependent dictionaries may perform significantly better than generic dictionaries even in classical signal processing tasks.

The k-sparse data model motivates a large amount of research in dictionary learning, where Φ is learned from data rather than being fixed in advance: given n samples $X_1, \ldots, X_n$ from a probability distribution μ in $\mathbb{R}^D$ representing the training data, an algorithm "learns" a dictionary $\widehat{\Phi}$ which provides sparse representations for the observations sampled from μ. This problem and its optimal algorithmic solutions are far from being well-understood, at least compared to the understanding that we have for classical dictionaries such as Fourier, wavelets, curvelets, and shearlets. These dictionaries arise in computational harmonic analysis approaches to image processing, and Donoho (1999) (for example) provides rigorous,

optimal approximation results for simple classes of images. The work of Gribonval et al. (2013) present general bounds for the complexity of learning the dictionaries (see also Maurer and Pontil, 2010; Vainsencher et al., 2011, and references therein). The algorithms used in dictionary learning are often computationally demanding, and many of them are based on high-dimensional non-convex optimization (Mairal et al., 2010). The emphasis of existing work is often made on the generality of the approach, where minimal assumptions are made on geometry of the distribution from which the sample is generated. These "pessimistic" techniques incur bounds dependent upon the ambient dimension D in general (even in the standard case of data lying on one hyperplane).

A different type of geometric assumption on the data gives rise to manifold learning, where the observations aggregate on a *suitably regular manifold* $\mathcal{M}$ of dimension d isometrically embedded in $\mathbb{R}^D$ (notable works include Belkin and Niyogi, 2003; Coifman et al., 2005a,b; Coifman and Maggioni, 2006; Donoho and Grimes, 2002, 2003; Fefferman et al.; Genovese et al., 2012b; Jones et al., 2008, 2010; Little et al., 2012, 2009; Roweis and Saul, 2000; Tenenbaum et al., 2000; Zhang and Zha, 2002, among others). This setting has been recognized as useful in a variety of applications (e.g. Causevic et al., 2006; Coifman et al., 2006; Rahman et al., 2005), influencing work in the applied mathematics and machine learning communities during the past several years. It has also been recognized that in many cases the data does not naturally aggregate on a smooth manifold (as in Little et al., 2012, 2009; Wakin et al., 2005), with examples arising in imaging that contradict the smoothness conditions. While this phenomenon is not as widely recognized as it probably could be, we believe that it is crucial to develop methods (both for dictionary and manifold learning) that are robust not only to noise, but also to modeling error. Such concerns motivated the work on intrinsic dimension estimation of noisy data sets (see Little et al., 2012, 2009), where smoothness of the underlying distribution of the data is not assumed, but only certain natural conditions (possibly varying with the scale of the data) are imposed. The central idea of the aforementioned works is to perform the multiscale singular value decomposition (SVD) of the data, an approach inspired by the works of David and Semmes (1993) and Jones (1990) in classical geometric measure theory. These techniques were further extended in several directions in the papers by Chen and Maggioni (2011); Chen et al. (2011a,b), while Allard et al. (2012); Chen and Maggioni (2010) built upon this work to construct multiscale dictionaries for the data based on the idea of Geometric Multi-Resolution Analysis (GMRA).

Until these recent works introduced the GMRA construction, connections between dictionary learning and manifold learning had not garnered much attention in the literature. These papers showed that, for intrinsically low-dimensional data, one may perform dictionary learning very efficiently by exploiting the underlying geometry, thereby illuminating the relationship between manifold learning and dictionary learning. In these papers, it was demonstrated that, in the infinite sample limit and under a manifold model assumption for the distribution of the data (with mild regularity conditions for the manifold), the GMRA algorithm efficiently learns a dictionary in which the data admits sparse representations. More interestingly, the examples in that paper show that the GMRA construction succeeds on real-world data sets which do not admit a structure consistent with the smooth manifold modeling assumption, suggesting that the GMRA construction exhibits robustness to modeling error. This desirable behavior follows naturally from design decisions; GMRA

combines two elements that add stability: a multiscale decomposition and localized SVD. Similar ideas appeared in work applying dictionary learning to computer vision problems, for example in the paper by Yu et al. (2009), where local linear approximations are used to create dictionaries. These techniques appeared at roughly the same time as GMRA (Chen and Maggioni (2010)), but were not multiscale in nature, and the selection of the local scale is crucial in applications. These techniques also lacked any finite or infinite sample guarantees, nor considered the effect of noise. They were however successfully applied in computer vision problems, most notably in the Pascal 2007 challenge.

In this paper, we analyze the finite sample behavior of (a slightly modified version of) that construction, and prove strong finite-sample guarantees for its behavior under general conditions on the geometry of a probability distribution generating the data. In particular, we show that these conditions are satisfied when the probability distribution is concentrated "near" a manifold, which robustly accounts for noise and modeling errors. In contrast to the pessimistic bounds mentioned above, the bounds that we prove only depend on the "intrinsic dimension" of the data. It should be noted that our method of proof produces non-asymptotic bounds, and requires several explicit geometric arguments not previously available in the literature (at least to the best of our knowledge). Some of our geometric bounds could be of independent interest to the manifold learning community.

The GMRA construction is therefore proven to simultaneously "learn" manifolds (in sense that it outputs a suitably close approximation to points on a manifold) and dictionaries in which data are represented sparsely. Moreover, the construction is guaranteed to be robust with respect to noise and to the "perturbations" of the manifold model. The GMRA construction is fast, linear in the size of the data matrix, inherently online, does not require nonlinear optimization, and is not iterative. Finally, our results may be combined with recent GMRA compressed sensing techniques and algorithms presented in Iwen and Maggioni (2013), yielding both a method to learn a dictionary in a stable way on a finite set of training data, and a way of performing compressive sensing and reconstruction (with guarantees) from a small number of (suitable) linear projections (again without the need for expensive convex optimization).

This paper is organized as follows: Section 2 introduces the main definitions and notation employed throughout the paper. Section 3 explains the main contributions, formally states the results and provides comparison with existing literature. Finally, Sections 4 and 5 are devoted to the proofs of our main results, Theorem 2 and Theorem 7.

2. Geometric Multi-Resolution Analysis (GMRA)

This section describes the main results of the paper, starting in a somewhat informal form. The statements will be made precise in the course of the exposition. In the statements below, "$\gtrsim$" and "$\lesssim$" denote inequalities up to multiplicative constants and logarithmic factors.

Statement of results. *Let $\sigma \geq 0$ be a fixed small constant, and let $\varepsilon \gtrsim \sigma$ be given. Suppose that $n \gtrsim \varepsilon^{-(1+d/2)}$, and let $\mathcal{X}_n = \{X_1, \ldots, X_n\}$ be an i.i.d. sample from Π, a probability distribution with density supported in a tube of radius σ around a smooth closed*

d-dimensional manifold $\mathcal{M} \hookrightarrow \mathbb{R}^D$, with $d > 1$. There exists an algorithm that, given $\mathcal{X}_n$, outputs the following objects:

- *a dictionary $\widehat{\Phi}_\varepsilon = \{\widehat{\varphi}_i\}_{i \in \mathcal{J}_\varepsilon} \subset \mathbb{R}^D$;*
- *a nonlinear "encoding" operator $\widehat{\mathcal{D}}_\varepsilon : \mathbb{R}^D \to \mathbb{R}^{\mathcal{J}_\varepsilon}$ which takes $x \in \mathbb{R}^D$ and returns the coefficients of its approximation by the elements of $\widehat{\Phi}_\varepsilon$;*
- *a "decoding" operator $\widehat{\mathcal{D}}_\varepsilon^{-1} : \mathbb{R}^{\mathcal{J}_\varepsilon} \to \mathbb{R}^D$ which maps a sequence of coefficients to an element of $\mathbb{R}^D$.*

Moreover, the following properties hold with high probability:

i. $\mathrm{Card}(\mathcal{J}_\varepsilon) \lesssim \varepsilon^{-d/2}$;

ii. the image of $\widehat{\mathcal{D}}_\varepsilon$ is contained in the set $S_{d+1} \subset \mathbb{R}^{\mathcal{J}_\varepsilon}$ of all $(d+1)$ - sparse vectors (i.e., vectors with at most $d+1$ nonzero coordinates);

iii. the reconstruction error satisfies

$$\sup_{x \in \mathrm{support}(\Pi)} \|x - \widehat{\mathcal{D}}_\varepsilon^{-1} \widehat{\mathcal{D}}_\varepsilon(x)\| \lesssim \varepsilon;$$

iv. the time complexity for computing

- *$\widehat{\Phi}_\varepsilon$ is $O(C^d(D+d^2)\varepsilon^{-(1+\frac{d}{2})}\log(1/\varepsilon))$, where C is a universal constant;*
- *$\widehat{\mathcal{D}}_\varepsilon(x)$ is $O(d(D + d\log(1/\varepsilon)))$, and for $\widehat{\mathcal{D}}_\varepsilon^{-1}(x)$ is $O(d(D+\log(1/\varepsilon)))$.*

If a new observation X_{n+1} from Π becomes available, $\widehat{\Phi}_\varepsilon$ may be updated in time $O(C^d(D+d^2)\log(1/\varepsilon))$.

In other words, we can construct a data-dependent dictionary $\widehat{\Phi}_\varepsilon$ of cardinality $O(\varepsilon^{-d/2})$ by looking at $O(\varepsilon^{-1-\frac{d}{2}})$ data points drawn from Π, such that $\widehat{\Phi}_\varepsilon$ provides both $(d+1)$-sparse approximations to data and has expected "reconstruction error" of order ε (with high probability). Note that the cost of encoding the $(d+1)$ non-zero coefficients requires $O((d+1)\log(\mathrm{Card}(\mathcal{J}_\varepsilon))) = O(d^2\log(1/\epsilon))$. Moreover, the algorithm producing this dictionary is fast and can be quickly updated if new points become available. We want to emphasize that the complexity of our construction only depends on the desired accuracy ε, and is independent of the total number of samples (for example, it is enough to use only the first $\simeq \varepsilon^{-(1+d/2)}$ data points). Many existing techniques in dictionary learning cannot guarantee a requested accuracy, or a given sparsity, and a certain computational cost as a function of the two. Our results above completely characterize the tradeoffs between desired precision, dictionary size, sparsity, and computational complexity for our dictionary learning procedure.

We also remark that a suitable version of compressed sensing applies to the dictionary representations used in the theorem: we refer the reader to the work by Iwen and Maggioni (2013), and its applications to hyperspectral imaging by Chen et al. (2012).

2.1 Notation

For $v \in \mathbb{R}^D$, $\|v\|$ denotes the standard Euclidean norm in $\mathbb{R}^D$. $B_d(0,r)$ is the Euclidean ball in $\mathbb{R}^d$ of radius r centered at the origin, and we let $B(0,r) := B_D(0,r)$. Proj_V stands for the orthogonal projection onto a linear subspace $V \subseteq \mathbb{R}^D$, $\dim(V)$ for its dimension and $V^\perp$ for its orthogonal complement. For $x \in \mathbb{R}^D$, let Proj_{x+V} be the affine projection onto the affine subspace $x+V$ defined by $\mathrm{Proj}_{x+V}(y) = x + \mathrm{Proj}_V(y-x)$, for $y \in \mathbb{R}^D$.

Given a matrix $A \in \mathbb{R}^{k \times l}$, we write $A = [a_1|\cdots|a_l]$, where a_i stands for the ith column of A. The operator norm is denoted by $\|A\|$, the Frobenius norm by $\|A\|_F$ and the matrix transpose by A^T. If $k = l$, $\mathrm{tr}\,(A)$ denotes the trace. For $v \in \mathbb{R}^k$, let $\mathrm{diag}(v)$ be the $k \times k$ diagonal matrix with $(\mathrm{diag}(v))_{ii} = v_i,\ i = 1, \ldots, k$. Finally, we use $\mathrm{span}\{a_i\}_{i=1}^l$ to denote the linear span of the columns of A.

Given a C^2 function $f : \mathbb{R}^l \to \mathbb{R}^k$, let f_i denote the ith coordinate of the function f for $i = 1, \ldots k$, $Df(v)$ the Jacobian of f at $v \in \mathbb{R}^l$, and $D^2 f_i(v)$ the Hessian of the ith coordinate at v.

We shall use $d\mathrm{Vol}$ to denote Lebesgue measure on $\mathbb{R}^D$, and if $U \subset \mathbb{R}^D$ is Lebesgue measurable, $\mathrm{Vol}(U)$ stands for the Lebesgue measure of U. We will use $\mathrm{Vol}_{\mathcal{M}}$ to denote the volume measure on a d-dimensional manifold $\mathcal{M}$ in $\mathbb{R}^D$ (note that this coincides with the d-dimensional Hausdorff measure for the subset $\mathcal{M}$ of $\mathbb{R}^D$), $U_{\mathcal{M}}$ - the uniform distribution over $\mathcal{M}$, and $d_{\mathcal{M}}(x,y)$ to denote the geodesic distance between two points $x, y \in \mathcal{M}$. For a probability measure Π on $\mathbb{R}^D$, $\mathrm{supp}(\Pi) := \cap_{C \text{ closed}, \Pi(C)=1} C$ stands for its support. Finally, for $x, y \in \mathbb{R}$, $x \vee y := \max(x,y)$.

2.2 Definition of the Geometric Multi-Resolution Analysis (GMRA)

We assume that the data are identically, independently distributed samples from a Borel probability measure Π on $\mathbb{R}^D$. Let $1 \le d \le D$ be an integer. A GMRA with respect to the probability measure Π consists of a collection of (nonlinear) operators $\{P_j : \mathbb{R}^D \to \mathbb{R}^D\}_{j \ge 0}$. For each "resolution level" $j \ge 0$, P_j is uniquely defined by a collection of pairs of subsets and affine projections, $\{(C_{j,k}, P_{j,k})\}_{k=1}^{N(j)}$, where the subsets $\{C_{j,k}\}_{k=1}^{N(j)}$ form a measurable partition of $\mathbb{R}^D$ (that is, members of $\{C_{j,k}\}_{k=1}^{N(j)}$ are pairwise disjoint and the union of all members is $\mathbb{R}^D$). P_j is constructed by piecing together local affine projections. Namely, let

$$P_{j,k}(x) := c_{j,k} + \mathrm{Proj}_{V_{j,k}}(x - c_{j,k}),$$

where $c_{j,k} \in \mathbb{R}^D$ and $V_{j,k}$ are defined as follows. Let $\mathbb{E}_{j,k}$ stand for the expectation with respect to the conditional distribution $d\Pi_{j,k}(x) = d\Pi(x | x \in C_{j,k})$. Then

$$c_{j,k} = \mathbb{E}_{j,k} x, \tag{1}$$

$$V_{j,k} = \operatorname*{argmin}_{\dim(V)=d} \mathbb{E}_{j,k} \|x - c_{j,k} - \mathrm{Proj}_V(x - c_{j,k})\|^2, \tag{2}$$

where the minimum is taken over all linear spaces V of dimension d. In other words, $c_{j,k}$ is the conditional mean and $V_{j,k}$ is the subspace spanned by eigenvectors corresponding to d largest eigenvalues of the conditional covariance matrix

$$\Sigma_{j,k} = \mathbb{E}_{j,k}[(x - c_{j,k})(x - c_{j,k})^T]. \tag{3}$$

Note that we have implicitly assumed that such a subspace $V_{j,k}$ is unique, which will always be the case throughout this paper. Given such a $\{(C_{j,k}, P_{j,k})\}_{k=1}^{N(j)}$, we define

$$P_j(x) := \sum_{k=1}^{N(j)} I\{x \in C_{j,k}\} P_{j,k}(x)$$

where $I\{x \in C_{j,k}\}$ is the indicator function of the set $C_{j,k}$.

It was shown in the paper by Allard et al. (2012) that if Π is supported on a smooth, closed d-dimensional submanifold $\mathcal{M} \hookrightarrow \mathbb{R}^D$, and if the partitions $\{C_{j,k}\}_{j=1}^{N(j)}$ satisfy some regularity conditions for each j, then, for any $x \in \mathcal{M}$, $\|x - P_j(x)\| \leq C(\mathcal{M})2^{-2j}$ for all $j \geq j_0(\mathcal{M})$. This means that the operators P_j provide an efficient "compression scheme" $x \mapsto P_j(x)$ for $x \in \mathcal{M}$, in the sense that every x can be well-approximated by a linear combination of at most $d+1$ vectors from the dictionary $\Phi_{2^{-2j}}$ formed by $\{c_{j,k}\}_{k=1}^{N(j)}$ and the union of the bases of $V_{j,k},\ k = 1 \dots N(j)$. Furthermore, operators efficiently encoding the "difference" between P_j and P_{j+1} were constructed, leading to a multiscale compressible representation of $\mathcal{M}$.

In practice, Π is unknown and we only have access to the *training data* $\mathcal{X}_n = \{X_1, \dots, X_n\}$, which are assumed to be i.i.d. with distribution Π. In this case, operators P_j are replaced by their estimators

$$\widehat{P}_j(x) := \sum_{k=1}^{N(j)} I\{x \in C_{j,k}\} \widehat{P}_{j,k}(x)$$

where $\{C_{j,k}\}_{k=1}^{N(j)}$ is a suitable partition of $\mathbb{R}^D$ *obtained from the data*,

$$\begin{aligned} \widehat{P}_{j,k}(x) &:= \widehat{c}_{j,k} + \mathrm{Proj}_{\widehat{V}_{j,k}}(x - \widehat{c}_{j,k}), \\ \widehat{c}_{j,k} &:= \frac{1}{|\mathcal{X}_{j,k}|} \sum_{x \in \mathcal{X}_{j,k}} x, \\ \widehat{V}_{j,k} &:= \operatorname*{argmin}_{\dim(V)=d} \frac{1}{|\mathcal{X}_{j,k}|} \sum_{x \in \mathcal{X}_{j,k}} \|x - \widehat{c}_{j,k} - \mathrm{Proj}_V(x - \widehat{c}_{j,k})\|^2, \end{aligned} \tag{4}$$

$\mathcal{X}_{j,k} = C_{j,k} \cap \mathcal{X}_n$, and $|\mathcal{X}_{j,k}|$ denotes the number of elements in $\mathcal{X}_{j,k}$. We shall call these $\widehat{P}_j$ the *empirical GMRA*.

Moreover, the dictionary $\widehat{\Phi}_{2^{-2j}}$ is formed by $\{\hat{c}_{j,k}\}_{k=1}^{N(j)}$ and the union of bases of $\hat{V}_{j,k},\ k = 1 \dots N(j)$. The "encoding" and "decoding" operators $\widehat{\mathcal{D}}_{2^{-2j}}$ and $\widehat{\mathcal{D}}^{-1}_{2^{-2j}}$ mentioned above are now defined in the obvious way, so that $\widehat{\mathcal{D}}^{-1}_{2^{-2j}} \widehat{\mathcal{D}}_{2^{-2j}}(x) = \widehat{P}_{j,k}(x)$ for any $x \in C_{j,k}$.

We remark that the "intrinsic dimension" d is assumed to be known throughout this paper. In practice, it can be estimated within the GMRA construction using the "multiscale SVD" ideas of Little et al. (2012, 2009). The estimation technique is based on inspecting (for a given point $x \in C_{j,k}$) the behavior of the singular values of the covariance matrix $\Sigma_{j,k}$ as j varies. For alternative methods, see Camastra and Vinciarelli (2001); Levina and Bickel (2004) and references therein and in the review section of Little et al. (2012).

3. Main Results

Our main goal is to obtain *probabilistic*, non-asymptotic bounds on the performance of the *empirical* GMRA under certain structural assumptions on the underlying distribution of the data. In practice, the data rarely belongs precisely to a smooth low-dimensional submanifold. One way to relax this condition is to assume that it is "sufficiently close" to a reasonably regular set. Here we assume that the underlying distribution is supported in a thin tube around a manifold. We may interpret the displacement from the manifold as noise, in which case we are making no assumption on distribution of the noise besides boundedness. Another way to model this situation is to allow *additive noise*, whence the observations are assumed to be of the form $X = Y + \xi$, where Y belongs to a submanifold of $\mathbb{R}^D$, ξ is independent of Y, and the distribution of ξ is known. This leads to a singular deconvolution problem (see Genovese et al., 2012b; Koltchinskii, 2000). Our assumptions however may also be interpreted as relaxing the "manifold assumption": even in the absence of noise we do allow data to be not exactly supported on a manifold. Our results will elucidate how the error of sparse approximation via GMRA depends on the "thickness" of the tube, which quantifies stability and robustness properties of our algorithm.

As we mentioned before, our GMRA construction is entirely data-dependent: it takes the point cloud of cardinality n as an input and for every $j \in \mathbb{Z}_+$ returns the partition $\{C_{j,k}\}_{k=1}^{N(j)}$ and associated affine projectors $\widehat{P}_{j,k}$. We will measure performance of the empirical GMRA by the $L_2(\Pi)$-error

$$\mathbb{E}\left\|X - \widehat{P}_j(X)\right\|^2 := \int\limits_{\mathrm{supp}(\Pi)} \left\|x - \widehat{P}_j(x)\right\|^2 d\Pi(x) \tag{5}$$

or by the $\|\cdot\|_{\infty,\Pi}$-error defined as

$$\left\|\mathrm{Id} - \widehat{P}_j\right\|_{\infty,\Pi} := \sup_{x \in \mathrm{supp}(\Pi)} \left\|x - \widehat{P}_j(x)\right\|, \tag{6}$$

where $\widehat{P}_j$ is defined by (4). Note, in particular, that these errors are "out-of-sample", i.e. measure the accuracy of the GMRA representations on all possible samples, not just those used to train the GMRA, which would not correspond to a learning problem.

The presentation is structured as follows: we start from the natural decomposition

$$\left\|x - \widehat{P}_j(x)\right\| \leq \underbrace{\|x - P_j(x)\|}_{\text{approximation error}} + \underbrace{\left\|P_j(x) - \widehat{P}_j(x)\right\|}_{\text{random error}}$$

and state the general conditions on the underlying distribution and partition scheme that suffice to guarantee that

1. the distribution-dependent operators P_j yield good approximation, as measured by $\mathbb{E}\|x - P_j(x)\|^2$: this is the bias (squared) term, which is non-random;

2. the empirical version $\widehat{P}_j$ is with high probability close to P_j, so that $\mathbb{E}\left\|\widehat{P}_j(x) - P_j(x)\right\|^2$ is small (with high probability): this is the variance term, which is random.

This leads to our first result, Theorem 2, where the error $\mathbb{E}\left\|x-\widehat{P}_j(x)\right\|^2$ of the empirical GMRA is bounded with high probability.

We will state this first result in a rather general setting (assumptions A1-A4) below), and after developing this general result, we consider the special but important case where the distribution Π generating the data is supported in thin tube around a smooth submanifold, and for a (concrete, efficiently computable, online) partition scheme we show that the conditions of Theorem 2 are satisfied. This is summarized in the statement of Theorem 7, that may be interpreted as proving finite-sample bounds for our GMRA-based dictionary learning scheme for high-dimensional data that suitably concentrates around a manifold. It is important to note that most of the constants in our results are explicit. The only geometric parameters involved in the bounds are the dimension d of the manifold (but not the ambient dimension D), its *reach* (see τ in (9)) and the "tube thickness" σ.

Among the existing literature, the papers Allard et al. (2012); Chen et al. (2012) introduced the idea of using multiscale geometric decomposition of data to estimate the distribution of points sampled in high-dimensions. However in the first paper no finite sample analysis was performed, and in the second the connection with geometric properties of the distribution of the data is not made explicit, with the conditions are expressed in terms of certain approximation spaces within the space of probability distributions in $\mathbb{R}^D$, with Wasserstein metrics used to measure distances and approximation errors.

The recent paper by Canas et al. (2012) is close in scope to our work; its authors present probabilistic guarantees for approximating a manifold with a global solution of the so-called k-flats (Bradley and Mangasarian, 2000) problem in the case of distributions supported on manifolds. It is important to note, however, that our estimator is explicitly and efficiently computable, while exact global solution of k-flats is usually unavailable and certain approximations have to be used in practice, with convergence to a global minimum is conditioned on suitable unknown initializations. In practice it is often reported that there exist many local minima with very different values, and good initializations are not trivial to find. In this work we obtain better convergence rates, with fast algorithms, and we also seamlessly tackle the case of noise and model error, which is beyond what was studied previously. We consider this development extremely relevant in applications, both because real data is corrupted by noise and the assumption that data lies exactly on a smooth manifold is often unrealistic. A more detailed comparison of theoretical guarantees for k-flats and for our approach is given after we state the main results in Subsection 3.2 below.

Another body of literature connected to this work studies the complexity of dictionary learning. For example, Gribonval et al. (2013) present general bounds for the convergence of global minimums of empirical risks in dictionary learning optimization problems (those results build on and generalize the works of Maurer and Pontil (2010); Vainsencher et al. (2011), among several others). While the rates obtained in those works seem to be competitive with our rates in certain regimes, the fact that their bounds must hold over entire families of dictionaries implies that those error rates generally involve a scaling constant of the order $\sqrt{Dk}$, where D is the ambient dimension and k is the number of "atoms" in the dictionary. Our bounds are independent of the ambient dimension D but implicitly include terms which depend upon the number of our "atoms." It should be noted that the number

of atoms in the dictionary learned by GMRA increase so as to approximate the fine structure of a dataset with more precision. As such, our attainment of the minimax lower bounds for manifold estimation in the Hausdorff metric (obtained in (Genovese et al., 2012a)) should be expected. While dictionaries produced from dictionary learning should reveal the fine structure of a dataset through careful examination of the representations they induce, these representations are often ambiguous unless additional structure is imposed on both the dictionaries and the datasets. On the other hand, the GMRA construction induces completely unambiguous sparse representations that can be used in regression and classification tasks with confidence.

In the course of the proof, we obtain several results that should be of independent interest. In particular, Lemma 18 gives upper and lower bounds for the volume of the tube around a manifold in terms of the reach (7) and tube thickness. While the exact tubular volumes are given by Weyl's tube formula (see Gray, 2004), our bound are exceedingly easy to state in terms of simple global geometric parameters.

For the details on numerical implementation of GMRA and its modifications, see the works by Allard et al. (2012); Chen and Maggioni (2010).

3.1 Finite Sample Bounds for Empirical GMRA

In this section, we shall present the finite sample bounds for the empirical GMRA described above. For a fixed resolution level j, we first state sufficient conditions on the distribution Π and the partition $\{C_{j,k}\}_{k=1}^{N(j)}$ for which these $L_2(\Pi)$-error bounds hold (see Theorem 2 below).

Suppose that for all integers $j_{\min} \le j \le j_{\max}$ the following is true:

(A1) There exists an integer $1 \le d \le D$ and a positive constant $\theta_1 = \theta_1(\Pi)$ such that for all $k = 1, \dots, N(j)$,

$$\Pi(C_{j,k}) \ge \theta_1 2^{-jd}\,.$$

(A2) There is a positive constant $\theta_2 = \theta_2(\Pi)$ such that for all $k = 1, \dots, N(j)$, if X is drawn from $\Pi_{j,k}$ then, Π - almost surely,

$$\|X - c_{j,k}\| \le \theta_2 2^{-j}\,.$$

(A3) Let $\lambda_1^{j,k} \ge \ldots \ge \lambda_D^{j,k} \ge 0$ denote the eigenvalues of the covariance matrix $\Sigma_{j,k}$ (defined in (3)). Then there exist $\sigma = \sigma(\Pi) \ge 0$, $\theta_3 = \theta_3(\Pi)$, $\theta_4 = \theta_4(\Pi) > 0$, and some $\alpha > 0$ such that for all $k = 1 \dots N(j)$,

$$\lambda_d^{j,k} \ge \theta_3 \frac{2^{-2j}}{d} \quad \text{and} \quad \sum_{l=d+1}^{D} \lambda_l^{j,k} \le \theta_4(\sigma^2 + 2^{-2(1+\alpha)j}) \le \frac{1}{2}\lambda_d^{j,k}.$$

If in addition

(A4) There exists $\theta_5 = \theta_5(\Pi)$ such that

$$\|\mathrm{Id} - P_j\|_{\infty,\Pi} \le \theta_5 \left(\sigma + 2^{-(1+\alpha)j}\right),$$

then the bounds are also guaranteed to hold for the $\|\cdot\|_{\infty,\Pi}$-error (6).

Remark 1

i. *Assumption* (A1) *entails that the distribution assigns a reasonable amount of probability to each partition element, assumption* (A2) *ensures that samples from partition elements are always within a ball around the centroid, and assumption* (A3) *controls the effective dimensionality of the samples within each partition element. Assumption* (A4) *just assumes a bound on the error for the theoretical GMRA reconstruction.*

ii. *Note that the constants* $\theta_i,\ i = 1 \dots 4$, *are independent of the resolution level* j.

iii. *It is easy to see that Assumption (A3) implies a bound on the "local approximation error": since* P_j *acts on* $C_{j,k}$ *as an affine projection on the first* d *"principal components", we have*

$$\begin{aligned}\mathbb{E}_{j,k}\|x - P_j(x)\|^2 &= \operatorname{tr}\left[\mathbb{E}_{j,k}\left(x - c_{j,k} - \operatorname{Proj}_{V_{j,k}}(x)\right)(x - c_{j,k} - \operatorname{Proj}_{V_{j,k}}(x))^T\right] \\ &= \sum_{l=d+1}^{D} \lambda_l^{j,k} \le \theta_4(\sigma^2 + 2^{-2(1+\alpha)j}).\end{aligned}$$

iv. *The parameter* σ *is introduced to cover "noisy" models, including the situations when* Π *is supported in a thin tube of width* σ *around a low-dimensional manifold* $\mathcal{M}$. *Whenever* Π *is supported on a smooth* d*-dimensional manifold,* σ *can be taken to be* 0.

v. *The stipulation*

$$\theta_4(\sigma^2 + 2^{-2(1+\alpha)j}) \le \frac{1}{2}\lambda_d^{j,k}$$

guarantees that the spectral gap $\lambda_d^{j,k} - \lambda_{d+1}^{j,k}$ *is sufficiently large.*

We are in position to state our main result.

Theorem 2 *Suppose that* ***(A1)-(A3)*** *are satisfied, let* $X, X_1, \dots, X_n$ *be an i.i.d. sample from* Π, *and set* $\bar{d} := 4d^2\theta_2^4/\theta_3^2$. *Then for any* $j_{\min} \le j \le j_{\max}$ *and any* $t \ge 1$ *such that* $t + \log(\bar{d} \vee 8) \le \frac{1}{2}\theta_1 n 2^{-jd}$,

$$\mathbb{E}\|X - \widehat{P}_j(X)\|^2 \le 2\theta_4\left(\sigma^2 + 2^{-2j(1+\alpha)}\right) + c_1 2^{-2j}\frac{(t + \log(\bar{d} \vee 8))d^2}{n2^{-jd}},$$

and if in addition ***(A4)*** *is satisfied,*

$$\left\|\mathrm{Id} - \widehat{P}_j\right\|_{\infty,\Pi} \le \theta_5\left(\sigma + 2^{-(1+\alpha)j}\right) + \sqrt{\frac{c_1}{2}2^{-2j}\frac{(t + \log(\bar{d} \vee 8))d^2}{n2^{-jd}}}$$

with probability $\ge 1 - \frac{2^{jd+1}}{\theta_1}\left(e^{-t} + e^{-\frac{\theta_1}{16}n2^{-jd}}\right)$, *where* $c_1 = 2\left(12\sqrt{2}\frac{\theta_2^3}{\theta_3\sqrt{\theta_1}} + 4\sqrt{2}\frac{\theta_2}{d\sqrt{\theta_1}}\right)^2$.

3.2 Distributions Concentrated near Smooth Manifolds

Of course, the statement of Theorem 2 has little value unless assumptions **(A1)-(A4)** can be verified for a rich class of underlying distributions. We now introduce an important class of models and an algorithm to construct suitable partitions $\{C_{j,k}\}$ which together satisfy these assumptions. Let $\mathcal{M}$ be a smooth (or at least C^2, so changes of coordinate charts admit continuous second-order derivatives), closed d-dimensional submanifold of $\mathbb{R}^D$. We recall the definition of the *reach* (see Federer, 1959), an important global characteristic of $\mathcal{M}$. Let

$$D(\mathcal{M}) = \{y \in \mathbb{R}^D : \exists! x \in \mathcal{M} \text{ s.t. } \|x - y\| = \inf_{z\in\mathcal{M}}\|z - y\|\}, \tag{7}$$

$$\mathcal{M}_r = \{y \in \mathbb{R}^D : \inf_{x\in\mathcal{M}}\|x - y\| < r\}. \tag{8}$$

Then

$$\operatorname{reach}(\mathcal{M}) := \sup\{r \ge 0 :\ \mathcal{M}_r \subseteq D(\mathcal{M})\}, \tag{9}$$

and we shall always use τ to denote the reach of the manifold $\mathcal{M}$.

Definition 3 *Assume that* $0 \le \sigma < \tau$. *We shall say that the distribution* Π *satisfies the* $(\boldsymbol{\tau}, \boldsymbol{\sigma})$**-model assumption** *if there exists a smooth (or at least* C^2*), compact submanifold* $\mathcal{M} \hookrightarrow \mathbb{R}^D$ *with reach* τ *such that* $\operatorname{supp}(\Pi) = \mathcal{M}_\sigma$, Π *and* $\mathcal{U}_{\mathcal{M}_\sigma}$ *(the uniform distribution on* $\mathcal{M}_\sigma$*) are absolutely continuous with respect to each other, and so Radon-Nikodym derivative* $\frac{d\Pi}{dU_{\mathcal{M}_\sigma}}$ *satisfies*

$$0 < \phi_1 \le \frac{d\Pi}{d\mathcal{U}_{\mathcal{M}_\sigma}} \le \phi_2 < \infty \qquad \mathcal{U}_{\mathcal{M}_\sigma}\text{- almost surely.} \tag{10}$$

Example 1 *Consider the unit sphere of radius* R *in* $\mathbb{R}^D$, $\mathcal{S}_R$. *Then* $\tau = R$ *for this manifold, and for any* $\sigma < R$, *the uniform distribution on the set* $B(0, R + \sigma) \setminus B(0, R - \sigma)$ *satisfies the* (σ, τ)*-model assumption. On the other hand, taking the uniform distribution on a* σ*-thickening of the union of two line segments emanating from the origin produces a distribution which does not satisfy the* (σ, τ) *model assumption. In particular,* $\tau = 0$ *for the underlying manifold.*

Remark 4 *We will implicitly assume that constants* ϕ_1 *and* ϕ_2 *do not depend on the ambient dimension* D *(or depend on a slowly growing function of* D, *such as* $\log D$*) - the bound of Theorem 7 shows that this is the "interesting case". On the other hand, we often do not need the full power of* (τ, σ) *- model assumption, see the Remark 9 after Theorem 7.*

Our partitioning scheme is based on the data structure known as the *cover tree* introduced by Beygelzimer et al. (2006) (see also Ciaccia et al., 1997; Karger and Ruhl, 2002; Yianilos, 1993). We briefly recall its definition and basic properties. Given a set of n distinct points $S_n = \{x_1, \dots, x_n\}$ in some metric space (S, ρ), the cover tree T on S_n satisfies the following: let $T_j \subset S_n,\ j = 0, 1, 2, \dots$ be the set of nodes of T at level j. Then

1. $T_j \subset T_{j+1}$;

2. for all $y \in T_{j+1}$, there exists $z \in T_j$ such that $\rho(y,z) < 2^{-j}$;

3. for all $y, z \in T_j$, $\rho(y,z) > 2^{-j}$.

Remark 5 *Note that these properties imply the following: for any $y \in S_n$, there exists $z \in T_j$ such that $\rho(y,z) < 2^{-j+1}$.*

Theorem 3 in (Beygelzimer et al., 2006) shows that the cover tree always exists; for more details, see the aforementioned paper.

We will construct a cover tree for the collection $X_1, \ldots, X_n$ of i.i.d. samples from the distribution Π with respect to the Euclidean distance $\rho(x,y) := \|x-y\|$. Assume that $T_j := T_j(X_1, \ldots, X_n) = \{a_{j,k}\}_{k=1}^{N(j)}$. Define the indexing map

$$k(x) := \operatorname*{argmin}_{1 \le k \le N(j)} \|x - a_{j,k}\|$$

(ties are broken by choosing the smallest value of k), and partition $\mathbb{R}^D$ into the Voronoi regions

$$C_{j,k} = \{x \in \mathbb{R}^D : k_j(x) = k\}. \tag{11}$$

Let $\varepsilon(n,t)$ be the smallest $\varepsilon > 0$ which satisfies

$$n \ge \frac{1}{\phi_1} \left(\frac{\tau+\sigma}{\tau-\sigma}\right)^d \beta_1 \left(\log \beta_2 + t\right), \tag{12}$$

where $\beta_1 = \frac{\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}{\cos^d(\delta_1)\mathrm{Vol}(B_d(0,\varepsilon/4))}$, $\beta_2 = \frac{\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}{\cos^d(\delta_2)\mathrm{Vol}(B_d(0,\varepsilon/8))}$, $\delta_1 = \arcsin(\varepsilon/8\tau)$, and $\delta_2 = \arcsin(\varepsilon/16\tau)$.

Remark 6 *For large enough n, this requirement translates into $n \ge C(\mathcal{M}, d, \phi_1)\left(\frac{1}{\varepsilon}\right)^d \left(\log \frac{1}{\varepsilon} + t\right)$ for some constant $C(\mathcal{M}, d, \phi_1)$.*

We are ready to state the main result of this section.

Theorem 7 *Suppose that Π satisfies the (τ,σ)-model assumption. Let $X_1, \ldots, X_n$ be an i.i.d. sample from Π, construct a cover tree T from $\{X_i\}_{i=1}^n$, and define $C_{j,k}$ as in (11). Assume that $\varepsilon(n,t) < \sigma$. Then, for all $j \in \mathbb{Z}_+$ such that $2^{-j} > 8\sigma$ and $3 \cdot 2^{-j} + \sigma < \tau/8$, the partition $\{C_{j,k}\}_{k=1}^{N(j)}$ and Π satisfy **(A1)**, **(A2)**, **(A3)**, and **(A4)** with probability $\ge 1 - e^{-t}$*

for

$$\theta_1 = \frac{\phi_1 \mathrm{Vol}(B_d(0,1))}{2^{4d}\,\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}\left(\frac{\tau-\sigma}{\tau+\sigma}\right)^d,$$
$$\theta_2 = 12,$$
$$\theta_3 = \frac{\phi_1/\phi_2}{2^{4d+8}\left(1+\frac{\sigma}{\tau}\right)^d},$$
$$\theta_4 = 2 \vee \frac{2^3 3^4}{\tau^2},$$
$$\theta_5 = \left(2 \vee \frac{2^2 3^2}{\tau}\right)\left(1 + 3\cdot 2^5\sqrt{2d}\left(1+\frac{\sigma}{\tau}\right)^{d/2}\left(\frac{1+\left(\frac{25}{71}\right)^2}{1-\frac{1}{9\cdot 2^{12}}}\right)^{d/4}\right),$$
$$\alpha = 1.$$

One may combine the results of Theorem 7 and Theorem 2 as follows: given an i.i.d. sample $X_1, \ldots, X_n$ from Π, use the first $\lceil \frac{n}{2} \rceil$ points $\{X_1, \ldots, X_{\lceil \frac{n}{2} \rceil}\}$ to obtain the partition $\{C_{j,k}\}_{k=1}^{N(j)}$, while the remaining $\{X_{\lceil \frac{n}{2} \rceil + 1}, \ldots, X_n\}$ are used to construct the operator $\hat{P}_j$ (see (4)). This makes our GMRA construction entirely (cover tree, partitions, affine linear projections) data-dependent. We observe that since our approximations are piecewise linear, they are insensitive to regularity of the manifold beyond first order, so the estimates saturate at $\alpha = 1$.

When σ is very small or equal to 0, the bounds resulting from Theorem 2 can be "optimized" over j to get the following statement (we present only the bounds for the $L_2(\Pi)$ error, but the results $\|\cdot\|_{\infty,\Pi}$ are similar).

Corollary 8 *Assume that conditions of Theorem 7 hold, and that n is sufficiently large. Then for all $A \ge 1$ such that $A \log n \le c_4 n$, the following holds:*

(a) *if $d \in \{1,2\}$,*

$$\inf_{j \in \mathbb{Z}: 2^{-j} < \tau/24} \mathbb{E}\|x - \widehat{P}_j(x)\|^2 \le C_1 \left(\frac{\log n}{n}\right)^{\frac{2}{d}};$$

(b) *if $d \ge 3$,*

$$\inf_{j \in \mathbb{Z}: 2^{-j} < \tau/24} \mathbb{E}\|x - \widehat{P}_j(x)\|^2 \le C_2 \left(\frac{\log n}{n}\right)^{\frac{4}{d+2}} \tag{13}$$

with probability $\ge 1 - c_3 n^{-A}$, where C_1 and C_2 depend only on $A, \tau, d, \phi_1/\phi_2, \mathrm{Vol}_{\mathcal{M}}(\mathcal{M})$ and c_3, c_4 depend only on $\tau, d, \phi_1/\phi_2, \mathrm{Vol}_{\mathcal{M}}(\mathcal{M})$.

Proof In case (a), it is enough to set $t := (A+1)\log n$, $2^{-j} := \left(\frac{16t}{\theta_1 n}\right)^{1/d}$, and apply Theorem 2. For case (b), set $t := (A+1)\log n$ and $2^{-j} := \left(\frac{A \log n}{n}\right)^{\frac{1}{d+2}}$. ■

Finally, we note that the claims *ii.* and *iii.* stated in the beginning of Section 2 easily follow

from our general results (it is enough to choose n such that $\varepsilon \simeq n^{-\frac{2}{d+2}}$ and $2^{-j} = \sqrt{\varepsilon}$). Claim *i.* follows from assumption **(A1)** and Theorem 7. Computational complexity bounds *iv.* follow from the associated computational cost estimates for the cover trees algorithm and the randomized singular value decomposition, and are discussed in detail in Sections 3 and 8 of (Allard et al., 2012).

Remark 9 *It follows from our proof that it is sufficient to assume a weaker (but somewhat more technical) form of (τ, σ)-model condition for the conclusion of Theorem 7 to hold. Namely, let $\tilde{\Pi}$ be the pushforward of Π under the projection $\mathrm{Proj}_{\mathcal{M}} : \mathcal{M}_\sigma \to \mathcal{M}$, and assume that there exists $\tilde{\phi}_1 > 0$ such that for any measurable $A \subseteq \mathcal{M}$*

$$\tilde{\Pi}(A) := \Pi\left(\mathrm{Proj}_{\mathcal{M}}^{-1}(A)\right) \geq \tilde{\phi}_1 U_{\mathcal{M}}(A).$$

Moreover, suppose that there exists $\tilde{\phi}_2 > 0$ such that for any $y \in \mathcal{M}$, any set $A \subset \mathcal{M}_\sigma$ and any $\tau > r \geq 2\sigma$ such that $B(y,r) \cap \mathcal{M}_\sigma \subseteq A \subseteq B(y, 12r)$, we have

$$\Pi(A) \leq \tilde{\phi}_2 \mathcal{U}_{\mathcal{M}_\sigma}(A).$$

In some circumstances, checking these two conditions is not hard (e.g., when $\mathcal{M}$ is a sphere, Y is uniformly distributed on $\mathcal{M}$, η is spherically symmetric "noise" independent of Y and such that $\|\eta\| \leq \sigma$, and Π is the distribution of $Y + \eta$), but (τ, σ) - assumption does not need to hold with constants ϕ_1 and ϕ_2 independent of D.

3.3 Connections to Previous Work and Further Remarks

It is useful to compare our rates with results of Theorem 4 in (Canas et al., 2012). In particular, this theorem implies that, given a sample of size n from the Borel probability measure Π on the smooth d-dimensional manifold $\mathcal{M}$, the $L_2(\Pi)$-error of approximation of $\mathcal{M}$ by $k_n = C_1(\mathcal{M}, \Pi) n^{d/(2(d+4))}$ affine subspaces is bounded by $C_2(\mathcal{M}, \Pi) n^{-2/(d+4)}$. Here, the dependence of k_n on n is "optimal" in a sense that it minimizes the upper bound for the risk obtained in (Canas et al., 2012). If we set $\sigma = 0$ in our results, then it easily follows from Theorems 7 and 2 that the $L_2(\Pi)$-error achieved by our GMRA construction for $2^j \simeq n^{\frac{1}{2(d+4)}}$ (so that $N(j) \simeq k_n$ to make the results comparable) is of the same order $n^{-\frac{2}{d+4}}$. However, this choice of j is not optimal in this case - in particular, setting $2^{j_n} \simeq n^{\frac{1}{d+2}}$, we obtain as in (13) a $L_2(\Pi)$-error of order $n^{-\frac{2}{d+2}}$, which is a faster rate. Moreover, we also obtain results in the sup norm, and not only for mean square error. We should note that technically our results require the stronger condition (10) on the underlying measure Π, while theoretical guarantees in (Canas et al., 2012) are obtained assuming only the upper bound $\frac{d\Pi}{dU_{\mathcal{M}}} \leq \phi_2 < \infty$, where $U_{\mathcal{M}} := \frac{d\,\mathrm{Vol}_{\mathcal{M}}}{\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}$ is the uniform distribution over $\mathcal{M}$.

The rate (13) is the same (up to log-factors) as the minimax rate obtained for the problem considered in (Genovese et al., 2012a) of estimating a manifold from the samples corrupted with the additive noise that is "normal to the manifold". Our theorems are stated under more general conditions, however, we only prove *robustness-type* results and do not address the problem of *denoising*. At the same time, the estimator proposed in (Genovese et al., 2012a) is (unlike our method) not suitable for applications. The paper (Genovese et al., 2012b) considers (among other problems) the noiseless case of manifold estimation

under Hausdorff loss, and obtains the minimax rate of order $n^{-\frac{2}{d}}$. Performed numerical simulation (see Section 6) suggest that our construction also appears to achieve this rate in the noiseless case. However, our main focus is on the case $\sigma > 0$.

The work of Fefferman et al. establishes the sampling complexity of testing the hypothesis if an unknown distribution is close to being on a manifold (with known reach, volume, dimension) in the Mean Squared sense, is also related to the work discussed in this section, and to the present one. While our results do imply that if we have enough points, as prescribed by our main theorems, and the MSE does not decay as prescribed, then the data with high probability does not satisfy the geometric assumptions in the corresponding theorem, this is still different from the hypothesis testing problem. There may distributions not satisfying our assumptions, such that GMRA still yields good approximations: in fact we welcome and do not rule out these situations. Fefferman et al. also present an algorithm for constructing an approximation to the manifold; however such an algorithm does not seem easy to implement in practice. The emphasis in this work is on moving to a more general setting than the manifold setting, focusing on multiscale approaches that are robust (locally, because of SVD, as well as across scales), and fast, easily implementable algorithms.

We remark that we analyze the case of one manifold $\mathcal{M}$, and its "perturbation" in the sense of having a measure supported in a tube around $\mathcal{M}$. Our construction however is multiscale and in particular local. Many extensions are immediate, for example to the case of multiple manifolds (possibly of different dimensions) with non-intersecting tubes around them. The case of unbounded noise is also of interest: if the noise has sub-Gaussian tails then very few points are outside a tube of radius dependent on the sub-Gaussian moment, and these "outliers" are easily disregarded as there are few and far away, so they do not affect the construction and the analysis at fine scales. Another situation is when there are many gross outliers, for example points uniformly distributed in high-dimension in, say, a cube containing $\mathcal{M}$. But then the volume of such cube is so large that unless the number of points is huge (at least exponential in the ambient dimension D), almost all of these points are in fact far from each other and from $\mathcal{M}$ with very high probability, so that again they do affect the analysis and the algorithms. These are some of the advantages of the multiscale approach, which would otherwise have the potential of corrupting the results (or complicating the analysis of) other global algorithms, such as k-flats.

4. Preliminaries

This section contains the remaining definitions and preliminary technical facts that will be used in the proofs of our main results.

Given a point y on the manifold $\mathcal{M}$, let $T_y\mathcal{M}$ be the associated tangent space, and let $T_y^\perp \mathcal{M}$ be the orthogonal complement of $T_y\mathcal{M}$ in $\mathbb{R}^D$. We define the projection from the tube $\mathcal{M}_\sigma$ (see (8)) onto the manifold $\mathrm{Proj}_{\mathcal{M}} : \mathcal{M}_\sigma \to \mathcal{M}$ by

$$\mathrm{Proj}_{\mathcal{M}}(x) = \operatorname*{argmin}_{y \in \mathcal{M}} \|x - y\|$$

and note that $\sigma < \tau$, together with (7), implies that $\mathrm{Proj}_{\mathcal{M}}$ is well-defined on $\mathcal{M}_\sigma$, and

$$\mathrm{Proj}_{\mathcal{M}}(y + \xi) = y$$

whenever $y \in \mathcal{M}$ and $\xi \in T_y^\perp \mathcal{M} \cap B(0,\sigma)$.

Next, we recall some facts about the volumes of parallelotopes that will prove useful in Section 5. For a matrix $A \in \mathbb{R}^{k\times l}$ with $l \leq k$, we shall abuse our previous notation and let $\mathrm{Vol}(A)$ also denote the volume of the parallelotope formed by the columns of A. Let A and B be $k \times l_1$ and $k \times l_2$ matrices respectively with $l_1 + l_2 \leq k$, and note that

$$\mathrm{Vol}([A\,|\,B]) \leq \mathrm{Vol}(A)\mathrm{Vol}(B)$$

where $([A\,|\,B])$ denotes the concatenation of A and B into a $k \times (l_1 + l_2)$ matrix. Moreover, if the columns of A and B are all mutually orthogonal, we clearly have that $\mathrm{Vol}([A\,|\,B]) = \mathrm{Vol}(A)\mathrm{Vol}(B)$. Assuming that I is the $l_1 \times l_1$ identity matrix, we have the bound $\mathrm{Vol}\begin{pmatrix} A \\ I \end{pmatrix} \geq 1$. The following proposition gives volume bounds for specific types of perturbations that we shall encounter.

Proposition 10 *Suppose $Y = [y_1|\cdots|y_d]$ is symmetric d by d matrix such that $\|Y\| \leq q < 1$. Then*

$$\begin{aligned} \mathrm{Vol}\begin{pmatrix} I+Y \\ X \end{pmatrix} &\leq (1+q)^d \,\mathrm{Vol}\begin{pmatrix} I \\ X \end{pmatrix} \\ \mathrm{Vol}\begin{pmatrix} I+Y & X^T \\ X & -I \end{pmatrix} &\geq (1-q)^d \,\mathrm{Vol}\begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}. \end{aligned}$$

This proof (as well as the proofs of our other supporting technical contributions) is given in the Appendix. Finally, let us recall several important geometric consequences involving the reach:

Proposition 11 *The following holds:*

i. For all $x, y \in \mathcal{M}$ such that $\|x - y\| \leq \tau/2$, we have

$$d_{\mathcal{M}}(x,y) \leq \tau - \tau\sqrt{1 - 2\frac{\|x-y\|}{\tau}} \leq 2\|x-y\|.$$

ii. Let $\gamma(t) : [0,1] \mapsto \mathcal{M}$ be the arclength-parameterized geodesic. Then $\|\gamma''(t)\| \leq \frac{1}{\tau}$ for all t.

iii. Let ϕ be the angle between $T_x\mathcal{M}$ and $T_y\mathcal{M}$, in other words,

$$\cos(\phi) := \min_{u\in T_x\mathcal{M}, \|u\|=1} \max_{v\in T_y\mathcal{M}, \|v\|=1} |\langle u, v\rangle|.$$

If $\|x-y\| \leq \frac{\tau}{2}$, then $\cos(\phi) \geq \sqrt{1 - 2\frac{\|x-y\|}{\tau}}$.

iv. If x is such that $\|x-y\| < \tau/2$, then x is a regular point of $\mathrm{Proj}_{y+T_y\mathcal{M}} : \mathcal{B}(y, \tau/2) \cap \mathcal{M} \to y + T_y\mathcal{M}$ (in other words, the Jacobian of $\mathrm{Proj}_{y+T_y\mathcal{M}}$ at x is nonsingular).

v. Let $y \in \mathcal{M}$, $r < \tau$ and $A = \mathcal{M} \cap B(y,r)$. Then

$$B_d(y, r\cos(\theta)) \subseteq \mathrm{Proj}_{y+T_y\mathcal{M}}(A),$$

where $\theta = \arcsin\left(\frac{r}{2\tau}\right)$.

Proof Part *i.* is the statement of Proposition 6.3 and part *ii.* - of Proposition 6.1 in (Niyogi et al., 2008). Part *iii.* is demonstrated in Lemma 5.4 of the same paper, and this lemma coincides with *iv.* Part *v.* is proven in Lemma 5.3 of (Niyogi et al., 2008). ■

5. Proofs of the Main Results

The rest of the paper is devoted to the proofs of our main results.

5.1 Overview of the Proofs

We begin by providing an overview of the main steps of the proofs to aid comprehension. The proof of Theorem 2 begins by invoking the bias-variance decomposition:

$$\|x - \widehat{P}_j(x)\|^2 \leq 2\|x - P_j(x)\|^2 + 2\|P_j(x) - \widehat{P}_j(x)\|^2.$$

Remark 1, part *iii.* and the decomposition

$$\mathbb{E}\|X - P_j(X)\|^2 = \sum_{k=1}^{N(j)} \Pi(C_{j,k})\mathbb{E}_{j,k}\|X - P_j(X)\|^2$$

gives us the first term in the bound of Theorem 2. Note that this contribution is deterministic.

The next step in the proof is to bound the stochastic error $\mathbb{E}\|P_j(x) - \widehat{P}_j(x)\|^2$ with high probability. We start with the bound

$$\begin{aligned} \|P_j(x) - \widehat{P}_j(x)\| &= \|c_{j,k} - \widehat{c}_{j,k} + \mathrm{Proj}_{V_{j,k}}(x - c_{j,k}) - \mathrm{Proj}_{\widehat{V}_{j,k}}(x - c_{j,k} + c_{j,k} - \widehat{c}_{j,k})\| \quad (14) \\ &\leq 2\|c_{j,k} - \widehat{c}_{j,k}\| + \|\mathrm{Proj}_{V_{j,k}} - \mathrm{Proj}_{\widehat{V}_{j,k}}\| \cdot \|x - c_{j,k}\|. \quad (15) \end{aligned}$$

for $x \in C_{j,k}$. We then use concentration of measure results (matrix Bernstein-type inequality) to bound the terms

$$\|c_{j,k} - \widehat{c}_{j,k}\| \text{ and } \left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\|$$

with high probability. The latter bound and Assumption **(A3)** allows us to invoke Theorem 14 to obtain a bound of the form

$$\|\mathrm{Proj}_{V_{j,k}} - \mathrm{Proj}_{\widehat{V}_{j,k}}\| \leq C\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\|.$$

Finally, the term $\|x - c_{j,k}\|$ is controlled by Assumption **(A2)**.

The proof of Theorem 7 is primarily supported by a volume comparison theorem that allows for the cancellation of the "noisy" terms that would imply dependency on D. That is, supposing that $\mathrm{Proj}_{\mathcal{M}} : \mathcal{M}_\sigma \to \mathcal{M}$ is the projection from the σ-tubular neighborhood onto the underlying manifold with reach τ, if $U \subset \mathcal{M}$ is $\mathrm{Vol}_{\mathcal{M}}$-measurable with $\mathrm{Vol}_{\mathcal{M}}(U) > 0$, we have that

$$\left(1 - \frac{\sigma}{\tau}\right)^d \leq \frac{\mathrm{Vol}(\mathrm{Proj}_{\mathcal{M}}^{-1}(U))}{\mathrm{Vol}_{\mathcal{M}}(U)\mathrm{Vol}(B_{D-d}(0,\sigma))} \leq \left(1 + \frac{\sigma}{\tau}\right)^d.$$

This is encapsulated in Lemma 18. This allows us to relate probabilities on the tubular neighborhood with probabilities on the manifold itself, which only involve d-dimensional volumes.

The first thing that this allows us to do is to ensure that a sufficiently large sample from $\mathcal{U}_{\mathcal{M}_\sigma}$, $\{X_i\}_{i=1}^N$, has that $\{\mathrm{Proj}_{\mathcal{M}}(X_i)\}_{i=1}^N$ is an ε-net for $\mathcal{M}$. Running the cover tree algorithm at the appropriate scale and invoking the cover tree properties at this scale yields the constant for Assumption **(A2)**. Cover tree properties also ensure that each partition element contains a large enough portion of the tubular neighborhood, which we then relate to a portions of the manifold whose volume is comparable to d-dimensional Euclidean volumes. This approach provides the constant for Assumption **(A1)**. Finally, the constants from Assumption **(A3)** and **(A4)** are obtained from local moment estimates based upon these volume bounds.

Now, the volume comparison bounds themselves are proven by considering coordinate systems that locally invert orthogonal projections onto tangent spaces. The fact that the manifold has reach τ imposes bounds on the Jacobians and second-order terms for these local inversions. These bounds are ultimately used to bound volume distortions, and lead to the volume comparison result above.

5.2 Proof of Theorem 2

Assumption **(A3)** above controls the $L_2(\Pi)$ approximation error of $x \in M$ by $P_j(x)$ (see Remark 1, part *iii.*), hence we will concentrate on the stochastic error $\|\widehat{P}_j(x) - P_j(x)\|$. To this end, we will need to estimate $\|c_{j,k} - \widehat{c}_{j,k}\|$ and $\|\mathrm{Proj}_{V_{j,k}} - \mathrm{Proj}_{\widehat{V}_{j,k}}\|$, $k = 1 \dots N(j)$.

One of the main tools required to obtain this bound is the noncommutative Bernstein's inequality.

Theorem 12 *(Minsker, 2013, Theorem 2.1) Let $Z_1, \dots, Z_n \in \mathbb{R}^{D\times D}$ be a sequence of independent symmetric random matrices such that $\mathbb{E}Z_i = 0$ and $\|Z_i\| \le U$ a.s., $1 \le i \le n$. Let*

$$\sigma^2 := \left\| \sum_{i=1}^n \mathbb{E}Z_i^2 \right\|.$$

Then for any $t \ge 1$

$$\left\| \sum_{i=1}^n Z_i \right\| \le 2\max\left(\sigma\sqrt{t + \log(\bar{D})}, U(t + \log(\bar{D})) \right) \tag{16}$$

with probability $\ge 1 - e^{-t}$, where $\bar{D} := 4\frac{\mathrm{tr}\left(\sum_{i=1}^n \mathbb{E}Z_i^2\right)}{\sigma^2}$.

Note that we always have $\bar{D} \le 4D$. We use this inequality to estimate $\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\|$: let $\Pi(dx|A)$ be the conditional distribution of X given that $X \in A$, and set $\Pi_{j,k}(dx) := \Pi(dx|C_{j,k})$. Let $m_{j,k} := \sum_{i=1}^n I\{X_i \in C_{j,k}\}$ to be the number of samples in $C_{j,k}$, $k = 1 \dots N(j)$. Let $I \subset \{1, \dots, n\}$ be such that $|I| = m$. Conditionally on the event $A_I := \{X_i \in C_{j,k} \text{ for } i \in I, \text{ and } X_i \notin C_{j,k} \text{ for } i \notin I\}$, the random variables $\{X_i,\ i \in I\}$ are independent with distribution $\Pi_{j,k}$. Then

$$\begin{aligned}\Pr\left(\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\| \ge s \,|\, m_{j,k} = m \right) &= \sum_{I\subset\{1,\dots,n\}, |I|=m} \Pr\left(\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\| \ge s \,|\, A_I \right) \frac{1}{\binom{n}{m}} \\ &= \Pr\left(\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\| \ge s \,|\, A_{\{1,\dots,m\}} \right).\end{aligned} \tag{17}$$

To estimate $\Pr\left(\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\| \ge s \,|\, A_{\{1,\dots,m\}} \right)$, we use the following inequality. Recall that

$$\bar{d} = 4d^2\frac{\theta_2^4}{\theta_3^2},$$

where θ_2, θ_3 are the constants in Assumptions **(A2)** and **(A3)**.

Lemma 13 *Let $X, X_1, \dots, X_m$ be an i.i.d. sample from $\Pi_{j,k}$. Set*

$$\widehat{c}_{j,k} = \frac{1}{m}\sum_{i=1}^m X_i \quad \text{and} \quad \widehat{\Sigma}_{j,k} := \frac{1}{m}\sum_{i=1}^m (X_i - \widehat{c}_{j,k})(X_i - \widehat{c}_{j,k})^T.$$

Assume that $m \ge t + \log(\bar{d} \vee 8)$. Then with probability $\ge 1 - 2e^{-t}$,

$$\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\| \le 6r^2\sqrt{\frac{t + \log(\bar{d} \vee 8)}{m}}.$$

Proof We want to estimate

$$\begin{aligned}\left\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\right\| &= \left\| \frac{1}{m}\sum_{i=1}^m (X_i - c_{j,k})(X_i - c_{j,k})^T - \Sigma_{j,k} + (c_{j,k} - \widehat{c}_{j,k})(c_{j,k} - \widehat{c}_{j,k})^T \right\| \\ &\le \left\| \frac{1}{m}\sum_{i=1}^m (X_i - c_{j,k})(X_i - c_{j,k})^T - \Sigma_{j,k} \right\| + \left\| (c_{j,k} - \widehat{c}_{j,k})(c_{j,k} - \widehat{c}_{j,k})^T \right\|.\end{aligned} \tag{18}$$

Set $r := \theta_2 \cdot 2^{-j}$. Recall that $\|x - c_{j,k}\| \le r$ for all $x, y \in C_{i,j}$ by assumption **(A2)**. It implies that

1. for all $1 \le i \le m$, $\|(X_i - c_{j,k})(X_i - c_{j,k})^T\| \le r^2$ almost surely,
2. $\left\|\mathbb{E}\left[(X_i - c_{j,k})(X_i - c_{j,k})^T\right]^2\right\| = \left\|\mathbb{E}\|X_i - c_{j,k}\|^2(X_i - c_{j,k})(X_i - c_{j,k})^T\right\| \le r^2\|\Sigma_{j,k}\|$.

Therefore, by Theorem 12 applied to $Z_i := \frac{1}{m}(X_i - c_{j,k})(X_i - c_{j,k})^T$, $i = 1 \dots m$,

$$\begin{aligned}\left\| \frac{1}{m}\sum_{i=1}^m (X_i - c_{j,k})(X_i - c_{j,k})^T - \Sigma_{j,k} \right\| &\le 2\left(r\sqrt{\frac{(t + \log(\bar{d}))\|\Sigma_{j,k}\|}{m}} \vee r^2\frac{t + \log(\bar{d})}{m} \right) \\ &= 2r^2\sqrt{\frac{(t + \log(\bar{d}))}{m}}\left(\sqrt{\frac{t + \log(\bar{d})}{m}} \vee \sqrt{\left\|\frac{\Sigma_{j,k}}{r^2}\right\|} \right)\end{aligned}$$

with probability $\geq 1 - e^{-t}$. Note that $\|\Sigma_{j,k}\| \leq \operatorname{tr}(\Sigma_{j,k}) \leq r^2$. Moreover,

$$\bar{D} = 4\frac{\operatorname{tr}(\mathbb{E}Z_1^2)}{\|\mathbb{E}Z_1^2\|} \leq 4\frac{\mathbb{E}(\operatorname{tr} Z_1)^2}{\left(\lambda_d^{j,k}\right)^2} \leq 4d^2\frac{r^4}{\theta_3^2 2^{-4j}} = 4d^2\frac{\theta_2^4}{\theta_3^2} = \bar{d}$$

by assumption **(A3)** and the definition of r. Since $\frac{t+\log(\bar{d})}{m} \leq 1$ by assumption,

$$\left\|\frac{1}{m}\sum_{i=1}^{m}(X_i - c_{j,k})(X_i - c_{j,k}) - \Sigma_{j,k}\right\| \leq 2r^2\sqrt{\frac{t+\log(\bar{d})}{m}}.$$

For the second term in (18), note that $\left\|(c_{j,k} - \widehat{c}_{j,k})(c_{j,k} - \widehat{c}_{j,k})^T\right\| = \|c_{j,k} - \widehat{c}_{j,k}\|^2$. We apply Theorem 12 to the symmetric matrices

$$G_i := \begin{pmatrix} 0 & (X_i - c_{j,k})^T \\ X_i - c_{j,k} & 0 \end{pmatrix}.$$

Noting that $\|G_i\| = \|X_i - c_{j,k}\| \leq r$ almost surely,

$$\|\mathbb{E}G_i^2\| = \mathbb{E}\|X_i - c_{j,k}\|^2 = \operatorname{tr}(\Sigma_{j,k}) \leq r^2,$$

and $\frac{\operatorname{tr}(\mathbb{E}G_i^2)}{\|\mathbb{E}G_i^2\|} = 2$, we get that for all t such that $t + \log 8 \leq m$, with probability $\geq 1 - e^{-t}$

$$\|\widehat{c}_{j,k} - c_{j,k}\| \leq 2\left[r\sqrt{\frac{(t+\log 8)}{m}} \vee r\frac{t+\log 8}{m}\right] \leq 2r\sqrt{\frac{t+\log 8}{m}}, \tag{19}$$

hence with the same probability

$$\left\|(c_{j,k} - \widehat{c}_{j,k})(c_{j,k} - \widehat{c}_{j,k})^T\right\| \leq 4r^2\frac{t+\log 8}{m},$$

and the claim follows. ∎

Given the previous result, we can estimate the angle between the eigenspaces of $\widehat{\Sigma}_{j,k}$ and $\Sigma_{j,k}$:

Theorem 14 *(Davis and Kahan, 1970), or (Zwald and Blanchard, 2006, Theorem 3).*
Let $\delta_d = \delta_d(\Sigma_{j,k}) := \frac{1}{2}(\lambda_d^{j,k} - \lambda_{d+1}^{j,k})$. If $\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\| < \delta_d/2$, then

$$\left\|\operatorname{Proj}_{V_{j,k}} - \operatorname{Proj}_{\widehat{V}_{j,k}}\right\| \leq \frac{\|\widehat{\Sigma}_{j,k} - \Sigma_{j,k}\|}{\delta_d},$$

Since $\delta_d \geq \frac{\theta_3}{2\theta_2^2}\frac{r^2}{d}$ by assumption **(A3)**, the previous result implies that, conditionally on the event $\{m_{j,k} = m\}$, with probability $\geq 1 - 2e^{-t}$,

$$\left\|\operatorname{Proj}_{V_{j,k}} - \operatorname{Proj}_{\widehat{V}_{j,k}}\right\| \leq 12d\frac{\theta_2^2}{\theta_3}\sqrt{\frac{t+\log(\bar{d}\vee 8)}{m}}.$$

It remains to obtain the unconditional bound. Set $n_{j,k} := n\Pi(C_{j,k})$ and note that $n_{j,k} \geq \theta_1 n 2^{-jd}$ by assumption **(A1)**. To this end, we have

$$\begin{aligned}
&\Pr\left(\max_{k=1\dots N(j)}\left\|\operatorname{Proj}_{V_{j,k}} - \operatorname{Proj}_{\widehat{V}_{j,k}}\right\| \geq 12\frac{\theta_2^2}{\theta_3}\sqrt{\frac{(t+\log(\bar{d}\vee 8))d^2}{n_{j,k}/2}}\right) \\
&\leq \Pr\left(\max_{k=1\dots N(j)}\left\|\operatorname{Proj}_{V_{j,k}} - \operatorname{Proj}_{\widehat{V}_{j,k}}\right\| \geq 12\frac{\theta_2^2}{\theta_3}\sqrt{\frac{(t+\log(\bar{d}\vee 8))d^2}{n_{j,k}/2}}\,\middle|\, m_{j,k} \geq n_{j,k}/2,\ k = 1\dots N(j)\right) \\
&+ \Pr\left(\bigcup_{k=1}^{N(j)}\{m_{j,k} < n_{j,k}/2\}\right) \leq N(j)e^{-t} + \sum_{k=1}^{N(j)}\Pr\left(m_{j,k} < n_{j,k}/2\right).
\end{aligned}$$

Recall that $m_{j,k} = \sum_{i=1}^{n} I\{X_i \in C_{j,k}\}$, hence $\mathbb{E}m_{j,k} = n_{j,k}$ and $\operatorname{Var}(m_{j,k}) \leq n_{j,k}$. Bernstein's inequality (see Lemma 2.2.9 in van der Vaart and Wellner, 1996) implies that

$$|m_{j,k} - n_{j,k}| \leq \left(2\sqrt{sn_{j,k}} \vee \frac{4}{3}s\right)$$

with probability $\geq 1 - e^{-s}$. Choosing $s = \frac{n_{j,k}}{16}$, we deduce that $\Pr(m_{j,k} < n_{j,k}/2) \leq e^{-\frac{\theta_1}{16}n2^{-jd}}$, and, since $N(j) \leq \frac{1}{\theta_1}2^{jd}$ by assumption **(A1)**,

$$\sum_{k=1}^{N(j)}\Pr\left(m_{j,k} < n_{j,k}/2\right) \leq \frac{1}{\theta_1}2^{jd}e^{-\frac{\theta_1}{16}n2^{-jd}}$$

and

$$\Pr\left(\max_{k=1\dots N(j)}\left\|\operatorname{Proj}_{V_{j,k}} - \operatorname{Proj}_{\widehat{V}_{j,k}}\right\| \geq 12\frac{\theta_2^2}{\theta_3}\sqrt{\frac{(t+\log(\bar{d}\vee 8))d^2}{n_{j,k}/2}}\right) \leq \frac{2^{jd}}{\theta_1}\left(e^{-t} + e^{-\frac{\theta_1}{16}n2^{-jd}}\right) \tag{20}$$

A similar argument implies that

$$\Pr\left(\max_{k=1\dots N(j)}\|c_{j,k} - \widehat{c}_{j,k}\| \geq 2r\sqrt{\frac{t+\log(\bar{d}\vee 8)}{n_{j,k}/2}}\right) \leq \frac{2^{jd}}{\theta_1}\left(e^{-t} + e^{-\frac{\theta_1}{16}n2^{-jd}}\right). \tag{21}$$

We are in position to conclude the proof of Theorem 2. With assumption **(A2)**, (20), and (21), the initial bound (14) implies that, with high probability,

$$\|P_j(x) - \widehat{P}_j(x)\| \leq 4\sqrt{2}\frac{\theta_2}{\sqrt{\theta_1}}2^{-j}\sqrt{\frac{t+\log(\bar{d}\vee 8)}{n2^{-jd}}} + 12\sqrt{2}\frac{\theta_2^3}{\theta_3\sqrt{\theta_1}}2^{-j}\sqrt{\frac{(t+\log(\bar{d}\vee 8))d^2}{n2^{-jd}}}.$$

Combined with assumption **(A3)** (see Remark 1, part *iii.*), this yields the result.

5.3 Proof of Theorem 7

Recall that $\mathcal{M} \hookrightarrow \mathbb{R}^D$ is a smooth (or at least C^2) compact manifold without boundary, with reach τ, and equipped with the volume measure $d\mathrm{Vol}_{\mathcal{M}}$. Our proof is divided into several steps, and each of them is presented in a separate subsection to improve readability.

5.3.1 Local Inversions of the Projection

In this section, we introduce lemmas which ensure that (for $r < \tau/8$) the projection map $\text{Proj}_{y+T_y\mathcal{M}}$ is injective on $B(y,r) \cap \mathcal{M}$, and hence invertible by part *iv.* of Proposition 11. We also demonstrate that the derivatives of this inverse are bounded in a suitable sense. These estimates shall allow us to develop bounds on volumes in $\mathcal{M}_\sigma$.

We begin by proving a bound on the local deviation of the manifold from a tangent plane.

Lemma 15 *Suppose $\eta \in T_y^\perp\mathcal{M}$ with $\|\eta\| = 1$ and $z \in B(y,r) \cap \mathcal{M}$, where $r \leq \tau/2$. Then*

$$|\langle \eta, z - y \rangle| \leq \frac{2r^2}{\tau}$$

Our next lemma quantitatively establishes the local injectivity of the affine projections onto tangent spaces. [1]

Lemma 16 *Suppose $y \in \mathcal{M}$ and $r < \tau/8$. Then $\text{Proj}_{y+T_y\mathcal{M}} : B(y,r) \cap \mathcal{M} \to y + T_y\mathcal{M}$ is injective.*

There are two important conclusions that Lemma 16 provides. First of all, it indicates that, under a certain radius bound, the manifold does not "curve back" into particular regions. This is helpful when we begin to examine upper bounds on local volumes. More importantly, if we let $J_{y,r} = \text{Proj}_{y+T_y\mathcal{M}}(B(y,r) \cap \mathcal{M})$, then there is a well-defined inverse map f of $\text{Proj}_{y+T_y\mathcal{M}}$, $f : J_{y,r} \to B(y,r) \cap \mathcal{M}$, when $r < \tau/8$. Part *iv* of Proposition 11 implies that f is at least a C^2 function, and part *v* of Proposition 11 implies that there is a d-dimensional ball inside of $J_{y,r}$ of radius $\cos(\theta)r$, where $\theta = \arcsin(r/2\tau)$.

Whenever we refer to such an f, we think of $J_{y,r}$ as a subset in the span of the first d canonical directions, and we identify f with the value f takes in the span of the remaining $D - d$ directions. Thus, we identify f with the function whose graph is a small part of the manifold. Such an identification is obtained via an affine transformation, so we may do this without any loss of generality. Using these assumptions, we may prove the following bounds.

Proposition 17 *Let $\varepsilon < \tau/8$, and assume f is defined above so that $v \longmapsto \begin{pmatrix} v \\ f(v) \end{pmatrix}$ is the inverse of $\text{Proj}_{y+T_y\mathcal{M}}$ in $B(y,\varepsilon)$ for some $y \in \mathcal{M}$. Then*

$$\sup_{v \in B_d(0,\varepsilon)} \|Df(v)\| \leq \frac{2\varepsilon}{\tau - 2\varepsilon} \tag{22}$$

and

$$\sup_{v \in B_d(0,\varepsilon)} \sup_{u \in \mathcal{S}^{D-d-1}} \left\| \sum_{i=1}^{D-d-1} u_i D^2 f_i(v) \right\| \leq \frac{\tau^2}{(\tau - 2\varepsilon)^3}. \tag{23}$$

1. In an independent work, Eftekhari and Wakin (2013) prove a slightly stronger result that holds for $r < \tau/4$.

5.3.2 Volume Bounds

The main result of this section is Lemma 18, which allows us to compare volumes in $\mathcal{M}_\sigma$ with volumes in $\mathcal{M}$. It also establishes an upper bound on volumes, which is an essential ingredient when we control the conditional distribution of Π subject to being in a particular $C_{j,k}$. The form of the bounds also allows us to cancel out noisy terms that would make the estimates depend upon the ambient dimension D.

Lemma 18 *Suppose $\sigma < \tau$, suppose $U \subseteq \mathcal{M}$ is measurable, and define $P : \mathcal{M}_\sigma \to \mathcal{M}$ so that $x \mapsto \text{Proj}_{\mathcal{M}}(x)$ under P. Then*

i. $\left(1 - \frac{\sigma}{\tau}\right)^d \text{Vol}_{\mathcal{M}}(U)\text{Vol}(B_{D-d}(0,\sigma)) \leq \text{Vol}(P^{-1}(U)) \leq \left(1 + \frac{\sigma}{\tau}\right)^d \text{Vol}_{\mathcal{M}}(U)\text{Vol}(B_{D-d}(0,\sigma))$

ii. If $r + \sigma \leq \tau/8$, then

$$\text{Vol}(\mathcal{M}_\sigma \cap B(y,r)) \leq \left(1 + \frac{\sigma}{\tau}\right)^d \left(1 + \left(\frac{2(r+\sigma)}{\tau - 2(r+\sigma)}\right)^2\right)^{d/2} \text{Vol}(B_d(0, r+\sigma))\text{Vol}(B_{D-d}(0,\sigma)).$$

5.3.3 Absolute Continuity of the Pushforward of $U_{\mathcal{M}_\sigma}$, and Local Moments

Recall that $U_{\mathcal{M}_\sigma}$ is the uniform distribution over $\mathcal{M}_\sigma$, and let $U_{\mathcal{M}} := \frac{d\,\text{Vol}_{\mathcal{M}}}{\text{Vol}_{\mathcal{M}}(\mathcal{M})}$ be the uniform distribution over $\mathcal{M}$. In this section, we exploit the volume bounds of the previous subsection to obtain control over probabilities and local moments of $U_{\mathcal{M}_\sigma}$. Our first result allows us to get the lower bounds for $U_{\mathcal{M}_\sigma}$ that are independent of the ambient dimension D.

Lemma 19 *Suppose $\sigma < \tau$, and let $\widetilde{U}_{\mathcal{M}_\sigma}$ denote the pushforward of $U_{\mathcal{M}_\sigma}$ under $\text{Proj}_{\mathcal{M}}$. Then $\widetilde{U}_{\mathcal{M}_\sigma}$ and $U_{\mathcal{M}}$ are mutually absolutely continuous with respect to each other, and*

$$\left(\frac{\tau - \sigma}{\tau + \sigma}\right)^d \leq \frac{d\widetilde{U}_{\mathcal{M}_\sigma}}{dU_{\mathcal{M}}} \leq \left(\frac{\tau + \sigma}{\tau - \sigma}\right)^d.$$

Proof This is a straightforward consequence of part *i.* of Lemma 18. ■

The next lemma quantitatively establishes the the decay of the local eigenvalues required in the second part of Assumption **(A3)**.

Lemma 20 *Suppose Π is a distribution supported on $\mathcal{M}_\sigma$, and let $r < \tau/2$. Further assume that Z is the random variable drawn from Π conditioned on the event $Z \in Q$ where $\mathcal{M}_\sigma \cap Q \subset B(y,r)$ for some $y \in \mathcal{M}$. If Σ is the covariance matrix of Z, then*

$$\sum_{i=d+1}^{D} \lambda_i(\Sigma) \leq 2\sigma^2 + \frac{8r^4}{\tau^2},$$

where $\lambda_i(\Sigma)$ are the eigenvalues of Σ arranged in the decreasing order.

Finally, we derive a lower bound on the upper eigenvalues of the local covariance for the uniform distribution (needed to satisfy the first part of assumption **(A3)**). This is done in the following lemma.

Lemma 21 *Suppose that $Q \subseteq \mathbb{R}^D$ is such that*

$$B(y, r_1) \subseteq Q \text{ and } \mathcal{M}_\sigma \cap Q \subset B(y, r_2)$$

for some $y \in \mathcal{M}$ and $\sigma < r_1 < r_2 < \tau/8 - \sigma$. Let Z be drawn from $U_{\mathcal{M}_\sigma}$ conditioned on the event $Z \in Q$, and suppose Σ is the covariance matrix of Z. Then

$$\lambda_d(\Sigma) \geq \frac{1}{4\left(1+\frac{\sigma}{\tau}\right)^d}\left(\frac{r_1-\sigma}{r_2+\sigma}\right)^d \left(\frac{1-\left(\frac{r_1-\sigma}{2\tau}\right)^2}{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}\right)^{d/2} \frac{(r_1-\sigma)^2}{d}.$$

The following statement is key to establishing the error bounds for GMRA measured in sup-norm.

Lemma 22 *Assume that conditions of Lemma 21 hold, and let $V_d := V_d(\Sigma)$ be the subspace corresponding to the first d principal components of Z. Then*

$$\sup_{x\in Q} \|x - \mathbb{E}Z - \mathrm{Proj}_{V_d}(x - \mathbb{E}Z)\| \leq 2\sigma + \frac{4r_2^2}{\tau} + \frac{r_2}{r_1-\sigma}\sqrt{4\sigma^2 + \frac{16r_2^4}{\tau^2}}\,\gamma(\sigma,\tau,d,r_1,r_2),$$

where $\gamma(\sigma,\tau,d,r_1,r_2) = 4\sqrt{2d}\left(1+\frac{\sigma}{\tau}\right)^{d/2}\left(\frac{r_2+\sigma}{r_1-\sigma}\right)^{d/2}\left(\frac{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}{1-\left(\frac{r_1-\sigma}{2\tau}\right)^2}\right)^{d/4}$.

Notice that the term containing $\gamma(\sigma,\tau,d,r_1,r_2)$ is often of smaller order, so that the approximation is essentially controlled by the maximum of σ and $\frac{r_2^2}{\tau}$.

5.3.4 Putting all the Bounds Together

In this final subsection, we prove Theorem 7. We begin by translating Proposition 3.2 in (Niyogi et al., 2008) into our setting. As before, let $\mathcal{X}_n = \{X_1, \ldots, X_n\}$ be an i.i.d. sample from Π, and the ϕ_1 be the constant defined by (10).

Proposition 23 *(Niyogi et al., 2008, Proposition 3.2) Suppose $0 < \varepsilon < \frac{\tau}{2}$, and also that n and t satisfy*

$$n \geq \varepsilon^{-d}\frac{1}{\phi_1}\left(\frac{\tau+\sigma}{\tau-\sigma}\right)^d \beta_1\left(\log(\varepsilon^{-d}\beta_2) + t\right), \tag{24}$$

where $\beta_1 = \frac{\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}{\cos^d(\delta_1)\mathrm{Vol}(B_d(0,1/4))}$, $\beta_2 = \frac{\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}{\cos^d(\delta_2)\mathrm{Vol}(B_d(0,1/8))}$, $\delta_1 = \arcsin(\varepsilon/8\tau)$, and $\delta_2 = \arcsin(\varepsilon/16\tau)$. Let $\mathcal{E}_{\varepsilon/2,n}$ be the event that

$$\mathcal{Y} = \{Y_j = \mathrm{Proj}_{\mathcal{M}}(X_j)\}_{j=1}^n$$

is $\varepsilon/2$-dense in $\mathcal{M}$ (that is, $\mathcal{M} \subseteq \bigcup_{i=1}^n B(Y_i, \varepsilon/2)$). Then, $\Pi^n(\mathcal{E}_{\varepsilon,n}) \geq 1 - e^{-t}$, where Π^n is the n-fold product measure of Π.

Proof The proof closely follows the one given in (Niyogi et al., 2008). The only additional observation to make is that, if $\widetilde{\Pi}$ is the pushforward measure of Π under $\mathrm{Proj}_{\mathcal{M}} : \mathcal{M}_\sigma \to \mathcal{M}$, then

$$\begin{aligned}
\widetilde{\Pi}\left(\mathcal{M} \cap B(y, \varepsilon/8)\right) &= \Pi(\mathrm{Proj}_{\mathcal{M}}^{-1}(\mathcal{M} \cap B(y, \varepsilon/8))) \\
&\geq \phi_1 U_{\mathcal{M}_\sigma}(\mathrm{Proj}_{\mathcal{M}}^{-1}(\mathcal{M} \cap B(y, \varepsilon/8))) \\
&= \phi_1 \widetilde{U}_{\mathcal{M}_\sigma}(\mathcal{M} \cap B(y, \varepsilon/8)) \\
&\geq \phi_1 \left(\frac{\tau-\sigma}{\tau+\sigma}\right)^d U_{\mathcal{M}}(\mathcal{M} \cap B(y, \varepsilon/8)).
\end{aligned}$$

by Lemma 18. ∎

If $\varepsilon \ll \tau$, previous proposition implies that we roughly need $n \geq \mathrm{Const}(\mathcal{M}, d)\left(\frac{1}{\varepsilon}\right)^d \log\frac{1}{\varepsilon}$ points to get an ε-net for $\mathcal{M}$. For the remainder of this section, we identify $\varepsilon := \varepsilon(n,t)$ with the smallest $\varepsilon > 0$ satisfying (24) in the statement of Proposition 23, and we also assume that $\varepsilon < \sigma$. Take $j \in \mathbb{Z}_+$ such that

$$\sigma < 2^{-j-2} < \tau. \tag{25}$$

Let $C_{j,k}$ be the partition of $\mathbb{R}^D$ into Voronoi cells defined by (11). Recall that $T_j = \{a_{j,k}\}_{k=1}^{N(j)} \subset \mathcal{X}_n$ is the set of nodes of the cover tree at level j, and set $z_{j,k} = \mathrm{Proj}_{\mathcal{M}}(a_{j,k})$.

Lemma 24 *With probability $\geq 1 - e^{-t}$, for all j satisfying (25) and $k = 1, \ldots, N(j)$,*

$$B\left(z_{j,k}, 2^{-j-2}\right) \subseteq C_{j,k} \text{ and } C_{j,k} \cap \mathcal{M}_\sigma \subseteq B(a_{j,k}, 3\cdot 2^{-j-2} + 2^{-j+1}) \subseteq B(z_{j,k}, 3\cdot 2^{-j}). \tag{26}$$

We now use Lemma 24 to obtain bounds on the constants θ_i for $i = 1, \ldots, 4$ and α. We prove a lemma for each of the assumptions **(A1)**, **(A2)**, and **(A3)** and then collect them as the proof of Theorem 7.

Proof [Proof of Theorem 7] Since the hypotheses of Lemma 24 are satisfied with high probability, we first obtain

$$\begin{aligned}
\Pi(C_{j,k}) &\geq \Pi(B(z_{j,k}, 2^{-j-2})) \\
&\geq \phi_1 U_{\mathcal{M}_\sigma}(B(z_{j,k}, 2^{-j-2})) \\
&= \phi_1 \frac{\mathrm{Vol}(\mathcal{M}_\sigma \cap B(z_{j,k}, 2^{-j-2}))}{\mathrm{Vol}(\mathcal{M}_\sigma)} \\
&\geq \phi_1 \frac{\mathrm{Vol}(\mathrm{Proj}_{\mathcal{M}}^{-1}(\mathcal{M} \cap B(z_{j,k}, 2^{-j-2} - \sigma)))}{\mathrm{Vol}(\mathcal{M}_\sigma)} \\
&\geq \phi_1 \left(\frac{\tau-\sigma}{\tau+\sigma}\right)^d \frac{\cos(\delta)^d \mathrm{Vol}(B_d(0, 2^{-j-2} - \sigma))}{\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})} \\
&\geq \frac{\phi_1 \mathrm{Vol}(B_d(0,1))}{2^{4d}\,\mathrm{Vol}_{\mathcal{M}}(\mathcal{M})}\left(\frac{\tau-\sigma}{\tau+\sigma}\right)^d 2^{-jd}.
\end{aligned}$$

where $\delta = \arcsin((2^{-j-2} - \sigma)/2\tau))$. Thus,

$$\theta_1 \geq \frac{\phi_1 \mathrm{Vol}(B_d(0,1))}{2^{4d}\,\mathrm{Vol}\,_{\mathcal{M}}(\mathcal{M})}\left(\frac{\tau-\sigma}{\tau+\sigma}\right)^d$$

Since the support is contained in a ball of radius $3 \cdot 2^{-j}$, we easily obtain that $\theta_2 \leq 12$. Finally, it is not difficult to deduce from Lemmas 20 and 21 that

$$\theta_3 \geq \frac{\phi_1/\phi_2}{2^{4d+8}\left(1+\frac{\sigma}{\tau}\right)^d},\ \theta_4 \leq \left(2 \vee \frac{2^3 3^4}{\tau^2}\right),\ \text{and } \alpha = 1.$$

Lemma 22 together with Lemma 24 imply that

$$\theta_5 \leq \left(2 \vee \frac{4\cdot 3^2}{\tau}\right)\left(1 + 3\cdot 2^5\sqrt{2d}\left(1+\frac{\sigma}{\tau}\right)^{d/2}\left(\frac{1+\left(\frac{25}{71}\right)^2}{1-\frac{1}{9\cdot 2^{12}}}\right)^{d/4}\right).$$

■

6. Numerical Experiments

In this section, we present some numerical experiments consistent with our results.

6.1 Spheres of Varying Dimension in $\mathbb{R}^D$

We consider n points $X_1, \ldots, X_n$ sampled i.i.d. from the uniform distribution on the unit sphere in $\mathbb{R}^{d+1}$

$$\mathcal{M} = \mathbb{S}^d := \{x \in \mathbb{R}^{d+1} : \|x\| = 1\}.$$

We then embed $\mathbb{S}^d$ into $\mathbb{R}^D$ for $D \in \{10, 100\}$ by applying a random orthogonal transformation $\mathbb{R}^{d+1} \to \mathbb{R}^D$. Of course, the actual realization of this projection is irrelevant since our construction is invariant under orthogonal transformations. After performing this embedding, we add two types of noise. In the first case, we add Gaussian noise ξ with distribution $\mathcal{N}(0, \frac{\sigma^2}{D} I_D)$: the scaling factor $\frac{1}{D}$ is chosen so that $\mathbb{E}\|\xi\|^2 = \sigma^2$. Since the norm of a Gaussian vector is tightly concentrated around its mean, this model is well-approximated by the "truncated Gaussian" model where the distribution of the additive noise is the same as the conditional distribution of ξ given $\|\xi\| \leq C\sigma$, where C is such that $C\sigma < 1$. In this case, the constants in $(1, C\sigma)$-model assumption would be prohibitively large, so instead we can verify the conditions given in Remark 9 directly: due to symmetry, we have that for any $A \subset \mathbb{S}^d$,

$$\Pi\left(\mathrm{Proj}\,_{\mathcal{M}}^{-1}(A)\right) = U_{\mathcal{M}}(A) = U_{\mathcal{M}_\sigma}\left(\mathrm{Proj}\,_{\mathcal{M}}^{-1}(A)\right). \tag{27}$$

On the other hand, it is a simple geometric exercise to show that, for any B such that $B(y,r) \cap \mathcal{M}_\sigma \subseteq B \subseteq B(y, 12r)$ and $\tau/2 = 1/2 > r \geq 2C\sigma$,

$$\mathrm{Proj}\,_{\mathcal{M}}^{-1}\left(\mathcal{M} \cap B(y, \tilde{r}_1)\right) \supseteq B \supseteq \mathrm{Proj}\,_{\mathcal{M}}^{-1}\left(\mathcal{M} \cap B(y, \tilde{r}_2)\right),$$

where $\tilde{r}_1 = \frac{r}{\sqrt{\frac{1+\sqrt{1-r^2}}{2}}}$ and $\tilde{r}_2 = r\sqrt{\frac{3}{4(1+C\sigma)}}$. Lemma 18 and (27) imply that

$$\begin{aligned}\Pi(B) &\leq U_{\mathcal{M}_\sigma}\left(\mathrm{Proj}\,_{\mathcal{M}}^{-1}\left(\mathcal{M} \cap B(y, \tilde{r}_1)\right)\right)\\ &\leq (1+C\sigma)^d\,\mathrm{Vol}\,_{\mathcal{M}}(\mathcal{M} \cap B(y, \tilde{r}_1))\frac{\mathrm{Vol}(B_{D-d}(0, C\sigma))}{\mathrm{Vol}(\mathcal{M}_{C\sigma})}\end{aligned}$$

and

$$\begin{aligned}U_{\mathcal{M}_\sigma}(B) &\geq U_{\mathcal{M}_\sigma}\left(\mathrm{Proj}\,_{\mathcal{M}}^{-1}\left(\mathcal{M} \cap B(y, \tilde{r}_2)\right)\right)\\ &\geq (1-C\sigma)^d\,\mathrm{Vol}\,_{\mathcal{M}}(\mathcal{M} \cap B(y, \tilde{r}_2))\frac{\mathrm{Vol}(B_{D-d}(0, C\sigma))}{\mathrm{Vol}(\mathcal{M}_{C\sigma})},\end{aligned}$$

hence $\Pi(B) \leq \tilde{\phi}_2 U_{\mathcal{M}_\sigma}(B)$ for some $\tilde{\phi}_2$ independent of the ambient dimension D.

We present the behavior of the $L^2(\Pi)$ error in this case in Figure 1, and the rate of approximation at the optimal scale as the number of samples varies in Figure 3, where it is compared to the rates obtained in Corollary 8. From Figure 1, we see that the approximations obtained satisfy our bound, and are typically better even for a modest number of samples in dimensions non-trivially low (e.g. 8000 samples on $\mathbb{S}^8$). In fact, the robustness with respect to sampling is such that the plots barely change from row to row.

The second type of noise is uniform in the radial direction, i.e. we let $\eta \sim \mathrm{Unif}[1-\sigma, 1+\sigma]$ and each noisy point is generated by $\tilde{X}_i = X_i + \eta_i \frac{X_i}{\|X_i\|}$. This is an example where the noise is not independent of X. Once again, it is easy to check directly that conditions of Remark 9 hold (the argument mimics the approach we used for the truncated Gaussian noise). Simulation results for this scenario are summarized in Figure 2, with the rate of approximation at the optimal scale again in Figure 3.

We considered various settings of the parameters, namely all combinations of: $d \in \{1,2,4,6,8\}$, $n \in \{8000, 16000, 32000, 64000, 128000\}$, $D \in \{100, 1000\}$, $\sigma \in \{0, 0.05, 0.1\}$. We only display some of the results for reasons of space constraints. [2]

6.2 Meyer's Staircase

We consider the (d-dimensional generalization of) Y. Meyer's staircase. Consider the cube $Q = [0,1]^d$ and the set of Gaussians $\mathcal{N}(x; \mu, \delta^2 I_d)$ where the mean μ is allowed to vary over Q, and the function is truncated to only accept arguments $x \in Q$. Varying μ in Q in this manner induces a smooth embedding of a d-dimensional manifold into the infinite dimensional Hilbert space $L^2(Q)$. That is, the Gaussian density centered at $\mu \in Q$ and truncated to $x \in Q$ is a point in $L^2(Q)$. By discretizing Q, we may sample this manifold and project it into a finite dimensional space. In particular, a grid $\Gamma_D \subseteq Q$ of D points (obtained by subdividing in $D^{-\frac{1}{d}}$ equal parts along each dimension) may be generated and considering the evaluations of the set of translated Gaussians on this grid produces an embedding of this manifold into $\mathbb{R}^D$. Sampling n points from this manifold by randomly drawing $\mu_1, \ldots, \mu_n$ uniformly from Q, we obtain a set $\{\mathcal{N}(x; \mu_i, \delta^2 I_d)|_{\Gamma_D}\}_{i=1,\ldots,n}$ of n samples from the "discretized" Meyer's staircase in $\mathbb{R}^D$. This is what we call a sample from Meyer's

2. The code provided at `www.math.duke.edu/~mauro/code.html` can generate all the figures, re-create the data sets, and is easily modified to do more experiments.

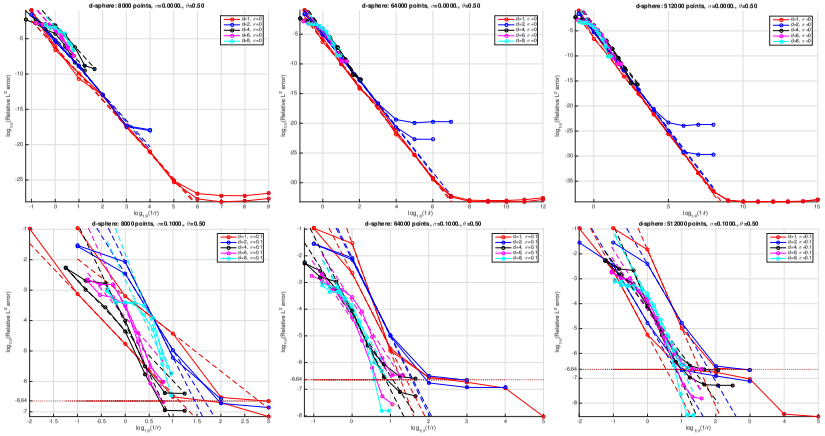


Figure 1: Experiment with $\mathbb{S}^d$, without (top row) and with Gaussian noise (bottom row). The columns correspond to different values of $n \in \{8000, 64000, 512000\}$. In the plots the dots represent the $L^2(\Pi)$ error squared (or MSE) of GMRA approximations (see (5)) as a function of the radius r at scale j; more precisely the abscissa is in terms of $\log_2(1/r_j)$, where r_j is the mean radius of $C_{j,k}$ for a fixed j, and the ordinate is $\log_2 \mathrm{MSE}_j$, where MSE_j is the mean squared error of the GMRA approximation at scale j. Different colors correspond to different intrinsic dimensions d (see legend). The two cases $D = 10, 100$ use the same colors for both the dots and the lines, all of which are essentially superimposed since our results are independent of the ambient dimension D. For each dimension we fit a line to measure the decay, which is $O(r^{-4})$ independently of d, consistently with our analysis. The horizontal dotted line, with corresponding tick mark on the Y axis, represents the noise level σ^2: the approximation error flattens out at roughly that level, as expected.

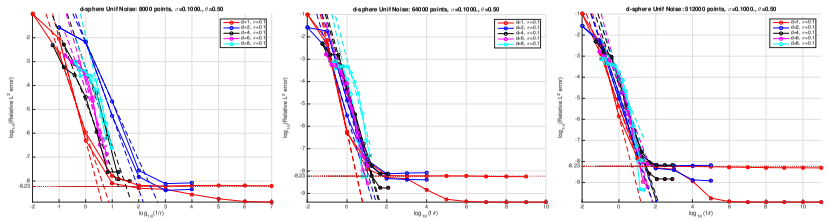


Figure 2: This figure is as the second row of Figure 1, but the noise is radially uniform with width parameter σ. Note that the variance of the noise is $\sigma^2/3$, which is indicated in the figure by horizontal line and an extra tick mark on the Y-axis in the figures. The MSE converges quickly to that level as a function of scale.

staircase, which is illustrated in Figure 4. This example is not artificial: for example, translating a white shape on a black background produces a set of $2 - D$ images with a similar structure to the d-dimensional Meyer's staircase for $d = 2$.

The manifold associated with Meyer's staircase is poorly approximated by subspaces of dimension smaller than $O(D \wedge 1/\delta^D)$, and besides spanning many dimensions in $\mathbb{R}^D$, it has a small reach, depending on d, D, δ. In our examples we considered

$$n = 8000, 16000, 32000, 640000, 128000, \ d = 1, 2, 4, \ D = 2000, \ \text{and } \delta = \frac{5}{100}.$$

We consider the noiseless case, as well as the case where Gaussian noise $\mathcal{N}(0, \frac{1}{D} I_D)$ is added to the data. Since this type of noise does not abide by the (σ, τ)-model assumption and τ is very small for Meyer's staircase, Figure 5 illustrates the behavior of the GMRA approximation outside of the regime where our theory is applicable.

6.3 The MNIST Dataset of Handwritten Digits

We consider the MNIST data set of images of handwritten digits[3], each of size 28×28, grayscale. There are total of $60,000$, from ten classes consisting of digits $0, 1, \ldots, 9$. The intrinsic dimension of this data set is variable across the data, perhaps because different digits have a different number of "degrees of freedom" and across scales, as it is observed in Little et al. (2012). We run GMRA by setting the cover tree scaling parameter θ equal to 0.9 (meaning that we replace $1/2$ with 0.9 in definition of cover trees in section 3.2) in order to slowly "zoom" into the data at multiple scales. As the intrinsic dimension is not well-defined, we set GMRA to pick the dimension of the planes $\mathbb{V}_{j,k}$ adaptively, as the smallest dimension needed to capture half of the "energy" of the data in $C_{j,k}$. The distribution of dimensions of the subspaces $\mathbb{V}_{j,k}$ has median 3 (consistently with the estimates in Little et al. (2012)) and is represented in figure 6. We then compute the L^2 relative approximation error, and compute various quantiles: this is reported in the same figure. The running time on a desktop was few minutes.

3. Available at `http://yann.lecun.com/exdb/mnist/`.

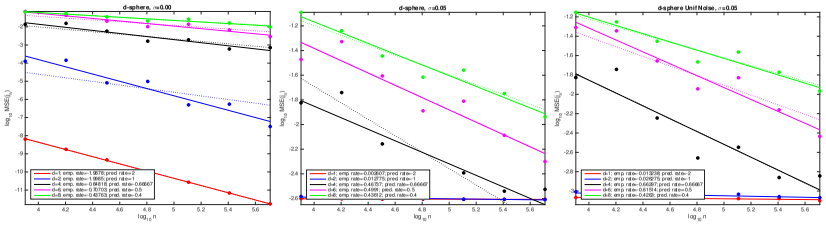


Figure 3: For the example of $\mathbb{S}^d$ considered in this section we consider the MSE error, i.e. $L^2(\Pi)$ squared error (as defined in (5)) at the optimal scale j_n (as in the proof of Corollary 8) as a function of the number of points $n \in \{8000, 16000, 32000, 64000, 128000, 256000, 512000\}$, and compare our empirical rates (solid linear, with the rate reported in the legend under "emp. rate") with the rates predicted by Corollary 8 (dotted lines, with rate reported in the legend under "pred. rate"), for various choices of the intrinsic dimension $d \in \{1, 2, 4, 6, 8\}$ and fixed ambient dimension $D = 10$ (the results are independent of D, so we do note report the - very similar - results obtained for $D = 100$). Left: noiseless case, middle: Gaussian noise, right: radial uniform noise (see text). The rates match our results quite well, except in the case $d = 2$ where we seem to obtain the same convergence rate as in the $d = 1$ case. Here we are choosing the optimal scale to be the finest scale such that, in every cell, we have at least $10d^2$ points. For the noisy cases, the approximation rates for $d = 1, 2$ are not meaningful simply because we have enough points to go the finest scale above the noise level.

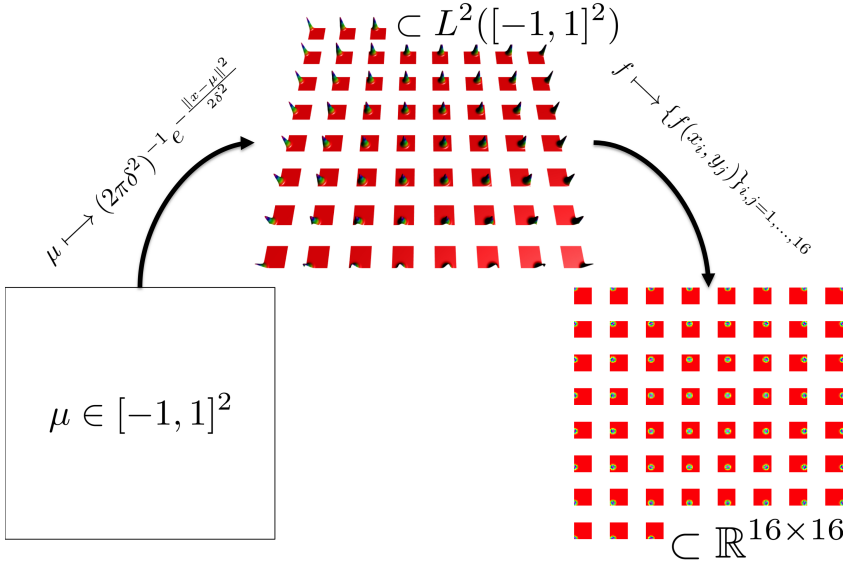


Figure 4: An illustration of Meyer's staircase for $d = 2$. We see that the square is mapped into a subset of $L^2([-1, 1]^2)$ consisting of truncated Gaussians. These are then sampled at points on a uniform, 16 by 16 grid to obtain an embedding of $[0, 1]^2$ into $\mathbb{R}^{16\times 16}$. For small δ, this embedding has a point very close to each coordinate axis in $\mathbb{R}^{16\times 16}$. Thus, it comes as no surprise that this embedding of $[-1, 1]^2$ into $\mathbb{R}^{256}$ has a high degree of curvature.

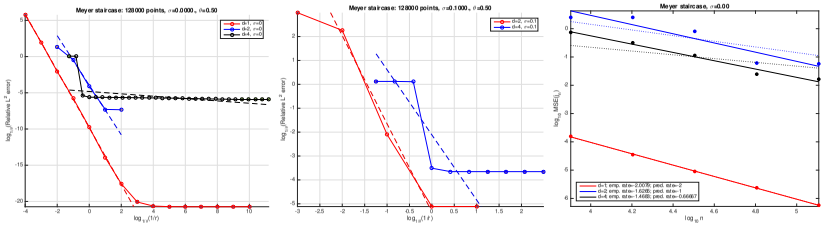


Figure 5: Left and middle: MSE as a function of scale r for the d-dimensional Meyer's staircase, for different values of $n =$, d and σ, standard deviation of Gaussian noise $\mathcal{N}(0, \frac{\sigma^2}{D})$. The small reach of Meyer's staircase makes it harder to approximate, and makes the approximation much more susceptible to noise. Moreover, Gaussian noise is unbounded, so this distribution violates the (σ, τ)-model assumption (albeit only at a small number of points, with high probability). Right: MSE at the optimal scale, chosen so that every cell contains at least $10d^2$ points.

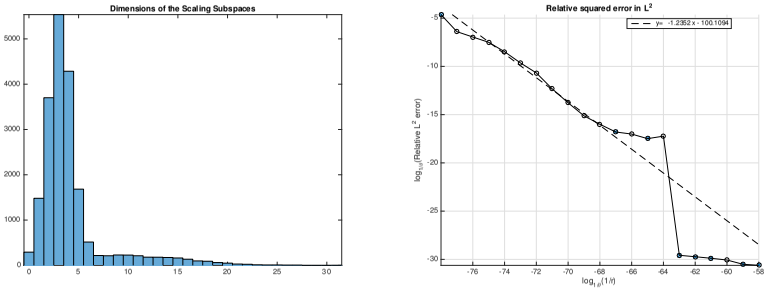


Figure 6: Left: histogram of the dimension of the scaling function subspaces for the MNIST data set. We see that many of the subspaces are very low-dimensional, with dimensions mostly between 2 and 5. Right: L^2 relative approximation error squared as a function of scale. We do not plot the quantiles since many of them are many orders of magnitude smaller (which is a good thing in terms of approximation error), creating artifacts in the plots; they do indicate thought that the structure of the data is highly complex and not heterogeneous. Note that the axis of this plot are in $\log_{1/\theta}$ scale, where $\theta = 0.9$ is the cover tree scaling factor used in this example. Note how the approximation error decreases slowly at the beginning, as there are many classes, rather far from each other, so that it takes a few scales before GMRA starts focusing into each class, at which point the approximation error decreases more rapidly. This phenomenon does not happen uniformly over the data (figure not shown).

6.4 Sonata Kreutzer

We consider a recording of the first movement of the Sonata Kreutzer by L.V. Beethoven, played by Y. Pearlman (violin) and V. Ashkenazy (piano) (EMI recordings). The recording is stereo, sampled at 44.1kHz. We map it to mono by simply summing the two audio channels, and then we generate a high-dimensional dataset as follows. We consider windows of width w seconds, overlapping by δw seconds, and consider the samples in each such time window $[i\delta w, i\delta w + w)$ as a high-dimensional vector X_i', of dimension equal to the sampling rate times w. In our experiment we choose $w = 0.1$ seconds, $\delta w = 0.05$ seconds, and the resulting vectors X_i' are $D' = 551$-dimensional. Since Euclidean distances between the X_i' are far from being perceptually relevant, we transform each X_i' to its cepstrum (see Oppenheim and Schafer (1975)), remove the central low-pass frequency, and discard the symmetric part of the spectrum (the signal is real), obtaining X_i, a vector with $D = 275$ dimensions, and i ranges from 0 to about 130,000. The running time on a desktop was few minutes.

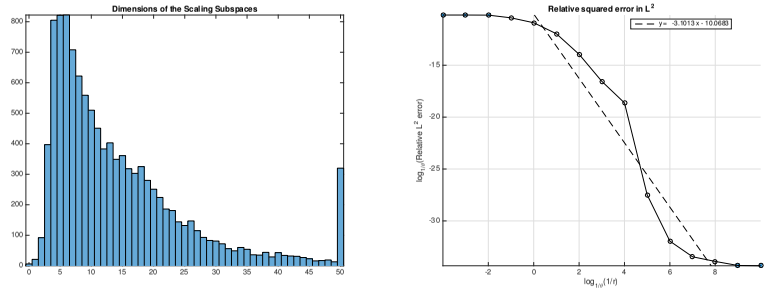


Figure 7: Left: histogram of the dimension of the scaling function subspaces for the Kreutzer sonata dataset. We see that the dimension of the scaling function subspaces is mostly between 4 and 25. Right: mean L^2 relative approximation error squared as a function of scale. We do not plot the quantiles since many of them are many orders of magnitude smaller (which is a good thing in terms of approximation error), creating artifacts in the plots; they do indicate that the structure of the data is highly complex and non-heterogeneous. Note that the axes of this plot are in $\log_{1/\theta}$ scale, where $\theta = 0.9$ is the scaling factor used in this example.

Acknowledgments

The authors gratefully acknowledge support from NSF DMS-0847388, NSF DMS-1045153, ATD-1222567, CCF-0808847, AFOSR FA9550-14-1-0033, DARPA N66001-11-1-4002. We would also like to thank Mark Iwen for his insightful comments.

Appendix: Proofs of Geometric Propositions and Lemmas

Proof [Proof of Proposition 10] For the first inequality, let

$$A = \begin{pmatrix} I \\ X \end{pmatrix} \text{ and } B = \begin{pmatrix} Y \\ 0 \end{pmatrix},$$

and for every $T \subset [d]$, we let V_T denote the volume of $\{a_i\}_{i\in T^c} \cup \{b_i\}_{i\in T}$, where a_i and b_i denote the ith columns of A and B respectively. By submultilinearity of the volume we have

$$\mathrm{Vol}(A+B) \leq \sum_{T\in 2^{[d]}} V_T,$$

where $2^{[d]} = \{S : S \subset \{1,\dots,d\}\}$. We now show that $V_T \leq q^{|T|}\mathrm{Vol}(A)$ for every $T \in 2^{[d]}$. The bound $\|Y\| \leq q$ implies $\|y_i\| \leq q$ for all $i = 1,\dots,d$, and so the fact that the volume is a submultiplicative function implies that

$$V_T \leq q^{|T|}\mathrm{Vol}(A_{T^c}).$$

On the other hand, letting $a_1^\perp$ be the orthogonal projection of a_1 onto $\mathrm{span}^\perp\{a_i\}_{i=2}^d$, we note that $\|a_1^\perp\| \geq 1$, and thus

$$\mathrm{Vol}(A_{\{1\}^c}) \leq \|a_1^\perp\|\mathrm{Vol}(A_{\{1\}^c}) = \mathrm{Vol}(A).$$

By induction and invariance of the volume under permutations, we see that $\mathrm{Vol}(A_{T^c}) \leq \mathrm{Vol}(A)$ for all $T \in 2^{[d]}$. Thus,

$$\mathrm{Vol}(A+B) \leq \sum_{T\in 2^{[d]}} q^{|T|}\mathrm{Vol}(A) = (1+q)^d\mathrm{Vol}(A).$$

For the second inequality, since Y is symmetric, we can represent it as $Y = F - G$ where F and G are symmetric positive semidefinite, $FG = GF = 0$, and $\|F\|, \|G\| \leq \|Y\|$. Indeed, if $Y = Q\Lambda Q^T$ is the eigenvalue decomposition of Y with $\Lambda = \mathrm{diag}(\lambda)$, set $\lambda_+ := (\max(0,\lambda_1),\dots,\max(0,\lambda_d))^T$, $\lambda_- := \lambda_+ - \lambda$, and define $F := Q\,\mathrm{diag}(\lambda_+)Q^T$, $G = Q\,\mathrm{diag}(\lambda_-)Q^T$.

Recall the *matrix determinant lemma*: let $T \in \mathbb{R}^{k\times k}$ be invertible, and let $U, V \in \mathbb{R}^{k\times l}$. Then

$$\mathrm{Vol}(T + UV^T) = \mathrm{Vol}(I + V^TT^{-1}U)\mathrm{Vol}(T).$$

Applying it in our case with $U = \begin{pmatrix} \sqrt{F} - \sqrt{G} \\ 0 \end{pmatrix}$, $V = \begin{pmatrix} \sqrt{F} + \sqrt{G} \\ 0 \end{pmatrix}$, and $T = \begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}$, we have that

$$\mathrm{Vol}\begin{pmatrix} I+Y & X^T \\ X & -I \end{pmatrix} = \mathrm{Vol}\left(I + \begin{pmatrix} \sqrt{F} + \sqrt{G} \\ 0 \end{pmatrix}^T \begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}^{-1} \begin{pmatrix} \sqrt{F} - \sqrt{G} \\ 0 \end{pmatrix}\right) \mathrm{Vol}\begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}.$$

By orthogonality of the columns in $\begin{pmatrix} I \\ X \end{pmatrix}$ with the columns in $\begin{pmatrix} X^T \\ -I \end{pmatrix}$, we have that

$$\left\|\begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}\right\| \geq \left\|\begin{pmatrix} u \\ v \end{pmatrix}\right\|,$$

and hence

$$\left\|\begin{pmatrix} \sqrt{F} + \sqrt{G} \\ 0 \end{pmatrix}^T \begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}^{-1} \begin{pmatrix} \sqrt{F} - \sqrt{G} \\ 0 \end{pmatrix}\right\| \leq \sqrt{q}\cdot 1 \cdot \sqrt{q} = q.$$

Therefore, we conclude that

$$\mathrm{Vol}\left(I + \begin{pmatrix} \sqrt{F} + \sqrt{G} \\ 0 \end{pmatrix}^T \begin{pmatrix} I & X^T \\ X & -I \end{pmatrix}^{-1} \begin{pmatrix} \sqrt{F} - \sqrt{G} \\ 0 \end{pmatrix}\right) \geq (1-q)^d,$$

and combining this with the expression from the matrix determinant lemma completes the proof. ■

Proof [Proof of Lemma 15] Let $\gamma : [0, d_{\mathcal{M}}(z,y)] \to \mathcal{M}$ denote the arclength-parameterized geodesic connecting y to z in $\mathcal{M}$. Since γ is a geodesic, there is a $v \in T_y\mathcal{M}$ with $\|v\| = 1$ such that the Taylor expansion

$$z = y + d_{\mathcal{M}}(z,y)v + \int_0^{d_{\mathcal{M}}(z,y)} \gamma''(t)\,(d_{\mathcal{M}}(z,y) - t)\,dt.$$

By Proposition 11, $\|\gamma''(t)\|_2 \leq 1/\tau$ for all t and $d_{\mathcal{M}}(z,y) \leq 2r$, so we have that

$$\begin{aligned} |\langle \eta, z-y\rangle| &= \left|\left\langle \eta, \int_0^{d_{\mathcal{M}}(z,y)} \gamma''(t)\,(d_{\mathcal{M}}(z,y)-t)\,dt\right\rangle\right| \\ &\leq \int_0^{d_{\mathcal{M}}(z,y)} |\langle \eta, \gamma''(t)\rangle|\,(d_{\mathcal{M}}(z,y)-t)\,dt \\ &\leq \frac{1}{\tau}\int_0^{d_{\mathcal{M}}(z,y)} (d_{\mathcal{M}}(z,y)-t)\,dt \\ &\leq \frac{d_{\mathcal{M}}(z,y)^2}{2\tau} \\ &\leq \frac{2r^2}{\tau}. \end{aligned}$$

■

Proof [Proof of Lemma 16] Suppose a and b are distinct in $B(y,r)\cap\mathcal{M}$. Now, $b-a = v+w$ where $v \in T_a\mathcal{M}$ and $w \in T_a^\perp\mathcal{M}$, and note that $\|w\| \leq \frac{2\|b-a\|^2}{\tau} \leq 4\frac{r}{\tau}$ by Lemma 15. This also implies that

$$\|v\| = \sqrt{\|a-b\|^2 - \|w\|^2} \geq \sqrt{\|a-b\|^2 - 4\frac{\|a-b\|^4}{\tau^2}} \geq \|a-b\|\sqrt{1-16\frac{r^2}{\tau^2}} \geq \|a-b\|\sqrt{1-4\frac{r}{\tau}}.$$

By part *iii.* of Proposition 11, there is a $u \in T_y\mathcal{M}$ such that $\langle u, v\rangle \geq \|v\|\cos(\phi)$ where ϕ is the angle between $T_y\mathcal{M}$ and $T_a\mathcal{M}$. Then

$$\begin{aligned}
|\langle u, b-a\rangle| &\geq |\langle u, v\rangle| - |\langle u, w\rangle| \\
&\geq \|v\|\cos(\phi) - \|w\| \\
&\geq \|a-b\|\sqrt{1-4\frac{r}{\tau}}\sqrt{1-2\frac{r}{\tau}} - 2\frac{\|a-b\|^2}{\tau} \\
&\geq \|b-a\|\left(\sqrt{1-4\frac{r}{\tau}}\sqrt{1-4\frac{r}{\tau}} - 4\frac{r}{\tau}\right) \\
&\geq \|b-a\|\left(1-8\frac{r}{\tau}\right).
\end{aligned}$$

It then follows from $r < \tau/8$ that $\text{Proj}_{T_y\mathcal{M}}(b-a) \neq 0$, and hence $\text{Proj}_{y+T_y\mathcal{M}}(a) \neq \text{Proj}_{y+T_y\mathcal{M}}(b)$ and injectivity holds. ■

Proof [Proof of Proposition 17] For $\varepsilon < \tau/8$, we may define the embedding

$$\begin{pmatrix} v \\ \beta \end{pmatrix} \longmapsto \begin{pmatrix} v \\ f(v) \end{pmatrix} + \begin{pmatrix} Df(v)^T \\ -I \end{pmatrix} \beta$$

where we have assumed (without loss of generality) that $y = 0$ and $T_y\mathcal{M}$ coincides with the span of the first d canonical orthonormal basis members. The domain of this map is the set

$$\Omega = \{(v,\beta) \in \mathbb{R}^d \times \mathbb{R}^{D-d} : v \in T_y\mathcal{M} \cap B(0,\varepsilon), \|\beta\|^2 + \|Df(v)^T\beta\|^2 < \tau^2\}$$

and the Jacobian of this map is

$$\begin{pmatrix} I + \sum_{i=1}^{D-d}\beta_i D^2 f_i(v) & Df(v)^T \\ Df(v) & -I \end{pmatrix}.$$

It is clear that the inverse of the above map is given by

$$x \longmapsto (\text{Proj}_{y+T_y\mathcal{M}}(\text{Proj}_{\mathcal{M}}(x)), \text{Proj}_{T_y^\perp\mathcal{M}}(x - \text{Proj}_{\mathcal{M}}(x))),$$

which is at least a C^1 map. Thus, a necessary condition for the τ-radius normal bundle to embed is that the Jacobian exhibited above is invertible, which in turn implies that

$$\begin{pmatrix} I + \sum_{i=1}^{D-d}\beta_i D^2 f_i(v) & Df(v)^T \\ Df(v) & -I \end{pmatrix}\begin{pmatrix} \zeta \\ Df(v)\zeta \end{pmatrix} \neq 0$$

for all $\zeta \neq 0$ when $(v,\beta) \in \Omega$. This reduces to $(I + \sum \beta_i D^2 f_i(v) + Df(v)^T Df(v))\zeta \neq 0$, and so a necessary condition for embedding is then that the norm of $\sum_{i=1}^{D-d}\beta_i D^2 f_i(v)$ does not exceed $1 + \|Df(v)\|^2$ whenever

$$\left\|\begin{pmatrix} Df(v)^T \\ -I \end{pmatrix}\beta\right\|^2 = \|\beta\|^2 + \|Df(v)^T\beta\|^2 < \tau^2.$$

In particular, this must be true if $\|\beta\| < \tau/\sqrt{1+\|Df(v)\|^2}$. This reduces to the condition that the operator norm

$$\sup_{u\in\mathcal{S}^{D-d-1}}\left\|\sum_{i=1}^{D-d} u_i D^2 f_i(v)\right\| < \frac{(1+\|Df(v)\|^2)^{3/2}}{\tau} < \frac{1}{\tau}\left(1+\|Df(v)\|\right)^3. \tag{28}$$

By the fundamental theorem of calculus, we have that

$$Df(v)x = Df(0)x + \int_0^{\|v\|}[u_v^T D^2 f_i(tu_v)x]dt = \int_0^{\|v\|}[u_v^T D^2 f_i(tu_v)x]dt,$$

where $u_v = v/\|v\|$ and $[u_v^T D^2 f_i(tu_v)x]$ indicates a vector with ith component $u_v^T D^2 f_i(tu_v)x$. Consequently, for any $x \in \mathbb{R}^d$, we have that

$$\begin{aligned}
\|Df(v)x\| &\leq \int_0^{\|v\|}\left\|[u_v^T D^2 f_i(tu_v)x]\right\| dt \leq \|v\| \sup_{t\in[0,\|v\|]}\left\|[u_v^T D^2 f_i(tu_v)x]\right\| \\
&\leq \varepsilon \sup_{t\in[0,\varepsilon]}\left\|[u_v^T D^2 f_i(tu_v)x]\right\|.
\end{aligned} \tag{29}$$

Now,

$$\begin{aligned}
\left\|[u_v^T D^2 f_i(tu_v)x]\right\| &= \sup_{u\in\mathcal{S}^{D-d-1}}\langle u, [u_v^T D^2 f_i(tu_v)x]\rangle = \sup_{u\in\mathcal{S}^{D-d-1}}\sum_{i=1}^{D-d} u_i(u_v^T D^2 f_i(tu_v)x) \\
&= \sup_{u\in\mathcal{S}^{D-d-1}} u_v^T\left(\sum_{i=1}^{D-d} u_i D^2 f_i(tu_v)\right)x \\
&\leq \sup_{u\in\mathcal{S}^{D-d-1}}\|u_v\|\left\|\sum_{i=1}^{D-d} u_i D^2 f_i(tu_v)\right\|\|x\| \\
&= \|x\|\sup_{u\in\mathcal{S}^{D-d-1}}\left\|\sum_{i=1}^{D-d} u_i D^2 f_i(tu_v)\right\|,
\end{aligned}$$

which together with (29) and (28) yields the bound

$$\|Df(v)\| < \frac{\varepsilon}{\tau}\left(1 + \sup_{t\in[0,\varepsilon]}\|Df(tu_v)\|\right)^3.$$

Since this inequality also holds for any v' with $\|v\| \leq \varepsilon'$, taking a supremum yields

$$\sup_{\varepsilon'\in[0,\varepsilon]}\|Df(tu_v)\| \leq \sup_{\varepsilon'\in[0,\varepsilon]}\frac{\varepsilon'}{\tau}\left(1+\sup_{t\in[0,\varepsilon']}\|Df(tu_v)\|\right)^3 \leq \frac{\varepsilon}{\tau}\left(1+\sup_{\varepsilon'\in[0,\varepsilon]}\|Df(tu_v)\|\right)^3,$$

and hence

$$\sup_{v\in B_d(0,\varepsilon)}\|Df(v)\| \leq \frac{\varepsilon}{\tau}\left(1+\sup_{v\in B_d(0,\varepsilon)}\|Df(v)\|\right)^3.$$

Setting $a(\varepsilon') = \sup_{v \in B_d(0,\varepsilon')} \|Df(v)\|$, we have that $a(0) = 0$,

$$a(\varepsilon') \leq \frac{\varepsilon'}{\tau}\left(1 + a(\varepsilon')\right)^3,$$

for all $\varepsilon' \geq 0$, and a is continuous by continuity of $\|Df(v)\|$. Setting $b(\varepsilon') = a(\varepsilon')/(1+a(\varepsilon'))$, we get

$$b(\varepsilon')(1 - b(\varepsilon'))^2 \leq \frac{\varepsilon'}{\tau}.$$

Examining the polynomial $x(1-x)^2$, we see that the sublevel set $x(1-x)^2 \leq \omega$ consists of two components when $\omega < 4/27$. Also note that if $\omega < 1/8$, then

$$2(1-2\omega)^2 = 2 - 8\omega + 8\omega^2 > 2 - 1 = 1,$$

and hence

$$2\omega(1-2\omega)^2 > \omega.$$

Consequently, if x is such that $x(1-x)^2 \leq \omega$ and is in the interval containing zero in the sublevel set $x(1-x)^2 \leq \omega < 1/8$, then $x \leq 2\omega$.

By these observations, continuity of $b(\varepsilon')$, and the fact that $b(0) = 0$, we have that $a(\varepsilon') \leq \frac{2\frac{\varepsilon'}{\tau}}{1-2\frac{\varepsilon'}{\tau}}$, and thus

$$\sup_{v \in B_d(0,\varepsilon)} \|Df(v)\| \leq \frac{2\varepsilon}{\tau - 2\varepsilon}.$$

From the bound in (28) we now acquire the bound

$$\sup_{v \in B_d(0,\varepsilon)} \sup_{u \in \mathcal{S}^{D-d-1}} \left\| \sum_{i=1}^{D-d-1} u_i D^2 f_i(v) \right\| \leq \frac{\tau^2}{(\tau - 2\varepsilon)^3}.$$

■

Proof [Proof of Lemma 18] We first prove part *i*. Let $\varepsilon > 0$ satisfy $\varepsilon < \tau/8$. Because of (23) and the fact that $\|\beta\| \leq \sigma$, we have that

$$\left\| \sum_{i=1}^{D-d} \beta_i D^2 f_i(v) \right\| \leq \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}.$$

Since this is also a bound for the columns of $\sum \beta_i D^2 f_i(v)$, Proposition 10 implies that

$$\mathrm{Vol}\begin{pmatrix} I + \sum \beta_i D^2 f_i(v) & Df(v)^T \\ Df(v) & -I \end{pmatrix} \leq \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}\right)^d \mathrm{Vol}\begin{pmatrix} I & Df(v)^T \\ Df(v) & -I \end{pmatrix}$$

in $T^{\perp}(\mathcal{M} \cap B(y,\varepsilon)) \cap \mathcal{M}_\sigma$.

On the other hand, we have that

$$\mathrm{Vol}\begin{pmatrix} Df^T(v) \\ -I \end{pmatrix} \leq \prod_{i=1}^{D-d} \sqrt{1 + \|\nabla f_i(v)\|^2} \leq \left(1 + \frac{4\varepsilon^2}{(\tau - 2\varepsilon)^2}\right)^{(D-d)/2}$$

since (22) implies the bounds $\|\frac{\partial f(v)}{\partial v_i}\| \leq \frac{2\varepsilon}{\tau - 2\varepsilon}$ for each $i = 1, \ldots, d$, and the above is the largest this quantity may be subject to these bounds.

When these estimates are joined together, we have an inequality

$$\begin{aligned}\mathrm{Vol}\begin{pmatrix} I + \sum_{i=1}^{D-d} \beta_i D^2 f_i(v) & Df(v)^T \\ Df(v) & -I \end{pmatrix} &\leq \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}\right)^d \mathrm{Vol}\begin{pmatrix} I & Df(v)^T \\ Df(v) & -I \end{pmatrix} \\ &\leq \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}\right)^d \left(1 + \frac{4\varepsilon^2}{(\tau - 2\varepsilon)^2}\right)^{(D-d)/2} \mathrm{Vol}\begin{pmatrix} I \\ Df(v) \end{pmatrix}.\end{aligned}$$

For an arbitrarily small $\varepsilon > 0$, let $\{U_\gamma\}_{\gamma \in \Gamma}$ denote a finite partition of U into measurable sets such that there for each $\gamma \in \Gamma$, there is a y_γ satisfying $U_\gamma \subset \mathcal{M} \cap B(y_\gamma, \varepsilon)$. Let f_γ denote the inverse of $P_\gamma = \mathrm{Proj}_{y_\gamma + T_{y_\gamma}\mathcal{M}}$ in U_γ, and set

$$E_{\gamma,v} = \{\beta \in \mathbb{R}^{D-d} : \|\beta\|^2 + \|Df_\gamma(v)\beta\|^2 \leq \sigma^2\}$$

for all $v \in P_\gamma(U_\gamma)$. Thus,

$$\begin{aligned}\int_{P_\gamma^{-1}(U_\gamma)} d\mathrm{Vol}(x) &= \int_{P_\gamma(U_\gamma)} \int_{E_{\gamma,v}} \mathrm{Vol}\begin{pmatrix} I + \sum_{i=1}^{D-d} \beta_i D^2 f_i(v) & Df(x)^T \\ Df(v) & -I \end{pmatrix} d\beta dv \\ &\leq \int_{P_\gamma(U_\gamma)} \int_{E_{\gamma,v}} \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}\right)^d \left(1 + \frac{4\varepsilon^2}{(\tau - 2\varepsilon)^2}\right)^{(D-d)/2} \mathrm{Vol}\begin{pmatrix} I \\ Df(v) \end{pmatrix} d\beta dv \\ &\leq \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^2}\right)^d \left(1 + \frac{4\varepsilon^2}{(\tau - 2\varepsilon)^2}\right)^{(D-d)/2} \mathrm{Vol}_{\mathcal{M}}(U_\gamma)\mathrm{Vol}(B_{D-d}(0,\sigma))\end{aligned}$$

since $E_{\gamma,v} \subset B_{D-d}(0,\sigma)$. Consequently, we have that

$$\begin{aligned}\mathrm{Vol}(P^{-1}(U)) &= \sum_{\gamma \in \Gamma} \mathrm{Vol}(P_\gamma^{-1}(U_\gamma)) \\ &\leq \sum_{\gamma \in \Gamma} \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}\right)^d \left(1 + \frac{4\varepsilon^2}{(\tau - 2\varepsilon)^2}\right)^{(D-d)/2} \mathrm{Vol}_{\mathcal{M}}(U_\gamma)\mathrm{Vol}(B_{D-d}(0,\sigma)) \\ &= \left(1 + \frac{\sigma\tau^2}{(\tau - 2\varepsilon)^3}\right)^d \left(1 + \frac{4\varepsilon^2}{(\tau - 2\varepsilon)^2}\right)^{(D-d)/2} \mathrm{Vol}_{\mathcal{M}}(U)\mathrm{Vol}(B_{D-d}(0,\sigma)).\end{aligned}$$

Since $\varepsilon > 0$ was arbitrary, we obtain

$$\mathrm{Vol}(P^{-1}(U)) \cap \mathcal{M}_\sigma) \leq \left(1 + \frac{\sigma}{\tau}\right)^d \mathrm{Vol}_{\mathcal{M}}(U)\mathrm{Vol}(B_{D-d}(0,\sigma)).$$

This completes the proof of upper bound in part *i*. Using a similar partition strategy, we have that

$$\begin{aligned}
\int_{P_\gamma^{-1}(U_\gamma)} d\mathrm{Vol}(x) &= \int_{P_\gamma(U_\gamma)} \int_{E_{\gamma,v}} \mathrm{Vol}\begin{pmatrix} I + \sum_{i=1}^{D-d} \beta_i D^2 f_i(v) & Df(x)^T \\ Df(v) & -I \end{pmatrix} d\beta dv \\
&\geq \int_{P_\gamma(U_\gamma)} \int_{E_{\gamma,v}} \left(1 - \frac{\sigma\tau^2}{(\tau-2\varepsilon)^3}\right)^d \mathrm{Vol}\begin{pmatrix} I & Df(v)^T \\ Df(v) & -I \end{pmatrix} d\beta dv \\
&= \int_{P_\gamma(U_\gamma)} \int_{E_{\gamma,v}} \left(1 - \frac{\sigma\tau^2}{(\tau-2\varepsilon)^3}\right)^d \mathrm{Vol}\begin{pmatrix} I \\ Df(v) \end{pmatrix} \mathrm{Vol}\begin{pmatrix} Df(v)^T \\ -I \end{pmatrix} d\beta dv \\
&\geq \int_{P_\gamma(U_\gamma)} \int_{E_{\gamma,v}} \left(1 - \frac{\sigma\tau^2}{(\tau-2\varepsilon)^3}\right)^d \mathrm{Vol}\begin{pmatrix} I \\ Df(v) \end{pmatrix} d\beta dv \\
&\geq \int_{P_\gamma(U_\gamma)} \int_{B_{D-d}\left(0, \frac{\sigma}{1+\frac{\varepsilon}{\tau-\varepsilon}}\right)} \left(1 - \frac{\sigma\tau^2}{(\tau-2\varepsilon)^3}\right)^d \mathrm{Vol}\begin{pmatrix} I \\ Df(v) \end{pmatrix} d\beta dv \\
&= \left(1 - \frac{\sigma\tau^2}{(\tau-2\varepsilon)^3}\right)^d \mathrm{Vol}_{\mathcal{M}}(U_\gamma) \mathrm{Vol}\left(B_{D-d}\left(0, \left(1-\frac{\varepsilon}{\tau}\right)\sigma\right)\right)
\end{aligned}$$

In the inequalities above, we have used the fact that there is a ball of radius $\left(1-\frac{\varepsilon}{\tau}\right)\sigma$ inside of $E_{\gamma,v}$ for each γ and each v. Aggregating all of the sums and letting $\varepsilon \to 0$ yields the lower bound in part *i*.

We now prove part *ii*. Note that

$$\mathrm{Vol}(\mathcal{M}_\sigma \cap B(y,r)) \leq \mathrm{Vol}(P^{-1}(\mathcal{M} \cap B(y, r+\sigma)))$$

since $\|\operatorname{Proj}_{\mathcal{M}}(x) - y\| \leq \|x-y\| + \|\operatorname{Proj}_{\mathcal{M}}(x) - x\| \leq r + \sigma$. Part *ii*. now follows from part *i*. and the fact that

$$\begin{aligned}
\mathrm{Vol}_{\mathcal{M}}(\mathcal{M} \cap B(y, r+\sigma)) &\leq \int\limits_{P(\mathcal{M}\cap B(y,r+\sigma))} \mathrm{Vol}\begin{pmatrix} I \\ Df(v) \end{pmatrix} dv \\
&\leq \left(1 + \left(\frac{2(r+\sigma)}{\tau - 2(r+\sigma)}\right)^2\right)^{d/2} \mathrm{Vol}(B_d(0, r+\sigma)).
\end{aligned}$$

■

Proof [Proof of Lemma 20] By the variational characterization of eigenvalues, we have that

$$\begin{aligned}
\sum_{i=d+1}^{D} \lambda_i(\Sigma) &= \operatorname*{argmin}_{\dim(V)=D-d} \mathrm{tr}\left(\operatorname{Proj}_V^T \Sigma \operatorname{Proj}_V\right) \\
&= \operatorname*{argmin}_{\dim(V)=D-d} \mathbb{E}\|\operatorname{Proj}_V(Z - \mathbb{E}Z)\|^2 \\
&= \operatorname*{argmin}_{\dim(V)=d} \mathbb{E}\|Z - \mathbb{E}Z - \operatorname{Proj}_V(Z - \mathbb{E}Z)\|^2.
\end{aligned}$$

Thus, we have that $\sum_{i=d+1}^{D} \lambda_i(\Sigma) \leq \mathbb{E}\|Z - \mathbb{E}Z - \operatorname{Proj}_{T_y\mathcal{M}}(Z - \mathbb{E}Z)\|^2$. Observe that

$$\begin{aligned}
\mathbb{E}\|Z - \mathbb{E}Z - \operatorname{Proj}_{T_y\mathcal{M}}(Z - \mathbb{E}Z)\|^2 =& \mathbb{E}\|Z - y + (y - \mathbb{E}Z) - \operatorname{Proj}_{T_y\mathcal{M}}((Z-y) + (y - \mathbb{E}Z))\|^2 \\
=& \mathbb{E}\|Z - y - \operatorname{Proj}_{T_y\mathcal{M}}(Z-y)\|^2 \\
&- \|(y - \mathbb{E}Z) - \operatorname{Proj}_{T_y\mathcal{M}}(y - \mathbb{E}Z)\|^2 \\
&\leq \mathbb{E}\|Z - y - \operatorname{Proj}_{T_y\mathcal{M}}(Z-y)\|^2.
\end{aligned}$$

Now for any $z \in \mathcal{M}_\sigma \cap B(y,r)$, we have that $z = \beta + x$ where $x \in \mathcal{M}$, and $\beta \in T_x^\perp \mathcal{M}$ satisfies $\|\beta\| \leq \sigma$. Moreover, there is a unique decomposition $x = \eta + v + y$ where $\eta \in T_y^\perp \mathcal{M}$ and $v \in T_y\mathcal{M}$. Thus,

$$\|z - y - \operatorname{Proj}_{T_y\mathcal{M}}(z-y)\| = \|\beta + \eta - \operatorname{Proj}_{T_y\mathcal{M}}\beta\| \leq \|\beta - \operatorname{Proj}_{T_y\mathcal{M}}(\beta)\| + \|\eta\| \leq \sigma + \frac{2r^2}{\tau}, \tag{30}$$

by Lemma 15, and we obtain the bound

$$\mathbb{E}\|Z - \mathbb{E}Z - \operatorname{Proj}_{T_y\mathcal{M}}(Z - \mathbb{E}Z)\|^2 \leq 2\sigma^2 + \frac{8r^4}{\tau^2}. \tag{31}$$

This establishes the required estimate. ■

Proof [Proof of Lemma 21] For any unit vector $u \in T_y\mathcal{M}$ we have

$$\begin{aligned}
\mathbb{E}\langle u, Z - \mathbb{E}Z\rangle^2 &= \frac{1}{\mathrm{Vol}(Q \cap \mathcal{M}_\sigma)} \int_{Q \cap \mathcal{M}_\sigma} \langle u, Z - \mathbb{E}Z\rangle^2 d\mathrm{Vol}(Z) \\
&\geq \frac{1}{\mathrm{Vol}(B(y, r_2) \cap \mathcal{M}_\sigma)} \int_{B(y,r_1)\cap\mathcal{M}_\sigma} \langle u, (Z-y) - \mathbb{E}(Z-y)\rangle^2 d\mathrm{Vol}(Z)
\end{aligned}$$

using the inclusion assumptions, and by adding and subtracting the constant vector y.

We now seek to reduce the domain of integration and perform a change of variables. Since $r_1 \leq \tau/8$, the inverse of the affine projection onto $y + T_y\mathcal{M}$ is injective. Without loss of generality, we assume $y = 0$ and $T_y\mathcal{M}$ is the span of the first d standard orthonormal vectors. Letting f denote the inverse of the affine projection onto $y + T_y\mathcal{M}$, we see that the map

$$\begin{pmatrix} v \\ \beta \end{pmatrix} \longmapsto \begin{pmatrix} v \\ f(v) + \beta \end{pmatrix}$$

is well-defined and injective on $\operatorname{Proj}_{T_y\mathcal{M}}(\mathcal{M} \cap B(y, r_1 - \sigma)) \times (T_y^\perp \mathcal{M} \cap B(0, \sigma))$. Let g denote this map, note that

$$\|x + \beta\| \leq \|x\| + \|\beta\| \leq (r - \sigma) + \sigma = r,$$

for $x \in \mathcal{M} \cap B(y, r_1 - \sigma)$, and hence the image of g is contained in $\mathcal{M}_\sigma \cap B(y, r_1)$. Since the absolute value of the determinant of the Jacobian of g is always 1 (it is lower triangular

with ones on the diagonal), employing the change of coordinates in the reduced domain of integration yields

$$\mathbb{E}\langle u, Z-\mathbb{E}Z\rangle^2 \geq \frac{1}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{\mathcal{A}}\int_{\mathcal{B}}\left\langle \begin{pmatrix} u \\ 0 \end{pmatrix}, \begin{pmatrix} v \\ f(v)+\beta \end{pmatrix} - \mathbb{E}(Z-y)\right\rangle^2 d\beta dv,$$

where

$$\mathcal{A} = \mathrm{Proj}_{T_y\mathcal{M}}(B(y,r_1-\sigma)\cap\mathcal{M}),\ \mathcal{B} = T_y^\perp\mathcal{M}\cap B(0,\sigma).$$

Note that $B(y,\cos(\theta)(r_1-\sigma))\cap(y+T_y\mathcal{M})\subset\mathcal{A}$. Setting $\mathcal{Q}=\mathrm{Proj}_{T_y\mathcal{M}}$, this immediately reduces to

$$\begin{aligned}\mathbb{E}\langle u, Z-\mathbb{E}Z\rangle^2 &\geq \frac{1}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{\mathcal{A}}\int_{\mathcal{B}}\langle u, v-\mathbb{E}\mathcal{Q}(Z-y)\rangle^2 d\beta dv\\ &= \frac{\mathrm{Vol}(B_{D-d}(0,\sigma))}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{\mathcal{A}}\langle u, v-\mathbb{E}\mathcal{Q}(Z-y)\rangle^2 dv\\ &\geq \frac{\mathrm{Vol}(B_{D-d}(0,\sigma))}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{B_d(0,q)}\langle u, v-\mathbb{E}\mathcal{Q}(Z-y)\rangle^2 dv,\end{aligned}$$

where $q=\cos(\delta)(r_1-\sigma)$ and $\delta=\arcsin((r_1-\sigma)/2\tau)$. Noting that $\int_{B_d(0,q)}\langle u,v\rangle dv = 0$ by symmetry, we now use linearity of the inner product to further reduce the integrand:

$$\begin{aligned}\mathbb{E}\langle u, Z-\mathbb{E}Z\rangle^2 &\geq \frac{\mathrm{Vol}(B_{D-d}(0,\sigma))}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{B_d(0,q)}\left(\langle u,v\rangle^2 - 2\langle u,v\rangle\langle u,\mathbb{E}\mathcal{Q}(Z-y)\rangle + \langle u,\mathbb{E}\mathcal{Q}(Z-y)\rangle^2\right)dv\\ &= \frac{\mathrm{Vol}(B_{D-d}(0,\sigma))}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{B_d(0,q)}\left(\langle u,v\rangle^2 + \langle u,\mathbb{E}\mathcal{Q}(Z-y)\rangle^2\right)dv\\ &\geq \frac{\mathrm{Vol}(B_{D-d}(0,\sigma))}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\int_{B_d(0,q)}\langle u,v\rangle^2 dv\\ &= \frac{\mathrm{Vol}(B_{D-d}(0,\sigma))\mathrm{Vol}(B_d(0,q))}{\mathrm{Vol}(B(y,r_2)\cap\mathcal{M}_\sigma)}\frac{q^2}{d}.\end{aligned}$$

By Lemma 18, we then obtain

$$\begin{aligned}\mathbb{E}\langle u, Z-\mathbb{E}Z\rangle^2 &\geq \left(\left(1+\frac{\sigma}{\tau}\right)\sqrt{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}\right)^{-d}\frac{\mathrm{Vol}(B_d(0,q))}{\mathrm{Vol}(B_d(0,r_2+\sigma))}\frac{q^2}{d}\\ &\geq \frac{1}{4\left(1+\frac{\sigma}{\tau}\right)^d}\left(\frac{r_1-\sigma}{r_2+\sigma}\right)^d\left(\frac{1-\left(\frac{r_1-\sigma}{2\tau}\right)^2}{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}\right)^{d/2}\frac{(r_1-\sigma)^2}{d}. \qquad (32)\end{aligned}$$

Let $V_{d-1}(\Sigma)$ be a subspace corresponding to the first $d-1$ principal components of Z:

$$V_{d-1} = \operatorname*{argmin}_{\dim(V)=d-1}\mathbb{E}\|Z-\mathbb{E}Z-\mathrm{Proj}_V(Z-\mathbb{E}Z)\|,$$

and note that $\lambda_d(\Sigma)=\max_{0\neq u\in V_{d-1}^\perp}\mathbb{E}\left\langle\frac{u}{\|u\|}, Z-\mathbb{E}Z\right\rangle^2$. Since $\dim(V_{d-1}^\perp)=D-d+1$ and $\dim(T_y\mathcal{M})=d$, it is easy to see that $V_{d-1}^\perp\cap T_y\mathcal{M}\neq\emptyset$. For any $u_*\in V_{d-1}^\perp\cap T_y\mathcal{M}$ such that

$\|u_*\|=1$ it follows from Courant-Fischer characterization of $\lambda_d(\Sigma)$ that

$$\lambda_d(\Sigma)\geq\mathbb{E}\langle u_*, Z-\mathbb{E}Z\rangle^2,$$

and (32) implies the desired bound. ■

Proof [Proof of Lemma 22] Let $Q\subset\mathbb{R}^D$ be such that $B(y,r_1)\subset Q$ and $\mathcal{M}_\sigma\cap Q\subset B(y,r_2)$ for some $y\in\mathcal{M}$ and $\sigma<r_1<r_2<\tau/8-\sigma$. Assume that Z is drawn from $U_{\mathcal{M}_\sigma\cap Q}$, let Σ be the covariance matrix of Z and $V_d:=V_d(\Sigma)$ - the subspace corresponding to the first d principal components of Z.

Let $\alpha\in[0,1]$ be such that $\cos(\phi):=\min_{u\in V_d,\|u\|=1}\max_{v\in T_y\mathcal{M},\|v\|=1}|\langle u,v\rangle| = \sqrt{1-\alpha^2}$ is the cosine of the angle between $T_y\mathcal{M}$ and V_d. Then there exists a unit vector $u_*\in(V_d)^\perp$ such that

$$\max_{v\in T_y\mathcal{M},\|v\|=1}|\langle u_*, v\rangle|\geq\alpha.$$

Indeed, let $u'\in V_d$, $v'\in T_y\mathcal{M}$ be unit vectors such that $\cos(\phi)=\langle u',v'\rangle$, Note that $\sqrt{1-\alpha^2}$ is equal to the smallest absolute value among the nonzero singular values of the operator $\mathrm{Proj}_{T_y\mathcal{M}}\mathrm{Proj}_{V_d}$. Since the spectra of the operators $\mathrm{Proj}_{T_y\mathcal{M}}\mathrm{Proj}_{V_d}$ and $\mathrm{Proj}_{V_d}\mathrm{Proj}_{T_y\mathcal{M}}$ coincide by well-known facts from linear algebra, we have that

$$\min_{u\in V_d,\|u\|=1}\max_{v\in T_y\mathcal{M},\|v\|=1}|\langle u,v\rangle| = \min_{v\in T_y\mathcal{M},\|v\|=1}\max_{u\in V_d,\|u\|=1}|\langle u,v\rangle|.$$

In other words, $\mathrm{Proj}_{T_y\mathcal{M}}(u')=\langle u',v'\rangle v'$ and $\mathrm{Proj}_{V_d}(v')=\langle u',v'\rangle u'$. This implies that there exists a unit vector $u_*\in(V_d)^\perp$ such that $v'=\langle v',u'\rangle u'+\langle v',u_*\rangle u_*$, hence $\langle u_*,v'\rangle^2 = 1-\langle v',u'\rangle^2=\alpha^2$, so u_* satisfies the requirement.

To simplify the expressions, let

$$\zeta=\frac{1}{\mathrm{Vol}(Q\cap\mathcal{M}_\sigma)}.$$

We shall now construct upper and lower bounds for

$$\zeta\int_{Q\cap\mathcal{M}_\sigma}\langle u_*, x-\mathbb{E}Z-\mathrm{Proj}_{V_d}(x-\mathbb{E}Z)\rangle^2\,d\mathrm{Vol}(x) = \zeta\int_{Q\cap\mathcal{M}_\sigma}\langle u_*, x-\mathbb{E}Z\rangle^2\,d\mathrm{Vol}(x)$$

which together yield an estimate for α. Write $u_*=u_*^{||}+u_*^\perp$, where $u_*^{||}\in T_y\mathcal{M}$ and $u_*^\perp\in T_y^\perp\mathcal{M}$. By our choice of u_*, we clearly have that $\|u_*^{||}\|=\max_{v\in T_y\mathcal{M},\|v\|=1}\langle u_*,v\rangle\geq\alpha$. Using the elementary inequality $(a+b)^2\geq\frac{a^2}{2}-b^2$, we further deduce that

$$\begin{aligned}\zeta\int_{Q\cap\mathcal{M}_\sigma}\langle u_*, x-\mathbb{E}Z\rangle^2\,d\mathrm{Vol}(x) &\geq \zeta\int_{Q\cap\mathcal{M}_\sigma}\frac{1}{2}\left\langle u_*^{||}, x-\mathbb{E}Z\right\rangle^2 d\mathrm{Vol}(x)\\ &\quad-\zeta\int_{Q\cap\mathcal{M}_\sigma}\left\langle u_*^\perp, x-\mathbb{E}Z\right\rangle^2 d\mathrm{Vol}(x). \qquad (33)\end{aligned}$$

It follows from the proof of Lemma 21 that

$$\zeta\int_{Q\cap\mathcal{M}_\sigma}\frac{1}{2}\left\langle u_*^{||}, x-\mathbb{E}Z\right\rangle^2 d\mathrm{Vol}(x)\geq\frac{\alpha^2}{8\left(1+\frac{\sigma}{\tau}\right)^d}\left(\frac{r_1-\sigma}{r_2+\sigma}\right)^d\left(\frac{1-\left(\frac{r_1-\sigma}{2\tau}\right)^2}{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}\right)^{d/2}\frac{(r_1-\sigma)^2}{d}.$$

For the last term in (33), Lemma 20 (see equation (31)) gives

$$\zeta \int_{Q\cap\mathcal{M}_\sigma} \left\langle u_*^\perp, x-\mathbb{E}Z\right\rangle^2 d\mathrm{Vol}(x) \le \zeta \int_{Q\cap\mathcal{M}_\sigma} \|x-\mathbb{E}Z-\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z)\|^2 d\mathrm{Vol}(x)$$
$$\le 2\sigma^2 + \frac{8r_2^4}{\tau^2},$$

hence (33) yields

$$\zeta \int_{Q\cap\mathcal{M}_\sigma} \langle u_*, x-\mathbb{E}Z\rangle^2 \, d\mathrm{Vol}(x) \ge \frac{\alpha^2}{8\left(1+\frac{\sigma}{\tau}\right)^d}\left(\frac{r_1-\sigma}{r_2+\sigma}\right)^d \left(\frac{1-\left(\frac{r_1-\sigma}{2\tau}\right)^2}{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}\right)^{d/2} \frac{(r_1-\sigma)^2}{d}$$
$$- 2\sigma^2 - \frac{8r_2^4}{\tau^2}. \tag{34}$$

On the other hand, invoking (31) once again, we have

$$\zeta \int_{Q\cap\mathcal{M}_\sigma} \langle u_*, x-\mathbb{E}Z\rangle^2 \, d\mathrm{Vol}(x) \le 2\sigma^2 + \frac{8r_2^4}{\tau^2}.$$

Combined with (34), this gives

$$\frac{\alpha^2}{8\left(1+\frac{\sigma}{\tau}\right)^d}\left(\frac{r_1-\sigma}{r_2+\sigma}\right)^d \left(\frac{1-\left(\frac{r_1-\sigma}{2\tau}\right)^2}{1+\left(\frac{2(r_2+\sigma)}{\tau-2(r_2+\sigma)}\right)^2}\right)^{d/2} \frac{(r_1-\sigma)^2}{d} \le 4\sigma^2 + \frac{16r_2^4}{\tau^2}, \tag{35}$$

and the upper bound for α follows.

Notice that for any $x \in Q\cap\mathcal{M}_\sigma$,

$$x-\mathbb{E}Z-\mathrm{Proj}_{V_d}(x-\mathbb{E}Z) = x-y-\mathrm{Proj}_{T_y\mathcal{M}}(x-y) + \underbrace{y-\mathbb{E}Z-\mathrm{Proj}_{T_y\mathcal{M}}(y-\mathbb{E}Z)}_{\mathrm{Proj}_{(T_y\mathcal{M})^\perp}(y-\mathbb{E}Z)}$$
$$+ (\mathrm{Proj}_{T_y\mathcal{M}} - \mathrm{Proj}_{V_d})(x-\mathbb{E}Z). \tag{36}$$

It follows from (30) that

$$\|x-y-\mathrm{Proj}_{T_y\mathcal{M}}(x-y)\| = \left\|\mathrm{Proj}_{T_y^\perp\mathcal{M}}(x-y)\right\| \le \sigma + \frac{2r_2^2}{\tau}.$$

Next,

$$\|\mathrm{Proj}_{(T_y\mathcal{M})^\perp}(y-\mathbb{E}Z)\| = \frac{1}{\mathrm{Vol}(Q\cap\mathcal{M}_\sigma)}\left\|\int_{Q\cap\mathcal{M}_\sigma} \mathrm{Proj}_{T_y^\perp\mathcal{M}}(y-z)d\mathrm{Vol}(z)\right\|$$
$$\le \frac{1}{\mathrm{Vol}(Q\cap\mathcal{M}_\sigma)}\int_{Q\cap\mathcal{M}_\sigma}\left\|\mathrm{Proj}_{T_y^\perp\mathcal{M}}(z-y)\right\| d\mathrm{Vol}(z)$$
$$\le \sigma + \frac{2r_2^2}{\tau}.$$

Finally, it is easy to see that

$$\|(\mathrm{Proj}_{T_y\mathcal{M}} - \mathrm{Proj}_{V_d})(x-\mathbb{E}Z)\| \le \|\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z) - \mathrm{Proj}_{V_d}\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z)\|$$
$$+ \|\mathrm{Proj}_{T_y^\perp\mathcal{M}}(x-y)\| + \|\mathrm{Proj}_{T_y^\perp\mathcal{M}}(\mathbb{E}Z-y)\|.$$

Let $u_x := \frac{\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z)}{\|\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z)\|}$ and note that for any $x \in Q\cap\mathcal{M}_\sigma$, $\|\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z)\| \le 2r_2$, hence

$$\|\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z) - \mathrm{Proj}_{V_d}\mathrm{Proj}_{T_y\mathcal{M}}(x-\mathbb{E}Z)\|^2 \le (2r_2)^2\left(1-\|\mathrm{Proj}_{V_d}u_x\|^2\right)$$
$$\le 4r_2^2\left(1-\min_{u\in T_y\mathcal{M},\|u\|=1}\max_{v\in V_d,\|v\|=1}\langle u,v\rangle^2\right)$$
$$= 4r_2^2\alpha^2.$$

Combining the previous bounds with (35) and (36), we obtain the result. ∎

Proof [Proof of Lemma 24] Assume the event $\mathcal{E}_{\varepsilon/2,n} = \{\{Y_1,\dots,Y_n\}$ is an $\varepsilon/2$ - net in $\mathcal{M}\}$ occurs. By Proposition 23, $\Pr(\mathcal{E}_{\varepsilon/2,n}) \ge 1-e^{-t}$.

Since the elements of T_j are 2^{-j}-separated, for any $1 \le k \le N(j)$, $B(a_{j,k}, 2^{-j-1}) \subseteq C_{j,k}$. Moreover, since $\sigma \le 2^{-j-2}$ and $\|a_{j,k}-z_{j,k}\| \le \sigma$,

$$B(z_{j,k}, 2^{-j-1}-2^{-j-2}) \subseteq B(z_{j,k}, 2^{-j-1}-\sigma) \subseteq B(a_{j,k}, 2^{-j-1}),$$

hence the inclusion $B\left(z_{j,k}, 2^{-j-2}\right) \subseteq C_{j,k}$ follows.

To show that $C_{j,k}\cap\mathcal{M}_\sigma \subseteq B(a_{j,k}, 3\cdot 2^{-j-2}+2^{-j+1})$, pick an arbitrary $z \in \mathcal{M}_\sigma$. Note that on the event $\mathcal{E}_{\varepsilon/2,n}$, there exists $y \in \{Y_1,\dots,Y_n\}$ satisfying $\|z-y\| \le \varepsilon/2+\sigma$. Let $x(y) \in \mathcal{X}_n$ be such that $y = \mathrm{Proj}_{\mathcal{M}}(x(y))$. By properties of the cover trees (see Remark 5), there exists $x_* \in T_j$ such that $\|x(y)-x_*\| \le 2^{-j+1}$. Then

$$\|z-x_*\| \le \|z-y\| + \|y-x(y)\| + \|x(y)-x_*\| \le \varepsilon/2 + 2\sigma + 2^{-j+1} \le 3\cdot 2^{-j-2} + 2^{-j+1}.$$

Since z was arbitrary, the result follows. Finally, $B(a_{j,k}, 3\cdot 2^{-j-2}+2^{-j+1}) \subset B(z_{j,k}, 3\cdot 2^{-j})$ holds since $\|a_{j,k}-z_{j,k}\| \le 2^{-j-2}$. ∎

References

M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11): 4311–4322, 2006.

W.K. Allard, G. Chen, and M. Maggioni. Multi-scale geometric methods for data sets II: Geometric multi-resolution analysis. *Applied and Computational Harmonic Analysis*, 32 (3):435–462, 2012. ISSN 1063-5203.

M. Belkin and P. Niyogi. Using manifold structure for partially labelled classification. *Advances in NIPS*, 15, 2003.

A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd International Conference on Machine learning*, pages 97–104. ACM, 2006.

Paul S Bradley and Olvi L Mangasarian. K-plane clustering. *Journal of Global Optimization*, 16(1):23–32, 2000.

F. Camastra and A. Vinciarelli. Intrinsic dimension estimation of data: an approach based on Grassberger–Procaccia's algorithm. *Neural Processing Letters*, 14(1):27–34, 2001.

G. Canas, T. Poggio, and L. Rosasco. Learning manifolds with K-Means and K-Flats. In *Advances in Neural Information Processing Systems 25*, pages 2474–2482, 2012.

E. Candes and T. Tao. The Dantzig selector: statistical estimation when p is much larger than n. *Annals of Statistics*, (6):2313–2351, 2007. math.ST/0506081.

E. Causevic, R.R. Coifman, R. Isenhart, A. Jacquin, E.R. John, M. Maggioni, L.S. Prichep, and F.J. Warner. QEEG-based classification with wavelet packets and microstate features for triage applications in the ER. volume 3. ICASSP Proc., May 2006.

G. Chen and Gilad Lerman. Foundations of a multi-way spectral clustering framework for hybrid linear modeling. *Foundations of Computational Mathematics*, 9(5):517–558, 2009.

G. Chen and M. Maggioni. Multiscale geometric wavelets for the analysis of point clouds. *To appear in Proc. CISS 2010*, 2010.

G. Chen and M. Maggioni. Multiscale geometric and spectral analysis of plane arrangements. In *Proc. CVPR*, 2011.

G. Chen, A.V. Little, and M. Maggioni. Multi-resolution geometric analysis for data in high dimensions. *Proc. FFT 2011*, 2011a.

G. Chen, A.V. Little, M. Maggioni, and L. Rosasco. *Wavelets and Multiscale Analysis: Theory and Applications*. Springer Verlag, 2011b.

G. Chen, M. Iwen, Sang Chin, and M. Maggioni. A fast multiscale framework for data in high-dimensions: Measure estimation, anomaly detection, and compressive measurements. In *Visual Communications and Image Processing (VCIP), 2012 IEEE*, pages 1–6, 2012.

S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

P. Ciaccia, M. Patella, F. Rabitti, and P. Zezula. Indexing metric spaces with M-Tree. In *SEBD*, volume 97, pages 67–86, 1997.

R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *PNAS*, 102(21):7426–7431, 2005a.

R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Multiscale methods. *PNAS*, 102(21):7432–7438, 2005b.

R.R. Coifman and M. Maggioni. Diffusion wavelets. *Appl. Comp. Harm. Anal.*, 21(1):53–94, July 2006.

R.R. Coifman, S. Lafon, M. Maggioni, Y. Keller, A.D. Szlam, F.J. Warner, and S.W. Zucker. Geometries of sensor outputs, inference, and information processing. In *Defense and Security Symposium*. SPIE, May 2006.

G. David and S. Semmes. *Analysis of and on uniformly rectifiable sets*, volume 38 of *Mathematical Surveys and Monographs*. American Mathematical Society, Providence, RI, 1993. ISBN 0-8218-1537-7.

C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

D. Donoho. Compressed sensing. *IEEE Tran. on Information Theory*, 52(4):1289–1306, April 2006.

D. L. Donoho. Wedgelets: Nearly minimax estimation of edges. *Annals of Statistics*, 27(3): 859–897, 1999.

D. L. Donoho and C. Grimes. When does isomap recover the natural parameterization of families of articulated images? Technical report, 2002.

D. L. Donoho and C. Grimes. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. *PNAS*, 100(10):5591–5596, 2003.

A. Eftekhari and M. B. Wakin. New analysis of manifold embeddings and signal recovery from compressive measurements. *arXiv preprint arXiv:1306.4748*, 2013.

E. Elhamifar and R. Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2790–2797. IEEE, 2009.

H. Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93 (3):418–491, 1959.

C. Fefferman, S. Mitter, and H. Narayanan. Testing the manifold hypothesis. *Journ. A.M.S.* URL `http://arxiv.org/abs/1310.0425`. conf. version appeared in NIPS, 2010, pages 1786–1794.

M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

C. R. Genovese, M. Perone-Pacifico, I. Verdinelli, and L. Wasserman. Minimax manifold estimation. *J. Mach. Learn. Res.*, 13(1):1263–1291, May 2012a. ISSN 1532-4435.

C. R. Genovese, M. Perone-Pacifico, I. Verdinelli, and L. Wasserman. Manifold estimation and singular deconvolution under Hausdorff loss. *The Annals of Statistics*, 40(2):941–963, 2012b.

A. Gray. *Tubes*, volume 221 of *Progress in Mathematics*. Birkhäuser Verlag, Basel, second edition, 2004. ISBN 3-7643-6907-8. doi: 10.1007/978-3-0348-7966-8. With a preface by Vicente Miquel.

R. Gribonval, R. Jenatton, F. Bach, M. Kleinsteuber, and M. Seibert. Sample complexity of dictionary learning and other matrix factorizations. *arXiv:1312.3790*, 2013.

J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman. Clustering appearances of objects under varying illumination conditions. In *CVPR 2003 Proceedings.*, volume 1, pages I–11. IEEE, 2003.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(4):17–441,498–520, 1933.

H. Hotelling. Relations between two sets of variates. *Biometrika*, 27:321–77, 1936.

M.A. Iwen and M. Maggioni. Approximation of points on low-dimensional manifolds via random linear projections. *Inference & Information*, 2(1):1–31, 2013.

P. W. Jones. Rectifiable sets and the traveling salesman problem. *Inventiones Mathematicae*, 102(1):1–15, 1990.

P.W. Jones, M. Maggioni, and R. Schul. Manifold parametrizations by eigenfunctions of the Laplacian and heat kernels. *Proc. Nat. Acad. Sci.*, 105(6):1803–1808, Feb. 2008.

P.W. Jones, M. Maggioni, and R. Schul. Universal local manifold parametrizations via heat kernels and eigenfunctions of the Laplacian. *Ann. Acad. Scient. Fen.*, 35:1–44, January 2010.

D. R. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 741–750. ACM, 2002.

V. I. Koltchinskii. Empirical geometry of multivariate data: a deconvolution approach. *Annals of Statistics*, pages 591–629, 2000.

K. Kreutz-Delgado, J. F. Murray, B. D. Rao, Kjersti Engan, T.-W. Lee, and T. J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Comput.*, 15(2):349–396, February 2003.

E. Levina and P. J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems*, pages 777–784, 2004.

M.S. Lewicki, T.J. Sejnowski, and H. Hughes. Learning overcomplete representations. *Neural Computation*, 12:337–365, 1998.

A. V. Little, M. Maggioni, and L. Rosasco. Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature. Technical report, MIT, September 2012. URL `http://dspace.mit.edu/handle/1721.1/72597`.

A.V. Little, Y.-M. Jung, and M. Maggioni. Multiscale estimation of intrinsic dimensionality of data sets. In *Proceedings of AAAI*, 2009.

G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 663–670, 2010.

Y. Ma, H. Derksen, W. Hong, and J. Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(9):1546–1562, 2007.

Y. Ma, A. Y. Yang, H. Derksen, and R. Fossum. Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM Review*, 50(3):413–458, 2008.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journ. Mach. Learn. Res.*, 11:19–60, 2010.

A. Maurer and M. Pontil. K-dimensional coding schemes in Hilbert Spaces. *IEEE Transactions on Information Theory*, 56(11):5839–5846, 2010.

S. Minsker. On some extensions of Bernstein's inequality for self-adjoint operators. *arXiv preprint arXiv:1112.5448*, 2013.

P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete and Computational Geometry*, 39:419–441, 2008.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, (37), 1997.

A.V. Oppenheim and R.W. Schafer. *Digital Signal Processing*. Prentice-Hall, 1975.

K. Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

G. Peyré. Sparse modeling of textures. *Journal of Mathematical Imaging and Vision*, 34 (1):17–31, 2009.

M. Protter and M. Elad. Sparse and redundant representations and motion-estimation-free algorithm for video denoising, 2007.

I. U. Rahman, I. Drori, V. C. Stodden, D. L. Donoho, and P. Schröder. Multiscale representations for manifold-valued data. *Multiscale Modeling & Simulation*, 4(4):1201–1232, 2005.

S. T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

Y. Sugaya and K. Kanatani. Multi-stage unsupervised learning for multi-body motion segmentation. *IEICE Transactions on Information and Systems*, 87(7):1935–1942, 2004.

J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

D. Vainsencher, S. Mannor, and A. M. Bruckstein. The sample complexity of dictionary learning. *J. Mach. Learn. Res.*, 12:3259–3281, November 2011.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer Series in Statistics. Springer-Verlag, New York, 1996. With applications to statistics.

R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(12):1945–1959, 2005.

M. B. Wakin, D. L. Donoho, H. Choi, and R. G. Baraniuk. The multiscale structure of non-differentiable image manifolds. In *SPIE Wavelets XI*, pages 59141B–59141B. International Society for Optics and Photonics, 2005.

J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Computer Vision–ECCV 2006*, pages 94–106. Springer, 2006.

P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete algorithms*, pages 311–321. Society for Industrial and Applied Mathematics, 1993.

K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. In *Advances in Neural Information Processing Systems*, pages 2223–2231, 2009.

T. Zhang, A. Szlam, Y. Wang, and G. Lerman. Randomized hybrid linear modeling by local best-fit flats. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1927–1934. IEEE, 2010.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26:313–338, 2002.

L. Zwald and G. Blanchard. On the convergence of eigenspaces in kernel principal component analysis. In *Advances in Neural Information Processing Systems 18*, pages 1649–1656. MIT Press, Cambridge, MA, 2006.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 7/13; Revised 12/14; Published 3/16

Consistent Algorithms for Clustering Time Series

Azadeh Khaleghi a.khaleghi@lancaster.co.uk
Department of Mathematics & Statistics
Lancaster University, Lancaster, LA1 4YF, UK

Daniil Ryabko daniil@ryabko.net
INRIA Lille
40, avenue de Halley
59650 Villeneuve d'Ascq, France

Jérémie Mary Jeremie.Mary@inria.fr
Philippe Preux Philippe.Preux@inria.fr
Université de Lille/CRIStAL (UMR CNRS)
40, avenue de Halley
59650 Villeneuve d'Ascq, France

Editor: Gábor Lugosi

Abstract

The problem of clustering is considered for the case where every point is a time series. The time series are either given in one batch (offline setting), or they are allowed to grow with time and new time series can be added along the way (online setting). We propose a natural notion of consistency for this problem, and show that there are simple, computationally efficient algorithms that are asymptotically consistent under extremely weak assumptions on the distributions that generate the data. The notion of consistency is as follows. A clustering algorithm is called consistent if it places two time series into the same cluster if and only if the distribution that generates them is the same. In the considered framework the time series are allowed to be highly dependent, and the dependence can have arbitrary form. If the number of clusters is known, the only assumption we make is that the (marginal) distribution of each time series is stationary ergodic. No parametric, memory or mixing assumptions are made. When the number of clusters is unknown, stronger assumptions are provably necessary, but it is still possible to devise nonparametric algorithms that are consistent under very general conditions. The theoretical findings of this work are illustrated with experiments on both synthetic and real data.

Keywords: clustering, time series, ergodicity, unsupervised learning

1. Introduction

Clustering is a widely studied problem in machine learning, and is key to many applications across different fields of science. The goal is to partition a given data set into a set of non-overlapping *clusters* in some natural way, thus hopefully revealing an underlying structure in the data. In particular, this problem for time series data is motivated by many research problems from a variety of disciplines, such as marketing and finance, biological and medical research, video/audio analysis, etc., with the common feature that the data are abundant while little is known about the nature of the processes that generate them.

The intuitively appealing problem of clustering is notoriously difficult to formalise. An intrinsic part of the problem is a similarity measure: the points in each cluster are supposed to be close to each other in some sense. While it is clear that the problem of finding the appropriate similarity measure is inseparable from the problem of achieving the clustering objectives, in the available formulations of the problem it is usually assumed that the similarity measure is given: it is either some fixed metric, or just a matrix of similarities between the points. Even with this simplification, it is still unclear how to define good clustering. Thus, Kleinberg (2002) presents a set of fairly natural properties that a good clustering function should have, and further demonstrate that there is no clustering function that satisfies these properties. A common approach is therefore to fix not only the similarity measure, but also some specific objective function — typically, along with the number of clusters — and to construct algorithms that maximise this objective. However, even this approach has some fundamental difficulties, albeit of a different, this time computational, nature: already in the case where the number of clusters is known, and the distance between the points is set to be the Euclidean distance, optimising some fairly natural objectives (such as k-means) turns out to be NP hard (Mahajan et al., 2009).

In this paper we consider a subset of the clustering problem, namely, clustering time series. That is, we consider the case where each data point is a sample drawn from some (unknown) time-series distribution. At first glance this does not appear to be a simplification (indeed, any data point can be considered as a time series of length 1). However, note that time series present a different dimension of asymptotic: with respect to their length, rather than with respect to the total number of points to be clustered. "Learning" along this dimension turns out to be easy to define, and allows for the construction of consistent algorithms under most general assumptions. Specifically, the assumption that each time series is generated by a stationary ergodic distribution is already sufficient to estimate any finite-dimensional characteristic of the distribution with arbitrary precision, provided the series is long enough. Thus, in contrast to the general clustering setup, in the time-series case it is possible to "learn" the distribution that generates each given data point. No assumptions on the dependence between time series are necessary for this. The assumption that a given time series is stationary ergodic is one of the most general assumptions used in statistics; in particular, it allows for arbitrary long-range serial dependence, and subsumes most of the nonparametric as well as modelling assumptions used in the literature on clustering time series, such as i.i.d., (hidden) Markov, or mixing time series.

This allows us to define the following clustering objective: *group a pair of time series into the same cluster if and only if the distribution that generates them is the same.*

Note that this intuitive objective is impossible to achieve outside the time-series framework. Even in the simplest case where the underlying distributions are Gaussian and the number of clusters is known, there is always a nontrivial likelihood for each point to be generated by any of the distributions. Thus, the best one can do is to estimate the parameters of the distributions, rather than to actually cluster the data points. The situation becomes hopeless in the nonparametric setting. In contrast, the fully nonparametric case is tractable for time-series data, both in offline and online scenarios, as explained in the next section.

1.1 Problem Setup

We consider two variants of the clustering problem in this setting: offline (batch) and online, defined as follows.

In the *offline (batch)* setting a finite number N of sequences $\mathbf{x}_1 = (X_1^1, \dots, X_{n_1}^1)$, $\dots, \mathbf{x}_N = (X_1^N, \dots, X_{n_N}^N)$ is given. Each sequence is generated by one of κ different *unknown* time-series distributions. The target clustering is the partitioning of $\mathbf{x}_1, \dots, \mathbf{x}_N$ into κ clusters, putting together those and only those sequences that were generated by the same time-series distribution. We call a batch clustering algorithm *asymptotically consistent* if, with probability 1, it stabilises on the target clustering in the limit as the lengths $n_1, \dots, n_N$ of each of the N samples tend to infinity. The number κ of clusters may be either known or unknown. Note that the asymptotic is not with respect to the number of sequences, but with respect to the individual sequence lengths.

In the *online* setting we have a growing body of sequences of data. The number of sequences as well as the sequences themselves grow with time. The manner of this evolution can be arbitrary; we only require that the length of each individual sequence tend to infinity. Similarly to the batch setting, the joint distribution generating the data is unknown. At time step 1 initial segments of some of the first sequences are available to the learner. At each subsequent time step, new data are revealed, either as an extension of a previously observed sequence, or as a new sequence. Thus, at each time step t a total of $N(t)$ sequences $\mathbf{x}_1, \dots, \mathbf{x}_{N(t)}$ are to be clustered, where each sequence $\mathbf{x}_i$ is of length $n_i(t) \in \mathbb{N}$ for $i = 1..N(t)$. The total number of observed sequences $N(t)$ as well as the individual sequence lengths $n_i(t)$ grow with time. In the online setting, a clustering algorithm is called asymptotically consistent, if almost surely for each fixed batch of sequences $\mathbf{x}_1, \dots, \mathbf{x}_N$, the clustering restricted to this sequences coincides with the target clustering from some time on.

At first glance it may seem that one can use the offline algorithm in the online setting by simply applying it to the entire data observed at every time step. However, this naive approach does not result in a consistent algorithm. The main challenge in the online setting can be identified with what we regard as "bad" sequences: sequences for which sufficient information has not yet been collected, and thus cannot be distinguished based on the process distributions that generate them. In this setting, using a batch algorithm at every time step results in not only mis-clustering such "bad" sequences, but also in clustering incorrectly those for which sufficient data are already available. That is, such "bad" sequences can render the entire batch clustering useless, leading the algorithm to incorrectly cluster even the "good" sequences. Since new sequences may arrive in an uncontrollable (even data-dependent, adversarial) fashion, any batch algorithm will fail in this scenario.

1.2 Results

The first result of this work is a formal definition of the problem of time-series clustering which is rather intuitive, and at the same time allows for the construction of efficient algorithms that are provably consistent under the most general assumptions. The second result is the construction of such algorithms.

More specifically, we propose clustering algorithms that, as we further show, are strongly asymptotically consistent, provided that the number κ of clusters is known, and the process

distribution generating each sequence is stationary ergodic. No restrictions are placed on the dependence between the sequences: this dependence may be arbitrary (and can be thought of as adversarial). This consistency result is established in each of the two settings (offline and online) introduced above.

As follows from the theoretical impossibility results of Ryabko (2010c) that are discussed further in Section 1.4, under the only assumption that the distributions generating the sequences are stationary ergodic, it is impossible to find the correct number of clusters κ, that is, the total number of different time-series distributions that generate the data. Moreover, non-asymptotic results (finite-time bounds on the probability of error) are also provably impossible to obtain in this setting, since this is already the case for the problem of estimating the probability of any finite-time event (Shields, 1996).

Finding the number κ of clusters, as well as obtaining non-asymptotic performance guarantees, is possible under additional conditions on the distributions. In particular, we show that if κ is unknown, it is possible to construct consistent algorithms provided that the distributions are mixing and bounds on the mixing rates are available.

However, the main focus of this paper remains on the general framework where no additional assumptions on the unknown process distributions are made (other than that they are stationary ergodic). As such, the main theoretical and experimental results concern the case of known κ.

Finally, we show that our methods can be implemented efficiently: they are at most quadratic in each of their arguments, and are linear (up to log terms) in some formulations. To test the empirical performance of our algorithms we evaluated them on both synthetic and real data. To reflect the generality of the suggested framework in the experimental setup, we had our synthetic data generated by stationary ergodic process distributions that do not belong to any "simpler" class of distributions, and in particular cannot be modelled as hidden Markov processes with countable sets of states. In the batch setting, the error rates of both methods go to zero with sequence length. In the online setting with new samples arriving at every time step, the error rate of the offline algorithm remains consistently high, whereas that of the online algorithm converges to zero. This demonstrates that, unlike the offline algorithm, the online algorithm is robust to "bad" sequences. To demonstrate the applicability of our work to real-world scenarios, we chose the problem of clustering motion-capture sequences of human locomotion. This application area has also been studied in the works (Li and Prakash, 2011) and (Jebara et al., 2007) that (to the best of our knowledge) constitute the state-of-the-art performance[1] on the data sets they consider, and against which we compare the performance of our methods. We obtained consistently better performance on the data sets involving motion that can be considered ergodic (walking, running), and competitive performance on those involving non-ergodic motions (single jumps).

1. After this paper had been accepted for publication and entered the production stage, the work (Tschannen and Bolcskei, 2015) appeared, which uses the data set of Li and Prakash (2011) and reports a better performance on it as compared to both our algorithm and that of Li and Prakash (2011).

1.3 Methodology and Algorithms

A crucial point of any clustering method is the similarity measure. Since the objective in our formulation is to cluster time series based on the distributions that generate them, the similarity measure must reflect the difference between the underlying distributions. As we aim to make as few assumptions as possible on the distributions that generate the data, and since we make no assumptions on the nature of differences between the distributions, the distance should take into account all possible differences between time-series distributions. Moreover, it should be possible to construct estimates of this distance that are consistent for arbitrary stationary ergodic distributions.

It turns out that a suitable distance for this purpose is the so-called distributional distance. The distributional distance $d(\rho_1, \rho_2)$ between a pair of process distributions ρ_1, ρ_2 is defined (Gray, 1988) as $\sum_{j\in\mathbb{N}} w_j|\rho_1(B_j) - \rho_2(B_j)|$ where, w_j are positive summable real weights, e.g. $w_j = 1/j(j+1)$, and B_j range over a countable field that generates the σ-algebra of the underlying probability space. For example, consider finite-alphabet processes with the binary alphabet $\mathcal{X} = \{0, 1\}$. In this case $B_j,\ j \in \mathbb{N}$ would range over the set $\mathcal{X}^* = \cup_{m\in\mathbb{N}}\mathcal{X}^m$; that is, over all tuples $0, 1, 00, 01, 10, 11, 000, 001, \ldots$; therefore, the distributional distance in this case is the weighted sum of the differences of the probability values (calculated with respect to ρ_1 and ρ_2) of all possible tuples. In this case, the distributional distance metrises the topology of weak convergence. In this work we consider real-valued processes so that the sets B_j range over a suitable sequence of intervals, all pairs of such intervals, triples, and so on (see the formal definitions in Section 2). Although this distance involves infinite summations, we show that its empirical approximations can be easily calculated. Asymptotically consistent estimates of this distance can be obtained by replacing unknown probabilities with the corresponding frequencies (Ryabko and Ryabko, 2010); these estimators have proved useful in various statistical problems concerning ergodic processes (Ryabko and Ryabko, 2010; Ryabko, 2012; Khaleghi and Ryabko, 2012).

Armed with an estimator of the distributional distance, it is relatively easy to construct a consistent clustering algorithm for the batch setting. In particular, we show that the following algorithm is asymptotically consistent. First, a set of κ cluster centres are chosen by κ farthest point initialisation (using an empirical estimate of the distributional distance). This means that the first sequence is assigned as the first cluster centre. Iterating over $2..\kappa$, at every iteration a sequence is sought which has the largest minimum distance from the already chosen cluster centres. Next, the remaining sequences are assigned to the closest clusters.

The online algorithm is based on a *weighted combination* of several clusterings, each obtained by running the offline procedure on different portions of data. The partitions are combined with weights that depend on the batch size and on an appropriate performance measure for each individual partition. The performance measure of each clustering is the minimum inter-cluster distance.

1.4 Related Work

Some probabilistic formulations of the time-series clustering problem can be considered related to ours. Perhaps the closest is in terms of mixture models (Bach and Jordan, 2004; Biernacki et al., 2000; Kumar et al., 2002; Smyth, 1997; Zhong and Ghosh, 2003):

it is assumed that the data are generated by a mixture of κ different distributions that have a particular known form (such as Gaussian, Hidden Markov models, or graphical models). Thus, each of the N samples is independently generated according to one of these κ distributions (with some fixed probability). Since the model of the data is specified quite well, one can use likelihood-based distances (and then, for example, the k-means algorithm), or Bayesian inference, to cluster the data. Another typical objective is to estimate the parameters of the distributions in the mixture (e.g., Gaussians), rather than actually clustering the data points. Clearly, the main difference between this setting and ours is in that we do not assume any known model for the data; we do not even require independence between the samples.

Taking a different perspective, the problem of clustering in our formulations generalises two classical problems in mathematical statistics, namely, homogeneity testing (or the two-sample problem) and process classification (or the three-sample problem). In the *two-sample problem*, given two sequences $\mathbf{x}_1 = (X_1^1, \ldots, X_{n_1}^1)$ and $\mathbf{x}_2 = (X_1^2, \ldots, X_{n_2}^2)$ it is required to test whether they are generated by the same or by different process distributions. This corresponds to the special case of clustering only $N = 2$ data points where the number κ of clusters is unknown: it can be either 1 or 2. In the *three-sample problem,* three sequences $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are given, and it is known that $\mathbf{x}_1$ and $\mathbf{x}_2$ are generated by different distributions, while $\mathbf{x}_3$ is generated either by the same distribution as $\mathbf{x}_1$ or by the same distribution as $\mathbf{x}_2$. It is required to find which one is the case. This can be seen as clustering $N = 3$ data points, with the number of clusters known: $\kappa = 2$. The classical approach is, of course, to consider Gaussian i.i.d. samples, but general nonparametric solutions exist not only for i.i.d. data (Lehmann, 1986), but also for Markov chains (Gutman, 1989), as well as under certain conditions on the mixing rates of the processes. Observe that the two-sample problem is more difficult to solve than the three-sample problem, since the number κ of clusters is unknown in the former while it is given in the latter. Indeed, as shown by Ryabko (2010c), in general for stationary ergodic (binary-valued) processes there is no solution for the two-sample problem, even in the weakest asymptotic sense. A solution to the three-sample problem for (real-valued) stationary ergodic processes was given by (Ryabko and Ryabko, 2010); it is based on estimating the distributional distance.

More generally, the area of nonparametric statistical analysis of stationary ergodic time series to which the main results of this paper belong, is full of both positive and negative results, some of which are related to the problem of clustering in our formulation. Among these we can mention change point problems (Carlstein and Lele, 1994; Khaleghi and Ryabko, 2012, 2014) hypothesis testing (Ryabko, 2012, 2014; Morvai and Weiss, 2005) and prediction (Ryabko, 1988; Morvai and Weiss, 2012).

1.5 Organization

The remainder of the paper is organised as follows. We start by introducing notation and definitions in Section 2. In Section 3 we define the considered clustering protocol. Our main theoretical results are given in Section 4, where we present our methods, prove their consistency, and discuss some extensions (Section 4.4). Section 5 is devoted to computational considerations. In Section 6 we provide some experimental evaluations on both synthetic and real data. Finally, in Section 7 we provide some concluding remarks and open questions.

2. Preliminaries

Let $\mathcal{X}$ be a measurable space (the domain); in this work we let $\mathcal{X} = \mathbb{R}$ but extensions to more general spaces are straightforward. For a sequence $X_1, \dots, X_n$ we use the abbreviation $X_{1..n}$. Consider the Borel σ-algebra $\mathcal{B}$ on $\mathcal{X}^\infty$ generated by the cylinders $\{B \times \mathcal{X}^\infty : B \in B^{m,l}, m, l \in \mathbb{N}\}$ where, the sets $B^{m,l}, m, l \in \mathbb{N}$ are obtained via the partitioning of $\mathcal{X}^m$ into cubes of dimension m and volume 2^{-ml} (starting at the origin). Let also $B^m := \cup_{l \in \mathbb{N}} B^{m,l}$. We may choose any other means to generate the Borel σ-algebra $\mathcal{B}$ on $\mathcal{X}$, but we need to fix a specific choice for computational reasons. Process distributions are probability measures on the space $(\mathcal{X}^\infty, \mathcal{B})$. Similarly, one can define distributions over the space $((\mathcal{X}^\infty)^\infty, \mathcal{B}_2)$ of infinite matrices with the Borel σ-algebra $\mathcal{B}_2$ generated by the cylinders $\{(\mathcal{X}^\infty)^k \times (B \times \mathcal{X}^\infty) \times (\mathcal{X}^\infty)^\infty : \ B \in B^{m,l}, k, m, l \in \{0\} \cup \mathbb{N}\}$. For $\mathbf{x} = X_{1..n} \in \mathcal{X}^n$ and $B \in B^m$ let $\nu(\mathbf{x}, B)$ denote the *frequency* with which $\mathbf{x}$ falls in B, i.e.

$$\nu(\mathbf{x}, B) := \frac{\mathbb{I}\{n \geq m\}}{n - m + 1} \sum_{i=1}^{n-m+1} \mathbb{I}\{X_{i..i+m-1} \in B\} \tag{1}$$

A process ρ is *stationary* if for any $i, j \in 1..n$ and $B \in \mathcal{B}$, we have

$$\rho(X_{1..j} \in B) = \rho(X_{i..i+j-1} \in B).$$

A stationary process ρ is called *ergodic* if for all $B \in \mathcal{B}$ with probability 1 we have

$$\lim_{n \to \infty} \nu(X_{1..n}, B) = \rho(B).$$

By virtue of the ergodic theorem (e.g., Billingsley, 1961), this definition can be shown to be equivalent to the standard definition of stationary ergodic processes (every shift-invariant set has measure 0 or 1; see, e.g., Gray, 1988).

Definition 1 (Distributional Distance) *The distributional distance between a pair of process distributions ρ_1, ρ_2 is defined as follows (Gray, 1988):*

$$d(\rho_1, \rho_2) = \sum_{m,l=1}^{\infty} w_m w_l \sum_{B \in B^{m,l}} |\rho_1(B) - \rho_2(B)|,$$

where we set $w_j := 1/j(j+1)$, but any summable sequence of positive weights may be used.

In words, this involves partitioning the sets $\mathcal{X}^m$, $m \in \mathbb{N}$ into cubes of decreasing volume (indexed by l) and then taking a sum over the absolute differences in probabilities of all of the cubes in these partitions. The differences in probabilities are weighted: smaller weights are given to larger m and finer partitions.

Remark 2 The distributional distance is more generally defined as

$$d'(\rho_1, \rho_2) := \sum_{i \in \mathbb{N}} w_i |\rho_1(A_i) - \rho_2(A_i)|$$

where A_i ranges over a countable field that generates the σ-algebra of the underlying probability space (Gray, 1988). Definition 1 above is more suitable for implementing and testing

algorithms, while the consistency results of this paper hold for either. Note also that the choice of sets B_i may result in a different topology on the space of time-series distributions. The particular choice we made results in the usual topology of weak convergence (Billingsley, 1999).

We use the following empirical estimates of the distributional distance.

Definition 3 (Empirical estimates of d) *Define the empirical estimate of the distributional distance between a sequence $\mathbf{x} = X_{1..n} \in \mathcal{X}^n, n \in \mathbb{N}$ and a process distribution ρ as*

$$\hat{d}(\mathbf{x}, \rho) := \sum_{m=1}^{m_n} \sum_{l=1}^{l_n} w_m w_l \sum_{B \in B^{m,l}} |\nu(\mathbf{x}, B) - \rho(B)|, \tag{2}$$

and that between a pair of sequences $\mathbf{x}_i \in \mathcal{X}^{n_i}$ $n_i \in \mathbb{N}$, $i = 1, 2$ as

$$\hat{d}(\mathbf{x}_1, \mathbf{x}_2) := \sum_{m=1}^{m_n} \sum_{l=1}^{l_n} w_m w_l \sum_{B \in B^{m,l}} |\nu(\mathbf{x}_1, B) - \nu(\mathbf{x}_2, B)|, \tag{3}$$

where m_n and l_n are any sequences that go to infinity with n, and $(w_j)_{j \in \mathbb{N}}$ are as in Definition 1.

Remark 4 (constants m_n, l_n) The sequences of constants $m_n, l_n (n \in \mathbb{N})$ in the definition of $\hat{d}$ are intended to introduce some computational flexibility in the estimate: while already the choice $m_n, l_n \equiv \infty$ is computationally feasible, other choices give better computational complexity without sacrificing the quality of the estimate; see Section 5. For the asymptotic consistency results this choice is irrelevant. Thus, in the proofs we tacitly assume $m_n, l_n \equiv \infty$; extension to the general case is straightforward.

Lemma 5 ($\hat{d}$ is asymptotically consistent: Ryabko and Ryabko, 2010) *For every pair of sequences $\mathbf{x}_1 \in \mathcal{X}^{n_1}$ and $\mathbf{x}_2 \in \mathcal{X}^{n_2}$ with joint distribution ρ whose marginals $\rho_i, i = 1, 2$ are stationary ergodic we have*

$$\lim_{n_1, n_2 \to \infty} \hat{d}(\mathbf{x}_1, \mathbf{x}_2) = d(\rho_1, \rho_2), \ \rho - a.s., \ \text{and} \tag{4}$$

$$\lim_{n_i \to \infty} \hat{d}(\mathbf{x}_i, \rho_j) = d(\rho_i, \rho_j), \ i, j \in 1, 2, \ \rho - a.s. \tag{5}$$

The proof of this lemma can be found in the work (Ryabko and Ryabko, 2010); since it is important for further development but rather short and simple, we reproduce it here.

Proof The idea of the proof is simple: for each set $B \in \mathcal{B}$, the frequency with which the sample $\mathbf{x}_i$, $i = 1, 2$ falls into B converges to the probability $\rho_i(B)$, $i = 1, 2$. When the sample sizes grow, there will be more and more sets $B \in \mathcal{B}$ whose frequencies have already converged to the corresponding probabilities, so that the cumulative weight of those sets whose frequencies have not converged yet will tend to 0.

Fix $\varepsilon > 0$. We can find an index J such that

$$\sum_{m,l=J}^{\infty} w_m w_l \leq \varepsilon/3.$$

Moreover, for each $m, l \in 1..J$ we can find a finite subset $S^{m,l}$ of $B^{m,l}$ such that

$$\rho_i(S^{m,l}) \geq 1 - \frac{\varepsilon}{6Jw_m w_l}.$$

There exists some N (which depends on the realization $X_1^i, \ldots, X_{n_i}^i,\ i = 1, 2$) such that for all $n_i \geq N,\ i = 1, 2$ we have,

$$\sup_{B \in S^{m,l}} |\nu(\mathbf{x}_i, B) - \rho_i(B)| \leq \frac{\varepsilon \rho_i(B)}{6Jw_m w_l},\ i = 1, 2. \tag{6}$$

For all $n_i \geq N,\ i = 1, 2$ we have

$$\begin{aligned}
|\hat{d}(\mathbf{x}_1, \mathbf{x}_2) - d(\rho_1, \rho_2)| &= \left| \sum_{m,l=1}^{\infty} w_m w_l \sum_{B \in B^{m,l}} \big(|\nu(\mathbf{x}_1, B) - \nu(\mathbf{x}_2, B)| - |\rho_1(B) - \rho_2(B)| \big) \right| \\
&\leq \sum_{m,l=1}^{\infty} w_m w_l \sum_{B \in B^{m,l}} |\nu(\mathbf{x}_1, B) - \rho_1(B)| + |\nu(\mathbf{x}_2, B) - \rho_2(B)| \\
&\leq \sum_{m,l=1}^{J} w_m w_l \sum_{B \in S^{m,l}} \big(|\nu(\mathbf{x}_1, B) - \rho_1(B)| + |\nu(\mathbf{x}_2, B) - \rho_2(B)| \big) + 2\varepsilon/3 \\
&\leq \sum_{m,l=1}^{J} w_m w_l \sum_{B \in S^{m,l}} \frac{\rho_1(B)\varepsilon}{6Jw_m w_l} + \frac{\rho_2(B)\varepsilon}{6Jw_m w_l} + 2\varepsilon/3 \leq \varepsilon,
\end{aligned}$$

which proves (4). The statement (5) can be proven analogously. ■

Remark 6 The triangle inequality holds for the distributional distance $d(\cdot, \cdot)$ and its empirical estimates $\hat{d}(\cdot, \cdot)$ so that for all distributions $\rho_i,\ i = 1..3$ and all sequences $\mathbf{x}_i \in \mathcal{X}^{n_i}\ n_i \in \mathbb{N},\ i = 1..3$ we have

$$\begin{aligned}
d(\rho_1, \rho_2) &\leq d(\rho_1, \rho_3) + d(\rho_2, \rho_3), \\
\hat{d}(\mathbf{x}_1, \mathbf{x}_2) &\leq \hat{d}(\mathbf{x}_1, \mathbf{x}_3) + \hat{d}(\mathbf{x}_2, \mathbf{x}_3), \\
\hat{d}(\mathbf{x}_1, \rho_1) &\leq \hat{d}(\mathbf{x}_1, \rho_2) + d(\rho_1, \rho_2).
\end{aligned}$$

3. Clustering Settings: Offline and Online

We start with a common general probabilistic setup for both settings (offline and online), which we use to formulate each of the two settings separately.

Consider the matrix $\mathbf{X} \in (\mathcal{X}^\infty)^\infty$ of random variables

$$\mathbf{X} := \begin{bmatrix} X_1^1 & X_2^1 & X_3^1 & \ldots \\ X_1^2 & X_2^2 & \ldots & \ldots \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix} \in (\mathcal{X}^\infty)^\infty \tag{7}$$

generated by some probability distribution P on $((\mathcal{X}^\infty)^\infty, \mathcal{B}_2)$. Assume that the marginal distribution of P on each row of $\mathbf{X}$ is one of κ *unknown stationary ergodic* process distributions $\rho_1, \rho_2, \ldots, \rho_\kappa$. Thus, the matrix $\mathbf{X}$ corresponds to infinitely many one-way infinite sequences, each of which is generated by a stationary ergodic distribution. Aside from this assumption, we do not make any further assumptions on the distribution P that generates $\mathbf{X}$. This means that the rows of $\mathbf{X}$ (corresponding to different time-series samples) are allowed to be dependent, and the dependence can be arbitrary; one can even think of the dependence between samples as *adversarial*. For notational convenience we assume that the distributions $\rho_k, k = 1..\kappa$ are ordered based on the order of appearance of their first rows (samples) in $\mathbf{X}$.

Among the various ways the rows of $\mathbf{X}$ may be partitioned into κ disjoint subsets, the most natural partitioning in this formulation is to group together those and only those rows for which the marginal distribution is the same. More precisely, we define the ground-truth partitioning of $\mathbf{X}$ as follows.

Definition 7 (Ground-truth $\mathcal{G}$) *Let*

$$\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_\kappa\}$$

be a partitioning of $\mathbb{N}$ into κ disjoint subsets $\mathcal{G}_k,\ k = 1..\kappa$, such that the marginal distribution of $\mathbf{x}_i,\ i \in \mathbb{N}$ is ρ_k for some $k \in 1..\kappa$ if and only if $i \in \mathcal{G}_k$. We call $\mathcal{G}$ the ground-truth clustering. We also introduce the notation $\mathcal{G}|_N$ for the restriction of $\mathcal{G}$ to the first N sequences:

$$\mathcal{G}|_N := \{\mathcal{G}_k \cap \{1..N\} : k = 1..\kappa\}.$$

3.1 Offline Setting

The problem is formulated as follows. We are given a finite set $S := \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of N samples, for some fixed $N \in \mathbb{N}$. Each sample is generated by one of κ unknown stationary ergodic process distributions $\rho_1, \ldots, \rho_\kappa$. More specifically, the set S is obtained from $\mathbf{X}$ as follows. Some (arbitrary) lengths $n_i \in \mathbb{N},\ i \in 1..N$ are fixed, and $\mathbf{x}_i$ for each $i = 1..N$ is defined as $\mathbf{x}_i := X_{1..n_i}^i$.

A clustering function f takes a finite set $S := \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of samples and an optional parameter κ (the number of target clusters) to produce a partition $f(S, \kappa) := \{C_1, \ldots, C_\kappa\}$ of the index set $\{1..N\}$. The goal is to partition $\{1..N\}$ so as to recover the ground-truth clustering $\mathcal{G}|_N$. For consistency of notation, in the offline setting we identify $\mathcal{G}$ with $\mathcal{G}|_N$. We call a clustering algorithm asymptotically consistent if it achieves this goal for long enough sequences $\mathbf{x}_i,\ i = 1..N$ in S:

Definition 8 (Consistency: offline setting) *A clustering function f is consistent for a set of sequences S if $f(S, \kappa) = \mathcal{G}$. Moreover, denoting $n := \min\{n_1, \ldots, n_N\}$, f is called strongly asymptotically consistent in the offline sense if with probability 1 from some n on it is consistent on the set S: $P(\exists n' \forall n > n' f(S, \kappa) = \mathcal{G}) = 1$. It is weakly asymptotically consistent if $\lim_{n \to \infty} P(f(S, \kappa) = \mathcal{G}) = 1$.*

Note that the notion of consistency above is asymptotic with respect to the minimum sample length n, and not with respect to the number N of samples.

When considering the offline setting with a known number of clusters κ we assume that N is such that $\{1..N\} \cap \mathcal{G}_k \neq \varnothing$ for every $k = 1..\kappa$ (that is, all κ clusters are represented); otherwise we could just take a smaller κ.

3.2 Online Setting

The online problem is formulated as follows. Consider the infinite matrix $\mathbf{X}$ given by (7). At every time step $t \in \mathbb{N}$, a part $S(t)$ of $\mathbf{X}$ is observed corresponding to the first $N(t) \in \mathbb{N}$ rows of $\mathbf{X}$, each of length $n_i(t), i \in 1..N(t)$, i.e.

$$S(t) = \{\mathbf{x}_1^t, \cdots \mathbf{x}_{N(t)}^t\} \text{ where } \mathbf{x}_i^t := X_{1..n_i(t)}^i.$$

This setting is depicted in Figure 1. We assume that the number of samples, as well as the individual sample-lengths grow with time. That is, the length $n_i(t)$ of each sequence $\mathbf{x}_i$ is nondecreasing and grows to infinity (as a function of time t). The number of sequences $N(t)$ also grows to infinity. Aside from these assumptions, the functions $N(t)$ and $n_i(t)$ are completely arbitrary.

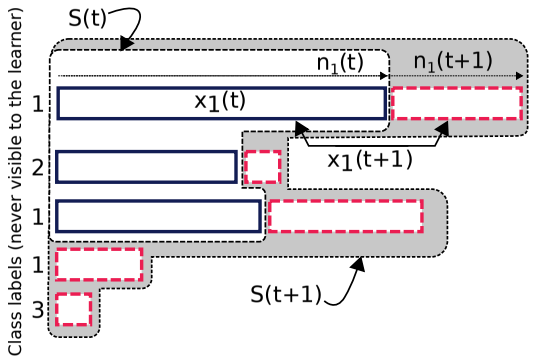


Figure 1: Online Protocol: solid rectangles correspond to sequences available at time t, dashed rectangles correspond to segments arrived at time $t+1$.

An online clustering function is strongly asymptotically consistent if, with probability 1, for each $N \in \mathbb{N}$ from some time on the first N sequences are clustered correctly (with respect to the ground-truth given by Definition 7).

Definition 9 (Consistency: online setting) *A clustering function is strongly (weakly) asymptotically consistent in the online sense, if for every $N \in \mathbb{N}$ the clustering $f(S(t), \kappa)|_N$ is strongly (weakly) asymptotically consistent in the offline sense, where $f(S(t), \kappa)|_N$ is the clustering $f(S(t), \kappa)$ restricted to the first N sequences:*

$$f(S(t), \kappa)|_N := \{f(S(t), \kappa) \cap \{1..N\}, k = 1..\kappa\}.$$

Remark 10 Note that even if the *eventual* number κ of different time-series distributions producing the sequences (that is, the number of clusters in the ground-truth clustering $\mathcal{G}$) is known, the number of observed distributions at each individual time step is unknown. That is, it is possible that at a given time t we have $\left|\mathcal{G}|_{N(t)}\right| < \kappa$.

Known and unknown κ. As mentioned in the introduction, in the general framework described above, consistent clustering (for both the offline and the online problems) with unknown number of clusters is impossible. This follows from the impossibility result of Ryabko (2010c) that states that when we have only two (binary-valued) samples generated (independently) by two stationary ergodic distributions, it is impossible to decide whether they have been generated by the same or by different distributions, even in the sense of weak asymptotic consistency. (This holds even if the distributions come from a smaller class: the set of all B-processes.) Therefore, if the number of clusters is unknown, we are bound to make stronger assumptions. Since our main interest in this paper is to develop consistent clustering algorithms under the general framework described above, for the most part of this paper we assume that the correct number κ of clusters is known. However, in Section 4.3 we also show that under certain mixing conditions on the process distributions that generate the data, it is possible to have consistent algorithms in the case of unknown κ as well.

4. Clustering Algorithms and their Consistency

In this section we present our clustering methods for both the offline and the online settings. The main results, presented in Sections 4.1 (offline) and 4.2 (online), concern the case where the number of clusters κ is known. In Section 4.3 we show that, given the mixing rates of the process distributions that generate the data, it is possible to find the correct number of clusters κ and thus obtain consistent algorithms for the case of unknown κ as well. Finally, Section 4.4 considers some extensions of the proposed settings.

4.1 Offline Setting

Given that we have asymptotically consistent estimates $\hat{d}$ of the distributional distance d, it is relatively simple to construct asymptotically consistent clustering algorithms for the offline setting. This follows from the fact that, since $\hat{d}(\cdot,\cdot)$ converges to $d(\cdot,\cdot)$, for large enough sequence lengths n the points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ have the so-called strict separation property: the points within each target cluster are closer to each other than to points in any other cluster (on strict separation in clustering see, for example, Balcan et al., 2008). This makes many simple algorithms, such as single or average linkage, or the k-means algorithm with certain initialisations, provably consistent.

We present below one specific algorithm that we show to be asymptotically consistent in the general framework introduced. What makes this simple algorithm interesting is that it requires only κN distance calculations (that is, much less than is needed to calculate the distance between each two sequences). In short, Algorithm 1 initialises the clusters using farthest-point initialisation, and then assigns each remaining point to the nearest cluster. More precisely, the sample $\mathbf{x}_1$ is assigned as the first cluster centre. Then a sample is found that is farthest away from $\mathbf{x}_1$ in the empirical distributional distance $\hat{d}$ and is assigned as

the second cluster centre. For each $k = 2..\kappa$ the k^{th} cluster centre is sought as the sequence with the largest minimum distance from the already assigned cluster centres for $1..k-1$. (This initialisation was proposed for use with k-means clustering by Katsavounidis et al., 1994.) By the last iteration we have κ cluster centres. Next, the remaining samples are each assigned to the closest cluster.

Algorithm 1 Offline clustering

INPUT: sequences $S := \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$**, Number** κ **of clusters**
Initialize κ***-farthest points as cluster-centres:***
$c_1 \leftarrow 1$
$C_1 \leftarrow \{c_1\}$
for $k = 2..\kappa$ **do**
  $c_k \leftarrow \operatorname{argmax}_{i=1..N} \min_{j=1..k-1} \hat{d}(\mathbf{x}_i, \mathbf{x}_{c_j})$***, where ties are broken arbitrarily***
  $C_k \leftarrow \{c_k\}$
end for
Assign the remaining points to closest centres:
for $i = 1..N$ **do**
  $k \leftarrow \operatorname{argmin}_{j \in \bigcup_{k=1}^{\kappa} C_k} \hat{d}(\mathbf{x}_i, \mathbf{x}_j)$
  $C_k \leftarrow C_k \cup \{i\}$
end for
OUTPUT: clusters $C_1, C_2, \cdots, C_\kappa$

Theorem 11 *Algorithm 1 is strongly asymptotically consistent (in the offline sense of Definition 8), provided that the correct number κ of clusters is known, and the marginal distribution of each sequence $\mathbf{x}_i, i = 1..N$ is stationary ergodic.*

Proof To prove the consistency statement we use Lemma 5 to show that if the samples in S are long enough, the samples that are generated by the same process distribution are closer to each other than to the rest of the samples. Therefore, the samples chosen as cluster centres are each generated by a different process distribution, and since the algorithm assigns the rest of the samples to the closest clusters, the statement follows. More formally, let n denote the shortest sample length in S:

$$n_{\min} := \min_{i \in 1..N} n_i.$$

Denote by δ the minimum nonzero distance between the process distributions:

$$\delta := \min_{k \neq k' \in 1..\kappa} \hat{d}(\rho_k, \rho_{k'}).$$

Fix $\varepsilon \in (0, \delta/4)$. Since there are a finite number N of samples, by Lemma 5 for all large enough $n_{\min}$ we have

$$\sup_{\substack{k \in 1..\kappa \\ i \in \mathcal{G}_k \cap \{1..N\}}} \hat{d}(\mathbf{x}_i, \rho_k) \leq \varepsilon. \tag{8}$$

where $\mathcal{G}_k$, $k = 1..\kappa$ denote the ground-truth partitions given by Definition 7. By (8) and applying the triangle inequality we obtain

$$\sup_{\substack{k \in 1..\kappa \\ i,j \in \mathcal{G}_k \cap \{1..N\}}} \hat{d}(\mathbf{x}_i, \mathbf{x}_j) \leq 2\varepsilon. \tag{9}$$

Thus, for all large enough $n_{\min}$ we have

$$\begin{aligned} \inf_{\substack{i \in \mathcal{G}_k \cap \{1..N\} \\ j \in \mathcal{G}_{k'} \cap \{1..N\} \\ k \neq k' \in 1..\kappa}} \hat{d}(\mathbf{x}_i, \mathbf{x}_j) &\geq \inf_{\substack{i \in \mathcal{G}_k \cap \{1..N\} \\ j \in \mathcal{G}_{k'} \cap \{1..N\} \\ k \neq k' \in 1..\kappa}} d(\rho_k, \rho_{k'}) - \hat{d}(\mathbf{x}_i, \rho_k) - \hat{d}(\mathbf{x}_j, \rho_{k'}) \\ &\geq \delta - 2\varepsilon \end{aligned} \tag{10}$$

where the first inequality follows from the triangle inequality, and the second inequality follows from (8) and the definition of δ. In words, (9) and (10) mean that the samples in $\mathcal{S}$ that are generated by the same process distribution are closer to each other than to the rest of the samples. Finally, for all $n_{\min}$ large enough to have (9) and (10) we obtain

$$\max_{i=1..N} \min_{k=1..\kappa-1} \hat{d}(\mathbf{x}_i, \mathbf{x}_{c_k}) \geq \delta - 2\varepsilon > \delta/2$$

where as specified by Algorithm 1, $c_1 := 1$ and $c_k := \operatorname{argmax}_{i=1..N} \min_{j=1..k-1} \hat{d}(\mathbf{x}_i, \mathbf{x}_{c_j})$, $k = 2..\kappa$. Hence, the indices $c_1, \ldots, c_\kappa$ will be chosen to index sequences generated by different process distributions. To derive the consistency statement, it remains to note that, by (9) and (10), each remaining sequence will be assigned to the cluster centre corresponding to the sequence generated by the same distribution. ■

4.2 Online Setting

The online version of the problem turns out to be more complicated than the offline one. The challenge is that, since new sequences arrive (potentially) at every time step, we can never rely on the distance estimates corresponding to all of the observed samples to be correct. Thus, as mentioned in the introduction, the main challenge can be identified with what we regard as "bad" sequences: recently-observed sequences, for which sufficient information has not yet been collected, and for which the estimates of the distance (with respect to any other sequence) are bound to be misleading. Thus, in particular, farthest-point initialisation would not work in this case. More generally, using any batch algorithm on all available data at every time step results in not only mis-clustering "bad" sequences, but also in clustering incorrectly those for which sufficient data are already available.

The solution, realised in Algorithm 2, is based on a *weighted combination* of several clusterings, each obtained by running the offline algorithm (Algorithm 1) on different portions of data. The clusterings are combined with weights that depend on the batch size and on the minimum inter-cluster distance. This last step of combining multiple clusterings with weights may be reminiscent of prediction with expert advice (see Cesa-Bianchi and Lugosi, 2006 for an overview), where experts are combined based on their past performance.

Algorithm 2 Online Clustering

INPUT: Number κ of target clusters
for $t = 1..\infty$ **do**
  Obtain new sequences $S(t) = \{\mathbf{x}_1^t, \cdots, \mathbf{x}_{N(t)}^t\}$
  Initialize the normalization factor: $\eta \leftarrow 0$
  Initialize the final clusters: $C_k(t) \leftarrow \varnothing,\ k = 1..\kappa$
  Generate $N(t) - \kappa + 1$ **candidate cluster centres:**
  for $j = \kappa..N(t)$ **do**
    $\{C_1^j, \ldots, C_\kappa^j\} \leftarrow \mathrm{Alg1}(\{\mathbf{x}_1^t, \cdots, \mathbf{x}_j^t\}, \kappa)$
    $\mu_k \leftarrow \min\{i \in C_k^j\}, k = 1..\kappa$ ▷ Set the smallest index as cluster centre.
    $(c_1^j, \ldots, c_\kappa^j) \leftarrow \mathrm{sort}(\mu_1, \ldots, \mu_\kappa)$ ▷ Sort the cluster centres increasingly.
    $\gamma_j \leftarrow \min_{k \neq k' \in 1..\kappa} \hat{d}(\mathbf{x}^t_{c_k^j}, \mathbf{x}^t_{c_{k'}^j})$ ▷ Calculate performance score.
    $w_j \leftarrow 1/j(j+1)$
    $\eta \leftarrow \eta + w_j \gamma_j$
  end for
  Assign points to clusters:
  for $i = 1..N(t)$ **do**
    $k \leftarrow \mathrm{argmin}_{k' \in 1..\kappa} \frac{1}{\eta} \sum_{j=1}^{N(t)} w_j \gamma_j \hat{d}(\mathbf{x}_i^t, \mathbf{x}^t_{c_{k'}^j})$
    $C_k(t) \leftarrow C_k(t) \cup \{i\}$
  end for
  OUTPUT: $\{C_1(t), \cdots, C_k(t)\}$
end for

However, the difference here is that the performance of each clustering cannot be measured directly.

More precisely, Algorithm 2 works as follows. Given a set $S(t)$ of $N(t)$ samples, the algorithm iterates over $j := \kappa, \ldots, N(t)$ where at each iteration Algorithm 1 is used to cluster the first j sequences $\{\mathbf{x}_1^t, \ldots, \mathbf{x}_j^t\}$ into κ clusters. In each cluster the sequence with the smallest index is assigned as the candidate cluster centre. A performance score γ_j is calculated as the minimum distance $\hat{d}$ between the κ candidate cluster centres obtained at iteration j. Thus, γ_j is an estimate of the minimum inter-cluster distance. At this point we have $N(t) - \kappa + 1$ sets of κ cluster centres $c_1^j, \ldots, c_\kappa^j,\ j = 1..N(t) - \kappa + 1$. Next, every sample $\mathbf{x}_i^t,\ i = 1..N(t)$ is assigned to a cluster, according to the weighted combination of the distances between $\mathbf{x}_i^t$ and the candidate cluster centres obtained at each iteration on j. More precisely, for each $i = 1..N(t)$ the sequence $\mathbf{x}_i^t$ is assigned to the cluster κ, where κ is defined as

$$k := \mathrm{argmin}_{k=1..\kappa} \sum_{j=\kappa}^{N(t)} w_j \gamma_j \hat{d}(\mathbf{x}_i^t, \mathbf{x}^t_{c_k^j}).$$

Theorem 12 *Algorithm 2 is strongly asymptotically consistent (in the online sense of Definition 9), provided the correct number of clusters κ is known, and the marginal distribution of each sequence $\mathbf{x}_i, i \in \mathbb{N}$ is stationary ergodic.*

Before giving the proof of Theorem 12, we provide an intuitive explanation as to how Algorithm 2 works. First, consider the following simple candidate solution. Take some fixed (reference) portion of the samples, run the batch algorithm on it, and then simply assign every remaining sequence to the nearest cluster. Since the offline algorithm is asymptotically consistent, this procedure would be asymptotically consistent as well, but only if we knew that the selected reference of the sequences contains at least one sequence sampled from each and every one of the κ distributions. However, there is no way to find a fixed (not growing with time) portion of data that would be guaranteed to contain a representative of each cluster (that is, of each time-series distribution). Allowing such a reference set of sequences to grow with time would guarantee that eventually it contains representatives of all clusters, but it would break the consistency guarantee for the reference set; since the set grows, this formulation effectively returns us back to the original online clustering problem.

A key observation we make to circumvent this problem is the following. If, for some $j \in \{\kappa, \ldots, N(t)\}$, each sample in the batch $\{\mathbf{x}_1^t, \ldots, \mathbf{x}_j^t\}$ is generated by *at most* $\kappa - 1$ process distributions, any partitioning of this batch into κ sets results in a minimum inter-cluster distance γ_j that, as follows from the asymptotic consistency of $\hat{d}$, converges to 0. On the other hand, if the set of samples $\{\mathbf{x}_1^t, \ldots, \mathbf{x}_j^t\}$ contains sequences generated by all κ process distributions, γ_j converges to a nonzero constant, namely, the minimum distance between the distinct process distributions $\rho_1, \ldots, \rho_\kappa$. In the latter case from some time on the batch $\{\mathbf{x}_1^t, \ldots, \mathbf{x}_j^t\}$ will be clustered correctly. Thus, instead of selecting one reference batch of sequences and constructing a set of clusters based on those, we consider all batches of sequences for $j = \kappa..N(t)$, and combine them with weights. Two sets of weights are involved in this step: γ_j and w_j, where

1. γ_j is used to penalise for small inter-cluster distance, cancelling the clustering results produced based on sets of sequences generated by less than κ distributions;

2. w_j is used to give precedence to chronologically earlier clusterings, protecting the clustering decisions from the presence of the (potentially "bad") newly formed sequences, whose corresponding distance estimates may still be far from accurate.

As time goes on, the batches in which not all clusters are represented will have their weight γ_j converge to 0, while the number of batches that have all clusters represented and are clustered correctly by the offline algorithm will go to infinity, and their total weight will approach 1. Note that, since we are combining different clusterings, it is important to use a consistent ordering of clusters, for otherwise we might sum up clusters generated by different distributions. Therefore, we always order the clusters with respect to the index of the first sequence in each cluster.

Proof [of Theorem 12] First, we show that for every $k \in 1..\kappa$ we have

$$\frac{1}{\eta} \sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) \to 0 \text{ a.s.} \tag{11}$$

Denote by δ the minimum nonzero distance between the process distributions:

$$\delta := \min_{k \neq k' \in 1..\kappa} \hat{d}(\rho_k, \rho_{k'}). \tag{12}$$

Fix $\varepsilon \in (0, \delta/4)$. We can find an index J such that $\sum_{j=J}^{\infty} w_j \le \varepsilon$. Let $S(t)|_j = \{\mathbf{x}_1^t, \cdots, \mathbf{x}_j^t\}$ denote the subset of $S(t)$ consisting of the first j sequences for $j \in 1..N(t)$. For $k = 1..\kappa$ let

$$s_k := \min\{i \in \mathcal{G}_k \cap 1..N(t)\} \tag{13}$$

index the first sequence in $S(t)$ that is generated by ρ_k where $\mathcal{G}_k,\ k = 1..\kappa$ are the ground-truth partitions given by Definition 7. Define

$$m := \max_{k \in 1..\kappa} s_k. \tag{14}$$

Recall that the sequence lengths $n_i(t)$ grow with time. Therefore, by Lemma 5 (consistency of $\hat{d}$) for every $j \in 1..J$ there exists some $T_1(j)$ such that for all $t \ge T_1(j)$ we have

$$\sup_{\substack{k \in 1..\kappa \\ i \in \mathcal{G}_k \cap \{1..j\}}} \hat{d}(\mathbf{x}_i^t, \rho_k) \le \varepsilon. \tag{15}$$

Moreover, by Theorem 11 for every $j \in m..J$ there exists some $T_2(j)$ such that $\mathrm{Alg1}(S(t)|_j, \kappa)$ is consistent for all $t \ge T_2(j)$. Let

$$T := \max_{\substack{i=1,2 \\ j \in 1..J}} T_i(j).$$

Recall that, by definition (14) of m, $S(t)|_m$ contains samples from all κ distributions. Therefore, for all $t \ge T$ we have

$$\begin{aligned}\inf_{k \ne k' \in 1..\kappa} \hat{d}(\mathbf{x}^t_{c_k^m}, \mathbf{x}^t_{c_{k'}^m}) &\ge \inf_{k \ne k' \in 1..\kappa} d(\rho_k, \rho_{k'}) - \sup_{k \ne k' \in 1..\kappa} (\hat{d}(\mathbf{x}^t_{c_k^m}, \rho_k) + \hat{d}(\mathbf{x}^t_{c_{k'}^m}, \rho_{k'})) \\ &\ge \delta - 2\varepsilon \ge \delta/2,\end{aligned} \tag{16}$$

where the first inequality follows from the triangle inequality and the second inequality follows from the consistency of $\mathrm{Alg1}(S(t)|_m, \kappa)$ for $t \ge T$, the definition of δ given by (12) and the assumption that $\varepsilon \in (0, \delta/4)$. Recall that (as specified in Algorithm 2) we have $\eta := \sum_{j=1}^{N(t)} w_j \gamma_j^t$. Hence, by (16) for all $t \ge T$ we have

$$\eta \ge w_m \delta/2. \tag{17}$$

By (17) and noting that by definition, $\hat{d}(\cdot,\cdot) \le 1$ for all $t \ge T$, for every $k \in 1..\kappa$ we obtain

$$\frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) \le \frac{1}{\eta}\sum_{j=1}^{J} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) + \frac{2\varepsilon}{w_m \delta}. \tag{18}$$

On the other hand, by the definition (14) of m, the sequences in $S(t)|_j$ for $j = 1..m-1$ are generated by *at most* $\kappa - 1$ out of the κ process distributions. Therefore, at every iteration on $j \in 1..m-1$ there exists at least one pair of distinct cluster centres that are generated by *the same* process distribution. Therefore, by (15) and (17), for all $t \ge T$ and every $k \in 1..\kappa$ we have,

$$\frac{1}{\eta}\sum_{j=1}^{m-1} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_i) \le \frac{1}{\eta}\sum_{j=1}^{m-1} w_j \gamma_j^t \le \frac{2\varepsilon}{w_m \delta}. \tag{19}$$

Noting that the clusters are ordered in the order of appearance of the distributions, we have $\mathbf{x}^t_{c_k^j} = \mathbf{x}^t_{s_k}$ for all $j = m..J$ and $k = 1..\kappa$, where the index s_k is defined by (13). Therefore, by (15) for all $t \ge T$ and every $k = 1..\kappa$ we have

$$\frac{1}{\eta}\sum_{j=m}^{J} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) = \frac{1}{\eta}\hat{d}(\mathbf{x}^t_{s_k}, \rho_k)\sum_{j=m}^{J} w_j \gamma_j^t \le \varepsilon. \tag{20}$$

Combining (18), (19), and (20) we obtain

$$\frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) \le \varepsilon(1 + \frac{4}{w_m \delta}). \tag{21}$$

for all $k = 1..\kappa$ and all $t \ge T$, establishing (11).

To finish the proof of the consistency, consider an index $i \in \mathcal{G}_r$ for some $r \in 1..\kappa$. By Lemma 5, increasing T if necessary, for all $t \ge T$ we have

$$\sup_{\substack{k \in 1..\kappa \\ j \in \mathcal{G}_k \cap 1..N}} \hat{d}(\mathbf{x}_j^t, \rho_k) \le \varepsilon. \tag{22}$$

For all $t \ge T$ and all $k \ne r \in 1..\kappa$ we have,

$$\begin{aligned}\frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}_i^t, \mathbf{x}^t_{c_k^j}) &\ge \frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}_i^t, \rho_k) - \frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) \\ &\ge \frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t (\hat{d}(\rho_k, \rho_r) - \hat{d}(\mathbf{x}_i^t, \rho_r)) - \frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}^t_{c_k^j}, \rho_k) \\ &\ge \delta - 2\varepsilon(1 + \frac{2}{w_m \delta}),\end{aligned} \tag{23}$$

where the first and second inequalities follow from the triangle inequality, and the last inequality follows from (22), (21) and the definition of δ. Since the choice of ε is arbitrary, from (22) and (23) we obtain

$$\operatorname*{argmin}_{k \in 1..\kappa} \frac{1}{\eta}\sum_{j=1}^{N(t)} w_j \gamma_j^t \hat{d}(\mathbf{x}_i^t, \mathbf{x}^t_{c_k^j}) = r. \tag{24}$$

It remains to note that for any fixed $N \in \mathbb{N}$ from some t on (24) holds for all $i = 1..N$, and the consistency statement follows. ■

4.3 Unknown Number κ of Clusters

So far we have shown that when κ is known in advance, consistent clustering is possible under the only assumption that the samples are generated by unknown stationary ergodic process distributions. However, as follows from the theoretical impossibility results

of Ryabko (2010c) discussed in Section 3, the correct number κ of clusters is not possible to be estimated with no further assumptions or additional constraints. One way to overcome this obstacle is to assume known rates of convergence of frequencies to the corresponding probabilities. Such rates are provided by assumptions on the mixing rates of the process distributions that generate the data.

Here we will show that under some assumptions on the mixing rates (and still without making any modelling or independence assumptions), consistent clustering is possible when the number of clusters is unknown.

The purpose of this section, however, is not to find the weakest assumptions under which consistent clustering (with κ unknown) is possible, nor is it to provide sharp bounds under the assumptions considered; our only purpose here is to demonstrate that asymptotic consistency is achievable in principle when the number of clusters is unknown, under some mild nonparametric assumptions on the time-series distributions. More refined analysis could yield sharper bounds under weaker assumptions, such as those in, for example, (Bosq, 1996; Rio, 1999; Doukhan, 1994; Doukhan et al., 2010).

We introduce *mixing coefficients*, mainly following Rio (1999) in formulations. Informally, mixing coefficients of a stochastic process measure how fast the process forgets about its past. Any one-way infinite stationary process $X_1, X_2, \dots$ can be extended backwards to make a two-way infinite process $\dots, X_{-1}, X_0, X_1, \dots$ with the same distribution. In the definition below we assume such an extension. Define the φ-mixing coefficients of a process μ as

$$\varphi_n(\mu) = \sup_{A\in\sigma(X_{-\infty..k}),B\in\sigma(X_{k+n..\infty}),\mu(A)\neq 0} |\mu(B|A) - \mu(B)|, \tag{25}$$

where $\sigma(..)$ stands for the σ-algebra generated by random variables in brackets. These coefficients are nonincreasing. Define also

$$\theta_n(\mu) := 2 + 8(\varphi_1(\mu) + \dots + \varphi_n(\mu)).$$

A process μ is called uniformly *φ-mixing* if $\varphi_n(\mu) \to 0$. Many important classes of processes satisfy mixing conditions. For example, a stationary irreducible aperiodic Hidden Markov process with finitely many states is uniformly φ-mixing with coefficients decreasing exponentially fast. Other probabilistic assumptions can be used to obtain bounds on the mixing coefficients, see, e.g., (Bradley, 2005) and references therein.

The method that we propose for clustering mixing time series in the offline setting, namely *Algorithm 3*, is very simple. Its inputs are: a set $S := \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ of samples each of length n_i, $i = 1..N$, the threshold level $\delta \in (0,1)$ and the parameters $m, l \in \mathbb{N}$, $B^{m,l,n}$. The algorithm assigns to the same cluster all samples which are at most δ-far from each other, as measured by $\hat{d}$ with $m_n = m, l_n = l$ and the summation over $B^{m,l}$ restricted to $B^{m,l,n}$. The sets $B^{m,l,n}$ have to be chosen so that in asymptotic they cover the whole space, $\cup_{n\in\mathbb{N}} B^{m,l,n} = B^{m,l}$. For example, $B^{m,l,n}$ may consist of the first b_n cubes around the origin, where $b_n \to \infty$ is a parameter sequence. We do not give a pseudocode implementation of this algorithm, since it is rather clear.

The idea is that the threshold level $\delta = \delta_n$ is selected according to the smallest sample-length $n := \min_{i=1..N} n_i$ and the (known bounds on) mixing rates of the process ρ generating the samples (see Theorem 13). As we show in Theorem 13, if the distribution of the samples satisfies $\varphi_n(\rho) \le \varphi_n \to 0$, where φ_n are known, then one can select (based on φ_n only)

the parameters of Algorithm 3 in such a way that it is weakly asymptotically consistent. Moreover, a bound on the probability of error before asymptotic is provided.

Theorem 13 (Algorithm 3 is consistent for unknown κ) *Fix sequences $m_n, l_n, b_n \in \mathbb{N}$, and let, for each $m, l \in \mathbb{N}$, $B^{m,l,n} \subset B^{m,l}$ be an increasing sequence of finite sets such that $\cup_{n\in\mathbb{N}} B^{m,l,n} = B^{m,l}$. Set $b_n := \max_{l\le l_n, m\le m_n} |B^{m,l,n}|$ and $n := \min_{i=1..N} n_i$. Let also $\delta_n \in (0,1)$. Let $N \in \mathbb{N}$, and suppose that the samples $\mathbf{x}_1, \dots, \mathbf{x}_N$ are generated in such a way that the (unknown marginal) distributions ρ_k, $k = 1..\kappa$ are stationary ergodic and satisfy $\varphi_n(\rho_k) \le \varphi_n$, for all $k = 1..\kappa$, $n \in \mathbb{N}$. Then there exist constants ε_ρ, δ_ρ and n_ρ that depend only on ρ_k, $k = 1..\kappa$, such that for all $\delta_n < \delta_\rho$ and $n > n_\rho$, Algorithm 3 satisfies*

$$P(T \neq \mathcal{G}|_N) \le 2N(N+1)(m_n l_n b_n \gamma_{n/2}(\delta_n) + \gamma_{n/2}(\varepsilon_\rho)) \tag{26}$$

where

$$\gamma_n(\varepsilon) := \sqrt{e}\exp(-n\varepsilon^2/\theta_n),$$

T is the partition output by the algorithm and $\mathcal{G}|_N$ is the ground-truth clustering. In particular, if $\varphi_n = o(1)$, then, selecting the parameters in such a way that $\delta_n = o(1)$, $m_n, l_n, b_n = o(n)$, $m_n, l_n \to \infty$, $\cup_{k\in\mathbb{N}} B^{m,l,k} = B^{m,l}$, $b_n^{m,l} \to \infty$, for all $m, l \in \mathbb{N}$, $\gamma_n(const) = o(1)$, and, finally, $m_n l_n b_n \gamma_n(\delta_n) = o(1)$, as is always possible, Algorithm 3 is weakly asymptotically consistent *(with the number of clusters κ unknown).*

Proof We use the following bound from (Rio, 1999, Corollary 2.1): for any zero-mean $[-1,1]$-valued random process $Y_1, Y_2, \dots$ with distribution P and every $n \in \mathbb{N}$ we have

$$P\left(|\sum_{i=1}^n Y_i| > n\varepsilon\right) \le \gamma_n(\varepsilon). \tag{27}$$

For every $j = 1..N$, every $m < n$, $l \in \mathbb{N}$, and $B \in B^{m,l}$, define the $[-1,1]$-valued processes $\mathbf{Y}^j := Y_1^j, Y_2^j, \dots$ as

$$Y_t^j := \mathbb{I}\{(X_t^j, \dots, X_{t+m-1}^j) \in B\} - \rho_k(X_{1..m}^j \in B),$$

where ρ_k is the marginal distribution of X^j (that is, k is such that $j \in \mathcal{G}_k$). It is easy to see that φ-mixing coefficients for this process satisfy $\varphi_n(\mathbf{Y}^j) \le \varphi_{n-2m}$. Thus, from (27) we have

$$P(|\nu(X_{1..n_j}^j, B) - \rho_k(X_{1..m}^j \in B)| > \varepsilon/2) \le \gamma_{n-2m_n}(\varepsilon/2). \tag{28}$$

Then for every $i, j \in \mathcal{G}_k \cap 1..N$ for some $k \in 1..\kappa$ (that is, $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same ground-truth cluster) we have

$$P(|\nu(X_{1..n_i}^i, B) - \nu(X_{1..n_j}^j, B)| > \varepsilon) \le 2\gamma_{n-2m_n}(\varepsilon/2).$$

Using the union bound, summing over m, l, and B, we obtain

$$P(\hat{d}(\mathbf{x}_i, \mathbf{x}_j) > \varepsilon) \le 2m_n l_n b_n \gamma_{n-2m_n}(\varepsilon/2). \tag{29}$$

Next, let $i \in \mathcal{G}_k \cap 1..N$ and $j \in \mathcal{G}_{k'} \cap 1..N$ for $k \neq k' \in 1..\kappa$ (i.e., $\mathbf{x}_i, \mathbf{x}_j$ are in two different target clusters). Then, for some $m_{i,j}, l_{i,j} \in \mathbb{N}$ there is $B_{i,j} \in B^{m_{i,j},l_{i,j}}$ such that for some $\tau_{i,j} > 0$ we have

$$|\rho_k(X^i_{1..|B_{i,j}|} \in B_{i,j}) - \rho_{k'}(X^j_{1..|B_{i,j}|} \in B_{i,j})| > 2\tau_{i,j}.$$

Then for every $\varepsilon < \tau_{i,j}/2$ we have

$$P(|\nu(X^i_{1..n_i}, B_{i,j}) - \nu(X^j_{1..n_j}, B_{i,j})| < \varepsilon) \leq 2\gamma_{n-2m_{i,j}}(\tau_{i,j}). \tag{30}$$

Moreover, for $\varepsilon < w_{m_{i,j}} w_{l_{i,j}} \tau_{i,j}/2$ we have

$$P(\hat{d}(\mathbf{x}_i, \mathbf{x}_j) < \varepsilon) \leq 2\gamma_{n-2m_{i,j}}(\tau_{i,j}). \tag{31}$$

Define

$$\delta_\rho := \min_{\substack{k\neq k'\in 1..\kappa \\ i\in\mathcal{G}_k\cap 1..N \\ j\in\mathcal{G}_{k'}\cap 1..N}} w_{m_{i,j}} w_{l_{i,j}} \tau_{i,j}/2, \quad \varepsilon_\rho := \min_{\substack{k\neq k'\in 1..\kappa \\ i\in\mathcal{G}_k\cap 1..N \\ j\in\mathcal{G}_{k'}\cap 1..N}} \tau_{i,j}/2$$

$$n_\rho := 2 \max_{\substack{k\neq k'\in 1..\kappa \\ i\in\mathcal{G}_k\cap 1..N \\ j\in\mathcal{G}_{k'}\cap 1..N}} m_{i,j}.$$

Clearly, from this and (30), for every $\delta < 2\delta_\rho$ and $n > n_\rho$ we obtain

$$P(\hat{d}(\mathbf{x}_i, \mathbf{x}_j) < \delta) \leq 2\gamma_{n/2}(\varepsilon_\rho). \tag{32}$$

Algorithm 3 produces correct results if for every pair i, j we have $\hat{d}(\mathbf{x}_i, \mathbf{x}_j) < \delta_n$ if and only if $i, j \in \mathcal{G}_k \cap 1..N$ for some $k \in 1..\kappa$. Therefore, taking the bounds (29) and (32) together for each of the $N(N+1)/2$ pairs of samples, we obtain (26). ■

For the online setting, consider the following simple extension of Algorithm 3, that we call Algorithm 3'. It applies Algorithm 3 to the first N_t sequences, where the parameters of Algorithm 3 and N_t are chosen in such a way that the bound (26) with $N = N_t$ is $o(1)$ and $N_t \to \infty$ as time t goes to infinity. It then assigns each of the remaining sequences ($\mathbf{x_i}$, $i > N_t$) to the nearest cluster. Note that in this case the bound (26) with $N = N_t$ bounds the error of Algorithm 3' on the first N_t sequences, as long as all of the κ clusters are already represented among the first N_t sequences. Since $N_t \to \infty$, we can formulate the following result.

Theorem 14 *Algorithm 3' is weakly asymptotically consistent in the online setting when the number κ of clusters is unknown, provided that the assumptions of Theorem 13 apply to the first N sequences $\mathbf{x}_1, \ldots, \mathbf{x}_N$ for every $N \in \mathbb{N}$.*

4.4 Extensions

In this section we show that some simple, yet rather general extensions of the main results of this paper are possible, namely allowing for non-stationarity and for slight differences in distributions in the same cluster.

4.4.1 AMS Processes and Gradual Changes

Here we argue that our results can be strengthened to a more general case where the process distributions that generate the data are Asymptotically Mean Stationary (AMS) ergodic. Throughout the paper we have been concerned with stationary ergodic process distributions. Recall from Section 2 that a process ρ is *stationary* if for any $i, j \in 1..n$ and $B \in B^m$, $m \in \mathbb{N}$, we have $\rho(X_{1..j} \in B) = \rho(X_{i..i+j-1} \in B)$. A stationary process is called ergodic if the limiting frequencies converge to their corresponding probabilities, so that for all $B \in \mathcal{B}$ with probability 1 we have $\lim_{n\to\infty} \nu(X_{1..n}, B) = \rho(B)$. This latter convergence of all frequencies is the only property of the process distributions that is used in the proofs (via Lemma 5) which give rise to our consistency results. We observe that this property also holds for a more general class of processes, namely those that are AMS ergodic. Specifically, a process ρ is called *AMS* if for every $j \in 1..n$ and $B \in B^m, m \mathbb{N}$ the series $\lim_{n\to\infty} \sum_{i=1}^{n-j+1} \frac{1}{n} \rho(X_{i..i+j-1} \in B)$ converges to a limit $\bar{\rho}(B)$, which forms a measure, i.e. $\bar{\rho}(X_{1..j} \in B) := \bar{\rho}(B)$, $B \in B^m$, $m \in \mathbb{N}$, called *asymptotic mean* of ρ. Moreover, if ρ is an AMS process, then for every $B \in B^m$, $m \in \mathbb{N}$, the frequency $\nu(X_{1..n}, B)$ converges ρ-a.s. to a random variable with mean $\bar{\rho}(B)$. Similarly to stationary processes, if the random variable to which $\nu(X_{1..n}, B)$ converges is a.s. constant, then ρ is called AMS ergodic. More information on AMS processes can be found in the work (Gray, 1988). However, the main characteristic pertaining to our work is that the class of all processes with AMS properties is composed precisely of those processes for which the almost sure convergence of frequencies to the corresponding probabilities holds. It is thus easy to check that all of the asymptotic results of Sections 4.1, 4.2 carry over to the more general setting where the unknown process distributions that generate the data are AMS ergodic.

4.4.2 Strictly Separated Clusters of Distributions

So far we have defined a cluster as a set of sequences generated by the same distribution. This seems to capture rather well the notion that in the same cluster the objects can be very different (as is the case for stochastically generated sequences), yet are intrinsically of the same nature (they have the same law).

However, one may wish to generalise this further, and allow each sequence to be generated by a different distribution, yet requiring that in the same clusters distributions must be close. Unlike the original formulation, such an extension would require fixing some similarity measure between distributions. The results of the preceding sections suggest using the distributional distance for this purpose.

Specifically, as discussed in Section 4.1, in our formulation, from some time on, the sequences possess the so-called strict separation property in the $\hat{d}$ distance: sequences in the same target cluster are closer to each other than to those in other clusters. One possible way to relax the considered setting is to impose the strict separation property on the distributions that generate the data. Here the separation would be with respect to the distributional distance d. That is, each sequence $\mathbf{x}_i, i = 1..N$, may be generated by its own distribution ρ_i, but the distributions $\{\rho_i : i = 1..N\}$ can be clustered in such a way that the resulting clustering has the strict separation property with respect to d. The goal would then be to recover this clustering based on the given samples. In fact, it can be shown that the offline Algorithm 1 of Section 4.1 is consistent in this setting as well. How this transfers

to the online setting remains open. For the offline case, we can formulate the following result, whose proof is analogous to that of Theorem 11.

Theorem 15 *Assume that each sequence $\mathbf{x}_i$, $i = 1..N$ is generated by a stationary ergodic distribution ρ_i. Assume further that the set of distributions $\{1, \dots, N\}$ admits a partitioning $\mathcal{G} = \{\mathcal{G}_1, \dots, \mathcal{G}_k\}$ that has the strict separation property with respect to d: for all $i, j = 1..k, i \neq j$, for all $\rho_1, \rho_2 \in \mathcal{G}_i$ and all $\rho_3 \in \mathcal{G}_j$ we have $d(\rho_1, \rho_2) < d(\rho_1, \rho_3)$. Then Algorithm 1 is strongly asymptotically consistent, in the sense that almost surely from some $n = \min\{n_1, \dots, n_N\}$ on it outputs the set $\mathcal{G}$.*

5. Computational Considerations

In this section we show that all of the proposed methods are efficiently computable. This claim is further illustrated by the experimental results in the next section. Note, however, that since the results presented are asymptotic, the question of what is the best achievable computational complexity of an algorithm that still has the same asymptotic performance guarantees is meaningless: for example, an algorithm could throw away most of the data and still be asymptotically consistent. This is why we do not attempt to find a resource-optimal way of computing the methods presented.

First, we show that calculating $\hat{d}$ is at most quadratic (up to log terms), and quasilinear if we use $m_n = \log n$. Let us begin by showing that calculating $\hat{d}$ is fully tractable with $m_n, l_n \equiv \infty$. First, observe that for fixed m and l, the sum

$$T^{m,l} := \sum_{B \in B^{m,l}} |\nu(X^1_{1..n_1}, B) - \nu(X^2_{1..n_2}, B)| \tag{33}$$

has not more than $n_1 + n_2 - 2m + 2$ nonzero terms (assuming $m \leq n_1, n_2$; the other case is obvious). Indeed, there are $n_i - m + 1$ tuples of size m in each sequence $\mathbf{x}_i$, $i = 1, 2$ namely, $X^i_{1..m}, X^i_{2..m+1}, \dots, X^i_{n_1-m+1..n_1}$. Therefore, $T^{m,l}$ can be obtained by a finite number of calculations.

Furthermore, let

$$s = \min_{\substack{X^1_i \neq X^2_j \\ i=1..n_1, j=1..n_2}} |X^1_i - X^2_j|, \tag{34}$$

and observe that $T^{m,l} = 0$ for all $m > n$ and for each m, for all $l > \log s^{-1}$ the term $T^{m,l}$ is constant. That is, for each fixed m we have

$$\sum_{l=1}^{\infty} w_m w_l T^{m,l} = w_m w_{\log s^{-1}} T^{m, \log s^{-1}} + \sum_{l=1}^{\log s^{-1}} w_m w_l T^{m,l}$$

so that we simply double the weight of the last nonzero term. (Note also that s is bounded above by the length of the binary precision in representing the random variables X^i_j.) Thus, even with $m_n, l_n \equiv \infty$ one can calculate $\hat{d}$ precisely. Moreover, for a fixed $m \in 1..\log n$ and $l \in 1..\log s^{-1}$ for every sequence $\mathbf{x}_i$, $i = 1, 2$ the frequencies $\nu(\mathbf{x}_i, B)$, $B \in B^{m,l}$ may be calculated using suffix trees or suffix arrays, with $\mathcal{O}(n)$ worst case construction and search complexity (see, e.g., Ukkonen, 1995). Searching all $z := n - m + 1$ occurrences of

subsequences of length m results in $\mathcal{O}(m + z) = \mathcal{O}(n)$ complexity. This brings the overall computational complexity of (3) to $\mathcal{O}(nm_n \log s^{-1})$; this can potentially be improved using specialized structures, e.g., (Grossi and Vitter, 2005).

The following consideration can be used to set m_n. For a fixed l the frequencies $\nu(\mathbf{x}_i, B)$, $i = 1, 2$ of cells in $B \in B^{m,l}$ corresponding to values of m much larger than $\log_n$ (in the asymptotic sense, that is, if $\log_n = o(m)$) are not, in general, consistent estimates of their probabilities, and thus only add to the estimation error. More specifically, for a subsequence $X_{j..j+m}$ with $j = 1..n-m$ of length m the probability $\rho_i(X_{j..j+m} \in B)$, $i = 1, 2$ is of order 2^{-mh_i}, $i = 1, 2$ where h_i denotes the entropy rate of ρ_i, $i = 1, 2$. Moreover, under some (general) conditions (including $h_i > 0$) one can show that, asymptotically, a string of length of order $\log n / h_i$ on average occurs at most once in a string of length n (Kontoyiannis and Suhov, 1994). Therefore, subsequences of length m larger than $\log n$ are typically met 0 or 1 times, and thus are not consistent estimates of probabilities. By the above argument, one can use m_n of order $\log n$. To choose $l_n < \infty$ one can either fix some constant based on the bound on the precision in real computations, or choose it in such a way that each cell B^{m,l_n} contains no more than $\log n$ points for all $m = 1..\log n$ largest values of l_n.

5.1 Complexity of the Algorithms.

The computational complexity of the presented algorithms is dominated by the complexity of calculations of $\hat{d}$ between different pairs of sequences. Thus, it is sufficient to bound the number of pairwise $\hat{d}$ computations.

It is easy to see that the offline algorithm for the case of known κ (Algorithm 1) requires at most κN distance calculations, while for the case of unknown κ all N^2 distance calculations are necessary.

The computational complexity of the updates in the online algorithm can be computed as follows. Assume that the pairwise distance values are stored in a database D, and that for every sequence $\mathbf{x}^{t-1}_i$, $i \in \mathbb{N}$ we have already constructed a suffix tree, using for example, the online algorithm of Ukkonen (1995). At time step t, a new symbol X is received. Let us first calculate the required computations to update D. We have two cases, either X forms a new sequence, so that $N(t) = N(t-1) + 1$, or it is the subsequent element of a previously received segment, say, $\mathbf{x}^t_j$ for some $j \in 1..N(t)$, so that $n_j(t) = n_j(t-1) + 1$. In either case, let $\mathbf{x}^t_j$ denote the updated sequence. Note that for all $i \neq j \in 1..N(t)$ we have $n_i(t) = n_i(t-1)$. Recall the notation $\mathbf{x}^t_i := X^{(i)}_1, \dots X^{(i)}_{n_i(t)}$ for $i \in 1..N(t)$. In order to update D we need to update the distance between $\mathbf{x}^t_j$ and $\mathbf{x}^t_i$ for all $i \neq j \in N(t)$. Thus, we need to search for all m_n new patterns induced by the received symbol X, resulting in complexity at most $\mathcal{O}(N(t) m_n^2 l_n)$. Let $n(t) := \max\{n_1(t), \dots n_{N(t)}(t)\}$, $t \in \mathbb{N}$. As discussed previously, we let $m_n := \log n(t)$; we also define $l_n := \log s(t)^{-1}$ where

$$s(t) := \min_{\substack{i,j \in 1..N(t) \\ u=1..n_i(t), v=1..n_j(t), X^{(i)}_u \neq X^{(j)}_v}} |X^{(i)}_u - X^{(j)}_v|, \ t \in \mathbb{N}.$$

Thus, the per symbol complexity of updating D is at most $\mathcal{O}(N(t) \log^3 n(t))$. However, note that if $s(t)$ decreases from one time step to the next, updating D will have a complexity

of order equivalent to its complete construction, resulting in a computational complexity of order $\mathcal{O}(N(t)n(t)\log^2 n(t))$. Therefore, we avoid calculating $s(t)$ at every time step; instead, we update $s(t)$ at prespecified time steps so that for every $n(t)$ symbols received, D is reconstructed at most $\log n(t)$ times. (This can be done, for example, by recalculating $s(t)$ at time steps where $n(t)$ is a power of 2.) It is easy to see that with the database D of distance values at hand, the rest of the computations are of order at most $\mathcal{O}(N(t)^2)$. Thus, the computational complexity of updates in Algorithm 2 is at most $\mathcal{O}(N(t)^2+N(t)\log^3 n(t))$.

6. Experimental Results

In this section we present empirical evaluations of Algorithms 1 and 2 on both synthetically generated and real data.

6.1 Synthetic Data

We start with synthetic experiments. In order for the experiments to reflect the generality of our approach we have selected time-series distributions that, while being stationary ergodic, cannot be considered as part of any of the usual smaller classes of processes, and are difficult to approximate by finite-state models. Namely, we consider rotation processes used, for example, by Shields (1996) as an example of stationary ergodic processes that are not B-processes. Such time series cannot be modelled by a hidden Markov model with a finite or countably infinite set of states. Moreover, while k-order Markov (or hidden Markov) approximations of this process converge to it in the distributional distance d, they do not converge to it in the $\bar{d}$ distance, a stronger distance than d whose empirical approximations are often used to study general (non-Markovian) processes (e.g., Ornstein and Weiss, 1990).

6.1.1 Time Series Generation

To generate a sequence $\mathbf{x} = X_{1..n}$ we proceed as follows: Fix some parameter $\alpha \in (0,1)$. Select $r_0 \in [0,1]$; then, for each $i = 1..n$ obtain r_i by shifting r_{i-1} by α to the right, and removing the integer part, i.e. $r_i := r_{i-1} + \alpha - \lfloor r_{i-1} + \alpha \rfloor$. The sequence $\mathbf{x} = (X_1, X_2, \cdots)$ is then obtained from r_i by thresholding at 0.5, that is $X_i := \mathbb{I}\{r_i > 0.5\}$. If α is irrational then $\mathbf{x}$ forms a stationary ergodic time series. (We simulate α by a `longdouble` with a long mantissa.)

For the purpose of our experiments, first we fix $\kappa := 5$ difference process distributions specified by $\alpha_1 = 0.31...$, $\alpha_2 = 0.33...$, $\alpha_3 = 0.35...$, $\alpha_4 = 0.37...$, $\alpha_5 = 0.39....$ The parameters α_i are intentionally selected to be close, in order to make the process distributions harder to distinguish. Next we generate an $N \times M$ data matrix $\mathbf{X}$, each row of which is a sequence generated by one of the process distributions. Our task in both the online and the batch setting is to cluster the rows of $\mathbf{X}$ into $\kappa = 5$ clusters.

6.1.2 Batch Setting

In this experiment we demonstrate that in the batch setting, the clustering errors corresponding to both the online and the offline algorithms converge to 0 as the sequence-lengths grow. To this end, at every time step t we take an $N \times n(t)$ submatrix $\mathbf{X}|_{\mathbf{n(t)}}$ of $\mathbf{X}$ composed of the rows of $\mathbf{X}$ terminated at length $n(t)$, where $n(t) = 5t$. Then at each iteration we let

each of the algorithms, (online and offline) cluster the rows of $\mathbf{X}|_{\mathbf{n(t)}}$ into five clusters, and calculate the clustering error rate of each algorithm. As shown in Figure 2 (top) the error rate of each algorithm decreases with sequence length.

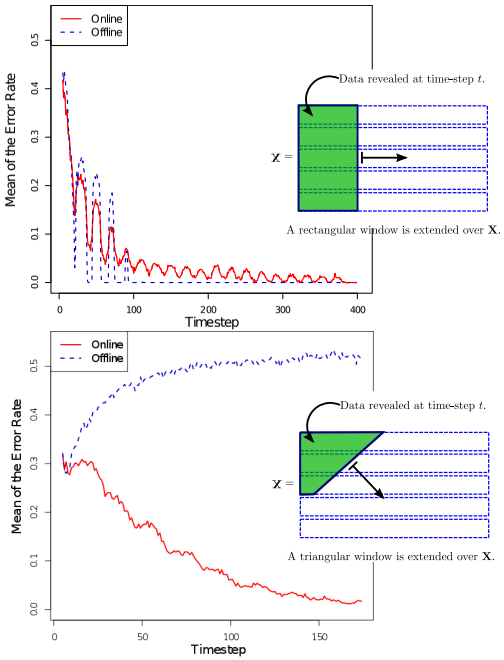


Figure 2: Top: error rate vs. sequence length in batch setting. Bottom: error rate vs. Number of observed samples in online setting. (error rates averaged over 100 runs.)

6.1.3 Online Setting

In this experiment we demonstrate that, unlike the online algorithm, the offline algorithm is consistently confused by the new sequences arriving at each time step in an online setting. To simulate an online setting, we proceed as follows: At every time step t, a triangular window is used to reveal the first $1..n_i(t)$, $i = 1..t$ elements of the first t rows of the data matrix $\mathbf{X}$, with $n_i(t) := 5(t-i)+1$, $i = 1..t$. This gives a total of t sequences, each of length $n_i(t)$, for $i = 1..t$, where the i^{th} sequence for $i = 1..t$ corresponds to the i^{th} row of $\mathbf{X}$ terminated at length $n_i(t)$. At every time step t the online and offline algorithms are each used in turn to cluster the observed t sequences into five clusters. Note that the performance

of both algorithms is measured on all sequences available at a given time, not on a fixed batch of sequences. As shown in Figure 2 (bottom), in this setting the clustering error rate of the offline algorithm remains consistently high, whereas that of the online algorithm converges to zero.

6.2 Real Data

As an application we consider the problem of clustering motion capture sequences, where groups of sequences with similar dynamics are to be identified. Data is taken from the Motion Capture database (MOCAP) which consists of time-series data representing human locomotion. The sequences are composed of marker positions on human body which are tracked spatially through time for various activities.

We compare our results to those obtained with two other methods, namely those of Li and Prakash (2011) and Jebara et al. (2007). Note that we have not implemented these reference methods, rather we have taken the numerical results directly from the corresponding articles. In order to have common grounds for each comparison we use the same sets of sequences[2] and the same means of evaluation as those used by Li and Prakash (2011); Jebara et al. (2007).

In the paper by Li and Prakash (2011) two MOCAP data sets[3] are used, where the sequences in each data set are labelled with either running or walking as annotated in the database. Performance is evaluated via the conditional entropy S of the true labelling with respect to the prediction, i.e., $S := -\sum_{i,j} \frac{\mathcal{M}_{ij}}{\sum_{i',j'} \mathcal{M}_{i'j'}} \log \frac{\mathcal{M}_{ij}}{\sum_{j'} \mathcal{M}_{ij'}}$ where $\mathcal{M}$ denotes the clustering confusion matrix. The motion sequences used by Li and Prakash (2011) are reportedly trimmed to equal duration. However, we use the original sequences as our method is not limited by variation in sequence lengths. Table 1 lists performance of Algorithm 1 as well as that reported for the method of Li and Prakash (2011); Algorithm 1 performs consistently better.

In the paper (Jebara et al., 2007) four MOCAP data sets[4] are used, corresponding to four motions: run, walk, jump and forward jump. Table 2 lists performance in terms of accuracy. The data sets in Table 2 constitute two types of motions:

1. motions that can be considered ergodic: walk, run, run/jog (displayed above the double line), and

2. non-ergodic motions: single jumps (displayed below the double line).

As shown in Table 2, Algorithm 1 achieves consistently better performance on the first group of data sets, while being competitive (better on one and worse on the other) on the non-ergodic motions. The time taken to complete each task is in the order of few minutes on a standard laptop computer.

2. The subject's right foot was used as marker position.
3. The corresponding subject references are #16 and #35.
4. The corresponding subject references are #7, #9, #13, #16 and #35.

	Data set	(Li and Prakash, 2011)	Algorithm 1
1.	Walk vs. Run (#35)	0.1015	**0**
2.	Walk vs. Run (#16)	0.3786	**0.2109**

Table 1: Comparison with the work (Li and Prakash, 2011): Performance in terms of entropy; data sets concern ergodic motion captures.

	Data set	(Jebara et al., 2007)	Algorithm 1
1.	Run(#9) vs. Run/Jog(#35)	**100%**	**100%**
2.	Walk(#7) vs. Run/Jog(#35)	95%	**100%**
3.	Jump vs. Jump fwd.(#13)	87%	**100%**
4.	Jump vs. Jump fwd.(#13, 16)	**66%**	60%

Table 2: Comparison with the work (Jebara et al., 2007): Performance in terms of accuracy; Rows 1 & 2 concern ergodic, Rows 3 & 4 concern non-ergodic motion captures.

7. Discussion

We have proposed a natural notion of consistency for clustering time series in both the online and the offline settings. While in this work we have taken some first steps in investigating the theoretical and algorithmic questions arising in the proposed framework, there are many open problems and exciting directions for future research remaining to be explored. Some of these are discussed in this section.

Rates, optimality. The main focus of this work is on the most general case of highly dependent time series. On the one hand, this captures best the spirit of the unsupervised learning problem in question: the nature of the data is completely unknown, and one tries to find some structure in it. On the other hand, as discussed above, in this generality rates of convergence and finite-sample performance guarantees are provably impossible to obtain, and thus one cannot argue about optimality. While we have provided some results on a more restrictive setting (time series with mixing, Section 4.3), the question of what the optimal performance guarantees are for different classes of time series remains open. In fact, the first interesting question in this direction is not about time series with mixing, but about i.i.d. series. What is the minimal achievable probability of clustering error in this setting, for finite sample sizes, and what algorithms attain it?

Online setting: bad points. In the online setting of Section 4.2, we have assumed that the length of each sequence grows to infinity with time. While this is a good first approximation, this assumption may not be practical. It is interesting to consider the situation in which some sequences stop growing at some point; moreover, it can be assumed that such sequences are not representative of the corresponding distribution. While this clearly makes the problem much more difficult, already in the setting considered in this work we have dealt with "bad" sequences at each time step: these are those sequences which are as yet too short to be informative. This hints at the possibility of obtaining consistent algorithms in the extended setting outlined.

Other metrics and non-metric-based methods. All of the methods presented in this paper are based on the distributional distance d. The main property of this distance that we

exploit is that it can be estimated consistently. In principle, one can use other distances with this property, in order to obtain consistent clustering algorithms. While there are not many known distances that can be consistently estimated for arbitrary stationary ergodic distributions, there is at least one, namely the telescope distance recently introduced by Ryabko and Mary (2013). Moreover, the definition of consistency proposed does not entail the need to use a distance between time-series distributions. As an alternative, the use of compression-based methods can be considered. Such methods have been used to solve various statistical problems concerning stationary ergodic time series (Ryabko and Astola, 2006; Ryabko, 2010a). Compression-based methods have also been used for clustering time-series data before, albeit without consistency analysis, by Cilibrasi and Vitanyi (2005). Combining our consistency framework with these compression-based methods is another promising direction for further research.

Acknowledgments

This paper is an extended version of two conference papers: (Ryabko, 2010b) and (Khaleghi et al., 2012). Most of the work by Azadeh Khaleghi has been done during her PhD at INRIA Lille - Nord Europe. This work is supported by the French Ministry of Higher Education and Research, by FP7/2007-2013 under grant agreements 270327 (CompLACS), by the Nord-Pas-de-Calais Regional Council and FEDER, and by the European Research Council (SMAC-ERC-280032). Jérémie Mary and Philippe Preux acknowledge the support of INRIA. Jérémie Mary further acknowledges INRIA for the support through partial secondments.

References

F.R. Bach and M.I. Jordan. Learning graphical models for stationary time series. *IEEE Transactions on Signal Processing*, 52(8):2189 – 2199, Aug. 2004.

M.F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 671–680, 2008.

C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(7):719–725, 2000.

P. Billingsley. Statistical methods in Markov chains. *The Annals of Mathematical Statistics*, 32(1):12–40, 1961.

P. Billingsley. *Convergence of Probability Measures*. John Wiley & Sons, 1999.

D. Bosq. *Nonparametric Statistics for Stochastic Processes*. Estimation and Prediction. Springer, 1996.

R.C. Bradley. Basic properties of strong mixing conditions: A survey and some open questions. *Probability Surveys*, 2:107–144, 2005.

E. Carlstein and S. Lele. Nonparametric change-point estimation for data from an ergodic sequence. *Theory of Probability & its Applications*, 38(4):726–733, 1994.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

R. Cilibrasi and P.M.B. Vitanyi. Clustering by compression. *IEEE Transactions on Information Theory*, 51:1523–1545, 2005.

P. Doukhan. *Mixing*. Springer, 1994.

P. Doukhan, G. Lang, D. Surgailis, and G. Teyssière. *Dependence in Probability and Statistics*. Springer, 2010.

R. Gray. *Probability, Random Processes, and Ergodic Properties*. Springer Verlag, 1988.

R. Grossi and J.S. Vitter. Compressed suffix arrays and suffix trees with applications to text indexing and string matching. *SIAM Journal on Computing*, 35(2):378–407, 2005.

M. Gutman. Asymptotically optimal classification for multiple tests with empirically observed statistics. *IEEE Transactions on Information Theory*, 35(2):402–408, 1989.

T. Jebara, Y. Song, and K. Thadani. Spectral clustering and embedding with hidden Markov models. *Machine Learning: ECML 2007*, pages 164–175, 2007.

I. Katsavounidis, C.-C. Jay Kuo, and Zhen Zhang. A new initialisation technique for generalised Lloyd iteration. *IEEE Signal Processing Letters*, 1:144–146, 1994.

A. Khaleghi and D. Ryabko. Locating changes in highly-dependent data with unknown number of change points. In *Neural Information Processing Systems (NIPS)*, Lake Tahoe, Nevada, United States, 2012.

A. Khaleghi, D. Ryabko, J. Mary, and P. Preux. Online clustering of processes. In *International Conference on Artificial Intelligence and Statistics (AI&STATS)*, JMLR W&CP 22, pages 601–609, 2012.

A. Khaleghi and D. Ryabko. Asymptotically consistent estimation of the number of change points in highly dependent time series. In *the Proceedings of the 29th International Conference on Machine Learning (ICML)*, JMLR W&CP, pages 539–547, Beijing, China, 2014.

J. Kleinberg. An impossibility theorem for clustering. In *Neural Information Processing Systems (NIPS)*, pages 446–453, Montreal, Canada, 2002.

I. Kontoyiannis and Y.M. Suhov. Prefixes and the entropy rate for long-range sources. In *IEEE International Symposium on Information Theory*, pages 194–194, 1994.

M. Kumar, N.R. Patel, and J. Woo. Clustering seasonality patterns in the presence of errors. In *Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD)*, pages 557–563. ACM, 2002.

E. Lehmann. *Testing Statistical Hypotheses, 2nd edition.* Wiley, New York, 1986.

L. Li and B.A. Prakash. Time series clustering: Complex is simpler! In *the Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 185–192, 2011.

M. Mahajan, P. Nimbhorkar, and K. Varadarajan. The planar k-means problem is NP-hard. In *the Proceedings of the 3rd International Workshop on Algorithms and Computation (WALCOM)*, pages 274–285, Berlin, Heidelberg, 2009.

G. Morvai and B. Weiss. On classifying processes. *Bernoulli*, 11(3):523–532, 2005.

MOCAP. CMU graphics lab motion capture database. Available at `http://mocap.cs.cmu.edu/`.

G. Morvai and B. Weiss. A note on prediction for discrete time series. *Kybernetika*, 48(4): 809–823, 2012.

D.S. Ornstein and B. Weiss. How sampling reveals a process. *Annals of Probability*, 18(3): 905–930, 1990.

E. Rio. *Théorie asymptotique des processus aléatoires faiblement dépendants*, volume 31. Springer, 1999.

B. Ryabko. Prediction of random sequences and universal coding. *Problems of Information Transmission*, 24:87–96, 1988.

B. Ryabko and J. Astola. Universal codes as a basis for time series testing. *Statistical Methodology*, 3:375–397, 2006.

B. Ryabko. Applications of universal source coding to statistical analysis of time series. *Selected Topics in Information and Coding Theory, World Scientific Publishing*, pages 289–338, 2010a.

D. Ryabko. Clustering processes. In *the Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 919–926, Haifa, Israel, 2010b.

D. Ryabko. Discrimination between B-processes is impossible. *Journal of Theoretical Probability*, 23(2):565–575, 2010c.

D. Ryabko. Testing composite hypotheses about discrete ergodic processes. *Test*, 21(2): 317–329, 2012.

D. Ryabko. Uniform hypothesis testing for finite-valued stationary processes. *Statistics*, 48 (1):121–128, 2014.

D. Ryabko and B. Ryabko. Nonparametric statistical inference for ergodic processes. *IEEE Transactions on Information Theory*, 56(3):1430–1435, 2010.

D. Ryabko and J. Mary. A binary-classification-based metric between time-series distributions and its use in statistical and learning problems. *Journal of Machine Learning Research*, 14:2837–2856, 2013.

P. Shields. *The Ergodic Theory of Discrete Sample Paths.* AMS Bookstore, 1996.

P. Smyth. Clustering sequences with hidden Markov models. In *Advances in Neural Information Processing Systems*, pages 648–654. MIT Press, 1997.

M. Tschannen and H. Bolcskei. Nonparametric nearest neighbor random process clustering. In *Proceedings of the 2015 IEEE International Symposium on Information Theory*, pages 1207–1211, Hong Kong, 2015. IEEE.

E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3):249–260, 1995.

S. Zhong and J. Ghosh. A unified framework for model-based clustering. *Journal of Machine Learning Research*, 4:1001–1037, 2003.

Journal of Machine Learning Research 17 (2016) 1-26 Submitted 2/14; Revised 3/15; Published 3/16

Random Rotation Ensembles

Rico Blaser R.Blaser@lse.ac.uk
Piotr Fryzlewicz P.Fryzlewicz@lse.ac.uk
Department of Statistics
London School of Economics
Houghton Street
London, WC2A 2AE, UK

Editor: Charles Elkan

Abstract

In machine learning, ensemble methods combine the predictions of multiple base learners to construct more accurate aggregate predictions. Established supervised learning algorithms inject randomness into the construction of the individual base learners in an effort to promote diversity within the resulting ensembles. An undesirable side effect of this approach is that it generally also reduces the accuracy of the base learners. In this paper, we introduce a method that is simple to implement yet general and effective in improving ensemble diversity with only modest impact on the accuracy of the individual base learners. By randomly rotating the feature space prior to inducing the base learners, we achieve favorable aggregate predictions on standard data sets compared to state of the art ensemble methods, most notably for tree-based ensembles, which are particularly sensitive to rotation.

Keywords: feature rotation, ensemble diversity, smooth decision boundary

1. Introduction

Modern statistical learning algorithms combine the predictions of multiple base learners to form ensembles, which typically achieve better aggregate predictive performance than the individual base learners (Rokach, 2010). This approach has proven to be effective in practice and some ensemble methods rank among the most accurate general-purpose supervised learning algorithms currently available. For example, a large-scale empirical study (Caruana and Niculescu-Mizil, 2006) of supervised learning algorithms found that decision tree ensembles consistently outperformed traditional single-predictor models on a representative set of binary classification tasks. Data mining competitions also frequently feature ensemble learning algorithms among the top ranked competitors (Abbott, 2012).

Achieving a good balance between the accuracy of the individual predictors and the diversity of the full ensemble is of critical importance: if the individual predictors are accurate but highly correlated, the benefits of combining them are modest; injecting randomness into the predictors reduces the correlation and promotes diversity but often does so at the expense of reduced accuracy for the individual predictors (Elghazel et al., 2011). A number of techniques have been devised to manage this trade-off and to promote diversity in learning ensembles in a constructive fashion; some methods merely perturb the training data, while others modify the internal structure of the predictors themselves. We now mention

some key examples. In bootstrap aggregation (Breiman, 1996), bootstrap replicates are used to construct multiple versions of a base predictor, which are subsequently aggregated via averaging or majority vote. This approach was found to be particularly effective for predictor classes that are unstable, in the sense that small variations of the input data lead to the construction of vastly different predictors (Hastie et al., 2009). Output smearing or flipping (Breiman, 2000) adds a different noise component to the dependent variable of each base predictor, which has a smoothing effect on the resulting decision boundary, leading to improved generalization performance. Boosting (Freund and Schapire, 1996) is an iterative procedure, where base learners are added sequentially in a forward stagewise fashion. By reweighting the data set at each iteration, later base learners are specialized to focus on the learning instances that proved the most challenging to the existing ensemble. In contrast to bootstrap aggregation, where each bootstrap sample is generated independently, boosting therefore does not lend itself naturally to parallel processing. Random decision forests (Ho, 1995, 1998) randomly select a feature subspace a priori and train a base learner in the chosen subspace using all available data. Instead of randomizing the training data, the structure of each predictor is altered by only including the chosen subset of predictors. Random forests (Breiman, 1999, 2001) combine bootstrap aggregation with the random projection method. At each tree node, a subset of the available predictors is randomly selected and the most favorable split point is found among these candidate predictors. This approach differs from random decision forests, where the selection of predictors is only performed once per tree. More generally, the framework also offers the possibility of using random linear combinations of two or more predictors. A summary of recent enhancements and applications of random forests can be found in Fawagreh et al. (2014). Perfect random tree ensembles (Cutler and Zhao, 2001), extremely random trees / extra trees (Geurts et al., 2006), and completely random decision trees (Liu et al., 2005; Fan et al., 2006) take randomization even further by not only selecting random predictor(s), as in random forests, but by also selecting a random split point, sometimes deterministically chosen from a small set of random candidate split points.

Some of the ensemble methods described specifically require the base learners to be decision trees. This is because decision trees are efficient to create (by recursive binary splitting), the models are straightforward to aggregate, and the individual trees can easily be turned into weak learners (which perform only slightly better than random) by restricting their depth (Kuhn and Johnson, 2013). Furthermore, decision trees exhibit a high variance and this inherent instability is beneficial to the diversity of the ensemble. In addition, decision trees contain a number of desirable features for general purpose data mining, including robustness to outliers and an ability to handle input variables of mixed type and scale, such as continuous and categorical variables, and even missing values (Hastie et al., 2009). However, a decision tree is merely an efficient representation for a set of hyper-rectangles that partition the decision space. For ordinary decision trees, each hyper-rectangle is aligned with at least one of the axes of the chosen coordinate system, resulting in axis parallel decision boundaries. This results in very characteristic piecewise constant stair shapes, even when the number of trees in the ensemble is large, as can be observed visually in low dimensional graphical examples. As a consequence, a much greater number of trees is needed to accurately approximate an oblique decision boundary than a decision boundary that is axis aligned with standard tree ensembles. In order to overcome this limitation, nonlinear

boosting projections (Garca-Pedrajas et al., 2007) provide a different, nonlinear view of the data to each base learner and oblique random forests (Menze et al., 2011) use linear discriminative models or ridge regression to select optimal oblique split directions at each tree node. Another approach that is related to but different from the method proposed in the present paper is embodied by rotation forests (Rodriguez et al., 2006; Kuncheva and Rodriguez, 2007), which take a subset of features and a bootstrap sample of the data and perform a principal component analysis (PCA), rotating the entire feature space before building the next base predictor. In addition to PCA, Kuncheva and Rodriguez (2007) experimented with nonparametric discriminate analysis (NDA) and sparse random projections and in De Bock and Poel (2011), independent component analysis (ICA) is found to yield the best performance.

The premise of the present paper is that it makes sense to rotate the feature space in ensemble learning, particularly for decision tree ensembles, but that it is neither necessary nor desirable to do so in a structured way. This is because structured rotations reduce diversity. Instead, we propose to rotate the feature space randomly before constructing the individual base learners. The random rotation effectively generates a unique coordinate system for each base learner, which we show increases diversity in the ensemble without a significant loss in accuracy. In addition to rotation, affine transformations also include translation, scaling, and shearing (non-uniform scaling combined with rotation). However, only transformations involving rotation have an impact on base learners that are insensitive to monotone transformations of the input variables, such as decision trees. Furthermore, a key difference between random rotation and random projection is that rotations are reversible, implying that there is no loss of information.

The remainder of this paper is structured as follows. Section 2 provides a motivational example for the use of random rotations using a well-known data set. In Section 3 we formally introduce random rotations and provide guidance as to their construction. Section 4 evaluates different application contexts for the technique and performs experiments to assesses its effectiveness. Conclusions and future research are discussed in Section 5. It is our premise that random rotations provide an intuitive, optional enhancement to a number of existing machine learning techniques. For this reason, we provide random rotation code in C/C++ and R in Appendix A, which can be used as a basis for enhancing existing software packages.

2. Motivation

Figure 1 motivates the use of random rotations on the binary classification problem from chapter 2 of Hastie et al. (2009). The goal is to learn the decision boundary, which separates the two classes, from a set of training points. In this example, the training data for each class came from a mixture of ten low-variance Gaussian distributions, with individual means themselves distributed as Gaussian. Since the data is artificially generated, the optimal decision boundary is known by construction.

In this motivational example, we compare two approaches: (1) a standard random forest classifier and (2) a random forest classifier in which each tree is generated on a randomly rotated feature space. It is evident that the random feature rotation has a significant impact on the resulting data partition: despite using the same sequence of random numbers

in the tree induction phase of the random forest algorithm – resulting in the same bootstrap samples and related feature subset selections at each decision branch for the two trees – the resulting tree is not merely a rotated version of the unrotated tree but is, in fact, a very different tree altogether, with a different orientation and a vastly different data partition. This demonstrates the power of the method; diversity is achieved with only a modest loss of information. However, the real benefit is illustrated on the bottom row of Figure 1 and arises from the aggregation of multiple randomly rotated trees. The rotated ensemble exhibits a visibly smoother decision boundary and one that is very close to optimal for this problem. The decision boundary is uncharacteristically smooth for a tree ensemble and is reminiscent of a kernel method, such as a k-nearest neighbor method or a support vector machine. In contrast, even with 10000 trees, the decision boundary for the standard random forest is still notably rectangular shaped. Another striking feature of the random rotation ensemble is the existence of a nearly straight diagonal piece of the decision boundary on the far left. This would be difficult to achieve with an axis-parallel base learner without rotation and it agrees well with the true decision boundary in this example.

3. Random Rotations

In this section, we formally introduce random rotations and describe two practical methods for their construction.

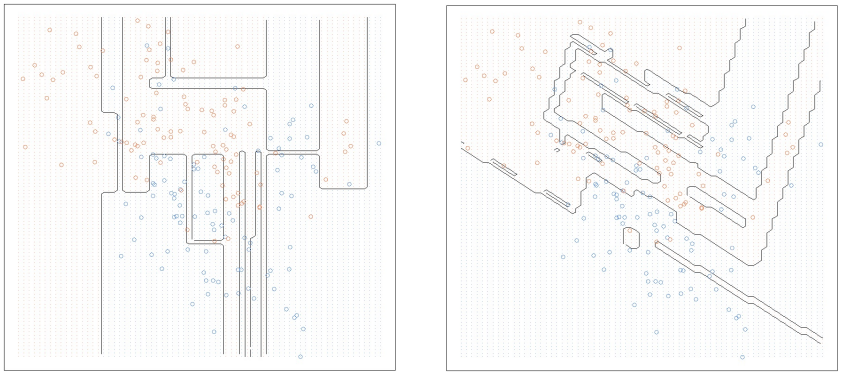
A (proper) rotation matrix R is a real-valued $n \times n$ orthogonal square matrix with unit determinant, that is

$$R^T = R^{-1} \text{ and } |R| = 1. \tag{1}$$

Using the notation from Diaconis and Shahshahani (1987), the set of all such matrices forms the special orthogonal group $SO(n)$, a subgroup of the orthogonal group $O(n)$ that also includes so-called improper rotations involving reflections (with determinant -1). More explicitly, matrices in $SO(n)$ have determinant $|R| = 1$, whereas matrices in $O(n)$ may have determinant $|R| = d$, with $d \in \{-1, 1\}$. Unless otherwise stated, the notation $O(n)$ always refers to the orthogonal group in this paper and is not related to the Bachman-Landau asymptotic notation found in complexity theory.

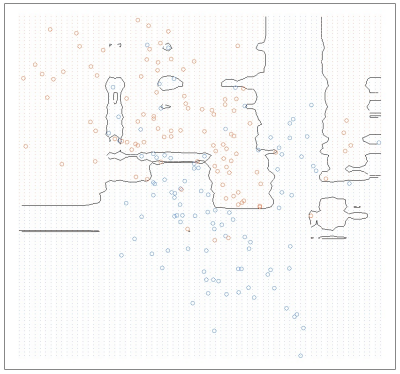
In order to perform a random rotation, we uniformly sample over all feasible rotations. Randomly rotating each angle in spherical coordinates does not lead to a uniform distribution across all rotations for $n > 2$, meaning that some rotations are more likely to be generated than others. It is easiest to see this is in 3 dimensions: suppose we take a unit sphere, denoting the longitude and latitude by the two angles $\lambda \in [\text{-}\pi, \pi]$ and $\phi \in [\text{-}\pi/2, \pi/2]$. If we divide the surface of this sphere into regions by dividing the two angles into equal sized intervals, then the regions closer to the equator ($\phi = 0$) are larger than the regions close to the poles ($\phi = \pm\pi/2$). By selecting random angles, we are equally likely to arrive in each region but due to the different sizes of these regions, points tend to cluster together at the poles. This is illustrated for $n = 3$ in Figure 2, where the undesirable concentration of rotation points near the two poles is clearly visible for the naive method. In this illustration, the spheres are tilted to better visualize the areas near the poles.

The group $O(n)$ does have a natural uniform distribution called the Haar measure, which offers the distribution we need. Using the probabilistic notation from Diaconis and Shahshahani (1987), the random matrix R is said to be uniformly distributed if $P(R \in U) =$

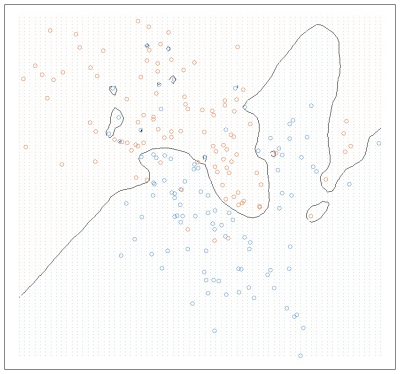


(a) **RF** (ntree=1, mtry=1)

(b) **RR-RF** (ntree=1, mtry=1)

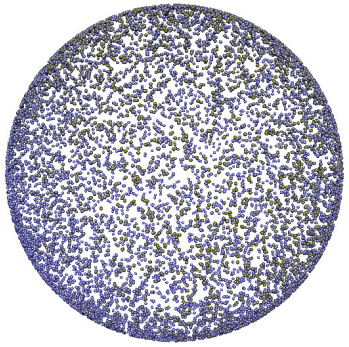


(c) **RF** (ntree=10000, mtry=1)

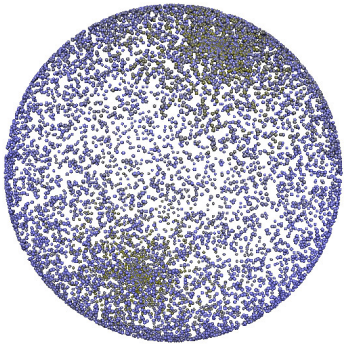


(d) **RR-RF** (ntree=10000, mtry=1)

Figure 1: Comparison of the decision boundary for the standard random forest algorithm (RF, left column) and the modified version with randomly rotated feature space for each tree (RR-RF, right column) on the binary classification task of chapter 2 of Hastie et al. (2009). The top row illustrates a typical decision boundary for a single tree, while the bottom row depicts a fully grown ensemble comprised of 10000 trees in each case. Ntree is the total number of trees in the forest, mtry the number of randomly selected features considered at each decision node.



(a) Uniformly random rotations



(b) Naive random rotations

Figure 2: Comparison of correctly executed uniformly random rotation (left) in three dimensions versus naive method of selecting two random angles in spherical coordinates (right). 10000 random rotations of the same starting vector were generated for each method and distances were computed between each pair of rotations to produce a rank-based gradient, with green dots representing those vectors with the lowest sums of distances.

$P(R \in \Gamma U)$ for every $U \subset O(n)$ and $\Gamma \in O(n)$. Several algorithms exist to generate random orthogonal matrices distributed according to the Haar measure over $O(n)$, some of which are documented in Anderson et al. (1987); Diaconis and Shahshahani (1987); Mezzadri (2007); Ledermann and Alexander (2011). We will focus on two basic approaches to illustrate the concept.

1. (*Indirect Method*) Starting with an $n \times n$ square matrix A, consisting of n^2 independent univariate standard normal random variates, a Householder QR decomposition (Householder, 1958) is applied to obtain a factorization of the form $A = QR$, with orthogonal matrix Q and upper triangular matrix R with positive diagonal elements. The resulting matrix Q is orthogonal by construction and can be shown to be uniformly distributed. In other words, it necessarily belongs to $O(n)$. Unfortunately, if Q does not feature a positive determinant then it is not a proper rotation matrix according to definition (1) above and hence does not belong to $SO(n)$. However, if this is the case then we can flip the sign on one of the (random) column vectors of A to obtain A^+ and then repeat the Householder decomposition. The resulting matrix Q^+ is identical to the one obtained earlier but with a change in sign in the corresponding column and $|Q^+| = 1$, as required for a proper rotation matrix.

2. (*Direct Method*) A second method of obtaining random rotations involves selecting random points on the unit n-sphere directly (Knuth, 1997). This can be accom-

plished by drawing n independent random normal $N(0,1)$ variates $\{v_1, v_2, ..., v_n\}$ and normalizing each by the square root of the sum of squares of all n variates, that is, $x_i = v_i/\sqrt{v_1^2 + v_2^2 + \ldots + v_n^2}$ for $i \in \{1, 2, \ldots, n\}$. In other words, x is a unit vector pointing to a random point on the n-sphere. This construction takes advantage of spherical symmetry in the multivariate Normal distribution. The method is asymptotically faster than the QR approach and an implementation is available in the GNU Scientific Library (Galassi, 2009) as function gsl_ran_dir_nd. It should also be noted that it is straightforward to obtain the individual rotation angles from the random vector x, which makes it possible to move from the (more compact) random vector notation to the random rotation matrix used in the indirect method.

Generating random rotations in software for problems involving fewer than 1000 dimensions is straightforward and fast, even using the simple algorithm described above. Listings 2 and 3 in Appendix A provide examples of the indirect method in C++ and R respectively, both presented without error checking or optimizations. The C++ code takes less than 0.5 seconds on a single core of an Intel Xeon E5-2690 CPU to generate a 1000x1000 random rotation matrix. It uses the Eigen template library (Guennebaud et al., 2010) and a Mersenne Twister (Matsumoto and Nishimura, 1998) pseudorandom number generator. Larger rotation matrices can be computed with GPU assistance (Kerr et al., 2009) and may be pre-computed for use in multiple applications. In addition, for problems exceeding 1000 dimensions it is practical to only rotate a random subset of axes in order to reduce the computational overhead. We recommend that a different random subset is selected for each rotation in this case.

For categorical variables, rotation is unnecessary and ill defined. Intuitively, if a category is simply mapped to a new rotated category, there is no benefit in performing such a rotation.

4. Experiments

Random rotations complement standard learning techniques and are easily incorporated into existing algorithms. In order to examine the benefit of random rotations to ensemble performance, we modified three standard tree ensemble algorithms to incorporate random rotations before the tree induction phase. The necessary modifications are illustrated in pseudo code in Listing 1 below.

All methods tested use classification or regression trees that divide the predictor space into disjoint regions G_j, where $1 \leq j \leq J$, with J denoting the total number of terminal nodes of the tree. Extending the notation in Hastie et al. (2009), we represent a tree as

$$T(x; \theta, \Omega) = \sum_{j=1}^{J} c_j I(R(x) \in G_j), \tag{2}$$

with optimization parameters $\Omega = \{G_j, c_j\}_1^J$, random parameters $\theta = \{R, \omega\}$, where R is the random rotation associated with the tree and ω represents the random sample of (x, y) pairs used for tree induction; $I(\cdot)$ is an indicator function. Each randomly rotated input $R(x)$ is thus mapped to a constant c_j, depending on which region G_j the input belongs to.

For regression, c_j is typically just the average or median of all y_j in region G_j. If we let $|G_j|$ denote the cardinality of G_j, this can be written as

$$c_j = \frac{1}{|G_j|} \sum_{R(x_k) \in G_j} y_k. \tag{3}$$

For classification trees, one of the modes is typically used instead.

Given a loss function $L(y_i, f(x_i))$, for example exponential loss for classification or squared loss for regression, a tree-induction algorithm attempts to approximate the optimization parameters for which the overall loss is minimized, that is

$$\hat{\Omega} = \arg\min_{\Omega} \sum_{j=1}^{J} \sum_{R(x_i) \in G_j} L(y_i, f(R(x_i))) = \arg\min_{\Omega} \sum_{j=1}^{J} \sum_{R(x_i) \in G_j} L(y_i, c_i). \tag{4}$$

This optimization is performed across all parameters Ω but the rotation is explicitly excluded from the search space ($R \in \theta$, but $R \notin \Omega$) because we are advocating a random rotation in this paper. However, conceptually it would be possible to include the rotation in the optimization in an attempt to focus on the most helpful rotations.

Listing 1: Testing Random Rotations (Pseudo Code)

```
Inputs: - training feature matrix X
        - testing feature matrix S
        - total number of classifiers M
        - standard base learner B
        - aggregation weights w

(A) Scale or rank numeric predictors x (Sect. 4.2):
    e.g. x' := (x - Q_k(x))/(Q_{1-k}(x) - Q_k(x))

(B) For m ∈ {1,2,...,M} do
  (1) generate random parameters: θ_m := {R_m, ω_m}
  (2) train standard learner B on R_m(x):
        Ω̂_m = arg min_Ω Σ_{j=1}^{J} Σ_{R(x_i)∈G_j} L(y_i, f(R_m(x_i)))
  (3) compute test or out-of-bag predictions
        T(x, θ_m, Ω_m), x ∈ R_m(S)

(C) Aggregate predictions (vote or average)
      f_M(x) = Σ_{m=1}^{M} w_m T(x; θ_m, Ω_m)
```

In all algorithms considered in this paper, the tree-induction is performed using standard greedy, top-down recursive binary partitioning. This approach will generally not arrive at the globally optimal solution to (4) but constructs a reasonable approximation quickly (Hastie et al., 2009).

The (regression) tree ensemble $f_M(x)$ can then be written as a weighted sum of the individual trees, that is

$$f_M(x) = \sum_{m=1}^{M} w_m T(x; \theta_m, \Omega_m), \tag{5}$$

where M denotes the total number of trees in the ensemble. For classification ensembles, a vote is typically taken instead. It should be noted that a separate rotation is associated with each tree in this notation but the same rotation could theoretically be associated with an entire group of trees. In particular, we can recover the standard setting without random rotation by setting R_m to the identity rotation for all m.

The difference between non-additive ensemble methods like random forests (Breiman, 2001) or extra trees (Geurts et al., 2006) and additive ensembles like boosted trees (Freund and Schapire, 1996) arises in the formulation of the joint model for multiple trees. As we will see, this difference makes testing random rotation with existing additive ensemble libraries much more difficult than with non-additive ensemble libraries. Specifically, random forests and extra trees place an equal weight of $w_m = 1/M$ on each tree, and trees are constructed independently of each other, effectively producing an average of M independent predictions:

$$\hat{\Omega}_m = \arg\min_{\Omega_m} \sum_{j=1}^{J} \sum_{R(x_i) \in G_j} L(y_i, T(x_i; \theta_m, \Omega_m)). \tag{6}$$

In contrast, boosted trees use $w_m = 1$ and each new tree in the sequence is constructed to reduce the residual error of the full existing ensemble $f_{m-1}(x)$, that is

$$\hat{\Omega}_m = \arg\min_{\Omega_m} \sum_{j=1}^{J} \sum_{R(x_i) \in G_j} L(y_i, f_{m-1}(x_i) + T(x_i; \theta_m, \Omega_m)). \tag{7}$$

There are other differences between the two approaches: for example, J, the number of leaf nodes in each tree is often kept small for boosting methods in order to explicitly construct weak learners, while non-additive methods tend to use large, unpruned trees in an effort to reduce bias, since future trees are not able to assist in bias reduction in this case.

We mainly focus on random forest and extra tree ensembles in this paper because both of these algorithms rely on trees that are constructed independently of each other. This provides the advantage that the original tree induction algorithm can be utilized unmodified as a black box in the rotated ensemble, ensuring that any performance differences are purely due to the proposed random rotation and are not the result of any subtle differences (or dependencies) in the construction of the underlying trees.

4.1 Data Sets & Preprocessing

For our comparative study of random rotation, we selected UCI data sets (Bache and Lichman, 2013) that are commonly used in the machine learning literature in order to make the results easier to interpret and compare. Table 5 in Appendix C summarizes the data sets, including relevant dimensional information.

Some algorithms tested were not able to handle categorical input variables or missing values and we performed the following automatic preprocessing steps for each data column:

1. Any column (predictors or response) with at least 10 distinct numeric values was treated as numeric and missing values were imputed using the column median.

2. Any column with fewer than 10 distinct values (numeric or otherwise) or with mostly non-numeric values was treated as categorical, and a separate category was explicitly created for missing values.

3. Categorical predictors with C categories were converted into $(C-1)$ 0/1 dummy variables, with the final dummy variable implied from the others to avoid adding multicollinearity.

Note that after evaluating these three rules, all predictors were either numeric without missing values or categorical dummy variables, with a separate category for missing values.

4.2 Variable Scaling

Rotation can be sensitive to scale in general and outliers in particular. In order to avoid biasing the results, we tested three different scaling methods, all of which only use in-sample information to calibrate the necessary parameters for out-of-sample scaling:

1. (*Basic Scaling*) Numeric values were scaled to $[0, 1]$ using the in-sample min and max values, that is $x' = \min(1, \max(0, (x - \min(x_{is}))/(\max(x_{is}) - \min(x_{is}))))$. This scaling method deals with scale but only avoids out-of-sample outliers. Outliers are dealt with in a relatively crude fashion by applying a fixed cutoff.

2. (*Quantile Scaling*) Numeric values were linearly scaled in such a way that the 5th and 95th percentile of the in-sample data map to 0 and 1 respectively, that is $x' = (x - Q_5(x_{is}))/(Q_{95}(x_{is}) - Q_5(x_{is}))$. In addition, any values that exceed these thresholds were nonlinearly winsorized by adding/subtracting $0.01 \times \log(1 + \log(1 + \Delta))$, where Δ is the absolute difference to the in-sample bounds $Q_5(x_{is})$ or $Q_{95}(x_{is})$. This robust scaling has a breakdown point of 5% and maintains the order of inputs that exceed the thresholds.

3. (*Relative Ranking*) In-sample numeric values v_i were augmented with $\{-\infty, +\infty\}$ and ranked as $R(v_i)$, such that $R(-\infty)$ maps to 0 and $R(+\infty)$ maps to 1. Out-of-sample data was ranked relative to this in-sample map. To accomplish this, the largest v_i is found that is smaller or equal to the out-of-sample data v_o (call it v_i^{max}) and the smallest v_i is found that is greater or equal to the out-of-sample data v_o (v_i^{min}). The out-of-sample rank is then $0.5 \times R(v_i^{min}) + 0.5 \times R(v_i^{max})$. In this way, test elements that match in-sample elements obtain the same rank, test elements that fall in between two elements obtain a rank in between, and because the in-sample values are augmented with infinity, it is never possible to encounter test elements that cannot be mapped. This is the most robust approach to outliers that was tested.

4.3 Method & Evaluation

In order to collect quantitative evidence of the effect of random rotations, we built upon the tree induction algorithms implemented in the widely used R packages *randomForest* (Liaw and Wiener, 2002) and *extraTrees* (Simm and de Abril, 2013). For comparison, we also used the following implementation of rotation forests: https://github.com/ajverster/RotationForest.

Prior to the tests, all data sets were preprocessed and scaled using each of the three techniques described in the previous section (basic scaling, quantile scaling, and ranking).

Random forest and extra trees were tested with and without random rotation (for each scaling), while rotation forests included their own deterministic PCA rotation (Rodriguez et al., 2006) but were also run for each scaling method. Random rotations were tested with and without flip rotations. The combination of tree induction algorithms, scalings, and rotation options resulted in a total of 21 distinct experiments per data set.

For each experiment we performed a random 70-30 split of the data; 70% training data and the remaining 30% served as testing data. The split was performed uniformly at random but enforcing the constraint that at least one observation of each category level had to be present in the training data for categorical variables. This constraint was necessary to avoid situations, where the testing data contained category levels that were absent in the training set. Experiments were repeated 100 times (with different random splits) and the average performance was recorded.

In all cases we used default parameters for the tree induction algorithms, except that we built 5000 trees for each ensemble in the hope of achieving full convergence.

To evaluate the performance of random rotations, we ranked each method for each data set and computed the average rank across all data sets. This allowed us to compare performance of each method across scaling methods and tree induction algorithms in a consistent, nonparametric fashion. In addition, we determined the number of data sets for which each method performed within one cross-sectional standard deviation of the best predictor in order to obtain a measurement of significance. This approach is advocated in (Kuhn and Johnson, 2013) and it can be more informative when there is a cluster of strong predictors that is distinct from the weaker predictors and which would not get detected by simple ranking.

4.4 Results

Table 6 in Appendix C displays the raw results of the detailed testing. As indicated in the previous section, we opted to compare ranks – with low ranks indicating better performance – in order to avoid the problem of comparing problems of different difficulty or comparing regression with classification problems. This is illustrated in Table 1.

	B-SCALE					Q-SCALE					RANKED				
NAME	**rf**	**rrrf**	**et**	**rret**	**rot**	**rf**	**rrrf**	**et**	**rret**	**rot**	**rf**	**rrrf**	**et**	**rret**	**rot**
anneal	11	7	1	**6**	14	10	11	**1**	**1**	9	8	11	**5**	**1**	15
audiology	**10**	**8**	1	**6**	13	**9**	**7**	**5**	1	14	12	**10**	**3**	**3**	15
balance	6	8	14	14	**1**	7	4	13	10	**2**	8	5	10	12	**2**
breast-w	**4**	1	9	4	14	**2**	**4**	9	8	15	**7**	**3**	9	9	13
breast-y	**1**	**6**	9	10	15	**1**	**4**	10	8	12	**5**	**1**	7	12	14
chess	10	7	**5**	**2**	13	9	11	**4**	**3**	15	12	8	**1**	**6**	13
cleveland	**6**	**2**	10	**5**	14	**8**	**4**	9	**1**	15	10	**7**	10	**2**	13
credit-a	**3**	9	13	15	12	**1**	**5**	11	14	8	**1**	**6**	**7**	10	**4**
flare	**1**	**1**	10	11	**1**	**1**	**1**	13	13	**1**	**1**	**1**	11	15	**1**
glass	**1**	12	**4**	11	13	**3**	9	**6**	7	14	**2**	8	**5**	9	15
hayes-ro	10	**7**	**1**	**1**	14	9	10	**1**	**1**	13	**7**	10	**1**	**1**	14
hepatitis	**5**	**2**	14	15	7	**1**	**4**	10	10	8	**6**	**3**	13	10	9
horse-c	**7**	**3**	13	10	**1**	**6**	**5**	13	11	**1**	**8**	**9**	15	12	4
ionosph	8	**1**	5	**3**	13	10	**2**	6	**4**	14	11	9	7	12	15
iris	14	7	10	9	**2**	15	7	10	6	**1**	13	3	12	4	5
led24	**2**	**6**	9	8	15	**1**	**3**	7	10	13	**5**	**4**	12	11	14
liver	**7**	**10**	14	15	**6**	**2**	**4**	13	12	**5**	**3**	**1**	11	**9**	**8**
lymph	15	5	10	8	**1**	14	5	12	4	**3**	13	8	11	5	**2**
nursery	8	11	**1**	**1**	15	9	9	**4**	**5**	14	12	7	**3**	**6**	13
pima	**8**	**1**	15	14	**3**	**7**	**2**	10	11	**4**	**6**	**9**	12	13	**5**
segment	**4**	10	**2**	9	14	**6**	12	**3**	11	13	**4**	**8**	**1**	**6**	15
solar	**4**	**8**	10	12	**1**	**7**	**5**	12	12	**3**	**9**	**5**	11	12	**2**
sonar	10	**7**	**1**	**5**	13	9	12	**2**	**6**	14	10	**3**	**3**	8	15
soybean	**12**	**7**	**6**	**5**	14	**11**	**7**	**1**	**1**	13	**9**	**10**	**3**	**3**	15
threeOf9	**9**	**7**	**1**	**3**	14	**10**	**12**	**3**	**1**	13	**7**	**11**	**3**	**3**	15
tic-tac-toe	**9**	**8**	**1**	**3**	15	**12**	**7**	**4**	**1**	13	**11**	**9**	**6**	**4**	14
votes	**8**	**5**	**1**	**12**	13	**1**	**8**	**10**	**10**	14	**5**	**5**	**1**	**1**	14
waveform	11	**1**	**7**	**2**	13	12	**3**	**8**	**5**	15	10	**4**	**9**	**6**	14
wine	**5**	**10**	**1**	**8**	14	**5**	**11**	**3**	**8**	15	**4**	**12**	**1**	**5**	13

Table 1: Ranked performance comparison between random forest with and without random rotation (rf, rrrf), extra trees with and without random rotation (et, rret), and rotation trees (rot). For all data sets (left-hand side), the comparison is performed over the three scaling methods described in the text as basic scaling (b-scale), quantile scaling (q-scale), and ranking (ranked). The values represent ranks of the classification errors for classification problems and the ranks of the RMSE for regression problems. Values that are within one cross-sectional standard deviation of the minimum error are in bold.

In this table, we have omitted flip rotations because their performance was comparable to the simple random rotation, with flip rotations outperforming in 36% of cases, simple rotations outperforming in 45% of cases and ties in the remaining 19% of cases.

The best overall average rank of 6.10 (of 15) was achieved by the random rotation random forest algorithm with simple scaling, followed by the same algorithm with complex scaling (6.48) and ranking (6.55). This algorithm outperformed regardless of scaling. The next lowest rank of 6.72 was achieved by random rotation extra trees using quantile scaling.

It is interesting to note that the average ranks for each scaling type were 7.50 for the complex scaling, 7.65 for the simple scaling and 7.81 for ranking. This indicates that the scaling method was less influential than the selection of the algorithm. In particular, the new method often improved on the original method even when only the ranks of the data were considered. We believe this to be an interesting result because it indicates that even rotating ranks can improve performance. Obviously, ranked data is completely robust to scaling effects and outliers.

The best average rank across scalings was achieved by random rotation random forests with 6.38, followed by extra trees (7.06), random forests (7.20), random rotation extra trees (7.26), and rotation forests (10.38).

In our tests, rotation forests underperformed overall but showed strong performance in some particular cases. The problem here was that when rotation forests did not excel at a problem, they often were the worst performer by a large margin, which had an impact on the average rank. In contrast, random rotation random forests rarely displayed the very best performance but often were among the top predictors. This insight led us to consider predictors that were within one cross-sectional standard deviation of the best predictor for each data set.

Random rotation random forests were within one standard deviation of the best result (highlighted in bold in Table 1) in 67.8% of cases, random forests without rotation in 64.3% of cases, extra trees (with and without rotation) in 49.4% of cases, and rotation forests in 27.6%. It appears to be clear that random rotation can improve performance for a variety of problems and should be included as a user option for standard machine learning packages.

Random rotation appears to work best when numerical predictors outnumber categorical predictors, which are not rotated, and when these numerical predictors exhibit a relatively smooth distribution (rather than a few pronounced clusters). An example of a suitable dataset is Cleveland, with more than half of the variables continuous and spread out evenly. In contrast, Balance is an example of a dataset for which we cannot expect random rotation to perform well. However, in general it is difficult to judge the utility of rotation in advance and we recommend running a small test version of the problem with and without rotation to decide which to use: when the approach is successful, this tends to be apparent early.

Constructing a random rotation matrix using the indirect method described above requires of the order of p^3 operations, where p is the number of predictors to be rotated (time complexity of QR factorization). Multiplying the resulting random rotation matrix with an input vector requires of the order of p^2 operations. During training, this step needs to be performed k times, where k is the number of instances in the training data, for a total of $k \times p^2$ operations. All but one of the UCI datasets contained fewer than 100 predictors, and it takes less than a millisecond to compute a 100x100 random rotation matrix. Hence, with 5000 trees in each ensemble, the additional computational overhead was at most a few

seconds. However, as indicated above, for larger problems it does make sense to restrict the number of rotated predictors to maintain adequate performance.

Next, we consider the question of robustness.

4.5 Parameter Sensitivity and Extensions

In addition to the full tests described above, we also ran a few smaller examples to examine the sensitivity of random rotation to the choice of parameters in the underlying base learners. For this, we used the well known UCI *iris* data set (4 numeric predictors, 3 classes, 150 rows). Table 2 compares the performance of the standard random forest algorithm (RF) and a modified version including random feature rotation (RR-RF) on this data set.

Random Forest Comparison (iris)

parameters		% error		% wins per method		
ntree	mtry	RF	RR-RF	RF	RR-RF	Ties
50	1	5.269	**4.464**	16.98	**53.40**	29.62
	2	5.011	**4.237**	15.80	**50.20**	34.00
	3	4.960	**4.155**	16.14	**51.10**	32.76
	4	4.963	**4.077**	15.30	**52.75**	31.95
500	1	5.246	**4.414**	11.76	**52.98**	35.26
	2	4.981	**4.226**	13.53	**48.41**	38.06
	3	4.904	**4.144**	14.90	**49.22**	35.88
	4	4.944	**4.096**	13.80	**51.53**	34.67
5000	1	5.227	**4.385**	10.29	**52.52**	37.19
	2	4.975	**4.196**	13.48	**49.57**	36.95
	3	4.860	**4.133**	15.23	**47.76**	37.01
	4	4.964	**4.132**	14.22	**51.14**	34.64

Table 2: Performance comparison of the standard random forest algorithm (RF) and a modified version with randomly rotated feature space for each tree (RR-RF) on the *iris* data set. Ntree is the total number of trees in the forest, mtry the number of randomly selected features considered at each decision node. Statistically significant differences in mean error percentage and win percentage at the 1% level are denoted in bold.

As in the detailed tests, both classifiers made use of the same tree induction algorithm, implemented in the *randomForest* R package, but the feature space was randomly rotated prior to the construction of each tree for the RR-RF algorithm. Since the iris data set only includes 4 predictors, the number of randomly selected features at each decision node (mtry) only has feasible values in 1-4, allowing for an exhaustive comparison. For each parameter setting, we selected 50% of the data (75 cases) at random as the learning data set, while the other half served as the test data set. The experiment was repeated 10000 times for each parameter setting and we kept track of the average error percentage of each method, as well as the percentage of times each method outperformed the other. Once

completed, a Wilcoxon signed-rank test was performed to compare the classification error percentage and win percentage of the original method with that of the modified classifier and to ascertain statistical significance at the 1% level. The modified ensemble featuring random rotation appears to universally outperform the original classifiers on this data set, regardless of parameter settings and in a statistically significant manner. However, it should be noted that the goal of this experiment was not to demonstrate the general usefulness of random rotations – this is achieved by the detailed experiments in the previous section – but rather to show the robustness to parameter changes for a specific data set.

Table 3 shows the analogous results for extra trees, which select both the split feature and the split point at random. Here we used the tree induction algorithm implemented in the *extraTrees* R package. In theory, there exist an infinite number of feasible split points (ncut) that could be chosen but for simplicity, we have only attempted ncut values in the range 1-4, meaning that at most 4 random split points were considered in the tests. The improvement due to rotation is again universal and statistically significant. For reasonable parameters (e.g. ntree ≥ 50), the new method matches or outperforms the original method in over 94% of the randomly generated cases and the performance improvement is 21.7% on average. This is again a very encouraging result, as it demonstrates that the results above are robust, even if non-default parameters are used for the base learners. It is also interesting to note that randomly rotated extra tree ensembles outperform randomly rotated random forests here and they tend to do best with lower ncut values, indicating that more randomness (via rotation, feature selection, and split selection) is helpful for this particular problem.

Table 4 shows comparable results with a gradient boosting machine from the *gbm* R package (Ridgeway, 2013). Since boosting is an additive procedure, where later trees have an explicit dependence on earlier trees in the ensemble, the comparison of the two methods is not as straightforward. More specifically, step (B).(2) in Listing 1 cannot be performed without knowing (and being able to reuse) f_{m-1} in the case of boosting. Unfortunately, the most common software packages for boosting (and *gbm* in particular) do not provide an interface for this. Of course, we could have implemented our own boosting library but then it would not be obvious that the improvement in predictive performance was entirely due to the rotation. For this reason, we opted to demonstrate boosting with a widely used package but on groups of trees, with one rotation per group of 50 trees. The rationale for this choice of group size was that the first few boosting iterations often lead to rapid improvements. In this case, we compared the original method, consisting of 5000 trees in a single ensemble, to a modified version with 100 sub-forests of 50 trees each, whereby each sub-forest was created on a randomly rotated feature space. In other words, the 50 trees in each sub-forest had a dependency, whereas the sub-forests themselves were independent of each other. The final classification was achieved through voting. As is evident from Table 4, randomly rotated gradient boosting machines even outperformed random forests and extra trees on this data set in terms of percentage error. Even when we handicapped the new method by only providing it with relative ranks of the data it outperformed the original (unrotated) method, although not by the same margin.

Extra Tree Ensemble Comparison (iris)

parameters		% error		% wins per method		
ntree	ncut	ET	RR-ET	ET	RR-ET	Ties
50	1	5.335	**3.994**	7.18	**66.11**	26.71
	2	5.281	**4.101**	7.84	**63.04**	29.12
	3	5.183	**4.078**	8.41	**60.69**	30.90
	4	5.238	**4.152**	8.86	**60.14**	31.00
500	1	5.244	**3.971**	4.09	**66.02**	29.89
	2	5.157	**4.045**	4.75	**60.85**	34.40
	3	5.118	**4.056**	5.16	**59.67**	35.17
	4	5.114	**4.111**	5.54	**57.68**	36.78
5000	1	5.257	**4.044**	4.73	**66.09**	29.18
	2	5.175	**4.003**	3.32	**60.86**	35.82
	3	5.038	**4.079**	4.71	**56.20**	39.09
	4	5.046	**4.053**	5.55	**56.92**	37.53

Table 3: Performance comparison of the standard extra trees algorithm (ET) and a modified version with randomly rotated feature space for each tree (RR-ET) on the *iris* data set. Ntree is the total number of trees in the ensemble, ncut the number of randomly selected split points considered at each decision node. Statistically significant differences in mean error percentage and win percentage at the 1% level are denoted in bold.

Gradient Boosting Comparison (iris)

parameters				performance	
rank only	type	ntree	shrinkage	%error	%wins
no	GBM	1x5000	0.0005	5.063	9.62
no	RR-GBM	100x50	0.0500	**3.831**	**57.26**
yes	GBM	1x5000	0.0005	5.063	22.97
yes	RR-GBM	100x50	0.0500	**4.385**	**46.17**

Table 4: Performance comparison of the standard gradient boosting machine (GBM) and a modified version with randomly rotated feature space for each sub-forest of 50 trees (RR-GBM) on the *iris* data set. A classifier was trained for each parameter setting on a random half of the data and tested on the remaining half. Ntree is the total number of trees in the ensemble, expressed as the product of the number of generated sub-forests (each on a randomly rotated feature space) times the number of trees in each sub-forest. The procedure was repeated 10000 times. Statistically significant differences in mean error percentage and win percentage at the 1% level are denoted in bold. For the robust rank only version, a ranking of each predictive variable was performed using the train data, while the test vectors received an interpolated ranking based solely on relative order information with respect to the train data.

4.6 A Note on Diversity

In Appendix B we closely follow Breiman (2001) to derive an ensemble diversity measure that is applicable to the case of random rotation ensembles. In particular, we show that just like for random forests we can express the average correlation of the raw margin functions across all classifiers in the ensemble in terms of quantities we can easily estimate, specifically

$$\bar{\rho}(\cdot) = \frac{V_{x,y}[\Psi(x,y)]}{E_{\theta_1,\theta_2}[\sigma(\psi(x,y,\theta))]^2}. \tag{8}$$

That is, average correlation $\bar{\rho}$ is the variance of the margin across instances $V_{x,y}[\Psi(x,y)]$, divided by the expectation of the standard deviation σ of the raw margin across (randomized) classifiers squared. Full definitions of these quantities can be found in Appendix B.

As an example of the usefulness of this correlation measure, we estimated $\bar{\rho}$ with $m_{try} = \{1, 4\}$ on the iris example and achieved a correlation of 0.32 and 0.61, respectively. Clearly, the random split selection decorrelates the base learners. We then performed the same calculation including random rotation and achieved 0.22 and 0.39, respectively. In both cases, the correlation decreased by approximately one third. In contrast, the expected margin only decreased by 3.5%, meaning that the accuracy of the individual base learners was only very modestly affected.

5. Conclusion

Random rotations provide a natural way to enhance the diversity of an ensemble with minimal or no impact on the performance of the individual base learners. Rotations are particularly effective for base learners that exhibit axis parallel decision boundaries, as is the case for all of the most common tree-based learning algorithms. The application of random rotation is most effective for continuous variables and is equally applicable to higher dimensional problems.

A generalization of random rotations only uses a subset of rotations for out of sample predictions. This subset is chosen by observing the out-of-bag performance of each rotation in sample. Initial tests revealed that dropping the least effective decile of all random rotations generally improved out of sample performance but more research is needed because the procedure potentially introduces model bias.

Random rotations may also prove to be useful for image analysis. For example, axis-aligned methods for image processing, such as wavelet smoothing, may benefit from repeated random rotations to ensure that the methods become axis-independent.

While random rotations are certainly not a panacea, they are helpful frequently enough that we contend standard data mining packages should provide users the option to randomly rotate the feature space prior to inducing each base learner.

Appendix A

The following listings provide illustrations in two commonly used programming languages for the generation of a random rotation matrix using the indirect method described in section 3 above. The code is kept simple for illustrative purposes and does not contain error checking or performance optimizations.

Listing 2: Random Rotation in C++ using Eigen

```
#include "MersenneTwister.h"
#include <Eigen/Dense>
#include <Eigen/QR>

using namespace Eigen;

// C++: generate random n x n rotation matrix
void random_rotation_matrix(MatrixXd& M, int n)
{
  MTRand mtrand; // twister with random seed

  MatrixXd A(n,n);
  const VectorXd ones(VectorXd::Ones(n));

  for(int i=0; i<n; ++i)
    for(int j=0; j<n; ++j)
      A(i,j) = mtrand.randNorm(0,1);

  const HouseholderQR<MatrixXd> qr(A);
  const MatrixXd Q = qr.householderQ();
  M = Q * (qr.matrixQR().diagonal().array()
           < 0).select(-ones,ones).asDiagonal();

  if(M.determinant() < 0)
    for(int i=0; i<n; ++i)
      M(i,0) = -M(i,0);
}
```

Listing 3: Random Rotation in R

```
# generate random member of orthogonal group O(n)
random_rotation_matrix_incl_flip <- function(n)
{
  QR <- qr(matrix(rnorm(n^2), ncol=n))          # A = QR
  M <- qr.Q(QR) %*% diag(sign(diag(qr.R(QR)))) # diag(R) > 0
  return(M)
}

# generate random member of special orthogonal group SO(n)
random_rotation_matrix <- function(n)
{
  M <- random_rotation_matrix_incl_flip(n)
  if(det(M)<0) M[,1] <- -M[,1]                  # det(M) = +1
  return(M)
}
```

Appendix B

In this appendix, we closely follow Breiman (2001) to derive an ensemble diversity measure that is applicable to random rotation ensembles.

For a given input vector x, we define the label of the class to which classifier $f_M(x)$ assigns the highest probability, save for the correct label y, to be j^M_{max}, that is

$$j^M_{max} := \arg\max_{j \neq y} P(f_M(x) = j). \tag{9}$$

Using this definition, we denote the raw margin function $\psi(x, y)$ for a classification tree ensemble $f_M(x)$ as

$$\psi(x, y, \theta) = I(f_M(x) = y) - I(f_M(x) = j^M_{max}), \tag{10}$$

with indicator function $I(\cdot)$. This expression evaluates to +1 if the classification is correct, -1 if the most probable incorrect class is selected, and 0 otherwise. The margin function $\Psi(x, y)$ is its expectation, that is

$$\begin{aligned} \Psi(x, y) &= E_\theta[\psi(x, y, \theta)] \\ &= P(f_M(x) = y) - P(f_M(x) = j^M_{max}). \end{aligned} \tag{11}$$

The margin function $\Psi(x, y)$ represents the probability of classifying an input x correctly minus the probability of selecting the most probable incorrect class.

If we denote the out-of-bag instances for classification tree $T(x; \theta_m, \Omega_m)$ as O_m, then these probabilities can be estimated as

$$\hat{P}(f_M(x) = k) = \frac{\sum^M_{m=1} I(T(x; \theta_m, \Omega_m) = k \wedge (x, y) \in O_m)}{\sum^M_{m=1} I((x, y) \in O_m)}, \tag{12}$$

where the denominator counts the number of base learners for which (x, y) is out-of-bag. If we were to use a separate testing data set S, as we do in our examples, this can be further simplified to

$$\hat{P}(f_M(x) = k) = \frac{1}{M} \sum_{m=1}^{M} I(T(x; \theta_m, \Omega_m) = k), \tag{13}$$

where any instance (x, y) must be selected from S. From this, the expected margin can be estimated as

$$\hat{E}_{x,y}[\Psi(x, y)] = \hat{E}_{x,y}[\hat{P}(f_M(x) = y) - \hat{P}(f_M(x) = j^M_{max})] \tag{14}$$

and its variance as

$$\hat{V}_{x,y}[\Psi(x, y)] = \hat{E}_{x,y}[(\hat{P}(f_M(x) = y) - \hat{P}(f_M(x) = j^M_{max}))^2] - \hat{E}_{x,y}[\Psi(x, y)]^2. \tag{15}$$

The expectations are computed over the training set for the out-of-bag estimator or the testing data set respectively, depending on which approach is used.

Using Chebyshev's inequality, we can derive a bound for the probability of achieving a negative margin, a measure of the generalization error:

$$\begin{aligned} P(\Psi(x, y) < 0) &\leq P(|E_{x,y}[\Psi(x, y)] - \Psi(x, y)| \geq E_{x,y}[\Psi(x, y)]) \\ &\leq \frac{V_{x,y}[\Psi(x, y)]}{E_{x,y}[\Psi(x, y)]^2}, \end{aligned} \tag{16}$$

which can be estimated from equations (14) and (15). Clearly, this inequality is only useful if the expected margin is positive because otherwise the classification is no better than random.

We now follow Breiman's argument for obtaining a measure of ensemble diversity in terms of the random classifier parameters θ, which in our case include the random rotation in addition to the bootstrap samples. First, we note that for independent and identically distributed (i.i.d.) random parameters θ_1 and θ_2, we have

$$\begin{aligned} E_{\theta_1,\theta_2}[\psi(x, y, \theta_1) \times \psi(x, y, \theta_2)] &= E_{\theta_1}[\psi(x, y, \theta_1)] \times E_{\theta_2}[\psi(x, y, \theta_2)] \\ &= \Psi(x, y) \times \Psi(x, y) \\ &= \Psi(x, y)^2. \end{aligned} \tag{17}$$

Therefore, the variance of $\Psi(x, y)$ can be reformulated as

$$\begin{aligned} V_{x,y}[\Psi(x, y)] &= E_{x,y}[\Psi(x, y)^2] - E_{x,y}[\Psi(x, y)]^2 \\ &= E_{x,y}[E_{\theta_1,\theta_2}[\psi(x, y, \theta_1) \times \psi(x, y, \theta_2)]] - E_{x,y}[E_{\theta_1}[\psi(x, y, \theta_1)]] \times E_{x,y}[E_{\theta_2}[\psi(x, y, \theta_2)]] \\ &= E_{\theta_1,\theta_2}[E_{x,y}[\psi(x, y, \theta_1) \times \psi(x, y, \theta_2)] - E_{x,y}[\psi(x, y, \theta_1)] \times E_{x,y}[\psi(x, y, \theta_2)]] \\ &= E_{\theta_1,\theta_2}[Cov_{x,y}[\psi(x, y, \theta_1), \psi(x, y, \theta_2)]] \\ &= E_{\theta_1,\theta_2}[\rho(\psi(x, y, \theta_1), \psi(x, y, \theta_2))] \times E_{\theta_1,\theta_2}[\sigma(\psi(x, y, \theta_1)) \times \sigma(\psi(x, y, \theta_2))] \\ &= E_{\theta_1,\theta_2}[\rho(\psi(x, y, \theta_1), \psi(x, y, \theta_2))] \times E_{\theta_1,\theta_2}[\sigma(\psi(x, y, \theta))]^2. \end{aligned} \tag{18}$$

This result allows us to express the average correlation of the raw margin functions across all classifiers in the ensemble in terms of quantities we can easily estimate, specifically

$$\bar{\rho}(\cdot) = \frac{V_{x,y}[\Psi(x, y)]}{E_{\theta_1,\theta_2}[\sigma(\psi(x, y, \theta))]^2}, \tag{19}$$

where we can use (15) as an estimate of the numerator. In other words, correlation represents the variance of the margin across instances, divided by the expectation of the standard deviation of the raw margin across (randomized) classifiers squared.

To estimate the denominator across all random parameters θ – i.e. the individual base learners and rotations – we can use

$$\begin{aligned} E_{\theta_1,\theta_2}[\sigma(x, y, \theta)] = \frac{1}{M} \sum_{m=1}^{M} &\sqrt{P(f_m(x) = y) + P(f_m(x) = j^m_{max}) -} \\ &\overline{(P(f_m(x) = y) - P(f_m(x) = j^m_{max}))^2}, \end{aligned} \tag{20}$$

where the probabilities are calculated for each individual classifier across all instances (x, y) in the out-of-bag or test set respectively, i.e. in the case of a test set S we would use

$$\hat{P}(f_m(x) = k) = \frac{1}{|S|} \sum_{(x_i,y)\in S} I(T(x_i; \theta_m, \Omega_m) = k), \tag{21}$$

with $|S|$ denoting the cardinality of S.

Appendix C

name	cases	preds
anneal	798	51
audiology	200	85
balance	625	16
breast-w	699	80
breast-y	286	34
chess	28056	34
cleveland	303	22
credit-a	690	14
flare	1066	21
glass	214	9
hayes-roth	132	4
hepatitis	155	29
horse-colic	300	80
ionosphere	351	33
iris	150	4
led24	3200	24
liver	345	44
lymph	148	18
nursery	12960	19
pima	768	167
segmentation	210	19
solar	323	22
sonar	208	60
soybean	307	97
threeOf9	512	9
tic-tac-toe	958	18
votes	435	32
waveform	5000	21
wine	178	13

Table 5: Description of UCI datasets used to perform the detailed tests of random rotations. The the total number of available instances, as well as the number of available predictor variables after preprocessing (including dummy variables for categories) is shown for each data set. The tests use a random 70% of the available instances as training set and the remaining 30% as a test set.

	B-SCALE					Q-SCALE					RANKED				
NAME	**rf**	**rrrf**	**et**	**rret**	**rot**	**rf**	**rrrf**	**et**	**rret**	**rot**	**rf**	**rrrf**	**et**	**rret**	**rot**
anneal	173	165	100	103	188	170	173	100	100	168	167	173	102	100	207
audiology	2060	2040	1687	1720	2927	2053	2027	1700	1687	2947	2400	2060	1693	1693	2960
balance	2694	2715	3728	3728	1957	2700	2679	3721	3715	1966	2715	2683	3715	3719	1966
breast-w	383	373	389	383	415	379	383	389	387	417	385	381	389	389	408
breast-y	2586	2623	2879	2888	2902	2586	2600	2888	2865	2893	2614	2586	2851	2893	2898
chess	4907	4905	3539	3536	4911	4906	4908	3537	3537	4912	4911	4906	3535	3540	4911
cleveland	4255	4233	4316	4251	4356	4273	4246	4312	4229	4378	4316	4268	4316	4233	4352
credit-a	1323	1431	1479	1546	1471	1313	1375	1448	1515	1394	1313	1383	1385	1446	1331
flare	336	336	418	419	336	336	336	420	420	336	336	336	419	423	336
glass	2191	2966	2246	2929	3218	2215	2763	2314	2671	3286	2203	2714	2283	2763	3415
hayes-ro	2100	2080	1840	1840	2500	2090	2100	1840	1840	2490	2080	2100	1840	1840	2500
hepatitis	1489	1455	1872	1881	1660	1447	1481	1838	1838	1694	1506	1464	1847	1838	1762
horse-c	1520	1489	1711	1680	1484	1516	1511	1711	1684	1484	1524	1547	1751	1698	1507
ionosph	543	343	475	415	611	547	389	483	426	645	555	547	532	604	736
iris	524	444	462	453	338	569	444	462	436	320	516	400	507	409	427
led24	2763	2769	2848	2847	2970	2759	2765	2845	2849	2967	2768	2765	2858	2852	2969
liver	2931	3046	3408	3496	2927	2900	2915	3315	3285	2923	2912	2846	3177	3008	2965
lymph	7270	6889	6952	6921	5831	7143	6889	7016	6762	5867	7111	6921	6984	6889	5849
nursery	487	489	79	79	896	487	487	80	80	889	490	485	79	80	888
pima	2506	2397	2810	2807	2428	2497	2424	2658	2706	2431	2488	2539	2731	2800	2469
segment	743	978	705	914	1168	762	1098	730	1035	1130	743	794	603	762	1340
solar	2775	2796	2994	3006	2763	2784	2779	3006	3006	2771	2804	2779	2998	3006	2767
sonar	1721	1530	1283	1403	1981	1714	1740	1365	1511	2076	1721	1397	1397	1606	2190
soybean	951	933	920	916	1419	946	933	903	903	1381	938	942	908	908	1432
threeOf9	109	104	75	78	366	112	122	78	75	364	104	114	78	78	369
tic-tac-toe	104	101	90	92	688	108	100	93	90	678	106	104	94	93	681
votes	369	366	363	379	437	363	369	373	373	440	366	366	363	363	440
waveform	1462	1330	1381	1335	1563	1465	1347	1386	1362	1594	1459	1353	1411	1378	1588
wine	200	230	126	207	556	200	244	141	207	570	185	259	126	200	430

Table 6: Raw performance comparison between random forest with and without random rotation (rf, rrrf), extra trees with and without random rotation (et, rret), and rotation trees (rot). For all data sets (left-hand side), the comparison is performed over the three scaling methods described in the text as basic scaling (b-scale), quantile scaling (q-scale), and ranking (ranked). The values represent classification errors for classification problems and RMSE for regression problems ($\times$10000).

References

Dean Abbott. Why ensembles win data mining competitions. In *Predictive Analytics Centre of Excellence Tech Talks*, University of California, San Diego, 2012.

Theodore W. Anderson, Ingram Olkin, and Les G. Underhill. Generation of random orthogonal matrices. *SIAM Journal on Scientific and Statistical Computing*, 8:625–629, 1987.

Kevin Bache and Moshe Lichman. *UCI machine learning repository*. University of California, Irvine, School of Information and Computer Science, 2013. URL `http://archive.ics.uci.edu/ml`.

Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

Leo Breiman. Random forests – random features. Technical report, University of California at Berkeley, Berkeley, California, 1999.

Leo Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40: 229–242, 2000.

Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 161–168, 2006.

Adele Cutler and Guohua Zhao. PERT-perfect random tree ensembles. *Computing Science and Statistics*, 33:490–497, 2001.

Koen W. De Bock and Dirk Van den Poel. An empirical evaluation of rotation-based ensemble classifiers for customer churn prediction. *Expert Systems with Applications*, 38: 12293–12301, 2011.

Persi Diaconis and Mehrdad Shahshahani. The subgroup algorithm for generating uniform random variables. *Probability in the Engineering and Informational Sciences*, 1:15–32, 1987.

Haytham Elghazel, Alex Aussem, and Florence Perraud. Trading-off diversity and accuracy for optimal ensemble tree selection in random forests. In *Ensembles in Machine Learning Applications*, Studies in Computational Intelligence, pages 169–179. Springer Berlin Heidelberg, 2011.

Wei Fan, Joe McCloskey, and Philip S. Yu. A general framework for accurate and fast regression by data summarization in random decision trees. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 136–146, New York, NY, USA, 2006. ACM.

Khaled Fawagreh, Mohamed Medhat Gaber, and Eyad Elyan. Random forests: from early developments to recent advancements. *Systems Science & Control Engineering*, 2(1): 602–609, September 2014.

Yoav Freund and Robert Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156, Bari, Italy, 1996. Morgan Kaufmann Publishers Inc.

Mark Galassi. *GNU scientific library reference manual - third edition*. Network Theory Ltd., January 2009.

Nicols Garca-Pedrajas, Csar Garca-Osorio, and Colin Fyfe. Nonlinear boosting projections for ensemble construction. *Journal of Machine Learning Research*, 8:1–33, 2007.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63:3–42, 2006.

Gal Guennebaud, Benot Jacob, et al. *Eigen: linear algebra template library*. 2010. URL `http://eigen.tuxfamily.org`.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer, New York, 2009.

Tin K. Ho. Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, volume 1, pages 278–282, 1995.

Tin K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:832–844, 1998.

Alston S. Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM*, 5:339–342, 1958.

Andrew Kerr, Dan Campbell, and Mark Richards. QR decomposition on GPUs. In *Proceedings of 2nd Workshop on General Purpose Processing on Graphics Processing Units*, GPGPU-2, pages 71–78, New York, NY, USA, 2009. ACM.

Donald E. Knuth. *Art of computer programming, volume 2: seminumerical algorithms*. Addison-Wesley Professional, Reading, Mass, 3 edition edition, November 1997.

Max Kuhn and Kjell Johnson. *Applied predictive modeling*. Springer, New York, 2013 edition edition, September 2013. ISBN 9781461468486.

Ludmila I. Kuncheva and Juan J. Rodriguez. An experimental study on rotation forest ensembles. In *Proceedings of the 7th International Conference on Multiple Classifier Systems*, MCS'07, pages 459–468, Berlin, Heidelberg, 2007. Springer-Verlag.

Dan Ledermann and Carol Alexander. ROM simulation with random rotation matrices. SSRN Scholarly Paper ID 1805662, Social Science Research Network, Rochester, NY, 2011.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002. URL `http://CRAN.R-project.org/doc/Rnews/`.

Fei Tony Liu, Kai Ming Ting, and Wei Fan. Maximizing tree diversity by building complete-random decision trees. In *Proceedings of the 9th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*, PAKDD'05, pages 605–610, Berlin, Heidelberg, 2005. Springer-Verlag.

Makoto Matsumoto and Takuji Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8:3–30, 1998.

Bjoern H. Menze, B. Michael Kelm, Daniel N. Splitthoff, Ullrich Koethe, and Fred A. Hamprecht. On oblique random forests. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II*, ECML PKDD'11, pages 453–469, Berlin, Heidelberg, 2011. Springer-Verlag.

Francesco Mezzadri. How to generate random matrices from the classical compact groups. *Notices of the AMS*, 54:592–604, 2007. NOTICES of the AMS, Vol. 54 (2007), 592-604.

Greg Ridgeway. gbm: generalized boosted regression models, 2013. URL `http://CRAN.R-project.org/package=gbm`. R package version 2.1.

Juan J. Rodriguez, Ludmila I. Kuncheva, and Carlos J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1619–1630, 2006.

Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33:1–39, 2010.

Jaak Simm and Ildefons Magrans de Abril. Extratrees: extratrees method, 2013. URL `http://CRAN.R-project.org/package=extraTrees`. R package version 0.4-5.

Journal of Machine Learning Research 17 (2016) 1-10 Submitted 11/14; Revised 3/15; Published 3/16

Should We Really Use Post-Hoc Tests Based on Mean-Ranks?

Alessio Benavoli ALESSIO@IDSIA.CH
Giorgio Corani GIORGIO@IDSIA.CH
Francesca Mangili FRANCESCA@IDSIA.CH
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)
Scuola Universitaria Professionale della Svizzera italiana (SUPSI)
Università della Svizzera italiana (USI)
Manno, Switzerland

Editor: Charles Elkan

Abstract

The statistical comparison of multiple algorithms over multiple data sets is fundamental in machine learning. This is typically carried out by the Friedman test. When the Friedman test rejects the null hypothesis, multiple comparisons are carried out to establish which are the significant differences among algorithms. The multiple comparisons are usually performed using the mean-ranks test. The aim of this technical note is to discuss the inconsistencies of the mean-ranks post-hoc test with the goal of discouraging its use in machine learning as well as in medicine, psychology, etc.. We show that the outcome of the mean-ranks test depends on the pool of algorithms originally included in the experiment. In other words, the outcome of the comparison between algorithms A and B depends also on the performance of the other algorithms included in the original experiment. This can lead to paradoxical situations. For instance the difference between A and B could be declared significant if the pool comprises algorithms C, D, E and not significant if the pool comprises algorithms F, G, H. To overcome these issues, we suggest instead to perform the multiple comparison using a test whose outcome only depends on the two algorithms being compared, such as the sign-test or the Wilcoxon signed-rank test.

Keywords: statistical comparison, Friedman test, post-hoc test

1. Introduction

The statistical comparison of multiple algorithms over multiple data sets is fundamental in machine learning; it is typically carried out by means of a statistical test. The recommended approach is the Friedman test (Demšar, 2006). Being non-parametric, it does *not* require commensurability of the measures across different data sets, it does *not* assume normality of the sample means and it is *robust* to outliers.

When the Friedman test rejects the null hypothesis of no difference among the algorithms, post-hoc analysis is carried out to assess which differences are significant. A series of pairwise comparison is performed adjusting the significance level via Bonferroni correction or other more powerful approaches (Demšar, 2006; Garcia and Herrera, 2008) to control the family-wise Type I error.

The mean-ranks post-hoc test (McDonald and Thompson, 1967; Nemenyi, 1963), is recommended as pairwise test for multiple comparisons in most books of nonparametric statistics: see for instance Gibbons and Chakraborti (2011, Sec. 12.2.1), Kvam and Vidakovic (2007, Sec. 8.2) and Sheskin (2003, Sec. 25.2). It is also commonly used in machine learning (Demšar, 2006; Garcia and Herrera, 2008). The mean-ranks test is based on the statistic:

$$z = |\bar{R}_A - \bar{R}_B| / \sqrt{\frac{m(m+1)}{6n}},$$

where $\bar{R}_A, \bar{R}_B$ are the mean ranks (as computed by the Friedman test) of algorithms A and B, m is the number of algorithms to be compared and n the number of datasets. The mean-ranks $\bar{R}_A, \bar{R}_B$ are computed considering the performance of all the m algorithms. Thus the outcome of the comparison between A and B depends also on the performance of the other (m-2) algorithms included in the original experiment. This can lead to paradoxical situations. For instance the difference between A and B could be declared *significant* if the pool comprises algorithms C, D, E and *not significant* if the pool comprises algorithms F, G, H. The performance of the remaining algorithms should instead be irrelevant when comparing algorithms A and B. This problem has been pointed out several times in the past by Miller (1966); Gabriel (1969); Fligner (1984) and also by Hollander et al. (2013, Sec. 7.3). Yet it is ignored by most literature on nonparametric statistics. However this issue should not be ignored, as it can increase the type I error when comparing two equivalent algorithms and conversely decrease the power when comparing algorithms whose performance is truly different. In this technical note, all these inconsistencies of the mean-ranks test will be discussed in details and illustrated by means of highlighting examples with the goal of discouraging its use in machine learning as well as in medicine, psychology, etc..

To avoid theses issues, we instead recommend to perform the pairwise comparisons of the post-hoc analysis using the *Wilcoxon signed-rank test* or the *sign test*. The decisions of such tests do not depend on the pool of algorithms included in the initial experiment. It is understood that, regardless the specific test adopted for the pairwise comparisons, it is necessary to control the family-wise type I error. This can be obtained through Bonferroni correction or through more powerful approaches (Demšar, 2006; Garcia and Herrera, 2008).

Even better would be the adoption of the Bayesian methods for hypothesis testing. They overcome the many drawbacks (Demšar, 2008; Goodman, 1999; Kruschke, 2010) of the null-hypothesis significance tests. For instance, Bayesian counterparts of the Wilcoxon and of the sign test have been presented in Benavoli et al. (2014); Benavoli et al. (2014); a Bayesian approach for comparing cross-validated algorithms on multiple data sets is discussed by Corani and Benavoli (2015).

2. Friedman Test

The performance of multiple algorithms tested on multiple datasets can be organized in a matrix:

$$\begin{array}{c} \\ Algorithms \end{array} \begin{array}{c} Datasets \\ \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1n} \\ X_{21} & X_{22} & \dots & X_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ X_{m1} & X_{m2} & \dots & X_{mn} \end{bmatrix} \end{array} \tag{1}$$

where X_{ij} denotes the performance of the i-th algorithm on the j-th dataset (for $i = 1, \dots, m$ and $j = 1, \dots, n$). The observations (performances) in different columns are assumed to be independent. The algorithms are ranked column-by-column and each entry X_{ij} is replaced by its rank relative to the other observations in the j-th column:

$$\mathbf{R} = \begin{bmatrix} R_{11} & R_{12} & \dots & R_{1n} \\ R_{21} & R_{22} & \dots & R_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ R_{m1} & R_{m2} & \dots & R_{mn} \end{bmatrix}, \tag{2}$$

where R_{ij} is the rank of the algorithm i in the j-th dataset. The sum of the i-th row $R_i = \sum_{j=1}^{n} R_{ij}$, $\forall\ i = 1, \dots, m$, depends on how the i-th algorithm performs w.r.t. the other $(m-1)$ algorithms. Under the null hypothesis of the Friedman test (no difference between the algorithms) the average value of R_i is $n(m+1)/2$. The statistic of the Friedman test is

$$S = \frac{12}{nm(m+1)} \sum_{j=1}^{n} \left[R_j - \frac{n(m+1)}{2} \right]^2, \tag{3}$$

which under the null hypothesis has a chi-squared distribution with $m-1$ degrees of freedom. For $m = 2$, the Friedman test corresponds to the sign test.

3. Mean Ranks Post-Hoc Test

If the Friedman test rejects the null hypothesis one has to establish which are the significant differences among the algorithms. If all classifiers are compared to each other, one has to perform $m(m-1)/2$ pairwise comparisons.

When performing multiple comparisons, one has to control the family-wise error rate, namely the probability of at least one erroneous rejection of the null hypothesis among the $m(m-1)/2$ pairwise comparisons. In the following example we control the family-wise error (FWER) rate through the Bonferroni correction, even though more powerful techniques are also available (Demšar, 2006; Garcia and Herrera, 2008). However our discussion of the shortcomings of the mean-ranks test is valid regardless the specific approach adopted to control the FWER.

The mean-rank test claims that the i-th and the j-th algorithm are significantly different if:

$$|\bar{R}_i - \bar{R}_j| \geq z^* \sqrt{\frac{m(m+1)}{6n}}. \tag{4}$$

where $\bar{R}_i = \frac{1}{n} R_i$ is the mean rank of the i-th algorithm and z^* is the Bonferroni corrected $\alpha/m(m-1)$ upper standard normal quantile (Gibbons and Chakraborti, 2011, Sec. 12.2.1). Equation (4) is based on the large sample ($n > 10$) approximation of the distribution of the statistic. The actual distribution of the statistic $|\bar{R}_i - \bar{R}_j|$ is derived assuming all the $(m!)^n$ ranks in (2) to be equally probable. Under this assumption the variance of $|\bar{R}_i - \bar{R}_j|$ is $m(m+1)/6n$, which originates the term under the square root in (4).

The sampling distribution of the statistic $|\bar{R}_i - \bar{R}_j|$ assumes all ranks configurations in (2) to be equally probable. Yet this assumption is not tenable: the post-hoc analysis is performed *because* the null hypothesis of the Friedman test has been rejected.

4. Inconsistencies of the Mean-Ranks Test

We illustrate the inconsistencies the mean-ranks test by presenting three examples. All examples refer to the analysis of the accuracy of different classifiers on multiple data sets. We show that the outcome of the test depends both on the actual difference of accuracy between algorithm A and B *and* on the accuracy of the remaining algorithms.

4.1 Example 1: Artificially Increasing Power

Assume we have tested five algorithms A, B, C, D, E on 20 datasets obtaining the accuracies:

	Datasets																			
A	50	50	50	50	50	50	50	50	50	50	80	80	80	80	80	80	80	80	80	80
B	80	80	80	80	80	80	80	80	80	80	50	50	50	50	50	50	50	50	50	50
C	55	55	55	55	55	55	55	55	55	55	45	45	45	45	45	45	45	45	45	45
D	60	60	60	60	60	60	60	60	60	60	85	85	85	85	85	85	85	85	85	85
E	65	65	65	65	65	65	65	65	65	65	90	90	90	90	90	90	90	90	90	90

The corresponding ranks are:

	Datasets																			
A	1	1	1	1	1	1	1	1	1	1	3	3	3	3	3	3	3	3	3	3
B	5	5	5	5	5	5	5	5	5	5	2	2	2	2	2	2	2	2	2	2
C	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1
D	3	3	3	3	3	3	3	3	3	3	4	4	4	4	4	4	4	4	4	4
E	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5

where better algorithms are given higher ranks. We aim at comparing A and B. Algorithm B is better than A in the first ten datasets, while A is better than B in the remaining ten. The two algorithms have the same mean performance and their differences are symmetrically distributed. Each algorithms wins on half the data sets. Different types of two-sided tests (t-test, Wilcoxon signed-rank test, sign-test) return the same p-value, $p = 1$. The mean-ranks test correspond in this case to the sign-test and thus also its p-value is 1. This is most extreme result in favor of the null hypothesis.

Now assume that we compare A, B together with C, D, E. In the first ten datasets, algorithm A is worse than C, D, E, which in turn are worse than B. In the remaining ten datasets, C is worse than A, B, which in turn are worse than D, E. The p-value of the

Friedman test is $p \approx 10^{-10}$ and, thus, it rejects the null hypothesis. We can thus perform the post-hoc test (4) with $z^* = 2.807$ (the Bonferroni corrected $\alpha/m(m-1)$ upper standard normal quantile for $\alpha = 0.05$ and $m = 5$). The significance level has been adjusted to $\alpha/m(m-1)$, since we are performing $m(m-1)/2$ two-sided comparisons. The mean ranks of A, B are respectively 2 and 3.5 and, thus, since $|\bar{R}_A - \bar{R}_B| = 1.5$ and $z^*\sqrt{\frac{m(m+1)}{6n}} \approx 1.4$ we can reject the null hypothesis. The result of the post-hoc test is that the algorithms A, B have significantly different performance.

The decisions of the mean-ranks test are not consistent:

- if it compares A, B alone, it does not reject the null hypothesis;
- if it compares A, B together with C, D, E, it rejects the null hypothesis concluding that A, B have significantly different performance.

The presence of C, D, E artificially introduces a difference between A, B by changing the mean ranks of A, B. For instance, D and E rank always better than A, while they never outperform B when it works well (i.e., datasets from one to ten); in a real case study, a similar result would probably indicate that while B is well suited for the first ten datasets, D, E and A are better suited for the last ten. The difference (in rank) between A and B is artificially amplified by the presence of D and E only when B is better than A. The point is that a large differences in the global ranks of two classifiers does not necessarily correspond to large differences in their accuracies (and vice versa, as we will see in the next example).

This issue can happen in practice.[1] Assume that a researcher presents a new algorithm A_0 and some of its weaker variations A_1, A_2,...,A_k and compares the new algorithms with an existing algorithm B. When B is better, the rank is $B \succ A_0 \succ \ldots \succ A_k$. When A_0 is better, the rank is $A_0 \succ A_1 \succ \ldots \succ A_k \succ B$. Therefore, the presence of A_1, A_2,...,A_k artificially increases the difference between A_0 and B.

4.2 Example 2: Low Power Due to the Remaining Algorithms

Assume the performance of algorithms A and B on different data sets to be normally distributed as follows:

$$A \sim N(0,1), \quad B \sim N(1.5,1).$$

The pool of algorithms comprises also C, D, E, whose performance is distributed as follows:

$$C \sim N(5,1), \quad D \sim N(6,1), \quad E \sim N(7,1).$$

A collection of 20 data sets is considered.

For the sake of simplicity, assume we want to compare only A and B. There is thus no need of correction for multiple comparisons.

When comparing A and B, the power of the two-sided sign test with $\alpha = 0.05$ is *very* high: 0.94 (we have evaluated the power numerically by Monte Carlo simulation). The power of the mean-ranks test is instead only 0.046. We can explain the large difference

1. We thank the anonymous reviewer for suggesting this example.

of power as follows. The sign test (under normal approximation of the distribution of the statistic) claims significance when:

$$|\bar{R}_A - \bar{R}_B| \geq z^*\sqrt{\frac{1}{n}}$$

while the mean-ranks test (4) claims significance when:

$$|\bar{R}_A - \bar{R}_B| \geq z^*\sqrt{\frac{m(m+1)}{6n}} = z^*\sqrt{\frac{5}{n}},$$

with $m = 5$. Since the algorithms C, D, E have mean performances that are much larger than those of A, B, the mean-ranks difference $|\bar{R}_A - \bar{R}_B|$ is equal for the two test. However the mean-ranks estimates the variance of the statistic $|\bar{R}_A - \bar{R}_B|$ to be five times larger compared to the sign test. The critical value of the mean-ranks test is inflated by $\sqrt{5}$, largely decreasing the power of the test. In fact for the mean-ranks test the variance of $|\bar{R}_A - \bar{R}_B|$ increases with the number of algorithms included in the initial experiment.

4.3 Example 3: Real Classifiers on UCI Data Sets

Finally, we compare the accuracies of seven classifiers on 54 datasets. The classifiers are: J48 decision tree (C_1); hidden naive Bayes (C_2); averaged one-dependence estimator (AODE) (C_3); naive-Bayes (C_4); J48 graft (C_5), locally weighted naive-Bayes (C_6), random forest (C_7). The whole set of results is given in Appendix. Each classifier has been assessed via 10 runs of 10-folds cross-validation. We performed all the experiments using WEKA.[2] All these classifiers are described in Witten and Frank (2005).

The accuracies are reported in Table 2. Assume that our aim is to compare C_1, C_2, C_3, C_4 alone. Therefore, we consider just the first 4 columns in Table 2. The mean ranks are:

$$C_2 = 2.676, \quad C_4 = 1.917, \quad C_1 = 2.518, \quad C_3 = 2.888.$$

The Friedman test rejects the null hypothesis. The pairwise comparisons for the pair C_2, C_4 gives the statistic

$$z = |\bar{R}_2 - \bar{R}_4|/\sqrt{m(m+1)/6n} = 3.06.$$

Since 3.06 is greater than $z^* = 2.64$ (the Bonferroni corrected $\alpha/m(m-1)$ upper standard normal quantile for $\alpha = 0.05$ and $m = 4$), the mean-ranks procedure finds the algorithms C_2, C_4 to be significantly different.

If we compare C_2, C_4 together with C_1, C_5, the mean ranks are:

$$C_2 = 2.713, \quad C_4 = 2.102, \quad C_1 = 2.528, \quad C_5 = 2.657.$$

Again, Friedman test rejects the null hypothesis. The pairwise comparisons for the pair C_2, C_4 gives the statistic

$$z = |\bar{R}_2 - \bar{R}_4|/\sqrt{m(m+1)/6n} = 2.46,$$

2. `http://www.cs.waikato.ac.nz/ml/weka/`

	Card=2	Card=3	Card=4
C_2 vs. C_4	7/10	9/10	3/5
C_2 vs. C_7	1/10	-	-
C_3 vs. C_7	2/10	-	-
C_4 vs. C_6	9/10	5/10	-

Table 1: Pairwise comparisons that are affected (numbers of decisions that are significantly different/number of subsets) by the performance of the other algorithms. Here Card=2 means that, for each pair C_a, C_b on the left column, we are considering the subsets $\{C_a, C_b, C_x, C_y\}$, Card=3 $\{C_a, C_b, C_x, C_y, C_z\}$ and Card=4 $\{C_a, C_b, C_x, C_y, C_z, C_w\}$. The symbol "-" means that the comparison does not depend on the subset of algorithms.

which is smaller than z^*. Thus the difference between algorithms C_2 and C_4 is *not* significant.

The accuracies of C_2 and C_4 are the same in the two cases but again the decisions of the mean-ranks are conditional to the group of classifiers we are considering.

Consider building a set of four classifiers $\{C_2, C_4, C_x, C_y\}$. By differently choosing C_x and C_y we can build ten different such sets. For each subset we run the mean-ranks test to check whether the difference between C_2 and C_4 is significantly different. The difference is claimed to be *significant* in 7 cases and *not significant* in 3 cases.

Now consider a set of five classifiers $\{C_2, C_4, C_x, C_y, C_z\}$. By differently choosing C_x, C_y and C_z we can build ten different such sets. This yields 10 further cases in which we compare again C_2 and C_4. Their difference is claimed to be significant in 9/10 cases.

Table 1 reports the pairwise comparisons for which the statistical decision changes with the pool of classifiers that are considered. The outcome of the mean-ranks test when comparing the same pair of classifiers clearly depends on the pool of alternative classifiers $\{C_x, C_y, \ldots\}$ which is assumed.

4.4 Maximum Type I Error

A further drawback of the mean-ranks test which has not been discussed in the previous examples is that it cannot control the maximum type I error, that is, the probability of falsely declaring any pair of algorithms to be different regardless of the other $m-2$ algorithms. If the accuracies of all algorithms but one are equal, it does not guarantee the family-wise Type I error to be smaller than α when comparing the $m-1$ equivalent algorithms. We point the reader to Fligner (1984) for a detailed discussion on this aspect.

5. A Suggested Procedure

Given the above issues, we recommend to avoid the mean-ranks test for the post-hoc analysis. One should instead perform the multiple comparison using tests whose decision depend only on the two algorithms being compared, such as the sign test or the Wilcoxon signed-rank test. The sign test is more robust, as it only assumes the observations to be identically distributed. Its drawback is low power. The Wilcoxon signed-rank test is more powerful

and thus it is generally recommended (Demšar, 2006). Compared to the sign test, the Wilcoxon signed-rank test makes the additional assumption of a symmetric distribution of the differences between the two algorithms being compared. The decision between sign test and signed-rank test thus depends on whether the symmetry assumption is tenable on to the analyzed data.

Regardless the adopted test, the multiple comparisons should be performed adjusting the significance level to control the family-wise Type-I error. This can be done using the correction for multiple comparison discussed by Demšar (2006); Garcia and Herrera (2008). If we adopt the Wilcoxon signed-rank test in Example 3 for comparing C_2, C_4, we obtain the p-value 0.0002, independently from the performance of the other algorithms. Thus, for any pool of algorithms C_2, C_4, C_x, C_y, we always report the same decision: C_2, C_4 are significantly different because the p-value is less than the Bonferroni corrected significance level $\alpha/m(m-1)$ (in the case $m=4$, $\alpha/m(m-1) = 0.0042$).

6. Software

The MATLAB scripts of the above examples can be downloaded from `ipg.idsia.ch/software/meanRanks/matlab.zip`

7. Conclusions

The mean-ranks post-hoc test is widely used test for multiple pairwise comparison. We discuss a number of drawbacks of this test, which we recommend to avoid. We instead recommend to adopt the sign-test or the Wilcoxon signed-rank, whose decision does not depend on the pool of classifiers included in the original experiment.

We moreover bring to the attention of the reader the Bayesian counterparts of these tests, which overcome the many drawbacks (Kruschke, 2010, Chap.11) of null-hypothesis significance testing.

References

A. Benavoli, F. Mangili, G. Corani, M. Zaffalon, and F. Ruggeri. A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2014)*, pages 1–9, 2014.

A. Benavoli, F. Mangili, F. Ruggeri, and M. Zaffalon. Imprecise Dirichlet Process with application to the hypothesis test on the probability that X ≤ Y. *Journal of Statistical Theory and Practice*, February 2014. doi: 10.1080/15598608.2014.985997. Accepted for publication.

G. Corani and A. Benavoli. A Bayesian approach for comparing cross-validated algorithms on multiple data sets. *Machine Learning*, 2015. Accepted for publication.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Janez Demšar. On the appropriateness of statistical tests in machine learning. In *Workshop on Evaluation Methods for Machine Learning in conjunction with ICML*, 2008.

Michael A. Fligner. A note on two-sided distribution-free treatment versus control multiple comparisons. *Journal of the American Statistical Association*, 79(385):pp. 208–211, 1984.

K Ruben Gabriel. Simultaneous test procedures–some theory of multiple comparisons. *The Annals of Mathematical Statistics*, pages 224–250, 1969.

Salvador Garcia and Francisco Herrera. An extension on" Statistical Comparisons of Classifiers over Multiple Data Sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9(12), 2008.

Jean Dickinson Gibbons and Subhabrata Chakraborti. *Nonparametric Statistical Inference*. Springer, 2011.

Steven N Goodman. Toward evidence-based medical statistics: The p–value fallacy. *Annals of Internal Medicine*, 130(12):995–1004, 1999.

Myles Hollander, Douglas A Wolfe, and Eric Chicken. *Nonparametric Statistical Methods*, volume 751. John Wiley & Sons, 2013.

John K Kruschke. Bayesian data analysis. *Wiley Interdisciplinary Reviews: Cognitive Science*, 1(5):658–676, 2010.

Paul H Kvam and Brani Vidakovic. *Nonparametric Statistics With Applications to Science and Engineering*, volume 653. John Wiley & Sons, 2007.

B. J. McDonald and Jr Thompson, W. A. Rank sum multiple comparisons in one- and two-way classifications. *Biometrika*, 54(3/4):pp. 487–497, 1967.

Rupert G Miller. *Simultaneous Statistical Inference*. Springer, 1966.

P. Nemenyi. *Distribution-free multiple comparisons*. Ph.D. thesis, Princeton University, 1963.

David J Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. CRC Press, 2003.

Ian H Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

Table of Accuracies Used in Example 3

Dataset	C1	C2	C3	C4	C5	C6	C7
anneal	98.44	98	98	96.43	98.55	98.33	99
audiology	78.32	73.42	71.66	71.23	78.32	77.41	73.89
wisconsin-breast-cancer	93.7	96.71	96.99	97.14	93.7	97.28	95.57
cmc	50.71	52.81	51.39	51.05	50.78	50.98	48.67
contact-lenses	81.67	68.33	71.67	71.67	81.67	65	78.33
credit	86.38	84.64	86.67	86.23	86.52	87.25	85.07
german-credit	72.4	76.6	76.6	76	72.4	75.3	73
pima-diabetes	73.7	74.09	75.01	74.36	73.56	74.75	72.67
ecoli	81.52	80.04	81.83	82.12	81.52	80.63	78.84
eucalyptus	64.28	63.2	58.71	51.1	64.01	59.52	59.4
glass	71.58	74.26	73.83	70.63	71.1	75.69	73.33
grub-damage	38.79	36.88	43.92	47.79	39.42	40.13	42.63
haberman	72.87	71.53	72.52	72.52	72.87	73.52	72.16
hayes-roth	60	56.88	60	60	60	60	59.38
cleeland-14	78.82	81.47	81.8	83.44	78.48	82.78	81.81
hungarian-14	78.64	84.39	84.39	84.74	78.64	84.38	81.97
hepatitis	79.46	85.13	83.79	82.5	79.46	82.5	81.25
hypothyroid	99.28	99.18	98.54	98.3	99.28	98.62	98.97
ionosphere	91.17	90.88	90.88	89.17	91.74	89.17	91.75
iris	93.33	92	92.67	92.67	93.33	92	93.33
kr-s-kp	99.44	92.46	91.24	87.89	99.37	91.21	98.87
labor	85	88	84.67	83	85	81.33	84.67
lier-disorders	56.25	56.25	56.25	56.25	56.25	56.25	56.25
lymphography	78.33	85	85.71	84.38	79	86.33	79.62
monks1	98.74	100	85.44	74.64	98.74	82.21	98.56
monks3	98.92	97.84	96.75	96.39	98.92	96.39	97.84
monks	64.72	64.57	63.73	62.24	64.72	64.9	70.72
mushroom	100	99.96	99.95	95.83	100	99.84	100
nursery	97.05	94.28	92.71	90.32	97.08	91.61	98.09
optdigits	78.97	96.17	96.9	92.3	81.01	94.2	91.8
page-blocks	96.62	96.84	96.95	93.51	96.66	94.15	96.97
pasture-production	75	85.83	80.83	80.83	75	81.67	75.83
pendigits	89.05	97.61	97.82	87.78	89.87	94.81	95.67
postoperatie	70	67.78	67.78	66.67	70	66.67	60
primary-tumor	40.11	48.08	47.49	46.89	40.11	49.55	38.31
segment	94.24	96.36	94.5	91.3	94.03	94.29	96.06
solar-flare-C	88.86	88.24	88.54	86.08	88.86	87.92	86.05
solar-flare-m	90.1	87.02	87.92	87	90.1	86.99	85.46
solar-flare-X	97.84	97.53	97.84	93.17	97.84	94.41	95.99
sonar	74.48	79.83	81.26	80.29	74.45	80.79	78.36
soybean	92.39	94.58	93.4	92.08	92.98	93.55	92.68
spambase	92.81	92.31	93.37	89.85	93.22	90.63	93.65
spect-reordered	78.29	82.07	80.93	79.03	78.29	83.15	80.56
splice	94.36	96.18	96.21	95.36	94.2	95.89	89.37
squash-stored	70	58	60	61.67	70	63.67	57.67
squash-unstored	76.67	69	70.67	61.67	76.67	68.67	77.33
tae	47	44.38	47	47	47	47	45.67
credit	84.93	83.91	85.07	84.2	84.93	85.22	83.33
owel	76.67	84.65	77.78	60.3	76.87	77.88	84.95
waveform	74.38	84.52	84.92	79.86	74.9	83.62	79.68
white-clover	56.9	79.29	68.57	66.9	56.9	64.76	70
wine	88.79	98.33	98.33	98.89	89.35	98.33	97.22
yeast	57.01	57.48	56.74	56.8	57.01	57.48	56.26
zoo	92.18	100	95.09	93.18	92.18	96.18	95.09

Table 2: Accuracy of classifiers on different data sets.

Journal of Machine Learning Research 17 (2016) 1-31 Submitted 10/13; Revised 3/15; Published 3/16

Minimax Rates in Permutation Estimation for Feature Matching

Olivier Collier OLIVIER.COLLIER@ENPC.FR
Imagine- LIGM
Université Paris EST
Marne-la-Vallée, FRANCE

Arnak S. Dalalyan ARNAK.DALALYAN@ENSAE.FR
Laboratoire de Statistique
ENSAE - CREST
Malakoff, FRANCE

Editor: Gabor Lugosi

Abstract

The problem of matching two sets of features appears in various tasks of computer vision and can be often formalized as a problem of permutation estimation. We address this problem from a statistical point of view and provide a theoretical analysis of the accuracy of several natural estimators. To this end, the minimax rate of separation is investigated and its expression is obtained as a function of the sample size, noise level and dimension of the features. We consider the cases of homoscedastic and heteroscedastic noise and establish, in each case, tight upper bounds on the separation distance of several estimators. These upper bounds are shown to be unimprovable both in the homoscedastic and heteroscedastic settings. Interestingly, these bounds demonstrate that a phase transition occurs when the dimension d of the features is of the order of the logarithm of the number of features n. For $d = O(\log n)$, the rate is dimension free and equals $\sigma(\log n)^{1/2}$, where σ is the noise level. In contrast, when d is larger than $c \log n$ for some constant $c > 0$, the minimax rate increases with d and is of the order of $\sigma(d \log n)^{1/4}$. We also discuss the computational aspects of the estimators and provide empirical evidence of their consistency on synthetic data. Finally, we show that our results extend to more general matching criteria.

Keywords: permutation estimation, minimax rate of separation, feature matching

1. Introduction

In this paper, we present a rigorous statistical analysis of the problem of permutation estimation and multiple feature matching from noisy observations. More precisely, let $\{X_1, \dots, X_n\}$ and $\{X_1^\sharp, \dots, X_m^\sharp\}$ be two sets of vectors from $\mathbb{R}^d$, hereafter referred to as noisy features, containing many matching elements. That is, for many X_i's there is a $X_j^\sharp$ such that X_i and $X_j^\sharp$ coincide up to an observation noise (or measurement error). Our goal is to estimate an application $\pi^* : \{1, \dots, n\} \to \{1, \dots, m\}$ for which each X_i matches with $X^\sharp_{\pi^*(i)}$ and to provide tight conditions which make it possible to accurately recover π^* from data.

In order to define a statistical framework allowing us to compare different estimators of π^*, we confine[1] our attention to the case $n = m$, that is when the two sets of noisy features have equal sizes. Furthermore, we assume that there exists a unique permutation of $\{1, \dots, n\}$, denoted π^*, leading to pairs of features $(X_i, X^\sharp_{\pi^*(i)})$ that match up to a measurement error. In such a situation, it is clearly impossible to recover the true permutation π^* if some features within the set $\{X_1, \dots, X_n\}$ are too close. Based on this observation, we propose to measure the quality of a procedure of permutation estimation by the minimal distance between pairs of different features for which the given procedure is still consistent. This quantity will be called *separation distance* and will be the main concept of interest in the present study. In this respect, the approach we adopted is close in spirit to the minimax theory of hypotheses testing (see, for instance, Spokoiny (1996); Ingster and Suslina (2003)).

1.1 A Motivating Example: Feature Matching in Computer Vision

Many tasks of computer vision, such as object recognition, motion tracking or structure from motion, are currently carried out using algorithms that contain a step of feature matching, cf. Szeliski (2010); Hartley and Zisserman (2003). The features are usually local descriptors that serve to summarize the images. The most famous examples of such features are perhaps SIFT (Lowe, 2004) and SURF (Bay et al., 2008). Once the features have been computed for each image, an algorithm is applied to match features of one image to those of another one. The matching pairs are then used for estimating the deformation of the object, for detecting the new position of the followed object, for creating a panorama, etc. In this paper, we are interested in simultaneous matching of a large number of features. The main focus is on the case when the two sets of features are extracted from the images that represent the same scene with a large overlap, and therefore the sets of features are (nearly) of the same size and every feature in the first image is also present in the second one. This problem is made more difficult by the presence of noise in the images, and thus in the features as well. Typically, due to the high resolution of most images, the number of features is large and their dimension is relatively large as well (128 for SIFT and 64 for SURF). It is therefore important to characterize the behavior of various matching procedures as a function of the number of features, the dimension and the noise level.

1.2 Main Contributions

We consider four procedures of permutation estimation that naturally arise in this context. The first one is a greedy procedure that sequentially assigns to each feature X_i the closest feature $X_j^\sharp$ among those features that have not been assigned at an earlier step. The three other estimators are defined as minimizers of the (profiled-)log-likelihood under three different modeling assumptions. These three modeling assumptions are that the noise level is constant across all the features (homoscedastic noise), that the noise level is variable (heteroscedastic noise) but known and that the noise level is variable and unknown. The corresponding estimators are respectively called least sum of squares (LSS) estimator, least sum of normalized squares (LSNS) estimator and least sum of logarithms (LSL) estimator.

1. These assumptions are imposed for the purpose of getting transparent theoretical results and are in no way necessary for the validity of the considered estimation procedures, as discussed later in the paper.

We first consider the homoscedastic setting and show that all the considered estimators are consistent under similar conditions on the minimal distance between distinct features κ. These conditions state that κ is larger than some function of the noise level σ, the sample size n and the dimension d. This function is the same for the four aforementioned procedures and is given, up to a multiplicative factor, by

$$\kappa^*(\sigma, n, d) = \sigma \max((\log n)^{1/2}, (d \log n)^{1/4}). \tag{1}$$

Then, we prove that this expression provides the optimal rate of the separation distance in the sense that for some absolute constant c if $\kappa \leq c\kappa^*(\sigma, n, d)$ then there is no procedure capable of consistently estimating π^*.

In the heteroscedastic case, we provide an upper bound on the identifiability threshold ensuring the consistency of the LSNS and LSL estimators. Up to a proper normalization by the noise level, this bound is of the same form as (1) and, therefore, the ignorance of the noise level does not seriously affect the quality of estimation. Furthermore, the LSL estimator is easy to adapt to the case $n \neq m$ and is robust to the presence of outliers in the features. We carried out a small experimental evaluation that confirms that in the heteroscedastic setting the LSL estimator is as good as the LSNS (pseudo-) estimator and that they outperform the two other estimators: the greedy estimator and the least sum of squares. We also argue that the three estimators stemming from the maximum likelihood methodology are efficiently computable either by linear programming or by the Hungarian algorithm.

Note that different loss functions may be used for measuring the distance between an estimated permutation and the true one. Most results of this paper are established for the 0-1 loss, which equals one if the estimator and the true permutation differ at least at one location and equals 0 otherwise. However, it is of interest to analyze the situation with the Hamming distance as well, since it amounts to controlling the proportion of the mismatched features and, hence, offers a more graduated evaluation of the quality of estimation. We show that in the case of the Hamming distance, in the regime of moderately large dimension (*i.e.,* $d \geq c \log n$ for some constant $c > 0$) the rate of separation is exactly the same as in the case of the 0-1 distance. The picture is more complex in the regime of small dimensionality $d = o(\log(n))$, in which we get the same upper bound as for the 0-1 loss but the lower bound is expressed in terms of the logarithm of the packing number of an ℓ_2 ball of the symmetric group. We conjecture that this quantity is of the order of $n \log(n)$ and check this conjecture for relatively small values of n. If this conjecture is correct, our lower bound coincides up to a multiplicative factor with the upper bound.

Finally, let us mention that some of the results of the present work have been presented in the AI-STATS 2013 conference and published in the proceedings (Collier and Dalalyan, 2013).

1.3 Plan of the Paper

We introduce in Section 2 a model for the problem of matching two sets of features and of estimation of a permutation. The estimating procedures analyzed in this work are presented in Section 3, whereas their performances in terms of rates of separation distance are described in Section 4. In Section 5, computational aspects of the estimating procedures are discussed

while Section 6 is devoted to the statement of some extensions of our results. We report in Section 7 the results of some numerical experiments. The proofs of the theorems and of the lemmas are postponed to Sections 9 and 10, respectively.

2. Notation and Problem Formulation

We begin with formalizing the problem of matching two sets of features $\{X_1, \dots, X_n\}$ and $\{X_1^\sharp, \dots, X_m^\sharp\}$ with $n, m \geq 2$. In what follows we assume that the observed features are randomly generated from the model

$$\begin{cases} X_i = \theta_i + \sigma_i \xi_i \ , \\ X_j^\sharp = \theta_j^\sharp + \sigma_j^\sharp \xi_j^\sharp, \end{cases} \qquad i = 1, \dots, n \text{ and } j = 1, \dots, m \tag{2}$$

where

- $\boldsymbol{\theta} = \{\theta_1, \dots, \theta_n\}$ and $\boldsymbol{\theta}^\sharp = \{\theta_1^\sharp, \dots, \theta_m^\sharp\}$ are two collections of vectors from $\mathbb{R}^d$, corresponding to the original features, which are unavailable,
- $\sigma_1, \dots, \sigma_n, \sigma_1^\sharp, \dots, \sigma_m^\sharp$ are positive real numbers corresponding to the levels of noise contaminating each feature,
- $\xi_1, \dots, \xi_n$ and $\xi_1^\sharp, \dots, \xi_m^\sharp$ are two independent sets of i.i.d. random vectors drawn from the Gaussian distribution with zero mean and identity covariance matrix.

The task of feature matching consists in finding a bijection π^* between the largest possible subsets S_1 and S_2 of $\{1, \dots, n\}$ and $\{1, \dots, m\}$ respectively, such that

$$\forall\, i \in S_2, \quad \theta_i^\sharp \equiv \theta_{\pi^*(i)}, \tag{3}$$

where $\equiv$ is an equivalence relation that we call *matching criterion*. The features that do not belong to S_1 or S_2 are called *outliers*. To ease presentation, we mainly focus on the case where the matching criterion is the equality of two vectors. However, as discussed in Section 6.2, most results carry over the equivalence corresponding to equality of two vectors transformed by a given linear transformation. Furthermore, it turns out that statistical inference for the matching problem is already quite involved when no outlier is present in the data. Therefore, we make the following assumption

$$m = n \qquad \text{and} \qquad S_1 = S_2 = \{1, \dots, n\}. \tag{4}$$

Note however that the procedures we consider below admit natural counterparts in the setting with outliers. We will also restrict ourselves to noise levels satisfying some constraints. The two types of constraints we consider, referred to as homoscedastic and heteroscedastic setting, correspond to the relations $\sigma_i = \sigma_i^\sharp = \sigma$, $\forall i = 1, \dots, n$, and $\sigma_{\pi^*(i)} = \sigma_i^\sharp$, $\forall i = 1, \dots, n$.

In this formulation, the data generating distribution is defined by the (unknown) parameters $\boldsymbol{\theta}$, $\boldsymbol{\sigma} = (\sigma_i, \dots, \sigma_n)$ and π^*. In the problem of matching, we focus our attention on the problem of estimating the parameter π^* only, considering $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$ as nuisance parameters. In what follows, we denote by $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}$ the probability distribution of the vector $(X_1, \dots, X_n, X_1^\sharp, \dots, X_n^\sharp)$ defined by (2) under the conditions (3) and (4). We write $\mathbf{E}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}$

for the expectation with respect to $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}$. The symmetric group, *i.e.,* the set of all permutations of $\{1,\dots,n\}$, will be denoted by $\mathfrak{S}_n$.

We will use two measures of quality for quantifying the error of an estimator $\hat{\pi}$ of the permutation π^*. These errors are defined as the 0-1 distance and the normalized Hamming distance between $\hat{\pi}$ and π^*, given by

$$\delta_{0\text{-}1}(\hat{\pi},\pi^*) \triangleq \mathbb{1}_{\{\hat{\pi}\neq\pi^*\}}, \qquad \delta_H(\hat{\pi},\pi^*) \triangleq \frac{1}{n}\sum_{k=1}^{n}\mathbb{1}_{\{\hat{\pi}(k)\neq\pi^*(k)\}}. \tag{5}$$

Our ultimate goal is to design estimators that have an expected error smaller than a prescribed level α under the weakest possible conditions on the nuisance parameter $\boldsymbol{\theta}$. The estimation of the permutation or, equivalently, the problem of matching is more difficult when the features are hardly distinguishable. To quantify this phenomenon, we introduce the separation distance $\kappa(\boldsymbol{\theta})$ and the relative separation distance $\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})$, which measure the minimal distance between distinct features and the minimal distance-to-noise ratio, respectively. The precise definitions are

$$\kappa(\boldsymbol{\theta}) \triangleq \min_{i\neq j}\|\theta_i-\theta_j\|, \qquad \bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma}) \triangleq \min_{i\neq j}\frac{\|\theta_i-\theta_j\|}{(\sigma_i^2+\sigma_j^2)^{1/2}}. \tag{6}$$

We will see that in the heteroscedastic case the last quantity is more suitable for characterizing the behavior of the estimators than the first one.

Clearly, if $\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})=0$ and σ_i's are all equal, then the parameter π^* is nonidentifiable, in the sense that there exist two different permutations π_1^* and π_2^* such that the distributions $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi_1^*}$ and $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi_2^*}$ coincide. Therefore, the condition $\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})>0$ is necessary for the existence of consistent estimators of π^*. Furthermore, good estimators are those consistently estimating π^* even if $\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})$ is small. To give a precise sense to these considerations, let $\alpha\in(0,1)$ be a prescribed tolerance level and let us call *perceivable separation distance* of a given estimation procedure $\hat{\pi}$ the quantity

$$\bar{\kappa}_\alpha(\hat{\pi}) \triangleq \inf\Big\{\kappa>0 \,\Big|\, \max_{\pi\in\mathfrak{S}_n}\sup_{\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})>\kappa}\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi}(\hat{\pi}\neq\pi)\le\alpha\Big\},$$

where we skip the dependence on n, d and $\boldsymbol{\sigma}$. Here, the perceivable separation distance is defined with respect to the 0-1 distance, the corresponding definition for the Hamming distance is obtained by replacing $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi}(\hat{\pi}\neq\pi)$ by $\mathbf{E}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi}[\delta_H(\hat{\pi},\pi)]$. Finally, we call *minimax separation distance* the smallest possible perceivable separation distance achieved by an estimator $\hat{\pi}$, *i.e.,*

$$\bar{\kappa}_\alpha \triangleq \inf_{\hat{\pi}}\ \bar{\kappa}_\alpha(\hat{\pi}), \tag{7}$$

where the infimum is taken over all possible estimators of π^*. In the following sections, we establish nonasymptotic upper and lower bounds on the minimax separation distance which coincide up to a multiplicative constant independent of n, d and $\boldsymbol{\sigma}$. We also show that a suitable version of the maximum profiled likelihood estimator is minimax-separation-rate-optimal both in the homoscedastic and heteroscedastic settings.

3. Estimation Procedures

As already mentioned, we will consider four estimators. The simplest one, called greedy algorithm and denoted by π^{gr} is defined as follows: $\pi^{\mathrm{gr}}(1)=\arg\min_{j\in\{1,\dots,n\}}\|X_j-X_1^\sharp\|$ and, for every $i\in\{2,\dots,n\}$, recursively define

$$\pi^{\mathrm{gr}}(i) \triangleq \arg\min_{j\notin\{\pi^{\mathrm{gr}}(1),\dots,\pi^{\mathrm{gr}}(i-1)\}}\|X_j-X_i^\sharp\|. \tag{8}$$

A drawback of this estimator is that it is not symmetric: the resulting permutation depends on the initial numbering of the features. However, we will show that this estimator is minimax-separation-rate-optimal in the homoscedastic setting.

A common approach for avoiding incremental estimation and taking into consideration all the observations at the same time consists in defining the estimator $\hat{\pi}$ as a maximizer of the profiled likelihood. In the homoscedastic case, which is the first setting we studied, the computations lead to the estimator

$$\pi^{\mathrm{LSS}} \triangleq \arg\min_{\pi\in\mathfrak{S}_n}\sum_{i=1}^{n}\|X_{\pi(i)}-X_i^\sharp\|^2. \tag{9}$$

which will be referred to as the *Least Sum of Squares* (LSS) estimator.

The LSS estimator takes into account the distance between the observations irrespectively of the noise levels. The fact of neglecting the noise levels, while harmless in the homoscedastic setting, turns out to cause serious loss of efficiency in terms of the perceivable distance of separation in the setting of heteroscedastic noise. Yet, in the latter setting, the distance between the observations is not as relevant as the signal-to-noise ratio $\|\theta_i-\theta_j\|^2/(\sigma_i^2+\sigma_j^2)$. Indeed, when the noise levels are small, two noisy but distinct vectors are easier to distinguish than when the noise levels are large.

The computation of the maximum likelihood estimator in the heteroscedastic case with known noise levels also suggests that the signal-to-noise ratio should be taken into account. In this setting, the likelihood maximization leads to the *Least Sum of Normalized Squares* (LSNS) estimator

$$\pi^{\mathrm{LSNS}} \triangleq \arg\min_{\pi\in\mathfrak{S}_n}\sum_{i=1}^{n}\frac{\|X_{\pi(i)}-X_i^\sharp\|^2}{\sigma_{\pi(i)}^2+\sigma_i^{\sharp\,2}}. \tag{10}$$

We will often call the LSNS a pseudo-estimator, to underline the fact that it requires the knowledge of the noise levels σ_i which are generally unavailable.

In the general setting, when no information on the noise levels is available, the likelihood is maximized over all nuisance parameters (features and noise levels). But this problem is underconstrained, and the result of this maximization is $+\infty$. This can be circumvented by assuming a proper relation between the noise levels. As mentioned earlier, we chose the assumption

$$\forall\, i\in\{1,\dots,n\}, \quad \sigma_i^\sharp=\sigma_{\pi^*(i)}. \tag{11}$$

The maximum likelihood estimator under this constraint is the *Least Sum of Logarithms* (LSL) defined as

$$\pi^{\mathrm{LSL}} \triangleq \arg\min_{\pi\in\mathfrak{S}_n}\sum_{i=1}^{n}\log\|X_{\pi(i)}-X_i^\sharp\|^2. \tag{12}$$

We will prove that this estimator is minimax-rate-optimal both in the homoscedastic and the heteroscedastic cases.

4. Performance of the Estimators

The purpose of this section is to assess the quality of the aforementioned procedures. To this end, we present conditions for the consistency of these estimators in the form of upper bounds on their perceivable separation distance. Furthermore, to compare this bounds with the minimax separation distance, we establish lower bounds on the latter and prove that it coincides up to a constant factor with the perceivable separation distance of the LSL estimator.

4.1 Homoscedastic Setup

We start by considering the homoscedastic case, in which upper and lower bounds matching up to a constant are obtained for all the estimators introduced in the previous section.

Theorem 1 *Let $\alpha \in (0,1)$ be a tolerance level and $\sigma_j = \sigma_j^\sharp = \sigma$ for all $j \in \{1,\dots,n\}$. If $\hat\pi$ denotes any one of the estimators (8)-(12), then*

$$\bar\kappa_\alpha(\hat\pi) \le 4\max\Big\{\Big(2\log\frac{8n^2}{\alpha}\Big)^{1/2}, \Big(d\log\frac{4n^2}{\alpha}\Big)^{1/4}\Big\}.$$

An equivalent way of stating this result is that if

$$\kappa = 4\sigma\max\Big\{\Big(4\log\frac{8n^2}{\alpha}\Big)^{1/2}, \Big(4d\log\frac{4n^2}{\alpha}\Big)^{1/4}\Big\}$$

and Θ_κ is the set of all $\boldsymbol{\theta} \in \mathbb{R}^{n\times d}$ such that $\kappa(\boldsymbol{\theta}) \ge \kappa$, then

$$\max_{\pi^*\in\mathfrak{S}_n}\sup_{\boldsymbol{\theta}\in\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat\pi \ne \pi^*) \le \alpha$$

for all the estimators defined in Section 3. Note that this result is nonasymptotic. Furthermore, it tells us that the perceivable separation distance of the procedures under consideration is at most of the order of

$$\max\Big\{(\log n)^{1/2}, (d\log n)^{1/4}\Big\}. \tag{13}$$

It is interesting to observe that there are two regimes in this rate, the boundary of which corresponds to the case where d is of the order of $\log n$. For dimensions that are significantly smaller than $\log n$, the perceivable distance of separation is dimension free. On the other hand, when d is larger than $c\log n$ for some absolute constant $c > 0$, the perceivable distance of separation deteriorates with increasing d at the polynomial rate $d^{1/4}$. However, this result does not allow us to deduce any hierarchy between the four estimators, since it provides the same upper bound for all of them. Moreover, as stated in the next theorem, this bound is optimal up to a multiplicative constant.

Theorem 2 *Assume that $n \ge 6$, Θ_κ is the set of all $\boldsymbol{\theta} \in \mathbb{R}^{n\times d}$ such that $\kappa(\boldsymbol{\theta}) \ge \kappa$. Then there exist two absolute constants $c, C > 0$ such that for*

$$\kappa = 2^{-5/2}\sigma\max\Big\{(\log n)^{1/2}, (cd\log n)^{1/4}\Big\},$$

the following lower bound holds

$$\inf_{\hat\pi}\max_{\pi^*\in\mathfrak{S}_n}\sup_{\boldsymbol{\theta}\in\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat\pi \ne \pi^*) > C,$$

where the infimum is taken over all permutation estimators.

An equivalent way of stating this result is to say that, for any $\alpha \le C$, the minimax distance of separation satisfies the inequality

$$\bar\kappa_\alpha \ge \frac{1}{8}\max\big\{(\log n)^{1/2}, (cd\log n)^{1/4}\big\}.$$

Combined with Theorem 1, this implies that the minimax rate of separation is given by the expression $\max\big\{(\log n)^{1/2}, (d\log n)^{1/4}\big\}$.

In order to avoid any possible confusion, we emphasize that the rate obtained in this and subsequent sections concerns the speed of decay of the separation distance and not the estimation risk measured by $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat\pi \ne \pi^*)$. For the latter, considering κ as fixed, one readily derives from Theorem 1 that

$$\max_{\pi^*\in\mathfrak{S}_n}\sup_{\boldsymbol{\theta}\in\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat\pi \ne \pi^*) \le \max\Big\{8n^2\exp\Big(-\frac{\kappa^2}{2^6\sigma^2}\Big), 4n^2\exp\Big(-\frac{\kappa^4}{2^{10}d\sigma^4}\Big)\Big\} \tag{14}$$

for the four estimators $\hat\pi$ defined in the previous section by equations (8)-(12). We do not know whether the right-hand side of this inequality is the correct rate for the minimax risk $R^{\mathrm{mmx}} = \inf_{\hat\pi}\sup_{(\boldsymbol{\theta},\pi^*)\in\Theta_\kappa\times\mathfrak{S}_n} \mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat\pi \ne \pi^*)$. In fact, one can adapt the proof of Theorem 2 to get a lower bound on R^{mmx} which is of the same form as the right-hand side in (14), but with constants 2^6 and 2^{10} replaced by smaller ones. The ratio of such a lower bound and the upper bound in (14) tends to 0 and, therefore, does not provide the minimax rate of estimation, in the most common sense of the term. However, one may note that the minimax rate of separation established in this work is the analogue of the minimax rate of estimation of the Bahadur risk, see Bahadur (1960); Korostelev (1996); Korostelev and Spokoiny (1996).

4.2 Heteroscedastic Setup

We switch now to the heteroscedastic setting, which allows us to discriminate between the four procedures. Note that the greedy algorithm, the LSS and the LSL have a serious advantage over the LSNS since they can be computed without knowing the noise levels $\boldsymbol{\sigma}$.

Theorem 3 *Let $\alpha \in (0,1)$ and condition (11) be fulfilled. If $\hat\pi$ is either π^{LSNS} (if the noise levels σ_i are known) or π^{LSL} (when the noise levels are unknown), then*

$$\bar\kappa_\alpha(\hat\pi) \le 4\max\Big\{\Big(2\log\frac{8n^2}{\alpha}\Big)^{1/2}, \Big(d\log\frac{4n^2}{\alpha}\Big)^{1/4}\Big\}.$$

This result tells us that the performance of the LSNS and LSL estimators, measured in terms of the order of magnitude of the separation distance, is not affected by the heteroscedasticity of the noise levels. Two natural questions arise: 1) is this performance the best possible over all the estimators of π^* and 2) is the performance of the LSS and the greedy estimator as good as that of the LSNS and LSL?

To answer the first question, we start by discarding the degenerate situations where faster rates can be achieved by estimators that are heavily based on the knowledge of the noise levels $\boldsymbol{\sigma}$. In fact, although considering $\boldsymbol{\sigma}$ as known limits considerably the scope of applications of the methods, the definition (7) involves a minimum over all possible estimators $\hat{\pi}$ which are allowed to depend on $\boldsymbol{\sigma}$. For some specific noise levels $\boldsymbol{\sigma}$, from a purely theoretical point of view, knowing $\boldsymbol{\sigma}$ may lead to a substantially better minimax rate and even to a separation equal to zero, which corresponds to an estimator that has no real practical interest. Indeed, under condition (11), it is possible to estimate the permutation π^* by fitting the noise levels. For instance, we can define an estimator $\hat{\pi}$ as follows: $\hat{\pi}(1) = \arg\min_{i=1,\dots,n} \left|\frac{1}{2d}\|X_i - X_1^\#\|^2 - \sigma_i^2\right|$ and, recursively, for every $j \in \{2,\dots,n\}$,

$$\hat{\pi}(j) = \arg \min_{i \notin \{\hat{\pi}(1),\dots,\hat{\pi}(j-1)\}} \left|\frac{1}{2d}\|X_i - X_j^\#\|^2 - \sigma_i^2\right|.$$

This provides an accurate estimator of π^*—for vectors $\{\theta_j\}$ that are very close—as soon as the noise levels are different enough from each other. In particular, the estimated permutation $\hat{\pi}$ coincides with the true permutation π^* on the event

$$\forall i \in \{1,\dots,n\}, \qquad \left|\frac{1}{2d}\|X_{\pi^*(i)} - X_i^\#\|^2 - \sigma_{\pi^*(i)}^2\right| < \min_{j \neq \pi^*(i)} \left|\frac{\|X_j - X_i^\#\|^2}{2d} - \sigma_j^2\right|,$$

which includes the event:

$$\left|\frac{1}{2d}\|X_{\pi^*(i)} - X_i^\#\|^2 - \sigma_{\pi^*(i)}^2\right| + \left|\frac{\|X_j - X_i^\#\|^2}{2d} - \frac{\sigma_j^2 + \sigma_{\pi^*(i)}^2}{2}\right| < \frac{|\sigma_j^2 - \sigma_{\pi^*(i)}^2|}{2} \qquad (15)$$

for all $(i,j) \in \{1,\dots,n\}^2$ such that $j \neq \pi^*(i)$. Using standard bounds on the tails of the χ^2 distribution recalled in Lemma 10 of Section 9, in the case when all the vectors θ_j are equal, one can check that the left-hand side in (15) is of the order of $(\sigma_{\pi^*(i)}^2 + \sigma_j^2)\max\left\{\sqrt{(\log n)/d}, (\log n)/d\right\}$. This implies that we can consistently identify the permutation when

$$\forall (i,j) \in \{1,\dots,n\}^2, \quad i \neq j, \qquad \left|\frac{\sigma_j^2}{\sigma_i^2} - 1\right| \gg \max\left\{\sqrt{\frac{\log n}{d}}, \frac{\log n}{d}\right\},$$

even if the separation distance is equal to zero. In order to discard such kind of estimators from the competition in the procedure of determining the minimax rates, we restrict our attention to the noise levels for which the values $|\frac{\sigma_j^2}{\sigma_i^2} - 1|$ are not larger than $C\max\left\{\sqrt{(\log n)/d}, (\log n)/d\right\}$ for $j \neq i$.

Theorem 4 *Assume that $n \geq 6$, $\bar{\Theta}_\kappa$ is the set of all $\boldsymbol{\theta} \in \mathbb{R}^{n\times d}$ such that $\bar{\kappa}(\boldsymbol{\theta}, \boldsymbol{\sigma}) \geq \kappa$ and*

$$\frac{\max_i \sigma_i^2}{\min_i \sigma_i^2} - 1 \leq \max\left\{\sqrt{\frac{\log n}{16d}}, \frac{\log n}{16d}\right\}.$$

Then there exist two constants $c, C > 0$ such that $\kappa < (1/8)\max\{(\log n)^{1/2}, (cd\log n)^{1/4}\}$, implies that

$$\inf_{\hat{\pi}} \max_{\pi^* \in \mathfrak{S}_n} \sup_{\boldsymbol{\theta} \in \bar{\Theta}_\kappa} \mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat{\pi} \neq \pi^*) > C,$$

where the infimum is taken over all permutation estimators.

It is clear that the constants c and C of the previous theorem are closely related. The inspection of the proof shows that, for instance, if $c \leq 1/20$ then C is larger than 17%.

Let us discuss now the second question raised earlier in this section and concerning the theoretical properties of the greedy algorithm and the LSS under heteroscedasticity. In fact, the perceivable distances of separation of these two procedures are significantly worse than those of the LSNS and the LSL especially for large dimensions d. We state the corresponding result for the greedy algorithm, a similar conclusion being true for the LSS as well. The superiority of the LSNS and LSL is also confirmed by the numerical simulations presented in Section 7 below. In the next theorem and in the sequel of the paper, we denote by id the identity permutation defined by $id(i) = i$ for all $i \in \{1,\dots,n\}$.

Theorem 5 *Assume that $d \geq 225\log 6$, $n = 2$, $\sigma_1^2 = 3$ and $\sigma_2^2 = 1$. Then the condition $\kappa < 0.1(2d)^{1/2}$ implies that*

$$\sup_{\boldsymbol{\theta} \in \bar{\Theta}_\kappa} \mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},id}(\pi^{gr} \neq id) \geq 1/2.$$

This theorem shows that if d is large, the necessary condition for π^{gr} to be consistent is much stronger than the one obtained for π^{LSL} in Theorem 3. Indeed, for the consistency of π^{gr}, κ needs to be at least of the order of $d^{1/2}$, whereas $d^{1/4}$ is sufficient for the consistency of π^{LSL}. Hence, the maximum likelihood estimators LSNS and LSL that take into account noise heteroscedasticity are, as expected, more interesting than the simple greedy estimator[2].

5. Computational Aspects

At first sight, the computation of the estimators (9)-(12) requires to perform an exhaustive search over the set of all possible permutations, the number of which, $n!$, is prohibitively large. This is in practice impossible to do on a standard PC as soon as $n \geq 20$. In this section, we show how to compute these (maximum likelihood) estimators in polynomial time using, for instance, algorithms of linear programming[3].

To explain the argument, let us consider the LSS estimator

$$\pi^{\mathrm{LSS}} = \arg\min_{\pi \in \mathfrak{S}_n} \sum_{i=1}^n \|X_{\pi(i)} - X_i^\#\|^2.$$

2. It should be noted that the conditions of Theorem 5 are not compatible with those of Theorem 4. Hence, strictly speaking, the former does not imply that the greedy estimator is not minimax under the conditions of the latter.
3. The idea of reducing the problem of permutation estimation to a linear program has been already used in the literature, however, without sufficient theoretical justification: see, for instance, Jebara (2003).

For every permutation π, we denote by P^π the $n \times n$ permutation matrix with coefficients $P^\pi_{ij} = \mathbb{1}_{\{j=\pi(i)\}}$. Then we can give the equivalent formulation

$$\pi^{\mathrm{LSS}} = \arg\min_{\pi \in \mathfrak{S}_n} \mathbf{tr}\big(MP^\pi\big), \tag{16}$$

where M is the matrix with coefficient $\|X_i - X_j^\sharp\|^2$ at the i^{th} row and j^{th} column. The cornerstone of our next argument is the Birkhoff-von Neumann theorem stated below, which can be found for example in (Budish et al., 2013).

Theorem 6 (Birkhoff-von Neumann Theorem) *Assume that $\mathcal{P}$ is the set of all doubly stochastic matrices of size n, i.e., the matrices whose entries are nonnegative and sum up to 1 in every row and every column. Then every matrix in $\mathcal{P}$ is a convex combination of matrices $\{P^\pi : \pi \in \mathfrak{S}_n\}$. Furthermore, permutation matrices are the vertices of the simplex $\mathcal{P}$.*

In view of this result, the combinatorial optimization problem (16) is equivalent to the following problem of continuous optimization:

$$P^{\mathrm{LSS}} = \arg\min_{P \in \mathcal{P}} \mathbf{tr}\big(MP\big), \tag{17}$$

in the sense that π is a solution to (16) if and only if P^π is a solution to (17). To prove this claim, let us remark that for every $P \in \mathcal{P}$, there exist coefficients $\alpha_1, \dots, \alpha_{n!} \in [0,1]$ such that $P = \sum_{i=1}^{n!} \alpha_i P^{\pi_i}$ and $\sum_{i=1}^{n!} \alpha_i = 1$. Therefore, we have $\mathbf{tr}\big(MP\big) = \sum_{i=1}^{n!} \alpha_i \mathbf{tr}\big(MP^{\pi_i}\big) \geq \min_{\pi \in \mathfrak{S}_n} \mathbf{tr}\big(MP^\pi\big)$ and $\mathbf{tr}\big(MP^{\mathrm{LSS}}\big) \geq \mathbf{tr}\big(MP^{\pi^{\mathrm{LSS}}}\big)$.

The great advantage of (17) is that it concerns the minimization of a linear function under linear constraints and, therefore, is a problem of linear programming that can be efficiently solved even for large values of n. The same arguments apply to the estimators π^{LSNS} and π^{LSL}, only the matrix M needs to be changed.

There is a second way to compute the estimators LSS, LSNS and LSL efficiently. Indeed, the computation of the aforementioned maximum likelihood estimators is a particular case of the assignment problem, which consists in finding a minimum weight matching in a weighted bipartite graph, where the matrix of the costs is the matrix M from above. This means that the cost of assigning the i^{th} feature of the first image to the j^{th} feature of the second image is either

- the squared distance $\|X_i - X_j^\sharp\|^2$,
- or the normalized squared distance $\|X_i - X_j^\sharp\|^2/(\sigma_i^2 + (\sigma_j^\sharp)^2)$,
- or the logarithm of the squared distance $\log \|X_i - X_j^\sharp\|^2$.

The so-called Hungarian algorithm presented in Kuhn (1955) solves the assignment problem in time $O(n^3)$.

6. Extensions

In this section, we briefly discuss possible extensions of the foregoing results to other distances, more general matching criteria and to the estimation of an arrangement.

$n =$	4	5	6	7	8	9	10	11	12
$M_n \geq$	19	57	179	594	1939	3441	11680	39520	86575
$\frac{\log M_n}{n \log n} \geq$	0.53	0.50	0.48	0.47	0.455	0.412	0.407	0.401	0.381
$\frac{\log M_n}{n \log n} \leq$	0.53	0.50	0.48	0.47	0.455	0.445	0.436	0.427	0.420

Table 1: The values of $M_n = \mathcal{M}(1/4, B_{2,n}(2), \delta_H)$ for $n \in \{4, \dots, 12\}$. The lower bound is just the cardinality of one ϵ-packing, not necessarily the largest one. The upper bound is merely the cardinality of the ℓ_2-ball.

6.1 Minimax Rates for the Hamming Distance

In the previous sections, the minimax rates were obtained for the error of estimation measured by the risk $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}(\hat{\pi} \neq \pi^*) = \mathbf{E}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}[\delta_{0\text{-}1}(\hat{\pi}, \pi^*)]$, which may be considered as too restrictive. Indeed, one could find acceptable an estimate having a small number of mismatches, if it makes it possible to significantly reduce the perceivable distance of separation. These considerations lead to investigating the behavior of the estimators in terms of the Hamming loss, *i.e.*, to studying the risk

$$\mathbf{E}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}[\delta_H(\hat{\pi}, \pi^*)] = \mathbf{E}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}\Big[\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{\hat{\pi}(i) \neq \pi^*(i)\}}\Big]$$

corresponding to the expected average number of mismatched features. Another advantage of studying the Hamming loss instead of the 0-1 loss is that the former sharpens the difference between the performances of various estimators. Note, however, that thanks to the inequality $\delta_H(\hat{\pi}, \pi^*) \leq \delta_{0\text{-}1}(\hat{\pi}, \pi^*)$, all the upper bounds established for the minimax rate of separation under the 0-1 loss directly carry over to the case of the Hamming loss. This translates into the following theorem.

Theorem 7 *Let $\alpha \in (0,1)$ and condition (11) be fulfilled. If $\hat{\pi}$ is either π^{LSNS} or π^{LSL}, then $\bar{\kappa}_\alpha(\hat{\pi}) \leq 4 \max\big\{\big(2\log\frac{8n^2}{\alpha}\big)^{1/2}, \big(d\log\frac{4n^2}{\alpha}\big)^{1/4}\big\}$. That is, if*

$$\kappa = 4\max\Big\{\Big(2\log\frac{8n^2}{\alpha}\Big)^{1/2}, \Big(d\log\frac{4n^2}{\alpha}\Big)^{1/4}\Big\}$$

and $\bar{\Theta}_\kappa$ is the set of all $\boldsymbol{\theta} \in \mathbb{R}^{n\times d}$ such that $\bar{\kappa}(\boldsymbol{\theta}, \sigma) \geq \kappa$, then

$$\max_{\pi^* \in \mathfrak{S}_n} \sup_{\boldsymbol{\theta} \in \bar{\Theta}_\kappa} \mathbf{E}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}[\delta_H(\hat{\pi}, \pi^*)] \leq \alpha.$$

While this upper bound is an immediate consequence of Theorem 3, getting lower bounds for the Hamming loss appears to be more difficult. To state the corresponding result, let is consider the case of homoscedastic noise and introduce some notation. We denote by $\delta_2(\cdot,\cdot)$ the normalized ℓ^2-distance on the space of permutations $\mathfrak{S}_n$: $\delta_2(\pi,\pi')^2 = \frac{1}{n}\sum_{k=1}^n \big(\pi(k) - \pi'(k)\big)^2$. Let $B_{2,n}(R)$ be the ball of $(\mathfrak{S}_n, \delta_2)$ with radius R centered at *id*. As usual, we denote by $\mathcal{M}(\epsilon, B_{2,n}(R), \delta_H)$ the ϵ-packing number of the ℓ_2-ball $B_{2,n}(R)$ in the metric δ_H. This

means that $\mathcal{M}(\epsilon, B_{2,n}(R), \delta_H)$ is the largest integer M such that there exist permutations $\pi_1, \dots, \pi_M \in B_{2,n}(R)$ satisfying $\delta_H(\pi_i, \pi_j) \geq \epsilon$ for every $i \neq j$. One can easily check that replacing $B_{2,n}(R)$ by any other ball of radius R leaves the packing number $\mathcal{M}(\epsilon, B_{2,n}(R), \delta_H)$ unchanged. We set $M_n = \mathcal{M}(1/4, B_{2,n}(2), \delta_H)$.

Theorem 8 *Let $\sigma_k = \sigma$ for all $k \in \{1, \dots, n\}$ and $\bar{\Theta}_\kappa$ is the set of all $\boldsymbol{\theta} \in \mathbb{R}^{n \times d}$ such that $\bar{\kappa}(\boldsymbol{\theta}, \boldsymbol{\sigma}) \geq \kappa$. Furthermore, assume that one of the following two conditions is fulfilled:*

- *$n \geq 3$ and $\kappa = (1/4)\big(\frac{\log M_n}{n}\big)^{1/2}$,*
- *$n \geq 26$, $d \geq 24 \log n$ and $\kappa \leq (1/8)(d \log n)^{1/4}$.*

Then $\inf_{\hat{\pi}} \max_{\pi^* \in \mathfrak{S}_n} \sup_{\boldsymbol{\theta} \in \bar{\Theta}_\kappa} \mathbf{E}_{\boldsymbol{\theta}, \boldsymbol{\sigma}, \pi^*}[\delta_H(\hat{\pi}, \pi^*)] > 2.15\%$.

This result implies that in the regime of moderately large dimension, $d \geq 24 \log n$, the minimax rate of the separation is the same as the one under the 0-1 loss and it is achieved by the LSL estimator. The picture is less clear in the regime of small dimensions, $d = o(\log n)$. If one proves that for some $c > 0$, the inequality $\log M_n \geq cn \log n$ holds for every $n \geq 3$, then the lower bound of the last theorem matches the upper bound of Theorem 7 up to constant factors and leads to the minimax rate of separation $\max\{(\log n)^{1/2}, (d \log n)^{1/4}\}$. Unfortunately, we were unable to find any result on the order of magnitude of $\log M_n$, therefore, we cannot claim that there is no gap between our lower bound and the upper one. However, we did a small experiment for evaluating M_n for small values of n. The result is reported in Table 1.

6.2 More General Matching Criteria

In the previous sections, we were considering two vectors θ_i and $\theta_j^\sharp$ as matching if $\theta_i \equiv \theta_j^\sharp$, and $\equiv$ was the usual equality. In this part, we show that our results can be extended to more general matching criteria, defined as follows. Let A, $A^\sharp$ be two known $p \times d$ matrices with some $p \in \mathbb{N}$ and b, $b^\sharp$ be two known vectors from $\mathbb{R}^d$. We write $\theta \equiv_{A,b} \theta^\sharp$, if

$$A(\theta - b) = A^\sharp(\theta^\sharp - b^\sharp). \tag{18}$$

Note that the case of equality studied in previous sections is obtained for $A = A^\sharp = \mathbf{I}_d$ and $b = b^\sharp = 0$, where $\mathbf{I}_d$ is the identity matrix of size d. Let us first note that without loss of generality, by a simple transformation of the features, one can replace (18) by the simpler relation

$$\bar{\theta} = B\bar{\theta}^\sharp, \tag{19}$$

where $\bar{\theta} \in \mathbb{R}^{d_1}$ for $d_1 = \operatorname{rank}(A)$, $\bar{\theta}^\sharp \in \mathbb{R}^{d_2}$ for $d_2 = \operatorname{rank}(A^\sharp)$ and B is a $d_1 \times d_2$ known matrix. Indeed, let $A = U^\top \Lambda V$ (resp. $A^\sharp = \tilde{U}^\top \tilde{\Lambda} \tilde{V}$) be the singular value decomposition of A (resp. $A^\sharp$), with orthogonal matrices $U \in \mathbb{R}^{d_1 \times p}$, $V \in \mathbb{R}^{d_1 \times d}$ and a diagonal matrix $\Lambda \in \mathbb{R}^{d_1 \times d_1}$ with positive entries. Then, one can deduce (19) from (18) by setting $\bar{\theta} = V(\theta - b)$, $\bar{\theta}^\sharp = \tilde{V}(\theta^\sharp - b^\sharp)$ and $B = \Lambda^{-1} U \tilde{U}^\top \tilde{\Lambda}$. Of course, the same transformation should be applied to the observed noisy features, which leads to $\bar{X}_i = V(X_i - b)$ and $\bar{X}_i^\sharp = \tilde{V}(X^\sharp - b^\sharp)$. Since V and $\tilde{V}$ are orthogonal matrices, *i.e.*, satisfy the relations $VV^\top = \mathbf{I}_{d_1}$ and $\tilde{V}\tilde{V}^\top = \mathbf{I}_{d_2}$, the noise component in the transformed noisy features is still white Gaussian.

All the four estimators introduced in Section 3 can be adapted to deal with such type of criterion. For example, denoting by M the matrix $B(B^\top B)^+ B^\top + BB^\top$ where M^+ is the Moore-Penrose pseudoinverse of the matrix M, the LSL estimator should be modified as follows

$$\pi^{\mathrm{LSL}} \triangleq \arg\min_{\pi \in \mathfrak{S}_n} \sum_{i=1}^n \log \|M^+(\bar{X}_{\pi(i)} - B\bar{X}_i^\sharp)\|^2. \tag{20}$$

All the results presented before can be readily extended to this case. In particular, if we assume that all the nonzero singular values of B are bounded and bounded away from 0 and $\operatorname{rank}(B) = q$, then the minimax rate of separation is given by $\max\big\{(\log n)^{1/2}, (q \log n)^{1/4}\big\}$. This rate is achieved, for instance, by the LSL estimator.

Let us briefly mention two situations in which this kind of general affine criterion may be useful. First, if each feature θ (resp. $\theta^\sharp$) corresponds to a patch in an image I (resp. $I^\sharp$), then for detecting pairs of patches that match each other it is often useful to neglect the changes in illumination. This may be achieved by means of the criterion $A\theta = A\theta^\sharp$ with $A = \mathbf{I}_d - \frac{1}{d}\mathbf{1}_d\mathbf{1}_d^\top$. Indeed, the multiplication of the vector θ by A corresponds to removing from pixel intensities the mean pixel intensity of the patch. This makes the feature invariant by change of illumination. The method described above applies to this case and the corresponding rate is $\max\big\{(\log n)^{1/2}, ((d-1)\log n)^{1/4}\big\}$ since the matrix A is of rank $d-1$. Second, consider the case when each feature combines the local descriptor of an image and the location in the image at which this descriptor is computed. If we have at our disposal an estimator of the transformation that links the two images and if this transformation is linear, then we are in the aforementioned framework. For instance, let each θ be composed of a local descriptor $\mathbf{d} \in \mathbb{R}^{d-2}$ and its location $\mathbf{x} \in \mathbb{R}^2$. Assume that the first image I from which the features $\theta_i = [\mathbf{d}_i, \mathbf{x}_i]$ are extracted is obtained from the image $I^\sharp$ by a rotation of the plane. Let R be an estimator of this rotation and $\theta_i^\sharp = [\mathbf{d}_i^\sharp, \mathbf{x}_i^\sharp]$ be the features extracted from the image $I^\sharp$. Then, the aim is to find the permutation π such that $\mathbf{d}_i^\sharp = \mathbf{d}_{\pi(i)}$ and $\mathbf{x}_i^\sharp = R\mathbf{x}_{\pi(i)}$ for every $i = 1, \dots, n$. This corresponds to taking in (19) the matrix B given by

$$B = \begin{pmatrix} \mathbf{I}_{d-2} & 0 \\ 0 & R \end{pmatrix}.$$

This matrix B being orthogonal, the resulting minimax rate of separation is exactly the same as when $B = \mathbf{I}_d$.

Remark 9 *An interesting avenue for future research concerns the determination of the minimax rates in the case when the equivalence of two features is understood under some transformation A which is not completely determined. For instance, one may consider that the features θ and θ' match if there is a matrix A in a given parametric family $\{A_\tau : \tau \in \mathbb{R}\} \subset \mathbb{R}^{d \times d}$ for which $\theta = A\theta'$. In other terms, $\theta \equiv \theta'$ is understood as $\inf_\tau \|\theta - A_\tau \theta'\| = 0$. Such a criterion of matching may be useful for handling various types of invariance (see Collier and Dalalyan (2015); Collier (2012) for invariance by translation).*

6.3 Estimation of an Arrangement

An interesting extension concerns the case of the estimation of a general arrangement, *i.e.*, the case when m and n are not necessarily identical. In such a situation, without loss of

generality, one can assume that $n \leq m$ and look for an injective function

$$\pi^* : \{1, \dots, n\} \to \{1, \dots, m\}.$$

All the estimators presented in Section 3 admit natural counterparts in this rectangular setting. Furthermore, the computational method using the Birkhoff-von Neumann theorem is still valid in this setting, and is justified by the extension of the Birkhoff-von Neumann theorem recently proved by Budish et al. (2013). In this case, the minimization should be carried out over the set of all matrices P of size (n, m) such that $P_{i,j} \geq 0$, and

$$\begin{cases} \sum_{i=1}^{n} P_{i,j} \leq 1 \\ \sum_{j=1}^{m} P_{i,j} = 1 \end{cases}, \quad (i,j) \in \{1, \dots, n\} \times \{1, \dots, m\}.$$

From a practical point of view, it is also important to consider the issue of robustness with respect to the presence of outliers, *i.e.,* when for some i there is no $X_j^\#$ matching with X_i. The detailed exploration of this problem being out of scope of the present paper, let us just underline that the LSL-estimator seems to be well suited for such a situation because of the robustness of the logarithmic function. Indeed, the correct matches are strongly rewarded because $\log(0) = -\infty$ and the outliers do not interfere too much with the estimation of the arrangement thanks to the slow growth of log in $+\infty$.

7. Experimental Results

We have implemented all the procedures in Matlab and carried out numerical experiments on synthetic data. To simplify, we have used the general-purpose solver SeDuMi (Sturm, 1999) for solving linear programs. We believe that it is possible to speed-up the computations by using more adapted first-order optimization algorithms, such as coordinate gradient descent. However, even with this simple implementation, the running times are reasonable: for a problem with $n = 500$ features, it takes about six seconds to compute a solution to (17) on a standard PC.

7.1 Homoscedastic noise

We chose $n = d = 200$ and randomly generated a $n \times d$ matrix $\boldsymbol{\theta}$ with i.i.d. entries uniformly distributed on $[0, \tau]$, with several values of τ varying between 1.4 and 3.5. Then, we randomly chose a permutation π^* (uniformly from $\mathfrak{S}_n$) and generated the sets $\{X_i\}$ and $\{X_i^\#\}$ according to (2) with $\sigma_i = \sigma_i^\# = 1$. Using these sets as data, we computed the four estimators of π^* and evaluated the average error rate $\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{\hat{\pi}(i) \neq \pi^*(i)\}}$. The result, averaged over 500 independent trials, is plotted in Fig. 1.

Note that the three estimators originating from the maximum likelihood methodology lead to the same estimators, while the greedy algorithm provides an estimator which is much worse than the others when the parameter κ is small.

7.2 Heteroscedastic noise

This experiment is similar to the previous one, but the noise level is not constant. We still chose $n = d = 200$ and defined $\boldsymbol{\theta} = \tau I_d$, where I_d is the identity matrix and τ varies between

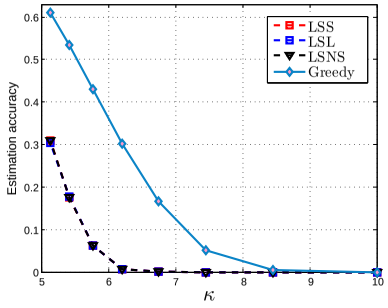


Figure 1: Average error rate of the four estimating procedures in the experiment with homoscedastic noise as a function of the minimal distance κ between distinct features. One can observe that the LSS, LSNS and LSL procedures are indistinguishable and perform much better than the greedy algorithm.

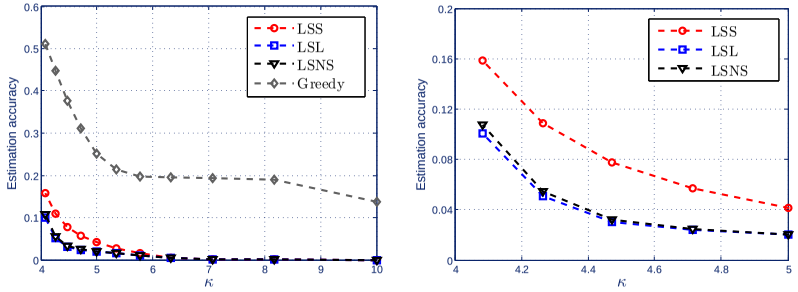


Figure 2: Left: Average error rate of the four estimating procedures in the experiment with heteroscedastic noise as a function of the minimal distance κ between distinct features. Right: zoom on the same plots. One can observe that the LSNS and LSL are almost indistinguishable and, as predicted by the theory, perform better than the LSS and the greedy algorithm.

4 and 10. Then, we randomly chose a permutation π^* (uniformly from $\mathfrak{S}_n$) and generated the sets $\{X_i\}$ and $\{X_i^\#\}$ according to (2) with $\sigma_{\pi^*(i)} = \sigma_i^\# = 1$ for 10 randomly chosen values of i and $\sigma_{\pi^*(i)} = \sigma_i^\# = 0.5$ for the others. Using these sets as data, we computed the four estimators of π^* and evaluated the average error rate $\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{\hat{\pi}(i) \neq \pi^*(i)\}}$. The result, averaged over 500 independent trials, is plotted in Fig. 2.

Note that among the noise-level-adaptive estimators, LSL outperforms the two others and is as accurate as, and even slightly better than the LSNS pseudo-estimator. This confirms the theoretical findings presented in foregoing sections.

8. Conclusion and Future Work

Motivated by the problem of feature matching, we proposed a rigorous framework for studying the problem of permutation estimation from a minimax point of view. The key notion in our framework is the minimax rate of separation, which plays the same role as in the statistical hypotheses testing theory (Ingster and Suslina, 2003). We established theoretical guarantees for several natural estimators and proved the optimality of some of them. The results appeared to be quite different in the homoscedastic and in the heteroscedastic cases. However, we have shown that the least sum of logarithms estimator outperforms the other procedures both theoretically and empirically.

Several avenues of future work have been already mentioned in previous sections. In particular, investigating the statistical properties of the arrangement estimation problem described in Section 6.3 and considering the case of unspecified transformation relating the features may have a significant impact on the practice of feature matching.

Another interesting question is to extend the statistical inference developed here for the problem of feature matching to the more general assignment problem. The latter aims at assigning m tasks to n agents such that the cost of assignment is as small as possible. Various settings of this problem have been considered in the literature (Pentico, 2007) and many algorithms for solving the problem have been proposed (Romeijn and Morales, 2000). However, to the best of our knowledge, the statistical aspects of the problem in the case where the cost matrix is corrupted by noise have not been studied so far.

9. Proofs of the Theorems

In this section we collect the proofs of the theorems. We start with the proof of Theorem 3, since it concerns the more general setting and the proof of Theorem 1 can be deduced from that of Theorem 3 by simple arguments. We then prove the other theorems in the usual order and postpone the proofs of some technical lemmas to the next section.

Proof of Theorem 3 To ease notation and without loss of generality, we assume that π^* is the identity permutation denoted by id. Furthermore, since there is no risk of confusion, we write $\mathbf{P}$ instead of $\mathbf{P}_{\boldsymbol{\theta},\boldsymbol{\sigma},\pi^*}$. We wish to bound the probability of the event $\Omega = \{\hat{\pi} \neq id\}$.

Let us first denote by $\hat{\pi}$ the maximum likelihood estimator π^{LSL} defined by (12). We have

$$\Omega \subset \bigcup_{\pi \neq id} \Omega_\pi,$$

where

$$\Omega_\pi = \Big\{ \sum_{i=1}^n \log \frac{\|X_i - X_i^\#\|^2}{\|X_{\pi(i)} - X_i^\#\|^2} \geq 0 \Big\} = \Big\{ \sum_{i:\pi(i)\neq i} \log \frac{\|X_i - X_i^\#\|^2}{\|X_{\pi(i)} - X_i^\#\|^2} \geq 0 \Big\}.$$

On the one hand, for every permutation π,

$$\begin{aligned}\sum_{\pi(i)\neq i} \log\Big(\tfrac{2\sigma_i^2}{\sigma_i^2+\sigma_{\pi(i)}^2}\Big) &= \sum_{i=1}^n \big(\log(2\sigma_i^2) - \log(\sigma_i^2+\sigma_{\pi(i)}^2)\big)\\ &= \sum_{i=1}^n \frac{\log(2\sigma_i^2)+\log(2\sigma_{\pi(i)}^2)}{2} - \log(\sigma_i^2+\sigma_{\pi(i)}^2)\end{aligned}$$

so, using the concavity of the logarithm, this quantity is nonpositive. Therefore,

$$\begin{aligned}\Omega_\pi &\subset \Big\{ \sum_{i:\pi(i)\neq i} \log \tfrac{\|X_i - X_i^\#\|^2/(2\sigma_i^2)}{\|X_{\pi(i)} - X_i^\#\|^2/(\sigma_i^2+\sigma_{\pi(i)}^2)} \geq 0 \Big\}\\ &\subset \bigcup_{i=1}^n \bigcup_{j\neq i} \Big\{ \frac{\|X_i - X_i^\#\|^2}{2\sigma_i^2} \geq \frac{\|X_j - X_i^\#\|^2}{\sigma_j^2+\sigma_i^2} \Big\}.\end{aligned}$$

This readily yields $\Omega \subset \bar{\Omega}$, where

$$\bar{\Omega} = \bigcup_{i=1}^n \bigcup_{j\neq i} \Big\{ \frac{\|X_i - X_i^\#\|^2}{2\sigma_i^2} \geq \frac{\|X_j - X_i^\#\|^2}{\sigma_j^2+\sigma_i^2} \Big\}. \tag{21}$$

Furthermore, the same inclusion is true for the LSNS estimator as well. Therefore, the rest of the proof is common for the estimators LSNS and LSL.

We set $\sigma_{i,j} = (\sigma_i^2+\sigma_j^2)^{1/2}$ and

$$\zeta_1 = \max_{i\neq j} \Big| \frac{(\theta_i-\theta_j)^\top(\sigma_i\xi_i - \sigma_j\xi_j^\#)}{\|\theta_i-\theta_j\|\sigma_{i,j}} \Big|, \quad \zeta_2 = d^{-1/2} \max_{i,j} \Big| \Big\| \frac{\sigma_i\xi_i - \sigma_j\xi_j^\#}{\sigma_{i,j}} \Big\|^2 - d \Big|.$$

Since $\pi^* = id$, it holds that for every $i \in \{1,\dots,n\}$,

$$\|X_i - X_i^\#\|^2 = \sigma_i^2\|\xi_i - \xi_i^\#\|^2 \leq 2\sigma_i^2(d + \sqrt{d}\zeta_2).$$

Similarly, for every $j \neq i$,

$$\|X_j - X_i^\#\|^2 = \|\theta_j-\theta_i\|^2 + \|\sigma_j\xi_j - \sigma_i\xi_i^\#\|^2 + 2(\theta_j-\theta_i)^\top(\sigma_j\xi_j - \sigma_i\xi_i^\#).$$

Therefore,

$$\|X_j - X_i^\#\|^2 \geq \|\theta_j-\theta_i\|^2 + \sigma_{i,j}^2(d - \sqrt{d}\zeta_2) - 2\|\theta_j-\theta_i\|\sigma_{i,j}\zeta_1.$$

This implies that on the event $\Omega_1 = \{\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma}) \geq \zeta_1\}$ it holds that

$$\frac{\|X_j - X_i^\#\|^2}{\sigma_{i,j}^2} \geq \bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})^2 - 2\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})\zeta_1 + d - \sqrt{d}\zeta_2.$$

Combining these bounds, we get that

$$\Omega \cap \Omega_1 \subset \Big\{ d + \sqrt{d}\zeta_2 \geq \bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})^2 - 2\bar{\kappa}(\boldsymbol{\theta},\boldsymbol{\sigma})\zeta_1 + d - \sqrt{d}\zeta_2 \Big\},$$

which implies that

$$\begin{aligned}\mathbf{P}(\Omega) &\le \mathbf{P}(\Omega_1^{\complement}) + \mathbf{P}\big(\Omega \cap \Omega_1\big)\\ &\le \mathbf{P}\big(\zeta_1 \ge \bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})\big) + \mathbf{P}(2\sqrt{d}\zeta_2 + 2\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})\zeta_1 \ge \bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})^2)\\ &\le 2\mathbf{P}\big(\zeta_1 \ge \tfrac{\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})}{4}\big) + \mathbf{P}\big(\zeta_2 \ge \tfrac{\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})^2}{4\sqrt{d}}\big). \end{aligned} \tag{22}$$

Finally, one easily checks that for suitably chosen random variables $\zeta_{i,j}$ drawn from the standard Gaussian distribution, it holds that $\zeta_1 = \max_{i\neq j}|\zeta_{i,j}|$. Therefore, using the well-known tail bound for the standard Gaussian distribution in conjunction with the union bound, we get

$$\mathbf{P}\big(\zeta_1 \ge \tfrac{1}{4}\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})\big) \le \sum\nolimits_{i\neq j} \mathbf{P}\big(|\zeta_{i,j}| \ge \tfrac{1}{4}\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})\big) \le 2n^2 e^{-\frac{1}{32}\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})^2}. \tag{23}$$

To bound the large deviations of the random variable ζ_2, we rely on the following result.

Lemma 10 (Laurent and Massart (2000), Eq. (4.3) and (4.4)) *If Y is drawn from the chi-squared distribution $\chi^2(D)$, where $D \in \mathbb{N}^*$, then, for every $x > 0$,*

$$\begin{cases} \mathbf{P}\big(Y - D \le -2\sqrt{Dx}\big) \le e^{-x},\\ \mathbf{P}\big(Y - D \ge 2\sqrt{Dx} + 2x\big) \le e^{-x}. \end{cases}$$

As a consequence, $\forall y > 0$, $\mathbf{P}\big(D^{-1/2}|Y - D| \ge y\big) \le 2\exp\big\{-\frac{1}{8}y(y \wedge \sqrt{D})\big\}$.

This inequality, combined with the union bound, yields

$$\mathbf{P}\Big(\zeta_2 \ge \tfrac{\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})^2}{4\sqrt{d}}\Big) \le 2n^2 \exp\Big\{-\frac{(\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma})/16)^2}{d}(\bar\kappa^2(\boldsymbol{\theta},\boldsymbol{\sigma}) \wedge 8d)\Big\}. \tag{24}$$

Combining inequalities (22)-(24), we obtain that as soon as

$$\bar\kappa(\boldsymbol{\theta},\boldsymbol{\sigma}) \ge 4\Big(\sqrt{2\log(8n^2/\alpha)} \vee \big(d\log(4n^2/\alpha)\big)^{1/4}\Big),$$

we have $\mathbf{P}(\hat\pi \neq \pi^*) = \mathbf{P}(\Omega) \le \alpha$.

Proof of Theorem 1 Without loss of generality, we assume $\pi^* = id$. It holds that, on the event

$$\mathcal{A} = \bigcap_{i=1}^n \bigcap_{j\neq i} \Big\{\|X_i - X_i^\#\| < \|X_j - X_i^\#\|\Big\},$$

all the four estimators coincide with the true permutation id. Therefore, we have

$$\{\hat\pi \neq id\} \subseteq \bigcup_{i=1}^n \bigcup_{j\neq i} \Big\{\|X_i - X_i^\#\| \ge \|X_j - X_i^\#\|\Big\}.$$

The latter event is included in $\bar\Omega$ at the right-hand side of (21), the probability of which has been already shown to be small in the previous proof.

Proof of Theorem 2 We refer the reader to the proof of Theorem 4 below, which concerns the more general situation. Indeed, when all the variances σ_j are equal, Theorem 4 boils down to Theorem 2.

Proof of Theorem 4 To establish lower bounds for various types of risks we will use the following lemma:

Lemma 11 (Tsybakov (2009), Theorem 2.5) *Assume that for some integer $M \ge 2$ there exist distinct permutations $\pi_0, \dots, \pi_M \in \mathfrak{S}_n$ and mutually absolutely continuous probability measures $\mathbf{Q}_0, \dots, \mathbf{Q}_M$ defined on a common probability space $(\mathcal{Z}, \mathscr{Z})$ such that*

$$\frac{1}{M}\sum_{j=1}^M K(\mathbf{Q}_j, \mathbf{Q}_0) \le \frac{1}{8}\log M.$$

Then, for every measurable mapping $\tilde\pi : \mathcal{Z} \to \mathfrak{S}_n$,

$$\max_{j=0,\dots,M} \mathbf{Q}_j(\tilde\pi \neq \pi_j) \ge \frac{\sqrt{M}}{\sqrt{M}+1}\Big(\frac{3}{4} - \frac{1}{2\sqrt{\log M}}\Big).$$

To prove Theorem 4, we split the inequality $8\kappa \le \max\{(\log n)^{1/2}, (cd\log n)^{1/4}\}$ into two cases

$$\begin{aligned} &\textbf{Case 1:} && 8\kappa \le (\log n)^{1/2},\\ &\textbf{Case 2:} && (\log n)^{1/2} \le 8\kappa \le (cd\log n)^{1/4}. \end{aligned}$$

Case 1: We assume that $8\kappa \le (\log n)^{1/2}$.
Denote by m the largest integer such that $2m \le n$. We assume without loss of generality that the noise levels are ranked in increasing order: $\sigma_1 \le \ldots \le \sigma_n$. Then, we construct a least favorable set of vectors for the estimation of the permutation. To ease notation, we set $\sigma_{i,j} = (\sigma_i^2 + \sigma_j^2)^{1/2}$.

Lemma 12 *Assume that m is the largest integer such that $2m \le n$. Then there is a set of vectors $\boldsymbol{\theta}$ such that*

$$\frac{\|\theta_1 - \theta_2\|}{\sigma_{1,2}} = \ldots = \frac{\|\theta_{2m-1} - \theta_{2m}\|}{\sigma_{2m-1,2m}} = \kappa,$$

and for every pair $\{i,j\}$ different from the pairs $\{1,2\}, \dots, \{2m-1, 2m\}$ we have

$$\frac{\|\theta_i - \theta_j\|}{\sigma_{i,j}} > \kappa\Big(1 + \frac{\max_{1\le\ell\le n}\sigma_\ell}{\min_{1\le\ell\le n}\sigma_\ell}\Big).$$

Let $\boldsymbol{\theta}^0$ be the set constructed in Lemma 12. The latter implies that $\boldsymbol{\theta}^0$ belongs to $\bar\Theta_\kappa$, so that, denoting for every $k \in \{1, \dots, m\}, \pi_k = (2k-1\ 2k)$ the transposition of $\mathfrak{S}_n$ that only permutes $2k-1$ and $2k$, and $\pi_0 = id$, we get the following lower bound for the risk:

$$\inf_{\hat\pi} \sup_{(\pi,\boldsymbol{\theta})\in\mathfrak{S}_n\times\bar\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\pi}(\hat\pi \neq \pi) \ge \inf_{\hat\pi} \max_{k=0,\dots,m} \mathbf{P}_{\boldsymbol{\theta}^0,\pi_k}(\hat\pi \neq \pi_k).$$

In order to use Lemma 11 with $\mathbf{Q}_j = \mathbf{P}_{\boldsymbol{\theta}^0,\pi_j}$, we compute for every $k \in \{1,\dots,m\}$

$$K(\mathbf{P}_{\boldsymbol{\theta}^0,\pi_k}, \mathbf{P}_{\boldsymbol{\theta}^0,\pi_0}) = \|\theta^0_{2k-1} - \theta^0_{2k}\|^2 \Big(\frac{1}{2\sigma^2_{2k}} + \frac{1}{2\sigma^2_{2k-1}}\Big) + \frac{d}{2}\Big(\frac{\sigma^2_{2k-1}}{\sigma^2_{2k}} + \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 2\Big).$$

Using the fact that $\|\theta^0_{2k-1} - \theta^0_{2k}\|^2 = \kappa^2(\sigma^2_{2k-1} + \sigma^2_{2k})$, we get

$$K(\mathbf{P}_{\boldsymbol{\theta}^k,\pi_k}, \mathbf{P}_{\boldsymbol{\theta}^0,\pi_0}) = \frac{\kappa^2}{2}\big(2 + \sigma^2_{2k-1}\sigma^{-2}_{2k} + \sigma^2_{2k}\sigma^{-2}_{2k-1}\big) + \frac{d}{2}\Big(\frac{\sigma^2_{2k-1}}{\sigma^2_{2k}} + \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 2\Big).$$

The inequality $\sigma^2_{2k-1} \le \sigma^2_{2k}$ implies the following two relations:

$$\frac{\sigma^2_{2k-1}}{\sigma^2_{2k}} + \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 2 \le \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 1$$
$$\frac{\sigma^2_{2k-1}}{\sigma^2_{2k}} + \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 2 = \frac{\sigma^2_{2k-1}}{\sigma^2_{2k}}\Big(\frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 1\Big)^2 \le \Big(\frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 1\Big)^2.$$

This readily yields $\frac{\sigma^2_{2k-1}}{\sigma^2_{2k}} + \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}} - 2 \le \frac{\log n}{16d}$ and, therefore,

$$K(\mathbf{P}_{\boldsymbol{\theta}^k,\pi_k}, \mathbf{P}_{\boldsymbol{\theta}^0,\pi_0}) \le \kappa^2\Big(\frac{3}{2} + \frac{\sigma^2_{2k}}{2\sigma^2_{2k-1}}\Big) + \frac{\log n}{32}.$$

Next, we apply the following result.

Lemma 13 *Let $a_1, a_2, \dots, a_m$ be real numbers larger than one such that $\prod_{k=1}^m a_k \le A$. Then, $\sum_{k=1}^m a_k \le m + \log A \max_k a_k$.*

Proof We use the simple inequality $e^x \le 1 + xe^x$ for all $x \ge 0$. Replacing x by $\log a_k$ and summing over $k = 1,\dots,m$ we get

$$\sum_{k=1}^m a_k \le \sum_{k=1}^m 1 + a_k \log a_k \le m + \max_{k=1,\dots,m} a_k \log \prod_{k=1}^m a_k \le m + \max_{k=1,\dots,m} a_k \log A.$$

This completes the proof of the lemma. ■

We apply this lemma to $a_k = \sigma^2_{2k}/\sigma^2_{2k-1}$. Since the variances are sorted in increasing order, we have $\prod_{k=1}^m a_k \le \prod_{i=1}^{n-1} \sigma^2_{i+1}/\sigma^2_i = \sigma^2_n/\sigma^2_1 \le 1 + \gamma_{n,d}$ with $\gamma_{n,d} = \max\big((\frac{\log n}{16d})^{1/2}, \frac{\log n}{16d}\big)$. In conjunction with the inequality $\log(1+x) \le x$, this entails that $\sum_{k=1}^m \sigma^2_{2k}/\sigma^2_{2k-1} \le m + \gamma_{n,d}(1+\gamma_{n,d})$. Then, since $\log n \le 1.8 \log m$ for $n \ge 6$ and $\gamma_{n,d} \le 0.2 \log n \le 0.36 \log m$, we get

$$\begin{aligned}\frac{1}{m}\sum_{k=1}^m K(\mathbf{P}_{\boldsymbol{\theta}^k,\pi_k}, \mathbf{P}_{\boldsymbol{\theta}^0,\pi_0}) &\le \kappa^2\Big(\frac{3}{2} + \frac{1}{2m}\sum_{k=1}^m \frac{\sigma^2_{2k}}{\sigma^2_{2k-1}}\Big) + \frac{9\log m}{160}\\ &\le \kappa^2\Big(2 + \frac{\gamma_{n,d}}{2m}\big\{1 + \gamma_{n,d}\big\}\Big) + \frac{9\log m}{160}\\ &\le \kappa^2\Big(2 + \frac{0.18\log m}{m}\big\{1 + 0.36\log m\big\}\Big) + \frac{9\log m}{160}.\end{aligned}$$

Finally, using the fact that $m \ge 3$, we get $\frac{1}{m}\sum_{k=1}^m K(\mathbf{P}_{\boldsymbol{\theta}^k,\pi_k}, \mathbf{P}_{\boldsymbol{\theta}^0,\pi_0}) \le 2.1\kappa^2 + \frac{9\log m}{160} \le \frac{\log m}{8}$ since $\kappa^2 \le \frac{1}{64}\log n \le \frac{9}{320}\log m$.

We conclude by Lemma 11 and by the monotonicity of the function $m \mapsto \frac{\sqrt{m}}{1+\sqrt{m}}\big(\frac{3}{4} - \frac{1}{2\sqrt{\log m}}\big)$ that

$$\inf_{\hat\pi} \sup_{(\pi,\boldsymbol{\theta}) \in \mathfrak{S}_n \times \bar\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\pi}(\hat\pi \ne \pi) \ge \frac{\sqrt{3}}{\sqrt{3}+1}\Big(\frac{3}{4} - \frac{1}{2\sqrt{\log 3}}\Big) \ge 0.17.$$

Case 2: We assume that $(\log n)^{1/2} \le 8\kappa \le (cd\log n)^{1/4}$ with $c \le 1/20$. In this case, we have $d \ge \frac{1}{c}\log n$. To get the desired result, we use Lemma 11 for a properly chosen family of probability measures described below.

Lemma 14 *Let $\epsilon_1, \dots, \epsilon_n$ be real numbers defined by*

$$\epsilon_k = \sqrt{2/d}\,\kappa\sigma_k, \qquad \forall k \in \{1,\dots,n\},$$

and let μ be the uniform distribution on $\mathcal{E} = \{\pm\epsilon_1\}^d \times \ldots \times \{\pm\epsilon_n\}^d$. We denote by $\mathbf{P}_{\mu,\pi}$ the probability measure on $\mathbb{R}^{d\times n}$ defined by $\mathbf{P}_{\mu,\pi}(A) = \int_{\mathcal{E}} \mathbf{P}_{\boldsymbol{\theta},\pi}(A)\,\mu(d\boldsymbol{\theta})$. Assume that $\sigma_1 \le \ldots \le \sigma_n$. For two positive integers $k < k' \le n$, set $\gamma = \frac{\sigma^2_{k'}}{\sigma^2_k}$ and let $\pi = (k\ k')$ be the transposition that only permutes k and k'. Then

$$K(\mathbf{P}_{\mu,\pi}, \mathbf{P}_{\mu,id}) \le 4\kappa^2\big(1 - \gamma^{-1}\big) + \frac{8\kappa^4}{d}\big(2 + \big(1 + (2/d)\kappa^2\big)^2\gamma^2\big) + \frac{1}{2}\big(d + 2\kappa^2\big)(\gamma - 1)^2$$

and $\mu(\mathcal{E}\backslash\bar\Theta_\kappa) \le (n(n-1)/2)\ e^{-d/8}$.

The assumption on the noise levels entails that, for any integer $k \in \{1,\dots,k'\}$, $1 \le \frac{\sigma^2_{k'}}{\sigma^2_k} \le 1 + \frac{1}{4}(\frac{\log n}{d})^{1/2}$, and consequently, $(\gamma-1)^2 = \big(\frac{\sigma^2_{k'}}{\sigma^2_k} - 1\big)^2 \le 4^{-2}\big(\frac{\log n}{d}\big)$. Furthermore, $\frac{\kappa^2}{d} \le \frac{c}{64} \le \frac{1}{64}$ provided that $c \le 1$. Finally, for the Kullback-Leibler divergence between $\mathbf{P}_{\mu,\pi_{k,k'}}$ and $\mathbf{P}_{\mu,id}$, where $\pi_{k,k'} = (k\ k')$ is the transposition from $\mathfrak{S}_n$ permuting only k and k', it holds

$$\begin{aligned}K(\mathbf{P}_{\mu,\pi_{k,k'}}, \mathbf{P}_{\mu,id}) &\le \kappa^2\sqrt{\frac{\log n}{d}} + \frac{8\kappa^4}{d}\Big(2 + \frac{33^2}{32^2}\big(1 + 0.25\big)^2\Big) + \frac{33d}{64} \times \frac{\log n}{16d}\\ &\le \frac{\log n}{8} \le \frac{\log n(n-1)/2}{8},\end{aligned}$$

where we have used once again the facts that $c \le 1$ and $n \ge 3$. Applying Lemma 11 with $M = n(n-1)/2$, $\mathbf{Q}_0 = \mathbf{P}_{\mu,id}$ and $\{\mathbf{Q}_j\}_{j=1,\dots,M} = \{\mathbf{P}_{\mu,\pi_{k,k'}}\}_{k\ne k'}$, we obtain

$$\begin{aligned}\max_{\pi^*\in\mathfrak{S}_n} \sup_{\boldsymbol{\theta}\in\bar\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\pi^*}\big(\hat\pi \ne \pi^*\big) &\ge \max_{\pi^*\in\{id\}\cup\{\pi_{k,k'}\}} \int_{\bar\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\pi^*}\big(\hat\pi \ne \pi^*\big)\frac{\mu(d\boldsymbol{\theta})}{\mu(\bar\Theta_\kappa)}\\ &\ge \max_{\pi^*\in\{id\}\cup\{\pi_{k,k'}\}} \mathbf{P}_{\mu,\pi^*}\big(\hat\pi \ne \pi^*\big) - \mu(\mathcal{E}\backslash\bar\Theta_\kappa)\\ &\ge \frac{\sqrt{15}}{\sqrt{15}+1}\Big(\frac{3}{4} - \frac{1}{2\sqrt{\log 15}}\Big) - \frac{n(n-1)}{2}e^{-d/8}.\end{aligned}$$

In view of the inequalities $d \ge (1/c)\log n$, $c \le 1/20$ and $n \ge 6$, we get the inequality $\max_{\pi^*\in\mathfrak{S}_n} \sup_{\boldsymbol{\theta}\in\bar\Theta_\kappa} \mathbf{P}_{\boldsymbol{\theta},\pi^*}\big(\hat\pi \ne \pi^*\big) \ge 22.4\%$.

Proof of Theorem 5 Since the event $\{\pi^{\rm gr} \neq id\}$ includes the event

$$\Omega_2 = \{\|X_1 - X_1^\sharp\|^2 > \|X_2 - X_1^\sharp\|^2\},$$

it is sufficient to bound from below the probability of Ω_2. To this end, we choose any $\boldsymbol{\theta} \in \mathbb{R}^{n\times d}$ satisfying $\|\theta_1 - \theta_2\| = 2\kappa$. This readily implies that $\boldsymbol{\theta}$ belongs to $\bar\Theta_\kappa$. Furthermore, for suitably chosen random variables $\eta_1 \sim \chi^2_d$, $\eta_2 \sim \chi^2_d$ and $\eta_3 \sim \mathcal{N}(0,1)$, it holds that $\|X_1 - X_1^\sharp\|^2 - \|X_2 - X_1^\sharp\|^2 = 6\eta_1 - 4\kappa^2 - 8\kappa\eta_3 - 4\eta_2$. The random terms in the last sum can be controlled using Lemma 10. More precisely, for every $x > 0$, each one of the following three inequalities holds true with probability at least $1 - e^{-x^2}$:

$$\eta_1 \geq d - 2\sqrt{d}x, \quad \eta_2 \leq d + 2\sqrt{d}x + 2x^2, \quad \eta_3 \leq \sqrt{2}x.$$

This implies that with probability at least $1 - 3e^{-x^2}$, we have

$$\|X_1 - X_1^\sharp\|^2 - \|X_2 - X_1^\sharp\|^2 \geq 2d - 20\sqrt{d}x - 4(\kappa + \sqrt{2}x)^2.$$

If $x = \sqrt{\log 6}$, then the conditions imposed on κ and d ensure that the right-hand side of the last inequality is positive. Therefore, $\mathbf{P}(\Omega_2) \geq 1 - 3e^{-x^2} = 1/2$.

Proof of Theorem 8 The proof is split into two parts. In the first part, we consider the case $\kappa \leq \frac{1}{4}\sqrt{\frac{\log M_n}{n}}$, while in the second part the case $\kappa \leq \frac{1}{8}\big(\frac{\log n}{d}\big)^{1/4}$ with $d \geq 24\log n$ and is analyzed. In both cases, the main tool we use is the following result.

Lemma 15 (Tsybakov (2009), Theorem 2.5) *Assume that for some integer $M \geq 2$ there exist distinct permutations $\pi_0, \dots, \pi_M \in \mathfrak{S}_n$ and mutually absolutely continuous probability measures $\mathbf{Q}_0, \dots, \mathbf{Q}_M$ defined on a common probability space $(\mathcal{Z}, \mathscr{Z})$ such that*

$$\begin{cases} \exists\ s > 0,\ \forall\ i \neq j,\ \delta(\pi_i, \pi_j) \geq 2s, \\ \frac{1}{M}\sum_{j=1}^M K(\mathbf{Q}_j, \mathbf{Q}_0) \leq \frac{1}{8}\log M. \end{cases}$$

Then, for every measurable mapping $\tilde\pi : \mathcal{Z} \to \mathfrak{S}_n$,

$$\max_{j=0,\dots,M} \mathbf{Q}_j\big(\delta(\hat\pi, \pi_j) \geq s\big) \geq \frac{\sqrt{M}}{\sqrt{M}+1}\Big(\frac{3}{4} - \frac{1}{2\sqrt{\log M}}\Big).$$

We now have to choose M and $\pi_0, \dots, \pi_M$ in a suitable manner, which will be done differently according to the relationship between n and d.

Case 1: We assume that $\kappa \leq \frac{1}{4}\sqrt{\frac{\log M_n}{n}}$.
Let $M = M_n$ with $M_n = \mathcal{M}(1/4, B_{2,n}(2), \delta_H)$ and let $\boldsymbol{\theta} = (\theta_1, \dots, \theta_n)$ be the set of vectors $\theta_k = k\kappa\sigma(1, 0, \dots, 0) \in \mathbb{R}^d$. Clearly, $\boldsymbol{\theta}$ belongs to $\bar\Theta_\kappa$. By definition of the packing number, there exist $\pi_1, \dots, \pi_{M_n}$, permutations from $\mathfrak{S}_n$, such that

$$\delta_2(\pi_j, id) \leq 2, \qquad \delta_H(\pi_i, \pi_j) \geq \frac{1}{4}; \qquad \forall i, j \in \{1, \dots, M_n\},\ i \neq j.$$

Defining $\mathbf{Q}_j = \mathbf{P}_{\boldsymbol{\theta},\pi_j}$ for $j = 1, \dots, M_n$ and $\mathbf{Q}_0 = \mathbf{P}_{\boldsymbol{\theta},id}$ we get

$$\begin{aligned} K(\mathbf{Q}_j, \mathbf{Q}_0) &= \frac{1}{2\sigma^2}\sum_{k=1}^n \|\theta_{\pi_j(k)} - \theta_k\|^2 = \frac{\kappa^2}{2}\sum_{k=1}^n (\pi_j(k) - k)^2 \\ &= \frac{n\kappa^2}{2}\,\delta_2(\pi_j, id)^2 \leq 2n\kappa^2. \end{aligned}$$

Therefore, using Lemma 15 with $s = 1/8$ we infer from $\kappa \leq \frac{1}{4}\big(\frac{\log M_n}{n}\big)^{1/2}$ that

$$\min_{\hat\pi} \max_{j=0,\dots,M_n} \mathbf{P}_{\boldsymbol{\theta},\pi_j}(\delta_H(\hat\pi, \pi_j) \geq 1/8) \geq \frac{\sqrt{3}}{\sqrt{3}+1}\Big(\frac{3}{4} - \frac{1}{2\sqrt{\log 3}}\Big) \approx 17.31\%.$$

As a consequence, we obtain that

$$\begin{aligned} \min_{\hat\pi} \max_{(\pi,\boldsymbol{\theta})\in\mathfrak{S}_n\times\bar\Theta_\kappa} \mathbf{E}_{\boldsymbol{\theta},\pi}[\delta_H(\hat\pi, \pi)] &\geq \min_{\hat\pi} \max_{j=0,\dots,M} \mathbf{E}_{\boldsymbol{\theta},\pi}[\delta_H(\hat\pi, \pi)\mathbb{1}_{\{\delta_H(\hat\pi,\pi)\geq 1/8\}}] \\ &\geq \frac{1}{8}\min_{\hat\pi} \max_{j=0,\dots,M} \mathbf{E}_{\boldsymbol{\theta},\pi}[\mathbb{1}_{\{\delta_H(\hat\pi,\pi)\geq 1/8\}}] \\ &\geq 2.15\%. \end{aligned}$$

This completes the proof of the first case.

Case 2: We assume that $d \geq 24\log n$ and $\kappa \leq \frac{1}{8}(d\log n)^{1/4}$. Let μ be the uniform distribution on $\{\pm\epsilon\}^{m\times d}$ with $\epsilon = \sqrt{2/d}\,\sigma\kappa$, as in Lemma 14. For any set of permutations $\{\pi_0, \dots, \pi_M\} \subset \mathfrak{S}_n$, in view of Markov's inequality,

$$\sup_{(\pi,\boldsymbol{\theta})\in\mathfrak{S}_n\times\bar\Theta_\kappa} \mathbf{E}_{\boldsymbol{\theta},\pi}\big[\delta_H(\hat\pi, \pi)\big] \geq \frac{3}{16}\Big(\max_{i=0,\dots,M} \mathbf{P}_{\mu,\pi_i}\Big(\delta_H(\hat\pi, \pi_i) \geq \frac{3}{16}\Big) - \mu(\bar\Theta_\kappa^{\complement})\Big).$$

We choose M and $\pi_0, \dots, \pi_M$ as in the following lemma.

Lemma 16 *For any integer $n \geq 4$ there exist permutations $\pi_0, \dots, \pi_M \in \mathfrak{S}_n$ such that*

$$\pi_0 = id, \qquad M \geq (n/24)^{n/6},$$

each π_i is a composition of at most $n/2$ transpositions with disjoint supports, and for every distinct pair of indices $i, j \in \{0, \dots, M\}$ we have

$$\delta_H(\pi_i, \pi_j) \geq 3/8.$$

As π_i is a product of transpositions, the Kullback-Leibler divergence between $\mathbf{P}_{\mu,\pi_i}$ and $\mathbf{P}_{\mu,\pi_0}$ can be computed by independence thanks to Lemma 14:

$$\frac{1}{M}\sum_{i=1}^M K(\mathbf{P}_{\mu,\pi_i}, \mathbf{P}_{\mu,\pi_0}) \leq \frac{n}{2} \times \frac{8\kappa^4}{d}\Big(2 + \Big[1 + \frac{2\kappa^2}{d}\Big]^2\Big) = \frac{16n\kappa^4}{d},$$

where the last inequality follows from the bound $\kappa \leq 0.45 d^{1/2}$. For $n \geq 26$, it holds that $M \geq 2$ and

$$\log M \geq \frac{n(\log n - \log 24)}{6} \geq \frac{\log n}{512}.$$

Consequently, $\frac{16n\kappa^4}{d} \le \frac{1}{8}\log M$ which allows us to apply Lemma 15. This yields

$$\begin{aligned}\inf_{\hat\pi}\sup_{(\pi,\boldsymbol{\theta})\in\mathfrak{S}_n\times\bar\Theta_\kappa}\mathbf{E}_{\boldsymbol{\theta},\pi}[\delta_H(\hat\pi,\pi)] &\ge \frac{3}{16}\Big[\frac{\sqrt{2}}{\sqrt{2}+1}\Big(\frac{3}{4}-\frac{1}{2\sqrt{\log 2}}\Big)-\frac{n^2}{2}e^{-d/8}\Big]\\ &\ge \frac{3}{16}\Big[0.077-\frac{n^2}{2}e^{-24\log(n)/8}\Big]\\ &\ge \frac{3}{16}\Big[0.077-\frac{1}{2n}\Big]\ge 5.81\%.\end{aligned}$$

10. Proofs of the Lemmas

Proof of Lemma 12 Let us denote $r_\sigma = \max_{1\le\ell\le n}\sigma_\ell/\min_{1\le\ell\le n}\sigma_\ell$. It suffices to set $\theta_1 = 0\in\mathbb{R}^d$,

$$\begin{aligned}\theta_{2k+1} &= \kappa\Big(\sigma_{1,2}+\ldots+\sigma_{2k-1,2k}+k\big(1+r_\sigma\big),0,\ldots,0\Big)\in\mathbb{R}^d,\\ \theta_{2k} &= \kappa\Big(\sigma_{1,2}+\ldots+\sigma_{2k-1,2k}+(k-1)\big(1+r_\sigma\big),0,\ldots,0\Big)\in\mathbb{R}^d\end{aligned}$$

for all $k=1,\ldots,m-1$. If n is impair, one can set $\theta_n=\theta_{n-1}+\kappa\big(1+r_\sigma\big)(1,0,\ldots,0)$. One readily checks that these vectors satisfy the desired conditions.

Proof of Lemma 14 Without loss of generality, we assume hereafter that $\pi\in\mathfrak{S}_n$ is the transposition permuting 1 and 2. Recall that the uniform distribution on $\{\pm\epsilon_1\}^d\times\ldots\times\{\pm\epsilon_n\}^d$ can also be written as the product $\mu=\bigotimes_{\ell=1}^n\mu_\ell$, where μ_ℓ is the uniform distribution on $\{\pm\epsilon_\ell\}^d$. Let us introduce an auxiliary probability distribution $\tilde\mu$ on $\mathbb{R}^{n\times d}$ defined as $\tilde\mu=\delta_{\mathbf{0}}\otimes\delta_{\mathbf{0}}\otimes\mu_2\otimes\ldots\otimes\mu_m$ with $\delta_{\mathbf{0}}$ being the Dirac delta measure at $\mathbf{0}\in\mathbb{R}^d$. We set $\mathbf{P}_{\tilde\mu,id}(\cdot)=\int_\Theta\mathbf{P}_{\boldsymbol{\theta},id}(\cdot)\tilde\mu(d\boldsymbol{\theta})$.

We first compute the density of $\mathbf{P}_{\mu,\pi}$ w.r.t. $\mathbf{P}_{\mu,id}$, which can be written as

$$\frac{d\mathbf{P}_{\mu,\pi}}{d\mathbf{P}_{\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)=\frac{d\mathbf{P}_{\mu,\pi}}{d\mathbf{P}_{\tilde\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)\bigg/\Big(\frac{d\mathbf{P}_{\mu,id}}{d\mathbf{P}_{\tilde\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)\Big),\quad \boldsymbol{X},\boldsymbol{X}^\sharp\in\mathbb{R}^{n\times d}.$$

For every $\theta_i\in\mathbb{R}^d$ we denote by $\mathbf{P}_{\theta_i,\sigma_i}$ the probability distribution of X_i from (2), given by

$$\frac{d\mathbf{P}_{\theta_i,\sigma_i}}{d\mathbf{P}_{0,\sigma_j}}(x)=\exp\Big\{-\frac{\|\theta_i\|^2}{2\sigma_i^2}+\frac{1}{\sigma_i^2}(x,\theta_i)-\frac{\|x\|^2}{2}(\sigma_i^{-2}-\sigma_j^{-2})\Big\},\quad\forall x\in\mathbb{R}^d.$$

With this notation, we have

$$\begin{aligned}\frac{d\mathbf{P}_{\mu,\pi}}{d\mathbf{P}_{\tilde\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)&=\mathbf{E}_\mu\bigg[\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_1}}(X_1)\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_2}}(X_2)\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_1}}(X_1^\sharp)\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_2}}(X_2^\sharp)\bigg]\\ &=\mathbf{E}_\mu\bigg[\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_1}}(X_1)\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_2}}(X_2^\sharp)\bigg]\times\mathbf{E}_\mu\bigg[\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_2}}(X_2)\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_1}}(X_1^\sharp)\bigg]\\ &=\prod_{k=1}^d\cosh\Big(\frac{\epsilon_1}{\sigma_1^2}(X_{1,k}+X_{2,k}^\sharp)\Big)\cosh\Big(\frac{\epsilon_2}{\sigma_2^2}(X_{2,k}+X_{1,k}^\sharp)\Big)\\ &\quad\times\exp\Big\{-\frac{1}{2}(\|X_1^\sharp\|^2-\|X_2^\sharp\|^2)(\sigma_2^{-2}-\sigma_1^{-2})\Big\}.\end{aligned}$$

Similarly,

$$\begin{aligned}\frac{d\mathbf{P}_{\mu,id}}{d\mathbf{P}_{\tilde\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)&=\mathbf{E}_\mu\bigg[\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_1}}(X_1)\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_2}}(X_2)\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_1}}(X_1^\sharp)\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_2}}(X_2^\sharp)\bigg]\\ &=\mathbf{E}_\mu\bigg[\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_1}}(X_1)\frac{d\mathbf{P}_{\theta_1,\sigma_1}}{d\mathbf{P}_{0,\sigma_1}}(X_1^\sharp)\bigg]\times\mathbf{E}_\mu\bigg[\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_2}}(X_2)\frac{d\mathbf{P}_{\theta_2,\sigma_2}}{d\mathbf{P}_{0,\sigma_2}}(X_2^\sharp)\bigg]\\ &=\prod_{k=1}^d\cosh\Big(\frac{\epsilon_1}{\sigma_1^2}(X_{1,k}+X_{1,k}^\sharp)\Big)\cosh\Big(\frac{\epsilon_2}{\sigma_2^2}(X_{2,k}+X_{2,k}^\sharp)\Big).\end{aligned}$$

Thus, we get that

$$\begin{aligned}\frac{d\mathbf{P}_{\mu,\pi}}{d\mathbf{P}_{\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)&=\prod_{k=1}^d\frac{\cosh\big(\frac{\epsilon_1}{\sigma_1^2}(X_{1,k}+X_{2,k}^\sharp)\big)}{\cosh\big(\frac{\epsilon_1}{\sigma_1^2}(X_{1,k}+X_{1,k}^\sharp)\big)}\times\prod_{k=1}^d\frac{\cosh\big(\frac{\epsilon_2}{\sigma_2^2}(X_{2,k}+X_{1,k}^\sharp)\big)}{\cosh\big(\frac{\epsilon_2}{\sigma_2^2}(X_{2,k}+X_{2,k}^\sharp)\big)}\\ &\quad\times\exp\Big\{-\frac{1}{2}(\|X_1^\sharp\|^2-\|X_2^\sharp\|^2)(\sigma_2^{-2}-\sigma_1^{-2})\Big\}.\end{aligned}$$

Then, we compute the Kullback-Leibler divergence,

$$\begin{aligned}K(\mathbf{P}_{\mu,\pi},\mathbf{P}_{\mu,id})&=\int\log\Big(\frac{d\mathbf{P}_{\mu,\pi}}{d\mathbf{P}_{\mu,id}}(\boldsymbol{X},\boldsymbol{X}^\sharp)\Big)d\mathbf{P}_{\mu,\pi}(\boldsymbol{X},\boldsymbol{X}^\sharp)\\ &=\sum_{k=1}^d\sum_{j=1}^2\bigg\{\mathbf{E}_\mu\Big[\int\log\cosh\Big[\frac{\epsilon_j}{\sigma_j^2}(2\theta_{j,k}+\sigma_j\sqrt{2}x)\Big]\varphi(x)dx\Big]\\ &\quad-\mathbf{E}_\mu\Big[\int\log\cosh\Big[\frac{\epsilon_j}{\sigma_j^2}(\theta_{1,k}+\theta_{2,k}+\sigma_{12}x)\Big]\varphi(x)dx\Big]\bigg\}\\ &\quad+\frac{d}{2}\mathbf{E}_\mu\Big[\int_{\mathbb{R}}((\theta_{1,1}+\sigma_1x)^2-(\theta_{2,1}+\sigma_2x)^2)\varphi(x)\,dx\Big](\sigma_2^{-2}-\sigma_1^{-2}),\end{aligned}$$

where φ is the density function of the standard Gaussian distribution. We evaluate the first two terms of the last display using the following inequalities:

$$\forall u\in\mathbb{R},\qquad\frac{u^2}{2}-\frac{u^4}{12}\le\log\cosh(u)\le\frac{u^2}{2},\tag{25}$$

while for the third term the exact computation yields:

$$\begin{aligned}\mathbf{E}_\mu\Big[\int_{\mathbb{R}}((\theta_{1,1}+\sigma_1x)^2-(\theta_{2,1}+\sigma_2x)^2)\varphi(x)\,dx\Big]&=\epsilon_1^2+\sigma_1^2-\epsilon_2^2-\sigma_2^2\\ &=(\sigma_1^2-\sigma_2^2)(1+(2/d)\kappa^2).\end{aligned}$$

In conjunction with the facts that $\epsilon_1/\sigma_1 = \epsilon_2/\sigma_2$, $\sigma_1 \le \sigma_2$ and $\epsilon_1 \le \epsilon_2$, this leads to

$$\begin{aligned}
&\mathbf{E}_\mu\biggl[\int \log\cosh\Bigl[\frac{\epsilon_j}{\sigma_j^2}(2\theta_{j,k}+\sigma_j\sqrt{2}x)\Bigr]\varphi(x)dx \le \frac{\epsilon_j^2}{\sigma_j^2}+2\frac{\epsilon_j^4}{\sigma_j^4} = \frac{\epsilon_2^2}{\sigma_2^2}+2\frac{\epsilon_2^4}{\sigma_2^4},\\
&\mathbf{E}_\mu\biggl[\int \log\cosh\Bigl[\frac{\epsilon_j}{\sigma_j^2}(\theta_{1,k}+\theta_{2,k}+\sigma_{1,2}x)\Bigr]\varphi(x)dx\biggr]\\
&\quad\ge \frac{\epsilon_j^2}{2\sigma_j^4}\Bigl(\epsilon_1^2+\epsilon_2^2+\sigma_1^2+\sigma_2^2\Bigr)\\
&\qquad -\frac{\epsilon_j^4}{12\sigma_j^8}\Bigl(\epsilon_1^4+\epsilon_2^4+3(\sigma_1^2+\sigma_2^2)^2+6\epsilon_1^2\epsilon_2^2+6(\sigma_1^2+\sigma_2^2)(\epsilon_1^2+\epsilon_2^2)\Bigr)\\
&\quad\ge \frac{\epsilon_2^2(\epsilon_1^2+\sigma_1^2)}{\sigma_2^4}-\frac{\epsilon_1^4(\epsilon_2^2+\sigma_2^2)^2}{\sigma_1^8}.
\end{aligned}$$

Thus, we get that

$$\begin{aligned}
(1/d)K(\mathbf{P}_{\mu,\pi},\mathbf{P}_{\mu,id}) &\le \frac{2\epsilon_2^2}{\sigma_2^2}+\frac{4\epsilon_2^4}{\sigma_2^4}-\frac{2\epsilon_2^2(\epsilon_1^2+\sigma_1^2)}{\sigma_2^4}+\frac{2\epsilon_1^4(\epsilon_2^2+\sigma_2^2)^2}{\sigma_1^8}\\
&\quad+\frac{1}{2}\bigl(1+(2/d)\kappa^2\bigr)(\sigma_1^2-\sigma_2^2)(\sigma_2^{-2}-\sigma_1^{-2})\\
&\le \frac{4\kappa^2}{d}\Bigl(1-\frac{\sigma_1^2}{\sigma_2^2}\Bigr)+\frac{16\kappa^4}{d^2}+\frac{8\kappa^4}{d^2}\bigl(1+(2/d)\kappa^2\bigr)^2\frac{\sigma_2^4}{\sigma_1^4}\\
&\quad+\frac{1}{2}\bigl(1+(2/d)\kappa^2\bigr)\Bigl(\frac{\sigma_2^2}{\sigma_1^2}-1\Bigr)^2.
\end{aligned}$$

To complete the proof, we need to evaluate $\mu(\mathcal{E}\setminus\bar\Theta_\kappa)$. We note that in view of the union bound,

$$\begin{aligned}
\mu(\mathcal{E}\setminus\bar\Theta_\kappa) &= \mu\Bigl(\bigcup_{k=1}^n\bigcup_{k'\ne k}\{\boldsymbol{\theta}:\|\theta_k-\theta_{k'}\|<\kappa\sigma_{k,k'}\}\Bigr)\\
&\le \frac{n(n-1)}{2}\max_{k\ne k'}\mu\bigl(\{\boldsymbol{\theta}:\|\theta_k-\theta_{k'}\|^2<\kappa^2\sigma_{k,k'}\}\bigr)\\
&= \frac{n(n-1)}{2}\max_{k\ne k'}\mathbf{P}\bigl(d\epsilon_k^2+d\epsilon_{k'}^2-2d\epsilon_k\epsilon_{k'}\bar\zeta<\kappa^2\sigma_{k,k'}^2\bigr),
\end{aligned}$$

where $\bar\zeta=\frac{1}{d}\sum_{j=1}^d\zeta_j$ with $\zeta_1,\dots,\zeta_d$ being i.i.d. Rademacher random variables (*i.e.*, random variables taking the values $+1$ and -1 with probability $1/2$). One easily checks that

$$\frac{d\epsilon_k^2+d\epsilon_{k'}^2-\kappa^2\sigma_{k,k'}^2}{2d\epsilon_k\epsilon_{k'}} = \frac{2\sigma_k^2+2\sigma_{k'}^2-(\sigma_k^2+\sigma_{k'}^2)}{4\sigma_k\sigma_{k'}} \ge \frac{1}{2}.$$

Therefore, using the Hoeffding inequality, we get $\mu(\mathcal{E}\setminus\bar\Theta_\kappa)\le\frac{1}{2}n(n-1)\mathbf{P}\bigl(\bar\zeta>1/2\bigr)\le\frac{1}{2}n(n-1)e^{-d/8}$.

Proof of Lemma 16 We first prove an auxiliary result.

Lemma 17 *For any integer $n\ge 2$ there exist permutations $\pi_0,\pi_1,\dots,\pi_M$ in $\mathfrak{S}_n$ such that*

$$\pi_0 = id, \qquad M\ge (n/8)^{n/2}$$

and for any pair $i,j\in\{0,\dots,M\}$ of distinct indices we have $\delta_H(\pi_i,\pi_j)\ge\frac{1}{2}$.

Proof When $n\le 8$, the claim of this lemma is trivial since one can always find at least one permutation that differs from the identity at all the positions and thus $M\ge 1\ge(n/8)^{n/2}$. Let us consider the case $n>8$. For every $\pi\in\mathfrak{S}_n$, denote

$$E_\pi \triangleq \Bigl\{\pi'\in\mathfrak{S}_n\,\Big|\,\frac{1}{n}\sum_{i=1}^n\mathbb{1}_{\{\pi(i)\ne\pi'(i)\}}\ge 1/2\Bigr\}.$$

We first notice that for every $\pi\in\mathfrak{S}_n$, there is a one-to-one correspondence between E_{id} and E_π through the bijection

$$\phi:\left|\begin{array}{ccc} E_{id} & \longrightarrow & E_\pi\\ \pi' & \longmapsto & \pi\circ\pi'\end{array}\right.,$$

so that $\#E_\pi=\#E_{id}$. The following lemma, proved later on in this section, gives a bound for this number.

Lemma 18 *Let $n\ge 2$ be an integer and m be the smallest integer such that $2m\ge n$. Then*

$$\#E_{id}^{\complement}\le\frac{4n!}{m!}.$$

Now we denote $\pi_0=id$ and choose π_1 in E_{id}. Then, it is sufficient to choose π_2 as any element from $E_{id}\cap E_{\pi_1}$, the latter set being nonempty since

$$\begin{aligned}
\#\bigl(E_{\pi_0}\cap E_{\pi_1}\bigr) &\ge \#\mathfrak{S}_n-\#E_{\pi_0}^{\complement}-\#E_{\pi_1}^{\complement}\\
&\ge n!\times\Bigl(1-\frac{8}{m!}\Bigr)>0.
\end{aligned}$$

We can continue the construction until π_i if

$$1-\frac{4i}{m!}>0 \qquad\Longleftrightarrow\qquad i<\frac{m!}{4}.$$

To conclude, we observe that

$$\frac{m!}{4}>\frac{1}{4}\Bigl(\frac{n}{2e}\Bigr)^{n/2}\ge\Bigl(\frac{n}{8}\Bigr)^{n/2}.$$

■

Let us denote by m the largest integer such that $2m\le n$, and choose

$$\tilde\pi_0,\tilde\pi_1,\dots,\tilde\pi_M\in\mathfrak{S}_m \quad\text{with}\quad M\ge(m/8)^{m/2}$$

as in Lemma 17, so that for every $i\ne j\in\{0,\dots,M\}$, $\delta_H(\tilde\pi_i,\tilde\pi_j)\ge\frac{1}{2}$. We use each permutation $\tilde\pi_i\in\mathfrak{S}_m$ to construct a permutation $\pi_i\in\mathfrak{S}_n$. The idea of the construction

is as follows: the permutation π_i is a product of m transpositions of distinct supports, and each transposition permutes an even integer with an odd one. We set $\pi_0 = id$ and for every i in $\{1, \dots, M\}$,

$$\pi_i = \big(1\ 2\tilde{\pi}_i(1)\big) \circ \big(3\ 2\tilde{\pi}_i(2)\big) \circ \ldots \circ \big(2m-1\ 2\tilde{\pi}_i(m)\big) \in \mathfrak{S}_n.$$

With these choices, the number of differences between π_i and π_j is exactly twice as much as the number of differences between $\tilde{\pi}_i$ and $\tilde{\pi}_j$. To sum up, for every pair of distinct indices $i, j \in \{0, \dots, M\}$,

$$\frac{1}{n}\sum_{k=1}^{n} \mathbb{1}_{\{\pi_i(k)\neq\pi_j(k)\}} = \frac{2m}{n} \times \frac{1}{m}\sum_{k=1}^{m} \mathbb{1}_{\{\tilde{\pi}_i(k)\neq\tilde{\pi}_j(k)\}} \geq \frac{m}{n} \geq \frac{3}{8}, \qquad \forall n \geq 4.$$

To complete the proof, we note that $m \geq n/3$.

Proof of Lemma 18 For every $\ell \in \{m, \dots, n\}$, counting all the permutations π such that $\sum_{k=1}^{n} \mathbb{1}_{\{\pi(k)\neq k\}} = \ell$, we get

$$\#E_{id} = !n + !(n-1)\binom{n}{1} + \ldots + !(n-m)\binom{n}{m},$$

where $!\ell$ is the number of derangements, the permutations such that none of the elements appear in their original position, in $\mathfrak{S}_\ell$ for $\ell \geq 1$. We know that

$$\forall\ \ell \geq 1, \qquad !\ell = \ell! \times \sum_{j=0}^{\ell} \frac{(-1)^j}{j!},$$

which, using the alternating series test, yields

$$\forall\ \ell \geq 1, \qquad !\ell \geq \ell! \times \Big(e^{-1} - \frac{1}{(\ell+1)!}\Big).$$

It follows that

$$\begin{aligned}
\#E_{id} &\geq n! \times \Big(e^{-1} - \frac{1}{(n-m+1)!}\Big) \times \Big(1 + \frac{1}{1!} + \ldots + \frac{1}{m!}\Big) \\
&\geq n! \times \Big(e^{-1} - \frac{1}{(n-m+1)!}\Big) \times \Big(e - \frac{e}{(m+1)!}\Big) \\
&\geq n! \times \Big(1 - \frac{e}{(n-m+1)!} - \frac{1}{(m+1)!}\Big).
\end{aligned}$$

Therefore,

$$\#E_{id}^{\complement} \leq n! \times \Big(\frac{e}{(n-m+1)!} + \frac{1}{(m+1)!}\Big) \leq \frac{4n!}{m!}.$$

Acknowledgments

This work was partially supported by the grants Investissements d'Avenir (ANR-11-IDEX-0003/Labex Ecodec/ANR-11-LABX-0047) and CALLISTO. The authors thank the Reviewers for many valuable suggestions. Special thanks to an anonymous Referee for pointing a mistake in the proof of Theorem 4.

References

R. R. Bahadur. On the asymptotic efficiency of tests and estimates. *Sankhyā*, 22:229–252, 1960.

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, jun 2008.

Eric Budish, Yeon-Koo Che, Fuhito Kojima, and Paul Milgrom. Designing random allocation mechanisms: Theory and applications. *American Economic Review*, 103(2):585–623, 2013.

Olivier Collier. Minimax hypothesis testing for curve registration. *Electron. J. Stat.*, 6: 1129–1154, 2012.

Olivier Collier and Arnak S. Dalalyan. Permutation estimation and minimax rates of identifiability. *Journal of Machine Learning Research*, W & CP 31 (AI-STATS 2013):10–19, 2013.

Olivier Collier and Arnak S. Dalalyan. Curve registration by nonparametric goodness-of-fit testing. *J. Statist. Plann. Inference*, 162:20–42, July 2015.

Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge, second edition, 2003.

Yu. I. Ingster and I. A. Suslina. *Nonparametric goodness-of-fit testing under Gaussian models*, volume 169 of *Lecture Notes in Statistics*. Springer-Verlag, New York, 2003.

Tony Jebara. Images as bags of pixels. In *ICCV*, pages 265–272. IEEE Computer Society, 2003.

A. P. Korostelev and V. G. Spokoiny. Exact asymptotics of minimax Bahadur risk in Lipschitz regression. *Statistics*, 28(1):13–24, 1996.

Alexander Korostelev. A minimaxity criterion in nonparametric regression based on large-deviations probabilities. *Ann. Statist.*, 24(3):1075–1083, 1996.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Res. Logist. Quart.*, 2:83–97, 1955.

B. Laurent and P. Massart. Adaptive estimation of a quadratic functional by model selection. *Ann. Statist.*, 28(5):1302–1338, 2000.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

David W. Pentico. Assignment problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774–793, 2007.

H. Edwin Romeijn and Dolores Romero Morales. A class of greedy algorithms for the generalized assignment problem. *Discrete Applied Mathematics*, 103(1-3):209–235, 2000.

V.G. Spokoiny. Adaptive hypothesis testing using wavelets. *The Annals of Statistics*, 24(6): 2477–2498, 1996.

Jos F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.*, 11/12(1-4):625–653, 1999.

Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

Alexandre B. Tsybakov. *Introduction to nonparametric estimation*. Springer Series in Statistics. Springer, New York, 2009.

Journal of Machine Learning Research 17 (2016) 1-33 Submitted 9/14; Revised 6/15; Published 3/16

Consistency and Fluctuations For Stochastic Gradient Langevin Dynamics

Yee Whye Teh Y.W.TEH@STATS.OX.AC.UK
Department of Statistics
University of Oxford
24-29 St Giles'
Oxford OX1 3LB
UK

Alexandre H. Thiery A.H.THIERY@NUS.EDU.SG
Department of Statistics and Applied Probability
National University of Singapore
21 Lower Kent Ridge Road
Singapore 119077

Sebastian J. Vollmer VOLLMER@STATS.OX.AC.UK
Department of Statistics
University of Oxford
24-29 St Giles'
Oxford OX1 3LB
UK

Editor: Nando de Freitas

Abstract

Applying standard Markov chain Monte Carlo (MCMC) algorithms to large data sets is computationally expensive. Both the calculation of the acceptance probability and the creation of informed proposals usually require an iteration through the whole data set. The recently proposed stochastic gradient Langevin dynamics (SGLD) method circumvents this problem by generating proposals which are only based on a subset of the data, by skipping the accept-reject step and by using decreasing step-sizes sequence $(\delta_m)_{m\geq 0}$.

We provide in this article a rigorous mathematical framework for analysing this algorithm. We prove that, under verifiable assumptions, the algorithm is consistent, satisfies a central limit theorem (CLT) and its asymptotic bias-variance decomposition can be characterized by an explicit functional of the step-sizes sequence $(\delta_m)_{m\geq 0}$. We leverage this analysis to give practical recommendations for the notoriously difficult tuning of this algorithm: it is asymptotically optimal to use a step-size sequence of the type $\delta_m \asymp m^{-1/3}$, leading to an algorithm whose mean squared error (MSE) decreases at rate $\mathcal{O}(m^{-1/3})$.

Keywords: Markov chain Monte Carlo, Langevin dynamics, big data

1. Introduction

We are entering the age of Big Data, where significant advances across a range of scientific, engineering and societal pursuits hinge upon the gain in understanding derived from the analyses of large scale data sets. Examples include recent advances in genome-wide association studies (Hirschhorn and Daly, 2005; McCarthy et al., 2008; Wang et al., 2005), speech recognition (Hinton et al., 2012), object recognition (Krizhevsky et al., 2012), and self-driving cars (Thrun, 2010). As the quantity of data available has been outpacing the computational resources available in recent years, there is an increasing demand for new scalable learning methods, for example methods based on stochastic optimization (Robbins and Monro, 1951a; Srebro and Tewari, 2010; Sato, 2001; Hoffman et al., 2010), distributed computational architectures (Ahmed et al., 2012; Neiswanger et al., 2013; Minsker et al., 2014), greedy optimization (Harchaoui and Jaggi, 2014), as well as the development of specialized computing systems supporting large scale machine learning applications (Gonzalez, 2014).

Recently, there has also been increasing interest in methods for Bayesian inference scalable to Big Data settings. Rather than attempting a single point estimate of parameters typical in optimization-based or maximum likelihood settings, Bayesian methods attempt to obtain characterizations of the full posterior distribution over the unknown parameters and latent variables in the model, hence providing better characterizations of the uncertainties inherent in the learning process, as well as providing protection against overfitting. Scalable Bayesian methods proposed in the recent literature include stochastic variational inference (Sato, 2001; Hoffman et al., 2010), which applies stochastic approximation techniques to optimizing a variational approximation to the posterior, parallelized Monte Carlo (Neiswanger et al., 2013; Minsker et al., 2014), which distributes the computations needed for Monte Carlo sampling across a large compute cluster, as well as subsampling-based Monte Carlo (Welling and Teh, 2011; Ahn et al., 2012; Korattikara et al., 2014), which attempt to reduce the computational complexity of Markov chain Monte Carlo (MCMC) methods by applying updates to small subsets of data.

In this paper we study the asymptotic properties of the stochastic gradient Langevin dynamics (SGLD) algorithm first proposed by Welling and Teh (2011). SGLD is a subsampling-based MCMC algorithm based on combining ideas from stochastic optimization, specifically using small subsets of data to estimate gradients, with Langevin dynamics, a MCMC method making use of gradient information to produce better parameter updates. Welling and Teh (2011) demonstrated that SGLD works well on a variety of models and this has since been extended by Ahn et al. (2012, 2014) and Patterson and Teh (2013b).

The stochastic gradients in SGLD introduce approximations into the Markov chain, whose effect has to be controlled by using a slowly decreasing sequence of step sizes. Welling and Teh (2011) provided an intuitive argument that as the step-size decreases the variations introduced by the stochastic gradients gets dominated by the natural stochasticity of Langevin dynamics, the result being that the stochastic gradient approximation should wash out asymptotically and that the Markov chain should converge to the true posterior distribution.

In this paper, we make this intuitive argument more precise by providing conditions under which SGLD converges to the targeted posterior distribution; we describe a number of characterizations of this convergence. Specifically, we show that estimators derived from SGLD are consistent (Theorem 7) and satisfy a central limit theorem (CLT) (Theorem 8); the bias-variance trade-off of the algorithm is discussed in details in Section 5. In Section 6 we prove that, when observed on the right (inhomogeneous) time scale, the sample path of the algorithm converges to a Langevin diffusion (Theorem 9).

Our analysis reveals that for a sequence of step-sizes with algebraic decay $\delta_m \asymp m^{-\alpha}$ the optimal choice, when measured in terms of rate of decay of the mean squared error (MSE), is given for $\alpha_\star = 1/3$; the choice $\delta_m \asymp m^{-\alpha_\star}$ leads to an algorithm that converges at rate $\mathcal{O}(m^{-1/3})$. This rate of convergence is worse than the standard Monte-Carlo $m^{-1/2}$-rate of convergence. This is not due to the stochastic gradients used in SGLD, but rather to the decreasing step-sizes.

These results are asymptotic in the sense that they characterise the behaviour of the algorithm as the number of steps approaches infinity. Therefore they do not necessarily translate into any insight into the behaviour for finite computational budgets which is the regime in which the SGLD might provide computational gains over alternatives. The mathematical framework described in this article show that the SGLD is a sound algorithm, an important result that has been missing in the literature.

In the remainder of this article, the notation $\mathrm{N}(\mu, \sigma^2)$ denotes a Gaussian distribution with mean μ and variance σ^2. For two positive functions $f, g : \mathbb{R} \to [0, \infty)$, one writes $f \lesssim g$ to indicate that there exists a positive constant $C > 0$ such that $f(\theta) \leq C\, g(\theta)$; we write $f \asymp g$ if $f \lesssim g \lesssim f$. For a probability measure π on a measured space $\mathcal{X}$, a measurable function $\varphi : \mathcal{X} \to \mathbb{R}$ and a measurable set $A \subset \mathcal{X}$, we define $\pi(\varphi; A) = \int_{\theta \in A} \varphi(\theta)\, \pi(d\theta)$ and $\pi(\varphi) = \pi(\varphi; \mathcal{X})$. Finally, densities of probability distributions on $\mathbb{R}^d$ are implicitly assumed to be defined with respect to the usual d-dimensional Lebesgue measure.

2. Stochastic Gradient Langevin Dynamics

Many MCMC algorithms evolving in a continuous state space, say $\mathbb{R}^d$, can be realised as discretizations of a continuous time Markov process $(\theta_t)_{t \geq 0}$. An example of such a continuous time process, which is central to SGLD as well as many other algorithms, is the Langevin diffusion, which is given by the stochastic differential equation

$$d\theta_t = \frac{1}{2} \nabla \log \pi(\theta_t)\, dt + dW_t, \tag{1}$$

where $\pi : \mathbb{R}^d \to (0, \infty)$ is a probability density and $(W_t)_{t \geq 0}$ is a standard Brownian motion in $\mathbb{R}^d$. The linear operator $\mathcal{A}$ denotes the generator of the Langevin diffusion (1): for a twice continuously differentiable test function $\varphi : \mathbb{R}^d \to \mathbb{R}$,

$$\mathcal{A}\varphi(\theta) = \frac{1}{2} \langle \nabla \log \pi(\theta), \nabla \varphi(\theta) \rangle + \frac{1}{2} \Delta \varphi(\theta), \tag{2}$$

where $\Delta\varphi \stackrel{\text{def}}{=} \sum_{i=1}^d \nabla_i^2 \varphi$ denotes the standard Laplacian operator. The motivation behind the choice of Langevin diffusions is that, under certain conditions, they are ergodic with respect to the distribution π; for example, (Roberts and Tweedie, 1996; Stramer and Tweedie, 1999a,b; Mattingly et al., 2002) describe drift conditions of the type described in Section 3.2 that ensure that the total variation distance from stationarity of the law at time t of the Langevin diffusion (1) decreases to zero exponentially quickly as $t \to \infty$.

Given a time-step $\delta > 0$ and a current position θ_t, it is often straightforward to simulate a random variable $\theta_\star$ that is approximately distributed as the law of $\theta_{t+\delta}$ given θ_t. For

stochastic differential equations, the Euler-Maruyama scheme (Maruyama, 1955) might be the simplest approach for approximating the law of $\theta_{t+\delta}$. For a Langevin diffusion this reads

$$\theta_\star = \theta_t + \frac{1}{2} \delta \nabla \log \pi(\theta_t) + \delta^{1/2}\, \eta \tag{3}$$

for a standard d-dimensional centred Gaussian random variable η. To fully correct the discretization error, one can adopt a Metropolis-Hastings accept-reject mechanism. The resulting algorithm is usually referred to as the Metropolis-Adjusted-Langevin algorithm (MALA) (Roberts and Tweedie, 1996). Other discretizations can be used as proposals. For example, the random walk Metropolis-Hastings algorithm uses the discretization of a standard Brownian motion as the proposal, while the Hamiltonian Monte Carlo (HMC) algorithm (Duane et al., 1987) is based on discretizations of an Hamiltonian system of differential equations. See the excellent review of Neal (2010) for further information.

In this paper, we shall consider the situation where the target π is the density of the posterior distribution under a Bayesian model where there are $N \gg 1$ i.i.d. observations, the so called Big Data regime,

$$\pi(\theta) \propto \mathrm{p}_0(\theta) \prod_{i=1}^N \mathrm{p}\left(y_i \,|\, \theta\right). \tag{4}$$

Here, both computing the gradient term $\nabla \log \pi(\theta_t)$ and evaluating the Metropolis-Hastings acceptance ratio require a computational budget that scales unfeasibly as $\mathcal{O}(N)$. One approach is to use a standard random walk proposal instead of Langevin dynamics, and to efficiently approximating the Metropolis-Hastings accept-reject mechanism using only a subset of the data (Korattikara et al., 2014; Bardenet et al., 2014).

This paper is concerned with stochastic gradient Langevin dynamics (SGLD), an alternative approach proposed by Welling and Teh (2011). This follows the opposite route and chooses to completely avoid the computation of the Metropolis-Hastings ratio. By choosing a discretization of the Langevin diffusion (1) with a sufficiently small step-size $\delta \ll 1$, because the Langevin diffusion is ergodic with respect to π, the hope is that even if the Metropolis-Hastings accept-reject mechanism is completely avoided, the resulting Markov chain still has an invariant distribution that is close to π. Choosing a decreasing sequence of step-sizes $\delta_m \to 0$ should even allow us to converge to the exact posterior distribution. To further make this approach viable in large N settings, the gradient term $\nabla \log \pi(\theta)$ can be further approximated using a subsampling strategy. For an integer $1 \leq n \leq N$ and a random subset $\tau \stackrel{\text{def}}{=} (\tau_1, \ldots, \tau_n)$ of $[N] \equiv \{1, \ldots, N\}$ generated by sampling with or without replacement from $[N]$, the quantity

$$\nabla \log \mathrm{p}_0(\theta) + \frac{N}{n} \sum_{i=1}^n \nabla \log \mathrm{p}(x_{\tau_i} \mid \theta) \tag{5}$$

is an unbiased estimator of $\nabla \log \pi(\theta)$. Most importantly, this stochastic estimate can be computed with a computational budget that scales as $\mathcal{O}(n)$ with n potentially much smaller than N. Indeed, the larger the quotient n/N, the smaller the variance of this estimate.

Stochastic gradient methods have a long history in optimisation and machine learning and are especially relevant in the large dataset regime considered in this article (Robbins

and Monro, 1951b; Bottou, 2010; Hoffman et al., 2013). In this paper we will adopt a slightly more general framework and assume that one can compute an unbiased estimate $\widehat{\nabla \log \pi}(\theta, \mathcal{U})$ to the gradient $\nabla \log \pi(\theta)$, where $\mathcal{U}$ is an auxiliary random variable which contains all the randomness involved in constructing the estimate. Without loss of generality we may assume (although this is unnecessary) that $\mathcal{U}$ is uniform on $(0,1)$. The unbiasedness of the estimator $\widehat{\nabla \log \pi}(\theta, \mathcal{U})$ means that

$$\mathbf{E}\left[H(\theta, \mathcal{U})\right] = 0 \qquad \text{with} \qquad H(\theta, \mathcal{U}) \stackrel{\text{def}}{=} \widehat{\nabla \log \pi}(\theta, \mathcal{U}) - \nabla \log \pi(\theta). \tag{6}$$

In summary, the SGLD algorithm can be described as follows. For a sequence of asymptotically vanishing time-steps $(\delta_m)_{m \geq 0}$ and an initial parameter $\theta_0 \in \mathbb{R}^d$, if the current position is θ_{m-1}, the next position θ_m is defined though the recursion

$$\theta_m = \theta_{m-1} + \frac{1}{2} \delta_m \widehat{\nabla \log \pi}(\theta_{m-1}, \mathcal{U}_m) + \delta_m^{1/2} \eta_m \tag{7}$$

for an i.i.d. sequence $\eta_m \sim \mathrm{N}(0, I_d)$, and an independent and i.i.d. sequence $\mathcal{U}_m$ of auxiliary random variables. This is the equivalent of the Euler-Maruyama discretization (3) of the Langevin diffusion (1) with a decreasing sequence of step-sizes and a stochastic estimate to the gradient term. The analysis presented in this article assumes for simplicity that the initial position θ_0 of the algorithm is deterministic; in the simulation study of Section 7, the algorithms are started at the MAP estimator. Indeed, more general situations could be analysed with similar arguments at the cost of slightly less transparent proofs. Note that the process $(\theta_m)_{m \geq 0}$ is a non-homogeneous Markov chain, and many standard analysis techniques for homogeneous Markov chains do not apply.

For a test function $\varphi : \mathbb{R}^d \to \mathbb{R}$, the expectation of φ with respect to the posterior distribution π can be approximated by the weighted sum

$$\pi_m(\varphi) \stackrel{\text{def}}{=} \frac{\delta_1 \varphi(\theta_0) + \ldots + \delta_m \varphi(\theta_{m-1})}{T_m} \tag{8}$$

with $T_m = \delta_1 + \ldots + \delta_m$. The quantity $\pi_m(\varphi)$ thus approximates the ergodic average $T_m^{-1} \int_0^{T_m} \varphi(\theta_t)\, dt$ between time zero and $t = T_m$. During the course of the proof of our fluctuation Theorem 8, we will need to consider more general averaging schemes than the one above. Instead, for a general positive sequence of weights $\omega = (\omega_m)_{m \geq 1}$, we define the ω-weighted sum

$$\pi_m^{\omega}(\varphi) \stackrel{\text{def}}{=} \frac{\omega_1 \varphi(\theta_0) + \ldots + \omega_m \varphi(\theta_{m-1})}{\Omega_m} \tag{9}$$

with $\Omega_m \stackrel{\text{def}}{=} \omega_1 + \ldots + \omega_m$. Indeed, $\pi_m^{\omega}(\varphi) = \pi_m(\varphi)$ in the particular case $(\omega_m)_{m \geq 1} = (\delta_m)_{m \geq 1}$; we will consider the weight sequence $\omega = \{\delta_m^2\}_{m \geq 1}$ in the proof of Theorem 8.

Let us mention several directions that can be explored to improve upon the basic SGLD algorithm explored in this paper. Langevin diffusions of the type $d\theta_t = \mathrm{drift}(\theta_t)\, dt + M(\theta_t)\, dW_t$, reversible with respect to the posterior distribution π, can be constructed for various choices of positive definite volatility matrix function $M : \mathbb{R}^d \to \mathbb{R}^{d,d}$. Note nonetheless that, for a non-constant volatility matrix function $\theta \mapsto M(\theta)$, the drift term typically

involves derivatives of M. Concepts of information geometry (Amari and Nagaoka, 2007) give principled ways (Livingstone and Girolami, 2014) of choosing the volatility matrix function M; when the Fisher information matrix is used, this leads to the Riemannian manifold MALA algorithm (Girolami and Calderhead, 2011). This approach has recently been applied to the Latent Dirichlet Allocation model for topic modelling (Patterson and Teh, 2013a). For high-dimensional state spaces $d \gg 1$, one can use a constant volatility function M, also known in this case as the preconditioning matrix, for taking into account the information contained in the prior distribution p_0 in the hope of obtaining better mixing properties (Beskos et al., 2008; Cotter et al., 2013); infinite dimensional limits are obtained in (Pillai et al., 2012; Hairer et al., 2014). Under an uniform-ellipticity condition and a growth assumption on the volatility matrix function $M : \mathbb{R}^d \to \mathbb{R}^{d,d}$, we believe that our framework could, at the cost of increasing complexity in the proofs, be extended to this setting. To avoid the slow random walk behaviour of Markov chains based on discretization of reversible diffusion processes, one can use instead discretizations of an Hamiltonian system of ordinary differential equations (Duane et al., 1987; Neal, 2010); when coupled with the stochastic estimates to the gradient above described, this leads to the stochastic gradient Hamiltonian Monte Carlo algorithm of (Chen et al., 2014).

In the rest of this paper, we will build a rigorous framework for understanding the properties of this SGLD algorithm, demonstrating that the heuristics and numerical evidences presented in Welling and Teh (2011) were indeed correct.

3. Assumptions and Stability Analysis

This section starts with the basics assumptions we will need for the asymptotic results to follow, and illustrates some of the potential stability issues that may occur, would the SGLD algorithm be applied without care.

3.1 Basic Assumptions

Throughout this text, we assume that the sequence of step-sizes $\delta = (\delta_m)_{m \geq 1}$ satisfies the following usual assumption.

Assumption 1 *The step-sizes $\delta = (\delta_m)_{m \geq 1}$ form a decreasing sequence with*

$$\lim_{m \to \infty} \delta_m = 0 \qquad \textit{and} \qquad \lim_{m \to \infty} T_m = \infty.$$

Indeed, this assumption is easily seen to also be necessary for the Law of Large Numbers of Section 4 to hold. Furthermore, we will need at several occasions to assume the following assumption on the oscillations of a sequence of step-sizes $(\omega_m)_{m \geq 1}$.

Assumption 2 *The step-sizes sequence $(\omega_m)_{m \geq 1}$ is such that $\omega_m \to 0$ and $\Omega_m \to \infty$ and*

$$\lim_{m \to \infty} \sum_{m \geq 1} \left| \Delta(\omega_m / \delta_m) \right| / \Omega_m < \infty \qquad \textit{and} \qquad \sum_{m \geq 1} \omega_m^2 / [\delta_m \Omega_m^2] < \infty.$$

where $\Delta(\omega_m/\delta_m) \stackrel{\text{def}}{=} \omega_{m+1}/\delta_{m+1} - \omega_m/\delta_m$.

Remark 3 *Assumption 2 holds if $\delta = (\delta_m)_{m\geq 1}$ satisfies Assumption (1) and the weights are defined as $\omega_m = \delta_m^p$, for some some exponent $p \geq 1$ small enough for $\Omega_m \to \infty$. This is because the first sum is less than $\sum_{m\geq 1} \left|\Delta(\omega_m/\delta_m)\right|/\Omega_1 = \delta_1^{p-1}/\Omega_1$, while the finiteness of the second sum can be seen as follows:*

$$\begin{aligned}\sum_{m\geq 1} \omega_m^2/\left(\delta_m \Omega_m^2\right) &\lesssim 1 + \sum_{m\geq 2} \left(\omega_m/\delta_m\right)^2 \left(1/\Omega_{m-1} - 1/\Omega_m\right) \\ &\lesssim 1 + \sum_{m\geq 2} \left(1/\Omega_{m-1} - 1/\Omega_m\right) = 1 + 1/\Omega_1.\end{aligned}$$

For any exponents $0 < \alpha < 1$ and $0 < p < 1/\alpha$ the sequences $\delta_m = (m_0 + m)^{-\alpha}$ and $\omega_m = \delta_m^p$ satisfy both Assumption 1 and Assumption 2.

3.2 Stability

Under assumptions on the tails of the posterior density π, the Langevin diffusion (1) is non-explosive and for any starting position $\theta_0 \in \mathbb{R}^d$ the total-variation distance $d_{\mathrm{TV}}\big(\mathbf{P}(\theta_t \in \cdot), \pi\big)$ converges to zero as $t \to \infty$. For instance, Theorem 2.1 of (Roberts and Tweedie, 1996) shows that it is sufficient to assume that the drift term satisfies the condition $(1/2)\left\langle \nabla \log \pi(\theta), \theta \right\rangle \leq \alpha\|\theta\|^2 + \beta$ for some constants $\alpha, \beta > 0$. We refer the interested reader to (Roberts and Tweedie, 1996; Stramer and Tweedie, 1999a,b; Roberts and Stramer, 2002; Mattingly et al., 2002) for a detailed study of the convergence properties of the Langevin diffusion (1).

Unfortunately, stability of the continuous time Langevin diffusion does not always translate into good behaviour for its Euler-Maruyama discretization. For example, even if the drift term points towards the right direction in the sense that $\langle \nabla \log \pi(\theta), \theta \rangle < 0$ for every parameter θ, it might happen that the magnitude of the drift term is too large so that the Euler-Maruyama discretization *overshoots* and becomes unstable. In a one dimensional setting, this would lead to a Markov chain that diverges in the sense that the sequence $(\theta_m)_{m\geq 0}$ alternates between taking arbitrarily large positive and negative values. Lemma 6.3 of (Mattingly et al., 2002) gives such an example with a target density $\pi(\theta) \propto \exp\{-\theta^4\}$. See also Theorem 3.2 of (Roberts and Tweedie, 1996) for examples of the same flavours.

Guaranteeing stability of the Euler-Maruyama discretization requires stronger Lyapunov type conditions. At a heuristic level, one must ensure that the drift term $\nabla \log \pi(\theta)$ points towards the centre of the state space. In addition, the previous discussion indicates that one must also ensure that the magnitude of this drift term is not too large. The following assumptions satisfy both heuristics, and we will show are enough to guarantee that the SGLD algorithm is consistent, with asymptotically Gaussian fluctuations.

Assumption 4 *The drift term $\theta \mapsto \frac{1}{2}\nabla \log \pi(\theta)$ is continuous. There exists a Lyapunov function $V : \mathbb{R}^d \to [1, \infty)$ that tends to infinity as $\|\theta\| \to \infty$, is twice differentiable with bounded second derivatives, and satisfies the following conditions.*

1. *There exists an exponent $p_H \geq 2$ such that*

$$\mathbf{E}\left[\left\|H(\theta, \mathcal{U})\right\|^{2p_H}\right] \lesssim V^{p_H}(\theta). \tag{10}$$

This implies that $\mathbf{E}\left[\left\|H(\theta,\mathcal{U})\right\|^{2p}\right] \lesssim V^p(\theta)$ for any exponent $0 \leq p \leq p_H$.

2. *For every $\theta \in \mathbb{R}^d$ we have*

$$\left\|\nabla V(\theta)\right\|^2 + \left\|\nabla \log \pi(\theta)\right\|^2 \lesssim V(\theta). \tag{11}$$

3. *There are constants $\alpha, \beta > 0$ such that for every $\theta \in \mathbb{R}^d$ we have*

$$\frac{1}{2}\left\langle \nabla V(\theta), \nabla \log \pi(\theta) \right\rangle \leq -\alpha V(\theta) + \beta. \tag{12}$$

Equation (12) ensures that on average the drift term $\widehat{\nabla \log \pi}(\theta)$ points towards the centre of the state space, while equations (10) and (11) provide control on the magnitude of the (stochastic) drift term. The drift condition (12) implies in particular that the Langevin diffusion (1) converges exponentially quickly towards the equilibrium distribution π (Mattingly et al., 2002; Roberts and Tweedie, 1996). The proof of the Law of Large Numbers (LLN) and the Central Limit Theorem (CLT) both exploit the following Lemma.

Lemma 5 (Stability) *Let the step-sizes $(\delta_m)_{m\geq 1}$ satisfy Assumption 1 and suppose that the stability Assumptions 4 hold. For any exponent $0 \leq p \leq p_H$ the following bounds hold almost surely,*

$$\sup_{m\geq 1} \pi_m(V^{p/2}) < \infty \qquad \text{and} \qquad \sup_{m\geq 1} \mathbf{E}\left[V^p(\theta_m)\right] < \infty. \tag{13}$$

Moreover, for any exponent $0 \leq p \leq p_H$ we have $\pi(V^p) < \infty$. If the sequence of weights $(\omega_m)_{m\geq 1}$ satisfies Assumption 2 the following holds almost surely,

$$\sup_{m\geq 1} \pi_m^{\omega}(V^{p/2}) < \infty \tag{14}$$

The technical proof can be found in Section B. The idea is to leverage condition (12) in order to establish that the function V^p satisfies both discrete and continuous drift conditions.

3.3 Scope of the Analysis

For a posterior density π of the form (4) and the usual unbiased estimate to $\nabla \log \pi$ described in Equation (5), to establish that Equations (10) and (11) hold it suffices to verify that the prior density p_0 is such that $\left\|\nabla \log \mathrm{p}_0(\theta)\right\|^2 \lesssim V(\theta)$ and that for any index $1 \leq i \leq N$ the likelihood term $\mathrm{p}(\, y_i \,|\, \theta)$ is such that

$$\left\|\nabla \log \mathrm{p}\left(\, y_i \,|\, \theta\right)\right\|^{2p_H} \lesssim V^{p_H}(\theta).$$

Indeed, in these circumstances, we have $\left\|H(\theta,\mathcal{U})\right\|^{2p_H} \lesssim \sum_{i=1}^N \left\|\nabla \log \mathrm{p}(y_i \mid \theta)\right\|^{2p_H}$. Several such examples are described in Section 7.

It is important to note that the drift Condition (12) typically does not hold for distributions with heavy tails such that $\nabla \log \pi(x) \to 0$ as $\|x\| \to \infty$ (Roberts and Tweedie, 1996). For example, the standard MALA algorithm is not geometrically ergodic when $\nabla \log \pi(x)$ converges to zero as $\|x\| \to \infty$ (Theorem 4.3 of (Roberts and Tweedie, 1996)); indeed, the analysis of standard local-move MCMC algorithms when applied to target densities with heavy tails is delicate and typically necessitate other tools Stramer and Tweedie (1999b);

Jarner and Roberts (2007); Kamatani (2014) than the approach based on drift conditions of the type (12). The analysis of the properties of the SGLD algorithm when applied to such heavy tail densities is out of the scope of this article. It is important to note that many more complex scenarios involving high-dimensionality, multi-modality, non-parametric settings where the complexity of the target distribution increases with the size of the data, or combination thereof, are examples of interesting and relevant situations where our analysis typically does not apply; analysing the SGLD algorithm when applied to these challenging target distributions is well out of the scope of this article.

4. Consistency

The problem of estimating the invariant distribution of a stochastic differential equation by using a diminishing step-size Euler discretization has been well explored in the literature (Lamberton and Pages, 2002, 2003; Lemaire, 2007; Panloup, 2008; Pages and Panloup, 2012), while (Mattingly et al., 2002) studied the bias and variance of similar algorithms when fixed step-sizes are used instead. We leverage some of these techniques and adapt it to our setting where the drift term can only be unbiasedly estimated, and establish in this section that the SGLD algorithm is consistent under Assumptions 1 and 4. More precisely, we prove that almost surely the sequence $(\pi_m)_{m\geq 1}$ defined in Equation (8) converges weakly towards π. Specifically, under growth assumptions on a test function $\varphi : \mathbb{R}^d \to \mathbb{R}$, the following strong law of large numbers holds almost surely,

$$\lim_{m\to\infty} \frac{\delta_1 \varphi(\theta_0) + \ldots + \delta_m \varphi(\theta_m)}{T_m} = \int_{\mathbb{R}^d} \varphi(\theta)\, \pi(d\theta),$$

with a similar result for ω-weighted empirical averages, under assumptions on the weight sequence ω. The proofs of several results of this paper make use of the following elementary lemma.

Lemma 6 *Let $(\Delta M_k)_{k\geq 0}$ and $(R_k)_{k\geq 0}$ be two sequences of random variables adapted to a filtration $(\mathcal{F}_k)_{k\geq 0}$ and let $(\Gamma_k)_{k\geq 0}$ be an increasing sequence of positive real numbers. The limit*

$$\lim_{m\to\infty} \frac{\sum_{k=0}^{m} \Delta M_k + R_k}{T_m} = 0 \tag{15}$$

holds almost surely if the following two conditions are satisfied.

1. *The process $M_m = \sum_{k\leq m} \Delta M_k$ is a martingale, i.e. $\mathbf{E}[\,\Delta M_k \,|\, \mathcal{F}_k] = 0$ and*

$$\lim_{k\to\infty} \sum_{k\geq 0} \frac{\mathbf{E}\left[|\Delta M_k|\right]^2}{T_k^2} < \infty. \tag{16}$$

2. *The sequence $(R_k)_{k\geq 0}$ is such that*

$$\lim_{k\to\infty} \sum_{k\geq 0} \frac{\mathbf{E}\left[|R_k|\right]}{T_k} < \infty. \tag{17}$$

The above lemma, whose proof can be found in the appendix A, is standard; Lamberton and Pages (2002) also follows this route to prove several of their results.

Theorem 7 *(Consistency) Let the step-sizes satisfy Assumption* (1) *and suppose that the stability Assumptions 4 hold for a Lyapunov function $V : \mathbb{R}^d \to [1,\infty)$. Let $0 \leq p < p_H/2$ and $\varphi : \mathbb{R}^d \to \mathbb{R}$ be a test function such that $|\varphi(\theta)|/V^p(\theta)$ is globally bounded. Then the following limit holds almost surely:*

$$\lim_{m\to\infty} \pi_m(\varphi) = \pi(\varphi). \tag{18}$$

If in addition the sequence of weights $\{\omega_m\}_{m\geq 1}$ satisfies Assumption (2)*, a similar result holds almost surely for the ω-weighted ergodic average:*

$$\lim_{m\to\infty} \pi_m^{\omega}(\varphi) = \pi(\varphi). \tag{19}$$

Proof In the following, we write $\mathbf{E}_k[\,\cdot\,]$ and $\mathbf{P}_k(\,\cdot\,)$ to denote the conditional expectation $\mathbf{E}[\,\cdot\,|\,\theta_k]$ and conditional probability $\mathbf{P}(\,\cdot\,|\,\theta_k)$ respectively. We use the notation $\Delta\theta_k \stackrel{\text{def}}{=} (\theta_{k+1} - \theta_k)$. Finally, for notational convenience, we only present the proof in the scalar case $d = 1$, the multidimensional case being entirely similar. We will give a detailed proof of Equation (18) and then briefly describe how the more general Equation (19) can be proven using similar arguments. To prove Equation (18), we first show that the sequence $(\pi_m)_{m\geq 1}$ almost surely converges weakly to π. Equation (18) is then proved in a second stage.

Weak convergence of $(\pi_m)_{m\geq 1}$. To prove that almost surely the sequence $(\pi_m)_{m\geq 1}$ converges weakly towards π it suffices to prove that the sequence is almost surely weakly pre-compact and that any weakly convergent subsequence of $(\pi_m)_{m\geq 0}$ necessarily (weakly) converges towards π. By Prokhorov's Theorem (Billingsley, 1995) and Equation (13), because the Lyapunov function V goes to infinity as $\|\theta\| \to \infty$, the sequence $(\pi_m)_{m\geq 1}$ is almost surely weakly pre-compact. It thus remains to show that if a subsequence converges weakly to a probability measure π_∞ then $\pi_\infty = \pi$.

Since the Langevin diffusion (1) has a unique strong solution and its generator $\mathcal{A}$ is uniformly elliptic, Theorem 9.17 of Chapter 4 of (Ethier and Kurtz, 1986) yields that it suffices to verify that for any smooth and compactly supported test function $\varphi : \mathbb{R} \to \mathbb{R}$ and any limiting distribution π_∞ of the sequence $(\pi_m)_{m\geq 1}$ the following holds,

$$\pi_\infty(\mathcal{A}\varphi) = 0. \tag{20}$$

To prove Equation (20) we use the following decomposition of $\pi_m(\mathcal{A}\varphi)$,

$$\left\{\frac{\sum_{k=1}^{m} \mathbf{E}_{k-1}[\varphi(\theta_k) - \varphi(\theta_{k-1})]}{T_m}\right\} - \left\{\frac{\sum_{k=1}^{m} \mathbf{E}_{k-1}[\varphi(\theta_k) - \varphi(\theta_{k-1})]}{T_m} - \pi_m(\mathcal{A}\varphi)\right\}. \tag{21}$$

- Let us prove that the first term of (21) converges almost surely to zero. The numerator is equal to the sum of $\sum_{k=1}^{m} \mathbf{E}_{k-1}[\varphi(\theta_k)] - \varphi(\theta_k)$ and $\varphi(\theta_m) - \varphi(\theta_0)$. By boundedness of φ, the term $\{\varphi(\theta_m) - \varphi(\theta_0)\}/T_m$ converges almost surely to zero. By Lemma 6, to

conclude is suffices to show that the martingale difference terms $\mathbf{E}_{k-1}\left[\varphi(\theta_k)\right] - \varphi(\theta_k)$ are such that

$$\sum_{k\geq 1} \frac{\mathbf{E}\left[\left|\mathbf{E}_{k-1}\left[\varphi(\theta_k)\right] - \varphi(\theta_k)\right|^2\right]}{T_k^2} \; < \; \infty.$$

Because φ is Lipschitz, it suffices to prove that $\sum_{k\geq 1} \mathbf{E}\left(\|\theta_{k+1} - \theta_k\|^2\right)/T_k^2$ is finite. The stability Assumption 4 and Lemma 5 imply that the supremum $\sup_m \mathbf{E}\left[V(\theta_m)\right]$ is finite. Since $\mathbf{E}_k\left[\|\theta_{k+1} - \theta_k\|^2\right] \lesssim \delta_{k+1}^2\, V(\theta) + \delta_{k+1}$, it follows that $\mathbf{E}\left(\|\theta_{k+1} - \theta_k\|^2\right)$ is less than a constant multiple of δ_{k+1}. Under Assumption 1, because the telescoping sum $\sum_{k\geq 1} T^{-1}(k) - T^{-1}(k+1)$ is finite, the sum $\sum_{k\geq 1} \delta_k/T_k^2$ is finite. This concludes the proof that the first term in (21) converges almost surely to zero.

- The second term of (21) equals $\big(R_0 + \ldots + R_{m-1}\big)/T_m$ with

$$R_k \stackrel{\text{def}}{=} \mathbf{E}_k\left[\varphi(\theta_{k+1}) - \varphi(\theta_k)\right] - \mathcal{A}\varphi(\theta_k)\,\delta_{k+1}. \tag{22}$$

We now show that there exists a constant C such that the bound $|R_k| \leq C\,\delta_{k+1}^{3/2}$ holds for any $k \geq 0$. To do so, let $K > 0$ be such that the support of the test function φ is included in the compact set $\Omega = [-K, K]$. We examine two cases separately.

– If $|\theta_k| > K+1$ then $\varphi(\theta_k) = \mathcal{A}\varphi(\theta_k) = 0$ so that $|R_k| \leq \|\varphi\|_\infty \times \mathbf{P}_k(\theta_{k+1} \in \Omega)$. Since $\theta_{k+1} - \theta_k = \left\{\frac{1}{2}\nabla \log \pi(\theta_k) + H(\theta_k, \mathcal{U})\right\}\delta_{k+1} + \sqrt{\delta_{k+1}}\,\eta$ we have

$$\begin{aligned}\mathbf{P}_k(\theta_{k+1} \in \Omega) \leq\; & \mathbb{I}\left(\left|\frac{1}{2}\nabla \log \pi(\theta_k)\right| \geq \frac{\mathrm{dist}(\theta_k, \Omega)}{3\,\delta_{k+1}}\right) \\ & + \mathbf{P}_k\left(|H(\theta_k, \mathcal{U})| \geq \frac{\mathrm{dist}(\theta_k, \Omega)}{3\,\delta_{k+1}}\right) + \mathbf{P}_k\left(|\eta| \geq \frac{\mathrm{dist}(\theta_k, \Omega)}{3\,\sqrt{\delta_{k+1}}}\right).\end{aligned}$$

We have used the notation $\mathbb{I}(A)$ for denoting the indicator function of the event A. Under Assumption 4 we have $|\nabla \log \pi(\theta)| \lesssim V(\theta)^{1/2} \lesssim 1 + \|\theta\|$ so that the quotient $|\nabla \log \pi(\theta)|/\mathrm{dist}(\theta, \Omega)$ is bounded on the set $\{\theta : |\theta| > K\}$; this shows that the first term equals zero for δ_k small enough. To prove that the second term is bounded by a constant multiple of δ_{k+1}^2, it suffices to use Markov's inequality and the fact that $\mathbf{E}[H(\theta_k, \mathcal{U})^2]/\mathrm{dist}^2(\theta, \Omega)$ is bounded on $\{\theta : |\theta| > K\}$; this is because $\mathbf{E}[H(\theta_k, \mathcal{U})^2]$ is less than a constant multiple of $V(\theta)$ and $V(\theta) \lesssim 1 + \|\theta\|^2$ by Assumption 4. The third term is less than a constant multiple of δ_{k+1}^2 by Markov's inequality and the fact that η has a finite moment of order four.

– If $|\theta_k| \leq K+1$, we decompose R_k into two terms. A second order Taylor formula yields

$$\begin{aligned}R_k &= \frac{1}{2}\,\delta_{k+1}^2\,\varphi''(\theta_k)\left\{\left[\nabla \log \pi(\theta_k)\right]^2 + \mathbf{E}_k\left[H^2(\theta_k, \mathcal{U})\right]\right\} \\ &\quad + (1/2)\;\mathbf{E}_k\left[(\Delta\theta_k)^3 \int_0^1 \varphi'''(\theta_k + u\,\Delta\theta_k)\,(1-u)^2\,du\right] \\ &= R_{k,1} + R_{k,2}.\end{aligned}$$

Under Assumption 4, the quantities $\left[\nabla \log \pi(\theta_k)\right]^2$ and $\mathbf{E}[H^2(\theta_k, \mathcal{U})]$ are upper bounded by a constant multiple of $V(\theta_k)$. Since the function $\theta \mapsto \varphi''(\theta)\,V(\theta)$ is globally bounded (because continuous with compact support) this shows that $R_{k,1}$ is less than a constant multiple of δ_{k+1}^2. Since $|\theta_k| \leq K+1$, the bounds $\mathbf{E}[H^3(\theta, \mathcal{U})] \lesssim V^{3/2}(\theta)$ and $\sup_{k\geq 0} \mathbf{E}[V^{3/2}(\theta_k)] < \infty$ (see Lemma 5) yield that $\mathbf{E}_k\left|\Delta\theta_k\right|^3 \leq 9\,\overline{C}\,(\delta_{k+1}^3 + \delta_{k+1}^{3/2}) \lesssim \delta_{k+1}^{3/2}$ with

$$\overline{C} = 1 + \sup_{\theta:|\theta|<K+1} |\nabla \log \pi(\theta)|^3 + \mathbf{E}\left[|H(\theta, \mathcal{U})|^3\right].$$

Note that $\overline{C}$ is finite by Assumption 4 and Lemma 5.

We have thus proved that there is a constant C such $|R_k| \leq C\,\delta_{k+1}^{3/2}$ for $k \geq 0$; it follows that the sum $(R_0 + \ldots + R_{m-1})/T_m$ is less than a constant multiple of $\left(\delta_1^{3/2} + \ldots + \delta_m^{3/2}\right)/T_m$. Under Assumption 1, this upper bound converges to zero as $m \to \infty$, hence the conclusion.

This ends the proof of the almost sure weak convergence of π_m towards π.

Proof of Equation (18). By assumption we have $|\varphi(\theta)| \leq C_p\,V^p(\theta)$ for some constant $C_p > 0$ and exponent $p < p_H/2$. To show that $\pi_m(\varphi) \to \pi(\varphi)$ almost surely, we will use Lemma 5 and the almost sure weak convergence, which guarantees that $\pi_m(\widetilde{\varphi}) \to \pi(\widetilde{\varphi})$ for a continuous and bounded test function $\widetilde{\varphi}$.

For any $t > 0$, the set $\Omega_t \stackrel{\text{def}}{=} \{\theta : V(\theta) \leq t\}$ is compact and Tietze's extension theorem (Rudin, 1986, Theorem 20.4) yields that there exists a continuous function $\widetilde{\varphi}_t$ with compact support that agrees with φ on Ω_t and such that $\|\widetilde{\varphi}_t\|_\infty = \sup\{|\varphi(\theta)| : \theta \in \Omega_t\}$. We can indeed also assume that $|\widetilde{\varphi}_t(\theta)| \leq C_p\,V^p(\theta)$. Since Lemma 5 states that $\sup_m \pi_m(V^{p_H/2})$ is almost surely finite, it follows that

$$|\pi_m(\varphi) - \pi_m(\widetilde{\varphi}_t)| \leq 2\,C_p\,\pi_m(V^p\,\mathbb{1}_{V\geq t}) \leq 2\,C_p\,\frac{\sup_m \pi_m(V^{p_H/2})}{t^{p_H/2-p}},$$

where the last inequality follows from the fact that for any probability measure μ, exponents $0 < p < q$ and scalar $t > 0$ we have $\mu(V^p\,\mathbb{1}_{V\geq t}) \leq \mu(V^q\,\mathbb{1}_{V\geq t})/t^{q-p}$. Similarly

$$|\pi(\varphi) - \pi(\widetilde{\varphi}_t)| \leq 2\,C_p\,\pi(V^{p_H/2})/t^{p_H/2-p}.$$

By the triangle inequality, we thus have,

$$|\pi_m(\varphi) - \pi(\varphi)| \leq 2\,C_p\,\frac{\sup_m \pi_m(V^{p_H/2})}{t^{p_H/2-p}} + \left|\pi_m(\widetilde{\varphi}_t) - \pi(\widetilde{\varphi}_t)\right| + 2\,C_p\,\frac{\pi(V^{p_H/2})}{t^{p_H/2-p}}.$$

On the right-hand-side, the term in the middle can be made arbitrarily small as $m \to \infty$ since π_m converges weakly towards π, while the other two terms converges to zero as $t \to \infty$. This concludes the proof of Equation (18).

Proof of Equation (19). The approach is very similar to the proof of Equation (18) and for this reason we only highlight the main differences. The same argument shows that the

sequence π_m^ω is tight and it suffices to show that $\pi_\infty^\omega(\mathcal{A}\varphi) = 0$ for any weak limit π_∞^ω of the sequence $(\pi_m^\omega)_{m\geq 0}$ for obtaining the almost sure weak convergences of $(\pi_m^\omega)_{m\geq 0}$ towards π. One can then upgrade this almost sure weak convergence to a Law of Large Numbers. To prove (19), we thus concentrate on proving that $\pi_\infty^\omega(\mathcal{A}\varphi) = 0$. For a smooth and compactly supported test function φ we use the decomposition $\pi_m^\omega(\mathcal{A}\varphi) = S_1(m) + S_2(m) + S_3(m)$ with

$$\left\{\begin{array}{ll} S_1(m) & = \frac{1}{\Omega_m}\sum_{k=1}^m \frac{\omega_k}{\delta_k}\big(\mathbf{E}_{k-1}[\varphi(\theta_k)] - \varphi(\theta_k)\big) \\ S_2(m) & = \frac{1}{\Omega_m}\sum_{k=1}^m \frac{\omega_k}{\delta_k}\big(\varphi(\theta_k) - \varphi(\theta_{k-1})\big) \\ S_3(m) & = \pi_m^\omega(\mathcal{A}\varphi) - \frac{1}{\Omega_m}\sum_{k=1}^m \frac{\omega_k}{\delta_k}\mathbf{E}_{k-1}[\varphi(\theta_k) - \varphi(\theta_{k-1})] \end{array}\right.$$

and prove that each term converges to zero almost surely. For $S_1(m)$, by Lemma 6 it suffices to show that $\sum_{k\geq 1}(\omega_k/\delta_k)^2\,\mathbf{E}\left[\{\mathbf{E}_{k-1}[\varphi(\theta_k)] - \varphi(\theta_k)\}^2\right]/\Omega_k^2$ is finite. This follows from the bound $\mathbf{E}\left[\big(\mathbf{E}_{k-1}[\varphi(\theta_k)] - \varphi(\theta_k)\big)^2\right] \lesssim \delta_k$ and the fact that $\sum_{m\geq 0}\omega_m^2/(\Omega_m^2\,\delta_m)$ is finite. For $S_2(m)$, we can write it as

$$S_2(m) = \frac{-\frac{\omega_1}{\delta_1}\varphi(\theta_0) + \frac{\omega_{m+1}}{\delta_{m+1}}\varphi(\theta_m) - \sum_{k=1}^m \varphi(\theta_k)\,\Delta(\omega_k/\delta_k)}{\Omega_m}.$$

Because $\Omega_m \to \infty$, $(\omega_{m+1}/\delta_{m+1})/\Omega_m \to 0$ and φ is bounded, one can concentrate on proving that $\Omega_m^{-1}\sum_{k=1}^m \varphi(\theta_k)\,\Delta(\omega_k/\delta_k)$ converges almost surely to zero. By Lemma 6, it suffices to verify that $\sum_{k\geq 1}\mathbf{E}\left[|\varphi(\theta_k)\,\Delta(\omega_k/\delta_k)|\right]/\Omega_k$ is finite; this directly follows from the boundedness of φ and Assumption 2. Finally, algebra shows that $S_3(m) = \Omega_m^{-1}\sum_1^m(\omega_k/\delta_k)\,R_{k-1}$ with the quantity R_k defined in Equation (22). It has been proved that there is a constant C such that, almost surely, $|R_k| \leq C\,\delta_{k+1}^{3/2}$ for all $k \geq 0$. Since $\delta_m \to 0$, the rescaled sum $\Omega_m^{-1}\sum_{k\leq m}\omega_k\delta_k^{1/2}$ converges to zero as $m \to \infty$. It follows that $S_3(m)$ converges almost surely to zero. ■

5. Fluctuations, Bias-Variance Analysis, and Central Limit Theorem

The previous section shows that, under suitable conditions, for a test function $\varphi : \mathbb{R}^d \to \mathbb{R}$ the quantity $\pi_m(\varphi)$ converges almost surely to $\pi(\varphi)$ as $m \to \infty$. In this section, we investigate the fluctuations of $\pi_m(\varphi)$ around its asymptotic value $\pi(\varphi)$. We establish that the asymptotic bias-variance decomposition of the SGLD algorithm is dictated by the behaviour of the sequence

$$\mathbb{B}_m \stackrel{\text{def}}{=} T_m^{-1/2}\sum_{k=0}^{m-1}\delta_{k+1}^2. \tag{23}$$

Indeed, the proof of Theorem 8 reveals that the fluctuations of $\pi_m(\varphi)$ are of order $\mathcal{O}\left(T_m^{-1/2}\right)$ and its bias is of order $\mathcal{O}\left(T_m^{-1}\sum_{k=0}^{m-1}\delta_{k+1}^2\right)$; the quantity $\mathbb{B}_m$ is thus the ratio of the typical scales of the bias and fluctuations. In the case where $\mathbb{B}_m \to 0$, the fluctuations dominate

the bias and the rescaled difference $T_m^{1/2} \times (\pi_m(\varphi) - \pi(\varphi))$ converges weakly to a centred Gaussian distribution. In the case where $\mathbb{B}_m \to \mathbb{B}_\infty \in (0,\infty)$, there is an exact balance between the scale of the bias and the scale of the fluctuations; the rescaled quantity $T_m^{1/2} \times (\pi_m(\varphi) - \pi(\varphi))$ converges to a non-centred Gaussian distribution. Finally, in the case where $\mathbb{B}_m \to \infty$, the bias dominates and the rescaled quantity $\left(T_m^{-1}\sum_{k=1}^m \delta_k^2\right)^{-1} \times (\pi_m(\varphi) - \pi(\varphi))$ converges in probability to a quantity $\mu(\varphi) \in \mathbb{R}$ whose exact value is described in the sequel. The strategy of the proof is standard; the solution h of the Poisson equation

$$\varphi - \pi(\varphi) = \mathcal{A}h \tag{24}$$

is introduced so that the additive functional $\pi_m(\varphi)$ of the trajectory of the Markov process $\{\theta_k\}_{k\geq 0}$ can be expressed as the sum of a martingale and a remainder term. A central limit for martingales can then be invoked to describe the asymptotic behaviour of the fluctuations

Theorem 8 *(Fluctuations) Let the step-sizes $(\delta_m)_{m\geq 1}$ satisfy Assumption 1 and assume that Assumption 4 holds for an exponent $p_H \geq 5$. Let $\varphi : \mathbb{R}^d \to \mathbb{R}$ be a test function and assume that the unique solution $h : \mathbb{R}^d \to \mathbb{R}$ to the Poisson Equation (24) satisfies $\|\nabla^n h(\theta)\| \lesssim V^{p_H}(\theta)$ for $n \leq 4$ and has a bounded fifth derivative. Define $\sigma^2(\varphi) = \pi\Big(\|\nabla h\|^2\Big)$.*

- *In case the fluctuations dominate, i.e. $\mathbb{B}_m \to 0$, the following convergence in distribution holds,*
$$\lim_{m\to\infty} T_m^{1/2}\left\{\pi_m(\varphi) - \pi(\varphi)\right\} \;=\; \mathrm{N}\left(0, \sigma^2(\varphi)\right). \tag{25}$$
- *In case the fluctuations and the bias are on the same scale, i.e. $\mathbb{B}_m \to \mathbb{B}_\infty \in (0,\infty)$, the following convergence in distribution holds,*
$$\lim_{m\to\infty} T_m^{1/2}\left\{\pi_m(\varphi) - \pi(\varphi)\right\} \;=\; \mathrm{N}\left(\mu(\varphi), \sigma^2(\varphi)\right), \tag{26}$$
with the asymptotic bias
$$\mu(\varphi) = -\mathbb{B}_\infty\,\mathbf{E}\left[\frac{1}{8}\nabla^2 h(\Theta)\widehat{\nabla\log\pi}(\Theta,\mathcal{U})^2 + \frac{1}{4}\nabla^3 h(\Theta)\nabla\log\pi(\Theta) + \frac{1}{24}\nabla^4 h(\Theta)\right]$$
where the random variables $\Theta \overset{\mathcal{D}}{\sim} \pi$ and $\mathcal{U}$ are independent.
- *In case the bias dominates, i.e. $\mathbb{B}_m \to \infty$, the following limit holds in probability,*
$$\lim_{m\to\infty} \frac{\pi_m(\varphi) - \pi(\varphi)}{T_m^{-1}\sum_{k=1}^m \delta_k^2} \;=\; \mu(\varphi). \tag{27}$$

Proof The proof follows the strategy described in Lamberton and Pages (2002), with the additional difficulty that only unbiased estimates of the drift term of the Langevin diffusion are available. We use the decomposition

$$\pi_m(\varphi) - \pi(\varphi) = \left\{\frac{\sum_{k=0}^{m-1}\delta_{k+1}\,\mathcal{A}h(\theta_k) - (h(\theta_{k+1}) - h(\theta_k))}{T_m}\right\} + \left\{\frac{h(\theta_m) - h(\theta_0)}{T_m}\right\}. \tag{28}$$

A fifth order Taylor expansion and Equation (7) yields that

$$h(\theta_{k+1}) - h(\theta_k) = \sum_{n=1}^{4} \left\{ \sum_{i=0}^{n} \mathcal{C}_{n,i}^{(k)} \, \delta_{k+1}^{(n+i)/2} \right\} + \nabla^5 h(\xi_k) \, (\theta_{k+1} - \theta_k)^5 / 5!. \tag{29}$$

In the above, we have defined $\mathcal{C}_{n,i}^{(k)} \equiv \left(2^i \, i! \, (n-i)!\right)^{-1} \nabla^n h(\theta_k) \widehat{\nabla \log \pi}(\theta_k, \mathcal{U}_{k+1})^i \eta_{k+1}^{n-i}$; the quantity ξ_k lies between θ_k and θ_{k+1}. It follows from the expression (2) of the generator of the $\mathcal{A}$ of the Langevin diffusion (1) and decomposition (28) that $\pi_m(\varphi) - \pi(\varphi) = \mathscr{F}_m + \mathscr{B}_m + \mathscr{R}_m$ where the fluctuation and bias terms are given by

$$\mathscr{F}_m \equiv -\frac{1}{T_m} \sum_{k=0}^{m-1} \mathcal{C}_{1,0}^{(k)} \delta_{k+1}^{1/2} \quad \text{and} \quad \mathscr{B}_m \equiv -\frac{1}{T_m} \sum_{k=0}^{m-1} \left\{ \mathcal{C}_{2,2}^{(k)} + \mathcal{C}_{3,1}^{(k)} + \mathcal{C}_{4,0}^{(k)} \right\} \delta_{k+1}^2$$

while the remainder term reads

$$\begin{aligned} \mathscr{R}_m \equiv & -\frac{1}{T_m} \sum_{k=0}^{m-1} \left\{ \frac{1}{2} H(\theta_k, \mathcal{U}_{k+1}) \, \nabla h(\theta_k) + \frac{1}{2} \left(\eta_{k+1}^2 - 1 \right) \nabla^2 h(\theta_k) \right\} \delta_{k+1} \\ & - \frac{1}{T_m} \sum_{k=0}^{m-1} \left\{ \sum_{(n,i) \in \mathcal{I}_{\mathscr{R}}} \mathcal{C}_{n,i}^{(k)} \, \delta_{k+1}^{(n+i)/2} \right\} - \frac{1}{T_m} \sum_{k=0}^{m-1} \nabla^5 h(\xi_k) \, (\theta_{k+1} - \theta_k)^5 / 5! \\ & + \left\{ \frac{h(\theta_m) - h(\theta_0)}{T_m} \right\} \end{aligned} \tag{30}$$

for $\mathcal{I}_{\mathscr{R}} = \bigcup_{p \in \{3,5,6,7,8\}} \mathcal{I}_{\mathscr{R},p}$ and $\mathcal{I}_{\mathscr{R},p} \equiv \{(n,i) \in [1:4] \times [0:4] \, : \, i \leq n, \, i+n = p\}$. We will show that the remainder term is negligible in the sense that each term on the R.H.S of Equation (30), when multiplied by either $T_m^{1/2}$ or $T_m(\sum_{k=0}^{m-1} \delta_{k+1}^2)^{-1}$, converges in probability to zero; in other words, each one of these terms is dominated asymptotically by either the fluctuations or the bias and is thus negligible. We then show that when multiplied by $T_m^{1/2}$, the fluctuation term converges in distribution to $\mathrm{N}(0, \sigma^2(\varphi))$. Finally, we show that the bias term converge to $\mu(\varphi)$ when rescaled by its typical scale, $T_m(\sum_{k=0}^{m-1} \delta_{k+1}^2)^{-1}$. Putting these results together under the three cases of $\mathbb{B}_m \to 0$, $\mathbb{B}_m \to \mathbb{B}_\infty \in (0, \infty)$ and $\mathbb{B}_m \to \infty$ leads to the results of the Theorem.

Remainder Term

we start by proving that the term $\mathscr{R}_m$ is negligible. The term $\{h(\theta_m) - h(\theta_0)\}/T_m^{1/2}$ converges to zero in probability because $|h(\theta)| \lesssim V^{p_H}(\theta)$ and Lemma 5 shows that

$$\sup_{m \geq 0} \mathbf{E}[V^{p_H}(\theta_m)] < \infty a.s..$$

Similarly, Assumptions 1 and 4 and Lemma 5 yield that

$$\mathbf{E}\left[\nabla^5 h(\xi_k) \, (\theta_{k+1} - \theta_k)^5 \right] \lesssim \mathbf{E}\left[|\eta_{k+1}|^5 \right] \delta_{k+1}^{5/2} + \mathbf{E}\left[\left| \widehat{\nabla \log \pi}(\theta_k, \mathcal{U}_{k+1}) \right|^5 \right] \delta_{k+1}^5 \lesssim \delta_{k+1}^{5/2}$$

from which it follows that $\left\{ \sum_{k=0}^{m-1} \nabla^5 h(\xi_k) \, (\theta_{k+1} - \theta_k)^5 \right\} / \left\{ \sum_{k=0}^{m-1} \delta_{k+1}^2 \right\}$ converges to zero in probability; we have exploited the fact that $\nabla^5 h$ is assumed to be globally bounded. Essentially the same argument yield that the high-order terms are asymptotically negligible: for $(n,i) \in \mathcal{I}_{\mathscr{R},p}$ and $p \in \{5,6,7,8\}$ the limit

$$\lim_{m \to \infty} \frac{\sum_{k=0}^{m-1} \mathcal{C}_{n,i}^{(k)} \, \delta_{k+1}^{(n+i)/2}}{\sum_{k=0}^{m-1} \delta_{k+1}^2} = 0$$

holds in probability because the coefficients $\mathcal{C}_{n,i}^{(k)}$ are uniformly bounded in expectation and the quantity $\left(\sum_{k=0}^{m-1} \delta_{k+1}^{(n+i)/2} \right) / \left(\sum_{k=0}^{m-1} \delta_{k+1}^2 \right)$ converges to zero since $(n+i)/2 \geq 5/2$ and $\delta_k \to 0$. To conclude, one needs to verify that the low order terms are also negligible in the sense that the limit

$$\lim_{m \to \infty} \frac{\sum_{k=0}^{m-1} X_{n,i}^{(k)} \delta_{k+1}^{(n+i)/2}}{T_m^{1/2}} = 0$$

holds in probability with $X_{1,1}^{(k)} = \nabla h(\theta_k) \, H(\theta_k, \mathcal{U}_{k+1})$ and $X_{2,0}^{(k)} = \nabla^2 h(\theta_k)(\eta_{k+1}^2 - 1)$ and $X_{2,1}^{(k)} = -\mathcal{C}_{2,1}^{(k)}$ and $X_{3,0}^{(k)} = -\mathcal{C}_{3,0}^{(k)}$. Since $\mathbf{E}\left[X_{n,i}^{(k)} \middle| \mathcal{F}_k \right] = 0$ where $\mathcal{F}_k = \sigma(\theta_0, \ldots, \theta_k)$ is the natural filtration associated to the process $(\theta_k)_{k \geq 0}$ it follows that

$$\mathbf{E}\left[\left(\frac{\sum_{k=0}^{m-1} X_{n,i}^{(k)} \delta_{k+1}^{(n+i)/2}}{T_m^{1/2}} \right)^2 \right] = \frac{\sum_{k=0}^{m-1} \mathbf{E}\left[(X_{n,i}^{(k)})^2 \right] \delta_{k+1}^{n+i}}{T_m} \lesssim \frac{\sum_{k=0}^{m-1} \delta_{k+1}^{n+i}}{T_m} \to 0.$$

We made use of the fact that the expectations $\mathbf{E}\left[(X_{n,i}^{(k)})^2 \right]$ are uniformly bounded for all $k \geq 0$ by the same arguments as above, and that the final expression converges to 0 since $n+i \geq 2$, $\delta_m \to 0$ and $T_m \to \infty$. This concludes the proof that the remainder term $\mathscr{R}_m$ is asymptotically negligible.

Fluctuation Term

we now prove that the fluctuations term converges in distribution at Monte-Carlo rate towards a Gaussian distribution,

$$T_m^{1/2} \, \mathscr{F}_m \equiv -\frac{\sum_{k=0}^{m-1} \nabla h(\theta_k) \, \delta_{k+1}^{1/2} \, \eta_{k+1}}{T_m^{1/2}} \to \mathrm{N}\left(0, \sigma^2(\varphi)\right).$$

Using the standard martingale central limit theorem (e.g. Theorem 3.2, Chapter 3 of (Hall and Heyde, 1980)), it suffices to verify that for any $\varepsilon > 0$ the following limits hold in probability,

$$\lim_{m \to \infty} \sum_{k=0}^{m-1} \frac{\mathbf{E}_k\left[Z_k^2 \mathbb{I}\left(Z_k^2 > T_m \varepsilon \right) \right]}{T_m} = 0 \quad \text{and} \quad \lim_{m \to \infty} \frac{\sum_{k=0}^{m-1} \mathbf{E}_k\left[Z_k^2 \right]}{T_m} = \sigma^2(\varphi)$$

with $Z_k \stackrel{\text{def}}{=} \nabla h(\theta_k)\,\delta_{k+1}^{1/2}\,\eta_{k+1}$. Since $\mathbf{E}_k\left[Z_k^2\right] = \nabla h(\theta_k)^2\,\delta_{k+1}$ and the function $\theta \mapsto \nabla h(\theta)^2$ satisfies the assumptions of Theorem 7, the second limit directly follows from Theorem 7. For proving the first limit, note that the Cauchy-Schwarz's inequality and the boundedness of ∇h imply that $\mathbf{E}_k\left[Z_k^2\,\mathbb{I}\big(Z_k^2 > T_m\varepsilon\big)\right] \lesssim \delta_{k+1} \times \mathbf{P}\left[\delta_{k+1}\,\|\nabla h\|_\infty^2\,\eta_{k+1}^2 > T_m\,\varepsilon\right]^{1/2}$; the Markov's inequality thus yields that

$$\sum_{k=0}^{m-1} \mathbf{E}_k\left[Z_k^2\,I\big(Z_k^2 > T_m\varepsilon\big)\right]/T_m \lesssim \frac{\sum_{k=0}^{m-1}\delta_{k+1}^2}{T_m^2\,\varepsilon}.$$

Since $T_m^{-2}\sum_{k=0}^{m-1}\delta_{k+1}^2 \to 0$, the conclusion follows.

Bias term: we conclude by proving that the bias term is such that the limit

$$\lim_{m\to\infty} \frac{\mathscr{B}_m}{\sum_{k=1}^m \delta_k^2/T_m} \;=\; \mu(\varphi)$$

holds in probability. The quantity $\mathscr{B}_m/\big(\sum_{k=1}^m \delta_k^2/T_m\big)^{-1}$ can also be expressed as

$$\frac{\sum_{k=0}^{m-1}\Psi(\theta_k)\,\delta_{k+1}^2}{\sum_{k=0}^{m-1}\delta_{k+1}^2} + \frac{\sum_{k=0}^{m-1}\Delta M_k\,\delta_{k+1}^2}{\sum_{k=0}^{m-1}\delta_{k+1}^2} \tag{31}$$

for a martingale difference term $\Delta M_k \equiv \left(\mathcal{C}_{2,2}^{(k)} + \mathcal{C}_{3,1}^{(k)} + \mathcal{C}_{4,0}^{(k)}\right) - \Psi(\theta_k)$ where $\Psi(\theta_k) \equiv \mathbf{E}\left[\mathcal{C}_{2,2}^{(k)} + \mathcal{C}_{3,1}^{(k)} + \mathcal{C}_{4,0}^{(k)}\,\middle|\,\mathcal{F}_k\right]$ and $\left(\mathcal{C}_{2,2}^{(k)} + \mathcal{C}_{3,1}^{(k)} + \mathcal{C}_{4,0}^{(k)}\right)$ equals

$$\frac{1}{8}\nabla^2 h(\theta_k)\widehat{\nabla\log\pi}(\theta_k,\mathcal{U}_{k+1})^2 + \frac{1}{4}\nabla^3 h(\theta_k)\widehat{\nabla\log\pi}(\theta_k,\mathcal{U}_{k+1})\eta_{k+1}^2 \quad + \frac{1}{24}\nabla^4 h(\theta_k)\eta_{k+1}^4.$$

Under the assumptions of Theorem 8, the function Ψ satisfies the hypothesis of Theorem 7 applied to the weight sequence $\{\delta_k^2\}_{k\geq 0}$; it follows that the first term in Equation (31) converge almost surely to $\mu(\varphi)$. It remains to prove that the second term in Equation (31) also converges almost surely to zero. By Lemma 6, it suffices to prove that the martingale

$$m \mapsto \sum_{k=0}^{m} \frac{\Delta M_k\,\delta_{k+1}^2}{\sum_{j=1}^{k+1}\delta_{j+1}^2}$$

is bounded in L^2. Under the Assumption of Theorem 8, Lemma 5 yields that the martingale difference term ΔM_k is uniformly bounded in L^2 from which the conclusion readily follows. ■

For the standard choice of step-sizes $\delta_m = (m_0+m)^{-\alpha}$ the statistical fluctuations dominate in the range $1/3 < \alpha \leq 1$, there is an exact balance between bias and fluctuations for $\alpha = 1/3$, and the bias dominates for $0 < \alpha < 1/3$. The optimal rate of convergence is obtained for $\alpha = 1/3$ and leads to an algorithm that converges at rate $m^{-1/3}$.

6. Diffusion Limit

In this section we show that, when observed on the right (inhomogeneous) time scale, the sample path of the SGLD algorithm converges to the continuous time Langevin diffusion of Equation (1), confirming the heuristic discussion in Welling and Teh (2011).

The result is based on the continuity properties of the Itô's map $\mathcal{I} : \mathcal{C}([0,T],\mathbb{R}^d) \to \mathcal{C}([0,T],\mathbb{R}^d)$, which sends a continuous path $w \in \mathcal{C}([0,T],\mathbb{R}^d)$ to the unique solution $v = \mathcal{I}(w)$ of the integral equation,

$$v_t = \theta_0 + \frac{1}{2}\int_{s=0}^t \nabla\log\pi(v_s)\,ds + w_t \qquad \text{for all} \quad t\in[0,T].$$

If the drift function $\theta \mapsto \frac{1}{2}\nabla\log\pi(\theta)$ is globally Lipschitz, then the Itô's map $\mathcal{I}$ is well defined and continuous. Further, the image $\mathcal{I}(W)$ under the Itô map of a standard Brownian motion W on $[0,T]$ can be seen to be described by Langevin diffusion (1).

The approach, inspired by ideas in Mattingly et al. (2012); Pillai et al. (2012), is to construct a sequence of coupled Markov chains $(\theta^{(r)})_{r\geq 1}$, each started at the same initial state $\theta_0 \in \mathbb{R}^d$ and evolved according to the SGLD algorithm with step-sizes $\delta^{(r)} \stackrel{\text{def}}{=} (\delta_k^{(r)})_{k=1}^{m(r)}$ such that

$$\sum_{k=1}^{m(r)} \delta_k^{(r)} = T$$

and with increasingly fine mesh sizes $\text{mesh}(\delta^{(r)}) \to 0$ with

$$\text{mesh}(\delta^{(r)}) \stackrel{\text{def}}{=} \max\left\{\delta_k^{(r)} \;:\; 1\leq k\leq m(r)\right\}.$$

Define $T_0^{(r)} = 0$ and $T_k^{(r)} = \delta_1^{(r)} + \cdots + \delta_k^{(r)}$ for each $k \geq 1$. The Markov chains are coupled to W as follows:

$$\left\{\begin{array}{ll} \eta_k^{(r)} & = (\delta_k^{(r)})^{-1/2}\Big(W(T_k^{(r)}) - W(T_{k-1}^{(r)})\Big) \\ \theta_k^{(r)} & = \theta_{k-1}^{(r)} + \frac{1}{2}\,\delta_k^{(r)}\Big\{\nabla\log\pi(\theta_{k-1}^{(r)}) + H(\theta_{k-1}^{(r)},\mathcal{U}_k^{(r)})\Big\} \;+\; (\delta_k^{(r)})^{1/2}\,\eta_k^{(r)}, \end{array}\right. \tag{32}$$

for an i.i.d. collection of auxiliary random variables $(\mathcal{U}_k^{(r)})_{r\geq 1,k\geq 1}$. Note that $(\eta_k^{(r)})_{k\geq 1}$ form an i.i.d. sequence of $\mathrm{N}(0,1)$ variables for each r. We can construct piecewise affine continuous time sample paths $(S^{(r)})_{r\geq 1}$ by linearly interpolating the Markov chains,

$$S^{(r)}\Big(xT_{k-1}^{(r)} + (1-x)T_k^{(r)}\Big) = x\,\theta_{k-1}^{(r)} + (1-x)\,\theta_k^{(r)}, \tag{33}$$

for $x \in [0,1]$. The approach then amounts to showing that each $S^{(r)}$ can be expressed as $\mathcal{I}(\widetilde{W}^{(r)}) + e^{(r)}$, where $\widetilde{W}^{(r)}$ is a sequence of stochastic processes converging to W and $e^{(r)}$ is asymptotically negligible, and making use of the continuity properties of the Itô map $\mathcal{I}$.

Theorem 9 *Let Assumption 4 holds and suppose that the drift function $\theta \mapsto (1/2)\nabla\,\log\pi(\theta)$ is globally Lipschitz on $\mathbb{R}^d$. If $\text{mesh}(\delta^{(r)}) \to 0$ as $r\to\infty$, then the sequence of continuous time processes $(S^{(r)})_{r\geq 1}$ defined in Equation (33) converges weakly on $\big(\mathcal{C}([0,T],\mathbb{R}^d),\|\cdot\|_\infty\big)$ to the Langevin diffusion (1) started at $S_0 = \theta_0$.*

Proof Since the drift term $s \mapsto (1/2)\nabla \log \pi(s)$ is globally Lipschitz on $\mathbb{R}^d$, Lemma 3.7 of (Mattingly et al., 2012) shows that the Itô's map $\mathcal{I} : \mathcal{C}([0,T],\mathbb{R}^d) \to \mathcal{C}([0,T],\mathbb{R}^d)$ is well-defined and continuous, under the topology over the space $\mathcal{C}([0,T],\mathbb{R}^d)$ induced by the supremum norm $\|w\|_\infty \equiv \sup\{|w_t| : 0 \leq t \leq T\}$. By the Continuous Mapping Theorem, because the Langevin diffusion (1) can be seen as the image under the Itô's map $\mathcal{I}$ of a standard Brownian motion on $[0,T]$ evolving in $\mathbb{R}^d$, it suffices to verify that the process $S^{(r)}$ can be expressed as $\mathcal{I}(\widetilde{W}^{(r)}) + e^{(r)}$ where $\widetilde{W}^{(k)}$ is a sequence of stochastic processes that converge weakly in $\mathcal{C}([0,T],\mathbb{R}^d)$ to a standard Brownian motion W and $e^{(r)}$ is an error term that is asymptotically negligible in the sense that $\|e^{(r)}\|_\infty$ converges to zero in probability.

For convenience, we define $\widetilde{W}^{(r)}$ as the continuous piecewise affine processes that satisfies $\widetilde{W}^{(r)}(T_k^{(r)}) = W(T_k^{(r)})$ for all $0 \leq k \leq m(r)$ and that is affine in between. It follows that for any time $T_{k-1}^{(r)} \leq t \leq T_k^{(r)}$ we have

$$
\begin{aligned}
S^{(r)}(t) &= S^{(r)}(T_{k-1}^{(r)}) + \left(\int_{T_{k-1}^{(r)}}^{t} \frac{1}{2} \nabla \log \pi\big(S^{(r)}(T_{k-1}^{(r)})\big) du + \widetilde{W}^{(r)}(t) - \widetilde{W}(T_{k-1}^{(r)}) \right) \\
&\quad + \frac{1}{2} \int_{T_{k-1}^{(r)}}^{t} H\big(S^{(r)}(T_{k-1}^{(r)}), \mathcal{U}_k^{(r)}\big) du \\
&= \underbrace{\theta_0 + \left(\int_0^t \frac{1}{2} \nabla \log \pi\big(S^{(r)}(u)\big) du + \widetilde{W}^{(r)}(t) \right)}_{\mathcal{I}(\widetilde{W})(t)} \\
&\quad + \underbrace{\int_0^t \frac{1}{2} \left(\nabla \log \pi\big(\widehat{S}^{(r)}(u)\big) - \nabla \log \pi\big(S^{(r)}(u)\big) \right) du}_{e_1^{(r)}(t)} \\
&\quad + \frac{1}{2} \underbrace{\int_0^t H\big(\widehat{S}^{(r)}(u), \mathcal{U}_k^{(r)}\big) du}_{e_2^{(r)}(t)},
\end{aligned}
$$

where $\widehat{S}^{(r)}$ is a piecewise constant (non-continuous) process, $\widehat{S}^{(r)}(t) = S^{(r)}(T_{k-1}^{(r)}) = \theta_{k-1}^{(r)}$ for $t \in [T_{k-1}^{(r)}, T_k^{(r)})$. The process $S^{(r)}$ can thus be expressed as the sum $\mathcal{I}(W^{(r)}) + e_1^{(r)} + e_2^{(r)}$. Since the mesh-size of the partition $\delta^{(r)}$ converges to zero as $r \to \infty$, standard properties of Brownian motions yield that $\widetilde{W}^{(r)}$ converges weakly in $\big(\mathcal{C}([0,t],\mathbb{R}^d), \|\cdot\|_{\infty,[0,T]}\big)$ to W, a standard Brownian motion in $\mathbb{R}^d$. To conclude the proof, we need to check that the quantities $\|e_1^{(r)}\|_\infty$ and $\|e_2^{(r)}\|_\infty$ converge to zero in probability. To prove $\mathbf{E}\left[\|e_2^{(r)}\|_\infty^2\right] \to 0$ in probability, we have,

$$
\begin{aligned}
\mathbf{E}\left[\|e_2^{(r)}\|_\infty^2\right] &\leq 4\, \mathbf{E}\left[\|e_2^{(r)}(T)\|^2\right] = 4 \sum_{k=1}^{m(r)} \big(\delta_k^{(r)}\big)^2\, \mathbf{E}\left[H\Big(\theta_{k-1}^{(r)}, \mathcal{U}_k^{(r)}\Big)^2\right] \\
&\lesssim \sum_{k=1}^{m(r)} \big(\delta_k^{(r)}\big)^2\, \mathbf{E}\left[V(\theta_{k-1}^{(r)})\right] \leq \mathrm{mesh}(\delta^{(r)}) \sum_{k=1}^{m(r)} \delta_k^{(r)}\, \mathbf{E}\left[V(\theta_{k-1}^{(r)})\right] \\
&\leq \mathrm{mesh}(\delta^{(r)}) \times T \times \sup\left\{ \mathbf{E}\left[V(\theta_{k-1}^{(r)})\right] : r \geq 1, 1 \leq k \leq m(r) \right\} \lesssim \mathrm{mesh}(\delta^{(r)}).
\end{aligned}
$$

We have used Doob's martingale inequality, Assumption 4 and Lemma 5. Since $\mathrm{mesh}(\delta^{(r)})$ converges to zero, the conclusion follows. To prove $\mathbf{E}\left[\|e_1^{(r)}\|_\infty\right] \to 0$ in probability, we use Equation (32) and note that since the drift function $\theta \mapsto \frac{1}{2}\nabla \log \pi(\theta)$ is globally Lipschitz, for each $T_{k-1}^{(r)} \leq u \leq T_k^{(r)}$ we have,

$$
\begin{aligned}
\left\| \nabla \log\big(\widehat{S}^{(r)}(u)\big) - \nabla \log\big(S^{(r)}(u)\big) \right\| &\lesssim \left\| \theta_k^{(r)} - \theta_{k-1}^{(r)} \right\| \\
&\lesssim \|\nabla \log \pi(\theta_{k-1}^{(r)})\|\, \delta_k^{(r)} + \|H(\theta_{k-1}^{(r)}, \mathcal{U}_k^{(r)})\|\, \delta_k^{(r)} + \sqrt{\delta_k^{(r)}}\, \|\eta_k^{(r)}\|.
\end{aligned}
$$

It follows that

$$
\mathbf{E}\left[\|e_1^{(r)}\|_\infty\right] \lesssim \sum_{k=1}^{m(r)} \delta_k^{(r)} \left(\|\nabla \log \pi(\theta_k^{(r)})\|\, \delta_k^{(r)} + \|H(\theta_k^{(r)}, \mathcal{U}_k)\|\, \delta_k^{(r)} + \sqrt{\delta_k^{(r)}}\, \|\eta_k^{(r)}\| \right).
$$

Since $\mathrm{mesh}(\delta^{(r)})$ converges to zero and by Assumption 4 and Lemma 5 the suprema

$$
\begin{cases}
\sup & \left\{ \mathbf{E}\left[\|\nabla \log \pi(\theta_k^{(r)})\|\right] : r \geq 1, 1 \leq k \leq m(r) \right\}, \\
\sup & \left\{ \mathbf{E}\left[\|H(\theta_k^{(r)}, \mathcal{U}_k)\|\right] : r \geq 1, 1 \leq k \leq m(r) \right\}
\end{cases}
$$

are finite, it readily follows that $\|e_1^{(r)}\|_\infty$ converges to zero in expectation. ■

7. Numerical Illustrations

In this section we illustrate the use of the SGLD method to a simple Gaussian toy model and to a Bayesian logistic regression problem. We verify that both models satisfy Assumption 4, the main assumption needed for our asymptotic results to hold. Simulations are then performed to empirically confirm our theory; for step-sizes sequences of the type $\delta_m = (m_0 + m)^{-\alpha}$, both the rate of decay of the MSE and the impact of the sub-sampling scheme are investigated. The main purpose of this article is to establish the missing theoretical foundation of stochastic gradient methods for the approximation of expectations. For more exhaustive simulation studies we refer to Welling and Teh (2011); S. Ahn and Welling (2012); Patterson and Teh (2013a); Chen et al. (2014). By considering a logistic regression model, we demonstrate that the SGLD can be advantageous over the Metropolis-Adjusted-Langevin (MALA) algorithm if the available computational budget only allows a few iterations through the whole data set, see Section 7.2.2.

7.1 Linear Gaussian model

Consider N independent and identically distributed observations $(x_i)_{i=1}^N$ from the two parameters location model given by

$$x_i \mid \theta \sim \mathrm{N}(\theta, \sigma_x^2).$$

We use a Gaussian prior $\theta \sim \mathrm{N}(0, \sigma_\theta^2)$ and assume that the variance hyper-parameters σ_θ^2 and σ_x^2 are both known. The posterior density $\pi(\theta)$ is normally distributed with mean μ_p and variance σ_p^2 given by

$$\mu_p = \bar{x}\left(1 + \frac{\sigma_x^2}{N\sigma_\theta^2}\right)^{-1} \qquad \text{and} \qquad \sigma_p^2 = \frac{\sigma_x^2}{N}\left(1 + \frac{\sigma_x^2}{N\sigma_\theta^2}\right)^{-1}$$

where $\bar{x} = (x_1 + \ldots + x_N)/N$ is the sample average of the observations. In this case, we have

$$\nabla \log \pi(\theta) = -\frac{\theta - \mu_p}{\sigma_p^2} \quad \text{and} \quad H(\theta, \mathcal{U}) = \Big\{(N/n) \sum_{j \in \mathcal{I}_n(\mathcal{U})} x_j - \sum_{1 \le i \le N} x_i\Big\}/\sigma_x^2$$

for a random subset $\mathcal{I}_n(\mathcal{U}) \subset [N]$ of cardinal n.

7.1.1 Verification of Assumption 4

We verify in this section that Assumption (4) is satisfied for the following choice of Lyapunov function,

$$V(\theta) = 1 + \frac{(\theta - \mu_p)^2}{2\,\sigma_p^2}.$$

Since the error term $H(\theta,\mathcal{U})$ is globally bounded, the drift $(1/2)\nabla \log \pi$ and the Lyapunov function V are linear, Assumptions (4).1 and (4).2 are satisfied. Finally, to verify Assumption (4).3, it suffices to note that since $\nabla \log \pi(\theta) = -(\theta - \mu_p)/\sigma_p^2$ we have

$$\left\langle \nabla V(\theta), \frac{1}{2}\nabla \log \pi(\theta) \right\rangle = -\frac{(\theta - \mu_p)^2}{2\,\sigma_p^4} = \frac{1 - V(\theta)}{\sigma_p^2}.$$

In other words, Assumption (4).3 holds with $\alpha = \beta = 1/\sigma_p^2$.

7.1.2 Simulations

We chose $\sigma_\theta = 1$, $\sigma_x = 5$ and created a data set consisting of $N = 100$ data points simulated from the model. We used $n = 10$ as the size of subsets used to estimate the gradients. We evaluated the convergence behaviour of SGLD using the test function $\mathcal{A}\varphi$ where $\varphi = \sin(x - \mu_p - 0.5\sigma_p)$.

We are interested in confirming the asymptotic convergence regimes of Theorem 8 by running SGLD with a range of step sizes, and plotting the mean squared error (MSE) achieved by the estimate $\pi_m(\mathcal{A}\varphi)$ against the number of steps m of the algorithm to determine the rates of convergence. We used step sizes $\delta_m = (m + m_0(\alpha))^{-\alpha}$, for $\alpha \in \{0.1, 0.2, 0.3, 0.33, 0.4, 0.5\}$ where $m_0(\alpha)$ is chosen such that δ_1 is less than the posterior

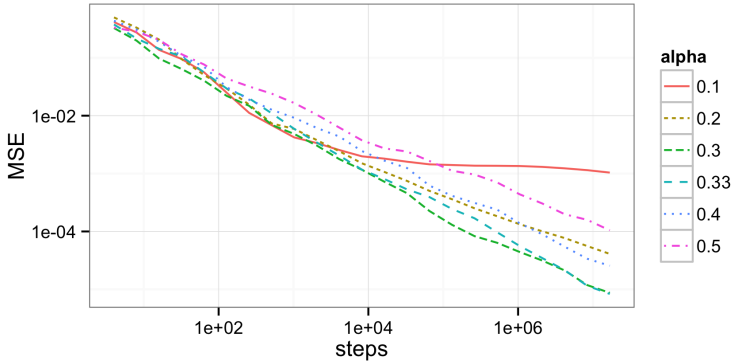


Figure 1: Decay of the MSE for step sizes $\delta_m \asymp m^{-\alpha}$, $\alpha \in \{0.1, 0.2, 0.3, 0.33, 0.4, 0.5\}$. The MSE decays algebraically for all step sizes, with fastest decay at approximately $\alpha = 0.33$.

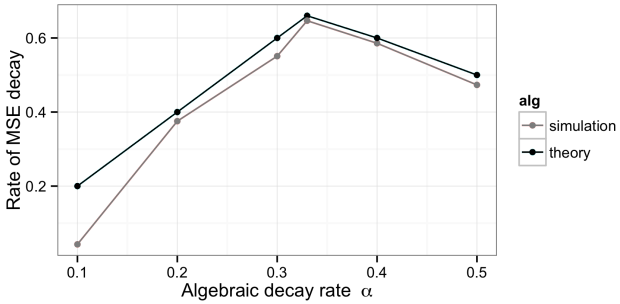


Figure 2: Rates of decay of the MSE obtained from estimating the asymptotic slopes of the plots in Figure 1, compared to theoretical findings of Theorem 8. The fastest convergence rate is achieved at $\alpha = 1/3$.

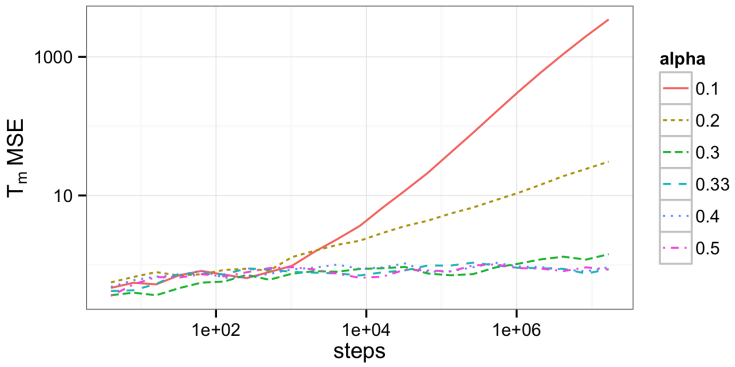


Figure 3: Plots of the MSE multiplied by T_m against the number of steps m. The plots are flat for $\alpha \geq 0.33$, demonstrating that the MSE scales as T_m^{-1} in this regime, while the plots diverge for $\alpha < 0.33$, demonstrating that it decays at a slower rate here.

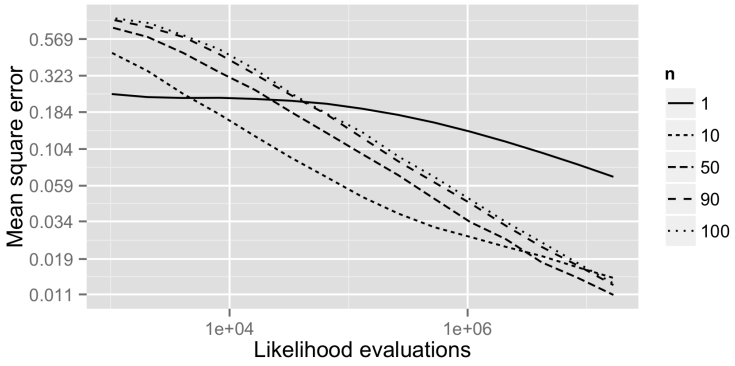


Figure 4: Behaviour of the mean squared error for different subsample sizes n.

standard deviation. According to the Theorem, the MSE should scale as T_m^{-1} for $\alpha > 1/3$, and $\sum_{k=1}^m \delta_k^2/T_m$ for $\alpha \leq 1/3$.

The observed MSE is plotted against m on a log-log plot in Figure 1. As predicted by the theory, the optimal rate of decay is around $\alpha_\star = 1/3$. To be more precise, we estimate the rates of decay by estimating the slopes on the log-log plots. This is plotted in Figure 2, which also shows a good match to the theoretical rates given in Theorem 8, where the best rate of decay is 2/3 achieved at $\alpha = 1/3$. Finally, to demonstrate that there are indeed two distinct regimes of convergence, in Figure 3 we have plotted the MSE multiplied by T_m. For $\alpha > 1/3$, the plots remain flat, showing that the MSE does indeed decay as T_m^{-1}. For $\alpha < 1/3$, the plots diverge, showing that the MSE decays at a slower rate than T_m^{-1}.

For $\alpha = 0.33$, Figure 4 depicts how the MSE decreases as a function of the number of likelihood evaluations for subsample sizes $n = 1, 5, 10, 50, 100$.

7.2 Logistic Regression

We verify in this section that Assumption (4) is satisfied for the following logistic regression model. Consider N independent and identically observations $(y_i)_{i=1}^N$ distributed as

$$\mathbb{P}(y_i = 1 \mid x_i, \theta) \;=\; 1 - \mathbb{P}(y_i = -1 \mid x_i, \theta) \;=\; \mathrm{logit}\big(\langle \theta, x_i \rangle\big) \tag{34}$$

for covariate $x_i \in \mathbb{R}^d$, unknown parameter $\theta \in \mathbb{R}^d$ and function $\mathrm{logit}(z) = e^z/(1+e^z)$. We assume a centred Gaussian prior on $\theta \in \mathbb{R}^d$ with positive definite symmetric covariance matrix $C \in \mathbb{R}^{d \times d}$. It follows that

$$\nabla \log \pi(\theta) = -C^{-1}\theta + \sum_{i=1}^N \mathrm{logit}\big(-y_i \langle \theta, x_i \rangle\big)\, y_i\, x_i$$

$$H(\theta, \mathcal{U}) = (N/n) \sum_{j \in \mathcal{I}_n(U)} \mathrm{logit}\big(-y_j \langle \theta, x_j \rangle\big)\, y_j\, x_j - \sum_{1 \leq i \leq N} \mathrm{logit}\big(-y_i \langle \theta, x_i \rangle\big)\, y_i\, x_i$$

for a random subset $\mathcal{I}_n(\mathcal{U}) \subset [N]$ of cardinal n.

7.2.1 Verification of Assumption 4

We verify in this section that Assumption (4) is satisfied for the Lyapunov function $V(\theta) = 1 + \|\theta\|^2$. Since $H(\theta, \mathcal{U})$ is globally bounded and $\|\nabla V(\theta)\|^2 = \|\theta\|^2$ and

$$\|\nabla \log \pi(\theta)\|^2 \lesssim 1 + \|C^{-1}\theta\|^2 \lesssim 1 + \|\theta\|^2 = V(\theta),$$

it is straightforward to see that Assumption (4).1 and (4).2 are satisfied. Finally,

$$\begin{aligned}\left\langle \nabla V(\theta), \frac{1}{2} \nabla \log \pi(\theta) \right\rangle &= -\frac{1}{2} \langle \theta, C^{-1}\theta \rangle + \frac{1}{2} \sum_{i=1}^N \mathrm{logit}\big(-y_i \langle \theta, x_i \rangle\big)\, y_i \, \langle \theta, x_i \rangle \\ &\leq -\frac{\lambda_{\min}}{2} \|\theta\|^2 + \frac{\sum_{i=1}^N \|x_i\|}{2} \|\theta\| \leq -\frac{\lambda_{\min}}{4} V(\theta) + \beta\end{aligned}$$

with $\lambda_{\min} > 0$ the smallest eigenvalue of C^{-1} and $\beta \in (0, \infty)$ the global maximum over $\theta \in \mathbb{R}^d$ of the function $\theta \mapsto -\frac{\lambda_{\min}}{4}\|\theta\|^2 + \frac{\sum_{i=1}^N \|x_i\|}{2}\|\theta\|$.

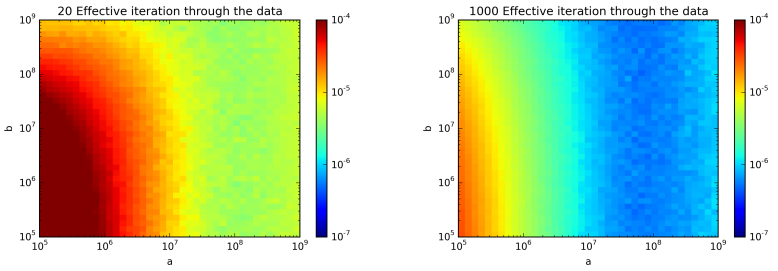


Figure 5: Expected MSE of the SGLD-based estimate variance estimate of the first component for $n = 30$ and step sizes $\delta_m = (a \cdot m + b)^{-0.38}$ after 20 and 1000 iterations through the data set

7.2.2 Comparison of the SGLD and the MALA for Logistic Regression

We consider a simulated dataset where $d = 3$ and $N = 1000$. We set the input covariates $x_i = (x_{i,1}, x_{i,2}, 1)$ with $x_{i,1}, x_{i,2} \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0,1)$ for $i = 1 \dots N$, and use a Gaussian prior $\theta \sim \mathrm{N}(0, I)$. We draw a $\theta_0 \sim \mathrm{N}(0, I)$ and based on it we generate y_i according to the model probabilities (34). In the following we compare MALA in SGLD by comparing their estimate for the variance of the first component.

The findings of this article show that SGLD-based expectation estimates converge at a slower rate of at most $n^{-\frac{1}{3}}$ compared to the standard rate of $n^{-\frac{1}{2}}$ for standard MCMC algorithms such as the MALA algorithm. In the following we demonstrate that in the non-asymptotic regime (allowing only a few passes through the data set) the SGLD can be advantageous. We start both algorithms at the MAP estimator and we ensure that this study is not biased due to different speeds in finding the mode of the posterior. For a fair comparison we tune the MALA to an acceptance rate of approximately 0.564 following the findings of Roberts and Rosenthal (1998). For the SGLD-based variance estimate of the first component for $n = 30$ we choose $\delta_m = (a \cdot m + b)^{-0.38}$ as step sizes and optimise over the choices of a and b. This is achieved by estimating the MSE for choices of a and b on a log-scale grid based on 512 independent runs. The estimates based on 20 and 1000 effective iterations through the data set the averages are visualised in the heat maps in Figure 5. That means we limit the algorithm to 200 and 1000000 likelihood evaluations, respectively. The figures indicate that the range of the good parameter choices seems to be the same in both cases. Using the heat map for the estimated MSE after 20 iterations through the data set, we pick $a = 5.89 \cdot 10^7$ and $b = 7.90 \cdot 10^8$ and compare the time behaviour of the SGLD and the MALA algorithm in Figure 6. The figure is a simulation evidence that the SGLD algorithm can be advantageous in the initial phase for the first few iterations through the data set. This recommends further investigation as the initial phase can be quite different from the asymptotic phase.

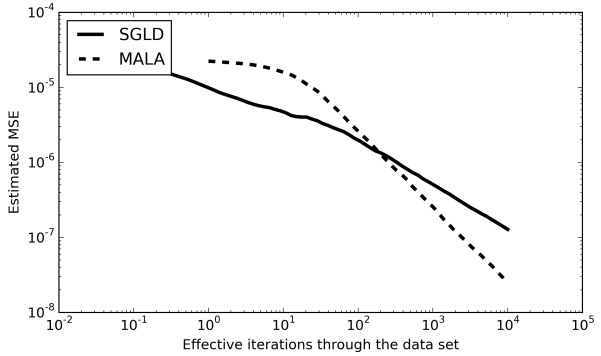


Figure 6: Behaviour of the MSE of estimating the posterior variance of the first component for 3-dimensional logistic regression of MALA and SGLD with tuned parameters

8. Conclusion

So far, the research on the SGLD algorithm has mainly been focused on extending the methodology. In particular, a parallel version has been introduced in Ahn et al. (2014) and it has been adapted to natural gradients in Patterson and Teh (2013b). This research has been accompanied by promising simulations. In contrast, we have focused in this article on providing rigorous mathematical foundations for the SGLD algorithm by showing that the step-size weighted estimator $\pi_m(f)$ is consistent, satisfies a central limit theorem and its asymptotic bias-variance decomposition can be characterised by an explicit functional $\mathbb{B}_m$ of the step-sizes sequence $(\delta_m)_{m \geq 0}$. The consistency of the algorithm is mainly due to the decreasing step-sizes procedure that asymptotically removes the bias from the discretization and ultimately mitigates the use of an unbiased estimate of the gradient instead of the exact value. Additionally, we have proved a diffusion limit result that establishes that, when observed on the right (inhomogeneous) time scale, the sample paths of the SGLD can be approximated by a Langevin diffusion.

The CLT and bias-variance decomposition can be leveraged to show that it is optimal to choose a step-sizes sequences $(\delta_m)_{m \geq 0}$ that scales as $\delta_m \asymp m^{-1/3}$; the resulting algorithm converges at rate $m^{-1/3}$. Note that this recommendation is different from the previously suggested Welling and Teh (2011) choice of $\delta_m \asymp m^{-1/2}$.

Our theory suggests that an optimally tuned SGLD method converges at rate $\mathcal{O}(m^{-1/3})$, and is thus asymptotically less efficient than a standard MCMC procedure. We believe that this result does not necessarily preclude SGLD to be more efficient in the initial transient phase, a result hinted at in Figure 4; the detailed study of this (non-asymptotic) phenomenon is an interesting venue of research. The asymptotic convergence rate of SGLD depends crucially on the decreasing step sizes, which is required to reduce the effect of the discretization bias due to the lack of a Metropolis-Hastings correction. Another avenue of exploration is to determine more precisely the bias resulting from the discretization of the

Langevin diffusion, and to study the effect of the choice of step sizes in terms of the trade-off between bias, variance, and computation.

Acknowledgement

SJV and YWT acknowledge EPSRC for research funding through grant EP/K009850/1, EP/N000188/1 and EP/K009362/1. AHT is grateful for financial support in carrying out this research from a Singaporean MoE grant.

Appendix A. Proof of Lemma 6

Recall Kronecker's Lemma (Shiryaev, 1996, Lemma IV.3.2) that states that for a non-decreasing and positive sequence $b_m \to \infty$ and another real valued sequence $(a_m)_{m\geq 0}$ such that the series $\sum_{m\geq 0} a_m/b_m$ converges the following limit holds,

$$\lim_{m\to\infty} \frac{\sum_{k=0}^{m} a_k}{b_m} \;=\; 0.$$

For proving Equation (15) it thus suffices to show that the sums $\sum_{k\geq 0} |\Delta M_k|/T_k$ and $\sum_{k\geq 0} |X_k|/T_k$ are almost surely finite. This follows from Condition (16) (L^2 martingale convergence theorem) and Condition (17).

Appendix B. Proof of Lemma 5

For clarity, the proof is only presented in the scalar case $d = 1$; the multidimensional setting is entirely similar. Before embarking on the proof, let us first mention some consequences of Assumptions 4 that will be repeatedly used in the sequel. Since the second derivative V'' is globally bounded and $(V')^2$ is upper bounded by a multiple of V, we have that

$$\left|(V^p)''(\theta)\right| \lesssim V^{p-1}(\theta) \tag{35}$$

and that the function $V^{1/2}$ is globally Lipschitz. By expressing the quantity $V^p(\theta+\varepsilon)$ as $\big(V^{1/2}(\theta) + [V^{1/2}(\theta+\varepsilon) - V^{1/2}(\theta)]\big)^{2p}$, it then follows that

$$V^p(\theta+\varepsilon) \lesssim V^p(\theta) + |\varepsilon|^{2p}. \tag{36}$$

Similarly, Definition (7), the bound $\|\nabla \log \mathrm{p}(\theta)\|^2 \lesssim V(\theta)$ and Equation (10) yield that for any exponent $0 \leq p \leq p_H$ the following holds,

$$\mathbf{E}_m[\,|\theta_{m+1} - \theta_m|^{2p}\,] \lesssim \delta_{m+1}^{2p}\, V^p(\theta) + \delta_{m+1}^p. \tag{37}$$

For clarity, the proof of Lemma (5) is separated into several steps. First, we establish that the process $m \mapsto V^p(\theta_m)$ satisfies a Lyapunov type condition; see Equation (38) below. We then describe how Equation (13) follows from this Lyapunov condition. The fact that $\pi(V^p)$ is finite can be seen as a consequence of Theorem 2.2 of (Roberts and Tweedie, 1996).

- *Discrete Lyapunov condition.*
  Let us prove that there exists an index $m_0 \geq 0$ and constants $\alpha_p, \beta_p > 0$ such that for any $m \geq m_0$ we have

$$\mathbf{E}_m\left[V^p(\theta_{m+1}) - V^p(\theta_m)\right]/\delta_{m+1} \;\leq\; -\alpha_p\, V^p(\theta_m) + \beta_p. \tag{38}$$

Since for any ε there exists C_ε such that $V^{p-1}(\theta) \leq C_\varepsilon + \varepsilon V^p(\theta)$, for proving (38) it actually suffices to verify that we have

$$\mathbf{E}_m\left[V^p(\theta_{m+1}) - V^p(\theta_m)\right]/\delta_{m+1} \;\leq\; -\widetilde{\alpha}_p\, V^p(\theta_m) + \widetilde{\beta}_p\, V^{p-1}(\theta_m) \tag{39}$$

for some constants $\widetilde{\alpha_p}, \widetilde{\beta_p} > 0$ and index $m \geq 1$ large enough. A second order Taylor expansion yields that the left hand side of (39) is less than

$$\mathbf{E}_m\left[(V^p)'(\theta_m)\,(\theta_{m+1} - \theta_m)\right]/\delta_{m+1} + \frac{1}{2}\,\mathbf{E}_m\left[(V^p)''(\xi)\,(\theta_{m+1} - \theta_m)^2\right]/\delta_{m+1} \tag{40}$$

for a random quantity ξ lying between θ_m and θ_{m+1}. Since $\mathbf{E}_m[\theta_{m+1} - \theta_m] = \frac{1}{2}\nabla \log \mathrm{p}(\theta_m)$, the drift condition (12) yields that the first term of (40) is less than

$$p\, V^{p-1}(\theta_m)\,(-\alpha\, V(\theta_m) + \beta) \tag{41}$$

for $\alpha, \beta > 0$ given by Equation (12). Consequently, for proving Equation (38), it remains to bound the second term of (40). Equation (35) shows that $|(V^p)''(\xi)|$ is upper bounded by a multiple of $|V^{p-1}(\xi)|$; the bound (36) then yields that $|V^{p-1}(\xi)|$ is less than a constant multiple of $|V^{p-1}(\theta_m)| + |\theta_{m+1} - \theta_m|^{2(p-1)}$. It follows from the bound (37) on the difference $(\theta_{m+1} - \theta_m)$ and the assumption $\mathbf{E}[\,\|H(\theta, \mathcal{U})\|^{2p_H}\,] \lesssim V^{p_H}(\theta)$ that for any $\varepsilon > 0$ one can find an index $m_0 \geq 1$ large enough such that for any index $m \geq m_0$ the second term of (39) is less than a constant multiple of

$$\varepsilon\, V^p(\theta_m) + \beta_{p,\varepsilon}\, V^{p-1}(\theta) \tag{42}$$

for a constant $\beta_{p,\varepsilon} > 0$. Equations (41) and (42) directly yield to Equation (39), which in turn implies to Equation (38).

- *Proof that* $\sup_{m\geq 1}\, \mathbf{E}[V^p(\theta_m)] \,<\, \infty$ *for any* $p \leq p_H$.
  Equations (36) and (37) show that if $\mathbf{E}[V^p(\theta_m)]$ is finite then so is $\mathbf{E}[V^p(\theta_{m+1})]$. Under the conditions of Lemma 5, this shows that $\mathbf{E}[V^p(\theta_m)]$ is finite for any $m \geq 0$. An inductive argument based on the discrete Lyapunov Equation (38) then yields that for any index $m \geq m_0$ the expectation $\mathbf{E}[V^p(\theta_m)]$ is less than

$$\max\Big(\beta_p/\alpha_p, \max\big\{\,\mathbf{E}[V^p(\theta_m)] :\;\; 0 \leq m \leq m_0\big\}\Big). \tag{43}$$

  It follows that $\sup_{m\geq 1}\, \mathbf{E}[V^p(\theta_m)]$ is finite.

- *Proof that* $\sup_{m\geq 1}\, \pi_m(V^p) \,<\, \infty$ *for any* $p \leq p_H/2$.
  One needs to prove that the sequence $(1/T_m)\sum_{k=m_0}^{m} \delta_{k+1} V^p(\theta_k)$ is almost surely bounded. The discrete Lyapunov Equation (38) yields that $\delta_{k+1} V^p(\theta_k)$ is less than $\delta_{k+1}\,\beta_p/\alpha_p - \mathbf{E}_k[V^p(\theta_{k+1}) - V^p(\theta_k)]/\alpha_p$; this yields that $(1/T_m)\sum_{k=m_0}^{m} \delta_{k+1} V^p(\theta_k)$ is less than a constant multiple of

$$1 + \frac{V^p(\theta_{m_0})}{T_m} + \frac{1}{T_m}\sum_{k=m_0}^{m} \Big\{V^p(\theta_{k+1}) - \mathbf{E}_k[V^p(\theta_{k+1})]\Big\}.$$

To conclude the proof, we prove that the last term in the above displayed Equation almost surely converges to zero; by Lemma 6, it suffices to prove that the quantity

$$\sum_{k\geq m_0} \mathbf{E}\left[\left|\frac{V^p(\theta_{k+1}) - \mathbf{E}_k[V^p(\theta_{k+1})]}{T_k}\right|^2\right] \tag{44}$$

is almost surely finite. We have $\mathbf{E}\left[|V^p(\theta_{k+1}) - \mathbf{E}_k[V^p(\theta_{k+1})]|^2\right] \leq 2 \times \mathbf{E}[|\,V^p(\theta_{k+1}) - V^p(\theta_k)\,|^2]$ and the mean value theorem yields that

$$|V^p(\theta_{k+1}) - V^p(\theta_k)| \lesssim V^{p-1}(\xi)\, V'(\xi)\, (\theta_{k+1} - \theta_k)$$

for some ξ lying between θ_k and θ_{k+1}. The bound $|V'(\theta)| \lesssim V^{1/2}(\theta)$ and Equation (36) then yield that $|V^p(\theta_{k+1}) - V^p(\theta_k)| \lesssim V^{p-1/2}(\theta_k)\left|\theta_{k+1} - \theta_k\right| + \left|\theta_{k+1} - \theta_k\right|^{2p}$. From the bound (37) and the assumption that $\mathbf{E}[H(\theta, \mathcal{U})^{2p_H}] \lesssim V^{p_H}(\theta)$ it follows that the quantity in Equation (44) is less than a constant multiple of

$$\sum_{k\geq m_0} \frac{\mathbf{E}\left[\,V^{2p}(\theta_k)\,\right] \times \delta_k}{T^2(k)}.$$

Since $\mathbf{E}\left[V^{2p}(\theta_k)\right]$ is uniformly bounded for any $p \leq p_H/2$ and $\sum_{m\geq m_0} \delta_m / T^2(m) < \infty$ (because the sum $\sum_m T^{-1}(m+1) - T^{-1}(m)$ is finite), the conclusion follows.

- *Proof of* $\pi(V^p) < \infty$ *for any* $p \geq 0$.
  Since $V(\theta) \lesssim 1 + \|\theta\|^2$, the drift condition (12) yields that Theorem 2.1 of (Roberts and Tweedie, 1996) holds. Moreover, the bound $V^{p-1}(\theta) \leq C_\varepsilon + \varepsilon\, V^p(\theta)$ implies that there are constants $\alpha_{p,*}.\beta_{p,*} > 0$ such that

$$\mathcal{A}V^p(\theta) \leq -\alpha_{p,*}\, V^p(\theta) + \beta_{p,*} \tag{45}$$

  where $\mathcal{A}$ is the generator of the Langevin diffusion (1). Theorem 2.2 of (Roberts and Tweedie, 1996) gives the conclusion.

Proof that $\sup_{m\geq 1}\ \pi_m^\omega(V^p) < \infty$ *for any* $p \leq p_H/2$.
One needs to prove that the sequence $[1/\Omega_m] \times \sum_{k=m_0}^m \omega_{k+1} V^p(\theta_k)$ is almost surely bounded. The bound $\delta_{k+1} V^p(\theta_k) \lesssim \delta_{k+1}\, \beta_p/\alpha_p - \mathbf{E}_k[V^p(\theta_{k+1}) - V^p(\theta_k)]/\alpha_p$ yields that $\pi_m^\omega(V^p)$ is less than a constant multiple of

$$1 + \frac{(\omega_{m_0}/\delta_{m_0})\, V^p(\theta_{m_0})}{T_m} \quad +\Omega^{-1}(m)\ \sum_{k=m_0+1}^m (\omega_k/\delta_k)\left\{V^p(\theta_{k+1}) - \mathbf{E}_k[V^p(\theta_{k+1})]\right\}$$
$$+\Omega^{-1}(m)\ \sum_{k=m_0}^{m-1} \Delta(\omega_k/\delta_k)\, V^p(\theta_k).$$

To conclude the proof, we establish that the following limits hold almost surely,

$$\lim_{m\to\infty}\ \Omega^{-1}(m)\ \sum_{k=m_0+1}^m (\omega_k/\delta_k)\left\{V^p(\theta_{k+1}) - \mathbf{E}_k[V^p(\theta_{k+1})]\right\} = 0 \tag{46}$$
$$\lim_{m\to\infty}\ \Omega^{-1}(m)\ \sum_{k=m_0}^{m-1} \Delta(\omega_k/\delta_k)\, V^p(\theta_k) = 0. \tag{47}$$

To prove Equation (46) it suffices to use the assumption that $\sum_{m\geq 0} \omega_m^2/[\delta_m \Omega_m^2] < \infty$ and then follow the same approach used to establish that the quantity (44) is finite. Lemma 6 shows that to prove Equation (47) it suffices to verify that

$$\mathbf{E}\Big[\sum_{m\geq 0} \left|\Delta(\omega_m/\delta_m)\right| V^p(\theta_m)/\Omega_m\Big] < \infty.$$

This directly follows from the assumption that $\sum_{m\geq 0} \left|\Delta(\omega_m/\delta_m)\right|/\Omega_m < \infty$ and the fact that $\sup_{m\geq 0} \mathbf{E}[V^p(\theta_m)]$ is finite.

References

A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. J. Smola. Scalable inference in latent variable models. In *Proceedings of the ACM international conference on Web search and data mining*, pages 123–132. ACM, 2012.

S. Ahn, A. Korattikara, and M. Welling. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *Proceedings of the International Conference on Machine Learning*, 2012.

S. Ahn, B. Shahbaba, and M. Welling. Distributed stochastic gradient MCMC. In *Proceedings of the International Conference on Machine Learning*, 2014.

Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Society, 2007.

Rémi Bardenet, Arnaud Doucet, and Chris C. Holmes. Towards scaling up MCMC: an adaptive subsampling approach. Proceedings of the International Conference on Machine Learning (ICML), 2014.

A. Beskos, G. O. Roberts, A. M. Stuart, and J. Voss. MCMC methods for diffusion bridges. *Stochastics and Dynamics*, 8(03):319–350, 2008.

Patrick Billingsley. *Probability and Measure*. Wiley-Interscience, 3 edition, 1995.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Tianqi Chen, Emily B Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. *arXiv preprint arXiv:1402.4102*, 2014.

S.L. Cotter, G.O. Roberts, A.M. Stuart, and D. White. MCMC methods for functions: modifying old algorithms to make them faster. *Statistical Science*, 28(3):424–446, 2013.

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987.

Stewart N. Ethier and Thomas G. Kurtz. *Markov processes*. Wiley Series in Probability and Mathematical Statistics: Probability and Mathematical Statistics. John Wiley & Sons Inc., New York, 1986. Characterization and convergence.

M. Girolami and B. Calderhead. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, 73 (2):123–214, 2011. With discussion and a reply by the authors.

J Gonzalez. Emerging systems for large-scale machine learning. ICML Tutorial, 2014.

M. Hairer, A. M. Stuart, and S. J. Vollmer. Spectral gaps for a metropolis-hastings algorithm in infinite dimensions. *The Annals of Applied Probability*, (to appear), 2014.

Peter Hall and Christopher C Heyde. *Martingale limit theory and its application*. Academic press New York, 1980.

Z. Harchaoui and M. Jaggi. Frank-Wolfe and greedy optimization for learning with big data. ICML Tutorial, 2014.

G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, P. Nguyen, and T. N. Sainath. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

J. N. Hirschhorn and M. J. Daly. Genome-wide association studies for common diseases and complex traits. *Nature Reviews Genetics*, 6(2):95–108, 2005.

M. D Hoffman, D. M. Blei, and F. R. Bach. Online learning for latent dirichlet allocation. In *NIPS*, volume 2, page 5, 2010.

M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

S. F. Jarner and G. O. Roberts. Convergence of heavy-tailed monte carlo markov chain algorithms. *Scandinavian Journal of Statistics*, 34(4):781–815, 2007.

K. Kamatani. Rate optimality of random walk metropolis algorithm in high-dimension with heavy-tailed target distribution. *arXiv preprint arXiv:1406.5392*, 2014.

A. Korattikara, Y. Chen, and M. Welling. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *Proceedings of the International Conference on Machine Learning*, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, volume 1, page 4, 2012.

D. Lamberton and G. Pages. Recursive computation of the invariant distribution of a diffusion. *Bernoulli*, 8(3):367–405, 2002.

D. Lamberton and G. Pages. Recursive computation of the invariant distribution of a diffusion: the case of a weakly mean reverting drift. *Stochastics and Dynamics*, 3(04): 435–451, 2003.

V. Lemaire. An adaptive scheme for the approximation of dissipative systems. *Stochastic Processes and their Applications*, 117(10):1491–1518, 2007.

S. Livingstone and M. Girolami. Information-geometric markov chain monte carlo methods using diffusions. *arXiv preprint arXiv:1403.7957*, 2014.

G. Maruyama. Continuous markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo*, 4(1):48–90, 1955.

J. C. Mattingly, A. M. Stuart, and D. J. Higham. Ergodicity for sdes and approximations: locally lipschitz vector fields and degenerate noise. *Stochastic processes and their applications*, 101(2):185–232, 2002.

J. C. Mattingly, N. S. Pillai, and A. M. Stuart. Diffusion limits of the random walk metropolis algorithm in high dimensions. *The Annals of Applied Probability*, 22(3):881–930, 2012.

M. I. McCarthy, G. R. Abecasis, L. R. Cardon, D. B. Goldstein, J. Little, J. P.A. Ioannidis, and J. N. Hirschhorn. Genome-wide association studies for complex traits: consensus, uncertainty and challenges. *Nature Reviews Genetics*, 9(5):356–369, 2008.

S. Minsker, S. Srivastava, L. Lin, and D. B. Dunson. Robust and scalable bayes via a median of subset posterior measures. *arXiv preprint arXiv:1403.2660*, 2014.

R. M. Neal. MCMC using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo: Methods and Applications*, page 113, 2010.

W. Neiswanger, C. Wang, and E. Xing. Asymptotically exact, embarrassingly parallel mcmc. *arXiv preprint arXiv:1311.4780*, 2013.

G. Pages and F. Panloup. Ergodic approximation of the distribution of a stationary diffusion: rate of convergence. *The Annals of Probability*, 22(3):1059–1100, 2012.

F. Panloup. Recursive computation of the invariant measure of a stochastic differential equation driven by a lévy process. *The Annals of Applied Probability*, 18(2):379–426, 2008.

S. Patterson and Y. W. Teh. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In *Advances in Neural Information Processing Systems*, 2013a.

S. Patterson and Y. W. Teh. Stochastic Gradient Riemannian Langevin dynamics on the probability simplex. In *Advances in Neural Information Processing Systems*, 2013b.

N. S. Pillai, A. M. Stuart, and A. H. Thiéry. Optimal scaling and diffusion limits for the Langevin algorithm in high dimensions. *The Annals of Applied Probability*, 22(6): 2320–2356, 2012.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951a.

H. Robbins and S. Monro. A Stochastic Approximation Method. *The Annals of Mathemaitcal Statistics*, 22:400–407, 1951b. ISSN 0003-4851.

G. O. Roberts and J. S. Rosenthal. Optimal Scaling of Discrete Approximations to Langevin Diffusions. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 60(1):255–268, 1998.

G. O. Roberts and O. Stramer. Langevin diffusions and Metropolis-Hastings algorithms. *Methodology and computing in applied probability*, 4(4):337–357, 2002.

G. O. Roberts and R. L. Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363, 1996.

Walter Rudin. *Real and complex analysis (3rd)*. New York: McGraw-Hill Inc, 1986.

A. Koratticara S. Ahn and M. Welling. Bayesian posterior sampling via stochastic gradient fisher scoring. In *ICML*, 2012.

M.-A. Sato. Online model selection based on the variational bayes. *Neural Computation*, 13(7):1649–1681, 2001.

Albert N Shiryaev. *Probability*. Graduate Texts in Mathematics, 1996.

Nathan Srebro and Ambuj Tewari. Stochastic optimization for machine learning. ICML Tutorial, 2010.

O. Stramer and R.L. Tweedie. Langevin-type models I: Diffusions with given stationary distributions and their discretizations*. *Methodology and Computing in Applied Probability*, 1(3):283–306, 1999a.

O. Stramer and R.L. Tweedie. Langevin-type models II: Self-targeting candidates for MCMC algorithms. *Methodology and Computing in Applied Probability*, 1(3):307–328, 1999b.

S. Thrun. Toward robotic cars. *Communications of the ACM*, 53(4):99–106, 2010.

W. Y.S. Wang, B. J. Barratt, D. G. Clayton, and J. A. Todd. Genome-wide association studies: theoretical and practical concerns. *Nature Reviews Genetics*, 6(2):109–118, 2005.

M. Welling and Y. W. Teh. Bayesian learning via Stochastic Gradient Langevin Dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 7/13; Revised 10/14; Published 3/16

Knowledge Matters: Importance of Prior Information for Optimization

Çağlar Gülçehre GULCEHRC@IRO.UMONTREAL.CA

Yoshua Bengio BENGIOY@IRO.UMONTREAL.CA
Département d'informatique et de recherche opérationnelle
Université de Montréal, Montréal, QC, Canada

Editor: Aaron Courville, Rob Fergus, and Christopher Manning

Abstract

We explored the effect of introducing prior knowledge into the intermediate level of deep supervised neural networks on two tasks. On a task we designed, all black-box state-of-the-art machine learning algorithms which we tested, failed to generalize well. We motivate our work from the hypothesis that, there is a training barrier involved in the nature of such tasks, and that humans learn useful intermediate concepts from other individuals by using a form of supervision or guidance using a curriculum. Our results provide a positive evidence in favor of this hypothesis. In our experiments, we trained a two-tiered MLP architecture on a dataset for which each input image contains three sprites, and the binary target class is 1 if all of three shapes belong to the same category and otherwise the class is 0. In terms of generalization, black-box machine learning algorithms could not perform better than chance on this task. Standard deep supervised neural networks also failed to generalize. However, using a particular structure and guiding the learner by providing intermediate targets in the form of intermediate concepts (the presence of each object) allowed us to solve the task efficiently. We obtained much better than chance, but imperfect results by exploring different architectures and optimization variants. This observation might be an indication of optimization difficulty when the neural network trained without hints on this task. We hypothesize that the learning difficulty is due to the *composition* of two highly non-linear tasks. Our findings are also consistent with the hypotheses on cultural learning inspired by the observations of training of neural networks sometimes getting stuck, even though good solutions exist, both in terms of training and generalization error.

Keywords: deep learning, neural networks, optimization, evolution of culture, curriculum learning, training with hints

1. Introduction

There is a recent emerging interest in different fields of science for *cultural learning* (Henrich and McElreath, 2003) and how groups of individuals that communicates within each other can learn in ways superior to solely individual learning. This is further witnessed by the emergence of new research fields such as "Social Neuroscience". Learning from other agents in an environment by the means of cultural transmission of knowledge with a peer-to-peer communication is an efficient and natural way of acquiring or propagating common knowledge. A popular belief on how the information is transmitted between individuals is that bits of information are transmitted by small units, called memes, which share some characteristics of genes, such as self-replication, mutation and response to selective pressures (Dawkins, 1976).

This paper investigates an aspect of the hypothesis (which is further elaborated in Bengio (2013a)) that human culture and the evolution of ideas have been crucial to counter an optimization issue: this difficulty would otherwise make it harder for human brains to capture high level knowledge of the world without the help of other educated humans. In this paper, machine learning experiments are used to explore some elements of this hypothesis by seeking answers for the following questions: are there machine learning tasks which are intrinsically hard for a lone learning agent, but those tasks may become easier when intermediate concepts are provided by another agent as an additional intermediate learning cues, in the spirit of curriculum learning (Bengio et al., 2009)? What makes learning such tasks more difficult? Can specific initial values of the neural-network parameters yield success when random initialization yield complete failure? Is it possible to verify that the problem being faced is an optimization problem or a regularization problem or does it influence both training and test behavior? These are the questions discussed (if not completely addressed) here, which relate to the following broader question: how can humans (and potentially one day, machines) learn complex concepts?

In the focus of this paper, results of different machine learning algorithms on an artificial learning task involving binary 64×64 images are presented. In that task, each image in the dataset contains 3 Pentomino tetris sprites (simple shapes). The task is to figure out if all the sprites in the image are the same or not. Several black-box state-of-the-art machine learning algorithms have been tested and none of them was able to perform better than a random predictor on the test set. Nevertheless, by providing hints about the intermediate concepts (the presence and location of particular sprite classes), can help a neural network to solve the task easily and fast. But the same model without the hints either fails to learn the task or learns it much more slowly and with substantially worse generalization accuracy. Surprisingly, our attempts at solving this problem with unsupervised pre-training algorithms also failed. For showing the impact of intermediate level guidance, we experimented with a two-tiered neural network, with supervised pre-training of the first part to recognize the category of sprites independently of their orientation and scale, at different locations, while the second part learns from the output of the first part and predicts the binary task of interest.

The objective of this paper is not to propose a novel learning algorithm or architecture, but rather to refine our understanding of the learning difficulties involved with composed tasks. Our results also bring empirical evidence in favor of some of the hypotheses from Bengio (2013a), as well as to introduce particular form of curriculum learning (Bengio et al., 2009) based on hints.

Building difficult AI problems has a long history in computer science. Hard AI problems are being studied to create CAPTCHAs that are easy to solve for humans, but hard to solve for machines (Von Ahn et al., 2003). In this paper we are investigating a difficult problem for the off-the-shelf black-box machine learning algorithms. The source code of some experiments presented in that paper is available at `https://github.com/caglar/kmatters`.

1.1 Curriculum Learning and Cultural Evolution for Effective Local Minima

What Bengio (2013a) calls an **effective local minimum**, is a point where iterative training stalls, either because of an actual local minimum or due to optimization algorithm is unable (in reasonable time) to find a descent path to a substantially better solution that exists. It is not yet clear why neural network training sometimes stalls in this way, although recent

work suggests that it would generally not be because of local minima (Dauphin et al., 2014; Choromanska et al., 2015). In this paper, we hypothesize that some more abstract learning tasks such as those obtained by composing simpler tasks are more likely to yield such effective local minima for neural networks, and are generally hard for general-purpose machine learning algorithms.

The idea that learning can be enhanced by guiding the learner through intermediate easier tasks is old, starting with animal training by *shaping* (Skinner, 1958; Peterson, 2004; Krueger and Dayan, 2009). Bengio et al. (2009) introduce a computational hypothesis related to a presumed issue with effective local minima when directly learning the target task: good solutions correspond to hard-to-find-by-chance effective local minima, and intermediate tasks prepare the learner's internal configuration (parameters) in a way that is similar to continuation methods in global optimization. A continuation method would go through a sequence of intermediate optimization problems, starting with a convex one where local minima are no issue, and gradually morphing into the target task of interest.

In a related vein, Bengio (2013a) makes the following inferences based on experimental observations about training of deep neural networks:

Point 1: Training deep architectures is easier when some hints are given about the function that the intermediate levels should compute (Hinton et al., 2006; Weston et al., 2008; Salakhutdinov and Hinton, 2009; Bengio, 2009). *The experiments performed here expand in particular on this point.*

Point 2: It is much easier to train a neural network with supervision (where examples are provided to it when a concept is present and not present in a variety of examples) than to expect unsupervised learning to discover the concept (which may also happen but usually leads to poorer renditions of the concept). *The poor results obtained here with unsupervised pre-training reinforce that hypothesis.*

Point 3: Directly training all layers of a deep network together does not only makes it more difficult to exploit all the extra modeling power of a deeper architecture but in some cases it can yield worse results as the number of *required layers* is increased (Larochelle et al., 2009; Erhan et al., 2010). *The experiments performed here also reinforce that hypothesis.*

Point 4: Erhan et al. (2010) observed that no two training trajectories ended up in the same effective local minimum, out of hundreds of runs, even when comparing solutions as functions from input to output, rather than in parameter space (thus eliminating from the picture the presence of symmetries and multiple local minima due to relabeling and other reparametrizations). This suggests that the number of different effective local minima (even when considering them only in function space) must be huge.

Point 5: Unsupervised pre-training changes the initial conditions of the descent procedure and it can allow to reach substantially better effective local minima (in terms of generalization error). But as empirically observed by (Erhan et al., 2010), better local minima do not appear to be reachable by chance alone on the same architectures. *The experiments performed here provide another piece of evidence in favor of the hypothesis that random initialization can yield rather poor results while specifically targeted initialization can have a drastic impact, that is, the effective local minima are not just numerous but that some small subset of them are much better and hard to reach by chance.* Nevertheless, recent work showed that rather deep feed-forward networks can be very successfully trained when large quantities of labeled data is available (Ciresan et al., 2010; Glorot et al., 2011; Krizhevsky et al., 2012). Nonetheless, the experiments reported here suggest that it all depends on the task and architecture being considered, since even with very large quantities of labeled examples, most the deep networks trained here were unsuccessful.

Based on the above points, Bengio (2013a) then proposed the following hypotheses regarding learning of high-level abstractions.

- **Deeper Networks are Harder Hypothesis:** Although solutions may exist, effective local minima are generally more likely to hamper learning as the required depth of the architecture increases.
- **Abstractions are Harder Hypothesis:** High-level abstractions are unlikely to be discovered by a single human learner by chance, because these abstractions are represented by a deep subnetwork of the brain, which learns by local descent, thus being sensitive to the issue associated with the above hypothesis.
- **Guided Learning Hypothesis:** A human brain can learn high level abstractions if guided by the signals produced by other agents that act as hints or indirect supervision for these high-level abstractions.
- **Memes Divide-and-Conquer Hypothesis:** Linguistic exchange, individual learning and the recombination of memes constitute an efficient evolutionary recombination operator in the meme-space. This helps human learners to *collectively* build better internal representations of their environment, including fairly high-level abstractions.

This paper particularly focuses on "*Point 1*" above and testing the "*Guided Learning Hypothesis*", using machine learning algorithms to provide experimental evidence. The experiments performed also provide evidence in favor of the "*Deeper Harder Hypothesis*" and are related to the "*Abstractions Harder Hypothesis*". Performance of machine learning algorithms are far from the current capabilities of humans on several tasks, and it is important to tackle the remaining obstacles to approach AI. For this purpose, the question to be answered is why on some tasks humans learn effortlessly from very few labeled examples, while machine learning algorithms fail miserably?

2. Optimization Difficulty of Learning High-level Concepts

As hypothesized in the "*Local Descent Hypothesis*", human brains would rely on a local approximate descent, just like a Multi-Layer Perceptron trained by a gradient-based iterative optimization. The main argument in favor of this hypothesis relies on the biologically-grounded assumption that although firing patterns in the brain change rapidly, synaptic strengths underlying these neural activities change only gradually, making sure that behaviors are generally consistent across time. If a learning algorithm is based on a form of local (for example gradient-based) descent, it can be sensitive to effective local minima (Bengio, 2013a).

When one trains a neural network, at some point in the training phase the evaluation of error seems to saturate, even if new examples are introduced. In particular Erhan et al. (2010) find that early examples have a much larger weight in the final solution. It looks like the learner

is stuck in or near a local minimum. But since it is difficult to verify if this is near a true local minimum or simply an effect of strong ill-conditioning or a saddle-point, we call such a "stuck" configuration an *effective local minimum*, whose definition depends not just on the optimization objective but also on the limitations of the optimization algorithm.

Erhan et al. (2010) highlighted both the issue of effective local minima and a regularization effect when initializing a deep network with unsupervised pre-training. Interestingly, as the network gets deeper the difficulty due to effective local minima seemed to be get more pronounced in these experiments. That might be because the number of effective local minima increases (more like an actual local minima issue), or maybe because the good ones are harder to reach (more like an ill-conditioning issue) and more work will be needed to clarify this question.

As a result of Point 4 we hypothesize that it is very difficult for an individual's brain to discover some higher level abstractions by chance only. As mentioned in the "*Guided Learning Hypothesis*" humans get hints from other humans and learn high-level concepts with the guidance of other humans. But some high-level concepts may also be hardwired in the brain, as assumed in the universal grammar hypothesis (Montague, 1970), or in nature vs nurture discussions in cognitive science. Curriculum learning (Bengio et al., 2009) and incremental learning (Solomonoff, 1989), are specific examples of this phenomena. Curriculum learning is done by properly choosing the sequence of examples seen by the learner, where simpler examples are introduced first and more complex examples shown when the learner is ready for them. One of the hypotheses on why a curriculum works states that curriculum learning acts as a continuation method that allows one to discover a good minimum: continuation methods first find a good minimum of a smoother objective function and then gradually change the objective function towards the desired one, while tracking local minima along the way. Recent experiments on human subjects also suggest that humans *teach* by using a curriculum strategy (Khan et al., 2011).

Some parts of the human brain are known to have a hierarchical organization (for example, the visual cortex) consistent with the deep architecture studied in the machine learning literature. As a stimulus is transmitted from the sensory level to higher levels of the visual cortex, higher level areas tend to respond to stimuli in a way corresponding to the detection of more concepts. This is consistent with the *Deep Abstractions Hypothesis*.

Training neural networks and machine learning algorithms by decomposing the learning task into sub-tasks and exploiting prior information about the task is well-established and in fact constitutes the main approach to solving industrial problems with machine learning. The contribution of this paper is rather on rendering explicitly, the effective local minima issue and providing evidence on the type of problems for which this difficulty arises. This prior information and hints given to the learner can be viewed as an inductive bias for a particular task, an important ingredient to obtain a good generalization error (Mitchell, 1980). An interesting earlier finding in that line of research was done with Explanation Based Neural Networks (EBNN) in which a neural network transfers knowledge across multiple learning tasks. An EBNN uses previously learned domain knowledge as an initialization or search bias (that is to constrain the learner in the parameter space) (O'Sullivan, 1996; Mitchell and Thrun, 1993).

Another related work in machine learning is mainly focused on reinforcement learning algorithms, based on incorporating prior knowledge in terms of logical rules to the learning algorithm as a prior knowledge to speed up and bias learning (Kunapuli et al., 2010; Towell and Shavlik, 1994). Also defining sub-tasks and sub-goals to guide the agent is a well-established principle in hierarchical reinforcement learning (Barto and Mahadevan, 2003).

3. Experimental Setup

Some tasks, which seem reasonably easy for humans to learn (keeping in mind that humans can exploit prior knowledge, either from previous learning or innate knowledge), are nonetheless appearing almost impossible to learn for current generic state-of-art machine learning algorithms.

Here we study more closely such a task, which becomes learnable if one provides hints to the learner about appropriate intermediate concepts. Interestingly, the task we used in our experiments is not only hard for deep neural networks but also for non-parametric machine learning algorithms such as SVMs, boosting and decision trees.

The result of the experiments for varying size of dataset with several off-the-shelf black box machine learning algorithms and some popular deep learning algorithms are provided in Table 4. The detailed explanations about the algorithms and the hyperparameters used for those algorithms are given in the Appendix Section B. We also provide some explanations about the methodologies conducted for the experiments at Section 3.3.

3.1 Pentomino Dataset

In order to test our hypothesis, we designed an artificial dataset for object recognition using 64×64 binary images. The source code for the script that generates the artificial Pentomino datasets (Arcade-Universe) is available at: `https://github.com/caglar/Arcade-Universe`. This implementation is based on Olivier Breuleux's bugland dataset generator. If the task is two tiered (i.e., with guidance provided), the task in the first part is to recognize and locate each Pentomino object class[1] in the image. The second part/final binary classification task is to figure out if all the Pentominoes in the image are of the same shape class or not. If a neural network learned to detect the categories of each object at each location in an image, the remaining task becomes an XOR-like operation over the detected object categories. The types of Pentomino objects that is used for generating the dataset are illustrated in Figure 1 and correspond to Pentomino sprites N, P, F, Y, J, and Q, along with the Pentomino N2 sprite (mirror of "Pentomino N" sprite), the Pentomino F2 sprite (mirror of "Pentomino F" sprite), and the Pentomino Y2 sprite (mirror of "Pentomino Y" sprite).

Figure 1: Different classes of Pentomino shapes used in our dataset.

As shown in Figures 2(a) and 2(b), the synthesized images are fairly simple and do not have any texture. Foreground pixels are "1" and background pixels are "0". Images of the training and test sets are generated iid. For notational convenience, we assume that the domain of raw input images is X, the set of sprites is S, the set of intermediate object categories for each possible location in the image is denoted as Y and the set of possible final binary task outcomes is $Z = \{0, 1\}$. Two different types of rigid body transformation is performed: sprite rotation, $rot(X, \gamma)$,where the set of all rotations is $\Gamma = \{\gamma\text{: } (\gamma = 90 \times \phi) \wedge [(\phi \in \mathbb{N}), (0 \leq \phi \leq 3)]\}$ and sprite scaling, $scale(X, \alpha)$, where $\alpha \in \{1, 2\}$ is the scaling factor. The data generating procedure is summarized below.

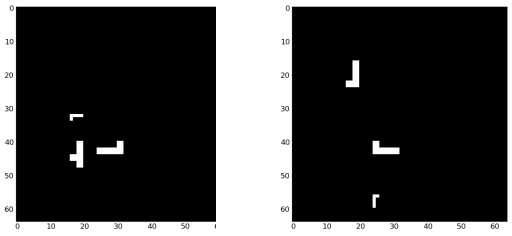
1. A human learner does not seem to need to be taught the shape categories of each Pentomino sprite in order to solve the task. On the other hand, humans have lots of previously learned knowledge about the notion of shape and how central it is in defining categories.

Sprite transformations: Before placing the sprites in an empty image, for each image $x \in X$, a value for $z \in Z$ is randomly sampled whether to have (or not) the same three Pentomino sprite shapes in the image. Conditioned on the constraint given by z, three sprites s_{ij} at location $(i,\ j)$ are randomly selected from the set S without replacement. Using a uniform probability distribution over all possible scales, a scale is chosen and accordingly each sprite in the image is scaled. Then each sprite is randomly rotated by a multiple of 90 degrees.

Sprite placement: Upon completion of sprite transformations, a 64×64 uniform grid is generated which is divided into 8×8 blocks, each block being of size 8×8 pixels, three different blocks are randomly selected from the $64 = 8 \times 8$ on the grid and the transformed objects are each placed into one of the three selected blocks. Consequently, the objects cannot overlap, by construction.

Each sprite is centered in the block in which it is located. Thus there is no object translation inside the blocks. The translation invariance can be achieved by being invariant to the location of the block inside the image.

A Pentomino sprite is guaranteed to not overflow the block in which it is located, and there are no collisions or overlaps between sprites, making the task simpler. The largest possible Pentomino sprite can actually be fit into an 8×4 mask.



(a) sprites, not all same type (b) sprites, all same type

Figure 2: Left (a): An example image from the dataset which has *a different sprite type* in it. Right (b): An example image from the dataset that has only one type of Pentomino object in it, but with different orientations and scales.

3.2 Control Experiments

To explore the effect of changing the complexity of the input representation on the difficulty of the task, a set of experiments have been designed with symbolic representations of the information in each patch. In all cases an empty patch is represented with a vector of all elements 0. In this section we are considering alternative representations to raw image representation for an abstract task similar to Pentomino. Basically we are seeking for an ideal representation that can be fed as an input to a regular feedforward MLP or another black-box classifier to perform well.

The following four experiments have been conducted, each one of them using a different input representation for each patch.

Experiment 1: One-hot representation without transformations: In this experiment several trials is done with a 10-input one-hot vector per patch. Each input corresponds to an object category given explicitly (corresponding to one input bit).

Experiment 2: Disentangled representations: In this experiment, we did trials with 16 binary inputs per patch, 10 one-hot bits for representing each object category, 4 for rotations and 2 for scaling, that is, the whole information about the input is given, but it is perfectly disentangled. This would ideally be reproduced by an unsupervised learning algorithm over each patch, if it was able to perfectly disentangle the image representation.

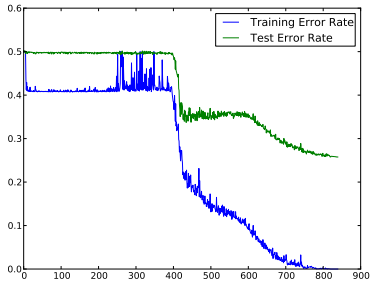
Experiment 3: One-hot representation with transformations: For each of the ten object types there are $8 = 4 \times 2$ possible transformations. Two objects in two different patches are the considered "the same" (for the final task) if their category is the same regardless of the transformations. The one-hot representation of a patch corresponds to the cross-product between the 10 object shape classes and the 4×2 transformations, that is, one out of $80 = 10 \times 4 \times 2$ possibilities represented in an 80-bit one-hot vector. This also contains all the information about the input image patch, but spread out in a kind of non-parametric and non-informative (not disentangled) way, like a perfect memory-based unsupervised learner (like clustering) could produce. Nevertheless, the shape class would be easier to read out from this representation than the image representation (it could be obtained by forming an OR over 8 of the bits).

Experiment 4: One-hot representation with 80 choices: This representation has the same 1 of 80 one-hot representation per patch but the target task is defined differently. Two objects in two different patches are considered the same iff they have exactly the same 80-bit one-hot representation (that are of the same object category with the same transformation applied).

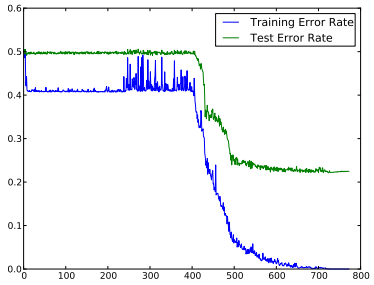
The first experiment is a sanity check. It was conducted with single hidden-layered MLPs using rectifier and tanh nonlinearity, and the task was learned perfectly (0 error on both training and test dataset) with very few training epochs.

The results of Experiment 2 are given in Table 1. To improve our results, we experimented with the Maxout non-linearity in a feedforward MLP (Goodfellow et al., 2013) with two hidden layers. Unlike the typical Maxout network mentioned in the original paper, regularizers have been deliberately avoided in order to focus on the optimization issue, i.e: no weight decay, norm constraint on the weights, or dropout. Although learning from a disentangled representation is more difficult than learning from perfect object detectors, it is feasible with some architectures such as the Maxout network. Note that this representation is the kind of representation that one could hope an unsupervised learning algorithm could discover, at best, as argued in Bengio et al. (2012).

The only results obtained on the validation set for Experiment 3 and Experiment 4 are shown respectively in Table 2 and Table 3. In these experiments a tanh MLP with two hidden layers have been tested with the same hyperparameters. In experiment 3 the complexity of the problem comes from the transformations ($8 = 4 \times 2$) and the number of object types.



(a) Training and Test Errors for Experiment 4



(b) Training and Test Errors for Experiment 3

Figure 3: tanh MLP training curves. Left (a): The training and test errors of Experiment 3 over 800 training epochs with 100k training examples using tanh MLP. Right (b):The training and test errors of Experiment 4 over 700 training epochs with 100k training examples using tanh MLP.

But in experiment 4, the only source of complexity of the task comes from the number of different object types. These results are in between the complete failure and complete success observed with other experiments, suggesting that the task could become solvable with better training or more training examples. Figure 3 illustrates the progress of training a tanh MLP, on both the training and test error, for Experiments 3 and 4. Clearly, MLPs were able to make significant progress on the task, but the task is far from being solved completely. On experiment 3 for both maxout and tanh, there was a long plateau where the training error and objective stays almost same. Maxout did just chance on the experiment for about 120 iterations on the training and the test set. But after 120^{th} iteration, the training and test error started to decline and eventually it was able to solve the task. Moreover as seen from the curves in Figure 3(a) and 3(b), the training and test error curves are almost the same for both tasks. This implies that for one-hot inputs, whether you increase the number of possible transformations for each object or the number of object categories, as soon as the number of possible configurations is same, the complexity of the problem is almost the same for the MLP.

Learning Algorithm	**Training Error**	**Test Error**
SVM	0.0	35.6
Random Forests	1.29	40.475
Tanh MLP	0.0	0.0
Maxout MLP	0.0	0.0

Table 1: Performance of different learning algorithms on disentangled representation in Experiment 2.

Learning Algorithm	**Training Error**	**Test Error**
SVM	11.212	32.37
Random Forests	24.839	48.915
Tanh MLP	0.0	22.475
Maxout MLP	0.0	0.0

Table 2: Performance of different learning algorithms using a dataset with one-hot vector and 80 inputs as discussed for Experiment 3.

Learning Algorithm	**Training Error**	**Test Error**
SVM	4.346	40.545
Random Forests	23.456	47.345
Tanh MLP	0	25.8

Table 3: Performance of different algorithms using a dataset with one-hot vector and 80 binary inputs as discussed in Experiment 4.

3.3 Learning Algorithms Evaluated

Initially the models are cross-validated by using 5-fold cross-validation. With $40,000$ examples, this gives $32,000$ examples for training and $8,000$ examples for testing. For training neural networks, we used stochastic gradient descent (SGD). The following standard learning algorithms were first evaluated: decision trees, SVMs with Gaussian kernel, ordinary fully-connected Multi-Layer Perceptrons, Random Forests, k-Nearest Neighbors, Convolutional Neural Networks, and Stacked Denoising Auto-Encoders with supervised fine-tuning. More details of the configurations and hyper-parameters for each of them are given in Appendix Section B. We obtained significantly better than chance results with only variations of the Structured Multi-Layer Perceptron that is described in this paper.

3.3.1 Structured Multi-Layer Perceptron (SMLP)

The neural network architecture that is used to solve this task is called the SMLP (Structured Multi-Layer Perceptron), a deep neural network with two parts as illustrated in Figure 4 and 6.

The bottom part, P1NN (*Part 1 Neural Network*, as it is called in the rest of the paper), has shared weights and local connectivity, with one identical MLP instance of the P1NN for each patch of the image, and typically an 11-element output vector per patch (unless otherwise noted). The idea is that these 11 outputs per patch could represent the detection of the sprite shape category (or the absence of sprite in the patch). The upper part, P2NN (*Part 2 Neural Network*) is a fully connected one hidden layer MLP that takes as input the concatenation of the outputs of the patch-wise P1NNs over all the patches. Note that the first layer of P1NN is similar to a convolutional layer but where the stride equals the kernel size, such that windows do not overlap, that is, P1NN can be decomposed into separate networks sharing the same parameters but applied on different patches of the input image, so that each network can

actually be trained patch-wise in the case where a target is provided for the P1NN outputs. The P1NN output for patch $\mathbf{p_i}$ which is extracted from the image $\mathbf{x}$ is computed as follows:

$$\mathrm{f}_\theta(\mathbf{p_i}) = \mathrm{g}_2(\mathbf{V}\mathrm{g}_1(\mathbf{U}\mathbf{p_i} + \mathbf{b}) + \mathbf{c}) \tag{1}$$

where $\mathbf{p_i} \in \mathcal{R}^d$ is the input patch/receptive field extracted from location i of a single image. $\mathbf{U} \in \mathcal{R}^{d_h \times d}$ is the weight matrix for the first layer of P1NN and $\mathbf{b} \in \mathcal{R}^d_h$ is the vector of biases for the first layer of P1NN. $\mathrm{g}_1(\cdot)$ is the activation function of the first layer and $\mathrm{g}_2(\cdot)$ is the activation function of the second layer. In many of the experiments, best results were obtained with $\mathrm{g}_1(\cdot)$ a rectified linear unit as non-linearity (a.k.a. as ReLU), which is $max(0,\ X)$ (Jarrett et al., 2009; Nair and Hinton, 2010; Glorot et al., 2011). $\mathbf{V} \in \mathcal{R}^{d_h \times d_o}$ is the second layer's weights matrix and $\mathbf{c} \in \mathcal{R}_{d_o}$ are the biases of the second layer of the P1NN, with d_o expected to be smaller than d_h.

In this way, $\mathrm{g}_1(\mathbf{U}\mathbf{p_i} + \mathbf{b})$ is an overcomplete representation of the input patch that can potentially represent all the possible Pentomino shapes for all factors of variations in the patch (rotation, scaling and Pentomino shape type). On the other hand, when trained with hints, $\mathrm{f}_\theta(\mathbf{p_i})$ is expected to be the lower dimensional representation of a Pentomino shape category invariant to scaling and rotation in the given patch.

In the experiments with SMLP trained with hints the P1NN is applied to each 8×8 patch and expected to predict it shape category (or the absence of a Pentomino object in the patch). The input representation of P2NN is obtained from the concatenated output of P1NN across all the 64 patch locations:

$\mathbf{h_o} = [\mathrm{f}_\theta(\mathbf{p_0}), \cdots, \mathrm{f}_\theta(\mathbf{p_i}), \cdots, \mathrm{f}_\theta(\mathbf{p_N}))]$ where N is the number of patches and the $\mathrm{h}_o \in \mathcal{R}^{d_i}, d_i = d_o \times N$. $\mathbf{h_o}$ is the concatenated output of the P1NN at each patch.

$\mathbf{h_o}$ is standardized for each training and test sample separately, so that the per-feature mean and variance across examples are respectively 0 and 1. That standardization layer seems to be crucial for successfully training the SMLP. Details of the standardization layer are given in Section A.

The concatenated output of P1NN is fed as an input to the P2NN. P2NN is a feedforward MLP with a sigmoid output layer using a single rectifier hidden layer. The task of P2NN is to perform a nonlinear logical operation on the representation provided at the output of P1NN.

3.3.2 Structured Multi Layer Perceptron Trained with Hints (SMLP-hints)

The SMLP-hints architecture exploits a hint about the presence and category of Pentomino objects, specifying a semantics for the P1NN outputs. P1NN is trained with the intermediate target Y, specifying the type of Pentomino sprite shape present (if any) at each of the 64 patches (8×8 non-overlapping blocks) of the image. Because a possible answer at a given location can be "none of the object types" that is, an empty patch, y_p (for patch p) can take one of the 11 possible values, 1 for rejection and the rest for the Pentomino shape classes, illustrated in Figure 1:

$$y_p = \begin{cases} 0 & \text{if patch } p \text{ is empty} \\ s \in S & \text{if the patch } p \text{ contains a Pentomino sprite .} \end{cases}$$

A similar task has been studied by Fleuret et al. (2011) (at SI appendix Problem 17), who compared the performance of humans vs computers.

The SMLP-hints architecture takes advantage of dividing the task into two subtasks during training with prior information about intermediate-level relevant factors. Because the sum of the training losses decomposes into the loss on each patch, the P1NN can be pre-trained patch-wise. Each patch-specific component of the P1NN is a fully connected MLP with 8×8 inputs and 11 outputs with a softmax output layer. SMLP-hints uses the standardization given in Equation 8 but with $\epsilon = 0$.

In general, standardization layer for SMLP seems to make the training much easier and the distribution of activations of P1NN becomes much more peaky.

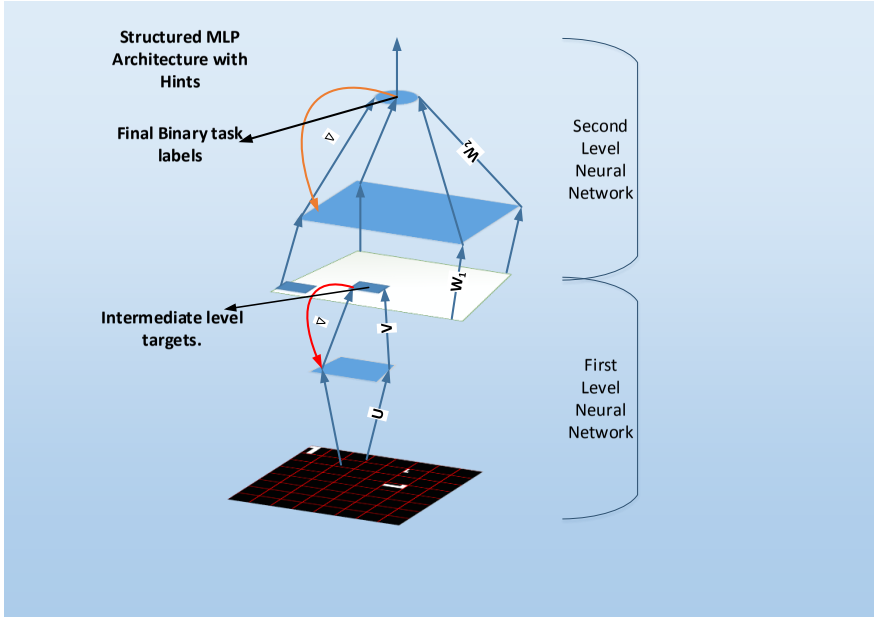


Figure 4: Structured MLP architecture, used with hints (trained in two phases, first P1NN, bottom two layers, then P2NN, top two layers). In SMLP-hints, P1NN is trained on each $8x8$ patch extracted from the image and the softmax output probabilities of all 64 patches are concatenated into a 64×11 vector that forms the input of P2NN. Only $\mathbf{U}$ and $\mathbf{V}$ parameters are learned in the P1NN and its output on each patch is fed into P2NN. The first level and the second level neural networks are trained separately, not jointly.

By default, the SMLP uses rectifier hidden units as activation function: we found a significant boost by using rectifier compared to hyperbolic tangent and sigmoid activation functions. The P1NN has a highly overcomplete architecture with 1024 hidden units per patch, and L1 and L2 weight decay regularization coefficients on the weights (not the biases) are respectively $1e-6$ and $1e-5$. The learning rate for the P1NN is 0.75. 1 training epoch was enough for the P1NN to learn the features of Pentomino shapes perfectly on the 40000 training examples. The

P2NN has 2048 hidden units. L1 and L2 penalty coefficients for the P2NN are 1e-6, and the learning rate is 0.1. These were selected by trial and error based on validation set error. Both P1NN (for each patch) and P2NN are fully-connected neural networks, even though P1NN globally is a special kind of convolutional neural network.

Filters of the first layer of SMLP are shown in Figure 5. These are the examples of the filters obtained with the SLMP-hints trained with 40k examples, whose results are given in Table 4. Those filters look very noisy but they work perfectly on the Pentomino task.

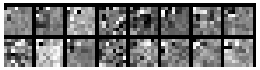


Figure 5: Filters of Structured MLP architecture, trained with hints on 40k examples.

3.3.3 Deep and Structured Supervised MLP without Hints (SMLP-nohints)

SMLP-nohints uses the same connectivity pattern (and deep architecture) that is also used in the SMLP-hints architecture, but without using the intermediate targets (Y). It directly predicts the final outcome of the task (Z), using the same number of hidden units, the same connectivity and the same activation function for the hidden units as SMLP-hints. 120 hyperparameter values have been evaluated by randomly selecting the number of hidden units from $[64, 128, 256, 512, 1024, 1200, 2048]$ and randomly sampling 20 learning rates uniformly in the log-domain within the interval of $[0.008, 0.8]$. Two fully connected hidden layers with 1024 hidden units (same as P1NN) per patch is used and 2048 (same as P2NN) for the last hidden layer, with twenty training epochs. For this network the best results are obtained with a learning rate of 0.05. The source code of the structured MLP is available at the GitHub repository: `https://github.com/caglar/structured_mlp`.

We decided to experiment with various SMLP-nohint architectures and optimization procedures, trying unsuccessfully to achieve as good results with SMLP-nohint as with SMLP-hints.

Rectifier Non-Linearity A rectifier nonlinearity is used for the activations of MLP hidden layers. We observed that using piecewise linear nonlinearity activation function such as the rectifier can make the optimization more tractable.

Intermediate Layer

The output of the P1NN is considered as an intermediate layer of the SMLP. For the SMLP-hints, only softmax output activations have been tried at the intermediate layer, and that sufficed to learn the task. Since things did not work nearly as well with the SMLP-nohints, several different activation functions have been tried: softmax$(\cdot)$, tanh$(\cdot)$, sigmoid$(\cdot)$ and linear activation functions.

Adaptive Learning Rates We have experimented with several different adaptive learning rate algorithms. We tried RMSprop (Tieleman and Hinton, 2012), Adadelta (Zeiler, 2012), Adagrad (Duchi et al., 2011) and a linearly $(\frac{1}{t})$ decaying learning rate (Bengio, 2013b). For the SMLP-nohints with sigmoid activation function we have found Adadelta(Zeiler, 2012) converging faster to an effective local minimum and usually yielding better generalization error compared to the others.

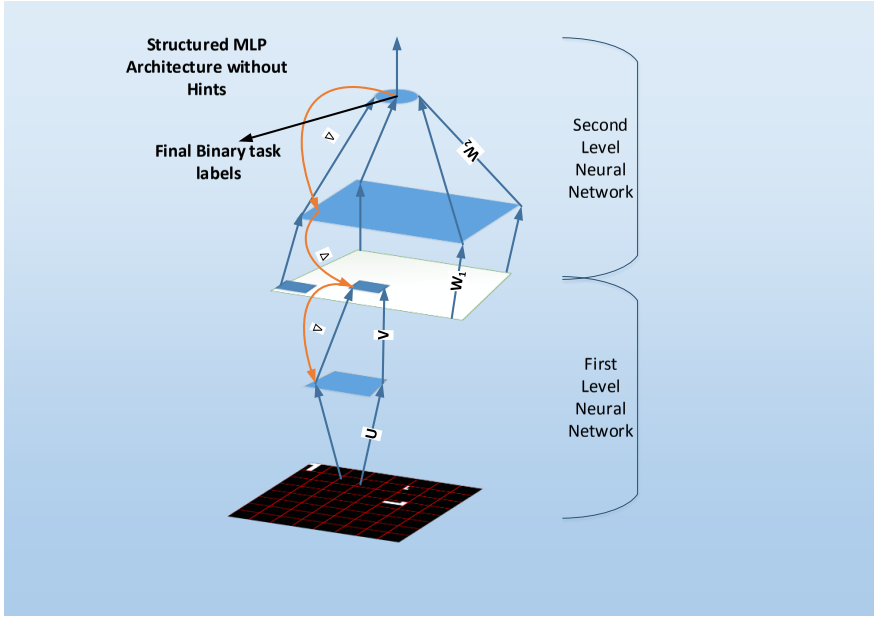


Figure 6: Structured MLP architecture, used without hints (SMLP-nohints). It is the same architecture as SMLP-hints (Figure 4) but with both parts (P1NN and P2NN) trained jointly with respect to the final binary classification task.

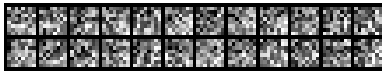


Figure 7: First layer filters learned by the Structured MLP architecture, trained without using hints on 447600 examples with online SGD and a sigmoid intermediate layer activation.

3.3.4 Deep and Structured MLP with Unsupervised Pre-Training

Several experiments have been conducted using an architecture similar to the SMLP-nohints, but by using unsupervised pre-training of P1NN, with Denoising Auto-Encoders (DAE) and/or Contractive Auto-Encoders (CAE). Supervised fine-tuning proceeds as in the deep and structured MLP without hints. Because an unsupervised learner may not focus the representation just on the shapes, a larger number of intermediate-level units at the output of P1NN has been explored: previous work on unsupervised pre-training generally found that larger hidden layers were optimal when using unsupervised pre-training, because not all unsupervised features will be relevant to the task at hand. Instead of limiting to 11 units per patch, we experimented with networks with up to 20 hidden (i.e., code) units per patch in the second-layer patch-wise auto-encoder.

In Appendix B we also provided the result of some experiments with binary-binary RBMs trained on 8×8 patches from the 40k training dataset.

For the second P1NN layer, a DAE with rectifier hidden units was trained with L1 sparsity and weight decay on the weights of the auto-encoder. The greedy layerwise unsupervised pre-training procedure is used to train the deep auto-encoder architecture (Bengio et al., 2007). In unsupervised pretraining experiments, tied weights have been used. Different combinations of CAE and DAE for unsupervised pretraining have been tested, but none of the configurations tested managed to learn the Pentomino task, as shown in Table 4. In terms of generalization, without hints for the permutation-invariant Pentomino dataset, an MLP using L_p units introduced in (Gulcehre et al., 2013) is the best performing model in Table 4.

Algorithm	20k dataset		40k dataset		80k dataset	
	Training Error	Test Error	Training Error	Test Error	Training Error	Test Error
SVM RBF	26.2	50.2	28.2	50.2	30.2	49.6
K Nearest Neighbors	24.7	50.0	25.3	49.5	25.6	49.0
Decision Tree	5.8	48.6	6.3	49.4	6.9	49.9
Randomized Trees	3.2	49.8	3.4	50.5	3.5	49.1
MLP	26.5	49.3	33.2	49.9	27.2	50.1
Convnet/Lenet5	50.6	49.8	49.4	49.8	50.2	49.8
Maxout Convnet	14.5	49.5	0.0	50.1	0.0	44.6
2 layer sDA	49.4	50.3	50.2	50.3	49.7	50.3
Supervised SMLP-nohints	0.0	48.6	0.0	36.0	0.0	12.4
Large SMLP-nohints	-	-	-	-	5.3	6.7
SMLP-nohints+CAE Supervised Finetuning	50.5	49.7	49.8	49.7	50.3	49.7
SMLP-nohints+CAE+DAE, Supervised Finetuning	49.1	49.7	49.4	49.7	50.1	49.7
SMLP-nohints+DAE+DAE, Supervised Finetuning	49.5	50.3	49.7	49.8	50.3	49.7
L_p Units	0.0	50.5	0.0	45.0	0.4	31.85
SMLP with Hints	**0.21**	**30.7**	**0**	**3.1**	**0**	**0.01**

Table 4: The error percentages with different learning algorithms on Pentomino dataset with different number of training examples.

3.4 Does the Effect Persist with Larger Training Set Sizes?

The results shown in this section indicate that the problem in the Pentomino task clearly is not just a regularization problem, but also involves an optimization difficulty. This is suggested by the experiments in which the training set size is increased (eventually to the point of doing online learning and never repeating the same example twice), without solving the problem. In the online mini-batch setting, parameter updates are performed as follows:

$$\theta_{t+1} = \theta_t - \Delta_{\theta_t}, \tag{2}$$

$$\Delta_{\theta_t} = \epsilon \frac{\sum_i^N \nabla_{\theta_t} \mathcal{L}(x_i, \theta_t)}{N}. \tag{3}$$

where $L(x_i, \theta_t)$ is the loss incurred on the i-th example x_i of the minibatch, with parameters θ_t, $t \in \mathcal{Z}^+$, N is the number of examples in the mini-batch, and ϵ is the learning rate.

Ordinary batch learning algorithms converge linearly to the optimum θ^*, however the noisy gradient estimates in the online SGD will cause parameter θ to fluctuate near the local optima. On the other hand, online SGD directly optimizes the expected risk, because the examples are drawn iid from the ground-truth distribution (Bottou, 2010). Thus:

$$\mathcal{L}_\infty = \mathrm{E}[\mathcal{L}(\mathbf{x}, \theta)] = \int_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \theta) p(\mathbf{x}) d\mathbf{x}, \tag{4}$$

where $\mathcal{L}_\infty$ is the generalization error. Therefore online SGD is trying to minimize the expected risk with noisy updates. Those noisy updates can have regularization effect:

$$\Delta_{\theta_t} = \epsilon \frac{\sum_t^N \nabla_{\theta_t} \mathcal{L}(x_t, \theta_t)}{N} = \epsilon \nabla_{\theta_t} \mathcal{L}(\mathbf{X}, \theta_t) + \epsilon \xi_t. \tag{5}$$

where $\nabla_{\theta_t} \mathcal{L}(\mathbf{X}, \theta_t)$ is the true gradient and ξ_t is the zero-mean stochastic gradient "noise" due to computing the gradient over a finite-size mini-batch sample.

An SMLP-nohints model was trained by online SGD with a randomly generated online Pentomino data stream. We used Adadelta (Zeiler, 2012) on mini-batches of 100 examples. In the online SGD experiments, two SMLP-nohints that are trained with and without standardization at the intermediate layer have the same hyperparameters. The SMLP-nohints P1NN patch-wise submodel has 2048 hidden units and the SMLP intermediate layer has $1152 = 64 \times 18$ units. The intermediate layer nonlinearity is sigmoid. P2NN has 2048 hidden units.

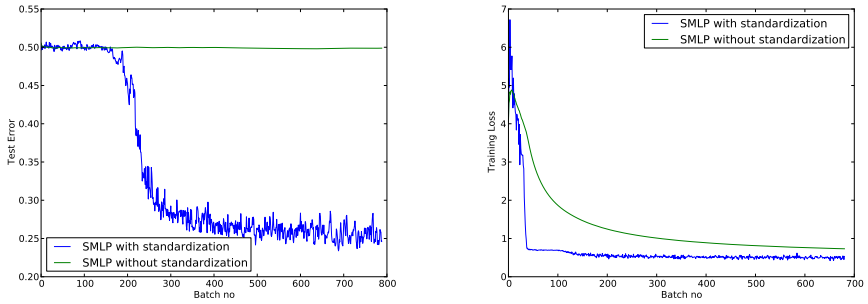
We trained SMLP-nohints both with and without standardization at the intermediate layer. The experiments illustrated in Figures 8(a) and 8(b) are using the same architecture as in the SMLP experiments reported in Table 4. The figures show the training and test results when training on the stream of 545400 randomly generated Pentomino samples.

Training of SMLP-nohints is done with mini-batch Adadelta and standardization in the intermediate layer, on 1046000 training examples from a randomly generated Pentomino data stream. At the end of the training, test error went down to 27.5%.

In SMLP-nohints experiment *without standardization*, the model is trained with the 1580000 Pentomino examples using online mini-batch SGD. P1NN has 2048 hidden units and 16 sigmoid output units per patch. P2NN has 1024 hidden units for the hidden layer. We used Adadelta to automatically adapt the learning rate. At the end of training this SMLP, the test error remained stuck, at 50.1%.

3.4.1 Experiments with Larger Training Sets

We consider the effect of training different learners with different numbers of training examples. For the experimental results shown in Table 4, we used 3 training set sizes of 20k, 40k and 80k examples. We generated each dataset with different random seeds (so they do not overlap). Figure 9 also shows the error bars for an ordinary MLP with three hidden layers, for a larger range of training set sizes, between 40k and 320k examples. The number of training epochs is 8 (more did not help), and there are 3 hidden layers with 2048 feature detectors. Learning rate



(a) (i) Test Error of SMLP-nohints (b) (ii) Training Error of SMLP-nohints

Figure 8: Online training and test errors of SMLP-nohints with and without standardization in the intermediate layer. Sigmoid nonlinearity has been used as an intermediate layer activation function. The x-axis is in units of blocks of 400 examples in the training set.

is 0.01 with tanh activation function. We used both L1, L2 penalty on weights and they are both $1e-6$.

Decision trees and SVMs can overfit the training set but they could not generalize on the test set. Note that the results reported in Table 4 are with hyper-parameters selected based on validation set error, and with early-stopping. Lower training errors are possible by avoiding regularizations and using large enough models. On the training set, an MLP with two large hidden layers (several thousands) could reach to almost 0% training error, but still it did not manage to achieve a good test error.

In the experimental results shown in Figure 9, we evaluated the impact of adding more training data for the fully-connected MLP. In those experiments, we used an MLP with three hidden layers where each layer has 2048 hidden units. The tanh$(\cdot)$ activation function is used with 0.05 learning rate and mini-batches of size 200.

As seen on Figure 9, adding more training examples did not help for both training and test error. This is reinforcing the hypothesis that the difficulty encountered is one of optimization, not of regularization.

3.5 Experiments on Effect of Initializing with Hints

Initialization of the parameters in a neural network can have a big impact on the learning and generalization (Glorot and Bengio, 2010). Previously Erhan et al. (2010) showed that initializing the parameters of a neural network with unsupervised pretraining guides the learning towards basins of attraction of effective local minima that provide better generalization. In this section we analyze the effect of hints to initialize the SMLP, continuing training without hints. The SMLP is pretrained for 1 epoch using the hints and then for 60 epochs without hints on the 40k examples of training set. We also compared the same architecture with the same hyperparameters, against the SMLP-nohints trained for 61 iterations on the same dataset. After one iteration of hint-based training SMLP obtained 9% training error and 39% test error.

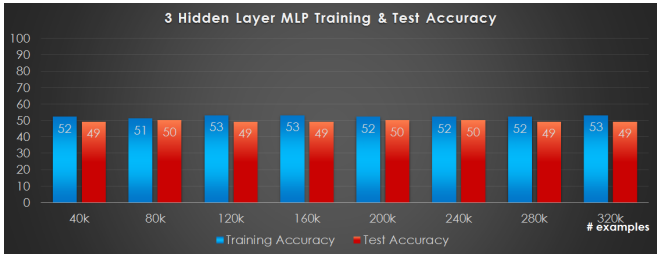


Figure 9: Training and test error bar charts for a regular MLP with 3 hidden layers. There is no significant improvement on the generalization error of the MLP as the new training examples are introduced.

Following the hint based training, SMLP is trained without hints for 60 epochs, but at epoch 18, it already got 0% training and 0% test error. The hyperparameters for this experiment and the experiment that the results shown for the SMLP-hints in Table 4 are the same. Figure 10 suggests that initializing with hints can give the same generalization performance but training takes longer.

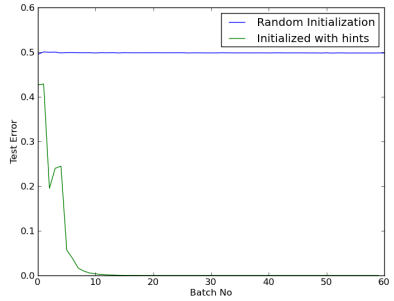


Figure 10: Plots showing the test error of SMLP with random initialization vs initializing with hint based training.

3.6 Experiments on Optimization for Pentomino Dataset

With extensive hyperparameter optimization and using standardization in the intermediate level of the SMLP with softmax nonlinearity, SMLP-nohints was able to get 5.3% training and 6.7% test error on the $80k$ Pentomino training dataset. The key ingredient was the use of a larger hidden layer. We used 2050 hidden units in the P1NN, 11 softmax outputs per patch, and 1024 hidden units in the P2NN. The network was trained with a learning rate 0.1 without using any adaptive learning rate. The SMLP uses a rectifier nonlinearity for hidden layers of both P1NN and P2NN. We also applied a small amount of L_1 and L_2 regularization on the weights of the network.

An MLP with 2 hidden layers, each 1024 rectifier units, was trained using **LBFGS** (the implementation from the scipy.optimize library) on $40k$ training examples, with gradients computed on batches of 10000 examples at each iteration. However, after convergence of training, the MLP was still doing chance on the test dataset.

In order to observe the effect of introducing the intermediate level hints into the SMLP for optimization, we compared the learning curves of the same SMLP model trained with and without hints. To train the SMLP with hints, we jointly optimize the loss for hints $\mathcal{L}_{\text{hints}}(\mathbf{X}; \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\})$, and for the Pentomino task $\mathcal{L}_{\text{pentomino}}(\mathbf{X}; \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3\})$ together as a joint loss $\mathcal{L}_{\text{joint}}(\mathbf{X}; \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3\})$ as in Eqn 6:

$$\mathcal{L}_{\text{joint}}(\mathbf{X}; \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3\}) = \lambda \mathcal{L}_{\text{hints}}(\mathbf{X}; \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}) + \ (1 - \lambda)\mathcal{L}_{\text{pentomino}}(\mathbf{X}; \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_3\}). \tag{6}$$

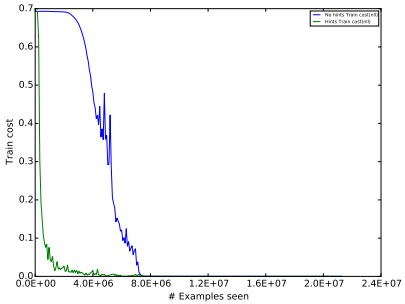


Figure 11: Plot showing the training learning curve of SMLP from random initialization when training with hints and without hints. SMLP training with hints converges faster than the training without hints. This means that the optimizing the training objective with hints is easier than optimizing without hints.

As seen on Figure 11 and 12, hint based training converges faster and the training of SMLP without hints overfits. This aligns with the hypothesis that hints make the optimization easier. But in the mean time, hints cost, puts the model into a basin of attraction to where a solution on the test set can easily be found. For the experiments in these plots, we cross-validated momentum, learning rate, standard deviation of the initialization and minibatch size. We trained the models with rectifier activation function using 512 hidden units at each layer. We trained the first level MLP on patches. It outputs 11 features for each such patch, with a softmax$(\cdot)$ activation function over the 11 units. We ran 64 different trials for the hyperparameters of the SMLP-hints. The learning rate is randomly sampled in log-space from the range $[5e-5, 1e-2]$, momentum is sampled from uniform distribution in the range $[0.25, 0.99]$, batch sizes and standard deviations of the weights are sampled from the grids $\{100, 200, 300, 400\}$ and $\{0.01, 0.025, 0.05\}$. We ran 150 different trials to explore hyperparameters for the SMLP-nohints. The learning rate is randomly sampled in log-space from the range $[6e-5, 8e-2]$, momentum is chosen randomly from the range $[0.25, 0.99]$, batch sizes and standard deviations of the weights are sampled from the grids $\{100, 200, 300, 400\}$ and $\{0.01, 0.025, 0.038, 0.05\}$ respectively for the without hints training. At the end of the hyper-parameter search, we chose the

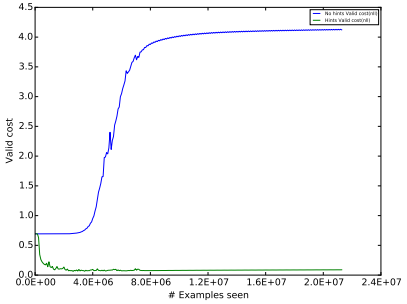


Figure 12: Plot showing validation learning curve of SMLP from random initialization when training with and without hints. Training of SMLP without hints overfits.

best hyper-parameters based on the training cost after 20000 updates. The codes to reproduce these experiments are available at `https://github.com/caglar/PentominoExps`.

We used a soft-curriculum strategy, for training our models. We started the training by only optimizing the easy hint objective function, namely setting λ to 1 in the beginning of the training and linearly annealing it towards to 0.1 in 10000 updates.

4. Conclusion and Discussion

In this paper we have shown an example of a toy task which seems to be quiet difficult to solve by standard black-box machine learning algorithms, but it can be solved almost perfectly when one encourages a particular semantics on the intermediate-level representation which is guided by the prior knowledge about the task. Both tasks have the particularity that they are defined by the composition of two non-linear sub-tasks (for example, object detection on one hand, and a non-linear logical operation similar to XOR on the other hand).

What is interesting is that for neural networks, we can compare two networks with exactly the same architecture but a different pre-training, one of which uses the known intermediate concepts to teach an intermediate representation to the network. With enough capacity and training time they can overfit. But they failed solve the task, as seen by test set performances.

We know that a structured deep network can learn the task, if it is initialized in the right place, and it can do that from a relatively small number of examples. Furthermore we have shown that if one pre-trains the SMLP with hints for only one epoch, eventually it is able to solve the task. But the same architecture trained from random initialization failed to generalize as well as the pretrained network.

Considering the fact that even SMLP-nohints with standardization after being trained using online SGD over 1046000 generated examples and still gets 27.5% test error. This suggests that in addition to the optimization effect of hints, there is also a regularization effect, since a much larger training set allowed to reduce test error, albeit not as much as using hints and a much smaller training set.

We also investigated the effect of introducing hints on the convergence of the optimization of SMLP in Figure 11. After hyperparameter optimization, we observed that SMLP trained with hints converges on training set faster and achieves lower validation error as well. This supports our claims that the optimizing with hints cost is easier.

What we hypothesize is that for most initializations and architectures (in particular the fully-connected ones), although it is possible to find a *good effective local minimum of training error* when enough capacity is provided, it is difficult (without the proper initialization) to find a good local minimum of generalization error. On the other hand, when the network architecture is constrained enough, even though a good solution exists (e.g. with the SMLP architecture), it seems that the optimization problem can still be difficult and even training error remains stuck high without the standardization layer. Standardization makes the training objective of the SMLP easier to optimize and helps it to find at least a *better effective local minimum of training error*. This finding suggests that by using specific architectural constraints and sometimes domain specific knowledge about the problem, one can alleviate the optimization difficulty that generic neural network architectures face.

It could be that the combination of the network architecture and training procedure produces a training dynamics that tends to yield into these minima that are poor from the point of view of generalization error, even when they manage to slowly nail training error by providing enough capacity. Of course, as the number of examples increases, we would expect this discrepancy to decrease, but then the optimization problem could still make the task unfeasible in practice. Note however that our preliminary experiments with increasing the training set size (8-fold) for MLPs did not reveal signs of potential improvements in test error yet, as shown in Figure 9. Even using online training on 545400 Pentomino examples, the SMLP-nohints architecture was still doing far from perfect in terms of generalization error (Figure 8(a)).

These findings bring supporting evidence to the "Guided Learning Hypothesis" and "Deeper Harder Hypothesis" from Bengio (2013a): higher level abstractions, which are expressed by composing simpler concepts are more difficult to learn (with the learner often getting in an effective local minimum), but that difficulty can be overcome if another agent provides hints about intermediate-level abstractions which are relevant to the task.

Many interesting questions remain open. Would a network without any guiding hint eventually find the 0% error solution provided enough training time and/or with alternate parameterizations? To what extent is ill-conditioning a core issue? The results with LBFGS were disappointing but changes in the architectures (such as standardization of the intermediate level) seem to make training much easier. Clearly, one can reach good solutions from an appropriate initialization, pointing in the direction of an issue with local minima, but it may be that good solutions are also reachable from other initializations, albeit going through a ill-conditioned path in parameter space. Why did our attempts at learning the intermediate concepts in an unsupervised way fail? Are these results specific to the task we are testing or a limitation of the unsupervised feature learning algorithm tested? Trying with many more unsupervised variants and exploring explanatory hypotheses for the observed failures could help us answer that. Finally, and most ambitious, can we solve these kinds of problems if we allow a community of learners to collaborate and collectively discover and combine partial solutions in order to obtain solutions to more abstract tasks like the one presented here? Indeed, we would like to discover learning algorithms that can solve such tasks without the use of prior knowledge as specific and strong as the one used in the SMLP here. These experiments could be

inspired by and inform us about potential mechanisms for collective learning through cultural evolutions in human societies.

Acknowledgments

We would like to thank to the ICLR 2013 reviewers for their insightful comments, and NSERC, CIFAR, Compute Canada and Canada Research Chairs for funding.

Appendix A. Standardization Layer

Normalization has been used occasionally in convolutional neural network to encourage the competition between the hidden units. (Jarrett et al., 2009) used a local contrast normalization layer in their architecture which performs subtractive and divisive normalization. A local contrast normalization layer enforces a local competition between adjacent features in the feature map and between features at the same spatial location in different feature maps. Similarly, (Krizhevsky et al., 2012) observed that using a local response layer that enjoys the benefit of using local normalization scheme aids generalization.

Standardization has been observed to be crucial for SMLP trained either with or without hints. In both SMLP-hints and SMLP-nohints experiments, the neural network was not able to generalize or even learn the training set without using standardization in the SMLP intermediate layer, doing just chance performance. More specifically, in the SMLP-nohints architecture, standardization is part of the computational graph, hence the gradients are being backpropagated through it. The mean and the standard deviation is computed for each hidden unit separately at the intermediate layer as in Equation 9. But in order to prevent numerical underflows during the backpropagation, we used $\epsilon = 1e - 8$ (Equation 8).

The benefit of having sparse activations may be specifically important for the ill-conditioned problems, for the following reasons. When a hidden unit is "off", its gradient (the derivative of the loss with respect to its activation before the non-linearity is applied) is usually close to 0 as well. That means that all off-diagonal second derivatives involving that hidden unit (for example, its input weights) are also near 0. This is basically like removing some columns and rows from the Hessian matrix associated with a particular example. It has been observed that the condition number of the Hessian matrix (specifically, its largest eigenvalue) increases as the size of the network increases (Dauphin and Bengio, 2013), making training considerably slower and inefficient. Hence one would expect that as sparsity of the gradients (obtained because of sparsity of the activations) increases, training would become more efficient, as if we were training a smaller sub-network for each example, with shared weights across examples, as in dropouts (Hinton et al., 2012).

Standardization layer is on top of the output of P1NN which centers the activations and performs divisive normalization by dividing by the standard deviation over a mini-batch of the activations of that layer. We denote the standardization function $z(\cdot)$. Standardization makes use of the mean and standard deviation computed for each hidden unit such that each hidden unit of $\mathbf{h_o}$ will have 0 activation and unit standard deviation on average over the mini-batch. X is the set of pentomino images in the mini-batch, where $X \in R^{d_{in} \times N}$ is a matrix with N images. $h_o^{(i)}(\mathbf{x_j})$ is the vector of activations of the i-th hidden unit of hidden layer $h_o(\mathbf{x_j})$ for the j-th example, with $x_j \in X$.

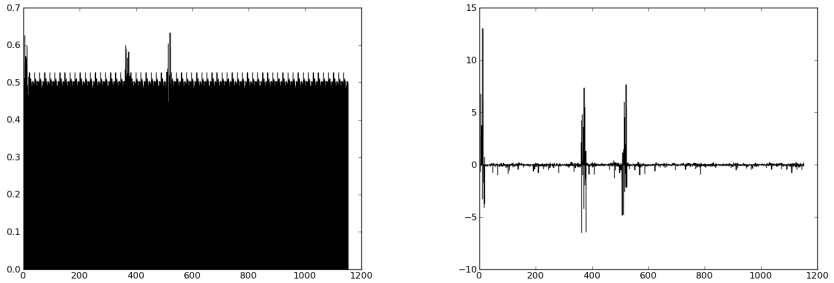
$$\mu_{h_o^{(i)}} = \frac{1}{N} \sum_{\mathbf{x_j} \in X} h_o^{(i)}(\mathbf{x_j}), \tag{7}$$

$$\sigma_{h_o^{(i)}} = \sqrt{\frac{\sum_j^N (h_o^{(i)}(\mathbf{x_j}) - \mu_{h_o^{(i)}})^2}{N} + \epsilon}, \tag{8}$$

$$z(h_o^{(i)}(\mathbf{x_j})) = \frac{h_o^{(i)}(\mathbf{x_j}) - \mu_{h_o^{(i)}}}{\max(\sigma_{h_o^{(i)}}, \epsilon)}. \tag{9}$$

where ϵ is a very small constant, that is used to prevent numerical underflows in the standard deviation. P1NN is trained on each 8×8 patches extracted from the image.

In Figure 13, the activation of each hidden unit in a bar chart is shown: the effect of standardization is significant, making the activations sparser.



(a) Before standardization.

(b) After standardization.

Figure 13: Activations of the intermediate-level hidden units of an SLMP-nohints for a particular examples (x-axis: hidden unit number, y-axis: activation value). Left (a): before standardization. Right (b): after standardization.

In Figure 15, one can see the activation histogram of the SMLP-nohints intermediate layer, showing the distribution of activation values, before and after standardization. Again the sparsifying effect of standardization is very apparent.

In Figures 15 and 13, the intermediate level activations of SMLP-nohints are shown before and after standardization. These are for the same SMLP-nohints architecture whose results are presented on Table 4. For that same SMLP, the Adadelta (Zeiler, 2012) adaptive learning rate scheme has been used, with 512 hidden units for the hidden layer of P1NN and rectifier activation function. For the output of the P1NN, 11 sigmoid units have been used while P2NN had 1200 hidden units with a rectifier activation function. The output nonlinearity of the P2NN is a sigmoid and the training objective is the binary cross-entropy.

Appendix B. Binary-Binary RBMs on Pentomino Dataset

We trained binary-binary RBMs (both visible and hidden are binary) on 8×8 patches extracted from the Pentomino Dataset using PCD (stochastic maximum likelihood), a weight decay of .0001 and a sparsity penalty[2]. We used 256 hidden units and trained by SGD with a batch size of 32 and an annealing learning rate (Bengio, 2013b) starting from 1e-3 with annealing rate 1.000015. The RBM is trained with momentum starting from 0.5 to 0.9. The biases are initialized to -2 in order to get a sparse representation. The RBM is trained for 120 epochs (approximately 50 million updates).

After pretraining the RBM, its parameters are used to initialize the first layer of an SMLP-nohints network. As in the usual architecture of the SMLP-nohints on top of P1NN, there is an intermediate layer. Both P1NN and the intermediate layer have a sigmoid nonlinearity, and the intermediate layer has 11 units per location. This SMLP-nohints is trained with Adadelta and standardization at the intermediate layer [3].

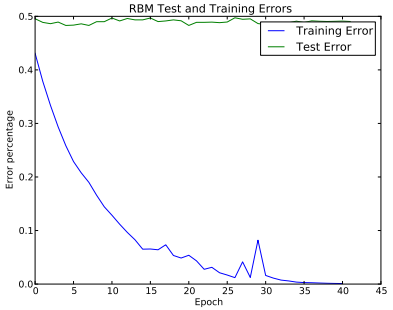


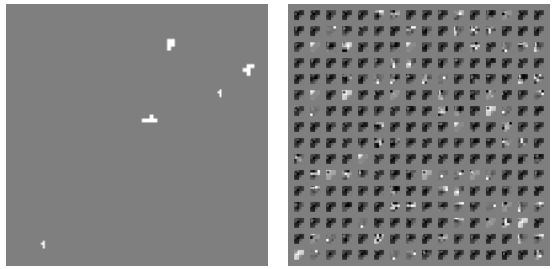
Figure 14: Training and test errors of an SMLP-nohints network whose first layer is pre-trained as an RBM. Training error reduces to 0% at epoch 42, but test error is still chance.

Experimental Setup and Hyper-parameters

B.1 Decision Trees

We used the decision tree implementation in the scikit-learn (Fabian Pedregosa, 2011) python package which is an implementation of the CART (Regression Trees) algorithm. The CART algorithm constructs the decision tree recursively and partitions the input space such that the samples belonging to the same category are grouped together (Breiman et al., 1984). We used The Gini index as the impurity criteria. We evaluated the hyper-parameter configurations with a grid-search. We cross-validated the maximum depth (max_depth) of the tree (for preventing the algorithm to severely overfit the training set) and minimum number of samples required to create a split (min_split). 20 different configurations of hyper-parameter values were evaluated. We obtained the best validation error with $max\_depth = 300$ and $min\_split = 8$.

2. implemented as TorontoSparsity in pylearn2, see the yaml file in the repository for more details
3. In our auto-encoder experiments we directly fed features to P2NN without standardization and Adadelta.



(a) Samples of RBM (b) Filters learned by RBM

Figure 15: Inspecting the representation learned by RBM. Left (a): 100 samples generated from trained RBM. All the generated samples are valid Pentomino shapes. Right (b): 100 Binary-Binary RBM Pentomino negative samples.

B.2 Support Vector Machines

We used the "Support Vector Classifier (SVC)" implementation from the scikit-learn package which in turn uses the libsvm's Support Vector Machine (SVM) implementation. Kernel-based SVMs are non-parametric models that map the data into a high dimensional space and separate different classes with hyperplane(s) such that the support vectors for each category will be separated by a large margin. We cross-validated three hyper-parameters of the model using grid-search: C, γ and the type of kernel($kernel\_type$). C is the penalty term (weight decay) for the SVM and γ is a hyper-parameter that controls the width of the Gaussian for the RBF kernel. For the polynomial kernel, γ controls the flexibility of the classifier (degree of the polynomial) as the number of parameters increases (Hsu et al., 2003; Ben-Hur and Weston, 2010). We evaluated forty-two hyper-parameter configurations. That includes, two kernel types: $\{RBF,\ Polynomial\}$; three gammas: $\{1e-2,\ 1e-3,\ 1e-4\}$ for the RBF kernel, $\{1, 2, 5\}$ for the polynomial kernel, and seven C values among: $\{0.1, 1, 2, 4, 8, 10, 16\}$. As a result of the grid search and cross-validation, we have obtained the best test error by using the RBF kernel, with $C = 2$ and $\gamma = 1$.

B.3 Multi Layer Perceptron

We have our own implementation of Multi Layer Perceptron based on the Theano (Bergstra et al., 2010) machine learning libraries. We have selected 2 hidden layers, the rectifier activation function, and 2048 hidden units per layer. We cross-validated three hyper-parameters of the model using random-search, sampling the learning rates ϵ in log-domain, and selecting $L1$ and $L2$ regularization penalty coefficients in sets of fixed values, evaluating 64 hyperparameter values. The range of the hyperparameter values are $\epsilon \in [0.0001, 1]$, $L1 \in \{0., 1e-6, 1e-5, 1e-4\}$ and $L2 \in \{0, 1e-6, 1e-5\}$. As a result, the following were selected: $L1 = 1e-6$, $L2 = 1e-5$ and $\epsilon = 0.05$.

B.4 Random Forests

We used scikit-learn's implementation of "Random Forests" decision tree learning. The Random Forests algorithm creates an ensemble of decision trees by randomly selecting for each tree a subset of features and applying bagging to combine the individual decision trees (Breiman, 2001). We have used grid-search and cross-validated the max_depth, min_split, and number of trees ($n\_estimators$). We have done the grid-search on the following hyperparameter values, $n\_estimators \in \{5, 10, 15, 25, 50\}$, $max\_depth \in \{100, 300, 600, 900\}$, and $min\_splits \in \{1, 4, 16\}$. We obtained the best validation error with $max\_depth = 300$, $min\_split = 4$ and $n\_estimators = 10$.

B.5 k-Nearest Neighbors

We used scikit-learn's implementation of k-Nearest Neighbors (k-NN). k-NN is an instance-based, lazy learning algorithm that selects the training examples closest in Euclidean distance to the input query. It assigns a class label to the test example based on the categories of the k closest neighbors. The hyper-parameters we have evaluated in the cross-validation are the number of neighbors (k) and $weights$. The $weights$ hyper-parameter can be either "uniform" or "distance". With "uniform", the value assigned to the query point is computed by the majority vote of the nearest neighbors. With "distance", each value assigned to the query point is computed by weighted majority votes where the weights are computed with the inverse distance between the query point and the neighbors. We have used $n\_neighbours \in \{1, 2, 4, 6, 8, 12\}$ and $weights \in \{"uniform", "distance"\}$ for hyper-parameter search. As a result of cross-validation and grid search, we obtained the best validation error with $k = 2$ and $weights$="uniform".

B.6 Convolutional Neural Nets

We used a Theano (Bergstra et al., 2010) implementation of Convolutional Neural Networks (CNN) from the deep learning tutorial at `deeplearning.net`, which is based on a vanilla version of a CNN LeCun et al. (1998). Our CNN has two convolutional layers. Following each convolutional layer, we have a max-pooling layer. On top of the convolution-pooling-convolution-pooling layers there is an MLP with one hidden layer. In the cross-validation we have sampled 36 learning rates in log-domain in the range $[0.0001, 1]$ and the number of filters from the range $[10, 20, 30, 40, 50, 60]$ uniformly. For the first convolutional layer we used 9×9 receptive fields in order to guarantee that each object fits inside the receptive field. As a result of random hyperparameter search and doing manual hyperparameter search on the validation dataset, the following values were selected:

- The number of features used for the first layer is 30 and the second layer is 60.
- For the second convolutional layer, 7×7 receptive fields. The stride for both convolutional layers is 1.
- Convolved images are downsampled by a factor of 2×2 at each pooling operation.
- The learning rate for CNN is 0.01 and it was trained for 8 epochs.

B.7 Maxout Convolutional Neural Nets

We used the pylearn2 (`https://github.com/lisa-lab/pylearn2`) implementation of maxout convolutional networks (Goodfellow et al., 2013). There are two convolutional layers in the selected architecture, without any pooling. In the last convolutional layer, there is a maxout non-linearity. The following were selected by cross-validation: learning rate, number of channels for the both convolution layers, number of kernels for the second layer and number of units and pieces per maxout unit in the last layer, a linearly decaying learning rate, momentum starting from 0.5 and saturating to 0.8 at the 200'th epoch. Random search for the hyperparameters was used to evaluate 48 different hyperparameter configurations on the validation dataset. For the first convolutional layer, 8×8 kernels were selected to make sure that each Pentomino shape fits into the kernel. Early stopping was used and test error on the model that has the best validation error is reported. Using norm constraint on the fan-in of the final softmax units yields slightly better result on the validation dataset.

As a result of cross-validation and manually tuning the hyperparameters we used the following hyperparameters:

- 16 channels per convolutional layer. 600 hidden units for the maxout layer.
- 6x6 kernels for the second convolutional layer.
- 5 pieces for the convolution layers and 4 pieces for the maxout layer per maxout units.
- We decayed the learning rate by the factor of 0.001 and the initial learning rate is 0.026367. But we scaled the learning rate of the second convolutional layer by a constant factor of 0.6.
- The norm constraint (on the incoming weights of each unit) is 1.9365.

Figure 16 shows the first layer filters of the maxout convolutional net, after being trained on the 80k training set for 85 epochs.

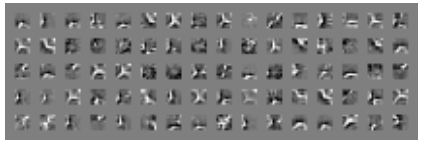


Figure 16: Maxout convolutional net first layer filters. Most of the filters were able to learn the basic edge structure of the Pentomino shapes.

B.8 Stacked Denoising Auto-Encoders

Denoising Auto-Encoders (DAE) are a form of regularized auto-encoder (Bengio et al., 2013). The DAE forces the hidden layer to discover more robust features and prevents it from simply learning the identity by reconstructing the input from a corrupted version of it (Vincent et al., 2010). Two DAEs were stacked, resulting in an unsupervised transformation with two hidden layers of 1024 units each. Parameters of all layers are then fine-tuned with supervised fine-tuning using logistic regression as the classifier and SGD as the gradient-based optimization

algorithm. The stochastic corruption process is binomial (0 or 1 replacing each input value, with probability 0.2). The selected learning rate is $\epsilon_0 = 0.01$ for the DAe and $\epsilon_1 = 0.1$ for supervised fine-tuning. Both L1 and L2 penalty for the DAEs and for the logistic regression layer are set to 1e-6.

B.8.1 CAE+MLP with Supervised Finetuning:

A regularized auto-encoder which sometimes outperforms the DAE is the Contractive Auto-Encoder (CAE), (Rifai et al., 2012), which penalizes the Frobenius norm of the Jacobian matrix of derivatives of the hidden units with respect to the CAE inputs. The CAE serves as pre-training for an MLP, and in the supervised fine-tuning state, the Adagrad method was used to automatically tune the learning rate (Duchi et al., 2011).

After training a CAE with 100 sigmoidal units patch-wise, the features extracted on each patch are concatenated and fed as input to an MLP. The selected Jacobian penalty coefficient is 2, the learning rate for pre-training is 0.082 with batch size of 200 and 200 epochs of unsupervised learning are performed on the training set. For supervised finetuning, the learning rate is 0.12 over 100 epochs, L1 and L2 regularization penalty terms respectively are 1e-4 and 1e-6, and the top-level MLP has 6400 hidden units.

B.8.2 Greedy Layerwise CAE+DAE Supervised Finetuning:

For this experiment we stack a CAE with sigmoid non-linearities and then a DAE with rectifier non-linearities during the pre-training phase. As recommended by Glorot et al. (2011) we have used a softplus nonlinearity for reconstruction, $softplus(x) = log(1 + e^x)$. We used an L1 penalty on the rectifier outputs to obtain a sparser representation with rectifier non-linearity and L2 regularization to keep the non-zero weights small.

The main difference between the DAE and CAE is that the DAE yields more robust reconstruction whereas the CAE obtains more robust features (Rifai et al., 2011).

As seen on Figure 6 the weights U and V are shared on each patch and we concatenate the outputs of the last auto-encoder on each patch to feed it as an input to an MLP with a large hidden layer.

We used 400 hidden units for the CAE and 100 hidden units for DAE. The learning rate used for the CAE is 0.82 and for DAE it is 9*1e-3. The corruption level for the DAE (binomial noise) is 0.25 and the contraction level for the CAE is 2.0. The L1 regularization penalty for the DAE is 2.25*1e-4 and the L2 penalty is 9.5*1e-5. For the supervised finetuning phase the learning rate used is 4*1e-4 with L1 and L2 penalties respectively 1e-5 and 1e-6. The top-level MLP has 6400 hidden units. The auto-encoders are each trained for 150 epochs while the whole MLP is fine-tuned for 50 epochs.

B.8.3 Greedy Layerwise DAE+DAE Supervised Finetuning:

For this architecture, we have trained two layers of denoising auto-encoders greedily and performed supervised finetuning after unsupervised pre-training. The motivation for using two denoising auto-encoders is the fact that rectifier nonlinearities work well with the deep networks but it is difficult to train CAEs with the rectifier non-linearity. We have used the same type of denoising auto-encoder that is used for the greedy layerwise CAE+DAE supervised finetuning experiment.

In this experiment we have used 400 hidden units for the first layer DAE and 100 hidden units for the second layer DAE. The other hyperparameters for DAE and supervised finetuning are the same as with the *CAE+DAE MLP Supervised Finetuning* experiment.

References

Journal of Machine Learning Research, -1.

Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(4):341–379, 2003.

Asa Ben-Hur and Jason Weston. A user's guide to support vector machines. *Methods in Molecular Biology*, 609:223–239, 2010.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. Also published as a book. Now Publishers, 2009.

Yoshua Bengio. Evolving culture vs local minima. In *Growing Adaptive Machines: Integrating Development and Learning in Artificial Neural Networks*, number also as ArXiv 1203.2990v1, pages T. Kowaliw, N. Bredeche & R. Doursat, eds. Springer-Verlag, March 2013a. URL `http://arxiv.org/abs/1203.2990`.

Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In K.-R. Müller, G. Montavon, and G. B. Orr, editors, *Neural Networks: Tricks of the Trade*. Springer, 2013b.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In Bernhard Schölkopf, John Platt, and Thomas Hoffman, editors, *Advances in Neural Information Processing Systems 19 (NIPS'06)*, pages 153–160. MIT Press, 2007.

Yoshua Bengio, Jerome Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In Léon Bottou and Michael Littman, editors, *Proceedings of the Twenty-sixth International Conference on Machine Learning (ICML'09)*. ACM, 2009.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. Technical Report arXiv:1206.5538, U. Montreal, 2012. URL `http://arxiv.org/abs/1206.5538`.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 2013.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. Classification and regression trees. *Belmont, Calif.: Wadsworth*, 1984.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surface of multilayer networks. *AISTATS 2015, Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 192–204, 2015.

Dan C. Ciresan, Ueli Meier, Luca M. Gambardella, and Jürgen Schmidhuber. Deep big simple neural nets for handwritten digit recognition. *Neural Computation*, 22:1–14, 2010.

Yann Dauphin and Yoshua Bengio. Big neural networks waste capacity. Technical Report arXiv:1301.3583, Universite de Montreal, 2013.

Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *NIPS'2014*, 2014.

Richard Dawkins. *The Selfish Gene*. Oxford University Press, 1976.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? In *Journal of Machine Learning Research* JML (-1), pages 625–660.

Alexandre Gramfort Vincent Michel Bertrand Thirion Olivier Grisel Mathieu Blondel et al Fabian Pedregosa, Gal Varoquaux. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

Franois Fleuret, Ting Li, Charles Dubout, Emma K. Wampler, Steven Yantis, and Donald Geman. Comparing machines and humans on a visual categorization test. *Proceedings of the National Academy of Sciences*, 108(43):17621–17625, 2011.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9, pages 249–256, May 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *JMLR W&CP: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011)*, April 2011.

Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *ICML'2013*, 2013.

Caglar Gulcehre, Kyunghyun Cho, Razvan Pascanu, and Yoshua Bengio. Learned-norm pooling for deep neural networks. *arXiv preprint arXiv:1311.1780*, 2013.

Joseph Henrich and Richard McElreath. The evolution of cultural evolution. *Evolutionary Anthropology: Issues, News, and Reviews*, 12(3):123–135, 2003.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. Technical report, arXiv:1207.0580, 2012.

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification, 2003.

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *Proc. International Conference on Computer Vision (ICCV'09)*, pages 2146–2153. IEEE, 2009.

Faisal Khan, Xiaojin Zhu, and Bilge Mutlu. How do humans teach: On curriculum learning and teaching dimension. In *Advances in Neural Information Processing Systems 24 (NIPS'11)*, pages 1449–1457, 2011.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25 (NIPS'2012)*. 2012.

Kai A. Krueger and Peter Dayan. Flexible shaping: how learning in small steps helps. *Cognition*, 110:380–394, 2009.

Gautam Kunapuli, Kristin P. Bennett, Richard Maclin, and Jude W. Shavlik. The adviceptron: Giving advice to the perceptron. *Proceedings of the Conference on Artificial Neural Networks In Engineering (ANNIE 2010)*, 2010.

Hugo Larochelle, Yoshua Bengio, Jerome Louradour, and Pascal Lamblin. Exploring strategies for training deep neural networks. *Journal of Machine Learning Research*, 10:1–40, 2009.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Tom M. Mitchell. *The Need for Biases in Learning Generalizations*. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ., 1980.

Tom M. Mitchell and Sebastian B. Thrun. Explanation-based neural network learning for robot control. *Advances in Neural information processing systems*, pages 287–287, 1993.

Richard Montague. Universal grammar. *Theoria*, 36(3):373–398, 1970.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

Joseph O'Sullivan. Integrating initialization bias and search bias in neural network learning, 1996.

Gail B. Peterson. A day of great illumination: B. F. Skinner's discovery of shaping. *Journal of the Experimental Analysis of Behavior*, 82(3):317–328, 2004.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of theTwenty-eight International Conference on Machine Learning (ICML'11)*, June 2011.

Salah Rifai, Yoshua Bengio, Yann Dauphin, and Pascal Vincent. A generative process for sampling contractive auto-encoders. In *Proceedings of the Twenty-nine International Conference on Machine Learning (ICML'12)*. ACM, 2012. URL `http://icml.cc/discuss/2012/590.html`.

Ruslan Salakhutdinov and Geoffrey E Hinton. Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pages 448–455, 2009.

Burrhus F. Skinner. Reinforcement today. *American Psychologist*, 13:94–99, 1958.

Ray J. Solomonoff. A system for incremental learning based on algorithmic probability. In *Proceedings of the Sixth Israeli Conference on Artificial Intelligence, Computer Vision and Pattern Recognition*, pages 515–527. Citeseer, 1989.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4, 2012.

Geoffrey G. Towell and Jude W. Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1):119–165, 1994.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. In *Journal of Machine Learning Research* JML (-1), pages 3371–3408.

Luis Von Ahn, Manuel Blum, Nicholas J Hopper, and John Langford. Captcha: Using hard ai problems for security. In *Advances in CryptologyEUROCRYPT 2003*, pages 294–311. Springer, 2003.

Jason Weston, Frédéric Ratle, and Ronan Collobert. Deep learning via semi-supervised embedding. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 1168–1175, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390303.

Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 11/14; Revised 5/15; Published 3/16

Harry: A Tool for Measuring String Similarity

Konrad Rieck KONRAD.RIECK@CS.UNI-GOETTINGEN.DE
Christian Wressnegger CHRISTIAN.WRESSNEGGER@CS.UNI-GOETTINGEN.DE
University of Göttingen
Goldschmidtstraße 7
37077 Göttingen, Germany

Editor: Antti Honkela

Abstract

Comparing strings and assessing their similarity is a basic operation in many application domains of machine learning, such as in information retrieval, natural language processing and bioinformatics. The practitioner can choose from a large variety of available similarity measures for this task, each emphasizing different aspects of the string data. In this article, we present Harry, a small tool specifically designed for measuring the similarity of strings. Harry implements over 20 similarity measures, including common string distances and string kernels, such as the Levenshtein distance and the Subsequence kernel. The tool has been designed with efficiency in mind and allows for multi-threaded as well as distributed computing, enabling the analysis of large data sets of strings. Harry supports common data formats and thus can interface with analysis environments, such as Matlab, Pylab and Weka.

Keywords: string kernels, string distances, similarity measures for strings

1. Introduction

The comparison of strings is a basic operation in many applications of machine learning. Several problems of information retrieval and natural language processing center on comparing strings (see Salton and McGill, 1986). Similarly, several tasks in bioinformatics revolve around assessing the similarity of sequences (see Borgwardt, 2011). The problem underlying these tasks—*measuring the similarity or dissimilarity of two strings*—has been a vivid topic of research for over five decades, ranging from early telecommunication to modern machine learning and data analysis. As a result, the practitioner can choose from a large variety of available methods for assessing the similarity of strings[1], each emphasizing different aspects and characteristics of the data.

In this article we present Harry, a small tool specifically designed for measuring the similarity of strings and making various comparison methods available for analysis of string data. Harry implements over 20 common similarity measures, including classic string distances, such as the Damerau (1964) and Levenshtein (1966) distance, as well as modern string kernels, such as the Spectrum and Subsequence kernel (Lodhi et al., 2002). As the pairwise comparison of strings usually induces a quadratic run-time, Harry has been designed with efficiency in mind and allows for multi-threaded as well as distributed computing, which enables calculating large distance and kernel matrices. Harry supports common data formats and thus can interface with analysis tools and environments, such as Matlab, Pylab, Weka and LibSVM.

1. For ease of presentation, we use the term *similarity* synonymously with *dissimilarity* in this article.

While there also exist other tools implementing similarity measures for strings, such as the popular module python-Levenshtein, the generic toolbox Shogun (Sonnenburg et al., 2010) and the R package KeBABS for biological sequences (Palme et al., 2015), none of these tools provides a similarly broad range of similarity measures in comparison with Harry. Moreover, Harry complements the tool Sally (Rieck et al., 2012) which maps strings to vectors and allows for applying vectorial comparison functions to strings, such as the Euclidean and Manhattan distance. In combination, Harry & Sally provide a versatile basis for analyzing string data.

2. A Brief Overview of Harry

Harry implements a generic framework for the comparison of strings and assessing their similarity. In the following, we briefly discuss the main features of this framework.

2.1 Supported Similarity Measures

The current version of Harry supports the similarity measures listed in Table 1. Included are classic string distances by Damerau (1964) and Levenshtein (1966) as well as more recent methods, such as distances by Jaro (1989) and Bennett et al. (1998). Furthermore, the tool implements string kernels as described by Shawe-Taylor and Cristianini (2004) and allows for mapping kernels to distances and vice versa. In addition to distances and kernels, Harry also implements so-called similarity coefficients, such as the Jaccard index, which assess similarity in terms of matching sets of characters or words (Sokal and Sneath, 1963).

String Distances (10)			
Bag distance	Hamming distance	Kernel-substitution distance	String alignment distance
Compression distance	Jaro distance	Lee distance	
Damerau-Levenshtein distance	Jaro-Winkler distance	Levenshtein distance	
String Kernels (4)			
Distance-substitution kernel	Spectrum kernel	Subsequence kernel	Weighted-degree kernel
Similarity Coefficients (7)			
Braun-Blanquet coefficient	Kulczynski coefficient	Simpson coefficient	Sokal-Sneath coefficient
Jaccard coefficient	Otsuka coefficient	Soerensen-Dice coefficient	

Table 1: Similarity measures for strings supported by Harry (version 0.4.1).

2.2 Scalable Computation

For the efficient processing of large sets of strings, Harry makes use of multi-threading and distributes the workload over multiple CPU cores (see option `-n`). The run-time performance of Harry thus scales linearly with the number of cores and enables computing large distance and kernel matrices. Furthermore, Harry can split the calculation of large matrices into smaller chunks, where these chunks are either defined manually as sub-matrices (see options `-x` and `-y`) or automatically from the number of requested splits (option `-s`). Using this splitting the computation of similarity measures can be easily distributed over multiple hosts and carried out with systems for high performance computing, such as the LSF or SGE platform.

The run-time performance of Harry is further improved by caching recurrent computations (option `-a`), such as during the normalization of string kernels and distances. The underlying cache builds on a lightweight synchronization that ensures little overhead, even if several threads concurrently access the data. The caching can also be applied directly to the computed similarity values (option `-G`) and speed up the comparison of data sets with duplicate strings.

2.3 Interfaces and Preprocessing

To interface with other analysis tools and environments, Harry implements support for common input and output formats. The tool can read string data from text files, directories and compressed archives (option `-i`). Moreover, it is able to store computed similarity matrices in output formats suitable for Matlab, Pylab, Weka and LibSVM (option `-o`). Additionally, Harry is bundled with a Python module that enables efficiently comparing strings in Python without storing data on disk.

Harry supports several preprocessing functions that improve string comparison in particular fields of application. For example, the tool enables the user to change the granularity of the comparison to either bytes, bits or tokens (option `-g`). In the latter setting the input strings are partitioned into tokens using a set of delimiter characters, thereby enabling the analysis of structured data, such as text and log entries. Moreover, Harry supports removing stop tokens from strings, transforming tokens to Soundex codes and encoding non-printable characters in texts.

The data format, preprocessing functions and the selected similarity measure can be specified on the command line as well as in a configuration file (option `-c`). As a result, experiments with Harry can be easily reproduced using stored configurations.

3. Experiments with Harry

We demonstrate the efficiency of Harry in an empirical evaluation, where we first study its scalability (Section 3.1) and then compare its performance to related tools (Section 3.2). In all experiments we consider the data sets listed in Table 2 which contain strings of DNA snippets, protein sequences, Twitter messages and network traces, respectively.

Data set	**ARTS**	**SPROT**	**TWEETS**	**WEBFP**
Type of strings	DNA snippets	Protein sequences	Twitter messages	Network traces
Average length of strings	2,400 bases	457 proteins	126 characters	1,312 packets
Size of alphabet	4 bases	22 proteins	69 characters	6 packet sizes

Table 2: Data sets for evaluation. Each data set consists of 1,000 strings randomly drawn from the original source: ARTS (Sonnenburg et al., 2006); SPROT (O'Donovan et al., 2002); TWEETS (Twitter.com); WEBFP (Cai et al., 2012).

3.1 Scalability of Harry

In our first experiment, we compute the Levenshtein (1966) distance, the normalized compression distance (Bennett et al., 1998) and the Subsequence kernel (Lodhi et al., 2002) on all four data sets using Harry. We repeat the computation with a different number of available CPU cores and measure the run-time in terms of comparisons per second.

Figure 1 shows the results of this experiment. The Levenshtein distance and the Subsequence kernel scale perfectly linear with the number of CPU cores, reaching peak performances of 10^5 and 10^4 comparisons per second, respectively. The compression distance scales almost linearly, as the caching used for normalization induces a slight overhead when more than 8 cores are used.

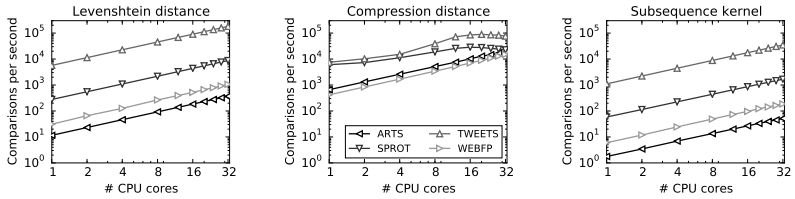


Figure 1: Run-time performance of Harry with varying number of CPU cores.

3.2 Comparative Evaluation

In the second experiment, we compare Harry with other tools for measuring string similarity. We consider the Python modules *python-Levenshtein* (0.11.2) and *python-jellyfish* (0.5.0) that implement the Levenshtein distance and its variants, the library *CompLearn* (1.1.7) that focuses on compression distances, and the machine learning toolbox *Shogun* (4.0.0) that provides several string kernels. For each of the four data sets, we randomly draw 100 strings, compute a full similarity matrix with each tool and measure the run-time over 10 runs.

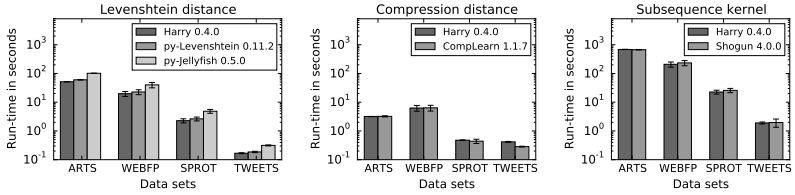


Figure 2: Comparative evaluation of Harry and related software tools.

The averaged results for the comparative evaluation are shown in Figure 2. In all settings, Harry is on par with the other tools and provides a similar and often even better performance. Given that each of the related tools focuses only on a subset of similarity measures, this experiment demonstrates the versatile yet efficient implementation of Harry.

4. Conclusions

Harry provides access to a wide range of similarity measures and enables their efficient computation on string data. In combination with Sally (Rieck et al., 2012), the tool is a perfect fit for analyzing and learning with strings. The source code of Harry along with documentation and a tutorial is available at *http://www.mlsec.org/harry*.

Acknowledgments

The authors gratefully acknowledge funding from BMBF (16KIS0154K) and DFG (RI 2469/1-1).

References

C. Bennett, P. Gacs, M. Li, P. Vitanyi, and W. Zurek. Information distance. *IEEE Transactions on Information Theory*, 44(4):1407–1423, July 1998.

K. M. Borgwardt. *Kernel Methods in Bioinformatics*, chapter Handbook of Statistical Bioinformatics, pages 317–334. Springer, 2011.

X. Cai, X. C. Zhang, B. Joshi, and R. Johnson. Touching from a distance: Website fingerprinting attacks and defenses. In *Proc. of ACM Conference on Computer and Communications Security (CCS)*, pages 605–616, 2012.

F. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

M. A. Jaro. Advances in record linkage methodology as applied to the 1985 census of tampa florida. *Journal of the American Statistical Association*, 84(406):414–420, 1989.

V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii Nauk SSSR*, 163(4):845–848, 1966.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

C. O'Donovan, M. Martin, A. Gattiker, E. Gasteiger, A. Bairoch, and R. Apweiler. High-quality protein knowledge resource: SWISS-PROT and TrEMBL. *Briefings in Bioinformatics*, 3(3): 275–284, 2002.

J. Palme, S. Hochreiter, and U. Bodenhofer. KeBABS: an R package for kernel-based analysis of biological sequences. *Bioinformatics*, 2015. (doi: 10.1093/bioinformatics/btv176).

K. Rieck, C. Wressnegger, and A. Bikadorov. Sally: A tool for embedding strings in vector spaces. *Journal of Machine Learning Research (JMLR)*, 13(Nov):3247–3251, Nov. 2012.

G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1986.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

R. Sokal and P. Sneath. *Principles of Numerical Taxonomy*. W.H. Freeman and Company, San Francisco, CA, USA, 1963.

S. Sonnenburg, A. Zien, and G. Rätsch. ARTS: Accurate recognition of transcription starts in human. *Bioinformatics*, 22(14):e472–e480, 2006.

S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, and V. Franc. The shogun machine learning toolbox. *Journal of Machine Learning Research (JMLR)*, 11(Jun):1799–1802, 2010.

Journal of Machine Learning Research 17 (2016) 1-29 Submitted 6/13; Revised 10/14; Published 3/16

Herded Gibbs Sampling

Yutian Chen yutian.chen@uci.edu
7 Pancras Square, Kings Cross, London, N1C 4AG, United Kingdom

Luke Bornn bornn@stat.harvard.edu
Statistics & Actuarial Science, Simon Fraser University, 8888 University Drive, Burnaby, BC, V5A1S6, Canada

Nando de Freitas nando@cs.ox.ac.uk
7 Pancras Square, Kings Cross, London, N1C 4AG, United Kingdom

Mareija Eskelin mareija@cs.ubc.ca
Dept of Computer Science, University of British Columbia, 2366 Main Mall, Vancouver, BC, V6T1Z4, Canada

Jing Fang jingf@cs.ubc.ca
1 Facebook Way, Menlo Park, CA, 94025, United States

Max Welling welling@ics.uci.edu
Informatics Institute, Science Park 904, Postbus 94323, 1090 GH, Amsterdam, Netherlands

Editor: Aaron Courville, Rob Fergus, and Christopher Manning

Abstract

The Gibbs sampler is one of the most popular algorithms for inference in statistical models. In this paper, we introduce a herding variant of this algorithm, called herded Gibbs, that is entirely deterministic. We prove that herded Gibbs has an $O(1/T)$ convergence rate for models with independent variables and for fully connected probabilistic graphical models. Herded Gibbs is shown to outperform Gibbs in the tasks of image denoising with MRFs and named entity recognition with CRFs. However, the convergence for herded Gibbs for sparsely connected probabilistic graphical models is still an open problem.

Keywords: Gibbs sampling, herding, deterministic sampling

1. Introduction

Over the last 60 years, we have witnessed great progress in the design of randomized sampling algorithms; see for example Liu (2001); Doucet et al. (2001); Andrieu et al. (2003); Robert and Casella (2004) and the references therein. In contrast, the design of deterministic algorithms for "sampling" from distributions is still in its inception (Chen et al., 2010; Holroyd and Propp, 2010; Chen et al., 2011; Murray and Elliott, 2012). There are, however, many important reasons for pursuing this line of attack on the problem. From a theoretical perspective, this is a well defined mathematical challenge whose solution might have important consequences. It also brings us closer to reconciling the fact that we typically use pseudo-random number generators to run Monte Carlo algorithms on classical, Von Neumann architecture, computers. Moreover, the theory for some of the recently proposed deterministic sampling algorithms has taught us that they can achieve $O(1/T)$ convergence rates (Chen et al., 2010; Holroyd and Propp, 2010), which are much faster than the standard Monte Carlo rates of $O(1/\sqrt{T})$ for computing ergodic averages. From a practical perspective, the design of deterministic sampling algorithms creates an opportunity for researchers to apply a great body of knowledge on optimization to the problem of sampling; see for example Bach et al. (2012) for an early example of this.

The domain of application of currently existing deterministic sampling algorithms is still very narrow. Importantly, the only available deterministic tool for sampling from unnormalized multivariate probability distributions is the Markov Chain Quasi-Monte Carlo method (Chen et al., 2011), but there is no theoretical result to show a better convergence rate than a standard MCMC method yet. This is very limiting because the problem of sampling from unnormalized distributions is at the heart of the field of Bayesian inference and the probabilistic programming approach to artificial intelligence (Lunn et al., 2000; Carbonetto et al., 2005; Milch and Russell, 2006; Goodman et al., 2008). At the same time, despite great progress in Monte Carlo simulation, the celebrated Gibbs sampler continues to be one of the most widely-used algorithms. For example, it is the inference engine behind popular statistics packages (Lunn et al., 2000), several tools for text analysis (Porteous et al., 2008), and Boltzmann machines (Ackley et al., 1985; Hinton and Salakhutdinov, 2006). The popularity of Gibbs stems from its generality and simplicity of implementation.

Without any doubt, it would be remarkable if we could design generic deterministic Gibbs samplers with fast (theoretical and empirical) rates of convergence. In this paper, we take steps toward achieving this goal by capitalizing on a recent idea for deterministic simulation known as herding. Herding (Welling, 2009a,b; Gelfand et al., 2010) is a deterministic procedure for generating samples $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$, such that the empirical moments $\boldsymbol{\mu}$ of the data are matched. The herding procedure, at iteration t, is as follows:

$$\begin{aligned} \mathbf{x}^{(t)} &= \arg\max_{\mathbf{x}\in\mathcal{X}} \langle \mathbf{w}^{(t-1)}, \boldsymbol{\phi}(\mathbf{x}) \rangle, \\ \mathbf{w}^{(t)} &= \mathbf{w}^{(t-1)} + \boldsymbol{\mu} - \boldsymbol{\phi}(\mathbf{x}^{(t)}), \end{aligned} \tag{1}$$

where $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{H}$ is a feature map (statistic) from $\mathcal{X}$ to a Hilbert space $\mathcal{H}$ with inner product $\langle \cdot, \cdot \rangle$, $\mathbf{w} \in \mathcal{H}$ is the vector of parameters, and $\boldsymbol{\mu} \in \mathcal{H}$ is the moment vector (expected value of $\boldsymbol{\phi}$ over the data) that we want to match. If we choose normalized features by making $\|\boldsymbol{\phi}(\mathbf{x})\|$ constant for all $\mathbf{x}$, then the update to generate samples $\mathbf{x}^{(t)}$ for $t = 1, 2, \ldots, T$ in Equation 1 is equivalent to minimizing the objective

$$J(\mathbf{x}_1, \ldots, \mathbf{x}_T) = \left\| \boldsymbol{\mu} - \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{\phi}(\mathbf{x}^{(t)}) \right\|^2, \tag{2}$$

where T may have no prior known value and $\|\cdot\| = \sqrt{\langle \cdot, \cdot \rangle}$ is the naturally defined norm based upon the inner product of the space $\mathcal{H}$ (Chen et al., 2010; Bach et al., 2012).

Herding can be used to produce samples from *normalized* probability distributions. This is done as follows. Let $\boldsymbol{\mu}$ denote a discrete, normalized probability distribution, with $\mu_i \in [0, 1]$ and $\sum_{i=1}^{n} \mu_i = 1$. A natural feature in this case is the vector $\boldsymbol{\phi}(x)$ that has all entries equal to zero, except for the entry at the position indicated by x. For instance, if $x = 2$ and $n = 5$, we have $\boldsymbol{\phi}(x) = (0, 1, 0, 0, 0)^T$. Hence, $\widehat{\boldsymbol{\mu}} = T^{-1} \sum_{t=1}^{T} \boldsymbol{\phi}(x^{(t)})$ is an empirical estimate of the distribution. In this case, one step of the herding algorithm

involves finding the largest component of the weight vector ($i^\star = \arg\max_{i\in\{1,2,..,n\}} \mathbf{w}_i^{(t-1)}$), setting $x^{(t)} = i^\star$, fixing the $i^\star$-entry of $\boldsymbol{\phi}(x^{(t)})$ to one and all other entries to zero, and updating the weight vector: $\mathbf{w}^{(t)} = \mathbf{w}^{(t-1)} + (\boldsymbol{\mu} - \boldsymbol{\phi}(x^{(t)}))$. The output is a set of samples $\{x^{(1)}, \ldots, x^{(T)}\}$ for which the empirical estimate $\widehat{\boldsymbol{\mu}}$ converges on the target distribution $\boldsymbol{\mu}$ as $O(1/T)$.

The herding method, as described thus far, only applies to normalized distributions or to problems where the objective is not to guarantee that the samples come from the right target, but to ensure that some moments are matched. An interpretation of herding in terms of Bayesian quadrature has been put forward recently by Huszar and Duvenaud (2012).

In this paper, we will show that it is possible to use herding to generate samples from more complex *unnormalized* probability distributions. In particular, we introduce a deterministic variant of the popular Gibbs sampling algorithm, which we refer to as *herded Gibbs*. While Gibbs relies on drawing samples from the *full-conditionals* at random, herded Gibbs generates the samples by matching the full-conditionals. That is, one simply applies herding to all the full-conditional distributions.

The experiments will demonstrate that the new algorithm outperforms Gibbs sampling and mean field methods in the domain of sampling from sparsely connected probabilistic graphical models, such as grid-lattice Markov random fields (MRFs) for image denoising and conditional random fields (CRFs) for natural language processing.

We advance the theory by proving that the deterministic Gibbs algorithm converges for distributions of independent variables and fully-connected probabilistic graphical models. However, a proof establishing suitable conditions that ensure convergence of herded Gibbs sampling for sparsely connected probabilistic graphical models is still unavailable.

2. Herded Gibbs Sampling

For a graph of discrete nodes $\mathcal{G} = (V, E)$, where the set of nodes are the random variables $V = \{X_i\}_{i=1}^N$, $X_i \in \mathcal{X}$, let π denote the *target distribution* defined on $\mathcal{G}$.

Gibbs sampling is one of the most popular methods to draw samples from π. Gibbs alternates (either systematically or randomly) the sampling of each variable X_i given $\mathbf{X}_{\mathcal{N}(i)} = \mathbf{x}_{\mathcal{N}(i)}$, where i is the index of the node, and $\mathcal{N}(i)$ denotes the neighbors of node i. That is, Gibbs generates each sample from its full-conditional distribution $p(X_i|\mathbf{x}_{\mathcal{N}(i)})$.

Herded Gibbs replaces the sampling from full-conditionals with herding at the level of the full-conditionals. That is, it alternates a process of matching the full-conditional distributions $p(X_i = x_i|\mathbf{X}_{\mathcal{N}(i)})$. To do this, herded Gibbs defines a set of auxiliary weights $\{w_{i,\mathbf{x}_{\mathcal{N}(i)}}\}$ for any value of $X_i = x_i$ and $\mathbf{X}_{\mathcal{N}(i)} = \mathbf{x}_{\mathcal{N}(i)}$. For ease of presentation, we assume the domain of X_i is binary, $\mathcal{X} = \{0, 1\}$, and we use one weight for every i and assignment to the neighbors $\mathbf{x}_{\mathcal{N}(i)}$. Herded Gibbs can be trivially generalized to the discrete setting by employing weight vectors in $\mathbb{R}^{|\mathcal{X}|}$ instead of scalars.

If the binary variable X_i has four binary neighbors $\mathbf{X}_{\mathcal{N}(i)}$, we must maintain $2^4 = 16$ weight vectors. Only the weight vector corresponding to the current instantiation of the neighbors is updated, as illustrated in Algorithm 1[1]. The memory complexity of herded Gibbs is exponential in the maximum node degree. Note the algorithm is a deterministic Markov process with state $(\mathbf{X}, \mathbf{W})$.

1. Code is available at `http://www.mareija.ca/research/code/`

Algorithm 1 Herded Gibbs Sampling

Input: T.

Step 1: Set $t = 0$. Initialize $\mathbf{X}^{(0)}$ in the support of π and $w^{(0)}_{i,\mathbf{x}_{\mathcal{N}(i)}}$ in $(\pi(X_i = 1|\mathbf{x}_{\mathcal{N}(i)}) - 1, \pi(X_i = 1|\mathbf{x}_{\mathcal{N}(i)}))$.

for $t = 1 \to T$ **do**

Step 2: Pick a node i according to some policy. Denote $w = w^{(t-1)}_{i,\mathbf{x}^{(t-1)}_{\mathcal{N}(i)}}$.

Step 3: If $w > 0$, set $x_i^{(t)} = 1$, otherwise set $x_i^{(t)} = 0$.

Step 4: Update weight $w^{(t)}_{i,\mathbf{x}^{(t)}_{\mathcal{N}(i)}} = w^{(t-1)}_{i,\mathbf{x}^{(t-1)}_{\mathcal{N}(i)}} + \pi(X_i = 1|\mathbf{x}^{(t-1)}_{\mathcal{N}(i)}) - x_i^{(t)}$.

Step 5: Keep the values of all the other nodes $x_j^{(t)} = x_j^{(t-1)}, \forall j \neq i$ and all the other weights $w^{(t)}_{j,\mathbf{x}_{\mathcal{N}(j)}} = w^{(t-1)}_{j,\mathbf{x}_{\mathcal{N}(j)}}, \forall j \neq i$ or $\mathbf{x}_{\mathcal{N}(j)} \neq \mathbf{x}^{(t-1)}_{\mathcal{N}(i)}$.

end for

Output: $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(T)}$

The initialization in the first step of Algorithm 1 guarantees that $\mathbf{X}^{(t)}$ always remains in the support of π with the reason to be explained in Section 3.1. For a deterministic scan policy in step 2, we take the value of variables $\mathbf{x}^{(tN)}, t \in \mathbb{N}$ as a sample sequence. Throughout the paper all experiments employ a fixed variable traversal for sample generation. We call one such traversal of the variables a *sweep*.

3. Analysis

As herded Gibbs sampling is a deterministic algorithm, the probability distribution of the sample at any step t is a single point mass and there is no stationary distribution of states. Instead, we examine the average of the sample states over time and hypothesize that it converges to the joint distribution, our target distribution, π. To make the treatment precise, we need the following definition:

Definition 1 *For a graph of discrete nodes $\mathcal{G} = (V, E)$, where the set of nodes $V = \{X_i\}_{i=1}^N$, $X_i \in \mathcal{X}$, $P_T^{(\tau)}$ is the empirical estimate of the joint distribution obtained by averaging over T samples acquired from $\mathcal{G}$. $P_T^{(\tau)}$ is derived from T samples, collected at the end of every sweep over N variables, starting from the τ^{th} sweep:*

$$P_T^{(\tau)}(\mathbf{X} = \mathbf{x}) = \frac{1}{T} \sum_{k=\tau}^{\tau+T-1} \mathbb{I}(\mathbf{X}^{(kN)} = \mathbf{x}). \tag{3}$$

The definition of $P_T^{(\tau)}$ is illustrated in Figure 1. Our goal is to prove that the limiting average sample distribution over time converges to the target distribution π. Specifically, we want to show the following:

$$\lim_{T\to\infty} P_T^{(\tau)}(\mathbf{x}) = \pi(\mathbf{x}), \forall \tau \geq 0. \tag{4}$$

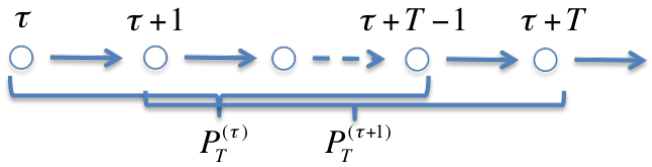


Figure 1: Sample distribution over T sweeps. Each node refers to the joint state at the end of one sweep.

If this holds, we also want to know what the convergence rate is.

3.1 Single Variable Models

We begin the theoretical analysis with a graph of one binary variable. For this graph, there is only one weight w and herded Gibbs is equivalent to the standard herding algorithm. Denote $\pi(X=1)$ as π for notational simplicity. The sequence of X is determined by the dynamics of w (shown in Figure 2a):

$$w^{(t)} = w^{(t-1)} + \pi - \mathbb{I}(w^{(t-1)} > 0), \quad X^{(t)} = \begin{cases} 1 & \text{if } w^{(t-1)} > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{5}$$

The following lemma shows that $(\pi-1, \pi]$ is the invariant interval of the dynamics.

Lemma 1 *If w is the weight of the herding dynamics of a single binary variable X with probability $P(X=1)=\pi$, and $w^{(s)} \in (\pi-1, \pi]$ at some step $s \geq 0$, then $w^{(t)} \in (\pi-1, \pi], \forall t \geq s$. Moreover, for $T \in \mathbb{N}$, we have:*

$$\sum_{t=s+1}^{s+T} \mathbb{I}[X^{(t)}=1] \in [T\pi - 1, T\pi + 1], \tag{6}$$

$$\sum_{t=s+1}^{s+T} \mathbb{I}[X^{(t)}=0] \in [T(1-\pi) - 1, T(1-\pi) + 1]. \tag{7}$$

See the proof in Appendix A. It follows immediately that the state $X=1$ is visited at a frequency close to π with an error:

$$|P_T^{(\tau)}(X=1) - \pi| \leq \frac{1}{T}. \tag{8}$$

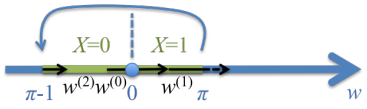
This is known as the fast moment matching property in Welling (2009a,b); Gelfand et al. (2010). When w is outside the invariant interval, it is easy to observe that w will move into it monotonically at a linear speed in a transient period. So we will always consider an initialization of $w \in (\pi-1, \pi]$ from now on.

Another immediate consequence of Lemma 1 is that the initialization in Algorithm 1 ensures that $\mathbf{X}^{(t)}$ always remain in the support of π. That is because when we consider

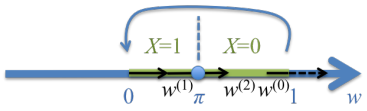
the set of iterations that involves a particular weight $w_{i,\mathbf{x}_{\mathcal{N}(i)}}$, the dynamics of that weight is equivalent to that of a single variable model with a probability $\pi(X_i = 1|\mathbf{x}_{\mathcal{N}(i)})$. If a joint state $\mathbf{X}$ is going to move outside the support at some iteration t, from e. g. $\mathbf{x}^{(t-1)} = (x_i = 0, \mathbf{x}_{-i})$ to $\mathbf{x}^{(t)} = (x_i = 1, \mathbf{x}_{-i})$ where $\mathbf{x}_{-i}$ denotes all the other variables but x_i, then the corresponding weight $w_{i,\mathbf{x}_{\mathcal{N}(i)}}^{(t-1)}$ must be positive according to the algorithm. However, the conditional probability $\pi(X_i = 1|\mathbf{x}_{\mathcal{N}(i)}) = 0$ because $\pi(\mathbf{x}^{(t-1)}) > 0$ and $\pi(\mathbf{x}^{(t)}) = 0$. Following Lemma 1 the weight $w_{i,\mathbf{x}_{\mathcal{N}(i)}}^{(t-1)} \in (-1, 0]$, leading to a contradiction. The same argument applies when $\mathbf{X}$ tries to move from $\mathbf{x}^{(t-1)} = (x_i = 1, \mathbf{x}_{-i})$ to $\mathbf{x}^{(t)} = (x_i = 0, \mathbf{x}_{-i})$. Therefore, once initialized inside the support, the samples of Algorithm 1 will remain in the support for any $t > 0$.

It will be useful for the next section to introduce an equivalent representation of the weight dynamics by taking a one-to-one mapping $w \leftarrow w \mod 1$ (we define $1 \mod 1 = 1$) with a little abuse of the symbol w. We think of the new variable w as updated by a constant translation vector in a circular unit interval $(0, 1]$ as shown in Figure 2b. That is,

$$w^{(t)} = (w^{(t-1)} + \pi) \mod 1, \quad X^{(t)} = \begin{cases} 1 & \text{if } w^{(t-1)} < \pi \\ 0 & \text{otherwise} \end{cases}. \tag{9}$$



(a) Representation in Equation 5.



(b) Equivalent representation.

Figure 2: Herding dynamics for a single variable. Black arrows show the trajectory of $w^{(t)}$ for 2 iterations.

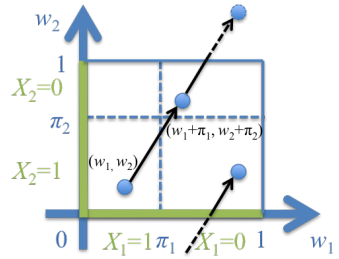


Figure 3: Herding dynamics for two independent variables in the equivalent representation.

3.2 Empty Graphs

The analysis of herded Gibbs for empty graphs is a natural extension of that for single variables. In an empty graph, all the variables are independent of each other and herded Gibbs reduces to running N one-variable chains in parallel. Denote the marginal distribution $\pi_i := \pi(X_i = 1)$.

Examples of failing convergence in the presence of rational ratios between the π_is were observed in Bach et al. (2012). There the need for further theoretical research on this matter

was pointed out. The following theorem provides a sufficient condition for convergence in the restricted domain of empty graphs.

Theorem 2 *For an empty graph, when herded Gibbs has a fixed scanning order, and* $\{1, \pi_1, \ldots, \pi_N\}$ *are rationally independent, the empirical distribution* $P_T^{(\tau)}$ *converges to the target distribution* π *as* $T \to \infty$ *for any* $\tau \geq 0$.

A set of n real numbers, $x_1, x_2, \ldots, x_n$, is said to be rationally independent if for any set of rational numbers, $a_1, a_2, \ldots, a_n$, we have $\sum_{i=1}^n a_i x_i = 0 \Leftrightarrow a_i = 0, \forall 1 \leq i \leq n$.

Proof For an empty graph of N independent vertices, the dynamics of the weight vector $\mathbf{w}$ after one sweep over all variables are equivalent to a constant translation mapping in an N-dimensional circular unit space $(0,1]^N$, as shown in Figure 3:

$$\begin{aligned} \mathbf{w}^{(t)} &= (\mathbf{w}^{(t-1)} + \boldsymbol{\pi}) \mod 1 \\ &= (\mathbf{w}^{(0)} + t\boldsymbol{\pi}) \mod 1, \quad x_i^{(t)} = \begin{cases} 1 & \text{if } w_i^{(t-1)} < \pi_i \\ 0 & \text{otherwise} \end{cases}, \forall 1 \leq i \leq N. \end{aligned} \tag{10}$$

The Kronecker-Weyl theorem (Weyl, 1916) states that the sequence $\tilde{\mathbf{w}}^{(t)} = t\pi \mod 1, t \in \mathbb{Z}^+$ is equidistributed (or uniformly distributed) on $(0,1]^N$ if and only if $(1, \pi_1, \ldots, \pi_N)$ is rationally independent. Intuitively, when $(1, \pi_1, \ldots, \pi_N)$ is rationally independent, the trajectory of $\tilde{\mathbf{w}}^{(t)}$ can not form a closed loop in the circular unit space and will thereby cover the entire space uniformly.

Since we can define a one-to-one volume preserving transformation between $\tilde{\mathbf{w}}^{(t)}$ and $\mathbf{w}^{(t)}$ as $(\tilde{\mathbf{w}}^{(t)} + \mathbf{w}^{(0)}) \mod 1 = \mathbf{w}^{(t)}$, the sequence of weights $\{\mathbf{w}^{(t)}\}$ is also uniformly distributed in $(0,1]^N$.

Now define the mapping from a state value x_i to an interval of w_i as

$$A_i(x) = \begin{cases} (0, \pi_i] & \text{if } x = 1 \\ (\pi_i, 1] & \text{if } x = 0 \end{cases} \tag{11}$$

and let $|A_i|$ be its measure. We obtain the limiting distribution of the joint state as

$$\begin{aligned} \lim_{T\to\infty} P_T^{(\tau)}(\mathbf{X} = \mathbf{x}) &= \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{I}\left[\mathbf{w}^{(t-1)} \in \prod_{i=1}^{N} A_i(x_i)\right] \\ &= \prod_{i=1}^{N} |A_i(x_i)| = \prod_{i=1}^{N} \pi(X_i = x_i) = \pi(\mathbf{X} = \mathbf{x}). \end{aligned} \tag{12}$$

■

The rational independence condition in the application of the Kronecker-Weyl theorem ensures that the probabilistic independence between different variables will be respected by the herding dynamics. When the condition fails, the convergence of joint distribution is not guaranteed but the marginal distribution of each variables will still converge to their target distribution because the N one-variable chains run independently from each other.

3.3 Fully-Connected Graphs

When herded Gibbs is applied to fully-connected (complete) graphs, convergence is guaranteed even with rationally dependent conditional probabilities. In fact, herded Gibbs converges to the target joint distribution at a rate of $O(1/T)$ with a $O(\log(T))$ burn-in period. This statement is formalized in Theorem 3 and a corollary when we ignore the burn-in period, with proofs provided respectively in Appendix B.4 and B.5.

Theorem 3 *For a fully-connected graph, when herded Gibbs has a fixed scanning order and a Dobrushin coefficient of the corresponding Gibbs sampler* $\eta < 1$*, there exist constants* $l > 0$*, and* $B > 0$ *such that*

$$d_v(P_T^{(\tau)} - \pi) \leq \frac{\lambda}{T}, \forall T \geq T^*, \tau > \tau^*(T), \tag{13}$$

where $\lambda = \frac{2N(1+\eta)}{l(1-\eta)}$, $T^* = \frac{2B}{l}$, $\tau^*(T) = \log_{\frac{2}{1+\eta}}\left(\frac{(1-\eta)lT}{4N}\right)$*, and* $d_v(\delta\pi) := \frac{1}{2}||\delta\pi||_1$.

Corollary 4 *When the conditions of Theorem 3 hold, and we start collecting samples at the end of every sweep from the beginning, that is setting* $\tau = 0$*, the error of the sample distribution is bounded by:*

$$d_v(P_T^{(\tau=0)} - \pi) \leq \frac{\lambda + \tau^*(T)}{T} = O(\frac{\log(T)}{T}), \quad \forall T \geq T^* + \tau^*(T^*). \tag{14}$$

The Dobrushin ergodic coefficient (Brémaud, 1999) measures the geometric convergence rate of a Markov chain. The constant l in the convergence rate can be interpreted as a lower bound of the transition probability between any pair of states in the support of the target distribution. For a strictly positive distribution, the constants l and B are

$$l = \pi_{\min}^N, \quad B = \pi_{\min}^N + \frac{1 - (2\pi_{\min})^N}{1 - 2\pi_{\min}}. \tag{15}$$

where $\pi_{\min}$ is the minimal conditional probability $\pi_{\min} = \min_{1\leq i\leq N, \mathbf{x}_{-i}} \pi(x_i|\mathbf{x}_{-i})$. We refer the readers to Equation 34 in Proposition 5 in the appendix for a general distribution. Notice that l has an exponential term, with N in the exponent, leading to an exponentially large constant. This is unavoidable for any sampling algorithm when considering the convergence to a joint distribution with 2^N states. As for the marginal distributions, it is obvious that the convergence rate of herded Gibbs is also $O(1/T)$ because marginal probabilities are linear functions of the joint distribution. In fact, we observe very rapid convergence results for the marginals in practice, so stronger theoretical results about the convergence of the marginal distributions seem plausible.

The proof proceeds by first bounding the discrepancy between the chain of empirical estimates of the joint obtained by averaging over T herded Gibbs samples, $\{P_T^{(s)}\}, s \geq \tau$, and a Gibbs chain initialized at $P_T^{(\tau)}$. After one sweep over N variables, this discrepancy is bounded above by $2N/lT$.

The Gibbs chain has geometric convergence to π and the distance between the Gibbs and herded Gibbs chains decreases as $O(1/T)$. When the distance between $P_T^{(\tau)}$ and π is

sufficiently large, the geometric convergence rate to π dominates the discrepancy of herded Gibbs and thus we infer that $P_T^{(\tau)}$ converges to a neighborhood of π geometrically in time τ for a fixed T. To round-off the proof, we must find a limiting value for τ. The proof concludes with an $O(\log(T))$ burn-in for τ.

When there exist conditional independencies in a distribution, we can still apply herded Gibbs with a fully connected graph and treat all the other variables as the Markov blanket of the variable to be sampled. Theorem 3 and its corollary still apply. Alternatively, we can apply herded Gibbs on a more compact representation with an incomplete graph. It requires less memory to run herded Gibbs because the number of weights depends exponentially on the neighborhood size. However, for a generic graph we have no mathematical guarantees for the convergence rate of herded Gibbs. In fact, one can easily construct synthetic examples for which herded Gibbs does not seem to converge to the true marginals and joint distribution. For the examples covered by our theorems and for examples with real data, herded Gibbs demonstrates good behaviour. The exact conditions under which herded Gibbs converges for sparsely connected graphs are still unknown.

4. Experiments

We illustrate the performance of herded Gibbs with two synthetic examples and two real experiments for image denoising and natural language processing respectively.

4.1 Simple Complete Graph

We begin with an illustration of how herded Gibbs substantially outperforms Gibbs and a deterministic Gibbs sampler based on MCQMC on a simple complete graph. The MCQMC algorithm replaces the random number generator of the regular Gibbs sampler with a completely uniformly distributed (CUD) sequence (Chen et al., 2011). We consider a fully-connected model of two variables, X_1 and X_2, as shown in Figure 4; the joint distribution of these variables is shown in Table 1. We run each sampler for 2.6×10^5 iterations. We use a small linear feedback shift registers (LFSR) described in (Chen et al., 2012) to generate the CUD sequence and choose the size of the LFSR so that the entire period of the sequence will be used for one run of the Markov chain. Both Gibbs and MCQMC-based Gibbs are run 100 times with different random seeds to assess their average performance. Herded Gibbs is run only once because different initialization does not show noticeable difference in its performance. Figure 5 shows the marginal distribution $P(X_1 = 1)$ and the joint distribution approximated by all the algorithms for different ϵ. As ϵ decreases, all the approaches require more iterations to converge, but herded Gibbs clearly outperforms the other two algorithms. Figure 5c also shows that herding does indeed exhibit a linear convergence rate. MCQMC-based Gibbs does not show any improvement on the error of the sample distribution compared to the standard Gibbs.

4.2 Simple Incomplete Graph

We also illustrate a simple counterexample where the sample distribution of herded Gibbs does not converge to the target distribution when the graph is incomplete. Figure 7 shows a four-variable graphical model with two missing edges, $X_1 - X_4$, $X_2 - X_3$. The unary

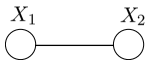


Figure 4: Two-variable model.

	$\mathbf{X_1 = 0}$	$\mathbf{X_1 = 1}$	$\mathbf{P(X_2)}$
$\mathbf{X_2 = 0}$	$1/4 - \epsilon$	ϵ	$1/4$
$\mathbf{X_2 = 1}$	ϵ	$3/4 - \epsilon$	$3/4$
$\mathbf{P(X_1)}$	$1/4$	$3/4$	1

Table 1: Joint distribution of the two-variable model.

and pairwise energies of existing edges are random sampled from a standard Gaussian distribution $\mathcal{N}(0, 1)$. We apply herded Gibbs for 10^8 iterations and compare the joint sample distribution with the true distribution. While the discrepancy is marginal as depicted in Figure 6b, the L_1 error plot in 6b does not show a tendency to converging to zero.

4.3 MRF for Image Denoising

Next, we consider the standard setting of a grid-lattice MRF for image denoising. Let us assume that we have a binary image corrupted by noise, and that we want to infer the original clean image. Let $X_i \in \{-1, +1\}$ denote the unknown true value of pixel i, and y_i the observed, noise-corrupted value of this pixel. We take advantage of the fact that neighboring pixels are likely to have the same label by defining an MRF with an Ising prior. That is, we specify a rectangular 2D lattice with the following pair-wise clique potentials:

$$\psi_{ij}(x_i, x_j) = \begin{pmatrix} e^{J_{ij}} & e^{-J_{ij}} \\ e^{-J_{ij}} & e^{J_{ij}} \end{pmatrix} \tag{16}$$

and joint distribution:

$$p(\mathbf{x}|\mathbf{J}) = \frac{1}{Z(\mathbf{J})} \prod_{i \sim j} \psi_{ij}(x_i, x_j) = \frac{1}{Z(\mathbf{J})} \exp\left(\frac{1}{2} \sum_{i \sim j} J_{ij} x_i x_j \right), \tag{17}$$

where $i \sim j$ is used to indicate that nodes i and j are connected. The known parameters J_{ij} establish the coupling strength between nodes i and j. Note that the matrix $\mathbf{J}$ is symmetric. If all the $J_{ij} > 0$, then neighboring pixels are likely to be in the same state.

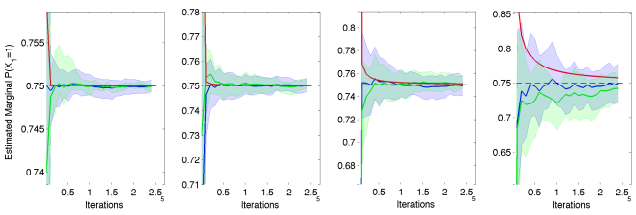
The MRF model combines the Ising prior with a likelihood model as follows:

$$p(\mathbf{x}, \mathbf{y}) \quad = \quad p(\mathbf{x}) p(\mathbf{y}|\mathbf{x}) = \left[\frac{1}{Z} \prod_{i \sim j} \psi_{ij}(x_i, x_j) \right] \cdot \left[\prod_i p(y_i|x_i) \right]. \tag{18}$$

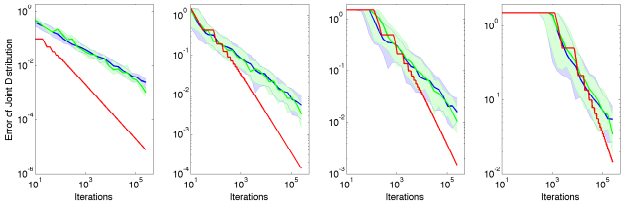
The potentials ψ_{ij} encourage label smoothness. The likelihood terms $p(y_i|x_i)$ are conditionally independent (e.g. Gaussians with known variance σ^2 and mean $\boldsymbol{\mu}$ centered at each value of x_i, denoted μ_{x_i}). In more precise terms,

$$p(\mathbf{x}, \mathbf{y}|\mathbf{J}, \boldsymbol{\mu}, \sigma) = \frac{1}{Z(\mathbf{J}, \boldsymbol{\mu}, \sigma)} \exp\left(\frac{1}{2} \sum_{i \sim j} J_{ij} x_i x_j - \frac{1}{2\sigma^2} \sum_i (y_i - \mu_{x_i})^2 \right). \tag{19}$$

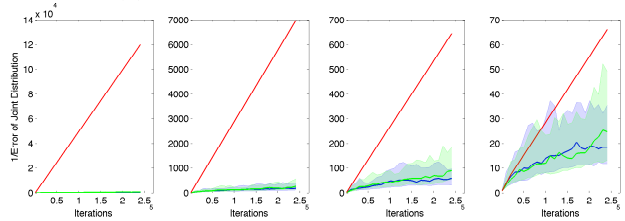
When the coupling parameters J_{ij} are identical, say $J_{ij} = J$, we have $\sum_{ij} J_{ij} f(x_i, x_j) = J \sum_{ij} f(x_i, x_j)$. Hence, different neighbor configurations result in the same value of $J \sum_{ij} f(x_i, x_j)$.



(a) Approximate marginals.

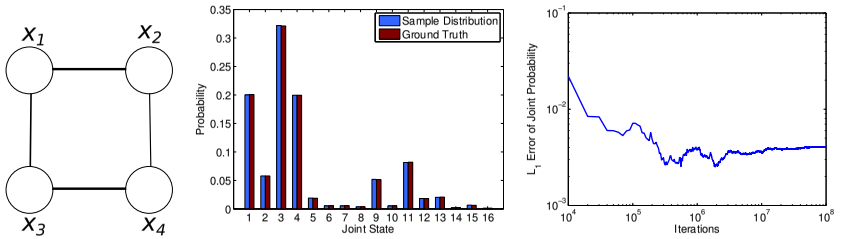


(b) Log-log plot of the L_1 error of the joint distribution.



(c) Inverse of the L_1 error of the joint distribution.

Figure 5: Approximating a marginal (a) and joint (b, c) distribution with Gibbs (blue), MCQMC-based Gibbs (green) and herded Gibbs (red) for an MRF of two variables, constructed so as to make the move from state $(0, 0)$ to $(1, 1)$ progressively more difficult as ϵ decreases. The four columns, from left to right, are for $\epsilon = 0.1$, $\epsilon = 0.01$, $\epsilon = 0.001$ and $\epsilon = 0.0001$. Table 1 provides the joint distribution for these variables. The shaded areas for Gibbs and MCQMC-based Gibbs correspond to 25%-75% quantile of 100 runs. Rows (b) and (c) illustrate that the empirical convergence rate of herded Gibbs matches the expected theoretical rate. As the error of herded Gibbs in (b) and (c) frequently drops to extremely small values for some iterations and jumps back, we plot the upper-bound (envelope) of the error for herded Gibbs defined as $\tilde{e}_t = \max_{\tau \geq t} e_\tau$ to remove the oscillating behavior for a better visualization.



(a) **(b)** **(c)**

Figure 7: Four-variable model represented as an incomplete graph. (a): Graphical Model. (b): Joint distribution of samples from herded Gibbs vs. the ground truth. (c): Log-log plot of the L_1 error of the joint sample distribution.

If we store the conditionals for configurations with the same sum together, we only need to store as many conditionals as different possible values that the sum could take. This enables us to develop a shared version of herded Gibbs that is more memory efficient where we only maintain and update weights for *distinct states* of the Markov blanket of each variable. The shared version of herded Gibbs also exhibits a different dynamics as the standard version as shown in the following result, and the convergence property of this algorithm remains an open problem.

In this exemplary image denoising experiment, noisy versions of the binary image, depicted in Figure 8 (left), were created through the addition of Gaussian noise, with varying σ. Figure 8 (right) shows a corrupted image with $\sigma = 4$. The L_2 reconstruction errors as a function of the number of iterations, for this example, are shown in Figure 9. The plot compares the herded Gibbs method against Gibbs and two versions of mean field with different damping factors (Murphy, 2012). The results demonstrate that the herded Gibbs techniques are among the best methods for solving this task.

A comparison for different values σ is presented in Table 2. As expected mean field does well in the low-noise scenario, but the performance of the shared version of herded Gibbs as the noise increases is significantly better.

4.4 CRF for Named Entity Recognition

Named entity recognition (NER) involves the identification of entities, such as people and locations, within a text sample. A conditional random field (CRF) for NER models the relationship between entity labels and sentences with a conditional probability distribution: $P(X|Y, \theta)$, where X is a labeling, Y is a sentence, and θ is a vector of coupling parameters. The parameters, θ, are feature weights and model relationships between variables X_i and Y_j or X_i and X_j. A chain CRF only employs relationships between adjacent variables, whereas a skip-chain CRF can employ relationships between variables where subscripts i and j differ dramatically. Skip-chain CRFs are important in language tasks, such as NER

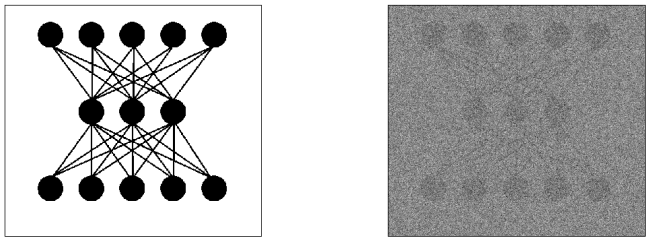


Figure 8: Original image (left) and its corrupted version (right), with noise parameter $\sigma = 4$.

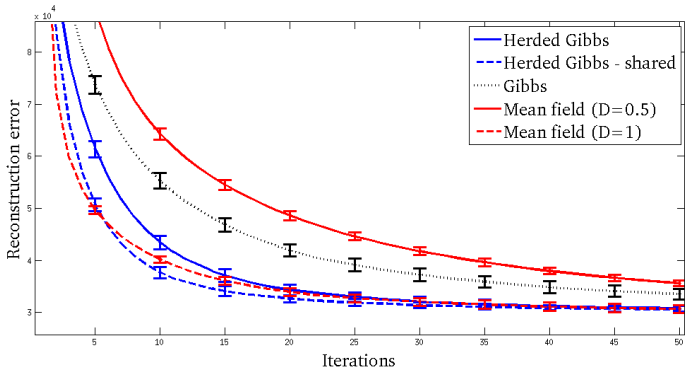


Figure 9: Reconstruction errors for the image denoising task. The results are averaged across 10 corrupted images with Gaussian noise $\mathcal{N}(0, 16)$. The error bars correspond to one standard deviation. Mean field requires the specification of the damping factor D.

and semantic role labeling, because they allow us to model long dependencies in a stream of words, see Figure 10.

Once the parameters have been learned, the CRF can be used for inference; a labeling for some sentence Y is found by maximizing the above probability. Inference for CRF models in the NER domain is typically carried out with the Viterbi algorithm. However, if we want to accommodate long term dependencies, thus resulting in the so called skip-chain CRFs,

Method \ σ	2	4	6	8
Herded Gibbs	21.58(0.26)	32.07(0.98)	47.52(1.64)	67.93(2.78)
Herded Gibbs - shared	22.24(0.29)	**31.40**(0.59)	**42.62**(1.98)	**58.49**(2.86)
Gibbs	21.63(0.28)	37.20(1.23)	63.78(2.41)	90.27(3.48)
Mean field (D=0.5)	**15.52**(0.30)	41.76(0.71)	76.24(1.65)	104.08(1.93)
Mean field (D=1)	17.67(0.40)	32.04(0.76)	51.19(1.44)	74.74(2.21)

Table 2: Errors of image denoising example after 30 iterations (all measurements have been scaled by $\times 10^{-3}$). We use an Ising prior with $J_{ij} = 1$ and four Gaussian noise models with different σ's. For each σ, we generated 10 corrupted images by adding Gaussian noise. The final results shown here are averages and standard deviations (in parentheses) across the 10 corrupted images. D denotes the damping factor in mean field.

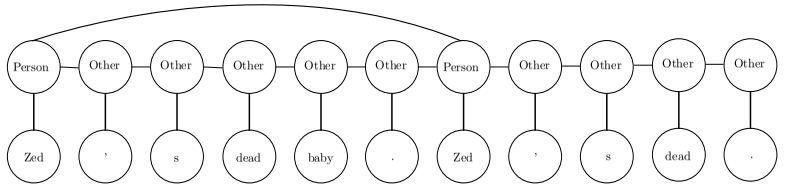


Figure 10: Typical skip-chain CRF model for named entity recognition.

Viterbi becomes prohibitively expensive. To surmount this problem, the Stanford named entity recognizer (J. R. Finkel and Manning, 2005) makes use of annealed Gibbs sampling.

To demonstrate herded Gibbs on a practical application of great interest in text mining, we modify the standard inference procedure of the Stanford named entity recognizer by replacing the annealed Gibbs sampler with the herded Gibbs sampler. The herded Gibbs sampler is not annealed. Notice that the label of a word X_i is a discrete variable with possibly multiple values. As discussed in Section 2 we generalize herded Gibbs for binary variables to discrete variables by assigning a different weight $w_{i,\mathbf{x}_{\mathcal{N}(i)}}$ for each value of X_i. To find the maximum a posteriori sequence X, we compute the joint discrete probability of every sample and choose the one with the highest probability as the prediction. The faster a sampler mixes in the state space, the more likely that a sample with high probability will be generated given a the same amount of time. In order to be able to compare against Viterbi, we have purposely chosen to use single-chain CRFs. We remind the reader, however, that the herded Gibbs algorithm could be used in cases where Viterbi inference is not possible.

We used the pre-trained 3-class CRF model in the Stanford NER package (J. R. Finkel and Manning, 2005). This model is a linear chain CRF with pre-defined features and pre-trained feature weights, θ. For the test set, we used the corpus for the NIST 1999 IE-ER Evaluation. Performance is measured in per-entity F_1 $\left(F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}\right)$. For all the

"Pumpkin" (*Tim Roth*) and "Honey Bunny" (*Amanda Plummer*) are having breakfast in a diner. They decide to rob it after realizing they could make money off the customers as well as the business, as they did during their previous heist. Moments after they initiate the hold-up, the scene breaks off and the title credits roll. As *Jules Winnfield* (*Samuel L. Jackson*) drives, *Vincent Vega* (*John Travolta*) talks about his experiences in **Europe**, from where he has just returned: the hash bars in **Amsterdam**, the French McDonald's and its "Royale with Cheese".

Figure 11: Results for the application of the NER CRF to a random Wikipedia sample (Wik, 2013). Entities are automatically classified as person (red italic), location (green boldfaced) and organization (orange underlined).

methods, except Viterbi, we show F_1 scores after 100, 400 and 800 iterations in Table 3. For Gibbs, the results shown are the averages and standard deviations over 5 random runs. We used a linear annealing schedule for Gibbs. As the results illustrate, herded Gibbs attains the same accuracy as Viterbi and it is faster than annealed Gibbs. Unlike Viterbi, herded Gibbs can be easily applied to skip-chain CRFs. After only 400 iterations (90.5 seconds), herded Gibbs already achieves an F_1 score of 84.75, while Gibbs, even after 800 iterations (115.9 seconds) only achieves an F_1 score of 84.61. The experiment thus clearly demonstrates that (i) herded Gibbs does no worse than the optimal solution, Viterbi, and (ii) herded Gibbs yields more accurate results for the same amount of computation than Gibbs sampling. Figure 11 provides a representative NER example of the performance of Gibbs, herded Gibbs and Viterbi (all methods produced the same annotation for this short example).

Method \ Iterations	100	400	800
Annealed Gibbs	84.36(0.16) [55.73s]	84.51(0.10) [83.49s]	84.61(0.05) [115.92s]
Herded Gibbs	84.70 [59.08s]	84.75 [90.48s]	84.81 [132.00s]
Viterbi			84.81[46.74s]

Table 3: F_1 scores for Gibbs, herded Gibbs and Viterbi on the NER task. The average computational time each approach took to do inference for the entire test set is listed (in square brackets). After only 400 iterations (90.48 seconds), herded Gibbs already achieves an F_1 score of 84.75, while Gibbs, even after 800 iterations (115.92 seconds) only achieves an F_1 score of 84.61. For the same computation, herded Gibbs is more accurate than Gibbs.

5. Conclusions and Future Work

In this paper, we introduced herded Gibbs, a deterministic variant of the popular Gibbs sampling algorithm. While Gibbs relies on drawing samples from the full-conditionals at random, herded Gibbs generates the samples by matching the full-conditionals. Importantly, the herded Gibbs algorithm is very close to the Gibbs algorithm and hence retains its simplicity of implementation.

The synthetic, denoising and named entity recognition experiments provided evidence that herded Gibbs outperforms Gibbs sampling. However, as discussed, herded Gibbs requires storage of the conditional distributions for all instantiations of the neighbors in the worst case. This storage requirement indicates that it is more suitable for sparse probabilistic graphical models, such as the CRFs used in information extraction. At the other extreme, the paper advanced the theory of deterministic sampling by showing that herded Gibbs converges with rate $O(1/T)$ for models with independent variables and fully-connected models. Thus, there is gap between theory and practice that needs to be narrowed. We do not anticipate that this will be an easy task, but it is certainly a key direction for future work.

We should mention that it is also possible to design parallel versions of herded Gibbs in an asynchronous Jacobi fashion. Preliminary study shows that these are less efficient than the synchronous Gauss-Seidel version of herded Gibbs discussed in this paper. However, if many cores are available, we strongly recommend the parallel implementation as it will likely outperform the current sequential implementation.

The design of efficient herding algorithms for densely connected probabilistic graphical models remains an important area for future research. Such algorithms, in conjunction with Rao Blackwellization, would enable us to attack many statistical inference tasks, including Bayesian variable selection and Dirichlet processes.

There are also interesting connections with other algorithms to explore. First, herding has ties to multicanonical sampling algorithms (Bornn et al., 2013), which while not deterministic, employ similar biasing/reweighting schemes. Second, if for a fully connected graphical model we build a new graph where every state is a node and directed connections exist between nodes that can be reached with a single herded Gibbs update, then herded Gibbs is very similar to the Rotor-Router model of Holroyd and Propp (2010)[2]. This deterministic analogue of a random walk has provably superior concentration rates for quantities such as normalized hitting frequencies, hitting times and occupation frequencies. In line with our own convergence results, it is shown that discrepancies in these quantities decrease as $O(1/T)$ instead of the usual $O(1/\sqrt{T})$. We expect that many of the results from this literature apply to herded Gibbs as well. The connection with the work of Art Owen and colleagues, see for example Chen et al. (2011), also needs to be explored further. Their work uses *completely uniformly distributed (CUD) sequences* to drive Markov chain Monte Carlo schemes. It is not clear, following discussions with Art Owen, that CUD sequences can be constructed in a greedy way as in herding.

2. We thank Art Owen for pointing out this connection.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 0914783, 0928427, 1018433, 1216045, by NSERC, CIFAR's Neural Computation and Adaptive Perception Program, by DARPA under Grant No. FA8750-14-2-0117, by ARO under Grant No. W911NF-15-1-0172, by the Amazon AWS Research Grant, and by the Natural Sciences and Engineering Research Council of Canada.

Appendix A. Proof of Lemma 1

Proof We first show that $w \in (\pi - 1, \pi], \forall t \geq s$. This is easy to observe by induction as $w^{(s)} \in (\pi - 1, \pi]$ and if $w^{(t)} \in (\pi - 1, \pi]$ for some $t \geq s$, then, following Equation 5, we have:

$$w^{(t+1)} = \begin{cases} w^{(t)} + \pi - 1 \in (\pi - 1, 2\pi - 1] \subseteq (\pi - 1, \pi], & \text{if } w^{(t)} > 0, \\ w^{(t)} + \pi \in (2\pi - 1, \pi] \subseteq (\pi - 1, \pi], & \text{otherwise.} \end{cases} \tag{20}$$

Summing up both sides of Equation 5 over t immediately gives us the result of Equation 6 since:

$$T\pi - \sum_{t=s+1}^{s+T} \mathbb{I}[X^{(t)} = 1] = w^{(s+T)} - w^{(s)} \in [-1, 1]. \tag{21}$$

In addition, Equation 7 follows by observing that $\mathbb{I}[X^{(t)} = 0] = 1 - \mathbb{I}[X^{(t)} = 1]$. ■

Appendix B. Proof of Theorem 3

In this appendix, we give an upper bound for the convergence rate of the sampling distribution in fully connected graphs. As herded Gibbs sampling is deterministic, the distribution of a variable's state at every iteration degenerates to a single state. As such, we study here the empirical distribution of a collection of samples.

The structure of the proof is as follows (with notation defined in the next subsection): We study the distribution distance between the invariant distribution π and the empirical distribution of T samples collected starting from sweep τ, $P_T^{(\tau)}$. We show that the distance decreases as $\tau \Rightarrow \tau + 1$ with the help of an auxiliary regular Gibbs sampling Markov chain initialized at $\pi^{(0)} = P_T^{(\tau)}$, as shown in Figure 12. On the one hand, the distance between the regular Gibbs chain after one iteration, $\pi^{(1)}$, and π decreases according to the geometric convergence property of MCMC algorithms on compact state spaces. On the other hand, we show that in one step the distance between $P_T^{(\tau+1)}$ and $\pi^{(1)}$ increases by at most $O(1/T)$. Since the $O(1/T)$ distance term dominates the exponentially small distance term, the distance between $P_T^{(\tau+1)}$ and π is bounded by $O(1/T)$. Moreover, after a short burn-in period, $L = O(\log(T))$, the empirical distribution $P_T^{(\tau+L)}$ will have an approximation error in the order of $O(1/T)$.

B.1 Notation

Assume without loss of generality that in the systematic scanning policy, the variables are sampled in the order $1, 2, \cdots, N$.

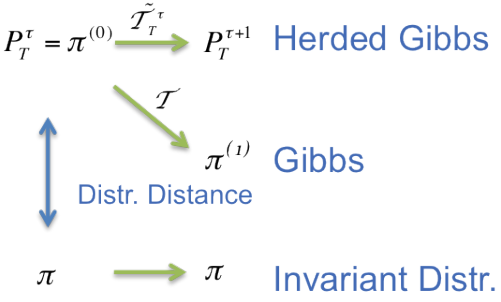


Figure 12: Transition kernels and relevant distances for the proof of Theorem 3.

B.1.1 State Distribution

- Denote by $\mathcal{X}_+$ the support of the distribution π, that is, the set of states with positive probability.
- We use τ to denote the time in terms of sweeps over all of the N variables, and t to denote the time in terms of steps where one step constitutes the updating of one variable. For example, at the end of τ sweeps, we have $t = \tau N$.
- Recall the sample/empirical distribution, $P_T^{(\tau)}$, presented in Definition 1.
- Denote the sample/empirical distribution at the i^{th} step within a sweep as $P_{T,i}^{(\tau)}, \tau \geq 0, T > 0, 0 \leq i \leq N$, as shown in Figure 13:

$$P_{T,i}^{(\tau)}(\mathbf{X} = \mathbf{x}) = \frac{1}{T} \sum_{k=\tau}^{\tau+T-1} \mathbb{I}(\mathbf{X}^{(kN+i)} = \mathbf{x}).$$

This is the distribution of T samples collected at the i^{th} step of every sweep, starting from the τ^{th} sweep. Clearly, $P_T^{(\tau)} = P_{T,0}^{(\tau)} = P_{T,N}^{(\tau-1)}$.

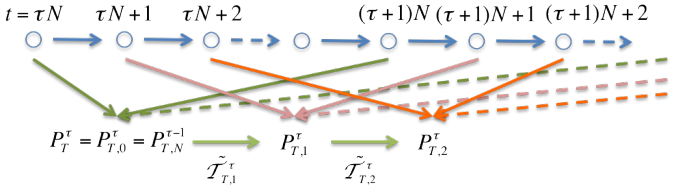


Figure 13: Distribution over time within a sweep.

- Denote the distribution of a regular Gibbs sampling Markov chain after L sweeps of updates over the N variables with $\pi^{(L)}, L \geq 0$.

  For a given time τ, we construct a Gibbs Markov chain with initial distribution $\pi^0 = P_T^{(\tau)}$ and the same scanning order of herded Gibbs, as shown in Figure 12.

B.1.2 Transition Kernel

- Denote the transition kernel of regular Gibbs for the step of updating a variable X_i with $\mathcal{T}_i$, and for a whole sweep with $\mathcal{T}$.

  By definition, $\pi^0\mathcal{T} = \pi^1$. The transition kernel for a single step can be represented as a $2^N \times 2^N$ matrix:

$$\mathcal{T}_i(\mathbf{x}, \mathbf{y}) = \begin{cases} 0, & \text{if } \mathbf{x}_{-i} \neq \mathbf{y}_{-i} \\ \pi(X_i = y_i | \mathbf{x}_{-i}), & \text{otherwise} \end{cases}, 1 \leq i \leq N, \mathbf{x}, \mathbf{y} \in \{0,1\}^N, \tag{22}$$

  where $\mathbf{x}$ is the current state vector of N variables, $\mathbf{y}$ is the state of the next step, and $\mathbf{x}_{-i}$ denotes all the components of $\mathbf{x}$ excluding the i^{th} component. If $\pi(\mathbf{x}_{-i}) = 0$, the conditional probability is undefined and we set it with an arbitrary distribution. Consequently, $\mathcal{T}$ can also be represented as:

$$\mathcal{T} = \mathcal{T}_1 \mathcal{T}_2 \cdots \mathcal{T}_N.$$

- Denote the Dobrushin ergodic coefficient (Brémaud, 1999) of the regular Gibbs kernel with $\eta \in [0,1]$. When $\eta < 1$, the regular Gibbs sampler has a geometric rate of convergence of

$$d_v(\pi^{(1)} - \pi) = d_v(\mathcal{T}\pi^{(0)} - \pi) \leq \eta d_v(\pi^{(0)} - \pi), \forall \pi^{(0)}. \tag{23}$$

  A common sufficient condition for $\eta < 1$ is that $\pi(\mathbf{X})$ is strictly positive.

- Consider the sequence of sample distributions $P_T^{(\tau)}, \tau = 0, 1, \cdots$ in Figures 1 and 13. We define the transition kernel of herded Gibbs for the step of updating variable X_i from $P_{T,i-1}^{(\tau)}$ to $P_{T,i}^{(\tau)}$ with $\tilde{\mathcal{T}}_{T,i}^{(\tau)}$, and for a whole sweep from $P_T^{(\tau)}$ to $P_T^{(\tau+1)}$ with $\tilde{\mathcal{T}}_T^{(\tau)}$. Unlike regular Gibbs, the transition kernel is not homogeneous. It depends on both the time τ and the sample size T. Nevertheless, we can still represent the single step transition kernel as a matrix:

$$\tilde{\mathcal{T}}_{T,i}^{(\tau)}(\mathbf{x}, \mathbf{y}) = \begin{cases} 0, & \text{if } \mathbf{x}_{-i} \neq \mathbf{y}_{-i} \\ P_{T,i}^{(\tau)}(X_i = y_i | \mathbf{x}_{-i}), & \text{if } \mathbf{x}_{-i} = \mathbf{y}_{-i} \end{cases}, 1 \leq i \leq N, \mathbf{x}, \mathbf{y} \in \{0,1\}^N, \tag{24}$$

  where $P_{T,i}^{(\tau)}(X_i = y_i | \mathbf{x}_{-i})$ is defined as:

$$P_{T,i}^{(\tau)}(X_i = y_i | \mathbf{x}_{-i}) = \frac{N_{\text{num}}}{N_{\text{den}}},$$

$$N_{\text{num}} = T P_{T,i}^{(\tau)}(\mathbf{X}_{-i} = \mathbf{x}_{-i}, X_i = y_i) = \sum_{k=\tau}^{\tau+T-1} \mathbb{I}(\mathbf{X}_{-i}^{(kN+i)} = \mathbf{x}_{-i}, X_i^{(kN+i)} = y_i),$$

$$N_{\text{den}} = T P_{T,i-1}^{(\tau)}(\mathbf{X}_{-i} = \mathbf{x}_{-i}) = \sum_{k=\tau}^{\tau+T-1} \mathbb{I}(\mathbf{X}_{-i}^{(kN+i-1)} = \mathbf{x}_{-i}), \tag{25}$$

where N_{num} is the number of occurrences of a joint state, and N_{den} is the number of occurrences of a conditioning state in the previous step. When $\pi(\mathbf{x}_{-i}) = 0$, we know that $N_{\text{den}} = 0$ with a proper initialization of herded Gibbs, and we simply set $\tilde{\mathcal{T}}_{T,i}^{(\tau)} = \mathcal{T}_i$ for these entries. It is not hard to verify the following identity by expanding every term with its definition

$$P_{T,i}^{(\tau)} = P_{T,i-1}^{(\tau)} \tilde{\mathcal{T}}_{T,i}^{(\tau)},$$

and consequently,

$$P_T^{(\tau+1)} = P_T^{(\tau)} \tilde{\mathcal{T}}_T^{(\tau)},$$

with

$$\tilde{\mathcal{T}}_T^{(\tau)} = \tilde{\mathcal{T}}_{T,1}^{(\tau)} \tilde{\mathcal{T}}_{T,2}^{(\tau)} \cdots \tilde{\mathcal{T}}_{T,N}^{(\tau)}.$$

B.2 Linear Visiting Rate

We prove in this section that every joint state in the support of the target distribution is visited, at least, at a linear rate. This result will be used to measure the distance between the Gibbs and herded Gibbs transition kernels.

Proposition 5 *If a graph is fully connected, herded Gibbs sampling scans variables in a fixed order, and the corresponding Gibbs sampling Markov chain is irreducible, then for any state $\mathbf{x} \in \mathcal{X}_+$ and any index $i \in [1, N]$, the state is visited at least at a linear rate. Specifically,*

$$\exists l > 0, B > 0, s.t., \forall i \in [1, N], \mathbf{x} \in \mathcal{X}_+, T \in \mathbb{N}, s \in \mathbb{N},$$
$$\sum_{k=s}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(t=Nk+i)} = \mathbf{x}\right] \geq lT - B. \tag{26}$$

Denote the minimum nonzero conditional probability as

$$\pi_{\min} = \min_{1 \leq i \leq N, \pi(x_i | \mathbf{x}_{-i}) > 0} \pi(x_i | \mathbf{x}_{-i}).$$

The following lemma, which is needed to prove Proposition 5, gives an inequality between the number of visits of two sets of states in consecutive steps. Please refer to Figure 14 for an illustration of the two sets of states and the mapping defined in the lemma.

Lemma 6 *Given any integer $i \in [1, N]$, a set of states $\mathbb{X} \subseteq \mathcal{X}_+$, and a mapping $F : \mathbb{X} \to \mathcal{X}_+$ that corresponds to any possible state transition for a Gibbs step at variable X_i, that is, F is any mapping satisfying*

$$\mathbf{F}(\mathbf{x})_{-i} = \mathbf{x}_{-i} \text{ and } \pi(\mathbf{F}(\mathbf{x})_i | \mathbf{x}) > 0, \forall \mathbf{x} \in \mathbb{X}, \tag{27}$$

let $\mathbb{Y} = \cup_{\mathbf{x} \in \mathbb{X}} F(\mathbf{x})$. We have that, if the graph is fully connected, for any $s \geq 0$ and $T > 0$, the number of times any state in $\mathbb{Y}$ is visited in the set of all i'th step in sweeps $s, s+1, \ldots, s+T-1$, denoted as $C_i = \{t = kN + i : s \leq k \leq s+T-1\}$, is lower

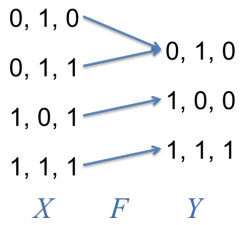


Figure 14: Example of the mapping defined in Lemma 6 with $i = 3$.

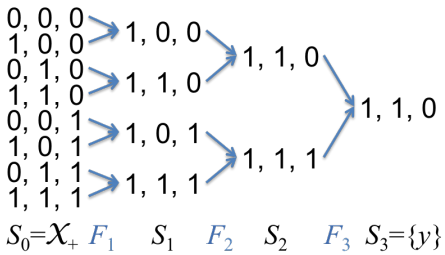


Figure 15: Example of the set of all paths from $\mathbf{x} \in \mathbb{X}_+$ to $\mathbf{y} = (1, 1, 0)$. Variables are updated in the order of $i = 1, 2, 3$. In this example, all the conditional distributions are positive and $\mathbf{y}$ can be reached from any state in $N = 3$ steps. Therefore, $t^* = 3$.

bounded by a function of the number of times any state in $\mathbb{X}$ is visited in the previous steps $C_{i-1} = \{t = kN + i - 1 : s \le k \le s + T - 1\}$ as:

$$\sum_{t \in C_i} \mathbb{I}\left[\mathbf{X}^{(t)} \in \mathbb{Y}\right] \ge \pi_{min} \sum_{t \in C_{i-1}} \mathbb{I}\left[\mathbf{X}^{(t)} \in \mathbb{X}\right] - |\mathbb{Y}|. \tag{28}$$

Proof As a complement to Condition 27, we can define F^{-1} as the inverse mapping from $\mathbb{Y}$ to subsets of $\mathbb{X}$ so that for any $\mathbf{y} \in \mathbb{Y}$, $\mathbf{x} \in F^{-1}(\mathbf{y})$, we have $\mathbf{x}_{-i} = \mathbf{y}_{-i}$, and $\cup_{\mathbf{y} \in \mathbb{Y}} F^{-1}(\mathbf{y}) = \mathbb{X}$. It's easy to observe that

$$\mathbb{I}\left[\mathbf{X}_{-i}^{(t)} = \mathbf{y}_{-i}\right] \ge \mathbb{I}\left[\mathbf{X}^{(t)} \in F^{-1}(\mathbf{y})\right], \forall t, \mathbf{y} \in \mathbb{Y}. \tag{29}$$

At every step of the herded Gibbs algorithm, only one weight is updated. Consider the sequence of steps when a weight $w_{i,\mathbf{x}_{N(i)}}$ is updated, we denote the segment of that sequence in sweeps $[s, s+T-1]$ as $C_i(\mathbf{x}_{N(i)})$. By definition of the herded Gibbs algorithm, $C_i(\mathbf{x}_{N(i)}) = \{t : t \in C_i, \mathbf{X}_{N(i)}^{(t-1)} = \mathbf{x}_{N(i)}\} = \{t : t \in C_i, \mathbf{X}_{N(i)}^{(t)} = \mathbf{x}_{N(i)}\} \subseteq C_i$.

Because the graph is fully connected, $N(i) = -i$, the full conditional state $\mathbf{X}_{-i}^{(t)}$ with $t \in C_i(\mathbf{x}_{N(i)})$ is uniquely determined. Since the value $X_i^{(t)}$ is determined by $w_{i,\mathbf{x}_{-i}}$ for any $t \in C_i(\mathbf{x}_{-i})$, we can apply Lemma 1 and get that for any $\mathbf{y} \in \mathbb{Y}$,

$$\sum_{t \in C_i} \mathbb{I}\left[\mathbf{X}^{(t)} = \mathbf{y}\right] = \sum_{t \in C_i(\mathbf{y}_{-i})} \mathbb{I}\left[X_i^{(t)} = y_i\right] \ge \pi(y_i|\mathbf{y}_{-i})|C_i(\mathbf{y}_{-i})| - 1 \ge \pi_{\min}|C_i(\mathbf{y}_{-i})| - 1. \tag{30}$$

Since the variables $\mathbf{X}_{-i}$ are not changed at steps in C_i, combining with Equation 29 we can show

$$|C_i(\mathbf{y}_{-i})| = \sum_{t \in C_i} \mathbb{I}\left[\mathbf{X}_{-i}^{(t)} = \mathbf{y}_{-i}\right] = \sum_{t \in C_{i-1}} \mathbb{I}\left[\mathbf{X}_{-i}^{(t)} = \mathbf{y}_{-i}\right] \ge \sum_{t \in C_{i-1}} \mathbb{I}\left[\mathbf{X}^{(t)} \in F^{-1}(\mathbf{y})\right]. \tag{31}$$

Combining the fact that $\cup_{\mathbf{y} \in \mathbb{Y}} F^{-1}(\mathbf{y}) = \mathbb{X}$ and summing up both sides of Equation 30 over $\mathbb{Y}$ proves the lemma:

$$\sum_{t \in C_i} \mathbb{I}\left[\mathbf{X}^{(t)} \in \mathbb{Y}\right] \ge \sum_{\mathbf{y} \in \mathbb{Y}} \left(\pi_{\min} \sum_{t \in C_{i-1}} \mathbb{I}\left[\mathbf{X}^{(t)} \in F^{-1}(\mathbf{y})\right] - 1\right) \ge \pi_{\min} \sum_{t \in C_{i-1}} \mathbb{I}\left[\mathbf{X}^{(t)} \in \mathbb{X}\right] - |\mathbb{Y}|. \tag{32}$$

■

Remark 7 *A fully connected graph is a necessary condition for the application of Lemma 1 in Equation 30. If a graph is not fully connected ($N(i) \neq -i$), a weight $w_{i,\mathbf{x}_{N(i)}}$ is associated with a partial conditional state and may be shared by multiple full conditional states. In that case $C_i(\mathbf{x}_{N(i)}) = \{t : t \in C_i, \mathbf{X}_{N(i)}^{(t)} = \mathbf{x}_{N(i)}\}$, and we can no longer use Lemma 1 to get a lower bound for the number of visits to a particular **joint** state, not to mention a linear visiting rate in Proposition 5.*

Now let us prove Proposition 5 by iteratively applying Lemma 6.

Proof [Proof of Proposition 5] Because the corresponding Gibbs sampler is irreducible and any Gibbs sampler is aperiodic, there exists a constant $t^* > 0$ such that for any state $\mathbf{y} \in \mathcal{X}_+$, and any step i in a sweep, we can find a path of length t^* from any state $\mathbf{x} \in \mathcal{X}_+$ with a positive transition probability, $Path(\mathbf{x}) = (\mathbf{x} = \mathbf{x}(0), \mathbf{x}(1), \ldots, \mathbf{x}(t^*) = \mathbf{y})$, where each step of the path follows the Gibbs updating scheme. For a strictly positive distribution, the minimum value of t^* is N. We show an example of the paths in a graph with 3 variables and a strictly positive distribution in Figure 15.

Denote $\tau^* = \lceil t^*/N \rceil$ and the j^{th} element of the path $Path(\mathbf{x})$ as $Path(\mathbf{x}, j)$. We can define $t^* + 1$ subsets $S_j \subseteq \mathcal{X}_+, 0 \le j \le t^*$ as the union of all the j^{th} states in the path from any state in $\mathcal{X}_+$:

$$S_j = \cup_{\mathbf{x} \in \mathcal{X}_+} Path(\mathbf{x}, j).$$

By definition of these paths, we know $S_0 = \mathcal{X}_+$ and $S_{t^*} = \{\mathbf{y}\}$, and there exits an integer $i(j)$ and a mapping $F_j : S_{j-1} \to S_j, \forall j$ that satisfy the condition in Lemma 6 ($i(j)$ is the index of the variable to be updated, and the mapping is defined by the transition paths). Also notice that any state in S_j can be different from $\mathbf{y}$ by at most $\min\{N, t^* - j\}$ variables, and therefore $|S_j| \le 2^{\min\{N, t^*-j\}}$.

Let us apply Lemma 6 recursively from $j = t^*$ to 1 as

$$\begin{aligned}
&\sum_{k=s}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(Nk+i)} = \mathbf{y}\right] \geq \sum_{k=s+\tau^*}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(Nk+i)} = \mathbf{y}\right] \\
&= \sum_{k=s+\tau^*}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(Nk+i)} \in S_{t^*}\right] && \text{(b.c. } S_{t^*} = \{\mathbf{y}\}) \\
&\geq \pi_{\min} \sum_{k=s+\tau^*}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(Nk+i-1)} \in S_{t^*-1}\right] - |S_{t^*}| && \text{(Lemma 6, } \mathbb{X} = S_{t^*-1}, \mathbb{Y} = S_{t^*}) \\
&\geq \pi_{\min}^2 \sum_{k=s+\tau^*}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(Nk+i-2)} \in S_{t^*-2}\right] - \pi_{\min}|S_{t^*-1}| - |S_{t^*}| && \text{(Lemma 6, } \mathbb{X} = S_{t^*-2}, \mathbb{Y} = S_{t^*-1}) \\
&\geq \cdots \\
&\geq \pi_{\min}^{t^*} \sum_{k=s+\tau^*}^{s+T-1} \mathbb{I}\left[\mathbf{X}^{(Nk+i-t^*)} \in S_0 = \mathcal{X}_+\right] - \sum_{j=0}^{t^*-1} \pi_{\min}^j |S_{t^*-j}| && \text{(Lemma 6, } \mathbb{X} = S_0, \mathbb{Y} = S_1) \\
&\geq \pi_{\min}^{t^*}(T - \tau^*) - \sum_{j=0}^{t^*-1} \pi_{\min}^j 2^{\min\{N,j\}}. && \text{(b.c. } \mathbb{I}\left[\mathbf{X}^{(t)} \in \mathcal{X}_+\right] = 1)
\end{aligned} \tag{33}$$

The proof is concluded by choosing the constants

$$l = \pi_{\min}^{t^*}, \quad B = \tau^* \pi_{\min}^{t^*} + \sum_{j=0}^{t^*-1} \pi_{\min}^j 2^{\min\{N,j\}}. \tag{34}$$

■

Remark 8 *For a strictly positive distribution, the constants reduce to*

$$l = \pi_{min}^N, \quad B = \pi_{min}^N + \sum_{j=0}^{N-1} (2\pi_{min})^j = \pi_{min}^N + \frac{1 - (2\pi_{min})^N}{1 - 2\pi_{min}}.$$

B.3 Herded Gibbs's Transition Kernel $\tilde{\mathcal{T}}_T^{(\tau)}$ is an Approximation to $\mathcal{T}$

The following proposition shows that $\tilde{\mathcal{T}}_T^{(\tau)}$ is an approximation to the regular Gibbs sampler's transition kernel $\mathcal{T}$ with an error of $O(1/T)$.

Proposition 9 *For a fully connected graph, if the herded Gibbs has a fixed scanning order and the corresponding Gibbs sampling Markov chain is irreducible, then for any $\tau \geq 0$, $T \geq T^* := \frac{2B}{l}$ where l and B are the constants in Proposition 5, the following inequality holds:*

$$\|\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T}\|_\infty \leq \frac{4N}{lT}. \tag{35}$$

Proof

When $\mathbf{x} \notin \mathcal{X}_+$, we have the equality $\tilde{\mathcal{T}}_{T,i}^{(\tau)}(\mathbf{x}, \mathbf{y}) = \mathcal{T}_i(\mathbf{x}, \mathbf{y})$ by definition. When $\mathbf{x} \in \mathcal{X}_+$ but $\mathbf{y} \notin \mathcal{X}_+$, then $N_{\text{den}} = 0$ (see the notation of $\tilde{\mathcal{T}}_T^{(\tau)}$ for definition of N_{den}) as $\mathbf{y}$ will never be visited and thus $\tilde{\mathcal{T}}_{T,i}^{(\tau)}(\mathbf{x}, \mathbf{y}) = 0 = \mathcal{T}_i(\mathbf{x}, \mathbf{y})$ also holds. Let us consider the entries in $\tilde{\mathcal{T}}_{T,i}^{(\tau)}(\mathbf{x}, \mathbf{y})$ with $\mathbf{x}, \mathbf{y} \in \mathcal{X}_+$ in the following.

Because $\mathbf{X}_{-i}$ is not updated at i^{th} step of every sweep, we can replace $i - 1$ in the definition of N_{den} by i and get

$$N_{\text{den}} = \sum_{k=\tau}^{\tau+T-1} \mathbb{I}(\mathbf{X}_{-i}^{(kN+i)} = \mathbf{x}_{-i}).$$

Notice that the set of times $\{t = kN + i : \tau \leq k \leq \tau + T - 1, \mathbf{X}_{-i}^t = \mathbf{x}_{-i})\}$, whose size is N_{den}, is a consecutive set of times when $w_{i,\mathbf{x}_{-i}}$ is updated. By Lemma 1, we obtain a bound for the numerator

$$\begin{aligned}
&N_{\text{num}} \in [N_{\text{den}}\pi(X_i = y_i|\mathbf{x}_{-i}) - 1, N_{\text{den}}\pi(X_i = y_i|\mathbf{x}_{-i}) + 1] \Leftrightarrow \\
&|P_{T,i}^{(\tau)}(X_i = y_i|\mathbf{x}_{-i}) - \pi(X_i = y_i|\mathbf{x}_{-i})| = |\frac{N_{\text{num}}}{N_{\text{den}}} - \pi(X_i = y_i|\mathbf{x}_{-i})| \leq \frac{1}{N_{\text{den}}}.
\end{aligned} \tag{36}$$

Also by Proposition 5, we know every state in $\mathcal{X}_+$ is visited at a linear rate, there hence exist constants $l > 0$ and $B > 0$, such that the number of occurrence of any conditioning state $\mathbf{x}_{-i}$, N_{den}, is bounded by

$$N_{\text{den}} \geq \sum_{k=\tau}^{\tau+T-1} \mathbb{I}(\mathbf{X}^{(kN+i)} = \mathbf{x}) \geq lT - B \geq \frac{l}{2}T, \quad \forall T \geq \frac{2B}{l}. \tag{37}$$

Combining equations (36) and (37), we obtain

$$|P_{T,i}^{(\tau)}(X_i = y_i|\mathbf{x}_{-i}) - \pi(X_i = y_i|\mathbf{x}_{-i})| \leq \frac{2}{lT}, \quad \forall T \geq \frac{2B}{l}. \tag{38}$$

Since the matrix $\tilde{\mathcal{T}}_{T,i}^{(\tau)}$ and $\mathcal{T}_i$ differ only at those elements where $\mathbf{x}_{-i} = \mathbf{y}_{-i}$, we can bound the L_1 induced norm of the transposed matrix of their difference by

$$\begin{aligned}
\|(\tilde{\mathcal{T}}_{T,i}^{(\tau)} - \mathcal{T}_i)^T\|_1 &= \max_{\mathbf{x}} \sum_{\mathbf{y}} |\tilde{\mathcal{T}}_{T,i}^{(\tau)}(\mathbf{x}, \mathbf{y}) - \mathcal{T}_i(\mathbf{x}, \mathbf{y})| \\
&= \max_{\mathbf{x}} \sum_{y_i} |P_{T,i}^{(\tau)}(X_i = y_i|\mathbf{x}_{-i}) - \pi(X_i = y_i|\mathbf{x}_{-i})| \\
&\leq \frac{4}{lT}, \quad \forall T \geq \frac{2B}{l}.
\end{aligned} \tag{39}$$

Observing that both $\tilde{\mathcal{T}}_T^{(\tau)}$ and $\mathcal{T}$ are multiplications of N component transition matrices, and the transition matrices, $\tilde{\mathcal{T}}_T^{(\tau)}$ and $\mathcal{T}_i$, have a unit L_1 induced norm as:

$$\|(\tilde{\mathcal{T}}_{T,i}^{(\tau)})^T\|_1 = \max_{\mathbf{x}} \sum_{\mathbf{y}} |\tilde{\mathcal{T}}_{T,i}^{(\tau)}(\mathbf{x}, \mathbf{y})| = \max_{\mathbf{x}} \sum_{\mathbf{y}} P_{T,i}^{(\tau)}(X_i = y_i|\mathbf{x}_{-i}) = 1, \tag{40}$$

$$\|(\mathcal{T}_i)^T\|_1 = \max_{\mathbf{x}} \sum_{\mathbf{y}} |\mathcal{T}_i(\mathbf{x}, \mathbf{y})| = \max_{\mathbf{x}} \sum_{\mathbf{y}} P(X_i = y_i|\mathbf{x}_{-i}) = 1, \tag{41}$$

we can further bound the L_1 norm of the difference, $(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})^T$. Let $P \in \mathbb{R}^N$ be any vector with nonzero norm. Using the triangular inequality, the difference of the resulting vectors after applying $\tilde{\mathcal{T}}_T^{(\tau)}$ and $\mathcal{T}$ is bounded by

$$\begin{aligned}\|P(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})\|_1 =& \|P\tilde{\mathcal{T}}_{T,1}^{(\tau)} \dots \tilde{\mathcal{T}}_{T,N}^{(\tau)} - P\mathcal{T} \dots \mathcal{T}_N\|_1 \\ \leq& \|P(\tilde{\mathcal{T}}_{T,1}^{(\tau)} - \mathcal{T}_1)\tilde{\mathcal{T}}_{T,2}^{(\tau)} \dots \tilde{\mathcal{T}}_{T,N}^{(\tau)}\|_1 + \\ & \|P\mathcal{T}_1(\tilde{\mathcal{T}}_{T,2}^{(\tau)} - \mathcal{T}_2)\tilde{\mathcal{T}}_{T,3}^{(\tau)} \dots \tilde{\mathcal{T}}_{T,N}^{(\tau)}\|_1 + \\ & \dots \\ & \|P\mathcal{T}_1 \dots \mathcal{T}_{N-1}(\tilde{\mathcal{T}}_{T,N}^{(\tau)} - \mathcal{T}_N)\|_1, \end{aligned} \tag{42}$$

where the i'th term is

$$\begin{aligned}\|P\mathcal{T}_1 \dots \mathcal{T}_{i-1}(\tilde{\mathcal{T}}_{T,i}^{(\tau)} - \mathcal{T}_i)\tilde{\mathcal{T}}_{T,i+1}^{(\tau)} \dots \tilde{\mathcal{T}}_{T,N}^{(\tau)}\|_1 &\leq \|P\mathcal{T}_1 \dots \mathcal{T}_{i-1}(\tilde{\mathcal{T}}_{T,i}^{(\tau)} - \mathcal{T}_i)\|_1 && \text{(Unit } L_1 \text{ norm, Eqn. 40)} \\ &\leq \|P\mathcal{T}_1 \dots \mathcal{T}_{i-1}\|_1 \frac{4}{lT} && \text{(Eqn. 39)} \\ &\leq \|P\|_1 \frac{4}{lT}. && \text{(Unit } L_1 \text{ norm, Eqn. 41)} \end{aligned} \tag{43}$$

Consequently, we get the L_1 induced norm of $(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})^T$ as

$$\|(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})^T\| = \max_P \frac{\|P(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})\|_1}{\|P\|_1} \leq \frac{4N}{lT}, \quad \forall T \geq \frac{2B}{l}. \tag{44}$$

■

B.4 Proof of Theorem 3

When we initialize the herded Gibbs and regular Gibbs with the same distribution (see Figure 12), since the transition kernel of herded Gibbs is an approximation to regular Gibbs and the distribution of regular Gibbs converges to the invariant distribution, we expect that herded Gibbs also approaches the invariant distribution.

Proof [Proof of Theorem 3] Construct an auxiliary regular Gibbs sampling Markov chain initialized with $\pi^{(0)}(\mathbf{X}) = P_T^{(\tau)}(\mathbf{X})$ and the same scanning order as herded Gibbs. As $\eta < 1$, the Gibbs Markov chain has uniform geometric convergence rate as shown in Equation (23).

Also, the Gibbs Markov chain must be irreducible due to $\eta < 1$ and therefore Proposition 9 applies here. We can bound the distance between the distributions of herded Gibbs after one sweep of all variables, $P_T^{(\tau+1)}$, and the distribution after one sweep of regular Gibbs sampling, $\pi^{(1)}$ by

$$\begin{aligned} & d_v(P_T^{(\tau+1)} - \pi^{(1)}) = d_v(\pi^{(0)}(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})) = \frac{1}{2}\|\pi^{(0)}(\tilde{\mathcal{T}}_T^{(\tau)} - \mathcal{T})\|_1 \\ & \leq \frac{2N}{lT}\|\pi^{(0)}\|_1 = \frac{2N}{lT}, \quad \forall T \geq T^*, \tau \geq 0. \end{aligned} \tag{45}$$

Now we study the change of discrepancy between $P_T^{(\tau)}$ and π as a function as τ. Applying the triangle inequality of d_v:

$$\begin{aligned} & d_v(P_T^{(\tau+1)} - \pi) = d_v(P_T^{(\tau+1)} - \pi^{(1)} + \pi^{(1)} - \pi) \leq d_v(P_T^{(\tau+1)} - \pi^{(1)}) + d_v(\pi^{(1)} - \pi) \\ & \leq \frac{2N}{lT} + \eta d_v(P_T^{(\tau)} - \pi), \quad \forall T \geq T^*, \tau \geq 0. \end{aligned} \tag{46}$$

The last inequality follows Equations (23) and (45). When the sample distribution is outside a neighborhood of π, $\mathcal{B}_{\epsilon_1}(\pi)$, with $\epsilon_1 = \frac{4N}{(1-\eta)lT}$, i.e.

$$d_v(P_T^{(\tau)} - \pi) \geq \frac{4N}{(1-\eta)lT}, \tag{47}$$

we get a geometric convergence rate toward the invariant distribution by combining the two equations above:

$$d_v(P_T^{(\tau+1)} - \pi) \leq \frac{1-\eta}{2} d_v(P_T^{(\tau)} - \pi) + \eta d_v(P_T^{(\tau)} - \pi) = \frac{1+\eta}{2} d_v(P_T^{(\tau)} - \pi). \tag{48}$$

So starting from $\tau = 0$, we have a burn-in period for herded Gibbs to enter $\mathcal{B}_{\epsilon_1}(\pi)$ in a finite number of rounds. Denote the first time it enters the neighborhood by τ'. According to the geometric convergence rate in Equations 48 and $d_v(P_T^{(0)} - \pi) \leq 1$

$$\tau' \leq \left\lceil \log_{\frac{1+\eta}{2}}\left(\frac{\epsilon_1}{d_v(P_T^{(0)} - \pi)}\right) \right\rceil \leq \left\lceil \log_{\frac{1+\eta}{2}}(\epsilon_1) \right\rceil = \lceil \tau^*(T) \rceil. \tag{49}$$

After that burn-in period, the herded Gibbs sampler will stay within a smaller neighborhood, $\mathcal{B}_{\epsilon_2}(\pi)$, with $\epsilon_2 = \frac{1+\eta}{1-\eta}\frac{2N}{lT}$, i.e.

$$d_v(P_T^{(\tau)} - \pi) \leq \frac{1+\eta}{1-\eta}\frac{2N}{lT}, \quad \forall \tau > \tau'. \tag{50}$$

This is proved by induction:

1. Equation (50) holds at $\tau = \tau' + 1$. This is because $P_T^{(\tau')} \in \mathcal{B}_{\epsilon_1}(\pi)$ and following Eqn. 46 we get
$$d_v(P_T^{(\tau'+1)} - \pi) \leq \frac{2N}{lT} + \eta\epsilon_1 = \epsilon_2. \tag{51}$$
2. For any $\tau \geq \tau' + 2$, assume $P_T^{(\tau-1)} \in \mathcal{B}_{\epsilon_2}(\pi)$. Since $\epsilon_2 < \epsilon_1$, $P_T^{(\tau-1)}$ is also in the ball $\mathcal{B}_{\epsilon_1}(\pi)$. We can apply the same computation as when $\tau = \tau' + 1$ to prove $d_v(P_T^{(\tau)} - \pi) \leq \epsilon_2$. So inequality (50) is always satisfied by induction.

Consequently, Theorem 3 is proved when combining (50) with the inequality $\tau' \leq \lceil \tau^*(T) \rceil$ in Equation(49). ■

Remark 10 *Similarly to the regular Gibbs sampler, the herded Gibbs sampler also has a burn-in period with geometric convergence rate. After that, the distribution discrepancy is in the order of* $O(1/T)$*, which is faster than the regular Gibbs sampler. Notice that the length of the burn-in period depends on* T*, specifically as a function of* $\log(T)$*.*

Remark 11 *Irrationality is not required to prove the convergence on a fully-connected graph.*

B.5 Proof of Corollary 4

Proof Since $\tau^*(T)$ is a monotonically increasing function of T, for any $T \geq T^* + \tau^*(T^*)$, we can find a number t so that

$$T = t + \tau^*(t), t \geq T^*.$$

Partition the sample sequence $S_{0,T} = \{\mathbf{X}^{(kN)} : 0 \leq k < T\}$ into two parts: the burn-in period $S_{0,\tau^*(t)}$ and the stable period $S_{\tau^*(t),T}$. The discrepancy in the burn-in period is bounded by 1 and according to Theorem 3, the discrepancy in the stable period is bounded by

$$d_v(\tilde{P}(S_{t,T}) - \pi) \leq \frac{\lambda}{t}.$$

Hence, the discrepancy of the whole set $S_{0,T}$ is bounded by

$$\begin{aligned}
&d_v(\tilde{P}(S_{0,T}) - \pi) = d_v\left(\frac{\tau^*(t)}{T}\tilde{P}(S_{0,\tau^*(t)}) + \frac{t}{T}\tilde{P}(S_{\tau^*(t),T}) - \pi\right) \\
&\leq d_v\left(\frac{\tau^*(t)}{T}(\tilde{P}(S_{0,\tau^*(t)}) - \pi\right) + d_v\left(\frac{t}{T}(\tilde{P}(S_{\tau^*(t),T}) - \pi\right) \\
&\leq \frac{\tau^*(t)}{T}d_v(\tilde{P}(S_{0,\tau^*(t)}) - \pi) + \frac{t}{T}d_v(\tilde{P}(S_{\tau^*(t),T}) - \pi) \\
&\leq \frac{\tau^*(t)}{T} \cdot 1 + \frac{t}{T}\frac{\lambda}{t} \leq \frac{\tau^*(T) + \lambda}{T}. \qquad (52)
\end{aligned}$$

■

References

Pulp Fiction - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Pulp_Fiction, 2013.

D. H. Ackley, G. Hinton, and T. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, 2003.

F. Bach, S. Lacoste-Julien, and G. Obozinski. On the equivalence between herding and conditional gradient algorithms. In *International Conference on Machine Learning*, 2012.

L. Bornn, P. E. Jacob, P. Del Moral, and A. Doucet. An adaptive interacting wang–landau algorithm for automatic density exploration. *Journal of Computational and Graphical Statistics*, 22(3):749–773, 2013.

P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*, volume 31. Springer, 1999.

P. Carbonetto, J. Kisynski, N. de Freitas, and D. Poole. Nonparametric Bayesian logic. In *Uncertainty in Artificial Intelligence*, pages 85–93, 2005.

S. Chen, J. Dick, and A. B. Owen. Consistency of Markov chain quasi-Monte Carlo on continuous state spaces. *The Annals of Statistics*, 39(2):673–701, 04 2011. doi: 10.1214/10-AOS831. URL http://dx.doi.org/10.1214/10-AOS831.

S. Chen, M. Matsumoto, T. Nishimura, and A. Owen. New inputs and methods for Markov chain quasi-Monte Carlo. In L. Plaskota and H. Wozniakowski, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2010*, volume 23 of *Springer Proceedings in Mathematics & Statistics*, pages 313–327. Springer Berlin Heidelberg, 2012. URL http://dx.doi.org/10.1007/978-3-642-27440-4_15.

Y. Chen, M. Welling, and A.J. Smola. Supersamples from kernel-herding. In *Uncertainty in Artificial Intelligence*, pages 109–116, 2010.

A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, 2001.

A. Gelfand, Y. Chen, L. van der Maaten, and M. Welling. On herding and the perceptron cycling theorem. In *Advances in Neural Information Processing Systems*, pages 694–702, 2010.

N. D. Goodman, V. K. Mansinghka, D. M. Roy, K. Bonawitz, and J. B. Tenenbaum. Church: a language for generative models. *Uncertainty in Artificial Intelligence*, 2008.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

A. E. Holroyd and J. Propp. Rotor walks and Markov chains. *Algorithmic Probability and Combinatorics*, 520:105–126, 2010.

F. Huszar and D. Duvenaud. Optimally-weighted herding is Bayesian quadrature. In *Proceedings of the Twenty-Eighth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-12)*, pages 377–386, Corvallis, Oregon, 2012. AUAI Press.

T. Grenager J. R. Finkel and C. Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219885. URL http://dx.doi.org/10.3115/1219840.1219885.

J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, 2001.

D. J. Lunn, A. Thomas, N. Best, and D. Spiegelhalter. WinBUGS a Bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing*, 10(4):325–337, 2000.

B. Milch and S. Russell. General-purpose MCMC inference over relational structures. In *Uncertainty in Artificial Intelligence*, pages 349–358, 2006.

K. P. Murphy. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2012.

I. Murray and L. T. Elliott. Driving Markov chain Monte Carlo with a dependent random stream. *arXiv preprint arXiv:1204.3187*, 2012.

I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling. Fast collapsed Gibbs sampling for latent Dirichlet allocation. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 569–577, 2008.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, 2nd edition, 2004.

M. Welling. Herding dynamical weights to learn. In *International Conference on Machine Learning*, pages 1121–1128, 2009a.

M. Welling. Herding dynamic weights for partially observed random field models. In *Uncertainty in Artificial Intelligence*, pages 599–606, 2009b.

H. Weyl. Über die gleichverteilung von zahlen mod. eins. *Mathematische Annalen*, 77: 313–352, 1916. ISSN 0025-5831. doi: 10.1007/BF01475864. URL `http://dx.doi.org/10.1007/BF01475864`.

Journal of Machine Learning Research 17 (2016) 1-28 Submitted 7/13; Revised 5/15; Published 4/16

Complexity of Representation and Inference in Compositional Models with Part Sharing

Alan Yuille ALAN.YUILLE@JHU.EDU
Departments of Cognitive Science and Computer Science
Johns Hopkins University
Baltimore, MD 21218

Roozbeh Mottaghi ROOZBEHM@ALLENAI.ORG
Allen Institute for Artificial Intelligence
Seattle WA 98103, USA

Editors: Aaron Courville, Rob Fergus, and Christopher Manning

Abstract

This paper performs a complexity analysis of a class of serial and parallel compositional models of multiple objects and shows that they enable efficient representation and rapid inference. Compositional models are generative and represent objects in a hierarchically distributed manner in terms of parts and subparts, which are constructed recursively by part-subpart compositions. Parts are represented more coarsely at higher level of the hierarchy, so that the upper levels give coarse summary descriptions (e.g., there is a horse in the image) while the lower levels represents the details (e.g., the positions of the legs of the horse). This hierarchically distributed representation obeys the *executive summary* principle, meaning that a high level executive only requires a coarse summary description and can, if necessary, get more details by consulting lower level executives. The parts and subparts are organized in terms of hierarchical dictionaries which enables *part sharing* between different objects allowing efficient representation of many objects. The first main contribution of this paper is to show that compositional models can be mapped onto a parallel visual architecture similar to that used by bio-inspired visual models such as deep convolutional networks but more explicit in terms of representation, hence enabling part detection as well as object detection, and suitable for complexity analysis. Inference algorithms can be run on this architecture to exploit the gains caused by part sharing and executive summary. Effectively, this compositional architecture enables us to perform exact inference simultaneously over a large class of generative models of objects. The second contribution is an analysis of the complexity of compositional models in terms of computation time (for serial computers) and numbers of nodes (e.g., "neurons") for parallel computers. In particular, we compute the complexity gains by part sharing and executive summary and their dependence on how the dictionary scales with the level of the hierarchy. We explore three regimes of scaling behavior where the dictionary size (i) increases exponentially with the level of the hierarchy, (ii) is determined by an unsupervised compositional learning algorithm applied to real data, (iii) decreases exponentially with scale. This analysis shows that in some regimes the use of shared parts enables algorithms which can perform inference in time linear in the number of levels for an exponential number of objects. In other regimes part sharing has little advantage for serial computers but can enable linear processing on parallel computers.

Keywords: compositional models, object detection, hierarchical architectures, part sharing

1. Introduction

A fundamental problem of vision is how to deal with the enormous complexity of images and visual scenes. The total number of possible images is astronomically large Kersten (1987). The number of objects is also huge and has been estimated at around 30,000 Biederman (1987). We observe also that humans do not simply detect objects but also *parse* them into their parts and subparts, which adds another level of complexity. How can a biological, or artificial, vision system deal with this complexity? For example, considering the enormous input space of images and output space of objects, how can humans interpret images in less than 150 msec Thorpe et al. (1996)?

There are three main issues involved. Firstly, how can a visual system be designed so that it can efficiently *represent* large numbers of objects, including their parts and subparts? Secondly, how can a visual system rapidly *infer* which object, or objects, are present in an input image and perform related tasks such as estimating the positions of their parts and subparts? And, thirdly, how can this representation be *learnt* in an unsupervised, or weakly supervised fashion? In summary, what visual *architectures* enable us to overcome these three issues?

Studies of mammalian visual systems offer some suggestions for how these complexity issues can be addressed. These visual systems are organized hierarchically with the lower levels (e.g., in areas V1 and V2) tuned to small image features while the higher levels (i.e. in area IT) are tuned to objects. This leads to a class of bio-inspired visual architectures, of varying degrees of biological plausibility, which represent the visual world in terms of hierarchies of features which are shared between multiple objects Fukushima (1988); Riesenhuber and Poggio (1999); Hinton et al. (2006); Serre et al. (2007); Adams and Williams (2003); Poon and Domingos (2010); Zeiler et al. (2010); LeCun and Bengio (1995); Borenstein and Ullman (2002); Kokkinos and Yuille (2011); Krizhevsky et al. (2012). This sharing of hierarchical features suggest ways to deal with the complexity issues but, to the best of our knowledge, there have been no complexity studies of these classes of architectures. We note that the purpose of this paper is not to attempt to model the known structure of mammalian visual systems. Instead, own goal of this work is to see whether it is possible to derive properties of mammalian visual systems from first principles by developing a visual architecture that addresses the complexity problems of vision.

In this paper, we address the complexity issues from the perspective of compositional models Geman et al. (2002) by extending and analyzing a specific class of models Zhu et al. (2008, 2010). Compositional models are cousins of the bio-inspired hierarchical models and we briefly discuss their similarities and differences in section (6). The key idea of compositionality is to represent objects by recursive composition from parts and subparts which enables: (i) part sharing between different objects, and (ii) hierarchically distributed representations of objects where the positions of parts are represented more coarsely than the position of subparts, which we call the executive-summary principle. This gives rise to natural learning and inference algorithms which proceed from sub-parts to parts to objects (e.g., inference is efficient because a leg detector can be used for detecting the legs of cows,

horses, and yaks). The explicitness of the object representations helps quantify the efficiency of part-sharing, and executive summary, and make mathematical analysis possible. They also enable us to parse the objects into their parts and subparts, instead of only detecting objects. The compositional models we study are based on prior work Zhu et al. (2008, 2010) but we make several technical modifications. These include re-formulating the models so that the positions of the parts are restricted to lie on a set of discrete lattices (coarser lattices for high-level parts). This enables a parallel implementation of compositional models which helps clarify the similarities, and differences, to bio-inspired models. For completeness, we briefly summarize how compositional models can be learnt in an unsupervised manner Zhu et al. (2008, 2010) and discuss how this relates to the memorization algorithms of Valiant (2000). But we emphasize that this paper does not directly address learning but instead explores the consequence of the representations which were learnt. We note that previous work has addressed the complexity of visual search and attention Blanchard and Geman (2005), Tsotsos (2011).

Our analysis assumes that objects are represented by hierarchical graphical probability models which are composed from more elementary models by *part-subpart compositions*. An object – a graphical model with $\mathcal{H}$ levels – is defined as a composition of r parts which are graphical models with $\mathcal{H}-1$ levels. These parts are defined recursively in terms of subparts which are represented by graphical models of increasingly lower levels. It is convenient to specify these compositional models in terms of a set of dictionaries $\{\mathcal{M}_h : h = 0, ..., \mathcal{H}\}$ where the level-h parts in dictionary $\mathcal{M}_h$ are composed in terms of r level-$(h-1)$ parts in dictionary $\mathcal{M}_{h-1}$. The highest level dictionaries $\mathcal{M}_{\mathcal{H}}$ represent the set of all objects. The lowest level dictionaries $\mathcal{M}_0$ represent the elementary features that can be measured from the input image. Part-subpart composition enables us to construct a very large number of objects by different compositions of elements from the lowest-level dictionary (like building an object from a lego kit). It enables us to perform *part-sharing* during learning and inference, which can lead to enormous reductions in complexity, as our mathematical analysis will show. We stress that the parts and subparts are deformable (i.e. non-rigid).

There are three factors which enable computational efficiency. The first is *part-sharing*, as described above, which means that we only need to perform inference on the dictionary elements. The second is the *executive-summary principle*. This principle allows us to represent the state of a part coarsely because we are also representing the state of its subparts (e.g., a top level executive of a business company will only want to know that "there is a horse in the field" and will not care about the exact positions of its legs, which will be known by the lower level executives). For example, consider a letter T which is composed of a horizontal and vertical bar. If the positions of these two bars are specified precisely, then we only need to represent the position of the letter T more crudely (it is sufficient for it to be "bound" to the two bars, and for their positions to be represented separately). The third factor is *parallelism* which arises because the part dictionaries can be implemented in parallel. This is roughly similar to having a set of non-linear receptive fields for each dictionary element. This enables extremely rapid inference at the cost of a larger, but parallel, graphical model. The inference proceeds in a single bottom-up and top-down pass, where the bottom-up pass rapidly estimates an executive summary which enables the the top-down pass to estimate the lower-level details.

The compositional section (2) introduces the key ideas of compositional models and section (3) describes how they are used to generate images. Section (4) describes the compositional architecture and the inference algorithms for serial and parallel implementations. Section (5) performs a complexity analysis and shows potential exponential gains by using compositional models. Section (6) discusses the relations between compositional models and bio-inspired hierarchical models, briefly discusses how compositional models can be learnt in an unsupervised manner, discusses robust variants of compositional models and summaries an analysis of robustness (which is presented in the appendix).

2. Compositional Models

Compositional models are based on the idea that objects are built by compositions of parts which, in turn, are compositions of more elementary parts. Section (2.1) describes part-subparts compositions which are the basic building block of our approach. In section (2.2) we discuss how to build objects by recursively making part-subpart compositions. Section (2.3) describes the state variables and how they take values on a hierarchy of lattices. In section (2.4) we discuss hierarchical dictionaries and part sharing. In section (2.5) we specify the probability distributions for objects.

2.1 Compositional Part-Subparts

We formulate part-subpart compositions by probabilistic graphical models which specify how a part is composed of its subparts. This consists of a parent node ν with a *type* τ_ν and a *state variable* x_ν. The parent node represents the part and has r child nodes $Ch(\nu) = (\nu_1, ..., \nu_r)$, representing the subparts, which have types $\tau_{Ch(\nu)} = (\tau_{\nu_1}, ..., \tau_{\nu_r})$ and state variables $x_{Ch(\nu)} = (x_{\nu_1}, ..., x_{\nu_r})$ (r is fixed for all part-subpart compositions in this paper). For concreteness the state variables represent the position of the parts, or subparts in the image (unless otherwise specified, our analysis is directly extended to other cases where they represent other image properties, such as mini-epitomes or active patches Papandreou et al. (2014); Mao et al. (2014)). Similarly, the types represent geometric entities (e.g., the letters T, L or vertical/horizontal bars indicted by V, H). The type τ_ν of the parent node is specified by the types τ_{ν_i} of the child nodes and by the *spatial relations* λ_ν between the child nodes. Hence we express $\tau_\nu = (\tau_{\nu_1}, ..., \tau_{\nu_r}, \lambda_\nu)$. For example, in Figure (1) we represent the letters T and L by part-subparts where the parent has type "T" or "L". In both cases, there are two child nodes (i.e. $r = 2$) with types "vertical bar" and "horizontal bar" and the spatial relations are represented by λ_T and λ_L respectively. This is illustrated in Figure (1).

The spatial configuration of the part-subpart is represented by the state of the parent x_ν and the states of the children $x_{Ch(\nu)}$. The state of the parent x_ν provides an *executive summary* description of the part (e.g., there is a cow in the right side of the image) while the states of the children $x_{Ch(\nu)}$ gives the details (e.g., the positions of the head, torso, and legs of the cow). In this paper, we restrict the spatial position x_ν to take a discrete set of values, see section (2.3), which differs from earlier work where they were allowed to be continuous Zhu et al. (2008, 2010).

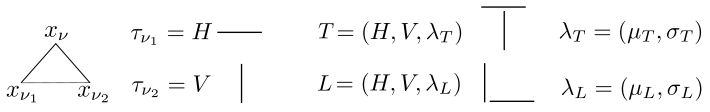


Figure 1: Compositional part-subpart models $T = (H, V, \lambda_T)$ and $L = (H, V, \lambda_L)$ for letters T and L are constructed from the same elementary components τ_H, τ_V, horizontal and vertical bar using different spatial relations $\lambda_T = (\mu_T, \sigma_T)$ and $\lambda_L = (\mu_L, \sigma_L)$, where μ denotes average displacement between the bars and σ^2 is the variance. The state x_ν of the parent node gives the summary position of the object, the *executive summary*, while the positions x_{ν_1} and x_{ν_2} of the components give the detailed positions of its components.

The probability distribution for the part-subpart model relates the states of the part and the subparts by:

$$P(x_{Ch(\nu)}|x_\nu; \tau_\nu) = \delta(x_\nu - f(x_{Ch(\nu)}))h(x_{Ch(\nu)}; \lambda_\nu). \quad (1)$$

Here $f(.)$ is a deterministic function, so the state of the parent node is determined uniquely by the state of the child nodes ($\delta(.)$ is a Dirac delta function if x_ν takes continuous values, and a Kronecker delta if x_ν takes discrete values). The function $h(.)$, which is parameterized by λ, specifies a distribution on the relative states of the child nodes. The distribution $P(x_{Ch(\nu)}|x_\nu; \tau_\nu)$ is local in the sense that $P(x_{Ch(\nu)}|x_\nu; \tau_\nu) = 0$, unless $|x_{\nu_i} - x_\nu|$ is smaller than a threshold for all $i = 1, .., r$. This requirement captures the intuition that subparts of a part are typically close together.

For example, in the simple example shown in Figure (1), the function $f(x_{\nu_1}, x_{\nu_2})$ specifies the average positions x_{ν_1} and x_{ν_2} of the horizontal and vertical bars, rounded off to the nearest position on the lattice. We set $h(.; \lambda)$ to be a discretized variant of the Gaussian distribution on the relative positions of the two bars, so $\lambda = (\mu, \sigma)$ where μ is the mean relative positions and σ is the covariance. Observe that the parts T and L have the same subparts – horizontal and vertical bars – but they have different spatial relations λ_T and λ_L (because the mean relative positions of the bars are different for T's and L's). Hence the two compositional models for the T and L have types $T = (H, V, \lambda_T)$ and $L = (H, V, \lambda_L)$.

2.2 Representing an Object Using a Hierarchical Graph

We model an object recursively from parts and subparts using part-subpart compositions. This is illustrated in Figure (2) where we combine T's and L's with other parts to form more complex objects. The basic idea is to build increasingly complex shapes by treating the parts as subparts of higher order parts.

More formally, an object is represented by a graph with nodes $\mathcal{V}$. State variables x_ν and types τ_ν are defined at the nodes $\nu \in \mathcal{V}$. The graph has a hierarchical structure with levels $h \in \{0, ..., \mathcal{H}\}$, where $\mathcal{V} = \bigcup_{h=0}^{\mathcal{H}} \mathcal{V}_h$. The edges in the graph specify the part-subpart compositions by connecting each node at level-h to r nodes at level-$h-1$. Each object has a single, root node, at level-$\mathcal{H}$. Any node $\nu \in \mathcal{V}_h$ (for $h > 0$) has r children nodes $Ch(\nu)$

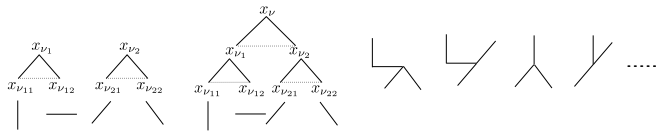


Figure 2: Object models are built recursively by part-subpart relations, Each object is represented by a hierarchical graph whose edges specify the part-subpart relationships. Each node ν has a type τ_ν and a state variable x_ν. Left Panel: Two part-subpart models, where the subparts are bars at four different angles (horizontal, vertical, forty-five degrees up, and forty-five degrees down). Center Panel: Combining two part-subpart models by composition to make a higher level model. Right Panel: Some examples of the higher-level models that can be obtained by different compositions.

in $\mathcal{V}_{h-1}$ indexed by $(\nu_1, ..., \nu_r)$ (corresponding to the subparts of the part). Hence there are $r^{\mathcal{H}-h}$ nodes at level-h (i.e. $|\mathcal{V}_h| = r^{\mathcal{H}-h}$).

The type of object is specified by the type $\tau_\mathcal{H}$ of the root node. Hence we use $\mathcal{V}^{\tau_\mathcal{H}}$ to refer to the nodes of the graph for object $\tau_\mathcal{H}$. The types of its descendant nodes are specified recursively by $\tau_{Ch(\nu)} = (\tau_{\nu_1}, ..., \tau_{\nu_r})$. The types of the lowest-level nodes, at $h = 0$ specify elementary features (e.g., horizontal and vertical bars). Hence we can think of an object as being constructed recursively by combining elementary structures with spatial relations (e.g., like building an object from a lego kit).

2.3 State Variables and Hierarchical Lattices

We require the state variables $\{x_\nu : \nu \in \mathcal{V}\}$ to satisfy the executive summary principle. To enforce this, we require that the positions of the subparts are specified at greater resolution than the positions of the parts. This is done by requiring that the state variables take values on a hierarchical lattice which we describe in this section. More precisely, the positions of the parts and subparts are specified by state variables x_ν which take values in a *set of lattices* $\{\mathcal{D}_h : h = 0, ..., \mathcal{H}\}$, so that a level-$h$ node, $\nu \in \mathcal{V}_h$, takes position $x_\nu \in \mathcal{D}_h$. The state variable of the root node gives the top-level executive summary of the object and lies on the lattice $\mathcal{D}_\mathcal{H}$. The state variables of the leaf nodes $\mathcal{V}_0$ of the graph take values on the image lattice $\mathcal{D}_0$.

The lattices are regularly spaced and the number of lattice points decreases by a factor of $q < 1$ for each level, so $|\mathcal{D}_h| = q^h|\mathcal{D}_0|$, see Figure (3)(left panel). This decrease in number of lattice points imposes the *executive summary principle*. The lattice spacing is designed so that parts do not overlap. At higher levels of the hierarchy the parts cover larger regions of the image and so the lattice spacing must be larger, and hence the number of lattice points smaller, to prevent overlapping. Note that previous work Zhu et al. (2008, 2010) was not formulated on lattices and used non-maximal suppression to achieve a similar effect.

We use the notation $\vec{x}_\nu$ to denote the state of node ν and all its descendants. This can be defined recursively by $\vec{x}_\nu = (x_\nu, x_{Ch(\nu)}, ...)$. Figure (3)(right panel) shows the object

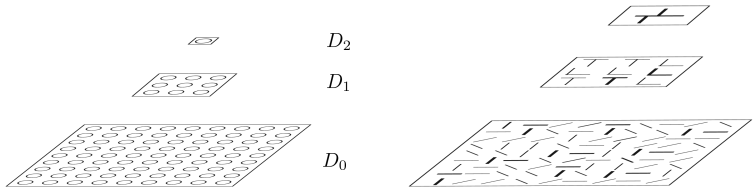


Figure 3: The hierarchical lattices. Left Panel: the size of the lattices decrease with scale by a factor $q = 1/9$ which helps enforce executive summary and prevent having multiple hypotheses which overlap too much. Right Panel: How an object is represented by a hierarchically distributed representation with coarse summary of position at the top level and more details at the lower levels.

represented hierarchically where the upper level encodes the summary and the lower-levels encode the details. By a slight abuse of notation, we specify the state vector of an object $\tau_{\mathcal{H}}$ by $\vec{x}_{\tau_{\mathcal{H}}}$. *In summary*, an object $\tau_{\mathcal{H}}$ is represented by a graph with nodes $\mathcal{V}^{\tau_{\mathcal{H}}}$ and state variables $\vec{x}_{\tau_{\mathcal{H}}}$.

2.4 Multiple Types of Objects, Shared Parts, and Hierarchical Dictionaries

Now suppose we have a large set of objects (each with $\mathcal{H}$ levels). We can represent each object separately by a hierarchical graph, as discussed in the last two subsections, but this is wasteful and cumbersome because it does not exploit the sharing of parts between different objects.

Instead, we represent objects and their parts by hierarchical dictionaries $\{\mathcal{M}_h : h = 0, ..., \mathcal{H}\}$. The top-level dictionary specifies the object types $\tau_{\mathcal{H}} \in \mathcal{M}_{\mathcal{H}}$. The level-$h$ dictionary specifies the types $\mathcal{M}_h$ of the parts at level h. The dictionary elements in $\mathcal{M}_h$ are built by compositions of r elements from the dictionary $\mathcal{M}_{h-1}$ at the previous level using part-subpart composition as described in section (2.1). An object is specified by the type of its root node $\tau_{\mathcal{H}} \in \mathcal{M}_{\mathcal{H}}$. It is composed of a set $Ch(\tau_{\mathcal{H}})$ of r parts from level-$\mathcal{H}-1$ with types plus spatial relations $\lambda_{\tau_{\mathcal{H}}}$ between them. Hence we express $\tau_{\mathcal{H}} = (\tau_{\mathcal{H},1}, ..., \tau_{\mathcal{H},r}; \lambda_{\tau_{\mathcal{H}}})$. In turn, these parts can be expressed in terms of lower-level subparts so that we can recursively specify the types of all parts of the object and the spatial relationships between them. More formally, any level-h part $\tau_h \in \mathcal{M}_h$ can be expressed as a composition of r level-$h-1$ parts $Ch(\tau_h) = \{\tau_{h,i} : i = 1, ..., r\}$, where $\tau_{h,i} \in \mathcal{M}_{h-1}$, and by spatial relations λ_{τ_h}. For example, in Figure (1) we have $\mathcal{M}_1 = \{T, L\}$ and $\mathcal{M}_0 = \{H, V\}$, where the compositions are given by $T = (H, V, \lambda_T)$ and $L = (H, V, \lambda_L)$ (here $r = 2$).

The hierarchical dictionaries enable us to make part-sharing explicit, which will enable us to analyze its effectiveness in section (5). Part-sharing is illustrated in Figure (4) by two models A and B which share level-1 part b. Hierarchical dictionaries are shown in Figure (5) where the dictionaries are $\mathcal{M}_2 = \{A, B\}, \mathcal{M}_1 = \{a, b, c\}$ and $\mathcal{M}_0 = \{\gamma, \epsilon, \delta, \}$. For example, the level-2 part A is composed from level-1 parts a and b (plus spatial relations, which are

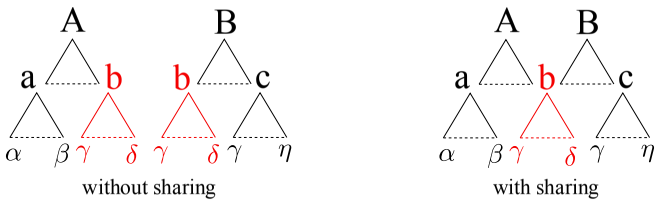


Figure 4: Part sharing. Two Level-2 parts A and B are composed from Lavel-1 parts a, b and c which, in turn, are composed from Level-0 parts $(\alpha, \beta), (\gamma, \delta)$ and (γ, ζ) respectively. Left Panel: the Level-2 parts are modeled separately and the fact that they share a Level-1 part is not exploited. Right Panel: In this paper we represent the part sharing explicitly and exploit it for efficient representation and inference.

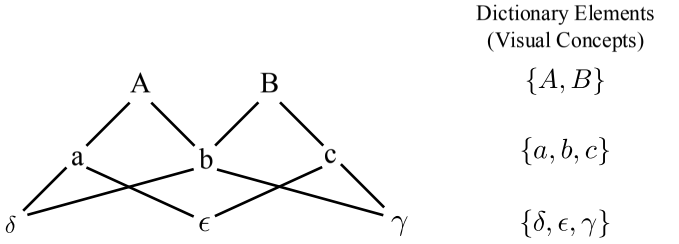


Figure 5: Representing objects in terms of hierarchical dictionaries with shared parts. There are two object models A, B at the top level, three parts a, b and c at the middle level, and three subparts γ, δ and ϵ at the bottom level. The Level-2 models A and B are composed from Level-1 parts a, b and c, which are composed from Level-0 parts. Here a is composed from δ and ϵ, b is composed from δ and γ, and c is composed from ϵ and γ. This illustrates part sharing, for example ϵ is shared between parts a and c, part b is shared between A and B.

not shown here for simplicity), while the level-1 part a is composed from level-0 parts δ and ϵ (also ignoring their spatial relations).

In summary, we represent an object $\tau_{\mathcal{H}} \in \mathcal{M}_{\mathcal{H}}$ by a graph $\mathcal{V}^{\tau_{\mathcal{H}}} = \bigcup_{h=0}^{\mathcal{H}} \mathcal{V}_h^{\tau_{\mathcal{H}}}$. There is a single node at level $\mathcal{H}$, r nodes at level $\mathcal{H}-1$, and $r^{\mathcal{H}-h}$ nodes at level h. The state $\vec{x}_{\tau_{\mathcal{H}}}$ is specified by the state $x_{\tau_{\mathcal{H}}}$ of the root node and its descendants.

We note that the unsupervised learning algorithm in Zhu et al. (2010) automatically generates this hierarchical dictionary, as reviewed in section (6.2).

2.5 Probability Distributions for Objects

We specify the probability model for an object of type $\tau_{\mathcal{H}} \in \mathcal{M}_{\mathcal{H}}$ in terms of the probabilities of its part-subpart relations:

$$P(\vec{x}_{\tau_{\mathcal{H}}}|\tau_{\mathcal{H}}) = U(x_{\tau_{\mathcal{H}}}) \prod_{\nu \in \mathcal{V}^{\tau_{\mathcal{H}}}/\mathcal{V}_0^{\tau_{\mathcal{H}}}} P(x_{Ch(\nu)}|x_\nu; \tau_\nu). \tag{2}$$

Here $\mathcal{V}^{\tau_{\mathcal{H}}}/\mathcal{V}_0^{\tau_{\mathcal{H}}}$ denotes all the nodes of the graph $\mathcal{V}^{\tau_{\mathcal{H}}}$ except the leaf nodes $\mathcal{V}_0^{\tau_{\mathcal{H}}}$. $U(x_{\tau_{\mathcal{H}}})$ is the uniform distribution, which means that the object is equally likely to be anywhere in the image.

This model can be used to generate the positions of parts and subparts by sampling. This is performed by sampling the position $x_{\tau_{\mathcal{H}}}$ of the root node from $U(x_{\tau_{\mathcal{H}}})$, then sampling recursively from $P(x_{Ch(\nu)}|x_\nu; \tau_\nu)$ to get the positions of the parts and subparts until we obtain samples $\{x_\nu : \nu \in \mathcal{V}_0^{\tau_{\mathcal{H}}}\}$ for the leaf nodes.

3. Generative Models of Images

The models in the previous section specify distributions for the positions $\vec{x}_{\tau_{\mathcal{H}}}$ of an object $\tau_{\mathcal{H}}$, its parts and, in particular, for the positions $\{x_\nu : \nu \in \mathcal{V}_0^{\tau_{\mathcal{H}}}\}$ of the leaf nodes. In this section we describe how we can use these models to generate an image $\mathbf{I} = \{I(x) : x \in \mathcal{D}_0\}$.

This is done by introducing distributions for local image properties depending on the types and positions of the leaf nodes of the objects (e.g., the horizontal and vertical bars). We also specify a default, or background, distribution for the image properties at places where no low-level parts are present (i.e. no leaf nodes take these states).

More formally, we specify distributions $P(I(x)|\tau)$ for image property $I(x)$ at position $x \in \mathcal{D}_0$ if there is a leaf-level part $\tau \in \mathcal{M}_0$ at x. In addition, we specify a default *background distribution* $P(I(x)|\tau_0)$ for the image property at x if no part is present, which we denote by τ_0. For example, we could use the distributions for the properties of features on, or off, edges as described in Konishi et al. (2003). Note, we only allow the leaf-level parts $\mathcal{M}_0$ to directly generate image properties.

This gives a generative model for images. We first select an object at random from $\mathcal{M}_{\mathcal{H}}$ and sample from equation (2) to get a configuration of the object and its parts including, in particular, the leaf nodes. Then for each leaf node we sample from the appropriate $P(I(x)|\tau)$ depending on the type of the leaf node. At places where no leaf node is present we sample from $P(I(x)|\tau_0)$.

Observe that we can extend this generative model in two ways. Firstly, we can sample a background image if no object is present at all, by sampling from $P(I(x)|\tau_0)$, at every

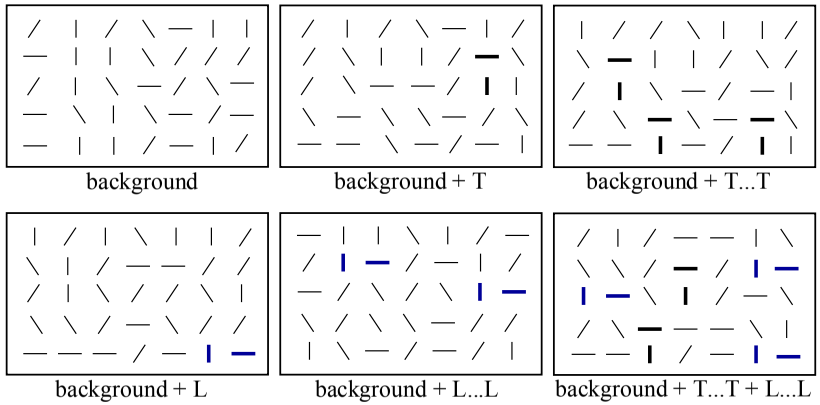


Figure 6: The generative model allows there to be several different objects in the image together with background clutter. This figure illustrates images generated by different models: (i) purely background, (ii) background with one T, (iii) background with several T's, (iv) background with an L, (v) background with several L's, and (vi) background with T's and L's.

position x. Secondly, we can sample for many objects being in the image provided they do not overlap, again by sampling from the models to find the positions of their leaf nodes and then sampling from $P(I(x)|\tau)$ or $P(I(x)|\tau_0)$ as appropriate. This can also be extended to sampling from objects and parts of objects. In this paper we mainly concentrate on the case where there is either a single object or no object in the image, although our analysis can be extended to these other more complex situations. We illustrate images sampled from different generative models in Figure (6).

3.1 The Generative Model for Objects and Backgrounds

We now specify the distributions more precisely. If there is no object in the image, then we generate it by sampling from the *background distribution*:

$$P_B(\mathbf{I}) = \prod_{x \in \mathcal{D}_0} P(I(x)|\tau_0). \tag{3}$$

This assumes that each image position x is background and so is sampled independently from $P(I(x)|\tau_0)$.

If there is a single object $\tau_{\mathcal{H}}$ present, then the prior $P(\vec{x}_{\tau_{\mathcal{H}}}|\tau_{\mathcal{H}})$, see equation (2), gives a distribution over a set of points $\mathcal{L}(\vec{x}_{\tau_{\mathcal{H}}}) = \{x_\nu : \nu \in \mathcal{V}_0^{\tau_{\mathcal{H}}}\}$ (the leaf nodes of the graph), which are a subset of the image lattice (i.e., $x_\nu \in \mathcal{D}_0$ if $\nu \in \mathcal{V}_0^{\tau_{\mathcal{H}}}$) and specifies their types $\{\tau_\nu : \nu \in \mathcal{V}_0^{\tau_{\mathcal{H}}}\}$. We denote these leaf nodes by $\{(x, \tau(x)) : x \in \mathcal{L}(\vec{x}_{\tau_{\mathcal{H}}})\}$ where $\tau(x)$ is specified in the natural manner (i.e. if $x = x_\nu$ for $\nu \in \mathcal{V}_0^{\tau_{\mathcal{H}}}$ then $\tau(x) = \tau_\nu$). Then we

sample from the distributions $P(I(x)|\tau(x))$ for the probability of the image $I(x)$ at positions $x \in \mathcal{L}(\vec{x}_{\tau_\mathcal{H}})$ conditioned on the type of the leaf node. We sample from the default $P(I(x)|\tau_0)$ at positions $x \in \mathcal{D}_0/\mathcal{L}(\vec{x}_{\tau_\mathcal{H}})$ where there is no leaf node of the object.

This specifies a likelihood function for the states $\vec{x}_{\tau_\mathcal{H}}$ of the object model $\tau_\mathcal{H}$. This likelihood function depends only on the leaf nodes $\nu \in \mathcal{V}_0^{\tau_\mathcal{H}}$:

$$P(\mathbf{I}|\vec{x}_{\tau_\mathcal{H}}) = \prod_{x \in \mathcal{L}(\vec{x}_{\tau_\mathcal{H}})} P(I(x)|\tau(x)) \times \prod_{x \in \mathcal{D}_0/\mathcal{L}(\vec{x}_{\tau_\mathcal{H}})} P(I(x)|\tau_0). \quad (4)$$

Combining this likelihood with the prior, specified in equation (2), gives the generative model for the image of an object:

$$P(\mathbf{I}|\vec{x}_{\tau_\mathcal{H}})P(\vec{x}_{\tau_\mathcal{H}}|\tau_\mathcal{H}) \quad (5)$$

, where the prior $P(\vec{x}_{\tau_\mathcal{H}}|\tau_\mathcal{H})$ is given by equation (2).

An important quality in our analysis is the *log-likelihood ratio* of the probability of an image $\mathbf{I}$ conditioned on whether there is an object in the image of type $\tau_\mathcal{H}$ and configuration $\vec{x}_{\tau_\mathcal{H}}$ or if no object is present, $P_B(\mathbf{I})$ from equation (3). This gives a log-likelihood ratio:

$$\log \frac{P(\mathbf{I}|\vec{x}_{\tau_\mathcal{H}})P(\vec{x}_{\tau_\mathcal{H}}|\tau_\mathcal{H})}{P_B(\mathbf{I})} = \sum_{x \in \mathcal{L}(\vec{x}_{\tau_\mathcal{H}})} \log \frac{P(I(x)|\tau(x))}{P(I(x)|\tau_0)} + \sum_{\nu \in \mathcal{V}^{\tau_\mathcal{H}}/\mathcal{V}_0^{\tau_\mathcal{H}}} \log P(x_{Ch(\nu)}|x_\nu;\tau_\nu) + \log U(x_{\tau_\mathcal{H}}), \quad (6)$$

where the $\sum_\nu$ is performed over all part-subpart relations for object type $\tau_\mathcal{H}$.

All the visual tasks we study in this paper can be reduced to computing the log-likelihood ratios for different objects and different configurations of their state variables. For example, if the log-likelihood ratios are small for all object configurations then we can decide that no object is present. Alternatively, the configuration which maximizes the log-likelihood ratio determines the most probable object in the image and the most likely positions of its parts, The next section shows how we can perform all these visual tasks exactly and efficiently using a visual architecture which exploits compositionality.

4. The Compositional Architecture and Exact Inference

We now discuss the inference algorithms required to perform the visual tasks of determining which objects, if any, are present in the image, and estimate their positions including the positions of their subparts. These visual tasks can be decomposed into two subtasks: (i) state estimation, to determine the optimal states for each model and hence the most probable positions of the objects and their parts, and (ii) model selection, to determine whether objects are present or not and which objects are most likely. As we will show, both tasks can be reduced to calculating and comparing log-likelihood ratios which can be performed efficiently using dynamic programming methods.

Firstly we describe, in section (4.1), how to do state estimation for each object separately and then discuss how we can perform model selection to decide which objects, if any, are

present in the image. Secondly, in section (4.2) we discuss how these inference tasks can be made more efficient for multiple object categories by exploiting part sharing using the hierarchical dictionaries. Thirdly, in section (4.3) we describe a parallel formulation for inference exploiting part sharing. We stress that we are performing *exact inference* and no approximations are made.

4.1 Inference for Single Objects: State Estimation and Model Selection

We first describe a standard dynamic programming algorithm for finding the optimal state of a single object model. Then we describe how the same computations can be used to perform model selection and hence applied to the detection and state estimation if the object appears multiple times in the image (provided it is non-overlapping).

Consider performing inference for a single object model $\tau_\mathcal{H}$ defined by equations (2) and (4). Calculating the MAP estimate of the state variables $\vec{x}_{\nu_\mathcal{H}}$ requires computing its most probable state $\vec{x}^*_{\tau_\mathcal{H}} = \arg\max_{\vec{x}_{\tau_\mathcal{H}}} \{\log P(\mathbf{I}|\vec{x}_{\tau_\mathcal{H}}) + \log P(\vec{x}_{\tau_\mathcal{H}}|\tau_\mathcal{H})\}$. This is equivalent to finding the state variables which maximize the log-likelihood ratio of $P(\mathbf{I}|\vec{x}_{\tau_\mathcal{H}})P(\vec{x}_{\tau_\mathcal{H}}|\tau_\mathcal{H})$ and the background distribution $P_B(\mathbf{I})$, which we specified in equation (6) (because $P_B(\mathbf{I})$ depends only on the image and is independent of the state variables). This requires estimating:

$$\vec{x}^*_{\tau_\mathcal{H}} = \arg\max_{\vec{x}_{\tau_\mathcal{H}}} \{ \sum_{x \in \mathcal{L}(\vec{x}_{\tau_\mathcal{H}})} \log \frac{P(I(x)|\tau(x))}{P(I(x)|\tau_0)} + \sum_{\nu \in \mathcal{V}^{\tau_\mathcal{H}}/\mathcal{V}_0^{\tau_\mathcal{H}}} \log P(x_{Ch(\nu)}|x_\nu;\tau_\nu) + \log U(x_{\tau_\mathcal{H}}) \}. \quad (7)$$

Here $\mathcal{L}$ denotes the positions of the leaf nodes of the graph, which must be determined during inference.

The optimal state $\vec{x}^*_{\tau_\mathcal{H}}$ can be estimated efficiently by dynamic programming. This involves a bottom-up pass which recursively computes the *local evidence* $\phi(x_\nu, \tau_\nu)$ for the *hypothesis* that there is a part τ_ν with state variable x_ν (i.e. with executive level summary x_ν). This is specified by $\phi(x_\nu, \tau_\nu) = \max_{\vec{x}_\nu/x_\nu} \{\log \frac{P(\mathbf{I}|\vec{x}_\nu)}{P_B(\mathbf{I})} + \log P(\vec{x}_\nu|\tau_\nu)\}$, where $\max_{\vec{x}_\nu/x_\nu}$ specifies that we fix the state of the position x_ν of the part τ_ν while maximizing over the states of its descendants (recall that $\vec{x}_\nu = (x_\nu, x_{Ch(\nu)}, ...)$). The local evidence can be interpreted as the log-ratio of the hypothesis test that there is a part τ_ν present at x_ν compared to the probability that it is not (i.e. the image properties are generated instead by the background model). The local evidences for the part hypotheses are computed bottom-up and we call it "local" because it ignores the context evidence for the part which will be provided during top-down processing (i.e. that evidence for other parts of the object, in consistent positions, will strengthen the evidence for this part). The local evidences for the part hypotheses are computed recursively by the formula:

$$\phi(x_\nu, \tau_\nu) = \max_{x_{Ch(\nu)}} \{ \sum_{i=1}^{r} \phi(x_{\nu_i}, \tau_{\nu_i}) + \log P(x_{Ch(\nu)}|x_\nu, \tau_\nu) \}. \quad (8)$$

At level-$\mathcal{H}$ the *bottom-up pass* outputs the *global evidence* $\phi(x_{\tau_\mathcal{H}}, \tau_\mathcal{H})$ for the hypothesis that object $\tau_\mathcal{H}$ occurs at position $x_{\tau_\mathcal{H}}$. This enables us to determine the best executive summary position for the object together with the log-likelihood ratio of the best state

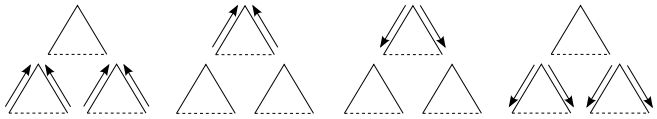


Figure 7: The feedforward (bottom-up) pass propagates hypotheses from the lowest level to the middle level (far left panel) and then from the middle level to the top level (left panel). The best top-level state is selected. The feedback (top-down) pass propagates information from the top node disambiguating the middle level nodes (right panel) and then from the middle level nodes to the lowest levels (far right panel). The bottom-up pass enables the algorithm to rapidly estimate the top-level executive summary description. The top-down pass enables high-level context to eliminate false hypotheses at the lower levels– informally, "high-level tells low-level to stop gossiping".

configuration:

$$x^*_{\tau_\mathcal{H}} = \arg\max_{x \in \mathcal{D}_\mathcal{H}} \phi(x, \tau_\mathcal{H}), \qquad \text{with evidence} \quad \phi(x^*_{\tau_\mathcal{H}}, \tau_\mathcal{H}). \tag{9}$$

We can compare the log-likelihood ratio of the best state configuration to perform *model selection* either by determining whether it is above a fixed threshold $T_\mathcal{H}$, to determine whether the object is present or not, or to compare its value for different objects to determine which object is most likely to be in the image.

Then we can perform a *top-down pass* of dynamic programming to estimate the most probable states $\vec{x}^*_{\tau_\mathcal{H}}$ of the entire model by recursively performing:

$$x^*_{Ch(\nu)} = \arg\max_{x_{Ch(\nu)}} \{\sum_{i=1}^{r} \phi(x_{\nu_i}, \tau_{\nu_i}) + \log P(x_{Ch(\nu)} | x^*_\nu, \tau_\nu)\}. \tag{10}$$

This outputs the most probable configuration of this object $\tau_\mathcal{H}$ in the image:

$$\vec{x}^*_{\tau_\mathcal{H}} = x^*_{\tau_\mathcal{H}}, ... \tag{11}$$

It should be emphasized that the algorithm first computes the best "executive summary" description $x_{\tau_\mathcal{H}}$ of the object as the output of the feedforward pass, together with the log-likelihood ratio for the optimal configuration, and only later determines the optimal estimates of the lower-level states of the object in the top-down pass. Hence, the algorithm is faster at detecting that there is a cow in the right side image (estimated in the bottom-up pass) and is slower at determining the position of the feet of the cow (estimated in the top-down pass). The algorithm is also quicker at determining whether there is a cow or a horse in the image, at the end of the bottom-up pass, than it is at estimating the most likely positions of the parts of the cow or the horse. This is illustrated in Figure (7).

The algorithm also has an intuitive interpretation. The bottom up pass propagates local hypotheses for parts up the hierarchy, see Figures (7,8). These hypotheses may be

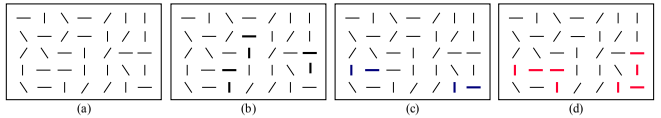


Figure 8: An alternative perspective on inference. Input image (far left panel), hypotheses for T's and L's (left and right panels), detections of the composite object (far right panel).

extremely ambiguous at the lower levels of the hierarchy due to the local ambiguity of images. But as we ascend the hierarchy the hypotheses become increasingly unambiguous because the parts become larger and more structured. This could be analyzed by using the techniques developed in Yuille et al. (2001). From another perspective, the top-down process "binds" the parts to the subparts and hence relates to the standard binding problem of neural networks Valiant (2000).

So far we have described how to use dynamic programming, with a bottom-up and top-down pass, to compute the MAP estimate of the configuration $\vec{x}_{\tau_\mathcal{H}}$ of an object $\tau_\mathcal{H}$ in the image. But this approach allows us to do more. For example, instead of detecting objects we can use the algorithm to detect parts again by using the evidence $\phi(x_\nu, \tau_\nu)$ as a log-likelihood ratio test to determine if the part is present or not. The log-likelihood test will tend to have a large error rate for low level parts (because of their ambiguity) but will increasingly have fewer errors for larger parts since these are less ambiguous. Intuitively, it is possible for a human to confuse the leg of a cow with the leg of a horse but it is extremely unlikely that an entire horse and cow cannot be distinguished. Using this strategy it may be possible to decide that an object part is present or not without waiting for the context provided by the top-down processing. Making premature decisions like this may cause problems, however, which we analyze in section (6.3).

In addition, we can compute the probability that the object occurs several times in the image, by computing the set $\{x_{\tau_\mathcal{H}} : \phi(x^*_{\tau_\mathcal{H}}, \tau_\mathcal{H}) > T_\mathcal{H}\}$, to compute the "executive summary" descriptions for each object (e.g., the coarse positions of each object). We then perform the top-down pass initialized at each coarse position (i.e. at each point of the set described above) to determine the optimal configuration for the states of the objects. Hence, we can reuse the computations required to detect a single object in order to detect multiple instances of the object (provided there are no overlaps). The number of objects in the image is determined by the log-likelihood ratio test with respect to the background model. It can be shown that this is equivalent to performing optimal inference simultaneously over a set of different generative models of the image, where one model assumes that there is one instance of the object in the image, another model assumes there are two (non-overlapping), and so on.

4.2 Compositional Architecture: Inference on Multiple Objects by Part Sharing using the Hierarchical Dictionaries

The previous section performed MAP estimation, and computed log-likelihood ratios, for each object separately. But this is wasteful since if the objects share parts because it computes the same quantities multiple times. This section describes how we can exploit part sharing to perform these computations more efficiently for many objects simultaneously.

The main idea is that we only need to compute the local evidence $\phi(x_\nu, \tau_\nu)$ for each part $\tau_\nu \in \mathcal{M}_h$ *once* independently of how many objects the part occurs in (or how many times it occurs within each object). This means that we can compute the evidence for all objects at all positions $\{\phi(\vec{x}_{\tau_\mathcal{H}}, \tau_\mathcal{H}) : \vec{x}_{\tau_\mathcal{H}} \in \mathcal{D}_\mathcal{H},\ \tau_\mathcal{H} \in \mathcal{M}_\mathcal{H}\}$ by recursively computing the evidences $\{\phi(x_\nu, \tau_\nu) : x_\nu \in \mathcal{D}_h,\ \tau_\nu \in \mathcal{M}_h\}$ using equation (8) for all $x_\nu \in \mathcal{D}_h,\ \tau_\nu \in \mathcal{M}_h$.

We then perform model selection at the top-level by thresholding the evidence for each object at each position. More precisely, for each object $\tau_\mathcal{H} \in \mathcal{M}_\mathcal{H}$ we determine the set of positions $\mathcal{S}_{\tau_\mathcal{H}} = \{x_{\tau_\mathcal{H}} \in \mathcal{D}_\mathcal{H} : s.t.\ \phi(x_{\tau_\mathcal{H}}, \tau_\mathcal{H}) > T_\mathcal{H}\}$ where the evidence for that object is above threshold. Then we perform the top-down pass, equation (10), starting at all positions $\mathcal{S}_{\tau_\mathcal{H}}$ for all $\tau_\mathcal{H} \in \mathcal{M}_\mathcal{H}$. In short, we perform bottom-up inference to detect which objects are present and the positions of the executive summaries. Then we perform top-down processing to estimate the positions of the parts, the full configurations, of the detected objects. Note, if this process detects two or more objects in the same position then we will need to compare their evidences to select which one is most likely, but our framework assumes that this is unlikely to occur because there is little ambiguity for complete objects.

Note that we are performing exact inference over multiple object models at the same time. This may be un-intuitive to some readers because this corresponds to doing exact inference over a probability model which can be expressed as a graph with a large number of closed loops, see Figure (4). We note that this applies only to a subclass of compositional models including Zhu et al. (2008, 2010). The main point is that part-sharing enables us perform inference efficiently for many models at the same time.

4.3 Parallel Formulation of the Compositional Architecture

Finally, we observe that all the computations required for performing inference on multiple objects can be parallelized.

Parallelization is possible for both the bottom-up and the top-down passes. Firstly, the bottom-up pass requires calculating the $\phi(x_\nu, \tau_\nu)$'s at each layer-h using equation (8). These calculations are done separately for each type $\tau_\nu \in \mathcal{M}_h$ and at each spatial position $x_\nu \in \mathcal{D}_h$, see equation (8). They depend only on input from local subregions of layer-$(h-1)$. Hence they can be computed in parallel. This enables us to compute the global evidence for objects at the top level: $\phi(x_{\tau_\mathcal{H}}, \tau_\mathcal{H})$ for all $x_{\tau_\mathcal{H}} \in \mathcal{D}_\mathcal{H}$ and $\tau_\mathcal{H} \in \mathcal{M}_\mathcal{H}$. We threshold at each node $x_{\tau_\mathcal{H}} \in \mathcal{D}_\mathcal{H}$ to determine if the object is detected, which is also a local operation. Secondly, the top-down pass is also parallelizable because its operation, see equation (10), is also local. Note that top-down processing is only done starting from nodes at level-$\mathcal{H}$ where objects have been detected.

This leads to a graphical architecture which contains $|\mathcal{M}_h| \times |\mathcal{D}_h|$ nodes at level h, and hence a total of $\sum_{h=0}^{\mathcal{H}} |\mathcal{M}_h| \times |\mathcal{D}_h|$ nodes. There are a total of $r \sum_{h=1}^{\mathcal{H}} |\mathcal{M}_h| \times |\mathcal{D}_h|$ connections. These connections are local and "convolutional" in the sense that they are

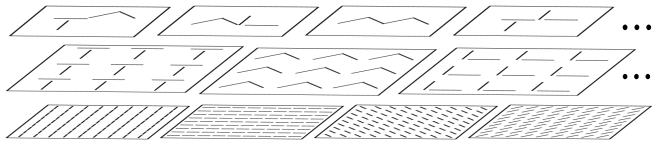


Figure 9: Parallel Hierarchical Implementation. Bottom Row: Four Level-0 parts are placed at each point in the image lattice $\mathcal{D}_0$. Middle Row: Level-1 parts are placed at each point on the coarser lattice $\mathcal{D}_1$ (we show three Level-1 models). Top Row: object models are placed at the nodes of the top level lattice ($\mathcal{H} = 2$ in this figure). This can be thought of as non-linear receptive fields analogous to bio-inspired hierarchical models.

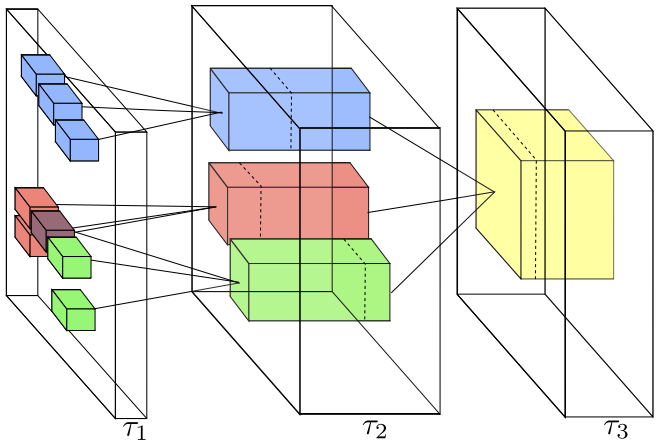


Figure 10: This figure illustrates the parallel hierarchical architecture. At each level there are nodes tuned to different types of parts (similar to the different features occurring in bio-inspired models).

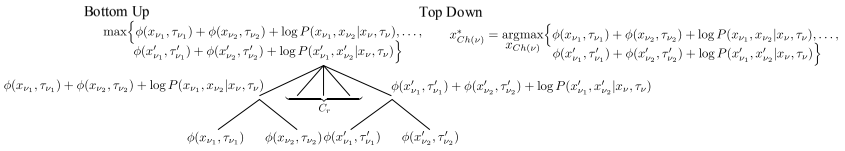


Figure 11: Parallel implementation of Dynamic Programming. The left part of the figure shows the bottom-up pass of dynamic programming. The local evidence for the parent node is obtained by taking the maximum of the scores of the C_r possible states of the child nodes. This can be computed by a two-layer network where the first level computes the scores for all C_r child node states, which can be done in parallel, and the second level compute the maximum score. This is like an AND operation followed by an OR. The top-down pass requires the parent node to select which of the C_r child configurations gave the maximum score, and suppressing the other configurations.

the same at different places in the image. Nodes at level-h can be represented by (τ_ν, x_ν) and so they represent a part of type $\tau_\nu \in \mathcal{M}_h$ at position $x_\nu \in \mathcal{D}_h$. This is illustrated in Figures (9) and (10).

We can think of the nodes at different levels of the hierarchy as being non-linear "receptive fields". At level-0, these receptive fields are tuned to specific image properties (e.g., horizontal or vertical bars). Note that the receptive fields are highly non-linear (i.e. they do not obey the superposition principle). Nevertheless they are broadly speaking, tuned to image stimuli which have the mean shape of the corresponding part τ_ν. In agreement, with findings about mammalian cortex, the receptive fields become more sensitive to image structure (e.g., from bars, to more complex shapes) at increasing levels. Moreover, their sensitivity to spatial position decreases because at higher levels the models only encode the executive summary descriptions, on coarser lattices, while the finer details of the object are represented more precisely at the lower levels. Moreover, they are influenced both by bottom-up processing (during the bottom-up pass) and by top-down processing (during the top-down pass). The bottom-up processing computes the local evidence while the top-down pass modifies it by the high-level context.

The computations required by this parallel implementation are illustrated in Figure (11). Each step of the bottom-up pass is performed by a two-layer network, see Figure (11), where the first layer performs an AND operation (to compute the local evidence for a specific configuration of the child nodes) and the second layer performs an OR, or max operation, to determine the local evidence (by max-ing over the possible child configurations). The top-down pass only has to perform an arg max computation to determine which child configuration gave the best local evidence.

5. Complexity Analysis

We now analyze the complexity of the inference algorithms for performing the tasks of object detection and object parsing. Firstly, we analyze complexity for a single object (without part-sharing). Secondly, we study the complexity for multiple objects with shared parts. Thirdly, we consider the complexity of the parallel implementation.

The complexity is expressed in terms of the following quantities: (I) The size $|\mathcal{D}_0|$ of the image. (II) The scale decrease factor q (enabling executive summary). (III) The number $\mathcal{H}$ of levels of the hierarchy. (IV) The sizes $\{|\mathcal{M}_h| : h = 0, ..., \mathcal{H}\}$ of the hierarchical dictionaries. (V) The number r of subparts of each part. (VI) The number C_r of possible part-subpart configurations.

5.1 Complexity for Single Objects and Ignoring Part Sharing

This section estimates the complexity of inference N_{s_o} for a single object and the complexity N_{m_o} for multiple objects when part sharing is not used. These results are for comparison to the complexities derived in the following section which use part sharing.

The inference complexity for a single object requires computing: (i) the number N_{bu} of computations required by the bottom-up pass, (ii) the number N_{m_s} of computations required by model selection at the top-level of the hierarchy, and (iii) the number N_{td} of computations required by the top-down pass.

The complexity N_{bu} of the bottom-up pass can be computed from equation (8). This requires a total of C_r computations for each position x_ν for each level-h node. There are $r^{\mathcal{H}-h}$ nodes at level-h and each can take $|\mathcal{D}_0|q^h$ positions. This gives a total of $|\mathcal{D}_0|C_r q^h r^{\mathcal{H}-h}$ computations at level h. This can be summed over all levels to yield:

$$N_{bu} = \sum_{h=1}^{\mathcal{H}} |\mathcal{D}_0|C_r r^{\mathcal{H}}(q/r)^h = |\mathcal{D}_0|C_r r^{\mathcal{H}} \sum_{h=1}^{\mathcal{H}} (q/r)^h = |\mathcal{D}_0|C_r \frac{qr^{\mathcal{H}-1}}{1-q/r}\{1-(q/r)^{\mathcal{H}}\}. \quad (12)$$

Observe that the main contributions to N_{bu} come from the first few levels of the hierarchy because the factors $(q/r)^h$ decrease rapidly with h. This calculation uses $\sum_{h=1}^{\mathcal{H}} x^h = \frac{x(1-x^{\mathcal{H}})}{1-x}$. For large $\mathcal{H}$ we can approximate N_{bu} by $|\mathcal{D}_0|C_r \frac{qr^{\mathcal{H}-1}}{1-q/r}$ (because $(q/r)^{\mathcal{H}}$ will be small).

We calculate $N_{m_s} = q^{\mathcal{H}}|\mathcal{D}_0|$ for the complexity of model selection (which only requires thresholding at every point on the top-level lattice).

The complexity N_{td} of the top-down pass is computed from equation (10). At each level there are $r^{\mathcal{H}-h}$ nodes and we must compute C_r computations for each. This yields complexity of $\sum_{h=1}^{\mathcal{H}} C_r r^{\mathcal{H}-h}$ for each possible root node. There are at most $q^{\mathcal{H}}|\mathcal{D}_0|$ possible root nodes (depending on the results of the model selection stage). This yields an upper bound:

$$N_{td} \leq |\mathcal{D}_0|C_r q^{\mathcal{H}} \frac{r^{\mathcal{H}-1}}{1-1/r}\{1-\frac{1}{r^{\mathcal{H}-1}}\}. \quad (13)$$

Clearly the complexity is dominated by the complexity N_{bu} of the bottom-up pass. For simplicity, we will bound/approximate this by:

$$N_{s_o} = |\mathcal{D}_0|C_r \frac{qr^{\mathcal{H}-1}}{1-q/r}. \quad (14)$$

Now suppose we perform inference for multiple objects simultaneously without exploiting shared parts. In this case the complexity will scale linearly with the number $|\mathcal{M}_{\mathcal{H}}|$ of objects. This gives us complexity:

$$N_{m_o} = |\mathcal{M}_{\mathcal{H}}||\mathcal{D}_0|C_r\frac{qr^{\mathcal{H}-1}}{1-q/r}. \tag{15}$$

5.2 Computation with Shared Parts in Series and in Parallel

This section computes the complexity using part sharing. We first study complexity for the standard serial implementation of part sharing and then for the parallel implementation.

Now suppose we perform inference on many objects with part sharing using a serial computer. This requires performing computations over the part-subpart compositions between elements of the dictionaries. At level h there are $|\mathcal{M}_h|$ dictionary elements. Each can take $|\mathcal{D}_h| = q^h|\mathcal{D}_0|$ possible states. The bottom-up pass requires performing C_r computations for each of them. This gives a total of $\sum_{h=1}^{\mathcal{H}}|\mathcal{M}_h|C_r|\mathcal{D}_0|q^h = |\mathcal{D}_0|C_r\sum_{h=1}^{\mathcal{H}}|\mathcal{M}_h|q^h$ computations for the bottom-up process. The complexity of model selection is $|\mathcal{D}_0|q^{\mathcal{H}} \times (|\mathcal{M}_{\mathcal{H}}|+1)$ (this is between all the objects, and the background model, at all points on the top lattice). As in the previous section, the complexity of the top-down process is less than the complexity of the bottom-up process. Hence the complexity for multiple objects using part sharing is given by:

$$N_{ps} = |\mathcal{D}_0|C_r\sum_{h=1}^{\mathcal{H}}|\mathcal{M}_h|q^h. \tag{16}$$

Next consider the parallel implementation. In this case almost all of the computations are performed in parallel and so the complexity is now expressed in terms of the number of "neurons" required to encode the dictionaries, see Figure (9). This is specified by the total number of dictionary elements multiplied by the number of spatial copies of them:

$$N_n = \sum_{h=1}^{\mathcal{H}}|\mathcal{M}_h|q^h|\mathcal{D}_0|. \tag{17}$$

The computation, both the forward and backward passes of dynamic programming, are linear in the number $\mathcal{H}$ of levels. We only need to perform the computations illustrated in Figure (11) between all adjacent levels.

Hence the parallel implementation gives speed which is linear in $\mathcal{H}$ at the cost of a possibly large number N_n of "neurons" and connections between them.

5.3 Advantages of Part Sharing in Different Regimes

The advantages of part-sharing depend on how the number of parts $|\mathcal{M}_h|$ scales with the level h of the hierarchy. In this section we consider three different regimes: (I) The *exponential growth regime* where the size of the dictionaries increases exponentially with the level h. (II) The *empirical growth regime* where we use the size of the dictionaries found experimentally by compositional learning Zhu et al. (2010). (III) The *exponential decrease regime* where the size of the dictionaries decreases exponentially with level h. For all these

regimes we compare the advantages of the serial and parallel implementations using part sharing by comparison to the complexity results obtained without sharing.

Exponential growth of dictionaries is a natural regime to consider. It occurs when subparts are allowed to combine with all other subparts (or a large fraction of them) which means that the number of part-subpart compositions is polynomial in the number of subparts. This gives exponential growth in the size of the dictionaries if it occurs at different levels (e.g., consider the enormous number of objects that can be built using lego).

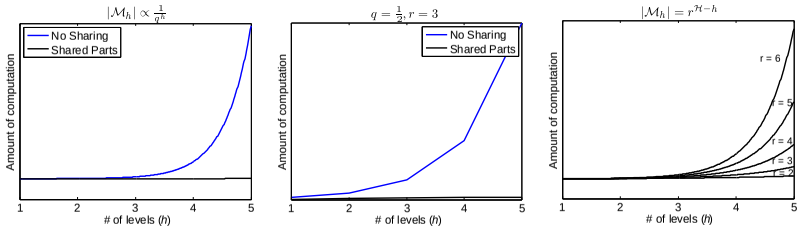
An interesting special case of the exponential growth regime is when $|\mathcal{M}_h|$ scales like $1/q^h$ (recall $q < 1$), see Figure (12) (left panel). In this case the complexity of computation for serial part-sharing, and the number of neurons required for parallel implementation, scales only with the number of levels $\mathcal{H}$. This follows from equations (16) and (17). But nevertheless the number of objects that can be detected scales exponentially with $\mathcal{H}$, as $(1/q)^{\mathcal{H}}$. By contrast, the complexity of inference without part-sharing also scales exponentially as $(1/q)^{\mathcal{H}}$, see equation (15), because we have to perform a fixed number of computations, given by equation (14), for each of an exponential number of objects. This is summarized by the following result.

Result 1: If the number of shared parts scales exponentially by $|\mathcal{M}_h| \propto \frac{1}{q^h}$ then we can perform inference for order $(1/q)^{\mathcal{H}}$ objects using part sharing in time linear in $\mathcal{H}$, or with a number of neurons linear in $\mathcal{H}$ for parallel implementation. By contrast, inference without part-sharing requires exponential complexity.

To what extent is exponential growth a reasonable assumption for real world objects? This motivates us to study the empirical growth regime using the dictionaries obtained by the compositional learning experiments reported in Zhu et al. (2010). In these experiments, the size of the dictionaries increased rapidly at the lower levels (i.e. small h) and then decreased at higher levels (roughly consistent with the findings of psychophysical studies – Biederman, personal communication). For these "empirical dictionaries" we plot the growth, and the number of computations at each level of the hierarchy, in Figure (12)(center panel). This shows complexity which roughly agrees with the exponential growth model. This can be summarized by the following result:

Result 2: If $|\mathcal{M}_h|$ grows slower than $1/q^h$ and if $|\mathcal{M}_h| < r^{\mathcal{H}-h}$ then there are gains due to part sharing using serial and parallel computers. This is illustrated in Figure (12)(center panel) based on the dictionaries found by unsupervised computational learning Zhu et al. (2010). In parallel implementations, computation is linear in $\mathcal{H}$ while requiring a limited number of nodes ("neurons").

Finally we consider the exponential decrease regime. To motivate this regime, suppose that the dictionaries are used to model image appearance, by contrast to the dictionaries based on geometrical features such as bars and oriented edges (as used in Zhu et al. (2010)). It is reasonable to assume that there are a large number of low-level dictionaries used to model the enormous variety of local intensity patterns. The number of higher-level dictionaries can decrease because they can be used to capture a cruder summary description of a larger image region, which is another instance of the executive summary principle. For example, the low-level dictionaries could be used to provide detailed modeling of the local appearance of a cat, or some other animal, while the higher-level dictionaries could give simpler descriptions like "cat-fur" or "dog-fur" or simply "fur". In this case, it is plausible



(a) (b) (c)

Figure 12: The curves are plotted as a function of h. Left panel: The first plot is the case where $M_h = a/(q^h)$. So we have a constant cost for the computations, when we have shared parts. Center panel: This plot is based on the experiment of Zhu et al. (2010). Right panel: The third plot is the case where M_h decreases exponentially. The amount of computation is the same for the shared and non-shared cases.

that the size of the dictionaries decreases exponentially with the level h. The results for this case emphasize the advantages of parallel computing.

Result 3: If $|\mathcal{M}_h| = r^{\mathcal{H}-h}$ then there is no gain for part sharing if serial computers are used, see Figure (12)(right panel). Parallel implementations can do inference in time which is linear in $\mathcal{H}$ but require an exponential number of nodes ("neurons").

Result 3 may appear negative at first glance even for the parallel version since it requires an exponentially large number of neurons to encode the lower level dictionaries. But it may relate to one of the more surprising facts about the visual cortex in monkeys and humans, if we identify the nodes of the compositional models with neurons in the visual cortex, namely that the first two visual areas, V1 and V2, where low-level dictionaries would be implemented are enormous compared to the higher levels such as IT where object detection takes places. Current models of V1 and V2 mostly relegate them to being a large filter bank which seems paradoxical considering their size. For example, Lennie (1998) has stated when reviewing the functions of V1 and V2 "perhaps the most troublesome objection to the picture I have delivered is that an enormous amount of cortex is used to achieve remarkably little". Perhaps the size of these visual areas is because they are encoding dictionaries of visual appearance.

6. Discussion

This section discusses three important topics. Section (6.1) discusses the relation of compositional models to the more traditional bio-inspired hierarchical models, such as deep convolutional networks. In section (6.2) we briefly describe how compositional models can be learnt in an unsupervised manner. Section (6.3) describes how to extend compositional models to make them more robust, and describes an analysis of robustness which is presented in detail in Appendix A.

6.1 Compositional Models and Bio-inspired Hierarchical Architectures

The parallel compositional architecture described in the last two sections has several similarities to bio-inspired models and in particular to deep convolutional networks Krizhevsky et al. (2012). The graphical structures of the models are similar and the nodes at each level of the hierarchy are indexed by position in the image lattice and by the type of the part, or correspondingly, the index of the feature detector (i.e. the class of feature detectors in a convolutional network corresponds to the set of part types for compositional models).

But there are also several differences which we now discuss. Compositional models were developed from the literature on deformable template models of objects where the representation of objects parts and spatial relations is fundamental. These deformable template models are designed to detect the parts of objects and not simply to detect which objects are present in an image. Compositional models, and the closely related grammar models, enable parts to be shared between multiple objects while, by contrast, bio-inspired models share features.

In particular, compositional models have explicit part-subpart relationships which enable them to represent spatial relations explicitly and to perform tasks like parsing in addition to detection. These part-subpart relations are learnt by hierarchical clustering algorithms Zhu et al. (2008, 2010) so that a part is composed from r subparts and the spatial relations between the subparts are explicitly modeled. This differs from the way that features are related in convolutional networks. The difference is most apparent at the final layers where convolutional networks are fully connected (i.e. all features at the top-level can be used for all objects) while compositional models still restrict the objects to be composed of r subparts. In other words, the final level of a compositional model is exactly the same as all the previous levels (i.e. it does not extract hierarchical features using one learning algorithm and then learn a classifier on top).

In addition, the compositional models are generative. This has several advantages including the possibility of dealing robustly with occlusion and missing parts, as we discuss in the next section, and the ability (in principle) to synthesize images of the object by activating top-level object nodes. These abilities suggest models for visual attention and other top-down processing.

Using this compositional architecture, the inference is performed by a bottom-up process which propagates hypotheses (for the states of nodes at the low-levels of the hierarchy) up the hierarchy. As they ascend the hierarchy they have access to more information about the image (i.e. the higher level nodes represent larger parts of the image) and hence become less ambiguous. A top-down process is used to disambiguate the lower level hypotheses using the context provided by the higher level nodes – "high-level tells low-level to stop gossiping". We stress that this is exact inference – similar (and sometimes identical) to dynamic programming on probabilistic context free grammars. Hence, as implemented in the compositional architecture, we see that we can have generative models of many objects but we can nevertheless perform inference rapidly by bottom-up processing followed by top-down stage which determines the positions of the object parts/subparts by eliminating false hypotheses.

More technically, we now specify a rough translation between compositional models and deep convolutional neural networks (DCNNs). The types τ in compositional models

translate into the feature channels of DCNNs. The local evidence $\phi(x_\nu, \tau_\nu)$ for type τ at position x_ν corresponds to the activation $z_{i,k}$ of the k^{th} channel at position i in a DCNN. If we ignore max-pooling, for simplicity, the DCNN will have update rule:

$$\phi(x_\nu, \tau_\nu) = \sum_{i, x_{\nu_i}} \omega_{i, x_{\nu_i}} \max\{0, \phi(x_{\nu_i}, \tau_{\nu_i})\},$$

with weights $\omega_{i,x_{\nu_i}}$ and where the summation is over the child nodes ν_i and over their positions x_{ν_i}. The max-pooling replaces $\phi(x_{\nu_i}, \tau_{\nu_i})$ by $\max_{y \in Nbh(x_{\nu_i})} \phi(y, \tau_{\nu_i})$, where $Nbh(x_{\nu_i})$ is a spatial neighbourhood centered on x_{ν_i}. Hence both update rules involve maximization steps but for DCNNs they are independent for each feature channel and for compositional models they depend on the states of all the children. DCNN includes weights $\omega_{i,x_{\nu_i}}$, which are trained discriminatively by backpropagation, while compositional models use probabilities $P(x_{Ch(\nu)}|x_\nu, \tau_\nu)$. The learning for compositional models is unsupervised and is discussed in section (6.2).

6.2 Unsupervised Learning

This section briefly sketches the unsupervised learning algorithm. We refer to Zhu et al. (2008, 2010); Yuille (2011) for more details. The strategy is hierarchical clustering which proceeds by recursively learning the dictionaries.

We start with a level-0 dictionary of parts $\tau \in \mathcal{M}_0$ with associated models $P(I(x)|\tau)$ and a default distribution $P(I(x)|\tau_0)$. These are assumed as inputs to the learning process. But they could be learnt by unsupervised learning as in Papandreou et al. (2014); Mao et al. (2014).

Then we select a threshold T_0 and for each level-0 type $\tau \in \mathcal{M}_0$ we detect a set of image positions $\{x_i^\tau : i = 1, ..., N\}$ where the log-likelihood test for τ is above threshold, i.e. $\log \frac{P(I(x_i^\tau)|\tau)}{P(I(x)|\tau_0)} > T_0$. This corresponds to the set of places where the part has been detected. This process depends on the threshold T_0 which will be discussed later.

Next we seek compositions of parts that occur frequently together with regular spatial arrangements. This is performed by clustering the detected parts using a set of probabilistic models of form $h(x_{\tau_1}, ..., x_{\tau_r}; \lambda)$, see equation (1), to estimate the λ's (the clustering is done separately for different combinations of level-0 types). This yields the level-1 dictionary $\mathcal{M}_1$, where each level-1 part is described by a probability distribution specified by the types of its children $\tau_1, ..., \tau_r$ and the spatial relations given by $h(.,.,...;\lambda)$. We then repeat the process. More specifically, we detect level-1 parts using the log-likelihood ratio test with a threshold T_1 and perform clustering to create the level-2 dictionary $\mathcal{M}_2$. We continue recursively using detected parts as inputs to another spatial clustering stage. The process terminates automatically when we fail to detect any new clusters. This will depend on our choice of clusters, the detection thresholds, and a threshold on the number of instances needed to constitute a cluster. For more details see Zhu et al. (2008, 2010). The whole process can be interpreted as a breadth first search through the space of possible generative models of the image, see Yuille (2011).

6.3 Robust Compositional Models

It is also important to study the robustness of compositional models when some parts are undetected. This is an important issue because objects can often be partially occluded in which case some parts are not observed directly. Also some forms of inference require thresholding the log-likelihood ratios of objects to determine if they have been detected (i.e. without waiting to detect the complete object) and we need to understand what happens for the false negatives when we threshold the log-likelihoods. As described above, thresholding is also used during learning, which gives more motivation for understanding the errors it causes and how to minimize them. Finally, we note that any realistic neural implementation of compositional models must be robust to failures of neural elements.

These issues were addressed in previous work Zhu et al. (2008, 2010) which showed empirically that compositional models could be made robust to some errors of this type. This section briefly describes some of these extensions. In addition to the motivations given above, we also want ways to extend compositional models to allow for parts to have variable numbers of subparts or to be partially invariant to image transformations. Firstly, the part-subpart distributions $P(\vec{x}_{Ch(\nu)}|x_\nu, \tau_\nu)$ were made insensitive to rotations and expansions in the image plane. Secondly, the 2-out-of-3 rule (described below) made the model robust to failure to detect parts or to malfunctioning neurons. Thirdly, the imaging terms could be extended so that the model generates image features (instead of the image values directly) and could include direct image input to higher levels. All these extensions were successfully tested on natural images Zhu et al. (2008, 2010).

The 2-out-of-3 rule is described as follows. In our implementations Zhu et al. (2008, 2010) a part-subpart composition has three subparts. The 2-out-of-3 rule allowed the parent node to be activated if only two subparts were detected, even if the image response at the third part is inconsistent with the object (e.g., if the object is partially occluded). The intuition is that if two parts are detected then the spatial relations between the parts, embedded in the term $h(\vec{x}; \lambda_\nu)$ of $P(\vec{x}_{Ch(\nu)}|x_\nu, \tau_\nu)$, is sufficient to predict the position of the third part. This can be used during inference while paying a penalty for not detecting the part. From another perspective, this is equivalent to having a mixture of different part-subparts compositions where the part could correspond to three subparts or two subparts. This can be extended to allow a larger range of possible subpart combinations thereby increasing the flexibility. Note that this would not affect the computational time for a parallel model.

We analyzed the robustness of the 2-out-of-3 rule when parts are missing. This analysis is presented in appendix A. It shows that provided the probability of detecting each part is above a threshold value (assuming the part is present) then the probability of detecting the entire object correctly tends to 1 as the number of levels of the hierarchy increases.

7. Summary

This paper provides a complexity analysis of what is arguably one of the most fundamental problem of visions – how, a biological or artificial vision system could rapidly detect and recognize an enormous number of different objects. We focus on a class of hierarchical compositional models Zhu et al. (2008, 2010) whose formulation makes it possible to perform this analysis. We conjecture that similar results, exploiting the sharing of parts and hierarchical distributed representations, will apply to the more sophisticated models needed

to address the full complexity of object detection and parsing. Similar results may apply to related bio-inspired hierarchical models of vision (e.g., those cited in the introduction). We argue that understanding complexity is a key issue in the design of artificial intelligent systems and understanding the biological nature of intelligence. Indeed perhaps the major scientific advances this century will involve the study of complexity Hawking (2000). We note that complexity results have been obtained for other visual tasks Blanchard and Geman (2005),Tsotsos (2011).

Technically this paper has required us to re-formulate compositional models to define a compositional architecture which facilitates the sharing of parts. It is noteworthy that we can effectively perform exact inference on a large number of generative image models (of this class) simultaneously using a single architecture. It is interesting to see if this ability can be extended to other classes of graphical models and be related to the literature on rapid ways to do exact inference, see Chavira et al. (2004).

Finally, we note that the parallel inference algorithms used by this class of compositional models have an interesting interpretation in terms of the bottom-up versus top-down debate concerning processing in the visual cortex DiCarlo et al. (2012). The algorithms have rapid parallel inference, in time which is linear in the number of layers, and which rapidly estimates a coarse "executive summary" interpretation of the image. The full interpretation of the image takes longer and requires a top-down pass where the high-level context is able to resolve ambiguities which occur at the lower levels. Of course, for some simple images the local evidence for the low level parts is sufficient to detect the parts in the bottom-up pass and so the top-down pass is not needed. But more generally, in the bottom-up pass the neurons are very active and represent a large number of possible hypotheses which are pruned out during the top-down pass using context, when "high-level tells low-level to stop gossiping".

Acknowledgments

Many of the ideas in this paper were obtained by analyzing models developed by L. Zhu and Y. Chen in collaboration with the first author. G. Papandreou and C. Coryell gave very useful feedback on drafts of this work. D. Kersten gave patient feedback on speculations about how these ideas might relate to the brain. The WCU program at Korea University, under the supervision of S-W Lee, gave peace and time to develop these ideas. We gratefully acknowledge support from the ARO 62250-CS and and the Center for Minds, Brains and Machines (CBMM), funded by NSF STC award CCF-1231216.

Appendix A

We analyze the 2-out-of-3 rule to show its robustness to missing parts. Consider two layers of part-subpart compositions with a parent part, three subparts, and nine sub-subparts. The 2-out-of-3 rule enables us to detect the part even in cases where only four of the nine sub-subparts are detected (provided that one subpart has no sub-subparts detected and the remaining two subparts have two sub-subparts detected each). To study this in more detail, let ρ be the probability of detecting a subpart and $f(\rho)$ be the probability of detecting the

part. Using the 2-out-of-3 rule, we obtain the update rule:

$$\rho_{h+1} = f(\rho_h), \quad \text{with } f(\rho) = \rho^3 + 3\rho^2(1-\rho). \tag{18}$$

We can analyze the robustness of the rule by studying the behavior of the iterative map $\rho_{t+1} = f(\rho_t)$. It can be calculated that there are fixed points at $\rho = 0, 0.5, 1$. Observe that $f(\rho) > \rho$ for $0.5 < \rho < 1$ and $f(\rho) < \rho$ for $0 < \rho < 0.5$, see Figure (13). Hence if the initial value of $\rho < 0.5$ (i.e. the detectors at the leaf nodes miss the object more than half the time) then the iterative map converges to zero and we cannot detect the object. Conversely, if $\rho > 0.5$ (the initial detectors miss parts less than half the time) then the iterative map converges to 1 and we always detect the object if there are a sufficient number of levels.

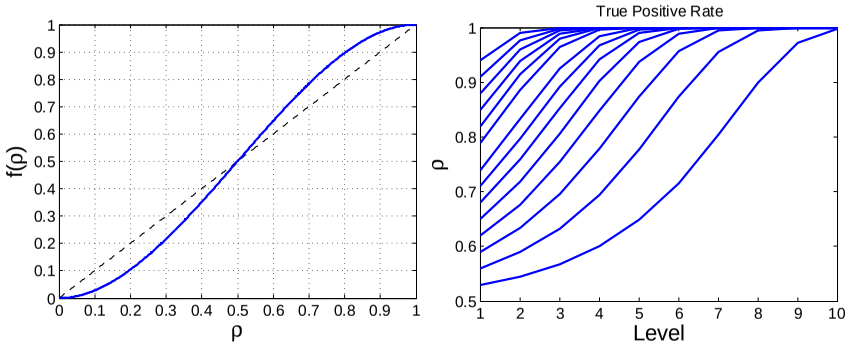


Figure 13: Left Panel: we plot $f(\rho)$ as a function of ρ, showing fixed points at $\rho = 0, 0.5, 1$, and that $f(\rho) > \rho$ for $0.5 < \rho < 1$, but $f(\rho) < \rho$ for $0 < \rho < 0.5$. Right Panel: we show the true positive rate (the detection rate) as a function of the number of levels and in terms of the ρ parameter.

We performed simulations to verify this result and to estimate how many levels are needed to obtain almost perfect detection as a function of ρ. These are shown in Figure (13)(right panel). Observe that if $\rho > 0.6$ then only a small number of levels are needed to get almost perfect performance.

References

N. J. Adams and C. K. I. Williams. Dynamic trees for image modelling. *Image and Vision Computing*, 20(10):865–877, 2003.

I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

G. Blanchard and D. Geman. Hierarchical testing designs for pattern recognition. *Annals of Statistics*, 35:11551202, 2005.

E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *European Conference on Computer Vision (ECCV)*, 2002.

M. Chavira, A. Darwiche, and M. Jaeger. Compiling relational bayesian networks for exact inference. In *International Journal of Approximate Reasoning*, pages 49–56, 2004.

J. J. DiCarlo, D. Zoccolan, and N. C. Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

K. Fukushima. Neocognitron - a hierarchical neural network capable of visual-pattern recognition. *Neural Networks*, 1(2):119–130, 1988.

S. Geman, D. F. Potter, and Z. Chi. Composition systems. *Quarterly of Applied Mathematics*, 60(4):707–736, 2002.

S.W. Hawking. I think the next century will be the century of complexity. *San Jose Mercury News*, January 23, 2000.

G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–54, 2006.

D. Kersten. Predictability and redundancy of natural images. *JOSA A*, 4(12):2395–2400, 1987.

I. Kokkinos and A. Yuille. Inference and learning with hierarchical shape models. *International Journal of Computer Vision (IJCV)*, 93(2):201–225, 2011.

S. M. Konishi, A. Yuille, J. Coughlan, and S. C. Zhu. Statistical edge detection: Learning and evaluating edge cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 25:57–74, 2003.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time-series. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*. MIT Press, 1995.

P. Lennie. Single units and visual cortical organization. *Perception*, 27:889–935, 1998.

J. Mao, J. Zhu, and A. Yuille. An active patch model for real world texture and appearance classification. In *European Conference on Computer Vision (ECCV)*, 2014.

G. Papandreou, L.-C. Chen, and A. Yuille. Modeling image patches with a generic dictionary of mini-epitomes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

H. Poon and P. Domingos. Sum-product networks: A new deep architecture. In *Uncertainty in Artificial Intelligence (UAI)*, 2010.

M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2:1019–1025, 1999.

T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29:411–426, 2007.

S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381(6582):520–522, 1996.

J. K. Tsotsos. *A Computational Perspective on Visual Attention*. The MIT Press, 1st edition, 2011. ISBN 0262015412, 9780262015417.

L. G. Valiant. *Circuits of the Mind*. Oxford University Press, 2000.

A. Yuille. Towards a theory of compositional learning and encoding of objects. In *ICCV Workshop on Information Theory in Computer Vision and Pattern Recognition*, 2011.

A. L. Yuille, J.M. Coughlan, Y.N. Wu, and S.C. Zhu. Order parameters for minimax entropy distributions: When does high level knowledge help? *International Journal of Computer Vision*, 41(1/2):9–33, 2001.

M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. Deconvolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

L. Zhu, C. Lin, H. Huang, Y. Chen, and A. Yuille. Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In *European Conference on Computer Vision (ECCV)*, 2008.

L. Zhu, Y. Chen, A. Torralba, W. Freeman, and A. Yuille. Part and appearance sharing: Recursive compositional models for multi-view multi-object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

Journal of Machine Learning Research 17 (2016) 1-41 Submitted 8/13; Revised 1/15; Published 3/16

Noisy Sparse Subspace Clustering

Yu-Xiang Wang YUXIANGW@CS.CMU.EDU
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213

Huan Xu MPEXUH@NUS.EDU.SG
Department of Mechanical Engineering
National University of Singapore
Singapore 117576

Editor: Matthias Hein

Abstract

This paper considers the problem of subspace clustering under noise. Specifically, we study the behavior of Sparse Subspace Clustering (SSC) when either adversarial or random noise is added to the unlabeled input data points, which are assumed to be in a union of low-dimensional subspaces. We show that a modified version of SSC is *provably effective* in correctly identifying the underlying subspaces, even with noisy data. This extends theoretical guarantee of this algorithm to more practical settings and provides justification to the success of SSC in a class of real applications.

Keywords: Subspace clustering, robustness, stability, compressive sensing, sparse

1. Introduction

Subspace clustering is a problem motivated by many real applications. It is now widely known that many high dimensional data including motion trajectories (Costeira and Kanade, 1998), face images (Basri and Jacobs, 2003), network hop counts (Eriksson et al., 2012), movie ratings (Zhang et al., 2012) and social graphs (Chen et al., 2014) can be modeled as samples drawn from the *union* of multiple low-dimensional linear subspaces (illustrated in Figure 1). Subspace clustering, arguably the most crucial step to understand such data, refers to the task of clustering the data into their original subspaces and uncovering the underlying structure of the data. The partitions correspond to different rigid objects for motion trajectories, different people for face data, subnets for network data, like-minded users in movie database and latent communities for social graph.

Subspace clustering has drawn significant attention in the last decade and a great number of algorithms have been proposed, including Expectation-Maximization-like local optimization algorithms, e.g., K-plane (Bradley and Mangasarian, 2000) and Q-flat (Tseng, 2000), algebraic methods, e.g., Generalized Principal Component Analysis (GPCA) (Vidal et al., 2005), matrix factorization methods (Costeira and Kanade, 1998, 1995), spectral clustering-based methods (Lauer and Schnorr, 2009; Chen and Lerman, 2009), bottom-up local sampling and affinity-based methods (e.g., Yan and Pollefeys, 2006; Rao et al., 2008), and the convex optimization-based methods: namely, Low Rank Representation (LRR) (Liu et al., 2010, 2013) and Sparse Subspace Clustering (SSC) (Elhamifar and Vidal, 2009, 2013). For a comprehensive survey and comparisons, we refer the readers to the tutorial (Vidal, 2010). Among these algorithms, SSC is known to enjoy superb empirical performance, *even for noisy data*. For example, it is the state-of-the-art algorithm for motion segmentation on Hopkins155 benchmark (Tron and Vidal, 2007; Elhamifar and Vidal, 2009), and has been shown to be more robust than LRR as the number of clusters increase (Elhamifar and Vidal, 2013).

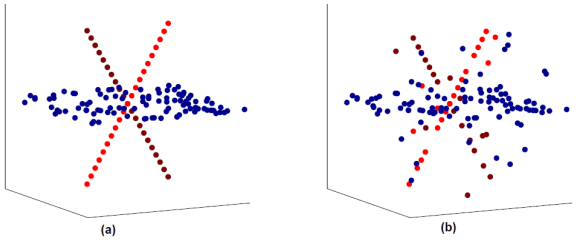


Figure 1: Exact (a) and noisy (b) data in union-of-subspace

The key idea of SSC is to represent each data point by a sparse linear combination of the remaining data points using ℓ_1 minimization. Without introducing the notations (which is deferred in Section 3), the noiseless and noisy versions of SSC solve respectively

$$\min_{c_i} \|c_i\|_1 \quad s.t. \quad x_i = X_{-i}c_i, \qquad \text{and} \qquad \min_{c_i} \|c_i\|_1 + \frac{\lambda}{2}\|x_i - X_{-i}c_i\|^2,$$

for each data column x_i, and the hope is that c_i will be supported only on indices of the data points from the same subspace as x_i. While this formulation is for linear subspaces, affine subspaces can also be dealt with by augmenting data points with an offset variable 1.

Effort has been made to explain the practical success of SSC by analyzing the noiseless version. Elhamifar and Vidal (2010) show that under certain conditions, *disjoint* subspaces (i.e., they are not overlapping) can be exactly recovered. A recent geometric analysis of SSC (Soltanolkotabi et al., 2012) broadens the scope of the results significantly to the case when subspaces can be overlapping. However, while these analyses advanced our understanding of SSC, one common drawback is that data points are assumed to be lying *exactly* on the subspaces. This assumption can hardly be satisfied in practice. For example, motion trajectories data are only *approximately* of rank-4 due to perspective distortion of camera, tracking errors and pixel quantization (Costeira and Kanade, 1998); similarly, face images are not precisely of rank-9 since human faces are at best *approximated* by a convex body (Basri and Jacobs, 2003).

In this paper, we address this problem and provide a theoretical analysis of SSC with noisy or corrupted data. Our main result shows that a modified version of SSC (see Eq. (3.2)) succeeds when the magnitude of noise does not exceed a threshold determined by a geometric gap between the *inradius* and the *subspace incoherence* (see below for precise definitions). This complements the result of Soltanolkotabi et al. (2012) that shows the

same geometric gap determines whether SSC succeeds for the noiseless case. Indeed, when the noise vanishes, our results reduce to the noiseless case results of Soltanolkotabi et al..

While our analysis is based upon the geometric analysis of Soltanolkotabi et al. (2012), the analysis is more involved: In SSC, sample points are used as the dictionary for sparse recovery, and therefore noisy SSC requires analyzing a noisy dictionary. We also remark that our results on noisy SSC are *exact*, i.e., as long as the noise magnitude is smaller than a threshold, the recovered subspace clusters are *correct*. This is in sharp contrast to the majority of previous work on structure recovery for noisy data where stability/perturbation bounds are given—i.e., the obtained solution is *approximately* correct, and the approximation gap goes to zero only when the noise diminishes.

Lastly, we remark that an independently developed work (Soltanolkotabi et al., 2014) analyzed the same algorithm *under a statistical model* that generates the data. In contrast, our main results focus on the cases when the data are deterministic. Moreover, when we specialize our general result to the same statistical model, we show that we can handle a significantly larger amount of noise under certain regimes.

The paper is organized as follows. In Section 2, we review previous and ongoing works related to this paper. In Section 3, we formally define the notations, explain our method and the models of our analysis. Then we present our main theoretical results in Section 4 with examples and remarks to explain the practical implications of each theorem. In Section 5 and 6, proofs of the deterministic and randomized results are provided. We then evaluate our method experimentally in Section 7 with both synthetic data and real-life data, which confirms the prediction of the theoretical results. Lastly, Section 8 summarizes the paper and discuss some open problems for future research in the task of subspace clustering.

2. Related works

In this section, we review previous and ongoing theoretical studies on the problem of subspace clustering.

2.1 Nominal performance guarantee for noiseless data

Most previous analyses concern about the nominal performance of a particular subspace clustering algorithm with noiseless data. The focus is to relax the assumptions on the underlying subspaces and data generation.

A number of methods have been shown working under the *independent subspace* assumption including the early factorization-based methods (Costeira and Kanade, 1998; Kanatani, 2001), LRR (Liu et al., 2010) and the initial guarantee of SSC (Elhamifar and Vidal, 2009). Recall that the data points are drawn from a union of subspaces, the *independent subspace* assumption requires each subspace to be linearly independent to the *span* of all other subspaces. Equivalently, this assumption requires the sum of each subspace's dimension to be equal to the dimension of the span of all subspaces. For example, in a two dimensional plane, one can only have 2 independent lines. If there are three lines intersecting at the origin, even if each pair of the lines are independent, they are not considered independent as a whole.

Independent Subspaces	$\dim[\mathcal{S}_1 \otimes ... \otimes \mathcal{S}_L] = \sum_{\ell=1}^{L} \dim[\mathcal{S}_\ell]$.
Disjoint Subspaces	$\mathcal{S}_\ell \cap \mathcal{S}_k = \mathbf{0}$ for all $\{(\ell, k) \vert \ell \neq k\}$.
Overlapping Subspaces	$\mathcal{S}_\ell \neq \mathcal{S}_k$ for all $\{(\ell, k) \vert \ell \neq k\}$ [1].

Table 1: Comparison of conditions on the underlying subspaces.

Disjoint subspace assumption only requires pairwise linear independence, and hence is more meaningful in practice. To the best of our knowledge, only GPCA (Vidal et al., 2005) and SSC (Elhamifar and Vidal, 2010, 2013) have been shown to provably handle the data under *disjoint subspace* assumption. GPCA however is not a polynomial time algorithm. Its computational complexity increases exponentially with respect to the number and dimension of the subspaces.

Soltanolkotabi et al. (2012) developed a geometric analysis that further extends the performance guarantee of SSC, and in particular it covers some cases when the underlying subspaces are *overlapping*, meaning that two subspaces can even share a basis. The analysis reveals that the success of SSC relies on the difference of two geometric quantities (inradius r and incoherence μ) to be greater than 0, which leads to by far the most general and strongest theoretical guarantee for noiseless SSC. A summary of these assumptions on the subspaces and their formal definition are given in Table 1.

We remark that our robust analysis extends from Soltanolkotabi et al. (2012) and therefore is inherently capable of handling the same range of problems, namely disjoint and overlapping subspaces. This is formalized later in Section 4.

2.2 Robust performance guarantee

Previous studies of the subspace clustering under noise have been mostly empirical. For instance, factorization, spectral clustering and local affinity based approaches, which we mentioned above, are able to produce a (sometimes good) solution even for noisy real data. Convex optimization based approaches like LRR and SSC can be naturally reformulated as a robust method by relaxing the hard equality constraints to a penalty term in the objective function. In fact, the superior results of SSC and LRR on motion segmentation and face clustering data are produced using the robust extension in Elhamifar and Vidal (2009) and Liu et al. (2010) instead of the well-studied noiseless version.

As of writing, there have been very few subspace clustering methods that is guaranteed to work when data are noisy. Besides the conference version of the current paper (Wang and Xu, 2013), an independent work (Soltanolkotabi et al., 2014) also analyzed SSC under noise. Subsequently, there has been noisy guarantees for other algorithms, e.g., thresholding based approach (Heckel and Bölcskei, 2013) and orthogonal matching pursuit (Dyer et al., 2013). The main difference between our work and Soltanolkotabi et al. (2014) is that our guarantee works for a more general set of problems when the data and noise may not be

1. "Overlapping subspace model" does not need any additional assumptions on subspaces, as long as the subspace membership for every data point is properly defined to resolve the identifiability issues. For instance, if data point $x \in \mathcal{S}_\ell \cap \mathcal{S}_k$, it is reasonable to assume x is from either $\mathcal{S}_\ell$ or $\mathcal{S}_k$; and SSC might be successful in clustering x to either $\mathcal{S}_\ell$ or $\mathcal{S}_k$, when their corresponding separation conditions hold. In fact, the separation conditions in Soltanolkotabi et al. (2012) could hold even if one subspace is completely contained in another.

	This paper	(Wang and Xu, 2013)	Soltanolkotabi et al. (2014)
Fully deterministic	$O(r(r-\mu))$	$O(r(r-\mu))$	N.A.
Deterministic + random noise	$O((n/d)^{\frac{1}{4}}(r-\mu))$	$O(r-\mu)$	N.A.
Semi-random data + random noise	$O\left(\frac{n^{\frac{1}{4}}}{\sqrt{d}}(1-\frac{\text{aff}}{\sqrt{d}})\right)$	$O\left(\frac{1}{\sqrt{d}}(1-\frac{\text{aff}}{\sqrt{d}})\right)$	$O\left(1-\frac{\text{aff}}{\sqrt{d}}\right)$
Fully-random data + random noise	$O\left(\frac{n^{\frac{1}{4}}}{\sqrt{d}}(1-\frac{\sqrt{d}}{\sqrt{n}})\right)$	$O\left(\frac{1}{\sqrt{d}}(1-\frac{\sqrt{d}}{\sqrt{n}})\right)$	$O\left(1-\frac{\sqrt{d}}{\sqrt{n}}\right)$

Table 2: Comparison of the level of noise tolerable for noisy subspace clustering methods. Note that "aff" is the *unnormalized* affinity defined in Soltanolkotabi et al. (2012)

.

random, whereas the key arguments in the proof in Soltanolkotabi et al. (2014) rely on the assumption that data points are uniformly distributed on the unit sphere within each subspace, which corresponds to the "semi-random model" in our paper. As illustrated in Elhamifar and Vidal (2013, Figure 9 and 10), the semi-random model is not a good fit for either the motion segmentation and the face clustering data sets, as in these data sets there is a fast decay in the singular values of each subspace. The uniform distribution assumption becomes even harder to justify as the dimension d of each subspace gets larger—a regime where the analysis in Soltanolkotabi et al. (2014) focuses on.

Moreover, with a minor modification in our analysis that sharpens the bound of the tuning parameter that ensures the solution is non-trivial, we are able to get a result that is stronger than Soltanolkotabi et al. (2014) in cases when the dimension of each subspace $d \leq O(\sqrt{n})$ [2]. This result extends the provably guarantee of SSC to a setting where the signal to noise ratio (SnR) is allowed to go to 0 as the ambient dimension gets large. In summary, we compare our results in terms of the level of noise that can be provably tolerated in Table 2. These comparisons are in the same setting modulo some slight differences in the noise model and successful criteria. It is worth noting that when $d > O(\sqrt{n})$, Soltanolkotabi et al. (2014)'s bound is sharper. We will provide more details in the Appendix.

Lastly, we note that the notion of robustness in this paper is confined to the noise/arbitrary corruptions added to the legitimate data. It is not the robustness against outliers in the data, unless otherwise specified. Handling outliers is a completely different problem. Solutions have been proposed for LRR in Liu et al. (2012) by decomposing a $\ell_{2,1}$ norm column-wise sparse components and for SSC in Soltanolkotabi et al. (2012) by objective value thresholding. However these results require non-outlier data points to be free of noise, therefore are not comparable to the study in this paper.

3. Problem setup

In this section, we specify the notations that will be use throughout the paper and explain the formal problem setup.

2. Admittedly, Soltanolkotabi et al. (2014) obtained better noise-tolerance than the comparable result in our conference version (Wang and Xu, 2013).

3.1 Notations

We denote the uncorrupted data matrix by $Y \in \mathbb{R}^{n\times N}$, where each column of Y (normalized to unit vector [3]) belongs to a union of L subspaces

$$\mathcal{S}_1 \cup \mathcal{S}_2 \cup ... \cup \mathcal{S}_L.$$

Each subspace $\mathcal{S}_\ell$ is of dimension d_ℓ and contains N_ℓ data samples with $N_1 + N_2 + ... + N_L = N$. We observe the noisy data matrix $X = Y + Z$, where Z is some arbitrary noise matrix. Let $Y^{(\ell)} \in \mathbb{R}^{n\times N_\ell}$ denote the selection of columns in Y that belongs to $\mathcal{S}_\ell$, and denote the corresponding columns in X and Z by $X^{(\ell)}$ and $Z^{(\ell)}$ respectively. Without loss of generality, let $X = [X^{(1)}, X^{(2)}, ..., X^{(L)}]$ be ordered. In addition, we use subscript "$-i$" to represent a matrix that excludes column i, e.g., $X^{(\ell)}_{-i} = [x^{(\ell)}_1, ..., x^{(\ell)}_{i-1}, x^{(\ell)}_{i+1}, ..., x^{(\ell)}_{N_\ell}]$. Calligraphic letters such as $\mathcal{X}, \mathcal{Y}_\ell$ represent the set containing all columns of the corresponding matrix (e.g., X and $Y^{(\ell)}$).

For any matrix X, $\mathcal{P}(X)$ represents the symmetrized convex hull of its columns, i.e., $\mathcal{P}(X) = \text{conv}(\pm\mathcal{X})$. Also let $\mathcal{P}^{(\ell)}_{-i} := \mathcal{P}(X^{(\ell)}_{-i})$ and $\mathcal{Q}^{(\ell)}_{-i} := \mathcal{P}(Y^{(\ell)}_{-i})$ for short. $\mathbb{P}_{\mathcal{S}}$ and $\text{Proj}_{\mathcal{S}}$ denote respectively the projection matrix and projection operator (acting on a set) to subspace $\mathcal{S}$. Throughout the paper, $\|\cdot\|$ represents 2-norm for vectors and operator norm for matrices; other norms will be explicitly specified (e.g., $\|\cdot\|_1, \|\cdot\|_\infty$).

3.2 Method and the criterion of success

Original SSC solves the linear program

$$\min_{c_i} \|c_i\|_1 \quad s.t. \quad x_i = X_{-i}c_i, \tag{3.1}$$

for each data point x_i. Solutions are arranged into matrix $C = [c_1, ..., c_N]$, then spectral clustering techniques such as Ng et al. (2002) are applied on the affinity matrix $W = |C| + |C|^T$ ($|\cdot|$ represents entrywise absolute value). Note that when $Z \neq 0$, this method breaks down: indeed (3.1) may even be infeasible.

To handle noisy X, a natural extension is to relax the equality constraint in (3.1) and solve the following unconstrained minimization problem instead (Elhamifar and Vidal, 2013):

$$\min_{c_i} \|c_i\|_1 + \frac{\lambda}{2}\|x_i - X_{-i}c_i\|^2. \tag{3.2}$$

We will focus on Formulation (3.2) in this paper. Notice that (3.2) coincides with standard LASSO. Yet, since our task is subspace clustering, the analysis of LASSO (mainly for the task of support recovery) does not extend to SSC. In particular, existing literature for LASSO to succeed requires the dictionary X_{-i} to satisfy the Restricted Isometry Property (RIP for short; Candès, 2008) or the Null-space property (Donoho et al., 2006), but neither of them is satisfied in the subspace clustering setup.[4]

3. We assume the normalization condition for ease of presentation. Our results can be extended to the case when each column of the noisy data points $X = Y + Z$ is normalized, as well as the case where no normalizing is performed at all, under simple modifications to the conditions.

4. As a simple illustrative example, suppose X obeys RIP and x be the first column of X, matrix $[X, x]$ violates RIP for 2-sparse signal $(1, 0, ..., 0, -1)$. On the contrary, the inradius of X and $[X, x]$ will be exactly the same.

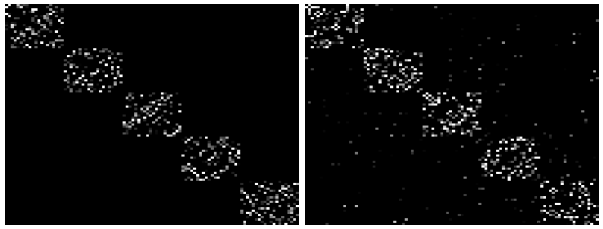


Figure 2: Illustration of LASSO-Subspace Detection Property/Self-Expressiveness Property. **Left:** SEP holds. **Right:** SEP is violated even though spectral clustering is likely to cluster this affinity graph perfectly into 5 blocks.

In the subspace clustering task, there is no single "ground-truth" C to compare the solution against. Instead, the algorithm succeeds if each sample is expressed as a linear combination of samples belonging to the same subspace, as the following definition states.

Definition 1 (LASSO Subspace Detection Property) *We say the subspaces $\{\mathcal{S}_\ell\}_{\ell=1}^k$ and noisy sample points X from these subspaces obey LASSO subspace detection property with parameter λ, if and only if it holds that for all i, the optimal solution c_i to* (3.2) *with parameter λ satisfies:*

(1) c_i is not a zero vector, i.e., the solution is non-trivial, (2) Nonzero entries of c_i correspond to only columns of X sampled from the same subspace as x_i.

This property ensures that the output matrix C and (naturally) the affinity matrix W are exactly block diagonal with each subspace cluster represented by a disjoint block. The property is illustrated in Figure 2. For convenience, we will refer to the second requirement alone as "*Self-Expressiveness Property*" (SEP), as defined in Elhamifar and Vidal (2013).

Note that the LASSO Subspace Detection Property is a strong condition. In practice, spectral clustering does not require the exact block diagonal structure for perfect segmentation (check Figure 8b in our simulation section for details). A caveat is that it is also not sufficient for perfect segmentation, since it does not guarantee each diagonal block forms a connected component. This is a known problem for SSC (Nasihatkon and Hartley, 2011), although we observe that in practice graph connectivity is usually not a big issue. Proving the high-confidence connectivity (even under probabilistic models) remains an open problem, except for the almost trivial cases when the subspaces are independent (Liu et al., 2013; Wang et al., 2013).

3.3 Models of analysis:

Our objective here is to provide sufficient conditions upon which the LASSO subspace detection properties hold in the four models given in Table 2, ranging from fully deterministic to fully random. Precise definitions of these models will be given in Section 4.

4. Main results

In this section, we present our theoretical guarantee for LASSO-SSC under the four aforementioned models. We will start by describing subspace clustering for a fixed dataset where each data point can be perturbed by an arbitrary noise of bounded size. Then we will incrementally add stochastic assumptions and state the corresponding results with conditions that reveal explicit dependence on parameters of the model.

4.1 Deterministic model

We start by defining two concepts adapted from the original proposal of Soltanolkotabi et al. (2012).

Definition 2 (Projected Dual Direction) *Let ν be the optimal solution to the dual optimization program*[5]

$$\max_{\nu} \langle x, \nu \rangle - \frac{1}{2\lambda}\nu^T\nu, \quad \textit{subject to:} \quad \|X^T\nu\|_\infty \leq 1;$$

and $\mathcal{S}$ is a low-dimensional subspace. The projected dual direction *v is defined as*

$$v(x, X, \mathcal{S}, \lambda) \triangleq \frac{\mathbb{P}_{\mathcal{S}}\nu}{\|\mathbb{P}_{\mathcal{S}}\nu\|}.$$

Definition 3 (Projected Subspace Incoherence Property) *Compactly denote projected dual direction $v_i^{(\ell)} = v(x_i^{(\ell)}, X_{-i}^{(\ell)}, \mathcal{S}_\ell, \lambda)$ and $V^{(\ell)} = [v_1^{(\ell)}, ..., v_{N_\ell}^{(\ell)}]$. We say that vector set $\mathcal{X}_\ell$ is μ-incoherent to other points if*

$$\mu \geq \mu(\mathcal{X}_\ell) := \max_{y \in \mathcal{Y}\setminus\mathcal{Y}_\ell} \|{V^{(\ell)}}^T y\|_\infty.$$

Here, μ measures the incoherence between corrupted subspace samples $\mathcal{X}_\ell$ and clean data points in other subspaces (illustrated in Figure 4). As $\|y\| = 1$ by the normalization assumption, the range of μ is $[0, 1]$. In case of random subspaces in high dimension, μ is close to zero. Moreover, as we will see later, for deterministic subspaces and random data points, μ is proportional to their expected angular distance (measured by cosine of canonical angles).

Definition 2 and 3 differ from the *dual direction* and *subspace incoherence property* of Soltanolkotabi et al. (2012) in that we require a projection to a particular subspace to cater to the analysis of the noise case. Also, since they reduce to the original definitions when data are noiseless and $\lambda \to \infty$, these definitions can be considered as a generalization of their original version.

Definition 4 (inradius) *The inradius of a convex body $\mathcal{P}$, denoted by $r(\mathcal{P})$, is defined as the radius of the largest Euclidean ball inscribed in $\mathcal{P}$.*

The inradius of a $\mathcal{Q}_{-i}^{(\ell)}$ describes the dispersion of the data points. Well-dispersed data lead to larger inradius and skewed/concentrated distribution of data have small inradius. An illustration is given in Figure 3.

5. This definition is related to (5.3), the dual problem of (3.2), which we will define in the proof.

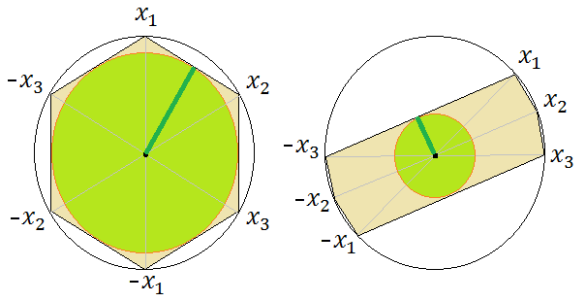


Figure 3: Illustration of inradius and data distribution. The inradius measures how well data points represent a subspace.

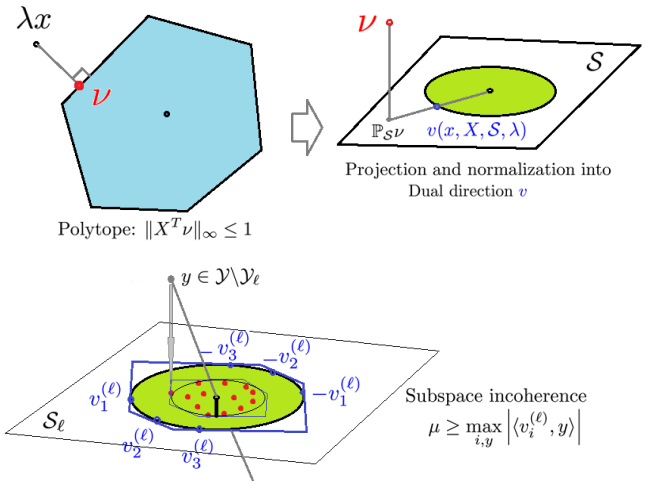


Figure 4: Illustrations of the projected dual direction and subspace incoherence property. The projected dual direction in Definition 2 is essentially an Euclidean projection to the polytope, followed by a projection to the subspace and normalization. There is a dual direction associated with each data point in the subspace. Jointly, $\left\{x \middle| \max_i \left|\langle v_i^{(\ell)}, x\rangle\right| \leq \mu\right\}$ defines a polygon in the subspace $\mathcal{S}_\ell$, and subspace incoherence μ is given by the smallest such polytope that contains the projections of all external point y into the this subspace.

Definition 5 (Deterministic noise model) *Consider arbitrary additive noise Z to Y, each column z_i is bounded by the two quantities below:*

$$\delta := \max_i \|z_i\|, \qquad \delta_1 := \max_{i,\ell} \|\mathbb{P}_{\mathcal{S}_\ell} z_i\|,$$

As we assume the uncorrupted data point y has unit norm, δ essentially describes the amount of allowable relative error.

Theorem 6 *Under the deterministic noise model, compactly denote*

$$\mu_\ell := \mu(\mathcal{X}_\ell), \qquad r_\ell := \min_{\{i:x_i\in\mathcal{X}_\ell\}} r(\mathcal{Q}_{-i}^{(\ell)}), \qquad r := \min_{\ell=1,\ldots,L} r_\ell.$$

If $\mu_\ell < r_\ell$ for each $\ell = 1, ..., L$, furthermore

$$\delta \leq \min_{\ell=1,\ldots,L} \frac{r(r_\ell - \mu_\ell)}{2 + 7r_\ell}$$

then LASSO subspace detection property holds for all weighting parameter λ in the range

$$\frac{1}{r - 2\delta - \delta^2} < \lambda < \min_{\ell=1,\ldots,L} \left\{ \frac{r_\ell - \mu_\ell - 2\delta_1}{\delta(1+\delta)(2 + r_\ell - \delta_1)} \right\}$$

which is guaranteed to be non-empty.

We now offer some discussions of the theorem and the proof will be given in Section 5.

Noiseless case. When $\delta = 0$, i.e., there is no noise, the condition reduces to $\mu_\ell < r_\ell$, which coincides with the result in Soltanolkotabi et al. (2012). The exact LP formulation (3.1) is equivalent to $\lambda \to \infty$. Our result implies that unconstrained LASSO formulation (3.2) works for any $\lambda > \frac{1}{r}$.

Signal-to-Noise Ratio. Condition $\delta \leq \frac{r(r-\mu)}{2+7r}$ can be interpreted as the breaking point under increasing magnitude of attack. This suggests that SSC by (3.2) is provably robust to arbitrary noise having signal-to-noise ratio (SnR) greater than $\Theta\left(\frac{1}{r(r-\mu)}\right)$. (Notice that $0 < r < 1$, and hence $7r + 2 = \Theta(1)$.)

Tuning parameter λ. The range of the parameter λ in the theorem depends on unknown parameters μ, r and δ, and therefore cannot be used in practice to choose the parameter in practice. It does however justify that when δ is small, the range of λ that LASSO-SSC works is large, therefore not hard to tune. In practice, we do not need to know λ in prior. One approach is to trace the Lasso path (Tibshirani et al., 2013) until we have about k non-zero entries in the coefficient vector. If we would like to use a single λ for all columns, a good point to start is to take λ to be in the order of $O\left(\frac{1}{\min_j \max_{i\neq j} |x_i^T x_j|}\right)$, this ensures the solution to be at least non-trivial.

Agnostic subspace clustering. The robustness to deterministic error is important, since in practice the union-of-subspace structures are usually only good approximations.

If each subspace has decaying singular values (e.g., motion segmentation, face clustering (Elhamifar and Vidal, 2013) and hybrid system identification(Vidal et al., 2003)), the deterministic guarantee allows for the flexibility in choosing the cut-off points, e.g., take 90% of the energy as signal and treat the remaining spectrum as noise. If one keeps a smaller number of singular values (a smaller subspace dimension), the inradius will likely to be larger [6], although the noise level also increases. It is possible that the conditions in Theorem 6 are satisfied for some decomposition (e.g., those with a large spectral gap) but not others. The nice thing is that this is not a tuning parameter, but rather a theoretical property that remains agnostic to the users. In fact, the algorithm will be provably effective as long as the conditions are satisfied for any signal noise decomposition (not restricted to rank-projection). None of these is possible if distributional assumptions are made to either the data or the noise.

4.2 Randomized models

We further analyze three randomized models with increasing level of randomness.

- **Deterministic+Random Noise.** Subspaces and samples in subspace are arbitrary; the noise obeys the Random Noise model (Definition 7).
- **Semi-random+Random Noise.** Subspace is deterministic, but samples in each subspace are drawn iid uniformly from the intersection of the unit sphere and the subspace; the noise obeys the Random Noise model.
- **Fully random.** Both subspace and samples are drawn uniformly at random from their respective domains; the noise is iid Gaussian.

In each of these models, we improve the performance guarantee over our conference version (Wang and Xu, 2013). In the most well-studied semi-random model, we are able to handle cases where the noise level is much larger than the signal, and hence improves upon the best known result for SSC Soltanolkotabi et al. (2014). A detailed comparison of the noise tolerance of these methods is given in Table 2.

Definition 7 (Random noise model) *Our random noise model is defined to be any additive Z that is (1) column wise iid; (2) spherical symmetric; and (3) $\|z_i\| \leq \delta$ for all $i = 1, ..., N$ with probability at least $1 - 1/N$.*

A good example of our random noise model is iid Gaussian noise. Let each entry $Z_{ij} \sim N(0, \sigma^2/n)$. It is known that (see Lemma 18) for some constant C

$$\mathbb{P}\left(\delta := \max_i \|z_i\| > \sqrt{1 + \frac{6\log N}{n}}\sigma\right) \leq C/N^2.$$

Theorem 8 (Deterministic+Random Noise) *Under random noise model, compactly denote r_ℓ, r and μ_ℓ as in Theorem 6, furthermore let*

$$\epsilon := \sqrt{\frac{6\log N}{n - \max_\ell d_\ell}} \leq \sqrt{\frac{C\log(N)}{n}}.$$

6. A formal relationship between inradius and smallest singular value is described in Wang et al. (2013).

If $\mu_\ell < r_\ell$ for all $\ell = 1, ..., k$,

$$\epsilon\delta < \min_{\ell=1,..,L} \frac{r_\ell - \mu_\ell}{2\sqrt{d_\ell} + 2}, \qquad \text{and} \qquad \epsilon\delta(1+\delta) < \min_{\ell=1,..,L} \frac{r(r_\ell - \mu_\ell)}{4r_\ell + 6},$$

then with probability at least $1 - 9/N$, LASSO subspace detection property holds for all weighting parameter λ in the range

$$\frac{1}{r - 2\epsilon\delta - \epsilon\delta^2} < \lambda < \min_{\ell=1,..,L} \left\{ \frac{r_\ell - \mu_\ell - \delta\epsilon - \delta\sqrt{d_\ell}\epsilon}{\epsilon\delta(1+\delta)(3 + r_\ell - \delta\sqrt{d_\ell}\epsilon)} \right\} \tag{4.1}$$

which is guaranteed to be non-empty.

Low SnR paradigm. Compared to Theorem 6, Theorem 8 considers a more benign noise which leads to a stronger result. In particular, without assuming any statistical model on how data are generated, we show that LASSO-SSC is able to tolerate noise of level $O\left((\frac{n}{\log N})^{1/4}(r(r_\ell - \mu_\ell))^{1/2}\right)$ or $O\left((\frac{n}{d\log N})^{1/2}(r_\ell - \mu_\ell)\right)$ (whichever is smaller). This extends SSC's guarantee with deterministic data to cases where the noise can be significantly larger than the signal. In fact, the SnR can go to 0 as the ambient dimension gets large.

On the other hand, Theorem 8 shows that LASSO-SSC is able to tolerate a constant level of noise when the geometric gap $r_\ell - \mu_\ell$ is as small as $O(\sqrt{d/n})$. This is arguably near-optimal (when d is small) as the projection of a constant-level random noise into a d-dimensional subspace has an expected magnitude of the same order, which could easily close up the small geometric gap for some non-trivial probability if the noise is much larger.

Margin of error. Since the bound depends critically on $(r_\ell - \mu_\ell)$—the difference of inradius and incoherence—which is the geometric gap that appears in the noiseless guarantee of Soltanolkotabi et al. (2012). We will henceforth call this gap the *margin of error*.

We now analyze this margin of error under different generative models. We start from the semi-random model, where the distance between two subspaces is measured as follows.

Definition 9 *The* affinity *between two subspaces is defined by:*

$$\text{aff}(\mathcal{S}_k, \mathcal{S}_\ell) = \sqrt{\cos^2\theta_{k\ell}^{(1)} + ... + \cos^2\theta_{k\ell}^{(\min(d_k,d_\ell))}},$$

where $\theta_{k\ell}^{(i)}$ is the i^{th} canonical angle between the two subspaces. Let U_k and U_ℓ be a set of orthonormal bases of each subspace, then $\text{aff}(\mathcal{S}_k, \mathcal{S}_\ell) = \|U_k^T U_\ell\|_F$.

When data points are randomly sampled from each subspace, the geometric entity $\mu(\mathcal{X}_\ell)$ can be expressed using this (more intuitive) subspace affinity, which leads to the following theorem.

Theorem 10 (Semi-random model+random noise) *Under the semi-random model with random noise, there exists a non-empty range of λ such that LASSO subspace detection property holds with probability $1 - \frac{9}{N} - \frac{1}{L^2}\sum_{\ell\neq\ell'}\frac{1}{(N_\ell+1)N_{\ell'}}e^{-\frac{t}{4}} - 6\sum_{\ell=1}^{L}(e^{\gamma_1(n-d_\ell)} + e^{\gamma_2 d_\ell} + e^{-\sqrt{N_\ell d_\ell}})$*

as long as the noise level obeys

$$\delta(1+\delta) \le \max_{\ell,\ell'} \sqrt{\frac{n-d}{6\log N}} \frac{\sqrt{\log\kappa}}{40K_2\sqrt{dd_\ell}} \left(1 - \frac{K_1K_2\mathrm{aff}(\mathcal{S}_\ell,\mathcal{S}_{\ell'})}{\sqrt{d_{\ell'}}}\right),$$

where $K_1 := (t\log[(N_\ell+1)N_{\ell'}] + \log L)$, $K_2 := 4\sqrt{\frac{1}{\log\kappa_\ell}}$, $\kappa_\ell := N_\ell/d_\ell$, $\frac{\log\kappa}{d} := \min_\ell \frac{\log\kappa_\ell}{d_\ell}$, *and* γ_1, γ_2 *are absolute constants.*

The proof is essentially substituting the incoherence and inradius parameters in Theorem 8 with meaningful bounds, so Theorem 10 can be regarded as a corollary of Theorem 8.

Overlapping subspaces. Similar to the results in Soltanolkotabi et al. (2012), Theorem 10 demonstrates that LASSO-SSC can handle overlapping subspaces with noisy samples. By Definition 9, $\mathrm{aff}(\mathcal{S}_k, \mathcal{S}_\ell)$ can be small even if $\mathcal{S}_k$ and $\mathcal{S}_\ell$ share a basis.

Comparison to Soltanolkotabi et al. (2014). In the high dimensional setting when $n \gg d$, our result is able to handle the low SnR regime when $\delta = \Theta(n^{1/4}/d^{1/2})$, while Soltanolkotabi et al. (2014) needs δ to be bounded by a small constant.

In the case when d is a constant fraction of n, however, our bound is worse by a factor of $\sqrt{d}$. Soltanolkotabi et al. (2014) is still able to handle a small constant noise while we needs $\delta < O(\frac{1}{\sqrt{d}})$. The suboptimal bound might be due to the fact that we are simply developing the theorem for the semi-random model as a corollary of Theorem 8 and haven not fully exploit the structure of the semi-random model in the proof.

We now turn to the fully random case.

Theorem 11 (Fully random model) *Suppose there are L subspaces each with dimension d, chosen independently and uniformly at random. For each subspace, $\kappa d + 1$ points are chosen independently and uniformly from the unit sphere inside each subspace. Each measurement is corrupted by iid Gaussian noise $\sim N(0, \sigma^2/n)$. Furthermore, if*

$$d < \frac{c(\kappa)^2\log\kappa}{24\log N} n, \qquad \text{and} \qquad \sigma(1+\sigma) < \frac{c(\kappa)^2\log\kappa}{20}\frac{\sqrt{n}}{d},$$

then with probability at least $1 - \frac{10}{N} - Ne^{-\sqrt{\kappa}d}$, the LASSO subspace detection property holds for any λ in the range

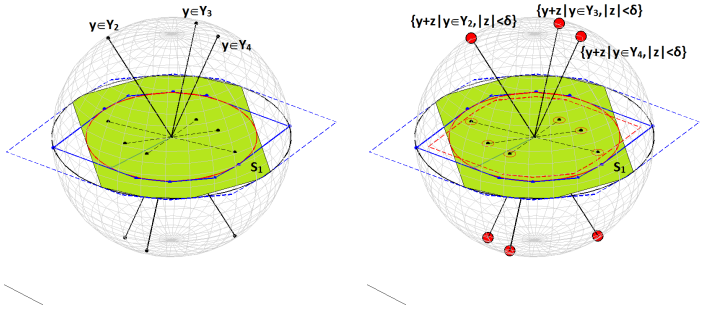
$$\frac{C_1\sqrt{d}}{c(\kappa)\sqrt{\log\kappa}} < \lambda < \frac{C_2 c(\kappa)\sqrt{n\log\kappa}}{\sigma\sqrt{d\log N}}, \tag{4.2}$$

which is guaranteed to be non-empty. Here, C_1, C_2 are absolute constants.

The results under this simple model are very interpretable. It provides intuitive guideline in how robustness of LASSO-SSC change with respect to the various parameters of the data. One one hand, it is sensitive to the dimension of each subspace d, since the $\sigma \le \tilde{\Theta}(\frac{n^{1/4}}{\sqrt{d}})$. This dependence on subspace dimension d is not a critical limitation as most interesting applications indeed have very low subspace-dimension, as summarized in Table 3. On the other hand, the dependence on the number of subspaces L (in both $\log\kappa$ and $\log N$ since $N = L(\kappa d + 1)$) is only logarithmic. This suggests that SSC is robust even when there are many clusters, and $Ld \gg n$.

Application	Cluster rank
3D motion segmentation (Costeira and Kanade, 1998)	rank = 4
Face clustering (with shadow) (Basri and Jacobs, 2003)	rank = 9
Diffuse photometric face (Zhou et al., 2007)	rank = 3
Network topology discovery (Eriksson et al., 2012)	rank = 2
Hand writing digits (Hastie and Simard, 1998)	rank = 12
Social graph clustering (Chen et al., 2014)	rank = 1

Table 3: Rank of real subspace clustering problems



(a) Noiseless SSC (b) Noisy LASSO-SSC

Figure 5: Geometric interpretation and comparison of the **(a)** noiseless SSC and **(b)** noisy LASSO-SSC.

4.3 Geometric interpretations

A geometric illustration of the condition in Theorem 6 is given in Figure 5 in comparison to the geometric separation condition in the noiseless case.

The left pane depicts the separation condition $\mu_\ell \le r_\ell$ in Theorem 2.5 of Soltanolkotabi et al. (2012). The blue polygon represents the the intersection of halfspaces defined with dual directions that are also the tangent to the red inscribing sphere. More precisely, this is $\left\{x \in \mathcal{S}_\ell \middle| \middle|\langle v_i^\ell, x\rangle\right| \le r_\ell\}$. From our illustration of μ in Figure 4, we can easily tell that $\mu_\ell \le r_\ell$ if and only if the projection of external data points fall inside this solid blue polygon. We call this solid blue polygon the successful region.

The right pane illustrates our guarantee of Theorem 6 under bounded deterministic noise. The successful condition requires that the whole red ball (analogous to uncertainty set in Robust Optimization (Ben-Tal and Nemirovski, 1998; Bertsimas and Sim, 2004)) around each external data point to fall inside the dashed red polygon, which is smaller than the blue polygon by a factor related to the noise level and the inradius.

The successful region is affected by the noise because the design matrix is also arbitrarily perturbed and the dual solution is no longer within each subspace $\mathcal{S}_\ell$. Specifically, as will

become clear in the proof, the key of showing SEP boils down to proving $\langle \nu_i^{(\ell)}, x_j \rangle < 1$ for all pairs of $(\nu_i^{(\ell)}, x_j)$ where

$$\nu_i^{(\ell)} = \arg\max_{\nu} \langle \nu, x_i^{(\ell)} \rangle - \frac{1}{2\lambda}\|\nu\|^2 \text{ s.t. } \|\nu^T X_{-i}^{(\ell)}\|_\infty \le 1,$$

and x_j is any point from another subspace. In the noiseless case we can always take $\nu_i^{(\ell)} \in \mathcal{S}_\ell$ and $\langle \nu_i^{(\ell)}, x_j \rangle \le \frac{\mu_\ell}{r_\ell}$. For noisy data and LASSO-SSC, we can no longer do that. In fact, for any fixed λ, the dual solution will be uniquely determined by a projection of $\lambda x_i^{(\ell)}$ on to the feasible region $\|\nu^T X_{-i}^{(\ell)}\|_\infty \le 1$ (see the first pane of Figure 4). The absolute value of the inner product $\langle \nu_i^{(\ell)}, x_j \rangle$ will depend on the magnitude of the dual solution, especially its component perpendicular to the current subspace. Indeed by carefully choosing the error, we can make $\mathbb{P}_{\mathcal{S}_\ell^\perp}\nu$ very correlated with some external data point x_j.

To illustrate this further, we plot the shape of this feasible region in 3D (see Figure 6(b)). From the feasible region alone, it seems that the magnitude of dual variable can potentially be quite large. Luckily, the quadratic penalty in the objective function allows us to exploit the optimality of the solution ν and bound the "out-of-subspace" component of ν, which results in a much smaller region where the solution can potentially be (given in Figure 6(c)). The region for the "in-subspace" component is also smaller as is shown in Figure 7. A more detailed argument of this is given in Section 5.3 of the proof.

Admittedly, the geometric interpretation under noise is slightly messier than the noiseless case, but it is clear that the largest deterministic noise LASSO-SSC can tolerate must be smaller than geometric gap $r_\ell - \mu_\ell$. Theorem 6 show that a sufficient condition is $\delta \le O(r(r_\ell - \mu_\ell))$. It remains unclear whether this gap can be closed without additional assumptions.

Finally, we note that for the random noise model in Theorem 8, the geometric interpretation is similar, except that the impact of the noise is weakened. Thanks to the randomness and the corresponding concentration of measure, we may bound the reduction of the successful region with a much smaller value comparing to the adversarial noise case.

5. Proof of the Deterministic Result

In this section, we provide the proof for Theorem 6.

Instead of analyzing (3.2) directly, we consider an equivalent constrained version by introducing slack variable e_i:

$$\mathbf{P}_0: \quad \min_{c_i, e_i} \|c_i\|_1 + \frac{\lambda}{2}\|e_i\|^2 \quad s.t. \quad x_i^{(\ell)} = X_{-i}c_i + e_i. \tag{5.1}$$

The constraint can be rewritten as

$$y_i^{(\ell)} + z_i^{(\ell)} = (Y_{-i} + Z_{-i})c_i + e_i. \tag{5.2}$$

The dual program of (5.1) is:

$$\mathbf{D}_0: \quad \max_{\nu} \langle x_i, \nu \rangle - \frac{1}{2\lambda}\nu^T\nu \quad s.t. \quad \|(X_{-i})^T\nu\|_\infty \le 1. \tag{5.3}$$

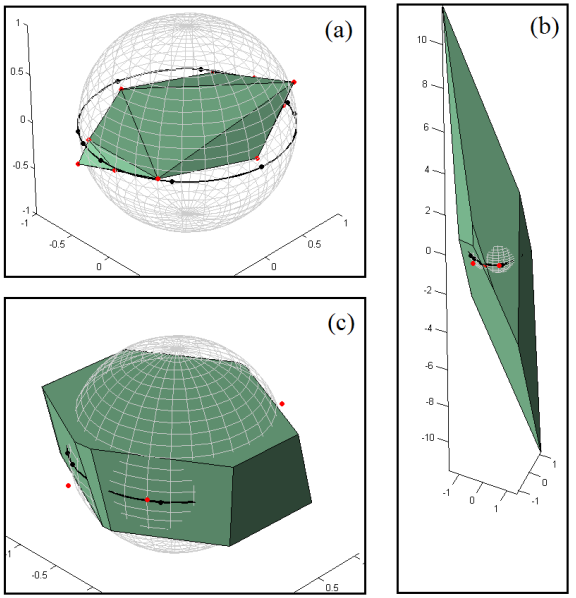


Figure 6: Illustration of **(a)** the convex hull of noisy data points, **(b)** its polar set and **(c)** the intersection of polar set and $\|\nu_2\|$ bound. The polar set (b) defines the feasible region of (5.7). It is clear that ν_2 can take very large value in (b) if we only consider feasibility. By considering optimality, we know the optimal ν must be inside the region in (c).

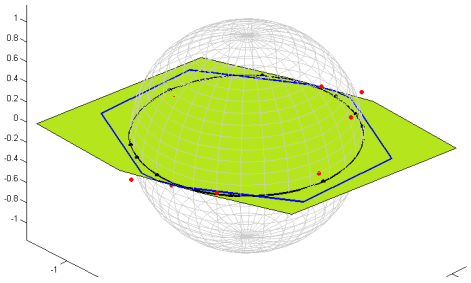


Figure 7: The projection of the polar set (the green area) in comparison to the projection of the polar set with $\|\nu_2\|$ bound (the blue polygon). It is clear that the latter is much smaller.

Recall that we want to establish the conditions on noise magnitude δ, structure of the data (μ and r in the deterministic model and affinity in the semi-random model), and ranges of valid λ such that by Definition 1, the solution c_i is *non-trivial* and has support indices inside the column set $X_{-i}^{(\ell)}$ (i.e., satisfies SEP).

The proof is hence organized into three main steps:

(1) Proving SEP by duality. First we establish a set of conditions on the optimal dual variable of D_0 corresponding to all primal solutions satisfying SEP. Then we construct such a dual variable ν as a certificate of proof. This is presented in Section 5.1, 5.2 and 5.3.

(2) Proving non-trivialness by showing that the optimal value is smaller than the value of the trivial solution (i.e., $c^* = 0$ and $e^* = x_i^{(\ell)}$). This step is given in Section 5.4.

(3) Showing the existence of a proper λ. As it will be made clear later, conditions for (1) include $\lambda < A$ and (2) requires $\lambda > B$ for some expression A and B. Then it is natural to request $B < A$, so that a valid λ exists. It turns out that this condition boils down to $\delta < C$ for some expression C. This argument is carried over in Section 5.5.

5.1 Optimality Condition

Consider a general convex optimization problem:

$$\min_{c,e} \|c\|_1 + \frac{\lambda}{2}\|e\|^2 \qquad s.t. \quad x = Ac + e. \tag{5.4}$$

We state Lemma 12, which extends Lemma 7.1 in Soltanolkotabi et al. (2012).

Lemma 12 *Consider a vector $y \in \mathbb{R}^d$ and a matrix $A \in \mathbb{R}^{d\times N}$. If there exists a triplet (c, e, ν) obeying $y = Ac + e$ and c has support $S \subseteq T$, furthermore the dual certificate vector ν satisfies*

$$A_s^T\nu = \mathrm{sgn}(c_S), \quad \nu = \lambda e, \quad \|A_{T\cap S^c}^T\nu\|_\infty \le 1, \quad \|A_{T^c}^T\nu\|_\infty < 1,$$

then any optimal solution (c^*, e^*) to (5.4) obeys $c^*_{T^c} = 0$.

Proof For optimal solution (c^*, e^*), we have:

$$\begin{aligned}
&\|c^*\|_1 + \frac{\lambda}{2}\|e^*\|^2 \\
=&\|c^*_S\|_1 + \|c^*_{T\cap S^c}\|_1 + \|c^*_{T^c}\|_1 + \frac{\lambda}{2}\|e^*\|^2 \\
\ge&\|c_S\|_1 + \langle \mathrm{sgn}(c_S), c^*_S - c_S\rangle + \|c^*_{T\cap S^c}\|_1 + \|c^*_{T^c}\|_1 + \frac{\lambda}{2}\|e\|^2 + \langle \lambda e, e^* - e\rangle \\
=&\|c_S\|_1 + \langle \nu, A_S(c^*_S - c_S)\rangle + \|c^*_{T\cap S^c}\|_1 + \|c^*_{T^c}\|_1 + \frac{\lambda}{2}\|e\|^2 + \langle \nu, e^* - e\rangle \\
=&\|c_S\|_1 + \frac{\lambda}{2}\|e\|^2 + \|c^*_{T\cap S^c}\|_1 - \langle \nu, A_{T\cap S^c}(c^*_{T\cap S^c})\rangle + \|c^*_{T^c}\|_1 - \langle \nu, A_{T^c}(c^*_{T^c})\rangle.
\end{aligned} \tag{5.5}$$

To see $\frac{\lambda}{2}\|e^*\|^2 \ge \frac{\lambda}{2}\|e\|^2 + \langle \lambda e, e^* - e\rangle$, note that the right hand side equals to $\lambda\left(-\frac{1}{2}e^Te + (e^*)^Te\right)$, which takes a maximal value of $\frac{\lambda}{2}\|e^*\|^2$ when $e = e^*$. The last equation holds because both

(c, e) and (c^*, e^*) are feasible solution, such that $\langle \nu, A(c^* - c)\rangle + \langle \nu, e^* - e\rangle = \langle \nu, Ac^* + e^* - (Ac + e)\rangle = 0$. Also, note that $\|c_S\|_1 + \frac{\lambda}{2}\|e\|^2 = \|c\|_1 + \frac{\lambda}{2}\|e\|^2$.

With the inequality constraints of ν given in the lemma statement, we have

$$\langle \nu, A_{T\cap S^c}(c^*_{T\cap S^c})\rangle = \langle A^T_{T\cap S^c}\nu, (c^*_{T\cap S^c})\rangle \le \|A^T_{T\cap S^c}\nu\|_\infty \|c^*_{T\cap S^c}\|_1 \le \|c^*_{T\cap S^c}\|_1.$$

Substitute into (5.5), we get:

$$\|c^*\|_1 + \frac{\lambda}{2}\|e^*\|^2 \ge \|c\|_1 + \frac{\lambda}{2}\|e\|^2 + (1 - \|A^T_{T^c}\nu\|_\infty)\|c^*_{T^c}\|_1,$$

where $(1 - \|A^T_{T^c}\nu\|_\infty)$ is strictly greater than 0.

Using the fact that (c^*, e^*) is an optimal solution, $\|c^*\|_1 + \frac{\lambda}{2}\|e^*\|^2 \le \|c\|_1 + \frac{\lambda}{2}\|e\|^2$. Therefore, $\|c^*_{T^c}\|_1 = 0$ and (c, e) is also an optimal solution. This concludes the proof. ■

The next step is to apply Lemma 12 with $x = x_i^{(\ell)}$ and $A = X_{-i}$ and then construct a triplet (c, e, ν) such that dual certificate ν satisfying all conditions and c satisfies SEP. Then we can conclude that all optimal solutions of (5.1) satisfy SEP.

5.2 Construction of Dual Certificate

To construct the dual certificate, we consider the following *fictitious* optimization problem (and its dual) that explicitly requires that all feasible solutions satisfy SEP[7] (note that one can not solve such problem in practice without knowing the subspace clusters, and hence the name "fictitious").

$$\mathbf{P}_1: \quad \min_{c_i^{(\ell)}, e_i} \|c_i^{(\ell)}\|_1 + \frac{\lambda}{2}\|e_i\|^2 \quad s.t. \quad y_i^{(\ell)} + z_i = (Y_{-i}^{(\ell)} + Z_{-i}^{(\ell)})c_i^{(\ell)} + e_i; \tag{5.6}$$

$$\mathbf{D}_1: \quad \max_{\nu} \langle x_i^{(\ell)}, \nu\rangle - \frac{1}{2\lambda}\nu^T\nu \quad s.t. \quad \|(X_{-i}^{(\ell)})^T\nu\|_\infty \le 1. \tag{5.7}$$

This optimization problem is feasible because $y_i^{(\ell)} \in span(Y_{-i}^{(\ell)}) = \mathcal{S}_\ell$ so any $c_i^{(\ell)}$ obeying $y_i^{(\ell)} = Y_{-i}^{(\ell)}c_i^{(\ell)}$ and corresponding $e_i = z_i - Z_{-i}^{(\ell)}c_i^{(\ell)}$ is a pair of feasible solution. Then by strong duality, the dual program is also feasible, which implies that for every optimal solution (c, e) of (5.6) with c supported on S, there exist ν satisfying:

$$\left\{\begin{array}{c} \|((Y_{-i}^{(\ell)})^T_{S^c} + (Z_{-i}^{(\ell)})^T_{S^c})\nu\|_\infty \le 1, \quad \nu = \lambda e, \\ ((Y_{-i}^{(\ell)})^T_S + (Z_{-i}^{(\ell)})^T_S)\nu = \mathrm{sgn}(c_S). \end{array}\right\}$$

This construction of ν satisfies all conditions in Lemma 12 with respect to

$$\begin{cases} c_i = [0, ..., 0, c_i^{(\ell)}, 0, ..., 0] \text{ with } c_i^{(\ell)} = c, \\ e_i = e, \end{cases} \tag{5.8}$$

except

$$\left\|[X_1, ..., X_{\ell-1}, X_{\ell+1}, ..., X_L]^T\nu\right\|_\infty < 1,$$

7. To be precise, it is the corresponding $c_i = [0, ..., 0, (c_i^{(\ell)})^T, 0, ..., 0]^T$ that satisfies SEP.

i.e., we must check for all data point $x \in \mathcal{X} \setminus \mathcal{X}^\ell$,

$$|\langle x, \nu \rangle| < 1. \tag{5.9}$$

Thus, if we show that the solution of (5.7) ν also satisfies (5.9), we can conclude that ν is a dual certificate required in Lemma 12, which implies that the candidate solution (5.8) associated with optimal (c, e) of (5.6) is indeed the optimal solution of (5.1) and therefore SEP holds.

5.3 Dual separation condition

Our strategy to show (5.9) is to provide an upper bound of $|\langle x, \nu \rangle|$ then impose the inequality on the upper bound.

First, we find it appropriate to project ν to the subspace $\mathcal{S}_\ell$ and its orthogonal complement subspace then analyze separately. For convenience, denote $\nu_1 := \mathbb{P}_{S_\ell}(\nu)$, $\nu_2 := \mathbb{P}_{\mathcal{S}_\ell^\perp}(\nu)$. Then

$$\begin{aligned} |\langle x, \nu \rangle| =& |\langle y + z, \nu \rangle| \leq |\langle y, \nu_1 \rangle| + |\langle y, \nu_2 \rangle| + |\langle z, \nu \rangle| \\ \leq& \mu(\mathcal{X}_\ell)\|\nu_1\| + \|y\|\|\nu_2\| |\cos(\angle(y, \nu_2))| + \|z\|\|\nu\| |\cos(\angle(z, \nu))|. \end{aligned} \tag{5.10}$$

To see the last inequality, check that by Definition 3, $|\langle y, \frac{\nu_1}{\|\nu_1\|} \rangle| \leq \mu(\mathcal{X}_\ell)$.

Since we are considering general (possibly adversarial) noise, we will use the relaxation $|\cos(\theta)| \leq 1$ for all cosine terms (a better bound under random noise will be given later). Thus, what left is to bound $\|\nu_1\|$ and $\|\nu_2\|$ (note $\|\nu\| = \sqrt{\|\nu_1\|^2 + \|\nu_2\|^2} \leq \|\nu_1\| + \|\nu_2\|$).

5.3.1 Bounding $\|\nu_1\|$

We first bound $\|\nu_1\|$ by exploiting the feasible region of ν_1 in (5.7):

$$\left\{ \nu \middle| \|(X_{-i}^{(\ell)})^T \nu\|_\infty \leq 1 \right\},$$

which is equivalent to

$$\left\{ \nu \middle| x_j^T \nu \leq 1 \quad \text{for every column } x_j \text{ of } X_{-i}^{(\ell)} \right\}.$$

Decompose the condition into

$$y_j^T \nu_1 + (\mathbb{P}_{\mathcal{S}_\ell} z_j)^T \nu_1 + z_j^T \nu_2 \leq 1.$$

and relax the expression into

$$y_j^T \nu_1 + (\mathbb{P}_{\mathcal{S}_\ell} z_j)^T \nu_1 \leq 1 - z_j^T \nu_2 \leq 1 + \delta \|\nu_2\|. \tag{5.11}$$

The relaxed condition contains the feasible region of ν_1 in (5.7). It turns out that the geometric properties of this relaxed feasible region provides an upper bound of $\|\nu_1\|$.

Definition 13 (polar set) *The polar set $\mathcal{K}^o$ of set $\mathcal{K} \in \mathbb{R}^d$ is defined as*

$$\mathcal{K}^o = \left\{ y \in \mathbb{R}^d : \langle x, y \rangle \leq 1 \text{ for all } x \in \mathcal{K} \right\}.$$

By the polytope geometry, we have

$$\|(Y_{-i}^{(\ell)} + \mathbb{P}_{\mathcal{S}_\ell}(Z_{-i}^{(\ell)}))^T \nu_1\|_\infty \leq 1 + \delta\|\nu_2\| \;\Leftrightarrow\; \nu_1 \in \left[\mathcal{P}\left(\frac{Y_{-i}^{(\ell)} + \mathbb{P}_{\mathcal{S}_\ell}(Z_{-i}^{(\ell)})}{1 + \delta\|\nu_2\|} \right) \right]^o := \mathcal{T}^o. \tag{5.12}$$

Now we introduce the concept of circumradius.

Definition 14 (circumradius) *The circumradius of a convex body $\mathcal{P}$, denoted by $R(\mathcal{P})$, is defined as the radius of the smallest Euclidean ball containing $\mathcal{P}$.*

The magnitude $\|\nu_1\|$ is bounded by $R(\mathcal{T}^o)$. Moreover, by the the following lemma we may find the circumradius by analyzing the polar set of $\mathcal{T}^o$ instead. By the property of polar operator, polar of a polar set gives the tightest convex envelope of the original set, i.e., $(\mathcal{K}^o)^o = conv(\mathcal{K})$. Since $\mathcal{T} = \text{conv}\left(\pm \frac{Y_{-i}^{(\ell)} + \mathbb{P}_{\mathcal{S}_\ell}(Z_{-i}^{(\ell)})}{1 + \delta\|\nu_2\|} \right)$ is convex in the first place, the polar set of $\mathcal{T}^o$ is $\mathcal{T}$.

Lemma 15 (Page 448 in Brandenberg et al. (2004)) *For a symmetric convex body $\mathcal{P}$, i.e. $\mathcal{P} = -\mathcal{P}$, inradius of $\mathcal{P}$ and circumradius of polar set of $\mathcal{P}$ satisfy:*

$$r(\mathcal{P}) R(\mathcal{P}^o) = 1.$$

Lemma 16 *Given $X = Y + Z$, denote $\rho := \max_i \|\mathbb{P}_{\mathcal{S}} z_i\|$, furthermore $Y \in \mathcal{S}$ where $\mathcal{S}$ is a linear subspace, then we have:*

$$r(\text{Proj}_{\mathcal{S}}(\mathcal{P}(X))) \geq r(\mathcal{P}(Y)) - \rho$$

Proof First note that projection to a subspace is a linear operator. Hence $\text{Proj}_{\mathcal{S}}(\mathcal{P}(X)) = \mathcal{P}(\mathbb{P}_{\mathcal{S}} X)$. Then by definition, the boundary set of $\mathcal{P}(\mathbb{P}_{\mathcal{S}} X)$ is $\mathcal{B} := \{y \mid y = \mathbb{P}_{\mathcal{S}} X c; \|c\|_1 = 1\}$. Inradius by definition is the largest ball containing in the convex body, hence $r(\mathcal{P}(\mathbb{P}_{\mathcal{S}} X)) = \min_{y \in \mathcal{B}} \|y\|$. Now we provide a lower bound of it:

$$\|y\| \geq \|Yc\| - \|\mathbb{P}_{\mathcal{S}} Z c\| \geq r(\mathcal{P}(Y)) - \sum\nolimits_j \|\mathbb{P}_{\mathcal{S}} z_j\| |c_j| \geq r(\mathcal{P}(Y)) - \rho \|c\|_1.$$

This concludes the proof. ■

A bound of $\|\nu_1\|$ follows directly from Lemma 15 and Lemma 16:

$$\begin{aligned} \|\nu_1\| \leq& (1 + \delta\|\nu_2\|) R(\mathcal{P}(Y_{-i}^{(\ell)} + \mathbb{P}_{\mathcal{S}_\ell}(Z_{-i}^{(\ell)}))) \\ =& \frac{1 + \delta\|\nu_2\|}{r(\mathcal{P}(Y_{-i}^{(\ell)} + \mathbb{P}_{\mathcal{S}_\ell}(Z_{-i}^{(\ell)}))} = \frac{1 + \delta\|\nu_2\|}{r(\text{Proj}_{\mathcal{S}_\ell}(\mathcal{P}(X_{-i}^{(\ell)})))} \leq \frac{1 + \delta\|\nu_2\|}{r(\mathcal{Q}_{-i}^\ell) - \delta_1}. \end{aligned} \tag{5.13}$$

This bound depends on $\|\nu_2\|$, which we analyze below.

5.3.2 Bounding $\|\nu_2\|$

Since ν is the optimal solution to $\mathbf{D}_1$, it obeys the second optimality condition in Lemma 12:

$$\nu = \lambda e_i = \lambda(x_i - X_{-i}^{(\ell)} c).$$

By projecting ν to $\mathcal{S}_\ell^\perp$, we get $\nu_2 = \lambda \mathbb{P}_{\mathcal{S}_\ell^\perp}(x_i - X_{-i}^{(\ell)} c) = \lambda \mathbb{P}_{\mathcal{S}_\ell^\perp}(z_i - Z_{-i}^{(\ell)} c)$. It follows that

$$\begin{aligned}\|\nu_2\| &\le \lambda \left(\|\mathbb{P}_{\mathcal{S}_\ell^\perp} z_i\| + \|\mathbb{P}_{\mathcal{S}_\ell^\perp} Z_{-i}^{(\ell)} c\| \right) \\ &\le \lambda \left(\|\mathbb{P}_{\mathcal{S}_\ell^\perp} z_i\| + \sum_j |c_j| \|\mathbb{P}_{\mathcal{S}_\ell^\perp} z_j\| \right) \\ &\le \lambda(\|c\|_1 + 1)\delta_2 \le \lambda(\|c\|_1 + 1)\delta. \end{aligned} \tag{5.14}$$

Now we bound $\|c\|_1$. Since (c, e) is the optimal solution, $\|c\|_1 + \frac{\lambda}{2}\|e\|^2 \le \|\tilde{c}\|_1 + \frac{\lambda}{2}\|\tilde{e}\|^2$ for any feasible solution $(\tilde{c}, \tilde{e})$. Let $\tilde{c}$ be the solution of

$$\min_c \ \|c\|_1 \quad s.t. \quad y_i^{(\ell)} = Y_{-i}^{(\ell)} c, \tag{5.15}$$

then by strong duality,

$$\|\tilde{c}\|_1 = \max_\nu \left\{ \langle \nu, y_i^{(\ell)} \rangle \mid \|[Y_{-i}^{(\ell)}]^T \nu\|_\infty \le 1 \right\}.$$

By Lemma 15, the optimal dual solution $\tilde{\nu}$ satisfies $\|\tilde{\nu}\| \le \frac{1}{r(\mathcal{Q}_{-i}^\ell)}$. It follows that

$$\|\tilde{c}\|_1 = \langle \tilde{\nu}, y_i^{(\ell)} \rangle = \|\tilde{\nu}\| \|y_i^{(\ell)}\| \le \frac{1}{r(\mathcal{Q}_{-i}^\ell)}.$$

On the other hand, $\tilde{e} = z_i - Z_{-i}^{(\ell)} \tilde{c}$, so $\|\tilde{e}\|^2 \le (\|z_i\| + \sum_j \|z_j\| |\tilde{c}_j|)^2 \le (\delta + \|\tilde{c}\|_1 \delta)^2$, thus

$$\|c\|_1 \le \|\tilde{c}\|_1 + \frac{\lambda}{2}\|\tilde{e}\|^2 - \frac{\lambda}{2}\|e\|^2 \le \frac{1}{r(\mathcal{Q}_{-i}^\ell)} + \frac{\lambda}{2}\delta^2 \left[1 + \frac{1}{r(\mathcal{Q}_{-i}^\ell)}\right]^2 - \frac{1}{2\lambda}\|\nu_2\|^2.$$

Note that we used the property $\frac{\lambda}{2}\|e\|^2 = \frac{1}{2\lambda}\|\nu\|^2 \ge \frac{1}{2\lambda}\|\nu_2\|^2$. Substitute the bound of $\|c_1\|_1$ into (5.14) we get

$$\begin{aligned} & \|\nu_2\| \le \lambda \left(\frac{1}{r(\mathcal{Q}_{-i}^\ell)} + \frac{\lambda}{2}\delta^2 \left[1 + \frac{1}{r(\mathcal{Q}_{-i}^\ell)}\right]^2 + 1 \right) \delta - \frac{\delta}{2}\|\nu_2\|^2 \\ \Leftrightarrow \quad & \|\nu_2\| + \frac{\delta}{2}\|\nu_2\|^2 \le \lambda\delta \left(\frac{1}{r(\mathcal{Q}_{-i}^\ell)} + 1 \right) + \frac{\delta}{2} \left[\lambda\delta \left(\frac{1}{r(\mathcal{Q}_{-i}^\ell)} + 1 \right) \right]^2. \end{aligned}$$

Since function $f(\alpha) = \alpha + \frac{\delta}{2}\alpha^2$ monotonically increases when $\alpha > 0$, the above inequality implies

$$\|\nu_2\| \le \lambda\delta \left(\frac{1}{r(\mathcal{Q}_{-i}^\ell)} + 1 \right), \tag{5.16}$$

which gives the desired bound for $\|\nu_2\|$.

5.3.3 Conditions for $|\langle x, \nu \rangle| < 1$

Putting together (5.10), (5.13) and (5.16), we have the upper bound of $|\langle x, \nu \rangle|$:

$$\begin{aligned} & |\langle x, \nu \rangle| \le (\mu(\mathcal{X}_\ell) + \|\mathbb{P}_{\mathcal{S}_\ell} z\|)\|\nu_1\| + (\|y\| + \|\mathbb{P}_{\mathcal{S}_\ell^\perp} z\|)\|\nu_2\| \\ \le & \frac{\mu(\mathcal{X}_\ell) + \delta_1}{r(\mathcal{Q}_{-i}^\ell) - \delta_1} + \left(\frac{(\mu(\mathcal{X}_\ell) + \delta_1)\delta}{r(\mathcal{Q}_{-i}^\ell) - \delta_1} + 1 + \delta \right) \|\nu_2\| \\ \le & \frac{\mu(\mathcal{X}_\ell) + \delta_1}{r(\mathcal{Q}_{-i}^\ell) - \delta_1} + \lambda\delta(1+\delta) \left(\frac{1}{r(\mathcal{Q}_{-i}^\ell)} + 1 \right) + \frac{\lambda\delta^2(\mu(\mathcal{X}_\ell) + \delta_1)}{r(\mathcal{Q}_{-i}^\ell) - \delta_1} \left(\frac{1}{r(\mathcal{Q}_{-i}^\ell)} + 1 \right). \end{aligned}$$

For convenience, we further relax the second $r(\mathcal{Q}_{-i}^\ell)$ into $r(\mathcal{Q}_{-i}^\ell) - \delta_1$. The dual separation condition is thus guaranteed with

$$\frac{\mu(\mathcal{X}_\ell) + \delta_1 + \lambda\delta(1+\delta) + \lambda\delta^2(\mu(\mathcal{X}_\ell) + \delta_1)}{r(\mathcal{Q}_{-i}^\ell) - \delta_1} + \lambda\delta(1+\delta) + \frac{\lambda\delta^2(\mu(\mathcal{X}_\ell) + \delta_1)}{r(\mathcal{Q}_{-i}^\ell)(r(\mathcal{Q}_{-i}^\ell) - \delta_1)} < 1.$$

Denote $\rho := \lambda\delta(1+\delta)$, assume $\delta < r(\mathcal{Q}_{-i}^\ell)$, $(\mu(\mathcal{X}_\ell) + \delta_1) < 1$ and simplify the form with

$$\frac{\lambda\delta^2(\mu(\mathcal{X}_\ell) + \delta_1)}{r(\mathcal{Q}_{-i}^\ell) - \delta_1} + \frac{\lambda\delta^2(\mu(\mathcal{X}_\ell) + \delta_1)}{r(\mathcal{Q}_{-i}^\ell)(r(\mathcal{Q}_{-i}^\ell) - \delta_1)} < \frac{\rho}{r(\mathcal{Q}_{-i}^\ell) - \delta_1},$$

we get a sufficient condition

$$\mu(\mathcal{X}_\ell) + 2\rho + \delta_1 < (1 - \rho)\left(r(\mathcal{Q}_{-i}^\ell) - \delta_1\right). \tag{5.17}$$

To generalize (5.17) to all data of all subspaces, the following must hold for each $\ell = 1, ..., k$:

$$\mu(\mathcal{X}_\ell) + 2\rho + \delta_1 < (1 - \rho)\left(\min_{\{i : x_i \in X^{(\ell)}\}} r(\mathcal{Q}_{-i}^{(\ell)}) - \delta_1 \right). \tag{5.18}$$

This gives a first condition on δ and λ (within ρ), which we call it "**dual separation condition**" under noise. Note that this reduces to exactly the geometric condition in Soltanolkotabi et al. (2012)'s Theorem 2.5 when $\delta = 0$.

5.4 Avoid trivial solutions

Besides SEP, we also need to show the solution is non-trivial. The idea is that when λ is large enough, the trivial solution $c^* = 0$, $e^* = x_i^{(\ell)}$ can never be optimal.

As we trace along the regularization path by increasing λ from 0, one column of the design matrix X_{-i} will enter the support set. This column will be the one that attains $\|X_{-i}^T x_i\|_\infty$, and $\lambda = \frac{1}{\|X_{-i}^T x_i\|_\infty}$ when it happens. Therefore, as long as $\lambda > \frac{1}{\|X_{-i}^T x_i\|_\infty}$, the solution will not be trivial.

Note that under the dual separation condition, we only need to consider points in the same subspace. So $\|X_{-i}^T x_i\|_\infty = \left\| [X_{-i}^{(\ell)}]^T x_i \right\|_\infty$. Let $x_j \in X_{-i}^{(\ell)}$ be the column that attains

the maximum in $\left\|[X_{-i}^{(\ell)}]^T x_i\right\|_\infty$ and $y_k \in Y_{-i}^{(\ell)}$ be the column that attains the maximum in $\left\|[Y_{-i}^{(\ell)}]^T y_i\right\|_\infty$ (if there are more than one maximizers, pick any one), we can write

$$\begin{aligned}\left\|[X_{-i}^{(\ell)}]^T x_i\right\|_\infty &= |\langle x_j, x_i\rangle| \geq |\langle x_k, x_i\rangle| \\ &= |\langle y_k, y_i\rangle + \langle y_k, z_i\rangle + \langle z_k, y_i\rangle + \langle z_k, z_i\rangle| \\ &\geq |\langle y_k, y_i\rangle| - |\langle y_k, z_i\rangle + \langle z_k, y_i\rangle + \langle z_k, z_i\rangle| \\ &= \left\|[Y_{-i}^{(\ell)}]^T y_i\right\|_\infty - |\langle y_k, z_i\rangle + \langle z_k, y_i\rangle + \langle z_k, z_i\rangle| \\ &\geq r(\mathcal{Q}_{-i}^{(\ell)}) - 2\delta - \delta^2. \end{aligned} \tag{5.19}$$

The last inequality follows from the upper bound of noise magnitude and the observation that the inradius of $\mathcal{Q}_{-i}^{(\ell)}$ defines a uniform lower bound of $\left\|[Y_{-i}^{(\ell)}]^T w\right\|_\infty$ for any unit vector $w \in \mathcal{S}_\ell$. Therefore, as long as

$$\lambda \geq \frac{1}{r(\mathcal{Q}_{-i}^{(\ell)}) - 2\delta - \delta^2}, \tag{5.20}$$

the solution c_i for ith column is not trivial. This bound is strictly better than what we obtain in the conference version (Wang and Xu, 2013) and is the key for improving the rate for noise tolerance over the previous version. Also, check that

$$\delta < \frac{r(r_\ell - \mu_\ell)}{2 + 7r_\ell} \tag{5.21}$$

under bound of δ in the theorem statement, $r(\mathcal{Q}_{-i}^{(\ell)}) - 2\delta - \delta^2 > 0$ for any i, ℓ.

A side remark is that the Lasso regularization path is formally described in Tibshirani et al. (2013) and it is unique whenever the data points are in general position. As a result, we can potentially calculate the entry point of kth non-zero coefficient for any $0 < k < d$, any x_i and X_{-i}. This would however complicate the results unnecessarily, as Lasso path is not monotone (some coefficient may leave the support set as λ increases). We therefore stick to the simpler requirement of c_i being non-trivial.

5.5 Existence of a proper λ

Basically, (5.18) and (5.20) must be satisfied simultaneously for all $\ell = 1, ..., L$. Essentially (5.20) gives a condition of λ from below, and (5.18) gives a condition from above. Recall that the denotations $r_\ell := \min_{\{i: x_i \in X^{(\ell)}\}} r(\mathcal{Q}_{-i}^{(\ell)})$, $\mu_\ell := \mu(\mathcal{X}_\ell)$ and $r = \min_\ell r_\ell$, the condition on λ is:

$$\max_\ell \frac{1}{r_\ell - 2\delta - \delta^2} < \lambda < \min_\ell \frac{r_\ell - \mu_\ell - 2\delta_1}{\delta(1+\delta)(2 + r_\ell - \delta_1)}.$$

With the observation that

$$\max_\ell \frac{1}{r_\ell - 2\delta - \delta^2} = \frac{1}{\min r_\ell - 2\delta - \delta^2},$$

it suffices to require λ to obey for each ℓ:

$$\frac{1}{r - 2\delta - \delta^2} < \lambda < \frac{r_\ell - \mu_\ell - 2\delta_1}{\delta(1+\delta)(2 + r_\ell - \delta_1)}. \tag{5.22}$$

We will now show that under condition (5.21), the range (5.22) is not an empty set. Again, we relax δ_1 to δ in (5.22) and get

$$\frac{1}{r - 2\delta - \delta^2} < \frac{r_\ell - \mu_\ell - 2\delta}{\delta(1+\delta)(2 + r_\ell - \delta)}. \tag{5.23}$$

Since all denominators are positive, we obtain the standard form of the inequality

$$A\delta^3 + B\delta^2 + C\delta + D < 0$$

with

$$\begin{cases} A = -3 \leq 0 \\ B = -3 + 2(r_\ell - \mu_\ell) + r_\ell \leq 0 \\ C = 2 + 4(r_\ell - \mu_\ell) + r_\ell + 2r \leq 2 + 7r_\ell \\ D = -r(r_\ell - \mu_\ell) \end{cases}$$

Check that (5.21) is sufficient for the above 3rd order inequality to hold. Therefore,

$$(5.21) \Rightarrow A\delta^3 + B\delta^2 + C\delta + D < 0 \Leftrightarrow (5.23) \Rightarrow (5.22) \text{ is not an empty set.}$$

This completes the proof of Theorem 6.

6. Proof of Results for Randomized Cases

In this section, we provide proofs to the theorems of the three randomized models:

- **Deterministic data+random noise;**
- **Semi-random data+random noise;**
- **Fully random.**

To do this, we need to bound δ_1, $\cos(\angle(z, \nu))$ and $\cos(\angle(y, \nu_2))$ when Z follows the *Random Noise Model*, such that a better dual separation condition can be obtained. Moreover, for the *Semi-random* and the *Random data model*, we need to bound $r(\mathcal{Q}_{-i}^{(\ell)})$ when data samples from each subspace are drawn uniformly and bound $\mu(\mathcal{X}_\ell)$ when subspaces are randomly generated. These require the following lemmas.

Lemma 17 (Upper bound on the area of spherical cap) *Let $a \in \mathbb{R}^n$ be a random vector sampled from a unit sphere and z is a fixed vector. Then we have:*

$$Pr\left(|a^T z| > \epsilon \|z\|\right) \leq 2e^{\frac{-n\epsilon^2}{2}}$$

This Lemma is extracted from an equation in page 29 of Soltanolkotabi et al. (2012), which is in turn adapted from the upper bound on the area of spherical cap in Ball (1997). By definition of the Random Noise Model, z_i is spherical symmetric, which implies that the direction of z_i is distributed uniformly on the n-dimensional unit sphere. Hence Lemma 17 applies whenever an inner product involves z. As an example, we write the following lemma.

Lemma 18 (Properties of Gaussian noise) *For Gaussian random matrix $Z \in \mathbb{R}^{n\times N}$, if each entry $Z_{i,j} \sim N(0, \frac{\sigma}{\sqrt{n}})$, then each column z_i satisfies:*

1. $Pr(\|z_i\|^2 > (1+t)\sigma^2) \le e^{\frac{n}{2}(\log(t+1)-t)}$
2. $Pr(|\langle z_i, z\rangle| > \epsilon\|z_i\|\|z\|) \le 2e^{\frac{-n\epsilon^2}{2}}$

where z is any fixed vector, or a random vector that is independent to z_i.

Proof The second property follows directly from Lemma 17 as Gaussian vector has a uniformly random direction.

To show the first property, we observe that the sum of n independent square Gaussian random variables follows χ^2 distribution with degree of freedom n. In other words, we have

$$\|z_i\|^2 = |Z_{1i}|^2 + ... + |Z_{ni}|^2 \sim \frac{\sigma^2}{n}\chi^2(n).$$

By Hoeffding's inequality, we have an approximation of its CDF (Dasgupta and Gupta, 2003), which gives us

$$Pr(\|z_i\|^2 > \alpha\sigma^2) = 1 - \mathrm{CDF}_{\chi^2_n}(\alpha) \le (\alpha e^{1-\alpha})^{\frac{n}{2}}.$$

Substitute $\alpha = 1+t$, we obtain the concentration statement in the lemma. ■

By Lemma 18, $\delta = \max_i \|z_i\|$ is bounded with high probability. δ_1 can be bounded even more tightly because each $\mathcal{S}_\ell$ is low-rank. Likewise, $\cos(\angle(z,\nu))$ is bounded by a small value with high probability. Moreover, since $\nu = \lambda e = \lambda(x_i - X_{-i}c)$, $\nu_2 = \lambda\mathbb{P}_{\mathcal{S}_\ell^\perp}(z_i - Z_{-i}c)$. Thus ν_2 is indeed a weighted sum of random noise in a $(n-d_\ell)$-dimensional subspace. Consider y a fixed vector, $\cos(\angle(y,\nu_2))$ is also bounded with high probability.

Replace these observations into (5.9) and the corresponding bound of $\|\nu_1\|$ and $\|\nu_2\|$, we obtain the equivalent *dual separation condition* under the random noise model (equivalent to (5.17) in the proof of the deterministic case). This is formalized in the following lemma.

Lemma 19 (Dual separation condition under random noise) *Let $\rho := \lambda\delta(1+\delta)$ and*

$$\epsilon := \sqrt{\frac{6\log N}{n - \max_\ell d_\ell}} \le \sqrt{\frac{C\log(N)}{n}}$$

for some constant C. Under random noise model, if for each $\ell = 1, ..., L$

$$\mu(\mathcal{X}_\ell) + \delta\epsilon + 3\rho\epsilon \le (1-\rho\epsilon)(\max_i r(\mathcal{Q}^{(\ell)}_{-i}) - \delta\sqrt{d_\ell}\epsilon), \tag{6.1}$$

then dual separation condition (5.9) *holds for all data points with probability at least $1-8/N$.*

Proof Recall that we want to find an upper bound of $|\langle x, \nu\rangle|$.

$$|\langle x,\nu\rangle| \le \mu\|\nu_1\| + \|y\|\|\nu_2\||\cos(\angle(y,\nu_2))| + \|z\|\|\nu\||\cos(\angle(z,\nu))| \tag{6.2}$$

Here we will bound the two cosine terms and δ_1 under the random noise model.

As discussed above, directions of z and ν_2 are independently and uniformly distributed on the n-dimension unit sphere. Then by Lemma 17,

$$\begin{cases} Pr\left(\cos(\angle(z,\nu)) > \sqrt{\frac{6\log N}{n}}\right) \le \frac{2}{N^3}; \\ Pr\left(\cos(\angle(y,\nu_2)) > \sqrt{\frac{6\log N}{n-d_\ell}}\right) \le \frac{2}{N^3}; \\ Pr\left(\cos(\angle(z,\nu_2)) > \sqrt{\frac{6\log N}{n}}\right) \le \frac{2}{N^3}. \end{cases}$$

Using the same technique, we derive a bound for δ_1. Given an orthonormal basis U of S_ℓ, $\mathbb{P}_{S_\ell}z = UU^Tz$, then

$$\|UU^Tz\| = \|U^Tz\| = \sqrt{\sum_{i=1,...,d_\ell} |U^T_{:,i}z|^2}.$$

Apply Lemma 17 for each i, then by union bound, we get:

$$Pr\left(\|\mathbb{P}_{S_\ell}z\| > \sqrt{\frac{6d_\ell\log N}{n}}\delta\right) \le \frac{2d_\ell}{N^3}.$$

Since δ_1 is the worse case bound for all L subspace and all N noise vector, then a union bound gives:

$$Pr\left(\delta_1 > \sqrt{\frac{6d_\ell\log N}{n}}\delta\right) \le \frac{2\sum_\ell d_\ell}{N^2}$$

Moreover, we can find a probabilistic bound for $\|\nu_1\|$ too by a variation of (5.11) for the random case, which now becomes

$$y_i^T\nu_1 + (\mathbb{P}_{\mathcal{S}_\ell}z_i)^T\nu_1 \le 1 - z_i^T\nu_2 \le 1 + \delta_2\|\nu_2\||\cos\angle(z_i,\nu_2)|. \tag{6.3}$$

Substituting the upper bound of the cosines to (6.2) and (6.3), we get respectively

$$|\langle x,\nu\rangle| \le \mu\|\nu_1\| + \|y\|\|\nu_2\|\sqrt{\frac{6\log N}{n-d_\ell}} + \|z\|\|\nu\|\sqrt{\frac{6\log N}{n}},$$

and

$$\|\nu_1\| \le \frac{1 + \delta\|\nu_2\|\sqrt{\frac{6\log N}{n}}}{r(\mathcal{Q}^\ell_{-i}) - \delta_1}.$$

This new bound of $\|\nu_1\|$ follows from (6.3), Lemma 15 and 16. For the bound of $\|\nu_2\|$ we simply use (5.16):

$$\|\nu_2\| \le \lambda\delta\left(\frac{1}{r(\mathcal{Q}^\ell_{-i})} + 1\right).$$

To lighten notations in this proof, denote

$$r := r(\mathcal{Q}^{\ell}_{-i}), \quad \epsilon := \sqrt{\frac{6\log N}{n - \max_\ell d_\ell}}, \quad \mu := \mu(\mathcal{X}_\ell).$$

Substitute them in the bound, we get

$$\begin{aligned}
|\langle x, \nu\rangle| \leq & \frac{\mu+\delta\epsilon}{r-\epsilon\sqrt{d_\ell}\delta} + \frac{\lambda\delta^2(\mu+\delta\epsilon)\epsilon}{r-\epsilon\sqrt{d_\ell}\delta}\left(\frac{1}{r}+1\right) + \lambda\delta\epsilon\left(\frac{1}{r}+1\right) + \lambda\delta^2\epsilon\left(\frac{1}{r}+1\right) \\
= & \frac{\mu+\delta\epsilon}{r-\epsilon\sqrt{d_\ell}\delta} + \frac{\lambda\epsilon\delta^2(\frac{\mu+\delta\epsilon}{r}) + \lambda\epsilon\delta^2(\mu+\delta\epsilon)}{r-\epsilon\sqrt{d_\ell}\delta} + \frac{\lambda\delta(\delta+1)\epsilon}{r} + \lambda\delta(\delta+1)\epsilon \\
\underset{*}{\leq} & \frac{\mu+\delta\epsilon}{r-\epsilon\sqrt{d_\ell}\delta} + \frac{\lambda\epsilon\delta^2}{r-\epsilon\sqrt{d_\ell}\delta} + \frac{\lambda\epsilon\delta^2}{r-\epsilon\sqrt{d_\ell}\delta} + \frac{\lambda\epsilon(\delta+\delta^2)}{r-\epsilon\sqrt{d_\ell}\delta} + \lambda\epsilon(\delta+\delta^2) \\
= & \frac{\mu+\delta\epsilon+\lambda\epsilon(\delta+3\delta^2)}{r-\epsilon\sqrt{d_\ell}\delta} + \lambda\epsilon(\delta+\delta^2) \underset{**}{\leq} \frac{\mu+\delta\epsilon+3\rho\epsilon}{r-\epsilon\sqrt{d_\ell}\delta} + \rho\epsilon. \qquad (6.4)
\end{aligned}$$

In the inequality "$*$", we used $(\mu+\epsilon\delta)/r < 1$ and $\mu+\epsilon\delta < 1$; and in the inequality "$**$", we used $\lambda\delta^2 \leq \lambda\epsilon(\delta+\delta^2)$ and replaced all such expression with $\rho\epsilon$ that we defined earlier.

Now impose the dual detection constraint on the upper bound, we get:

$$\rho\epsilon + \frac{\mu+\delta\epsilon+3\rho\epsilon}{r-\delta\sqrt{d_\ell}\epsilon} < 1.$$

Reorganized the inequality, we reach the desired condition:

$$\mu+\delta\epsilon < (1-\rho\epsilon)(r-\delta\sqrt{d_\ell}\epsilon) - 3\rho\epsilon.$$

There are N^2 instances for each of the three events related to the cosine value, apply union bound we get the failure probability $\frac{6}{N} + \frac{2\sum_\ell d_\ell}{N^2} \leq \frac{8}{N}$. Note $\sum_\ell d_\ell \leq N$ because one needs at least d_ℓ data points to span an d_ℓ dimensional subspace. This concludes the proof. ■

Lemma 20 (Avoid trivial solution under random noises) *Let* $\epsilon = \sqrt{\frac{6\log N}{n}}$ *and assume* $\min_\ell r_\ell - 2\epsilon\delta - \epsilon\delta^2 > 0$. *If we take*

$$\lambda > (r_\ell - 2\epsilon\delta - \epsilon\delta^2)^{-1} \qquad (6.5)$$

for every $\ell = 1, ..., n$, *then the solution* $c_i \neq 0$ *for all* i *with probability at least* $1 - 6/N^2$.

Proof We use the same argument as in Section 5.4, except that we now have a tighter probabilistic bound for (5.19). For any i, k in the equation, z_i and z_k are independent to each other and to y_k, y_i respectively. Therefore, we can invoke Lemma 17 and obtain

$$|\langle y_k, z_i\rangle + \langle z_k, y_i\rangle + \langle z_k, z_i\rangle| \leq 2\epsilon\delta + \epsilon\delta^2,$$

with probability greater than $2/N^3$. The proof is complete by taking the union bound over all $3\sum_i N_\ell = 3N$ instances. ■

6.1 Proof of Theorem 8 for Deterministic Data and Random Noise

We now prove Theorem 8. Lemma 19 has already provided the separation condition. The things left are to find the range of λ and update the condition of δ.

The range of λ: The range of valid λ for the random noise case can be obtained by substituting $\delta_1 < \delta\sqrt{d_\ell}\epsilon$ in (6.5) and rewriting (6.1) with respect to λ. This gives us

$$\frac{1}{r-2\epsilon\delta-\epsilon\delta^2} < \lambda < \frac{r_\ell - \mu_\ell - \delta\epsilon - \delta\sqrt{d_\ell}\epsilon}{\epsilon\delta(1+\delta)(3+r_\ell-\delta\sqrt{d_\ell}\epsilon)}. \qquad (6.6)$$

We remark that a critical difference from the deterministic noise model is that now there is a small ϵ in the denominator of the upper endpoint of the interval. Assume small μ, the valid range of λ expands to an order of $\Theta(1/r) \leq \lambda \leq \Theta(r/(\epsilon\max\{\delta^2, \delta\}))$.

The condition of δ: Now we will show that the two conditions

$$\epsilon\delta < \min_\ell \frac{r_\ell - \mu_\ell}{2\sqrt{d_\ell}+2}, \qquad \text{and} \qquad \epsilon\delta(1+\delta) < \min_\ell \frac{r(r_\ell-\mu_\ell)}{4r_\ell+6},$$

stated in the Theorem 8 are sufficient for the three inequalities

$$\begin{cases}
r_\ell - \mu_\ell - \delta\epsilon > 0; & (6.7)\\
r - 2\delta\epsilon - \epsilon\delta^2 > 0; & (6.8)\\
\dfrac{1}{r-2\epsilon\delta-\epsilon\delta^2} < \dfrac{r_\ell-\mu_\ell-\delta\epsilon-\delta\sqrt{d_\ell}\epsilon}{\epsilon\delta(1+\delta)(3+r_\ell-\delta\sqrt{d_\ell}\epsilon)}; & (6.9)
\end{cases}$$

to hold for $\ell = 1, ..., L$. Note that we used (6.7) in (6.4) when we derive the dual separation condition and (6.8) is assumed in Lemma 20, lastly (6.9) ensures a valid λ to exist in (6.6). Inequality (6.7) and (6.8) hold trivially given the two conditions, it remains to show (6.9):

$$\begin{aligned}
\delta(1+\delta) < \frac{r(r_\ell-\mu_\ell)}{\epsilon(4r_\ell+6)} & \Leftrightarrow \epsilon\delta(1+\delta)(2r_\ell+3) < \frac{r(r_\ell-\mu_\ell)}{2} \\
& \Rightarrow \epsilon\delta(1+\delta)(r_\ell-\mu_\ell+r_\ell+3) < \frac{r(r_\ell-\mu_\ell)}{2} \\
& \Leftrightarrow \epsilon\delta(1+\delta)(r_\ell+3) + 2\epsilon\delta(1+\delta)\frac{r_\ell-\mu_\ell}{2} < \frac{r(r_\ell-\mu_\ell)}{2} \\
& \Leftrightarrow \frac{1}{r-2\epsilon\delta-2\epsilon\delta^2} < \frac{r_\ell-\mu_\ell}{2\epsilon\delta(1+\delta)(r_\ell+3)} \\
& \Rightarrow \frac{1}{r-2\epsilon\delta-\epsilon\delta^2} < \frac{r_\ell-\mu_\ell}{2\epsilon\delta(1+\delta)(r_\ell+3-\delta\sqrt{d_\ell}\epsilon)} \overset{(a)}{\Rightarrow} (6.9),
\end{aligned}$$

where (a) holds by applying the first condition. This concludes the proof for Theorem 8.

6.2 Proof of Theorem 10 for the Semi-random Model with Random Noise

To prove Theorem 10, we only need to bound the inradius r and the incoherence parameter μ under the new assumptions, then plug them into Theorem 8.

Lemma 21 (Inradius bound of random samples) *In random sampling setting, when each subspace is sampled $N_\ell = \kappa_\ell d_\ell$ data points randomly, we have:*

$$Pr\left\{ c(\kappa_\ell)\sqrt{\frac{\beta \log(\kappa_\ell)}{d_\ell}} \le r(\mathcal{Q}_{-i}^{(\ell)}) \text{ for all pairs } (\ell, i) \right\} \ge 1 - \sum_{\ell=1}^{L} N_\ell e^{-d_\ell^\beta N_\ell^{1-\beta}}$$

This is extracted from Section-7.2.1 of Soltanolkotabi et al. (2012). $\kappa_\ell = (N_\ell - 1)/d_\ell$ is the relative number of iid samples. $c(\kappa)$ is some positive value for all $\kappa > 1$ and for a numerical value κ_0, if $\kappa > \kappa_0$, we can take $c(\kappa) = \frac{1}{\sqrt{8}}$. Take $\beta = 0.5$, we get the required bound of r in Theorem 10.

Now, we provide a probabilistic upper bound of the projected subspace incoherence condition under the semi-random model by adapting Lemma 7.5 of Soltanolkotabi et al. (2012) into our new setup.

Lemma 22 (Incoherence bound) *In deterministic subspaces/random sampling setting, the subspace incoherence is bounded from above:*

$$Pr\Big\{ \mu(\mathcal{X}_\ell) \le t\left(\log[(N_\ell+1)N_{\ell'}] + \log L\right) \frac{\text{aff}(S_\ell, S_{\ell'})}{\sqrt{d_\ell}\sqrt{d_{\ell'}}} \\ \text{for all pairs}(\ell, \ell') \text{ with } \ell \ne \ell' \Big\} \ge 1 - \frac{1}{L^2} \sum_{\ell \ne \ell'} \frac{1}{(N_\ell+1)N_{\ell'}} e^{-\frac{t}{4}}.$$

Proof The proof is an extension of a similar proof in Soltanolkotabi et al. (2012). First we will show that when noise $z_i^{(\ell)}$ is spherical symmetric, and clean data points $y_i^{(\ell)}$ has iid uniform random direction, projected dual directions $v_i^{(\ell)}$ also follows a uniform random distribution.

Now we prove the claim. First by definition,

$$v_i^{(\ell)} = v(x_i^{(\ell)}, X_{-i}^{(\ell)}, \mathcal{S}_\ell, \lambda) = \frac{\mathbb{P}_{S_\ell}\nu}{\|\mathbb{P}_{S_\ell}\nu\|} = \frac{\nu_1}{\|\nu_1\|}.$$

Recall that ν is the unique optimal solution of $\mathbf{D}_1$ (5.7). Fix λ, $\mathbf{D}_1$ depends on two inputs, so we denote $\nu(x, X)$ and consider ν a function. Moreover, $\nu_1 = \mathbb{P}_{\mathcal{S}}\nu$ and $\nu_2 = \mathbb{P}_{\mathcal{S}^\perp}\nu$. Let $U \in n \times d$ be a set of orthonormal basis of d-dimensional subspace $\mathcal{S}$ and a rotation matrix $R \in \mathbb{R}^{d\times d}$. Then rotation matrix within subspace is hence URU^T. Let

$$x_1 := \mathbb{P}_{\mathcal{S}} x = y + z_1 \sim URU^T y + URU^T z_1,$$
$$x_2 := \mathbb{P}_{\mathcal{S}^\perp} x = z_2.$$

As y is distributed uniformly on the unit sphere of $\mathcal{S}$, and z is a spherical symmetric noise (hence z_1 and z_2 are also spherical symmetric in subspace), for any fixed $\|x_1\|$, the distribution is uniform on the sphere, namely the conditional distribution $Pr(x_1 | \|x_1\| = \alpha)$ is uniform on the sphere with radius α. It suffices to show that with fixed $\|x_1\|$, v (the unit direction of projected dual variable ν_1) also follows a uniform distribution on a unit sphere of the subspace.

Since inner product $\langle x, \nu\rangle = \langle x_1, \nu_1\rangle + \langle x_2, \nu_2\rangle$, we argue that if ν is the optimal solution of

$$\max_\nu \langle x, \nu\rangle - \frac{1}{2\lambda}\nu^T\nu, \quad \text{subject to:} \quad \|X^T\nu\|_\infty \le 1,$$

then the optimal solution of the following optimization

$$\max_\nu \langle URU^T x_1 + x_2, \nu\rangle - \frac{1}{2\lambda}\nu^T\nu,$$
$$\text{subject to:} \quad \|(URU^T X_1 + X_2)^T\nu\|_\infty \le 1,$$

is indeed the transformed ν under the same R, i.e.,

$$\begin{aligned}\nu(R) &= \nu(URU^T x_1 + x_2, URU^T X_1 + X_2) \\ &= URU^T \nu_1(x, X) + \nu_2(x, X) = URU^T\nu_1 + \nu_2. \end{aligned} \tag{6.10}$$

To verify the argument, check that $\nu^T\nu = \nu(R)^T\nu(R)$ and

$$\langle URU^T x_1 + x_2, \nu(R)\rangle = \langle URU^T x_1, URU^T\nu_1\rangle + \langle x_1, \nu_2\rangle = \langle x, \nu\rangle$$

for all inner products in both objective function and constraints, preserving the optimality.

By projecting (6.10) to subspace, we show that operator $v(x, X, S)$ is linear *vis a vis* subspace rotation URU^T, i.e.,

$$v(R) = \frac{\mathbb{P}_{S_\ell}\nu(R)}{\|\mathbb{P}_{S_\ell}\nu(R)\|} = \frac{URU^T\nu_1}{\|URU^T\nu_1\|} = URU^T v. \tag{6.11}$$

On the other hand, we know that

$$URU^T x_1 + x_2 \sim x_1 + x_2, \qquad URU^T X_1 + X_2 \sim X_1 + X_2,$$

where $A \sim B$ means that the random variables A and B follows the same distribution. This is because when $\|x_1\|$ is fixed and each columns in X_1 has fixed magnitudes, $URU^T x_1 \sim x_1$ and $URU^T X_1 \sim X_1$. Also, adding additional random variables x_2 and X_2 changes the distribution the same way on both sides. Therefore,

$$v(R) = v(URU^T x_1 + x_2, URU^T X_1 + X_2, \mathcal{S}) \sim v(x, X, \mathcal{S}). \tag{6.12}$$

Combining (6.11) and (6.12), we conclude that for any rotation R

$$v_i^{(\ell)}(R) \sim URU^T v_i^{(\ell)}.$$

In other words, the distribution of $v_i^{(\ell)}$ is uniform on the unit sphere of $\mathcal{S}_\ell$.

After this key step, the rest is identical to the proof of Lemma 7.5 of Soltanolkotabi et al. (2012). The idea is to use Lemma 17 (upper bound of area of spherical caps) to provide a probabilistic bound of the pairwise inner product and Borell's inequality to show the concentration around the expected cosine canonical angles, namely, $\|U^{(k)^T}U^{(\ell)}\|_F/\sqrt{d_\ell}$. The proof is standard so we omit it in this paper. ■

6.3 Proof of Theorem 11 for the Fully Random Model with Gaussian Noises

The proof of Theorem 11 essentially applies Theorem 8 with specific inradius bound and incoherence bound. The bound for inradius is given in Lemma 21 and we use the following Lemma extracted from Step 2 of Section 7.3 in Soltanolkotabi et al. (2012) to bound the subspace incoherence.

Lemma 23 (Incoherence bound of random subspaces) *In random subspaces setting, the projected subspace incoherence is bounded from above:*

$$Pr\left\{\mu(\mathcal{X}_\ell) \le \sqrt{\frac{6\log N}{n}} \text{ for all } \ell\right\} \ge 1 - \frac{2}{N}.$$

Now that we have shown that the projected dual directions are randomly distributed in their respective subspace, together with the fact that the subspaces themselves are randomly generated, we conclude that all clean data points y and projected dual direction v from different subspaces can be considered iid generated from the ambient space. The proof of Lemma 23 follows by simply applying Lemma 17 and a union bound across all N^2 events.

7. Experiments

To demonstrate the practical implications of our robustness guarantees for LASSO-SSC, we conduct four numerical experiments including three with synthetic generated data and one with real data. For fast computation, we use ADMM implementation of LASSO solver[8] with default numerical parameters. Its complexity is proportional to the problem size and the convergence guarantee (Boyd et al., 2011). We also implement a simple ADMM solver for the matrix version SSC

$$\min_C \ \|C\|_1 + \frac{\lambda}{2}\|X - XC\|_F^2 \quad \text{s.t.} \quad \text{diag}(C) = 0,$$

which is consistently faster than the column-by-column LASSO version. This algorithm is first described in Elhamifar and Vidal (2013). To be self-contained, we provide the pseudocode and some numerical simulation in the appendix.

7.1 Numerical simulation

Our three numerical simulations test the effects of noise magnitude δ, subspace rank d and number of subspace L respectively.

7.1.1 Methods

To test our methods for all parameters, we scan through an exponential grid of λ ranging from $\sqrt{n}\times 10^{-2}$ to $\sqrt{n}\times 10^{3}$. In all experiments, ambient dimension $n = 100$, relative sampling $\kappa = 5$, subspace and data are drawn uniformly at random from unit sphere

8. Freely available at: http://www.stanford.edu/~boyd/papers/admm/

and then corrupted by Gaussian noise $Z_{ij} \sim N(0, \sigma/\sqrt{n})$. We measure the success of the algorithm by the relative violation of Self-Expressiveness Property defined below.

$$\text{RelViolation}\,(C, \mathcal{M}) = \frac{\sum_{(i,j)\notin\mathcal{M}} |C|_{i,j}}{\sum_{(i,j)\in\mathcal{M}} |C|_{i,j}}$$

where $\mathcal{M}$ is the ground truth mask containing all (i,j) such that $x_i, x_j \in \mathcal{X}^{(\ell)}$ for some ℓ. Note that $\text{RelViolation}\,(C,\mathcal{M}) = 0$ implies that SEP is satisfied. We also check that there is no all-zero columns in C, and the solution is considered trivial otherwise.

7.1.2 Results

The simulation results confirm our theoretical findings. In particular, Figure 8a shows that LASSO subspace detection property is possible for a very large range of λ and the dependence on noise magnitude is roughly $1/\sigma$ as predicted in (4.1). Comparing the Figure 8a and 8b, it is clear that exact subspace detection property is not necessary for perfect classification. In addition, the sharp contrast of Figure 9a and 9b demonstrates our observations on the sensitivity of d and L. Observe that in Figure 9a beyond a point, subspace detection property is not possible for any λ. Whereas in Figure 9b even at the point when $dL = 200$ (subspaces are highly dependent), subspace detection property still holds for a large range of λ.

7.2 Face Clustering Experiments

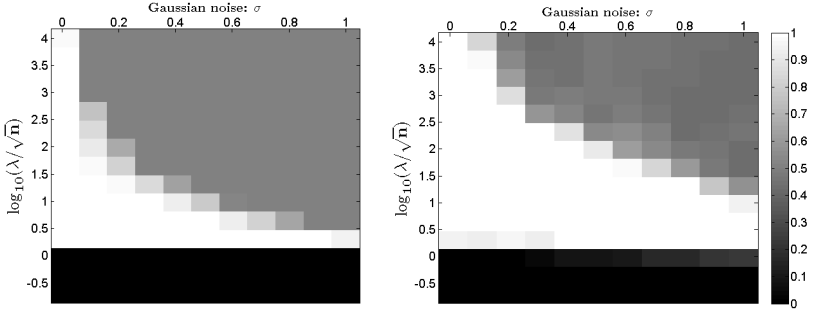
In this section, we evaluate the noise robustness of with LASSO-SSC on Extended YaleB (Lee et al., 2005), a real life face data set of 38 subjects. For each subject, 64 face images are taken under different illuminations.

7.2.1 Subspace Modeling of Face Images

According to Basri and Jacobs (2003), face images under different illuminations can be well-approximated by a 9-dimensional linear subspace. In addition, Zhou et al. (2007) reveals the underlying 3-dimensional subspace structure by assuming Lambert's reflectance and ignoring the shadow pixels. For the physics of this subspace model, we refer the readers to Basri and Jacobs (2003) and Zhou et al. (2007) for detailed explanations.

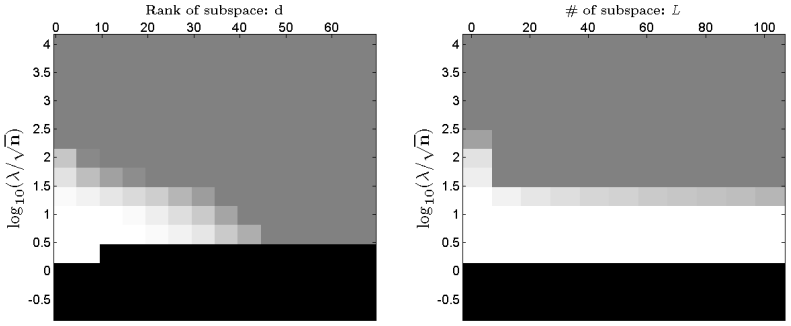
Method: We conduct face clustering experiments on Extended YaleB data set with both 9D and 3D representations of face images and compare them under varying number of subspaces L and different level of injected noise. Specifically, the 9D subspaces are generated by projecting the data matrix corresponding to each subject to a 9D subspace via PCA and the 3D subspaces are generated by a factorization-based robust matrix completion method (Wang et al., 2015). Then we scan through a random selection of $[2, 4, 6, 10, 12, 18, 38]$ subjects and for each experiment we inject additive Gaussian noise $N(0, \sigma/\sqrt{n})$ with $\sigma = [0, 0.01, ..., 0.99, 1]$[9]. Each photo is resized to 48×42 so we have ambient dimension $n = 2016$ and there are 64 sample points for each subspace, hence $N = 64L$.

9. In order to compare the effect of varying level of noise, we choose to inject artificial noise in this experiment. The performance of LASSO-SSC on real noise/data corruptions is well-documented in the motion segmentation experiments of Elhamifar and Vidal (2009, 2013)



(a) Effect of noise level on SEP. (b) Effect of noise level on clustering accuracy.

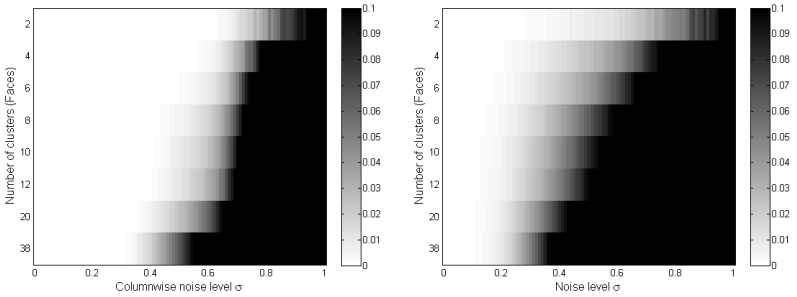
Figure 8: Effects of noise level on LASSO-SSC performance. Simulated under the fully random model with $n = 100, d = 4, L = 3, \kappa = 5$ against increasing Gaussian noise $N(0, \sigma/\sqrt{n})$. For each setting, results are plotted in grayscale across an exponential grid of tuning parameters λ along the vertical axis. **(a)** shows the effect of noise on subspace detection property (Definition 1). In particular, **Black** means trivial solution ($C = 0$); **Gray** means RelViolation > 0.1, and **White** denotes RelViolation $= 0$. **(b)** shows the corresponding clustering accuracy on the same experiment. The rate of accurate classification is represented in grayscale, where white region means perfect classification.



(a) Effect of cluster rank d. (b) Effect of number of subspaces L.

Black: trivial solution ($C = 0$); **Gray:** RelViolation > 0.1; **White:** RelViolation $= 0$.

Figure 9: Effects of other model parameters on LASSO-SSC performance. Simulated under the fully random model with default parameter $n = 100, d = 2, L = 3, \kappa = 5, \sigma = 0.2$. In **(a)** varies subspace dimension d **(b)** varies the number of subspaces L. For each setting, results are plotted in grayscale across an exponential grid of tuning parameters λ along the vertical axis.



(a) Rank 3 photometric face. (b) Rank 9 faces (after projection).

Figure 10: RelViolation of the two face clustering experiments.

The parameter λ is not carefully tuned, but simply chosen to be $\sqrt{n}$, which is order-wise correct for small σ according to (4.1).

7.2.2 Results

As we can see in Figure 10a and 10b, the range where LASSO-Subspace Detection Property holds is much larger for the rank-3 experiments than the rank-9 experiments. Also, the recovery is not sensitive to the number of faces we want to cluster. Indeed, LASSO-SSC is able to succeed for both rank-9 and rank-3 data with a considerable range of noise even for the full 38 subjects clustering task.

These observations confirm our theoretical and simulation results—on deterministic subspace data from a real-life problem—that noise robustness of LASSO-SSC is sensitive to the subspace dimension d but not the number of subspaces L.

8. Conclusion and Future Directions

We presented a theoretical analysis for noisy subspace segmentation problem that is of great practical interests. We showed that the popular SSC algorithm *exactly* (not approximately) detects the subspaces in the noisy case, which justified its empirical success on real problems. Our results are the first in showing LASSO-SSC to work under deterministic data and noise. For stochastic noise, we show that LASSO-SSC works even when noise is much larger than the signal. In addition, we discovered the orderwise relationship between LASSO-SSC's robustness to the level of noise and the subspace dimension, and we found that robustness is insensitive to the number of subspaces. These results lead to new theoretical understanding of SSC, and provide guidelines for practitioners and application-level researchers to judge whether SSC could possibly work well for their respective applications.

Open problems for subspace clustering include the graph connectivity problem (Nasihatkon and Hartley, 2011), missing data problem (a first attempt in Eriksson et al. (2012), which unfortunately requires an unrealistic number of data), sparse corruptions on data and

others. One direction closely related to this paper is to introduce a more practical metric of success. As we illustrated in the paper, subspace detection property is not necessary for perfect clustering. In fact from a pragmatic point of view, even perfect clustering is not necessary. Typical applications allow for a small number of misclassifications. It would be interesting to see whether stronger robustness results can be obtained for a more practical metric of success.

Acknowledgments

The work of H. Xu was partially supported by the Ministry of Education of Singapore through AcRF Tier Two grant R-265-000-443-112, and A*STAR SERC PSF grant R-265-000-540-305. The authors would like to thank the associate editor and the anonymous reviewers for their constructive comments that lead to significant improvement of this paper.

Appendix A. Differences to Soltanolkotabi et al. (2014)

As we reviewed above, Soltanolkotabi et al. (2014) analyzed almost the same algorithm under the semi-random model. Besides the comparisons we made in Section 2 regarding the model of analysis and allowable noise level, there are a few other minor differences which we list here.

"Non-trivial" v.s. "Many true discoveries". In LASSO-Subspace Detection Property, we only require the resulting regression coefficient to be non-zero, while Soltanolkotabi et al. (2014, Theorem 3.2) has a result showing the conditions under which the number of non-zero coefficient is a constant fraction of subspace dimension d. Our results are weaker but more general (works without the semi-random assumption).

In fact, the conditions are more similar than different. We both pick regression coefficient of the same order in the semi-random model. In addition, when d becomes smaller than $\log \rho(i)/c_0$, these two conditions are essentially the same.

Choosing parameter λ. Soltanolkotabi et al. (2014) provides a two-pass mechanism to adaptively tune the parameter λ for each LASSO-SSC and the results are proven for this particular λ. On the other hand, our results are stated for any λ in a specified range. The choice of λ can also be independently tuned for each LASSO-SSC.

In practice, it is advisable to choose λ slightly larger than what is required for it to be non-trivial. We described two strategies in the discussion underneath Theorem 6.

Proof techniques. The proofs are admittedly similar in many ways (since we solve the same problem!), but the key technical components in controlling the magnitude of dual variables ν_1 and ν_2 are different. Soltanolkotabi et al. (2014, Lemma 8.5) relies on the semi-random model, and the resulting restricted isometry property (of the block of data points corresponding to each subspace). In contrast, our bound for $\|\nu_2\|$ does not require any probabilistic assumptions, therefore more general. It is however looser than Soltanolkotabi et al. (2014, Lemma 8.5) by a factor of $\sqrt{d}$ when we specialized in the semi-random model. This is probably what led to our worse dependence on the subspace dimension d in the bound we described in the discussion of Theorem 10.

In conclusion, we find that our results are complementary to that in Soltanolkotabi et al. (2014) and provide a novel point of view to the theoretical analysis for subspace clustering problems.

Appendix B. Numerical algorithm to solve Matrix-LASSO-SSC

In this section we outline the steps of solving the matrix version of LASSO-SSC below. Note that Elhamifar and Vidal (2013) derived a more general version of Matrix-LASSO-SSC to account for not only noisy but also sparse corruptions. We include this appendix merely for the convenience of the readers. Consider

$$\min_C \|C\|_1 + \frac{\lambda}{2}\|X - XC\|_F^2 \quad \text{s.t.} \quad \operatorname{diag}(C) = 0. \tag{B.1}$$

While this convex optimization problem can be solved by some off-the-shelf general purpose solvers such as SeDuMi (Sturm, 1999) or SDPT3 (Toh et al., 1999), such approaches are usually slow and non-scalable. An ADMM (Boyd et al., 2011) version of the problem is described here for fast computation. It solves an equivalent optimization program

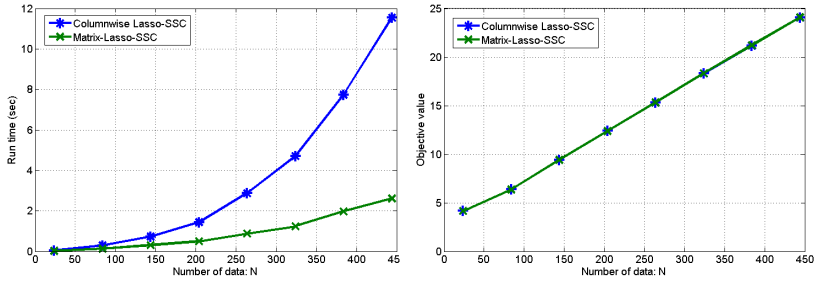
$$\min_C \|C\|_1 + \frac{\lambda}{2}\|X - XJ\|_F^2 \quad \text{s.t.} \quad J = C - \operatorname{diag}(C). \tag{B.2}$$

We add to the Lagrangian with an additional quadratic penalty term for the equality constraint and get the augmented Lagrangian

$$\mathcal{L} = \|C\|_1 + \frac{\lambda}{2}\|X - XJ\|_F^2 + \frac{\mu}{2}\|J - C + \operatorname{diag}(C)\|_F^2 + tr(\Lambda^T(J - C + \operatorname{diag}(C))),$$

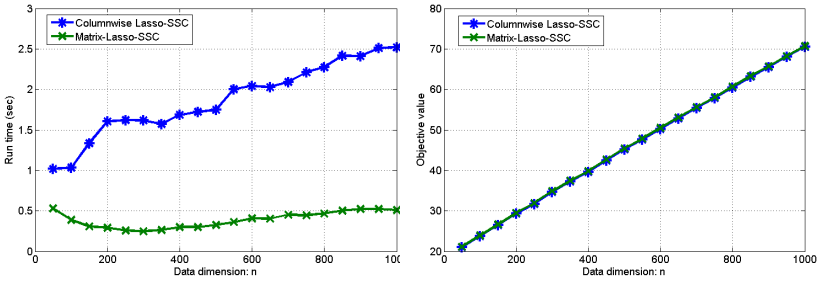
where Λ is the dual variable and μ is a parameter. Optimization is done by alternatingly optimizing over J, C and Λ until convergence. The update steps are derived by solving $\partial\mathcal{L}/\partial J = 0$ and $\partial\mathcal{L}/\partial C = 0$. Notice that the objective function is non-differentiable for C at origin so we use the now standard soft-thresholding operator (Donoho, 1995). For both variables, the solution is given in closed-forms. For the update of Λ, we simply use the gradient descent method. For details of the ADMM algorithm and its guarantee, please refer to Boyd et al. (2011). To accelerate the convergence, it is possible to introduce a parameter ρ and increase μ by $\mu = \rho\mu$ at every iteration. The full algorithm is summarized in Algorithm 1.

Note that for the special case when $\rho = 1$, the inverse of $(\lambda Y^T Y + \mu I)$ can be precomputed, and hence each iteration can be computed in linear time. Empirically, we found it good to set $\mu = \lambda$ and it takes roughly 50-100 iterations to converge to a sufficiently good points. We remark that the matrix version of the algorithm is much faster than the column-by-column ADMM-Lasso and achieves almost the same numerical accuracy; see our experiments in Figure 11a,11b,12a and 12b.



(a) Running time comparison. (b) Objective value comparison.

Figure 11: Comparisons of the two optimization procedures with an increasing number of data. Simulated under fully random model with $n = 100, d = 4, L = 3, \sigma = 0.2$, κ increases from 2 to 40 such that the number of data goes from 24 to 480. From Figure **(a)**, it appears that the matrix version scales better with increasing number of data compared to columnwise LASSO. Figure **(b)** shows that the objective value obtained at stop points of two algorithms are nearly the same.



(a) Running time comparison. (b) Objective value comparison.

Figure 12: Comparisons of the two optimization procedures with an increasing ambient dimension. Simulated with $\kappa = 5, d = 4, L = 3, \sigma = 0.2$, ambient dimension n increases from 50 to 1000. Note that the dependence on dimension is weak for both algorithm at this small scale due to the fast vectorized computation. Nevertheless, it is clear from Figure 12a that the matrix version of SSC runs faster. Figure **(b)** shows that the objective value obtained at stop points of two algorithms are nearly the same.

Algorithm 1 Matrix-LASSO-SSC

Input: Data points as columns in $X \in \mathbb{R}^{n \times N}$, tradeoff parameter λ, numerical parameters μ_0 and ρ.
Initialize $C = 0$, $J = 0$, $\Lambda = 0$, $k = 0$.
while not converged **do**

1. Update J by
$$J = (\lambda X^T X + \mu_k I)^{-1}(\lambda X^T X + \mu_k C - \Lambda).$$
2. Update C by
$$C' = \text{SoftThresh}_{\frac{1}{\mu_k}}\left(J + \Lambda/\mu_k\right),$$
$$C = C' - \text{diag}(C').$$
3. Update Λ by
$$\Lambda = \Lambda + \mu_k(J - C)$$
4. Update parameter $\mu_{k+1} = \rho\mu_k$.
5. Iterate $k = k + 1$;

end while
Output: Affinity matrix $W = |C| + |C|^T$

References

Keith Ball. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31:1–58, 1997.

Ronen Basri and David W Jacobs. Lambertian reflectance and linear subspaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(2):218–233, 2003.

A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

D. Bertsimas and M. Sim. The price of robustness. *Operations research*, 52(1):35–53, 2004.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Paul S Bradley and Olvi L Mangasarian. k-plane clustering. *Journal of Global Optimization*, 16(1):23–32, 2000.

René Brandenberg, Abhi Dattasharma, Peter Gritzmann, and David Larman. Isoradial bodies. *Discrete & Computational Geometry*, 32(4):447–457, 2004.

Emmanuel J Candès. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008.

Guangliang Chen and Gilad Lerman. Spectral curvature clustering (scc). *International Journal of Computer Vision*, 81(3):317–330, 2009.

Yudong Chen, Ali Jalali, Sujay Sanghavi, and Huan Xu. Clustering partially observed graphs via convex optimization. *The Journal of Machine Learning Research*, 15(1):2213–2238, 2014.

Joao Paulo Costeira and Takeo Kanade. A multi-body factorization method for motion analysis. In *International Conference on Computer Vision (ICCV-95)*, pages 1071–1076. IEEE, 1995.

João Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random structures and algorithms*, 22(1):60–65, 2003.

David L Donoho. De-noising by soft-thresholding. *Information Theory, IEEE Transactions on*, 41(3):613–627, 1995.

David L Donoho, Michael Elad, and Vladimir N Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *Information Theory, IEEE Transactions on*, 52(1):6–18, 2006.

Eva L Dyer, Aswin C Sankaranarayanan, and Richard G Baraniuk. Greedy feature selection for subspace clustering. *The Journal of Machine Learning Research*, 14(1):2487–2517, 2013.

Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition (CVPR-09)*, pages 2790–2797. IEEE, 2009.

Ehsan Elhamifar and René Vidal. Clustering disjoint subspaces via sparse representation. In *Acoustics Speech and Signal Processing (ICASSP-10)*, pages 1926–1929. IEEE, 2010.

Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(11): 2765–2781, 2013.

Brian Eriksson, Laura Balzano, and Robert Nowak. High-rank matrix completion. In *International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, pages 373–381, 2012.

Trevor Hastie and Patrice Y Simard. Metrics and models for handwritten character recognition. *Statistical Science*, pages 54–65, 1998.

Reinhard Heckel and Helmut Bölcskei. Noisy subspace clustering via thresholding. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 1382–1386. IEEE, 2013.

Kenichi Kanatani. Motion segmentation by subspace separation and model selection. In *International Conference on Computer Vision (ICCV-01)*, volume 2, pages 586–591. IEEE, 2001.

Fabien Lauer and Christoph Schnorr. Spectral clustering of linear subspaces for motion segmentation. In *International Conference on Computer Vision (ICCV-09)*, pages 678–685. IEEE, 2009.

Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(5):684–698, 2005.

G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013.

Guangcan Liu, Zhouchen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *International Conference on Machine Learning (ICML-10)*, pages 663–670, 2010.

Guangcan Liu, Huan Xu, and Shuicheng Yan. Exact subspace segmentation and outlier detection by low-rank representation. In *International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, pages 703–711, 2012.

Behrooz Nasihatkon and Richard Hartley. Graph connectivity in sparse subspace clustering. In *Computer Vision and Pattern Recognition (CVPR-11)*, pages 2137–2144. IEEE, 2011.

Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems (NIPS-02)*, 2:849–856, 2002.

Shankar R Rao, Roberto Tron, René Vidal, and Yi Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *Computer Vision and Pattern Recognition (CVPR-08)*, pages 1–8. IEEE, 2008.

Mahdi Soltanolkotabi, Emmanuel J Candes, et al. A geometric analysis of subspace clustering with outliers. *The Annals of Statistics*, 40(4):2195–2238, 2012.

Mahdi Soltanolkotabi, Ehsan Elhamifar, Emmanuel J Candes, et al. Robust subspace clustering. *The Annals of Statistics*, 42(2):669–699, 2014.

Jos F Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1-4):625–653, 1999.

Ryan J Tibshirani et al. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7:1456–1490, 2013.

Kim-Chuan Toh, Michael J Todd, and Reha H Tütüncü. Sdpt3a matlab software package for semidefinite programming, version 1.3. *Optimization methods and software*, 11(1-4): 545–581, 1999.

Roberto Tron and René Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *Computer Vision and Pattern Recognition (CVPR-07)*, pages 1–8. IEEE, 2007.

P Tseng. Nearest q-flat to m points. *Journal of Optimization Theory and Applications*, 105 (1):249–252, 2000.

René Vidal. A tutorial on subspace clustering. *IEEE Signal Processing Magazine*, 28(2): 52–68, 2010.

René Vidal, Stefano Soatto, Yi Ma, and Shankar Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 1, pages 167–172. IEEE, 2003.

Rene Vidal, Yi Ma, and Shankar Sastry. Generalized principal component analysis (GPCA). *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(12):1945–1959, 2005.

Yu-Xiang Wang and Huan Xu. Noisy sparse subspace clustering. In *International Conference on Machine Learning (ICML-13)*, pages 89–97, 2013.

Yu-Xiang Wang, Huan Xu, and Chenlei Leng. Provable subspace clustering: When LRR meets SSC. In *Advances in Neural Information Processing Systems (NIPS-13)*, pages 64–72, 2013.

Yu-Xiang Wang, Choon Meng Lee, Loong-Fah Cheong, and Kim-Chuan Toh. Practical matrix completion and corruption recovery using proximal alternating robust subspace minimization. *International Journal of Computer Vision*, 111(3):315–344, 2015.

Jingyu Yan and Marc Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Computer Vision–ECCV 2006*, pages 94–106. Springer, 2006.

A. Zhang, N. Fawaz, S. Ioannidis, and A. Montanari. Guess who rated this movie: Identifying users through subspace clustering. *arXiv preprint arXiv:1208.1544*, 2012.

Shaohua Kevin Zhou, Gaurav Aggarwal, Rama Chellappa, and David W Jacobs. Appearance characterization of linear lambertian objects, generalized photometric stereo, and illumination-invariant face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(2):230–245, 2007.

Journal of Machine Learning Research 17 (2016) 1-36 Submitted 8/14; Revised 12/15; Published 3/16

Learning the Variance of the Reward-To-Go

Aviv Tamar AVIVT@BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94709, USA

Dotan Di Castro DOT@YAHOO-INC.COM
Yahoo! Research Labs
MATAM, Haifa 31905, Israel

Shie Mannor SHIE@EE.TECHNION.AC.IL
Department of Electrical Engineering
The Technion - Israel Institute of Technology
Haifa 32000, Israel

Editor: Peter Auer

Abstract

In Markov decision processes (MDPs), the variance of the reward-to-go is a natural measure of uncertainty about the long term performance of a policy, and is important in domains such as finance, resource allocation, and process control. Currently however, there is no tractable procedure for calculating it in large scale MDPs. This is in contrast to the case of the expected reward-to-go, also known as the value function, for which effective simulation-based algorithms are known, and have been used successfully in various domains. In this paper[1] we extend temporal difference (TD) learning algorithms to estimating the variance of the reward-to-go for a fixed policy. We propose variants of both TD(0) and LSTD(λ) with linear function approximation, prove their convergence, and demonstrate their utility in an option pricing problem. Our results show a dramatic improvement in terms of sample efficiency over standard Monte-Carlo methods, which are currently the state-of-the-art.

Keywords: Reinforcement learning, Markov decision processes, variance estimation, simulation, temporal differences

1. Introduction

In sequential decision making within the Markov Decision Process (MDP) framework, whether in a planning setting (Puterman, 1994; Powell, 2011) or a reinforcement learning setting (RL; Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998), the decision maker ultimately obtains a policy π, often with some guarantees on its expected long-term performance. This typical conclusion of the policy optimization process is the starting point of our work.

We consider the policy π to be fixed[2], and we are interested in understanding how π performs in practice, with the natural quantity of interest being the reward-to-go from each state of the system. The *expected* reward-to-go J, also known as the *value function*, is often a part of an optimization process, and efficient methods for learning it are well known. In many applications, however, looking at expectations is not enough, and it seems only reasonable to estimate other statistics of the reward-to-go, such as its *variance*, denoted by V. Quite surprisingly, this topic has received very little attention; at the current state-of-the-art, the only solution for large-scale MDPs is a naive Monte-Carlo approach, demanding extensive simulations of the long-term outcomes *from each system state*. In this paper we explore much more efficient alternatives.

We further motivate policy evaluation with respect to the variance of the reward-to-go. The variance is an intuitive measure of uncertainty, and common practice in many domains such as finance, process control, and clinical decision making (Sharpe, 1966; Shortreed et al., 2011). As we show in the paper, in an option pricing domain, the uncertainty captured by the variance of the reward-to-go highlights important properties of the policy, that are not visible by looking at the value function alone.

The variance may also be used for policy selection. In some practical situations, a full policy-optimization procedure is not an option, and the agent can only select between several predefined policies. For example, it may be that each policy is designed by an expert (e.g., a private equity or investment fund), and the agent can simply select between several policies, given the current features of the system (e.g., current economic indicators). A related financial experiment in a non-sequential setting was reported by Moody and Saffell (2001), which selected between several investment types, and which emphasized the importance of incorporating variance-based objectives in the policy selection.

Finally, the value function has proved to be a fundamental ingredient in many policy optimization algorithms. The variance of the reward-to-go may thus prove valuable for *risk aware* optimization algorithms, a topic that has gained significant interest recently (Filar et al., 1995; Mihatsch and Neuneier, 2002; Geibel and Wysotzki, 2005; Mannor and Tsitsiklis, 2013). Therefore, the policy evaluation methods in this work may be used as a sub-procedure in policy optimization. Since the conference publication of this work, this idea has already been explored by Tamar and Mannor (2013) and Prashanth and Ghavamzadeh (2013). Both Tamar and Mannor (2013) and Prashanth and Ghavamzadeh (2013) suggested actor-critic algorithms, in which the critic uses the policy evaluation ideas introduced in this paper. We are certain that risk-aware policy evaluation would play a major role in future risk-aware optimization algorithms as well.

The principal challenge in policy evaluation arises when the state space is large, or continuous. Then, solving Bellman's equation for the value or its extension (Sobel, 1982) for the variance becomes intractable. This difficulty is even more pronounced in the learning setting, when a model of the process is not available, and the evaluation has to be *estimated* from a limited amount of samples. Fortunately, for the case of the value function, effective learning approaches are known.

Temporal Difference methods (TD; Sutton, 1988) typically employ *function approximation* to represent the value function in a lower dimensional subspace, and *learn* the approximation parameters efficiently, by fitting the spatiotemporal relations in Bellman's equation to the observed (or simulated) data. TD methods have been studied extensively, both theoretically (Bertsekas, 2012, Lazaric et al., 2010) and empirically (e.g., Tesauro, 1995, Powell, 2011, Section 14.5), and are considered to be the state-of-the-art in policy evaluation.

1. This paper extends an earlier work by the authors (Tamar et al., 2013).
2. This setting is also known as a Markov reward process.

However, when it comes to evaluating additional statistics of the reward-to-go, such as its variance, little is known. This may be due to the fact that the linearity of the expectation in Bellman's equation plays a key role in TD algorithms.

In this paper we present a TD framework for learning the variance of the reward-to-go, using function approximation, in problems where a model is not available, or too large to solve. To our knowledge, this is the first work that addresses the challenge of large state spaces, by considering an approximation scheme for the variance. Our approach is based on the following observation: the second moment of the reward-to-go, denoted by M, together with the value function J, satisfies a linear 'Bellman-like' equation. By extending TD methods to jointly estimate J and M with linear function approximation, we obtain a solution for estimating the variance, using the relation $V = M - J^2$.

We propose both a variant of Least Squares Temporal Difference (LSTD; Boyan 2002) and of TD(0) (Sutton and Barto, 1998) for jointly estimating J and M with linear function approximation. For these algorithms, we provide convergence guarantees and error bounds. In addition, we introduce novel methods for enforcing the approximate variance to be positive, through a constrained TD equation or through an appropriate choice of features. An empirical evaluation of our approach on an American-style option pricing problem demonstrates a dramatic improvement in terms of sample efficiency compared to Monte Carlo techniques—the current state of the art.

A previous study by Sato et al. (2001) suggested TD equations for J and V, without function approximation. Their approach relied on a non-linear equation for V, and it is not clear how it may be extended to handle large state spaces. More recently, Morimura et al. (2010) proposed TD learning rules for a parametric distribution of the return, albeit without function approximation nor formal guarantees. In the Bayesian Gaussian process temporal difference framework of Engel et al. (2005), the reward-to-go is assumed to have a Gaussian posterior distribution, and its mean and variance are estimated. However, the resulting variance is a product of both stochastic transitions and model uncertainty, and is thus different than the variance considered here. For average reward MDPs, several studies (e.g., Filar et al., 1989) considered the variation of the reward from its average. This measure of variability is not suitable for the discounted and episodic settings considered here. A different line of work considers MDPs with a *dynamic-risk* measure (Ruszczyński, 2010). In dynamic risk, instead of considering the reward-to-go as the random variable of interest, the risk is defined iteratively over the possible future trajectories. In an optimization setting, dynamic-risk has some favorable properties such as time-consistency, and a dynamic programming formulation (Ruszczyński, 2010). However, the variance of the reward-to-go considered here is considerably more intuitive, and leads to a much simpler approach.

This paper is organized as follows. In Section 2 we present our formal MDP setup. In Section 3 we derive the fundamental equations for jointly approximating J and M, and discuss their properties. A solution to these equations may be obtained by sampling, through the use of TD algorithms, as presented in Section 4. As it turns out, our approximation scheme may result in cases where the approximate variance is negative. We discuss this in Section 5, and propose methods for avoiding it. Section 6 presents an empirical evaluation on an option pricing problem, and Section 7 concludes, and discusses future directions.

2. Framework and Background

We consider an episodic MDP[3] (also known as a stochastic shortest path problem; Bertsekas 2012) in discrete time with a finite state space $X \triangleq \{1, \ldots, n\}$ and a terminal state x^*. A *fixed* policy π determines, for each $x \in X$, a stochastic transition to a subsequent state $x' \in \{X \cup x^*\}$ with probability $P(x'|x)$. We consider a deterministic and bounded reward function $r : X \to \mathbb{R}$, and assume zero reward at the terminal state. We denote by x_k the state at time k, where $k = 0, 1, 2, \ldots$.

A policy is said to be *proper* (Bertsekas, 2012) if there is a positive probability that the terminal state x^* will be reached after at most n transitions, from any initial state. Throughout this paper we make the following assumption:

Assumption 1 *The policy π is proper.*

Let $\gamma \in (0, 1]$ denote a discount factor. We emphasize that the case $\gamma = 1$, corresponding to a non-discounted setting, is allowed, and much of our effort in the sequel is to handle this special and important case. Let $\tau \triangleq \min\{k > 0 | x_k = x^*\}$ denote the first visit time to the terminal state, and let the random variable B denote the accumulated (and possibly discounted) reward along the trajectory until that time

$$B \triangleq \sum_{k=0}^{\tau-1} \gamma^k r(x_k).$$

In this work, we are interested in the mean-variance tradeoff in B, represented by the *value function*

$$J(x) \triangleq \mathbb{E}\left[B | x_0 = x\right], \quad x \in X,$$

and the *variance of the reward-to-go*

$$V(x) \triangleq \mathrm{Var}\left[B | x_0 = x\right], \quad x \in X.$$

We will find it convenient to define also the *second moment of the reward-to-go*

$$M(x) \triangleq \mathbb{E}\left[B^2 | x_0 = x\right], \quad x \in X.$$

Our goal is to estimate the functions $J(x)$ and $V(x)$ from trajectories obtained by simulating the MDP with policy π.

3. Approximation of the Variance of the Reward-To-Go

In this section we derive a projected equation method for approximating $J(x)$ and $M(x)$ using linear function approximation. The approximation of $V(x)$ will then follow from the relation $V(x) = M(x) - J(x)^2$.

Our starting point is a system of equations for $J(x)$ and $M(x)$, first derived by Sobel (1982) for a discounted infinite horizon case, and extended here to the episodic case. The equation for J is the well known Bellman equation for a fixed policy, and independent of the equation for M.

3. In particular, any finite horizon MDP is an episodic MDP, for which our results apply. Extending these results to the infinite horizon discounted setting is straightforward.

Proposition 2 *The following equations hold for $x \in X$*

$$\begin{aligned} J(x) &= r(x) + \gamma \sum_{x' \in X} P(x'|x) J(x'), \\ M(x) &= r(x)^2 + 2\gamma r(x) \sum_{x' \in X} P(x'|x) J(x') + \gamma^2 \sum_{x' \in X} P(x'|x) M(x'). \end{aligned} \tag{1}$$

Furthermore, under Assumption 1 a unique solution to Eq. (1) *exists.*

A straightforward proof is given in Appendix A.

At this point the reader may wonder why an equation for V is not presented. While such an equation may be derived (see, e.g., Sobel 1982, Tamar et al. 2012), it is not linear. The linearity of (1) in J and M is the key to our approach. As we show in the next subsection, the solution to (1) may be expressed as the fixed point of a linear mapping in the joint space of J and M. We will then show that a projection of this mapping onto a linear feature space is contracting, thus allowing us to use the TD methodology to estimate J and M.

3.1 A Projected Fixed Point Equation in the Joint Space of J and M

For the sequel, we introduce the following vector notations. We denote by $P \in \mathbb{R}^{n \times n}$ and $r \in \mathbb{R}^n$ the episodic MDP transition matrix and reward vector, i.e., $P_{x,x'} = P(x'|x)$ and $r_x = r(x)$, where $x, x' \in X$. Also, we define the diagonal matrix $R \triangleq diag(r)$.

For a vector $z \in \mathbb{R}^{2n}$ we let $z_J \in \mathbb{R}^n$ and $z_M \in \mathbb{R}^n$ denote its leading and ending n components, respectively. Thus, such a vector belongs to the joint space of J and M.

We define the mapping $T : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ by

$$\begin{aligned} [Tz]_J &= r + \gamma P z_J, \\ [Tz]_M &= Rr + 2\gamma RP z_J + \gamma^2 P z_M. \end{aligned} \tag{2}$$

It may easily be verified that a fixed point of T is a solution to (1), and by Proposition 2 such a fixed point exists and is unique.

When the state space X is large, however, a direct solution of (1) is not feasible. A popular approach in this case is to approximate $J(x)$ by restricting it to a lower dimensional subspace, and use simulation based TD algorithms to *learn* the approximation parameters (Bertsekas, 2012). In this paper we extend this approach to the approximation of $M(x)$ as well.

We consider a linear approximation architecture of the form

$$\tilde{J}(x) = \phi_J(x)^\top w_J, \quad \tilde{M}(x) = \phi_M(x)^\top w_M,$$

where $w_J \in \mathbb{R}^l$ and $w_M \in \mathbb{R}^m$ are the approximation parameter vectors, $\phi_J(x) \in \mathbb{R}^l$ and $\phi_M(x) \in \mathbb{R}^m$ are state dependent features, and $(\cdot)^\top$ denotes the transpose of a vector. The low dimensional subspaces are therefore

$$S_J = \{\Phi_J w | w \in \mathbb{R}^l\}, \quad S_M = \{\Phi_M w | w \in \mathbb{R}^m\},$$

where Φ_J and Φ_M are matrices whose rows are $\phi_J(x)^\top$ and $\phi_M(x)^\top$, respectively. We make the following standard independence assumption on the features.

Assumption 3 *The matrix Φ_J has rank l and the matrix Φ_M has rank m.*

We now discuss how the approximation parameters w_J and w_M are chosen. The idea behind TD methods is to fit the approximate $\tilde{J}$ and $\tilde{M}$ to obey Eq. (1) in some sense. Specifically, this is done by considering a *projection* of T onto the approximation subspaces S_J and S_M, and choosing $\tilde{J}$ and $\tilde{M}$ as the unique fixed point of this projected operator. As outlined earlier, our ultimate goal is to learn w_J and w_M from simulated trajectories of the MDP. Thus, it is constructive to consider projections onto S_J and S_M with respect to a norm that is weighted according to the state occupancy in these trajectories. We now define this projection.

For a trajectory $x_0, \ldots, x_{\tau-1}$, where x_0 is drawn from a fixed distribution $\zeta_0(x)$, and the states evolve according to the MDP with policy π, define the state occupancy probabilities

$$q_t(x) = P(x_t = x), \quad x \in X, \quad t = 0, 1, \ldots,$$

and let

$$\begin{aligned} q(x) &= \sum_{t=0}^{\infty} q_t(x), \quad x \in X, \\ Q &\triangleq diag(q). \end{aligned}$$

We make the following assumption on the policy π and initial distribution ζ_0

Assumption 4 *Each state has a positive probability of being visited, namely, $q(x) > 0$ for all $x \in X$.*

Note that if ζ_0 may be controlled (for example, if we have access to a simulator), Assumption 4 may be trivially satisfied by choosing a positive ζ_0 for all states. Alternatively, if some state has a zero probability of being visited under π, then it is irrelevant for policy evaluation, and we can remove it from the state space. In this case, so long as Assumption 3 still holds (i.e., the linear features remain identifiable) all our subsequent derivations remain valid.

For vectors in $\mathbb{R}^n$, we recall the weighted Euclidean norm

$$\|y\|_q = \sqrt{\sum_{i=1}^{n} q(i) \left(y(i)\right)^2}, \quad y \in \mathbb{R}^n,$$

and we denote by Π_J and Π_M the projections from $\mathbb{R}^n$ onto the subspaces S_J and S_M, respectively, with respect to this norm. Note that the projection operators Π_J and Π_M are linear, and may be written explicitly as $\Pi_J = \Phi_J(\Phi_J^\top Q \Phi_J)^{-1} \Phi_J^\top Q$, and similarly for Π_M.

For some $z \in \mathbb{R}^{2n}$ we denote by Π the projection of z_J onto S_J and z_M onto S_M, namely

$$\Pi = \begin{pmatrix} \Pi_J & 0 \\ 0 & \Pi_M \end{pmatrix}. \tag{3}$$

We are now ready to fully describe our approximation scheme. We consider the *projected* fixed point equation

$$z = \Pi T z, \tag{4}$$

and, letting z^* denote its solution (which we will show to be unique), propose the approximate value function $\tilde{J} = z_J^* \in S_J$ and second moment function $\tilde{M} = z_M^* \in S_M$.

We shall now derive an important property of the projected operator ΠT, namely, that it is a contraction. This leads to the uniqueness of z^*, and to a simple bound on the approximation error. As in regular TD algorithms, this contraction property also underlies the convergence of several sampling-based algorithms, to be presented in the next section.

We begin by stating a well known result (Proposition 7.1.1 of Bertsekas, 2012) regarding the contraction properties of the *projected Bellman operator* $\Pi_J T_J$, where $T_J y = r + \gamma P y$.

Lemma 5 *(Proposition 7.1.1 of Bertsekas, 2012) Let Assumptions 1, 3, and 4 hold. The linear operator P and the projected linear operator $\Pi_J P$ are non-expansions in the $\|\cdot\|_q$ norm, and satisfy*

$$\|\Pi_J P y\|_q \leq \|P y\|_q \leq \|y\|_q \quad \forall y \in \mathbb{R}^n.$$

In addition, $\Pi_J P$ is a contraction in some norm, i.e., there exists some norm $\|\cdot\|_J$ and some $\beta_J < 1$ such that

$$\|\Pi_J P y\|_J \leq \beta_J \|y\|_J, \quad \forall y \in \mathbb{R}^n.$$

Lemma 5 immediately leads to the following result:

Lemma 6 *Let Assumptions 1, 3, and 4 hold. Then, there exists some norm $\|\cdot\|_J$ and some $\beta_J < 1$ such that*

$$\|\gamma \Pi_J P y\|_J \leq \beta_J \|y\|_J, \quad \forall y \in \mathbb{R}^n.$$

Similarly, there exists some norm $\|\cdot\|_M$ and some $\beta_M < 1$ such that

$$\|\gamma \Pi_M P y\|_M \leq \beta_M \|y\|_M, \quad \forall y \in \mathbb{R}^n.$$

Note that for $\gamma < 1$, Lemma 6 holds with the norm $\|\cdot\|_q$ and contraction modulus γ, by the non-expansiveness property of $\Pi_J P$ in Lemma 5. The more difficult case $\gamma = 1$, however, requires the expressions in Lemma 6.

Next, we define a weighted-norm on $\mathbb{R}^{2n}$, in which a parameter α balances between the weight of the J components' norm, and the weight of the M components' norm, as defined in Lemma 6. The intuition behind this weighted-norm, is that by carefully selecting the balance α, we shall show that the contraction properties in Lemma 6 guarantee a contraction property for the projected operator ΠT, in this norm.

Definition 7 *For a vector $z \in \mathbb{R}^{2n}$ and a scalar $0 < \alpha < 1$, the α-weighted norm is*

$$\|z\|_\alpha = \alpha \|z_J\|_J + (1-\alpha)\|z_M\|_M, \tag{5}$$

where $\|\cdot\|_J$ and $\|\cdot\|_M$ are defined in Lemma 6.

Our main result of this section is given in the following proposition, where we show that the projected operator ΠT is a contraction with respect to a suitable α-weighted norm.

Proposition 8 *Let Assumptions 1, 3, and 4 hold. Then, there exists some $0 < \alpha < 1$ and some $\beta < 1$ such that ΠT is a β-contraction with respect to the α-weighted norm, i.e.,*

$$\|\Pi T z_1 - \Pi T z_2\|_\alpha \leq \beta \|z_1 - z_2\|_\alpha, \quad \forall z_1, z_2 \in \mathbb{R}^{2n}.$$

Proof From the definition of ΠT in (2) and (3), we have that for any $z_1, z_2 \in \mathbb{R}^{2n}$ we have $\|\Pi T z_1 - \Pi T z_2\|_\alpha = \|\Pi \mathcal{P}(z_1 - z_2)\|_\alpha$, where

$$\Pi \mathcal{P} = \begin{pmatrix} \gamma \Pi_J P & 0 \\ 2\gamma \Pi_M R P & \gamma^2 \Pi_M P \end{pmatrix}.$$

Thus, it suffices to show that for all $z \in \mathbb{R}^{2n}$

$$\|\Pi \mathcal{P} z\|_\alpha \leq \beta \|z\|_\alpha.$$

We will now show that $\|\Pi \mathcal{P} z\|_\alpha$ may be separated into two terms which may be bounded by Lemma 6, and an additional cross term. By balancing α and β, this term may be contained to yield the required contraction.

We have

$$\begin{aligned} \|\Pi \mathcal{P} z\|_\alpha =& \alpha \|\gamma \Pi_J P z_J\|_J \\ &+ (1-\alpha)\|2\gamma \Pi_M R P z_J + \gamma^2 \Pi_M P z_M\|_M \\ \leq& \alpha \|\gamma \Pi_J P z_J\|_J + (1-\alpha)\|\gamma^2 \Pi_M P z_M\|_M \\ &+ (1-\alpha)\|2\gamma \Pi_M R P z_J\|_M \\ \leq& \alpha \beta_J \|z_J\|_J + (1-\alpha)\gamma \beta_M \|z_M\|_M \\ &+ (1-\alpha)\|2\gamma \Pi_M R P z_J\|_M, \end{aligned} \tag{6}$$

where the equality is by definition of the α weighted norm (5), the first inequality is from the triangle inequality, and the second inequality is by Lemma 6. Now, we claim that there exists some finite C such that

$$\|2\gamma \Pi_M R P y\|_M \leq C \|y\|_J, \quad \forall y \in \mathbb{R}^n. \tag{7}$$

To see this, note that since $\mathbb{R}^n$ is a finite dimensional real vector space, all vector norms are equivalent (Horn and Johnson, 2012, Corollary 5.4.5) therefore there exist finite C_1 and C_2 such that for all $y \in \mathbb{R}^n$

$$C_1 \|2\gamma \Pi_M R P y\|_2 \leq \|2\gamma \Pi_M R P y\|_M \leq C_2 \|2\gamma \Pi_M R P y\|_2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm. Let λ denote the spectral norm of the matrix $2\gamma \Pi_M R P$, which is finite since all the matrix elements are finite. We have that

$$\|2\gamma \Pi_M R P y\|_2 \leq \lambda \|y\|_2, \quad \forall y \in \mathbb{R}^n.$$

Using again the fact that all vector norms are equivalent, there exists a finite C_3 such that

$$\|y\|_2 \leq C_3 \|y\|_J, \quad \forall y \in \mathbb{R}^n.$$

Setting $C = C_2 \lambda C_3$ we get the desired bound. Let $\tilde{\beta} = \max\{\beta_J, \gamma \beta_M\} < 1$, and choose $\epsilon > 0$ such that

$$\tilde{\beta} + \epsilon < 1.$$

Now, choose α such that $\alpha = \frac{C}{\epsilon+C}$. We have that

$$(1-\alpha)C = \alpha\epsilon,$$

and plugging this into (7) yields

$$(1-\alpha)\|2\gamma\Pi_M RPy\|_M \leq \alpha\epsilon\|y\|_J. \tag{8}$$

We now return to (6), where we have

$$\begin{aligned}
&\alpha\beta_J\|z_J\|_J + (1-\alpha)\gamma\beta_M\|z_M\|_M + (1-\alpha)\|2\gamma\Pi_M RPz_J\|_M \\
\leq&\alpha\beta_J\|z_J\|_J + (1-\alpha)\gamma\beta_M\|z_M\|_M + \alpha\epsilon\|z_J\|_J \\
\leq&(\tilde{\beta}+\epsilon)\left(\alpha\|z_J\|_J + (1-\alpha)\|z_M\|_M\right),
\end{aligned}$$

where the first inequality is by (8), and the second is by the definition of $\tilde{\beta}$. We have thus shown that

$$\|\Pi\mathcal{P}z\|_\alpha \leq (\tilde{\beta}+\epsilon)\|z\|_\alpha.$$

Finally, choose $\beta = \tilde{\beta}+\epsilon$. ■

Proposition 8 guarantees that the projected operator ΠT has a unique fixed point. Let us denote this fixed point by z^*, and let w_J^*, w_M^* denote the corresponding weights, which are unique due to Assumption 3

$$\begin{aligned}
\Pi T z^* &= z^*, \\
z_J^* &= \Phi_J w_J^*, \\
z_M^* &= \Phi_M w_M^*.
\end{aligned} \tag{9}$$

In the next proposition, using a standard result of Bertsekas and Tsitsiklis (1996), we provide a bound on the approximation error.

Proposition 9 *Let Assumptions 1, 3, and 4 hold. Denote by $z_{true} \in \mathbb{R}^{2n}$ the true value and second moment functions, i.e., $[z_{true}]_J = J$, and $[z_{true}]_M = M$. Then,*

$$\|z_{true} - z^*\|_\alpha \leq \frac{1}{1-\beta}\|z_{true} - \Pi z_{true}\|_\alpha,$$

with α and β defined in Proposition 8.

Proof This result is similar to Lemma 6.9 in Bertsekas and Tsitsiklis (1996). We have

$$\begin{aligned}
\|z_{true} - z^*\|_\alpha \leq& \|z_{true} - \Pi z_{true}\|_\alpha + \|\Pi z_{true} - z^*\|_\alpha \\
=& \|z_{true} - \Pi z_{true}\|_\alpha + \|\Pi T z_{true} - \Pi T z^*\|_\alpha \\
\leq& \|z_{true} - \Pi z_{true}\|_\alpha + \beta\|z_{true} - z^*\|_\alpha.
\end{aligned}$$

Rearranging gives the stated result. ■

Note that by definition, Πz_{true} is the best approximation we can hope for (in terms of the α-weighted squared error) in our approximation subspace. Thus, the approximation error $\|z_{true} - z^*\|_\alpha$ is ultimately bounded by the choice of features, which in practice should be chosen wisely.

At this point, the reader may question the usefulness of the projected fixed-point approximation over simpler approximation schemes, such as the direct projection Πz_{true}. As we show in the next section, the projected fixed-point architecture supports a family of sampling-based TD estimation algorithms, with efficient batch and online implementations. Furthermore, as we show empirically in Section 6, these TD algorithms perform well in practice, especially in the regime of a small sample size. For conventional TD algorithms, these benefits are well-established (Bertsekas, 2012), and gave rise to their popularity. Here we extend this to the variance of the reward-to-go.

4. Simulation Based Estimation Algorithms

In this section we propose algorithms that estimate $\tilde{J}$ and $\tilde{M}$ from sampled trajectories of the MDP, based on the approximation architecture of the previous section.

We begin by writing the projected equation (9) in matrix form. First, let us write the equation explicitly as

$$\begin{aligned}
\Pi_J\left(r + \gamma P\Phi_J w_J^*\right) &= \Phi_J w_J^*, \\
\Pi_M\left(Rr + 2\gamma RP\Phi_J w_J^* + \gamma^2 P\Phi_M w_M^*\right) &= \Phi_M w_M^*.
\end{aligned} \tag{10}$$

Recalling the definition of Q, projecting a vector y onto Φw satisfies the following orthogonality condition

$$\Phi^\top Q(y - \Phi w) = 0.$$

We therefore have

$$\begin{aligned}
\Phi_J^\top Q\left(\Phi_J w_J^* - \left(r + \gamma P\Phi_J w_J^*\right)\right) &= 0, \\
\Phi_M^\top Q\left(\Phi_M w_M^* - \left(Rr + 2\gamma RP\Phi_J w_J^* + \gamma^2 P\Phi_M w_M^*\right)\right) &= 0,
\end{aligned}$$

which can be written as

$$Aw_J^* = b, \quad Cw_M^* = d, \tag{11}$$

with

$$\begin{aligned}
A &= \Phi_J^\top Q\left(I - \gamma P\right)\Phi_J, \quad b = \Phi_J^\top Q r, \\
C &= \Phi_M^\top Q\left(I - \gamma^2 P\right)\Phi_M, \quad d = \Phi_M^\top Q R\left(r + 2\gamma P\Phi_J A^{-1} b\right),
\end{aligned} \tag{12}$$

and the matrices A and C are invertible since Proposition 8 guarantees a unique solution to (9) and Assumption 3 guarantees the unique weights of its projection.

Let us now outline our proposed algorithms. The first algorithm is a variant of the Least Squares Temporal Difference algorithm (LSTD; Boyan 2002), and aims to solve Eq. (11) directly, by forming sample based estimates of the terms A, b, C, and d. This is a batch algorithm that is known to make efficient use of data in its nominal version, and as we show empirically, demonstrates efficient performance in our case as well. The second algorithm

is a variant of online TD(0) (Sutton and Barto, 1998). In its nominal form, TD(0) has been successfully used as the critic in actor-critic algorithms (Konda and Tsitsiklis, 2003). Our extended TD(0) variant may be used similarly in a *risk-adjusted* actor-critic algorithm (Tamar and Mannor, 2013; Prashanth and Ghavamzadeh, 2013). The third algorithm is a variant of LSTD(λ), in which, similarly to standard LSTD(λ), Eq. (11) is extended to its multi-step counterpart. The fourth algorithm is not based on the TD equation (11), but uses least squares regression to estimate the direct projection Πz_{true}. We compare this algorithm with the LSTD variants in Section 6.

4.1 A Least Squares TD Algorithm

Our first simulation-based algorithm is an extension of the LSTD algorithm (Boyan, 2002). We simulate N trajectories of the MDP with the policy π and initial state distribution ζ_0. Let $x_0^k, x_1^k, \ldots, x_{\tau^k-1}^k$ and τ^k, where $k = 0, 1, \ldots, N$, denote the state sequence and visit times to the terminal state within these trajectories, respectively. We now use these trajectories to form the following estimates of the terms in (12)

$$\begin{aligned}
A_N &= \mathbb{E}_N\left[\sum_{t=0}^{\tau-1}\phi_J(x_t)(\phi_J(x_t)-\gamma\phi_J(x_{t+1}))^\top\right],\\
b_N &= \mathbb{E}_N\left[\sum_{t=0}^{\tau-1}\phi_J(x_t)r(x_t)\right],\\
C_N &= \mathbb{E}_N\left[\sum_{t=0}^{\tau-1}\phi_M(x_t)(\phi_M(x_t)-\gamma^2\phi_M(x_{t+1}))^\top\right],\\
d_N &= \mathbb{E}_N\left[\sum_{t=0}^{\tau-1}\phi_M(x_t)r(x_t)\left(r(x_t)+2\gamma\phi_J(x_{t+1})^\top A_N^{-1}b_N\right)\right],
\end{aligned} \tag{13}$$

where $\mathbb{E}_N$ denotes an empirical average over trajectories, i.e., $\mathbb{E}_N[f(x,\tau)] = \frac{1}{N}\sum_{k=1}^N f(x^k,\tau^k)$. The LSTD approximation is given by

$$\hat{w}_J^* = A_N^{-1}b_N, \quad \hat{w}_M^* = C_N^{-1}d_N.$$

The next theorem shows that LSTD converges.

Theorem 10 *Let Assumptions 1, 3, and 4 hold. Then $\hat{w}_J^* \to w_J^*$ and $\hat{w}_M^* \to w_M^*$ as $N \to \infty$ with probability 1.*

The proof involves a straightforward application of the law of large numbers and is described in Appendix B. For regular LSTD, $\mathcal{O}(1/\sqrt{n})$ convergence rates were derived under certain mixing conditions of the MDP by Konda (2002, based on a central limit theorem argument) and Lazaric et al. (2010, based on a finite time analysis), and may be extended to the algorithm presented here.

4.2 An Online TD(0) Algorithm

Our second estimation algorithm is an extension of the well known TD(0) algorithm (Sutton and Barto, 1998). Again, we simulate trajectories of the MDP corresponding to the policy

π and initial state distribution ζ_0, and we iteratively update our estimates at every visit to the terminal state. An extension to an algorithm that updates at every state transition is also possible, but we do not pursue such here.

For some $0 \le t < \tau^k$ and weights w_J, w_M, we introduce the TD terms

$$\begin{aligned}
\delta_J^k(t,w_J,w_M) =& r(x_t^k) + \left(\gamma\phi_J(x_{t+1}^k)^\top - \phi_J(x_t^k)^\top\right)w_J,\\
\delta_M^k(t,w_J,w_M) =& r^2(x_t^k) + 2\gamma r(x_t^k)\phi_J(x_{t+1}^k)^\top w_J\\
&+ \left(\gamma^2\phi_M(x_{t+1}^k)^\top - \phi_M(x_t^k)^\top\right)w_M.
\end{aligned}$$

Note that δ_J^k is the standard TD error (Sutton and Barto, 1998). For the intuition behind δ_M^k, observe that M in (1) is equivalent to the value function of an MDP with stochastic reward $r(x)^2 + 2\gamma r(x)J(x')$, where $x' \sim P(x'|x)$. The TD term δ_M^k is the equivalent TD error, with $\phi_J(x')^\top w_J$ substituting $J(x')$. The TD(0) algorithm is given by

$$\begin{aligned}
\hat{w}_{J;k+1} &= \hat{w}_{J;k} + \xi_k\sum_{t=0}^{\tau^k-1}\phi_J(x_t)\delta_J^k(t,\hat{w}_{J;k},\hat{w}_{M;k}),\\
\hat{w}_{M;k+1} &= \hat{w}_{M;k} + \xi_k\sum_{t=0}^{\tau^k-1}\phi_M(x_t)\delta_M^k(t,\hat{w}_{J;k},\hat{w}_{M;k}),
\end{aligned}$$

where $\{\xi_k\}$ are positive step sizes.

The next theorem shows that TD(0) converges.

Theorem 11 *Let Assumptions 1, 3, and 4 hold, and let the step sizes satisfy*

$$\sum_{k=0}^\infty \xi_k = \infty, \quad \sum_{k=0}^\infty \xi_k^2 < \infty.$$

Then $\hat{w}_{J;k} \to w_J^*$ and $\hat{w}_{M;k} \to w_M^*$ as $k \to \infty$ with probability 1.

Proof The proof is based on representing the algorithm as a stochastic approximation, and uses a result of Borkar (2008) to show that the iterates asymptotically track a certain ordinary differential equation (ODE). This ODE will then be shown to have a unique asymptotically stable equilibrium exactly at w_J^*, w_M^*.

A straightforward expectation calculation (see (22) and (23) in Appendix B for the derivation) shows that for all k we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=0}^{\tau^k-1}\phi_J(x_t)\delta_J^k(t,w_J,w_M)\right] &= \Phi_J^\top Q r - \Phi_J^\top Q\left(I-\gamma P\right)\Phi_J w_J,\\
\mathbb{E}\left[\sum_{t=0}^{\tau^k-1}\phi_M(x_t)\delta_M^k(t,w_J,w_M)\right] &= \Phi_M^\top Q R\left(r+2\gamma P\Phi_J w_J\right) - \Phi_M^\top Q\left(I-\gamma^2 P\right)\Phi_M w_M.
\end{aligned}$$

Letting $\hat{w}_k = (\hat{w}_{J;k}, \hat{w}_{M;k})$ denote a concatenated weight vector in the joint space $\mathbb{R}^l \times \mathbb{R}^m$ we can write the TD algorithm in a stochastic approximation form as

$$\hat{w}_{k+1} = \hat{w}_k + \xi_k\left(z + M\hat{w}_k + \delta M_{k+1}\right), \tag{14}$$

where

$$M = \begin{pmatrix} \Phi_J^\top Q\left(\gamma P - I\right)\Phi_J & 0 \\ 2\gamma\Phi_M^\top QRP\Phi_J & \Phi_M^\top Q\left(\gamma^2 P - I\right)\Phi_M \end{pmatrix},$$

$$z = \begin{pmatrix} \Phi_J^\top Qr \\ \Phi_M^\top QRr \end{pmatrix},$$

and the noise terms δM_{k+1} satisfy

$$\mathbb{E}\left[\delta M_{k+1}|F_n\right] = 0,$$

where F_n is the filtration $F_n = \sigma(\hat{w}_m, \delta M_m, m \leq n)$, since different trajectories are independent.

We first claim that the eigenvalues of M have a negative real part. To see this, observe that M is block triangular, and its eigenvalues are just the eigenvalues of $M_1 \triangleq \Phi_J^\top Q\left(\gamma P - I\right)\Phi_J$ and $M_2 \triangleq \Phi_M^\top Q\left(\gamma^2 P - I\right)\Phi_M$. Lemma 6.10 of Bertsekas and Tsitsiklis (1996), shows that under Assumptions 1 and 4, the matrix $Q(\gamma P - I)$ is negative definite in the sense that $x^\top(\gamma P - I)x < 0 \quad \forall x \neq 0$ (Lemma 6.10 of Bertsekas and Tsitsiklis, 1996 is stated for the case $\gamma = 1$, but an extension to the simpler discounted case is trivial). By Assumption 3, this implies that the matrices M_1 and M_2 are negative definite in the sense that $x^\top M_1 x < 0 \quad \forall x \neq 0$, and $x^\top M_2 x < 0 \quad \forall x \neq 0$. Example 6.6 of Bertsekas, 2012 shows that the eigenvalues of a negative definite matrix have a negative real part. It therefore follows that the eigenvalues of M_1 and M_2 have a negative real part. Thus, the eigenvalues of M have a negative real part.

Next, let $h(w) = Mw + z$, and observe that the following conditions hold.

Condition 1 *The map h is Lipschitz.*

Condition 2 *The step sizes satisfy*

$$\sum_{k=0}^{\infty}\xi_k = \infty, \quad \sum_{k=0}^{\infty}\xi_k^2 < \infty.$$

Condition 3 *$\{\delta M_n\}$ is a martingale difference sequence, i.e., $\mathbb{E}\left[\delta M_{n+1}|F_n\right] = 0$.*

The next condition also holds

Condition 4 *The functions $h_c(w) \triangleq h(cw)/c, c \geq 1$ satisfy $h_c(w) \to h_\infty(w)$ as $c \to \infty$, uniformly on compacts, and $h_\infty(w)$ is continuous. Furthermore, the ODE*

$$\dot{w}(t) = h_\infty(w(t))$$

has the origin as its unique globally asymptotically stable equilibrium.

This is easily verified by noting that $h(cw)/c = Mw + c^{-1}z$, and since z is finite, $h_c(w)$ converges uniformly as $c \to \infty$ to $h_\infty(w) = Mw$. The stability of the origin is guaranteed since the eigenvalues of M have a negative real part (Khalil and Grizzle, 1996).

Theorem 7 in Chapter 3 of Borkar (2008) states that if Conditions 1-4 hold, the following condition holds

Condition 5 *The iterates of (14) remain bounded almost surely, i.e., $\sup_k \|\hat{w}_k\| < \infty$, a.s.*

Finally, we use a standard stochastic approximation result that, given that the above conditions hold, relates the convergence of the iterates of (14) with the asymptotic behavior of the ODE

$$\dot{w}(t) = h(w(t)). \tag{15}$$

Since the eigenvalues of M have a negative real part, (15) has a unique globally asymptotically stable equilibrium point (Khalil and Grizzle, 1996), which by (11) is exactly $\hat{w}* = (\hat{w}_J^*, \hat{w}_M^*)$. Formally, by Theorem 2 in Chapter 2 of Borkar (2008) we have that if Conditions 1, 2, 3 and 5 hold, then $\hat{w}_k \to \hat{w}*$ as $k \to \infty$ with probability 1. ■

It is interesting to note that despite the fact that the update of w_M depends on w_J, the algorithm converges using a single time scale, i.e., the same step-size schedule, for both w_J and w_M. This is in contrast with, for example, actor critic algorithms, that also have dependent updates but require multiple time-scales for convergence (Konda and Tsitsiklis, 2003). An intuitive reason for this is that the update for w_J is independent of w_M, therefore w_J will converge regardless, and w_M will 'track' it until convergence. Asymptotic convergence rates for TD(0) may also be derived along the lines of Konda (2002).

4.3 Multistep LSTD(λ) Algorithms

A common method in value function approximation (Bertsekas, 2012) is to replace the single-step mapping T_J with a multistep version, that takes into account multi-step transitions. For some $0 < \lambda < 1$, the multistep Bellman operator $T_J^{(\lambda)}$ is given by

$$T_J^{(\lambda)}(y) \triangleq (1-\lambda)\sum_{l=0}^{\infty}\lambda^l T_J^{l+1}(y) = (I - \lambda\gamma P)^{-1}r + \gamma P^{(\lambda,\gamma)}y,$$

where $P^{(\lambda,\gamma)} = (1-\lambda)\sum_{l=0}^{\infty}\lambda^l\gamma^l P^{l+1}$. The projected equation (10) then becomes

$$\Pi_J T_J^{(\lambda)}\left(\Phi_J w_J^{*(\lambda)}\right) = \Phi_J w_J^{*(\lambda)}.$$

Similarly, we may write a multistep equation for M

$$\Pi_M T_M^{(\lambda)}\left(\Phi_M w_M^{*(\lambda)}\right) = \Phi_M w_M^{*(\lambda)}, \tag{16}$$

where

$$T_M^{(\lambda)} \triangleq (1-\lambda)\sum_{l=0}^{\infty}\lambda^l T_{M^*}^{l+1},$$

and

$$T_{M^*}(y) \triangleq Rr + 2\gamma RP\Phi_J w_J^{*(\lambda)} + \gamma^2 Py.$$

Note the difference between T_{M^*} and $[T]_M$ defined earlier: we are no longer working on the joint space of J and M but instead we have an independent equation for approximating J,

and its solution $w_J^{*(\lambda)}$ is part of Equation (16) for approximating M. We can also write $T_M^{(\lambda)}$ explicitly as:

$$T_M^{(\lambda)}(y) = (I - \lambda\gamma^2 P)^{-1}\left(Rr + 2\gamma RP\Phi_J w_J^{*(\lambda)}\right) + \gamma^2 P^{(\lambda,\gamma^2)} y,$$

where $P^{(\lambda,\gamma^2)} = (1-\lambda)\sum_{l=0}^{\infty}\lambda^l\gamma^{2l}P^{l+1}$.

Proposition 7.1.1 of Bertsekas (2012) shows that for any $0 < \lambda < 1$ and $0 < \gamma \leq 1$ the projected operator $\Pi_J P^{(\lambda,\gamma)}$ is a contraction in the $\|\cdot\|_q$ norm. Therefore, both $\Pi_J T_J^{(\lambda)}$ and $\Pi_M T_M^{(\lambda)}$ are contractions with respect to the $\|\cdot\|_q$ norm, and both multistep projected equations have a unique solution. In a similar manner to the single step version, the projected equations may be written in matrix form

$$A^{(\lambda)} w_J^{*(\lambda)} = b^{(\lambda)}, \quad C^{(\lambda)} w_M^{*(\lambda)} = d^{(\lambda)}, \tag{17}$$

where

$$\begin{aligned}
A^{(\lambda)} &= \Phi_J^\top Q\left(I - \gamma P^{(\lambda,\gamma)}\right)\Phi_J, \quad b^{(\lambda)} = \Phi_J^\top Q(I - \lambda\gamma P)^{-1} r,\\
C^{(\lambda)} &= \Phi_M^\top Q\left(I - \gamma^2 P^{(\lambda,\gamma^2)}\right)\Phi_M,\\
d^{(\lambda)} &= \Phi_M^\top Q(I - \lambda\gamma^2 P)^{-1} R\left(r + 2\gamma P\Phi_J w_J^{*(\lambda)}\right).
\end{aligned}$$

Simulation based estimates $A_N^{(\lambda)}$ and $b_N^{(\lambda)}$ of the expressions above may be obtained by using eligibility traces, as described in Section 6.3.6 of Bertsekas (2012), and the LSTD(λ) approximation is then given by $\hat{w}_J^{*(\lambda)} = (A_N^{(\lambda)})^{-1} b_N^{(\lambda)}$. By substituting $w_J^{*(\lambda)}$ with $\hat{w}_J^{*(\lambda)}$ in the expression for $d^{(\lambda)}$, a similar procedure may be used to derive estimates $C_N^{(\lambda)}$ and $d_N^{(\lambda)}$, and to obtain the LSTD(λ) approximation $\hat{w}_M^{*(\lambda)} = (C_N^{(\lambda)})^{-1} d_N^{(\lambda)}$. A convergence result similar to Theorem 10 may also be obtained. Due to the similarity to the LSTD procedure in (13), the details are omitted. Finally, we note that a straightforward modification of the TD(0) algorithm to a multistep TD(λ) variant is also possible, using eligibility traces and following the procedure described in Section 6.3.6 of Bertsekas (2012).

4.4 A Direct Least Squares Regression Algorithm

We conclude this section with a simple regression style algorithm, which is not based on the TD approximation architecture of Section 3, but to our knowledge has not been proposed before.

As before, we let $x_0^k, x_1^k, \ldots, x_{\tau^k-1}^k$ denote the state sequence of the k'th simulated trajectory, and define the regression targets as

$$\hat{B}_t^k = \sum_{i=t}^{\tau^k-1} \gamma^{i-t} r(x_t^k).$$

Our approximation weights are now given by the solutions to the least squares problems

$$\hat{w}_J^* = \arg\min_{w_J} \sum_{k=1}^{N}\sum_{t=0}^{\tau^k-1}\left(\phi_J(x_t^k)^\top w_J - \hat{B}_t^k\right)^2,$$

and

$$\hat{w}_M^* = \arg\min_{w_M} \sum_{k=1}^{N}\sum_{t=0}^{\tau^k-1}\left(\phi_M(x_t^k)^\top w_M - \left(\hat{B}_t^k\right)^2\right)^2.$$

It may easily be verified that the approximate value $\tilde{J}$ and second moment $\tilde{M}$ of such a procedure converge, as $N \to \infty$, to the *direct* approximations $\Pi_J J$ and $\Pi_M M$, respectively. We further explore this algorithm and its relation to TD based algorithms in the empirical evaluation of Section 6.

5. Non Negative Approximate Variance

The TD algorithms of the preceding section approximate J and M by the solution to the fixed point equation (9). While Proposition 9 shows that the approximation errors of $\tilde{J}$ and $\tilde{M}$ are bounded, it does not guarantee that the approximated variance $\tilde{V}$, given by $\tilde{M} - \tilde{J}^2$, is non-negative for all states. A trivial remedy is to set all negative values of $\tilde{V}$ to zero; however, by such we lose information in these states. In this section we propose two alternative approaches to this problem. The first is through the choice of features, where we show that for the direct approximation $\Pi_J J$ and $\Pi_M M$, we can choose features that guarantee non-negative variance.

The second approach is based on the observation that non-negativeness of the variance may be written as a linear constraint in the weights for M. By adding such constraints to the projection in the fixed point equation (9), we obtain a different approximation architecture, in which non-negative variance is inherent. We show that this approximation scheme may be computed efficiently.

5.1 A Suitable Features Approach

For this section consider the direct approximation of J and M, as in Section 4.4, where we have $\tilde{J} = \Pi_J J$ and $\tilde{M} = \Pi_M M$. We investigate conditions under which $\tilde{M}(x) - \tilde{J}(x)^2 \geq 0$ for all $x \in X$.

Consider the following assumptions on the features:

Assumption 12 *The same features are used for J and M, i.e., $\Phi_J = \Phi_M$.*

Assumption 13 *The features are able to exactly represent a constant function, i.e., there exists w such that $\phi_J(x)^\top w = 1$ for all $x \in X$.*

We claim that Assumptions 12 and 13 suffice for guaranteeing non-negative approximate variance.

Proposition 14 *Let Assumptions 12 and 13 hold. Then $\tilde{M}(x) - \tilde{J}(x)^2 \geq 0$ for all $x \in X$.*

Proof First, by definition we have

$$V(x) = M(x) - J(x)^2 \geq 0. \tag{18}$$

Next, observe that Assumption 12 implies $\Pi_J = \Pi_M$.

Let $x \in X$, and recall that the projection operator is linear, thus we can write

$$\tilde{J}(x) = \sum_{i \in X} J(i)\omega_x(i), \quad \tilde{M}(x) = \sum_{i \in X} M(i)\omega_x(i), \tag{19}$$

where $\omega_x(i)$ are the projection weights for state x. Let $\bar{\omega}_x = \sum_{i \in X} \omega_x(i)$. We have

$$\tilde{J}(x)^2 = \bar{\omega}_x^2 \left(\sum_{i \in X} J(i) \frac{\omega_x(i)}{\bar{\omega}_x} \right)^2 \leq \bar{\omega}_x^2 \sum_{i \in X} J(i)^2 \frac{\omega_x(i)}{\bar{\omega}_x} \leq \bar{\omega}_x \sum_{i \in X} M(i)\omega_x(i) = \bar{\omega}_x \tilde{M}(x),$$

where the first inequality is by Jensen's inequality, the second inequality is by (18), and the equalities are by (19). Thus, $\bar{\omega}_x \leq 1$ guarantees $\tilde{V}(x) \geq 0$. We now claim that Assumption 13 guarantees $\bar{\omega}_x = 1$ for all x. To see this, consider a constant value function $J = 1$ for all states; clearly we have $\tilde{J} = 1$, as the weighted Euclidean error for this approximation is zero. Plugging in (19) gives $\sum_{i \in X} \omega_x(i) = 1$ for all x. ■

Proposition 14 concerns the approximation architecture itself, and not the estimation procedure. Therefore, it applies to the algorithms discussed above only asymptotically.

Many popular linear function approximation features such as grid tiles and CMAC's (Sutton and Barto, 1998) are able to represent a constant function. For these schemes, $\tilde{V}(x) \geq 0$ is guaranteed. For other schemes, we can guarantee $\tilde{V}(x) \geq 0$ by simply adding a constant feature to the feature set. Thus, at least for the direct approximation, it appears that a non-negative approximate variance is easily obtained. Whether a similar procedure may be applied to the fixed-point approximation is currently not known. However, Proposition 9 suggests that at least when the contraction modulus is small, the fixed-point approximation should behave similarly to the direct approximation. In the next section we propose a different approach, which *modifies* the fixed-point approximation to guarantee non-negative variance, regardless of the choice of features.

5.2 A Linearly Constrained Projection Approach

In this section we show that by adding linear constraints to the projected fixed point equation, we can guarantee a non-negative approximate variance. This modified approximation architecture admits a computationally efficient solution by a modification of the LSTD algorithm of Section 4.

First, let us write the equation for the second moment weights (10) with the projection operator as an explicit minimization

$$w_M^* = \arg\min_w \|\Phi_M w - \left(Rr + 2\gamma RP\Phi_J w_J^* + \gamma^2 P\Phi_M w_M^*\right)\|_q.$$

Observe that a non-negative approximate variance in some state x may be written as a *linear* inequality in w_M^* (but non-linear in w_J^*)

$$\phi_M(x)^\top w_M^* - (\phi_J(x)^\top w_J^*)^2 \geq 0.$$

We propose to add such inequality constraints to the projection operator. Let $\{x_1, \ldots, x_s\}$ denote a set of states in which we demand that the variance be non-negative. Let $H \in \mathbb{R}^{s \times m}$

denote a matrix with the features $-\phi_M^\top(x_i)$ as its rows, and let $g \in \mathbb{R}^s$ denote a vector with elements $-(\phi_J(x_i)^\top w_J^*)^2$. We write the non-negative-variance projected equation for the second moment as

$$w_M^+ = \begin{cases} \arg\min_w & \|\Phi_M w - \left(Rr + 2\gamma RP\Phi_J w_J^* + \gamma^2 P\Phi_M w_M^+\right)\|_q \\ \text{s.t.} & Hw \leq g \end{cases}. \tag{20}$$

Here, w_M^+ denotes the weights of $\tilde{M}$ in the *modified* approximation architecture. We now discuss whether a solution to (20) exists, and how it may be obtained.

Let us assume that the constraints in (20) admit a feasible solution:

Assumption 15 *There exists w such that $Hw < g$.*

Note that a trivial way to satisfy Assumption 15 is to have some feature vector that is positive for all states. To see this, let i^+ denote the index of the positive feature vector, and choose w to be all zeros, except for the i^+ element, which should satisfy $w_{i^+} < -\left(\max_{1 \leq i \leq s} |g_i|\right) / \left(\max_{1 \leq i \leq s} H_{i,i^+}\right)$.

Equation (20) is a form of projected equation studied by Bertsekas (2011), the solution of which exists, and may be obtained by the following iterative procedure

$$w_{k+1} = \Pi_{\Xi, \hat{W}_M}[w_k - \eta \Xi^{-1}(Cw_k - d)], \tag{21}$$

where C and d are defined in (12), Ξ is an arbitrary positive definite symmetric matrix, $\eta \in \mathbb{R}$ is a positive step size, and $\Pi_{\Xi, \hat{W}_M}$ denotes a projection onto the convex set $\hat{W}_M = \{w | Hw \leq g\}$ with respect to the Ξ weighted Euclidean norm.

The following lemma, which is based on a convergence result of Bertsekas (2011), guarantees that for $\gamma < 1$, the iteration (21) converges. For the non-discounted setting a similar result may be obtained by using the multi-step approach with $\lambda > 0$, as detailed in Tamar et al. (2013).

Lemma 16 *Assume $\gamma < 1$, and let Assumptions 1, 3, 4, and 15 hold. Then (20) admits a unique solution w_M^+, and there exists $\bar{\eta} > 0$ such that $\forall \eta \in (0, \bar{\eta})$ and $\forall w_0 \in \mathbb{R}^m$ the iteration (21) converges at a linear rate to w_M^+ (i.e., $\|w_k - w_M^+\|$ converges to 0 at least as fast as a geometric progression).*

Proof Bertsekas (2011) shows that projected fixed-point equations of the form

$$w^* = \begin{cases} \arg\min_w & \|\Phi w - T_{\text{lin}}(\Phi w^*)\|_q \\ \text{s.t.} & w \in \Omega \end{cases},$$

where $T_{\text{lin}}(y) = A_{\text{lin}} y + b_{\text{lin}}$ is a contracting linear operator, and Ω is a polyhedral set, may be solved iteratively by

$$w_{k+1} = \Pi_{\Xi, \Omega}[w_k - \eta \Xi^{-1}(C_{\text{lin}} w_k - d_{\text{lin}})],$$

where $\Pi_{\Xi, \Omega}$ projects onto Ω w.r.t. the norm $\|y\|_\Xi = \sqrt{y^\top \Xi y}$ for an arbitrary symmetric and positive-definite matrix Ξ, $C_{\text{lin}} = \Phi^\top Q(I - A_{\text{lin}})\Phi$ and $d_{\text{lin}} = \Phi' Q b_{\text{lin}}$. Specifically, the convergence result of Bertsekas (2011) shows that when T_{lin} is a contraction in the $\|\cdot\|_q$

norm, Ω is polyhedral, and Φ is full rank, there exists $\bar{\eta} > 0$ such that for all $\eta \in (0, \bar{\eta})$, and for all $w_0 \in \mathbb{R}^m$, the preceding iteration converges at a linear rate to the unique solution of the projected fixed point equation described above.

Substituting $T_{\text{lin}}(y)$ with $T_M(y) = Rr + 2\gamma RP\Phi_J w_J^* + \gamma^2 Py$, and Ω with the set defined by $Hw \leq g$, we obtain the projected fixed point equation (20), and the corresponding iteration (21). To apply the convergence result, the full-rank of Φ_M is guaranteed by Assumption 3, and the contraction of T_M in the $\|\cdot\|_q$ norm is guaranteed by Lemma 5, since P is a non-expansion and $\gamma < 1$. ■

Generally, C, d, and w_J^* are not known in advance, and should be replaced in (21) with their simulation based estimates, C_N, d_N, and $\hat{w}_J^*$, proposed in the previous section. The convergence of these estimates, together with the result of Lemma 16, lead to the following result; the proof is detailed in Appendix C.

Theorem 17 *Consider the algorithm in* (21) *with C, d, and w_J^* replaced by C_N, d_N, and $\hat{w}_J^*$, respectively, and with $k(N)$ replacing k for a specific N. Also, let the assumptions in Lemma 16 hold, and let $\eta \in (0, \bar{\eta})$, with $\bar{\eta}$ defined in Lemma 16. Then $w_{k(N)} \to w_M^+$ as $N \to \infty$ and $k(N) \to \infty$ almost surely. Namely, for any $\bar{\epsilon} > 0$ w.p. 1 there is a $\bar{N}(\bar{\epsilon})$ such that for any $N > \bar{N}(\bar{\epsilon})$ there is a $k(N, \bar{\epsilon})$, such that for all $k > k(N, \bar{\epsilon})$ we have $\|w_{k;N} - w_M^+\| \leq \bar{\epsilon}$.*

We remark that we do not know how to quantify how the linear constraints affect the approximation error. While intuitively our constraints add prior information that is 'correct' in some sense (since we know that the true variance is positive), it is not hard to construct examples where the constraints actually increase the error. In the following, we provide an illustration of the linearly constrained projection approach on a toy problem. We qualitatively show that the method effectively produces a non-negative solution, without significantly affecting the approximation error.

Consider the Markov chain depicted in Figure 1, which consists of n states with reward -1 and a terminal state x^* with zero reward. Assume no discounting, i.e., $\gamma = 1$. The transitions from each state is either to a subsequent state (with probability p) or to a preceding state (with probability $1-p$), with the exception of the first state which transitions to itself instead. We chose to approximate J and M with polynomials of degree 1 and 2, respectively, i.e., $\Phi_J(x) = [1, x]^\top$ and $\Phi_M(x) = [1, x, x^2]^\top$. For such a small problem, the fixed point equation (17) may be solved exactly, yielding the approximation depicted in Figure 2 (dotted line), for $p = 0.7$, $N = 30$, and $\lambda = 0.95$. Note that the variance, in Figure 2C, is negative for the last two states. Using algorithm (21) we obtained a positive variance constrained approximation, which is depicted in Figure 2 (dashed line). Note how the approximate variance has been adjusted to be positive for all states.

6. Experiments

In this section we present numerical simulations of policy evaluation for an option pricing domain. We show that in terms of sample efficiency, our LSTD(λ) algorithm significantly outperforms the current state-of-the-art. We begin by describing the domain and its modeling as an MDP, and then present our policy evaluation results. We emphasize that our

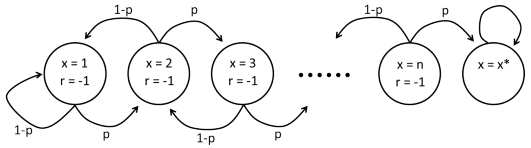


Figure 1: An example Markov chain.

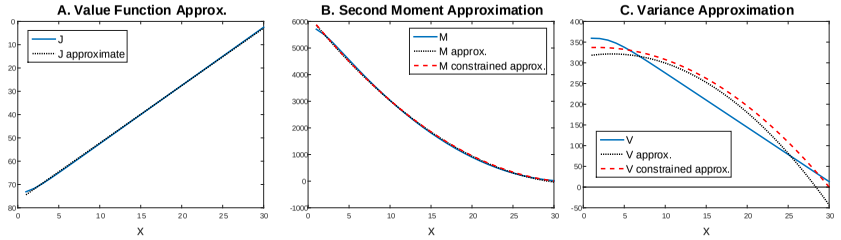


Figure 2: Value, second moment and variance approximation.

results only concern policy evaluation, and not policy optimization. The following MDP description is given for the purpose of presentation completeness.

6.1 Option Pricing

An American-style put option (Hull, 2006) is a contract which gives the owner the right, but not the obligation, to sell an asset at a specified strike price K on or before some maturity time t^*. Letting x_t denote the price (state) of the asset at time $t \leq t^*$, the immediate payoff of executing the option at that time is therefore $\max(0, K - x_t)$. Assuming Markov state transitions, an optimal execution policy may be found by solving a finite horizon MDP, and the expected profit under that policy is termed the 'fair' price of the option. Since the state space is typically continuous, an exact solution is infeasible, calling for approximate, sampling based techniques (Longstaff and Schwartz, 2001; Tsitsiklis and Van Roy, 2001; Li et al., 2009).

The option pricing problem may be formulated as an MDP as follows. To account for the finite horizon, we include time explicitly in the state, thus, the state at time t is $\{x_t; t\}$. The action set is binary, where 1 stands for executing the option and 0 for continuing to hold it. Once an option is executed, or when $t = t^*$, a transition to a terminal state takes place. Otherwise, the state transitions to $\{x_{t+1}; t+1\}$ where x_{t+1} is determined by a stochastic kernel $P(x_{t+1}|x_t, t)$. In our experiments we used a Bernoulli price fluctuation model (Cox

et al., 1979),

$$x_{t+1} = \begin{cases} f_u x_t, & \text{w.p. } p \\ f_d x_t, & \text{w.p. } 1-p \end{cases},$$

where the up and down factors, f_u and f_d, are constant. The reward for executing $u = 1$ at state x is $r(x) \triangleq \max(0, K - x)$ and zero otherwise. Note that by definition, for any state x in which the policy decides to execute, the reward-to-go is deterministic and equal to $r(x)$. Thus, we only need to estimate J and V for states in which the policy decides to hold. We focus on 'in-the-money' options, in which K is equal to the initial price x_0, and set $T = 20$.

A policy π was obtained using the LSPI algorithm (Lagoudakis and Parr, 2003; Li et al., 2009) with 2-dimensional (for x and t) radial basis function (RBF) features, as detailed in Tamar et al. (2014). It is well-known (Duffie, 2010), and intuitive, that the optimal policy (in terms of expected return) for the put option has a threshold structure—the policy executes if the price is below some boundary $\bar{x}_t$, and holds otherwise. It is also known, that $\bar{x}_t$ is monotonically increasing in t. Our policy π has such a structure as well. We emphasize, however, that the specific method of generating the policy π is not the focus of this work, as we are only interested in *evaluating* π. Thus, any policy generation method could have been used, and LSPI was chosen for convenience. In the following, we evaluate the value functions J and V for π.

6.2 Results

We now present our policy evaluation results for the put option domain. MATLAB® code for reproducing these results is available on the web-site `https://sites.google.com/site/variancetdcode/`.

We first calculate the 'true' value function J and standard deviation of reward-to-go $\sqrt{V}$, as shown in Figure 3. These plots were obtained using Monte Carlo (MC), by taking the empirical average and standard deviation of the reward of 10,000 trajectories starting from 323 equally spaced points in the state space for which the policy π decides to hold, a total of $N = 3,230,000$ trajectories. To our knowledge, an MC approach is the current state-of-the-art for obtaining an estimate of V.

Note the exercise boundary $\bar{x}_t$, emphasized with a dashed line in the value function plot. For x smaller than $\bar{x}_t$, the policy decides to exercise, therefore the value is linear in x and the variance is zero. Also note the discontinuous ridges on the J and $\sqrt{V}$ plots. These ridges are due to the discrete transition model, and occur when a transition to the next state (or the state following the next state) crosses the exercise boundary. To the risk-sensitive decision maker, these ridges are important, as they separate states with roughly the same expected return but with very different variance.

In Figure 4 we show the RMS error of the approximation $\sqrt{\tilde{V}}$ (compared to the 'true' $\sqrt{V}$) computed using the LSTD(0) algorithm of Section 4, for different budgets of sample trajectories N. We tested two popular feature sets: RBF features with 77 equally spaced centers, and tile features with 600 uniform non-overlapping tiles. In both cases the same features were used for both J and M. The sample trajectories were simulated independently, starting from uniformly distributed initial states. We compare our results to MC estimates obtained with the same trajectories.

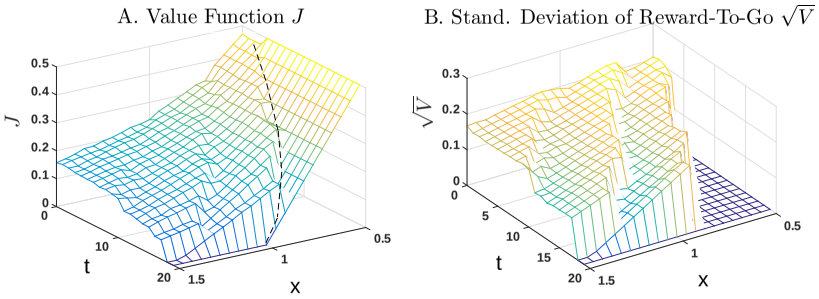


Figure 3: True value function J and standard deviation of the reward-to-go $\sqrt{V}$.

As can be seen, by exploiting relations *between* states and using the generalization capabilities of the function approximation, LSTD is able to fully exploit the data, and performs significantly better than MC for relatively small sample sizes. On the other hand, LSTD is limited by the expressiveness of its function approximation, and its error is therefore bounded.

Note that for $N \leq 323$ the MC estimate is meaningless, as the empirical standard deviation cannot be calculated from only one sample. LSTD however, is able to provide a reasonable result. Also note that the LSTD estimate is defined over the whole state-space, whereas the MC estimate is only defined for the discrete set of evaluation points.

To further appreciate the advantage of function approximation, we provide a visual comparison of the approximated standard deviation of reward-to-go $\sqrt{\tilde{V}}$. In Figure 5 we plot $\sqrt{\tilde{V}}$ obtained using a budget of $N = 2000$ sample trajectories starting from uniformly distributed states. In the left plot, we show the results of LSTD(λ) with RBF features (with 77 equally spaced centers in x and t). The variance in states where the policy decides to execute was set to zero manually, as there is no need to estimate it. In comparison, on the right plot we present the results of a Monte Carlo algorithm, with the same amount of data trajectories $N = 2000$. Clearly, LSTD(λ) makes better use of the limited data, with a plot that is much more similar to the true standard deviation (Figure 3; right). More importantly, the relevant structure in $\sqrt{V}$ outlined above is clear in the LSTD(λ) result (up to a smoothness limitation of the RBFs), yielding important information for the decision maker.

In Figure 6 we consider the LSTD(λ) algorithm with the tile features discussed above, and explore the effect of λ on the RMS error in $\sqrt{\tilde{V}}$. As in regular LSTD, λ can be seen to trade off estimation bias and variance (Bertsekas, 2012). In addition, we compare LSTD(λ) to the direct least squares algorithm of Section 4.4. For the case of the value function J, it is well-known (Bertsekas, 2012) that the direct approximation is equivalent to the limit $\lambda \to 1$. Our results suggest that a similar relation holds for the variance V as well. Furthermore, these results highlight the superior performance of the TD approach in the small sample size regime.

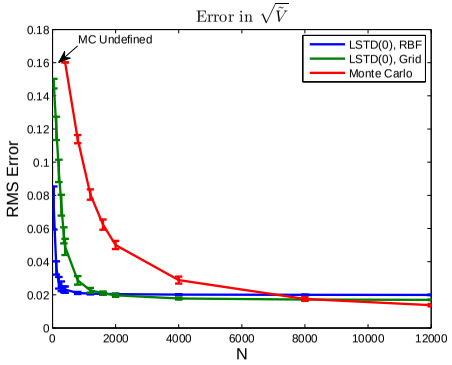


Figure 4: LSTD(0) vs. Monte Carlo. The RMS error of $\sqrt{\tilde{V}}$ on a set of evaluation points (see text) is shown vs. the budget of sample trajectories N, for LSTD(0) with two types of features and for Monte Carlo. Standard deviation error-bars from 20 runs are shown.

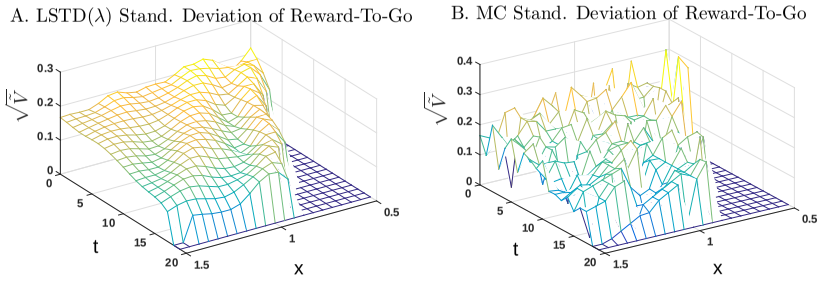


Figure 5: Approximate standard deviation of the reward-to-go $\sqrt{\tilde{V}}$. Left plot was obtained by LSTD(λ) with RBF features, using 2000 trajectories and $\lambda = 0.3$. Right plot was obtained using Monte Carlo, also with 2000 trajectories.

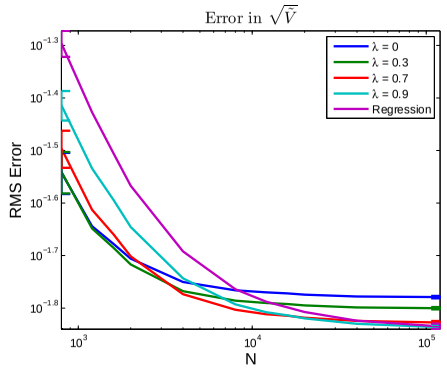


Figure 6: LSTD(λ) Results. The RMS error of $\sqrt{\tilde{V}}$ on a set of evaluation points (see text) is shown vs. the budget of sample trajectories N for different λ, and also for the least squares regression algorithm. For clarity, a log scale is used, and the error-bars (standard deviation from 20 runs) are shown only for marginal points.

7. Conclusion

We presented an extension of the TD framework for policy evaluation in MDPs with respect to the variance of the reward-to-go. Our framework deals with the curse of dimensionality by using function approximation, and uses a bootstrapping technique, based on an extension of the Bellman equation to the second moment, to achieve good performance even for a small sample size. We presented both formal guarantees and empirical evidence that this approach is useful in problems with a large state space, and limited sample budget.

A natural extension of this work is to consider higher moments, and statistical properties such as skewness and kurtosis of the reward-to-go. An extension of Bellman's equation to higher moments was proposed by Sobel (1982), and it may be used to derive TD equations similarly to the work presented here. This may also be useful for optimizing the expectation of a general function f of the accumulated reward $\mathbb{E}[f(B)]$, by looking at the first few terms in the Taylor expansion of f. It would be interesting to see whether a TD approach may be developed for other risk measures such as the value at risk or semi-deviation.

Another interesting direction is to use the variance of the reward-to-go to *guide feature selection*, or feature modification. For example, consider tile features. A large variance-to-go for states that belong to a particular tile may indicate that the value function in that tile varies greatly, and therefore it may be beneficial to split the tile into smaller segments. Of course, another explanation for the variance may be the inherent stochasticity of the system. Thus, a thoughtful feature-selection method should take that also into account. In a related topic, the variance of the reward-to-go may also be used to *guide exploration*, since intuitively, states with higher variance should be allocated more exploration resources, to potentially decrease the variance, if possible.

We conclude with a discussion on policy optimization with respect to a mean-variance tradeoff. While a naive variance-penalized policy iteration algorithm may be easily conceived, its usefulness should be questioned, as it was shown to be problematic for the standard deviation adjusted reward (Sobel, 1982) and the variance constrained reward (Mannor and Tsitsiklis, 2013). An alternative approach is to pursue *locally* optimal policies by using a gradient based method. Tamar et al. (2012) proposed policy gradient algorithms for a class of variance related criteria, and showed their convergence to local optima. These algorithms may be extended to use the variance function in an actor-critic type scheme (Sato et al., 2001), and recent work has extended these ideas to large-scale MDPs by employing function approximation, and using the TD policy evaluation algorithms presented here (Tamar and Mannor, 2013; Prashanth and Ghavamzadeh, 2013).

Acknowledgments

We would like to thank our anonymous reviewers for helpful comments and suggestions. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Program (FP/2007-2013) / ERC Grant Agreement n. 306638. Aviv Tamar is also partially funded by the Viterbi Scholarship, Technion.

Appendix A. Proof of Proposition 2

Proof The equation for $J(x)$ is well-known, and its proof is given here only for completeness. Choose $x \in X$. Then,

$$\begin{aligned}
J(x) &= \mathbb{E}\left[B|x_0 = x\right] \\
&= \mathbb{E}\left[\sum_{k=0}^{\tau-1} \gamma^k r(x_k) \middle| x_0 = x\right] \\
&= r(x) + \mathbb{E}\left[\sum_{k=1}^{\tau-1} \gamma^k r(x_k) \middle| x_0 = x\right] \\
&= r(x) + \gamma \mathbb{E}\left[\mathbb{E}\left[\sum_{k=1}^{\tau-1} \gamma^{k-1} r(x_k) \middle| x_0 = x, x_1 = x'\right]\right] \\
&= r(x) + \gamma \sum_{x' \in X} P(x'|x) J(x'),
\end{aligned}$$

where we excluded the terminal state from the last sum since reaching it ends the trajectory.

Similarly,

$$\begin{aligned}
M(x) &= \mathbb{E}\left[B^2|x_0 = x\right] \\
&= \mathbb{E}\left[\left(\sum_{k=0}^{\tau-1} \gamma^k r(x_k)\right)^2 \middle| x_0 = x\right] \\
&= \mathbb{E}\left[\left(r(x_0) + \sum_{k=1}^{\tau-1} \gamma^k r(x_k)\right)^2 \middle| x_0 = x\right] \\
&= r(x)^2 + 2r(x)\mathbb{E}\left[\sum_{k=1}^{\tau-1} \gamma^k r(x_k) \middle| x_0 = x\right] + \mathbb{E}\left[\left(\sum_{k=1}^{\tau-1} \gamma^k r(x_k)\right)^2 \middle| x_0 = x\right] \\
&= r(x)^2 + 2\gamma r(x) \sum_{x' \in X} P(x'|x) J(x') + \gamma^2 \sum_{x' \in X} P(x'|x) M(x').
\end{aligned}$$

The uniqueness of the value function J for a proper policy is well known, cf. Proposition 3.2.1 in Bertsekas (2012). The uniqueness of M follows by observing that in the equation for M, M may be seen as the value function of an MDP with the same transitions but with reward $r(x)^2 + 2\gamma r(x) \sum_{x' \in X} P(x'|x) J(x')$. Since only the rewards change, the policy remains proper and Proposition 3.2.1 in Bertsekas (2012) applies. ■

Appendix B. Proof of Theorem 10

Proof Let $\phi_1(x)$, $\phi_2(x)$ be some vector functions of the state. We claim that

$$\mathbb{E}\left[\sum_{t=0}^{\tau-1} \phi_1(x_t)\phi_2(x_t)^\top\right] = \sum_x q(x)\phi_1(x)\phi_2(x)^\top \equiv \Phi_1^\top Q \Phi_2, \tag{22}$$

where Φ_1 and Φ_2 are matrices with rows $\phi_1(x)$ and $\phi_2(x)$, respectively. To see this, let $\mathbb{1}(\cdot)$ denote the indicator function and write

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=0}^{\tau-1} \phi_1(x_t)\phi_2(x_t)^\top\right] &= \mathbb{E}\left[\sum_{t=0}^{\tau-1} \sum_x \phi_1(x)\phi_2(x)^\top \mathbb{1}(x_t = x)\right] \\
&= \mathbb{E}\left[\sum_x \phi_1(x)\phi_2(x)^\top \sum_{t=0}^{\tau-1} \mathbb{1}(x_t = x)\right] \\
&= \sum_x \phi_1(x)\phi_2(x)^\top \mathbb{E}\left[\sum_{t=0}^{\tau-1} \mathbb{1}(x_t = x)\right].
\end{aligned}$$

Now, note that the last term on the right hand side is an expectation (over all possible trajectories) of the number of visits to a state x until reaching the terminal state, which is

exactly $q(x)$ since

$$\begin{aligned} q(x) &= \sum_{t=0}^{\infty} P(x_t = x) \\ &= \sum_{t=0}^{\infty} \mathbb{E}[\mathbb{1}(x_t = x)] \\ &= \mathbb{E}\left[\sum_{t=0}^{\infty} \mathbb{1}(x_t = x)\right] \\ &= \mathbb{E}\left[\sum_{t=0}^{\tau-1} \mathbb{1}(x_t = x)\right], \end{aligned}$$

where the third equality is by the dominated convergence theorem (Grimmett and Stirzaker, 2001, Sec. 5.6), and last equality follows from the absorbing property of the terminal state. Similarly, we have

$$\mathbb{E}\left[\sum_{t=0}^{\tau-1} \phi_1(x_t)\phi_2(x_{t+1})^\top\right] = \sum_x \sum_{x'} q(x)P(x'|x)\phi_1(x)\phi_2(x')^\top \equiv \Phi_1^\top Q P \Phi_2, \qquad (23)$$

since

$$\begin{aligned} \mathbb{E}\left[\sum_{t=0}^{\tau-1} \phi_1(x_t)\phi_2(x_{t+1})^\top\right] &= \mathbb{E}\left[\sum_{t=0}^{\tau-1} \sum_x \sum_{x'} \phi_1(x)\phi_2(x')^\top \mathbb{1}(x_t = x, x_{t+1} = x')\right] \\ &= \mathbb{E}\left[\sum_x \sum_{x'} \phi_1(x)\phi_2(x')^\top \sum_{t=0}^{\tau-1} \mathbb{1}(x_t = x, x_{t+1} = x')\right] \\ &= \sum_x \sum_{x'} \phi_1(x)\phi_2(x')^\top \mathbb{E}\left[\sum_{t=0}^{\tau-1} \mathbb{1}(x_t = x, x_{t+1} = x')\right], \end{aligned}$$

and

$$\begin{aligned} q(x)P(x'|x) &= \sum_{t=0}^{\infty} P(x_t = x)P(x'|x) \\ &= \sum_{t=0}^{\infty} P(x_t = x, x_{t+1} = x') \\ &= \sum_{t=0}^{\infty} \mathbb{E}[\mathbb{1}(x_t = x, x_{t+1} = x')] \\ &= \mathbb{E}\left[\sum_{t=0}^{\infty} \mathbb{1}(x_t = x, x_{t+1} = x')\right] \\ &= \mathbb{E}\left[\sum_{t=0}^{\tau-1} \mathbb{1}(x_t = x, x_{t+1} = x')\right]. \end{aligned}$$

Since trajectories between visits to the recurrent state are statistically independent, the law of large numbers together with the expressions in (22) and (23) suggest that the approximate expressions in (13) converge to their expected values with probability 1, therefore we have

$$\begin{aligned} A_N &\to A, \quad b_N \to b, \\ C_N &\to C, \quad d_N \to D, \end{aligned}$$

and

$$\begin{aligned} \hat{w}^*_{J;N} &= A_N^{-1} b_N \to A^{-1} b = w^*_J, \\ \hat{w}^*_{M;N} &= C_N^{-1} d_N \to C^{-1} d = w^*_M. \end{aligned}$$

■

Appendix C. Proof of Theorem 17

To show the convergence of the simulation-based version of (21) to a solution of (20), we need to bound the effect of simulation noise on the fixed point of (21). The difficulty, is that simulation noise affects both the terms in the update, C and d, and terms in the projection step—the set $\hat{W}_M$ onto which we project. In addition, the noise in C and d effectively adds noise to the weights q of the norm in (20), which should also be bounded.

We begin by proving several intermediate results. The first concerns the continuity of fixed points of contraction operators.

Lemma 18 *Let T_1 be a γ-contraction in the q_1 norm, and T_2 be a γ-contraction in the q_2 norm. Assume that there exists some δ' such that*

$$\|T_1 x - T_2 x\|_{q_1} \le \delta + \delta' \|x\|_{q_1}, \quad \forall x.$$

Let x_1^* and x_2^* denote the fixed points of T_1 and T_2, respectively. Then the following holds:

$$\|x_2^* - x_1^*\|_{q_1} \le \frac{\delta + \delta' \|x_2^*\|_{q_1}}{1 - \gamma}.$$

Proof We have

$$\begin{aligned} \|x_2^* - x_1^*\|_{q_1} &= \|T_2 x_2^* - x_1^*\|_{q_1} \\ &= \|T_2 x_2^* + T_1 x_2^* - T_1 x_2^* - x_1^*\|_{q_1} \\ &\le \|T_1 x_2^* - x_1^*\|_{q_1} + \|T_2 x_2^* - T_1 x_2^*\|_{q_1} \\ &\le \|T_1 x_2^* - T_1 x_1^*\|_{q_1} + \delta + \delta' \|x_2^*\|_{q_1} \\ &\le \gamma \|x_2^* - x_1^*\|_{q_1} + \delta + \delta' \|x_2^*\|_{q_1}. \end{aligned}$$

Rearranging, gives:

$$\|x_2^* - x_1^*\|_{q_1} \le \frac{\delta + \delta' \|x_2^*\|_{q_1}}{1 - \gamma}.$$

■

The following results concerns the sensitivity of weighted Euclidean-norm projections.

Lemma 19 *Let $\|\cdot\|_q$ and $\|\cdot\|_{q'}$ denote weighted Euclidean-norms on $\mathbb{R}^n$ with weights $q > 0$ and $q' > 0$, respectively. Let Π and Π' denote projections onto a closed and convex set $S \subset \mathbb{R}^n$, w.r.t. the norms $\|\cdot\|_q$ and $\|\cdot\|_{q'}$, respectively. For any $x \in \mathbb{R}^n$ we have:*

$$\|\Pi x - \Pi' x\|_q^2 \leq 2\|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right).$$

Proof If $x \in S$ the result is trivial. We assume in the following $x \notin S$. Let $Q = diag(q)$ and $Q' = diag(q')$. For any $x, y \in \mathbb{R}^n$ we have

$$\begin{aligned}
\left|\|x-y\|_q^2 - \|x-y\|_{q'}^2\right| &= \left|(x-y)^\top Q(x-y) - (x-y)^\top Q'(x-y)\right| \\
&= \left|(x-y)^\top (Q-Q')(x-y)\right| \\
&\leq \sum_{i=1}^n |q_i - q_i'|(x_i - y_i)^2 \\
&\leq \|q-q'\|_\infty \|x-y\|_2^2.
\end{aligned} \tag{24}$$

Therefore, we have that

$$\|\Pi' x - x\|_{q'}^2 \geq \|\Pi' x - x\|_q^2 - \|q-q'\|_\infty \|\Pi' x - x\|_2^2. \tag{25}$$

Now, let H denote the hyper-plane that is orthogonal to the projection error $\Pi x - x$, and passes through Πx:

$$H \doteq \left\{y \in \mathbb{R}^n : (y - \Pi x)^\top Q(x - \Pi x) = 0\right\},$$

and let L denote a line that passes through x and $\Pi' x$:

$$L \doteq \left\{y \in \mathbb{R}^n : y = x + z(\Pi' x - x), \quad z \in \mathbb{R}\right\}.$$

By properties of the projection Πx (Hiriart-Urruty and Lemaréchal, 2013) we have $(y - \Pi x)^\top Q(x - \Pi x) \leq 0 \quad \forall y \in S$. Since $(x - \Pi x)^\top Q(x - \Pi x) > 0$, it follows that H separates x from S. Since $\Pi' x \in S$, H also separates x from $\Pi' x$. Let p^* denote the intersection of L and H. By the previous arguments, p^* exists, and

$$\Pi' x - x = \alpha(p^* - x), \tag{26}$$

with $\alpha \geq 1$. Now, we have

$$\begin{aligned}
\|\Pi' x - x\|_q^2 &= \|\alpha(p^* - x)\|_q^2 \\
&= \alpha^2 \|p^* - x\|_q^2 \\
&= \alpha^2 \|p^* - \Pi x\|_q^2 + \alpha^2 \|\Pi x - x\|_q^2 \\
&\geq \alpha^2 \|p^* - \Pi x\|_q^2 + \|\Pi x - x\|_{q'}^2 - \|q - q'\|_\infty \|\Pi x - x\|_2^2,
\end{aligned}$$

where the last equality is by the Pythagorean theorem, which holds due to the orthogonality of H to the error $\Pi x - x$, and the inequality is since $\alpha \geq 1$, and (24). Plugging in (25), we obtain:

$$\|\Pi' x - x\|_{q'}^2 - \|\Pi x - x\|_{q'}^2 \geq \alpha^2 \|p^* - \Pi x\|_q^2 - \|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right). \tag{27}$$

However, by definition of the projection $\Pi' x$, we must have $\|\Pi' x - x\|_{q'}^2 - \|\Pi x - x\|_{q'}^2 \leq 0$, therefore rearranging (27) leads to:

$$\alpha^2 \|p^* - \Pi x\|_q^2 \leq \|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right). \tag{28}$$

Now, let H' denote a parallel hyper-plane to H that passes through $\Pi' x$:

$$H' \doteq \left\{y \in \mathbb{R}^n : (y - \Pi' x)^\top Q(x - \Pi x) = 0\right\}.$$

Also, let L' denote the line between x and Πx:

$$L' \doteq \left\{y \in \mathbb{R}^n : y = x + z(\Pi x - x), \quad z \in \mathbb{R}\right\}.$$

By definition, H' is orthogonal to L'; denote by p^{**} their intersection. By triangle similarity (the triangles $\{x, \Pi x, p^*\}$ and $\{x, p^{**}, \Pi' x\}$), and (26) we have

$$\frac{\|\Pi' x - p^{**}\|_q^2}{\|\Pi x - p^*\|_q^2} = \frac{\|\Pi' x - x\|_q^2}{\|p^* - x\|_q^2} = \alpha^2, \tag{29}$$

therefore, using (28)

$$\|\Pi' x - p^{**}\|_q^2 = \alpha^2 \|p^* - \Pi x\|_q^2 \leq \|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right). \tag{30}$$

From the Pythagorean theorem (by the orthogonality of H' to L') we have:

$$\|\Pi' x - \Pi x\|_q^2 = \|\Pi x - p^{**}\|_q^2 + \|\Pi' x - p^{**}\|_q^2, \tag{31}$$

and

$$\|\Pi' x - x\|_q^2 = \|\Pi' x - p^{**}\|_q^2 + \|p^{**} - x\|_q^2,$$

and from the last equation we also have

$$\|\Pi' x - x\|_q^2 \geq \|p^{**} - x\|_q^2.$$

Now, from the last inequality:

$$\begin{aligned}
\|\Pi' x - x\|_q^2 &\geq \|p^{**} - x\|_q^2 \\
&\geq \|p^{**} - \Pi x\|_q^2 + \|\Pi x - x\|_q^2 \\
&\geq \|p^{**} - \Pi x\|_q^2 + \|\Pi x - x\|_{q'}^2 - \|q - q'\|_\infty \|\Pi x - x\|_2^2,
\end{aligned}$$

where the second inequality is since x, Πx, and p^{**} are on L', therefore $\|p^{**} - x\|_q = \|p^{**} - \Pi x\|_q + \|\Pi x - x\|_q$, and the last inequality is by (24). Proceeding similarly as in (27), we plug in (25) to obtain:

$$\|\Pi' x - x\|_{q'}^2 - \|\Pi x - x\|_{q'}^2 \geq \|p^{**} - \Pi x\|_q^2 - \|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right), \tag{32}$$

and similarly to (28), by definition of the projection $\Pi' x$, we must have $\|\Pi' x - x\|_{q'}^2 - \|\Pi x - x\|_{q'}^2 \leq 0$, therefore rearranging (32) leads to:

$$\|p^{**} - \Pi x\|_q^2 \leq \|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right). \tag{33}$$

Finally, plugging in (30), and (33) in (31) we obtain

$$\|\Pi' x - \Pi x\|_q^2 \leq 2\|q - q'\|_\infty \left(\|\Pi x - x\|_2^2 + \|\Pi' x - x\|_2^2\right).$$

■

We now proceed with the proof of Theorem 17. To simplify the presentation, we break the proof into several parts.
In part 1, we show show that the sampled version of algorithm (21) with N samples corresponds to solving (20) with P_N, a sampled version of the transition matrix, replacing P.
In part 2, we show that for each N, algorithm (21) would converge (in k) by Lemma 16 to a fixed point of the *sampled* projected equation.
In part 3, we show that the solution of the sampled projected equation converges (in N) to the the solution of the original projected equation. We do this by showing a continuity of the solution w.r.t. P and its derived quantities, q and w_J^*, from which convergence then follows by the law of large numbers.
In part 4, we collect our convergence results in k and N and complete the proof.

C.1 A Sampled Version of Eq. (21)

Let $S^+ = \{\Phi_M w | w \in \mathbb{R}^m, Hw \leq g\}$ denote the set onto which we project in the modified projection (20), and let Π_q^+ denote a projection onto S^+ w.r.t. the q-weighted Euclidean norm. Note that S^+ is a convex set, therefore Π_q^+ is a non-expansion in the $\|\cdot\|_q$ norm (Hiriart-Urruty and Lemaréchal, 2013). Furthermore, we can write Eq. (20) as follows:

$$w_M^+ = \Pi_q^+ T^+ w_M^+,$$

where $T^+(w) = Rr + 2\gamma RP\Phi_J w_J^* + \gamma^2 P\Phi_M w$.

After we have observed N trajectories, let P_N denote the corresponding empirical transition matrix, given by:

$$P_N(x'|x) = \left(\frac{1}{\sum_{k=1}^N \tau_k}\right) \sum_{k=1}^N \sum_{t=0}^{\tau_k - 1} \mathbb{1}(x_t^k = x, x_{t+1}^k = x'),$$

and let $\zeta_{0;N}$ denote the empirical initial state distribution, i.e.,

$$\zeta_{0;N}(x) = \left(\frac{1}{N}\right) \sum_{k=1}^N \mathbb{1}(x_0^k = x).$$

Also, let q_N denote the state occupancy probabilities in an MDP with P and ζ_0 replaced by P_N and $\zeta_{0;N}$ (cf. the definition of q in Section 3). For large enough N, q_N satisfies Assumption 4.

Let $\hat{w}_J^* = A_N^{-1} b_N$, with A_N and b_N defined in (13); for large enough N, $\hat{w}_J^*$ is well defined (Boyan, 2002).

Furthermore, let g_N denote a vector with elements $-(\phi_J(x_i)^\top \hat{w}_J^*)^2$.

We define the set $S_N^+ = \{\Phi_M w | w \in \mathbb{R}^m, Hw \leq g_N\}$, and denote by $\Pi_{q_N}^+$ a projection onto S_N^+ w.r.t. the q_N-weighted Euclidean norm. We also define the operator

$$T_N^+(w) \doteq Rr + 2\gamma RP_N\Phi_J \hat{w}_J^* + \gamma^2 P_N \Phi_M w,$$

which is the sampled version of T^+. Note that T_N^+ is a γ^2-contraction.

Consider now the following projected fixed point equation:

$$w_{M;N}^+ = \Pi_{q_N}^+ T_N^+ w_{M;N}^+, \tag{34}$$

and the iterative procedure

$$w_{k+1;N} = \Pi_{\Xi, \hat{W}_{M;N}}[w_{k;N} - \eta \Xi^{-1}(C_N w_{k;N} - d_N)], \tag{35}$$

where C_N and d_N are defined in (13), Ξ is an arbitrary positive definite matrix, $\eta \in \mathbb{R}$ is a positive step size, and $\Pi_{\Xi, \hat{W}_{M;N}}$ denotes a projection onto the convex set $\hat{W}_{M;N} = \{w | Hw \leq g_N\}$ with respect to the Ξ weighted Euclidean norm.

C.2 Convergence in k

By definition, the sampled C_N, d_N, q_N and $\hat{w}_J^*$ correspond to their non-sampled counterparts C, d, q and w_J^*, respectively, on an MDP with the empirical probabilities P_N and $\zeta_{0;N}$ replacing P and ζ_0. As a result, applying Lemma 16 to Eq. 35, we have that $w_{k;N}$ converges to $w_{M;N}^+$. Therefore, for each N and $\delta > 0$ there exists some $k(N, \delta)$ such that for all $k > k(N, \delta)$

$$\|w_{k;N} - w_{M;N}^+\| \leq \delta. \tag{36}$$

C.3 Convergence in N

We will now show that as $N \to \infty$, $w_{M;N}^+ \to w_M^+$.

Let $\epsilon, \tilde{\epsilon} > 0$. We claim that w.p. 1, there exists $N(\epsilon, \tilde{\epsilon})$, such that for all $N > N(\epsilon, \tilde{\epsilon})$ we have

$$\|\Pi_q^+ T^+ w - \Pi_{q_N}^+ T_N^+ w\|_q \leq \epsilon + \tilde{\epsilon}\|w\|_q, \quad \forall w. \tag{37}$$

We now prove (37). First, we have:

$$\begin{aligned}\|\Pi_q^+ T^+ w - \Pi_{q_N}^+ T_N^+ w\|_q &= \|\Pi_q^+ T^+ w + \Pi_q^+ T_N^+ w - \Pi_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q \\ &\leq \underbrace{\|\Pi_q^+ T^+ w - \Pi_q^+ T_N^+ w\|_q}_{A} + \underbrace{\|\Pi_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q}_{B}.\end{aligned} \tag{38}$$

C.3.1 A Bound on (A):

We have:

$$\begin{aligned}\|\Pi_q^+ T^+ w - \Pi_q^+ T_N^+ w\|_q &\leq \|T^+ w - T_N^+ w\|_q \\ &= \|2\gamma RP\Phi_J w_J^* - 2\gamma RP_N\Phi_J \hat{w}_J^* + \gamma^2(P - P_N)\Phi_M w\|_q \\ &\leq \|2\gamma RP\Phi_J w_J^* - 2\gamma RP_N\Phi_J \hat{w}_J^*\|_q + \|\gamma^2(P - P_N)\Phi_M w\|_q \\ &\triangleq \eta_1(N) + \|\gamma^2(P - P_N)\Phi_M w\|_q \\ &\leq \eta_1(N) + \eta_2(N)\|w\|_q,\end{aligned} \tag{39}$$

where the first inequality is by the non-expansion property of the projection, and the third inequality is by defining $\eta_2(N)$ as the $\|\cdot\|_q$ induced matrix norm of $\gamma^2(P-P_N)\Phi_M$ (Horn and Johnson, 2012, Definition 5.6.1).

C.3.2 A Bound on (B):

Denote by $\hat{\Pi}_q^+$ a projection onto S_N^+ w.r.t. the q-weighted Euclidean norm. We have

$$\begin{aligned}\|\Pi_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q &= \|\Pi_q^+ T_N^+ w + \hat{\Pi}_q^+ T_N^+ w - \hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q \\ &\le \underbrace{\|\Pi_q^+ T_N^+ w - \hat{\Pi}_q^+ T_N^+ w\|_q}_{B_1} + \underbrace{\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q}_{B_2}.\end{aligned}$$

C.3.3 A Bound on (B_1):

We bound B_1 using a result of Yen (1995), which gives a general Lipschitz bound for perturbations of projections onto convex polyhedra (S_N^+ by definition is a convex polyhedron). By theorem 2.1 of Yen (1995), for all w, there exists a constant K such that

$$\|\Pi_q^+ T_N^+ w - \hat{\Pi}_q^+ T_N^+ w\|_q \le K\|g - g_N\|_2 \triangleq \eta_3(N). \tag{40}$$

C.3.4 A Bound on (B_2):

We bound B_2 using Lemma 19, which yields:

$$\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q^2 \le 2\|q - q_N\|_\infty \left(\|\hat{\Pi}_q^+ T_N^+ w - T_N^+ w\|_2^2 + \|\Pi_{q_N}^+ T_N^+ w - T_N^+ w\|_2^2\right).$$

By norm equivalence on finite-dimensional spaces, there exists λ such that $\|x\|_2 \le \lambda\|x\|_q$ and $\|x\|_2 \le \lambda\|x\|_{q_N}$ for all x. Therefore

$$\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q^2 \le 2\|q - q_N\|_\infty \lambda^2 \left(\|\hat{\Pi}_q^+ T_N^+ w - T_N^+ w\|_q^2 + \|\Pi_{q_N}^+ T_N^+ w - T_N^+ w\|_{q_N}^2\right).$$

For any $\hat{s} \in S_N^+$ we now have, by definition of the projections $\hat{\Pi}_q^+$ and $\Pi_{q_N}^+$:

$$\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q^2 \le 2\|q - q_N\|_\infty \lambda^2 \left(\|\hat{s} - T_N^+ w\|_q^2 + \|\hat{s} - T_N^+ w\|_{q_N}^2\right).$$

As before, by norm equivalence on finite-dimensional spaces, there exists $\tilde{\lambda}$ such that $\|x\|_{q_N} \le \lambda\|x\|_q$ for all x, therefore

$$\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q^2 \le 2\|q - q_N\|_\infty \lambda^2 (1+\tilde{\lambda}^2)\|\hat{s} - T_N^+ w\|_q^2,$$

and setting $\bar{\lambda} = \sqrt{\lambda^2(1+\tilde{\lambda}^2)}$ we have

$$\begin{aligned}\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q &\le \sqrt{2\|q - q_N\|_\infty}\bar{\lambda}\|\hat{s} - T_N^+ w\|_q \\ &\le \sqrt{2\|q - q_N\|_\infty}\bar{\lambda}\left(\|\hat{s}\|_q + \|T_N^+ w\|_q\right) \\ &\le \sqrt{2\|q - q_N\|_\infty}\bar{\lambda}\left(\|\hat{s}\|_q + C + \|w\|_q\right),\end{aligned}$$

where the constant C exists since T_N^+ is linear and a contraction. Therefore, setting $\eta_4(N) = \sqrt{2\|q - q_N\|_\infty}\bar{\lambda}\left(\|\hat{s}\|_q + C\right)$ and $\eta_5(N) = \sqrt{2\|q - q_N\|_\infty}\bar{\lambda}$ we have

$$\|\hat{\Pi}_q^+ T_N^+ w - \Pi_{q_N}^+ T_N^+ w\|_q \le \eta_4(N) + \eta_5(N)\|w\|_q. \tag{41}$$

C.3.5 Proof of (37):

We now return to (38), where, using (39), (40), and (41) we have

$$\|\Pi_q^+ T^+ w - \Pi_{q_N}^+ T_N^+ w\|_q \le \eta_1(N) + \eta_2(N)\|w\|_q + \eta_3(N) + \eta_4(N) + \eta_5(N)\|w\|_q.$$

The uniform convergence of empirical distributions (Van der Vaart, 2000, Theorem 19.1) guarantees that P_N and $\zeta_{0;N}$ uniformly converge to P and ζ_0 w.p. 1, respectively, and therefore $q_N \to q$ and $\hat{w}_J^* \to w_J^*$ w.p. 1. Therefore, for every $\epsilon, \tilde{\epsilon} > 0$, w.p. 1 there is some $N(\epsilon, \tilde{\epsilon})$ such that for $N > N(\epsilon, \tilde{\epsilon})$ we have $\eta_1(N)+\eta_3(N)+\eta_4(N) \le \epsilon$, and $\eta_2(N)+\eta_5(N) \le \tilde{\epsilon}$, therefore Eq. (37) holds.

Using Lemma 18 and Eq. (37) we have that for $N > N(\epsilon, \tilde{\epsilon})$

$$\|w_{M;N}^+ - w_M^+\|_q \le \frac{\epsilon + \tilde{\epsilon}\|w_M^+\|_q}{1-\gamma}. \tag{42}$$

C.4 Convergence in k and N

Finally, using (42) and (36) we have that for any $\bar{\epsilon} > 0$, w.p. 1 there is a $N(\bar{\epsilon})$ such that for any $N > N(\bar{\epsilon})$ there is a $k(N, \bar{\epsilon})$, such that for all $k > k(N, \bar{\epsilon})$

$$\|w_{k;N} - w_M^+\| \le \bar{\epsilon}.$$

References

D. P. Bertsekas. Temporal difference methods for general projected equations. *IEEE Transactions on Automatic Control*, 56(9):2128–2139, 2011.

D. P. Bertsekas. *Dynamic Programming and Optimal Control, Vol II.* Athena Scientific, fourth edition, 2012.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, 1996.

V. S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint.* Cambridge University Press, 2008.

J. A. Boyan. Technical update: least-squares temporal difference learning. *Machine Learning*, 49(2):233–246, 2002.

J. C. Cox, S. A. Ross, and M. Rubinstein. Option pricing: A simplified approach. *Journal of Financial Economics*, 7(3):229–263, 1979.

D. Duffie. *Dynamic Asset Pricing Theory.* Princeton University Press, 2010.

Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with Gaussian processes. In *International Conference on Machine Learning*, 2005.

J. A. Filar, L. C. M. Kallenberg, and H. M. Lee. Variance-penalized Markov decision processes. *Mathematics of Operations Research*, 14(1):pp. 147–161, 1989.

J. A. Filar, D. Krass, and K. W. Ross. Percentile performance criteria for limiting average Markov decision processes. *IEEE Transaction on Automatic Control*, 40(1):2–10, 1995.

P. Geibel and F. Wysotzki. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research*, 24(1):81–108, 2005.

G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford university press, 2001.

J. B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms I: Fundamentals*. Springer science & business media, 2013.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, second edition, 2012.

J. C. Hull. *Options, Futures, and Other Derivatives (6th edition)*. Prentice Hall, 2006.

H. K. Khalil and J. W. Grizzle. *Nonlinear Systems*. Prentice hall New Jersey, 1996.

V. Konda. *Actor-Critic Algorithms*. PhD thesis, Department of Computer Science and Electrical Engineering, MIT, Cambridge, MA, 2002.

V. R. Konda and John N Tsitsiklis. On actor-critic algorithms. *SIAM Journal on Control and Optimization*, 42(4):1143–1166, 2003.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of LSTD. In *International Conference on Machine Learning*, 2010.

Y. Li, C. Szepesvari, and D. Schuurmans. Learning exercise policies for American options. In *International Conference on Artificial Intelligence and Statistics, JMLR: W&CP*, volume 5, pages 352–359, 2009.

F. A. Longstaff and E. S. Schwartz. Valuing American options by simulation: a simple least-squares approach. *Review of Financial Studies*, 14(1):113–147, 2001.

S. Mannor and J. N. Tsitsiklis. Algorithmic aspects of mean-variance optimization in Markov decision processes. *European Journal of Operational Research*, 231(3):645 – 653, 2013. ISSN 0377-2217.

O. Mihatsch and R. Neuneier. Risk-sensitive reinforcement learning. *Machine Learning*, 49 (2):267–290, 2002.

J. Moody and M. Saffell. Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 12(4):875–889, 2001.

T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka. Parametric return density estimation for reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

W. B. Powell. *Approximate Dynamic Programming*. John Wiley and Sons, 2011.

L. A. Prashanth and M. Ghavamzadeh. Actor-critic algorithms for risk-sensitive MDPs. In *Advances in Neural Information Processing Systems*, 2013.

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

A. Ruszczyński. Risk-averse dynamic programming for Markov decision processes. *Mathematical Programming*, 125(2):235–261, 2010.

M. Sato, H. Kimura, and S. Kobayashi. TD algorithm for the variance of return and mean-variance reinforcement learning. *Transactions of the Japanese Society for Artificial Intelligence*, 16:353–362, 2001.

W. F. Sharpe. Mutual fund performance. *The Journal of Business*, 39(1):119–138, 1966.

S. M. Shortreed, E. Laber, D. J. Lizotte, T. S. Stroup, J. Pineau, and S. A. Murphy. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine learning*, 84(1):109–136, 2011.

M. J. Sobel. The variance of discounted Markov decision processes. *Journal of Applied Probability*, pages 794–802, 1982.

R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, 1988.

R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998.

A. Tamar and S. Mannor. Variance adjusted actor critic algorithms. *arXiv preprint arXiv:1310.3697, http://arxiv.org/abs/1310.3697*, 2013.

A. Tamar, D. Di Castro, and S. Mannor. Policy gradients with variance related risk criteria. In *International Conference on Machine Learning*, 2012.

A. Tamar, D. Di Castro, and S. Mannor. Temporal difference methods for the variance of the reward to go. In *International Conference on Machine Learning*, 2013.

A. Tamar, S. Mannor, and H. Xu. Scaling up robust MDPs using function approximation. In *International Conference on Machine Learning*, 2014.

G. Tesauro. Temporal difference learning and TD-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

J. N. Tsitsiklis and B. Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

A. W. Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge university press, 2000.

N. D. Yen. Lipschitz continuity of solutions of variational inequalities with a parametric polyhedral constraint. *Mathematics of Operations Research*, 20(3):pp. 695–708, 1995.

Journal of Machine Learning Research 17 (2016) 1-45 Submitted 8/14; Revised 7/15; Published 3/16

Convex Calibration Dimension for Multiclass Loss Matrices

Harish G. Ramaswamy harish_gurup@csa.iisc.ernet.in
Shivani Agarwal shivani@csa.iisc.ernet.in
Department of Computer Science and Automation
Indian Institute of Science
Bangalore 560012, India

Editor: Ingo Steinwart

Abstract

We study consistency properties of surrogate loss functions for general multiclass learning problems, defined by a general multiclass loss matrix. We extend the notion of classification calibration, which has been studied for binary and multiclass 0-1 classification problems (and for certain other specific learning problems), to the general multiclass setting, and derive necessary and sufficient conditions for a surrogate loss to be calibrated with respect to a loss matrix in this setting. We then introduce the notion of *convex calibration dimension* of a multiclass loss matrix, which measures the smallest 'size' of a prediction space in which it is possible to design a convex surrogate that is calibrated with respect to the loss matrix. We derive both upper and lower bounds on this quantity, and use these results to analyze various loss matrices. In particular, we apply our framework to study various subset ranking losses, and use the convex calibration dimension as a tool to show both the existence and non-existence of various types of convex calibrated surrogates for these losses. Our results strengthen recent results of Duchi et al. (2010) and Calauzènes et al. (2012) on the non-existence of certain types of convex calibrated surrogates in subset ranking. We anticipate the convex calibration dimension may prove to be a useful tool in the study and design of surrogate losses for general multiclass learning problems.

Keywords: Statistical consistency, multiclass loss, loss matrix, surrogate loss, convex surrogates, calibrated surrogates, classification calibration, subset ranking.

1. Introduction

There has been significant interest and progress in recent years in understanding consistency properties of surrogate risk minimization algorithms for various learning problems, such as binary classification, multiclass 0-1 classification, and various forms of ranking and multi-label prediction problems (Lugosi and Vayatis, 2004; Jiang, 2004; Zhang, 2004a; Steinwart, 2005; Bartlett et al., 2006; Zhang, 2004b; Tewari and Bartlett, 2007; Steinwart, 2007; Cossock and Zhang, 2008; Xia et al., 2008; Duchi et al., 2010; Ravikumar et al., 2011; Buffoni et al., 2011; Gao and Zhou, 2011; Kotlowski et al., 2011). Any such problem that involves a finite number of class labels and predictions can be viewed as an instance of a general multiclass learning problem, whose structure is defined by a suitable loss matrix. While the above studies have enabled an understanding of learning problems corresponding to certain forms of loss matrices, a framework for analyzing consistency properties for a general multiclass problem, defined by a general loss matrix, has remained elusive.

In this paper, we develop a unified framework for studying consistency properties of surrogate losses for such general multiclass learning problems, defined by a general multiclass loss matrix. For algorithms minimizing a surrogate loss, the question of consistency with respect to the target loss matrix reduces to the question of *calibration* of the surrogate loss with respect to the target loss.[1] We start by giving both necessary and sufficient conditions for a surrogate loss function to be calibrated with respect to any given target loss matrix. These conditions generalize previous conditions for the multiclass 0-1 loss studied for example by Tewari and Bartlett (2007). We then introduce the notion of *convex calibration dimension* of a loss matrix, a fundamental quantity that measures the smallest 'size' of a prediction space in which it is possible to design a convex surrogate that is calibrated with respect to the given loss matrix. This quantity can be viewed as representing one measure of the intrinsic 'difficulty' of the loss, and has a non-trivial behavior in the sense that one can give examples of loss matrices defined on the same number of class labels that have very different values of the convex calibration dimension, ranging from one (in which case one can achieve consistency by learning a single real-valued function) to practically the number of classes (in which case one must learn as many real-valued functions as the number of classes). We give upper and lower bounds on this quantity in terms of various algebraic and geometric properties of the loss matrix, and apply these results to analyze various loss matrices.

As concrete applications of our framework, we use the convex calibration dimension as a tool to study various loss matrices that arise in subset ranking problems, including the normalized discounted cumulative gain (NDCG), pairwise disagreement (PD), and mean average precision (MAP) losses. A popular practice in subset ranking, where one needs to rank a set of r documents by relevance to a query, has been to learn r real-valued scoring functions by minimizing a convex surrogate loss in r dimensions, and to then sort the r documents based on these scores. As discussed recently by Duchi et al. (2010) and Calauzènes et al. (2012), such an approach cannot be consistent for the PD and MAP losses, since these losses do not admit convex calibrated surrogates in r dimensions that can be used together with the sorting operation. We obtain a stronger result; in particular, we show that the convex calibration dimension of these losses is lower bounded by a quadratic function of r, which means that if minimizing a convex surrogate loss, one necessarily needs to learn $\Omega(r^2)$ real-valued functions to achieve consistency for these losses.

1.1 Related Work

There has been much work in recent years on consistency and calibration of surrogate losses for various learning problems. We give a brief overview of this body of work here.

Initial work on consistency of surrogate risk minimization algorithms focused largely on binary classification. For example, Steinwart (2005) showed the consistency of support vector machines with universal kernels for the problem of binary classification; Jiang (2004)

1. Assuming the surrogate risk minimization procedure is itself consistent (with respect to the surrogate loss); in most cases, this can be achieved by minimizing the surrogate risk over a function class that approaches a universal function class as the training sample size increases, e.g. see Bartlett et al. (2006).

and Lugosi and Vayatis (2004) showed similar results for boosting methods. Bartlett et al. (2006) and Zhang (2004a) studied the calibration of margin-based surrogates for binary classification. In particular, in their seminal work, Bartlett et al. (2006) established that the property of 'classification calibration' of a surrogate loss is equivalent to its minimization yielding 0-1 consistency, and gave a simple necessary and sufficient condition for convex margin-based surrogates to be calibrated w.r.t. the binary 0-1 loss. More recently, Reid and Williamson (2010) analyzed the calibration of a general family of surrogates termed proper composite surrogates for binary classification. Variants of standard 0-1 binary classification have also been studied; for example, Yuan and Wegkamp (2010) studied consistency for the problem of binary classification with a reject option, and Scott (2012) studied calibrated surrogates for cost-sensitive binary classification.

Over the years, there has been significant interest in extending the understanding of consistency and calibrated surrogates to various multiclass learning problems. Early work in this direction, pioneered by Zhang (2004b) and Tewari and Bartlett (2007), considered mainly the multiclass 0-1 classification problem. This work generalized the framework of Bartlett et al. (2006) to the multiclass 0-1 setting and used these results to study calibration of various surrogates proposed for multiclass 0-1 classification, such as the surrogates of Weston and Watkins (1999), Crammer and Singer (2001), and Lee et al. (2004). In particular, while the multiclass surrogate of Lee et al. (2004) was shown to calibrated for multiclass 0-1 classification, it was shown that several other widely used multiclass surrogates are in fact not calibrated for multiclass 0-1 classification.

More recently, there has been much work on studying consistency and calibration for various other learning problems that also involve finite label and prediction spaces. For example, Gao and Zhou (2011) studied consistency and calibration for multi-label prediction with the Hamming loss. Another prominent class of learning problems for which consistency and calibration have been studied recently is that of subset ranking, where instances contain queries together with sets of documents, and the goal is to learn a prediction model that given such an instance ranks the documents by relevance to the query. Various subset ranking losses have been investigated in recent years. Cossock and Zhang (2008) studied subset ranking with the discounted cumulative gain (DCG) ranking loss, and gave a simple surrogate calibrated w.r.t. this loss; Ravikumar et al. (2011) further studied subset ranking with the normalized DCG (NDCG) loss. Xia et al. (2008) considered the 0-1 loss applied to permutations. Duchi et al. (2010) focused on subset ranking with the pairwise disagreement (PD) loss, and showed that several popular convex score-based surrogates used for this problem are in fact not calibrated w.r.t. this loss; they also conjectured that such surrogates may not exist. Calauzènes et al. (2012) showed conclusively that there do not exist any convex score-based surrogates that are calibrated w.r.t. the PD loss, or w.r.t. the mean average precision (MAP) or expected reciprocal rank (ERR) losses. Finally, in a more general study of subset ranking losses, Buffoni et al. (2011) introduced the notion of 'standardization' for subset ranking losses, and gave a way to construct convex calibrated score-based surrogates for subset ranking losses that can be 'standardized'; they showed that while the DCG and NDCG losses can be standardized, the MAP and ERR losses cannot be standardized.

We also point out that in a related but different context, consistency of ranking has also been studied in the instance ranking setting (Clémençon and Vayatis, 2007; Clémençon et al., 2008; Kotlowski et al., 2011; Agarwal, 2014).

Finally, Steinwart (2007) considered consistency and calibration in a very general setting. More recently, Pires et al. (2013) used Steinwart's techniques to obtain surrogate regret bounds for certain surrogates w.r.t. general multiclass losses, and Ramaswamy et al. (2013) showed how to design explicit convex calibrated surrogates for any low-rank loss matrix.

1.2 Contributions of this Paper

As noted above, we develop a unified framework for studying consistency and calibration for general multiclass (finite-output) learning problems, described by a general loss matrix. We give both necessary conditions and sufficient conditions for a surrogate loss to be calibrated w.r.t. a given multiclass loss matrix, and introduce the notion of *convex calibration dimension* of a loss matrix, which measures the smallest 'size' of a prediction space in which it is possible to design a convex surrogate that is calibrated with respect to the loss matrix. We derive both upper and lower bounds on this quantity in terms of certain algebraic and geometric properties of the loss matrix, and apply these results to study various subset ranking losses. In particular, we obtain stronger results on the non-existence of convex calibrated surrogates for certain types of subset ranking losses than previous results in the literature (and also positive results on the existence of convex calibrated surrogates for these losses in higher dimensions). The following is a summary of the main differences from the conference version of this paper (Ramaswamy and Agarwal, 2012):

- Enhanced definition of positive normal sets of a surrogate loss at a sequence of points (Definition 5; this is required for proofs of stronger versions of our earlier results).
- Stronger necessary condition for calibration (Theorem 7).
- Stronger versions of upper and lower bounds on the convex calibration dimension, with full proofs (Theorems 12, 16).
- Conditions under which the upper and lower bounds are tight (Section 4.3).
- Application to a more general setting of the PD loss (Section 5.2).
- Additional applications to the NDCG and MAP losses (Sections 5.1, 5.3).
- Additional examples and illustrations throughout (Examples 5, 6, 7, 9; Figures 3, 4).
- Minor improvements and changes in emphasis in notation and terminology.

1.3 Organization

We start in Section 2 with some preliminaries and examples that will be used as running examples to illustrate concepts throughout the paper, and formalize the notion of calibration with respect to a general multiclass loss matrix. In Section 3, we derive both necessary conditions and sufficient conditions for calibration with respect to general loss matrices; these are both of independent interest and useful in our later results. Section 4 introduces the notion of convex calibration dimension of a loss matrix and derives both upper and lower bounds on this quantity. In Section 5, we apply our results to study the convex calibration dimension of various subset ranking losses. We conclude with a brief discussion in Section 6. Shorter proofs are included in the main text; all longer proofs are collected in Section 7 so as to maintain easy readability of the main text. The only exception to

this is proofs of Lemma 2 and Theorem 3, which closely follow earlier proofs of Tewari and Bartlett (2007) and are included for completeness in Appendix A. Some calculations are given in Appendices B and C.

2. Preliminaries, Examples, and Background

In this section we set up basic notation (Section 2.1), give background on multiclass loss matrices and risks (Section 2.2) and on multiclass surrogates and calibration (Section 2.3), and then define certain properties associated with multiclass losses and surrogates that will be useful in our study (Section 2.4).

2.1 Notation

Throughout the paper, we denote $\mathbb{R} = (-\infty, \infty)$, $\mathbb{R}_+ = [0, \infty)$, $\overline{\mathbb{R}} = [-\infty, \infty]$, $\overline{\mathbb{R}}_+ = [0, \infty]$. Similarly, $\mathbb{Z}$ and $\mathbb{Z}_+$ denote the sets of all integers and non-negative integers, respectively. For $n \in \mathbb{Z}_+$, we denote $[n] = \{1, \ldots, n\}$. For a predicate ϕ, we denote by $\mathbf{1}(\phi)$ the indicator of ϕ, which takes the value 1 if ϕ is true and 0 otherwise. For $z \in \mathbb{R}$, we denote $z_+ = \max(0, z)$. For a vector $\mathbf{v} \in \mathbb{R}^n$, we denote $\|\mathbf{v}\|_0 = \sum_{i=1}^n \mathbf{1}(v_i \neq 0)$ and $\|\mathbf{v}\|_1 = \sum_{i=1}^n |v_i|$. For a set $\mathcal{A} \subseteq \mathbb{R}^n$, we denote by relint($\mathcal{A}$) the relative interior of $\mathcal{A}$, by cl($\mathcal{A}$) the closure of $\mathcal{A}$, by span($\mathcal{A}$) the linear span (or linear hull) of $\mathcal{A}$, by aff($\mathcal{A}$) the affine hull of $\mathcal{A}$, and by conv($\mathcal{A}$) the convex hull of $\mathcal{A}$. For a vector space $\mathcal{V}$, we denote by dim($\mathcal{V}$) the dimension of $\mathcal{V}$. For a matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, we denote by rank($\mathbf{M}$) the rank of $\mathbf{M}$, by affdim($\mathbf{M}$) the affine dimension of the set of columns of $\mathbf{M}$ (i.e. the dimension of the subspace parallel to the affine hull of the columns of $\mathbf{M}$), by null($\mathbf{M}$) the null space of $\mathbf{M}$, and by nullity($\mathbf{M}$) the nullity of $\mathbf{M}$ (i.e. the dimension of the null space of $\mathbf{M}$). We denote by Δ_n the probability simplex in $\mathbb{R}^n$: $\Delta_n = \{\mathbf{p} \in \mathbb{R}^n_+ : \sum_{i=1}^n p_i = 1\}$. Finally, we denote by Π_n the set of all permutations of $[n]$, i.e. the set of all bijective mappings $\sigma : [n] \to [n]$; for a permutation $\sigma \in \Pi_n$ and element $i \in [n]$, $\sigma(i)$ therefore represents the position of element i under σ.

2.2 Multiclass Losses and Risks

The general multiclass learning problem we consider can be described as follows: There is a finite set of *class labels* $\mathcal{Y}$ and a finite set of possible *predictions* $\widehat{\mathcal{Y}}$, which we take without loss of generality to be $\mathcal{Y} = [n]$ and $\widehat{\mathcal{Y}} = [k]$ for some $n, k \in \mathbb{Z}_+$. We are given training examples $(X_1, Y_1), \ldots, (X_m, Y_m)$ drawn i.i.d. from a distribution D on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is an instance space, and the goal is to learn from these examples a prediction model $h : \mathcal{X} \to \widehat{\mathcal{Y}}$ which given a new instance $x \in \mathcal{X}$, makes a prediction $\widehat{y} = h(x) \in \widehat{\mathcal{Y}}$. In many common learning problems, the label and prediction spaces are the same, i.e. $\widehat{\mathcal{Y}} = \mathcal{Y}$, but in general, these could be different (e.g. when there is an 'abstain' option available to a classifier, in which case $k = n + 1$).

The performance of a prediction model is measured via a *loss function* $\ell : \mathcal{Y} \times \widehat{\mathcal{Y}} \to \mathbb{R}_+$, or equivalently, by a *loss matrix* $\mathbf{L} \in \mathbb{R}_+^{n \times k}$, with (y, t)-th element given by $\ell_{yt} = \ell(y, t)$; here $\ell_{yt} = \ell(y, t)$ defines the penalty incurred on predicting $t \in [k]$ when the true label is $y \in [n]$. We will use the notions of loss matrix and loss function interchangeably. Some examples of common multiclass loss functions and corresponding loss matrices are given below:

Example 1 (0-1 loss) *Here $\mathcal{Y} = \widehat{\mathcal{Y}} = [n]$, and the loss incurred is 1 if the predicted label t is different from the actual class label y, and 0 otherwise:*

$$\ell^{0\text{-}1}(y, t) \;=\; \mathbf{1}\big(t \neq y\big) \qquad \forall y, t \in [n]\,.$$

The loss matrix $\mathbf{L}^{0\text{-}1}$ for $n = 3$ is shown in Figure 1(a). This is one of the most commonly used multiclass losses, and is suitable when all prediction errors are considered equal.

Example 2 (Ordinal regression loss) *Here $\mathcal{Y} = \widehat{\mathcal{Y}} = [n]$, and predictions t farther away from the actual class label y are penalized more heavily, e.g. using absolute distance:*

$$\ell^{\text{ord}}(y, t) \;=\; |t - y| \qquad \forall y, t \in [n]\,.$$

The loss matrix $\mathbf{L}^{\text{ord}}$ for $n = 3$ is shown in Figure 1(b). This loss is often used when the class labels satisfy a natural ordinal property, for example in evaluating recommender systems that predict the number of stars (say out of 5) assigned to a product by user.

Example 3 (Hamming loss) *Here $\mathcal{Y} = \widehat{\mathcal{Y}} = [2^r]$ for some $r \in \mathbb{Z}_+$, and the loss incurred on predicting t when the actual class label is y is the number of bit-positions in which the r-bit binary representations of $t - 1$ and $y - 1$ differ:*

$$\ell^{\text{Ham}}(y, t) \;=\; \sum_{i=1}^{r} \mathbf{1}\big((t-1)_i \neq (y-1)_i\big) \qquad \forall y, t \in [2^r]\,,$$

where for each $z \in \{0, \ldots, 2^r - 1\}$, $z_i \in \{0, 1\}$ denotes the i-th bit in the r-bit binary representation of z. The loss matrix $\mathbf{L}^{\text{Ham}}$ for $r = 2$ is shown in Figure 1(c). This loss is frequently used in sequence learning applications, where each element in $\mathcal{Y} = \widehat{\mathcal{Y}}$ is a binary sequence of length r, and the loss in predicting a sequence $\mathbf{t} \in \{0, 1\}^r$ when the true label sequence is $\mathbf{y} \in \{0, 1\}^r$ is simply the Hamming distance between the two sequences.

Example 4 ('Abstain' loss) *Here $\mathcal{Y} = [n]$ and $\widehat{\mathcal{Y}} = [n + 1]$, where $t = n + 1$ denotes a prediction of 'abstain' (or 'reject'). One possible loss function in this setting assigns a loss of 1 to incorrect predictions in $[n]$, 0 to correct predictions, and $\frac{1}{2}$ for abstaining:*

$$\ell^{(?)}(y, t) \;=\; \mathbf{1}\big(t \in [n]\big) \cdot \mathbf{1}\big(t \neq y\big) + \frac{1}{2} \cdot \mathbf{1}\big(t = n + 1\big) \qquad \forall y \in [n], t \in [n + 1]\,.$$

The loss matrix $\mathbf{L}^{(?)}$ for $n = 3$ is shown in Figure 1(d). This type of loss is suitable in applications where making an erroneous prediction is more costly than simply abstaining. For example, in medical diagnosis applications, when uncertain about the correct prediction, it may be better to abstain and request human intervention rather than make a misdiagnosis.

As noted above, given examples $(X_1, Y_1), \ldots, (X_m, Y_m)$ drawn i.i.d. from a distribution D on $\mathcal{X} \times [n]$, the goal is to learn a prediction model $h : \mathcal{X} \to [k]$. More specifically, given a target loss matrix $\mathbf{L} \in \mathbb{R}_+^{n \times k}$ with (y, t)-th element ℓ_{yt}, the goal is to learn a model $h : \mathcal{X} \to [k]$ with small expected loss on a new example drawn randomly from D, which we will refer to as the $\mathbf{L}$-*risk* or $\mathbf{L}$-*error* of h:

$$\text{er}_D^{\mathbf{L}}[h] \;\triangleq\; \mathbf{E}_{(X,Y)\sim D}\big[\ell_{Y,h(X)}\big]\,. \tag{1}$$

$$\underset{\text{(a)}}{\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}} \quad \underset{\text{(b)}}{\begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}} \quad \underset{\text{(c)}}{\begin{bmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 2 & 1 & 1 & 0 \end{bmatrix}} \quad \underset{\text{(d)}}{\begin{bmatrix} 0 & 1 & 1 & \frac{1}{2} \\ 1 & 0 & 1 & \frac{1}{2} \\ 1 & 1 & 0 & \frac{1}{2} \end{bmatrix}}$$

Figure 1: Loss matrices corresponding to Examples 1-4: (a) $\mathbf{L}^{0\text{-}1}$ for $n = 3$; (b) $\mathbf{L}^{\text{ord}}$ for $n = 3$; (c) $\mathbf{L}^{\text{Ham}}$ for $r = 2$ ($n = 4$); (d) $\mathbf{L}^{(?)}$ for $n = 3$.

Clearly, denoting the t-th column of $\mathbf{L}$ as $\boldsymbol{\ell}_t = (\ell_{1t}, \ldots, \ell_{nt})^\top \in \mathbb{R}^n_+$, and the class probability vector at an instance $x \in \mathcal{X}$ under D as $\mathbf{p}(x) = (p_1(x), \ldots, p_n(x))^\top \in \Delta_n$, where $p_y(x) = \mathbf{P}(Y = y \mid X = x)$ under D, the $\mathbf{L}$-risk of h can be written as

$$\text{er}^{\mathbf{L}}_D[h] \;=\; \mathbf{E}_X\left[\sum_{y=1}^{n} p_y(X)\, \ell_{y,h(X)}\right] \;=\; \mathbf{E}_X\left[\mathbf{p}(X)^\top \boldsymbol{\ell}_{h(X)}\right]. \tag{2}$$

The *optimal* $\mathbf{L}$*-risk* or *optimal* $\mathbf{L}$*-error* for a distribution D is then simply the smallest $\mathbf{L}$-risk or $\mathbf{L}$-error that can be achieved by any model h:

$$\text{er}^{\mathbf{L},*}_D \;\triangleq\; \inf_{h:\mathcal{X}\to[k]} \text{er}^{\mathbf{L}}_D[h] \;=\; \inf_{h:\mathcal{X}\to[k]} \mathbf{E}_X\left[\mathbf{p}(X)^\top \boldsymbol{\ell}_{h(X)}\right] \;=\; \mathbf{E}_X\left[\min_{t\in[k]} \mathbf{p}(X)^\top \boldsymbol{\ell}_t\right]. \tag{3}$$

Ideally, one would like to minimize (approximately) the $\mathbf{L}$-risk, e.g. by selecting a model that minimizes the average $\mathbf{L}$-loss on the training examples among some suitable class of models. However, minimizing the discrete $\mathbf{L}$-risk directly is typically computationally difficult. Consequently, one usually minimizes a (convex) *surrogate* risk instead.

2.3 Multiclass Surrogates and Calibration

Let $d \in \mathbb{Z}_+$, and let $\mathcal{C} \subseteq \mathbb{R}^d$ be a convex set. A *surrogate loss function* $\psi : \mathcal{Y} \times \mathcal{C} \to \mathbb{R}_+$ acting on the *surrogate prediction space* $\mathcal{C}$ assigns a penalty $\psi(y, \mathbf{u})$ on making a surrogate prediction $\mathbf{u} \in \mathcal{C}$ when the true label is $y \in [n]$. The ψ*-risk* or ψ*-error* of a surrogate prediction model $\mathbf{f} : \mathcal{X} \to \mathcal{C}$ w.r.t. a distribution D on $\mathcal{X} \times [n]$ is then defined as

$$\text{er}^{\psi}_D[\mathbf{f}] \;\triangleq\; \mathbf{E}_{(X,Y)\sim D}\Big[\psi\big(Y, \mathbf{f}(X)\big)\Big]. \tag{4}$$

The surrogate ψ can be represented via n real-valued functions $\psi_y : \mathcal{C} \to \mathbb{R}_+$ for $y \in [n]$, defined as $\psi_y(\mathbf{u}) = \psi(y, \mathbf{u})$; equivalently, we can also represent the surrogate ψ as a vector-valued function $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+$, defined as $\boldsymbol{\psi}(\mathbf{u}) = \big(\psi_1(\mathbf{u}), \ldots, \psi_n(\mathbf{u})\big)^\top$. Clearly, the ψ-risk of $\mathbf{f}$ can then be written as

$$\text{er}^{\psi}_D[\mathbf{f}] \;=\; \mathbf{E}_X\left[\sum_{y=1}^{n} p_y(X)\, \psi_y(\mathbf{f}(X))\right] \;=\; \mathbf{E}_X\Big[\mathbf{p}(X)^\top \boldsymbol{\psi}(\mathbf{f}(X))\Big]. \tag{5}$$

The *optimal* ψ*-risk* or *optimal* ψ*-error* for a distribution D is then simply the smallest ψ-risk or ψ-error that can be achieved by any model $\mathbf{f}$:

$$\text{er}^{\psi,*}_D \;\triangleq\; \inf_{\mathbf{f}:\mathcal{X}\to\mathcal{C}} \text{er}^{\psi}_D[\mathbf{f}] \;=\; \inf_{\mathbf{f}:\mathcal{X}\to\mathcal{C}} \mathbf{E}_X\Big[\mathbf{p}(X)^\top \boldsymbol{\psi}(\mathbf{f}(X))\Big] \;=\; \mathbf{E}_X\left[\inf_{\mathbf{u}\in\mathcal{C}} \mathbf{p}(X)^\top \boldsymbol{\psi}(\mathbf{u})\right]. \tag{6}$$

We will find it convenient to define the sets

$$\mathcal{R}_\psi \;\triangleq\; \boldsymbol{\psi}(\mathcal{C}) \;\subseteq\; \mathbb{R}^n_+ \tag{7}$$

$$\mathcal{S}_\psi \;\triangleq\; \text{conv}(\mathcal{R}_\psi) \;\subseteq\; \mathbb{R}^n_+. \tag{8}$$

Clearly, the optimal ψ-risk can then also be written as

$$\text{er}^{\psi,*}_D \;=\; \mathbf{E}_X\left[\inf_{\mathbf{z}\in\mathcal{R}_\psi} \mathbf{p}(X)^\top \mathbf{z}\right] \;=\; \mathbf{E}_X\left[\inf_{\mathbf{z}\in\mathcal{S}_\psi} \mathbf{p}(X)^\top \mathbf{z}\right]. \tag{9}$$

Example 5 (Crammer-Singer surrogate) *The Crammer-Singer surrogate was proposed as a hinge-like surrogate loss for 0-1 multiclass classification (Crammer and Singer, 2001). For $\mathcal{Y} = [n]$, the Crammer-Singer surrogate $\boldsymbol{\psi}^{\text{CS}}$ acts on the surrogate prediction space $\mathcal{C} = \mathbb{R}^n$ and is defined as follows:*

$$\psi^{\text{CS}}_y(\mathbf{u}) \;=\; \max_{y'\in[n], y'\neq y} \big(1 - (u_y - u_{y'})\big)_+ \qquad \forall y \in [n], \mathbf{u} \in \mathbb{R}^n.$$

A surrogate $\boldsymbol{\psi}$ is convex if ψ_y is convex $\forall y \in [n]$. As an example, the Crammer-Singer surrogate defined above is clearly convex. Given training examples $(X_1, Y_1), \ldots, (X_m, Y_m)$ drawn i.i.d. from a distribution D on $\mathcal{X} \times [n]$, a (convex) surrogate risk minimization algorithm using a (convex) surrogate loss $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+$ learns a surrogate prediction model by minimizing (approximately, based on the training sample) the ψ-risk; the learned model $\mathbf{f} : \mathcal{X} \to \mathcal{C}$ is then used to make predictions in the original space $[k]$ via some transformation $\text{pred} : \mathcal{C} \to [k]$. This yields a prediction model $h : \mathcal{X} \to [k]$ for the original multiclass problem given by $h = (\text{pred} \circ \mathbf{f})$: the prediction on a new instance $x \in \mathcal{X}$ is given by $\text{pred}(\mathbf{f}(x))$, and the $\mathbf{L}$-risk incurred is $\text{er}^{\mathbf{L}}_D[\text{pred} \circ \mathbf{f}]$. As an example, several surrogate risk minimizing algorithms for multiclass classification with respect to 0-1 loss (including that based on the Crammer-Singer surrogate) use a surrogate space $\mathcal{C} = \mathbb{R}^n$, learn a function of the form $\mathbf{f} : \mathcal{X} \to \mathbb{R}^n$, and predict according to $\text{pred}(\mathbf{f}(x)) = \text{argmax}_{t\in[n]}\, f_t(x)$.

Under suitable conditions, surrogate risk minimization algorithms that approximately minimize the ψ-risk based on a training sample are known to be consistent with respect to the ψ-risk, i.e. to converge (in probability) to the optimal ψ-risk as the number of training examples m increases. This raises the natural question of whether, for a given loss matrix $\mathbf{L}$, there are surrogate losses ψ for which consistency with respect to the ψ-risk also guarantees consistency with respect to the $\mathbf{L}$-risk, i.e. guarantees convergence (in probability) to the optimal $\mathbf{L}$-risk (defined in Eq. (3)). As we shall see below, this amounts to the question of *calibration* of surrogate losses ψ w.r.t. a given target loss matrix $\mathbf{L}$, and has been studied in detail for the 0-1 loss and for square losses of the form $\ell(y, t) = a_y \mathbf{1}(t \neq y)$, which can be analyzed similarly to the 0-1 loss (Zhang, 2004b; Tewari and Bartlett, 2007). In this paper, we consider this question for general multiclass loss matrices $\mathbf{L} \in \mathbb{R}^{n\times k}_+$, including rectangular loss matrices with $k \neq n$. The only assumption we make on $\mathbf{L}$ is that for each $t \in [k]$, $\exists \mathbf{p} \in \Delta_n$ such that $\text{argmin}_{t'\in[k]}\, \mathbf{p}^\top \boldsymbol{\ell}_{t'} = \{t\}$ (otherwise the element $t \in [k]$ never needs to be predicted and can simply be ignored).

We will need the following definitions and basic results, generalizing those of Zhang (2004b), Bartlett et al. (2006), and Tewari and Bartlett (2007). The notion of calibration will be central to our study; as Theorem 3 below shows, calibration of a surrogate loss ψ

w.r.t. $\mathbf{L}$ corresponds to the property that consistency w.r.t. ψ-risk implies consistency w.r.t. $\mathbf{L}$-risk. Proofs of Lemma 2 and Theorem 3 can be found in Appendix A.

Definition 1 ($(\mathbf{L},\mathcal{P})$-calibration) *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$ and $\mathcal{P} \subseteq \Delta_n$. A surrogate loss function $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ is said to be* $(\mathbf{L},\mathcal{P})$-calibrated *if there exists a function* $\text{pred} : \mathcal{C} \to [k]$ *such that*

$$\forall \mathbf{p} \in \mathcal{P} : \quad \inf_{\mathbf{u}\in\mathcal{C}:\text{pred}(\mathbf{u})\notin \text{argmin}_t \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \;>\; \inf_{\mathbf{u}\in\mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \,.$$

If $\boldsymbol{\psi}$ is $(\mathbf{L},\Delta_n)$-calibrated, we simply say $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated.

The above definition of calibration clearly generalizes that used in the binary case. For example, in the case of binary 0-1 classification, with $n = k = 2$ and $\mathbf{L}^{0\text{-}1} = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$, the probability simplex Δ_2 is equivalent to the interval $[0, 1]$, and it can be shown that one only need consider surrogates on the real line, $\mathcal{C} = \mathbb{R}$, and the predictor 'sign'; in this case one recovers the familiar definition of binary classification calibration of Bartlett et al. (2006), namely that a surrogate $\psi : \mathbb{R} \to \mathbb{R}_+^2$ is $(\mathbf{L}^{0\text{-}1}, \Delta_2)$-calibrated if

$$\forall p \in [0, 1], p \neq \frac{1}{2} : \quad \inf_{u\in\mathbb{R}:\text{sign}(u)\neq\text{sign}(p-\frac{1}{2})} p\,\psi_1(u)+(1-p)\psi_2(u) \;>\; \inf_{u\in\mathbb{R}} p\,\psi_1(u)+(1-p)\psi_2(u) \,.$$

Similarly, in the case of binary cost-sensitive classification, with $n = k = 2$ and $\mathbf{L}^c = \left[\begin{smallmatrix} 0 & 1-c \\ c & 0 \end{smallmatrix}\right]$ where $c \in (0, 1)$ is the cost of a false positive and $(1-c)$ that of a false negative, one recovers the corresponding definition of Scott (2012), namely that a surrogate $\psi : \mathbb{R} \to \mathbb{R}_+^2$ is $(\mathbf{L}^c, \Delta_2)$-calibrated if

$$\forall p \in [0, 1], p \neq c : \quad \inf_{u\in\mathbb{R}:\text{sign}(u)\neq\text{sign}(p-c)} p\,\psi_1(u)+(1-p)\psi_2(u) \;>\; \inf_{u\in\mathbb{R}} p\,\psi_1(u)+(1-p)\psi_2(u) \,.$$

The following lemma gives a characterization of calibration similar to that used by Tewari and Bartlett (2007):

Lemma 2 *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$ and $\mathcal{P} \subseteq \Delta_n$. Then a surrogate loss $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ is $(\mathbf{L},\mathcal{P})$-calibrated iff there exists a function* $\text{pred}' : \mathcal{S}_\psi \to [k]$ *such that*

$$\forall \mathbf{p} \in \mathcal{P} : \quad \inf_{\mathbf{z}\in\mathcal{S}_\psi:\text{pred}'(\mathbf{z})\notin \text{argmin}_t \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \mathbf{z} \;>\; \inf_{\mathbf{z}\in\mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} \,.$$

In this paper, we will mostly be concerned with $(\mathbf{L},\Delta_n)$-calibration, which as noted above, we refer to as simply $\mathbf{L}$-calibration. The following result, whose proof is a straightforward generalization of that of a similar result for the 0-1 loss given by Tewari and Bartlett (2007), explains why $\mathbf{L}$-calibration is useful:

Theorem 3 *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$. A surrogate loss $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ is $\mathbf{L}$-calibrated iff there exists a function* $\text{pred} : \mathcal{C} \to [k]$ *such that for all distributions D on $\mathcal{X} \times [n]$ and all sequences of (vector) functions $\mathbf{f}_m : \mathcal{X} \to \mathcal{C}$,*

$$\text{er}_D^{\boldsymbol{\psi}}[\mathbf{f}_m] \longrightarrow \text{er}_D^{\boldsymbol{\psi},*} \qquad \textit{implies} \qquad \text{er}_D^{\mathbf{L}}[\text{pred} \circ \mathbf{f}_m] \longrightarrow \text{er}_D^{\mathbf{L},*} \,.$$

In particular, Theorem 3 implies that a surrogate $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated if and only if $\exists$ a mapping $\text{pred} : \mathcal{C} \to [k]$ such that any ψ-consistent algorithm learning models of the form $\mathbf{f}_m : \mathcal{X} \to \mathcal{C}$ (from i.i.d. examples $(X_1, Y_1), \ldots, (X_m, Y_m)$) yields an $\mathbf{L}$-consistent algorithm learning models of the form $(\text{pred} \circ \mathbf{f}_m) : \mathcal{X} \to [k]$.

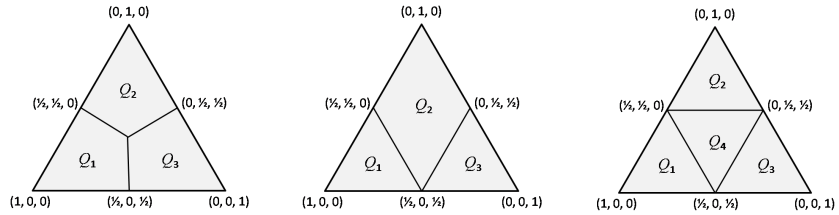


Figure 2: Trigger probability sets for (a) 0-1 loss $\mathbf{L}^{0\text{-}1}$; (b) ordinal regression loss $\mathbf{L}^{\text{ord}}$; and (c) 'abstain' loss $\mathbf{L}^{(?)}$; all for $n = 3$, for which the probability simplex can be visualized easily. Calculations of these sets can be found in Appendix B.

2.4 Trigger Probabilities and Positive Normals

Our goal is to study conditions under which a surrogate loss $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ is $\mathbf{L}$-calibrated for a target loss matrix $\mathbf{L} \in \mathbb{R}_+^{n\times k}$. To this end, we will now define certain properties of both multiclass loss matrices $\mathbf{L}$ and multiclass surrogates $\boldsymbol{\psi}$ that will be useful in relating the two. Specifically, we will define *trigger probability sets* associated with a multiclass loss matrix $\mathbf{L}$, and *positive normal sets* associated with a multiclass surrogate $\boldsymbol{\psi}$; in Section 3 we will use these to obtain both necessary and sufficient conditions for calibration.

Definition 4 (Trigger probability sets) *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$. For each $t \in [k]$, the* trigger probability set of $\mathbf{L}$ at t *is defined as*

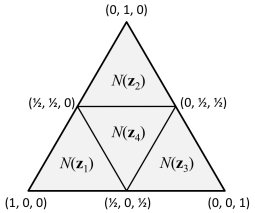
$$\mathcal{Q}_t^{\mathbf{L}} \;\triangleq\; \left\{\mathbf{p} \in \Delta_n : \mathbf{p}^\top(\boldsymbol{\ell}_t - \boldsymbol{\ell}_{t'}) \leq 0 \;\forall t' \in [k]\right\} \;=\; \left\{\mathbf{p} \in \Delta_n : t \in \text{argmin}_{t'\in[k]} \mathbf{p}^\top \boldsymbol{\ell}_{t'}\right\}.$$

In words, the trigger probability set $\mathcal{Q}_t^{\mathbf{L}}$ is the set of class probability vectors for which predicting t is optimal in terms of minimizing $\mathbf{L}$-risk. Such sets have also been studied by Lambert and Shoham (2009) and O'Brien et al. (2008) in a different context. Lambert and Shoham (2009) show that these sets form what is called a power diagram, which is a generalization of the Voronoi diagram. Trigger probability sets for the 0-1, ordinal regression, and 'abstain' loss matrices (described in Examples 1, 2 and 4) are illustrated in Figure 2; the corresponding calculations can be found in Appendix B.

Definition 5 (Positive normal sets) *Let $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$. For each point $\mathbf{z} \in \mathcal{S}_\psi$, the* positive normal set of $\boldsymbol{\psi}$ at $\mathbf{z}$ *is defined as*[2]

$$\mathcal{N}^\psi(\mathbf{z}) \;\triangleq\; \left\{\mathbf{p} \in \Delta_n : \mathbf{p}^\top(\mathbf{z} - \mathbf{z}') \leq 0 \;\forall \mathbf{z}' \in \mathcal{S}_\psi\right\} \;=\; \left\{\mathbf{p} \in \Delta_n : \mathbf{p}^\top \mathbf{z} = \inf_{\mathbf{z}'\in\mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}'\right\}.$$

2. For points $\mathbf{z}$ in the interior of $\mathcal{S}_\psi$, $\mathcal{N}^\psi(\mathbf{z})$ is empty.



$$\mathcal{N}^{\mathrm{CS}}(\mathbf{z}_1) = \{\mathbf{p} \in \Delta_3 : p_1 \geq \tfrac{1}{2}\}$$
$$\mathcal{N}^{\mathrm{CS}}(\mathbf{z}_2) = \{\mathbf{p} \in \Delta_3 : p_2 \geq \tfrac{1}{2}\}$$
$$\mathcal{N}^{\mathrm{CS}}(\mathbf{z}_3) = \{\mathbf{p} \in \Delta_3 : p_3 \geq \tfrac{1}{2}\}$$
$$\mathcal{N}^{\mathrm{CS}}(\mathbf{z}_4) = \{\mathbf{p} \in \Delta_3 : \max(p_1, p_2, p_3) \leq \tfrac{1}{2}\}$$

Figure 3: Positive normal sets for the Crammer-Singer surrogate $\boldsymbol{\psi}^{\mathrm{CS}}$ for $n = 3$, at 4 points $\mathbf{z}_i = \boldsymbol{\psi}^{\mathrm{CS}}(\mathbf{u}_i) \in \mathbb{R}^3_+$ $(i \in [4])$ for $\mathbf{u}_1 = (1,0,0)^\top$, $\mathbf{u}_2 = (0,1,0)^\top$, $\mathbf{u}_3 = (0,0,1)^\top$, and $\mathbf{u}_4 = (0,0,0)^\top$. Calculations of these sets are based on Lemma 9 and can be found in Appendix C.

For any sequence of points $\{\mathbf{z}_m\}$ *in* $\mathcal{S}_\psi$*, the* positive normal set of $\boldsymbol{\psi}$ at $\{\mathbf{z}_m\}$ *is defined as*[3]

$$\mathcal{N}^{\psi}(\{\mathbf{z}_m\}) \triangleq \left\{\mathbf{p} \in \Delta_n : \lim_{m\to\infty} \mathbf{p}^\top \mathbf{z}_m = \inf_{\mathbf{z}' \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}'\right\}.$$

In words, the positive normal set $\mathcal{N}^{\psi}(\mathbf{z})$ at a point $\mathbf{z} = \boldsymbol{\psi}(\mathbf{u}) \in \mathcal{R}_\psi$ is the set of class probability vectors for which predicting $\mathbf{u}$ is optimal in terms of minimizing ψ-risk. Such sets were also studied by Tewari and Bartlett (2007). The extension to sequences of points in $\mathcal{S}_\psi$ is needed for technical reasons in some of our proofs. Note that for $\mathcal{N}^{\psi}(\{\mathbf{z}_m\})$ to be well-defined, the sequence $\{\mathbf{z}_m\}$ need not converge itself; however if the sequence $\{\mathbf{z}_m\}$ does converge to some point $\mathbf{z} \in \mathcal{S}_\psi$, then $\mathcal{N}^{\psi}(\{\mathbf{z}_m\}) = \mathcal{N}^{\psi}(\mathbf{z})$. Positive normal sets for the Crammer-Singer surrogate (described in Example 5) at 4 points are illustrated in Figure 3; the corresponding calculations can be found in Appendix C.

3. Conditions for Calibration

In this section we give both necessary conditions (Section 3.1) and sufficient conditions (Section 3.2) for a surrogate $\boldsymbol{\psi}$ to be calibrated w.r.t. an arbitrary target loss matrix $\mathbf{L}$. Both sets of conditions involve the trigger probability sets of $\mathbf{L}$ and the positive normal sets of $\boldsymbol{\psi}$; in Section 3.3 we give a result that facilitates computation of positive normal sets for certain classes of surrogates $\boldsymbol{\psi}$.

3.1 Necessary Conditions for Calibration

We start by deriving necessary conditions for $\mathbf{L}$-calibration of a surrogate loss $\boldsymbol{\psi}$. Consider what happens if for some point $\mathbf{z} \in \mathcal{S}_\psi$, the positive normal set of $\boldsymbol{\psi}$ at $\mathbf{z}$, $\mathcal{N}^{\psi}(\mathbf{z})$, has a non-empty intersection with the interiors of two trigger probability sets of $\mathbf{L}$, say $\mathcal{Q}^{\mathbf{L}}_1$ and $\mathcal{Q}^{\mathbf{L}}_2$ (see Figure 4 for an illustration), which means $\exists \mathbf{q}_1, \mathbf{q}_2 \in \mathcal{N}^{\psi}(\mathbf{z})$ with $\operatorname{argmin}_{t\in[k]} \mathbf{q}_1^\top \boldsymbol{\ell}_t = \{1\}$ and $\operatorname{argmin}_{t\in[k]} \mathbf{q}_2^\top \boldsymbol{\ell}_t = \{2\}$. If $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated, then by Lemma 2, we have $\exists \mathrm{pred}' : \mathcal{S}_\psi \to [k]$

3. For sequences $\{\mathbf{z}_m\}$ for which $\lim_{m\to\infty} \mathbf{p}^\top \mathbf{z}_m$ does not exist for any $\mathbf{p}$, $\mathcal{N}^{\psi}(\mathbf{z})$ is empty.

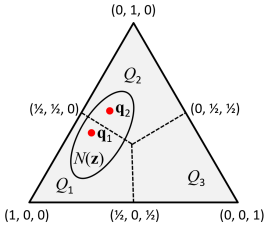


Figure 4: Visual proof of Theorem 6. If a surrogate $\boldsymbol{\psi}$ is such that its positive normal set $\mathcal{N}^{\psi}(\mathbf{z})$ at some point $\mathbf{z}$ has non-empty intersection with the interiors of two trigger probability sets (say $\mathcal{Q}^{\mathbf{L}}_1$ and $\mathcal{Q}^{\mathbf{L}}_2$) of $\mathbf{L}$, then $\boldsymbol{\psi}$ cannot be $\mathbf{L}$-calibrated.

such that

$$\inf_{\mathbf{z}' \in \mathcal{S}_\psi : \mathrm{pred}'(\mathbf{z}') \neq 1} \mathbf{q}_1^\top \mathbf{z}' = \inf_{\mathbf{z}' \in \mathcal{S}_\psi : \mathrm{pred}'(\mathbf{z}') \notin \operatorname{argmin}_t \mathbf{q}_1^\top \boldsymbol{\ell}_t} \mathbf{q}_1^\top \mathbf{z}' > \inf_{\mathbf{z}' \in \mathcal{S}_\psi} \mathbf{q}_1^\top \mathbf{z}' = \mathbf{q}_1^\top \mathbf{z}$$
$$\inf_{\mathbf{z}' \in \mathcal{S}_\psi : \mathrm{pred}'(\mathbf{z}') \neq 2} \mathbf{q}_2^\top \mathbf{z}' = \inf_{\mathbf{z}' \in \mathcal{S}_\psi : \mathrm{pred}'(\mathbf{z}') \notin \operatorname{argmin}_t \mathbf{q}_2^\top \boldsymbol{\ell}_t} \mathbf{q}_2^\top \mathbf{z}' > \inf_{\mathbf{z}' \in \mathcal{S}_\psi} \mathbf{q}_2^\top \mathbf{z}' = \mathbf{q}_2^\top \mathbf{z}.$$

The first inequality above implies $\mathrm{pred}'(\mathbf{z}) = 1$; the second inequality implies $\mathrm{pred}'(\mathbf{z}) = 2$, leading to a contradiction. This gives us the following necessary condition for $\mathbf{L}$-calibration of $\boldsymbol{\psi}$, which requires the positive normal sets of $\boldsymbol{\psi}$ at all points $\mathbf{z} \in \mathcal{S}_\psi$ to be 'well-behaved' w.r.t. $\mathbf{L}$ in the sense of being contained within individual trigger probability sets of $\mathbf{L}$ and generalizes the 'admissibility' condition used for 0-1 loss by Tewari and Bartlett (2007):

Theorem 6 *Let* $\mathbf{L} \in \mathbb{R}^{n\times k}_+$*, and let* $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+$ *be* $\mathbf{L}$*-calibrated. Then for all points* $\mathbf{z} \in \mathcal{S}_\psi$*, there exists some* $t \in [k]$ *such that* $\mathcal{N}^{\psi}(\mathbf{z}) \subseteq \mathcal{Q}^{\mathbf{L}}_t$*.*

Proof See above discussion. ■

In fact, we have the following stronger necessary condition, which requires the positive normal sets of $\boldsymbol{\psi}$ not only at all points $\mathbf{z} \in \mathcal{S}_\psi$ but also at all sequences $\{\mathbf{z}_m\}$ in $\mathcal{S}_\psi$ to be contained within individual trigger probability sets of $\mathbf{L}$:

Theorem 7 *Let* $\mathbf{L} \in \mathbb{R}^{n\times k}_+$*, and let* $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+$ *be* $\mathbf{L}$*-calibrated. Then for all sequences* $\{\mathbf{z}_m\}$ *in* $\mathcal{S}_\psi$*, there exists some* $t \in [k]$ *such that* $\mathcal{N}^{\psi}(\{\mathbf{z}_m\}) \subseteq \mathcal{Q}^{\mathbf{L}}_t$*.*

Proof Assume for the sake of contradiction that there is some sequence $\{\mathbf{z}_m\}$ in $\mathcal{S}_\psi$ for which $\mathcal{N}^{\psi}(\{\mathbf{z}_m\})$ is not contained in $\mathcal{Q}^{\mathbf{L}}_t$ for any $t \in [k]$. Then $\forall t \in [k]$, $\exists \mathbf{q}_t \in \mathcal{N}^{\psi}(\{\mathbf{z}_m\})$ such that $\mathbf{q}_t \notin \mathcal{Q}^{\mathbf{L}}_t$, i.e. such that $t \notin \operatorname{argmin}_{t'} \mathbf{q}_t^\top \boldsymbol{\ell}_{t'}$. Now, since $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated, by Lemma 25, there exists a function $\mathrm{pred}' : \mathcal{S}_\psi \to [k]$ such that for all $\mathbf{p} \in \mathcal{N}^{\psi}(\{\mathbf{z}_m\})$, we have $\mathrm{pred}'(\mathbf{z}_m) \in \arg\min_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}$ for all large enough m. In particular, for $\mathbf{p} = \mathbf{q}_t$, we get $\mathrm{pred}'(\mathbf{z}_m) \in \operatorname{argmin}_{t'} \mathbf{q}_t^\top \boldsymbol{\ell}_{t'}$ ultimately. Since this is true for each $t \in [k]$, we get $\mathrm{pred}'(\mathbf{z}_m) \in \cap_{t\in[k]} \operatorname{argmin}_{t'} \mathbf{q}_t^\top \boldsymbol{\ell}_{t'}$ ultimately. However by choice of $\mathbf{q}_t$, this intersection

is empty, thus yielding a contradiction. This completes the proof. ■

Note that Theorem 7 includes Theorem 6 as a special case, since $\mathcal{N}^{\psi}(\mathbf{z}) = \mathcal{N}^{\psi}(\{\mathbf{z}_m\})$ for the constant sequence $\mathbf{z}_m = \mathbf{z}\ \forall m$. We stated Theorem 6 separately above since it had a simple, direct proof that helps build intuition.

Example 6 (Crammer-Singer surrogate is not calibrated for 0-1 loss) *Looking at the positive normal sets of the Crammer-Singer surrogate $\boldsymbol{\psi}^{\mathrm{CS}}$ (for $n = 3$) shown in Figure 3 and the trigger probability sets of the 0-1 loss $\mathbf{L}^{0\text{-}1}$ shown in Figure 2(a), we see that $\mathcal{N}^{\mathrm{CS}}(\mathbf{z}_4)$ is not contained in any single trigger probability set of $\mathbf{L}^{0\text{-}1}$, and therefore applying Theorem 6, it is immediately clear that $\boldsymbol{\psi}^{\mathrm{CS}}$ is not $\mathbf{L}^{0\text{-}1}$-calibrated (this was also established by Tewari and Bartlett (2007) and Zhang (2004b)).*

3.2 Sufficient Condition for Calibration

We now give a sufficient condition for $\mathbf{L}$-calibration of a surrogate loss $\boldsymbol{\psi}$ that will be helpful in showing calibration of various surrogates. In particular, we show that for a surrogate loss $\boldsymbol{\psi}$ to be $\mathbf{L}$-calibrated, it is sufficient for the above property of positive normal sets of $\boldsymbol{\psi}$ being contained in trigger probability sets of $\mathbf{L}$ to hold for only a finite number of points in $\mathcal{S}_{\psi}$, as long as the corresponding positive normal sets jointly cover Δ_n:

Theorem 8 *Let $\mathbf{L} \in \mathbb{R}_+^{n \times k}$ and $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$. Suppose there exist $r \in \mathbb{Z}_+$ and $\mathbf{z}_1, \ldots, \mathbf{z}_r \in \mathcal{S}_{\psi}$ such that $\bigcup_{j=1}^{r} \mathcal{N}^{\psi}(\mathbf{z}_j) = \Delta_n$ and for each $j \in [r]$, $\exists t \in [k]$ such that $\mathcal{N}^{\psi}(\mathbf{z}_j) \subseteq \mathcal{Q}_t^{\mathbf{L}}$. Then $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated.*

Example 7 (Crammer-Singer surrogate is calibrated for $\mathbf{L}^{(?)}$ and $\mathbf{L}^{\mathrm{ord}}$ for $n = 3$) *Inspecting the positive normal sets of the Crammer-Singer surrogate $\boldsymbol{\psi}^{\mathrm{CS}}$ (for $n = 3$) in Figure 3 and the trigger probability sets of the 'abstain' loss matrix $\mathbf{L}^{(?)}$ in Figure 2(c), we see that $\mathcal{N}^{\mathrm{CS}}(\mathbf{z}_i) = \mathcal{Q}_i^{(?)}\ \forall i \in [4]$, and therefore by Theorem 8, the Crammer-Singer surrogate $\boldsymbol{\psi}^{\mathrm{CS}}$ is $\mathbf{L}^{(?)}$-calibrated. Similarly, looking at the trigger probability sets of the ordinal regression loss matrix $\mathbf{L}^{\mathrm{ord}}$ in Figure 2(b) and again applying Theorem 8, we see that the Crammer-Singer surrogate $\boldsymbol{\psi}^{\mathrm{CS}}$ is also $\mathbf{L}^{\mathrm{ord}}$-calibrated!*

Some additional examples of applications of Theorems 6 and 8 are provided in Section 3.3 below. Both the necessary and sufficient conditions above will also be used when we study the convex calibration dimension of a loss matrix $\mathbf{L}$ in Section 4.

3.3 Computation of Positive Normal Sets

Both the necessary and sufficient conditions for calibration above involve the positive normal sets $\mathcal{N}^{\psi}(\mathbf{z})$ at various points $\mathbf{z} \in \mathcal{S}_{\psi}$. Thus in order to use the above results to show that a surrogate $\boldsymbol{\psi}$ is (or is not) $\mathbf{L}$-calibrated, one needs to be able to compute or characterize the sets $\mathcal{N}^{\psi}(\mathbf{z})$. Here we give a method for computing these sets for certain surrogates $\boldsymbol{\psi}$ at certain points $\mathbf{z} \in \mathcal{S}_{\psi}$. Specifically, the following result gives an explicit method for computing $\mathcal{N}^{\psi}(\mathbf{z})$ for convex surrogate losses $\boldsymbol{\psi}$ operating on a convex surrogate space $\mathcal{C} \subseteq \mathbb{R}^d$, at points $\mathbf{z} = \boldsymbol{\psi}(\mathbf{u}) \in \mathcal{R}_{\psi}$ for which the subdifferential $\partial\psi_y(\mathbf{u})$ for each $y \in [n]$

can be described as the convex hull of a finite number of points in $\mathbb{R}^d$; this is particularly applicable for piecewise linear surrogates.[4,5]

Lemma 9 *Let $\mathcal{C} \subseteq \mathbb{R}^d$ be a convex set and let $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ be convex. Let $\mathbf{z} = \boldsymbol{\psi}(\mathbf{u})$ for some $\mathbf{u} \in \mathcal{C}$ such that $\forall y \in [n]$, the subdifferential of ψ_y at $\mathbf{u}$ can be written as*

$$\partial\psi_y(\mathbf{u}) = \mathrm{conv}\left(\{\mathbf{w}_1^y, \ldots, \mathbf{w}_{s_y}^y\}\right)$$

for some $s_y \in \mathbb{Z}_+$ and $\mathbf{w}_1^y, \ldots, \mathbf{w}_{s_y}^y \in \mathbb{R}^d$. Let $s = \sum_{y=1}^n s_y$, and let

$$\mathbf{A} = \left[\mathbf{w}_1^1 \ldots \mathbf{w}_{s_1}^1 \mathbf{w}_1^2 \ldots \mathbf{w}_{s_2}^2 \ldots\ldots \mathbf{w}_1^n \ldots \mathbf{w}_{s_n}^n\right] \in \mathbb{R}^{d \times s}; \qquad \mathbf{B} = [b_{yj}] \in \mathbb{R}^{n \times s},$$

where b_{yj} is 1 if the j-th column of $\mathbf{A}$ came from $\{\mathbf{w}_1^y, \ldots, \mathbf{w}_{s_y}^y\}$ and 0 otherwise. Then

$$\mathcal{N}^{\psi}(\mathbf{z}) = \left\{\mathbf{p} \in \Delta_n : \mathbf{p} = \mathbf{B}\mathbf{q} \text{ for some } \mathbf{q} \in \mathrm{null}(\mathbf{A}) \cap \Delta_s\right\},$$

where $\mathrm{null}(\mathbf{A}) \subseteq \mathbb{R}^s$ denotes the null space of the matrix $\mathbf{A}$.

The proof makes use of the fact that a convex function $\phi : \mathbb{R}^d \to \mathbb{R}$ attains its minimum at $\mathbf{u}_0 \in \mathbb{R}^d$ iff the subdifferential $\partial\phi(\mathbf{u}_0)$ contains $\mathbf{0} \in \mathbb{R}^d$ (e.g. see Bertsekas et al. (2003)). We will also make use of the fact that if $\phi_1, \phi_2 : \mathbb{R}^d \to \mathbb{R}$ are convex functions, then the subdifferential of their sum $\phi_1 + \phi_2$ at $\mathbf{u}_0$ is is equal to the Minkowski sum of the subdifferentials of ϕ_1 and ϕ_2 at $\mathbf{u}_0$:

$$\partial(\phi_1 + \phi_2)(\mathbf{u}_0) \;=\; \left\{\mathbf{w}_1 + \mathbf{w}_2 : \mathbf{w}_1 \in \partial\phi_1(\mathbf{u}_0), \mathbf{w}_2 \in \partial\phi_2(\mathbf{u}_0)\right\}.$$

Proof (Proof of Lemma 9)

We have for all $\mathbf{p} \in \mathbb{R}^n$,

$$\begin{aligned}
\mathbf{p} \in \mathcal{N}^{\psi}(\boldsymbol{\psi}(\mathbf{u})) &\iff \mathbf{p} \in \Delta_n,\ \mathbf{p}^{\top}\boldsymbol{\psi}(\mathbf{u}) \leq \mathbf{p}^{\top}\mathbf{z}'\ \forall \mathbf{z}' \in \mathcal{S}_{\psi} \\
&\iff \mathbf{p} \in \Delta_n,\ \mathbf{p}^{\top}\boldsymbol{\psi}(\mathbf{u}) \leq \mathbf{p}^{\top}\mathbf{z}'\ \forall \mathbf{z}' \in \mathcal{R}_{\psi} \\
&\iff \mathbf{p} \in \Delta_n,\ \text{and the convex function } \phi(\mathbf{u}') = \mathbf{p}^{\top}\boldsymbol{\psi}(\mathbf{u}') = \textstyle\sum_{y=1}^n p_y \psi_y(\mathbf{u}') \\
&\qquad \text{achieves its minimum at } \mathbf{u}' = \mathbf{u} \\
&\iff \mathbf{p} \in \Delta_n,\ \mathbf{0} \in \sum_{y=1}^{n} p_y \partial\psi_y(\mathbf{u}) \\
&\iff \mathbf{p} \in \Delta_n,\ \mathbf{0} = \sum_{y=1}^{n} p_y \sum_{j=1}^{s_y} v_j^y \mathbf{w}_j^y \text{ for some } \mathbf{v}^y \in \Delta_{s_y} \\
&\iff \mathbf{p} \in \Delta_n,\ \mathbf{0} = \sum_{y=1}^{n} \sum_{j=1}^{s_y} q_j^y \mathbf{w}_j^y \text{ for some } \mathbf{q}^y = p_y \mathbf{v}^y, \quad \mathbf{v}^y \in \Delta_{s_y} \\
&\iff \mathbf{p} \in \Delta_n, \mathbf{A}\mathbf{q} = \mathbf{0} \text{ for some } \mathbf{q} = (p_1\mathbf{v}^1, \ldots, p_n\mathbf{v}^n)^{\top} \in \Delta_s, \quad \mathbf{v}^y \in \Delta_{s_y} \\
&\iff \mathbf{p} = \mathbf{B}\mathbf{q} \text{ for some } \mathbf{q} \in \mathrm{null}(\mathbf{A}) \cap \Delta_s\,.
\end{aligned}$$

4. Recall that a vector function is convex if all its component functions are convex.
5. Recall that the subdifferential of a convex function $\phi : \mathbb{R}^d \to \overline{\mathbb{R}}$ at a point $\mathbf{u}_0 \in \mathbb{R}^d$ is defined as $\partial\phi(\mathbf{u}_0) = \left\{\mathbf{w} \in \mathbb{R}^d : \phi(\mathbf{u}) - \phi(\mathbf{u}_0) \geq \mathbf{w}^{\top}(\mathbf{u} - \mathbf{u}_0)\ \forall \mathbf{u} \in \mathbb{R}^d\right\}$ and is a convex set in $\mathbb{R}^d$ (e.g. see Bertsekas et al. (2003)).

■

We now give examples of computation of positive normal sets using Lemma 9 for two convex surrogates, both of which operate on the one-dimensional surrogate space $\mathcal{C} = \mathbb{R}$, and as we shall see, turn out to be calibrated w.r.t. the ordinal regression loss $\mathbf{L}^{\text{ord}}$ but not w.r.t. the 0-1 loss $\mathbf{L}^{\text{0-1}}$ or the 'abstain' loss $\mathbf{L}^{(?)}$. As another example of an application of Lemma 9, calculations showing computation of positive normal sets of the Crammer-Singer surrogate (as shown in Figure 3) are given in the appendix.

Example 8 (Positive normal sets of 'absolute' surrogate) *Let $n = 3$, and let $\mathcal{C} = \mathbb{R}$. Consider the 'absolute' surrogate $\boldsymbol{\psi}^{\text{abs}} : \mathbb{R} \to \mathbb{R}^3_+$ defined as follows:*

$$\psi^{\text{abs}}_y(u) = |u - y| \qquad \forall y \in [3],\, u \in \mathbb{R}\,. \tag{10}$$

Clearly, $\boldsymbol{\psi}^{\text{abs}}$ is a convex function (see Figure 5). Moreover, we have

$$\mathcal{R}_{\text{abs}} \;=\; \boldsymbol{\psi}^{\text{abs}}(\mathbb{R}) \;=\; \left\{(|u-1|, |u-2|, |u-3|)^\top : u \in \mathbb{R}\right\} \subset \mathbb{R}^3_+\,.$$

Now let $u_1 = 1$, $u_2 = 2$, and $u_3 = 3$, and let

$$\begin{array}{rcccl}
\mathbf{z}_1 &=& \boldsymbol{\psi}^{\text{abs}}(u_1) &=& \boldsymbol{\psi}^{\text{abs}}(1) \;=\; (0,1,2)^\top \in \mathcal{R}_{\text{abs}} \\
\mathbf{z}_2 &=& \boldsymbol{\psi}^{\text{abs}}(u_2) &=& \boldsymbol{\psi}^{\text{abs}}(2) \;=\; (1,0,1)^\top \in \mathcal{R}_{\text{abs}} \\
\mathbf{z}_3 &=& \boldsymbol{\psi}^{\text{abs}}(u_3) &=& \boldsymbol{\psi}^{\text{abs}}(3) \;=\; (2,1,0)^\top \in \mathcal{R}_{\text{abs}}\,.
\end{array}$$

Let us consider computing the positive normal sets of $\boldsymbol{\psi}^{\text{abs}}$ at the 3 points $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3$ above. To see that $\mathbf{z}_1$ satisfies the conditions of Lemma 9, note that

$$\begin{array}{rcccl}
\partial\psi^{\text{abs}}_1(u_1) &=& \partial\psi^{\text{abs}}_1(1) &=& [-1,1] = \text{conv}(\{+1,-1\})\,; \\
\partial\psi^{\text{abs}}_2(u_1) &=& \partial\psi^{\text{abs}}_2(1) &=& \{-1\} = \text{conv}(\{-1\})\,; \\
\partial\psi^{\text{abs}}_3(u_1) &=& \partial\psi^{\text{abs}}_3(1) &=& \{-1\} = \text{conv}(\{-1\})\,.
\end{array}$$

Therefore, we can use Lemma 9 to compute $\mathcal{N}^{\text{abs}}(\mathbf{z}_1)$. Here $s = 4$, and

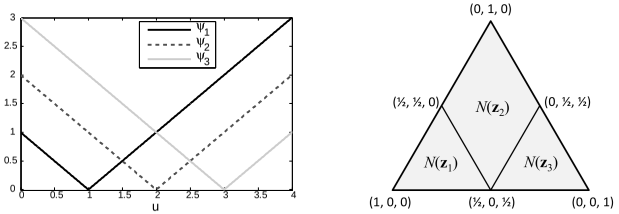
$$\mathbf{A} = \begin{bmatrix} +1 & -1 & -1 & -1 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

This gives

$$\begin{array}{rcl}
\mathcal{N}^{\text{abs}}(\mathbf{z}_1) &=& \left\{\mathbf{p} \in \Delta_3 : \mathbf{p} = (q_1 + q_2, q_3, q_4) \text{ for some } \mathbf{q} \in \Delta_4,\; q_1 - q_2 - q_3 - q_4 = 0\right\} \\
&=& \left\{\mathbf{p} \in \Delta_3 : \mathbf{p} = (q_1 + q_2, q_3, q_4) \text{ for some } \mathbf{q} \in \Delta_4,\; q_1 = \tfrac{1}{2}\right\} \\
&=& \left\{\mathbf{p} \in \Delta_3 : p_1 \geq \tfrac{1}{2}\right\}.
\end{array}$$

It is easy to see that $\mathbf{z}_2$ and $\mathbf{z}_3$ also satisfy the conditions of Lemma 9; similar computations then yield

$$\begin{array}{rcl}
\mathcal{N}^{\text{abs}}(\mathbf{z}_2) &=& \left\{\mathbf{p} \in \Delta_3 : p_1 \leq \tfrac{1}{2}, p_3 \leq \tfrac{1}{2}\right\} \\
\mathcal{N}^{\text{abs}}(\mathbf{z}_3) &=& \left\{\mathbf{p} \in \Delta_3 : p_3 \geq \tfrac{1}{2}\right\}.
\end{array}$$



(a) (b)

Figure 5: (a) The 'absolute' surrogate $\boldsymbol{\psi}^{\text{abs}} : \mathbb{R} \to \mathbb{R}^3_+$ (for $n = 3$), and (b) its positive normal sets at 3 points $\mathbf{z}_i = \boldsymbol{\psi}^{\text{abs}}(u_i) \in \mathbb{R}^3_+$ ($i \in [3]$) for $u_1 = 1, u_2 = 2, u_3 = 3$. See Example 8 for details.

The positive normal sets above are shown in Figure 5. Comparing these with the trigger probability sets in Figure 2, we have by Theorem 8 that $\boldsymbol{\psi}^{\text{abs}}$ is $\mathbf{L}^{\text{ord}}$-calibrated, and by Theorem 6 that $\boldsymbol{\psi}^{\text{abs}}$ is not calibrated w.r.t. $\mathbf{L}^{\text{0-1}}$ or $\mathbf{L}^{(?)}$.

Example 9 (Positive normal sets of 'ϵ-insensitive' surrogate) *Let $n = 3$, and let $\mathcal{C} = \mathbb{R}$. Let $\epsilon \in [0, 0.5)$, and consider the 'ϵ-insensitive' surrogate $\boldsymbol{\psi}^{\epsilon} : \mathbb{R} \to \mathbb{R}^3_+$ defined as follows:*

$$\psi^{\epsilon}_y(u) = \left(|u - y| - \epsilon\right)_+ \qquad \forall y \in [3],\, u \in \mathbb{R}\,. \tag{11}$$

For $\epsilon = 0$, we have $\boldsymbol{\psi}^{\epsilon} = \boldsymbol{\psi}^{\text{abs}}$. Clearly, $\boldsymbol{\psi}^{\epsilon}$ is a convex function (see Figure 6). Moreover, we have

$$\mathcal{R}_{\epsilon} \;=\; \boldsymbol{\psi}^{\epsilon}(\mathbb{R}) \;=\; \left\{\left((|u-1| - \epsilon)_+, (|u-2| - \epsilon)_+, (|u-3| - \epsilon)_+\right)^\top : u \in \mathbb{R}\right\} \subset \mathbb{R}^3_+\,.$$

For concreteness, we will take $\epsilon = 0.25$ below, but similar computations hold $\forall \epsilon \in (0, 0.5)$. Let $u_1 = 1 + \epsilon = 1.25$, $u_2 = 2 - \epsilon = 1.75$, $u_3 = 2 + \epsilon = 2.25$, and $u_4 = 3 - \epsilon = 2.75$, and let

$$\begin{array}{rcccl}
\mathbf{z}_1 &=& \boldsymbol{\psi}^{0.25}(u_1) &=& \boldsymbol{\psi}^{0.25}(1.25) \;=\; (0, 0.5, 1.5)^\top \in \mathcal{R}_{0.25} \\
\mathbf{z}_2 &=& \boldsymbol{\psi}^{0.25}(u_2) &=& \boldsymbol{\psi}^{0.25}(1.75) \;=\; (0.5, 0, 1)^\top \in \mathcal{R}_{0.25} \\
\mathbf{z}_3 &=& \boldsymbol{\psi}^{0.25}(u_3) &=& \boldsymbol{\psi}^{0.25}(2.25) \;=\; (1, 0, 0.5)^\top \in \mathcal{R}_{0.25} \\
\mathbf{z}_4 &=& \boldsymbol{\psi}^{0.25}(u_4) &=& \boldsymbol{\psi}^{0.25}(2.75) \;=\; (1.5, 0.5, 0)^\top \in \mathcal{R}_{0.25}\,.
\end{array}$$

Let us consider computing the positive normal sets of $\boldsymbol{\psi}^{0.25}$ at the 4 points $\mathbf{z}_i$ ($i \in [4]$) above. To see that $\mathbf{z}_1$ satisfies the conditions of Lemma 9, note that

$$\begin{array}{rcccl}
\partial\psi^{0.25}_1(u_1) &=& \partial\psi^{0.25}_1(1.25) &=& [0,1] = \text{conv}(\{0,1\})\,; \\
\partial\psi^{0.25}_2(u_1) &=& \partial\psi^{0.25}_2(1.25) &=& \{-1\} = \text{conv}(\{-1\})\,; \\
\partial\psi^{0.25}_3(u_1) &=& \partial\psi^{0.25}_3(1.25) &=& \{-1\} = \text{conv}(\{-1\})\,.
\end{array}$$

Therefore, we can use Lemma 9 to compute $\mathcal{N}^{0.25}(\mathbf{z}_1)$. *Here* $s = 4$, *and*

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & -1 & -1 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

This gives

$$\begin{aligned} \mathcal{N}^{0.25}(\mathbf{z}_1) &= \left\{\mathbf{p} \in \Delta_3 : \mathbf{p} = (q_1 + q_2, q_3, q_4) \text{ for some } \mathbf{q} \in \Delta_4,\ q_2 - q_3 - q_4 = 0\right\} \\ &= \left\{\mathbf{p} \in \Delta_3 : \mathbf{p} = (q_1 + q_2, q_3, q_4) \text{ for some } \mathbf{q} \in \Delta_4,\ q_1 + q_2 \geq q_3 + q_4\right\} \\ &= \left\{\mathbf{p} \in \Delta_3 : p_1 \geq \tfrac{1}{2}\right\}. \end{aligned}$$

Similarly, to see that $\mathbf{z}_2$ *satisfies the conditions of Lemma 9, note that*

$$\begin{aligned} \partial\psi_1^{0.25}(u_2) &= \partial\psi_1^{0.25}(1.75) = \{1\} = \text{conv}(\{1\}); \\ \partial\psi_2^{0.25}(u_2) &= \partial\psi_2^{0.25}(1.75) = [-1, 0] = \text{conv}(\{-1, 0\}); \\ \partial\psi_3^{0.25}(u_2) &= \partial\psi_3^{0.25}(1.75) = \{-1\} = \text{conv}(\{-1\}). \end{aligned}$$

Again, we can use Lemma 9 to compute $\mathcal{N}^{0.25}(\mathbf{z}_2)$*; here* $s = 4$, *and*

$$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 & -1 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

This gives

$$\begin{aligned} \mathcal{N}^{0.25}(\mathbf{z}_2) &= \left\{\mathbf{p} \in \Delta_3 : \mathbf{p} = (q_1, q_2 + q_3, q_4) \text{ for some } \mathbf{q} \in \Delta_4,\ q_1 - q_2 - q_4 = 0\right\} \\ &= \left\{\mathbf{p} \in \Delta_3 : p_1 \geq p_3,\ p_1 \leq \tfrac{1}{2}\right\}. \end{aligned}$$

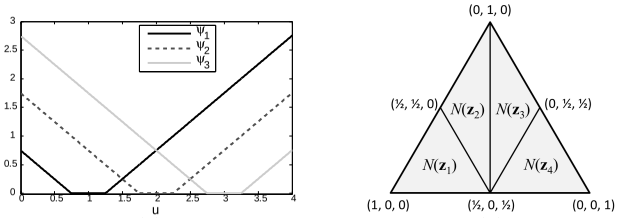
It is easy to see that $\mathbf{z}_3$ *and* $\mathbf{z}_4$ *also satisfy the conditions of Lemma 9; similar computations then yield*

$$\begin{aligned} \mathcal{N}^{0.25}(\mathbf{z}_3) &= \left\{\mathbf{p} \in \Delta_3 : p_1 \leq p_3,\ p_3 \leq \tfrac{1}{2}\right\} \\ \mathcal{N}^{0.25}(\mathbf{z}_4) &= \left\{\mathbf{p} \in \Delta_3 : p_3 \geq \tfrac{1}{2}\right\}. \end{aligned}$$

The positive normal sets above are shown in Figure 6. Comparing these with the trigger probability sets in Figure 2, we have by Theorem 8 that $\boldsymbol{\psi}^{0.25}$ *is* $\mathbf{L}^{\text{ord}}$*-calibrated, and by Theorem 6 that* $\boldsymbol{\psi}^{0.25}$ *is not calibrated w.r.t.* $\mathbf{L}^{0\text{-}1}$ *or* $\mathbf{L}^{(?)}$.

4. Convex Calibration Dimension

We now turn to the study of a fundamental quantity associated with the property of $\mathbf{L}$-calibration. Specifically, in Examples 6 and 7 above, we saw that to develop a surrogate calibrated w.r.t. to the ordinal regression loss $\mathbf{L}^{\text{ord}}$ for $n = 3$, it was sufficient to consider a surrogate prediction space $\mathcal{C} = \mathbb{R}$, with dimension $d = 1$; in addition, the surrogates we considered were convex, and can therefore be used in developing computationally efficient



(a) (b)

Figure 6: (a) The 'ϵ-insensitive' surrogate $\boldsymbol{\psi}^{\epsilon} : \mathbb{R} \to \mathbb{R}^3_+$ for $\epsilon = 0.25$ (and $n = 3$), and (b) its positive normal sets at 4 points $\mathbf{z}_i = \boldsymbol{\psi}^{\epsilon}(u_i) \in \mathbb{R}^3_+$ $(i \in [4])$ for $u_1 = 1.25, u_2 = 1.75, u_3 = 2.25, u_4 = 2.75$. See Example 9 for details.

algorithms. In fact the same surrogate prediction space with $d = 1$ can be used to develop similar convex surrogate losses calibrated w.r.t. the $\mathbf{L}^{\text{ord}}$ for *any* $n \in \mathbb{Z}_+$. However not all multiclass loss matrices $\mathbf{L}$ have such 'low-dimensional' convex surrogates. This raises the natural question of what is the smallest dimension d that supports a convex $\mathbf{L}$-calibrated surrogate for a given multiclass loss $\mathbf{L}$, and leads us to the following definition:

Definition 10 (Convex calibration dimension) *Let* $\mathbf{L} \in \mathbb{R}^{n \times k}_+$. *Define the* convex calibration dimension (CC dimension) *of* $\mathbf{L}$ *as*

$$\text{CCdim}(\mathbf{L}) \triangleq \min\left\{d \in \mathbb{Z}_+ : \exists \text{ a convex set } \mathcal{C} \subseteq \mathbb{R}^d \text{ and a convex surrogate } \boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+ \text{ that is } \mathbf{L}\text{-calibrated}\right\},$$

if the above set is non-empty, and $\text{CCdim}(\mathbf{L}) = \infty$ *otherwise.*

The CC-dimension of a loss matrix $\mathbf{L}$ provides an important measure of the 'complexity' of designing convex calibrated surrogates for $\mathbf{L}$. Indeed, while the computational complexity of minimizing a surrogate loss, as well as that of converting surrogate predictions into target predictions, can depend on factors other than the dimension d of the surrogate space $\mathcal{C} \subseteq \mathbb{R}^d$, in the absence of other guiding factors, one would in general prefer to use a surrogate in a lower dimension d since this involves learning a smaller number of real-valued functions.

From the above discussion, $\text{CCdim}(\mathbf{L}^{\text{ord}}) = 1$ for all n. In the following, we will be interested in developing an understanding of the CC dimension for general loss matrices $\mathbf{L}$, and in particular in deriving upper and lower bounds on this quantity.

4.1 Upper Bounds on the Convex Calibration Dimension

We start with a simple result that establishes that the CC dimension of any multiclass loss matrix $\mathbf{L}$ is finite, and in fact is strictly smaller than the number of class labels n.

Lemma 11 *Let $\mathbf{L} \in \mathbb{R}_+^{n \times k}$. Let $\mathcal{C} = \left\{\mathbf{u} \in \mathbb{R}_+^{n-1} : \sum_{j=1}^{n-1} u_j \leq 1\right\}$, and for each $y \in [n]$, let $\psi_y : \mathcal{C} \to \mathbb{R}_+$ be given by*

$$\psi_y(\mathbf{u}) \;=\; \mathbf{1}(y \neq n)\,(u_y - 1)^2 + \sum_{j \in [n-1], j \neq y} {u_j}^2 \,.$$

Then $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated. In particular, since $\boldsymbol{\psi}$ is convex, $\text{CCdim}(\mathbf{L}) \leq n-1$.

It may appear surprising that the convex surrogate $\boldsymbol{\psi}$ in the above lemma, operating on a surrogate space $\mathcal{C} \subset \mathbb{R}^{n-1}$, is $\mathbf{L}$-calibrated for *all* multiclass losses $\mathbf{L}$ on n classes. However this makes intuitive sense, since in principle, for any multiclass problem, if one can estimate the conditional probabilities of the n classes accurately (which requires estimating $n-1$ real-valued functions on $\mathcal{X}$), then one can predict a target label that minimizes the expected loss according to these probabilities. Minimizing the above surrogate effectively corresponds to such class probability estimation. Indeed, the above lemma can be shown to hold for any surrogate that is a strictly proper composite multiclass loss (Vernet et al., 2011).

In practice, when the number of class labels n is large (such as in a sequence labeling task, where n is exponential in the length of the input sequence), the above result is not very helpful; in such cases, it is of interest to develop algorithms operating on a surrogate prediction space in a lower-dimensional space. Next we give a different upper bound on the CC dimension that depends on the loss $\mathbf{L}$, and for certain losses, can be significantly tighter than the general bound above.

Theorem 12 *Let $\mathbf{L} \in \mathbb{R}_+^{n \times k}$. Then* $\text{CCdim}(\mathbf{L}) \leq \text{affdim}(\mathbf{L})$.

Proof Let $\text{affdim}(\mathbf{L}) = d$. We will construct a convex $\mathbf{L}$-calibrated surrogate loss $\boldsymbol{\psi}$ with surrogate prediction space $\mathcal{C} \subseteq \mathbb{R}^d$.

Let $\mathcal{V} \subseteq \mathbb{R}^n$ denote the (d-dimensional) subspace parallel to the affine hull of the column vectors of $\mathbf{L}$, and let $\mathbf{r} \in \mathbb{R}^n$ be the corresponding translation vector, so that $\mathcal{V} = \text{aff}(\{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_k\}) + \mathbf{r}$. Let $\mathbf{v}_1, \ldots, \mathbf{v}_d \in \mathcal{V}$ be d linearly independent vectors in $\mathcal{V}$. Let $\{\mathbf{e}_1, \ldots, \mathbf{e}_d\}$ denote the standard basis in $\mathbb{R}^d$, and define a linear function $\widetilde{\boldsymbol{\psi}} : \mathbb{R}^d \to \mathbb{R}^n$ by

$$\widetilde{\boldsymbol{\psi}}(\mathbf{e}_j) = \mathbf{v}_j \quad \forall j \in [d] \,.$$

Then for each $\mathbf{v} \in \mathcal{V}$, there exists a unique vector $\mathbf{u} \in \mathbb{R}^d$ such that $\widetilde{\boldsymbol{\psi}}(\mathbf{u}) = \mathbf{v}$. In particular, since $\boldsymbol{\ell}_t + \mathbf{r} \in \mathcal{V}\ \forall t \in [k]$, there exist unique vectors $\mathbf{u}_1, \ldots, \mathbf{u}_k \in \mathbb{R}^d$ such that for each $t \in [k]$, $\widetilde{\boldsymbol{\psi}}(\mathbf{u}_t) = \boldsymbol{\ell}_t + \mathbf{r}$. Let $\mathcal{C} = \text{conv}(\{\mathbf{u}_1, \ldots, \mathbf{u}_k\}) \subseteq \mathbb{R}^d$, and define $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ as

$$\boldsymbol{\psi}(\mathbf{u}) = \widetilde{\boldsymbol{\psi}}(\mathbf{u}) - \mathbf{r} \quad \forall \mathbf{u} \in \mathcal{C} \,.$$

To see that $\boldsymbol{\psi}(\mathbf{u}) \in \mathbb{R}_+^n\ \forall \mathbf{u} \in \mathcal{C}$, note that for any $\mathbf{u} \in \mathcal{C}$, $\exists \boldsymbol{\alpha} \in \Delta_k$ such that $\mathbf{u} = \sum_{t=1}^k \alpha_t \mathbf{u}_t$, which gives $\boldsymbol{\psi}(\mathbf{u}) = \widetilde{\boldsymbol{\psi}}(\mathbf{u}) - \mathbf{r} = \left(\sum_{t=1}^k \alpha_t \widetilde{\boldsymbol{\psi}}(\mathbf{u}_t)\right) - \mathbf{r} = \left(\sum_{t=1}^k \alpha_t (\boldsymbol{\ell}_t + \mathbf{r})\right) - \mathbf{r} = \sum_{t=1}^k \alpha_t \boldsymbol{\ell}_t$ (and $\boldsymbol{\ell}_t \in \mathbb{R}_+^n\ \forall t \in [k]$). The function $\boldsymbol{\psi}$ is clearly convex. To show $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated, we will use Theorem 8. Specifically, consider the k points $\mathbf{z}_t = \boldsymbol{\psi}(\mathbf{u}_t) = \boldsymbol{\ell}_t \in \mathcal{R}_\psi$ for $t \in [k]$. By definition of $\boldsymbol{\psi}$, we have $\mathcal{S}_\psi = \text{conv}(\boldsymbol{\psi}(\mathcal{C})) = \text{conv}(\{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_k\})$; from the definitions of positive normals and trigger probabilities, it then follows that $\mathcal{N}^\psi(\mathbf{z}_t) = \mathcal{N}^\psi(\boldsymbol{\ell}_t) = \mathcal{Q}_t^{\mathbf{L}}$ for all $t \in [k]$. Thus by Theorem 8, $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated. ■

Since $\text{affdim}(\mathbf{L})$ is equal to either $\text{rank}(\mathbf{L})$ or $\text{rank}(\mathbf{L}) - 1$, this immediately gives us the following corollary:

Corollary 13 *Let $\mathbf{L} \in \mathbb{R}_+^{n \times k}$. Then* $\text{CCdim}(\mathbf{L}) \leq \text{rank}(\mathbf{L})$.

Proof Follows immediately from Theorem 12 and the fact that $\text{affdim}(\mathbf{L}) \leq \text{rank}(\mathbf{L})$. ■

Example 10 (CC dimension of Hamming loss) *Let $n = 2^r$ for some $r \in \mathbb{Z}_+$, and consider the Hamming loss $\mathbf{L}^{\text{Ham}} \in \mathbb{R}_+^{n \times n}$ defined in Example 3. As in Example 3, for each $z \in \{0, \ldots, 2^r - 1\}$, let $z_i \in \{0, 1\}$ denote the i-th bit in the r-bit binary representation of z. For each $y \in [n]$, define $\boldsymbol{\sigma}_y \in \{\pm 1\}^r$ as*

$$\sigma_{yi} \;=\; 2\,(y-1)_i - 1 \;=\; \begin{cases} +1 & \text{if } (y-1)_i = 1 \\ -1 & \text{otherwise.} \end{cases}$$

Then we have

$$\begin{aligned} \ell_{yt}^{\text{Ham}} &= \sum_{i=1}^{r} \mathbf{1}\big((y-1)_i \neq (t-1)_i\big) \\ &= \sum_{i=1}^{r} \left(\frac{1 - \sigma_{yi}\sigma_{ti}}{2}\right) \\ &= \frac{r}{2} - \sum_{i=1}^{r} \frac{\sigma_{yi}\sigma_{ti}}{2} \qquad \forall y, t \in [n] \,. \end{aligned}$$

Thus $\text{affdim}(\mathbf{L}^{\text{Ham}}) \leq r$, and therefore by Theorem 12, we have $\text{CCdim}(\mathbf{L}^{\text{Ham}}) \leq r$. This is a significantly tighter upper bound than the bound of $2^r - 1$ given by Lemma 11.

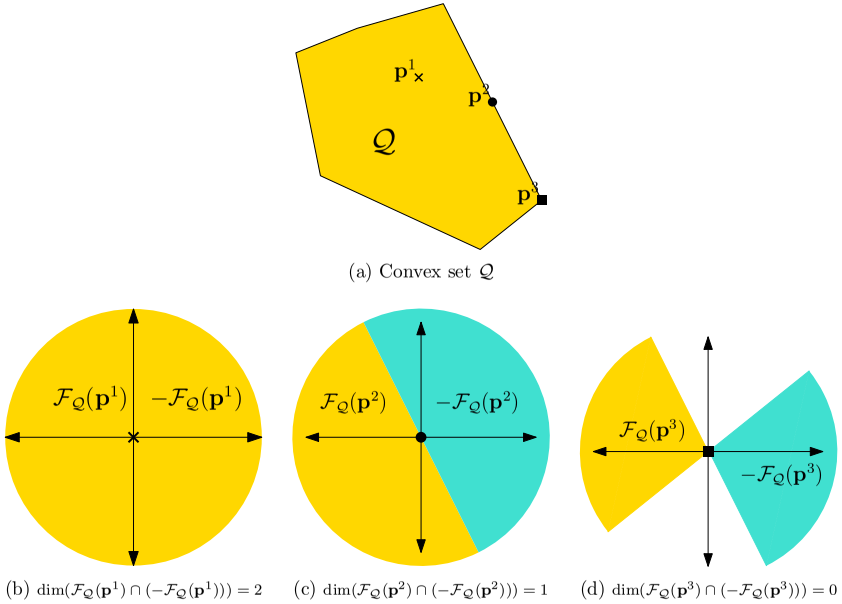
4.2 Lower Bound on the Convex Calibration Dimension

In this section we give a lower bound on the CC dimension of a loss matrix $\mathbf{L}$ and illustrate it by using it to calculate the CC dimension of the 0-1 loss. In Section 5 we will explore applications of the lower bound to obtaining impossibility results on the existence of convex calibrated surrogates in low-dimensional surrogate spaces for certain types of subset ranking losses. We will need the following definition:

Definition 14 (Feasible subspace dimension) *The* feasible subspace dimension *of a convex set $\mathcal{Q} \subseteq \mathbb{R}^n$ at a point $\mathbf{p} \in \mathcal{Q}$, denoted by $\mu_{\mathcal{Q}}(\mathbf{p})$, is defined as the dimension of the subspace $\mathcal{F}_{\mathcal{Q}}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}))$, where $\mathcal{F}_{\mathcal{Q}}(\mathbf{p})$ is the cone of feasible directions of $\mathcal{Q}$ at $\mathbf{p}$.*[6]

In essence, the feasible subspace dimension of a convex set Q at a point $\mathbf{p} \in \mathcal{Q}$ is simply the dimension of the smallest face of $\mathcal{Q}$ containing $\mathbf{p}$; see Figure 7 for an illustration.

6. For a set $\mathcal{Q} \subseteq \mathbb{R}^n$ and point $\mathbf{p} \in \mathcal{Q}$, the cone of feasible directions of $\mathcal{Q}$ at $\mathbf{p}$ is defined as $\mathcal{F}_{\mathcal{Q}}(\mathbf{p}) = \{\mathbf{v} \in \mathbb{R}^n :\ \exists \epsilon_0 > 0 \text{ such that } \mathbf{p} + \epsilon\mathbf{v} \in \mathcal{Q}\ \forall \epsilon \in (0, \epsilon_0)\}$.



(a) Convex set $\mathcal{Q}$

(b) $\dim(\mathcal{F}_{\mathcal{Q}}(\mathbf{p}^1) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}^1))) = 2$ (c) $\dim(\mathcal{F}_{\mathcal{Q}}(\mathbf{p}^2) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}^2))) = 1$ (d) $\dim(\mathcal{F}_{\mathcal{Q}}(\mathbf{p}^3) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}^3))) = 0$

Figure 7: Illustration of feasible subspace dimension $\mu_{\mathcal{Q}}(\mathbf{p})$ of a 2-dimensional convex set $\mathcal{Q}$ at three points $\mathbf{p} = \mathbf{p}^1, \mathbf{p}^2, \mathbf{p}^3$. Here $\mu_{\mathcal{Q}}(\mathbf{p}^1) = 2$, $\mu_{\mathcal{Q}}(\mathbf{p}^2) = 1$, and $\mu_{\mathcal{Q}}(\mathbf{p}^3) = 0$.

Both the proof of the lower bound we will provide below and its applications make use of the following lemma, which gives a method to calculate the feasible subspace dimension for certain convex sets $\mathcal{Q}$ and points $\mathbf{p} \in \mathcal{Q}$:

Lemma 15 *Let* $\mathcal{Q} = \left\{\mathbf{q} \in \mathbb{R}^n : \mathbf{A}^1\mathbf{q} \leq \mathbf{b}^1, \mathbf{A}^2\mathbf{q} \leq \mathbf{b}^2, \mathbf{A}^3\mathbf{q} = \mathbf{b}^3\right\}$. *Let* $\mathbf{p} \in \mathcal{Q}$ *be such that* $\mathbf{A}^1\mathbf{p} = \mathbf{b}^1$, $\mathbf{A}^2\mathbf{p} < \mathbf{b}^2$. *Then* $\mu_{\mathcal{Q}}(\mathbf{p}) = \text{nullity}\left(\begin{bmatrix}\mathbf{A}^1\\ \mathbf{A}^3\end{bmatrix}\right)$.

Proof We will show that $\mathcal{F}_{\mathcal{Q}}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p})) = \text{null}\left(\begin{bmatrix}\mathbf{A}^1\\ \mathbf{A}^3\end{bmatrix}\right)$, from which the lemma follows.

First, let $\mathbf{v} \in \text{null}\left(\begin{bmatrix}\mathbf{A}^1\\ \mathbf{A}^3\end{bmatrix}\right)$. Then for $\epsilon > 0$, we have

$$\begin{aligned}
\mathbf{A}^1(\mathbf{p} + \epsilon\mathbf{v}) &= \mathbf{A}^1\mathbf{p} + \epsilon\mathbf{A}^1\mathbf{v} = \mathbf{A}^1\mathbf{p} + \mathbf{0} = \mathbf{b}^1\\
\mathbf{A}^2(\mathbf{p} + \epsilon\mathbf{v}) &< \mathbf{b}^2 \quad \text{for small enough } \epsilon, \text{ since } \mathbf{A}^2\mathbf{p} < \mathbf{b}^2\\
\mathbf{A}^3(\mathbf{p} + \epsilon\mathbf{v}) &= \mathbf{A}^3\mathbf{p} + \epsilon\mathbf{A}^3\mathbf{v} = \mathbf{A}^3\mathbf{p} + \mathbf{0} = \mathbf{b}^3\,.
\end{aligned}$$

Thus $\mathbf{v} \in \mathcal{F}_{\mathcal{Q}}(\mathbf{p})$. Similarly, we can show $-\mathbf{v} \in \mathcal{F}_{\mathcal{Q}}(\mathbf{p})$. Thus $\mathbf{v} \in \mathcal{F}_{\mathcal{Q}}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}))$, giving $\text{null}\left(\begin{bmatrix}\mathbf{A}^1\\ \mathbf{A}^3\end{bmatrix}\right) \subseteq \mathcal{F}_{\mathcal{Q}}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}))$. Now let $\mathbf{v} \in \mathcal{F}_{\mathcal{Q}}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p}))$. Then for small enough $\epsilon > 0$, we have both $\mathbf{A}^1(\mathbf{p} + \epsilon\mathbf{v}) \leq \mathbf{b}^1$ and $\mathbf{A}^1(\mathbf{p} - \epsilon\mathbf{v}) \leq \mathbf{b}^1$. Since $\mathbf{A}^1\mathbf{p} = \mathbf{b}^1$, this gives $\mathbf{A}^1\mathbf{v} = \mathbf{0}$. Similarly, for small enough $\epsilon > 0$, we have $\mathbf{A}^3(\mathbf{p} + \epsilon\mathbf{v}) = \mathbf{b}^3$; since $\mathbf{A}^3\mathbf{p} = \mathbf{b}^3$, this gives $\mathbf{A}^3\mathbf{v} = \mathbf{0}$. Thus $\begin{bmatrix}\mathbf{A}^1\\ \mathbf{A}^3\end{bmatrix}\mathbf{v} = \mathbf{0}$, giving $\mathcal{F}_{\mathcal{Q}}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{Q}}(\mathbf{p})) \subseteq \text{null}\left(\begin{bmatrix}\mathbf{A}^1\\ \mathbf{A}^3\end{bmatrix}\right)$. ■

The following gives a lower bound on the CC dimension of a loss matrix $\mathbf{L}$ in terms of the feasible subspace dimension of the trigger probability sets $\mathcal{Q}_t^{\mathbf{L}}$ at points $\mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}}$:

Theorem 16 *Let* $\mathbf{L} \in \mathbb{R}_+^{n\times k}$. *Let* $\mathbf{p} \in \Delta_n$ *and* $t \in \arg\min_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}$ *(equivalently, let* $\mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}}$*). Then*

$$\text{CCdim}(\mathbf{L}) \geq \|\mathbf{p}\|_0 - \mu_{\mathcal{Q}_t^{\mathbf{L}}}(\mathbf{p}) - 1\,.$$

The above lower bound allows us to calculate precisely the CC dimension of the 0-1 loss:

Example 11 (CC dimension of 0-1 loss) *Let* $n \in \mathbb{Z}_+$, *and consider the 0-1 loss* $\mathbf{L}^{0\text{-}1} \in \mathbb{R}_+^{n\times n}$ *defined in Example 1. Take* $\mathbf{p} = (\frac{1}{n}, \ldots, \frac{1}{n})^\top \in \Delta_n$. *Then* $\mathbf{p} \in \mathcal{Q}_t^{0\text{-}1}$ *for all* $t \in [k] = [n]$ *(see Figure 2); in particular, we have* $\mathbf{p} \in \mathcal{Q}_1^{0\text{-}1}$. *Now* $\mathcal{Q}_1^{0\text{-}1}$ *can be written as*

$$\begin{aligned}
\mathcal{Q}_1^{0\text{-}1} &= \left\{\mathbf{q} \in \Delta_n : q_1 \geq q_y \;\forall y \in \{2, \ldots, n\}\right\}\\
&= \left\{\mathbf{q} \in \mathbb{R}^n : \begin{bmatrix}-\mathbf{e}_{n-1} & \mathbf{I}_{n-1}\end{bmatrix}\mathbf{q} \leq \mathbf{0}, -\mathbf{q} \leq \mathbf{0}, \mathbf{e}_n^\top\mathbf{q} = 1\right\},
\end{aligned}$$

where $\mathbf{e}_{n-1}$, $\mathbf{e}_n$ *denote the* $(n-1) \times 1$ *and* $n \times 1$ *all ones vectors, respectively, and* $\mathbf{I}_{n-1}$ *denotes the* $(n-1) \times (n-1)$ *identity matrix. Moreover, we have* $\begin{bmatrix}-\mathbf{e}_{n-1} & \mathbf{I}_{n-1}\end{bmatrix}\mathbf{p} = 0$, $-\mathbf{p} < \mathbf{0}$. *Therefore, by Lemma 15, we have*

$$\mu_{\mathcal{Q}_1^{0\text{-}1}}(\mathbf{p}) = \text{nullity}\left(\begin{bmatrix}-\mathbf{e}_{n-1} & \mathbf{I}_{n-1}\\ & \mathbf{e}_n^\top\end{bmatrix}\right) = \text{nullity}\left(\begin{bmatrix}-1 & 1 & 0 & \ldots & 0\\ -1 & 0 & 1 & \ldots & 0\\ & & \vdots & & \\ -1 & 0 & 0 & \ldots & 1\\ 1 & 1 & 1 & \ldots & 1\end{bmatrix}\right) = 0\,.$$

Moreover, $\|\mathbf{p}\|_0 = n$. *Thus by Theorem 16, we have* $\text{CCdim}(\mathbf{L}^{0\text{-}1}) \geq n - 1$. *Combined with the upper bound of Lemma 11, this gives* $\text{CCdim}(\mathbf{L}^{0\text{-}1}) = n - 1$.

4.3 Tightness of Bounds

The upper and lower bounds above are not necessarily tight in general. For example, for the n-class ordinal regression loss of Example 2, we know that $\text{CCdim}(\mathbf{L}^{\text{ord}}) = 1$; however the upper bound of Theorem 12 only gives $\text{CCdim}(\mathbf{L}^{\text{ord}}) \leq n - 1$. Similarly, for the n-class abstain loss of Example 4, it can be shown that $\text{CCdim}(\mathbf{L}^{(?)}) = O(\ln n)$ (in fact we conjecture it to be $\Theta(\ln n)$) (Ramaswamy et al., 2015), whereas the upper bound of Theorem 12 gives $\text{CCdim}(\mathbf{L}^{(?)}) \leq n$, and the lower bound of Theorem 16 yields only $\text{CCdim}(\mathbf{L}^{(?)}) \geq 1$. However, as we show below, for certain losses $\mathbf{L}$, the bounds of Theorems 12 and 16 are in fact tight (upto an additive constant of 1).

Lemma 17 *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$. Let $\mathbf{p} \in \text{relint}(\Delta_n)$ and $c \in \mathbb{R}_+$ be such that $\mathbf{p}^\top \boldsymbol{\ell}_t = c \ \forall t \in [k]$. Then $\forall t \in [k]$,*

$$\mu_{\mathcal{Q}_t^{\mathbf{L}}}(\mathbf{p}) \leq n - \text{affdim}(\mathbf{L})\,.$$

Proof Since $\mathbf{p}^\top \boldsymbol{\ell}_t = c \ \forall t \in [k]$, we have $\mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}} \ \forall t \in [k]$. In particular, we have $\mathbf{p} \in \mathcal{Q}_1^{\mathbf{L}}$. Now

$$\mathcal{Q}_1^{\mathbf{L}} \;=\; \left\{ \mathbf{q} \in \mathbb{R}^n : \begin{bmatrix} (\boldsymbol{\ell}_2 - \boldsymbol{\ell}_1)^\top \\ \vdots \\ (\boldsymbol{\ell}_k - \boldsymbol{\ell}_1)^\top \end{bmatrix} \mathbf{q} \geq \mathbf{0}, -\mathbf{q} \leq \mathbf{0}, \mathbf{e}_n^\top \mathbf{q} = 1 \right\}.$$

Moreover, $\begin{bmatrix} (\boldsymbol{\ell}_2 - \boldsymbol{\ell}_1)^\top \\ \vdots \\ (\boldsymbol{\ell}_k - \boldsymbol{\ell}_1)^\top \end{bmatrix} \mathbf{p} = \mathbf{0}$ and $-\mathbf{p} < \mathbf{0}$. Therefore, by Lemma 15, we have

$$\mu_{\mathcal{Q}_1^{\mathbf{L}}}(\mathbf{p}) \;=\; \text{nullity}\left(\begin{bmatrix} (\boldsymbol{\ell}_2 - \boldsymbol{\ell}_1)^\top \\ \vdots \\ (\boldsymbol{\ell}_k - \boldsymbol{\ell}_1)^\top \\ \mathbf{e}_n^\top \end{bmatrix} \right) \;=\; n - \text{rank}\left(\begin{bmatrix} (\boldsymbol{\ell}_2 - \boldsymbol{\ell}_1)^\top \\ \vdots \\ (\boldsymbol{\ell}_k - \boldsymbol{\ell}_1)^\top \\ \mathbf{e}_n^\top \end{bmatrix} \right) \;\leq\; n - \text{affdim}(\mathbf{L})\,.$$

A similar proof holds for $\mu_{\mathcal{Q}_t^{\mathbf{L}}}(\mathbf{p})$ for all other $t \in [k]$. ■

Combining the above result with Theorem 16 immediately gives the following:

Theorem 18 *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$. If $\exists \mathbf{p} \in \text{relint}(\Delta_n), c \in \mathbb{R}_+$ such that $\mathbf{p}^\top \boldsymbol{\ell}_t = c \ \forall t \in [k]$, then*

$$\text{CCdim}(\mathbf{L}) \geq \text{affdim}(\mathbf{L}) - 1\,.$$

Proof Follows immediately from Theorem 16 and Lemma 17. ■

Intuitively, the condition that $\exists \mathbf{p} \in \text{relint}(\Delta_n), c \in \mathbb{R}_+$ such that $\mathbf{p}^\top \boldsymbol{\ell}_t = c \ \forall t \in [k]$ in Lemma 17 and Theorem 18 above captures the essence of a hard problem: in this case, if the underlying label probability distribution $\mathbf{p}'$ is very close to $\mathbf{p}$, then it becomes hard to decide which element $t \in [k]$ is an optimal prediction. This is essentially what leads the lower bound on the CC-dimension to become tight in this case.

A particularly useful application of Theorem 18 is to losses $\mathbf{L}$ whose columns $\boldsymbol{\ell}_t$ can be obtained from one another by permuting entries:

Corollary 19 *Let $\mathbf{L} \in \mathbb{R}_+^{n\times k}$ be such that all columns of $\mathbf{L}$ can be obtained from one another by permuting entries, i.e. $\forall t_1, t_2 \in [k], \ \exists \sigma \in \Pi_n$ such that $\ell_{y,t_2} = \ell_{\sigma(y),t_1} \ \forall y \in [n]$. Then*

$$\text{CCdim}(\mathbf{L}) \geq \text{affdim}(\mathbf{L}) - 1\,.$$

Proof Let $\mathbf{p} = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)^\top \in \text{relint}(\Delta_n)$. Let $c = \frac{\|\boldsymbol{\ell}_1\|_1}{n}$. Then under the given condition, $\mathbf{p}^\top \boldsymbol{\ell}_t = c \ \forall t \in [k]$. The result then follows from Theorem 18. ■

We will use the above corollary in establishing lower bounds on the CC dimension of certain subset ranking losses below.

5. Applications to Subset Ranking

We now consider applications of the above framework to analyzing various subset ranking problems, where each instance $x \in \mathcal{X}$ consists of a query together with a set of r documents (for simplicity, $r \in \mathbb{Z}_+$ here is fixed), and the goal is to learn a prediction model which given such an instance predicts a ranking (permutation) of the r documents (Cossock and Zhang, 2008).[7] We consider three popular losses used for subset ranking: the normalized discounted cumulative gain (NDCG) loss, the pairwise disagreement (PD) loss, and the mean average precision (MAP) loss.[8] Each of these subset ranking losses can be viewed as a specific type of multiclass loss acting on a certain label space $\mathcal{Y}$ and prediction space $\widehat{\mathcal{Y}}$. In particular, for the NDCG loss, the label space $\mathcal{Y}$ contains r-dimensional multi-valued relevance vectors; for PD loss, $\mathcal{Y}$ contains directed acyclic graphs on r nodes; and for MAP loss, $\mathcal{Y}$ contains r-dimensional binary relevance vectors. In each case, the prediction space $\widehat{\mathcal{Y}}$ is the set of permutations of r objects: $\widehat{\mathcal{Y}} = \Pi_r$. We study the convex calibration dimension of these losses below. Specifically, we show that the CC dimension of the NDCG loss is upper bounded by r (Section 5.1), and that of both the PD and MAP losses is lower bounded by a quadratic function of r (Sections 5.2 and 5.3). Our result on the CC dimension of the NDCG loss is consistent with previous results in the literature showing the existence of r-dimensional convex calibrated surrogates for NDCG (Ravikumar et al., 2011; Buffoni et al., 2011); our results on the CC dimension of the PD and MAP losses strengthen previous results of Calauzènes et al. (2012), who showed non-existence of r-dimensional convex calibrated surrogates (with a fixed argsort predictor) for PD and MAP.

5.1 Normalized Discounted Cumulative Gain (NDCG)

The NDCG loss is widely used in information retrieval applications (Järvelin and Kekäläinen, 2000). Here $\mathcal{Y}$ is the set of r-dimensional relevance vectors with say s relevance levels, $\mathcal{Y} = \{0, 1, \ldots, s-1\}^r$, and $\widehat{\mathcal{Y}}$ is the set of permutations of r objects, $\widehat{\mathcal{Y}} = \Pi_r$ (thus here $n = |\mathcal{Y}| = s^r$ and $k = |\widehat{\mathcal{Y}}| = r!$). The loss on predicting a permutation $\sigma \in \Pi_r$ when the true label is $\mathbf{y} \in \{0, 1, \ldots, s-1\}^r$ is given by

$$\ell^{\text{NDCG}}(\mathbf{y}, \sigma) \;=\; 1 - \frac{1}{z(\mathbf{y})} \sum_{i=1}^{r} \frac{2^{y_i} - 1}{\log_2(\sigma(i) + 1)}\,,$$

where $z(\mathbf{y})$ is a normalizer that ensures the loss is non-negative and depends only on $\mathbf{y}$. The NDCG loss can therefore be viewed as a multiclass loss matrix $\mathbf{L}^{\text{NDCG}} \in \mathbb{R}_+^{s^r \times r!}$. Clearly, $\text{affdim}(\mathbf{L}^{\text{NDCG}}) \leq r$, and therefore by Theorem 12, we have

$$\text{CCdim}(\mathbf{L}^{\text{NDCG}}) \;\leq\; r\,.$$

Indeed, previous results in the literature have shown the existence of r-dimensional convex calibrated surrogates for NDCG (Ravikumar et al., 2011; Buffoni et al., 2011).

7. The term 'subset ranking' here refers to the fact that in a query-based setting, each instance involves a different 'subset' of documents to be ranked; see (Cossock and Zhang, 2008).
8. Note that NDCG and MAP are generally expressed as gains, where a higher value corresponds to better performance; we can express them as non-negative losses by subtracting them from a suitable constant.

5.2 Pairwise Disagreement (PD)

Here the label space $\mathcal{Y}$ is the set of all directed acyclic graphs (DAGs) on r vertices, which we shall denote as $\mathcal{G}_r$; for each directed edge (i,j) in a graph $G \in \mathcal{G}_r$ associated with an instance $x \in \mathcal{X}$, the i-th document in the document set in x is preferred over the j-th document. The prediction space $\widehat{\mathcal{Y}}$ is again the set of permutations of r objects, $\widehat{\mathcal{Y}} = \Pi_r$. The loss on predicting a permutation $\sigma \in \Pi_r$ when the true label is $G \in \mathcal{G}_r$ is given by

$$\begin{aligned}
\ell^{\mathrm{PD}}(G,\sigma) &= \sum_{(i,j)\in G} \mathbf{1}\big(\sigma(i) > \sigma(j)\big) \\
&= \sum_{i=1}^{r}\sum_{j=1}^{r} \mathbf{1}\big((i,j)\in G\big)\cdot \mathbf{1}\big(\sigma(i) > \sigma(j)\big) \\
&= \sum_{i=1}^{r}\sum_{j=1}^{i-1} \mathbf{1}\big((i,j)\in G\big)\cdot \mathbf{1}\big(\sigma(i) > \sigma(j)\big) + \mathbf{1}\big((j,i)\in G\big)\cdot\Big(1 - \mathbf{1}\big(\sigma(i) > \sigma(j)\big)\Big) \\
&= \sum_{i=1}^{r}\sum_{j=1}^{i-1} \Big(\mathbf{1}\big((i,j)\in G\big) - \mathbf{1}\big((j,i)\in G\big)\Big)\cdot \mathbf{1}\big(\sigma(i) > \sigma(j)\big) + \sum_{i=1}^{r}\sum_{j=1}^{i-1} \mathbf{1}\big((j,i)\in G\big)\,.
\end{aligned}$$

The PD loss can be viewed as a multiclass loss matrix $\mathbf{L}^{\mathrm{PD}} \in \mathbb{R}_+^{|\mathcal{G}_r|\times r!}$. Note that the second term in the sum above depends only the label G; removing this term amounts to simply subtracting a fixed vector from each column of the loss matrix, which does not change the properties of the minimizer of the loss or its CC dimension. We can therefore consider the following loss instead:

$$\widetilde{\ell}^{\mathrm{PD}}(G,\sigma) \;=\; \sum_{i=1}^{r}\sum_{j=1}^{i-1} \Big(\mathbf{1}\big((i,j)\in G\big) - \mathbf{1}\big((j,i)\in G\big)\Big)\cdot \mathbf{1}\big(\sigma(i) > \sigma(j)\big)\,.$$

The resulting loss matrix $\widetilde{\mathbf{L}}^{\mathrm{PD}}$ clearly has rank at most $\frac{r(r-1)}{2}$. Therefore, by Corollary 13, we have

$$\mathrm{CCdim}(\mathbf{L}^{\mathrm{PD}}) \;=\; \mathrm{CCdim}(\widetilde{\mathbf{L}}^{\mathrm{PD}}) \;\leq\; \frac{r(r-1)}{2}\,.$$

In fact one can show that the rank of $\widetilde{\mathbf{L}}^{\mathrm{PD}}$ is exactly $\frac{r(r-1)}{2}$:

Proposition 20 $\mathrm{rank}(\widetilde{\mathbf{L}}^{\mathrm{PD}}) = \frac{r(r-1)}{2}$.

Moreover, it is easy to see that the columns of $\widetilde{\mathbf{L}}^{\mathrm{PD}}$ can all be obtained from one another by permuting entries. Therefore, by Corollary 19, we also have

$$\mathrm{CCdim}(\mathbf{L}^{\mathrm{PD}}) \;=\; \mathrm{CCdim}(\widetilde{\mathbf{L}}^{\mathrm{PD}}) \;\geq\; \frac{r(r-1)}{2} - 2\,.$$

Informally, this implies that a convex surrogate that achieves calibration w.r.t. $\mathbf{L}^{\mathrm{PD}}$ over the full probability simplex must effectively 'estimate' all edge weights. Formally, this strengthens previous results of Duchi et al. (2010) and Calauzènes et al. (2012). In particular, Duchi et al. (2010) showed that certain popular r-dimensional convex surrogates are not

calibrated for the PD loss, and conjectured that such convex calibrated surrogates (in r dimensions) do not exist; Calauzènes et al. (2012) showed that indeed there do not exist any r-dimensional convex surrogates that use argsort as the predictor and are calibrated for the PD loss. The above result allows us to go further and conclude that in fact, one cannot design convex calibrated surrogates for the PD loss in any prediction space of less than $\frac{r(r-1)}{2} - 2$ dimensions (regardless of the predictor used).

5.3 Mean Average Precision (MAP)

Here the label space $\mathcal{Y}$ is the set of all (non-zero) r-dimensional binary relevance vectors, $\mathcal{Y} = \{0,1\}^r \setminus \{\mathbf{0}\}$, and the prediction space $\widehat{\mathcal{Y}}$ is again the set of permutations of r objects, $\widehat{\mathcal{Y}} = \Pi_r$. The loss on predicting a permutation $\sigma \in \Pi_r$ when the true label is $\mathbf{y} \in \{0,1\}^r \setminus \{\mathbf{0}\}$ is given by

$$\begin{aligned}
\ell^{\mathrm{MAP}}(\mathbf{y},\sigma) &= 1 - \frac{1}{\|\mathbf{y}\|_1}\sum_{i:y_i=1}\frac{1}{\sigma(i)}\sum_{j=1}^{\sigma(i)} y_{\sigma^{-1}(j)} \\
&= 1 - \frac{1}{\|\mathbf{y}\|_1}\sum_{i=1}^{r}\frac{y_i}{\sigma(i)}\sum_{j=1}^{\sigma(i)} y_{\sigma^{-1}(j)} \\
&= 1 - \frac{1}{\|\mathbf{y}\|_1}\sum_{i=1}^{r}\sum_{j=1}^{i}\frac{y_{\sigma^{-1}(i)}\,y_{\sigma^{-1}(j)}}{i} \\
&= 1 - \frac{1}{\|\mathbf{y}\|_1}\sum_{i=1}^{r}\sum_{j=1}^{i}\frac{y_i\,y_j}{\max(\sigma(i),\sigma(j))} \qquad (12)
\end{aligned}$$

Thus the MAP loss can be viewed as a multiclass loss matrix $\mathbf{L}^{\mathrm{MAP}} \in \mathbb{R}_+^{(2^r-1)\times r!}$. Clearly, $\mathrm{affdim}(\mathbf{L}^{\mathrm{MAP}}) \leq \frac{r(r+1)}{2}$, and therefore by Theorem 12, we have

$$\mathrm{CCdim}(\mathbf{L}^{\mathrm{MAP}}) \;\leq\; \frac{r(r+1)}{2}\,.$$

One can also show the following lower bound on the rank of $\mathbf{L}^{\mathrm{MAP}}$:

Proposition 21 $\mathrm{rank}(\mathbf{L}^{\mathrm{MAP}}) \geq \frac{r(r-1)}{2} - 2$.

Again, it is easy to see that the columns of $\mathbf{L}^{\mathrm{MAP}}$ can all be obtained from one another by permuting entries, and therefore by Corollary 19, we have

$$\mathrm{CCdim}(\mathbf{L}^{\mathrm{MAP}}) \;\geq\; \frac{r(r-1)}{2} - 4\,.$$

This again strengthens a previous result of Calauzènes et al. (2012), who showed that there do not exist any r-dimensional convex surrogates that use argsort as the predictor and are calibrated for the MAP loss. As with the PD loss, the above result allows us to go further and conclude that in fact, one cannot design convex calibrated surrogates for the MAP loss in any prediction space of less than $\frac{r(r-1)}{2} - 4$ dimensions (regardless of the predictor used).

6. Conclusion

We have developed a unified framework for studying consistency properties of surrogate risk minimization algorithms for general multiclass learning problems, defined by a general multiclass loss matrix. In particular, we have introduced the notion of *convex calibration dimension* (CC dimension) of a multiclass loss matrix, a fundamental quantity that measures the smallest 'size' of a prediction space in which it is possible to design a convex surrogate that is calibrated with respect to the given loss matrix, and have used this to analyze consistency properties of surrogate losses for various multiclass learning problems.

Our study both generalizes previous results and sheds new light on various multiclass losses. For example, our analysis shows that for the n-class 0-1 loss, any convex calibrated surrogate must necessarily entail learning at least $n-1$ real-valued functions, thus showing that the calibrated multiclass surrogate of Lee et al. (2004), whose minimization entails learning n real-valued functions, is essentially not improvable (in the sense of the number of real-valued functions that need to be learned). Another implication of our study is to the pairwise disagreement (PD) and mean average precision (MAP) losses for subset ranking: while previous results have shown that for subset ranking problems with r documents per query, there do not exist r-dimensional convex calibrated surrogates for the PD and MAP losses, our analysis shows that (a) these losses do admit convex calibrated surrogates in higher dimensions, and (b) to obtain such convex calibrated surrogates for these losses, one needs to operate in an $\Omega(r^2)$-dimensional surrogate prediction space (i.e. one needs to learn $\Omega(r^2)$ real-valued functions, rather than just r real-valued 'scoring' functions).

As discussed in Section 4.3, while the upper and lower bounds we have obtained on the CC dimension are tight (up to an additive constant of 1) for certain classes of loss matrices, they can be quite loose in general. An important open direction is to obtain a characterization of the CC dimension in more general settings. It would also be useful to develop methods for deriving explicit surrogate regret bounds for general calibrated surrogates, through which one can relate the excess target risk to the excess surrogate risk for any multiclass loss and corresponding calibrated surrogate. Finally, another interesting direction would be to develop a generic procedure for designing convex calibrated surrogates operating on a 'minimal' space according to the CC dimension of a given loss matrix. There has been some recent progress in this direction in (Ramaswamy et al., 2013), where a general method is described for designing convex calibrated surrogates in a surrogate space with dimension at most the rank of the given loss matrix. However, while the rank forms an upper bound on the CC dimension of the loss matrix, as discussed above, this bound is not always tight, giving rise to the possibility of designing convex calibrated surrogates in lower-dimensional spaces for certain losses. Resolving these issues will contribute significantly to our understanding of the conditions under which convex calibrated surrogates can be designed for a given multiclass learning problem.

7. Proofs

7.1 Proof of Theorem 8

The proof uses the following technical lemma:

Lemma 22 *Let $\mathbf{L} \in \mathbb{R}_+^{n \times k}$ and $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$. Suppose there exist $r \in \mathbb{N}$ and $\mathbf{z}_1, \ldots, \mathbf{z}_r \in \mathcal{R}_\psi$ such that $\bigcup_{j=1}^r \mathcal{N}^\psi(\mathbf{z}_j) = \Delta_n$ and for each $j \in [r]$, $\exists t \in [k]$ such that $\mathcal{N}^\psi(\mathbf{z}_j) \subseteq \mathcal{Q}_t^{\mathbf{L}}$. Then any element $\mathbf{z} \in \mathcal{S}_\psi$ can be written as $\mathbf{z} = \mathbf{z}' + \mathbf{z}''$ for some $\mathbf{z}' \in \text{conv}(\{\mathbf{z}_1, \ldots, \mathbf{z}_r\})$ and $\mathbf{z}'' \in \mathbb{R}_+^n$.*

Proof (Proof of Lemma 22)

Let $\mathcal{S}' = \{\mathbf{z}' + \mathbf{z}'' : \mathbf{z}' \in \text{conv}(\{\mathbf{z}_1, \ldots, \mathbf{z}_r\}), \mathbf{z}'' \in \mathbb{R}_+^n\}$, and suppose there exists a point $\mathbf{z} \in \mathcal{S}_\psi$ which cannot be decomposed as claimed, i.e. such that $\mathbf{z} \notin \mathcal{S}'$. Then by the Hahn-Banach theorem (e.g. see Gallier (2009), corollary 3.10), there exists a hyperplane that strictly separates $\mathbf{z}$ from $\mathcal{S}'$, i.e. $\exists \mathbf{w} \in \mathbb{R}^n$ such that $\mathbf{w}^\top \mathbf{z} < \mathbf{w}^\top \mathbf{a}\ \forall \mathbf{a} \in \mathcal{S}'$. It is easy to see that $\mathbf{w} \in \mathbb{R}_+^n$ (since a negative component in $\mathbf{w}$ would allow us to choose an element $\mathbf{a}$ from $\mathcal{S}'$ with arbitrarily small $\mathbf{w}^\top \mathbf{a}$).

Now consider the vector $\mathbf{q} = \mathbf{w} / \sum_{i=1}^n w_i \in \Delta_n$. Since $\bigcup_{j=1}^r \mathcal{N}^\psi(\mathbf{z}_j) = \Delta_n$, $\exists j \in [r]$ such that $\mathbf{q} \in \mathcal{N}^\psi(\mathbf{z}_j)$. By definition of positive normals, this gives $\mathbf{q}^\top \mathbf{z}_j \leq \mathbf{q}^\top \mathbf{z}$, and therefore $\mathbf{w}^\top \mathbf{z}_j \leq \mathbf{w}^\top \mathbf{z}$. But this contradicts our construction of $\mathbf{w}$ (since $\mathbf{z}_j \in \mathcal{S}'$). Thus it must be the case that every $\mathbf{z} \in \mathcal{S}_\psi$ is also an element of $\mathcal{S}'$. ■

Proof (Proof of Theorem 8)

We will show $\mathbf{L}$-calibration of $\boldsymbol{\psi}$ via Lemma 2. For each $j \in [r]$, let

$$T_j = \left\{ t \in [k] : \mathcal{N}^\psi(\mathbf{z}_j) \subseteq \mathcal{Q}_t^{\mathbf{L}} \right\};$$

by assumption, $T_j \neq \emptyset\ \forall j \in [r]$. By Lemma 22, for every $\mathbf{z} \in \mathcal{S}_\psi$, $\exists \boldsymbol{\alpha} \in \Delta_r, \mathbf{u} \in \mathbb{R}_+^n$ such that $\mathbf{z} = \sum_{j=1}^r \alpha_j \mathbf{z}_j + \mathbf{u}$. For each $\mathbf{z} \in \mathcal{S}_\psi$, arbitrarily fix a unique $\boldsymbol{\alpha}^{\mathbf{z}} \in \Delta_r$ and $\mathbf{u}^{\mathbf{z}} \in \mathbb{R}_+^n$ satisfying the above, i.e. such that

$$\mathbf{z} = \sum_{j=1}^r \alpha_j^{\mathbf{z}} \mathbf{z}_j + \mathbf{u}^{\mathbf{z}}.$$

Now define $\text{pred}' : \mathcal{S}_\psi \to [k]$ as

$$\text{pred}'(\mathbf{z}) = \min \left\{ t \in [k] : \exists j \in [r] \text{ such that } \alpha_j^{\mathbf{z}} \geq \tfrac{1}{r} \text{ and } t \in T_j \right\}.$$

We will show pred' satisfies the condition for ℓ-calibration.

Fix any $\mathbf{p} \in \Delta_n$. Let

$$J_{\mathbf{p}} = \left\{ j \in [r] : \mathbf{p} \in \mathcal{N}^\psi(\mathbf{z}_j) \right\};$$

since $\Delta_n = \bigcup_{j=1}^r \mathcal{N}^\psi(\mathbf{z}_j)$, we have $J_{\mathbf{p}} \neq \emptyset$. Clearly,

$$\forall j \in J_{\mathbf{p}} : \mathbf{p}^\top \mathbf{z}_j = \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} \tag{13}$$

$$\forall j \notin J_{\mathbf{p}} : \mathbf{p}^\top \mathbf{z}_j > \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} \tag{14}$$

Moreover, from definition of T_j, we have

$$\forall j \in J_{\mathbf{p}} : \quad t \in T_j \implies \mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}} \implies t \in \text{argmin}_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}.$$

Thus we get

$$\forall j \in J_{\mathbf{p}} : \quad T_j \subseteq \operatorname{argmin}_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'} . \tag{15}$$

Now, for any $\mathbf{z} \in \mathcal{S}_\psi$ for which $\text{pred}'(\mathbf{z}) \notin \arg\min_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}$, we must have $\alpha_j^{\mathbf{z}} \geq \frac{1}{r}$ for at least one $j \notin J_{\mathbf{p}}$ (otherwise, we would have $\text{pred}'(\mathbf{z}) \in T_j$ for some $j \in J_{\mathbf{p}}$, giving $\text{pred}'(\mathbf{z}) \in \arg\min_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}$, a contradiction). Thus we have

$$\begin{aligned}
\inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}'(\mathbf{z}) \notin \operatorname{argmin}_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}} \mathbf{p}^\top \mathbf{z} &= \inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}'(\mathbf{z}) \notin \operatorname{argmin}_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}} \sum_{j=1}^{r} \alpha_j^{\mathbf{z}} \mathbf{p}^\top \mathbf{z}_j + \mathbf{p}^\top \mathbf{u}^{\mathbf{z}} && (16) \\
&\geq \inf_{\boldsymbol{\alpha} \in \Delta_r : \alpha_j \geq \frac{1}{r} \text{ for some } j \notin J_{\mathbf{p}}} \sum_{j=1}^{r} \alpha_j \mathbf{p}^\top \mathbf{z}_j && (17) \\
&\geq \min_{j \notin J_{\mathbf{p}}} \inf_{\alpha_j \in [\frac{1}{r}, 1]} \alpha_j \mathbf{p}^\top \mathbf{z}_j + (1 - \alpha_j) \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} && (18) \\
&> \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} , && (19)
\end{aligned}$$

where the last inequality follows from Eq. (14). Since the above holds for all $\mathbf{p} \in \Delta_n$, by Lemma 2, we have that $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated. ■

7.2 Proof of Lemma 11

Proof For each $\mathbf{u} \in \mathcal{C}$, define $\mathbf{p}^{\mathbf{u}} = \begin{pmatrix} \mathbf{u} \\ 1 - \sum_{j=1}^{n-1} u_j \end{pmatrix} \in \Delta_n$. Define $\text{pred} : \mathcal{C} \to [k]$ as

$$\text{pred}(\mathbf{u}) \;=\; \min \left\{ t \in [k] : \mathbf{p}^{\mathbf{u}} \in Q_t^{\mathbf{L}} \right\} .$$

We will show that pred satisfies the condition of Definition 1.

Fix $\mathbf{p} \in \Delta_n$. It can be seen that

$$\mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \;=\; \sum_{j=1}^{n-1} \left(p_j (u_j - 1)^2 + (1 - p_j) \, {u_j}^2 \right) .$$

Minimizing the above over $\mathbf{u}$ yields the unique minimizer $\mathbf{u}^* = (p_1, \ldots, p_{n-1})^\top \in \mathcal{C}$, which after some calculation gives

$$\inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \;=\; \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}^*) \;=\; \sum_{j=1}^{n-1} p_j (1 - p_j) .$$

Now, for each $t \in [k]$, define

$$\text{regret}_{\mathbf{p}}^{\mathbf{L}}(t) \;\triangleq\; \mathbf{p}^\top \boldsymbol{\ell}_t - \min_{t' \in [k]} \mathbf{p}^\top \boldsymbol{\ell}_{t'} .$$

Clearly, $\text{regret}_{\mathbf{p}}^{\mathbf{L}}(t) = 0 \iff \mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}}$. Note also that $\mathbf{p}^{\mathbf{u}^*} = \mathbf{p}$, and therefore $\text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}^*)) = 0$. Let

$$\epsilon \;=\; \min_{t \in [k] : \mathbf{p} \notin \mathcal{Q}_t^{\mathbf{L}}} \text{regret}_{\mathbf{p}}^{\mathbf{L}}(t) \;>\; 0 .$$

Then we have

$$\begin{aligned}
\inf_{\mathbf{u} \in \mathcal{C} : \text{pred}(\mathbf{u}) \notin \operatorname{argmin}_t \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) &= \inf_{\mathbf{u} \in \mathcal{C} : \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u})) \geq \epsilon} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) && (20) \\
&= \inf_{\mathbf{u} \in \mathcal{C} : \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u})) \geq \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}^*)) + \epsilon} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) . && (21)
\end{aligned}$$

Now, we claim that the mapping $\mathbf{u} \mapsto \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}))$ is continuous at $\mathbf{u} = \mathbf{u}^*$. To see this, suppose the sequence $\mathbf{u}_m$ converges to $\mathbf{u}^*$. Then it is easy to see that $\mathbf{p}^{\mathbf{u}_m}$ converges to $\mathbf{p}^{\mathbf{u}^*} = \mathbf{p}$, and therefore for each $t \in [k]$, $(\mathbf{p}^{\mathbf{u}_m})^\top \boldsymbol{\ell}_t$ converges to $\mathbf{p}^\top \boldsymbol{\ell}_t$. Since by definition of pred we have that for all m, $\text{pred}(\mathbf{u}_m) \in \operatorname{argmin}_t (\mathbf{p}^{\mathbf{u}_m})^\top \boldsymbol{\ell}_t$, this implies that for all large enough m, $\text{pred}(\mathbf{u}_m) \in \operatorname{argmin}_t \mathbf{p}^\top \boldsymbol{\ell}_t$. Thus for all large enough m, $\text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}_m)) = 0$; i.e. the sequence $\text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}_m))$ converges to $\text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}^*))$, yielding continuity at $\mathbf{u}^*$. In particular, this implies $\exists \delta > 0$ such that

$$\|\mathbf{u} - \mathbf{u}^*\| < \delta \implies \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u})) - \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}^*)) < \epsilon .$$

This gives

$$\begin{aligned}
\inf_{\mathbf{u} \in \mathcal{C} : \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u})) \geq \text{regret}_{\mathbf{p}}^{\mathbf{L}}(\text{pred}(\mathbf{u}^*)) + \epsilon} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) &\geq \inf_{\mathbf{u} \in \mathcal{C} : \|\mathbf{u} - \mathbf{u}^*\| \geq \delta} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) && (22) \\
&> \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) , && (23)
\end{aligned}$$

where the last inequality holds since $\mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})$ is a strictly convex function of $\mathbf{u}$ and $\mathbf{u}^*$ is its unique minimizer. The above sequence of inequalities give us that

$$\inf_{\mathbf{u} \in \mathcal{C} : \text{pred}(\mathbf{u}) \notin \operatorname{argmin}_t \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \;>\; \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) . \tag{24}$$

Since this holds for all $\mathbf{p} \in \Delta_n$, we have that $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated. ■

7.3 Proof of Theorem 16

The proof will require the lemma below, which relates the feasible subspace dimensions of different trigger probability sets at points in their intersection; we will also make critical use of the notion of ϵ-subdifferentials of convex functions (Bertsekas et al., 2003), the main properties of which are also recalled below.

Lemma 23 *Let $\ell : [n] \times [k] \to \mathbb{R}_+^n$. Let $\mathbf{p} \in \text{relint}(\Delta_n)$. Then for any $t_1, t_2 \in \arg\min_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}$ (i.e. such that $\mathbf{p} \in \mathcal{Q}_{t_1}^{\mathbf{L}} \cap \mathcal{Q}_{t_2}^{\mathbf{L}}$),*

$$\mu_{\mathcal{Q}_{t_1}^{\mathbf{L}}}(\mathbf{p}) = \mu_{\mathcal{Q}_{t_2}^{\mathbf{L}}}(\mathbf{p}) .$$

Proof (Proof of Lemma 23)

Let $t_1, t_2 \in \arg\min_{t'} \mathbf{p}^\top \boldsymbol{\ell}_{t'}$ (i.e. $\mathbf{p} \in \mathcal{Q}_{t_1}^{\mathbf{L}} \cap \mathcal{Q}_{t_2}^{\mathbf{L}}$). Now

$$\mathcal{Q}_{t_1}^{\mathbf{L}} = \left\{ \mathbf{q} \in \mathbb{R}^n : -\mathbf{q} \leq \mathbf{0}, \mathbf{e}_n^\top \mathbf{q} = 1, (\boldsymbol{\ell}_{t_1} - \boldsymbol{\ell}_t)^\top \mathbf{q} \leq 0 \;\forall t \in [k] \right\} ,$$

where $\mathbf{e}_n$ denotes the $n \times 1$ all ones vector. Moreover, we have $-\mathbf{p} < \mathbf{0}$, and $(\boldsymbol{\ell}_{t_1} - \boldsymbol{\ell}_t)^\top \mathbf{p} = 0$ iff $\mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}}$. Let $\left\{t \in [k] : \mathbf{p} \in \mathcal{Q}_t^{\mathbf{L}}\right\} = \left\{\widetilde{t}_1, \ldots, \widetilde{t}_r\right\}$ for some $r \in [k]$. Then by Lemma 15, we have

$$\mu_{Q_{t_1}^{\mathbf{L}}} = \text{nullity}(\mathbf{A}_1)\,,$$

where $\mathbf{A}_1 \in \mathbb{R}^{(r+1)\times n}$ is a matrix containing r rows of the form $(\boldsymbol{\ell}_{t_1} - \boldsymbol{\ell}_{\widetilde{t}_j})^\top, j \in [r]$ and the all ones row. Similarly, we get

$$\mu_{Q_{t_2}^{\mathbf{L}}} = \text{nullity}(\mathbf{A}_2)\,,$$

where $\mathbf{A}_2 \in \mathbb{R}^{(r+1)\times n}$ is a matrix containing r rows of the form $(\boldsymbol{\ell}_{t_2} - \boldsymbol{\ell}_{\widetilde{t}_j})^\top, j \in [r]$ and the all ones row. It can be seen that the subspaces spanned by the first r rows of $\mathbf{A}_1$ and $\mathbf{A}_2$ are both equal to the subspace parallel to the affine space containing $\boldsymbol{\ell}_{\widetilde{t}_1}, \ldots, \boldsymbol{\ell}_{\widetilde{t}_r}$. Thus both $\mathbf{A}_1$ and $\mathbf{A}_2$ have the same row space and hence the same null space and nullity, and therefore $\mu_{\mathcal{Q}_{t_1}^{\mathbf{L}}}(\mathbf{p}) = \mu_{\mathcal{Q}_{t_2}^{\mathbf{L}}}(\mathbf{p})$. ■

ϵ-Subdifferentials of a Convex Function. For any $\epsilon > 0$, the ϵ-subdifferential of a convex function $\phi : \mathbb{R}^d \to \overline{\mathbb{R}}$ at a point $\mathbf{u}_0 \in \mathbb{R}^d$ is defined as follows (Bertsekas et al., 2003):

$$\partial_\epsilon \phi(\mathbf{u}_0) = \left\{\mathbf{w} \in \mathbb{R}^d : \phi(\mathbf{u}) - \phi(\mathbf{u}_0) \geq \mathbf{w}^\top(\mathbf{u} - \mathbf{u}_0) - \epsilon \;\; \forall \mathbf{u} \in \mathbb{R}^d\right\}.$$

We recall some important properties of ϵ-subdifferentials below:

- $\mathbf{0} \in \partial_\epsilon \phi(\mathbf{u}_0) \iff \phi(\mathbf{u}_0) \leq \inf_{\mathbf{u}\in\mathbb{R}^d} \phi(\mathbf{u}) + \epsilon\,.$
- For any $\lambda > 0$, $\partial_\epsilon(\lambda\phi(\mathbf{u}_0)) = \lambda\, \partial_{(\epsilon/\lambda)}\phi(\mathbf{u}_0)\,.$
- If $\phi = \phi_1 + \ldots + \phi_n$ for some convex functions $\phi_i : \mathbb{R}^d \to \overline{\mathbb{R}}$, then

$$\partial_\epsilon \phi(\mathbf{u}_0) \;\subseteq\; \partial_\epsilon \phi_1(\mathbf{u}_0) + \ldots + \partial_\epsilon \phi_n(\mathbf{u}_0) \;\subseteq\; \partial_{n\epsilon}\phi(\mathbf{u}_0)\,.$$

- $\epsilon_1 \leq \epsilon_2 \implies \partial_{\epsilon_1}\phi(\mathbf{u}_0) \subseteq \partial_{\epsilon_2}\phi(\mathbf{u}_0)\,.$

We are now ready to give the proof of the lower bound on the CC dimension.

Proof (Proof of Theorem 16)

Let $d \in \mathbb{Z}_+$ be such that there exists a convex set $\mathcal{C} \subseteq \mathbb{R}^d$ and surrogate loss $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}_+^n$ such that $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated. We will show that $d \geq \|\mathbf{p}\|_0 - \mu_{\mathcal{Q}_t^{\mathbf{L}}}(\mathbf{p}) - 1$. We consider two cases:

Case 1: $\mathbf{p} \in \text{relint}(\Delta_n)$.

In this case $\|\mathbf{p}\|_0 = n$. We will show that there exist $\mathcal{H} \subseteq \Delta_n$ and $t_0 \in [k]$ satisfying the following three conditions:

(a) $\mathbf{p} \in \mathcal{H}$;
(b) $\mu_{\mathcal{H}}(\mathbf{p}) = n - d - 1$; and
(c) $\mathcal{H} \subseteq \mathcal{Q}_{t_0}^{\mathbf{L}}$.

Clearly, conditions (a) and (c) above imply $\mathbf{p} \in \mathcal{Q}_{t_0}^{\mathbf{L}}$. Conditions (b) and (c) will then give

$$\mu_{\mathcal{Q}_{t_0}^{\mathbf{L}}}(\mathbf{p}) \;\geq\; \mu_{\mathcal{H}}(\mathbf{p}) \;=\; n - d - 1\,.$$

Further, by Lemma 23, we will then have that

$$\mu_{\mathcal{Q}_t^{\mathbf{L}}}(\mathbf{p}) \;=\; \mu_{\mathcal{Q}_{t_0}^{\mathbf{L}}}(\mathbf{p}) \;\geq\; n - d - 1\,,$$

thus proving the claim.

We now show how to construct $\mathcal{H}$ and t_0 satisfying the above conditions. Let $\{\mathbf{u}_m\}$ be a sequence in $\mathcal{C}$ such that

$$\mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_m) \longrightarrow \inf_{\mathbf{z}\in\mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} = \inf_{\mathbf{u}\in\mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})\,.$$

Let

$$\epsilon_m = \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_m) - \inf_{\mathbf{u}\in\mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})\,.$$

Then clearly $\epsilon_m \to 0$. Now, for each m, we have

$$\begin{aligned}
\mathbf{0} \;&\in\; \partial_{\epsilon_m}(\mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_m)) \\
&\subseteq\; \partial_{\epsilon_m}(p_1\psi_1(\mathbf{u}_m)) + \ldots + \partial_{\epsilon_m}(p_n\psi_n(\mathbf{u}_m)) \\
&=\; p_1\, \partial_{(\epsilon_m/p_1)}(\psi_1(\mathbf{u}_m)) + \ldots + p_n\, \partial_{(\epsilon_m/p_n)}(\psi_n(\mathbf{u}_m))\,.
\end{aligned}$$

Therefore $\exists \mathbf{w}_y^m \in \partial_{(\epsilon_m/p_y)}(\psi_y(\mathbf{u}_m))\;\forall y \in [n]$ such that

$$\sum_{y=1}^n p_y \mathbf{w}_y^m = \mathbf{0}\,.$$

Let

$$\mathbf{A}^m = \left[\mathbf{w}_1^m \ldots \mathbf{w}_n^m\right] \in \mathbb{R}^{d\times n}\,,$$

and define

$$\begin{aligned}
\mathcal{H}_m \;&=\; \left\{\mathbf{q} \in \Delta_n : \mathbf{A}^m\mathbf{q} = \mathbf{0}\right\} \\
&=\; \left\{\mathbf{q} \in \mathbb{R}^n : \mathbf{A}^m\mathbf{q} = \mathbf{0}\,, \mathbf{e}_n^\top\mathbf{q} = 1\,, -\mathbf{q} \leq \mathbf{0}\right\},
\end{aligned}$$

where $\mathbf{e}_n$ denotes the $n \times 1$ all ones vector. Clearly, $\mathbf{p} \in \mathcal{H}_m$ and $-\mathbf{p} < \mathbf{0}$; therefore by Lemma 15, we have

$$\mu_{\mathcal{H}_m}(\mathbf{p}) \;=\; \text{nullity}\left(\begin{bmatrix}\mathbf{A}^m \\ \mathbf{e}_n^\top\end{bmatrix}\right) \;\geq\; n - (d+1)\,.$$

This means that there exist $(n-d-1)$ orthonormal vectors $\mathbf{v}_1^m, \ldots, \mathbf{v}_{n-d-1}^m \in \mathbb{R}^n$ whose span $\mathcal{V}^m = \text{span}(\{\mathbf{v}_1^m, \ldots, \mathbf{v}_{n-d-1}^m\})$ is contained in $\mathcal{F}_{\mathcal{H}_m}(\mathbf{p}) \cap (-\mathcal{F}_{\mathcal{H}_m}(\mathbf{p}))$. It can be verified that this in turn implies

$$\{\mathbf{p} + \mathbf{v} : \mathbf{v} \in \mathcal{V}^m\} \cap \Delta_n \;\subseteq\; \mathcal{H}_m\,.$$

Now, $\{(\mathbf{v}_1^m, \ldots, \mathbf{v}_{n-d-1}^m)\}$ is a bounded sequence in $(\mathbb{R}^n)^{n-d-1}$ and must therefore have a convergent subsequence, say with indices $r_1, r_2, \ldots$, converging to some limit point $(\mathbf{v}_1, \ldots, \mathbf{v}_{n-d-1}) \in (\mathbb{R}^n)^{n-d-1}$. It can be verified that $\mathbf{v}_1, \ldots, \mathbf{v}_{n-d-1}$ must also form an orthonormal set of vectors. Let $\mathcal{V} = \text{span}(\{\mathbf{v}_1, \ldots, \mathbf{v}_{n-d-1}\})$, and define

$$\mathcal{H} \;=\; \{\mathbf{p} + \mathbf{v} : \mathbf{v} \in \mathcal{V}\} \cap \Delta_n \,.$$

Clearly $\mathbf{p} \in \mathcal{H}$, and moreover, since $\mathbf{p} \in \text{relint}(\Delta_n)$, we have $\mu_{\mathcal{H}}(\mathbf{p}) = n - d - 1$, thus satisfying conditions (a) and (b) above.

For condition (c), we will show that $\mathcal{H} \subseteq \mathcal{N}^{\psi}(\{\mathbf{z}_m\})$, where $\mathbf{z}_m = \boldsymbol{\psi}(\mathbf{u}_{r_m})$; the claim will then follow from Theorem 7. Consider any point $\mathbf{q} \in \mathcal{H}$. By construction of $\mathcal{H}$, we must have that $\mathbf{q}$ is the limit point of some convergent sequence $\{\mathbf{q}_m\}$ in Δ_n satisfying $\mathbf{q}_m \in \mathcal{H}_{r_m}\ \forall m$, i.e. $\mathbf{A}^{r_m}\mathbf{q}_m = \mathbf{0}\ \forall m$. Therefore for each m, we have

$$\begin{aligned}
\mathbf{0} \;=\; \sum_{y=1}^{n} q_{m,y}\mathbf{w}_y^{r_m} \;&\in\; \sum_{y=1}^{n} q_{m,y}\,\partial_{(\epsilon_{r_m}/p_y)}(\psi_y(\mathbf{u}_{r_m})) \\
&=\; \sum_{y=1}^{n} \partial_{(\epsilon_{r_m} q_{m,y}/p_y)}(q_y^m\,\psi_y(\mathbf{u}_{r_m})) \\
&\subseteq\; \sum_{y=1}^{n} \partial_{(\epsilon_{r_m}/p_{\min})}(q_{m,y}\,\psi_y(\mathbf{u}_{r_m})) \\
&\subseteq\; \partial_{(n\epsilon_{r_m}/p_{\min})}(\mathbf{q}_m{}^{\top}\boldsymbol{\psi}(\mathbf{u}_{r_m}))\,,
\end{aligned}$$

where $p_{\min} = \min_{y\in[n]} p_y > 0$ (since $\mathbf{p} \in \text{relint}(\Delta_n)$). This gives for each m:

$$\mathbf{q}_m{}^{\top}\mathbf{z}_m \;=\; \mathbf{q}_m{}^{\top}\boldsymbol{\psi}(\mathbf{u}_{r_m}) \;\leq\; \inf_{\mathbf{u}\in\mathbb{R}^d} \mathbf{q}_m{}^{\top}\boldsymbol{\psi}(\mathbf{u}) + \frac{n\epsilon_{r_m}}{p_{\min}} \;=\; \inf_{\mathbf{z}\in\mathcal{S}_{\psi}} \mathbf{q}_m{}^{\top}\mathbf{z} + \frac{n\epsilon_{r_m}}{p_{\min}}\,.$$

Taking limits as $m \to \infty$, we thus get

$$\lim_{m\to\infty} \mathbf{q}_m{}^{\top}\mathbf{z}_m \leq \lim_{m\to\infty} \inf_{\mathbf{z}\in\mathcal{S}_{\psi}} \mathbf{q}_m{}^{\top}\mathbf{z}\,. \tag{25}$$

Now, since $\{\mathbf{z}_m\}$ is bounded, we have

$$\lim_{m\to\infty} \mathbf{q}_m{}^{\top}\mathbf{z}_m = \lim_{m\to\infty} (\mathbf{q}_m - \mathbf{q})^{\top}\mathbf{z}_m + \lim_{m\to\infty} \mathbf{q}^{\top}\mathbf{z}_m = \lim_{m\to\infty} \mathbf{q}^{\top}\mathbf{z}_m\,. \tag{26}$$

Moreover, since the mapping $\mathbf{p} \mapsto \inf_{\mathbf{z}\in\mathcal{S}_{\psi}} \mathbf{p}^{\top}\mathbf{z}$ is continuous over its domain Δ_n (see Lemma 24), we have

$$\lim_{m\to\infty} \inf_{\mathbf{z}\in\mathcal{S}_{\psi}} \mathbf{q}_m{}^{\top}\mathbf{z} = \inf_{\mathbf{z}\in\mathcal{S}_{\psi}} \mathbf{q}^{\top}\mathbf{z}\,. \tag{27}$$

Putting together Equations (25), (26) and (27), we therefore get

$$\lim_{m\to\infty} \mathbf{q}^{\top}\mathbf{z}_m = \inf_{\mathbf{z}\in\mathcal{S}_{\psi}} \mathbf{q}^{\top}\mathbf{z}\,. \tag{28}$$

Thus $\mathbf{q} \in \mathcal{N}^{\psi}(\{\mathbf{z}_m\})$. Since $\mathbf{q}$ was an arbitrary point in $\mathcal{H}$, this gives $\mathcal{H} \subseteq \mathcal{N}^{\psi}(\{\mathbf{z}_m\})$. The claim follows.

Case 2: $\mathbf{p} \notin \text{relint}(\Delta_n)$.

For each $\mathbf{b} \in \{0,1\}^n \setminus \{\mathbf{0}\}$, define

$$\mathcal{P}^{\mathbf{b}} = \left\{\mathbf{q} \in \Delta_n : q_y > 0 \iff b_y = 1\right\}.$$

Clearly, the set $\{\mathcal{P}^{\mathbf{b}} : \mathbf{b} \in \{0,1\}^n \setminus \{\mathbf{0}\}\}$ forms a partition of Δ_n. Moreover, for $\mathbf{b} = \mathbf{e}_n$ (the $n \times 1$ all ones vector), we have

$$\mathcal{P}^{\mathbf{e}_n} \;=\; \left\{\mathbf{q} \in \Delta_n : q_y > 0\ \forall y \in [n]\right\} \;=\; \text{relint}(\Delta_n)\,.$$

Therefore we have $\mathbf{p} \in \mathcal{P}^{\mathbf{b}}$ for some $\mathbf{b} \in \{0,1\}^n \setminus \{\mathbf{0}, \mathbf{e}_n\}$, with $\|\mathbf{p}\|_0 = \|\mathbf{b}\|_0$. Now, define $\boldsymbol{\psi}^{\mathbf{b}} : \mathcal{C} \to \mathbb{R}_+^{\|\mathbf{b}\|_0}$, $\mathbf{L}^{\mathbf{b}} \in \mathbb{R}_+^{\|\mathbf{b}\|_0 \times k}$, and $\mathbf{p}^{\mathbf{b}} \in \Delta_{\|\mathbf{b}\|_0}$ as projections of $\boldsymbol{\psi}$, $\mathbf{L}$ and $\mathbf{p}$ onto the $\|\mathbf{b}\|_0$ coordinates $y : b_y = 1$, so that $\boldsymbol{\psi}^{\mathbf{b}}(\mathbf{u})$ contains the elements of $\boldsymbol{\psi}(\mathbf{u})$ corresponding to coordinates $y : b_y = 1$, the columns $\boldsymbol{\ell}_t^{\mathbf{b}}$ of $\mathbf{L}^{\mathbf{b}}$ contain the elements of the columns $\boldsymbol{\ell}_t$ of $\mathbf{L}$ corresponding to the same coordinates $y : b_y = 1$, and similarly, $\mathbf{p}^{\mathbf{b}}$ contains the strictly positive elements of $\mathbf{p}$. Since $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated, and therefore in particular is calibrated w.r.t. $\mathbf{L}$ over $\{\mathbf{q} \in \Delta_n : q_y = 0\ \forall y : b_y = 0\}$, we have that $\boldsymbol{\psi}^{\mathbf{b}}$ is $\mathbf{L}^{\mathbf{b}}$-calibrated (over $\Delta_{\|\mathbf{b}\|_0}$). Moreover, by construction, we have $\mathbf{p}^{\mathbf{b}} \in \text{relint}(\Delta_{\|\mathbf{b}\|_0})$. Therefore by Case 1 above, we have

$$d \;\geq\; \|\mathbf{b}\|_0 - \mu_{\mathcal{Q}_t^{\mathbf{L}^{\mathbf{b}}}}(\mathbf{p}^{\mathbf{b}}) - 1\,.$$

The claim follows since $\mu_{\mathcal{Q}_t^{\mathbf{L}^{\mathbf{b}}}}(\mathbf{p}^{\mathbf{b}}) \leq \mu_{\mathcal{Q}_t^{\mathbf{L}}}(\mathbf{p})$.

■

7.4 Proof of Proposition 20

Proof We will establish $\text{rank}(\widetilde{\mathbf{L}}^{\text{PD}}) \geq \frac{r(r-1)}{2}$ by showing the existence of $\frac{r(r-1)}{2}$ linearly independent rows in $\widetilde{\mathbf{L}}^{\text{PD}}$; the claim will then follow by combining this with the previously stated upper bound $\text{rank}(\widetilde{\mathbf{L}}^{\text{PD}}) \leq \frac{r(r-1)}{2}$.

Consider the $\frac{r(r-1)}{2}$ rows of $\widetilde{\mathbf{L}}^{\text{PD}}$ corresponding to graphs consisting of single directed edges (i,j) with $i < j$. We claim these rows are linearly independent. To see this, suppose for the sake of contradiction that this is not the case. Then one of these rows, say the row corresponding to the graph with directed edge (a,b) for some $a, b \in [r]$, $a < b$, can be written as a linear combination of the other rows:

$$\widetilde{\ell}^{\text{PD}}((a,b),\sigma) \;=\; \sum_{1\leq i<j\leq r\,,\,(i,j)\neq(a,b)} c_{ij}\,\widetilde{\ell}^{\text{PD}}((i,j),\sigma) \qquad \forall \sigma \in \Pi_r\,, \tag{29}$$

for some coefficients $c_{ij} \in \mathbb{R}$. Now consider two permutations $\sigma, \sigma' \in \Pi_r$ such that

$$\begin{aligned}
\sigma(a) &= \sigma'(b) \\
\sigma(b) &= \sigma'(a) \\
\sigma(i) &= \sigma'(i)\ \forall i \neq a, b\,.
\end{aligned}$$

Then applying Eq. (29) to these two permutations gives

$$\widetilde{\ell}^{\mathrm{PD}}((a,b),\sigma) \;=\; \widetilde{\ell}^{\mathrm{PD}}((a,b),\sigma')\,.$$

However from the definition of $\widetilde{\mathbf{L}}^{\mathrm{PD}}$ it is easy to verify that the columns corresponding to these two permutations have identical entries in all rows except for the row corresponding to the graph (a,b), giving

$$\begin{aligned}\widetilde{\ell}^{\mathrm{PD}}((i,j),\sigma) &= \widetilde{\ell}^{\mathrm{PD}}((i,j),\sigma') \qquad \forall i<j, (i,j)\neq(a,b)\,;\\ \widetilde{\ell}^{\mathrm{PD}}((a,b),\sigma) &\neq \widetilde{\ell}^{\mathrm{PD}}((a,b),\sigma')\,.\end{aligned}$$

This yields a contradiction, and therefore we must have that the $\frac{r(r-1)}{2}$ rows above are linearly independent, giving

$$\mathrm{rank}(\widetilde{\mathbf{L}}^{\mathrm{PD}}) \geq \frac{r(r-1)}{2}\,.$$

The claim follows. ■

7.5 Proof of Proposition 21

Proof From Eq. (12), we have that $\mathbf{L}^{\mathrm{MAP}} \in \mathbb{R}^{(2^r-1)\times r!}$ can be written as

$$\mathbf{L}^{\mathrm{MAP}} = \mathbf{e}_{(2^r-1)}\mathbf{e}_{r!}^\top - \mathbf{A}\mathbf{B}\,,$$

where $\mathbf{e}_{(2^r-1)}$ and $\mathbf{e}_{r!}$ denote the $(2^r-1)\times 1$ and $r!\times 1$ all ones vectors, respectively, and where $\mathbf{A}\in\mathbb{R}^{(2^r-1)\times\frac{r(r+1)}{2}}$ and $\mathbf{B}\in\mathbb{R}^{\frac{r(r+1)}{2}\times r!}$ are given by

$$\begin{aligned}A_{\mathbf{y},(i,j)} &= \frac{1}{\|\mathbf{y}\|_1}y_i y_j \qquad \forall \mathbf{y}\in\{0,1\}^r\setminus\{\mathbf{0}\},\; i,j\in[r]: i\leq j\,,\\ B_{(i,j),\sigma} &= \frac{1}{\max(\sigma(i),\sigma(j))} \qquad \forall i,j\in[r]: i\leq j,\; \sigma\in\Pi_r\,.\end{aligned}$$

We will show that

$$\mathrm{rank}(\mathbf{A}) \;\geq\; \frac{r(r+1)}{2}-1 \tag{30}$$

and

$$\mathrm{rank}(\mathbf{B}) \;\geq\; \frac{r(r-1)}{2}\,. \tag{31}$$

The result will then follow, since we will then have

$$\begin{aligned}\mathrm{rank}(\mathbf{L}^{\mathrm{MAP}}) &= \mathrm{rank}\left(\mathbf{e}_{(2^r-1)}\mathbf{e}_{r!}^\top - \mathbf{A}\mathbf{B}\right)\\ &\geq \mathrm{rank}(\mathbf{A}\mathbf{B}) - 1\\ &\geq \mathrm{rank}(\mathbf{B}) - 2\,, \quad \text{since } \mathbf{A} \text{ is away from full (column) rank by at most 1}\\ &\geq \frac{r(r-1)}{2} - 2\,.\end{aligned}$$

To see why Eq. (30) is true, consider the 2^r vectors $\mathbf{v}^\alpha\in\mathbb{R}^{2^r}$ defined as

$$v^\alpha_{\mathbf{y}} = \prod_{i\in\alpha} y_i \quad \forall \alpha\subseteq[r]\,, \mathbf{y}\in\{0,1\}^r\,.$$

It is easy to see that these vectors form a basis in $\mathbb{R}^{2^r}$. The columns of $\mathbf{A}$ can be obtained from the $\frac{r(r+1)}{2}$ vectors $\mathbf{v}^\alpha$ corresponding to subsets $\alpha\subseteq[r]$ of sizes 1 and 2, by removing the element corresponding to $\mathbf{y}=\mathbf{0}$ and dividing all other rows corresponding to $\mathbf{y}\in\{0,1\}^r\setminus\{\mathbf{0}\}$ by $\|\mathbf{y}\|_1$. This establishes the lower bound on $\mathrm{rank}(\mathbf{A})$ in Eq. (30).

To see why Eq. (31) is true, let us make the dependence of $\mathbf{B}$ on r explicit by denoting $\mathbf{B}_r=\mathbf{B}$, and observe that $\mathbf{B}_r$ can be decomposed as

$$\mathbf{B}_r = \begin{bmatrix}\mathbf{B}_{r-1} & \mathbf{D}\\ \mathbf{C} & \mathbf{E}\end{bmatrix}\,,$$

where the sub-matrix $\mathbf{B}_{r-1}\in\mathbb{R}^{\frac{r(r-1)}{2}\times(r-1)!}$ is obtained by taking the $\frac{r(r-1)}{2}$ rows (i,j) in $\mathbf{B}_r$ with $i\leq j<r$ and the $(r-1)!$ columns σ in $\mathbf{B}_r$ with $\sigma(r)=r$:

	$\Upsilon=\{\sigma\in\Pi_r:\sigma(r)=r\}$	$\Omega=\{\sigma\in\Pi_r:\sigma(r)\neq r\}$
$\Gamma=\{(i,j)\in[r]\times[r]: i\leq j<r\}$	$\mathbf{B}_{r-1}$	$\mathbf{D}$
$\Lambda=\{(i,j)\in[r]\times[r]: i\leq j=r\}$	$\mathbf{C}$	$\mathbf{E}$

Consider the matrix $\mathbf{C}\in\mathbb{R}^{r\times(r-1)!}$. Each entry in this matrix has the form $\frac{1}{\max(\sigma(i),\sigma(j))}$ with $i\leq j=r$ and $\sigma(r)=r$. Thus all entries in $\mathbf{C}$ are equal to $\frac{1}{r}$ and $\mathrm{rank}(\mathbf{C})=1$.

Next, consider the matrix $\mathbf{E}\in\mathbb{R}^{r\times(r!-(r-1)!)}$. We will show that there are $r-1$ linearly independent columns in $\mathbf{E}$. In particular, consider any permutations $\sigma^1,\sigma^2,\ldots,\sigma^{r-1}$ in the set Ω such that $\sigma^j(j)=r$ and $\sigma^j(r)=r-1$ (such permutations clearly exist). The sub-matrix of $\mathbf{E}$ corresponding to these columns is given by

	$\sigma^1(1)=r$ $\sigma^1(r)=r-1$	$\sigma^2(2)=r$ $\sigma^2(r)=r-1$	$\ldots$ $\ldots$	$\sigma^{r-1}(r-1)=r$ $\sigma^{r-1}(r)=r-1$
$(1,r)$	$\frac{1}{r}$	$\frac{1}{r-1}$	$\cdots$	$\frac{1}{r-1}$
$(2,r)$	$\frac{1}{r-1}$	$\frac{1}{r}$	$\cdots$	$\frac{1}{r-1}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$(r-1,r)$	$\frac{1}{r-1}$	$\frac{1}{r-1}$	$\cdots$	$\frac{1}{r}$
(r,r)	$\frac{1}{r-1}$	$\frac{1}{r-1}$	$\cdots$	$\frac{1}{r-1}$

Thus excluding the last row of $\mathbf{E}$, one gets a square $(r-1)\times(r-1)$ matrix with diagonal entries equal to $\frac{1}{r}$ and off-diagonal entries equal to $\frac{1}{r-1}$. The last row of $\mathbf{E}$ has all entries equal to $\frac{1}{r-1}$. Clearly, this gives $\mathrm{rank}(\mathbf{E})=r-1$. Moreover, the span of the $r-1$ column vectors of $\mathbf{E}$ does not intersect with column space of $\mathbf{C}$ non-trivially, since it does not contain the all ones vector. This implies that the $r-1$ columns of $\mathbf{B}_r$ corresponding to the permutations $\sigma^1,\sigma^2,\ldots,\sigma^{r-1}\in\Omega$ (which yield the linearly independent columns of $\mathbf{E}$),

together with the columns of $\mathbf{B}_r$ corresponding to permutations $\sigma \in \Upsilon$ that yield linearly independent columns of $\mathbf{B}_{r-1}$, are all linearly independent. Therefore, we have

$$\text{rank}(\mathbf{B}_r) \;\geq\; \text{rank}(\mathbf{B}_{r-1}) + r - 1\,.$$

Trivially, $\text{rank}(\mathbf{B}_1) \geq 0$. Expanding the above recursion therefore gives

$$\text{rank}(\mathbf{B}_r) \;\geq\; \frac{r(r-1)}{2}\,.$$

This establishes the lower bound on $\text{rank}(\mathbf{B})$ in Eq. (31). ■

Acknowledgments

HGR is supported by a TCS PhD Fellowship. SA thanks the Department of Science and Technology (DST) of the Government of India for a Ramanujan Fellowship, and the Indo-US Science and Technology Forum (IUSSTF) for their support.

Appendix A. Proofs of Lemma 2 and Theorem 3

A.1 Proof of Lemma 2

Proof Let $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+$. We will show that $\exists$ pred : $\mathcal{C} \to [k]$ satisfying the condition in Definition 1 if and only if $\exists$ pred$' : \mathcal{S}_\psi \to [k]$ satisfying the stated condition.

('if' direction) First, suppose $\exists$ pred$' : \mathcal{S}_\psi \to [k]$ such that

$$\forall \mathbf{p} \in \mathcal{P} : \quad \inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}'(\mathbf{z}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \mathbf{z} \;>\; \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}\,.$$

Define pred : $\mathcal{C} \to [k]$ as follows:

$$\text{pred}(\mathbf{u}) = \text{pred}'(\boldsymbol{\psi}(\mathbf{u})) \quad \forall \mathbf{u} \in \mathcal{C}\,.$$

Then for all $\mathbf{p} \in \mathcal{P}$, we have

$$\begin{aligned}
\inf_{\mathbf{u} \in \mathcal{C} : \text{pred}(\mathbf{u}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) &= \inf_{\mathbf{z} \in \mathcal{R}_\psi : \text{pred}'(\mathbf{z}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \mathbf{z} \\
&\geq \inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}'(\mathbf{z}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \mathbf{z} \\
&> \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} \\
&= \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})\,.
\end{aligned}$$

Thus $\boldsymbol{\psi}$ is $(\mathbf{L}, \mathcal{P})$-calibrated.

('only if' direction) Conversely, suppose $\boldsymbol{\psi}$ is $(\mathbf{L}, \mathcal{P})$-calibrated, so that $\exists$ pred : $\mathcal{C} \to [k]$ such that

$$\forall \mathbf{p} \in \mathcal{P} : \quad \inf_{\mathbf{u} \in \mathcal{C} : \text{pred}(\mathbf{u}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \;>\; \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})\,.$$

By Caratheodory's Theorem (e.g. see Bertsekas et al. (2003)), we have that every $\mathbf{z} \in \mathcal{S}_\psi$ can be expressed as a convex combination of at most $n+1$ points in $\mathcal{R}_\psi$, i.e. for every $\mathbf{z} \in \mathcal{S}_\psi$, $\exists \boldsymbol{\alpha} \in \Delta_{n+1}, \mathbf{u}_1, \ldots, \mathbf{u}_{n+1} \in \mathcal{C}$ such that $\mathbf{z} = \sum_{j=1}^{n+1} \alpha_j \boldsymbol{\psi}(\mathbf{u}_j)$; w.l.o.g., we can assume $\alpha_1 \geq \frac{1}{n+1}$. For each $\mathbf{z} \in \mathcal{S}_\psi$, arbitrarily fix a unique such convex combination, i.e. fix $\boldsymbol{\alpha}^{\mathbf{z}} \in \Delta_{n+1}, \mathbf{u}^{\mathbf{z}}_1, \ldots, \mathbf{u}^{\mathbf{z}}_{n+1} \in \mathcal{C}$ with $\alpha^{\mathbf{z}}_1 \geq \frac{1}{n+1}$ such that

$$\mathbf{z} = \sum_{j=1}^{n+1} \alpha^{\mathbf{z}}_j \boldsymbol{\psi}(\mathbf{u}^{\mathbf{z}}_j)\,.$$

Now, define pred$' : \mathcal{S}_\psi \to [k]$ as follows:

$$\text{pred}'(\mathbf{z}) = \text{pred}(\mathbf{u}^{\mathbf{z}}_1) \quad \forall \mathbf{z} \in \mathcal{S}_\psi\,.$$

Then for any $\mathbf{p} \in \mathcal{P}$, we have

$$\begin{aligned}
\inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}'(\mathbf{z}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \mathbf{z} &= \inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}(\mathbf{u}^{\mathbf{z}}_1) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \sum_{j=1}^{n+1} \alpha^{\mathbf{z}}_j \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}^{\mathbf{z}}_j) \\
&\geq \inf_{\boldsymbol{\alpha} \in \Delta_{n+1}, \mathbf{u}_1, \ldots, \mathbf{u}_{n+1} \in \mathcal{C} : \alpha_1 \geq \frac{1}{n+1}, \text{pred}(\mathbf{u}_1) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \sum_{j=1}^{n+1} \alpha_j \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_j) \\
&\geq \inf_{\boldsymbol{\alpha} \in \Delta_{n+1} : \alpha_1 \geq \frac{1}{n+1}} \sum_{j=1}^{n+1} \inf_{\mathbf{u}_j \in \mathcal{C} : \text{pred}(\mathbf{u}_1) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \alpha_j \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_j) \\
&\geq \inf_{\alpha_1 \in [\frac{1}{n+1}, 1]} \alpha_1 \inf_{\mathbf{u} \in \mathcal{C} : \text{pred}(\mathbf{u}) \notin \text{argmin}_t \, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) + (1 - \alpha_1) \sum_{j=2}^{n+1} \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \\
&> \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \\
&= \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}\,.
\end{aligned}$$

Thus pred$'$ satisfies the stated condition. ■

A.2 Proof of Theorem 3

The proof is similar to that for the multiclass 0-1 loss given by Tewari and Bartlett (2007). We will make use of the following two lemmas; the first is a straightforward generalization of a similar lemma in (Tewari and Bartlett, 2007), and the second follows directly from Lemma 2.

Lemma 24 *The map* $\mathbf{p} \mapsto \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}$ *is continuous over* Δ_n.

Lemma 25 *Let* $\mathbf{L} \in \mathbb{R}^{n \times k}_+$. *A surrogate* $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n$ *is* $\mathbf{L}$*-calibrated if and only if there exists a function* pred$' : \mathcal{S}_\psi \to [k]$ *such that the following holds: for all* $\mathbf{p} \in \Delta_n$ *and all sequences* $\{\mathbf{z}_m\}$ *in* $\mathcal{S}_\psi$ *such that* $\lim_{m \to \infty} \mathbf{p}^\top \mathbf{z}_m = \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}$, *we have* $\mathbf{p}^\top \boldsymbol{\ell}_{\text{pred}'(\mathbf{z}_m)} = \min_{t \in [k]} \mathbf{p}^\top \boldsymbol{\ell}_t$ *for all large enough* m.

Proof (Proof of Theorem 3)

Let $\boldsymbol{\psi} : \mathcal{C} \to \mathbb{R}^n_+$.

('only if' direction) First, suppose $\boldsymbol{\psi}$ is $\mathbf{L}$-calibrated. Then by Lemma 2, $\exists\ \text{pred}' : \mathcal{S}_\psi \to [k]$ such that

$$\forall \mathbf{p} \in \mathcal{P} : \quad \inf_{\mathbf{z} \in \mathcal{S}_\psi : \text{pred}'(\mathbf{z}) \notin \text{argmin}_t\, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \mathbf{z} \;>\; \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}\,.$$

Now, for each $\epsilon > 0$, define

$$H(\epsilon) = \inf_{\mathbf{p} \in \Delta_n, \mathbf{z} \in \mathcal{S}_\psi : \mathbf{p}^\top \boldsymbol{\ell}_{\text{pred}'(\mathbf{z})} - \min_{t \in [k]} \mathbf{p}^\top \boldsymbol{\ell}_t \geq \epsilon} \left\{ \mathbf{p}^\top \mathbf{z} - \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z} \right\}.$$

We claim that $H(\epsilon) > 0\ \forall \epsilon > 0$. Assume for the sake of contradiction that $\exists \epsilon > 0$ for which $H(\epsilon) = 0$. Then there must exist a sequence $(\mathbf{p}_m, \mathbf{z}_m)$ in $\Delta_n \times \mathcal{S}_\psi$ such that

$$\mathbf{p}_m{}^\top \boldsymbol{\ell}_{\text{pred}'(\mathbf{z}_m)} - \min_{t \in [k]} \mathbf{p}_m{}^\top \boldsymbol{\ell}_t \geq \epsilon \quad \forall m \tag{32}$$

and

$$\mathbf{p}_m{}^\top \mathbf{z}_m - \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}_m{}^\top \mathbf{z} \to 0\,. \tag{33}$$

Since $\mathbf{p}_m$ come from a compact set, we can choose a convergent subsequence (which we still call $\{\mathbf{p}_m\}$), say with limit $\mathbf{p}$. Then by Lemma 24, we have $\inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}_m{}^\top \mathbf{z} \longrightarrow \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}$, and therefore by Eq. (33), we get

$$\mathbf{p}_m{}^\top \mathbf{z}_m \;\longrightarrow\; \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}\,.$$

Now we show that $\mathbf{z}_m$ is a sequence such that $\mathbf{p}^\top \mathbf{z}_m \longrightarrow \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}$. Without loss of generality, we assume that the first a coordinates of $\mathbf{p}$ are non-zero and the rest are zero. Hence the first a coordinates of $\mathbf{z}_m$ are bounded for sufficiently large m, and we have

$$\limsup_m \mathbf{p}^\top \mathbf{z}_m = \limsup_m \sum_{y=1}^{a} p_{m,y} z_{m,y} \leq \lim_{m \to \infty} \mathbf{p}_m{}^\top \mathbf{z}_m = \inf_{\mathbf{z} \in \mathcal{S}_\psi} \mathbf{p}^\top \mathbf{z}\,.$$

By Lemma 25, we therefore have $\mathbf{p}^\top \boldsymbol{\ell}_{\text{pred}'(\mathbf{z}_m)} = \min_{t \in [k]} \mathbf{p}^\top \boldsymbol{\ell}_t$ for all large enough m, which contradicts Eq. (32) as $\mathbf{p}_m$ converges to $\mathbf{p}$. Thus we must have $H(\epsilon) > 0\ \forall \epsilon > 0$; the rest of the proof then follows from (Zhang, 2004a).

('if' direction)

Conversely, suppose $\boldsymbol{\psi}$ is not $\mathbf{L}$-calibrated. Consider any $\text{pred} : \mathcal{C} \to [k]$. Then $\exists \mathbf{p} \in \Delta_n$ such that

$$\inf_{\mathbf{u} \in \mathcal{C} : \text{pred}(\mathbf{u}) \notin \text{argmin}_t\, \mathbf{p}^\top \boldsymbol{\ell}_t} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}) \;=\; \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})\,.$$

In particular, this means there exists a sequence of points $\{\mathbf{u}_m\}$ in $\mathcal{C}$ such that

$$\text{pred}(\mathbf{u}_m) \notin \text{argmin}_t\, \mathbf{p}^\top \boldsymbol{\ell}_t \quad \forall m$$

and

$$\mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_m) \;\longrightarrow\; \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})\,.$$

Now consider a data distribution $D = D_X \times D_{Y|X}$ on $\mathcal{X} \times [n]$, with D_X being a point mass at some $x \in \mathcal{X}$ and $D_{Y|X=x} = \mathbf{p}$. Let $\mathbf{f}_m : \mathcal{X} \to \mathcal{C}$ be any sequence of functions satisfying $\mathbf{f}_m(x) = \mathbf{u}_m\ \forall m$. Then we have

$$\text{er}^{\boldsymbol{\psi}}_D[\mathbf{f}_m] = \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u}_m)\,; \quad \text{er}^{\boldsymbol{\psi},*}_D = \inf_{\mathbf{u} \in \mathcal{C}} \mathbf{p}^\top \boldsymbol{\psi}(\mathbf{u})$$

and

$$\text{er}^{\mathbf{L}}_D[\text{pred} \circ \mathbf{f}_m] = \mathbf{p}^\top \boldsymbol{\ell}_{\text{pred}(\mathbf{u}_m)}\,; \quad \text{er}^{\mathbf{L},*}_D = \min_t \mathbf{p}^\top \boldsymbol{\ell}_t\,.$$

This gives

$$\text{er}^{\boldsymbol{\psi}}_D[\mathbf{f}_m] \;\longrightarrow\; \text{er}^{\boldsymbol{\psi},*}_D$$

but

$$\text{er}^{\mathbf{L}}_D[\text{pred} \circ \mathbf{f}_m] \;\not\longrightarrow\; \text{er}^{\mathbf{L},*}_D\,.$$

This completes the proof. ■

Appendix B. Calculation of Trigger Probability Sets for Figure 2

(a) 0-1 loss $\ell^{\text{0-1}}$ $(n = 3)$.

$$\boldsymbol{\ell}_1 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}\,; \quad \boldsymbol{\ell}_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}\,; \quad \boldsymbol{\ell}_3 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}\,.$$

$$\begin{aligned} \mathcal{Q}^{\text{0-1}}_1 &= \{\mathbf{p} \in \Delta_3 : \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_2,\ \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_3\} \\ &= \{\mathbf{p} \in \Delta_3 : p_2 + p_3 \leq p_1 + p_3,\ p_2 + p_3 \leq p_1 + p_2\} \\ &= \{\mathbf{p} \in \Delta_3 : p_2 \leq p_1,\ p_3 \leq p_1\} \\ &= \{\mathbf{p} \in \Delta_3 : p_1 \geq \max(p_2, p_3)\} \end{aligned}$$

By symmetry,

$$\begin{aligned} \mathcal{Q}^{\text{0-1}}_2 &= \{\mathbf{p} \in \Delta_3 : p_2 \geq \max(p_1, p_3)\} \\ \mathcal{Q}^{\text{0-1}}_3 &= \{\mathbf{p} \in \Delta_3 : p_3 \geq \max(p_1, p_2)\} \end{aligned}$$

(b) Ordinal regression loss ℓ^{ord} $(n = 3)$.

$$\boldsymbol{\ell}_1 = \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix}\,; \quad \boldsymbol{\ell}_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}\,; \quad \boldsymbol{\ell}_3 = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}\,.$$

$$\begin{aligned}
\mathcal{Q}_1^{\text{ord}} &= \{\mathbf{p} \in \Delta_3 : \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_2,\ \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_3\} \\
&= \{\mathbf{p} \in \Delta_3 : p_2 + 2p_3 \leq p_1 + p_3,\ p_2 + 2p_3 \leq 2p_1 + p_2\} \\
&= \{\mathbf{p} \in \Delta_3 : p_2 + p_3 \leq p_1,\ p_3 \leq p_1\} \\
&= \{\mathbf{p} \in \Delta_3 : 1 - p_1 \leq p_1\} \\
&= \{\mathbf{p} \in \Delta_3 : p_1 \geq \tfrac{1}{2}\}
\end{aligned}$$

By symmetry,

$$\mathcal{Q}_3^{\text{ord}} = \{\mathbf{p} \in \Delta_3 : p_3 \geq \tfrac{1}{2}\}$$

Finally,

$$\begin{aligned}
\mathcal{Q}_2^{\text{ord}} &= \{\mathbf{p} \in \Delta_3 : \mathbf{p}^\top \boldsymbol{\ell}_2 \leq \mathbf{p}^\top \boldsymbol{\ell}_1,\ \mathbf{p}^\top \boldsymbol{\ell}_2 \leq \mathbf{p}^\top \boldsymbol{\ell}_3\} \\
&= \{\mathbf{p} \in \Delta_3 : p_1 + p_3 \leq p_2 + 2p_3,\ p_1 + p_3 \leq 2p_1 + p_2\} \\
&= \{\mathbf{p} \in \Delta_3 : p_1 \leq p_2 + p_3,\ p_3 \leq p_1 + p_2\} \\
&= \{\mathbf{p} \in \Delta_3 : p_1 \leq 1 - p_1,\ p_3 \leq 1 - p_3\} \\
&= \{\mathbf{p} \in \Delta_3 : p_1 \leq \tfrac{1}{2},\ p_3 \leq \tfrac{1}{2}\}
\end{aligned}$$

(c) 'Abstain' loss $\ell^{(?)}$ ($n = 3$).

$$\boldsymbol{\ell}_1 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix};\ \ \boldsymbol{\ell}_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix};\ \ \boldsymbol{\ell}_3 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix};\ \ \boldsymbol{\ell}_4 = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}.$$

$$\begin{aligned}
\mathcal{Q}_1^{(?)} &= \{\mathbf{p} \in \Delta_3 : \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_2,\ \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_3,\ \mathbf{p}^\top \boldsymbol{\ell}_1 \leq \mathbf{p}^\top \boldsymbol{\ell}_4\} \\
&= \{\mathbf{p} \in \Delta_3 : p_2 + p_3 \leq p_1 + p_3,\ p_2 + p_3 \leq p_1 + p_2,\ p_2 + p_3 \leq \tfrac{1}{2}(p_1 + p_2 + p_3)\} \\
&= \{\mathbf{p} \in \Delta_3 : p_2 \leq p_1,\ p_3 \leq p_1,\ p_2 + p_3 \leq \tfrac{1}{2}\} \\
&= \{\mathbf{p} \in \Delta_3 : p_1 \geq \tfrac{1}{2}\}
\end{aligned}$$

By symmetry,

$$\begin{aligned}
\mathcal{Q}_2^{(?)} &= \{\mathbf{p} \in \Delta_3 : p_2 \geq \tfrac{1}{2}\} \\
\mathcal{Q}_3^{(?)} &= \{\mathbf{p} \in \Delta_3 : p_3 \geq \tfrac{1}{2}\}
\end{aligned}$$

Finally,

$$\begin{aligned}
\mathcal{Q}_4^{(?)} &= \{\mathbf{p} \in \Delta_3 : \mathbf{p}^\top \boldsymbol{\ell}_4 \leq \mathbf{p}^\top \boldsymbol{\ell}_1,\ \mathbf{p}^\top \boldsymbol{\ell}_4 \leq \mathbf{p}^\top \boldsymbol{\ell}_2,\ \mathbf{p}^\top \boldsymbol{\ell}_4 \leq \mathbf{p}^\top \boldsymbol{\ell}_2\} \\
&= \{\mathbf{p} \in \Delta_3 : \tfrac{1}{2}(p_1 + p_2 + p_3) \leq \min(p_2 + p_3, p_1 + p_3, p_1 + p_2)\} \\
&= \{\mathbf{p} \in \Delta_3 : \tfrac{1}{2} \leq 1 - \max(p_1, p_2, p_3)\} \\
&= \{\mathbf{p} \in \Delta_3 : \max(p_1, p_2, p_3) \leq \tfrac{1}{2}\}
\end{aligned}$$

Appendix C. Calculation of Positive Normal Sets for Figure 3

For $n = 3$, the Crammer-Singer surrogate $\boldsymbol{\psi}^{\text{CS}} : \mathbb{R}^3 \to \mathbb{R}^3_+$ is given by

$$\begin{aligned}
\psi_1^{\text{CS}}(\mathbf{u}) &= \max(1 + u_2 - u_1, 1 + u_3 - u_1, 0) \\
\psi_2^{\text{CS}}(\mathbf{u}) &= \max(1 + u_1 - u_2, 1 + u_3 - u_2, 0) \\
\psi_3^{\text{CS}}(\mathbf{u}) &= \max(1 + u_1 - u_3, 1 + u_2 - u_3, 0) \quad \forall \mathbf{u} \in \mathbb{R}^3.
\end{aligned}$$

Clearly, $\boldsymbol{\psi}^{\text{CS}}$ is a convex function. Let $\mathbf{u}_1 = (1,0,0)^\top$, $\mathbf{u}_2 = (0,1,0)^\top$, $\mathbf{u}_3 = (0,0,1)^\top$, $\mathbf{u}_4 = (0,0,0)^\top$, and let

$$\begin{aligned}
\mathbf{z}_1 &= \boldsymbol{\psi}^{\text{CS}}(\mathbf{u}_1) = (0,2,2)^\top \\
\mathbf{z}_2 &= \boldsymbol{\psi}^{\text{CS}}(\mathbf{u}_2) = (2,0,2)^\top \\
\mathbf{z}_3 &= \boldsymbol{\psi}^{\text{CS}}(\mathbf{u}_3) = (2,2,0)^\top \\
\mathbf{z}_4 &= \boldsymbol{\psi}^{\text{CS}}(\mathbf{u}_4) = (1,1,1)^\top.
\end{aligned}$$

We apply Lemma 9 to compute the positive normal sets of $\boldsymbol{\psi}^{\text{CS}}$ at the 4 points $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4$ above. In particular, to see that $\mathbf{z}_4$ satisfies the conditions of Lemma 9, note that by Danskin's Theorem (Bertsekas et al., 2003), we have that

$$\partial \psi_1^{\text{CS}}(\mathbf{u}_4) = \text{conv}\left(\begin{bmatrix}-1\\+1\\0\end{bmatrix}, \begin{bmatrix}-1\\0\\+1\end{bmatrix}\right);$$

$$\partial \psi_2^{\text{CS}}(\mathbf{u}_4) = \text{conv}\left(\begin{bmatrix}+1\\-1\\0\end{bmatrix}, \begin{bmatrix}0\\-1\\+1\end{bmatrix}\right);$$

$$\partial \psi_3^{\text{CS}}(\mathbf{u}_4) = \text{conv}\left(\begin{bmatrix}+1\\0\\-1\end{bmatrix}, \begin{bmatrix}0\\+1\\-1\end{bmatrix}\right).$$

We can therefore use Lemma 9 to compute $\mathcal{N}^{\text{CS}}(\mathbf{z}_4)$. Here $s = 6$, and

$$\mathbf{A} = \begin{bmatrix} -1 & -1 & 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & -1 & 0 & 1 \\ 0 & 1 & 0 & 1 & -1 & -1 \end{bmatrix};\quad \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

By Lemma 9 (and some algebra), this gives

$$\begin{aligned}
\mathcal{N}^{\text{CS}}(\mathbf{z}_4) &= \big\{\mathbf{p} \in \Delta_3 : \mathbf{p} = (q_1 + q_2, q_3 + q_4, q_5 + q_6) \text{ for some } \mathbf{q} \in \Delta_6, \\
&\qquad q_1 + q_2 = q_3 + q_5,\ q_3 + q_4 = q_1 + q_6,\ q_5 + q_6 = q_2 + q_4\big\} \\
&= \big\{\mathbf{p} \in \Delta_3 : p_1 \leq \tfrac{1}{2}, p_2 \leq \tfrac{1}{2}, p_3 \leq \tfrac{1}{2}\big\}.
\end{aligned}$$

It is easy to see that $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3$ also satisfy the conditions of Lemma 9; similar computations then yield

$$\begin{aligned}
\mathcal{N}^{\text{CS}}(\mathbf{z}_1) &= \big\{\mathbf{p} \in \Delta_3 : p_1 \geq \tfrac{1}{2}\big\} \\
\mathcal{N}^{\text{CS}}(\mathbf{z}_2) &= \big\{\mathbf{p} \in \Delta_3 : p_2 \geq \tfrac{1}{2}\big\} \\
\mathcal{N}^{\text{CS}}(\mathbf{z}_3) &= \big\{\mathbf{p} \in \Delta_3 : p_3 \geq \tfrac{1}{2}\big\}.
\end{aligned}$$

References

Shivani Agarwal. Surrogate regret bounds for bipartite ranking via strongly proper losses. *Journal of Machine Learning Research*, 15:1653–1674, 2014.

Peter L. Bartlett, Michael Jordan, and Jon McAuliffe. Convexity, classification and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Dimitri Bertsekas, Angelia Nedic, and Asuman Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

David Buffoni, Clément Calauzènes, Patrick Gallinari, and Nicolas Usunier. Learning scoring functions with order-preserving losses and standardized supervision. In *International Conference on Machine Learning*, 2011.

Clément Calauzènes, Nicolas Usunier, and Patrick Gallinari. On the (non-)existence of convex, calibrated surrogate losses for ranking. In *Neural Information Processing Systems*, 2012.

Stéphan Clémençon and Nicolas Vayatis. Ranking the best instances. *Journal of Machine Learning Research*, 8:2671–2699, 2007.

Stéphan Clémençon, Gábor Lugosi, and Nicolas Vayatis. Ranking and empirical minimization of U-statistics. *Annals of Statistics*, 36:844–874, 2008.

David Cossock and Tong Zhang. Statistical analysis of Bayes optimal subset ranking. *IEEE Transactions on Information Theory*, 54(11):5140–5154, 2008.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

John Duchi, Lester Mackey, and Michael Jordan. On the consistency of ranking algorithms. In *International Conference on Machine Learning*, 2010.

Jean Gallier. Notes on convex sets, polytopes, polyhedra, combinatorial topology, Voronoi diagrams and Delaunay triangulations. Technical report, Department of Computer and Information Science, University of Pennsylvania, 2009.

Wei Gao and Zhi-Hua Zhou. On the consistency of multi-label learning. In *Conference on Learning Theory*, 2011.

Kalervo Järvelin and Jaana Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2000.

Wenxin Jiang. Process consistency for AdaBoost. *Annals of Statistics*, 32(1):13–29, 2004.

Wojciech Kotlowski, Krzysztof Dembczynski, and Eyke Huellermeier. Bipartite ranking through minimization of univariate loss. In *International Conference on Machine Learning*, 2011.

Nicolas Lambert and Yoav Shoham. Eliciting truthful answers to multiple-choice questions. In *ACM Conference on Electronic Commerce*, 2009.

Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

Gábor Lugosi and Nicolas Vayatis. On the Bayes-risk consistency of regularized boosting methods. *Annals of Statistics*, 32(1):30–55, 2004.

Deirdre O'Brien, Maya Gupta, and Robert Gray. Cost-sensitive multi-class classification from probability estimates. In *International Conference on Machine Learning*, 2008.

Bernardo Á. Pires, Csaba Szepesvari, and Mohammad Ghavamzadeh. Cost-sensitive multiclass classification risk bounds. In *International Conference on Machine Learning*, 2013.

Harish G. Ramaswamy and Shivani Agarwal. Classification calibration dimension for general multiclass losses. In *Neural Information Processing Systems*, 2012.

Harish G. Ramaswamy, Shivani Agarwal, and Ambuj Tewari. Convex calibrated surrogates for low-rank loss matrices with applications to subset ranking losses. In *Neural Information Processing Systems*, 2013.

Harish G. Ramaswamy, Ambuj Tewari, and Shivani Agarwal. Consistent algorithms for multiclass classification with a reject option. arXiv:1505.04137, 2015.

Pradeep Ravikumar, Ambuj Tewari, and Eunho Yang. On NDCG consistency of listwise ranking methods. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Mark D. Reid and Robert C. Williamson. Composite binary losses. *Journal of Machine Learning Research*, 11:2387–2422, 2010.

Clayton Scott. Calibrated asymmetric surrogate losses. *Electronic Journal of Statistics*, 6: 958–992, 2012.

Ingo Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

Ingo Steinwart. How to compare different loss functions and their risks. *Constructive Approximation*, 26:225–287, 2007.

Ambuj Tewari and Peter L. Bartlett. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8:1007–1025, 2007.

Elodie Vernet, Robert C. Williamson, and Mark D. Reid. Composite multiclass losses. In *Neural Information Processing Systems*, 2011.

Jason Weston and Chris Watkins. Support vector machines for multi-class pattern recognition. In *7th European Symposium On Artificial Neural Networks*, 1999.

Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: Theory and algorithm. In *International Conference on Machine Learning*, 2008.

Ming Yuan and Marten Wegkamp. Classification methods with reject option based on convex risk minimization. *Journal of Machine Learning Research*, 11:111–130, 2010.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 32(1):56–134, 2004a.

Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5:1225–1251, 2004b.

Journal of Machine Learning Research 17 (2016) 1-24 Submitted 7/14; Revised 5/15; Published 3/16

LLORMA: Local Low-Rank Matrix Approximation

Joonseok Lee JOONSEOK2010@GMAIL.COM
Seungyeon Kim SEUNGYEONK@GOOGLE.COM
Google Research, Mountain View, CA USA
Guy Lebanon GLEBANON@GMAIL.COM
LinkedIn, Mountain View, CA USA
Yoram Singer SINGER@GOOGLE.COM
Samy Bengio BENGIO@GOOGLE.COM
Google Research, Mountain View, CA USA

Editor: Jeff Bilmes

Abstract

Matrix approximation is a common tool in recommendation systems, text mining, and computer vision. A prevalent assumption in constructing matrix approximations is that the partially observed matrix is low-rank. In this paper, we propose, analyze, and experiment with two procedures, one parallel and the other global, for constructing *local* matrix approximations. The two approaches approximate the observed matrix as a weighted sum of low-rank matrices. These matrices are limited to a local region of the observed matrix. We analyze the accuracy of the proposed local low-rank modeling. Our experiments show improvements in prediction accuracy over classical approaches for recommendation tasks.

Keywords: Matrix approximation, non-parametric methods, kernel smoothing, collaborative Filtering, recommender systems.

1. Introduction

Matrix approximation and completion are prevalent tasks in machine learning. Given few observed matrix entries $\{M_{a_1,b_1}, \ldots, M_{a_m,b_m}\}$, matrix completion constructs a matrix $\hat{M}$ that approximates M at its unobserved entries. Matrix approximation is used heavily in recommendation systems, text processing, computer vision, and bioinformatics. In recommendation systems, for example, the matrix M corresponds to ratings of items (columns) by users (rows). Matrix approximation in this case corresponds to predicting the ratings of all users on all items based on a few observed ratings. In many cases, matrix approximation leads to state-of-the-art models that are used in industrial settings.

In general, the problem of completing a matrix M based on a few observed entries is ill-posed. There are uncountably infinite number of matrices that perfectly agree with the observed entries of M. Therefore, without additional assumptions, selecting or constructing the completion matrix $\hat{M}$ is under-specified and thus ill-defined. A popular assumption is that M is a low-rank matrix, which suggests that it is reasonable to assume that the completed matrix $\hat{M}$ has low-rank. More formally, we approximate a matrix $M \in \mathbb{R}^{n_1 \times n_2}$ by a rank r matrix $\hat{M} = UV^\top$, where $U \in \mathbb{R}^{n_1 \times r}$, $V \in \mathbb{R}^{n_2 \times r}$, and $r \ll \min(n_1, n_2)$. In many real datasets, the low-rank assumption is realistic. Further, low-rank approximations often yield matrices that generalize well to the unobserved entries.

In this paper, we extend low-rank matrix approximation in a way that significantly relaxes the low-rank assumption. Instead of assuming that M can be globally approximated by a low-rank matrix, we assume that M behaves as a low-rank matrix in the vicinity of certain row-column combinations. We therefore construct several low-rank approximations of M, each being accurate in a particular region of the matrix. We express our estimator as a smoothed convex combination of low-rank matrices each of which approximates M in a local region.

The local low-rank assumption can also be motivated as follows. In numerous settings there are a few key latent factors which determine whether a user would like an item or not. In the movie domain such factors may include the main actress, director, released year, genre, and more. However, the number of latent variable is typically limited to no more than twenty. These factors are tacitly learned and automatically constructed as we do not assume that information such as genre or actors are available. For example, if the rank is 5 the ratings given to an item by a user is the inner product of a vector of length 5 which describes the user preferences with a vector of length 5 that describes the item characteristics (these two vectors are the rows of U, V). The same assumption holds in the local low-rank case with the exception that the linear basis underlying the latent factors may change across different groups of users and different types of items.

We use techniques from non-parametric kernel smoothing to achieve two goals. The first goal is to formally develop a notion of local low-rank approximation, and the second is the aggregation of several local models into unified matrix approximation. Standard low-rank matrix approximation techniques achieve consistency in the limit of large data (convergence to the data generating process) assuming that M is low-rank. Our local method achieves consistency without the low-rank assumption. Instead, we require that sufficient number of samples is available in increasingly small neighborhoods. Our analysis mirrors the theory of non-parametric kernel smoothing that was primarily developed for continuous spaces. We also adapt and generalize well-known compressed sensing results to our setting. Our experiments show that local low-rank modeling is significantly more accurate than global low-rank modeling in the context of recommendation systems.

The rest of the paper is organized as follows. We introduce notations and briefly review low-rank matrix approximation in Section 2. In Section 3 we describe our proposed methods. Section 4 provides formal analysis of the proposed approach. Sections 5 and 6 describe in details the two low-rank approximation algorithms. These sections are followed by experimental evaluations described in Sections 7. We then discuss our contribution in the context of related work in Section 8 and conclude with a summary in Section 9.

2. Background: Low-rank matrix approximation

We describe in this section two standard approaches for low-rank matrix approximation (LRMA). We start by establishing the notation used throughout the paper. We denote matrices using upper case letters. The original (partially observed) matrix is denoted by $M \in \mathbb{R}^{n_1 \times n_2}$. A low-rank approximation of M is denoted by $\hat{M} = UV^\top$, where $U \in \mathbb{R}^{n_1 \times r}$, $V \in \mathbb{R}^{n_2 \times r}$, and $r \ll \min(n_1, n_2)$. The set of integers $\{1, \ldots, n\}$ is abbreviated as $[n]$. The

Notation	Explanation
n_1	Number of users.
n_2	Number of items.
m	Number of available ratings.
r	Rank of approximation matrix.
M	Rating matrix, $M \in \mathbb{R}^{n_1 \times n_2}$
U	"Users" profile matrix, $U \in \mathbb{R}^{n_1 \times r}$
V	"Items" profile matrix, $V \in \mathbb{R}^{n_2 \times r}$
Ω	Observed entries of M.
$\mathcal{P}_\Omega(M)$	Projection operator onto observed entries of Ω.
$\hat{\mathcal{T}}(a,b)$	Local approximation of M centered at (a,b).
$\hat{\bar{\mathcal{T}}}(a,b)$	Global approximation of M centered at (a,b).
$A \odot B$	Hadamard product of matrices A and B.
$\|X\|_F$	Frobenius norm of matrix X.
$\|X\|_*$	Nuclear (trace) norm of matrix X.
$\|X\|_\infty$	Sup-norm of matrix X.
$[n]$	Set of natural numbers $\{1, \ldots, n\}$.

Table 1: Summary of Notations and Matrix Operators.

set of indices of the observed entries of M is denoted by $\Omega \stackrel{\text{def}}{=} \{(a_1, b_1), \ldots, (a_m, b_m)\} \subseteq [n_1] \times [n_2]$. The training set is therefore $\{M_{a,b} : (a,b) \in \Omega\}$. Mappings from matrix indices to a matrix space are denoted in calligraphic letters, e.g. $\mathcal{T}$, and are operators of the form $\mathcal{T} : [n_1] \times [n_2] \to \mathbb{R}^{n_1 \times n_2}$. We denote the entry (i,j) of the matrix $\mathcal{T}(a,b)$ as $\mathcal{T}_{i,j}(a,b)$. A projection $\mathcal{P}_\mathtt{A}$ with respect to a set of matrix indices $\mathtt{A}$ is the function $\mathcal{P}_\mathtt{A} : \mathbb{R}^{n_1 \times n_2} \to \mathbb{R}^{n_1 \times n_2}$ defined by

$$[\mathcal{P}_\Omega(M)]_{a,b} \stackrel{\text{def}}{=} \begin{cases} M_{a,b} & (a,b) \in \Omega \\ 0 & \text{otherwise.} \end{cases}$$

We denote by $\odot$ the entry-wise product (also known as the Hadamard or Schur products) of two matrices $[A \odot B]_{i,j} = A_{i,j} B_{i,j}$. We use in this paper three matrix norms:

Frobenius norm $\|X\|_F \stackrel{\text{def}}{=} \sqrt{\sum_i \sum_j X_{i,j}^2}$

Sup-norm $\|X\|_\infty \stackrel{\text{def}}{=} \sup_{i,j} |X_{i,j}|$

Nuclear (trace) norm $\|X\|_* \stackrel{\text{def}}{=} \sum_{i=1}^{r} \sigma_i(X)$

For the nuclear norm $\sigma_i(X)$ is the i'th singular value of X where for symmetric matrices $\|X\|_* = \text{trace}(X)$. Table 1 summarizes notations and matrix operators used throughout the paper.

We describe below two popular approaches for constructing a low-rank approximation $\hat{M}$ of M. The first one, incomplete SVD, is based on minimizing the Frobenius norm of $\mathcal{P}_\mathtt{A}(M - \hat{M})$, while the second one is based on minimizing the nuclear norm of a matrix satisfying constraints constructed from the training set.

A1: Incomplete SVD. The incomplete SVD method constructs a low-rank approximation $\hat{M} = UV^\top$ by solving the problem

$$(U, V) = \arg\min_{U,V} \sum_{(a,b) \in \Omega} ([UV^\top]_{a,b} - M_{a,b})^2 \ , \tag{1}$$

or equivalently

$$\hat{M} = \arg\min_X \|\mathcal{P}_\Omega(X - M)\|_F \ \text{ s.t. } \text{rank}(X) = r. \tag{2}$$

A2: Nuclear norm minimization. An alternative to (2) that originated from the compressed sensing community (Candès and Tao, 2010) is to minimize the nuclear norm of a matrix subject to constraints constructed from the observed entries:

$$\hat{M} = \arg\min_X \|X\|_* \quad \text{s.t.} \quad \|\mathcal{P}_\Omega(X - M)\|_F < \delta \ . \tag{3}$$

Minimizing the nuclear norm $\|X\|_*$ is an effective surrogate for minimizing the rank of X, and solving (3) results in a low-rank matrix $\hat{M} = UV^\top$ that approximates the matrix M. One advantage of *A2* over *A1* is that we do not need to constrain the rank of $\hat{M}$ in advance. Note also that the problem defined by (3), while being convex, may not necessarily scale up easily to large matrices.

3. Local low-rank matrix approximation

In order to facilitate a local low-rank matrix approximation, we need to assume that there exists a metric structure over $[n_1] \times [n_2]$. The distance $d((a,b),(a',b'))$ reflects the similarity between the rows a and a' and columns b and b'. In the case of recommendation systems, for example, $d((a,b),(a',b'))$ expresses the relationship between users a, a' and items b, b'. The distance function may be constructed using the observed ratings $\mathcal{P}_\Omega(M)$ or additional information such as item-item similarity or side information on the users when available. We note that distance between two rows (users) or between two columns (items) is independent of the indices of those rows or columns. As we exchange the order of two rows or columns, the similarity still remains the same. See Section 5 for further details.

In the global matrix factorization setting in Section 2, we assume that the matrix $M \in \mathbb{R}^{n_1 \times n_2}$ has a low-rank structure. In the local setting, however, we assume that the model is characterized by multiple low-rank $n_1 \times n_2$ matrices. Specifically, we assume a mapping $\mathcal{T} : [n_1] \times [n_2] \to \mathbb{R}^{n_1 \times n_2}$ that associates with each row-column combination $[n_1] \times [n_2]$ a low rank matrix that describes the entries of M in its neighborhood (in particular this applies to the observed entries Ω):

$$\mathcal{T} : [n_1] \times [n_2] \to \mathbb{R}^{n_1 \times n_2} \quad \text{where} \quad \mathcal{T}_{a,b}(a,b) = M_{a,b} \ .$$

Without additional assumptions, it is impossible to estimate the mapping $\mathcal{T}$ from a set of $m < n_1 n_2$ observations. We assume, as is often done in non-parametric statistics, that the mapping $\mathcal{T}$ is slowly varying. Since the domain of $\mathcal{T}$ is discrete, the classical definitions of continuity or differentiability are not applicable in our setting. We assume instead that $\mathcal{T}$ is Hölder continuous (see Definition 1 in Section 4).

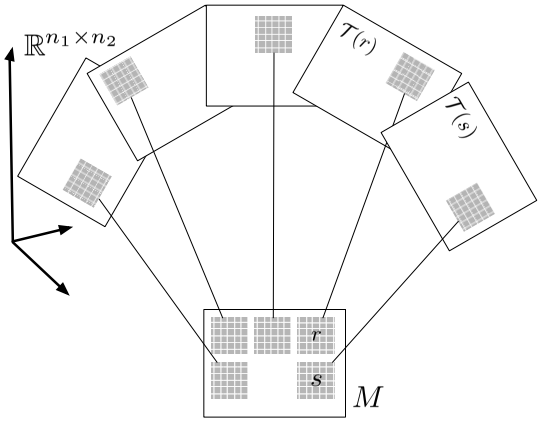


Figure 1: The locally low-rank linear assumption assumes an operator that maps matrix entries to matrices whose image is (a) low-rank, (b) slowly changing, and (c) agrees locally with the original matrix. We emphasize that we draw the figure as if adjacent rows (or columns) are semantically similar just for illustration purpose. In real data, similar users (or items) are usually scattered over the entire space. See text for more details.

Figures 1 shows a graphic illustration of the locally low-rank linear assumption: the operator $\mathcal{T}$ maps matrix entries to matrices whose image is (a) low-rank, (b) slowly changing, and (c) agrees locally with the original matrix. Assumption (b) implies that if $d(s,r)$ is small $\mathcal{T}(s)$ is similar to $\mathcal{T}(r)$, as shown by their spatial closeness in the embedding $\mathbb{R}^{n_1 \times n_2}$. Assumption (c) implies that for all $s \in [n_1] \times [n_2]$, the neighborhood $\{s' : d(s,s') < h\}$ in the original matrix M is approximately described by the corresponding entries of the low-rank matrix $\mathcal{T}(s)$ (shaded regions of M are matched by lines to the corresponding regions in $\mathcal{T}(s)$ that approximate them).

We would like to emphasize that for illustrative purposes, we assume in Figure 1 that there exists a distance function d whose neighborhood structure coincides with the natural order on indices. That is, $s = (a,b)$ is similar to $r = (c,d)$ if $|a-c|$ and $|b-d|$ are small.

Figure 2 shows the relationship between the neighboring entries of the original matrix and the operator image in more detail. The original matrix M (bottom) is described locally by two low-rank matrices $\mathcal{T}(t)$ (near t) and $\mathcal{T}(r)$ (near r). The lines connecting the three matrices identify identical entries: $M_t = \mathcal{T}_t(t)$ and $M_r = \mathcal{T}_r(r)$. The equation at the top right shows a relation tying the three patterned entries. Assuming the distance $d(t,r)$ is small, $\delta = \mathcal{T}_r(t) - \mathcal{T}_r(r) = \mathcal{T}_r(t) - M_r(r)$ is small as well.

Following common approaches in non-parametric statistics, we define a smoothing kernel $K_h(s_1, s_2),\ s_1, s_2 \in [n_1] \times [n_2]$, as a non-negative symmetric unimodal function that is

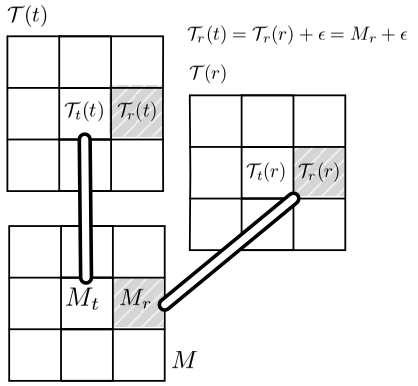


Figure 2: The relationship between the neighboring entries of the original matrix and the operator image. See text for more details.

parameterized by a bandwidth parameter $h > 0$. A large value of h implies that $K_h(s,\cdot)$ has a wide spread, while a small h corresponds to narrow spread of $K_h(s,\cdot)$. Often, it is further assumed that $K_h(x) = K_1(x/h)/h$ and that the kernel integrates to 1: $\int K_h(x)\,dx = 1$. In our case, however, we have a discrete domain rather than a continuous domain. See for instance (Wand and Jones, 1995) for more information on smoothing kernels. Three popular smoothing kernels are the uniform kernel, the triangular kernel, and the Epanechnikov kernel, defined respectively as

$$K_h(s_1,s_2) \propto \mathbf{1}[d(s_1,s_2) < h] \tag{4}$$

$$K_h(s_1,s_2) \propto (1 - h^{-1} d(s_1,s_2))\,\mathbf{1}[d(s_1,s_2) < h] \tag{5}$$

$$K_h(s_1,s_2) \propto (1 - d(s_1,s_2)^2)\,\mathbf{1}[d(s_1,s_2) < h]\ . \tag{6}$$

We denote by $K_h^{(a,b)}$ the matrix whose (i,j)-entry is $K_h((a,b),(i,j))$.

We describe below the local modifications of incomplete SVD (*A1*) and nuclear norm minimization (*A2*) matrix approximations. Both extensions estimate $\mathcal{T}(a,b)$ in the vicinity of $(a,b) \in [n_1] \times [n_2]$ given the samples $\mathcal{P}_\Omega(M)$.

Local-A1: Incomplete SVD

$$\hat{\mathcal{T}}(a,b) = \arg\min_X \|K_h^{(a,b)} \odot \mathcal{P}_\Omega(X - M)\|_F \quad \text{s.t.} \quad \text{rank}(X) = r\ . \tag{7}$$

Local-A2: Nuclear norm minimization

$$\hat{\mathcal{T}}(a,b) = \arg\min_X \|X\|_* \quad \text{s.t.} \quad \|K_h^{(a,b)} \odot \mathcal{P}_\Omega(X - M)\|_F < \delta\ . \tag{8}$$

The two optimization problems above describe how to estimate $\hat{\mathcal{T}}(a,b)$ for a particular choice of $(a,b) \in [n_1] \times [n_2]$. Conceptually, this technique can be applied at each test entry (a,b), resulting in the matrix approximation $\hat{M} \approx M$ where

$$\hat{M}_{a,b} = \hat{\mathcal{T}}_{a,b}(a,b), \quad (a,b) \in [n_1] \times [n_2] \ .$$

However, such a construction would require solving an optimization problem for each matrix entry (a,b) and is thus computationally prohibitive. Instead, we describe in the next subsection how to use a set of q local models $\hat{\mathcal{T}}(s_1), \ldots, \hat{\mathcal{T}}(s_q)$, $s_1, \ldots, s_q \in [n_1] \times [n_2]$ to obtain a computationally efficient estimate $\hat{\hat{\mathcal{T}}}(s)$ for all $s \in [n_1] \times [n_2]$.

3.1 Global Approximation

The problem of recovering a mapping $\mathcal{T}$ from q values without imposing a strong parametric form is known as non-parametric regression. We propose using a variation of locally constant kernel regression (Wand and Jones, 1995), also known as Nadaraya-Watson regression

$$\hat{\hat{\mathcal{T}}}(s) = \sum_{i=1}^{q} \frac{K_h(s_i, s)}{\sum_{j=1}^{q} K_h(s_j, s)} \hat{\mathcal{T}}(s_i) \ . \tag{9}$$

Equation (9) is simply a weighted average of $\hat{\mathcal{T}}(s_1), \ldots, \hat{\mathcal{T}}(s_q)$, where the weights ensure that values of $\hat{\mathcal{T}}$ at indices close to s contribute more than those further away from s. Note that both the left-hand side and the right-hand side of (9) denote matrices. The denominator in (9) ensures that the weights sum to one.

In contrast to $\hat{\mathcal{T}}$, the estimate $\hat{\hat{\mathcal{T}}}$ can be computed for all $s \in [n_1] \times [n_2]$ efficiently since computing $\hat{\hat{\mathcal{T}}}(s)$ simply requires evaluating and averaging $\hat{\mathcal{T}}(s_i)$, $i = 1, \ldots, q$. The resulting matrix approximation is $\hat{\hat{M}}_{a,b} = \hat{\hat{\mathcal{T}}}_{a,b}(a,b)$ and $(a,b) \in [n_1] \times [n_2]$.

The accuracy of $\hat{\hat{\mathcal{T}}}$ as an estimator of $\hat{\mathcal{T}}$ improves with the number of local models q and the degree of continuity of $\hat{\mathcal{T}}$. The accuracy of $\hat{\hat{\mathcal{T}}}$ as an estimator of $\mathcal{T}$ is limited by the quality of the local estimators $\hat{\mathcal{T}}(s_1), \ldots, \hat{\mathcal{T}}(s_q)$. However, assuming that $\hat{\mathcal{T}}(s_1), \ldots, \hat{\mathcal{T}}(s_q)$ are accurate in the neighborhoods of $s_1, \ldots, s_q$, and q is sufficiently large, the estimation error $\hat{\hat{\mathcal{T}}}_{a,b}(a,b) - \mathcal{T}_{a,b}(a,b)$ is likely to be small as we analyze in the next section. We term the resulting approach LLORMA standing for Local LOw Rank Matrix Approximation.

4. Estimation accuracy

In this section we analyze the estimation accuracy of LLORMA. Our analysis consists of two parts. In the first we analyze the large deviation of $\hat{\mathcal{T}}$ from $\mathcal{T}$. Then, based on this analysis, we derive a deviation bound on the global approximation $\hat{\hat{\mathcal{T}}}$. Our analysis technique is based on the seminal paper of Candès and Tao (2010). The goal of this section is to underscore the characteristics of estimation error in terms of parameters such as the train set size, matrix dimensions, and kernel bandwidth.

4.1 Analysis of $\hat{\mathcal{T}} - \mathcal{T}$

Candès and Tao (2010) established that it is possible to estimate an $n_1 \times n_2$ matrix M of rank r if the number of observations $m \geq C\mu r n \log^6 n$, where $n = \min(n_1, n_2)$, C is a

constant, and μ is the strong incoherence property parameter described in Candès and Tao (2010). This bound is tight in the sense that it is close to the information theoretic limit of $\Omega(r\, n \log n)$. As in Candès and Tao (2010), we assume that the observed entries are sampled at random without replacement, avoiding trivial situations in which a row or a column is unsampled.

The aforementioned result is not applicable in our case since the matrix M is not necessarily of low-rank. Concretely, when $r = O(n)$ the bound above degenerates into a sample complexity of $O(n^2 \log n)$ which is clearly larger than the number of entries in the matrix M. We develop below a variation on the results in Candès and Tao (2010) and Candès and Plan (2010) that applies to the *local-A2* compressed-sensing estimator $\hat{\mathcal{T}}$.

Definition 1. Let X be a metric space. A function $f : X \to \mathbb{R}^{n_1 \times n_2}$ is *Hölder continuous* with parameters $\alpha, \beta > 0$ if

$$\forall x, x' \in X : \ \|f(x) - f(x')\|_F \ \leq \ \alpha\, d^\beta(x, x') \ . \tag{10}$$

In our analysis we make the following assumptions: (i) $\mathcal{T}$ is Hölder continuous, (ii) $\mathcal{T}(s)$ is a rank r matrix that satisfies the strong incoherence property, and (iii) the kernel K_h is a uniform kernel based on a product distance function. The Hölder continuity assumption on $\mathcal{T}$ can be replaced by the following weaker condition without affecting the results

$$\|K_h^s \odot (\mathcal{T}(s) - \mathcal{T}(s'))\|_F \ \leq \ \alpha d^\beta(s, s'). \tag{11}$$

We denote by $B_h(s)$ the neighborhood of indices near s, $B_h(s) \stackrel{\text{def}}{=} \{s' \in [n_1] \times [n_2] : d(s,s') < h\}$ and we use $n_1(h,s)$ and $n_2(h,s)$ to denote the number of unique row and column indices, respectively, in $B_h(s)$. Finally, we denote $\gamma = \min(n_1(h,s), n_2(h,s))$.

The proposition below provides a bound on the average squared-error within a neighborhood of s

$$\mathcal{E}(\hat{\mathcal{T}})(s,h) = \sqrt{\frac{1}{|B_h(s)|} \sum_{s' \in B_h(s)} \left(\hat{\mathcal{T}}_{s'}(s) - \mathcal{T}_{s'}(s)\right)^2} \ .$$

Proposition 1. *If $|\Omega \cap B_h(s)| \geq C\mu^2 \gamma r \log^6 \gamma$, then with probability of at least $1 - \delta$,*

$$\mathcal{E}(\hat{\mathcal{T}})(s,h) \leq \frac{\alpha h^\beta}{\sqrt{|B_h(s)|}} \left(4\sqrt{\frac{\gamma(2+p)}{p}} + 2\right) \ ,$$

where $\gamma = \sqrt[3]{1/\delta}$ and $p = |\Omega \cap B_h(s)|/|B_h(s)|$.

Proof Assumptions (i) and (iii) above imply that if $K_h(s,s') > 0$ then

$$\|K_h^s \odot (\mathcal{T}(s) - \mathcal{T}(s'))\|_\infty < \alpha h^\beta \ .$$

We can thus assume that if $d(s,s') < h$, an observation $M_{s'} = \mathcal{T}_{s'}(s')$ is equal to $\mathcal{T}_{s'}(s) + Z$ where Z is a random variable whose absolute value is bounded by αh^β. This means that we can use observations $M_{s'} = \mathcal{T}_{s'}(s')$ for estimating the local model $\mathcal{T}(s)$ as long as we admit a noisy measurement process.

Since K is a uniform kernel based on a product distance by assumption (iii), the set $B_h(s)$ is a Cartesian product set. We view this product set as a matrix of dimensions $n_1(h,s) \times n_2(h,s)$ that we approximate. (Note that $n_1(h,s)$ and $n_2(h,s)$ are monotonically increasing with h, and as $h \to \infty$, $n_1(h,s) = n_1$, $n_2(h,s) = n_2$.) The number of observed entries in this matrix approximation problem is $|\Omega \cap B_h(s)|$.

Applying Theorem 7 in Candès and Plan (2010) to the matrix completion problem described above, we get that if $|\Omega \cap B_h(s)| \geq C\mu^2 \gamma r \log^6 \gamma$, then with probability greater than $1 - \gamma^{-3}$,

$$\|K_h^s \odot (\mathcal{T}(s) - \hat{\mathcal{T}}(s))\|_F \leq \alpha h^\beta \left(4\sqrt{\frac{\gamma(2+p)}{p}} + 2\right),$$

where $p = \frac{|\Omega \cap B_h(s)|}{|B_h(s)|}$ is the density of observed samples. Dividing by $\sqrt{|B_h(s)|}$ concludes the proof. ■

When the observed samples are uniformly spread over the matrix, we get $p = m/(n_1 n_2)$, so

$$\begin{aligned} 4\sqrt{\gamma\frac{2+p}{p}} + 2 &= 4\sqrt{\gamma\frac{2+m/(n_1n_2)}{m/(n_1n_2)}} + 2 \\ &= 4\sqrt{\frac{\gamma(2n_1n_2+m)}{m}} + 2. \end{aligned}$$

Multiplying $\alpha h^\beta / \sqrt{|B_h(s)|}$ yields Corollary 1.

Corollary 1. Assume that the conditions of Proposition 1 hold and in addition the observed samples are spread uniformly with respect to d. Then, the following inequality holds

$$\mathcal{E}(\hat{\mathcal{T}})(s,h) \leq \frac{4\alpha h^\beta}{\sqrt{|B_h(s)|}}\sqrt{\frac{\gamma(2n_1n_2 + m)}{m}} + \frac{2\alpha h^\beta}{\sqrt{|B_h(s)|}}.$$

If in addition the matrix M is squared ($n_1 = n_2 = n$) and the distribution of distances d is uniform, then $n_1(h,s) = n_2(h,s) = n/h$, $|B_h(s)| = (n/h)^2$, and $\gamma = n/h$. In this case, the bound on $\mathcal{E}(\hat{\mathcal{T}})(s,h)$ becomes

$$4\alpha h^{\beta+1/2}\sqrt{\frac{2n}{m} + \frac{1}{n}} + \frac{2\alpha h^{\beta+1}}{n}. \tag{12}$$

In the case of a square matrix with uniformly spread samples, it is instructive to view n, m, h as monotonically increasing sequences, indexed by $k \in \mathbb{N}$ and assume that $\lim_{k\to\infty} n_{[k]} = \lim_{k\to\infty} m_{[k]} = \infty$. In other words, we consider the limit of matrices of increasing sizes with an increasing number of samples. In the case of uniformly distributed distances, the bound (12) will converge to zero if

$$\lim_{k\to\infty} \frac{h_{[k]}^{\beta+1}}{n_{[k]}} = \lim_{k\to\infty} \frac{h_{[k]}^{2\beta+1}}{n_{[k]}} = \lim_{k\to\infty} \frac{h_{[k]}^{2\beta+1} n_{[k]}}{m_{[k]}} = 0.$$

4.2 Analysis of $\hat{\hat{\mathcal{T}}} - \mathcal{T}$

We start by showing that $\hat{\mathcal{T}}$ is Hölder continuous with high probability, and then proceed to analyze the estimation error of $\hat{\hat{\mathcal{T}}}$.

Proposition 2. *If $d(s,s') < h$ and Proposition 1 holds at s, s', then with probability at least $1 - \delta$,*

$$\|K_h^s \odot (\hat{\mathcal{T}}(s) - \hat{\mathcal{T}}(s'))\|_F \leq \alpha h^\beta \left(8\sqrt{\frac{\gamma(2+p)}{p}} + 5\right).$$

where $\gamma = \sqrt[3]{2/\delta}$.

Proof Using the triangle inequality for $\|\cdot\|_F$,

$$\begin{aligned} \|K_h^s \odot (\hat{\mathcal{T}}(s) - \hat{\mathcal{T}}(s'))\|_F &\leq \|K_h^s \odot (\hat{\mathcal{T}}(s) - \mathcal{T}(s))\|_F \\ &+ \|K_h^s \odot (\hat{\mathcal{T}}(s') - \mathcal{T}(s'))\|_F \\ &+ \|K_h^s \odot (\mathcal{T}(s) - \mathcal{T}(s'))\|_F. \end{aligned}$$

We apply the bound from Proposition 1 to the first two terms and use the assumption that $\mathcal{T}$ is Hölder continuous to bound the third term. The adjustment to the confidence level $2\gamma^{-3}$ is obtained using the union bound. ■

Proposition 3. *Assume that Proposition 1 holds. Then, with probability of at least $1 - \delta$,*

$$\mathcal{E}(\hat{\hat{\mathcal{T}}})(s,h) \leq \frac{\alpha h^\beta}{\sqrt{|B_h(s)|}}\left(12\sqrt{\frac{\gamma(2+p)}{p}} + 7\right).$$

where $\gamma = \sqrt[3]{(2|\Omega \cap B_h(s)| + 1)/\delta}$.

Proof Using the triangle inequality we get

$$\begin{aligned} &\|K_h^s \odot (\hat{\hat{\mathcal{T}}}(s) - \mathcal{T}(s))\|_F \leq \\ &\quad \|K_h^s \odot (\hat{\mathcal{T}}(s) - \mathcal{T}(s))\|_F + \|K_h^s \odot (\hat{\hat{\mathcal{T}}}(s) - \hat{\mathcal{T}}(s))\|_F. \end{aligned} \tag{13}$$

We bound the first term using Proposition 1. Since $\hat{\hat{\mathcal{T}}}(s)$ is a weighted average of $\hat{\mathcal{T}}(s_i)$, $i = 1, \ldots, q$ with $s_i \in B_h(s)$, the second term is bounded by

$$\begin{aligned} &\|K_h^s \odot (\hat{\hat{\mathcal{T}}}(s) - \hat{\mathcal{T}}(s))\|_F \\ &\quad = \left\|K_h^s \odot \left(\sum_i \frac{w_i}{\sum_j w_j}\hat{\mathcal{T}}(s_i) - \hat{\mathcal{T}}(s)\right)\right\|_F \\ &\quad = \left\|K_h^s \odot \sum_i \frac{w_i}{\sum_j w_j}(\hat{\mathcal{T}}(s_i) - \hat{\mathcal{T}}(s))\right\|_F \\ &\quad \leq \sum_i \left\|\frac{w_i}{\sum_j w_j} K_h^s \odot (\hat{\mathcal{T}}(s_i) - \hat{\mathcal{T}}(s))\right\|_F \\ &\quad \leq \sum_i \frac{w_i}{\sum_j w_j}\|K_h^s \odot (\hat{\mathcal{T}}(s_i) - \hat{\mathcal{T}}(s))\|_F \ . \end{aligned}$$

There are $|\Omega \cap B_h(s)|$ summands in the above term. We bound each of them using Proposition 2. Together with the bound (13) this gives the desired result (after dividing by $\sqrt{|B_h(s)|}$). The adjustment to the confidence level $(2|\Omega \cap B_h(s)| + 1)\gamma^{-3}$ is obtained using the union bound. ■

5. The Parallel LLORMA Algorithm

In the previous sections, we assumed a general kernel function $K_h(s_1, s_2)$, where $s_1, s_2 \in [n_1] \times [n_2]$. This kernel function may be defined in several ways. For simplicity, we assume a product form $K_h((a,b),(c,d)) = K_{h_1}(a,c)K'_{h_2}(b,d)$ where K and K' are kernels on the spaces $[n_1]$ and $[n_2]$, respectively. We used the Epanechnikov kernel (6) for both K, K' as it achieves the lowest integrated squared error (Wand and Jones, 1995), but other choices are possible as well.

The distance d in (6) can be defined using additional information from an outside source describing row (user) similarity or column (item) similarity. If there is no such information available (as is the case in our experiments), d can be computed solely based on the partially observed matrix M. In that case, we may use any distance measure between two row vectors (for K) or two column vectors (for K'). Empirically, we found that standard distance measures such as the 2-norm or cosine similarity do not perform well when M is sparse.

We therefore instead factorize M using standard incomplete SVD (1) $M \approx UV^\top$. Then, we proceed to compute d based on the distances between the rows of factor matrices U (and V). Concretely, we used arc-cosine between users a and c (and items b and d):

$$d(a,c) = \arccos\left(\frac{\langle U_a, U_c\rangle}{\|U_a\| \cdot \|U_c\|}\right) \quad d(b,d) = \arccos\left(\frac{\langle V_a, V_d\rangle}{\|V_b\| \cdot \|V_d\|}\right) \tag{14}$$

where U_i, V_i are the ith row of the matrix U and V. We tried numerous other distances and similarity scores such as the Euclidean distance and cosine similarity. The arc-cosine score empirically performed better than the other scores we experimented with.

Besides of the distance metric, the anchor points $(s_1, \ldots, s_q)$ that define $\hat{\hat{\mathcal{T}}}$ also play a significant role. There are several ways of choosing the anchor points. We randomly choose among training data points unless stated otherwise. Detailed discussion is on Section 7.3.

Algorithm 1 describes the learning algorithm for estimating the local models at the anchor points $\hat{\mathcal{T}}(s_i)$, with $i = 1, \ldots, q$. In line 14, we solve a weighted (by K_{h_1} and K_{h_2}) SVD problem with L_2 regularization. This minimization problem can be computed with gradient-based methods. After these models are estimated, they are combined using (9) to create the estimate $\hat{\hat{\mathcal{T}}}(s)$ for all $s \in [n_1] \times [n_2]$.

Algorithm 1 can actually run faster than vanilla SVD since (a) the q loops may be computed in parallel, and (b) the rank of the our local models can be significantly lower than the rank of global SVD for similar performance (see Section 7). Also, as the kernel K_h has limited support, (c) $K_h(s, s')$ will have few non-zero entries. The weighted SVD problem at line 14 should be sparser than the global SVD, which should result in an additional speedup.

Algorithm 1 The Parallel LLORMA Algorithm

1: **input:** $M \in \mathbb{R}^{n_1 \times n_2}$ whose entries are defined over Ω
2: **parameters:** kernel function $K(\cdot)$ of widths h_1 and h_2
3: rank r and number of local models q
4: regularization values λ_U, λ_V
5: **for all** $t = 1, \ldots, q$ **parallel do**
6: select (a_t, b_t) at random from Ω
7: **for all** $i = 1, \ldots, n_1$ **do**
8: construct entry i: $[K^{a_t}]_i := K_{h_1}(a_t, i)$
9: **end for**
10: **for all** $j = 1, = \ldots, n_2$ **do**
11: construct entry j: $[K^{b_t}]_j := K_{h_2}(b_t, j)$
12: **end for**
13: set $(U^{(t)}, V^{(t)})$ to be the minimizer of:
14: $$\sum_{(i,j)\in\Omega} [K^{(a_t)}]_i\,[K^{(b_t)}]_j \left([U\,V^\top]_{i,j} - M_{i,j}\right)^2 + \lambda_U \sum_{i,k} U_{i,k}^2 + \lambda_V \sum_{j,k} V_{j,k}^2$$
15: **end for**
16: **output:** $\left\{a_t,\, b_t,\, U^{(t)}, V^{(t)}\right\}_{t=1}^{q}$

6. Global LLORMA

Recall that the parallel LLORMA algorithm from Section 5 constructs q local models based on q different anchor points. It then combines the models via kernel regression to produce $\hat{M}$ using (9) which yields the following approximation,

$$\hat{M}_{u,i} = \sum_{t=1}^{q} \frac{K(u_t,u)K(i_t,i)}{\sum_s K(u_s,u)K(i_s,i)} [U^{(t)}{V^{(t)}}^\top]_{u,i}. \tag{15}$$

That is, the algorithm learns each local model independently based on different subsets of the matrix (with some potential overlap). Alternatively, we can directly optimize a joint loss using all local models while bearing in mind the form of the final model as given by (15). In this section we describe an algorithmic alternative that minimizes the following loss with respect to $\{U^{(t)}, V^{(t)}\}_{t=1}^{q}$,

$$\sum_{(u,i)\in\Omega} (\hat{M}_{u,i} - M_{u,i})^2 = \sum_{(u,i)\in\Omega} \left(\sum_{t=1}^{q} \frac{K(u_t,u)K(i_t,i)}{\sum_s K(u_s,u)K(i_s,i)} [(U^{(t)}{V^{(t)}}^\top]_{u,i} - M_{u,i}\right)^2 . \tag{16}$$

This optimization problem can be solved using gradient-based methods, as we use for the global incomplete SVD. Hence, we solve jointly multiple SVD problems with different weights $(K(u_t,u)K(i_t,i))$ multiplying the original matrix. Since we can decompose M into weighted sums,

$$M_{u,i} = \frac{\sum_t K(u_t,u)K(i_t,i)}{\sum_s K(u_s,u)K(i_s,i)} M_{u,i} \ ,$$

the objective function given by (16) can be rewritten as follows,

$$\sum_{(u,i)\in\Omega}\left(\sum_{t=1}^{q}\frac{K(u_t,u)K(i_t,i)}{\sum_s K(u_s,u)K(i_s,i)}[(U^{(t)}V^{(t)\top}]_{u,i}\;-\;\sum_{t=1}^{q}\frac{K(u_t,u)K(i_t,i)}{\sum_s K(u_s,u)K(i_s,i)}M_{u,i}\right)^2$$
$$=\sum_{(u,i)\in\Omega}\left(\sum_{t=1}^{q}\frac{K(u_t,u)K(i_t,i)}{\sum_s K(u_s,u)K(i_s,i)}\left([(U^{(t)}V^{(t)\top}]_{u,i}-M_{u,i}\right)\right)^2. \quad (17)$$

Let us now examine the difference between the above objective with the one tacitly employed by parallel LLORMA algorithm, which amounts to,

$$\sum_{(u,i)\in\Omega}\sum_{t=1}^{q}K(u_t,u)K(i_t,i)\left([(U^{(t)}V^{(t)\top}]_{u,i}-M_{u,i}\right)^2. \quad (18)$$

By construction, both objectives are minimized with respect to $\{U^{(t)},V^{(t)}\}_{t=1}^{q}$ using the squared deviation from M. The difference between the two models is that (17) has a square that encompasses a sum over anchor points. When expanded the term includes the individual squared terms that appear in (18) as well as additional interaction terms. Namely, parallel LLORMA minimizes the sum of square deviations while global LLORMA minimizes the square deviation of sums. In Algorithm 2 we provide the pseudocode of global LLORMA.

A priori we should expect the global version of LLORMA to result in more accurate estimates of M than parallel LLORMA described in Section 5. However, since the objective can no longer be decoupled, the run time global LLORMA is likely to be longer than its parallel counterpart. We provide experimental results which compare the two versions in terms of performance and running time in Section 7.2.

7. Experiments

We conducted several experiments with recommendation data. In Section 7.1, we compare LLORMA to SVD and other state-of-the-art techniques. We also examine in the section dependency of LLORMA on the rank r, the number of anchor points q, and the training set size. In Section 7.2, we compare the parallel and global versions of LLORMA. Section 7.3 introduces several anchor point selection schemes and compare them experimentally.

We used four popular recommendation systems datasets. The MovieLens[1] dataset is one of the most popular datasets in the literature. We used all versions of MovieLens dataset, namely: 100K ($1K\times 2K$ with 10^5 observations), 1M ($6K\times 4K$ with 10^6 observations), and 10M ($70K\times 10K$ with 10^7 observations). We also tested LLORMA on the Netflix dataset which is of size $480K\times 18K$ with 10^8 observations and the Bookcrossing dataset ($100K\times 300K$ with 10^6 observations). These two datasets are much larger than the MovieLens dataset. We also report results on the Yelp dataset ($40K\times 10K$ with 10^5 observations), which is a recent dataset that is part of the ACM RecSys 2013 challenge[2]. The Bookcrossing

1. http://www.grouplens.org/
2. http://recsys.acm.org/recsys13/recsys-2013-challenge-workshop/

Algorithm 2 The Global LLORMA Algorithm

input: $M\in\mathbb{R}^{n_1\times n_2}$ whose entries are defined over Ω
parameters: kernel function $K(\cdot)$ of widths h_1 and h_2
rank r and number of local models q
regularization values λ_U,λ_V
for all $t=1,\ldots,q$ **do**
  select (a_t,b_t) at random from Ω
  for all $i=1,\ldots,n_1$ **do**
    construct entry i: $[K^{a_t}]_i:=K_{h_1}(a_t,i)$
  end for
  for all $j=1,=\ldots,n_2$ **do**
    construct entry j: $[K^{b_t}]_j:=K_{h_2}(b_t,j)$
  end for
end for
minimize with respect to $\{(U^{(t)},V^{(t)})\}_{t=1}^{q}$:

$$\sum_{(i,j)\in\Omega}\left(\sum_{t=1}^{q}\frac{[K^{(a_t)}]_i\,[K^{(b_t)}]_j\,[U^{(t)}V^{(t)\top}]_{i,j}}{\sum_s[K^{(a_s)}]_i\,[K^{(b_s)}]_j}-M_{i,j}\right)^2+\lambda_U\sum_{i,k}\left[U_{i,k}^{(t)}\right]^2+\lambda_V\sum_{j,k}\left[V_{j,k}^{(t)}\right]^2$$

Output: $\left\{a_t,\,b_t,\,U^{(t)},V^{(t)}\right\}_{t=1}^{q}$

and Yelp datasets reflect a recent trend of very high sparsity, often exhibited in real-world recommendation systems.

Unless stated otherwise, we randomly divided the available data into training and test sets such that the ratio of training set size to test set size was 9:1. We created five random partitions and report the average performance over the five partitions. We used a default rating of $(max+min)/2$ for test users or items which lack any rating, where max and min indicate respectively the maximum and minimum possible rating in the dataset.

In our experiments, we used the Epanechnikov kernel with $h_1=h_2=0.8$, a fixed step-size for gradient descent of $\mu=0.01$, and a 2-norm regularization value of $\lambda_U=\lambda_V=0.001$. These values were selected using cross-validation. We set and did not attempt to optimize the parameters $T=100$ (maximum number of iterations), $\epsilon=0.0001$ (gradient descent convergence threshold), and $q=50$ (number of anchor points). We selected anchor points by sampling uniformly users and items from the training points without replacement. We examine more complex anchor point selection scheme in Section 7.3.

7.1 Performance of Parallel LLORMA

Table 2 lists the performance of LLORMA with 50 anchor points, SVD, and two recent state-of-the-art methods based on results published in (Mackey et al., 2011). For a fixed rank r, LLORMA always outperforms SVD. Both LLORMA and SVD perform better as r increases. Both SVD and LLORMA exhibit diminishing returns as the rank increases. LLORMA with a modest rank $r=5$ outperforms SVD of any rank. We can see that LLORMA also outperforms the Accelerated Proximal Gradient (APG) and Divide-and-Conquer Matrix Factorization (DFC) algorithms. For a reference, the Root Mean Square

Method	MovieLens 1M		MovieLens 10M		Netflix		Yelp		Bookcrossing	
APG	–		0.8005		0.8433		–		–	
DFC-NYS	–		0.8085		0.8486		–		–	
DFC-PROJ	–		0.7944		0.8411		–		–	
Rank	SVD	LLORMA	SVD	LLORMA	SVD	LLORMA	SVD	LLORMA	SVD	LLORMA
Rank-1	0.9201	0.9135	0.8723	0.8650	0.9388	0.9295	1.4988	1.4490	3.3747	3.1683
Rank-3	0.8838	0.8670	0.8348	0.8189	0.8928	0.8792	1.4824	1.3133	3.3679	3.0315
Rank-5	0.8737	0.8537	0.8255	0.8049	0.8836	0.8604	1.4775	1.2358	3.3583	2.9482
Rank-7	0.8678	0.8463	0.8234	0.7950	0.8788	0.8541	1.4736	1.1905	3.3488	2.8828
Rank-10	0.8650	0.8396	0.8219	0.7889	0.8765	0.8444	1.4708	1.1526	3.3283	2.8130
Rank-15	0.8652	0.8370	0.8225	0.7830	0.8758	0.8365	1.4685	1.1317	3.3098	2.7573
Rank-20	0.8647	0.8333	0.8220	0.7815	0.8742	0.8337				

Table 2: RMSE achieved by different algorithms on five datasets: MovieLens 1M, MovieLens 10M, Netflix, Bookcrossing, and Yelp. Results for APG (Toh and Yun, 2010) and DFC were taken from (Mackey et al., 2011).

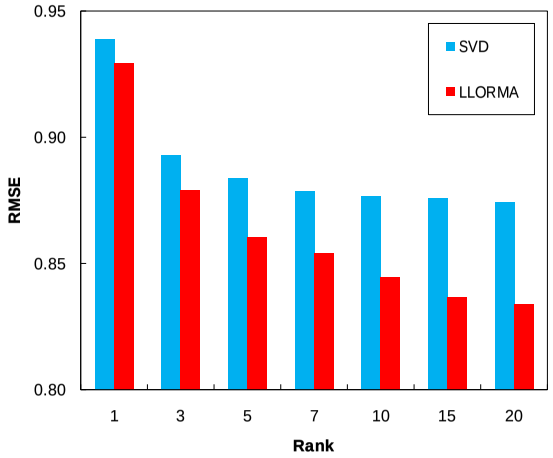


Figure 3: RMSE of LLORMA and SVD as a function of the rank on the Netflix dataset.

Error (RMSE) we achieved (0.8337) is a better score than the goal of Netflix competition (0.8567).[3]

As we can see from Figure 3, the improvement in the performance of SVD is rather minor beyond a rank of 7 while LLORMA's improvement is still evident until a rank of about 20. Both approaches cease to make substantial improvements beyond the aforementioned ranks and exhibit diminishing returns for high ranks. As discussed in earlier sections, the diminishing returns of SVD for high ranks can be interpreted in two ways: (i) The

3. We provide this number merely as reference since we measured our performance on a test set different from original Netflix test set, which is no longer available. Our result are based on a random sub-sampling of the Netflix training data as described above. See also the discussion in (Mackey et al., 2011).

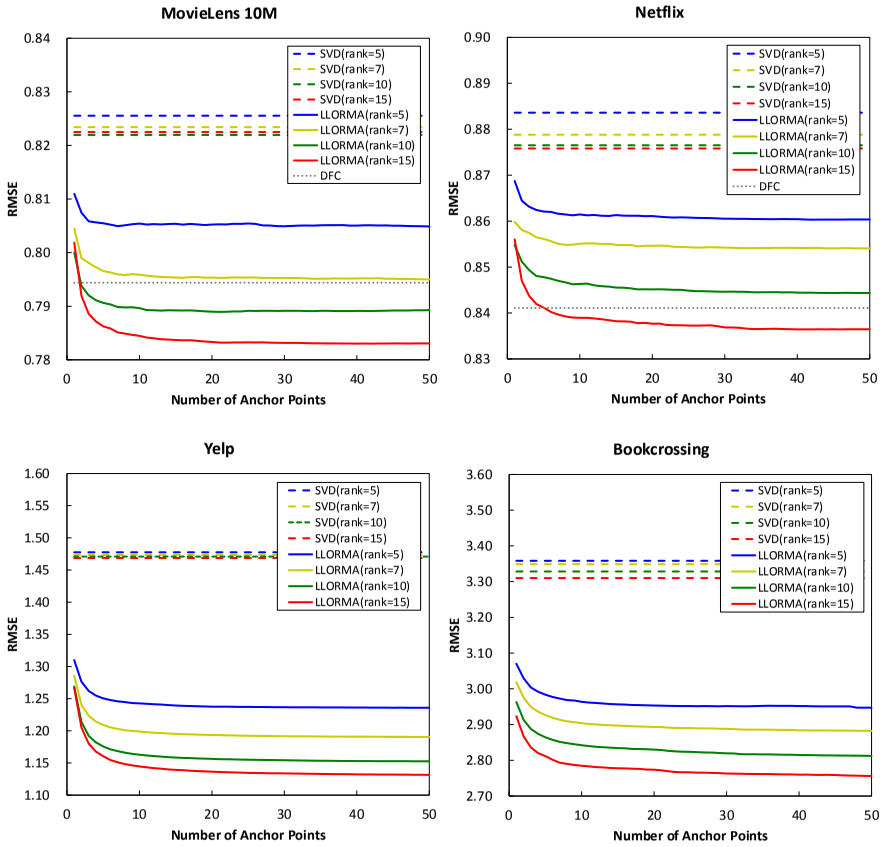


Figure 4: RMSE of LLORMA, SVD, and two baselines on MovieLens 10M (top-left), Netflix (top-right), Yelp (bottom-left), and Bookcrossing (bottom-right) datasets. The results for LLORMA are depicted by thick solid lines, while for SVD with dotted lines. Models of the same rank are have identical colors.

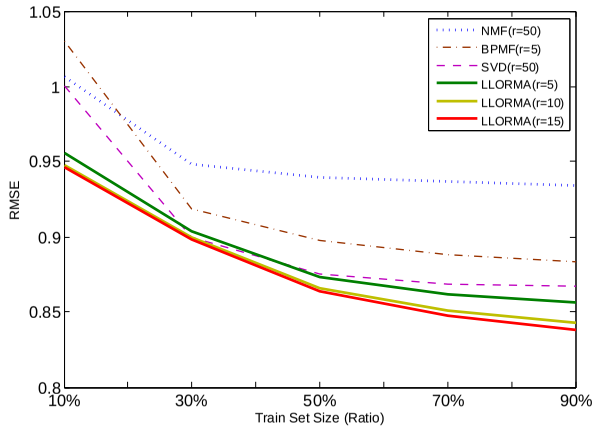


Figure 5: RMSE of SVD, LLORMA, NMF, and BPMF methods as a function of training set size for the MovieLens 1M dataset.

global low-rank assumption is correct and the SVD approximates the rating matrix almost optimally. (ii) The global low-rank assumption is incorrect and the diminishing returns are due to various deficiencies such as overfitting or reaching an inferior local minimum. Since LLORMA improves beyond SVD's best performance, it deems that the second hypothesis is more plausible. The fact that LLORMA's performance asymptotes at higher ranks than SVD may indicate the first hypothesis is to a large extent correct at a local region yet not globally.

Figure 4 compares the RMSE of LLORMA, SVD, and the DFC method of Mackey et al. (2011). We plot the RMSE of LLORMA as a function of the number of anchor points. As in the case of Table 2, both LLORMA and SVD improve as r increases. Here again LLORMA with local rank of at least 5 outperforms SVD of any rank. Moreover, LLORMA outperforms SVD even with only a few anchor points. Figure 5 shows the RMSE of LLORMA as a function of the training set size, and compares it with global SVD of 50. We also plot results for a few other methods that have shown to achieve very good approximation accuracy: non-negative matrix factorization (NMF) with rank 50 (Lee and Seung, 2001) and Bayesian probabilistic matrix factorization (BPMF) with rank 5 (Salakhutdinov and Mnih, 2008b). The test set size was fixed to 10% of the MovieLens 1M and the RMSE was averaged over five random train-test splits. The graph shows that all methods improve as the training set size increases while LLORMA consistently outperforms SVD and the other baselines.

To recap, the experimental results presented thus far indicate that LLORMA outperforms SVD and other state-of-the-art methods even when using relatively lower-rank approximation. Moreover, LLORMA is capable of achieving good accuracy with rather small number of anchor points and seems to perform well across a variety of training set sizes.

7.2 Comparison of Global and Parallel LLORMA

We proposed two implementations of LLORMA in this paper: a decoupled parallel approach (Section 5) and a global approximation version (Section 6). In earlier sections, we conjectured that the global version is likely to be more accurate in terms of RMSE as it directly optimizes the objective function. In terms of computational efficiency, however, we naturally expected the parallel version to be faster as it can take advantage of multicore and distributed computing architectures. In this subsection, we experimentally verify the two conjectures.

We compared the two versions of LLORMA on MovieLens 100K dataset. We tested local rank values in $\{1, 3, 5, 7, 10\}$. The experiment was conducted on a quad-core machine with 4 threads while suspending any other process. For both versions, we constructed 50 local models and repeated the experiment 5 times with different anchor points. Table 3 reports the average test RMSE and average elapsed time for training. As conjectured, global LLORMA results in more accurate estimations on unseen data than parallel LLORMA. However, the performance gap between the two approaches reduces as the rank increases. The parallel version LLORMA runs about 3 times faster than global LLORMA indicating that a fairly high utilization of the multicore architecture.

Method	Global LLORMA		Parallel LLORMA	
Rank	Test RMSE	Time	Test RMSE	Time
1	0.9072	6:04	0.9426	1:09
3	0.9020	10:27	0.9117	3:20
5	0.8990	14:40	0.9041	5:26
7	0.8986	19:43	0.9010	7:50
10	0.8975	28:59	0.8985	11:49

Table 3: RMSE and training time for Global and Parallel LLORMA on MovieLens 100K.

7.3 Anchor Points Selection

The method for selecting anchor points in LLORMA is important as it may affect the prediction time as well as generalization performance. If the row and column indices of the test data are provided in advance, we may choose anchor points from the test set distribution. However, in most applications the test set is not known apriori, and in fact is likely to increase and change in time. We therefore confined ourselves to three sampling methods and one clustering methods based on low-rank representation of the data. In the rest of the section we use q to denote the number of anchor points. The following are anchor point selection methods we tried.

Complete: Sample anchor points uniformly from the entire set of indices $[n_1] \times [n_2]$.

Trainset: Sample anchor points uniformly from the observed entries, Ω.

Coverage: Select anchor points such that no entry in $[n_1] \times [n_2]$ is too distant from the rest of the anchor points.

k-means: Run k-means clustering on the entries in Ω each represented using the induced d-dimensional space obtained by SVD with $k = q$.

The first two sampling methods are straightforward. The third method, termed "Coverage" was implemented by sampling aq with $a > 1$ user-item pairs and then adding an anchor point whose minimum distance to existing anchor points is the largest. The fourth selection methods that we tested is based on clustering the observed entries. It is based on the observation that an anchor point need not be an entry in the observation matrix but may rather consist of a combination of matrix entries. Recall that we weigh each local model based on user and item similarity. As explained in Section 5, each a user (item) is represented as a row of a low-rank matrices U (respectively, V) attained by the global SVD, namely, $M \approx UV^\top$. Denoting the intrinsic dimension of U and V by d, each user and item can be represented as a d-dimensional vector. Instead of each row of U and V, we can generalize the space of anchor points to any point in this d-dimensional vector space. A good set of anchor points may be the q cluster centers of the d-dimensional representations of the entries of M which is computed using the k-means algorithm.

Table 4 provides performance results of *Global* LLORMA using different anchor point selection methods. In the experiments we used the MovieLens 100K datasets with 5 random train-test partitions. The results we report are based on averages over the 5 random splits. As one may anticipate, the different selection schemes perform similarly when q is large. For small values of q (10 or 20), we would like to underscore the following observations. The first two methods (Complete and Trainset) perform similarly. The clustering-based anchor point construction generally performs slightly better than the three other methods. Somewhat surprisingly, the Coverage method performs the worst for small values of q. Nonetheless, when q is sufficiently large all methods achieve about the same results.

	10 Local models				20 Local Models			
Rank	Complete	Trainset	Coverage	k-means	Complete	Trainset	Coverage	k-means
1	0.9276	0.9296	0.9360	0.9230	0.9195	0.9206	0.9235	0.9166
3	0.9245	0.9237	0.9327	0.9208	0.9164	0.9152	0.9189	0.9134
5	0.9240	0.9221	0.9277	0.9190	0.9125	0.9128	0.9154	0.9111
7	0.9231	0.9251	0.9279	0.9202	0.9140	0.9129	0.9163	0.9103
10	0.9242	0.9271	0.9301	0.9236	0.9129	0.9125	0.9141	0.9112

Table 4: RMSE for various anchor point selection schemes on MovieLens 100K datset.

7.4 Comparison to Ensemble of Independent Models

The algebraic form of the parallel estimator $\hat{\hat{\mathcal{T}}}$ as given by (9) is a linear combination of local models, each of which focuses on a subset of user-item pairs. This algebraic form is reminiscent of ensemble methods (Jacobs et al., 1991) such as Bagging (Breiman, 1996), where the final model is a linear combination of simple models, each weighed by a predefined coefficient. Ensemble methods such as boosting and Bagging have been shown to be effective tools for combining models primarily for classification tasks. In this section we examine the

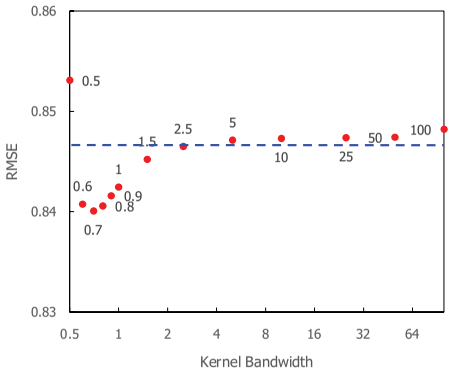


Figure 6: Performance of LLORMA as a function of the kernel width. The vertical axis indicates the approximations' root mean squared error (RMSE) and the horizontal axis the kernel width ($h_1 = h_2$) in log scale. The dotted blue line shows the average RMSE of Bagging for reference.

connection between LLORMA and an ensemble method based on Bagging for low rank matrix approximation.

There are two main differences between Bagging and LLORMA. In Bagging, the dataset constructed for training each sub-model is uniformly sampled with replacements. In contrast, LLORMA's sampling is based on a non-uniform distribution respecting locality as defined by the distance metric over user-item pairs. The second difference is that Bagging assigns equal weights to each base-model. In LLORMA, each *local* model is associated with a weight that is proportional to the proximity of the anchor point to the test point. That is, the weights of the local models vary and are determined at inference time.

We conducted two set of experiments comparing LLORMA with Bagging. In the first experiment, we varied the kernel widths gradually increasing the widths to the matrix dimensions, thus ending with a uniform kernel over Ω. We normalized the kernel width so that a value of 1 corresponds to the full dimension. As the width increases, the overlap between local models becomes more and more substantial. In addition, the bias of each local model increases due to the decrease in locality. Analogously, the variance of each model decreases due to the increase in the actual training set size.

Figure 6 shows performance of LLORMA on MovieLens 100K dataset for various kernel widths. The best performance is obtained for kernel width between 0.7 and 0.8. The performance rapidly deteriorates as the width decreases and slowly degrades as the width increases with an optimal width close to the middle of the range.

In our second experiment, we compared LLORMA with Bagging by taking $|\Omega|$ samples with replacements, which in expectation covers two thirds of the observed entries. Table 5 compares the performance of global SVD of rank 10 and 15, LLORMA with 100 local

models, and Bagging with 100 models. Each result is the average of 5 random splits of the dataset. It is apparent from the table that both LLORMA and Bagging outperform global SVD. Further, LLORMA achieves lower RMSE than Bagging. The improvement of LLORMA over Bagging is statistically significant based on a paired t-test with p-values of 0.0022 for MovieLens 100K and 0.0014 for MovieLens 1M. These p-values correspond to a confidence level over 99%. LLORMA also outperforms Bagging with respect to the median average error (MAE).

Dataset	MovieLens 100K		MovieLens 1M	
Method	MAE	RMSE	MAE	RMSE
SVD rank=10	0.7189	0.9108	0.6922	0.8683
SVD rank=15	0.7170	0.9094	0.6913	0.8676
Bagging	0.6985	0.8930	0.6620	0.8481
LLORMA	0.6936	0.8881	0.6577	0.8423

Table 5: Comparison of the median average error (MAE) and root mean squared error (RMSE) for SVD, Bagging, and LLORMA on MovieLens dataset.

8. Related work

Matrix factorization for recommender systems have been the focus of voluminous amount of research especially since the Netflix Prize competition. It is clearly impossible to review all of the existing approaches. We review here a few of the notable approaches. Billsus and Pazzani (1998) initially proposed applying SVD for collaborative filtering problems. Salakhutdinov and Mnih (2008a) presented probabilistic matrix factorization (PMF) and later Salakhutdinov and Mnih (2008b) extended matrix factorization to fully Bayesian approach. Lawrence and Urtasun (2009) proposed a non-linear version of PMF. Rennie and Srebro (2005) proposed a maximum-margin approach. Lee et al. (2012b) conducted a comprehensive experimental study comparing a number of state-of-the-art and traditional recommendation system methods using the PREA toolkit (Lee et al., 2012c). Further algorithmic improvements in matrix completion were demonstrated in Toh and Yun (2010); Keshavan et al. (2010). The work that is perhaps the most similar in to LLORMA is Divide-and-Conquer Matrix Factorization (DFC) (Mackey et al., 2011). DFC also divides the completion problems into a set of smaller matrix factorization problems. Our approach differs DFC in that we use a metric structure on $[n_1] \times [n_2]$ and use overlapping partitions. Another matrix approximation by sub-division based on clustering was reported in Mirbakhsh and Ling (2013) for the task of seeking user and item communities. In addition to monolithic matrix factorization scheme, several ensemble methods have also been proposed. DeCoste (2006) suggested ensembles of maximum margin matrix factorization (MMMF). The Netflix Prize winner (Bell et al., 2007; Koren, 2008) used combination of memory-based and matrix factorization methods. The Netflix Prize runner-up (Sill et al., 2009) devised Feature-Weighted Least Square (FWLS) solver, using a linear ensemble of learners with dynamic weights. Lee et al. (2012a) extended FWLS by introducing automatic stage-wise feature induction. Kumar et al. (2009) and Mackey et al. (2011) applied ensembles to Nystrom method and DFC, respectively. Other local learning paradigms were suggested in the context dimensionality such as local principal component analysis (Kambhatla and Leen, 1997) and local linear embedding (LLE) (Roweis and Saul, 2000). A relatively recent paper on matrix completion (Wang et al., 2013) applies low-rank factorization to clusters of points. Last, we would like to point to the formal work on low-rank matrix Completion that is closest to the analysis presented in this paper. Candès and Tao (2010) derived a bound on the performance of low-rank matrix completion. As mentioned in previous section our analysis is based on (Candès and Plan, 2010) who adapted the analysis of Candès and Tao (2010) to noisy settings. Some more remote related results were presented in (Shalev-Shwartz et al., 2011; Foygel and Srebro, 2011; Foygel et al., 2012).

9. Summary

We presented a new approach for low-rank matrix approximation based on the assumption that the matrix is locally low-rank. Our proposed algorithm, called LLORMA, is highly parallelizable and thus scales well with the number of observations and the dimension of the problem. Our experiments indicate that LLORMA outperforms several state-of-the-art methods without a significant computational overhead. We also presented a formal analysis of LLORMA by deriving bounds that generalize standard compressed sensing results and express the dependency of the modeling accuracy on the matrix size, training set size, and locality (kernel bandwidth parameter). Our method is applicable beyond recommendation systems so long as the locality assumption holds and a reasonable metric space can be identified.

Acknowledgments

We would like to thank Le Song for insightful discussions. Part of this work was done while the first and second authors were in Georgia Institute of Technology.

References

R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proc. of the ACM SIGKDD*, 2007.

D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proc. of the International Conference on Machine Learning*, 1998.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

E.J. Candès and Y. Plan. Matrix completion with noise. *Proc. of the IEEE*, 98(6):925–936, 2010.

E.J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

D. DeCoste. Collaborative prediction using ensembles of maximum margin matrix factorizations. In *Proc. of the ICML*, 2006.

R. Foygel and N. Srebro. Concentration-based guarantees for low-rank matrix reconstruction. *ArXiv Report arXiv:1102.3923*, 2011.

R. Foygel, N. Srebro, and R. Salakhutdinov. Matrix reconstruction with the local max norm. *ArXiv Report arXiv:1210.5196*, 2012.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Nandakishore Kambhatla and Todd K Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 99:2057–2078, 2010.

Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008.

S. Kumar, M. Mohri, and A. Talwalkar. Ensemble nystrom method. In *Advances in Neural Information Processing Systems*, 2009.

N. D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proc. of the International Conference on Machine Learning*, 2009.

D. Lee and H. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, 2001.

J. Lee, M. Sun, S. Kim, and G. Lebanon. Automatic feature induction for stagewise collaborative filtering. In *Advances in Neural Information Processing Systems*, 2012a.

J. Lee, M. Sun, and G. Lebanon. A comparative study of collaborative filtering algorithms. *ArXiv Report 1205.3193*, 2012b.

J. Lee, M. Sun, and G. Lebanon. Prea: Personalized recommendation algorithms toolkit. *Journal of Machine Learning Research*, 13:2699–2703, 2012c.

L. W. Mackey, A. S. Talwalkar, and M. I. Jordan. Divide-and-conquer matrix factorization. In *Advances in Neural Information Processing Systems*, 2011.

N. Mirbakhsh and C. X. Ling. Clustering-based matrix factorization. *ArXiv Report arXiv:1301.6659*, 2013.

J.D.M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proc. of the International Conference on Machine Learning*, 2005.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, 2008a.

R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. of the International Conference on Machine Learning*, 2008b.

S. Shalev-Shwartz, A. Gonen, and O. Shamir. Large-scale convex minimization with a low-rank constraint. In *Proc. of the International Conference on Machine Learning*, 2011.

J. Sill, G. Takacs, L. Mackey, and D. Lin. Feature-weighted linear stacking. *Arxiv preprint arXiv:0911.0460*, 2009.

K.C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 6(15):615–640, 2010.

M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman and Hall/CRC, 1995.

Yi Wang, Arthur Szlam, and Gilad Lerman. Robust locally linear analysis with applications to image denoising and blind inpainting. *SIAM Journal on Imaging Sciences*, 6(1):526–562, 2013.

Journal of Machine Learning Research 17 (2016) 1-26 Submitted 6/14; Revised 5/15; Published 4/16

A Consistent Information Criterion for Support Vector Machines in Diverging Model Spaces

Xiang Zhang XZHANG23@NCSU.EDU
Yichao Wu WU@STAT.NCSU.EDU
Department of Statistics
North Carolina State University
Raleigh, NC 27695, USA

Lan Wang WANGX346@UMN.EDU
Department of Statistics
The University of Minnesota
Minneapolis, MN 55455, USA

Runze Li RZLI@PSU.EDU
Department of Statistics and The Methodology Center
The Pennsylvania State University
University Park, PA 16802-2111, USA

Editor: Jie Peng

Abstract

Information criteria have been popularly used in model selection and proved to possess nice theoretical properties. For classification, Claeskens et al. (2008) proposed support vector machine information criterion for feature selection and provided encouraging numerical evidence. Yet no theoretical justification was given there. This work aims to fill the gap and to provide some theoretical justifications for support vector machine information criterion in both fixed and diverging model spaces. We first derive a uniform convergence rate for the support vector machine solution and then show that a modification of the support vector machine information criterion achieves model selection consistency even when the number of features diverges at an exponential rate of the sample size. This consistency result can be further applied to selecting the optimal tuning parameter for various penalized support vector machine methods. Finite-sample performance of the proposed information criterion is investigated using Monte Carlo studies and one real-world gene selection problem.

Keywords: Bayesian Information Criterion, Diverging Model Spaces, Feature Selection, Support Vector Machines

1. Introduction

We consider binary classification using linear support vector machines (SVMs). It is well known that the standard SVM uses all features while constructing the classification rule. In the extreme case of regression that the number of features is much larger than the sample size, if the true model is non-sparse, no method can identify the truth correctly due to the limited information from the data. This is the so-called *curse of dimensionality*. In many important applications, however, it is reasonable to simplify the problem by assuming the true model to be sparse. For example, in cancer classification using genomic data where the number of probes (or genes) can be tens of thousands and the number of patient samples is typically only a few dozens, biologists find it plausible to assume that only a small subset of genes are relevant. In this case, it is more desirable to build a classifier based only on those relevant genes. Yet in practice, it is largely unknown which genes are relevant and this calls for feature selection methods. The potential benefits of feature selection include reduced computational burden, improved prediction performance and simple model interpretation. See Guyon and Elisseeff (2003) for more discussions. Our goal in this paper is consistent feature selection for the SVM.

There has been a rich literature on feature selection for the SVM. Weston et al. (2000) proposed a scaling method to select important features. Guyon et al. (2002) suggested the SVM recursive feature elimination (SVM RFE) procedure. It has been shown that the SVM can be fitted in the regularization framework using the hinge loss and the L_2 penalty (Wahba et al., 1999). Thereafter various forms of penalized SVMs have been developed for simultaneous parameter estimation and feature selection. Bradley and Mangasarian (1998), Zhu et al. (2004) and Wegkamp and Yuan (2011) studied properties of the L_1 penalized SVM. Wang et al. (2006) proposed SVM with a combination of L_1 and L_2 penalties. Zou and Yuan (2008) considered the L_∞ penalized SVM when there is prior knowledge about the grouping information of features. Zhang et al. (2006) and Becker et al. (2011) suggested SVM with a non-convex penalty in the application of gene selection. Though all these methods target selecting the best subset of features, theoretical justification about how well the selected subset is estimating the true model is still largely underdeveloped. Recently Zhang et al. (2014) showed that the SVM penalized with a class of non-convex penalties enjoys the oracle property (Fan and Li, 2001), that is, the estimated classifier behaves as if the subset of all relevant features is known *a priori*. Yet this model selection consistency result relies heavily on the proper choice of the involved tuning parameter which is often selected by cross-validation in practice. However, Wang et al. (2007) showed that the generalized cross-validation criterion can lead to overfitting even with a very large sample size.

Information criteria such as AIC (Akaike, 1973) and BIC (Schwarz, 1978) have been used for model selection and their theoretical properties have been well studied, see Shao (1997), Shi and Tsai (2002) and references therein. It is well understood that the BIC can identify the true model consistently when the dimensionality is fixed. The idea of combining information criterion with support vector machine to select relevant features was first proposed in Claeskens et al. (2008). They proposed the SVM information criterion (SVMIC$_L$) and provided some encouraging numerical evidence. Yet its theoretical properties, such as model selection consistency, have not been investigated.

In this paper, we propose a consistent SVM information criterion for model selection in the diverging model spaces. We first fill the gap by providing theoretical justification for the criterion SVMIC$_L$ proposed in Claeskens et al. (2008). Our results show that this information criterion is model selection consistent in the fixed dimensional model space, but it can be too liberal when the candidate model space is diverging. To remedy this problem, a modified information criterion for high dimensional case (SVMIC$_H$) is introduced. The extension of model selection consistency from SVMIC$_L$ to SVMIC$_H$ is a challenging problem. The point-wise consistency of SVM solution is enough to justify the model selection consistency if the number of candidate models is fixed. Nevertheless, in the diverging model

spaces the probabilities for favoring an underfitted or overfitted model by the information criterion can accumulate at a very fast speed and alternative techniques are required. We develop the uniform consistency of SVM solution which has not been carefully studied in the literatures. Based on the uniform convergence rate, we prove that the new information criterion possesses model selection consistency even when the number of features diverges at an exponential rate of the sample size. That is, with probability arbitrarily close to one, we can identify the true model from all the underfitted or overfitted models in the diverging model spaces. To the best of our knowledge, this is the first result of model selection consistency for the SVM. We further apply this information criterion to the problem of tuning parameter selection in penalized SVMs. The proposed support vector machine information criterion can be computed easily after fitting the SVM with computation cost much lower than resampling methods like cross-validation. Simulation studies and real data examples confirm the superior performance of the proposed method in terms of model selection consistency and computational scalability.

In Section 2 we define the support vector machine information criterion. Its theoretical properties are studied in Section 3. Sections 4 and 5 present numerical results on simulation examples and real-world gene selection datasets, respectively. We conclude with some discussions in Section 6.

2. Support vector machine information criterion

In this paper we use normal font for scalars and bold font for vectors or matrices. Consider a random pair $(\mathbf{X}, Y)$ with $\mathbf{X}^T = (1, X_1, \ldots, X_p) = (1, (\mathbf{X}^+)^T) \in \mathbf{R}^{(p+1)}$ and $Y \in \{1, -1\}$. Let $\{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ be a set of training data independently drawn from the distribution of $(\mathbf{X}, Y)$. Denote $\boldsymbol{\beta}$ to be a $(p+1)$-dimensional vector of interest with $\boldsymbol{\beta}^T = (\beta_0, \beta_1, \ldots, \beta_p) = (\beta_0, (\boldsymbol{\beta}^+)^T) \in \mathbf{R}^{(p+1)}$. Let $||\cdot||$ be the Euclidean norm operator of a vector. The goal of linear SVM is to estimate a hyperplane defined by $\mathbf{X}^T\boldsymbol{\beta} = 0$ via solving the optimization problem

$$\min_{\boldsymbol{\beta}} \{C \sum_{i=1}^n \xi_i + \frac{1}{2}||\boldsymbol{\beta}^+||^2\} \tag{1}$$

subject to the constraints that $\xi_i \geq 0$ and $Y_i\mathbf{X}_i^T\boldsymbol{\beta} \geq 1 - \xi_i$ for all $i = 1, \ldots, n$, where $C > 0$ is a tuning parameter. This can be written equivalently into an unconstrained regularized empirical loss minimization problem:

$$\min_{\boldsymbol{\beta}} \{\frac{1}{n}\sum_{i=1}^n H(Y_i\mathbf{X}_i^T\boldsymbol{\beta}) + \frac{\lambda_n}{2}||\boldsymbol{\beta}^+||^2\}, \tag{2}$$

where $H(t) = (1-t)_+$ is the hinge loss function, $(z)_+ = \max(z, 0)$ and $\lambda_n > 0$ is a tuning parameter with $C = (n\lambda_n)^{-1}$.

Following the definition in Koo et al. (2008), we denote $(\boldsymbol{\beta}^*)^T = (\beta_0^*, \beta_1^*, \ldots, \beta_p^*) = (\beta_0^*, (\boldsymbol{\beta}^{*+})^T) \in \mathbf{R}^{(p+1)}$ to be the true parameter value that minimizes the population hinge loss. That is,

$$\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} \mathbb{E}(1 - Y\mathbf{X}^T\boldsymbol{\beta})_+.$$

Note that $\boldsymbol{\beta}^*$ is not necessarily always the same as the Bayes rule. However, it gives the optimal upper bound of the risk of the 0-1 loss through convex relaxation and its sparsity structure is exactly the same as the one of Bayes rule in special cases such as linear discriminant analysis. For more discussions see Zhang et al. (2014). Denote $S = \{j_1, \ldots, j_d\} \subset \{1, \ldots, p\}$ to be a candidate model, $\mathbf{X}_{i,S}^T = (1, X_{i,j1}, \ldots, X_{i,jd})$, $\boldsymbol{\beta}_S^T = (\beta_0, \beta_{j1}, \ldots, \beta_{jd})$ and $|S|$ the cardinality of S. The subset of all relevant features is defined by $S^* = \{j : 1 \leq j \leq p, \beta_j^* \neq 0\}$. We assume that the truth $\boldsymbol{\beta}^*$ is sparse (i.e., most of its components are exactly zero). Denote $q = |S^*|$ which characterizes the sparsity level. We assume that q is fixed and does not depend on n. In this paper, we consider the diverging model spaces in which the dimensionality $p = p_n$ is allowed to increase with n and can be potentially much larger than n. We also assume that $\lambda_n \to 0$ as $n \to \infty$ and only consider the non-separable case in the limit to ensure the uniqueness of the truth $\boldsymbol{\beta}^*$. Here by non-separable, we mean that the two classes cannot be linearly separated from each other.

To identify the true model S^*, Claeskens et al. (2008) proposed an information criterion for SVM (denoted by SVMIC_L) based on the slack variables $\{\xi_i\}_{i=1}^n$. That is,

$$\text{SVMIC}_L(S) = \sum_{i=1}^n \xi_i + |S|\log(n),$$

where $\{\xi_i\}_{i=1}^n$ are obtained from (1) only using the variables in S. This information criterion is equivalent to

$$\text{SVMIC}_L(S) = \sum_{i=1}^n (1 - Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+ + |S|\log(n), \tag{3}$$

where $\widehat{\boldsymbol{\beta}}_S = \arg\min\{1/n\sum_{i=1}^n(1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S)_+ + \lambda_n/2||\boldsymbol{\beta}_S^+||^2\}$. It is evident that the SVMIC_L directly follows the spirit of BIC. Claeskens et al. (2008) fixed $C = 1$ in (1) and found minor difference for different choices of C, which is equivalent to $\lambda_n = 1/n$ in (2). To be consistent with the work in Claeskens et al. (2008), we also consider this choice of λ_n in this paper. There are two potential drawbacks of this information criterion. First, though supported with numerical findings, theoretical properties of SVMIC_L, such as model selection consistency, are largely unknown even under the assumption of a fixed p. Second, in many real world datasets where the dimension can be much larger than the sample size, it would be more appropriate to consider the model selection problem in the framework of diverging model spaces. This extension from low dimensions to high dimensions can greatly change the theoretical properties of the information criterion. Chen and Chen (2008) showed that the ordinary BIC for linear regression cannot identify the true model consistently in the diverging p case. Wang et al. (2009) showed that the ordinary BIC fails to select a consistent shrinkage level in penalized least squares regression with a diverging p. Such results in the literature suggest that SVMIC_L may also suffer from inconsistency in high dimensions and alternative criterion is needed.

To overcome these issues, we propose a modified support vector machine information criterion for model selection in a high dimensional model space (denote by SVMIC_H). This criterion is adpated from SVMIC_L and defined as

$$\text{SVMIC}_H(S) = \sum_{i=1}^n (1 - Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+ + L_n|S|\log(n), \tag{4}$$

where $\widehat{\boldsymbol{\beta}}_S = \arg\min\{1/n\sum_{i=1}^n(1-Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S)_+ + \lambda_n/2||\boldsymbol{\beta}_S^+||^2\}$ and L_n is a constant sequence that diverges to infinity. Note that if L_n is a non-diverging constant then this reduces to SVMIC$_L$ in the limit. We will show that SVMIC$_H$ possesses the nice property of model selection consistency even when p increases at an exponential rate of n. Compared to SVMIC$_L$ in Claeskens et al. (2008), our information criterion SVMIC$_H$ adds larger penalty to the size of the selected subset and behaves more conservatively. As we will see, this additional preference for simpler models plays an important role in consistent model selection when we are searching over diverging model spaces.

We make two remarks about SVMIC$_H$. First, the choice of L_n in (4) is flexible. It is *not* a tuning parameter and does not need to be chosen by computationally intensive methods such as cross-validation. We will show that a wide spectrum of L_n can lead to a consistent information criterion. This is further confirmed in our simulations and real data analysis where we examine different choices of L_n. Therefore the computation cost of SVMIC$_H$ is the same as SVMIC$_L$ and much lower than cross-validation. Second, it is possible to define the information criterion as the log-transformed version, that is

$$\log(\sum_{i=1}^n(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+) + L_n|S|\log(n)/n$$

which is scalar invariant. It can be shown the model selection consistency still holds for this definition. However, we follow the advice from Guyon et al. (2002) to standardize variables before training SVM and as a consequence we automatically have scalar invariance of the sum of slack variables. To be consistent with SVMIC$_L$ defined in Claeskens et al. (2008), we take definition (4) in our paper.

3. Theoretical results

3.1 Notations and conditions

To facilitate technical proofs, we introduce some additional notation. Denote $L(\boldsymbol{\beta}) = \mathbb{E}(1-Y\mathbf{X}^T\boldsymbol{\beta})_+$. Recall that $\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} L(\boldsymbol{\beta})$. Let $\mathbf{S}(\boldsymbol{\beta}) = -\mathbb{E}[\mathbf{1}(1-Y\mathbf{X}^T\boldsymbol{\beta}\geq 0)Y\mathbf{X}]$, where $\mathbf{1}(\cdot)$ is the indicator function. Also define $\mathbf{H}(\boldsymbol{\beta}) = \mathbb{E}[\delta(1-Y\mathbf{X}^T\boldsymbol{\beta})\mathbf{X}\mathbf{X}^T]$, where $\delta(\cdot)$ is the Dirac delta function. Koo et al. (2008) showed that under some regularity conditions, $\mathbf{S}(\boldsymbol{\beta})$ and $\mathbf{H}(\boldsymbol{\beta})$ behave like the gradient and Hessian matrix of $L(\boldsymbol{\beta})$, respectively. Furthermore we denote f_+ and f_- to be the densities of $\mathbf{X}^+ \in \mathbf{R}^p$ conditioning on $Y=1$ and $Y=-1$, respectively.

Given the dimension p, the number of candidate models is 2^p-1. When p is very large, we cannot afford to calculate SVMIC$_H(S)$ for all possible subsets. Instead, we only search for the best model in a restricted model space. To be more specific, we denote $\widehat{S}$ the model chosen by SVMIC$_H$ such that

$$\widehat{S} = \arg\min_{S:|S|\leq M_n}\text{SVMIC}_H(S), \tag{5}$$

where M_n is a sequence of positive integers that bounds the size of the restricted model space from above. In this paper, we consider $M_n = O(n^\kappa)$ for some constant $0<\kappa<1/2$, that is, we only consider the candidate model with the size diverges slower than $\sqrt{n}$. One motivation for this choice of M_n is the "bet on sparsity" principle (Hastie et al., 2001).

Note that by Lemma 1 of Zhang et al. (2014), if the number of relevant features diverges faster than $\sqrt{n}$, the true parameter $\boldsymbol{\beta}^*$ cannot be estimated consistently even with the oracle information of the true model S^*. Therefore, there is no need to consider those models with sizes increasing faster than $\sqrt{n}$ as in general no method would work for them even when the true underlying model is known. Notice also that it is possible to prove the model selection consistency without the restricted model space. However, this would requires $p_n = o(\sqrt{n})$, which cannot diverge very fast with the sample size.

We now present the technical conditions that are needed for studying the theoretical properties of SVMIC$_H$.

(A1) f_+ and f_- are continuous and have some common support in $\mathbf{R}^p$.

(A2) $|X_j| \leq M < \infty$ for some positive constant M and $1\leq j\leq p$.

(A3) For all $S\in\{S:|S|\leq M_n, S\supseteq S^*\}$, $\lambda_{\max}(\mathbb{E}(\mathbf{X}_{i,S}\mathbf{X}_{i,S}^T))\leq c_1$, where $\lambda_{\max}(\cdot)$ is the largest eigenvalue of a matrix and $c_1>0$ is a constant.

(A4) The densities of $\mathbf{X}_{i,S^*}^T\boldsymbol{\beta}_{S^*}^*$ conditioning on $Y=1$ and $Y=-1$ are uniformly bounded away from zero and infinity at the neighborhood of $\mathbf{X}_{i,S^*}^T\boldsymbol{\beta}_{S^*}^*=1$ and $\mathbf{X}_{i,S^*}^T\boldsymbol{\beta}_{S^*}^* = -1$, respectively.

(A5) $M_n = O(n^\kappa)$ for some constant $0<\kappa<1/2$.

(A6) $p_n = O(\exp(n^\gamma))$ for some constant $0<\gamma<(1-2\kappa)/5$.

(A7) For all $S\in\{S:|S|\leq M_n, S\supseteq S^*\}$, there exist some positive constants c_2 and c_3 such that $\lambda_{\min}(\mathbf{H}(\boldsymbol{\beta}_S))\geq c_2$ and $\lambda_{\max}(\mathbf{H}(\boldsymbol{\beta}_S)) = O(|S|)$ over the set $\{\boldsymbol{\beta}:||\boldsymbol{\beta}-\boldsymbol{\beta}^*||\leq c_3\}$, where $\lambda_{\min}(\cdot)$ is the smallest eigenvalue of a matrix.

(A8) For all $S\in\{S:|S|\leq M_n, S\supseteq S^*\}$, $\lambda_{\max}(\mathbf{H}(\boldsymbol{\beta}_S^*))\log(p_n) = o(L_n\log(n))$, $L_n\log(n) = o(n)$.

Conditions (A1) is required so that $\mathbf{S}(\boldsymbol{\beta})$ and $\mathbf{H}(\boldsymbol{\beta})$ are well-defined, see Koo et al. (2008) for more details. Condition (A2) is assumed in the literature of high dimensional model selection consistency as in Wang et al. (2012) and Lee et al. (2014). Condition (A3) on the largest eigenvalue is similar to the sparse Riesz condition (Zhang and Huang, 2008) and is often assumed for model selection consistency in the diverging p scenario (Chen and Chen, 2008; Yuan, 2010; Zhang, 2010). Note that the lower bound on the eigenvalue of the covariance matrix of $\mathbf{X}_S$ is not specified. Condition (A4) assumes that as the sample size increases, there is enough information around the non-differentiable point of the hinge loss function. This condition is also required for model selection consistency of non-convex penalized SVM in high dimensions (Zhang et al., 2014). Condition (A6) specifies that p is allowed to diverge at an exponential rate of n. Conditions (A7) requires that the Hessian matrix is well-behaved. More specifically, (A7) requires a lower bound on the smallest eigenvalue of the Hessian matrix in the neighborhood of the true value. Koo et al. (2008) gave sufficient conditions for the positive-definiteness of the Hessian matrix at the true value and showed these conditions hold under the setting of Fisher's linear discriminant analysis with a fixed model size. Condition (A8) specifies the rate requirement of L_n in (4).

3.2 Consistency of SVMIC$_L$ for a fixed p

In this section we assume that the dimension p is fixed and study the theoretical properties of SVMIC$_L$. Let $\Omega = \{S:|S|\leq M\}$ be the candidate model space where M is a positive number. Furthermore, when p is fixed, the total number of candidate models is also fixed.

Note that, to prove the model selection consistency of SVMIC_L, we need to show that

$$\Pr(\inf_{S\in\Omega, S\neq S^*} \text{SVMIC}_L(S) > \text{SVMIC}_L(S^*)) \to 1$$

as $n \to \infty$. By the fact that Ω is a fixed model space, it is sufficient to show the result point-wisely in the model space, that is,

$$\Pr(\text{SVMIC}_L(S) > \text{SVMIC}_L(S^*)) \to 1 \tag{6}$$

for every $S \in \{S : S \in \Omega, S \neq S^*\}$. This point-wise version greatly simplifies the proof. Recall that $\widehat{\boldsymbol{\beta}}_S = \arg\min\{1/n \sum_{i=1}^n (1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S)_+ + \lambda_n/2 ||\boldsymbol{\beta}_S^+||^2\}$. As we will show in the appendix, it suffices to conclude (6) with the condition that $\widehat{\boldsymbol{\beta}}_S$ is a consistent estimator of $\boldsymbol{\beta}_S^*$ whenever the candidate subset S includes the true subset S^*. By Theorem 1 of Koo et al. (2008), we have $||\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*|| = O_p(n^{-1/2})$ for every fixed $S \supset S^*$ under the assumption of fixed p. Therefore the point-wise consistency holds. The result is summarized in Lemma 1.

Lemma 1 *Assuming p is a fixed number and $\lambda_n = 1/n$. Under conditions (A1)-(A4) and (A6)-(A7), we have*

$$\Pr(\widehat{S} = S^*) \to 1$$

as $n \to \infty$, where $\widehat{S} = \arg\min_{S:|S|\leq M} SVMIC_L(S)$.

3.3 Consistency of SVMIC_H for a diverging p

The proof becomes much more involved when p is diverging, especially when p diverges much faster than $O(\sqrt{n})$. Let $\Omega = \{S : |S| \leq M_n\}$, $\Omega_+ = \{S : |S| \leq M_n, S \supset S^*, S \neq S^*\}$ and $\Omega_- = \{S : |S| \leq M_n, S \not\supset S^*\}$, where Ω_+ and Ω_- are spaces of overfitted and underfitted models, respectively. Though the information criterion for the fixed model space can differentiate the true model from an arbitrary candidate model, this point-wise result is not sufficient for the overall consistency if the problem requires searching uniformly over a diverging model space. That is, even if (6) holds for every $S \in \Omega$, we still cannot conclude model selection consistency, as the probability of favoring an overfitted or underfitted candidate model rather than the true model can accumulate at very fast speed if the number of candidate models is diverging and hence lead to inconsistent model selection. To control the overall failing probability, we need a uniform convergence rate of SVM solution $\widehat{\boldsymbol{\beta}}_S$ over the diverging model space Ω. Note that $\Omega = \Omega_+ \cup \{S^*\} \cup \Omega_-$. It turns out that the uniform convergence rate of $\widehat{\boldsymbol{\beta}}_S$ over $S \in \Omega_+$ is sufficient for the technical proof. We summarize the uniform rate in Lemma 2 below.

Lemma 2 *Under conditions (A1)-(A7) and $\lambda_n = 1/n$, we have*

$$\sup_{S:|S|<M_n, S\supset S^*} ||\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*|| = O_p(\sqrt{|S| \log(p)/n}).$$

This uniform convergence rate of SVM solution is far from being a trivial result. Recently Zhang et al. (2014) showed that $||\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*|| = O_p(\sqrt{|S|/n})$ for a specific diverging model S which satisfies $S \supset S^*$ and $|S| = o(\sqrt{n})$. Although it is an extension of Theorem 1 in Koo et al. (2008) to the diverging p case, it is still only a point-wise result and cannot be applied directly to bound the overall failing probability. Not surprisingly, the uniform convergence rate in Lemma 2 is slower by a factor $\sqrt{\log(p)}$, which is the price we pay to search over the candidate model space uniformly. In fact, this additional term is the main reason for adding the extra penalty L_n in SVMIC_H.

We now give an intuitive explanation why SVMIC_L can fail in the diverging model space. Consider all the overfitted models in Ω_+. We have the following decomposition

$$\begin{aligned}
&\inf_{S\in\Omega_+} \text{SVMIC}_L(S) - \text{SVMIC}_L(S^*) \\
&= \inf_{S\in\Omega_+} \Big\{ \sum_{i=1}^n (1 - Y_i \mathbf{X}_{i,S}^T \widehat{\boldsymbol{\beta}}_S)_+ - \sum_{i=1}^n (1 - Y_i \mathbf{X}_{i,S^*}^T \boldsymbol{\beta}_S^*)_+ + (|S| - |S^*|) \log(n) \Big\}.
\end{aligned} \tag{7}$$

Note that the difference of the sum of hinge loss can be negative and the difference of model size is always positive. We will show in the appendix that the difference of the sum of hinge loss is of order $O_p(n||\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*||^2)$ under some regularity conditions. For fixed p, it implies that the difference of model size dominates for large n and the sign of (7) is always positive in the limit. For diverging p, however, this is not always the case. By Lemma 2, the difference of hinge loss in (7) is of order $O(|S| \log(p))$ and the difference of model size is of order $O(|S| \log(n))$, thus the sign of (7) can be negative in the limit. Therefore even if we have a very large sample size, SVMIC_L may still favoring the models that are overfitted due to the slower uniform convergence rate of SVM solution. Because SVMIC_L can be viewed as directly following the spirit of ordinary BIC, this result agrees with the findings reported in Chen and Chen (2008) that BIC can be too liberal in high dimensional model selection. Here by liberal, we mean that there is a positive probability that an overfitted model is more favored than the true model by the information criterion even with an infinite sample size.

For SVMIC_H, we can do a similar decomposition as in (7) for all the overfitted models

$$\begin{aligned}
&\inf_{S\in\Omega_+} \text{SVMIC}_H(S) - \text{SVMIC}_H(S^*) \\
&= \inf_{S\in\Omega_+} \Big\{ \sum_{i=1}^n (1 - Y_i \mathbf{X}_{i,S}^T \widehat{\boldsymbol{\beta}}_S)_+ - \sum_{i=1}^n (1 - Y_i \mathbf{X}_{i,S^*}^T \boldsymbol{\beta}_S^*)_+ + L_n(|S| - |S^*|) \log(n) \Big\}.
\end{aligned} \tag{8}$$

Note that the extra term L_n diverges to infinity and the sign of (8) in the limit is determined by the difference of model size, which is always positive. That is, SVMIC_H can identify the true model from all the overfitted models for sufficiently large n. This result is summarized in Lemma 3.

Lemma 3 *Under conditions (A1)-(A8) and $\lambda_n = 1/n$, we have*

$$\Pr(\inf_{S:S\in\Omega_+} SVMIC_H(S) > SVMIC_H(S^*)) \to 1.$$

as $n \to \infty$.

To conclude the model selection consistency, we also need to consider all the underfitted models. This requires a different analysis because for every underfitted model $S \in \Omega_-$, the

difference of the model size $|S| - |S^*|$ can be negative and thus the decomposition in (7) is not helpful. However, one can always add relevant features to the underfitted model and study the enlarged model instead. To be more specific, for every $S \in \Omega_-$, one can always create the enlarged model $\tilde{S}$ such that $\tilde{S} = S \cup S^*$. Note that $\tilde{S}$ is either an overfitted model or the true model. The model $\tilde{S}$ which includes all the signals bridges the underfitted and overfitted model space through the simple fact

$$\begin{aligned}&\inf_{S:S\in\Omega_-} \mathrm{SVMIC}_H(S) - \mathrm{SVMIC}_H(S^*)\\ =&\inf_{S:S\in\Omega_-} \{[\mathrm{SVMIC}_H(S) - \mathrm{SVMIC}_H(\tilde{S})] + [\mathrm{SVMIC}_H(\tilde{S}) - \mathrm{SVMIC}_H(S^*)]\}.\end{aligned}$$

By Lemma 3, in the limit the difference $\mathrm{SVMIC}_H(\tilde{S}) - \mathrm{SVMIC}_H(S^*)$ is non-negative for every $S \in \Omega_-$ (could be exactly 0). Note also that the difference between S and $\tilde{S}$ is at least one missing relevant feature. According to the assumption that the signals do not diminish to 0 as sample size increases, one can show the model with more signals always produces a strictly smaller sum of hinge loss in the limit. That is, for sufficiently large n, we always have

$$\sum_{i=1}^{n}(1 - Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+ - \sum_{i=1}^{n}(1 - Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{\tilde{S}})_+ \geq C > 0$$

for every $S \in \Omega_-$ and some constant C does not depend on S. Then we arrive at the following result for the underfitted model space.

Lemma 4 *Under conditions (A1)-(A8) and $\lambda_n = 1/n$, we have*

$$\Pr(\inf_{S:S\in\Omega_-} SVMIC_H(S) > SVMIC_H(S^*)) \to 1.$$

as $n \to \infty$.

By combing Lemma 3 and Lemma 4, we can conclude the model selection consistency of SVMIC_H in the diverging model space.

Theorem 5 *Under conditions (A1)-(A8) and $\lambda_n = 1/n$, we have*

$$\Pr(\hat{S} = S^*) \to 1.$$

as n, $p \to \infty$, where $\widehat{S} = \arg\min_{S:|S|\leq M_n} SVMIC_H(S)$.

3.4 Application to tuning parameter selection in penalized SVMs

Theorem 1 states that SVMIC_H can identify the true model from Ω. However, in practice it can be very time-consuming and even infeasible to calculate $\mathrm{SVMIC}_H(S)$ for every $S \in \Omega$. One possible approach is to form a solution path via penalized SVM and only consider the candidate models on the path. The idea of using solution path has been shown to greatly reduce the computation burden, see Mazumder et al. (2011). For the solution path of penalized SVM, Hastie et al. (2004) studied the L_1 penalized SVM and showed that the solution path is piece-wise linear in C which is the regularization parameter in (1).

Model selection on the solution path is essentially a tuning parameter selection problem. Recently, several methods have been proposed for choosing the tuning parameter based on the BIC-type information criterion, including Wang et al. (2009) for penalized linear regression, Kawano (2012) for bridge regression, Lee et al. (2014) for penalized quantile regression and Fan and Tang (2013) for penalized generalized linear model. Following the ideas therein, we propose to choose the shrinkage level of penalized SVMs based on the modified support vector machine information criterion. Let $\widehat{\boldsymbol{\beta}}_{\lambda_n}^T = (\hat{\beta}_{\lambda_n,0}, \hat{\beta}_{\lambda_n,1}, \ldots, \hat{\beta}_{\lambda_n,p})$ be the solution to some penalized SVM with a tuning parameter λ_n. That is,

$$\widehat{\boldsymbol{\beta}}_{\lambda_n} = \arg\min_{\boldsymbol{\beta}} \{\frac{1}{n}\sum_{i=1}^{n}(1 - Y_i\mathbf{X}_i^T\boldsymbol{\beta})_+ + \sum_{j=1}^{p} p_{\lambda_n}(\beta_j)\}, \tag{9}$$

where $p_{\lambda_n}(\cdot)$ is some penalty function with a tuning parameter λ_n. Denote $\widehat{S}_{\lambda_n} = \{j : 1 \leq j \leq p, \hat{\beta}_{\lambda_n,j} \neq 0\}$. We define the information criterion for choosing tuning parameter λ_n as

$$\mathrm{SVMIC}_H(\lambda_n) = \sum_{i=1}^{n}(1 - Y_i\mathbf{X}_i^T\widehat{\boldsymbol{\beta}}_{\lambda_n})_+ + L_n|\widehat{S}_{\lambda_n}|\log(n),$$

where L_n is defined in $\mathrm{SVMIC}_H(S)$. The selected tuning parameter is the one that minimizes the information criterion and results in the model size within the restricted model space, that is,

$$\widehat{\lambda}_n = \arg\min_{\lambda:|\widehat{S}_\lambda|\leq M_n} \mathrm{SVMIC}_H(\lambda).$$

This information criterion for selecting tuning parameter can be applied to various penalized approaches for sparse SVMs. Note that the feature selection consistency of the SCAD penalized SVM is shown to rely on the proper choice of the tuning parameter (Zhang et al., 2014), where resampling procedure such as five-fold cross-validation is commonly used in practice. As we will also show in our numerical findings in Section 4.2, the proposed information criterion $\mathrm{SVMIC}_H(\lambda_n)$ usually select the shrinkage level that leads to the correct model size. The tuning parameter selected by cross-validation, however, is more likely to be under-penalized and lead to an overfitted model. Furthermore, the cross-validation is more computationally intensive than information criterion and hence less desirable when the number of features is large.

4. Simulations

In this section we study the finite-sample performance of SVMIC_H. We are interested in the model selection ability of $\mathrm{SVMIC}_H(S)$ and the tuning parameter selection ability of $\mathrm{SVMIC}_H(\lambda_n)$. We also examine the effect of different choices of L_n in the definition of SVMIC_H. For all simulations, we consider the rates $\log(\log(n))$, $\sqrt{\log(n)}$, $\log(n)$ and $n^{-1/3}$ for L_n. We compare with SVMIC_L in Claeskens et al. (2008) and the extended Bayesian information criterion (EBIC) proposed in Chen and Chen (2008). Note that EBIC is originally proposed for model selection in the diverging model space in the framework of regression and it has not been applied into classification problem. However, the main strategy therein is to add an additional term $\log(\binom{p}{|S|})\log(n)$ in the BIC penalty, where $\binom{p}{|S|}$ is the number of $|S|$ combinations chosen from p items. We modify EBIC for model

selection of SVM by selecting the model S that minimizes the criterion

$$\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+ + |S|\log(n) + \log\left(\binom{p}{|S|}\right)\log(n).$$

This modification essentially follows the idea in Chen and Chen (2008) and we are interested in its finite-sample performance compared with SVMIC$_H$.

To investigate these issues, we conduct the SCAD penalized SVM, which has been shown to enjoy the model selection consistency for a properly chosen tuning parameter (Zhang et al., 2014). That is, given a specific λ_n, we solve (9) with $p_{\lambda_n}(\cdot)$ being the SCAD penalty defined in Fan and Li (2001). The corresponding optimization problem is a non-convex one, for which the local linear approximation (LLA) algorithm (Zou and Li, 2008) is implemented in all our numerical studies. To be more specific, for step $t \geq 1$, given the solution $\widehat{\boldsymbol{\beta}}^{(t-1)} = (\hat{\beta}_0^{(t-1)}, \ldots, \hat{\beta}_p^{(t-1)})^T$ at the previous step, we update by solving

$$\widehat{\boldsymbol{\beta}}^{(t)} = \arg\min_{\boldsymbol{\beta}}\{\frac{1}{n}\sum_{i=1}^{n}(1-Y_i\mathbf{X}_i^T\boldsymbol{\beta})_+ + \sum_{j=1}^{p}p'_{\lambda_n}(|\hat{\beta}_j^{(t-1)}|)|\beta_j|\},$$

where $p'_{\lambda_n}(\cdot)$ denotes the derivative of $p_{\lambda_n}(\cdot)$. Note that each update step can be easily recast as a linear programming (LP) problem and efficiently solved by many popular solvers. In this paper we take the initial value $\{\widehat{\boldsymbol{\beta}}^{(0)} : \hat{\beta}_j^{(0)} = 0, 0 \leq j \leq p\}$ and claim convergence if the value $||\widehat{\boldsymbol{\beta}}^{(t-1)} - \widehat{\boldsymbol{\beta}}^{(t)}||$ is small enough.

4.1 Model selection of SVMIC$_H(S)$

In this subsection we study the model selection ability of SVMIC$_H(S)$. The data are generated from two models. The first model is adapted from Fisher's linear discriminant analysis (LDA) and the second model is related to probit regression.

- Model 1: $\Pr(Y=1) = \Pr(Y=-1) = 0.5$, $\mathbf{X}|(Y=1) \sim MN(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\mathbf{X}|(Y=-1) \sim MN(-\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $q=4$, $\boldsymbol{\mu} = (0.25, 0.25, 0.25, 0.25, 0, \ldots, 0)^T \in \mathbf{R}^p$, $\boldsymbol{\Sigma} = (\sigma_{ij})$ with nonzero elements $\sigma_{ii} = 1$ for $i = 1, 2, \cdots, p$ and $\sigma_{ij} = \rho = -0.2$ for $1 \leq i \neq j \leq q$. The Bayes rule is given by sign$(X_1+X_2+X_3+X_4)$ with Bayes error 21.4%.

- Model 2: $\mathbf{X} \sim MN(\mathbf{0}_p, \boldsymbol{\Sigma})$, $\boldsymbol{\Sigma} = (\sigma_{ij})$ with nonzero elements $\sigma_{ii} = 1$ for $i = 1, 2, \cdots, p$ and $\sigma_{ij} = \rho = 0.4^{|i-j|}$ for $1 \leq i \neq j \leq q$, $\Pr(Y=1) = \Phi(\mathbf{X}^T\boldsymbol{\beta})$ where $\Phi(\cdot)$ is the CDF of the standard normal distribution, $q=4$, $\boldsymbol{\beta} = (0.8, 0.8, 0.8, 0.8, 0, \ldots, 0)^T$. The Bayes rule is sign$(0.57X_1+0.34X_2+0.34X_3+0.57X_4)$ with Bayes error 11.5%.

For both models, we construct the solution path using SCAD penalized SVM for candidate models with $|S| \leq M_n = 50$. Our goal is to check how different information criteria evaluate and select the optimal model from all the candidate models on the solution path. We consider three different (n, p) combinations with p ranging from 2000 to 4000 and n is only one tenth of p. Note that Model 1 is a very noisy model with high Bayes error and Model 2 is less noisy but with moderate correlation between the relevant features. We use 200 replications to see the variations of the results. The columns "Correct", "Underfit" and "Overfit" summarize the percentages over 200 replications for correct model selection, overfitting and underfitting, respectively. The numbers under columns "Signal" and "Noise" are the average numbers of selected relevant and irrelevant features, respectively. We also generate an independent dataset with sample size 10000 to evaluate the test error. Numbers in parentheses are the corresponding standard errors.

Table 1 summarizes the model selection results of SVMIC$_L$, SVMIC$_H$ and the criterion proposed in Chen and Chen (2008) for Model 1. For all (n, p) combinations, SVMIC$_H$ shows uniformly higher percentages to identify the correct model than SVMIC$_L$ regardless of the choices of L_n. It can be seen that in the cases p is much larger than n, SVMIC$_L$ behaves too liberal and tends to select an overfitted model. Note that SVMIC$_H$ also has a significantly lower testing error than SVMIC$_L$ in all settings even when SVMIC$_L$ includes slightly more signals in the model. This agrees with the findings in Fan and Fan (2008) that the accumulation of the noises can greatly blur the prediction power. Though the SVMIC$_H$ with different L_n all performs better than SVMIC$_L$, their performances are not exactly the same. For the criteria with a more aggressive penalty on the model size such as $\log(n)$ and $n^{-1/3}$, there are considerable underfitting when the sample size is small ($n = 200$). As the sample size increases, the difference of L_n decreases. This suggests that although asymptotically the choice of L_n can lie in a wide range of spectrums, for small sample sizes some choices of L_n can be too conservative and may not be much better than SVMIC$_L$ which is too liberal. In general, we find $L_n = \sqrt{\log(n)}$ seems to be a reasonable choice for many scenarios.

Another interesting finding is the comparison to the criterion following the spirit in Chen and Chen (2008). Though its theoretical property has not been investigated, the empirical results suggest that it performs similar to SVMIC$_H$ with $L_n = \sqrt{\log(n)}$ in finite samples. In fact, by using the approximation that $\binom{p}{|S|} \approx p^{|S|}$ when p is much larger than $|S|$, one can easily show that

$$\log(n)|S|\log(\log(n)) < \log(n)|S| + \log(n)\log\left(\binom{p}{|S|}\right) < \log(n)|S|\log(n)$$

for $n < p < 10^3 n$ and a very wide range of n. This provides some evidence that the criterion directly adapted from Chen and Chen (2008) behaves more libearal than SVMIC$_H$ with $L_n = \log(n)$ but is more aggressive than SVMIC$_H$ with $L_n = \log(\log(n))$.

Table 2 summarizes the model selection results for Model 2. The SVMIC$_H$ with $\log(\log(n))$ and $\sqrt{\log(n)}$ as L_n perform uniformly better than SVMIC$_L$ and the criterion in Chen and Chen (2008) for all scenarios. Due to the correlations among the signals, the more aggressive choices of L_n suffer from considerable underfitting. Again our empirical results suggest that $L_n = \sqrt{\log(n)}$ seems to be an appropriate choice for a wide range of problems.

4.2 Tuning parameter selection of SVMIC$_H(\lambda)$

In this subsection we examine the tuning parameter selection ability of SVMIC$_H(\lambda)$. The data is generated from the following model.

- $\Pr(Y=1) = \Pr(Y=-1) = 0.5$, $\mathbf{X}|(Y=+1) \sim MN(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $X|(Y=-1) \sim MN(-\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $q=5$, $\boldsymbol{\mu} = (0.1, 0.2, 0.3, 0.4, 0.5, 0\ldots, 0) \in \mathbf{R}^p$, $\boldsymbol{\Sigma} = (\sigma_{ij}) = \mathbf{1}_p$ nonzero

Table 1: Simulation results for Model 1 over 200 replications

Method	C(%)	O(%)	U(%)	Signal	Noise	Test Error(%)
		$n = 200,\ p = 2000$				
SVMIC_L	0.0	100.0	0.0	4.0	11.1	30.7(0.3)
$\text{SVMIC}_H(L_n = \log(\log(n)))$	34.0	64.0	2.0	3.8	1.6	25.4(0.3)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	48.0	38.0	14.0	3.6	1.0	26.1(0.4)
$\text{SVMIC}_H(L_n = \log(n))$	31.5	23.5	45.0	3.0	0.8	29.6(0.5)
$\text{SVMIC}_H(L_n = n^{-1/3})$	29.5	23.5	47.0	2.9	0.8	29.7(0.5)
Chen&Chen	47.0	26.5	26.5	3.3	0.8	27.5(0.5)
		$n = 300,\ p = 3000$				
SVMIC_L	2.0	98.0	0.0	4.0	14.1	28.7(0.3)
$\text{SVMIC}_H(L_n = \log(\log(n)))$	69.0	31.0	0.0	4.0	0.4	22.5(0.1)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	93.0	6.5	0.5	4.0	0.1	22.1(0.1)
$\text{SVMIC}_H(L_n = \log(n))$	69.5	3.5	27.0	3.5	0.1	25.1(0.4)
$\text{SVMIC}_H(L_n = n^{-1/3})$	53.5	3.5	43.0	3.3	0.1	26.6(0.4)
Chen&Chen	95.0	4.5	0.5	4.0	0.1	22.1(0.1)
		$n = 400,\ p = 4000$				
SVMIC_L	4.0	96.0	0.0	4.0	12.6	26.9(0.3)
$\text{SVMIC}_H(L_n = \log(\log(n)))$	77.5	22.5	0.0	4.0	0.3	22.2(0.1)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	95.5	4.5	0.0	4.0	0.1	21.9(0.1)
$\text{SVMIC}_H(L_n = \log(n))$	95.0	1.0	4.0	3.9	<0.1	22.4(0.1)
$\text{SVMIC}_H(L_n = n^{-1/3})$	71.0	1.0	28.0	3.5	<0.1	25.0(0.4)
Chen&Chen	98.5	1.0	0.5	4.0	<0.1	22.0(0.1)

elements $\sigma_{ii} = 1$ for $i = 1, 2, \cdots, p$ and $\sigma_{ij} = \rho = -0.2$ for $1 \le i \ne j \le q$. The Bayes rule is $\text{sign}(2.67X_1+2.83X_2+3X_3+3.17X_4+3.33X_5)$ with Bayes error: 6.3%.

We consider $p = 2000$ and 3000 and $n = 10^{-1}p$. Once the data is generated, we construct the solution path of SCAD penalized SVM on a fine grid of λ for candidate models with $|S| \le M_n = 50$. We then choose the best λ based on the definition of $\text{SVMIC}_H(\lambda)$. Similarly as the simulations for model selection, we compare with $\text{SVMIC}_L(\lambda)$ and the criterion in Chen and Chen (2008). We also implement five-fold cross-validation (denoted by 5-CV) and an adjusted version of five-fold cross-validation version (denoted by 5-CV Adj.). The adjusted 5-CV selects the most parsimonious model with MSE less than one standard error above the regular 5-CV. It is known that the adjusted 5-CV performs better than regular 5-CV in terms of selection consistency. An independent dataset with sample size 10000 is generated to evaluate the test error. This procedure is repeated for 100 replications to study the variations of the results.

Table 3 summarizes the tuning parameter selection results. It can be seen that the tuning parameter selected by SVMIC_L often leads to seriously overfitted models. As the sample size increases, SVMIC_H with all choices of L_n have a much higher chance to identify the correct model than SVMIC_L. The tuning parameter selected by SVMIC_H with $L_n = \sqrt{\log(n)}$ seems to give the most appropriate level of regularization to the model. It is not surprising that this SVMIC_H leads to great prediction power in these high dimensional cases. The

Table 2: Simulation results for Model 2 over 200 replications

Method	C(%)	O(%)	U(%)	Signal	Noise	Test Error(%)
		$n = 200,\ p = 2000$				
SVMIC_L	3.0	96.0	1.0	3.8	5.2	19.2(0.2)
$\text{SVMIC}_H(L_n = \log(\log(n)))$	31.0	31.5	37.5	3.3	0.5	17.2(0.2)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	25.0	2.5	72.5	3.0	0.1	18.0(0.2)
$\text{SVMIC}_H(L_n = \log(n))$	0.0	0.0	100.0	1.9	0.0	22.1(0.2)
$\text{SVMIC}_H(L_n = n^{-1/3})$	0.0	0.0	100.0	1.9	0.0	22.1(0.2)
Chen&Chen	1.0	0.0	99.0	2.2	0.0	20.4(0.2)
		$n = 300,\ p = 3000$				
SVMIC_L	6.5	93.5	0.0	4.0	7.4	18.0(0.2)
$\text{SVMIC}_H(L_n = \log(\log(n)))$	65.0	23.0	12.0	3.8	0.3	15.0(0.1)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	70.5	4.0	25.5	3.7	<0.1	15.3(0.1)
$\text{SVMIC}_H(L_n = \log(n))$	0.0	0.0	100.0	2.0	0.0	21.0(0.1)
$\text{SVMIC}_H(L_n = n^{-1/3})$	0.0	0.0	100.0	2.0	0.0	21.5(0.2)
Chen&Chen	33.5	0.5	66.0	3.1	<0.1	17.3(0.2)
		$n = 400,\ p = 4000$				
SVMIC_L	9.0	91.0	0.0	4.0	6.9	17.2(0.2)
$\text{SVMIC}_H(L_n = \log(\log n))$	82.0	16.5	1.5	4.0	0.2	14.5(0.1)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	89.0	2.5	8.5	3.9	<0.1	14.5(0.1)
$\text{SVMIC}_H(L_n = \log(n))$	0.0	0.0	100.0	2.2	0.0	20.1(0.1)
$\text{SVMIC}_H(L_n = n^{-1/3})$	0.0	0.0	100.0	2.1	0.0	20.8(0.1)
Chen&Chen	74.0	0.5	25.5	3.7	<0.1	15.2(0.1)

performances of five-fold cross-validation and its adjusted version are slightly worse than those of SVMIC_H with L_n fixed at $\log(\log(n))$ and $\sqrt{\log(n)}$. Notice that the computation burden of selecting tuning parameter via information criterion is much lower than cross-validation. This makes our proposed information criteria desirable especially in the case with very large p.

5. Real data examples

5.1 MAQC-II breast cancer data

In this section we consider a real-world example from the breast cancer dataset which is part of the MicroArray Quality Control (MAQC)-II project. The preprocessed data can be downloaded from GEO databases with accession number GSE20194. There are 278 patient samples in the data and each is described by 22283 genes. Among the 278 samples, 164 patients have positive estrogen receptor (ER) status and 114 have negative ER status. Our goal is to predict the biological endpoint labeled by ER status and pick up the relevant genes.

We randomly choose 50 samples from positive ER status and 50 samples from negative ER status as the training data. The remaining 114 positive and 64 negatives are used for evaluating the prediction error, resulting in a test data of 178 patients. The data are

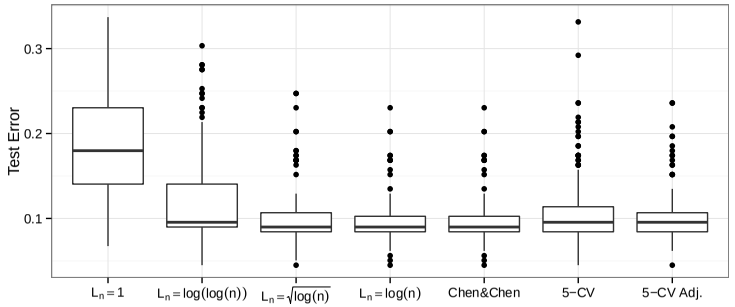
Table 3: Results for tuning parameter selection over 100 replications

Method	C(%)	O(%)	U(%)	Signal	Noise	Test Error(%)
	$n = 200,\ p = 2000$					
SVMIC_L	12	88	0	5.0	2.7	9.6(0.2)
$\text{SVMIC}_H(L_n = \log(\log(n)))$	78	22	0	5.0	0.3	7.4(0.1)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	97	3	0	5.0	<0.1	7.0(0.1)
$\text{SVMIC}_H(L_n = \log(n))$	64	0	36	4.3	0.0	11.5(0.7)
$\text{SVMIC}_H(L_n = n^{-1/3})$	58	0	42	4.1	0.0	12.5(0.8)
Chen&Chen	98	0	2	4.9	0.0	7.2(0.2)
5-CV	44	56	0	5.0	1.5	7.6(0.1)
5-CV Adj.	67	33	0	5.0	0.9	7.5(0.1)
	$n = 300,\ p = 3000$					
SVMIC_L	29	71	0	5.0	3.1	8.8(0.2)
$\text{SVMIC}_H(L_n = \log(\log n))$	94	6	0	5.0	0.1	6.9(0.1)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	100	0	0	5.0	0.0	6.8(0.1)
$\text{SVMIC}_H(L_n = \log(n))$	96	0	4	4.9	0.0	7.1(0.1)
$\text{SVMIC}_H(L_n = n^{-1/3})$	90	0	10	4.8	0.0	7.7(0.3)
Chen&Chen	100	0	0	5.0	0.0	6.8(0.1)
5-CV	58	42	0	5.0	1.0	7.2(0.1)
5-CV Adj.	76	24	0	5.0	0.7	7.1(0.1)

standardized before fitting the classifier. To reduce the computation burden, only 3000 genes with largest absolute values of the two sample t-statistics are used. Such simplification has been considered in Cai and Liu (2011). Though only 3000 genes are used, the classification result is satisfactory. We implement the SCAD penalized SVM to construct the solution path and set the range of λ as $\{2^{-15}, 2^{-14}, \ldots, 2^3\}$. The models on the solution path are selected by SVMIC_L (equivalent to $L_n = 1$), SVMIC_H with L_n at $\log(\log(n))$, $\sqrt{\log(n)}$ and $\log(n)$, the criterion adapted from Chen and Chen (2008), five-fold cross-validations and its adjusted version. This procedure is repeated for 200 replications. The corresponding standard errors are summarized in parentheses. Notice that the 3000 genes with the largest absolute values of t-statistics are pre-selected only using the training data to avoid overfitting so they may be different across the 200 random partitions of the data.

Table 4 summarizes the averages and standard errors for MAQC-II breast cancer data. The criterion $\text{SVMIC}_H(\lambda)$ performs uniformly better than $\text{SVMIC}_L(\lambda)$ regardless of the choice of L_n. It can be easily seen that $\text{SVMIC}_L(\lambda)$ leads to overfitted models in this dataset and has a significant higher misclassification rate. This is in accordance with the theoretical findings in Section 3 that SVMIC_L can be too liberal when the sample size is not comparable to the number of features, while SVMIC_H is a consistent model selection criterion. For this dataset, SVMIC_H with L_n at $\sqrt{\log(n)}$ and $\log(n)$ and the criterion from Chen and Chen (2008) perform the best and are slightly better than cross-validation methods. As in previous arguments, the criterion adapted from the EBIC in Chen and Chen (2008) performs similarly as SVMIC_H with L_n between $\log(\log(n))$ and $\log(n)$ for a wide range of combinations of n and p. Figure 1 summarizes the distributions of test errors over the 200 random partitions of the data for different methods. Note that SVMIC_H is a more

Figure 1: Test error for MAQC-II breast cancer datasets over 200 random partitions



stable method than cross-validations across the partitions of the data. Furthermore, cross-validation based on data resampling is more computationally intensive and this discrepancy is expected to increase dramatically if we take all the genes into consideration, which makes the cross-validation less feasible than information criterion method.

Table 4: Results for MAQC-II breast cancer datasets over 200 random partitions

Method	Size	Test Error(%)
SVMIC_L	4.0	18.5(0.4)
$\text{SVMIC}_H(L_n = \log(\log n))$	1.7	11.5(0.4)
$\text{SVMIC}_H(L_n = \sqrt{\log(n)})$	1.2	9.9(0.2)
$\text{SVMIC}_H(L_n = \log(n))$	1.1	9.6(0.2)
Chen&Chen	1.1	9.6(0.2)
5-CV	7.7	10.8(0.3)
5-CV Adj.	5.1	10.1(0.2)

6. Discussion

In this paper we consider model selection information criterion for support vector machines in the diverging model space. We show that the information criterion proposed in Claeskens et al. (2008) is consistent when the number of features is fixed but can be too liberal if the dimensionality is diverging. A new support vector machine information criterion is proposed for model selection in high dimensions. Based on the uniform convergence rate, we prove that the new information criterion enjoys the model selection consistency even when the number of variables diverges exponentially fast with the sample size. We also link this information criterion to tuning parameter selection for penalized support vector machines. The proposed information criterion is more scalable and easier to compute than resampling techniques such as cross-validation. Simulations and real data examples confirm the model

selection consistency and the ability of selecting tuning parameter when the number of features is much larger than the sample size.

There are several issues yet to be investigated. In this paper we assume that the size of the true model is fixed and the smallest signal does not diminish to zero as the sample size increases. Minimum signal condition has been used in many papers including Fan and Peng (2004) and Fan and Lv (2011). It seems that our condition is stronger than theirs. It is possible to relax this condition. We could possibly assume that $q = q_n$ diverges with n such that $q_n = O(n^{a_1})$ for some $0 \le a_1 < 1/2$. Then we can allow the minimum magnitude of the nonzero-signal to diminish to zero at an appropriate rate such as $\min_{1\le j\le q_n} |\beta_j^*| > an^{-(1-a_2)/2}$ for some constant $a > 0$ and $2a_1 < a_2 \le 1$. In general, the condition we impose on $\min_{1\le j\le q_n} |\beta_j^*|$ is intertwined with the conditions on q and the matrix $\mathbf{X}$, which would be the same for any other high-dimensional regression problem. For a detailed discussion on the beta-min condition in the setting of Lasso regression, we refer to Section 7.4 of Bühlmann et al. (2011). Another direction of interest is to extend the information criterion to nonlinear support vector machine. It is well known that the linear support vector machine can be easily extended to nonlinear feature space using the "kernel trick". Note that it is possible to extend the results in this paper to reproducing kernel Hilbert space with polynomial kernels. For Gaussian radial basis kernels, however, the direct generalization can be problematic as the corresponding reproducing kernel Hilbert space is infinite dimensional. A refined definition of the size of model will be needed in that case and will lead to a more comprehensive study of support vector machine information criterion.

Acknowledgments

We thank the Action Editor Professor Jie Peng and two referees for very constructive comments and suggestions which have improved the presentation of the paper. Wu's research is partially supported by NSF grant DMS-1055210, NIH/NCI grants P01-CA142538 and R01-CA149569. Wang's research is supported by NSF grant DMS-1308960. Li's research was partially supported by NIH/NIDA grants P50-DA10075 and P50-DA036107. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NSF, NIH, NCI, or NIDA.

Appendix A.

In this appendix we prove the following results from Section 3.2:

Lemma 1 *Assuming p is a fixed number and $\lambda_n = 1/n$. Under conditions (A1)-(A4) and (A6)-(A7), we have*

$$\Pr(\widehat{S} = S^*) \to 1$$

as $n \to \infty$, where $\widehat{S} = \arg\min_{S:|S|\le M} SVMIC_L(S)$. ■

Proof. Under regularity conditions, Koo et al. (2008) showed $\widehat{\boldsymbol{\beta}}_S$ is root-n consistent in fixed p case for every $S \in \{S : |S| \le M_n\}$. This pointwise result is enough for Lemma 1 since the model space is fixed. The proof is then similar to Lemma 3 and Lemma 4 for diverging p with the uniform convergence rate $\sqrt{|S|\log(p)/n}$ substituted by $\sqrt{n}^{-1}$ and thus is omitted here.

Lemma 2 *Under conditions (A1)-(A7) and $\lambda_n = 1/n$, we have*

$$\sup_{S:|S|<M_n, S\supset S^*} ||\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*|| = O_p(\sqrt{|S|\log(p)/n}).$$

■

Proof. Recall that $\widehat{\boldsymbol{\beta}}_S = \arg\min_{\boldsymbol{\beta}_S}\{1/n\sum_{i=1}^n(1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S)_+ + \lambda_n/2||\boldsymbol{\beta}_S^+||^2\}$. We will show that for any $0 < \eta < 1$, there exists a large constant $\triangle$ such that for sufficient large n,

$$\Pr(\inf_{|S|\le M_n, S\supset S^*}\inf_{||\mathbf{u}||=\triangle} l_S(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}) > l_S(\boldsymbol{\beta}_S^*)) > 1 - \eta$$

where $l_S(\boldsymbol{\beta}_S) = 1/n\sum_{i=1}^n(1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S)_+ + \lambda_n/2||\boldsymbol{\beta}_S^+||^2$. By the convexity of the hinge loss, this implies that with probability $1-\eta$, we have $\sup_{S:|S|\le M_n, S\supset S^*} ||\widehat{\boldsymbol{\beta}}_S - \boldsymbol{\beta}_S^*|| \le \triangle\sqrt{|S|\log(p)/n}$ and thus Lemma 2 holds.

Notice that we can decompose $l_S(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}) - l_S(\boldsymbol{\beta}_S^*)$ as

$$\begin{aligned}
&l_S(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}) - l_S(\boldsymbol{\beta}_S^*)\\
=&1/n\sum_{i=1}^n\{(1 - Y_i\mathbf{X}_{i,S}^T(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}))_+ - (1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S^*)_+\}\\
&+ \lambda_n/2||\boldsymbol{\beta}_S^{*+} + \sqrt{|S|\log(p)/n}\mathbf{u}^+||^2 - \lambda_n/2||\boldsymbol{\beta}_S^{*+}||^2. \qquad (10)
\end{aligned}$$

By the fact $||\boldsymbol{\beta}_S^{*+} + \sqrt{|S|\log(p)/n}\mathbf{u}^+||^2 - ||\boldsymbol{\beta}_S^{*+}||^2 \le \triangle|S|\sqrt{\log(p)|S|/n}$ and $\lambda_n = 1/n$, the difference of penalty terms in (10) is $n^{-1}|S|o(1)$. Denote

$$\begin{aligned}
g_{i,S}(\mathbf{u}) =&(1 - Y_i\mathbf{X}_{i,S}^T(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}))_+ - (1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S^*)_+\\
&+\sqrt{|S|\log(p)/n}Y_i\mathbf{X}_{i,S}^T\mathbf{u}\mathbf{1}(1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S^* \ge 0)\\
&+\mathbb{E}[(1 - Y_i\mathbf{X}_{i,S}^T(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}))_+] - \mathbb{E}[(1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S^*)_+].
\end{aligned}$$

It can easily checked that $\mathbb{E}[g_{i,S}(\mathbf{u})] = 0$ for $\{S : |S| \le M_n, S \supset S^*\}$ by the definition of $\boldsymbol{\beta}_S^*$ and $\mathbf{S}(\boldsymbol{\beta}^*) = \mathbf{0}$. Next we consider the difference of hinge loss in (10), which can be further composed as

$$1/n\sum_{i=1}^n\{(1 - Y_i\mathbf{X}_{i,S}^T(\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}))_+ - (1 - Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S^*)_+\} = 1/n(A_n + B_n),$$

where

$$A_n = \sum_{i=1}^{n} g_{i,S}(\mathbf{u})$$

and

$$B_n = \sum_{i=1}^{n} \big\{ -\sqrt{|S|\log(p)/n} Y_i \mathbf{X}_{i,S}^T \mathbf{u}\mathbf{1}(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^* \geq 0) + \mathbb{E}[(1 - Y_i \mathbf{X}_{i,S}^T (\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}))_+] - \mathbb{E}[(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^*)_+] \big\}.$$

The rest of the proof consists of three steps. Step 1 will show

$$\sup_{|S| \leq M_n, S \supset S^*} \sup_{||\mathbf{u}||=\triangle} |A_n| = |S| o_p(1).$$

Step 2 will show $\inf_{|S| \leq M_n, S \supset S^*} \inf_{||\mathbf{u}||=\triangle} B_n$ dominates the terms of order $|S|o_p(1)$. Step 3 will complete the proof by showing $\inf_{|S| \leq M_n, S \supset S^*} \inf_{||\mathbf{u}||=\triangle} B_n > 0$ for sufficient large n and $\triangle$.

Step 1: The main tool to prove this uniform rate is the covering number introduced in Van Der Vaart and Wellner (1996). It suffices to show that

$$\Pr(\sup_{|S| \leq M_n, S \supset S^*} \sup_{||\mathbf{u}||=\triangle} |S|^{-1} |\sum_{i=1}^{n} g_{i,S}(\mathbf{u}|) > \epsilon) \to 0$$

for any $\epsilon > 0$. Notice that the hinge loss satisfies Lipschitz condition and by condition (A3) $\max_i ||\mathbf{X}_{i,S}|| = O_p(\sqrt{|S|}\log(n))$. It can be easily shown that

$$|S|^{-1} g_{i,S}(\mathbf{u}) \leq 3\triangle |S|^{-1} \sqrt{|S|\log(p)/n} \max_i ||\mathbf{X}_{i,S}||$$

and thus $\sup_{|S| \leq M_n, S \supset S^*} \sup_{||\mathbf{u}||=\triangle} |S|^{-1} g_{i,S}(\mathbf{u}) = o_p(1)$. By Lemma 2.5 of van de Geer (2000), the ball $\{\mathbf{u} : ||\mathbf{u}|| \leq \triangle\}$ in $\mathbf{R}^{|S|+1}$ can be covered by N balls with radius δ where $N \leq ((4\triangle + \delta)/\delta)^{|S|+1}$. Denote $\mathbf{u}^1, \ldots, \mathbf{u}^N$ the centers of the N balls. By the fact that $\sup_{|S| \leq M_n, S \supset S^*} \sqrt{|S|\log(p)/n} \max_i ||\mathbf{X}_{i,S}|| = O_p(1)$, we can take $\delta = (nC)^{-1}|S|$ for some large constant C such that

$$\begin{aligned} &\min_{1 \leq k \leq N} \sup_{|S| \leq M_n, S \supset S^*} \sup_{||\mathbf{u}||=\triangle} |S|^{-1} |\sum_{i=1}^{n} g_{i,S}(\mathbf{u}) - \sum_{i=1}^{n} g_{i,S}(\mathbf{u}^k)| \\ \leq &\sup_{|S| \leq M_n, S \supset S^*} 3\triangle n |S|^{-1} \sqrt{|S|\log(p)/n} \max_i ||\mathbf{X}_{i,S}|| \delta \ \leq \ \epsilon/3 \end{aligned} \tag{11}$$

with probability tending to one. Based on (11), it can be easily shown

$$\begin{aligned} &\Pr(\sup_{|S| \leq M_n, S \supset S^*} \sup_{||\mathbf{u}||=\triangle} |S|^{-1} |\sum_{i=1}^{n} g_{i,S}(\mathbf{u})| > \epsilon) \\ \leq &\sum_{|S| \leq M_n, S \supset S^*} \sum_{k=1}^{N} \Pr(|S|^{-1} |\sum_{i=1}^{n} g_{i,S}(\mathbf{u}^k)| > \epsilon/2) \end{aligned}$$

and $\sum_{i=1}^{n} g_{i,S}(\mathbf{u}^k)$ is sum of independent zero-mean random variables. Notice that

$$\begin{aligned} &(1 - Y_i \mathbf{X}_{i,S}^T (\boldsymbol{\beta}_S^* + \sqrt{|S|\log(p)/n}\mathbf{u}))_+ - (1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^*)_+ \\ &+\sqrt{|S|\log(p)/n} Y_i \mathbf{X}_{i,S}^T \mathbf{u}\mathbf{1}(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^* \geq 0) = 0 \end{aligned}$$

when we have $|1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^*| > \sqrt{|S|\log(p)/n} \max_i ||\mathbf{X}_{i,S}|| \triangle$. Thus we have

$$\begin{aligned} &\sum_{i=1}^{n} \mathbb{E}[g_{i,S}(\mathbf{u}^k)]^2 = \sum_{i=1}^{n} \mathrm{Var}(g_{i,S}(\mathbf{u}^k)) \\ \leq &\sum_{i=1}^{n} \mathbb{E}[(2\sqrt{|S|\log(p)/n} Y_i \mathbf{X}_{i,S}^T \mathbf{u}^k)^2 \mathbf{1}(|1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^*| \leq \sqrt{|S|\log(p)/n} \max_i ||\mathbf{X}_{i,S}|| \triangle)]. \end{aligned} \tag{12}$$

By the bounded largest eigenvalue condition in (A3), we have

$$\sum_{i=1}^{n} \mathbb{E}\{[2\sqrt{|S|\log(p)/n} Y_i \mathbf{X}_{i,S}^T \mathbf{u}^k)]^2 \leq C|S|\log(p).$$

By the bounded conditional density condition (A4), we have

$$\Pr(|1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^*| \leq \sqrt{|S|\log(p)/n} \max_i ||\mathbf{X}_{i,S}|| \triangle) \leq C|S| \log n \sqrt{\log(p)/n}.$$

Then based on (12) and Cauchy inequality, we have

$$\sum_{i=1}^{n} \mathbb{E}[g_{i,S}(\mathbf{u}^k)]^2 \leq C|S|^2 \log n (\log(p))^{3/2} n^{-1/2}.$$

Then applying Bernstein inequality and condition (A6), we arrive

$$\begin{aligned} &\sum_{|S| \leq M_n, S \supset S^*} \sum_{k=1}^{N} \Pr(|S|^{-1} |\sum_{i=1}^{n} g_{i,S}(\mathbf{u}^k)| > \epsilon/2) \\ \leq &\exp\{M_n \log(p)\} \exp(N) \exp\{-C(\log(n))^{-1} (\log(p))^{-3/2} n^{1/2}\} \to 0 \end{aligned}$$

as $n \to \infty$. This completes the proof of Step 1.

Step 2: First notice that

$$|\sum_{i=1}^{n} Y_i \mathbf{X}_{i,S}^T \mathbf{u}\mathbf{1}(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^* \geq 0)| \leq (|S|+1)^{1/2} \triangle \max_{0 \leq j \leq p} |\sum_{i=1}^{n} Y_i X_{ij,S} \mathbf{1}(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^* \geq 0)|. \tag{13}$$

Note that $\mathbb{E}[Y_i X_{ij,S} \mathbf{1}(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^* \geq 0)] = 0$ for $0 \leq j \leq p$ by the definition of $\mathbf{S}(\boldsymbol{\beta}^*)$. By Lemma 14.24 of Bühlmann et al. (2011), we also have

$$\max_{0 \leq j \leq p} |\sum_{i=1}^{n} Y_i X_{ij,S} \mathbf{1}(1 - Y_i \mathbf{X}_{i,S}^T \boldsymbol{\beta}_S^* \geq 0)| = O_p(\sqrt{n\log(p)}). \tag{14}$$

By Taylor expansion of hinge loss function at $\boldsymbol{\beta}_S^*$, we have

$$\begin{aligned}&\sum_{i=1}^{n}\{\mathbb{E}[(1-Y_i\mathbf{X}_{i,S}^T(\boldsymbol{\beta}_S^*+\sqrt{|S|\log(p)/n}\mathbf{u}))_+]-\mathbb{E}[(1-Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_S^*)_+]\}\\ =&0.5|S|\log(p)\mathbf{u}^T\mathbf{H}(\boldsymbol{\beta}_S^*+t\sqrt{|S|\log(p)/n}\mathbf{u})\mathbf{u}\end{aligned}\tag{15}$$

for some $0<t<1$. As shown by Koo et al. (2008), under condition (A1) and (A2), $\mathbf{H}(\boldsymbol{\beta})$ is element-wise continuous at $\boldsymbol{\beta}_S^*$, thus

$$\mathbf{H}(\boldsymbol{\beta}_S^*+t\sqrt{|S|\log p/n}\mathbf{u})=\mathbf{H}(\boldsymbol{\beta}_S^*)+o_p(1).$$

It can be easily shown by (13), (14), (15) and condition (A7), $0.5|S|\log(p)\mathbf{u}^T\mathbf{H}(\boldsymbol{\beta}_S^*)\mathbf{u}$ dominates other terms in B_n for sufficient large $\triangle$. This completes the proof of Step 2.

Step 3: Notice that $0.5|S|\log(p)\mathbf{u}^T\mathbf{H}(\boldsymbol{\beta}_S^*)\mathbf{u}>0$ by condition (A7). Recall that the difference of penalty terms in (10) is $n^{-1}|S|o(1)$. Therefore $0.5|S|\log p\mathbf{u}^T\mathbf{H}(\boldsymbol{\beta}_S^*)\mathbf{u}$ dominates all the other terms in (10) for sufficient large n and $\triangle$, which completes the proof.

Lemma 3 *Under conditions (A1)-(A8) and $\lambda_n=1/n$, we have*

$$\Pr(\inf_{S:S\in\Omega_+}SVMIC_H(S)>SVMIC_H(S^*))\to 1.$$

as $n\to\infty$. ■

Proof. By definition we have

$$\begin{aligned}&\inf_{S\in\Omega_+}\mathrm{SVMIC}_H(S)-\mathrm{SVMIC}_H(S^*)\\ =&\inf_{S\in\Omega_+}\{\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+-\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S^*}^T\widehat{\boldsymbol{\beta}}_{S^*})_++(|S|-|S^*|)\log(n)L_n\}.\end{aligned}$$

Similar to the proof of Lemma 2, it can be shown that $|\sum_{i=1}^n(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+-\sum_{i=1}^n(1-Y_i\mathbf{X}_{i,S^*}^T\widehat{\boldsymbol{\beta}}_{S^*})_+|$ is dominated by $|S|\log(p)\mathbf{u}^T\mathbf{H}(\boldsymbol{\beta}_S^*)\mathbf{u}$ with probability tending to one. By conditions (A6)-(A8), we have

$$|\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+-\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S^*}^T\widehat{\boldsymbol{\beta}}_{S^*})_+|<(|S|-|S^*|)\log(n)L_n$$

for sufficient large n. Notice that $\inf_{S\in\Omega_+}|S|-|S^*|>0$, which completes the proof.

Lemma 4 *Under Conditions (A1)-(A8) and $\lambda_n=1/n$, we have*

$$\Pr(\inf_{S:S\in\Omega_-}SVMIC_H(S)>SVMIC_H(S^*))\to 1.$$

as $n\to\infty$. ■

Proof. For $S\in\Omega_-$, consider the set $\tilde{S}$ with additional signals such that $\tilde{S}=S\cup S^*$. Notice

$$\mathrm{SVMIC}_H(S)-\mathrm{SVMIC}_H(S^*)=\mathrm{SVMIC}_H(S)-\mathrm{SVMIC}_H(\tilde{S})+\mathrm{SVMIC}_H(\tilde{S})-\mathrm{SVMIC}_H(S^*)$$

for $S\in\Omega_-$. Since $|S^*|$ does not diverge with n, we have $|\tilde{S}|<2M_n$ for sufficiently large n and it can easily seen that Lemma 3 still holds for $\tilde{S}$ with any $S\in\Omega_-$. Therefore with high probability we have $\mathrm{SVMIC}_H(\tilde{S})-\mathrm{SVMIC}_H(S^*)\geq 0$. Thus it suffices to show

$$\Pr(\inf_{S\in\Omega_-}\{\mathrm{SVMIC}_H(S)-\mathrm{SVMIC}_H(\tilde{S})\}>0)\to 1$$

as $n\to\infty$. Notice that

$$\begin{aligned}&1/n\{\mathrm{SVMIC}_H(S)-\mathrm{SVMIC}_H(\tilde{S})\}\\ =&1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+-1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{\tilde{S}})_++1/n(|S|-|\tilde{S}|)\log(n)L_n\end{aligned}$$

and by condition (A8) $1/n(|S|-|\tilde{S}|)\log(n)L_n\to 0$, it suffices to show

$$\inf_{S\in\Omega_-}\{1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+-1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{\tilde{S}})_+\}\geq C$$

for some constant $C>0$ that does not depend on S.

Recall that $\widehat{\boldsymbol{\beta}}_S=(\beta_{0,S},\beta_{1,S},\ldots,\beta_{|S|,S})^T\in\mathbf{R}^{|S|+1}$. Denote $\widehat{\boldsymbol{\beta}}_{S,\tilde{S}}\in\mathbf{R}^{|\tilde{S}|+1}$ such that the intercept equals to $\beta_{0,S}$, the j-th element equals to $\beta_{j,S}$ if $j\in S$ and 0 if $j\notin S$ for all $j\in\tilde{S}$. Denote also $\delta=\min_{j\in S^*}|\beta_j^*|$ the smallest signal. Then it can be easily seen that $||\widehat{\boldsymbol{\beta}}_{S,\tilde{S}}-\boldsymbol{\beta}_{\tilde{S}}^*||>\delta$. By Lemma 2 we also have $||\widehat{\boldsymbol{\beta}}_{\tilde{S}}-\boldsymbol{\beta}_{\tilde{S}}^*||<\epsilon$ for arbitrary ϵ and sufficient large n. Therefore there exists $\bar{\boldsymbol{\beta}}_{\tilde{S}}=a\widehat{\boldsymbol{\beta}}_{\tilde{S}}+(1-a)\widehat{\boldsymbol{\beta}}_{S,\tilde{S}}$ for some $0<a<1$ such that

$$||\bar{\boldsymbol{\beta}}_{\tilde{S}}-\boldsymbol{\beta}_{\tilde{S}}^*||=\Delta,$$

where Δ is a positive constant such that $\Delta<c_3$ where c_3 is defined in condition (A7). By the definition of $\widehat{\boldsymbol{\beta}}_{\tilde{S}}$ and the convexity of hinge loss function we have

$$\begin{aligned}&1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_++\lambda_n/2||\bar{\boldsymbol{\beta}}_{\tilde{S}}^+||^2\\ <&a\{1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{\tilde{S}})_++\lambda_n/2||\widehat{\boldsymbol{\beta}}_{\tilde{S}}^+||^2\}\\ &+(1-a)\{1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_{S,\tilde{S}})_++\lambda_n/2||\widehat{\boldsymbol{\beta}}_{S,\tilde{S}}^+||^2\}\\ <&1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{S,\tilde{S}})_++\lambda_n/2||\widehat{\boldsymbol{\beta}}_{S,\tilde{S}}^+||^2\\ =&1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_++\lambda_n/2||\widehat{\boldsymbol{\beta}}_S^+||^2.\end{aligned}\tag{16}$$

By $\lambda_n=n^{-1}$ we have

$$\lambda_n/2||\bar{\boldsymbol{\beta}}_{\tilde{S}}^+||^2-\lambda_n/2||\widehat{\boldsymbol{\beta}}_S^+||^2\leq C\lambda_n(||\widehat{\boldsymbol{\beta}}_{\tilde{S}}^+||^2+||\widehat{\boldsymbol{\beta}}_S^+||^2)\to 0\tag{17}$$

as $n \to \infty$. Similar to the proof of Lemma 2, under condition (A6) and (A8), it can be shown

$$1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{\tilde{S}})_+ - 1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+ \le 1/nC|\tilde{S}|\log(p)\lambda_{\max}(\mathbf{H}(\boldsymbol{\beta}_{\tilde{S}}^*)) \to 0 \quad (18)$$

as $n \to \infty$. Notice that

$$\begin{aligned}
&\inf_{S\in\Omega_-}\{1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - 1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+\}\\
\ge& 1/n\Big\{\inf_{S\in\Omega_-} n\mathbb{E}[(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - (1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+]\\
&-\sup_{S\in\Omega_-}\{|\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - \sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+ - n\mathbb{E}[(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - (1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+]|\}\Big\}.
\end{aligned}$$

Similar to the proof of Lemma 2, it can be shown

$$\begin{aligned}
&\sup_{S\in\Omega_-}\{|\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - \sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+ - n\mathbb{E}[(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - (1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+]|\}\\
=&O_p(|\sum_{i=1}^{n}Y_i\mathbf{X}_{i,\tilde{S}}^T(\bar{\boldsymbol{\beta}}_{\tilde{S}}-\boldsymbol{\beta}_{\tilde{S}}^*)\mathbf{1}(1-Y_i\mathbf{X}_{i,S}^T\boldsymbol{\beta}_{\tilde{S}}^*\ge 0)|) = O_p(\sqrt{n|\tilde{S}|\log(p)}). \qquad (19)
\end{aligned}$$

By Taylor expansion of hinge loss function, we have

$$\mathbb{E}[(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\bar{\boldsymbol{\beta}}_{\tilde{S}})_+ - (1-Y_i\mathbf{X}_{i,\tilde{S}}^T\boldsymbol{\beta}_{\tilde{S}}^*)_+] \ge 0.5\lambda_{\min}(\mathbf{H}(\tilde{\boldsymbol{\beta}}_{\tilde{S}}^*))\Delta^2 > 0, \qquad (20)$$

where $\tilde{\boldsymbol{\beta}}_{\tilde{S}}^*$ lies in the set defined in condition (A7). By (16)-(20), we have

$$\inf_{S\in\Omega_-}\{1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,S}^T\widehat{\boldsymbol{\beta}}_S)_+ - 1/n\sum_{i=1}^{n}(1-Y_i\mathbf{X}_{i,\tilde{S}}^T\widehat{\boldsymbol{\beta}}_{\tilde{S}})_+\} \ge 0.5\lambda_{\min}(\mathbf{H}(\boldsymbol{\beta}_{\tilde{S}}^*))\Delta^2 > 0$$

for sufficient large n, which completes the proof.

Theorem 5 *Under conditions (A1)-(A8) and $\lambda_n = 1/n$, we have*

$$\Pr(\hat{S} = S^*) \to 1.$$

as n, $p \to \infty$, where $\widehat{S} = \arg\min_{S:|S|\le M_n} SVMIC_H(S)$. ■

Proof. The proof can be easily checked by combing the results from Lemma 3 and Lemma 4 and thus is omitted here.

References

Hirotugu Akaike. Information theory and an extension of the maximum likelihood principle. In *Second international symposium on information theory*, pages 267–281. Akademinai Kiado, 1973.

Natalia Becker, Grischa Toedt, Peter Lichter, and Axel Benner. Elastic scad as a novel penalization method for svm classification tasks in high-dimensional data. *BMC bioinformatics*, 12(1):138, 2011.

Paul S Bradley and Olvi L Mangasarian. Feature selection via concave minimization and support vector machines. In *ICML*, volume 98, pages 82–90, 1998.

Peter Lukas Bühlmann, Sara A van de Geer, and Sara Van de Geer. *Statistics for high-dimensional data*. Springer, 2011.

Tony Cai and Weidong Liu. A direct estimation approach to sparse linear discriminant analysis. *Journal of the American Statistical Association*, 106(496), 2011.

Jiahua Chen and Zehua Chen. Extended bayesian information criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771, 2008.

Gerda Claeskens, Christophe Croux, and Johan Van Kerckhoven. An information criterion for variable selection in support vector machines. *The Journal of Machine Learning Research*, 9:541–558, 2008.

Jianqing Fan and Yingying Fan. High dimensional classification using features annealed independence rules. *Annals of statistics*, 36(6):2605, 2008.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

Jianqing Fan and Jinchi Lv. Non-concave penalized likelihood with np-dimensionality. *IEEE Transactions on Information Theory*, 57:5467–5484, 2011.

Jianqing Fan and Heng Peng. On non-concave penalized likelihood with diverging number of parameters. *The Annals of Statistics*, 32:928–961, 2004.

Yingying Fan and Cheng Yong Tang. Tuning parameter selection in high dimensional penalized likelihood. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):531–552, 2013.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

Trevor. Hastie, Robert. Tibshirani, and J Jerome H Friedman. *The elements of statistical learning*, volume 1. Springer New York, 2001.

Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. In *Journal of Machine Learning Research*, pages 1391–1415, 2004.

Shuichi Kawano. Selection of tuning parameters in bridge regression models via bayesian information criterion. *Statistical Papers*, pages 1–17, 2012.

Ja-Yong Koo, Yoonkyung Lee, Yuwon Kim, and Changyi Park. A bahadur representation of the linear support vector machine. *The Journal of Machine Learning Research*, 9: 1343–1368, 2008.

Eun Ryung Lee, Hohsuk Noh, and Byeong U Park. Model selection via bayesian information criterion for quantile regression models. *Journal of the American Statistical Association*, 109(505):216–229, 2014.

Rahul Mazumder, Jerome H Friedman, and Trevor Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106(495), 2011.

Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2): 461–464, 1978.

Jun Shao. An asymptotic theory for linear model selection. *Statistica Sinica*, 7(2):221–242, 1997.

Peide Shi and Chih-Ling Tsai. Regression model selectiona residual likelihood approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(2):237–252, 2002.

Sara van de Geer. Empirical processes in m-estimation. cambridge series in statistical and probabilistic mathematics, 2000.

Aad W Van Der Vaart and Jon A Wellner. *Weak Convergence*. Springer, 1996.

Grace Wahba et al. Support vector machines, reproducing kernel hilbert spaces and the randomized gacv. *Advances in Kernel Methods-Support Vector Learning*, 6:69–87, 1999.

Hansheng Wang, Runze Li, and Chih-Ling Tsai. Tuning parameter selectors for the smoothly clipped absolute deviation method. *Biometrika*, 94(3):553–568, 2007.

Hansheng Wang, Bo Li, and Chenlei Leng. Shrinkage tuning parameter selection with a diverging number of parameters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):671–683, 2009.

Lan Wang, Yichao Wu, and Runze Li. Quantile regression for analyzing heterogeneity in ultra-high dimension. *Journal of the American Statistical Association*, 107(497):214–222, 2012.

Li Wang, Ji Zhu, and Hui Zou. The doubly regularized support vector machine. *Statistica Sinica*, 16(2):589, 2006.

Marten Wegkamp and Ming Yuan. Support vector machines with a reject option. *Bernoulli*, 17:1368–1385, 2011.

Jason Weston, Sayan Mukherjee, Olivier Chapelle, Massimiliano Pontil, Tomaso Poggio, and Vladimir Vapnik. Feature selection for svms. In *NIPS*, volume 12, pages 668–674, 2000.

Ming Yuan. High dimensional inverse covariance matrix estimation via linear programming. *The Journal of Machine Learning Research*, 99:2261–2286, 2010.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010.

Cun-Hui Zhang and Jian Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *The Annals of Statistics*, 36(4):1567–1594, 2008.

Hao Helen Zhang, Jeongyoun Ahn, Xiaodong Lin, and Cheolwoo Park. Gene selection using support vector machines with non-convex penalty. *Bioinformatics*, 22(1):88–95, 2006.

Xiang Zhang, Yichao Wu, Lan Wang, and Runze Li. Variable selection for support vector machines in moderately high dimensions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2014.

Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. *Advances in neural information processing systems*, 16(1):49–56, 2004.

Hui Zou and Runze Li. One-step sparse estimates in nonconcave penalized likelihood models. *Annals of statistics*, 36(4):1509, 2008.

Hui Zou and Ming Yuan. The fo-norm support vector machine. *Statistica Sinica*, 18: 379–398, 2008.

Journal of Machine Learning Research 17 (2016) 1-51 Submitted 4/15; Published 4/16

Extremal Mechanisms for Local Differential Privacy

Peter Kairouz KAIROUZ2@ILLINOIS.EDU
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

Sewoong Oh SWOH@ILLINOIS.EDU
Department of Industrial and Enterprise Systems Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61820, USA

Pramod Viswanath PRAMODV@ILLINOIS.EDU
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

Editor: Mehryar Mohri

Abstract

Local differential privacy has recently surfaced as a strong measure of privacy in contexts where personal information remains private even from data analysts. Working in a setting where both the data providers and data analysts want to maximize the utility of statistical analyses performed on the released data, we study the fundamental trade-off between local differential privacy and utility. This trade-off is formulated as a constrained optimization problem: maximize utility subject to local differential privacy constraints. We introduce a combinatorial family of extremal privatization mechanisms, which we call staircase mechanisms, and show that it contains the optimal privatization mechanisms for a broad class of information theoretic utilities such as mutual information and f-divergences. We further prove that for any utility function and any privacy level, solving the privacy-utility maximization problem is equivalent to solving a finite-dimensional linear program, the outcome of which is the optimal staircase mechanism. However, solving this linear program can be computationally expensive since it has a number of variables that is exponential in the size of the alphabet the data lives in. To account for this, we show that two simple privatization mechanisms, the binary and randomized response mechanisms, are universally optimal in the low and high privacy regimes, and well approximate the intermediate regime.

Keywords: local differential privacy, privacy-preserving machine learning algorithms, information theoretic utilities, f-divergences, mutual information, statistical inference, hypothesis testing, estimation

1. Introduction

In statistical analyses involving data from individuals, there is an increasing tension between the need to share data and the need to protect sensitive information about the individuals. For example, users of social networking sites are increasingly cautious about their privacy, but still find it inevitable to agree to share their personal information in order to benefit from customized services such as recommendations and personalized search (Acquisti, 2004; Acquisti and Grossklags, 2007). There is a certain utility in sharing data for both data providers and data analysts, but at the same time, individuals want *plausible deniability* when it comes to sensitive information.

For such applications, there is a natural core optimization problem to be solved. Assuming both the data providers and analysts want to maximize the utility of the released data, how can they do so while preserving the privacy of participating individuals? The formulation and study of a framework that addresses the fundamental tradeoff between utility and privacy is the focus of this paper.

1.1 Local differential privacy

The need for data privacy appears in two different contexts: the *local privacy* context, as in when individuals disclose their personal information (e.g., voluntarily on social network sites), and the *global privacy* context, as in when institutions release databases of information of several people or answer queries on such databases (e.g., US Government releases census data, companies like Netflix release proprietary data for others to test state of the art machine learning algorithms). In both contexts, privacy is achieved by *randomizing* the data before releasing it. We study the setting of local privacy, in which data providers do not trust the data collector (analyst). Local privacy dates back to Warner (1965), who proposed the *randomized response* method to provide plausible deniability for individuals responding to sensitive surveys.

A natural notion of privacy protection is making inference of information beyond what is released hard. *Differential privacy* has been proposed in the global privacy context to formally capture this notion of privacy (Dwork, 2006; Dwork et al., 2006b; Dwork and Lei, 2009). In a nutshell, differential privacy ensures that an adversary should not be able to reliably infer an individual's record in a database, even with unbounded computational power and access to every other record in the database. Recently, Duchi et al. (2013) extended the notion of differential privacy to the local privacy context. Formally, consider a setting where there are n data providers each owning a data X_i defined on an input alphabet $\mathcal{X}$. The X_i's are independently sampled from some distribution P_ν parameterized by ν. A statistical privatization mechanism Q is a conditional distribution that maps $X_i \in \mathcal{X}$ stochastically to $Y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is an output alphabet possibly larger than $\mathcal{X}$. The Y_i's are referred to as the privatized (sanitized) views of X_i's. In a non-interactive setting, the same privatization mechanism Q is used locally by all individuals. This setting is shown in Figure 1 for the special case of $n = 2$. For some non-negative ε, we follow the definition of Duchi et al. (2013) and say that a mechanism Q is *ε-locally differentially private* if

$$\sup_{S\subset\mathcal{Y},x,x'\in\mathcal{X}} \frac{Q(S|x)}{Q(S|x')} \leq e^{\varepsilon}, \tag{1}$$

where $Q(S|x) = \mathbb{P}(Y_i \in S|X_i = x)$ represents the privatization mechanism. This ensures that for small values of ε, given a privatized data Y_i, it is (almost) equally likely to have come from any data, i.e. x or x'. A small value of ε means that we require a high level of privacy and a large value corresponds to a low level of privacy. At one extreme, for $\varepsilon = 0$, the privatized output must be independent of the private data, and on the other extreme, for $\varepsilon = \infty$, the privatized output can be made equal to the private data.

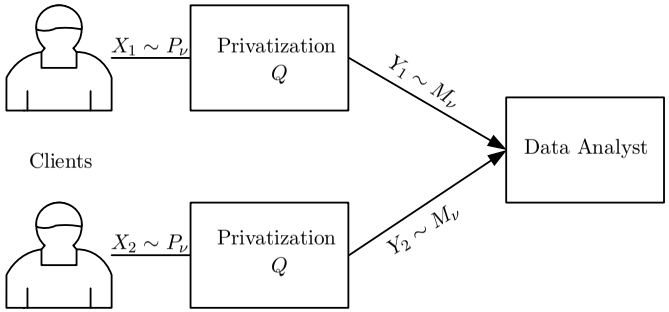


Figure 1: Client i owns X_i sampled from P_ν. Each X_i is privatized by the same ε-locally differentially private mechanism Q. The data analyst only observes the privatized data (Y_i's) and makes an inference on the statistics of the original distribution of the data.

1.2 Information theoretic utilities for statistical analysis

In analyses of statistical databases, the analyst is interested in the *statistics* of the data as opposed to individual records. Naturally, the utility should also be measured in terms of the distribution rather than sample quantities. Concretely, consider a client-server setting where each client with data X_i releases Y_i, a privatized version of the data, via a non-interactive ε-locally differentially private privatization mechanism Q. Assume all the clients use the same privatization mechanism Q, and each client's data is an i.i.d. sample from a distribution P_ν for some parameter ν. Given the privatized views $\{Y_i\}_{i=1}^n$, the data analyst would like to make inferences based on the induced marginal distribution

$$M_\nu(S) \quad \equiv \quad \sum_{x\in\mathcal{X}} Q(S|x)P_\nu(x)\,, \tag{2}$$

for $S \subseteq \mathcal{Y}$. We consider a broad class of convex utility functions, and identify the class of optimal mechanisms, which we call *staircase mechanisms*, in Section 2. We apply this framework to two specific applications: (a) hypothesis testing where the utility is measured in Kullback-Leibler divergence (Section 3) and (b) information preservation where the utility is measured in mutual information (Section 4).

In the binary hypothesis testing setting, $\nu \in \{0,1\}$; therefore, X can be generated by one of two possible distributions P_0 and P_1. The power to discriminate data generated from P_0 to data generated from P_1 depends on the 'distance' between the marginals M_0 and M_1. To measure the ability of such statistical discrimination, our choice of utility of a particular privatization mechanism Q is an information theoretic quantity called Csiszár's f-divergence defined as

$$D_f(M_0||M_1) = \sum_{x\in\mathcal{X}} f\Big(\frac{M_0(x)}{M_1(x)}\Big)\, M_1(x)\,, \tag{3}$$

for some convex function f such that $f(1) = 0$. The Kullback-Leibler (KL) divergence $D_{\mathrm{kl}}(M_0||M_1)$ is a special case with $f(x) = x\log x$, and so is the total variation distance $\|M_0 - M_1\|_{\mathrm{TV}}$ with $f(x) = (1/2)|x-1|$. Such f-divergences capture the quality of statistical inference, such as minimax rates of statistical estimation or error exponents in hypothesis testing (Tsybakov and Zaiats, 2009; Cover and Thomas, 2012). As a motivating example, suppose a data analyst wants to test whether the data is generated from P_0 or P_1 based on privatized views $Y_1, \ldots, Y_n$. According to Chernoff-Stein's lemma, for a bounded type I error probability, the best type II error probability scales as $e^{-n\, D_{\mathrm{kl}}(M_0||M_1)}$. Naturally, we are interested in finding a privatization mechanism Q that minimizes the probability of error by solving the following constraint maximization problem

$$\begin{array}{ll} \underset{Q}{\text{maximize}} & D_{\mathrm{kl}}(M_0\|M_1) \\ \text{subject to} & Q \in \mathcal{D}_\varepsilon \end{array}\,, \tag{4}$$

where $\mathcal{D}_\varepsilon$ is the set of all ε-locally differentially private mechanisms satisfying (1).

In the information preservation setting, X is generated from an underlying distribution P. We are interested in quantifying how much information can be preserved when releasing a private view of the data. In other words, the data provider would like to release an ε-locally differentially private view Y of X that preserves the amount of information in X as much as possible. The utility in this case is measured by the mutual information between X and Y

$$I\left(X;Y\right) = \sum_{\mathcal{X}}\sum_{\mathcal{Y}} P\left(x\right) Q\left(y|x\right) \log\left(\frac{Q\left(y|x\right)}{\sum_{l\in\mathcal{X}} P\left(l\right) Q\left(y|l\right)}\right). \tag{5}$$

Mutual information, as the name suggests, measures the mutual dependence between two random variables. It has been used as a criterion for feature selection and as a measure of similarity between two different clusterings of a data set, in addition to many other applications in signal processing and machine learning. To characterize the fundamental tradeoff between privacy and mutual information, we solve the following constrained maximization problem

$$\begin{array}{ll} \underset{Q}{\text{maximize}} & I(X;Y) \\ \text{subject to} & Q \in \mathcal{D}_\varepsilon \end{array}\,, \tag{6}$$

where $\mathcal{D}_\varepsilon$ is the set of all ε-locally differentially private mechanisms satisfying (1).

Motivated by such applications in statistical analysis, our goal is to provide a general framework for finding optimal privatization mechanisms that maximize information theoretic utilities under local differential privacy. We demonstrate the power of our techniques in a very general setting that includes both hypothesis testing and information preservation.

1.3 Our contributions

We study the fundamental tradeoff between local differential privacy and a rich class of convex utility functions. This class of utilities includes several information theoretic quantities such as mutual information and f-divergences. The privacy-utility tradeoff is posed as a constrained maximization problem: maximize utility subject to local differential privacy constraints. This maximization problem is (a) nonlinear: the utility functions we consider

are convex in Q; (b) non-standard: we are maximizing instead of minimizing a convex function; and (c) infinite dimensional: the space of all differentially private mechanisms is infinite dimensional. We show, in Theorem 2, that for all utility functions considered and any privacy level ε, a *finite* family of *extremal* mechanisms (a finite subset of the corner points of the space of differentially private mechanisms), which we call *staircase* mechanisms, contains the optimal privatization mechanism. We further prove, in Theorem 4, that solving the original privacy-utility problem is equivalent to solving a finite dimensional linear program, the outcome of which is the optimal mechanism. However, solving this linear program can be computationally expensive because it has $2^{|\mathcal{X}|}$ variables. To account for this, we show that two simple staircase mechanisms (the binary and randomized response mechanisms) are optimal in the high and low privacy regimes, respectively, and well approximate the intermediate regime. This contributes an important progress in the differential privacy area, where the privatization mechanisms have been few and almost no exact optimality results are known. As an application, we show that the effective sample size reduces from n to $\varepsilon^2 n$ under local differential privacy in the context of hypothesis testing.

We also study the fundamental tradeoff between utility and approximate differential privacy, a generalized notion of privacy that was first introduced in Dwork et al. (2006a). The techniques we develop for differential privacy do not generalize to approximate differential privacy. To account for this, we use an operational interpretation of approximate differential privacy (developed in Kairouz et al. (2014a)) to prove that a simple mechanism maximizes utility for all levels of privacy when the data is binary.

1.4 Related work

Our work is closely related to the recent work of Duchi et al. (2013) where an upper bound on $D_{\text{kl}}(M_0||M_1)$ was derived under the same local differential privacy setting. Precisely, Duchi et. al. proved that the KL-divergence maximization problem in (4) is at most $4(e^{\varepsilon}-1)^2\|P_1-P_2\|_{TV}^2$. This bound was further used to provide a minimax bound on statistical estimation using information theoretic converse techniques such as Fano's and Le Cam's inequalities. Such tradeoffs also provide tools for comparing various notions of privacy (Barber and Duchi, 2014).

In a similar spirit, we are also interested in maximizing information theoretic quantities under local differential privacy. We generalize the results of Duchi et al. (2013), and provide stronger results in the sense that we (a) consider a broader class of information theoretic utilities; (b) provide explicit constructions for the optimal mechanisms; and (c) recover the existing result of (Duchi et al., 2013, Theorem 1) (with a stronger condition on ε).

Our work also provides a formal connection to an information theoretic notion of privacy called information leakage (Chatzikokolakis et al., 2010; Sankar et al., 2013). Given a privatization mechanism Q, the information leakage is measured by the mutual information between the private data X and the released output Y, i.e. $I(X;Y)$. Information leakage has been widely studied as a practical notion of privacy. However, connections to differential privacy have been studied only indirectly through comparisons to how much distortion is incurred under the two notions of privacy (Sarwate and Sankar, 2014; Wang et al., 2014a). We show that under ε-local differential privacy, $I(X;Y)$ is upper bounded by

$0.5\varepsilon^2 \max_{A\subseteq\mathcal{X}} P(A)P(A^c)+O(\varepsilon^3)$ for small ε. Moreover, we provide an explicit privatization mechanism that achieves this bound.

While there is a vast literature on differential privacy, exact optimality results are only known for a few cases. The typical recipe is to propose a differentially private mechanism inspired by the work of Dwork (2006); Dwork et al. (2006b); McSherry and Talwar (2007) and Hardt and Rothblum (2010), and then establish its near-optimality by comparing the achievable utility to a converse, for example in linear dynamical systems (Wang et al., 2014b), principal component analysis (Chaudhuri et al., 2012; Blocki et al., 2012; Hardt and Roth, 2012; Kapralov and Talwar, 2013), linear queries (Hardt and Talwar, 2010; Hardt et al., 2012), logistic regression (Chaudhuri and Monteleoni, 2008) and histogram release (Lei, 2011). In this paper, we take a different route and solve the utility maximization problem *exactly*.

Optimal differentially private mechanisms are known only in a few cases. Ghosh et al. (2012) showed that the geometric noise adding mechanism is optimal (under a Bayesian setting) for monotone utility functions under count queries (sensitivity one). This was generalized by Geng et. al. (for a worst-case input setting) who proposed a family of mechanisms and proved its optimality for monotone utility functions under queries with arbitrary sensitivity (Geng and Viswanath, 2012, 2013a,b). The family of optimal mechanisms was called *staircase mechanisms* because for any y and any neighboring x and x', the ratio of $Q(y|x)$ to $Q(y|x')$ takes one of three possible values e^{ε}, $e^{-\varepsilon}$, or 1. Since the optimal mechanisms we develop also have an identical property, we retain the same nomenclature.

1.5 Organization

The remainder of the paper is organized as follows. In Section 2, we introduce the family of staircase mechanisms, prove its optimality for a broad class of convex utility functions, and study its combinatorial structure. In Section 3, we study the problem of private hypothesis testing and prove that two staircase mechanisms, the binary and randomized response mechanisms, are optimal for KL-divergence in the high and low privacy regimes, respectively, and (nearly) optimal the intermediate regime. We show, in Section 4, similar results for mutual information. In Section 5, we study approximate local differential privacy, a more general notion of local privacy. Finally, we conclude this paper in Section 6 with a few interesting and nontrivial extensions.

2. Main Results

In this section, we first present a formal definition for staircase mechanisms and prove that they are the optimal solutions to optimization problems of the form (8). We then provide a combinatorial representation for staircase mechanisms that allows us to reduce the infinite dimensional nonlinear program of (8) to a finite dimensional linear program with $2^{|\mathcal{X}|}$ variables. For any given privacy level ε and utility function $U(\cdot)$, one can solve this linear program to obtain the optimal privatization mechanism, albeit with significant computational challenges since the number of variables scales exponentially in the alphabet size. To address this issue, we prove, in Sections 3 and 4, that two simple staircase mechanisms, which we call the binary mechanism and the randomized response mechanism, are optimal

in the high and low privacy regimes, respectively, and well approximate the intermediate regime.

2.1 Optimality of staircase mechanisms

For an input alphabet $\mathcal{X}$ with $|\mathcal{X}| = k$, we represent the set of ε-locally differentially private mechanisms that lead to output alphabets $\mathcal{Y}$ with $|\mathcal{Y}| = \ell$ by

$$\mathcal{D}_{\varepsilon,\ell} = \mathcal{Q}_{k\times\ell} \cap \left\{ Q \ : \ \forall\, x, x' \in \mathcal{X}, S \subseteq \mathcal{Y}, \ \left| \ln \frac{Q\left(S|x\right)}{Q\left(S|x'\right)} \right| \leq \varepsilon \right\},$$

where $\mathcal{Q}_{k\times\ell}$ denotes the set of all $k \times \ell$ dimensional conditional distributions. The set of all ε-locally differentially private mechanisms is given by

$$\mathcal{D}_{\varepsilon} = \cup_{\ell\in\mathbb{N}} \mathcal{D}_{\varepsilon,\ell}. \tag{7}$$

The set of all conditional distributions acting on $\mathcal{X}$ is given by $\mathcal{Q} = \cup_{\ell\in\mathbb{N}} \mathcal{Q}_{k,\ell}$.

We consider two types of utility functions, one for the hypothesis testing setup and another for the information preservation setup. In the hypothesis testing setup, the utility is a function of the privatization mechanism and two priors defined on the input alphabet. Namely, $U\left(P_0, P_1, Q\right) : \mathbb{S}^k \times \mathbb{S}^k \times \mathcal{Q} \to \mathbb{R}_+$, where P_0 and P_1 are positive priors defined on $\mathcal{X}$, and $\mathbb{S}^k$ is the $(k-1)$-dimensional probability simplex. P_ν is said to be positive if $P_\nu\left(x\right) > 0$ for all $x \in \mathcal{X}$. In the information preservation setup, the utility is a function of the privatization mechanism and a prior defined on the input alphabet. Namely, $U\left(P, Q\right) : \mathbb{S}^k \times \mathcal{Q} \to \mathbb{R}_+$, where P is a positive prior defined on $\mathcal{X}$. For notational convenience, we will use $U\left(Q\right)$ to refer to both $U\left(P, Q\right)$ and $U\left(P_0, P_1, Q\right)$.

Definition 1 (Sublinear Functions) *A function $\mu\left(z\right) : \mathbb{R}^k \to \mathbb{R}$ is said to be sublinear if the following two conditions are met*

1. *$\mu\left(\gamma z\right) = \gamma\mu\left(z\right)$ for all $\gamma \in \mathbb{R}_+$.*
2. *$\mu\left(z_1 + z_2\right) \leq \mu\left(z_1\right) + \mu\left(z_2\right)$ for all $z_1, z_2 \in \mathbb{R}^k$.*

Let Q_y be the column of Q corresponding to $Q(y|\cdot)$ and μ be any sublinear function. We are interested in utilities that can be decomposed into a sum of sublinear functions. We study the *fundamental tradeoff between privacy and utility* by solving the following constrained maximization problem

$$\begin{aligned} \underset{Q}{\text{maximize}} \quad & U\left(Q\right) = \sum_{y\in\mathcal{Y}} \mu(Q_y) \\ \text{subject to} \quad & Q \in \mathcal{D}_{\varepsilon} \end{aligned} . \tag{8}$$

This includes maximization over information theoretic quantities of interest in statistical estimation and hypothesis testing such as mutual information, total variation, KL-divergence, and χ^2-divergence (Tsybakov and Zaiats, 2009). Since sub-linearity implies convexity, (8) is in general a complicated nonlinear program: we are maximizing (instead of minimizing) a convex function in Q; further, the dimension of Q might be unbounded: the optimal

privatization mechanism Q^* might produce an infinite output alphabet $\mathcal{Y}$. The following theorem proves that one never needs an output alphabet larger than the input alphabet in order to achieve the maximum utility, and provides a combinatorial representation for the optimal solution.

Theorem 2 *For any sublinear function μ and any $\varepsilon \geq 0$, there exists an optimal mechanism Q^* maximizing the utility in (8) over all ε-locally differentially private mechanisms, such that*

(a) the output alphabet size is at most the input alphabet size, i.e. $|\mathcal{Y}| \leq |\mathcal{X}|$; and

(b) for all $y \in \mathcal{Y}$, and $x, x' \in \mathcal{X}$

$$\left| \ln \frac{Q^*(y|x)}{Q^*(y|x')} \right| \in \{0, \varepsilon\} \ . \tag{9}$$

The first claim of bounded alphabet size is more generally true for any general utility $U\left(Q\right)$ that is convex in Q (not necessarily decomposing into a sum of sublinear functions as in (8)). The second claim establishes that there is an optimal mechanism with an extremal structure; the absolute value of the log-likelihood ratios can only take one of the two extremal values: 0 or e^ε (see Figure 2 for example). We refer to such a mechanism as a staircase mechanism, and define the *family of staircase mechanisms* formally as

$$\mathcal{S}_{\varepsilon} \ \equiv \ \{Q \,|\, \text{ satisfying (9)}\} \ .$$

For all choices of $U\left(Q\right) = \sum_{\mathcal{Y}} \mu(Q_y)$ and any $\varepsilon \geq 0$, Theorem 2 implies that the family of staircase mechanisms includes the optimal solutions to maximization problems of the form (8). Notice that staircase mechanisms are ε-locally differentially private, since any Q satisfying (9) implies that $Q(y|x)/Q(y|x') \leq e^\varepsilon$.

For global differential privacy, we can generalize the definition of staircase mechanisms to hold for all neighboring database queries x, x' (or equivalently within some sensitivity), and recover all known existing optimal mechanisms. Precisely, the geometric mechanism shown to be optimal in Ghosh et al. (2012), and the mechanisms shown to be optimal in Geng and Viswanath (2012, 2013a) (also called staircase mechanisms) are special cases of the staircase mechanisms defined above. We believe that the characterization of these extremal mechanisms and the analysis techniques developed in this paper can be of independent interest to researchers interested in optimal mechanisms for global privacy and more general utilities.

2.2 Combinatorial representation of the staircase mechanisms

Now that we know that staircase mechanisms are optimal, we can try to combinatorially search for the best staircase mechanism for an instance of the function μ and a fixed ε. To this end, we give a simple representation for all staircase mechanisms, exploiting the fact that they are scaled copies of a finite number of patterns.

Let $Q \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{Y}|}$ be a staircase mechanism, and $k = |\mathcal{X}|$ denote the size of the input alphabet. Then, from the definition of staircase mechanisms, $Q(y|x)/Q(y|x') \in \{e^{-\varepsilon}, 1, e^\varepsilon\}$ and each column $Q(y|\cdot)$ must be proportional to one of the canonical staircase patterns we define next.

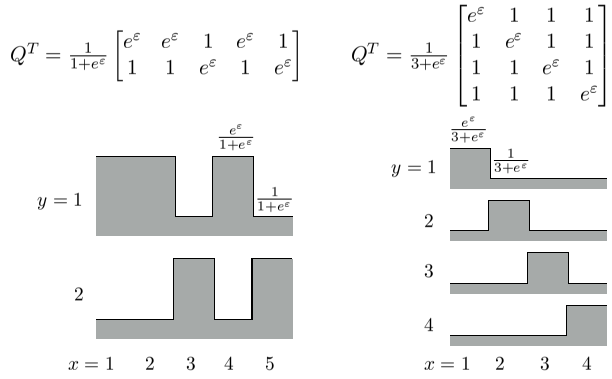


Figure 2: Examples of staircase mechanisms: the binary (left) and the randomized response (right) mechanisms.

Definition 3 (Staircase Pattern Matrix) *Let b_j be the k-dimensional binary vector corresponding to the binary representation of j for $j \leq 2^k - 1$. A matrix $S^{(k)} \in \{1, e^\varepsilon\}^{k \times 2^k}$ is called a staircase pattern matrix if the j-th column of $S^{(k)}$ is $S_j^{(k)} = (e^\varepsilon - 1)\, b_{j-1} + \mathbb{1}$, for $j \in \{1, \ldots, 2^k\}$. Each column of $S^{(k)}$ is a staircase pattern.*

When $k = 3$, there are $2^k = 8$ staircase patterns and the staircase pattern matrix is given by

$$S^{(3)} = \begin{bmatrix} 1 & 1 & 1 & 1 & e^\varepsilon & e^\varepsilon & e^\varepsilon & e^\varepsilon \\ 1 & 1 & e^\varepsilon & e^\varepsilon & 1 & 1 & e^\varepsilon & e^\varepsilon \\ 1 & e^\varepsilon & 1 & e^\varepsilon & 1 & e^\varepsilon & 1 & e^\varepsilon \end{bmatrix} .$$

For all values of k, there are exactly 2^k such patterns, and any column $Q(y|\cdot)$ of Q, a staircase mechanism, is a scaled version of one of the columns of $S^{(k)}$. Using this pattern matrix, we can show that any staircase mechanism Q can be represented as

$$Q \;=\; S^{(k)}\Theta \,, \tag{10}$$

where $\Theta = \mathrm{diag}(\theta)$ is a $2^k \times 2^k$ diagonal matrix and θ is a 2^k-dimensional vector representing the scaling of the columns of $S^{(k)}$. We can now formulate the problem of maximizing the utility as a linear program and prove their equivalence.

Theorem 4 *For any sublinear function μ and any $\varepsilon \geq 0$, the nonlinear program of (8) and the following linear program have the same optimal value*

$$\begin{aligned} \underset{\theta \in \mathbb{R}^{2^k}}{\text{maximize}} \quad & \sum_{j=1}^{2^k} \mu(S_j^{(k)})\theta_j = \mu^T\theta \\ \text{subject to} \quad & S^{(k)}\theta \;=\; \mathbb{1} \\ & \theta \geq 0 \,, \end{aligned} \tag{11}$$

and the optimal solutions are related by (10).

Thus, the infinite dimensional nonlinear program of (8) is now reduced to a finite dimensional linear program. The constraints in (11) ensure that we get a valid probability matrix $Q = S^{(k)}\Theta$ with rows that sum to one. One could potentially solve this LP with 2^k variables but its computational complexity scales exponentially in the alphabet size $k = |\mathcal{X}|$. For practical values of k this might not always be possible. However, in the following sections, we prove that in the high privacy regime ($\varepsilon \leq \varepsilon^*$ for some positive ε^*), there is a single optimal mechanism, which we call the *binary mechanism*, which dominates over all other mechanisms in a very strong sense for all utility functions of practical interest.

In order to understand the above theorem, observe that both the objective function and differential privacy constraints are invariant under *permutations* (or relabelling) of the columns of a privatization mechanism Q. In other words, shuffling the columns of an ε-locally differentially private mechanism Q results in a new ε-locally differentially private mechanism Q' that achieves the same utility. Similarly, both the objective function and differential privacy constraints are invariant under *merging/splitting* of outputs with the same pattern. To be specific, consider a privatization mechanism Q and suppose that there exist two outputs y and y' that have the same pattern, i.e. $Q(y|\cdot) = C\,Q(y'|\cdot)$ for some positive constant C. Then, we can consider a new mechanism Q' by merging the two columns corresponding to y and y'. Let y'' denote this new output. It follows that Q' satisfies the differential privacy constraints and the resulting utility is also preserved. Precisely, using the fact that $Q(y|\cdot) = C\,Q(y'|\cdot)$, it follows that

$$\mu(Q_y) + \mu(Q_{y'}) \;=\; \mu((1+C)Q_y) \;=\; \mu(Q'_{y''}) \,,$$

by the homogeneity property of μ. Therefore, we can naturally define equivalence classes for staircase mechanisms that are equivalent up to a permutation of columns and merging/splitting of columns with the same pattern:

$$[Q] = \{Q' \in \mathcal{S}_\varepsilon \,|\, \exists \text{ a sequence of permutations and merge/split of columns from } Q' \text{ to } Q\} \,.$$

To represent an equivalence class, we use a mechanism in $[Q]$ that is ordered and merged to match the patterns of the pattern matrix $S^{(k)}$. For any staircase mechanism Q, there exists a possibly different staircase mechanism $Q' \in [Q]$ such that $Q' = S^{(k)}\Theta$ for some diagonal matrix Θ with nonnegative entries. Therefore, to solve optimization problems of the form (8), we can restrict our attention to such representatives of equivalent classes. Further, for privatization mechanisms of the form $Q = S^{(k)}\Theta$, the objective function takes the form $\sum_j \mu(S_j^{(k)})\theta_j$, a simple linear function of Θ.

3. Hypothesis Testing

In this section, we study the fundamental tradeoff between local differential privacy and hypothesis testing. In this setting, there are n individuals each with data X_i sampled from a distribution P_ν for a fixed $\nu \in \{0,1\}$. Let Q be a non-interactive privatization mechanism guaranteeing ε-local differential privacy. The output of the privatization mechanism Y_i is distributed according to the induced marginal M_ν defined in (2). With a slight abuse of notation, we will use M_ν and P_ν to represent both probability distributions and probability mass functions. The power to discriminate data sampled from P_0 to data sampled from P_1 depends on the 'distance' between the marginals M_0 and M_1. To measure the ability of such statistical discrimination, our choice of utility of a privatization mechanism Q is an information theoretic quantity called Csiszár's f-divergence defined as

$$D_f(M_0||M_1) = \sum_{\mathcal{Y}} M_1(y) f\Big(\frac{M_0(y)}{M_1(y)}\Big) = U\left(P_0, P_1, Q\right) = U\left(Q\right) \ , \tag{12}$$

for some convex function f such that $f(1) = 0$. The Kullback-Leibler (KL) divergence $D_{\mathrm{kl}}(M_0||M_1)$ is a special case of f-divergence with $f(x) = x\log x$. The total variation distance $\|M_0 - M_1\|_{\mathrm{TV}}$ is also special case with $f(x) = (1/2)|x-1|$. Note that in general, the f-divergence is not necessarily a distance metric since it need not be symmetric or satisfy triangular inequality. We are interested in characterizing the optimal solution to

$$\begin{array}{ll} \underset{Q}{\text{maximize}} & D_f(M_0||M_1) \\ \text{subject to} & Q \in \mathcal{D}_\varepsilon \end{array} \ , \tag{13}$$

where $\mathcal{D}_\varepsilon$ is the set of all ε-locally differentially private mechanisms defined in (7).

A motivating example for this choice of utility is the Neyman-Pearson hypothesis testing framework (Cover and Thomas, 2012). Given the privatized views $\{Y_i\}_{i=1}^n$, the data analyst wants to test whether they are generated from M_0 or M_1. Let the null hypothesis be H_0 : Y_i's are generated from M_0, and the alternative hypothesis $H_1 : Y_i$'s are generated from M_1. For a choice of rejection region $R \subseteq \mathcal{Y}^n$, the probability of false alarm (type I error) is $\alpha = M_0^n(R)$ and the probability of miss detection (type II error) is $\beta = M_1^n(\mathcal{Y}^n \setminus R)$. Let $\beta^{\alpha^*} = \min_{R\subseteq\mathcal{Y}^n, \alpha<\alpha^*} \beta$ denote the minimum type II error achievable while keeping the type I error rate at most α^*. According to Chernoff-Stein lemma (Cover and Thomas, 2012), we know that

$$\lim_{n\to\infty} \frac{1}{n} \log \beta^{\alpha^*} = -D_{\mathrm{kl}}(M_0||M_1) \ .$$

Suppose the analyst knows P_0, P_1, and Q. Then in order to achieve optimal asymptotic error rate, one would want to maximize the KL divergence of the induced marginals, over all ε-locally differentially private mechanisms Q. The results we present in this section (Theorems 5 and 8 to be precise) provide an explicit construction of optimal mechanisms in high and low privacy regimes. Using these optimality results, we prove a fundamental limit on the achievable error rates under differential privacy. Precisely, with data collected from an ε-locally differentially privatization mechanism, one cannot achieve an asymptotic

type II error smaller than

$$\lim_{n\to\infty} \frac{1}{n} \log \beta^{\alpha^*} \geq -\frac{(1+\delta)(e^\varepsilon-1)^2}{(e^\varepsilon+1)} \|P_0 - P_1\|_{\mathrm{TV}}^2 \ \geq -\frac{(1+\delta)(e^\varepsilon-1)^2}{2(e^\varepsilon+1)} D_{\mathrm{kl}}(P_0||P_1) \ ,$$

whenever $\varepsilon \leq \varepsilon^*$, where ε^* is dictated by Theorem 5 and $\delta > 0$ is some arbitrarily small but positive constant. In the equation above, the second inequality follows from Pinsker's inequality. Since $(e^\varepsilon - 1)^2 = O(\varepsilon^2)$ for small ε, the effective sample size is now reduced from n to $\varepsilon^2 n$. This is the price of privacy. In the low privacy regime where $\varepsilon \geq \varepsilon^*$, for ε^* dictated by Theorem 8, one cannot achieve an asymptotic type II error smaller than

$$\lim_{n\to\infty} \frac{1}{n} \log \beta^{\alpha^*} \geq -D_{\mathrm{kl}}(P_0||P_1) + (1-\delta) G(P_0, P_1) e^{-\varepsilon} \ .$$

3.1 Optimality of staircase mechanisms

From the definition of $D_f(M_0||M_1)$, we have that

$$D_f(M_0||M_1) = \sum_{\mathcal{Y}} (P_1^T Q_y) f(P_0^T Q_y / P_1^T Q_y) = \sum_{\mathcal{Y}} \mu\left(Q_y\right) ,$$

where $P_\nu^T Q_y = \sum_{\mathcal{X}} P_\nu(x) Q(y|x)$ and $\mu\left(Q_y\right) = (P_1^T Q_y) f(P_0^T Q_y / P_1^T Q_y)$. For any $\gamma > 0$,

$$\begin{aligned} \mu\left(\gamma Q_y\right) &= \left(P_1^T(\gamma Q_y)\right) f\left(P_0^T\left(\gamma Q_y\right) / P_1^T\left(\gamma Q_y\right)\right) \\ &= \gamma \left(P_1^T Q_y\right) f\left(P_0^T Q_y / P_1^T Q_y\right) \\ &= \gamma \mu\left(Q_y\right) . \end{aligned}$$

Moreover, since the function $\phi(z,t) = tf\left(\frac{z}{t}\right)$ is convex in (z,t) for $0 \leq z,t \leq 1$, then μ is convex in Q_y. Convexity and homogeniety together imply sublinearlity. Therefore, Theorems 2 and 4 apply to $D_f(M_0||M_1)$ and we have that staircases are optimal.

3.2 Optimality of the binary mechanism

For a given P_0 and P_1, the *binary mechanism* is defined as a staircase mechanism with only two outputs $y \in \{0,1\}$ satisfying (see Figure 2)

$$Q(0|x) = \begin{cases} \dfrac{e^\varepsilon}{1+e^\varepsilon} & \text{if } P_0(x) \geq P_1(x) \ , \\ \dfrac{1}{1+e^\varepsilon} & \text{if } P_0(x) < P_1(x) \ . \end{cases} \quad Q(1|x) = \begin{cases} \dfrac{e^\varepsilon}{1+e^\varepsilon} & \text{if } P_0(x) < P_1(x) \ , \\ \dfrac{1}{1+e^\varepsilon} & \text{if } P_0(x) \geq P_1(x) \ . \end{cases} \tag{14}$$

Although this mechanism is extremely simple, perhaps surprisingly, we will establish that it is the optimal mechanism when a high level of privacy is required. Intuitively, the output should be very noisy in the high privacy regime, and we are better off sending just one bit of information that tells you whether your data is more likely to have come from P_0 or P_1.

Theorem 5 *For any pair of distributions P_0 and P_1, there exists a positive ε^* that depends on P_0 and P_1 such that for any f-divergences and any positive $\varepsilon \leq \varepsilon^*$, the binary mechanism maximizes the f-divergence between the induced marginals over all ε-locally differentially private mechanisms.*

This implies that in the high privacy regime, which is a typical setting studied in much of the differential privacy literature, the binary mechanism is universally optimal for all f-divergences. In particular this threshold ε^* is *universal*, in that it does not depend on the particular choice of which f-divergence we are maximizing. It is only a function of P_0 and P_1. This is established by proving a very strong statistical dominance using Blackwell's celebrated result on comparisons of statistical experiments Blackwell (1953). In a nutshell, we prove that any ε-locally differentially private mechanism can be simulated from the output of the binary mechanism for sufficiently small ε. Hence, the binary mechanism dominates over all other mechanisms and at the same time achieves the maximum divergence. A similar idea has been used previously in (Kairouz et al., 2013) to exactly characterize how much privacy degrades under composition attacks.

The optimality of binary mechanisms is not just for high privacy regimes. The next theorem shows that it is *the* optimal solution of (13) for all ε, when the objective function is the total variation distance: $D_f(M_0||M_1) = \|M_0 - M_1\|_{\text{TV}}$.

Theorem 6 *For any pair of distributions P_0 and P_1, and any $\varepsilon \geq 0$, the binary mechanism maximizes the total variation distance between the induced marginals M_0 and M_1 among all ε-locally differentially private mechanisms.*

When maximizing the KL divergence between the induced marginals, we show that the binary mechanism still achieves good performance for $\varepsilon \leq C$ where C is a constant that does not depend on P_0 and P_1. For the special case of KL divergence, let OPT denote the maximum value of (13) and BIN denote the KL divergence when the binary mechanism is used. The next theorem shows that

$$\text{BIN} \quad \geq \quad \frac{1}{2(e^\varepsilon + 1)^2}\text{OPT} \ .$$

Theorem 7 *For any ε and any pair of distributions P_0 and P_1, the binary mechanism is an $1/(2(e^\varepsilon+1)^2)$ approximation of the maximum KL divergence between the induced marginals M_0 and M_1 among all ε-locally differentially private mechanisms.*

Observe that $2(e^\varepsilon + 1)^2 \leq 32$ for $\varepsilon \leq 1$. Therefore, for any $\varepsilon \leq 1$, the simple binary mechanism is at most a constant factor away from the optimal mechanism.

3.3 Optimality of the randomized response mechanism

The *randomized response mechanism* (see Figure 2) is a staircase mechanism with $\mathcal{Y} = \mathcal{X}$ satisfying

$$Q(y|x) = \begin{cases} \dfrac{e^\varepsilon}{|\mathcal{X}| - 1 + e^\varepsilon} & \text{if } y = x \ , \\ \dfrac{1}{|\mathcal{X}| - 1 + e^\varepsilon} & \text{if } y \neq x \ . \end{cases} \tag{15}$$

In other words, the randomized response is a simple randomization over the same alphabet where the true data is released with probability $e^\varepsilon/\left(|\mathcal{X}| - 1 + e^\varepsilon\right)$. We view it as a multiple

choice generalization to the randomized response method proposed by Warner (1965). We now establish that for the special case of optimizing the KL divergence between the induced marginals, the randomized response mechanism is the optimal solution of (13) in the low privacy regime (i.e., $\varepsilon \geq \varepsilon^*$ for some threshold ε^* that depends on P_0 and P_1).

Theorem 8 *There exists a positive ε^* that depends on P_0 and P_1 such that for all P_0 and P_1, and all $\varepsilon \geq \varepsilon^*$, the randomized response mechanism maximizes the KL divergence between the induced marginals among all ε-locally differentially private mechanisms.*

The randomized response mechanism is particularly important because it does not depend on P_0 or P_1. Thus, even if the data providers and analysts do not have access to the priors, they can still use the randomized response mechanism to achieve the optimal (or near-optimal) utility in the moderate to low privacy regimes.

3.4 Numerical Experiments

A typical approach for achieving ε-local differential privacy is to add geometric noise with appropriately chosen variance. For an input with alphabet size $|\mathcal{X}| = k$, this amounts to relabelling the inputs as integers $\{1, \ldots, k\}$ and adding geometric noise, i.e., $Q(y|x) = ((1 - \varepsilon^{1/(k-1)})/(1 + \varepsilon^{1/(k-1)}))\varepsilon^{|y-x|/(k-1)}$ for $y \in \mathbb{Z}$. The output is then truncated at 1 and k to preserve the support.

For 100 instances of randomly chosen P_0 and P_1 defined over an input alphabet of size $|\mathcal{X}| = 6$, we compare the average performance of the binary, randomized response, and geometric mechanisms to the average performance of the optimal staircase mechanism for various values of ε. The optimal staircase mechanism is computed by solving the linear program in Equation (11) for each fixed pair (P_0, P_1) and ε. The left panel of Figure 3 shows the average performance measured by the normalized divergence $D_{\text{kl}}(M_0||M_1)/D_{\text{kl}}(P_0||P_1)$ for all 4 mechanisms. The average is taken over the 100 instances of P_0 and P_1. In the low privacy (large ε) regime, the randomized response achieves optimal performance (which converges exponentially in ε to 1) as predicted. In the high privacy regime (small ε), the binary mechanism achieves optimal performance (which converges quadratically in ε to 0) as predicted. In all regimes, both the binary and randomized response mechanisms provide significant gains over the geometric mechanism.

To illustrate how much worse the binary and the randomized response mechanisms can be relative to the optimal staircase mechanism, we plot in the right panel of Figure 3 the divergence under each mechanism normalized by the divergence under the optimal mechanism. This is done for all 100 instances of P_0 and P_1. In all instances, the binary mechanism is optimal for small ε and the randomized response mechanism is optimal for large ε. However, $D_{\text{kl}}(M_0||M_1)$ under the randomized response mechanism can be as bad as 10% of the optimal one (for small ε). Similarly, $D_{\text{kl}}(M_0||M_1))$ under the binary mechanism can be as bad as 25% of the optimal one (for large ε). To overcome this issue, we propose the following simple strategy: use the better among these two mechanisms. The performance of this strategy is illustrated in Figure 4. For various input alphabet size $|\mathcal{X}| \in \{3, 4, 6, 12\}$, we plot the performance of this mixed strategy for each value of ε and each of the 100 randomly generated instances of P_0 and P_1. This mixed strategy achieves at least 70% for $|\mathcal{X}| = 6$ (and 55% for $|\mathcal{X}| = 12$) of the optimal divergence for all instances. Figure 4 shows

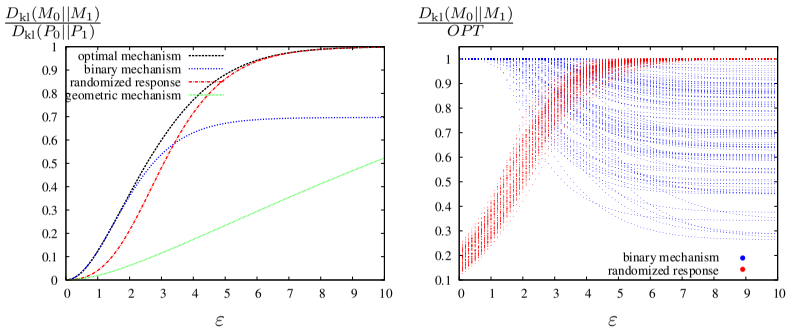


Figure 3: The binary and randomized response mechanisms are optimal in the high-privacy (small ε) and low-privacy (large ε) regimes, respectively, and improve over the geometric mechanism significantly (left). When the regimes are mismatched, $D_{\mathrm{kl}}(M_0||M_1)$ under these mechanisms can be as bad as 10% of the optimal one (right).

that this mixed strategy is not too sensitive to the size of the alphabet k. Therefore, this strategy provides a good mechanism that can be readily used in practice for any value of ε.

3.5 Lower bounds

In this section, we provide converse results on the fundamental limit of differentially private mechanisms; these results follow from our main theorems and are of independent interest in other applications where lower bounds in statistical analysis are studied (Beimel et al., 2008; Hardt and Talwar, 2010; Chaudhuri and Hsu, 2012; De, 2012). For example, a bound similar to (16) was used to provide converse results on the sample complexity for statistical estimation with differentially private data in Duchi et al. (2013).

Corollary 9 *For any $\varepsilon \geq 0$, let Q be any conditional distribution that guarantees ε-local differential privacy. Then, for any pair of distributions P_0, P_1 and any positive $\delta > 0$, there exists a positive ε^* that depends on P_0, P_1 and δ such that for any $\varepsilon \leq \varepsilon^*$ the induced marginals M_0 and M_1 satisfy the bound*

$$D_{\mathrm{kl}}\big(M_0||M_1\big) + D_{\mathrm{kl}}\big(M_1||M_0\big) \quad \leq \quad \frac{2(1+\delta)(e^\varepsilon-1)^2}{(e^\varepsilon+1)} \left\|P_0 - P_1\right\|_{\mathrm{TV}}^2 . \tag{16}$$

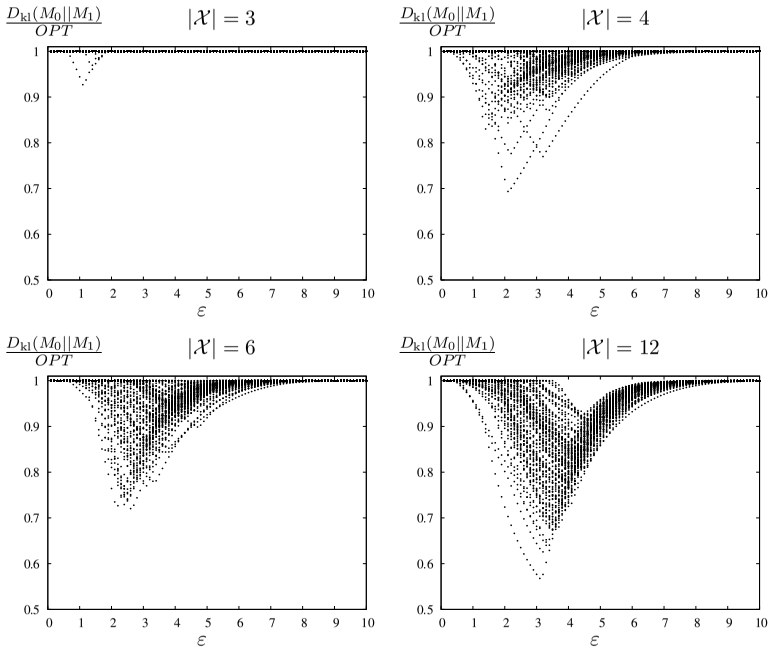


Figure 4: For varying input alphabet size $|\mathcal{X}| \in \{3, 4, 6, 12\}$, at least 55% of the optimal divergence can be achieved by taking the better one between the binary and the randomized response mechanisms.

This follows from Theorem 5 and observing that the binary mechanism achieves

$$\begin{aligned} & D_{\mathrm{kl}}\big(M_0||M_1\big) \\ &= \frac{(e^\varepsilon-1)P_0(T)+1}{e^\varepsilon+1} \log\Big(\frac{1+(e^\varepsilon-1)P_0(T)}{1+(e^\varepsilon-1)P_1(T)}\Big) \\ &\qquad + \frac{(e^\varepsilon-1)P_0(T^c)+1}{e^\varepsilon+1} \log\Big(\frac{1+(e^\varepsilon-1)P_0(T^c)}{1+(e^\varepsilon-1)P_1(T^c)}\Big) \\ &= \frac{(e^\varepsilon-1)^2}{e^\varepsilon+1}(P_0(T)-P_1(T)) + O(\varepsilon^3) \\ &= \frac{(e^\varepsilon-1)^2}{e^\varepsilon+1} \left\|P_0 - P_1\right\|_{\mathrm{TV}}^2 + O(\varepsilon^3) , \end{aligned}$$

where $T \subseteq \mathcal{X}$ is the set of x such that $P_0(x) \geq P_1(x)$. Compared to (Duchi et al., 2013, Theorem 1), we recover their bound of $4(e^\varepsilon-1)^2\|P_0 - P_1\|_{\mathrm{TV}}^2$ with a smaller constant.

We want to note that Duchi et al.'s bound holds for all values of ε and uses a different technique of bounding the KL divergence directly, however no achievable mechanism has been provided. We instead provide an explicit mechanism, that is optimal in high privacy regime.

Similarly, in the low privacy regime, we can show the following converse result.

Corollary 10 *For any $\varepsilon \geq 0$, let Q be any conditional distribution that guarantees ε-local differential privacy. Then, for any pair of distributions P_0 and P_1 and any positive $\delta > 0$, there exists a positive ε^* that depends on P_0 and P_1 and δ such that for any $\varepsilon \geq \varepsilon^*$ the induced marginals M_0 and M_1 satisfy the bound*

$$D_{\mathrm{kl}}\big(M_0||M_1\big) + D_{\mathrm{kl}}\big(M_1||M_0\big) \quad \leq \quad D_{\mathrm{kl}}(P_0||P_1) - (1-\delta)G(P_0,P_1)e^{-\varepsilon}\,. \tag{17}$$

where $G(P_0,P_1) = \sum_{\mathcal{X}}(1-P_0(x))\log(P_1(x)/P_0(x))$.

This follows directly from Theorem 8 and observing that the randomized response mechanism achieves $D_{\mathrm{kl}}(M_0||M_1) = D_{\mathrm{kl}}(P_0||P_1) - G(P_0,P_1)e^{-\varepsilon} + O(e^{-2\varepsilon})$.

Figure 5 illustrates the gap between the divergence achieved by the geometric mechanism described in the previous section and the optimal mechanisms (the binary mechanism for the high privacy regime and the randomized response mechanism for the low privacy regime). For each instance of the 100 randomly generated P_0 and P_1 defined over input alphabets of size $k = 6$, we plot the resulting divergence $D_{\mathrm{kl}}(M_0||M_1)$ as a function of $\|P_0 - P_1\|_{\mathrm{TV}}$ for $\varepsilon = 0.1$, and as a function of $D_{\mathrm{kl}}(P_0||P_1)$ for $\varepsilon = 10$. The binary and the randomized response mechanisms exhibit the scaling predicted by Equation (16) and (17), respectively. Similarly, for total variation, we can get the following converse result.

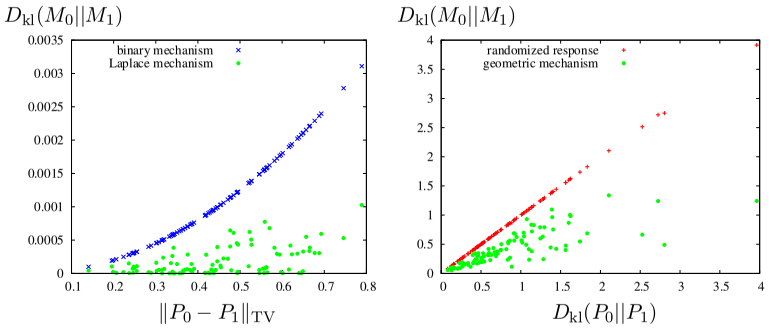


Figure 5: For small $\varepsilon = 0.1$ (left) the binary mechanism achieves the optimal KL divergence, which scales as Equation (16). For large $\varepsilon = 10$ (right) the randomized response achieves the optimal KL divergence, which scales as Equation (17). Both mechanisms improve significantly over the geometric mechanism.

Corollary 11 *For any $\varepsilon \geq 0$, let Q be any conditional distribution that guarantees ε-local differential privacy. Then, for any pair of distributions P_0 and P_1, the induced marginals M_0 and M_1 satisfy the bound $\big\|M_0 - M_1\big\|_{\mathrm{TV}} \leq ((e^{\varepsilon}-1)/(e^{\varepsilon}+1))\,\big\|P_0 - P_1\big\|_{\mathrm{TV}}$, and equality is achieved by the binary mechanism.*

This follows from Theorem 6 and explicitly computing the total variation achieved by the binary mechanism.

4. Information Preservation

In this section, we study the fundamental tradeoff between local privacy and mutual information. Consider a random variable X distributed according to P. The information content in X is captured by entropy

$$H\left(X\right) = -\sum_{\mathcal{X}} P\left(x\right)\log P\left(x\right).$$

We are interested in releasing a differentially private version of X represented by Y. The random variable Y should preserve the information content of X as much as possible while meeting the local differential privacy constraints. Similar to the hypothesis testing setting, we will show that a variant of the binary mechanism is optimal in the high privacy regime and the randomized response mechanism is optimal in the low privacy regime.

Let Q be a non-interactive privatization mechanism guaranteeing ε-local differential privacy. The output of the privatization mechanism Y is distributed according to the induced marginal M given by

$$M(S) \quad = \quad \sum_{x\in\mathcal{X}} Q(S|x)P(x)\,,$$

for $S \subseteq \mathcal{Y}$. With a slight abuse of notation, we will use M and P to represent both probability distributions and probability mass functions. The information content in Y about X is captured by the well celebrated information theoretic quantity called mutual information. The mutual information between X and Y is given by

$$I\left(X;Y\right) = \sum_{\mathcal{X}}\sum_{\mathcal{Y}} P\left(x\right)Q\left(y|x\right)\log\left(\frac{Q\left(y|x\right)}{\sum_{l\in\mathcal{X}}P\left(l\right)Q\left(y|l\right)}\right) = U\left(P,Q\right) = U\left(Q\right).$$

Notice that $I\left(X;Y\right) \leq H\left(X\right)$ and $I\left(X;Y\right)$ is convex in Q (Cover and Thomas, 2012). To preserve the information context in X, we wish to choose a privatization mechanism Q such that the mutual information between X and Y is maximized subject to differential privacy constraints. In other words, we are interested in characterizing the optimal solution to

$$\begin{array}{ll}\underset{Q}{\text{maximize}} & I\left(X;Y\right)\\ \text{subject to} & Q\in\mathcal{D}_{\varepsilon}\end{array}\,, \tag{18}$$

where $\mathcal{D}_{\varepsilon}$ is the set of all ε-locally differentially private mechanisms defined in (7). The above mutual information maximization problem can be thought of as a conditional entropy minimization problem since $I\left(X;Y\right) = H\left(X\right) - H\left(X|Y\right)$.

4.1 Optimal staircase mechanisms

From the definition of $I(X;Y)$, we have that

$$I(X;Y) = \sum_{\mathcal{Y}} \sum_{\mathcal{X}} P(x) Q(y|x) \log \left(\frac{Q(y|x)}{P^T Q_y} \right) = \sum_{\mathcal{Y}} \mu(Q_y),$$

where $P^T Q_y = \sum_{\mathcal{X}} P(x) Q(y|x)$ and $\mu(Q_y) = \sum_{\mathcal{X}} P(x) Q(y|x) \log \left(Q(y|x) / P^T Q_y \right)$. Notice that $\mu(\gamma Q_y) = \gamma \mu(Q_y)$, and by the log-sum inequality, μ is convex. Convexity and homogeneity together imply sublinearity. Therefore, Theorems 2 and 4 apply to $I(X;Y)$ and we have that staircase mechanisms are optimal.

For a given P, the *binary mechanism for mutual information* is defined as a staircase mechanism with only two outputs $y \in \{0,1\}$ (see Figure 2). Let $T \subseteq \mathcal{X}$ be the set that partitions $\mathcal{X}$ into two partitions, T and T^c, such that $|P(T) - P(T^c)|$ is minimized. Precisely,

$$T \in \arg\min_{A \subseteq \mathcal{X}} \left| P(A) - \frac{1}{2} \right|. \tag{19}$$

Observe that there are always multiple choices for T. Indeed, for any minimizing set T, T^c is also a minimizing set since $|P(T) - 1/2| = |P(T^c) - 1/2|$. When there is only one such pair, the binary mechanism is uniquely defined as

$$Q(0|x) = \begin{cases} \dfrac{e^\varepsilon}{1+e^\varepsilon} & \text{if } x \in T, \\ \dfrac{1}{1+e^\varepsilon} & \text{if } x \notin T. \end{cases} \qquad Q(1|x) = \begin{cases} \dfrac{e^\varepsilon}{1+e^\varepsilon} & \text{if } x \notin T, \\ \dfrac{1}{1+e^\varepsilon} & \text{if } x \in T. \end{cases} \tag{20}$$

When there are multiple pairs, any pair (T, T^c) can be chosen to define the binary mechanism. All resulting binary mechanisms are equivalent from a utility maximization perspective.

In what follows, we will establish that this simple mechanism is the optimal mechanism in the high privacy regime. Intuitively, in the high privacy regime, we cannot release more than one bit of information, and hence, the input alphabet is reduced to a binary output alphabet. In this case we have to maximize the information contained in the released bit by maximizing its entropy: $T \in \arg\max_{A \subseteq \mathcal{X}} \big(-P(A)\log P(A) - P(A^c)\log P(A^c) \big) = \arg\max_{A \subseteq \mathcal{X}} |P(A) - 1/2|$ (see Section 9.1 for a proof).

Theorem 12 *For any distribution P, there exists a positive ε^* that depends on P such that for any positive $\varepsilon \leq \varepsilon^*$, the binary mechanism maximizes the mutual information between the input and the output of a privatization mechanism over all ε-locally differentially private mechanisms.*

This implies that in the high privacy regime, the binary mechanism is the optimal solution for (18).

Next, we show that the binary mechanism achieves near-optimal performance for all $(\mathcal{X}, P)$ and $\varepsilon \leq 1$ even when $\varepsilon^* < 1$. Let OPT denote the maximum value of (18) and BIN denote the mutual information achieved by the binary mechanism given in (20). The next theorem shows that

$$\text{BIN} \geq \frac{1}{1+e^\varepsilon} \text{OPT}.$$

Theorem 13 *For any $\varepsilon \leq 1$ and any distribution P, the binary mechanism is an $(1+e^\varepsilon)$-approximation of the maximum mutual information between the input and the output of a privatization mechanism among all ε-locally differentially private mechanisms.*

Note that $1 + e^\varepsilon \leq 4$ for $\varepsilon \leq 1$ which is a commonly studied regime in differential privacy applications. Therefore, we can always use the simple binary mechanism and the resulting mutual information is at most a constant factor away from the optimal.

In the low privacy regime ($\varepsilon \geq \varepsilon^*$), the *randomized response mechanism* defined in(15) is optimal.

Theorem 14 *There exists a positive ε^* that depends on P such that for any distribution P and all $\varepsilon \geq \varepsilon^*$, the randomized response mechanism maximizes the mutual information between the input and the output of as privatization mechanism over all ε-locally differentially private mechanisms.*

4.2 Numerical Experiments

For 100 instances of randomly chosen P defined over input alphabet of size $|\mathcal{X}| = 6$, we compare the average performance of the binary, randomized response, and the geometric mechanisms to the optimal mechanism. We plot (in Figure 6, left) the average performance measured by the normalized mutual information $I(X;Y)/H(X)$ for all 4 mechanisms. The average is taken over the 100 instances of P. In the low privacy (large ε) regime, the randomized response achieves optimal performance as predicted, which converges to one. In the high privacy regime (small ε), the binary mechanism achieves optimal performance as predicted. In all regimes, both mechanisms significantly improve over the geometric mechanism. To illustrate how much worse the binary and randomized response mechanisms can be (relative to the optimal staircase mechanism), we plot (in Figure 6, right) the mutual information under each mechanism normalized by the mutual information under the optimal staircase mechanism. This is done for all 100 instances of P. In all instances, the binary mechanism is optimal for small ε and the randomized response mechanism is optimal for large ε. However, $I(X;Y)$ under the randomized response mechanism can be as bad as 35% of the optimal one (for small ε). Similarly, $I(X;Y)$ under the binary mechanism can be as bad as 40% of the optimal one (for large ε).

For $|\mathcal{X}| \in \{3, 4, 6, 12\}$, we plot (in Figure 7) the performance of better between the binary and randomized response mechanisms normalized by the optimal mechanism for all 100 randomly generated instances of P. This mixed strategy achieves at least 75% for $|\mathcal{X}| = 6$ (and 65% for $|\mathcal{X}| = 12$) of the optimal mutual infirmation for all instances of P. Moreover, it is not sensitive to the size of the alphabet $|\mathcal{X}|$.

4.3 Lower bounds

In this section, we provide converse results on the fundamental limit of locally differentially private mechanisms when utility is measured via mutual information.

Corollary 15 *For any $\varepsilon \geq 0$, let Q be any conditional distribution that guarantees ε-local differential privacy. Then, for any distribution P and any positive $\delta > 0$, there exists a*

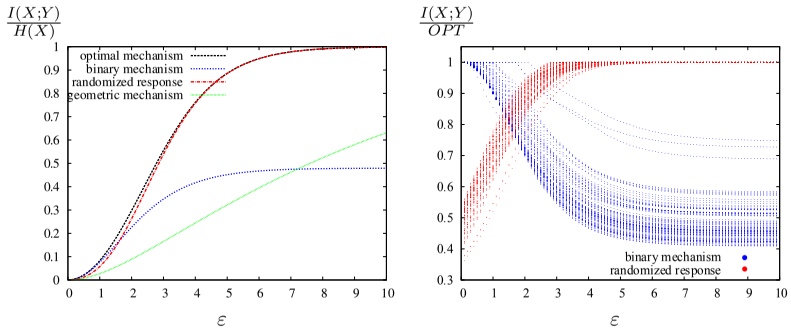


Figure 6: The binary and randomized response mechanisms are optimal in the high-privacy (small ε) and low-privacy (large ε) regimes, respectively, and improve over the geometric mechanism significantly (left). When the regimes are mismatched, $I(X;Y)$ under these mechanisms can each be as bad as 35% of the optimal one (right).

positive ε^* that depends on P and δ such that for any $\varepsilon \leq \varepsilon^*$ the following bound holds

$$I(X;Y) \leq (1+\delta)\frac{1}{2}P(T)P(T^c)\varepsilon^2, \tag{21}$$

where T is defined in (19).

This follows from Theorem 12 (optimality of the binary mechanism) and observing that the binary mechanism achieves

$$\begin{aligned} I(X;Y) &= \frac{1}{e^\varepsilon+1}\left\{P(T)e^\varepsilon \log\frac{e^\varepsilon}{P(T^c)+e^\varepsilon P(T)} + P(T^c)\log\frac{1}{P(T^c)+e^\varepsilon P(T)}\right\} \\ &\quad + \frac{1}{e^\varepsilon+1}\left\{P(T^c)e^\varepsilon \log\frac{e^\varepsilon}{P(T)+e^\varepsilon P(T^c)} + P(T)\log\frac{1}{P(T)+e^\varepsilon P(T^c)}\right\} \\ &= \frac{1}{2}P(T)P(T^c)\varepsilon^2 + O\left(\varepsilon^3\right). \end{aligned}$$

Similarly, in the low privacy regime, we can show the following converse result.

Corollary 16 *For any $\varepsilon \geq 0$, let Q be any conditional distribution that guarantees ε-local differential privacy. Then, for any distributions P and any positive $\delta > 0$, there exists a positive ε^* that depends on P and δ such that for any $\varepsilon \geq \varepsilon^*$ the following bound holds*

$$I(X;Y) \leq H(X) - (1-\delta)(k-1)\varepsilon e^{-\varepsilon}.$$

This follows directly from Theorem 14 (optimality of the randomized response mechanism) and observing that the randomized response mechanism achieves

$$I(X;Y) = H(X) - (k-1)\varepsilon e^{-\varepsilon} + O(e^{-2\varepsilon}). \tag{22}$$

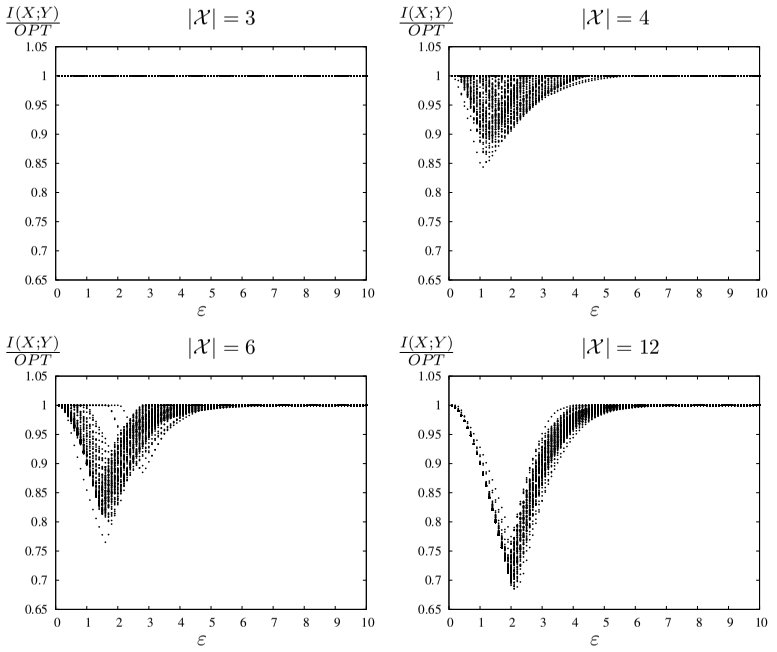


Figure 7: For varying input alphabet size $|\mathcal{X}| \in \{3,4,6,12\}$, at least 65% of the maximum $I(X;Y)$ can be achieved by taking the better one between the binary and the randomized response mechanisms.

Figure 8 illustrates the gap between the mutual information achieved by the geometric mechanism and the optimal mechanisms (the binary mechanism for the high privacy regime and the randomized response mechanism for the low privacy regime). For each instance of the 100 randomly generated P over input of size $k = 6$, we plot the resulting mutual information $I(X;Y)$ as a function of $P(T)P(T^c)$ for $\varepsilon = 0.1$, and as a function of $H(X)$ for $\varepsilon = 10$. The binary and the randomized response mechanisms exhibit the scaling predicted by Equations (21) and (22), respectively.

5. Generalizations to approximate differential privacy

In this section, we generalize the results of the previous sections in the following ways.

1. We consider the class of utility functions that obey the data processing inequality. Consider the composition of two privatization mechanisms $QW = Q \circ W$ where the

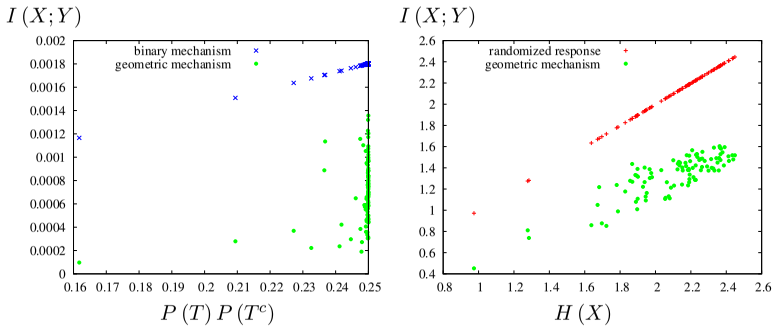


Figure 8: For $\varepsilon = 0.1$ (left) the binary mechanism achieves the maximum $I(X;Y)$, which scales as Equation (21). For $\varepsilon = 10$ (right) the randomized response mechanism achieves the optimal mutual information, which scales as Equation (22).

output of the first mechanism Q is applied to another mechanism W. We say that a utility function $U(\cdot)$ obeys the data processing inequality if the following inequality holds for all Q and W

$$U(QW) \quad \leq \quad U(Q) \, .$$

The following proposition, proved in Section 10, shows that the class of utilities obeying the data processing inequality includes all the utility functions we studied in Section 2.

Proposition 17 *Any utility function that can be written in the form of* $U(Q) = \sum_{\mathcal{Y}} \mu(Q_y)$*, where* μ *is any sublinear function, obeys the data processing inequality.*

2. We consider (ε, δ)-differential privacy which generalizes the notion of ε-differential privacy. (ε, δ)-differential privacy is commonly referred to as approximate differential privacy and it was first introduced in Dwork et al. (2006a). For the release of a random variable $X \in \mathcal{X}$, we say that a mechanism Q is (ε, δ)-locally differentially private if

$$Q(S|x) \leq e^{\varepsilon} Q\left(S|x'\right) + \delta, \tag{23}$$

for all $S \subseteq \mathcal{Y}$ and all $x, x' \in \mathcal{X}$. Note that ε-local differential privacy is a special case of (ε, δ)-local differential privacy where $\delta = 0$.

3. We prove that the *quaternary mechanism*, defined in Equation (24), is optimal for any ε and any δ. This is different from the treatment conducted in the previous sections where we proved the optimality of the binary (randomized response) mechanism for sufficiently small (large) ε and $\delta = 0$.

The treatment in this section, even though more general than the one in previous sections in the ways described above, holds only for binary alphabets (i.e., $|\mathcal{X}| = 2$). Finding optimal privatization mechanisms under (ε, δ)-local differential privacy for larger input alphabets (i.e., $|\mathcal{X}| > 2$) is an interesting open question. Unlike ε-local differential privacy, the privacy constraints under (ε, δ)-local differential privacy no longer decompose into separate constraints on each output y. This makes it difficult to generalize the techniques developed in previous sections of this paper. However, for the special case of binary input alphabets, we can prove the optimality of one mechanism for all values of (ε, δ) and all utility functions that obey the data processing inequality.

For a binary random variable $X \in \mathcal{X} = \{0, 1\}$, the *quaternary mechanism* maps X to a quaternary random variable $Y \in \mathcal{Y} = \{0, 1, 2, 3\}$ and is defined as

$$Q_{\mathrm{QT}}(0|x) = \begin{cases} \delta & \text{if } x = 0 \, , \\ 0 & \text{if } x = 1 \, . \end{cases} \quad Q_{\mathrm{QT}}(1|x) = \begin{cases} 0 & \text{if } x = 0 \, , \\ \delta & \text{if } x = 1 \, . \end{cases} \tag{24}$$

$$Q_{\mathrm{QT}}(2|x) = \begin{cases} (1-\delta)\frac{1}{1+e^{\varepsilon}} & \text{if } x = 0 \, , \\ (1-\delta)\frac{e^{\varepsilon}}{1+e^{\varepsilon}} & \text{if } x = 1 \, . \end{cases} \quad Q_{\mathrm{QT}}(3|x) = \begin{cases} (1-\delta)\frac{e^{\varepsilon}}{1+e^{\varepsilon}} & \text{if } x = 0 \, , \\ (1-\delta)\frac{1}{1+e^{\varepsilon}} & \text{if } x = 1 \, . \end{cases}$$

In other words, the quaternary mechanism passes X unchanged with probability δ and applies the binary mechanism (defined in previous sections) with probability $1 - \delta$. The main result of this section can be stated formally as follows.

Theorem 18 *If* $|\mathcal{X}| = 2$*, then for any* ε*, any* δ*, and any* $U(Q)$ *that obeys the data processing inequality, the quaternary mechanism maximizes* $U(Q)$ *subject to* $Q \in \mathcal{D}_{(\varepsilon,\delta)}$*, the set of all* (ε, δ)*-locally differentially private mechanism.*

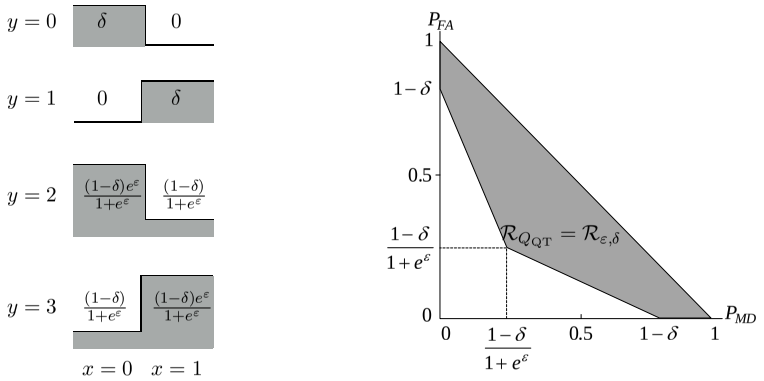
The proof of Theorem 18 depends on an *operational definition* of differential privacy which we describe next. Consider a privatization mechanism Q that maps $X \in \{0, 1\}$ stochastically to $Y \in \mathcal{Y}$. Given Y, construct a binary hypothesis test on whether $X = 0$ or $X = 1$. Any binary hypothesis test is completely described by a, possibly randomized, decision rule $\hat{X} : Y \to \{0, 1\}$. The two types of error associated with $\hat{X}$ are *false alarm:* $\hat{X} = 1$ when $X = 0$, and *miss detection:* $\hat{X} = 0$ when $X = 1$. The probability of false alarm is given by $P_{\mathrm{FA}} = \mathbb{P}(\hat{X} = 1|X = 0)$ while the probability of miss detection is given by $P_{\mathrm{MD}} = \mathbb{P}(\hat{X} = 0|X = 1)$. For a fixed Q, the convex hull of all pairs $(P_{\mathrm{MD}}, P_{\mathrm{FA}})$ for all decision rules $\hat{X}$ defines a two-dimensional *error region* where P_{MD} is plotted against P_{FA}. For example, the quaternary mechanism given in Figure 9a has an error region $\mathcal{R}_{Q_{\mathrm{QT}}}$ shown in Figure 9b.

It turns out that (ε, δ)-local differential privacy imposes the following conditions on the error region of all (ε, δ)-locally differentially private mechanisms

$$P_{\mathrm{FA}} + e^{\varepsilon} P_{\mathrm{MD}} \quad \geq \quad 1 - \delta \, , \quad \text{and} \quad e^{\varepsilon} P_{\mathrm{FA}} + P_{\mathrm{MD}} \quad \geq \quad 1 - \delta \, ,$$

for any decision rule $\hat{X}$. These two conditions define an error region $\mathcal{R}_{\varepsilon,\delta}$ shown in Figure 9b. Interestingly, the next theorem shows that the converse result is also true.

Theorem 19 *A mechanism* Q *is* (ε, δ)*-locally differentially private if and only if* $\mathcal{R}_Q \subseteq \mathcal{R}_{\varepsilon,\delta}$*.*



(a) Privatization mechanism (b) Error region

Figure 9: The quaternary mechanism

The proof of the above theorem can be found in Kairouz et al. (2013). Notice that it is no coincidence that $\mathcal{R}_{Q_{\mathrm{QT}}} = \mathcal{R}_{\varepsilon,\delta}$. This property will be essential in proving the optimality of the quaternary mechanism.

Theorem 19 allows us to benefit from the data processing inequality (DPI) and its converse, which follows from a celebrated result by Blackwell (1953). These inequalities, while simple by themselves, lead to surprisingly strong technical results. Indeed, there is a long line of such a tradition in the information theory literature (see Chapter 17 of Cover and Thomas (2012)). Consider two privatization mechanisms, $Q^{(1)}$ and $Q^{(2)}$. Let Y and Z denote the output of the mechanisms $Q^{(1)}$ and $Q^{(2)}$, respectively. We say that $Q^{(1)}$ dominates $Q^{(2)}$ if there exists a coupling of Y and Z such that X–Y–Z forms a Markov chain. In other words, we say $Q^{(1)}$ dominates $Q^{(2)}$ if there exists a stochastic mapping Q such that $Q^{(2)} = Q^{(1)} \circ Q$.

Theorem 20 *A mechanism $Q^{(1)}$ dominates a mechanism $Q^{(2)}$ if and only if $\mathcal{R}_{Q^{(2)}} \subseteq \mathcal{R}_{Q^{(1)}}$.*

The proof of the above theorem can be found in Blackwell (1953). Observe that by Theorems 20 and 19, and the fact that $\mathcal{R}_{Q_{\mathrm{QT}}} = \mathcal{R}_{\varepsilon,\delta}$, the quaternary mechanism dominates any other differentially private mechanism. In other words, for any differentially private mechanism Q, there exists a stochastic mapping W such that $Q = W \circ Q_{\mathrm{QT}}$. Therefore, for any (ε, δ) and any utility function $U(.)$ obeying the data processing inequality, we have that $U(Q) \leq U(Q_{\mathrm{QT}})$. This finishes the proof of Theorem 18.

6. Discussion

In this paper, we have considered a broad class of convex utility functions and assumed a setting where individuals cannot collaborate (communicate with each other) before releasing

their data. It turns out that the techniques developed in this work can be generalized to find optimal privatization mechanisms in a setting where different individuals can collaborate interactively and each individual can be an analyst (Kairouz et al., 2014b).

Binary hypothesis testing and information preservation are two canonical problems with a wide range of applications. However, there are a number of non-trivial and interesting extensions to our work.

Correlation among data. In some scenarios the X_i's could be correlated (e.g., when different individuals observe different functions of the same random variable). In this case, the data analyst is interested in inferring whether the data was generated from P_0^n or P_1^n, where P_ν^n is one of two possible joint priors on $X_1, ..., X_n$. This is a challenging problem because knowing X_i reveals information about X_j, $j \neq i$. Therefore, the utility maximization problems for different individuals are coupled in this setting.

Robust and m-ary hypothesis testing. In some cases the data analyst need not have access to P_0 and P_1, but rather two classes of prior distribution P_{θ_0} and P_{θ_1} for $\theta_0 \in \Lambda_0$ and $\theta_1 \in \Lambda_1$. Such problems are studied under the rubric of universal hypothesis testing and robust hypothesis testing. One possible direction is to select the privatization mechanism that maximizes the worst case utility: $Q^* = \arg\max_{Q \in \mathcal{D}_\varepsilon} \min_{\theta_0 \in \Lambda_0, \theta_1 \in \Lambda_1} D_f(M_{\theta_0} || M_{\theta_1})$, where M_{θ_ν} is the induced marginal under P_{θ_ν}.

The more general problem of private m-ary hypothesis testing is also an interesting but challenging one. In this setting, the X_i's can follow one of m distributions P_0, P_1, ..., P_{m-1}. Consequently, the Y_i's can follow one of m distributions M_0, M_1, ..., M_{m-1}. The utility can be defined as the average f-divergence between any two distributions: $1/(m(m-1)) \sum_{i \neq j} D_f(M_i || M_j)$, or the worst case one: $\min_{i \neq j} D_f(M_i || M_j)$.

Non-exchangeable utility functions. The utility studied in this paper was measured by functions that are exchangeable, i.e. the utility did not depend on the naming (labelling) of the private and privatized data (X and Y). This made sense for statistical learning applications that depend on information theoretic quantities such as f-divergences and mutual information. However, in some other applications, the utility might be defined over $\mathcal{X} \cup \mathcal{Y}$ in a metric space, where there exists a natural measure of distance (or distortion) between the data points. In this case, we can formulate the problem as a distortion minimization one

$$\text{minimize}_{Q \in \mathcal{D}_\varepsilon} \quad \sum_{x,y} d(x,y) P(x) Q(y|x) \ ,$$

where $d(x, y)$ is some distortion metric. Wang et al. (2014a) studied this problem, and showed that the mechanism $Q(y|x) \propto e^{\varepsilon(1-d(x,y))}/(k-1+e^\varepsilon)$ achieves near optimal performance when ε is large enough, which is the low privacy regime. Notice that when Hamming distance is used, $d(x, y) = \mathbb{I}(x \neq y)$, this recovers the randomized response mechanism exactly. This provides a starting point for generalizing the search for optimal mechanisms under non-exchangeable utility functions.

7. Proof of Theorems 2 and 4

We start the proof with a few definitions, a lemma, and a general result that applies to any convex utility function that obeys a mild assumption.

Recall that for an input alphabet $\mathcal{X}$ with $|\mathcal{X}| = k$, we represent the set of ε-locally differentially private mechanisms that lead to output alphabets $\mathcal{Y}$ with $|\mathcal{Y}| = \ell$ by $\mathcal{D}_{\varepsilon,\ell}$. The set of all ε-locally differentially private mechanisms is given by $\mathcal{D}_\varepsilon = \cup_{\ell\in\mathbb{N}}\mathcal{D}_{\varepsilon,\ell}$. A utility function $U(Q)$ is convex in Q if $U\left(\lambda Q^{(1)} + (1-\lambda)Q^{(2)}\right) \leq \lambda U\left(Q^{(1)}\right) + (1-\lambda)U\left(Q^{(2)}\right)$ for any $\lambda \in (0,1)$. Convex utility functions are ubiquitous in information theory and statistics.

Assumption 1 *If a $k\times\ell$ privatization mechanism $Q^{(1)} \in \mathcal{D}_{\varepsilon,\ell}$ is obtained by deleting an all-zero column of a $k\times\ell+1$ privatization mechanism $Q^{(2)} \in \mathcal{D}_{\varepsilon,\ell+1}$, then $U\left(Q^{(1)}\right) = U\left(Q^{(2)}\right)$.*

Naturally, one would expect that if we delete the zero columns of a privatization mechanism $Q^{(2)}$ to obtain a new privatization mechanism $Q^{(1)}$, we would still get the same utility. This is because a "reasonable" utility function should not depend on output alphabets with zero probability.

Theorem 21 *If $U(Q)$ is a convex utility function that satisfies Assumption 1, then the following holds*

$$\max_{Q\in\mathcal{D}_\varepsilon} U(Q) = \max_{Q\in\cup_{\ell=1}^k \mathcal{D}_{\varepsilon,\ell}} U(Q).$$

This theorem implies that among all ε-locally differentially private mechanisms, we only need to consider those that lead to output alphabets of size $\ell \leq k$. In other words, enlarging the input alphabet cannot further maximize the utility. The proof of Theorem 21 is given in Section 7.1.

Lemma 22 *A $k\times\ell$ conditional distribution Q is ε-locally differentially private if and only if it can be written as $Q = S\Theta$, where S is a $k\times\ell$ matrix with $S_{ij} \in [1, e^\varepsilon]$ and $\Theta = diag(\theta_1,\ldots,\theta_\ell)$ with its diagonal entries in $\mathbb{R}_+$.*

The proof of Lemma 22 is provided in Section 7.2. With the above results, we are now ready to prove Theorems 2 and 4. By Lemma 22, for any $Q \in \mathcal{D}_{\varepsilon,\ell}$ we have that $Q_j = \theta_j S_j$. Suppose $U(Q) = \sum_{j\in[\ell]}\mu(Q_j)$, where μ is a sublinear function. Since μ is sublinear, it is convex and $\mu(\theta_j S_j) = \theta_j\mu(S_j)$. $U(Q)$ is convex in Q because

$$\begin{aligned} U\left(\lambda Q^{(1)} + (1-\lambda)Q^{(2)}\right) &= \sum_{j\in[\ell]} \mu\left(\lambda\theta_j^{(1)}S_j^{(1)} + (1-\lambda)\theta_j^{(2)}S_j^{(2)}\right) \\ &\leq \sum_{j\in[\ell]} \lambda\mu\left(\theta_j^{(1)}S_j^{(1)}\right) + (1-\lambda)\mu\left(\theta_j^{(2)}S_j^{(2)}\right) \\ &= \lambda U\left(Q^{(1)}\right) + (1-\lambda)U\left(Q^{(2)}\right), \end{aligned} \tag{25}$$

for any $\lambda \in (0,1)$. Furthermore, $U(Q)$ satisfies Assumption 1 because $\mu(Q_j) = 0$ whenever $\theta_j = 0$. Let $Q^* = S^*\Theta^* \in \arg\max_{Q\in\cup_{\ell=1}^k\mathcal{D}_{\varepsilon,\ell}} U(Q)$ and note that by Theorem 21, $U(Q^*) =$

$\max_{Q\in\mathcal{D}_\varepsilon} U(Q)$. Suppose that Q^* is of dimensions $k\times\ell$, where $\ell \leq k$. Each of the ℓ columns of Q^* can be expressed as a convex combination of the columns of $S^{(k)}$, the staircase pattern matrix. This is because the 2^k columns of $S^{(k)}$ are the corner points of the cube $[1,e^\varepsilon]^k$ and each $S_j^* \in [1,e^\varepsilon]^k$. Therefore, $S_j^* = \sum_{i=1}^{2^k}\lambda_{ij}S_i^{(k)}$, where $\lambda_{ij} \geq 0$ for all i and j, and $\sum_{i=1}^{2^k}\lambda_{ij} = 1$ for all j. Create the 2^k-dimensional vector $\tilde\theta$ such that $\tilde\theta_i = \sum_{j=1}^{\ell}\lambda_{ij}\theta_j^*$ and let $\tilde Q = S^{(k)}\tilde\Theta$.

$$\begin{aligned} U(Q^*) - U(\tilde Q) &= \sum_{j=1}^{\ell}\mu\left(S_j^*\right)\theta_j^* - \sum_{i=1}^{2^k}\mu\left(\left(\sum_{j=1}^{\ell}\lambda_{ij}\theta_j^*\right)S_j^{(k)}\right) \\ &= \sum_{j=1}^{\ell}\mu\left(\sum_{i=1}^{2^k}\lambda_{ij}S_i^{(k)}\right)\theta_j^* - \sum_{i=1}^{2^k}\sum_{j=1}^{\ell}\lambda_{ij}\theta_j^*\mu\left(S_j^{(k)}\right) \\ &= \sum_{j=1}^{\ell}\theta_j^*\left\{\mu\left(\sum_{i=1}^{2^k}\lambda_{ij}S_i^{(k)}\right) - \sum_{i=1}^{2^k}\lambda_{ij}\mu\left(S_j^{(k)}\right)\right\} \\ &\leq 0, \end{aligned} \tag{26}$$

by the convexity of $\mu(z)$ and the non-negativity of θ_j^*'s. Moreover, observe that since $S^{(k)}\tilde\theta = \mathbb{1}$, $\tilde\theta$ is a valid choice for the linear program of (11). This implies that

$$\max_{S^{(k)}\theta=\mathbb{1},\theta\geq 0}\sum_{j=1}^{2^k}\mu\left(S_j^{(k)}\right)\theta_j \geq U(\tilde Q) \geq U(Q^*) = \max_{Q\in\mathcal{D}_\varepsilon} U(Q)$$

On the other hand, for any $\tilde Q = S^{(k)}\tilde\Theta$, where $\tilde\theta$ is valid for the linear program of (11), we have that $\tilde Q \in \mathcal{D}_{\varepsilon,2^k} \subset \mathcal{D}_\varepsilon$ and therefore, $\max_{S^{(k)}\theta=\mathbb{1},\theta\geq 0}\sum_{j=1}^{2^k}\mu\left(S_j^{(k)}\right)\theta_j \leq \max_{Q\in\mathcal{D}} U(Q)$. Thus, $\max_{S^{(k)}\theta=\mathbb{1},\theta\geq 0}\sum_{j=1}^{2^k}\mu\left(S_j^{(k)}\right)\theta_j = \max_{Q\in\mathcal{D}} U(Q)$. This proves Theorem 4.

The polytope given by $S^{(k)}\theta = \mathbb{1}$ and $\theta \geq 0$ is a closed and bounded one. Thus, the linear program of (11) is bounded and has a solution, say θ^*, at one of the corner points of the polytope. Since there are k equality constraints given by $S^{(k)}\theta = \mathbb{1}$ and 2^k inequality constraints given by $\theta \geq 0$, any corner point, including θ^*, cannot have more than k non-zero entries. Form $\tilde S$ by deleting the columns of $S^{(k)}$ corresponding to zero entries of θ^*. Similarly, form $\tilde\theta$ by deleting the zero entries of θ^* and let $\tilde Q = \tilde S\tilde\Theta$, where $\tilde\Theta = \text{diag}\tilde\theta$. It is easy to verify that $U(\tilde Q) = U(Q^*) = \mu^T\theta^*$; hence, $\tilde Q$ solves linear program of (11). Moreover, $\tilde Q$ has at most k columns and $\tilde S_{ij} = \{1, e^\varepsilon\}$. Therefore, $\tilde Q$ is a staircase mechanism of dimension $k\times\ell$, where $\ell \leq k$.

7.1 Proof of Theorem 21

We start the proof of Theorem 21 with an important lemma the proof of which is presented in Section 7.3.

Lemma 23 *The set of all $k\times\ell$, ε-locally differentially private mechanisms $\mathcal{D}_{\varepsilon,\ell}$ forms a closed and bounded polytope in $\mathbb{R}_+^{k\ell}$. Moreover, let Q be a corner point of the polytope formed by $\mathcal{D}_{\varepsilon,\ell}$, then Q has at most k non-zero columns.*

Fix an $\ell > k$. Since $U(Q)$ is convex in Q, it suffices to consider the corner points of $\mathcal{D}_{\varepsilon,\ell}$ when maximizing $U(Q)$ subject to $Q \in \mathcal{D}_{\varepsilon,\ell}$. By Lemma 23, any $Q^{(1)}$, a $k \times \ell$ corner point of $\mathcal{D}_{\varepsilon,\ell}$, has at most k non-zero columns. Therefore, the privatization mechanism $Q^{(2)}$, obtained by deleting the all-zero columns of $Q^{(1)}$, has at most k columns. Notice that $Q^{(2)} \in \cup_{i=1}^{k} \mathcal{D}_{\varepsilon,i}$. Since $U(Q)$ satisfies Assumption 1, we have that $U\left(Q^{(1)}\right) = U\left(Q^{(2)}\right)$ and therefore, it suffices to consider $Q \in \cup_{i=1}^{k} \mathcal{D}_{\varepsilon,i}$ when maximizing $U(Q)$ subject to $Q \in \mathcal{D}_{\varepsilon,\ell}$. Thus,

$$\begin{aligned} \sup_{Q \in \mathcal{D}_{\varepsilon}} U(Q) &= \sup_{\ell \in \mathbb{N}} \left\{ \max_{Q \in \mathcal{D}_{\varepsilon,\ell}} U(Q) \right\} \\ &= \sup_{\ell \in \mathbb{N}} \left\{ \max_{Q \in \cup_{i=1}^{k} \mathcal{D}_{\varepsilon,i}} U(Q) \right\} \\ &= \max_{Q \in \cup_{i=1}^{k} \mathcal{D}_{\varepsilon,i}} U(Q), \end{aligned} \tag{27}$$

which finishes the proof.

7.2 Proof of Lemma 22

Claim 1 *Let $Q \in \mathcal{D}_{\varepsilon,\ell}$. If $Q_{ij} = 0$ for some $j \in \{1, ..., \ell\}$ then $Q_{ij} = 0$ for all $i \in \{1, ..., k\}$.*

Proof Assume that $Q_{i_1 j} = 0$ and $Q_{i_2 j} \neq 0$ for some $i_1, i_2 \in \{1, ..., k\}$. It is obvious that $q(y_j|x_{i_2}) \leq q(y_j|x_{i_1}) e^{\varepsilon}$ is not satisfied. Therefore, Q is not a locally differentially private mechanism. ■

It is easy to check that any $k \times \ell$ stochastic matrix $Q = S\Theta$, where Θ is a diagonal matrix with non-negative entries and S is a $k \times \ell$ matrix with $S_{ij} \in [1, e^{\varepsilon}]$, satisfies the local differential privacy constraints. Thus, $Q \in \mathcal{D}_{\varepsilon,\ell}$. On the other hand, assume that $Q \in \mathcal{D}_{\varepsilon,\ell}$. If $Q_{ij} = 0$ for some j then by Claim 1 we have that $Q_{ij} = 0$ for all i and therefore, we can set $\theta_j = 0$ and $S_{ij} = 1$ for all i. If $Q_{ij} > 0$ then by Claim 1 we have that $Q_{ij} > 0$ for all i. In this case, let $\theta_j = \min_i Q_{ij}$ and observe that $\theta_j > 0$ since $Q_{ij} > 0$ for all i. Let $S_{ij} = Q_{ij}/\theta_i$, then it is clear (from the definition of θ_i) that $S_{ij} \geq 1$. On the other hand, from the differential privacy constraints, we have that $Q_{ij} \leq Q_{kj} e^{\varepsilon}$ for all k and thus, $Q_{ij} \leq \min_k Q_{kj} e^{\varepsilon}$ which proves that $S_{ij} = Q_{ij}/\min_k Q_{kj} \leq e^{\varepsilon}$. This establishes that any $Q \in \mathcal{D}_{\varepsilon,\ell}$ can be written as $Q = S\Theta$, where Θ is a diagonal matrix with non-negative entries and S is a $k \times \ell$ matrix with $S_{ij} \in [1, e^{\varepsilon}]$.

7.3 Proof of Lemma 23

We start by showing that $\mathcal{D}_{\varepsilon,\ell}$ forms a closed and bounded polytope in $\mathbb{R}_{+}^{k\ell}$. We are interested in studying the corner points of the polytope formed by $\mathcal{D}_{\varepsilon,\ell}$ because convex utility functions are maximized at one of these corner points whenever the space of privatization mechanisms is restricted to $\mathcal{D}_{\varepsilon,\ell}$.

Claim 2 *A privatization mechanism $Q \in \mathcal{D}_{\varepsilon,\ell}$ if and only if for all $x, x' \in \mathcal{X}$ and all $y \in \mathcal{Y}$ we have that $Q(y|x) \leq Q(y|x') e^{\varepsilon}$.*

Proof By definition, Q is differentially private if for all x, $x' \in \mathcal{X}$ and all $B \subseteq \mathcal{Y}$ we have that $Q(B|x) \leq Q(B|x') e^{\varepsilon}$. By choosing $B = \{y\}$ for some $y \in \mathcal{Y}$ the first direction of the above lemma is proven. In order to prove the other direction, assume that for all $x, x' \in \mathcal{X}$ and all $y \in \mathcal{Y}$ we have that $Q(y|x) \leq Q(y|x') e^{\varepsilon}$. Then for any $B \subseteq \mathcal{Y}$, the following holds

$$\begin{aligned} \sum_{y \in B} Q(y|x) &\leq \sum_{y \in B} Q\left(y|x'\right) e^{\varepsilon} \\ \Leftrightarrow \quad Q(B|x) &\leq Q\left(B|x'\right) e^{\varepsilon}. \end{aligned} \tag{28}$$

■

Let $Q \in \mathcal{D}_{\varepsilon,\ell}$, then by Claim 2, it is easy to see that Q must satisfy $\ell k(k-1)$ inequalities of the form $Q(y|x) \leq Q(y|x') e^{\varepsilon}$. These inequalities can be compactly represented by

$$\tilde{A} q \leq 0,$$

where $q = [Q(y_1|x_1), ..., Q(y_1|x_k), ..., Q(y_\ell|x_1), ..., Q(y_\ell|x_k)]^T$ and $\tilde{A}$ is a $\ell k(k-1) \times k\ell$ matrix that contains all the local differential privacy linear constraints. Observe that there is a one-to-one mapping between Q and q. Here is an example for the case when $k = \ell = 2$

$$\underbrace{\begin{bmatrix} 1 & -e^{\varepsilon} & 0 & 0 \\ -e^{\varepsilon} & 1 & 0 & 0 \\ 0 & 0 & 1 & -e^{\varepsilon} \\ 0 & 0 & -e^{\varepsilon} & 1 \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} Q(y_1|x_1) \\ Q(y_1|x_2) \\ Q(y_2|x_1) \\ Q(y_2|x_2) \end{bmatrix} \leq 0.$$

Moreover, since Q is a row stochastic matrix (a conditional distribution) it satisfies $Q\mathbb{1} = \mathbb{1}$, where $\mathbb{1}$ represents the all ones vector of appropriate dimensions. This condition can be rewritten as

$$Bq = \mathbb{1},$$

where B is a $k \times k\ell$ binary matrix. For the case when $k = \ell = 2$, we have that

$$\underbrace{\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} Q(y_1|x_1) \\ Q(y_1|x_2) \\ Q(y_2|x_1) \\ Q(y_2|x_2) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Finally, observe that $Q(y|x) \geq 0$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. These constraints can be incorporated as follows. Let $A = \left[\tilde{A}^T, -I_{\ell k}\right]^T$, where $I_{\ell k}$ is the $\ell k \times \ell k$ identity matrix, then $Aq \leq 0$. To summarize, $Q \in \mathcal{D}_{\varepsilon,\ell}$ if and only if

$$\begin{aligned} Aq &\leq 0 \\ Bq &= \mathbb{1}. \end{aligned} \tag{29}$$

Therefore, the set of all $k \times \ell$, ε-locally differentially private mechanisms $\mathcal{D}_{\varepsilon,\ell}$ forms a convex polytope in $\mathbb{R}_{+}^{k\ell}$.

We now proceed to proving that if Q is a corner point of the polytope formed by $\mathcal{D}_{\varepsilon,\ell}$, then Q has at most k non-zero columns. This claim is obvious for all $k \times \ell$ privatization mechanisms with $\ell \leq k$. Therefore, we restrict our attention to the case where $\ell > k$. Let A_j be the matrix including all the inequality constraints imposed on the j^{th} column of Q. Observe that the rows of A_j form a subset of the rows of A, defined in (29), and recall that there are $k(k-1)$ differential privacy and k non-negativity inequality constraints imposed on the j^{th} column of Q. Therefore, A_j is a $k^2 \times k$ matrix and we have that $A_j Q_j \leq 0$, where Q_j represents the j^{th} column of Q. By Claim 1, we know that Q_j is either equal to zero or contains non-zero entries.

Claim 3 *In what follows, the term linearly independent inequality constraints refers to linear independent rows of A_j.*

- *If $Q_j = 0$, then k linearly independent inequality constraints are achieved with equality.*
- *If $Q_j \neq 0$, then at most $k-1$ linearly independent inequality constraints can be achieved with equality.*

Proof In fact, the number of linearly independent inequality constraints (achieved or not) cannot exceed k because A_j has a rank less than or equal to k. If $Q_j = 0$, then the k non-negativity inequality constraints are achieved with equality and it is easy to see that they are all linearly independent (in fact, they form an orthonormal basis to $\mathbb{R}^k$). This proves the first part of the claim. We now establish the second part of the claim by showing that if $Q_j \neq 0$, we cannot have k linearly independent inequality constraints achieved with equality. Assume that $Q_j \neq 0$ and let $\tilde{A}_j$ be the matrix including the largest collection of linearly independent rows of A_j corresponding to the inequality constraints that are achieved with equality. In other words, $\tilde{A}_j Q_j = 0$. If $\tilde{A}_j$ contains k rows, then its rank is equal to k. However, this implies that $Q_j = 0$, a contradiction. Therefore, at most $k-1$ linearly independent inequality constraints can be achieved with equality when $Q_j \neq 0$. ■

Suppose that Q is a corner point of $\mathcal{D}_{\varepsilon,\ell}$ and out of its ℓ columns, $\ell_{>0}$ are non-zero and $\ell_{=0}$ ($\ell_{=0} = \ell - \ell_{>0}$) are zero. Moreover, assume that the number of non-zero columns of Q is larger than k (i.e., $\ell_{>0} > k$). In this case, from Claim 3, we can see that Q achieves at most $\ell_{>0}(k-1)+(\ell-\ell_{>0})k$ linearly independent inequality constraints with equality. Furthermore, at most k additional linearly independent equality constraints (linearly independent rows of the matrix B defined in (29)) can be met by Q. Therefore, the total number of linearly independent constraints that Q achieves with equality is at most $\ell_{>0}(k-1)+(\ell-\ell_{>0})k+k = -\ell_{>0} + (\ell+1)k < \ell k$, where the last strict inequality follows from the fact that $\ell_{>0} > k$. This implies that Q cannot be a corner point of $\mathcal{D}_{\varepsilon,\ell}$. Therefore, any corner point of $\mathcal{D}_{\varepsilon,\ell}$ must have at most k non-zero columns.

8. Proofs for Hypothesis Testing

8.1 Proof of Theorem 5

Let $T = \{x : P_0(x) \geq P_1(x)\}$. In other words, $P_0(T) - P_1(T) = \max_{A \subseteq \mathcal{X}} P_0(A) - P_1(A)$. Recall that for a given P_0 and P_1, the binary mechanism is defined as a staircase mechanism

with only two outputs $y \in \{0,1\}$ satisfying

$$Q(0|x) = \begin{cases} \frac{e^\varepsilon}{1+e^\varepsilon} & \text{if } P_0(x) \geq P_1(x)\,, \\ \frac{1}{1+e^\varepsilon} & \text{if } P_0(x) < P_1(x)\,. \end{cases} \quad Q(1|x) = \begin{cases} \frac{e^\varepsilon}{1+e^\varepsilon} & \text{if } P_0(x) < P_1(x)\,, \\ \frac{1}{1+e^\varepsilon} & \text{if } P_0(x) \geq P_1(x)\,. \end{cases} \tag{30}$$

Lemma 24 *For any pair of distributions P_0 and P_1, there exists a positive ε^* that depends on P_0 and P_1 such that for all $y \in \mathcal{Y}$, all $\ell \in \mathbb{N}$, and all $Q \in \mathcal{D}_{\varepsilon,\ell}$ with $\varepsilon \leq \varepsilon^*$, we have that*

$$\frac{(e^\varepsilon - 1)\,P_0\,(T^c) + 1}{(e^\varepsilon - 1)\,P_1\,(T^c) + 1} \leq \frac{M_0(y)}{M_1(y)} \leq \frac{(e^\varepsilon - 1)\,P_0\,(T) + 1}{(e^\varepsilon - 1)\,P_1\,(T) + 1}.$$

Moreover, the above upper and lower bounds are achieved by the binary mechanism given in (30).

Observe that because $P_0\,(T) \geq P_1\,(T)$ and $P_0\,(T^c) \leq P_1\,(T^c)$, the direction of the above inequalities makes sense.

Let $\tilde{M}_\nu$ be the induced marginal for the binary mechanism when P_ν is the original distribution. Following the analysis techniques developed in Kairouz et al. (2013), we define hypothesis testing region $R(\tilde{M}_0, \tilde{M}_1)$ as the convex hull of all achievable probabilities of missed detection and false alarm, when testing whether $\nu = 0$ or $\nu = 1$ based on Y_{bin} distributed as $\tilde{M}_\nu$:

$$R(\tilde{M}_0, \tilde{M}_1) \equiv \text{conv}\Big(\big\{(\tilde{M}_1(S), \tilde{M}_0(S^c)) : \forall S \subseteq \mathcal{Y}\big\}\Big)\,,$$

where $S \in \mathcal{Y}$ is the accept region for hypothesis $\nu = 0$. For the binary mechanism, this ends up being a very simple triangular region. The slopes defining the two sides of the triangular region are: $-\max_S \tilde{M}_0(S)/\tilde{M}_1(S) = -((e^\varepsilon - 1)P_0(T) + 1)/((e^\varepsilon - 1)P_1(T) + 1)$ and $-\min_S \tilde{M}_0(S^c)/\tilde{M}_1(S^c) = -((e^\varepsilon - 1)P_0(T^c) + 1)/((e^\varepsilon - 1)P_1(T^c) + 1)$.

For any other mechanism satisfying the ε-local differential privacy for $\varepsilon \leq \varepsilon^*$, the above lemma implies that for any choice of the rejection region S, the slopes satisfy $-M_0(S)/M_1(S) \geq -((e^\varepsilon - 1)P_0(T) + 1)/((e^\varepsilon - 1)P_1(T) + 1)$ and $-M_0(S^c)/M_1(S^c) \leq -((e^\varepsilon - 1)P_0(T^c) + 1)/((e^\varepsilon - 1)P_1(T^c) + 1)$. In the hypothesis testing region, this implies that

$$R(M_0, M_1) \subseteq R(\tilde{M}_0, \tilde{M}_1)\,,$$

as in the following Figure 10.

From Theorem 2.5 of Kairouz et al. (2013), we know that this implies a certain Markov property. Precisely, let Y_{bin} denote the output of the binary mechanism, and Y_{dp} denote the output of any ε-local differentially private mechanism. Then, it follows that there exists a coupling of Y_{bin} and Y_{dp} such that they form a Markov chain: ν–Y_{bin}–Y_{dp}, where ν is the hypothesis on P_ν whether the data was generated from $\nu = 0$ or $\nu = 1$. Then, it follows from the data processing inequality of f-divergences that

$$D_f(\tilde{M}_0, \tilde{M}_1) \geq D_f(M_0, M_1)\,.$$

It follows that there is no algorithm with larger f-divergence than the binary mechanism.

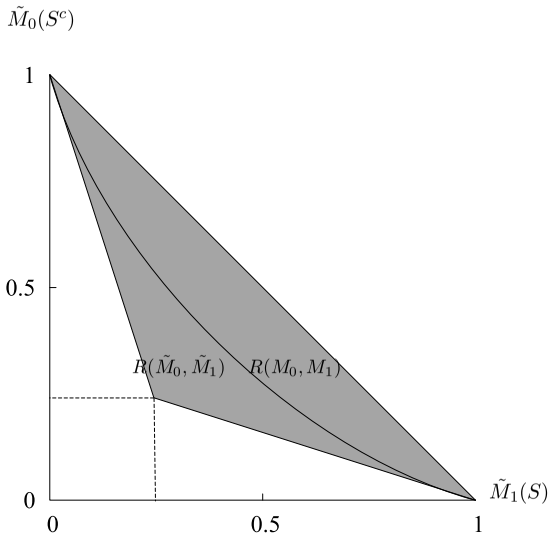


Figure 10: Hypothesis testing regions for two mechanisms.

8.2 Proof of Lemma 24

We start by showing that the binary mechanism achieves the upper and lower bounds presented in the statement of the lemma. Let M_0^B and M_1^B denote the induced marginals under the binary mechanism given in (30). For $\nu \in \{0,1\}$, we have that

$$\begin{aligned} M_\nu^B(0) &= \sum_{x\in\mathcal{X}} P_0(x)\, Q(0|x) = \frac{1}{e^\varepsilon+1}\left(\left(e^\varepsilon-1\right)P_\nu(T)+1\right) \\ M_\nu^B(1) &= \sum_{x\in\mathcal{X}} P_0(x)\, Q(1|x) = \frac{1}{e^\varepsilon+1}\left(\left(e^\varepsilon-1\right)P_\nu(T^c)+1\right). \end{aligned} \tag{31}$$

Computing $M_0^B(0)/M_1^B(0)$ and $M_0^B(1)/M_1^B(1)$ we see that the binary mechanism achieves the upper and lower bounds for all values of ε.

As in Lemma 22, for any $\ell \in \mathbb{N}$, $Q \in \mathcal{D}_{\varepsilon,\ell}$ can be represented as $Q = S\Theta$, where $S \in [1,e^\varepsilon]^{k\times\ell}$ and $\Theta = \operatorname{diag}(\theta_1,...,\theta_\ell)$ with its diagonal entries in $\mathbb{R}_+$. We now show that for any $\ell \in \mathbb{N}$ and any $Q \in \mathcal{D}_{\varepsilon,\ell}$, the following upper bound holds

$$\frac{M_0(y)}{M_1(y)} = \frac{\sum_{i\in[k]} P_0(x_i)\, S_{ij}}{\sum_{i\in[k]} P_1(x_i)\, S_{ij}} \leq \frac{\left(e^\varepsilon-1\right)P_0(T)+1}{\left(e^\varepsilon-1\right)P_1(T)+1},$$

for all $y \in \mathcal{Y}$ and sufficiently small ε. The above expression can be alternatively written as

$$\begin{aligned} \left(e^\varepsilon-1\right)\left(P_0(T)-P_1(T)\right) + \left(e^\varepsilon-1\right)\sum_{i\in[k]}\left(S_{ij}-1\right)\left(P_0(T)P_1(x_i)-P_1(T)P_0(x_i)\right) \\ -\sum_{i\in[k]}\left(S_{ij}-1\right)\left(P_0(x_i)-P_1(x_i)\right)\geq 0, \end{aligned} \tag{32}$$

where $S_j \in [1,e^\varepsilon]^k$. Equation (32) is linear in S_j and is therefore minimized (and maximized) at the corner points of $[1,e^\varepsilon]^{k\times\ell}$, a cube in $\mathbb{R}_+^{k\times\ell}$. The corner points of this cube are given by the staircase patterns: $S_j \in \{1,e^\varepsilon\}^k$. To begin with, let S_j be a staircase pattern with $T_j = \{x_i : S_{ij} = e^\varepsilon\} \neq T$. Then Equation (32) is equivalent to

$$\begin{aligned} \left(e^\varepsilon-1\right)\left\{\left(P_0(T)-P_1(T)\right)-\left(P_0(T_j)-P_1(T_j)\right)\right\} \\ +\left(e^\varepsilon-1\right)^2\left\{P_0(T)P_1(T_j)-P_1(T)P_0(T_j)\right\}\geq 0. \end{aligned} \tag{33}$$

Using the fact that $P_0(T)-P_1(T) > P_0(T_j)-P_1(T_j)$ for all $T_j \neq T$, the inequality in (32) holds true for all ε whenever $P_0(T)P_1(T_j) \geq P_1(T)P_0(T_j)$. If $P_0(T)P_1(T_j) < P_1(T)P_0(T_j)$, then the inequality in (32) holds true for all $\varepsilon \leq \varepsilon(j)$, where

$$\varepsilon(j) = \log\left(\frac{\left(P_0(T)-P_1(T)\right)-\left(P_0(T_j)-P_1(T_j)\right)}{P_1(T)P_0(T_j)-P_0(T)P_1(T_j)}+1\right) > 0.$$

On the other hand, it is easy to verify that when $T_j = T$, we have that

$$\begin{aligned} \left(e^\varepsilon-1\right)\left\{\left(P_0(T)-P_1(T)\right)-\left(P_0(T_j)-P_1(T_j)\right)\right. \\ \left.+\left(e^\varepsilon-1\right)\left(P_0(T)P_1(T_j)-P_1(T)P_0(T_j)\right)\right\} = 0, \end{aligned} \tag{34}$$

for all ε. In this case, set $\varepsilon(j) = 0$ and $\varepsilon_1 = \min_{j\in[2^k]}\varepsilon(j)$. Therefore, for any $\ell \in \mathbb{N}$ and any $Q \in \mathcal{D}_{\varepsilon,\ell}$, the upper bound in the statement of the lemma holds for all $\varepsilon \leq \varepsilon_1$.

We now show that for for any $\ell \in \mathbb{N}$ and any $Q \in \mathcal{D}_{\varepsilon,\ell}$, the following lower bound holds

$$\frac{\left(e^\varepsilon-1\right)P_0(T^c)+1}{\left(e^\varepsilon-1\right)P_1(T^c)+1} \leq \frac{M_0(y)}{M_1(y)} = \frac{\sum_{i\in[k]} P_0(x_i)\, S_{ij}}{\sum_{i\in[k]} P_1(x_i)\, S_{ij}},$$

for all $y \in \mathcal{Y}$ and sufficiently small ε. The above expression can be alternatively written as

$$\begin{aligned} \left(e^\varepsilon-1\right)\left(P_0(T)-P_1(T)\right) + \left(e^\varepsilon-1\right)\sum_{i\in[k]}\left(S_{ij}-1\right)\left(P_0(T)P_1(x_i)-P_1(T)P_0(x_i)\right) \\ +e^\varepsilon\sum_{i\in[k]}\left(S_{ij}-1\right)\left(P_0(x_i)-P_1(x_i)\right)\geq 0, \end{aligned} \tag{35}$$

where $S_j \in [1,e^\varepsilon]^k$. Equation (35) is linear in S_j and is therefore minimized at the corner points of $[1,e^\varepsilon]^k$, a cube in $\mathbb{R}_+^k$. The corner points of this cube are given by staircase patterns: $S_j \in \{1,e^\varepsilon\}^k$. To begin with, let S_j be a staircase pattern with $T_j = \{x_i : S_{ij} = e^\varepsilon\} \neq T^c$, then Equation (35) is equivalent to

$$\begin{aligned} \left(e^\varepsilon-1\right)\left\{\left(P_0(T)-P_1(T)\right)+e^\varepsilon\left(P_0(T_j)-P_1(T_j)\right)\right\} \\ +\left(e^\varepsilon-1\right)^2\left\{P_0(T)P_1(T_j)-P_1(T)P_0(T_j)\right\}\geq 0. \end{aligned} \tag{36}$$

Using the fact that $P_0(T) - P_1(T) > P_1(T_j) - P_0(T_j)$ for all $T_j \neq T^c$, then for sufficiently small ε, Equation (35) can be written as

$$\varepsilon\left\{(P_0(T) - P_1(T)) - (P_1(T_j) - P_0(T_j))\right\} + O\left(\varepsilon^2\right) > 0.$$

This proves that there exists a positive $\varepsilon(j)$ such that the left hand side of Equation (36) is positive for all $\varepsilon \leq \varepsilon(j)$. On the other hand, it is easy to verify that when $T_j = T^c$, we have that

$$\begin{aligned}(e^\varepsilon - 1)\left\{(P_0(T) - P_1(T)) + e^\varepsilon (P_0(T_j) - P_1(T_j))\right.\\ \left. + (e^\varepsilon - 1)(P_0(T) P_1(T_j) - P_1(T) P_0(T_j))\right\} = 0, \quad (37)\end{aligned}$$

for all ε. In this case, let $\varepsilon(j) = 0$ and let $\varepsilon_2 = \min_{j\in[2^k]} \varepsilon(j)$. Therefore, for any $\ell \in \mathbb{N}$ and any $Q \in \mathcal{D}_{\varepsilon,\ell}$, the lower bound in the statement of the lemma holds for all $\varepsilon \leq \varepsilon_2$. To conclude, let $\varepsilon^* = \min(\varepsilon_1, \varepsilon_2)$. Then both, the upper and lower bounds, hold for all $\varepsilon \leq \varepsilon^*$.

8.3 Proof of Theorem 6

The total variation (TV) distance $\|M_0 - M_1\|_{\text{TV}}$ is a special case of f-divergence $D_f(M_0||M_1)$ with $f(x) = \frac{1}{2}|x-1|$. Therefore, by Theorem 4, we have that

$$\max_{Q\in\mathcal{D}_\varepsilon} \left\|M_0 - M_1\right\|_{\text{TV}} = \begin{array}{ll}\underset{\theta}{\text{maximize}} & \mu^T\theta \\ \text{subject to} & S^{(k)}\theta = \mathbb{1} \\ & \theta \geq 0,\end{array} \quad (38)$$

where $\mu_j = \mu\left(S_j^{(k)}\right) = \frac{1}{2}\left|\sum_{i\in[k]} (P_0(x_i) - P_1(x_i)) S_{ij}^{(k)}\right|$ for $j \in \{1, \ldots, 2^k\}$ and $S^{(k)}$ is the $k \times 2^k$ staircase pattern matrix given in Definition 3.

The polytope given by $S^{(k)}\theta = \mathbb{1}$ and $\theta \geq 0$ is a closed and bounded one. Thus, there is no duality gap and solving the above linear program is equivalent to solving its dual

$$\begin{array}{ll}\underset{\alpha}{\text{minimize}} & \mathbb{1}^T\alpha \\ \text{subject to} & {S^{(k)}}^T\alpha \geq \mu.\end{array} \quad (39)$$

Note that any satisfiable solution α^* to (39) provides an upper bound to (38) since $\max \mu^T\theta = \min \mathbb{1}^T\alpha \leq \mathbb{1}^T\alpha^*$. Let $T = \{x : P_0(x) \geq P_1(x)\}$ and $T_j = \{x_i : S_{ij}^{(k)} = e^\varepsilon\}$ for $j \in [2^k]$. Consider the following choice of dual variable

$$\alpha_i^* = \frac{1}{2}\frac{e^\varepsilon - 1}{e^\varepsilon + 1}\left|P_0(x_i) - P_1(x_i)\right|,$$

for $i \in [k]$. Observe that

$$\begin{aligned}\mathbb{1}^T\alpha^* &= \frac{1}{2}\frac{e^\varepsilon - 1}{e^\varepsilon + 1}\sum_{i\in[k]}\left|P_0(x_i) - P_1(x_i)\right| \\ &= \frac{1}{2}\frac{e^\varepsilon - 1}{e^\varepsilon + 1}\left\|P_0 - P_1\right\|_1 \\ &= \frac{e^\varepsilon - 1}{e^\varepsilon + 1}\left\|P_0 - P_1\right\|_{\text{TV}}. \quad (40)\end{aligned}$$

We claim that α^* is a feasible dual variable for all values of ε. In order to prove that α^* is a feasible dual variable, we show that ${S^{(k)}}_j^T\alpha^* - \mu_j \geq 0$ for all $j \in [2^k]$ and all ε. For all $j \in [2^k]$, we have that

$$\begin{aligned}g_j &= 2\left({S^{(k)}}_j^T\alpha^* - \mu_j\right) \\ &= \frac{e^\varepsilon - 1}{e^\varepsilon + 1}\sum_{i\in[k]}\left|P_0(x_i) - P_1(x_i)\right| S_{ij}^{(k)} - \left|\sum_{i\in[k]}\left(P_0(x_i) - P_1(x_i)\right) S_{ij}^{(k)}\right| \\ &= \frac{e^\varepsilon - 1}{e^\varepsilon + 1}\left\{\sum_{x_i\in T}\left(P_0(x_i) - P_1(x_i)\right) S_{ij}^{(k)} + \sum_{x_i\in T^c}\left(P_1(x_i) - P_0(x_i)\right) S_{ij}^{(k)}\right\} \\ &\quad - \left|\sum_{x_i\in T}\left(P_0(x_i) - P_1(x_i)\right) S_{ij}^{(k)} - \sum_{x_i\in T^c}\left(P_1(x_i) - P_0(x_i)\right) S_{ij}^{(k)}\right|. \quad (41)\end{aligned}$$

Notice that we have arranged the equation such that all the summands are non-negative. Without loss of generality, we will assume that

$$\sum_{x_i\in T}\left(P_0(x_i) - P_1(x_i)\right) S_{ij}^{(k)} \geq \sum_{x_i\in T^c}\left(P_1(x_i) - P_0(x_i)\right) S_{ij}^{(k)}.$$

From the equality $\sum_{x_i\in T}(P_0(x_i) - P_1(x_i)) = \sum_{x_i\in T^c}(P_1(x_i) - P_0(x_i))$ and the fact that $S_{ij}^{(k)} \in \{1, e^\varepsilon\}$ for all i and j, it follows that

$$e^{-\varepsilon}\sum_{x_i\in T}\left(P_0(x_i) - P_1(x_i)\right) S_{ij}^{(k)} \leq \sum_{x_i\in T^c}\left(P_1(x_i) - P_0(x_i)\right) S_{ij}^{(k)}\ . \quad (42)$$

This is true because the right-hand side is minimized when the $S_{ij}^{(k)}$'s for $x_i \in T^c$ are all equal to 1 and the left-hand side is maximized when the $S_{ij}^{(k)}$'s for $x_i \in T$ are all equal to e^ε. Now, (41) can be written as

$$\begin{aligned}g_j &= \frac{1}{e^\varepsilon + 1}\left\{-2\sum_{x_i\in T}\left(P_0(x_i) - P_1(x_i)\right) S_{ij}^{(k)} + 2e^\varepsilon\sum_{x_i\in T^c}\left(P_1(x_i) - P_0(x_i)\right) S_{ij}^{(k)}\right\} \\ &\geq 0\,,\end{aligned}$$

where the last inequality follows from (42).

This establishes the satisfiability of α^* for all ε which, in turn, shows that (40) is indeed an upper bound to the primal problem. It remains to show that this upper bound can be achieved via the binary mechanism. To this extent, recall that for a given P_0 and P_1, the binary mechanism is defined as a staircase mechanism with only two outputs $y \in \{0, 1\}$ satisfying

$$Q(0|x) = \begin{cases}\frac{e^\varepsilon}{1+e^\varepsilon} & \text{if } P_0(x) \geq P_1(x)\,, \\ \frac{1}{1+e^\varepsilon} & \text{if } P_0(x) < P_1(x)\,.\end{cases} \quad Q(1|x) = \begin{cases}\frac{e^\varepsilon}{1+e^\varepsilon} & \text{if } P_0(x) < P_1(x)\,, \\ \frac{1}{1+e^\varepsilon} & \text{if } P_0(x) \geq P_1(x)\,.\end{cases} \quad (43)$$

Computing the TV distance between M_0 and M_1 under (43), we get that

$$\left\|M_0 - M_1\right\|_{\text{TV}} = \frac{e^\varepsilon - 1}{e^\varepsilon + 1}\left\|P_0 - P_1\right\|_{\text{TV}}.$$

Hence, the binary mechanism in (43) achieves the upper bound in (40). This proves the optimality of the binary mechanism for all ε.

8.4 Proof of Theorem 8

The Kullback-Leibler (KL) divergence $D_{\text{kl}}(M_0||M_1)$ is a special f-divergence $D_f(M_0||M_1)$ with $f(x) = x\log x$. Therefore, by Theorem 4, we have that

$$\begin{array}{rll} \max\limits_{Q\in\mathcal{D}_\varepsilon} D_{\text{kl}}(M_0||M_1) = & \underset{\theta}{\text{maximize}} & \mu^T\theta \\ & \text{subject to} & S^{(k)}\theta = \mathbb{1} \\ & & \theta \geq 0, \end{array} \tag{44}$$

where $\mu_j = \mu\left(S_j^{(k)}\right) = \sum_{i\in[k]} P_0(x_i)S_{ij}^{(k)}\log\left(\frac{\sum_{i\in[k]}P_0(x_i)S_{ij}^{(k)}}{\sum_{i\in[k]}P_1(x_i)S_{ij}^{(k)}}\right)$ for $j\in\{1,\ldots,2^k\}$ and $S^{(k)}$ is the $k\times 2^k$ staircase pattern matrix given in Definition 3.

The polytope given by $S^{(k)}\theta = \mathbb{1}$ and $\theta\geq 0$ is a closed and bounded one. Thus, there is no duality gap and solving the above linear program is equivalent to solving its dual

$$\begin{array}{ll} \underset{\alpha}{\text{minimize}} & \mathbb{1}^T\alpha \\ \text{subject to} & {S^{(k)}}^T\alpha \geq \mu. \end{array} \tag{45}$$

Note that any satisfiable solution α^* to (45) provides an upper bound to (44) since $\max\mu^T\theta = \min\mathbb{1}^T\alpha \leq \mathbb{1}^T\alpha^*$. Let $T = \{x : P_0(x)\geq P_1(x)\}$ and $T_j = \{x_i : S_{ij}^{(k)} = e^\varepsilon\}$ for $j\in[2^k]$. Set $j_i = \{j : T_j = x_i\}$ for $i\in[k]$, and consider the following choice of dual variable

$$\alpha_i^* = \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{(e^\varepsilon+k-2)\,\mu\left(S_{j_i}^{(k)}\right) - \sum_{l\in[k],l\neq i}\mu\left(S_{j_l}^{(k)}\right)\right\},$$

for $i\in[k]$. Observe that since $T_{j_i} = x_i$ we have that $P_\nu(T_{j_i}) = P_\nu(x_i)$ and since

$$\begin{aligned} \mu_j &= \sum_{i\in[k]} P_0(x_i)S_{ij}^{(k)}\log\left(\frac{\sum_{i\in[k]}P_0(x_i)S_{ij}^{(k)}}{\sum_{i\in[k]}P_1(x_i)S_{ij}^{(k)}}\right) \\ &= (P_0(T_j)(e^\varepsilon-1)+1)\log\frac{(P_0(T_j)(e^\varepsilon-1)+1)}{(P_1(T_j)(e^\varepsilon-1)+1)} \end{aligned} \tag{46}$$

we have that

$$\begin{aligned} \mathbb{1}^T\alpha^* &= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\sum_{i\in[k]}\left\{(e^\varepsilon+k-2)\,\mu\left(S_{j_i}^{(k)}\right) - \sum_{l\in[k],l\neq i}\mu\left(S_{j_l}^{(k)}\right)\right\} \\ &= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{(e^\varepsilon+k-2)\sum_{i\in[k]}\mu\left(S_{j_i}^{(k)}\right) - \sum_{i\in[k]}\sum_{l\in[k],l\neq i}\mu\left(S_{j_l}^{(k)}\right)\right\} \\ &= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{(e^\varepsilon+k-2)\sum_{i\in[k]}\mu\left(S_{j_i}^{(k)}\right) - (k-1)\sum_{i\in[k]}\mu\left(S_{j_i}^{(k)}\right)\right\} \\ &= \frac{1}{(e^\varepsilon+k-1)}\sum_{i\in[k]}\mu\left(S_{j_i}^{(k)}\right) \\ &= \frac{1}{(e^\varepsilon+k-1)}\sum_{i\in[k]}(P_0(x_i)(e^\varepsilon-1)+1)\log\frac{(P_0(x_i)(e^\varepsilon-1)+1)}{(P_1(x_i)(e^\varepsilon-1)+1)}. \end{aligned} \tag{47}$$

We claim that α^* is a feasible dual variable for sufficiently large ε. In order to prove that α^* is a feasible dual variable, we show that ${S^{(k)}}_j^T\alpha^* - \mu_j \geq 0$ for all $j\in[2^k]$ for all $\varepsilon\geq\varepsilon^*$, where ε^* is a positive quantity that depends on the priors P_0 and P_1. Using the facts that

$$\begin{aligned} \log(a+e^\varepsilon b) &= \varepsilon + \log b + O\left(e^{-\varepsilon}\right) \\ \frac{1}{e^\varepsilon+k-1} &= e^{-\varepsilon} + O\left(e^{-2\varepsilon}\right), \end{aligned} \tag{48}$$

for large ε, we get that

$$\begin{aligned} \mu_j &= (P_0(T_j)(e^\varepsilon-1)+1)\log\frac{(P_0(T_j)(e^\varepsilon-1)+1)}{(P_1(T_j)(e^\varepsilon-1)+1)} \\ &= \left(P_0(T_j)\log\frac{P_0(T_j)}{P_1(T_j)}\right)e^\varepsilon + (1-P_0(T_j))\log\frac{P_0(T_j)}{P_1(T_j)} + O\left(e^{-\varepsilon}\right). \end{aligned} \tag{49}$$

On the other hand,

$$\begin{aligned} {S^{(k)}}_j^T\alpha^* &= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{\sum_{i\in[k]}S_{ij}^{(k)}(e^\varepsilon+k-2)\left(P_0(x_i)\log\frac{P_0(x_i)}{P_1(x_i)}e^\varepsilon + O(1)\right)\right\} \\ &\quad - \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{\sum_{i\in[k]}\sum_{l\in[k],l\neq i}S_{ij}^{(k)}\left(P_0(x_l)\log\frac{P_0(x_l)}{P_1(x_l)}e^\varepsilon + O(1)\right)\right\} \\ &= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left(\left(\sum_{x_i\in T_j}P_0(x_i)\log\frac{P_0(x_i)}{P_1(x_i)}\right)e^{3\varepsilon} + O\left(e^{2\varepsilon}\right)\right) \\ &= \left(\sum_{x_i\in T_j}P_0(x_i)\log\frac{P_0(x_i)}{P_1(x_i)}\right)e^\varepsilon + O(1). \end{aligned} \tag{50}$$

Assume, to begin with, that $j \neq \{j_1, j_2, ..., j_k\}$. Then

$$S^{(k)}{}_j^T \alpha^* - \mu_j = \left(P_0(T_j) \log \frac{P_0(T_j)}{P_1(T_j)} - \sum_{x_i \in T_j} P_0(x_i) \log \frac{P_0(x_i)}{P_1(x_i)} \right) e^\varepsilon + O(1).$$

Notice that for $j \neq \{j_1, j_2, ..., j_k\}$, $P_0(T_j) \log \frac{P_0(T_j)}{P_1(T_j)} > \sum_{x_i \in T_j} P_0(x_i) \log \frac{P_0(x_i)}{P_1(x_i)}$ by the log-sum inequality. Therefore, there exists a $\varepsilon(j) > 0$ such that $S^{(k)}{}_j^T \alpha^* - \mu_j \geq 0$ for all $\varepsilon \geq \varepsilon(j)$. If $j \in \{j_1, j_2, ..., j_k\}$, it is not hard to check that $S^{(k)}{}_j^T \alpha^* - \mu_j = 0$ for all ε. In this case, set $\varepsilon(j) = 0$. This establishes the satisfiability of α^* for all $\varepsilon \geq \varepsilon^* = \max_{j \in [2^k]} \varepsilon(j)$. The satisfiability of α^*, in turn, shows that (47) is indeed an upper bound to the primal problem. It remains to show that this upper bound can be achieved via the randomized response. To this extent, recall that the randomized response is given by

$$Q(y|x) = \begin{cases} \frac{e^\varepsilon}{|\mathcal{X}|-1+e^\varepsilon} & \text{if } y = x, \\ \frac{1}{|\mathcal{X}|-1+e^\varepsilon} & \text{if } y \neq x. \end{cases} \tag{51}$$

Computing the KL divergence between M_0 and M_1 under (51), we get that

$$D_{\mathrm{kl}}(M_0||M_1) = \frac{1}{(e^\varepsilon + k - 1)} \sum_{i \in [k]} (P_0(x_i)(e^\varepsilon - 1) + 1) \log \frac{(P_0(x_i)(e^\varepsilon - 1) + 1)}{(P_1(x_i)(e^\varepsilon - 1) + 1)}.$$

Hence, the randomized response in (51) achieves the upper bound in (47). This proves the optimality of the randomized response for all $\varepsilon \geq \varepsilon^*$.

8.5 Proof of Theorem 7

We start the proof with a fundamental bound on the symmetrized KL divergence between the M_0 and M_1.

Lemma 25 *For any $\varepsilon \geq 0$, let Q be any conditional distribution that guarantees ε differential privacy. Then for any pair of distributions P_0 and P_1, the induced marginals M_0 and M_1 must satisfy the bound*

$$D_{\mathrm{kl}}\big(M_0||M_1\big) + D_{\mathrm{kl}}\big(M_1||M_0\big) \leq 4(e^\varepsilon - 1)^2 \big\|P_0 - P_1\big\|_{\mathrm{TV}}^2.$$

The above lemma appears as Theorem 1 in Duchi et al. (2013). By Lemma 25, we have that

$$\mathrm{OPT} = \max_{Q \in \mathcal{D}_\varepsilon} D_{\mathrm{kl}}\big(M_0||M_1\big) \leq 4(e^\varepsilon - 1)^2 \big\|P_0 - P_1\big\|_{\mathrm{TV}}^2. \tag{52}$$

Let M_0^B and M_1^B be the marginals obtained by using the binary mechanism given in (14). By Corollary 11, we have that $\|M_0^B - M_1^B\|_{\mathrm{TV}} = \frac{e^\varepsilon - 1}{e^\varepsilon + 1} \|P_0 - P_1\|_{\mathrm{TV}}$. Consequently, by applying Pinsker's inequality to the KL divergence between M_0^B and M_1^B we get that

$$\begin{aligned} \mathrm{BIN} &= D_{\mathrm{kl}}\big(M_0^B||M_1^B\big) \\ &\geq 2\big\|M_0^B - M_1^B\big\|_{\mathrm{TV}}^2 \\ &= 2\left(\frac{e^\varepsilon - 1}{e^\varepsilon + 1}\right)^2 \big\|P_0 - P_1\big\|_{\mathrm{TV}}^2. \end{aligned} \tag{53}$$

Combining (52) and (53) we get that $\mathrm{BIN} \geq \frac{1}{2(e^\varepsilon+1)^2}\mathrm{OPT}$ which was to be shown.

9. Proofs for Information Preservation

9.1 Proof of Theorem 12

By Theorem 4, we have that

$$\begin{aligned} \max_{Q \in \mathcal{D}_\varepsilon} I(X;Y) = \quad & \underset{\theta}{\text{maximize}} && \mu^T \theta \\ & \text{subject to} && S^{(k)}\theta = \mathbb{1} \\ & && \theta \geq 0, \end{aligned} \tag{54}$$

where $\mu_j = \mu\left(S_j^{(k)}\right) = \sum_{i \in [k]} P(x_i) S_{ij}^{(k)} \log\left(\frac{S_{ij}^{(k)}}{\sum_{i \in [k]} P(x_i) S_{ij}^{(k)}}\right)$ for $j \in \{1, \ldots, 2^k\}$ and $S^{(k)}$ is the $k \times 2^k$ staircase pattern matrix given in Definition 3. The polytope given by $S^{(k)}\theta = \mathbb{1}$ and $\theta \geq 0$ is a closed and bounded one. Thus, there is no duality gap and solving the above linear program is equivalent to solving its dual

$$\begin{aligned} & \underset{\alpha}{\text{minimize}} && \mathbb{1}^T \alpha \\ & \text{subject to} && S^{(k)^T} \alpha \geq \mu. \end{aligned} \tag{55}$$

Note that any satisfiable solution α^* to (55) provides an upper bound to (54) since $\max \mu^T\theta = \min \mathbb{1}^T\alpha \leq \mathbb{1}^T\alpha^*$. Let $T_j = \{x_i : S_{ij}^{(k)} = e^\varepsilon\}$ and set $j_1 = \{j : T_j = T\}$ and $j_2 = \{j : T_j = T^c\}$. Consider the following choice of dual variable

$$\alpha_i^* = \frac{1}{(e^\varepsilon + 1)(e^\varepsilon - 1)} \begin{cases} \frac{e^\varepsilon \mu\left(S_{j_1}^{(k)}\right) - \mu\left(S_{j_2}^{(k)}\right)}{|T|} & \forall i \in T \\ \frac{e^\varepsilon \mu\left(S_{j_2}^{(k)}\right) - \mu\left(S_{j_1}^{(k)}\right)}{|T^c|} & \forall i \in T^c \end{cases}.$$

Observe that since $T_{j_1} = T$, $T_{j_2} = T^c$, and

$$\mu_j = P(T_j) e^\varepsilon \log \frac{e^\varepsilon}{P(T_j{}^c) + e^\varepsilon P(T_j)} + P(T_j{}^c) \log \frac{1}{P(T_j{}^c) + e^\varepsilon P(T_j)}, \tag{56}$$

we have that

$$\begin{aligned} \mathbb{1}^T\alpha^* &= \frac{1}{(e^\varepsilon+1)(e^\varepsilon-1)} \left\{ \sum_{i \in T} \frac{1}{|T|} \left(e^\varepsilon \mu\left(S_{j_1}^{(k)}\right) - \mu\left(S_{j_2}^{(k)}\right)\right) \right. \\ &= \left. + \sum_{i \in T^c} \frac{1}{|T^c|} \left(e^\varepsilon \mu\left(S_{j_2}^{(k)}\right) - \mu\left(S_{j_1}^{(k)}\right)\right) \right\} \\ &= \frac{1}{(e^\varepsilon+1)} \left(\mu\left(S_{j_1}^{(k)}\right) + \mu\left(S_{j_1}^{(k)}\right)\right) \\ &= \frac{1}{e^\varepsilon+1} \left\{ P(T) e^\varepsilon \log \frac{e^\varepsilon}{P(T^c) + e^\varepsilon P(T)} + P(T^c) \log \frac{1}{P(T^c) + e^\varepsilon P(T)} \right\} + \\ &\quad \frac{1}{e^\varepsilon+1} \left\{ P(T^c) e^\varepsilon \log \frac{e^\varepsilon}{P(T) + e^\varepsilon P(T^c)} + P(T) \log \frac{1}{P(T) + e^\varepsilon P(T^c)} \right\}. \end{aligned} \tag{57}$$

We claim that α^* is a feasible dual variable for sufficiently small ε. In order to prove that α^* is a feasible dual variable, we show that $\left(S^{(k)^T}\alpha^*\right)_j - \mu_j \geq 0$ for all $j \in \{1,\ldots,2^k\}$ and all $\varepsilon \leq \varepsilon^*$, where ε^* is a positive quantity that depends on P. Using the following facts

$$\begin{aligned} e^{\varepsilon} &= 1+\varepsilon+\frac{1}{2}\varepsilon+O\left(\varepsilon^3\right) \\ \log\left(a+e^{\varepsilon}b\right) &= b\varepsilon+\frac{b(1-b)}{2}\varepsilon^2+O\left(\varepsilon^3\right) \\ \frac{1}{1+e^{\varepsilon}} &= \frac{1}{2}-\frac{1}{4}\varepsilon+O\left(\varepsilon^2\right), \end{aligned} \tag{58}$$

for small ε, we get that

$$\begin{aligned} \mu_j &= P\left(T_j\right)e^{\varepsilon}\log\frac{e^{\varepsilon}}{P\left(T_j{}^c\right)+e^{\varepsilon}P\left(T_j\right)}+P\left(T_j{}^c\right)\log\frac{1}{P\left(T_j{}^c\right)+e^{\varepsilon}P\left(T_j\right)} \\ &= P\left(T_j\right)e^{\varepsilon}\varepsilon-\left(P\left(T_j\right)\left(e^{\varepsilon}-1\right)+1\right)\log\left(P\left(T_j\right)\left(e^{\varepsilon}-1\right)+1\right) \\ &= \frac{1}{2}P\left(T_j\right)P\left(T_j^c\right)\varepsilon^2+O\left(\varepsilon^3\right). \end{aligned} \tag{59}$$

On the other hand,

$$\begin{aligned} \left(S^{(k)^T}\alpha^*\right)_j &= S_j^{(k)^T}\alpha^* \\ &= \frac{1}{\left(e^{\varepsilon}+1\right)\left(e^{\varepsilon}-1\right)}\left\{\sum_{i\in T}\frac{S_{ij}^{(k)}}{|T|}\left(e^{\varepsilon}\mu\left(S_{j_1}^{(k)}\right)-\mu\left(S_{j_2}^{(k)}\right)\right)\right. \\ &\qquad \left.+\sum_{i\in T^c}\frac{S_{ij}^{(k)}}{|T^c|}\left(e^{\varepsilon}\mu\left(S_{j_2}^{(k)}\right)-\mu\left(S_{j_1}^{(k)}\right)\right)\right\} \\ &= \frac{1}{\left(e^{\varepsilon}+1\right)\left(e^{\varepsilon}-1\right)}\left(e^{\varepsilon}\mu\left(S_{j_1}^{(k)}\right)-\mu\left(S_{j_2}^{(k)}\right)\right)\left(\frac{|T_j\cap T|}{|T|}e^{\varepsilon}+\frac{|T_j^c\cap T|}{|T|}\right) \\ &\quad +\frac{1}{\left(e^{\varepsilon}+1\right)\left(e^{\varepsilon}-1\right)}\left(e^{\varepsilon}\mu\left(S_{j_2}^{(k)}\right)-\mu\left(S_{j_1}^{(k)}\right)\right)\left(\frac{|T_j\cap T^c|}{|T^c|}e^{\varepsilon}+\frac{|T_j^c\cap T^c|}{|T^c|}\right) \\ &= \frac{1}{\left(e^{\varepsilon}+1\right)}\left(\frac{1}{2}P\left(T\right)P\left(T^c\right)\varepsilon^2+O\left(\varepsilon^3\right)\right)\left\{\frac{|T_j\cap T^c|}{|T^c|}+\frac{|T_j^c\cap T^c|}{|T^c|}\right. \\ &\qquad \left.+\frac{|T_j\cap T|}{|T|}+\frac{|T_j^c\cap T|}{|T|}+O\left(\varepsilon\right)\right\} \\ &= \frac{1}{2}P\left(T\right)P\left(T^c\right)\varepsilon^2+O\left(\varepsilon^3\right), \end{aligned} \tag{60}$$

where we have used the facts that $T_{j_1}=T$, $T_{j_2}=T^c$, and

$$\begin{aligned} \mu\left(S_{j_1}^{(k)}\right) &= \frac{1}{2}P\left(T\right)P\left(T^c\right)\varepsilon^2+O\left(\varepsilon^3\right) \\ \mu\left(S_{j_2}^{(k)}\right) &= \frac{1}{2}P\left(T\right)P\left(T^c\right)\varepsilon^2+O\left(\varepsilon^3\right). \end{aligned} \tag{61}$$

Let $f(z)=|z-\frac{1}{2}|$, $g(z)=-z\log z-(1-z)\log(1-z)$, and $h(z)=z(1-z)$ for $0\leq z\leq 1$. On the one hand, g and h are monotonically increasing over $0\leq z\leq\frac{1}{2}$ and monotonically

decreasing over $\frac{1}{2}\leq z\leq 1$ but on the other hand, f is monotonically decreasing over $0\leq z\leq\frac{1}{2}$ and monotonically increasing over $\frac{1}{2}\leq z\leq 1$. Therefore,

$$\begin{aligned} T\in\arg\min_{A\subseteq\mathcal{X}}\left|P(A)-\frac{1}{2}\right| &\Leftrightarrow T\in\arg\max_{A\subseteq\mathcal{X}} -P(A)\log P(A)-P(A^c)\log P(A^c) \\ &\Leftrightarrow T\in\arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c). \end{aligned} \tag{62}$$

Since the set T was chosen so that it maximizes $P\left(T\right)P\left(T^c\right)$, we have that $P\left(T\right)P\left(T^c\right)\geq P\left(T_j\right)P\left(T_j^c\right)$ for all $j\in\{1,\ldots,2^k\}$. Assume, to begin with, that $j\neq\{j_1,j_2\}$. Then by the uniqueness of the maximizer assumption stated in the theorem, we have that $P\left(T\right)P\left(T^c\right)>P\left(T_j\right)P\left(T_j^c\right)$.

$$\left(S^T\alpha^*\right)_j-\mu_j=\frac{1}{2}\left(P\left(T\right)P\left(T^c\right)-P\left(T_j\right)P\left(T_j^c\right)\right)\varepsilon^2+O\left(\varepsilon^3\right),$$

and thus, there exists an ε^* that depends on P such that $\left(S^{(k)^T}\alpha^*\right)_j-\mu_j\geq 0$ for all $\varepsilon\leq\varepsilon^*$. If $j=\{j_1,j_2\}$, it is not hard to check that $\left(S^{(k)^T}\alpha^*\right)_j-\mu_j=0$ for all ε. This establishes the satisfiability of α^* for all $\varepsilon\leq\varepsilon^*$ which proves an upper bound on the primal problem (given in (57)). It remains to show that the upper bound can be indeed achieved via the binary mechanism. To this extent, recall that the binary mechanism is given by

$$Q(0|x)=\begin{cases}\frac{e^{\varepsilon}}{1+e^{\varepsilon}} & \text{if } x\in T,\\ \frac{1}{1+e^{\varepsilon}} & \text{if } x\notin T.\end{cases}\qquad Q(1|x)=\begin{cases}\frac{e^{\varepsilon}}{1+e^{\varepsilon}} & \text{if } x\notin T,\\ \frac{1}{1+e^{\varepsilon}} & \text{if } x\in T.\end{cases} \tag{63}$$

Computing the $I\left(X;Y\right)$ under (63), we get that

$$\begin{aligned} I\left(X;Y\right) &= \frac{1}{e^{\varepsilon}+1}\left\{P\left(T\right)e^{\varepsilon}\log\frac{e^{\varepsilon}}{P\left(T^c\right)+e^{\varepsilon}P\left(T\right)}+P\left(T^c\right)\log\frac{1}{P\left(T^c\right)+e^{\varepsilon}P\left(T\right)}\right\}+ \\ &\quad \frac{1}{e^{\varepsilon}+1}\left\{P\left(T^c\right)e^{\varepsilon}\log\frac{e^{\varepsilon}}{P\left(T\right)+e^{\varepsilon}P\left(T^c\right)}+P\left(T\right)\log\frac{1}{P\left(T\right)+e^{\varepsilon}P\left(T^c\right)}\right\}. \end{aligned} \tag{64}$$

Hence, the binary mechanism in (63) achieves the upper bound in (57). This proves the optimality of the binary mechanism for all $\varepsilon\leq\varepsilon^*$.

9.2 Proof of Theorem 13

We start by proving an upper bound on $\max_{Q\in\mathcal{D}_\varepsilon}I\left(X;Y\right)$ which is tight for $\varepsilon\leq 1$. Recall that by Theorem 4, we have that

$$\begin{aligned} \text{OPT}=\max_{Q\in\mathcal{D}_\varepsilon}I\left(X;Y\right)= \quad & \underset{\theta}{\text{maximize}} && \sum_{j=1}^{2^k}\mu_j\theta_j \\ & \text{subject to} && S^{(k)}\theta=\mathbb{1} \\ & && \theta\geq 0, \end{aligned}$$

where

$$\begin{aligned}\mu_j &= \mu\left(S_j^{(k)}\right)\\ &= \sum_{i\in[k]} P(x_i)\, S_{ij}^{(k)} \log\left(\frac{S_{ij}^{(k)}}{\sum_{i\in[k]} P(x_i)\, S_{ij}^{(k)}}\right)\\ &= P(T_j)\, e^{\varepsilon}\varepsilon - \left(P(T_j)\left(e^{\varepsilon}-1\right)+1\right)\log\left(P(T_j)\left(e^{\varepsilon}-1\right)+1\right), \end{aligned} \tag{65}$$

$T_j = \{i : S_{ij}^{(k)} = e^{\varepsilon}\}$, and $S^{(k)}$ is the $k \times 2^k$ staircase pattern matrix given in Definition 3.

Lemma 26 *For all distributions P and all ε, the following bound holds*

$$\mathrm{OPT} = \max_{Q\in\mathcal{D}_\varepsilon} I(X;Y) \le \left(\max_j \mu_j\right)\frac{k}{e^{\varepsilon}+k-1}.$$

The proof of this lemma is given in Section 9.3. In what follows, we will make the dependency of μ_j on $P(T_j)$ and ε explicit by writing $\mu_j(P(T_j),\varepsilon)$ for μ_j. From the proof of Theorem 12, we have that the partition set T defined in (19) is given by $T \in \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$. It is easy to check that the binary mechanism given in (20) achieves the following utility

$$\mathrm{BIN} = \frac{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)}{e^{\varepsilon}+1}.$$

Lemma 27 *For all distributions P and all $\varepsilon \le 1$, the following bound holds*

$$\frac{\max_j \mu_j}{\mu(P(T),\varepsilon)+\mu(P(T^c),\varepsilon)} \le 1.$$

The proof of the above lemma is given in Section 9.4. Combining the results of lemmas 26 and 27 we get that

$$\begin{aligned}\frac{\mathrm{OPT}}{\mathrm{BIN}} &\le \frac{\max_j \mu_j}{\mu(P(T),\varepsilon)+\mu(P(T^c),\varepsilon)}\frac{k}{e^{\varepsilon}+k-1}\left(e^{\varepsilon}+1\right)\\ &\le \frac{k}{e^{\varepsilon}+k-1}\left(e^{\varepsilon}+1\right)\\ &\le e^{\varepsilon}+1, \end{aligned} \tag{66}$$

for all $\varepsilon \le 1$. This concludes the proof.

9.3 Proof of Lemma 26

To begin with, since $S_1^{(k)} = \mathbb{1} = \frac{1}{e^{\varepsilon}} S_{2^k}^{(k)}$ and μ is homogenous, we have that $\theta_1\mu_1 + \theta_{2^k}\mu_{2^k} = \left(\frac{1}{e^{\varepsilon}}\theta_1 + \theta_{2^k}\right)\mu_{2^k}$. Therefore, the following two maximization problems are equivalent

$$\begin{array}{ll}\underset{\theta}{\text{maximize}} & \displaystyle\sum_{j=1}^{2^k}\mu_j\theta_j\\ \text{subject to} & S^{(k)}\theta = \mathbb{1}\\ & \theta \ge 0\end{array} \quad = \quad \begin{array}{ll}\underset{\theta}{\text{maximize}} & \displaystyle\sum_{j=1}^{2^k-1}\tilde{\mu}_j\theta_j\\ \text{subject to} & \tilde{S}^{(k)}\theta = \mathbb{1}\\ & \theta \ge 0,\end{array}$$

where $\tilde{\mu}_j = \mu_{j+1}$ and $\tilde{S}^{(k)}$ is obtained by deleting the first column of $S^{(k)}$. Moreover, using the fact that $\max_{j\in[2^k-1]}\tilde{\mu}_j \le \max_{j\in[2^k]}\mu_j$ and weak duality, we get that

$$\begin{array}{ll}\underset{\theta}{\text{maximize}} & \tilde{\mu}^T\theta\\ \text{subject to} & \tilde{S}^{(k)}\theta = \mathbb{1}\\ & \theta \ge 0\end{array} \quad \le \quad \left(\max_{j\in[2^k-1]}\tilde{\mu}_j\right)\begin{array}{ll}\underset{\theta}{\text{maximize}} & \mathbb{1}^T\theta\\ \text{subject to} & \tilde{S}^{(k)}\theta = \mathbb{1}\\ & \theta \ge 0\end{array}$$

$$\le \left(\max_{j\in[2^k]}\mu_j\right)\begin{array}{ll}\underset{\alpha}{\text{minimize}} & \mathbb{1}^T\alpha\\ \text{subject to} & \tilde{S}^{(k)^T}\alpha \ge \mathbb{1}.\end{array} \tag{67}$$

Consider the following choice of dual variable $\alpha_i^* = \frac{1}{e^{\varepsilon}+k-1}$. We claim that α^* is satisfiable. This can be easily verified by noting that

$$\left(\tilde{S}^{(k)^T}\alpha^*\right)_j = \tilde{S}_j^{(k)^T}\alpha^* = \frac{|T_j|e^{\varepsilon} + (k-|T_j|)}{e^{\varepsilon}+k-1} = \frac{|T_j|(e^{\varepsilon}-1)+k}{e^{\varepsilon}+k-1} \ge 1$$

where the last inequality holds since $|T_j| \ge 1$ (this is true because we have deleted the first column of $S^{(k)}$). Therefore, $\mathrm{OPT} \le (\max_j \mu_j)\mathbb{1}^T\alpha^* = (\max_j \mu_j)\frac{k}{e^{\varepsilon}+k-1}$ which was to be shown.

9.4 Proof of Lemma 27

Let $\mu(z,\varepsilon)$ be the function obtained by replacing $P(T_j)$ by the continuous variable $z \in [0,1]$ in $\mu_j(P(T_j),\varepsilon)$. Taking the derivative of $\mu(z,\varepsilon)$ with respect to z we get

$$\mu'(z,\varepsilon) = e^{\varepsilon}\varepsilon - \left(e^{\varepsilon}-1\right) - \left(e^{\varepsilon}-1\right)\log\left(z(e^{\varepsilon}-1)+1\right).$$

Observe that $\mu'(z,\varepsilon) > 0$ for all

$$z < z^*(\varepsilon) = \frac{1}{e^{\varepsilon}-1}\left(e^{\left\{\frac{e^{\varepsilon}\varepsilon}{e^{\varepsilon}-1}-1\right\}}-1\right),$$

$\mu'(z,\varepsilon) < 0$ for all $z > z^*(\varepsilon)$, and $\mu'(z,\varepsilon) = 0$ for $z = z^*(\varepsilon)$. Combining this with the fact that $\mu(0,\varepsilon) = \mu(1,\varepsilon) = 0$ we get that $\mu(z,\varepsilon) \ge 0$ for all $z \in [0,1]$ and for any fixed ε, $\mu(z,\varepsilon)$ is maximized at $z^*(\varepsilon)$.

Set $x^* \in \arg\max_{x\in\mathcal{X}} P(x)$ and fix an $\varepsilon \le 1$. We will treat the following three cases separately.

Case 1: $P(x^*) \in [1-z^*(\varepsilon), 1]$.

Claim 4 *Let $T = \{x^*\}$. Then $\{T, T^c\} = \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$ and $\max_{A\subseteq\mathcal{X}}\mu(P(A),\varepsilon) = \max\left(\mu(P(T),\varepsilon), \mu(P(T^c),\varepsilon)\right)$.*

Proof Observe that $z^*(\varepsilon) \le \frac{1}{2}$ for all ε and $T^c = \mathcal{X}\setminus\{x^*\}$. The function $f(z) = z(1-z)$ decreases over the range $[\frac{1}{2}, 1] \supseteq [1-z^*(\varepsilon), 1]$. Thus, for all $A \supset T$, $P(T)P(T^c) >$

$P(A)P(A^c)$ because $P(T) \geq 1 - z^*(\varepsilon)$. This proves that $T \in \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$ and for all $A \supset T$, $A \notin \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$. Using a similar approach, we can show that $T^c \in \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$ and for all $A \subset T^c$, $A \notin \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$. Therefore, $\{T, T^c\} = \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$. This proves the first part of the claim. The function $\mu(z,\varepsilon)$ increases over the range $[0, z^*(\varepsilon)]$. Thus, for all $A \subseteq T^c$, $\mu(P(A),\varepsilon) \leq \mu(P(T^c),\varepsilon)$ because $P(T^c) \leq z^*(\varepsilon)$. On the other hand, note that $\mu(z,\varepsilon)$ decreases over the range $[z^*(\varepsilon), 1]$ which includes the range $[1 - z^*(\varepsilon), 1]$. Thus, for all A such that $A \supseteq T$, $\mu(P(A),\varepsilon) \leq \mu(P(T),\varepsilon)$ because $P(T) \geq 1 - z^*(\varepsilon)$. This proves that $\max(\mu(P(T),\varepsilon), \mu(P(T^c),\varepsilon)) = \max_{A\subseteq\mathcal{X}} \mu(P(A),\varepsilon)$. ■

Using the above claim, we can conclude that the partition set T defined in (19) is equal to $\{x^*\}$ and

$$\begin{aligned}\frac{\max_j \mu_j}{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)} &= \frac{\max_{A\subseteq\mathcal{X}} \mu(P(A),\varepsilon)}{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)} \\ &\leq \frac{\max_{A\subseteq\mathcal{X}} \mu(P(A),\varepsilon)}{\max(\mu(P(T),\varepsilon), \mu(P(T^c),\varepsilon))} \\ &= 1. \end{aligned} \tag{68}$$

Case 2: $P(x^*) \in [\frac{1}{2}, 1 - z^*(\varepsilon)]$. Using the first part of the proof of Claim 4, we can show that if $T = \{x^*\}$, then $\{T, T^c\} = \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)$. Therefore, the partition set T defined in (19) is equal to $\{x^*\}$ and

$$\begin{aligned}\frac{\max_j \mu_j}{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)} &= \frac{\max_{A\subseteq\mathcal{X}} \mu(P(A),\varepsilon)}{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)} \\ &\leq \frac{\mu(z^*(\varepsilon),\varepsilon)}{\mu(P(x^*),\varepsilon) + \mu(1 - P(x^*),\varepsilon)} \\ &\leq 1. \end{aligned} \tag{69}$$

Case 3: $P(x^*) \in [0, \frac{1}{2}]$.

Claim 5 *There exists a set $A \subset \mathcal{X}$ such that $\frac{1}{2} - P(x^*) \leq P(A) \leq \frac{1}{2}$.*

Proof Without loss of generality, assume that the sequence $P(x_i)$, $i \in [k]$, is sorted in increasing order. Let $l^* = \min\{l : \sum_{i=1}^{l} P(x_i) \geq \frac{1}{2}\}$. From the definition of l^*, $P(\{x_1, \ldots, x_{l^*-1}\}) < \frac{1}{2}$ and $P(\{x_1, \ldots, x_{l^*}\}) \geq \frac{1}{2}$. Further,

$$P(\{x_1, \ldots, x_{l^*-1}\}) = P(\{x_1, \ldots, x_{l^*}\}) - P(x_{l^*})$$

and since $x^* \in \arg\max_{x\in\mathcal{X}} P(x)$, $P(x_{l^*}) \leq P(x^*)$. Therefore, if $A = \{x_1, \ldots, x_{l^*-1}\}$, then $\frac{1}{2} - P(x^*) \leq P(A) \leq \frac{1}{2}$. ■

Let $P(T) = \min\{P(B) : B \in \arg\max_{A\subseteq\mathcal{X}} P(A)P(A^c)\}$. We claim that $\frac{1}{4} \leq P(T) \leq \frac{1}{2}$. The upper bound on $P(T)$ follows immediately from its definition. To prove the lower bound on $P(T)$, consider the set A given in Claim 5 and observe that

$$\begin{aligned} P(T) &\geq \max(P(x^*), P(A)) \\ &\geq \max(P(x^*), \frac{1}{2} - P(x^*)) \\ &\geq \frac{1}{4}. \end{aligned} \tag{70}$$

All the inequalities follow from Claim 5 and the fact that $P(x^*) \in [0, \frac{1}{2}]$.

Since $\frac{1}{4} \leq P(T) \leq \frac{1}{2}$, we have that $\frac{1}{2} \leq P(T^c) \leq \frac{3}{4}$. Moreover, the function $\mu(z,\varepsilon)$ decreases over the range $[z^*(\varepsilon), 1] \supset [\frac{1}{2}, \frac{3}{4}]$ and increases over the range $[\frac{1}{4}, z^*(\varepsilon)]$. Therefore, $\mu(P(T^c),\varepsilon) \geq \mu(\frac{3}{4},\varepsilon)$ and $\mu(P(T),\varepsilon) \geq \min\left(\mu(\frac{1}{2},\varepsilon), \mu(\frac{1}{4},\varepsilon)\right)$. Putting it all together, we have that

$$\begin{aligned}\frac{\max_j \mu_j}{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)} &= \frac{\max_{A\subseteq\mathcal{X}} \mu(P(A),\varepsilon)}{\mu(P(T),\varepsilon) + \mu(P(T^c),\varepsilon)} \\ &\leq \frac{\mu(z^*(\varepsilon),\varepsilon)}{\min\left(\mu(\frac{1}{2},\varepsilon), \mu(\frac{1}{4},\varepsilon)\right) + \mu(\frac{3}{4},\varepsilon)} \\ &\leq 1. \end{aligned} \tag{71}$$

9.5 Proof of Theorem 14

By Theorem 4, we have that

$$\max_{Q\in\mathcal{D}_\varepsilon} I(X;Y) = \begin{aligned} &\underset{\theta}{\text{maximize}} && \mu^T\theta \\ &\text{subject to} && S^{(k)}\theta = \mathbb{1} \\ &&& \theta \geq 0, \end{aligned} \tag{72}$$

where $\mu_j = \mu\left(S_j^{(k)}\right) = \sum_{i\in[k]} P(x_i) S_{ij}^{(k)} \log\left(\frac{S_{ij}^{(k)}}{\sum_{i\in[k]} P(x_i) S_{ij}^{(k)}}\right)$ for $j \in \{1, \ldots, 2^k\}$ and $S^{(k)}$ is the $k \times 2^k$ staircase pattern matrix given in Definition 3. The polytope given by $S^{(k)}\theta = \mathbb{1}$ and $\theta \geq 0$ is a closed and bounded one. Thus, there is no duality gap and solving the above linear program is equivalent to solving its dual

$$\begin{aligned} &\underset{\alpha}{\text{minimize}} && \mathbb{1}^T\alpha \\ &\text{subject to} && {S^{(k)}}^T\alpha \geq \mu. \end{aligned} \tag{73}$$

Note that any satisfiable solution α^* to (73) provides an upper bound to (72) since $\max \mu^T\theta = \min \mathbb{1}^T\alpha \leq \mathbb{1}^T\alpha^*$. Let $T_j = \{x_i : S_{ij}^{(k)} = e^\varepsilon\}$ and set $j_i = \{j : T_j = i\}$ for $i \in \{1, \ldots, k\}$. Consider the following choice of dual variable

$$\alpha_i^* = \frac{1}{(e^\varepsilon - 1)(e^\varepsilon + k - 1)} \left\{ (e^\varepsilon + k - 2)\, \mu\left(S_{j_i}^{(k)}\right) - \sum_{l\in[k], l\neq i} \mu\left(S_{j_l}^{(k)}\right) \right\},$$

for $i \in \{1, \ldots, k\}$. Observe that since $T_{j_i} = i$ we have that $P(T_{j_i}) = P(x_i)$ and since

$$\mu_j = P(T_j)\, e^\varepsilon \log\frac{e^\varepsilon}{P(T_j{}^c) + e^\varepsilon P(T_j)} + P(T_j{}^c) \log\frac{1}{P(T_j{}^c) + e^\varepsilon P(T_j)}, \tag{74}$$

we have that

$$\begin{aligned}
\mathbb{1}^T\alpha^* &= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\sum_{i\in[k]}\left\{(e^\varepsilon+k-2)\,\mu\left(S_{ji}^{(k)}\right)-\sum_{l\in[k],l\neq i}\mu\left(S_{jl}^{(k)}\right)\right\}\\
&= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{(e^\varepsilon+k-2)\sum_{i\in[k]}\mu\left(S_{ji}^{(k)}\right)-\sum_{i\in[k]}\sum_{l\in[k],l\neq i}\mu\left(S_{jl}^{(k)}\right)\right\}\\
&= \frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{(e^\varepsilon+k-2)\sum_{i\in[k]}\mu\left(S_{ji}^{(k)}\right)-(k-1)\sum_{i\in[k]}\mu\left(S_{ji}^{(k)}\right)\right\}\\
&= \frac{1}{(e^\varepsilon+k-1)}\sum_{i\in[k]}\mu\left(S_{ji}^{(k)}\right)\\
&= \frac{1}{(e^\varepsilon+k-1)}\sum_{i\in[k]}\left\{P(x_i)\,e^\varepsilon\log\frac{e^\varepsilon}{P(x_i)(e^\varepsilon-1)+1}\right.\\
&\qquad\left.+(1-P(x_i))\log\frac{1}{P(x_i)(e^\varepsilon-1)+1}\right\}. \quad (75)
\end{aligned}$$

We claim that α^* is a feasible dual variable for sufficiently large ε. In order to prove that α^* is a feasible dual variable, we show that $\left(S^{(k)^T}\alpha^*\right)_j-\mu_j\geq 0$ for all $j\in\{1,\ldots,2^k\}$ and all $\varepsilon\geq\varepsilon^*$, where ε^* is a positive quantity that depends on P. Using the fact that

$$\log(a+e^\varepsilon b)=\varepsilon+\log b+O\left(e^{-\varepsilon}\right),$$

for large ε, we get that

$$\begin{aligned}
\mu_j &= P(T_j)\,e^\varepsilon\log\frac{e^\varepsilon}{P(T_j{}^c)+e^\varepsilon P(T_j)}+P(T_j{}^c)\log\frac{1}{P(T_j{}^c)+e^\varepsilon P(T_j)}\\
&= P(T_j)\,e^\varepsilon\varepsilon-(P(T_j)(e^\varepsilon-1)+1)\log(P(T_j)(e^\varepsilon-1)+1)\\
&= P(T_j)\,e^\varepsilon\varepsilon-(P(T_j)(e^\varepsilon-1)+1)\left(\varepsilon+\log P(T_j)+O\left(e^{-\varepsilon}\right)\right)\\
&= -\left(P(T_j)\log P(T_j)\right)e^\varepsilon+O(\varepsilon). \quad (76)
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\left(S^{(k)^T}\alpha^*\right)_j &= S_j^{(k)^T}\alpha^*\\
&= e^\varepsilon\sum_{i\in T_j}\alpha_i^*+\sum_{i\in T_j^c}\alpha_i^*\\
&= -\frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{\sum_{i\in[k]}S_{ij}^{(k)}(e^\varepsilon+k-2)\left(P(x_i)\log P(x_i)\,e^\varepsilon\right.\right.\\
&\qquad\left.\left.+O(\varepsilon)\right)\right\}\\
&\quad+\frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left\{\sum_{i\in[k]}\sum_{l\in[k],l\neq i}S_{ij}^{(k)}\left(\left(P(x_l)\log P(x_l)\right)e^\varepsilon+O(\varepsilon)\right)\right\}\\
&= -\frac{1}{(e^\varepsilon-1)(e^\varepsilon+k-1)}\left(\left(\sum_{i\in T_j}P(x_i)\log P(x_i)\right)e^{3\varepsilon}+O\left(e^{2\varepsilon}\varepsilon\right)\right)\\
&= -\left(\sum_{i\in T_j}P(x_i)\log P(x_i)\right)e^\varepsilon+O(\varepsilon). \quad (77)
\end{aligned}$$

Assume, to begin with, that $j\neq\{j_1,j_2,...,j_k\}$. Then

$$\left(S^{(k)^T}\alpha^*\right)_j-\mu_j=\left(P(T_j)\log P(T_j)-\sum_{i\in T_j}P(x_i)\log P(x_i)\right)e^\varepsilon+O(\varepsilon).$$

Notice that for $j\neq\{j_1,j_2,...,j_k\}$, $P(T_j)\log P(T_j)>\sum_{i\in T_j}P(x_i)\log P(x_i)$. Therefore, there exists an $\varepsilon^*>0$ such that $\left(S^{(k)^T}\alpha^*\right)_j-\mu_j\geq 0$ for all $\varepsilon\geq\varepsilon^*$. If $j\in\{j_1,j_2,...,j_k\}$, it is not hard to check that $\left(S^{(k)^T}\alpha^*\right)_j-\mu_j=0$ for all ε. This establishes the satisfiability of α^* for all $\varepsilon\geq\varepsilon^*$. It remains to show that the upper bound can be indeed achieved via the randomized response mechanism. To this extent, recall that the randomized response is given by

$$Q(y|x)=\begin{cases}\dfrac{e^\varepsilon}{|\mathcal{X}|-1+e^\varepsilon} & \text{if } y=x\,,\\ \dfrac{1}{|\mathcal{X}|-1+e^\varepsilon} & \text{if } y\neq x\,.\end{cases} \quad (78)$$

Computing the $I(X;Y)$ under (78), we get that

$$\begin{aligned}
I(X;Y) &= \frac{1}{e^\varepsilon+k-1}\sum_{i\in[k]}\left\{P(x_i)\,e^\varepsilon\log\frac{e^\varepsilon}{P(x_i)(e^\varepsilon-1)+1}\right.\\
&\qquad\left.+(1-P(x_i))\log\frac{1}{P(x_i)(e^\varepsilon-1)+1}\right\}. \quad (79)
\end{aligned}$$

Hence, the randomized response mechanism achieves the upper bound (75). This proves the optimality of the randomized response for all $\varepsilon \geq \varepsilon^*$.

10. Proof of Proposition 17

Let $U(Q)$ be a utility mechanism of the form $U(Q) = \sum_{\mathcal{Y}} \mu(Q_y)$, where μ is a sublinear function. Consider a stochastic mapping W of dimensions $\ell \times m$ and let QW be the stochastic mapping obtained by first applying Q to $X \in \mathcal{X}$ to obtain $Y \in \mathcal{Y}$ and then applying W to Y to obtain $Z \in \mathcal{Z}$.

$$\begin{aligned} U(QW) &= \sum_{\mathcal{Z}} \mu\left((QW)_z\right) \\ &= \sum_{\mathcal{Z}} \mu\left(\sum_{\mathcal{Y}} Q_y W_{y,z}\right) \\ &\leq \sum_{\mathcal{Y},\mathcal{Z}} W_{y,z} \mu\left(Q_y\right) \\ &= \sum_{\mathcal{Y}} \mu(Q_y) \\ &= U(Q), \end{aligned} \tag{80}$$

where the inequality follows from sublinearity and the second to last equality follows from the row stochastic property of W. Therefore, $U(Q)$ obeys the data processing inequality.

Acknowledgments

This research is supported in part by NSF CISE award CCF-1422278, NSF SaTC award CNS-1527754, NSF CMMI award MES-1450848 and NSF ENG award ECCS-1232257.

References

A. Acquisti. Privacy in electronic commerce and the economics of immediate gratification. In *Proceedings of the 5th ACM conference on Electronic commerce*, pages 21–29. ACM, 2004.

A. Acquisti and J. Grossklags. What can behavioral economics teach us about privacy. *Digital Privacy*, page 329, 2007.

R. F. Barber and J. C. Duchi. Privacy and statistical risk: Formalisms and minimax bounds. *arXiv preprint arXiv:1412.4451*, 2014.

A. Beimel, K. Nissim, and E. Omri. Distributed private data analysis: Simultaneously solving how and what. In *Advances in Cryptology–CRYPTO 2008*, pages 451–468. Springer, 2008.

D. Blackwell. Equivalent comparisons of experiments. *The Annals of Mathematical Statistics*, 24(2):265–272, 1953.

J. Blocki, A. Blum, A. Datta, and O. Sheffet. The Johnson-Lindenstrauss transform itself preserves differential privacy. In *Foundations of Computer Science, 2012 IEEE 53rd Annual Symposium on*, pages 410–419. IEEE, 2012.

K. Chatzikokolakis, T. Chothia, and A. Guha. Statistical measurement of information leakage. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 390–404. Springer, 2010.

K. Chaudhuri and D. Hsu. Convergence rates for differentially private statistical estimation. *arXiv preprint arXiv:1206.6395*, 2012.

K. Chaudhuri and C. Monteleoni. Privacy-preserving logistic regression. In *NIPS*, volume 8, pages 289–296, 2008.

K. Chaudhuri, A. D. Sarwate, and K. Sinha. Near-optimal differentially private principal components. In *NIPS*, pages 998–1006, 2012.

T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

A. De. Lower bounds in differential privacy. In *Theory of Cryptography*, pages 321–338. Springer, 2012.

J. C Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *Foundations of Computer Science, 2013 IEEE 54th Annual Symposium on*, pages 429–438. IEEE, 2013.

C. Dwork. Differential privacy. In *Automata, languages and programming*, pages 1–12. Springer, 2006.

C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 371–380. ACM, 2009.

C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology-EUROCRYPT 2006*, pages 486–503. Springer, 2006a.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284. Springer, 2006b.

Q. Geng and P. Viswanath. The optimal mechanism in differential privacy. *arXiv preprint arXiv:1212.1186*, 2012.

Q. Geng and P. Viswanath. The optimal mechanism in (ϵ,δ)-differential privacy. *arXiv preprint arXiv:1305.1330*, 2013a.

Q. Geng and P. Viswanath. The optimal mechanism in differential privacy: Multidimensional setting. *arXiv preprint arXiv:1312.0655*, 2013b.

A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41(6):1673–1693, 2012.

M. Hardt and A. Roth. Beating randomized response on incoherent matrices. In *Proceedings of the 44th symposium on Theory of Computing*, pages 1255–1268. ACM, 2012.

M. Hardt and G. N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *Foundations of Computer Science, 2010 51st Annual IEEE Symposium on*, pages 61–70. IEEE, 2010.

M. Hardt and K. Talwar. On the geometry of differential privacy. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 705–714. ACM, 2010.

M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. In *NIPS*, pages 2348–2356, 2012.

P. Kairouz, S. Oh, and P. Viswanath. The composition theorem for differential privacy. *arXiv preprint arXiv:1311.0776*, 2013.

P. Kairouz, S. Oh, and P. Viswanath. Extremal mechanisms for local differential privacy. *arXiv preprint arXiv:1407.1338*, 2014a.

P. Kairouz, S. Oh, and P. Viswanath. Differentially private multi-party computation: Optimality of non-interactive randomized response. *arXiv preprint arXiv:1407.1546*, 2014b.

M. Kapralov and K. Talwar. On differentially private low rank approximation. In *SODA*, volume 5, page 1. SIAM, 2013.

J. Lei. Differentially private m-estimators. In *NIPS*, pages 361–369, 2011.

F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. 48th Annual IEEE Symposium on*, pages 94–103. IEEE, 2007.

L. Sankar, S. R. Rajagopalan, and H. V. Poor. Utility-privacy tradeoffs in databases: An information-theoretic approach. *Information Forensics and Security, IEEE Transactions on*, 8(6):838–852, 2013.

A. D. Sarwate and L. Sankar. A rate-disortion perspective on local differential privacy. In *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, pages 903–908. IEEE, 2014.

A. B. Tsybakov and V. Zaiats. *Introduction to nonparametric estimation*, volume 11. Springer, 2009.

W. Wang, L. Ying, and J. Zhang. On the relation between identifiability, differential privacy and mutual-information privacy. *arXiv preprint arXiv:1402.3757*, 2014a.

Y. Wang, Z. Huang, S. Mitra, and G.E. Dullerud. Entropy-minimizing mechanism for differential privacy of discrete-time linear feedback systems. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 2130–2135, Dec 2014b. doi: 10.1109/CDC.2014.7039713.

S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

Journal of Machine Learning Research 17 (2016) 1-40 Submitted 7/14; Revised 6/15; Published 4/16

Loss Minimization and Parameter Estimation with Heavy Tails

Daniel Hsu DJHSU@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
New York, NY 10027, USA

Sivan Sabato SABATOS@CS.BGU.AC.IL
Department of Computer Science
Ben-Gurion University of the Negev
Beer-Sheva 8410501, Israel

Editor: David Dunson

Abstract

This work studies applications and generalizations of a simple estimation technique that provides exponential concentration under heavy-tailed distributions, assuming only bounded low-order moments. We show that the technique can be used for approximate minimization of smooth and strongly convex losses, and specifically for least squares linear regression. For instance, our d-dimensional estimator requires just $\tilde{O}(d \log(1/\delta))$ random samples to obtain a constant factor approximation to the optimal least squares loss with probability $1-\delta$, without requiring the covariates or noise to be bounded or subgaussian. We provide further applications to sparse linear regression and low-rank covariance matrix estimation with similar allowances on the noise and covariate distributions. The core technique is a generalization of the median-of-means estimator to arbitrary metric spaces.

Keywords: Heavy-tailed distributions, unbounded losses, linear regression, least squares

1. Introduction

The minimax principle in statistical estimation prescribes procedures (*i.e.*, estimators) that minimize the worst-case risk over a large class of distributions generating the data. For a given loss function, the risk is the expectation of the loss of the estimator, where the expectation is taken over the data examined by the estimator. For example, for a large class of loss functions including squared loss, the empirical mean estimator minimizes the worst-case risk over the class of Gaussian distributions with known variance (Wolfowitz, 1950). In fact, Gaussian distributions with the specified variance are essentially the worst-case family of distributions for squared loss, at least up to constants (see, *e.g.*, Catoni, 2012, Proposition 6.1).

In this work, we are interested in estimators whose deviations from expected behavior are controlled with very high probability over the random draw of the data examined by the estimator. Deviations of the behavior of the estimator from its expected behavior are worrisome especially when data come from unbounded and/or heavy-tail distributions, where only very low order moments may be finite. For example, the Pareto distributions with shape parameter $\alpha > 0$ are unbounded and have finite moments only up to orders $< \alpha$; these distributions are commonly associated with the modeling of extreme events that manifest in data. Bounds on the expected behavior of an estimator are insufficient in these cases, since the high-probability guarantees that may be derived from such bounds (say, using Markov's inequality) are rather weak. For example, if the risk (*i.e.*, expected loss) of an estimator is bounded by ϵ, then all that we may derive from Markov's inequality is that the loss is no more than ϵ/δ with probability at least $1-\delta$. For small values of $\delta \in (0,1)$, the guarantee is not very reassuring, but it may be all one can hope for in these extreme scenarios—see Remark 7 in Section 3.1 for an example where this is tight. Much of the work in statistical learning theory is also primarily concerned with such high probability guarantees, but the bulk of the work makes either boundedness or subgaussian tail assumptions that severely limit the applicability of the results even in settings as simple as linear regression (see, *e.g.*, Srebro et al., 2010; Shamir, 2014).

Recently, it has been shown that it is possible to improve on methods which are optimal for expected behavior but suboptimal when high-probability deviations are concerned (Audibert and Catoni, 2011; Catoni, 2012; Brownlees et al., 2014). These improvements, which are important when dealing with heavy-tailed distributions, suggest that new techniques (*e.g.*, beyond empirical risk minimization) may be able to remove the reliance on boundedness or control of high-order moments. Bubeck et al. (2013) show how a more robust mean estimator can be used for solving the stochastic multi-armed bandit problem under heavy-tailed distributions.

This work applies and generalizes a technique for controlling large deviations from the expected behavior with high probability, assuming only bounded low-order moments such as variances. We show that the technique is applicable to minimization of smooth and strongly convex losses, and derive specific loss bounds for least squares linear regression, which match existing rates, but without requiring the noise or covariates to be bounded or subgaussian. This contrasts with several recent works (Srebro et al., 2010; Hsu et al., 2014; Shamir, 2014) concerned with (possibly regularized) empirical risk minimizers that require such assumptions. It is notable that in finite dimensions, our result implies that a constant factor approximation to the optimal loss can be achieved with a sample size that is independent of the size of the optimal loss. This improves over the recent work of Mahdavi and Jin (2013), which has a logarithmic dependence on the optimal loss, as well as a suboptimal dependence on specific problem parameters (namely condition numbers). We also provide a new generalization of the basic technique for general metric spaces, which we apply to least squares linear regression with heavy tail covariate and noise distributions, yielding an improvement over the computationally expensive procedure of Audibert and Catoni (2011).

The basic technique, found in the textbook of Nemirovsky and Yudin (1983, p. 243), is very simple, and can be viewed as a generalization of the median-of-means estimator used by Alon et al. (1999) and many others. The idea is to repeat an estimate several times, by splitting the sample into several groups, and then selecting a single estimator out of the resulting list of candidates. If an estimator from one group is good with noticeably better-than-fair chance, then the selected estimator will be good with probability exponentially close to one. This is remininscant of techniques from *robust statistics* (Huber, 1981),

although our aim is expressly different in that our aim is good performance on the same probability distribution generating the data, rather than an uncontaminated or otherwise better behaved distribution. Our new technique can be cast as a simple selection problem in general metric spaces that generalizes the scalar median.

We demonstrate the versatility of our technique by giving further examples in sparse linear regression (Tibshirani, 1996) under heavy-tailed noise and low-rank covariance covariance matrix approximation (Koltchinskii et al., 2011) under heavy-tailed covariate distributions. We also show that for prediction problems where there may not be a reasonable metric on the predictors, one can achieve similar high-probability guarantees by using median aggregation in the output space.

The initial version of this article (Hsu and Sabato, 2013, 2014) appeared concurrently with the simultaneous and independent work of Minsker (2013), which develops a different generalization of the median-of-means estimator for Banach and Hilbert spaces. We provide a new analysis and comparison of this technique to ours in Section 7. We have also since become aware of the earlier work by Lerasle and Oliveira (2011), which applies the median-of-means technique to empirical risks in various settings much like the way we do in Algorithm 3, although our metric formulation is more general. Finally, the recent work of Brownlees et al. (2014) vastly generalizes the techniques of Catoni (2012) to apply to much more general settings, although they retain some of the same deficiencies (such as the need to know the noise variance for the optimal bound for least squares regression), and hence their results are not directly comparable to ours.

2. Overview of Main Results

This section gives an overview of the main results.

2.1 Preliminaries

Let $[n] := \{1, 2, \ldots, n\}$ for any natural number $n \in \mathbb{N}$. Let $\mathbb{1}\{P\}$ take value 1 if the predicate P is true, and 0 otherwise. Assume an example space $\mathcal{Z}$, and a distribution $\mathcal{D}$ over the space. Further assume a space of predictors or estimators $\mathbb{X}$. We consider learning or estimation algorithms that accept as input an i.i.d. sample of size n drawn from $\mathcal{D}$ and a confidence parameter $\delta \in (0, 1)$, and return an estimator (or predictor) $\hat{\boldsymbol{w}} \in \mathbb{X}$. For a (pseudo) metric ρ on $\mathbb{X}$, let $B_\rho(\boldsymbol{w}_0, r) := \{\boldsymbol{w} \in \mathbb{X} : \rho(\boldsymbol{w}_0, \boldsymbol{w}) \leq r\}$ denote the ball of radius r around $\boldsymbol{w}_0$.

We assume a loss function $\ell : \mathcal{Z} \times \mathbb{X} \to \mathbb{R}_+$ that assigns a non-negative number to a pair of an example from $\mathcal{Z}$ and a predictor from $\mathbb{X}$, and consider the task of finding a predictor that has a small loss in expectation over the distribution of data points, based on an input sample of n examples drawn independently from $\mathcal{D}$. The expected loss of a predictor $\boldsymbol{w}$ on the distribution is denoted $L(\boldsymbol{w}) = \mathbb{E}_{Z \sim D}(\ell(Z, \boldsymbol{w}))$. Let $L_\star := \inf_{\boldsymbol{w}} L(\boldsymbol{w})$. Our goal is to find $\hat{\boldsymbol{w}}$ such that $L(\hat{\boldsymbol{w}})$ is close to $L_\star$.

In this work, we are interested in performance guarantees that hold with high probability over the random draw of the input sample and any internal randomization used by the estimation algorithm. Thus, for a given allowed probability of failure $\delta \in (0, 1)$, we study excess loss $L(\hat{\boldsymbol{w}}) - L_\star$ achieved by the predictor $\hat{\boldsymbol{w}} \equiv \hat{\boldsymbol{w}}(\delta)$ returned by the algorithm on a $1 - \delta$ probability subset of the sample space. Ideally, the excess loss only depends sub-

logarithmically on $1/\delta$, which is the dependence achieved when the distribution of the excess loss has exponentially decreasing tails. Note that we assume that the value of δ is provided as input to the estimation algorithm, and only demand the probabilistic guarantee for this given value of δ. Therefore, strictly speaking, the excess loss need not exhibit exponential concentration. Nevertheless, in this article, we shall say that an estimation algorithm achieves exponential concentration whenever it guarantees, on input δ, an excess loss that grows only as $\log(1/\delta)$.

2.2 Robust Distance Approximation

Consider an estimation problem, where the goal is to estimate an unknown parameter of the distribution, using a random i.i.d. sample from that distribution. We show throughout this work that for many estimation problems, if the sample is split into non-overlapping subsamples, and estimators are obtained independently from each subsample, then with high probability, this generates a set of estimators such that some fraction of them are close, under a meaningful metric, to the true, unknown value of the estimated parameter. Importantly, this can be guaranteed in many cases even under under heavy-tailed distributions.

Having obtained a set of estimators, a fraction of which are close to the estimated parameter, the goal is now to find a single good estimator based on this set. This goal is captured by the following general problem, which we term *Robust Distance Approximation*. A Robust Distance Approximation procedure is given a set of points in a metric space and returns a single point from the space. This single point should satisfy the following condition: If there is an element in the metric space that a certain fraction of the points in the set are close to, then the output point should also be close to the same element. Formally, let $(\mathbb{X}, \rho)$ be a metric space. Let $W \subseteq \mathbb{X}$ be a (multi)set of size k and let $w_\star$ be a distinguished element in $\mathbb{X}$. For $\alpha \in (0, \frac{1}{2})$ and $w \in \mathbb{X}$, denote by $\Delta_W(w, \alpha)$ the minimal number r such that $|\{v \in W \mid \rho(w, v) \leq r\}| > k(\frac{1}{2} + \alpha)$. We often omit the subscript W and write simply Δ when W is known.

We define the following problem:

Definition 1 (Robust Distance Approximation) *Fix $\alpha \in (0, \frac{1}{2})$. Given W and $(\mathbb{X}, \rho)$ as input, return $y \in \mathbb{X}$ such that $\rho(y, w_\star) \leq C_\alpha \cdot \Delta_W(w_\star, \alpha)$, for some constant $C_\alpha \geq 0$. C_α is the* approximation factor *of the procedure.*

In some cases, learning with heavy-tailed distributions requires using a metric that depends on the distribution. Then, the Robust Distance Estimation procedure has access only to noisy measurements of distances in the metric space, and is required to succeed with high probability. In Section 3 we formalize these notions, and provide simple implementations of Robust Distance Approximation for general metric spaces, with and without direct access to the metric. For the case of direct access to the metric our formulation is similar to that of Nemirovsky and Yudin (1983).

2.3 Convex Loss Minimization

The general approach to estimation described above has many applications. We give here the general form of our main results for applications, and defer the technical definitions and

results to the relevant sections. Detailed discussion of related work for each application is also provided in the appropriate sections.

First, we consider smooth and convex losses. We assume that the parameter space $\mathbb{X}$ is a Banach space with a norm $\|\cdot\|$ and a dual norm $\|\cdot\|_*$. We prove the following result:[1]

Theorem 2 *There exists an algorithm that accepts as input an i.i.d. sample of size n drawn from $\mathcal{D}$ and a confidence parameter $\delta \in (0,1)$, and returns $\hat{\boldsymbol{w}} \in \mathbb{X}$, such that if the following conditions hold:*

- *the dual norm $\|\cdot\|_*$ is γ-smooth;*
- *there exists $\alpha > 0$ and sample size n_α such that, with probability at least $1/2$, the empirical loss $\boldsymbol{w} \mapsto \hat{L}(\boldsymbol{w})$ is α-strongly convex with respect to $\|\cdot\|$ whenever the sample is of size at least n_α;*
- *$n \geq C \log(1/\delta) \cdot n_\alpha$ for some universal constant $C > 0$;*
- *$\boldsymbol{w} \mapsto \ell(z, \boldsymbol{w})$ is β-smooth with respect to $\|\cdot\|$ for all $z \in \mathcal{Z}$;*
- *$\boldsymbol{w} \mapsto L(\boldsymbol{w})$ is $\bar{\beta}$-smooth with respect to $\|\cdot\|$;*

then with probability at least $1-\delta$, for another universal constant $C' > 0$,

$$L(\hat{\boldsymbol{w}}) \leq \left(1 + \frac{C'\beta\bar{\beta}\gamma\lceil\log(1/\delta)\rceil}{n\alpha^2}\right) L_\star.$$

This gives a constant approximation of the optimal loss with a number of samples that does not depend on the value of the optimal loss. The full results for smooth convex losses are provided in Section 4. Theorem 2 is stated in full as Corollary 16, and we further provide a result with more relaxed smoothness requirements. As apparent in the result, the only requirements on the distribution are those that are implied by the strong convexity and smoothness parameters. This allows support for fairly general heavy-tailed distributions, as we show below.

2.4 Least Squares Linear Regression

A concrete application of our analysis of smooth convex losses is linear regression. In linear regression, $\mathbb{X}$ is a Hilbert space with an inner product $\langle\cdot,\cdot\rangle_{\mathbb{X}}$, and it is both the data space and the parameter space. The loss $\ell \equiv \ell^{\mathrm{sq}}$ is the squared loss

$$\ell^{\mathrm{sq}}((\boldsymbol{x}, y), \boldsymbol{w}) := \frac{1}{2}(\boldsymbol{x}^\top \boldsymbol{w} - y)^2.$$

L^{sq} and $L^{\mathrm{sq}}_\star$ are defined similarly to L and $L_\star$.

Unlike standard high-probability bounds for regression, we give bounds that make no assumption on the range or the tails of the distribution of the response variables, other than a trivial requirement that the optimal squared loss be finite. The assumptions on the distribution of the covariates are also minimal.

1. Formal definitions of terms used in the conditions are given in Section 4.

Let Σ be the second-moment operator $\boldsymbol{a} \mapsto \mathbb{E}(\boldsymbol{X}\langle \boldsymbol{X}, \boldsymbol{a}\rangle_{\mathbb{X}})$, where $\boldsymbol{X}$ is a random data point from the marginal distribution of $\mathcal{D}$ on $\mathbb{X}$. For a finite-dimensional $\mathbb{X}$, Σ is simply the (uncentered) covariance matrix $\mathbb{E}[\boldsymbol{X}\boldsymbol{X}^\top]$. First, consider the finite-dimensional case, where $\mathbb{X} = \mathbb{R}^d$, and assume Σ is not singular. Let $\|\cdot\|_2$ denote the Euclidean norm in $\mathbb{R}^d$. Under only bounded $4+\epsilon$ moments of the marginal on $\mathbb{X}$ (a condition that we specify in full detail in Section 5), we show the following guarantee.

Theorem 3 *Assume the marginal of $\mathbb{X}$ has bounded $4+\epsilon$ moments. There is a constant $C > 0$ and an algorithm that accepts as input a sample of size n and a confidence parameter $\delta \in (0,1)$, and returns $\hat{\boldsymbol{w}} \in \mathbb{X}$, such that if $n \geq Cd\log(1/\delta)$, with probability at least $1-\delta$,*

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \leq L^{\mathrm{sq}}_\star + O\left(\frac{\mathbb{E}(\|\Sigma^{-1/2}\boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{w}_\star - Y)\|_2^2)\log(1/\delta)}{n}\right).$$

This theorem is stated in full as Theorem 19 in Section 5. Under standard finite fourth-moment conditions, this result translates to the bound

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \leq \left(1 + O\left(\frac{d\log(1/\delta)}{n}\right)\right) L^{\mathrm{sq}}_\star,$$

with probability $\geq 1-\delta$. These results improve over recent results by Audibert and Catoni (2011), Catoni (2012), and Mahdavi and Jin (2013). We provide a full comparison to related work in Section 5.

Theorem 3 can be specialized for specific cases of interest. For instance, suppose $\boldsymbol{X}$ is bounded and well-conditioned in the sense that there exists $R < \infty$ such that $\Pr[\boldsymbol{X}^\top\Sigma^{-1}\boldsymbol{X} \leq R^2] = 1$, but Y may still be heavy-tailed. Under this assumption we have the following result.

Theorem 4 *Assume Σ is not singular. There exists an algorithm that accepts as input a sample of size n and a confidence parameter $\delta \in (0,1)$, and returns $\hat{\boldsymbol{w}} \in \mathbb{X}$, such that with probability at least $1-\delta$, for $n \geq O(R^2\log(R)\log(e/\delta))$,*

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \leq \left(1 + O\left(\frac{R^2\log(1/\delta)}{n}\right)\right) L^{\mathrm{sq}}_\star.$$

This theorem is stated in full as Theorem 20 in Section 5. Note that

$$\mathbb{E}(\boldsymbol{X}^\top\Sigma^{-1}\boldsymbol{X}) = \mathbb{E}\,\mathrm{tr}(\boldsymbol{X}^\top\Sigma^{-1}\boldsymbol{X}) = \mathrm{tr}(\mathrm{Id}) = d,$$

so $R = \Omega(\sqrt{d})$. R^2 is closely related to a *condition number* for the distribution of $\boldsymbol{X}$. For instance, if $\mathbb{P}[\|\boldsymbol{X}\| = 1] = 1$, then $R^2 \leq d\frac{\lambda_{\max}(\Sigma)}{\lambda_{\min}(\Sigma)}$. This result is minimax optimal up to logarithmic factors (see, *e.g.*, Nussbaum, 1999). We also remark that the boundedness assumption can be replaced by a subgaussian assumption on $\boldsymbol{X}$, in which case the sample size requirement becomes $O(d\log(1/\delta))$. We give analogous guarantees for the case of regularized least squares in a possibly finite-dimensional Hilbert space in Theorem 21, Section 5.

2.5 Other Applications, Comparisons, and Extensions

The general method studied here allows handling heavy tails in other applications as well. We give two examples in Section 6. First, we consider parameter estimation using L^1-regularized linear least squares regression (Lasso) under random subgaussian design. We show that using the above approach, parameter estimation bounds can be guaranteed for general bounded variance noise, including heavy-tailed noise. This contrasts with standard results that assume sub-Gaussian noise. Second, we show that low-rank covariance matrix approximation can be obtained for heavy-tailed distributions, under a bounded $4+\epsilon$ moment assumption. These two applications have been analyzed also in the independent and simultaneous work of Minsker (2013).

All the results above are provided using a specific solution to the Robust Distance Approximation problem, which is easy to implement for any metric space. For the case of a fully known metric, in a Banach or a Hilbert space, Minsker (2013) proposed a different solution, which is based on the geometric median. In Section 7, we provide a detailed comparison of the approximation factor achieved by each approach, as well as some general lower bounds. Several interesting open questions remain regarding this general problem.

Lastly, in Section 8, we give a short proof to the intuitive fact that in some prediction problem, one can replace Robust Distance Approximation with taking the median of the predictions of the input estimators. This gives a possible improper-learning algorithm for relevant learning settings.

All of the techniques we have developed in this work are simple enough to implement and empirically evaluate, and indeed in some simulated experiments, we have verified the improvements over standard methods such as the empirical mean when the data follow heavy-tailed distributions. However, at present, the relatively large constant factors in our bounds are real enough to restrict the empirical improvements only to settings where very high confidence (*i.e.*, small values of δ) is required. By contrast, with an appropriately determined noise variance, the techniques of Catoni (2012) and Brownlees et al. (2014) may yield improvements more readily. Nevertheless, since our techniques are more general in some respects, it is worth investigating whether they can be made more practical (*e.g.*, with greater sample reuse or overlapping groups), and we plan to do this in future work.

3. The Core Techniques

In this section we present the core technique used for achieving exponential concentration. We first demonstrate the underlying principle via the median-of-means estimator, and then explain the generalization to arbitrary metric spaces. Finally, we show a new generalization that supports noisy feature measurements.

3.1 Warm-up: Median-of-Means Estimator

We first motivate the estimation procedure by considering the special case of estimating a scalar population mean using a *median-of-means* estimator, given in Algorithm 1. This estimator, heavily used in the streaming algorithm literature (Alon et al., 1999) (though a similar technique also appears in Nemirovsky and Yudin (1983) as noted in Levin (2005)), partitions a sample into k equal-size groups, and returns the median of the sample means

Algorithm 1 Median-of-means estimator

input Sample $S \subset \mathbb{R}$ of size n, number of groups $k \in \mathbb{N}$ such that $k \leq n/4$.
output Population mean estimate $\hat{\mu} \in \mathbb{R}$.
1: Randomly partition S into k subsets $S_1, S_2, \ldots, S_k$, each of size at least $\lfloor n/k \rfloor$.
2: For each $i \in [k]$, let $\mu_i \in \mathbb{R}$ be the sample mean of S_i.
3: Return $\hat{\mu} := \text{median}\{\mu_1, \mu_2, \ldots, \mu_k\}$.

of each group. Note that the possible non-uniqueness of the median does not affect the result; the arguments below apply to any one of them. The input parameter k should be thought of as a constant determined by the desired confidence level (*i.e.*, $k = \Theta(\log(1/\delta))$ for confidence $\delta \in (0,1)$). It is well known that the median-of-means achieves estimation with exponential concentration. The following proposition gives a simple statement and proof. The constant 6 in the statement (see Eq. (1) below) is lower the constant in the analysis of Lerasle and Oliveira (2011, Proposition 1), which is $2\sqrt{6e} \approx 8.08$, but we require a larger value of n. By requiring an even larger n, the constant in the statement below can approach $3\sqrt{3}$.

Proposition 5 *Let x be a random variable with mean μ and variance $\sigma^2 < \infty$, and let S be a set of n independent copies of x. Assume $k \leq n/2$. With probability at least $1-e^{-k/4.5}$, the estimate $\hat{\mu}$ returned by Algorithm 1 on input (S,k) satisfies $|\hat{\mu}-\mu| \leq \sigma\sqrt{8k/n}$. Therefore, if $k = 4.5\lceil \log(1/\delta) \rceil$ and $n \geq 18\lceil \log(1/\delta) \rceil$, then with probability at least $1-\delta$,*

$$|\hat{\mu}-\mu| \leq 6\sigma\sqrt{\frac{\lceil \log(1/\delta) \rceil}{n}}. \tag{1}$$

Proof First, assume k divides n. Pick any $i \in [k]$, and observe that S_i is an i.i.d. sample of size n/k. Therefore, by Chebyshev's inequality, $\Pr[|\mu_i - \mu| \leq \sqrt{6\sigma^2 k/n}] \geq 5/6$. For each $i \in [k]$, let $b_i := \mathbb{1}\{|\mu_i - \mu| \leq \sqrt{6\sigma^2 k/n}\}$. Note that the b_i are independent indicator random variables, each with $\mathbb{E}(b_i) \geq 5/6$. By Hoeffding's inequality, $\Pr[\sum_{i=1}^k b_i > k/2] \geq 1-e^{-k/4.5}$. In the event that $\sum_{i=1}^k b_i > k/2$, at least half of the μ_i are within $\sqrt{6\sigma^2 k/n}$ of μ, which means that the same holds for the median of the μ_i. If k does not divide n then the analysis can be carried out by substituting n with $\lfloor n/k \rfloor k \geq n - k \geq \frac{3}{4}n$, which scales the guarantee by a factor of $\sqrt{4/3}$. ■

Using the terminology of Robust Distance Approximation with the metric $\rho(x,y) = |x-y|$, the proof shows that with high probability over the choice of W, $\Delta_W(\mu, 0) \leq \sqrt{12\sigma^2 k/n}$. The result then immediately follows because on the space $(\mathbb{R}, \rho)$, the median is a Robust Distance Approximation procedure with $C_0 = 1$.

Remark 6 (Catoni's M-estimator) *Catoni (2012) proposes a mean estimator $\hat{\mu}$ that satisfies $|\hat{\mu}-\mu| = O(\sigma\sqrt{\log(1/\delta)/n})$ with probability at least $1-\delta$. Remarkably, the leading constant in the bound is asymptotically optimal: it approaches $\sqrt{2}$ as $n \to \infty$. However, the estimator takes both δ and σ as inputs. Catoni also presents an estimator that takes only σ as an input; this estimator guarantees a $O(\sigma \log(1/\delta)/\sqrt{n})$ bound for all values of $\delta > \exp(1-n/2)$* simultaneously.

Remark 7 (Empirical mean) *Catoni (2012) shows that the empirical mean cannot provide a qualitatively similar guarantee. Specifically, for any $\sigma > 0$ and $\delta \in (0, 1/(2e))$, there is a distribution with mean zero and variance σ^2 such that the empirical average $\hat{\mu}_{\text{emp}}$ of n i.i.d. draws satisfies*

$$\Pr\left[|\hat{\mu}_{\text{emp}}| \geq \frac{\sigma}{\sqrt{2n\delta}}\left(1 - \frac{2e\delta}{n}\right)^{\frac{n-1}{2}}\right] \geq 2\delta. \tag{2}$$

Therefore the deviation of the empirical mean necessarily scales with $1/\sqrt{\delta}$ rather than $\sqrt{\log(1/\delta)}$ (with probability $\Omega(\delta)$).

3.2 Generalization to Arbitrary Metric Spaces

We now consider a simple generalization of the median-of-means estimator for arbitrary metric spaces, first mentioned in Nemirovsky and Yudin (1983). Let $\mathbb{X}$ be the parameter (solution) space, $\boldsymbol{w}_\star \in \mathbb{X}$ be a distinguished point in $\mathbb{X}$ (the target solution), and ρ a metric on $\mathbb{X}$ (in fact, a pseudometric suffices).

The first abstraction captures the generation of candidate solutions obtained from independent subsamples. We assume there is an oracle $\text{APPROX}_{\rho,\varepsilon}$ which satisfies the following assumptions.

Assumption 1 *A query to $\text{APPROX}_{\rho,\varepsilon}$ returns a random $\boldsymbol{w} \in \mathbb{X}$ such that*

$$\Pr\Big[\rho(\boldsymbol{w}_\star, \boldsymbol{w}) \leq \varepsilon\Big] \geq 2/3.$$

Note that the 2/3 could be replaced by another constant larger than half; we have not optimized the constants. The second assumption regards statistical independence. For an integer k, let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ be responses to k separate queries to $\text{APPROX}_{\rho,\varepsilon}$.

Assumption 2 *$\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ are statistically independent.*

The proposed procedure, given in Algorithm 2, generates k candidate solutions by querying $\text{APPROX}_{\rho,\varepsilon}$ k times, and then selecting a single candidate using a generalization of the median. Specifically, for each $i \in [k]$, the smallest ball centered at $\boldsymbol{w}_i$ that contains more than half of $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_k\}$ is determined; the $\boldsymbol{w}_i$ with the smallest such ball is returned. If there are multiple such $\boldsymbol{w}_i$ with the smallest radius ball, any one of them may be selected. This selection method is a Robust Distance Approximation procedure. The proof is given below and illustrated in Figure 1. Nemirovsky and Yudin (1983) proposed a similar technique, however their formulation relies on knowledge of ε.

Proposition 8 *Let $r_i := \min\{r \geq 0 : |B_\rho(\boldsymbol{w}_i, r) \cap W| > k/2\}$. Selecting $\boldsymbol{w}_{i_\star}$ such that $i_\star = \text{argmin}_i\, r_i$ is a Robust Distance Approximation procedure with $C_0 = 3$.*

Proof Assume that $\Delta(w_\star, 0) \leq \varepsilon$. Then $|B_\rho(\boldsymbol{w}_\star, \varepsilon) \cap W| > k/2$. For any $\boldsymbol{v} \in B_\rho(\boldsymbol{w}_\star, \varepsilon) \cap W$, by the triangle inequality, $|B_\rho(\boldsymbol{v}, 2\varepsilon) \cap W| > k/2$. This implies that $r_{i_\star} \leq 2\varepsilon$, and so $|B_\rho(\boldsymbol{w}_{i_\star}, 2\varepsilon) \cap W| > k/2$. By the pigeonhole principle, $B_\rho(\boldsymbol{w}_\star, \varepsilon) \cap B_\rho(\boldsymbol{w}_{i_\star}, 2\varepsilon) \neq \emptyset$. Therefore, by the triangle inequality again, $\rho(\boldsymbol{w}_\star, \boldsymbol{w}_{i_\star}) \leq 3\varepsilon$. ■

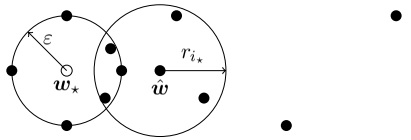


Figure 1: The argument in the proof of Proposition 8, illustrated on the Euclidean plane. If more than half of the $\boldsymbol{w}_i$ (depicted by full circles) are within ε of $\boldsymbol{w}_\star$ (the empty circle), then the selected $\boldsymbol{w}_{i_\star}$ is within $\varepsilon + r_{i_\star} \leq 3\varepsilon$ of $\boldsymbol{w}_\star$.

Algorithm 2 Robust approximation

input Number of candidates k, query access to $\text{APPROX}_{\rho,\varepsilon}$.

output Approximate solution $\hat{\boldsymbol{w}} \in \mathbb{X}$.

1: Query $\text{APPROX}_{\rho,\varepsilon}$ k times. Let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ be the responses to the queries; set $W := \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_k\}$.

2: For each $i \in [k]$, let $r_i := \min\{r \geq 0 : |B_\rho(\boldsymbol{w}_i, r) \cap W| > k/2\}$; set $i_\star := \arg\min_{i \in [k]} r_i$.

3: Return $\hat{\boldsymbol{w}} := \boldsymbol{w}_{i_\star}$.

Again, the number of candidates k determines the resulting confidence level. The following theorem provides a guarantee for Algorithm 2. We note that the resulting constants here might not be optimal in specific applications, since they depend on the arbitrary constant in Assumption 1.

Proposition 9 *Suppose that Assumption 1 and Assumption 2 hold. Then, with probability at least $1 - e^{-k/18}$, Algorithm 2 returns $\hat{\boldsymbol{w}} \in \mathbb{X}$ satisfying $\rho(\boldsymbol{w}_\star, \hat{\boldsymbol{w}}) \leq 3\varepsilon$.*

Proof For each $i \in [k]$, let $b_i := \mathbb{1}\{\rho(\boldsymbol{w}_\star, \boldsymbol{w}_i) \leq \varepsilon\}$. Note that the b_i are independent indicator random variables, each with $\mathbb{E}(b_i) \geq 2/3$. By Hoeffding's inequality, $\Pr[\sum_{i=1}^k b_i > k/2] \geq 1 - e^{-k/18}$. In the event that $\sum_{i=1}^k b_i > k/2$, more than half of the $\boldsymbol{w}_i$ are contained in the ball of radius ε around $\boldsymbol{w}_\star$, that is $\Delta_W(w_\star, 0) \leq \varepsilon$. The result follows from Proposition 8. ■

3.3 Random Distance Measurements

In some problems, the most appropriate metric on $\mathbb{X}$ in which to measure accuracy is not directly computable. For instance, the metric may depend on population quantities which can only be estimated; moreover, the estimates may only be relatively accurate with some constant probability. For instance, this is the case when the metric depends on the population covariance matrix, a situation we consider in Section 5.2.3.

To capture such cases, we assume access to a metric estimation oracle as follows. Let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ be responses to k queries to $\text{APPROX}_{\rho,\epsilon}$. The metric estimation oracle, denoted DIST_ρ^j, provides (possibly via a random process) a function $f_j : \mathbb{X} \to \mathbb{R}_+$. $f_j(\boldsymbol{v})$ will be used

as an estimate of $\rho(\boldsymbol{v}, \boldsymbol{w}_j)$. This estimate is required to be weakly accurate, as captured by the following definition of the random variables Z_j. Let $f_1, \ldots, f_k$ be responses to queries to $\mathrm{DIST}^1_\rho, \ldots, \mathrm{DIST}^k_\rho$, respectively. For $j \in [k]$, define

$$Z_j := \mathbb{1}\{\forall \boldsymbol{v} \in \mathbb{X}, \quad (1/2)\rho(\boldsymbol{v}, \boldsymbol{w}_j) \le f_j(\boldsymbol{v}) \le 2\rho(\boldsymbol{v}, \boldsymbol{w}_j)\}.$$

$Z_j = 1$ indicates that f_j provides a sufficiently accurate estimate of the distances from $\boldsymbol{w}_j$. Note that f_j need not correspond to a metric. We assume the following.

Assumption 3 *For any $j \in [k]$, $\Pr[Z_j = 1] \ge 8/9$.*

We further require the following independence assumption.

Assumption 4 *The random variables $Z_1, \ldots, Z_k$ are statistically independent.*

Note that there is no assumption on the statistical relationship between $Z_1, \ldots, Z_k$ and $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$.

Algorithm 3 is a variant of Algorithm 2 that simply replaces computation of ρ-distances with computations using the functions returned by querying the DIST^j_ρ's. The resulting selection procedure is, with high probability, a Robust Distance Approximation.

Lemma 10 *Consider a run of Algorithm 3, with output $\hat{\boldsymbol{w}}$. Let $Z_1, \ldots, Z_k$ as defined above, and suppose that Assumption 3 and Assumption 4 hold. Then, with probability at least $1 - e^{-k/648}$,*

$$\rho(\hat{\boldsymbol{w}}, \boldsymbol{w}_\star) \le 9 \cdot \Delta_W(\boldsymbol{w}_\star, \frac{5}{36}),$$

where $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k\}$.

Proof By Assumptions 3 and 4, and by Hoeffding's inequality,

$$\Pr\left[\sum_{j=1}^{k} Z_j > \frac{31}{36}k\right] \ge 1 - e^{-k/648} \tag{3}$$

Assume this event holds, and denote $\varepsilon = \Delta_W(\boldsymbol{w}_\star, \frac{5}{36})$. We have $|B(\boldsymbol{w}_\star, \varepsilon) \cap W| \ge \frac{23}{36}k$.

Let $i \in [k]$ such that $\boldsymbol{w}_i \in B_\rho(\boldsymbol{w}_\star, \varepsilon)$. Then, for any $j \in [k]$ such that $\boldsymbol{w}_j \in B_\rho(\boldsymbol{w}_\star, \varepsilon)$, by the triangle inequality $\rho(\boldsymbol{w}_i, \boldsymbol{w}_j) \le 2\varepsilon$. There are at least $\frac{23}{36}k$ such indices j, therefore for more than $k/2$ of the indices j, we have

$$\rho(\boldsymbol{w}_i, \boldsymbol{w}_j) \le 2\varepsilon \text{ and } Z_j = 1.$$

For j such that this holds, by the definition of Z_j, $f_j(\boldsymbol{w}_i) \le 4\varepsilon$. It follows that $r_i := \mathrm{median}\{f_j(\boldsymbol{w}_i) \mid j \in [k]\} \le 4\epsilon$.

Now, let $i \in [k]$ such that $\boldsymbol{w}_i \notin B(\boldsymbol{w}_\star, 9\epsilon)$. Then, for any $j \in [k]$ such that $\boldsymbol{w}_j \in B_\rho(\boldsymbol{w}_\star, \varepsilon)$, by the triangle inequality $\rho(\boldsymbol{w}_i, \boldsymbol{w}_j) \ge \rho(\boldsymbol{w}_\star, \boldsymbol{w}_i) - \rho(\boldsymbol{w}_\star, \boldsymbol{w}_j) > 8\varepsilon$. As above, for more than $k/2$ of the indices j,

$$\rho(\boldsymbol{w}_i, \boldsymbol{w}_j) > 8\varepsilon \text{ and } Z_j = 1.$$

For j such that this holds, by the definition of Z_j, $f_j(\boldsymbol{w}_i) > 4\varepsilon$. It follows that $r_i := \mathrm{median}\{f_j(\boldsymbol{w}_i) \mid j \in [k]\} > 4\epsilon$.

By Eq. (3), We conclude that with probability at least $1 - \exp(-k/648)$,

Algorithm 3 Robust approximation with random distances

input Number of candidates k, query access to $\mathrm{APPROX}_{\rho,\varepsilon}$, query access to DIST_ρ.
output Approximate solution $\hat{\boldsymbol{w}} \in \mathbb{X}$.

1: Query $\mathrm{APPROX}_{\rho,\varepsilon}$ k times. Let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ be the responses to the queries; set $W := \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_k\}$.
2: For $i \in [k]$, let f_i be the response of DIST^j_ρ, and set $r_i := \mathrm{median}\{f_j(\boldsymbol{w}_i) : j \in [k]\}$; set $i_\star := \arg\min_{i \in [k]} r_i$.
3: Return $\hat{\boldsymbol{w}} := \boldsymbol{w}_{i_\star}$.

1. $r_i \le 4\varepsilon$ for all $\boldsymbol{w}_i \in W \cap B_\rho(\boldsymbol{w}_\star, \varepsilon)$, and
2. $r_i > 4\varepsilon$ for all $\boldsymbol{w}_i \in W \setminus B_\rho(\boldsymbol{w}_\star, 9\varepsilon)$.

In this event the $\boldsymbol{w}_i \in W$ with the smallest r_i satisfies $\boldsymbol{w}_i \in B_\rho(\boldsymbol{w}_\star, 9\varepsilon)$. ■

The properties of the approximation procedure and of $\mathrm{APPROX}_{\rho,\epsilon}$ are combined to give a guarantee for Algorithm 3.

Theorem 11 *Suppose that Assumptions 1,2,3,4 all hold. With probability at least $1 - 2e^{-k/648}$, Algorithm 3 returns $\hat{\boldsymbol{w}} \in \mathbb{X}$ satisfying $\rho(\boldsymbol{w}_\star, \hat{\boldsymbol{w}}) \le 9\varepsilon$.*

Proof For each $i \in [k]$, let $b_i := \mathbb{1}\{\rho(\boldsymbol{w}_\star, \boldsymbol{w}_i) \le \varepsilon\}$. By Assumptions 1 and 2, the b_i are independent indicator random variables, each with $\mathbb{E}(b_i) \ge 2/3$. By Hoeffding's inequality, $\Pr[\sum_{i=1}^k b_i > \frac{23}{36}k] \ge 1 - e^{-k/648}$. The result follows from Lemma 10 and a union bound. ■

In the following sections we show several applications of these general techniques.

4. Minimizing Strongly Convex Losses

In this section we apply the core techniques to the problem of approximately minimizing strongly convex losses, which includes least squares linear regression as a special case.

4.1 Preliminaries

Suppose $(\mathbb{X}, \|\cdot\|)$ is a Banach space, with the metric ρ induced by the norm $\|\cdot\|$. We sometimes denote the metric by $\|\cdot\|$ as well. Denote by $\|\cdot\|_*$ the dual norm, so $\|\boldsymbol{y}\|_* = \sup\{\langle \boldsymbol{y}, \boldsymbol{x}\rangle \colon \boldsymbol{x} \in \mathbb{X}, \|\boldsymbol{x}\| \le 1\}$ for $\boldsymbol{y} \in \mathbb{X}^*$.

The derivative of a differentiable function $f \colon \mathbb{X} \to \mathbb{R}$ at $\boldsymbol{x} \in \mathbb{X}$ in direction $\boldsymbol{u} \in \mathbb{X}$ is denoted by $\langle \nabla f(\boldsymbol{x}), \boldsymbol{u}\rangle$. We say f is *α-strongly convex with respect to* $\|\cdot\|$ if

$$f(\boldsymbol{x}) \ge f(\boldsymbol{x}') + \langle \nabla f(\boldsymbol{x}'), \boldsymbol{x} - \boldsymbol{x}'\rangle + \frac{\alpha}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2$$

for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{X}$; it is *$\beta$-smooth with respect to* $\|\cdot\|$ if for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{X}$

$$f(\boldsymbol{x}) \le f(\boldsymbol{x}') + \langle \nabla f(\boldsymbol{x}'), \boldsymbol{x} - \boldsymbol{x}'\rangle + \frac{\beta}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2.$$

We say $\|\cdot\|$ is γ-*smooth* if $\boldsymbol{x} \mapsto \frac{1}{2}\|\boldsymbol{x}\|^2$ is γ-smooth with respect to $\|\cdot\|$. We define n_α to be the smallest sample size such that the following holds: With probability $\geq 5/6$ over the choice of an i.i.d. sample T of size $|T| \geq n_\alpha$ from $\mathcal{D}$, for all $\boldsymbol{w} \in \mathbb{X}$,

$$L_T(\boldsymbol{w}) \geq L_T(\boldsymbol{w}_\star) + \langle \nabla L_T(\boldsymbol{w}_\star), \boldsymbol{w} - \boldsymbol{w}_\star \rangle + \frac{\alpha}{2}\|\boldsymbol{w} - \boldsymbol{w}_\star\|^2. \tag{4}$$

In other words, the sample T induces a loss L_T which is α-strongly convex around $\boldsymbol{w}_\star$.[2] We assume that $n_\alpha < \infty$ for some $\alpha > 0$.

We use the following facts in our analysis.

Proposition 12 (Srebro et al., 2010) *If a non-negative function $f\colon \mathbb{X} \to \mathbb{R}_+$ is β-smooth with respect to $\|\cdot\|$, then $\|\nabla f(\boldsymbol{x})\|_*^2 \leq 4\beta f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{X}$.*

Proposition 13 (Juditsky and Nemirovski, 2008) *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent copies of a zero-mean random vector $\boldsymbol{X}$, and let $\|\cdot\|$ be γ-smooth. Then $\mathbb{E}\|n^{-1}\sum_{i=1}^n \boldsymbol{X}_i\|^2 \leq (\gamma/n)\mathbb{E}\|\boldsymbol{X}\|^2$.*

Recall that $\mathcal{Z}$ is a data space, and $\mathcal{D}$ is a distribution over $\mathcal{Z}$. Let Z be a $\mathcal{Z}$-valued random variable with distribution $\mathcal{D}$. Let $\ell\colon \mathcal{Z} \times \mathbb{X} \to \mathbb{R}_+$ be a non-negative loss function, and for $\boldsymbol{w} \in \mathbb{X}$, let $L(\boldsymbol{w}) := \mathbb{E}(\ell(Z, \boldsymbol{w}))$ be the expected loss. Also define the empirical loss with respect to a sample T from $\mathcal{Z}$, $L_T(\boldsymbol{w}) := |T|^{-1}\sum_{z\in T}\ell(z, \boldsymbol{w})$. To simplify the discussion throughout, we assume ℓ is differentiable, which is anyway our primary case of interest. We assume that L has a unique minimizer $\boldsymbol{w}_\star := \arg\min_{\boldsymbol{w}\in\mathbb{X}} L(\boldsymbol{w})$.[3] Let $L_\star := \min_{\boldsymbol{w}} L(\boldsymbol{w})$. Set $\boldsymbol{w}_\star$ such that $L_\star = L(\boldsymbol{w}_\star)$.

4.2 Subsampled Empirical Loss Minimization

To use Algorithm 2, we implement $\text{APPROX}_{\|\cdot\|_*,\varepsilon}$ based on loss minimization over subsamples, as follows: Given a sample $S \subseteq \mathcal{Z}$, randomly partition S into k groups $S_1, S_2, \ldots, S_k$, each of size at least $\lfloor |S|/k \rfloor$, and let the response to the i-th query to $\text{APPROX}_{\|\cdot\|_*,\varepsilon}$ be the loss minimizer on S_i, *i.e.*, $\boldsymbol{w}_i = \arg\min_{\boldsymbol{w}\in\mathbb{X}} L_{S_i}(\boldsymbol{w})$. We call this implementation *subsampled empirical loss minimization*. Clearly, if S is an i.i.d. sample from $\mathcal{D}$, then $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ are statistically independent, and so Assumption 2 holds. Thus, to apply Proposition 9, it is left to show that Assumption 1 holds as well.[4]

The following lemma proves that Assumption 1 holds under these assumptions with

$$\varepsilon := \sqrt{\frac{32\gamma k \mathbb{E}\|\nabla\ell(Z, \boldsymbol{w}_\star)\|_*^2}{n\alpha^2}}. \tag{5}$$

Lemma 14 *Let ε be as defined in Eq. (5). Assume $k \leq n/4$, and that S is an i.i.d. sample from $\mathcal{D}$ of size n such that $\lfloor n/k \rfloor \geq n_\alpha$. Then subsampled empirical loss minimization using the sample S is a correct implementation of $\text{APPROX}_{\|\cdot\|_*,\varepsilon}$ for up to k queries.*

2. Technically, we only need the sample size to guarantee Eq. (4) for all $\boldsymbol{w} \in B_{\|\cdot\|}(\boldsymbol{w}_\star, r)$ for some $r > 0$.
3. This holds, for instance, if L is strongly convex.
4. An approach akin to the bootstrap technique (Efron, 1979) could also seem natural here: In this approach, $S_1, \ldots, S_k$ would be generated by randomly sub-sampling from S, with possible overlap between the subsamples. However, this approach does not satisfy Assumption 2, since loss minimizers of overlapping samples are not statistically independent.

Proof Let $T = \lfloor n/k \rfloor$. Since $n \geq 4k$, we have $\lfloor n/k \rfloor k \geq n - k \geq \frac{3}{4}n$, therefore $1/T \leq \frac{4k}{3n}$. It is clear that $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_k$ are independent by the assumption. Fix some $i \in [k]$. Observe that $\nabla L(\boldsymbol{w}_\star) = \mathbb{E}(\nabla\ell(Z, \boldsymbol{w}_\star)) = 0$, and therefore by Proposition 13:

$$\mathbb{E}\|\nabla L_{S_i}(\boldsymbol{w}_\star)\|_*^2 \leq (\gamma/T)\mathbb{E}\|\nabla\ell(Z, \boldsymbol{w}_\star)\|_*^2 \leq \frac{4\gamma k}{3n}\mathbb{E}\|\nabla\ell(Z, \boldsymbol{w}_\star)\|_*^2.$$

By Markov's inequality,

$$\Pr\left[\|\nabla L_{S_i}(\boldsymbol{w}_\star)\|_*^2 \leq \frac{8\gamma k}{n}\mathbb{E}(\|\nabla\ell(Z, \boldsymbol{w}_\star)\|_*^2)\right] \geq \frac{5}{6}.$$

Moreover, the assumption that $\lfloor n/k \rfloor \geq n_\alpha$ implies that with probability at least 5/6, Eq. (4) holds for $T = S_i$. By a union bound, both of these events hold simultaneously with probability at least 2/3. In the intersection of these events, letting $\boldsymbol{w}_i := \arg\min_{\boldsymbol{w}\in\mathbb{X}} L_{S_i}(\boldsymbol{w})$,

$$\begin{aligned}(\alpha/2)\|\boldsymbol{w}_i - \boldsymbol{w}_\star\|^2 &\leq -\langle \nabla L_{S_i}(\boldsymbol{w}_\star), \boldsymbol{w}_i - \boldsymbol{w}_\star\rangle + L_{S_i}(\boldsymbol{w}_i) - L_{S_i}(\boldsymbol{w}_\star)\\ &\leq \|\nabla L_{S_i}(\boldsymbol{w}_\star)\|_*\|\boldsymbol{w}_i - \boldsymbol{w}_\star\|,\end{aligned}$$

where the last inequality follows from the definition of the dual norm, and the optimality of $\boldsymbol{w}_i$ on L_{S_i}. Rearranging and combining with the above probability inequality implies

$$\Pr\left[\|\boldsymbol{w}_i - \boldsymbol{w}_\star\| \leq \varepsilon\right] \geq \frac{2}{3}$$

as required. ■

Combining Lemma 14 and Proposition 9 gives the following theorem.

Theorem 15 *Let n_α be as defined in Section 4.1, and assume that $\|\cdot\|_*$ is γ-smooth. Also, assume $k := 18\lceil\log(1/\delta)\rceil$, $n \geq 72\lceil\log(1/\delta)\rceil$, and that S is an i.i.d. sample from $\mathcal{D}$ of size n such that $\lfloor n/k \rfloor \geq n_\alpha$. Finally, assume Algorithm 3 uses the subsampled empirical loss minimization to implement $\text{APPROX}_{\|\cdot\|_*,\varepsilon}$, where ε is as in Eq. (5). Then with probability at least $1 - \delta$, the parameter $\hat{\boldsymbol{w}}$ returned by Algorithm 2 satisfies*

$$\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\| \leq 72\sqrt{\frac{\gamma\lceil\log(1/\delta)\rceil\mathbb{E}\|\nabla\ell(Z, \boldsymbol{w}_\star)\|_*^2}{n\alpha^2}}.$$

We give an easy corollary of Theorem 15 for the case where ℓ is smooth. This is the full version of Theorem 2.

Corollary 16 *Assume the same conditions as Theorem 15, and also that:*

- *$\boldsymbol{w} \mapsto \ell(z, \boldsymbol{w})$ is β-smooth with respect to $\|\cdot\|$ for all $z \in \mathcal{Z}$;*
- *$\boldsymbol{w} \mapsto L(\boldsymbol{w})$ is $\bar{\beta}$-smooth with respect to $\|\cdot\|$.*

Then with probability at least $1 - \delta$,

$$L(\hat{\boldsymbol{w}}) \leq \left(1 + \frac{10368\beta\bar{\beta}\gamma\lceil\log(1/\delta)\rceil}{n\alpha^2}\right)L(\boldsymbol{w}_\star).$$

Proof This follows from Theorem 15 by first concluding that $\mathbb{E}[\|\nabla\ell(Z,\boldsymbol{w}_\star)\|_*^2] \le 4\beta L(\boldsymbol{w}_\star)$, using the β-strong smoothness assumption on ℓ and Proposition 12, and then noting that $L(\hat{\boldsymbol{w}}) - L(\boldsymbol{w}_\star) \le \frac{\bar\beta}{2}\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|^2$, due to the strong smoothness of L and the optimality of $L(\boldsymbol{w}_\star)$. ■

Corollary 16 implies that for smooth losses, Algorithm 2 provides a constant factor approximation to the optimal loss with a sample size $\max\{n_\alpha, \gamma\beta\bar\beta/\alpha^2\} \cdot O(\log(1/\delta))$ (with probability at least $1-\delta$). In subsequent sections, we exemplify cases where the two arguments of the max are roughly of the same order, and thus imply a sample size requirement of $O(\gamma\bar\beta\beta/\alpha^2 \log(1/\delta))$. Note that there is no dependence on the optimal loss $L(\boldsymbol{w}_\star)$ in the sample size, and the algorithm has no parameters besides $k = O(\log(1/\delta))$.

We can also obtain a variant of Theorem 15 based on Algorithm 3 and Theorem 11, in which we assume that there exists some sample size $n_{k,\mathrm{DIST}_{\|\cdot\|}}$ that allows $\mathrm{DIST}_{\|\cdot\|}$ to be correctly implemented using an i.i.d. sample of size at least $n_{k,\mathrm{DIST}_{\|\cdot\|}}$. Under such an assumption, essentially the same guarantee as in Theorem 15 can be afforded to Algorithm 3 using the subsampled empirical loss minimization to implement $\mathrm{APPROX}_{\|\cdot\|,\varepsilon}$ (for ε as in Eq. (5)) and the assumed implementation of $\mathrm{DIST}_{\|\cdot\|}$. Note that since Theorem 11 does not require $\mathrm{APPROX}_{\|\cdot\|,\varepsilon}$ and $\mathrm{DIST}_{\|\cdot\|}$ to be statistically independent, both can be implemented using the same sample.

Theorem 17 *Let n_α be as defined in Section 4.1, $n_{k,\mathrm{DIST}_{\|\cdot\|}}$ be as defined above, and assume that $\|\cdot\|_*$ is γ-smooth. Also, assume $k := 648\lceil\log(2/\delta)\rceil$, S is an i.i.d. sample from $\mathcal{D}$ of size n such that $n \ge \max\{4k, n_{k,\mathrm{DIST}_{\|\cdot\|}}\}$, and $\lfloor n/k\rfloor \ge n_\alpha$. Further, assume Algorithm 3 implements $\mathrm{APPROX}_{\|\cdot\|,\varepsilon}$ using S with subsampled empirical loss minimization, where ε is as in Eq. (5), and implements $\mathrm{DIST}_{\|\cdot\|}$ using S as well. Then with probability at least $1-\delta$, the parameter $\hat{\boldsymbol{w}}$ returned by Algorithm 3 satisfies*

$$\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\| \le 1296\sqrt{\frac{\gamma\lceil\log(2/\delta)\rceil\mathbb{E}\|\nabla\ell(Z,\boldsymbol{w}_\star)\|_*^2}{n\alpha^2}}.$$

Remark 18 (Mean estimation and empirical risk minimization) *The problem of estimating a scalar population mean is a special case of the loss minimization problem, where $\mathcal{Z} = \mathbb{X} = \mathbb{R}$, and the loss function of interest is the square loss $\ell(z,w) = (z-w)^2$. The minimum population loss in this setting is the variance σ^2 of Z,* i.e., *$L(w_\star) = \sigma^2$. Moreover, in this setting, we have $\alpha = \beta = \bar\beta = 2$, so the estimate $\hat w$ returned by Algorithm 2 satisfies, with probability at least $1-\delta$,*

$$L(\hat w) = \left(1 + O\Big(\frac{\log(1/\delta)}{n}\Big)\right)L(w_\star).$$

*In Remark 7 a result from Catoni (2012) is quoted which implies that if $n = o(1/\delta)$, then the empirical mean $\hat w_{\mathrm{emp}} := \arg\min_{w\in\mathbb{R}} L_S(w) = |S|^{-1}\sum_{z\in S} z$ (*i.e., *empirical risk (loss) minimization for this problem) incurs loss*

$$L(\hat w_{\mathrm{emp}}) = \sigma^2 + (\hat w_{\mathrm{emp}} - w_\star)^2 = (1+\omega(1))L(w_\star)$$

with probability at least 2δ. Therefore empirical risk minimization cannot provide a qualitatively similar guarantee as Corollary 16. It is easy to check that minimizing a regularized objective also does not work, since any non-trivial regularized objective necessarily provides an estimator with a positive error for some distribution with zero variance.

In the next section we use the analysis for general smooth and convex losses to derive new algorithms and bounds for linear regression.

5. Least Squares Linear Regression

In linear regression, the parameter space $\mathbb{X}$ is a Hilbert space with inner product $\langle\cdot,\cdot\rangle_{\mathbb{X}}$, and $\mathcal{Z} := \mathbb{X}\times\mathbb{R}$, where in the finite-dimensional case, $\mathbb{X} = \mathbb{R}^d$ for some finite integer d. The loss here is the squared loss, denoted by $\ell = \ell^{\mathrm{sq}}$, and defined as

$$\ell^{\mathrm{sq}}((\boldsymbol{x},y),\boldsymbol{w}) := \frac{1}{2}(\boldsymbol{x}^\top\boldsymbol{w} - y)^2.$$

The regularized squared loss, for $\lambda \ge 0$, is denoted

$$\ell^\lambda((\boldsymbol{x},y),\boldsymbol{w}) := \frac{1}{2}(\langle\boldsymbol{x},\boldsymbol{w}\rangle_{\mathbb{X}} - y)^2 + \frac{1}{2}\lambda\langle\boldsymbol{w},\boldsymbol{w}\rangle_{\mathbb{X}}.$$

Note that $\ell^0 = \ell^{\mathrm{sq}}$. We analogously define L^{sq}, L_T^{sq}, $L_\star^{\mathrm{sq}}$, L^λ, *etc.* as the squared-loss equivalents of $L, L_T, L_\star$. Finally, denote by Id the identity operator on $\mathbb{X}$.

The proposed algorithm for regression (Algorithm 4) is as follows. Set $k = C\log(1/\delta)$, where C is a universal constant. First, draw k independent random samples i.i.d. from $\mathcal{D}$, and perform linear regression with λ-regularization on each sample separately to obtain k linear regressors. Then, use the same k samples to generate k estimates of the covariance matrix of the marginal of $\mathcal{D}$ on the data space. Finally, use the estimated covariances to select a single regressor from among the k at hand. The slightly simpler variants of steps 4 and 5 can be used in some cases, as detailed below.

In Section 5.1, the full results for regression, mentioned in Section 2, are listed in full detail, and compared to previous work. The proofs are provided in Section 5.2.

5.1 Results

Let $\boldsymbol{X} \in \mathbb{X}$ be a random vector drawn according to the marginal of $\mathcal{D}$ on $\mathbb{X}$, and let $\Sigma : \mathbb{X}\to\mathbb{X}$ be the second-moment operator $\boldsymbol{a} \mapsto \mathbb{E}(\boldsymbol{X}\langle\boldsymbol{X},\boldsymbol{a}\rangle_{\mathbb{X}})$. For a finite-dimensional $\mathbb{X}$, Σ is simply the (uncentered) covariance matrix $\mathbb{E}[\boldsymbol{X}\boldsymbol{X}^\top]$. For a sample $T := \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_m\}$ of m independent copies of $\boldsymbol{X}$, denote by $\Sigma_T : \mathbb{X}\to\mathbb{X}$ the empirical second-moment operator $\boldsymbol{a} \mapsto m^{-1}\sum_{i=1}^m \boldsymbol{X}_i\langle\boldsymbol{X}_i,\boldsymbol{a}\rangle_{\mathbb{X}}$.

Consider first the finite-dimensional case, where $\mathbb{X} = \mathbb{R}^d$, and assume Σ is not singular. Let $\|\cdot\|_2$ denote the Euclidean norm in $\mathbb{R}^d$. In this case we obtain a guarantee for ordinary least squares with $\lambda = 0$. The guarantee holds whenever the empirical estimate of Σ is close to the true Σ *in expectation*, a mild condition that requires only bounded low-order moments. For concreteness, we assume the following condition.[5]

5. As shown by Srivastava and Vershynin (2013), Condition 1 holds for various heavy-tailed distributions (*e.g.*, when $\boldsymbol{X}$ has a product distribution with bounded $4+\epsilon$ moments for some $\epsilon > 0$). Condition 1 may be easily substituted with other moment conditions, yielding similar results, at least up to logarithmic factors.

Algorithm 4 Regression for heavy-tails

input $\lambda \geq 0$, sample size n, confidence $\delta \in (0,1)$.

output Approximate predictor $\hat{\boldsymbol{w}} \in \mathbb{X}$.

1: Set $k := \lceil C \ln(1/\delta) \rceil$.

2: Draw k random i.i.d. samples $S_1, \ldots, S_k$ from D, each of size $\lfloor n/k \rfloor$.

3: For each $i \in [k]$, let $\boldsymbol{w}_i \in \operatorname{argmin}_{\boldsymbol{w} \in \mathbb{X}} L^{\lambda}_{S_i}(\boldsymbol{w})$.

4: For each $i \in [k]$, $\Sigma_{S_i} \leftarrow \frac{1}{|S_i|} \sum_{(\boldsymbol{x},\cdot) \in S_i} \boldsymbol{x}\boldsymbol{x}^\top$.
[**Variant**: $S \leftarrow \cup_{i \in [k]} S_i$; $\Sigma_S \leftarrow \frac{1}{|S|} \sum_{(\boldsymbol{x},\cdot) \in S} \boldsymbol{x}\boldsymbol{x}^\top$].

5: For each $i \in [k]$, let r_i be the median of the values in

$$\{\langle \boldsymbol{w}_i - \boldsymbol{w}_j, (\Sigma_{S_j} + \lambda \operatorname{Id})(\boldsymbol{w}_i - \boldsymbol{w}_j) \rangle \mid j \in [k] \setminus \{i\}\}.$$

[**Variant**: Use Σ_S instead of Σ_{S_j}].

6: Set $i_\star := \arg\min_{i \in [k]} r_i$.

7: Return $\hat{\boldsymbol{w}} := \boldsymbol{w}_{i_\star}$.

Condition 1 (Srivastava and Vershynin 2013) *There exists $c, \eta > 0$ such that*

$$\Pr\left[\|\Pi \Sigma^{-1/2} \boldsymbol{X}\|_2^2 > t\right] \;\leq\; c t^{-1-\eta}, \quad \text{for } t > c \cdot \operatorname{rank}(\Pi)$$

for every orthogonal projection Π in $\mathbb{R}^d$.

Under this condition, we show the following guarantee for least squares regression.

Theorem 19 *Assume Σ is not singular. If $\boldsymbol{X}$ satisfies Condition 1 with some fixed parameters $c > 0$ and $\eta > 0$, then if Algorithm 4 is run with $n \geq O(d \log(1/\delta))$ and $\delta \in (0,1)$, with probability at least $1 - \delta$,*

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \leq L^{\mathrm{sq}}_\star + O\left(\frac{\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)\|_2^2 \log(1/\delta)}{n}\right).$$

Our loss bound is given in terms of the following population quantity

$$\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)\|_2^2 \tag{6}$$

which we assume is finite. This assumption only requires bounded low-order moments of $\boldsymbol{X}$ and Y and is essentially the same as the conditions from Audibert and Catoni (2011) (see the discussion following their Theorem 3.1). Define the following finite fourth-moment conditions:

$$\kappa_1 := \frac{\sqrt{\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}\|_2^4}}{\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}\|_2^2} = \frac{\sqrt{\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}\|_2^4}}{d} < \infty \quad \text{and}$$

$$\kappa_2 := \frac{\sqrt{\mathbb{E}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)^4}}{\mathbb{E}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)^2} = \frac{\sqrt{\mathbb{E}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)^4}}{L^{\mathrm{sq}}_\star} < \infty.$$

Under these conditions, $\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)\|_2^2 \leq \kappa_1 \kappa_2 d L^{\mathrm{sq}}_\star$ (via Cauchy-Schwartz); if κ_1 and κ_2 are constant, then we obtain the bound

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \leq \left(1 + O\left(\frac{d \log(1/\delta)}{n}\right)\right) L^{\mathrm{sq}}_\star$$

with probability $\geq 1 - \delta$. In comparison, the recent work of Audibert and Catoni (2011) proposes an estimator for linear regression based on optimization of a robust loss function which achieves essentially the same guarantee as Theorem 19 (with only mild differences in the moment conditions, see the discussion following their Theorem 3.1). However, that estimator depends on prior knowledge about the response distribution, and removing this dependency using Lepski's adaptation method (Lepski, 1991) may result in a suboptimal convergence rate. It is also unclear whether that estimator can be computed efficiently.

Other analyses for linear least squares regression and ridge regression by Srebro et al. (2010) and Hsu et al. (2014) consider specifically the empirical minimizer of the squared loss, and give sharp rates of convergence to $L^{\mathrm{sq}}_\star$. However, both of these require either boundedness of the loss or boundedness of the approximation error. In Srebro et al. (2010), the specialization of the main result to square loss includes additive terms of order $O(\sqrt{L(\boldsymbol{w}_\star) b \log(1/\delta)/n} + b \log(1/\delta)/n)$, where $b > 0$ is assumed to bound the square loss of any predictions almost surely. In Hsu et al. (2014), the convergence rate includes an additive term involving almost-sure bounds on the approximation error/non-subgaussian noise (The remaining terms are comparable to Eq. (9) for $\lambda = 0$, and Eq. (7) for $\lambda > 0$, up to logarithmic factors). The additional terms preclude multiplicative approximations to $L(\boldsymbol{w}_\star)$ in cases where the loss or approximation error is unbounded. In recent work, Mendelson (2014) proposes a more subtle 'small-ball' criterion for analyzing the performance of the risk minimizer. However, as evident from the lower bound in Remark 18, the empirical risk minimizer cannot obtain the same type of guarantees as our estimator.

The next result is for the case where there exists $R < \infty$ such that $\Pr[\boldsymbol{X}^\top \Sigma^{-1} \boldsymbol{X} \leq R^2] = 1$ (and, here, we do not assume Condition 1). In contrast, Y may still be heavy-tailed. Then, the following result can be derived using Algorithm 4. Moreover, the simpler **variant** of Algorithm 4 suffices here.

Theorem 20 *Assume Σ is not singular. Let $\hat{\boldsymbol{w}}$ be the output of the variant of Algorithm 4 with $\lambda = 0$. With probability at least $1 - \delta$, for $n \geq O(R^2 \log(R) \log(1/\delta))$,*

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \leq \left(1 + O\left(\frac{R^2 \log(2/\delta)}{n}\right)\right) L^{\mathrm{sq}}_\star.$$

Note that $\mathbb{E}(\boldsymbol{X}^\top \Sigma^{-1} \boldsymbol{X}) = \mathbb{E} \operatorname{tr}(\boldsymbol{X}^\top \Sigma^{-1} \boldsymbol{X}) = \operatorname{tr}(\operatorname{Id}) = d$, therefore $R = \Omega(\sqrt{d})$. If indeed $R = \Theta(\sqrt{d})$, then a total sample size of $O(d \log(d) \log(1/\delta))$ suffices to guarantee a constant factor approximation to the optimal loss. This is minimax optimal up to logarithmic factors (Nussbaum, 1999). We also remark that the boundedness assumption can be replaced by a subgaussian assumption on $\boldsymbol{X}$, in which case the sample size requirement becomes $O(d \log(1/\delta))$.

In recent work of Mahdavi and Jin (2013), an algorithm based on stochastic gradient descent obtains multiplicative approximations to $L_\star$, for general smooth and strongly convex

losses ℓ, with a sample complexity scaling with $\log(1/\tilde{L})$. Here, $\tilde{L}$ is an upper bound on $L_\star$, which must be known by the algorithm. The specialization of Mahdavi and Jin's main result to square loss implies a sample complexity of $\tilde{O}(dR^8\log(1/(\delta L_\star^{\mathrm{sq}}))$ if $L_\star^{\mathrm{sq}}$ is known. In comparison, Theorem 20 shows that $\tilde{O}(R^2\log(1/\delta))$ suffice when using our estimator. It would be interesting to understand whether the bound for the stochastic gradient method of Mahdavi and Jin (2013) can be improved, and whether knowledge of $L_\star$ is actually necessary in the stochastic oracle model. We note that the main result of Mahdavi and Jin (2013) can be more generally applicable than Theorem 15, because Mahdavi and Jin (2013) only assumes that the population loss $L(\boldsymbol{w})$ is strongly convex, whereas Theorem 15 requires the empirical loss $L_T(\boldsymbol{w})$ to be strongly convex for large enough samples T. While our technique is especially simple for the squared loss, it may be more challenging to implement well for other losses, because the local norm around $\boldsymbol{w}_\star$ may be difficult to approximate with an observable norm. We thus leave the extension to more general losses as future work.

Finally, we also consider the case where $\mathbb{X}$ is a general, infinite-dimensional Hilbert space, $\lambda > 0$, the norm of $\boldsymbol{X}$ is bounded, and Y again may be heavy-tailed.

Theorem 21 *Let $V > 0$ such that $\Pr[\langle \boldsymbol{X}, \boldsymbol{X}\rangle_{\mathbb{X}} \le V^2] = 1$. Let $\hat{\boldsymbol{w}}$ be the output of the variant of Algorithm 4 with $\lambda > 0$. With probability at least $1-\delta$, as soon as $n \ge O((V^2/\lambda)\log(V/\sqrt{\lambda})\log(2/\delta))$,*

$$L^{\lambda}(\hat{\boldsymbol{w}}) \le \left(1 + O\left(\frac{(1+V^2/\lambda)\log(2/\delta)}{n}\right)\right) L_\star^{\lambda}.$$

If the optimal unregularized squared loss $L_\star^{\mathrm{sq}}$ is achieved by $\bar{\boldsymbol{w}} \in \mathbb{X}$ with $\langle \bar{\boldsymbol{w}}, \bar{\boldsymbol{w}}\rangle_{\mathbb{X}} \le B^2$, the choice $\lambda = \Theta(\sqrt{L_\star^{\mathrm{sq}} V^2 \log(2/\delta)/(B^2 n)})$ yields that if $n \ge \tilde{O}(B^2V^2\log(2/\delta)/L_\star^{\mathrm{sq}})$ then

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) \le L_\star^{\mathrm{sq}} + O\left(\sqrt{\frac{L_\star^{\mathrm{sq}} B^2 V^2 \log(1/\delta)}{n}} + \frac{(L_\star^{\mathrm{sq}} + B^2V^2)\log(1/\delta)}{n}\right). \tag{7}$$

By this analysis, a constant factor approximation for $L_\star^{\mathrm{sq}}$ is achieved with a sample of size $\tilde{O}(B^2V^2\log(1/\delta)/L_\star^{\mathrm{sq}})$. As in the finite-dimensional setting, this rate is known to be optimal up to logarithmic factors (Nussbaum, 1999). It is interesting to observe that in the non-parametric case, our analysis, like previous analyses, does require knowledge of $L_\star$ if λ is to be set correctly, as in Mahdavi and Jin (2013).

5.2 Analysis

We now show how the analysis of Section 4 can be applied to analyze Algorithm 4. For a sample $T \subseteq \mathcal{Z}$, if L_T is twice-differentiable (which is the case for squared loss), by Taylor's theorem, for any $\boldsymbol{w} \in \mathbb{X}$, there exist $t \in [0,1]$ and $\tilde{\boldsymbol{w}} = t\boldsymbol{w}_\star + (1-t)\boldsymbol{w}$ such that

$$L_T(\boldsymbol{w}) = L_T(\boldsymbol{w}_\star) + \langle \nabla L_T(\boldsymbol{w}_\star), \boldsymbol{w} - \boldsymbol{w}_\star\rangle_{\mathbb{X}} + \frac{1}{2}\langle \boldsymbol{w} - \boldsymbol{w}_\star, \nabla^2 L_T(\tilde{\boldsymbol{w}})(\boldsymbol{w} - \boldsymbol{w}_\star)\rangle_{\mathbb{X}},$$

Therefore, to establish a bound on n_α, it suffices to control

$$\Pr\left[\inf_{\boldsymbol{\delta}\in\mathbb{X}\setminus\{\boldsymbol{0}\}, \tilde{\boldsymbol{w}}\in\mathbb{R}^d} \frac{\langle \boldsymbol{\delta}, \nabla^2 L_T(\tilde{\boldsymbol{w}})\boldsymbol{\delta}\rangle_{\mathbb{X}}}{\|\boldsymbol{\delta}\|^2} \ge \alpha\right] \tag{8}$$

for an i.i.d. sample T from $\mathcal{D}$. The following lemma allows doing just that.

Lemma 22 (Specialization of Lemma 1 from Oliveira 2010) *Fix any $\lambda \ge 0$, and assume $\langle \boldsymbol{X}, (\Sigma + \lambda\,\mathrm{Id})^{-1}\boldsymbol{X}\rangle_{\mathbb{X}} \le r_\lambda^2$ almost surely. For any $\delta \in (0,1)$, if $m \ge 80 r_\lambda^2 \ln(4m^2/\delta)$, then with probability at least $1-\delta$, for all $\boldsymbol{a} \in \mathbb{X}$,*

$$\frac{1}{2}\langle \boldsymbol{a}, (\Sigma + \lambda\,\mathrm{Id})\boldsymbol{a}\rangle_{\mathbb{X}} \le \langle \boldsymbol{a}, (\Sigma_T + \lambda\,\mathrm{Id})\boldsymbol{a}\rangle_{\mathbb{X}} \le 2\langle \boldsymbol{a}, (\Sigma + \lambda\,\mathrm{Id})\boldsymbol{a}\rangle_{\mathbb{X}}.$$

We use the boundedness assumption for sake of simplicity; it is possible to remove the boundedness assumption, and the logarithmic dependence on the cardinality of T, under different conditions on $\boldsymbol{X}$ (*e.g.*, assuming $\Sigma^{-1/2}\boldsymbol{X}$ has subgaussian projections, see Litvak et al. 2005). We now prove Theorem 20, Theorem 21 and Theorem 19.

5.2.1 Ordinary Least Squares in Finite Dimensions

Consider first ordinary least squares in the finite-dimensional case. In this case $\mathbb{X} = \mathbb{R}^d$, the inner product $\langle \boldsymbol{a}, \boldsymbol{b}\rangle_{\mathbb{X}} = \boldsymbol{a}^\top\boldsymbol{b}$ is the usual coordinate dot product, and the second-moment operator is $\Sigma = \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^\top)$. We assume that Σ is non-singular, so L has a unique minimizer. Here Algorithm 4 can be used with $\lambda = 0$. It is easy to see that Algorithm 4 with the **variant** steps is a specialization of Algorithm 2 with subsampled empirical loss minimization when $\ell = \ell^{\mathrm{sq}}$, with the norm defined by $\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a}^\top\Sigma_S\boldsymbol{a}}$. We now prove the guarantee for finite dimensional regression.

Proof [of Theorem 20] The proof is derived from Corollary 16 as follows. First, suppose for simplicity that $\Sigma_s = \Sigma$, so that $\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a}^\top\Sigma\boldsymbol{a}}$. It is easy to check that $\|\cdot\|_*$ is 1-smooth, ℓ is R^2-smooth with respect to $\|\cdot\|$, and L^{sq} is 1-smooth with respect to $\|\cdot\|$. Moreover, consider a random sample T. By definition

$$\frac{\boldsymbol{\delta}^\top\nabla^2 L_T(\tilde{\boldsymbol{w}})\boldsymbol{\delta}}{\|\boldsymbol{\delta}\|^2} = \frac{\boldsymbol{\delta}^\top\Sigma_T\boldsymbol{\delta}}{\boldsymbol{\delta}^\top\Sigma\boldsymbol{\delta}}.$$

By Lemma 22 with $\lambda = 0$, $\Pr[\inf\{\boldsymbol{\delta}^\top\Sigma_T\boldsymbol{\delta}/(\boldsymbol{\delta}^\top\Sigma\boldsymbol{\delta}) : \boldsymbol{\delta} \in \mathbb{R}^d \setminus \{\boldsymbol{0}\}\} \ge 1/2] \ge 5/6$, provided that $|T| \ge 80R^2\log(24|S|^2)$. Therefore $n_{0.5} = O(R^2\log R)$. We can thus apply Corollary 16 with $\alpha = 0.5$, $\beta = R^2$, $\bar{\beta} = 1$, $\gamma = 1$, and $n_{0.5} = O(R^2\log R)$, so with probability at least $1-\delta$, the parameter $\hat{\boldsymbol{w}}$ returned by Algorithm 4 satisfies

$$L(\hat{\boldsymbol{w}}) \le \left(1 + O\left(\frac{R^2\log(1/\delta)}{n}\right)\right) L(\boldsymbol{w}_\star), \tag{9}$$

as soon as $n \ge O(R^2\log(R)\log(1/\delta))$.

Now, by Lemma 22, if $n \ge O(R^2\log(R/\delta))$, with probability at least $1-\delta$, the norm induced by Σ_S satisfies $(1/2)\boldsymbol{a}^\top\Sigma\boldsymbol{a} \le \boldsymbol{a}^\top\Sigma_S\boldsymbol{a} \le 2\boldsymbol{a}^\top\Sigma\boldsymbol{a}$ for all $\boldsymbol{a} \in \mathbb{R}^d$. Therefore, by a union bound, the norm used by the algorithm is equivalent to the norm induced by the true Σ up to constant factors, and thus leads to the same guarantee as given above (where the constant factors are absorbed into the big-O notation). ■

The rate achieved in Eq. (9) is well-known to be optimal up to logarithmic factors (Nussbaum, 1999). A standard argument for this, which we reference in the sequel, is as follows. Consider a distribution over $\mathbb{R}^d \times \mathbb{R}$ where $\boldsymbol{X} \in \mathbb{R}^d$ is distributed uniformly over some

orthonormal basis vectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_d$, and $Y := \boldsymbol{X}^\top \boldsymbol{w}_\star + Z$ for $Z \sim \mathcal{N}(0, \sigma^2)$ independent of $\boldsymbol{X}$. Here, $\boldsymbol{w}_\star$ is an arbitrary vector in $\mathbb{R}^d$, $R = \sqrt{d}$, and the optimal square loss is $L(\boldsymbol{w}_\star) = \sigma^2$. Among n independent copies of $(\boldsymbol{X}, Y)$, let n_i be the number of copies with $\boldsymbol{X} = \boldsymbol{e}_i$, so $\sum_{i=1}^d n_i = n$. Estimating $\boldsymbol{w}_\star$ is equivalent to d Gaussian mean estimation problems, with a minimax loss of

$$\inf_{\hat{\boldsymbol{w}}} \sup_{\boldsymbol{w}_\star} \mathbb{E}\big(L(\hat{\boldsymbol{w}})\big) - L(\boldsymbol{w}_\star) \;=\; \inf_{\hat{\boldsymbol{w}}} \sup_{\boldsymbol{w}_\star} \mathbb{E}\left(\frac{1}{d}\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_2^2\right)$$
$$= \frac{1}{d}\sum_{i=1}^d \frac{\sigma^2}{n_i} \;\geq\; \frac{d\sigma^2}{n} \;=\; \frac{dL(\boldsymbol{w}_\star)}{n}. \tag{10}$$

Note that this also implies a lower bound for any estimator with exponential concentration. That is, for any estimator $\hat{\boldsymbol{w}}$, if there is some $A > 0$ such that for any $\delta \in (0,1)$, $\mathbb{P}[L(\hat{\boldsymbol{w}}) > L(\boldsymbol{w}_\star) + A\log(1/\delta)] < \delta$, then $A \geq \mathbb{E}(L(\hat{\boldsymbol{w}}) - L(\boldsymbol{w}_\star)) \geq dL(\boldsymbol{w}_\star)/n$.

5.2.2 Ridge Regression

In a general, possibly infinite-dimensional, Hilbert space $\mathbb{X}$, Algorithm 4 can be used with $\lambda > 0$. In this case, Algorithm 4 with the **variant** steps is again a specialization of Algorithm 2 with subsampled empirical loss minimization when $\ell = \ell^\lambda$, with the norm defined by $\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a}^\top(\Sigma_S + \lambda \mathrm{Id})\boldsymbol{a}}$.

Proof [of Theorem 21] As in the finite-dimensional case, assume first that $\Sigma_S = \Sigma$, and consider the norm $\|\cdot\|$ defined by $\|\boldsymbol{a}\| := \sqrt{\langle \boldsymbol{a}, (\Sigma + \lambda \mathrm{Id})\boldsymbol{a}\rangle_{\mathbb{X}}}$. It is easy to check that $\|\cdot\|_*$ is 1-smooth. Moreover, since we assume that $\Pr[\langle \boldsymbol{X}, \boldsymbol{X}\rangle_{\mathbb{X}} \leq V^2] = 1$, we have $\langle \boldsymbol{x}, (\Sigma + \lambda I)^{-1}\boldsymbol{x}\rangle_{\mathbb{X}} \leq \langle \boldsymbol{x}, \boldsymbol{x}\rangle_{\mathbb{X}}/\lambda$ for all $\boldsymbol{x} \in \mathbb{X}$, so $\Pr[\langle \boldsymbol{X}, (\Sigma + \lambda I)^{-1}\boldsymbol{X}\rangle_{\mathbb{X}} \leq V^2/\lambda] = 1$. Therefore ℓ^λ is $(1 + V^2/\lambda)$-smooth with respect to $\|\cdot\|$. In addition, L^λ is 1-smooth with respect to $\|\cdot\|$. Using Lemma 22 with $r_\lambda = V/\lambda$, we have, similarly to the proof of Theorem 20, $n_{0.5} = O((V^2/\lambda)\log(V/\sqrt{\lambda}))$. Setting $\alpha = 0.5$, $\beta = 1 + V^2/\lambda$, $\bar{\beta} = 1$, $\gamma = 1$, and $n_{0.5}$ as above, we conclude that with probability $1 - \delta$,

$$L^\lambda(\hat{\boldsymbol{w}}) \leq \left(1 + O\left(\frac{(1 + V^2/\lambda)\log(1/\delta)}{n}\right)\right) L^\lambda(\boldsymbol{w}_\star),$$

as soon as $n \geq O((V^2/\lambda)\log(V/\sqrt{\lambda})\log(1/\delta))$. Again as in the proof of Theorem 20, by Lemma 22 Algorithm 4 may use the observable norm $\boldsymbol{a} \mapsto \langle \boldsymbol{a}, (\Sigma_S + \lambda I)\boldsymbol{a}\rangle_{\mathbb{X}}^{1/2}$ instead of the unobservable norm $\boldsymbol{a} \mapsto \langle \boldsymbol{a}, (\Sigma + \lambda I)\boldsymbol{a}\rangle_{\mathbb{X}}^{1/2}$ by applying a union bound, if $n \geq O((V^2/\lambda)\log(2V/(\delta\sqrt{\lambda})))$, losing only constant factors, .

We are generally interested in comparing to the minimum square loss $L_\star^{\mathrm{sq}} := \inf_{\boldsymbol{w}\in\mathbb{X}} L^{\mathrm{sq}}(\boldsymbol{w})$, rather than the minimum regularized square loss $\inf_{\boldsymbol{w}\in\mathbb{X}} L^\lambda(\boldsymbol{w})$. Assuming the minimizer is achieved by some $\bar{\boldsymbol{w}} \in \mathbb{X}$ with $\langle \bar{\boldsymbol{w}}, \bar{\boldsymbol{w}}\rangle_{\mathbb{X}} \leq B^2$, the choice $\lambda = \Theta(\sqrt{L_\star^{\mathrm{sq}} V^2 \log(2/\delta)/(B^2 n)})$ yields

$$L^{\mathrm{sq}}(\hat{\boldsymbol{w}}) + \lambda\langle \hat{\boldsymbol{w}}, \hat{\boldsymbol{w}}\rangle_{\mathbb{X}} \leq L_\star^{\mathrm{sq}} + O\left(\sqrt{\frac{L_\star^{\mathrm{sq}} B^2 V^2 \log(2/\delta)}{n}} + \frac{(L_\star^{\mathrm{sq}} + B^2V^2)\log(2/\delta)}{n}\right)$$

as soon as $n \geq \tilde{O}(B^2V^2\log(2/\delta)/L_\star^{\mathrm{sq}})$. ∎

By this analysis, a constant factor approximation for $L_\star^{\mathrm{sq}}$ is achieved with a sample of size $\tilde{O}(B^2V^2\log(1/\delta)/L_\star^{\mathrm{sq}})$. As in the finite-dimensional setting, this rate is known to be optimal up to logarithmic factors (Nussbaum, 1999). Indeed, a similar construction to that from Section 5.2.1 implies

$$\inf_{\hat{\boldsymbol{w}}} \sup_{\boldsymbol{w}_\star} \mathbb{E}\big(L(\hat{\boldsymbol{w}}) - L(\boldsymbol{w}_\star)\big) \;\geq\; \Omega\left(\frac{1}{d}\cdot\frac{L_\star B^2V^2\sum_{i=1}^d n_i^{-1}}{B^2V^2 + L_\star\sum_{i=1}^d n_i^{-1}}\right) \;\geq\; \Omega\left(\frac{1}{d}\cdot\frac{L_\star B^2V^2 d^2/n}{B^2V^2 + L_\star d^2/n}\right) \tag{11}$$

(here, $\boldsymbol{X} \in \{V\boldsymbol{e}_i : i \in [d]\}$ has Euclidean length V almost surely, and B is a bound on the Euclidean length of $\boldsymbol{w}_\star$). For $d = \sqrt{B^2V^2n/\sigma^2}$, the bound becomes

$$\inf_{\hat{\boldsymbol{w}}} \sup_{\boldsymbol{w}_\star} \mathbb{E}\big(L(\hat{\boldsymbol{w}}) - L(\boldsymbol{w}_\star)\big) \;\geq\; \Omega\left(\sqrt{\frac{L_\star B^2V^2}{n}}\right).$$

As before, this minimax bound also implies a lower bound on any estimator with exponential concentration.

5.2.3 Heavy-tail Covariates

When the covariates are not bounded or subgaussian, the empirical second-moment matrix may deviate significantly from its population counterpart with non-negligible probability. In this case it is not possible to approximate the norm $\|\boldsymbol{a}\| = \sqrt{\boldsymbol{a}^\top(\Sigma + \lambda\mathrm{Id})\boldsymbol{a}}$ in Step 2 of Algorithm 2 using a single small sample (as discussed in Section 5.2.1 and Section 5.2.2). However, we may use Algorithm 3 instead of Algorithm 2, which only requires the stochastic distance measurements to be relatively accurate with some constant probability. The full version of Algorithm 4 is exactly such an implementation.

We now prove Theorem 19. Define $c_\eta := 512(48c)^{2+2/\eta}(6 + 6/\eta)^{1+4/\eta}$ (which is C_{main} from Srivastava and Vershynin, 2013). The following lemma shows that $O(d)$ samples suffice so that the expected spectral norm distance between the empirical second-moment matrix and Σ is bounded.

Lemma 23 (Implication of Corollary 1.2 from Srivastava and Vershynin, 2013) *Let $\boldsymbol{X}$ satisfy Condition 1, and let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent copies of $\boldsymbol{X}$. Let $\widehat{\Sigma} := \frac{1}{n}\sum_{i=1}^n \boldsymbol{X}_i\boldsymbol{X}_i^\top$. For any $\epsilon \in (0,1)$, if $n \geq c_\eta\epsilon^{-2-2/\eta}d$, then*

$$\mathbb{E}\|\Sigma^{-1/2}\widehat{\Sigma}\Sigma^{-1/2} - \mathrm{Id}\,\|_2 \;\leq\; \epsilon.$$

Lemma 23 implies that $n_{0.5} = O(c'_\eta d)$ where $c'_\eta = c_\eta \cdot 2^{O(1+1/\eta)}$. Therefore, for $k = O(\log(1/\delta))$, subsampled empirical loss minimization requires $n \geq k \cdot n_{0.5} = O(c'_\eta d\log(1/\delta))$ samples to correctly implement $\mathrm{APPROX}_{\|\cdot\|,\varepsilon}$, for ε as in Eq. (5).

Step 5 in Algorithm 4 implements $\mathrm{DIST}^j_{\|\cdot\|}$ as returning f_j such that $f_j(\boldsymbol{v}) := \|\Sigma_{S_j}^{1/2}(\boldsymbol{v} - \boldsymbol{w}_j)\|_2$. First, we show that Assumption 3 holds. By Lemma 23, an i.i.d. sample T of size $O(c'_\eta d)$ suffices so that with probability at least 8/9, for every $\boldsymbol{v} \in \mathbb{R}^d$,

$$(1/2)\|\Sigma^{1/2}(\boldsymbol{v} - \boldsymbol{w}_j)\|_2 \leq \|\Sigma_T^{1/2}(\boldsymbol{v} - \boldsymbol{w}_j)\|_2 \leq 2\|\Sigma^{1/2}(\boldsymbol{v} - \boldsymbol{w}_j)\|_2.$$

In particular, this holds for $T = S_j$, as long as $|S_j| \geq O(c'_\eta d)$. Thus, for $k = O(\log(1/\delta))$, Assumption 3 holds if $n \geq O(c'_\eta d \log(1/\delta))$. Assumption 4 (independence) also holds, since f_j depends only on S_j, and $S_1, \ldots, S_k$ are statistically independent.

Putting everything together, we have (as in Section 5.2.1) $\alpha = 0.5$ and $\gamma = 1$. We obtain the final bound from Theorem 17 as follows: if $n \geq O(c'_\eta d \log(1/\delta))$, then with probability at least $1 - \delta$,

$$L(\hat{\boldsymbol{w}}) - L(\boldsymbol{w}_\star) = \|\Sigma^{1/2}(\hat{\boldsymbol{w}} - \boldsymbol{w}_\star)\|_2^2 \leq O\left(\frac{\mathbb{E}\|\Sigma^{-1/2}\boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{w}_\star - Y)\|_2^2 \log(1/\delta)}{n}\right). \quad (12)$$

6. Other Applications

In this section we show how the core techniques we discuss can be used for other applications, namely Lasso and low-rank matrix approximation.

6.1 Sparse Parameter Estimation with Lasso

In this section we consider L^1-regularized linear least squared regression (Lasso) (Tibshirani, 1996) with a random subgaussian design, and show that Algorithm 2 achieves the same fast convergence rates for sparse parameter estimation as Lasso, even when the noise is heavy-tailed.

Let $\mathcal{Z} = \mathbb{R}^d \times \mathbb{R}$ and $\boldsymbol{w}_\star \in \mathbb{R}^d$. Let D be a distribution over $\mathcal{Z}$, such that for $(\boldsymbol{X}, Y) \sim D$, we have $Y = \boldsymbol{X}^\top \boldsymbol{w}_\star + \varepsilon$ where ε is an independent random variable with $\mathbb{E}[\varepsilon] = 0$ and $\mathbb{E}[\varepsilon^2] \leq \sigma^2$. We assume that $\boldsymbol{w}_\star$ is sparse: Denote the support of a vector $\boldsymbol{w}$ by $\operatorname{supp}(\boldsymbol{w}) := \{j \in [d] : \boldsymbol{w}_j \neq 0\}$. Then $s := |\operatorname{supp}(\boldsymbol{w}_\star)|$ is assumed to be small compared to d. The *design matrix* for a sample $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ is an $l \times d$ matrix with the rows $\boldsymbol{x}_i^\top$.

For $\lambda > 0$, consider the Lasso loss $\ell((\boldsymbol{x}, y), \boldsymbol{w}) = \frac{1}{2}(\boldsymbol{x}^\top \boldsymbol{w} - y)^2 + \lambda\|\boldsymbol{w}\|_1$. Let $\|\cdot\|$ be the Euclidean norm in $\mathbb{R}^d$. A random vector $\boldsymbol{X}$ in $\mathbb{R}^d$ is *subgaussian* (with moment 1) if for every vector $\boldsymbol{u} \in \mathbb{R}^d$, $\mathbb{E}[\exp(\boldsymbol{X}^\top \boldsymbol{u})] \leq \exp(\|\boldsymbol{u}\|_2^2/2)$.

The following theorem shows that when Algorithm 2 is used with subsampled empirical loss minimization over the Lasso loss, and D generates a subgaussian random design, then $\boldsymbol{w}$ can be estimated for any type of noise ε, including heavy-tailed noise.

In order to obtain guarantees for Lasso the design matrix must satisfy some regularity conditions. We use the *Restricted Eigenvalue condition* (RE) proposed in Bickel et al. (2009), which we presently define. For $\boldsymbol{w} \in \mathbb{R}^d$ and $J \subseteq [d]$, let $[\boldsymbol{w}]_J$ be the $|J|$-dimensional vector which is equal to $\boldsymbol{w}$ on the coordinates in J. Denote by $\boldsymbol{w}_{[s]}$ the s-dimensional vector with coordinates equal to the s largest coordinates (in absolute value) of $\boldsymbol{w}$. Let $\boldsymbol{w}_{[s]^C}$ be the $(d-s)$-dimensional vector which includes the coordinates not in $\boldsymbol{w}_{[s]}$. Define the set $E_s = \{\boldsymbol{u} \in \mathbb{R}^d \setminus \{0\} \mid \|\boldsymbol{u}_{[s]^C}\|_1 \leq 3\|\boldsymbol{u}_{[s]}\|_1\}$. For an $l \times d$ matrix Ψ (for some integer l), let $\gamma(\Psi, s) = \min_{\boldsymbol{u} \in E_s} \frac{\|\Psi \boldsymbol{u}\|_2}{\|\boldsymbol{u}_{[s]}\|_2}$. The RE condition for Ψ with sparsity s requires that $\gamma(\Psi, s) > 0$. We further denote $\eta(\Psi, s) = \max_{\boldsymbol{u} \in \mathbb{R}^d \setminus \{0\} : |\operatorname{supp}(\boldsymbol{u})| \leq s} \frac{\|\Psi \boldsymbol{u}\|_2}{\|\boldsymbol{u}\|_2}$.

Theorem 24 *Let $C, c > 0$ be universal constants. Let $\Sigma \in \mathbb{R}^{d \times d}$ be a positive semi definite matrix. Denote $\eta := \eta(\Sigma^{\frac{1}{2}}, s)$ and $\gamma := \gamma(\Sigma^{\frac{1}{2}}, s)$. Assume the random design setting defined above, with $\boldsymbol{X} = \Sigma^{\frac{1}{2}}\boldsymbol{Z}$, where $\boldsymbol{Z}$ is a subgaussian random vector. Suppose Algorithm 2 uses subsampled empirical loss minimization with the empirical Lasso loss, with $\lambda = 2\sqrt{\sigma^2 \eta^2 \log(2d)\log(1/\delta)/n}$. If $n \geq cs\frac{\eta^2}{\gamma^2}\log(d)\log(1/\delta)$, then with probability $1 - \delta$, The vector $\hat{\boldsymbol{w}}$ returned by Algorithm 2 satisfies*

$$\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_2 \leq \frac{C\sigma\eta}{\gamma^2}\sqrt{\frac{s\log(2d)\log(1/\delta)}{n}}.$$

For the proof of Theorem 24, we use the following theorem, adapted from Bickel et al. (2009) and Zhang (2009). The proof is provided in Appendix A for completeness.

Theorem 25 (Bickel et al. (2009); Zhang (2009)) *Let $\Psi = [\Psi_1|\Psi_2|\ldots|\Psi_d] \in \mathbb{R}^{n \times d}$ and $\boldsymbol{\varepsilon} \in \mathbb{R}^n$. Let $y = \Psi \boldsymbol{w}_\star + \boldsymbol{\varepsilon}$ and $\hat{\boldsymbol{w}} \in \operatorname{argmin}_{\boldsymbol{w}} \frac{1}{2}\|\Psi \boldsymbol{w} - y\|_2^2 + \lambda\|\boldsymbol{w}\|_1$. Assume that $|\operatorname{supp}(\boldsymbol{w}_\star)| = s$ and that $\gamma(\Psi, s) > 0$. If $\|\Psi^\top \boldsymbol{\varepsilon}\|_\infty \leq \lambda/2$, then*

$$\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_2 \leq \frac{12\lambda\sqrt{s}}{\gamma^2(\Psi, s)}.$$

Proof [of Theorem 24] Fix $i \in [k]$, and let $n_i = n/k$. Let $\Psi \in \mathbb{R}^{n_i \times d}$ be the design matrix for S_i and let $\boldsymbol{w}_i$ be the vector returned by the algorithm in round i, $\boldsymbol{w}_i \in \operatorname{argmin} \frac{1}{2n}\|\Psi \boldsymbol{w} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{w}\|_1$. It is shown in Zhou (2009) that if $n_i \geq C\frac{\eta^2}{\gamma^2}s\log(d)$ for a universal constant C, then with probability 5/6, $\min_{\boldsymbol{u} \in E_s} \frac{\|\Psi \boldsymbol{u}\|_2}{\|\Sigma^{\frac{1}{2}}\boldsymbol{u}\|_2} \geq \sqrt{n_i}/2$. Call this event $\mathcal{E}$. By the definition of γ, we have that under $\mathcal{E}$,

$$\gamma(\Psi, s) = \min_{\boldsymbol{u} \in E_s} \frac{\|\Psi \boldsymbol{u}\|_2}{\|\boldsymbol{u}_{[s]}\|_2} = \min_{\boldsymbol{u} \in E_s} \frac{\|\Psi \boldsymbol{u}\|_2}{\|\Sigma^{\frac{1}{2}}\boldsymbol{u}\|_2}\frac{\|\Sigma^{\frac{1}{2}}\boldsymbol{u}\|_2}{\|\boldsymbol{u}_{[s]}\|_2} \geq \sqrt{n}\gamma/2.$$

If $\mathcal{E}$ holds and $\|\Psi^\top \boldsymbol{\varepsilon}\|_\infty \leq n\lambda/2$, then we can apply Theorem 25 (with $n\lambda$ instead of λ). We now show that this inequality holds with a constant probability. Fix the noise vector $\boldsymbol{\varepsilon} = \boldsymbol{y} - \Psi \boldsymbol{w}_\star$. For $l \in [d]$, since the coordinates of $\boldsymbol{\varepsilon}$ are independent and each row of Ψ is an independent copy of the vector $\boldsymbol{X} = \Sigma^{\frac{1}{2}}\boldsymbol{Z}$, we have

$$\mathbb{E}[\exp([\Psi^\top \boldsymbol{\varepsilon}]_l) \mid \boldsymbol{\varepsilon}] = \prod_{j \in [n]} \mathbb{E}[\exp(\Psi_{j,l}\boldsymbol{\varepsilon}_j) \mid \boldsymbol{\varepsilon}] = \prod_{j \in [n]} \mathbb{E}[\exp(\boldsymbol{Z}(\boldsymbol{\varepsilon}_j \Sigma^{\frac{1}{2}}\boldsymbol{e}_l)) \mid \boldsymbol{\varepsilon}].$$

Since $\|\boldsymbol{\varepsilon}_j \Sigma^{\frac{1}{2}}\boldsymbol{e}_l\|_2 \leq \boldsymbol{\varepsilon}_j \eta$, we conclude that

$$\mathbb{E}[\exp([\Psi^\top \boldsymbol{\varepsilon}]_l) \mid \boldsymbol{\varepsilon}] \leq \prod_{j \in [n]} \exp(\boldsymbol{\varepsilon}_j^2/2) = \exp(\eta^2\|\boldsymbol{\varepsilon}\|_2^2/2).$$

Therefore, for $\xi > 0$

$$\begin{aligned}
\xi\mathbb{E}[\|\Psi^\top \boldsymbol{\varepsilon}\|_\infty \mid \boldsymbol{\varepsilon}] &= \mathbb{E}[\max_l(\xi|[\Psi^\top \boldsymbol{\varepsilon}]_l|) \mid \boldsymbol{\varepsilon}] = \mathbb{E}[\log \max_l \exp(\xi|[\Psi^\top \boldsymbol{\varepsilon}]_l|) \mid \boldsymbol{\varepsilon}] \\
&\leq \mathbb{E}[\log\left(\sum_l \exp(\xi[\Psi^\top \boldsymbol{\varepsilon}]_l) + \exp(-\xi[\Psi^\top \boldsymbol{\varepsilon}]_l)\right) \mid \boldsymbol{\varepsilon}] \\
&\leq \log\left(\sum_l \mathbb{E}[\exp(\xi[\Psi^\top \boldsymbol{\varepsilon}]_l) \mid \boldsymbol{\varepsilon}] + \mathbb{E}[\exp(-\xi[\Psi^\top \boldsymbol{\varepsilon}]_l) \mid \boldsymbol{\varepsilon}]\right) \\
&\leq \log(2d) + \xi^2\eta^2\|\boldsymbol{\varepsilon}\|_2^2/2.
\end{aligned}$$

Since $\mathbb{E}[\boldsymbol{\varepsilon}_j^2] \le \sigma^2$ for all j, we have $\mathbb{E}[\|\boldsymbol{\varepsilon}\|^2] \le n_i\sigma^2/2$. Therefore

$$\mathbb{E}[\|\Psi^\top \boldsymbol{\varepsilon}\|_\infty] \le \frac{\log(2d)}{\xi} + \xi n_i \eta^2 \sigma^2/2.$$

Minimizing over $\xi > 0$ we get $\mathbb{E}[\|\Psi^\top\boldsymbol{\varepsilon}\|_\infty] \le 2\sqrt{\sigma^2\eta^2\log(2d)n_i/2}$. therefore by Markov's inequality, with probability at least 5/6, $\frac{1}{n_i}\|\Psi^\top\boldsymbol{\varepsilon}\|_\infty \le 2\sqrt{\sigma^2\eta^2\log(2d)/n_i} = \lambda$. With probability at least 2/3 this holds together with $\mathcal{E}$.

In this case, by Theorem 25,

$$\|\boldsymbol{w}_i - \boldsymbol{w}_\star\|_2 \le \frac{12\lambda\sqrt{s}}{\gamma^2(\Psi,s)} \le \frac{24}{\gamma^2}\sqrt{\frac{s\sigma^2\eta^2\log(2d)}{n_i}}.$$

Therefore $\mathrm{APPROX}_{\|\cdot\|,\epsilon}$ satisfies Assumption 1 with ϵ as in the right hand side above. The statement of the theorem now follows by applying Proposition 9 with $k = O(\log(1/\delta)$, and noting that $n_i = O(n/\log(1/\delta))$. ■

It is worth mentioning that we can apply our technique to the fixed design setting, where design matrix $X \in \mathbb{R}^{n\times d}$ is fixed and not assumed to come from any distribution. If X satisfies the RE condition, as well as a certain low-leverage condition—specifically, that the *statistical leverage scores* (Chatterjee and Hadi, 1986) of any $n \times O(s)$ submatrix of X be roughly $O(1/(ks\log d))$—then Algorithm 2 can be used with the subsampled empirical loss minimization implementation of $\mathrm{APPROX}_{\|\cdot\|,\varepsilon}$ to obtain similar guarantees as in the random subgaussian design setting.

We note that while standard analyses of sparse estimation with mean-zero noise assume light-tailed noise (Zhang, 2009; Bickel et al., 2009), there are several works that analyze sparse estimation with heavy-tailed noise under various assumptions. For example, several works assume that the median of the noise is zero (*e.g.*, Wang 2013; Belloni and Chernozhukov 2011; Zou and Yuan 2008; Wu and Liu 2009; Wang et al. 2007; Fan et al. 2012). van de Geer and Müller (2012) analyze a class of optimization functions that includes the Lasso and show polynomial convergence under fourth-moment bounds on the noise. Chatterjee and Lahiri (2013) study a two-phase sparse estimator for mean-zero noise termed the Adaptive Lasso, proposed in Zou (2006), and show asymptotic convergence results under mild moment assumptions on the noise.

6.2 Low-rank Matrix Approximation

The proposed technique can be easily applied also to low-rank covariance matrix approximation for heavy tailed distributions. Let $\mathcal{D}$ be a distribution over $\mathcal{Z} = \mathbb{R}^d$ and suppose our goal is to estimate $\Sigma = \mathbb{E}[\boldsymbol{X}\boldsymbol{X}^\top]$ to high accuracy, assuming that Σ is (approximately) low rank. Here $\mathbb{X}$ is the space of $\mathbb{R}^{d\times d}$ matrices, and $\|\cdot\|$ is the spectral norm. Denote the Frobenius norm by $\|\cdot\|_F$ and the trace norm by $\|\cdot\|_{\mathrm{tr}}$. For $S = \{\boldsymbol{X}_1,\ldots,\boldsymbol{X}_n\} \subseteq \mathbb{R}^d$, define the empirical covariance matrix $\Sigma_S = \frac{1}{n}\sum_{i\in[n]}\boldsymbol{X}_i\boldsymbol{X}_i^\top$. We have the following result for low-rank estimation:

Lemma 26 (Koltchinskii et al. 2011) *Let $\hat{\Sigma} \in \mathbb{R}^{d\times d}$. Assume $\lambda \ge \|\hat{\Sigma} - \Sigma\|$, and let*

$$\Sigma_\lambda \in \operatorname*{argmin}_{A\in\mathbb{R}^{d\times d}} \frac{1}{2}\|\hat{\Sigma} - A\|_F^2 + \lambda\|A\|_{\mathrm{tr}}, \tag{13}$$

If $\lambda \ge \|\hat{\Sigma} - \Sigma\|$, then

$$\frac{1}{2}\|\hat{\Sigma}_\lambda - \Sigma\|_F^2 \le \inf_{A\in\mathbb{R}^{d\times d}}\left\{\frac{1}{2}\|A-\Sigma\|_F^2 + \frac{1}{2}(\sqrt{2}+1)^2\lambda^2\,\mathrm{rank}(A)\right\}.$$

Now, assume condition 1 holds for $\mathcal{X} \sim \mathcal{D}$, and suppose for simplicity that $\|\Sigma\| \le 1$. In this case, by Lemma 23, A random sample S of size $n' = c'_\eta\epsilon^{-2-2/\eta}d$, where $c'_\eta = c_\eta(3/2)^{2+2/\eta}$ suffices to get an empirical covariance matrix Σ_S such that $\|\Sigma_S - \Sigma\| \le \epsilon$ with probability at least 2/3.

Given a sample of size n from $\mathcal{D}$, We can thus implement $\mathrm{APPROX}_{\|\cdot\|,\varepsilon}$ that simply returns the empirical covariance matrix of a sub-sample of size $n' = n/k$, so that Assumption 1 holds for an appropriate ε. By Proposition 9, Algorithm 2 returns $\hat{\Sigma}$ such that with probability at least $1-\exp(-k/18)$, $\|\hat{\Sigma} - A\| \le 3\epsilon$. The resulting $\hat{\Sigma}$ can be used to minimize Eq. (13) with $\lambda = 3\epsilon := O\left((c'_\eta d\log(1/\delta)/n)^{1/2(1+1/\eta)}\right)$. The output matrix Σ_λ satisfies, with probability at least $1-\delta$,

$$\frac{1}{2}\|\Sigma_\lambda - \Sigma\|_F^2 \le \inf_{A\in\mathbb{R}^{d\times d}}\left\{\frac{1}{2}\|A-\Sigma\|_F^2 + O\left((c'_\eta d\log(1/\delta)/n)^{1/(1+1/\eta)}\right)\cdot\mathrm{rank}(A)\right\}.$$

7. A Comparison of Robust Distance Approximation Methods

The approach described in Section 3 for selecting a single $\boldsymbol{w}_i$ out of the set $\boldsymbol{w}_1,\ldots,\boldsymbol{w}_k$, gives one Robust Distance Approximation procedure (see Def. 1), in which the $\boldsymbol{w}_i$ with the lowest median distance from all others is selected. In this section we consider other Robust Distance Approximation procedures and their properties. We distinguish between procedures that return $y \in W$, which we term *set-based*, and procedures that might return any $y \in \mathbb{X}$, which we term *space-based*.

Recall that we consider a metric space $(\mathbb{X},\rho)$, with $W \subseteq \mathbb{X}$ a (multi)set of size k and $w_\star$ a distinguished element. Let $W_+ := W \cup \{w_\star\}$. In this formalization, the procedure used in Algorithm 2 is to simply select $y \in \operatorname{argmin}_{w\in W}\Delta_W(w,0)$, a set-based procedure. A natural variation of this is the space-based procedure: select $y \in \operatorname{argmin}_{w\in\mathbb{X}}\Delta_W(w,0)$.[6] A different approach, proposed by Minsker (2013), is to select $y \in \operatorname{argmin}_{w\in\mathbb{X}}\sum_{\bar{w}\in W}\rho(w,\bar{w})$, that is to minimize the geometric median over the space. Minsker analyzes this approach for Banach and Hilbert spaces. We show that minimizing the geometric median also achieves similar guarantees in general metric spaces.

In the following, we provide detailed guarantees for the approximation factor C_α of the two types of procedures, for general metric spaces as well as for Banach and Hilbert spaces, and for set-based and sample-based procedures. We further provide lower bounds for specific procedures, as well as lower bounds that hold for any procedure. In Section 7.4

6. The space-based median distance approach might not always be computationally feasible; see discussion in Section 7.4.

we summarize the results and compare the guarantees of the two procedures and the lower bounds. For a more useful comparison, we take into account the fact that the value of α usually affects not only the approximation factor, but also the upper bound obtained for $\Delta_W(w_\star, \alpha)$.

7.1 Minimizing the Median Distance

Minimizing the median distance over the set of input points was shown in Proposition 8 to achieve an approximation factor of 3. In this section we show that this upper bound on the approximation factor is tight for this procedure, even in a Hilbert space. Here and below, we say that an approximation factor upper bound is *tight* if for any constant smaller than this upper bound, there are a suitable space and a set of points in that space, such that the procedure achieves for this input a larger approximation factor than said constant.

The approximation factor can be improved to 2 for a sample-based procedure. This factor is tight as well, even assuming a Hilbert space. The following theorem summarizes these facts.

Theorem 27 *Let $k \geq 2$, and suppose that $\Delta_W(w_\star, \gamma) \leq \epsilon$ for some $\gamma > 0$. Let $y \in \operatorname{argmin}_{w \in W} \Delta_W(w, 0)$. Further, suppose that $W_+ \subseteq \mathbb{X}$, and let $\bar{y} \in \operatorname{argmin}_{w \in \mathbb{X}} \Delta_W(w, 0)$. Then*

- *For any metric space, $\rho(w_\star, y) \leq 3\epsilon$;*
- *For any metric space, $\rho(w_\star, \bar{y}) \leq 2\epsilon$;*
- *There exists a set on the real line such that $\rho(w_\star, y) = 3\epsilon$, where ρ is the distance induced by the inner product;*
- *There exists a set on the real line such that $\rho(w_\star, \bar{y}) = 2\epsilon$, where ρ is the distance induced by the inner product.*

Proof First, we prove the two upper bounds. Since $\Delta_W(w_\star, \gamma) \leq \epsilon$, we have $|B(w_\star, \epsilon) \cap W| > k/2$. Let $w \in |B(w_\star, \epsilon) \cap W|$. Then by the triangle inequality, $B(w, 2\epsilon) \supseteq B(w_\star, \epsilon)$. Therefore $\Delta_W(w, 0) \leq 2\epsilon$. It follows that $\Delta_W(y, 0) \leq 2\epsilon$, hence $|B(y, 2\epsilon) \cap W| \geq k/2$. By the pigeon hole principle, $|B(w_\star, \epsilon) \cap B(y, 2\epsilon)| > 0$, therefore $\rho(w_\star, y) \leq 3\epsilon$.

As for $\bar{y}$, since this is a minimizer over the entire space $\mathbb{X}$ which includes $w_\star$, we have $\Delta_W(y, \gamma) \leq \Delta_W(w_\star, \gamma) \leq \epsilon$. Therefore, similarly to the argument for y, we have $\rho(w_\star, y) \leq 2\epsilon$.

To see that these bounds are tight, we construct simple examples on the real line. For y, suppose $w_\star = \epsilon$, and consider W with k points as follows: $k/2-1$ points at 0, 2 points at 2ϵ, and $k/2-1$ points at 4ϵ. The points at 4ϵ are clearly in $\operatorname{argmin}_{w \in W} \Delta_W(w, 0)$, therefore $\rho(w_\star, y) = 3\epsilon$.

For $\bar{y}$, suppose $w_\star = \epsilon$, and consider W with k points as follows: 2 points at 0, $k/2-1$ points at 2ϵ, and $k/2-1$ points at 3ϵ. The points at 3ϵ are clearly in $\operatorname{argmin}_{w \in W_+} \Delta_W(w, 0)$, therefore $\rho(w_\star, \bar{y}) = 2\epsilon$. ■

The non-uniqueness of the median distance minimizer is exploited in the lower bounds in Theorem 27. This suggests that some kind of aggregation of the median distance minimizers may provide a smaller bound at least in certain scenarios.

7.2 The Geometric Median

For $w \in \mathbb{X}$, denote the sum of distances from points in the input set by $\operatorname{sumd}(w) := \sum_{v \in W} \rho(w, v)$. Minsker (2013) suggests to minimize the sum of distances over the entire space, that is, to select the geometric median. Minsker shows that when this procedure is applied in a Hilbert space, $C_\alpha \leq \frac{\frac{1}{2}+\alpha}{\sqrt{2\alpha}}$, and for a Banach space $C_\alpha \leq 1 + \frac{1}{2\alpha}$. Here we show that in fact $C_\alpha \leq 1 + \frac{1}{2\alpha}$ for general metric spaces. The proof holds, in particular, for Banach spaces, and thus this provide a more direct argument that does not require the special properties of Banach spaces. We further show that for general metric spaces, this upper bound on the approximation factor is tight.

Minimizing over the entire space is a computationally intensive procedure, involving convex approximation. Moreover, if the only access to the metric is via estimated distances based on samples, as in Algorithm 3, then there are additional statistical challenges. It is thus of interest to also consider the simpler set-based procedure, and we provide approximation guarantees for this procedure as well. We show that an approximation factor of $2 + \frac{1}{2\alpha}$ can be guaranteed for set-based procedures in general metric spaces, and this is also tight, even for Banach spaces.

The following theorem provides a bound that holds in several of these settings.

Theorem 28 *Let $k \geq 2$. Let $y \in \operatorname{argmin}_{w \in W} \operatorname{sumd}(w)$, and let $\bar{y} \in \operatorname{argmin}_{w \in W_+} \operatorname{sumd}(w)$. Then*

1. *For any metric space $(\mathbb{X}, \rho)$ and W, W_+,*

$$\rho(w_\star, y) \leq \left(2 + \frac{1}{2\alpha}\right) \Delta_W(w_\star, \alpha).$$

2. *For any constant $C < (2 + \frac{1}{2\alpha})$, there exists a problem in a Banach space such that $\rho(w_\star, y) > C \cdot \Delta_W(w_\star, \alpha)$. Thus the upper bound above is tight.*

3. *For any metric space $(\mathbb{X}, \rho)$ and W, W_+,*

$$\rho(w_\star, \bar{y}) \leq \left(1 + \frac{1}{2\alpha}\right) \Delta_W(w_\star, \alpha).$$

4. *For any constant $C < (1 + \frac{1}{2\alpha})$, there exists a problem in a metric space such that $\rho(w_\star, \bar{y}) > C \cdot \Delta_W(w_\star, \alpha)$. Thus the upper bound above is tight for general metric spaces.*

Proof Let $w \in \operatorname{argmin}_{w \in B(w_\star, \epsilon) \cap W} \rho(w, y)$. Let $Z \subset B(w_\star, \epsilon) \cap W$ such that $|Z| = k(\frac{1}{2} + \alpha)$ (we assume for simplicity that $k(\frac{1}{2} + \alpha)$ is an integer; the proof can be easily modified to

accommodate the general case). For $v \in Z$, $\rho(w,v) \le \rho(w,w_\star) + \rho(w_\star,v)$. For $v \in W \setminus Z$, $\rho(w,v) \le \rho(w,y) + \rho(y,v)$. Therefore

$$\mathrm{sumd}(w) \le \sum_{v\in Z}(\rho(w,w_\star)+\rho(w_\star,v)) + \sum_{v\in W\setminus Z}(\rho(w,y)+\rho(y,v)).$$

By the definition of w as a minimizer, for $v \in Z$, $\rho(y,v) \ge \rho(y,w)$. Thus

$$\mathrm{sumd}(y) \ge \sum_{v\in Z}\rho(y,w) + \sum_{v\in W\setminus Z}\rho(y,v).$$

Since $\mathrm{sumd}(y) \le \mathrm{sumd}(w)$, we get

$$\sum_{v\in Z}\rho(y,w) + \sum_{v\in W\setminus Z}\rho(y,v) \le \sum_{v\in Z}(\rho(w,w_\star)+\rho(w_\star,v)) + \sum_{v\in W\setminus Z}(\rho(w,y)+\rho(y,v)).$$

Hence, since $\rho(v,w_\star) \le \epsilon$ for $v \in Z$,

$$(|Z| - |W\setminus Z|)\rho(w,y) \le 2|Z|\epsilon.$$

Since $|Z| = k(\frac{1}{2}+\alpha)$ it follows that $\rho(w,y) \le (1+\frac{1}{2\alpha})\epsilon$. In addition,

$$\rho(w_\star,y) \le \rho(w,w_\star) + \rho(w,y) \le \epsilon + \rho(w,y),$$

therefore

$$\rho(w_\star,y) \le \left(2+\frac{1}{2\alpha}\right)\epsilon.$$

This shows that for any metric space, the set-based geometric median gives an approximation factor of $2+\frac{1}{2\alpha}$, proving item 1.

For the space-based geometric median, consider $\bar{w} \in \mathrm{argmin}_{w\in B(w_\star,\epsilon)\cap W}\rho(w,\bar{y})$. We have $\mathrm{sumd}(\bar{y}) \le \mathrm{sumd}(w_\star)$. In addition,

$$\mathrm{sumd}(w_\star) \le \sum_{v\in Z}\rho(w_\star,v) + \sum_{v\in W\setminus Z}(\rho(w_\star,\bar{w})+\rho(\bar{w},\bar{y})+\rho(\bar{y},v)).$$

Therefore,

$$\sum_{v\in Z}\rho(\bar{y},\bar{w}) + \sum_{v\in W\setminus Z}\rho(\bar{y},v) \le \sum_{v\in Z}\rho(w_\star,v) + \sum_{v\in W\setminus Z}(\rho(w_\star,w)+\rho(w,\bar{y})+\rho(\bar{y},v)).$$

Since $\rho(w_\star,v) \le \epsilon$ for $v \in Z$, and $\rho(w_\star,\bar{w}) \le \epsilon$, it follows

$$(|Z| - |W\setminus Z|)\rho(\bar{w},\bar{y}) \le k\epsilon.$$

Therefore $\rho(\bar{w},\bar{y}) \le \frac{1}{2\alpha}\epsilon$, hence

$$\rho(w_\star,\bar{y}) \le \rho(w_\star,\bar{w}) + \rho(\bar{w},\bar{y}) \le \left(1+\frac{1}{2\alpha}\right)\epsilon.$$

This gives an approximation factor of $1+\frac{1}{2\alpha}$ for space-based geometric median, proving item 3.

To see that both of these bounds are tight, let $n = k(\frac{1}{2}+\alpha)$, and let $\mathbb{X} = W_+ = \{v_1,\ldots,v_n,y_1,\ldots,y_{k-n},w_\star\}$. Define $\rho(\cdot,\cdot)$ as follows (for all pairs $i \ne j$, $l \ne t$):

$$\begin{aligned}\rho(w_\star,v_i) &= \epsilon\\ \rho(w_\star,y_l) &= \beta\\ \rho(v_i,v_j) &= 2\epsilon\\ \rho(v_i,y_t) &= \beta-\epsilon\\ \rho(y_t,y_l) &= 0.\end{aligned}$$

One can verify that for any $\beta \le (2+\frac{1}{2\alpha}-\frac{1}{k\alpha})\epsilon$, $\mathrm{sumd}(y_l) \le \mathrm{sumd}(v_i)$ for all l,i. Therefore, the approximation factor for set-based geometric median in a general metric space is lower-bounded by $2+\frac{1}{2\alpha}$ for general k. This holds also for Banach spaces as well, Since any metric space can be embedded into a Banach space (Kuratowski, 1935). This proves item 2.

For space-based geometric median, note that if $\beta \le (1+\frac{1}{2\alpha})\epsilon$, then $\mathrm{sumd}(w_\star) \ge \mathrm{sumd}(y_l)$. Therefore the space-based upper bound is tight for a general metric space. This proves item 4. ■

Since $\alpha \in (0,\frac{1}{2})$, the guarantee for the geometric median in these settings is always worse than the guarantee for minimizing the median distance. Factoring in the dependence on α, the difference is even more pronounced. The full comparison is given in Section 7.4 below.

7.3 Optimal Approximation Factor

In this section we give lower bounds that hold for any robust distance approximation procedure. A lower bound of $C > 0$ for a category of metric spaces and a type of procedure indicates that if a procedure of this type guarantees a distance approximation C_α for all metric spaces of the given category, then necessarily $C_\alpha \ge C$. As shown below, in many cases the lower bounds provided here match the upper bounds obtained by either the median distance or the geometric median.

The following theorem gives a lower bound of 3 for the achievable approximation factor of set-based procedures in Banach spaces (and so, also in general metric spaces). This factor is achieved by the median distance minimizer, as shown in Theorem 27.

Theorem 29 *Consider set-based robust distance approximation procedures. For any $\alpha \in (0,\frac{1}{2})$, and for any such procedure, there exists a problem in a Banach space for which the approximation factor of the procedure is at least 3.*

Proof Fix α, and let $n = \lceil \frac{1}{\frac{1}{2}-\alpha} \rceil$. Define the metric space $\mathbb{X} = \{a_1,\ldots,a_n,b_1,\ldots,b_n\}$ with the metric $\rho(\cdot,\cdot)$ defined as follows: For all $i \ne j$, $\rho(a_i,a_j) = 2$, $\rho(a_i,b_j) = 1$, $\rho(b_i,b_j) = 2$. For all i, $\rho(a_i,b_i) = 3$. See Figure 2 for illustration.

Consider the multi-set W with k/n elements at every b_i. It is easy to check that for every a_i, $\Delta_W(a_i,\alpha) \le \Delta_W(a_i,1/2-1/n) = 1$. On the other hand, since the problem is

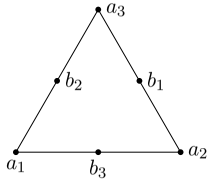


Figure 2: The metric defined in Theorem 29 for $n = 3$. The distances are shortest paths on the underlying undirected graph, where all edges are the same length.

symmetric for permutations of the indices $1, \ldots, n$, no procedure can distinguish the cases $w_\star = a_i$ for different $i \in [n]$. For any choice $y = b_i \in W$, if $w_\star = a_i$ then $\rho(w_\star, y) = 3$. Therefore the approximation factor of any procedure is at least 3. Since any metric space can be embedded into a Banach space (Kuratowski, 1935) this result holds also for Banach spaces. ■

Next, we give a lower bound of 2 for space-based procedures over general metric spaces. Theorem 27 shows that this factor is also achieved by minimizing the median distance.

Theorem 30 *Consider robust space-based distance approximation procedures. For any $\alpha \in (0, \frac{1}{2})$, and for any such procedure, there exists a problem for which the approximation factor of the procedure is at least* 2.

Proof Fix α, and let $n = \lceil \frac{1}{\frac{1}{2}-\alpha} \rceil$. Define the metric space $\mathbb{X} = \{a_1, \ldots, a_n, b_1, \ldots, b_n\}$ with the metric $\rho(\cdot, \cdot)$ defined as follows: For all $i \neq j$, $\rho(a_i, a_j) = 2$, $\rho(a_i, b_j) = 1$, $\rho(b_i, b_j) = 1$. For all i, $\rho(a_i, b_i) = 2$. See Figure 3 for illustration.

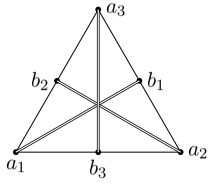


Figure 3: The metric defined in Theorem 29 for $n = 3$. The distances are shortest paths on the underlying undirected graph. The full lines are edges of length 1, the double lines from a_i to b_i are edges of length 2.

Consider the multi-set W with k/n points at every b_i. It is easy to check that for every a_i, $\Delta_W(a_i, \alpha) \leq \Delta_W(a_i, 1/2 - 1/n) = 1$. On the other hand, since the problem is symmetric

for permutations of the indices $1, \ldots, n$, no procedure can distinguish the cases $w_\star = a_i$ for different $i \in [n]$. Moreover, any point y in the space has $\rho(a_i, y) = 2$ for at least one $i \in [n]$. Therefore the approximation factor of any procedure is at least 2. ■

For lower bounds on Hilbert spaces and Banach spaces, we require the following lemma, which gives the radius of the ball inscribing the regular simplex in a p-normed space.

Lemma 31 *Consider $\mathbb{R}^n$ with the p-norm for $p > 1$. Let $e_1, \ldots, e_n$ be the standard basis vectors, and let $r_{n,p}$ be the minimal number for which there exists an $x \in \mathbb{R}^n$ such that $B(x, r) \supseteq \{e_1, \ldots, e_n\}$. Then $r_{n,p} = ((1+(n-1)^{-1/(p-1)})^{-p} + (n-1)(1+(n-1)^{1/(p-1)})^{-p})^{1/p}$. This radius is obtained with the center x such that for all i, $x_i = (1 + (n-1)^{1/(p-1)})^{-1}$.*

Proof It is easy to see that due to symmetry, $x = (a, a, \ldots, a)$ for some real number a. Thus $r_{n,p} = \inf_{a \in \mathbb{R}} \|e_1 - (a, \ldots, a)\|_p$. We have $\|e_1 - (a, \ldots, a)\|_p^p = |1 - a|^p + (n-1)|a|^p$. Minimizing over a gives $a = (1 + (n-1)^{1/(p-1)})^{-1}$, and

$$r_{n,p}^p = |1 - a|^p + (n-1)|a|^p = (1 + (n-1)^{-1/(p-1)})^{-p} + (n-1)(1 + (n-1)^{1/(p-1)})^{-p}.$$

■

We now prove a lower bound for robust distance approximation in Hilbert spaces. Unlike the previous lower bounds, this lower bound depends on the value of α.

Theorem 32 *Consider robust distance approximation procedures for $(\mathbb{X}, \rho)$ a Hilbert space. For any $\alpha \in (0, \frac{1}{2})$, the following holds:*

- *For any set-based procedure, there exists a problem such that the procedure achieves an approximation factor at least*

$$\sqrt{1 + \frac{2}{\left\lceil \frac{1}{\frac{1}{2}-\alpha} \right\rceil - 2}}.$$

- *For any space-based procedure, there exists a problem such that the procedure achieves an approximation factor at least*

$$\sqrt{1 + \frac{1}{\left\lceil \frac{1}{\frac{1}{2}-\alpha} \right\rceil^2 - 2\left\lceil \frac{1}{\frac{1}{2}-\alpha} \right\rceil}}.$$

The space-based bound given in Theorem 32 is tight for $\alpha \to 1/2$. This can be seen by noting that the limit of the space-based lower bound for $\alpha \to 1/2$ is $(\frac{1}{2} + \alpha)/\sqrt{2\alpha}$, which is exactly the guarantee provided in Minsker (2013) for the space-based geometric median procedure. For smaller α, there is a gap between the guarantee of Minsker for the geometric median and our lower bound.

Proof Fix α, and let $n = \lceil \frac{1}{\frac{1}{2}-\alpha} \rceil$. Consider the Euclidean space $\mathbb{R}^n$ with $\rho(x, y) = \|x - y\|$. Let $e_1, \ldots, e_n$ be the standard basis vectors. These are the vertices of a regular simplex

with side length $\|e_i - e_j\| = \sqrt{2}$. Let $b_1, \ldots, b_n$ such that b_i is the center of the hyperface of the simplex opposing e_i. Then $\|b_i - e_j\| = r_{n-1,2}$ for all $j \neq i$, where $r_{n,2} = \sqrt{\frac{n-1}{n}}$ is as defined in Lemma 31. (see Figure 4).

Consider W with k/n points at each of $b_1, \ldots, b_n$. Then $\Delta_W(e_i, \alpha) \leq \Delta_W(e_i, 1 - \frac{1}{n}) = \|e_i - b_j\| = r_{n-1,2}$ for any $j \neq i$. Any set-based procedure must select b_i for some i. if $w_\star = e_i$, the resulting approximation factor is $\|e_i - b_i\|/r_{n-1,2} = \sqrt{\frac{n-2}{n-1}}\|e_i - b_i\|$. For $\|b_i - e_i\|$, consider for instance b_1 and e_1. We have $b_1 = (0, \frac{1}{n-1}, \ldots, \frac{1}{n-1})$, therefore $\|b_1 - e_1\| = \sqrt{\frac{n}{n-1}}$. The approximation factor of the procedure is thus at least $\sqrt{\frac{n}{n-2}}$.

For a set-based procedure, whatever y it returns, there exists at least one i such that $\|y - a_i\| \geq r_{n,2}$. Therefore the approximation factor is at least $r_{n,2}/r_{n-1,2} = \sqrt{\frac{n-1}{n}}/\sqrt{\frac{n-2}{n-1}} = \sqrt{1 + \frac{1}{n^2-2n}}$.

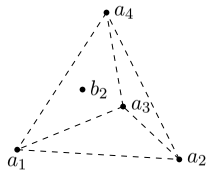


Figure 4: The regular simplex in $\mathbb{R}^3$, $n = 4$. a_i is a vertex, b_i is the center of the face opposite a_i.

■

For space-based procedures, we have seen that while there exists a lower bound of 2 for general metric spaces, in a Hilbert space better approximation factors can be achieved. Is it possible that in Banach spaces the same approximation factor can also be achieved? The following theorem shows that the answer is no.

Theorem 33 *Let $\alpha = 1/6$. There exists a Banach space for which an approximation factor of $(\frac{1}{2} + \alpha)/\sqrt{2\alpha}$ cannot be achieved.*

Proof Consider the space $\mathbb{R}^n$ with the distance defined by a p-norm. Let $n = 1/(\frac{1}{2} - \alpha) = 3$. Construct W as in the proof of Theorem 32, with k/n points in each of $b_1, \ldots, b_n$, where b_i is the center (in the p-norm) of the hyperface opposing the basis vector e_i. As in the proof of Theorem 32, the approximation factor for any space-based procedure for this problem is at least $r_{n,p}/r_{n-1,p}$. For $p = 3/2$, we have $r_{n,p}/r_{n-1,p} = \frac{2}{5^{1/3}} > \frac{2}{\sqrt{3}} = \frac{\frac{1}{2}+\alpha}{\sqrt{2\alpha}}$. ■

	General Metric	Banach	Hilbert
Set-based			
Optimal	$= 3$	$= 3$	$\geq \sqrt{1 + \frac{2}{\left\lceil \frac{1}{\frac{1}{2}-\alpha} \right\rceil - 2}}$ $\xrightarrow{\alpha \to 1/2} 1/\sqrt{2\alpha}$
Median distance	$= 3$	$= 3$	$= 3$
Geometric median	$= 2 + 1/(2\alpha)$	$= 2 + 1/(2\alpha)$	Open
Space-based			
Optimal	$= 2$	Strictly larger than for Hilbert spaces	$\geq \sqrt{1 + \frac{1}{\left\lceil \frac{1}{\frac{1}{2}-\alpha} \right\rceil^2 - 2\left\lceil \frac{1}{\frac{1}{2}-\alpha} \right\rceil}}$ $\xrightarrow{\alpha \to 1/2} \frac{\frac{1}{2}+\alpha}{\sqrt{2\alpha}}$
Median distance	$= 2$	$= 2$	$= 2$
Geometric median	$= 1 + 1/(2\alpha)$	$\leq 1 + 1/(2\alpha)$ $(\star)$	$\leq (\frac{1}{2} + \alpha)/\sqrt{2\alpha}$ $(\star)$

Table 1: Approximation factors for $\alpha \in (0, 1/2)$, based on type of procedure and type of space. Results marked with $(\star)$ are due to Minsker (2013). Equality indicates matching upper and lower bounds.

	General Metric	Banach	Hilbert
Set-based			
Optimal	$= 6$	$= 6$	≥ 3.46
Median distance	$= 6$	$= 6$	$= 6$
Geometric median	$= 14.92$	$= 14.92$	Open
Space-based			
Optimal	$= 4$	Open	≥ 2.31
Median distance	$= 4$	$= 4$	$= 4$
Geometric median	$= 11.65$	≤ 11.65	≤ 3.33

Table 2: Optimal normalized approximation factors based on the values of C_α given in Table 1. The value in each case is $\inf_{\alpha \in (0, \frac{1}{2})} \frac{C_\alpha}{(\frac{1}{2} - \alpha)}$ for the corresponding C_α. All non-integers are rounded to 2 decimal places.

7.4 Comparison of Selection Procedures

The results provided above are summarized in Table 1. When comparing different procedures for different values of α, it is useful to compare not only the respective approximation factors but also the upper bound that can be obtained for $\Delta_W(w_\star, \alpha)$. Typically, as in the proof of Proposition 9, this upper bound will stem from first bounding $\mathbb{E}[\rho(w_\star, w)] \leq \epsilon$, where the expectation is taken over random i.i.d. draws of w, and then applying Markov's inequality to obtain $\mathbb{P}[\rho(w_\star, w) \leq \frac{\epsilon}{\frac{1}{2}-\alpha}] \geq \frac{1}{2} + \alpha$. In the final step Hoeffding's inequality guarantees that if k is large enough, $|B(w_\star, \epsilon/(\frac{1}{2}-\alpha)) \cap W|$ approaches $k(\frac{1}{2}+\alpha)$. Therefore, for a large k and a procedure for α with an approximation factor C_α, the guarantee approaches $\rho(y, w_\star) \leq \frac{C_\alpha}{(\frac{1}{2}-\alpha)} \cdot \epsilon$. For a procedure with an approximation factor C_α, we call $\frac{C_\alpha}{(\frac{1}{2}-\alpha)}$ the *normalized* approximation factor of the procedure. This is the approximation factor with respect to $\mathbb{E}[\rho(w_\star, \alpha)]$. When the procedure supports a range of α, the optimal normalized factor can be found by minimizing $\frac{C_\alpha}{(\frac{1}{2}-\alpha)}$ over $\alpha \in (0, \frac{1}{2})$. If $C_\alpha = C$ is a constant, the optimal normalized approximation factor is $2C$, achieved when $\alpha = 0$. The optimal normalized approximation factors, based on the known approximation factors as a function of α, are given in Table 2.

We observe that for set-based procedures, the median distance is superior to the geometric median for general metric spaces as well as for general Banach spaces. It is an open question whether better results can be achieved for Hilbert spaces using set-based procedures.

For space-based procedures, the median distance is again superior, except in the case of a Hilbert space, where the geometric median is superior. The case of a Hilbert space is arguably the most useful in common applications such as linear regression. Nevertheless, gaps still remain and it would be interesting to develop optimal methods.

Implementing the geometric median procedure in a space-based formulation is computationally efficient for Hilbert spaces when accurate distances are available Minsker (2013). However, it is unknown whether and how the procedure can be implemented when only unreliable distance estimations are available, as in Section 3.3. A useful implementation should be both computationally feasible and statistically efficient, while degrading the approximation factors as little as possible.

8. Predicting Without a Metric on Predictors

The core technique presented above allows selecting a good candidate out of a set that includes mostly good candidates, in the presence of a metric between candidates. If the final goal is prediction of a scalar label, good prediction can still be achieved without access to a metric between candidates, using the following simple procedure: For every input data point, calculate the prediction of every candidate, and output the median of the predictions. This is a straight-forward generalization of voting techniques for classification such as when using bagging (Breiman, 1996).[7] The following lemma shows that this approach leads to guarantees similar to those achieved by Proposition 9.

7. Note, however, that the usual implementation of bagging for regression involves averaging over the outputs of the classifiers, and not taking the median.

Lemma 34 *Let D, $\ell : \mathcal{Z} \times \mathbb{X} \to \mathbb{R}_+$ and $L : \mathbb{X} \to \mathbb{R}_+$ be defined as in Section 4. Assume that $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, and there are functions $f : \mathcal{X} \times \mathbb{X} \to \mathbb{R}$ (the prediction function) and $g : \mathbb{R} \times \mathbb{R}$ (the link function) such that $\ell((\boldsymbol{x}, y), \boldsymbol{w}) = g(f(\boldsymbol{x}, \boldsymbol{w}), y)$. Assume that g is convex its first argument. Suppose that we have k predictors $w_1, \ldots, w_k$ such that for at least $(\frac{1}{2}+\gamma)k$ of them, $L(\boldsymbol{w}) \leq \bar{\ell}$. For $x \in \mathcal{X}, y \in \mathcal{Y}$, let $\hat{y}(\boldsymbol{x})$ be the median of $f(\boldsymbol{x}, \boldsymbol{w}_1), \ldots, f(\boldsymbol{x}, \boldsymbol{w}_k)$, and let $\hat{\ell}(\boldsymbol{x}, y) = g(\hat{y}(\boldsymbol{x}), y)$. Let $\hat{L} := \mathbb{E}[\hat{\ell}(\hat{y}(\boldsymbol{x}))]$. Then*

$$\hat{L} \leq \left(\frac{1}{2\gamma} + 1\right)\bar{\ell}.$$

Proof Let $I = \{i : L(\boldsymbol{w}_i) \leq \bar{\ell}\}$. Assume without loss of generality that for $i \in [k-1]$, $f(\boldsymbol{x}, \boldsymbol{w}_i) \leq f(\boldsymbol{x}, \boldsymbol{w}_{i+1})$. Let $t \in [k]$ such that $\hat{y}(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{w}_t)$. By the convexity of g, at least one of $g(f(\boldsymbol{x}, \boldsymbol{w}_t), y) \leq g(f(\boldsymbol{x}, \boldsymbol{w}_{t-1}, y))$ and $g(f(\boldsymbol{x}, \boldsymbol{w}_t), y) \leq g(f(\boldsymbol{x}, \boldsymbol{w}_{t+1}, y))$ holds. assume without loss of generality that the first inequality holds. It follows that for all $i \in [t]$, $g(f(\boldsymbol{x}, \boldsymbol{w}_i), y) \geq g(f(\boldsymbol{x}, \boldsymbol{w}_t, y))$. Therefore,

$$\begin{aligned}\hat{\ell}(\boldsymbol{x}, y) = g(f(\boldsymbol{x}, \boldsymbol{w}_t), y)) &\leq \frac{1}{|I \cap [t]|} \sum_{i \in I \cap [t]} g(f(\boldsymbol{x}, \boldsymbol{w}_i), y) \\ &\leq \frac{1}{|I \cap [t]|} \sum_{i \in I} g(f(\boldsymbol{x}, \boldsymbol{w}_i), y) = \frac{1}{|I \cap [t]|} \sum_{i \in I} \ell((\boldsymbol{x}, y), \boldsymbol{w}_i).\end{aligned}$$

Taking expectation over $(\boldsymbol{x}, y)$,

$$\hat{L} \leq \frac{1}{|I \cap [t]|} \sum_{i \in I} L(\boldsymbol{w}_i) \leq \frac{|I|}{|I \cap [t]|}\bar{\ell} \leq \frac{\frac{1}{2}+\gamma}{\gamma}\bar{\ell},$$

where the last inequality follows from the assumption that $|I| \geq (\frac{1}{2}+\gamma)k$. ■

A downside of this approach is that each prediction requires many applications of a predictor. If there is also access to unlimited unlabeled data, a possible approach to circumvent this issue is to generate predictions for a large set of random unlabeled data points based on the aggregate predictor, and then use the resulting labeled pairs as a training set to find a single predictor with a loss that approaches the loss of the aggregate predictor. A similar approach for derandomizing randomized classifiers was suggested by Kääriäinen (2005).

9. Conclusion

In this paper we show several applications of a generalized median-of-means approach to estimation. In particular, for linear regression we establish convergence rates for heavy-tailed distributions that match the min-max rates up to logarithmic factors. We further show conditions that allow parameter estimation using the Lasso under heavy-tailed noise, and cases under which low-rank covariance matrix approximation is possible for heavy-tailed distributions.

The core technique is based on performing independent estimates on separate random samples, and then combining these estimates. Other works have considered approaches

which resemble this general scheme but provide other types of guarantees. For instance, in Zhang et al. (2013), faster parallel kernel ridge regression is achieved by performing loss minimizations on independent samples and then averaging the resulting estimators. In Rakhlin et al. (2013), faster rates of convergence for regression for some classes of estimators are achieved, using linear combinations of risk minimizers over subsets of the class of estimators. These works, together with ours, demonstrate that empirical risk minimization can be used as a black box to generate new algorithms with improved statistical performance.

Acknowledgments

Part of this work was completed while the authors were at Microsoft Research New England. Daniel Hsu was supported by a Yahoo Academic Career Enhancement Award. Sivan Sabato is supported by the Lynne and William Frankel Center for Computer Science.

Appendix A. Proof of Theorem 25

From the definition of $\hat{\boldsymbol{w}}$ as a minimizer we have

$$\|\Psi(\boldsymbol{w}_\star - \hat{\boldsymbol{w}})\|_2^2 + 2\lambda\|\hat{\boldsymbol{w}}\|_1 \leq 2\lambda\|\boldsymbol{w}_\star\|_1 + 2\varepsilon^\top\Psi(\hat{\boldsymbol{w}} - \boldsymbol{w}_\star). \tag{14}$$

By Hölder's inequality the assumptions of the theorem, $2\varepsilon^\top\Psi(\hat{\boldsymbol{w}} - \boldsymbol{w}_\star) \leq 2\|\varepsilon^\top\Psi\|_\infty\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_1 \leq \lambda\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_1$. Combining this with Eq. (14) gives

$$\|\Psi(\boldsymbol{w}_\star - \hat{\boldsymbol{w}})\|_2^2 \leq 2\lambda\|\boldsymbol{w}_\star\|_1 - 2\lambda\|\hat{\boldsymbol{w}}\|_1 + \lambda\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_1.$$

Adding $\lambda\|(\hat{\boldsymbol{w}} - \boldsymbol{w})\|_1$ to both sides we get

$$\begin{aligned}
\|\Psi(\boldsymbol{w}_\star - \hat{\boldsymbol{w}})\|_2^2 + \lambda\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_1 &\leq 2\lambda\Big(\|\hat{\boldsymbol{w}} - \boldsymbol{w}_\star\|_1 + \|\boldsymbol{w}_\star\|_1 - \|\hat{\boldsymbol{w}}\|_1\Big) \\
&= 2\lambda\sum_{j=1}^{d}\Big(|\hat{\boldsymbol{w}}[j] - \boldsymbol{w}_\star[j]| + |\boldsymbol{w}_\star[j]| - |\hat{\boldsymbol{w}}[j]|\Big) \\
&= 2\lambda\sum_{j\in\mathrm{supp}(\boldsymbol{w})}\Big(|\hat{\boldsymbol{w}}[j] - \boldsymbol{w}_\star[j]| + |\boldsymbol{w}_\star[j]| - |\hat{\boldsymbol{w}}[j]|\Big) \\
&\leq 4\lambda\sum_{j\in\mathrm{supp}(\boldsymbol{w})}|\hat{\boldsymbol{w}}[j] - \boldsymbol{w}_\star[j]| \\
&= 4\lambda\|[\hat{\boldsymbol{w}} - \boldsymbol{w}_\star]_{\mathrm{supp}(\boldsymbol{w})}\|_1.
\end{aligned}$$

It follows that

$$\|[\hat{\boldsymbol{w}} - \boldsymbol{w}_\star]_{\mathrm{supp}(\boldsymbol{w}_\star)^C}\|_1 \leq 3\|[\hat{\boldsymbol{w}} - \boldsymbol{w}_\star]_{\mathrm{supp}(\boldsymbol{w}_\star)}\|,$$

therefore $\hat{\boldsymbol{w}} - \boldsymbol{w}_\star \in E_s$. Denote $\boldsymbol{\delta} = \hat{\boldsymbol{w}} - \boldsymbol{w}$. The above derivation also implies

$$\|\Psi\boldsymbol{\delta}\|_2^2 \leq 3\lambda\|[\boldsymbol{\delta}]_{\mathrm{supp}(\boldsymbol{w}_\star)}\|_1 \leq 3\lambda\|\boldsymbol{\delta}_{[s]}\|_1 \leq 3\lambda\sqrt{s}\|\boldsymbol{\delta}_{[s]}\|_2.$$

Denote for brevity $\gamma = \gamma(\Psi, s)$. From the definition of γ,

$$\|\boldsymbol{\delta}_{[s]}\|_2^2 \leq \frac{1}{\gamma^2}\|\Psi\boldsymbol{\delta}\|_2^2 \leq \frac{3\lambda\sqrt{s}\|\boldsymbol{\delta}_{[s]}\|_2}{\gamma^2},$$

Therefore $\|\boldsymbol{\delta}_{[s]}\|_2 \leq \frac{3\lambda\sqrt{s}}{\gamma^2}$. Now,

$$\|\boldsymbol{\delta}\|_2 = \|\boldsymbol{\delta}_{[s]^C}\|_2 + \|\boldsymbol{\delta}_{[s]}\|_2 \leq \sqrt{\|\boldsymbol{\delta}_{[s]^C}\|_1\|\boldsymbol{\delta}_{[s]^C}\|_\infty} + \|\boldsymbol{\delta}_{[s]}\|_2.$$

From $\boldsymbol{\delta} \in E_s$ we get $\|\boldsymbol{\delta}_{[s]^C}\|_1 \leq 3\|\boldsymbol{\delta}_{[s]}\|_1$. In addition, since $\boldsymbol{\delta}_{[s]}$ spans the largest coordinates of $\boldsymbol{\delta}$ in absolute value, $\|\boldsymbol{\delta}_{[s]^C}\|_\infty \leq \|\boldsymbol{\delta}_{[s]}\|_1/s$. Combining these with the inequality above we get

$$\|\boldsymbol{\delta}\|_2 \leq 3\|\boldsymbol{\delta}_{[s]}\|_1/\sqrt{s} + \|\boldsymbol{\delta}_{[s]}\|_2 \leq 4\|\boldsymbol{\delta}_{[s]}\|_2 \leq \frac{12\lambda\sqrt{s}}{\gamma^2}.$$

■

References

Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58:137–147, 1999.

Jean-Yves Audibert and Olivier Catoni. Robust linear least squares regression. *Ann. Stat.*, 39(5):2766–2794, 2011.

Alexandre Belloni and Victor Chernozhukov. l1-penalized quantile regression in high-dimensional sparse models. *The Annals of Statistics*, 39(1):82–130, 2011.

Peter J Bickel, Yaacov Ritov, and Alexandre B Tsybakov. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, 37(4):1705–1732, 2009.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

C. Brownlees, E. Joly, and G. Lugosi. Empirical risk minimization for heavy-tailed losses. *ArXiv e-prints*, June 2014.

S. Bubeck, N. Cesa-Bianchi, and G. Lugosi. Bandits with heavy tail. *IEEE Transactions on Information Theory*, 59:7711–7717, 2013.

Olivier Catoni. Challenging the empirical mean and empirical variance: a deviation study. *Ann. Inst. H. Poincar Probab. Statist.*, 48(4):1148–1185, 2012.

A Chatterjee and SN Lahiri. Rates of convergence of the adaptive lasso estimators to the oracle distribution and higher order refinements by the bootstrap. *The Annals of Statistics*, 41(3):1232–1259, 2013.

Samprit Chatterjee and Ali S Hadi. Influential observations, high leverage points, and outliers in linear regression. *Statistical Science*, 1(3):379–393, 1986.

Bradley Efron. Bootstrap methods: another look at the jackknife. *The annals of Statistics*, pages 1–26, 1979.

Jianqing Fan, Yingying Fan, and Emre Barut. Adaptive robust variable selection. *arXiv preprint arXiv:1205.4795*, 2012.

Daniel Hsu and Sivan Sabato. Approximate loss minimization with heavy tails. *CoRR*, abs/1307.1827, 2013. URL `http://arxiv.org/abs/1307.1827`.

Daniel Hsu and Sivan Sabato. Heavy-tailed regression with a generalized median-of-means. In *Thirty-First International Conference on Machine Learning*, 2014.

Daniel Hsu, Sham M. Kakade, and Tong Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3):569–600, 2014.

P. J. Huber. *Robust Statistics*. Wiley, 1981.

Anatoli Juditsky and Arkadii S. Nemirovski. Large deviations of vector-valued martingales in 2-smooth normed spaces. *ArXiv e-prints*, 0809.0813, 2008.

Matti Kääriäinen. Generalization error bounds using unlabeled data. In *Learning Theory*, pages 127–142. Springer, 2005.

V. Koltchinskii, K. Lounici, and A. B. Tsybakov. Nuclear norm penalization and optimal rates for noisy low rank matrix completion. *Annals of Statistics*, 39(5):2302–2329, 2011.

Casimir Kuratowski. Quelques problèmes concernant les espaces métriques non-séparables. *Fundamenta Mathematicae*, 25(1):534–545, 1935.

O. V. Lepski. Asymptotically minimax adaptive estimation I: Upper bounds. optimally adaptive estimates. *Theory Probab. Appl.*, 36(4):682–697, 1991.

M. Lerasle and R. I. Oliveira. Robust empirical mean Estimators. *ArXiv e-prints*, December 2011.

Leonid A. Levin. Notes for miscellaneous lectures. *CoRR*, abs/cs/0503039, 2005.

Alexander E. Litvak, Alain Pajor, Mark Rudelson, and Nicole Tomczak-Jaegermann. Smallest singular value of random matrices and geometry of random polytopes. *Adv. Math.*, 195(2):491–523, 2005. ISSN 0001-8708. doi: 10.1016/j.aim.2004.08.004. URL `http://dx.doi.org/10.1016/j.aim.2004.08.004`.

Mehrdad Mahdavi and Rong Jin. Passive learning with target risk. In *Twenty-Sixth Conference on Learning Theory*, 2013.

S. Mendelson. Learning without Concentration. *ArXiv e-prints*, January 2014.

Stanislav Minsker. Geometric median and robust estimation in banach spaces. *arXiv preprint arXiv:1308.1334*, 2013.

A. S. Nemirovsky and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience, 1983.

M. Nussbaum. Minimax risk: Pinsker bound. In S. Kotz, editor, *Encyclopedia of Statistical Sciences, Update Volume 3*, pages 451–460. Wiley, New York, 1999.

Roberto Oliveira. Sums of random Hermitian matrices and an inequality by Rudelson. *Electron. Commun. Probab.*, 15(19):203–212, 2010.

Alexander Rakhlin, Karthik Sridharan, and Alexandre B. Tsybakov. Empirical entropy, minimax regret and minimax risk. *arXiv preprint arXiv:1308.1147*, 2013.

O. Shamir. The Sample Complexity of Learning Linear Predictors with the Squared Loss. *ArXiv e-prints*, June 2014.

Nathan Srebro, Karthik Sridharan, and Ambuj Tewari. Smoothness, low noise and fast rates. In *Advances in Neural Information Processing Systems 23*, 2010.

N. Srivastava and R. Vershynin. Covariance estimation for distributions with $2+\epsilon$ moments. *Annals of Probability*, 41:3081–3111, 2013.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Sara van de Geer and Patric Müller. Quasi-likelihood and/or robust estimation in high dimensions. *Statistical Science*, 27(4):469–480, 2012.

Hansheng Wang, Guodong Li, and Guohua Jiang. Robust regression shrinkage and consistent variable selection through the lad-lasso. *Journal of Business & Economic Statistics*, 25(3):347–355, 2007.

Lie Wang. L1 penalized lad estimator for high dimensional linear regression. *Journal of Multivariate Analysis*, 2013.

J. Wolfowitz. Minimax estimates of the mean of a normal distribution with known variance. *The Annals of Mathematical Statistics*, 21:218–230, 1950.

Yichao Wu and Yufeng Liu. Variable selection in quantile regression. *Statistica Sinica*, 19 (2):801, 2009.

Tong Zhang. Some sharp performance bounds for least squares regression with l1 regularization. *The Annals of Statistics*, 37(5A):2109–2144, 2009.

Yuchen Zhang, John C Duchi, and Martin J Wainwright. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *arXiv preprint arXiv:1305.5029*, 2013.

Shuheng Zhou. Restricted eigenvalue conditions on subgaussian random matrices. *arXiv preprint arXiv:0912.4045*, 2009.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429, 2006.

Hui Zou and Ming Yuan. Composite quantile regression and the oracle model selection theory. *The Annals of Statistics*, 36(3):1108–1126, 2008.

Journal of Machine Learning Research 17 (2016) 1-30 Submitted 12/10; Revised 9/14; Published 4/16

Analysis of Classification-based Policy Iteration Algorithms

Alessandro Lazaric[1] alessandro.lazaric@inria.fr
Mohammad Ghavamzadeh[2,1] mohammad.ghavamzadeh@inria.fr
Rémi Munos[3,1] remi.munos@inria.fr
[1] *INRIA Lille - Team SequeL, France*
[2] *Adobe Research, USA*
[3] *Google DeepMind, UK*

Editor: Shie Mannor

Abstract

We introduce a variant of the classification-based approach to policy iteration which uses a cost-sensitive loss function weighting each classification mistake by its actual *regret*, that is, the difference between the action-value of the greedy action and of the action chosen by the classifier. For this algorithm, we provide a full finite-sample analysis. Our results state a performance bound in terms of the number of policy improvement steps, the number of rollouts used in each iteration, the capacity of the considered policy space (classifier), and a capacity measure which indicates how well the policy space can approximate policies that are greedy with respect to any of its members. The analysis reveals a tradeoff between the estimation and approximation errors in this classification-based policy iteration setting. Furthermore it confirms the intuition that classification-based policy iteration algorithms could be favorably compared to value-based approaches when the policies can be approximated more easily than their corresponding value functions. We also study the consistency of the algorithm when there exists a sequence of policy spaces with increasing capacity.

Keywords: reinforcement learning, policy iteration, classification-based approach to policy iteration, finite-sample analysis.

1. Introduction

Policy iteration (Howard, 1960) is a method of computing an optimal policy for any given Markov decision process (MDP). It is an iterative procedure that discovers a deterministic optimal policy by generating a sequence of monotonically improving policies. Each iteration k of this algorithm consists of two phases: *policy evaluation* in which the action-value function Q^{π_k} of the current policy π_k is computed (i.e., the expected sum of discounted rewards collected by acting according to policy π_k), and *policy improvement* in which the new (improved) policy π_{k+1} is generated as the greedy policy w.r.t. Q^{π_k}, that si, $\pi_{k+1}(x) = \arg\max_{a \in \mathcal{A}} Q^{\pi_k}(x,a)$. Unfortunately, in MDPs with large (or continuous) state and action spaces, the policy evaluation problem cannot be solved exactly and approximation techniques are required. In approximate policy iteration (API), a function approximation scheme is usually employed in the policy evaluation phase. The most common approach is to find a good approximation of the value function of π_k in a real-valued function space (see e.g., Bradtke and Barto 1996; Lagoudakis and Parr 2003a). The main drawbacks of this approach are: **1)** the action-value function, Q^{π_k}, is not known in advance and its high-quality samples are often very expensive to obtain, if this option is possible at all, **2)** it is often difficult to find a function space rich enough to represent the action-value function accurately, and thus, careful hand-tuning is needed to achieve satisfactory results, **3)** for the success of policy iteration, it is not necessary to estimate Q^{π_k} accurately at every state-action pair, what is important is to have an approximation of the action-value function whose greedy policy improves over the previous policy, and **4)** this method may not be the right choice in domains where good policies are easier to represent and learn than the corresponding value functions.

To address the above issues, mainly **3** and **4**,[1] variants of API have been proposed that replace the usual value function learning step (approximating the action-value function over the entire state-action space) with a learning step in a policy space (Lagoudakis and Parr, 2003b; Fern et al., 2004). The main idea is to cast the policy improvement step as a *classification* problem. The training set is generated using rollout estimates of Q^π over a finite number of states $\mathcal{D} = \{x_i\}_{i=1}^N$, called the *rollout set*, and for any action $a \in \mathcal{A}$.[2] For each $x \in \mathcal{D}$, if the estimated value $\widehat{Q}^\pi(x, a^+)$ for action a^+ is greater than the estimated value of all other actions with *high confidence*, the state-action pair (x, a^+) is added to the training set with a positive label. In this case, (x, a) for the rest of the actions are labeled negative and added to the training set. The policy improvement step thus reduces to solving a classification problem to find a policy in a given hypothesis space that best predicts the greedy action at every state. Although whether selecting a suitable policy space is any easier than a value function space is highly debatable, we can argue that the classification-based API methods can be advantageous in problems where good policies are easier to represent and learn than their value functions.

The classification-based API algorithms can be viewed as a type of reduction from reinforcement learning (RL) to classification, that is, solving a MDP by generating and solving a series of classification problems. There have been other proposals for reducing RL to classification. Langford and Zadrozny (2005) provided a formal reduction from RL to classification, showing that ϵ-accurate classification implies near optimal RL. This approach uses an optimistic variant of sparse sampling to generate h classification problems, one for each horizon time step. The main limitation of this work is that it does not provide a practical method for generating training examples for these classification problems. Bagnell et al. (2003) introduced an algorithm, called policy search by dynamic programming (PSDP) for learning non-stationary policies in RL. For a specified horizon h, their approach learns a sequence of h policies. At each iteration, all policies are fixed except for one, which is optimized by forming a classification problem via policy rollout. Perhaps the closest approach to the classification-based API methods proposed and analyzed in this paper is the group of algorithms that are introduced and analyzed in (Kakade and Langford, 2002) and (Kakade, 2003) under the name *conservative policy iteration* (CPI).[3] The main algorithmic difference between CPI and the classification-based API methods studied in

1. The first drawback is shared by all reinforcement learning algorithms and the second one is common to all practical applications of machine learning methods.
2. It is worth stressing that Q^π is estimated just on states in $\mathcal{D}$ and not over the entire state-action space.
3. While in (Kakade and Langford, 2002) the algorithm is presented as a rollout value function based approach, in the more detailed description and analysis of CPI found in (Kakade, 2003), the algorithm is presented as a classification-based API method.

this paper is that while the output of the classifier is directly assigned to the next policy in our algorithms, CPI algorithms perform a more conservative policy update in which the new policy π_{k+1} is a mixture distribution of the current policy π_k and the output of the classifier (policies might be stochastic). This conservative update gives CPI two desirable properties: **1)** it guarantees to improve the policy at each iteration, that is, the value function of π_{k+1} is larger than the value function of π_k, and **2)** it has a stopping condition based on the quality of the generated policy (it stops whenever it cannot guarantee that the new policy has a better performance than the previous one). These properties can potentially make CPI a very appealing API algorithm, mainly because other API methods have no guarantee to generate monotonically improving policies and they only converge to a region (i.e., they may repeatedly oscillate among different policies). This includes both value function based API algorithms such as LSPI (Lagoudakis and Parr, 2003a) and classification-based API methods. However, Ghavamzadeh and Lazaric (2012) showed that CPI's desirable properties do not come for free. The analysis of Ghavamzadeh and Lazaric (2012) reveals that in order to achieve the same level of accuracy, CPI requires more iterations, and thus, more samples than the classification-based API algorithms proposed in this paper. This indicates that although CPI's conservative update allows it to have a monotonically improving behavior, it slows down the algorithm and increases its sample complexity. On the other hand, CPI retains the advantage of a concentrability coefficient (or density ratios), which can be much smaller for CPI whenever prior knowledge about the stationary distribution of the optimal policy is used to properly tune the sampling distribution.[4] Nonetheless, Ghavamzadeh and Lazaric (2012) further show that CPI may converge to suboptimal policies whose performance is not better than those returned by the algorithms studied in this paper. Given the advantages and disadvantages, the classification-based API algorithm proposed in this paper and CPI remain two valid alternatives to implement the general approximate policy iteration scheme.

Although the classification-based API algorithms have been successfully applied to benchmark problems (Lagoudakis and Parr, 2003b; Fern et al., 2004) and have been modified to become more computationally efficient (Dimitrakakis and Lagoudakis, 2008b), a full theoretical understanding of them is still lacking. Fern et al. (2006) and Dimitrakakis and Lagoudakis (2008a) provide a preliminary theoretical analysis of their algorithm. In particular, they both bound the difference in performance at each iteration between the learned policy and the true greedy policy. Their analysis is limited to one step policy update (they do not show how the error in the policy update is propagated through the iterations of the API algorithm) and either to finite class of policies (in Fern et al., 2006) or to a specific architecture (a uniform grid in Dimitrakakis and Lagoudakis, 2008a). Moreover, the bound reported in (Fern et al., 2006) depends inversely on the minimum Q-value gap between a greedy and a sub-greedy action over the state space. In some classes of MDPs this gap can be arbitrarily small so that the learned policy can be arbitrarily worse than the greedy policy. In order to deal with this problem Dimitrakakis and Lagoudakis (2008a) assume the action-value functions to be smooth and the probability of states with a small Q-value gap to be small.

4. In Section 4.2 we show that the same concentrability coefficient describes the performance loss of DPI whenever it converges to a fixed point.

In this paper, we derive a full finite-sample analysis of a classification-based API algorithm, called *direct policy iteration* (DPI). It is based on a cost-sensitive loss function weighting each classification error by its actual *regret*, that is, the difference between the action-value of the greedy action and of the action chosen by DPI. A partial analysis of DPI is developed in (Lazaric et al., 2010) where it is shown that using this loss, we are able to derive a performance bound with no dependency on the minimum Q-value gap and no assumption on the probability of states with small Q-value gap. In this paper we provide a more thorough analysis which further extends those in (Fern et al., 2006) and (Dimitrakakis and Lagoudakis, 2008a) by considering arbitrary policy spaces, and by showing how the error at each step is propagated through the iterations of the API algorithm. We also analyze the consistency of DPI when there exists a sequence of policy spaces with increasing capacity. We first use a counterexample and show that DPI is not consistent in general, and then prove its consistency for the class of Lipschitz MDPs. We conclude the paper with a discussion on different theoretical and practical aspects of DPI. Since its introduction by Lagoudakis and Parr (2003b) and Fern et al. (2004) and its extension by Lazaric et al. (2010), the idea of classification-based API has been integrated in a variety of different dynamic programming algorithms (see e.g., Gabillon et al. 2011; Scherrer et al. 2012; Farahmand et al. 2013) and it has been shown to be empirically competitive in a series of testbeds and challenging applications (see e.g., Farahmand et al. 2013; Gabillon et al. 2013).

The rest of the paper is organized as follows. In Section 2, we define the basic concepts and set up the notation used in the paper. Section 3 introduces the general classification-based approach to policy iteration and details the DPI algorithm. In Section 4, we provide a finite-sample analysis for the DPI algorithm. The approximation error and the consistency of the algorithm are discussed in Section 5. While all the main results are derived in case of two actions, that is, $|\mathcal{A}| = 2$, in Section 6 we show how they can be extended to the general case of multiple actions. In Section 7, we conclude the paper and discuss the obtained results.

2. Preliminaries

In this section, we set the notation used throughout the paper. A discounted Markov decision process (MDP) $\mathcal{M}$ is a tuple $\langle \mathcal{X}, \mathcal{A}, r, p, \gamma \rangle$, where the state space $\mathcal{X}$ is a bounded closed subset of a Euclidean space $\mathbb{R}^d$, the set of actions $\mathcal{A}$ is finite ($|\mathcal{A}| < \infty$), the reward function $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is uniformly bounded by $R_{\max}$, the transition model $p(\cdot|x,a)$ is a distribution over $\mathcal{X}$, and $\gamma \in (0,1)$ is a discount factor. Let $\mathcal{B}^V(\mathcal{X}; V_{\max})$ and $\mathcal{B}^Q(\mathcal{X} \times \mathcal{A}; Q_{\max})$ be the space of Borel-measurable value and action-value functions bounded by $V_{\max}$ and $Q_{\max}$ ($V_{\max} = Q_{\max} = \frac{R_{\max}}{1-\gamma}$), respectively. We also use $\mathcal{B}^\pi(\mathcal{X})$ to denote the space of deterministic policies $\pi : \mathcal{X} \to \mathcal{A}$. The value function of a policy π, V^π, is the unique fixed-point of the Bellman operator $\mathcal{T}^\pi : \mathcal{B}^V(\mathcal{X}; V_{\max}) \to \mathcal{B}^V(\mathcal{X}; V_{\max})$ defined by

$$(\mathcal{T}^\pi V)(x) = r\big(x, \pi(x)\big) + \gamma \int_{\mathcal{X}} p\big(dy|x, \pi(x)\big) V(y).$$

The action-value function Q^π is defined as

$$Q^\pi(x,a) = r(x,a) + \gamma \int_{\mathcal{X}} p(dy|x,a) V^\pi(y).$$

Similarly, the optimal value function, V^*, is the unique fixed-point of the optimal Bellman operator $\mathcal{T} : \mathcal{B}^V(\mathcal{X}; V_{\max}) \to \mathcal{B}^V(\mathcal{X}; V_{\max})$ defined as

$$(\mathcal{T}V)(x) = \max_{a \in \mathcal{A}} \left[r(x,a) + \gamma \int_{\mathcal{X}} p(dy|x,a) V(y) \right],$$

and the optimal action-value function Q^* is defined by

$$Q^*(x,a) = r(x,a) + \gamma \int_{\mathcal{X}} p(dy|x,a) V^*(y).$$

We say that a deterministic policy $\pi \in \mathcal{B}^\pi(\mathcal{X})$ is *greedy* w.r.t. an action-value function Q, if $\pi(x) \in \arg\max_{a \in \mathcal{A}} Q(x,a), \forall x \in \mathcal{X}$. Greedy policies are important because any greedy policy w.r.t. Q^* is optimal. We define the greedy policy operator $\mathcal{G} : \mathcal{B}^\pi(\mathcal{X}) \to \mathcal{B}^\pi(\mathcal{X})$ as[5]

$$(\mathcal{G}\pi)(x) = \arg\max_{a \in \mathcal{A}} Q^\pi(x,a). \tag{1}$$

In the analysis of this paper, $\mathcal{G}$ plays a role similar to the one played by the optimal Bellman operator, $\mathcal{T}$, in the analysis of the fitted value iteration algorithm (Munos and Szepesvári 2008, Section 5).

3. The DPI Algorithm

In this section, we outline the direct policy iteration (DPI) algorithm. DPI shares the same structure as the algorithms in (Lagoudakis and Parr, 2003b) and (Fern et al., 2004). Although it can benefit from improvements in **1)** selecting states for the rollout set $\mathcal{D}$, **2)** the criteria used to add a sample to the training set, and **3)** the rollout strategy, as discussed in (Lagoudakis and Parr, 2003b) and (Dimitrakakis and Lagoudakis, 2008b), here we consider its basic form in order to ease the analysis.

DPI receives as input a policy space Π and starting from an arbitrary policy $\pi_0 \in \Pi$, at each iteration k, it computes a new policy π_{k+1} from π_k, as the best approximation of $\mathcal{G}\pi_k$, by solving a cost-sensitive classification problem. More formally, DPI is based on the following loss function:

Definition 1 *The loss function at iteration k for a policy π is denoted by $\ell_{\pi_k}(\cdot;\pi)$ and is defined as*

$$\ell_{\pi_k}(x;\pi) = \max_{a \in \mathcal{A}} Q^{\pi_k}(x,a) - Q^{\pi_k}\big(x, \pi(x)\big), \qquad \forall x \in \mathcal{X}.$$

Given a distribution ρ over $\mathcal{X}$, we define the expected error as the expectation of the loss function $\ell_{\pi_k}(\cdot;\pi)$ according to ρ,[6]

$$\mathcal{L}_{\pi_k}(\rho;\pi) = \int_{\mathcal{X}} \ell_{\pi_k}(x;\pi)\rho(dx) = \int_{\mathcal{X}} \Big[\max_{a \in \mathcal{A}} Q^{\pi_k}(x,a) - Q^{\pi_k}\big(x,\pi(x)\big) \Big] \rho(dx). \tag{2}$$

5. In (1), ties among the actions maximizing $Q^\pi(x,a)$ are broken in an arbitrary but consistent manner.
6. The expected error $\mathcal{L}_{\pi_k}(\rho;\pi)$ can be seen as the $L_{1,\rho}$-norm of the loss function $\ell_{\pi_k}(\cdot;\pi)$.

Input: policy space $\Pi \subseteq \mathcal{B}^\pi(\mathcal{X})$, state distribution ρ, number of rollout states N, number of rollouts per state-action pair M, rollout horizon H
Initialize: Let $\pi_0 \in \Pi$ be an arbitrary policy
for $k = 0, 1, 2, \ldots$ **do**
  Construct the rollout set $\mathcal{D}_k = \{x_i\}_{i=1}^N$, $x_i \overset{\text{iid}}{\sim} \rho$
  for all states $x_i \in \mathcal{D}_k$ and actions $a \in \mathcal{A}$ **do**
    for $j = 1$ to M **do**
      Perform a rollout according to policy π_k and return
      $$R_j^{\pi_k}(x_i,a) = r(x_i,a) + \sum_{t=1}^{H-1} \gamma^t r\big(x^t, \pi_k(x^t)\big),$$
      with $x^t \sim p\big(\cdot\,|x^{t-1}, \pi_k(x^{t-1})\big)$ and $x^1 \sim p(\cdot|x_i,a)$
    end for
    $\widehat{Q}^{\pi_k}(x_i,a) = \frac{1}{M}\sum_{j=1}^M R_j^{\pi_k}(x_i,a)$
  end for
  $\pi_{k+1} = \arg\min_{\pi \in \Pi} \widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)$ **(classifier)**
end for

Figure 1: The Direct Policy Iteration (DPI) algorithm.

While in (Lagoudakis and Parr, 2003b) the goal is to minimize the number of misclassifications using a 0/1 loss function, DPI learns a policy trying to minimize the error $\mathcal{L}_{\pi_k}$. Similar to other classification-based RL algorithms (Bagnell et al., 2003; Kakade, 2003; Fern et al., 2004; Langford and Zadrozny, 2005; Li et al., 2007), DPI does not focus on finding a uniformly accurate approximation of the actions taken by the greedy policy, but rather on finding actions leading to a similar performance. This is consistent with the final objective of policy iteration, which is to obtain a policy with similar performance to an optimal policy, and not necessarily one that takes actions similar to an optimal policy.[7]

As illustrated in Figure 1, for each state $x_i \in \mathcal{D}_k$ and for each action $a \in \mathcal{A}$, an estimate of the action-value function of the current policy is computed through M independent rollouts. A H-horizon rollout of a policy π_k for a state-action pair (x_i, a) is

$$R^{\pi_k}(x_i,a) = r(x_i,a) + \sum_{t=1}^{H-1} \gamma^t r\big(x^t, \pi_k(x^t)\big), \tag{3}$$

where $x^t \sim p\big(\cdot\,|x^{t-1}, \pi_k(x^{t-1})\big)$ and $x^1 \sim p(\cdot|x_i,a)$. The action-value function estimation is then obtained by averaging M independent rollouts $\{R_j^{\pi_k}(x_i,a)\}_{1 \le j \le M}$ as

$$\widehat{Q}^{\pi_k}(x_i,a) = \frac{1}{M}\sum_{j=1}^M R_j^{\pi_k}(x_i,a). \tag{4}$$

Given the outcome of the rollouts, the empirical loss is defined as follows:

7. We refer the readers to (Li et al., 2007) for a simple example in which a good approximation (in terms of the number of mismatch in selecting actions) of the greedy policy has a very poor performance w.r.t. it.

Definition 2 *For any $x \in \mathcal{D}_k$, the empirical loss function at iteration k for a policy π is*

$$\widehat{\ell}_{\pi_k}(x;\pi) = \max_{a\in\mathcal{A}} \widehat{Q}^{\pi_k}(x,a) - \widehat{Q}^{\pi_k}\big(x,\pi(x)\big),$$

where $\widehat{Q}^{\pi_k}(x,a)$ is a H-horizon rollout estimation of the action-value of π_k in (x,a) as defined by Equations 3 and 4. Similar to Definition 1, the empirical error is defined as the average over states in $\mathcal{D}_k$ of the empirical loss,[8]

$$\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi) = \frac{1}{N}\sum_{i=1}^{N}\Big[\max_{a\in\mathcal{A}} \widehat{Q}^{\pi_k}(x_i,a) - \widehat{Q}^{\pi_k}\big(x_i,\pi(x_i)\big)\Big],$$

where $\widehat{\rho}$ is the empirical distribution induced by the samples in $\mathcal{D}_k$.

Finally, DPI makes use of a classifier which returns a policy that minimizes the empirical error $\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)$ over the policy space Π (see Section 6.2 for further details on the implementation of such a classifier). Note that this gap-weighted loss function has been previously used in other algorithms such as PSDP (Bagnell et al., 2003) and CPI (Kakade, 2003). Furthermore, while here we use a loss perspective, the minimization of the empirical loss $\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)$ is equivalent to the maximization of the average Q-value and the theoretical development in the next sections would apply mostly unchanged.

4. Finite-sample Analysis of DPI

In this section, we first provide a finite-sample analysis of the error incurred at each iteration of DPI in Theorem 5, and then show how this error is propagated through the iterations of the algorithm in Theorem 7. In the analysis, we explicitly assume that the action space contains only two actions, that is, $\mathcal{A} = \{a_1, a_2\}$ and $|\mathcal{A}| = 2$. We will discuss this assumption and other theoretical and practical aspects of DPI in Section 6.

4.1 Error Bound at Each Iteration

Here we study the error incurred at each iteration k of the DPI algorithm. In particular, we compare the quality of the policy π_{k+1} obtained by minimizing the empirical loss $\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\cdot)$ to the policy that better approximate the greedy policy $\mathcal{G}\pi_k$ among the policies in Π (i.e., the policy minimizing the expected loss $\mathcal{L}_{\pi_k}(\rho;\cdot)$). Comparing the definition of the expected and empirical errors, we notice that there are three sources of error in the algorithm of Figure 1. The first one depends on the use of a finite number of samples, i.e., N states in the rollout set, to approximate the expectation w.r.t. the distribution ρ. The second one is due to using rollouts with finite horizon H to approximate the action-value function Q^{π_k} of the current policy π_k. Finally, the third one depends on the use of M rollouts to approximate the action-value function of the current policy for any of the N states in the rollout set $\mathcal{D}_k$ and any action in the action space $\mathcal{A}$. Before stating our main result, i.e., Theorem 5, we prove bounds for the first and third sources of errors in Lemmas 3 and 4, and have a discussion on the effect of finite horizon rollouts to approximate the action-value function.

8. Alternatively, the empirical error $\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)$ can be seen as the $L_{1,\widehat{\rho}}$-norm of the empirical loss $\widehat{\ell}_{\pi_k}(\cdot;\pi)$.

The proofs of the lemmas rely on tools from concentration inequalities of empirical processes and statistical learning theory (notably VC-bounds), and they are reported in Appendix A. Lemma 3 shows that the difference between the approximation obtained by averaging over the samples in the rollout set and the true expectation can be controlled and reduces to zero as the number of states in the rollout set N grows.

Lemma 3 *Let Π be a policy space with finite VC-dimension $h = VC(\Pi) < \infty$ and $N > 0$ be the number of states in the rollout set $\mathcal{D}_k$, drawn i.i.d. from the state distribution ρ at iteration k, then*

$$\mathbb{P}_{\mathcal{D}_k}\left[\sup_{\pi\in\Pi}\Big|\mathcal{L}_{\pi_k}(\widehat{\rho};\pi) - \mathcal{L}_{\pi_k}(\rho;\pi)\Big| > \epsilon\right] \le \delta\,,$$

where $\mathbb{P}_{\mathcal{D}_k}[\cdot]$ is the probability w.r.t. the random rollout set $\mathcal{D}_k$ conditioned on all the previous iterations[9] *and $\epsilon = 16 Q_{\max}\sqrt{\frac{2}{N}\left(h\log\frac{eN}{h} + \log\frac{8}{\delta}\right)}$.*

Proof See Appendix A. ■

The second source of error in the algorithm of Figure 1 is due to the use of finite horizon rollout estimates of the action-value function on the states in the rollout set. We define the true action-value for a state-action pair (x,a) with a finite horizon H as

$$Q_H^{\pi_k}(x,a) = \mathbb{E}\left[r(x,a) + \sum_{t=1}^{H-1}\gamma^t r\big(x^t,\pi_k(x^t)\big)\right].$$

It is easy to see that the H-horizon rollout estimates are stochastic estimations of $Q_H^{\pi_k}(x,a)$ which in turn satisfy

$$|Q^{\pi_k}(x,a) - Q_H^{\pi_k}(x,a)| = \left|\mathbb{E}\left[\sum_{t=H}^{\infty}\gamma^t r\big(x^t,\pi_k(x^t)\big)\right]\right| \le \gamma^H Q_{\max}. \tag{5}$$

In the proof of the main theorem we also need to bound the difference between the action values (of the N states in the rollout set $\mathcal{D}_k$ and all the actions in the action space $\mathcal{A}$) estimated with M rollouts and their true values. We thus report the following lemma to bound this source of error.

Lemma 4 *Let Π be a policy space with finite VC-dimension $h = VC(\Pi) < \infty$ and $x_1,\ldots,x_N$ be an arbitrary sequence of states. In each state we simulate M independent truncated rollouts, then*

$$\mathbb{P}_{\mathcal{D}_k}\left[\sup_{\pi\in\Pi}\Big|\frac{1}{N}\sum_{i=1}^{N}\frac{1}{M}\sum_{j=1}^{M}R_j^{\pi_k}(x_i,\pi(x_i)) - \frac{1}{N}\sum_{i=1}^{N}Q_H^{\pi_k}(x_i,\pi(x_i))\Big| > \epsilon\right] \le \delta\,,$$

with $\epsilon = 8(1-\gamma^H)Q_{\max}\sqrt{\frac{2}{MN}\left(h\log\frac{eMN}{h} + \log\frac{8}{\delta}\right)}$.

9. More precisely, the conditioning is w.r.t. all the rollout sets $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{k-1}$, which define all the policies returned at iterations 0 to $k-1$, including π_k.

Proof See Appendix A. ■

We are now ready to prove the main result of this section. We show a high probability bound on $\mathcal{L}_{\pi_k}(\rho;\pi_{k+1})$, the expected error at any iteration k of the DPI algorithm.

Theorem 5 *Let Π be a policy space with finite VC-dimension $h = VC(\Pi) < \infty$ and ρ be a distribution over the state space $\mathcal{X}$. Let N be the number of states in $\mathcal{D}_k$ drawn i.i.d. from ρ at each iteration, H be the horizon of the rollouts, and M be the number of rollouts per state-action pair used in the estimation of the action-value functions. Let $\pi_{k+1} = \arg\min_{\pi\in\Pi} \widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)$ be the policy computed at the k-th iteration of* DPI. *Then, for any $\delta > 0$, we have*

$$\mathcal{L}_{\pi_k}(\rho;\pi_{k+1}) \le \inf_{\pi\in\Pi} \mathcal{L}_{\pi_k}(\rho;\pi) + 2(\epsilon_1+\epsilon_2+\gamma^H Q_{\max}), \tag{6}$$

with probability $1-\delta$, where

$$\epsilon_1 = 16Q_{\max}\sqrt{\frac{2}{N}\left(h\log\frac{eN}{h}+\log\frac{32}{\delta}\right)}, \quad \epsilon_2 = 8(1-\gamma^H)Q_{\max}\sqrt{\frac{2}{MN}\left(h\log\frac{eMN}{h}+\log\frac{32}{\delta}\right)}.$$

Remark (dependency on M and N). The bound in Equation 6 can be decomposed into an approximation error ($\inf_{\pi\in\Pi}\mathcal{L}_{\pi_k}(\rho;\pi)$) and an estimation error consisting of three terms ϵ_1, ϵ_2, and $\gamma^H Q_{\max}$. This is similar to generalization bounds in classification, where the approximation error is the distance between the target function (here the greedy policy w.r.t. π_k) and the function space Π. The first estimation term, ϵ_1, grows with the capacity of Π, measured by its VC-dimension h, and decreases with the number of sampled states N. Thus in order to avoid overfitting, we should have $N \gg h$. The second estimation term, ϵ_2, comes from the error in the estimation of the action-values due to the finite number of rollouts M. It is important to note the nice rate of $1/\sqrt{MN}$ instead of $1/\sqrt{M}$. This is due to the fact that we do not need a uniformly good estimation of the action-value function at all sampled states, but only an averaged estimation of those values at the sampled points. An important consequence of this is that the algorithm works perfectly well if we consider only $M = 1$ rollout per state-action. Therefore, given a fixed budget (number of rollouts per iteration) and a fixed rollout horizon H, the best allocation of M and N would be to choose $M = 1$ and sample as many states as possible, thus, reducing the risk of overfitting. The third estimation term, $\gamma^H Q_{\max}$, is due to the fact that we consider a finite horizon H for the rollouts. This term decreases exponentially fast as the rollout horizon H grows.

Remark (choice of the parameters). In Remark 1, we considered the tradeoff between the number of states, N, and the number of rollouts at each state-action pair, M, when a finite budget (number of rollouts per iteration) is given. It is also interesting to analyze the tradeoff with the rollout horizon, H, when the number of interactions with the generative model is fixed to a maximum value $S = N \times M \times H$. The term γ^H decreases exponentially with a rate depending on γ, thus, it easy to see that by setting $M = 1$, a rough optimization of the bound in Theorem 5 leads to $H = O(\frac{\log S}{\log 1/\gamma})$ and $N = O(S/H)$. Similar to the tradeoff between M and N, this suggests that most of the resources should be allocated so as to have

a large number of states, while the rollouts may have a fairly short horizon. Nonetheless, it is clear from the value of H that the discount factor is critical, and when it approaches 1 the horizon increases correspondingly.

Remark (comparison with other classification-based methods). The performance of classification-based methods have been analyzed before by Fern et al. (2006) and Dimitrakakis and Lagoudakis (2008a). As discussed in the Introduction, the bound reported in Theorem 5 for DPI improves existing results over multiple dimensions. Using a regret-based loss function allows DPI to remove the inverse dependency on the smallest gap appearing in the analysis by Fern et al. (2006). Furthermore, this also allows us to drop the Lipschitz and the separability assumptions employed by Dimitrakakis and Lagoudakis (2008a) and extend the result to any sampling strategy ρ. In this sense, Theorem 5 provides a stronger and more general guarantee on the performance of DPI, where the only constraint is relative to using a policy space with finite VC-dimension, conditioned enjoyed by many standard classifiers (e.g., linear separators, neural networks).

Proof *[Theorem 5]* Let $a^+(x) = \arg\max_{a\in\mathcal{A}} Q^{\pi_k}(x,a)$ be the greedy action in state x.[10] We prove the following series of inequalities:

$$\begin{aligned}
\mathcal{L}_{\pi_k}(\rho;\pi_{k+1}) &\overset{(a)}{\le} \mathcal{L}_{\pi_k}(\widehat{\rho};\pi_{k+1}) + \epsilon_1 && \text{w.p. } 1-\delta' \\
&= \frac{1}{N}\sum_{i=1}^N \Big[Q^{\pi_k}(x_i,a^+) - Q^{\pi_k}\big(x_i,\pi_{k+1}(x_i)\big)\Big] + \epsilon_1 \\
&\overset{(b)}{\le} \frac{1}{N}\sum_{i=1}^N \Big[Q^{\pi_k}(x_i,a^+) - Q_H^{\pi_k}\big(x_i,\pi_{k+1}(x_i)\big)\Big] + \epsilon_1 + \gamma^H Q_{\max} && \text{w.p. } 1-\delta' \\
&\overset{(c)}{\le} \frac{1}{N}\sum_{i=1}^N \Big[Q^{\pi_k}(x_i,a^+) - \widehat{Q}^{\pi_k}\big(x_i,\pi_{k+1}(x_i)\big)\Big] + \epsilon_1 + \epsilon_2 + \gamma^H Q_{\max} && \text{w.p. } 1-2\delta' \\
&\overset{(d)}{\le} \frac{1}{N}\sum_{i=1}^N \Big[Q^{\pi_k}(x_i,a^+) - \widehat{Q}^{\pi_k}\big(x_i,\pi^+(x_i)\big)\Big] + \epsilon_1 + \epsilon_2 + \gamma^H Q_{\max} \\
&\overset{(e)}{\le} \frac{1}{N}\sum_{i=1}^N \Big[Q^{\pi_k}(x_i,a^+) - Q_H^{\pi_k}\big(x_i,\pi^+(x_i)\big)\Big] + \epsilon_1 + 2\epsilon_2 + \gamma^H Q_{\max} && \text{w.p. } 1-3\delta' \\
&\overset{(f)}{\le} \frac{1}{N}\sum_{i=1}^N \Big[Q^{\pi_k}(x_i,a^+) - Q^{\pi_k}\big(x_i,\pi^+(x_i)\big)\Big] + \epsilon_1 + 2(\epsilon_2 + \gamma^H Q_{\max}) && \text{w.p. } 1-3\delta' \\
&= \mathcal{L}_{\pi_k}(\widehat{\rho};\pi^+) + \epsilon_1 + 2(\epsilon_2+\gamma^H Q_{\max}) \\
&\overset{(g)}{\le} \mathcal{L}_{\pi_k}(\rho;\pi^+) + 2(\epsilon_1+\epsilon_2+\gamma^H Q_{\max}) && \text{w.p. } 1-4\delta' \\
&= \inf_{\pi'\in\Pi} \mathcal{L}_{\pi_k}(\rho;\pi') + 2(\epsilon_1+\epsilon_2+\gamma^H Q_{\max}).
\end{aligned}$$

The statement of the theorem is obtained by setting $\delta' = \delta/4$.

10. To simplify the notation, we remove the dependency of a^+ on states and use a^+ instead of $a^+(x)$ in the following.

(a) It is an immediate application of Lemma 3, bounding the difference between $\mathcal{L}_{\pi_k}(\rho;\pi)$ and $\mathcal{L}_{\pi_k}(\widehat{\rho};\pi)$ for any policy $\pi \in \Pi$.
(b) We use the inequality in Equation 5.
(c) Here we introduce the estimated action-value function $\widehat{Q}^{\pi_k}$ by bounding

$$\sup_{\pi\in\Pi}\left[\frac{1}{N}\sum_{i=1}^{N}\widehat{Q}^{\pi_k}\big(x_i,\pi(x_i)\big)-\frac{1}{N}\sum_{i=1}^{N}Q_H^{\pi_k}\big(x_i,\pi(x_i)\big)\right],$$

i.e., the maximum over all the policies in the policy space[11] of the difference between the true action-value function with horizon H and its rollout estimates averaged over the states in the rollout set $\mathcal{D}_k=\{x_i\}_{i=1}^N$. We bound this term using the result of Lemma 4.
(d) From the definition of π_{k+1} in the DPI algorithm (see Figure 1), we have

$$\pi_{k+1}=\arg\min_{\pi\in\Pi}\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)=\arg\max_{\pi\in\Pi}\frac{1}{N}\sum_{i=1}^{N}\widehat{Q}^{\pi_k}\big(x_i,\pi(x_i)\big),$$

thus, $-\frac{1}{N}\sum_{i=1}^{N}\widehat{Q}^{\pi_k}\big(x_i,\pi_{k+1}(x_i)\big)$ can be maximized by replacing π_{k+1} with any other policy, particularly with

$$\pi^+=\arg\inf_{\pi'\in\Pi}\int_{\mathcal{X}}\Big(\max_{a\in\mathcal{A}}Q^{\pi_k}(x,a)-Q^{\pi_k}\big(x,\pi'(x)\big)\Big)\rho(dx).$$

(e)-(f)-(g) The final result follows by the same arguments in steps (a), (b), and (c) but in reversed order. ■

4.2 Error Propagation

In this section, we first show how the expected error is propagated through the iterations of DPI. We then analyze the error between the value function of the policy obtained by DPI after K iterations and the optimal value function. Unlike the per-iteration analysis, in the propagation we consider the general case where the error is evaluated according to a *testing* distribution μ which may differ from the *sampling* distribution ρ used to construct the rollout sets $\mathcal{D}_k$ over iterations.

Before stating the main result, we define the *inherent greedy error* of a policy space Π.

Definition 6 *We define the inherent greedy error of a policy space* $\Pi\subseteq\mathcal{B}^\pi(\mathcal{X})$ *as*

$$d(\Pi,\mathcal{G}\Pi)=\sup_{\pi\in\Pi}\inf_{\pi'\in\Pi}\mathcal{L}_\pi(\rho;\pi').$$

The inherent greedy error is the worst expected error that a error-minimizing policy $\pi'\in\Pi$ can incur in approximating the greedy policy $\mathcal{G}\pi$, for any policy $\pi\in\Pi$. This measures how well Π is able to approximate policies that are greedy w.r.t. any policy in Π.

11. The supremum over all the policies in the policy space Π is due to the fact that π_{k+1} is a random object, whose randomness comes from all the randomly generated samples at the k-th iteration (i.e., the states in the rollout set and all the generated rollouts).

In order to simplify the notation, we introduce P^π as the transition kernel for policy π, i.e., $P^\pi(dy|x)=p\big(dy|x,\pi(x)\big)$. We define the right-linear operator, $P^\pi\cdot$, which maps any $V\in\mathcal{B}^V(\mathcal{X};V_{\max})$ to $(P^\pi V)(x)=\int V(y)P^\pi(dy|x)$, i.e., the expected value of V w.r.t. the next states achieved by following policy π in state x.

From the definitions of ℓ_{π_k}, $\mathcal{T}^\pi$, and $\mathcal{T}$, we have $\ell_{\pi_k}(\pi_{k+1})=\mathcal{T}V^{\pi_k}-\mathcal{T}^{\pi_{k+1}}V^{\pi_k}$. We deduce the following pointwise inequalities:

$$\begin{aligned}V^{\pi_k}-V^{\pi_{k+1}}&=\mathcal{T}^{\pi_k}V^{\pi_k}-\mathcal{T}^{\pi_{k+1}}V^{\pi_k}+\mathcal{T}^{\pi_{k+1}}V^{\pi_k}-\mathcal{T}^{\pi_{k+1}}V^{\pi_{k+1}}\\&\le\ell_{\pi_k}(\pi_{k+1})+\gamma P^{\pi_{k+1}}(V^{\pi_k}-V^{\pi_{k+1}}),\end{aligned}\tag{7}$$

which gives us $V^{\pi_k}-V^{\pi_{k+1}}\le(I-\gamma P^{\pi_{k+1}})^{-1}\ell_{\pi_k}(\pi_{k+1})$. Since $\mathcal{T}V^{\pi_k}\ge\mathcal{T}^{\pi^*}V^{\pi_k}$, we also have

$$\begin{aligned}V^*-V^{\pi_{k+1}}&=\mathcal{T}^{\pi^*}V^*-\mathcal{T}V^{\pi_k}+\mathcal{T}V^{\pi_k}-\mathcal{T}^{\pi_{k+1}}V^{\pi_k}+\mathcal{T}^{\pi_{k+1}}V^{\pi_k}-\mathcal{T}^{\pi_{k+1}}V^{\pi_{k+1}}\\&\le\gamma P^*(V^*-V^{\pi_k})+\ell_{\pi_k}(\pi_{k+1})+\gamma P^{\pi_{k+1}}(V^{\pi_k}-V^{\pi_{k+1}}),\end{aligned}$$

where $P^*=P^{\pi^*}$. Using Equation 7 this yields to

$$\begin{aligned}V^*-V^{\pi_{k+1}}&\le\gamma P^*(V^*-V^{\pi_k})+\big[\gamma P^{\pi_{k+1}}(I-\gamma P^{\pi_{k+1}})^{-1}+I\big]\ell_{\pi_k}(\pi_{k+1})\\&=\gamma P^*(V^*-V^{\pi_k})+(I-\gamma P^{\pi_{k+1}})^{-1}\ell_{\pi_k}(\pi_{k+1}).\end{aligned}$$

Finally, by defining the operator $E_k=(I-\gamma P^{\pi_{k+1}})^{-1}$, which is well defined since $P^{\pi_{k+1}}$ is a stochastic kernel and $\gamma<1$, and by induction, we obtain

$$V^*-V^{\pi_K}\le(\gamma P^*)^K(V^*-V^{\pi_0})+\sum_{k=0}^{K-1}(\gamma P^*)^{K-k-1}E_k\ell_{\pi_k}(\pi_{k+1}).\tag{8}$$

Equation 8 shows how the error at each iteration k of DPI, $\ell_{\pi_k}(\pi_{k+1})$, is propagated through the iterations and appears in the final error of the algorithm: $V^*-V^{\pi_K}$. In particular, the previous equation reveals the final performance loss in a state x is influenced by all the iterations where the losses in different states are combined and weighted according to the distribution obtained as the combination of the P^* and E_k operators. Since we are interested in bounding the final error in μ-norm, which might be different from the sampling distribution ρ, we need to state some assumptions. We first introduce the left-linear operator of the kernel P^π as $\cdot P^\pi$ such that $(\mu P^\pi)(dy)=\int P^\pi(dy|x)\mu(dx)$ for any distribution μ over $\mathcal{X}$. In words, μP^π correspond to the distribution over states obtained by starting from a random state drawn from μ and then taking the action suggested by π.

Assumption 1 *For any policy* $\pi\in\mathcal{B}^\pi(\mathcal{X})$ *and any non-negative integers* s *and* t*, there exists a constant* $C_{\mu,\rho}(s,t)<\infty$ *such that* $\mu(P^*)^s(P^\pi)^t\le C_{\mu,\rho}(s,t)\rho$.[12] *We assume that the cumulative coefficient* $C_{\mu,\rho}=(1-\gamma)^2\sum_{s=0}^{\infty}\sum_{t=0}^{\infty}\gamma^{s+t}C_{\mu,\rho}(s,t)$ *is bounded, i.e.,* $C_{\mu,\rho}<\infty$.

Assumption 2 *For any* $x\in\mathcal{X}$ *and any* $a\in\mathcal{A}$*, there exist a constant* $C_\rho<\infty$ *such that* $p(\cdot|x,a)\le C_\rho\rho(\cdot)$.

12. Given two distributions P and Q on $\mathcal{X}$ and a real constant $c>0$, $P\le cQ$ is equivalent to the condition $\forall B\subseteq\mathcal{X}$, $P(B)\le cQ(B)$.

Note that *concentrability coefficients* similar to $C_{\mu,\rho}$ and C_ρ were previously used in the L_p-analysis of fitted value iteration (Munos, 2007; Munos and Szepesvári, 2008) and approximate policy iteration (Antos et al., 2008). See also Farahmand et al. (2010) for a more refined analysis. We now state our main result.

Theorem 7 *Let Π be a policy space with finite VC-dimension h and π_K be the policy generated by* DPI *after K iterations. Let M be the number of rollouts per state-action and N be the number of samples drawn i.i.d. from a distribution ρ over $\mathcal{X}$ at each iteration of* DPI. *Then, for any $\delta > 0$, we have*

$$||V^* - V^{\pi_K}||_{1,\mu} \le \frac{C_{\mu,\rho}}{(1-\gamma)^2}\Big[d(\Pi, \mathcal{G}\Pi) + 2(\epsilon_1 + \epsilon_2 + \gamma^H Q_{\max})\Big] + \frac{2\gamma^K R_{\max}}{1-\gamma}, \quad \textit{(under Asm. 1)}$$

$$||V^* - V^{\pi_K}||_{\infty} \le \frac{C_{\rho}}{(1-\gamma)^2}\Big[d(\Pi, \mathcal{G}\Pi) + 2(\epsilon_1 + \epsilon_2 + \gamma^H Q_{\max})\Big] + \frac{2\gamma^K R_{\max}}{1-\gamma}, \quad \textit{(under Asm. 2)}$$

with probability $1-\delta$, where

$$\epsilon_1 = 16 Q_{\max}\sqrt{\frac{2}{N}\left(h\log\frac{eN}{h} + \log\frac{32K}{\delta}\right)} \quad \textit{and} \quad \epsilon_2 = 8(1-\gamma^H)Q_{\max}\sqrt{\frac{2}{MN}\left(h\log\frac{eMN}{h} + \log\frac{32K}{\delta}\right)}.$$

Remark (sample complexity). From the previous bound on the performance loss of DPI after K iterations, we can deduce the full sample complexity of the algorithm. Let ϵ be the desired performance loss when stopping the algorithm, from the remarks of Theorem 5 and the previous bound, we see that a logarithmic number of iterations $K(\epsilon) = O(\log(1/\epsilon)/(1-\gamma))$ is enough to reduce the last term to $O(\epsilon)$. On the other hand, for the leading term, if we ignore the inherent greedy error, which is a constant bias term and set $M = 1$, the number of samples required per iteration is[13]

$$N(\epsilon) = O\left(\frac{hQ_{\max}^2}{\epsilon^2(1-\gamma)^4}\right),$$

which amounts to a total of $N(\epsilon)K(\epsilon)$ samples across iterations. In this case, the final bound is $||V^* - V^{\pi_K}||_{1,\mu} \le C_{\mu,\rho}(d(\Pi, \mathcal{G}\Pi) + \epsilon)$. As discussed in the Introduction and analyzed in details by Ghavamzadeh and Lazaric (2012), this result is competitive with other approximate policy iteration (API) schemes such as conservative policy iteration (CPI).

Remark (comparison with other value-function-based API). Although a direct and detailed comparison between classification-based and value-function-based approaches to API is not straightforward, it is interesting to discuss their similarities and differences. For value-function-based API, we refer to, e.g., LSPI (Lagoudakis and Parr, 2003a) and in particular the theoretical analysis developed by Lazaric et al. (2012) (see Theorem 8 therein). Although here we analyzed DPI for a generic policy space Π and the performance is evaluated in L_1-norm, while LSPI explicitly relies on linear spaces and the norm is L_2, high-level similarities and differences can be remarked in the performance of the two methods. The structure of the bounds is similar for both methods and notably the dependency on

13. Note that the range of the Q-values $Q_{\max}$ contains an additional factor $1/(1-\gamma)$.

the number of samples N, number of iterations K, and discount factor γ is the same. The major difference lays in the concentrability coefficients and in the shape of the approximation error. While assumption on the coefficients $C_{\mu,\rho}(s,t)$ for DPI is less tight since it requires to bound the distribution $\mu(P^*)^s(P^\pi)^t$ instead of the distribution $\mu P^{\pi_1}\dots P^{\pi_m}$ obtained for any possible sequence of policies, the final coefficients $C_{\mu,\rho}$ are more involved and difficult to compare. On the other hand, it is interesting to notice that the approximation errors share the same structure. In fact, they both consider the worst approximation error w.r.t. all the possible approximation problems that could be faced across iterations. While this reveals the importance of the choice of an appropriate approximation space in both cases, it also supports the claim that a classification-based method may be preferable whenever it is easier to design a set of "good" policies rather than a set of "good" value functions.

Remark (convergence). Similar to other API algorithms, Theorem 7 states that DPI may oscillate over different policies whose performance loss in bounded. Nonetheless, depending on the policy space Π and the MDP at hand, in practice DPI sometimes converges to a fixed policy $\bar\pi$. Let $\mathcal{D} : \mathcal{B}^\pi(\mathcal{X}) \to \mathcal{B}^\pi(\mathcal{X})$ be the policy operator corresponding to the approximation performed by DPI at each iteration (i.e., constructing rollouts from a given policy and solving the classification problem), then DPI can be written compactly as $\pi_{k+1} = \mathcal{D}\mathcal{G}\pi_k$. If DPI converges to $\bar\pi$, then the joint operator $\mathcal{D}\mathcal{G}$ admits $\bar\pi$ as a fixed point, i.e., $\bar\pi = \mathcal{D}\mathcal{G}\bar\pi$ and the per-iteration error $\ell_{\pi_k}(\pi_{k+1})$, which is propagated in the analysis of Theorem 7, converges to $\ell_{\bar\pi}(\bar\pi)$. In this case, the performance loss of $\bar\pi$ can be directly studied as a function of the error $\mathcal{L}_{\bar\pi}(\rho, \bar\pi)$ as (see Munos, 2007, Section 5.2 for a similar argument for approximate value iteration)

$$\begin{aligned} V^* - V^{\bar\pi} &= \mathcal{T}^{\pi^*}V^* - \mathcal{T}V^{\bar\pi} + \mathcal{T}V^{\bar\pi} - V^{\bar\pi} \\ &\le \mathcal{T}^{\pi^*}V^* - \mathcal{T}^{\pi^*}V^{\bar\pi} + \mathcal{T}V^{\bar\pi} - V^{\bar\pi} \\ &= \gamma P^*(V^* - \mathcal{T}^{\pi^*}V^{\bar\pi}) + \mathcal{T}V^{\bar\pi} - \mathcal{T}^{\bar\pi}V^{\bar\pi}. \end{aligned}$$

Using the definition of $\ell_{\bar\pi}(\bar\pi)$ we obtain the following component-wise performance loss

$$V^* - V^{\bar\pi} \le (I - \gamma P^*)^{-1}\ell_{\bar\pi}(\bar\pi).$$

Finally, integrating on both sides w.r.t. the measure μ we have

$$||V^* - V^{\bar\pi}||_{1,\mu} \le \frac{C^*_{\mu,\rho}}{1-\gamma}\Big[\inf_{\pi'\in\Pi}\mathcal{L}_{\bar\pi}(\rho;\pi') + 2(\epsilon_1 + \epsilon_2 + \gamma^H Q_{\max})\Big],$$

where the concentrability coefficient $C^*_{\mu,\rho}$ is such that $\mu\sum_{t=0}^\infty(\gamma P^*)^t \le C^*_{\mu,\rho}\rho$. Unlike the coefficients introduced in Assumption 1, $C^*_{\mu,\rho}$ only involves the optimal policy and notably the discounted stationary distribution of π^*. This term coincides with the coefficient appearing in the performance of CPI and it can be made small by appropriately choosing the sampling distribution ρ when prior knowledge about the states visited by the optimal policy is available.[14] Furthermore, in case of convergence, the dependency on the discount

14. Consider a simple pole balancing problem. In this case, given the desired target distribution μ, it is relatively easy to guess that the optimal policy will mostly visit states close to the equilibrium and define the sampling distribution ρ accordingly.

factor reduces by a factor of $1/(1-\gamma)$ and the approximation error appearing in the final bound no longer depends on the inherent greedy error. It only depends on the loss of the policy that better approximates $\mathcal{G}\bar{\pi}$ in Π.

Proof [Theorem 7] We have $C_{\mu,\rho} \leq C_\rho$ for any μ. Thus, if the L_1-bound holds for any μ, choosing μ to be a Dirac at each state implies that the L_∞-bound holds as well. Hence, we only need to prove the L_1-bound. By taking the absolute value point-wise in Equation 8 we obtain

$$|V^* - V^{\pi_K}| \leq (\gamma P^*)^K |V^* - V^{\pi_0}| + \sum_{k=0}^{K-1} (\gamma P^*)^{K-k-1} (I - \gamma P^{\pi_{k+1}})^{-1} |\ell_{\pi_k}(\pi_{k+1})| \,.$$

From the fact that $|V^* - V^{\pi_0}| \leq \frac{2}{1-\gamma} R_{\max} \mathbf{1}$, and by integrating both sides w.r.t. μ, and expanding $(I - \gamma P^{\pi^{k+1}})^{-1} = \sum_{t=0}^{\infty} (\gamma P^{\pi_{k+1}})^t$ we have

$$||V^* - V^{\pi_K}||_{1,\mu} \leq \frac{2\gamma^K}{1-\gamma} R_{\max} + \sum_{k=0}^{K-1} \sum_{t=0}^{\infty} \gamma^{K-k-1} \gamma^t \int_{\mathcal{X}} \left[\mu (P^*)^{K-k-1} (P^{\pi^{k+1}})^t\right](dx) |\ell_{\pi_k}(dx; \pi_{k+1})| \,.$$

The integral in the second term corresponds to the expected loss w.r.t. to the distribution over states obtained by starting from μ and then applying $K-k-1$ steps of the optimal policy and t steps of policy π^{k+1}. This term does not correspond to what is actually minimized by DPI at each iteration, and thus, we need to apply Assumption 1 and obtain

$$||V^* - V^{\pi_K}||_{1,\mu} \leq \frac{2\gamma^K}{1-\gamma} R_{\max} + \sum_{k=0}^{K-1} \sum_{t=0}^{\infty} \gamma^{K-k-1} \gamma^t C_{\mu,\rho}(K-k-1, t) \mathcal{L}_{\pi_k}(\rho; \pi_{k+1}) \,.$$

From the definition of $C_{\mu,\rho}$ we obtain

$$||V^* - V^{\pi_K}||_{1,\mu} \leq \frac{2\gamma^K}{1-\gamma} R_{\max} + \frac{C_{\mu,\rho}}{(1-\gamma)^2} \max_{0 \leq k \leq K} \mathcal{L}_{\pi_k}(\rho; \pi_{k+1}) \,.$$

The claim follows from bounding $\mathcal{L}_{\pi_k}(\rho; \pi_{k+1})$ using Theorem 5 with a union bound argument over the K iterations and from the definition of the inherent greedy error. ■

5. Approximation Error

In Section 4.2, we analyzed how the expected error at each iteration k of DPI, $\mathcal{L}_{\pi_k}(\rho; \pi_{k+1})$, propagates through iterations. The final approximation error term in Theorem 7 is the inherent greedy error of Definition 6, $d(\Pi, \mathcal{G}\Pi)$, which depends on the MDP and the richness of the policy space Π. The main question in this section is whether this approximation error can be made small by increasing the capacity of the policy space Π. The answer is not obvious because when the policy space Π grows, on the one hand we can expect it to better approximate any greedy policy w.r.t. a policy in Π, but on the other hand the number of such greedy policies itself grows as well. We start our analysis of this approximation error by introducing the notion of *universal family of policy spaces*.

Definition 8 *A sequence of policy spaces $\{\Pi_n\}$ is a universal family of policy spaces, if there exists a sequence of real numbers $\{\beta_n\}$ with $\lim_{n\to\infty} \beta_n = 0$, such that for any $n > 0$, Π_n is induced by a partition $P_n = \{\mathcal{X}_i\}_{i=1}^{S_n}$ over the state space $\mathcal{X}$ (i.e., for each S_n-tuple $(b_1, \ldots, b_{S_n})$ with $b_i \in \{0,1\}$, there exists a policy $\pi \in \Pi_n$ such that $\pi(x) = b_i$ for all $x \in \mathcal{X}_i$ and for all $i \in \{1, \ldots, S_n\}$) in such a way that*

$$\max_{1 \leq i \leq S_n} \max_{x,y \in \mathcal{X}_i} ||x - y|| \leq \beta_n.$$

This definition requires that for any $n > 0$, Π_n is the space of policies induced by a partition P_n and the diameters of the elements $\mathcal{X}_i$ of this partition shrink to zero as n goes to infinity. The main property of such a sequence of spaces is that any fixed policy π can be approximated arbitrary well by policies of Π_n when $n \to \infty$. Although other definitions of universality could be used, Definition 8 seems natural and it is satisfied by widely-used classifiers such as k-nearest neighbor, uniform grid, and histogram.

In the next section, we first show that the universality of a policy space (Definition 8) does not guarantee that $d(\Pi_n, \mathcal{G}\Pi_n)$ converges to zero in a general MDP. In particular, we present an MDP in which $d(\Pi_n, \mathcal{G}\Pi_n)$ is constant (does not depend on n) even when $\{\Pi_n\}$ is a universal family of classifiers. We then prove that in Lipschitz MDPs, $d(\Pi_n, \mathcal{G}\Pi_n)$ converges to zero for a universal family of policy spaces.

5.1 Counterexample

In this section, we illustrate a simple example in which $d(\Pi_n, \mathcal{G}\Pi_n)$ does not go to zero, even when $\{\Pi_n\}$ is a universal family of classifiers. We consider an MDP with state space $\mathcal{X} = [0,1]$, action space $\mathcal{A} = \{0,1\}$, and the following transitions and rewards

$$x_{t+1} = \begin{cases} \min(x_t + 0.5, 1) & \text{if } a = 1, \\ x_t & \text{otherwise,} \end{cases} \qquad r(x,a) = \begin{cases} 0 & \text{if } x = 1, \\ R_1 & \text{else if } a = 1, \\ R_0 & \text{otherwise,} \end{cases}$$

where $(1-\gamma^2)R_1 < R_0 < R_1$. (9)

We consider the policy space Π_n of piecewise constant policies obtained by uniformly partitioning the state space $\mathcal{X}$ into n intervals. This family of policy spaces is universal. The inherent greedy error of Π_n, $d(\Pi_n, \mathcal{G}\Pi_n)$, can be decomposed into the sum of the expected errors at each interval

$$d(\Pi_n, \mathcal{G}\Pi_n) = \sup_{\pi \in \Pi_n} \inf_{\pi' \in \Pi_n} \sum_{i=1}^{n} \mathcal{L}_\pi^{(i)}(\rho; \pi') \,,$$

where $\mathcal{L}_\pi^{(i)}(\rho; \pi')$ is the same as $\mathcal{L}_\pi(\rho; \pi')$, but with the integral limited to the i-th interval instead of the entire state space $\mathcal{X}$. In the following we show that for the MDP and the universal class of policies considered here, $d(\Pi_n, \mathcal{G}\Pi_n)$ does not converge to zero as n grows.

Let n be odd and $\pi \in \Pi_n$ be one in odd and zero in even intervals (see Figure 2). For any $x > 0.5$, the agent either stays in the same state forever by taking action 0, or goes

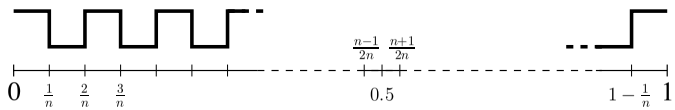


Figure 2: The policy used in the counterexample. It is one in odd and zero in even intervals. Note that the number of intervals, n, is assumed to be odd.

out of bound in one step by taking action 1. Thus, given the assumption of Equation 9, it can be shown that for any x belonging to the intervals $i \geq \frac{n+1}{2}$ (the interval containing 0.5 and above), $(\mathcal{G}\pi)(x) = 0$. This means that there exists a policy $\pi' \in \Pi_n$ such that $\mathcal{L}_\pi^{(i)}(\rho;\pi') = 0$ for all the intervals $i \geq \frac{n+1}{2}$. However, $\mathcal{G}\pi$ does not remain constant in the intervals $i \leq \frac{n-1}{2}$, and changes its value in the middle of the interval. Using Equation 9, we can show that

$$\inf_{\pi' \in \Pi_n} \sum_{i=1}^{n} \mathcal{L}_\pi^{(i)}(\rho;\pi') = C\Big(1 + \frac{1}{1-\gamma}\Big)\frac{n-1}{8n} \geq \frac{C}{16}\Big(1 + \frac{1}{1-\gamma}\Big),$$

where $C = \min\big\{(1-\gamma)(R_1 - R_0), R_0 - (1-\gamma^2)R_1\big\}$. This means that for any odd n, it is always possible to find a policy $\pi \in \Pi_n$ such that $\inf_{\pi' \in \Pi_n} \mathcal{L}_\pi(\rho;\pi')$ is lower bounded by a constant independent of n, and thus, $\lim_{n\to\infty} d(\Pi_n, \mathcal{G}\Pi_n) \neq 0$.

5.2 Lipschitz MDPs

In this section, we prove that for Lipschitz MDPs, $d(\Pi_n, \mathcal{G}\Pi_n)$ goes to zero when $\{\Pi_n\}$ is a universal family of classifiers. We start by defining a Lipschitz MDP.

Definition 9 *A MDP is Lipschitz if both its transition probability and reward functions are Lipschitz, i.e.,* $\forall (B, x, x', a) \in \mathcal{B}(\mathcal{X}) \times \mathcal{X} \times \mathcal{X} \times \mathcal{A}$

$$|r(x,a) - r(x',a)| \leq L_r \|x - x'\|,$$
$$|p(B|x,a) - p(B|x',a)| \leq L_p \|x - x'\|,$$

with L_r and L_p being the Lipschitz constants of the transitions and reward, respectively.

An important property of Lipschitz MDPs is that for any function $Q \in \mathcal{B}^Q(\mathcal{X}\times\mathcal{A}; Q_{\max})$, the function obtained by applying the Bellman operator $\mathcal{T}^\pi$ to $Q(\cdot,a)$, $(\mathcal{T}^\pi Q)(\cdot,a)$, is Lipschitz with constant $L = (L_r + \gamma Q_{\max} L_p)$, for any action $a \in \mathcal{A}$. In fact, for any policy π, any action $a \in \mathcal{A}$ and any pair $x, x' \in \mathcal{X}$ we have

$$\begin{aligned}
&(\mathcal{T}^\pi Q)(x,a) - (\mathcal{T}^\pi Q)(x',a) \\
&\quad = r(x,a) + \gamma \int_{\mathcal{X}} p(dy|x,a) Q^\pi(y,\pi(y)) - r(x',a) - \gamma \int_{\mathcal{X}} p(dy|x',a) Q^\pi(y,\pi(y)) \\
&\quad \leq L_r \|x - x'\| + \gamma \int_{\mathcal{X}} \big|p(dy|x,a) - p(dy|x',a)\big| Q(y,\pi(y)) \\
&\quad \leq (L_r + \gamma L_p Q_{\max}) \|x - x'\|.
\end{aligned}$$

As a result, the function $Q^\pi(\cdot,a)$, which is the unique fixed point of the Bellman operator $\mathcal{T}^\pi$, is Lipschitz with constant L, for any policy $\pi \in \mathcal{B}^\pi(\mathcal{X})$ and any action $a \in \mathcal{A}$.

Theorem 10 *Let $\mathcal{M}$ be a Lipschitz MDP with $|\mathcal{A}| = 2$ and $\{\Pi_n\}$ be a universal family of policy spaces (Definition 8). Then $\lim_{n\to\infty} d(\Pi_n, \mathcal{G}\Pi_n) = 0$.*

Proof

$$\begin{aligned}
d(\Pi_n, \mathcal{G}\Pi_n) &= \sup_{\pi \in \Pi_n} \inf_{\pi' \in \Pi_n} \int_{\mathcal{X}} \ell_\pi(x;\pi') \rho(dx) \\
&\overset{(a)}{=} \sup_{\pi \in \Pi_n} \inf_{\pi' \in \Pi_n} \int_{\mathcal{X}} \mathbb{I}\big\{(\mathcal{G}\pi)(x) \neq \pi'(x)\big\} \Delta^\pi(x) \rho(dx) \\
&\overset{(b)}{=} \sup_{\pi \in \Pi_n} \inf_{\pi' \in \Pi_n} \sum_{i=1}^{S_n} \int_{\mathcal{X}_i} \mathbb{I}\big\{(\mathcal{G}\pi)(x) \neq \pi'(x)\big\} \Delta^\pi(x) \rho(dx) \\
&\overset{(c)}{=} \sup_{\pi \in \Pi_n} \sum_{i=1}^{S_n} \min_{a \in \mathcal{A}} \int_{\mathcal{X}_i} \mathbb{I}\big\{(\mathcal{G}\pi)(x) \neq a\big\} \Delta^\pi(x) \rho(dx) \\
&\overset{(d)}{\leq} \sup_{\pi \in \Pi_n} \sum_{i=1}^{S_n} \min_{a \in \mathcal{A}} \int_{\mathcal{X}_i} \mathbb{I}\big\{(\mathcal{G}\pi)(x) \neq a\big\} 2L \inf_{y:\Delta^\pi(y)=0} \|x - y\| \, \rho(dx) \\
&\overset{(e)}{\leq} 2L \sup_{\pi \in \Pi_n} \sum_{i=1}^{S_n} \min_{a \in \mathcal{A}} \int_{\mathcal{X}_i} \mathbb{I}\big\{(\mathcal{G}\pi)(x) \neq a\big\} \beta_n \rho(dx) \\
&\overset{(f)}{\leq} 2L\beta_n \sum_{i=1}^{S_n} \int_{\mathcal{X}_i} \rho(dx) = 2L\beta_n.
\end{aligned}$$

(a) We rewrite Definition 6, where $\Delta^\pi(x) = \max_{a\in\mathcal{A}} Q^\pi(x,a) - \min_{a' \in \mathcal{A}}(x,a')$ is the regret of choosing the wrong action in state x.
(b) Since Π_n contains piecewise constants policies induced by the partition $P_n = \{\mathcal{X}_i\}$, we split the integral as the sum over the regions.
(c) Since the policies in Π_n can take any action in each possible region, the policy π' minimizing the loss is the one which takes the best action in each region.
(d) Since $\mathcal{M}$ is Lipschitz, both $\max_{a\in\mathcal{A}} Q^\pi(\cdot,a)$ and $\min_{a'\in\mathcal{A}} Q^\pi(\cdot,a')$ are Lipschitz, and thus, $\Delta^\pi(\cdot)$ is $2L$-Lipschitz. Furthermore, Δ^π is zero in all the states in which the policy $\mathcal{G}\pi$ changes (see Figure 3). Thus, for any state x the value $\Delta^\pi(x)$ can be bounded using the Lipschitz property by taking y as the closest state to x in which $\Delta^\pi(y) = 0$.
(e) If $\mathcal{G}\pi$ is constant in a region $\mathcal{X}_i$, the integral can be made zero by setting a to the greedy action (thus making $\mathbb{I}\{(\mathcal{G}\pi)(x) \neq a\} = 0$ for any $x \in \mathcal{X}_i$). Otherwise, if $\mathcal{G}\pi$ changes in a state $y \in \mathcal{X}_i$, then $\Delta^\pi(y) = 0$ and we can replace $\|x-y\|$ by the diameter of the region which is bounded by β_n according to the definition of the universal family of spaces (Definition 8).
(f) We simply take $\mathbb{I}\{(\mathcal{G}\pi)(x) \neq a\} = 1$ in each region.
The claim follows using the definition of the universal family of policy spaces. ■

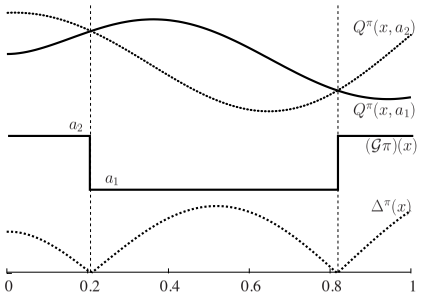


Figure 3: This figure is used as an illustrative example in the proof of Theorem 10. It shows the action-value function of a Lipschitz MDP for a policy π, $Q^\pi(\cdot, a_1)$ and $Q^\pi(\cdot, a_2)$ (top), the corresponding greedy policy $\mathcal{G}\pi$ (middle), and the regret of selecting the wrong action, Δ^π, (bottom).

Theorem 10 together with the counter-example in Section 5.1 show that the assumption on the policy space is not enough to guarantee a small approximation error and additional assumptions on the smoothness of the MDP (e.g., Lipschitz condition) must be satisfied.

5.3 Consistency of DPI

A highly desirable property of any learning algorithm is *consistency*, i.e., as the number of samples grows to infinity, the error of the algorithm converges to zero. It can be seen that as the number of samples N and the rollout horizon H grow in Theorem 5, ϵ_1 and ϵ_2 become arbitrarily small, and thus, the expected error at each iteration, $\mathcal{L}_{\pi_k}(\rho; \pi_{k+1})$, is bounded by the inherent greedy error $d(\Pi_n, \mathcal{G}\Pi_n)$. We can conclude from the results of this section that DPI is not consistent in general, but it is consistent for the class of Lipschitz MDPs, when a universal family of policy spaces is used and n tends to infinite and $d(\Pi_n, \mathcal{G}\Pi_n)$ can be reduced to zero. However, it is important to note that as we increase the index n to reduce the inherent greedy error, the capacity of the policy space Π (its VC-dimension h is indeed a function of n) grows as well, and thus, the error terms ϵ_1 and ϵ_2 may no longer decrease to zero. As a result, to guarantee consistency, we need to link the growth of the policy space Π to the number of samples N, so that as N goes to infinity, the capacity of Π grows at a lower rate and the estimation errors still vanish. More formally, for any number of samples N, we choose an index n so that the corresponding space Π has a VC-dimension $h(N)$ such that $\lim_{N\to\infty} h(N)/N = 0$. We deduce the following result.

Corollary 11 *Let $\mathcal{M}$ be a Lipschitz MDP with $|\mathcal{A}| = 2$, $\{\Pi_n\}$ be a universal family of policy spaces (Definition 8). We define a mapping from the number of samples N to the index n, so that the VC-dimension $h(N)$ is such that $\lim_{N\to\infty} \frac{h(N)}{N} = 0$. Then under either*

Assumption 1 or 2, DPI *is consistent:*

$$\lim_{\substack{N,H,K\to\infty \\ \delta\to 0}} V^{\pi_K} = V^* , \qquad w.p.\ 1.$$

Notice that the result in the previous corollary is possible because the Assumption 1 already covers any policy π in $\mathcal{B}^\pi(\mathcal{X})$ and is not limited to the policies in Π_n.

6. Extension to Multiple Actions

The analysis of Sections 4 and 5 are for the case that the action space contains only two actions. In Section 6.1 we extend the previous theoretical analysis to the general case of an action space with $|\mathcal{A}| > 2$. While the theoretical analysis is completely independent from the specific algorithm used to solve the empirical error minimization problem (see DPI algorithm of Figure 1), in Section 6.2 we discuss which algorithms could be employed to solve this problem in the case of multiple actions.

6.1 Theoretical Analysis

From the theoretical point of view, the extension of the previous results to multiple actions is straightforward. The definitions of loss and error functions do not change and we just need to use an alternative complexity measure for multi-class classification. We rely on the following definitions from (Ben-David et al., 1995).

Definition 12 *Let $\Pi \subseteq \mathcal{B}^\pi(\mathcal{X})$ be a set of deterministic policies and $\Psi = \{\psi : \mathcal{A} \to \{0,1,*\}\}$ be a set of mappings from the action space to the set $\{0,1,*\}$. A finite set of N states $\mathcal{X}_N = \{x_i\}_{i=1}^N \subseteq \mathcal{X}$ is Ψ-shattered by Π if there exists a vector of mappings $\psi^N = \left(\psi^{(1)}, \dots, \psi^{(N)}\right)^\top \in \Psi^N$ such that for any vector $v \in \{0,1\}^N$, there exist a policy $\pi \in \Pi$ such that $\psi^{(i)} \circ \pi(x_i) = v_i, \quad 1 \le i \le N$. The Ψ-dimension of Π is the maximal cardinality of a subset of $\mathcal{X}$, Ψ-shattered by Π.*

Definition 13 *Let $\Pi \subseteq \mathcal{B}^\pi(\mathcal{X})$ be a set of deterministic policies and $\Psi = \{\psi_{k,l} : \mathcal{A} \to \{0,1,*\},\ 1 \le k \ne l \le L\}$ be a set of possible mappings such that*

$$\psi_{k,l}(a) = \begin{cases} 1 & \text{if } a = k, \\ 0 & \text{if } a = l, \\ * & \text{otherwise,} \end{cases}$$

then the Natarajan dimension of Π, N-dim(Π), is the Ψ-dimension of Π.

By using a policy space with finite Natarajan dimension, we derive the following corollary to Theorem 5.

Corollary 14 *Let $\Pi \subseteq \mathcal{B}^\pi(\mathcal{X})$ be a policy space with finite Natarajan dimension $h =$ N-dim(Π) $< \infty$. Let ρ be a distribution over the state space $\mathcal{X}$, N be the number of states in $\mathcal{D}_k$ drawn i.i.d. from ρ, and M be the number of rollouts per state-action pair used by*

DPI *in the estimation of the action-value functions. Let* $\pi_{k+1} = \arg\min_{\pi\in\Pi} \widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi)$ *be the policy computed at the* k*-th iteration of* DPI*. Then, for any* $\delta > 0$*, we have*

$$\mathcal{L}_{\pi_k}(\rho;\pi_{k+1}) \le \inf_{\pi\in\Pi} \mathcal{L}_{\pi_k}(\rho;\pi) + 2(\epsilon_1 + \epsilon_2 + \gamma^H Q_{\max}), \tag{10}$$

with probability $1-\delta$*, where*

$$\epsilon_1 = 16Q_{\max}\sqrt{\frac{2}{N}\left(h\log\frac{|\mathcal{A}|e(N+1)^2}{h} + \log\frac{32}{\delta}\right)}, \quad \epsilon_2 = (1-\gamma^H)Q_{\max}\sqrt{\frac{2}{MN}\log\frac{4|\mathcal{A}|}{\delta}}\ .$$

Proof In order to prove this corollary we just need a minor change in Lemma 3, which now becomes a concentration of measures inequality for a space of multi-class classifiers Π with finite Natarajan dimension. By using similar steps as in the proof of Lemma 3 and by recalling the Sauer's lemma for finite Natarajan dimension spaces (Ben-David et al., 1995), we obtain

$$\mathbb{P}\left[\sup_{\pi\in\Pi}\left|\mathcal{L}_{\pi_k}(\widehat{\rho};\pi) - \mathcal{L}_{\pi_k}(\rho;\pi)\right| > \epsilon\right] \le \delta\ ,$$

with $\epsilon = 16Q_{\max}\sqrt{\frac{2}{N}\left(h\log\frac{|\mathcal{A}|e(N+1)^2}{h} + \log\frac{8}{\delta}\right)}$. The rest of the proof is exactly the same as in Theorem 5. ■

Similarly, the consistency analysis in case of Lipschitz MDPs remains mostly unaffected by the introduction of multiple actions.

Corollary 15 *Let* $\{\Pi_n\}$ *be a universal family of policy spaces (Definition 8), and* $\mathcal{M}$ *be a Lipschitz MDP (Definition 9). Then* $\lim_{n\to\infty} d(\Pi_n, \mathcal{G}\Pi_n) = 0$*.*

Proof The critical part in the proof is the definition of the gap function, which now compares the performance of the greedy action to the performance of the action chosen by the policy π':

$$\Delta^{\pi,\pi'}(x) = \max_{a\in\mathcal{A}} Q^\pi(x,a) - Q^\pi\big(x,\pi'(x)\big).$$

Note that $\Delta^{\pi,\pi'}(\cdot)$ is no longer a Lipschitz function because it is a function of x through the policy π'. However, $\Delta^{\pi,\pi'}(x)$ is Lipschitz in each region $\mathcal{X}_i$, $i = 1\ldots,S_n$, because in each region $\mathcal{X}_i$, by the definition of the policy space, π' is forced to be constant. Therefore, in a region $\mathcal{X}_i$ in which $\pi'(x) = a$, $\forall x \in \mathcal{X}_i$, $\Delta^{\pi,\pi'}(x)$ may be written as

$$\Delta^{\pi,\pi'}(x) = \Delta^{\pi,a}(x) = \max_{a'\in\mathcal{A}} Q^\pi(x,a') - Q^\pi(x,a).$$

The proof here is exactly the same as in Theorem 10 up to step **(c)**, and then we have

$$\begin{aligned}
d(\Pi_n,\mathcal{G}\Pi_n) &= \sup_{\pi\in\Pi_n}\inf_{\pi'\in\Pi_n}\int_{\mathcal{X}} \ell_\pi(x;\pi')\rho(dx)\\
&= \sup_{\pi\in\Pi_n}\inf_{\pi'\in\Pi_n}\int_{\mathcal{X}} \mathbb{I}\big\{(\mathcal{G}\pi)(x)\neq\pi'(x)\big\}\,\Delta^{\pi,\pi'}(x)\rho(dx)\\
&= \sup_{\pi\in\Pi_n}\inf_{\pi'\in\Pi_n}\sum_{i=1}^{S_n}\int_{\mathcal{X}_i} \mathbb{I}\big\{(\mathcal{G}\pi)(x)\neq\pi'(x)\big\}\,\Delta^{\pi,\pi'}(x)\rho(dx)\\
&= \sup_{\pi\in\Pi_n}\sum_{i=1}^{S_n}\min_{a\in\mathcal{A}}\int_{\mathcal{X}_i} \mathbb{I}\{(\mathcal{G}\pi)(x)\neq a\}\,\Delta^{\pi,a}(x)\rho(dx)\\
&\le \sup_{\pi\in\Pi_n}\sum_{i=1}^{S_n}\min_{a\in\mathcal{A}}\int_{\mathcal{X}_i} \Delta^{\pi,a}(x)\rho(dx).
\end{aligned}\tag{11}$$

If the greedy action does not change in a region $\mathcal{X}_i$, i.e., $\forall x\in\mathcal{X}_i$, $(\mathcal{G}\pi)(x) = a'$, for an action $a'\in\mathcal{A}$, then the minimizing policy π' must select action a' in $\mathcal{X}_i$, and thus, the loss will be zero in $\mathcal{X}_i$. Now let assume that the greedy action changes at a state $y\in\mathcal{X}_i$ and the action $b_i \in \arg\max_{a\in\mathcal{A}} Q^\pi(y,a)$. In this case, we have

$$\min_{a\in\mathcal{A}}\int_{\mathcal{X}_i}\Delta^{\pi,a}(x)\rho(dx) \le \int_{\mathcal{X}_i}\Delta^{\pi,b_i}(x)\rho(dx) \le \int_{\mathcal{X}_i}\big(\Delta^{\pi,b_i}(y) + 2L\|x-y\|\big)\rho(dx),$$

since the function $x\mapsto\Delta^{\pi,b_i}(x)$ is $2L$-Lipschitz. Now since $\Delta^{\pi,b_i}(y) = 0$, we deduce from Equation 11 that

$$d(\Pi_n,\mathcal{G}\Pi_n) \le \sup_{\pi\in\Pi_n}\sum_{i=1}^{S_n}\int_{\mathcal{X}_i} 2L||x-y||\rho(dx) \le \sup_{\pi\in\Pi_n}\sum_{i=1}^{S_n}\int_{\mathcal{X}_i} 2L\beta_n\rho(dx) = 2L\beta_n.$$

The claim follows using the definition of the universal family of policy spaces. ■

6.2 Algorithmic Approaches

From an algorithmic point of view, the most critical part of the DPI algorithm (Figure 1) is minimizing the empirical error, which in the case of $|\mathcal{A}| > 2$ is in the following form:

$$\begin{aligned}
\min_{\pi\in\Pi}\widehat{\mathcal{L}}_{\pi_k}(\widehat{\rho};\pi) &= \min_{\pi\in\Pi}\frac{1}{N}\sum_{i=1}^{N}\Big[\max_{a\in\mathcal{A}}\widehat{Q}^{\pi_k}(x_i,a) - \widehat{Q}^{\pi_k}\big(x_i,\pi(x_i)\big)\Big]\\
&= \min_{\pi\in\Pi}\sum_{i=1}^{N}\mathbb{I}\left\{\arg\max_{a\in\mathcal{A}}\widehat{Q}^{\pi_k}(x_i,a)\neq\pi(x_i)\right\}\Big[\max_{a\in\mathcal{A}}\widehat{Q}^{\pi_k}(x_i,a) - \widehat{Q}^{\pi_k}\big(x_i,\pi(x_i)\big)\Big].
\end{aligned}$$

Unlike the two-action case, this is a multi-class cost-sensitive (MCCS) classification problem in which any classification mistake is weighted by a cost function that depends on the action

taken by policy π. It is important to note that here the main difference with regression is that the goal is not to have a good approximation of the action-value function over the entire state and action space. The main objective is to have a good enough estimate of the action-value function to find the greedy action in each state. A thorough discussion on the possible approaches to MCCS classification is out of the scope of this paper, and thus, we only mention a few recent methods that could be suitable for our problem. The reduction methods proposed by Beygelzimer et al. (2005, 2009) reduce the MCCS classification problem to a series of weighted binary classification problems (which in turn can be reduced to binary classification as in Zadrozny et al. 2003), whose solutions can be combined to obtain a multi-class classifier. The resulting multi-class classifier is guaranteed to have a performance which is upper-bounded by the performance of each binary classifier used in solving the weighted binary problems. Another common approach to MCCS classification is to use boosting-based methods (e.g., Lozano and Abe 2008; Busa-Fekete and Kégl 2010). A recent regression-based approach has been proposed by Tu and Lin (2010), which reduces the MCCS classification to a one-sided regression problem that can be effectively solved by a variant of SVM. Finally, a theoretical analysis of the risk bound for MCCS classification is derived by Ávila Pires et al. (2013), while Mineiro (2010) studies error bounds in the case of a reduction from MCCS to regression.

7. Conclusions

In this paper, we presented a variant of the classification-based approach to approximate policy iteration (API) called direct policy iteration (DPI) and provided its finite-sample performance bounds. To the best of our knowledge, this is the first complete finite-sample analysis for this class of API algorithms. The main difference of DPI with the existing classification-based API algorithms (Lagoudakis and Parr, 2003b; Fern et al., 2004) is in weighting each classification error by its actual regret, i.e., the difference between the action-values of the greedy action and the action selected by DPI. Our results extend the only theoretical analysis of a classification-based API algorithm (Fern et al., 2006) by **1)** having a performance bound for the full API algorithm instead of being limited to one step policy update, **2)** considering any policy space instead of finite class of policies, and **3)** deriving a bound which does not depend on the Q-advantage, i.e., the minimum Q-value gap between a greedy and a sub-greedy action over the state space, which can be arbitrarily small in a large class of MDPs. Note that the final bound in (Fern et al., 2006) depends inversely on the Q-advantage. We also analyzed the consistency of DPI and showed that although it is not consistent in general, it is consistent for the class of Lipschitz MDPs. This is similar to the consistency results for fitted value iteration in (Munos and Szepesvári, 2008).

One of the main motivations of this work is to have a better understanding of how the classification-based API methods can be compared with their widely-used regression-based counterparts. It is interesting to note that the bound of Equation 6 shares the same structure as the error bounds for the API algorithm in (Antos et al., 2008) and the fitted value iteration in (Munos and Szepesvári, 2008). The error at each iteration can be decomposed into an approximation error, which depends on the MDP and the richness of the hypothesis space – the inherent greedy error in Equation 6 and the inherent Bellman error in (Antos et al., 2008) and (Munos and Szepesvári, 2008), and an estimation error

which mainly depends on the number of samples and rollouts. The difference between the approximation error of the two approaches depends on how well the hypothesis space fits the MDP at hand. This confirms the intuition that whenever the policies generated by policy iteration are easier to represent and learn than their value functions, a classification-based approach can be preferable to regression-based methods.

Possible directions for future work are:

- *The classification problem:* As discussed in Section 6.2 the main issue in the implementation of DPI is the solution of the multi-class cost-sensitive classification problem at each iteration. Although some existing algorithms might be applied to this problem, further investigation is needed to identify which one is better suited for DPI. In particular, the main challenge is to solve the classification problem without first solving a regression problem on the cost function which would eliminated the main advantage of classification-based approaches (i.e., no approximation of the action-value function over the whole state-action space).

- *Rollout allocation:* In DPI, the rollout set is build with states drawn i.i.d. from an arbitrary distribution and the rollouts are performed the same number of times for each action in $\mathcal{A}$. A significant advantage could be obtained by allocating resources (i.e., the rollouts) to regions of the state space and to actions whose action-values are more difficult to estimate. This would result in a more accurate training set for the classification problem and a better approximation of the greedy policy at each iteration. Although some preliminary results in (Dimitrakakis and Lagoudakis, 2008b) and (Gabillon et al., 2010) show encouraging results, a full analysis of what is the best allocation strategy of rollouts over the state-action space is still missing.

Acknowledgments

This work was supported by the Ministry of Higher Education and Research, the CPER Nord-Pas de Calais/FEDER DATA Advanced data science and technologies 2015-2020, CRIStAL (Centre de Recherche en Informatique et Automatique de Lille), and the French National Research Agency (ANR) under project ExTra-Learn n.ANR-14-CE24-0010-01.

Appendix A. Proofs

Proof [Lemma 3] Although Π is a space of binary policies, standard VC bounds (see e.g.,Vapnik, 1998) cannot be directly employed since the loss function ℓ is not a 0-1 loss function. Let $\mathcal{F}_k$ be the space of the loss functions at iteration k induced by the policies in Π, i.e., $\mathcal{F}_k = \{\ell_{\pi_k}(\cdot;\pi)|\ \pi \in \Pi\}$. We first introduce the notion of the L_1-cover number (Györfi et al., 2002, Section 9.2) of a space $\mathcal{F} = \{f : \mathcal{X} \to [0; B]\}$ of bounded functions on a set of N points $X_1, \dots, X_N$. Given a desired level of accuracy $\epsilon > 0$, $\bar{\mathcal{F}} \subseteq \mathcal{F}$ is an *ϵ-covered* of $\mathcal{F}$ if for any function $f \in \mathcal{F}$ there exists a $\bar{f} \in \bar{\mathcal{F}}$ such that

$$\left|\frac{1}{N}\sum_{i=1}^{N} f(X_i) - \frac{1}{N}\sum_{i=1}^{N} \bar{f}(X_i)\right| \le \epsilon.$$

The number of functions in $\bar{\mathcal{F}}$, denoted by $\mathcal{N}(\mathcal{F}, \epsilon, X_1^N)$, is the cover number of $\mathcal{F}$. Note that all the functions $\ell_{\pi_k}(\cdot;\pi) \in \mathcal{F}_k$ are uniformly bounded by $2Q_{\max}$. As a result, we can apply the Pollard's inequality (Pollard, 1984, Thm. 24) to the bounded space $\mathcal{F}_k$ and obtain

$$\mathbb{P}_{\mathcal{D}_k}\left[\sup_{\ell_{\pi_k}\in\mathcal{F}_k}\left|\frac{1}{N}\sum_{i=1}^{N}\ell_{\pi_k}(x_i) - \int \ell_{\pi_k}(x)\rho(dx)\right| > \epsilon\right] \le 8\mathbb{E}\left[\mathcal{N}_1\left(\frac{\epsilon}{8}, \mathcal{F}_k, X_1^N\right)\right]\exp\left(-\frac{N\epsilon^2}{128(2Q_{\max})^2}\right).$$

Note that at each iteration k, the policy π_k is a random variable because it is the minimizer of the empirical error $\widehat{\mathcal{L}}_{\pi_{k-1}}(\widehat{\rho};\pi)$. However, π_k depends only on the previous policies and rollout sets up to $\mathcal{D}_{k-1}$, and is completely independent of the samples in $\mathcal{D}_k$, thus Pollard's inequality applies conditioned on all the previous iterations. We now show how the covering number of the space $\mathcal{F}_k$ can be directly related to the VC-dimension of Π. Since $\mathcal{A}$ only contains two actions, we can rewrite the loss function as $\ell_{\pi_k}(x;\pi) = \mathbb{I}\{(\mathcal{G}\pi_k)(x) \neq \pi(x)\}\,\Delta^{\pi_k}(x)$, where

$$\Delta^{\pi_k}(x) = \max_{a\in\mathcal{A}} Q^{\pi_k}(x,a) - \min_{a'\in\mathcal{A}} Q^{\pi_k}(x,a')$$

is the gap between the two available actions (i.e., the regret of choosing the wrong action). Let $\bar{\Pi}$ be an $\frac{\epsilon}{2Q_{\max}}$-cover of Π over the states $\{x_i\}_{i=1}^N$ such that for any policy $\pi \in \Pi$ there exists a policy $\bar{\pi} \in \bar{\Pi}$ for which we have

$$\left|\frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\{(\pi(x_i) \neq \bar{\pi}(x_i)\}\right| \le \frac{\epsilon}{2Q_{\max}}.$$

Then $\bar{\mathcal{F}}_k = \{\bar{\ell}_{\pi_k}(\cdot) = \ell_{\pi_k}(\cdot;\bar{\pi})|\bar{\pi} \in \bar{\Pi}\}$ is an ϵ-cover of $\mathcal{F}_k$. In fact for any $\ell_{\pi_k} \in \mathcal{F}_k$, there exist a $\bar{\ell}_{\pi_k} \in \bar{\mathcal{F}}_k$ such that

$$\begin{aligned}
\frac{1}{N}\sum_{i=1}^{N}\left|\ell_{\pi_k}(x_i) - \bar{\ell}_{\pi_k}(x_i)\right| &= \frac{1}{N}\sum_{i=1}^{N}\Big|\mathbb{I}\{(\mathcal{G}\pi_k)(x_i) \neq \pi(x_i)\}\,\Delta^{\pi_k}(x_i) \\
&\quad -\mathbb{I}\{(\mathcal{G}\pi_k)(x_i) \neq \bar{\pi}(x_i)\}\,\Delta^{\pi_k}(x_i)\Big| \\
&\le 2Q_{\max}\frac{1}{N}\sum_{i=1}^{N}\left|\mathbb{I}\{(\mathcal{G}\pi_k)(x_i) \neq \pi(x_i)\} - \mathbb{I}\{(\mathcal{G}\pi_k)(x_i) \neq \bar{\pi}(x_i)\}\right| \\
&= 2Q_{\max}\frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\{(\pi(x_i) \neq \bar{\pi}(x_i)\} \le 2Q_{\max}\frac{\epsilon}{2Q_{\max}} = \epsilon,
\end{aligned}$$

where the last inequality follows from the fact that $\bar{\mathcal{F}}$ is an ϵ-cover of Π. We can now relate the covering number of $\mathcal{F}_k$ to the VC-dimension of Π as

$$\mathcal{N}_1\left(\frac{\epsilon}{8}, \mathcal{F}_k, X_1^N\right) \le \mathcal{N}_1\left(\frac{\epsilon}{16Q_{\max}}, \Pi, X_1^N\right) \le S_\Pi(N) \le \left(\frac{eN}{h}\right)^h,$$

where the first inequality follows from the relationship between the cover numbers of $\mathcal{F}$ and Π, the second inequality bounds the cover number of Π by its growth function $S_\Pi(N)$ (Haussler, 1995), and the last inequality follows from the Sauer's lemma. Since $\mathcal{L}_{\pi_k}(\widehat{\rho};\pi) = \frac{1}{N}\sum_{i=1}^N \ell_{\pi_k}(x_i;\pi)$ and $\mathcal{L}_{\pi_k}(\rho;\pi) = \int \ell_{\pi_k}(x;\pi)\rho(dx)$, the final statement is obtained by inverting the Pollard's bound. ■

Proof [Lemma 4] Similar to the proof of Lemma 3, we rely on the Pollard's inequality to prove the statement. We first introduce a sequence of random events ω_{ij} such that for any $i = 1, \dots, N$ the event ω_{ij} is independently drawn from a suitable distribution ν_i. As a result, we may rewrite the rollout random variables as $R_j^{\pi_k}\big(x_i, \pi(x_i)\big) = R^{\pi_k}(\omega_{ij};\pi)$ and the statement of the theorem as

$$\mathbb{P}\left[\sup_{\pi\in\Pi}\left|\frac{1}{MN}\sum_{i,j}R^{\pi_k}(\omega_{ij};\pi) - \frac{1}{MN}\sum_{i,j}\mathbb{E}_{\nu_i}\left[R^{\pi_k}(\omega_{ij};\pi)\right]\right| > \epsilon\right] \le \delta.$$

Let $\mathcal{H}_k$ be the space of the rollout functions induced by the policies in Π at iteration k, i.e., $\mathcal{H}_k = \{R^{\pi_k}(\cdot;\pi)|\ \pi \in \Pi\}$. Note that all the functions $R^{\pi_k}(\cdot;\pi) \in \mathcal{H}_k$ are uniformly bounded by $(1-\gamma^H)Q_{\max}$. By Pollard's inequality (Pollard, 1984), for the bounded space $\mathcal{H}_k$, we have[15]

$$\mathbb{P}\left[\sup_{\pi\in\Pi}\left|\frac{1}{MN}\sum_{i,j}R^{\pi_k}(\omega_{ij};\pi) - \frac{1}{MN}\sum_{i,j}\mathbb{E}_{\nu_i}[R^{\pi_k}(\omega_{ij};\pi)]\right| > \epsilon\right] \le 8\mathbb{E}\left[\mathcal{N}_1\left(\frac{\epsilon}{8}, \mathcal{H}_k, \omega_1^{MN}\right)\right]\exp\left(-\frac{MN\epsilon^2}{128(1-\gamma^H)^2Q_{\max}^2}\right).$$

15. Note that since here the samples are independent but not identically distributed, we use a slight variation of the standard Pollard's inequality. We refer the reader to the proof of Pollard's inequality (e.g., Pollard 1984 or Devroye et al. 1996) to see that the standard proof can be easily extended to this case.

We now show how the covering number of the space $\mathcal{H}_k$ is related to the VC-dimension of Π. Let $\bar{\Pi}$ be an $\frac{\epsilon}{2(1-\gamma^H)Q_{\max}}$-cover of Π using the empirical distance defined at the states $\{x_i\}_{i=1}^N$, then $\bar{\mathcal{H}}_k = \{\bar{R}^{\pi_k}(\cdot) = R^{\pi_k}(\cdot;\bar{\pi}) | \bar{\pi} \in \bar{\Pi}\}$ is an ϵ-cover of $\mathcal{H}_k$. In fact for any $R^{\pi_k} \in \mathcal{H}_k$, there exist a $\bar{R}^{\pi_k} \in \bar{\mathcal{H}}_k$ such that

$$\frac{1}{MN}\sum_{i,j}\left|R^{\pi_k}(\omega_{ij}) - \bar{R}^{\pi_k}(\omega_{ij})\right| = \frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}\left|R_j^{\pi_k}\big(x_i,\pi(x_i)\big) - R_j^{\pi_k}\big(x_i,\bar{\pi}(x_i)\big)\right|$$
$$\leq 2(1-\gamma^H)Q_{\max}\frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left\{\pi(x_i)\neq\bar{\pi}(x_i)\right\} \leq 2(1-\gamma^H)Q_{\max}\frac{\epsilon}{2(1-\gamma^H)Q_{\max}} = \epsilon.$$

We can now relate the covering number of $\mathcal{F}_k$ to the VC-dimension of Π

$$\mathcal{N}_1\left(\frac{\epsilon}{8},\mathcal{F}_k,\omega_1^{MN}\right) \leq \mathcal{N}_1\left(\frac{\epsilon}{16(1-\gamma^H)Q_{\max}},\Pi,\omega_1^{MN}\right) \leq S_\Pi(MN) \leq \left(\frac{eMN}{h}\right)^h,$$

where $S_\Pi(N)$ is the growth function of Π and the last inequality follows from the Sauer's lemma. The final statement is obtained by inverting the Pollard's bound. ■

References

A. Antos, Cs. Szepesvári, and R. Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning Journal*, 71:89–129, 2008.

B. Ávila Pires, M. Ghavamzadeh, and Cs. Szepesvári. Cost-sensitive multiclass classification risk bounds. In *Proceedings of the Thirtieth International Conference on Machine Learning*, pages 1391–1399, 2013.

J. Bagnell, S. Kakade, A. Ng, and J. Schneider. Policy search by dynamic programming. In *Proceedings of Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

S. Ben-David, N. Cesa-Bianchi, D. Haussler, and P. M. Long. Characterizations of learnability for classes of {0...n}-valued functions. *Journal of Computer and System Sciences*, 50:74–86, 1995.

A. Beygelzimer, V. Dani, T. Hayes, J. Langford, and B. Zadrozny. Error limiting reductions between classification tasks. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pages 49–56, 2005.

Alina Beygelzimer, John Langford, and Pradeep Ravikumar. Error-correcting tournaments. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory*, pages 247–262, 2009.

S. Bradtke and A. Barto. Linear least-squares algorithms for temporal difference learning. *Journal of Machine Learning*, 22:33–57, 1996.

R. Busa-Fekete and B. Kégl. Fast boosting using adversarial bandits. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pages 49–56, 2010.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, 1996.

C. Dimitrakakis and M. Lagoudakis. Algorithms and bounds for sampling-based approximate policy iteration. In *Recent Advances in Reinforcement Learning (EWRL-2008)*. Springer, 2008a.

C. Dimitrakakis and M. Lagoudakis. Rollout sampling approximate policy iteration. *Machine Learning Journal*, 72(3):157–171, 2008b.

A. M. Farahmand, R. Munos, and Cs. Szepesvári. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems*, 2010.

A. M. Farahmand, D. Precup, A. Barreto, and M. Ghavamzadeh. CAPI: Generalized classification-based approximate policy iteration. In *Proceedings of the Multidisciplinary Conference on Reinforcement Learning and Decision Making*, 2013.

A. Fern, S. Yoon, and R. Givan. Approximate policy iteration with a policy language bias. In *Proceedings of Advances in Neural Information Processing Systems 16*, 2004.

A. Fern, S. Yoon, and R. Givan. Approximate policy iteration with a policy language bias: Solving relational Markov decision processes. *Journal of Artificial Intelligence Research*, 25:85–118, 2006.

V. Gabillon, A. Lazaric, and M. Ghavamzadeh. Rollout allocation strategies for classification-based policy iteration. In *ICML Workshop on Reinforcement Learning and Search in Very Large Spaces*, 2010.

V. Gabillon, A. Lazaric, M. Ghavamzadeh, and B. Scherrer. Classification-based policy iteration with a critic. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, pages 1049–1056, 2011.

V. Gabillon, M. Ghavamzadeh, and B. Scherrer. Approximate dynamic programming finally performs well in the game of Tetris. In *Proceedings of Advances in Neural Information Processing Systems 26*, pages 1754–1762, 2013.

M. Ghavamzadeh and A. Lazaric. Conservative and greedy approaches to classification-based policy iteration. In *Proceedings of the Twenty-Sixth Conference on Artificial Intelligence*, pages 914–920, 2012.

L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A distribution-free theory of nonparametric regression*. Springer, New York, Berlin, Paris, 2002.

D. Haussler. Sphere packing numbers for subsets of the boolean n-cube with bounded Vapnik-Chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 69(2):217–232, 1995.

R. Howard. *Dynamic Programming and Markov Processes*. The MIT Press, Cambridge, MA, 1960.

S. Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, Gatsby Computational Neuroscience Unit., University College London, 2003.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 267–274, 2002.

M. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003a.

M. Lagoudakis and R. Parr. Reinforcement learning as classification: Leveraging modern classifiers. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 424–431, 2003b.

J. Langford and B. Zadrozny. Relating reinforcement learning performance to classification performance. In *Proceedings of the Twenty-Second international conference on Machine learning*, pages 473–480, 2005.

A. Lazaric, M. Ghavamzadeh, and R. Munos. Analysis of a classification-based policy iteration algorithm. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pages 607–614, 2010.

A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13:3041–3074, 2012.

L. Li, V. Bulitko, and R. Greiner. Focus of attention in reinforcement learning. *Journal of Universal Computer Science*, 13(9):1246–1269, 2007.

A. Lozano and N. Abe. Multi-class cost-sensitive boosting with p-norm loss functions. In *Proceeding of the Fourteenth ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 506–514, 2008.

P. Mineiro. Error and regret bounds for cost-sensitive multi-class classification reduction to regression, 2010. URL `http://www.machinedlearnings.com/2010/08/error-and-regret-bounds-for-cost.html`.

R. Munos. Performance bounds in Lp norm for approximate value iteration. *SIAM Journal of Control and Optimization*, 2007.

R. Munos and Cs. Szepesvári. Finite time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9:815–857, 2008.

D. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, 1984.

B. Scherrer, M. Ghavamzadeh, V. Gabillon, and M. Geist. Approximate modified policy iteration. In *Proceedings of the Twenty-Ninth International Conference on Machine Learning*, pages 1207–1214, 2012.

H. Tu and H. Lin. One-sided support vector regression for multiclass cost-sensitive classification. In *Proceedings of the Twenty-Seventh International Conference on Machine learning*, pages 49–56, 2010.

V. Vapnik. *Statistical Learning Theory*. John Wiley, 1998.

B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the Third IEEE International Conference on Data Mining*, page 435, 2003.

Journal of Machine Learning Research 17 (2016) 1-54 Submitted 10/11; Revised 8/15; Published 4/16

Operator-valued Kernels for Learning from Functional Response Data

Hachem Kadri hachem.kadri@lif.univ-mrs.fr
Aix-Marseille Université, LIF (UMR CNRS 7279)
F-13288 Marseille Cedex 9, France

Emmanuel Duflos emmanuel.duflos@ec-lille.fr
Ecole Centrale de Lille, CRIStAL (UMR CNRS 9189)
59650 Villeneuve d'Ascq, France

Philippe Preux philippe.preux@univ-lille3.fr
Université de Lille, CRIStAL (UMR CNRS 9189)
59650 Villeneuve d'Ascq, France

Stéphane Canu scanu@insa-rouen.fr
INSA de Rouen, LITIS (EA 4108)
76801, St Etienne du Rouvray, France

Alain Rakotomamonjy alain.rakotomamonjy@insa-rouen.fr
Université de Rouen, LITIS (EA 4108)
76801, St Etienne du Rouvray, France

Julien Audiffren julien.audiffren@cmla.ens-cachan.fr
ENS Cachan, CMLA (UMR CNRS 8536)
94235 Cachan Cedex, France

Editor: John Shawe-Taylor

Abstract

In this paper[1] we consider the problems of supervised classification and regression in the case where attributes and labels are functions: a data is represented by a set of functions, and the label is also a function. We focus on the use of reproducing kernel Hilbert space theory to learn from such functional data. Basic concepts and properties of kernel-based learning are extended to include the estimation of function-valued functions. In this setting, the representer theorem is restated, a set of rigorously defined infinite-dimensional operator-valued kernels that can be valuably applied when the data are functions is described, and a learning algorithm for nonlinear functional data analysis is introduced. The methodology is illustrated through speech and audio signal processing experiments.

Keywords: nonlinear functional data analysis, operator-valued kernels, function-valued reproducing kernel Hilbert spaces, audio signal processing

1. Introduction

In this paper, we consider the supervised learning problem in a functional setting: each attribute of a data is a function, and the label of each data is also a function. For the sake of simplicity, one may imagine real functions, though the work presented here is much more general; one may also think about those functions as being defined over time, or space, though again, our work is not tied to such assumptions and is much more general. To this end, we extend the traditional scalar-valued attribute setting to a function-valued attribute setting.

This shift from scalars to functions is required by the simple fact that in many applications, attributes are functions: functions may be one dimensional such as economic curves (variation of the price of "actions"), load curve of a server, a sound, etc., or two or higher dimensional (hyperspectral images, etc.). Due to the nature of signal acquisition, one may consider that in the end, a signal is always acquired in a discrete fashion, thus providing a real vector. However, with the resolution getting finer and finer in many sensors, the amount of discrete data is getting huge, and one may reasonably wonder whether a functional point of view may not be better than a vector point of view. Now, if we keep aside the application point of view, the study of functional attributes may simply come as an intellectual question which is interesting for its own sake.

From a mathematical point of view, the shift from scalar attributes to function attributes will come as a generalization from scalar-valued functions to function-valued functions, a.k.a."operators". Reproducing Kernel Hilbert Spaces (RKHS) has become a widespread tool to deal with the problem of learning a function mapping the set $\mathbb{R}^p$ to the set of real numbers $\mathbb{R}$. Here, we have to deal with RKHS of operators, that are functions that map a function, belonging to a certain space of functions, to a function belonging to an other space of functions. This shift in terminology is accompanied with a dramatic shift in concepts, and technical difficulties that have to be properly handled.

This *functional regression problem*, or *functional supervised learning*, is a challenging research problem, from statistics to machine learning. Most previous work has focused on the discrete case: the multiple-response (finite and discrete) function estimation problem. In the machine learning literature, this problem is better known under the name of vector-valued function learning (Micchelli and Pontil, 2005a), while in the field of statistics, researchers prefer to use the term multiple output regression (Breiman and Friedman, 1997). One possible solution is to approach the problem from a univariate point of view, that is, assuming only a single response variable output from the same set of explanatory variables. However it would be more efficient to take advantage of correlation between the response variables by considering all responses simultaneously. For further discussion of this point, we refer the reader to Hastie et al. (2001) and references therein. More recently, relevant works in this context concern regularized regression with a sequence of ordered response variables. Many variable selection and shrinkage methods for single response regression are extended to the multiple response data case and several algorithms following the corresponding solution paths are proposed (Turlach et al., 2005; Simila and Tikka, 2007; Hesterberg et al., 2008).

Learning from multiple responses is closely related to the problem of multi-task learning where the goal is to improve generalization performance by learning multiple tasks simultaneously. There is a large literature on this subject, in particular Evgeniou and Pontil (2004); Jebara (2004); Ando and Zhang (2005); Maurer (2006); Ben-david and Schuller-Borbely (2008); Argyriou et al. (2008) and references therein. One paper that has come to our attention is that of Evgeniou et al. (2005) who showed how Hilbert spaces of vector-valued

1. This is a combined and expanded version of previous conference papers (Kadri et al., 2010, 2011c).

functions (Micchelli and Pontil, 2005a) and matrix-valued reproducing kernels (Micchelli and Pontil, 2005b; Reisert and Burkhardt, 2007) can be used as a theoretical framework to develop nonlinear multi-task learning methods.

A primary motivation for this paper is to build on these previous studies and provide a similar framework for addressing the general case where the output space is infinite dimensional. In this setting, the output space is a space of functions and elements of this space are called functional response data. Functional responses are frequently encountered in the analysis of time-varying data when repeated measurements of a continuous response variable are taken over a small period of time (Faraway, 1997; Yao et al., 2005). The relationships among the response data are difficult to explore when the number of responses is large, and hence one might be inclined to think that it could be helpful and more natural to consider the response as a smooth real function. Moreover, with the rapid development of accurate and sensitive instruments and thanks to the currently available large storage resources, data are now often collected in the form of curves or images. The statistical framework underlying the analysis of these data as a single function observation rather than a collection of individual observations is called functional data analysis (FDA) and was first introduced by Ramsay and Dalzell (1991).

It should be pointed out that in earlier studies a similar but less explicit statement of the functional approach was addressed in Dauxois et al. (1982), while the first discussion of what is meant by "functional data" appears to be by Ramsay (1982). Functional data analysis deals with the statistical description and modeling of random functions. For a wide range of statistical tools, ranging from exploratory and descriptive data analysis to linear models and multivariate techniques, a functional version has been recently developed. Reviews of theoretical concepts and prospective applications of functional data can be found in the two monographs by Ramsay and Silverman (2005, 2002). One of the most crucial questions related to this field is "What is the correct way to handle large data? Multivariate or Functional?" Answering this question requires better understanding of complex data structures and relationship among variables. Until now, arguments for and against the use of a functional data approach have been based on methodological considerations or experimental investigations (Ferraty and Vieu, 2003; Rice, 2004). However, we believe that without further improvements in theoretical issues and in algorithm design of functional approaches, exhaustive comparative studies will remain hard to conduct.

This motivates the general framework we develop in this paper. To the best of our knowledge, nonlinear methods for functional data is a topic that has not been sufficiently addressed in the FDA literature. Unlike previous studies on nonlinear supervised classification or real response regression of functional data (Rossi and Villa, 2006; Ferraty and Vieu, 2004; Preda, 2007), this paper addresses the problem of learning tasks where the output variables are functions. From a machine learning point of view, the problem can be viewed as that of learning a function-valued function $f : \mathcal{X} \longrightarrow \mathcal{Y}$ where $\mathcal{X}$ is the input space and $\mathcal{Y}$ the (possibly infinite-dimensional) Hilbert space of the functional output data. Various situations can be distinguished according to the nature of input data attributes (scalars or/and functions). We focus in this work on the case where input attributes are functions, too, but it should be noted that the framework developed here can also be applied when the input data are either discrete, or continuous. Lots of practical applications involve a blend of both functional and non functional attributes, but we do not mix non functional

attributes with functional attributes in this paper. This point has been discussed in (Kadri et al., 2011b). To deal with non-linearity, we adopt a kernel-based approach and we design operator-valued kernels that perform the mapping between the two spaces of functions. Our main results demonstrate how basic concepts and properties of kernel-based learning known in the case of multivariate data can be restated for functional data.

Extending learning methods from multivariate to functional response data may lead to further progress in several practical problems of machine learning and applied statistics. To compare the proposed nonlinear functional approach with other multivariate or functional methods and to apply it in a real world setting, we are interested in the problems of speech inversion and sound recognition, which have attracted increasing attention in the speech processing community in the recent years (Mitra et al., 2010; Rabaoui et al., 2008). These problems can be cast as a supervised learning problem which include some components (predictors or responses) that may be viewed as random curves. In this context, though some concepts on the use of RKHS for functional data similar to those presented in this work can be found in Lian (2007), the present paper provides a much more complete view of learning from functional data using kernel methods, with extended theoretical analysis and several additional experimental results.

This paper is a combined and expanded version of our previous conference papers (Kadri et al., 2010, 2011c). It gives the full justification, additional insights as well as new and comprehensive experiments that strengthen the results of these preliminary conference papers. The outline of the paper is as follows. In Section 2, we discuss the connection between the two fields Functional Data Analysis and Machine Learning, and outline our main contributions. Section 3 defines the notation used throughout the paper. Section 4, describes the theory of reproducing kernel Hilbert spaces of function-valued functions and shows how vector-valued RKHS concepts can be extended to infinite-dimensional output spaces. In Section 5, we exhibit a class of operator-valued kernels that perform the mapping between two spaces of functions and discuss some ideas for understanding their associated feature maps. In Section 6, we provide a function-valued function estimation procedure based on inverting block operator kernel matrices, propose a learning algorithm that can handle functional data, and analyze theoretically its generalization properties. Finally in Section 7, we illustrate the performance of our approach through speech and audio processing experiments.

2. The Interplay of FDA and ML Research

To put our work in context, we begin by discussing the interaction between functional data analysis (FDA) and machine learning (ML). Then, we give an overview of our contributions.

Starting from the fact that "new types of data require new tools for analysis", FDA emerges as a well-defined and suitable concept to further improve classical multivariate statistical methods when data are functions (Levitin et al., 2007). This research field is currently very active, and considerable progress has been made in recent years in designing statistical tools for infinite-dimensional data that can be represented by real-valued functions rather than by discrete, finite dimensional vectors (Ramsay and Silverman, 2005; Ferraty and Vieu, 2006; Shi and Choi, 2011; Horváth and Kokoszka, 2012). While the FDA viewpoint is conventionally adopted in the mathematical statistics community to deal

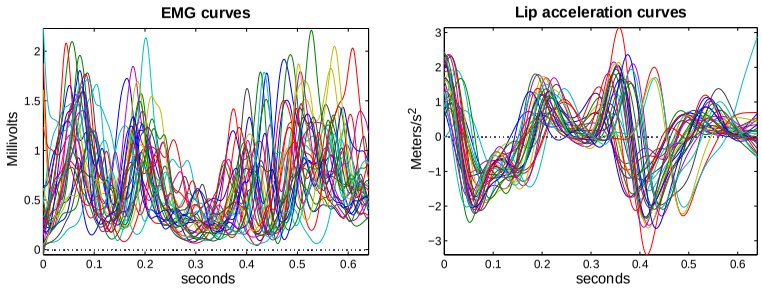


Figure 1: Electromyography (EMG) and lip acceleration curves. The left panel displays EMG recordings from a facial muscle that depresses the lower lip, the depressor labii inferior. The right panel shows the accelerations of the center of the lower lip of a speaker pronouncing the syllable "bob", embedded in the phrase "Say bob again", for 32 replications (Ramsay and Silverman, 2002, Chapter 10).

with data in infinite-dimensional spaces, it does not appear to be commonplace for machine learners. One possible reason for this lack of success is that the formal use of infinite dimensional spaces for practical ML applications may seem unjustified; because in practice traditional measurement devices are limited in providing discrete and not functional data, and a machine learning algorithm can process only finitely represented objects. We believe that for applied machine learners it should be vital to know the full range of applicability of functional data analysis and infinite-dimensional data representations. But due to limitation of space we shall say only few words about the occurrence of functional data in real applications and about the real learning task lying behind this kind of approach. The reader is referred to Ramsay and Silverman (2002) for more details and references. Areas of application discussed and cited there include medical diagnosis, economics, meteorology, biomechanics, and education. For almost all these applications, the high-sampling rate of today's acquisition devices makes it natural to directly handle functions/curves instead of discretized data. Classical multivariate statistical methods may be applied to such data, but they cannot take advantage of the additional information implied by the smoothness of the underlying functions. FDA methods can have beneficial effects in this direction by extracting additional information contained in the functions and their derivatives, not normally available through traditional methods (Levitin et al., 2007).

To get a better idea about the natural occurrence of functional data in ML tasks, Figure 1 depicts a functional data set introduced by Ramsay and Silverman (2002). The data set consists of 32 records of the movement of the center of the lower lip when a subject was repeatedly required to say the syllable "bob", embedded in the sentence, "Say bob again" and the corresponding EMG activities of the primary muscle depressing the lower lip, the depressor labii inferior (DLI)[2]. The goal here is to study the dependence of the acceleration

2. The data set is available at `http://www.stats.ox.ac.uk/~silverma/fdacasebook/lipemg.html`. More information about the data collection process can be found in Ramsay and Silverman (2002, Chapter 10).

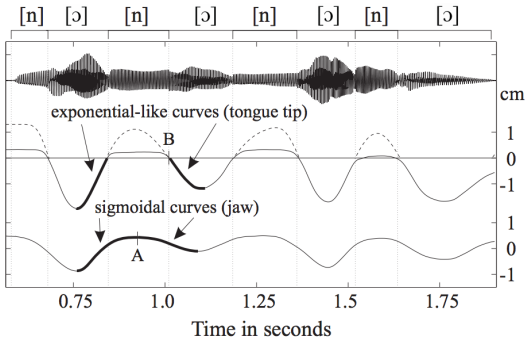


Figure 2: "Audio signal (top), tongue tip trajectory (middle), and jaw trajectory (bottom) for the utterance [ˌnɔnɔˈnɔnɔ]. The trajectories were measured by electromagnetic articulography (EMA) for coils on the tongue tip and the lower incisors. Each trajectory shows the displacement along the first principal component of the original two-dimensional trajectory in the midsagittal plane. The dashed curves show hypothetical continuations of the tongue tip trajectory towards and away from virtual targets during the closure intervals." (Birkholz et al., 2010).

of the lower lip in speech on neural activity. EMG and lip accelerations curves can be well modeled by continuous functions of time that allow to capture functional dependencies and interactions between samples (feature values). Thus, we face a regression problem where both input and output data are functions. In much the same way, Figure 2 also shows a "natural" representation of data in terms of functions [3]. It represents a speech signal used for acoustic-articulatory speech inversion and produced by a subject pronouncing a sequence of [ˌCVCVˈCVCV] (C=consonant, V=vowel) by combining the vowel {/ɔ/} with the consonant {/n/}. The articulatory trajectories are represented by the upper and lower solid curves that show the displacement of fleshpoints on the tongue tip and the jaw along the main movement direction of these points during the repeated opening and closing gestures. This example is from a recent study on the articulatory modeling of speech signals (Birkholz et al., 2010). The concept of articulatory gestures in the context of speech-to-articulatory inversion will be explained in more details in Section 7. As shown in the figure, the observed articulatory trajectories are typically modeled by smooth functions of time with periodicity properties and exponential or sigmoidal shape, and the goal of speech inversion is to predict and recover geometric data of the vocal tract from the speech information.

In both examples given above, response data clearly present a functional behavior that should be taken into account during the learning process. We think that handling these data as what they really are, that is functions, is a promising way to tackle prediction problems and design efficient ML systems for continuous data variables. Moreover, ML methods which

3. This figure is from Birkholz et al. (2010).

can handle functional features can open up plenty of new areas of application, where the flexibility of functional and infinite-dimensional spaces would allow to enable us to achieve significantly better performance while managing huge amounts of training data.

In the light of these observations, there is an interest in overcoming methodological and practical problems that hinder the wide adoption and use of functional methods built for infinite-dimensional data. Regarding the practical issue related to the application and implementation of infinite-dimensional spaces, a standard means of addressing it is to choose a functional space *a priori* with a known predefined set of basis functions in which the data will be mapped. This may include a preprocessing step, which consists in converting the discretized data into functional objects using interpolation or approximation techniques. Following this scheme, parametric FDA methods have emerged as a common approach to extend multivariate statistical analysis in functional and infinite-dimensional situations (Ramsay and Silverman, 2005). More recently, nonparametric FDA methods have received increasing attention because of their ability to avoid fixing a set of basis functions for the functional data beforehand (Ferraty and Vieu, 2006). These methods are based on the concept of semi-metrics for modeling functional data. The reason for using a semi-metric rather than a metric is that the coincidence axiom, namely $d(x_i, x_j) = 0 \Leftrightarrow x_i = x_j$, may result in curves with very similar shapes being categorized as distant (not similar to each other). To define closeness between functions in terms of shape rather than location semi-metrics can be used. In this spirit, Ferraty and Vieu (2006) provided a semi-metric based methodology for nonparametric functional data analysis and argued that this can be a sufficiently general theoretical framework to tackle infinite-dimensional data without being "too heavy" in terms of computational time and implementation complexity.

Thus, although both parametric and nonparametric functional data analyses deal with infinite-dimensional data, they are computationally feasible and quite practical since the observed functional data are approximated in a basis of the function space with possibly finite number of elements. What we really need is the inner or semi-inner product of the basis elements and the representation of the functions with respect to that basis. We think that Machine Learning research can profit from exploring other representation formalisms that support the expressive power of functional data. Machine learning methods which can accommodate functional data should open up new possibilities for handling practical applications for which the flexibility of infinite-dimensional spaces could be exploited to achieve performance benefits and accuracy gains. On the other hand, in the FDA field, there is clearly a need for further development of computationally efficient and understandable algorithms that can deliver near-optimal solutions for infinite-dimensional problems and that can handle a large number of features. The transition from infinite-dimensional statistics to efficient algorithmic design and implementation is of central importance to FDA methods in order to make them more practical and popular. In this sense, Machine Learning can have a profound impact on FDA research.

In reality, ML and FDA have more in common than it might seem. There are already existing machine learning algorithms that can also be viewed as FDA methods. For example, these include kernel methods which use a certain type of similarity measure (called a kernel) to map observed data in a high dimensional feature space, in which linear methods are used for learning problems (Shawe-Taylor and Cristanini, 2004; Schölkopf and Smola,

2002). Depending on the choice of the kernel function, the feature space can be infinite-dimensional. The kernel trick is used, allowing to work with finite Gram matrix of inner products between the possibly infinite-dimensional features which can be seen as functional data. This connection between kernel and FDA methods is clearer with the concept of kernel embedding of probability distributions, where, instead of (observed) single points, kernel means are used to represent probability distributions (Smola et al., 2007; Sriperumbudur et al., 2010). The kernel mean corresponds to a mapping of a probability distribution in a feature space which is rich enough so that its expectation uniquely identifies the distribution. Thus, rather than relying on large collections of vector data, kernel-based learning can be adapted to probability distributions that are constructed to meaningfully represent the discrete data by the use of kernel means (Muandet et al., 2012). In some sense, this represents a similar design to FDA methods, where data are assumed to lie in a functional space even though they are acquired in a discrete manner. There are also other papers that deal with machine learning problems where covariates are probability distributions and discuss their relation with FDA (Poczos et al., 2012, 2013; Oliva et al., 2013). At that point, however, the connection between ML and FDA is admittedly weak and needs to be bolstered by the delivery of more powerful and flexible learning machines that are able to deal with functional data and infinite-dimensional spaces.

In the FDA field, linear models have been explored extensively. Nonlinear modeling of functional data is, however, a topic that has not been sufficiently investigated, especially when response data are functions. Reproducing kernels provide a powerful tool for solving learning problems with nonlinear models, but to date they have been used more to learn scalar-valued or vector-valued functions than function-valued functions. Consequently, kernels for functional response data and their associated function-valued reproducing kernel Hilbert spaces have remained mostly unknown and poorly studied. In this work, we aim to rectify this situation, and highlight areas of overlap between the two fields FDA and ML, particularly with regards to the applicability and relevance of the FDA paradigm coupled with machine learning techniques. Specifically, we provide a learning methodology for nonlinear FDA based on the theory of reproducing kernels. The main contributions are as follows:

- we introduce a set of rigorously defined operator-valued kernels suitable for functional response data, that can be valuably applied to model dependencies between samples and take into account the functional nature of the data, like the smoothness of the curves underlying the discrete observations,
- we propose an efficient algorithm for learning function-valued functions (operators) based on the spectral decomposition of block operator matrices,
- we study the generalization performance of our learned nonlinear FDA model using the notion of algorithmic stability,
- we show the applicability and suitability of our framework to two problems in audio signal processing, namely speech inversion and sound recognition, where features are functions that are dependent on each other.

3. Notations and Conventions

We start by some standard notations and definitions used all along the paper. Given a Hilbert space $\mathcal{H}$, $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and $\| \cdot \|_{\mathcal{H}}$ refer to its inner product and norm, respectively. $H^n = \underbrace{\mathcal{H} \times \ldots \times \mathcal{H}}_{n \text{ times}}$, $n \in \mathbb{N}_+$, denotes the topological product of n spaces $\mathcal{H}$. We denote by $\mathcal{X} = \{x : \Omega_x \longrightarrow \mathbb{R}\}$ and $\mathcal{Y} = \{y : \Omega_y \longrightarrow \mathbb{R}\}$ the separable Hilbert spaces of input and output real-valued functions whose domains are Ω_x and Ω_y, respectively. In functional data analysis domain, the space of functions is generally assumed to be the Hilbert space of equivalence classes of square integrable functions, denoted by L^2. Thus, in the rest of the paper, we consider $\mathcal{Y}$ to be the space $L^2(\Omega_y)$, where Ω_y is a compact set. The vector space of functions from $\mathcal{X}$ into $\mathcal{Y}$ is denoted by $\mathcal{Y}^{\mathcal{X}}$ endowed with the topology of uniform convergence on compact subsets of $\mathcal{X}$. We denote by $\mathcal{C}(\mathcal{X}, \mathcal{Y})$ the vector space of continuous functions from $\mathcal{X}$ to $\mathcal{Y}$, by $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$ the Hilbert space of function-valued functions $F : \mathcal{X} \longrightarrow \mathcal{Y}$, and by $\mathcal{L}(\mathcal{Y})$ the set of bounded linear operators from $\mathcal{Y}$ to $\mathcal{Y}$.

We now fix the following conventions for bounded linear operators and block operator matrices.

Definition 1 *(adjoint, self-adjoint, and positive operators)*
Let $A \in \mathcal{L}(\mathcal{Y})$. Then:

(i) *A^*, the adjoint operator of A, is the unique operator in $\mathcal{L}(\mathcal{Y})$ that satisfies*

$$\langle Ay, z \rangle_{\mathcal{Y}} = \langle y, A^* z \rangle_{\mathcal{Y}} \ , \ \forall y \in \mathcal{Y}, \forall z \in \mathcal{Y},$$

(ii) *A is self-adjoint if $A = A^*$,*

(iii) *A is positive if it is self-adjoint and $\forall y \in \mathcal{Y}$, $\langle Ay, y \rangle_{\mathcal{Y}} \geq 0$ (we write $A \geq 0$),*

(iv) *A is larger or equal than $B \in \mathcal{L}(\mathcal{Y})$, if $A - B$ is positive, i.e., $\forall y \in \mathcal{Y}$, $\langle Ay, y \rangle_{\mathcal{Y}} \geq \langle By, y \rangle_{\mathcal{Y}}$ (we write $A \geq B$).*

Definition 2 *(block operator matrix)*
Let $n \in \mathbb{N}$, let $\mathcal{Y}^n = \underbrace{\mathcal{Y} \times \ldots \times \mathcal{Y}}_{n \text{ times}}$.

(i) *$\mathbf{A} \in \mathcal{L}(\mathcal{Y}^n)$, given by*

$$\mathbf{A} = \begin{pmatrix} A_{11} & \ldots & A_{1n} \\ \vdots & & \vdots \\ A_{n1} & \ldots & A_{nn} \end{pmatrix}$$

where each $A_{ij} \in \mathcal{L}(\mathcal{Y})$, $i, j = 1, \ldots, n$, is called a block operator matrix,

(ii) *the adjoint (or transpose) of $\mathbf{A}$ is the block operator matrix $\mathbf{A}^* \in \mathcal{L}(\mathcal{Y}^n)$ such that $(A^*)_{ij} = (A_{ji})^*$,*

(iii) *self-adjoint and order relations of block operator matrices are defined in the same way as for bounded operators (see Definition 1).*

real numbers	α, β, γ, ...	Greek characters
integers	i, j, m, n	
vector spaces[4]	$\mathcal{X}$, $\mathcal{Y}$, $\mathcal{H}$, ...	Calligraphic letters
subsets of the real plain	Ω, Λ, Γ, ...	capital Greek characters
functions[5](or vectors)	x, y, f, ...	small Latin characters
vector of functions	$\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$, ...	small bold Latin characters
operators (or matrices)	A, B, K, ...	capital Latin characters
block operator matrices	$\mathbf{A}$, $\mathbf{B}$, $\mathbf{K}$, ...	capital bold Latin characters
adjoint operator	$*$	A^* adjoint of operator A
identical equality	$\equiv$	equality of mappings
definition	$\triangleq$	equality by definition

Table 1: Notations used in this paper.

Note that item (ii) in Definition 2 is obtained from the definition of adjoint operator. It is easy to see that $\forall \mathbf{y} \in \mathcal{Y}^n$ and $\forall \mathbf{z} \in \mathcal{Y}^n$; we have: $\langle \mathbf{Ay}, \mathbf{z} \rangle_{\mathcal{Y}^n} = \sum_{i,j} \langle A_{ij} y_j, z_i \rangle_{\mathcal{Y}} = \sum_{i,j} \langle y_j, A_{ij}^* z_i \rangle_{\mathcal{Y}} = \sum_{i,j} \langle y_j, (A^*)_{ji} z_i \rangle_{\mathcal{Y}} = \langle \mathbf{y}, \mathbf{A}^* \mathbf{z} \rangle_{\mathcal{Y}^n}$, where $(A^*)_{ji} = (A_{ij})^*$.

To help the reader, notations frequently used in the paper are summarized in Table 1.

4. Reproducing Kernel Hilbert Spaces of Function-valued Functions

Hilbert spaces of scalar-valued functions with reproducing kernels were introduced and studied in Aronszajn (1950). Due to their crucial role in designing kernel-based learning methods, these spaces have received considerable attention over the last two decades (Shawe-Taylor and Cristanini, 2004; Schölkopf and Smola, 2002). More recently, interest has grown in exploring reproducing Hilbert spaces of vector functions for learning vector-valued functions (Micchelli and Pontil, 2005a; Carmeli et al., 2006; Caponnetto et al., 2008; Carmeli et al., 2010; Zhang et al., 2012), even though the idea of extending the theory of Reproducing Kernel Hilbert Spaces from the scalar-valued case to the vector-valued one is not new and dates back to at least Schwartz (1964). For more details, see the review paper by Álvarez et al. (2012).

In the field of machine learning, Evgeniou et al. (2005) have shown how Hilbert spaces of vector-valued functions and matrix-valued reproducing kernels can be used in the context of multi-task learning, with the goal of learning many related regression or classification tasks simultaneously. Since this seminal work, it has been demonstrated that these kernels and their associated spaces are capable of solving various other learning problems such as multiple output learning (Baldassarre et al., 2012), manifold regularization (Minh and Sindhwani, 2011), structured output prediction (Brouard et al., 2011; Kadri et al., 2013a), multi-view learning (Minh et al., 2013; Kadri et al., 2013b) and network inference (Lim et al., 2013, 2015).

4. We also use the standard notations such as $\mathbb{R}^n$ and L^2.
5. We denote by small Latin characters scalar-valued functions. Operator-valued functions (or kernels) are denoted by capital Latin characters $A(\cdot, \cdot)$, $B(\cdot, \cdot)$, $K(\cdot, \cdot)$, ...

In contrast to most of these previous works, here we are interested in the general case where the output space is a space of vectors with infinite dimension. This may be valuable from a variety of perspectives. Our main motivation is the supervised learning problem when output data are functions that could represent, for example, one-dimensional curves (this was mentioned as future work in Szedmak et al. 2006). One of the simplest ways to handle these data is to treat them as multivariate vectors. However this method does not consider any dependency of different values over subsequent time-points within the same functional datum and suffers when data dimension is very large. Therefore, we adopt a functional data analysis viewpoint (Zhao et al., 2004; Ramsay and Silverman, 2005; Ferraty and Vieu, 2006) in which multiple curves are viewed as functional realizations of a single function. It is important to note that matrix-valued kernels for infinite-dimensional output spaces, commonly known as operator-valued kernels, have been considered in previous studies (Micchelli and Pontil, 2005a; Caponnetto et al., 2008; Carmeli et al., 2006, 2010); however, they have been only studied in a theoretical perspective. Clearly, further investigations are needed to illustrate the practical benefits of the use of operator-valued kernels, which is the main focus of this work.

We now describe how RKHS theory can be extended from real or vector to functional response data. In particular, we focus on reproducing kernel Hilbert spaces whose elements are function-valued functions (or operators) and we demonstrate how basic properties of real-valued RKHS can be restated in the functional case, if appropriate conditions are satisfied. Extension to the functional case is not so obvious and requires tools from functional analysis (Rudin, 1991). Spaces of operators whose range is infinite-dimensional can exhibit unusual behavior, and standard topological properties may not be preserved in the infinite-dimensional case because of functional analysis subtleties. So, additional restrictions imposed on these spaces are needed for extending the theory of RKHS towards infinite-dimensional output spaces. Following Carmeli et al. (2010), we mainly focus on separable Hilbert spaces with reproducing operator-valued kernels whose elements are continuous functions. This is a sufficient condition to avoid topological and measurability problems encountered with this extension. For more details about vector or function-valued RKHS of measurable and continuous functions, see Carmeli et al. (2006, Sections 3 and 5). Note that the framework developed in this section should be valid for any type of input data (vectors, functions, or structures). In this paper, however, we consider the case where both input and output data are functions.

Definition 3 *(Operator-valued kernel)*
An $\mathcal{L}(\mathcal{Y})$-valued kernel K on $\mathcal{X}^2$ is a function $K(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$;

(i) *K is Hermitian if $\forall x, z \in \mathcal{X}$, $K(w,z) = K(z,w)^*$, (where the superscript * denotes the adjoint operator),*

(ii) *K is nonnegative on $\mathcal{X}$ if it is Hermitian and for every natural number r and all $\{(w_i, u_i)_{i=1,\dots,r}\} \in \mathcal{X} \times \mathcal{Y}$, the matrix with ij-th entry $\langle K(w_i, w_j)u_i, u_j\rangle_{\mathcal{Y}}$ is nonnegative (positive-definite).*

Definition 4 *(Block operator kernel matrix)*
Given a set $\{w_i\} \in \mathcal{X}$, $i = 1, \dots, n$ with $n \in \mathbb{N}_+$, and an operator-valued kernel K, the corresponding block operator kernel matrix is the matrix $\mathbf{K} \in \mathcal{L}(\mathcal{Y}^n)$ with entries

$$\mathbf{K}_{ij} = K(w_i, w_j).$$

The block operator kernel matrix is simply the kernel matrix associated to an operator-valued kernel. Since the kernel outputs an operator, the kernel matrix is in this case a block matrix where each block is an operator in $\mathcal{L}(\mathcal{Y})$. It is easy to see that an operator-valued kernel K is nonnegative if and only if the associated block operator kernel matrix $\mathbf{K}$ is positive.

Definition 5 *(Function-valued RKHS)*
A Hilbert space $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathcal{Y}$ is called a reproducing kernel Hilbert space if there is a nonnegative $\mathcal{L}(\mathcal{Y})$-valued kernel K on $\mathcal{X}^2$ such that:

(i) *the function $z \longmapsto K(w,z)g$ belongs to $\mathcal{F}, \forall z, w \in \mathcal{X}$ and $g \in \mathcal{Y}$,*

(ii) *for every $F \in \mathcal{F}, w \in \mathcal{X}$ and $g \in \mathcal{Y}$, $\langle F, K(w,\cdot)g\rangle_{\mathcal{F}} = \langle F(w), g\rangle_{\mathcal{Y}}$.*

On account of (ii), the kernel is called the reproducing kernel of $\mathcal{F}$. In Carmeli et al. (2006, Section 5), the authors provided a characterization of RKHS with operator-valued kernels whose functions are continuous and proved that $\mathcal{F}$ is a subspace of $\mathcal{C}(\mathcal{X}, \mathcal{Y})$, the vector space of continuous functions from $\mathcal{X}$ to $\mathcal{Y}$, if and only if the reproducing kernel K is locally bounded and separately continuous. Such a kernel is qualified as Mercer (Carmeli et al., 2010). In the following, we will only consider separable RKHS $\mathcal{F} \subset \mathcal{C}(\mathcal{X}, \mathcal{Y})$.

Theorem 1 *(Uniqueness of the reproducing operator-valued kernel)*
If a Hilbert space $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathcal{Y}$ admits a reproducing kernel, then the reproducing kernel K is uniquely determined by the Hilbert space $\mathcal{F}$.

Proof: Let K be a reproducing kernel of $\mathcal{F}$. Suppose that there exists another reproducing kernel K' of $\mathcal{F}$. Then, for all $\{w, w'\} \in \mathcal{X}$ and $\{h, g\} \in \mathcal{Y}$, applying the reproducing property for K and K' we get

$$\langle K'(w',\cdot)h, K(w,\cdot)g\rangle_{\mathcal{F}} = \langle K'(w',w)h, g\rangle_{\mathcal{Y}}, \tag{1}$$

we have also

$$\begin{aligned} \langle K'(w',\cdot)h, K(w,\cdot)g\rangle_{\mathcal{F}} &= \langle K(w,\cdot)g, K'(w',\cdot)h\rangle_{\mathcal{F}} &&= \langle K(w,w')g, h\rangle_{\mathcal{Y}} \\ &= \langle g, K(w,w')^*h\rangle_{\mathcal{Y}} &&= \langle g, K(w',w)h\rangle_{\mathcal{Y}}. \end{aligned} \tag{2}$$

(1) and (2) $\Rightarrow K(w,w') \equiv K'(w,w'), \forall w, w' \in \mathcal{X}$. ■

A key point for learning with kernels is the ability to express functions in terms of a kernel providing the way to evaluate a function at a given point. This is possible because there exists a bijection relationship between a large class of kernels and associated reproducing kernel spaces which satisfy a regularity property. Bijection between scalar-valued kernels and RKHS was first established by Aronszajn (1950, Part I, Sections 3 and 4). Then Schwartz (1964, Chapter 5) shows that this is a particular case of a more general situation. This bijection in the case where input and output data are continuous and belong to the infinite-dimensional functional spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively, is still valid and is given by the following theorem (see also Theorem 4 of Senkene and Tempel'man, 1973).

Theorem 2 *(Bijection between function-valued RKHS and operator-valued kernel)*
A $\mathcal{L}(\mathcal{Y})$-valued Mercer kernel K on $\mathcal{X}^2$ is the reproducing kernel of some Hilbert space $\mathcal{F}$, if and only if it is nonnegative.

We give a proof of this theorem by extending the scalar-valued case $\mathcal{Y} = \mathbb{R}$ in Aronszajn (1950) to the domain of functional data analysis domain where $\mathcal{Y}$ is $L^2(\Omega_y)$.[6] The proof is performed in two steps. The necessity is an immediate result from the reproducing property. For the sufficiency, the outline of the proof is as follows: we assume $\mathcal{F}_0$ to be the space of all $\mathcal{Y}$-valued functions F of the form $F(\cdot) = \sum_{i=1}^{n} K(w_i, \cdot)u_i$, where $w_i \in \mathcal{X}$ and $u_i \in \mathcal{Y}$, with the following inner product $\langle F(\cdot), G(\cdot)\rangle_{\mathcal{F}_0} = \sum_{i=1}^{n}\sum_{j=1}^{m}\langle K(w_i, z_j)u_i, v_j\rangle_{\mathcal{Y}}$ defined for any $G(\cdot) = \sum_{j=1}^{m} K(z_j, \cdot)v_j$ with $z_j \in \mathcal{X}$ and $v_j \in \mathcal{Y}$. We show that $(\mathcal{F}_0, \langle\cdot,\cdot\rangle_{\mathcal{F}_0})$ is a pre-Hilbert space. Then we complete this pre-Hilbert space via Cauchy sequences $\{F_n(\cdot)\} \subset \mathcal{F}_0$ to construct the Hilbert space $\mathcal{F}$ of $\mathcal{Y}$-valued functions. Finally, we conclude that $\mathcal{F}$ is a reproducing kernel Hilbert space, since $\mathcal{F}$ is a real inner product space that is complete under the norm $\|\cdot\|_{\mathcal{F}}$ defined by $\|F(\cdot)\|_{\mathcal{F}} = \lim_{n\to\infty} \|F_n(\cdot)\|_{\mathcal{F}_0}$, and has $K(\cdot,\cdot)$ as reproducing kernel.

Proof: *Necessity.* Let K be the reproducing kernel of a Hilbert space $\mathcal{F}$. Using the reproducing property of the kernel K we obtain for any $\{w_i, w_j\} \in \mathcal{X}$ and $\{u_i, u_j\} \in \mathcal{Y}$

$$\sum_{i,j=1}^{n}\langle K(w_i, w_j)u_i, u_j\rangle_Y = \sum_{i,j=1}^{n}\langle K(w_i,\cdot)u_i, K(w_j,\cdot)u_j\rangle_{\mathcal{F}}$$
$$= \langle\sum_{i=1}^{n} K(w_i,\cdot)u_i, \sum_{i=1}^{n} K(w_i,\cdot)u_i\rangle_{\mathcal{F}} = \|\sum_{i=1}^{n} K(w_i,\cdot)u_i\|_{\mathcal{F}}^2 \;\geq\; 0.$$

Sufficiency. Let $\mathcal{F}_0 \subset \mathcal{Y}^{\mathcal{X}}$ be the space of all $\mathcal{Y}$-valued functions F of the form $F(\cdot) = \sum_{i=1}^{n} K(w_i,\cdot)u_i$, where $w_i \in \mathcal{X}$ and $u_i \in \mathcal{Y}$, $i = 1, \ldots, n$. We define the inner product of the functions $F(\cdot) = \sum_{i=1}^{n} K(w_i,\cdot)u_i$ and $G(\cdot) = \sum_{j=1}^{m} K(z_j,\cdot)v_j$ from $\mathcal{F}_0$ as follows

$$\langle F(\cdot), G(\cdot)\rangle_{\mathcal{F}_0} = \langle\sum_{i=1}^{n} K(w_i,\cdot)u_i, \sum_{j=1}^{m} K(z_j,\cdot)v_j\rangle_{\mathcal{F}_0} = \sum_{i=1}^{n}\sum_{j=1}^{m}\langle K(w_i, z_j)u_i, v_j\rangle_{\mathcal{Y}}.$$

$\langle F(\cdot), G(\cdot)\rangle_{\mathcal{F}_0}$ is a symmetric bilinear form on $\mathcal{F}_0$ and due to the positivity of the kernel K, $\|F(\cdot)\|$ defined by

$$\|F(\cdot)\| = \sqrt{\langle F(\cdot), F(\cdot)\rangle_{\mathcal{F}_0}}$$

is a quasi-norm in $\mathcal{F}_0$. The reproducing property in $\mathcal{F}_0$ is verified with the kernel K. In fact, if $F \in \mathcal{F}_0$ then

$$F(\cdot) = \sum_{i=1}^{n} K(w_i,\cdot)u_i,$$

6. The proof should be applicable to arbitrarily separable output Hilbert spaces $\mathcal{Y}$.

and $\forall\ (w, u) \in \mathcal{X} \times \mathcal{Y}$,

$$\langle F, K(w,\cdot)u\rangle_{\mathcal{F}_0} = \langle\sum_{i=1}^{n} K(w_i,\cdot)u_i, K(w,\cdot)u\rangle_{\mathcal{F}_0} = \langle\sum_{i=1}^{n} K(w_i, w)u_i, u\rangle_{\mathcal{Y}} = \langle F(w), u\rangle_{\mathcal{Y}}.$$

Moreover using the Cauchy-Schwartz inequality, we have: $\forall\ (w, u) \in \mathcal{X} \times \mathcal{Y}$,

$$\langle F(w), u\rangle_{\mathcal{Y}} = \langle F(\cdot), K(w,\cdot)u\rangle_{\mathcal{F}_0} \leq \|F(\cdot)\|_{\mathcal{F}_0}\|K(w,\cdot)u\|_{\mathcal{F}_0}.$$

Thus, if $\|F\|_{\mathcal{F}_0} = 0$, then $\langle F(w), u\rangle_{\mathcal{Y}} = 0$ for any w and u, and hence $F \equiv 0$. Thus $(\mathcal{F}_0, \langle.,.\rangle_{\mathcal{F}_0})$ is a pre-Hilbert space. This pre-Hilbert space is in general not complete, but it can be completed via Cauchy sequences to build the $\mathcal{Y}$-valued Hilbert space $\mathcal{F}$ which has K as reproducing kernel, which concludes the proof. The completion of $\mathcal{F}_0$ is given in Appendix A (we refer the reader to the monograph by Rudin, 1991, for more details about completeness and the general theory of topological vector spaces). ■

We now give an example of a function-valued RKHS and its operator-valued kernel. This example serves to illustrate how these spaces and their associated kernels generalize the standard scalar-valued case or the vector-valued one to functional and infinite-dimensional output data. Thus, we first report an example of a scalar-valued RKHS and the corresponding scalar-valued kernel. We then extend this example to the case of vector-valued Hilbert spaces with matrix-valued kernels, and finally to function-valued RKHS where the output space is infinite dimensional. For the sake of simplicity, the input space $\mathcal{X}$ in these examples is assumed to be a subset of $\mathbb{R}$.

Example 1 *(Scalar-valued RKHS and its scalar-valued kernel; see Canu et al. (2003))*
Let $\mathcal{F}$ be the space defined as follows:

$$\begin{cases} \mathcal{F} = \big\{f : [0,1] \longrightarrow \mathbb{R} \text{ absolutely continuous},\ \exists f' \in L^2([0,1]), f(x) = \displaystyle\int_0^x f'(z)dz\big\}, \\ \langle f_1, f_2\rangle_{\mathcal{H}} = \langle f_1', f_2'\rangle_{L^2([0,1])}. \end{cases}$$

$\mathcal{F}$ is the Sobolev space of degree 1, also called the Cameron-Martin space, and is a scalar-valued RKHS of functions $f : [0,1] \longrightarrow \mathbb{R}$ with the scalar-valued reproducing kernel $k(x,z) = min(x,z)$, $\forall x, z \in \mathcal{X} = [0,1]$.

Example 2 *(Vector-valued RKHS and its matrix-valued kernel)*
Let $\mathcal{X} = [0,1]$ and $\mathcal{Y} = \mathbb{R}^n$. Consider the matrix-valued kernel K defined by:

$$K(x,z) = \begin{cases} diag(x) & if \quad x \leq z, \\ diag(z) & otherwise, \end{cases} \tag{3}$$

where, $\forall a \in \mathbb{R}$, $diag(a)$ is the $n \times n$ diagonal matrix with diagonal entries equal to a. Let $\mathcal{M}$ be the space of vector-valued functions from $\mathcal{X}$ onto $\mathbb{R}^n$ whose norm $\|g\|_{\mathcal{M}}^2 = \sum_{i=1}^{n}\int_{\mathcal{X}}[g(x)]_i^2 dx$ is finite.

The matrix-valued mapping K is the reproducing kernel of the vector-valued RKHS $\mathcal{F}$ defined as follows:

$$\begin{cases} \mathcal{F} = \left\{ f : [0,1] \longrightarrow \mathbb{R}^n,\ \exists f' = \dfrac{df(x)}{dx} \in \mathcal{M}, [f(x)]_i = \displaystyle\int_0^x [f'(z)]_i dz, \forall i = 1, \ldots, n \right\}, \\ \langle f_1, f_2 \rangle_{\mathcal{F}} = \langle f_1', f_2' \rangle_{\mathcal{M}}. \end{cases}$$

Indeed, K is nonnegative and we have, $\forall x \in \mathcal{X}$, $y \in \mathbb{R}^n$ and $f \in \mathcal{F}$,

$$\begin{aligned} &\langle f, K(x,\cdot)y \rangle_{\mathcal{F}} = \langle f', [K(x,\cdot)y]' \rangle_{\mathcal{M}} \\ &= \sum_{i=1}^n \int_0^1 [f'(z)]_i [K(x,z)y]_i' dz \\ &= \sum_{i=1}^n \int_0^x [f'(z)]_i y_i dz \quad (dK(x,z)/dz = diag(1) \text{ if } z \leq x, \text{ and } = diag(0) \text{ otherwise}) \\ &= \sum_{i=1}^n [f(x)]_i y_i dz \ = \ \langle f(x), y \rangle_{\mathbb{R}^n}. \end{aligned}$$

■

Example 3 *(Function-valued RKHS and its operator-valued kernel)*

Here we extend Example 2 to the case where the output space is infinite dimensional. Let $\mathcal{X} = [0,1]$ and $\mathcal{Y} = L^2(\Omega)$ the space of square integrable functions on a compact set $\Omega \subset \mathbb{R}$. We denote by $\mathcal{M}$ the space of $L^2(\Omega)$-valued functions on $\mathcal{X}$ whose norm $\|g\|_{\mathcal{M}}^2 = \displaystyle\int_\Omega \int_{\mathcal{X}} [g(x)(t)]^2 dx dt$ is finite.

Let $(\mathcal{F}; \langle \cdot, \cdot \rangle_{\mathcal{F}})$ be the space of functions from $\mathcal{X}$ to $L^2(\Omega)$ such that:

$$\begin{cases} \mathcal{F} = \left\{ f,\ \exists f' = \dfrac{df(x)}{dx} \in \mathcal{M}, f(x) = \displaystyle\int_0^x f'(z) dz \right\}, \\ \langle f_1, f_2 \rangle_{\mathcal{F}} = \langle f_1', f_2' \rangle_{\mathcal{M}}. \end{cases}$$

$\mathcal{F}$ is a function-valued RKHS with the operator-valued kernel $K(x,z) = M_{\varphi(x,z)}$. M_φ is the multiplication operator associated with the function φ where $\varphi(x,z)$ is equal to x if $x \leq z$ and z otherwise. Since φ is a positive-definite function, K is Hermitian and nonnegative. Indeed,

$$\begin{aligned} \langle K(z,x)^* y, w \rangle_{\mathcal{Y}} &= \langle y, K(z,x)w \rangle_{\mathcal{Y}} = \int_0^1 \varphi(z,x) w(t) y(t) dt = \int_0^1 \varphi(x,z) y(t) z(t) dt \\ &= \langle K(x,z)y, w \rangle_{\mathcal{Y}}, \end{aligned}$$

and

$$\begin{aligned} \sum_{i,j} \langle K(x_i,x_j) y_i, y_j \rangle_{\mathcal{Y}} &= \sum_{i,j} \int_0^1 \varphi(x_i,x_j) y_i(t) y_j(t) dt \\ &= \int_0^1 \sum_{i,j} y_i(t) \varphi(x_i,x_j) y_j(t) dt \ \geq \ 0 \ (\text{since } \varphi \geq 0). \end{aligned}$$

Now we show that the reproducing property holds for any $f \in \mathcal{F}$, $y \in L^2(\Omega)$ and $x \in \mathcal{X}$:

$$\begin{aligned} &\langle f, K(x,\cdot)y \rangle_{\mathcal{F}} = \langle f', [K(x,\cdot)y]' \rangle_{\mathcal{M}} \\ &= \int_\Omega \int_0^1 [f'(z)](t) [K(x,z)y]'(t) dz dt \\ &\overset{\text{kern. def}}{=} \int_\Omega \int_0^x [f'(z)](t) y(t) dz dt = \int_\Omega [f(x)](t) y(t) dt \\ &= \langle f(x), y \rangle_{L^2(\Omega)}. \end{aligned}$$

■

Theorem 2 states that it is possible to construct a pre-Hilbert space of operators from a nonnegative operator-valued kernel and with some additional assumptions it can be completed to obtain a function-valued reproducing kernel Hilbert space. Therefore, it is important to consider the problem of constructing nonnegative operator-valued kernels. This is the focus of the next section.

5. Operator-valued Kernels for Functional Data

Reproducing kernels play an important role in statistical learning theory and functional estimation. Scalar-valued kernels are widely used to design nonlinear learning methods which have been successfully applied in several machine learning applications (Schölkopf and Smola, 2002; Shawe-Taylor and Cristanini, 2004). Moreover, their extension to matrix-valued kernels has helped to bring additional improvements in learning vector-valued functions (Micchelli and Pontil, 2005a; Reisert and Burkhardt, 2007; Caponnetto and De Vito, 2006). The most common and most successful applications of matrix-valued kernel methods are in multi-task learning (Evgeniou et al., 2005; Micchelli and Pontil, 2005b), even though some successful applications also exist in other areas, such as image colorization (Minh et al., 2010), link prediction (Brouard et al., 2011) and network inference (Lim et al., 2015). A basic, albeit not obvious, question which is always present with reproducing kernels concerns how to build these kernels and what is the optimal kernel choice. This question has been studied extensively for scalar-valued kernels, however it has not been investigated enough in the matrix-valued case. In the context of multi-task learning, matrix-valued kernels are constructed from scalar-valued kernels which are carried over to the vector-valued setting by a positive definite matrix (Micchelli and Pontil, 2005b; Caponnetto et al., 2008).

In this section we consider the problem from a more general point of view. We are interested in the construction of operator-valued kernels, generalization of matrix-valued kernels in infinite dimensional spaces, that perform the mapping between two spaces of functions and which are suitable for functional response data. Our motivation is to build operator-valued kernels that are capable of giving rise to nonlinear FDA methods. It is worth recalling that previous studies have provided examples of operator-valued kernels with infinite-dimensional output spaces (Micchelli and Pontil, 2005a; Caponnetto et al., 2008; Carmeli et al., 2010); however, they did not focus either on building methodological connections with the area of FDA, or on the practical impact of such kernels on real-world applications.

Motivated by building kernels that capture dependencies between samples of functional (infinite-dimensional) response variables, we adopt a FDA modeling formalism. The

design of such kernels will doubtless prove difficult, but it is necessary to develop reliable nonlinear FDA methods. Most FDA methods in the literature are based on linear parametric models. Extending these methods to nonlinear contexts should render them more powerful and efficient. Our line of attack is to construct operator-valued kernels from operators already used to build linear FDA models, particularly those involved in functional response models. Thus, it is important to begin by looking at these models.

5.1 Linear Functional Response Models

FDA is an extension of multivariate data analysis suitable when data are functions. In this framework, a data is a single function observation rather than a collection of observations. It is true that the data measurement process often provides a vector rather than a function, but the vector is a discretization of a real attribute which is a function. Hence, a functional datum i is acquired as a set of discrete measured values, $y_{i1}, \ldots, y_{ip}$; the first task in parametric (linear) FDA methods is to convert these values to a function y_i with values $y_i(t)$ computable for any desired argument value t. If the discrete values are assumed to be noiseless, then the process is interpolation; but if they have some observational error, then the conversion from discrete data to functions is a regression task (*e.g.*, smoothing) (Ramsay and Silverman, 2005).

A functional data model takes the form $y_i = f(x_i) + \epsilon_i$ where one or more of the components y_i, x_i and ϵ_i are functions. Three subcategories of such models can be distinguished: predictors x_i are functions and responses y_i are scalars; predictors are scalars and responses are functions; both predictors and responses are functions. In the latter case, which is the context we face, the function f is a compact operator between two infinite-dimensional Hilbert spaces. Most previous works on this model suppose that the relation between functional responses and predictors is linear; for more details, see Ramsay and Silverman (2005) and references therein.

For functional input and output data, the functional linear model commonly found in the literature is an extension of the multivariate linear one and has the following form:

$$y(t) = \alpha(t) + \beta(t)x(t) + \epsilon(t), \tag{4}$$

where α and β are the functional parameters of the model (Ramsay and Silverman, 2005, Chapter 14). This model is known as the "concurrent model" where "concurrent" means that $y(t)$ only depends on x at t. The concurrent model is similar to the varying coefficient model proposed by Hastie and Tibshirani (1993) to deal with the case where the parameter β of a multivariate regression model can vary over time. A main limitation of this model is that the response y and the covariate x are both functions of the same argument t, and the influence of a covariate on the response is concurrent or point-wise in the sense that x only influences $y(t)$ through its value $x(t)$ at time t. To overcome this restriction, an extended linear model in which the influence of a covariate x can involve a range of argument values $x(s)$ was proposed; it takes the following form:

$$y(t) = \alpha(t) + \int x(s)\beta(s,t)ds + \epsilon(t), \tag{5}$$

where, in contrast to the concurrent model, the functional parameter β is now a function of both s and t, and $y(t)$ depends on $x(s)$ for an interval of values of s (Ramsay and Silverman,

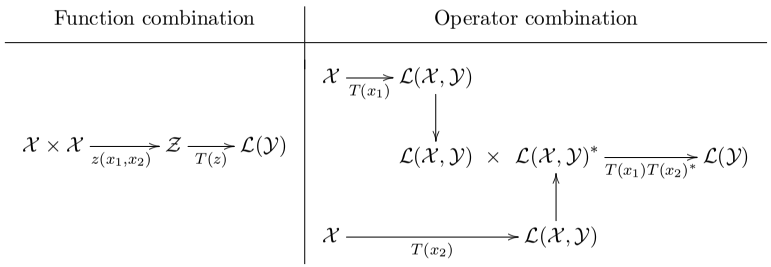


Figure 3: Illustration of building an operator-valued kernel from $\mathcal{X} \times \mathcal{X}$ to $\mathcal{L}(\mathcal{Y})$ using a combination of functions or a combination of operators. (left) The operator-valued kernel is constructed by combining two functions (x_1 and x_2) and by applying a positive $\mathcal{L}(\mathcal{Y})$-valued mapping T to the combination. (right) the operator-valued kernel is generated by combining two operators ($T(x_1)$ and $T(x_2)^*$) built from an $\mathcal{L}(\mathcal{X},\mathcal{Y})$-valued mapping T.

2005, Chapter 16). Estimation of the parameter function $\beta(\cdot,\cdot)$ is an inverse problem and requires regularization. Regularization can be implemented in a variety of ways, for example by penalized splines (James, 2002) or by truncation of series expansions (Müller, 2005). A review of functional response models can be found in Chiou et al. (2004).

The operators involved in the functional data models described above are the multiplication operator (Equation 4) and the integral operator (Equation 5). We think that operator-valued kernels constructed using these operators could be a valid alternative to extend linear FDA methods to nonlinear settings. In Subsection 5.4 we provide examples of multiplication and integral operator-valued kernels. Before that, we identify building schemes that can be common to many operator-valued kernels and applied to functional data.

5.2 Operator-valued Kernel Building Schemes

In our context, constructing an operator-valued kernel turns out to build an operator that maps a couple of functions to a function: in $\mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ from two functions x_1 and x_2 in $\mathcal{X}$. This can be performed in one of two ways: either combining the two functions x_1 and x_2 into a variable $z \in \mathcal{Z}$ and then adding an operator function $T : \mathcal{Z} \longrightarrow \mathcal{L}(\mathcal{Y})$ that performs the mapping from space $\mathcal{Z}$ to $\mathcal{L}(\mathcal{Y})$, or building an $\mathcal{L}(\mathcal{X},\mathcal{Y})$-valued function T, where $\mathcal{L}(\mathcal{X},\mathcal{Y})$ is the set of bounded operators from $\mathcal{X}$ to $\mathcal{Y}$, and then combining the resulting operators $T(x_1)$ and $T(x_2)$ to obtain the operator in $\mathcal{L}(\mathcal{Y})$. In the latter case, a natural way to combine $T(x_1)$ and $T(x_2)$ is to use the composition operation and the kernel $K(x_1,x_2)$ will be equal to $T(x_1)T(x_2)^*$. Figure 3 describes the construction of an operator-valued kernel function using the two schemes which are based on combining functions (x_1 and x_2) or operators ($T(x_1)$ and $T(x_2)$), respectively. Note that separable operator-valued kernels (Álvarez et al., 2012), which are kernels that can be formulated as a product of

a scalar-valued kernel function for the input space alone and an operator that encodes the interactions between the outputs, are a particular case of the function combination building scheme, when we take $\mathcal{Z}$ as the set of real numbers $\mathbb{R}$ and the scalar-valued kernel as combination function. In contrast, the operator combination scheme is particularly amenable to the design of nonseparable operator-valued kernels. This scheme was already used in various problems of operator theory, system theory and interpolation (Alpay et al., 1997; Dym, 1989).

To build an operator-valued kernel and then construct a function-valued reproducing kernel Hilbert space, the operator T is of crucial importance. Choosing T presents two major difficulties. Computing the adjoint operator is not always easy to do, and then, not all operators verify the Hermitian condition of the kernel. On the other hand, since the kernel must be nonnegative, we suggest to construct operator-valued kernels from positive definite scalar-valued kernels which can be the reproducing kernels of real-valued Hilbert spaces. In this case, the reproducing property of the operator-valued kernel allows us to compute an inner product in a space of operators by an inner product in a space of functions which can be, in turn, computed using the scalar-valued kernel. The operator-valued kernel allows the mapping between a space of functions and a space of operators, while the scalar one establishes the link between the space of functions and the space of measured values. It is also useful to define combinations of nonnegative operator-valued kernels that allow to build a new nonnegative one.

5.3 Combinations of Operator-valued Kernels

We have shown in Section 4 that there is a bijection between nonnegative operator-valued kernels and function-valued reproducing kernel Hilbert spaces. So, as in the scalar case, it will be helpful to characterize algebraic transformations, like sum and product, that preserve the nonnegativity of operator-valued kernels. Theorem 3 stated below gives some building rules to obtain a positive operator-valued kernel from combinations of positive existing ones. Similar results for the case of matrix-valued kernels can be found in Reisert and Burkhardt (2007), and for a more general context we refer the reader to Caponnetto et al. (2008) and Carmeli et al. (2010). In our setting, assuming H and G be two nonnegative kernels constructed as described in the previous subsection, we are interested in constructing a nonnegative kernel K from H and G.

Theorem 3 *Let $H : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$ and $G : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$ two nonnegative operator-valued kernels*

(i) *$K \equiv H + G$ is a nonnegative kernel,*

(ii) *if $H(w,z)G(w,z) = G(w,z)H(w,z)$, $\forall w, z \in \mathcal{X}$, then $K \equiv HG$ is a nonnegative kernel,*

(iii) *$K \equiv THT^*$ is a nonnegative kernel for any $\mathcal{L}(\mathcal{Y})$-valued function $T(\cdot)$.*

Proof: Obviously (i) follows from the linearity of the inner product. (ii) can be proved by showing that the "element-wise" multiplication of two positive block operator matrices can be positive (see below). For the proof of (iii), we observe that

$$K(w,z)^* = [T(z)H(w,z)T(w)^*]^* = T(w)H(z,w)T(z)^* = K(z,w),$$

and

$$\begin{aligned}\sum_{i,j}\langle K(w_i,w_j)u_i,u_j\rangle &= \sum_{i,j}\langle T(w_j)H(w_i,w_j)T(w_i)^*u_i,u_j\rangle \\ &= \sum_{i,j}\langle H(w_i,w_j)T(w_i)^*u_i, T(w_j)^*u_j\rangle,\end{aligned}$$

which implies the nonnegativity of the kernel K since H is nonnegative.

To prove (ii), i.e., the kernel $K \equiv HG$ is nonnegative in the case where H and G are nonnegative kernels such that $H(w,z)G(w,z) = G(w,z)H(w,z)$, $\forall w, z \in \mathcal{X}$, we show below that the block operator matrix $\mathbf{K}$ associated to the operator-valued kernel K for a given set $\{w_i\}, i = 1, \ldots, n$ with $n \in \mathbb{N}$, is positive. By construction, we have $\mathbf{K} = \mathbf{H} \circ \mathbf{G}$ where $\mathbf{H}$ and $\mathbf{G}$ are the block operator kernel matrices corresponding to the kernels H and G, and '$\circ$' denotes the "element-wise" multiplication defined by $(\mathbf{H} \circ \mathbf{G})_{ij} = H(w_i,w_j)G(w_i,w_j)$. $\mathbf{K}$, $\mathbf{H}$ and $\mathbf{G}$ are all in $\in \mathcal{L}(\mathcal{Y}^n)$.

Since the kernels H and G are Hermitian and $HG = GH$, it is easy to see that

$$\begin{aligned}(\mathbf{K}^*)_{ij} = (\mathbf{K}_{ji})^* = K(w_j,w_i)^* &= \big(H(w_j,w_i)G(w_j,w_i)\big)^* = G(w_j,w_i)^*H(w_j,w_i)^* \\ &= G(w_i,w_j)H(w_i,w_j) = H(w_i,w_j)G(w_i,w_j) \\ &= \mathbf{K}_{ij}.\end{aligned}$$

Thus, $\mathbf{K}$ is self-adjoint. It remains, then, to prove that $\langle \mathbf{Ku}, \mathbf{u}\rangle \geq 0$, $\forall \mathbf{u} \in \mathcal{Y}^n$, in order to show the positivity of $\mathbf{K}$.

The "element-wise" multiplication can be rewritten as a tensor product. Indeed, we have

$$\mathbf{K} = \mathbf{H} \circ \mathbf{G} = \mathbf{L}^*(\mathbf{H} \otimes \mathbf{G})\mathbf{L},$$

where $\mathbf{L} : \mathcal{Y}^n \longrightarrow \mathcal{Y}^n \otimes \mathcal{Y}^n$ is the mapping defined by $\mathbf{Le_i} = \mathbf{e_i} \otimes \mathbf{e_i}$ for an orthonormal basis $\{\mathbf{e_i}\}$ of the separable Hilbert space $\mathcal{Y}^n$, and $\mathbf{H} \otimes \mathbf{G}$ is the tensor product defined by $(\mathbf{H} \otimes \mathbf{G})(\mathbf{u} \otimes \mathbf{v}) = \mathbf{Hu} \otimes \mathbf{Gv}$, $\forall \mathbf{u}, \mathbf{v} \in \mathcal{Y}^n$. To see this, note that

$$\begin{aligned}\langle \mathbf{L}^*(\mathbf{H} \otimes \mathbf{G})\mathbf{Le_i}, \mathbf{e_j}\rangle = \langle (\mathbf{H} \otimes \mathbf{G})\mathbf{Le_i}, \mathbf{Le_j}\rangle &= \langle (\mathbf{H} \otimes \mathbf{G})(\mathbf{e_i} \otimes \mathbf{e_i}), \mathbf{e_j} \otimes \mathbf{e_j}\rangle \\ &= \langle \mathbf{He_i} \otimes \mathbf{Ge_i}, \mathbf{e_j} \otimes \mathbf{e_j}\rangle = \langle \mathbf{He_i}, \mathbf{e_j}\rangle\langle \mathbf{Ge_i}, \mathbf{e_j}\rangle \\ &= \mathbf{H}_{ij}\mathbf{G}_{ij} = \langle (\mathbf{H} \circ \mathbf{G})\mathbf{e_i}, \mathbf{e_j}\rangle.\end{aligned}$$

Now since H and G are positive, we have

$$\begin{aligned}\langle \mathbf{Ku}, \mathbf{u}\rangle &= \langle \mathbf{L}^*(\mathbf{H} \otimes \mathbf{G})\mathbf{Lu}, \mathbf{u}\rangle = \langle \mathbf{L}^*(\mathbf{H}^{\frac{1}{2}}\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}}\mathbf{G}^{\frac{1}{2}})\mathbf{Lu}, \mathbf{u}\rangle \\ &= \langle \mathbf{L}^*(\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})(\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})\mathbf{Lu}, \mathbf{u}\rangle = \langle (\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})\mathbf{Lu}, (\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})^*\mathbf{Lu}\rangle \\ &= \langle (\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})\mathbf{Lu}, (\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})\mathbf{Lu}\rangle = \|(\mathbf{H}^{\frac{1}{2}} \otimes \mathbf{G}^{\frac{1}{2}})\mathbf{Lu}\|^2 \geq 0.\end{aligned}$$

This concludes the proof. ■

5.4 Examples of Nonnegative Operator-valued Kernels

We provide here examples of operator-valued kernels for functional response data. All these examples deal with operator-valued kernels constructed following the schemes described above and assuming that $\mathcal{Y}$ is an infinite-dimensional function space. Motivated by building kernels suitable for functional data, the first two examples deal with operator-valued kernels constructed from the multiplication and the integral self-adjoint operators in the case where $\mathcal{Y}$ is the Hilbert space $L^2(\Omega_y)$ of square integrable functions on Ω_y endowed with the inner product $\langle \phi, \psi \rangle = \int_{\Omega_y} \phi(t)\psi(t)dt$. We think that these kernels represent an interesting alternative to extend linear functional models to nonlinear settings. The third example based on the composition operator shows how to build such kernels from non self-adjoint operators (this may be relevant when the functional linear model is based on a non self-adjoint operator). It also illustrates the kernel combination defined in Theorem 3(iii).

1. Multiplication operator:

   In Kadri et al. (2010), the authors attempted to extend the widely used Gaussian kernel to functional data domain using a multiplication operator and assuming that input and output data belong to the same space of functions. Here we consider a slightly different setting, where the input space $\mathcal{X}$ can be different from the output space $\mathcal{Y}$.

   A multiplication operator on $\mathcal{Y}$ is defined as follows:

$$\begin{array}{rccc} T^h: & \mathcal{Y} & \longrightarrow & \mathcal{Y} \\ & y & \longmapsto & T^h_y \;\; ; \;\; T^h_y(t) \triangleq h(t)y(t). \end{array}$$

   The operator-valued kernel $K(\cdot,\cdot)$ is the following:

$$\begin{array}{rccc} K: & \mathcal{X}\times\mathcal{X} & \longrightarrow & \mathcal{L}(\mathcal{Y}) \\ & x_1, x_2 & \longmapsto & k_x(x_1,x_2)T^{k_y}, \end{array}$$

   where $k_x(\cdot,\cdot)$ is a positive definite scalar-valued kernel and k_y a positive real function. It is easy to see that $\langle T^h x, y\rangle = \langle x, T^h y\rangle$, then T^h is a self-adjoint operator. Thus $K(x_2,x_1)^* = K(x_2,x_1)$ and K is Hermitian since $K(x_1,x_2) = K(x_2,x_1)$.

   Moreover, we have

$$\begin{aligned} \sum_{i,j}\langle K(x_i,x_j)y_i, y_j\rangle_{\mathcal{Y}} &= \sum_{i,j} k_x(x_i,x_j)\langle k_y(\cdot)y_i(\cdot), y_j(\cdot)\rangle_{\mathcal{Y}} \\ &= \sum_{i,j} k_x(x_i,x_j)\int k_y(t)y_i(t)y_j(t)dt = \int \sum_{i,j} y_i(t)[k_x(x_i,x_j)k_y(t)]y_j(t)dt \geq 0, \end{aligned}$$

   since the product of two positive-definite scalar-valued kernels is also positive-definite. Therefore K is a nonnegative operator-valued kernel.

2. Hilbert-Schmidt integral operator:

   A Hilbert-Schmidt integral operator on $\mathcal{Y}$ associated with a kernel $h(\cdot,\cdot)$ is defined as follows:

$$\begin{array}{rccc} T^h: & \mathcal{Y} & \longrightarrow & \mathcal{Y} \\ & y & \longmapsto & T^h_y \;\; ; \;\; T^h_y(t) \triangleq \int h(s,t)y(s)ds. \end{array}$$

   In this case, an operator-valued kernel K is a Hilbert-Schmidt integral operator associated with positive definite scalar-valued kernels k_x and k_y, and it takes the following form:

$$\begin{array}{rccc} K(x_1,x_2)[\cdot]: & \mathcal{Y} & \longrightarrow & \mathcal{Y} \\ & f & \longmapsto & g \end{array}$$

   where $g(t) = k_x(x_1,x_2)\int k_y(s,t)f(s)ds$.

   The Hilbert-Schmidt integral operator is self-adjoint if k_y is Hermitian. This condition is verified and then it is easy to check that K is also Hermitian. K is nonnegative since

$$\sum_{i,j}\langle K(x_i,x_j)y_i, y_j\rangle_{\mathcal{Y}} = \iint \sum_{i,j} y_i(s)[k_x(x_i,x_j)k_y(s,t)]y_j(t)dsdt,$$

   which is positive because of the positive-definiteness of the scalar-valued kernels k_x and k_y.

3. Composition operator:

   Let φ be an analytic map. The composition operator associated with φ is the linear map:

$$C_\varphi : f \longmapsto f\circ\varphi$$

   First, we look for an expression of the adjoint of the composition operator C_φ acting on $\mathcal{Y}$ in the case where $\mathcal{Y}$ is a scalar-valued RKHS of functions on Ω_y and φ an analytic map of Ω_y into itself. For any f in the space $\mathcal{Y}$ associated with the real kernel k,

$$\begin{array}{rcccc} \langle f, C^*_\varphi k_t(\cdot)\rangle & = & \langle C_\varphi f, k_t\rangle & = & \langle f\circ\varphi, k_t\rangle \\ & = & f(\varphi(t)) & = & \langle f, k_{\varphi(t)}\rangle. \end{array}$$

   This is true for any $f\in\mathcal{Y}$ and then $C^*_\varphi k_t = k_{\varphi(t)}$. In a similar way, $C^*_\varphi f$ can be computed at each point of the function f:

$$(C^*_\varphi f)(t) = \langle C^*_\varphi f, k_t\rangle = \langle f, C_\varphi k_t\rangle = \langle f, k_t\circ\varphi\rangle$$

   Once we have expressed the adjoint of a composition operator in a reproducing kernel Hilbert space, we consider the following operator-valued kernel:

$$\begin{array}{rccc} K: & \mathcal{X}\times\mathcal{X} & \longrightarrow & \mathcal{L}(\mathcal{Y}) \\ & x_1, x_2 & \longmapsto & C_{\psi(x_1)}C^*_{\psi(x_2)} \end{array}$$

   where $\psi(x_1)$ and $\psi(x_2)$ are maps of Ω_y into itself. It is easy to see that the kernel K is Hermitian. Using Theorem 3(iii) we obtain the nonnegativity property of the kernel.

5.5 Multiple Functional Data and Kernel Feature Map

Until now, we discussed operator-valued kernels and their corresponding RKHS from the perspective of extending Aronszajn (1950) pioneering work from scalar-valued or vector-valued cases to the function-valued case. However, it is also interesting to explore these kernels from a feature space point of view (Schölkopf et al., 1999; Caponnetto et al., 2008). In this subsection, we provide some ideas targeted at advancing the understanding of feature spaces associated with operator-valued kernels and we show how these kernels can design more suitable feature maps than those associated with scalar-valued kernels, especially when input data are infinite dimensional objects like curves. To explore the potential of adopting an operator-valued kernel feature space approach, we consider a supervised learning problem with multiple functional data where each observation is composed of more than one functional variable (Kadri et al., 2011b,c). Working with multiple functions allows to deal in a natural way with a lot of applications. There are many practical situations where a number of potential functional covariates are available to explain a response variable. For example, in audio and speech processing where signals are converted into different functional features providing information about their temporal, spectral and cepstral characteristics, or in meteorology where the interaction effects between various continuous variables (such as temperature, precipitation, and winds) is of particular interest.

Similar to the scalar case, operator-valued kernels provide an elegant way of dealing with nonlinear algorithms by reducing them to linear ones in some feature space F nonlinearly related to input space. A feature map associated with an operator-valued kernel K is a continuous function

$$\Phi : \mathcal{X} \times \mathcal{Y} \longrightarrow \mathcal{L}(\mathcal{X}, \mathcal{Y}),$$

such that for every $x_1, x_2 \in \mathcal{X}$ and $y_1, y_2 \in \mathcal{Y}$

$$\langle K(x_1, x_2)y_1, y_2\rangle_{\mathcal{Y}} = \langle \Phi(x_1, y_1), \Phi(x_2, y_2)\rangle_{\mathcal{L}(\mathcal{X},\mathcal{Y})},$$

where $\mathcal{L}(\mathcal{X}, \mathcal{Y})$ is the set of linear mappings from $\mathcal{X}$ into $\mathcal{Y}$. By virtue of this property, Φ is called a *feature map associated with* K. Furthermore, from the reproducing property, it follows that in particular

$$\langle K(x_1, \cdot)y_1, K(x_2, \cdot)y_2\rangle_{\mathcal{F}} = \langle K(x_1, x_2)y_1, y_2\rangle_{\mathcal{Y}},$$

which means that any operator-valued kernel admits a feature map representation Φ with a feature space $\mathcal{F} \subset \mathcal{L}(\mathcal{X}, \mathcal{Y})$ defined by $\Phi(x_1, y_1) = K(x_1, \cdot)y_1$, and corresponds to an inner product in another space.

From this feature map perspective, we study the geometry of a feature space associated with an operator-valued kernel and we compare it with the geometry obtained by a scalar-valued kernel. More precisely, we consider two reproducing kernel Hilbert spaces $\mathcal{F}$ and $\mathcal{H}$. $\mathcal{F}$ is a RKHS of function-valued functions on $\mathcal{X}$ with values in $\mathcal{Y}$. $\mathcal{X} \subset (L^2(\Omega_x))^p$ [7], $\mathcal{Y} \subset L^2(\Omega_y)$ and let K be the reproducing operator-valued kernel of $\mathcal{F}$. $\mathcal{H}$ is also a RKHS, but of scalar-valued functions on $\mathcal{X}$ with values in $\mathbb{R}$, and k its reproducing scalar-valued

7. p is the number of functions that represent input data. In the field of FDA, such data are called multivariate functional data.

kernel. The mappings Φ_K and Φ_k associated, respectively, with the kernels K and k are defined as follows

$$\Phi_K : (L^2)^p \to \mathcal{L}((L^2)^p, L^2), \quad x \mapsto K(x, \cdot)y,$$

and

$$\Phi_k : (L^2)^p \to \mathcal{L}((L^2)^p, \mathbb{R}), \quad x \mapsto k(x, \cdot).$$

These feature maps can be seen as a mapping of the input data x_i, which are vectors of functions in $(L^2)^p$, into a feature space in which the inner product can be computed using the kernel functions. This idea leads to design nonlinear methods based on linear ones in the feature space. In a supervised classification problem for example, since kernels map input data into a higher dimensional space, kernel methods deal with this problem by finding a linear separation in the feature space. We now compare the dimension of feature spaces obtained by the maps Φ_K and Φ_k. To do this, we adopt a functional data analysis point of view where observations are composed of sets of functions. Direct understanding of this FDA viewpoint comes from the consideration of the "atom" of a statistical analysis. In a basic course in statistics, atoms are "numbers", while in multivariate data analysis the atoms are vectors and methods for understanding populations of vectors are the focus. FDA can be viewed as the generalization of this, where the atoms are more complicated objects, such as curves, images or shapes represented by functions (Zhao et al., 2004). Based on this, the dimension of the input space is p since $x_i \in (L^2)^p$ is a vector of p functions. The feature space obtained by the map Φ_k is a space of functions, so its dimension from a FDA viewpoint is equal to one. The map Φ_K projects the input data into a space of operators $\mathcal{L}(\mathcal{X}, \mathcal{Y})$. This means that using the operator-valued kernel K corresponds to mapping the functional data x_i into a higher, possibly infinite, dimensional space $(L^2)^d$ with $d \to \infty$. In a binary functional classification problem, we have higher probability to achieve linear separation between the classes by projecting the functional data into a higher dimensional feature space rather than into a lower one (Cover's theorem), that is why we think that it is more suitable to use operator-valued than scalar-valued kernels in this context.

6. Function-valued Function Learning

In this section, we consider the problem of estimating an unknown function F such that $F(x_i) = y_i$ when observed data $(x_i(s), y_i(t))_{i=1}^n \in \mathcal{X} \times \mathcal{Y}$ are assumed to be elements of the space of square integrable functions L^2. $X = \{x_1, \ldots, x_n\}$ denotes the training set with corresponding targets $Y = \{y_1, \ldots, y_n\}$. Since $\mathcal{X}$ and $\mathcal{Y}$ are spaces of functions, the problem can be thought of as an operator estimation problem, where the desired operator maps a Hilbert space of factors to a Hilbert space of targets. Among all functions in a linear space of operators $\mathcal{F}$, an estimate $\widetilde{F} \in \mathcal{F}$ of F may be obtained by minimizing:

$$\widetilde{F} = \arg\min_{F \in \mathcal{F}} \sum_{i=1}^{n} \|y_i - F(x_i)\|_{\mathcal{Y}}^2.$$

Depending on $\mathcal{F}$, this problem can be ill-posed and a classical way to turn it into a well-posed problem is to use a regularization term. Therefore, we may consider the solution of

the problem as the function $\widetilde{F} \in \mathcal{F}$ that minimizes:

$$\widetilde{F}_\lambda = \arg\min_{F\in\mathcal{F}} \sum_{i=1}^{n} \|y_i - F(x_i)\|_{\mathcal{Y}}^2 + \lambda \|F\|_{\mathcal{F}}^2, \tag{6}$$

where $\lambda \in \mathbb{R}^+$ is a regularization parameter. Existence of $\widetilde{F}_\lambda$ in the optimization problem (6) is guaranteed for $\lambda > 0$ by the generalized Weierstrass theorem and one of its corollary that we recall from Kurdila and Zabarankin (2005).

Theorem 4 *Let $\mathcal{Z}$ be a reflexive Banach space and $\mathcal{C} \subseteq \mathcal{Z}$ a weakly closed and bounded set. Suppose $J : \mathcal{C} \to \mathbb{R}$ is a proper lower semi-continuous function. Then J is bounded from below and has a minimizer on $\mathcal{C}$.*

Corollary 5 *Let $\mathcal{H}$ be a Hilbert space and $J : \mathcal{H} \to \mathbb{R}$ is a strongly lower semi-continuous, convex and coercive function. Then J is bounded from below and attains a minimizer.*

This corollary can be straightforwardly applied to problem (6) by defining:

$$J_\lambda(F) = \sum_{i=1}^{n} \|y_i - F(x_i)\|_{\mathcal{Y}}^2 + \lambda \|F\|_{\mathcal{F}}^2,$$

where F belongs to the Hilbert space $\mathcal{F}$. It is easy to note that J_λ is continuous and convex. Besides, J_λ is coercive for $\lambda > 0$ since $\|F\|_{\mathcal{F}}^2$ is coercive and the sum involves only positive terms. Hence $\widetilde{F}_\lambda = \arg\min_{F\in\mathcal{F}} J_\lambda(F)$ exists.

6.1 Learning Algorithm

We are now interested in solving the minimization problem (6) in a reproducing kernel Hilbert space $\mathcal{F}$ of function-valued functions. In the scalar case, it is well-known that under general conditions on real-valued RKHS, the solution of this minimization problem can be written as:

$$\widetilde{F}(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x),$$

where $\alpha_i \in \mathbb{R}$ and k is the reproducing kernel of a real-valued Hilbert space (Wahba, 1990). An extension of this solution to the domain of functional data analysis takes the following form:

$$\widetilde{F}(\cdot) = \sum_{i=1}^{n} K(x_i, \cdot) u_i, \tag{7}$$

where $u_i(\cdot)$ are in $\mathcal{Y}$ and the reproducing kernel K is a nonnegative operator-valued function. With regards to the classical representer theorem, here the kernel K outputs an operator and the "weights" u_i are functions. A proof of the representer theorem in the case of function-valued reproducing kernel Hilbert spaces is given in Appendix B (see also Micchelli and Pontil, 2005a).

Substituting (7) in (6) and using the reproducing property of $\mathcal{F}$, we come up with the following minimization problem over the scalar-valued functions $u_i \in \mathcal{Y}$ ($\mathbf{u}$ is the vector of functions $(u_i)_{i=1,\ldots,n} \in (\mathcal{Y})^n$) rather than the function-valued function (or operator) F:

$$\widetilde{\mathbf{u}}_\lambda = \arg\min_{\mathbf{u}\in(\mathcal{Y})^n} \sum_{i=1}^{n} \|y_i - \sum_{j=1}^{n} K(x_i, x_j) u_j\|_{\mathcal{Y}}^2 + \lambda \sum_{i,j}^{n} \langle K(x_i, x_j) u_i, u_j \rangle_{\mathcal{Y}}. \tag{8}$$

Problem (8) can be solved in three ways:

1. Assuming that the observations are made on a regular grid $\{t_1, \ldots, t_m\}$, one can first discretize the functions x_i and y_i and then solve the problem using multivariate data analysis techniques (Kadri et al., 2010). However, as this is well-known in the FDA domain, this has the drawback of not taking into consideration the relationships that exist between samples.

2. The second way consists in considering the output space $\mathcal{Y}$ to be a scalar-valued reproducing Hilbert space. In this case, the functions u_i can be approximated by a linear combination of a scalar-valued kernel $\hat{u}_i = \sum_{l=1}^{m} \alpha_{il} k(s_l, \cdot)$ and then the problem (8) becomes a minimization problem over the real values α_{il} rather than the discrete values $u_i(t_1), \ldots, u_i(t_m)$. In the FDA literature, a similar idea has been adopted by Ramsay and Silverman (2005) and by Prchal and Sarda (2007) who expressed not only the functional parameters u_i but also the observed input and output data in a basis functions specified a priori (*e.g.*, Fourier basis or B-spline basis).

3. Another possible way to solve the minimization problem (8) is to compute its derivative using the directional derivative and setting the result to zero to find an analytic solution of the problem. It follows that the vector of functions $\mathbf{u} \in \mathcal{Y}^n$ satisfies the system of linear operator equations:

$$(\mathbf{K} + \lambda I)\mathbf{u} = \mathbf{y}, \tag{9}$$

where $\mathbf{K} = [K(x_i, x_j)]_{i,j=1}^n$ is a $n \times n$ block operator kernel matrix ($\mathbf{K}_{ij} \in \mathcal{L}(\mathcal{Y})$) and $\mathbf{y} \in \mathcal{Y}^n$ the vector of functions $(y_i)_{i=1}^n$. In this work, we are interested in this third approach which extends to functional data analysis domain results and properties known from multivariate statistical analysis. One main obstacle for this extension is the inversion of the block operator kernel matrix $\mathbf{K}$. Block operator matrices generalize block matrices to the case where the block entries are linear operators between infinite dimensional Hilbert spaces. These matrices and their inverses arise in some areas of mathematics (Tretter, 2008) and signal processing (Asif and Moura, 2005). In contrast to the multivariate case, inverting such matrices is not always feasible in infinite dimensional spaces. To overcome this problem, we study the eigenvalue decomposition of a class of block operator kernel matrices obtained from operator-valued kernels having the following form:

$$K(x_i, x_j) = g(x_i, x_j) T, \quad \forall x_i, x_j \in \mathcal{X}, \tag{10}$$

where g is a scalar-valued kernel and T is an operator in $\mathcal{L}(\mathcal{Y})$. This separable kernel construction is adapted from Micchelli and Pontil (2005a,b). The choice of T depends

on the context. For multi-task kernels, T is a finite dimensional matrix which models relations between tasks. In FDA, Lian (2007) suggested the use of the identity operator, while in Kadri et al. (2010) the authors showed that it is better to choose other operators than identity to take into account functional properties of the input and output spaces. They introduced a functional kernel based on the multiplication operator. In this work, we are more interested in kernels constructed from the integral operator. This seems to be a reasonable choice since functional linear model (see Equation 5) are based on this operator (Ramsay and Silverman, 2005, Chapter 16). So we can consider for example the following positive definite operator-valued kernel:

$$(K(x_i, x_j)y)(t) = g(x_i, x_j) \int_{\Omega_y} e^{-|t-s|} y(s) ds, \tag{11}$$

where $y \in \mathcal{Y} = L^2(\Omega_y)$ and $\{s, t\} \in \Omega_y = [0, 1]$. Note that a similar kernel was proposed as an example in Caponnetto et al. (2008) for linear spaces of functions from $\mathbb{R}$ to $\mathcal{G}_y$.

The $n \times n$ block operator kernel matrix $\mathbf{K}$ of operator-valued kernels having the form (10) can be expressed as a Kronecker product between the Gram matrix $G = \big(g(x_i, x_j)\big)_{i,j=1}^{n}$ in $\mathbb{R}^{n \times n}$ and the operator $T \in \mathcal{L}(\mathcal{Y})$, and is defined as follows:

$$\mathbf{K} = \begin{pmatrix} g(x_1, x_1)T & \dots & g(x_1, x_n)T \\ \vdots & \ddots & \vdots \\ g(x_n, x_1)T & \dots & g(x_n, x_n)T \end{pmatrix} = G \otimes T.$$

It is easy to show that basic properties of the Kronecker product between two finite matrices can be restated for this case. So, $\mathbf{K}^{-1} = G^{-1} \otimes T^{-1}$ and the eigendecomposition of the matrix $\mathbf{K}$ can be obtained from the eigendecompositions of G and T (see Algorithm 1).

Theorem 6 *If $T \in \mathcal{L}(\mathcal{Y})$ is a compact, normal operator ($TT^* = T^*T$) on the Hilbert space $\mathcal{Y}$, then there exists an orthonormal basis of eigenfunctions $\{\phi_i, i \geq 1\}$ corresponding to eigenvalues $\{\lambda_i, i \geq 1\}$ such that*

$$Ty = \sum_{i=1} \lambda_i \langle y, \phi_i \rangle \phi_i, \quad \forall y \in \mathcal{Y}.$$

Proof: See Naylor and Sell (1971)[theorem 6.11.2] ■

Let θ_i and $\mathbf{z_i}$ be, respectively, the eigenvalues and the eigenfunctions of $\mathbf{K}$. From Theorem 6 it follows that the inverse operator $\mathbf{K}^{-1}$ is given by

$$\mathbf{K}^{-1}\mathbf{c} = \sum_i \theta_i^{-1} \langle \mathbf{c}, \mathbf{z_i} \rangle \mathbf{z_i}, \quad \forall \mathbf{c} \in \mathcal{Y}^n.$$

Now we are able to solve the system of linear operator equations (9) and the functions u_i can be computed from eigenvalues and eigenfunctions of the matrix $\mathbf{K}$, as described in Algorithm 1.

Algorithm 1 L^2-Regularized Function-valued Function Learning Algorithm

Input
Examples:
(function) data $x_i \in (L^2([0,1]))^p$, size n
(function) labels $y_i \in L^2([0,1])$, size n
Parameters: g, T, κ, λ
Eigendecomposition of G, the Gram matrix of the scalar-valued kernel g
Comment: $G = g(x_i, x_j)_{i,j=1}^{n} \in \mathbb{R}^{n \times n}$
Let $\alpha_i \in \mathbb{R}$ the n eigenvalues
Let $v_i \in \mathbb{R}^n$ the n eigenvectors
Eigendecomposition of the operator $T \in \mathcal{L}(Y)$
Choose κ, the number of computed eigenfunctions
Compute κ $(\delta_i \in \mathbb{R}, w_i \in L^2([0,1]))$ pairs of (eigenvalue, eigenfunction)
Eigendecomposition of $\mathbf{K} = G \otimes T$
Comment: $\mathbf{K} = K(x_i, x_j)_{i,j=1}^{n} \in (\mathcal{L}(Y))^{n \times n}$
The eigenvalues $\theta_i \in \mathbb{R}$, size $n \times \kappa$, are obtained as: $\theta = \alpha \otimes \delta$
The eigenfunctions $\mathbf{z_i} \in (L^2([0,1]))^n$, size $n \times \kappa$, are obtained as: $\mathbf{z} = v \otimes w$
Solution of problem (8) $\mathbf{u} = (\mathbf{K} + \lambda I)^{-1}\mathbf{y}$
Initialize λ: regularization parameter
$\mathbf{u} = \sum_{i=1}^{n \times \kappa} [(\theta_i + \lambda)^{-1} \sum_{j=1}^{n} \langle z_{ij}, y_j \rangle \mathbf{z}_i]$

To put our algorithm into context, we remind that a crucial question about the applicability of functional data is how one can find an appropriate space and a basis in which the functions can be decomposed in a computationally feasible way while taking into account the functional nature of the data. This is exactly what Algorithm 1 does. In contrast to parametric FDA methods, the basis function here is not fixed in advance but implicitly defined by choosing a *reproducing operator-valued kernel* acting on both input and output data. The spectral decomposition of the block operator kernel matrix naturally allows the assignment of an appropriate basis function to the learning process for representing input and output functions. Moreover, the formulation is flexible enough to be used with different operators and then to be adapted for various applications involving functional data. Also, in the context of nonparametric FDA where the notion of semi-metric plays an important role in modeling functional data, we note that Algorithm 1 is based on computing and choosing a finite number of eigenfunctions. This is strongly related to the semi-metric building scheme in Ferraty and Vieu (2006) which is based on, for example, functional principal components or successive derivatives. Operator-valued kernels constructed from the covariance operator (Kadri et al., 2013b) or the derivative operator will allow to design semi-metrics similar to those just mentioned. In this sense, the eigendecomposition of the block operator kernel matrix offers a new way of producing semi-metrics.

6.2 Generalization Analysis

Here, we provide an analysis of the generalization error of the function-valued function learning model (6) using the notion of algorithmic stability. For more details and results

with the least squares loss and other loss function (including ϵ-sensitive loss and logistic loss), see Audiffren and Kadri (2013). In the case of vector-valued functions, the effort in this area has already produced several successful results, including Baxter (2000), Ando and Zhang (2005), Maurer (2006), and Maurer and Pontil (2013). Yet, these studies have considered only the case of finite-dimensional output spaces, and have focused rather on linear machines than on nonlinear ones. To our knowledge, the first work investigating the generalization performance of nonlinear vector-valued function learning methods when output spaces can be infinite-dimensional is that of Caponnetto and De Vito (2006). In their study, from a theoretical analysis based on the concept of effective dimension, the authors have derived generalization bounds for the learning model (6) when the hypothesis space is an RKHS with operator-valued kernels.

The convergence rates in Caponnetto and De Vito (2006), although optimal in the case of finite-dimensional output spaces, require assumptions on the kernel that can be restrictive in the infinite-dimensional case. Indeed, their proof depends upon the fact that the trace of the operator $K(x,x)$ is finite (K is the operator-valued kernel function), and this restricts the applicability of their results when the output space is infinite-dimensional. To illustrate this, let us consider the identity operator-valued kernel $K(\cdot,\cdot) = k(\cdot,\cdot)I$, where k is a scalar-valued kernel and I is the identity operator. This simple kernel does not satisfy the finite trace condition and therefore the results of Caponnetto and De Vito (2006) cannot be applied in this case. Regarding the examples of operator-valued kernels given in Subsection 5.4, the kernel built from the integral operator satisfies the finite trace condition, while that based on the multiplication operator does not. To address this issue, we first show that our learning algorithm is uniformly stable, and then we derive under mild assumption on the kernel, using a result from Bousquet and Elisseeff (2002), a generalization bound which holds even when the finite trace condition is not satisfied.

We now state and discuss the main assumptions we need to prove a stability-based bound on the generalization error of our method. In the following, we consider a training set $Z = \{(x_1, y_1), ..., (x_n, y_n)\}$ of size n in $\mathcal{X} \times \mathcal{Y}$ drawn i.i.d. from an unknown distribution P, and we denote by $Z^i = Z \setminus (x_i, y_i)$ the set Z from which the couple (x_i, y_i) is removed. We will use a cost function $c : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$. The loss of an hypothesis F with respect to an example (x, y) is then defined as $\ell(y, F, x) = c(F(x), y)$. The generalization error is defined as:

$$R(F) = \int \ell(y, F(x), x) dP(x, y),$$

and the empirical error as:

$$R_{emp}(F, Z) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, F, x_i).$$

A learning algorithm can be viewed as a function which maps a training set Z onto a function F_Z from $\mathcal{X}$ to $\mathcal{Y}$ (Bousquet and Elisseeff, 2002). In our case, F_Z is the solution of the optimization problem (6) which is an instance of the following scheme

$$F_Z = \arg\min_{F \in \mathcal{F}} R_{reg}(F, Z), \tag{12}$$

where $R_{reg}(F, Z) = R_{emp}(F, Z) + \lambda \|F\|^2_{\mathcal{F}}$.

Assumption 1 *$\exists \kappa > 0$ such that $\forall x \in \mathcal{X}$,*

$$\|K(x,x)\|_{op} \leq \kappa^2,$$

where $\|K(x,x)\|_{op} = \sup_{y \in \mathcal{Y}} \frac{\|K(x,x)y\|_{\mathcal{Y}}}{\|y\|_{\mathcal{Y}}}$ is the operator norm of $K(x,x)$ on $L(\mathcal{Y})$.

Assumption 2 *The real function from $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$*

$$(x_1, x_2) \mapsto \langle K(x_1, x_2)y_1, y_2 \rangle_{\mathcal{Y}} \text{ is measurable } \forall y_1, y_2 \in \mathcal{Y}.$$

Assumption 3 *The application $(y, f, x) \mapsto \ell(y, F, x)$ is σ-admissible, i.e. convex with respect to F and Lipschitz continuous with respect to $F(x)$, with σ its Lipschitz constant.*

Assumption 4 *$\exists \xi > 0$ such that $\forall (x, y) \in \mathcal{X} \times \mathcal{Y}$ and $\forall Z$ a training set,*

$$\ell(y, F_Z, x) \leq \xi.$$

Note that Assumption 1 is a direct extension from the scalar-valued to the operator-valued case of the boundedness condition of the kernel function. It replaces and weakens the finite trace assumption of the operator $K(x,x)$ used in Caponnetto and De Vito (2006); see Remark 1 for more details. Assumption 2 was also used by Caponnetto and De Vito (2006) to avoid problems with measurability. This assumption with the fact that $\mathcal{F}$ is separable ensures that all functions in $\mathcal{F}$ are measurable from $\mathcal{X}$ to $\mathcal{Y}$. Assumptions 3 and 4 are the same as those used by Bousquet and Elisseeff (2002) for learning scalar-valued functions. As a consequence of Assumption 1, we immediately obtain the following elementary lemma which allows to control $\|F(x)\|_{\mathcal{Y}}$ with $\|F\|_{\mathcal{F}}$.

Lemma 1 *Let K be a nonnegative operator-valued kernel satisfying Assumption 1. Then $\forall F \in \mathcal{F}$, $\|F(x)\|_{\mathcal{Y}} \leq \kappa \|F\|_{\mathcal{F}}$.*

Proof:

$$\begin{aligned}
\|F(x)\|_{\mathcal{Y}} &= \sup_{\|y\|=1} |\langle F(x), y \rangle_{\mathcal{Y}}| = \sup_{\|y\|=1} |\langle F(\cdot), K(x,\cdot)y \rangle_{\mathcal{F}}| \\
&\leq \|F(\cdot)\|_{\mathcal{F}} \sup_{\|y\|=1} \|K(x,\cdot)y\|_{\mathcal{F}} \leq \|F(\cdot)\|_{\mathcal{F}} \sup_{\|y\|=1} \sqrt{\langle K(x,x)y, y \rangle_{\mathcal{Y}}} \\
&\leq \|F(\cdot)\|_{\mathcal{F}} \sup_{\|y\|=1} \|K(x,x)y\|^{\frac{1}{2}}_{\mathcal{Y}} \leq \|F(\cdot)\|_{\mathcal{F}} \|K(x,x)\|^{\frac{1}{2}}_{op} \leq \kappa \|F\|_{\mathcal{F}}
\end{aligned}$$

■

Now we are ready to state the stability theorem for our function-valued function learning algorithm. This result is a straightforward extension of Theorem 22 in Bousquet and Elisseeff (2002) to the case of infinite-dimensional output spaces. It is worth pointing out that the proof does not differ much from the scalar-valued case and requires only minor modifications to fit the operator-valued kernel approach. For the convenience of the reader, we present in Appendix C the proof taking into account these modifications. Before stating the theorem we would like to recall the definition of uniform algorithmic stability from Bousquet and Elisseeff (2002).

Definition 6 *A learning algorithm $Z \mapsto F_Z$ has uniform stability β with respect to the loss function ℓ if the following holds*

$$\forall n \geq 1,\ \forall 1 \leq i \leq n,\ \forall Z \text{ a training set},\ \ \|\ell(\cdot, F_Z, \cdot) - \ell(\cdot, F_{Z^i}, \cdot)\|_\infty \leq \beta$$

Theorem 7 *Under Assumptions 1, 2 and 3, a learning algorithm that maps a training set Z to the function F_Z defined in (12) is β stable with*

$$\beta = \frac{\sigma^2 \kappa^2}{2\lambda n}.$$

Proof: See Appendix C. ■

β scales as $1/n$. This allows to get a bound on the generalization error using a result from Bousquet and Elisseeff (2002).

Theorem 8 *Let $Z \mapsto F_Z$ be a learning algorithm with uniform stability β with respect to a loss ℓ that satisfies Assumption 4. Then, $\forall n \geq 1$, $\forall\ 0 \leq \delta \leq 1$, the following bound holds with probability at least $1 - \delta$ over the random draw of training samples*

$$R \leq R_{emp} + 2\beta + (4n\beta + \xi)\sqrt{\frac{\ln(1/\delta)}{2n}}.$$

Proof: See Theorem 12 in Bousquet and Elisseeff (2002). ■

For our learning model (6), we should note that Assumption 3 is in general not satisfied with the least squares loss function $\ell(y, F, x) = \|y - F(x)\|^2_{\mathcal{Y}}$. To address this issue, one can add a boundedness assumption on $\mathcal{Y}$, which is a sufficient condition to prove the uniform stability when Assumption 1 is satisfied.

Assumption 5 *$\exists \sigma_y > 0$ such that $\|y\|_{\mathcal{Y}} < \sigma_y$, $\forall y \in \mathcal{Y}$.*

Lemma 2 *Let $\ell(y, F, x) = \|y - F(x)\|^2_{\mathcal{Y}}$. If Assumptions 1 and 5 hold, then*

$$|\ell(y, F_Z, x) - \ell(y, F_{Z^i}, x)| \leq \sigma \|F_Z(x) - F_{Z^i}(x)\|_{\mathcal{Y}},$$

with $\sigma = 2\sigma_y(1 + \frac{\kappa}{\sqrt{\lambda}})$.

Proof: See Appendix D. ■

This Lemma can replace the Lipschitz property of ℓ in the proof of Theorem 7. Moreover, Assumptions 1 and 5 are sufficient to satisfy Assumption 4 with $\xi = (\sigma/2)^2$ (see Appendix D). We can then use Theorem 7 to prove the uniform stability of our function-valued function learning algorithm with

$$\beta = \frac{2\kappa^2 \sigma_y^2 (1 + \frac{\kappa}{\sqrt{\lambda}})^2}{\lambda n}.$$

Theorem 8 thus gives us a bound on the generalization error of our method equal, with probability at least $1 - \delta$, to

$$R \leq R_{emp} + \frac{4\kappa^2 \sigma_y^2 (1 + \frac{\kappa}{\sqrt{\lambda}})^2}{\lambda n} + \sigma_y^2 (1 + \frac{\kappa}{\sqrt{\lambda}})^2 (\frac{8\kappa^2}{\lambda} + 1)\sqrt{\frac{\ln(1/\delta)}{2n}}.$$

Remark 1 *It is important to stress that even though the stability analysis of function-valued function learning algorithms follows in a quite straightforward fashion from the earlier results presented in Bousquet and Elisseeff (2002) and provides convergence rates which are not optimal, it allows to derive generalization error bounds with operator-valued kernels for which the trace of the operator $K(x, x)$ is not necessarily finite. Assumption 1 is weaker than the one used in Caponnetto and De Vito (2006) which requires that the operator K_x is Hilbert-Schmidt[8] and $\sup_{x \in \mathcal{X}} Tr(K(x, x)) < \kappa$. While the two assumptions are equivalent when the output space $\mathcal{Y}$ is finite dimensional, this is no longer the case when, as in this paper, $\dim \mathcal{Y} = +\infty$. Moreover, we observe that if the assumption of Caponnetto and De Vito (2006) is satisfied, then our Assumption 1 holds (see proof in Appendix E). The converse is not true (see Remark 2 for a counterexample).*

Remark 2 *Note that the operator-valued kernel based on the multiplication operator and described in Subsection 5.4 satisfies Assumption 1 but not the finite trace condition as assumed in Caponnetto and De Vito (2006). Let k be a positive-definite scalar-valued kernel such that $\sup_{x \in \mathcal{X}} k(x, x) < +\infty$, $\mathcal{I}$ an interval of $\mathbb{R}$, $\mu > 0$, and $\mathcal{Y} = L^2(\mathcal{I}, \mathbb{R})$. Let $f \in L^\infty(\mathcal{I}, \mathbb{R})$ be such that $\|f\|_\infty < \mu$. Consider the following multiplication operator-valued kernel K:*

$$K(x, z)y(\cdot) = k(x, z) f^2(\cdot) y(\cdot) \in \mathcal{Y}.$$

K is a nonnegative operator-valued kernel. While K always satisfies Assumption 1, the Hilbert-Schmidt property of K_x depends on the choice of f and does not hold in general. For instance, let $f(t) = \frac{\mu}{2}(\exp(-t^2) + 1)$, then

$$\|K(x, x)\|_{op} \leq \mu^2 k(x, x),$$

and

$$Tr(K(x, x)) = \sum_{j \in \mathbb{N}} \langle K(x, x) y_j, y_j \rangle \geq k(x, x) \frac{\mu}{2} \sum_{i \in \mathbb{N}} \|y_j\|_2^2 = \infty,$$

where $(y_j)_{j \in \mathbb{N}}$ is an orthonormal basis of $\mathcal{Y}$ (which exists since $\mathcal{Y}$ is separable).

7. Experiments

In this experimental section, we essentially aim at illustrating the potential of adopting a functional data analysis perspective for learning multi-output functions when the data are curves. First, we are interested in the problem of acoustic-to-articulatory speech inversion where the goal is to learn vocal tract (VT) time functions from the acoustic speech signal (Mitra et al., 2010). Then we show, through experiments on sound recognition (Rabaoui et al., 2008), that the proposed framework can be applied beyond functional response regression, for problems like multiple functional classification where each sound to be classified is represented by more than one functional parameters.

8. The operator K_x from $\mathcal{Y}$ to $\mathcal{F}$, defined by $y \mapsto K(x, \cdot)y$, $\forall y \in \mathcal{Y}$, is a Hilbert-Schmidt operator if, for some any basis $(y_j)_{j \in \mathbb{N}}$ of $\mathcal{Y}$, it holds that $Tr(K_x^* K_x) = \sum_j \langle K(x, \cdot)y_j, K(x, \cdot)y_j \rangle_{\mathcal{F}} < +\infty$. This is equivalent to saying that the operator $K(x, x) \in \mathcal{L}(\mathcal{Y})$ is of trace class, since by the reproducing property we have $\langle K(x, \cdot)y_j, K(x, \cdot)y_j \rangle = \langle K(x, x)y_j, y_j \rangle_{\mathcal{Y}}$.

The operator-valued kernel used in these experiments is the kernel K defined by Equation (11). We use the inner product in $\mathcal{X}^p$ for the scalar-valued kernel g, where p is the number of functional parameters of a speech or a sound signal. Also, extending real-valued functional kernel, as in Rossi and Villa (2006), to multiple functional inputs could be possible. Eigenvalues δ_i and eigenfunctions w_i of the Hilbert-Schmidt integral operator T associated with the operator-valued kernel K are equal to $\frac{2}{1+\mu_i^2}$ and $\mu_i \cos(\mu_i x) + \sin(\mu_i x)$ respectively, where μ_i are solutions of the equation $\cot \mu = \frac{1}{2}(\mu - \frac{1}{\mu})$. Eigendecomposition of an infinite dimensional operator T is computed in general by solving a differential equation obtained from the equality $Tw_i = \delta_i w_i$.

In order to choose the regularization parameter λ and the number of eigenfunctions κ that guarantee optimal solutions, one may use the cross-validation score based on the one-curve-leave-out prediction error (Rice and Silverman, 1991). Then we choose λ and κ so as to minimize the cross-validation score based on the squared prediction error

$$CV(\lambda) = \sum_{i=1}^{n} \sum_{j=1}^{N_i} \{y_{ij} - \hat{y}_i^{(-i)}(t_{ij})\}^2, \tag{13}$$

where n is the number of functions y_i, y_{ij} is the observed value at time t_{ij}, N_i the number of measurements made on y_i and $\hat{y}_i^{(-i)}$ the predicted curve for the i^{th} function, computed after removing the data for this function.

7.1 Speech Inversion

The problem of speech inversion has received increasing attention in the speech processing community in the recent years (see Schroeter and Sondhi (1994); Mitra et al. (2010); Kadri et al. (2011a) and references therein). This problem, aka acoustic-articulatory inversion, involves inverting the forward process of speech production (see Figure 4). In other words, for a given acoustic speech signal we aim at estimating the underlying sequence of articulatory configurations which produced it (Richmond, 2002). Speech inversion is motivated by several applications in which it is required to estimate articulatory parameters from the acoustic speech signal. For example, in speech recognition, the use of articulatory information has been of interest since speech recognition efficiency can be significantly improved (Kirchoff, 1999). This is due to the fact that automatic speech recognition (ASR) systems suffer from performance degradation in the presence of noise and spontaneous speech. Moreover, acoustic-to-articulatory speech inversion is also useful in many other interesting applications such as speech analysis and synthesis (Toda et al., 2004) or helping individuals with speech and hearing disorders by providing visual feedback (Toutios and Margaritis, 2005).

Most of current research on acoustic-to-articulatory inversion focuses on learning Electromagnetic Articulography (EMA) trajectories from acoustic parameters and frequently uses the MOCHA `fsew0` data set as training and test data (Richmond, 2002). In a recent work, Mitra et al. (2010) suggest the use of the TAsk Dynamics Application (TADA) model (Nam et al., 2004) to generate acoustic-articulatory database which contains synthetic speech and the corresponding vocal tract time functions. Their results show that tract variables can be better candidates than EMA trajectories for articulatory feature based ASR systems. In our experiments, we follow this work by addressing the issue of

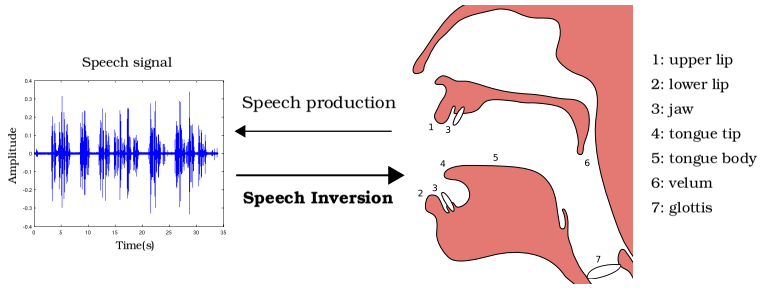


Figure 4: Principle of speech inversion (a.k.a. acoustic-articulatory inversion). Human beings produce an audible speech signal by moving their articulators (*e.g.* tongue, lips, velum, etc.) to modify a source of sound energy in the vocal tract. In performing the inversion mapping, we aim to invert this forward direction of speech production. In other words, we aim to take a speech signal and estimate the underlying articulatory movements which are likely to have created it (Richmond, 2002).

finding the mapping between acoustic parameters and vocal tract variables. In this context, we use Mel-Frequency Cepstral Coefficients (MFCCs) as input and consider as output eight different vocal tract constriction variables, lip aperture (LA), lip protrusion (LP), tongue tip constriction degree (TTCD), tongue tip constriction location (TTCL), tongue body constriction degree (TBCD), tongue body constriction location (TBCL), Velum (VEL) and Glottis (GLO). Table 2 shows the eight vocal tract variables we used in this study and the corresponding constriction organs and articulators (Mitra et al., 2009).

Moreover, articulators move relatively slowly and smoothly, and their movements are continuous. Indeed, the mouth cannot "jump" from one configuration to a completely different one (Richmond, 2002). For this reason, functional data analysis approaches are well suited for the speech inversion task. In other words, even if the measurement process itself is discrete, vocal tract variables are really smooth functions (see Figure 5) rather than vectors and taking into account such prior knowledge on the nature of the data can significantly improve performance. In our proposed method, smoothness is guaranteed by the use of smooth eigenfunctions obtained from the spectral decomposition of the integral operator associated with a Mercer kernel used to construct the operator-valued kernel defined in Equation 11. By this way, our approach does not need the filtering post-processing step, which is necessary in vectorial vocal-tract learning methods to transform the predicted functions on smooth curves and which has the drawback of changing the behavior of the predicted vocal tract time functions.

Various nonlinear acoustic-to-articulatory inversion techniques (Richmond, 2002; Mitra et al., 2010), and particularly kernel-based methods (Toutios and Margaritis, 2005; Mitra

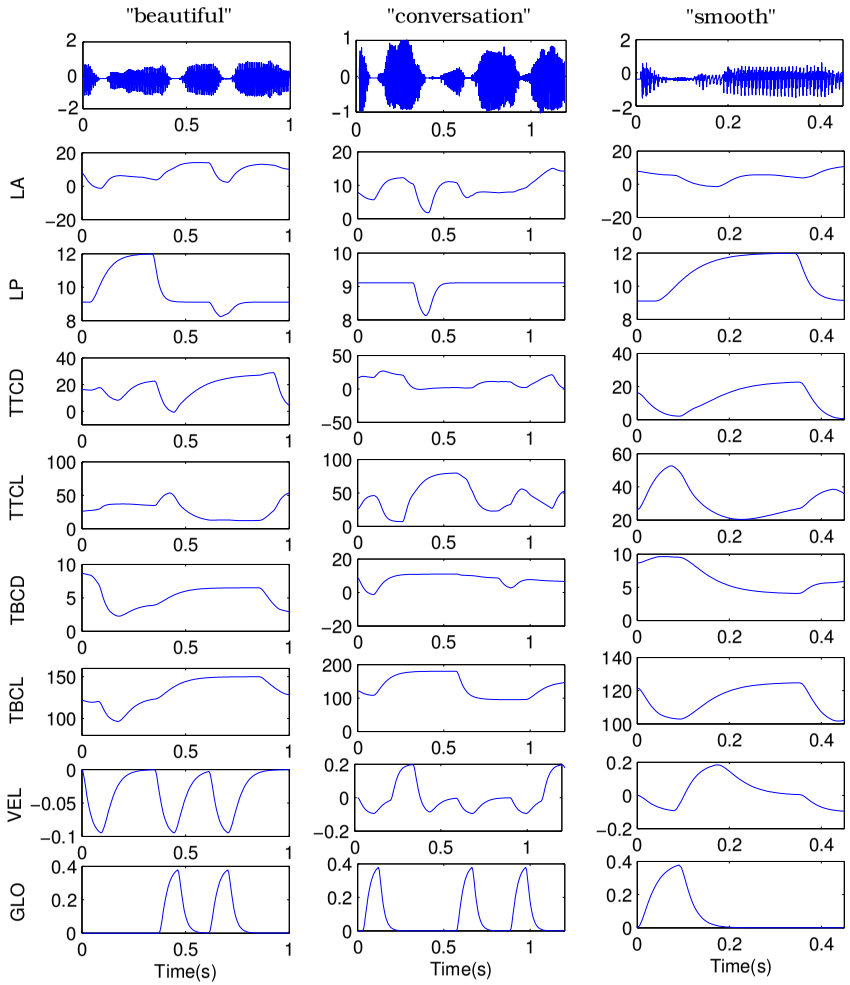


Figure 5: Acoustic waveforms and derived vocal tract time functions for the utterances "beautiful", "conversation" and "smooth". The vocal tract variables are: lip aperture (LA), lip protrusion (LP), tongue tip constriction degree (TTCD), tongue tip constriction location (TTCL), tongue body constriction degree (TBCD), tongue body constriction location (TBCL), Velum (VEL) and Glottis (GLO).

Constriction organ	VT variables	Articulators
lip	lip aperture (LA)	upper lip, lower lip, jaw
	lip protrusion (LP)	
tongue tip	tongue tip constriction degree (TTCD)	tongue body, tip, jaw
	tongue tip constriction location (TTCL)	
tongue body	tongue body constriction degree (TBCD)	tongue body, jaw
	tongue body constriction location (TBCL)	
velum	velum (VEL)	velum
glottis	glottis (GLO)	glottis

Table 2: Constriction organ, vocal-tract (VT) variables and involved articulators (Mitra et al., 2009).

et al., 2009), have been proposed in the literature. In most cases, these works address the articulatory estimation problem within a single-task learning perspective. However, in Richmond (2007) and more recently in Kadri et al. (2011a), the authors put forward the idea that we can benefit from viewing the acoustic-articulatory inversion problem from a multi-task learning perspective. Motivated by comparing our functional operator-valued kernel based approach with multivariate kernel methods, we report on experiments similar to those performed by Mitra et al. (2009) and Kadri et al. (2011a). The tract variables learning technique proposed by Mitra et al. (2009) is based on a hierarchical ε-SVR architecture constructed by associating different SVRs, a SVR for each tract variable. To consider the dependencies between VT time functions, the SVRs corresponding to independent VT variables are first created and then used for constructing the others. Otherwise, the acoustic-to-articulatory method in Kadri et al. (2011a) is based on learning a vector-valued function using a matrix-valued kernel proposed in Caponnetto et al. (2008).

Following Mitra et al. (2010), acoustic-articulatory database is generated by the TADA model (Nam et al., 2004) which is a computational implementation of articulatory phonology. The generated data set consists of acoustic signals for 416 words chosen from the Wisconsin X-ray microbeam data (Westbury et al., 1994) and corresponding Vocal Tract (VT) trajectories sampled at 5 ms. The speech signal was parameterized into 13 Mel-Frequency Cepstral Coefficients. These cepstral coefficients were acquired each 5 ms (synchronized with the TVs) with window duration of 10 ms.

For evaluating the performance of the VT time functions estimation, we use the residual sum of squares error (RSSE) defined as follows

$$RSSE = \int \sum_i \{y_i(t) - \widehat{y}_i(t)\}^2 dt, \tag{14}$$

VT variables	ε-SVR	Multi-task	Functional
LA	2.763	2.341	**1.562**
LP	0.532	**0.512**	0.528
TTCD	3.345	1.975	**1.647**
TTCL	7.752	5.276	**3.463**
TBCD	2.155	2.094	**1.582**
TBCL	15.083	9.763	**7.215**
VEL	0.032	0.034	**0.029**
GLO	**0.041**	0.052	0.064
Total	3.962	2.755	**2.011**

Table 3: Average RSSE for the tract variables using hierarchical ε-SVR (Mitra et al., 2009), the multi-task kernel method (Kadri et al., 2011a) and the proposed functional operator-valued kernel based approach.

where $\widehat{y}_i(t)$ is the prediction of the VT curve $y_i(t)$. Table 3 reports average RSSE results obtained using the hierarchical ε-SVR algorithm (Mitra et al., 2009), the multi-task kernel method (Kadri et al., 2011a) after smoothing the estimated VT trajectories using a Kalman filter as described in Mitra et al. (2009), and the functional operator-valued kernel based approach. The proposed functional approach consistently produced significant performance improvements over the supervised baseline ε-SVR. It also outperforms the discrete multi-task method (Evgeniou et al., 2005; Kadri et al., 2011a) except for the LP and GLO variables. The multi-task and also the ε-SVR methods perform well for these two vocal tract variables and slightly improve our functional approach. This can be explained by the fact that, contrary to other vocal tract variables, LP and GLO time functions are not completely smooth for all times and positions, while our method with the integral operator-valued kernel, as defined in Equation 11, tends to favor the prediction of smooth functions. Building operator-valued kernels suitable for heterogeneous functions, i.e., smooth in some parts and non-smooth in others, could be a good alternative to improve the prediction of these two vocal tract time functions. Note that the number of eigenfunctions κ affects performance. κ has to be well chosen to provide a reasonable approximation of the infinite-dimensional process. In the case of complex output functions, like heterogeneous functions, we need to use many eigenfunctions to have a good approximation, but even for this case, κ remains (very) small compared to the number of examples n.

7.2 Sound Recognition

A second application of the ideas that we present in this paper is sound recognition. Many previous works in the context of sound recognition problem have concentrated on classifying environmental sounds other than speech and music (Dufaux et al., 2000; Peltonen et al., 2002). Such sounds are extremely versatile, including signals generated in domestic, business, and outdoor environments. A system that is able to recognize such sounds may be of great importance for surveillance and security applications (Istrate et al., 2006; Rabaoui et al., 2008). The classification of a sound is usually performed in two steps. First, a pre-

Classes	Number	Train	Test	Total	Duration (s)
Human screams	C1	40	25	65	167
Gunshots	C2	36	19	55	97
Glass breaking	C3	48	25	73	123
Explosions	C4	41	21	62	180
Door slams	C5	50	25	75	96
Phone rings	C6	34	17	51	107
Children voices	C7	58	29	87	140
Machines	C8	40	20	60	184
Total		327	181	508	18mn 14s

Table 4: Classes of sounds and number of samples in the database used for performance evaluation.

processor applies signal processing techniques to generate a set of features characterizing the signal to be classified. Then, in the feature space, a decision rule is implemented to assign a class to a pattern.

Operator-valued kernels can be used in a classification setting by considering the labels y_i to be functions in some function space rather than real values. Similarly to the scalar case, a natural choice for y_i would seem to be the Heaviside step function in $L^2([0,1])$ scaled by a real number. In this context, our method can be viewed as an extension of the Regularized Least Squares Classification (RLSC) algorithm (Rifkin et al., 2003) to the FDA domain (we called it Functional RLSC (Kadri et al., 2011c)). The performance of the proposed algorithm described in Section 6 is evaluated on a data set of sounds collected from commercial databases which include sounds ranging from screams to explosions, such as gun shots or glass breaking, and compared with the RLSC method.

7.2.1 Database Description

As in Rabaoui et al. (2008), the major part of the sound samples used in the recognition experiments is taken from two sound libraries (Leonardo Software; Real World Computing Paternship, 2000). All signals in the database have a 16 bits resolution and are sampled at 44100 Hz, enabling both good time resolution and a wide frequency band, which are both necessary to cover harmonic as well as impulsive sounds. The selected sound classes are given in Table 4, and they are typical of surveillance applications. The number of items in each class is deliberately not equal.

Note that this database includes impulsive and harmonic sounds such as phone rings (C6) and children voices (C7). These sounds are quite likely to be recorded by a surveillance system. Some sounds are very similar to a human listener: in particular, explosions (C4) are pretty similar to gunshots (C2). Glass breaking sounds include both bottle breaking and window breaking situations. Phone rings are either electronic or mechanic alarms.

Temporal representations and spectrograms of some sounds are depicted in Figures 6 and 7. Power spectra are extracted through the Fast Fourier Transform (FFT) every 10 ms from 25 ms frames. They are represented vertically at the corresponding frame indexes.

The frequency range of interest is between 0 and 22 kHz. A lighter shade indicates a higher power value. These figures show that in the considered database we can have both: (1) many similarities between sounds belonging to different classes, (2) diversities within the same class of sounds.

7.2.2 Sound Classification Results

Following Rifkin and Klautau (2004), the 1-vs-all multi-class classifier is selected in these experiments. So we train N (number of classes) different binary classifiers, each one trained to distinguish the data in a single class from the examples in all remaining classes. We run the N classifiers to classify a new example.

The adopted sound data processing scheme is the following. Let $\mathcal{X}$ be the set of training sounds, shared in N classes denoted $\mathcal{C}_1, \ldots, \mathcal{C}_N$. Each class contains m_i sounds, $i = 1, \ldots, N$. Sound number j in class $\mathcal{C}_i$ is denoted $\mathbf{s}_{i,j}$, $(i = 1, \ldots, N, j = 1, \ldots, m_i)$. The pre-processor converts a recorded acoustic signal $\mathbf{s}_{i,j}$ into a time/frequency localized representation. In multivariate methods, this representation is obtained by splitting the signal $\mathbf{s}_{i,j}$ into $T_{i,j}$ overlapping short frames and computing a vector of features $z_{t,i,j}$, $t = 1, \ldots, T_{i,j}$ which characterize each frame. Since the pre-processor is a series of continuous time-localized features, it will be useful to take into account the relationships between feature samples along the time axis and consider dependencies between features. That is why we use a FDA-based approach in which features representing a sound are modeled by functions $z_{i,j}(t)$. In this work, Mel Frequency Cepstral Coefficients (MFCCs) features are used to describe the spectral shape of each signal. These coefficients are obtained using 23 channels Mel filterbank and a Hamming analysis window of length 25 ms with 50% overlap. We also use the energy parameter measured for each window along all the sound signal. So, each sound is characterized by 14 functional parameters: 13 cepstral functions and 1 energy function.

Performance of the proposed functional approach in the context of classification is compared to the results obtained by the RLSC algorithm, see Tables 5 and 6. The performance is measured as the percentage number of sounds correctly recognized and it is given by $(\mathrm{W_r}/\mathrm{T_n}) \times 100\%$, where $\mathrm{W_r}$ is the number of well recognized sounds and $\mathrm{T_n}$ is the total number of sounds to be recognized. The use of the Functional RLSC is fully justified by the results presented here, as it yields consistently a high classification accuracy for the major part of the sound classes.

RLSC setup is similar to that of FLRSC. The major difference is in the modeling of sound features. In RLSC, all the functional parameters which characterize a sound are combined in the same vector which is considered to be in $\mathbb{R}^d$. Functional RLSC considers each input sound as a vector of functions in $(L^2([0,1]))^p$ where p is the number of functional parameters. By using operator-valued kernels rather than scalar-valued ones, we project these functional data into a higher dimensional feature space in which we define a distance measure from the spectral decomposition of the operator-valued kernel, suitable for functions and which allows the learning module to take into account the sequential nature of the data and the dependencies along the time-axis. Moreover, compared to RLSC, κ the number of eigenfunctions in FRLSC can be seen as one more degree of freedom which can be used to improve performance when input data are complex and not represented by a vector in $\mathbb{R}^d$ as usual. Note that the usual scalar case (output space is $\mathbb{R}$ and then $\kappa = 1$) can be recovered

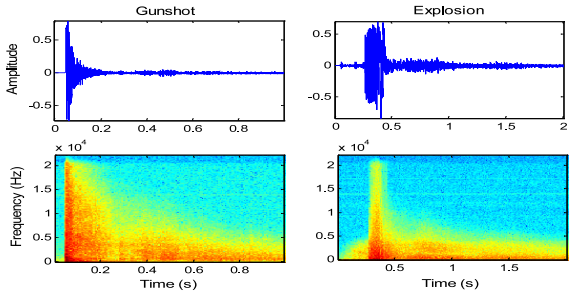


Figure 6: Structural similarities between two different classes. Gunshot and Explosion are two sound signals belonging to two different classes, but they have similar temporal and spectral representations.

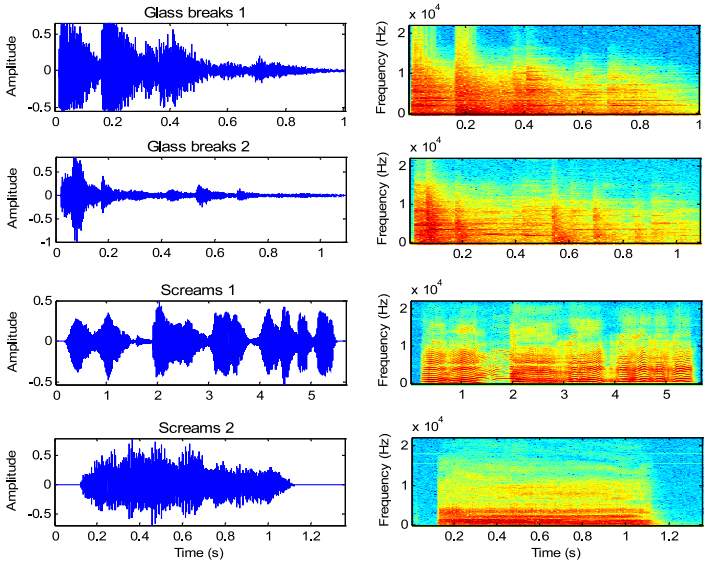


Figure 7: Structural diversity inside the same sound class and between classes. Glass breaks 1 and 2 (resp. Screams 1 and 2) are two sounds from the same class, however they present different temporal (resp. spectral) properties.

	C1	C2	C3	C4	C5	C6	C7	C8
C1	92	4	4.76	0	5.27	11.3	6.89	0
C2	0	52	0	14	0	2.7	0	0
C3	0	20	76.2	0	0	0	17.24	5
C4	0	16	0	66	0	0	0	0
C5	4	8	0	4	84.21	0	6.8	0
C6	4	0	0	0	10.52	86	0	0
C7	0	0	0	8	0	0	69.07	0
C8	0	0	19.04	8	0	0	0	95
Total Recognition Rate = 77.56%								

Table 5: Confusion Matrix obtained when using the Regularized Least Squares Classification (RLSC) algorithm.

	C1	C2	C3	C4	C5	C6	C7	C8
C1	**100**	0	0	2	0	5.3	3.4	0
C2	0	**82**	0	8	0	0	0	0
C3	0	14	**90.9**	8	0	0	3.4	0
C4	0	4	0	**78**	0	0	0	0
C5	0	0	0	1	**89.47**	0	6.8	0
C6	0	0	0	0	10.53	**94.7**	0	0
C7	0	0	0	0	0	0	**86.4**	0
C8	0	0	9.1	3	0	0	0	**100**
*Total Recognition Rate = **90.18%***								

Table 6: Confusion Matrix obtained when using the proposed Functional Regularized Least Squares Classification (FRLSC) algorithm.

from the functional case; for example when the operator-valued kernel is constructed from the identity operator or/and the output space is the space of constant functions.

8. Conclusion

We have presented a learning methodology for nonlinear functional data analysis, which is an extension of scalar-valued and matrix-valued kernel based methodologies to the functional response setting. The problem of functional supervised learning is formalized as the problem of learning an operator between two infinite dimensional scalar-valued Hilbert spaces in a reproducing kernel Hilbert space of function-valued functions. We have introduced a set of rigorously defined operator-valued kernels that can be valuably applied to nonparametric operator learning when input and output data are continuous smooth functions, and we have

showed their use for solving the problem of minimizing a regularized risk functional in the case of functional outputs without the need to discretize covariate and target functions. Our fully functional approach has been successfully applied to the problems of speech inversion and sound recognition, showing that the proposed framework is particularly relevant for audio signal processing applications where attributes are functions and dependent of each other.

In future work, it would be interesting to explore further the potential of our proposed method in other machine learning problems such as collaborative filtering (Abernethy et al., 2009) and structured output prediction (Brouard et al., 2011; Kadri et al., 2013b) by building operator-valued kernels that can capture not only the functional information of responses, but also other types of output structure. In this context, learning the operator-valued kernel would be interesting to find the right model of dependencies between outputs. Recent works in this direction includes the papers of Dinuzzo et al. (2011), Kadri et al. (2012), Sindhwani et al. (2013) and Lim et al. (2015), but further investigations are needed in this area. On the algorithmic side, possible extensions of this work include on-line implementations to deal with the case where the functional data set is made available step by step (Audiffren and Kadri, 2015). Learning, sequentially and without re-training from scratch at each iteration, a new function-valued function for each new observed pair of functional samples would be of practical interest. Finally, although not covered in this paper, the analysis we present is likely to be applicable to learning problems that involve functional data with different functional profiles (*e.g.*, both smooth and spiky functions). Designing nonseparable operator-valued kernels that can exhibit better ability to characterize different smoothing levels is also an interesting future research direction.

Acknowledgments

We thank the anonymous reviewers for several insightful comments that helped improve the original version of this paper. We also thank M. Mbekhta for fruitful discussions on the spectral theory of block operator matrices and A. Rabaoui for providing the sound recognition data set. A large part of this research was done while H.K. was at SequeL (INRIA-Lille) and J.A. at Aix-Marseille Université and LIF. This work was supported by Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council and FEDER through the 'Contrat de Projets Etat Region (CPER) 2007-2013'. H.K. acknowledges the support of a Junior Researcher Contract No. 4297 from the Nord-Pas-de-Calais region. H.K. and P.P. acknowledge the support of the French National Research Agency (ANR-09-EMER-007 project LAMPADA). H.K. was also supported in part by French grants from the CNRS-LAGIS (ANR KernSig Project). A.R. was supported by the PASCAL2 Network of Excellence, ICT-216886, ANR Project ASAP ANR-09-EMER-001 and the INRIA ARC MABI. P.P. also acknowledges the support of INRIA.

Appendix A. Proof of Theorem 2 - Completion of $\mathcal{F}_0$

We show below how to construct the Hilbert space $\mathcal{F}$ of $\mathcal{Y}$-valued functions, that is the completion of the function-valued pre-Hilbert space $\mathcal{F}_0$. $\mathcal{F}_0 \subset \mathcal{Y}^{\mathcal{X}}$ is the space of all $\mathcal{Y}$-

valued functions F of the form $F(\cdot) = \sum_{i=1}^{n} K(w_i, \cdot)u_i$, where $w_i \in \mathcal{X}$ and $u_i \in \mathcal{Y}$, $i = 1, \ldots, n$. Consider the inner product of the functions $F(\cdot) = \sum_{i=1}^{n} K(w_i, \cdot)u_i$ and $G(\cdot) = \sum_{j=1}^{m} K(z_j, \cdot)v_j$ from $\mathcal{F}_0$ defined as follows

$$\langle F(\cdot), G(\cdot)\rangle_{\mathcal{F}_0} = \langle \sum_{i=1}^{n} K(w_i, \cdot)u_i, \sum_{j=1}^{m} K(z_j, \cdot)v_j\rangle_{\mathcal{F}_0} = \sum_{i=1}^{n}\sum_{j=1}^{m} \langle K(w_i, z_j)u_i, v_j\rangle_{\mathcal{Y}}.$$

We have shown that $(\mathcal{F}_0, \langle ., .\rangle_{\mathcal{F}_0})$ is a pre-Hilbert space. This pre-Hilbert space is in general not complete, but It can be completed via Cauchy sequences to build the $\mathcal{Y}$-valued reproducing kernel Hilbert space $\mathcal{F}$.

Consider any Cauchy sequence $\{F_n(\cdot)\} \subset \mathcal{F}_0$, for every $w \in \mathcal{X}$ the functional $F(w)$ is bounded, since

$$\begin{aligned}
\|F(w)\|_{\mathcal{Y}} &= \sup_{\|u\|=1} |\langle F(w), u\rangle_{\mathcal{Y}}| = \sup_{\|u\|=1} |\langle F(\cdot), K(w, \cdot)u\rangle_{\mathcal{F}_0}| \\
&\leq \|F(\cdot)\|_{\mathcal{F}_0} \sup_{\|u\|=1} \|K(w, \cdot)u\|_{\mathcal{F}_0} \leq \|F(\cdot)\|_{\mathcal{F}_0} \sup_{\|u\|=1} \sqrt{\langle K(w, w)u, u\rangle_{\mathcal{Y}}} \\
&\leq M_w \|F(\cdot)\|_{\mathcal{F}_0} \text{ with } M_w = \sup_{\|u\|=1} \sqrt{\langle K(w, w)u, u\rangle_{\mathcal{Y}}}.
\end{aligned}$$

Moreover, if the kernel K is Mercer, it is locally bounded (see Carmeli et al., 2010, Proposition 2). It is easy to see that in this case $\|F(w)\|_{\mathcal{Y}} \leq M\|F(\cdot)\|_{\mathcal{F}_0}$, where M here does not depend on w. Consequently,

$$\|F_n(w) - F_m(w)\|_{\mathcal{Y}} \leq M\|F_n(\cdot) - F_m(\cdot)\|_{\mathcal{F}_0}.$$

It follows that $\{F_n(w)\}$ is a Cauchy sequence in $\mathcal{Y}$ and by the completeness of the space $\mathcal{Y}$, there exists a $\mathcal{Y}$-valued function F where, $\forall w \in \mathcal{X}$, $F(w) = \lim_{n\to\infty} F_n(w)$. So the Cauchy sequence $\{F_n(\cdot)\}$ defines a function $F(\cdot)$ to which it is convergent at every point of $\mathcal{X}$.

Let us denote $\mathcal{F}$ the linear space containing all the functions $F(\cdot)$, the limits of Cauchy sequences $\{F_n(\cdot)\} \subset \mathcal{F}_0$, and consider the norm in $\mathcal{F}$ defined by $\|F(\cdot)\|_{\mathcal{F}} = \lim_{n\to\infty} \|F_n(\cdot)\|_{\mathcal{F}_0}$, where $F_n(\cdot)$ is a Cauchy sequence of $\mathcal{F}_0$ converging to $F(\cdot)$. This norm is well defined since it does not depend on the choice of the Cauchy sequence. In fact, suppose that two Cauchy sequences $\{F_n(\cdot)\}$ and $\{G_n(\cdot)\}$ in $\mathcal{F}_0$ define the same function $F(\cdot) \in \mathcal{F}$. Then $\{F_n(\cdot) - G_n(\cdot)\}$ is also a Cauchy sequence and $\forall w \in \mathcal{X}$, $\lim_{n\to\infty} F_n(w) - G_n(w) = 0$. Hence, $\lim_{n\to\infty} \langle F_n(w) - G_n(w), u\rangle_{\mathcal{Y}} = 0$ for any $u \in \mathcal{G}_y$ and using the reproducing property, it follows that $\lim_{n\to\infty} \langle F_n(\cdot) - G_n(\cdot), H(\cdot)\rangle_{\mathcal{F}_0} = 0$ for any function $H(\cdot) \in \mathcal{F}_0$ and thus

$$\lim_{n\to\infty} \|F_n(\cdot) - G_n(\cdot)\|_{\mathcal{F}_0} = 0.$$

Consequently,

$$|\lim_{n\to\infty} \|F_n\| - \lim_{n\to\infty} \|G_n\| | = \lim_{n\to\infty} | \|F_n\| - \|G_n\| | \leq \lim_{n\to\infty} \|F_n - G_n\| = 0.$$

So that for any function $F(\cdot) \in \mathcal{F}$ defined by two different Cauchy sequences $\{F_n(\cdot)\}$ and $\{G_n(\cdot)\}$ in $\mathcal{F}_0$, we have $\lim_{n\to\infty} \|F_n(\cdot)\|_{\mathcal{F}_0} = \lim_{n\to\infty} \|G_n(\cdot)\|_{\mathcal{F}_0} = \|F(\cdot)\|_{\mathcal{F}}$. $\|.\|_{\mathcal{F}}$ has all the

properties of a norm and defines in $\mathcal{F}$ an inner product which on $\mathcal{F}_0$ coincides with $\langle ., .\rangle_{\mathcal{F}_0}$ already defined. It remains to be shown that $\mathcal{F}_0$ is dense in $\mathcal{F}$ which is a complete space.

For any $F(\cdot)$ in $\mathcal{F}$ defined by the Cauchy sequence $F_n(\cdot)$, we have $\lim_{n\to\infty} \|F(\cdot) - F_n(\cdot)\|_{\mathcal{F}} = \lim_{n\to\infty} \lim_{m\to\infty} \|F_m(\cdot) - F_n(\cdot)\|_{\mathcal{F}_0} = 0$. It follows that $F(\cdot)$ is a strong limit of $F_n(\cdot)$ in $\mathcal{F}$ and then $\mathcal{F}_0$ is dense in $\mathcal{F}$. To prove that $\mathcal{F}$ is a complete space, we consider $\{F_n(\cdot)\}$ any Cauchy sequence in $\mathcal{F}$. Since $\mathcal{F}_0$ is dense in $\mathcal{F}$, there exists a sequence $\{G_n(\cdot)\} \subset \mathcal{F}_0$ such that $\lim_{n\to\infty} \|G_n(\cdot) - F_n(\cdot)\|_{\mathcal{F}} = 0$. Besides $\{G_n(\cdot)\}$ is a Cauchy sequence in $\mathcal{F}_0$ and thus defines a function $H(\cdot) \in \mathcal{F}$ which verifies $\lim_{n\to\infty} \|G_n(\cdot) - h(\cdot)\|_{\mathcal{F}} = 0$. So $\{G_n(\cdot)\}$ converges strongly to $H(\cdot)$ and then $\{F_n(\cdot)\}$ also converges strongly to $H(\cdot)$ which means that the space $\mathcal{F}$ is complete. In addition, $K(\cdot, \cdot)$ has the reproducing property in $\mathcal{F}$. To see this, let $F(\cdot) \in \mathcal{F}$, then $F(\cdot)$ is defined by a Cauchy sequence $\{F_n(\cdot)\} \subset \mathcal{F}_0$ and we have from the continuity of the inner product in $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$ (endowed with the uniform topology) that, for all $w \in \mathcal{X}$ and $u \in \mathcal{Y}$,

$$\begin{aligned}
\langle F(w), u\rangle_{\mathcal{Y}} &= \langle \lim_{n\to\infty} F_n(w), u\rangle_{\mathcal{Y}} = \lim_{n\to\infty} \langle F_n(w), u\rangle_{\mathcal{Y}} = \lim_{n\to\infty} \langle F_n(\cdot), K(w, \cdot)u\rangle_{\mathcal{F}_0} \\
&= \langle \lim_{n\to\infty} F_n(\cdot), K(w, \cdot)u\rangle_{\mathcal{F}} = \langle F(\cdot), K(w, \cdot)u\rangle_{\mathcal{F}}.
\end{aligned}$$

Finally, we conclude that $\mathcal{F}$ is a reproducing kernel Hilbert space since $\mathcal{F}$ is a real inner product space that is complete under the norm $\|.\|_{\mathcal{F}}$ defined above, and has $K(\cdot, \cdot)$ as reproducing kernel. ■

Appendix B. Representer Theorem

We provide here a proof of the analog of the representer theorem in the case of function-valued reproducing kernel Hilbert spaces.

Theorem 9 *(representer theorem)*
Let K a nonnegative Mercer operator-valued kernel and $\mathcal{F}$ its corresponding function-valued reproducing kernel Hilbert space. The solution $\widetilde{F}_\lambda \in \mathcal{F}$ of the regularized optimization problem

$$\widetilde{F}_\lambda = \arg\min_{F\in\mathcal{F}} \sum_{i=1}^{n} \|y_i - F(x_i)\|_{\mathcal{Y}}^2 + \lambda\|F\|_{\mathcal{F}}^2$$

has the following form

$$\widetilde{F}_\lambda(\cdot) = \sum_{i=1}^{n} K(x_i, \cdot)u_i,$$

where $u_i \in \mathcal{Y}$.

Proof: We use the Frechet derivative which is the strongest notion of derivative in a normed linear space; see, for example, Chapter 4 of Kurdila and Zabarankin (2005). We use the standard notation D_F for the Frechet derivative operator. Let $J_\lambda(F) = \sum_{i=1}^{n} \|y_i - F(x_i)\|_{\mathcal{Y}}^2 + \lambda\|F\|_{\mathcal{F}}^2$ be the functional to be minimized. $\widetilde{F}$ is the operator in $\mathcal{F}$ such that

$\widetilde{F} = \arg\min_{F\in\mathcal{F}} J_\lambda(F) \Rightarrow D_F J_\lambda(\widetilde{F}) = 0$. To compute $D_F J_\lambda(F)$, we use the Gateaux derivative D_G of J_λ with respect to F in the direction H, which is defined by:

$$D_G J_\lambda(F,H) = \lim_{\tau\longrightarrow 0} \frac{J_\lambda(F+\tau H) - J_\lambda(F)}{\tau}.$$

J_λ can be written as $J_\lambda(F) = \sum_{i=1}^{n} G_i(F) + \lambda L(F)$ and using the fact that $D_G J_\lambda(F,H) = \langle D_G J_\lambda(F), H\rangle$ we obtain

i. $L(F) = \|F\|_{\mathcal{F}}^2$

$$\lim_{\tau\longrightarrow 0} \frac{\|F+\tau H\|_{\mathcal{F}}^2 - \|F\|_{\mathcal{F}}^2}{\tau} = 2\langle F, H\rangle \implies D_G L(F) = 2F.$$

ii. $G_i(F) = \|y_i - F(x_i)\|_{\mathcal{Y}}^2$

$$\lim_{\tau\longrightarrow 0} \frac{\|y_i - F(x_i) - \tau H(x_i)\|_{\mathcal{Y}}^2 - \|y_i - F(x_i)\|_{\mathcal{Y}}^2}{\tau} = -2\langle y_i - F(x_i), H(x_i)\rangle_{\mathcal{Y}}$$
$$= -2\langle K(x_i,\cdot)(y_i - F(x_i)), H\rangle_{\mathcal{F}} = -2\langle K(x_i,\cdot)u_i, H\rangle_{\mathcal{F}} \text{ with } u_i = y_i - F(x_i)$$
$$\implies D_G G_i(F) = -2K(x_i,\cdot)u_i.$$

When the kernel K is Mercer, Corollary 4.1.1 in Kurdila and Zabarankin (2005) can be applied to show that J_λ is Fréchet differentiable and that, $\forall F \in \mathcal{F}$, $D_F J_\lambda(F) = D_G J_\lambda(F)$. Using (i), (ii), and $D_F J_\lambda(\widetilde{F}) = 0 \implies \widetilde{F}(\cdot) = \frac{1}{\lambda}\sum_{i=1}^{n} K(x_i,\cdot)u_i$. ■

Appendix C. Proof of Theorem 7

We show here that under Assumptions 1, 2 and 3, a learning algorithm that maps a training set Z to the function F_Z defined in (12) is β stable with $\beta = \frac{\sigma^2\kappa^2}{2n\lambda}$. First, since ℓ is convex with respect to F, we have $\forall 0 \le t \le 1$

$$\ell(y, F_Z + t(F_{Z^i} - F_Z), x) - \ell(y, F_Z, x) \le t\left(\ell(y, F_{Z^i}, x) - \ell(y, F_Z, x)\right).$$

Then, by summing over all couples (x_k, y_k) in Z^i,

$$R_{emp}(F_Z + t(F_{Z^i} - F_Z), Z^i) - R_{emp}(F_Z, Z^i) \le t\left(R_{emp}(F_{Z^i}, Z^i) - R_{emp}(F_Z, Z^i)\right). \quad (15)$$

Symmetrically, we also have

$$R_{emp}(F_{Z^i} + t(F_Z - F_{Z^i}), Z^i) - R_{emp}(F_{Z^i}, Z^i) \le t\left(R_{emp}(F_Z, Z^i) - R_{emp}(F_{Z^i}, Z^i)\right). \quad (16)$$

Thus, by summing (15) and (16), we obtain

$$\begin{aligned} &R_{emp}(F_Z + t(F_{Z^i} - F_Z), Z^i) - R_{emp}(F_Z, Z^i) \\ &\quad + R_{emp}(F_{Z^i} + t(F_Z - F_{Z^i}), Z^i) - R_{emp}(F_{Z^i}, Z^i) \le 0. \end{aligned} \quad (17)$$

Now, by definition of F_Z and F_{Z^i},

$$\begin{aligned} &R_{reg}(F_Z, Z) - R_{reg}(F_Z + t(F_{Z^i} - F_Z), Z) \\ &\quad + R_{reg}(F_{Z^i}, Z^i) - R_{reg}(F_{Z^i} + t(F_Z - F_{Z^i}), Z^i) \le 0. \end{aligned} \quad (18)$$

Combining (17) and (18), we find

$$\begin{aligned} &\ell(y_i, F_Z, x_i) - \ell(y_i, F_Z + t(F_{Z^i} - F_Z), x_i) \\ &\quad + n\lambda\left(\|F_Z\|_{\mathcal{F}}^2 - \|F_Z + t(F_{Z^i} - F_Z)\|_{\mathcal{F}}^2 + \|F_{Z^i}\|_{\mathcal{F}}^2 - \|F_{Z^i} + t(F_Z - F_{Z^i})\|_{\mathcal{F}}^2\right) \le 0. \end{aligned} \quad (19)$$

Moreover, we have

$$\begin{aligned} &\|F_Z\|_{\mathcal{F}}^2 - \|F_Z + t(F_{Z^i} - F_Z)\|_{\mathcal{F}}^2 + \|F_{Z^i}\|_{\mathcal{F}}^2 - \|F_{Z^i} + t(F_Z - F_{Z^i})\|_{\mathcal{F}}^2 \\ &\quad = \|F_Z\|_{\mathcal{F}}^2 - \|F_Z\|_{\mathcal{F}}^2 - t^2\|F_{Z^i} - F_Z\|_{\mathcal{F}}^2 - 2t\langle F_Z, F_{Z^i} - F_z\rangle_{\mathcal{F}} \\ &\quad + \|F_{Z^i}\|_{\mathcal{F}}^2 - \|F_{Z^i}\|_{\mathcal{F}}^2 - t^2\|F_Z - F_{Z^i}\|_{\mathcal{F}}^2 - 2t\langle F_{Z^i}, F_Z - F_{Z^i}\rangle_{\mathcal{F}} \\ &\quad = -2t^2\|F_{Z^i} - F_Z\|_{\mathcal{F}}^2 - 2t\langle F_Z, F_{Z^i} - F_z\rangle_{\mathcal{F}} - 2t\langle F_{Z^i}, F_Z - F_{Z^i}\rangle_{\mathcal{F}} \\ &\quad = -2t^2\|F_{Z^i} - F_Z\|_{\mathcal{F}}^2 + 2t\|F_{Z^i} - F_Z\|_{\mathcal{F}}^2 \\ &\quad = 2t(1-t)\|F_{Z^i} - F_Z\|_{\mathcal{F}}^2. \end{aligned} \quad (20)$$

Hence, since ℓ is σ-Lipschitz continuous with respect to $F(x)$, we obtain from (19) and (20), $\forall t \in]0,1[$,

$$\begin{aligned} \|F_Z - F_{Z^i}\|_{\mathcal{F}}^2 &\le \frac{1}{2t(1-t)}\left(\|F_Z\|_{\mathcal{F}}^2 - \|F_Z + t(F_{Z^i} - F_Z)\|_{\mathcal{F}}^2 + \|F_{Z^i}\|_{\mathcal{F}}^2 - \|F_{Z^i} + t(F_Z - F_{Z^i})\|_{\mathcal{F}}^2\right) \\ &\le \frac{1}{2t(1-t)n\lambda}\left(\ell(y_i, F_Z + t(F_{Z^i} - F_Z), x_i) - \ell(y_i, F_Z, x_i)\right) \\ &\le \frac{\sigma}{2(1-t)n\lambda}\|F_{Z^i}(x_i) - F_Z(x_i)\|_{\mathcal{Y}}. \end{aligned}$$

In particular, when t tends to 0, we have

$$\|F_Z - F_{Z^i}\|_{\mathcal{F}}^2 \le \frac{\sigma}{2n\lambda}\|F_{Z^i}(x_i) - F_Z(x_i)\|_{\mathcal{Y}} \le \frac{\sigma\kappa}{2n\lambda}\|F_{Z^i} - F_Z\|_{\mathcal{F}},$$

which gives that

$$\|F_Z - F_{Z^i}\|_{\mathcal{F}} \le \frac{\sigma\kappa}{2n\lambda}.$$

This implies that, $\forall (x,y)$,

$$|\ell(y, F_Z, x) - \ell(y, F_{Z^i}, x)| \le \sigma\|F_Z(x) - F_{Z^i}(x)\|_{\mathcal{Y}} \le \sigma\kappa\|F_Z - F_{Z^i}\|_{\mathcal{F}} \le \frac{\sigma^2\kappa^2}{2n\lambda},$$

which concludes the proof. ■

Appendix D. Proof of Lemma 2

We show here that Assumption 4 is satisfied for the least squares loss function when Assumption 5 holds and use that to prove Lemma 2. First, note that ℓ is convex with respect to its second argument. Since $\mathcal{F}$ is a vector space, $0 \in \mathcal{H}$. Thus,

$$\lambda \|F_Z\|^2 \leq R_{reg}(F_Z, Z) \leq R_{reg}(0, Z) \leq \frac{1}{n}\sum_{k=1}^{n} \|y_k\|^2 \leq \sigma_y^2, \tag{21}$$

where we used the definition of F_Z (see Equation 12) and the bound on Y (Assumption 5). This inequality is uniform over Z, and thus holds for F_{Z^i}. Moreover, $\forall x \in \mathcal{X}$,

$$\|F_Z(x)\|_{\mathcal{Y}}^2 = \langle F_Z(x), F_Z(x)\rangle_{\mathcal{Y}} = \langle K(x,x)F_Z, F_Z\rangle_{\mathcal{F}} \leq \|K(x,x)\|_{op}\|F_Z\|_{\mathcal{F}}^2 \leq \kappa^2 \frac{\sigma_y^2}{\lambda}.$$

Hence, using Lemma 1 and (21), we obtain

$$\|y - F_Z(x)\|_{\mathcal{Y}} \leq \|y\|_{\mathcal{Y}} + \|f_Z(x)\|_{\mathcal{Y}} \leq \sigma_y + \kappa\frac{\sigma_y}{\sqrt{\lambda}},\ \forall (x,y) \in \mathcal{X}\times\mathcal{Y}.$$

Then it follows that

$$\begin{aligned}
&\left|\|y - F_Z(x)\|_{\mathcal{Y}}^2 - \|y - F_{Z^i}(x)\|_{\mathcal{Y}}^2\right| \\
&= \left|\|y - F_Z(x)\|_{\mathcal{Y}} - \|y - F_{Z^i}(x)\|_{\mathcal{Y}}\right| \left|\|y - F_Z(x)\|_{\mathcal{Y}} + \|y - F_{Z^i}(x)\|_{\mathcal{Y}}\right| \\
&\leq 2\sigma_y(1 + \frac{\kappa}{\sqrt{\lambda}})\|F_Z(x) - F_{Z^i}(x)\|_{\mathcal{Y}}.
\end{aligned}$$

■

Appendix E. Proof of Remark 1

We show here that if the finite trace assumption of the operator $K(x,x)$ in Caponnetto and De Vito (2006) is satisfied, then our Assumption 1 on the kernel holds. Let K be an operator-valued kernel satisfying the hypotheses of Caponnetto and De Vito (2006), i.e K_x is Hilbert-Schmidt and $\sup_{x\in\mathcal{X}} Tr(K(x,x)) < +\infty$. Then, $\exists \eta > 0$, $\forall x \in \mathcal{X}$, $\exists \left(e_j^x\right)_{j\in\mathbb{N}}$ an orthonormal basis of $\mathcal{Y}$, $\exists \left(h_j^x\right)_{j\in\mathbb{N}}$ an orthogonal family of $\mathcal{F}$ with $\sum_{j\in\mathbb{N}} \|h_j^x\|_{\mathcal{F}}^2 \leq \eta$ such that $\forall y \in \mathcal{Y}$,

$$K(x,x)y = \sum_{j,\ell} \left\langle h_j^x, h_\ell^x\right\rangle_{\mathcal{F}} \left\langle y, e_j^x\right\rangle_{\mathcal{Y}} e_\ell^x.$$

Thus, $\forall i \in \mathbb{N}$,

$$K(x,x)e_i^x = \sum_{j,\ell} \left\langle h_j^x, h_\ell^x\right\rangle_{\mathcal{F}} \left\langle e_i^x, e_j^x\right\rangle_{\mathcal{Y}} e_\ell^x = \sum_{\ell} \langle h_i^x, h_\ell^x\rangle_{\mathcal{F}}\, e_\ell^x.$$

Hence

$$\begin{aligned}
\|K(x,x)\|_{op}^2 &= \sup_{i\in\mathbb{N}} \|K(x,x)e_i^x\|_{\mathcal{Y}}^2 = \sup_{i\in\mathbb{N}} \sum_{j,\ell} \langle h_i^x, h_\ell^x\rangle_{\mathcal{F}} \left\langle h_i^x, h_j^x\right\rangle_{\mathcal{F}} \left\langle e_j^x, e_\ell^x\right\rangle_{\mathcal{Y}} \\
&= \sup_{i\in\mathbb{N}} \sum_{\ell} (\langle h_i^x, h_\ell^x\rangle_{\mathcal{F}})^2 \leq \sup_{i\in\mathbb{N}} \|h_i^x\|_{\mathcal{F}}^2 \sum_{\ell} \|h_\ell^x\|_{\mathcal{F}}^2 \leq \eta^2.
\end{aligned}$$

■

References

J. Abernethy, F. Bach, T. Evgeniou, and J. P. Vert. A new approach to collaborative filtering: operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826, 2009.

D. Alpay, A. Dijksma, J. Rovnyak, and H. de Snoo. *Schur Functions, Operator Colligations, and Reproducing Kernel Pontryagin Spaces*, volume 96 of *Operator theory: Advances and Applications*. Birkhäuser Verlag, 1997.

M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. *Foundation and Trends in Machine Learning*, 4(3):195–266, 2012.

R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

A. Asif and J. Moura. Block matrices with L-block-banded inverse: inversion algorithms. *IEEE Transactions on Signal Processing*, 53(2):630–642, February 2005.

J. Audiffren and H. Kadri. Stability of multi-task kernel regression algorithms. In *Asian Conference on Machine Learning (ACML)*, volume 29, pages 1–16, 2013.

J. Audiffren and H. Kadri. Online learning with operator-valued kernels. In *European symposium on artificial neural networks (ESANN)*, 2015.

L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. Multi-output learning via spectral filtering. *Machine Learning*, 87(3):259–301, 2012.

Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

S. Ben-david and R. Schuller-Borbely. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning*, 73:273–287, 2008.

P. Birkholz, B. J. Kröger, and C. Neuschaefer-Rube. Articulatory synthesis and perception of plosive-vowel syllables with virtual consonant targets. In *Interspeech*, pages 1017–1020. ISCA, 2010.

O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

L. Breiman and J. Friedman. Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society, Series B*, 59:3–54, 1997.

C. Brouard, F. d'Alché-Buc, and M. Szafranski. Semi-supervised penalized output kernel regression for link prediction. In *International Conference on Machine Learning (ICML)*, 2011.

S. Canu, X. Mary, and A. Rakotomamonjy. Functional learning through kernel. *in Advances in Learning Theory: Methods, Models and Applications. NATO Science Series III: Computer and Systems Sciences*, 2003.

A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2006.

A. Caponnetto, C. A. Micchelli, M. Pontil, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 68:1615–1646, 2008.

C. Carmeli, E. De Vito, and A. Toigo. Vector-valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem. *Analysis and Applications*, 4:377–408, 2006.

C. Carmeli, E. De Vito, and A. Toigo. Vector-valued reproducing kernel Hilbert spaces and universality. *Analysis and Applications*, 8:19–61, 2010.

J. M. Chiou, H. G. Müller, and J. L. Wang. Functional response models. *Statistica Sinica*, 14:675–693, 2004.

J. Dauxois, A. Pousse, and Y. Romain. Asymptotic theory for the principal component analysis of a vector random function: some applications to statistical inference. *Journal of Multivariate Analysis*, 12:136–154, 1982.

F. Dinuzzo, C. S. Ong, P. Gehler, and G. Pillonetto. Learning output kernels with block coordinate descent. In *International Conference on Machine Learning (ICML)*, 2011.

A. Dufaux, L. Besacier, M. Ansorge, and F. Pellandini. Automatic sound detection and recognition for noisy environment. In *European Signal Processing Conference (EUSIPCO)*, pages 1033–1036, 2000.

H. Dym. *J Contractive Matrix Functions, Reproducing Kernel Spaces and Interpolation*. American Mathematical Society, 1989.

T. Evgeniou and M. Pontil. Regularized multi-task learning. In *International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD)*, 2004.

T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

J. Faraway. Regression analysis for a functional response. *Technometrics*, 39(3):254–261, 1997.

F. Ferraty and P. Vieu. Curves discrimination: a nonparametric functional approach. *Computational Statistics & Data Analysis*, 44(1-2):161–173, 2003.

F. Ferraty and P. Vieu. Nonparametric models for functional data, with applications in regression, time series prediction and curves discrimination. *Journal of Nonparametric Statistics*, 16:111–125, 2004.

F. Ferraty and P. Vieu. *Nonparametric Functional Data Analysis*. Springer Verlag, 2006.

T. Hastie and R. Tibshirani. Varying-coefficient models. *Journal of the Royal Statistical Society B*, 55:757–796, 1993.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

T. C. Hesterberg, N. H. Choi, L. Meier, and C. Fraley. Least angle and ℓ_1 penalized regression: a review. *Statistics Surveys*, 2:61–93, 2008.

L. Horváth and P. Kokoszka. *Inference for Functional Data with Applications*. Springer, 2012.

D. Istrate, E. Castelli, M. Vacher, L. Besacier, and J. F. Serignat. Information extraction from sound for medical telemonitoring. *IEEE Transactions on Information Technology in Biomedicine*, 10:264–274, 2006.

G. James. Generalized linear models with functional predictors. *Journal of the Royal Statistical Society Series B*, 64(3):411–432, 2002.

T. Jebara. Multi-task feature and kernel selection for SVMs. In *International Conference on Machine Learning (ICML)*, 2004.

H. Kadri, E. Duflos, Ph. Preux, S. Canu, and M. Davy. Nonlinear functional regression: a functional RKHS approach. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 111–125, 2010.

H. Kadri, E. Duflos, and Ph. Preux. Learning vocal tract variables with multi-task kernels. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2011a.

H. Kadri, E. Duflos, Ph. Preux, and S. Canu. Multiple functional regression with both discrete and continuous covariates. In *International Workshop on Functional and Operatorial Statistics (IWFOS)*. Springer, 2011b.

H. Kadri, A. Rabaoui, Ph. Preux, E. Duflos, and A. Rakotomamonjy. Functional regularized least squares classification with operator-valued kernels. In *International Conference on Machine Learning (ICML)*, pages 993–1000, 2011c.

H. Kadri, A. Rakotomamonjy, F. Bach, and Ph. Preux. Multiple operator-valued kernel learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

H. Kadri, S. Ayache, C. Capponi, S. Koço, F. X. Dupé, and E. Morvant. The multi-task learning view of multimodal data. In *Asian Conference on Machine Learning (ACML)*, pages 261–276, 2013a.

H. Kadri, M. Ghavamzadeh, and Ph. Preux. A generalized kernel approach to structured output learning. In *International Conference on Machine Learning (ICML)*, 2013b.

K. Kirchoff. *Robust Speech Recognition Using Articulatory Information*. PhD thesis, University of Bielefeld, 1999.

A. Kurdila and M. Zabarankin. *Convex Functional Analysis*. Birkhauser Verlag, 2005.

Leonardo Software. http://www.leonardosoft.com.

D. J. Levitin, R. L. Nuzzo, B. W. Vines, and J. O. Ramsay. Introduction to functional data analysis. *Canadian Psychology*, 48(3):135–155, 2007.

H. Lian. Nonlinear functional models for functional responses in reproducing kernel Hilbert spaces. *The Canadian Journal of Statistics*, 35:597–606, 2007.

N. Lim, Y. Senbabaoglu, G. Michailidis, and F. d'Alché-Buc. OKVAR-Boost: a novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks. *Bioinformatics*, 29(11):1416–1423, 2013.

N. Lim, F. d'Alché-Buc, C. Auliac, and G. Michailidis. Operator-valued kernel-based vector autoregressive models for network inference. *Machine Learning*, 99(3):489–513, 2015.

A. Maurer. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:117–139, 2006.

A. Maurer and M. Pontil. Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory (COLT)*, pages 55–76, 2013.

C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17:177–204, 2005a.

C. A. Micchelli and M. Pontil. Kernels for multi–task learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 921–928, 2005b.

H. Q. Minh and V. Sindhwani. Vector-valued manifold regularization. In *International Conference on Machine Learning (ICML)*, 2011.

H. Q. Minh, S. H. Kang, and T. M. Le. Image and video colorization using vector-valued reproducing kernel Hilbert spaces. *Journal of Mathematical Imaging and Vision*, 37(1): 49–65, 2010.

H. Q. Minh, L. Bazzani, and V. Murino. A unifying framework for vector-valued manifold regularization and multi-view learning. In *International Conference on Machine Learning (ICML)*, 2013.

V. Mitra, Y. Ozbek, H. Nam, X. Zhou, and C. Y. Espy-Wilson. From acoustics to vocal tract time functions. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4497–4500, 2009.

V. Mitra, H. Nam, C. Espy-Wilson, E. Saltzman, and L. Goldstein. Retrieving tract variables from acoustics: a comparison of different machine learning strategies. *IEEE Journal of Selected Topics in Signal Processing*, pages 1027–1045, 2010.

K. Muandet, K. Fukumizu, F. Dinuzzo, and B. Schölkopf. Learning from distributions via support measure machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 10–18, 2012.

H. G. Müller. Functional modeling and classification of longitudinal data. *Scandinavian Journal of Statistics*, 32:223–240, 2005.

H. Nam, L. Goldstein, E. Saltzman, and D. Byrd. TADA: an enhanced, portable task dynamics model in MATLAB. *Journal of the Acoustical Society of America*, 115:2430, 2004.

A. W. Naylor and G. R. Sell. *Linear Operator Theory in Engineering and Science*. Holt, Rinehart and Winston, Inc., New York, 1971.

J. Oliva, B. Poczos, and J. Schneider. Distribution to distribution regression. In *International Conference on Machine Learning (ICML)*, 2013.

V. Peltonen, J. Tuomi, A. Klapuri, J. Huopaniemi, and T. Sorsa. Computational audiroty scene recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2002.

B. Poczos, L. Xiong, D. Sutherland, and J. Schneider. Support distribution machines. Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 2012.

B. Poczos, A. Rinaldo, A. Singh, and L. Wasserman. Distribution-free distribution regression. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 507–515, 2013.

L. Prchal and P. Sarda. Spline estimator for the functional linear regression with functional response. *Preprint*, 2007.

C. Preda. Regression models for functional data by reproducing kernel Hilbert spaces methods. *Journal of Statistical Planning and Inference*, 137:829–840, 2007.

A. Rabaoui, M. Davy, S. Rossignol, and N. Ellouze. Using one-class SVMs and wavelets for audio surveillance. *IEEE Transactions on Information Forensics and Security*, 3(4): 763–775, 2008.

J. O. Ramsay. When the data are functions. *Psychometrika*, 47:379–396, 1982.

J. O. Ramsay and J. L. Dalzell. Some tools for functional data analysis. *Journal of the Royal Statistical Society*, B(53):539–572, 1991.

J. O. Ramsay and B. W. Silverman. *Applied Functional Data Analysis*. Springer Verlag, New York, 2002.

J. O. Ramsay and B. W. Silverman. *Functional Data Analysis, 2nd ed.* Springer Verlag, New York, 2005.

Real World Computing Paternship. Cd-sound scene database in real acoustical environments, 2000. URL `http://tosa.mri.co.jp/sounddb/indexe.htm`.

M. Reisert and H. Burkhardt. Learning equivariant functions with matrix-valued kernels. *Journal of Machine Learning Research*, 8:385–408, 2007.

J. A. Rice. Functional and longitudinal data analysis: perspectives on smoothing. *Statistica Sinica*, 14:613–629, 2004.

J. A. Rice and B. W. Silverman. Estimating the mean and covariance structure nonparametrically when the data are curves. *Journal of the Royal Statistical Society. Series B*, 53(1):233–243, 1991.

K. Richmond. *Estimating Articulatory Parameters from the Acoustic Speech Signal*. PhD thesis, The Center for Speech Technology Research, Edinburgh University, 2002.

K. Richmond. A multitask learning perspective on acoustic-articulatory inversion. In *Interspeech*. ISCA, 2007.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

R. Rifkin, G. Yeo, and T. Poggio. Regularized least squares classification. *Advances in Learning Theory: Methods, Model and Applications NATO Science Series III: Computer and Systems Sciences*, 190:131–153, 2003.

F. Rossi and N. Villa. Support vector machine for functional data classification. *Neurocomputing*, 69(7–9):730–742, 2006.

W. Rudin. *Functional Analysis*. McGraw-Hill Science, 1991.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2002.

B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. R. Müller, G. Rätsch, and A. J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.

J. Schroeter and M. M. Sondhi. Techniques for estimating vocal-tract shapes from the speech signal. *IEEE Transactions on Speech and Audio Processing*, 2:133–150, 1994.

L. Schwartz. Sous-espaces hilbertiens d'espaces vectoriels topologiques et noyaux associés (noyaux reproduisants). *Journal d'Analyse Mathématique*, 13:115–256, 1964.

E. Senkene and A. Tempel'man. Hilbert spaces of operator-valued functions. *Lithuanian Mathematical Journal*, 1973.

J. Shawe-Taylor and N. Cristanini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

J. Q. Shi and T. Choi. *Gaussian Process Regression Analysis for Functional Data*. CRC Press, 2011.

T. Simila and J. Tikka. Input selection and shrinkage in multiresponse linear regression. *Computational Statistics and Data Analysis*, 52:406–422, 2007.

V. Sindhwani, H. Q. Minh, and A. C. Lozano. Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and granger causality. In *Uncertainty in Artificial Intelligence (UAI)*, 2013.

A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *Algorithmic Learning Theory (ALT)*, pages 13–31, 2007.

B. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: maximum margin regression; multiclass and multiview learning at one-class complexity. Technical report, PASCAL, Southampton, UK, 2006. URL `http://arxiv.org/pdf/1106.6251v1`.

T. Toda, A. W. Black, and K. Tokuda. Mapping from articulatory movements to vocal tract spectrum with gaussian mixture model for articulatory speech synthesis. In *ISCA Speech Synthesis Workshop*, 2004.

A. Toutios and K. Margaritis. A support vector approach to the acoustic-to-articulatory mapping. In *Interspeech*, pages 3221–3224. ISCA, 2005.

C. Tretter. *Spectral theory of block operator matrices and applications*. Imperial College Press, London, 2008.

B. A. Turlach, W. N. Venables, and S. J. Wright. Simultaneous variable selection. *Technometrics*, 47:349–363, 2005.

G. Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics (SIAM), 1990.

J. R. Westbury, G. Turner, and J. Dembovski. X-ray microbeam speech production database user's handbook, 1994.

F. Yao, H. G. Müller, and J. L. Wang. Functional linear regression analysis for longitudinal data. *Annals of Statistics*, 33:2873–2903, 2005.

H. Zhang, Y. Xu, and Q. Zhang. Refinement of operator-valued reproducing kernels. *Journal of Machine Learning Research*, 13:91–136, 2012.

X. Zhao, J. S. Marron, and M. T. Wells. The functional data analysis view of longitudinal data. *Statistica Sinica*, 14:789–808, 2004.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 5/12; Revised 6/15; Published 2/16

Meka: A Multi-label/Multi-target Extension to Weka

Jesse Read jesse.read@aalto.fi
Helsinki Institute for Information Technology (HIIT), and
Aalto University, Dept. of Computer Science, Espoo, Finland
Peter Reutemann fracpete@waikato.ac.nz
Bernhard Pfahringer bernhard@cs.waikato.ac.nz
Geoff Holmes geoff@cs.waikato.ac.nz
Department of Computer Science
University of Waikato, New Zealand

Editor: Balazs Kegl

Abstract

Multi-label classification has rapidly attracted interest in the machine learning literature, and there are now a large number and considerable variety of methods for this type of learning. We present Meka: an open-source Java framework based on the well-known Weka library. Meka provides interfaces to facilitate practical application, and a wealth of multi-label classifiers, evaluation metrics, and tools for multi-label experiments and development. It supports multi-label and multi-target data, including in incremental and semi-supervised contexts.

Keywords: classification, learning, multi-label, multi-target, incremental

1. Introduction

In *multi-label learning* a data instance may be associated with multiple binary class labels. This is as opposed to the traditional task of single-label (i.e., multi-class, or binary) classification where each instance is only associated with a single class label. The multi-label context is receiving a lot of attention and is relevant to a wide variety of domains, including text, music, images and video, and bioinformatics; as reviewed by Tsoumakas et al. (2010).

The multi-label learning problem is in fact a special case of *multi-target* learning (also known as multi-dimensional or multi-objective), where each label can take multiple values, as opposed to binary labels indicating relevance or not (Bielza et al., 2011; Appice and Džeroski, 2007).

The context of multiple target variables has important implications for classification (how to model dependencies between variables) and evaluation (how to score multiple target classifications for each instance) that traditional single-label frameworks do not deal with. Meka has been designed specifically for this context.

2. The Meka Framework

Meka was created to perform and evaluate multi-label classification using the popular and effective family of *problem transformation* methods, which make use of existing off-the-shelf single-label (binary or multi-class) methods as 'base classifiers'. The Weka framework (Hall et al., 2009) contains a plethora of well-documented single-label classifiers used by the machine learning community for many years that can be used as such.

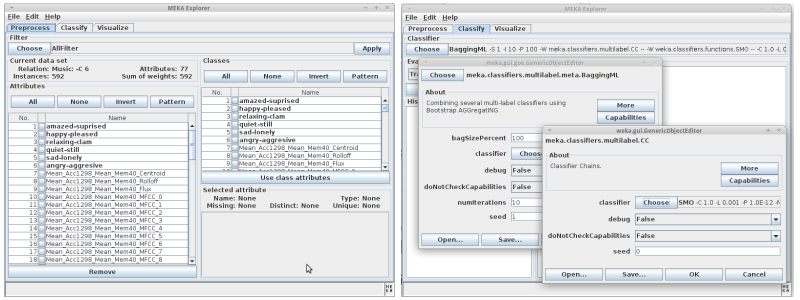


Figure 1: MEKA's GUI interface; having loaded and setup the Music data set (left) and selecting and configuring a classifier (right).

Meka contains all the basic problem transformation methods as reviewed by Tsoumakas et al. (2010), advanced methods including many of those surveyed by Madjarov et al. (2012) and Zhang and Zhou (2013), many varieties of classifier chains (Read et al., 2011) which have often been used as a benchmark in the recent multi-label literature, and also algorithm adaptations such as multi-label neural networks and deep neural networks (Hinton and Salakhutdinov, 2006). It includes two strategies for automatic threshold calibration, and a variety of evaluation metrics from the literature.

Meka is easily used from either the command line interface (CLI) or graphical user interface (GUI). Thus no programming is required to parameterize, run, and evaluate classifiers, making it suitable for practitioners unfamiliar with Java. However, it is straightforward to extend Meka with new classifiers and integrate it in other frameworks. Those familiar with Weka will have almost no learning curve (much of Weka's documentation and modus operandi is directly applicable). Any new Meka classifier can also be combined within any of Meka's existing ensemble schemes and any Weka base classifier without writing extra code, and be compared easily with benchmark and state-of-the-art methods. Meka also offers support for semi-supervised and incremental classification in the multi-label context (Read et al., 2015), as well as the general multi-target context considered by Appice and Džeroski (2007), making it a good platform for working in these areas.

2.1 Using Meka

Meka can be run on any machine with Java installed (version 1.7 or above). Everything needed to run Meka is included in the distribution package and detailed instructions and examples on how to setup data sets and run experiments are included in the tutorial. Figure 1 shows screen captures of Meka's GUI.

Meka uses Weka's thoroughly-documented ARFF file format,[1] using multiple binary attributes to specify the label relevances (or nominal attributes in the case of multi-target). The target attributes can easily be configured using the GUI, or directly from within the ARFF file (optionally using Weka's filters).

Meka's CLI follows the style of Weka. For example, to run five fold cross validation of an ensemble of 50 chain classifiers (Read et al., 2011) on the Music data set with support vector machines as the base classifier (as also shown on the right of Figure 1):

```
java meka.classifiers.multilabel.meta.BaggingML -x 5 -t data/Music.arff -I 50 \
    -W meka.classifiers.multilabel.CC -- -W weka.classifiers.trees.SMO
```

A list of methods implemented in Meka along with examples, is given at `http://meka.sourceforge.net/methods.html`. Further examples and documentation can be found in the tutorial and API reference, where relevant publications are also mentioned. Meka also includes a wrapper to Mulan (the `MULAN` class) to allow additional classifiers to be run from Meka's interfaces. Meka includes over a dozen evaluation metrics, including Hamming loss, 0/1 loss, Jaccard index, rank loss, log loss, and F1-measure (macro and micro averages). Several metrics can be output overall and per-label.

2.2 Extending Meka

Meka can be extended by writing classifiers that implement the `MultiLabelClassifier` interface. A multi-label classifier uses the same methods as a Weka classifier, namely the `buildClassifier(Instances)` and `distributionForInstance(Instance)` methods. The difference in Meka is that `classIndex()` will indicate the *number* of target attributes. As in Weka, the `distributionForInstance` method returns a vector which may be votes or posterior probabilities. In the latter case, a threshold or thresholds can automatically be calibrated, or defined ad-hoc according to user preference, to produce 0/1 label relevances. The tutorial deals with the general multi-target case.

Incremental classifiers simply implement `IncrementalMultiLabelClassifier` and thus, as with Weka's `UpdateableClassifier` interface, have an `updateClassifier(Instance)` method. Meka automatically carries out incremental learning and evaluation for any updateable classifiers. Semi-supervised classifiers implement `SemisupervisedClassifier` interface and must implement the method `setUnlabelledData(Instances)` which is called by the evaluation routine.

3. Related frameworks

Mulan (Tsoumakas et al., 2011) is a programmatic API also based on Weka providing a number of multi-label algorithms implemented from the literature. There is some overlap in terms of classifiers available, but the usage and performance is different. Mulan is an API, whereas Meka provides both graphical and command-line interfaces and this allows novel classifier and ensemble combinations and parameterizations thereof to be trialled rapidly without writing any code; allowing much flexibility when tackling a multi-label problem. A single command line is often sufficient to reproduce the results of publications. Both frameworks have attracted considerable interest and both are mentioned in the literature.

1. See, for example, `http://weka.wikispaces.com/ARFF`.

	Clus	Mulan	Meka
interface	config. file		CLI, GUI
multi-target	✓	regression	classification
hierarchical	✓	✓	✓
incremental	✓		✓
semi-supervised	✓		✓
num. of classifiers	2+	20+	20+
classification paradigms	decision trees/rules	Weka+others	Weka+others

Table 1: Comparing Meka with related frameworks.

Data set	Meka	Mulan
Enron	8	33
Corel-5k	10	35
MediaMill	449	2104

Table 2: Running time (s) of `RAkEL` under Meka and Mulan, with `SMO`, $k = 3, m = 10$.

Meka specializes in the problem transformation methods, in particular the classifier chains paradigm, implementing at least 10 variations; and also meta methods for combining them together in many combinations.

Clus is a decision tree and rule learning system (Appice and Džeroski, 2007) that can also carry out multi-label and multi-target classification. Classifiers are configured in a text file. Clus's implementations of decision tree and rule algorithms are scalable and competitive; but it does not include many of the popular problem transformation methods.

Table 1 roughly compares the frameworks in terms of features. Clus has fewer classification paradigms, but can be used in many different contexts. Mulan has a similar number of classifiers but Meka has different methods. Meka is noticeably faster than Mulan in some implementations under the same configurations, for example `RAkEL`, see Table 2.

Meka itself integrates well in other frameworks. For example, it is already part of the Moa (data streams), DKPro (text classification), and ADAMS (workflow engine) frameworks, and available via Maven Central.[2]

4. Summary and Further Notes

Meka provides a multi-label and multi-target framework with comfortable command line and graphical interfaces. This includes traditional, benchmark, and state of the art algorithms; and a multitude of multi-label specific evaluation measures. It also includes methods for incremental, multi-target, and semi-supervised learning, and is easy to use and extend. Meka is released under the GNU GPL 3 licence; can run on any Java (1.7 or above) machine. The software with source code, API reference, data sets, list of methods with examples, and a tutorial can be found at at `http://meka.sourceforge.net`, along with links to additional material about learning from multi-label and multi-target data.

2. See `http://moa.cms.waikato.ac.nz/`, `https://code.google.com/p/dkpro-tc/`, `https://adams.cms.waikato.ac.nz/`, and `https://search.maven.org`, respectively

References

Annalisa Appice and Saso Džeroski. Stepwise induction of multi-target model trees. In *ECML '07: Proceedings of the 18th European Conference on Machine Learning*, pages 502–509, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-74957-8. doi: 10.1007/978-3-540-74958-5_46. URL http://dx.doi.org/10.1007/978-3-540-74958-5_46.

Concha Bielza, Guangdi Li, and Pedro Larrañaga. Multi-dimensional classification with bayesian networks. *International Journal of Approximate Reasoning*, 52:705–727, 2011.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Reutemann Peter, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11 (1), 2009.

Geoffrey Hinton and Ruslan Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504 – 507, 2006.

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Saso Dzeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recogn.*, 45(9): 3084–3104, 2012. ISSN 0031-3203. doi: 10.1016/j.patcog.2012.03.004. URL http://dx.doi.org/10.1016/j.patcog.2012.03.004.

Jesse Read, Bernhard Pfahringer, Geoffrey Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, 2011.

Jesse Read, Albert Bifet, and Fernando Perez-Cruz. Deep learning in partially-labelled data streams. In *SAC 2015: 30th ACM Symposium on Applied Computing*. ACM, 2015.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas. Mining multi-label data. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*. 2nd edition, Springer, 2010.

Grigorios Tsoumakas, Jozef Vilcek, Eleftherios Spyromitros Xioufis, and Ioannis Vlahavas. MULAN: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints):1, 2013. ISSN 1041-4347. doi: http://doi.ieeecomputersociety.org/10.1109/TKDE.2013.39.

Journal of Machine Learning Research 17 (2016) 1-34 Submitted 8/13; Revised 2/16; Published 4/16

Gradients Weights improve Regression and Classification

Samory Kpotufe* SAMORY@PRINCETON.EDU
Princeton University, Princeton, NJ

Abdeslam Boularias ABDESLAM.BOULARIAS@CS.RUTGERS.EDU
Rutgers University, New Brunswick, NJ

Thomas Schultz SCHULTZ@CS.UNI-BONN.DE
University of Bonn, Germany

Kyoungok Kim KYOUNGOK.KIM@SEOULTECH.AC.KR
Seoul National University of Science & Technology (SeoulTech), Korea

Editor: Hui Zou

Abstract

In regression problems over $\mathbb{R}^d$, the unknown function f often varies more in some coordinates than in others. We show that weighting each coordinate i according to an estimate of the variation of f along coordinate i – e.g. the L_1 norm of the ith-directional derivative of f – is an efficient way to significantly improve the performance of distance-based regressors such as kernel and k-NN regressors. The approach, termed Gradient Weighting (GW), consists of a first pass regression estimate f_n which serves to evaluate the directional derivatives of f, and a second-pass regression estimate on the re-weighted data. The GW approach can be instantiated for both regression and classification, and is grounded in strong theoretical principles having to do with the way regression bias and variance are affected by a generic feature-weighting scheme. These theoretical principles provide further technical foundation for some existing feature-weighting heuristics that have proved successful in practice.

We propose a simple estimator of these derivative norms and prove its consistency. The proposed estimator computes efficiently and easily extends to run online. We then derive a classification version of the GW approach which evaluates on real-worlds datasets with as much success as its regression counterpart.

Keywords: Nonparametric learning, feature selection, feature weighting, nonparametric sparsity, metric learning.

1. Introduction

High-dimensional regression is the problem of inferring an unknown function f from data pairs (X_i, Y_i), $i = 1, 2, \dots n$, where the input X_i belongs to a Euclidean subspace $\mathcal{X} \subset \mathbb{R}^d$ and the output Y_i is a noisy version of $f(X_i)$. The problem is significantly harder for larger dimension d, and various pre-processing approaches have been devised over time to alleviate this so-called *curse of dimension*. A simple and common approach is that of reducing the dimension of the input X by properly selecting a few coordinate-variables with the most influence on the problem. The general motivating assumption for these methods is that of (approximate) *sparsity*: the unknown function f only varies along a few relevant coordinates in some subset R of $[d] \doteq \{1, 2, \dots, d\}$, that is $f(X) = f\left(X_{(R)}\right)$ where $X_{(R)}$ picks out the set of relevant coordinates. However, as illustrated by Fig. 3, there are many real-world examples in which f varies significantly along all coordinates, but varies more in some coordinates than in others. The natural approach in this case, implicit in some methods and heuristics (see Section 1.2), is to *weight* each coordinate according to some measure of relevance learned from data. The learned coordinate-relevance would typically rely on various assumptions on the form of f, for example f might be assumed linear.

In the case of nonparametric regression, where little is assumed about the form of f, the question of how to properly weight coordinates has not received much theoretical attention. We present and analyze a simple approach termed Gradient Weighting (GW), consisting of weighting each coordinate i according to the (unknown) variation of f along i. To this end, f is estimated from data in a first pass as f_n, where f_n serves to assess the coordinate-wise variation of f and accordingly weight the data; the transformed data is then used to re-estimate f in a second pass. This procedure can be iterated into a multi-pass procedure, although we only consider the two-pass version just described. We show that such weighting can be learned efficiently, is easily extended online, and can significantly improve the performance of distance-based regressors (e.g. kernel and k-NN regression) in real-world applications. Moreover the method easily extends to distance-based classification methods such as k-NN classification and ϵ-NN classification.

The GW approach is grounded in strong theoretical principles (developed in Section 2) having to do with the way regression bias and variance are affected by the distribution of weights in a generic feature-weighting method. We argue in Section 2 that a good situation for distance-based regressors is one where the unknown function f varies in a few coordinates more than in others, and the weights correlate with the variation of f along coordinates. The theoretical intuition developed is kept general enough to also explain the practical success of some existing heuristics (see Section 6.2) which inherently learn weights that are correlated with the variation of the unknown f along coordinates. We validate the theoretical intuition in extensive experiments on many real-world datasets in Section 5.

There are many possible ways of capturing the variation of f along coordinates, thus the particular instantiations of GW considered here are simply ones that work well in practice. Our aim is therefore not of arguing in favor of a particular way of capturing the coordinate-wise variation of f, but rather that the general approach of weighting coordinates according to this variation of f can yield significant improvements in learning performance. The theoretical intuition developed in Section 2 uses, as a measure of the variation of f along i, the maximum variation $|f_i'|_{\text{sup}} \triangleq \sup_x |f_i'(x)|$ along coordinate i. The maximum variation is a natural measure of smoothness and as such is intuitive to argue about; however it is hard to estimate. Therefore, for practical instantiations of the GW approach, we instead measure the average variation of f along coordinates, specifically we estimate the norms $\|f_i'\|_{1,\mu} = \mathbb{E}_{X\sim\mu} |f_i'(X)|$, where μ denotes the marginal measure over X.

A significant portion of this work is dedicated to efficiently estimating the gradient-norms $\|f_i'\|_{1,\mu}$. The aim is to obtain a simple, practical and successful procedure grounded in the theoretical intuition developed. We show in Section 3 that these gradient-norms can be estimated efficiently (a brief overview is introduced in the subsection below), and we prove in Section 4 that the resulting method is statistically consistent. As previously mentioned, the resulting instantiations of GW evaluate successfully in practice as shown in Section 5.

*. A significant part of this work was conducted when the authors were at the Max Planck Institute for Intelligent Systems, Tuebingen, Germany.

1.1 GW for distance-based nonparametric methods

For distance-based methods, the weights can be incorporated into a distance function of the form

$$\rho(x,x') \triangleq \left((x-x')^\top \mathbf{W}(x-x')\right)^{1/2}, \quad (1)$$

where each element $\mathbf{W}_i$ of the diagonal matrix $\mathbf{W}$ is an estimate of the variation of f along coordinate i, as captured for instance by $\|f_i'\|_{1,\mu}$. In our evaluations we set $\mathbf{W}_i$ to an estimate $\nabla_{n,i}$ of $\|f_i'\|_{1,\mu}$, or to the square estimate $\nabla^2_{n,i}$.

To estimate $\|f_i'\|_{1,\mu}$, one does not need to estimate f_i' well everywhere, just well on average. While many elaborate derivative estimators exist (see e.g. (Härdle and Gasser, 1985)), we have to keep in mind our need for a fast but consistent estimator of $\|f_i'\|_{1,\mu}$. We propose a simple estimator $\nabla_{n,i}$ which averages the differences along i of an estimator $f_{n,h}$ of f. More precisely (see Section 3) $\nabla_{n,i}$ has the form $\mathbb{E}_n\left|f_{n,h}(X+te_i)-f_{n,h}(X-te_i)\right|/2t$ where $\mathbb{E}_n$ denotes empirical expectation over a sample $\{X_i\}_1^n$. $\nabla_{n,i}$ can therefore be updated online at the cost of just two estimates of $f_{n,h}$, given a proper online version of $f_{n,h}$ (see e.g. Gu and Lafferty (2012)).

In this paper $f_{n,h}$ is a kernel estimator, although any regression method might be used in estimating $\|f_i'\|_{1,\mu}$. We prove in Section 4 that, under mild conditions, $\mathbf{W}_i$ is a consistent estimator of the unknown norm $\|f_i'\|_{1,\mu}$. Moreover we prove finite sample convergence bounds to help guide the practical tuning of the two parameters t and h.

1.2 Related Work

The GW approach is close in spirit to *metric learning* (Weinberger and Tesauro, 2007; Xiao et al., 2009; Shalev-shwartz et al., 2004; Davis et al., 2007), where the best metric ρ is found by optimizing over a sufficiently large space of possible metrics. Clearly metric learning can only yield better performance, but the optimization over a larger space will result in heavier preprocessing time, often $O(n^2)$ on datasets of size n. Yet, preprocessing time is especially important in many modern applications where data sizes are large, or where training and prediction have real-time constraints (e.g. robotics, finance, advertisement, recommendation systems). Here we do not optimize over a space of metrics, but rather estimate a *single* metric ρ based on the coordinate-wise variation of f. Our metric ρ is efficiently obtained, can be estimated online, and still significantly improves the performance of distance-based regressors.

We also note that there are actually few metric learning approaches for regression and these are typically designed around a particular regression approach or problem. The method by Weinberger and Tesauro (2007) is designed for Gaussian-kernel regression, the one by Xiao et al. (2009) is tuned to the particular problem of age estimation. For the problem of classification, the metric-learning approaches of Shalev-shwartz et al. (2004); Davis et al. (2007) are meant for online applications – they are therefore relatively efficient methods – but cannot be used in regression.

In the case of kernel regression and local polynomial regression, multiple bandwidths can be used, one for each coordinate. However, tuning d bandwidth parameters requires searching a d-dimensional grid, i.e. the number of possible settings is exponential in d, which is impractical even in batch mode. The *RODEO* method of Lafferty and Wasserman (2005) alleviates this problem, however only in the particular case of local linear regression. Our method applies to any distance-based regressor.

The ideas presented here are related to recent notions of nonparametric sparsity where it is assumed that the target function is well approximated by a *sparse* function, i.e. one which varies in

just a few coordinates (e.g. Hoffmann and Lepski (2002); Lafferty and Wasserman (2005); Rigollet and Tsybakov (2011); Rosasco et al. (2012)). The method of Rosasco et al. (2012) is most related to the present work in that they employ a penalized learning objective based on the coordinate-wise variation of f, as captured by the L_2 gradient norms $\|f_i'\|_{2,\mu}$. However, as in the other works on sparsity just mentioned, Rosasco et al. (2012) relies on f being actually sparse or at least close to sparse. In the present work we do not need sparsity, instead we only need the target function f to vary in some coordinates more than in others which is most likely the case in practice. Our approach therefore works in practice even in cases where the target function is far from sparse.

One line of work which also does away with the assumption of sparsity of f, is that of *anisotropic* regression (Nusbaum (1983); Hoffmann and Lepski (2002)). Anisotropic regression assumes that the target function f does not have the same degree of smoothness in all coordinate directions, where smoothness is roughly captured by the number of bounded derivatives of f. The attainable rates in anisotropic regression are better than the usual minimax rates for nonparametric regression (e.g. Stone (1980)) which consider the worst-case degree of smoothness across all coordinates. In the present work, we only consider the first derivatives of f across coordinates, in other words f is allowed to have the same degree of smoothness across coordinates, but we are interested in the case where these coordinate-wise derivatives have different magnitudes. We show in Section 2 that this is enough to attain better rates than the usual minimax rates, in particular by using the GW approach proposed here.

As previously mentioned, the theoretical intuition developed in this work helps explain the practical success of some existing heuristics. In particular the popular Relief family of heuristics (e.g. Kira and Rendell (1992); Kononenko (1994); Robnik-Šikonja and Kononenko (2003)) can be viewed as inherently learning weights that are correlated with the coordinate-wise variation of f. Our work therefore offers new insights about existing heuristics and opens possible avenues of further development of these heuristics. This theme is further developed in Section 6.2.

Finally, part of this work appeared as a conference version Kpotufe and Boularias (2012) covering the case of regression. The present work covers both regression and classification and further differs in the technical motivation offered for the GW approach. While Kpotufe and Boularias (2012) argues for GW under strong uniform assumptions on the marginal distribution μ on $\mathcal{X}$, the theoretical intuition developed in the present work assumes a general distribution μ. The more general assumptions are made possible by introducing a new set of techniques dealing with the covering numbers of the space $\mathcal{X}$ after data weighting.

1.3 Paper Outline

In summary, we develop theoretical intuition for GW in Section 2. In Section 3 we derive a concrete method for estimating the coordinate-wise variablity of f; the method is both simple and efficient. We show in Section 4 that the method is a consistent estimator of the gradient norms $\|f_i'\|_{1,\mu}$. In Section 5 we validate our theory on various real-world applications. We finish with a general discussion of our results in Section 6, including possible future directions.

2. Theoretical Justification of GW

In this section we develop theoretical intuition about why GW works. We focus on the problem of nonparametric regression since the same intuition is easily implied for the case of nonparametric classification (see Section 2.3). We will argue that it is possible to attain good regression rates

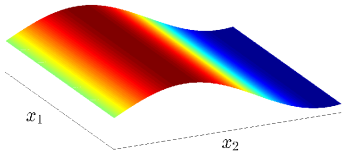


Figure 1: Illustration of a sparse function. Here $x = (x_1, x_2)$, and $f(x) = f(x_2)$. There is no variation in f along coordinate 1, in other words $|f_1'|_{\sup} = 0$ and $\|f_1'\|_{1,\mu} = 0$.

even when the target function f depends on all coordinates, provided f does not vary equally in all coordinates, and the gradient weights $\mathbf{W}_i$ are correlated with the coordinate-wise variation in f. The coordinate-wise variation will be captured in this discussion by the quantities $|f_i'|_{\sup} \triangleq \sup_{x\in\mathcal{X}} |f_i'(x)|$, $i \in [d]$, as previously mentioned.

We will consider the metric ρ generally: instead of assuming a particular form, we will let the analysis uncover a form of ρ which yields *improved* regression rates. Improvement is measured here in a minimax sense which will soon be made clear. The analysis of this section will thus yield intuition not only about GW, but about coordinate-weighting generally.

We have the following assumption throughout the section.

Assumption 2.1 *The input space $\mathcal{X}$ is full-dimensional in $\mathbb{R}^d$, connected*[1]*, and has bounded diameter $\|\mathcal{X}\| \triangleq \sup_{x,x'\in\mathcal{X}} \|x - x'\| = 1$. The output space is $\mathcal{Y} = [0, 1]$.*

2.1 Rough Intuition: the sparse case

We start with a simple case where the unknown function is actually R-sparse, i.e. depends on a small set of coordinates $R \subsetneq [d]$ (illustrated in Figure 2.1). The function f then varies only along coordinates in R, i.e. $f_i' \neq 0$ only for coordinates $i \in R$. Hence if the metric ρ is defined by setting the gradient weights $\mathbf{W}_i$ to either $|f_i'|_{\sup}$ and $\|f_i'\|_{1,\mu}$, the resulting space $(\mathcal{X}, \rho)$ is a (weighted) projection of the original Euclidean $\mathcal{X}$ down to just the relevant coordinates $R \subsetneq [d]$. Thus regression or classification on $(\mathcal{X}, \rho)$ would have performance depending on the lower-dimension $|R|$ of this space, rather than the high-dimension d of the original space.

To make this intuition precise, we introduce the following definition and minimax theorem.

Definition 1 (The class $\mathcal{F}_\lambda$) *Given $\lambda > 0$, we let $\mathcal{F}_\lambda$ denote all distributions $P_{X,Y}$ on $\mathcal{X} \times [0, 1]$ such that, for all $i \in [d]$, the directional derivatives of $f(x) \triangleq \mathbb{E}[Y|X = x]$ satisfy $|f_i'|_{sup} \triangleq \sup_{x\in\mathcal{X}} |f_i'(x)| \leq \lambda$.*

The worst-case rate for the class $\mathcal{F}_\lambda$ is given in the following minimax theorem of Stone.

Theorem 2 (Minimax rate for $\mathcal{F}_\lambda$: Stone (1982)) *There exists $\tilde{c} < 1$, independent of n, such that for all sample sizes $n \in \mathbb{N}$,*

$$\inf_{f_n} \sup_{f\in\mathcal{F}_\lambda} \mathbb{E}_{\mathbf{X}^n,\mathbf{Y}^n} \|f_n - f\|^2 \geq 2\tilde{c}^{2/(2+d)}(d\lambda)^{2d/(2+d)} n^{-2/2+d},$$

1. This is needed so that $\|f_i'\|_{1,\mu} = 0$ implies that f is a.e. constant along coordinate i.

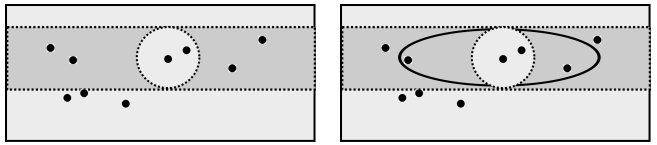


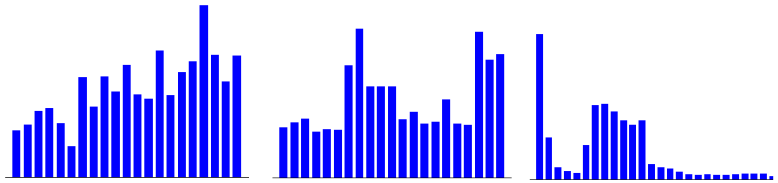
Figure 2: Balls $B(x, h)$ before and after projection or reweighting. The dotted points are sample $\{X_i\}$. **Left:** after projection onto 1 dimension, the new ball $B(x, h)$ contains all the points shown in the dotted rectangle, much more than the original ball (dotted circle). **Right:** feature-reweighting has the approximate effect of a projection; the new ball $B(x, h)$ is an ellipsoid containing more points in the directions with small weight.

where the infimum is taken over all regressors f_n mapping samples $\mathbf{X}^n, \mathbf{Y}^n$ to an L_2 measureable function (also denoted f_n for simplicity of notation), and the expectation is taken over the draw of an n-sample from a distribution where $\mathbb{E}[Y|X = x] = f(x)$.

Thus, in a minimax sense, the best rate achievable for (non-sparse) Lipschitz functions is f the form $O\left(n^{-2/(2+d)}\right)$. However when the unknown function happens to be R-sparse, the better rate of $O\left(n^{-2/(2+|R|)}\right)$ is achievable (see e.g. Lafferty and Wasserman (2005)). The better rates are achieved for instance by performing regression on the data projected to the span of R. This is the same as setting the weights $\mathbf{W}_i$, $i \notin R$, to 0. In other words, we would let $\mathbf{W}_i$ be correlated with the variation of f along coordinate i as discussed earlier.

To better understand why better rates are possible after a dimension reduction to the span of R as described above, let's consider the case of a kernel estimate $f_{n,h}(x)$ using a bandwidth h. The error of $f_{n,h}(x)$ depends on its variance and bias. The variance itself decreases with the number of points contributing most to the estimate: this is roughly (depending on the kernel) the number of points falling in the ball $B(x, h)$ in the given metric space. Since balls of a fixed radius have larger mass in smaller dimensional space (illustrated in Fig. 2), projection decreases the variance of the kernel estimate $f_{n,h}(x)$. In addition, if the unknown f does not vary along those directions $i \notin R$ eliminated by the projection, the bias of the estimate $f_{n,h}(x)$ remains unaffected; in other words, the projection loses no information about the unknown f. This combined effect on variance and bias decreases the error of the estimate.

But what if f is not actually sparse, but close to being sparse? i.e. f varies in all coordinates, but varies little in most coordinates. Intuitively, given the above discussion, better rates should also be achievable in this case. More interestingly, what if f varies considerably in all coordinates but much more in some than in most? This is a more practical situation which is more realistic with real-world data (see e.g. Figure 3). The above variance-bias intuition still applies if we reweight coordinates according to how f varies; as illustrated in Figure 2 (right), this acts as an approximate dimension reduction which also reduces the variance of regression estimates while keeping the bias relatively unaffected. We formalize this intuition in the next section.



(a) SARCOS robot, joint 7. (b) Parkinson's. (c) Telecom.

Figure 3: Typical gradient norms (estimates of $\|f_i'\|_{1,\mu}$, $i \in [d]$) for some real-world datasets.

2.2 Technical intuition: the non-sparse case

In this section we aim to understand how regression performance is affected by the distribution of weights $\mathbf{W}_i$ with respect to the variation of f along coordinates. The main situation of interest is one where f varies in all coordinates, but in some coordinates more than in others. This situation is captured by assuming the quantities $|f_i'|_{\text{sup}}$ are all nonzero but some are considerably smaller than others. We also assume that the weights $\mathbf{W}_i$ are nonzero for all $i \in [d]$.

We will be bounding the error of a box-kernel regressor operating on the transformed space $(\mathcal{X}, \rho)$, in terms of how the weights $\mathbf{W}_i$ scale relative to each other, and relative to the variation $|f_i'|_{\text{sup}}$. We will see that, even in non-sparse situations where all $|f_i'|_{\text{sup}}$ are far from 0, it is possible to achieve finite-sample rates better than the minimax rate of Theorem 2, provided (i) the weights $\mathbf{W}_i$ are sufficiently larger in scale for a small subset R of the coordinates (for low variance), and (ii) each $\mathbf{W}_i$ is sufficiently correlated with $|f_i'|_{\text{sup}}$ (for low bias).

The results of this section uncover an interesting phenomenon, also observed in experiments, that improvement might only be possible in a specific mid-range sample size regime depending on problem parameters. This is unsurprising: when the sample size is too small, any algorithm will likely only fit noise and good rates would not be possible; as the sample size gets quite large, algorithms operating in the original space tend to also do well, and the advantage of operating in $(\mathcal{X}, \rho)$ becomes negligible (see Remark below). We believe this behavior is not limited to the GW approach, because the results here are not directly tied to GW since our analysis is in terms of a general metric ρ of the form (1).

We will see that the attainable convergence rates are smaller than the minimax rate of $\Omega(n^{-2/(2+d)})$ for n greater than some problem-specific n_0 (Corollary 8). These rates tend towards the minimax rate from below as $n \to \infty$.

Remark 2.1 *There is theoretical intuition as to why improvement over the minimax rate is unlikely in the asymptotic regime where $n \to \infty$. Remember that the metric ρ is norm-induced and all norms are equivalent on finite-dimensional spaces. In other words there exist C, C' such that for all $x, x' \in \mathbb{R}^d$, $C\|x - x'\| \leq \rho(x, x') \leq C'\|x - x'\|$. As a consequence there exists C'' such that, for ϵ sufficiently small, any ϵr-cover of a ρ-ball $B(x, r)$ centered on $x \in \mathcal{X}$ has size at least $C''\epsilon^{-d}$. Here C'' depends on the metric $(\mathcal{X}, \rho)$. It is known (see e.g. the lower-bound of Kpotufe (2011)) that such space covering properties influence the attainable regression rates, where larger data sizes n correspond to finer coverings of the space, hence to small ϵ. We therefore postulate that, for large n, the worst-case asymptotic rate is no better than $\Omega\left(n^{-2/(2+d)}\right)$. Thus we can only expect improvements over the minimax rate in small sample regimes, where* small *is problem-specific. Note that this remark is of independent interest since other approaches such as metric learning would typically use norm-induced metrics.*

The analysis in this section, although focusing on kernel regression, yields general intuition about the behavior of other related distance-based regressors such as k-NN, since such regressors are similarly affected by characteristics of the regression problem (e.g. smoothness of f, intrinsic dimension of $(\mathcal{X}, \rho)$).

We start by defining quantities which serve to describe the distribution of weights $\mathbf{W}_i$.

Definition 3 *For any subset of coordinates $R \subset [d]$, define $\kappa_R \triangleq \sqrt{\max_{i \in R} \mathbf{W}_i / \min_{i \in R} \mathbf{W}_i}$, and let $\epsilon_{R\!\!R} \triangleq 2\sqrt{\max_{i \notin R} \mathbf{W}_i}$. Finally we define the* ρ**-diameter** *of $\mathcal{X}$ as $\rho(\mathcal{X}) \triangleq \sup_{x,x' \in \mathcal{X}} \rho(x, x')$.*

As discussed earlier, regression variance is small if there exists a small subset R for which the weights $\mathbf{W}_i$ are relatively larger than those for coordinates not in R. Formally, we want the quantities $|R|$, $\epsilon_{R\!\!R}$ and κ_R relatively small for some $R \subset [d]$. How small will become gradually clearer.

We start with two results (Lemmas 4 and 5 below) on properties of the metric space $(\mathcal{X}, \rho)$ which affect the behavior of a distance-based regressor such as $f_{n,\epsilon,\rho}$. All omitted proofs are given in the appendix.

The first Lemma 4 concerns the size of minimal covers (at different scales) of the metric $(\mathcal{X}, \rho)$. Such cover sizes influence the variance of a distance-based regressor operating on $(\mathcal{X}, \rho)$. If $(\mathcal{X}, \rho)$ has small cover-sizes, it can typically be covered by large balls, each ball likely to contain enough data for small regression variance. Lemma 4 roughly states that, while a minimal ϵ-cover of the Euclidean space $\mathcal{X} \subset \mathbb{R}^d$ has size $O\left(\epsilon^{-d}\right)$, an $\epsilon\rho(\mathcal{X})$-cover of $(\mathcal{X}, \rho)$ has smaller size $C\epsilon^{-r}$ where $|R| \leq r \xrightarrow{\epsilon \to 0} d$. Both C and the function $r(\epsilon)$ depend on the relative distribution of weights $\mathbf{W}_i$ as captured by the quantities $|R|$, $\epsilon_{R\!\!R}$ and κ_R.

Lemma 4 (Covering numbers) *Consider $R \subset [d]$ such that $\max_{i \notin R} \mathbf{W}_i < \min_{i \in R} \mathbf{W}_i$. There exist $C \leq C'(4\kappa_R)^{|R|}$ such that, for any $\epsilon > 0$, the smallest $\epsilon\rho(\mathcal{X})$-cover of $(\mathcal{X}, \rho)$ has size at most $C\epsilon^{-r(\epsilon)}$, where $r(\epsilon)$ is a nondecreasing function of ϵ satisfying*

$$r(\epsilon) \leq \begin{cases} |R| & \text{if } \epsilon \geq \epsilon_{R\!\!R}/\rho(\mathcal{X}) \\ d - (d - |R|) \cdot \frac{\log(\rho(\mathcal{X})/\epsilon_{R\!\!R})}{\log(1/\epsilon)} & \text{if } \epsilon < \epsilon_{R\!\!R}/\rho(\mathcal{X}) \end{cases}$$

The function $r(\epsilon)$ captures the *dimension* of $(\mathcal{X}, \rho)$[2]. We thus want r to be small so that the transformation ρ acts like a low-dimensional projection and hence helps reduce variance. The function r is smallest when most weights are concentrated in a small subset R, i.e. we want small $|R|$ and small $\epsilon_{R\!\!R}/\rho(\mathcal{X})$ (this term captures the difference in magnitude between weights not in R and weights in R). It might therefore seem preferable to choose ρ in this way, but we have to be careful: not all dimension reduction is good since such a transformation ρ might introduce additional regression bias. We therefore need to understand how regression bias is affected by the distribution of weights $\mathbf{W}_i$ in the transformation ρ.

Lemma 5 below captures the smoothness properties of the target function f in the transformed metric $(\mathcal{X}, \rho)$. These smoothness properties affect the bias of distance-based regressors on $(\mathcal{X}, \rho)$.

2. The logarithm of covering numbers is a common measure of metric dimension, see e.g. Clarkson (2005) for an overview of the subject.

Lemma 5 (Change in Lipschitz smoothness for f) *Suppose each derivative f_i' is bounded on $\mathcal{X}$ by $|f_i'|_{sup}$. Assume $\mathbf{W}_i > 0$ whenever $|f_i'|_{sup} > 0$. Denote by R the largest subset of $[d]$ such that $|f_i'|_{sup} > 0$ for $i \in R$. We have for all $x, x' \in X$,*

$$\left|f(x) - f(x')\right| \leq \left(\sum_{i \in R} \frac{|f_i'|_{sup}}{\sqrt{\mathbf{W}_i}}\right) \rho(x, x').$$

We want f to be as smooth as possible, in other words we want the Lipschitz parameter $\left(\sum_{i \in R} \frac{|f_i'|_{\sup}}{\sqrt{\mathbf{W}_i}}\right)$ as small as possible. Suppose $\mathbf{W}_i$ is uncorrelated with the variation of f along i, e.g. $\mathbf{W}_i$ is small for those coordinates where $|f_i'|_{\sup}$ is large, then the function f might end up lacking smoothness in the modified space $(\mathcal{X}, \rho)$. While it is hard to estimate $|f_i'|_{\sup}$, we can expect it to be correlated with $\|f_i'\|_{1,\mu}$ which is easier to estimate (Section 3). Thus by keeping the weights $\mathbf{W}_i$ correlated with the gradient norms $\|f_i'\|_{1,\mu}$, we expect the function f to remain relatively smooth in the space $(\mathcal{X}, \rho)$ and hence we expect to maintain control on regression bias. Since it is unlikely in practice that f varies equally in all coordinates (Figure 3), in light of Lemma 20, we will expect better regression performance in the space $(\mathcal{X}, \rho)$ as variance should decrease while bias remains controlled.

Remark 2.2 *As previously mentioned, there are many ways to incorporate the coordinate-wise variability of f into the weights ρ, and GW (or Relief heuristics discussed in Section 6.2) is just one of this. In light of the Lipschitz parameter $\left(\sum_{i \in R} \frac{|f_i'|_{sup}}{\sqrt{\mathbf{W}_i}}\right)$, we could reasonably set $\mathbf{W}_i$ to approximate $\|f_i'\|_{1,\mu}^q$ for some $q > 0$ ($q = 1, 2$ in this work), or to $\|f_i'\|_{1,\mu}^{q_i}$ for some power q_i depending on i. These are interesting questions deserving further investigation.*

We now go further in formalizing the intuition discussed so far by considering the case of kernel regression in $(\mathcal{X}, \rho)$. We will derive exact conditions on the distribution of weights $\mathbf{W}_i$ that allow improvements over the minimax rate of $O\left(n^{-2/(2+d)}\right)$ in the non-asymptotic regime.

The box-kernel regression estimate is defined as follows.

Definition 6 *Given $\epsilon > 0$ and $x \in \mathcal{X}$, the box-kernel estimate at x is defined as follows. Recall that $\rho(\mathcal{X}) \triangleq \sup_{x,x' \in \mathcal{X}} \rho(x, x')$ denotes the ρ-diameter of $\mathcal{X}$:*

$$f_{n,\epsilon,\rho}(x) \triangleq \text{average } Y_i \text{ of points } X_i \in B(x, \epsilon\rho(\mathcal{X})), \text{ or } 0 \text{ if } B(x, \epsilon\rho(\mathcal{X})) \text{ is empty.}$$

The next lemma establishes the convergence rate for a box-kernel regressor using any given bandwidth $\epsilon\rho(\mathcal{X})$. The lemma is a simple application of known results on the bias and variance of a kernel regressor combined with the previous two lemmas on the dimension of $(\mathcal{X}, \rho)$ and the smoothnness of f on $(\mathcal{X}, \rho)$.

Lemma 7 (Rate for $f_{n,\epsilon,\rho}$, arbitrary ϵ) *Consider any $R \subset [d]$ such that $\max_{i \notin R} \mathbf{W}_i < \min_{i \in R} \mathbf{W}_i$. There exist $1 \leq C_{\kappa_R} \leq C'(4\kappa_R)^{|R|}$, a universal constant C, and $\lambda_\rho \geq \sup_i |f_i'|_{sup} / \sqrt{\mathbf{W}_i}$ such that*

$$\mathop{\mathbb{E}}_{\mathbf{X}^n, \mathbf{Y}^n} |f_{n,\epsilon,\rho} - f|^2 \leq C_{\kappa_R} \frac{\epsilon^{-r(\epsilon)}}{n} + C^2 d^2 \lambda_\rho^2 \epsilon^2 \rho(\mathcal{X})^2,$$

where $r(\epsilon)$ is defined as in Lemma 4.

Proof By Lemma 5, f is $(d\lambda_\rho)$-Lipschitz on $(\mathcal{X}, \rho)$. We can then apply Theorem 5.2 of Gyorfi et al. (2002)[3] to bound the L_2 error as

$$\mathop{\mathbb{E}}_{\mathbf{X}^n, \mathbf{Y}^n} \mathop{\mathbb{E}}_{X} |f_{n,\epsilon,\rho}(X) - f(X)|^2 \leq C_1^2 \frac{N_\epsilon}{n} + C^2 d^2 \lambda_\rho^2 \epsilon^2 \rho(\mathcal{X})^2,$$

for some universal constants C_1, C, where N_ϵ denotes the size of a minimal $\epsilon\rho(\mathcal{X})$-cover of $(\mathcal{X}, \rho)$. Apply Lemma 4 to conclude. ■

We can now derive conditions on the distribution of weights $(\mathcal{X}, \rho)$ that permit good performance relative to the minimax rate of $O\left(n^{-2/(2+d)}\right)$. These conditions are given in equation (2). The main message of Corollary 8 below is that improvement in rate is possible in the non-asymptotic regime under rather mild conditions on the distribution of weights $\mathbf{W}_i$, even though improvement might not be possible in the asymptotic regime. In light of (2) sparseness (as described in Section 2.1) is not required, we simply need the function f to vary more along a small subset of coordinates ($R \subset [d]$) than along other coordinates, provided the weights $\mathbf{W}_i$ are properly correlated with the variation in f along coordinates. The correlation between $\mathbf{W}_i$ and gradients of f is implicit in the ratio $\lambda_\rho \rho(\mathcal{X})/\lambda$ of (2). The quantity λ captures the smoothness of f before the data transformation ρ, while λ_ρ captures the smoothness of f after the transformation ρ. The ratio $\lambda_\rho \rho(\mathcal{X})/\lambda$ thus captures the loss in smoothness (taking into account the change in diameter from 1 to $\rho(\mathcal{X})$) due to the transformation ρ, and this loss is controlled if $\mathbf{W}_i$ is correlated with the magnitude of the coordinate-wise derivatives of f.

Corollary 8 (Rate for $f_{n,\epsilon,\rho}$, optimal ϵ) *Let $\lambda \triangleq \sup_{i \in [d]} |f_i'|_{sup}$ and $\lambda_\rho \triangleq \sup_{i \in [d]} |f_i'|_{sup} / \sqrt{\mathbf{W}_i}$. Note that by definition $f \in \mathcal{F}_\lambda$. Let C_{κ_R} and C be defined as in Lemma 7, and $\tilde{c}$ as in Theorem 2. Suppose the following holds for some $R \subset [d]$:*

$$(d - |R|) \log\left(\frac{\rho(\mathcal{X})}{\epsilon_R}\right) \geq d \log\left(C \frac{\lambda_\rho \rho(\mathcal{X})}{\lambda}\right) + \log\left(\frac{C_{\kappa_R}}{\tilde{c}}\right). \tag{2}$$

Then there exists n_0 for which the following holds. For all $n \geq n_0$, there exist a bandwidth ϵ_n, and $r = r(\epsilon_n)$, where $|R| \leq r < d$, such that,

$$\mathbb{E} \|f_{n,\epsilon_n,\rho} - f\|^2 \leq 2C_{\kappa_R}^{2/2+r} (Cd\lambda_\rho \rho(\mathcal{X}))^{2r/(2+r)} n^{-2/2+r} < \inf_{f_n} \sup_{\mathcal{F}_\lambda} \mathbb{E} \|f_n - f\|^2.$$

Proof For $\epsilon > 0$, and $n \in \mathbb{N}$. Let $r(\epsilon)$ as in Lemma 4. Define the functions $\psi_{n,\rho}(\epsilon) = C_{\kappa_R} \epsilon^{-r(\epsilon)}/n$, and $\psi_{n,\not\rho}(\epsilon) = C_1' \epsilon^{-d}/n$, where $C_1' = \tilde{c}\left(\lambda / C\lambda_\rho \rho(\mathcal{X})\right)^d$. We also define $\phi(\epsilon) = C^2 d^2 \lambda_\rho^2 \rho(\mathcal{X})^2 \cdot \epsilon^2$.

Now recall (Theorem 2) that the minimax rate can be bounded below by

$$2\tilde{c}^{2/(2+d)} (d\lambda)^{2d/(2+d)} n^{-2/2+d}.$$

For any fixed n, let $\epsilon_{n,\not\rho}$ be a solution to $\psi_{n,\not\rho}(\epsilon) = \phi(\epsilon)$. Solving for $\epsilon_{n,\not\rho}$, we see that the minimax rate is bounded below by

$$2\phi\left(\epsilon_{n,\not\rho}\right) = 2\tilde{c}^{2/(2+d)} (d\lambda)^{2d/(2+d)} n^{-2/2+d}.$$

3. The theorem is stated for a Euclidean metric, but extends directly to any metric.

For any $n \in \mathbb{N}$, there exists a solution $\epsilon_{n,\rho}$ to the equation $\psi_{n,\rho}(\epsilon) = \phi(\epsilon)$ since $r(\epsilon)$ is nondecreasing. Therefore, by Lemma 7, we have

$$\mathop{\mathbb{E}}_{\mathbf{X}^n,\mathbf{Y}^n} \|f_{n,\epsilon,\rho} - f\|^2 \leq 2\phi\left(\epsilon_{n,\rho}\right).$$

We therefore want to show for a certain range of $n \in \mathbb{N}$ that $\phi\left(\epsilon_{n,\rho}\right) < \phi\left(\epsilon_{n,\not\rho}\right)$, equivalently that $\epsilon_{n,\rho} < \epsilon_{n,\not\rho}$. First notice that, since ϕ is independent of n, and both $\psi_{n,\rho}$ and $\psi_{n,\not\rho}$ are strictly decreasing functions of n, we have that $\epsilon_{n,\rho}$ and $\epsilon_{n,\not\rho}$ both tend to 0 as $n \to \infty$. Therefore we can define n_0 such that, for all $n \geq n_0$, both $\epsilon_{n,\rho}$ and $\epsilon_{n,\not\rho}$ are less than $\epsilon_{\not R}/\rho(\mathcal{X})$.

Thus, $\forall n \geq n_0$, we have $\epsilon_{n,\rho} < \epsilon_{n,\not\rho}$ if, for all $0 < \epsilon < \epsilon_{\not R}/\rho(\mathcal{X})$, $\psi_{n,\rho}(\epsilon) < \psi_{n,\not\rho}(\epsilon)$. This is insured by the conditions of equation (2), which are derived by recalling the bound of Lemma 4 on $r(\epsilon)$ for the range $0 < \epsilon < \epsilon_{\not R}/\rho(\mathcal{X})$. ■

2.3 The Case of Classification

We continue the intuition developed in the last section about the GW method with the case of classification, more precisely *plug-in* classification, defined as follows. Let $Y \in \{0,1\}$, and let η_n denote an estimate of the error function $\eta(x) \triangleq \mathbb{E}[Y|x] = \mathbb{P}(Y = 1|x)$. Then $\mathbf{1}\{\eta_n(x) > 1/2\}$ is a plug-in classification rule, emulating the Bayes optimal-classification rule $\mathbf{1}\{\eta(x) > 1/2\}$.

Two common examples of plug-in classification rules are the k-NN classifier and the ϵ-NN classifier which estimate Y at x as the majority label amongst, respectively, the k nearest neighbors of x, and the neighbors within distance ϵ of x. For both methods, the implicit estimate η_n of η is the average Y value of the neighbors of x.

The GW method for classification naturally corresponds to estimating the gradient norms $\|\eta_i'\|_{1,\mu}$ of the directional derivatives η_i'.

Since η_n is actually a regression estimate of the function $\eta(x)$, the 0-1 classification error of plug-in methods is related to that of regression as shown in the following well-known result.

Lemma 9 (Devroye et al. (1996)) *Let $\eta_n(x)$ be an estimator of $\eta(x)$, and let err(η), err(η_n), denote respectively the classification error rates of the Bayes classifier and that of the plug-in classification rule $\mathbf{1}\{\eta_n(x) > 1/2\}$. We have*

$$err(\eta_n) - err(\eta) \leq 2 \mathop{\mathbb{E}}_{X} |\eta_n(X) - \eta(X)|.$$

A bound on the classification error of ϵ-NN (operating in $(\mathcal{X},\rho)$) easily follows from Lemma 9 above and the analysis of the previous section on the properties of the space $(\mathcal{X},\rho)$.

Lemma 10 *Define $\eta_{n,\epsilon,\rho}(x)$ as the average Y value of the points in $\mathbf{X}^n \cap B_\rho(x, \epsilon\rho(\mathcal{X}))$ for some $\epsilon > 0$. Let $r(\epsilon)$ be defined as in Corollary 4. There exist a constant $1 \leq C_{\kappa_R} \leq C'(4\kappa_R)^{|R|/2}$, a universal constant C, and a constant $\lambda_\rho \geq \sup_i |\eta_i'|_{sup}/\sqrt{\mathbf{W}_i}$ such that*

$$\mathop{\mathbb{E}}_{\mathbf{X}^n,\mathbf{Y}^n} |err(\eta_{n,\epsilon,\rho}) - err(\eta)| \leq C_{\kappa_R}\sqrt{\frac{\epsilon^{-r(\epsilon)}}{n}} + Cd\lambda_\rho\epsilon\rho(\mathcal{X}).$$

Proof Using Lemma 9 and applying Jensen's inequality twice, we have

$$\mathop{\mathbb{E}}_{\mathbf{X}^n,\mathbf{Y}^n} |\text{err}(\eta_{n,\epsilon,\rho}) - \text{err}(\eta)| \leq \sqrt{\mathop{\mathbb{E}}_{\mathbf{X}^n,\mathbf{Y}^n} \mathop{\mathbb{E}}_{X} |\eta_{n,\epsilon,\rho}(X) - \eta(X)|^2}.$$

Thus, we just need to bound the L_2 error of $\eta_{n,\epsilon,\rho}$, which is a kernel regressor with a box kernel of bandwidth $\epsilon\rho(\mathcal{X})$. Apply Lemma 7 and conclude. ■

It follows similarly as in the case of regression that it is possible to achieve faster rates than the minimax rates even when the regression function η is not sparse, provided η does not vary equally in all coordinates. The exact conditions are exactly those of equation (2) with the constants C_{κ_R} and C of Lemma 10 above (this is easily derived from the above lemma as it was done for regression).

3. Estimating the GW $\mathbf{W}_i$

In all that follows we are given n i.i.d samples $(\mathbf{X}^n, \mathbf{Y}^n) = \{(X_i, Y_i)\}_{i=1}^n$ from some unknown distribution with marginal μ. The marginal μ has support $\mathcal{X} \subset \mathbb{R}^d$ while the output $Y \in \mathbb{R}$.

The kernel estimate at x is defined using any kernel $K(u)$, positive on $[0, 1/2]$, and 0 for $u > 1$. If $B(x,h) \cap \mathbf{X}^n = \emptyset$, $f_{n,h}(x) = \mathbb{E}_n Y$, otherwise

$$f_{n,\bar\rho,h}(x) = \sum_{i=1}^n \frac{K(\bar\rho(x,X_i)/h)}{\sum_{j=1}^n K(\bar\rho(x,X_j)/h)} \cdot Y_i = \sum_{i=1}^n w_i(x) Y_i, \tag{3}$$

for some metric $\bar\rho$ and a bandwidth parameter h.

For the kernel regressor $f_{n,h}$ used to learn the metric ρ below, $\bar\rho$ is the Euclidean metric. In the analysis we assume the bandwidth for $f_{n,h}$ is set as $h \geq \left(\log^2(n/\delta)/n\right)^{1/d}$, given a confidence parameter $0 < \delta < 1$. In practice we would learn h by cross-validation, but for the analysis we only need to know the existence of a good setting of h.

We estimate the norm $\|f_i'\|_{1,\mu}$ as follows:

$$\nabla_{n,i} \triangleq \mathbb{E}_n \frac{|f_{n,h}(X + te_i) - f_{n,h}(X - te_i)|}{2t} \cdot \mathbf{1}\{A_{n,i}(X)\} = \mathbb{E}_n\left[\Delta_{t,i} f_{n,h}(X) \cdot \mathbf{1}\{A_{n,i}(X)\}\right], \tag{4}$$

where $A_{n,i}(X)$ is the event that *enough* samples contribute to the estimate $\Delta_{t,i} f_{n,h}(X)$. For the consistency result, we assume the following setting:

$$A_{n,i}(X) \equiv \min_{s\in\{-t,t\}} \mu_n(B(X + se_i, h/2)) \geq \alpha_n \text{ where } \alpha_n \triangleq \frac{2d\ln 2n + \ln(4/\delta)}{n}.$$

The metric ρ is then obtained by setting the weights $\mathbf{W}_i$ to either $\nabla_{n,i}$ or the squared estimate $\nabla_{n,i}^2$ in all our experiments.

4. Consistency of the estimator $\mathbf{W}_i$ of $\|f_i'\|_{1,\mu}$

4.1 Theoretical setup

4.1.1 MARGINAL μ

Without loss of generality we assume $\mathcal{X}$ has bounded diameter 1. The marginal is assumed to have a continuous density on $\mathcal{X}$ and has mass everywhere on $\mathcal{X}$: $\forall x \in \mathcal{X}, \forall h > 0, \mu(B(x,h)) \geq C_\mu h^d$.

This is for instance the case if μ has a lower-bounded density on $\mathcal{X}$. Under this assumption, for samples X in dense regions, $X \pm te_i$ is also likely to be in a dense region.

4.1.2 REGRESSION FUNCTION AND NOISE

The output $Y \in \mathbb{R}$ is given as $Y = f(X) + \eta(X)$, where $\mathbb{E}\eta(X) = 0$. We assume the following general noise model: $\forall \delta > 0$ there exists $c > 0$ such that $\sup_{x \in \mathcal{X}} \mathbb{P}_{Y|X=x}(|\eta(x)| > c) \leq \delta$.

We denote by $C_Y(\delta)$ the infimum over all such c. For instance, suppose $\eta(X)$ has exponentially decreasing tail, then $\forall \delta > 0$, $C_Y(\delta) \leq O(\ln 1/\delta)$. A last assumption on the noise is that the variance of $(Y|X = x)$ is upper-bounded by a constant σ_Y^2 uniformly over all $x \in \mathcal{X}$.

Define the *τ-envelope* of $\mathcal{X}$ as $\mathcal{X} + B(0,\tau) \triangleq \{z \in B(x,\tau), x \in \mathcal{X}\}$. We assume there exists τ such that f is continuously differentiable on the τ-envelope $\mathcal{X} + B(0,\tau)$. Furthermore, each derivative $f_i'(x) = e_i^\top \nabla f(x)$ is upper bounded on $\mathcal{X} + B(0,\tau)$ by $|f_i'|_{\sup}$ and is uniformly continuous on $\mathcal{X} + B(0,\tau)$ (this is automatically the case if the support $\mathcal{X}$ is compact).

4.1.3 DISTRIBUTIONAL PARAMETERS

Our consistency results are expressed in terms of the following distributional quantities. For $i \in [d]$, define the *(t,i)-boundary* of $\mathcal{X}$ as $\partial_{t,i}(\mathcal{X}) \triangleq \{x : \{x + te_i, x - te_i\} \not\subset \mathcal{X}\}$. The smaller the mass $\mu(\partial_{t,i}(\mathcal{X}))$ at the boundary, the better we approximate $\|f_i'\|_{1,\mu}$.

The second type of quantity is $\epsilon_{t,i} \triangleq \sup_{x \in \mathcal{X}, s \in [-t,t]} |f_i'(x) - f_i'(x + se_i)|$.

Since μ has continuous density on $\mathcal{X}$ and ∇f is uniformly continuous on $\mathcal{X} + B(0,\tau)$, we automatically have $\mu(\partial_{t,i}(\mathcal{X})) \xrightarrow{t \to 0} 0$ and $\epsilon_{t,i} \xrightarrow{t \to 0} 0$.

4.2 Main theorem

Our main theorem bounds the error in estimating each norm $\|f_i'\|_{1,\mu}$ with $\nabla_{n,i}$. The main technical hurdles are in handling the various sample inter-dependencies introduced by both the estimates $f_{n,h}(X)$ and the events $A_{n,i}(X)$, and in analyzing the estimates at the boundary of $\mathcal{X}$.

Theorem 11 *Let $t + h \leq \tau$, and let $0 < \delta < 1$. There exist $C = C(\mu, K(\cdot))$ and $N = N(\mu)$ such that the following holds with probability at least $1 - 2\delta$. Define $A(n) \triangleq Cd \cdot \log(n/\delta) \cdot C_Y^2(\delta/2n) \cdot \sigma_Y^2 / \log^2(n/\delta)$. Let $n \geq N$, we have for all $i \in [d]$:*

$$\left| \nabla_{n,i} - \left\| f_i' \right\|_{1,\mu} \right| \leq \frac{1}{t} \left(\sqrt{\frac{A(n)}{nh^d}} + h \cdot \sum_{i \in [d]} \left| f_i' \right|_{sup} \right) + 2 \left| f_i' \right|_{sup} \left(\sqrt{\frac{\ln 2d/\delta}{n}} + \mu\left(\partial_{t,i}(\mathcal{X})\right) \right) + \epsilon_{t,i}.$$

The bound suggests to set t in the order of h or larger. We need t to be small in order for $\mu(\partial_{t,i}(\mathcal{X}))$ and $\epsilon_{t,i}$ to be small, but t needs to be sufficiently large (relative to h) for the estimates $f_{n,h}(X + te_i)$ and $f_{n,h}(X - te_i)$ to differ sufficiently so as to capture the variation in f along e_i.

The theorem immediately implies consistency for $t \xrightarrow{n \to \infty} 0$, $h \xrightarrow{n \to \infty} 0$, $h/t \xrightarrow{n \to \infty} 0$, and $(n/\log n) h^d t^2 \xrightarrow{n \to \infty} \infty$. This is satisfied for many settings, for example $t \propto \sqrt{h}$ and $h \propto 1/\log n$.

4.3 Proof of Theorem 11

The main difficulty in bounding $\left| \nabla_{n,i} - \|f_i'\|_{1,\mu} \right|$ results from certain dependencies between random quantities: both quantities $f_{n,h}(X)$ and $A_{n,i}(X)$ depend not just on $X \in \mathbf{X}^n$, but on other samples

in $\mathbf{X}^n$, and thus introduce inter-dependencies between the estimates $\Delta_{t,i} f_{n,h}(X)$ for different points X in the sample $\mathbf{X}^n$.

To handle these dependencies, we carefully decompose $\left| \nabla_{n,i} - \|f_i'\|_{1,\mu} \right|$, $i \in [d]$, starting with:

$$\left| \nabla_{n,i} - \left\| f_i' \right\|_{1,\mu} \right| \leq \left| \nabla_{n,i} - \mathbb{E}_n \left| f_i'(X) \right| \right| + \left| \mathbb{E}_n \left| f_i'(X) \right| - \left\| f_i' \right\|_{1,\mu} \right|. \tag{5}$$

The following simple lemma bounds the second term of (5).

Lemma 12 *With probability at least $1 - \delta$, we have for all $i \in [d]$,*

$$\left| \mathbb{E}_n \left| f_i'(X) \right| - \left\| f_i' \right\|_{1,\mu} \right| \leq \left| f_i' \right|_{sup} \cdot \sqrt{\frac{\ln 2d/\delta}{n}}.$$

Proof Apply a Chernoff bound, and a union bound on $i \in [d]$. ■

Now the first term of equation (5) can be further bounded as

$$\begin{aligned} \left| \nabla_{n,i} - \mathbb{E}_n \left| f_i'(X) \right| \right| &\leq \left| \nabla_{n,i} - \mathbb{E}_n \left| f_i'(X) \right| \cdot \mathbf{1}\{A_{n,i}(X)\} \right| + \mathbb{E}_n \left| f_i'(X) \right| \cdot \mathbf{1}\left\{ \bar{A}_{n,i}(X) \right\} \\ &\leq \left| \nabla_{n,i} - \mathbb{E}_n \left| f_i'(X) \right| \cdot \mathbf{1}\{A_{n,i}(X)\} \right| + \left| f_i' \right|_{\sup} \cdot \mathbb{E}_n \mathbf{1}\left\{ \bar{A}_{n,i}(X) \right\}. \end{aligned} \tag{6}$$

We will bound each term of (6) separately.

The next lemma bounds the second term of (6). It is proved in the appendix. The main technicality in this lemma is that, for any X in the sample $\mathbf{X}^n$, the event $\bar{A}_{n,i}(X)$ depends on other samples in $\mathbf{X}^n$.

Lemma 13 *Let $\partial_{t,i}(\mathcal{X})$ be defined as in Section (4.1.3). For $n \geq n(\mu)$, with probability at least $1 - 2\delta$, we have for all $i \in [d]$,*

$$\mathbb{E}_n \mathbf{1}\left\{ \bar{A}_{n,i}(X) \right\} \leq \sqrt{\frac{\ln 2d/\delta}{n}} + \mu\left(\partial_{t,i}(\mathcal{X})\right).$$

It remains to bound $|\nabla_{n,i} - \mathbb{E}_n |f_i'(X)| \cdot \mathbf{1}\{A_{n,i}(X)\}|$. To this end we need to bring in f through the following quantities:

$$\widetilde{\nabla}_{n,i} \triangleq \mathbb{E}_n \left[\frac{|f(X + te_i) - f(X - te_i)|}{2t} \cdot \mathbf{1}\{A_{n,i}(X)\} \right] = \mathbb{E}_n \left[\Delta_{t,i} f(X) \cdot \mathbf{1}\{A_{n,i}(X)\} \right]$$

and for any $x \in \mathcal{X}$, define $\tilde{f}_{n,h}(x) \triangleq \mathbb{E}_{\mathbf{Y}^n | \mathbf{X}^n} f_{n,h}(x) = \sum_i w_i(x) f(x_i)$.

The quantity $\widetilde{\nabla}_{n,i}$ is easily related to $\mathbb{E}_n |f_i'(X)| \cdot \mathbf{1}\{A_{n,i}(X)\}$. This is done in Lemma 14 below. The quantity $\tilde{f}_{n,h}(x)$ is needed when relating $\nabla_{n,i}$ to $\widetilde{\nabla}_{n,i}$.

Lemma 14 *Define $\epsilon_{t,i}$ as in Section (4.1.3). With probability at least $1 - \delta$, we have for all $i \in [d]$,*

$$\left| \widetilde{\nabla}_{n,i} - \mathbb{E}_n \left| f_i'(X) \right| \cdot \mathbf{1}\{A_{n,i}(X)\} \right| \leq \epsilon_{t,i}.$$

Proof We have $f(x+te_i) - f(x-te_i) = \int_{-t}^{t} f_i'(x+se_i)\,ds$ and therefore

$$2t\left(f_i'(x) - \epsilon_{t,i}\right) \leq f(x+te_i) - f(x-te_i) \leq 2t\left(f_i'(x) + \epsilon_{t,i}\right).$$

It follows that $\left|\frac{1}{2t}\left|f(x+te_i) - f(x-te_i)\right| - \left|f_i'(x)\right|\right| \leq \epsilon_{t,i}$, therefore

$$\left|\widetilde{\nabla}_{n,i} - \mathbb{E}_n\left|f_i'(X)\right| \cdot \mathbf{1}\{A_{n,i}(X)\}\right| \leq \mathbb{E}_n\left|\frac{1}{2t}\left|f(x+te_i) - f(x-te_i)\right| - \left|f_i'(x)\right|\right| \leq \epsilon_{t,i}.$$

■

It remains to relate $\mathbf{W}_i$ to $\widetilde{\nabla}_{n,i}$. We have

$$\begin{aligned}
2t\left|\nabla_{n,i} - \widetilde{\nabla}_{n,i}\right| =& 2t\left|\mathbb{E}_n(\Delta_{t,i}f_{n,h}(X) - \Delta_{t,i}f(X)) \cdot \mathbf{1}\{A_{n,i}(X)\}\right| \\
\leq& 2\max_{s\in\{-t,t\}} \mathbb{E}_n\left|f_{n,h}(X+se_i) - f(X+se_i)\right| \cdot \mathbf{1}\{A_{n,i}(X)\} \\
\leq& 2\max_{s\in\{-t,t\}} \mathbb{E}_n\left|f_{n,h}(X+se_i) - \tilde{f}_{n,h}(X+se_i)\right| \cdot \mathbf{1}\{A_{n,i}(X)\} \qquad (7) \\
&+ 2\max_{s\in\{-t,t\}} \mathbb{E}_n\left|\tilde{f}_{n,h}(X+se_i) - f(X+se_i)\right| \cdot \mathbf{1}\{A_{n,i}(X)\}. \qquad (8)
\end{aligned}$$

We first handle the bias term (8) in the next lemma which is given in the appendix.

Lemma 15 (Bias) *Let $t+h \leq \tau$. We have for all $i \in [d]$, and all $s \in \{t,-t\}$:*

$$\mathbb{E}_n\left|\tilde{f}_{n,h}(X+se_i) - f(X+se_i)\right| \cdot \mathbf{1}\{A_{n,i}(X)\} \leq h \cdot \sum_{i\in[d]} \left|f_i'\right|_{sup}.$$

The variance term in (7) is handled in the lemma below. The proof is given in the appendix.

Lemma 16 (Variance terms) *There exist $C = C(\mu, K(\cdot))$ such that, with probability at least $1-2\delta$, we have for all $i \in [d]$, and all $s \in \{-t,t\}$:*

$$\mathbb{E}_n\left|f_{n,h}(X+se_i) - \tilde{f}_{n,h}(X+se_i)\right| \cdot \mathbf{1}\{A_{n,i}(X)\} \leq \sqrt{\frac{Cd\cdot\log(n/\delta)C_Y^2(\delta/2n)\cdot\sigma_Y^2}{n(h/2)^d}}.$$

The next lemma summarizes the above results:

Lemma 17 *Let $t+h \leq \tau$ and let $0<\delta<1$. There exist $C = C(\mu, K(\cdot))$ such that the following holds with probability at least $1-2\delta$. Define $A(n) \triangleq Cd\cdot\log(n/\delta)\cdot C_Y^2(\delta/2n)\cdot\sigma_Y^2/\log^2(n/\delta)$. We have*

$$\left|\nabla_{n,i} - \mathbb{E}_n\left|f_i'(X)\right| \cdot \mathbf{1}\{A_{n,i}(X)\}\right| \leq \frac{1}{t}\left(\sqrt{\frac{A(n)}{nh^d}} + h\cdot\sum_{i\in[d]}\left|f_i'\right|_{sup}\right) + \epsilon_{t,i}.$$

Proof Apply lemmas 14, 15 and 16, in combination with equations 7 and 8. ■

To complete the proof of Theorem 11, apply lemmas 17 and 12 in combination with equations 5 and 6.

5. Experimental Evaluation of the GW Approach

We have so far derived GW based on the theoretical principles of Section 2, namely that performance improvements are possible if data coordinates are weighted according to the coordinate-wise variation of the unknown f, and if f varies unevenly across coordinates. In this section, we verify these theoretical principles empirically on various real-world datasets. The code and all the data sets used in these experiments are publicly available at `http://goo.gl/bCfS78`

We consider kernel, k-NN and SVM (support vector) approaches on a variety of controlled (artificial) and real-world datasets. We emphasize that our goal is to demonstrate the benefits of GW in improving the performance of these successful and popular procedures on a wide range of datasets. We do not aim to beat results that may have been obtained on these data using procedures other than kernel, k-NN and SVM approaches, since this is not required for a practical validation of our theoretical results. We also note that, throughout the experiments, we only retain the numerical attributes in each data set, and discard all the categorical attributes. Therefore, our reported prediction errors on some datasets might differ from others reported in the literature at large.

Parameter settings and general comments: Recall that, for the GW approach, we might set the components $\mathbf{W}_i$ of the metric ρ to $\nabla_{n,i}^q$. The exponent q (cf. Remark 2.2) is a parameter left open by our theoretical analysis. In our experiments, we explore the choices $q=1$, as in Kpotufe and Boularias (2012), and $q=2$ which serves to further emphasize the difference in importance between coordinates.

The resulting performance of the GW approach depends on the parameters used to learn $\nabla_{n,i} \triangleq \mathbb{E}_n\left[\Delta_{t,i}f_{n,h}(X)\cdot\mathbf{1}\{A_{n,i}(X)\}\right]$. These are the bandwidth h used in the estimate $f_{n,h}(X)$ and the parameter t in $\Delta_{t,i}f_{n,h}(X) \triangleq \left|f_{n,h}(X+te_i) - f_{n,h}(X-te_i)\right|/2t$. In the majority of experiments (reported in the main body of the paper) we tune h, but we don't tune t and simply set $t = h/2$ as a rule of thumb. This results in faster training time, and although not optimal, still results in significant performance gains for the various regression and classification procedures where GW is used to preprocess the data. If in addition we properly tune t, the observed performance gains are even more significant as reported in Tables 4 and 5 of the Appendix.

We emphasize that the GW approach is computationally cheap: it only adds to training time since it only involves pre-processing the data. No significant difference is observed in estimation time, i.e. in computing regression or classification estimates using the preprocessed data vs using the original data. In fact estimation time can even be smaller after preprocessing since GW can act as an approximate dimension-reduction given the sparsity in the data. The average prediction times are reported in Table 6 of the appendix.

Our experiments are divided as follows. First we show the attainable performance gains by using GW for regression, then we show that GW works well also for classification. At the end of the section we explore the tradeoffs between feature selection and feature weigthing.

5.1 Regression experiments

In this section, we present experiments on several real-world regression data sets. We compare the performances of both kernel regression and k-NN regression in the Euclidean metric space and in the learned gradient weights metric space.

5.1.1 DATA DESCRIPTION

The first two data sets describe the dynamics of 7 degrees of freedom of robotic arms, Barrett WAM and SARCOS (Nguyen-Tuong et al., 2009; Nguyen-Tuong and Peters, 2011). The input points are 21-dimensional and correspond to samples of the positions, velocities, and accelerations of the 7 joints. The output points correspond to the torque of each joint. The far joints (1, 5, 7) correspond to different regression problems and are the only results reported. As expected, results for other joints were found to be similarly good.

Another data set describes the probabilities of achieving successful grasping actions performed by a robot on different piles of objects (Boularias et al., 2014a,b). Each data point describes one grasping action performed at a particular location on the surface of a pile of objects. The objects are mostly rocks and rubble with unknown and irregular shapes. An input point is a 150-dimensional vector and corresponds to a patch of a depth image obtained by projecting the robotic hand on the scene. The output is a value between 0 and 1.

The other data sets are taken from the UCI repository (Frank and Asuncion, 2012) and from (Torgo, 2012). The concrete strength data set (Concrete Strength) contains 8-dimensional input points, describing age and ingredients of concrete, the output points are the compressive strength. The wine quality data set (Wine Quality) contains 11-dimensional input points corresponding to the physicochemistry of wine samples, the output points are the wine quality. The ailerons data set (Ailerons) is taken from the problem of flying a F16 aircraft. The 5-dimensional input points describe the status of the aeroplane, while the goal is to predict the control action on the ailerons of the aircraft. The housing data set (Housing) concerns the task of predicting housing values in areas of Boston, the input points are 13-dimensional. The Parkinson's Telemonitoring data set (Parkison's) is used to predict the clinician's Parkinson's disease symptom score using biomedical voice measurements represented by 21-dimensional input points. We also consider a telecommunication problem (Telecom), wherein the 47-dimensional input points and the output points describe the bandwidth usage in a network.

5.1.2 EXPERIMENTAL SETUP

For all data sets, we normalize each coordinate with its standard deviation from the training data. To learn the metric, we set h by cross-validation on half the training points, and we set $t = h/2$ for all data sets. Note that in practice we might want to also tune t in the range of h for even better performance than reported here. The event $A_{n,i}(X)$ is set to reject the gradient estimate $\Delta_{n,i} f_{n,h}(X)$ at X if no sample contributed to one of the estimates $f_{n,h}(X \pm te_i)$.

In each experiment, we compare kernel regression in the Euclidean metric space (KR) and in the learned metric space with gradient weights (KR-ρ) and with squared gradient weights (KR-ρ^2), where we use a box kernel for the three methods. Similar comparisons are made using k-NN, k-NN-ρ and k-NN-ρ^2. All methods are implemented using a fast neighborhood search procedure, namely the cover-tree of (Beygelzimer et al., 2006), and we also report in the supplementary material the average prediction times so as to confirm that, on average, time-performance is not affected by using the metric.

The parameter k in k-NN, k-NN-ρ, k-NN-ρ^2, and the bandwidth in KR, KR-ρ, KR-ρ^2 are learned by cross-validation on half of the training points. We try the same range of k (from 1 to $5 \log n$) for the three k-NN methods (k-NN, k-NN-ρ). We try the same range of bandwidth/space-diameter h (a grid of size 0.02 from 1 to 0.02) for the three KR methods (KR, KR-ρ, KR-ρ^2): this

is done efficiently by starting with a log search to quickly reduce the search space, followed by a grid search on the resulting smaller range.

Table 1 shows the normalized Mean Square Errors (nMSE) where the MSE on the test set is normalized by variance of the test output. We use 1000 training points in the robotic, Telecom, Parkinson's, and Ailerons data sets, and 2000 training points in Wine Quality, 730 in Concrete Strength, and 300 in Housing. We used 2000 test points in all of the problems, except for Concrete, 300 points, Housing, 200 points, and Robot Grasping, 10000 points. Averages over 10 random experiments are reported. For the larger data sets (SARCOS, Ailerons, Telecom, Grasping) we also report the behavior of the algorithms, with and without metric, as the training size n increases (Figure 4).

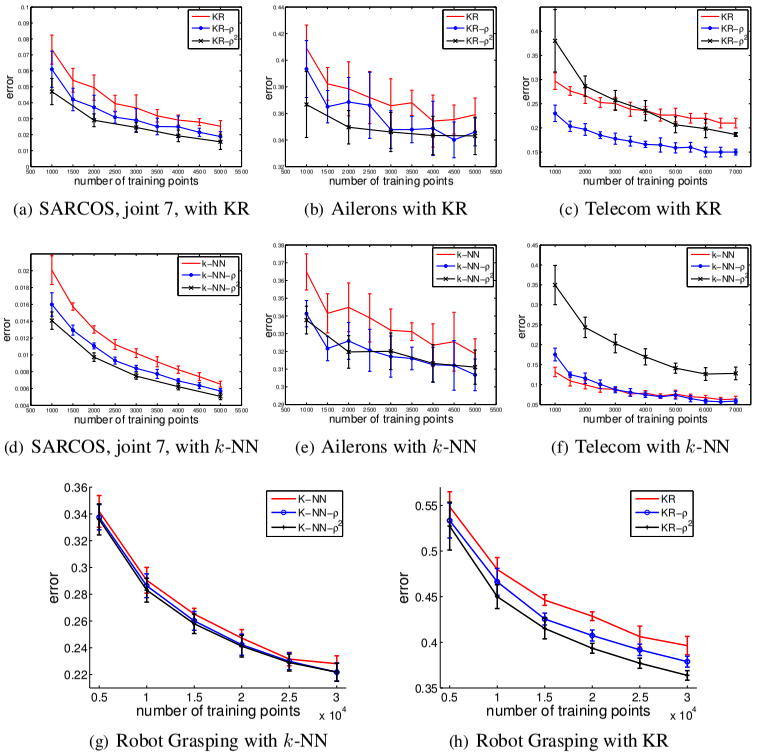
	Barrett 1	Barrett 5	SARCOS 1	SARCOS 5	Housing
KR-unnormalized	0.98 ± 0.03	0.90 ± 0.03	0.16 ± 0.02	0.32 ± 0.03	0.73 ± 0.09
KR-normalized	0.50 ± 0.02	0.50 ± 0.03	0.16 ± 0.02	0.14 ± 0.02	0.37 ± 0.08
KR-normalized-ρ	0.38 ± 0.03	0.35 ± 0.02	0.14 ± 0.02	$\mathbf{0.12} \pm 0.01$	0.25 ± 0.06
KR-normalized-ρ^2	$\mathbf{0.30} \pm 0.03$	$\mathbf{0.28} \pm 0.03$	$\mathbf{0.11} \pm 0.02$	$\mathbf{0.12} \pm 0.01$	$\mathbf{0.21} \pm 0.04$
	Concrete Strength	Wine Quality	Telecom	Ailerons	Parkinson's
KR-unnormalized	0.45 ± 0.03	0.92 ± 0.01	0.23 ± 0.02	0.43 ± 0.02	0.75 ± 0.09
KR-normalized	0.42 ± 0.05	0.75 ± 0.03	0.30 ± 0.02	0.40 ± 0.02	0.38 ± 0.03
KR-normalized-ρ	0.37 ± 0.03	0.75 ± 0.02	$\mathbf{0.23} \pm 0.02$	0.39 ± 0.02	$\mathbf{0.34} \pm 0.03$
KR-normalized-ρ^2	$\mathbf{0.31} \pm 0.02$	$\mathbf{0.72} \pm 0.02$	0.37 ± 0.08	$\mathbf{0.37} \pm 0.02$	$\mathbf{0.34} \pm 0.02$

	Barrett 1	Barrett 5	SARCOS 1	SARCOS 5	Housing
k-NN-unnormalized	0.96 ± 0.01	0.80 ± 0.03	0.11 ± 0.01	0.19 ± 0.01	0.53 ± 0.08
k-NN-normalized	0.41 ± 0.02	0.40 ± 0.02	0.08 ± 0.01	0.08 ± 0.01	0.28 ± 0.09
k-NN-normalized-ρ	0.29 ± 0.01	0.30 ± 0.02	0.07 ± 0.01	0.07 ± 0.01	0.22 ± 0.06
k-NN-normalized-ρ^2	$\mathbf{0.21} \pm 0.02$	$\mathbf{0.23} \pm 0.01$	$\mathbf{0.06} \pm 0.01$	$\mathbf{0.06} \pm 0.01$	$\mathbf{0.18} \pm 0.03$
	Concrete Strength	Wine Quality	Telecom	Ailerons	Parkinson's
k-NN-unnormalized	0.40 ± 0.07	0.88 ± 0.01	0.15 ± 0.02	0.42 ± 0.02	0.63 ± 0.04
k-NN-normalized	0.40 ± 0.04	0.73 ± 0.04	$\mathbf{0.13} \pm 0.02$	0.37 ± 0.01	0.22 ± 0.01
k-NN-normalized-ρ	0.38 ± 0.03	0.72 ± 0.03	0.17 ± 0.02	$\mathbf{0.34} \pm 0.01$	$\mathbf{0.20} \pm 0.01$
k-NN-normalized-ρ^2	$\mathbf{0.31} \pm 0.06$	$\mathbf{0.70} \pm 0.01$	0.34 ± 0.05	$\mathbf{0.34} \pm 0.01$	$\mathbf{0.20} \pm 0.01$

Table 1: Normalized mean square prediction errors show that, in almost all cases, gradient weights improve accuracy in practice. The top three tables are for KR vs KR-ρ and KR-ρ^2, the bottom three for k-NN vs k-NN-ρ and k-NN-ρ^2. k-NN-unnormalized and KR-unnormalized refer to k-NN and KR used on unnormalized data. For all the other methods, data vectors are normalized by dividing each data vector by the standard deviation of the training data in each dimension.

5.1.3 DISCUSSION OF RESULTS

From the results in Table 1 we see that virtually on all data sets the metric helps improve the performance of the distance based-regressor even though we did not tune t to the particular problem (remember $t = h/2$ for all experiments). The only exception is for Telecom with k-NN. We noticed



(a) SARCOS, joint 7, with KR (b) Ailerons with KR (c) Telecom with KR

(d) SARCOS, joint 7, with k-NN (e) Ailerons with k-NN (f) Telecom with k-NN

(g) Robot Grasping with k-NN (h) Robot Grasping with KR

Figure 4: Plotting regression error as a function of the number of training points shows that gradient weights lead to a clear improvement even for small sample sizes, indicating that our estimator of $\|f_i'\|_{1,\mu}$ produces useful results even from relatively few samples.

that the Telecom data set has a lot of outliers and this probably explains the discrepancy, besides the fact that we did not attempt to tune t (see Trivedi et al. (2014) for experiments where t is additionally tuned, and where GW clearly outperforms the baselines for the Telecom dataset).

For the baseline methods (kernel and k-NN), we report both the errors when the data is unnormalized, and when the features are normalized by their standard deviation. For all other methods, the data is normalized. This is to show that, while variance normalization is a first step in reducing the error of the baseline, such errors are further reduced, significantly, through our approach of weighing by estimated derivatives.

Also notice that the error of k-NN is already low for small sample sizes, making it harder to outperform. However, as shown in Figure 4, for larger training sizes k-NN-ρ gains on k-NN. We also note that methods using squared gradient weights (k-NN-ρ^2 and KR-ρ^2) achieved a better performance compared to other methods. The only exception here is also Telecom, where the non-squared gradient weights yield a lower prediction error. The rest of the results in Figure 4 where we vary n are self-descriptive: gradient weighting clearly improves the performance of the distance-based regressors.

Finally, we note that the average prediction times (reported in the supplementary material) is nearly the same for all the methods. Last, remember that the metric can be learned online at the cost of only $2d$ times the average kernel estimation time reported.

5.2 Classification experiments

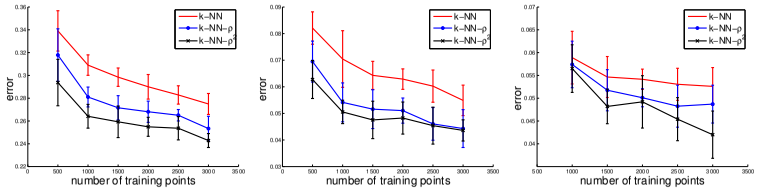
5.2.1 DATA DESCRIPTION

We tested the gradient weights method on six different data sets taken from the UCI repository (Frank and Asuncion, 2012) and from the LIBSVM website (Fan, 2012). The covertype data set contains predictions of binary forest cover types from cartographic features given by 10 real variables among 54 other variables. This data set originally consists of seven different cover types, but only the two largest categories are selected for binary classification. The MAGIC gamma data set consists of 10 features and predicts the registration of high energy gamma particles in a ground-based atmospheric Cherenkov gamma telescope. The IJCNN data set contains predictions of one binary output from four different time series, described by 10 categorical variables and 12 real variables. The shuttle data set contains 9 numerical attributes. The original data set has seven different categories, but for binary classification we merged all the classes into one class, except the first class which corresponds to approximately 80% of all the data. The page blocks data set predicts whether a block in a given document is a text block using 10 real-valued features. We also consider the thyroid data set where the problem is to determine whether a given patient is hypothyroid. There are three different output classes in this data set, the condition of a patient is described by 6 real variables and 15 categorical variables.

5.2.2 EXPERIMENTAL SETUP

The setup for the classification experiments is similar to the one used in the regression experiments. For all data sets, we normalize each coordinate with its standard deviation from the training data. We use the training data to compute the gradient weights. Parameter t is set proportionally to the difference between the minimum and the maximum values of each feature to account for the differences between features scales that remain after normalization. One can also consider using the learned gradient weights to set t and to re-estimate the gradient weights again, in a repeated iterative process. The probability $P(C_i|\mathbf{x})$ of each class C_i, used for calculating the feature weights, is estimated by weighted k-NN with Gaussian kernel.

In each experiment, we compare a k nearest neighbor classifier in the Euclidean metric space (k-NN), the learned metric space with gradient weights (k-NN-ρ), and a metric space in which gradient weights have been squared (k-NN-ρ^2). Analogous results have been obtained using an ϵ-NN classifier (ϵ-NN, ϵ-NN-ρ, ϵ-NN-ρ^2) which uses all training samples within an ϵ-ball around the test point, rather than the k nearest neighbors. Parameters k and ϵ have been set by cross-validation with half the training points. As in the regression experiments, k and ϵ are found by using a log search,



(a) Covertype with k-NN (b) IJCNN with k-NN (c) Thyroid with k-NN

Figure 5: As in the regression case, classification benefits from gradient weights even if they were estimated from relatively few samples.

	Covertype	IJCNN	MAGIC Gamma	Shuttle	Page Blocks
k-NN error	0.29±0.01	0.0629±0.0038	0.1884±0.0118	0.0031±0.0011	0.0346±0.0044
k-NN-ρ error	0.27±0.01	0.0510±0.0047	**0.1858**±0.0086	**0.0019**±0.0010	0.0338±0.0052
k-NN-ρ^2 error	**0.25**±0.01	**0.0482**±0.0060	0.1875±0.0092	**0.0019**±0.0008	**0.0337**±0.0046
ϵ-NN error	0.28±0.01	0.0841±0.0061	0.1773±0.0080	0.0126±0.0026	0.0450±0.0035
ϵ-NN-ρ error	0.26±0.01	0.0716±0.0059	0.1741±0.0069	0.0109±0.0028	0.0427±0.0031
ϵ-NN-ρ^2 error	**0.25**±0.01	**0.0651**±0.0041	**0.1721**±0.0073	**0.0093**±0.0020	**0.0404**±0.0032

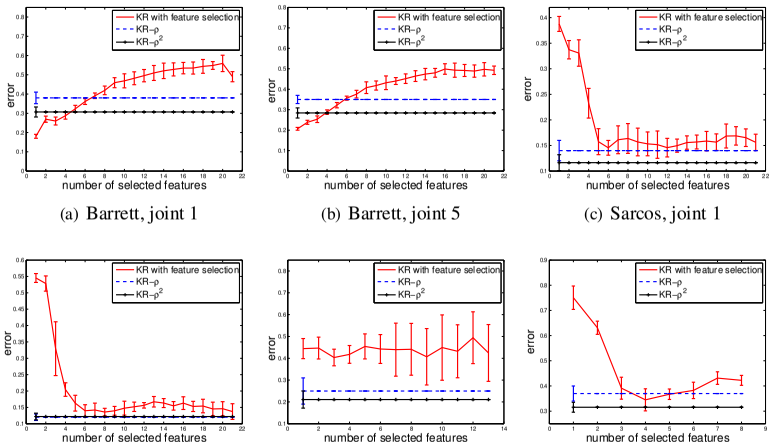
Table 2: Error rates with and without gradient weights show that they improve classification accuracy, especially when they are used in their squared form. This is true for two different distance-based classifiers, k-NN (top), and ϵ-NN (bottom).

followed by a linear search in a smaller interval. All classification experiments are performed using 2000 points for testing and up to 3000 points for learning. Averages over 10 random experiments are reported. The purpose of varying the size of training data is to report the performance as a function of the number of training points (Figure 5). Table 2 shows the classification error rates of the different methods.

5.2.3 DISCUSSION OF RESULTS

From the results in Table 2, we see that the gradient weights metric improves the performance of the two different distance-based classifiers, k-NN and ϵ-NN. We also notice that the squared gradient weights perform better than the non-squared weights in almost all cases. The only exception is the MAGIC Gamma data set where the performance of the different classifiers seems unaffected by the choice of the metric. This is due to the fact that the gradient weights in this particular problem are nearly the same for every feature, as shown in the appendix.

Figure 5 shows improvements for GW over the baseline even with relatively small sample sizes. The improvement of GW over the baseline decreases with larger training sizes except in the Thyroid data set where the advantage of GW is more pronounced with a larger training size. Recall that, from the theoretical insights developed in Section 2, we only expect large improvements in a sample size



(a) Barrett, joint 1 (b) Barrett, joint 5 (c) Sarcos, joint 1

(d) Sarcos, joint 5 (e) Housing (f) Concrete Strength

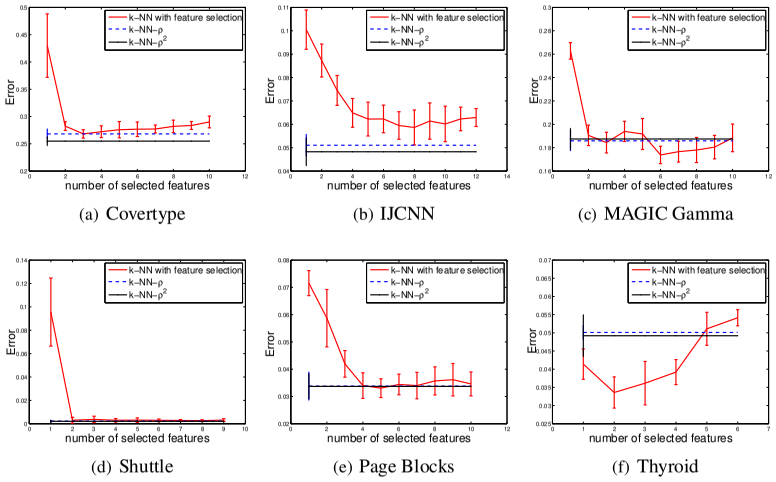
Figure 6: Kernel regression with feature selection, versus GW. On these datasets, feature weighting tends to outperform feature selection, especially when all features happen to be relevant.

regime depending on the problem, so the same behavior (as for the other data sets) is likely for the Thyroid case if we had larger samples to work with.

5.3 Feature selection vs feature weighting

How does feature weighting compare with feature selection? Feature weighting, as done with GW, has the obvious advantage of avoiding the combinatorial problem of having to select a subset of features, and gets rid of the ill-defined problem of selecting a good *importance threshold* for feature selection. Another less obvious advantage of feature weighting is that we do not lose much in performance if it so happens that all features are relevant, since weighting uses all features. However, feature selection reduces dimension and therefore variance, and would be expected to be the better option if some features are much more important than all others (i.e. f is nearly sparse); how much do we lose by feature weighting in this sort of situation, i.e. does the computational advantage of weighting justify its use? This of course depends on problem-specific costs, but we will attempt here to better understand the differences between the two approaches.

We compare our proposed GW method with feature selection, where features are added in order of importance, i.e. we first add the features most correlated with the output Y. The same training sets are used to compute the gradient weigths and feature correlations with Y, under similar time complexity. Keep in mind that feature selection here requires the additional complexity of comparing the performance gain from various combinations of features; here for simplicity we would just compare n ordered subsets, although more subsets might be compared in practice. An alternative to



(a) Covertype (b) IJCNN (c) MAGIC Gamma

(d) Shuttle (e) Page Blocks (f) Thyroid

Figure 7: Experiments on k-NN classification with feature selection versus GW. Again, GW generally performs better, while little is lost when feature selection is preferable.

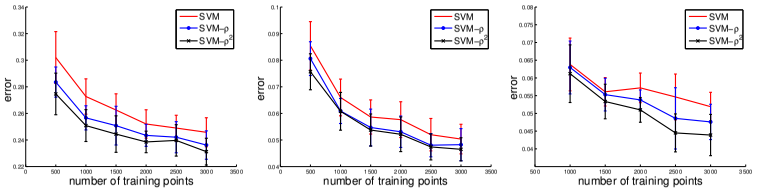
comparing combinations of features is to use some kind of thresholding, but as mentioned earlier, it is often unclear how to properly threshold feature importance.

Figure 6 shows that overall GW with kernel regression is competitive with the more expensive feature selection, and even often achieves better performance in those situations where all features happen to be relevant (Sarcos, Housing, Concrete). Similar results are achieved for k-NN regression, and are reported in the supplementary material.

Notice that, in those cases where feature selection performs best (Barret datasets), its gain over GW is smallest when we used the squared version $\nabla^2_{n,i}$ of GW (denoted KR-ρ^2 in the figure). This is because squaring emphasizes the differences in variability of f between features and is thus closer to feature selection.

Figure 7 repeats the same experiments in the case of classification with k-NN. Here the performance of feature selection depends more crucially on the number of selected features, and can be bad in most cases if too few features are selected. GW outperforms feature selection in most cases but the Thyroid dataset. However, even in the Thyroid case, the advantage is small, less than 2%.

While a more elaborate feature selection procedure might produce a bigger advantage over feature weighting, the computational cost might be even higher. Feature weighting with GW offers a cheap alternative which moreover often outperforms natural feature selection routines such as the one just discussed. However, we do not claim that our method can replace sophisticated feature selection schemes that are specialized for high-dimensional applications in *parametric* settings (e.g., (Zhou et al., 2014)); we emphasize however that there are few alternatives to the present approach for nonparametric settings; one popular such alternative, Relieff, is discussed below.



(a) Covertype with SVM (b) IJCNN with SVM (c) Thyroid with SVM

Figure 8: Classification error rates of a support vector machine suggest that pre-multiplying features by their gradient weight also improves performance of that classifier.

6. Discussion

In this section we further discuss the ideas presented so far by addressing some interesting questions that arise naturally. In particular we will consider the applicability of the GW approach outside the context of distance-based learning methods in the next section. In the following section we take a look at existing heuristics for feature weighting and show how, although not by design, their success might be explained by the theoretical intuition developed in the present work. We finish the section and the paper with open questions and a discussion of future directions.

6.1 Feature Weighting for Support Vector Machines

Even though the intuition for our method has been developed for distance-based regressors and classifiers, we have found empirically that pre-processing features by multiplying them with their corresponding gradient weight also improves the performance of other popular classifiers, such as support vector machines (SVMs). This is demonstrated in Figure 8, which reports results on the same classification tasks as in Figure 5, but uses an SVM instead of a k-NN classifier. We have used a Gaussian kernel, and we have cross-validated its kernel bandwidth h separately for the Euclidean and the learned metric space on half the training points. As before, h is found by a log search, followed by a linear search.

6.2 Relation to the Relief family of Heuristics

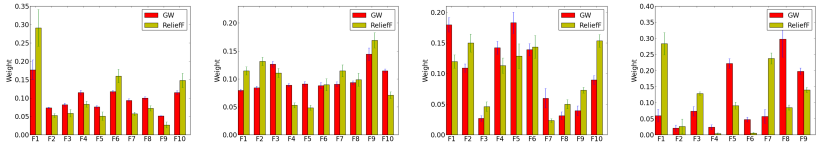
Early approaches for feature selection have exhaustively enumerated all possible subsets of features, or have employed heuristics to reduce the search space. In this section we relate our GW approach to the Relief approach, which from its introduction in Kira and Rendell (1992), has gained much popularity and evolved into a larger family of related heuristics (Kononenko, 1994; Robnik-Šikonja and Kononenko, 2003).

The success of Relief is due to its ease of implementation, computational efficiency in avoiding the combinatorial problems of earlier approaches, and more importantly its good performance on real-world problems. While Relief has mostly been used for binary feature selection, some variants were also used for feature weighting (Wettschereck et al., 1997; Sun, 2007), similar to our proposed GW approach.

	Covertype	MAGIC Gamma	Shuttle	Page Blocks
Gradient Weights	0.0113±0.0067	0.0050±0.0039	**0.0006**±0.0011	**0.0007**±0.0026
ReliefF	**0.0229**±0.0075	**0.0147**±0.0072	-0.0019±0.0024	-0.0019±0.0049

Table 3: Comparing the improvement in classification error over the k-NN baseline when using squared gradient weights or ReliefF shows that none of the two methods dominates the other one. Negative numbers indicate cases where ReliefF led to increased errors.

While Relief and our GW approach have similar practical benefits, GW is grounded in the theoretical insights developed earlier in this work. Corollary 8 allows us to theoretically understand the conditions under which GW improves regression rates in a minimax sense, opening up potential directions for further development of feature weighting methods. To the best of our knowledge, no such theoretical results are available for Relief although various works have provided theoretical interpretation (e.g. (Sun, 2007)) without actually analyzing the direct effect of Relief weights on regression or classification convergence rates. We will argue here that the theoretical intuition developed in this work helps explain some of the success of Relief: the weights computed by Relief, similar to those of GW, are generally correlated with the coordinate-wise variation of the unknown regression function f.



(a) Covertype (b) MAGIC Gamma (c) Page Blocks (d) Shuttle

Figure 9: Comparing GW feature weights (i.e. $\nabla_{n,i}$) and those of ReliefF, averaged over 10 random repeats of the experiment. There is a clear correlation in most cases but the right-most.

First, upon computing weights for several real-world data sets, we can observe empirically that the weights assigned by ReliefF (Kononenko, 1994) are often correlated with those computed by GW which are by design correlated with the coordinate-wise variation of f (Figure 9).

Both methods, when used for feature-weighting, yield similar improvement in classification error rates, as shown in Table 3. For GW we use $\mathbf{W}_i = \nabla^2_{n,i}$, i.e. coordinates are weighted by $\nabla_{n,i}$; correspondingly, we pre-multiply features by their ReliefF weights as done in Wettschereck et al. (1997). For a fair comparison, the number k of neighbors used in ReliefF was found by cross-validation on the training data, just as in our method. None of the two methods dominates the other on all examples. However, unlike the weights found by ReliefF, gradient weights never led to an average increase in error.

We can gain additional insight on the relationship to the Relief family by considering RReliefF, an extension to regression problems (Robnik-Šikonja and Kononenko, 2003). In RReliefF, the

weight $\hat{\mathbf{W}}_\mathbf{i}$ of a feature i is estimated as

$$\hat{\mathbf{W}}_i = \frac{\sum_x \sum_{x' \in N(x)} |f_n(x) - f_n(x')||x_i - x'_i| \text{dis}(x,x')}{\sum_x \sum_{x' \in N(x)} |f_n(x) - f_n(x')| \text{dis}(x,x')} - \frac{\sum_x \sum_{x' \in N(x)} \Big(|x_i - x'_i| \text{dis}(x,x') - |f_n(x) - f_n(x')||x_i - x'_i| \text{dis}(x,x') \Big)}{n - \sum_x \sum_{x' \in N(x)} |f_n(x) - f_n(x')| \text{dis}(x,x')},$$

where x is a training input point, x_i is the ith attribute of x, n is the number of training points, f_n is a k-NN estimate of f, $N(x)$ is the set of k nearest neighbors of x with respect to the Euclidean metric, and dis is a dissimilarity function, defined as $\text{dis}(x,x') = \exp\Big(-\frac{1}{h}\big(\text{rank}(x,x')\big)^2\Big)$ where $\text{rank}(x,x')$ is obtained by ranking the neighbors of x according to their increasing distance from x, and h is some bandwidth. By rearranging the terms of the equation above, we have

$$\begin{aligned}\hat{\mathbf{W}}_i = & \left(\frac{1}{A(f_n)} + \frac{1}{B(f_n)}\right) \sum_x \sum_{x' \in N(x)} |f_n(x) - f_n(x')||x_i - x'_i| \text{dis}(x,x') \\ & - \frac{1}{B(f_n)} \sum_x \sum_{x' \in N(x)} |x_i - x'_i| \text{dis}(x,x') \\ = & \left(\frac{1}{A(f_n)} + \frac{1}{B(f_n)}\right) \hat{\mathbf{W}}_{i,I} + \frac{1}{B(f_n)} \hat{\mathbf{W}}_{i,II},\end{aligned}$$

where

$$\begin{aligned}A(f_n) = & \sum_x \sum_{x' \in N(x)} |f_n(x) - f_n(x')| \text{dis}(x,x'), \\ B(f_n) = & \; n - \sum_x \sum_{x' \in N(x)} |f_n(x) - f_n(x')| \text{dis}(x,x').\end{aligned}$$

Notice that $A(f_n)$ and $B(f_n)$ are global parameters that do not depend on the feature i for which the weight $\hat{\mathbf{W}}_i$ is calculated.

The term $\frac{1}{B(f_n)} \hat{\mathbf{W}}_{i,II}$ can be interpreted as a measure of the spread of the input points around axis i, and has little dependence on the output Y. Moreover this term would generally be negligible: suppose w.l.o.g. that $|Y|$ is normalized to be at most 1, then $B(f_n) = \Omega(n)$ so the term goes to 0.

The other term $\left(\frac{1}{A(f_n)} + \frac{1}{B(f_n)}\right) \hat{\mathbf{W}}_{i,I}$ is most important and is correlated with the variation of f_n (hence of f) in direction i. In particular, $\hat{\mathbf{W}}_{i,I}$ has the essential ingredients of an estimator of $\|f'_i\|_{1,\mu}$: it is a weighted average of differences in f_n, where the weights $(|x_i - x'_i| \text{dis}(x,x'))$ are (1) larger for pairs of points aligned along coordinate i (via $|x_i - x'_i|$), and (2) larger for the closest pairs of points (via the dissimilarity $\text{dis}(x,x')$).

In fact the differences between $\hat{\mathbf{W}}_{i,I}$ and our estimator $\nabla_{n,i}$ are simply in the way we accomplish (1) and (2) above: we directly look at pairs of points $(x \pm te_i)$, t-close and aligned with the coordinate i. This is similar to setting the neighborhood $N(x)$ in Relieff to $x \pm te_i$ and using a dissimilarity of the form $\text{dis}(x,x') = \big(2nt^2\big)^{-1}$.

The success of GW is best understood in terms of how they reduce the variance of distance-based regressors while controlling bias, as elucidated in Section 2.2. Even though no such analysis is available for Relief, our results extend the same theoretical intuition to the RReliefF heuristic which also estimates a quantity that is correlated with the coordinate-wise variation of f.

6.3 Final Remarks and Future Directions

We have shown both theoretically and empirically that it is possible to gain significantly in regression and classification performance by weighting features according to the way the unknown regression function f varies along coordinates, provided f does not vary equally across coordinates, which is often the case. We derived a simple procedure to estimate the variation in f and showed its consistency. The present brings up many new questions which we hope would be the subject of future investigation. We list some of these questions below.

The approach results in a two-phase prediction procedure, which however might be iterated into a multiple phase procedure for a potentially better estimation of f. A natural question is how to detect convergence of a multiple phase approach. This is unclear for now, but is worth pursuing since even the simple two-phase approach discussed here works well.

The approach presented in this paper weights coordinates with some power of the estimate $\nabla_{n,i}$ of the gradient norm $\|f_i'\|_{1,\mu}$. Should all coordinates be weighted with the same power of $\nabla_{n,i}$, or is there a better way to weights coordinates according to the way f varies? This question requires refined theoretical understanding of how coordinate-weighting affects particular regression and classification procedures.

The current approach does not take into account the fact that the input X might not be full-dimensional. It is often the case that data lies near lower-dimensional subspaces of $\mathbb{R}^d$. How does one capture the directional variation of f in these cases?

Given the simplicity and success of the approach presented here, we believe even better feature-weigthing approaches are lurking close, and some of the above questions are potential directions for further improvement.

Acknowledgments

A significant part of this work was conducted when the authors were at the Max Planck Institute for Intelligent Systems, Tuebingen, Germany.

Appendix A. Consistency lemmas

We need the following VC result.

Lemma 18 (Relative VC bounds (Vapnik and Chervonenkis, 1971)) *Let $0 < \delta < 1$ and define $\alpha_n = (2d \ln 2n + \ln(4/\delta))/n$.*

Then with probability at least $1-\delta$ over the choice of $\mathbf{X}^n$, all balls $B \in \mathbb{R}^d$ satisfy

$$\mu(B) \leq \mu_n(B) + \sqrt{\mu_n(B)\alpha_n} + \alpha_n.$$

Proof [Lemma 13] Let $\bar{A}_i(X)$ denote the event that $\min_{s\in\{-t,t\}} \mu(B(X+se_i, h/2)) < 3\alpha_n$. By Lemma 18, with probability at least $1-\delta$, $\forall i \in [d]$, $\bar{A}_{n,i}(X) \implies \bar{A}_i(X)$ so that $\mathbb{E}_n \mathbf{1}\{\bar{A}_{n,i}(X)\} \leq \mathbb{E}_n \mathbf{1}\{\bar{A}_i(X)\}$.

Using a Chernoff bound, followed by a union bound on $[d]$, we also have with probability at least $1-\delta$ that $\mathbb{E}_n \mathbf{1}\{\bar{A}_i(X)\} \leq \mathbb{E}\mathbf{1}\{\bar{A}_i(X)\} + \sqrt{\ln(2d/\delta)/n}$.

Finally, $\mathbb{E}\mathbf{1}\{\bar{A}_i(X)\} \leq \mathbb{E}\left[\mathbf{1}\{\bar{A}_i(X)\}|X \in \mathcal{X} \setminus \partial_{t,i}(\mathcal{X})\right] + \mu\left(\partial_{t,i}(\mathcal{X})\right)$. The first term is 0 for large n. This is true since, for $x \in \mathcal{X} \setminus \partial_{t,i}(\mathcal{X})$, for all $i \in [d]$ and $s \in \{-t, t\}$, $x + se_i \in \mathcal{X}$, and

therefore $\mu(B(x+se_i, h/2)) \geq C_\mu (h/2)^d \geq 3\alpha_n$ for our setting of h (see Section 3). ■

Proof [Lemma 15] Let $x = X + se_i$. For any $X_i \in \mathbf{X}^n$, let v_i denote the unit vector in direction $(X_i - x)$. We have

$$\begin{aligned}\left|\tilde{f}_{n,h}(x) - f(x)\right| &\leq \sum_i w_i(x) \left|f(X_i) - f(x)\right| = \sum_i w_i(x) \left|\int_0^{\|X_i - x\|} v_i^\top \nabla f(x + sv_i)\, ds\right| \\ &\leq \sum_i w_i(x) \left\|X_i - x\right\| \cdot \max_{x' \in \mathcal{X} + B(0,\tau)} \left\|v_i^\top \nabla f(x')\right\| \leq h \cdot \sum_{i\in[d]} \left|f_i'\right|_{\text{sup}}.\end{aligned}$$

Multiply the l.h.s. by $\mathbf{1}\{A_{n,i}(X)\}$, take the empirical expectation and conclude. ■

The variance (Lemma 16) is handled in a way similar to an analysis of (Kpotufe, 2011) on k-NN regression. The additional technicality in the present result is due to the fact that, unlike in the case of k-NN, the number of points contributing to the estimate (and hence the variance) is not a constant.

Proof [Lemma 16] Assume that $A_{n,i}(X)$ is true, and fix $x = X + se_i$. The following variance bound is quickly obtained:

$$\mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n} \left|f_{n,h}(x) - \tilde{f}_{n,h}(x)\right|^2 \leq \sigma_Y^2 \cdot \sum_{i\in[n]} w_i(x) \leq \sigma_Y^2 \cdot \max_{i\in[n]} w_i(x).$$

Let $\mathbf{Y}^n{}_x$ denote the Y values of samples $X_i \in B(x,h)$, and write $\psi(\mathbf{Y}^n{}_x) \triangleq \left|f_{n,h}(x) - \tilde{f}_{n,h}(x)\right|$. We next relate $\psi(\mathbf{Y}^n{}_x)$ to the above variance.

Let $\mathcal{Y}_\delta$ denote the event that for all $Y_i \in \mathbf{Y}^n$, $|Y_i - f(X_i)| \leq C_Y(\delta/2n) \cdot \sigma_Y$. By definition of $C_Y(\delta/2n)$, the event $\mathcal{Y}_\delta$ happens with probability at least $1 - \delta/2 \geq 1/2$. We therefore have that

$$\begin{aligned}\mathbb{P}_{\mathbf{Y}^n|\mathbf{X}^n}\left(\psi(\mathbf{Y}^n{}_x) > 2\mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n}\psi(\mathbf{Y}^n{}_x) + \epsilon\right) &\leq \mathbb{P}_{\mathbf{Y}^n|\mathbf{X}^n}\left(\psi(\mathbf{Y}^n{}_x) > \mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n,\mathcal{Y}_\delta}\psi(\mathbf{Y}^n{}_x) + \epsilon\right) \\ &\leq \mathbb{P}_{\mathbf{Y}^n|\mathbf{X}^n,\mathcal{Y}_\delta}\left(\psi(\mathbf{Y}^n{}_x) > \mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n,\mathcal{Y}_\delta}\psi(\mathbf{Y}^n{}_x) + \epsilon\right) + \delta/2.\end{aligned}$$

Now, it can be verified that, by McDiarmid's inequality, we have

$$\mathbb{P}_{\mathbf{Y}^n|\mathbf{X}^n,\mathcal{Y}_\delta}\left(\psi(\mathbf{Y}^n{}_x) > \mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n,\mathcal{Y}_\delta}\psi(\mathbf{Y}^n{}_x) + \epsilon\right) \leq \exp\left\{-2\epsilon^2 / C_Y^2(\delta/2n) \cdot \sigma_Y^2 \sum_{i\in[n]} w_i^2(x)\right\}.$$

Notice that the number of possible sets $\mathbf{Y}^n{}_x$ (over $x \in \mathcal{X}$) is at most the n-shattering number of balls in $\mathbb{R}^d$. By Sauer's lemma we know this number is bounded by $(2n)^{d+2}$. We therefore have by a union bound that, with probability at least $1-\delta$, for all $x \in \mathcal{X}$ satisfying $B(x, h/2) \cap \mathbf{X}^n \neq \emptyset$,

$$\begin{aligned}\psi(\mathbf{Y}^n{}_x) &\leq 2\mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n}\psi(\mathbf{Y}^n{}_x) + \sqrt{(d+2)\cdot\log(n/\delta) C_Y^2(\delta/2n)\cdot\sigma_Y^2 \sum_{i\in[n]} w_i^2(x)} \\ &\leq 2\left(\mathbb{E}_{\mathbf{Y}^n|\mathbf{X}^n}\psi^2(\mathbf{Y}^n{}_x)\right)^{1/2} + \sqrt{(d+2)\cdot\log(n/\delta) C_Y^2(\delta/2n)\cdot\sigma_Y^2 \cdot \max_i w_i(x)} \\ &\leq \sqrt{Cd\cdot\log(n/\delta) C_Y^2(\delta/2n)\cdot\sigma_Y^2/n\mu_n((B(x,h/2))},\end{aligned}$$

where the second inequality is obtained by applying Jensen's, and the last inequality is due to the fact that the kernel weights are upper and lower-bounded on $B(x, h/2)$.

Now by Lemma 18, with probability at least $1-\delta$, for all X such that $A_{n,i}(X)$ is true, we have for all $s \in \{-t, t\}$, $3\mu_n((B(x, h/2)) \geq \mu((B(x, h/2)) \geq C_\mu(h/2)^d$. Integrate this into the above inequality, take the empirical expectation and conclude. ■

Appendix B. Properties of the metric space $(\mathcal{X}, \rho)$

B.1 Covering numbers

We assume throughout this section that $\Delta_\mathcal{X} \triangleq \sup_{x,x' \in \mathcal{X}} \|x - x'\| = 1$. Recall Definition 3 of Section 2.2.

We start with the following easily obtained lemma.

Lemma 19 *Consider $R \subset [d]$ such that $\max_{i \notin R} < \min_{i \in R} \mathbf{W}_i$. Define $\rho_R(x, x') \triangleq \sqrt{\min_{i \in R} \mathbf{W}_i} \cdot \sum_{i \in R} \left(x^i - x'^i\right)^2$. For any $x, x' \in \mathcal{X}$,*

$$\rho_R(x, x') \leq \rho(x, x') \leq \kappa_R \rho_R(x, x') + \epsilon_{\not R}/2.$$

Proof We have

$$\begin{aligned}
\rho^2(x, x') &= \sum_{i \in R} \mathbf{W}_i (x^i - x'^i)^2 + \sum_{i \notin R} \mathbf{W}_i (x^i - x'^i)^2 \\
&\leq \kappa_R^2 \cdot \min_{i \in R} \mathbf{W}_i \cdot \left\|x - x'\right\|_R^2 + \max_{i \notin R} \mathbf{W}_i \cdot \|\mathcal{X}\| \\
&\leq \left(\kappa_R \rho_R(x, x') + \epsilon_{\not R}/2\right)^2.
\end{aligned}$$

■

The next lemma bounds ϵ-covering numbers for large ϵ (relative to $\epsilon_{\not R}$).

Lemma 20 (Covering numbers at large scale) *Consider any $R \subsetneq [d]$ such that $\max_{i \notin R} \mathbf{W}_i < \min_{i \in R} \mathbf{W}_i$. Let $\rho(\mathcal{X})$ denote the ρ-diameter of $\mathcal{X}$. For any $\epsilon\rho(\mathcal{X}) \geq \epsilon_{\not R}$, $(\mathcal{X}, \rho)$ can be covered by $C_R \epsilon^{-|R|}$ ρ-balls of radius $\epsilon\rho(\mathcal{X})$, where $C_R \leq C(4\kappa_R)^{|R|}$.*

Proof Let $x, x' \in \mathcal{X}$ and define $\|x - x'\|_R \triangleq \sqrt{\sum_{i \in R} (x^i - x'^i)^2}$. Notice that in the pseudo-metric space $(\mathcal{X}, \|x - x'\|_R)$, every ball B of radius r can be covered by at most $C\epsilon^{-|R|}$ balls of radius ϵr for any $\epsilon > 0$. This is also true for any scaling of $\|x - x'\|_R$, in particular for ρ_R. We next relate the covering numbers of $(\mathcal{X}, \rho)$ to those of $(\mathcal{X}, \rho_R)$.

Let $\mathbf{Z}$ denote an $\epsilon\rho(\mathcal{X})$-packing of $(\mathcal{X}, \rho)$, i.e. $\rho(z, z') > \epsilon\rho(\mathcal{X})$ for all $z, z' \in \mathbf{Z}$. The size of $\mathbf{Z}$ is an upper-bound on the mininum $\epsilon\rho(\mathcal{X})$-cover size of $(\mathcal{X}, \rho)$. We have by Lemma 19, $\rho_R(z, z') > (\epsilon\rho(\mathcal{X}) - \epsilon_{\not R}/2)/\kappa_R \geq (\epsilon\rho(\mathcal{X})/2\kappa_R)$. Thus, the size of $\mathbf{Z}$ is at most that of a $(\epsilon\rho(\mathcal{X})/4\kappa_R)$-cover of $(\mathcal{X}, \rho_R)$. Since the ρ_R-diameter of $\mathcal{X}$ is at most $\rho(R)$, we have $|\mathbf{Z}| \leq C(4\kappa_R)^{|R|}\epsilon^{-|R|}$. ■

Using the above lemma, we can now prove Lemma 4 which bounds covering numbers of the space $(\mathcal{X}, \rho)$ at all scales.

Proof [Lemma 4] The first part of Lemma 4 was obtained in Lemma 20 above. The second part is obtained as follows.

We only consider dyadic values $\epsilon = 2^{-m} < \epsilon_{\not R}/\rho(\mathcal{X})$, since, for nondyadic values of ϵ, the bound on the smallest cover can only be within a constant factor depending on d.

To construct a small $\epsilon\rho(\mathcal{X})$-cover, start with an $\epsilon_{\not R}$-cover Z of $\mathcal{X}$. This has size at most $C_R(\epsilon_{\not R}/\rho(\mathcal{X}))^{-|R|}$ by Lemma 20. Consider any $z \in Z$. The ball $B(z, \epsilon_{\not R})$ has an $\epsilon\rho(\mathcal{X})$-cover of size at most $C'(\epsilon_{\not R}/\epsilon\rho(\mathcal{X}))^d$ by the doubling property of bounded subsets of $\mathbb{R}^d$. The union over $z \in Z$ of the covers of the balls $B(z, \epsilon_{\not R})$ is an $\epsilon\rho(\mathcal{X})$-cover of $\mathcal{X}$, and has size at most $C_R \cdot C'\epsilon^{-r(\epsilon)} \leq C_R(\epsilon_{\not R}/\rho(\mathcal{X}))^{-|R|} \cdot C'(\epsilon_{\not R}/\epsilon\rho(\mathcal{X}))^d$, for some $r(\epsilon)$ defined as in the lemma statement. ■

B.2 Change in Lipschitz constant

Next, Lemma 5 which bounds the change in Lipschitz constant is obtained as follows.

Proof [Lemma 5] Let $x \neq x'$ and $v = (x - x')/\|x - x'\|$. Clearly $v^i \leq \rho(x, x')/(\|x - x'\| \cdot \sqrt{\mathbf{W}_i})$. We have

$$\begin{aligned}
\left|f(x) - f(x')\right| &\leq \int_0^{\|x-x'\|} \left|v^\top \nabla f(x + sv)\right| \, ds \leq \int_0^{\|x-x'\|} \sum_{i \in R} \left|v^i \cdot f_i'(x + sv)\right| \, ds \\
&\leq \sum_{i \in R} \int_0^{\|x-x'\|} \left|v^i\right| \cdot \left|f_i'\right|_{\text{sup}} \leq \sum_{i \in R} \frac{\rho(x, x')}{\sqrt{\mathbf{W}_i}} \left|f_i'\right|_{\text{sup}}.
\end{aligned}$$

■

Appendix C. Asymptotic rate for norm-induced metrics

A norm-induced metric ρ on a space $\mathcal{X} \subset \mathbb{R}^d$ is one where there exists a norm $\bar{\rho} : \mathcal{X} \to \mathbb{R}$ such that for every $x, x' \in \mathcal{X}$, $\rho(x, x') = \bar{\rho}(x - x')$. We show in this section that, for such metrics, a regressor operating on the space $(\mathcal{X}, \rho)$ has worst-case rate of $\Omega\left(n^{-2/(2+d)}\right)$ for large n.

Without loss of generality, let $(\mathcal{X}, \rho)$ have diameter 1. We assume further, throughout the section, that the space $\mathcal{X}$ is compact under ρ.

The main argument consists of showing that ϵ-cover sizes for $(\mathcal{X}, \rho)$ are of the form ϵ^{-d} for sufficiently small ϵ. This is then enough to call on the regression lower-bound result of Kpotufe (2011) to conclude the argument.

Appendix D. Additional experimental results

Dataset	k-NN	k-NN-ρ
Ailerons	0.3364 ± 0.0087	0.3161 ± 0.0058
Concrete	0.2884 ± 0.0311	$\mathbf{0.2040} \pm 0.0234$
Housing	0.2897 ± 0.0632	$\mathbf{0.2389} \pm 0.0604$
Wine	0.6633 ± 0.0119	0.6615 ± 0.0134
Barrett1	0.1051 ± 0.0150	$\mathbf{0.0843} \pm 0.0229$
Barrett5	0.1095 ± 0.0096	$\mathbf{0.0984} \pm 0.0244$
Sarcos1	0.1222 ± 0.0074	$\mathbf{0.0769} \pm 0.0037$
Sarcos5	0.0870 ± 0.0051	0.0779 ± 0.0026
Parkinson	0.3638 ± 0.0443	$\mathbf{0.3181} \pm 0.0477$
TeleComm	0.0864 ± 0.0094	0.0688 ± 0.0074

Table 4: Regression results, with ten random runs per data set. For each method, the values of k as well as t (the bandwidth used to estimate finite differences for calculating the gradients) were set by two fold cross-validation on the training set.

Dataset	k-NN	k-NN–ρ
Cover Type	0.2279 ± 0.0091	0.2135 ± 0.0064
Gamma	0.1775 ± 0.0070	0.1680 ± 0.0075
Page Blocks	0.0349 ± 0.0042	0.0361 ± 0.0048
Shuttle	0.0037 ± 0.0025	0.0024 ± 0.0016
Musk	0.2279 ± 0.0091	0.2135 ± 0.0064
IJCNN	0.0540 ± 0.0061	0.0459 ± 0.0058
RNA	0.1042 ± 0.0063	0.0673 ± 0.0062

Table 5: Classification results, with ten random runs per data set. For each method, the values of k as well as t (the bandwidth used to estimate finite differences for calculating the gradients) were set by two fold cross-validation on the training set.

	Barrett joint 1	Barrett joint 5	SARCOS joint 1	SARCOS joint 5	Housing
KR error	0.50 ± 0.02	0.50 ± 0.03	0.16 ± 0.02	0.14 ± 0.02	0.37 ± 0.08
KR-ρ error	0.38 ± 0.03	0.35 ± 0.02	0.14 ± 0.02	0.12 ± 0.01	0.25 ± 0.06
KR-ρ^2 error	$\mathbf{0.30} \pm 0.03$	$\mathbf{0.28} \pm 0.03$	$\mathbf{0.11} \pm 0.02$	$\mathbf{0.12} \pm 0.01$	$\mathbf{0.21} \pm 0.04$
KR-ρ^3 error	0.18 ± 0.02	0.20 ± 0.01	0.18 ± 0.03	0.14 ± 0.02	0.25 ± 0.03
KR-ρ^4 error	0.17 ± 0.01	0.20 ± 0.01	0.37 ± 0.02	0.20 ± 0.01	0.39 ± 0.08
KR time	0.39 ± 0.02	0.37 ± 0.01	0.28 ± 0.05	0.23 ± 0.03	0.10 ± 0.01
KR-ρ time	0.41 ± 0.03	0.38 ± 0.02	0.32 ± 0.05	0.23 ± 0.02	0.11 ± 0.01
	Concrete Strength	Wine Quality	Telecom	Ailerons	Parkinson's
KR error	0.42 ± 0.05	0.75 ± 0.03	0.30 ± 0.02	0.40 ± 0.02	0.38 ± 0.03
KR-ρ error	0.37 ± 0.03	0.75 ± 0.02	$\mathbf{0.23} \pm 0.02$	0.39 ± 0.02	$\mathbf{0.34} \pm 0.03$
KR-ρ^2 error	$\mathbf{0.31} \pm 0.02$	$\mathbf{0.72} \pm 0.02$	0.37 ± 0.08	$\mathbf{0.37} \pm 0.02$	$\mathbf{0.34} \pm 0.02$
KR-ρ^3 error	0.28 ± 0.04	0.73 ± 0.03	0.57 ± 0.02	0.38 ± 0.02	0.31 ± 0.02
KR-ρ^4 error	0.37 ± 0.04	0.78 ± 0.02	0.54 ± 0.03	0.41 ± 0.01	0.30 ± 0.01
KR time	0.14 ± 0.02	0.19 ± 0.02	0.15 ± 0.01	0.20 ± 0.01	0.30 ± 0.03
KR-ρ time	0.14 ± 0.01	0.19 ± 0.02	0.16 ± 0.01	0.21 ± 0.01	0.30 ± 0.03

	Barrett joint 1	Barrett joint 5	SARCOS joint 1	SARCOS joint 5	Housing
k-NN error	0.41 ± 0.02	0.40 ± 0.02	0.08 ± 0.01	0.08 ± 0.01	0.28 ± 0.09
k-NN-ρ error	0.29 ± 0.01	0.30 ± 0.02	0.07 ± 0.01	0.07 ± 0.01	0.22 ± 0.06
k-NN-ρ^2 error	$\mathbf{0.21} \pm 0.02$	$\mathbf{0.23} \pm 0.01$	$\mathbf{0.06} \pm 0.01$	$\mathbf{0.06} \pm 0.01$	$\mathbf{0.18} \pm 0.03$
k-NN-ρ^3 error	0.11 ± 0.02	0.19 ± 0.01	0.08 ± 0.01	0.05 ± 0.01	0.23 ± 0.03
k-NN-ρ^4 error	0.15 ± 0.01	0.20 ± 0.01	0.37 ± 0.01	0.16 ± 0.01	0.31 ± 0.07
k-NN time	0.21 ± 0.04	0.16 ± 0.03	0.13 ± 0.01	0.13 ± 0.01	0.08 ± 0.01
k-NN-ρ time	0.13 ± 0.04	0.16 ± 0.03	0.14 ± 0.01	0.13 ± 0.01	0.08 ± 0.01
	Concrete Strength	Wine Quality	Telecom	Ailerons	Parkinson's
k-NN error	0.40 ± 0.04	0.73 ± 0.04	$\mathbf{0.13} \pm 0.02$	0.37 ± 0.01	0.22 ± 0.01
k-NN-ρ error	0.38 ± 0.03	0.72 ± 0.03	0.17 ± 0.02	$\mathbf{0.34} \pm 0.01$	$\mathbf{0.20} \pm 0.01$
k-NN-ρ^2 error	$\mathbf{0.31} \pm 0.06$	$\mathbf{0.70} \pm 0.01$	0.34 ± 0.05	0.34 ± 0.01	$\mathbf{0.20} \pm 0.01$
k-NN-ρ^3 error	0.26 ± 0.02	0.71 ± 0.01	0.55 ± 0.03	0.36 ± 0.01	0.22 ± 0.01
k-NN-ρ^4 error	0.38 ± 0.05	0.78 ± 0.01	0.52 ± 0.02	0.45 ± 0.01	0.25 ± 0.01
k-NN time	0.10 ± 0.01	0.15 ± 0.01	0.16 ± 0.02	0.12 ± 0.01	0.14 ± 0.01
k-NN-ρ time	0.11 ± 0.01	0.15 ± 0.01	0.15 ± 0.01	0.11 ± 0.01	0.15 ± 0.01

Table 6: Normalized mean square prediction errors and average prediction time per point (in milliseconds). The top five tables are for KR vs KR-ρ, KR-ρ^2,KR-ρ^3 and KR-ρ^4, the bottom five for k-NN vs k-NN-ρ, k-NN-ρ^2, k-NN-ρ^3 and k-NN-ρ^4.

References

A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbors. *ICML*, 2006.

Abdeslam Boularias, James Andrew Bagnell, and Anthony Stentz. Efficient Optimization for Autonomous Robotic Manipulation of Natural Objects. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 2520–2526, 2014a.

Abdeslam Boularias, James Andrew Bagnell, and Anthony Stentz. Robot Grasping Data Set. http://www.cs.rutgers.edu/~ab1544/data/AAAI2014Data.tar.bz2. Carnegie Mellon University, Robotics Department, 2014b.

K. Clarkson. Nearest-neighbor searching and metric space dimensions. *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice*, 2005.

Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216, 2007.

L. Devroye, L. Gyorfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

Rong-En Fan. LIBSVM Data: Classification, Regression, and Multi-label, 2012. URL http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.

A. Frank and A. Asuncion. UCI machine learning repository. http://archive.ics.uci.edu/ml. University of California, Irvine, School of Information and Computer Sciences, 2012.

Haijie Gu and John Lafferty. Sequential nonparametric regression. *arXiv preprint arXiv:1206.6408*, 2012.

L. Gyorfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution Free Theory of Nonparametric Regression*. Springer, New York, NY, 2002.

W. Härdle and T. Gasser. On robust kernel estimation of derivatives of regression functions. *Scandinavian journal of statistics*, pages 233–240, 1985.

Marc Hoffmann and Oleg Lepski. Random rates in anisotropic regression. *Annals of statistics*, pages 325–358, 2002.

Kenji Kira and Larry A. Rendell. A practical approach to feature selection. In *Proceedings of the Ninth International Workshop on Machine Learning*, ML92, pages 249–256, 1992.

Igor Kononenko. Estimating attributes: Analysis and extensions of RELIEF. In *Proc. European Conf. on Machine Learning*, pages 171–182, 1994.

S. Kpotufe. k-NN Regression Adapts to Local Intrinsic Dimension. *NIPS*, 2011.

S. Kpotufe and A. Boularias. Gradient weights help nonparametric regressors. *NIPS*, 2012.

J. Lafferty and L. Wasserman. Rodeo: Sparse nonparametric regression in high dimensions. *Arxiv preprint math/0506342*, 2005.

Duy Nguyen-Tuong and Jan Peters. Incremental online sparsification for model learning in real-time robot control. *Neurocomputing*, 74(11):1859–1867, 2011.

Duy Nguyen-Tuong, Matthias W. Seeger, and Jan Peters. Model learning with local gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009.

M Nusbaum. Optimal filtration of a function of many variables in white gaussian noise. *PROB. INFO. TRANSMISSION.*, 19(2):105–111, 1983.

Philippe Rigollet and Alexandre Tsybakov. Exponential screening and optimal rates of sparse estimation. *The Annals of Statistics*, 39(2):731–771, 2011.

Marko Robnik-Šikonja and Igor Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Machine learning*, 53(1-2):23–69, 2003.

L. Rosasco, S. Villa, S. Mosci, M. Santoro, and A. Verri. Nonparametric sparsity and regularization. *http://arxiv.org/abs/1208.2572*, 2012.

Shai Shalev-shwartz, Yoram Singer, and Andrew Y. Ng. Online and batch learning of pseudometrics. In *ICML*, pages 743–750. ACM Press, 2004.

C. J. Stone. Optimal rates of convergence for non-parametric estimators. *Ann. Statist.*, 8:1348–1360, 1980.

C. J. Stone. Optimal global rates of convergence for non-parametric estimators. *Ann. Statist.*, 10: 1340–1353, 1982.

Yijun Sun. Iterative RELIEF for feature weighting: Algorithms, theories, and applications. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(6):1035–1051, 2007.

Luis Torgo. Regression datasets. http://www.liaad.up.pt/~ltorgo. University of Porto, Department of Computer Science, 2012.

Shubhendu Trivedi, Jialei Wang, Samory Kpotufe, and Gregory Shakhnarovich. A consistent estimator of the expected gradient outerproduct. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 819–828, 2014.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their expectation. *Theory of probability and its applications*, 16:264–280, 1971.

Kilian Q. Weinberger and Gerald Tesauro. Metric learning for kernel regression. *Journal of Machine Learning Research - Proceedings Track*, 2:612–619, 2007.

Dietrich Wettschereck, David W. Aha, and Takao Mohri. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 11:273–314, 1997.

Bo Xiao, Xiaokang Yang, Yi Xu, and Hongyuan Zha. Learning distance metric for regression by semidefinite programming with application to human age estimation. In *Proceedings of the 17th ACM international conference on Multimedia*, pages 451–460, 2009.

Yingbo Zhou, Utkarsh Porwal, Ce Zhang, Hung Q Ngo, Long Nguyen, Christopher Ré, and Venu Govindaraju. Parallel feature selection inspired by group testing. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3554–3562. Curran Associates, Inc., 2014.

Journal of Machine Learning Research 17 (2016) 1-21 Submitted 11/13; Revised 10/15; Published 4/16

A Closer Look at Adaptive Regret

Dmitry Adamskiy ADAMSKIY@CS.RHUL.AC.UK
Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK

Wouter M. Koolen WMKOOLEN@CWI.NL
Centrum Wiskunde & Informatica
Science Park 123, 1098XG Amsterdam, The Netherlands

Alexey Chernov A.CHERNOV@BRIGHTON.AC.UK
School of Computing, Engineering and Mathematics, University of Brighton
Moulsecoomb, Brighton, BN2 4GJ, UK

Vladimir Vovk VOVK@CS.RHUL.AC.UK
Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK

Editor: Manfred Warmuth

Abstract

For the prediction with expert advice setting, we consider methods to construct algorithms that have low adaptive regret. The adaptive regret of an algorithm on a time interval $[t_1, t_2]$ is the loss of the algorithm minus the loss of the best expert over that interval. Adaptive regret measures how well the algorithm approximates the best expert locally, and so is different from, although closely related to, both the classical regret, measured over an initial time interval $[1, t]$, and the tracking regret, where the algorithm is compared to a good sequence of experts over $[1, t]$.

We investigate two existing intuitive methods for deriving algorithms with low adaptive regret, one based on specialist experts and the other based on restarts. Quite surprisingly, we show that both methods lead to the same algorithm, namely Fixed Share, which is known for its tracking regret. We provide a thorough analysis of the adaptive regret of Fixed Share. We obtain the exact worst-case adaptive regret for Fixed Share, from which the classical tracking bounds follow. We prove that Fixed Share is optimal for adaptive regret: the worst-case adaptive regret of any algorithm is at least that of an instance of Fixed Share.

Keywords: online learning, adaptive regret, Fixed Share, specialist experts

1. Introduction

This paper deals with the prediction with expert advice setting. Nature generates outcomes step by step. At every step Learner tries to predict the outcome. Then the actual outcome is revealed and the quality of Learner's prediction is measured by a loss function.

No assumptions are made about the nature of the data. Instead, at every step Learner is presented with the predictions of a pool of experts and he may base his predictions on these. The goal of Learner in the classical setting is to guarantee small regret, that is, to suffer cumulative loss that is not much larger than that of the best (in hindsight) expert from the pool. Several classical algorithms exist for this task, including the Aggregating Algorithm (Vovk, 1990) and the Exponentially Weighted Forecaster (Cesa-Bianchi and Lugosi, 2006). In the standard logarithmic loss game the regret incurred by those algorithms when competing with N experts is at most $\ln N$, independent of the number of steps.

A common extension of the framework takes into account the fact that the best expert could change with time. In this case we may be interested in competing with the best *sequence* of experts from the pool. Known algorithms for this task include Fixed Share (Herbster and Warmuth, 1998) and Mixing Past Posteriors (Bousquet and Warmuth, 2002).

In this paper we focus on the related task of obtaining small *adaptive* regret, a notion first considered by Littlestone and Warmuth (1994) and later studied by Hazan and Seshadhri (2009). The adaptive regret of an algorithm on a time interval $[t_1, t_2]$ is the loss that the algorithm accumulates there, minus the loss of the best expert for that interval:

$$R_{[t_1,t_2]} \coloneqq L_{[t_1,t_2]} - \min_n L^n_{[t_1,t_2]} \,. \tag{1}$$

The goal is now to ensure small adaptive regret on all intervals simultaneously. Note that adaptive regret was defined by Hazan and Seshadhri (2009) with a maximum over intervals, but we need the fine-grained dependence on the endpoint times to be able to prove matching upper and lower bounds.

Our results. The contribution of our paper is threefold.

1. We study two constructions to get adaptive regret algorithms from existing classical regret algorithms. The first one is a simple construction proposed by Freund et al. (1997) and slightly generalised by Chernov and Vovk (2009) that involves so called specialists (sleeping experts), and the second one uses restarts, as proposed by Hazan and Seshadhri (2009). Although conceptually dissimilar, we show that both constructions yield the Fixed Share algorithm with a time-varying switching rate.

2. We compute the exact worst-case adaptive regret of Fixed Share. We re-derive the tracking regret bounds from these adaptive regret bounds, showing that the latter are in fact more fundamental.

3. We show that Fixed Share is the optimal algorithm for adaptive regret, in the sense that the worst-case adaptive regret of any candidate algorithm is at least that of an instance of Fixed Share.

Here is a sneak preview of the adaptive bounds we obtain, presented in a slightly relaxed form for simplicity. The refined statement can be found in Theorem 4 below. In the logarithmic loss game for each of the following adaptive regret bounds there is an algorithm satisfying it, simultaneously for all intervals $[t_1, t_2]$:

$$\ln N + \ln t_2 \,, \tag{2a}$$
$$\ln N + \ln t_1 + \ln\ln t_2 + 2 \,, \tag{2b}$$
$$\ln N + 2\ln t_1 + 1 \,, \tag{2c}$$

where $\ln\ln 1$ is interpreted as 0.

Protocol 1 Mix-loss prediction

for $t = 1, 2, \ldots$ **do**
  Learner announces probability vector $\boldsymbol{u}_t \in \triangle_N$
  Reality announces vector $\boldsymbol{\ell}_t \in (-\infty, \infty]^N$ of expert losses
  Learner suffers loss $\ell_t := -\ln \sum_n u_t^n e^{-\ell_t^n}$
end for

Outline. The structure of the paper is as follows. In Section 2 we give the description of the protocol and review standard algorithms. In Section 3 we study two intuitive ways of obtaining adaptive regret algorithms from classical algorithms. We show that curiously both resulting algorithms turn out to be Fixed Share. In Section 4 we study in detail the adaptive regret of Fixed Share and establish its optimality.

2. Setup

We phrase our results in the setting defined in Protocol 1, which, for lack of a standard name, we call *mix loss*. We choose this fundamental setting because it is universal, in the sense that many other common settings reduce to it. For example probability forecasting, sequential investment and data compression are straightforward instances (Cesa-Bianchi and Lugosi, 2006).[1] In addition, mix loss is the baseline for the wider class of *mixable loss functions*, which includes e.g. square loss (Vovk, 2001). Classical (entire $[1, t]$ interval) regret upper bounds transfer from mix loss to mixable losses almost by definition, and the same reasoning extends to adaptive regret bounds (see Appendix A). In addition, mix-loss methods and upper bounds carry over in a modular way (via the individual-sequence versions of Hoeffding-type bounds, e.g. by Cesa-Bianchi et al. 2012) to non-mixable games, which include the Hedge setting (Freund and Schapire, 1997) and Online Convex Optimisation (Zinkevich, 2003). The number N of experts is fixed throughout the paper.

Let us review the specialisation for our setup of two standard algorithms.[2] The *Aggregating Algorithm*, or AA, by Vovk (1998) predicts[3] in trial t with

$$u_t^n := \frac{e^{-\sum_{s<t} \ell_s^n}}{\sum_j e^{-\sum_{s<t} \ell_s^j}}, \tag{3a}$$

which we may also maintain incrementally using the update rule

$$u_{t+1}^n = \frac{u_t^n e^{-\ell_t^n}}{\sum_j u_t^j e^{-\ell_t^j}}. \tag{3b}$$

1. Namely, if ℓ_t^n contains the negative log returns of stock n over trading round t, the mix loss is the negative log return of the portfolio $\boldsymbol{u}_t$. Similarly, if ℓ_t^n contains the negative log likelihoods that probability model n assign to the t-th outcome, then the mix loss is the negative log likelihood of the model average $\boldsymbol{u}_t$.
2. The algorithms we review maintain weights on experts, and originally come with a strategy for subsequently issuing predictions by aggregating the experts' predictions. For mix loss the predictions are abstracted away and an algorithm is evaluated just by its weights.
3. The AA can be parametrised by a prior distribution. As we only need the uniform prior in this section we specialised to that case immediately. The same holds for Fixed Share below.

For this algorithm the classical regret bound states that for each expert n

$$\sum_{t=1}^T \ell_t - \sum_{t=1}^T \ell_t^n \leq \ln N \tag{4}$$

(here and below $\infty - \infty$ is interpreted as 0; i.e., Learner never feels regret w.r.t. an expert who suffers infinite loss). Note that the AA is minimax for classical mix-loss regret since regret $\geq \ln N$ can be inflicted on any algorithm already in the first round.

The second algorithm, *Fixed Share* by Herbster and Warmuth (1998), requires a sequence of switching rates $\alpha_2, \alpha_3, \ldots$. Intuitively, α_t is the probability of a switch to a different expert in the sequence of "best-at-the-step" experts between trials $t-1$ and t. Like the AA, FS starts with uniform weights $u_1^n = 1/N$. The weights are now updated as

$$u_{t+1}^n := \frac{\alpha_{t+1}}{N-1} + \left(1 - \frac{N}{N-1}\alpha_{t+1}\right) \frac{u_t^n e^{-\ell_t^n}}{\sum_j u_t^j e^{-\ell_t^j}}. \tag{5}$$

The intuition behind this expression is that if an expert's normalized weight is w and each expert redistributes a fraction α of his weight uniformly to the other experts, his resulting weight will become $\frac{\alpha}{N-1}(1-w) + (1-\alpha)w = \frac{\alpha}{N-1} + (1 - \frac{N}{N-1}\alpha)w$. We see that the AA is the special case when all α_t are 0 (on the other hand, Fixed Share is a special case of the AA for a certain infinite set of experts as mentioned by Vovk 1999). The tracking regret bound by Herbster and Warmuth (1998) for Fixed Share with constant $\alpha_t = \alpha$ switching rate states that for any reference sequence $n_1, \ldots, n_T$ of experts with m blocks (and hence $m-1$ switches)

$$\sum_{t=1}^T \ell_t - \sum_{t=1}^T \ell_t^{n_t} \leq \ln N + (m-1)\ln(N-1) - (m-1)\ln\alpha - (T-m)\ln(1-\alpha). \tag{6}$$

We will see later (Lemma 11 below) that the interesting values of α_t are in the range $[0, \frac{N-1}{N}]$. Intuitively, a value $\alpha_t > \frac{N-1}{N}$ corresponds to assigning larger weights to poor experts, and this always hurts the worst-case adaptive regret (in the borderline case $\alpha = \frac{N-1}{N}$, (5) becomes $u_{t+1}^n := \frac{1}{N}$). We will also set

$$\alpha_1 := \frac{N-1}{N}; \tag{7}$$

since the algorithm only involves $\alpha_2, \alpha_3, \ldots$ this causes no harm (but simplifies some formulas). Having introduced the standard classical and tracking regret algorithms, we now turn to adaptive regret.

3. Intuitive Algorithms with Low Adaptive Regret

Two methods have been proposed in the literature that can be used to obtain adaptive regret bounds: specialists (sleeping experts) (Freund et al., 1997) and restarts (Hazan and Seshadhri, 2009). We discuss both and show that each of them yields Fixed Share with a particular choice of time-dependent switching rate α_t.

3.1 Specialist Experts

To discuss the first method we need a simple extension of the mix-loss prediction protocol to the case of *specialist experts*, who are absent at some steps ("are asleep"). At the beginning of each round t the subset $A_t \subseteq \{1,\dots,N\}$ of experts who are awake is revealed, and the other experts are said to be asleep. The algorithm is required to assign probabilities only to the experts who are awake, and its loss is now defined by the formula $\ell_t := -\ln \sum_{n\in A_t} u_t^n e^{-\ell_t^n}$. The *specialist AA* is the extension of the AA to specialists. Like the AA it maintains weights on all experts, starting from some prior $\boldsymbol{u}_1$. Each round, it predicts by conditioning the current weights $\boldsymbol{u}_t$ on the set A_t of experts who are awake, thus assigning to such an expert n weight $u_t^n / \sum_{j\in A_t} u_t^j$. After observing the losses the weight of each expert who is awake, $n \in A_t$, is updated multiplicatively as $u_{t+1}^n := u_t^n e^{\ell_t - \ell_t^n}$ and the weight of each sleeping expert $n \notin A_t$ stays put at $u_{t+1}^n := u_t^n$.

The AA and specialist AA are related in a useful manner: Chernov and Vovk (2009) obtain the specialist AA from the AA by imagining that all sleeping experts suffer the same loss as the algorithm. This observation immediately leads to a relativised analogue of regret bound (4). The specialist AA guarantees, for each specialist n,

$$\sum_{t\le T: n\in A_t} \ell_t - \sum_{t\le T: n\in A_t} \ell_t^j \;\le\; \ln N. \tag{8}$$

The loss of expert n is defined only during the rounds when he is awake. This bound tells us that the cumulative loss of the specialist AA incurred during those rounds does not exceed the cumulative loss of expert n by much.

We now turn to obtaining adaptive regret bounds for the vanilla expert setting by running the specialist AA on imaginary (virtual) sleeping experts of our own design.

3.1.1 Specialist Experts

One way of getting an adaptive algorithm is the following. We create a pool of virtual experts. For each real expert n and time t, we include a virtual expert that sleeps during the first $t-1$ trials, and subsequently predicts as expert n from trial t onward. The specialist regret (in the sense of 8) w.r.t. this virtual expert on $[1,T]$ is the same as the adaptive regret w.r.t. the real expert n on $[t,T]$. The natural idea is to feed all those virtual experts into the existing algorithm capable of obtaining good classical regret, the specialist AA. For fixed t_2, the uniform prior on wake-up time $t_1 \le t_2$ and expert n this would lead to adaptive regret $\ln(Nt_2)$. It turns out that the same holds even without knowledge of t_2.

At first glance, it is very inefficient, even in the case of a finite horizon T, to maintain weights of TN specialists. However, we do not need to, since we may merge the weights of all specialists who are awake and associated to the same real expert, resulting in Algorithm 1. To verify that this algorithm is correct, denote this merged (unnormalised) weight in trial t by v_t^n for each real expert n. The merged (unnormalised) weight v_{t+1}^n of this real expert n in the next trial $t+1$ consists of the prior weight, denoted $p(t+1)$, of the newly awaken virtual specialist plus v_t^n, the sum of the weights of the previously awaken specialists, each multiplied by the same factor $e^{\ell_t - \ell_t^n}$ (as they were all awake). Thus we can update the sum directly, and this is reflected by our update rule.

Algorithm 1 Adaptive Aggregating Algorithm

Input: Prior nonnegative weights $p(t)$, $t = 1,2,\dots$, with $p(1) > 0$
$v_1^n := p(1)$, $n = 1,\dots,N$
for $t = 1,2,\dots$ **do**
  Play weights $u_t^n := \frac{v_t^n}{\sum_{j=1}^N v_t^j}$
  Read the experts losses ℓ_t^n, $n = 1,\dots,N$
  Set $v_{t+1}^n := p(t+1) + v_t^n \frac{e^{-\ell_t^n}}{\sum_{j=1}^N u_t^j e^{-\ell_t^j}}$, $n = 1,\dots,N$
end for

Note that for simplicity, we have taken the (unnormalised) priors on experts and wake-up times independent, i.e.

$$p^{(n,t)} \;=\; p(t)\,.$$

(There is no need for the prior weights $p^{(n,t)}$ to normalise, as the predictions are normalised explicitly.)

Now we will see that Algorithm 1 turns out to be Fixed Share with variable switching rate. In the rest of this section we derive this. Let $P(t) = \sum_{s=1}^t p(s)$.

Fact 1 *The update step of Algorithm 1 preserves the following: for all $t \ge 1$*

$$\sum_n v_t^n = \sum_n \sum_{s\le t} p(s) = NP(t)\,.$$

Proof This follows immediately from expanding the one-step update rule:

$$\begin{aligned}
\sum_n v_{t+1}^n &= \sum_n p(t+1) + \sum_n v_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \sum_n p(t+1) + \sum_n v_t^n \frac{e^{-\ell_t^n}}{\sum_k \frac{v_t^k}{\sum_n v_t^n} e^{-\ell_t^k}} \\
&= Np(t+1) + \sum_n v_t^n \overset{\text{Induction}}{=} NP(t+1)\,. \qquad \blacksquare
\end{aligned}$$

We now show that Algorithm 1 can be seen as Fixed Share (and vice versa).

Lemma 2 *Suppose the probabilities $\alpha_t \in [0, \frac{N-1}{N}]$ of a Fixed Share switch before trial t and the prior weights $p(t)$ of a specialist waking up in trial t in Algorithm 1 satisfy*

$$p(t) \;=\; \frac{\frac{N}{N-1}\alpha_t}{\prod_{s=2}^t (1 - \frac{N}{N-1}\alpha_s)}$$

(where we use the convention 7) or, equivalently,

$$\alpha_t \;=\; \frac{N-1}{N} \frac{p(t)}{\sum_{s=1}^t p(s)}\,.$$

Then the two algorithms output identical predictions.

Proof Let us rewrite the update step of Algorithm 1 for the normalised weights.

$$\begin{aligned} u_{t+1}^n &= \frac{v_{t+1}^n}{\sum_j v_{t+1}^j} = \frac{p(t+1)}{NP(t+1)} + \frac{1}{NP(t+1)} v_t^n \frac{e^{-\ell_t^n}}{\sum_j u_t^j e^{-\ell_t^j}} \\ &= \frac{p(t+1)}{NP(t+1)} + \frac{1}{NP(t+1)} NP(t) u_t^n \frac{e^{-\ell_t^n}}{\sum_j u_t^j e^{-\ell_t^j}} \\ &= \frac{\alpha_{t+1}}{N-1} + \frac{P(t+1) - p(t+1)}{P(t+1)} u_t^n \frac{e^{-\ell_t^n}}{\sum_j u_t^j e^{-\ell_t^j}} \\ &= \frac{\alpha_{t+1}}{N-1} + \left(1 - \frac{N}{N-1}\alpha_{t+1}\right) u_t^n \frac{e^{-\ell_t^n}}{\sum_j u_t^j e^{-\ell_t^j}}. \end{aligned}$$

We see that the weight update is the update of the Fixed Share algorithm with variable switching rate α_t. ■

The idea to use specialist experts for obtaining adaptive bounds was introduced by Freund et al. (1997). There a virtual specialist is created for every interval $[t_1, t_2]$ which leads to redundancy and suboptimal bounds. Their adaptive regret bounds include a term which exceeds $2 \ln t_2$ whereas our bounds (2) have at most a single $\ln t_2$.

3.2 Restarts

A second intuitive method to obtain adaptive regret bounds, called Follow the Leading History (FLH), was introduced by Hazan and Seshadhri (2007, 2009).[4] One starts with a base algorithm that ensures low classical regret. FLH then obtains low adaptive regret by restarting a copy of this base algorithm at each trial, and aggregating the predictions of these copies. To get low adaptive regret w.r.t. N experts[5], it is natural to take the AA as the base algorithm. We now show that FLH with this choice equals Fixed Share with switching rate $\alpha_t = \frac{N-1}{Nt}$.

For each n, s and $t \geq s$, let $p_t^{n|s}$ denote the weight allocated to expert n by the copy of the AA started at time s. By definition $p_s^{n|s} = 1/N$, and these weights evolve according to (3b). We denote by p_t^s the weight allocated by FLH in trial $t \geq s$ to the copy of the AA started at time s. Hazan and Seshadhri (2009) define these weights as follows. Initially $p_1^1 = 1$ and subsequently

$$p_{t+1}^{t+1} = \frac{1}{t+1} \quad \text{and} \quad p_{t+1}^s = \left(1 - p_{t+1}^{t+1}\right) \frac{p_t^s e^{-\left(-\ln \sum_n p_t^{n|s} e^{-\ell_t^n}\right)}}{\sum_{r=1}^t p_t^r e^{-\left(-\ln \sum_n p_t^{n|r} e^{-\ell_t^n}\right)}} \quad \text{for all } 1 \leq s \leq t.$$

4. Here we present the version of Follow the Leading History with the best performancee guarantee. This version is called FLH1 by Hazan and Seshadhri (2007). It can be recovered from FLH by Hazan and Seshadhri (2009) by omitting the pruning step, which considerably improves computational efficiency at the cost of predictive performance. Alghough such tradeoffs are not the focus of our paper, they are of great practical significance. We refer to György et al. (2012) for an in-depth analysis.
5. More broadly, for any exp-concave loss function FLH can upgrade classic regret bounds for any baseline algorithm to their adaptive analogues, resulting in efficient adaptive regret algorithms for continuous online optimisation.

We now show that this construction is a reparametrisation of Fixed Share. In fact, this is true for any choice of the restart probabilities p_t^t.

Lemma 3 *For mix loss, FLH with the AA as the base algorithm issues the same predictions as Fixed Share with learning rate* $\alpha_t = \frac{N-1}{N} p_t^t$.

Proof We prove by induction on t that the FS and FLH weights coincide:

$$u_t^n = \sum_{s=1}^t p_t^{n|s} p_t^s.$$

The base case $t = 1$ is obvious. For the induction step we expand

$$\begin{aligned} \sum_{s=1}^{t+1} p_{t+1}^{n|s} p_{t+1}^s &= \sum_{s=1}^t p_{t+1}^{n|s} p_{t+1}^s + p_{t+1}^{t+1}/N \\ &= \left(1 - p_{t+1}^{t+1}\right) \sum_{s=1}^t \left(\frac{p_t^{n|s} e^{-\ell_t^n}}{\sum_n p_t^{n|s} e^{-\ell_t^n}} \frac{p_t^s \left(\sum_n p_t^{n|s} e^{-\ell_t^n}\right)}{\sum_{r=1}^t p_t^r \left(\sum_n p_t^{n|r} e^{-\ell_t^n}\right)} \right) + \frac{1}{N} p_{t+1}^{t+1} \\ &= \left(1 - p_{t+1}^{t+1}\right) \frac{\sum_{s=1}^t p_t^s p_t^{n|s} e^{-\ell_t^n}}{\sum_{r=1}^t \sum_n p_t^r p_t^{n|r} e^{-\ell_t^n}} + \frac{1}{N} p_{t+1}^{t+1} \\ &\overset{\text{Induction}}{=} \left(1 - p_{t+1}^{t+1}\right) \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}} + \frac{1}{N} p_{t+1}^{t+1} = u_{t+1}^n, \end{aligned}$$

and find the Fixed Share update equation (5) for switching rate $\alpha_t = \frac{N-1}{N} p_t^t$. ■

Discussion. This unification points out that the Fixed Share algorithm can be viewed/implemented in three different ways. Depending on the situation, one of these may be more attractive. For example, Hazan and Seshadhri (2009) show that the FLH viewpoint scales up naturally to continuous optimization, and Luo and Schapire (2015) use our specialists viewpoint to design parameterless adaptive regret algorithms for the Hedge setting.

4. The Adaptive Regret of Fixed Share

We have seen in the previous section that both intuitive approaches to obtain algorithms with low adaptive regret result in Fixed Share. We take this convergence to mean that Fixed Share is the most fundamental adaptive algorithm. The tracking regret for Fixed Share is already well-studied. In Section 4.1 we thoroughly analyse the adaptive regret of Fixed Share. We obtain the worst-case adaptive regret for mix loss. This result implies the known tracking regret bounds. Then in Section 4.2 we characterise the achievable (by means of any algorithm) bounds on worst-case adaptive regret. We prove an information-theoretic lower bound for mix loss that must hold for any algorithm, and which is tight for Fixed Share. We show that the Pareto optimal bounds are exactly the Fixed Share bounds. This establishes Fixed Share as *the* answer for adaptive regret. Finally, in Section 4.3, we investigate the possibility of improving the adaptive regret for all "late" intervals by completely forgoing the regret guarantees on "early" intervals. We conclude that this is basically impossible.

4.1 The Exact Worst-case Mix-loss Adaptive Regret for Fixed Share

Define the *worst-case adaptive regret* on $[t_1, t_2]$ to be the supremum of (1) over all data sequences (cf. Definition 7 below). In this section we first compute the exact worst-case adaptive regret of Fixed Share with arbitrary switching rate $\alpha_t \in [0, \frac{N-1}{N}]$. Then we obtain certain regret bounds of interest, including the tracking regret bound, for particular choices of α_t.

Theorem 4 *The worst-case adaptive regret of Fixed Share with $\alpha_t \in [0, \frac{N-1}{N}]$ and with N experts on interval $[t_1, t_2]$ equals*

$$-\ln\left(\frac{\alpha_{t_1}}{N-1}\prod_{t=t_1+1}^{t_2}(1-\alpha_t)\right) \tag{9}$$

with the convention (7).

Proof The proof consists of two parts. First we claim that the worst-case data for the interval $[t_1, t_2]$ in the setting of Protocol 1 is rather simple: on the interval there is one *good* expert (all others get infinite losses) and on the single trial before the interval (if $t_1 > 1$) this expert suffers infinite loss while others do not. The proof of this fact can be found in Appendix B.

Now we will compute the regret on this data. The regret of Fixed Share on the interval $[t_1, t_2]$ is $-\ln$ of the product of the weights put on the good expert (say, n) on this interval:

$$R^{\mathrm{FS}}_{[t_1,t_2]} = -\ln \prod_{t_1 \le t \le t_2} u^n_t \,.$$

It is straightforward to derive u^n_t from (5):

$$u^n_{t_1} \;=\; \frac{\alpha_{t_1}}{N-1} \qquad \text{and} \qquad u^n_t \;=\; 1-\alpha_t \qquad \text{for } t \in [t_1+1, t_2]$$

(this is also true when $t_1 = 1$); this implies the statement. ■

Next we discuss the bounds resulting from three settings of the switching rate α_t.

4.1.1 Example 1: Constant Switching Rate

This is the original Fixed Share by Herbster and Warmuth (1998).

Corollary 5 *Fixed Share with constant switching rate $\alpha_t = \alpha$ for $t > 1$ (recall that $\alpha_1 = \frac{N-1}{N}$) has worst-case adaptive regret equal to*

$$\begin{aligned} &\ln(N-1) - \ln\alpha - (t_2 - t_1)\ln\bigl(1-\alpha\bigr) && \text{for } t_1 > 1, \text{ and} \\ &\ln N - (t_2 - 1)\ln\bigl(1-\alpha\bigr) && \text{for } t_1 = 1. \end{aligned}$$

A slightly weaker upper bound was obtained by Cesa-Bianchi et al. (2012). The clear advantage of our analysis with equality is that we can obtain the standard Fixed Share

tracking regret bound by summing the above adaptive regret bounds on individual intervals. Comparing Fixed Share with the best sequence S of experts on the interval $[1, T]$ with m blocks, we obtain the bound

$$L^{\mathrm{FS}}_{[1,T]} - L^S_{[1,T]} \;\le\; \ln N + (m-1)\ln(N-1) - (m-1)\ln\alpha - (T-m)\ln(1-\alpha)\,,$$

which is exactly the standard Fixed Share tracking bound (6). So we see that the reason why Fixed Share can effectively compete with switching sequences is that it can, in fact, effectively compete with any expert on any interval, that is, has small adaptive regret.

4.1.2 Example 2: Slowly Decreasing Switching Rate

The idea of slowly decreasing the switching rate was considered by Shamir and Merhav (1999) in the context of source coding, and later analysed for expert switching by Koolen and De Rooij (2008); we saw in Section 3.2 that it also underlies Follow the Leading History of Hazan and Seshadhri (2009). It results in tracking regret bounds that are almost as good as the bounds for constant α *with optimally tuned* α. These tracking bounds are again implied by the following corresponding adaptive regret bound.

Corollary 6 *Fixed Share with switching rate $\alpha_t = 1/t$ (except for $\alpha_1 = \frac{N-1}{N}$) has worst-case adaptive regret*

$$-\ln\left(\frac{1}{(N-1)t_1}\prod_{t=t_1+1}^{t_2}\frac{t-1}{t}\right) \;=\; \ln(N-1) + \ln t_2 \qquad \text{for } t_1 > 1, \text{ and} \tag{10a}$$

$$-\ln\left(\frac{1}{N}\prod_{t=2}^{t_2}\frac{t-1}{t}\right) \;=\; \ln N + \ln t_2 \qquad \text{for } t_1 = 1. \tag{10b}$$

Comparing Fixed Share with the best sequence S of experts on $[1, T]$ with m blocks we obtain, by summing the bound in Corollary 6 over all blocks,

$$L^{\mathrm{FS}}_{[1,T]} - L^S_{[1,T]} \;\le\; \ln N + (m-1)\ln(N-1) + m\ln T. \tag{11}$$

For comparison, the bound for Fixed Share aggregated over the αs with a suitable prior is

$$L^{\mathrm{AFS}}_{[1,T]} - L^S_{[1,T]} \;\le\; \ln N + (m-1)\ln(N-1) + m\ln T - \ln\bigl((m-1)!\bigr) \tag{12}$$

(see Vovk (1999), Theorem 2, setting $\eta := 1$, $k := m-1$, and $\epsilon := 1$). Our bound (11) is not as good as (12) in that the latter has the nonpositive (negative for $m > 2$) addend $-\ln\bigl((m-1)!\bigr) \sim -m\ln m$. However, the difference does not appear great unless the number m of blocks is very large.[6] And whereas Fixed Share can be implemented in time $O(N)$ per trial, this aggregate seems to require work per-trial scaling with $\sqrt{t}$ (Monteleoni and Jaakkola, 2003; De Rooij and Van Erven, 2009).

6. If $m \le T^c$ for some $c \in (0,1)$, then $(1-c)m\ln T \;\le\; m\ln\frac{T}{m} \;\le\; m\ln T$, so the block timing overheads of (11) and (12) differ by at most a constant factor.

4.1.3 Example 3: Quickly Decreasing Switching Rate

The bounds we have obtained so far depend on t_2 either linearly or logarithmically. To get bounds that depend on t_2 sub-logarithmically, or even not at all, one may instead decrease the switching rate faster than $1/t$, as analysed by Shamir and Merhav (1999) and Koolen and De Rooij (2013). To obtain a controlled trade-off, we consider setting the switching rate to $\alpha_t = \frac{1}{t \ln t}$, except for $\alpha_1 = \frac{N-1}{N}$ and, in the case $N \in \{2,3\}$, $\alpha_2 = \frac{N-1}{N}$ (this is needed since $\frac{1}{2\ln 2} \approx 0.72 > \frac{N-1}{N}$). This leads to adaptive regret at most

$$\ln(N-1) + \ln t_1 + \ln\ln t_1 - \sum_{t=t_1+1}^{t_2} \ln\left(1 - \frac{1}{t \ln t}\right) \;\leq\; \ln(N-1) + \ln t_1 + \ln\ln t_2 \quad (13a)$$

when $t_1 > 2$ or both $t_1 = 2$ and $N > 3$, at most

$$\ln N - \sum_{t=3}^{t_2} \ln\left(1 - \frac{1}{t \ln t}\right) \;\leq\; \ln N + \ln\ln t_2 + 0.37 \quad (13b)$$

when $t_1 = 2$ and $N \in \{2,3\}$, and at most

$$\ln N - \sum_{t=2}^{t_2} \ln\left(1 - \frac{1}{t \ln t}\right) \;\leq\; \ln N + \ln\ln t_2 + 1.65 \quad (13c)$$

when $t_1 = 1$ and $N > 3$ (remember that $\ln\ln 1$ is understood to be 0). In the case where $t_1 = 1$ and $N \in \{2,3\}$, the term $\frac{1}{2\ln 2}$ in (13c) (when it is present, i.e., when $t_2 > 1$) should be replaced by the smaller term $\frac{N-1}{N}$; this does not affect the validity of the bound. In all the cases, the bounds, (13a)–(13c), are stronger than (2b).

The dependence on t_2 in (13) is extremely mild. We can suppress it completely by increasing the dependence on t_1 just ever so slightly. If we set $\alpha_t = t^{-1-\epsilon}$, where $\epsilon > 0$, then the sum of the series $\sum_t \alpha_t$ is finite and the bound becomes

$$\ln(N-1) + (1+\epsilon)\ln t_1 + c_\epsilon \qquad \text{for } t_1 > 1, \text{ and} \quad (14a)$$

$$\ln N + c_\epsilon \qquad \text{for } t_1 = 1, \quad (14b)$$

where $c_\epsilon = -\sum_{t=2}^{\infty} \ln(1 - t^{-1-\epsilon})$. It is clear that the bound (14a) is far from optimal when t_1 is large: c_ϵ can be replaced by a quantity that tends to 0 as $O(t_1^{-\epsilon})$ as $t_1 \to \infty$. In particular, for $\epsilon = 1$ we have the bound

$$\ln N + 2\ln t_1 + \ln 2.$$

An interesting feature of this switching rate is that for the full interval $[t_1, t_2] = [1, T]$ the bound differs from the standard AA bound only by an additive term less than 1. In words, the overhead for small adaptive regret is negligible.

4.2 Fixed Share is Pareto Optimal for Adaptive Regret

We started by considering several intuitive constructions for adaptive algorithms, and saw that they all result in Fixed Share. We then obtained the worst-case adaptive regret of

Fixed Share. Intuitive as it may be, we have not answered the question whether Fixed Share is a good algorithm in the sense that its worst-case adaptive regret bounds are small. It is conceivable that there are smarter algorithms with better adaptive regret guarantees. See for example the palette of tracking algorithms proposed by Koolen and De Rooij (2013). And even if no better algorithms exist, there may still be algorithms that exhibit different trade-offs, in the sense that their worst-case adaptive regret is incomparable to that of Fixed Share.

So in this section we start from the other end and derive lower bounds that hold for any algorithm. As expected, we conclude that the bounds of Fixed Share (with any switching rate sequence $\alpha_t \leq \frac{N-1}{N}$) are Pareto optimal. But it came to us as a surprise that actually *all other bounds are strictly dominated*. No matter how smart the algorithm, its worst-case adaptive regret will be dominated by that of an instance of Fixed Share.

We call a mapping ϕ of intervals to regrets a *candidate guarantee*. Such a candidate guarantee is *realisable* if there is an algorithm for mix-loss prediction (Protocol 1) with adaptive regret at most ϕ. That is, we demand

$$R_{[t_1,t_2]} \;\leq\; \phi(t_1, t_2)$$

for all sequences of expert losses $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots$ in $(-\infty, \infty]^N$ and all choices of the interval $1 \leq t_1 \leq t_2$. We say that a realizable guarantee ϕ *dominates* a realizable guarantee ψ if $\phi(t_1, t_2) \leq \psi(t_1, t_2)$ for all intervals $[t_1, t_2]$; and we say that ϕ *strictly dominates* ψ if, furthermore, $\phi(t_1, t_2) < \psi(t_1, t_2)$ for some interval $[t_1, t_2]$. A realisable guarantee ϕ is *Pareto optimal* if it is not strictly dominated by any realizable guarantee.

We are interested in Pareto-optimal guarantees. Every such guarantee is, by definition, the worst-case regret of an algorithm. Let us make that precise:

Definition 7 *We say that ϕ is the* worst-case adaptive regret *of a given algorithm if*

$$\phi(t_1, t_2) \;=\; \sup_{\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots} R_{[t_1,t_2]} \qquad \text{for all } 1 \leq t_1 \leq t_2.$$

The main step in characterising the Pareto optimal guarantees is showing that worst-case adaptive regrets cannot be too small.

Theorem 8 *Let ϕ be the worst-case adaptive regret of some algorithm. Then*

$$\phi(t, t) \;\geq\; \ln N \qquad \text{for all } 1 \leq t \text{ and} \quad (15a)$$

$$\phi(t_1, t_2) \;\geq\; \phi(t_1, t_1) + \sum_{t=t_1+1}^{t_2} -\ln\left(1 - (N-1)e^{-\phi(t,t)}\right) \qquad \text{for all } 1 \leq t_1 < t_2. \quad (15b)$$

Proof We first show (15a). Since at any time t the N weights played must sum to one, the smallest weight must be $\leq 1/N$. By hitting all others with infinite loss we force regret at least $\ln N$ over the interval $[t, t]$. Now suppose (15a) holds throughout, and consider any interval $[t_1, t_2]$. Fix $\epsilon > 0$. Let $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{t_1}$ be data on which the regret over $[t_1, t_1]$ exceeds $\phi(t_1, t_1) - \epsilon$. Let n^* be the (any) best expert in trial t_1. Now for all $t \in (t_1, t_2]$ choose $\ell_t^{n^*} = 0$ and $\ell_t^n = \infty$ for all $n \neq n^*$. In any trial t, the algorithm must allocate at least

weight $e^{-\phi(t,t)}$ to each expert to guarantee ϕ on the singleton interval $\{t\}$. Since the weights sum to one, the weight allocated to expert n^* during each trial $t \in (t_1, t_2]$ can be at most $1-(N-1)e^{-\phi(t,t)}$. Hence the loss of the algorithm must be at least $-\ln\bigl(1-(N-1)e^{-\phi(t,t)}\bigr)$. Since the loss of the best expert n^* on $(t_1, t_2]$ is zero, the regret is at least the right-hand side of (15b) minus ϵ. The worst-case regret $\phi(t_1, t_2)$ must be at least as large. Since this holds for all ϵ, we have proved (15b). ■

A realisable guarantee is witnessed by some algorithm, and is therefore dominated by the worst-case adaptive regret of that algorithm. We proved that this worst-case adaptive regret must satisfy (15). We now show that any guarantee satisfying (15) is realised by an instance of Fixed Share.

Theorem 9 *Let ϕ satisfy* (15). *Then Fixed Share with switching rate sequence $\alpha_2, \alpha_3, \ldots$ where $\alpha_t = (N-1)e^{-\phi(t,t)}$ guarantees ϕ.*

Proof From (15a) we know that $\alpha_t \le (N-1)/N$, so the worst case regret of Fixed Share is given by Theorem 4. In particular (9) with our choice of α_t equals the right-hand side of (15b), and so FS guarantees ϕ. Note that the fact that α_1 is always set to the specific value $(N-1)/N$ only works in our favour here. ■

By combining the preceding two theorems, in the following two corollaries we characterize realizable guarantees and obtain canonical representatives of the Pareto guarantees for adaptive regret.

Corollary 10 *Let ϕ be a candidate guarantee. The following are equivalent:*

- *ϕ is realisable*
- *there exists $\psi \le \phi$ such that ψ satisfies* (15)
- *ϕ is dominated by the worst-case adaptive regret of a Fixed Share.*

Before stating the second corollary we need to tackle the problem of big α_ts.

Lemma 11 *Fixed Share with $\alpha_t \in [0, \frac{N-1}{N}]$ is Pareto optimal. If $\sup_t \alpha_t > \frac{N-1}{N}$, Fixed Share with such parameters α_t is strictly dominated by Fixed Share with parameters $\alpha_t \wedge \frac{N-1}{N}$.*

Proof Suppose Fixed Share with parameters $\alpha_t \in [0, \frac{N-1}{N}]$ dominates Fixed Share with parameters $\beta_t \in [0, \frac{N-1}{N}]$. Equation (9) with $t_1 = t_2$ then implies $\alpha_t \ge \beta_t$ for all t. Combining this with (9) with $t_1 = 1$ now implies $\alpha_t = \beta_t$ for all t. In combination with the last statement of Corollary 10 this proves the first statement of the lemma.

For the second statement of the lemma, we should prove that in the case $\sup_t \alpha_t > \frac{N-1}{N}$ the worst-case regret of Fixed Share with parameters α_t for a fixed interval $[t_1, t_2]$ containing t with $\alpha_t > \frac{N-1}{N}$ will be greater than the worst-case regret of Fixed Share with parameters $\alpha_t \wedge \frac{N-1}{N}$) (given by (9) with α_t replaced by $\alpha_t \wedge \frac{N-1}{N}$). To see this, modify the "worst-case data" described in the proof of Theorem 4 (which are no longer worst-case for α_t) as follows: if $\alpha_{t_1} > \frac{N-1}{N}$ (and so $t_1 > 1$), make the loss of the good expert at step $t_1 - 1$ finite and

make the losses of all other experts at step $t_1 - 1$ infinite; if $\alpha_t > \frac{N-1}{N}$ for $t \in (t_1, t_2]$, make the loss of the good expert at step $t-1$ infinite and make the loss of another expert at step $t-1$ finite. ■

We conclude that Fixed Share is *the* answer for worst-case adaptive regret bounds.

Corollary 12 *Let ϕ be a candidate guarantee. The following are equivalent:*

- *ϕ is Pareto optimal*
- *ϕ satisfies* (15a) *with equality for $t = 1$ and satisfies* (15b) *with equality throughout*
- *ϕ is the worst-case adaptive regret of a Fixed Share with parameters $\alpha_t \in [0, \frac{N-1}{N}]$.*

Proof First, let ϕ be Pareto optimal. Since it is then the worst-case adaptive regret of some algorithm, it satisfies (15). By Theorem 9 ϕ is guaranteed by a Fixed Share with $\alpha_t \in [0, \frac{N-1}{N}]$, and so ϕ is the worst-case adaptive regret of such a Fixed Share. And, as noted in the proof of Theorem 9, the worst case adaptive regret of such a Fixed Share satisfies (15a) with equality for $t = 1$ and (15b) with equality throughout. Second, let ϕ satisfy (15a) with equality for $t = 1$ and (15b) with equality throughout. By Theorem 9 ϕ is guaranteed by Fixed Share with $\alpha_t = (N-1)e^{\phi(t,t)} \in [0, \frac{N-1}{N}]$. Now Theorem 4 implies that ϕ is the worst-case adaptive regret of this Fixed Share. Third, let ϕ be the worst-case adaptive regret of a Fixed Share with parameters $\alpha_t \in [0, \frac{N-1}{N}]$. Then ϕ is Pareto optimal by Lemma 11. ■

4.3 Sacrifice Early to Benefit Later? Impossible

We have seen that the Pareto optimal guarantees are those witnessed by Fixed Share. We saw several bounds (2), worked out in detail in Section 4.1, with different dependencies on the interval endpoints t_1 and t_2. In this section we investigate the possibility of forgoing completely the guarantees on early intervals to substantially improve the guarantees on all late intervals. We conclude that this is essentially impossible.

The main tool in this section is a slight relaxation of Theorem 8 that holds for all guarantees, not only for worst-case regrets.

Corollary 13 *If $\phi(t_1, t_2)$ is realisable, then*

$$\phi(t_1, t_2) \;\ge\; \ln N + \sum_{t=t_1+1}^{t_2} -\ln\Bigl(1-(N-1)e^{-\phi(t,t)}\Bigr) \qquad \text{for all } 1 \le t_1 \le t_2. \tag{16}$$

Proof Plug (15a) into (15b) and notice that increasing ϕ increases the left-hand side and decreases the right-hand side of (16). ■

The following corollary shows that the stronger form (10) of (2a) is essentially tight: for large t_1 and t_2 we cannot even improve the right-hand side of (10a) by a positive constant (it is not sufficient to ignore all intervals with $t_1 < T_0$ for an arbitrarily large T_0).

Corollary 14 *Fix a constant $C < \ln(N-1)$ and an arbitrarily large positive integer constant T_0. The following guarantee is not realisable:*

$$\phi(t_1,t_2) \;=\; \begin{cases} C+\ln t_2 & \text{if } t_1 \ge T_0 \\ \infty & \text{otherwise.} \end{cases} \tag{17}$$

Proof For $t_1 \ge T_0 - 1$, the right hand side of (16) is at least (using $-\ln(1-x) \ge x$ and Euler's summation formula)

$$\ln N - \sum_{t=t_1+1}^{t_2} \ln\left(1 - \frac{N-1}{e^C t}\right) \;\ge\; \ln N + \frac{N-1}{e^C} \sum_{t=t_1+1}^{t_2} \frac{1}{t} \;\ge\; \ln N + \frac{N-1}{e^C}\left(\ln t_2 - \ln t_1 - D\right),$$

for some constant D. (The inequality $-\ln(1-x) \ge x$ assumes only $x < 1$ and so is applicable if T_0 is sufficiently large.) For a fixed t_1 (say, $t_1 = T_0$), this will exceed $C + \ln t_2$ (the guarantee 17) when t_2 is sufficiently large. Contradiction. ■

Our next corollary is about the tightness of (2b) and its elaboration (13) (especially 13a).

Corollary 15 *Fix a constant $C < \ln(N-1)$ and positive integer T_0. The following candidate guarantee is not realisable:*

$$\phi(t_1,t_2) \;=\; \begin{cases} C+\ln t_1 + \ln\ln t_2 & \text{if } t_1 \ge T_0 \\ \infty & \text{otherwise.} \end{cases}$$

Proof As in the previous proof, we can see that for $t_1 \ge T_0 - 1$ the right hand side of (16) is at least

$$\ln N - \sum_{t=t_1+1}^{t_2} \ln\left(1 - \frac{N-1}{e^C t \ln t}\right) \;\ge\; \ln N + \sum_{t=t_1+1}^{t_2} \frac{N-1}{e^C t \ln t} \;\ge\; \ln N + \frac{N-1}{e^C}\left(\ln\ln t_2 - \ln\ln t_1 - D\right).$$

For a fixed t_1 and a large enough t_2 this will exceed $C + \ln t_1 + \ln\ln t_2$. Contradiction. ■

Finally, we explore the tightness of (2c) and its elaboration given later in the paper: see (14) and the discussion afterwards. Let us say that a sequence $a(1), a(2), \ldots$ is $O(1)$-*realisable* if there is a C such that the candidate guarantee $\phi(t_1,t_2) = a(t_1) + C$ is realisable. Let us say that a sequence $a(1), a(2), \ldots$ is $O(1)$-*realisable in the long run* if there are C and T_0 such that the candidate guarantee

$$\phi(t_1,t_2) \;=\; \begin{cases} a(t_1) + C & \text{if } t_1 \ge T_0 \\ \infty & \text{otherwise} \end{cases} \tag{18}$$

is realisable.

Corollary 16 *A sequence $a(t)$ is $O(1)$-realisable*

- *if and only if it is $O(1)$-realisable in the long run, and*

- *if and only if $\sum_t e^{-a(t)} < \infty$.*

Proof Suppose $\sum_t e^{-a(t)} < \infty$. Then $a(t)$ is $O(1)$-realisable by Theorem 4 (set $\alpha_t = e^{-a(t)}$ from some t on). To prove that the series converges when $a(t)$ is $O(1)$-realisable in the long run, suppose $\sum_t e^{-a(t)} = \infty$. If (18) is realisable, we can again see that for $t_1 \ge T_0 - 1$ the right hand side of (16) is at least

$$\ln N + \sum_{t=t_1+1}^{t_2} -\ln\left(1 - (N-1)e^{-C}e^{-a(t)}\right) \;\ge\; \ln N + \sum_{t=t_1+1}^{t_2} (N-1)e^{-C}e^{-a(t)}.$$

For a fixed t_1 and a large enough t_2 this will exceed $a(t_1) + C$. Contradiction. ■

The first statement of Corollary 16 can be interpreted as a weak statement of impossibility to sacrifice early in order to benefit later: we can gain at most a constant when we sacrifice early. (And we saw below (14) that we can indeed gain a constant.) This statement, however, is very general, as the following simple argument shows.

Let us say that a candidate guarantee $\phi(t_1,t_2)$ is $O(1)$-*realisable* if there is a C such that the candidate guarantee $\psi(t_1,t_2) = \phi(t_1,t_2) + C$ is realisable. Let us say that the candidate guarantee $\phi(t_1,t_2)$ is $O(1)$-*realisable in the long run* if there are C and T_0 such that the candidate guarantee

$$\psi(t_1,t_2) \;=\; \begin{cases} \phi(t_1,t_2) + C & \text{if } t_1 \ge T_0 \\ \infty & \text{otherwise} \end{cases} \tag{19}$$

is realisable.

Lemma 17 *A candidate guarantee ϕ is $O(1)$-realisable if and only if it is $O(1)$-realisable in the long run.*

Proof Suppose ϕ is $O(1)$-realisable in the long run and let C and T_0 be such that (19) is realisable. Aggregate using the AA the following prediction strategies: a prediction algorithm that realises ϕ; a prediction algorithm suffering a bounded regret over all $[1,t]$; a prediction algorithm suffering a bounded regret over all $[2,t]$;...; a prediction algorithm suffering a bounded regret over all $[T_0 - 1, t]$. ■

5. Conclusion

We examined the problem of guaranteeing small adaptive regret for the setting of prediction with expert advice. In the first part we considered two techniques to obtain adaptive algorithms: using virtual specialist experts and restarting classical algorithms. We showed that both result in Fixed Share with a variable switching rate. In the second part we computed the exact worst-case adaptive regret for Fixed Share, thus tightening the existing upper bounds. So much, in fact, that by summing these worst-case regrets over a partition of the interval $[1,T]$ we recover the standard Fixed Share tracking bound. In other words, the tracking performance of Fixed Share is a consequence of its adaptivity.

Protocol 2 Prediction with Expert Advice

for $t = 1, 2, \ldots$ **do**
  Expert n announces prediction $\gamma_t^n \in \Gamma$ for each $n = 1, \ldots, N$
  Learner announces prediction $\gamma_t \in \Gamma$
  Reality announces outcome $\omega_t \in \Omega$
  Learner suffers loss $\lambda(\gamma_t, \omega_t)$
end for

We then give an information-theoretic characterisation of the achievable worst-case adaptive regrets, and conclude that the Pareto optimal regrets are exactly the Fixed Share regrets. Fixed Share simply is the optimal adaptive algorithm.

Open problem. Whereas upper bounds readily transfer to mixable losses, obtaining adaptive regret lower bounds for mixable losses is much more tricky. It is fair to call the lower bound argument by Vovk (1998) for classical regret complicated, and this would be a special case for adaptive regret lower bounds.

Acknowledgments

First author supported by Veterinary Laboratories Agency (VLA) of Department for Environment, Food and Rural Affairs (Defra) and by a Leverhulme Trust research project grant RPG-2013-047 "On-line Self-Tuning Learning Algorithms for Handling Historical Information". Second author supported by NWO Rubicon grant 680-50-1010. Third author supported by EPSRC grant EP/I030328/1 and AFOSR grant "Semantic Completions".

Appendix A. Adaptive Regret for Mix Loss Transfers to Mixable Losses

The protocol of prediction with expert advice is given as Protocol 2. Predictions are made sequentially, their quality is measured by the loss function λ and Learner has access to a (finite) pool of N experts. An important class of loss functions that allow for effective algorithms are mixable losses.

Definition 18 *A loss function λ is called* η-mixable*, where $\eta > 0$, if for every N, every sequence of predictions $\gamma^1, \ldots, \gamma^N$ and every sequence of normalised nonnegative weights $u^1, \ldots, u^N$ there exists a prediction $\gamma \in \Gamma$ such that for every outcome $\omega \in \Omega$*

$$\lambda(\gamma, \omega) \;\le\; -\frac{1}{\eta} \ln \left(\sum_{n=1}^{N} u^n e^{-\eta \lambda(\gamma^n, \omega)} \right). \tag{20}$$

A function Σ that maps every sequence of predictions $\gamma^1, \ldots, \gamma^N$ and every sequence of normalised nonnegative weights $u^1, \ldots, u^N$ of the same length to $\gamma \in \Gamma$ satisfying (20) is called an η-perfect substitution function.

To establish mixability it is sufficient to verify (20) for $N = 2$. Examples of mixable games are discussed by Vovk (2001). Note that $1/\eta$-scaled mix loss is the baseline used in

the definition of mixability: see (20). In this sense the mix loss is the hardest mixable loss. It is hence no surprise that adaptive regret bounds for mix loss immediately transfer to any mixable loss:

Fact 19 *Let X be a mix-loss algorithm with worst-case adaptive regret $\phi(t_1, t_2)$. If λ is η-mixable then there is an algorithm Y with adaptive regret at most $\phi(t_1, t_2)/\eta$.*

Proof Let Σ be an η-perfect substitution function for λ. We choose Y to be the algorithm that operates as follows. At each trial $t = 1, 2, \ldots$ it obtains prediction $\boldsymbol{u}_t$ from X, predicts with $\gamma_t = \Sigma(\boldsymbol{u}_t, \boldsymbol{\gamma}_t)$, and feeds into X losses $\ell_t^n = \eta\lambda(\gamma_t^n, \omega_t)$ (from the point of view of Y these are scaled losses rather than losses). Then for each interval $[t_1, t_2]$ and reference expert n we have

$$\begin{aligned} L^Y_{[t_1,t_2]} - L^n_{[t_1,t_2]} \;&=\; \sum_{t=t_1}^{t_2} \lambda(\gamma_t, \omega_t) - \sum_{t=t_1}^{t_2} \lambda(\gamma_t^n, \omega_t) \\ &\le\; -\frac{1}{\eta} \sum_{t=t_1}^{t_2} \ln \sum_j u_t^j e^{-\eta\lambda(\gamma_t^j, \omega_t)} - \sum_{t=t_1}^{t_2} \lambda(\gamma_t^n, \omega_t) \\ &=\; -\frac{1}{\eta} \sum_{t=t_1}^{t_2} \ln \sum_j u_t^j e^{-\ell_t^j} - \frac{1}{\eta} \sum_{t=t_1}^{t_2} \ell_t^n \;=\; \frac{1}{\eta} \left(L^X_{[t_1,t_2]} - L^n_{[t_1,t_2]} \right) \;\le\; \frac{1}{\eta}\phi(t_1, t_2), \end{aligned}$$

where the first inequality follows from the definition of an η-perfect substitution function and the last one from our assumption about X. ■

Fact 19 shows that all our performance guarantees for the mix-loss protocol carry over to the protocol of prediction with expert advice with a mixable loss function.

Appendix B. Worst-case Adaptive Regret Data for Fixed Share

In this subsection we prove that the worst-case data for Fixed Share has the following form. On the interval $[t_1, t_2]$ we are interested in all but one expert suffer infinite loss and on the step preceding t_1 (if $t_1 \neq 1$) this one expert suffers infinite loss himself. The construction is iterative and we start constructing the data from the end of the interval.

Lemma 20 *For any history prior to the step t_2 the adaptive regret $R^n_{[t_1,t_2]}$ w.r.t. expert n on the interval $[t_1, t_2]$ is maximised with $\ell^k_{t_2} = \infty$ for $k \neq n$.*

Proof Let us differentiate the adaptive regret w.r.t. $\ell^k_{t_2}$:

$$\frac{\partial R^n_{[t_1,t_2]}}{\partial \ell^k_{t_2}} \;=\; \frac{\partial(\ell_{t_2} - \ell^n_{t_2})}{\partial \ell^k_{t_2}} \;=\; -\frac{\partial}{\partial \ell^k_{t_2}} \ln \sum_j u^j_{t_2} e^{-\ell^j_{t_2}} - \mathbf{1}_{\{n=k\}} \;=\; \frac{u^k_{t_2} e^{-\ell^k_{t_2}}}{\sum_j u^j_{t_2} e^{-\ell^j_{t_2}}} - \mathbf{1}_{\{n=k\}}.$$

This is positive for all $k \neq n$ and becomes zero for $k = n$ when we plug in $\ell^k_{t_2} = \infty$ for those. ■

Lemma 21 *Consider switching rates $\alpha_t \in [0, \frac{N-1}{N}]$. Fix a comparator expert n. Let $t \in [t_1, t_2]$. Suppose that the losses for steps $s = t+1, \ldots, t_2$ satisfy $\ell_s^k = \infty$ for $k \neq n$. Then for the adaptive regret $R^n_{[t_1,t_2]}$ is maximised with $\ell_t^k = \infty$ for $k \neq n$.*

Proof Let us start with showing that on the steps $t+1$ and $t+2$ the data is organised as we want to, that is, the n-th expert is good and all others suffer infinite loss, then Learner's loss on step $t+2$ is not dependent on what happens at time t and before. This follows immediately from (5), as

$$\ell_{t+2} \;=\; -\ln\left(1-\alpha_{t+2}\right)\,.$$

Now let us differentiate the adaptive regret $R^n_{[t_1,t_2]}$ w.r.t. ℓ_t^k assuming that the future losses are set up as we want. Let us show that the derivatives w.r.t. ℓ_t^k where $k \neq n$ are all positive. For those,

$$\frac{\partial R^n_{[t_1,t_2]}}{\partial \ell_t^k} \;=\; \frac{\partial \ell_t}{\partial \ell_t^k} + \frac{\partial \ell_{t+1}}{\partial \ell_t^k}\,.$$

Expanding the second one gives (as before, $k \neq n$):

$$\begin{aligned}\frac{\partial \ell_{t+1}}{\partial \ell_t^k} \;&=\; \frac{\partial}{\partial \ell_t^k} - \ln\left(\frac{\alpha_{t+1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t+1})u_t^n e^{\ell_t - \ell_t^n}\right)\\ &=\; -\frac{(1-\frac{N}{N-1}\alpha_{t+1})u_t^n e^{\ell_t-\ell_t^n}\frac{\partial}{\partial \ell_t^k}\ell_t}{\frac{\alpha_{t+1}}{N-1} + (1-\frac{N}{N-1}\alpha_{t+1})u_t^n e^{\ell_t-\ell_t^n}}\,.\end{aligned}$$

So we see that

$$\begin{aligned}\frac{\partial R^n_{[t_1,t_2]}}{\partial \ell_t^k} \;&=\; \frac{\partial \ell_t}{\partial \ell_t^k}\left(1 - \frac{(1-\frac{N}{N-1}\alpha_{t+1})u_t^n e^{\ell_t-\ell_t^n}}{\frac{\alpha_{t+1}}{N-1} + (1-\frac{N}{N-1}\alpha_{t+1})u_t^n e^{\ell_t-\ell_t^n}}\right)\\ &=\; \frac{\partial \ell_t}{\partial \ell_t^k}\left(\frac{\frac{\alpha_{t+1}}{N-1}}{\frac{\alpha_{t+1}}{N-1} + (1-\frac{N}{N-1}\alpha_{t+1})u_t^n e^{\ell_t-\ell_t^n}}\right) \;>\; 0\,.\end{aligned}$$

So our worst-case pattern of losses extends one trial backwards. ■

Finally, we need to state the almost obvious fact that in order to maximise the adaptive regret we need to insert an infinite loss for the comparator expert right before the start of the interval, thus killing all the previous weight on him.

Lemma 22 *Consider switching rates $\alpha_t \in [0, \frac{N-1}{N}]$. Fix a comparator expert n. Suppose that the losses for steps $s = t_1, \ldots, t_2$ satisfy $\ell_s^k = \infty$ for $k \neq n$. Then the adaptive regret $R^n_{[t_1,t_2]}$ is maximised with $\ell^n_{t-1} = \infty$.*

Proof As before, the adaptive regret on steps starting from t_1+1 does not depend on $\ell^k_{t_1-1}$. So let us show that $\frac{\partial R^n_{[t_1,t_2]}}{\partial \ell^n_{t_1-1}} > 0$. We can reuse the proofs of previous lemmas for that:

$$\frac{\partial R^n_{[t_1,t_2]}}{\partial \ell^n_{t_1-1}} \;=\; \frac{\partial \ell_{t_1}}{\partial \ell^n_{t_1-1}} \;=\; -\frac{(1-\frac{N}{N-1}\alpha_{t_1})u^n_{t_1-1}e^{\ell_{t_1-1}-\ell^n_{t_1-1}}}{\frac{\alpha_{t_1}}{N-1} + (1-\frac{N}{N-1}\alpha_{t_1})u^n_{t_1-1}e^{\ell_{t_1-1}-\ell^n_{t_1-1}}}\frac{\partial\left(\ell_{t_1-1}-\ell^n_{t_1-1}\right)}{\partial \ell^n_{t_1-1}} \;>\; 0\,,$$

since $\frac{\partial\left(\ell_{t_1-1}-\ell^n_{t_1-1}\right)}{\partial \ell^n_{t_1-1}}$ is negative as follows from the proof of Lemma 20. ■

References

Olivier Bousquet and Manfred K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3:363–396, 2002.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Nicolò Cesa-Bianchi, Pierre Gaillard, Gábor Lugosi, and Gilles Stoltz. Mirror Descent meets Fixed Share (and feels no regret). In *NIPS Proceedings*, pages 989–997, 2012.

Alexey Chernov and Vladimir Vovk. Prediction with expert evaluators' advice. In *ALT Proceedings*, volume 5809 of *Lecture Notes in Computer Science*, pages 8–22, 2009.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

Yoav Freund, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. Using and combining predictors that specialize. In *STOC Proceedings*, pages 334–343, 1997.

András György, Tamás Linder, and Gábor Lugosi. Efficient tracking of large classes of experts. *IEEE Transactions on Information Theory*, 58(11):6709–6725, 2012.

Elad Hazan and Comandur Seshadhri. Adaptive algorithms for online optimization. In *Electronic Colloquium on Computational Complexity*, 2007. TR07-88.

Elad Hazan and Comandur Seshadhri. Efficient learning algorithms for changing environments. In *ICML Proceedings*, pages 393–400, 2009.

Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32:151–178, 1998.

Wouter M. Koolen and Steven de Rooij. Combining expert advice efficiently. In *COLT Proceedings*, pages 275–286, 2008.

Wouter M. Koolen and Steven de Rooij. Universal codes from switching strategies. *IEEE Transactions on Information Theory*, 59(11):7168–7185, November 2013.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

Haipeng Luo and Robert E. Schapire. Achieving all with no parameters: Adaptive NormalHedge. In *COLT Proceedings*, pages 1286–1304, June 2015.

Claire Monteleoni and Tommi Jaakkola. Online learning of non-stationary sequences. In *NIPS Proceedings*, pages 1093–1100, 2003.

Steven de Rooij and Tim van Erven. Learning the switching rate by discretising Bernoulli sources online. In *AISTATS Proceedings*, pages 432–439, 2009.

Gil I. Shamir and Neri Merhav. Low complexity sequential lossless coding for piecewise stationary memoryless sources. *IEEE Transactions on Information Theory*, 45:1498–1519, 1999.

Vladimir Vovk. Aggregating strategies. In *COLT Proceedings*, pages 371–383, 1990.

Vladimir Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56:153–173, 1998.

Vladimir Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35: 247–282, 1999.

Vladimir Vovk. Competitive on-line statistics. *International Statistical Review*, 69:213–248, 2001.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML Proceedings*, pages 928–936, 2003.

Journal of Machine Learning Research 17 (2016) 1-42 Submitted 12/13; Revised 10/15; Published 4/16

Learning Using Anti-Training with Sacrificial Data

Michael L. Valenzuela MVALENZ@ECE.ARIZONA.EDU
Jerzy W. Rozenblit JR@ECE.ARIZONA.EDU
Electrical and Computer Engineering Department
University of Arizona
1230 E. Speedway Blvd.
Tucson, AZ 85721, UNITED STATES

Editor: Martin Pelikan

Abstract

Traditionally the machine-learning community has viewed the No Free Lunch (NFL) theorems for search and optimization as a limitation. We review, analyze, and unify the NFL theorem with the perspectives of "blind" search and meta-learning to arrive at necessary conditions for improving black-box optimization. We survey meta-learning literature to determine when and how meta-learning can benefit machine learning. Then, we generalize meta-learning in the context of the NFL theorems, to arrive at a novel technique called *anti-training with sacrificial data* (ATSD). Our technique applies at the meta level to arrive at domain specific algorithms. We also show how to generate sacrificial data. An extensive case study is presented along with simulated annealing results to demonstrate the efficacy of the ATSD method.

Keywords: Machine Learning, Optimization, Meta Optimization, No Free Lunch, Anti-Training, Sacrificial Data

1. Introduction

Most types of machine learning involve fitting a model to data. These models may be artificial neural networks, Bayesian networks, cluster centers, decision trees, etc. The fitting process uses objective/fitness functions to determine how well the learned model represents the training data. While many models (*e.g.,* artificial neural networks) have training algorithms (*e.g.,* backpropagation), it is not always clear which algorithm is best to use for a particular data set. This is the issue John Rice describes as the algorithm selection problem (Rice, 1976).

Knowledge about a problem must be used to solve it efficiently (Wolpert and Macready, 1997). It is a poor choice to use an algorithm that ignores derivatives or convex properties when the optimization problem exhibits these features. Similarly, it is a poor choice to use such features when they are absent. It may even be that the features change as the problem's parameters change. For instance, the number of local minima for a single neuron grows exponentially in the dimensionality of the input (Auer et al., 1995). The standard learning algorithm may not remain the most efficient as the dimensionality grows. Worse yet, some learning depends upon black-box optimizers such as simulated annealing (Goodsell and Olson, 1990), genetic algorithms (Wang et al., 2001), particle swarm (Zhang et al., 2004), and random search (Bergstra and Bengio, 2012).

The same virtue that makes black-box optimizers so widely used is also their inherent weakness—black-box optimizers use only a history of inputs into, and outputs from an objective function. This allows black-box optimizers to function on symbolic representations, experimental data, or results from simulations. The black-box optimizer does not use any knowledge about the symbolic function, data, or simulation. The No Free Lunch (NFL) theorems for search and optimization state that any such black-box optimization is expected to perform on average as well as a random guesser (Wolpert and Macready, 1997). If the objective function comes from a slow simulation or a physical experiment, then every function evaluation is precious. Thus, an algorithm tailored toward a particular problem distribution can make better use of the function evaluations.

In this work, we introduce a completely novel concept: *anti-training with sacrificial data*, for tailoring learning and optimization algorithms to problem distributions. At its base, anti-training is a generalization of a type of meta-learning. We take advantage of the fact that all optimization algorithms perform the same on average (according to the NFL theorems). Essentially, anti-training worsens the performance over problems suspected to be impossible or exceedingly unlikely to occur. Since the average performance remains constant, the performance elsewhere must increase. Consider the following water balloon analogy: the NFL theorems function as the conservation of volume. Squeezing the water (performance) out of one area of the balloon will transfer it to other areas.

Unlike many forms of meta-learning, anti-training inherently adheres to the NFL theorems since it is derived through the manipulation of the NFL theorems. Anti-training is expected to benefit learning performance whenever the objective functions are compressible. Moreover, because the sacrificial data used by anti-training is inexpensive to generate, it is possible to use anti-training where meta-learning cannot be applied. It can also be applied in conjunction with meta-learning and can prevent overtraining.

Following preliminaries and nomenclature in Section 2, we rewrite an NFL theorem to derive our technique and proceed to analyze it in Section 3. Section 4 discusses how to generate the sacrificial data required for anti-training. We extensively discuss several experiments in Section 5 and the results in Section 6. We then conclude with a discussion and future work in Section 7.

2. Preliminaries and Background

The original NFL theorems apply to arbitrarily large finite domains (*e.g.,* combinatorial optimization problems). Wolpert and Macready (1997) argue that this makes sense since computers only have a finite number of bits; the search space $\mathcal{X}$ and output space $\mathcal{Y}$ are typically bit sequences. Continuing with their notation, let f be a function representing an optimization problem (*e.g.,* error function, distance, etc.) and $\mathcal{F}$ be the set of all functions having an input $x \in \mathcal{X}$ and an output $y \in \mathcal{Y}$. If $|\mathcal{X}|$ is the size of the search space and $|\mathcal{Y}|$ is the size of the output space, then the size of the function space is $|\mathcal{F}| = |\mathcal{Y}|^{|\mathcal{X}|}$ (Wolpert and Macready, 1997). (Just to realize the order of magnitude of $|\mathcal{F}|$, consider a function with a 16-bit input and an 8-bit output; then $|\mathcal{F}| = 2^{2^{19}} \approx 2.6 \cdot 10^{157826}$. This is a large number of possible functions, but it is still finite.)

Moreover, let

- $f \in \mathcal{F}$ be a function to be optimized,

- $P(f)$ be a probability mass function, describing the probability of f occurring,
- $\hat{P}(f)$ be an estimate of $P(f)$,
- a be a black-box algorithm that samples f and uses the input-output data to generate the next input into f,
- $m \in \{1, \ldots, |\mathcal{X}|\}$ be the number of unique inputs,
- d_m^x be a set of the first m unique inputs,[1]
- d_m^y be a set of the corresponding outputs,
- d_m be shorthand for d_m^y and d_m^x combined,
- $P\left(d_m^y | f, m, a\right)$ be the conditional probability of finding d_m^y given an algorithm a iterated m times on a fitness function f, and
- $\Phi\left(d_m^y\right)$ be a function that converts the output to some performance metric.

For brevity, we provide the following terms:

- *anti-training with sacrificial data* is shortened to *ATSD*,
- *anti-training with sacrificial data combined with meta-learning* is abbreviated as *ATSD+ML*,
- *meta-learning* is abbreviated as *ML*, and
- *No Free Lunch* to be abbreviated as *NFL*.

2.1 NFL

Briefly, the NFL theorems state that for both static and time-varying optimization problems, all algorithms perform the same when averaged across all problems (Wolpert and Macready, 1997). We will focus on static problems. The NFL theorems are cast in a probabilistic framework:

Theorem 1

$$\sum_{f \in \mathcal{F}} P\left(d_m^y | f, m, a_1\right) = \sum_{f \in \mathcal{F}} P\left(d_m^y | f, m, a_2\right). \tag{1}$$

In the words of the authors: "this means in particular that if some algorithm's performance is superior to that of another algorithm over some set of optimization problems, then the reverse must be true over the set of all other optimization problems." The above equation is true regardless of the problem distribution, $P(f)$. This is different from saying all algorithms are equal. For all algorithms to be equal we require:

1. Algorithms are considered to never reevaluate a previously investigated input (allow previous function evaluations to be recalled without incrementing m).

Theorem 2

$$\sum_{f \in \mathcal{F}} P\left(f\right) P\left(d_m^y | f, m, a_1\right) = \sum_{f \in \mathcal{F}} P\left(f\right) P\left(d_m^y | f, m, a_2\right). \tag{2}$$

This explicitly depends on $P(f)$. Following the proof provided in Wolpert and Macready (1997), (2) requires

$$\sum_{f \in \mathcal{F}} P\left(f\right) P\left(d_1^y | f, m = 1, a\right) = \sum_{f \in \mathcal{F}} P\left(f\right) \delta\left(d_1^y, f\left(d_1^x\right)\right) \tag{3}$$

to be independent of d_1^x and hence a. This sum is used in the base case for the inductive proof in Wolpert and Macready (1997), where for information theoretic reasons, they assume a $P(f)$ to be uniform. $P(f)$ could also be any distribution following $\prod_{x \in \mathcal{X}} P(y = f(x))$, or certain distributions with specific correlations between costs and inputs (Wolpert and Macready, 1997).

The NFL theorems have been extended to include additional priors and results. Schumacher et al. (2001) extend this to include sets of functions closed under permutations of the input-space. Thus, all algorithms sum to the same performance and generate the same collection of d_m when all functions are considered (Schumacher et al., 2001). Other analyses and extensions can be found in Culberson (1998); Droste et al. (2002); Corne and Knowles (2003a,b); Igel and Toussaint (2003, 2004); Giraud-Carrier and Provost (2005); Whitley and Watson (2005); Wolpert and Macready (2005); Auger and Teytaud (2007, 2010)). The results of these studies show that the No Free Lunch theorems are valid, but their relevance varies depending on the situation. Yet, despite all the extensions and debate, "there are no specific methods or algorithms that directly follow from NFL" (Whitley and Watson, 2005). In the paper, we provide a method that stems from the NFL theorems.

The NFL theorems treat the definition of "algorithms" differently than the typical definition of algorithm implies. In the NFL theorems, the "algorithms" are more about exploring the search space (points) in some order than about the actual instructions carried out by the computer. Two "algorithms" are considered identical if they always visit points in the same order, even if the underlying programming code is different. Likewise, the NFL theorems call two "algorithms" different even if they share the same code, but run with different hyper-parameters (*e.g.,* step-size, branching preference, etc.) so long as points are visited in a different order.

In the following subsections, we briefly summarize information theory and meta-learning prerequisites needed to establish the foundations for ATSD.

2.2 Information Theory

Information theory is a useful tool for studying learning (Bialek et al., 2001; Rissanen, 1984, 1986, 1987, 1989) and analyzing the NFL theory (Schumacher et al., 2001). In information theory, there are two common measures of complexity: "entropy" and "Kolmogorov complexity." Entropy is typically defined as:

$$H(X) = \sum_{x \in \mathcal{X}} -P(x) \log P(x),$$

where X is a discrete random variable and $\mathcal{X}$ is the support of X. Entropy is sometimes interpreted as random variable's uncertainty. Kolmogorov complexity, sometimes called algorithmic entropy or complexity under analysis, cannot be specified so concisely. Essentially Kolmogorov complexity describes the size of the smallest program needed to reproduce a specified sequence (*e.g.,* bits, characters, numbers, etc.). Since there are multiple languages and machine architectures, Kolmogorov complexity is an incomputable function (Cover and Thomas, 2006; Li and Vitányi, 2008). In this paper, when we say something is compressible or incompressible, we mean this in the sense of Kolmogorov complexity. Information theory shows most functions are incompressible (a proof can be found in the work of Cover and Thomas (2006)). One should be careful to distinguish the ideas of a function being compressible versus being represented in polynomial space. If the number of bits needed to represent a sequence can be halved, there are still an exponential number of such bit sequences.

Culberson (1998) indirectly uses entropy when discussing the NFL theorems from different degrees of "blindness." The degrees of "blindness" range from complete information (the objective function is known) to no information. When these degrees of "blindness" are recast in a probabilistic framework, one can see that the random variable describing the objective function has zero entropy in the case of complete information and maximal entropy when nothing is known. Thus a main point of Culberson (1998) is that optimization performance degrades as the problem distribution becomes more complex.

2.3 Meta-learning

Before we begin our formulation of ATSD, due to the strong ties between ML and ATSD, it is important to address questions pertaining to the NFL theorems and ML. The first and probably most important question is: "can ML successfully improve performance in light of the NFL theorems?" If so, when does ML help? How can one judge the quality of an algorithm's learning biases?

ML is generally concerned with learning to learn better. Nevertheless, it can be applied to learning to optimize better. One perspective of ML is the algorithm selection problem (Rice, 1976). That is, given a specific instance of a problem, which algorithm solves the problem best? Giraud-Carrier and Provost (2005) address whether ML can escape the limitations imposed by the NFL theorems. Their ultimate conclusion is that only techniques that learn about the problem distribution $P(f)$ can overcome the NFL theorems. Hence, ML can be useful in light of the NFL theorems.

Vilalta and Drissi (2002) introduce two important ML concepts. One is that of structured versus random problems. Structured problems have lower Kolmogorov complexity than random problems. This structure can be exploited by algorithms to solve these problems more efficiently. By this definition, random problems have no such structure that ML can exploit. Consequently, ML is limited (with respect to its effectiveness) to structured problems. In terms of entropy, structured problems may still require exponential space to represent. To see this let there be $|\mathcal{F}|$ functions represented by 2^n bits, then there are about $2^{\alpha\, n}$ compressible functions with a Kolmogorov complexity of $2^{\alpha\, n}$, for $\alpha < 1$. The more structured the problem domain, the more quickly the percentage of structured problems approaches zero. Therefore, the percentage of structured problems decays geometrically

with the size of $|\mathcal{F}|$. In other words, ML applies as long as most problems are considered irrelevant.

One may like to know how to measure one's success in selecting an algorithm. To do so, we first need a definition of success. Vilalta and Drissi (2002) introduce the concept of algorithm biases expressed as restrictions and rankings of potential hypotheses (the true f). In our formal notation, this means each algorithm assumes some $P(f)$. Then successfully selecting an algorithm means that the selected algorithm's assumed problem distribution matches the actual problem distribution. Both a theoretical and realistic measure exist to judge how well these match. Wolpert and Macready (1997) propose using the inner product between $P(f)$ and $P(d_m^y|f,m,a)$ over $f \in \mathcal{F}$ for a desired d_m^y. However, this method is only theoretical as computing the inner product is difficult due to the size of $\mathcal{F}$. A much more realistic measure is the off-training-set error (Wolpert, 2001).

We can use the answers to these three questions to lead us down a path of investigation. When ML samples from the problem distribution, the distributions are compressible (most problems are irrelevant), and the current method is suboptimal, then ML may operate within the NFL framework. Therefore, we will investigate how ML may manifest itself in the NFL framework. Second, due to the similarities between ML and ATSD, we will derive ATSD by applying an opposite goal to the opposite data. This in effect forms a "dual problem," which leads to our formulation of ATSD.

3. Anti-training

Section 2.3 gives us a research direction: derive ML from the NFL theorems, and subsequently, ATSD from that. To ensure our technique follows from (1), we begin by rewriting one side. First, we split $\mathcal{F}$ into three partitions (non-overlapping sets that cover the whole set):

$$\sum_{f\in\mathcal{F}} P\left(d_m^y|f,m,a\right) = \sum_{f\in\mathcal{F}_+} P\left(d_m^y|f,m,a\right) + \sum_{f\in\mathcal{F}_0} P\left(d_m^y|f,m,a\right) + \sum_{f\in\mathcal{F}_-} P\left(d_m^y|f,m,a\right). \quad (4)$$

Here $\mathcal{F}_+$, is the partition of problems already encountered, considered likely, or relevant; $\mathcal{F}_-$ represents the partition of problems that cannot occur (in the given context) or are considered unlikely to occur; and $\mathcal{F}_0$ consists of problems that are borderline likely or undefined. Equation 4 explicitly shows that we can trade-off performance between problem sets. Performance gained over $f \in \mathcal{F}_+$ must be lost over $f \in \mathcal{F}_0 \cup \mathcal{F}_-$.

The inquisitive reader may wonder why we use three partitions and particularly why we include the $\mathcal{F}_0$ partition. We include $\mathcal{F}_0$ for three reasons: first, an objective function does not always cleanly lie in either $\mathcal{F}_+$ or $\mathcal{F}_-$. This is because we avoid strict mathematical definitions of $\mathcal{F}_+$, $\mathcal{F}_0$, and $\mathcal{F}_-$. One may choose, for example, $\mathcal{F}_+ = \{f|P(f) > {}^{100}/_{|\mathcal{F}|}\}$ and $\mathcal{F}_- = \{f|P(f) < {}^{0.01}/_{|\mathcal{F}|}\}$. This flexibility necessitates $\mathcal{F}_0$. Second, $\mathcal{F}_0$ is useful for describing an important difference between ML and ATSD: the difference between the forthcoming (6) and (7). Third, when approximating $\mathcal{F}_+$ and $\mathcal{F}_-$, $\mathcal{F}_0$ is a catch-all for the remaining cases. Thus, three partitions are the minimum number needed to capture these degrees of freedom.

Equation 1 implies (4) is independent of a. However, each sum over a partition of $\mathcal{F}$ is generally allowed to change provided one important condition: $\mathcal{F}_+$, $\mathcal{F}_0$, and $\mathcal{F}_-$ are

sets of functions not closed under permutation.[2] Schumacher et al. (2001) show in their Lemmas 1 and 2 that the NFL theorems apply if and only if the set of functions is closed under permutation. Since the fraction of sets closed under permutation is exceedingly rare (Igel and Toussaint, 2003), the NFL theorems fail to apply, with high probability, to each partitioning of $\mathcal{F}$. In turn, this means that each sum is generally allowed to change, implying that it is possible to improve the performance over $\mathcal{F}_+$.

Let us extend (4) in two ways. First, multiply by $\Phi\left(d_m^y\right)$ (a performance metric such as final error) and bring it inside the sum. Second, sum several $\sum_{f\in\mathcal{F}} \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right)$ together for different values of d_m^y. Then we have

$$\sum_{d_m^y}\sum_{f\in\mathcal{F}} u = \sum_{d_m^y}\left(\sum_{f\in\mathcal{F}_+} u + \sum_{f\in\mathcal{F}_0} u + \sum_{f\in\mathcal{F}_-} u\right), \tag{5}$$

where

$$u = \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right).$$

3.1 Anti-training as Meta-learning

Recall that a type of ML taking into account the NFL theorems is the one that attempts to learn $P(f)$ directly (Giraud-Carrier and Provost, 2005). However, this is often infeasible due to the cardinality of $\mathcal{F}$. A simple alternative to approximate this method is to learn an optimizer that performs well on the empirically common functions. This is a meta-optimization problem to improve the performance over $\mathcal{F}_+$. More accurately, the meta-optimization will improve the conditional expected value of $\Phi\left(d_m^y\right)$. Without loss of generality, hence forth assume Φ is a performance metric to be minimized (*e.g.,* an error function). Then, improving (minimizing) the conditional expected value means:

$$a^\star = \arg\min_a \sum_{d_m^y}\sum_{f\in\mathcal{F}_+} \Phi\left(d_m^y\right)\hat{P}_+(f)P\left(d_m^y|f,m,a\right).$$

However, the above equation is neither governed by nor does it follow from the NFL theorems! The introduction of a function dependent on f, in this case the approximation $\hat{P}_+(f)$, generally voids the NFL requirement that (3) is independent of a. Nevertheless, if we consider the typical approximation of $\hat{P}_+(f)$ used in practice, we realize that rarely does an identical objective function occur twice. See Appendix A for further justification of this assertion. This means $\hat{P}_+(f)$ is practically going to be a two-valued function: 0 and $\left|\hat{P}_+(f)\right|^{-1}$. Thus, in practice we drop the term $\hat{P}_+(f)$. This recovers the first term on the right-hand-side of (5), which follows from the NFL theorems:

$$a^\star = \arg\min_a \sum_{d_m^y}\sum_{f\in\mathcal{F}_+} \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right). \tag{6}$$

2. A set of functions being closed under permutation means no new functions may be generated when the input space is bijectively remapped into itself (*i.e.,* a random permutation of the input space).

We make several more notes about the above equation. First, when using a deterministic a (or a random algorithm with a fixed random seed) $P\left(d_m^y|f,m,a\right)$ will be either 0 or 1; only a single d_m^y will be produced. This makes it easier to optimize in practice. Second, $\Phi\left(d_m^y\right)$ determines the behavior of $a^\star$; $a^\star$ could produce a low final error, low cumulative error, etc. While it is also possible to modify (6) to optimize over the cross-validation error, care must be taken as cross-validation alone is subject to the NFL theorem (Wolpert, 2001). Typically $\mathcal{F}_+$ will be approximated using samples from a problem distribution $P(f)$ arising from a limited domain. Additionally, (6) says nothing about the search for $a^\star$. Unless a is being modified intelligently, the meta-optimization problem will be subject to the NFL theorems itself. This means that finding better algorithms will be possible, but costly.

The NFL theorems say that any performance gained over a set of functions must be lost over the set's complement. Equation 4 must remain constant. Thus, ML improves the performance over $\mathcal{F}_+$ at the cost of performance over $\mathcal{F}_0 \cup \mathcal{F}_-$. This is potentially detrimental for prediction, generalization, and avoiding overfitting. To help mitigate this issue we now introduce *anti-training with sacrificial data* (ATSD). This means ATSD is about the optimization of one of the terms from (5):

$$a^\star = \arg\max_a \sum_{d_m^y}\sum_{f\in\mathcal{F}_-} \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right). \tag{7}$$

This translates into

$$a^\star = \arg\min_a \sum_{d_m^y}\sum_{f\in\mathcal{F}_0\cup\mathcal{F}_+} \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right).$$

Again, without loss of generality, we treat Φ as an error function to be minimized. As one can see, ATSD is an approximate dual of ML. ATSD achieves an improvement in the *sum* of performance over $\mathcal{F}_+ \cup \mathcal{F}_0$ by directly worsening the performance over $\mathcal{F}_-$. If the performance over $\mathcal{F}_-$ decreases, then by (4), the sum over $\mathcal{F}_0 \cup \mathcal{F}_+$ must improve. Despite ATSD being less efficient than ML, it has other benefits. For a more complete discussion, please see Sections 3.2 and 3.3.

It is critical to note that the performance improvement is over $\mathcal{F}_0 \cup \mathcal{F}_+$. Without additional care, ATSD could drop the performance over $\mathcal{F}_+$, but improve the performance over $\mathcal{F}_0$. To help avoid this, ATSD should be combined with ML (ATSD+ML). Combining (6) and (7), we have

$$a^\star = \arg\min_a \sum_{d_m^y}\left(\sum_{f\in\mathcal{F}_+} \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right) - \beta \sum_{f\in\mathcal{F}_-} \Phi\left(d_m^y\right) P\left(d_m^y|f,m,a\right)\right) \tag{8}$$

where $\beta \geq 0$ determines the relative strength of anti-training. The above optimization technique is just one straightforward approach to combining ML and ATSD. ATSD+ML can be achieved by using a meta-optimization technique similar to the aforementioned process for ML. According to Bergstra and Bengio (2012), a random search for $a^\star$ is in some sense optimal. Although, preliminary tests suggest that local searches find locally optimal continuous hyper-parameters on continuous domains.

We feel it is important to note that ATSD+ML is not strictly limited to operating over $\sum_{f\in\mathcal{F}_+}$ and $\sum_{f\in\mathcal{F}_-}$. Other statistics can be used, too. For instance, the mean reduces

to simply using (8) with an adjusted β. The median is another statistic that could be used. Meta-optimizing using the medians would be the same as dynamically selecting one or two f to improve or worsen their performance. For the same reason, percentiles are valid statistics to improve or degrade. Since percentiles could be used, one may wonder if the interquartile-range (IQR) is a valid statistic for ATSD+ML. ATSD+ML should avoid using the IQR statistic over $\mathcal{F}_-$, as this amounts to improving the performance over some $f \in \mathcal{F}_-$.

Algorithm 1 demonstrates the basic approach. The parameter *metaSearch* is a search algorithm for finding black-box search/optimization/learning algorithms; *metaMetric* is

Algorithm 1 Anti-Training Combined with Meta-Learning

Procedure Anti-Meta-Search(
metaSearch, *metaMetric*, *pMetric*, $\hat{\mathcal{F}}_+$, $\hat{\mathcal{F}}_-$, m)
$metaOut \leftarrow empty$
repeat
  $searchAlg \leftarrow metaSearch(metaOut)$
  $listPosPerfs \leftarrow empty$
  for all $objFunc \in \hat{\mathcal{F}}_+$ **do**
    $optResults \leftarrow$ Optimize($searchAlg$, $pMetric$, $objFunc$, m)
    Append $optResults \rightarrow listPosPerfs$ {Accumulate results from $\hat{\mathcal{F}}_+$}
  end for
  $listNegPerfs \leftarrow empty$
  for all $objFunc \in \hat{\mathcal{F}}_-$ **do**
    $optResults \leftarrow$ Optimize($searchAlg$, $pMetric$, $objFunc$, m)
    Append $optResults \rightarrow listNegPerfs$ {Accumulate results from $\hat{\mathcal{F}}_-$}
  end for
  $algPerformance \leftarrow$ metaMetric ($listPosPerfs, listNegPerfs$) {e.g., (8)}
  Append $(searchAlg, algPerformance) \rightarrow metaOut$
until $algPerformance$ is good enough
Return Argmin($metaOut$)

Procedure Optimize($searchAlg$, $performanceMetric$, $objFunc$, m)
$d_m \leftarrow empty$
for $itr = 1$ to $itr = m$ **do**
  $x \leftarrow searchAlg(d_m)$
  Append $(x) \rightarrow d_m^x$
  $y \leftarrow objFunc(d_m^x)$
  Append $(y) \rightarrow d_m^y$
  Append $(d_m^x, d_m^y) \rightarrow d_m$
  $performance \leftarrow performanceMetric(d_m)$
end for{Alternatively end when performance is good enough}
return $performance$ {If ending when performance is good enough return number of iterations}

the metric for preferring one black-box algorithm over another, such as (8); and *pMetric* is $\Phi\left(d_m^y\right)$, which rates the quality of outputs. The subroutine Optimize is a black-box search/optimization algorithm that runs the *searchAlg* to generate the input d_m^x into the black-box objective function *objFunc*.

Algorithm 1 starts by selecting, generating, or modifying a search/optimization/learning algorithm. For simplicity, let us assume the goal is optimization. The optimization subroutine generates the next input x from the history of inputs and outputs. The objective function (from $\hat{\mathcal{F}}_-$ or $\hat{\mathcal{F}}_+$) is evaluated at x. This process is repeated, building up the d_m vector. After enough iterations, the subroutine returns the performance metric, $\Phi\left(d_m^y\right)$. The performance over each $f \in \hat{\mathcal{F}}_+ \cup \hat{\mathcal{F}}_-$ is computed and stored using this subroutine. The meta-fitness function evaluates the optimization algorithm using ATSD+ML. This whole process is repeated for enough iterations or until the algorithm's performance is good enough.

3.2 Analysis

We now present how effective the ATSD is from a theoretical perspective. An extensive, empirical study follows in Section 5. It is difficult to analyze ATSD without making strong assumptions. Thus, we make the following assumptions as general as possible and defend them using logic similar to that used in the original NFL theorems (Wolpert and Macready, 1997).

Allow an algorithm to be represented by its learning biases as discussed by Vilalta and Drissi (2002). We choose to interpret these biases as an *a priori* rank ordering of functions. To help illustrate this point, imagine an algorithm represented by a deck of cards, where each card represents an objective function f to be optimized. Denote $R_a(f) \in \left\{1, 2, \ldots, |\mathcal{F}|\right\}$ as algorithm a's ranking (i.e., position in the deck) of f. A lower ranking means the algorithm is more biased toward finding the optimal solution in fewer iterations. The function with rank one will have its optimum input guessed first. Then all $f \in \mathcal{F}$ in violation of the first input-output pair, d_1, are removed from the deck of hypotheses and the process is repeated.

Allow $\hat{\mathcal{F}}_+$ and $\hat{\mathcal{F}}_-$ to be samples from $\mathcal{F}_+$ and $\mathcal{F}_-$ respectively. Furthermore, since we are working with domain specific optimizers, we may assume the majority of problems are not highly relevant:

$$\left|\mathcal{F}_+\right| \ll |\mathcal{F}_0|$$
$$\left|\mathcal{F}_+\right| \ll \left|\mathcal{F}_-\right|.$$

Rather than debate the percent $f \in \mathcal{F}_-$, allow

$$|\mathcal{F}| = c\left|\mathcal{F}_-\right|,$$

for a constant c. For example, if $\left|\mathcal{F}_-\right| \approx |\mathcal{F}_0|$ then $c \approx 2.0$. Nevertheless, we will continue to use c symbolically to remain more flexible and general, allowing the reader to impose their own assumptions.

We will assume that the learning tasks are compressible, as this is necessary for meta-learning to work (Vilalta and Drissi, 2002). We will interpret compressibility as implying a correlation between rankings of functions. Particularly, if through meta-optimization some

$f \in \hat{\mathcal{F}}_+$ lowers in rank, then on average allow all $g \in \mathcal{F}_+$ to lower proportionally by a factor of ρ_+.[3] This will provide the average behavior for describing why ML works. To capture the corresponding effects of ATSD, when some $f' \in \hat{\mathcal{F}}_-$ rises in rank due to meta-optimization, allow all $g' \in \mathcal{F}_-$ to rise proportionally by a factor of ρ_-. We use the symbols g and g' to denote objective functions that are "dragged" along due to correlations in rankings. The two factors, ρ_+ and ρ_-, are functions of the compressibility of $\hat{\mathcal{F}}_+$ and $\hat{\mathcal{F}}_-$ respectively.

Assume that the meta-optimization task improves/degrades the rankings by p_c (percent change). That is, for $f \in \hat{\mathcal{F}}_+$ and $f \in \hat{\mathcal{F}}_-$ the ranking changes from $R_a(f)$ to

$$R_a\left(f\right)\left(1-p_c\right)+p_c, \text{ and}$$
$$\left|\mathcal{F}\right| p_c + R_a\left(f\right)\left(1-p_c\right)$$

respectively. Thus the differences in rank after ML and subsequently ATSD are approximately

$$\left(R_a\left(f\right)-1\right) p_c, \text{ and}$$
$$\left(\left|\mathcal{F}\right| - R_a\left(f\right)\right) p_c.$$

If we assume the starting algorithm is chosen at random, then we can assume the rankings are uniformly distributed: $P(R_a(f)) \sim U\left(1, \left|\mathcal{F}\right|\right)$. This follows from the same reasoning Wolpert and Macready (1997) use to argue that $P(f)$ ought to assume a uniform distribution when lacking further information. Then

$$E\left[R_a\left(f\right)\right] = \frac{\left|\mathcal{F}\right|+1}{2}$$

Now, we will proceed to show that the expected rank improvements from ML and ATSD are comparable. After accounting for rank correlations, improving a single $f \in \hat{\mathcal{F}}_+$ produces on average

$$\mu_+\left(f\right) = p_c\left(R_a\left(f\right) - 1 - \left(R_a(f)-1\right)\rho_+ \sum_{g\in\mathcal{F}_+}\left(R_a(g)-1\right)\rho_+\right)$$

rank improvements. The term $\left(R_a(f)-1\right)\rho_+$ is subtracted to avoid double counting f as $f \in \mathcal{F}_+$, and the term $R_a(g)|g \in \mathcal{F}_+$ is any function that may have its rank improved due to correlations between problems. Substituting in the expected values of $R_a(f)$ and $R_a(g)$ produces

$$\begin{aligned} E\left[\mu_+\left(f\right)\right] &= p_c\left(\frac{\left|\mathcal{F}\right|-1}{2}\left(1-\rho_+\right) + \frac{\left|\mathcal{F}\right|-1}{2}\left|\mathcal{F}_+\right|\rho_+\right) \\ &= p_c\left(\frac{\left|\mathcal{F}\right|-1}{2}\right)\left(1+\left(\left|\mathcal{F}_+\right|-1\right)\rho_+\right). \end{aligned} \tag{9}$$

3. Since the effects of correlation are modeled in a deterministic fashion, this model can only be used to assess the expected values and not higher moments, such as variance.

Next we will compute the number of ranks degraded during ATSD. After accounting for correlations, but being careful to avoid double counting f', the number of ranks degraded on average for anti-training a single $f' \in \hat{\mathcal{F}}_-$ is

$$\mu_-\left(f'\right) = p_c\left(\left(\left|\mathcal{F}\right| - R_a\left(f'\right)\right)\left(1-\rho_-\right) + \sum_{g'\in\mathcal{F}_-}\left(\left|\mathcal{F}\right| - R_a\left(g'\right)\right)\rho_-\right).$$

Substituting in the expected values of $R_a(f')$ and $R_a(g')$ produces

$$E\left[\mu_-\left(f'\right)\right] = p_c\left(\frac{\left|\mathcal{F}\right|-1}{2}\right)\left(1+\left(\left|\mathcal{F}_-\right|-1\right)\rho_-\right). \tag{10}$$

We cannot directly compare (9) and (10) yet. Equation 10 is in terms of ranks degraded for $f' \in \mathcal{F}_-$, not ranks improved for $f \in \mathcal{F}_+$. We can calculate the expected number of $f \in \mathcal{F}_+$ promotions as a result of anti-training demotions by multiplying (10) by the ratio $\left|\mathcal{F}_+\right| / \left|\mathcal{F}\right|$. Recalling $\left|\mathcal{F}\right| = c\left|\mathcal{F}_-\right|$, we have

$$E\left[\mu'_-(f)\right] = p_c\left(\frac{\left|\mathcal{F}\right|-1}{2}\right)\left(\zeta + \left(\frac{\left|\mathcal{F}_+\right|}{c} - \zeta\right)\rho_-\right)$$

where $\zeta = \frac{\left|\mathcal{F}_+\right|}{c\left|\mathcal{F}_-\right|}$. Thus the relative power of ATSD versus ML is given by

$$\frac{E\left[\mu'_-(f)\right]}{E\left[\mu_+(f)\right]} = \frac{\left(\zeta + \left(\frac{\left|\mathcal{F}_+\right|}{c} - \zeta\right)\rho_-\right)}{\left(1+\left(\left|\mathcal{F}_+\right|-1\right)\rho_+\right)}. \tag{11}$$

Note $\zeta = \frac{\left|\mathcal{F}_+\right|}{\left|\mathcal{F}\right|} \approx 0$ by assumption. Provided a sufficiently large $\left|\mathcal{F}_+\right|$, we are justified in dropping additive terms on the order of 1 or less. Then (11) becomes

$$\frac{E\left[\mu'_-\right]}{E\left[\mu_+\right]} \approx \frac{\rho_-}{c\,\rho_+}. \tag{12}$$

If we allow ρ_- and ρ_+ to be on the same order of magnitude and let $c \approx 2.0$, then ATSD will be roughly half as powerful per training sample as ML.

3.3 Other Considerations

Here, we consider several questions about ATSD+ML. Can performance over some elements in $\mathcal{F}_-$ decrease, only to increase performance on other elements in $\mathcal{F}_-$? What happens if $\mathcal{F}_-$ is specified incorrectly? What benefits does ATSD+ML have over traditional ML?

Even though performance over the sacrificial data $\mathcal{F}_-$ is minimized, there is no reason to believe that all $f \in \mathcal{F}_-$ always perform worse after optimization. Yes, performance on

some elements in $\mathcal{F}_-$ may increase. However, ATSD+ML only requires the *net* performance over $f \in \mathcal{F}_-$ to decrease. There are two things that can make the net performance over $\mathcal{F}_-$ improve during the meta-optimization procedure (8). First, $\mathcal{F}_-$ may be too "similar" to $\mathcal{F}_+$, meaning the algorithms being investigated treat both sets of problems similarly. This can occur when the search for $a^\star$ is incomplete, such as when searching only for an optimizer's hyper-parameters instead of program code. The fixed program code may be biased toward treating both sets similarly. Alternatively, the β value in (8) needs to be greater.

ATSD's behavior is similar to ML's behavior when the data sets are specified incorrectly. To the contrary, ATSD+ML is more robust than ML when $\hat{\mathcal{F}}_-$ is specified correctly. Consider the case when $\hat{\mathcal{F}}_+$ poorly approximates $\mathcal{F}_+$. If the approximation is sufficiently poor, performance over $\mathcal{F}_+$ will decrease and performance over $\mathcal{F}_-$ will change randomly. ATSD+ML helps ensure a drop in performance over $\mathcal{F}_-$, regardless. This is assuming $\hat{\mathcal{F}}_-$ is specified correctly, which is generally easy. Section 4 addresses this issue. Alternatively, consider the scenario where $\hat{\mathcal{F}}_+ = \hat{\mathcal{F}}_-$. One of three things may happen according to (8). If $\beta < 1$, then the behavior defaults back to traditional ML. If $\beta = 1$, all algorithms appear equal and nothing happens. If $\beta > 1$, then the performance $\hat{\mathcal{F}}_+$ worsens. In general, even when $\hat{\mathcal{F}}_+ \neq \hat{\mathcal{F}}_-$, a safe value for β is $\beta < \left|\hat{\mathcal{F}}_+\right| / \left|\hat{\mathcal{F}}_-\right|$, as this ensures the traditional ML term dominates the meta-optimization procedure. However, to strictly follow the NFL derivation one should set $\beta = \left|\hat{\mathcal{F}}_+\right| / \left|\hat{\mathcal{F}}_-\right|$ so that each objective function is weighted equally.

ATSD's use of $\mathcal{F}_-$ helps define where the learning algorithm will fail. Specifying some of the failure cases removes some unknown failure cases. This follows from the fact, as shown by Schumacher et al. (2001), that all algorithms produce the same collection of d_m^y when all functions are considered. Also, in theory when $\mathcal{F}_-$ includes a model of noise, then the algorithm learns to fail at fitting the noise. This is entirely different from using a less powerful model that cannot represent the noise. The algorithm could still represent the noise, but ATSD would make it more difficult for the algorithm to find models that do so.

ATSD also helps when data is hard to come by. When real problems or data are scarce or expensive, ML is limited. Just like normal data fitting, if few data points are provided then overfitting is problematic. However, $\mathcal{F}_-$ can be easy and cheap to generate (for more on this see Section 4). As long as there is a balance between preserving/maximizing performance over a small $\mathcal{F}_+$ and minimizing the performance over a large $\mathcal{F}_-$, then this should help avoid overfitting $\mathcal{F}_+$.

4. Generation of Sacrificial Data

In this section, we present five methods for generating sacrificial data (*i.e.,* $f \in \mathcal{F}_-$). We classify these methods into three categories: randomized, empirical, and theoretical. First, we discuss the possibility and consequences of generating the sacrificial data purely randomly. Yet randomly generated sacrificial data may still lie in $\mathcal{F}_+$, so next we discuss rejecting randomly generated sacrificial data that is too similar to $\hat{\mathcal{F}}_+$ (using confidence intervals or other measures of similarity). A similar approach starts with functions in $\mathcal{F}_+$ and "destroys" patterns that are initially present. A different approach is to use empirical "pure" noise. Last, we analyze the efficacy of using a model similar to the source for $f \in \mathcal{F}_+$, but changing the model to violate logical/physical laws, use unlikely distributions,

etc. Since $\mathcal{F}_-$ will usually be large, $\hat{\mathcal{F}}_-$ may be generated with any combination of these techniques.

One technique is to randomly generate functions. Surprisingly, this method works with a high probability of success given a sufficiently large $|\mathcal{F}|$. Vilalta and Drissi (2002) conjecture that failing to learn unstructured tasks (*i.e.,* tasks with a large relative Kolmogorov complexity) carries no negative consequences. As discussed in Section 2.3, the percentage of structured problems decays geometrically with the size of $|\mathcal{F}|$. Thus, for a sufficiently large problem space, randomly generated objective functions will be, with high probability, unstructured.[4] Therefore, we can exploit structure present in learning tasks with high probability, even without knowing the structure. Ergo, sacrificial data can be randomly generated for problems with sufficiently large input and output spaces. While randomly generated functions may work with high probability, they may be a poor choice for the sacrificial data. Consider the result given by (12). Due to the nature of randomly generated functions, they will probably have larger Kolmogorov complexity and hence a smaller ρ_- than other methods.

If randomly generated sacrificial data is too similar to a problem in $\hat{\mathcal{F}}_+$, it may be rejected. This can be accomplished with confidence intervals or other measures of similarity. Suppose the objective function's output is suspected to follow an unknown ergodic process. Since ergodic processes have estimable statistical properties (by definition), confidence intervals can be constructed on these statistics. New random functions can be generated to have their output lie outside these confidence intervals. One could analyze other metrics besides the functions' outputs, such as correlations, covariance, and mutual information between the input and output. By appropriately selecting the confidence intervals, the randomly generated functions can be in $\mathcal{F}_-$ with arbitrary confidence. However, randomly generating functions will (probabilistically) have a small ρ_-.

To improve ρ_-, one approach would be to take problems from $\hat{\mathcal{F}}_+$ and modify the functions with a few simple steps. This will barely alter the Kolmogorov complexity, by its very definition (*i.e.,* minimally sized program to generate the data). Yet, this approach may be ineffective unless these modifications are "natural." Since the Kolmogorov complexity is dependent on the description language (Li and Vitányi, 2008) and the description language is dependent on the optimizer, the types of modifications should somehow match the possible optimizers. This is what we mean when we say "natural." For instance, consider an attempt to determine an optimal step-size for a local search over a set of smooth objective functions. Generating $\hat{\mathcal{F}}_-$ using simple pseudo-randoms would be "unnatural," and possibly result in a minuscule ρ_- in (12). Whereas, scaling the smooth functions may be more useful for generating $\hat{\mathcal{F}}_-$. Again, to ensure the modified functions belong to $\mathcal{F}_-$, the modifications should destroy any patterns originally present, modify correlations, and alter other metrics.

For objective functions that are data-based, another technique is to generate objective functions $f \in \mathcal{F}_-$ from signal-less empirical noise. Most would agree that a learner failing to see patterns in empirical noise is a positive quality. If it is expensive to collect large quantities of empirical noise, then pseudo-random noise could be generated from a random process modeling the empirical noise. If the empirical noise cannot be measured

4. Hardware random generators make it easier to generate functions with high Kolmogorov complexity.

Table 1: Experiment summary

Name	Objective type	Meta-iterations	Trials × subtrials	Tests	Hypothesis
Satellite Trajectory	Continuous optimization	≈ 900	4 × 4	100	ATSD works
High-dimensional TSP	Discrete optimization	10,000	5 × 4	10,000	ATSD fails
Low-dimensional TSP	Discrete optimization	2,000	4 × 200	400	ATSD works
$50k Classification	Learning with a SVM	3,500	10 × 1	1	ATSD works

directly, time-series analysis offers standard procedures for isolating noise from data carrying trends (Gershenfeld, 1999; Brockwell and Davis, 2002; Shumway and Stoffer, 2011).

The last procedure for generating $\hat{\mathcal{F}}_-$ makes use of models, simulations, or subject matter experts (SMEs). The models or simulations can be altered to contradict logical or physical laws. Replacing typical distributions with unlikely or impossible distributions can likewise lead to $f \in \hat{\mathcal{F}}_-$. Those $f \in \mathcal{F}_-$ that come from models and simulations will tend to have lower Kolmogorov complexity than randomly generated functions. We conjecture that this translates into a larger ρ_-. Even without a simulation, SMEs can help provide high quality $\mathcal{F}_-$ due to their knowledge of what is likely, unlikely, and impossible. Furthermore, models, simulations, and SMEs provide an implicit distribution of $P(f)$. Yet, the downside to this approach is that it takes a significant amount of knowledge about the domain from which the problems come, so it might be impractical. This approach is most useful when much is known about the problems, but little is known about which optimizer or learner to use.

5. Experiments

To help verify the theory and investigate the possible interactions between traditional ML and ATSD, we conducted four, very extensive experiments mimicking how meta-optimization procedures are used in practice. The first experiment focuses on finding an efficient algorithm for optimizing a satellite's trajectory to monitor space debris. The second experiment involves the search for an algorithm that performs well on highly random traveling salesperson problems (TSP). The third experiment is also a TSP, but this time the cities occur on a ring and their distances are calculated from that. The last experiment tries to maximize the predictive accuracy on a classification problem. We summarize these experiments, their objectives, size, and purpose in Table 1.

In Table 1 "objective type" shows what the meta-optimization process was optimizing. The column titled "meta-iterations" describes how many unique algorithms were tried in the search for the best algorithm for the task. "Trials × subtrials" describes how many ML and ATSD trials were conducted. Specifically, a trial refers to a specific configuration with a fixed $\hat{\mathcal{F}}_+$ and $\hat{\mathcal{F}}_-$. Subtrials were conducted to capture the variability of the meta-

optimized algorithms found after the specified number of meta-iterations. Each resulting algorithm produced by the meta-optimization process was then tested across the number of tests listed in the "tests" column. Lastly, the column titled "hypothesis," describes our hypothesized outcomes according to theory.

Throughout the course of conducting the experiments, we discovered multiple possible ways the meta-optimization process may fail, for both ML and ATSD. When the search for optimizer algorithms or supervised learning algorithms is constrained (*e.g.,* only adjusting hyper-parameters), the algorithms may fail to exploit structure, which is embedded in the problem. This was demonstrated with the high-dimensional TSP; it has structure that the investigated optimizers have no capabilities to exploit. Another way for the process to fail is when the number of unique function evaluations is not fixed. Unless the number of unique function evaluations is fixed, some algorithms will use fewer unique function evaluations on problems from $\mathcal{F}_-$. The next problem is two fold: (1) the meta-optimization may fail to return a near optimal algorithm for the approximate problem distribution ($\hat{\mathcal{F}}_+$ and $\hat{\mathcal{F}}_-$), and (2) this approximation of the problem distribution may be poor. Thus, the accuracy of the approximation and the quality of the meta-optimization solution must be balanced;[5] the optimal algorithm for a poor approximation is undesirable. These problems affect both ML and ATSD alike.

Below we describe each experiment in more detail, including the domain where the objective functions come from, the meta-optimization procedure, which common ML practices we follow, individual tests within the experiment, and the computer setup. Results for each experiment are presented in Section 6. Just a quick note before delving into the first experiment. Whenever we say something is "low-level," we are referring to the base optimization or learning problem. Similarly, if we mention something is "meta-level" we are referring to the meta-optimization required for ML and ATSD that optimizes the "low-level" algorithm.

5.1 Satellite Trajectory Problem

For the first experiment we use a physics based domain where the pilot problems involve monitoring space debris around a small artificial planet. Monitoring space debris presents a trade-off between the quality of observations and the time until revisiting the same debris. If the satellite were to have nearly the same orbit as a single piece of debris, then the satellite would pass by the debris slower and closer, resulting in higher quality observations. Similarly, a different orbit that is closer to or further from the planet would result in seeing the debris more frequently, but from further away. The low-level optimization objective is to find an orbit for a satellite that visits all debris within a specified distance and with minimal mean-time between visits.

We simplify the domain for the experiment in the following ways. Gravity of the planet is assumed to be uniform. Space debris are treated as points, meaning they have no physical size. Collisions are ignored. We use only 10 pieces of space debris to minimize simulation time. All orbits are in a two-dimensional space. Thus, an orbit's degrees of freedom are its: eccentricity (shape), orientation (angle of the ellipse), semi-major axis (size), and true

5. One may liken this scenario to the processing inequality in information theory.

Table 2: Comparison of Pilot Problem Distributions

Data Set	$\mathcal{F}_+$	$\mathcal{F}_-$
Semi-Major Axis	$\sim N(400km, 30km)$	half from $\sim N(520km, 30km)$ half from $\sim N(280km, 30km)$
Eccentricity	$\sim N(0.1, 0.1)$	$\sim Exp(1/3)$, but ≤ 1.0
True Anomaly	$\sim U(0, 2\pi)$	$\sim U(\pi/4, 7\,\pi/4)$
Orientation	$\sim U(0, 2\pi)$	$\sim U(\pi/2, 2\,\pi)$

anomaly (phase of the orbit). In spite of these simplifications and the fact all the variables are continuous, the multiple pieces of space debris introduce many local extrema.

Recall that our methods discussed in Section 3.1 require multiple objective functions. We create multiple instantiations of problems from this domain. Each problem has space debris in a different configuration, but drawn from one of two distributions. These distributions correspond to $\mathcal{F}_+$ and $\mathcal{F}_-$. More details can be found in Table 2. To ensure there were no large gaps, we used Latin hypercube sampling (McKay et al., 1979).

5.1.1 Meta-Optimization Procedure

The meta-level problem that we investigate is the meta-optimization of an optimizer. We chose to use Matlab's simulated annealing (SA) procedure as our low-level optimizer. SA was chosen for this experiment due to its large number of hyper-parameters and behavior dependent upon these hyper-parameters (Ingber, 1993, 1996). The hyper-parameters we investigate are:

- step length function (two choices),
- initial temperatures (four real numbers),
- cooling function (three choices),
- reannealing time (one integer),
- upper bounds (four real numbers), and
- lower bounds (four real numbers).

The meta-optimization procedure is designed to find an optimizer that quickly minimizes the mean-time between the satellite visiting debris and the satellite's closest approach to each piece of debris. It does this by searching for good values for the above hyper-parameters. We searched for hyper-parameters that produce a low cumulative objective value, meaning finding a lower objective value sooner was preferred (*i.e.,* $\Phi\left(d_m^y\right) = \sum_{i=1}^{m} \min\{d_i^y\}$).

The meta-optimization consisted of three phases: differentiation, SA optimization, and a local search. The starting hyper-parameters were differentiated using SA for 35 iterations. While we recorded only the best solution during these 35 iterations, all remaining optimization iterations were recorded. Then, SA was used to further investigate the hyper-parameters, due to its global search properties.[6] Last, a gradient descent method (Matlab's

6. We acknowledge the NFL-theorems apply to meta-search.

Table 3: Four Types of Trials

Methods	ATSD+ML3	ML9	ML3	Control
$\hat{\mathcal{F}}_+$	3	9	3	1
$\hat{\mathcal{F}}_-$	9	0	0	0
Subtrials	4	4	1	1
Control?	No	No	Yes	Yes

`fmincon`) was used to finish the search process. We discovered that gradient descent worked well to search for real-valued hyper-parameters on this particular problem. The meta-optimization problem was setup to minimize the median of $\hat{\mathcal{F}}_+$ and maximize the median of $\hat{\mathcal{F}}_-$. We used $\beta = 1/3$, because our ATSD+ML test uses three times as many sacrificial functions as probable functions.

5.1.2 Practices Followed

One common practice involves searching for hyper-parameters (e.g., model selection) to minimize the bootstrap- or cross-validation error (Kohavi, 1995; Arlot and Celisse, 2010). Following these practices, we tested ATSD+ML according to (8), to search for good hyper-parameters for an optimization procedure on a collection of pilot problems. However, our experiment differs in three key aspects from common practice. First, we only test (i.e., judge the final performance) against data the optimizer has never seen, as stipulated by Wolpert (2001); Giraud-Carrier and Provost (2005). Second, the meta-optimization occurs across multiple objective functions from a limited domain, as advised by Giraud-Carrier and Provost (2005). Another common practice we purposefully chose to ignore is the partitioning of the data into training, probe/validation, and test partitions. We only use training and test partitions. The reason for this is the need to test if ATSD+ML limits the effects of overtraining. We test the algorithms from only the final iteration for the same reason.

5.1.3 Trials

We conduct four types of trials, summarized in Table 3: ATSD+ML3, ML with three samples (ML3), ML with nine samples (ML9), and a control where no ML nor ATSD are applied. ML9's $\hat{\mathcal{F}}_+$ consisted of nine sample problems. ML3 and ATSD+ML3 share a subset of ML9's $\hat{\mathcal{F}}_+$. The control uses one problem from the set ATSD+ML3 and ML3 share. We chose these trials to demonstrate how performance can be improved. The ML3 performance could be improved by either increasing the number of samples from $\mathcal{F}_+$ (ML9) or by introducing ATSD (ATSD+ML3). Both these alternatives are tested four times. Each algorithm generated from the table is then tested against 100 samples of never seen before data from the $\mathcal{F}_+$ distribution. We leave the analysis over $\mathcal{F}_0$ for another experiment.

5.1.4 Experiment Computer Setup

We ran the experiments using a cluster of three standard-grade (as of the year 2008) computers. One computer is a Gateway E6610Q model, using the Intel QX6700 quad-core processor. Two other computers are custom built, one using the Intel Q6600 quad-core

processor and the third using the Intel i7-3770K quad-core processor. All computers have 4GB of RAM (DDR2 800, DDR3 1033, DDR3 1600). All file I/O is carried out using a virtual drive hosted over the Internet. Both results and computations that could be reused are saved to the virtual drive. Two of the three computers run Matlab on Windows 7. The third computer runs Matlab in Ubuntu.

We acknowledge that this hardware is now dated and that there are better suited environments that should, and will, be used in the future. However, due to the lengthy process to gain access to high performance computing clusters, we only use a small cluster here. Anyone wishing to reproduce the results or to extend them will easily be able to use the coarse-grain parallelism that we built into our code to exploit larger clusters.

The meta-optimization code is written to exploit coarse-grain parallelism, to take advantage of each core available, while avoiding the need to use Matlab's parallel computing toolbox. Each computer performs meta-optimization for one of the ATSD+ML3, ML9, and ML3 trials. To keep one computer responsive for daily tasks, only one trial was performed on ML3. The control case did not use any meta-optimization.

The meta-optimization process takes about 1.5 months to finish for the ML3 experiment, about 4.5 months to finish the ML9 experiment, and approximately 6.0 months to complete the ATSD+ML3 experiment. After six months, we collect about 1.7 GB of data.[7] After the meta-optimization, we run the algorithms over 100 different problems from the $\mathcal{F}_+$ to collect statistical performance data. Each algorithm takes about 10 hours to run over this set of 100 problems. Due to how relatively quick it is to test an algorithm over 100 problems, we did not need to partition the trials over the cluster. This generates another 1.4 GB of data over the course of a week. All 3.1 GB of the data is available, but we also offer just the Matlab scripts (only 100KB) that can generate statistically similar data.

Because the meta-optimization run times were deemed too slow to reproduce statistically similar results, we found ways to improve the execution speed. Since the core kernel of the simulation is already optimized and Matlab's ODE45 was still the bottleneck (taking over 98% of the time), we copied the ODE45 code and removed all the code for options. Further improvements were made by turning off Matlab's ODE45's interpolation, as we were already interpolating ODE45 results at specific points.

These changes give us approximately a 2x speedup in total. Trimming the ODE45 code improves the speed by about 50%. We are surprised how much this helps; we attribute it to avoiding the majority of the ODE45's if-branches, removing its switch statement, and reduction in code size. Turning off Matlab's ODE45's interpolation provides another 30% speed up. With modern hardware and these revisions, it should take only approximately two months to produce statistically similar results.

5.2 High Dimensional Traveling Salesperson Problems

In this experiment we try to demonstrate that theory correctly predicts when ATSD should fail. Theory predicts that if ML fails, then so should ATSD. We believe this is an important experiment as it tries to disprove the theory from the opposite direction; we would be

7. Most of the data was saved computation to avoid resimulating space debris trajectories and check-pointing to prevent loss of data in advent of system crashes. Unfortunately some of the earlier data was lost, as the check-pointing system was implemented after the first system crash.

Table 4: Correlation Matrix by Problem Distribution

	$\mathcal{F}_+$	$\mathcal{F}_0$	$\mathcal{F}_-$
$\Sigma_{i,j} =$	$\frac{2.82}{(1+\lvert i-j\rvert)^2}$	$\frac{0.5}{\lvert i-j\rvert^2}$	$\delta(i,j)$

surprised to find that ATSD works where ML fails. ML and ATSD require structure in the input-output pairs. Structure, only in the problem itself but not the input-output pairs, is of little use for an optimizer or learner unless one is attempting to discover the underlying problem (*e.g.,* the TSP cost matrix) and then solve it with a non-black-box solver. This is actually one method discussed in (Culberson, 1998).

For all TSP problems in this experiment, we limit the input to only valid sequences of cities. Thus, the entire input space is valid. Only total distance traveled is analyzed. No information from partial solutions is used. The problem distributions are generated with the following technique:

- For each row of the cost matrix, sample a 20 dimensional multivariate normal distribution.
- Compose a cost matrix from 20 such samples.
- Make the cost matrix symmetric by adding it to its own transpose.
- The minimum value is set to zero.

The covariance matrix is defined in Table 4. $\mathcal{F}_+$ has a nearly singular covariance matrix, producing 20 random numbers, which approximately follow a random walk.[8] Before making the cost matrix symmetric, each row is an independent sample. This structure is not readily exploitable by the low-level optimizer. The low-level optimizer avoids analyzing trends in these random walks. $\mathcal{F}_-$ produces a random cost matrix where each element is independently normally distributed. We will assume this is sufficient to make the probability of particular outputs independent of the inputs, implying $P(f) = \prod_x P(y = f(x))$.

5.2.1 Meta-Optimization Procedure

The meta-optimization problem is to find the hyper-parameters for a custom genetic algorithm. The hyper parameters include five initial paths, a mutation rate, a max swap-size schedule, and a swap-size parameter. The initial paths and mutation rate should be self-explanatory. The swap-size parameter defines a random variable distributed according to the exponential distribution, but this value is clamped to never exceed the max swap-size. The max swap-size schedule provides 10 maximum values, each one valid for 10% of the low-level optimization procedure. The clamped exponentially distributed swap-size determines how many cities are swapped at once.

The custom genetic algorithm primarily differs from a traditional genetic algorithm in that it only generates valid inputs for the TSP problems. Random mutations are implemented as random swaps. The crossover operation uses the differences in the paths

8. It is not technically a random walk, but when plotted the values resemble a random walk.

Table 5: Five Types of Trials

Methods	ML5	ML10	ML15	ATSD250+ML5	ATSD250+ML5 Weak
$\left\|\hat{\mathcal{F}}_+\right\|$	5	10	15	5	5
$\left\|\hat{\mathcal{F}}_-\right\|$	0	0	0	250	250
Subtrials	4	4	4	4	4
β	0	0	0	1	0.2

of the two top performing individuals to guide which cities to swap. We use a population size of five. At the time the experiment was run, the implementation permitted duplicate individuals. This means it is probable that the number of unique function evaluations differ for all trials; this is significant since any subtrial may be put at a disadvantage by exploring fewer possible solutions. However, considering the cost to rerun the experiment, we still present the results. A future experiment should retest this experiment with the number of unique function evaluations fixed. The number of unique function evaluations is made more consistent for the ring-based TSP experiment.

We chose to use Matlab's SA algorithm to perform the meta-optimization, primarily due to its global search property. Any global search meta-optimization routine could be used instead. Instead of using (8) verbatim, we replace the sums with averages. This allows us to vary $\left|\hat{\mathcal{F}}_+\right|$ and $\left|\hat{\mathcal{F}}_-\right|$ while continuing to give the same relative importance β.

5.2.2 Practices Followed

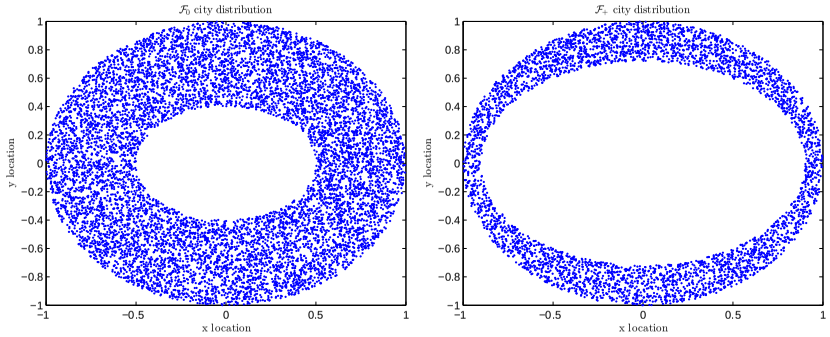
We followed the same practices as discussed in Section 5.1.2.

5.2.3 Trials

We conduct five types of trials, outlined in Table 5: ML5, ML10, ML15, ATSD250+ML5, and ATSD250+ML5 Weak. ML5 can be viewed as the control. We chose the trials ML5, ML10, and ML15 to demonstrate the effects of increasing $\left|\hat{\mathcal{F}}_+\right|$ on a domain where little to no patterns should be exploitable. The ATSD+ML trials, ATSD250+ML5 and ATSD250+ML5 Weak, demonstrate the effects of using random TSP problems for sacrificial data at differing weights of ML compared to ATSD.

Four subtrials are conducted per trial, all using the same $\hat{\mathcal{F}}_+$, $\hat{\mathcal{F}}_-$, or both. The subtrials only differ in the random seed used in the meta-optimization process (*i.e.,* the simulated annealing random seed). This is designed to roughly estimate the variability of the quality of the algorithms returned from the meta-optimization process. Since we hypothesize ATSD and ML should fail, we want to give every chance for ML and ATSD to work. So we use 10,000 simulated annealing iterations to have a greater chance to exploit structure that only works on $\hat{\mathcal{F}}_+$.

We test each resulting algorithm (a total of 20 algorithms) against 10,000 samples of never seen before data from the $\mathcal{F}_+$, $\mathcal{F}_0$, and $\mathcal{F}_-$ distributions.



(a) The $\mathcal{F}_0$ city distribution (b) The $\mathcal{F}_+$ city distribution

Figure 1: City distributions by problem distribution

5.2.4 Computer setup

The computer setup is very similar to that described in Section 5.1.4. We only use the first two computers as this experiment ran faster, taking only about three days to complete. We use the same Internet based virtual drive for all I/O. The meta-optimization process generates about 963 MB of data including checkpointing every 50 iterations. The code is written to exploit coarse-grain parallelism, scaling up to about 20 instances of Matlab (avoids using the Matlab parallel computing toolbox).

5.3 Traveling Salesperson on a Ring

In this experiment, we investigate ATSD and ML over a discrete optimization problem where multiple properties from the domain should be exploitable. This experiment is in contrast to the high dimensional TSP. As in the previous experiment, we limit the input to only valid sequences of cities. Thus, all input is valid. Only the total distance traveled is analyzed. No information from partial solutions is used. However, rather than generate a correlated cost matrix from random values directly, the cities are randomly placed on a ring and their cost values are computed as the Euclidean distance between cities (*cf.* Figure 1). As the ring's inner and outer diameter become the same, the optimal path converges to traveling in a circular loop. Furthermore, cities are labeled in a clockwise fashion.

The problem distribution comes from the following. $\mathcal{F}_+$ comes from a narrow ring (*cf.* Figure 1b). $\mathcal{F}_0$ comes from a wide ring (*cf.* Figure 1a). $\mathcal{F}_-$ are TSP problems with randomly generated cost matrices, similar to the $\mathcal{F}_-$ in the previous experiment (the high dimensional TSP experiment).

5.3.1 Meta-Optimization Procedure

The meta-optimization steps are similar to the high dimensional TSP's meta-optimization case. We investigate the same hyper-parameters as before: initial paths, mutation rate, max

Table 6: Four Types of Trials

Methods	ML5	ML10	ATSD15+ML5	ATSD50+ML5
$\left\|\hat{\mathcal{F}}_+\right\|$	5	10	5	5
$\left\|\hat{\mathcal{F}}_-\right\|$	0	0	15	50
Subtrials	200	200	200	200
β	0	0	0.2	0.2

swap-size schedule, and swap-size parameter. We use the same custom genetic algorithm as before, with two exceptions. First, we modify the low-level algorithm to force all individuals in the population to represent unique paths. Duplicate paths are replaced with random paths. Because this slows down the low-level algorithm to about half its original speed, we further optimize the low-level algorithm to nearly restore its original performance. We still use Matlab's SA algorithm to find the hyper-parameters. The sums are still replaced with the averages in (8).

5.3.2 Practices Followed

We follow the same practices as discussed in Section 5.1.2.

5.3.3 Trials

We conduct four types of trials, summarized in Table 6: ML5, ML10, ATSD15+ML5, and ATSD50+ML5. ML5 can be viewed as the control. We chose the trials ML5 and ML10 to demonstrate the effects of increasing $\left|\hat{\mathcal{F}}_+\right|$. The ATSD+ML trials, ATSD15+ML5 and ATSD50+ML5, demonstrate the effects of increasing the number of random TSP problems.

200 subtrials are conducted per trial, all using the same $\hat{\mathcal{F}}_+$, $\hat{\mathcal{F}}_-$, or both. The subtrials only differ in the random seed used in the meta-optimization process (*i.e.,* the simulated annealing random seed). This is designed to accurately estimate the variability of the quality of the algorithms returned from the meta-optimization process. Because we are using a total of 800 subtrials, we limit the meta-optimization search to 2,000 simulated annealing iterations for time considerations.

We test each resulting algorithm (a total of 800 algorithms) against 400 samples of never seen before data from the $\mathcal{F}_+$, $\mathcal{F}_0$, and $\mathcal{F}_-$ distributions.

5.3.4 Computer setup

The computer setup is very similar to that described in Section 5.1.4. We only use the first two computers as this experiment took only about one week to complete. We use the same Internet based virtual drive for all I/O. The meta-optimization process generates about 1020 MB of data including checkpointing every 100 iterations. The code is written to exploit coarse-grain parallelism, scaling up to about 800 instances of Matlab. Our parallelism techniques avoid the need for Matlab's parallel computing toolbox.

5.4 Classification Problem

This experiment tests ATSD+ML against a supervised machine learning problem. Specifically, the task is to learn to classify whether an individual makes more than $50K per year based on 14 attributes such as the sector in which the person works (private, local government, state government, etc.), job type, level of education, sex, age, hours per week, native-country, etc. This data, referred to as the "adult" data, is made publicly available and can be found in the UCI Machine Learning Repository (Bache and Lichman, 2013).

This experiment's goal is to demonstrate how ATSD can be used to boost machine learning performance, not to beat previous classification performance. We compare the classification accuracy of support vector machines (SVMs). The kernel and its parameters are meta-optimized using differing numbers of cross-validation samples and sacrificial problems. Cross-validations are used for $\hat{\mathcal{F}}_+$, which is arguably insufficient (Wolpert, 2001). This is not so much a drawback for the experiment as it demonstrates how quality $\hat{\mathcal{F}}_+$ may be difficult to obtain, further motivating the use of ATSD.

We use three types of sacrificial data. One third of $\hat{\mathcal{F}}_-$ uses real tuples of the attributes, but with random classification results. Another third of $\hat{\mathcal{F}}_-$ uses both random tuples of attributes and classification results. The last third of $\hat{\mathcal{F}}_-$ uses real tuples of attributes, but with a very simple classification problem where the result is wholly determined by a logical conjunction based on the sex, age, and number of years of education. This third set of sacrificial data exhibits several properties absent in the actual data: it is noise free, it depends only on three parameters (ignoring important information such as the job type) and has very low Kolmogorov complexity.

5.4.1 Meta-Optimization Procedure

The meta-optimization objective is to find a good set of SVM hyper-parameters. Specifically, the SVM's kernel, box constraint (training error versus complexity), and kernel parameters (*e.g.* Gaussian kernel width). We investigate seven SVM kernels including:

- Linear: $u^\intercal v$
- Gaussian: $\exp(-\left|u-v\right|^2/(2\sigma^2))$
- Laplace: $\exp(-\left|u-v\right|/\sigma)$
- Cauchy: $(1+\left|u-v\right|^2/\sigma^2)^{-1}$
- Tanh (Multilayer Perception): $\tanh(\alpha u^\intercal v + c)$
- Inhomogeneous polynomial: $(\alpha u^\intercal v + c)^p$
- Logarithmic kernel: $-\log(c+\left|x-y\right|^p)$.[9]

σ, α, c, and p in this context are kernel parameters. u and v are data. Note that some of these kernels are only conditionally positive definite. Meaning, if used, the SVM may fail to converge to globally optimal results.

9. Less common kernels were adopted from Cesar Souza's blog: `http://crsouza.blogspot.com/2010/03/kernel-functions-for-machine-learning.html`

The meta-optimization procedure is conducted differently in this experiment than in the other experiments. Unlike the other experiments' meta-optimization procedure, we first sample the algorithm space blindly, then optimize after sampling. This ensures each trial gets the same quality search since they all evaluate the same hyper-parameters. For each kernel, we sample 500 hyper-parameter configurations using low discrepancy Halton sequences (Kuipers and Niederreiter, 1974; Kalos and Whitlock, 2008). This samples the hyper parameters in a roughly uniform manner. All $\hat{\mathcal{F}}_+$ and $\hat{\mathcal{F}}_-$ are evaluated and their results are stored. To evaluate each ML or ATSD trial, we calculate the meta-objective function from the stored results. The hyper-parameters that optimize the meta-objective function for each trial are selected for testing. This makes the search for the optimal algorithm less dependent on a random search (the other experiments used SA). All trials get the same quality search. This also removes the need to test hundreds of different seeds in the meta-optimization process.

When meta-optimizing to select the kernel and kernel parameters that work best for a particular trial, we use a slightly modified (8). Instead of using (8) verbatim, we replace the sums with averages. This allows us to vary $\left|\hat{\mathcal{F}}_+\right|$ and $\left|\hat{\mathcal{F}}_-\right|$ while continuing to give the same relative importance β.

5.4.2 Practices Followed

We mostly follow the same practices as discussed in Section 5.1.2, with one exception. Because we are working with real data in this experiment, we only have a single $f \in \mathcal{F}_+$. So in order to make $\left|\hat{\mathcal{F}}_+\right|$ larger than one, we partition the data into training, validation, and test partitions. We partitioned the data as follows. For the test partition, we used the official test data, `adult_test.csv`, from the UCI Machine Learning Repository (Bache and Lichman, 2013). After removing incomplete data and NaNs, the `adult_test.csv` contains 30162 records. We partition the `adult_data.csv` data into 1/4 for training and 3/4 for validation for each $f \in \mathcal{F}_+$. By making the training data smaller than the validation data, we decreas the time the meta-optimization process uses.

We take the following additional steps to decrease SVM training time. We allow the SVM decision boundary to violate up to 15% of the Karush-Kuhn-Tucker (KKT) conditions (Bazaraa et al., 2006). This means that the result of training the SVM may not be a globally optimal solution, but should in no way invalidate the effects of ATSD. Moreover, we increase the KKT tolerance from 0.001 to 0.05. This means that the KKT conditions close to being satisfied will count as satisfied. This should not invalidate the effects of ATSD, either.

It is trivial to show that relaxing the KKT constraints translates to testing different learning algorithms, other than SVMs. In the NFL framework for supervised learning, each learner is trying to guess the function generating the data (Wolpert, 2001). SVMs trained with relaxed constraints may produce different predictions (*i.e.,* guesses at the function generating the data). This translates as using different learning algorithms than the canonical SVM algorithm. Using a different learning algorithm other than SVMs would have the same consequence. Hence, relaxing the constraints does not invalidate the experiment.

5.4.3 Trials

Because we are meta-optimizing differently, reusing the same function evaluations for each trial, we can test more trials quickly. We vary $\left|\hat{\mathcal{F}}_+\right|$ and $\left|\hat{\mathcal{F}}_-\right|$ with $\beta = 0.1774$ to get 20 trials. We also vary β over zero and 13 logarithmically spaced levels with $\left|\hat{\mathcal{F}}_+\right| = 2$ and $\left|\hat{\mathcal{F}}_-\right| = 4$. Similarly, we vary β over zero and 13 logarithmically spaced levels with $\left|\hat{\mathcal{F}}_+\right| = 4$ and $\left|\hat{\mathcal{F}}_-\right| = 4$. Thus, in total there are 46 unique trials, each denoted using the notation ATSD+X-Y@B. Here $X = \left|\hat{\mathcal{F}}_+\right|$, $Y = \left|\hat{\mathcal{F}}_-\right|$, and $B = \beta$. Trial ATSD+2-12@0.1774 corresponds to the trial with two cross-validations, 12 sacrificial problems, and using $\beta = 0.1774$. When $\beta = 0$, the sacrificial data is given no weight and ATSD+ML degenerates to ML.

5.4.4 Computer setup

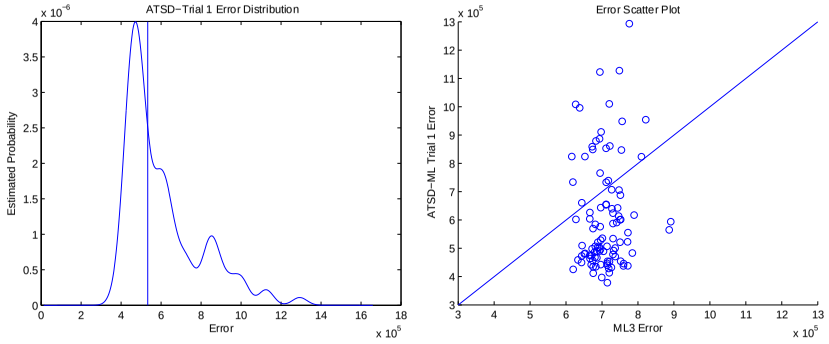
The computer setup is very similar to that described in Section 5.1.4. We only use the first two computers as this experiment takes only about two weeks to complete. We use the same Internet based virtual drive for all I/O. The meta-optimization process generates about 11.9 GB of data, saving all function evaluations (required for optimization after sampling). The code is written to exploit coarse-grain parallelism, scaling up to about 3500 instances of Matlab. Our parallelism techniques avoid the need for Matlab's parallel computing toolbox.

6. Results

We present the results of the four experiments here. In summary: the satellite trajectory experiment showed ATSD helped, but to lesser extent than ML, as predicted in (12). ATSD also had two of four subtrials drop degrees of freedom in their search. The high dimensional traveling salesperson experiment showed that neither ATSD nor ML have any benefit when no problem knowledge is exploitable. The traveling salesperson on a ring experiment is inconclusive due to an NFL artifact. While the classification problem lacks statistical backing—due to there being a single test problem—some general trends suggest, but are inconclusive, that in the case of a limited search for a better learner, it may be best to limit the sum $\left|\hat{\mathcal{F}}_+\right| + \left|\hat{\mathcal{F}}_-\right|$. Increasing either $\left|\hat{\mathcal{F}}_+\right|$ or $\left|\hat{\mathcal{F}}_-\right|$ after a certain degree produced worse results. A single cross-validation combined with nine sacrificial problems ($\left|\hat{\mathcal{F}}_+\right| = 1, \left|\hat{\mathcal{F}}_-\right| = 9$) outperformed any ML alone.

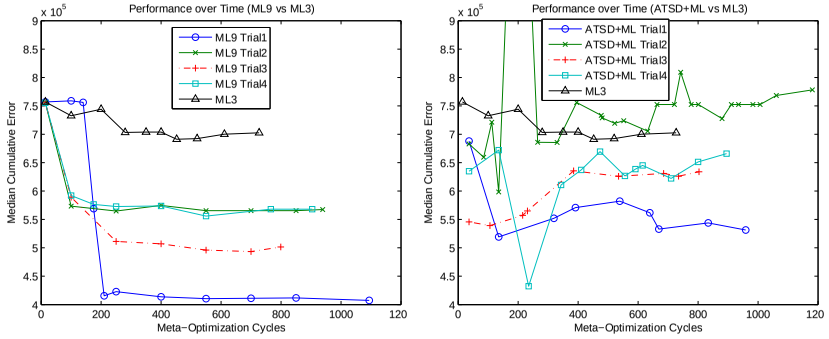
6.1 Satellite Trajectory Results

We analyze the median performance of our algorithms for two primary reasons. First, we optimized over the median performance. Second, after we collected the data from our experiments, it became evident that the performance distributions of the algorithms are highly skewed (see Figure 2a). Thereupon, we follow the advice of Luke (2013) and Wineberg (2004) and analyze our results using the median rather than the mean.



(a) The estimated error distribution (b) Error from ML3 and ATSD+ML3

Figure 2: Depictions of the cumulative error distribution



(a) ML9 and ML3 performance over time (b) ATSD+ML3 and ML3 performance over time

Figure 3: Depictions of the cumulative error over time

While we do not analyze the performance over time, we still present the data in Figure 3a and Figure 3b. Notice that while all the trials in Figure 3a start at the same value, the same cannot be said about the trials in Figure 3b (this is due to an unexpected system crash). Thus, the optimization number of function evaluations are incorrect on the ATSD+ML3 trials. This did not influence the final analysis, just the temporal performance seen in Figure 3b. Meta-optimization stopped when Matlab determined insufficient progress was being made.

Table 7 summarizes the median cumulative error of each technique, meaning lower is better. Since there were four trials for each ATSD+ML3 and ML9, to make their analysis easier, we introduced the terms "ATSD+ML3-1," "ATSD+ML3-2," etc. to indicate

Table 7: Summary of Median Performance between Methods

	Median Cumulative Error
Initial	$7.461 \cdot 10^5$
ML3	$7.026 \cdot 10^5$
ATSD+ML3	$6.637 \cdot 10^5$
ML9	$5.249 \cdot 10^5$

particular trials. "ATSD+ML3" and "ML9" are 100 median samples, where the median was taken over the corresponding four trials. Thus, the table reflects the median of medians for ATSD+ML3 and ML9. This table shows that the control performed worst, ML3 was third, ATSD+ML3 was second, and ML9 performed best. We discuss these results after making sure they are statistically significant.

We must analyze the data further to determine if the differences in medians are statistically significant. Due to the great asymmetry, possible dependencies in the data, and multimodal nature of the ATSD+ML3 performance distributions, we avoid using the standard T-tests to compare performance between algorithms. While the Wilcoxon signed rank test is more robust, it is usually employed in the context of symmetric distributions, especially for paired (one-sample) tests (Lehmann, 2004). For this reason, we use the paired sign-test. It usually lacks the power of the other tests but makes fewer assumptions about the data (Lehmann, 2004). Due to the lack of power of the paired sign test, we may permit a lower significance (Lehmann and Romano, 2008).

Table 8 shows the results of the paired sign-tests comparing ATSD+ML3 with ML9. We present the results from the algorithms at the final iteration to reflect the possible effects of overtraining. This table confirms that ML9 outperforms ATSD+ML3: this is to be expected according to theory derived in Section 3.2. According to (12), even with equal compressibility of the meta data and sacrificial data ($\rho_- = \rho_+$), the samples for $\hat{\mathcal{F}}_-$ are expected to be half as efficient.[10] Since ATSD+ML3 uses three samples for $\hat{\mathcal{F}}_+$ and nine samples for $\hat{\mathcal{F}}_-$, according to theory this should perform on par with ML using 7.5 ($3 + 9/2$) samples for $\hat{\mathcal{F}}_+$.

Table 9 shows the results of the paired sign-tests comparing ATSD+ML3 with ML3. This shows that ATSD appears to improve performance, at least on the median. The paired-sign test shows $p = 4.431 \cdot 10^{-2}$, indicating that the null hypothesis (ATSD+ML3 = ML3) is unlikely. Again, the paired-sign test lacks statistical power in general, so $p = 4.431 \cdot 10^{-2}$ is statistically significant.

We also calculated a Bayes factor to directly compare the two hypotheses: median(ML3 − ATSD+ML3) > 0 and median(ML3 − ATSD+ML3) < 0. The difference between ML3 and ATSD+ML3 is distributed approximately as an extreme-value distribution (*cf.* Figure 4).[11] Matlab's extreme-value distribution has its median at $\mu + \sigma \log(\log(2))$. Thus, for our first hypothesis, we integrated over the range of parameters where $\mu > -\log(\log(2))\sigma$ and for the second hypothesis we used $\mu < -\log(\log(2))\sigma$. Since Bayes factors use a prior

10. Our $\hat{\mathcal{F}}_-$ comes from a more complicated distribution, so we expect a lower ρ_- than ρ_+. This is supported by the fact that ATSD+ML3 performs more similarly to ML3 than ML9.
11. We have no reason to believe the data should be distributed as such, but extreme-value distributions provided the best fit.

Table 8: Sign Test (H_a: ATSD+ML3 > ML9)

	ML9-1	ML9-2	ML9-3	ML9-4	ML9
ATSD+ML3-1	$1.30 \cdot 10^{-12}$	0.972	$6.02 \cdot 10^{-3}$	0.903	0.382
ATSD+ML3-2	$7.97 \cdot 10^{-29}$	$6.55 \cdot 10^{-12}$	$1.60 \cdot 10^{-19}$	$1.35 \cdot 10^{-10}$	$1.53 \cdot 10^{-17}$
ATSD+ML3-3	$1.32 \cdot 10^{-25}$	$9.05 \cdot 10^{-8}$	$6.26 \cdot 10^{-23}$	$9.05 \cdot 10^{-8}$	$1.00 \cdot 10^{-21}$
ATSD+ML3-4	$1.27 \cdot 10^{-16}$	$2.04 \cdot 10^{-4}$	$5.58 \cdot 10^{-10}$	$2.04 \cdot 10^{-4}$	$2.76 \cdot 10^{-8}$
ATSD+ML3	$1.32 \cdot 10^{-25}$	$1.35 \cdot 10^{-10}$	$1.32 \cdot 10^{-25}$	$2.41 \cdot 10^{-13}$	$6.26 \cdot 10^{-23}$

Table 9: Sign Test (H_a: ATSD+ML3 < ML3)

	ML3
ATSD+ML3-1	$9.050 \cdot 10^{-8}$
ATSD+ML3-2	0.9334
ATSD+ML3-3	$1.759 \cdot 10^{-3}$
ATSD+ML3-4	0.2421
ATSD+ML3	$4.431 \cdot 10^{-2}$

distribution, we chose μ to be distributed normally about σ with a standard deviation corresponding to the range of Matlab's `evfit`'s 95% confidence interval. σ is distributed half-normal with a standard deviation corresponding to the range of its 95% confidence interval. The resulting Bayes factor, $k = 5.571$, means that it is more than five times as likely that ATSD+ML3 has a lower (better) median than ML3. According to Robert et al. (2009), this is *substantial* evidence.

There were two surprising, unanticipated results that only occurred for the experiments using anti-training with sacrificial data combined with meta-learning (ATSD+ML3). Two out of the four ATSD+ML3 meta-optimization runs (trials 2 and 3) halted early with the message "distance between lower and upper bounds, in dimension 4 is too small to compute finite-difference approximation of derivative." This corresponds to the two runs limiting their search by dropping a degree of freedom corresponding to the elliptical orientation of the satellite's orbit.[12] This is plausible: the debris in $\mathcal{F}_+$ have no large angular gaps, but the debris in $\mathcal{F}_-$ did have angular gaps. Thus, on average no orientation is to be preferred on $\mathcal{F}_+$, but some orientations are significantly worse for $\mathcal{F}_-$. Recall that in general ATSD+ML maximizes the performance difference between $\mathcal{F}_+$ and $\mathcal{F}_-$. As such, we suspect that ATSD+ML3 extracted this information from this difference. It seems unlikely that this result occurred due to random chance as two trials from ATSD+ML3 dropped this degree of freedom, but none of the five ML did so.

Another surprise is that the second ATSD+ML3 subtrial also dropped a second degree of freedom from its search: the initial true anomaly. Dropping this variable appears to be a valid choice for speeding the search for good solutions, albeit less obvious. Provided that the ratio of orbital periods are irrational (or requiring very large rational numbers) and ignoring highly eccentric orbits, the equidistribution theorem can be applied to show that

12. Meta-optimization was allowed to continue with the variable replaced by the average of the lower and upper bounds.

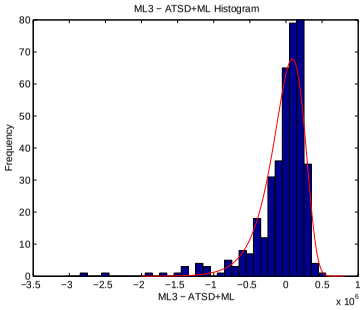


Figure 4: ML3 − ATSD+ML3 and its best fit distribution

the difference in orbital-phases will be uniform (or nearly uniform). Thus, provided enough time, the initial true anomaly may also be ignored. How can we reconcile this with the fact that the second ATSD+ML3 trial performed the worst out of all the ATSD+ML trials? We do not know the answer, but speculate it is because we only tested the performance over $\hat{\mathcal{F}}_+$. It is possible that this reduced degree of freedom would be beneficial in the more general $\mathcal{F}_+ \cup \mathcal{F}_0$. Further testing is required.

6.2 Random TSP Results

As expected, neither ML nor ATSD produce optimization algorithms that reduce error when no problem structure may be exploited. Table 10 shows the mean and standard deviation (taken across the four subtrials) of the mean cumulative error across 10000 tests. Notice that the cumulative error for ML15 over $\mathcal{F}_+$ is worse than ML10's error, although by a statistically insignificant amount. ML10's improvement over ML5's error is also statistically insignificant ($p = 0.4315$). The evidence suggests that increasing the number of meta-learning samples has an insignificant effect on algorithm performance. Similarly, the two ATSD tests (ATSD250+ML5 and ATSD250+ML5 Weak) show no improvement over ML5 alone. They actually show higher mean error over $\mathcal{F}_+$. This could be due to random chance or by the fact that this version of the GA optimizer permits duplicate function evaluations, putting those trials at a disadvantage.[13]

Another issue of concern is whether the meta-optimization procedure succeeded in optimizing (8). Note that we replaced the sums with averages, so β reflects the relative weight of ATSD compared to ML. This is used in Table 11 for the meta-fitness column. It shows how well the discovered algorithms perform according to their meta-optimization criteria. The ATSD250+ML5 trial has the most negative meta-fitness when $\beta = 1.0$, meaning it provides the better solution to (8) than any other algorithms investigated. However, ATSD250+ML5 Weak which was optimized with respect to $\beta = 0.2$, fails to have the lowest meta-fitness for that column. This means that the algorithms discovered

13. The following experiment, the TSP problem on a ring, forces the GA optimizer to use unique function evaluations within each iteration.

Table 10: Mean and Standard Deviation of Means

Trial	$\mathcal{F}_-$		$\mathcal{F}_0$		$\mathcal{F}_+$	
	μ	σ	μ	σ	μ	σ
ML5	47.20	0.4175	47.37	0.3774	47.12	0.3710
ML10	47.12	0.3888	47.28	0.3629	47.07	0.3642
ML15	47.13	0.5617	47.31	0.5370	47.07	0.5196
ATSD250+ML5	49.83	1.981	49.90	1.870	49.56	1.811
ATSD250+ML5 Weak	48.35	1.043	48.47	0.9683	48.18	0.9645

Table 11: Meta Objective Function Values

Trial	mean meta-fitness ($\beta = 1.0$)	mean meta-fitness ($\beta = 0.2$)
ML5	$-7.972 \cdot 10^{-2}$	37.68
ML10	$-4.700 \cdot 10^{-2}$	37.65
ML15	$-5.931 \cdot 10^{-2}$	37.65
ATSD250+ML5	-0.2708	39.59
ATSD250+ML5 Weak	-0.1695	38.51

in ATSD250+ML5 Weak are, even according to theory, inferior to the algorithms from the ML15 trials. This is a side-effect of the NFL theorems: it may be better to search for something else rather than the desired objective function.

6.3 TSP Ring Results

Because this experiment uses a large number of subtrials, we can illustrate the differential error distributions. Figure 5 shows how the ATSD15+ML5 and ATSD50+ML5 trials compare to the ML5 and ML10 trials. Since these distributions are symmetric with few outliers, it makes sense to use the standard T-test to compare means. Furthermore, since the individual test problems are the same for each algorithm, we are permitted to use the paired T-test. Negative values favor ATSD and positive values favor ML.

Contrary to what we anticipated, all the distributions in Figure 5 have a positive average, meaning the algorithms found from the ML trials are to be preferred. The mean error and its standard deviation for each trial over each problem distribution $\mathcal{F}_-, \mathcal{F}_0, \mathcal{F}_+$ are shown in Table 12. Table 13 shows the results of the T-tests, where the alternative hypotheses are that ATSD15+ML5 and ATSD50+ML5 have greater means than ML5 and ML10.[14] Since all the p values are sufficiently small, the null hypotheses should be rejected in favor of the alternatives. The algorithms found in the ATSD15+ML5 and ATSD50+ML5 trials perform worse than the algorithms found in the ML5 and ML10 trials.

Even though the algorithms found in the ATSD trials perform statistically significantly worse than the algorithms found in the ML trials, these results do not contradict our theory. To see why these results do not contradict our theory, we need to investigate the meta-objective function values. These values are shown in Table 14. The meta-optimization

14. The significances in Table 13 were so small, arbitrary precision arithmetic was required to compute them.

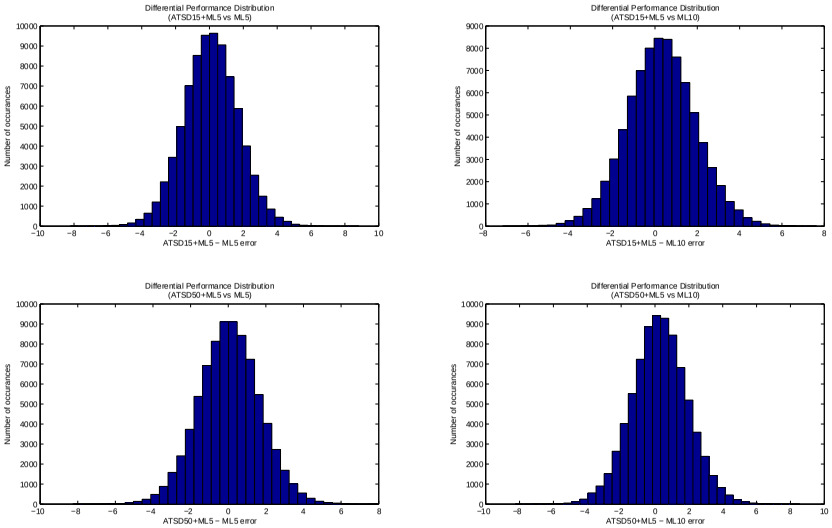


Figure 5: Error differences as a distribution

Table 12: Mean and Standard Deviation of Means

Trial	$\mathcal{F}_-$		$\mathcal{F}_0$		$\mathcal{F}_+$	
	μ	σ	μ	σ	μ	σ
ML5	17.29	1.210	15.57	1.309	16.96	1.438
ML10	17.19	1.216	15.39	1.309	16.74	1.448
ATSD15+ML5	17.37	1.212	15.65	1.304	17.05	1.427
ATSD50+ML5	17.35	1.221	15.62	1.323	17.02	1.453

Table 13: Significance of Differential Performance

Distribution	Significance
ATSD15+ML5 - ML5	$3.783 \cdot 10^{-63}$
ATSD15+ML5 - ML10	$1.150 \cdot 10^{-642}$
ATSD50+ML5 - ML5	$9.566 \cdot 10^{-30}$
ATSD50+ML5 - ML10	$7.845 \cdot 10^{-508}$

Table 14: Meta Objective Function Values when $\beta = 0.2$

Trial	mean meta-fitness
ML5	13.50
ML10	13.30
ATSD15+ML5	13.58
ATSD50+ML5	13.55

for ML5 and ML10 found better solutions for the ATSD+ML meta-objective function than the ATSD15+ML5 and ATSD50+ML5 trials! This is a side-effect of the NFL theorems. Without knowing anything about the search for optimization algorithms, we naïvely used a SA search. Using SA to reduce the meta-objective function with $\beta = 0.2$ actually performed worse at reducing it than when trying to reduce meta-objective function with $\beta = 0.0$ (ML alone). To better understand what happened consider the following analogy. When a dog (an algorithm) was told to find (optimize) the yellow ball (the ATSD objective function), it found nothing. However, when told to find the violet ball (the ML objective function), it found both the yellow and violet balls (good solutions to both meta-optimization functions).

So while this experiment does not confirm our theory, neither does the experiment deny it. Instead, it provided meaningful feedback. Matlab's SA algorithm performs better on ML than ATSD+ML for the small set of problems investigated in this experiment. We can prevent this problem from occurring by ensuring all trials evaluate an identical set of algorithms. This effectively amounts to sampling algorithms first, recording their performance, then optimizing after all algorithms have been tested on all problems. Thus, it is impossible for ML to find an algorithm that outperforms ATSD+ML on ATSD+ML's own objective function. If ML were to find such an algorithm, because ATSD+ML would test it too, ATSD+ML must at least match it (assuming the function values are deterministic). The following experiment uses this adjustment.

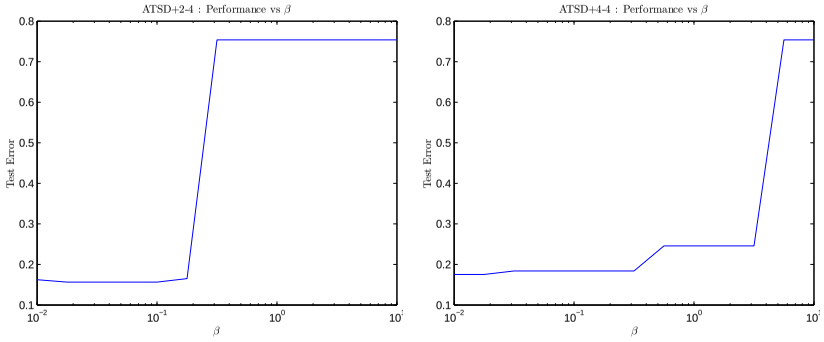
6.4 Classification Results

The results from this experiment lack statistical backing since we use only a single test problem. We considered using bootstrapping to estimate the variance of the error rates presented in this experiment but ultimately avoided it due to time considerations—the experiment already took two weeks to complete. A ten-fold cross-validation would take much longer than an additional two weeks.[15] Future work will include bootstrapping to estimate the variance in this experiment.

Despite this, some general trends are still evident. We modified (8) by replacing the sums with averages so β reflects the relative weight of ATSD. Figures 6a and 6b show that large values of β cause more harm than good. Figure 6a shows $2 \cdot 10^{-2} < \beta < 10^{-1}$ as being ideal, whereas Figure 6b shows that ATSD should be avoided. Since this difference in behavior could be noise, we further investigated the error rate's behavior.

Table 15 shows several patterns. First, in terms of error, ATSD+1-9@0.1778 and ATSD+1-12@0.1778 perform best. These two trials even outperform the best performing ML (ATSD+3-0@0). Another point is that the best performing trials have a negatively

15. Although, it should take less than twenty weeks to complete a ten-fold cross validation study.



(a) Error rate when $\left|\hat{\mathcal{F}}_+\right| = 2$ and $\left|\hat{\mathcal{F}}_-\right| = 12$ (b) Error rate when $\left|\hat{\mathcal{F}}_+\right| = 4$ and $\left|\hat{\mathcal{F}}_-\right| = 12$

Figure 6: Error rate versus β

Table 15: Error versus $\left|\hat{\mathcal{F}}_+\right|$ and $\left|\hat{\mathcal{F}}_-\right|$ with β=0.1778

	$\left\|\hat{\mathcal{F}}_-\right\|$=0	$\left\|\hat{\mathcal{F}}_-\right\| = 3$	$\left\|\hat{\mathcal{F}}_-\right\| = 6$	$\left\|\hat{\mathcal{F}}_-\right\| = 9$	$\left\|\hat{\mathcal{F}}_-\right\| = 12$
$\left\|\hat{\mathcal{F}}_+\right\|$=1	0.1624	0.1624	0.1624	**0.1558**	**0.1558**
$\left\|\hat{\mathcal{F}}_+\right\|$=2	0.1624	0.1629	0.1624	**0.1564**	0.1652
$\left\|\hat{\mathcal{F}}_+\right\|$=3	**0.1564**	0.1579	0.1579	**0.1564**	0.1839
$\left\|\hat{\mathcal{F}}_+\right\|$=4	**0.1752**	**0.1752**	0.1839	0.1839	0.1839

correlated $\left|\hat{\mathcal{F}}_-\right|$ and $\left|\hat{\mathcal{F}}_+\right|$. The bold diagonal highlights this. This pattern lacks statistical significance, thus one should not read into it too much. Figure 7 also visualizes these results.

While we are not trying to beat historical classification performance on this data set, it is worthwhile to note preexisting performance on this data set. Kohavi (1996) used naïve Bayes classification trees to produce predictions with errors ranging from about 15.7% to 16.7%, and the C4.5 algorithm produced predictions with errors ranging from about 13.7% to 14.8%.

7. Discussion and Conclusion

We used the statistical entropy of the objective function distributions to address when meta-learning (ML) is possible. For ML to benefit learning, the typical problems should have low Kolmogorov complexity. This implies that it is necessary for the problem distribution to be at least compressible, but not necessarily fit in polynomial space.

We introduced a novel theoretical approach, anti-training with sacrificial data, to improve machine-learning performance. ATSD directly follows from and relies on the No Free Lunch (NFL) theorems. It can be seen as an approximate dual of ML. As

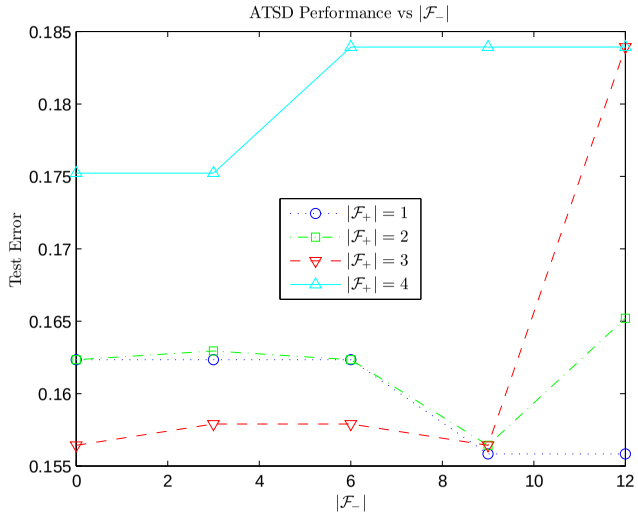


Figure 7: Error rate versus $\left|\hat{\mathcal{F}}_-\right|$

such, the theory dictates that ATSD works where ML works, namely when the probable problems are compressible. While ML directly improves performance over a set of common problems, ATSD worsens performance over sacrificial (impossible and unlikely) problems. Due to the NFL theorems, performance must increase over the remaining sum of all other problem, although it is possible for performance to transfer from the typical problems to any unspecified problems. To help counter-balance this possibility, ATSD ought to be combined with ML (ATSD+ML).

We examined the theoretical efficiency of ATSD using a deck of cards analogy. We arrived at a function relating the percent of implausible functions, the compressibility of typical functions, and the compressibility of implausible functions with the efficiency of ATSD. Assuming that about half of all functions are exceedingly unlikely and the ML objective functions are as compressible as the sacrificial objective functions, then the relative efficiency of ATSD is half as efficient as ML. Moreover, we addressed how to generate sacrificial data and qualitatively assess their compressibility.

ATSD+ML has other helpful properties. Since sacrificial data is easy to generate, it can be applied even when real data is scarce. The theory suggests this is when ATSD+ML is expected to be most effective. The fourth experiment provides supportive, but inconclusive evidence of this. According to the theory, it also helps prevent overtraining. By providing an empirical definition of noise, theoretically the learning algorithm can fail to learn noise. Also when combined with ML, ATSD+ML makes it less likely to overfit the original ML objective function. ATSD+ML also allows for the specification of failure cases, reducing the number of unexpected failures.

ATSD+ML is a technique still in its infancy. Much work remains to be done on accurately estimating how much and what kind of sacrificial data ATSD+ML needs. We only touched on the proper balance between ATSD and ML. In general, a safe value for β in (8) is $0 \leq \beta < \left|\hat{\mathcal{F}}_+\right| / \left|\hat{\mathcal{F}}_-\right|$, or when the sums are replaced with averages $0 \leq \beta < 1$. Another future research direction includes studying the implications of the NFL theorems approximately holding, when the equality in (1) is replaced with upper and lower bounds. We conjecture that the benefits from anti-training would diminish as the equality becomes less strict. However, this would need to be investigated more thoroughly. ATSD+ML could be viewed as a form of regularization and studied as such. It might also be able to be applied to model selection rather than algorithm selection. A probabilistic model would be useful in answering these questions.

The basic concept of ATSD—improving performance by training to perform worse on irrelevant data—may have benefits for selecting solutions (not algorithms) to improve predictive accuracy. This baseline ATSD does not follow from the NFL theorems in the same manner. It may still follow from the NFL theorems, but for a different reason. However, baseline ATSD may be justifiable in terms of regularization. If a solution fits random noise in addition to the actual data, then the solution may be unnecessarily complex.

We conducted four experiments showing how ATSD may function in practice. The first experiment produced results showing ATSD improves performance for optimizers. The second experiment confirmed that ATSD should fail when ML fails. The third experiment was inconclusive as the algorithms discovered from ML outperformed the ATSD candidates even at their own meta-objective function; a side effect of the NFL algorithms. The fourth experiment provides supportive, but inconclusive evidence that ATSD benefits classification algorithms. More experiments are left for future work. Such experiments include repeating similar experiments that take significantly less time to complete, using 10-fold cross-validation for machine learning experiments, and comparing direct meta-optimization (as in the first three experiments) to sampling based approached (as in the fourth experiment).

Convex optimization is a desirable property needed for more sophisticated machine learning procedures. This is the issue when extending ATSD to work with support-vector machines, support-vector regression, or any other optimization that depends on convex optimization problems. The results from the third experiment motivate a convex meta-optimization framework to ensure algorithms actually optimize their meta-objective function. The issue is that it is possible for the global minimum to occur whenever the sacrificial performance approaches positive infinity; this drives the objective function to negative infinity. This in turn causes the traditional objective function to be largely ignored. The most natural approach would be to apply some convex non-decreasing saturating function (*e.g.,* $exp(-x)$) to the sacrificial term. The effects of this saturating ATSD on convex optimization techniques should be studied, as well as other techniques to extend ATSD to a convex optimization framework.

References

Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010. doi: 10.1214/09-SS054. URL `http://arxiv.org/abs/0907.4728`.

Peter Auer, Mark Herbster, and Manfred K. Warmuth. Exponentially many local minima for single neurons, 1995.

Anne Auger and Olivier Teytaud. Continuous lunches are free! In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, GECCO '07, pages 916–922, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-697-4. doi: 10.1145/1276958.1277145. URL http://doi.acm.org/10.1145/1276958.1277145.

Anne Auger and Olivier Teytaud. Continuous lunches are free plus the design of optimal optimization algorithms. *Algorithmica*, 57:121–146, 2010. ISSN 0178-4617. doi: 10.1007/s00453-008-9244-5. URL http://dx.doi.org/10.1007/s00453-008-9244-5.

K. Bache and M. Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*, chapter 4, pages 188–195. Wiley, 3rd edition, 2006.

J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13:281–305, 2012.

W. Bialek, I. Nemenman, and N. Tishby. Predictability, complexity, and learning. *Neural Computation*, 13(11):2409–2463, 2001.

Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. Springer Verlag, 2002. ISBN: 9780387953519.

D. Corne and J. Knowles. No free lunch and free leftovers theorems for multiobjective optimisation problems. In *Evolutionary Multi-Criterion Optimization*, pages 66–66. Springer, 2003a. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.15.1480.

D. Corne and J. Knowles. Some multiobjective optimizers are better than others. In *Proceedings Congress Evolutionary Computation CEC '03*, volume 4, pages 2506–2512, 2003b. doi: 10.1109/CEC.2003.1299403.

T.M. Cover and J.A. Thomas. *Elements of Information Theory*, chapter Kolmogorov Complexity, pages 463–508. John Wiley & Sons, Inc., Hoboken, New Jersey, second edition, 2006. ISBN 978-0471241959.

Joseph C. Culberson. On the futility of blind search: An algorithmic view of 'no free lunch'. *Evolutionary Computation*, 6:109–127, June 1998. ISSN 1063-6560. doi: http://dx.doi.org/10.1162/evco.1998.6.2.109. URL http://dx.doi.org/10.1162/evco.1998.6.2.109.

Stefan Droste, Thomas Jansen, and Ingo Wegener. Optimization with randomized search heuristics – the (a)nfl theorem, realistic scenarios, and difficult functions. *Theoretical Computer Science*, 287(1):131–144, 2002. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.5850.

N. Gershenfeld. *The nature of mathematical modeling*, chapter Linear and Nonlinear Time Series, pages 204–224. Cambridge University Press, 1999.

C. Giraud-Carrier and F. Provost. Toward a justification of meta-learning: Is the no free lunch theorem a show-stopper? In *Proceedings of the ICML-2005 Workshop on Meta-learning*, pages 12–19, Bonn, Germany, 2005. URL http://dml.cs.byu.edu/~cgc/pubs/ICML2005WS.pdf.

David S. Goodsell and Arthur J. Olson. Automated docking of substrates to proteins by simulated annealing. *Proteins: Structure, Function, and Bioinformatics*, 8(3):195–202, 1990. ISSN 1097-0134. doi: 10.1002/prot.340080302. URL http://dx.doi.org/10.1002/prot.340080302.

Christian Igel and Marc Toussaint. On classes of functions for which no free lunch results hold. *Information Processing Letters*, 86(6):317–321, June 2003. ISSN 0020-0190. doi: 10.1016/S0020-0190(03)00222-9. URL http://dx.doi.org/10.1016/S0020-0190(03)00222-9.

Christian Igel and Marc Toussaint. A no-free-lunch theorem for non-uniform distributions of target functions. *Journal of Mathematical Modelling and Algorithms*, 3:2004, 2004. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.71.8446.

L. Ingber. Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling*, 18(11):29 – 57, 1993. ISSN 0895-7177. doi: http://dx.doi.org/10.1016/0895-7177(93)90204-C. URL http://www.sciencedirect.com/science/article/pii/089571779390204C.

Lester Ingber. Adaptive simulated annealing (asa): Lessons learned. *Control and Cybernetics*, 25:33–54, 1996.

Malvin H Kalos and Paula A Whitlock. *Monte carlo methods*, chapter Quasi-Monte Carlo, pages 101–103. John Wiley & Sons, second edition, 2008.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8. URL http://dl.acm.org/citation.cfm?id=1643031.1643047.

Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 202–207, Menlo Park, California, 1996. AAAI Press.

Lauwerens Kuipers and Harald Niederreiter. *Uniform distribution of sequences*, chapter Special Sequences, pages 129–130. Courier Dover Publications, 1st edition, 1974. URL http://web.maths.unsw.edu.au/~josefdick/preprints/KuipersNied_book.pdf.

E.L. Lehmann. *Elements of Large-Sample Theory*, chapter 3.4 Comparison of tests: Relative efficiency, pages 173–187. Springer, 2004.

E.L. Lehmann and Joseph P. Romano. *Testing statistical hypotheses*, chapter 3.1 Stating The Problem, pages 56–59. Springer, Spring Street, New York, NY 10013, USA, 2008.

Ming Li and Paul Vitányi. *An introduction to Kolmogorov complexity and its applications*, chapter Algorithmic Complexity, pages 101 – 107. Springer, Spring Street, New York, NY, 3rd edition, 2008. doi: 10.1007/978-0-387-49820-1.

Sean Luke. *Essentials of Metaheuristics*. Lulu, second edition, 2013. URL `http://cs.gmu.edu/~sean/book/metaheuristics/Essentials.pdf`. Available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/.

M.D. McKay, R.J. Beckman, and W.J. Conover. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

John R. Rice. The algorithm selection problem. *Advances in Computers*, 15:65–118, 1976. URL `http://www.cs.purdue.edu/research/technical_reports/1975/TR%2075-152.pdf`.

J. Rissanen. Universal coding, information, prediction, and estimation. *Information Theory, IEEE Transactions on*, 30(4):629 – 636, jul 1984. ISSN 0018-9448. doi: 10.1109/TIT.1984.1056936.

J. Rissanen. Stochastic complexity and modeling. *The Annals of Statistics*, pages 1080–1100, 1986.

J. Rissanen. Stochastic complexity. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 223–239, 1987.

J. Rissanen. *Stochastic complexity and statistical inquiry*. World Scientific, Singapore, 1989.

Christian P. Robert, Nicolas Chopin, and Judith Rousseau. Harold jeffreyss theory of probability revisited. *Statistical Science*, 24(2):141–172, 2009. doi: 10.1214/09-STS284.

C. Schumacher, M.D. Vose, and L.D. Whitley. The no free lunch and problem description length. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 565–570, 2001.

Robert H Shumway and David S Stoffer. *Time series analysis and its applications: with R examples*. Springer, New York, Dordrecht, Heidelberg, London, 3rd edition, 2011.

Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18:77–95, 2002.

Guan Ming Wang, Estela Blaisten-Barojas, A. E. Roitberg, and T. P. Martin. Strontium clusters: Many-body potential, energetics, and structural transitions. *The Journal of Chemical Physics*, 115(8):3640–3646, 2001. doi: 10.1063/1.1384454. URL `http://link.aip.org/link/?JCP/115/3640/1`.

Darrell Whitley and Jean Watson. Complexity theory and the no free lunch theorem. In Edmund K. Burke and Graham Kendall, editors, *Search Methodologies*, pages 317–339. Springer US, 2005. ISBN 978-0-387-28356-2. doi: 10.1007/0-387-28356-0_11. URL `http://dx.doi.org/10.1007/0-387-28356-0_11`.

Mark Wineberg. Introductory statistics for evolutionary computation. In *The Sixth Genetic and Evolutionary Computation Conference (GECCO 2004)*, 2004.

David H. Wolpert. The supervised learning no-free-lunch theorems. In *Proceedings 6th Online World Conference on Soft Computing in Industrial Applications*, pages 25–42, 2001. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.99.133`.

D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67 –82, apr 1997. ISSN 1089-778X. doi: 10.1109/4235.585893.

D.H. Wolpert and W.G. Macready. Coevolutionary free lunches. *IEEE Transactions on Evolutionary Computation*, 9(6):721–735, dec. 2005. ISSN 1089-778X. doi: 10.1109/TEVC.2005.856205.

Xiaoguang Zhang, Li Yu, Yuan Zheng, Yu Shen, Guangtao Zhou, Lin Chen, Lixia Xi, Tiecheng Yuan, Jianzhong Zhang, and Bojun Yang. Two-stage adaptive pmd compensation in a 10 gbit/s optical communication system using particle swarm optimization algorithm. *Optics Communications*, 231(1–6):233 – 242, 2004. ISSN 0030-4018. doi: 10.1016/j.optcom.2003.12.045. URL `http://www.sciencedirect.com/science/article/pii/S003040180302385X`.

Appendix A. Proof of the two-valued nature of $\hat{P}_+(f)$ in practice

The estimate of $P_+(f)$ is denoted $\hat{P}_+(f)$. We argue that $\hat{P}_+(f)$ is practically going to be a two-valued function: 0 and $1/\left|\hat{\mathcal{F}}_+(f)\right|$. This is demonstrated by showing that when independently sampling from $\mathcal{F}_+$, the probability that any two functions are identical is exceedingly small in practice. If one stops to ponder for a moment about similar problems, one realizes this problem reduces to the well known birthday problem in probability theory.

The birthday problem asks "what is the probability that out of n people (samples), some pair of them will have the same birthday (function)." The answer is 100% if the number of people exceed the number of days in a year, but otherwise it is:

$$1 - \frac{n!\binom{365}{n}}{365^n}.$$

When replacing days with functions and people with samples this becomes

$$1 - \frac{n!\binom{|\mathcal{F}_+|}{n}}{\left|\mathcal{F}_+\right|^n}. \tag{13}$$

While (13) is exact, it must be used iteratively to figure out how much larger $\left|\mathcal{F}_+\right|$ must be than n to ensure a small probability of failure (that two functions will be identical). We provide an approximate n which limits the probability of failure to ϵ. Consider the probability of two samples having different functions,

$$\left(1 - \frac{1}{\left|\mathcal{F}_+\right|}\right).$$

If there are n samples, then there exist $n(n-1)/2$ pairs of samples. Assuming each comparison is independent (this is approximately true if $n \ll \left|\mathcal{F}_+\right|$), then the probability that all samples are unique is

$$\left(1 - \frac{1}{\left|\mathcal{F}_+\right|}\right)^{n(n-1)/2}$$

Using the binomial theorem, this expands to

$$1 - \binom{\frac{n^2-n}{2}}{1}\frac{1}{\left|\mathcal{F}_+\right|} + \binom{\frac{n^2-n}{2}}{2}\frac{1}{\left|\mathcal{F}_+\right|^2} - \binom{\frac{n^2-n}{2}}{3}\frac{1}{\left|\mathcal{F}_+\right|^3} + \binom{\frac{n^2-n}{2}}{4}\frac{1}{\left|\mathcal{F}_+\right|^4} - \ldots. \tag{14}$$

We will bound the lth term of (14) using the well known inequality $\binom{n}{l} \leq \frac{n^l}{l!}$ to get

$$\begin{aligned}
\binom{\frac{n^2-n}{2}}{l}\frac{1}{\left|\mathcal{F}_+\right|^l} &\leq \frac{\left(\frac{n^2-n}{2}\right)^l}{l!}\frac{1}{\left|\mathcal{F}_+\right|^l} \\
&\leq \left(\frac{n^2-n}{2}\right)^l \frac{1}{\left|\mathcal{F}_+\right|^l} \\
&= \left(\frac{n^2-n}{2\left|\mathcal{F}_+\right|}\right)^l.
\end{aligned}$$

Thus, all higher order terms are negligible so long as

$$\frac{n^2-n}{2\left|\mathcal{F}_+\right|} \ll 1$$

The first two terms are kept and set equal to the desired level of probability

$$\begin{aligned}
1 - \binom{\frac{n^2-n}{2}}{1}\frac{1}{\left|\mathcal{F}_+\right|} &\geq 1 - \epsilon \\
\binom{\frac{n^2-n}{2}}{1}\frac{1}{\left|\mathcal{F}_+\right|} &\leq \epsilon \\
\frac{n^2-n}{2\left|\mathcal{F}_+\right|} &\leq \epsilon \\
n^2 - n &\leq 2\left|\mathcal{F}_+\right|\epsilon \\
n &\leq \frac{\sqrt{8\left|\mathcal{F}_+\right|\epsilon + 1} + 1}{2}
\end{aligned} \tag{15}$$

where ϵ is the probability that two functions will be the same. When $\left|\mathcal{F}_+\right| \cdot \epsilon$ is much larger than one, then this is well approximated by:

$$n \leq \sqrt{2\left|\mathcal{F}_+\right|\epsilon}. \tag{16}$$

For example, problems as small as those with a 16-bit input and an 8-bit output have $|\mathcal{F}| \approx 2.6 \cdot 10^{157826}$. Assuming $n = 10^{12}$ and $\left|\mathcal{F}_+\right| = 10^{100}$ (hundreds of thousands of orders of magnitude smaller), (13) states the probability that any two samples will have the same function is approximately $5.0 \cdot 10^{-77}$. Thus, in sampling the problem domain to build $\hat{P}_+(f)$, it is extremely likely that a particular function will be sampled once or not at all. This leads to the two-valued nature of $\hat{P}_+(f)$ in practice. Equation (16) shows how many samples are permitted while maintaining the two-valued nature with approximately $1 - \epsilon$ confidence. Continuing the above example, when setting the probability of failure $\epsilon \approx e^{-100}$, we see n may be as large as $2.7 \cdot 10^{28}$.

■

Journal of Machine Learning Research 17 (2016) 1-72 Submitted 1/14; Revised 3/15; Published 4/16

A Unifying Framework in Vector-valued Reproducing Kernel Hilbert Spaces for Manifold Regularization and Co-Regularized Multi-view Learning

Hà Quang Minh MINH.HAQUANG@IIT.IT
Pattern Analysis and Computer Vision (PAVIS)
Istituto Italiano di Tecnologia (IIT), Via Morego 30, Genova 16163, ITALY

Loris Bazzani LORIS.BAZZANI@DARTMOUTH.EDU
Pattern Analysis and Computer Vision (PAVIS)
Istituto Italiano di Tecnologia (IIT), Via Morego 30, Genova 16163, ITALY
Department of Computer Science, Dartmouth College, Hanover, NH 03755, USA

Vittorio Murino VITTORIO.MURINO@IIT.IT
Pattern Analysis and Computer Vision (PAVIS)
Istituto Italiano di Tecnologia (IIT), Via Morego 30, Genova 16163, ITALY

Editor: John Shawe-Taylor

Abstract

This paper presents a general vector-valued reproducing kernel Hilbert spaces (RKHS) framework for the problem of learning an unknown functional dependency between a structured input space and a structured output space. Our formulation encompasses both Vector-valued Manifold Regularization and Co-regularized Multi-view Learning, providing in particular a unifying framework linking these two important learning approaches. In the case of the least square loss function, we provide a closed form solution, which is obtained by solving a system of linear equations. In the case of Support Vector Machine (SVM) classification, our formulation generalizes in particular both the binary Laplacian SVM to the multi-class, multi-view settings and the multi-class Simplex Cone SVM to the semi-supervised, multi-view settings. The solution is obtained by solving a single quadratic optimization problem, as in standard SVM, via the Sequential Minimal Optimization (SMO) approach. Empirical results obtained on the task of object recognition, using several challenging data sets, demonstrate the competitiveness of our algorithms compared with other state-of-the-art methods.

Keywords: kernel methods, vector-valued RKHS, multi-view learning, multi-modality learning, multi-kernel learning, manifold regularization, multi-class classification

1. Introduction

Reproducing kernel Hilbert spaces (RKHS) and kernel methods have been by now established as among the most powerful paradigms in modern machine learning and statistics (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004). While most of the literature on kernel methods has so far focused on scalar-valued functions, RKHS of vector-valued functions have received increasing research attention in machine learning recently, from both theoretical and practical perspectives (Micchelli and Pontil, 2005; Carmeli et al., 2006; Reisert and Burkhardt, 2007; Caponnetto et al., 2008; Brouard et al., 2011; Dinuzzo et al., 2011; Kadri et al., 2011; Minh and Sindhwani, 2011; Zhang et al., 2012; Sindhwani et al., 2013). In this paper, we present a general learning framework in the setting of vector-valued RKHS that encompasses learning across three different paradigms, namely vector-valued, multi-view, and semi-supervised learning, simultaneously.

The direction of Multi-view Learning we consider in this work is Co-Regularization, see e.g. (Brefeld et al., 2006; Sindhwani and Rosenberg, 2008; Rosenberg et al., 2009; Sun, 2011). In this approach, different hypothesis spaces are used to construct target functions based on different views of the input data, such as different features or modalities, and a data-dependent regularization term is used to enforce consistency of output values from different views of the same input example. The resulting target functions, each corresponding to one view, are then naturally combined together in a principled way to give the final solution.

The direction of Semi-supervised Learning we follow here is Manifold Regularization (Belkin et al., 2006; Brouard et al., 2011; Minh and Sindhwani, 2011), which attempts to learn the geometry of the input space by exploiting the given unlabeled data. The latter two papers are recent generalizations of the original scalar version of manifold regularization of (Belkin et al., 2006) to the vector-valued setting. In (Brouard et al., 2011), a vector-valued version of the graph Laplacian L is used, and in (Minh and Sindhwani, 2011), L is a general symmetric, positive operator, including the graph Laplacian. The vector-valued setting allows one to capture possible dependencies between output variables by the use of, for example, an output graph Laplacian.

The formulation we present in this paper gives a unified learning framework for the case the hypothesis spaces are vector-valued RKHS. Our formulation is general, encompassing many common algorithms as special cases, including both Vector-valued Manifold Regularization and Co-regularized Multi-view Learning. The current work is a significant extension of our conference paper (Minh et al., 2013). In the conference version, we stated the general learning framework and presented the solution for multi-view least square regression and classification. In the present paper, we also provide the solution for multi-view multi-class Support Vector Machine (SVM), which includes multi-view binary SVM as a special case. Furthermore, we present a principled optimization framework for computing the optimal weight vector for combining the different views, which correspond to different kernels defined on the different features in the input data. An important and novel feature of our formulation compared to traditional multiple kernel learning methods is that it does *not* constrain the combining weights to be non-negative, leading a considerably simpler optimization problem, with an almost closed form solution in the least square case.

Our numerical experiments were performed using a special case of our framework, namely Vector-valued Multi-view Learning. For the case of least square loss function, we give a closed form solution which can be implemented efficiently. For the multi-class SVM case, we implemented our formulation, under the simplex coding scheme, using a Sequential Minimal Optimization (SMO) algorithm, which we obtained by generalizing the SMO technique of (Platt, 1999) to our setting.

We tested our algorithms on the problem of multi-class image classification, using three challenging, publicly available data sets, namely Caltech-101 (Fei-Fei et al., 2006), Caltech-UCSD-Birds-200-2011 (Wah et al., 2011), and Oxford Flower 17 (Nilsback and Zisserman,

2006). The results obtained are promising and demonstrate the competitiveness of our learning framework compared with other state-of-the-art methods.

1.1 Related Work

Recent papers in the literature that are closely related to our work include (Rosenberg et al., 2009; Sun, 2011; Luo et al., 2013a,b; Kadri et al., 2013). We analyze and compare each of these methods to our proposed framework in the following.

In the scalar setting, two papers that seek to generalize the manifold regularization framework of (Belkin et al., 2006) to the multi-view setting are (Rosenberg et al., 2009; Sun, 2011). In (Sun, 2011), the author proposed a version of the Multi-view Laplacian SVM, which, however, only deals with *two views* and is not generalizable to any number of views. In (Rosenberg et al., 2009), the authors formulated a version of the semi-supervised Multi-view learning problem for any number of views, but instead of solving it directly like we do, they proposed to compute the Multi-view kernel and reduce the problem to the supervised case. One problem with this approach is that the Multi-view kernel is complicated analytically, which makes it difficult to implement efficiently in practice. It is also unclear how this approach can be generalized to the multi-class setting.

In the vector-valued setting, papers dealing with multi-view learning include (Luo et al., 2013a,b), where each view is used to define a kernel and a graph Laplacian, and the resulting kernels and graph Laplacians are respectively linearly combined to give the final kernel and final graph Laplacian. Thus this approach does not take into account between-view consistency as in our approach. In (Luo et al., 2013a), which generalizes the vector-valued regularization formulation of (Minh and Sindhwani, 2011), the loss function is the least square loss. In (Luo et al., 2013b), the authors employed a multi-class SVM loss function, which is the average of the binary SVM hinge loss across all components of the output vector. To the best of our knowledge, we are not aware of any theoretical result on the statistical consistency of this loss function.

In the direction of multi-class learning, many versions of multi-class SVM have appeared in the literature, e.g. (Lee et al., 2004; Weston and Watkins, 1999; Crammer and Singer, 2001; Mroueh et al., 2012). In this paper, we employ a generalization of the multi-class Simplex Cone SVM (SC-SVM) loss function proposed in (Mroueh et al., 2012), where it was proved to be theoretically consistent.

Another work dealing with multi-view learning in the vector-valued approach is (Kadri et al., 2013), which considers multi-view learning from the multi-task learning perspective, see e.g. (Evgeniou et al., 2005), where different views of the same input example correspond to different tasks which share the same output label. Their formulation does not have an explicit view combination mechanism and is restricted to scalar-valued tasks and the supervised setting. The resulting optimization problem is vector-valued regularized least square regression in (Micchelli and Pontil, 2005), which is a special case of our general learning framework.

Our multi-view learning approach can also be viewed as a form of multiple kernel learning, but it is different from typical multiple kernel learning approaches, see e.g. (Bach et al., 2004; Bucak et al., 2014) in several aspects. First, it is formulated in both supervised and semi-supervised settings. Second, it incorporates between-view interactions. Third,

it makes no mathematical constraints, such as non-negativity, on the combining weights. This last aspect of our framework contrasts sharply with typical multiple kernel learning methods, which need to constrain the combining weights to be non-negative in order to guarantee the positive definiteness of the combined kernel. As a consequence, our optimization procedure for the combining weights is considerably simpler and has an almost closed form solution in the least square case. We give a brief technical description on the connections between our framework and multiple kernel and multi-task learning in the final part of the paper. Empirically, experimental results reported in the current paper show that our framework performs very favorably compared with state of the art multiple kernel learning methods.

We compared the proposed framework from a methodological point of view with approaches that focus on combining different features in the input data. Our work is complementary to other approaches such as (Zeiler and Fergus, 2014; Razavian et al., 2014; He et al., 2015), which are focused on engineering or learning the best features for the task at hand. In fact, an interesting research direction would be the application of our framework on top of those methods, which will be explored in a future work.

1.2 Our Contributions

Our learning framework provides a unified formulation for Manifold Regularization and Co-regularized Multi-view Learning in the vector-valued setting. In particular, it generalizes the Vector-valued Manifold Regularization framework of (Minh and Sindhwani, 2011), which was formulated in the single-view setting, with the least square loss, to the multi-view setting, with both least square and multi-class SVM loss functions. Consequently, it generalizes the Vector-valued Regularized Least Square formulation of (Micchelli and Pontil, 2005), which was formulated in the supervised, single-view settings, with the least square loss, to the semi-supervised, multi-view settings, with both the least square and multi-class SVM loss functions.

For the case of SVM classification, our framework is a generalization of the multi-class SC-SVM of (Mroueh et al., 2012), which is supervised and single-view, to the semi-supervised and multi-view learning settings. The loss function that we employ here is also a generalization of the SC-SVM loss functions proposed in (Mroueh et al., 2012). We also show that our formulation is a generalization of the semi-supervised Laplacian SVM of (Belkin et al., 2006), which is binary and single-view, to the multi-class and multi-view learning settings.

The generality and advantage of our vector-valued RKHS approach is illustrated by the fact that it can simultaneously (i) deal with any number of classes in multi-class classification, (ii) combine any number of views, (iii) combine the views using an arbitrary weight vector, and (iv) compute all the different output functions associated with the individual views, all by solving a single system linear of equations (in the case of least square loss) or a single quadratic optimization problem (in the case of SVM loss). To the best of our knowledge, this work is the first attempt to present a unified general learning framework whose components have been only individually and partially covered in the literature.

Our optimization framework for computing the optimal weight vector for combining the different views is also novel compared to typical multiple kernel learning methods in that

it does *not* constrain the combining weights to be non-negative, leading to a considerably simpler optimization problem, with an almost closed form solution in the least square case.

1.3 Organization

We start by giving a review of vector-valued RKHS in Section 2. In Section 3, we state the general optimization problem for our learning formulation, together with the Representer Theorem, the explicit solution for the vector-valued least square case, and the quadratic optimization problem for the vector-valued SVM case. We describe Vector-valued Multi-view Learning in Section 4 and its implementations in Section 5, both for the least square and SVM loss functions. Section 6 provides the optimization of the operator that combines the different views for the least square case. Empirical experiments are described in detail in Section 7. Connections between our framework and multi-kernel learning and multi-task learning are briefly described in Section 8. **Proofs for all mathematical results in the paper are given in Appendix A.**

2. Vector-Valued RKHS

In this section, we give a brief review of RKHS of vector-valued functions[1], for more detail see e.g. (Carmeli et al., 2006; Micchelli and Pontil, 2005; Caponnetto et al., 2008; Minh and Sindhwani, 2011). In the following, denote by $\mathcal{X}$ a nonempty set, $\mathcal{W}$ a real, separable Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathcal{W}}$, $\mathcal{L}(\mathcal{W})$ the Banach space of bounded linear operators on $\mathcal{W}$. Let $\mathcal{W}^{\mathcal{X}}$ denote the vector space of all functions $f : \mathcal{X} \to \mathcal{W}$. A function $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{W})$ is said to be an **operator-valued positive definite kernel** if for each pair $(x, z) \in \mathcal{X} \times \mathcal{X}$, $K(x, z)^* = K(z, x)$, and

$$\sum_{i,j=1}^{N} \langle y_i, K(x_i, x_j) y_j \rangle_{\mathcal{W}} \geq 0 \tag{1}$$

for every finite set of points $\{x_i\}_{i=1}^N$ in $\mathcal{X}$ and $\{y_i\}_{i=1}^N$ in $\mathcal{W}$. Given such a K, there exists a unique $\mathcal{W}$-valued RKHS $\mathcal{H}_K$ with reproducing kernel K, which is constructed as follows. For each $x \in \mathcal{X}$ and $y \in \mathcal{W}$, form a function $K_x y = K(., x)y \in \mathcal{W}^{\mathcal{X}}$ defined by

$$(K_x y)(z) = K(z, x)y \quad \text{for all} \quad z \in \mathcal{X}.$$

Consider the set $\mathcal{H}_0 = \text{span}\{K_x y \mid x \in \mathcal{X},\ y \in \mathcal{W}\} \subset \mathcal{W}^{\mathcal{X}}$. For $f = \sum_{i=1}^N K_{x_i} w_i$, $g = \sum_{i=1}^N K_{z_i} y_i \in \mathcal{H}_0$, we define the inner product

$$\langle\ f, g\ \rangle_{\mathcal{H}_K} = \sum_{i,j=1}^{N} \langle w_i, K(x_i, z_j) y_j \rangle_{\mathcal{W}},$$

which makes $\mathcal{H}_0$ a pre-Hilbert space. Completing $\mathcal{H}_0$ by adding the limits of all Cauchy sequences gives the Hilbert space $\mathcal{H}_K$. The **reproducing property** is

$$\langle f(x), y \rangle_{\mathcal{W}} = \langle f, K_x y \rangle_{\mathcal{H}_K} \quad \text{for all} \quad f \in \mathcal{H}_K. \tag{2}$$

1. Some authors, e.g. (Kadri et al., 2011) employ the terminology *function-valued*, which is equivalent to *vector-valued*: a function is a vector in a vector space of functions (e.g. a Hilbert space of functions), and an n-dimensional vector is a discrete function defined on n points.

Sampling Operators. For each $x \in \mathcal{X}$, let $K_x : \mathcal{W} \to \mathcal{H}_K$ be the operator with $K_x y$ defined as above, then

$$||K_x y||^2_{\mathcal{H}_K} = \langle K(x, x)y, y \rangle_{\mathcal{W}} \leq ||K(x, x)||\ ||y||^2_{\mathcal{W}},$$

which implies that

$$||K_x : \mathcal{W} \to \mathcal{H}_K|| \leq \sqrt{||K(x, x)||},$$

so that K_x is a bounded operator. Let $K_x^* : \mathcal{H}_K \to \mathcal{W}$ be the adjoint operator of K_x, then from (2), we have

$$f(x) = K_x^* f \qquad \text{for all} \quad x \in \mathcal{X}, f \in \mathcal{H}_K. \tag{3}$$

From this we deduce that for all $x \in \mathcal{X}$ and all $f \in \mathcal{H}_K$,

$$||f(x)||_{\mathcal{W}} \leq ||K_x^*||\ ||f||_{\mathcal{H}_K} \leq \sqrt{||K(x, x)||}\ ||f||_{\mathcal{H}_K},$$

that is the *sampling operator* $S_x : \mathcal{H}_K \to \mathcal{W}$ defined by

$$S_x f = K_x^* f = f(x)$$

is bounded. Let $\mathbf{x} = (x_i)_{i=1}^l \in \mathcal{X}^l$, $l \in \mathbb{N}$. For the sampling operator $S_{\mathbf{x}} : \mathcal{H}_K \to \mathcal{W}^l$ defined by $S_{\mathbf{x}}(f) = (f(x_i))_{i=1}^l$, for any $\mathbf{y} = (y_i)_{i=1}^l \in \mathcal{W}^l$,

$$\begin{aligned} \langle S_{\mathbf{x}} f, \mathbf{y} \rangle_{\mathcal{W}^l} &= \sum_{i=1}^{l} \langle f(x_i), y_i \rangle_{\mathcal{W}} = \sum_{i=1}^{l} \langle K_{x_i}^* f, y_i \rangle_{\mathcal{H}_K} \\ &= \sum_{i=1}^{l} \langle f, K_{x_i} y_i \rangle_{\mathcal{H}_K} = \langle f, \sum_{i=1}^{l} K_{x_i} y_i \rangle_{\mathcal{H}_K}. \end{aligned}$$

Thus the adjoint operator $S_{\mathbf{x}}^* : \mathcal{W}^l \to \mathcal{H}_K$ is given by

$$S_{\mathbf{x}}^* \mathbf{y} = S_{\mathbf{x}}^*(y_1, \ldots, y_l) = \sum_{i=1}^{l} K_{x_i} y_i, \quad \mathbf{y} \in \mathcal{W}^l, \tag{4}$$

and the operator $S_{\mathbf{x}}^* S_{\mathbf{x}} : \mathcal{H}_K \to \mathcal{H}_K$ is given by

$$S_{\mathbf{x}}^* S_{\mathbf{x}} f = \sum_{i=1}^{l} K_{x_i} f(x_i) = \sum_{i=1}^{l} K_{x_i} K_{x_i}^* f. \tag{5}$$

Data-dependent Semi-norms. Let $(x_1, \ldots, x_{u+l}) \subset \mathcal{X}$, $u, l \in \mathbb{N}$. Let $M : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l} \in \mathcal{L}(\mathcal{W}^{u+l})$ be a symmetric, positive operator, that is $\langle y, My \rangle_{\mathcal{W}^{u+l}} \geq 0$ for all $y \in \mathcal{W}^{u+l}$. For $f \in \mathcal{H}_K$, let $\mathbf{f} = (f(x_1), \ldots, f(x_{u+l})) \in \mathcal{W}^{u+l}$. The operator $M : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ can be expressed as an operator-valued matrix $M = (M_{ij})_{i,j=1}^{u+l}$ of size $(u+l) \times (u+l)$, with each $M_{ij} : \mathcal{W} \to \mathcal{W}$ being a linear operator, so that

$$(M\mathbf{f})_i = \sum_{j=1}^{u+l} M_{ij} \mathbf{f}_j = \sum_{j=1}^{u+l} M_{ij} f(x_j). \tag{6}$$

We can then define the following semi-norm for f, which depends on the x_i's:

$$\langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} = \sum_{i,j=1}^{u+l} \langle f(x_i), M_{ij} f(x_j) \rangle_{\mathcal{W}}. \tag{7}$$

This form of semi-norm was utilized in vector-valued manifold regularization (Minh and Sindhwani, 2011).

3. General Learning Framework

In this section, we state the general minimization problem that we wish to solve, which includes Vector-valued Manifold Regularization and Multi-view Learning as special cases.

Let the input space be $\mathcal{X}$, an arbitrary non-empty set. Let $\mathcal{Y}$ be a separable Hilbert space, denoting the output space. Assume that there is an unknown probability measure ρ on $\mathcal{X} \times \mathcal{Y}$, and that we have access to a random training sample $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^{l} \cup \{x_i\}_{i=l+1}^{u+l}$ of l labeled and u unlabeled examples.

Let $\mathcal{W}$ be a separable Hilbert space. Let $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{W})$ be an operator-valued positive definite kernel and $\mathcal{H}_K$ its induced Reproducing Kernel Hilbert Space of $\mathcal{W}$-valued functions.

Let $M : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ be a symmetric, positive operator. For each $f \in \mathcal{H}_K$, let

$$\mathbf{f} = (f(x_1), \ldots, f(x_{u+l})) \in \mathcal{W}^{u+l}. \tag{8}$$

Let $V : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ be a convex loss function. Let $C : \mathcal{W} \to \mathcal{Y}$ be a bounded linear operator, with $C^* : \mathcal{Y} \to \mathcal{W}$ its adjoint operator.

The following is the general minimization problem that we wish to solve:

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^{l} V(y_i, Cf(x_i)) + \gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}}, \tag{9}$$

with regularization parameters $\gamma_A > 0$, $\gamma_I \geq 0$.

Let us give a general multi-view learning interpretation of the different terms in our framework. If each input instance x has many views, then $f(x) \in \mathcal{W}$ represents the output values from all the views, constructed by their corresponding hypothesis spaces. These values are combined by the operator C to give the final output value in $\mathcal{Y}$, which is not necessarily the same as $\mathcal{W}$. In (9), the first term measures the error between the final output $Cf(x_i)$ for x_i with the given output y_i, $1 \leq i \leq l$.

The second summand is the standard RKHS regularization term.

The third summand, Multi-view Manifold Regularization, is a generalization of vector-valued Manifold Regularization in (Minh and Sindhwani, 2011) and Multi-view Point Cloud regularization in (Rosenberg et al., 2009): if there is only one view, then it is simply manifold regularization; if there are many views, then it consists of manifold regularization along each view, as well as consistency regularization across different views. We describe one concrete realization of this term in Section 4.2.

Remark 1 *The framework is readily generalizable to the case the point evaluation functional $f(x)$ is replaced by a general bounded linear operator. We describe this in Appendix B.*

3.1 Representer Theorem

The minimization problem (9) is guaranteed to always have a unique global solution, whose form is given by the following Representer Theorem.

Theorem 2 *The minimization problem (9) has a unique solution, given by $f_{\mathbf{z},\gamma} = \sum_{i=1}^{u+l} K_{x_i} a_i$ for some vectors $a_i \in \mathcal{W}$, $1 \leq i \leq u+l$.*

In the next two sections, we derive the forms of the solution $f_{\mathbf{z},\gamma}$ for the cases where V is the least square loss and the SVM loss, both in the binary and multi-class settings.

3.2 Least Square Case

For the case V is the least square loss function, we solve the following problem:

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^{l} ||y_i - Cf(x_i)||^2_{\mathcal{Y}} + \gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}}, \tag{10}$$

which has an explicit solution, given by the following.

Theorem 3 *The minimization problem (10) has a unique solution $f_{\mathbf{z},\gamma} = \sum_{i=1}^{u+l} K_{x_i} a_i$, where the vectors $a_i \in \mathcal{W}$ are given by*

$$l\gamma_I \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + C^* C (\sum_{j=1}^{u+l} K(x_i, x_j) a_j) + l\gamma_A a_i = C^* y_i, \tag{11}$$

for $1 \leq i \leq l$, and

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + \gamma_A a_i = 0, \tag{12}$$

for $l+1 \leq i \leq u+l$.

3.2.1 Operator-valued Matrix Formulation

The system of equations (11) and (12) can be reformulated in matrix form, which is more readable and more convenient to implement efficiently. Let $K[\mathbf{x}]$ denote the $(u+l) \times (u+l)$ operator-valued matrix whose (i,j) entry is $K(x_i, x_j)$. Let $J_l^{\mathcal{W},u+l} : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ denote the diagonal matrix whose first l entries on the main diagonal are the identity operator $I : \mathcal{W} \to \mathcal{W}$, with the rest being 0. Let $\mathbf{C}^*\mathbf{C} : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ be the $(u+l) \times (u+l)$ diagonal matrix, with each diagonal entry being $C^*C : \mathcal{W} \to \mathcal{W}$. Let $\mathbf{C}^* : \mathcal{Y}^l \to \mathcal{W}^{u+l}$ be the $(u+l) \times l$ block matrix defined by $\mathbf{C}^* = I_{(u+l)\times l} \otimes C^*$, where $I_{(u+l)\times l} = [I_l, 0_{l \times u}]^T$ and $C^* : \mathcal{Y} \to \mathcal{W}$.

Theorem 4 *The system of equations (11) and (12) in Theorem 3 is equivalent to*

$$(\mathbf{C}^*\mathbf{C} J_l^{\mathcal{W},u+l} K[\mathbf{x}] + l\gamma_I M K[\mathbf{x}] + l\gamma_A I)\mathbf{a} = \mathbf{C}^*\mathbf{y}, \tag{13}$$

which has a unique solution $\mathbf{a}$, where $\mathbf{a} = (a_1, \ldots, a_{u+l})$, $\mathbf{y} = (y_1, \ldots, y_l)$ are considered as column vectors in $\mathcal{W}^{u+l}$ and $\mathcal{Y}^l$, respectively.

3.3 Vector-valued Multi-view SVM

In this section, we give the solution of the optimization problem (9) when V is a generalization of the binary SVM hinge loss function to the multi-class setting. We first point out one main difference between the least square and SVM cases. In the least square case, there is a natural generalization from the scalar setting to the vector-valued setting, which we treated in the previous section. In contrast, in the SVM case, many different versions of the multi-class SVM loss function have been proposed. In the following, we consider a generalization of the Simplex Cone SVM (SC-SVM) loss function proposed by (Mroueh et al., 2012), where it was shown to be theoretically consistent.

Let the input space $\mathcal{X}$ be an arbitrary non-empty set and the output label space be the discrete set $\text{cl}(\mathcal{Y}) = \{1, \ldots, P\}$, with $P \in \mathbb{N}$, $P \geq 2$, representing the number of classes. In this setting, the random sample $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^{l} \cup \{x_i\}_{i=l+1}^{u+l}$ is drawn from $\mathcal{X} \times \text{cl}(\mathcal{Y})$.

Let $\mathcal{W}$ be a separable Hilbert space, $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{W})$ be a positive definite kernel with value in the Banach space of bounded linear operators $\mathcal{L}(\mathcal{W})$ on $\mathcal{W}$, and $\mathcal{H}_K$ be the RKHS of $\mathcal{W}$-valued functions induced by K. Let $\mathcal{Y}$ be a separable Hilbert space. Let $S = [s_1, \ldots, s_P]$ as a matrix, which is potentially infinite, with the ith column being $s_i \in \mathcal{Y}$, then S can be considered as a linear operator $S : \mathbb{R}^P \to \mathcal{Y}$, so that for $\mathbf{b} = (b_i)_{i=1}^P$, $S\mathbf{b} = \sum_{i=1}^P b_i s_i$.

Let $C : \mathcal{W} \to \mathcal{Y}$ be a bounded linear operator. Consider the following minimization problem

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^{l} \sum_{k=1, k \neq y_i}^{P} \max\left(0, -\langle s_k, s_{y_i} \rangle_{\mathcal{Y}} + \langle s_k, Cf(x_i) \rangle_{\mathcal{Y}}\right) + \gamma_A ||f||_{\mathcal{H}_K}^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}}, \tag{14}$$

with regularization parameters $\gamma_A > 0$ and $\gamma_I \geq 0$.

The components of (14) and their multi-class and multi-view learning interpretations are as follows.

The vectors s_k's in S represent the P different classes. One particular case for S, which we employ in our numerical experiments, is the *simplex coding* for multi-class encoding, see e.g. (Hill and Doucet, 2007; Wu and Lange, 2010; Saberian and Vasconcelos, 2011; Mroueh et al., 2012). Recall that a simplex coding is a map $s : \{1, \ldots, P\} \to \mathbb{R}^{P-1}$, such that: (i) $||s_k||^2 = 1$; (ii) $\langle s_j, s_k \rangle = -\frac{1}{P-1}$, $j \neq k$; and (iii) $\sum_{k=1}^P s_k = 0$. The simplex codes s_k's form P maximally and equally separated vectors on the sphere $\mathbb{S}^{P-2}$ in $\mathbb{R}^{P-1}$, each representing one category. For example, for $P = 3$, one set of three $\mathbb{R}^2$-valued code vectors is: $s_1 = (1, 0)$, $s_2 = (-1/2, \sqrt{3}/2)$, $s_3 = (-1/2, -\sqrt{3}/2)$. In general, the simplex codes can be computed by a recursive algorithm, see e.g. (Mroueh et al., 2012). The decoding process is straightforward: given a vector $\mathbf{b} \in \mathbb{R}^{P-1}$, the category we assign to $\mathbf{b}$ is

$$\text{argmax}_{1 \leq k \leq P} \langle \mathbf{b}, s_k \rangle. \tag{15}$$

In the following, we assume that the map s is fixed for each P and also refer to the matrix $S = [s_1, \ldots, s_P]$, with the ith column being s_i, as the simplex coding, whenever this coding scheme is being used.

If the number of classes is P and S is the simplex coding, then $\mathcal{Y} = \mathbb{R}^{P-1}$ and S is a $(P-1) \times P$ matrix. Let the number of views be $m \in \mathbb{N}$. Let $\mathcal{W} = \mathcal{Y}^m = \mathbb{R}^{(P-1)m}$. Then K is a matrix-valued kernel: for each pair $(x, t) \in \mathcal{X} \times \mathcal{X}$, $K(x, t)$ is a $(P-1)m \times (P-1)m$ matrix. The Hilbert space $\mathcal{H}_K$ induced by K consists of functions $f : \mathcal{X} \to \mathcal{W} = \mathbb{R}^{(P-1)m}$, that is for each $x \in \mathcal{X}$, $f(x) = (f^1(x), \ldots, f^m(x)) \in \mathbb{R}^{(P-1)m}$.

In the first component of (14), the loss function

$$\sum_{k=1, k \neq y_i}^{P} \max\left(0, -\langle s_k, s_{y_i} \rangle_{\mathcal{Y}} + \langle s_k, Cf(x_i) \rangle_{\mathcal{Y}}\right)$$

measures the error between the combined outputs from all the views for x_i with every code vector s_k such that $k \neq y_i$. It is a generalization of the SC-SVM loss function proposed in (Mroueh et al., 2012).

For any $x \in \mathcal{X}$,

$$f_{\mathbf{z},\gamma}(x) \in \mathcal{W}, \quad Cf_{\mathbf{z},\gamma}(x) \in \mathcal{Y}, \tag{16}$$

and the category assigned to x is

$$\text{argmax}_{1 \leq k \leq P} \langle s_k, Cf_{\mathbf{z},\gamma}(x) \rangle_{\mathcal{Y}}. \tag{17}$$

Remark 5 *We give the multi-class and multi-view learning interpretations and provide numerical experiments for $\mathcal{Y} = \mathbb{R}^{P-1}$, $\mathcal{W} = \mathcal{Y}^m = \mathbb{R}^{(P-1)m}$, with S being the simplex coding. However, we wish to emphasize that optimization problems (14) and (18) and Theorems 6 and 7 are formulated for $\mathcal{W}$ and $\mathcal{Y}$ being arbitrary separable Hilbert spaces.*

3.3.1 Solution of the Soft-Margin Multi-view SVM

Introducing slack variables ξ_{ki}'s into the optimization problem (14), we obtain the minimization problem

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f \in \mathcal{H}_K, \xi_{ki} \in \mathbb{R}} \frac{1}{l} \sum_{i=1}^{l} \sum_{k=1, k \neq y_i}^{P} \xi_{ki} + \gamma_A ||f||_{\mathcal{H}_K}^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}}, \tag{18}$$

subject to the constraints

$$\xi_{ki} \geq -\langle s_k, s_{y_i} \rangle_{\mathcal{Y}} + \langle s_k, Cf(x_i) \rangle_{\mathcal{Y}}, \quad 1 \leq i \leq l, k \neq y_i, \tag{19}$$

$$\xi_{ki} \geq 0, \quad 1 \leq i \leq l, k \neq y_i. \tag{20}$$

Let $\alpha_i = (\alpha_{1i}, \ldots, \alpha_{Pi})^T \in \mathbb{R}^P$ as a column vector, with $\alpha_{y_i,i} = 0$. Let $\alpha = (\alpha_1, \ldots, \alpha_l) \in \mathbb{R}^{P \times l}$ as a matrix of size $P \times l$.

Theorem 6 *The minimization problem (18) has a unique solution given by*

$$f_{\mathbf{z},\gamma}(x) = \sum_{i=1}^{u+l} K(x, x_i) a_i, \quad a_i \in \mathcal{W}, 1 \leq i \leq u+l, \tag{21}$$

with $\mathbf{a} = (a_1, \ldots, a_{u+l}) \in \mathcal{W}^{u+l}$ given by

$$\mathbf{a} = -\frac{1}{2}(\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1} (I_{(u+l) \times l} \otimes C^* S) \text{vec}(\alpha^{\text{opt}}), \tag{22}$$

where $\otimes$ denotes the Kronecker tensor product, $K[\mathbf{x}]$ is the $(u+l)\times(u+l)$ operator-valued matrix, with entry $K[\mathbf{x}]_{ij}$ being the operator $K(x_i,x_j):\mathcal{W}\to\mathcal{W}$, $I_{(u+l)\times l}$ is the $(u+l)\times l$ matrix of the form $I_{(u+l)\times l}=[I_l\ \ 0_{l\times u}]^T$, and $\alpha^{\text{opt}}=(\alpha_1^{\text{opt}},\ldots,\alpha_l^{\text{opt}})\in\mathbb{R}^{P\times l}$ is a solution of the quadratic minimization problem

$$\alpha^{\text{opt}}=\text{argmin}_{\alpha\in\mathbb{R}^{P\times l}}\frac{1}{4}\text{vec}(\alpha)^TQ[\mathbf{x},C]\text{vec}(\alpha)+\sum_{i=1}^{l}\sum_{k=1}^{P}\langle s_k,s_{y_i}\rangle_{\mathcal{Y}}\alpha_{ki}, \tag{23}$$

subject to the constraints

$$0\le\alpha_{ki}\le\frac{1}{l}(1-\delta_{k,y_i}),\quad 1\le i\le l, 1\le k\le P. \tag{24}$$

The symmetric, positive semidefinite, $Pl\times Pl$ matrix $Q[\mathbf{x},C]$ is given by

$$Q[\mathbf{x},C]=(I_{(u+l)\times l}^T\otimes S^*C)K[\mathbf{x}](\gamma_IMK[\mathbf{x}]+\gamma_AI)^{-1}(I_{(u+l)\times l}\otimes C^*S). \tag{25}$$

If S is the simplex coding, then

$$\alpha^{\text{opt}}=\text{argmin}_{\alpha\in\mathbb{R}^{P\times l}}\frac{1}{4}\text{vec}(\alpha)^TQ[\mathbf{x},C]\text{vec}(\alpha)-\frac{1}{P-1}\mathbf{1}_{Pl}^T\text{vec}(\alpha), \tag{26}$$

with $\mathbf{1}_{Pl}=(1,\ldots,1)^T\in\mathbb{R}^{Pl}$, under the same constraints.

Special case: Simplex Cone Support Vector Machine (Mroueh et al., 2012). For $u=0$, $\gamma_I=0$, $\mathcal{W}=\mathcal{Y}=\mathbb{R}^{P-1}$, $C=I_{P-1}$ (single-view), we obtain

$$\mathbf{a}=-\frac{1}{2\gamma_A}(I_l\otimes S)\text{vec}(\alpha^{\text{opt}}), \tag{27}$$

$$Q[\mathbf{x},C]=\frac{1}{\gamma_A}(I_l\otimes S^*)K[\mathbf{x}](I_l\otimes S), \tag{28}$$

If S is the simplex coding, these together give us the quadratic optimization problem for the Simplex Cone Support Vector Machine (SC-SVM) of (Mroueh et al., 2012).

3.3.2 An Equivalent Formulation

For $P=2$, the simplex coding is $S=[1,-1]$. With this choice of S and $\mathcal{W}=\mathcal{Y}=\mathbb{R}$, $C=1$, our formulation reduces to single-view binary SVM with manifold regularization, which is precisely the Laplacian SVM of (Belkin et al., 2006). In this section, we give an equivalent result to Theorem 6, namely Theorem 7 below, which includes the formulation of the Laplacian SVM as a special case.

Let $S_{\hat{y}_i}$ be the matrix obtained from S by removing the y_ith column and $\beta_i\in\mathbb{R}^{P-1}$ be the vector obtained from α_i by deleting the y_ith entry, which is equal to zero by assumption. As a linear operator, $S_{\hat{y}_i}:\mathbb{R}^{P-1}\to\mathcal{Y}$ and

$$S\alpha_i=\sum_{k=1,k\ne y_i}^{P}\alpha_{ki}s_k=S_{\hat{y}_i}\beta_i. \tag{29}$$

Let $\text{diag}(S_{\hat{\mathbf{y}}})$ be the $l\times l$ block diagonal matrix, with block (i,i) being $S_{\hat{y}_i}$ and $\beta=(\beta_1,\ldots,\beta_l)$ be the $(P-1)\times l$ matrix with column i being β_i. As a linear operator, $\text{diag}(S_{\hat{\mathbf{y}}}):\mathbb{R}^{(P-1)l}\to\mathcal{Y}^l$.

Theorem 7 *The minimization problem (18) has a unique solution given by $f_{\mathbf{z},\gamma}(x)=\sum_{i=1}^{u+l}K(x,x_i)a_i$, with $\mathbf{a}=(a_1,\ldots,a_{u+l})\in\mathcal{W}^{u+l}$ given by*

$$\mathbf{a}=-\frac{1}{2}(\gamma_IMK[\mathbf{x}]+\gamma_AI)^{-1}(I_{(u+l)\times l}\otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta^{\text{opt}}), \tag{30}$$

where $\beta^{\text{opt}}=(\beta_1^{\text{opt}},\ldots,\beta_l^{\text{opt}})\in\mathbb{R}^{(P-1)\times l}$ is a solution of the quadratic minimization problem

$$\beta^{\text{opt}}=\text{argmin}_{\beta\in\mathbb{R}^{(P-1)\times l}}\frac{1}{4}\text{vec}(\beta)^TQ[\mathbf{x},\mathbf{y},C]\text{vec}(\beta)+\sum_{i=1}^{l}\langle s_{y_i},S_{\hat{y}_i}\beta_i\rangle_{\mathcal{Y}}, \tag{31}$$

subject to the constraints

$$0\le\beta_{ki}\le\frac{1}{l},\quad 1\le i\le l, 1\le k\le P-1. \tag{32}$$

The symmetric, positive semidefinite, $(P-1)l\times(P-1)l$ matrix $Q[\mathbf{x},\mathbf{y},C]$ is given by

$$Q[\mathbf{x},\mathbf{y},C]=\text{diag}(S_{\hat{\mathbf{y}}}^*)(I_{(u+l)\times l}^T\otimes C)K[\mathbf{x}](\gamma_IMK[\mathbf{x}]+\gamma_AI)^{-1}(I_{(u+l)\times l}\otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}}). \tag{33}$$

If S is the simplex coding, then, under the same constraints,

$$\beta^{\text{opt}}=\text{argmin}_{\beta\in\mathbb{R}^{(P-1)\times l}}\frac{1}{4}\text{vec}(\beta)^TQ[\mathbf{x},\mathbf{y},C]\text{vec}(\beta)-\frac{1}{P-1}\mathbf{1}_{(P-1)l}^T\text{vec}(\beta). \tag{34}$$

It is straightforward to switch between α and β. Let $I_{P,\hat{y}_i}$ be the $P\times(P-1)$ matrix obtained by removing the y_i column from the $P\times P$ identity matrix, then

$$\alpha_i=I_{P,\hat{y}_i}\beta_i\quad\text{and}\quad\beta_i=I_{P,\hat{y}_i}^T\alpha_i. \tag{35}$$

Binary case with simplex coding. For $P=2$, we represent the discrete output label set $\text{cl}(\mathcal{Y})$ as $\text{cl}(\mathcal{Y})=\{\pm1\}$. In this case, β is simply a vector in $\mathbb{R}^l$, and we solve the optimization problem

$$\beta^{\text{opt}}=\text{argmin}_{\beta\in\mathbb{R}^l}\frac{1}{4}\beta^TQ[\mathbf{x},\mathbf{y},C]\beta-\mathbf{1}_l^T\beta, \tag{36}$$

subject to the constraints $0\le\beta_i\le\frac{1}{l}$, $1\le i\le l$. The binary simplex code is $S=[1,-1]$, with $S_{\hat{1}}=-1$ and $S_{\hat{-1}}=1$. Thus $S_{\hat{y}_i}=-y_i$. Furthermore, because $\mathcal{Y}=\mathbb{R}$, by the Riesz representation theorem, the bounded linear operator $C:\mathcal{W}\to\mathbb{R}$ and its adjoint $C^*:\mathbb{R}\to\mathcal{W}$ necessarily have the form

$$Cf(x)=\langle\mathbf{c},f(x)\rangle_{\mathcal{W}}\quad\text{and}\quad C^*y=y\mathbf{c}, \tag{37}$$

respectively, for a unique vector $\mathbf{c}\in\mathcal{W}$. It follows immediately that

Corollary 8 (Binary case) *Let S be the simplex coding and $P=2$. Then in Theorem 7,*

$$\mathbf{a}=\frac{1}{2}(\gamma_IMK[\mathbf{x}]+\gamma_AI)^{-1}(I_{(u+l)\times l}\otimes\mathbf{c})\text{diag}(\mathbf{y})(\beta^{\text{opt}}), \tag{38}$$

$$Q[\mathbf{x},\mathbf{y},C]=\text{diag}(\mathbf{y})(I_{(u+l)\times l}^T\otimes\mathbf{c}^T)K[\mathbf{x}](\gamma_IMK[\mathbf{x}]+\gamma_AI)^{-1}(I_{(u+l)\times l}\otimes\mathbf{c})\text{diag}(\mathbf{y}). \tag{39}$$

Special case: Laplacian SVM (Belkin et al., 2006). In (38) and (39), by setting $\mathbf{c}=1$ ($\mathcal{W}=\mathcal{Y}=\mathbb{R}$) (single-view) and M to be the graph Laplacian on the training data $\{x_i\}_{i=1}^{u+l}$, we obtain the Laplacian SVM of (Belkin et al., 2006).

3.4 Previous Work as Special Cases of the Current Framework

We have shown above that in the SVM case, our framework includes the multi-class, supervised Simplex Cone SVM of (Mroueh et al., 2012) and the binary, semi-supervised Laplacian SVM of (Belkin et al., 2006) as special cases. Before delving into concrete implementations, in this section we give a list of other common kernel-based learning algorithms which are special cases of our learning framework.

Vector-valued Regularized Least Squares. If $\mathbf{C}^*\mathbf{C} = I : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$, then (13) reduces to

$$(J_l^{\mathcal{W},u+l} K[\mathbf{x}] + l\gamma_I M K[\mathbf{x}] + l\gamma_A I)\mathbf{a} = \mathbf{C}^*\mathbf{y}. \tag{40}$$

If $u = 0$, $\gamma_I = 0$, and $\gamma_A = \gamma$, then we have

$$(K[\mathbf{x}] + l\gamma I)\mathbf{a} = \mathbf{C}^*\mathbf{y}. \tag{41}$$

One particular case for this scenario is when $\mathcal{W} = \mathcal{Y}$ and $C : \mathcal{Y} \to \mathcal{Y}$ is a unitary operator, that is $C^*C = CC^* = I$. If $\mathcal{Y} = \mathbb{R}^n$ and $C : \mathbb{R}^n \to \mathbb{R}^n$ is real, then C is an orthogonal matrix. If $C = I$, then we recover the vector-valued Regularized Least Squares algorithm (Micchelli and Pontil, 2005).

Vector-valued Manifold Regularization. Let $\mathcal{W} = \mathcal{Y}$ and $C = I$. Then we obtain the minimization problem for vector-valued Manifold Regularization (Minh and Sindhwani, 2011):

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f \in \mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l} V(y_i, f(x_i)) + \gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}. \tag{42}$$

Scalar Multi-view Learning. Let us show that the scalar multi-view learning formulation of (Sindhwani and Rosenberg, 2008; Rosenberg et al., 2009) can be cast as a special case of our framework. Let $\mathcal{Y} = \mathbb{R}$ and $k^1, \ldots, k^m$ be real-valued positive definite kernels on $\mathcal{X} \times \mathcal{X}$, with corresponding RKHS $\mathcal{H}_{k^i}$ of functions $f^i : \mathcal{X} \to \mathbb{R}$, with each $\mathcal{H}_{k^i}$ representing one view. Let $f = (f^1, \ldots, f^m)$, with $f^i \in \mathcal{H}_{k^i}$. Let $\mathbf{c} = (c_1, \ldots, c_m) \in \mathbb{R}^m$ be a fixed weight vector. In the notation of (Rosenberg et al., 2009), let

$$\mathbf{f} = (f^1(x_1), \ldots, f^1(x_{u+l}), \ldots, f^m(x_1), \ldots, f^m(x_{u+l}))$$

and $M \in \mathbb{R}^{m(u+l)\times m(u+l)}$ be positive semidefinite. The objective of Multi-view Point Cloud Regularization (formula (4) in (Rosenberg et al., 2009)) is

$$\text{argmin}_{\varphi:\varphi(x)=\langle \mathbf{c}, f(x)\rangle} \frac{1}{l}\sum_{i=1}^{l} V(y_i, \varphi(x_i)) + \sum_{i=1}^{m} \gamma_i ||f^i||^2_{k^i} + \gamma \langle \mathbf{f}, M\mathbf{f}\rangle_{\mathbb{R}^{m(u+l)}}, \tag{43}$$

for some convex loss function V, with $\gamma_i > 0$, $i = 1, \ldots, m$, and $\gamma \geq 0$. Problem (43) admits a natural formulation in vector-valued RKHS. Let

$$K = \text{diag}(\frac{1}{\gamma_1}, \ldots, \frac{1}{\gamma_m}) * \text{diag}(k^1, \ldots, k^m) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{m\times m}, \tag{44}$$

then $f = (f^1, \ldots, f^m) \in \mathcal{H}_K : \mathcal{X} \to \mathbb{R}^m$, with

$$||f||^2_{\mathcal{H}_K} = \sum_{i=1}^{m} \gamma_i ||f^i||^2_{k^i}. \tag{45}$$

By the reproducing property, we have

$$\langle \mathbf{c}, f(x)\rangle_{\mathbb{R}^m} = \langle f, K_x\mathbf{c}\rangle_{\mathcal{H}_K}. \tag{46}$$

We can now recast (43) into

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l} V(y_i, \langle \mathbf{c}, f(x_i)\rangle_{\mathbb{R}^m}) + ||f||^2_{\mathcal{H}_K} + \gamma\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathbb{R}^{m(u+l)}}. \tag{47}$$

This is a special case of (9), with $\mathcal{W} = \mathbb{R}^m$, $\mathcal{Y} = \mathbb{R}$, and $C : \mathbb{R}^m \to \mathbb{R}$ given by

$$Cf(x) = \langle \mathbf{c}, f(x)\rangle_{\mathbb{R}^m} = c_1 f^1(x) + \cdots + c_m f^m(x). \tag{48}$$

The vector-valued formulation of scalar multi-view learning has the following advantages:

(i) The kernel K is diagonal matrix-valued and is obviously positive definite. In contrast, it is nontrivial to prove that the multi-view kernel of (Rosenberg et al., 2009) is positive definite.

(ii) The kernel K is independent of the c_i's, unlike the multi-view kernel of (Rosenberg et al., 2009), which needs to be recomputed for each different set c_i's.

(iii) One can recover all the component functions f^i's using K. In contrast, in (Sindhwani and Rosenberg, 2008), it is shown how one can recover the f^i's only when $m = 2$, but not in the general case.

4. Vector-valued Multi-view Learning

In this and subsequent sections, we focus on a special case of our formulation, namely vector-valued multi-view learning. For a general separable Hilbert space $\mathcal{Y}$, let $\mathcal{W} = \mathcal{Y}^m$ and $C_1, \ldots, C_m : \mathcal{Y} \to \mathcal{Y}$ be bounded linear operators. For $f(x) = (f^1(x), \ldots, f^m(x))$, with each $f^i(x) \in \mathcal{Y}$, we define the combination operator $C = [C_1, \ldots, C_m] : \mathcal{Y}^m \to \mathcal{Y}$ by

$$Cf(x) = C_1 f^1(x) + \cdots + C_m f^m(x) \in \mathcal{Y}. \tag{49}$$

This gives rise to a vector-valued version of multi-view learning, where outputs from m views, each one being a vector in the Hilbert space $\mathcal{Y}$, are linearly combined. In the following, we give concrete definitions of both the combination operator C and the multi-view manifold regularization term M for our multi-view learning model.

4.1 The Combination Operator

In the present context, the bounded linear operator $C : \mathcal{W} \to \mathcal{Y}$ is a (potentially infinite) matrix of size $\dim(\mathcal{Y}) \times m\dim(\mathcal{Y})$. This operator transforms the output vectors obtained from the m views f^i's in $\mathcal{Y}^m$ into an output vector in $\mathcal{Y}$. The simplest form of C is the average operator:

$$Cf(x) = \frac{1}{m}(f^1(x) + \cdots + f^m(x)) \in \mathcal{Y}. \tag{50}$$

Let $\otimes$ denote the Kronecker tensor product. For $m \in \mathbb{N}$, let $\mathbf{1}_m = (1, \ldots, 1)^T \in \mathbb{R}^m$. The matrix C is then

$$C = \frac{1}{m}\mathbf{1}_m^T \otimes I_{\mathcal{Y}} = \frac{1}{m}[I_{\mathcal{Y}}, \ldots, I_{\mathcal{Y}}]. \tag{51}$$

More generally, we consider a weight vector $\mathbf{c} = (c_1, \ldots, c_m)^T \in \mathbb{R}^m$ and define C as

$$C = \mathbf{c}^T \otimes I_{\mathcal{Y}}, \text{ with } Cf(x) = \sum_{i=1}^{m} c_i f^i(x) \in \mathcal{Y}. \tag{52}$$

4.2 Multi-view Manifold Regularization

Generalizing the formulation in (Minh et al., 2013), we decompose the multi-view manifold regularization term $\gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}}$ in (9) into two components

$$\gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} = \gamma_B \langle \mathbf{f}, M_B \mathbf{f} \rangle_{\mathcal{W}^{u+l}} + \gamma_W \langle \mathbf{f}, M_W \mathbf{f} \rangle_{\mathcal{W}^{u+l}}, \tag{53}$$

where $M_B, M_W : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ are symmetric, positive operators, and $\gamma_B, \gamma_W \geq 0$. We call the first term *between-view regularization*, which measures the consistency of the component functions across different views, and the second term *within-view regularization*, which measures the smoothness of the component functions in their corresponding views. We describe next two concrete choices for M_B and M_W.

Between-view Regularization. Let

$$M_m = mI_m - \mathbf{1}_m \mathbf{1}_m^T. \tag{54}$$

This is the $m \times m$ matrix with $(m-1)$ on the diagonal and -1 elsewhere. Then for $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{R}^m$,

$$\mathbf{a}^T M_m \mathbf{a} = \sum_{j,k=1, j<k}^{m} (a_j - a_k)^2. \tag{55}$$

If each $a_i \in \mathcal{Y}$, then we have $\mathbf{a} \in \mathcal{Y}^m$ and

$$\mathbf{a}^T (M_m \otimes I_{\mathcal{Y}}) \mathbf{a} = \sum_{j,k=1, j<k}^{m} ||a_j - a_k||_{\mathcal{Y}}^2. \tag{56}$$

We define M_B by

$$M_B = I_{u+l} \otimes (M_m \otimes I_{\mathcal{Y}}). \tag{57}$$

Then M_B is a diagonal block matrix of size $m(u+l)\dim(\mathcal{Y}) \times m(u+l)\dim(\mathcal{Y})$, with each block (i,i) being $M_m \otimes I_{\mathcal{Y}}$. For $\mathbf{f} = (f(x_1), \ldots, f(x_{u+l})) \in \mathcal{Y}^{m(u+l)}$, with $f(x_i) \in \mathcal{Y}^m$,

$$\langle \mathbf{f}, M_B \mathbf{f} \rangle_{\mathcal{Y}^{m(u+l)}} = \sum_{i=1}^{u+l} \langle f(x_i), (M_m \otimes I_{\mathcal{Y}}) f(x_i) \rangle_{\mathcal{Y}^m} = \sum_{i=1}^{u+l} \sum_{j,k=1, j<k}^{m} ||f^j(x_i) - f^k(x_i)||_{\mathcal{Y}}^2. \tag{58}$$

This term thus enforces the consistency between the different components f^i's which represent the outputs on the different views. For $\mathcal{Y} = \mathbb{R}$, this is precisely the Point Cloud regularization term for scalar multi-view learning (Rosenberg et al., 2009; Brefeld et al., 2006). In particular, for $m = 2$, we have $M_2 = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$, and

$$\langle \mathbf{f}, M_B \mathbf{f} \rangle_{\mathbb{R}^{2(u+l)}} = \sum_{i=1}^{u+l} (f^1(x_i) - f^2(x_i))^2, \tag{59}$$

which is the Point Cloud regularization term for co-regularization (Sindhwani and Rosenberg, 2008).

Within-view Regularization. One way to define M_W is via the graph Laplacian. For view i, $1 \leq i \leq m$, let G^i be a corresponding undirected graph, with symmetric, nonnegative weight matrix W^i, which induces the scalar graph Laplacian L^i, a matrix of size $(u+l) \times (u+l)$. For a vector $\mathbf{a} \in \mathbb{R}^{u+l}$, we have

$$\mathbf{a}^T L^i \mathbf{a} = \sum_{j,k=1, j<k}^{u+l} W_{jk}^i (a_j - a_k)^2.$$

Let L be the block matrix of size $(u+l) \times (u+l)$, with block (i,j) being the $m \times m$ diagonal matrix given by

$$L_{i,j} = \text{diag}(L_{ij}^1, \ldots L_{ij}^m). \tag{60}$$

Then for $\mathbf{a} = (a_1, \ldots, a_{u+l})$, with $a_j \in \mathbb{R}^m$, we have

$$\mathbf{a}^T L \mathbf{a} = \sum_{i=1}^{m} \sum_{j,k=1, j<k}^{u+l} W_{jk}^i (a_j^i - a_k^i)^2. \tag{61}$$

If $a_j \in \mathcal{Y}^m$, with $a_j^i \in \mathcal{Y}$, then

$$\mathbf{a}^T (L \otimes I_{\mathcal{Y}}) \mathbf{a} = \sum_{i=1}^{m} \sum_{j,k=1, j<k}^{u+l} W_{jk}^i ||a_j^i - a_k^i||_{\mathcal{Y}}^2. \tag{62}$$

Define

$$M_W = L \otimes I_{\mathcal{Y}}, \quad \text{then} \tag{63}$$

$$\langle \mathbf{f}, M_W \mathbf{f} \rangle_{\mathcal{Y}^{m(u+l)}} = \sum_{i=1}^{m} \sum_{j,k=1, j<k}^{u+l} W_{jk}^i ||f^i(x_j) - f^i(x_k)||_{\mathcal{Y}}^2. \tag{64}$$

The ith summand in the sum $\sum_{i=1}^m$ is precisely a manifold regularization term within view i. This term thus enforces the consistency of the output along each view i, $1 \leq i \leq m$.

Single View Case. When $m = 1$, we have $M_m = 0$ and therefore $M_B = 0$. In this case, we simply carry out manifold regularization within the given single view, using M_W.

5. Numerical Implementation

In this section, we give concrete forms of Theorems 4, for vector-valued multi-view least squares regression, and Theorem 6, for vector-valued multi-view SVM, that can be efficiently implemented. For our present purposes, let $m \in \mathbb{N}$ be the number of views and $\mathcal{W} = \mathcal{Y}^m$. Consider the case $\dim(\mathcal{Y}) < \infty$. Without loss of generality, we set $\mathcal{Y} = \mathbb{R}^{\dim(\mathcal{Y})}$.

The Kernel. For the current implementations, we define the kernel $K(x,t)$ by

$$K(x,t) = G(x,t) \otimes R, \tag{65}$$

where $G : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{m \times m}$ is a matrix-valued positive definite kernel, with $G(x,t)$ being an $m \times m$ matrix for each pair $(x,t) \in \mathcal{X} \times \mathcal{X}$. A concrete example of G, which we use in our experiments, is given in Section 7. The bounded linear operator $R : \mathcal{Y} \to \mathcal{Y}$ is symmetric and positive, and when $\dim(\mathcal{Y}) < \infty$, R is a symmetric, positive semi-definite matrix of size $\dim(\mathcal{Y}) \times \dim(\mathcal{Y})$. The Gram matrices of K and G are block matrices $K[\mathbf{x}]$ and $G[\mathbf{x}]$, respectively, of size $(u+l) \times (u+l)$, with blocks (i,j) given by $(K[\mathbf{x}])_{ij} = K(x_i, x_j)$ and $(G[\mathbf{x}])_{ij} = G(x_i, x_j)$. They are related by

$$K[\mathbf{x}] = G[\mathbf{x}] \otimes R. \tag{66}$$

Lemma 9 *The matrix-valued kernel K is positive definite.*

5.1 Numerical Implementation for Vector-valued Multi-view Least Squares

With the kernel K as defined in (65) and C and M as defined in Section 4, the system of linear equations (13) in Theorem 4 becomes a Sylvester equation, which can be solved efficiently, as follows.

Theorem 10 *For $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$, $\mathbf{c} \in \mathbb{R}^m$, $M_W = L \otimes I_{\mathcal{Y}}$, $M_B = I_{u+l} \otimes (M_m \otimes I_{\mathcal{Y}})$, and the kernel K as defined in (65), the system of linear equations (13) in Theorem 4 is equivalent to the Sylvester equation*

$$BAR + l\gamma_A A = Y_C, \tag{67}$$

where

$$B = \left((J_l^{u+l} \otimes \mathbf{c}\mathbf{c}^T) + l\gamma_B (I_{u+l} \otimes M_m) + l\gamma_W L \right) G[\mathbf{x}], \tag{68}$$

which is of size $(u+l)m \times (u+l)m$, A is the matrix of size $(u+l)m \times \dim(\mathcal{Y})$ such that $\mathbf{a} = \mathrm{vec}(A^T)$, and Y_C is the matrix of size $(u+l)m \times \dim(\mathcal{Y})$ such that $\mathbf{C}^*\mathbf{y} = \mathrm{vec}(Y_C^T)$. $J_l^{u+l} : \mathbb{R}^{u+l} \to \mathbb{R}^{u+l}$ is a diagonal matrix of size $(u+l) \times (u+l)$, with the first l entries on the main diagonal being 1 and the rest being 0.

Special cases. For $m = 1$, $\mathbf{c} = 1$, equation (67) reduces to Equation 17 in (Minh and Sindhwani, 2011). For $R = I_{\mathcal{Y}}$, with $\mathcal{Y} = \mathbb{R}^P$, equation (67) reduces to Equation 43 in (Minh et al., 2013).

Evaluation on a testing sample. Having solved for the matrix A, and hence the vector $\mathbf{a}$ in Theorem 10, we next show how the resulting functions can be efficiently evaluated on a testing set. Let $\mathbf{v} = \{v_1, \ldots, v_t\} \in \mathcal{X}$ be an arbitrary set of testing input examples, with $t \in \mathbb{N}$. Let $\mathbf{f}_{\mathbf{z},\gamma}(\mathbf{v}) = (\{f_{\mathbf{z},\gamma}(v_1), \ldots, f_{\mathbf{z},\gamma}(v_t)\})^T \in \mathcal{Y}^{mt}$, with

$$f_{\mathbf{z},\gamma}(v_i) = \sum_{j=1}^{u+l} K(v_i, x_j) a_j.$$

Let $K[\mathbf{v}, \mathbf{x}]$ denote the $t \times (u+l)$ block matrix, where block (i,j) is $K(v_i, x_j)$ and similarly, let $G[\mathbf{v}, \mathbf{x}]$ denote the $t \times (u+l)$ block matrix, where block (i,j) is the $m \times m$ matrix $G(v_i, x_j)$. Then

$$\mathbf{f}_{\mathbf{z},\gamma}(\mathbf{v}) = K[\mathbf{v}, \mathbf{x}]\mathbf{a} = (G[\mathbf{v}, \mathbf{x}] \otimes R)\mathbf{a} = \mathrm{vec}(RA^T G[\mathbf{v}, \mathbf{x}]^T),$$

Algorithm 1 $\mathcal{Y}$-valued, m-view, semi-supervised least square regression and classification

This algorithm implements and evaluates the solution of Theorem 10.

Input:
- Training data $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^l \cup \{x_i\}_{i=l+1}^{u+l}$, with l labeled and u unlabeled examples.
- Number of views: m.
- Output values: vectors in $\mathcal{Y}$.
- Testing example: v.

Parameters:
- The regularization parameters $\gamma_A, \gamma_B, \gamma_W$.
- The weight vector $\mathbf{c}$.
- A matrix-valued kernel G, with $G(x,t)$ being an $m \times m$ matrix for each pair (x,t).

Procedure:
- Compute kernel matrix $G[\mathbf{x}]$ on input set $\mathbf{x} = (x_i)_{i=1}^{u+l}$.
- Compute matrix C according to (52).
- Compute graph Laplacian L according to (60).
- Compute matrices B, Y_C according to Theorem 10.
- Solve matrix equation $BAR + l\gamma_A A = Y_C$ for A.
- Compute kernel matrix $G[v, \mathbf{x}]$ between v and $\mathbf{x}$.

Output: $f_{\mathbf{z},\gamma}(v) = \mathrm{vec}(RA^T G[v, \mathbf{x}]^T) \in \mathcal{Y}^m$.
$\mathcal{Y}$-valued regression: return $Cf_{\mathbf{z},\gamma}(v) \in \mathcal{Y}$.
Multi-class classification: return index of $\max(Cf_{\mathbf{z},\gamma}(v))$.

In particular, for $\mathbf{v} = \mathbf{x} = (x_i)_{i=1}^{u+l}$, the original training sample, we have $G[\mathbf{v}, \mathbf{x}] = G[\mathbf{x}]$.

Algorithm. All the necessary steps for implementing Theorem 10 and evaluating its solution are summarized in Algorithm 1. For P-class classification, $\mathcal{Y} = \mathbb{R}^P$, and $y_i = (-1, \ldots, 1, \ldots, -1)$, $1 \le i \le l$, with 1 at the kth location if x_i is in the kth class.

5.2 Numerical Implementation for Vector-valued Multi-view SVM

This section gives a concrete form of Theorem 6 for vector-valued multi-view SVM which can be efficiently implemented. Let $\{\lambda_{i,R}\}_{i=1}^{\dim(\mathcal{Y})}$ be the eigenvalues of R, which are all nonnegative, with corresponding orthonormal eigenvectors $\{\mathbf{r}_i\}_{i=1}^{\dim(\mathcal{Y})}$. Then R admits the orthogonal spectral decomposition

$$R = \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T. \tag{69}$$

Under this representation of R and with the kernel K as defined in (65), Theorem 6 takes the following concrete form.

Theorem 11 *Let $\gamma_I M = \gamma_B M_B + \gamma_W M_W$, $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$, and $K(x,t)$ be defined as in (65). Then in Theorem 6,*

$$\mathbf{a} = -\frac{1}{2}[\sum_{i=1}^{\dim(\mathcal{Y})} M_{\mathrm{reg}}^i (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \mathbf{r}_i \mathbf{r}_i^T S] \mathrm{vec}(\alpha^{\mathrm{opt}}), \tag{70}$$

$$Q[\mathbf{x}, C] = \sum_{i=1}^{\dim(\mathcal{Y})} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) G[\mathbf{x}] M^i_{\text{reg}} (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \lambda_{i,R} S^* \mathbf{r}_i \mathbf{r}_i^T S, \tag{71}$$

where

$$M^i_{\text{reg}} = [\lambda_{i,R}(\gamma_B I_{u+l} \otimes M_m + \gamma_W L) G[\mathbf{x}] + \gamma_A I_{m(u+l)}]^{-1}. \tag{72}$$

Evaluation phase. Having solved for α^{opt} and hence $\mathbf{a}$ in Theorem 11, we next show how the resulting functions can be efficiently evaluated on a testing set $\mathbf{v} = \{v_i\}_{i=1}^t \subset \mathcal{X}$.

Proposition 12 *Let $f_{\mathbf{z},\gamma}$ be the solution obtained in Theorem 11. For any example $v \in \mathcal{X}$,*

$$f_{\mathbf{z},\gamma}(v) = -\frac{1}{2}\text{vec}[\sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[v, \mathbf{x}]^T]. \tag{73}$$

The combined function, using the combination operator C, is $g_{\mathbf{z},\gamma}(v) = C f_{\mathbf{z},\gamma}(v)$ and is given by

$$g_{\mathbf{z},\gamma}(v) = -\frac{1}{2} \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[v, \mathbf{x}]^T \mathbf{c}. \tag{74}$$

The final SVM decision function is $h_{\mathbf{z},\gamma}(v) = S^T g_{\mathbf{z},\gamma}(v) \in \mathbb{R}^P$ and is given by

$$h_{\mathbf{z},\gamma}(v) = -\frac{1}{2} \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} S^T \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[v, \mathbf{x}]^T \mathbf{c}. \tag{75}$$

On a testing set $\mathbf{v} = \{v_i\}_{i=1}^t \subset \mathcal{X}$,

$$h_{\mathbf{z},\gamma}(\mathbf{v}) = -\frac{1}{2} \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} S^T \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[\mathbf{v}, \mathbf{x}]^T (I_t \otimes \mathbf{c}), \tag{76}$$

as a matrix of size $P \times t$, with the ith column being $h_{\mathbf{z},\gamma}(v_i)$.

Algorithm. All the necessary steps for implementing Theorem 11 and Proposition 12 are summarized in Algorithm 2.

5.2.1 Special case

Consider the case $R = I_{\mathcal{Y}}$. Then Theorem 11 and Proposition 12 simplify to the following.

Theorem 13 *Let $\gamma_I M = \gamma_B M_B + \gamma_W M_W$, $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$, and $K(x,t)$ be defined as in (65) with $R = I_{\mathcal{Y}}$. Then in Theorem 6,*

$$\mathbf{a} = -\frac{1}{2}[M_{\text{reg}}(I_{(u+l)\times l} \otimes \mathbf{c}) \otimes S]\text{vec}(\alpha^{\text{opt}}), \tag{77}$$

and

$$Q[\mathbf{x}, C] = (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) G[\mathbf{x}] M_{\text{reg}} (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes S^* S, \tag{78}$$

where

$$M_{\text{reg}} = [(\gamma_B I_{u+l} \otimes M_m + \gamma_W L) G[\mathbf{x}] + \gamma_A I_{m(u+l)}]^{-1}. \tag{79}$$

Algorithm 2 Multi-class Multi-view SVM

This algorithm implements Theorem 11 and Proposition 12. In the case $R = I_{\mathcal{Y}}$, it implements Theorem 13 and Proposition 14, with $M^i_{\text{reg}} = M_{\text{reg}}$ in (79), and equations (71), (73), and (75) are replaced by (78), (80), and (82), respectively.

Input:
- Training data $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^l \cup \{x_i\}_{i=l+1}^{u+l}$, with l labeled and u unlabeled examples.
- Number of classes: P. Number of views: m.
- Testing example: v.

Parameters:
- The regularization parameters $\gamma_A, \gamma_B, \gamma_W$.
- The weight vector $\mathbf{c}$.
-A matrix-valued kernel G, with $G(x,t)$ being an $m \times m$ matrix for each pair (x,t).

Procedure:
- Compute kernel matrices $G[\mathbf{x}]$ on $\mathbf{x} = (x_i)_{i=1}^{u+l}$ and $G[v, \mathbf{x}]$ between v and $\mathbf{x}$.
- Compute graph Laplacian L according to (60).
- Compute matrices M^i_{reg} according to (72).
- Compute matrix $Q[\mathbf{x}, C]$ according to (71)
- Solve quadratic optimization problem (23) for α^{opt}.

Output: $f_{\mathbf{z},\gamma}(v)$, computed according to (73).
Classification: return $\text{argmax}(h_{\mathbf{z},\gamma}(v))$, with $h_{\mathbf{z},\gamma}(v) \in \mathbb{R}^P$ computed according to (75).

Proposition 14 *Let $f_{\mathbf{z},\gamma}$ be the solution obtained in Theorem 13. For any example $v \in \mathcal{X}$,*

$$f_{\mathbf{z},\gamma}(v) = -\frac{1}{2}\text{vec}(S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) M^T_{\text{reg}} G[v, \mathbf{x}]^T). \tag{80}$$

The combined function, using the combination operator C, is $g_{\mathbf{z},\gamma}(v) = C f_{\mathbf{z},\gamma}(v) \in \mathbb{R}^{P-1}$ and is given by

$$g_{\mathbf{z},\gamma}(v) = -\frac{1}{2} S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) M^T_{\text{reg}} G[v, \mathbf{x}]^T \mathbf{c}. \tag{81}$$

The final SVM decision function is $h_{\mathbf{z},\gamma}(v) = S^T g_{\mathbf{z},\gamma}(v) \in \mathbb{R}^P$ and is given by

$$h_{\mathbf{z},\gamma}(v) = -\frac{1}{2} S^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) M^T_{\text{reg}} G[v, \mathbf{x}]^T \mathbf{c}. \tag{82}$$

On a testing set $\mathbf{v} = \{v_i\}_{i=1}^t$, let $h_{\mathbf{z},\gamma}(\mathbf{v}) \in \mathbb{R}^{P \times t}$ be the matrix with the ith column being $h_{\mathbf{z},\gamma}(v_i)$, then

$$h_{\mathbf{z},\gamma}(\mathbf{v}) = -\frac{1}{2} S^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) M^T_{\text{reg}} G[\mathbf{v}, \mathbf{x}]^T (I_t \otimes \mathbf{c}). \tag{83}$$

5.2.2 Sequential Minimal Optimization (SMO)

We provide an SMO algorithm, which is described in detail in Appendix A.4, to solve the quadratic optimization problem (23) in Theorem 6, as part of Algorithm 2.

6. Optimizing the Combination Operator

In the learning formulation thus far, we have assumed that the combination operator C is given and fixed. Our task then is to find the optimal function $f_{\mathbf{z},\gamma} \in \mathcal{H}_K$ that minimizes the general learning objective (9) in Section 3, given the training data $\mathbf{z}$ and C. In this section, we go one step further and show that both $f_{\mathbf{z},\gamma}$ and C can be simultaneously optimized given the training data $\mathbf{z}$ alone.

For the time being, we consider the m-view least square learning setting, where C is represented by a vector $\mathbf{c} \in \mathbb{R}^m$. Let $\mathcal{S}_\alpha^{m-1}$ denote the sphere centered at the origin in $\mathbb{R}^m$ with radius $\alpha > 0$, that is $\mathcal{S}_\alpha^{m-1} = \{x \in \mathbb{R}^m : \|x\| = \alpha\}$. Consider the problem of optimizing over both $f \in \mathcal{H}_K$ and $\mathbf{c} \in \mathcal{S}_\alpha^{m-1}$,

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f \in \mathcal{H}_K, \mathbf{c} \in \mathcal{S}_\alpha^{m-1}} \frac{1}{l} \sum_{i=1}^{l} ||y_i - Cf(x_i)||_{\mathcal{Y}}^2 + \gamma_A \|f\|_{\mathcal{H}_K}^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}u+l}. \quad (84)$$

We first point out a *crucial difference* between our framework and a typical multi-kernel learning approach. Since our formulation does not place any constraint on $\mathbf{c}$, we do not require that $c_i \geq 0$, $i = 1, \ldots, m$. Thus $\mathbf{c}$ is allowed to range over the whole sphere S_α^{m-1}, which considerably simplifies the optimization procedure.

The optimization problem (84) is not convex and one common approach to tackle it is via Alternating Minimization. First we fix $\mathbf{c} \in S_\alpha^{m-1}$ and solve for the optimal $f_{\mathbf{z},\gamma} \in \mathcal{H}_K$, which is what we have done so far. Then we fix f and solve for $\mathbf{c}$. Consider f of the form

$$f = \sum_{j=1}^{u+l} K_{x_j} a_j.$$

Then

$$f(x_i) = \sum_{j=1}^{u+l} K(x_i, x_j) a_j = K[x_i]\mathbf{a},$$

where $K[x_i] = (K(x_i, x_1), \ldots, K(x_i, x_{u+l}))$. Since $K[x_i] = G[x_i] \otimes R$, we have

$$f(x_i) = (G[x_i] \otimes R)\mathbf{a}, \quad G[x_i] \in \mathbb{R}^{m \times m(u+l)}.$$

Since A is a matrix of size $m(u+l) \times \dim(\mathcal{Y})$, with $\mathbf{a} = \text{vec}(A^T)$, we have

$$Cf(x_i) = (\mathbf{c}^T \otimes I_{\mathcal{Y}})(G[x_i] \otimes R)\mathbf{a} = (\mathbf{c}^T G[x_i] \otimes R)\mathbf{a} = \text{vec}(RA^T G[x_i]^T \mathbf{c}) = RA^T G[x_i]^T \mathbf{c} \in \mathcal{Y}.$$

Let $F[\mathbf{x}]$ be an $l \times 1$ block matrix, with block $F[\mathbf{x}]_i = RA^T G[x_i]^T$, which is of size $\dim(\mathcal{Y}) \times m$, so that $F[\mathbf{x}]$ is of size $\dim(\mathcal{Y})l \times m$ and $F[\mathbf{x}]\mathbf{c} \in \mathcal{Y}^l$. Then

$$\frac{1}{l} \sum_{i=1}^{l} ||y_i - Cf(x_i)||_{\mathcal{Y}}^2 = \frac{1}{l} ||\mathbf{y} - F[\mathbf{x}]\mathbf{c}||_{\mathcal{Y}^l}^2.$$

Thus for f fixed, so that $F[\mathbf{x}]$ is fixed, the minimization problem (84) over $\mathbf{c}$ is equivalent to the following optimization problem

$$\min_{\mathbf{c} \in \mathcal{S}_\alpha^{m-1}} \frac{1}{l} ||\mathbf{y} - F[\mathbf{x}]\mathbf{c}||_{\mathcal{Y}^l}^2. \quad (85)$$

While the sphere S_α^{m-1} is not convex, it is a compact set and consequently, any continuous function on S_α^{m-1} attains a global minimum and a global maximum. We show in the next section how to obtain an almost closed form solution for the global minimum of (85) in the case $\dim(\mathcal{Y}) < \infty$.

6.1 Quadratic Optimization on the Sphere

Let A be an $n \times m$ matrix, $\mathbf{b}$ be an $n \times 1$ vector, and $\alpha > 0$. Consider the optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^m} ||A\mathbf{x} - \mathbf{b}||_{\mathbb{R}^n} \text{ subject to } ||\mathbf{x}||_{\mathbb{R}^m} = \alpha. \quad (86)$$

The function $\psi(\mathbf{x}) = ||A\mathbf{x} - \mathbf{b}||_{\mathbb{R}^n} : \mathbb{R}^m \to \mathbb{R}$ is continuous. Thus over the sphere $\|\mathbf{x}\|_{\mathbb{R}^m} = \alpha$, which is a compact subset of $\mathbb{R}^m$, $\psi(\mathbf{x})$ has a global minimum and a global maximum.

The optimization problem (86) has been analyzed before in the literature under various assumptions, see e.g. (Forsythe and Golub, 1965; Gander, 1981; Golub and von Matt, 1991). In this work, we employ the singular value decomposition approach described in (Gander, 1981), but we *do not impose any constraint* on the matrix A (in (Gander, 1981), it is assumed that $\text{rank}\begin{pmatrix} A \\ I \end{pmatrix} = m$ and $n \geq m$). We next describe the form of the global minimum of $\psi(\mathbf{x})$.

Consider the singular value decomposition for A,

$$A = U\Sigma V^T, \quad (87)$$

where $U \in \mathbb{R}^{n \times n}$, $\Sigma \in \mathbb{R}^{n \times m}$, $V \in \mathbb{R}^{m \times m}$, with $UU^T = U^T U = I_n$, $VV^T = V^T V = I_m$. Let $r = \text{rank}(A)$, $1 \leq r \leq \min\{m, n\}$, then the main diagonal of Σ has the form $(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$, with $\sigma_1 \geq \cdots \sigma_r > 0$. Then

$$A^T A = V\Sigma^T \Sigma V^T = VDV^T, \quad (88)$$

where $D = \Sigma^T \Sigma = \text{diag}(\sigma_1^2, \ldots, \sigma_r^2, 0, \ldots, 0) = \text{diag}(\mu_1, \ldots, \mu_m) \in \mathbb{R}^{m \times m}$, with μ_i, $1 \leq i \leq m$, being the eigenvalues of $A^T A \in \mathbb{R}^{m \times m}$.

Theorem 15 *Assume that $A^T\mathbf{b} = 0$. A global solution of the minimization problem (86) is an eigenvector $\mathbf{x}^*$ of $A^T A$ corresponding to the smallest eigenvalue μ_m, appropriately normalized so that $||\mathbf{x}^*||_{\mathbb{R}^m} = \alpha$. This solution is unique if and only if μ_m is single. Otherwise, there are infinitely many solutions, each one being a normalized eigenvector in the eigenspace of μ_m.*

Theorem 16 *Assume that $A^T\mathbf{b} \neq 0$. Let $\mathbf{c} = U^T\mathbf{b}$. Let γ^* be the unique real number in the interval $(-\sigma_r^2, \infty)$ such that*

$$s(\gamma^*) = \sum_{i=1}^{r} \frac{\sigma_i^2 c_i^2}{(\sigma_i^2 + \gamma^*)^2} = \alpha^2. \quad (89)$$

(I) The vector

$$\mathbf{x}(\gamma^*) = (A^T A + \gamma^* I_m)^{-1} A^T \mathbf{b}, \quad (90)$$

is the unique global solution of the minimization problem (86) in one of the following cases:

1. $\text{rank}(A) = m$.
2. $\text{rank}(A) = r < m$ *and* $\gamma^* > 0$.
3. $\text{rank}(A) = r < m$, $\gamma^* < 0$, *and* $\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} > \alpha^2$.

(II) In the remaining case, namely $\text{rank}(A) = r < m$, $\gamma^* \leq 0$, *and* $\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} \leq \alpha^2$, *then the global solution of the minimization problem (86) is given by*

$$\mathbf{x}(0) = V\mathbf{y}, \tag{91}$$

where $y_i = \frac{c_i}{\sigma_i}$, $1 \leq i \leq r$, *with* y_i, $r+1 \leq i \leq m$, *taking arbitrary values such that*

$$\sum_{i=r+1}^{m} y_i^2 = \alpha^2 - \sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2}. \tag{92}$$

This solution is unique if and only if $\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} = \alpha^2$. *If* $\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} < \alpha^2$, *then there are infinitely many solutions.*

Remark 17 *To solve equation (89), the so-called secular equation, we note that the function* $s(\gamma) = \sum_{i=1}^{r} \frac{\sigma_i^2 c_i^2}{(\sigma_i^2+\gamma)^2}$ *is monotonically decreasing on* $(-\sigma_r^2, \infty)$ *and thus (89) can be solved via a bisection procedure.*

Remark 18 *We have presented here the solution to the problem of optimizing* C *in the least square case. The optimization of* C *in the SVM case is substantially different and will be treated in a future work.*

7. Experiments

In this section, we present an extensive empirical analysis of the proposed methods on the challenging tasks of multiclass image classification and species recognition with attributes. We show that the proposed framework[2] is able to combine different types of views and modalities and that it is competitive with other state-of-the-art approaches that have been developed in the literature to solve these problems.

The following methods, which are instances of the presented theoretical framework, were implemented and tested: multi-view learning with least square loss function (MVL-LS), MVL-LS with the optimization of the combination operator (MVL-LS-optC), multi-view learning with binary SVM loss function in the one-vs-all setup (MVL-binSVM), and multi-view learning with multi-class SVM loss function (MVL-SVM).

Our experiments demonstrate that: 1) multi-view learning achieves significantly better performance compared to single-view learning (Section 7.4); 2) unlabeled data can be particularly helpful in improving performance when the number of labeled data is small

2. The code for our multi-view learning methods is available at `https://github.com/lorisbaz/Multiview-learning`.

(Section 7.4 and Section 7.5); 3) the choice and therefore the optimization of the combination operator C is important (Section 7.6); and 4) the proposed framework outperforms other state-of-the-art approaches even in the case when we use fewer views (Section 7.7).

In the following sections, we first describe the designs for the experiments: the construction of the kernels is described in Section 7.1, the used data sets and evaluation protocols in Section 7.2 and the selection/validation of the regularization parameters in Section 7.3. Afterward, Sections 7.4, 7.5, 7.6, and 7.7 report the analysis of the obtained results with comparisons to the literature.

7.1 Kernels

Assume that each input x has the form $x = (x^1, \ldots, x^m)$, where x^i represents the ith view. We set $G(x,t)$ to be the diagonal matrix of size $m \times m$, with

$$(G(x,t))_{i,i} = k^i(x^i, t^i), \quad \text{that is} \quad G(x,t) = \sum_{i=1}^{m} k^i(x^i, t^i)\mathbf{e}_i\mathbf{e}_i^T, \tag{93}$$

where k^i is a scalar-valued kernel defined on view i and $\mathbf{e}_i = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^m$ is the ith coordinate vector. The corresponding Gram matrices are related by

$$G[\mathbf{x}] = \sum_{i=1}^{m} k^i[\mathbf{x}] \otimes \mathbf{e}_i\mathbf{e}_i^T. \tag{94}$$

Note that for each pair (x,t), $G(x,t)$ is a diagonal matrix, but it is *not* separable, that is it cannot be expressed in the form $k(x,t)D$ for a scalar kernel k and a positive semi-definite matrix D, because the kernels k^i's are in general different.

To carry out multi-class classification with P classes, $P \geq 2$, using vector-valued least squares regression (Algorithm 1), we set $\mathcal{Y} = \mathbb{R}^P$, and $K(x,t) = G(x,t) \otimes R$, with $R = I_P$. For each y_i, $1 \leq i \leq l$, in the labeled training sample, we set $y_i = (-1, \ldots, 1, \ldots, -1)$, with 1 at the kth location if x_i is in the kth class. When using vector-valued multi-view SVM (Algorithm 2), we set S to be the simplex coding, $\mathcal{Y} = \mathbb{R}^{P-1}$, and $K(x,t) = G(x,t) \otimes R$, with $R = I_{P-1}$.

We remark that since the views are coupled by both the loss functions and the multi-view manifold regularization term M, even in the simplest scenario, that is fully supervised multi-view binary classification, Algorithm 1 with a diagonal $G(x,t)$ is not equivalent to solving m independent scalar-valued least square regression problems, and Algorithm 2 is not equivalent to solving m independent binary SVMs.

We used $R = I_{\mathcal{Y}}$ for the current experiments. For multi-label learning applications, one can set R to be the output graph Laplacian as done in (Minh and Sindhwani, 2011).

We empirically analyzed the optimization framework of the combination operator $\mathbf{c}$ in the least square setting, as theoretically presented in Section 6. For the experiments with the SVM loss, we set the weight vector $\mathbf{c}$ to be the uniform combination $\mathbf{c} = \frac{1}{m}(1, \ldots, 1)^T \in \mathbb{R}^m$, leaving its optimization, which is substantially different from the least square case, to future work.

In all experiments, the kernel matrices are used as the weight matrices for the graph Laplacians, unless stated otherwise. This is not necessarily the best choice in practice but

we did not use additional information to compute more informative Laplacians at this stage to have a fair comparison with other state of the art techniques.

7.2 Data sets and Evaluation Protocols

Three data sets were used in our experiments to test the proposed methods, namely, the Oxford flower species (Nilsback and Zisserman, 2006), Caltech-101 (Fei-Fei et al., 2006), and Caltech-UCSD Birds-200-2011 (Wah et al., 2011). For these data sets, the views are the different features extracted from the input examples as detailed below.

The Flower species data set (Nilsback and Zisserman, 2006) consists of 1360 images of 17 flower species segmented out from the background. We used the following 7 extracted features in order to fairly compare with (Gehler and Nowozin, 2009): HOG, HSV histogram, boundary SIFT, foreground SIFT, and three features derived from color, shape and texture vocabularies. The features, the respective χ^2 kernel matrices and the training/testing splits[3] are taken from (Nilsback and Zisserman, 2006) and (Nilsback and Zisserman, 2008). The total training set provided by (Nilsback and Zisserman, 2006) consists of 680 labeled images (40 images per class). In our experiments, we varied the number of labeled data $l_c = \{1, 5, 10, 20, 40\}$ images per category and used 85 unlabeled images ($u_c = 5$ per class) taken from the validation set in (Nilsback and Zisserman, 2006) when explicitly stated. The testing set consists of 20 images per class as in (Nilsback and Zisserman, 2006).

The Caltech-101 data set (Fei-Fei et al., 2006) is a well-known data set for object recognition that contains 102 classes of objects and about 40 to 800 images per category. We used the features and χ^2 kernel matrices[4] provided in (Vedaldi et al., 2009), consisting of 4 descriptors extracted using a spatial pyramid of three levels, namely PHOW gray and color, geometric blur, and self-similarity. In our experiments, we selected only the lower level of the pyramid, resulting in 4 kernel matrices as in (Minh et al., 2013). We report results using all 102 classes (background class included) averaged over three splits as provided in (Vedaldi et al., 2009). In our tests, we varied the number of labeled data ($l_c = \{5, 10, 15\}$ images per category) in the supervised setup. The test set contained 15 images per class for all of the experiments.

The Caltech-UCSD Birds-200-2011 data set (Wah et al., 2011) is used for bird categorization and contains both images and manually-annotated attributes (two modalities)[5]. This data set is particularly challenging because it contains 200 very similar bird species (classes) for a total of 11,788 annotated images split between training and test sets. We used the same evaluation protocol and kernel matrices of (Minh et al., 2013). Different training sets were created by randomly selecting 5 times a set of $l_c = \{1, 5, 10, 15\}$ images for each class. All testing samples were used to evaluate the method. We used 5 unlabeled images per class in the semi-supervised setup. The descriptors consist of two views: PHOW gray (Vedaldi et al., 2009) from images and the 312-dimensional binary vector representing attributes provided in (Wah et al., 2011). The χ^2 and Gaussian kernels were used for the appearance and attribute features, respectively.

3. The complete data is available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html`.
4. The complete data is available at `http://www.robots.ox.ac.uk/~vgg/software/MKL/`.
5. The data set is available at `http://www.vision.caltech.edu/visipedia/CUB-200-2011.html`.

7.3 Regularization Parameters

Let us specify the parameters we used in the experiments. Each method has three regularization parameters, namely, γ_A for the standard RKHS regularization, and γ_B and γ_W for the multi-view manifold regularization. The only data set for which it was possible to perform independent cross-validation is the Flower species data set which has a separate validation set from the training set. For the other data sets, cross-validation was omitted in order to have the same number of training examples and therefore to have a fair comparison with the other state-of-the-art methods.

Cross-validation on the flower species data set was performed using the following set of parameters: $\gamma_A = \{10^{-5}, 10^{-6}, 10^{-7}\}$, $\gamma_B = \{10^{-6}, 10^{-8}, 10^{-9}\}$ and $\gamma_W = \{10^{-6}, 10^{-8}, 10^{-9}\}$. Cross-validation was run on the experiment with $l_c = 10$ labeled data per category. The parameters found during validation were left the same for all the other experiments $l_c = \{1, 5, 20, 40\}$ to have a fair comparison.

The parameters that performed the best on the validation set for the Flower species data set are reported in Table 1, column a. We also report the parameters chosen for Caltech-101 and the Caltech-UCSD Birds-200-2011 data set in Table 1 columns b and c, respectively. Notice that the parameters vary across the different implementations of the proposed framework and especially across the different data sets, as might be expected.

	(a) Flower species			(b) Caltech-101			(c) Caltech Birds		
Method	γ_A	γ_B	γ_W	γ_A	γ_B	γ_W	γ_A	γ_B	γ_W
MVL-LS	10^{-7}	10^{-9}	10^{-8}	10^{-5}	10^{-6}	10^{-6}	10^{-5}	10^{-6}	10^{-6}
MVL-binSVM	10^{-7}	10^{-8}	10^{-9}	10^{-5}	10^{-6}	10^{-6}	10^{-5}	10^{-6}	0
MVL-SVM	10^{-6}	10^{-8}	10^{-8}	10^{-6}	10^{-8}	10^{-8}	10^{-5}	10^{-6}	0

Table 1: Parameters for Flower species, Caltech-101 and Caltech-UCSD Birds-200-2011 data sets.

7.4 Single-view Vs. Multi-view

The purpose of the experimental analysis in this section is to demonstrate that multi-view learning significantly outperforms single-view learning.

First, we analyzed the contributions of each of the between-view and within-view regularization terms, given by (58) and (64), respectively, in the optimization problem (9). To this end, we tested multi-view learning with the least squares loss function on Caltech-101. A subset of 10 images for each class were randomly selected, with half used as labeled data $l_c = 5$ and the other half as unlabeled data $u_c = 5$ (see Table 2, last column). We also tested the proposed method in the one-shot learning setup, where the number of labeled images is one per class $l_c = 1$ (see Table 2, third column). The testing set consisted of 15 images per category. For this test, we selected the features at the bottom of each pyramid, because they give the best performance in practice. We can see from Table 2 that both the between-view and within-view regularization terms contribute to increase the recognition rate, *e.g.* with $l_c = 1$ the improvement is 2.35%. As one would expect, the improvement

resulting from the use of unlabeled data is bigger when there are more unlabeled data than labeled data, which can be seen by comparing the third and forth columns.

γ_B	γ_W	Accuracy $l_c = 1, u_c = 5$	Accuracy $l_c = u_c = 5$
0	0	30.59%	63.68%
0	10^{-6}	31.81%	63.97%
10^{-6}	0	32.44%	64.18%
10^{-6}	10^{-6}	**32.94**%	**64.2**%

Table 2: Results of MVL-LS on Caltech-101 using PHOW color and gray L2, SSIM L2 and GB. The training set consists of 1 or 5 labeled data l_c and 5 unlabeled data per class u_c, and 15 images per class are left for testing.

To demonstrate that multi-view learning is able to combine features properly, we report in Table 3 the performance in terms of average accuracy of each feature independently and of the proposed methods with all 10 views combined (last three rows). The improvement with respect to the view that gives the best results (PHOW gray L2) is 4.77% for the case with $l_c = 1$ (second column) and 5.62% for the case with $l_c = 5$ (last column). It is also worth noticing that all the proposed methods outperform the best single view (PHOW gray L2). Moreover, it is important to point out that the best views for each feature correspond to the L2 level. We show in Section 7.6 that the optimization of the combination operator leads to very similar findings.

Feature	Accuracy $l_c = 1, u_c = 5$	Accuracy $l_c = u_c = 5$
PHOW color L0	13.66%	33.14%
PHOW color L1	17.1%	42.03%
PHOW color L2	18.71%	45.86%
PHOW gray L0	20.31%	45.38%
PHOW gray L1	24.53%	54.86%
PHOW gray L2	25.64%	56.75%
SSIM L0	15.27%	35.27%
SSIM L1	20.83%	45.12%
SSIM L2	22.64%	48.47%
GB	25.01%	44.49%
MVL-LS	**30.41**%	61.46%
MVL-binSVM	30.20%	**62.37**%
MVL-SVM	27.23%	60.04%

Table 3: Results on Caltech-101 using each feature in the single-view learning framework and all 10 features in the multi-view learning framework (last three rows).

To further demonstrate the performance of multi-view learning, we run a similar experiment on the Caltech-UCSD Birds-200-2011 data set, with the results shown in Table 4. We compare the results obtained by the single views (PHOW and attributes) with the proposed

	$l_c = 1$	$l_c = 5$	$l_c = 10$	$l_c = 15$
PHOW	2.75%	5.51%	8.08%	9.92%
Attributes	13.53%	30.99%	38.96%	43.79%
MVL-LS	14.31%	33.25%	41.98%	46.74%
MVL-binSVM	**14.57**%	**33.50**%	**42.24**%	**46.88**%
MVL-SVM	14.15%	31.54%	39.30%	43.86%

Table 4: Results on the Caltech-UCSD Birds-200-2011 data set in the semi-supervised setup.

	$l_c = 1$	$l_c = 5$	$l_c = 10$	$l_c = 15$
MKL	N/A	42.1% (1.2%)	55.1% (0.7%)	62.3% (0.8%)
LP-B	N/A	46.5% (0.9%)	59.7% (0.7%)	66.7% (0.6%)
LP-β	N/A	54.2% (0.6%)	65.0% (0.9%)	70.4% (0.7%)
MVL-LS	*31.2*% (1.1%)	64.0% (1.0%)	71.0% (0.3%)	73.3% (1.3%)
MVL-binSVM	31.0% (1.3%)	*64.1*% (0.7%)	**71.4**% (0.3%)	**74.1**% (0.9%)
MVL-SVM	30.6% (1.0%)	63.6% (0.4%)	70.6% (0.2%)	*73.5*% (1.0%)
MVL-binSVM (semi-sup. $u_c = 5$)	**32.4**% (1.2%)	**64.4**% (0.4%)	*71.4*% (0.2%)	N/A

Table 5: Results on Caltech-101 when increasing the number of labeled data and comparisons with other state of the art methods reported by (Gehler and Nowozin, 2009). Best score in bold, second best score in italic.

multi-view learning methods (last three rows) when increasing the number of labeled data per class $l_c = \{1, 5, 10, 15\}$. In all the cases shown in the table, we obtain better results using the proposed multi-view learning framework compared with single-view learning.

7.5 Increasing the Label Set Size

In this section, we analyze the behavior of the proposed methods when increasing the size of the set of labeled data, in both supervised and semi-supervised settings.

In Table 5, we reported the results in terms of accuracy and its standard deviation (between brackets) on the Caltech-101 data set comparing with other state of the art methods. The first three rows report the results of the methods tested by (Gehler and Nowozin, 2009). The forth, fifth and sixth rows show the statistics of the proposed methods in the supervised setup. We also reported the results of the best methods among the proposed ones in the semi-supervised setup (with 5 unlabeled data for each class).

First, the results demonstrate that the proposed methods improve significantly when increasing the size of the labeled set. This fact can be observed also for the Caltech-UCSD Birds-200-2011 experiment in Table 4. More interestingly, when the number of labeled data

	$l_c = 1$	$l_c = 5$	$l_c = 10$	$l_c = 20$	$l_c = 40$
MVL-LS	39.41% (1.06%)	*65.78%* (3.68%)	74.41% (1.28%)	81.76% (3.28%)	**86.37%** (1.80%)
MVL-binSVM	39.71% (1.06%)	64.80% (4.42%)	*74.41%* (0.29%)	81.08% (3.09%)	86.08% (2.21%)
MVL-SVM	39.31% (1.62%)	65.29% (4.04%)	74.41% (1.28%)	81.67% (2.78%)	86.08% (1.80%)
MVL-LS (semi-sup.)	**41.86%** (2.50%)	**66.08%** (3.45%)	**75.00%** (1.06%)	**82.35%** (2.70%)	85.78% (2.78%)
MVL-binSVM (semi-sup.)	*40.59%* (2.35%)	65.49% (4.58%)	74.22% (0.68%)	81.57% (2.67%)	85.49% (0.74%)
MVL-SVM (semi-sup.)	34.80% (1.11%)	65.49% (4.17%)	74.41% (0.49%)	*81.78%* (2.61%)	*86.08%* (1.51%)

Table 6: Results on the Flower data set (17 classes) when increasing the number of training images per class. Best score in bold, second best score in italic.

is 5 per class (third column), our methods strongly improve the best result of (Gehler and Nowozin, 2009) by at least 9.4 percentage points. Similar observations can be made by examining the results obtained by Bucak et al. (2014) for $l_c = 10$ (Table 4 in their paper): our best result in Table 5 (71.4%) outperforms their best result (60.3%) by 11.1 percentage points. Moreover, one can see that the improvement when using unlabeled data (last row) is bigger when there are many more of them compared with labeled data, as expected (see the columns with 1 and 5 labeled images per class). When the number of labeled data increases, the proposed methods in the supervised setup can give comparable or better results (see the column with 10 labeled images per class). A similar behavior is shown in Table 6, when dealing the problem of species recognition with the Flower data set. The best improvement we obtained in the semi-supervised setup is with 1 labeled data per category. This finding suggests that the unlabeled data provide additional information about the distribution in the input space when there are few labeled examples. On the other hand, when there are sufficient labeled data to represent well the distribution in the input space, the unlabeled data will not provide an improvement of the results.

7.6 Optimizing the Combination Operator

In the previous experiments, the combination weight vector $\mathbf{c}$ was uniform, meaning that each view (*i.e.* kernel) has the same importance during classification. However, in practice it often happens that some views are more useful and informative than others. We observed this in our experiments, where different choices of the weights $\mathbf{c}$ gave rise to different classification accuracies. In particular, we empirically found for the Flower data set using MVL-LS that $\mathbf{c} = (0.1431, 0.1078, 0.1452, 0.1976, 0.0991, 0.1816, 0.1255)^T$ yields an accuracy of 87.75%, the state-of-the-art result for that data set. This suggests that there exists at least one better choice for $\mathbf{c}$.

In this section, we carry out an empirical analysis of the strategy presented in Section 6 which performs optimization to obtain the optimal weight vector $\mathbf{c}$. We call this method MVL-LS-optC. The analysis was performed on the Caltech-101 data set and the Flower data set. For the experiment using the Caltech-101 data set, we created a validation set by selecting 5 examples for each class from the training set. For the experiment using the Flower data set, the validation set was already provided (see Section 7.2 for detail). The

validation set is used to determine the best value of $\mathbf{c}$ found over all the iterations using different initializations. We carried out the iterative optimization procedure 20 times, each time with a different random unit vector as the initialization vector for $\mathbf{c}$, and reported the run with the best performance over the validation set.

The results of MVL-LS-optC for the Caltech-101 data set and the Flower data set are reported in Tables 7, 8 and 9. We empirically set $\alpha = 2$ and $\alpha = 1$ in the optimization problem (86) for the Caltech-101 data set and the Flower data set, respectively. MVL-LS-optC is compared with MVL-LS which uses uniform weights. We analyze in the next section how MVL-LS-optC compares with MVL-binSVM, MVL-SVM, and the state of the art.

We first discuss the results on the Caltech-101 data set using all 10 kernels. Table 7 shows that there is a significant improvement from 0.4% to 2.5% with respect to the results with uniform weights for the Caltech-101 data set. The best $\mathbf{c}$ found during training in the case of $l_c = 10$ was $\mathbf{c}^* = (0.1898, 0.6475, -0.7975, 0.3044, 0.1125, -0.4617, -0.1531, 0.1210, 1.2634, 0.9778)^T$. Note that the c_i's can assume negative values (as is the case here) and as we show in Section 8.1, the contribution of the ith view is determined by the square weight c_i^2. This experiment confirms our findings in Section 7.4: the best 4 views are PHOW color L2, PHOW gray L2, SSIM L2 and GB, which are the c_3, c_6, c_9 and c_{10} components of $\mathbf{c}$, respectively.

We now focus on the top 4 views and apply again the optimization method to see if there is still a margin of improvement. We expect to obtain better results with respect to 10 views because the 4-dimensional optimization should in practice be easier than the 10-dimensional one, given that the size of the search space is smaller. Table 8 shows the results with the top 4 kernels. We observe that there is an improvement with respect to MVL-LS that varies from 0.3% to 1.1%. We can also notice that there is not a significant improvement of the results when using more iteration (25 vs. 50 iterations). We again inspected the learned combination weights and discovered that in average they are very close to the uniform distribution, *i.e.* $\mathbf{c}^* = (-0.4965, -0.5019, -0.4935, -0.5073)^T$. This is mainly because we pre-selected the best set of 4 kernels accordingly to the previous 10-kernel experiment.

We finally used the best $\mathbf{c}$ learned in the case of $l_c = 10$ to do an experiment[6] with $l_c = 15$ on the Caltech-101. MVL-LS-optC obtains an accuracy of 73.85%, outperforming MVL-LS (uniform), which has an accuracy of 73.33% (see Table 10).

For the Flower data set, Table 9 shows consistent results with the previous experiment. MVL-LS-optC outperforms MVL-LS (uniform weights) in terms of accuracy with an improvement ranging from 0.98% to 4.22%. To have a deeper understanding about which views are more important, we analyzed the combination weights of the best result in Table 9 (last row, last column). The result of the optimization procedure is $\mathbf{c}^* = (-0.3648, -0.2366, 0.3721, 0.5486, -0.4108, 0.3468, 0.2627)^T$ which suggests that the best accuracy is obtained by exploiting the complementarity between shape-based features (c_3 and c_4) and color-based features (c_5) relevant for flower recognition[7].

6. We did not run the optimization of $\mathbf{c}$ for $l_c = 15$ because there is no validation set available for this case.
7. In our experiments, we used the following order: c_1 = HOG, c_2 = HSV, c_3 = boundary SIFT, c_4 = foreground SIFT, c_5 = color bag-of-features, c_6 = texture bag-of-features, c_7 = shape bag-of-features.

	$l_c = 1$	$l_c = 5$	$l_c = 10$
MVL-LS (uniform)	28.4% (1.8%)	61.4% (1.1%)	68.1% (0.3%)
MVL-LS-optC (25 it.)	**28.8**% (1.7%)	**63.1**% (0.1%)	**70.6**% (0.5%)

Table 7: Results using the procedure to optimize the combination operator on Caltech-101 considering all 10 kernels. Best score in bold.

	$l_c = 1$	$l_c = 5$	$l_c = 10$
MVL-LS (uniform)	31.2% (1.1%)	64.0% (1.0%)	71.0% (0.3%)
MVL-LS-optC (25 it.)	**32.1**% (1.5%)	*64.5*% (0.9%)	**71.3**% (0.4%)
MVL-LS-optC (50 it.)	*32.1*% (2.3%)	**64.7**% (1.1%)	*71.3*% (0.5%)

Table 8: Results using the procedure to optimize the combination operator on Caltech-101 using the top 4 kernels. Best score in bold, second best score in italic.

The proposed optimization procedure is powerful, with clear improvements in classification accuracies over the uniform weight approach. However, it comes with a price during the training phase. Firstly, it is an iterative procedure, and therefore it is more computationally expensive with respect to the original MVL-LS formulation. In particular, it is N_C times more expensive than MVL-LS, where N_C is the number of iterations. Secondly, since the joint optimization of $(\mathbf{c}, f_{\mathbf{z},\gamma})$ is non-convex, even though we are guaranteed to obtain the global minimum for $\mathbf{c}$ *during each single iteration*, the final $\mathbf{c}$ is *not* guaranteed to be the global minimum of the joint optimization problem itself.

7.7 Comparing with the State of the Art

In this section, we show how the proposed methods compare with other state-of-the-art approaches for each recognition problem.

In Table 10, we reported the best results we obtained for the task of object recognition using Caltech-101 in the supervised setup. Observe that all the proposed methods outperform the other techniques, even though they use much less information: 4 kernels versus e.g. 39 kernels in (Gehler and Nowozin, 2009).

In particular, we obtained the best result with the binary version of MVL in the one-vs-all setup. This is not surprising since the one-vs-all approach has often been shown to be very competitive in many computer vision tasks compared to proper multi-class formulations. The second best result is obtained by MVL-LS-optC since it uses an additional optimization step (of $\mathbf{c}$) with respect to the other methods. The optimization of $\mathbf{c}$ for MVL-binSVM and MVL-SVM is substantially different from the least square case and will be treated in a future work.

In Table 11, we reported the best results obtained for the task of species recognition using the Flower data set in the supervised setup. The proposed methods are compared

	$l_c = 1$	$l_c = 5$	$l_c = 10$	$l_c = 20$	$l_c = 40$
MVL-LS (uniform)	39.41% (1.06%)	65.78% (3.68%)	74.41% (1.28%)	*81.76*% (3.28%)	86.37% (1.80%)
MVL-LS-optC (25 it.)	*43.14*% (3.38%)	*68.53*% (2.90%)	*75.00*% (0.29%)	81.47% (2.06%)	*87.25*% (1.51%)
MVL-LS-optC (50 it.)	**43.63**% (3.25%)	**68.63**% (2.86%)	**75.39**% (0.90%)	**82.45**% (3.51%)	**87.35**% (1.35%)

Table 9: Results using the procedure to optimize the combination operator on the Flower data set. Best score in bold, second best score in italic.

Method	# of Kernels	Accuracy
(Yang et al., 2009)	≥ 10	73.2%
(Christoudias et al., 2009)	4	73.00%
LP-β (Gehler and Nowozin, 2009)	39	70.40%
MKL (Vedaldi et al., 2009)	10	71.10%
MVL-LS	4	73.33%
MVL-LS-optC	4	*73.85*%
MVL-binSVM	4	**74.05**%
MVL-SVM	4	73.55%

Table 10: Comparison with state-of-the-art methods on the Caltech-101 data set using PHOW color and gray L2, SSIM L2 and GB in the supervised setup (15 labeled images per class). Best score in bold, second best score in italic.

with MKL, LP-B and LP-β by (Gehler and Nowozin, 2009) as well as the more recent results of MK-SVM Shogun, MK-SVM OBSCURE and MK-FDA from (Yan et al., 2012). For this data set, our best result is obtained by the MVL-LS-optC method outperforming also the recent method MK-FDA from (Yan et al., 2012). We note also, that even with the uniform weight vector (MVL-LS), our methods outperform MK-FDA on Caltech-101, which uses 10 kernels, see Figures 6 and 9 in (Yan et al., 2012).

Method	Accuracy
MKL (SILP)	85.2% (1.5%)
MKL (Simple)	85.2% (1.5%)
LP-B	85.4% (2.4%)
LP-β	85.5% (3.0%)
MK-SVM Shogun	86.0% (2.4%)
MK-SVM OBSCURE	85.6% (0.0%)
MK-FDA	*87.2*% (1.6%)
MVL-LS	86.4% (1.8%)
MVL-LS-optC	**87.35**% (1.3%)
MVL-binSVM	86.1% (2.2%)
MVL-SVM	86.1% (1.8%)

Table 11: Results on the Flower data set comparing the proposed method with other state-of-the-art techniques in the supervised setup. The first four rows are from (Gehler and Nowozin, 2009) while rows 5-7 are methods presented by (Yan et al., 2012).

8. Further Theoretical Analysis

There are two purposes in this brief section, which shows the close connection between our framework and standard approaches in multi-kernel learning and multi-task learning. Firstly, we show that in the supervised setting, our framework is a form of multi-kernel learning, but with a crucial difference compared to typical multi-kernel learning methods, namely the combination weight vector is *not constrained* to be non-negative. Secondly, we also point out explicitly that several common scenarios in multi-task learning are special cases of our general formulation.

8.1 Connection with Multiple Kernel Learning

In this section, we briefly explore the connection between our multi-view learning framework and multiple kernel learning, see e.g. (Bach et al., 2004). We show that in the purely supervised setting, when $\gamma_I = 0$, $u = 0$, that is without unlabeled data and without between-view regularization, for $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$, $K(x,t) = G(x,t) \otimes I_{\mathcal{Y}}$, $G(x,t) = \sum_{i=1}^m k^i(x,t)\mathbf{e}_i\mathbf{e}_i^T$, we obtain supervised learning (vector-valued least square regression and SVM) with the combined kernel $\sum_{i=1}^m c_i^2 k^i(x,t) I_{\mathcal{Y}}$, where k^i is a scalar-valued kernel corresponding to view i. In particular, for $\mathcal{Y} = \mathbb{R}$, we obtain scalar-values least square regression and binary SVM with the combined kernel $\sum_{i=1}^m c_i^2 k^i(x,t)$. Specifically, we have the following results.

Corollary 19 *Consider the special case $\gamma_I = 0$, $u = 0$. The system of linear equations (13) in Theorem 4 has solution*

$$\mathbf{a} = (I_l \otimes C^*)\left[(I_l \otimes C)K[\mathbf{x}](I_l \otimes C^*) + l\gamma_A I_{\mathcal{Y}^l}\right]^{-1}\mathbf{y}. \tag{95}$$

For $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$, $K(x,t) = G(x,t) \otimes I_{\mathcal{Y}}$, and $G(x,t) = \sum_{i=1}^m k^i(x,t)\mathbf{e}_i\mathbf{e}_i^T$, for any $v \in \mathcal{X}$,

$$Cf_{\mathbf{z},\gamma}(v) = \left\{\sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}]\left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}] + l\gamma_A I_l\right)^{-1} \otimes I_{\mathcal{Y}}\right\}\mathbf{y}. \tag{96}$$

In particular, if $\mathcal{Y} = \mathbb{R}$, then

$$Cf_{\mathbf{z},\gamma}(v) = \left\{\sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}]\left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}] + l\gamma_A I_l\right)^{-1}\right\}\mathbf{y}. \tag{97}$$

This is precisely the solution of scalar-valued regularized least square regression with the combined kernel $\sum_{i=1}^m c_i^2 k^i(x,t)$.

Corollary 20 *Consider the special case $\gamma_I = 0$, $u = 0$. Then in Theorem 7,*

$$Q[\mathbf{x},\mathbf{y},C] = \frac{1}{\gamma_A}\mathrm{diag}(S^*_{\hat{\mathbf{y}}})(I_l \otimes C)K[\mathbf{x}](I_l \otimes C^*)\mathrm{diag}(S_{\hat{\mathbf{y}}}).$$

For $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$, $K(x,t) = G(x,t) \otimes I_{\mathcal{Y}}$, $G(x,t) = \sum_{i=1}^m k^i(x,t)\mathbf{e}_i\mathbf{e}_i^T$,

$$Q[\mathbf{x},\mathbf{y},C] = \frac{1}{\gamma_A}\mathrm{diag}(S^*_{\hat{\mathbf{y}}})\left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}] \otimes I_{\mathcal{Y}}\right)\mathrm{diag}(S_{\hat{\mathbf{y}}}),$$

and for any $v \in \mathcal{X}$,

$$Cf_{\mathbf{z},\gamma}(v) = -\frac{1}{2\gamma_A}\left(\sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}] \otimes I_{\mathcal{Y}}\right)\mathrm{diag}(S_{\hat{\mathbf{y}}})\mathrm{vec}(\beta^{\mathrm{opt}}).$$

In the binary case, $\mathcal{Y} = \mathbb{R}$, so that

$$Q[\mathbf{x},\mathbf{y},C] = \frac{1}{\gamma_A}\mathrm{diag}(\mathbf{y})\left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}]\right)\mathrm{diag}(\mathbf{y}),$$

$$Cf_{\mathbf{z},\gamma}(v) = -\frac{1}{2\gamma_A}\left(\sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}]\right)\mathrm{diag}(\mathbf{y})\beta^{\mathrm{opt}}.$$

This is precisely the solution of binary SVM with the combined kernel $\sum_{i=1}^m c_i^2 k^i(x,t)$.

Remark 21 *In the sum $\sum_{i=1}^m c_i^2 k^i(x,t)$, the coefficients c_i's are automatically non-negative. This is in accordance with the fact that our formulation makes no mathematical constraint on the coefficients c_i's in the sum $\sum_{i=1}^m c_i f^i(x)$. This is one difference between our approach and the typical multiple kernel learning setting (Bach et al., 2004), where one considers a sum of the form $\sum_{i=1}^m d_i k^i(x,t)$, where the d_i's must be non-negative to guarantee the positive definiteness of the combined kernel.*

8.2 Connection with Multi-task Learning

In this section, we briefly explore the connection between our learning formulation and multi-task learning, see e.g. (Evgeniou et al., 2005) . Let n be the number of tasks, $n \in \mathbb{N}$.

Consider the case where the tasks have the same input space. Let $\mathcal{T}$ be a separable Hilbert space. Let $G : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{T})$ be an operator-valued positive definite kernel, which induces an RKHS of functions with values in the Hilbert space $\mathcal{T}$. Consider the kernel $K(x,t)$ of the form

$$K(x,t) = R \otimes G(x,t), \tag{98}$$

where R is a symmetric, positive semidefinite matrix of size $n \times n$. The kernel $K(x,t)$ induces an RKHS of functions with values in the Hilbert space $\mathcal{T}^n$. Each function $f \in \mathcal{H}_K$ has the form $f(x) = (f^1(x), \ldots, f^n(x))$, with $f^k \in \mathcal{H}_G$, where $f^k(x)$ represents the output corresponding to the kth task.

In the simplest scenario, $\mathcal{W} = \mathcal{Y} = \mathcal{T}^n$, $C = I$, and the minimization problem (9) thus gives us a vector-valued semi-supervised multi-task learning formulation.

The tasks f^k's are related by the following, which is a generalization of (Evgeniou et al., 2005) (see their formulas (19), (20), (23)) to the nonlinear setting.

Lemma 22 *Let K be defined by (98), where R is strictly positive definite. For $f = (f^1, \ldots, f^n) \in \mathcal{H}_K$, with $f^k \in \mathcal{H}_G$, we have*

$$||f||^2_{\mathcal{H}_K} = \sum_{k,l=1}^n R^{-1}_{kl}\langle f^k, f^l\rangle_{\mathcal{H}_G}. \tag{99}$$

In particular, for

$$R = I_n + \frac{1-\lambda}{n\lambda}\mathbf{1}_n\mathbf{1}_n^T, \quad 0 < \lambda \leq 1, \tag{100}$$

we have

$$||f||^2_{\mathcal{H}_K} = \lambda \sum_{k=1}^{n} ||f^k||^2_{\mathcal{H}_G} + (1-\lambda)\sum_{k=1}^{n} ||f^k - \frac{1}{n}\sum_{l=1}^{n} f^l||^2_{\mathcal{H}_G}. \tag{101}$$

Consider the case when the tasks have different input spaces, such as in the approach to multi-view learning (Kadri et al., 2013), in which each view corresponds to one task and the tasks all share the same output label. Then we have m tasks for m views and we define

$$K(x,t) = G(x,t) \otimes R,$$

as in Section 5, where $G : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{m\times m}$ is a matrix-valued positive definite kernel, $R \in \mathcal{L}(\mathcal{T})$ is a symmetric, positive operator, so that each task has output in the Hilbert space $\mathcal{T}$. We obtain the formulation of (Kadri et al., 2013) if we set $\mathcal{T} = \mathbb{R}$, so that $R = 1$, duplicate each label $y_i \in \mathbb{R}$ into a vector $(y_i, \ldots, y_i) \in \mathbb{R}^m$, and set $G(x,t)$ to be their covariance-based kernel, with $\gamma_I = 0$, $u = 0$.

We have thus shown how two different scenarios in multi-task learning fall within the scope of our learning formulation. A more in-depth study of our framework in connection with multi-task learning is left to future work.

9. Discussion, Conclusion, and Future Work

We have presented a general learning framework in vector-valued RKHS which encompasses and generalizes many kernel-based learning algorithms in the literature. In particular, we generalize

- the Vector-valued Manifold Regularization framework of (Minh and Sindhwani, 2011), and thus also the vector-valued Regularized Least Square regression formulation of (Micchelli and Pontil, 2005), which are single-view and formulated with the least square loss, to the multi-view setting, formulated with both the least square and multi-class SVM loss functions;
- the Simplex Cone SVM of (Mroueh et al., 2012) , which is supervised, to the multi-view and semi-supervised settings, together with a more general loss function;
- the Laplacian SVM of (Belkin et al., 2006), which is binary and single-view, to the multi-class and multi-view settings.

The generality of the framework and the competitive numerical results we have obtained so far demonstrate that this is a promising venue for further research exploration. Some potential directions for our future work include

- a principled optimization framework for the weight vector $\mathbf{c}$ in the SVM setting, as well as the study of more general forms of the combination operator C;
- numerical experiments with different forms of the matrix-valued kernel K;

- theoretical and empirical analysis for the SVM under different coding schemes other than the simplex coding;
- theoretical analysis of our formulation, in particular when the numbers of labeled and unlabeled data points go to infinity;
- further connections between our framework and Multi-task learning;
- exploration of our framework in combination with feature learning methods, particularly those coming from deep learning;
- further analysis to optimize the framework for large-scale classification problems.

Apart from the numerical experiments on object recognition reported in this paper, practical applications for our learning framework so far include person re-identification in computer vision (Figueira et al., 2013) and user recognition and verification in Skype chats (Roffo et al., 2013). As we further develop and refine the current formulation, we expect to apply it to other applications in computer vision, image processing, and bioinformatics.

Appendices.

The Appendices contain three sections. First, in Appendix A, we give the proofs for all the main mathematical results in the paper. Second, in Appendix B, we provide a natural generalization of our framework to the case the point evaluation operator $f(x)$ is replaced by a general bounded linear operator. Last, in Appendix C, we provide an exact description of Algorithm 1 with the Gaussian or similar kernels in the degenerate case, when the kernel width $\sigma \to \infty$.

Appendix A. Proofs of Main Results

Notation: The definition of $\mathbf{f}$ as given by

$$\mathbf{f} = (f(x_1), \ldots, f(x_{u+l})) \in \mathcal{W}^{u+l}, \tag{102}$$

is adopted because it is also applicable when $\mathcal{W}$ is an infinite-dimensional Hilbert space. For $\mathcal{W} = \mathbb{R}^m$,

$$\mathbf{f} = (f^1(x_1), \ldots, f^m(x_1), \ldots, f^1(x_{u+l}), \ldots, f^m(x_{u+l})).$$

This is different from (Rosenberg et al., 2009), where

$$\mathbf{f} = (f^1(x_1), \ldots, f^1(x_{u+l}), \ldots, f^m(x_1), \ldots, f^m(x_{u+l})).$$

This means that our matrix M is necessarily a permutation of the matrix M in (Rosenberg et al., 2009) when they give rise to the same semi-norm.

A.1 Proof of the Representer Theorem

Since $f(x) = K_x^* f$, the minimization problem (9) is

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l} V(y_i, CK_{x_i}^* f) + \gamma_A ||f||_{\mathcal{H}_K}^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}. \tag{103}$$

Consider the operator $E_{C,\mathbf{x}} : \mathcal{H}_K \to \mathcal{Y}^l$, defined by

$$E_{C,\mathbf{x}} f = (CK_{x_1}^* f, \ldots, CK_{x_l}^* f), \tag{104}$$

with $CK_{x_i}^* : \mathcal{H}_K \to \mathcal{Y}$ and $K_{x_i} C^* : \mathcal{Y} \to \mathcal{H}_K$. For $\mathbf{b} = (b_1, \ldots, b_l) \in \mathcal{Y}^l$, we have

$$\langle \mathbf{b}, E_{C,\mathbf{x}} f\rangle_{\mathcal{Y}^l} = \sum_{i=1}^{l} \langle b_i, CK_{x_i}^* f\rangle_{\mathcal{Y}} = \sum_{i=1}^{l} \langle K_{x_i} C^* b_i, f\rangle_{\mathcal{H}_K}. \tag{105}$$

The adjoint operator $E_{C,\mathbf{x}}^* : \mathcal{Y}^l \to \mathcal{H}_K$ is thus

$$E_{C,\mathbf{x}}^* : (b_1, \ldots, b_l) \to \sum_{i=1}^{l} K_{x_i} C^* b_i. \tag{106}$$

The operator $E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} : \mathcal{H}_K \to \mathcal{H}_K$ is then

$$E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} f \to \sum_{i=1}^{l} K_{x_i} C^* C K_{x_i}^* f, \tag{107}$$

with $C^*C : \mathcal{W} \to \mathcal{W}$.

Proof of Theorem 2. Denote the right handside of (9) by $I_l(f)$. Then $I_l(f)$ is coercive and strictly convex in f, and thus has a unique minimizer. Let $\mathcal{H}_{K,\mathbf{x}} = \{\sum_{i=1}^{u+l} K_{x_i} w_i : \mathbf{w} \in \mathcal{W}^{u+l}\}$. For $f \in \mathcal{H}_{K,\mathbf{x}}^{\perp}$, the operator $E_{C,\mathbf{x}}$ satisfies

$$\langle \mathbf{b}, E_{C,\mathbf{x}} f\rangle_{\mathcal{Y}^l} = \langle f, \sum_{i=1}^{l} K_{x_i} C^* b_i\rangle_{\mathcal{H}_K} = 0,$$

for all $\mathbf{b} \in \mathcal{Y}^l$, since $C^* b_i \in \mathcal{W}$. Thus

$$E_{C,\mathbf{x}} f = (CK_{x_1}^* f, \ldots, CK_{x_l}^* f) = 0.$$

Similarly, by the reproducing property, the sampling operator $S_{\mathbf{x}}$ satisfies

$$\langle S_{\mathbf{x}} f, \mathbf{w}\rangle_{\mathcal{W}^{u+l}} \quad = \quad \langle f, \sum_{i=1}^{u+l} K_{x_i} w_i\rangle_{\mathcal{H}_K} = 0,$$

for all $\mathbf{w} \in \mathcal{W}^{u+l}$. Thus

$$\mathbf{f} = S_{\mathbf{x}} f = (f(x_1), \ldots, f(x_{u+l})) = 0.$$

For an arbitrary $f \in \mathcal{H}_K$, consider the orthogonal decomposition $f = f_0 + f_1$, with $f_0 \in \mathcal{H}_{K,\mathbf{x}}$, $f_1 \in \mathcal{H}_{K,\mathbf{x}}^{\perp}$. Then, because $||f_0 + f_1||_{\mathcal{H}_K}^2 = ||f_0||_{\mathcal{H}_K}^2 + ||f_1||_{\mathcal{H}_K}^2$, the result just obtained shows that

$$I_l(f) = I_l(f_0 + f_1) \geq I_l(f_0)$$

with equality if and only if $||f_1||_{\mathcal{H}_K} = 0$, that is $f_1 = 0$. Thus the minimizer of (9) must lie in $\mathcal{H}_{K,\mathbf{x}}$. ■

A.2 Proofs for the Least Square Case

We have for the least square case:

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f\in\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l} ||y_i - CK_{x_i}^* f||_{\mathcal{Y}}^2 + \gamma_A ||f||_{\mathcal{H}_K}^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}. \tag{108}$$

With the operator $E_{C,\mathbf{x}}$, (108) is transformed into the minimization problem

$$f_{\mathbf{z},\gamma} = \text{argmin}_{f\in\mathcal{H}_K} \frac{1}{l}||E_{C,\mathbf{x}} f - \mathbf{y}||_{\mathcal{Y}^l}^2 + \gamma_A ||f||_{\mathcal{H}_K}^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}. \tag{109}$$

Proof of Theorem 3. By the Representer Theorem, (10) has a unique solution. Differentiating (109) and setting the derivative to zero gives

$$(E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} + l\gamma_A I + l\gamma_I S_{\mathbf{x},u+l}^* M S_{\mathbf{x},u+l}) f_{\mathbf{z},\gamma} = E_{C,\mathbf{x}}^* \mathbf{y}.$$

By definition of the operators $E_{C,\mathbf{x}}$ and $S_{\mathbf{x}}$, this is

$$\sum_{i=1}^{l} K_{x_i} C^* C K_{x_i}^* f_{\mathbf{z},\gamma} + l\gamma_A f_{\mathbf{z},\gamma} + l\gamma_I \sum_{i=1}^{u+l} K_{x_i} (M\mathbf{f}_{\mathbf{z},\gamma})_i = \sum_{i=1}^{l} K_{x_i} C^* y_i,$$

which we rewrite as

$$f_{\mathbf{z},\gamma} = -\frac{\gamma_I}{\gamma_A}\sum_{i=1}^{u+l} K_{x_i} (M\mathbf{f}_{\mathbf{z},\gamma})_i + \sum_{i=1}^{l} K_{x_i} \frac{C^* y_i - C^* C K_{x_i}^* f_{\mathbf{z},\gamma}}{l\gamma_A}.$$

This shows that there are vectors a_i's in $\mathcal{W}$ such that

$$f_{\mathbf{z},\gamma} = \sum_{i=1}^{u+l} K_{x_i} a_i.$$

We have $f_{\mathbf{z},\gamma}(x_i) = \sum_{j=1}^{u+l} K(x_i, x_j) a_j$, and

$$(M\mathbf{f}_{\mathbf{z},\gamma})_i = \sum_{k=1}^{u+l} M_{ik} \sum_{j=1}^{u+l} K(x_k, x_j) a_j = \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j.$$

Also $K^*_{x_i} f_{\mathbf{z},\gamma} = f_{\mathbf{z},\gamma}(x_i) = \sum_{j=1}^{u+l} K(x_i, x_j) a_j$. Thus for $1 \leq i \leq l$:

$$a_i = -\frac{\gamma_I}{\gamma_A} \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + \frac{C^* y_i - C^* C(\sum_{j=1}^{u+l} K(x_i, x_j) a_j)}{l \gamma_A},$$

which gives the formula

$$l\gamma_I \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + C^* C(\sum_{j=1}^{u+l} K(x_i, x_j) a_j) + l\gamma_A a_i = C^* y_i.$$

Similarly, for $l+1 \leq i \leq u+l$,

$$a_i = -\frac{\gamma_I}{\gamma_A} \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j,$$

which is equivalent to

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + \gamma_A a_i = 0.$$

This completes the proof. ■

Proof (first proof) of Theorem 4. This is straightforward to obtain from Theorem 3 using the operator-valued matrix formulation described in the main paper. ■

In the following, we give a second proof of Theorem 4, which is based entirely on operator-theoretic notations. The proof technique should be of interest in its own right.

Proof (second proof) of Theorem 4. By the Representer Theorem, (10) has a unique solution. Differentiating (109) and setting the derivative to zero gives

$$(E^*_{C,\mathbf{x}} E_{C,\mathbf{x}} + l\gamma_A I + l\gamma_I S^*_{\mathbf{x},u+l} M S_{\mathbf{x},u+l}) f_{\mathbf{z},\gamma} = E^*_{C,\mathbf{x}} \mathbf{y}. \tag{110}$$

For $\gamma_A > 0, \gamma_I \geq 0$, the operator

$$E^*_{C,\mathbf{x}} E_{C,\mathbf{x}} + l\gamma_A I + l\gamma_I S^*_{\mathbf{x},u+l} M S_{\mathbf{x},u+l} \tag{111}$$

is clearly symmetric and strictly positive, so that the unique solution $f_{\mathbf{z},\gamma}$ is given by

$$f_{\mathbf{z},\gamma} = (E^*_{C,\mathbf{x}} E_{C,\mathbf{x}} + l\gamma_A I + l\gamma_I S^*_{\mathbf{x},u+l} M S_{\mathbf{x},u+l})^{-1} E^*_{C,\mathbf{x}} \mathbf{y}.$$

Recall the definitions of the operators $S_{\mathbf{x},u+l} : \mathcal{H}_K \to \mathcal{W}^{u+l}$ and $S^*_{\mathbf{x},u+l} : \mathcal{W}^{u+l} \to \mathcal{H}_K$:

$$S_{\mathbf{x},u+l} f = (K^*_{x_i} f)_{i=1}^{u+l}, \quad S^*_{\mathbf{x},u+l} \mathbf{b} = \sum_{i=1}^{u+l} K_{x_i} b_i$$

with the operator $S_{\mathbf{x},u+l} S^*_{\mathbf{x},u+l} : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ given by

$$S_{\mathbf{x},u+l} S^*_{\mathbf{x},u+l} \mathbf{b} = \left(K^*_{x_i} \left(\sum_{j=1}^{u+l} K_{x_j} b_j \right) \right)_{i=1}^{u+l} = \left(\sum_{j=1}^{u+l} K(x_i, x_j) b_j \right)_{i=1}^{u+l} = K[\mathbf{x}]\mathbf{b},$$

so that

$$S_{\mathbf{x},u+l} S^*_{\mathbf{x},u+l} = K[\mathbf{x}].$$

The operator $E_{C,\mathbf{x}} : \mathcal{H}_K \to \mathcal{Y}^l$ is

$$E_{C,\mathbf{x}} f = (C K^*_{x_i} f)_{i=1}^l = (I^T_{(u+l)\times l} \otimes C) S_{\mathbf{x},u+l} f,$$

so that

$$E_{C,\mathbf{x}} = (I^T_{(u+l)\times l} \otimes C) S_{\mathbf{x},u+l}, \tag{112}$$

and the operator $E^*_{C,\mathbf{x}} : \mathcal{Y}^l \to \mathcal{H}_K$ is

$$E^*_{C,\mathbf{x}} = S^*_{\mathbf{x},u+l} (I_{(u+l)\times l} \otimes C^*). \tag{113}$$

As operators, $I_{(u+l)\times l} \otimes C^* : \mathcal{Y}^l \to \mathcal{W}^{u+l}$ and $I^T_{(u+l)\times l} \otimes C : \mathcal{W}^{u+l} \to \mathcal{Y}^l$. The operator $E^*_{C,\mathbf{x}} E_{C,\mathbf{x}} : \mathcal{H}_K \to \mathcal{H}_K$ is then given by

$$E^*_{C,\mathbf{x}} E_{C,\mathbf{x}} = S^*_{\mathbf{x},u+l} (J^{u+l}_l \otimes C^* C) S_{\mathbf{x},u+l} : \mathcal{H}_K \to \mathcal{H}_K, \tag{114}$$

where $J^{u+l}_l = I_{(u+l)\times l} I^T_{(u+l)\times l}$ is the $(u+l) \times (u+l)$ diagonal matrix, with the first l entries on the main diagonal being 1, and the rest 0. As an operator, $J^{u+l}_l \otimes C^* C : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$. The operator $E_{C,\mathbf{x}} E^*_{C,\mathbf{x}} : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ is given by

$$E_{C,\mathbf{x}} E^*_{C,\mathbf{x}} = (I^T_{(u+l)\times l} \otimes C) S_{\mathbf{x},u+l} S^*_{\mathbf{x},u+l} (I_{(u+l)\times l} \otimes C^*) = (I^T_{(u+l)\times l} \otimes C) K[\mathbf{x}] (I_{(u+l)\times l} \otimes C^*). \tag{115}$$

Equation (110) becomes

$$\left[S^*_{\mathbf{x},u+l} (J^{u+l}_l \otimes C^* C + l\gamma_I M) S_{\mathbf{x},u+l} + l\gamma_A I \right] f_{\mathbf{z},\gamma} = S^*_{\mathbf{x},u+l} (I_{(u+l)\times l} \otimes C^*) \mathbf{y}, \tag{116}$$

which gives

$$f_{\mathbf{z},\gamma} = S^*_{\mathbf{x},u+l} \left[\frac{-(J^{u+l}_l \otimes C^* C + l\gamma_I M) S_{\mathbf{x},u+l} f_{\mathbf{z},\gamma} + (I_{(u+l)\times l} \otimes C^*) \mathbf{y}}{l\gamma_A} \right] \tag{117}$$

$$= S^*_{\mathbf{x},u+l} \mathbf{a}, \tag{118}$$

where $\mathbf{a} = (a_i)_{i=1}^{u+l} \in \mathcal{W}^{u+l}$ is

$$\mathbf{a} = \frac{-(J^{u+l}_l \otimes C^* C + l\gamma_I M) S_{\mathbf{x},u+l} f_{\mathbf{z},\gamma} + (I_{(u+l)\times l} \otimes C^*) \mathbf{y}}{l\gamma_A}. \tag{119}$$

By definition of $S_{\mathbf{x},u+l}$ and $S^*_{\mathbf{x},u+l}$,

$$S_{\mathbf{x},u+l} f_{\mathbf{z},\gamma} = S_{\mathbf{x},u+l} S^*_{\mathbf{x},u+l} \mathbf{a} = K[\mathbf{x}]\mathbf{a}.$$

Substituting this into equation (119), we obtain

$$\mathbf{a} = \frac{-(J_l^{u+l} \otimes C^*C + l\gamma_I M)K[\mathbf{x}]\mathbf{a} + (I_{(u+l)\times l} \otimes C^*)\mathbf{y}}{l\gamma_A},$$

or equivalently

$$[(J_l^{u+l} \otimes C^*C + l\gamma_I M)K[\mathbf{x}] + l\gamma_A I_{\mathcal{W}^{u+l}}]\mathbf{a} = (I_{(u+l)\times l} \otimes C^*)\mathbf{y}. \tag{120}$$

The operator-valued matrix on the left hand side,

$$(J_l^{u+l} \otimes C^*C + l\gamma_I M)K[\mathbf{x}] + l\gamma_A I_{\mathcal{W}^{u+l}} : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l},$$

is invertible by Lemma 25, with a bounded inverse. Thus the above system of linear equations always has a unique solution

$$\mathbf{a} = [(J_l^{u+l} \otimes C^*C + l\gamma_I M)K[\mathbf{x}] + l\gamma_A I_{\mathcal{W}^{u+l}}]^{-1}(I_{(u+l)\times l} \otimes C^*)\mathbf{y}.$$

This completes the proof of the theorem. ■

Remark 23 (Uniqueness of a) . *While the solution $\mathbf{f}_{\mathbf{z},\gamma} = \sum_{i=1}^{u+l} K_{x_i} a_i$ in Theorem 2 is always unique, the expansion coefficient vectors a_i's for $f_{\mathbf{z},\gamma}$ need not be unique. In fact, we have*

$$||f_{\mathbf{z},\gamma}||^2_{\mathcal{H}_K} = \langle S^*_{\mathbf{x},u+l}\mathbf{a}, S^*_{\mathbf{x},u+l}\mathbf{a}\rangle_{\mathcal{H}_K} = \langle \mathbf{a}, S_{\mathbf{x},u+l}S^*_{\mathbf{x},u+l}\mathbf{a}\rangle_{\mathcal{W}^{u+l}} = \langle \mathbf{a}, K[\mathbf{x}]\mathbf{a}\rangle_{\mathcal{W}^{u+l}}.$$

By the reproducing property,

$$f_{\mathbf{z},\gamma} = 0 \Longleftrightarrow ||f_{\mathbf{z},\gamma}||_{\mathcal{H}_K} = 0 \Longleftrightarrow \mathbf{a} = 0 \text{ or } \mathbf{a} \in \text{null}(K[\mathbf{x}]).$$

Thus $\mathbf{a}$ is unique if and only if $K[\mathbf{x}]$ is invertible, or equivalently, $K[\mathbf{x}]$ is of full rank. For us, our choice for $\mathbf{a}$ is always the unique solution of the system of linear equations (13) in Theorem 4 (see also Remark 24 below).

Remark 24 *The coefficient matrix of the system of linear equations (13) in Theorem 4 has the form $(\gamma I + AB)$, where A, B are two symmetric, positive operators on a Hilbert space $\mathcal{H}$. We show in Lemma 25 that the operator $(\gamma I + AB)$ is always invertible for $\gamma > 0$ and that the inverse operator $(\gamma I + AB)^{-1}$ is bounded, so that the system (13) is always guaranteed a unique solution, as we claim in Theorem 4. Furthermore, the eigenvalues of AB, when they exist, are always non-negative, as we show in Lemma 26. This gives another proof of the invertibility of $(\gamma I + AB)$ when $\mathcal{H}$ is finite-dimensional, in Corollary 27. This invertibility is also necessary for the proofs of Theorems 6 and 7 in the SVM case.*

Lemma 25 *Let $\mathcal{H}$ be a Hilbert space and $A, B : \mathcal{H} \to \mathcal{H}$ be two bounded, symmetric, positive operators. Then the operator $(\gamma I + AB)$ is invertible for any $\gamma > 0$ and the inverse $(\gamma I + AB)^{-1}$ is bounded.*

Proof Let $T = \gamma I + AB$. We need to show that T is 1-to-1 and onto. First, to show that T is 1-to-1, suppose that

$$Tx = \gamma x + ABx = 0.$$

This implies that

$$BTx = \gamma Bx + BABx = 0 \Longrightarrow \langle x, BTx\rangle = \gamma\langle x, Bx\rangle + \langle x, BABx\rangle = 0.$$

By the symmetry and positivity of A and B, this is equivalent to

$$\gamma||B^{1/2}x||^2 + ||A^{1/2}Bx||^2 = 0.$$

This is possible if and only if $x = 0$ or $B^{1/2}x = 0$. If $B^{1/2}x = 0$, $x \neq 0$, then $Tx = \gamma x \neq 0$. Thus

$$Tx = 0 \Longleftrightarrow x = 0.$$

This shows that T is 1-to-1. Similar arguments show that its adjoint $T^* = \gamma I + BA$ is 1-to-1, so that

$$\overline{\text{range}(T)} = (\ker(T^*))^\perp = \{0\}^\perp = \mathcal{H}.$$

It thus remains for us to show that range(T) is closed. Let $\{y_n\}_{n\in\mathbb{N}}$ be a Cauchy sequence in range(T), with $y_n = Tx_n$ for $x_n \in \mathcal{H}$. Then we have

$$By_n = \gamma Bx_n + BABx_n \Longrightarrow \langle x_n, By_n\rangle = \gamma\langle x_n, Bx_n\rangle + \langle x_n, BABx_n\rangle.$$

By the symmetry and positivity of A and B, this is

$$\langle x_n, By_n\rangle = \gamma||B^{1/2}x_n||^2 + ||A^{1/2}Bx_n||^2.$$

It follows that

$$\gamma||B^{1/2}x_n||^2 \leq \langle x_n, By_n\rangle \leq ||B^{1/2}x_n||\ ||B^{1/2}y_n||,$$

so that

$$\gamma||B^{1/2}x_n|| \leq ||B^{1/2}y_n|| \leq ||B^{1/2}||\ ||y_n||.$$

From the assumption $y_n = Tx_n = \gamma x_n + ABx_n$, we have

$$\gamma x_n = y_n - ABx_n.$$

This implies that

$$\gamma||x_n|| \leq ||y_n|| + ||AB^{1/2}||\ ||B^{1/2}x_n|| \leq ||y_n|| + \frac{||AB^{1/2}||\ ||B^{1/2}||}{\gamma}\ ||y_n||,$$

which simplifies to

$$||x_n|| \leq \frac{1}{\gamma}\left(1 + \frac{||AB^{1/2}||\ ||B^{1/2}||}{\gamma}\right)||y_n||.$$

Since T is linear, $y_{n+1} - y_n = T(x_{n+1} - x_n)$ and thus

$$||x_{n+1} - x_n|| \leq \frac{1}{\gamma}\left(1 + \frac{||AB^{1/2}||\,||B^{1/2}||}{\gamma}\right)||y_{n+1} - y_n||.$$

Thus if $\{y_n\}_{n\in\mathbb{N}}$ is a Cauchy sequence in $\mathcal{H}$, then $\{x_n\}_{n\in\mathbb{N}}$ is also a Cauchy sequence in $\mathcal{H}$. Let $x_0 = \lim_{n\to\infty} x_n$ and $y_0 = Tx_0$, then clearly $\lim_{n\to\infty} y_n = y_0$. This shows that range$(T)$ is closed, as we claimed, so that range(T) = range(T) $= \mathcal{H}$, showing that T is onto. This completes the proof. ■

Lemma 26 *Let $\mathcal{H}$ be a Hilbert space. Let A and B be two symmetric, positive, bounded operators in $\mathcal{L}(\mathcal{H})$. Then all eigenvalues of the product operator AB, if they exist, are real and non-negative.*

Proof Let λ be an eigenvalue of AB, corresponding to eigenvector x. Then

$$ABx = \lambda x \Longrightarrow BABx = \lambda Bx \Longrightarrow \langle x, BABx\rangle = \lambda\langle x, Bx\rangle.$$

Since both A and B are symmetric, positive, the operator BAB is symmetric, positive, and therefore $\langle x, BABx\rangle \geq 0$. Since B is symmetric, positive, we have $\langle x, Bx\rangle \geq 0$, with $\langle x, Bx\rangle = ||B^{1/2}x||^2 = 0$ if and only if $x \in$ null$(B^{1/2})$.

If $x \in$ null$(B^{1/2})$, then $ABx = 0$, so that $\lambda = 0$.

If $x \notin$ null$(B^{1/2})$, then $\langle x, Bx\rangle > 0$, and

$$\lambda = \frac{\langle x, BABx\rangle}{\langle x, Bx\rangle} \geq 0.$$

Consequently, we always have $\lambda \geq 0$. ■

Corollary 27 *Let A and B be two symmetric positive semi-definite matrices. Then the matrix $(\gamma I + AB)$ is invertible for any $\gamma > 0$.*

Proof The eigenvalues of $(\gamma I + AB)$ have the form $\gamma + \lambda$, where λ is an eigenvalue of AB and satisfies $\lambda \geq 0$ by Lemma 26. Thus all eigenvalues of $(\gamma I + AB)$ are strictly positive, with magnitude at least γ. It follows that $\det(\gamma I + AB) > 0$ and therefore $(\gamma I + AB)$ is invertible. ■

Proof of Theorem 10. Recall some properties of the Kronecker tensor product:

$$(A \otimes B)(C \otimes D) = AC \otimes BD, \tag{121}$$

$$(A \otimes B)^T = A^T \otimes B^T, \tag{122}$$

and

$$\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B). \tag{123}$$

Thus the equation

$$AXB = C \tag{124}$$

is equivalent to

$$(B^T \otimes A)\text{vec}(X) = \text{vec}(C). \tag{125}$$

In our context,$\gamma_I M = \gamma_B M_B + \gamma_W M_W$, which is

$$\gamma_I M = \gamma_B I_{u+l} \otimes M_m \otimes I_{\mathcal{Y}} + \gamma_W L \otimes I_{\mathcal{Y}}.$$

Using the properties stated in 121 and 122, we have for $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$,

$$C^*C = (\mathbf{c} \otimes I_{\mathcal{Y}})(\mathbf{c}^T \otimes I_{\mathcal{Y}}) = (\mathbf{c}\mathbf{c}^T \otimes I_{\mathcal{Y}}). \tag{126}$$

So then

$$\mathbf{C}^*\mathbf{C} = (I_{u+l} \otimes \mathbf{c}\mathbf{c}^T \otimes I_{\mathcal{Y}}). \tag{127}$$

$$J_l^{\mathcal{W},u+l} = J_l^{u+l} \otimes I_m \otimes I_{\mathcal{Y}}. \tag{128}$$

It follows that

$$\mathbf{C}^*\mathbf{C}J_l^{\mathcal{W},u+l} = (J_l^{u+l} \otimes \mathbf{c}\mathbf{c}^T \otimes I_{\mathcal{Y}}). \tag{129}$$

Then with

$$K[\mathbf{x}] = G[\mathbf{x}] \otimes R,$$

we have

$$\mathbf{C}^*\mathbf{C}J_l^{\mathcal{W},u+l}K[\mathbf{x}] = (J_l^{u+l} \otimes \mathbf{c}\mathbf{c}^T)G[\mathbf{x}] \otimes R.$$
$$\gamma_I MK[\mathbf{x}] = (\gamma_B I_{u+l} \otimes M_m + \gamma_W L)G[\mathbf{x}] \otimes R.$$

Consider again now the system

$$(\mathbf{C}^*\mathbf{C}J_l^{\mathcal{W},u+l}K[\mathbf{x}] + l\gamma_I MK[\mathbf{x}] + l\gamma_A I)\mathbf{a} = \mathbf{C}^*\mathbf{y}.$$

The left hand side is

$$(B \otimes R + l\gamma_A I_{(u+l)m} \otimes I_{\mathcal{Y}})\text{vec}(A^T),$$

where $\mathbf{a} = \text{vec}(A^T)$, A is of size $(u+l)m \times \dim(\mathcal{Y})$, and

$$B = \left((J_l^{u+l} \otimes \mathbf{c}\mathbf{c}^T) + l\gamma_B(I_{u+l} \otimes M_m) + l\gamma_W L\right)G[\mathbf{x}].$$

Then we have the linear system

$$(B \otimes R + l\gamma_A I_{(u+l)m} \otimes I_{\mathcal{Y}})\text{vec}(A^T) = \text{vec}(Y_C^T),$$

which, by properties (124) and (125), is equivalent to

$$RA^TB^T + l\gamma_A A^T = Y_C^T \Longleftrightarrow BAR + l\gamma_A A = Y_C.$$

This completes the proof. ■

Remark 28 *The* vec *operator is implemented by the flattening operation* (:) *in MATLAB. To compute the matrix* Y_C^T*, note that by definition*

$$\text{vec}(Y_C^T) = \mathbf{C}^*\mathbf{y} = (I_{(u+l)\times l} \otimes C^*)\mathbf{y} = \text{vec}(C^* Y_l I_{(u+l)\times l}^T) = \text{vec}(C^* Y_{u+l}),$$

where Y_l *is the* $\dim(\mathcal{Y}) \times l$ *matrix, whose* i*th column is* y_i*,* $1 \leq i \leq l$*, that is*

$$Y_l = [y_1, \ldots, y_l], \quad \textit{with} \quad \mathbf{y} = \text{vec}(Y_l),$$

and Y_{u+l} *is the* $\dim(\mathcal{Y}) \times (u+l)$ *matrix with the* i*th column being* y_i*,* $1 \leq i \leq l$*, with the remaining* u *columns being zero, that is*

$$Y_{u+l} = [y_1, \ldots, y_l, 0, \ldots, 0] = [Y_l, 0, \ldots, 0] = Y_l I_{(u+l)\times l}^T.$$

Note that Y_C^T *and* $C^* Y_{u+l}$ *in general are not the same:* Y_C^T *is of size* $\dim(\mathcal{Y}) \times (u+l)m$*, whereas* $C^* Y_{u+l}$ *is of size* $\dim(\mathcal{Y})m \times (u+l)$*.*

Proof of Corollary 19 For $\gamma_I = 0$, $u = 0$, equation (110) becomes

$$(E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} + l\gamma_A I) f_{\mathbf{z},\gamma} = E_{C,\mathbf{x}}^* \mathbf{y},$$

which is equivalent to

$$f_{\mathbf{z},\gamma} = (E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} + l\gamma_A I_{\mathcal{H}_K})^{-1} E_{C,\mathbf{x}}^* \mathbf{y} = E_{C,\mathbf{x}}^* (E_{C,\mathbf{x}} E_{C,\mathbf{x}}^* + l\gamma_A I_{\mathcal{Y}^l})^{-1} \mathbf{y},$$

that is

$$f_{\mathbf{z},\gamma} = S_{\mathbf{x},l}^* (I_l \otimes C^*) \left[(I_l \otimes C) K[\mathbf{x}] (I_l \otimes C^*) + l\gamma_A I_{\mathcal{Y}^l} \right]^{-1} \mathbf{y}.$$

Thus in this case $f_{\mathbf{z},\gamma} = S_{\mathbf{x},l}^* \mathbf{a}$, where $\mathbf{a} = (a_i)_{i=1}^l$ is given by

$$\mathbf{a} = (I_l \otimes C^*) \left[(I_l \otimes C) K[\mathbf{x}] (I_l \otimes C^*) + l\gamma_A I_{\mathcal{Y}^l} \right]^{-1} \mathbf{y}.$$

In this expression, the operator $[(I_l \otimes C) K[\mathbf{x}] (I_l \otimes C^*) + l\gamma_A] : \mathcal{Y}^l \to \mathcal{Y}^l$ is clearly symmetric and strictly positive, hence is invertible. For $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$ and $K[\mathbf{x}] = G[\mathbf{x}] \otimes R$, we have

$$\mathbf{a} = (I_l \otimes \mathbf{c} \otimes I_{\mathcal{Y}}) \left[(I_l \otimes \mathbf{c}^T) G[\mathbf{x}] (I_l \otimes \mathbf{c}) \otimes R + l\gamma_A I_{\mathcal{Y}^l} \right]^{-1} \mathbf{y}.$$

With $R = I_{\mathcal{Y}}$, this becomes

$$\mathbf{a} = \{ (I_l \otimes \mathbf{c}) \left[(I_l \otimes \mathbf{c}^T) G[\mathbf{x}] (I_l \otimes \mathbf{c}) + l\gamma_A I_l \right]^{-1} \otimes I_{\mathcal{Y}} \} \mathbf{y}.$$

For any $v \in \mathcal{X}$,

$$f_{\mathbf{z},\gamma}(v) = K[v, \mathbf{x}]\mathbf{a} = \{ G[v, \mathbf{x}] (I_l \otimes \mathbf{c}) \left[(I_l \otimes \mathbf{c}^T) G[\mathbf{x}] (I_l \otimes \mathbf{c}) + l\gamma_A I_l \right]^{-1} \otimes I_{\mathcal{Y}} \} \mathbf{y}.$$

$$C f_{\mathbf{z},\gamma}(v) = \{ \mathbf{c}^T G[v, \mathbf{x}] (I_l \otimes \mathbf{c}) \left[(I_l \otimes \mathbf{c}^T) G[\mathbf{x}] (I_l \otimes \mathbf{c}) + l\gamma_A I_l \right]^{-1} \otimes I_{\mathcal{Y}} \} \mathbf{y}.$$

With $G[\mathbf{x}] = \sum_{i=1}^m k^i[\mathbf{x}] \otimes \mathbf{e}_i \mathbf{e}_i^T$, we have

$$(I_l \otimes \mathbf{c}^T) G[\mathbf{x}] (I_l \otimes \mathbf{c}) = (I_l \otimes \mathbf{c}^T) (\sum_{i=1}^m k^i[\mathbf{x}] \otimes \mathbf{e}_i \mathbf{e}_i^T) (I_l \otimes \mathbf{c}) = \sum_{i=1}^m c_i^2 k^i[\mathbf{x}],$$

$$\mathbf{c}^T G[v, \mathbf{x}] (I_l \otimes \mathbf{c}) = \mathbf{c}^T (\sum_{i=1}^m k^i[v, \mathbf{x}] \otimes \mathbf{e}_i \mathbf{e}_i^T) (I_l \otimes \mathbf{c}) = \sum_{i=1}^m c_i^2 k^i[v, \mathbf{x}].$$

With these, we obtain

$$C f_{\mathbf{z},\gamma}(v) = \left\{ \sum_{i=1}^m c_i^2 k^i[v, \mathbf{x}] \left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}] + l\gamma_A I_l \right)^{-1} \otimes I_{\mathcal{Y}} \right\} \mathbf{y}.$$

In particular, for $\mathcal{Y} = \mathbb{R}$, we obtain

$$C f_{\mathbf{z},\gamma}(v) = \left\{ \sum_{i=1}^m c_i^2 k^i[v, \mathbf{x}] \left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}] + l\gamma_A I_l \right)^{-1} \right\} \mathbf{y}.$$

This completes the proof. ■

Proof of Lemma 22 Consider the function $f \in \mathcal{H}_K$ of the form

$$f(x) = \sum_{i=1}^m K(x, x_i) a_i = \sum_{i=1}^m [R \otimes G(x, x_i)] a_i \in \mathcal{T}^n,$$

where $a_i \in \mathcal{T}^n$. Let A_i be the (potentially infinite) matrix of size $\dim(\mathcal{T}) \times n$ such that $a_i = \text{vec}(A_i)$. Then

$$f(x) = \sum_{i=1}^m [R \otimes G(x, x_i)] \text{vec}(A_i) = \sum_{i=1}^m \text{vec}(G(x, x_i) A_i R),$$

with norm

$$\begin{aligned} ||f||_{\mathcal{H}_K}^2 &= \sum_{i,j=1}^m \langle a_i, K(x_i, x_j) a_j \rangle_{\mathcal{T}^n} = \sum_{i,j=1}^m \langle a_i, (R \otimes G(x_i, x_j)) a_j \rangle_{\mathcal{T}^n} \\ &= \sum_{i,j=1}^m \langle \text{vec}(A_i), \text{vec}(G(x_i, x_j) A_j R) \rangle_{\mathcal{T}^n} = \sum_{i,j=1}^m \text{tr}(A_i^T G(x_i, x_j) A_j R). \end{aligned}$$

Each component f^k, $1 \leq k \leq n$, has the form

$$f^k(x) = \sum_{i=1}^m G(x, x_i) A_i R_{:,k} \in \mathcal{H}_G,$$

where $R_{:,k}$ is the kth column of R, with norm

$$||f^k||_{\mathcal{H}_G}^2 = \sum_{i,j=1}^m \langle A_i R_{:,k}, G(x_i, x_j) A_j R_{:,k} \rangle_{\mathcal{T}} = \sum_{i,j=1}^m R_{:,k}^T A_i^T G(x_i, x_j) A_j R_{:,k}.$$

For

$$f^l(x) = \sum_{i=1}^m G(x, x_i) A_i R_{:,l},$$

we have

$$\langle f^k, f^l\rangle_{\mathcal{H}_G} = \sum_{i,j=1}^{m} R^T_{:,k} A_i^T G(x_i, x_j) A_j R_{:,l}.$$

Let B be a symmetric, positive definite matrix of size $n \times n$. Consider the form

$$\begin{aligned}\sum_{k,l=1}^{n} B_{kl}\langle f^k, f^l\rangle_{\mathcal{H}_G} &= \sum_{k,l=1}^{n}\sum_{i,j=1}^{m} B_{kl} R^T_{:,k} A_i^T G(x_i, x_j) A_j R_{:,l} \\ &= \sum_{i,j=1}^{m}\sum_{k,l=1}^{n} B_{kl} R^T_{:,k} A_i^T G(x_i, x_j) A_j R_{:,l} = \sum_{i,j=1}^{m} \mathrm{tr}(BR^T A_i^T G(x_i, x_j) A_j R) \\ &= \sum_{i,j=1}^{m} \mathrm{tr}(BRA_i^T G(x_i, x_j) A_j R), \quad \text{since } R \text{ is symmetric.}\end{aligned}$$

It follows that for R strictly positive definite and $B = R^{-1}$, we have

$$||f||^2_{\mathcal{H}_K} = \sum_{k,l=1}^{n} B_{kl}\langle f^k, f^l\rangle_{\mathcal{H}_G}.$$

In particular, for $0 < \lambda \leq 1$ and

$$R = I_n + \frac{1-\lambda}{n\lambda}\mathbf{1}_n\mathbf{1}_n^T,$$

we have

$$B = R^{-1} = I_n - \frac{1-\lambda}{n}\mathbf{1}_n\mathbf{1}_n^T.$$

Then

$$\begin{aligned}||f||^2_{\mathcal{H}_K} &= \sum_{k,l=1}^{n} B_{kl}\langle f^k, f^l\rangle_{\mathcal{H}_G} = \sum_{k=1}^{n} ||f^k||^2_{\mathcal{H}_G} - \frac{1-\lambda}{n}\sum_{k,l=1}^{n}\langle f^k, f^l\rangle_{\mathcal{H}_G} \\ &= \lambda\sum_{k=1}^{n} ||f^k||^2_{\mathcal{H}_G} + (1-\lambda)\sum_{k=1}^{n}||f^k - \frac{1}{n}\sum_{l=1}^{n} f^l||^2_{\mathcal{H}_G}.\end{aligned}$$

This result then extends to all $f \in \mathcal{H}_K$ by a limiting argument. This completes the proof. ■

A.3 Proofs for the SVM case

Recall the optimization problem that we aim to solve

$$f_{\mathbf{z},\gamma} = \mathrm{argmin}_{f\in\mathcal{H}_K, \xi_{ki}\in\mathbb{R}} \frac{1}{l}\sum_{i=1}^{l}\sum_{k=1,k\neq y_i}^{P} \xi_{ki} + \gamma_A||f||^2_{\mathcal{H}_K} + \gamma_I\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}},$$

subject to the constraints

$$\begin{aligned}&\xi_{ki} \geq -\langle s_k, s_{y_i}\rangle_{\mathcal{Y}} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}}, \quad 1 \leq i \leq l, k \neq y_i, \\ &\xi_{ki} \geq 0, \quad 1 \leq i \leq l, k \neq y_i.\end{aligned}$$

Proof of Theorem 6 The Lagrangian is

$$\begin{aligned}L(f, \xi, \alpha, \beta) = &\frac{1}{l}\sum_{i=1}^{l}\sum_{k\neq y_i}\xi_{ki} + \gamma_A||f||^2_{\mathcal{H}_K} + \gamma_I\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}} \\ &- \sum_{i=1}^{l}\sum_{k\neq y_i}\alpha_{ki}\left(\xi_{ki} - [-\langle s_k, s_{y_i}\rangle_{\mathcal{Y}} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}}]\right) - \sum_{i=1}^{l}\sum_{k\neq y_i}\beta_{ki}\xi_{ki}, \quad (130)\end{aligned}$$

where

$$\alpha_{ki} \geq 0, \beta_{ki} \geq 0, \quad 1 \leq i \leq l, k \neq y_i. \quad (131)$$

By the reproducing property

$$\langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} = \langle C^* s_k, f(x_i)\rangle_{\mathcal{W}} = \langle f, K_{x_i}(C^* s_k)\rangle_{\mathcal{H}_K}. \quad (132)$$

Thus the Lagrangian is

$$\begin{aligned}L(f, \xi, \alpha, \beta) = &\frac{1}{l}\sum_{i=1}^{l}\sum_{k\neq y_i}\xi_{ki} + \gamma_A||f||^2_{\mathcal{H}_K} + \gamma_I\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}} \\ &- \sum_{i=1}^{l}\sum_{k\neq y_i}\alpha_{ki}\left(\xi_{ki} - [-\langle s_k, s_{y_i}\rangle_{\mathcal{Y}} + \langle f, K_{x_i}(C^* s_k)\rangle_{\mathcal{H}_K}]\right) - \sum_{i=1}^{l}\sum_{k\neq y_i}\beta_{ki}\xi_{ki}. \quad (133)\end{aligned}$$

Since

$$\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}} = \langle S_{\mathbf{x},u+l} f, MS_{\mathbf{x},u+l} f\rangle_{\mathcal{W}^{u+l}} = \langle f, S^*_{\mathbf{x},u+l} MS_{\mathbf{x},u+l} f\rangle_{\mathcal{H}_K}, \quad (134)$$

we have

$$\frac{\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}}{\partial f} = 2S^*_{\mathbf{x},u+l} MS_{\mathbf{x},u+l} f = 2\sum_{i=1}^{u+l} K_{x_i}(M\mathbf{f})_i. \quad (135)$$

Differentiating the Lagrangian with respect to ξ_{ki} and setting to zero, we obtain

$$\frac{\partial L}{\partial \xi_{ki}} = \frac{1}{l} - \alpha_{ki} - \beta_{ki} = 0 \Longleftrightarrow \alpha_{ki} + \beta_{ki} = \frac{1}{l}. \quad (136)$$

Differentiating the Lagrangian with respect to f, we obtain

$$\frac{\partial L}{\partial f} = 2\gamma_A f + 2\gamma_I S^*_{\mathbf{x},u+l} MS_{\mathbf{x},u+l} f + \sum_{i=1}^{l}\sum_{k\neq y_i}\alpha_{ki} K_{x_i}(C^* s_k). \quad (137)$$

Setting this derivative to zero, we obtain

$$f = -\frac{\gamma_I}{\gamma_A}\sum_{i=1}^{u+l} K_{x_i}(M\mathbf{f})_i - \frac{1}{2\gamma_A}\sum_{i=1}^{l}\sum_{k\neq y_i}\alpha_{ki} K_{x_i}(C^* s_k). \quad (138)$$

This means there are vectors $a_i \in \mathcal{W}$, $1 \leq i \leq u+l$, such that

$$f = \sum_{i=1}^{u+l} K_{x_i} a_i.$$

This gives

$$\mathbf{f}_k = f(x_k) = \sum_{j=1}^{u+l} K(x_k, x_j) a_j,$$

so that

$$(M\mathbf{f})_i = \sum_{k=1}^{u+l} M_{ik}\mathbf{f}_k = \sum_{k=1}^{u+l} M_{ik} \sum_{j=1}^{u+l} K(x_k, x_j) a_j = \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j. \tag{139}$$

For $1 \leq i \leq l$,

$$a_i = -\frac{\gamma_I}{\gamma_A} \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j - \frac{1}{2\gamma_A} \sum_{k \neq y_i} \alpha_{ki} (C^* s_k), \tag{140}$$

or equivalently,

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + \gamma_A a_i = -\frac{1}{2} \sum_{k \neq y_i} \alpha_{ki} (C^* s_k) = -\frac{1}{2} C^* S \alpha_i, \tag{141}$$

since $\alpha_{y_i, i} = 0$. For $l+1 \leq i \leq u+l$,

$$a_i = -\frac{\gamma_I}{\gamma_A} \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j, \tag{142}$$

or equivalently,

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik} K(x_k, x_j) a_j + \gamma_A a_i = 0. \tag{143}$$

In operator-valued matrix notation, (141) and (143) together can be expressed as

$$(\gamma_I M K[\mathbf{x}] + \gamma_A I)\mathbf{a} = -\frac{1}{2}(I_{(u+l)\times l} \otimes C^* S)\mathrm{vec}(\alpha). \tag{144}$$

By Lemma 25, the operator $(\gamma_I M K[\mathbf{x}] + \gamma_A I) : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ is invertible, with a bounded inverse, so that

$$\mathbf{a} = -\frac{1}{2}(\gamma_I M K[\mathbf{x}] + \gamma_A I)^{-1}(I_{(u+l)\times l} \otimes C^* S)\mathrm{vec}(\alpha). \tag{145}$$

With condition (136), the Lagrangian (133) simplifies to

$$\begin{aligned} L(f, \xi, \alpha, \beta) = {} & \gamma_A ||f||_K^2 + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} \\ & + \sum_{i=1}^{l} \sum_{k \neq y_i} \alpha_{ki} \left([-\langle s_k, s_{y_i} \rangle_{\mathcal{Y}} + \langle s_k, C f(x_i) \rangle_{\mathcal{Y}}] \right). \end{aligned} \tag{146}$$

From expression (137), we have

$$\frac{\partial L}{\partial f} = 0 \Longleftrightarrow \gamma_A f + \gamma_I S^*_{\mathbf{x}, u+l} M S_{\mathbf{x}, u+l} f = -\frac{1}{2} \sum_{i=1}^{l} \sum_{k \neq y_i} \alpha_{ki} K_{x_i}(C^* s_k). \tag{147}$$

Taking inner product with f on both sides, we get

$$\gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} = -\frac{1}{2} \sum_{i=1}^{l} \sum_{k \neq y_i} \alpha_{ki} \langle f, K_{x_i}(C^* s_k) \rangle_{\mathcal{H}_K}. \tag{148}$$

With $f = \sum_{j=1}^{u+l} K_{x_j} a_j$, we have

$$\langle f, K_{x_i}(C^* s_k) \rangle_{\mathcal{H}_K} = \sum_{j=1}^{u+l} \langle K(x_i, x_j) a_j, C^* s_k \rangle_{\mathcal{W}}, \tag{149}$$

so that

$$\begin{aligned} \sum_{k \neq y_i} \alpha_{ki} \langle f, K_{x_i}(C^* s_k) \rangle_{\mathcal{H}_K} &= \sum_{j=1}^{u+l} \langle K(x_i, x_j) a_j, \sum_{k \neq y_i} \alpha_{ki} C^* s_k \rangle_{\mathcal{W}} \\ &= \sum_{j=1}^{u+l} \langle K(x_i, x_j) a_j, C^* S \alpha_i \rangle_{\mathcal{W}}. \end{aligned} \tag{150}$$

Combining this with (148), we obtain

$$\begin{aligned} \gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} &= -\frac{1}{2} \sum_{i=1}^{l} \langle \sum_{j=1}^{u+l} K(x_i, x_j) a_j, C^* S \alpha_i \rangle_{\mathcal{W}} \\ &= -\frac{1}{2} \sum_{i=1}^{l} \langle S^* C \sum_{j=1}^{u+l} K(x_i, x_j) a_j, \alpha_i \rangle_{\mathbb{R}^P}. \end{aligned} \tag{151}$$

In operator-valued matrix notation, this is

$$\gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} = -\frac{1}{2} \mathrm{vec}(\alpha)^T (I^T_{(u+l)\times l} \otimes S^* C) K[\mathbf{x}]\mathbf{a}. \tag{152}$$

Substituting the expression for $\mathbf{a}$ in (145) into (152), we obtain

$$\begin{aligned} \gamma_A ||f||^2_{\mathcal{H}_K} + \gamma_I \langle \mathbf{f}, M\mathbf{f} \rangle_{\mathcal{W}^{u+l}} = {} & \frac{1}{4} \mathrm{vec}(\alpha)^T (I^T_{(u+l)\times l} \otimes S^* C) K[\mathbf{x}] \\ & \times (\gamma_I M K[\mathbf{x}] + \gamma_A I)^{-1} (I_{(u+l)\times l} \otimes C^* S) \mathrm{vec}(\alpha). \end{aligned} \tag{153}$$

Combining (146), (148), and (153), we obtain the final form of the Lagrangian

$$L(\alpha) = -\sum_{i=1}^{l} \sum_{k \neq y_i} \langle s_k, s_{y_i} \rangle_{\mathcal{Y}} \alpha_{ki} - \frac{1}{4} \mathrm{vec}(\alpha)^T Q[\mathbf{x}, C] \mathrm{vec}(\alpha), \tag{154}$$

where the matrix $Q[\mathbf{x}, C]$ is given by

$$Q[\mathbf{x}, C] = (I^T_{(u+l)\times l} \otimes S^* C) K[\mathbf{x}] (\gamma_I M K[\mathbf{x}] + \gamma_A I)^{-1} (I_{(u+l)\times l} \otimes C^* S). \tag{155}$$

We need to maximize the Lagrangian subject to the constraints

$$0 \leq \alpha_{ki} \leq \frac{1}{l}, \quad 1 \leq i \leq l, k \neq y_i. \tag{156}$$

Since $\alpha_{y_i,i} = 0$, these constraints can be written as

$$0 \leq \alpha_{ki} \leq \frac{1}{l}(1 - \delta_{k,y_i}), \quad 1 \leq i \leq l, 1 \leq k \leq P. \tag{157}$$

Equivalently, under the same constraints, we minimize

$$D(\alpha) = \frac{1}{4}\text{vec}(\alpha)^T Q[\mathbf{x}, C]\text{vec}(\alpha) + \sum_{i=1}^{l}\sum_{k=1}^{P}\langle s_k, s_{y_i}\rangle_{\mathcal{Y}}\alpha_{ki}. \tag{158}$$

When S is the simplex coding, we have $\langle s_k, s_{y_i}\rangle_{\mathcal{Y}} = -\frac{1}{P-1}$ for $k \neq y_i$, and $\alpha_{y_i,i} = 0$, so that

$$\sum_{i=1}^{l}\sum_{k=1}^{P}\langle s_k, s_{y_i}\rangle_{\mathcal{Y}}\alpha_{ki} = -\frac{1}{P-1}\sum_{i=1}^{l}\sum_{k=1}^{P}\alpha_{ki} = -\frac{1}{P-1}\mathbf{1}_{Pl}^T\text{vec}(\alpha).$$

This gives the last expression of the theorem.

Let us show that $Q[\mathbf{x}, C]$ is symmetric and positive semidefinite. To show that $Q[\mathbf{x}, C]$ is symmetric, it suffices to show that $K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1}$ is symmetric. We have

$$(\gamma_I K[\mathbf{x}]M + \gamma_A I)K[\mathbf{x}] = K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I),$$

which is equivalent to

$$K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1} = (\gamma_I K[\mathbf{x}]M + \gamma_A I)^{-1}K[\mathbf{x}] = (K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1})^T$$

by the symmetry of $K[\mathbf{x}]$ and M, showing that $K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1}$ is symmetric. The positive semidefiniteness of $Q[\mathbf{x}, C]$ simply follows from (153). This completes the proof of the theorem. ■

Proof of Theorem 7 Let $S_{\hat{y}_i}$ be the matrix obtained from S by removing the y_ith column and $\beta_i \in \mathbb{R}^{P-1}$ be the vector obtained from α_i by deleting the y_ith entry, which is equal to zero by assumption. As in the proof of Theorem 6, for $1 \leq i \leq l$,

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik}K(x_k, x_j)a_j + \gamma_A a_i = -\frac{1}{2}C^* S\alpha_i = -\frac{1}{2}C^* S_{\hat{y}_i}\beta_i. \tag{159}$$

For $l + 1 \leq i \leq u + l$,

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik}K(x_k, x_j)a_j + \gamma_A a_i = 0. \tag{160}$$

Let $\text{diag}(S_{\hat{\mathbf{y}}})$ be the $l \times l$ block diagonal matrix, with block (i, i) being $S_{\hat{y}_i}$. Let $\beta = (\beta_1, \ldots, \beta_l)$ be the $(P-1) \times l$ matrix with column i being β_i. In operator-valued matrix notation, (159) and (160) together can be expressed as

$$(\gamma_I MK[\mathbf{x}] + \gamma_A I)\mathbf{a} = -\frac{1}{2}(I_{(u+l)\times l} \otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta). \tag{161}$$

By Lemma 25, the operator $(\gamma_I MK[\mathbf{x}] + \gamma_A I) : \mathcal{W}^{u+l} \to \mathcal{W}^{u+l}$ is invertible, with a bounded inverse, so that

$$\mathbf{a} = -\frac{1}{2}(\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1}(I_{(u+l)\times l} \otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta). \tag{162}$$

As in the proof of Theorem 6,

$$\begin{aligned}\gamma_A\|f\|_{\mathcal{H}_K}^2 + \gamma_I\langle\mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}} &= -\frac{1}{2}\sum_{i=1}^{l}\langle\sum_{j=1}^{u+l}K(x_i, x_j)a_j, C^*S\alpha_i\rangle_{\mathcal{W}} \\ &= -\frac{1}{2}\sum_{i=1}^{l}\langle\sum_{j=1}^{u+l}K(x_i, x_j)a_j, C^*S_{\hat{y}_i}\beta_i\rangle_{\mathcal{W}} \\ &= -\frac{1}{2}\sum_{i=1}^{l}\langle S_{\hat{y}_i}^* C\sum_{j=1}^{u+l}K(x_i, x_j)a_j, \beta_i\rangle_{\mathbb{R}^{P-1}}.\end{aligned} \tag{163}$$

In operator-valued matrix notation, this is

$$\gamma_A\|f\|_{\mathcal{H}_K}^2 + \gamma_I\langle\mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}} = -\frac{1}{2}\text{vec}(\beta)^T\text{diag}(S_{\hat{\mathbf{y}}}^*)(I_{(u+l)\times l}^T \otimes C)K[\mathbf{x}]\mathbf{a}. \tag{164}$$

Substituting the expression for $\mathbf{a}$ in (162) into (164), we obtain

$$\begin{aligned}\gamma_A\|f\|_{\mathcal{H}_K}^2 + \gamma_I\langle\mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}} = &\frac{1}{4}\text{vec}(\beta)^T\text{diag}(S_{\hat{\mathbf{y}}}^*)(I_{(u+l)\times l}^T \otimes C)K[\mathbf{x}] \\ &\times (\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1}(I_{(u+l)\times l} \otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta).\end{aligned} \tag{165}$$

We now note that

$$\sum_{i=1}^{l}\sum_{k\neq y_i}\alpha_{ki}\langle s_k, s_{y_i}\rangle_{\mathcal{Y}} = \sum_{i=1}^{l}\langle s_{y_i}, S_{\hat{y}_i}\beta_i\rangle_{\mathcal{Y}}. \tag{166}$$

Combining (146), (166), (148), and (165), we obtain the final form of the Lagrangian

$$L(\beta) = -\sum_{i=1}^{l}\langle s_{y_i}, S_{\hat{y}_i}\beta_i\rangle_{\mathcal{Y}} - \frac{1}{4}\text{vec}(\beta)^T Q[\mathbf{x}, \mathbf{y}, C]\text{vec}(\beta), \tag{167}$$

where the matrix $Q[\mathbf{x}, \mathbf{y}, C]$ is given by

$$Q[\mathbf{x}, \mathbf{y}, C] = \text{diag}(S_{\hat{\mathbf{y}}}^*)(I_{(u+l)\times l}^T \otimes C)K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1}(I_{(u+l)\times l} \otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}}). \tag{168}$$

We need to maximize the Lagrangian subject to the constraints

$$0 \leq \beta_{ki} \leq \frac{1}{l}, \quad 1 \leq i \leq l, 1 \leq k \leq P - 1. \tag{169}$$

Equivalently, under the same constraints, we minimize

$$D(\beta) = \frac{1}{4}\text{vec}(\beta)^T Q[\mathbf{x}, \mathbf{y}, C]\text{vec}(\beta) + \sum_{i=1}^{l}\langle s_{y_i}, S_{\hat{y}_i}\beta_i\rangle_{\mathcal{Y}}. \tag{170}$$

If S is the simplex coding, then

$$\langle s_{y_i}, S_{\hat{y}_i}\beta_i\rangle_{\mathcal{Y}} = \langle S_{\hat{y}_i}^T s_{y_i}, \beta_i\rangle_{\mathcal{Y}} = -\frac{1}{P-1}\mathbf{1}_{P-1}^T\beta_i.$$

It follows then that

$$\sum_{i=1}^{l}\langle s_{y_i}, S_{\hat{y}_i}\beta_i\rangle_{\mathcal{Y}} = -\frac{1}{P-1}\mathbf{1}_{(P-1)l}^T \mathrm{vec}(\beta),$$

giving the last expression of the theorem. This completes the proof. ■

Lemma 29 *The matrix-valued kernel K is positive definite.*

Proof Let $d = \dim(\mathcal{Y})$. Consider an arbitrary set of points $\mathbf{x} = \{x_i\}_{i=1}^N$ in $\mathcal{X}$ and an arbitrary set of vectors $\{y_i\}_{i=1}^N$ in $\mathbb{R}^{md}$. We need to show that

$$\sum_{i,j=1}^{N}\langle y_i, K(x_i, x_j)y_j\rangle_{\mathbb{R}^{md}} = \mathbf{y}^T K[\mathbf{x}]\mathbf{y} \geq 0,$$

where $\mathbf{y} = (y_1, \ldots, y_N) \in \mathbb{R}^{mdN}$ as a column vector. This is equivalent to showing that the Gram matrix $K[\mathbf{x}]$ of size $mdN \times mdN$ is positive semi-definite for any set $\mathbf{x}$.

By assumption, G is positive definite, so that the Gram matrix $G[\mathbf{x}]$ of size $mN \times mN$ is positive semi-definite for any set $\mathbf{x}$. Since the Kronecker tensor product of two positive semi-definite matrices is positive semi-definite, the matrix

$$K[\mathbf{x}] = G[\mathbf{x}] \otimes R$$

is positive semi-definite for any set $\mathbf{x}$. This completes the proof. ■

To prove Theorem 11, we need the following result.

Lemma 30 *Let $N, n \in \mathbb{N}$ and $\gamma > 0$. Let U be an orthogonal matrix of size $n \times n$, with columns $\mathbf{u}_1, \ldots, \mathbf{u}_n$. Let A_i be $N \times N$ matrices such that $(A_i + \gamma I_N)$ is invertible for all i, $1 \leq i \leq n$. Then*

$$\left(\sum_{i=1}^{n} A_i \otimes \mathbf{u}_i\mathbf{u}_i^T + \gamma I_{Nn}\right)^{-1} = \sum_{i=1}^{n}(A_i + \gamma I_N)^{-1} \otimes \mathbf{u}_i\mathbf{u}_i^T. \tag{171}$$

Proof By definition of orthogonal matrices, we have $UU^T = I_n$, which is equivalent to $\sum_{i=1}^n \mathbf{u}_i\mathbf{u}_i^T = I_n$, so that

$$\sum_{i=1}^{n} A_i \otimes \mathbf{u}_i\mathbf{u}_i^T + \gamma I_{Nn} = \sum_{i=1}^{n} A_i \otimes \mathbf{u}_i\mathbf{u}_i^T + \gamma I_N \otimes \sum_{i=1}^{n}\mathbf{u}_i\mathbf{u}_i^T = \sum_{i=1}^{n}(A_i + \gamma I_N) \otimes \mathbf{u}_i\mathbf{u}_i^T.$$

Noting that $\langle \mathbf{u}_i, \mathbf{u}_j\rangle = \delta_{ij}$, the expression for the inverse matrix then follows immediately by direct verification. ■

Proof of Theorem 11 From the property $K[\mathbf{x}] = G[\mathbf{x}] \otimes R$ and the definitions $M_B = I_{u+l} \otimes M_m \otimes I_{\mathcal{Y}}$, $M_W = L \otimes I_{\mathcal{Y}}$, we have

$$\begin{aligned}
&\gamma_I MK[\mathbf{x}] + \gamma_A I_{\mathcal{W}^{u+l}} = (\gamma_B M_B + \gamma_W M_W)K[\mathbf{x}] + \gamma_A I_{u+l} \otimes I_{\mathcal{W}} \\
&= (\gamma_B I_{u+l} \otimes M_m \otimes I_{\mathcal{Y}} + \gamma_W L \otimes I_{\mathcal{Y}})(G[\mathbf{x}] \otimes R) + \gamma_A I_{u+l} \otimes I_m \otimes I_{\mathcal{Y}} \\
&= (\gamma_B I_{u+l} \otimes M_m + \gamma_W L)G[\mathbf{x}] \otimes R + \gamma_A I_{m(u+l)} \otimes I_{\mathcal{Y}}.
\end{aligned}$$

With the spectral decomposition of R,

$$R = \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R}\mathbf{r}_i\mathbf{r}_i^T,$$

we have

$$(\gamma_B I_{u+l} \otimes M_m + \gamma_W L)G[\mathbf{x}] \otimes R = \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R}(\gamma_B I_{u+l} \otimes M_m + \gamma_W L)G[\mathbf{x}] \otimes \mathbf{r}_i\mathbf{r}_i^T.$$

It follows from Lemma 30 that

$$(\gamma_I MK[\mathbf{x}] + \gamma_A I_{\mathcal{W}^{u+l}})^{-1} = \sum_{i=1}^{\dim(\mathcal{Y})} [\lambda_{i,R}(\gamma_B I_{u+l} \otimes M_m + \gamma_W L)G[\mathbf{x}] + \gamma_A I_{m(u+l)}]^{-1} \otimes \mathbf{r}_i\mathbf{r}_i^T$$

$$= \sum_{i=1}^{\dim(\mathcal{Y})} M_{\mathrm{reg}}^i \otimes \mathbf{r}_i\mathbf{r}_i^T, \quad \text{where} \quad M_{\mathrm{reg}}^i = [\lambda_{i,R}(\gamma_B I_{u+l} \otimes M_m + \gamma_W L)G[\mathbf{x}] + \gamma_A I_{m(u+l)}]^{-1}.$$

For $C = \mathbf{c}^T \otimes I_{\mathcal{Y}} \in \mathbb{R}^{\dim(\mathcal{Y}) \times m\dim(\mathcal{Y})}$, we have

$$\begin{aligned}
C^*S &= (\mathbf{c} \otimes I_{\mathcal{Y}})S = \mathbf{c} \otimes S, \\
S^*C &= S^*(\mathbf{c}^T \otimes I_{\mathcal{Y}}) = \mathbf{c}^T \otimes S^*, \\
I_{(u+l)\times l}^T \otimes S^*C &= I_{(u+l)\times l}^T \otimes \mathbf{c}^T \otimes S^*, \\
I_{(u+l)\times l} \otimes C^*S &= I_{(u+l)\times l} \otimes \mathbf{c} \otimes S.
\end{aligned}$$

It follows that

$$\begin{aligned}
(\gamma_I MK[\mathbf{x}] + \gamma_A I_{\mathcal{W}^{u+l}})^{-1}(I_{(u+l)\times l} \otimes C^*S) &= (\sum_{i=1}^{\dim(\mathcal{Y})} M_{\mathrm{reg}}^i \otimes \mathbf{r}_i\mathbf{r}_i^T)(I_{(u+l)\times l} \otimes \mathbf{c} \otimes S) \\
&= \sum_{i=1}^{\dim(\mathcal{Y})} M_{\mathrm{reg}}^i(I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \mathbf{r}_i\mathbf{r}_i^T S,
\end{aligned}$$

from which we obtain the expression for $\mathbf{a}$. Next,

$$\begin{aligned}
K[\mathbf{x}](\gamma_I MK[\mathbf{x}] + \gamma_A I)^{-1} &= (G[\mathbf{x}] \otimes R)(\sum_{i=1}^{\dim(\mathcal{Y})} M_{\mathrm{reg}}^i \otimes \mathbf{r}_i\mathbf{r}_i^T) \\
= (\sum_{i=1}^{\dim(\mathcal{Y})} G[\mathbf{x}] \otimes \lambda_{i,R}\mathbf{r}_i\mathbf{r}_i^T)(\sum_{i=1}^{\dim(\mathcal{Y})} M_{\mathrm{reg}}^i \otimes \mathbf{r}_i\mathbf{r}_i^T) &= \sum_{i=1}^{\dim(\mathcal{Y})} G[\mathbf{x}]M_{\mathrm{reg}}^i \otimes \lambda_{i,R}\mathbf{r}_i\mathbf{r}_i^T.
\end{aligned}$$

Thus for $Q[\mathbf{x}, C]$, we have

$$\begin{aligned}
Q[\mathbf{x}, C] &= (I^T_{(u+l)\times l} \otimes S^* C) K[\mathbf{x}](\gamma_I M K[\mathbf{x}] + \gamma_A I)^{-1}(I_{(u+l)\times l} \otimes C^* S) \\
&= (I^T_{(u+l)\times l} \otimes \mathbf{c}^T \otimes S^*)(\sum_{i=1}^{\dim(\mathcal{Y})} G[\mathbf{x}] M^i_{\text{reg}} \otimes \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T)(I_{(u+l)\times l} \otimes \mathbf{c} \otimes S) \\
&= (\sum_{i=1}^{\dim(\mathcal{Y})} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) G[\mathbf{x}] M^i_{\text{reg}} \otimes \lambda_{i,R} S^* \mathbf{r}_i \mathbf{r}_i^T)(I_{(u+l)\times l} \otimes \mathbf{c} \otimes S) \\
&= \sum_{i=1}^{\dim(\mathcal{Y})} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T) G[\mathbf{x}] M^i_{\text{reg}} (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \lambda_{i,R} S^* \mathbf{r}_i \mathbf{r}_i^T S.
\end{aligned}$$

This completes the proof of the theorem. ■

Proof of Theorem 13 For $R = I_{\mathcal{Y}}$ we have $\lambda_{i,R} = 1$, $1 \le i \le \dim(\mathcal{Y})$, so that in Theorem 11

$$M^i_{\text{reg}} = M_{\text{reg}} = [(\gamma_B I_{u+l} \otimes M_m + \gamma_W L) G[\mathbf{x}] + \gamma_A I_{m(u+l)}]^{-1}.$$

Since $\sum_{i=1}^{\dim(\mathcal{Y})} \mathbf{r}_i \mathbf{r}_i^T = I_{\mathcal{Y}}$, by substituting $M^i_{\text{reg}} = M_{\text{reg}}$ into the formulas for $\mathbf{a}$ and $Q[\mathbf{x}, C]$ in Theorem 11, we obtain the corresponding expressions (77) and (78). ■

Proof of Propositions 12 and 14 By Theorems 6 and 11, we have

$$\begin{aligned}
f_{\mathbf{z},\gamma}(v) &= \sum_{j=1}^{u+l} K(v, x_j) a_j = K[v, \mathbf{x}]\mathbf{a} = (G[v, \mathbf{x}] \otimes R)\mathbf{a} \\
&= -\frac{1}{2}(G[v, \mathbf{x}] \otimes \sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T)[\sum_{i=1}^{\dim(\mathcal{Y})} M^i_{\text{reg}} (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \mathbf{r}_i \mathbf{r}_i^T S]\text{vec}(\alpha^{\text{opt}}) \\
&= -\frac{1}{2}[\sum_{i=1}^{\dim(\mathcal{Y})} G[v, \mathbf{x}] M^i_{\text{reg}} (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S]\text{vec}(\alpha^{\text{opt}}) \\
&= -\frac{1}{2}\text{vec}(\sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[v, \mathbf{x}]^T).
\end{aligned}$$

The combined function, using the combination operator C, is

$$\begin{aligned}
g_{\mathbf{z},\gamma}(v) &= C f_{\mathbf{z},\gamma}(v) = (\mathbf{c}^T \otimes I_{\mathcal{Y}})(G[v, \mathbf{x}] \otimes R)\mathbf{a} = (\mathbf{c}^T G[v, \mathbf{x}] \otimes R)\mathbf{a} \\
&= -\frac{1}{2}[\sum_{i=1}^{\dim(\mathcal{Y})} \mathbf{c}^T G[v, \mathbf{x}] M^i_{\text{reg}} (I_{(u+l)\times l} \otimes \mathbf{c}) \otimes \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S]\text{vec}(\alpha^{\text{opt}}) \\
&= -\frac{1}{2}\text{vec}(\sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[v, \mathbf{x}]^T \mathbf{c}) \\
&= -\frac{1}{2}\sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[v, \mathbf{x}]^T \mathbf{c} \in \mathbb{R}^{\dim(\mathcal{Y})}.
\end{aligned}$$

It follows that on a set $\mathbf{v} = \{v_i\}_{i=1}^t \subset \mathcal{X}$,

$$g_{\mathbf{z},\gamma}(\mathbf{v}) = -\frac{1}{2}\sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[\mathbf{v}, \mathbf{x}]^T (I_t \otimes \mathbf{c}) \in \mathbb{R}^{\dim(\mathcal{Y})\times t}.$$

The final SVM decision function is then given by

$$\begin{aligned}
h_{\mathbf{z},\gamma}(\mathbf{v}) &= S^T g_{\mathbf{z},\gamma}(\mathbf{v}) \\
&= -\frac{1}{2}\sum_{i=1}^{\dim(\mathcal{Y})} \lambda_{i,R} S^T \mathbf{r}_i \mathbf{r}_i^T S \alpha^{\text{opt}} (I^T_{(u+l)\times l} \otimes \mathbf{c}^T)(M^i_{\text{reg}})^T G[\mathbf{v}, \mathbf{x}]^T (I_t \otimes \mathbf{c}) \in \mathbb{R}^{P\times t}.
\end{aligned}$$

This completes the proof for Proposition 12. Proposition 14 then follows by noting that in Theorem 13, with $R = I_{\mathcal{Y}}$, we have $M^i_{\text{reg}} = M_{\text{reg}}$, $\lambda_{i,R} = 1$, $1 \le i \le \dim(\mathcal{Y})$, and $\sum_{i=1}^{\dim(\mathcal{Y})} \mathbf{r}_i \mathbf{r}_i^T = I_{\mathcal{Y}}$. ■

Proof of Corollary 20 Clearly, for $\gamma_I = 0$ and $u = 0$, we have

$$Q[\mathbf{x}, \mathbf{y}, C] = \frac{1}{\gamma_A}\text{diag}(S^*_{\hat{\mathbf{y}}})(I_l \otimes C) K[\mathbf{x}](I_l \otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}}).$$

For $C = \mathbf{c}^T \otimes I_{\mathcal{Y}}$ and $K[\mathbf{x}] = G[\mathbf{x}] \otimes R$, this is

$$Q[\mathbf{x}, \mathbf{y}, C] = \frac{1}{\gamma_A}\text{diag}(S^*_{\hat{\mathbf{y}}})[(I_l \otimes \mathbf{c}^T) G[\mathbf{x}](I_l \otimes \mathbf{c}) \otimes R]\text{diag}(S_{\hat{\mathbf{y}}}).$$

For $R = I_{\mathcal{Y}}$, we have

$$Q[\mathbf{x}, \mathbf{y}, C] = \frac{1}{\gamma_A}\text{diag}(S^*_{\hat{\mathbf{y}}})[(I_l \otimes \mathbf{c}^T) G[\mathbf{x}](I_l \otimes \mathbf{c}) \otimes I_{\mathcal{Y}}]\text{diag}(S_{\hat{\mathbf{y}}}).$$

With $G[\mathbf{x}] = \sum_{i=1}^m k^i[\mathbf{x}] \otimes \mathbf{e}_i \mathbf{e}_i^T$,

$$(I_l \otimes \mathbf{c}^T) G[\mathbf{x}](I_l \otimes \mathbf{c}) = \sum_{i=1}^m c_i^2 k^i[\mathbf{x}],$$

so that

$$Q[\mathbf{x},\mathbf{y},C] = \frac{1}{\gamma_A}\text{diag}(S^*_{\hat{\mathbf{y}}})\left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}]\otimes I_{\mathcal{Y}}\right)\text{diag}(S_{\hat{\mathbf{y}}}),$$

which, when $\mathcal{Y}=\mathbb{R}$, reduces to

$$Q[\mathbf{x},\mathbf{y},C] = \frac{1}{\gamma_A}\text{diag}(\mathbf{y})\left(\sum_{i=1}^m c_i^2 k^i[\mathbf{x}]\right)\text{diag}(\mathbf{y}).$$

Similarly, when $\gamma_I = 0$, $u = 0$, we have

$$\mathbf{a} = -\frac{1}{2\gamma_A}(I_l\otimes C^*)\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta^{\text{opt}}).$$

For $C = \mathbf{c}\otimes I_{\mathcal{Y}}$, $K(x,t) = G(x,t)\otimes R$, we have for any $v\in\mathcal{X}$,

$$\begin{aligned} Cf_{\mathbf{z},\gamma}(v) = CK[v,\mathbf{x}]\mathbf{a} &= -\frac{1}{2\gamma_A}(\mathbf{c}^T\otimes I_{\mathcal{Y}})(G[v,\mathbf{x}]\otimes R)(I_l\otimes\mathbf{c}\otimes I_{\mathcal{Y}})\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta^{\text{opt}}) \\ &= -\frac{1}{2\gamma_A}[\mathbf{c}^T G[v,\mathbf{x}](I_l\otimes\mathbf{c})\otimes R]\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta^{\text{opt}}), \end{aligned}$$

which for $R = I_{\mathcal{Y}}$, simplifies to

$$Cf_{\mathbf{z},\gamma}(v) = -\frac{1}{2\gamma_A}[\mathbf{c}^T G[v,\mathbf{x}](I_l\otimes\mathbf{c})\otimes I_{\mathcal{Y}}]\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta^{\text{opt}}),$$

With $G(x,t) = \sum_{i=1}^m k^i(x,t)\otimes\mathbf{e}_i\mathbf{e}_i^T$,

$$\mathbf{c}^T G[v,\mathbf{x}](I_l\otimes\mathbf{c}) = \sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}],$$

so that

$$Cf_{\mathbf{z},\gamma}(v) = -\frac{1}{2\gamma_A}\left(\sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}]\otimes I_{\mathcal{Y}}\right)\text{diag}(S_{\hat{\mathbf{y}}})\text{vec}(\beta^{\text{opt}}).$$

For $\mathcal{Y}=\mathbb{R}$, this simplifies to

$$Cf_{\mathbf{z},\gamma}(v) = -\frac{1}{2\gamma_A}\left(\sum_{i=1}^m c_i^2 k^i[v,\mathbf{x}]\right)\text{diag}(\mathbf{y})\beta^{\text{opt}}.$$

This completes the proof. ■

A.4 Sequential Minimal Optimization

This section describes the Sequential Minimal Optimization (SMO) algorithm we use to solve the quadratic optimization problem for MV-SVM in Theorem 6. It is a generalization of the one-step SMO technique described in (Platt, 1999). For simplicity and clarity, we

consider the case of the simplex coding, that is the quadratic optimization problem (26). The ideas presented here are readily extendible to the general setting.

Let us first consider the SMO technique for the quadratic optimization problem

$$\text{argmin}_{\alpha\in\mathbb{R}^{Pl}} D(\alpha) = \frac{1}{4}\alpha^T Q\alpha - \frac{1}{P-1}\mathbf{1}_{Pl}^T\alpha, \tag{172}$$

where Q is a symmetric, positive semidefinite matrix of size $Pl\times Pl$, such that $Q_{ii} > 0$, $1\le i\le Pl$, under the constraints

$$0\le\alpha_i\le\frac{1}{l},\quad 1\le i\le Pl. \tag{173}$$

For i fixed, $1\le i\le Pl$, as a function of α_i,

$$D(\alpha) = \frac{1}{4}Q_{ii}\alpha_i^2 + \frac{1}{2}\sum_{j=1,j\neq i}^{Pl} Q_{ij}\alpha_i\alpha_j - \frac{1}{P-1}\alpha_i + Q_{\text{const}}, \tag{174}$$

where Q_{const} is a quantity constant in α_i. Differentiating with respect to α_i gives

$$\frac{\partial D}{\partial\alpha_i} = \frac{1}{2}Q_{ii}\alpha_i + \frac{1}{2}\sum_{j=1,j\neq i}^{Pl} Q_{ij}\alpha_j - \frac{1}{P-1}. \tag{175}$$

Under the condition that $Q_{ii} > 0$, setting this partial derivative to zero gives

$$\alpha_i^* = \frac{1}{Q_{ii}}\left(\frac{2}{P-1} - \sum_{j=1,j\neq i}^{Pl} Q_{ij}\alpha_j\right) = \alpha_i + \frac{1}{Q_{ii}}\left(\frac{2}{P-1} - \sum_{j=1}^{Pl} Q_{ij}\alpha_j\right). \tag{176}$$

Thus the iterative sequence for α_i at step t is

$$\alpha_i^{t+1} = \alpha_i^t + \frac{1}{Q_{ii}}\left(\frac{2}{P-1} - \sum_{j=1}^{Pl} Q_{ij}\alpha_j^t\right), \tag{177}$$

after which we perform a clipping operation, defined by

$$\text{clip}(\alpha_i) = \begin{cases} 0 & \text{if } \alpha_i < 0, \\ \alpha_i & \text{if } 0\le\alpha_i\le\frac{1}{l}, \\ \frac{1}{l} & \text{if } \alpha_i > \frac{1}{l}. \end{cases} \tag{178}$$

Let us now apply this SMO technique for the quadratic optimization (26) in Theorem 6. Recall that this problem is

$$\alpha^{\text{opt}} = \text{argmin}_{\alpha\in\mathbb{R}^{P\times l}}\left\{D(\alpha) = \frac{1}{4}\text{vec}(\alpha)^T Q[\mathbf{x},C]\text{vec}(\alpha) - \frac{1}{P-1}\mathbf{1}_{Pl}^T\text{vec}(\alpha)\right\},$$

with $\mathbf{1}_{Pl} = (1,\ldots,1)^T\in\mathbb{R}^{Pl}$, subject to the constraints

$$0\le\alpha_{ki}\le\frac{1}{l}(1-\delta_{k,y_i}),\quad 1\le i\le l, 1\le k\le P.$$

The choice of which α_{ki} to update at each step is made via the Karush-Kuhn-Tucker (KKT) conditions. In the present context, the KKT conditions are:

$$\alpha_{ki}\left(\xi_{ki} - \left[\frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}}\right]\right) = 0, \quad 1 \leq i \leq l, k \neq y_i, \tag{179}$$

$$\left(\frac{1}{l} - \alpha_{ki}\right)\xi_{ki} = 0, \quad 1 \leq i \leq l, k \neq y_i. \tag{180}$$

At an optimal point α^{opt},

$$\xi_{ki}^{\text{opt}} = \max\left(0, \frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}}\right), \quad 1 \leq i \leq l, k \neq y_i. \tag{181}$$

We have the following result.

Lemma 31 *For $1 \leq i \leq l$, $k \neq y_i$,*

$$\alpha_{ki}^{\text{opt}} = 0 \Longrightarrow \langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}} \leq -\frac{1}{P-1}, \tag{182}$$

$$0 < \alpha_{ki}^{\text{opt}} < \frac{1}{l} \Longrightarrow \langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}} = -\frac{1}{P-1}, \tag{183}$$

$$\alpha_{ki}^{\text{opt}} = \frac{1}{l} \Longrightarrow \langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}} \geq -\frac{1}{P-1}. \tag{184}$$

Conversely,

$$\langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}} < -\frac{1}{P-1} \Longrightarrow \alpha_{ki}^{\text{opt}} = 0, \tag{185}$$

$$\langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}} > -\frac{1}{P-1} \Longrightarrow \alpha_{ki}^{\text{opt}} = \frac{1}{l}. \tag{186}$$

Remark 32 *Note that the inequalities in (182) and (184) are not strict. Thus from $\langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}} = -\frac{1}{P-1}$ we cannot draw any conclusion about α_{ki}^{opt}.*

Proof To prove (182), note that if $\alpha_{ki}^{\text{opt}} = 0$, then from (180), we have $\xi_{ki}^{\text{opt}} = 0$. From (181), we have

$$\frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} \leq 0 \Longrightarrow \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} \leq -\frac{1}{P-1}.$$

To prove (183), note that if $0 < \alpha_{ki}^{\text{opt}} < \frac{1}{l}$, then from (180), we have $\xi_{ki}^{\text{opt}} = 0$. On the other hand, from (179), we have

$$\xi_{ki}^{\text{opt}} = \frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}}.$$

It follows that

$$\frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} = 0 \Longleftrightarrow \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} = -\frac{1}{P-1}.$$

For (184), note that if $\alpha_{ki}^{\text{opt}} = \frac{1}{l}$, then from (179), we have

$$\xi_{ki}^{\text{opt}} = \frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} \geq 0 \Longrightarrow \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} \geq -\frac{1}{P-1}.$$

Conversely, if $\langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} < -\frac{1}{P-1}$, then from (181), we have $\xi_{ki}^{\text{opt}} = 0$. It then follows from (179) that $\alpha_{ki}^{\text{opt}} = 0$. If $\langle s_k, Cf(x_i)\rangle_{\mathcal{Y}} > -\frac{1}{P-1}$, then from (181) we have $\xi_{ki}^{\text{opt}} = \frac{1}{P-1} + \langle s_k, Cf(x_i)\rangle_{\mathcal{Y}}$. Then from (180) it follows that $\alpha_{ki}^{\text{opt}} = \frac{1}{l}$. ■

Binary case ($P = 2$). The binary simplex code is $S = [-1, 1]$. Thus $k \neq y_i$ means that $s_k = -y_i$. Therefore for $1 \leq i \leq l$, $k \neq y_i$, the KKT conditions are:

$$\alpha_{ki}^{\text{opt}} = 0 \Longrightarrow y_i\langle \mathbf{c}, f_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{W}} \geq 1, \tag{187}$$

$$0 < \alpha_{ki}^{\text{opt}} < \frac{1}{l} \Longrightarrow y_i\langle \mathbf{c}, f_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{W}} = 1, \tag{188}$$

$$\alpha_{ki}^{\text{opt}} = \frac{1}{l} \Longrightarrow y_i\langle \mathbf{c}, f_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{W}} \leq 1. \tag{189}$$

Conversely,

$$y_i\langle \mathbf{c}, f_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{W}} > 1 \Longrightarrow \alpha_{ki}^{\text{opt}} = 0, \tag{190}$$

$$y_i\langle \mathbf{c}, f_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{W}} < 1 \Longrightarrow \alpha_{ki}^{\text{opt}} = \frac{1}{l}. \tag{191}$$

Algorithm 3 summarizes the SMO procedure described in this section.

A.4.1 Numerical implementation of SMO

Let us elaborate on the steps of Algorithm 3 under the hypotheses of Theorem 13, that is the simplex coding with $K[\mathbf{x}] = G[\mathbf{x}] \otimes R$ for $R = I_{P-1}$, which we implement numerically.

Verifying the Karush-Kuhn-Tucker conditions on the labeled training data. To verify Lemma 31 on the set of labeled training data $\mathbf{x}_{1:l} = \{x_i\}_{i=1}^l \subset \mathbf{x}$, according to Proposition 14, we compute

$$h_{\mathbf{z},\gamma}(\mathbf{x}_{1:l}) = -\frac{1}{2}S^T S\alpha^{\text{opt}}(I_{(u+l)\times l}^T \otimes \mathbf{c}^T)M_{\text{reg}}^T G[\mathbf{x}_{1:l}, \mathbf{x}]^T(I_l \otimes \mathbf{c}) \in \mathbb{R}^{P\times l}, \tag{192}$$

as a matrix of size $P \times l$, with the ith column being $h_{\mathbf{z},\gamma}(x_i) = (\langle s_k, Cf_{\mathbf{z},\gamma}(x_i)\rangle_{\mathcal{Y}})_{k=1}^P$, which is then compared with the margin value $-\frac{1}{P-1}$.

Efficient evaluation of the update step (193). The most important factor underlying the efficiency of Algorithm 3 is that we never compute the whole matrix Q of size $Pl \times Pl$, which can be prohibitively large. At each update step, i.e. (193), we only use the ith row of Q, which we denote $Q(i, :)$, which need not be computed explicitly. Recall that we have

$$Q = Q[\mathbf{x}, C] = (I_{(u+l)\times l}^T \otimes \mathbf{c}^T)G[\mathbf{x}]M_{\text{reg}}(I_{(u+l)\times l} \otimes \mathbf{c}) \otimes S^*S = Q_G \otimes Q_S,$$

where

$$Q_G = (I_{(u+l)\times l}^T \otimes \mathbf{c}^T)G[\mathbf{x}]M_{\text{reg}}(I_{(u+l)\times l} \otimes \mathbf{c}), \tag{194}$$

Algorithm 3 Sequential Minimal Optimization for Multi-class Multi-view SVM

Note: We use $\alpha \in \mathbb{R}^{P\times l}$ as a matrix and $\alpha_{\text{vec}} = \text{vec}(\alpha) \in \mathbb{R}^{Pl}$ as a column vector interchangeably.
Initialization: Set $\alpha^0 = 0$.
Stopping criterion: $\frac{|D(\alpha^{t+1}) - D(\alpha^t)|}{|D(\alpha^{t+1})|} < \epsilon$, for some $\epsilon > 0$.
Repeat: - Verify KKT conditions according to Lemma 31.
- Randomly pick an $i \in \mathbb{N}$ such that $\alpha^t_{\text{vec},i}$ is a KKT violator.
- Perform update:

$$\alpha^{t+1}_{\text{vec},i} = \text{clip}\left(\alpha^t_{\text{vec},i} + \frac{1}{Q_{ii}}\left(\frac{2}{P-1} - \sum_{j=1}^{Pl} Q_{ij}\alpha^t_{\text{vec},j}\right)\right), \tag{193}$$

where $Q = Q[\mathbf{x}, C]$.
Until: There are no KKT violators or the stopping criterion is met.

and

$$Q_S = S^*S. \tag{195}$$

Thus for each i, the ith row of Q is

$$Q(i,:) = Q_G(i_G,:) \otimes Q_S(i_S,:), \tag{196}$$

for a unique pair of indices i_G and i_S. It then follows that

$$\begin{aligned} Q(i,:)\text{vec}(\alpha) &= (Q_G(i_G,:) \otimes Q_S(i_S,:))\text{vec}(\alpha) = \text{vec}(Q_S(i_S,:)\alpha Q_G(i_G,:)^T) \\ &= Q_S(i_S,:)\alpha Q_G(i_G,:)^T = Q_S(i_S,:)\alpha Q_G(:,i_G) \end{aligned} \tag{197}$$

since Q_G is symmetric. Also

$$Q_{ii} = Q_G(i_G, i_G)Q_S(i_S, i_S). \tag{198}$$

When proceeding in this way, each update step (193) only uses *one* row from the $l \times l$ matrix Q_G and *one* row from the $P \times P$ matrix Q_S. This is the key to the computational efficiency of Algorithm 3.

Remark 33 *In the more general setting of Theorem 11, with $K[\mathbf{x}] = G[\mathbf{x}] \otimes R$, where R is a positive semi-definite matrix, the evaluation of the matrix $Q = Q[\mathbf{x}, C]$ is done in the same way, except that we need to sum over all non-zero eigenvalues of R, as in (71).*

A.5 Proofs for the Optimization of the Combination Operator

In this section, we prove Theorems 15 and 16 stated in Section 6.1. Consider the optimization problem (86), namely

$$\min_{\mathbf{x}\in\mathbb{R}^m} ||A\mathbf{x} - \mathbf{b}||_{\mathbb{R}^n} \text{ subject to } ||\mathbf{x}||_{\mathbb{R}^m} = \alpha.$$

The Lagrangian, with Lagrange multiplier γ, is given by

$$L(\mathbf{x}, \gamma) = ||A\mathbf{x} - \mathbf{b}||^2 + \gamma(||\mathbf{x}||^2 - \alpha^2).$$

Setting $\frac{\partial L}{\partial \mathbf{x}} = 0$ and $\frac{\partial L}{\partial \gamma} = 0$, we obtain the normal equations

$$(A^TA + \gamma I_m)\mathbf{x} = A^T\mathbf{b}, \tag{199}$$

$$||\mathbf{x}||^2 = \alpha^2. \tag{200}$$

The solutions of the normal equations (199) and (200), if they exist, satisfy the following properties (Gander, 1981).

Lemma 34 *If $(\mathbf{x}_1, \gamma_1)$ and $(\mathbf{x}_2, \gamma_2)$ are solutions of the normal equations (199) and (200), then*

$$||A\mathbf{x}_2 - \mathbf{b}||^2 - ||A\mathbf{x}_1 - \mathbf{b}||^2 = \frac{\gamma_1 - \gamma_2}{2}||\mathbf{x}_1 - \mathbf{x}_2||^2. \tag{201}$$

Lemma 35 *The right hand side of equation (201) is equal to zero only if $\gamma_1 = \gamma_2 = -\mu$, where $\mu \geq 0$ is an eigenvalue of A^TA and*

$$\mathbf{x}_1 = \mathbf{x}_2 + \mathbf{v}(\mu), \tag{202}$$

where $\mathbf{v}(\mu)$ is an eigenvector corresponding to μ.

According to Lemmas 34 and 35, if $(\mathbf{x}_1, \gamma_1)$ and $(\mathbf{x}_2, \gamma_2)$ are solutions of the normal equations (199) and (200), then

$$\gamma_1 > \gamma_2 \Longrightarrow ||A\mathbf{x}_2 - \mathbf{b}|| > ||A\mathbf{x}_1 - \mathbf{b}||. \tag{203}$$

Consequently, among all possible solutions of the normal equations (199) and (200), we choose the solution $(\mathbf{x}, \gamma)$ with the largest γ.
Proof of Theorem 15 Using the assumption $A^T\mathbf{b} = 0$, the first normal equation (199) implies that

$$A^TA\mathbf{x} = -\gamma\mathbf{x}, \tag{204}$$

so that $-\gamma$ is an eigenvalue of A^TA and $\mathbf{x}$ is its corresponding eigenvector, which can be appropriately normalized such that $||\mathbf{x}||_{\mathbb{R}^m} = \alpha$. Since we need the largest value for γ, we have $\gamma^* = -\mu_m$. The minimum value is then

$$\begin{aligned} ||A\mathbf{x}^* - \mathbf{b}||^2_{\mathbb{R}^n} &= \langle A\mathbf{x}^*, A\mathbf{x}^*\rangle_{\mathbb{R}^n} - 2\langle A\mathbf{x}^*, \mathbf{b}\rangle_{\mathbb{R}^n} + ||\mathbf{b}||^2_{\mathbb{R}^n} \\ &= \langle \mathbf{x}^*, A^TA\mathbf{x}^*\rangle_{\mathbb{R}^m} - 2\langle \mathbf{x}^*, A^T\mathbf{b}\rangle_{\mathbb{R}^m} + ||\mathbf{b}||^2_{\mathbb{R}^n} = -\gamma^*||\mathbf{x}^*||^2_{\mathbb{R}^m} + ||\mathbf{b}||^2_{\mathbb{R}^n} \\ &= \mu_m\alpha^2 + ||\mathbf{b}||^2_{\mathbb{R}^n}. \end{aligned}$$

This solution is clearly unique if and only if μ_m is a single eigenvalue. Otherwise, there are infinitely many solutions, each being a vector of length α in the eigenspace of μ_m. This completes the proof of the theorem. ■

Proof of Theorem 16 We first show that under the assumption $A^T\mathbf{b} \neq 0$ and $\mathbf{c} = U^T\mathbf{b}$, we have $||\mathbf{c}_{1:r}||_{\mathbb{R}^r} \neq 0$, that is $c_i \neq 0$ for at least one index i, $1 \leq i \leq r$. To see this, assume that $c_i = 0$ for all i, $1 \leq i \leq r$. Then

$$A^T\mathbf{b} = V\Sigma^T U^T\mathbf{b} = V\Sigma^T\mathbf{c} = 0,$$

which is a contradiction. Thus $||\mathbf{c}_{1:r}||_{\mathbb{R}^r} \neq 0$.

There are two cases in this scenario.

(I) If $\gamma \neq -\mu_i$, $1 \leq i \leq m$, then the matrix $(A^TA + \gamma I_m)$ is nonsingular, thus

$$\begin{aligned}\mathbf{x}(\gamma) &= (A^TA + \gamma I_m)^{-1}A^T\mathbf{b} = (VDV^T + \gamma I_m)^{-1}V\Sigma^T U^T\mathbf{b} \\ &= V(DV^TV + \gamma I_m)^{-1}\Sigma^T U^T\mathbf{b} = V(D + \gamma I_m)^{-1}\Sigma^T U^T\mathbf{b}.\end{aligned}$$

Since the matrix V is orthogonal, we have

$$||\mathbf{x}(\gamma)||^2_{\mathbb{R}^m} = ||(D + \gamma I_m)^{-1}\Sigma^T U^T\mathbf{b}||^2_{\mathbb{R}^m} = \sum_{i=1}^{r} \frac{\sigma_i^2 c_i^2}{(\sigma_i^2 + \gamma)^2},$$

where $\mathbf{c} = U^T\mathbf{b}$. We now need to find γ such that $||\mathbf{x}(\gamma)||_{\mathbb{R}^m} = \alpha$. Consider the function

$$s(\gamma) = \sum_{i=1}^{r} \frac{\sigma_i^2 c_i^2}{(\sigma_i^2 + \gamma)^2} \tag{205}$$

on the interval $(-\sigma_r^2, \infty)$. Under the condition that at least one of the c_i's, $1 \leq i \leq r$, is nonzero, the function s is strictly positive and monotonically decreasing on $(-\sigma_r^2, \infty)$, with

$$\lim_{\gamma\to\infty} s(\gamma) = 0, \quad \lim_{\gamma\to-\sigma_r^2} s(\gamma) = \infty. \tag{206}$$

Thus there must exist a unique $\gamma^* \in (-\sigma_r^2, \infty)$ such that

$$s(\gamma^*) = \sum_{i=1}^{r} \frac{\sigma_i^2 c_i^2}{(\sigma_i^2 + \gamma^*)^2} = \alpha^2. \tag{207}$$

1) If $\text{rank}(A) = m$, then $r = m$ and $\gamma^* > -\sigma_m^2 = -\mu_m \geq -\mu_i$ for all $1 \leq i \leq m$. Thus $\mathbf{x}(\gamma^*)$ is the unique global solution.

2) If $\text{rank}(A) < m$ but $\gamma^* > 0$, then we still have $\gamma^* > -\mu_i$ for all i, $1 \leq i \leq m$, and thus $\mathbf{x}(\gamma^*)$ is the unique global solution.

(II) Consider now the case $\text{rank}(A) < m$ and $\gamma^* \leq 0$.

Since $\mu_m = \ldots = \mu_{r+1} = 0$ and $-\mu_r = -\sigma_r^2 < \gamma^* \leq 0$, we need to consider the possible solution of the normal equations with $\gamma = 0$. For $\gamma = 0$, we have

$$A^TA\mathbf{x} = A^T\mathbf{b} \Longleftrightarrow VDV^T\mathbf{x} = V\Sigma^T U^T\mathbf{b} \Longleftrightarrow DV^T\mathbf{x} = \Sigma^T U^T\mathbf{b}. \tag{208}$$

Let $\mathbf{y} = V^T\mathbf{x} \in \mathbb{R}^m$. By assumption, the vector $D\mathbf{y} \in \mathbb{R}^m$ satisfies $(D\mathbf{y})_i = 0$, $r+1 \leq i \leq m$. The vector $\mathbf{z} = \Sigma^T U^T\mathbf{b} \in \mathbb{R}^m$ also satisfies $z_i = 0$, $r + 1 \leq i \leq m$. Thus the equation

$$D\mathbf{y} = \mathbf{z} \tag{209}$$

has infinitely many solutions, with y_i, $r + 1 \leq i \leq m$, taking arbitrary values. Let $\mathbf{y}_{1:r} = (y_i)_{i=1}^r$, $\mathbf{z}_{1:r} = (z_i)_{i=1}^r$, $D_r = \text{diag}(\mu_1, \ldots, \mu_r)$, $\Sigma_r = \Sigma(:, 1:r)$ consisting of the first r columns of Σ. Then

$$\mathbf{y}_{1:r} = D_r^{-1}\mathbf{z}_{1:r}, \tag{210}$$

or equivalently,

$$y_i = \frac{c_i}{\sigma_i}, \quad 1 \leq i \leq r.$$

Since V is orthogonal, we have

$$\mathbf{x} = (V^T)^{-1}\mathbf{y} = V\mathbf{y},$$

with

$$||\mathbf{x}||_{\mathbb{R}^m} = ||V\mathbf{y}||_{\mathbb{R}^m} = ||\mathbf{y}||_{\mathbb{R}^m}.$$

The second normal equation, namely

$$||\mathbf{x}||_{\mathbb{R}^m} = \alpha,$$

then is satisfied if and only if

$$||\mathbf{y}_{1:r}||_{\mathbb{R}^r} \leq ||\mathbf{y}||_{\mathbb{R}^m} = ||\mathbf{x}||_{\mathbb{R}^m} = \alpha. \tag{211}$$

This condition is equivalent to

$$\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} \leq \alpha^2. \tag{212}$$

Assuming that this is satisfied, then

$$\begin{aligned}A\mathbf{x}(0) &= U\Sigma V^T\mathbf{x} = U\Sigma V^TV\mathbf{y} = U\Sigma\mathbf{y} = U\Sigma_r\mathbf{y}_{1:r} = U\Sigma_r D_r^{-1}\mathbf{z}_{1:r} \\ &= U\Sigma_r D_r^{-1}\Sigma_r^T(U^T\mathbf{b}) = UJ_r^nU^T\mathbf{b}.\end{aligned}$$

The minimum value is thus

$$\begin{aligned}||A\mathbf{x}(0) - \mathbf{b}||_{\mathbb{R}^n} &= ||(UJ_r^nU^T - I_n)\mathbf{b}||_{\mathbb{R}^n} = ||(UJ_r^nU^T - UU^T)\mathbf{b}||_{\mathbb{R}^n} \\ &= ||U(J_r^n - I_n)U^T\mathbf{b}||_{\mathbb{R}^n} = ||(J_r^n - I_n)U^T\mathbf{b}||_{\mathbb{R}^n}.\end{aligned}$$

If $r = n$, then $J_r^n = I_n$, and

$$||A\mathbf{x}(0) - \mathbf{b}||_{\mathbb{R}^m} = 0.$$

Since $s(0) = \sum_{i=1}^r \frac{c_i^2}{\sigma_i^2}$ and s is monotonically decreasing on $(-\sigma_r^2, \infty)$, we have

$$\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} = \alpha^2 \Longleftrightarrow \gamma^* = 0. \tag{213}$$

In this case, because $\sum_{i=1}^r y_i^2 = \alpha^2$, we must have $y_{r+1} = \cdots = y_m = 0$ and consequently $\mathbf{x}(0)$ is the unique global minimum. If

$$\sum_{i=1}^{r} \frac{c_i^2}{\sigma_i^2} < \alpha^2, \tag{214}$$

then $\gamma^* \neq 0$. In this case, we can choose arbitrary values $y_{r+1}, \ldots, y_m$ such that $y_{r+1}^2 + \cdots + y_m^2 = \alpha^2 - \sum_{i=1}^r \frac{c_i^2}{\sigma_i^2}$. Consequently, there are infinitely many solutions $\mathbf{x} = V\mathbf{y}$ which achieve the global minimum.

If condition (212) is not met, that is $\sum_{i=1}^r \frac{c_i^2}{\sigma_i^2} > \alpha^2$, then the second normal equation $||\mathbf{x}||_{\mathbb{R}^m} = \alpha$ cannot be satisfied and thus there is no solution for the case $\gamma = 0$. Thus the global solution is still $\mathbf{x}(\gamma^*)$. This completes the proof of the theorem. ■

For completeness, we provide the proofs of Lemmas 34 and 35 here. Lemma 34 is a special case of Theorem 1 in (Gander, 1981) and thus the proof given here is considerably simpler. Our proof for Lemma 35 is different from that given in (Gander, 1981), since we do *not* make the assumption that rank $\begin{pmatrix} A \\ I \end{pmatrix} = m$.

Proof of Lemma 34 By equation (200), we have

$$\frac{\gamma_1 - \gamma_2}{2}||\mathbf{x}_1 - \mathbf{x}_2||^2 = \frac{\gamma_1 - \gamma_2}{2}(||\mathbf{x}_1||^2 + ||\mathbf{x}_2||^2 - 2\langle \mathbf{x}_1, \mathbf{x}_2\rangle) = (\gamma_1 - \gamma_2)\alpha^2 + (\gamma_2 - \gamma_1)\langle \mathbf{x}_1, \mathbf{x}_2\rangle$$

From equation (199),

$$\begin{aligned}
&||A\mathbf{x}_2 - \mathbf{b}||^2 - ||A\mathbf{x}_1 - \mathbf{b}||^2 = (\langle \mathbf{x}_2, A^TA\mathbf{x}_2\rangle - 2\langle \mathbf{x}_2, A^T\mathbf{b}\rangle) - (\langle \mathbf{x}_1, A^TA\mathbf{x}_1\rangle - 2\langle \mathbf{x}_1, A^T\mathbf{b}\rangle) \\
&= (\langle \mathbf{x}_2, A^T\mathbf{b} - \gamma_2\mathbf{x}_2\rangle - 2\langle \mathbf{x}_2, A^T\mathbf{b}\rangle) - (\langle \mathbf{x}_1, A^T\mathbf{b} - \gamma_1\mathbf{x}_1\rangle - 2\langle \mathbf{x}_1, A^T\mathbf{b}\rangle) \\
&= \gamma_1||\mathbf{x}_1||^2 + \langle \mathbf{x}_1, A^T\mathbf{b}\rangle - \gamma_2||\mathbf{x}_2||^2 - \langle \mathbf{x}_2, A^T\mathbf{b}\rangle = (\gamma_1 - \gamma_2)\alpha^2 + \langle \mathbf{x}_1, A^T\mathbf{b}\rangle - \langle \mathbf{x}_2, A^T\mathbf{b}\rangle.
\end{aligned}$$

Also from equation (199), we have

$$\langle \mathbf{x}_1, (A^TA + \gamma_2 I_m)\mathbf{x}_2\rangle = \langle \mathbf{x}_1, A^T\mathbf{b}\rangle,$$

$$\langle \mathbf{x}_2, (A^TA + \gamma_1 I_m)\mathbf{x}_1\rangle = \langle \mathbf{x}_2, A^T\mathbf{b}\rangle,$$

Subtracting the second expression from the first, we obtain

$$\langle \mathbf{x}_1, A^T\mathbf{b}\rangle - \langle \mathbf{x}_2, A^T\mathbf{b}\rangle = (\gamma_2 - \gamma_1)\langle \mathbf{x}_1, \mathbf{x}_2\rangle.$$

Thus

$$||A\mathbf{x}_2 - \mathbf{b}||^2 - ||A\mathbf{x}_1 - \mathbf{b}||^2 = (\gamma_1 - \gamma_2)\alpha^2 + (\gamma_2 - \gamma_1)\langle \mathbf{x}_1, \mathbf{x}_2\rangle = \frac{\gamma_1 - \gamma_2}{2}||\mathbf{x}_1 - \mathbf{x}_2||^2.$$

This completes the proof. ■

Proof of Lemma 35 There are two possible cases under which the right hand side of (201) is equal to zero.

(I) $\gamma_1 = \gamma_2 = \gamma$ and $\mathbf{x}_1 \neq \mathbf{x}_2$. By equation (199),

$$(A^TA + \gamma I_m)(\mathbf{x}_1 - \mathbf{x}_2) = 0 \Longleftrightarrow A^TA(\mathbf{x}_1 - \mathbf{x}_2) = -\gamma(\mathbf{x}_1 - \mathbf{x}_2).$$

This means that $\gamma = -\mu$, where $\mu \geq 0$ is an eigenvalue of A^TA and

$$\mathbf{x}_1 = \mathbf{x}_2 + \mathbf{v}(\mu),$$

where $\mathbf{v}(\mu)$ is an eigenvector corresponding to μ.

(II) $\gamma_1 \neq \gamma_2$ and $\mathbf{x}_1 = \mathbf{x}_2 = \mathbf{x}$. This case is not possible, since by equation (199), we have

$$(A^TA + \gamma_1 I_m)\mathbf{x} = A^T\mathbf{b} = (A^TA + \gamma_2 I_m)\mathbf{x} \Longrightarrow (\gamma_1 - \gamma_2)\mathbf{x} = 0 \Longrightarrow \mathbf{x} = 0,$$

contradicting the assumption $\alpha > 0$. ■

Appendix B. Learning with General Bounded Linear Operators

The present framework generalizes naturally beyond the point evaluation operator

$$f(x) = K_x^* f.$$

Let $\mathcal{H}$ be a separable Hilbert space of functions on $\mathcal{X}$. We are *not* assuming that the functions in $\mathcal{H}$ are defined pointwise or with values in $\mathcal{W}$, rather we assume that $\forall x \in \mathcal{X}$, there is a bounded linear operator

$$E_x : \mathcal{H} \to \mathcal{W}, \quad ||E_x|| < \infty, \tag{215}$$

with adjoint $E_x^* : \mathcal{W} \to \mathcal{H}$. Consider the minimization

$$\begin{aligned}
f_{\mathbf{z},\gamma} = \text{argmin}_{\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^l V(y_i, CE_{x_i}f) + \gamma_A||f||_{\mathcal{H}}^2 \\
+\gamma_I\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}, \quad \text{where} \quad \mathbf{f} = (E_{x_i}f)_{i=1}^{u+l},
\end{aligned} \tag{216}$$

and its least square version

$$f_{\mathbf{z},\gamma} = \text{argmin}_{\mathcal{H}_K} \frac{1}{l}\sum_{i=1}^l ||y_i - CE_{x_i}f||_{\mathcal{Y}}^2 + \gamma_A||f||_{\mathcal{H}}^2 + \gamma_I\langle \mathbf{f}, M\mathbf{f}\rangle_{\mathcal{W}^{u+l}}. \tag{217}$$

Following are the corresponding Representer Theorem and Proposition stating the explicit solution for the least square case. When $\mathcal{H} = \mathcal{H}_K$, $E_x = K_x^*$, we recover Theorem 2 and Theorem 3, respectively.

Theorem 36 *The minimization problem (216) has a unique solution, given by* $f_{\mathbf{z},\gamma} = \sum_{i=1}^{u+l} E_{x_i}^* a_i$ *for some vectors* $a_i \in \mathcal{W}$, $1 \leq i \leq u + l$.

Proposition 37 *The minimization problem (217) has a unique solution* $f_{\mathbf{z},\gamma} = \sum_{i=1}^{u+l} E_{x_i}^* a_i$, *where the vectors* $a_i \in \mathcal{W}$ *are given by*

$$l\gamma_I \sum_{j,k=1}^{u+l} M_{ik}E_{x_k}E_{x_j}^* a_j + C^*C(\sum_{j=1}^{u+l} E_{x_i}E_{x_j}^* a_j) + l\gamma_A a_i = C^* y_i, \tag{218}$$

for $1 \leq i \leq l$, *and*

$$\gamma_I \sum_{j,k=1}^{u+l} M_{ik}E_{x_k}E_{x_j}^* a_j + \gamma_A a_i = 0, \tag{219}$$

for $l + 1 \leq i \leq u + l$.

The reproducing kernel structures come into play through the following.

Lemma 38 *Let $E : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{W})$ be defined by*

$$E(x,t) = E_x E_t^*. \tag{220}$$

Then E is a positive definite operator-valued kernel.

Proof of Lemma 38. For each pair $(x,t) \in \mathcal{X} \times \mathcal{X}$, the operator $E(x,t)$ satisfies

$$E(t,x)^* = (E_t E_x^*)^* = E_x E_t^* = E(x,t).$$

For every set $\{x_i\}_{i=1}^N$ in $\mathcal{X}$ and $\{w_i\}_{i=1}^N$ in $\mathcal{W}$,

$$\sum_{i,j=1}^N \langle w_i, E(x_i,x_j)w_j\rangle_{\mathcal{W}} = \sum_{i,j=1}^N \langle w_i, E_{x_i}E_{x_j}^* w_j\rangle_{\mathcal{W}}$$

$$= \sum_{i,j=1}^N \langle E_{x_i}^* w_i, E_{x_j}^* w_j\rangle_{\mathcal{H}} = ||\sum_{i=1}^N E_{x_i}^* w_i||_{\mathcal{H}}^2 \geq 0.$$

Thus E is an $\mathcal{L}(\mathcal{W})$-valued positive definite kernel. ■

Proof of Theorem 36 and Proposition 37. These are entirely analogous to those of Theorem 2 and Theorem 3, respectively. Instead of the sampling operator $S_{\mathbf{x}}$, we consider the operator $E_{\mathbf{x}} : \mathcal{H} \to \mathcal{W}^l$, with

$$E_{\mathbf{x}} f = (E_{x_i} f)_{i=1}^l, \tag{221}$$

with the adjoint $E_{\mathbf{x}}^* : \mathcal{W}^l \to \mathcal{H}$ given by

$$E_{\mathbf{x}}^* \mathbf{b} = \sum_{i=1}^l E_{x_i}^* b_i. \tag{222}$$

for all $\mathbf{b} = (b_i)_{i=1}^l \in \mathcal{W}^l$. The operator $E_{C,\mathbf{x}} : \mathcal{H} \to \mathcal{Y}^l$ is now defined by

$$E_{C,\mathbf{x}} f = (CE_{x_1} f, \ldots, CE_{x_l} f). \tag{223}$$

The adjoint $E_{C,\mathbf{x}}^* : \mathcal{Y}^l \to \mathcal{H}$ is

$$E_{C,\mathbf{x}}^* \mathbf{b} = \sum_{i=1}^l E_{x_i}^* C^* b_i, \tag{224}$$

for all $\mathbf{b} \in \mathcal{Y}^l$, and $E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} : \mathcal{H} \to \mathcal{H}$ is

$$E_{C,\mathbf{x}}^* E_{C,\mathbf{x}} f = \sum_{i=1}^l E_{x_i}^* C^* C E_{x_i} f. \tag{225}$$

We then apply all the steps in the proofs of Theorem 2 and Theorem 3 to get the desired results. ■

Remark 39 *We stress that in general, the function $f_{\mathbf{z},\gamma}$ is not defined pointwise, which is the case in the following example. Thus one cannot make a statement about $f_{\mathbf{z},\gamma}(x)$ for all $x \in \mathcal{X}$ without additional assumptions.*

Example 1 *Wahba (1977) $\mathcal{X} = [0,1]$, $\mathcal{H} = L^2(\mathcal{X})$, $\mathcal{W} = \mathbb{R}$. Let $G : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be continuous and*

$$E_x f = \int_0^1 G(x,t) f(t) dt. \tag{226}$$

for $f \in \mathcal{H}$. One has the reproducing kernel

$$E_x E_t^* = E(x,t) = \int_0^1 G(x,u)G(t,u)du. \tag{227}$$

Appendix C. The Degenerate Case

This section considers the Gaussian kernel $k(x,t) = \exp\left(-\frac{||x-t||^2}{\sigma^2}\right)$ when $\sigma \to \infty$ and other kernels with similar behavior. We show that for $G(x,t) = \sum_{i=1}^m k^i(x,t)\mathbf{e}_i\mathbf{e}_i^T$, $R = I_{\mathcal{Y}}$, the matrix A in Theorem 10 has an analytic expression. This can be used to verify the correctness of an implementation of Algorithm 1.

At $\sigma = \infty$, for $R = I_{\mathcal{Y}}$, for each pair (x,t), we have

$$K(x,t) = I_{\mathcal{Y}^m},$$

and

$$f_{\mathbf{z},\gamma}(x) = \sum_{i=1}^{u+l} K(x_i,x)a_i = \sum_{i=1}^{u+l} a_i.$$

Thus $f_{\mathbf{z},\gamma}$ is a constant function. Let us examine the form of the coefficients a_i's for the case

$$C = \frac{1}{m}\mathbf{1}_m^T \otimes I_{\mathcal{Y}}.$$

We have

$$G[\mathbf{x}] = \mathbf{1}_{u+l}\mathbf{1}_{u+l}^T \otimes I_m.$$

For $\gamma_I = 0$, we have

$$B = \frac{1}{m^2}(J_l^{u+l} \otimes \mathbf{1}_m\mathbf{1}_m^T)(\mathbf{1}_{u+l}\mathbf{1}_{u+l}^T \otimes I_m),$$

which is

$$B = \frac{1}{m^2}(J_l^{u+l}\mathbf{1}_{u+l}\mathbf{1}_{u+l}^T \otimes \mathbf{1}_m\mathbf{1}_m^T).$$

Equivalently,

$$B = \frac{1}{m^2}(J_{ml}^{(u+l)m}\mathbf{1}_{(u+l)m}\mathbf{1}_{(u+l)m}^T).$$

The inverse of $B + l\gamma_A I_{(u+l)m}$ in this case has a closed form

$$(B + l\gamma_A I_{(u+l)m})^{-1} = \frac{I_{(u+l)m}}{l\gamma_A} - \frac{J_{ml}^{(u+l)m}\mathbf{1}_{(u+l)m}\mathbf{1}_{(u+l)m}^T}{l^2 m\gamma_A(m\gamma_A + 1)}, \tag{228}$$

where we have used the identity

$$\mathbf{1}_{(u+l)m}\mathbf{1}_{(u+l)m}^T J_{ml}^{(u+l)m}\mathbf{1}_{(u+l)m}\mathbf{1}_{(u+l)m}^T = ml\mathbf{1}_{(u+l)m}\mathbf{1}_{(u+l)m}^T. \tag{229}$$

We have thus

$$A = (B + l\gamma_A I_{(u+l)m})^{-1}Y_C = \left(\frac{I_{(u+l)m}}{l\gamma_A} - \frac{J_{ml}^{(u+l)m}\mathbf{1}_{(u+l)m}\mathbf{1}_{(u+l)m}^T}{l^2 m\gamma_A(m\gamma_A+1)}\right)Y_C. \tag{230}$$

Thus in this case we have an analytic expression for the coefficient matrix A, as we claimed.

References

F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2004.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

U. Brefeld, T. Gärtner, T. Scheffer, and S. Wrobel. Efficient co-regularised least squares regression. In *Proceedings of the International Conference on Machine Learning(ICML)*, 2006.

C. Brouard, F. D'Alche-Buc, and M. Szafranski. Semi-supervised penalized output kernel regression for link prediction. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

S. Bucak, R. Jin, and A.K. Jain. Multiple kernel learning for visual object recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1354–1369, 2014.

A. Caponnetto, M. Pontil, C.Micchelli, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 9:1615–1646, 2008.

C. Carmeli, E. De Vito, and A. Toigo. Vector-valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem. *Analysis and Applications*, 4:377–408, 2006.

M. Christoudias, R. Urtasun, and T. Darrell. Bayesian localized multiple kernel learning. *Univ. California Berkeley, Berkeley, CA*, 2009.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

F. Dinuzzo, C.S. Ong, P. Gehler, and G. Pillonetto. Learning output kernels with block coordinate descent. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

T. Evgeniou, M. Pontil, and C.A. Micchelli. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594 –611, 2006.

D. Figueira, L. Bazzani, H.Q. Minh, M. Cristani, A. Bernardino, and V. Murino. Semi-supervised multi-feature learning for person re-identification. In *Proceedings of the IEEE International Conference on Advanced Video and Signal-based Surveillance (AVSS)*, 2013.

G.E. Forsythe and G. Golub. On the stationary values of a second-degree polynomial on the unit sphere. *Journal of the Society for Industrial & Applied Mathematics*, 13(4): 1050–1068, 1965.

W. Gander. Least squares with a quadratic constraint. *Numerische Mathematik*, 36:291–307, 1981.

P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009.

G. Golub and U. von Matt. Quadratically constrained least squares and quadratic problems. *Numerische Mathematik*, 59:561–580, 1991.

K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

S. Hill and A. Doucet. A framework for kernel-based multi-category classification. *Journal of Artificial Intelligence Research*, 30(1):525–564, 2007.

H. Kadri, A. Rabaoui, P. Preux, E. Duflos, and A. Rakotomamonjy. Functional regularized least squares classification with operator-valued kernels. In *Proceedings of the International Conference on Machine Learning(ICML)*, 2011.

H. Kadri, S. Ayache, C. Capponi, S. Koo, F.-X. Dup, and E. Morvant. The multi-task learning view of multimodal data. In *Proceedings of the Asian Conference on Machine Learning (ACML)*, 2013.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99:67–81, 2004.

Y. Luo, D. Tao, C. Xu, D. Li, and C. Xu. Vector-valued multi-view semi-supervised learning for multi-label image classification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2013a.

Y. Luo, D. Tao, C. Xu, C. Xu, H. Liu, and Y. Wen. Multiview vector-valued manifold regularization for multilabel image classification. *IEEE Transactions on Neural Networks and Learning Systems*, 24(5):709–722, 2013b.

C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17:177–204, 2005.

H. Q. Minh and V. Sindhwani. Vector-valued manifold regularization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

H.Q. Minh, L. Bazzani, and V. Murino. A unifying framework for vector-valued manifold regularization and multi-view learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

Y. Mroueh, T. Poggio, L. Rosasco, and J.-J. Slotine. Multiclass learning with simplex coding. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

M-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

M-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, 2008.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in kernel methods*, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.

A.S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2014.

M. Reisert and H. Burkhardt. Learning equivariant functions with matrix valued kernels. *Journal of Machine Learning Research*, 8:385–408, 2007.

G. Roffo, M. Cristani, L. Bazzani, H.Q. Minh, and V. Murino. Trusting Skype: Learning the way people chat for fast user recognition and verification. In *International Conference on Computer Vision Workshops (ICCVW)*, 2013.

D. Rosenberg, V. Sindhwani, P. Bartlett, and P. Niyogi. A kernel for semi-supervised learning with multi-view point cloud regularization. *IEEE Signal Processing Magazine*, 26(5):145–150, 2009.

M. J. Saberian and N. Vasconcelos. Multiclass boosting: Theory and algorithms. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

B. Schölkopf and A. Smola. *Learning with kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, Cambridge, 2002.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

V. Sindhwani and D. Rosenberg. An RKHS for multi-view learning and manifold co-regularization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

V. Sindhwani, H.Q. Minh, and A.C. Lozano. Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and granger causality. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013.

S. Sun. Multi-view Laplacian support vector machines. In *Proceedings of the International Conference on Advanced Data Mining and Applications (ADMA)*, 2011.

A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 dataset. Technical report, California Institute of Technology, 2011. URL `http://www.vision.caltech.edu/visipedia/CUB-200-2011.html`.

G. Wahba. Practical approximate solutions to linear operator equations when the data are noisy. *SIAM Journal on Numerical Analysis*, 14(4):651–667, 1977.

J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the European Symposium on Artificial Neural Networks (ESANN)*, 1999.

T.T. Wu and K. Lange. Multicategory vertex discriminant analysis for high-dimensional data. *The Annals of Applied Statistics*, 4(4):1698–1721, 2010.

F. Yan, J. Kittler, K. Mikolajczyk, and A. Tahir. Non-sparse multiple kernel Fisher discriminant analysis. *Journal of Machine Learning Research*, 13:607–642, 2012.

J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao. Group-sensitive multiple kernel learning for object categorization. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009.

M. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

H. Zhang, Y. Xu, and Q. Zhang. Refinement of operator-valued reproducing kernels. *Journal of Machine Learning Research*, 13:91–136, Jan 2012.

Journal of Machine Learning Research 17 (2016) 1-41 Submitted 4/14; Revised 7/15; Published 4/16

Quantifying Uncertainty in Random Forests via Confidence Intervals and Hypothesis Tests

Lucas Mentch LKM54@CORNELL.EDU
Giles Hooker GILES.HOOKER@CORNELL.EDU
Department of Statistical Science
Cornell University
Ithaca, NY 14850, USA

Editor: Bin Yu

Abstract

This work develops formal statistical inference procedures for predictions generated by supervised learning ensembles. Ensemble methods based on bootstrapping, such as bagging and random forests, have improved the predictive accuracy of individual trees, but fail to provide a framework in which distributional results can be easily determined. Instead of aggregating full bootstrap samples, we consider predicting by averaging over trees built on subsamples of the training set and demonstrate that the resulting estimator takes the form of a U-statistic. As such, predictions for individual feature vectors are asymptotically normal, allowing for confidence intervals to accompany predictions. In practice, a subset of subsamples is used for computational speed; here our estimators take the form of incomplete U-statistics and equivalent results are derived. We further demonstrate that this setup provides a framework for testing the significance of features. Moreover, the internal estimation method we develop allows us to estimate the variance parameters and perform these inference procedures at no additional computational cost. Simulations and illustrations on a real data set are provided.

Keywords: trees, u-statistics, bagging, subbagging, random forests

1. Introduction

This paper develops tools for performing formal statistical inference for predictions generated by a broad class of methods developed under the algorithmic framework of data analysis. In particular, we focus on ensemble methods—combinations of many individual, frequently tree-based, prediction functions—which have played an important role. We present a variant of bagging and random forests, both initially introduced by Breiman (1996, 2001b), in which base learners are built on randomly chosen subsamples of the training data and the final prediction is taken as the average over the individual outputs. We demonstrate that this fits into the statistical framework of U-statistics, which were shown to have minimum variance by Halmos (1946) and later demonstrated to be asymptotically normal by Hoeffding (1948). This allows us to demonstrate that under weak regularity conditions, predictions generated by these subsample ensemble methods are asymptotically normal. We also provide a method to consistently estimate the variance in the limiting distribution without increasing the computational cost so that we may produce confidence intervals and formally test feature significance in practice. Though not the focus of this paper, it is worth noting that this subbagging procedure—suggested by Andonova et al. (2002) for use in model selection—was shown by Zaman and Hirose (2009) to outperform traditional bagging in many situations.

We consider a general supervised learning framework in which an outcome $Y \in \mathbb{R}$ is predicted as a function of d features $\boldsymbol{X} = (X_1, ..., X_d)$ by the function $\mathbb{E}[Y|\boldsymbol{X}] = F(\boldsymbol{X})$. We also allow binary classification so long as the model predicts the probability of success, as opposed to a majority vote, so that the prediction remains real valued. Additionally, we assume a training set $\{(\boldsymbol{X}_1, Y_1), ..., (\boldsymbol{X}_n, Y_n)\}$ consisting of n independent examples from the process that is used to produce the prediction function $\hat{F}$. Throughout the remainder of this paper, we implicitly assume that the dimension of the feature space d remains fixed, though nothing in the theory provided prohibits a growing number of features so long as our other explicit conditions on the statistical behavior of trees are met.

Statistical inference proceeds by asking the counterfactual question, "What would our results look like if we regenerated these data." That is, if a new training set was generated and we reproduced $\hat{F}$, how different might we expect the predictions to be? To illustrate, consider the hypothesis that the feature X_1 does not contribute to the outcome at any point in the feature space:

$$H_0 : \exists\, F_1 \;\; s.t. \;\; F(x_1, ..., x_d) = F_1(x_2, ..., x_d) \quad \forall\, (x_1, ..., x_d) \in \mathcal{X}$$

A formal statistical test begins by calculating a test statistic $t_0 = t((\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n))$ and asks, "If our data was generated according to H_0 and we generated a new training set and recalculated t, what is the probability that this new statistic would be larger than t_0?" That is, we are interested in estimating $P(t > t_0|H_0)$. In most fields, a probability of less than 0.05 is considered sufficient evidence to reject the assumption that the data was generated according to H_0. Of course, a 0.05 chance can be obtained by many methods (tossing a biassed coin, for example) so we also seek a statistic t such that when H_0 is false, we are likely to reject. This probability of correctly rejecting H_0 is known as the power of the test, with more powerful tests clearly being more useful.

Here we propose to conduct the above test by comparing predictions generated by $\hat{F}$ and $\hat{F}_1$. Before doing so however, we consider the simpler hypothesis involving the value of a prediction:

$$H_0' : F(x_1, \ldots, x_p) = f_0.$$

Though often of less scientific importance, hypotheses of this form allow us to generate confidence intervals for predictions. These intervals are defined to be those values of f_0 for which we do not have enough evidence to reject H_0'. In practice, we choose f_0 to be the prediction $\hat{F}(x_1, ..., x_p)$ generated by the ensemble method in order to provide a formalized notion of plausible values of the prediction, which is, of course, of genuine interest. Our results begin here because the statistical machinery we develop will provide a distribution for the values of the prediction. This allows us to address H_0', after which we can combine these tests to address hypotheses like H_0.

Although this form of statistical analysis is ubiquitous in scientific literature, it is worthwhile contrasting this form of analysis with an alternative based on probably approximately correct (PAC) theory, as developed by Vapnik and Chervonenkis (1971) and Valiant (1984).

PAC theory provides a uniform bound on the difference between the true error and observed training error of a particular estimator, also referred to as a hypothesis. In this framework, $err(F)$ is some error of the function F which is estimated by $\widehat{err}(F)$ based on the data. A bound is then found for $P(\sup_{F\in\mathcal{F}}|\widehat{err}(F)-err(F)|>\epsilon)$ where $\mathcal{F}$ is some class of functions that includes $\hat{F}$. Since this bound is uniform over $\mathcal{F}$, it applies to $\hat{F}$ and we might think of comparing $\widehat{err}(\hat{F})$ with $\widehat{err}(\hat{F}_1)$ using such bounds. While appealing, these bounds provide the accuracy of our estimate of $err(\hat{F})$ but do not account for how the true $err(\hat{F})$ might change when $\hat{F}$ is reproduced with new training data. The uniformity of these bounds could be used to account for the uncertainty in $\hat{F}$ if it is chosen to minimize $\widehat{err}(F)$ over $\mathcal{F}$, but this is not always the case, for example, when using tree-based methods. We also expect the same uniformity to make PAC bounds conservative, thereby resulting in tests with lower power than those we develop.

Our analysis relies on the structure of subsample-based ensemble methods, specifically making use of classic U-statistic theory. These estimators have a long history (see, for example, original work by Kendall (1938) or Wilcoxon (1945), or Lee (1990) which has a modern overview), frequently focussed on rank-based non-parametric tests, and have been shown to have an asymptotically normal sampling distribution by Hoeffding (1948). Our application to subsample ensembles requires the extension of these results to some new cases as well as methods to estimate the asymptotic variance, both of which we provide.

U-statistics have traditionally been employed in the context of statistical parameter estimation. From this classical statistical perspective, we treat ensemble-tree methods like bagging and random forests as estimators and thus the limiting distributions and inference procedures we develop are with respect to the expected prediction generated by the ensemble. That is, given a particular prediction point $\boldsymbol{x}^*$, our limiting normal distributions are centered at the expected ensemble-based prediction at $\boldsymbol{x}^*$ and not necessarily $F(\boldsymbol{x}^*)$. Such forms of distributional analysis are common in other nonparametric regression settings—see Eubank (1999) Section 4.8, for example. More details on appropriate interpretations of the results are provided throughout the paper, in particular in Section 4.1.

In order to claim that the inference procedures proposed here are asymptotically valid for $F(\boldsymbol{x}^*)$, the ensemble must consistently predict $F(\boldsymbol{x}^*)$ at a rate of $\sqrt{n}$ or faster. Though this is the case for many classical estimators, establishing fast uniform rates of convergence for tree-based ensembles has proven extremely difficult. Breiman et al. (1984) discuss consistency of general partition-type models in the final chapter of their seminal book in the context of both classification and regression. Biau et al. (2008) restrict their attention to classification, but prove consistency of certain idealized bagging and random forest estimators, provided the individual trees are consistent. This paper also discusses a more general version of bagging, where the samples used to construct individual base learners may be proper subsamples of the training set taken with replacement as opposed to full bootstrap samples, so as to include the subbagging approach. Biau (2012) further examines the consistency of random forests and investigates their behavior in the presence of a sparse feature space. Recently, Denil et al. (2013) proved consistency for a mathematically tractable variant of random forests and in some cases, achieved empirical performance on par with the original random forest procedure suggested by Breiman. Zhu et al. (2015) prove consistency for their Reinforcement Learning Trees, where embedded random forests are used to decide splitting variables, and achieve significant improvements in empirical MSE for some data

sets. However, no rates of convergence have been developed that could be applied to analyze the ensemble methods we consider here.

Beyond these consistency efforts, mathematical analyses of ensemble learners has been somewhat limited. Sexton and Laake (2009) propose estimating the standard error of bagged trees and random forests using jackknife and bootstrap estimators. Recently, Wager et al. (2014) proposed applying the jackknife and infinitesimal jackknife procedures introduced by Efron (2014) for estimating standard errors in random forest predictions. Chipman et al. (2010) have received significant attention for developing *BART*, a Bayesian "sum-of-trees" statistical model for the underlying regression function that allows for pointwise posterior inference throughout the feature space as well as estimates for individual feature effects. Recently, Bleich et al. (2014) extended the BART approach by suggesting a permutation-based approach for determining feature relevance and by introducing a procedure to allow variable importance information to be reflected in the prior.

The layout of this paper is as follows: we demonstrate in Section 2 that ensemble methods based on subsampling can be viewed as U-statistics. In Section 3 we provide consistent estimators of the limiting variance parameters so that inference may be carried out in practice. Inference procedures, including a test of significance for features, are discussed in Section 4. Simulations illustrating the limiting distributions and inference procedures are provided in Section 5 and the inference procedures are applied to a real data set provided by Cornell University's Lab of Ornithology in Section 6.

2. Ensemble Methods as U-statistics

We begin by introducing the subbagging and subsampled random forest procedures that result in estimators in the form of U-statistics. In both cases, we provide an algorithm to make the procedure explicit.

2.1 Subbagging

We begin with a brief introduction to U-statistics; see Lee (1990) for a more thorough treatment. Let $Z_1,...,Z_n \overset{iid}{\sim} F_{Z,\theta}$ where θ is the parameter of interest and suppose that there exists an unbiased estimator h of θ that is a function of $k\leq n$ arguments. Then we can write

$$\theta = \mathbb{E}h(Z_1,...,Z_k)$$

and without loss of generality, we may further assume that h is permutation symmetric in its arguments since any given h may be replaced by an equivalent permutation symmetric version. The minimum variance unbiased estimator for θ is given by

$$U_n = \frac{1}{\binom{n}{k}}\sum_{(i)} h(Z_{i_1},...,Z_{i_k}) \tag{1}$$

where the sum is taken over all $\binom{n}{k}$ subsamples of size k and is referred to as a U-statistic with kernel h of rank k. When both the kernel and rank remain fixed, Hoeffding (1948) showed that these statistics are asymptotically normal with limiting variance $\frac{k^2}{n}\zeta_{1,k}$ where

$$\zeta_{1,k} = cov(h(Z_1, ..., Z_k), h(Z_1, Z_2^{'}, ..., Z_k^{'})) \tag{2}$$

and $Z_2^{'}, ..., Z_k^{'} \overset{iid}{\sim} F_{Z,\theta}$. The 1 in the subscript comes from the fact that there is 1 example in common between the two subsamples. In general, $\zeta_{c,k}$ denotes a covariance in the form of (2) with c examples in common.

Given infinite computing power and a consistent estimate of $\zeta_{1,k}$, Hoeffding's original result is enough to produce a subbagging procedure with asymptotically normal predictions. Suppose that as our training set, we observe $Z_1 = (\boldsymbol{X}_1, Y_1), ..., Z_n = (\boldsymbol{X}_n, Y_n) \overset{iid}{\sim} F_{\boldsymbol{X},Y}$ where $\boldsymbol{X} = (X_1, ..., X_d)$ is a vector of features and $Y \in \mathbb{R}$ is the response. Fix $k \leq n$ and let $(\boldsymbol{X}_{i_1}, Y_{i_1}), ..., (\boldsymbol{X}_{i_k}, Y_{i_k})$ be a subsample of the training set. Given a feature vector $\boldsymbol{x}^* \in \mathcal{X}$ where we are interested in making a prediction, we can write the prediction at $\boldsymbol{x}^*$ generated by a tree that was built using the subsample $(\boldsymbol{X}_{i_1}, Y_{i_1}), ..., (\boldsymbol{X}_{i_k}, Y_{i_k})$ as a function $T_{\boldsymbol{x}^*}$ from $(\mathcal{X} \times \mathbb{R}) \times \cdots \times (\mathcal{X} \times \mathbb{R})$ to $\mathbb{R}$. Taking all $\binom{n}{k}$ subsamples, building a tree and predicting at $\boldsymbol{x}^*$ with each, we can write our final subbagged prediction at $\boldsymbol{x}^*$ as

$$b_n(\boldsymbol{x}^*) = \frac{1}{\binom{n}{k}} \sum_{(i)} T_{\boldsymbol{x}^*}((\boldsymbol{X}_{i_1}, Y_{i_1}), ..., (\boldsymbol{X}_{i_k}, Y_{i_k})). \tag{3}$$

by averaging the $\binom{n}{k}$ tree-based predictions. Treating each ordered pair as one of k inputs into the function $T_{\boldsymbol{x}^*}$, the estimator in (3) is in the form of a U-statistic since tree-based estimators produce the same predictions independent of the order of the training data. Thus, provided the distribution of predictions at $\boldsymbol{x}^*$ has a finite second moment and $\zeta_{1,k} > 0$, the distribution of subbagged predictions at $\boldsymbol{x}^*$ is asymptotically normal. Note that in this context, $\zeta_{1,k}$ is the covariance between predictions at $\boldsymbol{x}^*$ generated by trees trained on data sets with 1 sample in common.

Of course, building $\binom{n}{k}$ trees is compuationally infeasible for even moderately sized training sets and an obvious substantial improvement in computationally efficiency can be achieved by building and averaging over only $m_n < \binom{n}{k}$ trees. In this case, the estimator in (3), appropriately scaled, is called an *incomplete* U-statistic. When the m_n subsamples are selected uniformly at random with replacement from the $\binom{n}{k}$ possibilities, the resulting incomplete U-statistic remains asymptotically normal; see Janson (1984) or Lee (1990) page 200 for details.

Though more computationally efficient, there remains a major shortcomming with this approach: the number of samples used to build each tree, k, remains fixed as $n \to \infty$. We would instead like k to grow with n so that trees can be grown to a greater depth, thereby presumably producing more accurate predictions. Incorporating this, our estimator becomes

$$b_{n,k_n,m_n}(\boldsymbol{x}^*) = \frac{1}{m_n} \sum_{(i)} T_{\boldsymbol{x}^*,k_n}((\boldsymbol{X}_{i_1}, Y_{i_1}), ..., (\boldsymbol{X}_{i_{k_n}}, Y_{i_{k_n}})). \tag{4}$$

Statistics of this form were discussed by Frees (1989) and called *Infinite Order* U-statistics (IOUS) in the complete case, when $m_n = \binom{n}{k_n}$, and *resampled* statistics in the incomplete case. Specifically, Frees considers the situation where, given an *i.i.d* sample $Z_1, Z_2, ...$ and kernel h_{k_n}, $\lim_{n\to\infty} h_{k_n}(Z_{i_1}, ..., Z_{i_{k_n}}) = h(Z_1, Z_2, ...)$ and $\theta = \mathbb{E}h(Z_1, Z_2, ...)$

and goes on to develop sufficient conditions for consistency and asymptotic normality whenever m_n grows faster than n. In contrast, the theorem below introduces a central limit theorem for estimators of the same form as in (4) but with respect to their individual means $\mathbb{E}b_{n,k_n,m_n}(\boldsymbol{x}^*)$ and covers all possible growth rates of m_n with respect to n. In this context, only minimal regularity conditions are required for asymptotic normality. We begin with an assumption on the distribution of estimates for the general U-statistic case.

Condition 1: *Let $Z_1, Z_2, ... \overset{iid}{\sim} F_Z$ with $\theta_{k_n} = \mathbb{E}h_{k_n}(Z_1, ..., Z_{k_n})$ and define $h_{1,k_n}(z) = \mathbb{E}h_{k_n}(z, Z_2, ..., Z_{k_n}) - \theta_{k_n}$. Then for all $\delta > 0$,*

$$\lim_{n\to\infty} \frac{1}{\zeta_{1,k_n}} \int_{|h_{1,k_n}(Z_1)| \geq \delta\sqrt{n\zeta_{1,k_n}}} h_{1,k_n}^2(Z_1) dP = 0.$$

This condition serves to control the tail behavior of the predictions and allows us to satisfy the Lindeberg condition needed to obtain part *(i)* of Theorem 1 below.

Theorem 1 *Let $Z_1, Z_2, ... \overset{iid}{\sim} F_Z$ and let U_{n,k_n,m_n} be an incomplete, infinite order U-statistic with kernel h_{k_n} that satisfies Condition 1. Let $\theta_{k_n} = \mathbb{E}h_{k_n}(Z_1, ..., Z_{k_n})$ such that $\mathbb{E}h_{k_n}^2(Z_1, ..., Z_{k_n}) \leq C < \infty$ for all n and some constant C, and let $\lim \frac{n}{m_n} = \alpha$. Then as long as $\lim \frac{k_n}{\sqrt{n}} = 0$ and $\lim \zeta_{1,k_n} \neq 0$,*

(i) if $\alpha = 0$, then $\frac{\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{k_n^2 \zeta_{1,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$.

(ii) if $0 < \alpha < \infty$, then $\frac{\sqrt{m_n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{\frac{k_n^2}{\alpha}\zeta_{1,k_n} + \zeta_{k_n,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$.

(iii) if $\alpha = \infty$, then $\frac{\sqrt{m_n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{\zeta_{k_n,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$.

Condition 1, though necessary for the general U-statistic setting, is a bit obscure. However, in our regression context, when the regression function is bounded and the errors have exponential tails, a more intuitive Lipschitz-type condition given in Proposition 1 is sufficient. Though stronger than necessary, this alternative condition allows us to satisfy the Lindeberg condition and is reasonable to expect of any supervised learning method.

Proposition 1: *For a bounded regression function F, if there exists a constant c such that for all $k_n \geq 1$,*

$$\begin{aligned}\Big|h((\boldsymbol{X}_1, Y_1), ..., (\boldsymbol{X}_{k_n}, Y_{k_n}), (\boldsymbol{X}_{k_n+1}, Y_{k_n+1})) - h((\boldsymbol{X}_1, Y_1), ..., (\boldsymbol{X}_{k_n}, Y_{k_n}), (\boldsymbol{X}_{k_n+1}, Y_{k_n+1}^*))\Big| \\ \leq c\Big|Y_{k_n+1} - Y_{k_n+1}^*\Big|\end{aligned}$$

where $Y_{k_n+1} = F(\boldsymbol{X}_{k_n+1}) + \epsilon_{k_n+1}$, $Y_{k_n+1}^* = F(\boldsymbol{X}_{k_n+1}) + \epsilon_{k_n+1}^*$, and where ϵ_{k_n+1} and $\epsilon_{k_n+1}^*$ are i.i.d. with exponential tails, then Condition 1 is satisfied.

A number of important aspects of these results are worth pointing out. First, note from Theorem 1 that the trees are built with subsamples that are approximately square root of the size of the full training set. This condition is not necessary for the proof, but ensures that the variance of the U-statistic in part (i) converges to 0 as is typically the case in central limit theorems. By maintaining this relatively small subsample size, we can build many more trees and maintain a procedure that is computationally equivalent to traditional bagging based on full bootstrap samples. Also note that no particular assumptions are placed on the dimension d of the feature space; the number of features may grow with n so long as the stated conditions remain satisfied.

The final condition of Theorem 1, that $\lim \zeta_{1,k_n} \neq 0$, though not explicitly controllable, should be easily satisfied in many cases. As an example, suppose that the terminal node size is bounded by $\mathcal{T}$ so that trees built with larger training sets are grown to greater depths. Then if the form of the response is $Y = F(\boldsymbol{X}) + \epsilon$ where ϵ has variance σ^2, ζ_{1,k_n} will be bounded below by $\sigma^2/\mathcal{T}$. Finally, note that the assumption of exponential tails on the distribution of regression errors in Proposition 1 is stronger than necessary. Indeed, so long as $k_n = o(\sqrt{n})$, we need only insist that $nP\left(\left|\epsilon\right| > \sqrt{n}\right) \to 0$.

The proofs of Theorem and Proposition 1 are provided in Appendix A. The subbagging algorithm that produces asymptotically normal predictions at each point in the feature space is provided in Algorithm 1.

Algorithm 1 Subbagging

Load training set
Select size of subsamples k_n and number of subsamples m_n
for i in 1 to m_n **do**
  Take subsample of size k_n from training set
  Build tree using subsample
  Use tree to predict at $\boldsymbol{x}^*$
end for
Average the m_n predictions to get final estimate $b_{n,k_n,m_n}(\boldsymbol{x}^*)$

Note that this procedure is precisely the original bagging algorithm suggested by Breiman, but with proper subsamples used to build trees instead of full bootstrap samples. In Section 3, we provide consistent estimators for the limiting variance parameters in Theorem 1 so that we may carry out inference in practice.

We would also like to acknowledge similar work currently in progress by Wager (2014). Wager builds upon the potential nearest neighbor framework introduced by Lin and Jeon (2006) and seeks to provide a limiting distribution for the case where many trees are used in the ensemble, roughly corresponding to our result (i) in Theorems 1 and 2. The author considers only an idealized class of trees based on the assumptions in Meinshausen (2006) as well as additional *honesty* and *regularity* conditions that allow k_n to grow at a faster rate, and demonstrates that when many Monte Carlo samples are employed, the infinitesimal jackknife estimator of variance is consistent and predictions are asymptotically normal. This estimator has roughly the same computational complexity as those we propose in Section 3 and should scale well subject to some additional bookkeeping. In contrast, the theory we provide here takes into account all possible rates of Monte Carlo sampling via the

three cases discussed in Theorems 1 and 2 and we provide a consistent means for estimating each corresponding variance.

2.2 Random Forests

The distributional results described above for subbagging do not insist on a particular tree building method. So long as the trees generate predictions that satisfy minimal regularity conditions, the experimenter is free to use whichever building method is preferred. The subbagging procedure does, however, require that each tree in the ensemble is built according to the same method.

This insistence on a uniform, non-randomized building method is in contrast with random forests. The original random forests procedure suggested by Breiman (2001b) dictates that at each node in each tree, the split may occur on only a randomly selected subset of features. Thus, we may think of each tree in a random forest as having an additional randomization parameter ω that determines the eligible features that may potentially be split at each node. In a general U-statistic context, we can write this *random kernel* U-statistic as

$$U_{\omega;n,k_n,m_n} = \frac{1}{m_n}\sum_{(i)} h_{k_n}^{(\omega_i)}(Z_{i_1},...,Z_{i_{k_n}}) \tag{5}$$

so that we can write a random forest estimator as

$$r_{n,k_n,m_n}(\boldsymbol{x}^*) = \frac{1}{m_n}\sum_{(i)} T_{\boldsymbol{x}^*,k_n}^{(\omega_i)}((\boldsymbol{X}_{i_1},Y_{i_1}),...,(\boldsymbol{X}_{i_{k_n}},Y_{i_{k_n}})).$$

Due to this additional randomness, random forests and random kernel U-statistics in general do not fit within the framework developed in the previous section so we develop new theory for this expanded class. Suppose $\omega_1,...,\omega_{m_n} \overset{iid}{\sim} F_\omega$ and that these randomization parameters are selected independently of the original sample $Z_1,...,Z_n$. Consider the statistic

$$U^*_{\omega;n,k_n,m_n} = \mathbb{E}_\omega\left(\frac{1}{m_n}\sum_{(i)} h_{k_n}^{(\omega_i)}(Z_{i_1},...,Z_{i_{k_n}})\right)$$

so that $U^*_{\omega;n,k_n,m_n} = \mathbb{E}_\omega U_{\omega;n,k_n,m_n}$. Taking the expectation with respect to ω, the kernel becomes fixed and hence $U^*_{\omega;n,k_n,m_n}$ conforms to the non-random kernel U-statistic theory. Thus, $U^*_{\omega;n,k_n,m_n}$ is asymptotically normal in both the complete and incomplete cases, as well as in the complete and incomplete infinite order cases, by Theorem 1. Given this asymptotic normality of $U^*_{\omega;n,k_n,m_n}$, in order to retain asymptotic normality of the corresponding random kernel version, we need only show that

$$\sqrt{n}(U^*_{\omega;n,k_n,m_n} - U_{\omega;n,k_n,m_n}) \xrightarrow{P} 0.$$

We make use of this idea in the proof of the following theorem, which is provided in Appendix A.

Theorem 2 *Let $U_{\omega;n,k_n,m_n}$ be a random kernel U-statistic of the form defined in equation (5) such that $U^*_{\omega;n,k_n,m_n}$ satisfies Condition 1 and suppose that $\mathbb{E}h^2_{k_n}(Z_1,...,Z_{k_n}) < \infty$ for all n, $\lim \frac{k_n}{\sqrt{n}} = 0$, and $\lim \frac{n}{m_n} = \alpha$. Then, letting β index the subsamples, so long as $\lim \zeta_{1,k_n} \neq 0$ and*

$$\lim_{n\to\infty} \mathbb{E}\left(h^{(\omega)}_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}}) - \mathbb{E}_\omega h^{(\omega)}_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}})\right)^2 \neq \infty,$$

$U_{\omega;n,k_n,m_n}$ is asymptotically normal and the limiting distributions are the same as those provided in Theorem 1.

Note that the variance parameters ζ_{1,k_n} and ζ_{k_n,k_n} in the context of random kernel U-statistics are still defined as the covariance between estimates generated by the (now random) kernels. Thus, in the specific context of random forests, these variance parameters correspond to the covariance between predictions generated by trees, but each tree is built according to its own randomization parameter ω and this covariance is taken over ω as well. The final condition of Theorem 2 that

$$\lim_{n\to\infty} \mathbb{E}\left(h^{(\omega)}_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}}) - \mathbb{E}_\omega h^{(\omega)}_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}})\right)^2 \neq \infty,$$

simply ensures that the randomization parameter ω does not continually pull predictions from the same subsample further apart as $n \to \infty$. This condition is satisfied, for example, if the response Y is bounded and should also be easily satisfied for any reasonable implementation of random forests.

The subsampled random forest algorithm that produces asymptotically normal predictions is provided in Algorithm 2. As with subbagging, this subsampled random forest algorithm is exactly a random forest with subsamples used to build trees instead of full bootstrap samples.

Algorithm 2 Subsampled Random Forest

Load training set
Select size of subsamples k_n and number of subsamples m_n
for i in 1 to m_n **do**
  Select subsample of size k_n from training set
  Build tree based on randomization parameter ω_i
  Use this tree to predict at $\boldsymbol{x}^*$
end for
Average the m_n predictions to obtain final estimate $r_{n,k_n,m_n}(\boldsymbol{x}^*)$

Another random-forest-type estimator based on a crossed design that results in an infinite order *generalized* U-statistic is provided in Appendix B. However, the above formulation most resembles Breiman's original procedure and is more computationally feasible than the method mentioned in Appendix B, so we consider only this random kernel version of random forests in the simulations and other work that follows.

3. Estimating the Variance

The limiting distributions provided in Theorem 1 depend on the unknown mean parameter $\theta_{k_n} = \mathbb{E}U_{n,k_n,m_n}$ as well as the unknown variance parameters ζ_{1,k_n} and ζ_{k_n,k_n}. In order for us to be able to use these distributions for statistical inference in practice, we must establish consistent estimators of these parameters. It is obvious that we can use the sample mean—i.e. the prediction from our ensemble—as a consistent estimate of θ_{k_n}, but determining an appropriate variance estimate is less straightforward.

In equation (2) of the previous section, we defined ζ_{c,k_n} as the covariance between two instances of the kernel with c shared arguments, so the sample covariance between predictions may serve as a consistent estimator for both ζ_{1,k_n} and ζ_{k_n,k_n}. However, in practice we find that this often results in estimates close to 0, which may then lead to an overall negative variance estimate.

It is not difficult to show - see Lee (1990) page 11 for details - that an equivalent expression for ζ_{c,k_n} is given by

$$\zeta_{c,k_n} = var\Big(\mathbb{E}\big(h_{k_n}(Z_1,...,Z_{k_n}) \mid Z_1 = z_1,...,Z_c = z_c\big)\Big).$$

To estimate ζ_{c,k_n} for our tree-based ensembles, we begin by selecting c observations $\tilde{\boldsymbol{z}}_1,...,\tilde{\boldsymbol{z}}_c$, which we refer to as initial fixed points, from the training set. We then select several subsamples of size k_n from the training set, each of which must include $\tilde{\boldsymbol{z}}_1,...,\tilde{\boldsymbol{z}}_c$, build a tree with each subsample, and record the mean of the predictions at $\boldsymbol{x}^*$. Let n_{MC} (MC for "Monte Carlo") denote the number of subsamples drawn so that this average is taken over n_{MC} predictions. We then repeat the process for $n_{\tilde{\boldsymbol{z}}}$ initial sets of fixed points and take our final estimate of ζ_{c,k_n} as the variance over the $n_{\tilde{\boldsymbol{z}}}$ final averages, yielding the estimator

$$\hat{\zeta}_{c,k_n} = var\left(\frac{1}{n_{MC}}\sum_{i=1}^{n_{MC}} T_{\boldsymbol{x}^*,k_n}(\mathcal{S}_{\tilde{\boldsymbol{z}}^{(1)},i})\ ,...,\ \frac{1}{n_{MC}}\sum_{i=1}^{n_{MC}} T_{\boldsymbol{x}^*,k_n}(\mathcal{S}_{\tilde{\boldsymbol{z}}^{(n_{\tilde{\boldsymbol{z}}})},i})\right)$$

where $\tilde{\boldsymbol{z}}^{(j)}$ denotes the j^{th} set of initial fixed points and $\mathcal{S}_{\tilde{\boldsymbol{z}}^{(j)},i}$ denotes the i^{th} subsample that includes $\tilde{\boldsymbol{z}}^{(j)}$ (which is used here as shorthand for the argument to the tree function $T_{\boldsymbol{x}^*,k_n}$). Now, since we assume that the orginal data in the training set is i.i.d., the random variables $\frac{1}{n_{MC}}\sum_{i=1}^{n_{MC}} T_{\boldsymbol{x}^*,k_n}(\mathcal{S}_{\tilde{\boldsymbol{z}}^{(1)},i})$ are also i.i.d. and since the sample variance is a U-statistic, $\hat{\zeta}_{c,k_n}$ is a consistent estimator. The algorithm for calculating $\hat{\zeta}_{1,k_n}$ is provided in Algorithm 3. Note that when $c = k_n$, each of the subsamples is identical so we need only use $n_{MC} = 1$ which simplifies the estimation procedure for ζ_{k_n,k_n}. The procedure for calculating $\hat{\zeta}_{k_n,k_n}$ is provided in Algorithm 4.

Choosing the values of $n_{\tilde{\boldsymbol{z}}}$ and n_{MC} will depend on the situation. The number of iterations required to accurately estimate the variance depends on a number of factors, including the tree building method and true underlying regression function. Of course, ideally these estimation parameters should be chosen as large as is computationally feasible. In our simulations, we find that in most cases, only a relatively small number of initial fixed point sets are needed, but many more Monte Carlo samples are often necessary for accurate estimation. In most cases, we used an n_{MC} of at least 500. Recall that because our trees are built with small subsamples, we can build correspondingly more trees at the same computational cost.

Algorithm 3 ζ_{1,k_n} Estimation Procedure

for i in 1 to $n_{\tilde{\boldsymbol{z}}}$ **do**
  Select initial fixed point $\tilde{\boldsymbol{z}}^{(i)}$
  for j in 1 to n_{MC} **do**
    Select subsample $\mathcal{S}_{\tilde{\boldsymbol{z}}^{(i)},j}$ of size k_n from training set that includes $\tilde{\boldsymbol{z}}^{(i)}$
    Build tree using subsample $\mathcal{S}_{\tilde{\boldsymbol{z}}^{(i)},j}$
    Use tree to predict at $\boldsymbol{x}^*$
  end for
  Record average of the n_{MC} predictions
end for
Compute the variance of the $n_{\tilde{\boldsymbol{z}}}$ averages

Algorithm 4 ζ_{k_n,k_n} Estimation Procedure

for i in 1 to $n_{\tilde{\boldsymbol{z}}}$ **do**
  Select subsample of size k_n from training set
  Build tree using subsample this subsample
  Use tree to predict at $\boldsymbol{x}^*$
end for
Compute the variance of the $n_{\tilde{\boldsymbol{z}}}$ predictions

3.1 Internal vs. External Estimation

The algorithms for producing the subbagged or subsampled random forest predictions as well as the above algorithms for estimating the variance parameters are all that is needed to perform statistical inference. We can begin with Algorithm 1 or 2 to generate the predictions, followed by Algorithms 3 and 4 to estimate the variance parameters ζ_{1,k_n} and ζ_{k_n,k_n}. This procedure of running these 3 algorithms seperately is what we will refer to as the *external* variance estimation method, since the the variance parameters are estimated outside of the orginal ensemble. By contrast, we could instead generate the predictions and estimate the variance parameters in one procedure by taking the mean and variance of the predictions generated by the trees used to estimate ζ_{1,k_n}. Algorithm 5 outlines the steps in this *internal* variance estimation method.

This internal variance estimation method is more computationally efficient and has the added benefit of producing variance estimates by simply changing the way in which the subsamples are selected. This means that we are able to obtain all parameter estimates we need to conduct inference at no greater computational cost than building the original ensemble. Although Theorems 1 and 2 dictate that the subsamples used in the ensemble be selected uniformly at random, we find that the additional correlation introduced by selecting the subsamples in this way and using the same subsamples to estimate all parameters does not affect the limiting distribution.

Algorithm 5 Internal Variance Estimation Method

for i in 1 to $n_{\tilde{\boldsymbol{z}}}$ **do**
  Select initial fixed point $\tilde{\boldsymbol{z}}^{(i)}$
  for j in 1 to n_{MC} **do**
    Select subsample $\mathcal{S}_{\tilde{\boldsymbol{z}}^{(i)},j}$ of size k_n from training set that includes $\tilde{\boldsymbol{z}}^{(i)}$
    Build tree using subsample $\mathcal{S}_{\tilde{\boldsymbol{z}}^{(i)},j}$
    Use tree to predict at $\boldsymbol{x}^*$ and record prediction
  end for
  Record average of the n_{MC} predictions
end for
Compute the variance of the $n_{\tilde{\boldsymbol{z}}}$ averages to estimate ζ_{1,k_n}
Compute the variance of all predictions to estimate ζ_{k_n,k_n}
Compute the mean of all predictions to estimate θ_{k_n}

4. Inference Procedures

In this section, we describe the inference procedures that may be carried out after performing the estimation procedures.

4.1 Confidence Intervals

In Section 2, we showed that predictions from subbagging and subsampled random forests are asymptotically normal and in Section 3 we provided consistent estimators for the parameters in the limiting normal distributions. Thus, given a training set, we can estimate the approximate distribution of predictions at any given feature vector of interest $\boldsymbol{x}^*$. To produce a confidence interval for predictions at $\boldsymbol{x}^*$, we need only estimate the variance parameters and take quantiles from the appropriate limiting distribution. Formally, our confidence interval is $[LB, UB]$ where the lower and upper bounds, LB and UB, are the $\alpha/2$ and $1-\alpha/2$ quantiles respectively of the normal distribution with mean $\hat{\theta}_{k_n}$ and variance $\frac{k_n^2}{\hat{\alpha}}\hat{\zeta}_{1,k_n} + \hat{\zeta}_{k_n,k_n}$ where $\hat{\zeta}_{1,k_n}$ and $\hat{\zeta}_{k_n,k_n}$ are the variance estimates and $\hat{\alpha} = n/m_n$. This limiting distribution is that given in result (ii) of Theorem 1 which is the distribution we recommend using in practice.

As mentioned in the introduction, these confidence intervals can also be used to address hypotheses of the form

$$H_0 : \theta_{k_n} = c$$
$$H_1 : \theta_{k_n} \neq c.$$

Formally, we can define the test statistic

$$t = \frac{\hat{\theta}_{k_n} - c}{sd(\hat{\theta}_{k_n})}$$

and reject H_0 if $|t|$ is greater than the the $1 - \frac{\alpha}{2}$ quantile of the standard normal. This corresponds to a test with type 1 error rate α so that $P[\text{reject } H_0 \mid H_0 \text{ true}] = P[\ |t| >$

$1 - \frac{\alpha}{2} \mid \theta_{k_n} = c] = \alpha$. However, this testing procedure is equivalent to simply checking whether c is within the calculated confidence interval: if c is in the confidence interval, then we fail to reject this hypothesis that the true mean prediction is equal to c, otherwise we reject.

Finally, recall that these confidence intervals are for the expected prediction θ_{k_n} and not necessarily for the true value of the underlying regression function $\theta = F(\boldsymbol{x}^*)$. If the tree building method employed is consistent so that $\theta_{k_n} \xrightarrow{P} \theta$, then as the sample size increases, the tree should be (on average) producing more accurate predictions, but in order to claim that our confidence intervals are asymptotically valid for θ, we need for this convergence to occur at rate of $\sqrt{n}$ or faster. However, in general, the rate of convergence will depend on not only the tree-building method and true underlying regression function, but also on the location of the prediction point within the feature space.

Note that Theorems 1 and 2 apply not only to tree-based ensembles, but to any estimator that can be written in the form of an infinite order U-statistic, as in equation (4). Some of these ensembles may be straightforward to analyze, but for others, such as random forest predictions near the edge of the feature space, it may be difficult to establish a universal rate of consistency. However, even when $\sqrt{n}$-consistency cannot be guaranteed, these intervals still provide valuable information not currently available with existing tools. In these cases, the confidence interval provides a reasonable range of values for where the prediction might fall if the ensemble was recomputed using a new training set; areas of the feature space where confidence intervals are relatively large indicate regions where the ensemble is particularly unstable.

Compare this, for example, to the standard approach of withholding some (usually small) portion of the training set and comparing predictions made at these hold-out points to the corresponding known responses. Such an approach provides some information as to the *accuracy* of the learner at specific locations throughout the feature space, but says nothing about the *stability* of these predictions. Thus, instead of relying only on measures of overall goodness-of-fit such as MSE or SSE, these intervals allow users to investigate prediction variability at particular points or regions and in this sense, can be seen as a measure of how much the accuracy of predictions at that point is due to chance.

4.2 Tests of Significance

The limiting distributions developed in Theorems 1 and 2 also allow us a way to test the significance of features. In many situations, data are recorded for a large number of features but a sparse true regression structure is suspected. Suppose that the training set consists of d features, $X_1, ..., X_d$ and consider a reduced set $\boldsymbol{X}^{(R)} \subset \{X_1, ..., X_d\}$. Let $\boldsymbol{x}_{\text{TEST}} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ be a set of feature vectors where we are interested in making predictions. Also, let g denote the function that maps feature vectors to their corresponding true mean prediction and let $g^{(R)}$ denote the same type of function that maps from the reduced feature space. That is, for a particular prediction point of interest $\boldsymbol{x}^*$, $g(\boldsymbol{x}^*)$ is the true mean prediction θ_{k_n} generated by trees built using the full feature space, and $g^{(R)}(\boldsymbol{x}^*)$ is the true mean prediction $\theta^{(R)}_{k_n}$ generated by trees that are only permitted to utilize features in the reduced set. Then, for each test point $\boldsymbol{x}_i \in \boldsymbol{x}_{\text{TEST}}$, we would like to know whether

$g(\boldsymbol{x}_i) = g^{(R)}(\boldsymbol{x}_i)$ so that we can determine the predictive influence of features not in $\boldsymbol{X}^{(R)}$. More formally, we would like to test the hypothesis

$$\begin{aligned} H_0 &: g(\boldsymbol{x}_i) = g^{(R)}(\boldsymbol{x}_i) \;\; \forall\, \boldsymbol{x}_i \in \boldsymbol{x}_{\text{TEST}} \\ H_1 &: g(\boldsymbol{x}_i) \neq g^{(R)}(\boldsymbol{x}_i) \;\; \text{for some } \boldsymbol{x}_i \in \boldsymbol{x}_{\text{TEST}}. \end{aligned} \tag{6}$$

Rejecting this null hypothesis means that a feature not in the reduced feature space $\boldsymbol{X}^{(R)}$ makes a significant contribution to the prediction at at least one of the test points.

To perform this test with a training set of size n, we take m_n subsamples, each of size k_n, and build a tree with each subsample. Denote these subsamples $S_1, ..., S_{m_n}$ and for a given feature vector $\boldsymbol{x}_i$, let $\hat{g}(\boldsymbol{x}_i)$ denote the average over the predictions at $\boldsymbol{x}_i$ generated from the m_n trees. Then, using the same subsamples $S_1, ..., S_{m_n}$, again build a tree with each, but using only those features in $\boldsymbol{X}^{(R)}$, and let $\hat{g}^{(R)}(\boldsymbol{x}_i)$ be the average prediction at $\boldsymbol{x}_i$ generated by these trees. Finally, define the difference function

$$\hat{D}(\boldsymbol{x}_i) = \hat{g}(\boldsymbol{x}_i) - \hat{g}^{(R)}(\boldsymbol{x}_i)$$

as the difference between the two ensemble predictions. Note that we can write

$$\begin{aligned} \hat{D}(\boldsymbol{x}_i) &= \hat{g}(\boldsymbol{x}_i) - \hat{g}^{(R)}(\boldsymbol{x}_i) \\ &= \frac{1}{m_n} \sum_{(j)} T_{\boldsymbol{x}_i, k_n}(S_j) - \frac{1}{m_n} \sum_{(j)} T^{(R)}_{\boldsymbol{x}_i, k_n}(S_j) \\ &= \frac{1}{m_n} \sum_{(j)} \left(T_{\boldsymbol{x}_i, k_n}(S_j) - T^{(R)}_{\boldsymbol{x}_i, k_n}(S_j) \right) \end{aligned}$$

so that this difference function is a U-statistic. Thus, if we have only a single test point of interest, $\hat{D}$ is asymptotically normal, so $\hat{D}^2$ is asymptotically χ^2_1 and we can use $\hat{D}^2$ as a test statistic.

However, it is more often the case that we have several test points of interest. In this case, define $\hat{\mathbb{D}}$ to be the vector of observed differences in the predictions

$$\hat{\mathbb{D}} = \left(\hat{D}(\boldsymbol{x}_1), ..., \hat{D}(\boldsymbol{x}_N) \right)$$

so that, provided a joint distribution exists with respect to Lebesgue measure, $\hat{\mathbb{D}}$ has a multivariate normal distribution with mean vector

$$\boldsymbol{\mu} = \left(g(\boldsymbol{x}_1) - g^{(R)}(\boldsymbol{x}_1), ..., g(\boldsymbol{x}_N) - g^{(R)}(\boldsymbol{x}_N) \right)^T$$

which we estimate with

$$\hat{\boldsymbol{\mu}} = \hat{\mathbb{D}}^T$$

as well as a covariance matrix Σ. This covariance matrix has parameters Σ_{1,k_n} and Σ_{k_n,k_n}, the multivariate analogues of ζ_{1,k_n} and ζ_{k_n,k_n}. Consistent estimators for these multivariate

parameters can be obtained by simply replacing the variance calculation in Algorithms 3 and 4 with a covariance. For clarity, the procedure for obtaining $\hat{\Sigma}_{1,k_n}$ is provided in Algorithm 6 in Appendix C.

Finally, combining these predictions to form a consistent estimator $\hat{\Sigma}$ we have that

$$\hat{\boldsymbol{\mu}}^T \hat{\Sigma}^{-1} \hat{\boldsymbol{\mu}} \sim \chi^2_N$$

under H_0. Thus, in order to test the hypothesis in (6), we compare the test statistic $\hat{\boldsymbol{\mu}}^T \hat{\Sigma}^{-1} \hat{\boldsymbol{\mu}}$ to the $1-\alpha$ quantile of the χ^2_N distribution to produce a test with type 1 error rate α. If our test statistic is larger than this critical value, we reject the null hypothesis.

4.3 Further Testing Procedures

This setup, though straightforward, may not always definitively decide the significance of features. In some cases, even randomly generated features that are unrelated to the response can be reported significant. Depending on the building method, tree-based algorithms may take advantage of additional randomness in features even when the particular values of those features do not directly contribute to the response. For this reason, we also recommend repeating the testing procedure by comparing predictions generated using the full data set to predictions generated by a data set with randomly generated values—commonly obtained by permuting the values in the training set—for the features not in the reduced feature set to test hypostheses of the form

$$H_0 : g(\boldsymbol{x}_i) = g^{(RAND)}(\boldsymbol{x}_i) \;\; \forall \; \boldsymbol{x}_i \in \boldsymbol{x}_{\text{TEST}}$$
$$H_1 : g(\boldsymbol{x}_i) \neq g^{(RAND)}(\boldsymbol{x}_i) \;\; \text{for some } \boldsymbol{x}_i \in \boldsymbol{x}_{\text{TEST}}.$$

The testing procedure remains exactly the same except that to calculate the second set of trees, we simply substitute the reduced training set for a training set with the same number of features, but with randomized values taking the place of the original values for the additional features. Rejecting this null hypothesis allows us to conclude that not only do the additional features not in the reduced training set make a significant contribution to the predictions, but that the contribution is significantly more than could be obtained simply by adding additional randomness.

There are also two additional tests that may be performed. First, we can test whether predictions generated by a training set with randomized values for the additional features are significantly different from predictions generated by the reduced feature set. If a significant difference is found, then the trees in the ensemble are making use of the additional randomness or possibly an accidental structure in the randomized features. As a final check, we can compare predictions generated by two training sets, each with randomized values for the features not in the reduced set. In the unlikely event that a significant difference is found between these predictions, it is again likely due to an accidental structure in the randomized values. Both of these tests can be performed in exactly the same fashion by substituting the appropriate training sets.

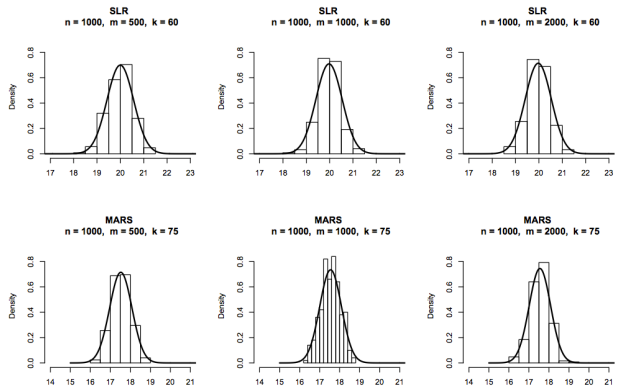


Figure 1: Histograms of subbagged predictions at $x_1 = 10$ in the SLR case (top row) and at $x_1 = \cdots = x_5 = 0.5$ in the MARS case (bottom row). The total sample size, number of subsamples, and size of each subsample are denoted by n,m, and k, respectively in the plot titles.

5. Simulations

We present here a small simulation study in order to illustrate the limiting distributions derived in Section 2 and also to demonstrate the inference procedures proposed in the previous section. We consider two different underlying regression functions:

1. $g(x_1) = 2x_1; \quad \mathcal{X} = [0, 20]$
2. $g(\boldsymbol{x}) = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.05)^2 + 10x_4 + 5x_5; \quad \mathcal{X} = [0, 1]^5$

The first function corresponds to simple linear regression (SLR) and was chosen for simplicity and ease of visualization. The second was initially considered by Friedman (1991) in development of the Multivariate Adaptive Regression Spline (MARS) procedure and was recently investigated by Biau (2012). In each case, features were selected uniformly at random from the feature spaces and responses were sampled from $g(\boldsymbol{x})+\epsilon$, where $\epsilon \overset{iid}{\sim} \mathcal{N}(0, 10)$, to form the training sets.

5.1 Limiting Distributions

We begin by illustrating the distributions of subbagged predictions. In the SLR case, predictions were made at $x_1 = 10$ and in the MARS case, predictions were made at $x_1 = \cdots = x_5 = 0.5$. The histograms of subbagged predictions are shown in Figure 1. Each histogram is comprised of 250 simulations.

For each histogram, the size of the training set n, number of subsamples m, and size of each subsample k, is provided in the title. Each tree in the ensembles was built using the

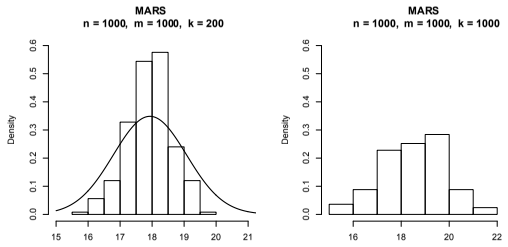


Figure 2: Histograms of subbagged predictions with larger subsample size k and full bootstrap samples. Predictions are made at $x_1 = \cdots = x_5 = 0.5$.

`rpart` function in `R`, with the additional restriction that at least 3 observations per node were needed in order for the algorithm to consider splitting on that node. Overlaying each histogram is the density obtained by estimating the parameters in the limiting distribution. In each case, we take the limiting distribution to be that given in result *(ii)* of Theorem 1; namely that the predictions are normally distributed with mean $\mathbb{E}b_{n,k,m}(\boldsymbol{x}^*)$ and variance $\frac{1}{\alpha}\frac{k^2}{m}\zeta_{1,k} + \frac{1}{m}\zeta_{k,k}$.

The mean $\mathbb{E}b_{n,k,m}(\boldsymbol{x}^*) = \theta_k$ was estimated as the empirical mean across the 250 subbagged predictions. To estimate $\zeta_{k,k}$, 5000 new subsamples of size k were selected and with each subsample, a tree was built and used to predict at $x_1 = 10$ and $\hat{\zeta}_{k,k}$ was taken as the empirical variance between these predictions. To estimate $\zeta_{1,k}$, we follow the procedure in Algorithm 3 with $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 1000$ in the SLR cases and with $n_{\tilde{\boldsymbol{z}}} = 250$ and $n_{MC} = 1000$ in the MARS cases. Note that since we are only interested in verifying the distributions of predictions, the variance parameters are estimated only once for each case and not for each ensemble.

It is worth noting that the same variance estimation procedure with $n_{\tilde{\boldsymbol{z}}} = 250$ and $n_{MC} = 250$ lead to an overestimate of the variance, so we reiterate that using a large n_{MC} seems to provide better results, even when $n_{\tilde{\boldsymbol{z}}}$ is relatively small. In each case, we use $\frac{n}{m}$ as a plug-in estimate for $\alpha = \lim \frac{n}{m}$. We also repeated this procedure and generated the distribution of predictions according to the internal variance estimation method described in Algorithm 5. Details and histograms are provided in Appendix D. These distributions appear to be the same as when the subsamples are selected uniformly at random, as in the external variance estimation method.

Note that the distributional results in Theorem 1 require $\lim \frac{k}{\sqrt{n}} = 0$, so in practice, the subsample size k should be small relative to n. In the above simulations, we choose k slightly larger than $\sqrt{n}$ and the distributions appear normally distributed with the correct limiting distribution. However, though this restriction on the growth rate of the subsample size is sufficient for asymptotic normality, it is perhaps not necessary. In our simulations, we found that ensembles built with larger k are still approximately normal, but begin to

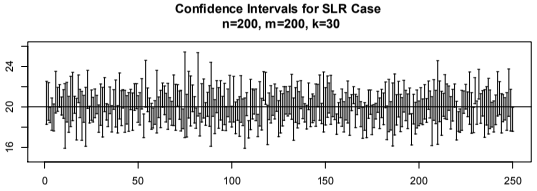


Figure 3: Confidence Intervals for subbagged predictions.

look increasingly further from normal as k increases. The histograms in Figure 2 show the distribution of subbagged predictions in the MARS case with $n = m = 1000$ and $k = 200$ and also with $n = m = k = 1000$ so that we are using full bootstrap samples to build the ensembles in the latter case. The parameters in the limiting distribution are estimated in exactly the same manner as with the smaller k for the case where $k = 200$. In the bootstrap case, we cannot follow our subbagging procedure exactly since the bootstrap samples used to build each tree in the ensemble must be taken with replacement, so we do not attempt to estimate the variance. These distributions look less normal and we begin to overestimate the variance in the case where $k = 200$.

5.2 Confidence Intervals

We move now to building confidence intervals for predictions and examine their coverage probabilities. We begin with the SLR case, with $n = 200$, $m = 200$, and $k = 30$ and as above, predict at $x_1 = 10$. To build the confidence intervals, we generate 250 data sets and with each data set, we produce a subbagged ensemble, estimate the parameters in the limiting distribution, and take the 0.025 and 0.975 quantiles from the estimated limiting normal distribution to form an approximate 95% confidence interval. The mean of this limiting normal θ_k was estimated as the mean of the predictions generated by the ensemble. The variance parameter $\zeta_{k,k}$ was estimated by drawing 500 new subsamples, not necessarily used in the ensemble, and calculating the variance between predictions generated by the resulting 500 trees and $\zeta_{1,k}$ was estimated externally using $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 250$.

In order to assess the coverage probability of our confidence intervals, we first need to estimate the true mean prediction θ_k at $x_1 = 10$ that would be generated by this subbagging ensemble. To estimate this true mean, we built 1000 subbagged ensembles and took the mean prediction generated by these ensembles, which we found to be 20.02 - very close to the true underlying value of 20. In this case, we found a coverage probability of 0.912, which means that 228 of our 250 confidence intervals contained our estimate of the true mean prediction. These confidence intervals are shown in Figure 3. The horizontal line in the plot is at 20.02 and represents our estimate of the true expected prediction.

This same procedure was repeated for the SLR case with $n = m = 1000$ and $k = 60$, the MARS case with $n = m = 500$ and $k = 50$, and the MARS case with $n = m = 1000$ and $k = 75$. In each case, we produced 250 confidence intervals and the coverage probabilities

are shown in Table 1. The parameters in the limiting distributions were estimated externally in exactly the same fashion, using $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 250$ to estimate $\zeta_{1,k}$. These slightly higher coverage probabilities mean that we are overestimating the variance, which is likely due to smaller values of the estimation parameters $n_{\tilde{\boldsymbol{z}}}$ and n_{MC} being used to estimate $\zeta_{1,k}$.

Underlying Function	n	k	θ_k	**Coverage Probability**	
				External Variance Est.	Internal Variance Est.
SLR	200	30	19.94	0.912	0.912
SLR	1000	60	19.99	0.956	0.936
MARS	500	50	17.43	0.980	0.984
MARS	1000	75	17.56	0.996	0.996

Table 1: Coverage probabilities

We also repeated this procedure for generating confidence intervals using the internal variance estimation method. The resulting coverage probabilities are remarkably similar to the external variance estimation method and are shown in Table 1. These ensembles were built using $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 250$.

5.3 Hypothesis Testing

We now explore the hypothesis testing procedure for assessing feature significance. We focus on the MARS case, where our training set now consists of 6 features $X_1, ..., X_6$, but the response Y depends only on the first 5. The values of the additional feature X_6 are sampled uniformly at random from the interval $[0, 1]$ and independently of the first 5 features.

We begin by looking at the distribution of test statistics when the test set consists 41 equally spaced points between 0 and 1. That is, the first test point is $x_1 = \cdots = x_6 = 0$, the second is $x_1 = \cdots = x_6 = 0.025$, and so on so that the last test point is $x_1 = \cdots = x_6 = 1$. For this test, we are interested in looking at the difference between trees built using all features and those built using only the first 5 so that in the notation in Section 4.2, $\boldsymbol{X}^{(R)} = \{X_1, ..., X_5\}$. We ran 250 simulations with $n = 1000$, $m = 1000$, and $k = 75$ using a test set consisting of all 41 test points, the 20 central-most points, and the 5 central-most points. The parameter $\Sigma_{1,k}$ was estimated externally using $n_{\tilde{\boldsymbol{z}}} = 100$ and $n_{MC} = 5000$ and $\Sigma_{k,k}$ was estimated by taking the covariance of the difference in predictions generated by 5000 trees. These covariance parameters are estimated only once instead of within each ensemble since we are only interested in the distribution of test statistics. Histograms of the resulting test statistics along with an overlay of the estimated χ^2 densities are shown in the top row of Figure 4.

We repeated this procedure, this time with a test set consisting of points in the center of the hypercube so we are not predicting close to any edge of the feature space. The value of each feature in the test set was selected uniformly at random from $[0.25, 0.75]$. A total of 41 such test points were generated, and histograms of the 250 resulting test statistics were produced in the cases where we use 5 of these test points, 20 test points, and all 41 test points. These histograms and estimated χ^2 densities are shown in the bottom row of

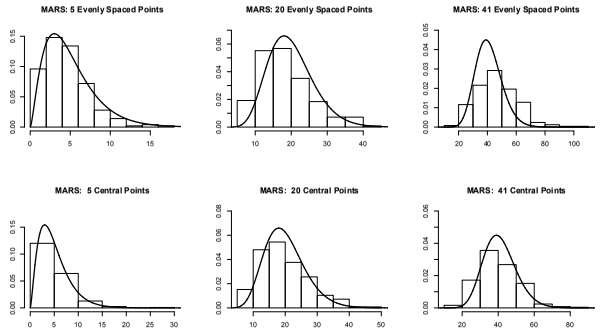


Figure 4: Histograms of simulated test statistics with estimated χ^2 overlay. The top row of histograms involve test points equally spaced between 0 and 1 and the bottom row corresponds to points randomly selected from the interior of the feature space.

Figure 4. Note that the bottom row appears to be a better fit and thus there appears to be some bias occuring when test points are selected near the edges of the feature space.

To check the alpha level of the test—the probability of incorrectly rejecting the null hypothesis—we simulated 250 new training sets and used the test set consisting of 41 randomly selected central points. For each training set, we built full and reduced subbagged estimates, allowed and not allowed to utilize X_6 respectively, estimated the parameters, and performed the hypothesis test. For each ensemble, the variance parameter $\Sigma_{1,k}$ was estimated externally using $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 1000$ and $\Sigma_{k,k}$ was estimated externally on an independently generated set of trees.

In this setup, none of the 250 simulations resulted in rejecting the null hypothesis, so our empirical alpha level was 0. A histogram of the resulting test statistics is shown on the left of Figure 5; the critical value, the 0.95 quantile of the χ^2_{41}, is 56.942. Thus, as with the confidence intervals, we are being conservative. Recall that our confidence interval simulations with $n = 1000$, $m = 1000$, and $k = 75$ predicting at $\boldsymbol{x} = (0.5, ..., 0.5)$ captured our true value 99.6% of the time, so this estimate of 0, though conservative, is not necessarily unexpected. We also repeated this procedure using an internal variance estimate with $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 1000$ and found an alpha level of 0.14. The histogram of test statistics resulting from the internal variance estimation method is shown on the right in Figure 5. Here we see that the correlation introduced by not taking an i.i.d. selection of subsamples to build the ensemble may be slightly inflating the test statistics.

5.4 Random Forests

Thus far, our simulations have dealt only with subbagged ensembles, but recall that Theorem 2 established the asymptotic normality for predictions generated by random forests as

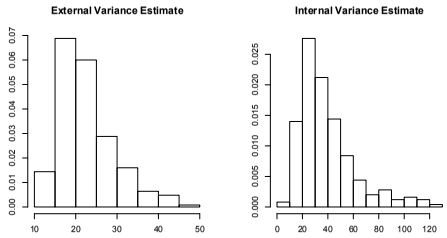


Figure 5: Histograms of simulated test statistics

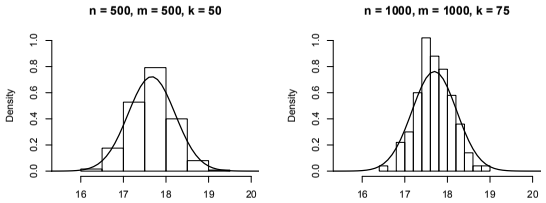


Figure 6: Histograms of random forest predictions at $x_1 = \cdots = x_5 = 0.5$ with estimated normal density overlaid.

well. The histograms in Figure 6 show the distribution of predictions generated by subsampled random forests at $x_1 = \cdots = x_5 = 0.5$ when the true underlying function is the MARS function. These trees were grown using the `randomForest` function in `R` with the `ntree` argument set to 1. At each node in each tree, 3 of the 5 features $X_1, ..., X_5$ were selected at random as candidates for splits and we insisted on at least 2 observations in each terminal node. The histograms show the empirical distribution of 250 subsampled random forest predictions and the overlaid density is the limiting normal $\mathcal{N}(\mathbb{E}r_{n,k,m}(\boldsymbol{x}^*), \frac{1}{\alpha}\frac{k^2}{m}\zeta_{1,k} + \frac{1}{m}\zeta_{k,k})$ with the variance parameters estimated externally. Our estimate of the mean of this distribution was taken as the empirical mean of the 250 predictions. Our estimate of $\zeta_{k,k}$ was taken as the empirical variance of predictions across 5000 new trees and the estimate for $\zeta_{1,k}$ was calculated with $n_{\tilde{\boldsymbol{z}}} = 250$ and $n_{MC} = 2000$.

6. Real Data

We now apply our inference methods to a real data set provided by the Lab of Ornithology at Cornell University. The data is part of the ongoing *eBird* citizen science project described in Sullivan et al. (2009). This project is hosted by Cornell's Lab of Ornithology and relies

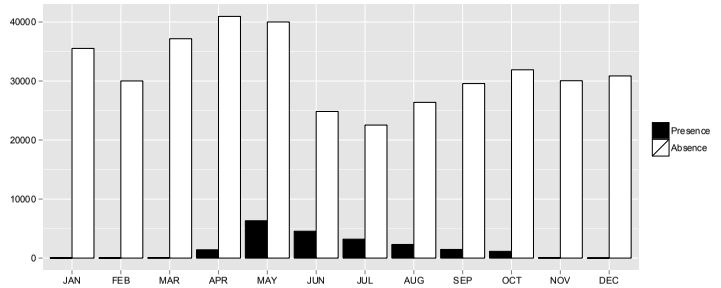


Figure 7: Monthly counts of Indigo Bunting observations.

on citizens, referred to as *birders*, to submit reports of bird observations. Location, bird species observed and not observed, effort level, and number of birds of each species observed are just a few of the variables participants are asked to provide. In addition to the data contained in these reports, landcover characteristics as reported in the 2006 United States National Land Cover Database are also available so that information about the local terrain may be used to help predict species abundance.

For our analysis, we restrict our attention to observations (and non-observations) of the Indigo Bunting species. For the first part of our analysis, we further restrict our attention to observations made during the year 2010. A little more than 400,000 reports of either presence or absence of Indigo Buntings were recorded during 2010 and the data set consists of 23 features. Like many species, the abundance of Indigo Buntings is known to fluctuate throughout the year, so we have two primary goals: (1) to produce confidence intervals for monthly abundance and (2) to show that the feature 'month' is significant for predicting abundance.

A presence/absence plot of Indigo Buntings by month is shown in Figure 7. A few features of this plot are worth pointing out. Most obviously, there are many more absence observations each month than presence observations. This makes sense because each time a birder submits a report, they note when Indigo Buntings are not present. Next, we see that this species is only observed during the warmer months, so month seems highly significant for predicting abundance. Finally, we see that all months have a large number of reports, so we need not worry about underreporting issues throughout the year.

First we produce the confidence intervals. The goal is to get an idea of the monthly abundance, which we can think of as 'probability of observation', so our test points will be 12 vectors, one for each month. For the values of the other features, we will use the average of the values recorded for that feature, or, in the case of categorical features, use the most popular category. Some features, such as elevation, have missing values which are not included in calculating the averages. We also removed the *day* feature from the training set, since day of the year is able to capture any effect of month. Since the training set consists of approximately 400,000 observations, we use a subsample size $k = 650$, slightly more than

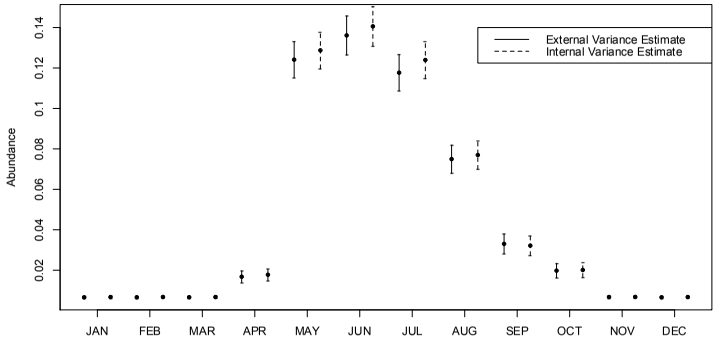


Figure 8: Monthly confidence intervals for Indigo Bunting abundance.

the square root of the training set size. We build a total of $m = 5000$ trees and take the variance of these predictions at each point to be our estimates of $\zeta_{k,k}$. We use an external estimate with $n_{\tilde{\boldsymbol{z}}} = 250$ and $n_{MC} = 5000$ to estimate $\zeta_{1,k}$. For each tree built, we require a minimum of 20 observations to consider splitting an internal node. We also repeat this procedure using an internal estimate of variance with $n_{\tilde{\boldsymbol{z}}} = 250$ and $n_{MC} = 5000$.

The confidence intervals are shown in Figure 8. Note that the pattern seen in Figure 7 is mirrored in the confidence interval plot: the confidence intervals are higher during months where more positive observations are recorded. It is also interesting to note that the width of the confidence intervals is larger during months of higher abundance. Observe in Figure 7 that even for months with many positive observations reported, many more negative observations are also reported. Thus, for these months there are likely a number of trees in the ensemble with nearly all positive or negative observations so we expect a higher variance in predictions. For months when very few positive observations are made, nearly all observations in the terminal node will be abscence observations, thus resulting in a very small variance and much narrower confidence intervals. Based on a visual inspection of the confidence intervals, there appears to be a clear significant difference in abundance between certain months, but we need to account for correlations in our predictions so we also conduct hypothesis tests.

We conduct these formal tests for the significance of month, following the procedure in Section 4.2. To perform this test, we randomly selected 20 points from the training set as the test set and calculated the test statistic based on an internal variance estimate with $n_{\tilde{\boldsymbol{z}}} = 250$ and $n_{MC} = 5000$. We calculated a test statistic of 4233.10 and a critical value, the 0.95 quantile of the χ^2_{20}, of only 31.41 which yields a p-value of approximately 0, so it seems that month is highly significant for predicting abundance. However, when we generated random values for month and repeated the testing procedure, we calculated a

test statistic of 58.02 which, though significantly smaller than the test statistic calculated on the original training set, is still significant. To ensure that the randomized months we selected did not add any accidental structure, we compared predictions generated using this training set to those generated by another training set, also with randomized months. Here we find a test statistic of only 2.36 and thus there is no significant difference between these predictions, so the trees are simply taking advantage of additional randomness. Finally, we test for a difference in predictions generated by the original training set and those generated by the training set with random values of month. In this case, we calculated a test statistic of 2336.14 which is highly significant so we can conclude that month is significant for predicting abundance and the contribution to the prediction is significantly more than would be expected by simply adding a random feature to the model.

This significant effect of month comes as no surprise as Indigo Buntings are known to be a migratory species. However, many scientists also believe that migrations may change from year to year and thus year may also be significant for predicting abundance. For this test, we used the full training data set consisting of approximately 1 million observations from 2004 to 2010 and as a test set, randomly selected 20 observations from the training set. Given this larger training set, we increased our subsample size to $k = 1000$ and our Monte Carlo sample size to $n_{MC} = 8000$ and again performed the tests using the internel variance estimation method. In the first setup where we test predictions from the full training set against predictions generated from the training set without year, we find a test statistic of 94.43 which means that year is significant for making predictions as it is larger than the critical value of 31.41. However, as was the case in testing the significance of month, we find that a randomly generated year feature is also significant with a test statistic of 52.51. Following in the same manner as above, we compare predictions from two training sets, each with a randomized year feature, and we find no significant difference in these predictions with a test statistic of only 4.70. Finally, we test for a difference in predictions between the full training set and a data set with random year and find that there is a significant difference in these predictions with a test statistic of 109.72. Thus, as was the case with month, we can conclude that year is significant for predicting abundance and the contribution to the prediction is significantly more than would be expected by simply adding a random feature to the model.

7. Discussion

This work demonstrates that formal statistical inference procedures are possible within the context of supervised ensemble learning, even when individual base learners are difficult to analyze mathematically. Demonstrating that ensembles built with subsamples can be viewed as U-statistics allows us to calculate limiting distributions for predictions and consistent estimation procedures for the parameters allow us to compute confidence intervals for predictions and formally test the significance of features. Moreover, using the internal variance estimation procedure, we are able to do so at no additional computational cost. In his controversial paper, Breiman (2001a) contrasted traditional statistical data analysis models that emphasize interpretation with the modern algorithmic approach of machine learning where the primary goal is prediction and among the differences invoked was the statisti-

cian's concern for formalized inference. Our hope is that this work be seen as something of a bridge between Breiman's two cultures.

The distributions and procedures we discuss apply to a very general class of supervised learning algorithms. We focus on bagging and random forests with tree-based base learners due to their popularity, but any supervised ensemble method that satisfies the conditions in Theorems 1 and 2 will generate predictions that have these limiting distributions and the inference procedures can be carried out in the same way. By the same reasoning, our procedures also make no restrictions on the tree-building method.

There are also some small modifications that can be made to our procedure which would still allow for an asymptotically normal limiting distribution. First, we assumed that our subsamples were selected with replacement so that in theory, the same subsample could be selected multiple times. This choice was based primarily on the fact that ensuring the exact subsample was not taken twice would typically require extra computational work. However, even for relatively small data sets, the probability of selecting the same subsample more than once is very small, and not surprisingly, the limiting distributions remain identical when the subsamples are taken without replacement. Furthermore, we also did not allow repeat observations within subsamples, which made the resulting estimator a U-statistic. Had we selected the subsamples themselves with replacement, our estimator would be a V-statistic—a closely related class of estimators introduced by von Mises (1947)—and similar theory could be developed. It is also worth pointing out that in practice, we always selected the same number of subsamples m_n and subsample size k_n for each prediction point $\boldsymbol{x}^*$ but in theory, these could be chosen differently for each point of interest and the same can be said of the estimation parameters $n_{\tilde{\boldsymbol{z}}}$ and n_{MC}. However, choosing different values of these parameters for different prediction points involves significantly more bookkeeping and we advise against it in practice.

Our approach also raises some issues. Perhaps most obviously, the parameter in our inference procedures is the expected prediction generated by trees built in the prescribed fashion, as opposed to the true value of the underlying regression function, $F(\boldsymbol{x}^*)$. Though some tree-building methods have been shown to be consistent, the bias introduced may not be negligible so we have to be careful about interpreting our results. It may be possible to employ a residual bootstrap to try and correct for bias and we plan to explore this in future work. Another open question that we hope to address in future work is how to select the test points when testing feature significance. In the eBird application, we randomly selected points from the training set, but it would be beneficial to investigate optimal strategies for selecting both the number and location of test points. Finally, the distributional results we provide could potentially allow us to test more complex hypotheses about the structure of the underlying regression function, for example, the interaction between covariates.

Acknowledgments

We would like to thank the action editor and reviewers for their careful and thoughtful remarks. We would also like to thank the Lab of Ornithology at Cornell University for providing interesting data. This work was supported by the following grants: NSF DEB-125619, NSF CDI Type II 1125098, and NSF DMS-1053252.

Appendix A.

We present here the proofs of the theorems provided in Section 2. We begin with a lemma from Lee (1990).

Lemma 3 *(Lee 1990, Lemma A, page 201) Let $a_1, a_2, ...$ be a sequence of constants such that $\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^n a_i = 0$ and $\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^n a_i^2 = \sigma^2$ and let the random variables $M_1, ..., M_n$ have a multinomial distribution, multinomial$(m_n; \frac{1}{m_n}, ..., \frac{1}{m_n})$. Then as $m_n, n \to \infty$, the limiting distribution of*

$$m_n^{-1/2}\sum_{i=1}^n a_i(M_i - m_n/n)$$

is $\mathcal{N}(0, \sigma^2)$.

Additionally, it will be useful to have the limiting distribution of complete infinite order U-statistics, which we provide in the lemma below.

Lemma 4 *Let $Z_1, Z_2, ... \overset{iid}{\sim} F_Z$ and let U_{n,k_n} be a complete, infinite order U-statistic with kernel h_{k_n} satisfying Condition 1 and $\theta_{k_n} = \mathbb{E}h_{k_n}(Z_1, ..., Z_{k_n})$ such that $\mathbb{E}h_{k_n}^2(Z_1, ..., Z_{k_n}) \leq C < \infty$ for all n and some constant C and $\lim \frac{k_n}{\sqrt{n}} = 0$. Then*

$$\frac{\sqrt{n}(U_{n,k_n} - \theta_{k_n})}{\sqrt{k_n^2\zeta_{1,k_n}}} \xrightarrow{d} \mathcal{N}(0,1).$$

The proof of Lemma 3 is provided in Lee (1990) page 201. The proof of Lemma 4 follows in exactly the same fashion as the proof of result *(i)* in Theorem 1 below. We take advantage of these lemmas in the following proofs.

Theorem 1 *Let $Z_1, Z_2, ... \overset{iid}{\sim} F_Z$ and let U_{n,k_n,m_n} be an incomplete, infinite order U-statistic with kernel h_{k_n} that satisfies Condition 1. Let $\theta_{k_n} = \mathbb{E}h_{k_n}(Z_1, ..., Z_{k_n})$ such that $\mathbb{E}h_{k_n}^2(Z_1, ..., Z_{k_n}) \leq C < \infty$ for all n and some constant C, and let $\lim \frac{n}{m_n} = \alpha$. Then as long as $\lim \frac{k_n}{\sqrt{n}} = 0$ and $\lim \zeta_{1,k_n} \neq 0$,*

(i) if $\alpha = 0$, then $\frac{\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{k_n^2\zeta_{1,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$.

(ii) if $0 < \alpha < \infty$, then $\frac{\sqrt{m_n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{\frac{k_n^2}{\alpha}\zeta_{1,k_n} + \zeta_{k_n,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$.

(iii) if $\alpha = \infty$, then $\frac{\sqrt{m_n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{\zeta_{k_n,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$.

Proof:

(i) Suppose first that $\alpha = 0$. In the interest of clarity, we follow the Hájek projection method discussed in Van der Vaart (2000) chapters 11 and 12. Define the Hájek projection of $U_{n,k_n,m_n} - \theta_{k_n}$ as

$$\begin{aligned}\hat{U}_{n,k_n,m_n} &= \sum_{i=1}^{n} \mathbb{E}(U_{n,k_n,m_n} - \theta_{k_n} \mid Z_i) - (n-1)\mathbb{E}(U_{n,k_n,m_n} - \theta_{k_n})\\ &= \sum_{i=1}^{n} \mathbb{E}(U_{n,k_n,m_n} - \theta_{k_n} \mid Z_i)\end{aligned}$$

so that for each term in the sum, we have

$$\begin{aligned}\mathbb{E}(U_{n,k_n,m_n} - \theta_{k_n} \mid Z_i) &= \mathbb{E}\left(\frac{1}{m_n}\sum_{\beta} h_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}}) - \theta_{k_n} \;\middle|\; Z_i\right)\\ &= \frac{1}{m_n}\sum_{\beta} \mathbb{E}\big(h_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}}) - \theta_{k_n} \mid Z_i\big) \qquad (7)\end{aligned}$$

where, in keeping with the notation in Van der Vaart (2000), we let β index the subsamples. Define $h_{1,k_n}(x) = \mathbb{E}h_{k_n}(x, Z_2, ..., Z_{k_n}) - \theta_{k_n}$ and let W be the number of subsamples that contain i. Since we assume that the subsamples are selected uniformly at random with replacement,

$$W \sim Binom\left(m_n, \frac{\binom{n-1}{k_n-1}}{\binom{n}{k}}\right)$$

so we can rewrite (7) as

$$\begin{aligned}&\frac{1}{m_n}\sum_{\beta}\mathbb{E}\Big(\mathbb{E}(h_{k_n}(Z_{\beta_1},...,Z_{\beta_{k_n}}) - \theta_{k_n}|Z_i) \mid W\Big)\\ &= \frac{1}{m_n}\mathbb{E}(W h_{1,k_n}(Z_i))\\ &= \frac{1}{m_n}\left(m_n\left(\frac{\binom{n-1}{k_n-1}}{\binom{n}{k}}\right)h_{1,k_n}(Z_i)\right)\\ &= \frac{k_n}{n}h_{1,k_n}(Z_i)\end{aligned}$$

so that taking the sum yields

$$\hat{U}_{n,k_n,m_n} = \frac{k_n}{n}\sum_{i=1}^{n} h_{1,k_n}(Z_i).$$

Now we establish the asymptotic normality of $\hat{U}_{n,k_n,m_n}$. Define the triangular array

$$\begin{array}{l}k_{n_1}h_{1,k_{n_1}}(Z_1), \;\dots\;, k_{n_1}h_{1,k_{n_1}}(Z_{n_1})\\ k_{n_1+1}h_{1,k_{n_1+1}}(Z_1), \;\dots\;\dots\;, k_{n_1+1}h_{1,k_{n_1+1}}(Z_{n_1+1})\\ \quad\vdots\\ k_{n_1+j}h_{1,k_{n_1+j}}(Z_1), \;\dots\;\dots\;\dots\;\dots\;\dots\;\dots\;, k_{n_1+j}h_{1,k_{n_1+j}}(Z_{n_1+j})\end{array}$$

so that for each variable in the array, we have

$$\mathbb{E}(k_n h_{1,k_n}(Z_i)) = k_n(\theta_{k_n} - \theta_{k_n}) = 0$$

and

$$var(k_n h_{1,k_n}(Z_i)) = k_n^2 var(h_{1,k_n}(Z_i)) = k_n^2\zeta_{1,k_n}$$

and thus the row-wise sum of the variances is

$$s_n^2 = \sum_{i=1}^{n} var(k_n h_{1,k_n}(Z_i)) = nk_n^2\zeta_{1,k_n}.$$

For $\delta > 0$, the Lindeberg condition is given by

$$\begin{aligned}&\lim_{n\to\infty}\sum_{i=1}^{n}\frac{1}{nk_n^2\zeta_{1,k_n}}\int_{|k_n h_{1,k_n}(Z_i)|\geq\delta k_n\sqrt{n\zeta_{1,k_n}}} k_n^2 h_{1,k_n}^2(Z_i)dP\\ &= \lim_{n\to\infty}\sum_{i=1}^{n}\frac{1}{n\zeta_{1,k_n}}\int_{|h_{1,k_n}(Z_i)|\geq\delta\sqrt{n\zeta_{1,k_n}}} h_{1,k_n}^2(Z_i)dP\\ &\leq \lim_{n\to\infty}\max_{1\leq i\leq n}\frac{1}{\zeta_{1,k_n}}\int_{|h_{1,k_n}(Z_i)|\geq\delta\sqrt{n\zeta_{1,k_n}}} h_{1,k_n}^2(Z_i)dP\\ &= \lim_{n\to\infty}\frac{1}{\zeta_{1,k_n}}\int_{|h_{1,k_n}(Z_1)|\geq\delta\sqrt{n\zeta_{1,k_n}}} h_{1,k_n}^2(Z_1)dP \qquad (8)\\ &= 0\end{aligned}$$

by Condition 1, and thus the Lindeberg condition is satisfied. Thus, by the Lindeberg-Feller central limit theorem,

$$\frac{\sum_j k_n h_{1,k_n}(Z_j)}{s_n} \xrightarrow{d} \mathcal{N}(0,1)$$

or, rewriting,

$$\frac{\sqrt{n}\hat{U}_{n,k_n,m_n}}{\sqrt{k_n^2\zeta_{1,k_n}}} \xrightarrow{d} \mathcal{N}(0,1).$$

Now, we need to compare the limiting variance ratio of U_{n,k_n,m_n} and its Hájek projection $\hat{U}_{n,k_n,m_n}$. For incomplete U-statistics of fixed rank, Blom (1976) showed that the variance of the incomplete U-statistic U_m consisting of m subsamples selected uniformly at random with replacement is given by

$$\frac{\zeta_k}{m_n} + \left(1 - \frac{1}{m_n}\right) var(U)$$

where U is the complete U-statistic analogue. Extending this result to our situation where k and m may both depend on n, we have

$$var(U_{n,k_n,m_n}) = \frac{\zeta_{k_n,k_n}}{m_n} + \left(1 - \frac{1}{m_n}\right) var(U_{n,k_n})$$

where the variance of the complete U-statistic U_{n,k_n} is given by

$$\sum_{c=1}^{k_n} \frac{k_n!^2}{c!(k_n-c)!^2} \frac{(n-k_n)(n-k_n-1)\cdots(n-2k_n+c+1)}{n(n-1)\cdots(n-k_n+1)} \zeta_{c,k_n}.$$

The details of this calculation are described in Van der Vaart (2000) page 163. Thus, looking at the limit of the variance ratio, we have

$$\begin{aligned}
\lim_{n\to\infty}\left(\frac{var(U_{n,k_n,m_n})}{var(\hat{U}_{n,k_n,m_n})}\right) &= \lim_{n\to\infty}\left(\frac{\frac{\zeta_{k_n,k_n}}{m_n} + \left(1-\frac{1}{m_n}\right) var(U_{n,k_n})}{\frac{k_n^2}{n}\zeta_{1,k_n}}\right) \\
&= \lim_{n\to\infty}\left(\frac{n\ \zeta_{k_n,k_n}}{m_n\ k_n^2\ \zeta_{1,k_n}}\right) + \lim_{n\to\infty}\left(1-\frac{1}{m_n}\right)\lim_{n\to\infty}\left(\frac{n\ var(U_{n,k_n})}{k_n^2\ \zeta_{1,k_n}}\right) \\
&= 0 + \lim_{n\to\infty}\left(\frac{n\ var(U_{n,k_n})}{k_n^2\ \zeta_{1,k_n}}\right) \\
&= \lim_{n\to\infty}\left(\frac{\sum_{c=1}^{k_n} \frac{k_n!^2}{c!(k_n-c)!^2} \frac{(n-k_n)(n-k_n-1)\cdots(n-2k_n+c+1)}{(n-1)\cdots(n-k_n+1)} \zeta_{c,k_n}}{k_n^2\zeta_{1,k_n}}\right) \\
&= \lim_{n\to\infty}\left(\frac{\frac{k_n!^2}{(k_n-1)!^2} \frac{(n-k_n)(n-k_n-1)\cdots(n-2k_n+2)}{(n-1)\cdots(n-k_n+1)} \zeta_{1,k_n}}{k_n^2\zeta_{1,k_n}}\right) \\
&= \lim_{n\to\infty}\left(\frac{(n-k_n)(n-k_n-1)\cdots(n-2k_n+2)}{(n-1)\cdots(n-k_n+1)}\right) \\
&= 1.
\end{aligned}$$

Note that in the second line, $\lim\left(\frac{n\ \zeta_{k_n,k_n}}{m_n\ k_n^2\ \zeta_{1,k_n}}\right) = 0$ since $\frac{n}{m_n} \to 0$, $\zeta_{1,k_n} \nrightarrow 0$ by assumption, and $\zeta_{k_n,k_n} \nrightarrow \infty$ since $\mathbb{E}h_{k_n}^2(Z_1,...,Z_{k_n})$ is bounded. Finally, by Slutsky's Theorem and Theorem 11.2 in Van der Vaart (2000), we have

$$\frac{\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{k_n^2\zeta_{1,k_n}}} = \frac{\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n} - \hat{U}_{n,k_n,m_n} + \hat{U}_{n,k_n,m_n})}{\sqrt{k_n^2\zeta_{1,k_n}}}$$

$$= \frac{\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n} - \hat{U}_{n,k_n,m_n})}{\sqrt{k_n^2\zeta_{1,k_n}}} + \frac{\sqrt{n}\hat{U}_{n,k_n,m_n}}{\sqrt{k_n^2\zeta_{1,k_n}}}$$

where $\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n} - \hat{U}_{n,k_n,m_n})/\sqrt{k_n^2\zeta_{1,k_n}} \xrightarrow{P} 0$ and

$$\frac{\sqrt{n}\hat{U}_{n,k_n,m_n}}{\sqrt{k_n^2\zeta_{1,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$$

so that

$$\frac{\sqrt{n}(U_{n,k_n,m_n} - \theta_{k_n})}{\sqrt{k_n^2\zeta_{1,k_n}}} \xrightarrow{d} \mathcal{N}(0,1)$$

as desired. □

(ii) & (iii) Now suppose $\alpha > 0$. We follow the proof technique in Lee (1990) page 200 which is based on the work of Janson (1984). Let $\mathcal{S}_{(n,k_n)} = \{S_i : i = 1,...,\binom{n}{k_n}\}$ denote the set of all possible subsamples of size k_n. In the following work, we use the notation (n,k_n) in place of $\binom{n}{k_n}$ in subscripts and summation notation. Consider the random vector $M_{n,k_n} = (M_{S_1},...,M_{S_{(n,k_n)}})$, where the i^{th} element denotes the number of times the i^{th} subsample appears in U_{n,k_n,m_n}. Since the subsamples are selected uniformly at random with replacement, $M_{n,k_n} \sim multinomial(m_n; \frac{1}{\binom{n}{k_n}},...,\frac{1}{\binom{n}{k_n}})$. Let ϕ_{n,k_n,m_n} be the characteristic function of $\sqrt{m_n}(U_{n,k_n,m_n} - \theta_{k_n})$ and let ϕ denote the limiting characteristic function of $\sqrt{n}(U_{n,k_n} - \theta_{k_n})$ where U_{n,k_n} is the corresponding complete U-statistic. Additionally, let $\phi^{(M)}_{n,k_n,m_n}$ be the characteristic function of the random variable

$$m_n^{-1/2} \sum_{i=1}^{(n,k)} \left(M_{S_i} - \frac{m_n}{\binom{n}{k_n}}\right)\Big(h_{k_n}(S_i) - \theta_{k_n}\Big)\Big| Z_1,...,Z_n.$$

Then we have

$$\begin{aligned}
\phi_{n,k_n,m_n}(t) &= \mathbb{E}\left(exp\left[it\sqrt{m_n}(U_{n,k_n,m_n} - \theta_{k_n})\right]\right) \\
&= \mathbb{E}\left(exp\left[itm_n^{-1/2}\left(\sum_{i=1}^{(n,k_n)} M_{S_i}(h_{k_n}(S_i) - \theta_{k_n})\right)\right]\right) \\
&= \mathbb{E}\left(\mathbb{E}\left(exp\left[itm_n^{-1/2}\left(\sum_{i=1}^{(n,k_n)} M_{S_i}(h_{k_n}(S_i) - \theta_{k_n})\right)\right]\Bigg| Z_1,...,Z_n\right)\right) \\
&= \mathbb{E}\Bigg(\mathbb{E}\Bigg(exp\Bigg[itm_n^{-1/2}\Bigg(\sum_{i=1}^{(n,k_n)}\left(M_{S_i} + \frac{m_n}{\binom{n}{k_n}} - \frac{m_n}{\binom{n}{k_n}}\right) \\
&\qquad\qquad \times \left(h_{k_n}(S_i) - \theta_{k_n}\right)\Bigg)\Bigg]\Bigg| Z_1,...,Z_n\Bigg)\Bigg)
\end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}\left(\mathbb{E}\left(exp\left[itm_n^{-1/2}\left(\sum_{i=1}^{(n,k_n)}\frac{m_n}{\binom{n}{k_n}}(h_{k_n}(S_i)-\theta_{k_n})\right)\right]\right.\right.\\
&\quad\left.\left.\times exp\left[itm_n^{-1/2}\left(\sum_{i=1}^{(n,k)}\left(M_{S_i}-\frac{m_n}{\binom{n}{k_n}}\right)(h_{k_n}(S_i)-\theta_{k_n})\right)\right]\Big|Z_1,...,Z_n\right)\right)\\
&= \mathbb{E}\left(exp\left[itm_n^{-1/2}\left(\sum_{i=1}^{(n,k_n)}\frac{m_n}{\binom{n}{k_n}}(h_{k_n}(S_i)-\theta_{k_n})\right)\right]\right.\\
&\quad\left.\times\,\mathbb{E}\left(exp\left[itm_n^{-1/2}\left(\sum_{i=1}^{(n,k_n)}\left(M_{S_i}-\frac{m_n}{\binom{n}{k_n}}\right)(h_{k_n}(S_i)-\theta_{k_n})\right)\right]\Big|Z_1,...,Z_n\right)\right)\\
&= \mathbb{E}\left(exp\left[it\sqrt{m_n}U_{n,k_n}\right]\;\phi^{(M)}_{n,k_n,m_n}(t)\right)
\end{aligned}
$$

and now, taking limits, we have

$$
\begin{aligned}
\lim_{n\to\infty}\phi_{n,k_n,m_n}(t) &= \lim_{n\to\infty}\mathbb{E}\left(exp\left[it\sqrt{m_n}U_{n,k_n}\right]\;\phi^{(M)}_{n,k_n,m_n}(t)\right)\\
&= \mathbb{E}\left(\lim_{n\to\infty}exp\left[it\sqrt{m_n}U_{n,k_n}\right]\;\lim_{n\to\infty}\phi^{(M)}_{n,k_n,m_n}(t)\right)
\end{aligned}
$$

so that by the preceeding lemmas,

$$
\begin{aligned}
\lim_{n\to\infty}\phi_{n,k_n,m_n}(t) &= \lim_{n\to\infty}\mathbb{E}\left(exp\left[it\sqrt{m_n}U_{n,k_n}\right]\right)exp\left[-t^2\zeta_{k_n,k_n}/2\right]\\
&= \lim_{n\to\infty}\mathbb{E}\left(exp\left[it\Big(\frac{\sqrt{m_n}}{\sqrt{n}}\Big)\sqrt{n}U_{n,k_n}\right]\right)exp\left[-t^2\zeta_{k_n,k_n}/2\right]\\
&= \phi(\alpha^{-1/2}t)exp\left[-t^2\zeta_{k_n,k_n}/2\right]\\
&= exp\left[-(t\alpha^{-1/2})^2k_n^2\zeta_{1,k_n}/2\right]exp\left[-t^2\zeta_{k_n,k_n}/2\right]\\
&= exp\left[-t^2\Big(\frac{k_n^2}{\alpha}\zeta_{1,k_n}+\zeta_{k_n,k_n}\Big)/2\right]
\end{aligned}
$$

which is the characteristic function of a Normal distribution with mean 0 and variance $\frac{k_n^2}{\alpha}\zeta_{1,k_n}+\zeta_{k_n,k_n}$. Note that when $\alpha=\infty$, the first term in the variance is 0, so the limiting variance reduces to ζ_{k_n,k_n}, as desired. ■

Proposition 1: *For a bounded regression function F, if there exists a constant c such that for all $k_n \geq 1$,*

$$
\begin{aligned}
\big|h((\boldsymbol{X}_1,Y_1),...,(\boldsymbol{X}_{k_n},Y_{k_n}),(\boldsymbol{X}_{k_n+1},Y_{k_n+1})) - h((\boldsymbol{X}_1,Y_1),...,(\boldsymbol{X}_{k_n},Y_{k_n}),(\boldsymbol{X}_{k_n+1},Y^*_{k_n+1}))\big|\\
\leq c\big|Y_{k_n+1}-Y^*_{k_n+1}\big|
\end{aligned}
$$

where $Y_{k_n+1}=F(\boldsymbol{X}_{k_n+1})+\epsilon_{k_n+1}$, $Y^*_{k_n+1}=F(\boldsymbol{X}_{k_n+1})+\epsilon^*_{k_n+1}$, and where ϵ_{k_n+1} and $\epsilon^*_{k_n+1}$ are i.i.d. with exponential tails, then Condition 1 is satisfied.

Proof. First consider the particular $Y^*_j=F(\boldsymbol{X}_j)$. Since F is bounded, we can define

$$\sup_{\boldsymbol{X}_j\in\mathcal{X}}\big|F(\boldsymbol{X}_j)\big|\leq M<\infty$$

and since tree-based predictions cannot fall outside the range of responses in the training set, $\big|h\big((\boldsymbol{X}_1,Y^*_1),...,(\boldsymbol{X}_{k_n},Y^*_{k_n})\big)\big|\leq M$ for all possible $\boldsymbol{X}_1,...,\boldsymbol{X}_{k_n}$. Furthermore, by the Lipschitz condition,

$$\big|h\big((\boldsymbol{X}_1,Y^*_1),...,(\boldsymbol{X}_{k_n},Y^*_{k_n})\big)-h\big((\boldsymbol{X}_1,Y^*_1),(\boldsymbol{X}_2,Y_2),...,(\boldsymbol{X}_{k_n},Y_{k_n})\big)\big|\leq c\sum_{j=2}^{k_n}\big|\epsilon_j\big|$$

and thus, applying Jensen's Inequality, we have

$$
\begin{aligned}
&\sup_{\boldsymbol{X}_1\in\mathcal{X}}\big|h_{1,k_n}\big((\boldsymbol{X}_1,Y^*_1)\big)\big|\\
&=\sup_{\boldsymbol{X}_1\in\mathcal{X}}\left|\int h\big((\boldsymbol{X}_1,Y^*_1),(\boldsymbol{X}_2,Y_2),...,(\boldsymbol{X}_{k_n},Y_{k_n})\big)-h\big((\boldsymbol{X}_1,Y^*_1),...,(\boldsymbol{X}_{k_n},Y^*_{k_n})\big)\right.\\
&\qquad\left.+\,h\big((\boldsymbol{X}_1,Y^*_1),...,(\boldsymbol{X}_{k_n},Y^*_{k_n})\big)dP-\theta_{k_n}\right|\\
&\leq\sup_{\boldsymbol{X}_1\in\mathcal{X},...,\boldsymbol{X}_{k_n}\in\mathcal{X}}\int\Big|h\big((\boldsymbol{X}_1,Y^*_1),(\boldsymbol{X}_2,Y_2),...,(\boldsymbol{X}_{k_n},Y_{k_n})\big)-h\big((\boldsymbol{X}_1,Y^*_1),...,(\boldsymbol{X}_{k_n},Y^*_{k_n})\big)\Big|dP\\
&\qquad+\sup_{\boldsymbol{X}_1\in\mathcal{X}}h\big((\boldsymbol{X}_1,Y^*_1),...,(\boldsymbol{X}_{k_n},Y^*_{k_n})\big)-\theta_{k_n}\\
&\leq ck_n\mathbb{E}\big|\epsilon_1\big|+M-\theta_{k_n}.
\end{aligned}
$$

Now, define the set of interest in the Lindeberg condition as A_n so that we may write

$$
\begin{aligned}
A_n &= \left\{ \left| h_{1,k_n}\big((\boldsymbol{X}_1, Y_1)\big) \right| \geq \delta \sqrt{n\zeta_{1,k_n}} \right\} \\
&= \left\{ \left| h_{1,k_n}\big((\boldsymbol{X}_1, Y_1)\big) - h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*)\big) + h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*)\big) \right| \geq \delta \sqrt{n\zeta_{1,k_n}} \right\} \\
&\subseteq \left\{ \left| h_{1,k_n}\big((\boldsymbol{X}_1, Y_1)\big) - h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*)\big) \right| \geq \delta \sqrt{n\zeta_{1,k_n}} + \left| h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*)\big) \right| \right\} \\
&\subseteq \left\{ \left|\epsilon_1\right| \geq \frac{1}{c} \left(\delta \sqrt{n\zeta_{1,k_n}} + M - \theta_{k_n} \right) + k_n \mathbb{E}\left|\epsilon_1\right| \right\} \\
&=: A_n^*
\end{aligned}
$$

Finally, continuing from equation (8) of Theorem 1,

$$
\begin{aligned}
&\lim_{n\to\infty} \frac{1}{\zeta_{1,k_n}} \int_{A_n} h_{1,k_n}^2\big((\boldsymbol{X}_1, Y_1)\big) dP \\
&= \lim_{n\to\infty} \frac{1}{\zeta_{1,k_n}} \int_{A_n} \left(h_{1,k_n}\big((\boldsymbol{X}_1, Y_1)\big) - h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*) + h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*)\big) \right)^2 dP \\
&\leq \lim_{n\to\infty} \frac{2}{\zeta_{1,k_n}} \int_{A_n} \left(h_{1,k_n}\big((\boldsymbol{X}_1, Y_1)\big) - h_{1,k_n}\big((\boldsymbol{X}_1, Y_1^*)\big) \right)^2 dP \\
&\qquad\qquad + \lim_{n\to\infty} \frac{2}{\zeta_{1,k_n}} \int_{A_n} h_{1,k_n}^2\big((\boldsymbol{X}_1, Y_1^*) dP \\
&\leq \lim_{n\to\infty} \frac{2}{\zeta_{1,k_n}} \int_{A_n^*} c^2 \epsilon_1^2 dP + \lim_{n\to\infty} \frac{2}{\zeta_{1,k_n}} P(A_n^*) \left(c k_n \mathbb{E}\left|\epsilon_1\right| + M - \theta_{k_n} \right)^2 \\
&= \lim_{n\to\infty} \frac{2}{\zeta_{1,k_n}} P\left[\left|\epsilon_1\right| \geq \frac{1}{c} \left(\sqrt{n\zeta_{1,k_n}} + M - \theta_{k_n} \right) + k \mathbb{E}\left|\epsilon_1\right| \right] \\
&\qquad\qquad \times \left(c k_n \mathbb{E}\left|\epsilon_1\right| + M - \theta_{k_n} \right)^2 \\
&= 0
\end{aligned}
$$

as desired, so long as $nP\left(\left|\epsilon_1\right| > \sqrt{n}\right) \to 0$, which is the case with exponential tails, and $k_n = o(\sqrt{n})$. ■

Theorem 2 *Let $U_{\omega;n,k_n,m_n}$ be a random kernel U-statistic of the form defined in equation (5) such that $U^*_{\omega;n,k_n,m_n}$ satisfies Condition 1 and suppose that $\mathbb{E}h^2_{k_n}(Z_1, ..., Z_{k_n}) < \infty$ for all n, $\lim \frac{k_n}{\sqrt{n}} = 0$, and $\lim \frac{n}{m_n} = \alpha$. Then, letting β index the subsamples, so long as $\lim \zeta_{1,k_n} \neq 0$ and*

$$
\lim_{n\to\infty} \mathbb{E}\left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \neq \infty,
$$

$U_{\omega;n,k_n,m_n}$ is asymptotically normal and the limiting distributions are the same as those provided in Theorem 1.

Proof. We begin with the case where $\alpha = 0$ and we make use of this result in the proof of the case where $\alpha > 0$. As in Section 2.2, define $U^*_{\omega;n,k_n,m_n} = \mathbb{E}_\omega U_{\omega;n,k_n,m_n}$. We have

$$
\begin{aligned}
&\mathbb{E}(U_{\omega;n,k_n,m_n} - U^*_{\omega;n,k_n,m_n})^2 \\
&= \mathbb{E}\left[\left(\frac{1}{m_n} \sum_\beta h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega \left(\frac{1}{m_n} \sum_\beta h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right) \right)^2 \right] \\
&= \mathbb{E}\left[\frac{1}{m_n^2} \left(\sum_\beta h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega \left(\sum_\beta h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right) \right)^2 \right] \\
&= \mathbb{E}\frac{1}{m_n^2} \left(\sum_\beta h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \left(\sum_\beta \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right) \right)^2 \\
&= \mathbb{E}\left[\frac{1}{m_n^2} \sum_\beta \left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \right] \\
&\quad + \mathbb{E}\left[\frac{1}{m_n^2} \sum_{\beta_i \neq \beta_j} \left(h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) \right) \right. \\
&\qquad\qquad \left. \times \left(h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) \right) \right]
\end{aligned}
$$

We focus now on the second term, involving the cross terms with different subsamples and randomization parameters. Splitting apart the expectation and moving the expectation with respect to ω inside, we can write the second term as

$$
\begin{aligned}
&\mathbb{E}\left[\frac{1}{m_n^2} \sum_{\beta_i \neq \beta_j} \left(h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) \right) \right. \\
&\qquad\qquad \left. \times \left(h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) \right) \right] \\
&= \mathbb{E}_{\boldsymbol{X}}\left[\frac{1}{m_n^2} \sum_{\beta_i \neq \beta_j} \mathbb{E}_\omega \left(h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) \right) \right. \\
&\qquad \left. \times \mathbb{E}_\omega \left(h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) \right) \right] \\
&= \mathbb{E}_{\boldsymbol{X}}\left[\frac{1}{m_n^2} \sum_{\beta_i \neq \beta_j} \left(\mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{i_1}}, ..., Z_{\beta_{i_{k_n}}}) \right) \right.
\end{aligned}
$$

$$\times \left(\mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_{j_1}}, ..., Z_{\beta_{j_{k_n}}}) \right) \Bigg]$$
$$= \mathbb{E}_{\boldsymbol{X}} \left[\frac{1}{m_n^2} \sum_{\beta_i \neq \beta_j} 0 \times 0 \right]$$
$$= 0$$

and thus we need only investigate the first term. We have

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{m_n^2} \sum_{\beta} \left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \right] \\
&= \frac{1}{m_n^2} \sum_{\beta} \mathbb{E}\left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \\
&= \frac{1}{m_n^2} m_n \mathbb{E}\left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \\
&= \frac{1}{m_n} \mathbb{E}\left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2.
\end{aligned}$$

Putting all of this together, we have

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{E}\left(\frac{\sqrt{n}\left(U_{\omega;n,k_n,m_n} - U^*_{\omega;n,k_n,m_n} \right)}{\sqrt{k_n^2 \zeta_{1,k_n}}} \right)^2 \\
&= \lim_{n\to\infty} \frac{n}{m_n} \frac{1}{k_n^2 \zeta_{1,k_n}} \mathbb{E}\left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \\
&= 0
\end{aligned}$$

so long as $k_n^2 \zeta_{1,k_n} \nrightarrow 0$ and $\lim_{n\to\infty} \mathbb{E}\left(h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) - \mathbb{E}_\omega h_{k_n}^{(\omega)}(Z_{\beta_1}, ..., Z_{\beta_{k_n}}) \right)^2 \neq \infty$, as desired. □

Now we handle the case where $\alpha > 0$. First note that when $k_n = 1$ for all n, this reduces to simply averaging over an *i.i.d.* sample and thus asymptotic normality can be obtained via the classic central limit theorem so assume that eventually $k_n > 1$. The remaining steps in this proof are nearly identical to the proof of results *(ii)* and *(iii)* of Theorem 1. Again, let $\mathcal{S}_{(n,k_n)} = \{S_i : i = 1, ..., \binom{n}{k_n}\}$ denote the set of all possible subsamples of size k_n and let $M_{n,k_n} = (M_{S_1}, ..., M_{S_{(n,k_n)}})$ denote the random vector that counts the number of times each subsample appears so that $M_{n,k_n} \sim multinomial(m_n; \frac{1}{\binom{n}{k_n}}, ..., \frac{1}{\binom{n}{k_n}})$ since we assume the subsamples are selected uniformly at random with replacement. Define

$\theta^*_{k_n} = \mathbb{E}U^*_{\omega;n,k_n,m_n}$, let ϕ_{n,k_n,m_n} be the characteristic function of $\sqrt{m_n}(U_{\omega;n,k_n,m_n} - \theta^*_{k_n})$, and let ϕ denote the limiting characteristic function of $\sqrt{n}(U^*_{\omega;n,k_n,m_n} - \theta^*_{k_n})$. Finally, let $\phi^{(M)}_{n,k_n,m_n}$ be the characteristic function of the random variable

$$m_n^{-1/2} \sum_{i=1}^{(n,k)} \left(M_{S_i} - \frac{m_n}{\binom{n}{k_n}} \right) \left(h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n} \right) \Big| Z_1, ..., Z_n, \omega.$$

Then we have

$$\begin{aligned}
\phi_{n,k_n,m_n}(t) &= \mathbb{E}\left(exp\left[it\sqrt{m_n}(U_{\omega;n,k_n,m_n} - \theta^*_{k_n}) \right] \right) \\
&= \mathbb{E}\left(exp\left[itm_n^{-1/2} \left(\sum_{i=1}^{(n,k_n)} M_{S_i}(h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n}) \right) \right] \right) \\
&= \mathbb{E}\left(\mathbb{E}\left(exp\left[itm_n^{-1/2} \left(\sum_{i=1}^{(n,k_n)} M_{S_i}(h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n}) \right) \right] \Big| Z_1, ..., Z_n, \omega \right) \right) \\
&= \mathbb{E}\Bigg(\mathbb{E}\Bigg(exp\Bigg[itm_n^{-1/2} \Bigg(\sum_{i=1}^{(n,k_n)} \left(M_{S_i} + \frac{m_n}{\binom{n}{k_n}} - \frac{m_n}{\binom{n}{k_n}} \right) \\
&\qquad \times \left(h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n} \right) \Bigg) \Bigg] \Bigg| Z_1, ..., Z_n, \omega \Bigg) \Bigg) \\
&= \mathbb{E}\Bigg(\mathbb{E}\Bigg(exp\Bigg[itm_n^{-1/2} \left(\sum_{i=1}^{(n,k_n)} \frac{m_n}{\binom{n}{k_n}} (h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n}) \right) \Bigg] \\
&\qquad \times exp\left[itm_n^{-1/2} \left(\sum_{i=1}^{(n,k)} \left(M_{S_i} - \frac{m_n}{\binom{n}{k_n}} \right) (h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n}) \right) \right] \Big| Z_1, ..., Z_n, \omega \Bigg) \Bigg) \\
&= \mathbb{E}\Bigg(exp\Bigg[itm_n^{-1/2} \left(\sum_{i=1}^{(n,k_n)} \frac{m_n}{\binom{n}{k_n}} (h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n}) \right) \Bigg] \\
&\quad \times \mathbb{E}\Bigg(exp\left[itm_n^{-1/2} \left(\sum_{i=1}^{(n,k_n)} \left(M_{S_i} - \frac{m_n}{\binom{n}{k_n}} \right) (h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n}) \right) \right] \Big| Z_1, ..., Z_n, \omega \Bigg) \Bigg) \\
&= \mathbb{E}\Bigg(exp\Bigg[itm_n^{-1/2} \Bigg(\sum_{i=1}^{(n,k_n)} \frac{m_n}{\binom{n}{k_n}} \Big(h_{k_n}^{(\omega_i)}(S_i) - \mathbb{E}_\omega h_{k_n}^{(\omega_i)}(S_i) \\
&\qquad + \mathbb{E}_\omega h_{k_n}^{(\omega_i)}(S_i) - \theta^*_{k_n} \Big) \Bigg) \Bigg] \ \phi^{(M)}_{n,k_n,m_n}(t) \Bigg) \\
&= \mathbb{E}\Bigg(exp\Bigg[it\sqrt{m_n} \Bigg(\frac{1}{\binom{n}{k_n}} \sum_{i=1}^{(n,k_n)} \left(h_{k_n}^{(\omega_i)}(S_i) - \mathbb{E}_\omega h_{k_n}^{(\omega_i)}(S_i) \right)
\end{aligned}$$

$$+ \frac{1}{\binom{n}{k_n}} \sum_{i=1}^{(n,k_n)} \left(\mathbb{E}_\omega h_{k_n}^{(\omega_i)}(S_i) - \theta_{k_n}^* \right) \Bigg) \Bigg] \phi_{n,k_n,m_n}^{(M)}(t) \Bigg).$$

Now, note that since we are in the case where $\alpha > 0$, $m_n = O(n) \ll (n, k_n)$ and thus, by the previous result in the case where $\alpha = 0$, the first term converges to 0 and we have

$$\begin{aligned} \lim_{n\to\infty} \phi_{n,k_n,m_n}(t) &= \mathbb{E}\left(\lim_{n\to\infty} exp\left[it\sqrt{m_n}\, \frac{1}{\binom{n}{k_n}} \sum_{i=1}^{(n,k_n)} \left(\mathbb{E}_\omega h_{k_n}^{(\omega_i)}(S_i) - \theta_{k_n}^* \right) \right] \lim_{n\to\infty} \phi_{n,k_n,m_n}^{(M)}(t) \right) \\ &= \mathbb{E}\left(\lim_{n\to\infty} exp\left[it\sqrt{m_n}\, U_{\omega;n,k_n,m_n}^* \right] \lim_{n\to\infty} \phi_{n,k_n,m_n}^{(M)}(t) \right) \end{aligned}$$

so that by exactly the same arguments as in the proof of Theorem 1

$$\lim_{n\to\infty} \phi_{n,k_n,m_n}(t) = exp\left[-t^2 \Big(\frac{k_n^2}{\alpha} \zeta_{1,k_n}^* + \zeta_{k_n,k_n}^* \Big) / 2 \right]$$

which is the characteristic function of a Normal distribution with mean 0 and variance $\frac{k_n^2}{\alpha}\zeta_{1,k_n}^* + \zeta_{k_n,k_n}^*$. Further, when $\alpha = \infty$, the variance reduces to ζ_{k_n,k_n}^*, as desired. ■

Appendix B.

Crossed Design Random Forests

Typically, each tree in a random forest is built according to an independently selected randomization parameter ω. Alternatively, for each ω, we could choose to build an entire set of trees, one for each of the m_n subsamples, so that if Ω_n randomization parameters are used, a total of $\Omega_n \times m_n$ trees are built. We could then write the prediction at $\boldsymbol{x}^*$ generated by this alternative random forest estimator as

$$\frac{1}{\Omega_n} \frac{1}{m_n} \sum_{(i)} \sum_{(j)} T_{\boldsymbol{x}^*,k_n}((\boldsymbol{X}_{i_1}, Y_{i_1}), ..., (\boldsymbol{X}_{i_{k_n}}, Y_{i_{k_n}});\ \omega_j) \tag{9}$$

where here, we explicitly treat ω as an input to the function. Statistics of the form in (9) are referred to as *two-sample* or *generalized* U-statistics and similar results regarding asymptotic normality have been established for fixed-rank kernels; see Lee (1990) or Van der Vaart (2000) for details.

We mention this as a possible alternative only because the resulting estimator takes the well established form of a generalized U-statistic. Readers familiar with U-statistics may be more comfortable with this approach than with the random kernel approach that more closely resembles the type of random forests used in practice. However, since this formulation strays from Breiman's original procedure and is far more computationally intensive, we consider only the random kernel version described in Section 2.2.

Appendix C.

The algorithm for estimating Σ_{1,k_n} as needed for the hypothesis testing procedure is given below.

Algorithm 6 Σ_{1,k_n} Estimation Procedure

```
for i in 1 to n_z̃ do
    Select initial fixed point z̃^(i)
    for j in 1 to n_MC do
        Select subsample S_z̃^(i),j of size k_n from training set that includes z̃^(i)
        Build full tree using subsample S_z̃^(i),j
        Build reduced tree using subsample S_z̃^(i),j utilizing only reduced feature space
        Use both full tree and reduced tree to predict at each test point
        Record difference in predictions
    end for
    Record average of the n_MC differences in predictions
end for
Compute the covariance of the n_z̃ averages
```

Appendix D.

Distribution of Subbagged Predictions using an Internal Variance Estimate

Here we examine the distribution of predictions when the ensemble is built according to the internal variance estimation method described in Algorithm 5. Since we are only interested in the distribution of predictions, we omit the steps in Algorithm 5 for estimating the variance parameters and use the same estimates as in the external case above. For both the SLR case with $n = 1000, k = 60$ and the MARS case with $n = 1000, k = 75$, we use $n_{\tilde{\boldsymbol{z}}} = 50$ and $n_{MC} = 250$ to produce a total of $m = 12500$ predictions in each ensemble. A total of 250 ensembles were built and the resulting histograms with estimated normal densities overlaid are shown in Figure 9 below. We see that these distributions appear to be the same as when the subsamples are selected uniformly at random, as in the external variance estimation method.

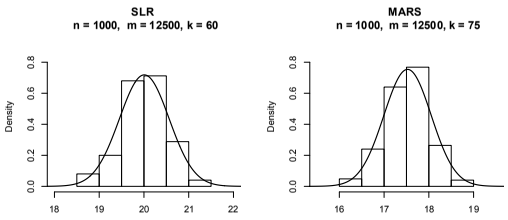


Figure 9: Histograms of subbagged predictions with using an internal estimate of variance. Predictions are made at $x_1 = 10$ in the SLR case and at $x_1 = \cdots = x_5 = 0.5$ in the MARS case.

References

Savina Andonova, Andre Elisseeff, Theodoros Evgeniou, and Massimiliano Pontil. A simple algorithm for learning stable machines. In *ECAI*, pages 513–517, 2002.

Gérard Biau. Analysis of a Random Forests Model. *The Journal of Machine Learning Research*, 98888:1063–1095, 2012.

Gérard Biau, Luc Devroye, and Gábor Lugosi. Consistency of Random Forests and Other Averaging Classifiers. *The Journal of Machine Learning Research*, 9:2015–2033, 2008.

Justin Bleich, Adam Kapelner, Edward I George, Shane T Jensen, et al. Variable selection for bart: An application to gene regulation. *The Annals of Applied Statistics*, 8(3): 1750–1781, 2014.

Gunnar Blom. Some Properties of Incomplete U-Statistics. *Biometrika*, 63(3):573–580, 1976.

Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199–231, 2001a.

Leo Breiman. Random Forests. *Machine Learning*, 45:5–32, 2001b.

Leo Breiman, Jerome Friedman, Charles J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1st edition, 1984.

Hugh A Chipman, Edward I George, and Robert E McCulloch. BART: Bayesian Additive Regression Trees. *The Annals of Applied Statistics*, 4(1):266–298, 2010.

Misha Denil, David Matheson, and Nando de Freitas. Narrowing the gap: Random forests in theory and in practice. arXiv:1310.1415v1 [stat.ML], October 2013.

Bradley Efron. Estimation and accuracy after model selection. *Journal of the American Statistical Association*, 109(507):991–1007, 2014.

Randall L Eubank. *Nonparametric regression and spline smoothing*. CRC press, 1999.

Edward W Frees. Infinite Order U-Statistics. *Scandinavian Journal of Statistics*, pages 29–45, 1989.

Jerome H Friedman. Multivariate Adaptive Regression Splines. *The Annals of Statistics*, pages 1–67, 1991.

Paul R Halmos. The theory of unbiased estimation. *The Annals of Mathematical Statistics*, 17(1):34–43, 1946.

Wassily Hoeffding. A Class of Statistics with Asymptotically Normal Distribution. *The Annals of Mathematical Statistics*, 19(3):293–325, 1948.

Svante Janson. The Asymptotic Distributions of Incomplete u-Statistics. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 66(4):495–505, 1984.

Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

Alan J Lee. *U-Statistics: Theory and Practice*. CRC Press, New York, 1990.

Yi Lin and Yongho Jeon. Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, 101(474):578–590, 2006.

Nicolai Meinshausen. Quantile regression forests. *The Journal of Machine Learning Research*, 7:983–999, 2006.

Joseph Sexton and Petter Laake. Standard errors for bagged and random forest estimators. *Computational Statistics & Data Analysis*, 53(3):801–811, 2009.

Brian L Sullivan, Christopher L Wood, Marshall J Iliff, Rick E Bonney, Daniel Fink, and Steve Kelling. ebird: A citizen-based bird observation network in the biological sciences. *Biological Conservation*, 142(10):2282–2292, 2009.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Aad W Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2): 264–280, 1971.

R von Mises. On the Asymptotic Distribution of Differentiable Statistical Functions. *The Annals of Mathematical Statistics*, 18(3):309–348, 1947.

Stefan Wager. Asymptotic Theory for Random Forests. arXiv:1405.0352 [math.ST], May 2014.

Stefan Wager, Trevor Hastie, and Bradley Efron. Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *Journal of Machine Learning Research*, 15: 1625–1651, 2014.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6): 80–83, 1945.

Faisal Zaman and Hideo Hirose. Effect of subsampling rate on subbagging and related ensembles of stable classifiers. In *Pattern Recognition and Machine Intelligence*, pages 44–49. Springer, 2009.

Ruoqing Zhu, Donglin Zeng, and Michael R Kosorok. Reinforcement learning trees. *Journal of the American Statistical Association*, (just-accepted), 2015.

Journal of Machine Learning Research 17 (2016) 1-57 Submitted 8/14; Revised 5/15; Published 4/16

Statistical-Computational Tradeoffs in Planted Problems and Submatrix Localization with a Growing Number of Clusters and Submatrices*

Yudong Chen YUDONG.CHEN@CORNELL.EDU
Department of Operations Research and Information Engineering
Cornell University
Ithaca, NY 14853, USA

Jiaming Xu JIAMINGX@WHARTON.UPENN.EDU
Department of Statistics, The Wharton School
University of Pennsylvania
Philadelphia, PA 19104, USA

Editor: Gabor Lugosi

Abstract

We consider two closely related problems: planted clustering and submatrix localization. In the planted clustering problem, a random graph is generated based on an underlying cluster structure of the nodes; the task is to recover these clusters given the graph. The submatrix localization problem concerns locating hidden submatrices with elevated means inside a large real-valued random matrix. Of particular interest is the setting where the number of clusters/submatrices is allowed to grow unbounded with the problem size. These formulations cover several classical models such as planted clique, planted densest subgraph, planted partition, planted coloring, and the stochastic block model, which are widely used for studying community detection, graph clustering and bi-clustering.

For both problems, we show that the space of the model parameters (cluster/submatrix size, edge probabilities and the mean of the submatrices) can be partitioned into four disjoint regions corresponding to decreasing statistical and computational complexities: (1) the *impossible* regime, where all algorithms fail; (2) the *hard* regime, where the computationally expensive Maximum Likelihood Estimator (MLE) succeeds; (3) the *easy* regime, where the polynomial-time convexified MLE succeeds; (4) the *simple* regime, where a local counting/thresholding procedure succeeds. Moreover, we show that each of these algorithms provably fails in the harder regimes.

Our results establish the minimax recovery limits, which are tight up to universal constants and hold even with a growing number of clusters/submatrices, and provide order-wise stronger performance guarantees for polynomial-time algorithms than previously known. Our study demonstrates the tradeoffs between statistical and computational considerations, and suggests that the minimax limits may not be achievable by polynomial-time algorithms.

Keywords: planted partition, planted clique, planted coloring, submatrix localization, graph clustering, bi-clustering, minimax recovery, computational hardness, convex relaxation

*. Partial preliminary results are presented in the conference paper Chen and Xu (2014).

1. Introduction

In this paper we consider two closely related problems: planted clustering and submatrix localization, both concerning the recovery of hidden structures from a noisy random graph or matrix.

- *Planted Clustering:* Suppose that out of a total of n nodes, rK of them are partitioned into r clusters of size K, and the remaining $n - rK$ nodes do not belong to any clusters; each pair of nodes is connected by an edge with probability p if they are in the same cluster, and with probability q otherwise. Given the adjacency matrix A of the graph, the goal is to recover the underlying clusters (up to a permutation of cluster indices). By varying the values of the model parameters, this formulation covers several classical models including planted clique, planted coloring, planted densest subgraph, planted partition, and stochastic block model (cf. Definition 1 and discussion thereafter).

- *Submatrix Localization:* Suppose $A \in \mathbb{R}^{n_L \times n_R}$ is a random matrix with independent Gaussian entries with unit variance, where there are r submatrices of size $K_L \times K_R$ with disjoint row and column supports, such that the entries inside these submatrices have mean $\mu > 0$, and the entries outside have mean zero. The goal is to identify the locations of these hidden submatrices given A. This formulation generalizes the submatrix detection and bi-clustering models with a single bi-submatrix/cluster that are studied in previous work (cf. Definition 14 and discussion thereafter).

We are particularly interested in the setting where the number r of clusters or submatrices may grow unbounded with the problem dimensions n, n_L, and n_R at an arbitrary rate. We call this the *high-rank* setting because r equals the rank of a matrix representation of the clusters and submatrices (cf. Definitions 1 and 14). The other parameters K, p, q, and μ are also allowed to scale with n or (n_L, n_R).

These two problems have been studied extensively, under various names such as *community detection*, *graph clustering/bi-clustering*, and *reconstruction in stochastic block models*, and have a broad range of applications. They are used as generative models for approximating real-world networks and data arrays with natural cluster/community structures, such as social networks (Fortunato, 2010), gene expressions (Shabalin et al., 2009), and online ratings (Xu et al., 2014). They serve as benchmarks in the evaluation of algorithms for clustering (Mathieu and Schudy, 2010), bi-clustering (Balakrishnan et al., 2011a), community detection (Newman and Girvan, 2004), and other network inference problems. They also provide a venue for studying the average-case behaviors of many graph theoretic problems including max-clique, max-cut, graph partitioning, and coloring (Bollobás and Scott, 2004; Condon and Karp, 2001). The importance of these two problems are well-recognized in areas across computer science, statistics, and physics (Rohe et al., 2011; Arias-Castro and Verzelen, 2014; Nadakuditi and Newman, 2012; Decelle et al., 2011; Mossel et al., 2013; Lelarge et al., 2013; Anandkumar et al., 2014; Bickel and Chen, 2009; Amini et al., 2013).

The planted clustering and submatrix localization problems exhibit an interplay between *statistical* and *computational* considerations. From a statistical point of view, we are interested in identifying the range of the model parameters for which the hidden structures—in this case the clusters and submatrices—can be recovered from the noisy data A. The values of the parameters n, r, K, p, q, μ govern the statistical hardness of the problems: the problems become more difficult with smaller values of $p - q$, μ, K, and larger r, because the observations are noisier and the sought-

after structures are more complicated. A statistically powerful algorithm is one that can recover the hidden structures for a large region of the model parameter space.

From a computational point of view, we are concerned with the running time of different recovery algorithms. An exhaustive search over the solution space (i.e., all possible clusterings or submatrix locations) may make for a statistically powerful algorithm, but is computationally intractable. A simpler algorithm with lower running time is computationally more desirable, but may succeed only in a smaller region of the model parameter space and thus has weaker statistical power.

Therefore, it is important to take a joint statistical-computational view to the planted clustering and submatrix localization problems, and to understand the *tradeoffs* between these two considerations. How do algorithms with different computational complexity achieve different statistical performance? For these two problems, what are the *information limit* (under what conditions on the model parameters does recovery become infeasible for any algorithm) and the *computational limit* (when does it become infeasible for computationally tractable algorithms)?

We take a step in answering questions in this paper. For both problems, our results demonstrate, in a precise quantitative way, the following phenomenon: the parameter space can be partitioned into four disjoint regions, such that each region corresponds to statistically easier instances of the problem than the previous one, and recovery can be achieved by simpler algorithms with lower running time. Significantly, there might exist a large gap between the statistical performance of computationally intractable algorithms and that of computationally efficient algorithms. We elaborate in the next two subsections.

1.1 Planted Clustering: The Four Regimes

For concreteness, we first consider the planted clustering problem in the setting $r \geq 2$, $p > q$ and $p/q = \Theta(1)$. This covers the standard planted bisection/partition/r-disjoint-clique models.

The statistical hardness of cluster recovery is captured by the quantity $\frac{(p-q)^2}{q(1-q)}$, which is essentially a measure of the Signal-to-Noise Ratio (SNR). Our main theorems identify the following four regimes of the problem defined by the value of this quantity. (Here for simplicity, we use the notation $\dot{\gtrsim}$ and $\dot{\lesssim}$, which ignore constant and $\log n$ factors; note that our main theorems do sharply characterize the $\log n$ factors.)

- *The Impossible Regime:* $\frac{(p-q)^2}{q(1-q)} \dot{\lesssim} \frac{1}{K}$. In this regime, there is no algorithm, regardless of its computational complexity, that can recover the clusters with a vanishing probability of error.
- *The Hard Regime:* $\frac{1}{K} \dot{\lesssim} \frac{(p-q)^2}{q(1-q)} \dot{\lesssim} \frac{n}{K^2}$. There exists a computationally expensive algorithm—specifically the Maximum Likelihood Estimator (MLE)—that recovers the clusters with high probability in this regime (as well as in the next two easier regimes; we omit such implications in the sequel). There is no known polynomial-time algorithm that succeeds in this regime.
- *The Easy Regime:* $\frac{n}{K^2} \dot{\lesssim} \frac{(p-q)^2}{q(1-q)} \dot{\lesssim} \frac{\sqrt{n}}{K}$. There exists a polynomial-time algorithm—specifically a convex relaxation of the MLE—that recovers the clusters with high probability in this regime. Moreover, this algorithm provably fails in the hard regime above.
- *The Simple Regime:* $\frac{(p-q)^2}{q(1-q)} \dot{\gtrsim} \frac{\sqrt{n}}{K}$. A simple algorithm based on counting node degrees and common neighbors recovers the clusters with high probability in this regime, and provably fails outside this regime (i.e., in the hard and easy regimes).

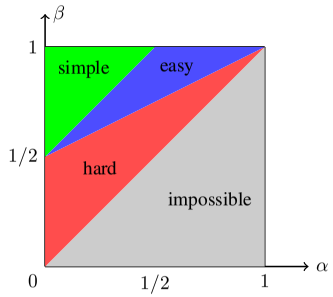


Figure 1: Illustration of the four regimes. The figure applies to the planted clustering problem with $p = 2q = \Theta(n^{-\alpha})$ and $K = \Theta(n^{\beta})$, as well as to the submatrix localization problem with $n_L = n_R = n$, $\mu^2 = \Theta(n^{-\alpha})$ and $K_L = K_R = \Theta(n^{\beta})$.

We illustrate these four regimes in Figure 1 assuming the scaling $p = 2q = \Theta(n^{-\alpha})$ and $K = \Theta(n^{\beta})$ for two constants $\alpha, \beta \in (0, 1)$. Here cluster recovery becomes harder with larger α and smaller β. In this setting, the four regimes correspond to four disjoint and non-empty regions of the parameter space. Therefore, a computationally more expensive algorithm leads to a *significant* (polynomial in n) enhancement in the statistical power. For example, when $\alpha = 1/4$, the simple, polynomial-time, and computationally intractable algorithms succeeds for β larger than 0.75, 0.625, and 0.25, respectively. There is a similar hierarchy for the allowable sparsity of the graph, given by $\alpha < 0.25$, $\alpha < 0.5$, and $\alpha < 0.75$ assuming $\beta = 0.75$.

The results in the impossible and hard regimes together establish the *minimax recovery boundary* of the planted clustering problem, and show that the MLE is statistically order-wise optimal. These two regimes are separated by an "information barrier": in the impossible regime the graph does not carry enough information to distinguish different cluster structures, so recovery is statistically impossible.

Our performance guarantees for the convexified MLE improve upon existing results for polynomial time algorithms in terms of scaling with n, particularly in the setting when the number of clusters are allowed to grow with n.

We *conjecture* that no polynomial-time algorithm can perform significantly better and succeed in the hard regime, i.e., the convexified MLE achieves the *computational limit* order-wise. While we do not prove the conjecture, there are many supporting evidences; cf. Section 2.3. For instance, there is a "spectral barrier", determined by the spectrum of an appropriately defined noise matrix, that prevents the convexified MLE and spectral clustering algorithms from succeeding in the hard regime. In the special setting with a single cluster, the work by Ma and Wu (2015); Hajek et al. (June, 2014) proves that no polynomial-time algorithm can reliably recover the cluster if $\beta < \alpha/4 + 1/2$ conditioned on the planted clique hardness hypothesis.

The simple counting algorithm is an example of an algorithm that uses only "local information": the cluster relation between a pair of nodes is inferred from only their two-hop connection. In

contrast, the convexified MLE crucially uses the global information in the graph spectrum. The counting algorithm fails outside the simple regime due to a "variance barrier" associated with the fluctuations in the node degrees and the numbers of common neighbors, and is statistically order-wise weaker than the global approach of the convexified MLE.

Our main theorems apply beyond the above specific setting and allow for general scalings of p, q, K, and r. The four regimes and the statistical-computational tradeoffs are observed for a broad spectrum of planted problems, including planed partition, planted coloring, planted r-disjoint-clique, and planted densest-subgraph models. Table 1 summarizes the implications of our results for some of these models. More precise and general results are given in Section 2.

1.2 Submatrix Localization: The Four Regimes

Similar results hold for the submatrix localization problem. Consider the setting with $n_L = n_R = n$ and $K_L = K_R = K$. The statistical hardness of submatrix localization is captured by the quantity μ^2, which is again a measure of the SNR. In the high SNR setting with $\mu^2 = \Omega(\log n)$, the submatrices can be trivially identified by element-wise thresholding. In the more interesting low SNR setting with $\mu^2 = O(\log n)$, our main theorems identify the following four regimes, which have the same meanings as before:

- *The Impossible Regime:* $\mu^2 \dot{\lesssim} \frac{1}{K}$. All algorithms fail in this regime.
- *The Hard Regime:* $\frac{1}{K} \dot{\lesssim} \mu^2 \dot{\lesssim} \frac{n}{K^2}$. The computationally expensive MLE succeeds, and it is conjectured that no polynomial-time algorithm succeeds here.
- *The Easy Regime:* $\frac{n}{K^2} \dot{\lesssim} \mu^2 \dot{\lesssim} \frac{\sqrt{n}}{K}$. The polynomial-time convexified MLE succeeds, and provably fails in the hard regime.
- *The Simple Regime:* $\frac{\sqrt{n}}{K} \dot{\lesssim} \mu^2 \dot{\lesssim} 1$. A simple thresholding algorithm succeeds, and provably fails outside this regime.

We illustrate these four regimes in Figure 1 assuming $\mu^2 = \Theta(n^{-\alpha})$ and $K = \Theta(n^{\beta})$. In fact, the results above hold in the more general setting where the entries of A are *sub-Gaussian*.

1.3 Discussions

This paper presents a systematic study of planted clustering and submatrix localization with a growing number of clusters/submatrices. We provide sharp characterizations of the minimax recovery boundary with the lower and upper bounds matching up to constants. We also give improved performance guarantees for convex optimization approaches and the simple counting/thresholding algorithms. In addition, complementary results are given for the *failure conditions* for these algorithms, hence characterizing their performance limits. Our analysis addresses several technical challenges that arise in the high-rank setting. The results in this paper highlight the similarity between planted clustering and submatrix localization, and place several classical problems under a unified framework including planted clique, planted partition, planted coloring, and planted densest subgraph.

The central theme of our investigation is the interaction between the statistical and the computational aspects in the problems, i.e., how to handle more noise and more complicated structures using more computation. Our study parallels a recent line of work that takes a joint statistical and

	Planted r-Disjoint-Clique $1 = p > q \geq 0, r \geq 1$	**Planted Partition/ Stochastic Blockmodel** $1 \geq p > q \geq 0, rK = n$	**Planted Coloring** $0 = p < q \leq 1, rK = n$
Impossible Thm 2, Cor 3	$K \lesssim \left(\frac{q}{1-q} \vee \frac{1}{\log(1/q)}\right) \log n$	$(p-q)^2 \lesssim \frac{p(1-q)}{K} \frac{\log n}{n}$	$q \gtrsim \frac{\log n}{K}$
MLE Thm 4, Cor 5	$K \gtrsim \left(\frac{q}{1-q} \vee \frac{1}{\log(1/q)}\right) \log n$	$(p-q)^2 \gtrsim \frac{p(1-q)}{K} \frac{\log n}{n}$	$q \gtrsim \frac{\log n}{K}$
Convexified MLE Thm 6	$K \gtrsim \frac{\log n}{1-q} + \sqrt{\frac{qn}{1-q}}$	$(p-q)^2 \gtrsim \frac{p(1-q)}{K} \frac{\log n}{n} + \frac{q(1-q)n}{K^2}$	$q \gtrsim \frac{\log n}{K} + \frac{(1-q)n}{K^2}$
Simple Counting Thm 10, Rem 11	$K \gtrsim \frac{\log n}{1-q} + \sqrt{\frac{qn \log n}{1-q}}$	$(p-q)^4 \gtrsim \left[\frac{p^2(1-q)}{K} + \frac{nq(1-q)(q \vee p^2)}{K^2}\right] \log n$	$q^2 \gtrsim \frac{(1-q)n \log n}{K^2}$

Table 1: Our results specialized to different planted models. Here the notations $\gtrsim$ and $\lesssim$ ignore constant factors. This table shows the *necessary conditions* for any algorithm to succeed under a mild assumption $K \gtrsim \log(rK)$, as well as the *sufficient* conditions under which the algorithms in this paper succeed, thus corresponding to the four regimes described in Section 1.1. The relevant theorems/corollaries are also listed. The conditions for convexified MLE and simple counting can further be shown to be also *necessary* in a broad range of settings; cf. Theorems 8 and 12. The results in this table are not the strongest possible; see the referenced theorems for more precise statements.

computational view on inference problems (Balakrishnan et al., 2011a; Oymak et al., 2015; Berthet and Rigollet, 2013; Chandrasekaran and Jordan, 2013; Ma and Wu, 2015); several of these works are closely related to special cases of the planted clustering and bi-clustering models. In this sense, we investigate an emblematic and fundamental problem, and therefore expect that the phenomena and principles described in this paper are relevant more generally. Below we provide additional discussion, and comment on connections with the existing work.

1.3.1 HIGH RANK VERSUS RANK ONE

Several recent works investigate the problems of single-submatrix detection/localization (Kolar et al., 2011; Arias-Castro et al., 2011), planted densest subgraph detection (Arias-Castro and Verzelen, 2014), and sparse principal component analysis (PCA) (Amini and Wainwright, 2009) (cf. Section 1.4 for a literature review). Even earlier is the extensive study of the statistical/computational hardness of Planted Clique. The majority of these results focus on the *rank-one* setting with a single clique, cluster, submatrix or principal component. This paper considers the more general *high-rank* setting, where the number r of clusters/submatrices may grow rapidly with the problem size. This setting is important in many real-world networks (see e.g., Leskovec et al., 2008; Rohe et al., 2011), and poses significant challenges to the analysis. Moreover, there are qualitative differences between these two settings. We discuss one such difference in the next paragraph.

1.3.2 THE POWER OF CONVEX RELAXATION

In previous work on the rank-one case of submatrix detection/localization (Ma and Wu, 2015; Balakrishnan et al., 2011a) and sparse PCA (Krauthgamer et al., 2015), it is shown that simple algorithms based on averaging/thresholding have order-wise similar statistical performance as more sophisticated convex relaxation approaches. In contrast, for the problems of finding multiple clusters/submatrices, we show that convex relaxation of MLE is statistically much more powerful than the simple counting/thresholding algorithm. Our analysis reveals that the power of convex relaxation lies in *separating different clusters/submatrices*, but not in identifying a single cluster/submatrix. Our results thus provide one explanation for the (somewhat curious) observation in previous work regarding the lack of benefit of using sophisticated methods, and demonstrate a finer spectrum of computational-statistical tradeoffs.

1.3.3 DETECTION VERSUS ESTIMATION

Several recent works on planted densest subgraph and submatrix detection have focused on the *detection* or *hypothesis testing* version of the problems, i.e., detecting the existence of a dense cluster or an elevated submatrix (cf. Section 1.4 for literature review). In this paper, we study the (support) *estimation* version of the problems, where the goal is to find the precise locations of the clusters/submatrices. In general estimation appears to be harder than detection. For example, if we consider the scalings of μ and K in Figure 1 of this paper, and compare with Figure 1 in Ma and Wu (2015) which studies submatrix detection, we see that the minimax localization boundary is $\beta = \alpha$, whereas the minimax detection boundary is at a lower value $\beta = \min\{\alpha, \alpha/4 + 1/2\}$. For the planted densest subgraph problem, we see a similar gap between the minimax detection and estimation boundaries if we compare our results with results in Arias-Castro and Verzelen (2014); Hajek et al. (June, 2014). In addition, it is shown in (Ma and Wu, 2015; Hajek et al., June, 2014) that if $\beta > \alpha/4 + 1/2$, the planted submatrix or densest subgraph can be detected in linear time; if

$\beta < \alpha/4+1/2$, no polynomial-time test exists assuming the hardness of the planted clique detection problem. For estimation, we prove the sufficient condition $\beta > \alpha/2+1/2$, which is the best known performance guarantee for polynomial-time algorithms—again we see a gap between detection and estimation. For detecting a sparse principal component, see the seminar work Berthet and Rigollet (2013) for proving computational lower bounds conditioned on the hardness of Planted Clique.

1.3.4 EXTENSIONS

It is a simple exercise to extend our results to a variant of the planted clustering model where the graph adjacency matrix has sub-Gaussian entries instead of Bernoulli, corresponding to a weighted graph clustering problem. Similarly, we can also extend the submatrix location problem to the setting with Bernoulli entries, which is the bi-clustering problem on an unweighted graph and covers the *planted bi-clique* problem (Feldman et al., 2013; Ames and Vavasis, 2011) as a special case.

1.4 Related Work

There is a large body of literature, from the physics, computer science and statistics communities, on models and algorithms for graph clustering and bi-clustering, as well as on their various extensions and applications. A complete survey is beyond the scope of this paper. Here we focus on theoretical work on planted clustering/submatrix localization concerning *exact recovery* of the clusters/submatrices. Detailed comparisons of existing results with ours are provided after we present each of our theorems in Sections 2 and 3. We emphasize that our results are *non-asymptotic* for finite values of n, n_L and n_R, whereas some of the results below require $n \to \infty$.

1.4.1 PLANTED CLIQUE, PLANTED DENSEST SUBGRAPH

The planted clique model ($r = 1$, $p = 1$, $q = 1/2$) is the most widely studied planted model. If the clique has size $K = o(\log n)$, recovery is impossible as the random graph $\mathcal{G}(n, 1/2)$ will have a clique with at least the same size; if $K = \Omega(\log n)$, an exhaustive search succeeds (Alon et al., 1998); if $K = \Omega(\sqrt{n})$, various polynomial-time algorithms work (Alon et al., 1998; Dekel et al., 2014; Deshpande and Montanari, 2013); if $K = \Omega(\sqrt{n \log n})$, the nodes in the clique can be easily identified by counting degrees (Kučera, 1995). It is an open problem to find polynomial-time algorithms which succeed in the regime with $K = o(\sqrt{n})$, and it is believed that this cannot be done (Hazan and Krauthgamer, 2011; Juels and Peinado, 2000; Alon et al., 2007; Feldman et al., 2013). The four regimes above can be considered as a special case of our results for the general planted clustering model. The planted densest subgraph model generalizes the planted clique model by allowing general values of p and q. The detection version of this problem is studied in Arias-Castro and Verzelen (2014); Verzelen and Arias-Castro (2013), and conditional computational hardness results are obtained in Hajek et al. (June, 2014).

1.4.2 PLANTED r-DISJOINT-CLIQUES, PARTITION, AND COLORING

Subsequent work considers the setting with $r \geq 1$ planted cliques (McSherry, 2001), as well as the planted partition model (a.k.a. stochastic block model) with general values of $p > q$ (Condon and Karp, 2001; Holland et al., 1983). A subset of these results allow for a growing number r of clusters (e.g., Rohe et al., 2011). Most existing work focuses on the recovery performance of polynomial-time algorithms. The state-of-the-art for planted r-disjoint-clique are given in McSherry

(2001); Chen et al. (2012); Ames and Vavasis (2014), and for planted partition in Chen et al. (2012); Anandkumar et al. (2014); Cai and Li (2015); see Chen et al. (2014b) for a survey. The setting with $p < q$ is sometimes called the *heterophily* case, with the planted coloring model ($p = 0$) as an important special case (Alon and Kahale, 1997; Coja-Oghlan, 2004). Our results on the convexified MLE (cf. Table 1) improve upon the previously known statistical performance of polynomial-time algorithms in the general r setting. The information-theoretic limits (both lower and upper bounds) of cluster recovery were largely unknown when r is growing. Here we identify these limits up to constant factors for general values of p, q, K and r. In particular, our results show that the information limit is achievable by MLE order-wise, while there is a significant gap between the information limit and the performance guarantee of the convexified MLE.

1.4.3 CONVERSE RESULTS FOR PLANTED PROBLEMS

Complementary to the *achievability* results, another line of work focuses on *converse* results, i.e., identifying necessary conditions for recovery, either for any algorithm, or for any algorithm in a specific class. For the planted partition model with $K = \Theta(n)$, necessary conditions for any algorithm to succeed are obtained by information-theoretic arguments in Chaudhuri et al. (2012); Chen et al. (2012); Balakrishnan et al. (2011b); Abbe et al. (2014). For spectral clustering algorithms and convex optimization approaches, more stringent conditions are shown to be needed (Nadakuditi and Newman, 2012; Vinayak et al., 2014). We generalize and improve upon these existing results in the setting with general r and K.

1.4.4 SHARP THRESHOLDS WITH A BOUNDED NUMBER OF CLUSTERS

Since the conference version of this paper was published (Chen and Xu, 2014), several papers have appeared on the asymptotic information-theoretic limits for exact recovery of planted partition. In the restricted setting with $r = 2$ and $K = n/2$, the recovery thresholds *with sharp constants* are identified and shown to be achievable in polynomial-time in Abbe et al. (2014) for $p, q = O(\log n/n)$, and in Mossel et al. (2015b) for more general scalings of (p, q). Very recently, Abbe and Sandon (2015) established the sharp recovery threshold for the case where $r = O(1)$, $K = \Theta(n)$, and the in-cluster and cross-cluster edge probabilities are heterogeneous and scale as $\log n/n$; the recovery threshold is further shown to be achievable in $o(n^{1+\epsilon})$ time for any $\epsilon > 0$. In this bounded r setting, it is further shown in Hajek et al. (2014); Bandeira (2015); Hajek et al. (2015) that the sharp recovery thresholds are achieved by the semidefinite programming relaxation of the MLE. In comparison, the results in this paper are non-asymptotic and optimal up to universal constant factors, and apply to the general setting with a growing number of clusters/submatrices of size sublinear in n.

1.4.5 APPROXIMATE RECOVERY

While not the focus of this paper, approximate cluster recovery (under various criteria) has also been studied, e.g., for planted partition with $r = O(1)$ clusters in Mossel et al. (2015a, 2013); Massoulié (2014); Yun and Proutiere (2014); Decelle et al. (2011). These results are not directly comparable to ours, but often the approximate recovery conditions differ from the ones for exact recovery by a $\log n$ factor. When constant factors are concerned, the existence of a hard regime was conjectured in Decelle et al. (2011); Mossel et al. (2015a).

1.4.6 SUBMATRIX LOCALIZATION

The statistical-computational tradeoffs in locating a single submatrix (i.e., $r = 1$) are discussed in Balakrishnan et al. (2011a); Kolar et al. (2011); the information limit is shown to be achieved (order-wise) by a computationally intractable algorithm, and the success and failure conditions for various polynomial-time procedures are also derived. The work in Ames (2013) focuses on success conditions for a convex relaxation approach; we improve on their results particularly in the high-rank setting. The single-submatrix *detection* problem is studied in Butucea and Ingster (2013); Shabalin et al. (2009); Sun and Nobel (2013); Arias-Castro et al. (2011); Bhamidi et al. (2012), and conditional hardness results are established in the recent work in Ma and Wu (2015).

1.5 Paper Organization and Notation

The remainder of this paper is organized as follows. In Section 2 we set up the planted clustering model and present our main theorems for the impossible, hard, easy, and simple regimes. In Section 3 we turn to the submatrix localization problem and provide the corresponding theorems for the four regimes. Section 4 provides a brief summary with a discussion of future directions. We prove the main theorems for planted clustering and submatrix localization in Sections 5 and 6, respectively.

The following notation is used in the paper. Let $a \vee b = \max\{a, b\}$ and $a \wedge b = \min\{a, b\}$, and $[m] = \{1, 2, \ldots, m\}$ for any positive integer m. We use c_1, c_2 etc. to denote absolute numerical constants whose values can be made explicit and are independent of the model parameters. We use the standard big-O notations: for two sequences $\{a_n\}, \{b_n\}$, we write $a_n \lesssim b_n$ or $a_n = O(b_n)$ to mean $a_n \le c_1 b_n$ for an absolute constant c_1 and all n; similarly, $a_n \gtrsim b_n$ and $a_n = \Omega(b_n)$ mean $a_n \ge c_2 b_n$. Moreover, $a_n \asymp b_n$ and $a_n = \Theta(b_n)$ mean both $a_n \lesssim b_n$ and $a_n \gtrsim b_n$ hold.

2. Main Results for Planted Clustering

The planted clustering problem is defined by five parameters $n, r, K \in \mathbb{N}$ and $p, q \in [0, 1]$ with $n \ge rK$.

Definition 1 (Planted Clustering) *Suppose n nodes (which are identified with $[n]$) are divided into two subsets V_1 and V_2 with $|V_1| = rK$ and $|V_2| = n - rK$. The nodes in V_1 are partitioned into r disjoint clusters $C_1^*, \ldots, C_r^*$ (called* true clusters*), where $|C_m^*| = K$ for each $m \in [r]$ and $\bigcup_{m=1}^r C_m^* = V_1$. Nodes in V_2 do not belong to any of the clusters and are called* isolated nodes*. A random graph is generated based on the cluster structure: for each pair of nodes and independently of all others, we connect them by an edge with probability p (called* in-cluster edge density*) if they are in the same cluster, and otherwise with probability q (called* cross-cluster edge density*).*

We emphasize again that the values of p, q, r, and K are allowed to be functions of n. The goal is to exactly recover the true clusters $\{C_m^*\}_{m=1}^r$ up to a permutation of cluster indices given the random graph.

The model parameters (p, q, r, K) are assumed to be known to the algorithms. This assumption is often not necessary and can be relaxed (Chen et al., 2012; Arias-Castro and Verzelen, 2014). It is also possible to allow for non-uniform cluster sizes (Ailon et al., 2013) as well as heterogeneous edge probabilities and node degrees (Chaudhuri et al., 2012; Chen et al., 2014b; Cai and Li, 2015). These extensions are certainly important in practical applications; we do not delve into them, and point to the referenced papers above and the references therein for work in this direction.

To facilitate subsequent discussion, we introduce a matrix representation of the planted clustering problem. We represent the true clusters $\{C_m^*\}_{m=1}^r$ by a *cluster matrix* $Y^* \in \{0,1\}^{n\times n}$, where $Y_{ii}^* = 1$ for $i \in V_1$, $Y_{ii}^* = 0$ for $i \in V_2$, and $Y_{ij}^* = 1$ if and only if nodes i and j are in the same true cluster. Note that the rank of Y^* equals r, hence the name of the high-rank setting. The adjacency matrix of the graph is denoted as A, with the convention $A_{ii} = 0, \forall i \in [n]$. Under the planted clustering model, we have $\mathbb{P}(A_{ij} = 1) = p$ if $Y_{ij}^* = 1$ and $\mathbb{P}(A_{ij} = 1) = q$ if $Y_{ij}^* = 0$ for all $i \neq j$. The problem reduces to recovering Y^* given A.

The planted clustering model generalizes several classical planted models.

- *Planted r-Disjoint-Clique* (McSherry, 2001). Here $p = 1$ and $0 < q < 1$, so r cliques of size K are planted into an Erdős-Rényi random graph $G(n, q)$. The special case with $r = 1$ is known as the *planted clique* problem (Alon et al., 1998).

- *Planted Densest Subgraph* (Arias-Castro and Verzelen, 2014). Here $0 < q < p < 1$ and $r = 1$, so there is a subgraph of size K and density p planted into a $G(n, q)$ graph.

- *Planted Partition* (Condon and Karp, 2001). Also known as the *stochastic blockmodel* (Holland et al., 1983). Here $n = rK$ and $p, q \in (0, 1)$. The special case with $r = 2$ can be called *planted bisection* (Condon and Karp, 2001). The case with $p < q$ is sometimes called *planted noisy coloring* or *planted r-cut* (Decelle et al., 2011; Bollobás and Scott, 2004).

- *Planted r-Coloring* (Alon and Kahale, 1997). Here $n = rK$ and $0 = p < q < 1$, so each cluster corresponds to a group of disconnected nodes that are assigned with the same color.

For clarity we shall focus on the homophily setting with $p > q$; results for the $p < q$ case are similar. In fact, any achievability or converse result for the $p > q$ case immediately implies a corresponding result for $p < q$. To see this, observe that if the graph A is generated from the planted clustering model with $p < q$, then the flipped graph $A' := J - A - I$ (J is the all-one matrix and I is the identity matrix) can be considered as generated with in/cross-cluster edge densities $p' = 1-p$ and $q' = 1-q$, where $p' > q'$. Therefore, a problem with $p < q$ can be reduced to one with $p' > q'$. Clearly the reduction can also be done in the other direction.

2.1 The Impossible Regime: Minimax Lower Bounds

In this section, we characterize the necessary conditions for cluster recovery. Let $\mathcal{Y}$ be the set of cluster matrices corresponding to r clusters of size K; i.e.,

$$\mathcal{Y} = \big\{ Y \in \{0,1\}^{n\times n} \,\big|\, \text{there exist disjoint clusters } \{C_m\}_{m=1}^r \text{ such that } |C_m| = K, \forall m \in [r], \\ \text{and } Y \text{ is the corresponding cluster matrix} \big\}.$$

We use $\widehat{Y} \equiv \widehat{Y}(A)$ to denote an estimator which takes as input the graph A and outputs an element of $\mathcal{Y}$ as an estimate of the true Y^*. Our results are stated in terms of the Kullback-Leibler (KL) divergence between two Bernoulli distributions with means u and v, denoted by $D(u\|v) := u \log \frac{u}{v} + (1-u) \log \frac{1-u}{1-v}$. The following theorem gives a lower bound on the minimax error probability of recovering Y^*.

Theorem 2 (Impossible) *Suppose $128 \le K \le n/2$. Under the planted clustering model with $p > q$, if one of the following two conditions holds:*

$$K \cdot D(q\|p) \le \frac{1}{192} \left[\log(rK) \wedge K\right], \tag{1}$$

$$K \cdot D(p\|q) \le \frac{1}{192} \log n, \tag{2}$$

then

$$\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \frac{1}{4},$$

where the infimum ranges over all measurable function of the graph.

The theorem shows it is fundamentally impossible to recover the clusters with success probability close to 1 in the regime where (1) or (2) holds, which is thus called the *impossible regime*. This regime arises from an *information/statistical barrier*: The KL divergence on the LHSs of (1) and (2) determines how much information of Y^* is contained in the data A. If the in-cluster and cross-cluster edge distributions are close (measured by the KL divergence) or the cluster size is small, then A does not carry enough information to distinguish different cluster matrices.

It is sometimes more convenient to use the following corollary, derived by upper-bounding the KL divergence in (1) and (2) using its Taylor expansion. This corollary was used when we overviewed our results in Section 1.1. See table 1 for its implications for specific planted models.

Corollary 3 *Suppose $128 \le K \le n/2$. Under the planted clustering model with $p > q$, if any one of the following three conditions holds:*

$$K(p-q)^2 \le \frac{1}{192} q(1-q) \log n, \tag{3}$$

$$Kp \le \frac{1}{193} \left[\log(rK) \wedge K\right], \tag{4}$$

$$Kp \log \frac{p}{q} \le \frac{1}{192} \log n, \tag{5}$$

then $\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}[\widehat{Y} \neq Y^*] \ge \frac{1}{4}$.

Note the asymmetry between the roles of p and q in the conditions (1) and (2); this is made apparent in Corollary 3. To see why the asymmetry is natural, recall that by a classical result of Grimmett and McDiarmid (1975), the largest clique in a random graph $G(n, q)$ has size $k_q = \Theta(\log n / \log(1/q))$ almost surely. Such a clique cannot be distinguished from a true cluster if $K \lesssim k_q$, even when $p = 1$. This is predicted by the condition (5). When $q = 0$, cluster recovery requires $p \gtrsim \frac{\log(rK)}{K}$ to ensure all true clusters are connected within themselves, matching the condition (4). The term K on the RHS of (1) and (4) is relevant only when $K \le \log(rK)$. Potential improvement on this term is left to future work.

Comparison with previous work: When $r = 1$ and $q = 1/2$, our results recover the $K = \Theta(\log n)$ threshold for the classical planted clique problem. For planted partition with $r = O(1)$ clusters of size $K = \Theta(n)$ and $p/q = \Theta(1)$, the work in Chaudhuri et al. (2012); Chen et al. (2014a) establishes the necessary condition $p - q \lesssim \sqrt{p/n}$; our result is stronger by a logarithmic factor.

The work in Abbe et al. (2014) also considers planted partition with $r = 2$ and focus on the special case with the scaling $p, q = \Theta(\log(n)/n)$; they establish the condition $p+q-2\sqrt{pq} < 2\log(n)/n$, which is consistent with our results up to constants in this regime. Compared to previous work, we handle the more general setting where p, q and r may scale arbitrarily with n.

2.2 The Hard Regime: An Optimal Algorithm

In this subsection, we characterize the sufficient conditions for cluster recovery which match the necessary conditions given in Theorem 2 up to constant factors. We consider the Maximum Likelihood Estimator of Y^* under the planted clustering model, which we now derive. The log-likelihood of observing the graph A given a cluster matrix $Y \in \mathcal{Y}$ is

$$\begin{aligned}\log \mathbb{P}_Y(A) &= \log \prod_{i<j} p^{A_{ij}Y_{ij}} q^{A_{ij}(1-Y_{ij})}(1-p)^{(1-A_{ij})Y_{ij}}(1-q)^{(1-A_{ij})(1-Y_{ij})} \\ &= \log \frac{p(1-q)}{q(1-p)} \sum_{i<j} A_{ij}Y_{ij} + \log \frac{1-p}{1-q} \sum_{i<j} Y_{ij} + \log \frac{q}{1-q} \sum_{i<j} A_{ij} + \sum_{i<j} \log(1-q). \quad (6)\end{aligned}$$

Given A, the MLE maximizes the the log-likelihood over the set $\mathcal{Y}$ of all possible cluster matrices. Note that $\sum_{i<j} Y_{ij} = r\binom{K}{2}$ for all $Y \in \mathcal{Y}$, so the last three terms in (6) are independent of Y. Therefore, the MLE for the $p > q$ case is given as in Algorithm 1.

Algorithm 1 Maximum Likelihood Estimator ($p > q$)

$$\widehat{Y} = \arg\max_{Y} \sum_{i,j} A_{ij}Y_{ij} \quad (7)$$

$$\text{s.t. } Y \in \mathcal{Y}. \quad (8)$$

Algorithm 1 is equivalent to finding r disjoint clusters of size$K$ that maximize the number of edges inside the clusters (similar to Densest K-Subgraph), or minimize the number of edges outside the clusters (similar to Balanced Cut) or the disagreements between A and Y (similar to Correlation Clustering in Bansal et al. 2004). Therefore, while Algorithm 1 is derived from the planted clustering model, it is in fact quite general and not tied to the modeling assumptions. Enumerating over the set $\mathcal{Y}$ is computationally intractable in general since $|\mathcal{Y}| = \Omega(e^{rK})$.

The following theorem provides a success condition for the MLE.

Theorem 4 (Hard) . *Under the planted clustering model with $p > q$, there exists a universal constant c_1 such that for any $\gamma \geq 1$, the optimal solution $\widehat{Y}$ to the problem (7)–(8) is unique and equal to Y^* with probability at least $1 - 16(\gamma rK)^{-1} - 256n^{-1}$ if both of the following hold:*

$$\begin{aligned} K \cdot D(q\|p) &\geq c_1 \log(\gamma rK), \\ K \cdot D(p\|q) &\geq c_1 \log n. \end{aligned} \quad (9)$$

We refer to the regime in which the condition (9) holds but (14) below fails as the *hard regime*, as clustering is statistically possible but conjectured to be computationally hard (cf. Conjecture 9). The conditions (9) above and (1)–(2) in Theorem 2 match up to a constant factor under the mild assumption $K \geq \log(rK)$. This establishes the minimax recovery boundary for planted clustering and the minimax optimality of the MLE up to constant factors.

By lower bounding the KL divergence, we obtain the following corollary, which is sometimes more convenient to use. See Table 1 for its implications for specific planted models.

Corollary 5 *For planted clustering with $p > q$, there exists a universal constant c_2 such that for any $\gamma \geq 1$, the optimal solution $\widehat{Y}$ to the problem (7)–(8) is unique and equal to Y^* with probability at least $1 - 16(\gamma rK)^{-1} - 256n^{-1}$ provided*

$$K(p-q)^2 \geq c_2 q(1-q)\log n, \quad Kp \geq c_2 \log(\gamma rK) \quad \textit{and} \quad Kp \log \frac{p}{q} \geq c_2 \log n. \quad (10)$$

The condition (10) can be simplified to $K(p-q)^2 \gtrsim q(1-q)\log n$ if $q = \Theta(p)$, and to $Kp\log\frac{p}{q} \gtrsim \log n, Kp \gtrsim \log(rK)$ if $q = o(p)$. These match the converse conditions in Corollary 3 up to constants.

Comparison with previous work: Theorem 4 provides the first minimax results (tight up to constant factors) when the number of clusters is allowed to grow, potentially at a *nearly-linear* rate $r = O(n/\log n)$. Interestingly, for a fixed cluster size, the recovery boundary (9) depends very weakly on the number of clusters r though the logarithmic term. For $r = 1$ and $p = 2q = 1$, we recover the recovery boundary for planted clique $K \asymp \log n$. For the planted densest subgraph model where $p/q = \Theta(1)$, p bounded away from 1 and $Kq \gg 1$, the minimax *detection* boundary is shown in Arias-Castro and Verzelen (2014) to be $\frac{(p-q)^2}{q} \asymp \min\{\frac{1}{K}\log\frac{n}{K}, \frac{n^2}{K^4}\}$; our results show that the minimax *recovery* boundary is $\frac{(p-q)^2}{q} \asymp \frac{\log n}{K}$, which is strictly above the detection boundary because $\frac{n^2}{K^4}$ can be much smaller than $\frac{\log n}{K}$. For the planted bisection model with two equal-sized clusters: if $p, q = \Theta(\log(n)/n)$, the sharp recovery boundary is found in Abbe et al. (2014) and Mossel et al. (2015b) to be $K(\sqrt{p}-\sqrt{q})^2 > \log n$, which is consistent with our results up to constants; if $p, q = O(1/n)$, the correlated recovery limit is shown in Mossel et al. (2015a); Massoulié (2014); Mossel et al. (2013) to be $K(p-q)^2 > p+q$, which is consistent with our results up to a logarithmic factor.

2.3 The Easy Regime: Polynomial-Time Algorithms

In this subsection, we present a polynomial-time algorithm for the planted clustering problem and show that it succeeds in the easy regime described in the introduction.

Our algorithm is based on taking a convex relaxation of the MLE in Algorithm 1. Note that the objective function (7) in the MLE is linear, but the constraint $Y \in \mathcal{Y}$ involves a set $\mathcal{Y}$ that is discrete, non-convex and exponentially large. We replace this non-convex constraint with a trace norm (a.k.a. nuclear norm) constraint and a set of linear constraints. This leads to the convexified MLE given in Algorithm 2. Here the trace norm $\|Y\|_*$ is defined as the sum of the singular values of Y. Note that the true Y^* is feasible to the optimization problem (11)–(13) since $\|Y^*\|_* = \text{trace}(Y^*) = rK$.

Algorithm 2 Convexified Maximum Likelihood Estimator ($p > q$)

$$\widehat{Y} = \arg\max_{Y \in \mathbb{R}^{n\times n}} \sum_{i,j} A_{ij} Y_{ij} \tag{11}$$

$$\text{s.t. } \|Y\|_* \le rK, \tag{12}$$

$$\sum_{i,j} Y_{ij} = rK^2, \quad 0 \le Y_{ij} \le 1, \forall i,j. \tag{13}$$

The optimization problem in Algorithm 2 is a semidefinite program (SDP) and can be solved in polynomial time by standard interior point methods or various fast specialized algorithms such as ADMM; e.g., see Jalali and Srebro (2012); Ames (2013). Similarly to Algorithm 1, this algorithm is not strictly tied to the planted clustering model as it can also be considered as a relaxation of Correlation Clustering or Balanced Cut. In the case where the values of r and K are unknown, one may replace the hard constraints (12) and (13) with an appropriately weighted objective function; cf. Chen et al. (2014b).

The following theorem provides a sufficient condition for the success of the convexified MLE. See Table 1 for its implications for specific planted models.

Theorem 6 (Easy) *Under the planted clustering model with $p > q$, there exists a universal constant c_1 such that with probability at least $1 - n^{-10}$, the optimal solution to the problem (11)–(13) is unique and equal to Y^* provided*

$$K^2(p-q)^2 \ge c_1 \left[p(1-q)K \log n + q(1-q)n\right]. \tag{14}$$

When $r = 1$, we refer to the regime where the condition (14) holds and (17) below fails as the *easy regime*. When $r > 1$, the easy regime is where (14) holds and (17) or (18) below fails.

If $p/q = \Theta(1)$, it is easy to see that the smallest possible cluster size allowed by (14) is $K = \Theta(\sqrt{n})$ and the largest number of clusters is $r = \Theta(\sqrt{n})$, both of which are achieved when $p, q, |p-q| = \Theta(1)$. This generalizes the tractability threshold $K = \Omega(\sqrt{n})$ of the classic planted clique problem. If $q = o(p)$ (we call it the high SNR setting), the condition (14) becomes to $Kp \gtrsim \max\{\log n, \sqrt{qn}\}$. In this case, it is possible to go beyond the $\sqrt{n}$ limit on the cluster size. In particular, when $p = \Theta(1)$, the smallest possible cluster size is $K = \Theta(\log n \vee \sqrt{qn})$, which can be much smaller than $\sqrt{n}$.

Remark 7 *Theorem 6 immediately implies guarantees for other tighter convex relaxations. Define the sets $\mathcal{B} := \{Y \mid Eq.(13)\ holds\}$ and*

$$\mathcal{S}_1 := \{Y \mid \|Y\|_* \le rK\},$$
$$\mathcal{S}_2 := \{Y \mid Y \succeq 0; trace(Y) = rK\}.$$

The constraint in Algorithm 2 corresponds to $Y \in \mathcal{S}_1 \cap \mathcal{B}$, while $Y \in \mathcal{S}_2 \cap \mathcal{B}$ is the constraint in a so-called standard SDP relaxation. Clearly $(\mathcal{S}_1 \cap \mathcal{B}) \supseteq (\mathcal{S}_2 \cap \mathcal{B}) \supseteq \mathcal{Y}$. Therefore, if we replace the constraint (12) with $Y \in \mathcal{S}_2$, we obtain a tighter *relaxation of the MLE, and Theorem 6 guarantees that it also succeeds to recover Y^* under the condition (14). The same is true if we consider other tighter relaxations, such as those involving the triangle inequalities (Mathieu and Schudy, 2010),*

the row-wise constraints $\sum_j Y_{ij} \le K, \forall i$ (Ames, 2013), the max norm (Jalali and Srebro, 2012) or the Fantope constraint (Vu et al., 2013). For the purpose of this work, these variants of the convex formulation make no significant difference, and we choose to focus on (11)–(13) for generality.

Comparison with previous work: We refer to Chen et al. (2014b) for a survey of the performance of state-of-the-art polynomial-time algorithms under various planted models. Theorem 6 matches and in many cases improves upon existing results in terms of the scaling. For example, for planted partition with general r, the previous best results are $(p-q)^2 \gtrsim p(K\log^4 n + n)/K^2$ in Chen et al. (2012) and $(p-q)^2 \gtrsim pn\,\mathrm{polylog}\, n/K^2$ in Anandkumar et al. (2014). Theorem 6 removes some extra $\log n$ factors, and is also order-wise better when $q = o(p)$ (the high SNR case) or $1-q = o(1)$. For planted r-disjoint-clique, existing results require $1-q$ to be $\Omega((rn + rK\log n)/K^2)$ (McSherry, 2001), $\Omega(\sqrt{n}/K)$ (Ames and Vavasis, 2014) or $\Omega((n + K\log^4 n)/K^2)$ (Chen et al., 2012). We improve them to $\Omega((n + K\log n)/K^2)$.

2.3.1 CONVERSE FOR THE TRACE NORM RELAXATION APPROACH

We have a partial converse to the achievability result in Theorem 6. The following theorem characterizes the conditions under which the trace norm relaxation (11)–(13) provably fails with high probability; we suspect the standard SDP relaxation with the constraint $Y \in \mathcal{S}_2 \cap \mathcal{B}$ also fails with high probability under the same conditions, but we do not have a proof.

Theorem 8 (Easy, Converse) *Under the planted clustering model with $p > q$, for any constant $1 > \epsilon_0 > 0$, there exist positive universal constants c_1, c_2 for which the following holds. Suppose $c_1 \log n \le K \le \frac{n}{2}$, $q \ge c_1 \frac{\log n}{n}$ and $p \le 1 - \epsilon_0$. If*

$$K^2(p-q)^2 \le c_2(Kp + qn),$$

then with probability at least $1 - n^{-10}$, Y^* is not an optimal solution of the program (11)–(13).

Theorem 8 proves the failure of our trace norm relaxation that has access to the *exact* number and sizes of the clusters. Consequently, replacing the constraints (12) and (13) with a Lagrangian penalty term in the objective would not help for *any* value of the Lagrangian multipliers. Under the assumptions of Theorems 6 and 8, by ignoring log factors, the *sufficient and necessary* condition for the success of our convexified MLE is

$$\frac{p}{K(p-q)^2} + \frac{qn}{K^2(p-q)^2} \dot{\lesssim} 1. \tag{15}$$

We can compare (15) with the success condition (10) for the MLE, which simplifies to

$$\frac{p}{K(p-q)^2} \dot{\lesssim} 1.$$

We see that the convexified MLE is statistically sub-optimal due to the extra second term in (15). This term is responsible for the $K = \Omega(\sqrt{n})$ threshold on the cluster size for the tractability of planted clique. The term has an interesting interpretation. Let $\widetilde{A} := A - q\mathbf{1}\mathbf{1}^\top + qI$ be the centered adjacency matrix. The matrix $E := (Y - \mathbf{1}\mathbf{1}^\top) \circ (\widetilde{A} - \mathbb{E}\widetilde{A})$, [1] i.e., the deviation $\widetilde{A} - \mathbb{E}\widetilde{A}$ restricted to

1. Here $\circ$ denotes the element-wise product.

the inter-cluster node pairs, can be viewed as the "cross-cluster noise matrix". Note that the squared largest singular value of the matrix $\mathbb{E}\widetilde{A} = (p-q)Y^*$ is $K^2(p-q)^2$, whereas the squared largest singular value of E concentrates around $\Theta(qn)$ (see e.g., Chatterjee 2014). Therefore, the second term $\frac{qn}{K^2(p-q)^2}$ in (15) is the "spectral noise-to-signal ratio" that determines the performance of the convexified MLE. In fact, our proofs for Theorems 6 and 8 build on this intuition.

Comparison with previous work: Our converse result in Theorem 8 is inspired by, and improves upon, the recent work in Vinayak et al. (2014), which focuses on the special case $p > 1/2 > q$ and considers a convex relaxation approach that is equivalent to our relaxation (11)–(13) but without the additional equality constraint in (13). The approach is shown to fail when $K^2(p-\frac{1}{2})^2 \lesssim qn$. Our result is stronger in the sense that it applies to a tighter relaxation and a larger region of the parameter space.

2.3.2 LIMITS OF POLYNOMIAL-TIME ALGORITHMS

We compare the minimax recovery threshold in Theorems 2 and 4 with the performance boundary of the polynomial-time convexified MLE in Theorem 6. In general, there exists a substantial gap between these two boundaries (cf. Figure 1). We conjecture that no polynomial-time algorithm has order-wise better statistical performance than the convexified MLE and succeeds significantly beyond the condition (14) in Theorem 6.

Conjecture 9 *For any constant $\epsilon > 0$, there is no algorithm with running time polynomial in n that, for all n and with probability at least $1/2$, outputs the true Y^* of the planted clustering problem with $p > q$ and*

$$(p-q)^2K^2 \le n^{-\epsilon}\left(Kp(1-p) + q(1-q)n\right). \tag{16}$$

If the conjecture is true, then in the asymptotic setting $p = 2q = n^{-\alpha}$ and $K = n^{\beta}$, the *computational limit* for the cluster recovery is given by $\beta = \frac{\alpha}{2} + \frac{1}{2}$, i.e., the boundary between the green and red regimes in Figure 1.

A rigorous proof of Conjecture 9 seems difficult with current techniques. There are other possible convex formulations for planted clustering. The space of possible polynomial-time algorithms is even larger. It is impossible for us to study each of them separately and obtain a converse result as in Theorem 8. There are however several evidences that support the conjecture:

- The special case with $p = 2q = 1$ corresponds to the $K = o(\sqrt{n})$ regime for the classical Planted Clique problem, which is conjectured to be computationally hard (Alon et al., 2007; Rossman, 2010; Feldman et al., 2013), and has been used as an assumption for proving other hardness results (Hazan and Krauthgamer, 2011; Juels and Peinado, 2000; Koiran and Zouzias, 2014). Graphically, the Planted Clique hardness corresponds to the division of the $\alpha = 0$ line of the parameter space shown in Figure 1 (with $r = 1$). Conjecture 9 can be considered as *a generalization of the Planted Clique conjecture* to the whole parameter space, that is, to the setting with multiple clusters and general values of p and q. This generalized conjecture may be used to study the hardness of other problems (Chen, 2015).

- A weaker version of such generalization is proved in Hajek et al. (June, 2014): they show that in the setting with a single cluster, no polynomial-time algorithm can reliably recover the planted clusters if $\beta < \alpha/4 + 1/2$ conditioned on the planted clique hardness hypothesis.

- As discussed earlier, if (16) holds, then the graph spectrum is dominated by noise and fails to reveal the underlying cluster structure. The condition (16) therefore represents a "spectral barrier" for clustering. The work in Nadakuditi and Newman (2012) uses this spectral barrier argument to prove the failure of a large class of algorithms that rely on the graph spectrum. The proof of our Theorem 8 reveals that the convexified MLE fails for a similar reason.

- In the sparse graph case with $p, q = O(1/n)$, it is argued in Decelle et al. (2011), using non-rigorous but deep arguments from statistical physics, that it is intractable to achieve the correlated recovery under Condition (16).

We note that in the "high SNR" case with $\frac{p}{q} \gtrsim \frac{n}{K\log n}$ and $\log K \gtrsim \log n$, the minimax limit and performance boundary of the convexified MLE coincide up to constant factors at $K(p-q)^2 \asymp p(1-q)\log n$. This means, up to constants, the convex relaxation is tight and hence a computationally efficient and statistically order-optimal estimator, so the hard regime disappears. [2] Similar phenomenon can be observed in the setting with linear size clusters $K \asymp n$ (which implies $r \lesssim 1$) and $p \asymp q$, as is illustrated by the $\beta = 1$ line in Figure 1.

2.4 The Simple Regime: A Counting Algorithm

In this subsection, we consider a simple recovery procedure in Algorithm 3, which is based on counting node degrees and common neighbors.

Algorithm 3 A Simple Counting Algorithm

1. (Identify isolated nodes) For each node i, compute its degree d_i. Declare i as isolated if $d_i < \frac{(p-q)K}{2} + qn$.
2. (Identify clusters when $r > 1$) For every pair of non-isolated nodes i, j, compute the number of common neighbors $S_{ij} := \sum_{k:k\neq i,k\neq j} A_{ik}A_{jk}$, and assign them into the same cluster if $S_{ij} > \frac{(p-q)^2K}{3} + 2Kpq + q^2(n-2K)$. Declare error if inconsistency found.

We note that steps 1 and 2 of Algorithm 3 are considered in Kučera (1995) and Dyer and Frieze (1989) respectively for the special cases of recovering a single planted clique or two planted clusters. Let E be the set of edges. It is not hard to see that step 1 runs in time $O(|E|)$ and step 2 runs in time $O(n|E|)$, since each node only needs to look up its local neighborhood up to distance two. It is possible to achieve even smaller expected running time using clever data structures.

The following theorem provides sufficient conditions for the simple counting algorithm to succeed. Compared to the previous work in Kučera (1995); Dyer and Frieze (1989), our results apply to general values of p, q, r, and K. See Table 1 for its implications for specific planted models.

Theorem 10 (Simple) *For planted clustering with $p > q$, there exist universal constants c_1, c_2 such that Algorithm 3 correctly finds the isolated nodes with probability at least $1 - 2n^{-1}$ if*

$$K^2(p-q)^2 \ge c_1[Kp(1-q) + nq(1-q)]\log n, \tag{17}$$

and finds the clusters with probability at least $1 - 4n^{-1}$ if further

$$K^2(p-q)^4 \ge c_2[Kp^2(1-q^2) + nq^2(1-q^2)]\log n. \tag{18}$$

2. The hard regime may still exist if constant factors are concerned; cf. Mossel et al. 2015a; Decelle et al. 2011.

Remark 11 *If $p, q \to 1$ as $n \to \infty$, we can obtain slightly better performance by counting the common* non-*neighbors in Step 2, which succeeds under condition (18) with p and q replaced by $1-p$ and $1-q$, respectively, i.e., the RHS of (18) simplifies to $c_2 n(1-q)^2 \log n$.*

In the case with a single clusters $r = 1$, we refer to the regime where the condition (17) holds as the *simple regime*; in the case with $r > 1$, the simple regime is where both conditions (17) and (18) hold. It is instructive to compare these conditions with the success condition (14) for the convexified MLE. The condition (17) has an additional $\log n$ factor on the RHS. This means when $r = 1$ and the only task is to find the isolated nodes, the counting algorithm performs nearly as well as the sophisticated convexified MLE. On the other hand, when $r > 1$ and one needs to distinguish between different clusters, the convexified MLE order-wise outperforms the counting algorithm whenever $p/q = \Theta(1)$, as the condition (18) is order-wise more restrictive than (14). Nevertheless, when $p, q, p-q = \Theta(1)$, both algorithms can recover $\widetilde{O}(\sqrt{n})$ clusters of size $\widetilde{\Omega}(\sqrt{n})$, making the simple counting algorithm a legitimate candidate in such a setting and a benchmark to which other algorithms can be compared with.

In the high SNR case with $q = o(p)$, the counting algorithm can recover clusters with size much smaller than $\sqrt{n}$; e.g., if $p = \Theta(1)$ and $q = o(1)$, it only requires $K \gtrsim \max\{\log n, \sqrt{qn \log n}\}$.

2.4.1 CONVERSE FOR THE COUNTING ALGORITHM

We have a (nearly-)matching converse to Theorem 10. The following theorem characterizes when the counting algorithm provably fails.

Theorem 12 (Simple, Converse) *Under the planted clustering model with $p > q$, for any constant $0 < \epsilon_0 < 1$, there exist universal constants $c_1, c_2 > 0$ for which the following holds. Suppose $K \le \frac{n}{2}$, $p \le 1 - \epsilon_0$, $q \ge c_1 \log n/n$ and $Kp^2 + nq^2 \ge c_1 \log n$. Algorithm 3 fails to correctly identify all the isolated nodes with probability at least $1/4$ if*

$$K^2(p-q)^2 < c_2 \left[(Kp+nq)\log(rK) + nq\log(n-rK)\right], \tag{19}$$

and fails to correctly recover all the clusters with probability at least $1/4$ if

$$K^2(p-q)^4 < c_2(Kp^2+nq^2)\log(rK). \tag{20}$$

Remark 13 *Theorem 12 requires a technical condition $Kp^2+nq^2 \ge c_1 \log n$, which is actually not too restrictive. If $Kp^2+nq^2 = o(\log n)$, then two nodes from the same cluster will have no common neighbor with probability $(1-p^2)^K(1-q^2)^{n-K} \ge \exp[-\Theta(p^2K+q^2(n-K))] = \exp[-o(\log n)]$, so Algorithm 3 cannot succeed with the probability specified in Theorem 10.*

Apart from some technical conditions, Theorems 10 and 12 show that the conditions (17) and (18) are both sufficient and necessary. In particular, the counting algorithm cannot succeed outside the simple regime, and is indeed strictly weaker in separating different clusters as compared to the convexified MLE. Our proof reveals that the performance of the counting algorithm is limited by a *variance barrier*: The RHS of (17) and (18) are associated with the variance of the node degrees and common neighbors (i.e., d_i and S_{ij} in Algorithm 3), respectively. There exist nodes whose degrees deviate from their expected value on the order of the standard deviation, and if the condition (17) does not hold, then the deviation will outweigh the difference between the expected degrees of the isolated nodes and those of the non-isolated nodes. A similar argument applies to the number of common neighbors.

3. Main Results for Submatrix Localization

In this section, we turn to the submatrix localization problem, sometimes known as bi-clustering (Balakrishnan et al., 2011a). We consider the following specific setting, which is defined by six parameters $n_L, n_R, K_L, K_R, r \in \mathbb{N}$, and $\mu \in \mathbb{R}_+$ such that $n_L \ge rK_L$ and $n_R \ge rK_R$. We use the shorthand notation $n := n_L \vee n_R$.

Definition 14 (Submatrix Localization) *A random matrix $A \in \mathbb{R}^{n_L \times n_R}$ is generated as follows. Suppose that rK_L rows of A are partitioned into r disjoint subsets $\{C_1^*, \dots, C_r^*\}$ of equal size K_L, and rK_R columns of A are partitioned into r disjoint subsets $\{D_1^*, \dots, D_r^*\}$ of equal size K_R. For each (i,j), we have $A_{ij} = \mu + \Delta_{ij}$ if $(i,j) \in C_m^* \times D_m^*$ for some $m \in [r]$ and $A_{ij} = \Delta_{ij}$ otherwise, where $\mu > 0$ is a fixed number and (Δ_{ij}) are i.i.d. zero-mean sub-Gaussian random variables with parameter* 1. [3] *The goal is to recover the locations of the hidden submatrices $\{(C_m^*, D_m^*), m \in [r]\}$ given the matrix A.*

In the language of bi-clustering, the sets $\{C_1^*, \dots, C_r^*\}$ are called *left clusters* and $\{D_1^*, \dots, D_r^*\}$ are called *right clusters*. Row (column, resp.) indices which do not belong to any cluster are called *isolated* left (right, resp.) nodes. One can think of A as the bipartite affinity matrix between the n_L left nodes and n_R right nodes, and the goal is to recover the left and right clusters. Similarly as before, we define the *bi-clustering matrix* $Y^* \in \{0,1\}^{n_L \times n_R}$, where $Y_{ij}^* = 1$ if and only if $(i,j) \in C_m^* \times D_m^*$ for some $m \in [r]$. The problem reduces to recovering Y^* given A.

As before, all the parameters μ, K_L, K_R, r are allowed to scale with n_L and n_R, and we assume that their values are known. Note that it is without loss of generality to assume the mean of A_{ij} is zero outside the submatrices and the variance of A_{ij} is one, because otherwise we can shift and rescale A. The above model generalizes the previous submatrix localization/detection models (Ma and Wu, 2015; Butucea and Ingster, 2013; Arias-Castro et al., 2011) and bi-clustering models (Kolar et al., 2011; Balakrishnan et al., 2011a) which consider the special case with a *single* submatrix (i.e., $r = 1$).

In the next four subsections, we shall focus on the low-SNR setting $\mu^2 = O(\log n)$ and present theorems establishing the four regimes. These results parallel those for the planted clustering. In the high SNR setting $\mu^2 = \Omega(\log n)$, the submatrices can be easily identified by naive element-wise thresholding, so we deal with this case separately in the last subsection.

3.1 The Impossible Regime: Minimax Lower Bounds

The following theorem gives conditions on $(n_L, n_R, K_L, K_R, \mu)$ under which the minimax error probability is large and thus it is statistically impossible to reliably locate the submatrices. With slight abuse of notation, we use $\mathcal{Y} \subset \{0,1\}^{n_L \times n_R}$ to denote the set of all possible bi-clustering matrices corresponding to r left (right, resp.) clusters of equal size K_L (K_R, resp.).

Theorem 15 (Impossible) *Under the submatrix localization model, suppose $\{A_{ij}\}$ are Gaussian random variables, $K_L \le n_L/2$, $K_R \le n_R/2$, and $n_L, n_R \ge 128$. If*

$$\mu^2 \le \frac{1}{12} \max\left\{ \frac{\log(n_R - K_R)}{K_L}, \frac{\log(n_L - K_L)}{K_R} \right\}, \tag{21}$$

then $\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}\left[\widehat{Y} \ne Y^*\right] \ge \frac{1}{2}$, where the infimum ranges over all measurable functions of A.

3. A random variable X is said to be sub-Gaussian with parameter 1 if $\mathbb{E}[e^{tX}] \le e^{t^2/2}$ for all $t \in \mathbb{R}$.

The regime where (21) holds is called the *impossible* regime, corresponding to an information barrier that no algorithm can break. We note the similarity between the impossible regimes for submatrix localization and planted clustering. In particular, if we assume the in/cross-cluster edges in planted clustering have comparable variance, i.e., $\frac{p(1-p)}{q(1-q)} = \Theta(1)$, then the conditions (21) and (3) coincide up to constant factors by setting $n_L = n_R = n, K_L = K_R = K$, and $\mu = \frac{p-q}{\sqrt{q(1-q)}}$. Such correspondence also exists in the next three regimes.

Comparison with previous work: Theorem 15 holds in the general high rank setting with arbitrary $r \geq 1$. In $r = 1$ case, our result recovers the minimax lower bound in Kolar et al. (2011).

3.2 The Hard Regime: Optimal Algorithm

Recall that $\mathcal{Y}$ is the set of all valid bi-clustering matrices. We consider the combinatorial optimization problem given in Algorithm 4. In the setting where $\{\Delta_{ij}\}$ are Gaussian random variables, this can be shown to be the MLE of Y^*.

Algorithm 4 Maximum Likelihood Estimator

$$\widehat{Y} = \arg\max_{Y \in \mathcal{Y}} \sum_{i,j} A_{ij} Y_{ij}. \tag{22}$$

Theorem 16 below provides a success condition for Algorithm 4.

Theorem 16 (Hard) *Suppose $K_L, K_R \geq 8$. There exists a constant c_1 such that with probability at least $1 - 512en^{-1}$, the optimal solution to the problem (22) is unique and equals Y^* if*

$$\mu^2 \geq c_1 \frac{\log n}{K_L \wedge K_R}. \tag{23}$$

We refer to the regime where the condition (23) holds and (27) fails as the *hard* regime. Note that the bound (23) matches (21) up to a constant factor, so Algorithm 4 is minimax optimal up to a constant factor. Theorems 15 and 16 together establish the minimax recovery boundary for submatrix localization at $\mu^2 \asymp \frac{\log n}{K_L \wedge K_R}$.

Comparison with previous work: Theorem 16 provides the first minimax-optimal achievability result when the number r of submatrices may grow with n_L and n_R. In particular, r is allowed to grow at a nearly linear rate $r = O(n/\log n)$ assuming $n_L = n_R = n$. In the special case with a single planted submatrix ($r = 1$), Theorem 16 recovers the achievability result in Kolar et al. (2011).

3.3 The Easy Regime: Polynomial-Time Algorithms

As previous, we obtain a convex relaxation of the combinatorial MLE formulation (22) by replacing the constraint $Y \in \mathcal{Y}$ with the trace norm and linear constraints, for which we use the fact that the true Y^* satisfies $\|Y^*\|_* = r\sqrt{K_L K_R}$. This is given as Algorithm 5, which is a semidefinite program (SDP) and can be solved in polynomial time.

Algorithm 5 Convexified Maximum Likelihood Estimator

$$\widehat{Y} = \arg\max_{Y \in \mathbb{R}^{n \times n}} \sum_{i,j} A_{ij} Y_{ij} \tag{24}$$

$$\|Y\|_* \leq r\sqrt{K_L K_R}, \tag{25}$$

$$\sum_{i,j} Y_{ij} = rK_L K_R, \quad 0 \leq Y_{ij} \leq 1, \forall i, j. \tag{26}$$

The following theorem provides a sufficient condition for the success of Algorithm 5.

Theorem 17 (Easy) *There exists a universal constant c_1 such that with probability at least $1 - n^{-10}$, the optimal solution to the program (24)–(26) in Algorithm 5 is unique and equals Y^* if*

$$\mu^2 \geq c_1 \left(\frac{\log n}{K_L \wedge K_R} + \frac{n}{K_L K_R} \right). \tag{27}$$

When $r = 1$, the *easy regime* refers to where the condition (27) holds but (29) fails. When $r > 1$, the easy regime is where the condition (27) holds but (30) fails. Suppose $n_L = n_R = n$ and $K_L = K_R = K$; the convexified MLE is guaranteed to succeed when $\mu^2 \gtrsim \frac{K \log n + n}{K^2}$.

The following theorem provides a nearly matching converse to Theorem 17.

Theorem 18 (Easy, Converse) *There exist positive universal constants c_1, c_2 such that the following holds. Under the submatrix localization model, suppose $\mu \leq 1/100$, $n_L = n_R = n$, $K_L = K_R = K$, $c_1 \log n \leq K \leq \frac{n}{2}$, and (Δ_{ij}) are Gaussian random variables. If*

$$\mu^2 \leq c_2 \frac{n}{K^2}, \tag{28}$$

then with probability at least $1 - n^{-10}$, Y^* is not an optimal solution of the program (24)–(26).

Theorems 17 and 18 together establish that the recovery boundary for the convexified MLE in Algorithm 5 is $\mu^2 \asymp \frac{n}{K^2}$ ignoring logarithmic factors. There is a substantial gap from the minimax boundary $\mu^2 \asymp \frac{1}{K}$ established in the last two subsections (again ignoring logarithmic factors). Our analysis reveals that the performance of the convexified MLE is determined by a spectral barrier similar to that in planted clustering. In particular, the squared largest singular values of the *signal matrix* Y^* and the *noise matrix* $A - \mathbb{E}A$ are $\Theta(\mu^2 K^2)$ and $\Theta(n)$, respectively, so the condition $\mu^2 \gtrsim \frac{n}{K^2}$ for the convexified MLE can be seen as an spectral SNR condition.

As in the planted clustering model, we conjecture that no polynomial-time algorithm can achieve better statistical performance than the convexified MLE.

Conjecture 19 *For any constant $\epsilon > 0$, there is no algorithm with running time polynomial in n that, for all n and with probability at least $1/2$, outputs the true Y^* for the submatrix localization problem with $\mu \leq 1$, $n_L = n_R = n$, $K_L = K_R = K \geq c_1 \log n$ and*

$$\mu^2 \leq \frac{n^{1-\epsilon}}{K^2}.$$

Comparison with previous work: The achievability and converse results in Theorems 17 and 18 hold even when r grows with n. In the special case with $r = 1$, the work in Kolar et al. (2011) considers a convex relaxation of sparse singular value decomposition; they focus on the high SNR regime with $\mu^2 \gtrsim \log n$, and show that the performance of their convex relaxation is no better than a simple element-wise thresholding approach (cf. Section 3.5). Our convex program is different from theirs, and succeeds in the low SNR regime provided $\mu^2 \gtrsim \frac{K \log n + n}{K^2}$. The work in Ames (2013) studies the success conditions of a convex formulation similar to Kolar et al. (2011); with the additional assumption of bounded support of the distribution of A_{ij}, they show that their approach succeeds under an order-wise more restricted condition $\mu^2 \gtrsim \frac{n \cdot r}{K^2}$.

3.4 The Simple Regime: A Thresholding Algorithm

We consider a simple thresholding algorithm as given in Algorithm 6. The algorithm computes the column and row sums of A as well as the correlation between the columns and rows. It is similar in spirit to the simple counting Algorithm 3 for the planted clustering problem.

Algorithm 6 A Simple Thresholding Algorithm

1. (Identify isolated nodes) For each left node $i \in [n_L]$, declare it as isolated if the row sum $d_i := \sum_{j=1}^{n_R} A_{ij} \leq \frac{\mu K_R}{2}$. For each right node $j \in [n_R]$, declare it as isolated if the column sum $d'_j := \sum_{i=1}^{n_L} A_{ij} \leq \frac{\mu K_L}{2}$.
2. (Identify clusters when $r > 1$) For each pair of non-isolated left nodes $i, i' \in [n_L]$, assign them to the same cluster if $S_{ii'} := \sum_{j=1}^{n_R} A_{ij} A_{i'j} \geq \frac{\mu^2 K_R}{2}$. Declare error if inconsistency is found. Assign the non-isolated right nodes into clusters in a similar manner. Let $\{C_k\}$ and $\{D_k\}$ be the resulting left and right clusters.
3. (Associate left and right clusters) For each $k \in [r]$ and $l \in [r]$, associate the left cluster C_k with the right cluster D_l if the block sum $B_{kl} := \sum_{i \in C_k, j \in D_l} A_{ij} \geq \mu K_L K_R / 2$.

Steps 1, 2 and 3 of the algorithm run in time $O(n_L n_R)$, $O(n_L^2 n_R + n_R^2 n_L)$ and $O(n_L n_R)$, respectively. We note that Step 1 is previously considered in Kolar et al. (2011) for locating a single submatrix. The following theorem provides success conditions for this simple algorithm.

Theorem 20 (Simple) *There exist universal constants c_1, c_2 such that Algorithm 6 identifies the isolated nodes with probability at least $1 - en_L^{-1} - en_R^{-1}$ if*

$$\mu^2 \geq c_1 \max\left\{ \frac{n_L \log n_R}{K_L^2}, \frac{n_R \log n_L}{K_R^2} \right\}, \tag{29}$$

and exactly recovers Y^* with probability at least $1 - e(rK_L)^{-1} - e(rK_R)^{-1} - en^{-1}$ if further

$$\mu^4 \geq c_2 \max\left\{ \frac{n_L \log(rK_R)}{K_L^2}, \frac{n_R \log(rK_L)}{K_R^2} \right\}. \tag{30}$$

When $r = 1$, we refer to the regime for which the condition (29) holds as the *simple* regime. When $r > 1$, the simple regime is where both conditions (29) and (30) hold.

We provide a converse to Theorem 20. The following theorem shows that the conditions (29) and (30) are also (nearly) necessary for the simple thresholding algorithm to succeed.

Theorem 21 (Simple, Converse) *Under the submatrix localization model where the distributions of $\{A_{ij}\}$ are Gaussian, there exist universal constants c_1, c_2 such that with probability at least $1 - e^{-(rK_L)^{\Omega(1)}} - e^{-(rK_R)^{\Omega(1)}}$, Algorithm 6 fails to correctly identify all the isolated nodes if*

$$\mu^2 \leq c_1 \max\left\{ \frac{n_L \log n_R}{K_L^2}, \frac{n_R \log n_L}{K_R^2} \right\}, \tag{31}$$

and fails to correctly recover all the clusters if $n_L = \Omega(rK_R)$, $n_R = \Omega(rK_L)$ and

$$\mu^4 \leq c_2 \max\left\{ \frac{n_L \log(rK_R)}{K_L^2}, \frac{n_R \log(rK_L)}{K_R^2} \right\}. \tag{32}$$

When $n_L = n_R = n$, $K_L = K_R = K$, Theorems 20 and Theorem 21 establish that the recovery boundary for the simple thresholding algorithm is $\mu^2 \asymp \frac{n \log n}{K^2}$ if $r = 1$, and $\mu^2 \asymp \frac{\sqrt{n \log n}}{K}$ if $r > 1$ and $rK = \Theta(n)$. Comparing with the success condition (27) for the convex optimization approach, we see that the simple thresholding algorithm is order-wise less powerful in separating different submatrices. Similar to planted clustering, the performance is determined by the variance barrier associated with the variance of the quantities d_i and $S_{ii'}$ computed in Algorithm 6.

3.5 The High SNR Setting

As mentioned before, the high SNR setting with $\mu^2 = \Omega(\log n)$ can be handled by a simple element-wise thresholding algorithm, which is given in Algorithm 7.

Algorithm 7 Element-wise Thresholding for Submatrix Localization

For each $(i, j) \in [n_L] \times [n_R]$, set $\widehat{Y}_{ij} = 1$ if $A_{ij} \geq \frac{1}{2}\mu$, and $\widehat{Y}_{ij} = 0$ otherwise. Output $\widehat{Y}$.

For the special case with one submatrix ($r = 1$), the success of element-wise thresholding in the high SNR setting is proved in Kolar et al. (2011). Their result can be easily extended to the general case with $r \geq 1$. We record this extension in Theorem 22 below. The theorem also shows that element-wise thresholding fails if $\mu^2 = o(\log n)$, so it is not very useful in the low SNR setting.

Theorem 22 (Element-wise Thresholding) *There exists a universal constant $c_1 > 4$ such that the following holds. Algorithm 7 outputs $\widehat{Y} = Y^*$ with probability at least $1 - n^{-3}$ provided*

$$\mu^2 > c_1 \log n. \tag{33}$$

If the distributions of the A_{ij}'s are Gaussian, then with probability at least $1 - n^{-3}$, the output of Algorithm 7 satisfies $\widehat{Y} \neq Y^*$ provided

$$\mu^2 \leq 4 \log n. \tag{34}$$

4. Discussion and Future Work

In this paper, we show that the planted clustering problem and the submatrix localization problem admit successively faster algorithms with weaker statistical performance. We provide sufficient and

necessary conditions for the success of the intractable MLE, the convexified MLE and the simple counting/thresholding algorithm, showing that they work in progressively smaller regions of the parameter space. This thus represents a series of tradeoffs between the statistical and computational performance. Our results hold in the high-rank setting with a growing number of clusters or submatrices. Our results indicate that there may exist a large gap between the information limit and the computational limit, i.e., the information limit might not be achievable via polynomial-time algorithms.

Several future directions are of interest. Immediate goals include removing some of the technical assumptions in our theorems. It is useful in practice to identify a finer spectrum, ideally close to a continuum, of computational-statistical tradeoffs. It is also interesting to extend to the settings with overlapping clusters and submatrices, and to the those where the values of the model parameters are unknown. Proving our conjectures on the computational hardness in the hard regime is also interesting, and this direction is pursued in Ma and Wu (2015); Hajek et al. (June, 2014).

5. Proofs for Planted Clustering

Throughout this section, we consider the planted clustering model with $p > q$. Let $n_1 := rK = |V_1|$ and $n_2 := n - rK = |V_2|$ be the numbers of non-isolated and isolated nodes, respectively.

5.1 Proof of Theorem 2 and Corollary 3

In the sequel we will make use of the following upper and lower bounds on the KL divergence $D(u\|v)$ between two Bernoulli distributions with parameter $u \in [0,1]$ and $v \in [0,1]$. We have

$$D(u\|v) := u\log\frac{u}{v} + (1-u)\log\frac{1-u}{1-v} \overset{(a)}{\le} u\frac{u-v}{v} + (1-u)\frac{v-u}{1-v} = \frac{(u-v)^2}{v(1-v)}, \tag{35}$$

where (a) follows from the inequality $\log x \le x-1, \forall x \ge 0$. Moreover, viewing $D(x\|v)$ as a function of x and using the Taylor's expansion, we can find some $\xi \in [u\wedge v, u\vee v]$ such that

$$D(u\|v) = D(v\|v) + (u-v)D'(v\|v) + \frac{(u-v)^2}{2}D''(\xi\|v) \overset{(b)}{\ge} \frac{(u-v)^2}{2(u\vee v)[1-(u\wedge v)]}, \tag{36}$$

where (b) follows from $D'(v\|v) = 0$ and $D''(\xi\|v) = 1/[\xi(1-\xi)]$.

Theorem 2 is established through the following three lemmas, each of which provides a sufficient condition for having a non-vanishing error probability.

Lemma 23 *Suppose that* $128 \le K \le n/2$. *Let* $\delta := \frac{n_1(K-1)}{n(n-1)}$ *and* $\bar{p} := \delta p + (1-\delta)q$. *We have* $\inf_{\widehat{Y}}\sup_{Y^*\in\mathcal{Y}}\mathbb{P}\left[\widehat{Y} \ne Y^*\right] \ge \frac{1}{2}$ *if*

$$K\cdot D(p\|\bar{p}) + \frac{n^2}{n_1}(1-\delta)\frac{(q-\bar{p})^2}{\bar{p}(1-\bar{p})} \le \frac{1}{4}\log\frac{n}{K}. \tag{37}$$

Moreover, the condition (37) *is implied by*

$$K(p-q)^2 \le \frac{1}{4}q(1-q)\log\frac{n}{K}. \tag{38}$$

Proof We use an information theoretical argument via Fano's inequality. Recall that $\mathcal{Y}$ is the set of all cluster matrices corresponding to r clusters of size K. Let $\mathbb{P}_{(Y^*,A)}$ be the joint distribution of (Y^*, A) when Y^* is sampled from $\mathcal{Y}$ uniformly at random and then A is generated according to the planted clustering model with Y^* being the true cluster matrix. Lower-bounding the supremum by the average, we have

$$\inf_{\widehat{Y}}\sup_{Y^*\in\mathcal{Y}}\mathbb{P}\left[\widehat{Y} \ne Y^*\right] \ge \inf_{\widehat{Y}}\mathbb{P}_{(Y^*,A)}\left[\widehat{Y} \ne Y^*\right].$$

Therefore it suffices to bound $\mathbb{P}_{(Y^*,A)}\left[\widehat{Y} \ne Y^*\right]$ from below. Let $H(X)$ denote the entropy of a random variable X and $I(X;Z)$ the mutual information between X and Z. By Fano's inequality, we have for any $\widehat{Y}$,

$$\mathbb{P}_{(Y^*,A)}\left[\widehat{Y} \ne Y^*\right] \ge 1 - \frac{I(Y^*;A)+1}{\log|\mathcal{Y}|}. \tag{39}$$

We first bound $\log|\mathcal{Y}|$. Simple counting gives that $|\mathcal{Y}| = \binom{n}{n_1}\frac{n_1!}{r!(K!)^r}$. Note that $\binom{n}{n_1} \ge (\frac{n}{n_1})^{n_1}$ and $\sqrt{n}(\frac{n}{e})^n \le n! \le e\sqrt{n}(\frac{n}{e})^n$. It follows that

$$|\mathcal{Y}| \ge (n/n_1)^{n_1}\frac{\sqrt{n_1}(n_1/e)^{n_1}}{e\sqrt{r}(r/e)^r e^r K^{r/2}(K/e)^{n_1}} \ge \left(\frac{n}{K}\right)^{n_1}\frac{1}{e(r\sqrt{K})^r}.$$

This implies $\log|\mathcal{Y}| \ge \frac{1}{2}n_1\log\frac{n}{K}$ under the assumption that $8 \le K \le n/2$ and $n \ge 32$.

We next upper bound $I(Y^*;A)$. Note that $H(A) \le \binom{n}{2}H(A_{12})$ because the A_{ij}'s are identically distributed by symmetry. Furthermore, the A_{ij}'s are mutually independent conditioned on Y^*, so $H(A|Y^*) = \binom{n}{2}H(A_{12}|Y^*_{12})$. It follows that $I(Y^*;A) = H(A) - H(A|Y^*) \le \binom{n}{2}I(Y^*_{12};A_{12})$. We can bound $I(Y^*_{12};A_{12})$ below. Direct counting gives

$$\mathbb{P}(Y^*_{12}=1) = \frac{\binom{n-2}{K-2}\binom{n-K}{K}\cdots\binom{n-rK+K}{K}\frac{1}{(r-1)!}}{|\mathcal{Y}|} = \frac{n_1(K-1)}{n(n-1)} = \delta,$$

and thus $\mathbb{P}(A_{12}=1) = \bar{p} := \delta p + (1-\delta)q$. Therefore, $I(Y^*_{12};A_{12}) = \delta D(p\|\bar{p}) + (1-\delta)D(q\|\bar{p})$. Using the upper bound (35) on the KL divergence and the condition (37), we obtain

$$I(Y^*_{12};A_{12}) = \delta D(p\|\bar{p}) + (1-\delta)D(q\|\bar{p}) \le \delta D(p\|\bar{p}) + (1-\delta)\frac{(q-\bar{p})^2}{\bar{p}(1-\bar{p})} \le \frac{n_1}{4n^2}\log\frac{n}{K}.$$

It follows that $I(Y^*;A) \le \binom{n}{2}I(Y^*_{12}\|A_{12}) \le \frac{n_1}{8}\log\frac{n}{K}$. Substituting into (39) gives

$$\mathbb{P}_{(Y^*,A)}[Y \ne Y^*] \ge 1 - \frac{\frac{n_1}{4}\log\frac{n}{K}+2}{n_1\log\frac{n}{K}} = \frac{3}{4} - \frac{2}{n_1\log\frac{n}{K}} \ge \frac{1}{2},$$

where the last inequality holds because $K \le n/2$ and $n_1 \ge 32$. This proves the sufficiency of (37).

We turn to the second part of the lemma. Observe that

$$\begin{aligned} K\cdot D(p\|\bar{p}) + \frac{n^2}{n_1}(1-\delta)\frac{(q-\bar{p})^2}{\bar{p}(1-\bar{p})} &\overset{(a)}{\le} K\frac{(p-\bar{p})^2}{\bar{p}(1-\bar{p})} + \frac{K}{\delta}(1-\delta)\frac{(q-\bar{p})^2}{\bar{p}(1-\bar{p})} \\ &= K\frac{\delta(1-\delta)(p-q)^2}{\bar{p}(1-\bar{p})} \overset{(b)}{\le} \frac{K(p-q)^2}{q(1-q)}, \end{aligned}$$

where (a) holds due to $\delta \le \frac{n_1 K}{n^2}$ and (35), and (b) holds because $\bar{p}(1-\bar{p}) \ge \delta p(1-p) + (1-\delta)q(1-q) \ge (1-\delta)q(1-q)$ thanks to the concavity of $x(1-x)$. Combining the last display equation with (38) gives (37). ■

Lemma 24 *Suppose* $128 \le K \le n/2$. *We have* $\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \frac{1}{2}$ *if*

$$K \max\{D(p\|q), D(q\|p)\} \le \frac{1}{24}\log(n-K). \tag{40}$$

Proof Let $\bar{M} := n - K$, and $\bar{\mathcal{Y}} := \{Y_0, Y_1, \ldots, Y_{\bar{M}}\}$ be a subset of $\mathcal{Y}$ with cardinality $\bar{M}+1$ to be specified later. Let $\bar{\mathbb{P}}_{(Y^*,A)}$ denote the joint distribution of (Y^*, A) when we first sample Y^* from $\bar{\mathcal{Y}}$ uniformly at random, and then sample A according to the planted clustering model with Y^* being the true cluster matrix. By Fano's inequality, we have

$$\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \inf_{\widehat{Y}} \bar{\mathbb{P}}_{(Y^*,A)}\left[\widehat{Y} \neq Y^*\right] \ge 1 - \frac{I(Y^*; A) + 1}{\log|\bar{\mathcal{Y}}|}. \tag{41}$$

We construct $\bar{\mathcal{Y}}$ as follows. Let Y_0 be the cluster matrix such that the clusters $\{C_l\}_{l=1}^r$ are given by $C_l = \{(l-1)K+1, \ldots, lK\}$. Informally, each Y_i with $i \ge 1$ is obtained from Y_0 by swapping the cluster memberships of the nodes K and $K+i$. Formally, for each $i \in [\bar{M}]$: (1) if the node $(K+i)$ belongs to cluster C_l for some l, then Y_i is the cluster matrix for which the first cluster consists of the nodes $\{1, 2, \ldots, K-1, K+i\}$, the l-th cluster is given by $C_l \setminus \{K+i\} \cup \{K\}$, and all the other clusters are identical to those given by Y_0; (2) if the node $(K+i)$ is an isolated node in Y_0 (i.e., it does not belong to any cluster), then Y_i is the cluster matrix for which the first cluster consists of the nodes $\{1, 2, \ldots, K-1, K+i\}$, the node K is an isolated node, and all the other clusters identical to those given by Y_0.

Let $\mathbb{P}_i$ be the distribution of the graph A conditioned on $Y^* = Y_i$. Note that each $\mathbb{P}_i$ is the product of $\frac{1}{2}n(n-1)$ Bernoulli distributions. We have the following chain of inequalities:

$$I(Y^*; A) \overset{(a)}{\le} \frac{1}{(\bar{M}+1)^2}\sum_{i,i'=0}^{\bar{M}} D\left(\mathbb{P}_i \| \mathbb{P}_{i'}\right) \overset{(b)}{\le} 3K \cdot D\left(p\|q\right) + 3K \cdot D\left(q\|p\right),$$

where (a) follows from the convexity of KL divergence, and (b) follows by our construction of $\{Y_i\}$. If the condition (40) in the lemma holds, then $I(Y^*; A) \le \frac{1}{4}\log(n-K) = \frac{1}{4}\log|\bar{\mathcal{Y}}|$. Since $\log(n-K) \ge \log(n/2) \ge 4$ when $n \ge 128$, it follows from (41) that the minimax error probability is at least $1/2$. ■

Lemma 25 *Suppose* $128 \le K \le n/2$. *We have* $\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \frac{1}{4}$ *if*

$$Kp \le \frac{1}{8}\min\{\log(rK/2), K\}, \tag{42}$$

$$\textbf{\textit{or}} \quad K(1-q) \le \frac{1}{4}\log K. \tag{43}$$

Proof We first prove the sufficiency of the condition (42). We call a node a *disconnected node* if it is not connected to any other node in its own cluster. Let E be the event that there exist two disconnected nodes from two different clusters, and set $\rho := \mathbb{P}[E|Y^*]$, which is in fact independent of what value Y^* takes in $\mathcal{Y}$. Suppose Y^* is uniformly distributed over $\mathcal{Y}$; we claim that $\mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \rho/2$ for all $\widehat{Y}$. To see this, consider the maximum likelihood estimator (MLE) of Y^*, which is given by $\widehat{Y}_{\mathrm{ML}}(a) := \arg\max_y \mathbb{P}[A = a | Y^* = y]$ with tie broken uniformly at random. It is a standard fact that the MLE minimizes the error probability under the uniform prior, so for all $\widehat{Y}$ we have

$$\mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \frac{1}{|\mathcal{Y}|}\sum_{y \in \mathcal{Y}}\sum_{a \in \{0,1\}^{n\times n}} \mathbb{P}\left[\widehat{Y}_{\mathrm{ML}}(a) \neq y\right]\mathbb{P}\left[A = a | Y^* = y\right]. \tag{44}$$

Let $\mathcal{A}_y \subseteq \{0,1\}^{n\times n}$ denote the set of adjacency matrices with at least two disconnected nodes in two different clusters with respect to the clusters defined by $y \in \mathcal{Y}$. For each $a \in \mathcal{A}_y$, let $y'(a)$ denote the cluster matrix obtained by swapping the two rows and columns of y corresponding to the first two disconnected nodes in a. It is easy to check that for each $a \in \mathcal{A}_y$, the likelihood satisfies $\mathbb{P}[A = a|Y^* = y] \le \mathbb{P}[A = a|Y^* = y'(a)]$ and therefore $\mathbb{P}[\widehat{Y}_{\mathrm{ML}}(a) \neq y] \ge 1/2$. It follows from (44) that

$$\mathbb{P}\left[\widehat{Y} \neq Y^*\right] \ge \frac{1}{|\mathcal{Y}|}\sum_{y}\sum_{a \in \mathcal{A}_y}\frac{1}{2}\cdot\mathbb{P}[A = a|Y^* = y] = \frac{1}{2}\rho,$$

where the last equality holds because $\mathbb{P}[\mathcal{A}_y|Y^* = y] \equiv \rho$ independently of y. This proves our claim.

Since the maximum error probability is lower bounded by the average error probability, to establish the lemma it suffices to show $\rho \ge 1/2$. Without loss of generality, suppose r is even and we fix the clusters Y^* to be such that the first $rK/2$ nodes $\{1, \ldots, rK/2\}$ form $r/2$ clusters, and the next $rk/2$ nodes $\{rK/2+1, \ldots, rK\}$ form another $r/2$ clusters. For each $i \in [rK/2]$, let ξ_i be the indicator random variable for node i being a disconnected node. Then $\rho_1 := \mathbb{P}\left[\sum_{i=1}^{rK/2}\xi_i \ge 1\right]$ is the probability that there exits at least one disconnected node among the first $rK/2$ nodes. We use a second moment argument (Durrett, 2007) to lower-bound ρ_1. Observe that $\xi_1, \ldots, \xi_{rK/2}$ are (dependent) Bernoulli variables with mean $\mu := (1-p)^{K-1}$. For $i \neq j$, we have

$$\mathbb{E}\left[\xi_i\xi_j\right] = \mathbb{P}\left[\xi_i = 1, \xi_j = 1\right] \le (1-p)^{2K-3} = \frac{1}{1-p}\mu^2.$$

Therefore, we obtain

$$\begin{aligned}\mathrm{Var}\left[\sum_{i=1}^{rK/2}\xi_i\right] &\le \frac{1}{2}rK\mu(1-\mu) + \frac{1}{2}rK(rK/2-1)\left(\frac{1}{1-p}-1\right)\mu^2 \\ &\le \frac{1}{2}rK\mu + \frac{1}{4}r^2K^2\mu^2\frac{p}{1-p}.\end{aligned}$$

Under the condition (42) we have $p \le 1/8$ and

$$\mu = (1-p)^{K-1} \overset{(a)}{\ge} e^{-2(K-1)p} \ge (rK/2)^{-1/4}, \tag{45}$$

where (a) uses the inequality $1-x \geq e^{-2x}, \forall x \in [0, \frac{1}{2}]$. Applying the Chebyshev's inequality, we get

$$\mathbb{P}\left[\left|\sum_{i=1}^{rK/2} \xi_i - rK\mu/2\right| \geq rK\mu/2\right] \leq \frac{\frac{1}{2}rK\mu + \frac{1}{4}(rK\mu)^2 \frac{p}{1-p}}{r^2K^2\mu^2/4} \leq \frac{2}{rK\mu} + \frac{p}{1-p} \leq \frac{1}{4},$$

where the last inequality holds due to (45) and $p \leq 1/8$. It follows that $\rho_1 \geq \frac{3}{4}$. If we let ρ_2 denote the probability that there exits a disconnected node among the next $rK/2$ nodes $\{rK/2+1, \ldots, rK\}$, then by symmetry $\rho_2 \geq \frac{3}{4}$. Therefore $\rho \geq \rho_1\rho_2 \geq 1/2$, proving the sufficiency of (42).

We next prove the sufficiency of the condition (43) by bounding the error probability using a similar strategy. For $k = 1, 2$, we call a node in cluster k a *betrayed node* if it is connected to all nodes in cluster $3-k$. Let E' be the event of having a betrayed node in each of the clusters 1 and 2, and let $\rho' := \mathbb{P}[E']$. Suppose Y^* is uniformly distributed over $\mathcal{Y}$; we can use a similar argument as above to show that $\mathbb{P}[\widehat{Y} \neq Y^*] \geq \rho'/2$ for any $\widehat{Y}$. Fix Y^* to be such that the clusters 1 and 2 are given by the nodes $[K]$ and $\{K+1, \ldots, 2K\}$, respectively. For each $i \in [K]$, let ξ_i' be the indicator for node i being is a betrayed node. Then $\rho_1' := \mathbb{P}\left[\sum_{i=1}^K \xi_i' > 0\right]$ is the probability of having a betrayed node in cluster 1. Note that the condition (43) implies $1-q \leq 1/2$. We have

$$\mathbb{P}\left[\sum_{i=1}^K \xi_i' = 0\right] = \left(1-q^K\right)^K \leq \exp\left(-Kq^K\right) \overset{(b)}{\leq} \exp\left[-K\exp\left(-2(1-q)K\right)\right] \overset{(c)}{\leq} \exp(-K^{1/2}) \leq 1/4,$$

where (b) follows from the inequality $q^K = (1-(1-q))^K \geq \exp(-2(1-q)K)$ for $1-q \leq 1/2$, and (c) holds under the condition (43). We therefore obtain $\rho_1' \geq \frac{3}{4}$. Let ρ_2' be the probability of having a betrayed node in cluster 2; by symmetry $\rho_2' \geq 3/4$. By the union bound we get $\rho' \geq 1-(1-\rho_1')-(1-\rho_2') \geq 1/2$, proving the sufficiency of (43). ■

We can now prove Theorem 2 by combining the above three lemmas.

Proof [of Theorem 2] Since $256 \leq 2K \leq n$, we have the following relations for the logarithmic terms:

$$\log(n-K) \geq \log(n/2) \geq \frac{1}{2}\log n, \quad \text{and} \quad \log(rK/2) \geq \frac{1}{2}\log(rK). \tag{46}$$

Our goal is to show that if the condition (1) or (2) holds, then we can draw the conclusion that the minimax error probability is large.

First assume (1) holds. By (36), we know the condition (1) implies

$$K(p-q)^2 \leq \frac{1}{96}p(1-q)\left(\log(rK) \wedge K\right). \tag{47}$$

We distinguish between two cases. (i) If $p \leq 2q$, then (47) implies $K(p-q)^2 \leq \frac{1}{48}q(1-q)\log(rK)$; it follows from (35) and (46) that $KD(p\|q) \leq \frac{1}{48}\log(rK) \leq \frac{1}{24}\log(n-K)$, and thus Lemma 24 proves the conclusion. (ii) If $p > 2q$, then (47) implies $Kp \leq \frac{1}{24}\log(rK) \wedge K \leq \min\{\frac{1}{24}K, \frac{1}{12}\log(\frac{rK}{2})\}$, and Lemma 25 proves the conclusion.

Next assume the condition (2) holds. Using the lower-bound (36) on the KL divergence, we know that (2) implies

$$K(p-q)^2 \leq \frac{1}{96}p(1-q)\log n. \tag{48}$$

We distinguish between two cases. (i) If $1-q \leq 2(1-p)$, then (48) implies $K(p-q)^2 \leq \frac{1}{48}p(1-p)\log n$; it follows from (35) and (46) that $KD(q\|p) \leq \frac{1}{48}\log n \leq \frac{1}{24}\log(n-K)$, and thus Lemma 24 implies the conclusion. (ii) If $1-q > 2(1-p)$ and $K \geq \log n$, then (48) implies

$$K(1-q) \leq \frac{1}{24}\log n \leq \frac{1}{12}\max\left\{\log\frac{n}{K}, \log K\right\}. \tag{49}$$

We further divide the analysis into two sub-cases.

Case $(ii.1)$: $K \geq \log n$. It follows from (49) that $1-q \leq \frac{1}{24}$, i.e., $q \geq \frac{23}{24}$, and thus $(p-q)^2 \leq 2q(1-q)^2$. Therefore, the inequality (49) implies either the condition (38) in Lemma 23 or the condition (43) in Lemma 25, which proves the conclusion.

Case $(ii.2)$: $K < \log n$. It follows that $\delta := \frac{n_1(K-1)}{n(n-1)} \leq \frac{1}{10}$ and $\log\frac{n}{K} \geq \frac{1}{2}\log n$. Note that $\bar{p} := \delta p + (1-\delta)q \geq \max\{\delta p, q\}$ and $1-\bar{p} \geq \frac{9}{10}(1-q)$. Therefore, we have

$$\frac{n^2(q-\bar{p})^2}{n_1\bar{p}(1-\bar{p})} = \frac{n^2\delta^2(p-q)^2}{n_1\bar{p}(1-\bar{p})} \leq \frac{2n^2\delta(p-q)^2}{n_1p(1-q)} \overset{(a)}{\leq} 4KD(p\|q) \overset{(b)}{\leq} \frac{1}{24}\log\frac{n}{K}, \tag{50}$$

where we use (36) in (a) and (2) in (b). On the other hand, we have

$$\begin{aligned} D(p\|\bar{p}) &= p\log\frac{p}{\bar{p}} + (1-p)\log\frac{1-p}{1-\bar{p}} \leq p\log\frac{p}{q} + (1-p)\log\frac{10(1-p)}{9(1-q)} \\ &\leq D(p\|q) + (1-q)\log\frac{10}{9} \leq \frac{1}{6K}\log\frac{n}{K}, \end{aligned} \tag{51}$$

where the last inequality follows from (2) and (49). Equations (50) and (51) imply the assumption (37) in Lemma 23, and therefore the conclusion follows. ■

5.1.1 PROOF OF COROLLARY 3

The corollary is derived from Theorem 2 using the upper bound (35) on the KL divergence. In particular, condition (3) implies condition (2) in Theorem 2 in view of (35). Similarly, condition (4) implies condition (1) because $D(q\|p) \leq \frac{p}{1-p}$ in view of (35) and $p \leq \frac{1}{193}$. Finally, condition (5) implies condition (2) because $D(p\|q) \leq p\log\frac{p}{q}$ by definition of the KL divergence.

5.2 Proof of Theorem 4 and Corollary 5

Let $\langle X, Y\rangle := \mathrm{Tr}(X^\top Y)$ denote the trace inner product between two matrices. For each feasible solution $Y \in \mathcal{Y}$ of the optimization problem (7), we define $\Delta(Y) := \langle A, Y^* - Y\rangle$ and $d(Y) := \langle Y^*, Y^* - Y\rangle$. To prove the theorem, it suffices to show that $\Delta(Y) > 0$ for all feasible Y with $Y \neq Y^*$. For simplicity, in this proof we use a different convention that $Y_{ii}^* = 0$ and $Y_{ii} = 0$ for all

$i \in V$. Note that $\mathbb{E}[A] = qJ + (p-q)Y^* - qI$, where J is the $n \times n$ all-one matrix and I is the $n \times n$ identity matrix. We may decompose $\Delta(Y)$ into an expectation term and a fluctuation term:

$$\Delta(Y) = \langle \mathbb{E}[A], Y^* - Y \rangle + \langle A - \mathbb{E}[A], Y^* - Y \rangle = (p-q)d(Y) + \langle A - \mathbb{E}[A], Y^* - Y \rangle, \quad (52)$$

where the second equality follows from $\sum_{i,j} Y_{ij} = \sum_{i,j} Y^*_{ij}$ by feasibility of Y. For the second fluctuation term above, observe that

$$\langle A - \mathbb{E}[A], Y^* - Y \rangle = 2 \underbrace{\sum_{(i<j): \substack{Y^*_{ij}=1 \\ Y_{ij}=0}} (A_{ij} - p)}_{T_1(Y)} - 2 \underbrace{\sum_{(i<j): \substack{Y^*_{ij}=0 \\ Y_{ij}=1}} (A_{ij} - q)}_{T_2(Y)}.$$

Here each of $T_1(Y)$ and $T_2(Y)$ is the sum of $\frac{1}{2}d(Y)$ i.i.d. centered Bernoulli random variables with parameter p and q, respectively.

Using the Chernoff bound, we can bound the fluctuation for each fixed $Y \in \mathcal{Y}$:

$$\mathbb{P}\left\{T_1(Y) \le -\frac{p-q}{4}d(Y)\right\} \le \exp\left[-\frac{1}{2}d(Y)D\left(\frac{p+q}{2}\middle\|p\right)\right], \quad (53)$$

$$\mathbb{P}\left\{T_2(Y) \ge \frac{p-q}{4}d(Y)\right\} \le \exp\left[-\frac{1}{2}d(Y)D\left(\frac{p+q}{2}\middle\|q\right)\right]. \quad (54)$$

We need to control the fluctuation uniformly over $Y \in \mathcal{Y}$. Define the equivalence class $[Y] := \{Y' \in \mathcal{Y} : Y'_{ij} = Y_{ij}, \forall (i,j) \in \text{support}(Y^*)\}$, where $\text{support}(Y^*) := \{(i,j) : Y^*_{ij} = 1\}$. Obverse that all cluster matrices in the equivalence class $[Y]$ have the same value of $T_1(Y)$. The following combinatorial lemma upper bounds the number of Y's and $[Y]$'s such that $d(Y) = t$. Note that $2(K-1) \le d(Y) \le rK^2$ for all feasible $Y \ne Y^*$.

Lemma 26 *For each integer $t \in [K, rK^2]$, we have*

$$\left|\{Y \in \mathcal{Y} : d(Y) = t\}\right| \le \left(\frac{16t^2}{K^2}\right)^2 n^{32t/K},$$

$$\left|\{[Y] : d(Y) = t\}\right| \le \frac{16t^2}{K^2}(rK)^{16t/K}.$$

We prove this lemma in Appendix A. We also need the following lemma, which lower-bounds $D\left(\frac{p+q}{2}\|q\right)$ and $D\left(\frac{p+q}{2}\|p\right)$ using $D(p\|q)$ and $D(q\|p)$, respectively. The proof is given in Appendix B.

Lemma 27 *For any $0 \le p \le q \le 1$, we have*

$$D\left(\frac{p+q}{2}\middle\|q\right) \ge \frac{1}{36}D(p\|q), \quad (55)$$

$$D\left(\frac{p+q}{2}\middle\|p\right) \ge \frac{1}{36}D(q\|p). \quad (56)$$

Using the union bound, (53), Lemma 26 and Lemma 27, we obtain

$$\begin{aligned}
&\mathbb{P}\left\{\exists[Y] : Y \ne Y^*, T_1(Y) \le -\frac{p-q}{4}d(Y)\right\} \\
&\le \sum_{t=K}^{rK^2} \mathbb{P}\left\{\exists[Y] : d(Y) = t, T_1(Y) \le -\frac{p-q}{4}t\right\} \\
&\le \sum_{t=K}^{rK^2} \left|\{\exists[Y] : d(Y) = t\}\right| \cdot \mathbb{P}\left\{T_1(Y) \le -\frac{p-q}{4}t\right\} \\
&\le \sum_{t=K}^{rK^2} \frac{16t^2}{K^2}(rK)^{16t/K} \exp\left(-\frac{1}{72}tD(q\|p)\right) \\
&\overset{(a)}{\le} 16\sum_{t=K}^{rK^2}(rK)^2(\gamma rK)^{-5t/K} \le 16(\gamma rK)^{-1},
\end{aligned}$$

where (a) follows from the theorem assumption that $D(q\|p) \ge c_1 \log(\gamma rK)/K$ for a sufficiently large constant c_1. Similarly, using (54) we have

$$\begin{aligned}
&\mathbb{P}\left\{\exists Y \in \mathcal{Y} : Y \ne Y^*, T_2(Y) \ge \frac{p-q}{4}d(Y)\right\} \\
&\le \sum_{t=K}^{rK^2} \mathbb{P}\left\{\exists Y \in \mathcal{Y} : d(Y) = t, T_2(Y) \ge \frac{p-q}{4}t\right\} \\
&\le \sum_{t=K}^{rK^2} \left|\{Y \in \mathcal{Y} : d(Y) = t\}\right| \cdot \mathbb{P}\left\{T_2(Y) \ge \frac{p-q}{4}t\right\} \\
&\le \sum_{t=K}^{rK^2} \frac{256t^4}{K^4} n^{32t/K} \cdot \exp\left(-\frac{1}{72}tD(p\|q)\right) \overset{(b)}{\le} 256n^{-1},
\end{aligned}$$

where (b) follows from the theorem assumption that $D(p\|q) \ge c_1 \log n/K$ for a sufficiently large constant c_1. Combining the above two bounds with (52), we obtain

$$\mathbb{P}\{\exists Y \in \mathcal{Y} : \Delta(Y) \le 0\} \le 16(\gamma rK)^{-1} + 256n^{-1}.$$

Therefore Y^* is the unique optimal solution with the same probability. This proves the theorem.

5.2.1 PROOF OF COROLLARY 5

The corollary is derived from Theorem 4 using the lower bound (36) on the KL divergence. First assume $e^2q \ge p$. Then $K(p-q)^2 \gtrsim q(1-q)\log n$ implies condition (9) in view of (36). Next assume $e^2q < p$. It follows that $\log\frac{p}{q} \le 2\log\frac{p}{eq}$. By definition, $D(p\|q) \ge p\log\frac{p}{q} + (1-p)\log(1-p) \ge p\log\frac{p}{eq}$. Hence, $Kp\log\frac{p}{q} \gtrsim \log n$ implies $KD(p\|q) \gtrsim \log n$. Furthermore, $D(q\|p) \ge \frac{1}{2}(1-1/e^2)p$ in view of (36) and $p > e^2q$. Therefore, $Kp \gtrsim \log(rK)$ implies $KD(q\|p) \gtrsim \log(rK)$.

5.3 Proof of Theorem 6

We prove Theorem 6 and Theorem 17 (for submatrix localization) together in this section. Our proof relies only on two standard concentration results for the adjacency matrix A (Proposition 28 below).

We need some unified notation for the two models. For both models we use n_L and n_R to denote the problem dimensions, with the understanding that $n_L = n_R = n$ for planted clustering. Similarly, for planted clustering the left and right clusters are identical and $K_L = K_R = K$. Let $U \in \mathbb{R}^{n_L \times r}$ and $V \in \mathbb{R}^{n_R \times r}$ be the normalized characteristic matrices of the left and right clusters, respectively:

$$U_{ik} = \begin{cases} \frac{1}{\sqrt{K_L}}, & \text{if the left node } i \text{ is in the } k\text{-th left cluster,} \\ 0, & \text{otherwise,} \end{cases}$$

and similarly for V. Here $U = V$ for planted clustering. The true cluster matrix Y^* has the rank-r Singular Value Decomposition given by $Y^* = \sqrt{K_L K_R} U V^\top$. Define the projections $\mathcal{P}_T(M) = UU^\top M + MVV^\top - UU^\top MVV^\top$ and $\mathcal{P}_{T^\perp}(M) = M - \mathcal{P}_T(M)$. Several matrix norms will be used: the spectral norm $\|X\|$ (the largest singular value of X), the nuclear norm $\|X\|_*$ (the sum of the singular values), the ℓ_1 norm $\|X\|_1 = \sum_{i,j} |X_{ij}|$ and the ℓ_∞ norm $\|X\|_\infty = \max_{i,j} |X_{ij}|$.

We define a quantity $\nu > 0$ and a matrix $\bar{A} \in \mathbb{R}^{n_L \times n_R}$, which roughly correspond to the signal strength and the mean of A. For planted clustering, let $\nu := p - q$ and $\bar{A} := qJ + (p-q)Y^*$, where J is the all-one matrix. For submatrix localization, let $\nu := \mu$ and $\bar{A} := \mu Y^*$. The proof hinges on the following probabilistic property of the random matrix $A - \bar{A}$.

Proposition 28 *Under the condition (14) for planted clustering,* or *the condition (27) for submatrix localization, the following holds with probability at least* $1 - n^{-10}$*:*

$$\|A - \bar{A}\| \le \frac{1}{8}\nu\sqrt{K_L K_R}, \tag{57}$$

$$\|\mathcal{P}_T(A - \bar{A})\|_\infty \le \frac{1}{8}\nu. \tag{58}$$

We prove the proposition in Section 5.3.1 to follow. In the rest of the proof we assume the event that (57) and (58) hold. To establish the theorems, it suffices to show that $\langle Y^* - Y, A\rangle > 0$ for all feasible solution Y of the convex program with $Y \neq Y^*$. For any feasible Y, we may write

$$\begin{aligned}\langle Y^* - Y, A\rangle &= \langle \bar{A}, Y^* - Y\rangle + \langle A - \bar{A}, Y^* - Y\rangle \\ &= \nu\langle Y^*, Y^* - Y\rangle + \langle A - \bar{A}, Y^* - Y\rangle = \frac{\nu}{2}\|Y^* - Y\|_1 + \langle A - \bar{A}, Y^* - Y\rangle, \end{aligned} \tag{59}$$

where the second equality follows from the definition of $\bar{A}$, and the third equality holds because Y obeys the linear constraints $\sum_{i,j} Y_{ij} = \sum_{i,j} Y^*_{ij}$ and $Y_{ij} \in [0,1], \forall i,j$.

On the other hand, we have $\|Y^*\|_* \ge \|Y\|_*$ thanks to the constraint (12) or (25). Let $W := \frac{8(A-\bar{A})}{\nu\sqrt{K_L K_R}}$. By (57) we have $\|\mathcal{P}_{T^\perp}(W)\| \le \|W\| \le 1$, so $UV^\top + \mathcal{P}_{T^\perp}(W)$ is a subgradient of $f(X) := \|X\|_*$ at $X = Y^*$ (cf. Recht et al. 2010 for characterization of the subgradient of the nuclear norm). It follows that

$$0 \ge \|Y\|_* - \|Y^*\|_* \ge \left\langle UV^\top + \mathcal{P}_{T^\perp}(W), Y - Y^*\right\rangle = \langle W, Y - Y^*\rangle + \left\langle UV^\top - \mathcal{P}_T(W), Y - Y^*\right\rangle.$$

Rearranging terms and using the definition of W gives

$$\left\langle A - \bar{A}, Y^* - Y\right\rangle = \frac{\nu\sqrt{K_L K_R}}{8}\langle W, Y^* - Y\rangle \ge \frac{\nu\sqrt{K_L K_R}}{8}\left\langle -UV^\top + \mathcal{P}_T(W), Y^* - Y\right\rangle. \tag{60}$$

Assembling (59) and (60), we obtain that for any feasible Y,

$$\begin{aligned}\langle Y^* - Y, A\rangle &\ge \frac{\nu}{2}\|Y^* - Y\|_1 + \frac{\nu\sqrt{K_L K_R}}{8}\left\langle -UV^\top + \mathcal{P}_T(W), Y^* - Y\right\rangle \\ &\ge \left(\frac{\nu}{2} - \frac{\nu\sqrt{K_L K_R}}{8}\|UV^\top\|_\infty - \|\mathcal{P}_T(A - \bar{A})\|_\infty\right)\|Y^* - Y\|_1,\end{aligned}$$

where the last inequality follows from the duality between the ℓ_1 and ℓ_∞ norms. Using (58) and the fact that $\|UV^\top\|_\infty = 1/\sqrt{K_L K_R}$, we get

$$\langle Y^* - Y, A\rangle \ge \left(\frac{\nu}{2} - \frac{\nu}{8} - \frac{\nu}{8}\right)\|Y^* - Y\|_1 = \frac{\nu}{4}\|Y^* - Y\|_1,$$

where the R.H.S. is strictly positive for all $Y \neq Y^*$. This completes the proof of Theorems 6 and 17.

5.3.1 PROOF OF PROPOSITION 28

We first prove (58). By definition of $\mathcal{P}_T$, we have

$$\begin{aligned}\|\mathcal{P}_T(A - \bar{A})\|_\infty \le& \|UU^\top(A - \bar{A})\|_\infty + \|(A - \bar{A})VV^\top\|_\infty + \|UU^\top(A - \bar{A})VV^\top\|_\infty \\ \le& 3\max\left(\|UU^\top(A - \bar{A})\|_\infty, \|(A - \bar{A})VV^\top\|_\infty\right).\end{aligned} \tag{61}$$

Suppose the left node i belongs to the left cluster k. Then

$$(UU^\top(A - \bar{A}))_{ij} = \frac{1}{K_L}\sum_{l \in C_k^*}(A - \bar{A})_{lj} = \frac{1}{K_L}\sum_{l \in C_k^*}(A - \mathbb{E}A)_{lj} + \frac{1}{K_L}\sum_{l \in C_k^*}(\mathbb{E}A - \bar{A})_{lj}. \tag{62}$$

To proceed, we consider the two models separately.

Planted clustering: The entries of the matrix $A - \mathbb{E}A$ are centered Bernoulli random variables with variance bounded by $p(1-q)$ and mutually independent up to symmetry with respect to the diagonal. The first term of (62) is the average of K_L such random variables; by Bernstein's inequality (stated as Theorem 32 in Appendix C), we have with probability at least $1 - n^{-13}$ and for some universal constant c_2,

$$\left|\sum_{l \in C_k^*}(A - \mathbb{E}A)_{lj}\right| \le \sqrt{26p(1-q)K\log n} + 9\log n \le c_2\sqrt{p(1-q)K\log n},$$

where the last inequality follows because $Kp(1-q) > c_1 \log n$ in view of the condition (14). By definition of $\bar{A}$, $\mathbb{E}[A] - \bar{A}$ is a diagonal matrix with each diagonal entry equal to $-p$ or $-q$, so the second term of (62) has magnitude at most $1/K$. By the union bound over all (i,j) and substituting back to (61), we have with probability at least $1 - 2n^{-11}$,

$$\|\mathcal{P}_T(A - \bar{A})\|_\infty \le 3c_2\sqrt{p(1-q)\log n/K} + 3/K \le (p-q)/8 = \nu/8,$$

where the last inequality follows from the condition (14). This proves (58) in the proposition.

Submatrix localization: We have $\bar{A} = \mathbb{E}A$ by definition, so the second term of (62) is zero. The first term is the average of K_L independent centered random variables with unit sub-Gaussian norm. By a standard sub-Gaussian concentration inequality (e.g., Proposition 5.10 in Vershynin 2012), we have for some universal constant c_3 and with probability at least $1 - n^{-13}$,

$$\left|\sum_{l\in\mathcal{C}_k}(A - \mathbb{E}A)_{lj}\right| \le c_3\sqrt{K_L \log n}.$$

So $\|UU^\top(A-\mathbb{E}A)\|_\infty \le c_3\sqrt{\log n/K_L}$ with probability at least $1-n^{-11}$ by the union bound. Similarly, $\|(A-\mathbb{E}A)VV^\top\|_\infty \le c_2\sqrt{\log n/K_R}$ with the same probability. Combining with (61) gives

$$\|\mathcal{P}_T(A - \bar{A})\|_\infty \le \sqrt{\log n/\min\{K_L, K_R\}} \le \nu/8 = \mu/8,$$

where the last inequality holds under the condition (27). This proves (58) in the proposition.

We now turn to (57) in the proposition, and again consider the two models separately.

- *Planted clustering:* Note that $\|A - \bar{A}\| \le \|A - \mathbb{E}[A]\| + \|\bar{A} - \mathbb{E}[A]\| \le \|A - \mathbb{E}[A]\| + 1$. Under the condition (14), $Kp(1-q) \ge c_1 \log n$. We bound the spectral norm term in the lemma below.

  Lemma 29 *If $Kp(1-q) \ge c_1 \log n$, then there exists some universal constant c_4 such that $\|A - \mathbb{E}[A]\| \le c_4\sqrt{p(1-q)K\log n + q(1-q)n}$ with probability at least $1 - n^{-10}$.*

  We prove the lemma in Section 5.3.2 to follow. Applying the lemma, we obtain

  $$\|A - \bar{A}\| \le c_4\sqrt{p(1-q)K\log n + q(1-q)n} + 1 \le \frac{K(p-q)}{8} = \frac{K\nu}{8},$$

  where the second inequality holds under the condition (14).

- *Submatrix localization:* The matrix $A - \bar{A} = A - \mathbb{E}A$ has i.i.d. sub-Gaussian entries. Using a standard concentration bound for the spectral norm of such a matrix (e.g., Theorem 5.39 in Vershynin 2012), we get that for a universal constant c_5 and with probability at least $1 - n^{-10}$,

  $$\|A - \mathbb{E}A\| \le c_5\sqrt{n} \le \frac{\mu}{8}\sqrt{K_L K_R} = \frac{\nu}{8}\sqrt{K_L K_R},$$

  where the second inequality holds under the condition (27).

5.3.2 PROOF OF LEMMA 29

Let $R := \text{support}(Y^*)$ and $\mathcal{P}_R(\cdot) : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ be the operator that sets the entries outside R to zero. Set $B_1 := \mathcal{P}_R(A - \mathbb{E}[A])$ and $B_2 := A - \mathbb{E}[A] - B_1$. Then B_1 is a block-diagonal symmetric matrix with r blocks of size $K \times K$ and its upper-triangular entries are independent with zero mean and variance bounded by $p(1-q)$. Applying the matrix Bernstein inequality in Tropp (2012) and using the assumption that $Kp(1-q) \ge c_1 \log n$ in the lemma, we get that there exits some universal constant c_6 such that $\|B_1\| \le c_6\sqrt{p(1-q)K\log n}$ with probability at least $1 - n^{-11}$.

On the other hand, B_2 is symmetric and its upper-triangular entries are independent centered Bernoulli random variables with variance bounded by $\sigma^2 := \max\{q(1-q), c_7 \log n/n\}$ for any

universal constant c_7. If $\sigma^2 \ge \frac{\log^7 n}{n}$, then Theorem 8.4 in Chatterjee (2014) implies that $\|B_2\| \le 3\sigma\sqrt{n}$ with probability at least $1-n^{-11}$. If $c_7\frac{\log n}{n} \le \sigma^2 \le \frac{\log^7 n}{n}$ for a sufficiently large constant c_7, then Lemma 2 in Massoulié and Tomozei (2014) implies that $\|B_2\| \le c_8\sigma\sqrt{n}$ with probability at least $1 - n^{-11}$ for some universal constant c_8. (See Lemma 8 in Vu 2014 for a similar derivation.) Putting together, we conclude that with probability at least $1 - 2n^{-11}$,

$$\begin{aligned}\|A - \mathbb{E}[A]\| \le \|B_1\| + \|B_2\| &\le c_6\sqrt{p(1-q)K\log n} + c_8\max\{\sqrt{q(1-q)n}, \sqrt{\log n}\}\\ &\le c_4\sqrt{p(1-q)K\log n + q(1-q)n},\end{aligned}$$

where the last inequality holds because $Kp(1-q) \ge c_1 \log n$ by assumption. This proves the lemma.

5.4 Proof of Theorem 8

We first claim that $K(p-q) \le c_2\sqrt{Kp+qn}$ implies $K(p-q) \le c_2\sqrt{2qn}$ under the assumption that $K \le n/2$ and $qn \ge c_1 \log n$. In fact, if $Kp \le qn$, then the claim trivially holds. If $Kp > qn$, then $q < Kp/n \le p/2$. It follows that

$$Kp/2 < K(p-q) \le c_2\sqrt{Kp+qn} \le c_2\sqrt{2Kp}.$$

Therefore, we have $Kp < 8c_2^2$, which contradicts the assumption that $Kp > qn \ge c_1 \log n$. So $Kp > qn$ cannot hold. Consequently, it suffices to show that if $K(p-q) \le c_2\sqrt{2qn}$, then Y^* is not an optimal solution. We do this by deriving a contradiction assuming the optimality of Y^*.

Let J be the $n \times n$ all-ones matrix. Let $\mathcal{R} := \text{support}(Y^*)$ and $\mathcal{A} := \text{support}(A)$. Recall the cluster characteristic matrix U and the projection $\mathcal{P}_T(M) = UU^\top M + MUU^\top - UU^\top MUU^\top$ defined in Section 5.3, and that $Y^* = KUU^\top$ is the SVD of Y^*. Consider the Lagrangian

$$L(Y;\lambda,\mu,F,G) := -\langle A, Y\rangle + \lambda\left(\|Y\|_* - \|Y^*\|_*\right) + \eta\left(\langle J, Y\rangle - rK^2\right) - \langle F, Y\rangle + \langle G, Y - J\rangle,$$

where $\lambda, \eta \in \mathbb{R}$ and $F, G \in \mathbb{R}^{n\times n}$ are the Lagrangian multipliers. Since $Y = \frac{rK^2}{n^2}J$ is strictly feasible, strong duality holds by Slater's condition. Therefore, if Y^* is an optimal solution, then there must exist some $F, G \in \mathbb{R}^{n\times n}$ and λ for which the KKT conditions hold:

$$\left.\begin{aligned} 0 \in \frac{\partial L(Y;\lambda,\mu,F,G)}{\partial Y}\Big|_{Y=Y^*},\end{aligned}\right\}\text{Stationary condition}$$

$$\left.\begin{aligned} F_{ij} \ge 0, G_{ij} \ge 0, \forall(i,j),\\ \lambda \ge 0,\end{aligned}\right\}\text{Dual feasibility}$$

$$\left.\begin{aligned} F_{ij} = 0, \forall(i,j) \in \mathcal{R},\\ G_{ij} = 0, \forall(i,j) \in \mathcal{R}^c.\end{aligned}\right\}\text{Complementary slackness}$$

Recall that $M \in \mathbb{R}^{n\times n}$ is a sub-gradient of $\|X\|_*$ at $X = Y^*$ if and only if $\mathcal{P}_T(M) = UU^\top$ and $\|M - \mathcal{P}_T(M)\| \le 1$. Set $H = F - G$; then the KKT conditions imply that there exist some numbers $\lambda \ge 0, \eta \in \mathbb{R}$ and matrices W, H obeying

$$A - \lambda\left(UU^\top + W\right) - \eta J + H = 0; \tag{63}$$

$$\mathcal{P}_T W = 0; \qquad \|W\| \le 1; \tag{64}$$

$$H_{ij} \le 0, \ \forall(i,j) \in \mathcal{R}; \qquad H_{ij} \ge 0, \ \forall(i,j) \in \mathcal{R}^c. \tag{65}$$

Now observe that $UU^\top WUU^\top = 0$ by (64). We left and right multiply (63) by $UU^\top$ to obtain

$$\breve{A} - \lambda UU^\top - \eta J + \breve{H} = 0,$$

where for any $X \in \mathbb{R}^{n\times n}$, $\breve{X} := UU^\top XUU^\top$ is the matrix obtained by averaging each $K \times K$ block of X. Consider the last display equation on the entries in $\mathcal{R}$ and $\mathcal{R}^c$ respectively. Applying the Bernstein inequality (Theorem 32) on each entry $\breve{A}_{ij}$, we have with probability at least $1 - 2n^{-11}$,

$$p - \frac{\lambda}{K} - \eta + \breve{H}_{ij} \geq -\frac{c_3\sqrt{p(1-p)\log n}}{K} - \frac{c_4 \log n}{2K^2} \overset{(a)}{\geq} -\frac{\epsilon_0}{8}, \quad \forall (i,j) \in \mathcal{R}, \tag{66}$$

$$q - \eta + \breve{H}_{ij} \leq \frac{c_3\sqrt{q(1-q)\log n}}{K} + \frac{c_4 \log n}{2K^2} \overset{(b)}{\leq} \frac{\epsilon_0}{8}, \quad \forall (i,j) \in \mathcal{R}^c \tag{67}$$

for some universal constants $c_3, c_4 > 0$, where (a) and (b) follow from the assumption $K \geq c_1 \log n$ with a sufficiently large universal constant c_1. In the rest of the proof, we assume (66) and (67) hold. Using (65), we get that

$$\begin{aligned} \eta &\geq q - \frac{c_3\sqrt{q(1-q)\log n}}{K} - \frac{c_4 \log n}{2K^2} \geq q - \frac{\epsilon_0}{8}, \\ \eta &\leq p + \frac{c_3\sqrt{p(1-p)\log n}}{K} + \frac{c_4 \log n}{2K^2} - \frac{\lambda}{K} \leq p + \frac{\epsilon_0}{8} - \frac{\lambda}{K}. \end{aligned} \tag{68}$$

It follows that

$$\begin{aligned} \lambda &\leq K(p-q) + c_3(\sqrt{p(1-p)\log n} + \sqrt{q(1-q)\log n}) + \frac{c_4 \log n}{K} \\ &\leq 4\max\left\{K(p-q), c_3\sqrt{p(1-p)\log n}, c_3\sqrt{q(1-q)\log n}, \frac{c_4}{c_1}\right\}. \end{aligned} \tag{69}$$

On the other hand, the conditions (64) and (63) imply

$$\begin{aligned} \lambda^2 &= \left\|\lambda(UU^\top + W)\right\|^2 \geq \frac{1}{n}\left\|\lambda(UU^\top + W)\right\|_F^2 \\ &= \frac{1}{n}\left\|A - \eta J + H\right\|_F^2 \geq \frac{1}{n}\left\|A_{\mathcal{R}^c} - \eta J_{\mathcal{R}^c} + H_{\mathcal{R}^c}\right\|_F^2 \geq \frac{1}{n}\sum_{(i,j)\in\mathcal{R}^c}(1-\eta)^2 A_{ij}, \end{aligned}$$

where $X_{\mathcal{R}^c}$ denotes that matrix obtained from X by setting the entries outside $\mathcal{R}^c$ to zero. Using (68), $\lambda \geq 0$ and the assumption $p \leq 1 - \epsilon_0$, we obtain $\eta \leq 1 - \frac{7}{8}\epsilon_0$, and therefore

$$\lambda^2 \geq \frac{49}{64n}\epsilon_0^2 \sum_{(i,j)\in\mathcal{R}^c} A_{ij}. \tag{70}$$

Note that $\sum_{(i,j)\in\mathcal{R}^c} A_{ij}$ equals twice the sum of $\binom{n}{2} - r\binom{K}{2}$ i.i.d. Bernoulli random variables with parameter q. By the Chernoff bound of Binomial distributions and the assumption that $qn \geq c_1 \log n$, we have with probability at least $1 - n^{-11}$, $\sum_{(i<j)\in\mathcal{R}^c} A_{ij} \geq c_5 qn^2$ for some universal constant c_5. It follows from (70) that $\lambda^2 \geq \frac{1}{2}\epsilon_0^2 c_5 qn$. Combining with (69) and the assumption that $qn \geq c_1 \log n$, we conclude that with probability at least $1 - 3n^{-11}$, $K^2(p-q)^2 \geq \frac{1}{32}\epsilon^2 c_5 qn$. Choosing c_2 in the theorem assumption to be sufficiently small such that $2c_2^2 < \frac{1}{32}\epsilon^2 c_5$, we obtain $K(p-q) > c_2\sqrt{2qn}$, which leads to a contradiction. This completes the proof of the theorem.

5.5 Proof of Theorem 10

Define event

$$\mathcal{E}_1 = \left\{\min_{i\in V_1} d_i > \frac{(p-q)K}{2} + qn\right\} \cap \left\{\max_{i\in V_2} d_i < \frac{(p-q)K}{2} + qn\right\}.$$

Define $\mathcal{E}_2$ to be the event that $S_{ij} > \frac{(p-q)^2K}{3} + 2Kpq + q^2n$ for all nodes i, j from the same true cluster and $S_{ij} < \frac{(p-q)^2K}{3} + 2Kpq + q^2n$ for all pairs of nodes (i,j) from two different true clusters. On event $\mathcal{E}_1$, all nodes in V_1 are correctly declared to be non-isolated, and all nodes in V_2 are correctly declared to be isolated. One event $\mathcal{E}_1 \cap \mathcal{E}_2$, the counting algorithm correctly identifies all the true clusters. Hence, to prove the theorem, it suffices to show $\mathbb{P}\{\mathcal{E}_1^c\} \leq 2n^{-1}$ and $\mathbb{P}\{\mathcal{E}_2^c\} \leq 2n^{-1}$.

Let $\mathrm{Bin}(n,\alpha)$ denote the binomial distribution with n trials and success probability α. For each non-isolated node $i \in V_1$, its degree d_i is the sum of two independent binomial random variables distributed respectively as $\mathrm{Bin}(K-1,p)$ and $\mathrm{Bin}(n-K,q)$. For each isolated node $i \in V_2$, its degree d_i is distributed as $\mathrm{Bin}(n-1,q)$. It follows that $\mathbb{E}[d_i] = (n-1)q + (K-1)(p-q)$ if $i \in V_1$ and $\mathbb{E}[d_i] = (n-1)q$ if $i \in V_2$. Define $\sigma_1^2 := Kp(1-q) + nq(1-q)$, then $\mathrm{Var}[d_i] \leq \sigma_1^2$ for all i. Set $t_1 := \frac{1}{2}(K-1)(p-q)$. Since $p - q \leq p(1-q)$, it follows that $t_1 \leq \sigma_1^2$. Hence, Bernstein inequality (Theorem 32) gives

$$\mathbb{P}\{|d_i - \mathbb{E}[d_i]| \geq t_1\} \leq 2\exp\left(-\frac{t_1^2}{2\sigma_1^2 + 2t_1/3}\right) \leq 2\exp\left(-\frac{(K-1)^2(p-q)^2}{12\sigma_1^2}\right) \leq 2n^{-2},$$

where the last inequality follows from the assumption (17). By the union bound, we get that $\mathbb{P}\{\mathcal{E}_1^c\} \leq 2n^{-1}$.

For two nodes i and j in the same true cluster, the number of their common neighbors S_{ij} is the sum of two independent binomial random variables distributed respectively as $\mathrm{Bin}(K-2,p^2)$ and $\mathrm{Bin}(n-K,q^2)$. Similarly, for two nodes i, j in two different true clusters, S_{ij} is the sum of two independent binomial variables $\mathrm{Bin}(2(K-1),pq)$ and $\mathrm{Bin}(n-2K,q^2)$. Hence, $\mathbb{E}[S_{ij}]$ equals $(K-2)p^2 + (n-K)q^2$ if i and j are in the same true cluster and $2(K-1)pq + (n-2K)q^2$ if they are in two different true clusters. The difference of the expectations in two cases equals $K(p-q)^2 - 2p(p-q)$. Let $\sigma_2^2 := 2Kp^2(1-q^2) + nq^2(1-q^2)$, then $\mathrm{Var}[S_{ij}] \leq \sigma_2^2$. Set $t_2 := K(p-q)^2/3$. Since $p - q \leq p(1-q)$, it follows that $t_2 \leq \sigma_2^2$. Applying the Bernstein inequality (Theorem 32), we obtain that

$$\mathbb{P}\{|S_{ij} - \mathbb{E}[S_{ij}]| \geq t_2\} \leq 2\exp\left(-\frac{t_2^2}{2\sigma_2^2 + 2t_2/3}\right) \leq 2\exp\left(-\frac{K^2(p-q)^4}{24\sigma_2^2}\right) \leq 2n^{-3},$$

where the last inequality follows from the assumption (18). By the union bound, we get that $\mathbb{P}\{\mathcal{E}_2^c\} \leq 2n^{-1}$.

5.6 Proof of Theorem 12

For simplicity we assume K and n_2 are even numbers; the case where K or n_2 is odd can be proved similarly. We partition the non-isolated nodes V_1 into two equal-sized subsets V_{1+} and V_{1-} such that half of the nodes in each cluster are in V_{1+}. Similarly, the isolated nodes V_2 are partitioned into two equal-sized subsets V_{2+} and V_{2-}. The idea is to use the following large-deviation *lower* bound to the quantities d_i and S_{ij}.

Theorem 30 (Theorem 7.3.1 in Matoušek and Vondrák (2008)) *Let $X_1, \ldots, X_N$ be independent random variables such that $0 \le X_i \le 1$ for all i. Suppose $X = \sum_{i=1}^N X_i$ and $\sigma^2 := \sum_{i=1}^N Var[X_i] \ge 200$. Then for all $0 \le \tau \le \sigma^2/100$ and some universal constant $c_3 > 0$, we have*

$$\mathbb{P}\left[X \ge \mathbb{E}[X] + \tau\right] \ge c_3 e^{-\tau^2/(3\sigma^2)}.$$

The main hurdle is that the entries of the graph adjacency matrix A are not completely independent due to the symmetry of A, so we need to take into account the dependence among d_i's and the dependence among S_{ij}'s when applying Theorem 30.

We first argue that under the assumption (19), the simple counting algorithm fails to identify the isolated nodes with probability at least $1/4$. For each node i in $V_{1+} \cup V_{2+}$, let d_{i+} and d_{i-} be the numbers of its neighbors in $V_{1+} \cup V_{2+}$ and $V_{1-} \cup V_{2-}$, respectively, so its total degree is $d_i = d_{i+} + d_{i-}$. Let $\mathrm{Bin}(N, \alpha)$ denote the binomial distribution with N trials and success probability α. We consider the following two cases.

Case 1: $(Kp + (n-K)q)\log n_1 \ge nq \log n_2$. Recall that $n_1 = rK$ and $n_2 = n - n_1$. In this case, it follows from (19) that

$$(K-1)^2(p-q)^2 \le 2c_2(Kp + nq)\log n_1. \tag{71}$$

For each node $i \in V_{1+}$, the quantity d_{i-} is the sum of two independent Binomial random variables distributed as $\mathrm{Bin}(K/2, p)$ and $\mathrm{Bin}((n-K)/2, q)$, respectively. Define

$$\begin{aligned} t &:= (K-1)(p-q) + 2, \\ \gamma_d^- &:= \mathbb{E}[d_{i-}] - t = \frac{1}{2}nq + \frac{1}{2}K(p-q) - t, \\ \sigma_d^2 &:= \mathrm{Var}[d_{i-}] = \frac{1}{2}Kp(1-p) + \frac{1}{2}(n-K)q(1-q). \end{aligned}$$

By assumption of the theorem, we have $K \le n/2$, $q \le p \le 1 - c_0$ and thus $\sigma_d^2 \ge \frac{c_0}{4}(Kp + nq)$. Since $Kp^2 + nq^2 \ge c_1 \log n$ by assumption, it follows that $\sigma_d^2 \ge \frac{1}{4}c_0 c_1 \log n \ge 400$ by choosing the constant c_1 in the assumption sufficiently large. Furthermore, it follows from (71) that by choosing c_1 sufficiently large and c_2 sufficiently small, we have

$$\sigma_d^4 \ge \frac{1}{4}c_0(Kp+nq)\sigma_d^2 \ge \frac{c_0}{8c_2}\frac{(K-1)^2(p-q)^2}{\log n_1} \times \frac{1}{4}c_0 c_1 \log n \ge 200^2(K-1)^2(p-q)^2.$$

Moreover, we have shown that $\sigma_d^2 \ge 400$. Hence, we get that $\sigma_d^2 \ge 100t$. We can now apply Theorem 30 with (71) to get that

$$\mathbb{P}\left[d_{i-} \le \gamma_d^-\right] \ge c_3 \exp\left(-\frac{t^2}{3\sigma_d^2}\right) \ge c_3 n_1^{-c_4}$$

for some universal constant $c_4 > 0$ that can be made arbitrarily small by choosing c_2 in the assumption sufficiently small; the last inequality holds due to $\sigma_d^2 \ge \frac{c_0}{4}(Kp+nq)$, $\sigma_d^2 \ge 400$, and (71). Let $i^* := \arg\min_{i \in V_{1+}} d_{i-}$. Since the random variables $\{d_{i-} : i \in V_{1+}\}$ are mutually independent, we have

$$\mathbb{P}\left[d_{i^*-} > \gamma_d^-\right] = \prod_{i \in V_{1+}} \mathbb{P}\left[d_{i-} > \gamma_d^-\right] \le (1 - c_3 n_1^{-c_4})^{n_1/2} \le \exp\left(-c_3 n_1^{1-c_4}/2\right) \le 1/4,$$

where the last equality follows from letting c_4 sufficiently small and n_1 sufficiently large. Furthermore, for each $i \in V_{1+}$, the quantity d_{i+} is the sum of two independent Binomial random variables distributed as $\mathrm{Bin}(K/2-1, p)$ and $\mathrm{Bin}((n-K)/2, q)$, respectively. Since the median of $\mathrm{Bin}(N, \alpha)$ is at most $N\alpha + 1$, we know that with probability at least $1/2$, we have $d_{i+} \le \gamma_d^+ := nq/2 + K(p-q)/2 - p + 2$. Now observe that the two sets of random variables $\{d_{i+}, i \in V_{1+}\}$ and $\{d_{i-}, i \in V_{1+}\}$ are independent of each other, so d_{i+} is independent of i^* for each $i \in V_{1+}$. It follows that

$$\mathbb{P}\left[d_{i^*+} \le \gamma_d^+\right] = \sum_{i \in V_{1+}} \mathbb{P}\left[d_{i+} \le \gamma_d^+ | i^* = i\right]\mathbb{P}\left[i^* = i\right] = \sum_{i \in V_{1+}} \mathbb{P}\left[d_{i+} \le \gamma_d^+\right]\mathbb{P}\left[i^* = i\right] \ge \frac{1}{2}.$$

Combining the two display equations above with the union bound, we obtain that with probability at least $1/4$,

$$d_{i^*} = d_{i^*-} + d_{i^*+} \le \gamma_d^- + \gamma_d^+ = (n-1)q.$$

On this event the node i^* will be incorrectly declared as an isolated node.

Case 2: $(Kp+nq)\log n_1 \le nq \log n_2$. In this case we have $(K-1)^2(p-q)^2 \le 2c_2 nq \log n_2$ in view of (19). Set $i^* := \arg\max_{i \in V_{2+}} d_{i-}$. Following the same argument as in Case 1 and using the assumption $nq \ge c_1 \log n$, we can show that $d_{i^*} \ge nq + K(p-q)$ with probability at least $1/4$, and on this event node i^* will incorrectly be declared as a non-isolated node.

We next show that under the assumption (20), the simple algorithm fails to recover the clusters with probability at least $1/4$. For two nodes i and j in V_1, let S_{ij+} be the number of their common neighbors in $V_{1+} \cup V_{2+}$ and S_{ij-} the number of their common neighbors in $V_{1-} \cup V_{2-}$, so the total number of their common neighbors is $S_{ij} = S_{ij+} + S_{ij-}$.

For each pair of nodes (i, j) in V_{1+} from the same cluster, S_{ij-} is the sum of two independent Binomial random variables distributed as $\mathrm{Bin}(K/2, p^2)$ and $\mathrm{Bin}((n-K)/2, q^2)$, respectively. Define

$$\begin{aligned} t' &:= K(p-q)^2 + 4, \\ \gamma_S^- &:= \mathbb{E}[S_{ij-}] - t' = nq^2/2 + K(p^2-q^2)/2 - t', \\ \sigma_S^2 &:= \mathrm{Var}[S_{ij-}] = \frac{1}{2}Kp^2(1-p^2) + \frac{1}{2}(n-K)q^2(1-q^2). \end{aligned}$$

By assumption, we have $K \le n/2$, $q \le p \le 1 - c_0$ and hence $\sigma_S^2 \ge \frac{c_0}{4}(Kp^2 + nq^2)$. Since by assumption $Kp^2 + nq^2 \ge c_1 \log n$, it follows that $\sigma_S^2 \ge 200$ by choosing c_1 sufficiently large. Moreover, recall that condition (20) reads:

$$K^2(p-q)^4 < c_2(Kp^2 + nq^2)\log n_1.$$

Hence, by choosing the constant c_2 sufficiently small and c_1 sufficiently large, we have that $\sigma_S^2 \ge 100t'$. Theorem 30 with (20) then implies that

$$\mathbb{P}\left[S_{ij-} \le \gamma_S^-\right] \ge c_3 \exp\left(-\frac{(t')^2}{3\sigma_S^2}\right) \ge c_3 n_1^{-c_5},$$

where the universal constant $c_5 > 0$ can be made sufficiently small by choosing c_2 sufficiently small in (18); the last inequality holds due to $\sigma_S^2 \ge 200$, $\sigma_S^2 \ge \frac{c_0}{4}(Kp^2 + nq^2)$, and (20).

Without loss of generality, we may re-label the nodes such that $V_{1+} = \{1, 2, \ldots, n_1/2\}$ and for each $k = 1, \ldots, n_1/4$, the nodes $2k-1$ and $2k$ are in the same cluster. Note that the random variables $\{S_{(2k-1)(2k)-} : k = 1, 2, \ldots, n_1/4\}$ are mutually independent. Let $i^* := -1 + 2 \arg\min_{k=1,2,\ldots,n_1/4} S_{(2k-1)(2k)-}$ and $j^* := i^* + 1$. It follows that

$$\mathbb{P}\left[S_{i^*j^*-} \geq \gamma_S^-\right] \leq (1 - c_3 n_1^{-c_5})^{n_1/4} \leq \exp(-c_3 n_1^{1-c_5}/4) \leq 1/4.$$

Furthermore, notice that S_{ij+} is the sum of two independent Binomial random variables $\mathrm{Bin}(K/2-2, p^2)$ and $\mathrm{Bin}((n-K)/2, q^2)$. We use a median argument introduced in the first part of the proof to conclude that for all i, j, $S_{ij+} \leq \gamma_S^+ := nq^2/2 + K(p^2-q^2)/2 - 2p^2 + 2$ with probability at least $1/2$. Since $\{S_{ij+}, i, j \in V_{1+}\}$ only depends on the edges between V_{1+} and $V_{1+} \cup V_{2+}$, and (i^*, j^*) only depends on the edges between V_{1+} and $V_{1-} \cup V_{2-}$, it follows that $\{S_{ij+}, i, j \in V_{1+}\}$ and (i^*, j^*) are independent of each other. Therefore, $S_{i^*j^*+} \leq \gamma_S^+$ with probability at least $1/2$. Applying the union bound, we get that with probability at least $1/4$,

$$S_{i^*j^*} = S_{i^*j^*-} + S_{i^*j^*+} \leq \gamma_S^- + \gamma_S^+ = 2(K-1)pq + (n-2K)q^2.$$

On this event the nodes i^*, j^* will be incorrectly assigned to two different clusters.

6. Proofs for Submatrix Localization

In this section we prove the theoretical results in Section 3 for submatrix localization. Recall that $n := \max\{n_L, n_R\}$.

6.1 Proof of Theorem 15

We prove the theorem using Fano's inequality. Our arguments extend those used in Kolar et al. (2011). Recall that $\mathcal{Y}$ is the set of all valid bi-clustering matrices. Let $M = n_R - K_R$ and $\bar{\mathcal{Y}} = \{Y_0, Y_1, \ldots, Y_M\}$ be a subset of $\mathcal{Y}$ with cardinality $M+1$, which is specified later. Let $\mathbb{P}_{(Y^*,A)}$ denote the joint distribution of (Y^*, A) when Y^* is sampled from $\bar{\mathcal{Y}}$ uniformly at random and then A is generated according to the submatrix localization model with the true cluster matrix being Y^*. The minimax error probability can be bounded using the average error probability and Fano's inequality:

$$\inf_{\widehat{Y}} \sup_{Y^* \in \mathcal{Y}} \mathbb{P}\left[\widehat{Y} \neq Y^*\right] \geq \inf_{\widehat{Y}} \mathbb{P}_{(Y^*,A)}\left[\widehat{Y} \neq Y^*\right] \geq 1 - \frac{I(Y^*; A) + 1}{\log|\bar{\mathcal{Y}}|}, \tag{72}$$

where the mutual information is defined under the distribution $\mathbb{P}_{(Y^*,A)}$.

We construct $\bar{\mathcal{Y}}$ as follows. Let Y_0 be the bi-clustering matrix such that the left clusters $\{C_k\}_{k=1}^r$ are $C_k = \{(k-1)K_L + 1, \ldots, kK_L\}$ and the right clusters $\{D_l\}_{l=1}^r$ are $D_l = \{(l-1)K_R + 1, \ldots, lK_R\}$. Informally, each Y_i with $i \geq 1$ is obtained from Y_0 by keeping the left clusters and swapping two right nodes in two different right clusters. More specifically, for each $i \in [M]$: (1) Y_i has the same left clusters as Y_0; (2) if the right node $K_R + i \in D_l$ for Y_0, then Y_i has the same right clusters as Y_0 except that the first right cluster is $\{1, 2, \ldots, K_R - 1, K_R + i\}$ and the l-th right cluster is $D_l \setminus \{K_R + i\} \cup \{K_R\}$; (3) if the right node $K_R + i$ does not belong to any D_l for Y_0, then Y_i has the same right clusters as Y_0 except that the first right cluster is $\{1, 2, \ldots, K_R - 1, K_R + i\}$ instead.

Let $\mathbb{P}_i$ be the distribution of A conditioned on $Y^* = Y_i$, and $D(\mathbb{P}_i \| \mathbb{P}_{i'})$ the KL divergence between $\mathbb{P}_i$ and $\mathbb{P}_{i'}$. Since each $\mathbb{P}_i$ is a product of $n_L \times n_R$ Gaussian distributions, we have

$$\begin{aligned} I(Y^*; A) &\leq \frac{1}{(M+1)^2} \sum_{i,i'=0}^{M} D(\mathbb{P}_i \| \mathbb{P}_{i'}) \\ &\leq 3K_L \left[D\left(\mathcal{N}(\mu_1, \sigma^2) \| \mathcal{N}(\mu_2, \sigma^2)\right) + D\left(\mathcal{N}(\mu_2, \sigma^2) \| \mathcal{N}(\mu_1, \sigma^2)\right)\right] = 3K_L \frac{(\mu_1 - \mu_2)^2}{\sigma^2}, \end{aligned}$$

where we use the convexity of KL divergence in the first inequality, the definition of Y_i in the second inequality, and the expression for the KL divergence between two Gaussian distributions in the equality. If $(\mu_1 - \mu_2)^2 \leq \frac{\sigma^2 \log(n_R - K_R)}{12K_L}$, then $I(Y; A) \leq \frac{1}{2}\log(n_R - K_R) = \frac{1}{2}\log|\bar{\mathcal{Y}}|$. Since $\log(n_R - K_R) \geq \log(n_R/2) \geq 4$ if $n_R \geq 128$, it follows from (72) that the minimax error probability is at least $1/2$.

Alternatively, we can construct $Y_i, i \geq 1$ from Y_0 by keeping the right clusters and swapping two left nodes in two different left clusters. A similar argument shows that if $(\mu_1 - \mu_2)^2 \leq \frac{\sigma^2 \log(n_L - K_L)}{12K_R}$, the minimax error probability is at least $1/2$.

6.2 Proof of Theorem 16

Recall that $\langle X, Y\rangle := \mathrm{Tr}(X^\top Y)$ is the inner product between two matrices. For any feasible solution $Y \in \mathcal{Y}$ of (22), we define $\Delta(Y) := \langle A, Y^* - Y\rangle$ and $d(Y) := \langle Y^*, Y^* - Y\rangle$. To prove the theorem, it suffices to show that $\Delta(Y) > 0$ for all feasible Y with $Y \neq Y^*$. We may write

$$\Delta(Y) = \langle \mathbb{E}[A], Y^* - Y\rangle + \langle A - \mathbb{E}[A], Y^* - Y\rangle = \mu d(Y) + \langle A - \mathbb{E}[A], Y^* - Y\rangle \tag{73}$$

since $\mathbb{E}[A] = \mu Y^*$. The second term above can be written as

$$\langle A - \mathbb{E}[A], Y^* - Y\rangle = \underbrace{\sum_{(i,j): Y^*_{ij}=1, Y_{ij}=0} (A_{ij} - \mu)}_{T_1(Y)} + \underbrace{\sum_{(i,j): Y^*_{ij}=0, Y_{ij}=1} (-A_{ij})}_{T_2(Y)}.$$

Here each of $T_1(Y)$ and $T_2(Y)$ is the sum of $d(Y)$ i.i.d. centered sub-Gaussian random variables with parameter 1. By the sub-Gaussian concentration inequality given in Proposition 5.10 in Vershynin (2012), we obtained that for each $i = 1, 2$ and each fixed $Y \in \mathcal{Y}$,

$$\mathbb{P}\left\{T_i(Y) \leq -\frac{\mu}{2} d(Y)\right\} \leq e \exp\left(-C\mu^2 d(Y)\right),$$

where $C > 0$ is an absolute constant. Combining with the union bound and (73), we get

$$\mathbb{P}\left\{\Delta(Y) \leq 0\right\} \leq 2e \exp\left(-C\mu^2 d(Y)\right), \quad \text{for each } Y \in \mathcal{Y}. \tag{74}$$

Define the equivalence class $[Y] = \{Y' \in \mathcal{Y} : Y'_{ij} = Y_{ij}, \forall (i,j) \in \mathrm{support}(Y^*)$. The following combinatorial lemma (proved in Appendix A) upper-bounds the number of Y's and $[Y]$'s with a fixed value of $d(Y)$. Note that $K_L \wedge K_R \leq d(Y) \leq rK_LK_R$ for any feasible $Y \neq Y^*$.

Lemma 31 *For each integer $t \in [K_L \wedge K_R, rK_LK_R]$, we have*

$$\left|\{Y \in \mathcal{Y} : d(Y) = t\}\right| \le \left(\frac{16t^2}{K_LK_R}\right)^2 n_L^{16t/K_R} n_R^{16t/K_L}, \tag{75}$$

$$\left|\{[Y] : d(Y) = t\}\right| \le \frac{16t^2}{K_LK_R}(rK_L)^{8t/K_R}(rK_R)^{8t/K_L}. \tag{76}$$

Combining Lemma 31 with (74) and the union bound, we obtain

$$\begin{aligned}
&\mathbb{P}\left\{\exists Y \in \mathcal{Y} : Y \neq Y^*, \Delta(Y) \le 0\right\} \\
\le &\sum_{t=K_L\wedge K_R}^{rK_LK_R} \mathbb{P}\left\{\exists Y \in \mathcal{Y} : d(Y) = t, \Delta(Y) \le 0\right\} \\
\le &2e \sum_{t=K_L\wedge K_R}^{rK_LK_R} \left|\{Y \in \mathcal{Y} : d(Y) = t\}\right| \cdot \mathbb{P}\left\{d(Y) = t, \Delta(Y) \le 0\right\} \\
\le &2e \sum_{t=K_L\wedge K_R}^{rK_LK_R} \left(\frac{16t^2}{K_LK_R}\right)^2 n_L^{16t/K_R} n_R^{16t/K_L} \cdot \exp\left(-C\mu^2 t\right) \\
\overset{(a)}{\le} &2e \sum_{t=K_L\wedge K_R}^{rK_LK_R} 256 n^4 n^{-7t/(K_L\wedge K_R)} \le 512 e K_LK_R r n^{-3} \le 512 e n^{-1},
\end{aligned}$$

where (a) follows from the assumption that $\mu^2(K_L \wedge K_R) \ge C'\sigma^2 \log n$ for a sufficiently large constant C'. This means Y^* is the unique optimal solution to (22) with high probability.

6.3 Proof of Theorem 17

We have proved the theorem in Section 5.3.

6.4 Proof of Theorem 18

Note that the theorem assumes $n = n_L = n_R$ and $K = K_L = K_R$. Recall that J is the $n \times n$ all-one matrix, $\mathcal{R} := \text{support}(Y^*)$ and $\mathcal{A} := \text{support}(A)$, and $U, V \in \mathbb{R}^{n\times r}$ are the cluster characteristic matrices defined in Section 5.3, and $Y^* = KUV^\top$ is the SVD of Y^*. By relabeling the left nodes and right nodes, we can always make $U = V$ and thus we assume $U = V$ in the following proof.

Suppose Y^* is an optimal solution to the program. Then by the same argument used in the proof of Theorem 8, there must exist some $\lambda \ge 0$, η, W and H obeying the KKT conditions (63)–(65). Since $UU^\top WUU^\top = 0$ by (64), we can left and right multiply (63) by $UU^\top$ to obtain

$$\breve{A} - \lambda UU^\top - \eta J + \bar{H} = 0,$$

where for any matrix $X \in \mathbb{R}^{n\times n}$, we define the block-averaged matrix $\breve{X} := UU^\top XUU^\top$. Consider the last display equation on each entries in $\mathcal{R}$ and $\mathcal{R}^c$. By the Gaussian probability tail bound, there exists a universal constant $c_3 > 0$ such that with probability at least $1 - 2n^{-11}$,

$$\mu - \frac{\lambda}{K} - \eta + \bar{H}_{ij} \ge -\frac{c_3\sqrt{\log n}}{K}, \forall (i,j) \in \mathcal{R}, \tag{77}$$

$$-\eta + \bar{H}_{ij} \le \frac{c_3\sqrt{\log n}}{K}, \forall (i,j) \in \mathcal{R}^c. \tag{78}$$

Combining the last two display equations with (65), we get that

$$-\frac{c_3\sqrt{\log n}}{K} \le \eta \le \mu + \frac{c_3\sqrt{\log n}}{K} - \frac{\lambda}{K}.$$

It follows that

$$\lambda \le K\mu + 2c_3\sqrt{\log n} \le 4\max\left\{K\mu, c_3\sqrt{\log n}\right\}. \tag{79}$$

Furthermore, due to (77), (78) and $\lambda \ge 0$, we have

$$\bar{H}_{ij} \le \mu + \frac{2c_3\sqrt{\log n}}{K} \le \mu + \frac{1}{40}, \forall (i,j) \in \mathcal{R}^c, \tag{80}$$

where the last inequality holds when $K \ge c_1 \log n$.

On the other hand, the conditions (64) and (63) imply that

$$\begin{aligned}
\lambda^2 &= \left\|\lambda(UU^\top + W)\right\|^2 \ge \frac{1}{n}\left\|\lambda(UU^\top + W)\right\|_F^2 = \frac{1}{n}\left\|A - \eta J + H\right\|_F^2 \\
&= \frac{1}{n}\left(\left\|A_{\mathcal{R}} - \eta J_{\mathcal{R}} + H_{\mathcal{R}}\right\|_F^2 + \left\|A_{\mathcal{R}^c} - \eta J_{\mathcal{R}^c} + H_{\mathcal{R}^c}\right\|_F^2\right).
\end{aligned} \tag{81}$$

We now lower bound the RHS of (81). For each (i,j), define the Bernoulli random variables $\bar{b}_{ij} = \mathbf{1}(A_{ij} - \mathbb{E}A_{ij} \ge 1)$ and $\underline{b}_{ij} = \mathbf{1}(A_{ij} - \mathbb{E}A_{ij} \le -1)$, where $\mathbf{1}(\cdot)$ is the indicator function. By tail bounds of the standard Gaussian distribution, we have

$$\mathbb{P}\left(\bar{b}_{ij} = 1\right) = \mathbb{P}\left(\underline{b}_{ij} = 1\right) \ge \rho := \frac{1}{2\sqrt{2\pi}} e^{-1/2}.$$

Note that $\rho \ge \frac{1}{12}$. By Hoeffding's inequality, we know that with probability at least $1 - 2n^{-11}$,

$$\sum_{i,j\in\mathcal{R}^c} \bar{b}_{ij} \ge \frac{1}{2}\rho|\mathcal{R}^c| \quad \text{and} \quad \sum_{i,j\in\mathcal{R}^c} \underline{b}_{ij} \ge \frac{1}{2}\rho|\mathcal{R}^c|. \tag{82}$$

We consider two cases below.

- Case 1: $\eta \ge 40\mu$. By (80) and the Markov inequality, there is at most a fraction of $\frac{1}{30}$ of pairs (i,j) in $\mathcal{R}^c$ that satisfy $H_{ij} > 30\left(\mu + \frac{1}{40}\right)$. Let $\mathcal{D}$ denote the set of pairs (i,j) satisfying both $H_{ij} \le 30\left(\mu + \frac{1}{40}\right)$ and $A_{ij} \le -1$. In view of the second inequality in (82), we have $|\mathcal{D}|/|\mathcal{R}^c| \ge \rho/2 - 1/30 \ge 1/150$. Therefore, for $(i,j) \in \mathcal{D}$, we get that $-\eta + H_{\mathcal{R}^c} \le -10\mu + \frac{3}{4}$, and thus

$$\begin{aligned}
\left\|A_{\mathcal{R}^c} - \mu J_{\mathcal{R}^c} + H_{R^c}\right\|_F^2 &\ge \sum_{(i,j)\in\mathcal{D}} \left\|A_{\mathcal{R}^c} - \eta J_{\mathcal{R}^c} + H_{\mathcal{R}^c}\right\|_F^2 \\
&\ge \sum_{(i,j)\in\mathcal{D}} \left(-1 - 10\mu + \frac{3}{4}\right)^2 \ge \frac{1}{150}|\mathcal{R}^c| \cdot \frac{1}{16}.
\end{aligned}$$

- Case 2: $\eta \le 40\mu$. Since $\mu \le \frac{1}{100}$ by assumption, we have $\eta \le 1/2$. Therefore,

$$\|A_{\mathcal{R}^c} - \eta J_{\mathcal{R}^c} + H_{R^c}\|_F^2 \ge \sum_{(i,j)\in\mathcal{R}^c, \bar{b}_{ij}=1} \|A_{\mathcal{R}^c} - \eta J_{\mathcal{R}^c} + H_{\mathcal{R}^c}\|_F^2$$
$$\ge \sum_{(i,j)\in\mathcal{R}^c, \bar{b}_{ij}=1} (1-\eta)^2 \ge \frac{1}{2}\rho|\mathcal{R}^c| \cdot \frac{1}{4}.$$

Combining the two cases and substituting into (81), we obtain $\lambda^2 \ge c_4|\mathcal{R}^c|/n$ for some constant $c_4 > 0$. Since $|\mathcal{R}^c| = n^2 - rK^2 \ge n(n-K) \ge n^2/2$, we have $\lambda^2 \ge c_4 n/2$. It follows from (79) that

$$\max\left\{K\mu, c_3\sqrt{\log n}\right\} \ge \frac{\sqrt{c_4}}{4\sqrt{2}}\sqrt{n}.$$

Since $n \ge K \ge c_1 \log n$ with a sufficiently large constant c_1, we must have $K\mu \ge \frac{\sqrt{c_4}}{4\sqrt{2}}\sqrt{n}$. This violates the condition (28) in the theorem statement if we choose the universal constant c_2 sufficiently small. We conclude from this contradiction that Y^* is not an optimal solution of the convex program.

6.5 Proof of Theorem 20

We prove that with high probability, each of the three steps of the simple thresholding algorithm succeeds and thus Y^* is exactly recovered.

We first show that the simple thresholding algorithm correctly identifies all isolated nodes. Recall that $d_i = \sum_{j=1}^{n_R} A_{ij}$ is the row sum corresponding to the left node i. Observe that $d_i - \mathbb{E}[d_i]$ is the sum of n_R independent centered sub-Gaussian random variables with parameter 1. Moreover, $\mathbb{E}[d_i] = K_R\mu$ if node i is non-isolated; otherwise, $\mathbb{E}[d_i] = 0$. By Proposition 5.10 in Vershynin (2012), there exists a universal constant $c_3 > 0$ such that

$$\mathbb{P}\{|d_i - \mathbb{E}[d_i]| \ge K_R\mu/2\} \le e\exp\left(-\frac{c_3 K_R^2\mu^2}{n_R}\right) \le en_L^{-2},$$

where the last inequality follows from the assumption (29) by choosing the universal constant c_1 sufficiently large. By the union bound, we have with probability at least $1 - en_L^{-1}$, $d_i > \mu K_R/2$ for all non-isolated left nodes i and $d_i < \mu K_R/2$ for all isolated left nodes i. On this event, all isolated left nodes are correctly identified in Step 1 of the algorithm. A similar argument shows that all isolated right nodes are correctly identified with probability at least $1 - en_R^{-1}$. We use E_1 to denote the event that all the left and right isolated nodes are identified by the algorithm.

We first show that the simple thresholding algorithm correctly identifies all the left and right clusters. Recall that $S_{ii'} = \sum_{j=1}^{n_R} A_{ij}A_{i'j}$ is the inner product of two rows of A corresponding to the left nodes i and i'. If the two left nodes i, i' are in the same cluster, then $\mathbb{E}[S_{ii'}] = K_R\mu^2$; otherwise $\mathbb{E}[S_{ii'}] = 0$. Moreover, $A_{ij}A_{i'j}$ is the product of two independent sub-Gaussian random variables. We use $\|X\|_{\psi_2}$ and $\|X\|_{\psi_1}$ to denote the sub-Gaussian norm and sub-exponential norm[4]

4. The sub-exponential norm and sub-Gaussian norm of a random variable X are defined as $\|X\|_{\psi_i} = \sup_{p\ge 1} p^{-1/i}(\mathbb{E}|X|^p)^{1/p}$ for $i = 1, 2$, respectively (Vershynin, 2012).

It follows that

$$\left\|A_{ij}A_{i'j} - \mathbb{E}[A_{ij}]\mathbb{E}[A_{i'j}]\right\|_{\psi_1}$$
$$\overset{(a)}{\le} \left\|(A_{ij} - \mathbb{E}[A_{ij}])(A_{i'j} - \mathbb{E}[A_{i'j}])\right\|_{\psi_1} + \left\|(A_{ij} - \mathbb{E}[A_{ij}])\mathbb{E}[A_{i'j}]\right\|_{\psi_1} + \left\|(A_{ij} - \mathbb{E}[A_{ij}])\mathbb{E}[A_{i'j}]\right\|_{\psi_1}$$
$$\overset{(b)}{\le} 2\left\|A_{ij} - \mathbb{E}[A_{ij}]\right\|_{\psi_2}\left\|A_{i'j} - \mathbb{E}[A_{i'j}]\right\|_{\psi_2} + 2\mu\left\|A_{ij} - \mathbb{E}[A_{ij}]\right\|_{\psi_2} + 2\mu\left\|A_{i'j} - \mathbb{E}[A_{i'j}]\right\|_{\psi_2}$$
$$\overset{(c)}{\le} c'(4\mu + 2),$$

where (a) and (b) follow from $\|X+Y\|_{\psi_1} \le \|X\|_{\psi_1} + \|Y\|_{\psi_1}$ and $\|XY\|_{\psi_1} \le 2\|X\|_{\psi_2}\|Y\|_{\psi_2}$ for any random variables X, Y, and (c) holds for some universal constant $c' > 0$ because for each (i,j) $A_{ij} - \mathbb{E}[A_{ij}]$ is sub-Gaussian with parameter 1. By the Bernstein inequality for sub-exponential random variables given in Proposition 5.16 in Vershynin (2012), there exists some universal constant $c_4 > 0$ such that

$$\mathbb{P}\left\{|S_{ii'} - \mathbb{E}[S_{ii'}]| \ge K_R\mu^2/2\right\} \le e\exp\left[-c_4\min\left(\frac{K_R^2\mu^4}{n_R c'^2(4\mu+2)^2}, \frac{K_R\mu^2}{c'(4\mu+2)}\right)\right] \le e(rK_L)^{-3},$$

where the last inequality follows from the conditions (30) and (29). By the union bound, we get that with probability at least $1 - e(rK_L)^{-1}$, $S_{ii'} > \frac{\mu^2 K_R}{2}$ for all left nodes i, i' from the same left cluster and $S_{ii'} < \frac{\mu^2 K_R}{2}$ for all left nodes i, i' from two different left clusters. On the intersection of this event and the event E_1 defined above, Step 2 of the algorithm identifies the true left clusters. A similar argument shows that the algorithm also identifies the true right clusters with probability at least $1 - e(rK_R)^{-1}$. We use E_2 to denote that event that all the true left and right clusters are identified by the algorithm.

Finally, we show that the simple thresholding algorithm correctly associates all the left and right clusters. Let $B^*_{kl} := \sum_{i\in C_k^*, j\in D_l^*} A_{ij}$ be the block sum of A corresponding to the true let and right clusters C_k^* and D_l^*. By model assumptions, $B^*_{kl} - \mathbb{E}[B^*_{kl}]$ is a sum of $K_L K_R$ independent centered sub-Gaussian random variables with parameter 1. Moreover, $\mathbb{E}[B^*_{kl}] = \mu K_L K_R$ if $k = l$, and $\mathbb{E}[B^*_{kl}] = 0$ otherwise. By the standard sub-Gaussian concentration inequality given in Proposition 5.10 in Vershynin (2012), there exists some universal constant $c_5 > 0$ such that

$$\mathbb{P}\{|B^*_{kl} - \mathbb{E}[B^*_{kl}]| \ge \mu K_L K_R/2\} \le e\exp\left(-\frac{c_5\mu^2 K_L^2 K_R^2}{K_L K_R}\right) \le en^{-3},$$

where the last inequality holds because $\mu^2 K_L K_R \ge c_1 \log n$ in view of (29). By the union bound, we get that with probability at least $1 - en^{-1}$, $B^*_{kl} < \mu K_L K_R/2$ for all $k = l$ and $B^*_{kl} > \mu K_L K_R/2$ for all $k \ne l$. On the intersection of this event and the event E_2defined above, the quantities $\{B_{kl}\}$ used in Step 3 of the algorithm satisfy $B_{kl} = B^*_{kl}$, and the algorithm correctly associates the left and right clusters.

6.6 Proof of Theorem 21

We focus on identifying left isolated nodes and left clusters. The proof for the right nodes is identical. The main idea is to show that some of the d_i and $S_{ii'}$'s will have large deviation from their expectations.

Assume $rK_L \geq n_L/2$ first. We will show that if $K_R^2\mu^2 \leq c_1 n_R \log n_L$ for a sufficiently small universal constant c_1, then with high probability there exists a non-isolated left node i^* that is incorrectly declared as isolated. Recall that $d_i = \sum_{j=1}^{n_R} A_{ij}$ is the row sum corresponding to the left node i. If the left node i is non-isolated, then d_i is Gaussian with mean $K_R\mu$ and variance n_R. For a standard Gaussian random variable Z, its tail probability is lower bounded as $Q(t) := \mathbb{P}[Z \geq t] \geq \frac{1}{\sqrt{2\pi}}\frac{t}{t^2+1}\exp(-t^2/2)$. It follows that for a non-isolated left node i, there exist two positive universal constants c_3, c_4 such that

$$\mathbb{P}\left[d_i - \mathbb{E}[d_i] \leq K_R\mu/2\right] \geq c_3 \exp\left(-\frac{c_4 K_R^2\mu^2}{n_R}\right) \geq c_3 n_L^{-c_1 c_4}.$$

Let i^* be the non-isolated left node with the minimum d_i. Since $\{d_i\}_{i=1}^{n_L}$ are mutually independent, we have

$$\mathbb{P}\left[d_{i^*} > \frac{K_R\mu}{2}\right] \leq \left(1 - c_3 n_L^{-c_1 c_4}\right)^{rK_L} \leq \exp\left(-\frac{1}{2}c_3 n_L^{1-c_1 c_4}\right),$$

where the last inequality holds because $rK_L \geq n_L/2$. By choosing c_1 sufficiently small, we conclude that with high probability the non-isolated left node i^* will be incorrectly declared as an isolated node.

If $rK_L \leq n_L/2$, then we can similarly show that if $K_R^2\mu^2 \leq c_1 n_R \log n_L$ for a sufficiently small c_1, then with high probability there exists an isolated left node i^{**} incorrectly declared as non-isolated.

We next show that if

$$K_R^2\mu^4 \leq c_2 n_R \log(rK_L) \tag{83}$$

for a sufficiently small constant c_2, then there exist two left nodes i_1, i_2 in two different clusters that will be incorrectly assigned to the same cluster.

By the assumption that $n_R = \Omega(rK_L)$, the inequality (83) implies $K_R\mu^3 \leq c_2^{3/4} n_R$. Recall that $S_{ii'} = \sum_{j=1}^{n_R} A_{ij}A_{i'j}$. For two left nodes i, i' from two different clusters, we have

$$\mathbb{E}[S_{ii'}] = 0, \quad \mathrm{Var}[S_{ii'}] = 2K_R\mu^2 + n_R,$$
$$\sum_{j=1}^{n_R}\mathbb{E}[|A_{ij}A_{i'j}|^3] = \sum_{j=1}^{n_R}\mathbb{E}[|A_{ij}|^3]\mathbb{E}[|A_{i'j}|^3] \leq c_5(K_R\mu^3 + n_R) \leq c_5(c_2^{3/4}+1)n_R,$$

where c_5 is some universal positive constant. By the Berry-Esseen theorem, there exists a positive universal constant c_6 such that

$$\begin{aligned}
\mathbb{P}\left[S_{ii'} \geq \frac{\mu^2 K_R}{2}\right] &\geq Q\left(\frac{\mu^2 K_R}{2\sqrt{2K_R\mu^2 + n_R}}\right) - \frac{c_6(K_R\mu^3 + n_R)}{(2K_R\mu^2 + n_R)^{3/2}} \\
&\overset{(a)}{\geq} Q\left(\frac{\mu^2 K_R}{\sqrt{n_R}}\right) - \frac{c_6 c_5(c_2^{3/4}+1)}{\sqrt{n_R}} \\
&\overset{(b)}{\geq} Q\left(\sqrt{c_2\log(rK_L)}\right) - \frac{c_6 c_5(c_2^{3/4}+1)}{\sqrt{rK_L}} \\
&\overset{(c)}{\geq} c_3(rK_L)^{-c_4 c_2} - c_6 c_5(c_2^{3/4}+1)(rK_L)^{-1/2} \overset{(d)}{\geq} c_7(rK_L)^{-c_4 c_2},
\end{aligned}$$

where (a) holds because $Q(t)$ is non-increasing in t, (b) holds in view of (83) and the assumption that $n_R \geq rK_L$, (c) follows from $Q(t) \geq c_3\exp(-c_4 t^2)$, and (d) holds for some universal constant $c_7 > 0$ by choosing c_2 sufficiently small.

Set $(i_1, i_2) := \arg\max_{(i,i')\in W} S_{ii'}$, where W is a maximal set of node pairs (i, i') satisfying 1) i and i' are from two different clusters, and 2) for any $(i, i'), (j, j') \in W$, the nodes i, i', j, j' are all distinct. Then $|W| \geq rK_L/4$ and $\{S_{ii'} : (i, i') \in W\}$ are mutually independent. It follows that

$$\mathbb{P}\left[S_{i_1 i_2} < \frac{\mu^2 K_R}{2}\right] \leq \left(1 - c_7(rK_L)^{-c_4 c_2}\right)^{rK_L/4} \leq \exp\left(-\frac{1}{4}c_7(rK_L)^{1-c_4 c_2}\right).$$

Therefore, with probability at least $1 - \exp\left(-\frac{1}{4}c_7(rK_L)^{1-c_4 c_2}\right)$, we have $S_{i_1 i_2} \geq \frac{\mu^2 K_R}{2}$. On this event, (i_1, i_2) will be incorrectly assigned to the same cluster.

6.7 Proof of Theorem 22

We prove the first part of the theorem. Since A_{ij} are sub-Gaussian, there exists a universal constant $c_1 > 0$ such that $\mathbb{P}\left[|A_{ij} - \mathbb{E}A_{ij}| \leq \frac{1}{2}\sqrt{c_1\log n}\right] \geq 1 - n^{-12}$ for each (i, j). Recall that $\mathcal{R} := \mathrm{support}(Y^*)$. By the union bound over all (i, j), we obtain that with probability at least $1 - n^{-3}$,

$$\min_{(i,j)\in\mathcal{R}} A_{ij} > \mu - \frac{1}{2}\sqrt{c_1\log n} \overset{(a)}{>} \frac{1}{2}\mu \qquad \text{and} \qquad \max_{(i,j)\in\mathcal{R}^c} A_{ij} < \frac{1}{2}\sqrt{c_1\log n} \overset{(b)}{<} \frac{1}{2}\mu,$$

where (a) and (b) holds in view of the assumption (33). Therefore, the algorithm sets $\widehat{Y}_{ij} = 1$ for $(i, j) \in \mathcal{R}$ and $\widehat{Y}_{ij} = 0$ for $(i, j) \in \mathcal{R}^c$, which implies $\widehat{Y} = Y^*$.

For the second part of the theorem, note that $\{A_{ij}\}$ are Gaussian variables and thus

$$\mathbb{P}\left[A_{ij} \geq \mathbb{E}A_{ij} + \sqrt{2\log n}\right] = Q\left(\sqrt{2\log n}\right) \geq \frac{\sqrt{2\log n}}{\sqrt{2\pi}(2\log n + 1)n},$$
$$\mathbb{P}\left[A_{ij} \leq \mathbb{E}A_{ij} - \sqrt{2\log n}\right] = Q\left(\sqrt{2\log n}\right) \geq \frac{\sqrt{2\log n}}{\sqrt{2\pi}(2\log n + 1)n},$$

where the last inequality holds due to $Q(t) \geq \frac{1}{\sqrt{2\pi}}\frac{t}{t^2+1}\exp(-t^2/2)$. By independence of the entries of A, we obtain

$$\mathbb{P}\left[\max_{i,j\in\mathcal{R}^c} A_{ij} < \sqrt{2\log n}\right] \leq \left(1 - \frac{\sqrt{2\log n}}{\sqrt{2\pi}(2\log n+1)n}\right)^{|\mathcal{R}^c|} \leq \exp\left(-\frac{\sqrt{2\log n}|\mathcal{R}^c|}{\sqrt{2\pi}(2\log n+1)n}\right),$$
$$\mathbb{P}\left[\min_{i,j\in\mathcal{R}} A_{ij} > \mu - \sqrt{2\log n}\right] \leq \left(1 - \frac{\sqrt{2\log n}}{\sqrt{2\pi}(2\log n+1)n}\right)^{|\mathcal{R}|} \leq \exp\left(-\frac{\sqrt{2\log n}|\mathcal{R}|}{\sqrt{2\pi}(2\log n+1)n}\right),$$

Since $|\mathcal{R}| + |\mathcal{R}^c| = n^2$, we must have

$$\min\left\{\mathbb{P}\left[\max_{i,j\in\mathcal{R}^c} A_{ij} < \sqrt{2\log n}\right], \mathbb{P}\left[\min_{i,j\in\mathcal{R}} A_{ij} > \mu - \sqrt{2\log n}\right]\right\} \leq e^{-\Omega(n/\sqrt{\log n})}.$$

This inequality, together with the assumption (34), implies that either with probability at least $1 - e^{-\Omega(n/\sqrt{\log n})}$, at least one of the following must occur: $\max_{(i,j)\in\mathcal{R}^c} A_{ij} \geq \sqrt{2\log n} \geq \frac{3}{5}\mu$ and

$\min_{(i,j)\in R} A_{ij} \le \mu - \sqrt{2\log n} \le \frac{2}{5}\mu$. On this event, the output of the element-wise thresholding algorithm satisfies $\widehat{Y} \neq Y^*$.

Acknowledgments

The authors would like to thank Sivaraman Balakrishnan, Bruce Hajek and Martin J. Wainwright for inspiring discussions. Y. Chen was supported in part by NSF grant CIF-31712-23800, ONR MURI grant N00014-11-1-0688, and a start-up fund from the School of Operations Research and Information Engineering at Cornell University. J. Xu was supported in part by the National Science Foundation under Grant ECCS 10-28464, IIS-1447879, and CCF-1423088, and Strategic Research Initiative on Big-Data Analytics of the College of Engineering at the University of Illinois, and DOD ONR Grant N00014-14-1-0823, and Grant 328025 from the Simons Foundation.

Appendix A. Proof of Lemmas 26 and 31

Notice that Lemma 26 is a special case of Lemma 31 with $n_L = n_R$, $K_L = K_R$ and the left clusters identical to the right clusters. Hence we only need to prove Lemma 31.

Recall that $C_1^*, \ldots, C_r^*$ ($D_1^*, \ldots, D_r^*$, resp.) denote the true left (right, resp.) clusters associated with Y^*. The nodes in $V_L \setminus (\cup_{k=1}^r C_k^*)$ do not belong to any left clusters and are called isolated left nodes. Isolated right nodes are similarly defined.

Fix a $Y \in \mathcal{Y}$ with $d(Y) := \langle Y^*, Y - Y^* \rangle = t$. Based on Y, we construct a new ordered partition $(C_1, \ldots, C_{r+1})$ of V_L and a new ordered partition $(D_1, \ldots, D_{r+1})$ of V_R as follows.

1. Let $C_{r+1} := \{i : Y_{ij} = 0, \forall j\}$ and $D_{r+1} := \{j : Y_{ij} = 0, \forall i\}$.

2. The left nodes in $V_L \setminus C_{r+1}$ are further partitioned into r new left clusters of size K_L, such that the left nodes i and i' are in the same cluster if and only if the i-th and i'-th rows of Y are identical. Similarly, the right nodes in $V_R \setminus D_{r+1}$ are partitioned into r new right clusters of size K_R according to the columns of Y. We now define an ordering $C_1, \ldots, C_r$ of these r new left clusters and an ordering $D_1, \ldots, D_r$ for the new right clusters using the following procedure.

   (a) For each new left cluster C, if there exists a $k \in [r]$ such that $|C \cap C_k^*| > K_L/2$, then we label this new left cluster as C_k; this label is unique because the left cluster size is K_L. We associate C_k with the right cluster $\{j : Y_{ij} = 1, \forall i \in C_k\}$, which is labeled as D_k.

   (b) For each remaining unlabeled right cluster D, if there exists a $k \in [r]$ such that $|D \cap D_k^*| > K_R/2$, then we label this new right cluster as D_k; again this label is unique. We associate D_k with the left cluster $\{i : Y_{ij} = 1, \forall j \in D_k\}$, which is labeled as C_k.

   (c) The remaining unlabeled left clusters are labeled arbitrarily. For each remaining unlabeled right cluster, we label it according to $D_k := \{j : Y_{ij} = 1, \forall i \in C_k\}$.

For each $(k, k') \in [r] \times [r+1]$, we use $\alpha_{kk'} := |C_k^* \cap C_{k'}|$ and $\beta_{kk'} := |D_k^* \cap D_{k'}|$ to denote the sizes of intersections of the true and new clusters. We observe that the new clusters $(C_1, \ldots, C_{r+1}, D_1, \ldots, D_{r+1})$ have the following three properties:

(A0) $(C_1, \ldots, C_r, C_{r+1})$ is a partition of V_L with $|C_k| = K_L$ for each $k \in [r]$; $(D_1, \ldots, D_r, D_{r+1})$ is a partition of V_R with $|D_k| = K_R$ for each $k \in [r]$.

(A1) For each $k \in [r]$, exactly one of the following is true: (1) $\alpha_{kk} > K_L/2$; (2) $\alpha_{kk'} \le K_L/2$ for all $k' \in [r]$ and $\beta_{kk} > K_R/2$; (3) $\alpha_{kk'} \le K_L/2$ and $\beta_{kk'} \le K_R/2$ for all $k' \in [r]$.

(A2) We have

$$\sum_{k=1}^{r} \left(\alpha_{k(r+1)}\beta_{k(r+1)} + \sum_{k', k'' : k' \neq k''} \alpha_{kk'}\beta_{kk''} \right) = t;$$

here and henceforth, all summations involving k' or k'' (as the indices of the new clusters) are over the range $[r+1]$ unless defined otherwise.

Here, Property (A0) holds due to $Y \in \mathcal{Y}$, Property (A1) is direct consequence of how we label the new clusters, and Property (A2) follows from the following:

$$\begin{aligned} t = d(Y) &= \sum_{k=1}^{r} |\{(i,j) : (i,j) \in C_k^* \times D_k^*, Y_{ij} = 0\}| \\ &= \sum_{k=1}^{r} |\{(i,j) : (i,j) \in C_k^* \times D_k^*, (i,j) \in C_{r+1} \times D_{r+1}\}| \\ &\quad + \sum_{k=1}^{r} \sum_{(k',k'') : k' \neq k''} |\{(i,j) : (i,j) \in C_k^* \times D_k^*, (i,j) \in C_{k'} \times D_{k''}\}|. \end{aligned}$$

Since a different Y corresponds to a different ordered partition, and the ordered partition for any given Y with $d(Y) = t$ must satisfy the above three properties, we obtain the following bound on the cardinality of the set of interest:

$$|\{Y \in \mathcal{Y} : d(Y) = t\}| \le |\{(C_1, \ldots, C_{r+1}, D_1, \ldots, D_{r+1}) : \text{it satisfies (A0)–(A2)}\}|. \tag{84}$$

It remains to upper-bound the right hand side of (84).

Fix any ordered partition $(C_1, \ldots, C_r, C_{r+1}, D_1, \ldots, D_r, D_{r+1})$ with Properties (A0)–(A2). Consider the first true left cluster C_1^*. Define $m_1^{(L)} := \sum_{k' : k' \neq 1} \alpha_{1k'}$, which can be considered as the number of nodes in C_1^* that are misclassified by Y. Analogously define $m_1^{(R)} := \sum_{k'' : k'' \neq 1} \beta_{1k''}$. We consider the following two cases for the values of α_{11}.

- If $\alpha_{11} > K_L/4$, then

$$\sum_{(k',k'') : k' \neq k''} \alpha_{1k'}\beta_{1k''} \ge \alpha_{11} \sum_{k'' : k'' \neq 1} \beta_{1k''} > \frac{1}{4} m_1^{(R)} K_L.$$

- If $\alpha_{11} \le K_L/4$, then $m_1^{(L)} \ge 3K_L/4$, and we must also have $\alpha_{1k'} \le K_L/2$ for all $1 \le k' \le r$ by Property (A1). Hence,

$$\begin{aligned}&\sum_{(k',k''):k'\neq k''} \alpha_{1k'}\beta_{1k''} + \alpha_{1(r+1)}\beta_{1(r+1)}\\ \ge &\sum_{(k',k''):k'\neq k''} \mathbf{1}\left\{k' \neq 1\right\}\mathbf{1}\left\{k'' \neq 1\right\}\alpha_{1k'}\beta_{1k''} + \alpha_{1(r+1)}\beta_{1(r+1)}\\ =&m_1^{(L)}m_1^{(R)} - \sum_{2\le k'\le r}\alpha_{1k'}\beta_{1k'} \ge m_1^{(L)}m_1^{(R)} - \frac{1}{2}K_L m_1^{(R)} \ge \frac{1}{4}m_1^{(R)}K_L.\end{aligned}$$

  Similarly, we consider the following three cases for the values of β_{11}.

- If $\beta_{11} > K_R/4$, then

$$\sum_{(k',k''):k'\neq k''} \alpha_{1k'}\beta_{1k''} \ge \beta_{11}\sum_{k':k'\neq 1}\alpha_{1k'} > \frac{1}{4}m_1^{(L)}K_R.$$

- If $\beta_{11} \le K_R/4$ and $\beta_{1k''} \le K_L/2$ for all $1 < k'' \le r$, then, similarly to the second case for α_{11} above, we have

$$\sum_{(k',k''):k'\neq k''} \alpha_{1k'}\beta_{1k''} + \alpha_{1(r+1)}\beta_{1(r+1)} \ge \frac{1}{4}m_1^{(L)}K_R.$$

- If $\beta_{11} \le K_R/4$ and $\beta_{1k_0} > K_R/2$ for some $1 < k_0 \le r$, then by Property (A1) we must have $\alpha_{11} > K_L/2$. It follows that $m_1^{(L)} < K_L/2$ and

$$\sum_{(k',k''):k'\neq k''} \alpha_{1k'}\beta_{1k''} \ge \alpha_{11}\beta_{1k_0} > K_L K_R/4 \ge \frac{1}{2}m_1^{(L)}K_R.$$

Combining the above five cases, we conclude that the following is always true:

$$\sum_{(k',k''):k'\neq k''} \alpha_{1k'}\beta_{1k''} + \alpha_{1(r+1)}\beta_{1(r+1)} \ge \frac{1}{4}\left(m_1^{(L)}K_R \vee m_1^{(R)}K_L\right).$$

This inequality continues to hold if we replace $\alpha_{1k'}$, $\beta_{1k''}$, $m_1^{(L)}$ and $m_1^{(R)}$ respectively by $\alpha_{kk'}$, $\beta_{kk''}$, $m_k^{(L)}$ and $m_k^{(R)}$ (defined in a similar manner) for each $k \in [r]$. Summing these inequalities over $k \in [r]$ and using Property (A2), we obtain

$$t = \sum_{k=1}^r\left\{\alpha_{k(r+1)}\beta_{k(r+1)} + \sum_{(k',k''):k'\neq k''}\alpha_{kk'}\beta_{kk''}\right\} \ge \left(\frac{K_L}{4}\sum_{k=1}^r m_k^{(R)}\right) \vee \left(\frac{K_R}{4}\sum_{k=1}^r m_k^{(L)}\right).$$

Consequently, we have $\sum_{k\in[r]} m_k^{(L)} \le 4t/K_R$ and $\sum_{k\in[r]} m_k^{(R)} \le 4t/K_L$; i.e., the total number of misclassified non-isolated left (right, resp.) nodes is upper bounded by $4t/K_R$ ($4t/K_L$, resp.). This means that the total number of misclassified isolated left (right, resp.) nodes is also upper bounded

by $4t/K_R$ ($4t/K_L$, resp.), because by the cluster size constraint in Property (A0), one misclassified isolated node must produce one misclassified non-isolated node.

We can now upper-bound the right hand side of (84) using the above relation between the value of t and the misclassified nodes. For each Y with $d(Y) = t$, the pair

$$(\#\text{misclassified isolated left nodes},\ \ \#\text{misclassified non-isolated left nodes})$$

can take at most $(4t/K_R)^2$ different values. Similarly for the right nodes we have the bound $(4t/K_L)^2$. Given these numbers of misclassified nodes, there are at most $n_L^{8t/K_R}n_R^{8t/K_L}$ different ways to choose the identity of these misclassified nodes. Each misclassified non-isolated left node can then be assigned to one of $r-1 \le n_L$ different left clusters or left isolated, and each misclassified isolated left node can be assigned to one of $r \le n_L$ different left clusters; an analogous statement holds for the right nodes. Hence, the right hand side of (84) is upper bounded by $\left(\frac{16t^2}{K_L K_R}\right)^2 n_L^{16t/K_R}n_R^{16t/K_L}$. This proves the first part of the lemma.

To count the number of possible equivalence classes $[Y]$, we use a similar argument but only need to consider the misclassified *non-isolated* nodes. The number of misclassified non-isolated left (right, resp.) nodes can take at most $4t/K_R$ ($4t/K_L$, resp.) different values. Given these numbers, there are at most $(rK_L)^{4t/K_R}(rK_R)^{4t/K_L}$ different ways to choose the identity of the misclassified non-isolated nodes. Each misclassified non-isolated left (right, resp.) node can then be assigned to one of $r-1$ different left (right, resp.) clusters or left isolated. Therefore, the number of possible equivalence classes $[Y]$ with $d(Y) = t$ is upper bounded by $\frac{16t^2}{K_L K_R}(rK_L)^{8t/K_R}(rK_R)^{8t/K_L}$.

Appendix B. Proof of Lemma 27

The inequality (56) follows from (55) by replacing p with $1-q'$ and q with $1-p'$, so it suffices to prove (55). Note that for $1 \ge u \ge v \ge 0$, we have

$$D(u\|v) = u\log\frac{u}{v} + (1-u)\log\frac{1-u}{1-v} \le u\log\frac{u}{v}, \tag{85}$$

$$D(u\|v) \ge u\log\frac{u}{v} + (1-u)\log(1-u) \overset{(a)}{\ge} u\log\frac{u}{ev}, \tag{86}$$

where (a) follows from the inequality $x\log x \ge x-1, \forall x \in [0,1]$. We consider two cases:

Case 1: $p \le 8q$. In view of (35) and (36), we have $D(p\|q) \le \frac{(p-q)^2}{q(1-q)}$ and $D\left(\frac{p+q}{2}\|q\right) \ge \frac{(p-q)^2}{4(p+q)(1-q)}$. Since $p \le 8q$, it follows that $D\left(\frac{p+q}{2}\|q\right) \ge \frac{(p-q)^2}{36q(1-q)} \ge \frac{1}{36}D(p\|q)$.

Case 2: $p > 8q$. In view of (85) and (86), we have $D(p\|q) \le p\log\frac{p}{q}$ and $D\left(\frac{p+q}{2}\|q\right) \ge \frac{p+q}{2}\log\frac{p+q}{2eq}$. Since $p > 8q$ and $8 > (2e)^{(6/5)}$, we have $\log\frac{p}{q} > \frac{6}{5}\log(2e)$ and thus $\log\frac{p+q}{2eq} > \log\frac{p}{2eq} = \log\frac{p}{q} - \log(2e) > \frac{1}{6}\log\frac{p}{q}$. It follows that $D\left(\frac{p+q}{2}\|q\right) \ge \frac{p}{2}\cdot\frac{1}{6}\log\frac{p}{q} \ge \frac{1}{12}D(p\|q)$.

Appendix C. The Bernstein Inequality

Theorem 32 (Bernstein) *Let $X_1, \ldots, X_N$ be independent random variables such that $|X_i| \le M$ almost surely. Let $\sigma^2 = \sum_{i=1}^N Var(X_i)$. Then for any $t \ge 0$,*

$$\mathbb{P}\left[\sum_{i=1}^N X_i \ge t\right] \le \exp\left(\frac{-t^2}{2\sigma^2 + \frac{2}{3}Mt}\right).$$

A consequent of the above inequality is $\mathbb{P}\left[\sum_{i=1}^{N} X_i \geq \sqrt{2\sigma^2 u} + \frac{2Mu}{3}\right] \leq e^{-u}$ *for any* $u > 0$.

References

E. Abbe and C. Sandon. Community detection in general stochastic block models: fundamental limits and efficient recovery algorithms. *arXiv:1503.00609*, 2015.

E. Abbe, A. S. Bandeira, and G. Hall. Exact recovery in the stochastic block model. *arXiv:1405.3267*, 2014.

N. Ailon, Y. Chen, and H. Xu. Breaking the small cluster barrier of graph clustering. In *Proceedings of the 30th International Conference on Machine Learning*, pages 995–1003, 2013.

N. Alon and N. Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM Journal on Computing*, 26(6):1733–1748, 1997.

N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. *Random Structures and Algorithms*, 13(3-4):457–466, 1998.

N. Alon, A. Andoni, T. Kaufman, K. Matulef, R. Rubinfeld, and N. Xie. Testing k-wise and almost k-wise independence. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 496–505. ACM, 2007.

B. P. W. Ames. Guaranteed clustering and biclustering via semidefinite programming. *Mathematical Programming*, pages 1–37, 2013.

B. P. W. Ames and S. A. Vavasis. Nuclear norm minimization for the planted clique and biclique problems. *Mathematical programming*, 129(1):69–89, 2011.

B. P. W. Ames and S.A. Vavasis. Convex optimization for the planted k-disjoint-clique problem. *Mathematical Programming*, 143(1–2):299–337, 2014.

A. Amini, A. Chen, P. J. Bickel, and E. Levina. Pseudo-likelihood methods for community detection in large sparse networks. *The Annals of Statistics*, 41(4):2097–2122, 2013.

A. A. Amini and M. J. Wainwright. High-dimensional analysis of semidefinite relaxations for sparse principal components. *The Annals of Statistics*, 37(5):2877–2921, 2009.

A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor spectral approach to learning mixed membership community models. *Journal of Machine Learning Research*, 15:2239–2312, June 2014.

E. Arias-Castro and N. Verzelen. Community detection in dense random networks. *The Annals of Statistics*, 42(3):940–969, 06 2014.

E. Arias-Castro, E. J. Candès, and A. Durand. Detection of an anomalous cluster in a network. *The Annals of Statistics*, 39(1):278–304, 2011.

S. Balakrishnan, M. Kolar, A. Rinaldo, A. Singh, and L. Wasserman. Statistical and computational tradeoffs in biclustering. In *NIPS 2011 Workshop on Computational Trade-offs in Statistical Learning*, 2011a.

S. Balakrishnan, M. Xu, A. Krishnamurthy, and A. Singh. Noise thresholds for spectral clustering. In *Advances in Neural Information Processing Systems 25*, 2011b.

A. S. Bandeira. Random Laplacian matrices and convex relaxations. *arXiv:1504.03987*, 2015.

N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1):89–113, 2004.

Q. Berthet and P. Rigollet. Complexity theoretic lower bounds for sparse principal component detection. *Journal of Machine Learning Research: Workshop and Conference Proceedings*, 30: 1046–1066, 2013.

S. Bhamidi, P. S. Dey, and A. B. Nobel. Energy landscape for large average submatrix detection problems in Gaussian random matrices. *arXiv:1211.2284*, 2012.

P. J. Bickel and A. Chen. A nonparametric view of network models and Newman–Girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073, 2009.

B. Bollobás and A. D. Scott. Max cut for random graphs with a planted partition. *Combinatorics, Probability and Computing*, 13(4-5):451–474, 2004.

C. Butucea and Y. I. Ingster. Detection of a sparse submatrix of a high-dimensional noisy matrix. *Bernoulli*, 19(5B):2652–2688, 2013.

T. T. Cai and X. Li. Robust and computationally feasible community detection in the presence of arbitrary outlier nodes. *The Annals of Statistics*, 43(3):1027–1059, 06 2015.

V. Chandrasekaran and M. I. Jordan. Computational and statistical tradeoffs via convex relaxation. *Proceedings of the National Academy of Sciences*, 110(13):E1181–E1190, 2013.

S. Chatterjee. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2014.

K. Chaudhuri, F. Chung, and A. Tsiatas. Spectral clustering of graphs with general degrees in the extended planted partition model. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, pages 35.1–35.23, 2012.

Y. Chen. Incoherence-optimal matrix completion. *IEEE Transactions on Information Theory*, 61 (5):2909–2923, 2015.

Y. Chen and J. Xu. Statistical-computational phase transitions in planted models: The high-dimensional setting. In *Proceedings of the 31st International Conference on Machine Learning*, pages 244–252, 2014.

Y. Chen, S. Sanghavi, and H. Xu. Clustering sparse graphs. In *Proceedings of the Neural Information Processing Systems Conferece*, pages 2204–2212, 2012.

Y. Chen, A. Jalali, S. Sanghavi, and H. Xu. Clustering partially observed graphs via convex optimization. *Journal of Machine Learning Research*, 15:2213–2238, June 2014a.

Y. Chen, S. Sanghavi, and H. Xu. Improved graph clustering. *IEEE Transactions on Information Theory*, 60(10):6440–6455, 2014b.

A. Coja-Oghlan. Coloring semirandom graphs optimally. *Automata, Languages and Programming*, pages 383–395, 2004.

A. Condon and R. M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Struct. Algorithms*, 18(2):116–140, Mar 2001.

A. Decelle, F. Krzakala, C. Moore, and L. Zdeborova. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physics Review E*, 84:066106, 2011.

Y. Dekel, O. Gurel-Gurevich, and Y. Peres. Finding hidden cliques in linear time with high probability. *Combinatorics, Probability and Computing*, 23(01):29–49, 2014.

Y. Deshpande and A. Montanari. Finding hidden cliques of size $\sqrt{N/e}$ in nearly linear time. *Foundations of Computational Mathematics*, pages 1–60, September 2013.

R. Durrett. *Random Graph Dynamics*. Cambridge University Press, New York, NY, 2007.

M. E. Dyer and A. M. Frieze. The solution of some random NP-hard problems in polynomial expected time. *Journal of Algorithms*, 10(4):451 – 489, 1989.

V. Feldman, E. Grigorescu, L. Reyzin, S. Vempala, and Y. Xiao. Statistical algorithms and a lower bound for detecting planted cliques. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 655–664. ACM, 2013.

S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

G. R. Grimmett and C. J. H. McDiarmid. On colouring random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 77(2):313–324, 1975.

B. Hajek, Y. Wu, and J. Xu. Achieving exact cluster recovery threshold via semidefinite programming. *arXiv:1412.6156*, 2014.

B. Hajek, Y. Wu, and J. Xu. Achieving exact cluster recovery threshold via semidefinite programming: Extensions. *arXiv:1502.07738*, 2015.

B. Hajek, Y. Wu, and J. Xu. Computational lower bounds for community detection on random graphs. *arXiv:1406.6625. The conference version appeared in Proceedings of COLT 2015*, June, 2014.

E. Hazan and R. Krauthgamer. How hard is it to approximate the best Nash equilibrium? *SIAM Journal on Computing*, 40(1):79–91, 2011.

P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

A. Jalali and N. Srebro. Clustering using max-norm constrained optimization. In *Proceedings of the 29th International Conference on Machine Learning*, pages 481–488, 2012.

A. Juels and M. Peinado. Hiding cliques for cryptographic security. *Designs, Codes and Cryptography*, 20(3):269–280, 2000.

P. Koiran and A. Zouzias. Hidden cliques and the certification of the restricted isometry property. *IEEE Transactions on Information Theory*, 60(8):4999–5006, 2014.

M. Kolar, S. Balakrishnan, A. Rinaldo, and A. Singh. Minimax localization of structural information in large noisy matrices. In *Advances in Neural Information Processing Systems*, 2011.

R. Krauthgamer, B. Nadler, and D. Vilenchik. Do semidefinite relaxations solve sparse PCA up to the information limit? *The Annals of Statistics*, 43(3):1300–1322, 06 2015.

L. Kučera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57 (2–3):193–212, Feb. 1995.

M. Lelarge, L. Massoulié, and J. Xu. Reconstruction in the labeled stochastic block model. In *IEEE Information Theory Workshop (ITW)*, pages 1–5, 2013.

J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *Proceedings of the 17th international conference on World Wide Web*, pages 695–704. ACM, 2008.

Z. Ma and Y. Wu. Computational barriers in minimax submatrix detection. *The Annals of Statistics*, 43(3):1089–1116, 2015.

L. Massoulié. Community detection thresholds and the weak Ramanujan property. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 694–703. ACM, 2014.

L. Massoulié and D. Tomozei. Distributed user profiling via spectral methods. *Stochastic Systems*, 4(1):1–43, 2014.

C. Mathieu and W. Schudy. Correlation clustering with noisy input. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 712–728. SIAM, 2010.

J. Matoušek and J. Vondrák. The probabilistic method, lecture notes. *Available at http://kam.mff.cuni.cz/~matousek/prob-ln-2pp.ps.gz*, 2008.

F. McSherry. Spectral partitioning of random graphs. In *42nd IEEE Symposium on Foundations of Computer Science*, pages 529 – 537, Oct. 2001.

E. Mossel, J. Neeman, and A. Sly. A proof of the block model threshold conjecture. *arxiv:1311.4115*, 2013.

E. Mossel, J. Neeman, and A. Sly. Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162(3-4):431–461, 2015a. ISSN 0178-8051.

E. Mossel, J. Neeman, and A. Sly. Consistency thresholds for the planted bisection model. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '15, pages 69–75, New York, NY, USA, 2015b. ACM.

R. R. Nadakuditi and M. E. J. Newman. Graph spectra and the detectability of community structure in networks. *Physical Review Letters*, 108(18):188–701, 2012.

M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, Feb 2004.

S. Oymak, A. Jalali, M. Fazel, Y. C. Eldar, and B. Hassibi. Simultaneously structured models with application to sparse and low-rank matrices. *IEEE Transactions on Information Theory*, 61(5): 2886–2908, 2015.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(471), 2010.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

B. Rossman. *Average-case complexity of detecting cliques*. PhD thesis, Massachusetts Institute of Technology, 2010.

A. A. Shabalin, V. J. Weigman, C. M. Perou, and A. B Nobel. Finding large average submatrices in high dimensional data. *The Annals of Applied Statistics*, 3(3):985–1012, 2009.

X. Sun and A. B. Nobel. On the maximal size of large-average and ANOVA-fit submatrices in a gaussian random matrix. *Bernoulli*, 19(1):275–294, 2013.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In Yonina C. Eldar and Gitta Kutyniok, editors, *Compressed Sensing*, pages 210–268. Cambridge University Press, 2012.

N. Verzelen and E. Arias-Castro. Community detection in sparse random networks. *arXiv:1308.2955*, 2013.

R. K. Vinayak, S. Oymak, and B. Hassibi. Sharp performance bounds for graph clustering via convex optimization. In *38th International Conference on Acoustics, Speech, and Signal Processing*, 2014.

V. H. Vu. A simple SVD algorithm for finding hidden partitions. *arXiv:1404.3918*, 2014.

V. Q. Vu, J. Cho, J. Lei, and K. Rohe. Fantope projection and selection: A near-optimal convex relaxation of sparse pca. In *Advances in Neural Information Processing Systems*, pages 2670–2678, 2013.

J. Xu, R. Wu, K. Zhu, B. Hajek, R. Srikant, and L. Ying. Jointly clustering rows and columns of binary matrices: Algorithms and trade-offs. In *SIGMETRICS*, pages 29–41, 2014.

S. Yun and A. Proutiere. Community detection via random and adaptive sampling. In *Proceedings of the 27th Conference on Learning Theory*, 2014.

Journal of Machine Learning Research 17 (2016) 1-39 Submitted 9/14; Revised 1/16; Published 4/16

Non-linear Causal Inference using Gaussianity Measures

Daniel Hernández-Lobato DANIEL.HERNANDEZ@UAM.ES
Universidad Autónoma de Madrid
Calle Francisco Tomás y Valiente 11,
Madrid 28049, Spain

Pablo Morales-Mombiela PABLO.MORALES@ESTUDIANTE.UAM.ES
Quantitative Risk Research
Calle Faraday 7,
Madrid 28049, Spain

David Lopez-Paz* DLP@FB.COM
Facebook AI Research
6 rue Menars,
Paris 75002, France

Alberto Suárez ALBERTO.SUAREZ@UAM.ES
Universidad Autónoma de Madrid
Calle Francisco Tomás y Valiente 11,
Madrid 28049, Spain

Editor: Isabelle Guyon and Alexander Statnikov

Abstract

We provide theoretical and empirical evidence for a type of asymmetry between causes and effects that is present when these are related via linear models contaminated with additive non-Gaussian noise. Assuming that the causes and the effects have the same distribution, we show that the distribution of the residuals of a linear fit in the anti-causal direction is closer to a Gaussian than the distribution of the residuals in the causal direction. This Gaussianization effect is characterized by reduction of the magnitude of the high-order cumulants and by an increment of the differential entropy of the residuals. The problem of non-linear causal inference is addressed by performing an embedding in an expanded feature space, in which the relation between causes and effects can be assumed to be linear. The effectiveness of a method to discriminate between causes and effects based on this type of asymmetry is illustrated in a variety of experiments using different measures of Gaussianity. The proposed method is shown to be competitive with state-of-the-art techniques for causal inference.

Keywords: causal inference, Gaussianity of the residuals, cause-effect pairs

1. Introduction

The inference of causal relationships from data is one of the current areas of interest in the artificial intelligence community, *e.g.* (Chen et al., 2014; Janzing et al., 2012; Morales-Mombiela et al., 2013). The reason for this surge of interest is that discovering the causal structure of a complex system provides an explicit description of the mechanisms that generate the data, and allows us to understand the consequences of interventions in the system (Pearl, 2000). More precisely, automatic causal inference can be used to determine how modifications of the value of certain relevant variables (the causes) influence the values of other related variables (the effects). Therefore, understanding cause-effect relations is of paramount importance to control the behavior of complex systems and has applications in industrial processes, medicine, genetics, economics, social sciences or meteorology.

Causal relations can be determined in complex systems in three different ways. First, they can be inferred from domain knowledge provided by an expert, and incorporated in an ad-hoc manner in the description of the system. Second, they can be discovered by performing interventions in the system. These are controlled experiments in which one or several variables of the system are forced to take particular values. Interventions constitute a primary tool for identifying causal relationships. However, in many situations they are unethical, expensive, or technically infeasible. Third, they can be estimated using causal discovery algorithms that use as input purely uncontrolled and static data.

This last approach for causal discovery has recently received much attention from the machine learning community (Shimizu et al., 2006; Hoyer et al., 2009; Zhang and Hyvärinen, 2009). These methods assume a particular model for the *mapping mechanisms* that link causes to effects. By specifying particular conditions on the mapping mechanism and the distributions of the cause and noise variables, the causal direction becomes identifiable Chen et al. (2014). For instance, Hoyer et al. (2009) assume that the effect is a non-linear transformation of the cause plus some independent additive noise. A potential drawback of these methods is that the assumptions made by the particular model considered could be unrealistic for the data under study.

In this paper we propose a general method for causal inference that belongs to the third of the categories described above. Specifically, we assume that the cause and the effect variables have the same distribution and are linked by a linear relationship contaminated with non-Gaussian noise. For the univariate case we prove that, under these assumptions, the magnitude of the cumulants of the residuals of order higher than two is smaller for the linear fit in the anti-causal direction than in the causal one. Since the Gaussian is the only distribution whose cumulants of order higher than 2 are zero, statistical tests based on measures of Gaussianity can be used for causal inference. An antecedent of this result is the observation that, when cause and effect have the same distribution, the residuals of a fit in the anti-causal direction have higher entropy than in the causal direction (Hyvärinen and Smith, 2013; Kpotufe et al., 2014). Since the residuals of the causal and anti-causal linear models have the same variance and the Gaussian is the distribution that maximizes the entropy for a fixed variance, this means that the distribution of the latter is more Gaussian than the former.

For multivariate cause-effect pairs that have the same distribution and are related by a linear model with additive non-Gaussian noise the proof given by Hyvärinen and Smith (2013) and Kpotufe et al. (2014) can be extended to show that the entropy of the vector of residuals of a linear fit in the anti-causal direction is larger than the corresponding residuals of a linear fit in the causal direction. We conjecture that also in this case there is a reduction of the magnitude of the tensor cumulants of the anti-causal multivariate residuals and provide some numerical evidence of this effect in two dimensions.

∗. The vast majority of the work was done while being at the Max Planck Institute for Intelligent Systems and at Cambridge University.

The problem of non-linear causal inference is addressed by embedding the original problem in an expanded feature space. We then make the assumption that the non-linear relation between causes and effects in the original space is linear in the expanded feature space. The computations required to make inference on the causal direction based on this embedding can be readily carried out using kernel methods.

In summary, the proposed method for causal inference proceeds by first making a transformation of the original variables so that causes and effects have the same distribution. Then we perform kernel ridge regression in both the causal and the anti-causal directions. The dependence between causes and effects, which is non-linear in the original space, is assumed to be linear in the kernel-induced feature space. A statistical test is then used to quantify the degree of similarity between the distributions of these residuals and a Gaussian distribution with the same variance. Finally, the direction in which the residuals are less Gaussian is identified as the causal one.

The performance of this method is evaluated in both synthetic and real-world cause-effect pairs. From the results obtained it is apparent that the anti-causal residuals of a linear fit in the expanded feature space are more Gaussian than the causal residuals. In general, it is difficult to estimate the entropy from a finite sample (Beirlant et al., 1997). Empirical estimators of high order cumulants involve high order moments, which means they often have large variance. As an alternative, we propose to use statistical tests based on the *energy distance* to characterize the Gaussianization effect for the residuals of linear fits in the causal and anti-causal directions. Tests based on the energy distance were analyzed in depth by Székely and Rizzo (2005). They have been shown to be related to homogeneity tests based on embeddings in a Reproducing Kernel Hilbert Space (Gretton et al., 2012). An advantage of energy distance-based statistics is that they can be readily estimated from a sample by computing expectations of pairwise Euclidean distances. The energy distance generally provides better results than the entropy or cumulant-based Gaussianity measures. In the problems investigated, the accuracy of the proposed method, using the energy distance to the Gaussian, is comparable to other state-of-the-art techniques for causal discovery.

The rest of the paper is organized as follows: Section 2 illustrates that, under certain conditions, the residuals of a linear regression fit are closer to a Gaussian in the anti-causal direction than in the causal one, based on a reduction of the high-order cumulants and on an increment of the entropy. This section considers both the univariate and multivariate cases. Section 3 adopts a kernel approach to carry out a feature expansion that can be used to detect non-linear causal relationships. We also show here how to compute the residuals in the expanded feature space, and how to choose the different hyper-parameters of the proposed method. Section 4 contains a detailed description of the implementation. In section 6 we present the results of an empirical assessment of the proposed method in both synthetic and real-world cause-effect data pairs. Finally, Section 7 summarizes the conclusions and puts forth some ideas for future research.

2. Asymmetry Based on the Non-Gaussianity of the Residuals of Linear Models

Let $\mathcal{X}$ and $\mathcal{Y}$ be two random variables that are causally related. The direction of the causal relation is not known. Our goal is to determine whether $\mathcal{X}$ causes $\mathcal{Y}$, *i.e.*, $\mathcal{X} \to \mathcal{Y}$ or,

alternatively $\mathcal{Y}$ causes $\mathcal{X}$, *i.e.*, $\mathcal{Y} \to \mathcal{X}$. For this purpose, we exploit an asymmetry between causes an effects. This type of asymmetry can be uncovered using statistical tests that measure the non-Gaussianity of the residuals of linear regression models obtained from fits in the causal and in the anti-causal direction.

To motivate the methodology that we have developed, we will proceed in stepwise manner. First we analyze a special case in one dimension: We assume that $\mathcal{X}$ and $\mathcal{Y}$ have the same distribution and are related via a linear model contaminated with additive i.i.d. non-Gaussian noise. The noise is independent of the cause. Under these assumptions we show that the distribution of the residuals of a linear fit in the incorrect (anti-causal) direction is closer to a Gaussian distribution than the distribution of the residuals in the correct (causal) direction. For this, we use an argument based on the reduction of the magnitude of the cumulants of order higher than 2. The cumulants are defined as the derivatives of the logarithm of the moment-generating function evaluated at zero (Cornish and Fisher, 1938; McCullagh, 1987).

The Gaussianization effect can be characterized also in terms of an increase of the entropy. The proof is based on the results of Hyvärinen and Smith (2013), which are extended in this paper to the multivariate case. In particular, we show that the entropy of the residuals of a linear fit in the anti-causal direction is larger or equal than the entropy of the residuals of a linear fit in the causal direction. Since the Gaussian it the distribution that has maximum entropy, given a particular covariance matrix, an increase of the entropy of the residuals means that their distribution becomes closer to the Gaussian.

Finally, we note that it is easy to guarantee that $\mathcal{X}$ and $\mathcal{Y}$ have the same distribution in the case that these variables are unidimensional and continuous. To this end we only have to transform one of the variables (typically the cause random variable) using the probability integral transform, as described in Section 4. However, after the data have been transformed, the relation between the variables will no longer be linear in general. Thus, to address non-linear cause-effect problems involving univariate random variables the linear model is formulated in an expanded feature space, where the multivariate analysis of the Gaussianization effect is also applicable. In this feature space all the computations required for causal inference can be formulated in terms of kernels. This can be used to detect non-linear causal relations in the original input space and allows for an efficient implementation of the method. The only assumption is that the non-linear relation in the original input space is linear in the expanded feature space induced by the selected kernel.

2.1 Analysis of the Univariate Case Based on Cumulants

Let $\mathcal{X}$ and $\mathcal{Y}$ be one-dimensional random variables that have the same distribution. Without further loss of generality, we will assume that they have zero mean and unit variance. Let $\mathbf{x} = (x_1, \ldots, x_N)^{\mathrm{T}}$ and $\mathbf{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ be N paired samples drawn i.i.d. from $P(\mathcal{X}, \mathcal{Y})$. Assume that the causal direction is $\mathcal{X} \to \mathcal{Y}$ and that the measurements are related by a linear model

$$y_i = w x_i + \epsilon_i \,, \quad \epsilon_i \perp x_i \,, \quad \forall i \,, \tag{1}$$

where $w = \mathrm{corr}(\mathcal{X}, \mathcal{Y}) \in [-1, 1]$ and ϵ_i is independent i.d. non-Gaussian additive noise.

A linear model in the opposite direction, *i.e.*, $\mathcal{Y} \to \mathcal{X}$, can be built using least squares

$$x_i = w y_i + \tilde{\epsilon}_i, \tag{2}$$

where $w = \text{corr}(\mathcal{Y}, \mathcal{X})$ is the same coefficient as in the previous model. The residuals of this reversed linear model are defined as $\tilde{\epsilon}_i = x_i - w y_i$.

Following an argument similar to that of Hernández-Lobato et al. (2011) we show that the residuals $\{\tilde{\epsilon}_i\}_{i=1}^N$ in the anti-causal direction are more Gaussian than the residuals $\{\epsilon_i\}_{i=1}^N$ in the actual causal direction $\mathcal{X} \to \mathcal{Y}$ based on a reduction of the magnitude of the cumulants. The proof is based on establishing a relation between the cumulants of the distribution of the residuals in both the causal and the anti-causal direction. First, we show that $\kappa_n(y_i)$, the n-th order cumulant of $\mathcal{Y}$, can be expressed in terms of $\kappa_n(\epsilon_i)$, the n-th order cumulant of the residuals:

$$\kappa_n(y_i) = w^n \kappa_n(x_i) + \kappa_n(\epsilon_i) = w^n \kappa_n(y_i) + \kappa_n(\epsilon_i) = \frac{1}{1 - w^n} \kappa_n(\epsilon_i)\,. \tag{3}$$

To derive this relation we have used (1), that x_i and y_i have the same distribution (and hence have the same cumulants), and standard properties of cumulants (Cornish and Fisher, 1938; McCullagh, 1987). Furthermore,

$$\begin{aligned}
\kappa_n(\tilde{\epsilon}_i) &= \kappa_n(x_i - w y_i) = \kappa_n(x_i - w^2 x_i - w\epsilon_i) = (1 - w^2)^n \kappa_n(x_i) + (-w)^n \kappa_n(\epsilon_i) \\
&= (1 - w^2)^n \kappa_n(y_i) + (-w)^n \kappa_n(\epsilon_i) = \frac{(1 - w^2)^n}{1 - w^n} \kappa_n(\epsilon_i) + (-w)^n \kappa_n(\epsilon_i) \\
&= c_n(w) \kappa_n(\epsilon_i)\,,
\end{aligned} \tag{4}$$

where we have used the definition of $\tilde{\epsilon}_i$ and (3) In Figure 1 the value of

$$c_n(w) = \frac{(1 - w^2)^n}{1 - w^n} + (-w)^n. \tag{5}$$

is displayed as a function of $w \in [-1, 1]$. Note that $c_1(w) = c_2(w) = 1$ independently of the value of w. This means that the mean and the variance of the residuals are the same in both the causal and anti-causal directions. For $n > 2$, $|c_n(w)| \leq 1$ with equality only for $w = 0$ and $w = \pm 1$. The result is that the high-order cumulants of the residuals in the anti-causal direction are smaller in magnitude that the corresponding cumulants in the causal direction. Using the observation that all the cumulants of the Gaussian distribution of order higher than two are zero (Marcinkiewicz, 1938), we conclude that the distribution of the residuals in the anti-causal direction is closer to the Gaussian distribution than in the causal direction.

In summary, we can infer the causal direction by (i) fitting a linear model in each possible direction, *i.e.*, $\mathcal{X} \to \mathcal{Y}$ and $\mathcal{Y} \to \mathcal{X}$, and (ii) carrying out statistical tests to detect the level of Gaussianity of the two corresponding residuals. The direction in which the residuals are less Gaussian is expected to be the correct one.

2.2 Analysis of the Multivariate Case Based on Cumulants

In this section we argue that the Gaussianization effect of the residuals in the anti-causal direction also takes place when the two random variables $\mathcal{X}$ and $\mathcal{Y}$ are multidimensional.

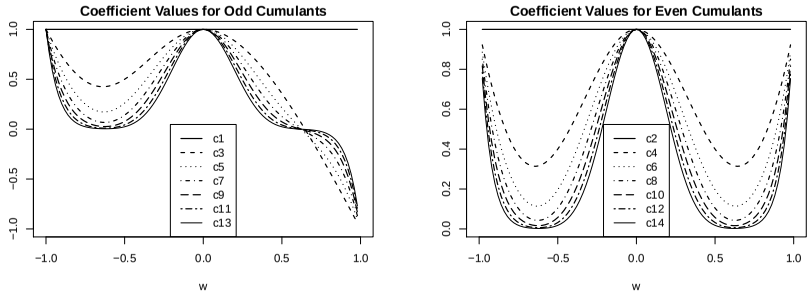


Figure 1: Values of the function $c_n(\cdot)$ as a function of w for each cumulant number n (odd in the left plot, even in the right plot). All values for $c_n(\cdot)$ lie in the interval $[-1, 1]$.

We will assume that these variables follow the same distribution and, without further loss of generality, that they have been whitened (*i.e.*, they have a zero mean vector and the identity matrix as the covariance matrix). Let $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^{\text{T}}$ and $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_N)^{\text{T}}$ be N paired samples drawn i.i.d. from $P(\mathcal{X}, \mathcal{Y})$. In this case, the model assumed for the actual causal relation is

$$\mathbf{y}_i = \mathbf{A}\mathbf{x}_i + \boldsymbol{\epsilon}_i\,, \quad \boldsymbol{\epsilon}_i \perp \mathbf{x}_i\,, \quad \forall i\,, \tag{6}$$

where $\mathbf{A} = \text{corr}(\mathcal{Y}, \mathcal{X})$ is a $d \times d$ matrix of model coefficients and $\boldsymbol{\epsilon}_i$ is i.i.d. non-Gaussian additive noise. The model in the anti-causal direction is the one that results from the least squares fit:

$$\mathbf{x}_i = \tilde{\mathbf{A}}\mathbf{y}_i + \tilde{\boldsymbol{\epsilon}}_i\,, \tag{7}$$

where we have defined $\tilde{\boldsymbol{\epsilon}}_i = \mathbf{x}_i - \tilde{\mathbf{A}}\mathbf{y}_i$ and $\tilde{\mathbf{A}} = \text{corr}(\mathcal{X}, \mathcal{Y}) = \mathbf{A}^{\text{T}}$.

As in the univariate case, we start by expressing the cumulants of $\mathcal{Y}$ in terms of the cumulants of the residuals. However, the cumulants are now tensors (McCullagh, 1987):

$$\kappa_n(\mathbf{y}_i) = \kappa_n(\mathbf{A}\mathbf{x}_i) + \kappa_n(\boldsymbol{\epsilon}_i)\,. \tag{8}$$

In what follows, the notation $\text{vect}(\cdot)$ stands for the vectorization of a tensor. For example, in the case of a tensor $\mathbf{T}$ with dimensions $d \times d \times d$

$$\text{vect}(\mathbf{T}) = (T_{1,1,1}, T_{2,1,1}, \cdots, T_{d,1,1}, T_{1,2,1}, \cdots, T_{d,d,d})^{\text{T}}.$$

Using this notation we obtain

$$\text{vect}(\kappa_n(\mathbf{y}_i)) = \mathbf{A}^n \text{vect}(\kappa_n(\mathbf{x}_i)) + \text{vect}(\kappa_n(\boldsymbol{\epsilon})) = (\mathbf{I} - \mathbf{A}^n)^{-1} \text{vect}(\kappa_n(\boldsymbol{\epsilon}))\,, \tag{9}$$

where $\mathbf{A}^n = \mathbf{A} \otimes \mathbf{A} \otimes \mathbf{A} \cdots \otimes \mathbf{A}$, n times, is computed using the Kronecker matrix product. To derive this expression we have used (6), the fact that $\mathcal{Y}$ and $\mathcal{X}$ are equally distributed and hence have the same cumulants. We also have used the properties of the tensor cumulants $\text{vect}(\kappa_n(\mathbf{A}\mathbf{x}_i)) = \mathbf{A}^n \text{vect}(\kappa_n(\mathbf{x}_i))$, where the powers of the matrix $\mathbf{A}$ are computed using the Kronecker product (McCullagh, 1987).

Similarly, for the reversed linear model

$$\begin{aligned}\kappa_n(\tilde{\boldsymbol{\epsilon}}_i) &= \kappa_n(\mathbf{x}_i - \mathbf{A}^{\mathrm{T}}\mathbf{y}_i) = \kappa_n(\mathbf{x}_i - \mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}_i - \mathbf{A}^{\mathrm{T}}\boldsymbol{\epsilon}_i) = \kappa_n((\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x}_i - \mathbf{A}^{\mathrm{T}}\boldsymbol{\epsilon}_i) \\ &= \kappa_n((\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x}_i) + \kappa_n(-\mathbf{A}^{\mathrm{T}}\boldsymbol{\epsilon}_i)\,. \end{aligned} \tag{10}$$

Using again the notation for the vectorized tensor cumulants

$$\begin{aligned}\text{vect}(\kappa_n(\tilde{\boldsymbol{\epsilon}}_i)) &= (\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{A})^n \text{vect}(\kappa_n(\mathbf{x}_i)) + (-1)^n (\mathbf{A}^{\mathrm{T}})^n \text{vect}(\kappa_n(\boldsymbol{\epsilon}_i)) \\ &= (\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{A})^n (\mathbf{I} - \mathbf{A}^n)^{-1} \text{vect}(\kappa_n(\boldsymbol{\epsilon}_i)) + (-1)^n (\mathbf{A}^{\mathrm{T}})^n \text{vect}(\kappa_n(\boldsymbol{\epsilon}_i)) \\ &= \left((\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{A})^n (\mathbf{I} - \mathbf{A}^n)^{-1} + (-1)^n (\mathbf{A}^{\mathrm{T}})^n\right) \text{vect}(\kappa_n(\boldsymbol{\epsilon}_i))\,, \end{aligned} \tag{11}$$

where the powers of matrices are computed using the Kronecker product as well, and where we have used (9) and that $\mathcal{Y}$ and $\mathcal{X}$ are equally distributed and have the same cumulants.

We now give some evidence to support that the magnitude of $\text{vect}(\kappa_n(\tilde{\boldsymbol{\epsilon}}_i))$ is smaller than the magnitude of $\text{vect}(\kappa_n(\boldsymbol{\epsilon}_i))$ in terms of the ℓ_2-norm, for cumulants of order higher than 2. That is, the tensors corresponding to high-order cumulants become closer to a tensor with all its components equal to zero. For this, we introduce the following definition:

Definition 1 *The operator norm of a matrix* $\mathbf{M}$ *induced by the* ℓ_p *vector norm is* $||\mathbf{M}||_{op} = min\{c \geq 0 : ||\mathbf{M}\mathbf{v}||_p \leq c||\mathbf{v}||_p, \forall \mathbf{v}\}$*, where* $||\cdot||_p$ *denotes the* ℓ_p*-norm for vectors.*

The consequence is that $||\mathbf{M}||_{\text{op}} \geq ||\mathbf{M}\mathbf{v}||_p / ||\mathbf{v}||_p$, $\forall \mathbf{v}$. This means that $||\mathbf{M}||_p$ can be understood as a measure of the size of the matrix $\mathbf{M}$. In the case of the ℓ_2-norm, the operator norm of a matrix $\mathbf{M}$ is equal to its largest singular value or, equivalently, to the square root of the largest eigenvalue of $\mathbf{M}^{\mathrm{T}}\mathbf{M}$. Let $\mathbf{M}_n = (\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{A})^n (\mathbf{I} - \mathbf{A}^n)^{-1} + (-1)^n (\mathbf{A}^{\mathrm{T}})^n$. That is, $\mathbf{M}_n$ is the matrix that relates the cumulants of order n of the residuals in the causal and anti-causal directions in (11). We now evaluate $||\mathbf{M}_n||_{\text{op}}$, and show that in most cases its value is smaller than one for high-order cumulants $\kappa_n(\cdot)$, leading to a Gaussianization of the residuals in the anti-causal direction. From (11) and the definition given above, we know that $||\mathbf{M}_n||_{\text{op}} \geq ||\text{vect}(\kappa_n(\tilde{\boldsymbol{\epsilon}}_i))||_2 / ||\text{vect}(\kappa_n(\boldsymbol{\epsilon}_i))||_2$. This means that if $||\mathbf{M}_n||_{\text{op}} < 1$ the cumulants of the residuals in the incorrect causal direction are shrunk to the origin. Because the multivariate Gaussian distribution has all cumulants of order higher than two equal to zero (McCullagh, 1987), this translates into a distribution for the residuals in the anti-causal direction that is closer to the Gaussian distribution.

In the causal direction, we have that $\mathbb{E}[\mathbf{y}\mathbf{y}^T] = \mathbb{E}[(\mathbf{A}\mathbf{x}_i + \boldsymbol{\epsilon}_i)(\mathbf{A}\mathbf{x}_i + \boldsymbol{\epsilon}_i)^{\mathrm{T}}] = \mathbf{A}\mathbf{A}^T + \mathbf{C} = \mathbf{I}$, where $\mathbf{C}$ is the positive definite covariance matrix of the actual residuals[1]. Thus, $\mathbf{A}\mathbf{A}^T = \mathbf{I} - \mathbf{C}$ and hence the singular values of $\mathbf{A}$, denoted $\sigma_1, \ldots, \sigma_d$, satisfy $0 \leq \sigma_i = \sqrt{1 - \alpha_i} \leq 1$, where α_i is the corresponding positive eigenvalue of $\mathbf{C}$. Assume that $\mathbf{A}$ is symmetric (this

1. We assume such matrix exists and that it is positive definite.

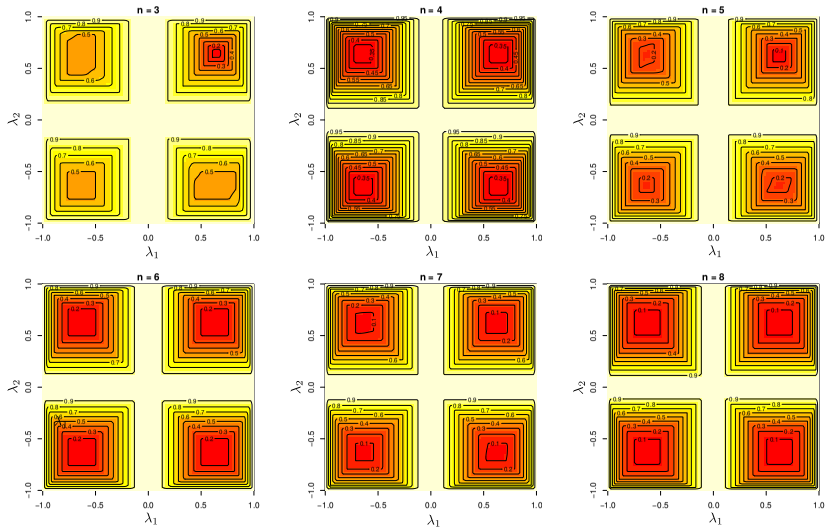


Figure 2: Contour curves of the values of $||\mathbf{M}_n||_{\text{op}}$ for $d = 2$ and for $n > 2$ as a function of λ_1 and λ_2, *i.e.*, the two eigenvalues of $\mathbf{A}$. $\mathbf{A}$ is assumed to be symmetric. Similar results are obtained for higher-order cumulants.

also means that $\mathbf{M}_n$ is symmetric). Denote by $\lambda_1, \ldots, \lambda_d$ to the eigenvalues of $\mathbf{A}$. That is, $\sigma_i = \sqrt{\lambda_i^2}$ and $0 \leq \lambda_i^2 \leq 1$, $i = 1, \ldots, d$. For a fixed cumulant of order n we have that

$$||\mathbf{M}_n||_{\text{op}} = \max_{\mathbf{v} \in \mathcal{S}} \left| \prod_{j=1}^{n} \frac{\left[1 - \lambda_{v_j}^2\right]}{1 - \prod_{i=1}^{n} \lambda_{v_i}} + (-1)^n \prod_{j=1}^{n} \lambda_{v_j} \right|, \tag{12}$$

where $\mathcal{S} = \{1, \ldots, d\}^n$, $|\cdot|$ denotes absolute value, and we have employed standard properties of the Kronecker product about eigenvalues and eigenvectors (Laub, 2004). Note that this expression does not depend on the eigenvectors of $\mathbf{A}$, but only on its eigenvalues.

Figure 2 shows, for symmetric $\mathbf{A}$, the value of $||\mathbf{M}_n||_{\text{op}}$ for $n = 3, \ldots, 8$, and $d = 2$ when the two eigenvalues of $\mathbf{A}$ range in the interval $(-1, 1)$. We observe that $||\mathbf{M}_n||_{\text{op}}$ is always smaller than one. As described before, this will lead to a reduction in the ℓ_2-norm of the cumulants in the anti-causal direction due to (11), and will in consequence produce a Gaussianization effect on the distribution of the residuals. For $n \leq 2$ it can be readily shown that $||\mathbf{M}_1||_{\text{op}} = ||\mathbf{M}_2||_{\text{op}} = 1$.

In general, the matrix $\mathbf{A}$ need not be symmetric. In this case, $||\mathbf{M}_1||_{\text{op}} = ||\mathbf{M}_2||_{\text{op}} = 1$ as well. However, the evaluation of $||\mathbf{M}_n||_{\text{op}}$ for $n > 2$ is more difficult, but feasible for small n.

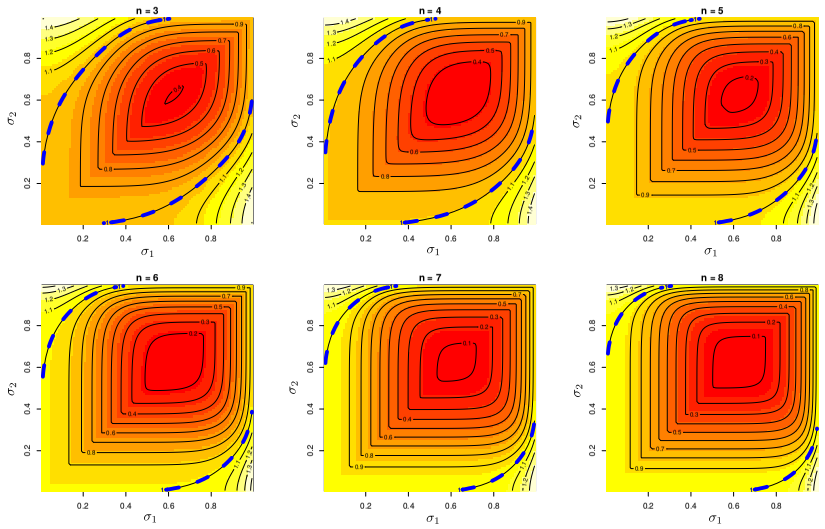


Figure 3: Contour curves of the values of $||\mathbf{M}_n||_{\text{op}}$ for $d = 2$ and for $n > 2$ as a function of σ_1 and σ_2, *i.e.*, the two singular values of $\mathbf{A}$. $\mathbf{A}$ is not assumed to be symmetric. The singular vectors of $\mathbf{A}$ are chosen at random. A dashed blue line highlights the boundary of the region where $||\mathbf{M}_n||_{\text{op}}$ is strictly smaller than one.

Figure 3 displays the values of $||\mathbf{M}_n||_{\text{op}}$, for $d = 2$, as the two singular values of $\mathbf{A}$, σ_1 and σ_2, vary in the interval $(0, 1)$. The left singular vectors and the right singular vectors of $\mathbf{A}$ are chosen at random. In this figure a dashed blue line highlights the boundary of the region where $||\mathbf{M}_n||_{\text{op}}$ is strictly smaller than one. We observe that for most values of σ_1 and σ_2, $||\mathbf{M}_n||_{\text{op}}$ is smaller than one, leading to a Gaussianization effect in the distribution of the residuals in the anti-causal direction. However, for some singular values, $||\mathbf{M}_n||_{\text{op}}$ is strictly larger than one. Of course, this does not mean that there is not such a Gaussianization effect also in those cases. The definition given for $||\mathbf{M}_n||_{\text{op}}$ assumes that all potential vectors $\mathbf{v}$ represent valid cumulants of a probability distribution, which need not be the case in practice. For example, it is well known that cumulants exhibit some form of symmetry (McCullagh, 1987). This can be seen in the second order cumulant, which is a covariance matrix. The consequence is that $||\mathbf{M}_n||_{\text{op}}$ is simply an upper bound on the reduction of the ℓ_2-norm of the cumulants in the anti-causal model. Thus, we also expect a Gaussianization effect to occur also for these cases. Furthermore, the numerical simulations presented in Section 6 provide evidence of this effect for asymmetric $\mathbf{A}$.

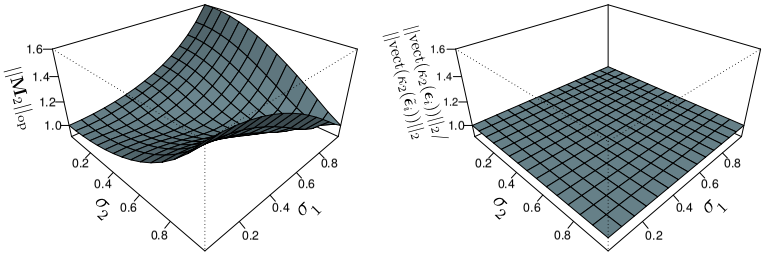


Figure 4: (left) Value of $||\mathbf{M}_2||_{\text{op}}$, for $d = 2$, as a function of σ_1 and σ_2, *i.e.*, the two singular values of $\mathbf{A}$. (right) Actual ratio between $||\text{vect}(\kappa_2(\boldsymbol{\epsilon}_i))||_2$ and $||\text{vect}(\kappa_2(\tilde{\boldsymbol{\epsilon}}_i))||_2$, for $d = 2$, as a function of σ_1 and σ_2. $\mathbf{A}$ is not assumed to be symmetric. The singular vectors of $\mathbf{A}$ are chosen at random.

The fact that $||\mathbf{M}_n||_{\text{op}}$ is only an upper bound is illustrated in Figure 4. This figure considers the particular case of the second cumulant $\kappa_2(\cdot)$, which can be analyzed in detail. On the left plot the value of $||\mathbf{M}_2||_{\text{op}}$ is displayed as a function of σ_1 and σ_2, the two singular values of $\mathbf{A}$. We observe that $||\mathbf{M}_2||_{\text{op}}$ takes values that are larger than one. In this case it is possible to evaluate in closed form the ℓ_2-norm of $\text{vect}(\kappa_2(\boldsymbol{\epsilon}_i))$ and $\text{vect}(\kappa_2(\tilde{\boldsymbol{\epsilon}}_i))$, *i.e.*, the vectors that contain the second order cumulant of the residuals in each direction. In particular, it is well known that the second order cumulant is equal to the covariance matrix (McCullagh, 1987). In the causal direction, the covariance matrix of the residuals is $\mathbf{C} = \mathbf{I} - \mathbf{A}\mathbf{A}^{\text{T}}$, as shown in the previous paragraphs. The covariance matrix of the residuals in the anti-causal direction, denoted by $\tilde{\mathbf{C}}$, can be computed similarly. Namely, $\tilde{\mathbf{C}} = \mathbf{I} - \mathbf{A}^{\text{T}}\mathbf{A}$. These two matrices, *i.e.*, $\mathbf{C}$ and $\tilde{\mathbf{C}}$, respectively give $k_2(\boldsymbol{\epsilon}_i)$ and $k_2(\tilde{\boldsymbol{\epsilon}}_i)$. Furthermore, they have the same singular values. This means that $||\text{vect}(k_2(\boldsymbol{\epsilon}_i))||_2 / ||\text{vect}(k_2(\tilde{\boldsymbol{\epsilon}}_i))||_2 = 1$, as illustrated by the right plot in Figure 4. Thus, $||\mathbf{M}_2||_{\text{op}}$ is simply an upper bound on the actual reduction of the ℓ_2-norm of the second order cumulant of the residuals in the anti-causal direction. The same behavior is expected for $||\mathbf{M}_n||_{\text{op}}$, with $n > 2$. In consequence, one should expect that the cumulants of the distribution of the residuals of a model fitted in the anti-causal direction are smaller in magnitude. This will lead to an increased level of Gaussianity measured in terms of a reduction of the magnitude of the high-order cumulants.

2.3 Analysis of the Multivariate Case Based on Information Theory

The analysis of the multivariate case carried out in the previous section is illustrative. Nevertheless, it does not prove that the distribution of the residuals in the anti-casual direction is more Gaussian based on a reduction of the magnitude of the cumulants. Further evidence of this increased level of Gaussianity can be obtained based on an increase of the entropy obtained by using information theory. In this case we also assume that the multi-dimensional variables $\mathcal{X}$ and $\mathcal{Y}$ follow the same distribution, but unlike in the previous

section, they need not be whitened, only centered. Here we closely follow Section 2.4 of the work by Hyvärinen and Smith (2013) and extend their results to the multivariate case.

Under the assumptions specified earlier, the model in the causal direction is $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i + \boldsymbol{\epsilon}_i$, with $\mathbf{x}_i \perp \boldsymbol{\epsilon}_i$ and $\mathbf{A} = \text{Cov}(\mathcal{Y},\mathcal{X})\text{Cov}(\mathcal{X},\mathcal{X})^{-1}$. Similarly, the model in the anti-causal direction is $\mathbf{x}_i = \tilde{\mathbf{A}}\mathbf{y}_i + \tilde{\boldsymbol{\epsilon}}_i$ with $\tilde{\mathbf{A}} = \text{Cov}(\mathcal{X},\mathcal{Y})\text{Cov}(\mathcal{Y},\mathcal{Y})^{-1}$. By making use of the causal model it is possible to show that $\tilde{\boldsymbol{\epsilon}}_i = (\mathbf{I} - \tilde{\mathbf{A}}\mathbf{A})\mathbf{x}_i - \tilde{\mathbf{A}}\boldsymbol{\epsilon}_i$, where $\mathbf{I}$ is the identity matrix. Thus, the following equations are satisfied:

$$\begin{pmatrix} \mathbf{x}_i \\ \mathbf{y}_i \end{pmatrix} = \mathbf{P}\begin{pmatrix} \mathbf{x}_i \\ \boldsymbol{\epsilon}_i \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A} & \mathbf{I} \end{pmatrix}\begin{pmatrix} \mathbf{x}_i \\ \boldsymbol{\epsilon}_i \end{pmatrix}, \tag{13}$$

and

$$\begin{pmatrix} \mathbf{y}_i \\ \tilde{\boldsymbol{\epsilon}}_i \end{pmatrix} = \tilde{\mathbf{P}}\begin{pmatrix} \mathbf{x}_i \\ \boldsymbol{\epsilon}_i \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{I} \\ \mathbf{I} - \tilde{\mathbf{A}}\mathbf{A} & -\tilde{\mathbf{A}} \end{pmatrix}\begin{pmatrix} \mathbf{x}_i \\ \boldsymbol{\epsilon}_i \end{pmatrix}. \tag{14}$$

Let $H(\mathbf{x}_i, \mathbf{y}_i)$ be the entropy of the joint distribution of the two random variables associated with samples $\mathbf{x}_i$ and $\mathbf{y}_i$. Because (13) is a linear transformation, we can use the entropy transformation formula (Hyvärinen and Smith, 2013) to get that $H(\mathbf{x}_i, \mathbf{y}_i) = H(\mathbf{x}_i, \boldsymbol{\epsilon}_i) + \log|\text{det}\mathbf{P}|$, where $\text{det}\mathbf{P} = \text{det}\mathbf{I} = 1$. Thus, we have that $H(\mathbf{x}_i, \mathbf{y}_i) = H(\mathbf{x}_i, \boldsymbol{\epsilon}_i)$. Conversely, if we use (14) we have $H(\mathbf{y}_i, \boldsymbol{\epsilon}_i) = H(\mathbf{x}_i, \boldsymbol{\epsilon}_i) + \log|\text{det}\tilde{\mathbf{P}}|$, where $\text{det}\tilde{\mathbf{P}} = \text{det}\mathbf{A} \cdot \text{det}(-\tilde{\mathbf{A}} - (\mathbf{I} - \tilde{\mathbf{A}}\mathbf{A})\mathbf{A}^{-1}\mathbf{I}) = -\text{det}\mathbf{I} = -1$, under the assumption that $\mathbf{A}$ is invertible. The result is that $H(\mathbf{x}_i, \mathbf{y}_i) = H(\mathbf{y}_i, \boldsymbol{\epsilon}_i) = H(\mathbf{x}_i, \boldsymbol{\epsilon}_i)$.

Denote the mutual information between the cause and the noise with $I(\mathbf{x}_i, \boldsymbol{\epsilon}_i)$. Similarly, let $I(\mathbf{y}_i, \tilde{\boldsymbol{\epsilon}}_i)$ be the mutual information between the random variables corresponding to the observations $\mathbf{y}_i$ and $\tilde{\boldsymbol{\epsilon}}_i$. Then,

$$\begin{aligned} I(\mathbf{x}_i, \boldsymbol{\epsilon}_i) - I(\mathbf{y}_i, \tilde{\boldsymbol{\epsilon}}_i) &= H(\mathbf{x}_i) + H(\boldsymbol{\epsilon}_i) - H(\mathbf{x}_i, \boldsymbol{\epsilon}_i) - H(\mathbf{y}_i) - H(\tilde{\boldsymbol{\epsilon}}_i) + H(\mathbf{y}_i, \tilde{\boldsymbol{\epsilon}}_i) \\ &= H(\mathbf{x}_i) + H(\boldsymbol{\epsilon}_i) - H(\mathbf{y}_i) - H(\tilde{\boldsymbol{\epsilon}}_i)\,. \end{aligned} \tag{15}$$

Furthermore, from the actual causal model assumed we know that $I(\mathbf{x}_i, \boldsymbol{\epsilon}_i) = 0$. By contrast, we have that $I(\mathbf{y}_i, \tilde{\boldsymbol{\epsilon}}_i) \geq 0$, since both $\mathbf{y}_i$ and $\tilde{\boldsymbol{\epsilon}}_i$ depend on $\mathbf{x}_i$ and $\boldsymbol{\epsilon}_i$. We also know that $H(\mathbf{x}_i) = H(\mathbf{y}_i)$ because we have made the hypothesis that both $\mathcal{X}$ and $\mathcal{Y}$ follow the same distribution. The result is that:

$$H(\boldsymbol{\epsilon}_i) \leq H(\tilde{\boldsymbol{\epsilon}}_i)\,, \tag{16}$$

with equality iff the residuals are Gaussian. We note that an alternative but equivalent way to obtain this last result is to consider a multivariate version of Lemma 1 by Kpotufe et al. (2014), under the assumption of the same distribution for the cause and the effect. In particular, even though Kpotufe et al. (2014) assume univariate random variables, their work can be easily generalized to multiple variables.

Although the random variables corresponding to $\boldsymbol{\epsilon}_i$ and $\tilde{\boldsymbol{\epsilon}}_i$ have both zero mean, they need not have the same covariance matrix. Denote with $\text{Cov}(\boldsymbol{\epsilon}_i)$ and $\text{Cov}(\tilde{\boldsymbol{\epsilon}}_i)$ to these matrices and let $\hat{\boldsymbol{\epsilon}}_i$ and $\hat{\tilde{\boldsymbol{\epsilon}}}_i$ be the whitened residuals (*i.e.*, the residuals multiplied by the Cholesky factor of the inverse of the corresponding covariance matrix). Then,

$$H(\hat{\boldsymbol{\epsilon}}_i) \leq H(\hat{\tilde{\boldsymbol{\epsilon}}}_i) - \frac{1}{2}\log|\text{detCov}(\tilde{\boldsymbol{\epsilon}}_i)| + \frac{1}{2}\log|\text{detCov}(\boldsymbol{\epsilon}_i)|\,. \tag{17}$$

As shown in Appendix A, although not equal, the matrices $\text{Cov}(\boldsymbol{\epsilon}_i)$ and $\text{Cov}(\tilde{\boldsymbol{\epsilon}}_i)$ have the same determinant. Thus, the two determinants cancel in the equation above. This gives,

$$H(\hat{\boldsymbol{\epsilon}}_i) \leq H(\hat{\tilde{\boldsymbol{\epsilon}}}_i)\,. \tag{18}$$

The consequence is that the entropy of the whitened residuals in the anti-causal direction is expected to be higher or equal than the entropy of the whitened residuals in the causal direction. Because the Gaussian distribution is the continuous distribution with the highest entropy for a fixed covariance matrix, we conclude that the level of Gaussianity of the residuals in the anti-causal direction, measured in terms of differential entropy, has to be larger or equal to the level of Gaussianity of the residuals in the causal direction.

In summary, if the causal relation between the two identically distributed random variables $\mathcal{X}$ and $\mathcal{Y}$ is linear the residuals of a least squares fit in the anti-causal direction are more Gaussian than those of a linear fit in the causal direction. This Gaussianization can be characterized by a reduction of the magnitude of the corresponding high-order cumulants (although we have not formally proved this, we have provided some evidence that this is the case) and by an increase of the entropy (we have proved this in this section). A causal inference method can take advantage of this asymmetry to determine the causal direction. In particular, statistical tests based on measures of Gaussianity can be used for this purpose. However and importantly, these measures of Gaussianity need not be estimates of the differential entropy or the high-order cumulants. In particular, the entropy is a quantity that is particularly difficult to estimate in practice (Beirlant et al., 1997). The same occurs with the high-order cumulants. Their estimators involve high-order moments, and hence suffer from high variance.

When the distribution of the residuals in (6) is Gaussian, the causal direction cannot be identified. In this case, it is possible to show that the distribution of the reversed residuals $\tilde{\boldsymbol{\epsilon}}_i$, the cause $\mathbf{x}_i$, and the effect $\mathbf{y}_i$, is Gaussian as a consequence of (9) and (11). This non-identifiability agrees with the general result of Shimizu et al. (2006), which indicates that non-Gaussian distributions are strictly required in the disturbance variables to carry out causal inference in linear models with additive independent noise.

Finally, the fact that the Gaussianization effect is also expected in the multivariate case suggests a method to address causal inference problems in which the relationship between cause and effect is non-linear: It consists in mapping each observation x_i and y_i to a vector in an expanded feature space. One can them assume that the non-linear relation in the original input space is linear in the expanded feature space and compute the residuals of kernel ridge regressions in both directions. The direction in which the residuals are less Gaussian is identified as the causal one.

3. A Feature Expansion to Address Non-linear Causal Inference Problems

We now proceed to relax the assumption that the causal relationship between the unidimensional random variables $\mathcal{X}$ and $\mathcal{Y}$ is linear. For this purpose, instead of working in the original space in which the samples $\{(x_i, y_i)\}_{i=1}^N$ are observed, we will assume that the model is linear in some expanded feature space $\{(\phi(x_i), \phi(y_i))\}_{i=1}^N$ for some mapping function $\phi(\cdot) : \mathbb{R} \to \mathbb{R}^d$. Importantly, this map preserves the property that if x_i and y_i are

equally distributed, so will be $\phi(x_i)$ and $\phi(y_i)$. According to the analysis presented in the previous section, the residuals of a linear model in the expanded space should be more Gaussian in the anti-causal direction than the residuals of a linear model in the causal direction, based on an increment of the differential entropy and on a reduction of the magnitude of the cumulants. The assumption we make is that the non-linear relation between $\mathcal{X}$ and $\mathcal{Y}$ in the original input space is linear in the expanded feature space.

In this section we focus on obtaining the normalized residuals of linear models formulated in the expanded feature space. For this purpose, we assume that a kernel function $k(\cdot,\cdot)$ can be used to evaluate dot products in the expanded feature space. In particular, $k(x_i,x_j) = \phi(x_i)^{\mathrm{T}}\phi(x_j)$ and $k(y_i,y_j) = \phi(y_i)^{\mathrm{T}}\phi(y_j)$ for arbitrary x_i and x_j and y_i and y_j. Furthermore, we will not assume in general that $\phi(x_i)$ and $\phi(y_i)$ have been whitened, only centered. Whitening is a linear transformation which is not expected to affect to the level of Gaussianity of the residuals. However, once these residuals have been obtained they will be whitened in the expanded feature space. Later on we describe how to center the data in the expanded feature space. For now on, we will assume this step has already been done.

3.1 Non-linear Model Description and Fitting Process

Assume that the relation between $\mathcal{X}$ and $\mathcal{Y}$ is linear in an expanded feature space

$$\phi(y_i) = \mathbf{A}\phi(x_i) + \boldsymbol{\epsilon}_i\,, \quad \boldsymbol{\epsilon}_i \perp x_i, \tag{19}$$

where $\boldsymbol{\epsilon}_i$ is i.i.d. non-Gaussian additive noise.

Given N paired observations $\{(x_i,y_i)\}_{i=1}^N$ drawn i.i.d. from $P(\mathcal{X},\mathcal{Y})$, define the matrices $\boldsymbol{\Phi}_x = (\phi(x_1),\ldots,\phi(x_N))$ and $\boldsymbol{\Phi}_y = (\phi(y_1),\ldots,\phi(y_N))$ of size $d \times N$. The estimate of $\mathbf{A}$ that minimizes the sum of squared errors is $\hat{\mathbf{A}} = \boldsymbol{\Gamma}\boldsymbol{\Sigma}^{-1}$, where $\boldsymbol{\Gamma} = \boldsymbol{\Phi}_y\boldsymbol{\Phi}_x^{\mathrm{T}}$ and $\boldsymbol{\Sigma} = \boldsymbol{\Phi}_x\boldsymbol{\Phi}_x^{\mathrm{T}}$. Unfortunately, when $d > N$, where d is the number of variables in the feature expansion, the matrix $\boldsymbol{\Sigma}^{-1}$ does not exist and $\hat{\mathbf{A}}$ is not unique. This means that there is an infinite number of solutions for $\hat{\mathbf{A}}$ with zero squared error.

To avoid the indetermination described above and also to alleviate over-fitting, we propose a regularized estimator. Namely,

$$\mathcal{L}(\mathbf{A}) = \sum_{i=1}^{N} \frac{1}{2}||\phi(y_i) - \mathbf{A}\phi(x_i)||_2^2 + \tau\frac{1}{2}||\mathbf{A}||_{\mathcal{F}}^2\,, \tag{20}$$

where $||\cdot||_2$ denotes the ℓ_2-norm and $||\cdot||_{\mathcal{F}}$ denotes the Frobenius norm. In this last expression $\tau > 0$ is a parameter that controls the amount of regularization. The minimizer of (20) is $\hat{\mathbf{A}} = \boldsymbol{\Gamma}\boldsymbol{\Sigma}^{-1}$, where $\boldsymbol{\Gamma} = \boldsymbol{\Phi}_y\boldsymbol{\Phi}_x^{\mathrm{T}}$ and $\boldsymbol{\Sigma} = \tau\mathbf{I} + \boldsymbol{\Phi}_x\boldsymbol{\Phi}_x^{\mathrm{T}}$. The larger the value of τ, the closer the entries of $\hat{\mathbf{A}}$ are to zero. Furthermore, using the matrix inversion lemma we have that $\boldsymbol{\Sigma}^{-1} = \tau^{-1}\mathbf{I} - \tau^{-1}\boldsymbol{\Phi}_x\mathbf{V}^{-1}\boldsymbol{\Phi}_x^{\mathrm{T}}$, where $\mathbf{V} = (\tau\mathbf{I} + \mathbf{K}_{x,x})^{-1}$ and $\mathbf{K}_{x,x} = \boldsymbol{\Phi}_x^{\mathrm{T}}\boldsymbol{\Phi}_x$ is a kernel matrix whose entries are given by $k(x_i,x_j)$. After some algebra it is possible to show that

$$\hat{\mathbf{A}} = \boldsymbol{\Gamma}\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Phi}_y\mathbf{V}\boldsymbol{\Phi}_x^{\mathrm{T}}\,, \tag{21}$$

which depends only on the matrix $\mathbf{V}$. This matrix be computed with cost $\mathcal{O}(N^3)$. We note that the estimate obtained in (21) coincides with the kernel conditional embedding operator described by Song et al. (2013) for mapping conditional distributions into infinite dimensional feature spaces using kernels.

3.2 Obtaining the Matrix of Inner Products of the Residuals

A first step towards obtaining the whitened residuals in feature space (which will be required for the estimation of their level of Gaussianity) is to compute the matrix of inner products of these residuals (kernel matrix). For this, we define $\boldsymbol{\epsilon}_i = \phi(y_i) - \hat{\mathbf{A}}\phi(x_i)$. Thus,

$$\begin{aligned}\boldsymbol{\epsilon}_i^{\mathrm{T}}\boldsymbol{\epsilon}_j &= \left[\phi(y_i) - \hat{\mathbf{A}}\phi(x_i)\right]^{\mathrm{T}}\left[\phi(y_j) - \hat{\mathbf{A}}\phi(x_j)\right] \\ &= \phi(y_i)^{\mathrm{T}}\phi(y_j) - \phi(y_i)^{\mathrm{T}}\hat{\mathbf{A}}\phi(x_j) - \phi(x_i)^{\mathrm{T}}\hat{\mathbf{A}}\phi(y_j) + \phi(x_i)^{\mathrm{T}}\hat{\mathbf{A}}\hat{\mathbf{A}}\phi(x_j)\,,\end{aligned} \tag{22}$$

for two arbitrary residuals $\boldsymbol{\epsilon}_i$ and $\boldsymbol{\epsilon}_j$ in feature space. In general, if we denote with $\mathbf{K}_\epsilon$ to the matrix whose entries are given by $\boldsymbol{\epsilon}_i^{\mathrm{T}}\boldsymbol{\epsilon}_j$ and define $\mathbf{K}_{y,y} = \boldsymbol{\Phi}_y^{\mathrm{T}}\boldsymbol{\Phi}_y$, we have that

$$\mathbf{K}_\epsilon = \mathbf{K}_{y,y} - \mathbf{K}_{y,y}\mathbf{V}\mathbf{K}_{x,x} - \mathbf{K}_{x,x}\mathbf{V}\mathbf{K}_{y,y} + \mathbf{K}_{x,x}\mathbf{V}\mathbf{K}_{y,y}\mathbf{V}\mathbf{K}_{x,x}\,, \tag{23}$$

where we have used the definition of $\hat{\mathbf{A}}$ in (21). This expression only depends on the kernel matrices $\mathbf{K}_{x,x}$ and $\mathbf{K}_{y,y}$ and the matrix $\mathbf{V}$, and can be computed with cost $\mathcal{O}(N^3)$.

3.3 Centering the Input Data and Centering and Whitening the Residuals

An assumption made in Section 2 was that the samples of the random variables $\mathcal{X}$ and $\mathcal{Y}$ are centered, *i.e.*, they have zero mean. In this section we show how to carry out this centering process in feature space. Furthermore, we also show how to center the residuals of the fitting process, which are also whitened. Whitening is a standard procedure in which the data are transformed to have the identity matrix as the covariance matrix. It also corresponds to projecting the data onto all the principal components, and scaling them to have unit standard deviation.

We show how to center the data in feature space. For this, we follow Schölkopf et al. (1997) and work with:

$$\tilde{\phi}(x_i) = \phi(x_i) - \frac{1}{N}\sum_{j=1}^{N}\phi(x_j)\,, \qquad \tilde{\phi}(y_i) = \phi(y_i) - \frac{1}{N}\sum_{j=1}^{N}\phi(y_j)\,. \tag{24}$$

The consequence is that now the kernel matrices $\mathbf{K}_{x,x}$ and $\mathbf{K}_{y,y}$ are replaced by

$$\begin{aligned}\tilde{\mathbf{K}}_{x,x} &= \mathbf{K}_{x,x} - \mathbf{1}_N\mathbf{K}_{x,x} - \mathbf{K}_{x,x}\mathbf{1}_N + \mathbf{1}_N\mathbf{K}_{x,x}\mathbf{1}_N\,, \\ \tilde{\mathbf{K}}_{y,y} &= \mathbf{K}_{y,y} - \mathbf{1}_N\mathbf{K}_{y,y} - \mathbf{K}_{y,y}\mathbf{1}_N + \mathbf{1}_N\mathbf{K}_{y,y}\mathbf{1}_N\,,\end{aligned} \tag{25}$$

where $\mathbf{1}_N$ is a $N \times N$ matrix with all entries equal to $1/N$. The residuals can be centered also in a similar way. Namely, $\tilde{\mathbf{K}}_\epsilon = \mathbf{K}_\epsilon - \mathbf{1}_N\mathbf{K}_\epsilon - \mathbf{K}_\epsilon\mathbf{1}_N + \mathbf{1}_N\mathbf{K}_\epsilon\mathbf{1}_N$.

We now explain the whitening of the residuals, which are now assumed to be centered. This process involves the computation of the eigenvalues and eigenvectors of the $d \times d$ covariance matrix $\mathbf{C}$ of the residuals. This is done as in kernel PCA (Schölkopf et al., 1997). Denote by $\tilde{\boldsymbol{\epsilon}}_i$ to the centered residuals. The covariance matrix is $\mathbf{C} = N^{-1}\sum_{i=1}^{N}\tilde{\boldsymbol{\epsilon}}_i\tilde{\boldsymbol{\epsilon}}_i^{\mathrm{T}}$. The eigenvector expansion implies that $\mathbf{C}\mathbf{v}_i = \lambda_i\mathbf{v}_i$, where $\mathbf{v}_i$ denotes the i-th eigenvector and λ_i the i-th eigenvalue. The consequence is that $N^{-1}\sum_{k=1}^{N}\tilde{\boldsymbol{\epsilon}}_k\tilde{\boldsymbol{\epsilon}}_k^{\mathrm{T}}\mathbf{v}_i = \lambda_i\mathbf{v}_i$. Thus, the eigenvectors can be expressed as a combination of the residuals. Namely, $\mathbf{v}_i = \sum_{j=1}^{N} b_{i,j}\tilde{\boldsymbol{\epsilon}}_j$,

where $b_{i,j} = N^{-1}\tilde{\boldsymbol{\epsilon}}_j^{\mathrm{T}}\mathbf{v}_i$. Substituting this result in the previous equation we have that $N^{-1}\sum_{k=1}^{N}\tilde{\boldsymbol{\epsilon}}_k\tilde{\boldsymbol{\epsilon}}_k^{\mathrm{T}}\sum_{j=1}^{N}b_{i,j}\tilde{\boldsymbol{\epsilon}}_j = \lambda_i\sum_{j=1}^{N}b_{i,j}\tilde{\boldsymbol{\epsilon}}_j$. When we multiply both sides by $\tilde{\boldsymbol{\epsilon}}_l^{\mathrm{T}}$ we obtain $N^{-1}\sum_{k=1}^{N}\tilde{\boldsymbol{\epsilon}}_l^{\mathrm{T}}\tilde{\boldsymbol{\epsilon}}_k\tilde{\boldsymbol{\epsilon}}_k^{\mathrm{T}}\sum_{j=1}^{N}b_{i,j}\tilde{\boldsymbol{\epsilon}}_j = \lambda_i\sum_{j=1}^{N}b_{i,j}\tilde{\boldsymbol{\epsilon}}_l^{\mathrm{T}}\tilde{\boldsymbol{\epsilon}}_j$, for $l = 1, \dots d$, which is written in terms of kernels as $\tilde{\mathbf{K}}_\epsilon\tilde{\mathbf{K}}_\epsilon\mathbf{b}_i = \lambda_i N\tilde{\mathbf{K}}_\epsilon\mathbf{b}_i$, where $\mathbf{b}_i = (b_{i,1}, \dots, b_{i,N})^{\mathrm{T}}$. A solution to this problem is found by solving the eigenvalue problem $\tilde{\mathbf{K}}_\epsilon\mathbf{b}_i = \lambda_i N\mathbf{b}_i$. We also require that the eigenvectors have unit norm. Thus, $1 = \mathbf{v}_i^{\mathrm{T}}\mathbf{v}_i = \sum_{j=1}^{N}\sum_{k=1}^{N}b_{i,j}b_{i,k}\tilde{\boldsymbol{\epsilon}}_j^{\mathrm{T}}\tilde{\boldsymbol{\epsilon}}_k = \mathbf{b}_i^{\mathrm{T}}\tilde{\mathbf{K}}_\epsilon\mathbf{b}_i = \lambda_i N\mathbf{b}_i^{\mathrm{T}}\mathbf{b}_i$, which means that $\mathbf{b}_i$ has norm $1/\sqrt{\lambda_i N}$. Consider now that $\tilde{\mathbf{b}}_i$ is one eigenvector of $\tilde{\mathbf{K}}_\epsilon$. Then, $\mathbf{b}_i = 1/\sqrt{\lambda_i N}\tilde{\mathbf{b}}_i$. Similarly, let $\tilde{\lambda}_i$ be an eigenvalue of $\tilde{\mathbf{K}}_\epsilon$. Then $\lambda_i = \tilde{\lambda}_i/N$. In summary, λ_i and $b_{i,j}$, with $i = 1, \dots, N$ and $j = 1, \dots, N$ can be found with cost $\mathcal{O}(N^3)$ by finding the eigendecomposition of $\tilde{\mathbf{K}}_\epsilon$.

The whitening process is carried out by projecting each residual $\tilde{\boldsymbol{\epsilon}}_k$ onto each eigenvector $\mathbf{v}_i$ and then multiplying by $1/\sqrt{\lambda_i}$. The corresponding i-th component for the k-th residual, denoted by $Z_{k,i}$, is $Z_{k,i} = 1/\sqrt{\lambda_i}\mathbf{v}_i^{\mathrm{T}}\tilde{\boldsymbol{\epsilon}}_k = 1/\sqrt{\lambda_i}\sum_{j=1}^{N}b_{i,j}\tilde{\boldsymbol{\epsilon}}_j^{\mathrm{T}}\tilde{\boldsymbol{\epsilon}}_k$, and in consequence, the whitened residuals are $\mathbf{Z} = \tilde{\mathbf{K}}_\epsilon\mathbf{B}\mathbf{D} = N\mathbf{B}\mathbf{D}^{-1} = \sqrt{N}\tilde{\mathbf{B}}$, where $\mathbf{B}$ is a matrix whose columns contain each $\mathbf{b}_i$, $\tilde{\mathbf{B}}$ is a matrix whose columns contain each $\tilde{b}_i$ and $\mathbf{D}$ is a diagonal matrix whose entries are equal to $1/\sqrt{\lambda_i}$. Each row of $\mathbf{Z}$ now contains the whitened residuals.

3.4 Inferring the Most Likely Causal Direction

After having trained the model and obtained the matrix of whitened residuals $\mathbf{Z}$ in each direction, a suitable Gaussianity test can be used to determine the correct causal relation between the variables $\mathcal{X}$ and $\mathcal{Y}$. Given the theoretical results of Section 2 one may be tempted to use tests based on entropy or cumulants estimation. Such tests may perform poorly in practice due to the difficulty of estimating high-order cumulants or differential entropy. In particular, the estimators of the cumulants involve high-order moments and hence, suffer from high variance. As a consequence, in our experiments we use a statistical test for Gaussianity based on the *energy distance* (Székely and Rizzo, 2005), which has good power, is robust to noise, and does not have any adjustable hyper-parameters. Furthermore, in Appendix B we motivate that in the anti-causal direction one should also expect a smaller energy distance to the Gaussian distribution.

Assume $\mathcal{X}$ and $\mathcal{Y}$ are two independent random variables whose probability distribution functions are $F(\cdot)$ and $G(\cdot)$. The energy distance between these distributions is defined as

$$D^2(F, G) = 2\mathbb{E}[||\mathcal{X} - \mathcal{Y}||] - \mathbb{E}[||\mathcal{X} - \mathcal{X}'||] - \mathbb{E}[||\mathcal{Y} - \mathcal{Y}'||]\,, \tag{26}$$

where $||\cdot||$ denotes some norm, typically the ℓ_2-norm; $\mathcal{X}$ and $\mathcal{X}'$ are independent and identically distributed (i.i.d.); $\mathcal{Y}$ and $\mathcal{Y}'$ are i.i.d; and $\mathbb{E}$ denotes expected value. The energy distance satisfies all axioms of a metric and hence characterizes the equality of distributions. Namely, $D^2(F, G) = 0$ if and only if $F = G$. Furthermore, in the case of univariate random variables the energy distance is twice the Cramér-von Mises distance given by $\int (F(x) - G(x))^2\, dx$.

Assume $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N)^{\mathrm{T}}$ is a matrix that contains N random samples (one per each row of the matrix) from a d dimensional random variable with probability density f. The statistic described for testing for Gaussianity, *i.e.*, $\mathrm{H}_0 : f = \mathcal{N}(\cdot|\mathbf{0}, \mathbf{I})$ vs $\mathrm{H}_1 : f \neq \mathcal{N}(\cdot|\mathbf{0}, \mathbf{I})$,

that is described by Székely and Rizzo (2005) is:

$$\mathrm{Energy}(\mathbf{X}) = N\left(\frac{2}{N}\sum_{j=1}^{N}\mathbb{E}[||\mathbf{x}_j - \mathcal{Y}||] - \mathbb{E}[||\mathcal{Y} - \mathcal{Y}'||] - \frac{1}{N^2}\sum_{j,k=1}^{N}||\mathbf{x}_j - \mathbf{x}_k||\right), \tag{27}$$

where $\mathcal{Y}$ and $\mathcal{Y}'$ are independent random variables distributed as $\mathcal{N}(\cdot|\mathbf{0}, \mathbf{I})$ and $\mathbb{E}$ denotes expected value. Furthermore, the required expectations with respect to the Gaussian random variables $\mathcal{Y}$ and $\mathcal{Y}'$ can be efficiently computed as described by Székely and Rizzo (2005). The idea is that if f is similar to a Gaussian density $\mathcal{N}(\cdot|0, \mathbf{I})$, then $\mathrm{Energy}(\mathbf{X})$ is close to zero. Conversely, the null hypothesis H_0 is rejected for large values of $\mathrm{Energy}(\mathbf{X})$.

The data to test for Gaussianity is in our case $\mathbf{Z}$, *i.e.*, the matrix of whitened residuals, which has size $N \times N$. Thus, the whitened residuals have N dimensions. The direct introduction of these residuals into a statistical test for Gaussianity is not expected to provide meaningful results, as a consequence of the high dimensionality. Furthermore, in our experiments we have observed that it is often the case that a large part of the total variance is explained by the first principal component (see the supplementary material for evidence supporting this). That is, λ_i, *i.e.*, the eigenvalue associated to the i-th principal component, is almost negligible for $i \geq 2$. Additionally, we motivate in Appendix C that one should also obtain more Gaussian residuals, after projecting the data onto the first principal component, in terms of a reduction of the magnitude of the high-order cumulants. Thus, in practice, we consider only the first principal component of the estimated residuals in feature space. This is the component i with the largest associated eigenvalue λ_i. We denote such N-dimensional vector by $\mathbf{z}$.

Let $\mathbf{z}_{x\to y}$ be the vector of coefficients of the first principal component of the residuals in feature space when the linear fit is performed in the direction $\mathcal{X} \to \mathcal{Y}$. Let $\mathbf{z}_{y\to x}$ be the vector of coefficients of the first principal component of the residuals in feature space when the linear fit is carried out in the direction $\mathcal{Y} \to \mathcal{X}$. We define the measure of Gaussianization of the residuals as $\mathcal{G} = \mathrm{Energy}(\mathbf{z}_{x\to y})/N - \mathrm{Energy}(\mathbf{z}_{y\to x})/N$, where $\mathrm{Energy}(\cdot)$ computes the statistic of the energy distance test for Gaussianity described above. Note that we divide each statistic by N to cancel the corresponding factor that is considered in (27). Since in this test larger values for the statistic corresponds to larger deviations from Gaussianity, if $\mathcal{G} > 0$ the direction $\mathcal{X} \to \mathcal{Y}$ is expected to be more likely the causal direction. Otherwise, the direction $\mathcal{Y} \to \mathcal{X}$ is preferred.

The variance of $\mathcal{G}$ will depend on the sample size N. Thus, ideally one should use the difference between the p-values associated to each statistic as the confidence in the decision taken. Unfortunately, computing these p-values is expensive since the distribution of the statistic under the null hypothesis must be approximated via random sampling. In our experiments we measure the confidence of the decision in terms of the absolute value of $\mathcal{G}$, which is faster to obtain and we have found to perform well in practice.

3.5 Parameter Tuning and Error Evaluation

Assume that a squared exponential kernel is employed in the method described above. This means that $k(x_i, x_j) = \exp\left(-\gamma(x_i - x_j)^2\right)$, where $\gamma > 0$ is the bandwidth of the kernel. The same is assumed for $k(y_i, y_j)$. Therefore, two hyper-parameters require adjustment

in the method described. These are the ridge regression regularization parameter τ and the kernel bandwidth γ. They must be tuned in some way to produce the best possible fit in each direction. The method chosen to guarantee this is a grid search guided by a 10-fold cross-validation procedure, which requires computing the squared prediction error over unseen data. In this section we detail how to evaluate these errors.

Assume that M new paired data instances are available for validation. Let the two matrices $\mathbf{\Phi}_{y^{\text{new}}} = (\phi(y_1^{\text{new}}), \ldots, \phi(y_M^{\text{new}}))$ and $\mathbf{\Phi}_{x^{\text{new}}} = (\phi(x_1^{\text{new}}), \ldots, \phi(x_M^{\text{new}}))$ summarize these data. Define $\boldsymbol{\epsilon}_i^{\text{new}} = \phi(y_i^{\text{new}}) - \hat{\mathbf{A}}\phi(x_i^{\text{new}})$. After some algebra, it is possible to show that the sum of squared errors for the new instances is:

$$E = \sum_{i=1}^{M} (\boldsymbol{\epsilon}_i^{\text{new}})^{\text{T}} \boldsymbol{\epsilon}_i^{\text{new}} = \text{trace}\Big(\mathbf{K}_{y^{\text{new}},y^{\text{new}}} - \mathbf{K}_{y^{\text{new}},y}\mathbf{V}\mathbf{K}_{x^{\text{new}},x}^{\text{T}} - \mathbf{K}_{x^{\text{new}},x}\mathbf{V}\mathbf{K}_{y^{\text{new}},y}^{\text{T}} + \mathbf{K}_{x^{\text{new}},x}\mathbf{V}\mathbf{K}_{y,y}\mathbf{V}\mathbf{K}_{x^{\text{new}},x}^{\text{T}} \Big) . \quad (28)$$

where $\mathbf{K}_{y^{\text{new}},y^{\text{new}}} = \mathbf{\Phi}_{y^{\text{new}}}^{\text{T}}\mathbf{\Phi}_{y^{\text{new}}}$, $\mathbf{K}_{y^{\text{new}},y} = \mathbf{\Phi}_{y^{\text{new}}}^{\text{T}}\mathbf{\Phi}_{y}$ and $\mathbf{K}_{x,x^{\text{new}}} = \mathbf{\Phi}_{x}^{\text{T}}\mathbf{\Phi}_{x^{\text{new}}}$.

Of course, the new data must be centered before computing the error estimate. This process is similar to the one described in the previous section. In particular, centering can be simply carried out by working with the modified kernel matrices:

$$\begin{aligned}
\tilde{\mathbf{K}}_{x^{\text{new}},x} &= \mathbf{K}_{x^{\text{new}},x} - \mathbf{M}_N\mathbf{K}_{x,x} - \mathbf{K}_{x^{\text{new}},x}\mathbf{1}_N + \mathbf{M}_N\mathbf{K}_{x,x}\mathbf{1}_N \,, \\
\tilde{\mathbf{K}}_{y^{\text{new}},y} &= \mathbf{K}_{y^{\text{new}},y} - \mathbf{M}_N\mathbf{K}_{y,y} - \mathbf{K}_{y^{\text{new}},y}\mathbf{1}_N + \mathbf{M}_N\mathbf{K}_{y,y}\mathbf{1}_N \,, \\
\tilde{\mathbf{K}}_{y^{\text{new}},y^{\text{new}}} &= \mathbf{K}_{y^{\text{new}},y^{\text{new}}} - \mathbf{M}_N\mathbf{K}_{y,y^{\text{new}}} - \mathbf{K}_{y^{\text{new}},y}\mathbf{M}_N^{\text{T}} + \mathbf{M}_N\mathbf{K}_{y,y}\mathbf{M}_N^{\text{T}} \,, \quad (29)
\end{aligned}$$

where $\mathbf{M}_N$ is a matrix of size $M \times N$ with all components equal to $1/N$. In this process, the averages employed for the centering step are computed using only the observed data.

A disadvantage of the squared error is that it strongly depends on the kernel bandwidth parameter γ. This makes it difficult to choose this hyper-parameter in terms of such a performance measure. A better approach is to choose both γ and τ in terms of the explained variance by the model. This is obtained as follows: Explained-Variance $= 1 - \text{E}/M\text{Var}_{\text{ynew}}$, where E denotes the squared prediction error and Var_{ynew} the variance of the targets. The computation of the error E is done as described previously and Var_{ynew} is simply the average of the diagonal entries in $\tilde{\mathbf{K}}_{y^{\text{new}},y^{\text{new}}}$.

3.6 Finding Pre-images for Illustrative Purposes

The kernel method described above expresses its solution as feature maps of the original data points. Since the feature map $\phi(\cdot)$ is usually non-linear, we cannot guarantee the existence of a pre-image under $\phi(\cdot)$. That is, a point y such that $\phi(y) = \hat{\mathbf{A}}\phi(x)$, for some input point x. An alternative to amend this issue is to find approximate pre-images, which can be useful to make predictions or plotting results (Schölkopf and Smola, 2002). In this section we describe how to find this approximate pre-images.

Assume that we have a new data instance x_{new} for which we would like to know the associated target value y_{new}, after our kernel model has been fitted. The predicted value in

feature space is:

$$\phi(y_{\text{new}}) = \mathbf{\Phi}_y\mathbf{V}\mathbf{\Phi}_x^{\text{T}}\phi(x_{\text{new}}) = \mathbf{\Phi}_y\mathbf{V}\mathbf{k}_{x,x_{\text{new}}} = \sum_{i=1}^{n} \alpha_i\phi(y_i) \,, \quad (30)$$

where $\mathbf{k}_{x,x_{\text{new}}}$ contains the kernel evaluations between each entry in $\mathbf{x}$ (*i.e.*, the observed samples of the random variable $\mathcal{X}$) and the new instance. Finally, each α_i is given by a component of the vector $\mathbf{V}\mathbf{k}_{x,x_{\text{new}}}$. The approximate pre-image of $\phi(y_{\text{new}})$, y_{new}, is found by solving the following optimization problem:

$$y_{\text{new}} = \arg\min_u \; ||\phi(y_{\text{new}}) - \phi(u)||_2^2 = \arg\min_u \; -2\boldsymbol{\alpha}^{\text{T}}\mathbf{k}_{y,u} + k(u,u) \,, \quad (31)$$

where $\mathbf{k}_{y,u}$ is a vector with the kernel values between each y_i and u, and $k(u,u) = \phi(u)^{\text{T}}\phi(u)$. This is a non-linear optimization problem than can be solved approximately using standard techniques such as gradient descent. In particular, the computation of the gradient of $\mathbf{k}_{y,u}$ with respect to u is very simple in the case of the squared exponential kernel.

4. Data Transformation and Detailed Causal Inference Algorithm

The method for causal inference described in the previous section relies on the fact that both random variables $\mathcal{X}$ and $\mathcal{Y}$ are equally distributed. In particular, if this is the case, $\phi(x_i)$ and $\phi(y_i)$, *i.e.*, the maps of x_i and y_i in the expanded feature space, will also be equally distributed. This means that under such circumstances one should expect residuals that are more Gaussian in the anti-causal direction due to a reduction of the magnitude of the high order cumulants and an increment of the differential entropy. The requirement that $\mathcal{X}$ and $\mathcal{Y}$ are equally distributed can be easily fulfilled in the case of continuous univariate data by transforming $\mathbf{x}$, the samples of $\mathcal{X}$, to have the same empirical distribution as $\mathbf{y}$, the samples of $\mathcal{Y}$.

Consider $\mathbf{x} = (x_1, \ldots, x_N)^{\text{T}}$ and $\mathbf{y} = (y_1, \ldots, y_N)^{\text{T}}$ to be N paired samples of $\mathcal{X}$ and $\mathcal{Y}$, respectively. To guarantee the same distribution for these samples we only have to replace $\mathbf{x}$ by $\tilde{\mathbf{x}}$, where each component of $\tilde{\mathbf{x}}$, $\tilde{x}_i$, is given by $\tilde{x}_i = \hat{F}_y^{-1}(\hat{F}_x(x_i))$, with $\hat{F}_y^{-1}(\cdot)$ the empirical quantile distribution function of the random variable $\mathcal{Y}$, estimated using $\mathbf{y}$. Similarly, $\hat{F}_x(\cdot)$ is the empirical cumulative distribution function of $\mathcal{X}$, estimated using $\mathbf{x}$. This operation is known as the probability integral transform.

One may wonder why should $\mathbf{x}$ be transformed instead of $\mathbf{y}$. The reason is that by transforming $\mathbf{x}$ the additive noise hypothesis made in (1) and (6) is preserved. In particular, we have that $y_i = f(\hat{F}_x^{-1}(\hat{F}_y(\tilde{x}_i))) + \epsilon_i$. On the other hand, if $\mathbf{y}$ is transformed instead, the additive noise model will generally not be valid anymore. More precisely, the transformation that computes $\tilde{\mathbf{y}}$ in such a way that it is distributed as $\mathbf{x}$ is $\tilde{y}_i = \hat{F}_x^{-1}(\hat{F}_y(y_i))$, $\forall i$. Thus, under this transformation we have that $\tilde{y}_i = \hat{F}_x^{-1}(\hat{F}_y(f(x_i) + \epsilon_i))$, which will lead to the violation of the additive noise model.

Of course, transforming $\mathbf{x}$ requires the knowledge of the causal direction. In practice, we will transform both $\mathbf{x}$ and $\mathbf{y}$ and consider that the correct transformation is the one that leads to the highest level of Gaussianization of the residuals in the feature space, after fitting the model in each direction. That is, the transformation that leads to the highest value of $\mathcal{G}$ is expected to be the correct one. We expect that when $\mathbf{y}$ is transformed instead

of $\mathbf{x}$, the Gaussianization effect of the residuals is not as high as when $\mathbf{x}$ is transformed, as a consequence of the violation of the additive noise model. This will allow to determine the causal direction. We do not have a theoretical result confirming this statement, but the good results obtained in Section 6.1 indicate that this is the case.

The details of the complete causal inference algorithm proposed are given in Algorithm 1. Besides a causal direction, *e.g.*, $\mathcal{X} \to \mathcal{Y}$ or $\mathcal{Y} \to \mathcal{X}$, this algorithm also outputs a confidence level in the decision made which is defined as $\max(|\mathcal{G}_{\tilde{x}}|, |\mathcal{G}_{\tilde{y}}|)$, where $\mathcal{G}_{\tilde{x}} = \text{Energy}(\mathbf{z}_{\tilde{x}\to y})/N - \text{Energy}(\mathbf{z}_{y\to\tilde{x}})/N$ denotes the estimated level of Gaussianization of the residuals when $\mathbf{x}$ is transformed to have the same distribution as $\mathbf{y}$. Similarly, $\mathcal{G}_{\tilde{y}} = \text{Energy}(\mathbf{z}_{\tilde{y}\to x})/N - \text{Energy}(\mathbf{z}_{\tilde{y}\to x})/N$ denotes the estimated level of Gaussianization of the residuals when $\mathbf{x}$ and $\mathbf{y}$ are swapped and $\mathbf{y}$ is transformed to have the same distribution as $\mathbf{x}$. Here $\mathbf{z}_{\tilde{x}\to y}$ contains the first principal component of the residuals in the expanded feature space when trying to predict $\mathbf{y}$ using $\tilde{\mathbf{x}}$. The same applies for $\mathbf{z}_{y\to\tilde{x}}$, $\mathbf{z}_{\tilde{y}\to x}$ and $\mathbf{z}_{x\to\tilde{y}}$. However, the residuals are obtained this time when trying to predict $\tilde{\mathbf{x}}$ using $\mathbf{y}$, when trying to predict $\mathbf{x}$ using $\tilde{\mathbf{y}}$ and when trying to predict $\tilde{\mathbf{y}}$ using $\mathbf{x}$, respectively. Recall that the reason for keeping only the first principal component of the residuals is described in Section 3.4.

Assume $|\mathcal{G}_{\tilde{x}}| > |\mathcal{G}_{\tilde{y}}|$. In this case we prefer the transformation of $\mathbf{x}$ to guarantee that the cause and the effect have the same distribution. The reason is that it leads to a higher level of Gaussianization of the residuals, as estimated by the energy statistical test. Now consider that $\mathcal{G}_{\tilde{x}} > 0$. We prefer the direction $\mathcal{X} \to \mathcal{Y}$ because the residuals of a fit in that direction are less Gaussian and hence have a higher value of the statistic of the energy test. By contrast, if $\mathcal{G}_{\tilde{x}} < 0$ we prefer the direction $\mathcal{Y} \to \mathcal{X}$ for the same reason. In the case that $|\mathcal{G}_{\tilde{x}}| < |\mathcal{G}_{\tilde{y}}|$ the reasoning is the same and we prefer the transformation of $\mathbf{y}$. However, because we have swapped $\mathbf{x}$ and $\mathbf{y}$ for computing $\mathcal{G}_{\tilde{y}}$, the decision is the opposite as the previous one. Namely, if $\mathcal{G}_{\tilde{y}} > 0$ we prefer the direction $\mathcal{Y} \to \mathcal{X}$ and otherwise we prefer the direction $\mathcal{X} \to \mathcal{Y}$. The confidence in the decision (*i.e.*, the estimated level of Gaussianization) is always measured by $\max(|\mathcal{G}_{\tilde{x}}|, |\mathcal{G}_{\tilde{y}}|)$.

The algorithm uses a squared exponential kernel with bandwidth parameter γ and the actual matrices $\hat{\mathbf{A}}_{\tilde{x}\to y}$ and $\hat{\mathbf{A}}_{y\to\tilde{x}}$, of potentially infinite dimensions, need not be evaluated in closed form in practice. As indicated in Section 3, all computations are carried out efficiently with cost $\mathcal{O}(N^3)$ using inner products, which are evaluated in terms of the corresponding kernel function. All hyper-parameters, *i.e.*, τ and γ, are chosen using a grid search method guided by a 10-fold cross-validation process. This search maximizes the explained variance of the left-out data and 10 potential values are considered for both τ and γ.

5. Related Work

The Gaussianity of residuals was first employed for causal inference by Hernández-Lobato et al. (2011). These authors analyze auto-regressive (AR) processes and show that a similar asymmetry as the one described in this paper can be used to determine the temporal direction of a time series in the presence of non-Gaussian noise. Namely, when fitting an AR process to a reversed time series, the residuals obtained follow a distribution that is closer to a Gaussian distribution. Nevertheless, unlike the work described here, the method proposed by Hernández-Lobato et al. (2011) cannot be used to tackle multidimensional or non-linear causal inference problems. In their work, Hernández-Lobato et al. (2011) show

Algorithm 1: Causal Inference Based on the Gaussianity of the Residuals (GR-AN)

Data: Paired samples $\mathbf{x}$ and $\mathbf{y}$ from the random variables $\mathcal{X}$ and $\mathcal{Y}$.
Result: An estimated causal direction alongside with a confidence level.

1 Standardize $\mathbf{x}$ and $\mathbf{y}$ to have zero mean and unit variance;
2 Transform $\mathbf{x}$ to compute $\tilde{\mathbf{x}}$; `// This guarantees that` $\tilde{\mathbf{x}}$ `is distributed as` $\mathbf{y}$.
3 $\hat{\mathbf{A}}_{\tilde{x}\to y} \leftarrow \text{FitModel}(\tilde{\mathbf{x}}, \mathbf{y})$; `// This also finds the hyper-parameters` τ `and` γ.
4 $\mathbf{z}_{\tilde{x}\to y} \leftarrow \text{ObtainResiduals}(\tilde{\mathbf{x}}, \mathbf{y}, \hat{\mathbf{A}}_{\tilde{x}\to y})$; `// First PCA component in feature space.`
5 $\hat{\mathbf{A}}_{y\to\tilde{x}} \leftarrow \text{FitModel}(\mathbf{y}, \tilde{\mathbf{x}})$; `// Fit the model in the other direction`
6 $\mathbf{z}_{y\to\tilde{x}} \leftarrow \text{ObtainResiduals}(\mathbf{y}, \tilde{\mathbf{x}}, \hat{\mathbf{A}}_{y\to\tilde{x}})$; `// First PCA component in feature space.`
7 $\mathcal{G}_{\tilde{x}} \leftarrow \text{Energy}(\mathbf{z}_{\tilde{x}\to y})/N - \text{Energy}(\mathbf{z}_{y\to\tilde{x}})/N$; `// Get the Gaussianization level.`
8 Swap $\mathbf{x}$ and $\mathbf{y}$ and repeat lines 2-7 of the algorithm to compute $\mathcal{G}_{\tilde{y}}$.
9 **if** $|\mathcal{G}_{\tilde{x}}| > |\mathcal{G}_{\tilde{y}}|$ **then**
10 **if** $\mathcal{G}_{\tilde{x}} > 0$ **then**
11 **Output**: $\mathcal{X} \to \mathcal{Y}$ with confidence $|\mathcal{G}_{\tilde{x}}|$
12 **else**
13 **Output**: $\mathcal{Y} \to \mathcal{X}$ with confidence $|\mathcal{G}_{\tilde{x}}|$
14 **end**
15 **else**
16 **if** $\mathcal{G}_{\tilde{y}} > 0$ **then**
17 **Output**: $\mathcal{Y} \to \mathcal{X}$ with confidence $|\mathcal{G}_{\tilde{y}}|$
18 **else**
19 **Output**: $\mathcal{X} \to \mathcal{Y}$ with confidence $|\mathcal{G}_{\tilde{y}}|$
20 **end**
21 **end**

some advantages of using statistical tests based on measures of Gaussianity to determine the temporal direction of a time series, as a practical alternative to statistical tests based on the independence of the cause and the residual. The motivation for these advantages is that the former tests are one-sample tests while the later ones are two-sample tests.

The previous paper is extended by Morales-Mombiela et al. (2013) to consider multidimensional AR processes. However, this work lacks a theoretical result that guarantees that the residuals obtained when fitting a vectorial AR process in the reversed (anti-chronological) direction will follow a distribution closer to a Gaussian distribution. In spite of this issue, extensive experiments with simulated data suggest the validity of such conjecture. Furthermore, a series of experiments show the superior results of the proposed rule to determine the direction of time, which is based on measures of Gaussianity, and compared with other state-of-the-art methods based on tests of independence.

The problem of causal inference under continuous-valued data has also been analyzed by Shimizu et al. (2006). The authors propose a method called LINGAM that can identify the causal order of several variables when assuming that (a) the data generating process is linear, (b) there are no unobserved co-founders, and (c) the disturbance variables have non-Gaussian distributions with non-zero variances. These assumptions are required because LINGAM relies on the use of Independent Component Analysis (ICA). More specifically, let $\mathbf{x}$ denote a vector that contains the variables we would like to determine the causal

order of. LINGAM assumes that $\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{e}$, where $\mathbf{B}$ is a matrix that can be permuted to strict lower triangularity if one knows the actual causal ordering in $\mathbf{x}$, and $\mathbf{e}$ is a vector of non-Gaussian independent disturbance variables. Solving for $\mathbf{x}$, one gets $\mathbf{x} = \mathbf{A}\mathbf{e}$, where $\mathbf{A} = (\mathbf{I} - \mathbf{B})^{-1}$. The $\mathbf{A}$ matrix can be inferred using ICA. Furthermore, given an estimate of $\mathbf{A}$, $\mathbf{B}$ can be obtained to find the corresponding connection strengths among the observed variables, which can then be used to determine the true causal ordering. LINGAM has been extended to consider linear relations among groups of variables (Entner and Hoyer, 2012; Kawahara et al., 2012).

In real-world data, causal relationships tend to be non-linear, a fact that questions the usefulness of linear methods. Hoyer et al. (2009) show that a basic linear framework for causal inference can be generalized to non-linear models. For non-linear models with additive noise, almost any non-linearities (invertible or not) will typically yield identifiable models. In particular, Hoyer et al. (2009) assume that $y_i = f(x_i) + \epsilon_i$, where $f(\cdot)$ is a possibly non-linear function, x_i is the cause variable, and ϵ_i is some independent and random noise. The proposed causal inference mechanism consists in performing a non-linear regression on the data to get an estimate of $f(\cdot)$, $\hat{f}(\cdot)$, and then calculate the corresponding residuals $\hat{\epsilon}_i = y_i - \hat{f}(x_i)$. Then, one may test whether $\hat{\epsilon}_i$ is independent of x_i or not. The same process is repeated in the other direction. The direction with the highest level of independence is chosen as the causal one. In practice, the estimate $\hat{f}(\cdot)$ is obtained using Gaussian processes for regression, and the HSIC test (Gretton et al., 2008) is used as the independence criterion. This method has obtained good performance results (Janzing et al., 2012) and it has been extended to address problems where the model is $y_i = h(f(x_i) + \epsilon_i)$, for some invertible function $h(\cdot)$ (Zhang and Hyvärinen, 2009). A practical difficulty is however that such a model is significantly harder to fit to the data.

In the work by Mooij et al. (2010), a method for causal inference is proposed based on a latent variable model, used to incorporate the effects of un-observed noise. In this context, it is considered that the effect variable is a function of the cause variable and an independent noise term, not necessarily additive, that is, $y_i = f(x_i, \epsilon_i)$, where x_i is the cause variable and ϵ_i is some independent and random noise. The causal direction is then inferred using standard Bayesian model selection. In particular, the preferred direction is the one under which the corresponding model has the largest marginal likelihood, where the marginal likelihood is understood as a proxy for the Kolmogorov complexity. This method suffers from several implementation difficulties, including the intractability of the marginal likelihood computation. However, it has shown encouraging results on synthetic and real-world data.

Janzing et al. (2010) consider the problem of inferring linear causal relations among multi-dimensional variables. The key point here is to use an asymmetry between the distributions of the cause and the effect that occurs if the covariance matrix of the cause and the matrix mapping the cause to the effect are independently chosen. This method exhibits the nice property that applies to both deterministic and stochastic causal relations, provided that the dimensionality of the involved random variables is sufficiently high. The method assumes that $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i + \boldsymbol{\epsilon}_i$, where $\mathbf{x}_i$ is the cause and $\boldsymbol{\epsilon}_i$ is additive noise. Namely, denote with $\hat{\boldsymbol{\Sigma}}$ to the empirical covariance matrix of the variables in each $\mathbf{x}_i$. Given an estimate of $\mathbf{A}$, $\hat{\mathbf{A}}$, the method computes $\boldsymbol{\Delta}_{\mathbf{x}\to\mathbf{y}} = \log\text{trace}(\hat{\mathbf{A}}\hat{\boldsymbol{\Sigma}}\hat{\mathbf{A}}^{\mathrm{T}}) - \log\text{trace}(\hat{\boldsymbol{\Sigma}}) + \log\text{trace}(\hat{\mathbf{A}}\hat{\mathbf{A}}^{\mathrm{T}}) + d$, where d is the dimension of $\mathbf{x}_i$. This process is repeated to compute $\boldsymbol{\Delta}_{\mathbf{y}\to\mathbf{x}}$ where $\mathbf{x}_i$ and

$\mathbf{y}_i$ are swapped. The asymmetry described states that $\boldsymbol{\Delta}_{\mathbf{x}\to\mathbf{y}}$ should be close to zero while $\boldsymbol{\Delta}_{\mathbf{y}\to\mathbf{x}}$ should not. Thus, if $|\boldsymbol{\Delta}_{\mathbf{x}\to\mathbf{y}}| > |\boldsymbol{\Delta}_{\mathbf{y}\to\mathbf{x}}|$, $\mathbf{x}_i$ is expected to be the cause. Otherwise, the variables in $\mathbf{y}_i$ are predicted to be cause instead. Finally, a kernelized version of this method is also described by Chen et al. (2013).

Most of the methods introduced in this section assume some form of noise in the generative process of the effect. Thus, their use is not justified in the case of noiseless data. Janzing et al. (2012) describe a method to deal with these situations. In particular, the method makes use of information geometry to identify an asymmetry that can be used for causal inference. The asymmetry relies on the idea that the marginal distribution of the cause variable, denoted by $p(x)$, is expected to be chosen independently from the mapping mechanism producing the effect variable, denoted by the conditional distribution $p(y|x)$. Independence is defined here as orthogonality in the information space, which allows to describe a dependence that occurs between $p(y)$ and $p(x|y)$ in the anti-causal direction. This dependence can be then used to determine the causal order. A nice property of this method is that this asymmetry between the cause and the effect becomes very simple if both random variables are deterministically related. Remarkably, the method also performs very well in noisy scenarios, although no theoretical guarantees are provided in this case.

A similar method for causal inference to the last one is described by Chen et al. (2014). These authors also consider that $p(x)$ and $p(y|x)$ fulfill some sort of independence condition, and that this independence condition does not hold for the anti-causal direction. Based on this, they define an uncorrelatedness criterion between $p(x)$ and $p(y|x)$, and show an asymmetry between the cause and the effect in terms of a certain complexity metric on $p(x)$ and $p(y|x)$, which is less than the same complexity metric on $p(y)$ and $p(x|y)$. The complexity metric is calculated in terms of a reproducing kernel Hilbert space embedding (EMD) of probability distributions. Based on the complexity metric, the authors propose an efficient kernel-based algorithm for causal discovery.

In Section 2.3 we have shown that in the multivariate case one should expect higher entropies in the anti-causal direction. Similar results have been obtained in the case of non-linear relations and the univariate data case (Hyvärinen and Smith, 2013; Kpotufe et al., 2014). Assume $x, y \in \mathbb{R}$ and the actual causal model to be $y = f(x) + d$, with $x \perp d$ and $f(\cdot)$ an arbitrary function. Let e be the residual of a fit performed in the anti-causal direction. Section 5.2 of the work by Hyvärinen and Smith (2013) shows that the likelihood ratio R of each model (*i.e.*, the model fitted in the causal direction and the model fitted in the anti-causal direction) converges in the presence of infinite data to the difference between the sum of the entropies of the independent variable and the residual in each direction. Namely, $R \to -H(x) - H(d/\sigma_d) + H(y) + H(e/\sigma_e) + \log\sigma_d - \log\sigma_e$, where σ_d and σ_e denote the standard deviation of the errors in each direction. If $R > 0$, the causal direction is chosen. By contrast, if $R < 0$ the anti-causal direction is preferred. The process of evaluating R involves the estimation of the entropies of four univariate random variables, *i.e.*, x, d, y and e and the standard deviation of the errors d and e, which need not be equal. The non-linear functions are estimated as in the work by Hoyer et al. (2009) using a Gaussian process. The entropies are obtained using a maximum entropy approximation under the hypothesis that the distributions of these variables are not far from Gaussian (Hyvärinen, 1998). The resulting method is called non-linear maximum entropy (NLME). A practical difficulty is however that the estimation of the entropy is a very difficult task,

even in one dimension (Beirlant et al., 1997). Thus, the NLME method is adapted in an *ad-hoc* manner with the aim of obtaining better results in certain difficult situations with sparse residuals. More precisely, if $H(x)$ and $H(y)$ are ignored and Laplacian residuals are assumed $R \rightarrow \log \sigma_e - \log \sigma_d$. That is, the model with the minimum error is preferred. The errors are estimated however in terms of the absolute deviations (because of the Laplacian assumption). This method is called mean absolute deviation (MAD). Finally, Kpotufe et al. (2014) show the consistency of the noise additive model, give a formal proof for $R \geq 0$ (see Lemma 1), and propose to estimate $H(x)$, $H(y)$, $H(d)$ and $H(e)$ using kernel density estimators. Note that if x and y are equally distributed, $H(x) = H(y)$ and the condition $R \geq 0$ implies $H(e) \geq H(d)$. Nevertheless, σ_d and σ_e are in general different (see the supplementary material for an illustrative example). This means that in the approach of Hyvärinen and Smith (2013) and Kpotufe et al. (2014) it is not possible to make a decision directly on the basis of a Gaussianization effect on the residuals.

The proposed method GR-AN, introduced in Section 4, differs from the approaches described in the previous paragraph in that it does not have to deal with the estimation of four univariate entropies, which can be a particularly difficult task. By contrast, it relies on statistical tests of deviation from Gaussianity to infer the causal direction. Furthermore, the tests employed in our method need not be directly related to entropy estimation. This is particularly the case of the energy test suggested in Section 3.4. Not having to estimate differential entropies is an advantage of our method confirmed by the results that are obtained in the experiments section. In particular, we have empirically observed that that GR-AN performs better than the two methods for causal inference NLME and MAD that have been described in the previous paragraph. GR-AN also performs better than GR-ENT, a method that uses, instead of statistical tests of Gaussianity, a non-parametric estimator of the entropy (Singh et al., 2003).

6. Experiments

We carry out experiments to validate the method proposed in this paper, and empirically verify that the model residuals in the anti-causal direction are more Gaussian that the model residuals in the causal direction due to a reduction of the high-order cumulants and an increment of the differential entropy. From now on, we refer to our method as GR-AN (Gaussianity of the Residuals under Additive Noise). Furthermore, we compare the performance of GR-AN with four other approaches from the literature on causal inference, reviewed in Section 5. First, LINGAM (Shimizu et al., 2006), a method which assumes an additive noise model, but looks for independence between the cause and the residuals. Second, IR-AN (Independence of the Residuals under Additive Noise), by Hoyer et al. (2009). Third, a method based on information geometry, IGCI (Janzing et al., 2012). Fourth, a method based on Reproducing Kernel Hilbert Space Embeddings (EMD) of probability distributions (Chen et al., 2014). Fifth, the two methods for non-linear causal inference based on entropy estimation described by Hyvärinen and Smith (2013), NLME and MAD. Sixth, the same GR-AN method, but where we omit the transformation to guarantee that the random variables $\mathcal{X}$ and $\mathcal{Y}$ are equally distributed. This method is called GR-AN*. Last, we also compare results with two variants of GR-AN that are not based on the energy distance to measure the level of Gaussianity of the residuals. These are GR-K4, which uses

the empirical estimate of the fourth cumulant (kurtosis) to determine the causal direction (it chooses the direction with the largest estimated fourth cumulant); and GR-ENT, which uses a non-parametric estimator of the entropy (Singh et al., 2003) to determine the causal direction (the direction with the smallest entropy is preferred);

The hyper-parameters of the different methods are set as follows. In LINGAM, we use the parameters recommended by the implementation provided by the authors. In IR-AN, NLME and MAD we employ a Gaussian process whose hyper-parameters are found by type-II maximum likelihood. Furthermore, in IR-AN the HSIC test is used to assess independence between the causes and the residuals. In NLME the entropy estimator is the one described by Hyvärinen (1998). In IGCI, we test different normalizations (uniform and Gaussian) and different criteria (entropy or integral) and report the best observed result. In EMD and synthetic data, we follow Chen et al. (2014) to select the hyper-parameters. In EMD and real-world data, we evaluate different hyper-parameters and report the results for the best combination found. In GR-AN, GR-K4 and GR-ENT the hyper-parameters are found via cross-validation, as described in Section 4. The number of neighbors in the entropy estimator of GR-ENT is set to 10, a value that we have observed to give a good trade-off between bias and variance. Finally, in GR-AN, GR-K4, and GR-ENT we transform the data so that both variables are equally distributed, as indicated in Section 4.

The confidence in the decision is computed as indicated by Janzing et al. (2012). More precisely, in LINGAM the confidence is given by the maximum absolute value of the entries in the connection strength matrix $\mathbf{B}$. In IGCI we employ the absolute value of the difference between the corresponding estimates (entropy or integral) in each direction. In IR-AN the confidence level is obtained as the maximum of the two p-values of the HSIC test. In EMD we use the absolute value of the difference between the estimates of the corresponding complexity metric in each direction, as described in (Chen et al., 2014). In NLME and MAD the confidence level is given by the absolute value of the difference between the outputs of the entropy estimators in each direction (Hyvärinen and Smith, 2013). In GR-K4 we use the absolute difference between the estimated fourth cumulants. In GR-ENT we use the absolute difference between the estimates of the entropy. Finally, in GR-AN we follow the details given in Section 4 to estimate the confidence in the decision.

To guarantee the exact reproducibility of the different experiments described in this paper, the source-code for all methods and data sets is available in the public repository `https://bitbucket.org/dhernand/gr_causal_inference`.

6.1 Experiments with Synthetic Data

We carry out a first batch of experiments on synthetic data. In these experiments, we employ the four causal mechanisms that map $\mathcal{X}$ to $\mathcal{Y}$ described by Chen et al. (2014). They involve linear and non-linear functions, and additive and multiplicative noise effects:

- M_1: $y_i = 0.8x_i + \epsilon_i$.
- M_2: $y_i = x_i\epsilon_i$.
- M_3: $y_i = 0.3x_i^3 + \epsilon_i$.
- M_4: $y_i = \text{atan}(x_i)^3 + \epsilon_i$.

The noise ϵ_i can follow four different types of distributions: (i) A generalized Gaussian distribution with shape parameter equal to 10 (an example of a sub-Gaussian distribution); (ii) a Laplace distribution (an example of a super-Gaussian distribution); (iii) a Gaussian distribution; and (iv) a bimodal distribution with density $p(\epsilon_i) = 0.5\mathcal{N}(\epsilon_i|m, s)+0.5\mathcal{N}(\epsilon_i|-m, s)$, where $m = .63$ and $s = .1$. The Laplace distribution and the Gaussian distribution are adjusted to have the same variance as the generalized Gaussian distribution. The bimodal distribution already has the same variance as the generalized Gaussian distribution.

As indicated by Chen et al. (2014), in these experiments, the samples from the cause variable $\mathcal{X}$ are generated from three potential distributions:

- $p_1(x) = \frac{1}{\sqrt{2\pi}} \exp\{-x^2/2\}$.
- $p_2(x) = \frac{1}{2\sqrt{0.5\pi}} \exp\{-(x+1)^2/0.5\} + \frac{1}{2\sqrt{0.5\pi}} \exp\{-(x-1)^2/0.5\}$.
- $p_3(x) = \frac{1}{4\sqrt{0.5\pi}} \exp\{-(x+1.5)^2/0.5\}+\frac{1}{2\sqrt{0.5\pi}} \exp\{-x^2/0.5\}+\frac{1}{4\sqrt{0.5\pi}} \exp\{-(x-1.5)^2/0.5\}$.

These are unimodal, bimodal, and trimodal distributions, respectively.

Figure 5 displays a representative example of the plots of different combinations of distributions and mapping mechanisms when the noise follows a generalized Gaussian distribution with shape parameter equal to 10. The plots for Laplace, Gaussian or bimodal distributed noise look similar to these ones. The assumptions made by proposed method, *i.e.*, GR-AN, are valid in the case of all the causal mechanisms, except for M2, which considers multiplicative noise, and in the case of all cause distributions, p_1, p_2 and p_3. The only type of noise that violates the assumptions made by GR-AN is the case of Gaussian noise. In particular, under Gaussian noise GR-AN cannot infer the causal direction using Gaussianity measures because the actual residuals are already Gaussian.

The average results of each method on 100 repetitions of each potential causal mechanism, distribution for the effect, and noise distribution are displayed in Table 1. The size of each paired samples of $\mathcal{X}$ and $\mathcal{Y}$ is set to 500 in these experiments. We observe that when the assumptions made by the proposed method, GR-AN, are satisfied, it identifies the causal direction on a very high fraction of the 100 repetitions considered. However, when these assumptions are not valid, *e.g.*, in the case of the M2 causal mechanism, which has multiplicative noise, the performance worsens. The same happens when the distribution of the residuals is Gaussian. In these experiments, LINGAM tends to fail when the causal relation is strongly non-linear. This is the case of the causal mechanism M3. LINGAM also has problems when all the independent variables are Gaussian. Furthermore, all methods generally fail in the case of independent Gaussian variables that are linearly related. This corresponds to the causal mechanism M1, the distribution $p_1(x)$ for the cause, and the Gaussian distribution for the noise. The reason for this is that in this particular scenario the causal direction is not identifiable (Shimizu et al., 2006). IGCI and EMD sometimes fail in the case of the causal mechanism M1 and M4. However, they typically correctly identify the causal direction in the case of the mechanism M2, which has non-additive noise, and where the other methods tend to fail. Finally, IR-AN performs slightly better than GR-AN, especially in the case of additive Gaussian noise, where GR-AN is unable to identify the causal direction. MAD provides very bad results for some of the mechanism considered, *i.e.*, M1 and M4. Surprisingly this is the case even for Laplacian additive noise, which is

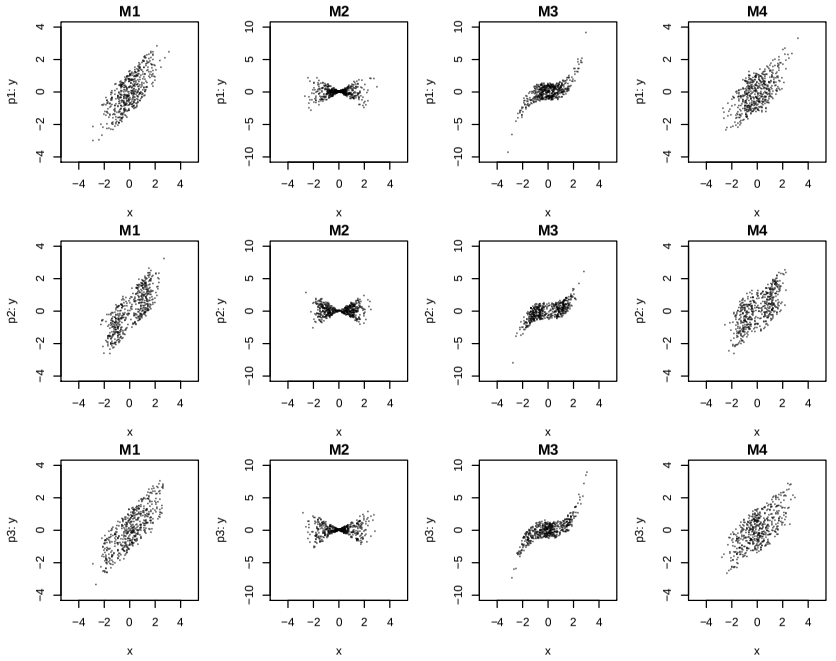


Figure 5: Plots of different distributions and mechanisms when the noise follows a generalized Gaussian distribution with shape parameter equal to 10.

the hypothesis made by MAD. This bad behavior is probably a consequence of ignoring the entropies of $\mathcal{X}$ and $\mathcal{Y}$ in this method. NLME and GR-ENT give worse results than GR-AN in some particular cases, *e.g.*, $p1$ and the causal mechanism M3. We believe this is related to the difficulty of estimating differential entropies in practice. GR-AN* performs very similar to GR-AN. This indicates that in practice one may ignore the transformation that guarantees that $\mathcal{X}$ and $\mathcal{Y}$ are equally distributed. GR-K4 also gives similar results to GR-AN, probably because in these experiments the tails of the residuals are not very heavy.

An overall comparison of the different methods evaluated is shown in Figure 6. This figure displays several radar charts that indicate the average accuracy of each method for the different types of noise considered and for each mechanism M1, M2, M3 and M4. In particular, for a given method and a given type of noise, the radius of each portion of the pie is proportional to the corresponding average accuracy of the method across the distributions p_1, p_2 and p_3 for the cause. The pie at the bottom corresponds to 100% accuracy for each causal mechanism. The conclusions derived from this figure are similar to the ones obtained

from Table 1. In particular, IR-AN performs very well, except for multiplicative noise (M3), closely followed by GR-AN, GR-AN*, and GR-K4, which give similar results. The methods perform very poorly in the case of additive Gaussian noise, since they cannot infer the actual causal direction in that situation. NLME and GR-ENT have problems in the case of the causal mechanism M3 and MAD in the case of the mechanisms M1 and M4. LINGAM also performs bad in the case of M3 and IGCI and EMD have problems in the case of the mechanisms M1 and M4. IGCI, EMD and MAD are the only methods performing well in the case of M2, the mechanism with non-additive noise.

We have repeated these experiments for other samples sizes, *e.g.*, 100, 200 300 and 1000. The results obtained are very similar to the ones reported here, except when the number of samples is small and equal to 100. In that case NLME performs slightly better than the proposed approach GR-AN, probably because with 100 samples it is very difficult to accurately estimate the non-linear transformation that is required to guarantee that $\mathcal{X}$ and $\mathcal{Y}$ are equally distributed. These results of these additional experiments are found in the supplementary material.

In summary, the good results provided by GR-AN and its variants in the experiments described indicate that (i) when the assumptions made by GR-AN are valid, the method has a good performance and (ii) there is indeed a Gaussianization effect in the residuals when the model is fitted under the anti-causal direction. Because GR-AN* also performs well in these experiments, this indicates that a Gaussianization of the residuals may happen even when $\mathcal{X}$ and $\mathcal{Y}$ do not follow the same distribution.

We give further evidence of the Gaussianization of the distribution of the residuals obtained when fitting the model under the anti-causal direction. For this, we analyze in detail three particular cases of GR-AN corresponding to the causal mechanism M3, the distribution $p_2(x)$ for the cause and each of the three types of additive noise considered. Namely, generalized Gaussian noise, Laplacian noise and bimodal noise. Figure 7 shows the predicted pre-images for new data instances when the model has been fitted in the causal ($\tilde{\mathcal{X}} \to \mathcal{Y}$) and the anti-causal ($\mathcal{Y} \to \tilde{\mathcal{X}}$) direction alongside with a histogram of the first principal component of the residuals in feature space. A Gaussian approximation is also displayed as a solid black line on top of the histogram. In this case $\mathbf{x}$, *i.e.*, the samples of $\mathcal{X}$, have been transformed to be equally distributed to $\mathbf{y}$, *i.e.*, the samples from $\mathcal{Y}$. We observe that the distribution of the residuals in the anti-causal direction ($\mathcal{Y} \to \tilde{\mathcal{X}}$) is more similar to a Gaussian distribution. Furthermore, for the direction $\tilde{\mathcal{X}} \to \mathcal{Y}$ the statistic of the energy based Gaussianity test for the first principal component of the residuals is respectively 4.02, 4.64 and 11.46, for generalized Gaussian, Laplacian and bimodal noise. Recall that the larger the value the larger the deviation from Gaussianity. In the case of the direction $\mathcal{Y} \to \tilde{\mathcal{X}}$, the energy statistic associated to the residuals is 0.97, 0.68 and 1.31, respectively. When $\mathbf{y}$ is transformed to have the same distribution as $\mathbf{x}$ similar results are observed (results not shown). However, the Gaussianization effect is not as strong as in this case, probably because it leads to the violation of the addtitive noise assumption. In summary, the figure displayed illustrates in detail the Gaussianization effect of the residuals when fitting the model in the anti-causal direction.

Table 1: Accuracy on the synthetic data for each method, causal mechanisms and type of noise.

Noise	Generalized Gaussian				Laplacian				Gaussian				Bimodal			
	Mechanism				Mechanism				Mechanism				Mechanism			
Algorithm	**M1**	**M2**	**M3**	**M4**	**M1**	**M2**	**M3**	**M4**	**M1**	**M2**	**M3**	**M4**	**M1**	**M2**	**M3**	**M4**
$p_1(x)$																
LINGAM	100%	1%	7%	93%	100%	0%	26%	89%	58%	0%	11%	12%	100%	0%	3%	100%
IGCI	28%	96%	100%	34%	73%	100%	100%	96%	49%	100%	100%	63%	29%	98%	100%	44%
EMD	34%	96%	100%	43%	70%	100%	100%	77%	48%	100%	100%	50%	50%	99%	100%	55%
IR-AN	100%	31%	100%	99%	99%	26%	100%	97%	45%	34%	100%	75%	100%	46%	100%	100%
NLME	100%	31%	22%	100%	100%	20%	15%	100%	45%	22%	8%	96%	100%	42%	17%	100%
MAD	0%	94%	100%	0%	100%	99%	100%	100%	50%	99%	100%	97%	0%	92%	97%	0%
GR-AN	100%	46%	95%	100%	100%	44%	80%	100%	48%	47%	6%	20%	100%	51%	100%	100%
GR-AN*	100%	0%	94%	100%	100%	0%	94%	100%	45%	0%	1%	17%	100%	0%	100%	100%
GR-K4	100%	70%	94%	100%	99%	51%	96%	100%	52%	56%	2%	19%	100%	53%	100%	100%
GR-ENT	100%	71%	76%	97%	93%	52%	51%	93%	41%	58%	26%	29%	100%	50%	84%	99%
$p_2(x)$																
LINGAM	100%	32%	68%	100%	100%	30%	99%	100%	100%	13%	91%	100%	100%	53%	91%	100%
IGCI	40%	97%	100%	71%	98%	100%	100%	99%	72%	100%	100%	97%	39%	99%	100%	54%
EMD	96%	99%	100%	98%	96%	100%	100%	96%	94%	100%	100%	90%	92%	95%	100%	95%
IR-AN	100%	55%	100%	100%	100%	42%	100%	100%	100%	44%	100%	100%	92%	43%	100%	100%
NLME	100%	47%	100%	100%	95%	36%	100%	100%	99%	36%	100%	100%	100%	38%	100%	100%
MAD	0%	100%	100%	13%	5%	100%	100%	100%	2%	100%	100%	95%	0%	100%	85%	0%
GR-AN	100%	46%	100%	100%	98%	32%	96%	100%	29%	40%	16%	54%	100%	32%	100%	100%
GR-AN*	100%	0%	94%	100%	100%	0%	65%	100%	39%	0%	0%	6%	100%	0%	100%	100%
GR-K4	100%	50%	100%	100%	87%	44%	100%	98%	26%	51%	23%	47%	100%	30%	100%	100%
GR-ENT	96%	56%	90%	98%	78%	40%	73%	92%	46%	45%	42%	48%	100%	36%	97%	100%
$p_3(x)$																
LINGAM	100%	0%	19%	100%	100%	0%	98%	100%	92%	0%	56%	67%	100%	8%	7%	100%
IGCI	76%	100%	100%	83%	100%	100%	100%	100%	92%	100%	100%	97%	67%	100%	100%	87%
EMD	90%	100%	100%	94%	98%	100%	100%	100%	92%	100%	100%	97%	96%	100%	100%	94%
IR-AN	100%	40%	100%	100%	100%	26%	100%	100%	96%	35%	100%	100%	100%	45%	100%	100%
NLME	100%	44%	100%	100%	100%	26%	100%	100%	74%	34%	98%	100%	100%	43%	99%	100%
MAD	0%	97%	100%	1%	100%	100%	100%	100%	38%	98%	100%	98%	0%	95%	99%	0%
GR-AN	100%	52%	90%	100%	100%	58%	100%	100%	46%	53%	36%	45%	100%	54%	98%	100%
GR-AN*	100%	0%	100%	100%	99%	0%	100%	100%	51%	0%	12%	26%	100%	0%	100%	100%
GR-K4	100%	64%	94%	100%	99%	47%	100%	97%	43%	54%	15%	24%	100%	57%	100%	100%
GR-ENT	100%	60%	42%	95%	94%	54%	86%	91%	47%	48%	48%	32%	100%	47%	43%	99%

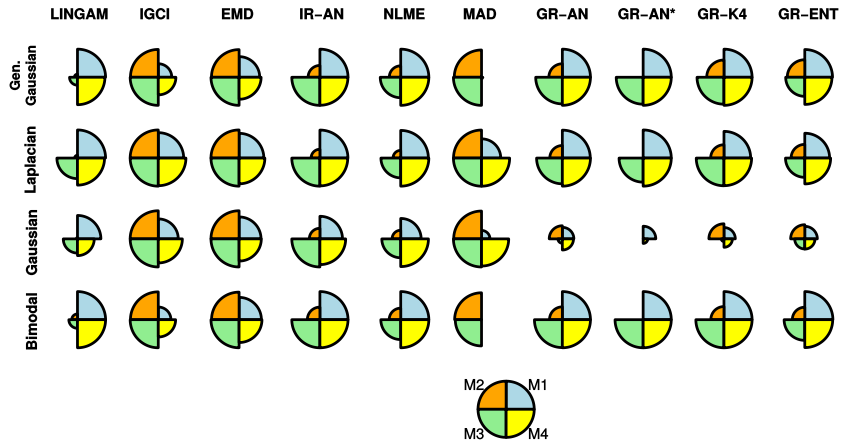


Figure 6: Radar charts showing the average accuracy of each method for the different types of noise considered and for each mechanism M1, M2, M3 and M4. For a particular method and type of noise, the radius of each portion of the pie is proportional to the corresponding average accuracy of the method across the distributions p_1, p_2 and p_3 for the cause. The pie at the bottom corresponds to 100% accuracy for each mechanism.

6.2 Experiments with Real Cause-effect Pairs

A second batch of experiments is performed on the cause-effect pairs from the ChaLearn challenge[2]. This challenge contains 8073 cause-effect data pairs with a labeled causal direction. From these pairs, we consider a subset for our experiments. In particular, we select the 184 pairs that have (i) at least 500 samples, and (ii) a fraction of repeated instances for each random variable of at most 1%. The first criterion guarantees that there is enough data to make a decision with high confidence. The second criterion removes the pairs with discrete random variables, motivated by the transformation required by the GR-AN method to guarantee the equal distribution of $\mathcal{X}$ and $\mathcal{Y}$. In particular, this transformation cannot be carried out on discrete data. Another advantage is that this filtering process of the data facilitates the experiments since several of the methods considered in the comparison (*i.e.*, GR-AN, GR-ENT, GR-K4, IR-AN, NLME, MAD and EMD) are computationally very expensive. More precisely, they have a cubic cost with respect to the number of samples and they require tuning several hyper-parameters. The consequence is that evaluating these methods on the 8073 pairs available is therefore not feasible.

2. See https://www.codalab.org/competitions/1381 for more information.

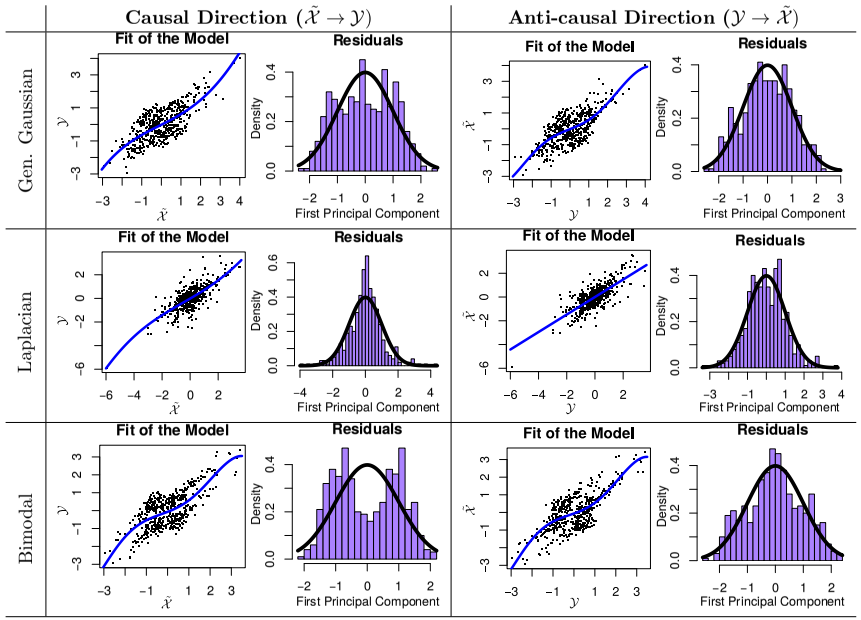


Figure 7: (left column) Predicted pre-images obtained in the casual direction $\tilde{\mathcal{X}} \to \mathcal{Y}$ alongside with a histogram of the first principal component of the residuals in feature space. A Gaussian fit is displayed as a solid black line. Results are shown for each type of additive noise considered. (right column) Same plots for the anti-causal direction $\mathcal{Y} \to \tilde{\mathcal{X}}$.

Using these 184 pairs we evaluate each of the methods considered in the previous section and report the corresponding accuracy as a function of the decisions made. In these experiments we sample at random 500 instances from each cause-effect pair. This is a standard number of samples that has been previously employed by other authors in their experiments with cause-effect pairs (Janzing et al., 2012). Furthermore, a threshold value is fixed and the obtained confidence in the decision by each method is compared to such threshold. Only if the confidence is above the threshold value, the cause-effect pair is considered in the evaluation of the accuracy of the corresponding method. A summary of the results is displayed in Figure 8. This figure shows for each method, as a fraction of the decisions made, the accuracy on the filtered data sets on which the confidence on the decision is above the threshold value. A gray area has been drawn to indicate accuracy values that are not statistically different from random guessing (accuracy equal to 50%) using a bino-

mial test (p-value above 5%). We observe that IR-AN obtains the best results, followed by GR-AN, GR-AN*, GR-K4, GR-ENT, IGCI, NLME and EMD. NLME, IGCI and EMD perform worse than GR-AN and GR-AN* when a high number of decisions are made. The differences in performance between IR-AN and GR-AN, when 100% of the decisions are made, are not statistically significant (a paired t-test returns a p-value equal to 25%). The fact that the performance of GR-AN* is similar to the performance of GR-AN also indicates that there is some Gaussianization of the residuals even though the two random variables $\mathcal{X}$ and $\mathcal{Y}$ are not equally distributed. We observe that the results of LINGAM and MAD are not statistically different from random guessing. This remarks the importance of non-linear models and questions the practical utility of the MAD method. In these experiments, GR-ENT and GR-K4 perform worse than GR-AN, which remarks the benefits of using the energy distance to estimate the deviation from the Gaussian distribution, as a practical alternative to entropy or cumulant based measures.

In summary, the results displayed in Figure 8 confirm that the level of Gaussianity of the residuals, estimated using statistical tests, is a useful metric that can be used to identify the causal order of two random variables. Furthermore, this figure also validates the theoretical results obtained in Section 2 which state that one should expect residuals whose distribution is closer to the Gaussian distribution when performing a fit in the anti-causal direction.

In the supplementary material we include additional results for other sample sizes. Namely, 100, 200 and 300 samples. The results obtained are similar to the ones reported in Figure 8. However, the differences between GR-ENT, NLME, GR-AN and GR-AN* are smaller. Furthermore, when the number of samples is small (*i.e.*, equal to 100) GR-AN performs worse than NLME, probably because with such a small number of samples it is difficult to estimate the non-linear transformation that guarantees that $\mathcal{X}$ and $\mathcal{Y}$ are equality distributed.

In this section we have also evaluated the different methods compared in the previous experiments on a random subset of 184 cause-effect pairs chosen across the 8073 pairs of the ChaLearn challenge (results not shown). In this case, the ranking of the curves obtained looks similar to the ranking displayed in Figure 8, *i.e.*, IR-AN performs best followed by GR-AN, GR-AN*, NLME and EMD. However, all methods obtain worse results in general and none of them, except IR-AN, perform significantly different from random guessing.

Finally, we also have evaluated the different methods in a subset of 82 cause-effect pairs extracted from the Tübingen cause-effect pairs[3]. We only considered those pairs with scalar cause and effect. The results obtained are displayed in Figure 9. In this case, the performance of the different methods is worse than the one displayed in Figure 8. Only IR-AN, IGCI and MAD perform significantly better than random guessing. Furthermore, GR-AN and GR-AN* do not perform well in this set of cause-effect pairs. This is also the case of NLME. We believe that the reason for this bad performance is that in most of these pairs some of the variables take discrete or repeated values. In the case of GR-AN this makes infeasible to transform the two random variables, $\mathcal{X}$ and $\mathcal{Y}$, so that they are equally distributed. Furthermore, the discrete random variables may have a strong impact in the tests for Gaussianity and in the estimation of the differential entropy. This could explain the bad performance of GR-AN*, NLME and GR-ENT.

3. See http://webdav.tuebingen.mpg.de/cause-effect/ for more details.

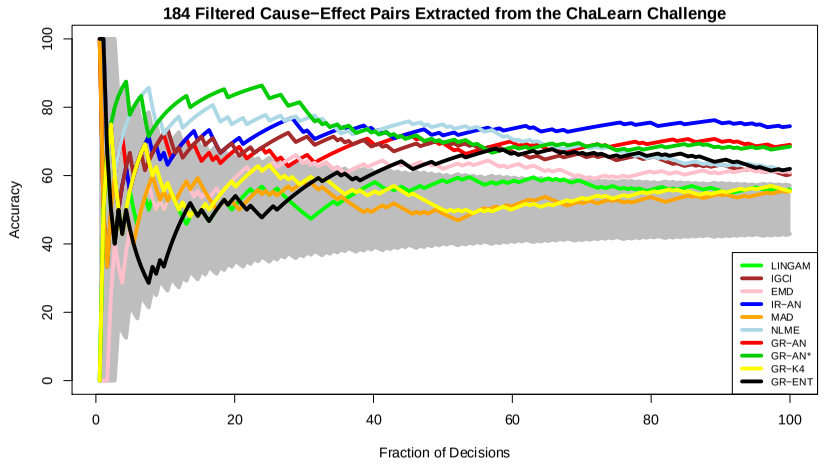


Figure 8: Accuracy of each method, as a fraction of the decisions made, on the 184 filtered cause-effect pairs extracted from the ChaLearn challenge. The number of samples of each pair is equal to 500. Best seen in color.

In summary, the results reported in this section have shown that in some cause-effect pairs, when the assumptions made by the proposed method are satisfied, there is indeed a Gaussianization effect in the residuals obtained when fitting the model in the anti-causal direction, and this asymmetry is useful to carry out causal inference on both synthetic and real inference problems. Our experiments also show that the transformation employed to guarantee that $\mathcal{X}$ and $\mathcal{Y}$ are equally distributed can be ignored in some cases without decreasing the performance. This indicates that our statement about the increased level of Gaussianity of the residuals, in terms of the increase of the entropy and the reduction of the high-order cumulants, may be true under more general assumptions.

7. Conclusions

In this paper we have shown that in the case of cause-effect pairs with additive non-Gaussian noise there is an asymmetry that can be used for causal inference. In particular, assuming that the cause and the effect are equally distributed random variables, that are linearly related, the residuals of a least squares fit in the anti-causal direction are more Gaussian than the residuals of a linear fit in the causal direction due a reduction of the magnitude of the high-order cumulants. Furthermore, by extending the results of Hyvärinen and Smith (2013) based on information theory, we have shown that this Gaussianization effect is also present when the two random variables are multivariate due to an increment of the

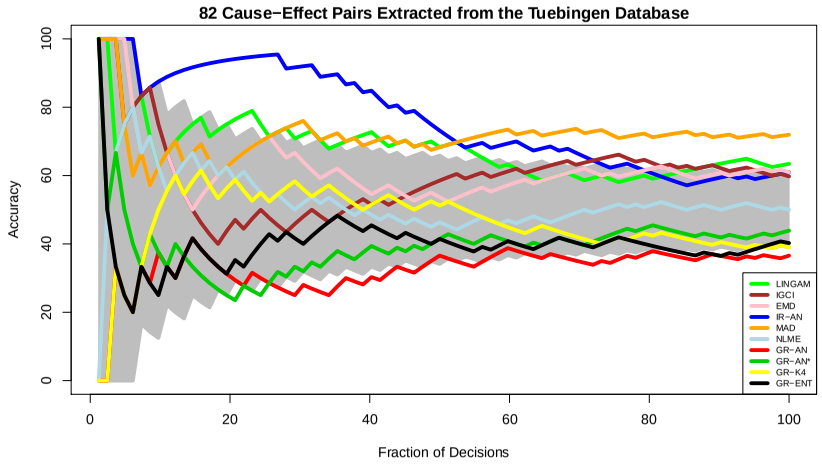


Figure 9: Accuracy of each method, as a fraction of the decisions made, on the 82 cause-effect pairs extracted from the Tuebingen database. Best seen in color.

differential entropy. This motivates the use of kernel methods to work in an expanded feature space. This enables addressing non-linear cause-effect inference problems using simple techniques. Specifically, kernel methods allow to fit a linear model in an expanded feature space which will be non-linear in the original input space.

Taking advantage of the asymmetry described, we have designed a method for non-linear causal inference, GR-AN (Gaussianity of the Residuals under Additive Noise). The method consists in computing the residuals of a linear model in an expanded feature space in both directions, *i.e.*, $\mathcal{X} \to \mathcal{Y}$ and $\mathcal{Y} \to \mathcal{X}$. The expected causal direction is the one in which the residuals appear to be more Gaussian (*i.e.*, the magnitude of the high-order cumulants is reduced and the entropy is increased). Thus, a suitable statistical test that measures the level of non-Gaussianity of the residuals can be used to determine the causal direction. In principle, one may be tempted to use statistical tests based on entropy or cumulant estimation. However, our experiments show that one can obtain better results by using a test based on an *energy distance* to quantify the Gaussianity of the residuals. In particular, entropy estimation is an arguably difficult task and the estimators of the cumulants involve high-order moments which can lead to high variance.

The effectiveness of the proposed method GR-AN has been illustrated in both synthetic and real-world causal inference problems. We have shown that in certain problems GR-AN is competitive with state-of-the-art methods and that it performs better than related methods based on entropy estimation (Hyvärinen and Smith, 2013). The entropy can be understood as a measure of non-Gaussianity. Nevertheless, it is very difficult to estimate

in practice. By contrast, the statistical test employed by GR-AR is not directly related to entropy estimation. This may explain the improvements observed. A limitation of the current formulation of GR-AN is that the distributions of the cause and the effect have to be equal. In the case of continuous univariate variables finding a transformation to make this possible is straightforward. Additionally, our experiments show that such a transformation can be side-stepped in some cases without a deterioration in performance. In any case, further research is needed to extend this analysis to remove this restriction.

Finally, the performance of GR-AN on cause-effect pairs with discretized values is rather poor. We believe this is due to the fact that in this case, finding a transformation so that the cause and the effect are equally distributed is infeasible. Furthermore, the discretization process probably has a strong impact on the Gaussianity tests. Further evidence that make these observations more plausible is the fact that discretization has also a strong negative impact in the performance of the methods based on entropy estimation.

Acknowledgments

Daniel Hernández-Lobato and Alberto Suárez gratefully acknowledge the use of the facilities of Centro de Computacin Cientfica (CCC) at Universidad Autónoma de Madrid. These authors also acknowledge financial support from the Spanish Plan Nacional I+D+i, Grants TIN2013-42351-P and TIN2015-70308-REDT, and from Comunidad de Madrid, Grant S2013/ICE-2845 CASI-CAM-CM. David Lopez-Paz acknowledges support from *Fundación la Caixa*.

Appendix A.

In this appendix we show that if $\mathcal{X}$ and $\mathcal{Y}$ follow the same distribution and they have been centered, then the determinant of the covariance matrix of the random variable corresponding to $\boldsymbol{\epsilon}_i$, denoted with $\mathrm{Cov}(\boldsymbol{\epsilon}_i)$, coincides with the determinant of the covariance matrix corresponding to the random variable $\tilde{\boldsymbol{\epsilon}}_i$, denoted with $\mathrm{Cov}(\tilde{\boldsymbol{\epsilon}}_i)$.

From the causal model, *i.e.*, $\mathbf{y}_i = \mathbf{A}\mathbf{x}_i + \boldsymbol{\epsilon}_i$, we have that:

$$\mathrm{Cov}(\mathcal{Y}) = \mathbf{A}\mathrm{Cov}(\mathcal{X})\mathbf{A}^{\mathrm{T}} + \mathrm{Cov}(\boldsymbol{\epsilon}_i)\,. \tag{32}$$

Since $\mathcal{X}$ and $\mathcal{Y}$ follow the same distribution we have that $\mathrm{Cov}(\mathcal{Y}) = \mathrm{Cov}(\mathcal{X})$. Furthermore, we know from the causal model that $\mathbf{A} = \mathrm{Cov}(\mathcal{Y}, \mathcal{X})\mathrm{Cov}(\mathcal{X})^{-1}$. Then,

$$\mathrm{Cov}(\boldsymbol{\epsilon}_i) = \mathrm{Cov}(\mathcal{X}) - \mathrm{Cov}(\mathcal{Y}, \mathcal{X})\mathrm{Cov}(\mathcal{X})^{-1}\mathrm{Cov}(\mathcal{X}, \mathcal{Y})\,. \tag{33}$$

In the case of $\tilde{\boldsymbol{\epsilon}}_i$ we know that the relation $\tilde{\boldsymbol{\epsilon}}_i = (\mathbf{I} - \tilde{\mathbf{A}}\mathbf{A})\mathbf{x}_i - \tilde{\mathbf{A}}\boldsymbol{\epsilon}_i$ must be satisfied, where $\tilde{\mathbf{A}} = \text{Cov}(\mathcal{X}, \mathcal{Y})\text{Cov}(\mathcal{Y})^{-1} = \text{Cov}(\mathcal{X}, \mathcal{Y})\text{Cov}(\mathcal{X})^{-1}$. Thus, we have that:

$$\begin{aligned}
\text{Cov}(\tilde{\boldsymbol{\epsilon}}_i) &= (\mathbf{I} - \tilde{\mathbf{A}}\mathbf{A})\text{Cov}(\mathcal{X})(\mathbf{I} - \mathbf{A}^{\text{T}}\tilde{\mathbf{A}}^{\text{T}}) + \tilde{\mathbf{A}}\text{Cov}(\boldsymbol{\epsilon}_i)\tilde{\mathbf{A}}^{\text{T}} \\
&= \text{Cov}(\mathcal{X}) - \text{Cov}(\mathcal{X})\mathbf{A}^{\text{T}}\tilde{\mathbf{A}}^{\text{T}} - \tilde{\mathbf{A}}\mathbf{A}\text{Cov}(\mathcal{X}) + \tilde{\mathbf{A}}\mathbf{A}\text{Cov}(\mathcal{X})\mathbf{A}^{\text{T}}\tilde{\mathbf{A}}^{\text{T}} \\
&\quad + \tilde{\mathbf{A}}\text{Cov}(\mathcal{X})\tilde{\mathbf{A}}^{\text{T}} - \tilde{\mathbf{A}}\text{Cov}(\mathcal{Y}, \mathcal{X})\text{Cov}(\mathcal{X})^{-1}\text{Cov}(\mathcal{X}, \mathcal{Y})\tilde{\mathbf{A}}^{\text{T}} \\
&= \text{Cov}(\mathcal{X}) - \text{Cov}(\mathcal{X})\mathbf{A}^{\text{T}}\tilde{\mathbf{A}}^{\text{T}} - \tilde{\mathbf{A}}\mathbf{A}\text{Cov}(\mathcal{X}) + \tilde{\mathbf{A}}\text{Cov}(\mathcal{X})\tilde{\mathbf{A}}^{\text{T}} \\
&= \text{Cov}(\mathcal{X}) - \text{Cov}(\mathcal{X}, \mathcal{Y})\text{Cov}\mathcal{X}^{-1}\text{Cov}(\mathcal{Y}, \mathcal{X}) - \text{Cov}(\mathcal{X}, \mathcal{Y})\text{Cov}\mathcal{X}^{-1}\text{Cov}(\mathcal{Y}, \mathcal{X}) \\
&\quad + \text{Cov}(\mathcal{X}, \mathcal{Y})\text{Cov}\mathcal{X}^{-1}\text{Cov}(\mathcal{Y}, \mathcal{X}) \\
&= \text{Cov}(\mathcal{X}) - \text{Cov}(\mathcal{X}, \mathcal{Y})\text{Cov}(\mathcal{X})^{-1}\text{Cov}(\mathcal{Y}, \mathcal{X})\,. \qquad (34)
\end{aligned}$$

By the matrix determinant theorem we have that $\text{detCov}(\tilde{\boldsymbol{\epsilon}}_i) = \text{detCov}(\boldsymbol{\epsilon}_i)$. See (Murphy, 2012, p. 117) for further details.

Appendix B.

In this Appendix we motivate that, if the distribution of the residuals is not Gaussian, but is close to Gaussian, one should also expected more Gaussian residuals in the anti-causal direction in terms of the energy distance described in Section 3.4. For simplicity we will consider the univariate case. We use the fact that the energy distance in the one-dimensional case is the squared distance between the cumulative distribution functions of the residuals and a Gaussian distribution (Székely and Rizzo, 2013). Thus,

$$\tilde{D}^2 = \int_{-\infty}^{\infty} \left[\tilde{F}(x) - \Phi(x)\right]^2 dx\,, \qquad D^2 = \int_{-\infty}^{\infty} \left[F(x) - \Phi(x)\right]^2 dx\,, \qquad (35)$$

where $\tilde{D}^2$ and D^2 are the energy distances to the Gaussian distribution in the anti-causal and the causal direction respectively; $\tilde{F}(x)$ and $F(x)$ are the c.d.f. of the residuals in the anti-causal and the causal direction, respectively; and finally, $\Phi(x)$ is the c.d.f. of a standard Gaussian.

One should expect that $\tilde{D}^2 \leq D^2$. To motivate this, we use the Gram-Charlier series and compute an expansion of $\tilde{F}(x)$ and $F(x)$ around the standard Gaussian distribution (Patel and Read, 1996). Such an expansion only converges in the case of distributions that are close to be Gaussian (see Section 17.6.6a of (Cramér, 1946) for further details). Namely,

$$\begin{aligned}
\tilde{F}(x) &= \Phi(x) - \phi(x)\left(\frac{\tilde{a}_3}{3!}H_2(x) + \frac{\tilde{a}_4}{4!}H_3(x) + \cdots\right)\,, \\
F(x) &= \Phi(x) - \phi(x)\left(\frac{a_3}{3!}H_2(x) + \frac{a_4}{4!}H_3(x) + \cdots\right)\,, \qquad (36)
\end{aligned}$$

where $\phi(x)$ is the p.d.f. of a standard Gaussian, $H_n(x)$ are Hermite polynomials and $\tilde{a}_n$ and a_n are coefficients that depend on the cumulants, *e.g.*, $a_3 = \kappa_3$, $a_4 = \kappa_4$, $\tilde{a}_3 = \tilde{\kappa}_3$, $\tilde{a}_4 = \tilde{\kappa}_4$. Note, however, that coefficients a_n and $\tilde{a}_n$ for $n > 5$ depend on combinations of

the cumulants. Using such an expansion we find:

$$\begin{aligned}
\tilde{D}^2 &= \int_{-\infty}^{\infty} \phi(x)^2 \left[-\sum_{n=3}^{\infty} \frac{\tilde{a}_n}{n!} H_{n-1}(x)\right]^2 dx \approx \int_{-\infty}^{\infty} \phi(x)^2 \left[-\sum_{n=3}^{4} \frac{\tilde{\kappa}_n}{n!} H_{n-1}(x)\right]^2 dx \\
&\approx \frac{\tilde{\kappa}_3^2}{36}\mathbb{E}[H_2(x)^2\phi(x)] + \frac{\tilde{\kappa}_4^2}{576}\mathbb{E}[H_3(x)^2\phi(x)]\,, \qquad (37)
\end{aligned}$$

where $\mathbb{E}[\cdot]$ denotes expectation with respect to a standard Gaussian and we have truncated the Gram-Charlier expansion after $n = 4$. Truncation of the Gram-Charlier expansion after $n = 4$ is a standard procedure that is often done in the ICA literature for entropy approximation. See for example Section 5.5.1 of (Hyvärinen et al., 2004). We have also used the fact that $\mathbb{E}[H_3(x)H_2(x)\phi(x)] = 0$. The same approach can be followed in the case of D^2, the energy distance in the causal direction. The consequence is that $D^2 \approx \kappa_3^2/36 \cdot \mathbb{E}[H_2(x)^2\phi(x)] + \kappa_4^2/576 \cdot \mathbb{E}[H_3(x)^2\phi(x)]$. Finally, the fact that one should expect $\tilde{D}^2 \leq D^2$ is obtained by noting that $\tilde{\kappa}_n = c_n\kappa_n$, where c_n is some constant that lies in the interval $(-1, 1)$, as indicated in Section 2.1. We expect that this result extends to the multivariate case.

Appendix C.

In this Appendix we motivate that one should expect also more Gaussian residuals in the anti-causal direction, based on a reduction of the cumulants, when the residuals in feature space are projected onto the first principal component. That is, when they are multiplied by the first eigenvector of the covariance matrix of the residuals, and scaled by the corresponding eigenvalue. Recall from Section 2.2 that these covariance matrices are $\mathbf{C} = \mathbf{I} - \mathbf{A}\mathbf{A}^{\text{T}}$ and $\tilde{\mathbf{C}} = \mathbf{I} - \mathbf{A}^{\text{T}}\mathbf{A}$, in the causal and anti-causal direction respectively. Note that both matrices have the same eigenvalues.

If $\mathbf{A}$ is symmetric we have that both $\mathbf{C}$ and $\tilde{\mathbf{C}}$ have the same matrix of eigenvectors $\mathbf{P}$. Let $\mathbf{p}_1^n$ be the first eigenvector multiplied n times using the Kronecker product. The cumulants in the anti-causal and the causal direction, after projecting the data onto the first eigenvector are $\tilde{\kappa}_n^{\text{proj}} = (\mathbf{p}_1^{\text{n}})^{\text{T}}\mathbf{M}_n\text{vect}(\tilde{\kappa}_n) = c(\mathbf{p}_1^{\text{n}})^{\text{T}}\text{vect}(\tilde{\kappa}_n)$ and $\kappa_n^{\text{proj}} = (\mathbf{p}_1^{\text{n}})^{\text{T}}\text{vect}(\kappa_n)$, respectively, where $\mathbf{M}_n$ is the matrix that relates the cumulants in the causal and the anti-causal direction (see Section 2.2) and c is one of the eigenvalues of $\mathbf{M}_n$. In particular, if $\mathbf{A}$ is symmetric, it is not difficult to show that $\mathbf{p}_1^{\text{n}}$ is one of the eigenvectors of $\mathbf{M}_n$. Furthermore, we also showed in that case that $||\mathbf{M}_n||_{\text{op}} < 1$ for $n \geq 3$ (see Section 2.2). The consequence is that $c \in (-1, 1)$, which combined with the fact that $||\mathbf{p}_1^n|| = 1$ leads to smaller cumulants in magnitude in the case of the projected residuals in the anti-causal direction.

If $\mathbf{A}$ is not symmetric we motivate that one should also expect more Gaussian residuals in the anti-causal direction due to a reduction in the magnitude of the cumulants. For this, we derive a smaller upper bound on their magnitude. This smaller upper bound is based on an argument that uses the operator norm of vectors.

Definition 2 *The operator norm of a vector* $\mathbf{w}$ *induced by the* ℓ_p *norm is* $||\mathbf{w}||_{op} = min\{c \geq 0 : ||\mathbf{w}^T\mathbf{v}||_p \leq c||\mathbf{v}||_p, \forall\mathbf{v}\}$.

The consequence is that $||\mathbf{w}||_{\text{op}} \geq ||\mathbf{w}^{\text{T}}\mathbf{v}||_p/||\mathbf{v}||_p$, $\forall\mathbf{v}$. Thus, the smallest the operator norm of $\mathbf{w}$, the smallest the expected value obtained when multiplying any vector by the

vector $\mathbf{w}$. Furthermore, it is clear that $||\mathbf{w}||_{\text{op}} = ||\mathbf{w}||_2$, in the case of the ℓ_2-norm. From the previous paragraph, in the anti-causal direction we have $||\tilde{\kappa}_n^{\text{proj}}||_2 = ||(\tilde{\mathbf{p}}_1^n)^{\text{T}}\mathbf{M}_n\text{vect}(\kappa_n)||_2$, where $\tilde{\mathbf{p}}_1$ is the first eigenvector of $\tilde{\mathbf{C}}$, while in the causal direction we have $||\kappa_n^{\text{proj}}||_2 = ||(\mathbf{p}_1^n)^{\text{T}}\text{vect}(\kappa_n)||_2$, where $\mathbf{p}_1$ is the first eigenvector of $\mathbf{C}$. Thus, because the norm of each vector $\tilde{\mathbf{p}}_1^n$ and $\mathbf{p}_1^n$ is one, we have that $||\mathbf{p}_1^n||_{\text{op}} = 1$. However, because we expect $\mathbf{M}_n$, to reduce the norm of $(\tilde{\mathbf{p}}_1^n)^{\text{T}}$, as motivated in Section 2.2, $||(\tilde{\mathbf{p}}_1^n)^{\text{T}}\mathbf{M}_n||_{\text{op}} < 1$ should follow. This is expected to lead to smaller cumulants in magnitude in the anti-causal direction.

References

J. Beirlant, E. J. Dudewicz, L. Györfi, and E. C. Van Der Meulen. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences*, 6(1):17–39, 1997.

Z. Chen, K. Zhang, and L. Chan. Nonlinear causal discovery for high dimensional data: A kernelized trace method. In *IEEE 13th International Conference on Data Mining*, pages 1003–1008, 2013.

Z. Chen, K. Zhang, L. Chan, and B. Schölkopf. Causal discovery via reproducing kernel Hilbert space embeddings. *Neural Computation*, 26(7):1484–1517, 2014.

E. A. Cornish and R. A. Fisher. Moments and cumulants in the specification of distributions. *Revue de l'Institut International de Statistique / Review of the International Statistical Institute*, 5(4):307–320, 1938.

H. Cramér. *Mathematical methods of statistics*. PhD thesis, 1946.

D. Entner and P. O. Hoyer. Estimating a causal order among groups of variables in linear models. In *Internatinal Conference on Artificial Neural Networks*, pages 84–91. 2012.

A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*, pages 585–592. 2008.

A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

J. M. Hernández-Lobato, P. Morales-Mombiela, and A. Suárez. Gaussianity measures for detecting the direction of causal time series. In *International Joint Conference on Artificial Intelligence*, pages 1318–1323, 2011.

P. O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems 21*, pages 689–696, 2009.

A. Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In *Advances in Neural Information Processing Systems 10*, pages 273–279. MIT Press, 1998.

A. Hyvärinen and S. M. Smith. Pairwise likelihood ratios for estimation of non-Gaussian structural equation models. *Journal of Machine Learning Research*, 14(1):111–152, 2013.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, 2004.

D. Janzing, P. O. Hoyer, and B. Schölkopf. Telling cause from effect based on high-dimensional observations. In *International Conference on Machine Learning*, pages 479–486, 2010.

D. Janzing, J. M. Mooij, K. Zhang, J. Lemeire, J. Zscheischler, P. Daniušis, B. Steudel, and B. Schölkopf. Information-geometric approach to inferring causal directions. *Artificial Intelligence*, 182-183:1–31, 2012.

Y. Kawahara, K. Bollen, S. Shimizu, and T. Washio. GroupLiNGAM: Linear non-Gaussian acyclic models for sets of variables. 2012. arXiv:1006.5041.

S. Kpotufe, E. Sgouritsa, D. Janzing, and B. Schölkopf. Consistency of causal inference under the additive noise model. In *International Conference on Machine Learning*, pages 478–486, 2014.

A. J. Laub. *Matrix Analysis For Scientists And Engineers*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2004. ISBN 0898715768.

J. T. Marcinkiewicz. Sur une propriété de la loi de gauss. *Mathematische Zeitschrift*, 44: 612–618, 1938.

P. McCullagh. *Tensor methods in statistics*. Chapman and Hall, 1987.

J. M. Mooij, O. Stegle, D. Janzing, K. Zhang, and B. Schölkopf. Probabilistic latent variable models for distinguishing between cause and effect. In *Advances in Neural Information Processing Systems 23*, pages 1687–1695. 2010.

P. Morales-Mombiela, D. Hernández-Lobato, and A. Suárez. Statistical tests for the detection of the arrow of time in vector autoregressive models. In *International Joint Conference on Artificial Intelligence*, 2013.

K. Murphy. *Machine Learning: a Probabilistic Perspective*. The MIT Press, 2012.

J. K. Patel and C. B. Read. *Handbook of the normal distribution*, volume 150. CRC Press, 1996.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, NY, USA, 2000. ISBN 0-521-77362-8.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2002. ISBN 0262194759.

B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In *International Conference on Artificial Neural Networks*, pages 583–588. Springer, 1997.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

H. Singh, N. Misra, V. Hnizdo, A. Fedorowicz, and E. Demchuk. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23(3-4):301–321, 2003.

L. Song, K. Fukumizu, and A. Gretton. Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *Signal Processing Magazine, IEEE*, 30:98–111, 2013.

G. J. Székely and M. L. Rizzo. A new test for multivariate normality. *Journal of Multivariate Analysis*, 93(1):58–80, 2005.

G. J. Székely and M. L. Rizzo. Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249–1272, 2013.

K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *International Conference on Uncertainty in Artificial Intelligence*, pages 647–655, 2009.

Journal of Machine Learning Research 17 (2016) 1-54 Submitted 10/14; Revised 9/15; Published 2/16

Consistent Distribution-Free K-Sample and Independence Tests for Univariate Random Variables

Ruth Heller RUHELLER@POST.TAU.AC.IL
Department of Statistics and Operations Research
Tel Aviv University
Tel Aviv 69978, Israel

Yair Heller HELLER.YAIR@GMAIL.COM

Shachar Kaufman SHACHARK@POST.TAU.AC.IL
Barak Brill BARAKBRI@MAIL.TAU.AC.IL
Malka Gorfine GORFINEM@POST.TAU.AC.IL
Department of Statistics and Operations Research
Tel Aviv University
Tel Aviv 69978, Israel

Editor: Kenji Fukumizu

Abstract

A popular approach for testing if two univariate random variables are statistically independent consists of partitioning the sample space into bins, and evaluating a test statistic on the binned data. The partition size matters, and the optimal partition size is data dependent. While for detecting simple relationships coarse partitions may be best, for detecting complex relationships a great gain in power can be achieved by considering finer partitions. We suggest novel consistent distribution-free tests that are based on summation or maximization aggregation of scores over all partitions of a fixed size. We show that our test statistics based on summation can serve as good estimators of the mutual information. Moreover, we suggest regularized tests that aggregate over all partition sizes, and prove those are consistent too. We provide polynomial-time algorithms, which are critical for computing the suggested test statistics efficiently. We show that the power of the regularized tests is excellent compared to existing tests, and almost as powerful as the tests based on the optimal (yet unknown in practice) partition size, in simulations as well as on a real data example.

Keywords: bivariate distribution, nonparametric test, statistical independence, mutual information, two-sample test, HHG R package

1. Introduction

Testing if two univariate random variables X and Y are independent of one another, given a random paired sample $(x_i, y_i)_{i=1}^N$, is a fundamental and extensively studied problem in statistics. Classical methods have focused on testing linear (Pearson's correlation coefficient) or monotone (Spearman's ρ, Kendall's τ) univariate dependence, and have little power to detect non-monotone relationships. Recently, there has been great interest in developing methods to capture complex dependencies between pairs of random variables. This interest follows from the recognition that in many modern applications, dependencies of interest may not be of simple forms, and therefore the classical methods cannot capture them. Moreover, in modern applications, thousands of variables are measured simultaneously, thus making it impossible to view the scatter-plots of all the potential pairs of variables of interest. For example, Steuer et al. (2002) searched for pairs of genes that are co-dependent, among thousands of genes measured, using the estimated mutual information as a dependence measure. Reshef et al. (2011) searched for any type of relationship, not just linear or monotone, in large data sets from global health, gene expression, major-league baseball, and the human gut microbiota. They proposed a novel criterion which generated much interest but has also been criticised for lacking power by Simon and Tibshirani (2011) and Gorfine et al. (2011), and for other theoretical grounds by Kinney and Atwal (2014).

A special important case is when X is categorical. In this case, the problem reduces to that of testing the equality of distributions, usually referred to as the K-sample problem (where K is the number of categories X can have). Jiang et al. (2014) searched for genes that are differentially expressed across two conditions (i.e., the 2-sample problem), using a novel test that has higher power over traditional methods such as Kolmogorov–Smirnov tests (Darling, 1957).

For modern applications, where all types of dependency are of interest, a desirable property for a test of independence is consistency against any alternative. A consistent test will have power increasing to one as the sample size increases, for any type of dependency between X and Y. Recently, several consistent tests of independence between univariate or multivariate random variables were proposed. Székely et al. (2007) suggested the distance covariance test statistic, that is the distance (in weighted L_2 norm) of the joint empirical characteristic function from the product of marginal characteristic functions. Gretton et al. (2008) and Gretton and Gyorfi (2010) considered a family of kernel based methods, and Sejdinovic et al. (2013) elegantly showed that the test of Székely et al. (2007) is a kernel based test with a particular choice of kernel. Heller et al. (2013) suggested a permutation test for independence between two random vectors X and Y, which uses as test statistics the sum over all pairs of points $(i, j), i \neq j$, of a score that tests for association between the two binary variables $I\{d(x_i, X) \leq d(x_i, x_j)\}$ and $I\{d(y_i, Y) \leq d(y_i, y_j)\}$, where $I(\cdot)$ is the indicator function and $d(\cdot, \cdot)$ is a distance metric, on the remaining $N-2$ sample points. Gretton and Gyorfi (2010) also considered dividing the underlying space into partitions that are refined with increasing sample size. For the K-sample problem, Székely and Rizzo (2004) suggested the energy test. This test was also proposed by Baringhaus and Franz (2004) and mentioned in Sejdinovic et al. (2013) to be related to the maximum mean discrepancy (MMD) test proposed in Gretton et al. (2007) and Gretton et al. (2012a). Harchaoui et al. (2008) adopted the kernel approach of Gretton et al. (2007) and incorporated the covariance into the test statistic by using the kernel Fisher discriminant.

The tests in the previous paragraph are not distribution-free, i.e., the null distribution of the test statistics depends on the marginal distributions of X and Y. Therefore, the significance of these

tests is typically computed by a permutation test. Another option is to try to estimate the null distribution, but this can be very difficult and can lead to over-conservative bounds. For example, Székely et al. (2007) suggested an asymptotically valid null bound for the significance which they showed was too conservative (see Sejdinovic et al. (2013) for suggestions on how to directly estimate the null distribution). The computational burden of applying the permutation tests to a large family of hypotheses may be great. For example, the yeast gene expression data set from Hughes et al. (2000) contained $N = 300$ expression levels for each of $6,325$ *Saccharomyces cerevisiae* genes. In order to test each pair of genes for co-expression, it is necessary to account for multiplicity of $M = 2 \times 10^7$ hypotheses. For the permutation tests of Heller et al. (2013) and Székely et al. (2007), the number of permutations required for deriving a p-value that is below $0.05/M$ is therefore of the order of 10^{10}. Since these test statistics are relatively costly to compute for each hypothesis, e.g., $O(N^2)$ in Székely et al. (2007), and $O(N^2 \log N)$ in Heller et al. (2013), applying them to the family of $M = 2 \times 10^7$ hypotheses is computationally very challenging, even with sophisticated re-sampling approaches such as that of Yu et al. (2011). Distribution-free tests have the advantage over non-distribution-free tests, that quantiles of the null distribution of the test statistic can be tabulated once per sample size, and repeating the test on new data for the same sample size will not require recomputing the null distribution. Therefore, the computational cost is only that of computing the test statistic for each of the hypotheses.

We note that for univariate random variables Székely and Rizzo (2009a) considered using the ranks of each random variable instead of the actual values in the test of distance covariance (Székely et al., 2007), resulting in a distribution-free test. Similarly, for the test of Heller et al. (2013) replacing data with ranks results in a distribution-free test. An earlier work by Feuerverger (1993) defined another test based on the empirical characteristic functions for univariate random variables. The test statistic of Feuerverger (1993) was based on a different distance metric of the joint empirical characteristic function from the product of marginal characteristic functions than that of Székely et al. (2007). Moreover, in Feuerverger's test the X's and Y's are first replaced by their normal scores, where the normal scores of the X's depend on the data only through their ranks among the X's, and similarly the normal scores of the Y's depend on the data only through their ranks among the Y's, thus making this test distribution-free. Pczos et al. (2012) suggested a kernel-based test which uses the MMD after transforming the data to ranks.

A popular approach for developing distribution-free tests of independence considers partitioning the sample space, and evaluating a test statistic on the binned data. A detailed review of distribution-free partition-based tests is provided in Section 1.1 for the independence problem, and in Section 1.2 for the K-sample problem. In 1.3 we describe our goals and the outline of the present paper.

1.1 Review of Distribution-Free Tests of Independence Based on Sample Space Partitions

For detecting any type of dependence between X and Y, the null hypothesis states that X and Y are independent, $H_0 : F_{XY} = F_X F_Y$, where the joint distribution of (X, Y) is denoted by F_{XY}, and the marginal distributions of X and Y, respectively, are denoted by F_X and F_Y. The alternative is that X and Y are dependent, $H_1 : F_{XY} \neq F_X F_Y$.

Figure 1 shows example partitions of the sample space based on the ranked observations, $rank(Y)$ versus $rank(X)$, where a $m \times m$ partition is based on $m - 1$ observations. We refer to such partitions as data derived partitions (DDP). The dependence in the data can be captured by many partitions, and some partitions are better than others.

Hoeffding (1948b) suggested a test based on summation of a score over all N 2×2 DDP of the sample space, which is consistent against any form of dependence if the bivariate density is continuous. Hoeffding's test statistic is

$$\iint N \left\{ \hat{F}_{XY}(x,y) - \hat{F}_X(x)\hat{F}_Y(y) \right\}^2 d\hat{F}_{XY}(x,y),$$

where $\hat{F}$ denotes the empirical cumulative distribution function. Blum et al. (1961) showed that Hoeffding's test statistic is asymptotically equivalent to $\sum_{i=1}^{N}(o^i_{1,1}o^i_{2,2} - o^i_{1,2}o^i_{2,1})^2/N^4$, where $o^i_{u,v}$, $u, v \in \{1, 2\}$, is the observed count of cell (u, v) in the 2×2 contingency table defined by the ith observation. Thas and Ottoy (2004) noted that by appropriately normalizing each term in the sum, the test statistic becomes the average of all Pearson statistics for independence applied to the contingency tables that are induced by 2×2 sample space partitions centered about observation $i \in \{1, \ldots, N\}$. They proved that the weighted version of Hoeffding's test statistic is still consistent.

Partitioning the sample space into finer partitions than the 2×2 quadrants of the classical tests, based on the observations, was also considered in Thas and Ottoy (2004). They suggested that the average of all Pearson statistics on finer partitions of fixed size $m \times m$ may improve the power, but did not provide a proof that the resulting tests are consistent. They examined in simulations only 3×3 and 4×4 partitions. Reshef et al. (2011) suggested the maximal information coefficient, which is a test statistic based on the maximum over dependence scores taken for partitions of various sizes, after normalization by the partition size, where the purpose of the normalization is equitability rather than power. Since computing the statistic exactly is often infeasible, they resort to a heuristic for selecting which partitions to include. Thus, in practice, their algorithm goes over only a small fraction of the partitions they set out to examine. In Section 4 we show that the power of this test is typically low.

1.2 Review of the K-Sample Problem

As in Section 1.1, we focus on consistent partition-based distribution-free tests. For testing equality of distributions, i.e., for a categorical X, one of the earliest and still very popular distribution-free consistent tests is the Kolmogorov–Smirnov test (Darling, 1957), which is based on the maximum score of all N partitions of the sample space based on an observed data point. Aggregation by summation over all N partitions has been considered by Cramer and von Mises (Darling, 1957), Pettitt (Pettitt, 1976) who constructed a test-statistic of the Anderson and Darling family (Anderson and Darling, 1952), and Scholz and Stephens (1987).

Thas and Ottoy (2007) suggested the following extension of the Anderson–Darling type test. For random samples of size N_1 and $N - N_1$, respectively, from two continuous densities, for a fixed m, they consider all possible partitions into m intervals of the sample space of the univariate continuous random variable. They compute Pearson's chi-square score for the observed versus expected (under the null hypothesis that the two samples come from the same distribution) counts, then aggregate by summation to get their test statistics. A permutation test is applied on the resulting test statistic, since under the null all $\binom{N}{N_1}$ assignments of the group labels are equally likely. They show that the suggested statistic for $m = 2$ is the Anderson–Darling test. They examined in simulations only partitions into $m \leq 4$ intervals.

Jiang et al. (2014) suggested a penalized score that aggregates by maximization the penalized log likelihood ratio statistic for testing equality of distributions, with a penalty for fine partitions.

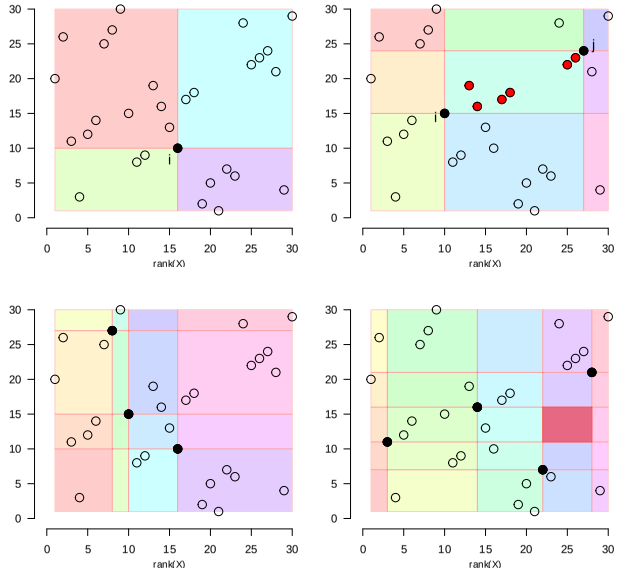


Figure 1: A visualization of the partitioning of the rank–rank plane which is at the basis of the data derived partitions (DDP) tests. Here, $N = 30$, and circles represent observed points. Full black circles represent those observations that were chosen to induce the partition, and different shades represent partition cells. With $m = 2$, all cells are corner cells (top-left); with $m = 3$, the center cell has two vertices which are observed sample points (top-right); with $m = 4$, all internal cells, i.e., cells that are not on the boundary, have at least one observed point vertex (bottom-left); only with $m \geq 5$, there exists at least one internal cell free of observed vertices (bottom-right, internal cell with no vertex which is a sample point is marked in red).

They developed an efficient dynamic programming algorithm to determine the optimal partition, and suggested a distribution-free permutation test to compute the p-value.

Although there are many additional tests for the two-sample problem, the list above contains the most common as well as the most recent developments in this field. Interestingly, when working with ranks, the energy test of Székely and Rizzo (2004) and the Cramer–von Mises test turn out to be equivalent.

1.3 Overview of This Paper

In this work, we suggest several novel distribution-free tests that are based on sample space partitions. The novelty of our approach lies in the fact that we consider aggregation of scores over all partitions of size $m \times m$ (or m for the K-sample case), where m can increase with sample size N, as well as consideration of all ms simultaneously without any assumptions on the underlying distributions. In Section 2 we present the new tests both for the independence problem and for the K-sample problem, with a focus on our regularized scores (that consider all ms) in Section 2.1. We prove that all suggested tests are consistent, including those presented in Thas and Ottoy (2004), and show the connection between our tests and mutual information (MI). In Section 3 we present innovative algorithms for the computation of the tests, which are essential for large m since the computational complexity of the naive algorithm is exponential in m. Simulations are presented in Section 4. Specifically, in Section 4.1 we show that for the two-sample problem for complex distributions there is a clear advantage for fine partitions, while for simple distributions rougher partitions have an advantage. In Section 4.2 we show that, for the independence problem, typically finer partitions have a clear advantage for complex non-monotone relationships. On the other hand, for simpler relationships, there is an advantage for rougher partitions. We further demonstrate the ability of our regularized method (which aggregates over all partitions) to adapt and find the best partition size. Moreover, in simulations we show that for complex relationships all these tests are more powerful than other existing distribution-free tests. In Section 5 we analyze the yeast gene expression data set from Hughes et al. (2000). With our distribution-free tests, we discover interesting non-linear relationships in this data set that could not have been detected by the classical tests, contrary to the conclusion in Steuer et al. (2002) that there are no non-monotone associations. Efficient implementations of all statistics and tests described herein are available in the R package *HHG*, which can be freely downloaded from the Comprehensive R Archive Network, `http://cran.r-project.org/`. Null tables can be downloaded from the first author's web site.

2. The Proposed Statistics

We assume that Y is a continuous random variable, and that X is either continuous or discrete. We have N independent realizations $(x_1, y_1), \ldots, (x_N, y_N)$ from the joint distribution of X and Y. Our test statistics will only depend on the marginal ranks, and therefore are distribution free, i.e., their null distributions are free of the marginal distributions F_X and F_Y.

Test statistics for the K-sample problem We first consider the case that X is categorical with $K \geq 2$ categories. In this case, a test of association is also a K-sample test of equality of distributions. For N observations, there are $\binom{N+1}{2}$ possible cells, and $\binom{N-1}{m-1}$ possible partitions of the observations into m cells, where a cell is an interval on the real line. Since the cell membership of

observations is the same regardless of whether the partition is defined on the original observations or on the ranked observations, and the statistics we suggest only depend on these cell memberships, we describe the proposed test statistics on the ranked observations, $rank(Y) \in \{1, ..., N\}$. Let Π_m denote the set of partitions into m cells. For any fixed partition $\mathcal{I} = \{i_1, \ldots, i_{m-1}\} \subset \{1.5, \ldots, N - 0.5\}$, $i_1 < i_2 < \ldots < i_{m-1}$, $\mathcal{C}(\mathcal{I})$ is the set of m cells defined by the partition. For a cell $C \in \mathcal{C}(\mathcal{I})$, let $o_C(g)$ and $e_C(g)$ be the observed and expected counts inside the cell for distribution $g \in \{1, \ldots, K\}$, respectively. The expected count $e_C(g)$ is the width of cell C based on ranks multiplied by N_g/N, where N_g is the total number observations from distribution g: $e_{[i_l, i_{l+1}]}(g) = (i_{l+1} - i_l) \times N_g/N$, where $l \in \{0, \ldots, m-1\}$, $i_0 = 0.5$ and $i_m = N + 0.5$. We consider either Pearson's score or the likelihood ratio score for a given cell C,

$$t_C \in \left\{ \sum_{g=1}^{K} \frac{[o_C(g) - e_C(g)]^2}{e_C(g)},\ \sum_{g=1}^{K} o_C(g) \log \frac{o_C(g)}{e_C(g)} \right\}. \tag{1}$$

For a given partition $\mathcal{I}$, the score is $T^{\mathcal{I}} = \sum_{C \in \mathcal{C}(\mathcal{I})} t_C$ (where if $t_C = \sum_{g=1}^{K} o_C(g) \log \frac{o_C(g)}{e_C(g)}$ then $T^{\mathcal{I}}$ is the likelihood ratio given the partition). Our test statistics aggregate over all partitions by summation (Cramer–von Mises-type statistics) and by maximization (Kolmogorov–Smirnov-type statistics):

$$S_m = \sum_{\mathcal{I} \in \Pi_m} T^{\mathcal{I}}, \quad M_m = \max_{\mathcal{I} \in \Pi_m} T^{\mathcal{I}}. \tag{2}$$

Tables of critical values for given sample sizes $N_1, \ldots, N_K$ can be obtained for (very) small sample sizes by generating all possible $N!/(\Pi_{g=1}^{K} N_g!)$ reassignments of ranks $\{1, \ldots, N\}$ to K groups of sizes $N_1, \ldots, N_K$ and computing the test statistic for each reassignment. The p-value is the fraction of reassignments for which the computed test statistics are at least as large as observed. When the number of possible reassignments is large, the null tables are obtained by large scale Monte Carlo simulations (we used $B = 10^6$ replicates for each given sample size $N_1, \ldots, N_K$). For each of the B reassignments selected at random from all possible reassignments, the test statistic is computed. Clearly, the B computations do not depend on the data, hence the tests based on these statistics are distribution free. Again, the p-value is the fraction of reassignments for which the computed test statistics are at least as large as the one observed, but here the fraction is computed out of the $B + 1$ assignments that include the B reassignments selected at random and the one observed assignment, see Chapter 15 in Lehmann and Romano (2005). The test based on each of these statistics is consistent:

Theorem 1 *Let Y be continuous, and X categorical with K categories. Let N_g be the total number of observations from distribution $g \in \{1, \ldots, K\}$, and $N = \sum_{g=1}^{K} N_g$. If the distribution of Y differs at a continuous density point y_0 across values of X in at least two categories, label these 1 and 2, $\lim_{N\to\infty} \frac{\min(N_1, N_2)}{N} > 0$, and m finite or $\lim_{N\to\infty} m/N = 0$, then the distribution-free permutation tests based on S_m and M_m are consistent.*

We omit the proof, since it is similar to (yet simpler than) the proof of Theorem 2 below.

Test statistics for the independence problem We now consider the case that X is continuous. For N pairs of observations, there are $\binom{N-1}{m-1} \times \binom{N-1}{m-1}$ partitions of the sample space into $m \times m$

cells, where a cell is a rectangular area in the plane. We refer to these partitions as the all derived partitions (ADP) and denote this set by Π_m^{ADP}. Since the cell membership of observations is the same regardless of whether the partition is defined on the original observations or on the ranked observations, and the statistics we suggest only depend on these cell memberships, we describe the proposed test statistics on the ranked observations, so the N pairs of observations are on the grid $\{1, \ldots, N\}^2$. For any fixed partition $\mathcal{I} = \{(i_1, j_1), \ldots, (i_{m-1}, j_{m-1})\} \subset \{1.5, \ldots, N - 0.5\}^2$, , $i_1 < i_2 < \ldots < i_{m-1}$, $j_1 < j_2 < \ldots < j_{m-1}$, $\mathcal{C}(\mathcal{I})$ is the set of $m \times m$ cells defined by the partition. For a cell $C \in \mathcal{C}(\mathcal{I})$, let o_C and e_C be the observed and expected counts inside the cell, respectively. The expected count in cell C with boundaries $[i_k, i_{k+1}] \times [j_l, j_{l+1}]$ is $e_C = (i_{k+1} - i_k) \times (j_{l+1} - j_l)/N$, where $k, l \in \{0, \ldots, m-1\}$, $i_0 = j_0 = 0.5, i_m = j_m = N + 0.5$. As with the K-sample problem, we consider either Pearson's score or the likelihood ratio score for a given cell C,

$$t_C \in \left\{ \frac{(o_C - e_C)^2}{e_C},\ o_C \log \frac{o_C}{e_C} \right\}. \tag{3}$$

For a given partition $\mathcal{I}$, the score is $T^{\mathcal{I}} = \sum_{C \in \mathcal{C}(\mathcal{I})} t_C$ (where if $t_C = o_C \log \frac{o_C}{e_C}$ then $T^{\mathcal{I}}$ is the likelihood ratio given the partition). As above, we consider as test statistics aggregation by summation and by maximization:

$$S_{m\times m}^{ADP} = \sum_{\mathcal{I} \in \Pi_m^{ADP}} T^{\mathcal{I}}, \quad M_{m\times m}^{ADP} = \max_{\mathcal{I} \in \Pi_m^{ADP}} T^{\mathcal{I}}. \tag{4}$$

We consider another test statistic based on DDP, where each set of $m - 1$ observed points in their turn define a partition (see Figure 1). This variant has a computational advantage over the ADP statistic for $m < 5$, see Remark 3.1. Since all observations have unique values, the remaining $N - (m-1)$ points are inside cells defined by the partition. There are $\binom{N}{m-1}$ partitions, denote this set of partitions by Π_m^{DDP}. As before, since the cell membership of observations is the same regardless of whether the partition is defined on the original observations or on the ranked observations, and the statistics we suggest only depend on these cell memberships, we describe the proposed test statistics on the ranked observations. For a cell $C \in \mathcal{C}(\mathcal{I})$, where $\mathcal{I} \in \Pi_m^{DDP}$, the boundaries of C are not necessarily defined by two sample points, as depicted at the bottom right panel of Figure 1. We refer to r_l and r_h as the lower and upper values of the ranks of X in C, and to s_l and s_h as the lower and upper values of the ranks of Y in C, where $r_l, r_h, s_l, s_h \in \{1, \ldots, N\}$. Let o_C and e_C be the observed and expected counts strictly inside the cell, respectively. The expected count in cell C with rank range $[r_l, r_h] \times [s_l, s_h]$ is $e_C = (r_h - r_l - 1)(s_h - s_l - 1)/[N - (m-1)]$. We consider Pearson's score or the likelihood ratio score for a given cell C, and define t_C as in (3). For a given partition $\mathcal{I}$, the score is $T^{\mathcal{I}} = \sum_{C \in \mathcal{C}(\mathcal{I})} t_C$, and similarly to (4) we define

$$S_{m\times m}^{DDP} = \sum_{\mathcal{I} \in \Pi_m^{DDP}} T^{\mathcal{I}}, \quad M_{m\times m}^{DDP} = \max_{\mathcal{I} \in \Pi_m^{DDP}} T^{\mathcal{I}}. \tag{5}$$

For each of the test statistics in (4) and (5), tables of exact critical values for a given sample size N can be obtained for small N by generating all possible $N!$ permutations of $\{1, \ldots, N\}$. For each permutation $(\pi(1), \ldots, \pi(N))$, the test statistic is computed for the reassigned pairs $(1, \pi(1)), \ldots, (N, \pi(N))$. Clearly, the computation of these null distributions does not depend on the data, hence the tests based on these statistics are distribution free. As in the case of the K-sample problem, the p-value is the fraction of permutations for which the computed test statistics

are at least as large as the one observed, and when the number of possible permutations is large, the critical values are obtained by large scale Monte Carlo simulations. The test based on each of these statistics is consistent:

Theorem 2 *Let the joint density of X and Y be $h(x,y)$, with marginal densities $f(x)$ and $g(y)$. If there exists a point (x_0,y_0) such that $h(x_0,y_0)$ is continuous and $h(x_0,y_0) \neq f(x_0)g(y_0)$, i.e., there is local dependence at a continuous density point, and if m is finite or $\lim_{N\to\infty} m/\sqrt{N} = 0$, then the distribution-free permutation tests based on the following test statistics are consistent:*

1. *The test statistics aggregated by summation: $S^{DDP}_{m\times m}$ and $S^{ADP}_{m\times m}$.*
2. *The test statistics aggregated by maximization: $M^{DDP}_{m\times m}$ and $M^{ADP}_{m\times m}$.*

A proof is given in Appendix A. For finite m, as $N\to\infty$, our test statistics converge to population measures of deviation from independence that are zero if and only if the independence hypothesis is true (see Hoeffding (1948a) for a discussion of a similar population quantity when $m=2$). For m growing with N, our test statistics estimate the mutual information, as detailed below.

We note that Thas and Ottoy suggested S_m with $t_C = \sum_{g=1}^{2} \frac{(o_C(g)-e_C(g))^2}{e_C(g)}$ in Thas and Ottoy (2007), and $S^{\mathrm{DDP}}_{m\times m}$ using Pearson's score for finite m in Thas and Ottoy (2004). However, they examined in simulations only $m \leq 4$. Thanks to the efficient algorithms we developed, detailed in Section 3, we are able to test for any $m \leq N$ in the K-sample problem, and for aggregation by summation in the test of independence. If the aggregation is by maximization in the test of independence, the algorithm, detailed in Section 3, is exponential in m and thus the computations are feasible only for $m \leq 4$.

We shall show in Section 4 that the power of the test based on a summation statistic can be different from the power of the test based on a maximization statistic, and which is more powerful depends on the joint distribution. However, for both aggregation methods, using $m > 3$ partitions improves power considerably for complex settings. Therefore, in complex settings our tests with $m > 3$ have a power advantage over the classical distribution-free tests, which focused on rough partitions, typically $m = 2$.

Connection to the MI An attractive feature of the statistics S_m and $S^{ADP}_{m\times m}$, for m large enough, is that they are directly associated with the MI ($I_{XY} = \int h(x,y)\log[h(x,y)/\{f(x)g(y)\}]dxdy$ for continuous X and Y). MI is a useful measure of statistical dependence. The variables X and Y are independent if and only if the MI is zero. Estimated MI is used in many applications to quantify the relationships between variables, see Steuer et al. (2002), Paninski (2003), Kinney and Atwal (2014) and references within. Although many works on MI estimation exist, no single one has been accepted as a state-of-the-art solution in all situations (Kinney and Atwal, 2014). A popular estimator among practitioners due to its simplicity and consistency is the *histogram* estimator, where the data are binned according to some scheme and the empirical mutual information of the resulting partition, i.e, the likelihood ratio score, is computed. Intuitively, one can expect that the statistic $S^{ADP}_{m\times m}$, properly normalized, can also serve as a consistent estimator of the mutual information, when the contingency tables are summarized by the likelihood ratio statistic, since it is the average of histogram estimators, over all partitions. This intuition is true despite the fact that the number of partitions goes to infinity, since we show that the convergence is uniform and that the fraction of "bad" partitions (i.e., partitions with cells that are too big or too small) is small, as long as m goes to infinity at a slow enough rate.

Theorem 3 *Suppose X is categorical with K categories and Y is continuous. Let N_g be the total number of observations from distribution $g \in \{1,\ldots,K\}$, and $N = \sum_{g=1}^{K} N_g$. If $\lim_{N\to\infty} \frac{N_g}{N} > 0$ for $g = 1,\ldots,K$, $\lim_{N\to\infty} \frac{m}{N} = 0$, and $\lim_{N\to\infty} m = \infty$, then $\frac{S_m}{N\binom{N-1}{m-1}}$ is a consistent estimator of the MI.*

Theorem 4 *Suppose the bivariate density of (X,Y) is continuous with bounded mutual information. If $\lim_{N\to\infty} m/\sqrt{N} = 0$, and $\lim_{N\to\infty} m = \infty$, then $\frac{S^{ADP}_{m\times m}}{N\binom{N-1}{m-1}\times\binom{N-1}{m-1}}$ is a consistent estimator of the MI.*

See Appendix B for a proof of Theorem 4. The proof of Theorem 3 is omitted since it is similar to that of Theorem 4. See Appendix D for a simulated example of MI estimation using $S^{DDP}_{m\times m}$, $S^{ADP}_{m\times m}$, and the histogram estimator. The ADP estimator is the least variable, as is intuitively expected since it is the average over many partitions.

Remark 2.1 *In this work we assume there are no ties among the continuous variables. In our software, tied data are broken randomly, so that our test remains distribution free. An alternative approach, which is no longer distribution free, is a permutation test on the ranks, with average ranks for ties. Then a tied observation, that falls on the border of a contingency table cell, receives equal weight in each of the cells it borders with.*

2.1 The Proposed Regularized Statistics

An important parameter of the statistics proposed above is m, the partition size. A poor choice of m may lead to substantial power loss: if m is too small or too large, it may lack power to discover complex non-monotone relationships. For example, consider the three simulation settings for the two-sample problem in the first row of Figure 2. The best partition for setting 1, "normal vs. normal with delta", for small sample sizes, is intuitively to divide the real line into three cells: until the start of the narrow peak, the support of the narrow peak, and after the peak ends. Moreover, the best aggregation method is by maximization, not summation, since there are very few good partitions that capture the peak and aggregation by summation using $m = 3$ will aggregate many bad partitions that miss the peak. Therefore, we expect that M_3 will be the most powerful test statistic for setting 1. For setting 2, "Mix. vs. Mix. 3 vs. 4 components", intuitively it seems best to partition into more than seven cells, and that many partitions will work well. For setting 3, "normal vs. normal with many deltas", it seems best to partition into many cells. Indeed, the power curves in Figure 3 show that for the first setting, M_m is optimal at value $m = 3$, yet if we use this value for the second setting, the test has 20% lower power than optimal power (which is 86% at $m = 10$), and if we use this value for the third setting, the test has 58% less power than the optimal power (which is 88% at $m = 34$). Since the optimal choice of m is unknown in practice, we suggest two types of regularizations which take into consideration the scores from all partition sizes.

The combined p-values statistic The first type of regularization we suggest is to combine the p-values from each m, so that the test statistic becomes the combined p-value. Specifically, let p_m be the p-value from a test statistic based on partition size m, be it S_m or M_m for the K-sample problem, or $S^{ADP}_{m\times m}$ or $S^{DDP}_{m\times m}$ for the independence problem. Due to the computational complexity, we do not consider a regularized score for $M_{m\times m}$. We consider as test statistics the minimum p-value,

$\min_{m\in\{2,\ldots,m_{\max}\}} p_m$, as well as the Fisher combined p-value, $-\sum_{m=2}^{m_{\max}} \log p_m$. These combined p-values are not p-values in themselves, but their null distribution can be easily obtained from the null distributions of the test statistics for fixed ms, as follows: (1) for each of B permutations, compute the test statistics for each $m \in \{2,\ldots,m_{\max}\}$; (2) compute the p-value of each of the resulting statistics, so for each permutation, we have a set of p-values $p_2,\ldots,p_{m_{\max}}$ to combine; (3) combine the p-values for each of the B permutations. Choose B to be large enough for the desired accuracy of approximation of the quantiles of the null distribution of the combined p-values used for testing. Obviously, since the combined p-values are based on the ranks of the data, they are distribution-free. Since they do not require fixing m in advance, they are a practical alternative to the tests that require m as input.

In order to examine how close this regularized score is to the optimal m (i.e., the m with highest power), we looked at the distribution of the ms with minimum p-values in 20,000 data samples from the above-mentioned three simulation settings. For these settings, using the aggregation by maximization statistic, the median m of the minimal p-value was: 3 for the first setting, 9 for the second setting, and 33 for the third setting. Moreover, the first and third quartiles were 3 to 5 for the first setting, 7 to 14 for the second setting, and 19 to 60 for the third setting. We conclude that for these examples, the m that achieves the minimum p-values in most runs was remarkably close to the optimal m (which was 3, 10, and 34 for settings 1,2, and 3, respectively), suggesting that the power of the minimum p-value statistic is close to that of the statistic with optimal m. Indeed, the power of the minimum p-values in settings 1-3 using aggregation by maximum was 0.825, 0.799, and 0.785, whereas the power using the (unknown in practice) optimal m in settings 1-3 was 0.894, 0.86, and 0.88, respectively. Further empirical investigations detailed in Section 4 give additional support to this regularization method.

An extensive numerical investigation, partially summarized in Appendix G, led us to choose the minimum p-value as the preferred regularization method. Between the two combining functions, we preferred the minimum over Fisher, since Fisher was far more sensitive to the choice of the range of m for combining (see Table 6). This regularized statistic is consistent, as the next theorems show.

Theorem 5 *Let Y be continuous, and X categorical. Let N_g be the total number of observations from distribution $g \in \{1,\ldots,K\}$, and $N = \sum_{g=1}^{K} N_g$. If the distribution of Y differs at a continuous density point y_0 across values of X in at least two categories, label these 1 and 2, $\lim_{N\to\infty} \frac{\min(N_1,N_2)}{N} > 0$, then the permutation test based on $\min_{m\in\{2,\ldots,m_{\max}\}} p_m$ is consistent, if:*

1. *it is based on $S_m, m \in \{2,\ldots,m_{\max}\}$, and $\lim_{N\to\infty} m_{\max}/\sqrt{N} = 0$ or $m_{\max}$ is finite.*

2. *it is based on $M_m, m \in \{2,\ldots,m_{\max}\}$ and $\lim_{N\to\infty} m_{\max}/N = 0$ or $m_{\max}$ is finite.*

Theorem 6 *Let the joint density of X and Y be $h(x,y)$, with marginal densities $f(x)$ and $g(y)$. If there exists a point (x_0,y_0) such that $h(x_0,y_0)$ is continuous and $h(x_0,y_0) \neq f(x_0)g(y_0)$, i.e., there is local dependence at a continuous density point, then the permutation test based on $\min_{m\in\{2,\ldots,m_{\max}\}} p_m$ is consistent, if*

1. *it is based on $S^{DDP}_{m\times m}$ or $S^{ADP}_{m\times m}$, and $\lim_{N\to\infty} m_{\max}/N^{1/3} = 0$ or $m_{\max}$ is finite.*

2. *it is based on $M^{DDP}_{m\times m}$ and $M^{ADP}_{m\times m}$, and $\lim_{N\to\infty} m_{\max}/\sqrt{N} = 0$ or $m_{\max}$ is finite.*

The proof of Theorem 6 follows in a straightforward way from the proofs of Theorem 2, see Appendix C for details. The proof of Theorem 5 follows similarly from the proof of Theorem 1, and it is omitted.

The penalized statistic The test statistic is the maximum (over all ms) of the statistic plus penalty. For the K-sample problem, Jiang et al. (2014) suggested assigning a prior on the partition scheme and they regularized the likelihood ratio score using this prior. Specifically, they assumed the partition size is Poisson and the conditional distribution on the m partition widths (normalized to sum to one) is $Dirichlet(1,\ldots,1)$. This led to their penalty term $-\lambda_0(\log N)(m-1)$, where $\lambda_0 > 0$ has to be fixed. We assume that the marginal distribution on the partition size is $\pi(m)$ (e.g., Poisson or Binomial), and that the prior probability of selecting $\mathcal{I}$ given m, $\pi(\mathcal{I}|m)$, is uniform. There is an important difference between our uniform discrete prior distribution on partitions of size m and the continuous Dirichlet uniform prior of Jiang et al. (2014). Our prior is uniform on all partitions that truly divide the sample space into m cells, i.e., we cannot have two partition lines between two consecutive samples, since this is actually an $m-1$ partition. Using the continuous Dirichlet prior results in practice in at most m partitions, but the partition size may also be strictly smaller than m if two partition points lie between two sample points. Therefore, their conditional distribution given the partition size parameter is not necessarily the true size of the partition. Their penalty translates to a conditional probability given a true partition size m of $\frac{(m-1)!}{(N-1)^{(m-1)}}$, compared to our $\pi(\mathcal{I}|m) = 1/\binom{N-1}{m-1}$. Therefore, their score penalizes more severely large ms, and their regularized test statistic has less power when the optimal m is large in our simulations.

For aggregation by maximum in the K-sample problem, we consider the regularized statistic,

$$\max_{m\in\{2,\ldots,m_{\max}\}} \{M_m + \log[\pi(\mathcal{I}|m)\pi(m)]\}, \tag{6}$$

where we use the likelihood ratio score per partition. Due to the computational complexity, we do not consider a regularized score for $M_{m\times m}$. For aggregation by summation, our efficient algorithms described in Section 3 enable us to consider the penalized average score per m,

$$\max_{m\in\{2,\ldots,m_{\max}\}} \{SLR_m\pi(\mathcal{I}|m) + \log\pi(m)\} \tag{7}$$

where $SLR_m\pi(\mathcal{I}|m)$ is S_m divided by the number of partitions of size m for the K-sample test, and $S^{ADP}_{m\times m}$ (or $S^{DDP}_{m\times m}$) divided by the number of partitions of size $m\times m$ for the test of independence, using the likelihood ratio score per partition. The null distribution of these regularized statistics is computed by a permutation test, and they are distribution free.

Regularization using priors was less effective than using combined p-values, except when the Poisson prior was used with parameter $\lambda = \sqrt{N}$ (see Table 7). We preferred the first type of regularization since it was at least as effective as regularizing by a Poisson prior, without requiring setting any additional parameters.

3. Efficient Algorithms

For computing the above test statistics for a given N and partition size m, the computational complexity of a naive implementation is exponential in m. We show in Section 3.1 more sophisticated algorithms for computing the aggregation by sum statistics for all $m \in \{2,\ldots,N\}$ at once that have complexity $O(N^2)$ for the K-sample problem, and $O(N^4)$ for the independence problem. This is

possible since instead of iterating over partitions, the algorithms iterate over cells. Moreover, the algorithms also enable calculating the regularized sum statistics of Section 2.1 in $O(N^2)$ and $O(N^4)$ for the K-sample and independence problems, respectively, since we just need to go over the list of m scores and for each score S_m check its p-value in our pre-calculated null tables, which requires just an additional $O(N\log(B))$, where B is the null-table size.

We show in Section 3.2 an algorithm with complexity $O(N^3)$ for the K-sample problem for computing the aggregation by maximum for all m at once. This algorithm also enables calculating the regularized maximum statistic of Section 2.1 in $O(N^3)$. The algorithms for aggregating by maximum in the independence problem are exponential in m, and therefore infeasible for modest N and $m > 4$. However, for $m = 3$ and $m = 4$ we provide efficient algorithms with $O(N^2)$ and $O(N^3)$ complexity, respectively, for the DDP test statistics.

3.1 Aggregation by Summation

The algorithms for aggregation by summation are efficient due to two key observations. First, the score per partition is a sum of contributions of individual cells, and the total number of cells is much smaller than the number of partitions (unless $m = 2$ in the K-sample problem, and $m \le 4$ when using DDP in the independence problem, see Remark 3.1 below). Therefore, we can interchange the order of summation between cells and partitions and thus achieve a big gain in computational efficiency, since it is easy to calculate in how many partitions each cell appears, see equations (8) and (11).

Second, because for a fixed m the number of partitions in which a specific cell appears depends only on the width (and for independence testing, also length) of the cell, the data-dependent computations do not depend on m: the test statistics are the sum of cell scores for every width for the K-sample test, and for every combination of width and length for the independence test, see equations (9) and (12). The complexity of the algorithm remains the same even when the scores are computed for all ms, since the complexity is determined by a preprocessing phase which is shared by all ms. Therefore, the complexity for the regularized scores is the same as the complexity for a single m.

3.1.1 Algorithm for the K-Sample Problem

For $g = 1,\dots,K$ (the categories of X) and $r = 1,\dots,N$ (the ranks of Y), we first compute in $O(N)$ A as follows:

$$A(g,r) = \sum_{i=1}^{r} I(g_i = g),$$

and let $A(g,0) = 0$. For a cell with rank range $[r_l, r_h]$, where $r_l, r_h \in \{1,\dots,N\}$, using A, the count of observations in category g that fall inside the cell can be computed in $O(1)$ operations as $o_C(g) = A(g,r_h) - A(g,r_l-1)$. Therefore, for each cell C the contribution of the cell, t_C, can be computed in $O(1)$ time.

Because the score per partition is a sum of contributions of individual cells, S_m is the sum over the score per cell, multiplied by the number of times the cell appears in a partition of size m.

Considering further summing cells of width 1 to N, we may write S_m as follows:

$$S_m = \sum_{\mathcal{I}\in\Pi_m} T^{\mathcal{I}} = \sum_{C\in\mathcal{C}} t_C \sum_{\mathcal{I}\in\Pi_m} I[C\in\mathcal{C}(\mathcal{I})] = \sum_{w=1}^{N}\sum_{C\in\mathcal{C}(w)} t_C n(w,m,C), \tag{8}$$

where $\mathcal{C}(w)$ is the collection of cells of width w and $n(w,m,C)$ is the number of partitions that include C. For computing $n(w,m,C)$, we differentiate between two possible types of cells: edge cells and internal cells. Edge cells differ from internal cells by having either $r_l = 1$ or $r_h = N$. The number of partitions of order m that include an edge cell of width $w = r_h - r_l + 1$ is given by $\binom{N-1-w}{m-2}$. The number of partitions including a similarly wide internal cell is $\binom{N-2-w}{m-3}$. Therefore, we may write S_m as follows:

$$S_m = \sum_{w=1}^{N}\binom{N-2-w}{m-3} T_i(w) + \sum_{w=1}^{N}\binom{N-1-w}{m-2} T_e(w), \tag{9}$$

where $T_i(w) = \sum_{\substack{C\in\mathcal{C}(w),\\ r_l\neq 1\cap r_h\neq N}} t_C$ and $T_e(w) = \sum_{\substack{C\in\mathcal{C}(w),\\ r_l=1\cup r_h=N}} t_C$. The algorithm proceeds as follows. First, in a preprocessing phase, we calculate $T_i(w)$ and $T_e(w)$ for all $w \in \{1,\dots,N\}$. Since t_C can be calculated in $O(1)$, as described above, the calculation of $T_i(w)$ and $T_e(w)$ for a fixed w takes $O(N)$. Since there are N values for w, we can compute and store all values of $T_i(w)$ and $T_e(w)$ in $O(N^2)$. Also in the preprocessing phase, for all $u, v \in \{0,\dots,N\}$ we calculate and store all $\binom{u}{v}$. This can be done in $O(N^2)$ using Pascal's triangle method.

Given $T_i(w), T_e(w), w = 1,\dots,N-1$ (which are independent of m!), and all $\binom{u}{v}$, we can clearly calculate S_m according to equation (9) for any m in $O(N)$ and therefore for all ms in $O(N^2)$, since $m < N$. Therefore the overall complexity of computing the scores for all ms is $O(N^2)$.

3.1.2 Algorithm for the Independence Problem

Let r_i be the rank of x_i among the observed x values, and s_i be the rank of y_i among the y values The algorithm first computes the empirical cumulative distribution in $O(N^2)$ time and space,

$$A(r,s) = \sum_{i=1}^{N} I(r_i \le r \text{ and } s_i \le s), \quad (r,s) \in \{0,1,\dots,N\}^2 \tag{10}$$

where $A(0,s) = 0, A(r,0) = 0$ and $\hat{F}(r,s) = A(r,s)/N$. First, let B be the $(N+1)\times(N+1)$ zero matrix, and initialize to one $B(r_i,s_i)$ for each observation $i = 1,\dots,N$. Next, go over the grid in s-major order, i.e., for every s go over all values of r, and compute:

1. $A(r,s) = B(r,s-1) + B(r-1,s) - B(r-1,s-1) + B(r,s)$, and
2. $B(r,s) = A(r,s)$.

We describe the algorithm for the ADP statistic, which selects partitions on the grid $\{1.5,\dots,N-0.5\}^2$ on ranked data. The main modifications for the DDP statistic are provided in Appendix E. The count of samples inside a cell with rank ranges $r \in [r_l, r_h]$ and $s \in [s_l, s_h]$ can be computed in $O(1)$ operations via the inclusion-exclusion principle:

$$o_C = A(r_h,s_h) - A(r_l-1,s_h) - A(r_h,s_l-1) + A(r_l-1,s_l-1).$$

Therefore, for each cell C the contribution of the cell t_C can be computed in $O(1)$ time. Because the score per partition is a sum of contributions of individual cells, we may write $S^{ADP}_{m\times m}$ as follows:

$$\sum_{\mathcal{I}\in\Pi^{ADP}_m} T^{\mathcal{I}} = \sum_{C\in\mathcal{C}} t_C \sum_{\mathcal{I}\in\Pi^{ADP}_m} I[C\in\mathcal{C}(\mathcal{I})] = \sum_{w=1}^{N-2}\sum_{l=1}^{N-2}\sum_{C\in\mathcal{C}(w,l)} t_C n(w,l,m,C), \qquad (11)$$

where $\mathcal{C}(w,l)$ is the collection of cells of width w and length l and $n(w,l,m,C)$ is the number of partitions that include C. As in the algorithm for the K-sample problem, $n(w,l,m,C)$ depends only on w,l,m, and whether the cell is an internal cell or an edge cell. For simplification, we discuss only the computation of the contribution of internal cells to the sum statistic, and non-internal cells can be handled similarly (as discussed in the algorithm for the K-sample problem). Therefore, our aim is to compute:

$$\sum_{w=1}^{N-2}\sum_{l=1}^{N-2} n(w,l,m)T(w,l), \qquad (12)$$

where $T(w,l)=\sum_{C\in\mathcal{C}(w,l)} t_C$ and $n(w,l,m)$ is the number of partitions that include an internal cell of width w and length l when the partition size is m and $\mathcal{C}(w,l)$ is relabelled to be the collection of internal cells of width w and length l.

The algorithm proceeds as follows. First in a preprocessing phase we perform two computations: 1) calculate and store $T(w,l)$ for all pairs $(w,l)\in\{1...N-2\}^2$. Since t_C can be calculated in $O(1)$, as described above, the calculation of $T(w,l)$ for a fixed (w,l) takes $O(N^2)$ and since there are $(N-2)^2$ pairs (w,l) the total preprocessing phase takes $O(N^4)$; 2) for all $u,v\in\{0,\ldots,N\}$ we calculate and store all $\binom{u}{v}$ in $O(N^2)$ steps using Pascal's triangle method.

Given $T(w,l)$, and all $\binom{u}{v}$, since $n(w,l,m)=\binom{N-2-w}{m-3}\binom{N-2-l}{m-3}$, we can clearly calculate equation (12) for a fixed m in $O(N^2)$ and therefore for all ms in $O(N^3)$. Due to the preprocessing phase the total complexity is $O(N^4)$.

Remark 3.1 *When it is desired to only compute the statistic for very small m, faster alternatives exist. For the two-sample problem, for $m=2$, the number of partitions is $O(N)$ and therefore an $O(N\log N)$ algorithm can be applied that aggregates over all partitions, and the complexity is dominated by the sorting of the N observations (for $m=3$, the number of partitions is already $O(N^2)$). Similarly, for the test of independence, the ADP statistic can be calculated in $O(N^2)$ steps for $m=2$, and the DDP statistic in $O(N^2)$ for $m=3$, and in $O(N^3)$ for $m=4$, since this is the order of the number of partitions. Per partition, the computation of the score for $m\le 4$ is computed in $O(1)$ time since the contribution of a cell can be computed in $O(1)$ time (as shown above). The DDP statistic for $m=2$ can be computed in $O(N\log N)$, using a similar sorting scheme as that detailed in Heller et al. (2013).*

3.2 Aggregation by Maximization

Algorithm for the K-sample problem Jiang et al. (2014) suggested an elegant and simple dynamic programming algorithm for calculating $\max_m\{M_m - m\lambda(N)\}$ for any function $\lambda(\cdot)$ in $O(N^2)$. We present a modification of their algorithm that enables us to calculate M_m for all ms in $O(N^3)$. As a first step, for all $i\le N$ and for all $j<i$ we calculate iteratively $M(i,j)$, the maximum score which partitions the first i samples into j partitions. We compute $M(i+1,j)$ from $M(a,j-1), a\le i$ using:

$$M(i+1,j)=\max_{a\in\{2,\ldots,i\}}\{M(a,j-1)+t_{[a+0.5,i+1+0.5]}\},$$

where $t_{[a+0.5,i+1+0.5]}$ is the score of the cell from $a+0.5$ to $i+1+0.5$. This calculation takes $O(N)$, and since we have $O(N^2)$ such items to calculate, this step takes $O(N^3)$. Since $M_m=M(N,m)$, the overall complexity for computing the scores for all ms is $O(N^3)$. Note that this algorithm enables us to calculate $\max_{m\in\{2,\ldots,m_{\max}\}}\{M_m+\log[\pi(\mathcal{I}|m)\pi(m)]\}$ in $O(N^3)$ for any function $\pi(m)$, thus the regularized test statistic in Section 2.1 can also be computed in $O(N^3)$.

Algorithm for the independence problem The algorithm is the same as described in Remark 3.1 for the ADP statistic for $m=2$ and for the DDP statistic for $m=3$ and $m=4$, with the difference that the aggregation is by maximization (not summation) over the scores per partition. We are not aware of a polynomial-time algorithm for arbitrary m. We discuss ways to reduce the computational complexity in Section 6.

Remark 3.2 *We show in Appendix F that for univariate data the test of Heller et al. (2013) with an arbitrary distance metric, with or without ties, can be computed in $O(N^2)$ in a similar fashion, thus improving their algorithm by a factor of $\log N$ when X and Y are univariate.*

4. Simulations

In simulations, we compared the power of our different test statistics in a wide range of scenarios. All tests were performed at the 0.05 significance level. Look-up tables of the quantiles of the null distributions of the test statistics for a given N were stored. Power was estimated by the fraction of test statistics that were at least as large as the 95th percentile of the null distribution. The null tables were based on 10^6 permutations.

The noise level was chosen separately for each configuration and sample size, so that the power is reasonable for at least some of the variants. This enables a clear comparison using a range of scenarios of interest. Since the power was very similar for the Pearson and likelihood ratio test statistics, only the results of the likelihood ratio test statistic are presented.

The simulations for the two-sample problem are detailed in 4.1, and for the independence problem in 4.2. The analysis was done with the R package *HHG*, now available on CRAN.

4.1 The Two-Sample Problem

We examined the power properties of the statistic aggregated by summation as well as by maximization for $m\in\{2,\ldots,N/2\}$, as well as the minimum p-value statistic, $\min_{m\in\{2,\ldots,m_{\max}\}} p_m$. We display here the results for $m_{\max}=149$ and $N=500$. However, the choice of $m_{\max}$ has little effect on power, see Appendix G for results with other values of $m_{\max}$. Also, see Appendix G for the results for $m_{\max}=29$ and $N=100$.

We compared these tests to six two-sample distribution-free tests suggested in the literature. We compared to Wilcoxon's rank sum test, since it is one of the most widely used tests to detect location shifts. We compared to two consistent tests suggested recently in the literature, the test of Jiang et al. (2014), referred to as DS, and the test of Heller et al. (2013) on ranks, referred to as HHG on ranks. Finally, we compared to the classical consistent tests of Kolmogorov and Smirnov,

referred to as KS, of Cramer and von Mises (which is equivalent to the energy test of Székely and Rizzo (2004) on ranks), referred to as CVM, and of Anderson and Darling, referred to as AD.

We examined the distributions depicted in Figure 2. The three scenarios in the third row were examined in Jiang et al. (2014). The remaining scenarios were chosen to have different numbers of intersections in the densities, ranging from 2 to 18, in order to examine the effect of partition size m on power when the optimal partition size increases, as well as verify that the regularized statistic has good power. The scenarios also differ by the range of support of where the differences in the distributions lie (specifically, in the first and third scenario in the first row the difference between the distributions is very local), since this makes the comparison between the two aggregation methods interesting. We considered symmetric as well as asymmetric distributions. Gaussian shift and scale setups were considered in Appendix G. Such setups are less interesting in the context of this work, because if the two distributions differ only in shift or scale then specialized tests such as Wilcoxon rank-sum for shift will be preferable, but it is important to know that the suggested tests do not break down in this case. We used 20,000 simulated data sets, in each of the configurations of Figure 2.

Table 1 and Figure 3 show the power for the setups in Figure 2. These results show that if the number of intersections of the two densities is at least four, tests statistics with $m \geq 4$ have an advantage. Since the classical competitors, KS, CVM and AD, are based on $m = 2$, they perform far worse in these setups. Moreover, although HHG and DS have better power than the classical tests, HHG is essentially an $m \leq 3$ test, and DS penalizes large ms severely, therefore their power is still too low when fine partitioning is advantageous. The minimum p-value statistic, which does not require to preset m, is remarkably efficient: in Figure 2 we see that in all settings considered, it is close to the power of the optimal m.

The choice of aggregation by maximization versus summation depends on how local the differences are between the distributions. In Figure 3 we see clearly that when the differences are in very local areas, maximization achieves the greatest power and the test based on minimum p-value has more power if the aggregation is by maximization rather than by summation (setups 1, 2, and 13), and aggregation by summation is better otherwise. Note that the optimal m for aggregation by summation is always larger than for aggregation by maximization. The reason is that in order to have a powerful statistic aggregated by maximization, it is enough to have one good partition (i.e., contain cells where the distributions clearly differ) for a fixed m, whereas by summation it is necessary to have a large fraction of good partitions among all partitions of size m.

4.2 The Independence Problem

We examined the power properties of the ADP and DDP statistics, aggregated by summation for $m \in \{1, \ldots, \sqrt{N}\}$, aggregated by maximization for $m \leq 4$, as well as the minimum p-value statistic $\min_{\{m \in 2,..,m_{max}\}} p_m$ based on aggregation by summation. We display here the results for $m_{\max} = 10$ and $N = 100$.

We compared these tests to seven tests of independence suggested in the literature. We compared to Spearman's ρ, since it is perhaps the most widely used test to detect monotone associations. We also compared to previous tests suggested in the literature with the same two important properties as our suggested tests, namely proven consistency and distribution-freeness, as well as an available implementation: the test of Hoeffding (1948b), referred to as Hoeffding; the tests of Székely et al. (2007) and Heller et al. (2013) that first transform the observations of each variable into ranks, referred to as dCov and HHG, respectively. We also compared to the test of Reshef et al. (2011),

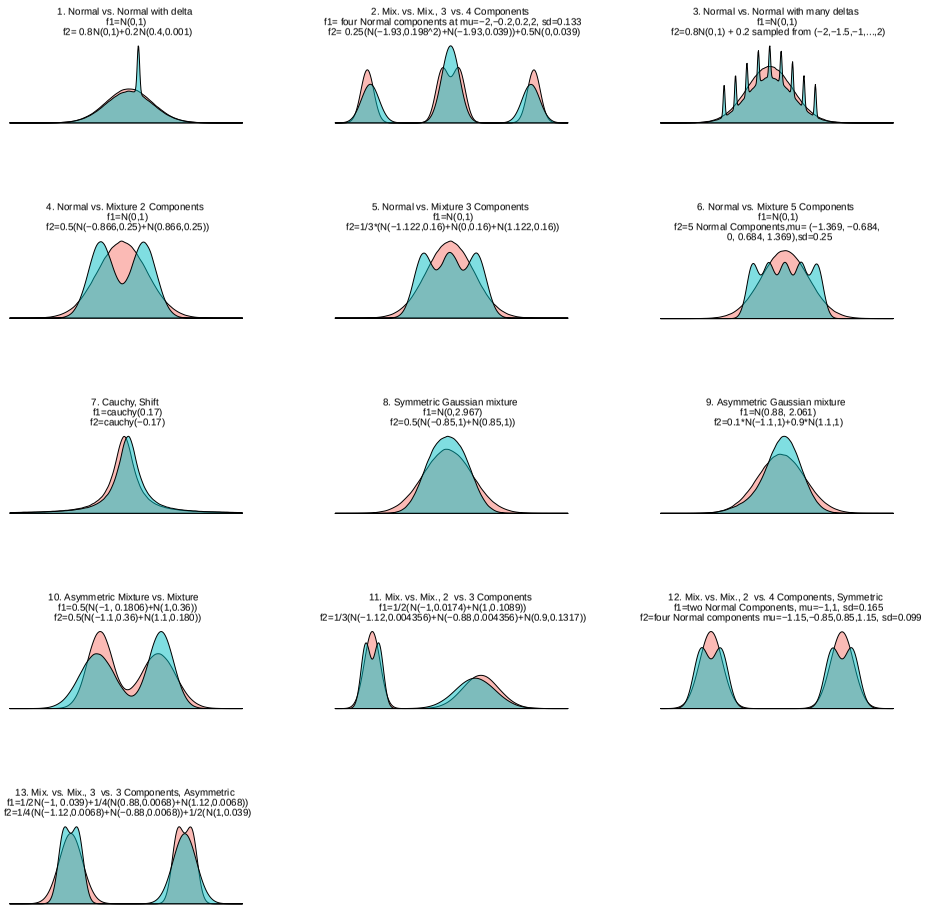


Figure 2: The two-sample problem in 13 different setups considered for $N = 500$, which differ in the number of intersections of the densities, the range of support where the differences lie, and whether they are symmetric or not.

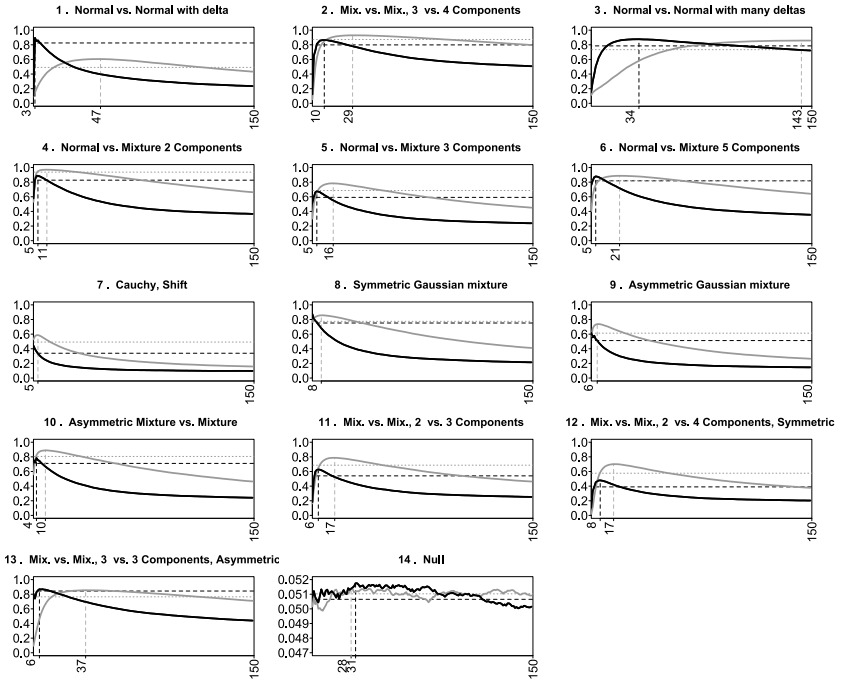


Figure 3: Estimated power with $N = 500$ sample points for the M_m (black) and S_m (grey) statistics for $m \in \{2, \ldots, 149\}$ for the setups of Figure 2. The score per partition was the likelihood ratio test statistic. The power of the minimum p-value is the horizontal dashed black line when it combines the p-values based on M_m, and the horizontal dotted grey line when it combines the p-values based on S_m. The vertical lines show the optimal m for M_m (black) and S_m (grey).

	Setup	Min. p-value aggregation by Max	by Sum	Wilcox	KS	CVM	AD	HHG	DS
1	Normal vs. Normal with delta	0.825	0.491	0.072	0.149	0.108	0.099	0.175	<u>0.849</u>
2	Mix. Vs. Mix., 3 Vs. 4 Components	0.799	<u>0.873</u>	0.000	0.020	0.001	0.021	0.344	0.560
3	Normal vs. Normal with many deltas	<u>0.785</u>	0.733	0.051	0.078	0.073	0.099	0.142	0.245
4	Normal vs. Mixture 2 Components	0.827	<u>0.937</u>	0.053	0.531	0.458	0.495	0.855	0.796
5	Normal vs. Mixture 5 Components	0.592	<u>0.686</u>	0.048	0.238	0.179	0.246	0.484	0.556
6	Normal vs. Mixture 10 Components	0.818	<u>0.820</u>	0.048	0.240	0.211	0.310	0.561	0.789
7	Cauchy, Shift	0.339	0.492	0.542	0.620	0.627	0.577	<u>0.641</u>	0.436
8	Symmetric Gaussian mixture	0.752	0.775	0.033	0.194	0.242	0.617	0.749	<u>0.835</u>
9	Asymmetric Gaussian mixture	0.512	0.613	0.050	0.253	0.277	0.469	<u>0.678</u>	0.599
10	Asymmetric Mixture vs. Mixture	0.711	<u>0.806</u>	0.000	0.159	0.119	0.395	0.690	0.747
11	Mix. Vs. Mix., 2 Vs. 3 Components	0.540	<u>0.686</u>	0.004	0.093	0.057	0.116	0.302	0.440
12	Mix. Vs. Mix., 2 Vs. 4 Comp., Sym.	0.390	<u>0.577</u>	0.000	0.005	0.000	0.005	0.079	0.270
13	Mix. Vs. Mix., 3 Vs. 3 Comp., Asym.	<u>0.844</u>	0.764	0.000	0.001	0.000	0.013	0.042	0.780
14	Null	0.051	0.051	0.050	0.043	0.050	0.050	0.050	0.050

Table 1: Power of competitors (columns 4-9), along with the minimum p-value statistic using the M_m p-values (column 2) and the S_m p-values (column 3), for $N = 500$. The score per partition was the likelihood ratio test statistic. The standard error was at most 0.0035. The advantage of the test based on the minimum p-value is large when the number of intersections of the two densities is at least four (setups 2,3,4,5,6,10,11,12, and 13). The best competitors are HHG and DS, but HHG is essentially an $m \leq 3$ test, and DS penalizes large ms severely, therefore in setups where $m \geq 4$ partitions are better they can perform poorly. Among the two variants in columns 2 and 3, the better choice clearly depends on the range of support in which the differences in distributions occur: aggregation by maximum has better power when the difference between the distributions is very local (setups 1, 3, and 13), and aggregation by summation has better power otherwise. The highest power per row is underlined.

referred to as MIC, which is not consistent due to the computational shortcuts they have to use. We note that the power of the original dCov and HHG was fairly similar to the power of their distribution-free variants, see Appendix I.

We examine complex bivariate relationships depicted in Figure 4. Most of these scenarios were collected from the literature illustrating the performance of other methods. Specifically, the first two rows were examined in Newton (2009), the next two rows are similar to the relationships examined in Reshef et al. (2011), and the Heavisine and Doppler examples in the last row were used extensively in the literature on denoising, see e.g., Donoho and Johnstone (1995). In all but the 4 Independent Clouds setup, there is dependence. The 4 Independent Clouds setup allows us to verify that the tests maintain the nominal level. We used 2000 simulated data sets for $N = 100$ and $N = 300$, in each of the configurations of Figure 4. Monotone setups are presented in Appendix H in Figure 12. Monotone setups are less interesting in the context of this work, because if there is reason to believe that the dependence is monotone, specialized tests such as Spearman's ρ or Kendall's τ will be preferable, but it is important to know that the suggested tests have reasonable power, as demonstrated in the results in Appendix H, Figure 13 and Table 10.

Tables 2 and 3, and Figure 5 show the power for the settings depicted in Figure 4. We only considered the test based on the DDP minimum p-value statistic in Tables 2 and 3, since for the

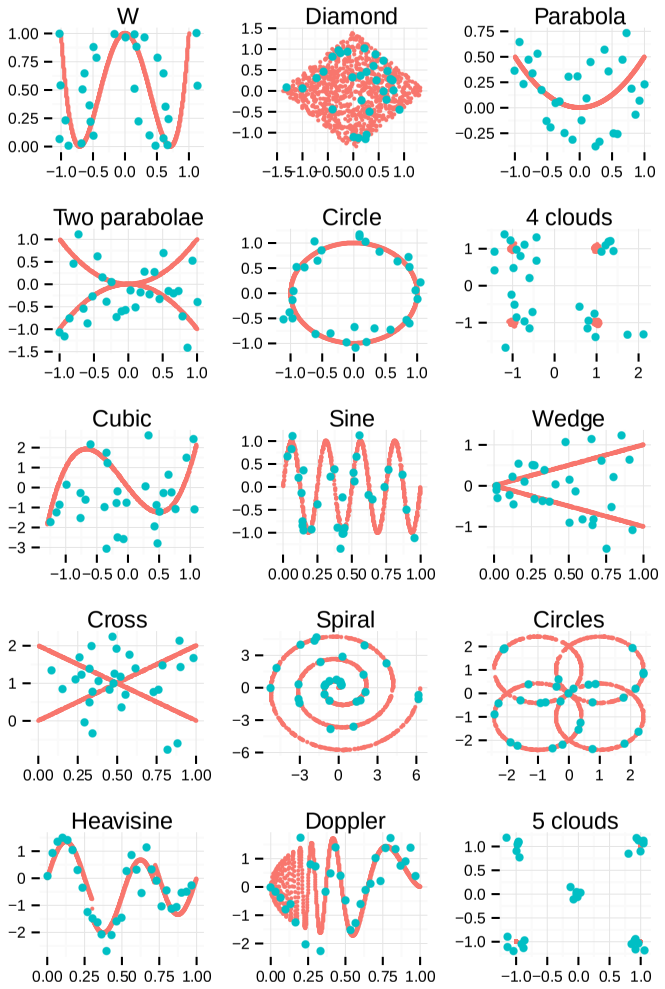


Figure 4: Bivariate relationships (in red), along with a sample $N = 100$ noisy observations (in blue). The "four clouds" relationship is a null relationship, where the two random variables are independent.

Setup	$\min_{\{m\in 2,\ldots,10\}} p_m$	Spearman	Hoeffding	MIC	dCov	HHG
W	0.655	0.000	0.414	0.526	0.361	<u>0.798</u>
Diamond	0.919	0.013	0.116	0.074	0.074	<u>0.965</u>
Parabola	<u>0.847</u>	0.028	0.413	0.211	0.386	0.784
2Parabolas	<u>0.844</u>	0.095	0.135	0.048	0.124	0.723
Circle	<u>0.886</u>	0.000	0.033	0.046	0.002	0.850
Cubic	0.731	0.344	0.655	0.515	0.627	<u>0.768</u>
Sine	0.995	0.368	0.494	<u>1.000</u>	0.415	0.804
Wedge	0.595	0.064	0.360	0.303	0.338	<u>0.673</u>
Cross	0.704	0.089	0.160	0.069	0.130	<u>0.706</u>
Spiral	<u>0.949</u>	0.112	0.141	0.251	0.140	0.337
Circles	<u>0.999</u>	0.048	0.084	0.093	0.061	0.354
Heavisine	<u>0.710</u>	0.396	0.493	0.532	0.492	0.585
Doppler	0.949	0.513	0.784	<u>0.975</u>	0.744	0.912
5Clouds	<u>0.996</u>	0.000	0.000	0.561	0.004	0.904
4Clouds	0.051	0.050	0.057	0.050	0.051	0.050

Table 2: Power of competitors (columns 3–7), along with the DDP minimum p-value statistic for $N = 100$. The standard error is at most 0.011. The score per partition was the likelihood ratio test statistic. The DDP minimum p-value statistic performs very well in comparison to the other tests. Although the competitors may have greater power in some examples, the advantage is usually small. By far the best competitor is HHG, yet it has a disadvantage when the relationship is more complex, thus benefiting from the finer partition of the minimum p-value test, especially in the Sine, Heavisine, Spiral and Circles examples. The highest power per row is underlined.

minimum p-value statistic the tests of ADP and DDP are almost identical. These results provide strong evidence that for non-monotone noisy dependencies our tests have excellent power properties. Specifically, $S^{DDP}_{m\times m}$ with $m \in \{3,\ldots,10\}$ is more powerful than all other tests in Table 2 in most settings. For example, it had greater power than all competitors in 9 settings with $m = 4$ and in 11 settings with $m = 5$, out of the 14 non-null settings. The test based on the minimum p-value has greater power than all competitors in 7 settings, and it is very close to the best competitor in most of the other settings. The MIC is best for the Sine example but performs poorly in all other examples. The minimum p-value statistic is a close second best in the Sine example, with a difference of only 0.005 from MIC, yet all other tests are more than 0.19 below MIC in power. Overall, the HHG test is the best competitor, but its power is lower than that of the minimum p-value statistic when the optimal m is greater than 4. Table 3 shows that when aggregating by maximization, the choice of m matters and the power is higher for $m > 2$. However, the minimum p-value statistic, which is aggregated by summation and considers finer partitions, is more powerful for most settings, and is a close second in the remaining settings.

5. Application to Real Data

We examine the co-dependence between pairs of genes on chromosome 1 in the yeast gene expression data set from Hughes et al. (2000), which contained $N = 300$ expression levels. After removing genes with missing values, we had 94 genes and a family of $\binom{94}{2} = 4371$ pairs to examine simultaneously. Each pair was tested for independence by the tests of Spearman, Hoeffding,

Setup	$\min_{\{m\in 2,\ldots,10\}} p_m$	$M^{DDP}_{2\times 2}$	$M^{DDP}_{3\times 3}$	$M^{DDP}_{4\times 4}$	$M^{ADP}_{2\times 2}$
W	0.655	0.190	0.637	0.574	0.155
Diamond	0.919	0.272	0.931	0.912	0.247
Parabola	0.847	0.533	0.855	0.803	0.597
2Parabolas	0.844	0.466	0.907	0.897	0.578
Circle	0.886	0.222	0.880	0.884	0.170
Cubic	0.731	0.496	0.654	0.653	0.496
Sine	0.995	0.768	0.958	0.998	0.774
Wedge	0.595	0.410	0.536	0.478	0.455
Cross	0.704	0.268	0.680	0.673	0.341
Spiral	0.949	0.116	0.489	0.764	0.189
Circles	0.999	0.085	0.606	0.844	0.088
Heavisine	0.710	0.519	0.642	0.692	0.534
Doppler	0.949	0.828	0.969	0.977	0.833
5Clouds	0.996	0.062	0.999	0.999	0.076
4Clouds	0.051	0.052	0.051	0.051	0.052

Table 3: The power of different variants aggregated by maximization (columns 3–6), along with the DDP minimum p-value statistic (column 2) for $N = 100$. The standard error is at most 0.011. The score per partition was the likelihood ratio test statistic. Although maximization is better than summation in some examples, the advantage is usually small. The advantage of the minimum p-value statistic, which is based on aggregation by summation, is large in the Cubic, Cross, Spiral, and Circles relationships. In most examples, power increases with m. The power differences between the ADP and DDP variants are small. We present only the maximum variants that take at most $O(N^3)$ to compute, therefore for ADP only results with $m = 2$ are presented. The highest power per row is underlined.

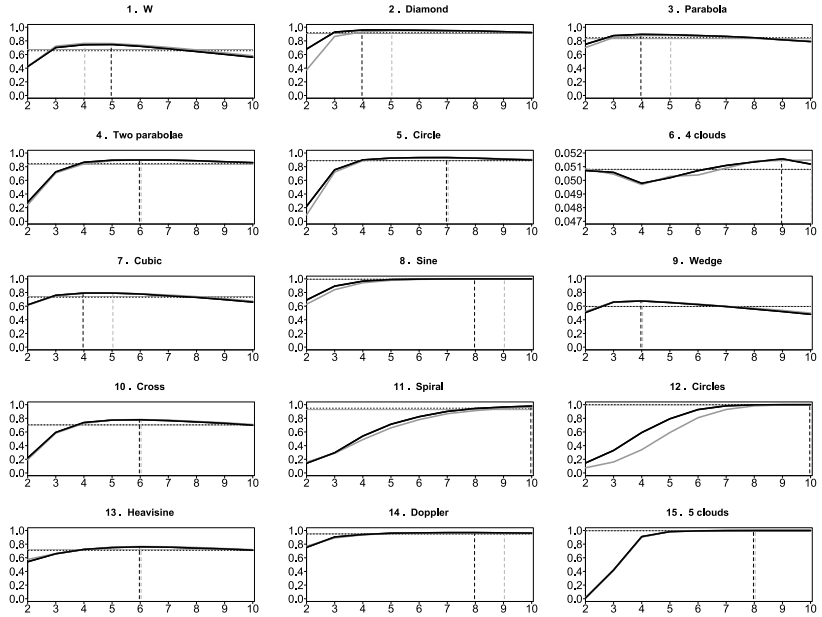


Figure 5: Estimated power as a function of partition size m, with $N = 100$ sample points, for the DDP (black) and ADP (grey) summation variants using the likelihood ratio score for the setups of Figure 4. The score per partition was the likelihood ratio test statistic. For DDP (black) and ADP (grey), the horizontal dashed line is the power of the minimum p-value statistic, and the vertical lines is the optimal m.

Test	Number of rejections	Number of intersections
Spearman	2488	2445
MIC	245	245
Hoeffding	2890	2844
HHG on ranks	3283	3199
dCov on ranks	2889	2845
minimum p-value based on ADP	3310	3294

Table 4: Benjamini–Hochberg rejections at level 0.05 for the gene expression problem of Hughes et al. (2000). For different test statistics (rows), the number of rejections (column 2), and their intersection with the rejections using the minimum p-value statistic on DDP (column 3). The minimum p-value statistic on DDP had the highest number of rejections, 3312.

MIC, dCov and HHG on ranks, as well as by our new tests with m ranging from 2 to $m_{\max} = 17$. The null tables were based on $20,000$ permutations for $N = 300$. The adjusted p-values from the Benjamini–Hochberg procedure (Benjamini and Hochberg, 1995) were computed for each test statistic.

Table 4 shows the pairwise agreements between the Benjamini–Hochberg procedure at level 0.05 using the different test statistics considered in each row, with the minimum p-value statistic based on DDP. Clearly, a large number of pairwise associations are missed when testing is performed with Spearman's ρ compared to the minimum p-value statistic, and only a small number of gene pairs detected with Spearman are missed by the minimum p-value statistic (row 1 in Table 4). These findings contradict an earlier examination of the data. Steuer et al. (2002) concluded that the most widely used approach for pairwise association testing, namely Spearman correlation, performs equivalently to a mutual information based testing approach. The authors speculated that actual dependencies, if any, are linear. The number of discoveries using MIC, Hoeffding, and dCov are much smaller than using the minimum p-value statistic. HHG on ranks also discovers less co-dependencies compared with the minimum p-value statistic. The agreement between the tests based on DDP and ADP was very high, as seen in the last row of Table 4 and in Figure 6, which shows the number of rejections for $S^{DDP}_{m\times m}$ and $S^{ADP}_{m\times m}$ for $m = 2,\ldots,17$. We conclude that in this data set there are many nonlinear associations, but powerful tests are necessary in order to detect such associations in light of the large number of simultaneous tests that have to be carried out, and that the suggested tests can be valuable tools for this task.

Note that the data had ties due to low precision of the documented expression levels. Ties were broken randomly, see remark 2.1. Repeated analysis with different seeds provided similar results.

6. Discussion

In this paper we proposed new partition-based test statistics for both the independence problem and the two-sample problem. We proved that the statistics are consistent for general alternatives and demonstrated in simulations that the power advantage of the tests based on finer partitions can be great. We further showed that the power of our regularized statistics is very close to that of the statistics based on the optimal partition size. We recommend the test using the minimum p-value statistic based on aggregation by summation, unless the alternative is suspected to be of very local nature. Specifically, in the K-sample problem if the difference between the distributions is on a

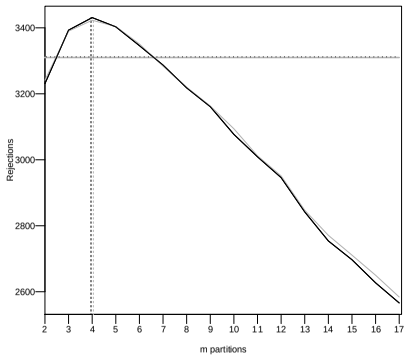


Figure 6: Number of discoveries of genes with associated expression patterns in the data of Hughes et al. (2000), by the Benjamini–Hochberg procedure at level 0.05 using $S^{DDP}_{m\times m}$ (black) and $S^{ADP}_{m\times m}$ (gray) for $m = 2,\ldots,17$. In addition, number of rejections using the minimum p-value statistic with $m_{\max} = 17$, using the DDP (black horizontal line) and the ADP (gray horizontal line).

very small range of the support, then the aggregation by maximization is preferred over aggregation by summation.

The algorithms described in Section 3.1 for the test of independence based on regularized scores for a range of $m \times m$ partitions can easily be generalized to include $m_x \times m_y$ partitions, where $m_x \neq m_y$, with the same complexity for the ADP statistic (for the DDP statistic $m_x = m_y$). Considering unequal partition sizes for X and Y is expected to improve power when the (unknown) optimal partition has an m_x value very different than the m_y value. Moreover, when the (unknown) optimal partition has $m_x \approx m_y$, the power loss from considering the minimum p-value over all $m_x \times m_y$ values instead of over all $m \times m$ values is expected to be small.

The algorithms we suggested for the K-sample problem are $O(N^2)$ and therefore are feasible even for large N. For the test of independence, even though the complexity of our suggested algorithms is $O(N^4)$, for small N these algorithms can be quite efficient in the following quite common multiple testing setting in modern studies. If M hypotheses are simultaneously examined with the same sample size, then the computational complexity of using our distribution-free tests is $O(M \times N^4)$ if null table is available for this N, or $\max\{O(M \times N^4), O(B \times N^4)\}$ if the null table is generated by the user using B Monte-Carlo replicates for the sample size N. If we needed to recompute the null distribution for every one of the M hypotheses (as required for permutation tests that are not distribution-free, such as dCov and HHG), then the computational complexity would have been $O(M \times B \times N^4)$, which may be infeasible in modern studies where the number of hypotheses tests simultaneously examined can be several thousands or hundreds of thousands. Since the null distribution needs to be generated only once in order to compute the significance

of all the M test statistics, due to the distribution-free property of our tests, they can be feasible with today's computing power even for a few thousands samples. However, computing $O(N^4)$ test statistics is unfeasible for larger sample sizes. To reduce the computational complexity when N is large, statistics which do not go over all partitions but rather just over a representative sample can be considered. This approach was used for example in Jiang (2014) for the K-sample problem. A simple way of doing this is to divide the data into $\sqrt{N} \times \sqrt{N}$ bins and only consider partitions that do not break up these bins. We expect such statistics to also be consistent and the algorithms that accompany them to be computable in $O(N^2)$.

If one expects relatively simple dependence structures, for large N, the $S_{3\times3}^{DDP}$ is recommended, since it is both distribution-free and computable in $O(N^2)$ (see Remark 3.1). In our simulations it was as powerful as HHG and more powerful than dCov, and it has the advantage of being distribution-free.

We focused our attention mainly on comparisons of distribution-free tests. We provided in Appendix I a comparison of our recommended tests with the non distribution-free HHG test of Heller et al. (2013) and the dCov test of Székely and Rizzo (2009b). These tests had less power than our novel distribution-free tests when there is advantage to finer partitions. It is of interest to examine how the extended α-distance dependence measure suggested in Székely and Rizzo (2009b) perform in these complex relationships. It is also of interest to compare to the non distribution-free kernel methods of Gretton et al. (2007) and Gretton et al. (2008). For this purpose, care must be taken in the choice of kernel bandwidth, see Gretton et al. (2012b).

In this work we restricted our attention to testing an independent sample from the joint distribution. If the paired observations are not independent, but are independent within blocks, we can easily modify our test by restricting the permutation tests to be within the blocks. When there is temporal dependence, an interesting work by Chwialkowski et al. (2014) suggested a bootstrap based approach for kernel based tests. How to modify our test in the face of temporal dependence is an open question.

A thorough investigation of the suggested mutual information estimator in Section 2 was outside the scope of this paper, but is of interest for future research. We suspect the asymptotic distribution of our mutual information estimator has a simple form. The bias of the estimator can be dealt with by modifying our estimator, and our algorithms accordingly, to only include partitions with cells of a minimum size, and by bias correction methods suggested in the literature, e.g., Vu et al. (2007). Although in this work we limited ourselves to a theoretical examination of the ADP summation statistic for mutual information estimation, we recognize that an estimator based only on the DDP may be useful, and we plan to explore it in the future.

Acknowledgments

R. Heller and B. Brill are supported by Israel Science Foundation grant 2012896. S. Kaufman is supported by a fellowship from the Edmond J. Safra Center for Bioinformatics at Tel Aviv University.

Appendix A. Proof of Theorem 2

The proof for the DDP and ADP tests are given in Sections A.1 and A.2 respectively.

A.1 The DDP Test

Proof Denote $S_{m\times m} = S_{m\times m}^{DDP}$. For simplicity, we show the proof using Pearson's test statistic. The proof using the likelihood ratio test statistic is very similar and therefore omitted. We want to show that for an arbitrary fixed $\alpha \in (0,1)$, if H_0 is false, then $\lim_{N\to\infty} Pr(S_{m\times m} > S_{1-\alpha}^{tab}) = 1$, where $S_{1-\alpha}^{tab}$ denotes the $1-\alpha$ quantile of the null distribution of $S_{m\times m}$.

If H_0 is false, then without loss of generality $h(x_0,y_0) > f(x_0)g(y_0)$. Moreover, there exists a distance $R > 0$ such that $h(x,y) > f(x)g(y)$ for all points (x,y) in the set $\mathcal{A} = \{(x,y) : x_0 \le x \le x_0 + R, y_0 \le y \le y_0 + R\}$. The set $\mathcal{A}$ has positive probability, and moreover

$$\min_{\mathcal{A}}[h(x,y) - f(x)g(y)] > 0.$$

Denote this minimum by $c > 0$. Clearly, the following two subsets of $\mathcal{A}$ have positive probability as well:

$$\mathcal{A}_1 = \{(x,y) : x_0 \le x \le x_0 + R/4, y_0 \le y \le y_0 + R/4\}$$
$$\mathcal{A}_2 = \{(x,y) : x_0 + 3R/4 \le x \le x_0 + R, y_0 + 3R/4 \le y \le y_0 + R\}.$$

Denote the probabilities of $\mathcal{A}_1$ and $\mathcal{A}_2$ by f_1 and f_2, respectively.
Let $\Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$ be the set of partitions of size m based on at least one sample point in $\mathcal{A}_1$ and on at least one sample point in $\mathcal{A}_2$. Let N_i denote the number of sample points in $\mathcal{A}_i, i \in \{1,2\}$. Let $\mathcal{I} \in \Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$ be such a (arbitrary fixed) partition. So for $\mathcal{I}$ there exists $(i,j) \subseteq \mathcal{I}$ such that $(x_i,y_i) \in \mathcal{A}_1$ and $(x_j,y_j) \in \mathcal{A}_2$. Consider the cell C defined by the two points (i,j).

The fraction of observed counts in the cell C is a linear combination of empirical cumulative distribution functions

$$\frac{o_C}{N-(m-1)} = \hat{F}_{XY}(x_i,y_i) + \hat{F}_{XY}(x_j,y_j) - \hat{F}_{XY}(x_i,y_j) - \hat{F}_{XY}(x_j,y_i),$$

and the expected fraction under the null is a function of the marginal cumulative distributions

$$\frac{e_C}{N-(m-1)} = \{\hat{F}_X(x_j) - \hat{F}_X(x_i)\}\{\hat{F}_Y(y_j) - \hat{F}_Y(y_i)\}.$$

where $\hat{F}$ denotes the empirical distribution function based on $N-(m-1)$ sample points.

By the Glivenko-Cantelli theorem, uniformly almost surely,

$$\lim_{N\to\infty}\left(\frac{o_C}{N-(m-1)} - \int_{\{(x,y):x\in(x_i,x_j],y\in(y_i,y_j]\}} h(x,y)dxdy\right) = 0,$$
$$\lim_{N\to\infty}\left\{\frac{e_C}{N-(m-1)} - \left(\int_{\{x:x\in(x_i,x_j]\}} f(x)dx\right)\left(\int_{\{y:y\in(y_i,y_j]\}} g(y)dy\right)\right\} = 0. \tag{13}$$

It is important that the convergence be uniform in order to obtain a positive lower bound on the average over all partitions, see the discussion before equation (18).

Therefore, by Slutsky's theorem and the continuous mapping theorem, we have that uniformly almost surely

$$\begin{aligned}
&\lim_{N\to\infty} \frac{1}{N-(m-1)} \frac{(o_C - e_C)^2}{e_C} = \lim_{N\to\infty} \frac{\left(\frac{o_C}{N-(m-1)} - \frac{e_C}{N-(m-1)}\right)^2}{\frac{e_C}{N-(m-1)}} \\
&\geq \lim_{N\to\infty} \left(\frac{o_C}{N-(m-1)} - \frac{e_C}{N-(m-1)}\right)^2 \\
&= \lim_{N\to\infty} \left[\int_{\{(x,y):x\in(x_i,x_j],y\in(y_i,y_j]\}} \{h(x,y) - f(x)g(y)\}dxdy\right]^2, \qquad (14)
\end{aligned}$$

where the inequality follows from the fact that $\frac{e_C}{N-(m-1)} \leq 1$.

We shall show that this limit can be bounded from below by a positive constant that depends on (x_0, y_0) and R but not on $\mathcal{I}$. Since

$$\begin{aligned}
&\{(x,y) : x \in (x_0 + R/4, x_0 + 3R/4], y \in (y_0 + R/4, y_0 + 3R/4]\} \\
&\subseteq \{(x,y) : x \in (x_i, x_j], y \in (y_i, y_j]\},
\end{aligned}$$

a positive lower bound on expression (14) can be obtained:

$$\begin{aligned}
&\lim_{N\to\infty} \left[\int_{\{(x,y):x\in(x_i,x_j],y\in(y_i,y_j]\}} \{h(x,y) - f(x)g(y)\}dxdy\right]^2 \\
&\geq \left[\int_{\{(x,y):x\in(x_0+R/4,x_0+3R/4],y\in(y_0+R/4,y_0+3R/4]\}} \{h(x,y) - f(x)g(y)\}dxdy\right]^2 \\
&\geq c^2 \int_{\{(x,y):x\in(x_0+R/4,x_0+3R/4],y\in(y_0+R/4,y_0+3R/4]\}} dxdy = c^2R^2/4, \qquad (15)
\end{aligned}$$

where the first inequality follows since $h(x,y) - f(x)g(y) > 0$ in $\mathcal{A}$, and the second inequality follows since the minimum value is $c > 0$. Therefore, it follows that $\frac{1}{N-m+1}\frac{(o_C-e_C)^2}{e_C}$ converges uniformly almost surely to a positive constant greater than $c' = c^2R^2/4$,

$$Pr\left(\lim_{N\to\infty} \frac{1}{N-m+1}\frac{(o_C - e_C)^2}{e_C} \geq c'\right) = 1. \qquad (16)$$

The partition $\mathcal{I}$ either contains the cell C, or a group of cells that divide C. By Jensen's inequality, it follows that if the partition $\mathcal{I}$ contains a group of cells that divide C, the score is made larger, since for any partition of the cell C, $C = \cup_l C_l$,

$$\left(\frac{o_C - e_C}{e_C}\right)^2 = \left(\sum_l \frac{e_{C_l}\left(\frac{o_{C_l}}{e_{C_l}} - 1\right)}{\sum_h e_{C_h}}\right)^2 \leq \frac{\sum_l e_{C_l}\left(\frac{o_{C_l}}{e_{C_l}} - 1\right)^2}{\sum_l e_{C_l}} = \frac{\sum_l \frac{(o_{C_l}-e_{C_l})^2}{e_{C_l}}}{e_C},$$

and therefore

$$\frac{(o_C - e_C)^2}{e_C} \leq \sum_l \frac{(o_{C_l} - e_{C_l})^2}{e_{C_l}}. \qquad (17)$$

Since $\sum_l \frac{(o_{C_l}-e_{C_l})^2}{e_{C_l}}$ or $\frac{(o_C-e_C)^2}{e_C}$ is part of the sum that defines $T^{\mathcal{I}}$, it follows from equations (17) and (16) that $\frac{T^{\mathcal{I}}}{N-m+1}$ converges uniformly almost surely to a positive constant greater than c'.

Let $|\Gamma|$ denote the cardinality of $\Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$. Since $\mathcal{I} \in \Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$ was arbitrary fixed, and since the convergence for fixed $\mathcal{I}$ of $t_C/[N-(m-1)]$ to a limit bounded from below by a positive constant was uniform, it follows that $\frac{1}{|\Gamma|}\sum_{\mathcal{I}\in\Gamma}\frac{T^{\mathcal{I}}}{N-m+1}$ converges almost surely to a positive constant at least as large as c'. To see this, note that from the uniform convergence in equation (16), it follows that for an arbitrary fixed $\epsilon > 0$, there exists $N(\epsilon)$ (which does not depend on C) such that for all $N > N(\epsilon)$, $\frac{1}{N-(m-1)}\frac{(o_C-e_C)^2}{e_C} \geq c' - \epsilon$ for every C, and therefore that $\frac{1}{|\Gamma|}\sum_{\mathcal{I}\in\Gamma}\frac{T^{\mathcal{I}}}{N-m+1} \geq c' - \epsilon$ for all $N > N(\epsilon)$.

Since $S_{m\times m} \geq \sum_{\mathcal{I}\in\Gamma} T^{\mathcal{I}}$, it follows that almost surely

$$\lim_{N\to\infty} \frac{S_{m\times m}}{|\Gamma|(N-m+1)} > c'. \qquad (18)$$

We shall show that $\lim_{N\to\infty} |\Gamma|/\binom{N}{m-1}$ is bounded below by a positive constant. First, we shall consider the case where m is finite. Then, a subset of $\Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$ is the set of all partitions with $m-2$ sample points in $\mathcal{A}_1$, and one sample point in $\mathcal{A}_2$. Therefore, $|\Gamma| \geq \binom{N_1}{m-2}N_2$. Simple algebraic manipulations lead to the following expression for $\frac{\binom{N_1}{m-2}N_2}{\binom{N}{m-1}}$:

$$(m-1)\frac{N_2}{N}\frac{N_1}{N-1}\cdots\frac{N_1-m+3}{N-m+2}.$$

Since N_1/N converges almost surely to f_1 , and similarly N_2/N converges almost surely to f_2, then for $m \geq 3$ finite it follows that $\frac{\binom{N_1}{m-2}N_2}{\binom{N}{m-1}}$ converges almost surely to a positive constant. Therefore, $|\Gamma|/\binom{N-1}{m-1}$ is bounded away from zero. Second, we shall consider the case that $m \to \infty$. The complement of Γ, Γ^C, is the set of contingency tables with no points in $\mathcal{A}_1$ or in $\mathcal{A}_2$. An upper bound for $|\Gamma^C|$ is

$$\binom{N-N_1}{m-1} + \binom{N-N_2}{m-1}.$$

Note that in order to show that $\lim_{N\to\infty} |\Gamma|/\binom{N}{m-1}$ is bounded below by a positive constant, since $|\Gamma| = \binom{N}{m-1} - |\Gamma^C|$, it is enough to show that $|\Gamma^C|/\binom{N}{m-1}$ converges to zero as $N \to \infty$. Simple algebraic manipulations lead to the following expression for $\frac{\binom{N-N_1}{m-1}}{\binom{N}{m-1}}$:

$$\left(1 - \frac{N_1}{N}\right)\cdots\left(1 - \frac{N_1}{N-(m-2)}\right) \leq \left(1 - \frac{N_1}{N}\right)^m.$$

Since N_1/N converges almost surely to $f_1 \in (0,1)$, it follows that $\frac{\binom{N-N_1}{m-1}}{\binom{N}{m-1}}$ converges almost surely to zero as $m \to \infty$. Similarly, since N_2/N converges almost surely to $f_2 \in (0,1)$, it follows that $\frac{\binom{N-N_2}{m-1}}{\binom{N}{m-1}}$ converges almost surely to zero. Therefore, $|\Gamma^C|/\binom{N}{m-1}$ converges to zero as $N \to \infty$.

Since $\lim_{N\to\infty} |\Gamma|/\binom{N}{m-1}$ is bounded below by a positive constant, it follows from (18) that almost surely

$$\lim_{N\to\infty} \frac{S_{m\times m}}{\binom{N}{m-1}(N-m+1)} \geq c'', \tag{19}$$

for some constant $c'' > 0$.

Consider now a random permutation $(\pi y_1), \ldots, (\pi y_N)$ of the y-values $y_1, \ldots, y_N$. Let $S^{\pi}_{m\times m}$ be the test statistic that is computed from the data $(x_1, \pi y_1), \ldots, (x_N, \pi y_N)$. Therefore, by Markov's inequality,

$$\begin{aligned} Pr\left(S^{\pi}_{m\times m} \geq c''\binom{N}{m-1}(N-m+1) \mid \vec{x}, \vec{y}\right) &\leq \frac{E(S^{\pi}_{m\times m}|\vec{x},\vec{y})}{c''\binom{N}{m-1}(N-m+1)} \\ \approx \quad \frac{\binom{N}{m-1}(m-1)^2}{c''\binom{N}{m-1}(N-m+1)}. \end{aligned} \tag{20}$$

where $\vec{x} = (x_1, \ldots, x_N)$ and $\vec{y} = (y_1, \ldots, y_N)$. The approximation in (20) becomes more accurate the larger N is, since each of the contingency tables is approximately χ^2 with $(m-1)^2$ degrees of freedom. The right hand side of equation (20) goes to 0 as $N \to \infty$, as long as $\lim_{N\to\infty} \frac{m}{\sqrt{N}} = 0$. Thus,

$$\lim_{N\to\infty, m/\sqrt{N}\to 0} Pr\left(S^{\pi}_{m\times m} \geq c''\binom{N}{m-1}(N-m+1) \mid \vec{x}, \vec{y}\right) = 0. \tag{21}$$

We now have all the necessary results to complete the proof. Specifically,

$$\lim_{N\to\infty} Pr(S_{m\times m} \leq S^{tab}_{1-\alpha}) \leq \lim_{N\to\infty} Pr\left\{S_{m\times m} \leq c''\binom{N}{m-1}(N-m+1)\right\} = 0,$$

where the inequality follows from (21), since $S^{tab}_{1-\alpha}$ is below $c''\binom{N}{m-1}(N-m+1)$ for N large enough, and the equality follows from (19), thus proving item 1 of Theorem 2.

To prove item 2, we will use the following inequality for chi-square distributions, which appears in equation (4.3) of Laurent and Massart (2000): for U a χ^2 statistic with D degrees of freedom, for any positive x, $Pr(U - D \geq 2\sqrt{Dx} + 2x) \leq e^{-x}$.

Let $\mathcal{I}$ be a fixed arbitrary partition of size m. Since for N large enough, under the null hypothesis, $T^{\mathcal{I}}$ is approximately a χ^2 statistic with $D = (m-1)^2$ degrees of freedom, it thus follows that for $x > D$,

$$Pr_{H_0}(T^{\mathcal{I}} - D \geq 4x) \leq e^{-x}. \tag{22}$$

Let $x = \frac{c'}{8}(N-m+1) - D/4$. Then for N large enough, $x > D$. It thus follows that

$$Pr_{H_0}(T^{\mathcal{I}} \geq \frac{c'}{2}(N-m+1)) \leq e^{-(\frac{c'}{8}(N-m+1)-D/4)} \leq e^{-(\frac{c'}{8}(N-m+1)-(m-1)^2/4)} \tag{23}$$

By Bonferroni's inequality,

$$\begin{aligned} Pr_{H_0}\left(M^{DDP}_{m\times m} \geq \frac{c'}{2}(N-m+1)\right) &\leq \sum_{\mathcal{I}\in\Pi^{DDP}_m} Pr_{H_0}\left(T^{\mathcal{I}} \geq \frac{c'}{2}(N-m+1)\right) \\ &\leq \binom{N}{m} e^{-(\frac{c'}{8}(N-m+1)-(m-1)^2/4)}, \end{aligned} \tag{24}$$

where the last inequality follows from (23). Since $\binom{N}{m}$ is at most $O(N^{\sqrt{N}})$, and since

$$e^{-(\frac{c'}{8}(N-m+1)-(m-1)^2/4)} = O(e^{-(\frac{c'}{8}N)}),$$

it follows that the expression in (24) goes to zero as $N \to \infty$.

Since we found contingency tables for which under the alternative the test statistic $\frac{T^{\mathcal{I}}}{N-m+1}$ converges uniformly almost surely to a positive constant greater than c' (16), it follows that $\frac{M^{DDP}_{m\times m}}{N-m+1}$ converges uniformly almost surely to a positive constant greater than c' when the null is false. From (24) it follows that as $N \to \infty$, with $\lim_{N\to\infty} \frac{m}{\sqrt{N}} = 0$, the probability that the test statistics $\frac{M^{DDP}_{m\times m}}{N-m+1}$ will be above $\frac{c'}{2}$ goes to zero when the null is true. It follows that the null hypothesis will be rejected with asymptotic probability one when it is false. ■

A.2 The ADP Test

Proof We want to show that if H_0 is false, then for an arbitrary fixed α, $\lim_{N\to\infty} Pr(S^{ADP}_{m\times m} > S^{tab}_{1-\alpha}) = 1$, where $S^{tab}_{1-\alpha}$ denotes the $1-\alpha$ quantile of the null distribution of $S^{ADP}_{m\times m}$. We use $\mathcal{A}, c, \mathcal{A}_1, \mathcal{A}_2, f_1, f_2$ as defined in the beginning of Appendix A of the main text.

For the ADP test, recall that the partitioning is based on selecting $m-1$ points from $1.5, \ldots, N-0.5$ for the partitions of the ranked x-values, and separately for the partitions of the ranked y-values. For a fixed rectangle, we say a grid point $(i+0.5, j+0.5)$ is in the rectangle if the two x-values with ranks i and $i+1$, and the two y-values with ranks j and $j+1$, are in the rectangle, for $(i,j) \in \{1, \ldots, N\}^2$. Let $\Gamma\{(x_1,y_1), \ldots, (x_N,y_N)\}$ be the set of partitions of size m with at least one grid point in $\mathcal{A}_1$ and at least one grid point in $\mathcal{A}_2$. Let N_{ix} be the number of x-coordinates of the grid points in $\mathcal{A}_i, i \in \{1,2\}$, and N_{iy} be the number of y-coordinates of the grid points in $\mathcal{A}_i, i \in \{1,2\}$.

Let $\mathcal{I} \in \Gamma\{(x_1,y_1), \ldots, (x_N,y_N)\}$ define an (arbitrary fixed) ADP partition in Γ. There exist two x-values in $\mathcal{A}_1$ that are separated by a grid point in $\mathcal{I}$, and two x-values in $\mathcal{A}_2$ that are separated by a grid point in $\mathcal{I}$, denote the average of these two x-values by x^*_1 and x^*_2. Let y^*_1 and y^*_2 be similarly defined for the y-values.

Let C be the cell defined by the points $(x^*_i, y^*_i), i = 1,2$. The fraction of observed counts in the cell C is a linear combination of empirical cumulative distribution functions

$$\frac{o_C}{N} = \hat{F}_{XY}(x^*_1, y^*_1) + \hat{F}_{XY}(x^*_2, y^*_2) - \hat{F}_{XY}(x^*_1, y^*_2) - \hat{F}_{XY}(x^*_2, y^*_1),$$

and the expected fraction under the null, is a function of the cumulative marginal distributions

$$\frac{e_C}{N} = \{\hat{F}_X(x^*_2) - \hat{F}_X(x^*_1)\}\{\hat{F}_Y(y^*_2) - \hat{F}_Y(y^*_1)\},$$

where $\hat{F}$ denotes the empirical cumulative distribution function based on N sample points.

By the Glivenko-Cantelli theorem, uniformly almost surely,

$$\lim_{N\to\infty}\left(\frac{o_C}{N}-\int_{\{(x,y):x\in(x_1^*,x_2^*],y\in(y_1^*,y_2^*]\}}h(x,y)dxdy\right)=0,$$
$$\lim_{N\to\infty}\left\{\frac{e_C}{N}-\left(\int_{\{x:x\in(x_1^*,x_2^*]\}}f(x)dx\right)\left(\int_{\{y:y\in(y_1^*,y_2^*]\}}g(y)dy\right)\right\}=0. \tag{25}$$

Therefore, by Slutsky's theorem and the continuous mapping theorem, we have that uniformly almost surely

$$\begin{aligned}&\lim_{N\to\infty}\frac{1}{N}\frac{(o_C-e_C)^2}{e_C}=\lim_{N\to\infty}\frac{\left(\frac{o_C}{N}-\frac{e_C}{N}\right)^2}{\frac{e_C}{N}}\\&\geq\lim_{N\to\infty}\left(\frac{o_C}{N}-\frac{e_C}{N}\right)^2\\&=\lim_{N\to\infty}\left[\int_{\{(x,y):x\in(x_1^*,x_2^*],y\in(y_1^*,y_2^*]\}}\{h(x,y)-f(x)g(y)\}dxdy\right]^2,\end{aligned} \tag{26}$$

where the inequality follows from the fact that $\frac{e_C}{N}\leq 1$.

We shall show that the limit (26) can be bounded from below by a positive constant that depends on (x_0,y_0) and R but not on $\mathcal{I}$. Since

$$\begin{aligned}&\{(x,y):x\in(x_0+R/4,x_0+3R/4],y\in(y_0+R/4,y_0+3R/4]\}\\&\subseteq\{(x,y):x\in(x_1^*,x_2^*],y\in(y_1^*,y_2^*]\},\end{aligned}$$

a positive lower bound can be obtained:

$$\begin{aligned}&\lim_{N\to\infty}\left[\int_{\{(x,y):x\in(x_1^*,x_2^*],y\in(y_1^*,y_2^*]\}}\{h(x,y)-f(x)g(y)\}dxdy\right]^2\\&\geq\left[\int_{\{(x,y):x\in(x_0+R/4,x_0+3R/4],y\in(y_0+R/4,y_0+3R/4]\}}\{h(x,y)-f(x)g(y)\}dxdy\right]^2\\&\geq c^2\int_{\{(x,y):x\in(x_0+R/4,x_0+3R/4],y\in(y_0+R/4,y_0+3R/4]\}}dxdy=c^2R^2/4,\end{aligned}$$

where the first inequality follows since $h(x,y)-f(x)g(y)>0$ in $\mathcal{A}$, and the second inequality follows since the minimum value is $c>0$. Therefore, it follows that $\frac{1}{N}\frac{(o_C-e_C)^2}{e_C}$ converges uniformly almost surely to a positive constant greater than $c'=c^2R^2/4$,

$$Pr\left(\lim_{N\to\infty}\frac{1}{N}\frac{(o_C-e_C)^2}{e_C}\geq c'\right)=1. \tag{27}$$

The partition $\mathcal{I}$ either contains the cell C, or a group of cells that divide C. By Jensen's inequality, it follows that in the latter case the score is made larger, see the arguments leading to expression

(17). It thus follows that the score $T^{\mathcal{I}}/N$ converges uniformly almost surely to a positive constant greater than c'.

Let $|\Gamma|$ denote the number of $\Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$. Since $\mathcal{I}\in\Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$ was arbitrarily fixed, it follows that $\frac{1}{|\Gamma|}\sum_{\mathcal{I}\in\Gamma}\frac{T^{\mathcal{I}}}{N}$ converges almost surely to a positive constant greater than $c'/2$. Since $S_{m\times m}\geq\sum_{\mathcal{I}\in\Gamma}T^{\mathcal{I}}$, it follows that almost surely,

$$\lim_{N\to\infty}\frac{S_{m\times m}}{|\Gamma|N}\geq\frac{c'}{2}. \tag{28}$$

We shall show that $\lim_{N\to\infty}|\Gamma|/\{\binom{N-1}{m-1}\binom{N-1}{m-1}\}$ is bounded below by a positive constant. First, we shall consider the case that m is finite. In this case, a subset of $\Gamma\{(x_1,y_1),\ldots,(x_N,y_N)\}$ is the set of all partitions with $m-2$ grid points in $\mathcal{A}_1$, and one grid point in $\mathcal{A}_2$, for both x-values and y-values. Therefore, $|\Gamma|\geq\binom{N_{1x}}{m-2}N_{2x}\binom{N_{1y}}{m-2}N_{2y}$. Simple algebraic manipulations lead to the following expression for $\frac{\binom{N_{1x}}{m-2}N_{2x}}{\binom{N-1}{m-1}}$:

$$(m-1)\frac{N_{2x}}{N-1}\frac{N_{1x}}{N-2}\cdots\frac{N_{1x}-m+3}{N-1-m+2}.$$

Since N_{1x}/N converges almost surely to $\int_{x_0}^{x_0+R/4}f(x)dx$, and similarly N_{2x}/N converges almost surely to $\int_{x_0+3R/4}^{x_0+R}f(x)dx$, then for $m\geq 3$ finite it follows that $\frac{\binom{N_{1x}}{m-2}N_{2x}}{\binom{N-1}{m-1}}$ converges almost surely to a positive constant. Similarly, $\frac{\binom{N_{1y}}{m-2}N_{2y}}{\binom{N-1}{m-1}}$ converges almost surely to a positive constant. Therefore, $|\Gamma|/\{\binom{N-1}{m-1}\binom{N-1}{m-1}\}$ is bounded away from zero.

Second, we shall consider the case that $m\to\infty$. The complement of Γ, Γ^C, is the set of contingency tables with no grid point in $\mathcal{A}_1$ or in $\mathcal{A}_2$. An upper bound for $|\Gamma^C|$ is:

$$\binom{N-1}{m-1}\left\{\binom{N-1-N_{1x}}{m-1}+\binom{N-1-N_{2x}}{m-1}+\binom{N-1-N_{1y}}{m-1}+\binom{N-1-N_{2y}}{m-1}\right\}$$

Note that since $|\Gamma|=\{\binom{N-1}{m-1}\binom{N-1}{m-1}\}-|\Gamma^C|$, it is enough to show that $|\Gamma^C|/\{\binom{N-1}{m-1}\binom{N-1}{m-1}\}$ converges to zero as $N\to\infty$. Simple algebraic manipulations lead to the following expression for $\frac{\binom{N-1-N_{1x}}{m-1}}{\binom{N-1}{m-1}}$:

$$\left(1-\frac{N_{1x}}{N-1}\right)\cdots\left(1-\frac{N_{1x}}{N-1-(m-2)}\right)\leq\left(1-\frac{N_{1x}}{N-1}\right)^m.$$

Since N_{1x}/N converges almost surely to a positive fraction $\int_{x_0}^{x_0+R/4}f(x)dx$, it follows that $\frac{\binom{N-1-N_{1x}}{m-1}}{\binom{N-1}{m-1}}$ converges almost surely to zero. Similarly, since N_{2x}/N, N_{1y}/N and N_{2y}/N converge almost surely to positive fractions, it follows that respectively, $\frac{\binom{N-1-N_{2x}}{m-1}}{\binom{N-1}{m-1}}$, $\frac{\binom{N-1-N_{1y}}{m-1}}{\binom{N-1}{m-1}}$, and $\frac{\binom{N-1-N_{2y}}{m-1}}{\binom{N-1}{m-1}}$ converge almost surely to zero. Thus $|\Gamma^C|/\{\binom{N-1}{m-1}\binom{N-1}{m-1}\}$ converges almost surely to zero.

Since $\lim_{N\to\infty} |\Gamma|/\{\binom{N-1}{m-1}\binom{N-1}{m-1}\}$ is bounded below by a positive constant, it follows from (28) that almost surely,

$$\lim_{N\to\infty} \frac{S_{m\times m}}{\binom{N-1}{m-1}\binom{N-1}{m-1}N} \geq c'', \tag{29}$$

for some constant $c'' > 0$.

Consider now a random permutation $(\pi y_1), \ldots, (\pi y_N)$ of the y-values $y_1, \ldots, y_N$. Let $S^{\pi}_{m\times m}$ be the test statistic that is computed from the data $(x_1, \pi y_1), \ldots, (x_N, \pi y_N)$. By Markov's inequality,

$$\begin{aligned} & Pr\left(S^{\pi}_{m\times m} \geq c''\binom{N-1}{m-1}\binom{N-1}{m-1}N \mid \vec{x}, \vec{y}\right) \leq \frac{E(S^{\pi}_{m\times m} \mid \vec{x}, \vec{y})}{c''\binom{N-1}{m-1}\binom{N-1}{m-1}N} \\ \approx & \frac{\binom{N-1}{m-1}\binom{N-1}{m-1}(m-1)^2}{c''\binom{N-1}{m-1}\binom{N-1}{m-1}N}, \end{aligned} \tag{30}$$

where $\vec{x} = (x_1, \ldots, x_N)$ and $\vec{y} = (y_1, \ldots, y_N)$. The approximation in (30) becomes more accurate the larger N is, since each of the contingency tables is approximately χ^2 with $(m-1)^2$ degrees of freedom. The right hand side of equation (30) goes to zero as $N \to \infty$, as long as $\lim_{N\to\infty} \frac{m}{\sqrt{N}} = 0$. Thus,

$$\lim_{N\to\infty, m/\sqrt{N}\to 0} Pr\left(S^{\pi}_{m\times m} \geq c''\binom{N-1}{m-1}\binom{N-1}{m-1}N \mid \vec{x}, \vec{y}\right) = 0. \tag{31}$$

We now have all the necessary results to complete the proof. Specifically,

$$\lim_{N\to\infty} Pr(S_{m\times m} \leq S^{tab}_{1-\alpha}) \leq Pr\left(S^{\pi}_{m\times m} \leq c''\binom{N-1}{m-1}\binom{N-1}{m-1}N\right) = 0, \tag{32}$$

where the inequality follows from (31), since $S^{tab}_{1-\alpha}$ is below $c''\binom{N-1}{m-1}\binom{N-1}{m-1}N$ for N large enough, and the equality follows from (29), thus proving item 1 of Theorem 1 for the ADP summation statistic.

■

Appendix B. Proof of Theorem 4

Proof We want to show that for all $\epsilon > 0, \lim_{N\to\infty} Pr\left(\left|\frac{S^{ADP}_{m\times m}(L)}{N|\Pi|} - I_{XY}\right| > \epsilon\right) = 0$ if $\lim_{N\to\infty} \frac{m}{\sqrt{N}} = 0$ and $\lim_{N\to\infty} m = \infty$, where $|\Pi| = \left(\binom{N-1}{m-1}\right)^2$ is the number of partitions.

For continuous marginals, the copula function of the joint distribution of (X, Y) is unique, denote it by $c(u, v)$. The mutual information is the negative copula entropy, $H_{UV} = -\int c(u, v) \log c(u, v) du dv$,

$$\begin{aligned} I_{XY} &= \int c(F_X(x), F_Y(y)) f(x) g(y) \log c(F_X(x), F_Y(y)) dx dy \\ &= \int c(u, v) \log c(u, v) du dv = -H_{UV}. \end{aligned} \tag{33}$$

Consider an arbitrary fixed partition $\mathcal{I} = \{(i_1, j_1), \ldots, (i_{m-1}, j_{m-1})\} \subset \{1.5, \ldots, N - 0.5\}^2$. Recall that $\mathcal{C}(\mathcal{I})$ is the set of $m \times m$ cells that are defined by the partition. For a cell C, let $r_l(C)$ and $r_h(C)$ be, respectively, the lowest and highest x-grid integer values in C. Similarly, let $s_l(C)$ and $s_h(C)$ be, respectively, the lowest and highest y-grid integer values in C.

The entropy of the partition $\mathcal{I}$ is

$$H^{\mathcal{I}}_{UV} = -\sum_{C\in\mathcal{C}(\mathcal{I})} Pr\left(\frac{r_l(C)}{N} \leq U \leq \frac{r_h(C)}{N}, \frac{s_l(C)}{N} \leq V \leq \frac{s_h(C)}{N}\right) \log\left\{Pr\left(\frac{r_l(C)}{N} \leq U \leq \frac{r_h(C)}{N}, \frac{s_l(C)}{N} \leq V \leq \frac{s_h(C)}{N}\right)\right\}.$$

The corresponding empirical (plug in) estimator is

$$\hat{H}^{\mathcal{I}}_{UV} = -\sum_{C\in\mathcal{C}(\mathcal{I})} \frac{o_C}{N} \log\left(\frac{o_C}{N}\right), \quad o_C = \sum_{i=1}^{N} I(r_l(C) \leq r_i \leq r_h(C), s_l(C) \leq s_i \leq s_h(C)).$$

Let $H^{\mathcal{I}}_U$ and $H^{\mathcal{I}}_V$ be the fixed marginal entropies of the partition $\mathcal{I}$:

$$\begin{aligned} H^{\mathcal{I}}_U &= -\sum_{C_x\in\mathcal{C}_x(\mathcal{I})} \frac{r_h(C_x) - r_l(C_x)}{N} \log\left(\frac{r_h(C_x) - r_l(C_x)}{N}\right), \\ H^{\mathcal{I}}_V &= -\sum_{C_y\in\mathcal{C}_y(\mathcal{I})} \frac{r_h(C_y) - r_l(C_y)}{N} \log\left(\frac{r_h(C_y) - r_l(C_y)}{N}\right), \end{aligned}$$

where $\mathcal{C}_x(\mathcal{I})$ and $\mathcal{C}_y(\mathcal{I})$ are the intervals induced by $\mathcal{I}$ in x and in y, respectively. Note that given $\mathcal{I}$, the observed and expected margins of the partitions are fixed, and therefore

$$H^{\mathcal{I}}_U = -\sum_{C\in\mathcal{C}(\mathcal{I})} o_C \log\left(\frac{r_h(C) - r_l(C)}{N}\right) \tag{34}$$

$$= -\sum_{C\in\mathcal{C}(\mathcal{I})} Pr(r_l(C) \leq U \leq r_h(C), s_l(C) \leq V \leq s_h(C)) \log\left(\frac{r_h(C) - r_l(C)}{N}\right) \tag{35}$$

$$H^{\mathcal{I}}_V = -\sum_{C\in\mathcal{C}(\mathcal{I})} o_C \log\left(\frac{s_h(C) - s_l(C)}{N}\right) \tag{36}$$

$$= -\sum_{C\in\mathcal{C}(\mathcal{I})} Pr(r_l(C) \leq U \leq r_h(C), s_l(C) \leq V \leq s_h(C)) \log\left(\frac{s_h(C) - s_l(C)}{N}\right) \tag{37}$$

The following simple derivation shows that the likelihood ratio score $T^{\mathcal{I}}$ is a linear combination of $\hat{H}^{\mathcal{I}}_{UV}, H^{\mathcal{I}}_U$ and $H^{\mathcal{I}}_V$:

$$\begin{aligned} T^{\mathcal{I}} &= \sum_{C\in\mathcal{C}(\mathcal{I})} o_C \log\frac{o_C}{e_C} \\ &= \sum_{C\in\mathcal{C}(\mathcal{I})} o_C \log\frac{o_C}{N} - \sum_{C\in\mathcal{C}(\mathcal{I})} o_C \log\frac{e_C}{N} \\ &= -N\hat{H}^{\mathcal{I}}_{UV} - \sum_{C\in\mathcal{C}(\mathcal{I})} o_C \log\left(N\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}\frac{1}{N}\right) \\ &= -N\hat{H}^{\mathcal{I}}_{UV} + NH^{\mathcal{I}}_U + NH^{\mathcal{I}}_V, \end{aligned}$$

where the last equality follows from equations (34) and (36).

Let $E(\cdot)$ denote the expectation of a random variable. We bound from above our probability of interest by a sum of three probabilities as follows.

$$
\begin{aligned}
&Pr\left(\left|\frac{S^{ADP}_{m\times m}}{N|\Pi|} - I_{XY}\right| > \epsilon\right) = Pr\left(\left|\frac{\sum_{\mathcal{I}} T^{\mathcal{I}}_{m\times m}(L)}{N|\Pi|} - I_{XY}\right| > \epsilon\right)\\
&= Pr\left(\left|\frac{\sum_{\mathcal{I}}(H^{\mathcal{I}}_U + H^{\mathcal{I}}_V - \hat{H}^{\mathcal{I}}_{UV})}{|\Pi|} + H_{UV}\right| > \epsilon\right)\\
&= Pr(|\sum_{\mathcal{I}}(H^{\mathcal{I}}_U + H^{\mathcal{I}}_V - \hat{H}^{\mathcal{I}}_{UV} + H_{UV})| > |\Pi|\epsilon)\\
&= Pr(|\sum_{\mathcal{I}}(-\hat{H}^{\mathcal{I}}_{UV} + E(\hat{H}^{\mathcal{I}}_{UV})) + \sum_{\mathcal{I}}(-E(\hat{H}^{\mathcal{I}}_{UV}) + H^{\mathcal{I}}_{UV})\\
&+\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})| > |\Pi|\epsilon)\\
&\le Pr(|\sum_{\mathcal{I}}(-\hat{H}^{\mathcal{I}}_{UV} + E(\hat{H}^{\mathcal{I}}_{UV}))| + |\sum_{\mathcal{I}}(-E(\hat{H}^{\mathcal{I}}_{UV}) + H^{\mathcal{I}}_{UV})|\\
&+|\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})| > |\Pi|\epsilon)\\
&\le Pr(|\sum_{\mathcal{I}}(-\hat{H}^{\mathcal{I}}_{UV} + E(\hat{H}^{\mathcal{I}}_{UV}))| > |\Pi|\epsilon/3) \qquad (38)\\
&+Pr(|\sum_{\mathcal{I}}(-E(\hat{H}^{\mathcal{I}}_{UV}) + H^{\mathcal{I}}_{UV})| > |\Pi|\epsilon/3) \qquad (39)\\
&+Pr(|\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})| > |\Pi|\epsilon/3), \qquad (40)
\end{aligned}
$$

where the last inequality follows from $\{|\sum_{\mathcal{I}}(-\hat{H}^{\mathcal{I}}_{UV} + E(\hat{H}^{\mathcal{I}}_{UV}))| + |\sum_{\mathcal{I}}(-E(\hat{H}^{\mathcal{I}}_{UV}) + H^{\mathcal{I}}_{UV})| + |\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})| > |\Pi|\epsilon\} \subseteq \{(|\sum_{\mathcal{I}}(-\hat{H}^{\mathcal{I}}_{UV} + E(\hat{H}^{\mathcal{I}}_{UV}))| > |\Pi|\epsilon/3) \cup (|\sum_{\mathcal{I}}(-E(\hat{H}^{\mathcal{I}}_{UV}) + H^{\mathcal{I}}_{UV})| > |\Pi|\epsilon/3) \cup |\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})| > |\Pi|\epsilon/3\}$ and Bonferroni's inequality.

We will show that the three probabilities (38)–(40) vanish as $N \to \infty, m \to \infty, \frac{m}{\sqrt{N}} \to 0$, thus proving the theorem.

The probability (38) can be upper-bounded as follows,

$$
\begin{aligned}
&Pr\left(|\sum_{\mathcal{I}}(\hat{H}^{\mathcal{I}}_{UV} - E(\hat{H}^{\mathcal{I}}_{UV}))| \ge |\Pi|\epsilon/3\right) \le \sum_{\mathcal{I}} Pr\left(|(\hat{H}^{\mathcal{I}}_{UV} - E(\hat{H}^{\mathcal{I}}_{UV}))| \ge \epsilon/3\right)\\
&\le |\Pi| 3e^{-\frac{N}{2}\epsilon^2/9\frac{1}{(\log N)^2}} \qquad (41)
\end{aligned}
$$

where the first inequality follows from the fact that $\{|\sum_{\mathcal{I}}(\hat{H}^{\mathcal{I}}_{UV} - E(\hat{H}^{\mathcal{I}}_{UV}))| \ge |\Pi|\epsilon/3\} \subseteq \{\cup_{\mathcal{I}\in\Pi}(\hat{H}^{\mathcal{I}}_{UV} - E(\hat{H}^{\mathcal{I}}_{UV})) \ge \epsilon/3\}$ and Bonferroni's inequality, and the second inequality follows from the upper bound (3.4) in Paninski (2003) for the plug in estimator for a given partition $\mathcal{I}$. This probability goes to zero as $N \to \infty$, since $|\Pi|$ is $O(N^{\sqrt{N}})$ and $\lim_{N\to\infty} O(N^{\sqrt{N}})e^{-\frac{N}{2}\epsilon^2/9\frac{1}{(\log N)^2}} = 0$.

The event in the second probability (39) is not random, so we need to show that $|\sum_{\mathcal{I}}(E(\hat{H}^{\mathcal{I}}_{UV}) - H^{\mathcal{I}}_{UV})| < |\Pi|\epsilon/3$ for N large enough. Proposition 1 in Paninski (2003) states that $0 \le (H^{\mathcal{I}}_{UV} - E(\hat{H}^{\mathcal{I}}_{UV})) \le \log(1 + \frac{(m-1)^2-1}{N})$. Therefore,

$$
|\sum_{\mathcal{I}}(E(\hat{H}^{\mathcal{I}}_{UV}) - H^{\mathcal{I}}_{UV})| \le |\Pi|\log(1 + \frac{(m-1)^2-1}{N})
$$

Clearly, the RHS is below $|\Pi|\epsilon/3$ for N large enough, if $\lim_{N\to\infty}\frac{m}{\sqrt{N}} = 0$.

It remains to show that (40) vanishes as $N \to \infty$. This event is not random, so we will show that

$$
\lim_{N\to\infty}\frac{|\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})|}{|\Pi|} < \epsilon/3.
$$

By the mean value theorem, for cell C there exists a point (u_C, v_C) in C such that

$$
Pr(r_l(C) \le U \le r_h(C), s_l(C) \le V \le s_h(C)) = c(u_C, v_C)\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}.
$$

Therefore,

$$
\begin{aligned}
&-H^{\mathcal{I}}_{UV}\\
&= \sum_{C\in\mathcal{C}(\mathcal{I})} c(u_C, v_C)\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}\log\left(c(u_C, v_C)\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}\right)\\
&= \sum_{C\in\mathcal{C}(\mathcal{I})}\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}c(u_C, v_C)\log c(u_C, v_C) \qquad (42)\\
&+ \sum_{C\in\mathcal{C}(\mathcal{I})} Pr(r_l(C) \le U \le r_h(C), s_l(C) \le V \le s_h(C))\log\frac{r_h(C) - r_l(C)}{N}\\
&+ \sum_{C\in\mathcal{C}(\mathcal{I})} Pr(r_l(C) \le U \le r_h(C), s_l(C) \le V \le s_h(C))\log\frac{s_h(C) - s_l(C)}{N}\\
&= \sum_{C\in\mathcal{C}(\mathcal{I})}\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}c(u_C, v_C)\log c(u_C, v_C) - H^{\mathcal{I}}_U - H^{\mathcal{I}}_V, \qquad (43)
\end{aligned}
$$

where the last equality follows from equations (35) and (37).

By the definition of the Riemann integral, expression (42) can be made arbitrarily close to $-H_{UV}$. Specifically, there exists a $0 < d(\epsilon) < 1$ such that if all cells satisfy $\frac{r_h(C)-r_l(C)}{N} < d$ and $\frac{s_h(C)-s_l(C)}{N} < d$, then

$$
|\sum_{C\in\mathcal{C}(\mathcal{I})}\frac{r_h(C) - r_l(C)}{N}\frac{s_h(C) - s_l(C)}{N}c(u_C, v_C)\log c(u_C, v_C) + H_{UV}| < \epsilon/3.
$$

Therefore, it follows that for any partition $\mathcal{I} \in \Pi$ for which all cells satisfy $\frac{r_h(C)-r_l(C)}{N} < d$ and $\frac{s_h(C)-s_l(C)}{N} < d$, then we have

$$
|(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_U + H^{\mathcal{I}}_V + H_{UV})| < \epsilon/3.
$$

It remains to show that the contribution of the fraction of partitions that do not satisfy $\frac{r_h(C)-r_l(C)}{N} < d$ and $\frac{s_h(C)-s_l(C)}{N} < d$ goes to zero as $N \to \infty$. Since the probability of selecting an x-value (or y-value) for a partition that will have a cell larger than d is $1-d$, the fraction of "bad" partitions is upper-bounded by

$$\frac{2m\binom{N(1-d)}{m-1}\binom{N}{m-1}}{\left(\binom{N-1}{m-1}\right)^2} \leq 2m(1-d)^{m-1}.$$

Since $m \to \infty$ the fraction of bad partitions goes to zero.

Note that $|(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_{U} + H^{\mathcal{I}}_{V} + H_{UV})|$ is at most $O(\log m^2)$ because by Jensen's inequality, $|H^{\mathcal{I}}_{UV}| \leq \log m^2, |H^{\mathcal{I}}_{U}| \leq \log m^2, |H^{\mathcal{I}}_{V}| \leq \log m^2$, and $|H_{UV}| = I_{XY}$ is assumed to be bounded. Therefore,

$$\frac{|\sum_{\mathcal{I}}(-H^{\mathcal{I}}_{UV} + H^{\mathcal{I}}_{U} + H^{\mathcal{I}}_{V} + H_{UV})|}{|\Pi|} \leq \epsilon/3 + O\left(\log m^2 m(1-d)^{m-1}\right).$$

Since the second term of the RHS goes to zero as $N \to \infty, m \to \infty, \frac{m}{\sqrt{N}} \to 0$, the proof is complete. ■

Appendix C. Proof of Theorem 6

Proof We shall prove items 1 and 2 for the DDP statistic only, since the proof for the ADP statistic is very similar. We shall use the notation of Section A.1. Let $\hat{m}$ be the value of m with minimum p-value,

$$\hat{m} = \arg \min_{m \in \{2,\ldots,m_{\max}\}} p_2, \ldots, p_m.$$

To prove item 1, we note that from equation (19) it follows that under the alternative,

$$\lim_{N\to\infty} \frac{S_{\hat{m}\times\hat{m}}}{\binom{N}{\hat{m}-1}(N-\hat{m}+1)} \geq c''.$$

Under the null,

$$\begin{aligned}
&Pr\left(S^{\pi}_{\hat{m}\times\hat{m}} \geq c''\binom{N}{\hat{m}-1}(N-\hat{m}+1) \mid \vec{x},\vec{y}\right) \\
&\leq \sum_{m=2}^{m_{\max}} Pr\left(S^{\pi}_{m\times m} \geq c''\binom{N}{m-1}(N-m+1) \mid \vec{x},\vec{y}\right) \\
&\leq m_{\max} \max_{m\in\{2,\ldots,m_{\max}\}} \frac{\binom{N}{m-1}(m-1)^2}{c''\binom{N}{m-1}(N-m+1)} \leq m_{\max}\frac{(m_{\max}-1)^2}{c''(N-m_{\max}+1)},
\end{aligned}$$

where the first inequality is the Bonferroni inequality, and the second inequality follows from equation (20). Since the last term goes to 0 as $N \to \infty$ if $\lim_{N\to\infty} m/N^{1/3} = 0$, the proof of item 1 is complete.

The proof of item 2 is very similar to the proof in Section A.1, the only modification is an additional application of Bonferroni's inequality under the null:

$$\begin{aligned}
&Pr_{H_0}\left(M^{DDP}_{\hat{m}\times\hat{m}} \geq \frac{c'}{2}(N-\hat{m}+1)\right) \leq \sum_{m=2}^{m_{\max}} Pr_{H_0}\left(M^{DDP}_{m\times m} \geq \frac{c'}{2}(N-m+1)\right) \\
&\leq m_{\max} \max_{m\in\{2,\ldots,m_{\max}\}} \binom{N}{m} e^{-(\frac{c'}{8}(N-m+1)-(m-1)^2/4)} \\
&\leq m_{\max}\binom{N}{m_{\max}} e^{-(\frac{c'}{8}(N-m_{\max}+1)-(m_{\max}-1)^2/4)},
\end{aligned}$$

where the first inequality in the last row follows from (24). Since $m_{\max}\binom{N}{m_{\max}}$ is at most $O(N \times N^{\sqrt{N}})$, and since

$$e^{-(\frac{c'}{8}(N-m_{\max}+1)-(m_{\max}-1)^2/4)} = O(e^{-(\frac{c'}{8}N)}),$$

it follows that the last expression goes to zero as $N \to \infty$. ■

Appendix D. An Example of Mutual Information Estimation

We examined our suggested estimator in the following setup. For $m = 15$ and $N \in \{300, 1000\}$ sample points drawn from a two–component Gaussian mixture, we simulated 50 data sets and computed the estimated mutual information using $S^{DDP}_{m\times m}$, $S^{ADP}_{m\times m}$, and the histogram estimator. The Gaussian mixture density was

$$0.8 \times f_{\mathcal{N}}\left(\begin{pmatrix}0.5\\0.5\end{pmatrix}, \begin{pmatrix}0.05 & 0.025\\0.025 & 0.05\end{pmatrix}\right) + 0.2 \times f_{\mathcal{N}}\left(\begin{pmatrix}0.125\\0.675\end{pmatrix}, \begin{pmatrix}0.01 & 0\\0 & 0.01\end{pmatrix}\right),$$

where $f_{\mathcal{N}}(\mu, \Sigma)$ is the bivariate normal distribution with mean μ and covariance matrix Σ. For each partition, we applied the Miller–Madow correction (Paninski, 2003), a simple and well-known modification that estimates the systematic error of the histogram estimators and improves the finite-sample properties of these estimators. We compared it to the histogram estimator, as well as to the estimator based on the DDP statistic that considers only a subset of all possible partitions. Table 5 shows that the variability and the bias decrease as N increases for all estimators, and that the ADP estimator is the least variable, as is intuitively expected since it is the average over many partitions. In practice, it is difficult to identify the optimal m: it should not be too small so that the local dependence structure is not missed, causing large bias, nor should it be too large so that the grid created is too sparse, causing large variance.

Appendix E. Algorithm for the DDP Statistic

For the DDP statistic, the algorithm is very similar to that for the ADP statistic, except that only DDP are considered on the grid of ranked data $\{1, \ldots, N\}^2$. The algorithm is slightly more complex because the number of partitions that include a cell C depends on the partition size m, on the data, and on the type of cell, with four possible types. Specifics follow for internal cells. The first type is a cell C for which there is a sample point that falls on the boundary of C but not on one of its

	$\mathbf{N = 300}$	$\mathbf{N = 1000}$
Histogram	0.3165 (0.0052)	0.1854 (0.0029)
Data derived partitions	0.3010 (0.0030)	0.1879 (0.0022)
All derived partitions	0.2954 (0.0028)	0.1860 (0.0021)

Table 5: The average (standard error) of the mutual information estimates using random samples of size 300 (column 2) and 1000 (column 3) from the two component Gaussian mixture, by the following methods: the naive histogram estimator that partitions each axis to 15 intervals of equal count, $S^{ADP}_{15\times15}/(N|\Pi^{ADP}_{15}|)$, and $S^{DDP}_{15\times15}/(N|\Pi^{DDP}_{15}|)$. The true mutual information value was 0.1784.

corners. Then no DDP can ever have C as a cell, and therefore the number of DDP that include C is zero. For example, in Figure 7 (middle panel), if the open circle is an observation, and therefore the filled circle with the same y value is not, since there are no ties, then any DDP with this y value will necessarily partition C at the x value of the open circle observation, and thus C cannot be a cell in any DDP. For the remaining three types of cells, if there are sample points that fall on the boundary of C they are necessarily on the corners of C. These types of cells differ by the number of observations that determine the cell. The second type is a cell C defined by two observed points. Then, the number of DDP that include C is the number of ways to choose $m-3$ points from the points in the four outer areas defined by $(0, r_l) \times (0, s_l)$, $(0, r_l) \times (s_h, N]$, $(r_h, N] \times (0, s_l)$, and $(r_h, N] \times (s_h, N]$, see Figure 7 (left panel) for illustration. The number of points in the four areas is calculable in $O(1)$ using A, as defined in equation (10). Specifically, the count of samples that fall strictly inside any cell with rank ranges $r \in [r_l, r_h]$ and $s \in [s_l, s_h]$ is:

$$o_C = A(r_h - 1, s_h - 1) - A(r_l, s_h - 1) - A(r_h - 1, s_l) + A(r_l, s_l).$$

The third and fourth type are cells defined by three or four observed points, respectively, see Figure 7 (middle and right panels). Now the number of DDP that include C is exactly the number of ways to choose $m-4$ and $m-5$ points, respectively, from the points in the four outer areas. Again, the number of points in the four outer areas is calculable in $O(1)$ using A. Since all cells are defined by two, three, or four points, there are no additional types of cells.

Let us denote the number of sample points in the four outer areas of a cell C by OUT. Further denote by $C(OUT)$ the group of all cells which have exactly OUT points in the outer areas. For the sake of brevity, lets consider only cells of type 2. Similarly to equation (12) for the ADP statistic, the contribution to the score $S^{DDP}_{m\times m}$ of internal cells of type 2 can be written as

$$\sum_{OUT=2}^{N-2} n(OUT, m) \sum_{C\in C(OUT)} t_C = \sum_{OUT=2}^{N-2} n(OUT, m) T(OUT). \tag{44}$$

The algorithm proceeds as follows. First in a preprocessing phase we perform two computations: 1)go over all cells and calculate t_C and OUT for each cell and update $T(OUT) = T(OUT) + t_C$. This stage takes $O(N^4)$; 2) for all $u, v \in \{0, \ldots, N\}$ we calculate and store all $\binom{u}{v}$ in $O(N^2)$ steps using Pascal's triangle method.

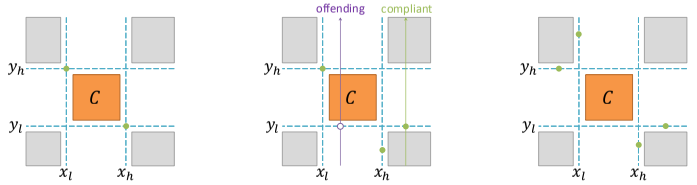


Figure 7: Inner cell example C (orange), and the sample points that define a partition where C is a cell (green). If any of the sample points ranked on a boundary column or row of C is not on the corner of C, then C can never be a DDP cell (offending point in purple, middle panel). An inner cell can be defined either by two data points (left), by three data points (middle), or by four data points (right).

Now for each m, since $n(OUT, m) = \binom{OUT}{m-3}$, given $T(OUT)$, clearly equation (44) can be calculated in $O(N)$ for a fixed m and in $O(N^2)$ for all ms. Therefore the total complexity is again $O(N^4)$ due to the preprocessing phase.

Appendix F. A Fast Algorithm for Computing the HHG Statistic

Here we describe a fast algorithm for computing the univariate original (distribution-dependent) HHG test statistic.

If $D_X(x_1, x_2)$ and $D_Y(y_1, y_2)$ are distance metrics in the variables tested for independence (for which, again, we have a paired sample with N i.i.d samples), the HHG test requires computing $N(N-1)$ different 2×2 contingency tables according to the following partitions of the distance-distance plane. For every "origin" sample i, and for every "radius" sample j, the remaining $N-2$ samples are classified according to whether their X and Y distances from i are both smaller than the X and Y distances from j to i, or if only X, only Y, or neither are smaller than the respective j to i distances. In the general case, generating contingency tables for all pairs can be done in $N^2 \log N$, as described in (Heller et al., 2013). In the univariate case, an $O(N^2)$ algorithm proceeds as follows.

Instead of working in the distance-distance plane, the algorithm is specified in terms of the (x, y), i.e., the sample plane. It is sufficient to consider the discrete grid expanded from unique X samples and unique Y samples actually observed (these can be identified in $O(N \log N)$). The double cumulative sum over this $N \times N$ grid is computed as in Section 3.1 of the main text, with the only difference being that after A is initialized to all zeros, it is updated sequentially with $A(r_i, s_i) = A(r_i, s_i) + 1$ for every sample of ranks in x and ranks in y, (r_i, s_i), $i = 1, 2, \ldots, N$, to account for possible ties.

Since, in the univariate case, $D_X(x_i, x_k) < D_X(x_i, x_j)$ is equivalent to $|x_k - x_i| < |x_j - x_i|$, and similarly for y, partition cells are simply axis-aligned rectangles, as in the distribution-free test. Here, however, only sample j is a vertex, and sample i is the center of mass. The diagonal-opposing vertex from j may not even be a point in the sample, and thus is not directly captured by A. Still, the appropriate point to sample A in, for computing the contingency table cell in $O(1)$, can be found in $O(1)$ additional amortized time, as follows:

1. Sort once the unique values of x, and do the same for y.

2. When traversing all pairs, first traverse i.

   (a) For every i, traverse the sorted values of x with two concurrent iterators starting from x_i, one moving right (i.e., from low to high x) and one advancing left.

   (b) Subsequently to each step taken with the right iterator, arriving at an x_j, advance the left iterator until a value is encountered which is farther from x_i than x_j is, and this is the opposing vertex x coordinate value for the rectangle for the pair i, j.

   The process, which is depicted in Figure 8, takes $O(N)$ time for N values, and is repeated for the y axis.

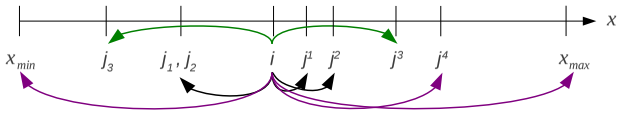


Figure 8: Finding the grid coordinates to sample for the $O(N^2)$ univariate (distribution-dependent) HHG algorithm.

Appendix G. Additional Simulations for the Two-Sample Problem

Table 6 shows a comparison of the performance of the minimum p-value and Fisher-combined test statistics. The variant with highest power in most setups (specifically, setups 2,4,5,6,8,9,10,11,12) is the Fisher-combined p-value using aggregation by summation and $m_{\max} = 50$. However, the Fisher-combined p-value test is sensitive to the choice of $m_{\max}$, which is unknown in practice. Since we view this as a significant weakness of the Fisher-combined statistic, and since the minimum p-value does not have this weakness and has very good power when compared with Fisher as well as when compared with other tests (with a large range of $m_{\max}$ values examined), we recommend the minimum p-value test statistic.

Table 7 shows the power using different priors for regularization. The priors are as follows: $\pi(m) = \sqrt{N}^m exp(-\sqrt{N})/m!$ for Poisson; $\pi(m) = \binom{N-1}{m-1} p^m (1-p)^{(N-m)}$ for Binomial, with $p = 0.119$ so that the penalty becomes that of the Akaike information criterion (i.e., $\log \pi(\mathcal{I}|m)\pi(m) = -2m$); $\pi(m) = 1/K$ for fixed K for Uniform. We also considered the prior in Jiang et al. (2014), resulting in the DS test with penalty $-\lambda_0 \log N(m-1)$. According to the recommendation in Jiang et al. (2014), we chose the value of λ_0 so that the level under the null is as close as possible to 0.05 from below, so $\lambda_0 = 1.11088$ for $N = 100$, and $\lambda_0 = 0.904$ for $N = 500$. The Poisson prior (with rate $\sqrt{N}$) was by far the best among all priors considered, and its power was comparable to that of the minimum p-value displayed in Table 6.

For sample size $N = 100$, we examined the distributions depicted in Figure 10, and we used 20,000 simulated data sets, in each of the configurations. Table 9 and Figure 11 show the power for

	Minimum p-value				Fisher combined p-value			
	Max aggregation, $m_{\max}$		Sum aggregation, $m_{\max}$		Max aggregation, $m_{\max}$		Sum aggregation, $m_{\max}$	
Setup	$= 50$	$= 149$	$= 50$	$= 149$	$= 50$	$= 149$	$= 50$	$= 149$
1	0.836	0.825	0.500	0.491	0.631	0.424	0.549	0.560
2	0.810	0.799	0.883	0.873	0.819	0.701	0.917	0.898
3	0.783	0.785	0.553	0.733	0.867	0.847	0.502	0.782
4	0.836	0.827	0.945	0.937	0.749	0.567	0.952	0.882
5	0.607	0.592	0.708	0.686	0.531	0.381	0.760	0.655
6	0.829	0.818	0.836	0.820	0.749	0.567	0.882	0.821
7	0.358	0.339	0.516	0.492	0.201	0.135	0.430	0.287
8	0.770	0.752	0.795	0.775	0.542	0.361	0.814	0.672
9	0.531	0.512	0.641	0.613	0.344	0.227	0.635	0.470
10	0.728	0.711	0.824	0.806	0.569	0.393	0.849	0.721
11	0.556	0.540	0.707	0.686	0.509	0.380	0.756	0.657
12	0.402	0.390	0.599	0.577	0.399	0.293	0.655	0.557
13	0.852	0.844	0.777	0.764	0.809	0.663	0.823	0.822

Table 6: Power of the minimum p-value test statistic (columns 2–5) as well as the Fisher-combined p-value test statistic (columns 6–9), using the different aggregation methods and two values of $m_{\max}$, for $N = 500$ and the setups of Figure 2. The difference between $m_{\max} = 50$ and $m_{\max} = 149$ is much larger for the Fisher-combined than for the minimum p-value test statistic when aggregated by summation (columns 8–9 versus columns 4–5). Moreover, the power is typically lower for the Fisher-combined that for the minimum p-value test statistic when the aggregation is by maximization (columns 6–7 versus columns 2–3).

Setup	Poisson prior Max aggregation	Poisson prior Sum aggregation	Uniform Prior Max aggregation	Binomial Prior Max aggregation	DS prior Max aggregation
1	0.829	0.597	0.848	0.450	0.845
2	0.855	0.924	0.641	0.745	0.565
3	0.715	0.639	0.282	0.874	0.239
4	0.861	0.920	0.813	0.593	0.794
5	0.659	0.746	0.592	0.411	0.573
6	0.854	0.874	0.804	0.597	0.787
7	0.291	0.310	0.430	0.139	0.432
8	0.719	0.760	0.835	0.371	0.836
9	0.489	0.545	0.599	0.232	0.600
10	0.709	0.793	0.749	0.406	0.744
11	0.607	0.720	0.466	0.390	0.433
12	0.460	0.629	0.298	0.318	0.268
13	0.860	0.857	0.802	0.697	0.776

Table 7: Power using different priors , for $N = 500$ and the setups of Figure 2.

the setups in Figure 10. These results concur with the results in Section 4.1 for $N = 500$, and show that if the number of intersections of the two distributions is at least four, tests statistics with $m \geq 4$ have an advantage.

Appendix H. Simulations of Monotone Relationships

Four monotone relationships are presented in Figure 12. Figure 13 shows that the differences for the summation variants between ADP and DDP are negligible, and that the power decreases with m for all of these variants. Similar conclusions hold for the maximum variants in Table 10. Table 10 further shows that for a linear relationship, Pearson, Spearman, Hoeffding and dCov are clearly

		Minimum p-value							
N	Setup	Max aggregation	Sum Aggregation	Wilcoxon	KS	CVM	AD	HHG	DS
500	Normal shift	0.59	0.82	0.91	0.81	0.89	0.90	0.85	0.73
500	Normal scale	0.76	0.85	0.05	0.36	0.46	0.77	0.88	0.83
500	Normal shift & scale	0.83	0.90	0.46	0.69	0.74	0.88	0.92	0.90
100	Normal shift	0.39	0.58	0.68	0.53	0.65	0.67	0.60	0.49
100	Normal scale	0.51	0.59	0.06	0.20	0.23	0.41	0.63	0.54
100	Normal shift & scale	0.54	0.64	0.49	0.51	0.58	0.64	0.68	0.62

Table 8: Power of competitors (columns 4–9), along with the minimum p-value using the M_m p-values (column 1) and the S_m p-values (column 2).

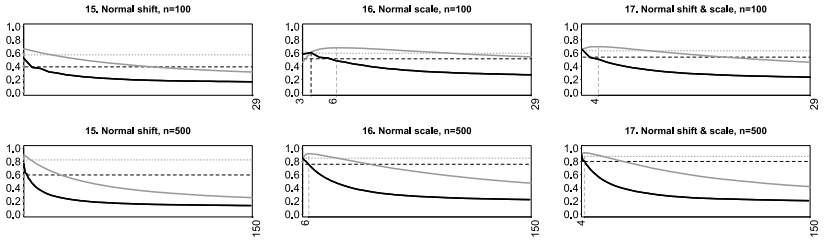


Figure 9: Estimated power for the M_m (black) and S_m (grey) statistics for $m \in \{2, \ldots, 149\}$ for the Gaussian shift difference (first column, $N(0,1)$ versus $N(0.5,1)$ for $N = 100$, and versus $N(0.3,1)$ for $N = 500$), Gaussian scale difference (second column, $N(0,1)$ versus $N(0, 0.6^2)$ for $N = 100$, and versus $N(0, 0.75^2)$ for $N = 500$), and for the Gaussian shift and scale difference (third column, $N(0,1)$ versus $N(0.36, 0.7^2)$ for $N = 100$, and versus $N(0.2, 0.8^2)$ for $N = 500$). The power of the minimum p-value is the horizontal dashed black line when it combines the p-values based on M_m, and the horizontal dotted grey line when it combines the p-values based on S_m. The vertical lines show the optimal m for M_m (grey) and S_m (black).

superior, but for the remaining three monotone relationships $\min_{m \in \{2,\ldots,m_{\max}\}} p_m$ has quite good power properties.

Appendix I. Comparisons of DCov/HHG on Ranks and on Data

In Section 4 we only considered distribution-free competitors. Therefore, we applied dCov and HHG on ranks rather than on data, even though they were designed as permutation tests on data. Usually, the computational advantage of performing the tests on $(rank(X), rank(Y))$ instead of on (X, Y), due to the distribution-free property of the tests on $(rank(X), rank(Y))$, comes at a cost of lower power, as noted by Székely et al. (2007). Table 11 shows a power comparison of these two permutation tests on ranks and on data. The power in most settings is indeed greater when the tests are used on data, and the maximal difference is almost 30% (in the Spiral setting for HHG) in favour of using the data. However, in some settings the power is actually larger for the test on ranks, e.g.,

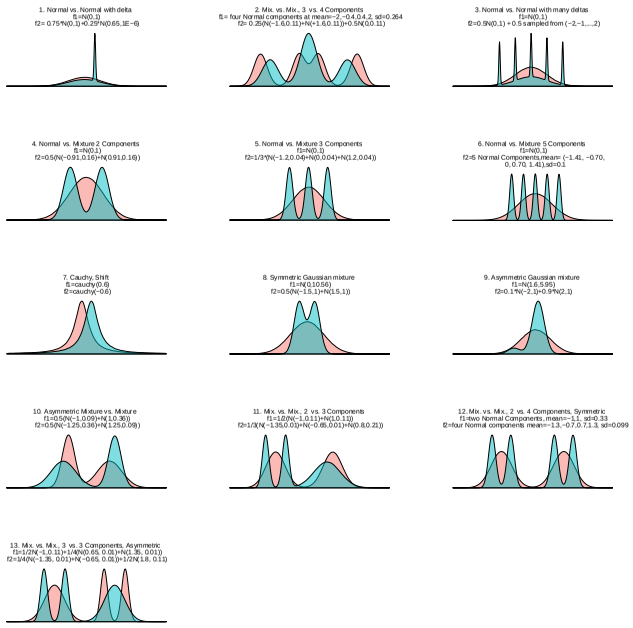


Figure 10: The two distributions in 13 different setups considered for $N = 100$, which differ in the number of intersections of the densities, the range of support where the differences lie, and the whether they are symmetric or not.

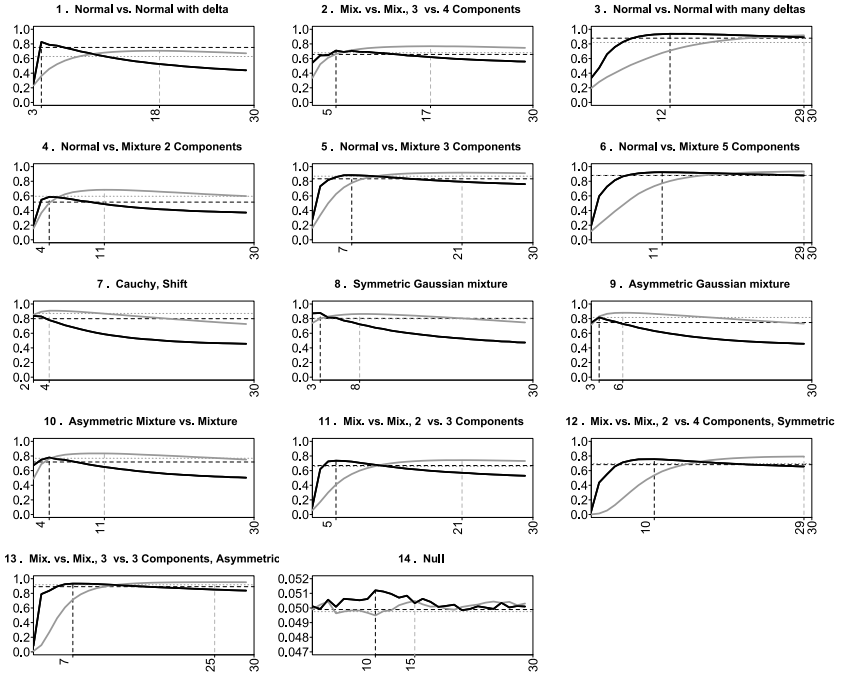


Figure 11: Estimated power with $N = 100$ sample points for the M_m (black) and S_m (grey) statistics for $m \in \{2, \ldots, 29\}$ for the setups of Figure 10. The power of the minimum p-value is the horizontal dashed black line when it combines the p-values based on M_m, and the horizontal dotted grey line when it combines the p-values based on S_m. The vertical lines show the optimal m for M_m (black) and S_m (grey).

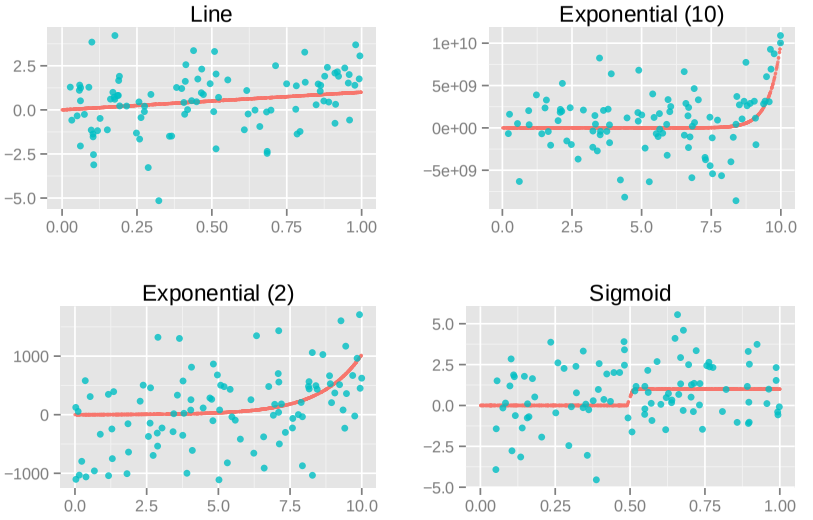


Figure 12: Bivariate monotone relationships (in red), along with a sample of $N = 100$ noisy observations (in blue).

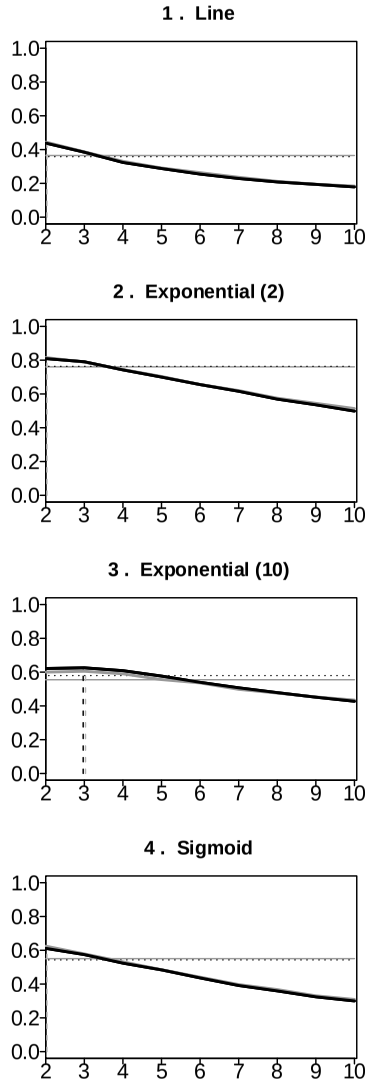


Figure 13: Estimated power for the DDP (black) and ADP (gray) summation variants, using the likelihood ratio score, for the setups from Figure 12, for a sample of size $N = 100$. The horizontal lines are the power of $\min_{m\in\{2,..,m_{\max}\}} p_m$ using DDP (dotted black) and using ADP (gray) .

		Min. p-value aggregation							
	Setup	by Max	by Sum	Wilcox	KS	CVM	AD	HHG	DS
1	Normal vs. Normal with delta	<u>0.752</u>	0.628	0.187	0.331	0.294	0.265	0.433	0.660
2	Mix. Vs. Mix., 3 Vs. 4 Components	0.656	<u>0.680</u>	0.000	0.003	0.000	0.063	0.465	0.505
3	Normal vs. Normal with many deltas	<u>0.878</u>	0.819	0.053	0.125	0.100	0.165	0.281	0.342
4	Normal vs. Mixture 2 Components	0.515	<u>0.595</u>	0.052	0.231	0.153	0.154	0.397	0.382
5	Normal vs. Mixture 3 Components	0.834	<u>0.869</u>	0.053	0.177	0.106	0.134	0.317	0.534
6	Normal vs. Mixture 5 Components	0.879	<u>0.880</u>	0.051	0.123	0.084	0.100	0.200	0.373
7	Cauchy, Shift	0.799	0.871	0.847	0.917	0.920	0.893	<u>0.933</u>	0.846
8	Symmetric Gaussian mixture	0.803	0.798	0.035	0.182	0.191	0.491	0.727	<u>0.812</u>
9	Asymmetric Gaussian mixture	0.747	0.816	0.046	0.369	0.407	0.593	<u>0.845</u>	0.740
10	Asymmetric Mixture vs. Mixture	0.718	<u>0.769</u>	0.000	0.157	0.128	0.306	0.655	0.652
11	Mix. Vs. Mix., 2 Vs. 3 Components	<u>0.670</u>	0.656	0.000	0.032	0.011	0.031	0.129	0.441
12	Mix. Vs. Mix., 2 Vs. 4 Comp., Sym.	0.682	<u>0.696</u>	0.000	0.000	0.000	0.000	0.005	0.238
13	Mix. Vs. Mix., 3 Vs. 3 Comp., Asym.	0.891	<u>0.917</u>	0.000	0.000	0.000	0.000	0.053	0.575
14	Null	0.050	0.050	0.049	0.039	0.049	0.050	0.050	0.041

Table 9: Power of competitors (columns 4–9), along with the minimum p-value statistic using the M_m p-values (column 1) and the S_m p-values (column 2), for $N = 100$. The standard error was at most 0.0035. The advantage of the test based on the minimum p-value is large when the number of intersections of the two densities is at least four (setups of rows 2,3, 4,5,6,10,11,12, and 13). The best competitors are HHG and DS, but HHG is essentially an $m \leq 3$ test, and DS penalizes large ms severely, therefore in setups where $m \geq 4$ partitions are better they can perform poorly. Among the two variants in columns 1 and 2, the better choice clearly depends on the range of support in which the differences in distributions occur: aggregation by maximum has better power when the difference between the distributions is very local (setups of rows 1 and 3), and aggregation by summation has better power otherwise. The highest power per row is underlined.

Test	Line	Exp2x	Exp10x	Sigmoid
$\min_{m\in\{2,\ldots,10\}} p_m$ using DDP	0.358	0.763	0.580	0.543
$\min_{m\in\{2,\ldots,10\}} p_m$ using ADP	0.365	0.760	0.555	0.550
Spearman	0.459	0.758	0.396	0.630
Hoeffding	0.446	0.750	0.409	0.637
MIC	0.282	0.198	0.312	0.130
dCov	0.433	0.746	0.395	0.637
HHG	0.337	0.706	0.509	0.545
$M^{DDP}_{2\times2}$	0.287	0.688	0.678	0.438
$M^{DDP}_{3\times3}$	0.203	0.579	0.569	0.355
$M^{DDP}_{4\times4}$	0.177	0.511	0.479	0.301
$M^{ADP}_{2\times2}$	0.294	0.715	0.746	0.440

Table 10: The power of competitors (rows 3–7), along with the minimum p-value statistic based on DDP (row 1) and on ADP (row 2) and the different maximum variants (rows 8–11), for $N = 100$. The standard error is at most 0.011.

in the Heavisine and 5Clouds settings for HHG, where the difference is 10% and 16%, respectively, in favour of using the ranked observations. Comparing the power of the HHG and dCov tests on data (in Table 11) to the power of our suggested minimum p-value statistic (in Table 2), we see that although dCov on data may have more power than dCov on ranks, it has far less power than HHG on data, and that HHG on data has less power than our test when the relationship is more complex, especially in the Sine, Heavisine, Spiral and Circles examples.

	dCov		HHG		
Setup	on data	on ranks	on data	on ranks	$\min_{\{m\in 2,\ldots,10\}} p_m$
W	0.467	0.351	0.876	0.813	0.655
Diamond	0.136	0.076	0.997	0.968	0.919
Parabola	0.418	0.369	0.727	0.795	0.847
2Parabolas	0.183	0.120	0.873	0.726	0.844
Circle	0.001	0.003	0.791	0.858	0.886
Cubic	0.631	0.612	0.660	0.742	0.731
Sine	0.412	0.427	0.803	0.788	0.995
Wedge	0.471	0.327	0.755	0.661	0.595
Cross	0.184	0.138	0.795	0.704	0.704
Spiral	0.182	0.130	0.598	0.335	0.949
Circles	0.054	0.059	0.479	0.357	0.999
Heavisine	0.476	0.470	0.471	0.570	0.710
Doppler	0.747	0.736	0.901	0.914	0.949
5Clouds	0.000	0.001	0.738	0.903	0.996
4Clouds	0.050	0.050	0.050	0.050	0.051

Table 11: Power of the competitors dCov and HHG on ranks as well as on data for $N = 100$ (columns 2-4). The results on ranks are not identical numerically (though very close) to those of Table 2 due to the use of a different seed to generate the data for these same settings. The tests on data have greater power than on ranks in most, but not all, examples. HHG on data has the best power out of these four competitors, but like the HHG on ranks it has a disadvantage in comparison with our novel minimum p-value test when the relationship is more complex, especially in the Sine, Heavisine, Spiral, Circles, and 5Clouds examples (the power for this test was reproduced in column 6 here from Table 2 for convenience).

References

T.W. Anderson and D.A. Darling. Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *The Annals of Mathematical Statistics*, 23(2):193–212, 1952.

L. Baringhaus and C. Franz. On a new multivariate two-sample test. *Journal of Multivariate Analysis*, 88:190–206, 2004.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate - a practical and powerful approach to multiple testing. *J. Roy. Stat. Soc. B Met.*, 57 (1):289–300, 1995.

J. Blum, J. Kiefer, and M. Rosenblatt. Distribution free tests of independence based on the sample distribution function. *The Annals of Mathematical Statistics*, pages 485–498, 1961.

K.P. Chwialkowski, D. Sejdinovic, and A. Gretton. A wild bootstrap for degenerate kernel tests. In Z. Ghahramani, M. Welling, C. Cortes, N.d. Lawrence, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3608–3616. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5452-a-wild-bootstrap-for-degenerate-kernel-tests.pdf`.

D.A. Darling. The kolmogorov-Smirnov, Cramer-von Mises tests. *The Annals of Mathematical Statistics*, 28(4):823–838, 1957.

D. Donoho and I. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90(432):1200–1224, 1995.

A. Feuerverger. A consistent test for bivariate dependence. *International Statistical Review*, 61(3): 419–433, 1993.

M. Gorfine, R. Heller, and Y. Heller. Comment on detecting novel associations in large data sets by Reshef et al., science. *Available: `http://iew3technionacil/~gorfinm/files/science6.pdf`*, 2011.

A. Gretton and L. Gyorfi. Consistent nonparametric tests of independence. *Journal of Machine Learning Research*, 11:1391–1423, 2010.

A. Gretton, K.M. Bogwardt, M.J. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two-sample problem. *Advances in Neural Information Processing Systems (NIPS)*, 19, 2007.

A. Gretton, K. Fukumizu, C.H. TEO, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. *Advances in Neural Information Processing Systems*, 20:585–592, 2008.

A. Gretton, K.M. Borgwardt, M.J. Rasch, Schölkopf, and A. Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13:723–773, 2012a.

A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B.K. Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *Advances in neural information processing systems*, pages 1205–1213, 2012b.

Z Harchaoui, F. Bach, and E. Moulines. Testing for homogeneity with kernel fisher discriminant analysis. *Advances in Neural Information Processing Systems (NIPS), long version: arXiv:0804.1026v1,*, 20:609–616, 2008.

R. Heller, Y. Heller, and M. Gorfine. A consistent multivariate test of association based on ranks of distances. *Biometrika*, 100(2):503–510, 2013.

W. Hoeffding. A non-parametric test of independence. *The Annals of Mathematical Statistics*, 19 (4):546–557, 1948a.

W. Hoeffding. A non-parametric test of independence. *The Annals of Mathematical Statistics*, 19 (4):546–557, 1948b.

T.R. Hughes, M.J. Marton, A.R. Jones, C.J. Roberts, R. Stoughton, C.D. Armour, H.A. Bennett, E. Coffey, H. Dai, Y.D. He, et al. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–126, 2000.

B. Jiang, C. Ye, and J. Liu. Non-parametric k-sample tests via dynamic slicing. *Journal of the American Statistical Association*, DOI:10.1080/01621459.2014.920257, 2014.

J. Kinney and G. Atwal. Equitability, mutual information, and the maximal information coefficient. *Proceedings of the national academy of sciences of the USA*, doi:10.1073, 2014.

B. Laurent and P. Massart. Adaptive estimation of a quadratic functional by model selection. *The Annals of Statistics*, 28(5):1302–1338, 2000.

E.L. Lehmann and J.P. Romano. *Testing Statistical Hypotheses, 3rd Edition*. Springer, New York, 2005.

M. Newton. Introducing the discussion paper by Szekely and Rizzo. *The Annals of Applied Statistics*, 3 (4):1233–1235, 2009.

L. Paninski. Estimation of entropy and mutual information. *Neural Computation*, 15(6):1191–1253, 2003.

B. Póczos, Z. Ghahramani, and J. Schneider. Copula-based kernel dependency measures. In *International Conference on Machine Learning (ICML)*, 2012.

A.N. Pettitt. A two-sample Anderson–Darling rank statistic. *Biometrika*, 63(1):161–168, 1976.

D.N. Reshef, Y.A. Reshef, H.K. Finucane, S.R. Grossman, G. McVean, P.J. Turnbaugh, E.S. Lander, M. Mitzenmacher, and P.C. Sabeti. Detecting novel associations in large data sets. *Science*, 334 (6062):1518–1524, 2011.

F.W. Scholz and M.A. Stephens. K-sample Anderson-Darling tests. *Journal of the American Statistical Association*, 82(399):918–924, 1987.

D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu. Equivalence of distance-based and rkhs-based statistics in hypothesis testing. *Annals of Statistics*, 41 (5):2263–2291, 2013.

N. Simon and R. Tibshirani. Comment on detecting novel associations in large data sets by Reshef et al., science. *arXiv:1401.7645*, 2011.

R. Steuer, J. Kurths, C. Daub, J. Weise, and J. Selbig. The mutual information: detecting and evaluating dependencies between variables. *Bioinformatics*, 18(suppl 2):S231–S240, 2002.

G. Székely and M. Rizzo. Testing for equal distributions in high dimensions. *InterStat*, 2004.

G. Székely and M. Rizzo. Brownian distance covariance. *The Annals of Applied Statistics*, 3 (4): 1236–1265, 2009a.

G. Székely, M. Rizzo, and N. Bakirov. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35:2769–2794, 2007.

Gabor J Székely and Maria L Rizzo. Brownian distance covariance. *The annals of applied statistics*, 3(4):1236–1265, 2009b.

O. Thas and J. Ottoy. An extension of the Anderson-Darling k-sample test to arbitrary sample space partition sizes. *Journal of Statistical Computation and Simulation*, 74(9):651–665, 2007.

O. Thas and J.P. Ottoy. A nonparamteric test for independence based on sample space partitions. *Communcations in Statistics - Simulation and Computation*, 33 (3):711–728, 2004.

Q. Vu, B. Yu, and R. Kass. Coverage-adjusted entropy estimation. *Statistics in medicine*, 26: 4039–4060, 2007.

K. Yu, F. Liang, J. Ciampa, and N. Chatterjee. Efficient p-value evaluation for resampling-based tests. *Biostatistics*, 12(3):582–593, 2011.

Journal of Machine Learning Research 17 (2016) 1-39 Submitted 11/14; Revised 11/15; Published 4/16

A Gibbs Sampler for Learning DAGs

Robert J. B. Goudie ROBERT.GOUDIE@MRC-BSU.CAM.AC.UK
Medical Research Council Biostatistics Unit
Cambridge CB2 0SR, UK

Sach Mukherjee SACH.MUKHERJEE@DZNE.DE
German Centre for Neurodegenerative Diseases (DZNE)
Bonn 53175, Germany

Editor: Peter Spirtes

Abstract

We propose a Gibbs sampler for structure learning in directed acyclic graph (DAG) models. The standard Markov chain Monte Carlo algorithms used for learning DAGs are random-walk Metropolis-Hastings samplers. These samplers are guaranteed to converge asymptotically but often mix slowly when exploring the large graph spaces that arise in structure learning. In each step, the sampler we propose draws entire sets of parents for multiple nodes from the appropriate conditional distribution. This provides an efficient way to make large moves in graph space, permitting faster mixing whilst retaining asymptotic guarantees of convergence. The conditional distribution is related to variable selection with candidate parents playing the role of covariates or inputs. We empirically examine the performance of the sampler using several simulated and real data examples. The proposed method gives robust results in diverse settings, outperforming several existing Bayesian and frequentist methods. In addition, our empirical results shed some light on the relative merits of Bayesian and constraint-based methods for structure learning.

Keywords: structure learning, DAGs, Bayesian networks, Gibbs sampling, Markov chain Monte Carlo, variable selection

1. Introduction

We consider structure learning for graphical models based on directed acyclic graphs (DAGs). The basic structure learning task can be stated simply: given data $\mathbf{X}$ on p variables, assumed to be generated from a graphical model based on an unknown DAG G, the goal is to make inferences concerning the edge set of G (the vertex set is treated as fixed and known). Structure learning appears in a variety of applications (for an overview see Korb and Nicholson, 2011) and is a key subtask in many analyses involving graphical models, including causal inference.

In a Bayesian framework, structure learning is based on the posterior distribution $P(G \mid \mathbf{X})$ (Koller and Friedman, 2009). The domain of the distribution is the space $\mathcal{G}$ of all DAGs with p vertices. The size of the space grows super-exponentially with p, precluding exhaustive enumeration for all but the smallest problems. Markov chain Monte Carlo (MCMC) methods are widely used to sample DAGs and thereby approximate the posterior $P(G \mid \mathbf{X})$. Available methods include MC^3 (Madigan and York, 1995) and related samplers (Giudici and Castelo, 2003; Grzegorczyk and Husmeier, 2008) and algorithms that sample from the space of total orders (Friedman and Koller, 2003). Order-space sampling entails some restrictions on graph priors (Eaton and Murphy, 2007; Ellis and Wong, 2008), because the number of total orders with which a DAG is consistent is not constant. Order-space approaches have more recently led to exact methods based on dynamic programming (Koivisto and Sood, 2004; Parviainen and Koivisto, 2009; Tian and He, 2009; Tamada et al., 2011; Parviainen and Koivisto, 2013; Nikolova et al., 2013). These are currently feasible only in small domain settings (typically $p < 20$ but $p < 30$ is feasible on large cluster computers) and usually share the same restrictions on graph priors as order-space samplers. Frequentist constraint-based methods such as the PC-algorithm (Spirtes et al., 2000; Colombo and Maathuis, 2014) and the approach of Xie and Geng (2008) are an alternative. These methods make firm decisions about the DAG structure via a series of tests of conditional independence.

In this paper we propose a Gibbs sampler for structure learning of DAGs that ameliorates key deficiencies in existing samplers. Random-walk Metropolis-Hastings algorithms currently used for structure learning propose 'small' changes at each iteration, which can leave the algorithms unable to escape local modes. The Gibbs sampler proposed here considers the parents of a set of nodes as a single component (a so-called 'block'), sampling an entire parent set for each node in the block in one step. These 'large' moves are sampled from the appropriate conditional posterior distribution. This enables the sampler to efficiently locate and explore areas of significant posterior mass. The method is based on the simple heuristic that the parents of a node are similar to the covariates or inputs in variable selection, with the node as the output variable, but accounts for acyclicity exactly so that the equilibrium distribution is indeed the correct posterior distribution over DAGs. The sampler does not impose restrictions on priors or graph space beyond those (maximum in-degree, modular prior) common to most samplers for structure learning (e.g. Friedman and Koller, 2003; Ellis and Wong, 2008; Grzegorczyk and Husmeier, 2008). The maximum in-degree restriction formally precludes large-sample consistency in the general case, but facilitates effective and robust inference by reducing the size of the model space (see Discussion).

2. Notation and Model

Let G denote a DAG with vertex set $V(G) = \{1, \ldots, p\}$, and directed edge set $E(G) \subset V(G) \times V(G)$; often we will refer to vertex and edge sets simply as V and E respectively, leaving G implicit. The binary adjacency matrix corresponding to G is denoted A^G, with entries specified as $A^G_{uv} = 1 \iff (u, v) \in E(G)$ and diagonal entries equal to zero. For inference, G is treated as a latent graph in the space $\mathcal{G}$ of all possible DAGs with p vertices. When a DAG is used to define a probabilistic graphical model (a Bayesian network), each vertex (or node; we use both terms interchangeably) is associated with a component of a p-dimensional random vector $X = (X_1 \ldots X_p)^{\mathrm{T}}$. X_Z denotes the random variables (RVs) corresponding to variable indices $Z \subseteq \{1, \ldots, p\}$. We use bold type for the corresponding data; n samples are collected in the $n \times p$ matrix $\mathbf{X}$, with $\mathbf{X}_u$ denoting the column corresponding to variable u and $\mathbf{X}_Z$ the submatrix corresponding to variable index set $Z \subseteq \{1, \ldots, p\}$ (for notational simplicity we assume columns are ordered by increasing variable index).

A graph G

$$A^G = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$\mathrm{pa}_1^G = \emptyset$
$\mathrm{pa}_2^G = \{1\}$
$\mathrm{pa}_3^G = \{1, 2, 4\}$
$\mathrm{pa}_4^G = \emptyset$

Figure 1: Illustration of notation. A graph G, with vertex set $V = \{1, 2, 3, 4\}$, edge set $E = \{(1, 2), (1, 3), (2, 3), (4, 3)\}$ and adjacency matrix A^G as shown, can be specified using parent sets as $G = \langle \mathrm{pa}_1 = \emptyset, \mathrm{pa}_2 = \{1\}, \mathrm{pa}_3 = \{1, 2, 4\}, \mathrm{pa}_4 = \emptyset \rangle$ or abbreviated to $G = \langle \emptyset, \{1\}, \{1, 2, 4\}, \emptyset \rangle$ (see text for description of notation). For the vertex subset $Z = \{2, 4\}$, $\mathrm{pa}_Z^G = (\{1\}, \emptyset)$ and $\mathrm{pa}_{-Z}^G = (\emptyset, \{1, 2, 4\})$ and thus the graph can be specified as $G = \langle \mathrm{pa}_Z = \mathrm{pa}_Z^G, \mathrm{pa}_{-Z} = \mathrm{pa}_{-Z}^G \rangle$. We abbreviate this as $G = \langle \mathrm{pa}_Z^G, \mathrm{pa}_{-Z}^G \rangle$.

The proposed Gibbs sampler changes entire parent sets *en masse*, not just for one node but for a set of nodes, and so it will often be convenient and natural to specify a DAG G in terms of the parents of each node. Let $\mathrm{pa}_v^G = \{u : (u, v) \in E(G)\}$ denote the set of nodes that are parents of node v in G. A tuple of p parent sets $\langle \mathrm{pa}_1 = \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p = \mathrm{pa}_p^G \rangle$ fully specifies the edge set E, since $u \in \mathrm{pa}_v^G \iff (u, v) \in E(G)$. Thus, structure learning can be viewed as inference of parent sets pa_v. The parent set pa_v takes values in the power set $\mathcal{P}(V \setminus \{v\})$ of nodes excluding node v, subject to acyclicity. We will usually suppress the labels, and simply write $\langle \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p^G \rangle$ when specifying parent sets. Since pa_v^G is a subset of the variable indices, we can use $\mathbf{X}_{\mathrm{pa}_v^G}$ to denote the corresponding data submatrix.

We denote the tuple of parent sets of a set of nodes $Z = \{z_1, \ldots, z_s\} \subseteq V$ by $\mathrm{pa}_Z^G = (\mathrm{pa}_z^G)_{z \in Z}$. This is a tuple of s components (for notational convenience we take these to be ordered by increasing vertex index), each of which is the parent set for the corresponding node in G. We wish to make inferences about pa_Z, which takes values in $\mathcal{P}(V \setminus \{z_1\}) \times \ldots \times \mathcal{P}(V \setminus \{z_s\})$, subject to acyclicity. The tuple of parent sets for the complement $Z^{\mathsf{C}} = V \setminus Z$ is denoted $\mathrm{pa}_{-Z}^G = (\mathrm{pa}_z^G)_{z \in Z^{\mathsf{C}}}$. Clearly, a tuple $\langle \mathrm{pa}_{Z_1}^G, \ldots, \mathrm{pa}_{Z_s}^G \rangle$ of tuples of parent sets specifies the parents of *every* node in a graph whenever $\{Z_1, \ldots, Z_s\}$ forms a partition of V; the entire edge set can thus be specified by such a tuple. In particular, note that any graph G can be specified as $\langle \mathrm{pa}_Z^G, \mathrm{pa}_{-Z}^G \rangle$ for any $Z \subset V$. Some of the notation we use is illustrated in Figure 1.

Our statistical formulation for Bayesian networks is standard. We briefly summarize the main points and refer the reader to the references below for details. Given a DAG G, the joint distribution of X is $p(X \mid G, \{\theta_v\}) = \prod_{v \in V} p(X_v \mid X_{\mathrm{pa}_v^G}, \theta_v)$, i.e. the joint distribution factors over nodes, and each node is conditioned on its parents in G, parameterized by θ_v. For structural inference, interest focuses on the posterior distribution $P(G \mid \mathbf{X})$. This is proportional to the product of the marginal likelihood $p(\mathbf{X} \mid G)$ and a graph prior $\pi(G)$. Our sampler is compatible with essentially any specific model, but inherits computational costs associated with evaluation of the relevant quantities. In all examples we assume conjugate priors for θ_v, as well as local parameter independence and modularity; this leads to a closed-form marginal likelihood in both multinomial (Heckerman et al., 1995) and Gaussian

(Geiger and Heckerman, 1994) cases. Computations are simplified if the graph prior is modular (Friedman and Koller, 2003), meaning it factorises as $\pi(G) = \prod_{v \in V} \pi_v(\mathrm{pa}_v^G)$; we assume modular priors in all examples. Note that the prior is not specified over the space of orders and so is not subject to the restrictions this entails (Ellis and Wong, 2008; Eaton and Murphy, 2007). Under these assumptions, the posterior factorises across nodes as

$$\begin{aligned} P(G \mid \mathbf{X}) &= P(\mathrm{pa}_1 = \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p = \mathrm{pa}_p^G \mid \mathbf{X}) \\ &\propto \prod_{v \in V} p(\mathbf{X}_v \mid \mathbf{X}_{\mathrm{pa}_v^G}) \pi_v(\mathrm{pa}_v^G), \end{aligned}$$

where $p(\mathbf{X}_v \mid \mathbf{X}_{\mathrm{pa}_v^G})$ is the marginal likelihood for node v given the graph $G = \langle \mathrm{pa}_1^G, \ldots, \mathrm{pa}_p^G \rangle$. The distribution $P(G \mid \mathbf{X})$ is the target distribution for our sampler.

3. A Gibbs Sampler for Structure Learning

In this section we provide a high-level description of the Gibbs sampler for structure learning. We first recall the standard random-walk Metropolis-Hastings sampler (known as MC^3) and then describe a naïve Gibbs sampler, which offers no gains over MC^3, but prepares the ground for the introduction of ideas from the Gibbs sampling literature. Specifically we discuss a strategy known as blocking that can improve mixing in Gibbs samplers and show how to use blocking for structure learning of DAGs. Several points relating to computation are central to developing a practical Gibbs sampler in this setting, but for clarity of exposition we defer discussion of computational aspects to Section 4.

3.1 MC^3 Sampler

The standard sampler for structure learning of DAGs is MC^3 (Madigan and York, 1995). This is a classical Metropolis-Hastings sampler with proposal G' drawn uniformly at random from the set $\mathrm{neigh}(G)$ of DAGs that differ from the current DAG G by the addition or removal of a single edge. The acceptance probability is $\min(1, r(G', G))$, where

$$r(G', G) = \min \left\{ 1, \frac{P(G' \mid \mathbf{X}) \, |\mathrm{neigh}(G')|^{-1}}{P(G \mid \mathbf{X}) \, |\mathrm{neigh}(G)|^{-1}} \right\}.$$

Variants of MC^3 include single-edge direction reversal proposals (Giudici and Castelo, 2003).

3.2 A Naïve (and Inefficient) Gibbs Sampler

Constructing a Gibbs sampler that is analogous to MC^3 is straightforward. The posterior distribution on DAGs is a joint distribution over the off-diagonal entries in the adjacency matrix A^G, i.e. over the $p(p-1)$ binary RVs A_{uv}^G, $u \neq v$. At each iteration, MC^3 can be thought of as proposing to toggle the value A_{uv}^G for some $u \neq v$, subject to the restriction that the resulting graph must be acyclic.

A simple Gibbs sampler works in a similar way. At each step, a move from graph G to a new graph G' is chosen by sampling from the conditional distribution of $A_{uv}^{G'}$ (for some $u \neq v$) given the rest of the graph. If $(u, v) \in E(G)$, define the graph $G^{(u,v)}$ to be identical to G, and $G^{-(u,v)}$ to be the graph that differs from G only in lacking an edge from u to

v; conversely, if $(u,v) \notin E(G)$, define $G^{-(u,v)}$ to be identical to G, and $G^{(u,v)}$ to be the graph that differs from G only in including an edge from u to v. If $G^{(u,v)}$ is cyclic, $G^{-(u,v)}$ is sampled as the new graph G' with probability 1. If $G^{(u,v)}$ is acyclic, the conditional distribution of $A^{G'}_{uv}$ is Bernoulli, i.e.

$$P(A^{G'}_{uv} = a \mid A^{G}_{-uv}) = \begin{cases} \dfrac{P(G^{-(u,v)} \mid \mathbf{X})}{P(G^{-(u,v)} \mid \mathbf{X}) + P(G^{(u,v)} \mid \mathbf{X})} & a = 0, \\ \dfrac{P(G^{(u,v)} \mid \mathbf{X})}{P(G^{-(u,v)} \mid \mathbf{X}) + P(G^{(u,v)} \mid \mathbf{X})} & a = 1. \end{cases}$$

The choice of u and v can either be made sequentially (systematically) or randomly (Roberts and Sahu, 1997); in this paper, random-scan Gibbs samplers are used throughout. We prove convergence of the Gibbs sampler to the target distribution in Appendix A.

3.3 Inefficiency in Gibbs Sampling

The mixing of Metropolis-Hastings samplers depends upon the proposal distribution, which for convenience is often chosen as a random-walk. In contrast, Gibbs samplers make moves according to conditional distributions that reflect local structure of the target distribution. Nonetheless, Gibbs sampling is not always efficient. In particular, correlation between the components being sampled can lead to inefficiency. To see this, consider a Gibbs sampler for a multivariate continuous distribution with highly correlated components. At each step, a single component is sampled according to its conditional distribution, but since it is strongly correlated with other component(s), the conditional is concentrated on only a small part of the support. This means the sampler is likely to make only small moves. Analogous issues arise with discrete distributions.

For graphical models based on DAGs, there may be strong dependence between the edge indicators A^G_{uv}, particularly for the collections of RVs corresponding to parent sets. For example, there may be RVs X_u and X_v that in combination score highly as parents of node w, but not individually. Then, A^G_{uw} and A^G_{vw} will be correlated. In addition, the acyclicity restriction may induce strong dependence. For example, suppose two RVs X_u and X_v are strongly correlated and both edges (u,v) and its reverse (v,u) have high posterior probability. If the edge (u,v) is present, the probability of it being removed is low, but its presence precludes the reversed edge (v,u) from being added. This possibility motivates the edge reversal move used in variants of MC^3 (Giudici and Castelo, 2003). Figure 2 shows a three node scenario. Here, both graphs (a) and (b) have high probability. Since reversing the edge (w,x) forms a cycle in (a), moves that consider the parents of only the pair w and x at the same time will not move between graphs (a) and (b) easily. Samplers that alter only a single edge indicator, such as MC^3, will also fail. In contrast, sampling the parents of all three nodes jointly would make it easy to move between graphs (a) and (b). Decorrelating transformations could alleviate such problems, but in general finding a suitable transformation is difficult, and for DAGs must of course encapsulate the requirement for acyclicity.

Figure 2: Illustrative scenario in which small local moves limit transitions between two regions of high probability. If both graphs (a) and (b) have high probability, the near-cyclic nature of the graphs makes transitions between (a) and (b) difficult.

3.4 A Gibbs Sampler with Blocking

We address the deficiencies of the naïve Gibbs sampler by grouping a number of the components together and sampling from their joint conditional distribution. In Gibbs sampling, this is known as *blocking*. Sampling from such a joint conditional can ameliorate difficulties caused by correlations between components because the joint conditional naturally accounts for the correlation structure. In the multivariate normal case, Roberts and Sahu (1997) have shown that blocking improves convergence for random-scan Gibbs sampling.

In principle, any group of components can be taken as a block, but for the algorithm to be useful in practice, sampling from a block's joint conditional distribution must be feasible, and ideally simple. The blocks that we consider correspond to groups (specifically tuples) of parent sets, so that the parent sets of several nodes are considered simultaneously. It is natural that each block is a tuple of parent sets because, as described in Section 2, we can specify a graph G using a tuple $\langle \mathrm{pa}^G_1, \ldots, \mathrm{pa}^G_p \rangle$ of parent sets. It is therefore convenient to describe the sampler using this alternative graph specification, in which $G = \langle \mathrm{pa}^G_1, \ldots, \mathrm{pa}^G_p \rangle$.

We denote the set of q nodes whose parent sets will be sampled together as a block by $W = \{w_1, \ldots, w_q\} \subseteq V$. Suppose the current graph is $G = \langle \mathrm{pa}^G_W, \mathrm{pa}^G_{-W} \rangle$ (recall that any graph can be written in this way with respect to any partition of the node set). A move to a new graph $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^{G'}_{-W} \rangle$ is formed by changing the parents of the nodes in W from $\mathrm{pa}^G_W = (\mathrm{pa}^G_{w_1}, \ldots, \mathrm{pa}^G_{w_q})$ to $\mathrm{pa}^{G'}_W = (\mathrm{pa}^{G'}_{w_1}, \ldots, \mathrm{pa}^{G'}_{w_q})$ and setting $\mathrm{pa}^{G'}_{-W} = \mathrm{pa}^G_{-W}$ (i.e. leaving the parents of nodes not in W unchanged). We sample the tuple $\mathrm{pa}^{G'}_W$ of parent sets jointly, conditional on the tuple pa^G_{-W} of parent sets of nodes not in W (that remain unchanged). In terms of the adjacency matrix A^G, each block consists of the indicators specifying the parents of the nodes in W, i.e. A^G_{vw} for $v \in V, w \in W, v \neq w$.

To construct a Gibbs sampler using these blocks, we need to find the conditional posterior distribution on the tuple pa_W of parent sets, given that the remaining tuple pa_{-W} of parent sets is set to pa^G_{-W}. The conditional distribution depends on whether the graph G' formed using the proposed parent sets is acyclic. We therefore introduce $P\mathrm{a}^G_W$ to denote the set of permissible tuples, i.e. tuples $\mathrm{pa}^{G'}_W$ of parent sets such that $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \rangle$ is acyclic. For tuples $\mathrm{pa}^{G'}_W \notin P\mathrm{a}^G_W$, the conditional probability is 0. For $\mathrm{pa}^{G'}_W \in P\mathrm{a}^G_W$, the

Algorithm 1 A Gibbs sampler for learning DAGs, with blocks

Initialise starting point $G_0 = \langle \mathrm{pa}_1^{G_0}, \ldots, \mathrm{pa}_p^{G_0} \rangle$
for t in 1 to N **do**
  Sample q nodes uniformly at random from V, and call this set of nodes W
  Sample $\mathrm{pa}_W^{G'}$ from $P(\mathrm{pa}_W = \mathrm{pa}_W^{G'} \mid \mathrm{pa}_{-W}^{G_{t-1}}, \mathbf{X})$
  Set $G_t \leftarrow G = \langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G_{t-1}} \rangle$
end for

conditional probability is

$$P(\mathrm{pa}_W = \mathrm{pa}_W^{G'} \mid \mathrm{pa}_{-W} = \mathrm{pa}_{-W}^{G}, \mathbf{X}) = \frac{P(\langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^{G} \rangle \mid \mathbf{X})}{P(\mathrm{pa}_{-W}^{G} \mid \mathbf{X})} = \frac{P(G' \mid \mathbf{X})}{\sum_{\mathrm{pa}_W^{G''} \in Pa_W^G} P(\mathrm{pa}_W^{G''}, \mathrm{pa}_{-W}^{G} \mid \mathbf{X})}. \quad (1)$$

This is the conditional distribution needed to specify the blocked Gibbs sampler; Algorithm 1 outlines the procedure at a high level, with the set W of nodes chosen uniformly at random at each step. Asymptotic convergence follows from the argument in Appendix A (in fact the requirements on the graph prior will be weaker than in the naïve case).

3.5 Tuning Parameters

To reduce the computational costs of structure learning it is common to set a maximum in-degree (e.g. Friedman and Koller, 2003; Grzegorczyk and Husmeier, 2008). We set a maximum in-degree $\kappa = 3$ in all empirical examples, except where stated otherwise (Section 5). This facilitates sampling from the conditional distribution in Equation 1 by reducing the computational cost of evaluating the normalising constant. We set the block size $q = |W| = 3$. While q can be chosen freely in principle, evaluating the conditional distribution in Equation 1 becomes unmanageable when q is large. Thus, both κ and q act as tuning parameters (and are in addition to any hyper-parameters in the Bayesian formulation).

3.6 Structure Learning and Variable Selection

It is interesting to note that when $q = 1$ (and thus $W = w$ for some $w \in V$) if no choice of parent set induces a cycle then $Pa_W^G = \mathcal{P}(V \setminus \{w\})$, the power set of the remaining nodes. In this case, the conditional distribution in Equation 1 can be viewed as the posterior distribution of a variable selection problem with response or output variable w, and the other variables as covariates or inputs. If the addition of particular nodes introduces a cycle, we have a constrained problem. In particular, suppose adding any of the nodes $Z \subset V$ as parents of node w would create a cycle. Then $Pa_W^G = \mathcal{P}(V \setminus (\{w\} \cup Z))$ and the conditional distribution in Equation 1 is a constrained variable selection problem in which the nodes Z are excluded from the set of candidate inputs.

4. Computational Aspects

Designing a computationally efficient sampler is not straightforward in this setting. To see why, note that to sample from the conditional in Equation 1 we need to be able to identify the set Pa_W^G of tuples of parent sets that is permissible (in the sense of maintaining acyclicity). The cardinality of this set is typically large, and the interdependence between the parent sets of each node in W makes decomposing the problem into subproblems non-trivial. A simple but naïve approach would list all possible parent sets for each node in W and check each such combination for cyclicity, but this approach is slow and cumbersome.

We propose a partitioning scheme that leads to a two-stage sampler. The key idea is to choose the partition of Pa_W^G so that, conditional on a component of the partition, the parents of each node are independent. This enables an efficient two-stage sampling method that we describe in Section 4.3. Acyclicity constraints are met by an efficient dynamic algorithm.

4.1 A Partition on Permissible Tuples of Parent Sets

We partition the set Pa_W^G of permissible tuples of parent sets as $Pa_W^G = \{Pa_W^{G,H_1}, \ldots, Pa_W^{G,H_\eta}\}$. Each component Pa_W^{G,H_h} is a set of permissible tuples of parent sets for nodes in W. It is convenient to label the partition components using secondary (unrelated to G) DAGs $H_h \in \mathcal{H}$, where $\mathcal{H}$ is the space of DAGs with $q = |W|$ vertices; η is the cardinality of $\mathcal{H}$. We describe the relationship between each Pa_W^{G,H_h} and DAG H_h using the following elements, illustrated in Figure 3 and Table 1:

- The reduced graph $\overline{G} = \langle \mathrm{pa}_1^{\overline{G}}, \ldots, \mathrm{pa}_p^{\overline{G}} \rangle$, which is a function of the graph G, and is identical to G except that edges directed into nodes in W are removed.

$$\mathrm{pa}_w^{\overline{G}} = \begin{cases} \emptyset & w \in W, \\ \mathrm{pa}_w^G & w \notin W \end{cases}$$

- The (reflexive) transitive closure, which is the directed graph T^G on nodes V with edges E^{T}, where $(u, v) \in E^{\mathrm{T}}$ if and only if $u = v$ or a (directed) path from u to v exists in G (i.e. there exists a sequence of nodes $z_1, \ldots, z_s \in V(G)$, with $z_1 = u$ and $z_s = v$, such that $(z_1, z_2), (z_2, z_3), \ldots, (z_{s-1}, z_s) \in E(G)$).

- The descendant nodes $\mathrm{de}_w^{\overline{G}} = \{v \in V : \mathrm{T}_{wv}^{\overline{G}} = 1\}$ and the non-descendant nodes $\mathrm{nd}_w^{\overline{G}} = \{v \in V : \mathrm{T}_{wv}^{\overline{G}} = 0\}$ of $w \in W$ in the reduced graph $\overline{G}$. Note that $w \in \mathrm{de}_w^{\overline{G}}$ and $w \notin \mathrm{nd}_w^{\overline{G}}$ by definition.

- The nodes $\mathrm{nd}^{\overline{G}} = \bigcap_{w \in W} \mathrm{nd}_w^{\overline{G}}$ that are not descendants in the reduced graph $\overline{G}$ of any node in W.

- The nodes $\mathrm{de}_w^{\overline{G},H} = \bigcup_{x \in \mathrm{pa}_w^H} \mathrm{de}_x^{\overline{G}}$ that are descendants in the reduced graph $\overline{G}$ of any node $x \in \mathrm{pa}_w^H$, for a given node $w \in W$. Each node $x \in \mathrm{pa}_w^H$ is a parent node of the node w in the graph $H = \langle \mathrm{pa}_{w_1}^H, \ldots, \mathrm{pa}_{w_q}^H \rangle$.

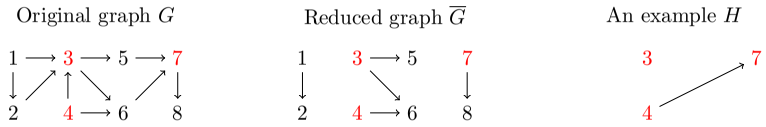


Figure 3: An illustrative example of the relevant graphs and sets, with $W = \{3, 4, 7\}$ shown in red. The edges into W are removed from the original graph G, to form $\overline{G}$. Here $\mathcal{H}$ is the set of all DAGs on the nodes $\{3, 4, 7\}$ and thus $\eta = |\mathcal{H}| = 25$. For H as shown, $\mathrm{nd}^{\overline{G}} = \{1, 2\}$.

	$\mathrm{de}_w^{\overline{G}}$	$\mathrm{nd}_w^{\overline{G}}$	$\mathrm{de}_w^{\overline{G},H}$	$\mathrm{de}_{-w}^{\overline{G},H}$	$Q_w^{\overline{G},H}$	Condition (B)
$w = 3$	$\{3, 5, 6\}$	$\{1, 2, 4, 7, 8\}$	$\emptyset$	$\{3, 4, 5, 6, 7, 8\}$	$\{1, 2\}$	No restriction
$w = 4$	$\{4, 6\}$	$\{1, 2, 3, 5, 7, 8\}$	$\emptyset$	$\{3, 4, 5, 6, 7, 8\}$	$\{1, 2\}$	No restriction
$w = 7$	$\{7, 8\}$	$\{1, 2, 3, 4, 5, 6\}$	$\{4, 6\}$	$\{3, 5, 6, 7, 8\}$	$\{1, 2, 4\}$	$\mathrm{pa}_7^{G'} \cap \{4\} \neq \emptyset$

Table 1: The relevant sets and requirements of conditions (A) and (B) for the illustrative graph G shown in Figure 3, with H as shown. Recall $\mathrm{nd}^{\overline{G}} = \{1, 2\}$. Condition (B) depends on $R_{7,4}^{\overline{G},H} = \{4\}$, but it imposes no restriction when $w \in \{3, 4\}$ because $\mathrm{pa}_3^H = \mathrm{pa}_4^H = \emptyset$. We find that $Pa_W^{G,H}$ contains all tuples of parent sets for which $\mathrm{pa}_3^{G'} \subseteq \{1, 2\}$, $\mathrm{pa}_4^{G'} \subseteq \{1, 2\}$ and $\mathrm{pa}_7^{G'} = \{4\} \cap Z$ where $Z \subseteq \{1, 2\}$.

- The nodes $\mathrm{de}_{-w}^{\overline{G},H} = \bigcup_{x \in W \setminus \mathrm{pa}_w^H} \mathrm{de}_x^{\overline{G}}$ that are descendants in the reduced graph $\overline{G}$ of any node $x \in W \setminus \mathrm{pa}_w^H$, for a given node $w \in W$. Each node $x \in W \setminus \mathrm{pa}_w^H$ is not a parent node of w in the graph $H = \langle \mathrm{pa}_{w_1}^H, \ldots, \mathrm{pa}_{w_q}^H \rangle$.

Using this notation, we define $Pa_w^{G,H}$, for given $G \in \mathcal{G}$, $W \subseteq V$, $w \in W$ and $H = \langle \mathrm{pa}_{w_1}^H, \ldots, \mathrm{pa}_{w_q}^H \rangle \in \mathcal{H}$, as the set of parent sets $\mathrm{pa}_w^{G'}$ that satisfy the following conditions.

(A) $\mathrm{pa}_w^{G'} \subseteq Q_w^{\overline{G},H} = \left(\mathrm{nd}^{\overline{G}} \cup \mathrm{de}_w^{\overline{G},H} \right) \setminus \mathrm{de}_{-w}^{\overline{G},H}$

(B) $\mathrm{pa}_w^{G'} \cap R_{w,x}^{\overline{G},H} \neq \emptyset$ for all nodes $x \in \mathrm{pa}_w^H$, with $R_{w,x}^{\overline{G},H} = \mathrm{de}_x^{\overline{G}} \setminus \mathrm{de}_{-w}^{\overline{G},H}$.

Note that (B) depends on $\mathrm{de}_x^{\overline{G}}$ not $\mathrm{de}_x^{\overline{G},H}$. We define $Pa_W^{G,H}$ as the set of tuples formed by the Cartesian product of the sets $Pa_w^{G,H}$ of parent sets for $w \in W$; in other words $\mathrm{pa}_W^{G'} = (\mathrm{pa}_{w_1}^{G'}, \ldots, \mathrm{pa}_{w_q}^{G'}) \in Pa_W^{G,H}$ if and only if $\mathrm{pa}_{w_1}^{G'} \in Pa_{w_1}^{G,H}, \ldots, \mathrm{pa}_{w_q}^{G'} \in Pa_{w_q}^{G,H}$.

Lemma 1 *$\{Pa_W^{G,H_1}, \ldots, Pa_W^{G,H_\eta}\}$ is a partition of Pa_W^G.*

Proof See Appendix B. ■

Condition (A) ensures all graphs $G' = \langle \mathrm{pa}_W^{G'}, \mathrm{pa}_{-W}^G \rangle$, with parents $\mathrm{pa}_W^{G'} \in Pa_W^{G,H}$, are acyclic. It requires that all parents of $w \in W$ are either not descendants in the reduced graph $\overline{G}$ of any node in W, or a node whose ancestors in the reduced graph $\overline{G}$ are all parents in the graph H of node w. In particular, no descendant of w is added as an ancestor of w.

Condition (B) ensures each DAG G' is a member of $Pa_W^{G,H}$ for only a single $H \in \mathcal{H}$ for any given $G \in \mathcal{G}$ and thus that $\{Pa_W^{G,H_1}, \ldots, Pa_W^{G,H_\eta}\}$ is a partition of the set of permissible tuples of parent sets. The condition checks that each edge in the graph H is 'used' i.e. that $\mathrm{pa}_W^{G'} = (\mathrm{pa}_{w_1}^{G'}, \ldots, \mathrm{pa}_{w_q}^{G'}) \in Pa_W^{G,H}$ would not be allowed under any $H' \neq H$, $H' \in \mathcal{H}$. Specifically, it ensures that each parent set $\mathrm{pa}_w^{G'}$ contains at least one descendant node $v \in V$ of each parent pa_w^H of node w in the graph H, and that v is not a descendant in $\overline{G}$ of any node $x \in W$ that is not a parent in H of $w \in W$. To see why this condition is required, consider a graph G' in which all nodes in W have no parents. Then $\mathrm{pa}_w^{G'} = \emptyset$ for all nodes $w \in W$ and so condition (A) holds for all $H \in \mathcal{H}$. Thus if condition (B) is disregarded, $\mathrm{pa}_W^{G'} \in Pa_W^{G,H}$ for all $H \in \mathcal{H}$, implying that $\{Pa_W^{G,H_1}, \ldots, Pa_W^{G,H_\eta}\}$ is not a partition of Pa_W^G.

4.2 Fast Identification of Partition Components

Parent sets in each set $Pa_w^{G,H}$ can be identified easily using simple set operations, and consequently the partition components $Pa_W^{G,H}$ can be easily identified via Cartesian products of these sets. Let $Pa_w^{\mathrm{all}} = \mathcal{P}(V \setminus \{w\})$ denote the set of all possible parent sets of node w, subject to maximum in-degree κ, and $Pa_w^{\mathrm{all}}(v) = Pa_w^{\mathrm{all}} \setminus \mathcal{P}(V \setminus \{w, v\})$ denote the subset of Pa_w^{all} containing only parent sets that contain node $v \in V$. Parent sets in $Pa_w^{G,H}$ must (A) not include any nodes *not* in $Q_w^{\overline{G},H}$, and (B) include at least one node in $R_{w,x}^{\overline{G},H}$ for each $x \in \mathrm{pa}_w^H$, and thus

$$Pa_w^{G,H} = \begin{cases} Pa_w^{(B)} \setminus Pa_w^{(A)} & \text{if } \mathrm{pa}_w^H \neq \emptyset \\ Pa_w^{\mathrm{all}} \setminus Pa_w^{(A)} & \text{if } \mathrm{pa}_w^H = \emptyset \,, \end{cases}$$

where $Pa_w^{(A)} = \bigcup_{v \notin Q_w^{\overline{G},H}} Pa_w^{\mathrm{all}}(v)$ and $Pa_w^{(B)} = \bigcap_{x \in \mathrm{pa}_w^H} \bigcup_{r \in R_{w,x}^{\overline{G},H}} Pa_w^{\mathrm{all}}(r)$.

We fix $q = |W|$ as a small constant for all p and set a maximum in-degree κ. This means permissible tuples of parents sets can be identified in $\mathcal{O}(p^{\kappa+1})$ time by storing the 'lookup tables' $Pa_w^{\mathrm{all}}(v)$ as bit maps, assuming that $\mathcal{O}(1)$ querying of the descendants and non-descendants is available.

This can be achieved using a dynamic algorithm that provides access to the transitive closure. Giudici and Castelo (2003) previously used a similar approach. For a graph with transitive closure T^G with edges E^{T}, the descendants and non-descendants of node u are $\{v : (u, v) \in E^{\mathrm{T}}\}$ and $\{v : (u, v) \notin E^{\mathrm{T}}\}$ respectively. Thus, the descendants and non-descendants of a node can be identified in $\mathcal{O}(1)$ time when the transitive closure is known. The transitive closure for an arbitrary directed graph with p vertices can be determined in $\mathcal{O}(p^\omega)$ time (Munro, 1971), where ω is the best known exponent for matrix multiplication (Coppersmith and Winograd, 1990, show $\omega < 2.376$). However, in the present setting only incremental changes are made to the current state G of the sampler, so it is not necessary to use an offline algorithm at each iteration. Instead, we can use a fully dynamic transitive closure algorithm (Demetrescu et al., 2010) that provides procedures both for querying the

transitive closure, and for incrementally updating it when an edge is added or removed from the graph. We choose to implement the algorithm introduced by King and Sagert (2002), which performs queries in $\mathcal{O}(1)$ time, and updates in $\mathcal{O}(p^2)$ worst-case time (see reference for details of required assumptions). A trade-off exists between the performance of these two operations, but this bound for updates is thought to be the best possible whilst retaining $\mathcal{O}(1)$ queries (Demetrescu and Italiano, 2005) and the algorithm is simple to implement.

The algorithm maintains a $p \times p$ path count matrix C^G whose elements C^G_{uv} are the number of distinct paths from node u to node v in the graph G for $u \neq v$, and $C^G_{uv} = 1$ for $u = v$. The transitive closure can be derived from the path count matrix by noting that $\mathrm{T}^G_{uv} = 1$ if and only if $C^G_{uv} > 0$. Thus query operations are performed in $\mathcal{O}(1)$ by simply checking whether the relevant component of C^G is positive. When an edge is added or removed C^G is updated as follows. First consider adding an edge (u, v) to a graph G to form a new graph G'. The increase in the number of distinct paths between any two nodes x and y is given by the (x, y) element of $C^G_{*u} \otimes C^G_{v*}$, where C^G_{*u} denotes the u^{th} column of C^G, C^G_{v*} denotes the v^{th} row and $\otimes$ denotes the outer product. Thus, the path count matrix is updated simply as $C^{G'} = C^G + C^G_{*u} \otimes C^G_{v*}$. Similarly, when an edge (u, v) is removed from the graph the update is $C^{G'} = C^G - C^G_{*u} \otimes C^G_{v*}$.

4.3 Two-stage Sampling Method

We draw a new graph $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \rangle$ starting from a graph G using a two-stage method: we sample first a component $Pa^{G,H'}_W$ of the partition of permissible tuples of parent sets and then a tuple $\mathrm{pa}^{G'}_W$ of parent sets from the selected partition component.

In the first stage, $Pa^{G,H'}_W$ is drawn from the conditional distribution, given the tuple of parent sets of nodes not in W.

$$\begin{aligned} P(Pa^{G,H'}_W \mid \mathrm{pa}^G_{-W}, \mathbf{X}) &= \frac{P(Pa^{G,H'}_W, \mathrm{pa}^G_{-W} \mid \mathbf{X})}{P(\mathrm{pa}^G_{-W} \mid \mathbf{X})} \\ &= \frac{\sum_{\mathrm{pa}^{G'}_W \in Pa^{G,H'}_W} \prod_{w \in W} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G'}_w}) \pi_w(\mathrm{pa}^{G'}_w)}{\sum_{H'' \in \mathcal{H}} \sum_{\mathrm{pa}^{G''}_W \in Pa^{G,H''}_W} \prod_{w \in W} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G''}_w}) \pi_w(\mathrm{pa}^{G''}_w)} \\ &= \frac{\prod_{w \in W} \sum_{\mathrm{pa}^{G'}_w \in Pa^{G,H'}_w} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G'}_w}) \pi_w(\mathrm{pa}^{G'}_w)}{\sum_{H'' \in \mathcal{H}} \prod_{w \in W} \sum_{\mathrm{pa}^{G''}_w \in Pa^{G,H''}_w} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G''}_w}) \pi_w(\mathrm{pa}^{G''}_w)} \end{aligned}$$

The final equality follows by an interchange of sum and product that is proved in Lemma 2. This makes evaluation more efficient by allowing the sums to be evaluated separately for each node. Friedman and Koller (2003) used a similar interchange.

Lemma 2 *The following identity holds for any $H \in \mathcal{H}$, $W \subseteq V$ and $G \in \mathcal{G}$.*

$$\sum_{\mathrm{pa}_W \in Pa^{G,H}_W} \prod_{w \in W} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}_w}) \pi_w(\mathrm{pa}_w) = \prod_{w \in W} \sum_{\mathrm{pa}_w \in Pa^{G,H}_w} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}_w}) \pi_w(\mathrm{pa}_w)$$

Proof See Appendix C. ■

Algorithm 2 A Gibbs sampler for learning DAGs, with general blocks

Initialise starting point $G_0 = \langle \mathrm{pa}^{G_0}_1, \ldots, \mathrm{pa}^{G_0}_p \rangle$
Compute initial path count matrix C^{G_0}
for t in 1 to N **do**
  Sample q nodes uniformly at random from V, and call this set of nodes W
  Let $G = G_{t-1}$
  Form $\overline{G}$ as defined in Section 4.1
  Evaluate $C^{\overline{G}}$ from C^G as described in Section 4.2
  for $w \in W$ **do**
    Evaluate $\mathrm{de}^{\overline{G}}_w = \{v : C^{\overline{G}}_{wv} \geq 1\}$
    Evaluate $\mathrm{nd}^{\overline{G}}_w = \{v : C^{\overline{G}}_{wv} = 0\}$
  end for
  for $H \in \mathcal{H}$ **do**
    for $w \in W$ **do**
      Evaluate $Pa^{G,H}_w$ as described in Section 4.2
      Let $K^H_w = \sum_{\mathrm{pa}^{G'}_w \in Pa^{G,H}_w} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G'}_w}) \pi_w(\mathrm{pa}^{G'}_w)$
    end for
    Let $K^H = \prod_{w \in W} K^H_w$
  end for
  Sample $Pa^{G,H'}_W$ according to $P(Pa^{G,H'}_W \mid \mathrm{pa}^G_{-W}, \mathbf{X}) = \frac{K^{H'}}{\sum_{H'' \in \mathcal{H}} K^{H''}}$
  for $w \in W$ **do**
    Sample $\mathrm{pa}^{G'}_w$ according to $P(\mathrm{pa}^{G'}_w \mid Pa^{G,H'}_W, \mathrm{pa}^G_{-W}, \mathbf{X})$
  end for
  Set $G_t \leftarrow G = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^{G_{t-1}}_{-W} \rangle$
  Update C^{G_t}
end for

In the second stage, we sample new parents $\mathrm{pa}^{G'}_W$ from the selected partition component, and form the new graph $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \rangle$. The parents of each node $w \in W$, conditional on H', are independent, and so can be sampled separately from the following conditional distribution:

$$P(\mathrm{pa}^{G'}_w \mid Pa^{G,H'}_W, \mathrm{pa}^G_{-W}, \mathbf{X}) = \frac{p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G'}_w}) \pi_w(\mathrm{pa}^{G'}_w)}{\sum_{\mathrm{pa}^{G''}_w \in Pa^{G,H'}_w} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}^{G''}_w}) \pi_w(\mathrm{pa}^{G''}_w)}.$$

This step is straightforward because this distribution is simply the posterior distribution of a constrained variable selection with response w and $Pa^{G,H'}_w$ as the set of possible active sets (i.e. selected covariates).

Algorithm 2 outlines the complete algorithm. The methods in Sections 4.2 enable fast identification of each partition component. Run-time is a function of p, maximum in-degree κ, and the number q of nodes in the block. We choose a small, fixed q for all p, so the run-time is determined by the evaluation of $Pa^{G,H}_w$, which is $\mathcal{O}(p^{\kappa+1})$.

5. Results

In this section, we empirically assess the performance of the proposed sampler, comparing it with existing samplers as well as frequentist methods for structure learning of DAGs.

5.1 Evaluation Setup

We compare the Gibbs sampler with the MC^3 sampler (Madigan and York, 1995) and the REV sampler (Grzegorczyk and Husmeier, 2008), a variant of MC^3 that uses a more extensive edge reversal move. We also compare with two frequentist constraint-based methods: the PC-algorithm (Spirtes et al., 2000), and the Xie and Geng (2008) method, that is shown by its authors to outperform the PC-algorithm in some settings.

Tuning parameters for each method were set as follows. For the constraint-based methods, the significance level was $\alpha = 0.05$ by default, but we also show some results for $\alpha = 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1$. The Gibbs sampler we use is a random-scan sampler, with $q = 3$ (i.e. the parent sets of three nodes are sampled jointly at each iteration). To permit a fair comparison, for MC^3 we use the same fast online transitive closure algorithm (Section 4.2), and pre-computation and caching of local marginal likelihoods used in the Gibbs sampler (REV uses a similar pre-computation and caching scheme). We constrain all of the samplers to maximum in-degree $\kappa = 3$ and set the graph prior as $\pi(G) \propto 1$. We use conjugate formulations throughout, specifically multinomial-Dirichlet (Heckerman et al., 1995) for discrete data and Normal with a g-prior (with $g = n$) (Geiger and Heckerman, 1994; Zellner, 1986) for continuous data.

We consider six examples: a small domain example, where comparison with the exact posterior (Tian and He, 2009) is possible; data simulated from the ALARM network and from randomly-generated networks of varying sparsity; data sets from social science and biology; and a pathological 4-node example designed to highlight a failure case for our method.

In practical use samplers can only be run for a finite number of iterations (depending on available time and computational resources). We set the maximum number of iterations as follows. In total, we drew 10^6 iterations of REV (retaining only every 10^{th} iteration to reduce storage requirements). Following Grzegorczyk and Husmeier (2008), 85% of proposals within REV were MC^3 proposals (without MC^3 proposals, the REV sampler is not irreducible). In our implementation, the computational costs of the Gibbs sampler are an order of magnitude lower than REV's (accounting for MC^3 moves), but we nevertheless treated the computational costs as the same for the purposes of comparison and drew 10^6 iterations of the Gibbs sampler (again retaining every 10^{th} iteration). The computational costs of our implementation of MC^3 are roughly 1/10 of a Gibbs iteration, and so we performed 10^7 iterations of MC^3 (retaining every 100^{th}).

For each sampler, 10 independent runs starting from different initial graphs were performed. We discarded the first 1/4 of samples, but as an additional filter we considered trace plots (log marginal likelihood versus iteration) for each run of each sampler and discarded further samples if they appeared to be from a pre-convergent phase. Specifically, if there was a clear 'step change' in the trace for a certain run we discarded all samples in that run drawn prior to the highest 'step' being reached. We note that this simple filter cannot be regarded as detecting convergence because the highest 'step' may correspond to a region

near a local rather than a global mode. We therefore also study convergence in detail via other metrics.

5.2 Evaluation Metrics

Each sampler gives an estimated posterior distribution over DAGs. From this we obtain a point estimate either as the **maximum a posteriori** (MAP) graph G^{MAP}, or as a **thesholded graph** G_τ formed by including all directed edges whose marginal posterior edge probabilities are at least τ (note that G_τ need not be acyclic). When $\tau = 0.5$ we get the **median probability model** G^{med} (for normal linear models Barbieri and Berger, 2004, show that in some settings this is the optimal model for prediction). We also consider thresholding to match sparsity levels seen in frequentist point estimates, i.e. setting τ such that $|E(G_\tau)| = |E(\hat{G})|$ for a point estimate $\hat{G}$. By G_τ^{PC} and G_τ^{Xie} we denote thresholded graphs whose sparsity is matched to the PC and Xie-Geng estimates respectively.

We use the structural Hamming distance (SHD) to quantify differences between DAGs. This is the minimum number of edge insertions and deletions needed to transform one graph into another. Comparisons are made using completed partially directed acyclic graphs (Chickering, 2002a). To show the behaviour of the samplers at shorter MCMC run lengths some metrics are shown against iteration t. These are based on the final 3/4 of samples drawn up to t (except for log marginal likelihoods), and so may incorporate some samples drawn before convergence.

We assess convergence and stability via the following metrics:

- **Trace plots** of log marginal likelihood against iteration t.
- Between-run agreement between **posterior edge probabilities**. These are visualized as scatter plots. When edge probabilities agree between a pair of runs, all points in the plot will lie close to the $y = x$ line. We show two variants: a hexagonally-binned version, to avoid over-plotting (Carr et al., 1987); and a panelled plot, in which each panel shows one pair of runs. We also consider the number of edges with estimated posterior probability greater than 0.9 in one run, and less than 0.1 in another run. Such **'major discrepancies'** represent serious Monte Carlo artefacts.
- **Potential scale reduction factor (PSRF)** is a multiple-chain convergence diagnostic, developed by Gelman and Rubin (1992), that compares within- and between-chain means and variances of a suitable summary statistic. PSRF < 1.1 is often taken to indicate that a sampler has run sufficiently long. The natural summary statistics in this context are the posterior edge probabilities. However, we find that the resulting PSRF is not a reliable indicator of convergence. In simulations (not shown) using a 20 node network for which we could calculate true posterior edge probabilities (Tian and He, 2009), we found little association between single-edge PSRF and the (absolute) error in posterior edge probability. In particular, the diagnostic appeared to be very conservative for edges with true posterior probability near 0 or 1. We therefore suggest that a reasonable use of PSRF in this context may be to assess relative convergence between different samplers, rather than as an absolute indicator of convergence. Below, we compare samplers in terms of the proportion of edges with PSRF < 1.1 (a larger proportion suggests better mixing). To calculate PSRF we consider the 10 runs

of each sampler as a collection of 5 pairs of runs and calculate PSRF separately for each edge using the final three quarters of the samples drawn up to that point.

- For the real data, we assess **stability under resampling** ("shaking the data") by comparing estimates across bootstrap samples.

For experiments in which the true data-generating graph G^* is known we use the following metrics to assess accuracy:

- **Structural Hamming distance** (SHD) between G^* and the estimated graphs.
- **Receiver-operating characteristic (ROC) curves**. These show agreement between G^* and an estimate G_τ by plotting true positive against false positive rates parameterized by threshold τ. We consider also the **area under the ROC curve** (AUROC), focusing in particular on the small false positive rate region of the curve that is often of interest in applications.

Finally, when the posterior edge probabilities can be computed exactly (Tian and He, 2009) (i.e. in small p settings) we also consider the **maximum and average absolute error in posterior edge probability**, calculated across the set of all possible edges.

We note that while SHD and ROC scores are useful in assessing performance, they are not convergence measures, as they do not assess accuracy of the posterior distribution *per se* (to see why, consider a degenerate sampler that samples only G^*, giving perfect scores on SHD and ROC, but an incorrect posterior distribution).

5.3 Small Domain Comparison to Exact Posterior

We applied the methods to the Zoo data set (Newman et al., 1998) that records $p = 17$ (discrete) characteristics of $n = 101$ animals. The maximum log marginal likelihood found by the Gibbs and REV samplers was consistent across runs, but less so for the MC^3 sampler, and the Gibbs sampler reached a plateau of high probability after the fewest iterations (Figure A13a). REV required about ten times as many iterations as Gibbs to achieve the same proportion of edges with PSRF < 1.1 while MC^3 needed about 100 times as many (Figure A14a). The estimated edge probabilities given by the Gibbs sampler were stable between runs (Figure A15a). The results from the REV sampler were also stable, but MC^3 less so, with major disparities between some runs. Figure 4 shows the error in posterior edge probabilities as a function of iterations. Convergence was quickest for the Gibbs sampler, followed by REV and then MC^3. All runs of the Gibbs sampler reached a point at which the maximum absolute error was 0.05 (after 67,000 iterations on average). In contrast, only 5/10 runs of REV and 6/10 runs of MC^3 reached the same level of accuracy in their complete run. Similarly, the Gibbs sampler achieved an average absolute error of less than 0.01 in the fewest iterations.

5.4 The ALARM Network

The ALARM network (Beinlich et al., 1989) consists of 37 discrete nodes and 46 edges and has been widely used in studying structure learning (e.g. Friedman and Koller, 2003; Grzegorczyk and Husmeier, 2008). We simulated data sets from ALARM with sample sizes

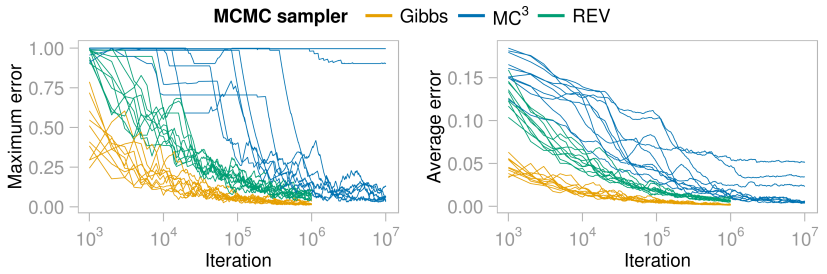


Figure 4: For the Zoo data set, maximum and average (across all possible 272 edges) error in posterior edge probabilities versus iteration number (on a $\log_{10}$ scale). For each MCMC algorithm, 10 independent runs, initialised at disparate initial values, are shown.

$n = 100$, 500, 1000, 2500, 5000. Figure 5 shows trace plots for the $n = 1000$ case as an illustrative example. The maximum log marginal likelihood found by the Gibbs sampler across runs was $-10{,}608.20 \pm 0.8$ (mean $\pm$ standard deviation), whereas for REV it was $-10{,}627.2 \pm 40.4$ and for MC^3 $-11{,}176.9 \pm 273.7$. The highest scoring graph discovered by any of the 10 runs of the MC^3 sampler had log marginal likelihood $-10{,}856.6$, far below the Gibbs maximum, suggesting non-convergence in all 10 runs. REV appeared to reach convergence in all but two runs (runs 9 and 10). The Gibbs sampler consistently (and rapidly) reached a high scoring plateau and appeared to have converged in all 10 runs. PSRF results were in line with these findings (Figure A14b); more than 10 times as many iterations of REV were needed to give the same proportion of edges with PSRF < 1.1 as the Gibbs sampler.

Figure 6 compares pairs of runs for $n = 1000$; this pair of runs is typical of all 10 runs (except for runs 9 and 10 of REV which disagree considerably; all runs shown in Figure A15b). There were no major between-run discrepancies (as defined in Section 5.2) for the Gibbs sampler at any sample size. The mean number of major discrepancies (across pairs of runs) increased from 0 ($n = 100$) to 8 and 91 for MC^3 and REV respectively (when $n = 5000$).

Figure 7 shows ROC curves for false positive rates < 0.05. The Gibbs sampler performs better and with less variability than the other methods. Sample size is influential. For small n the Bayesian methods outperform the constraint-based methods. However, counter to the increase in statistical information, REV and MC^3 perform less well with larger n: e.g. at $n = 100$ for a false positive rate (FPR) of 0, the Gibbs sampler found 21.9 ± 1.9 (mean $\pm$ standard deviation) true edges; REV 20.1 ± 1.4; and MC^3 21.7 ± 2.1. But at $n = 5000$, for the same FPR, Gibbs found 43.0 ± 0.0 true edges; REV 13.2 ± 21.3; and MC^3 did not find any true edges (the true graph has 46 edges). The constraint-based methods performed well for large sample sizes, as anticipated by the asymptotic consistency of the PC-algorithm (Kalisch and Bühlmann, 2007). The Xie-Geng method performed particularly well for $n = 5000$. Figure 8 shows AUROC as a function of iterations; the Gibbs sampler performed

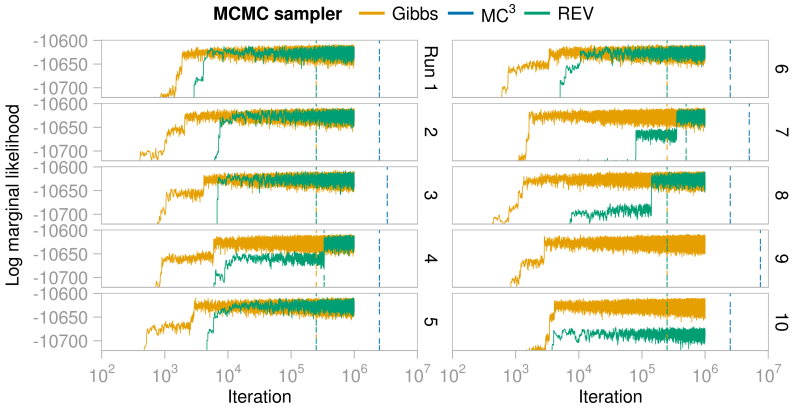


Figure 5: Log marginal likelihood of the graphs visited by the three MCMC samplers in 10 independent runs, initialised at disparate initial conditions, on the ALARM data, with data sample size $n = 1000$. Iteration number is displayed on a $\log_{10}$ scale. The dashed lines indicate where the burn-in phase ended. In all cases the log marginal likelihood of the graphs reached by the MC^3 sampler are below the displayed range.

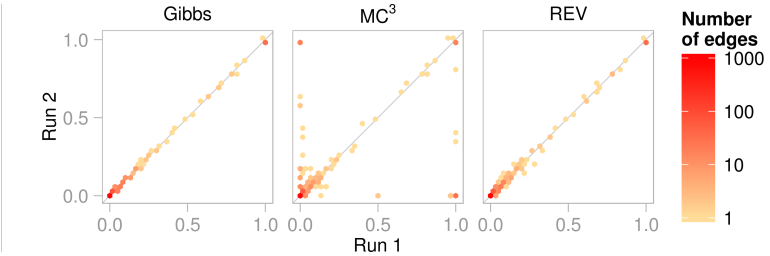


Figure 6: Convergence diagnostic plots for each MCMC sampler for the ALARM data, with $n = 1000$. The posterior edge probabilities given by two independent MCMC runs are plotted against each other, binned into hexagonal areas to avoid over-plotting. When the edge probabilities of the two runs agree, all of the points in the plot will lie close to the $y = x$ line; strongly off-diagonal points indicate extreme discrepancies between the runs.

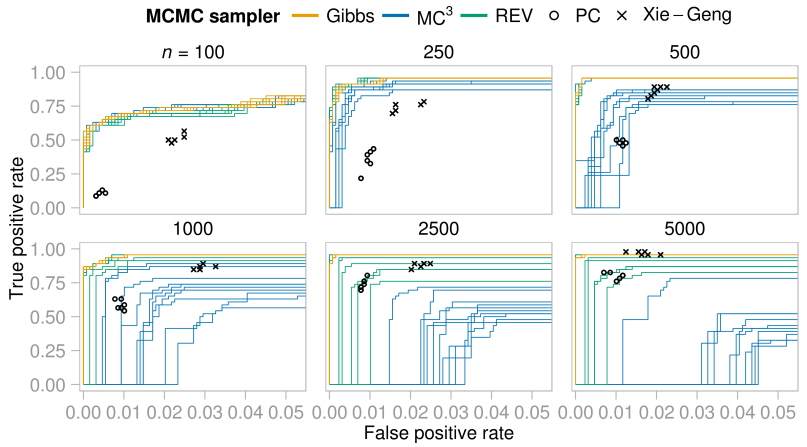


Figure 7: ALARM data, receiver operating characteristic (ROC) curves for each of 10 independent MCMC runs for each MCMC sampler, and point estimates for the constraint-based methods. Note that the horizontal axis shows only false positives rates < 0.05, corresponding to the case in which interest focuses on high-ranked edges (the complete curves are shown in Figures A16). Point estimates from Xie-Geng's constraint-based method and the PC-algorithm are also shown for all 8 significance levels considered.

best after all run lengths and sample sizes considered. These results are supported by SHDs between point estimates and the true graph (Table A2).

5.5 Networks of Varying Sparseness

Sparsity can be used to statistical and computational advantage but in practice it may be hard to know what level of sparsity is reasonable for a given application. We therefore sought to investigate the effect of varying network sparsity, including scenarios where the data-generating graph can violate the in-degree constraint we impose. We simulated data following a procedure described in Kalisch and Bühlmann (2007) that we outline below. We first generated a DAG G via its adjacency matrix A^G, by drawing entries as $A^G_{uv} \sim$ Bernoulli(ρ), where $\rho \in (0, 1)$ is a parameter controlling sparsity. Entries were drawn independently for each $u < v$, with $A^G_{uv} = 0$ otherwise. Larger ρ gives a denser graph and the expected number of neighbours (parents or immediate children) of a node is $\rho(p-1)$. We set $p = 25, n = 1000$ and considered 5 values of the sparseness parameter ρ (corresponding to expected neighbourhood sizes 2, 3, 4, 5 and 6). For each ρ we drew 10 DAGs, simulating a data set from each DAG by ancestral sampling using a Normal linear model (see Kalisch and Bühlmann, 2007, for full details).

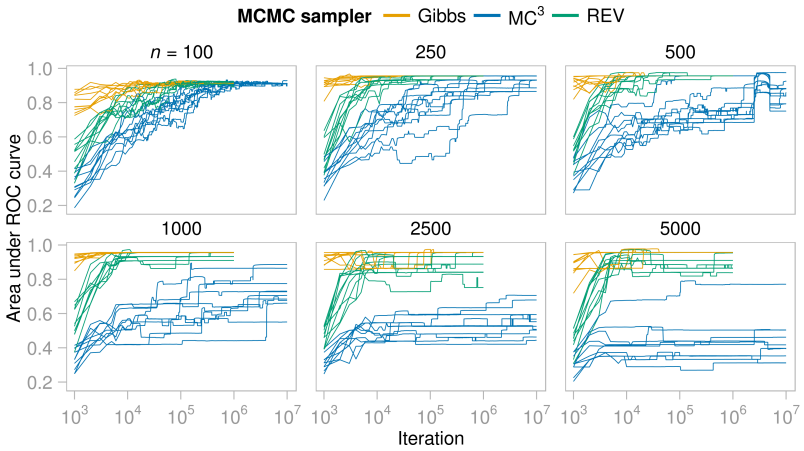


Figure 8: ALARM data, area under the receiver operating characteristic curve (AUROC) against iteration number ($\log_{10}$ scale) for each of the 10 independent runs for each MCMC algorithm. Each panel shows results for a particular sample size $n = 100, \ldots, 5000$.

Figure 9 shows boxplots over SHDs. We see that accuracy decreases with increasing density, echoing results in Kalisch and Bühlmann (2007) for the PC algorithm. As throughout, all the samplers had an in-degree constraint (maximum in-degree $\kappa = 3$), while the frequentist methods did not. Nevertheless, the performance of the Bayesian methods did not appear to deteriorate any more rapidly than the frequentist methods. At all ρ's, the median probability graph G^{med} outperformed the frequentist methods which in turn outperformed the MAP graph. In this context, we draw attention to a difference between the median probability and MAP graphs: the former, although obtained by averaging over DAGs satisfying the in-degree constraint, may itself have in-degree greater than κ, while the latter is necessarily subject to the constraint.

5.6 Survey Data

The publicly available Behavioral Risk Factor Surveillance System Survey (BRFSS) (Centers for Disease Control and Prevention, 2008) is a household-level random-digit telephone survey, collected by the U.S. National Center for Chronic Disease Prevention and Health, that has been conducted throughout the United States since 1984. We considered (discrete) responses to $p = 24$ questions (see Appendix D), spanning most of the topics covered in the survey. We considered the responses from New York in the 2008 survey, removing samples for respondents who refused or were unsure of their response, or whose response was

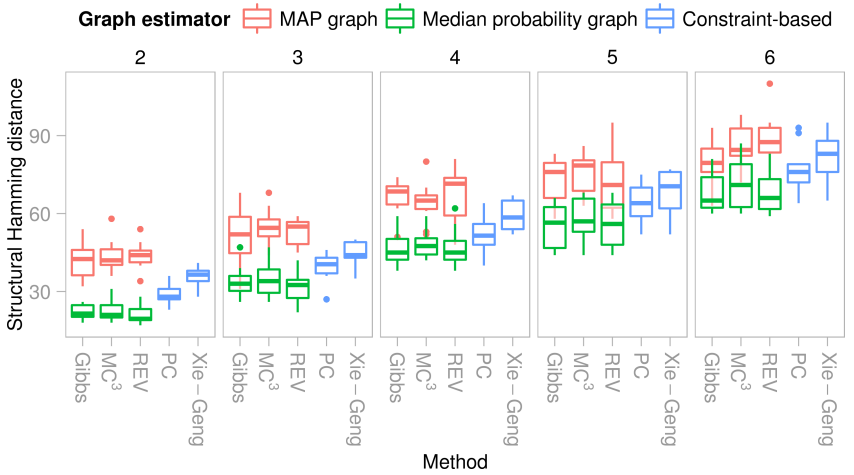


Figure 9: Synthetic data, accuracy of estimation for networks with different levels of sparsity. The panels correspond to simulations in which the expected number of neighbours for each node in the data-generating graph is respectively 2, 3, 4, 5, and 6. Accuracy is quantified by structural Hamming distance (SHD) between estimates and data generating graphs (smaller SHDs correspond to more accurate estimates). Box plots are over the 10 independent networks/data sets simulated for each sparsity level. For MCMC methods, results from the MAP graph G^{MAP} and median probability model G^{med} are shown.

missing, to any of the 24 questions. The resulting sample size was $n = 4,197$. The median probability graph G^{med} estimated by the Gibbs sampler is shown in Figure A17a.

Figure 10 shows between-run agreement. The Gibbs runs showed better agreement than the REV runs and the MC^3 runs disagreed considerably (these pairs of runs were typical; all runs shown in Figure A18a). Indeed there were no major between-run discrepancies (in the sense of Section 5.2) for the Gibbs sampler, whereas there were on average 2.4 major discrepancies for REV and 10.9 for MC^3. The Gibbs sampler also had the highest proportion of edges with PSRF < 1.1 (Figure A14c).

The maximum log marginal likelihoods found by each of the three samplers differed considerably as did the number of iterations needed to reach a plateau (Figure A13b). The Gibbs sampler typically reached a plateau after around 500 samples (although in one run 10^3 samples were needed). REV took longer to reach a (usually lower) plateau. MC^3 appeared to become stuck in a region with even lower log marginal likelihood.

To investigate stability under resampling of the data we applied the methods to 10 bootstrap resamples of the data set. The Gibbs and REV samplers were more stable than the frequentist methods as well as than MC^3. For example, using G^{PC}_{τ} as a point estimate

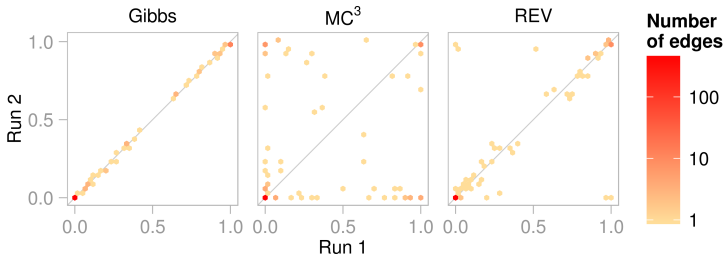


Figure 10: Convergence diagnostic plots for each MCMC sampler for the survey data. The posterior edge probabilities given by two independent MCMC runs are plotted against each other, binned into hexagonal areas to avoid over-plotting. When the two runs give the same estimates of the posterior edge probabilities all of the points appear on the line $y = x$; strongly off-diagonal points indicate extreme discrepancies between the runs. This pair of runs was typical of all pairs.

for the Bayesian methods (i.e. thresholding to give the same number of edges as PC), the mean SHD between results from pairs of bootstrap data sets was 21.7 (Gibbs), 22.3 (REV), 33.4 (MC^3) and 30.8 (PC-algorithm). Results for G^{MAP} and G^{Xie}_{τ} are shown in Figure A19a.

5.7 Large-sample, Single-cell Molecular Data

We used single-cell molecular data from Bendall et al. (2011) with $p = 34$ continuous variables (see Appendix D) and $n = 21,691$. The median probability graph G^{med} estimated by the Gibbs sampler is shown in Figure A17b.

Figure 11 shows between-run agreement for the samplers for a typical pair of runs; the Gibbs sampler shows better agreement than REV or MC^3. All but one of the 10 runs of the Gibbs sampler were consistent with each other (Figure A18b) and the one inconsistent run nonetheless showed better agreement with the other Gibbs runs than any pair of runs of the other samplers. The Gibbs sampler had no major discrepancies (as defined in Section 5.2) between any pairs of runs, while there were on average 26 and 87 major discrepancies for REV and MC^3 respectively. Smaller discrepancies followed a similar pattern: the average number of edges differing by 0.1 or more in posterior probability between runs were 6.4 (Gibbs sampler), 56.6 (REV), and 151.8 (MC^3). The Gibbs sampler also had the highest proportion of edges with PSRF < 1.1 after any run length (Figure A14d). Trace plots are shown in Figure A13c. The Gibbs sampler found a region of higher log marginal likelihood than the other samplers and did so consistently. Indeed, the maximum log marginal likelihood achieved across all REV runs was exceeded in every Gibbs run and after only around 5000 iterations. Finally, we considered stability under resampling of the data (Figure A19b), including also the frequentist point estimators for comparison. The mean SHD between PC estimates from pairs of bootstrap samples was 214.6; the corresponding mean SHDs for the samplers (using G^{PC}_{τ} point estimates) were 128.8 (Gibbs), 140.1 (REV) and 225.2 (MC^3). We note that the SHDs are particularly high here because the PC-algorithm estimates a

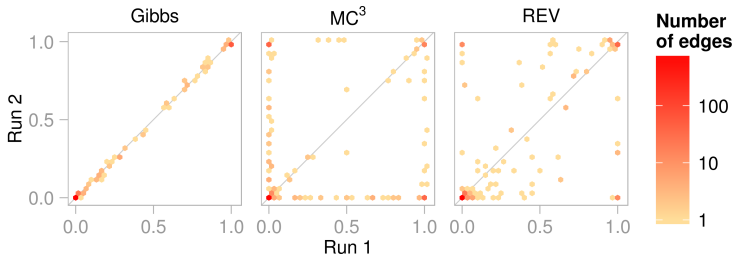


Figure 11: Convergence diagnostic plots for each MCMC sampler for the single-cell molecular data. The posterior edge probabilities given by two independent MCMC runs are plotted against each other, binned into hexagonal areas to avoid over-plotting. When the two runs give the same estimates of the posterior edge probabilities all of the points appear on the line $y = x$; strongly off-diagonal points indicate extreme discrepancies between the runs. This pair of runs was typical of all pairs of runs.

network with a high density (recall that under G^{PC}_{τ} all methods choose the same number of edges as PC).

5.8 A Pathological 4-node Example

Our final example demonstrates the potential sensitivity of the Gibbs sampler to choice of $q = |W|$ (the number of nodes whose parent sets are sampled together in a single iteration). The example was constructed to highlight a situation in which the sampler can become stuck in a local mode unless q is large enough, potentially leading to extremely slow convergence.

We used as data 10^5 simulated samples from a 4-node network in which the parents of node 2 were nodes 1 and 4; the parents of node 3 were nodes 1 and 2; and nodes 1 and 4 had no parents. Each node was Bernoulli distributed. The probability of success was 0.6 for nodes 1 and 4, while nodes 2 and 3 were noisy XORs with probability of success 0.9 when either parent (but not both) was 'true' and 0.1 otherwise.

For the purposes of demonstration, we set $q = 2$. As shown in Figure 12, after 10^6 iterations the maximum error in edge probability for the Gibbs sampler was 0.016 ± 0.015 (mean $\pm$ standard deviation across 10 runs). Given that there are only 543 DAGs with $p = 4$ nodes, this magnitude of error is unexpectedly large. The slow convergence is due to the concentration of posterior mass on two graphs that the sampler cannot easily move between. The MAP graph (posterior probability 0.61) is the graph in which the parents of node 2 are nodes 1 and 3; the parents of node 4 are nodes 1 and 2; and nodes 1 and 3 have no parents. The data-generating graph has posterior probability 0.38. The graphs differ in the parents of 3 nodes, and so with $q = 2$ the sampler cannot move between them in a single step. At the same time, the large sample size leads to all other graphs having low posterior probability (< 0.002), making multi-step transitions between the two graphs unlikely. Thus, the sampler becomes stuck on one of the two graphs. In this example,

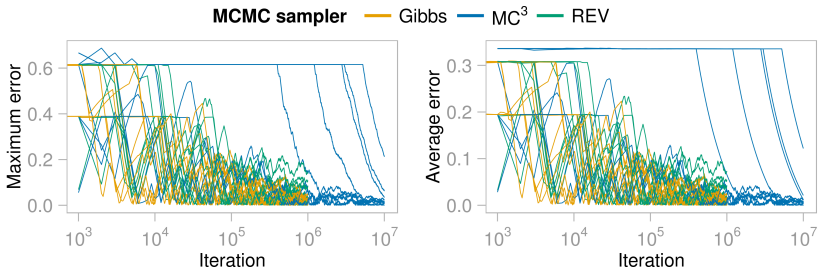


Figure 12: For the pathological 4-node example, maximum and average (across all possible 12 edges) error in posterior edge probabilities by iteration number (on a $\log_{10}$ scale). For each MCMC algorithm, 10 independent runs, initialised at disparate initial values, are shown.

setting $q = 3$ improved convergence (maximum error 0.0011 ± 0.0009 after 10^6 iterations), but in real-world examples it is difficult to rule out the possibility that analogous issues may arise.

6. Discussion

We introduced a Gibbs sampler for structure learning of DAGs that can converge more easily than existing samplers due to its ability to efficiently make large moves in DAG space. We showed that it provides often substantial gains in accuracy and stability in comparison with existing (Bayesian and frequentist) methods in a range of settings.

The formulation of the sampler develops and exploits the connection between variable selection and structure learning. This connection has been widely studied for undirected graphs (e.g. Meinshausen and Bühlmann, 2006), but for DAGs is complicated by the acyclicity requirement. Our approach accounts for acyclicity but further work in this area may ease the adaptation of results and methods from Bayesian variable selection to the case of DAGs. In the proposed sampler, existing variable selection methods could be of direct utility in sampling from the conditional distribution $P(\mathrm{pa}_W \mid \mathrm{pa}^G_{-W}, \mathbf{X})$. When this sampling step is difficult, a Metropolis-within-Gibbs approach (i.e. substituting a Metropolis step in place of the Gibbs step) could be considered. With $q = |W| = 1$, the conditional $P(\mathrm{pa}_W \mid \mathrm{pa}^G_{-W}, \mathbf{X})$ is identical to the posterior of the corresponding variable selection problem. Then, a Metropolis-within-Gibbs move could directly exploit existing variable selection methods.

In common with most of the structure learning literature, we used an in-degree constraint. This gives a smaller DAG space and in addition controls the number of parameters needed to specify conditional distributions. However, it is a strong constraint and a natural concern is whether it excludes higher in-degree models that could be appropriate for the data. This possibility cannot be ruled out, but the use of model averaging and marginal posterior summaries may ameliorate the concern to some extent, because even when the

true number of parents of a node exceeds the restriction, important candidate parents may still have high edge scores (and appear in a summary graph). For example, suppose the true in-degree of a node is 4, but at most 3 incoming edges are permitted. In this case, although no model including all 4 parents can be considered, provided the signal can be detected considering only 3 nodes at a time, the posterior probability of all 4 edges may nonetheless be high.

In our empirical examples the REV sampler of Grzegorczyk and Husmeier (2008) showed impressive gains over MC^3. The Gibbs sampler generally seemed to outperform both. The use of conditional distributions in REV is a point of similarity with the Gibbs sampler proposed here. The performance gains of Gibbs could be explained by two key differences from REV. First, REV does not use the natural conditional distribution and requires an accept-reject step. Second, REV must include at least some MC^3 proposals (otherwise the sampler is not irreducible), and these steps are not tailored to the shape of the posterior distribution (Grzegorczyk and Husmeier, 2008, make REV proposals with probability 1/15 and so most steps are based on MC^3 proposals).

If there is strong correlation between the parent sets of more than $q = |W|$ nodes, the Gibbs sampler may not mix well. In this situation, constraint-based methods may be useful. Alternatively, q could be chosen at each step according to some distribution, so that a mixture of different block sizes is used. This would in particular allow larger blocks to be used without increasing the computational demands of the algorithm excessively. In this paper we fixed $q = 3$, and found this simple choice gave a well-behaved and effective sampler. But there is a trade-off: increasing q increases the time taken to evaluate $P(\mathrm{pa}_W \mid \mathrm{pa}^G_{-W}, \mathbf{X})$, but also increases move size, with the potential to improve convergence.

Practical use of the Gibbs sampler in the form described here requires exact sampling from the conditional $P(\mathrm{pa}_W \mid \mathrm{pa}^G_{-W}, \mathbf{X})$ and there are situations related to this requirement in which other methods may be more suitable. First, when an appropriate maximum in-degree cannot be set, MC^3 or a variant could be more appropriate (although convergence could be very slow). Alternatively, search procedures such as GES (Chickering, 2002b) or HCMC (Castelo and Kočka, 2003) could be used. Second, exact sampling from the conditional distribution is challenging in settings with thousands of nodes. In this case, efficient constraint-based methods (such as the PC-algorithm) may be a better choice, particularly in the large sample setting. As noted by a referee, an interesting area for future research would be combining the Gibbs sampler with some aspects of other methods—such as PC—that are relatively well suited to the truly high-dimensional setting.

Acknowledgments

The authors are grateful to Karen Sachs for providing the single-cell molecular data; to Marco Grzegorczyk and Dirk Husmeier for their inspiring work in this area and for providing their implementation of the REV sampler; to the referees for numerous suggestions that improved the article; and to Frances Griffiths, David Lunn and Paul Newcombe for helpful discussions. Part of this work was carried out while the authors were at the University of Warwick—whose excellent scientific environment the authors warmly acknowledge—and

supported by the Economic and Social Research Council (ESRC) and Engineering and Physical Sciences Research Council (EPSRC).

Appendix A. Convergence Conditions for Gibbs Samplers

Convergence of a Gibbs sampler for DAGs does not follow from the usual justification of Gibbs sampling that relies on the Hammersley-Clifford theorem (Besag, 1974). The theorem gives a positivity condition that is sufficient to prove that the univariate conditional distributions used by the Gibbs sampler uniquely define the joint distribution. The required condition is that the support of the joint distribution is given by the Cartesian product of the supports of the marginal distributions. An example of when this condition does not hold is the joint density $p(x, y)$ for a pair of random variables X and Y with support only on $[0,1] \times [0,1]$ and $[2,3] \times [2,3]$. Clearly $p(x)$ and $p(y)$ are both positive on $[0,1]$ and $[2,3]$ but neither $[0,1] \times [2,3]$ or $[2,3] \times [0,1]$ are in the support of the joint distribution (Hobert et al., 1997).

In the DAG setting, the acyclicity requirement means that this positivity condition is not satisfied. Consider a DAG consisting of two correlated random variables $\mathbf{X}_1$ and $\mathbf{X}_2$. The correlation means that both the graph with a single edge $(1,2)$ and the graph with a single edge $(2,1)$ have positive posterior probability. Thus $P(A^G_{12} = 1 \mid \mathbf{X}) > 0$ and $P(A^G_{21} = 1 \mid \mathbf{X}) > 0$ in the posterior marginal distributions. However, in the joint posterior distribution $P(A^G_{12} = 1, A^G_{21} = 1 \mid \mathbf{X}) = 0$ because the corresponding graph (the complete graph) is cyclic. The complete graph is thus not in the support of the joint distribution but is in the Cartesian product of the supports of the marginal distributions.

An alternative sufficient condition for uniqueness of the joint distribution and convergence of the Gibbs sampler when positivity is not satisfied is given by Besag (1994), which was extended for continuous settings by Hobert et al. (1997). In the present context, the condition requires that for every pair $G', G'' \in \mathcal{G}$ of DAGs with p nodes there exists a finite sequence $G_1, \ldots, G_N$, with $G_1 = G'$, $G_N = G''$ and $N \in \mathbb{N}$, and such that G_t and G_{t-1} differ in only a single component (in this context, a single edge), and that the joint posterior distribution $P(G_t \mid \mathbf{X}) > 0$ for all $t = 0, \ldots N$. When the graph prior $\pi(G) > 0$ for all $G \in \mathcal{G}$, this condition is clearly satisfied: as an example, one such finite sequence removes every edge of G', one at a time, and then adds every edge of G'', one at a time. Each graph in the sequence is clearly acyclic, since the sequence is composed of subgraphs of the acyclic G' and G'', and so has positive probability in the joint distribution when the graph prior is positive everywhere in $\mathcal{G}$. A similar proof follows if the graph prior has support on all subgraphs of graphs with support in the graph prior, as is true for most widely used priors.

Appendix B. Proof of Lemma 1

Lemma 1 $\{Pa^{G,H_1}_W, \ldots Pa^{G,H_\eta}_W\}$ *is a partition of* Pa^G_W.

Proof. We show: (i) $\bigcup_{h=1,\ldots\eta} Pa^{G,H_h}_W \subseteq Pa^G_W$; (ii) $\bigcup_{h=1,\ldots\eta} Pa^{G,H_h}_W \supseteq Pa^G_W$; and (iii) $Pa^{G,H_{h_1}}_W \cap Pa^{G,H_{h_2}}_W = \emptyset$ for $H_{h_1} \neq H_{h_2}$, $H_{h_1}, H_{h_2} \in \mathcal{H}$.

(i) We prove the relationship by showing that $Pa^{G,H_h}_W \subseteq Pa^G_W$ for each $h = 1, \ldots, \eta$. By the definition of Pa^G_W in Section 3.4 we need to show that $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \rangle$ is acyclic for all tuples $\mathrm{pa}^{G'}_W \in Pa^{G,H_h}_W$ of parent sets, for all $h = 1, \ldots, \eta$.

We proceed by contradiction. Suppose some graph $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \rangle$ with $\mathrm{pa}^{G'}_W \in Pa^{G,H_h}_W$ is cyclic. First note that $\overline{G}$ is acyclic because $\overline{G}$ is a subgraph of the acyclic G. Since the graph G' differs from the acyclic graph $\overline{G}$ only in the parents of nodes in W, each cycle in G' must include at least one node in W. Denote by $x_1 \rightsquigarrow x_2$ the existence of a path (that obeys the edge directions) in the graph G' from node $x_1 \in W$ to node $x_2 \in W$ that does not include any nodes in W (except x_1 and x_2). Let $w_1, \ldots, w_r \in W$ be the (minimal) complete set of nodes in W included in some cycle in G'. Without loss of generality suppose that $w_1 \rightsquigarrow w_2 \rightsquigarrow \ldots \rightsquigarrow w_r$ in G'. Note that since $w_1, \ldots, w_r$ is the complete set of nodes in W in the cycle, no node between w_i and w_{i+1} in the path can be in W, $i \in \{1, \ldots, r-1\}$.

We now show that for $x_1, x_2 \in W$, $x_1 \rightsquigarrow x_2$ only if an edge (x_1, x_2) links node x_1 to x_2 in the graph H_h. Since $x_1 \rightsquigarrow x_2$, there must exist a node $v \in V$ that is a parent of x_2 in G' such that v is a descendant in G' of x_1. Note that if the edge (x_1, x_2) is in the graph G' then $v = x_1$. Since v is a parent of node x_2 in the graph G' and $x_2 \in W$, then $v \in \left(\mathrm{nd}^{\overline{G}} \cup \mathrm{de}^{\overline{G},H_h}_{x_2}\right) \setminus \mathrm{de}^{\overline{G},H_h}_{-x_2}$ by condition (A). Also since $x_1 \rightsquigarrow x_2$ does not include any nodes in W, v is also a descendant in $\overline{G}$ of x_1 because G' and $\overline{G}$ differ only in which nodes are parents of nodes in W. We proceed by contradiction. Suppose no edge (x_1, x_2) exists in H_h. Then v is a descendant of x_1 in the graph $\overline{G}$, but x_1 is not a parent of x_2 in the graph H_h. So $v \in \mathrm{de}^{\overline{G},H_h}_{-x_2}$, which by condition (A) is a contradiction. Thus $x_1 \rightsquigarrow x_2$ only if (x_1, x_2) is present in H_h, for $x_1, x_2 \in W$.

Now, recall that $w_1 \rightsquigarrow w_2 \rightsquigarrow \ldots \rightsquigarrow w_r$. Since a cycle is formed we must in addition have a path in G' from node w_r to node w_1. Since $w_1, \ldots, w_r$ is the complete set of nodes in W involved in the cycle, no node on the path from w_r to w_1 can be in W. Thus $w_r \rightsquigarrow w_1$. However, this implies that the edges $(w_1, w_2), (w_2, w_3), \ldots, (w_{r-1}, w_r)$, (w_r, w_1) are all in H_h, which implies H_h is cyclic. But H_h is acyclic by assumption, and so we have a contradiction.

(ii) Suppose we start from a graph $G = \langle \mathrm{pa}^G_W, \mathrm{pa}^G_{-W} \rangle$. We want to show that for each DAG $G' = \langle \mathrm{pa}^{G'}_W, \mathrm{pa}^G_{-W} \rangle$ there is some $H' \in \mathcal{H}$ such that $\mathrm{pa}^{G'}_W \in Pa^{G,H'}_W$. We will show that $\mathrm{pa}^{G'}_W \in Pa^{G,H'}_W$, where $H' = \langle \mathrm{pa}^{H'}_1, \ldots, \mathrm{pa}^{H'}_q \rangle \in \mathcal{H}$ is a DAG on nodes in W and $\mathrm{pa}^{H'}_w = \{x \in W : \text{there exists some } v \in \mathrm{pa}^{G'}_w \text{ such that } v \in \mathrm{de}^{\overline{G}}_x\}$. As usual, $\mathrm{pa}^{G'}_w$ is the parent set in the graph G' of the node w; and $\mathrm{de}^{\overline{G}}_x$ is the descendants in $\overline{G}$ of the node x. Note that H' is a subgraph (on the nodes in W) of the transitive closure $\mathrm{T}^{G'}$. By definition, G' is a DAG, so $\mathrm{T}^{G'}$ is also a DAG, and thus H' is a DAG.

We show that $\mathrm{pa}^{G'}_W \in Pa^{G,H'}_W$ by showing that for each node $w \in W$, both conditions (A) and (B) that specify membership of $Pa^{G,H'}_w$ are satisfied.

For (A) we need $\mathrm{pa}_w^{G'} \subseteq \left(\mathrm{nd}^{\overline{G}} \cup \mathrm{de}_w^{\overline{G},H'}\right) \setminus \mathrm{de}_{-w}^{\overline{G},H'}$. Let $v \in \mathrm{pa}_w^{G'}$ meaning that v is a parent of the node w in the graph G'. First we show that $v \notin \mathrm{de}_{-w}^{\overline{G},H'}$, and then show that $v \in \mathrm{nd}^{\overline{G}} \cup \mathrm{de}_w^{\overline{G},H'}$.

To show that $v \notin \mathrm{de}_{-w}^{\overline{G},H'}$, note that if $v \in \mathrm{de}_{-w}^{\overline{G},H'}$ then v must be a descendant in $\overline{G}$ of some node $y \in W$ that is not in $\mathrm{pa}_w^{H'}$. However, every such y is in $\mathrm{pa}_w^{H'}$ by the definition of $\mathrm{pa}_w^{H'}$, thus $v \notin \mathrm{de}_{-w}^{\overline{G},H'}$.

To show that $v \in \mathrm{nd}^{\overline{G}} \cup \mathrm{de}_w^{\overline{G},H'}$, we suppose $v \notin \mathrm{nd}^{\overline{G}}$ and show this implies that $v \in \mathrm{de}_w^{\overline{G},H'}$. This follows because if $v \notin \mathrm{nd}^{\overline{G}}$ then it must be the descendant in $\overline{G}$ of some node $y \in W$. Then $y \in \mathrm{pa}_w^{H'}$ by definition of H'. Therefore $v \in \mathrm{de}_w^{\overline{G},H'}$, as required.

For (B), we need $\mathrm{pa}_w^{G'} \cap \left(\mathrm{de}_x^{\overline{G}} \setminus \mathrm{de}_{-w}^{\overline{G},H'}\right) \neq \emptyset$ for all $x \in \mathrm{pa}_w^{H'}$. Consider $x \in \mathrm{pa}_w^{H'}$. By the definition of $\mathrm{pa}_w^{H'}$, this means that there exists some node $v \in \mathrm{pa}_w^{G'}$ such that $v \in \mathrm{de}_x^{\overline{G}}$. Additionally, any $y \in W$ for which $v \in \mathrm{de}_y^{\overline{G}}$ is such that $y \in \mathrm{pa}_w^{H'}$, by definition of $\mathrm{pa}_w^{H'}$. Thus v is not a descendant in $\overline{G}$ of any node in W that is not in $\mathrm{pa}_w^{H'}$. Then $v \in \mathrm{pa}_w^{G'} \cap \left(\mathrm{de}_x^{\overline{G}} \setminus \mathrm{de}_{-w}^{\overline{G},H'}\right)$, and thus the condition is satisfied.

(iii) Since $H_{h_1} \neq H_{h_2}$, at least one node has different parents in H_{h_1} and H_{h_2}. Suppose that the node $w \in W$ is such a node, and that $x \in W$ is a parent of w in H_{h_1} but not in H_{h_2}. Consider a parent set $\mathrm{pa}_w^{G_1} \in Pa_w^{G,H_{h_1}}$. We will prove that $\mathrm{pa}_w^{G_1} \notin Pa_w^{G,H_{h_2}}$, and the result follows.

By condition (B), there must exist some $v \in \mathrm{de}_x^{\overline{G}} \setminus \mathrm{de}_{-w}^{\overline{G},H_{h_1}}$ such that $v \in \mathrm{pa}_w^{G_1}$. We will show that $v \notin \mathrm{pa}_w^{G_2}$ for every parent set $\mathrm{pa}_w^{G_2} \in Pa_w^{G,H_{h_2}}$. This follows because $v \in \mathrm{de}_x^{\overline{G}}$ and so is a descendant in $\overline{G}$ of x, which is not a parent of w in H_{h_2}. Thus $v \in \mathrm{de}_{-w}^{H_{h_2}}$, and so $v \notin \left(\mathrm{nd}^{\overline{G}} \cup \mathrm{de}_w^{\overline{G},H_{h_2}}\right) \setminus \mathrm{de}_{-w}^{\overline{G},H_{h_2}}$. Therefore v cannot be a parent of w in G_2. ■

Appendix C. Proof of Lemma 2

Lemma 2 *The following identity holds for any $H \in \mathcal{H}$, $W \subseteq V$ and $G \in \mathcal{G}$.*

$$\sum_{\mathrm{pa}_W \in Pa_W^{G,H}} \prod_{w \in W} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}_w}) \pi_w(\mathrm{pa}_w) = \prod_{w \in W} \sum_{\mathrm{pa}_w \in Pa_w^{G,H}} p(\mathbf{X}_w \mid \mathbf{X}_{\mathrm{pa}_w}) \pi_w(\mathrm{pa}_w)$$

Proof. Define $\lambda_w^{(i)} = p(X_w \mid \mathbf{X}_{\mathrm{pa}_w^{(i)}}) \pi_w(\mathrm{pa}_w^{(i)})$, $i \in \{1, \dots \mathbb{P}\}$, where $\mathbb{P}$ is the cardinality of $Pa_W^{G,H}$, and where $\mathrm{pa}_w^{(i)}$ is the parent set of node w for the i^{th} member of $Pa_W^{G,H}$ i.e. we have that $Pa_W^{G,H} = \left\{(\mathrm{pa}_{w_1}^{(1)}, \dots, \mathrm{pa}_{w_q}^{(1)}), \dots, (\mathrm{pa}_{w_1}^{(\mathbb{P})}, \dots, \mathrm{pa}_{w_q}^{(\mathbb{P})})\right\}$. Similarly, define $Pa_w^{G,H} = \left\{\mathrm{pa}_w^{(1)}, \dots, \mathrm{pa}_w^{(\mathbb{P}_w)}\right\}$, and thus $\mathbb{P}_w$ is the number of tuples of parent sets in $Pa_w^{G,H}$. Recall that $Pa_W^{G,H}$ is the Cartesian product of the sets $Pa_w^{G,H}$ of parent sets for $w \in W$, thus

$$\begin{aligned}
\prod_{w \in W} \sum_{\mathrm{pa}_w^{(i)} \in Pa_w^{G,H}} \lambda_w^{(i)} &= \prod_{w \in W} \left(\lambda_w^{(1)} + \dots + \lambda_w^{(\mathbb{P}_w)}\right) \\
&= \sum_{i_1 \in \{1,\dots,\mathbb{P}_1\}, \dots, i_q \in \{1,\dots,\mathbb{P}_q\}} \lambda_{w_1}^{(i_1)} \dots \lambda_{w_q}^{(i_q)} \\
&= \sum_{(\mathrm{pa}_{w_1}^{(i)}, \dots, \mathrm{pa}_{w_q}^{(i)}) \in Pa_W^{G,H}} \lambda_{w_1}^{(i)} \dots \lambda_{w_q}^{(i)} \\
&= \sum_{(\mathrm{pa}_{w_1}^{(i)}, \dots, \mathrm{pa}_{w_q}^{(i)}) \in Pa_W^{G,H}} \prod_{w \in W} \lambda_w^{(i)} .
\end{aligned}$$

■

Appendix D. Details of Data Analysed

We included in our analysis the following variables from the survey data (Centers for Disease Control and Prevention, 2008): `SEX`, `_AGE_G`, `_RACEGR2`, `MARITAL`, `_CHLDCNT`, `_INCOMG`, `USEEQUIP`, `_HCVU65`, `MEDCOST`, `_SMOKER3`, `_ASTHMST`, `_RFDRHV3`, `_RFBING4`, `QLREST2`, `_RFSEAT3`, `_TOTINDA`, `_BMI4CAT`, `DIABETE2`, `EMTSUPRT`, `LSATISFY`, `_EXTETH2`, `_AIDTST2`, `_DENVST1`, `IMONTH`. In our analysis of the single-cell molecular data (Bendall et al., 2011) we included the following quantities, including the binding of antibodies, viability and DNA content: `191-DNA`, `193-DNA`, `103-Viability`, `115-CD45`, `139-CD45RA`, `141-pPLCgamma2`, `142-CD19`, `144-CD11b`, `145-CD4`, `146-CD8`, `148-CD34`, `150-pSTAT5`, `147-CD20`, `152-Ki67`, `154-pSHP2`, `151-pERK1/2`, `153-pMAPKAPK2`, `156-pMAPKAPK2AP70/Syk`, `158-CD33`, `160-CD123`, `159-pSTAT3`, `164-pSLP-76`, `165-pNFkB`, `166-IkBalpha`, `167-CD38`, `168-pH3`, `170-CD90`, `169-pP38`, `171-pBtk/Itk`, `172-pS6`, `174-pSrcFK`, `176-pCREB`, `175-pCrkL`, `110_114-CD3`.

Appendix E. Additional Table

Method	$n = 100$	250	500	1000	2500	5000
Gibbs	69.2 ± 6.5	32.0 ± 3.7	25.8 ± 2.8	22.1 ± 3.2	14.5 ± 3.3	8.7 ± 2.4
	23.8 ± 0.4	8.3 ± 0.5	8.9 ± 0.3	15.1 ± 0.6	6.0 ± 0.0	10.1 ± 0.3
MC^3	67.4 ± 4.9	35.8 ± 6.6	44.7 ± 7.8	47.8 ± 13.3	63.7 ± 11.2	78.5 ± 16.5
	23.9 ± 0.9	12.5 ± 3.1	27.3 ± 8.8	43.8 ± 14.2	65.4 ± 11.6	82.3 ± 17.0
REV	68.2 ± 5.5	34.7 ± 5.8	27.4 ± 4.6	28.1 ± 4.0	24.4 ± 9.5	20.4 ± 6.7
	25.9 ± 2.1	8.3 ± 1.1	4.0 ± 0.0	11.5 ± 2.0	11.6 ± 9.9	13.5 ± 8.0
PC	48.0	39.0	38.0	28.0	22.0	14.0
Xie-Geng	54.0	40.0	34.0	43.0	32.0	17.0

Table A2: ALARM data, structural Hamming distances (SHD) between estimated graphs and the true data-generating graph. For Bayesian methods we compare to both the *maximum a posteriori* (top line) and the median probability graph (bottom line), and report mean SHD over 10 independent Monte Carlo runs, along with the corresponding standard deviation. For constraint-based methods, we report results for $\alpha = 0.05$.

Appendix F. Additional Figures

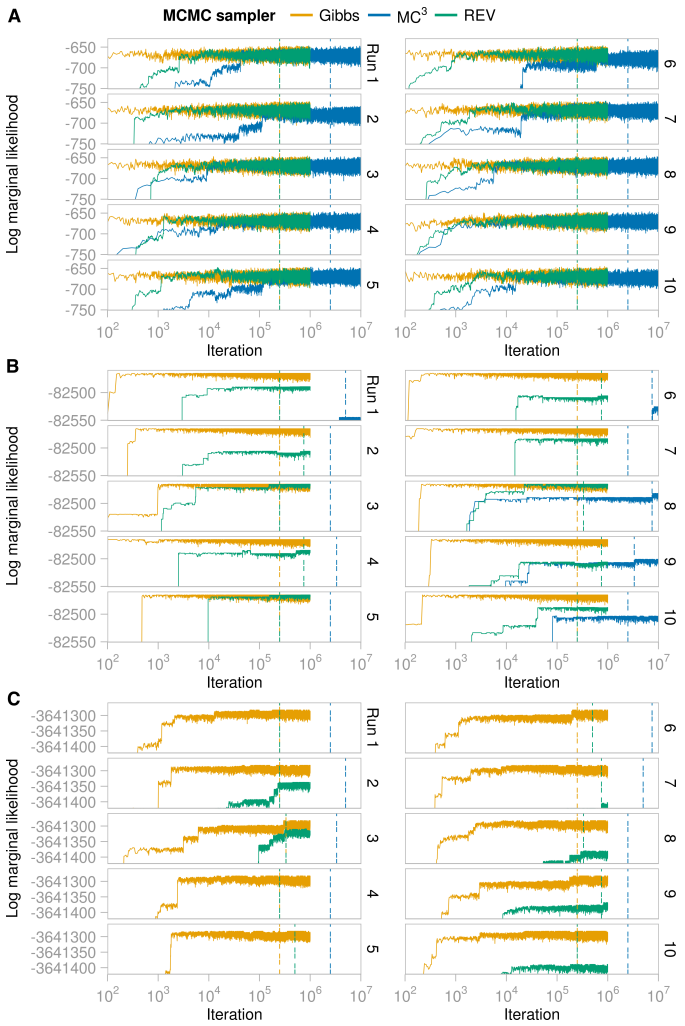


Figure A13: Log marginal likelihood of the graphs visited by the three MCMC samplers in 10 independent runs, initialised at disparate initial conditions. In (A) Zoo data; (B) survey data; (C) single-cell molecular data. Iteration number is displayed on a $\log_{10}$ scale. The dashed lines indicate where the burn-in phase ended.

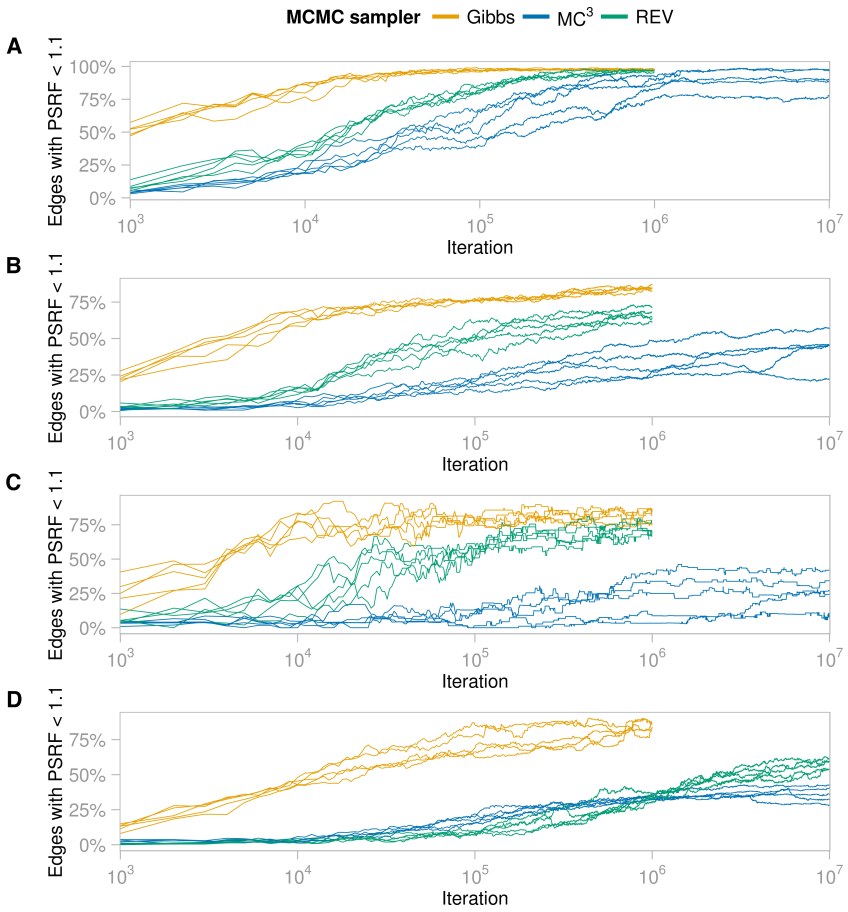


Figure A14: Proportion of edges with PSRF < 1.1 against iteration number. (A) Zoo data; (B) ALARM data, $n = 1000$; (C) survey data; (D) single-cell molecular data.

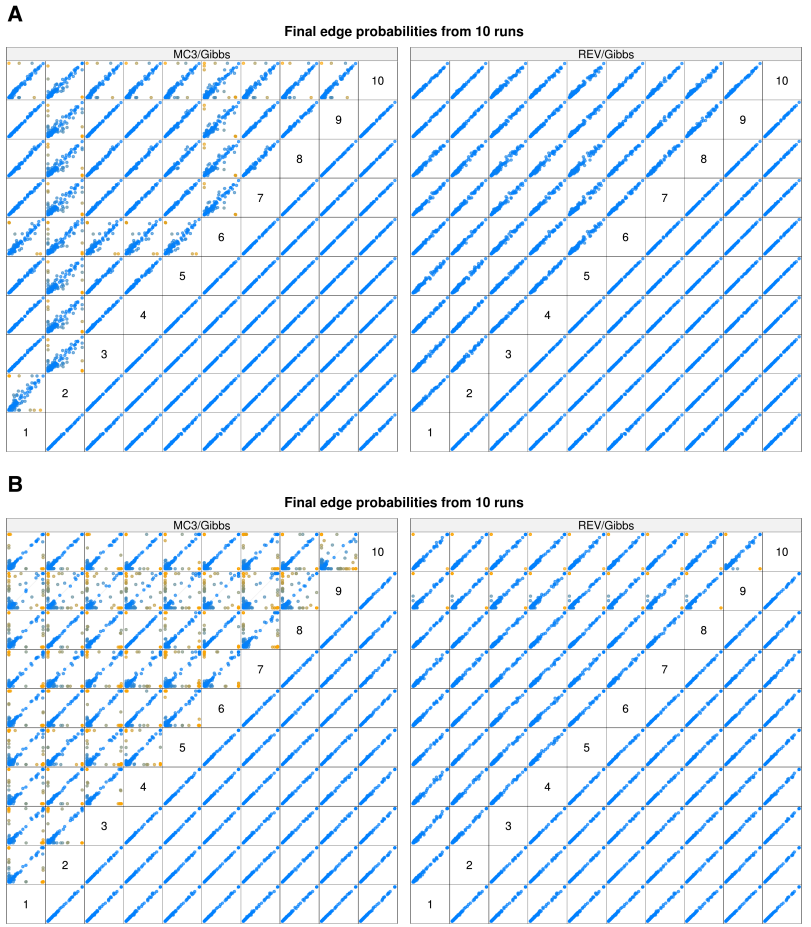


Figure A15: Convergence diagnostic plots for all pairs of 10 independent MCMC runs for each sampler for (A) Zoo data and (B) ALARM data, $n = 1000$. In each cell, the posterior edge probabilities given by two independent runs are plotted against each other. Each point represents a single edge. The lower half of both panels compares runs of the Gibbs sampler; the upper half compares runs of the MC^3 and the REV sampler respectively. When the two runs give the same estimates of the posterior edge probabilities, all of the points appear on the line $y = x$. The blue to orange colour scale represents the distance from this line, with orange points the furthest away.

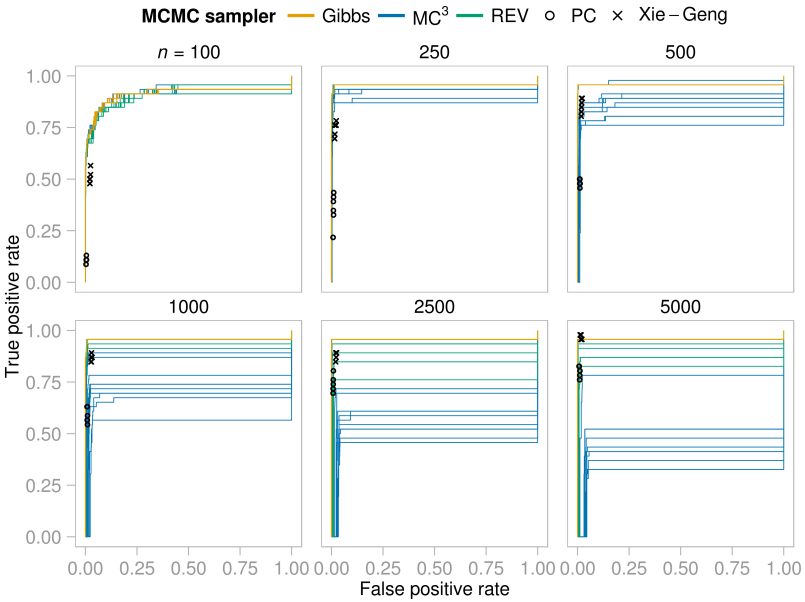


Figure A16: ALARM data, receiver operating characteristic (ROC) curves for each of 10 independent MCMC runs for each MCMC sampler, and point estimates for the constraint-based methods. Point estimates from Xie-Geng's constraint-based method and the PC-algorithm are also shown for all 8 significance levels.

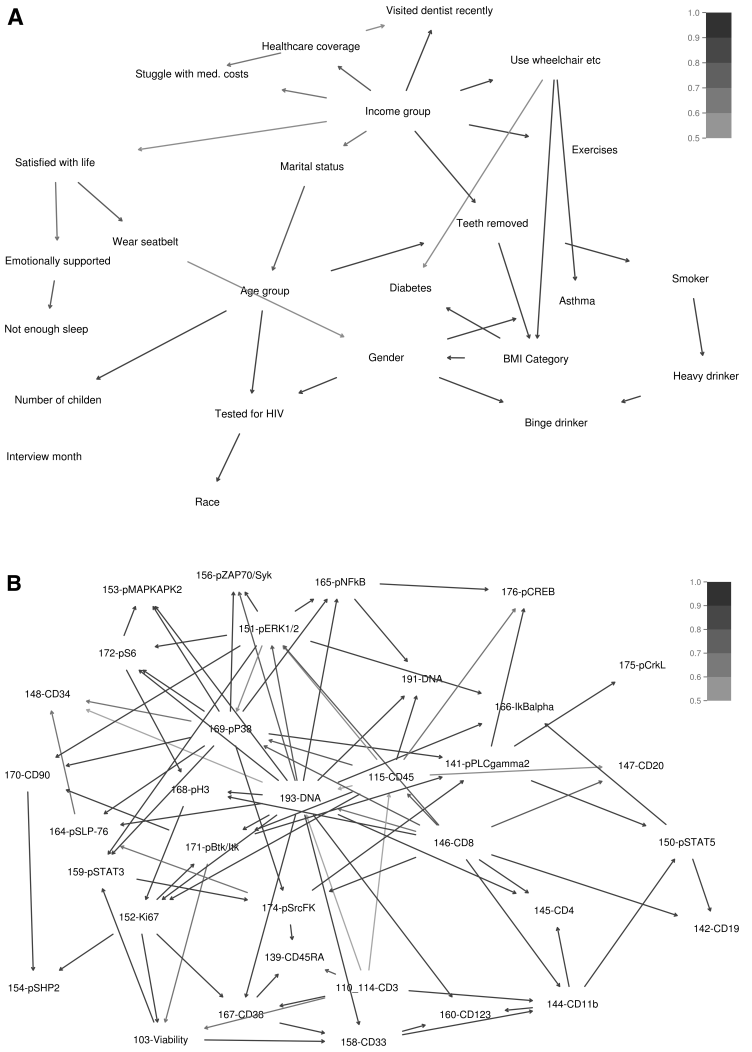


Figure A17: The median probability model G^{med} estimated by the Gibbs sampler, for (A) survey data and (B) single-cell molecular data. Darker shading indicates higher posterior edge probability. Note that no hard constraints were specified to ensure, for example, an in-degree of 0 for 'Age Group' in (A); such constraints were omitted to keep the implementations of the various methods simple.

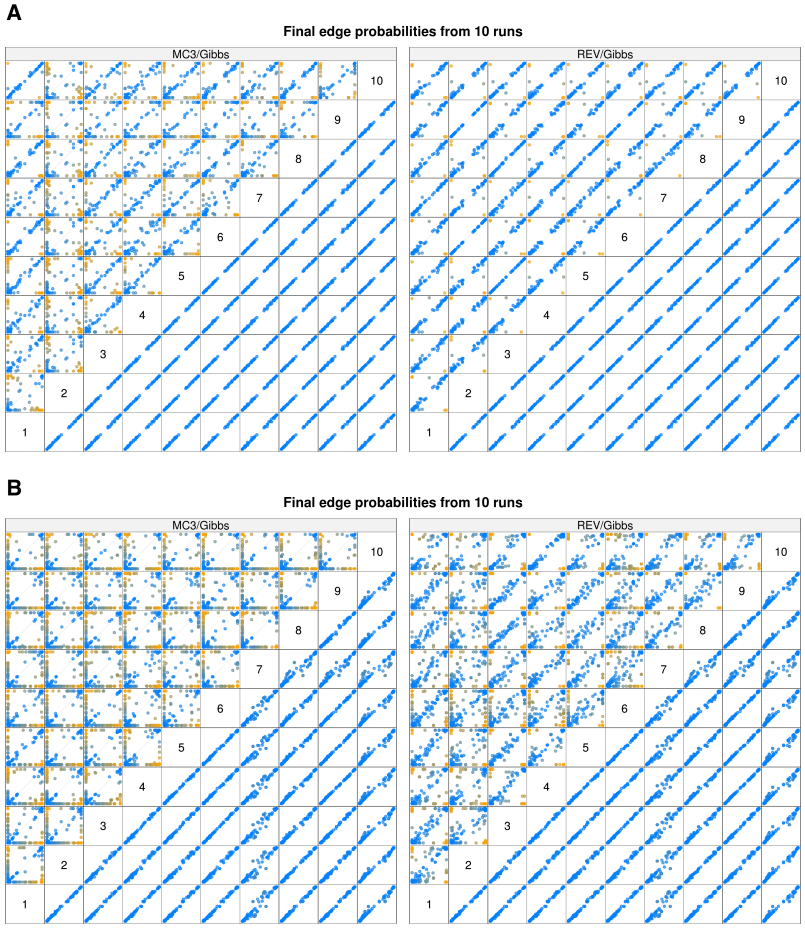


Figure A18: Convergence diagnostic plots for all pairs of 10 independent MCMC runs for each sampler for (A) survey data and (B) single-cell molecular data. In each cell, the posterior edge probabilities given by two independent runs are plotted against each other. Each point represents a single edge. The lower half of both panels compares runs of the Gibbs sampler; the upper half compares runs of the MC^3 and the REV sampler respectively. When the two runs give the same estimates of the posterior edge probabilities, all of the points appear on the line $y = x$. The blue to orange colour scale represents the distance from this line, with orange points the furthest away.

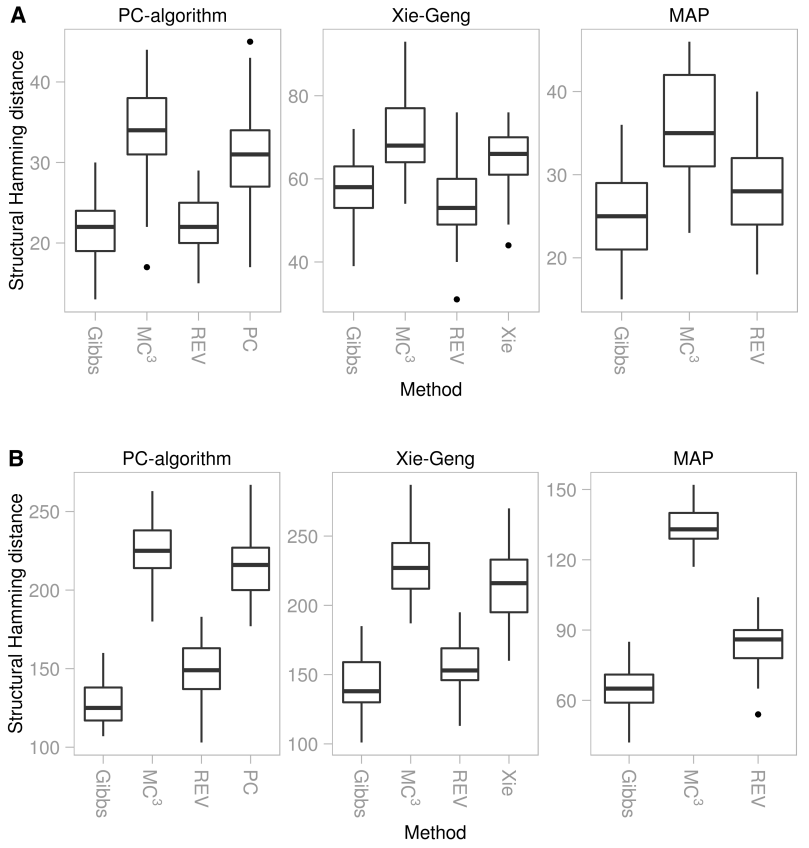


Figure A19: Stability of estimators under bootstrapping, as quantified by structural Hamming distance (SHD) between estimates obtained from pairs of bootstrap iterations for (A) survey data and (B) single-cell molecular data. In each case 10 bootstrap data sets were drawn and each estimator was run on each bootstrap data set. Smaller SHDs indicate stable estimation in the sense of graph estimates that are robust to resampling. Boxplots are shown over all pairs of bootstrap iterations. The edge probabilities were thresholded so that the resulting graphs had the same number of edges as the graph indicated in the panel title: G_τ^{PC} (the graph estimated by the PC-algorithm), G_τ^{Xie} (Xie-Geng method), or G^{MAP} (MAP graph).

References

M. M. Barbieri and J. O. Berger. Optimal predictive model selection. *Annals of Statistics*, 32(3):870–897, 2004.

I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Second European Conference on Artificial Intelligence in Medicine*, pages 247–256. Springer-Verlag, Berlin, 1989.

S. C. Bendall, E. F. Simonds, P. Qiu, E. D. Amir, P. O. Krutzik, R. Finck, R. V. Bruggner, R. Melamed, A. Trejo, O. I. Ornatsky, R. S. Balderas, S. K. Plevritis, K. Sachs, D. Pe'er, S. D. Tanner, and G. P. Nolan. Single-cell mass cytometry of differential immune and drug responses across a human hematopoietic continuum. *Science*, 332(6030):687–696, 2011.

J. E. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):192–236, 1974.

J. E. Besag. Discussion of "Markov chains for exploring posterior distributions". *Annals of Statistics*, 22(4):1734–1741, 1994.

D. B. Carr, R. J. Littlefield, W. L. Nicholson, and J. S. Littlefield. Scatterplot matrix techniques for large N. *Journal of the American Statistical Association*, 82(398):424–436, 1987.

R. Castelo and T. Kočka. On inclusion-driven learning of Bayesian networks. *Journal of Machine Learning Research*, 4:527–574, 2003.

Centers for Disease Control and Prevention. *Behavioral Risk Factor Surveillance System Survey Data*. U.S. Department of Health and Human Services, Atlanta, Georgia, 2008.

D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, 2002a.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002b.

D. Colombo and M. H. Maathuis. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 15:3921–3962, 2014.

D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.

C. Demetrescu and G. F. Italiano. Trade-offs for fully dynamic transitive closure on DAGs: breaking through the $O(n^2)$ barrier. *Journal of the ACM*, 52(2):147–156, 2005.

C. Demetrescu, D. Eppstein, Z. Galil, and G. F. Italiano. Dynamic graph algorithms. In M. J. Atallah and M. Blanton, editors, *Algorithms and Theory of Computation Handbook: General Concepts and Techniques*, pages 9.1–9.28. Chapman and Hall/CRC, Boca Raton, FL, 2010.

D. Eaton and K. Murphy. Bayesian structure learning using dynamic programming and MCMC. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*, pages 101–108, Corvallis, Oregon, 2007. AUAI Press.

B. Ellis and W. H. Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103(482):778–789, 2008.

N. Friedman and D. Koller. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1-2):95–125, 2003.

D. Geiger and D. Heckerman. Learning Gaussian networks. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 235–240, San Francisco, CA, 1994. Morgan Kaufmann.

A. Gelman and D. B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472, 1992.

P. Giudici and R. Castelo. Improving Markov chain Monte Carlo model search for data mining. *Machine Learning*, 50(1-2):127–158, 2003.

M. Grzegorczyk and D. Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2-3):265–305, 2008.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

J. P. Hobert, C. P. Robert, and C. Goutis. Connectedness conditions for the convergence of the Gibbs sampler. *Statistics & Probability Letters*, 33(3):235–240, 1997.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-Algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007.

V. King and G. Sagert. A fully dynamic algorithm for maintaining the transitive closure. *Journal of Computer and System Sciences*, 65(1):150–167, 2002.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA, 2009.

K. B. Korb and A. E. Nicholson. *Bayesian Artificial Intelligence*. Chapman & Hall/CRC Press, Boca Raton, FL, 2011.

D. Madigan and J. C. York. Bayesian graphical models for discrete data. *International Statistical Review / Revue Internationale de Statistique*, 63(2):215–232, 1995.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the Lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

I. Munro. Efficient determination of the transitive closure of a directed graph. *Information Processing Letters*, 1(2):56–58, 1971.

D. Newman, S. Hettich, C. Blake, and C. Merz. *UCI Repository of Machine Learning Databases*. University of California, Department of Information and Computer Science, Irvine, CA, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

O. Nikolova, J. Zola, and S. Aluru. Parallel globally optimal structure learning of Bayesian networks. *Journal of Parallel and Distributed Computing*, 73(8):1039–1048, 2013.

P. Parviainen and M. Koivisto. Exact structure discovery in Bayesian networks with less space. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 436–443, Corvallis, Oregon, 2009. AUAI Press.

P. Parviainen and M. Koivisto. Finding optimal Bayesian networks using precedence constraints. *Journal of Machine Learning Research*, 14:1387–1415, 2013.

G. O. Roberts and S. K. Sahu. Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler. *Journal of the Royal Statistical Society: Series B (Methodological)*, 59(2):291–317, 1997.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. The MIT Press, Cambridge, MA, 2000.

Y. Tamada, S. Imoto, and S. Miyano. Parallel algorithm for learning optimal Bayesian network structure. *Journal of Machine Learning Research*, 12:2437–2459, 2011.

J. Tian and R. He. Computing posterior probabilities of structural features in Bayesian networks. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 538–547, Corvallis, Oregon, 2009. AUAI Press.

X. Xie and Z. Geng. A recursive method for structural learning of directed acyclic graphs. *Journal of Machine Learning Research*, 9:459–483, 2008.

A. Zellner. On assessing prior distributions and Bayesian regression analysis with g-prior distributions. In P. K. Goel and A. Zellner, editors, *Bayesian Inference and Decision Techniques: Essays in Honour of Bruno de Finetti*, pages 233–243. North-Holland, Amsterdam, 1986.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 12/14; Revised 6/15; Published 4/16

Dimension-free Concentration Bounds on Hankel Matrices for Spectral Learning

François Denis francois.denis@lif.univ-mrs.fr
Mattias Gybels mattias.gybels@lif.univ-mrs.fr
Aix Marseille Université, CNRS
LIF UMR 7279
13288 Marseille Cedex 9, FRANCE

Amaury Habrard amaury.habrard@univ-st-etienne.fr
Université de Lyon, UJM-Saint-Etienne, CNRS, UMR 5516
Laboratoire Hubert Curien
F-42023 Saint-Etienne, FRANCE

Editor: Mehryar Mohri

Abstract

Learning probabilistic models over strings is an important issue for many applications. Spectral methods propose elegant solutions to the problem of inferring weighted automata from finite samples of variable-length strings drawn from an unknown target distribution p. These methods rely on a singular value decomposition of a matrix $\mathbf{H}_S$, called the empirical Hankel matrix, that records the frequencies of (some of) the observed strings S. The accuracy of the learned distribution depends both on the quantity of information embedded in $\mathbf{H}_S$ and on the distance between $\mathbf{H}_S$ and its mean $\mathbf{H}_p$. Existing concentration bounds seem to indicate that the concentration over $\mathbf{H}_p$ gets looser with its dimensions, suggesting that it might be necessary to bound the dimensions of $\mathbf{H}_S$ for learning. We prove new *dimension-free* concentration bounds for classical Hankel matrices and several variants, based on prefixes or factors of strings, that are useful for learning. Experiments demonstrate that these bounds are tight and that they significantly improve existing (dimension-dependent) bounds. One consequence of these results is that the spectral learning approach remains consistent even if all the observations are recorded within the empirical matrix.

Keywords: Hankel matrices, Matrix Bernstein bounds, Probabilistic Grammatical Inference, Rational series, Spectral learning

1. Introduction

Many applications in natural language processing, text analysis or computational biology require learning probabilistic models over finite variable-size strings such as probabilistic automata, Hidden Markov Models (HMM), or more generally, weighted automata. Weighted automata exactly model the class of rational series, and their algebraic properties have been widely studied in that context (Droste et al., 2009). In particular, they admit algebraic representations that can be characterized by a set of finite-dimensional linear operators whose ranks are closely linked to the minimum number of states needed to define the automaton. From a machine learning perspective, the objective is then to infer good estimates of these linear operators from finite samples. In this paper, we consider the problem of learning the linear representation of a weighted automaton, from a finite sample, composed of variable-size strings i.i.d. from an unknown target distribution.

Recently, the seminal papers of (Hsu et al., 2009) for learning HMM and (Bailly et al., 2009) for weighted automata, have defined a new category of approaches - the so-called *spectral methods* - for learning distributions over strings represented by finite state models (Siddiqi et al., 2010; Song et al., 2010; Balle et al., 2012; Balle and Mohri, 2012). Extensions to probabilistic models for tree-structured data (Bailly et al., 2010; Parikh et al., 2011; Cohen et al., 2012), transductions (Balle et al., 2011) or other graphical models (Anandkumar et al., 2012c,b,a; Luque et al., 2012) have also attracted a lot of interest.

Spectral methods suppose that the main parameters of a model can be expressed as the spectrum of a linear operator and estimated from the spectral decomposition of a matrix that sums up the observations. Given a rational series r, the values taken by r can be arranged in a matrix $\mathbf{H}_r$ whose rows and columns are indexed by strings, such that the linear operators defining r can be recovered directly from the right singular vectors of $\mathbf{H}_r$. This matrix is called the Hankel matrix of r.

In a learning context, given a learning sample S drawn from a target distribution p, an empirical estimate $\mathbf{H}_S$ of $\mathbf{H}_p$ is built and then, a rational series $\tilde{p}$ is inferred from the right singular vectors of $\mathbf{H}_S$. However, the size of $\mathbf{H}_S$ increases drastically with the size of S and state of the art approaches consider smaller matrices $\mathbf{H}_S^{U,V}$ indexed by limited subset of strings U and V. It can be shown that the above learning scheme, or slight variants of it, are consistent as soon as the matrix $\mathbf{H}_S^{U,V}$ has full rank (Hsu et al., 2009; Bailly, 2011; Balle et al., 2012) and that the accuracy of the inferred series is directly connected to the concentration distance $\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|_2$ between the empirical Hankel matrix and its mean (Hsu et al., 2009; Bailly, 2011).

On the one hand, limiting the size of the Hankel matrix avoids prohibitive calculations. Moreover, most existing concentration bounds on sum of random matrices depend on their size and suggest that $\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|_2$ may become significantly looser with the size of U and V, compromising the accuracy of the inferred model.

On the other hand, limiting the size of the Hankel matrix implies a drastic loss of information: only the strings of S compatible with U and V will be considered. In order to limit the loss of information when dealing with restricted sets U and V, a general trend is to work with other functions than the target p, such as the *prefix* function $\overline{p}(u) \coloneqq \sum_{v\in\Sigma^*} p(uv)$ or the *factor* function $\widehat{p} \coloneqq \sum_{v,w\in\Sigma^*} p(vuw)$ (Balle et al., 2013; Luque et al., 2012). These functions are rational, they have the same rank as p, a representation of p can easily be derived from representations of $\overline{p}$ or $\widehat{p}$ and they allow a better use of the information contained in the learning sample.

A first contribution of this paper is to provide a *dimension free* concentration inequality for $\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|_2$, by using recent results on tail inequalities for sum of random matrices (Tropp, 2012), and in particular a dimension-free Matrix Bernstein Bound Theorem stated in (Hsu et al., 2011). As a consequence, the spectral learning approach is consistent whatever sets U and V are chosen, and even if they are set to Σ^*, showing that restricting the dimensions of $\mathbf{H}$ is not mandatory.

However, this Matrix Bernstein Bound Theorem cannot be directly applied as such to the prefix and factor series, since the norm of the corresponding random matrices is unbounded. A second contribution of the paper is then to define two classes of parametrized functions,

$\overline{p}_\eta$ and $\widehat{p}_\eta$, that constitute continuous intermediates between p and $\overline{p}$ (resp. p and $\widehat{p}$), and to provide analogous dimension-free concentration bounds for these two classes. Lastly, we adapt a Matrix Bernstein bound theorem for subexponential matrices from (Tropp, 2012) to the dimension free case, using a technique similar as the one used in (Hsu et al., 2011) and we apply it to the prefix Hankel matrices.

These bounds are evaluated on a benchmark made of 11 problems extracted from the PAutomaC challenge (Verwer et al., 2012). These experiments show that the bounds derived from our theoretical results for bounded random matrices are quite tight - compared to the exact values - and that they significantly improve existing bounds, even on matrices of fixed dimensions. By contrast, the bounds obtained in the subexponential case are somewhat loose.

Our theoretical results entail that spectral learning is consistent whatever dimensions of the Hankel matrix are chosen but they give no indication on what should be done in practical cases. We have computed the distance between the spaces spanned by the first right singular vectors of $\mathbf{H}_S^{U,V}$ and $\mathbf{H}_p^{U,V}$ for various sizes of U and V, for each target of our benchmark. These experiments seem to indicate that the best results are obtained by limiting one dimension and taking the other as large as possible but a theoretical justification remains to be provided.

The paper is organized as follows. Section 2 introduces the main notations, definitions and concepts. Section 3 provides some Matrix Berstein bounds that will be used to prove the different results of the paper. Section 4.1 presents a first dimension free-concentration inequality for the standard Hankel matrices. Then, we introduce the prefix and the factor variants and provide analogous concentration results in Sections 4.3 and 4.5 respectively. Section 6 describes some experiments before the conclusion presented in Section 7. The Appendix contains the proof of an original result, which states that the series $u \mapsto p(\Sigma^* u \Sigma^*)$, i.e. the probability that a random string contains a substring u as a factor, may be not rational even if p is rational, explaining why we have considered the less natural series $\widehat{p}$. It also contains two small proofs of known results, in order to keep the paper self-contained.

2. Preliminaries

We first present some preliminary definitions and results about matrices, rational languages, Hankel matrices and spectral learning algorithms for the inference of rational stochastic languages.

2.1 Matrices

The identity matrix of size n is denoted by $\mathbf{I}_n$, or simply by $\mathbf{I}$. Let $\mathbf{M} \in \mathbb{R}^{m \times n}$ be a $m \times n$ real matrix. The *singular values* of $\mathbf{M}$ are the square roots of the eigenvalues of the matrix $\mathbf{M}^\top\mathbf{M}$, where $\mathbf{M}^\top$ denotes the transpose of $\mathbf{M}$: $\sigma_{\max}(\mathbf{M})$ and $\sigma_{\min}(\mathbf{M})$ denote the largest and smallest singular value of $\mathbf{M}$, respectively. The *spectral radius* $\rho(\mathbf{M})$ of a square matrix $\mathbf{M}$ is the supremum among the modulus of the eigenvalues of $\mathbf{M}$. If $\mathbf{M}$ is symmetric, $\sigma_{\max}(\mathbf{M})$ coincides with $\rho(\mathbf{M})$.

Every rank-d matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ admits a factorization of the form $\mathbf{M} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$, called a *reduced singular value decomposition* (SVD), where $\mathbf{U} \in \mathbb{R}^{m \times d}$ and $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_d$, $\mathbf{V} \in \mathbb{R}^{n \times d}$ and $\mathbf{V}^\top\mathbf{V} = \mathbf{I}_d$ and $\mathbf{D}$ is a diagonal matrix whose diagonal elements, listed in descending order,

are the singular values of $\mathbf{M}$. The columns of $\mathbf{U}$ (resp. of $\mathbf{V}$) are called the right-singular vectors of $\mathbf{M}$ (resp. left-singular vectors of $\mathbf{M}$).

The notion of singular values, singular vectors and singular value decomposition can be extended to infinite matrices via the notion of *Hilbert spaces compact operators* (see (Stein and Shakarchi, 2005) for example). Let $(e_i)_{i \in \mathbb{N}}$ be an orthonormal basis of a separable Hilbert space $\mathcal{H}$. A bounded operator $\mathbf{T}$ on $\mathcal{H}$ can be represented by the matrix $(\langle \mathbf{T}(e_i), e_j \rangle_{\mathcal{H}})_{i,j \in \mathbb{N}}$. Compact operators are the closure of finite-rank operators in the uniform operator topology: $\max_{\|x\|=1} \|\mathbf{T}_n(x) - \mathbf{T}(x)\| \to 0$. A sufficient condition for a matrix $\mathbf{M}$ to represent a compact operator is that it has a finite Frobenius norm: $\sum_{i,j \in \mathbb{N}} M[i,j]^2 < \infty$. The matrix of any compact operator admits a reduced SVD. In particular, if $\mathbf{M}$ is the matrix of a finite rank bounded operator, it admits a reduced singular value decomposition $\mathbf{M} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$.

The *operator norm* $\|\cdot\|_k$ induced by the corresponding vector norm on $\mathbb{R}^n$ is defined by $\|\mathbf{M}\|_k \coloneqq \max_{x \neq 0} \frac{\|\mathbf{M}x\|_k}{\|x\|_k}$. It can be shown that

$$\|\mathbf{M}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |\mathbf{M}[i,j]|, \|\mathbf{M}\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |\mathbf{M}[i,j]| \text{ and } \|\mathbf{M}\|_2 = \sigma_{\max}(\mathbf{M}).$$

We will mainly use the *spectral norm* $\|\cdot\|_2$ and we will omit the sub index 2 for the sake of simplicity. It can be shown that

$$\|\mathbf{M}\| \leq \sqrt{\|\mathbf{M}\|_1 \|\mathbf{M}\|_\infty} \tag{1}$$

These norms can be extended, under certain conditions, to infinite matrices. For example, the previous inequality remains true (with possibly infinite right-hand side term) if $\mathbf{M}$ represents the matrix of a compact operator in an orthonormal basis of a separable Hilbert space.

A symmetric matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ is *positive semidefinite* if $u^\top \mathbf{M} u \geq 0$ for all vectors $u \in \mathbb{R}^n$. Let $\preccurlyeq$ denotes the *positive semidefinite ordering* (or *Löwner ordering*) on symmetric matrices: $\mathbf{A} \preccurlyeq \mathbf{B}$ means that $\mathbf{B} - \mathbf{A}$ is positive semidefinite. The family of positive semidefinite matrices in $\mathbb{R}^{n \times n}$ forms a convex closed cone.

Any real valued function can be extended to symmetric matrices by the following method: let $\mathbf{A} = \mathbf{U}^\top \mathrm{diag}(\lambda_1, \ldots, \lambda_n)\mathbf{U}$ where $\mathrm{diag}(x_1, \ldots, x_n)$ is the diagonal matrix built over $x_1, \ldots, x_n$ and where $\mathbf{U} \in \mathbb{R}^{n \times n}$ is unitary, i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{U}\mathbf{U}^\top = \mathbf{I}$; define the matrix $f(\mathbf{A})$ by $f(\mathbf{A}) \coloneqq \mathbf{U}^\top \mathrm{diag}(f(\lambda_1), \ldots, f(\lambda_n))\mathbf{U}$. It can be shown that this definition is independent of the chosen eigenvalue decomposition. The *transfer rule* states that $f \leq g$ implies that $f(\mathbf{A}) \preccurlyeq g(\mathbf{A})$ for any symmetric matrix $\mathbf{A}$. The definition above can be used to define the exponential $e^{\mathbf{A}}$ of a symmetric matrix $\mathbf{A}$ and the logarithm $\log \mathbf{B}$ of a positive semidefinite matrix $\mathbf{B}$. It can be shown that the logarithm preserves the semidefinite order: $\mathbf{0} \preccurlyeq \mathbf{A} \preccurlyeq \mathbf{B}$ implies $\log \mathbf{A} \preccurlyeq \log \mathbf{B}$. See (Tropp, 2012) for a short overview of matrix properties.

2.2 Rational stochastic languages and Hankel matrices

Most classical results on rational series can be found in one of the following references (Berstel and Reutenauer, 1988; Salomaa and Soittola, 1978). Let Σ be a finite alphabet. The set of all finite strings over Σ is denoted by Σ^*, the empty string is denoted by ϵ, the length of

string w is denoted by $|w|$ and Σ^n (resp. $\Sigma^{\leq n}$, resp. $\Sigma^{\geq n}$) denotes the set of all strings of length n (resp. $\leq n$, resp. $\geq n$). For any string w, let $\text{Pref}(w) \coloneqq \{u \in \Sigma^* | \exists v \in \Sigma^* \quad w = uv\}$ and $\text{Suff}(w) \coloneqq \{v \in \Sigma^* | \exists u \in \Sigma^* \quad w = uv\}$.

A *series* is a mapping $r : \Sigma^* \to \mathbb{R}$. The *support* of the series r is the set $\text{supp}(r) = \{u \in \Sigma^* : r(u) \neq 0\}$. A series r is *non negative* if it takes only non negative values. A non negative series r is *convergent* if the sum $\sum_{u\in\Sigma^*} r(u)$ is bounded: for any $A \subseteq \Sigma^*$, let us denote $r(A) \coloneqq \sum_{u\in A} r(u)$. A *stochastic language* p is a probability distribution over Σ^*, i.e. a non negative series p satisfying $p(\Sigma^*) = 1$.

Let $n \geq 1$ and $\mathbf{M}$ be a morphism defined from Σ^* to $\mathbb{M}_n$, the set of square $n \times n$ matrices with real coefficients. For all $u \in \Sigma^*$, let us denote $\mathbf{M}(u)$ by $\mathbf{M}_u$ and $\Sigma_{x\in\Sigma}\mathbf{M}_x$ by $\mathbf{M}_\Sigma$. A series r over Σ is *rational* if there exists an integer $n \geq 1$, two vectors $I, T \in \mathbb{R}^n$ and a morphism $\mathbf{M} : \Sigma^* \to \mathbb{M}_n$ such that for all $u \in \Sigma^*$, $r(u) = I^\top \mathbf{M}_u T$. The triplet $\langle I, \mathbf{M}, T\rangle$ is called an n-dimensional *linear representation* of r. The vector I can be interpreted as a vector of initial weights, T as a vector of terminal weights and the morphism $\mathbf{M}$ as a set of matrix parameters associated with the letters of Σ. A *rational stochastic language* is thus a stochastic language admitting a linear representation.

Let $U, V \subseteq \Sigma^*$, the *Hankel matrix* $\mathbf{H}_r^{U,V}$, associated with a series r, is the matrix indexed by $U \times V$ and defined by $\mathbf{H}_r^{U,V}[u, v] \coloneqq r(uv)$, for any $(u, v) \in U \times V$. If $U = V = \Sigma^*$, $\mathbf{H}_r^{U,V}$, simply denoted by $\mathbf{H}_r$, is a bi-infinite matrix. In the following, we always assume that $\epsilon \in U \cap V$ and that U and V are ordered in quasi-lexicographic order: strings are first ordered by increasing length and then, according to the lexicographic order. It can be shown that a series r is rational if and only if the rank of the matrix $\mathbf{H}_r$ is finite. The rank of $\mathbf{H}_r$ is equal to the minimal dimension of a linear representation of r: it is called the rank of r. The Hankel matrix $\mathbf{H}_r$ represents a bounded operator if and only if $\sum_{u\in\Sigma^*} r^2(u) < \infty$; in particular, if r is a non negative convergent rational series, then $\mathbf{H}_r$ represents a compact operator, which admits a reduced singular value decomposition.

Let r be a non negative convergent rational series and let $\langle I, \mathbf{M}, T\rangle$ be a minimal d-dimensional linear representation of r. Then, the matrix $\mathbf{I}_d - \mathbf{M}_\Sigma$ is invertible and the sum $\mathbf{I}_d + \mathbf{M}_\Sigma + \ldots + \mathbf{M}_\Sigma^n + \ldots$ converges to $(\mathbf{I}_d - \mathbf{M}_\Sigma)^{-1}$. For any ρ_r such that $\rho(\mathbf{M}_\Sigma) < \rho_r < 1$, there exists a constant $C_r > 0$ such that $r(\Sigma^{\geq n}) \leq C_r \rho_r^n$ for any integer n (we show in Section 6.1 how such constants can be computed in pratical cases). For any integer $k \geq 1$, let us define the moments $S_r^{(k)} \coloneqq \sum_{u_1 u_2 \ldots u_k \in \Sigma^*} r(u_1 u_2 \ldots u_k)$. It can easily be shown that

$$S_r^{(k)} = I^\top (\mathbf{I}_d - \mathbf{M}_\Sigma)^{-k} T. \tag{2}$$

Several rational series can be naturally associated with a rational non negative convergent series r (see (Balle et al., 2014) for example):

- $\overline{r}$, defined by $\overline{r}(u) \coloneqq \sum_{v\in\Sigma^*} r(uv) = r(u\Sigma^*)$, associated with the *prefixes* of the support of r,
- $\widehat{r}$, defined by $\hat{r}(u) \coloneqq \sum_{v,w\in\Sigma^*} r(vuw)$, associated with the *factors* of the support of r.

If p is a stochastic language, it can be noticed that $\overline{p}(u)$ is the probability that a string begins with u and that $\widehat{p}(u) = \mathbb{E}_{v\sim p}|v|_u$, where $|v|_u = \sum_{x,y\in\Sigma^*} \mathbf{1}_{xuy=v}$. We have $\widehat{p}(u) \geq p(\Sigma^* u \Sigma^*)$, the probability that a string contains u as a substring. The function $u \mapsto p(\Sigma^* u \Sigma^*)$ has a

simpler probabilistic interpretation than $\widehat{p}$. However, this function is not rational in general and cannot easily be used in a learning context.

Proposition 1 *There exists a rational stochastic language p of rank one and built on a two-letter alphabet Σ such that the series $u \mapsto p(\Sigma^* u \Sigma^*)$ is not rational.*

Proof See Appendix. ■

If $\langle I, \mathbf{M}, T\rangle$ is a minimal d-dimensional linear representation of r, then $\langle I, \mathbf{M}, (\mathbf{I}_d - \mathbf{M}_\Sigma)^{-1}T\rangle$ (resp. $\langle [I^\top(\mathbf{I}_d - \mathbf{M}_\Sigma)^{-1}]^\top, \mathbf{M}, (I_d - \mathbf{M}_\Sigma)^{-1}T\rangle$) is a minimal linear representation of $\overline{r}$ (resp. of $\widehat{r}$). Conversely, a linear representation of r can be deduced from any linear representation of $\overline{r}$ or of $\widehat{r}$. Clearly,

$$r(\Sigma^*) = S_r^{(1)}, \overline{r}(\Sigma^*) = S_r^{(2)} \text{ and } \widehat{r}(\Sigma^*) = S_r^{(3)}.$$

Let $U, V \subseteq \Sigma^*$. For any string $w \in \Sigma^*$, let us define the matrices $\mathbf{H}_w^{U,V}$, $\overline{\mathbf{H}}_w^{U,V}$ and $\widehat{\mathbf{H}}_w^{U,V}$ by

$$\mathbf{H}_w^{U,V}[u, v] \coloneqq \mathbf{1}_{uv=w}, \overline{\mathbf{H}}_w^{U,V}[u, v] \coloneqq \mathbf{1}_{uv\in\text{Pref}(w)} \text{ and } \widehat{\mathbf{H}}_w^{U,V}[u, v] \coloneqq \sum_{x,y\in\Sigma^*} \mathbf{1}_{xuvy=w}$$

for any $(u, v) \in U \times V$.

For any non empty multiset of strings S, let us define the matrices $\mathbf{H}_S^{U,V}$, $\overline{\mathbf{H}}_S^{U,V}$ and $\widehat{\mathbf{H}}_S^{U,V}$ by

$$\mathbf{H}_S^{U,V} \coloneqq \frac{1}{|S|}\sum_{w\in S} \mathbf{H}_w^{U,V}, \overline{\mathbf{H}}_S^{U,V} \coloneqq \frac{1}{|S|}\sum_{w\in S} \overline{\mathbf{H}}_w^{U,V} \text{ and } \widehat{\mathbf{H}}_S^{U,V} \coloneqq \frac{1}{|S|}\sum_{w\in S} \widehat{\mathbf{H}}_w^{U,V}.$$

Let p_S be the empirical stochastic language associated with S, defined by $p_S(u) \coloneqq \frac{|\{u\in S\}|}{|S|}$. We have

$$\mathbf{H}_{p_S}^{U,V} = \mathbf{H}_S^{U,V}, \mathbf{H}_{\overline{p_S}}^{U,V} = \overline{\mathbf{H}}_S^{U,V} \text{ and } \mathbf{H}_{\widehat{p_S}}^{U,V} = \widehat{\mathbf{H}}_S^{U,V}.$$

For example, let $S = \{a, ab\}$, $U = V = \{\epsilon, a, b\}$. We have

$$\mathbf{H}_S^{U,V} = \begin{pmatrix} 0 & 1/2 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 0 & 0 \end{pmatrix}, \quad \overline{\mathbf{H}}_S^{U,V} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1/2 \\ 0 & 0 & 0 \end{pmatrix} \text{ and } \widehat{\mathbf{H}}_S^{U,V} = \begin{pmatrix} 5/2 & 1 & 1/2 \\ 1 & 0 & 1/2 \\ 1/2 & 0 & 0 \end{pmatrix}.$$

2.3 Spectral Algorithm for Learning Rational Stochastic Languages

Rational series admit *canonical linear representations* determined by their Hankel matrix. Let r be a rational series of rank d and $U \subset \Sigma^*$ such that the matrix $\mathbf{H}_r^{U\times\Sigma^*}$ (denoted by $\mathbf{H}$ in the following) has rank d. Moreover, suppose that $\sum_{u\in\Sigma^*} r(u)^2 < \infty$.

- For any string s, let $\mathbf{T}_s$ be the constant matrix whose rows and columns are indexed by Σ^* and defined by $\mathbf{T}_s[u, v] \coloneqq 1$ if $v = us$ and 0 otherwise.
- Let E be a vector indexed by Σ^* whose coordinates are all zero except the first one equals to 1: $E[u] = \mathbf{1}_{u=\epsilon}$ and let P be the vector indexed by Σ^* defined by $P[u] \coloneqq r(u)$.

- Let $\mathbf{H} = \mathbf{L}\mathbf{D}\mathbf{R}^\top$ be a reduced singular value decomposition of $\mathbf{H}$: $\mathbf{R}$ (resp. $\mathbf{L}$) is a matrix whose columns form a set of orthonormal vectors - the right (resp. left) singular vectors of $\mathbf{H}$ - and $\mathbf{D}$ is a $d \times d$ diagonal matrix, composed of the singular values of $\mathbf{H}$.

Then, $\langle \mathbf{R}^\top E, (\mathbf{R}^\top \mathbf{T}_x \mathbf{R})_{x\in\Sigma}, \mathbf{R}^\top P\rangle$ is a linear representation of r (Bailly et al., 2009; Hsu et al., 2009; Bailly, 2011; Balle et al., 2012).

Proposition 2 *$\langle \mathbf{R}^\top E, (\mathbf{R}^\top \mathbf{T}_x \mathbf{R})_{x\in\Sigma}, \mathbf{R}^\top P\rangle$ is a linear representation of r*

Proof See Appendix. ■

The basic spectral algorithm for learning rational stochastic languages aims at identifying the canonical linear representation of the target p determined by its Hankel matrix $\mathbf{H}_p$.

Let S be a sample independently drawn according to p:

- choose sets $U, V \subseteq \Sigma^\star$ and build the Hankel matrix $\mathbf{H}_S^{U\times V}$,
- choose a rank d, compute a SVD of $\mathbf{H}_S^{U\times V}$, and consider the d right singular vectors $\mathbf{R}_S$ associated with the d largest singular values,
- build the canonical linear representation $\langle \mathbf{R}_S^\top E, (\mathbf{R}_S^\top \mathbf{T}_x \mathbf{R}_S)_{x\in\Sigma}, \mathbf{R}_S^\top P_S\rangle$ where E and P are the vectors indexed by V s.t. $E[v] = \mathbf{1}_{v=\epsilon}$ and $P[v] \coloneqq p_S(v)$.

Alternative learning strategies consist in learning $\overline{p}$ or $\widehat{p}$, using the same algorithm, and then to compute an estimate of p. In all cases, the accuracy of the learned representation mainly depends on the estimation of $\mathbf{R}$. The Stewart formula (Stewart, 1990) bounds the principle angle θ between the spaces spanned by the right singular vectors of $\mathbf{R}$ and $\mathbf{R}_S$:

$$|\sin(\theta)| \leq \frac{\left\|\mathbf{H}_S^{U\times V} - \mathbf{H}_r^{U\times V}\right\|}{\sigma_{\min}(\mathbf{H}_r^{U\times V})}.$$

According to this formula, the concentration of the Hankel matrix around its mean is critical and the question of limiting the sizes of U and V naturally arises. Note that the Stewart inequality does not give any clear indication on the impact or on the interest of limiting these sets. Indeed, it can be shown that both the numerator and the denominator of the right part of the inequality increase with U and V (see Appendix).

3. Matrix Bernstein bounds

Let p be a rational stochastic language over $\Sigma^\star$, let ξ be a random variable distributed according to p, let $U, V \subseteq \Sigma^\star$ and let $\mathbf{Z}(\xi) \in \mathbb{R}^{|U|\times|V|}$ be a random matrix. For instance, $\mathbf{Z}(\xi)$ may be equal to $\mathbf{H}_\xi^{U,V}$, $\overline{\mathbf{H}}_\xi^{U,V}$ or $\widehat{\mathbf{H}}_\xi^{U,V}$ (ξ will be often omitted for the sake of simplicity). Let S be sample of strings drawn independently according to p.

Concentration bounds for sum of random matrices can be used to estimate the spectral distance between the empirical matrix $\mathbf{Z}_S$ computed on the sample S and its mean.

However, most of classical inequalities depend on the dimensions of the matrices. For eample, the following result describes a simple matrix Bernstein inequality on sum of random matrices (Ahlswede and Winter, 2002; Tropp, 2012).

Suppose that $\mathbb{E}\mathbf{Z}(\xi) = 0$ and let $\nu(\mathbf{Z}) = \max\{\|\mathbb{E}\mathbf{Z}\mathbf{Z}^\top\|, \|\mathbb{E}\mathbf{Z}^\top\mathbf{Z}\|\}$. Then,

$$\Pr\left(\|\mathbf{Z}_S\| \geq \frac{t}{N}\right) \leq (d_1 + d_2)\ \exp\left(-\frac{t^2}{2N\nu(\mathbf{Z}) + 2Mt/3}\right) \tag{3}$$

where N is the size of S, d_1 and d_2 are the dimensions of the matrix $\mathbf{Z}$ and $\|\mathbf{Z}\| \leq M$ almost surely.

We would like to apply this result to $\mathbf{Z} = \mathbf{H}_\xi^{U,V} - \mathbb{E}\mathbf{H}_\xi^{U,V}$, $\mathbf{Z} = \overline{\mathbf{H}}_\xi^{U,V} - \mathbb{E}\overline{\mathbf{H}}_\xi^{U,V}$ and $\mathbf{Z} = \widehat{\mathbf{H}}_\xi^{U,V} - \mathbb{E}\widehat{\mathbf{H}}_\xi^{U,V}$. However, we will see that while $\left\|\mathbf{H}_\xi^{U,V}\right\|$ is bound, $\left\|\overline{\mathbf{H}}_\xi^{U,V}\right\| = \Omega(\max(|U|^{1/2}, |V|^{1/2}))$ in the worst case, and $\left\|\widehat{\mathbf{H}}_\xi^{U,V}\right\|$may be unbounded even for fixed U and V.

These concentration bounds get worse with both sizes of the matrices. Coming back to the discussion at the end of Section 2, they suggest to limit the size of the sets U and V, and therefore, to design strategies to choose optimal sets. However, dimension-free bounds can be obtained.

3.1 A dimension-free Matrix Bernstein bound theorem

We then use recent results from (Tropp, 2012; Hsu et al., 2012) to obtain dimension-free concentration bounds for Hankel matrices.

Theorem 3 *(Hsu et al., 2012). Let $\xi_1, \ldots, \xi_N$ be random variables, and for each $i = 1, \ldots, N$, let $\mathbf{X}_i(\xi_i)$ be a symmetric matrix-valued functional of ξ_i[1]. For any $\eta \in \mathbb{R}$ and any $t > 0$,*

$$Pr\left[\left\|\eta \sum_{i=1}^N \mathbf{X}_i - \sum_{i=1}^N \log \mathbb{E}[\exp(\eta \mathbf{X}_i)]\right\| > t\right] \leq \operatorname{Tr}\left(\mathbb{E}\left[-\eta \sum_{i=1}^N \mathbf{X}_i + \sum_{i=1}^N \log \mathbb{E}[\exp(\eta \mathbf{X}_i)]\right]\right) \cdot (e^t - t - 1)^{-1}.$$

A matrix Bernstein bound can be derived from previous Theorem.

Theorem 4 *(Hsu et al., 2012). If there exists $b > 0, \sigma > 0, k > 0$ s.t. for all $i = 1, \ldots, N$, $\mathbb{E}_i[\mathbf{X}_i] = 0$, $\|\mathbf{X}_i\| \leq b$, $\left\|\frac{1}{N}\sum_{i=1}^N \mathbb{E}_i(\mathbf{X}_i^2)\right\| \leq \sigma^2$ and $\mathbb{E}\left[\operatorname{Tr}\left(\frac{1}{N}\sum_{i=1}^N \mathbb{E}_i(\mathbf{X}_i^2)\right)\right] \leq \sigma^2 k$ almost surely, then for all $t > 0$,*

$$Pr\left[\left\|\frac{1}{N}\sum_{i=1}^N \mathbf{X}_i\right\| > \sqrt{\frac{2\sigma^2 t}{N}} + \frac{bt}{3N}\right] \leq k \cdot t(e^t - t - 1)^{-1}.$$

1. (Hsu et al., 2011) consider the more general case, where $\mathbf{X}_i$ be a symmetric matrix-valued functional of $\xi_1, \ldots, \xi_i$.

Previous theorem is valid for symmetric matrices, but it can be extended to general real-valued matrices thanks to the principle of dilation.

Let $\mathbf{Z}$ be a matrix, the *dilation* of $\mathbf{Z}$ is the symmetric matrix $\mathbf{X}$ defined by

$$\mathbf{X} = \begin{bmatrix} 0 & \mathbf{Z} \\ \mathbf{Z}^T & 0 \end{bmatrix}. \text{ Then } \mathbf{X}^2 = \begin{bmatrix} \mathbf{Z}\mathbf{Z}^T & 0 \\ 0 & \mathbf{Z}^T\mathbf{Z} \end{bmatrix}$$

and $\|\mathbf{X}\| = \|\mathbf{Z}\|$, $\mathrm{Tr}(\mathbf{X}^2) = \mathrm{Tr}(\mathbf{Z}\mathbf{Z}^T) + \mathrm{Tr}(\mathbf{Z}^T\mathbf{Z})$ and $\left\|\mathbf{X}^2\right\| \leq \max(\left\|\mathbf{Z}\mathbf{Z}^T\right\|, \left\|\mathbf{Z}^T\mathbf{Z}\right\|)$.

We can then reformulate previous theorem as follows.

Theorem 5 *Let $\xi_1, \ldots, \xi_N$ be i.i.d. random variables, and for $i = 1, \ldots, N$, let $\mathbf{Z}_i = \mathbf{Z}(\xi_i)$ be i.i.d. matrices and $\mathbf{X}_i$ the dilation of $\mathbf{Z}_i$. If there exists $b > 0, \sigma > 0$, and $k > 0$ such that $\mathbb{E}[\mathbf{X}_1] = 0$, $\|\mathbf{X}_1\| \leq b$, $\left\|\mathbb{E}(\mathbf{X}_1^2)\right\| \leq \sigma^2$ and $\mathrm{Tr}(\mathbb{E}(\mathbf{X}_1^2)) \leq \sigma^2 k$ almost surely, then for all $t > 0$,*

$$Pr\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i\right\| > \sqrt{\frac{2\sigma^2 t}{N}} + \frac{bt}{3N}\right] \leq k \cdot t(e^t - t - 1)^{-1}.$$

We will then make use of this theorem to derive our new concentration bounds. Section 4.1 deals with the standard case, Section 4.3 with the prefix case and Section 4.5 with the factor case.

3.2 The subexponential case

Theorem 5 needs that the random matrices are bounded. However, the norm of $\overline{\mathbf{H}}_{\xi}^{U,V}$ depends on the size of U and V and $\widehat{\mathbf{H}}_{\xi}^{U,V}$ may be unbounded even in U and V are finite. Fortunately, Bernstein inequalities for subexponential random variables have been extended to unbounded random matrices whose moments grow at a limited rate [Th. 6.2 in (Tropp, 2012)]. We adapt this result to the dimension-free case in a similar way as what has been done in (Hsu et al., 2012) for Theorem 4.

Theorem 6 *[Matrix Bernstein bound: subexponential case.] If there exist $k > 0$, $R > 0$, and a symmetric matrix $\mathbf{A}$ such that for all $i = 1, \ldots, N$, $\mathbb{E}\mathbf{X}_i = 0$, $\mathbb{E}\mathbf{X}_i^p \preccurlyeq \frac{p!}{2}R^{p-2}\mathbf{A}^2$ for any integer $p \geq 2$, $\mathrm{Tr}(\mathbf{A}^2) \leq k\sigma^2$ where $\sigma^2 = \left\|\mathbf{A}^2\right\|$, then for any $t > 0$,*

$$Pr\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i\right\| > \frac{Rt}{N} + \sqrt{\frac{2\sigma^2 t}{N}}\right] \leq k \cdot t(e^t - t - 1)^{-1}.$$

Proof Let $0 < \eta < 1/R$. We have

$$\mathbb{E}e^{\eta\mathbf{X}_i} = \mathbb{E}\sum_{p\geq 0}\frac{1}{p!}\eta^p\mathbf{X}_i^p = \mathbf{I} + \sum_{p\geq 2}\frac{1}{p!}\eta^p\mathbb{E}\mathbf{X}_i^p \preccurlyeq \mathbf{I} + \frac{\eta^2}{2(1-\eta R)}\mathbf{A}^2.$$

Hence, by using the monotonicity of the logarithm function and the transfer rule applied to the inequality $\log(1+x) \leq x$,

$$\log\mathbb{E}e^{\eta\mathbf{X}_i} \preccurlyeq \frac{\eta^2}{2(1-\eta R)}\mathbf{A}^2.$$

Now, let η and t such that

$$\left\|\eta\sum_{i=1}^{N}\mathbf{X}_i - \sum_{i=1}^{N}\log\mathbb{E}e^{\eta\mathbf{X}_i}\right\| \leq t.$$

Then,

$$\left\|\eta\sum_{i=1}^{N}\mathbf{X}_i\right\| \leq t + \left\|\sum_{i=1}^{N}\log\mathbb{E}e^{\eta\mathbf{X}_i}\right\| \leq t + N\frac{\eta^2}{2(1-\eta R)}\left\|\mathbf{A}^2\right\|$$

and

$$\left\|\frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i\right\| \leq \frac{t}{N\eta} + \frac{\eta}{2(1-\eta R)}\sigma^2.$$

Moreover,

$$\mathrm{Tr}\left(\mathbb{E}\left[-\eta\sum_{i=1}^{N}\mathbf{X}_i + \sum_{i=1}^{N}\log\mathbb{E}e^{\eta\mathbf{X}_i}\right]\right) \leq \frac{N\eta^2}{2(1-\eta R)}k\sigma^2.$$

Hence,

$$\begin{aligned}\Pr\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i\right\| > \frac{t}{N\eta} + \frac{\eta}{2(1-\eta R)}\sigma^2\right] &\leq \Pr\left[\left\|\eta\sum_{i=1}^{N}\mathbf{X}_i - \sum_{i=1}^{N}\log\mathbb{E}e^{\eta\mathbf{X}_i}\right\| > t\right] \\ &\leq \frac{N\eta^2}{2(1-\eta R)}k\sigma^2(e^t - t - 1)^{-1} \text{from Theorem 3.}\end{aligned}$$

It can easily be checked that $\frac{t}{N\eta} + \frac{\eta}{2(1-\eta R)}\sigma^2$ takes its minimal positive value m at

$$\eta_{\min} = \frac{\sqrt{\frac{2t}{N\sigma^2}}}{1 + R\sqrt{\frac{2t}{N\sigma^2}}} \text{ and that } m = \frac{Rt}{N} + \sqrt{\frac{2\sigma^2 t}{N}}.$$

We have also

$$\frac{\eta_{\min}^2}{1 - R\eta_{\min}} \leq \left(\frac{\eta_{\min}}{1 - R\eta_{\min}}\right)^2 = \frac{2t}{N\sigma^2}$$

which entails the theorem. ■

4. Concentration bounds for Hankel matrices: main results

In this section, we present the main results of the paper. All proofs are reported in Section 5.

Let p be a rational stochastic language over Σ^*, let S be a sample independently drawn according to p, and let $U, V \subseteq \Sigma^*$ be finite set of strings.

4.1 Concentration Bound for the Hankel Matrix $\mathbf{H}_p^{U,V}$

We first describe a bound on $\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|$, independent from the sizes of U and V.

Let ξ be a random variable distributed according to p, let $\mathbf{Z}(\xi) = \mathbf{H}_{\xi}^{U,V} - \mathbf{H}_p^{U,V}$ be the random matrix defined by $\mathbf{Z}[u,v] = \mathbf{1}_{\xi=uv} - p(uv)$ and let $\mathbf{X}$ be the dilation of $\mathbf{Z}$.

Clearly, $\mathbb{E}\,\mathbf{X} = 0$. In order to apply Theorem 5, it is necessary to compute the parameters b, σ and k. We show in Lemma 14 that $\|\mathbf{X}\| \leq 2$, $\mathbb{E}\,Tr(\mathbf{X}^2) \leq 2S_p^{(2)}$ and $\left\|\mathbb{E}\,\mathbf{X}^2\right\| \leq S_p^{(2)}$ which entails that 4 conditions of Theorem 5 are fulfilled with $b = 2, \sigma^2 = S_p^{(2)}$ and $k = 2$.

Theorem 7 *Let p be a rational stochastic language and let S be a sample of N strings drawn i.i.d. from p. For all $t > 0$,*

$$Pr\left[\left\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\right\| > \sqrt{\frac{2S_p^{(2)}t}{N}} + \frac{2t}{3N}\right] \leq 2t(e^t - t - 1)^{-1}.$$

This bound is independent from U and V. It can be noticed that the proof of Lemma 14 also provides a dimension dependent bound by replacing $S_p^{(2)}$ with $\sum_{(u,v)\in U\times V} p(uv)$, which may result in a significative improvement if U or V are small.

The moment $S_p^{(2)}$ is generally unknown. However, it can be estimated from S. Indeed, $S_p^{(2)} = \sum_{u,v\in\Sigma^*} p(uv) = \sum_{w\in\Sigma^*}(|w|+1)p(w) = \mathbb{E}|\xi|+1$ and $\frac{1}{N}\sum_{w\in S}|w|+1$ is an natural estimate for $S_p^{(2)}$.The random variable $|\xi|$ is sub-exponential and its concentration around its mean can be estimated using Bernstein-type inequalities (see (Vershynin, 2012) for example). Thus, Theorem 7 can easily be reformulated replacing $S_p^{(2)}$ by its estimate.

4.2 Concentration Bound for the smoothed prefix Hankel Matrix $H_{\overline{p}_\eta}^{U,V}$

The random matrix $\overline{\mathbf{Z}}(\xi) = \overline{\mathbf{H}}_\xi^{U,V} - \mathbf{H}_{\overline{p}}^{U,V}$ is defined by $\overline{\mathbf{Z}}[u,v] = \mathbf{1}_{uv\in Pref(\xi)} - \overline{p}(uv)$, where references to U and V are omitted for the sake of readability. It can easily be shown that $\left\|\overline{\mathbf{Z}}\right\|$ may be unbounded if U or V are unbounded. For example, consider the stochastic language defined on a one-letter alphabet $\Sigma = \{a\}$ by $p(a^n) = (1-\rho)\rho^n$. If $U = V = \Sigma^{\leq n}$, $\overline{\mathbf{Z}}$ may be equal to the $(n+1)\times(n+1)$ upper triangular all-ones matrix whose norm is $\Theta(n)$. Hence, Theorem 5 cannot be directly applied to obtain dimension-free bounds. This suggests that the concentration of $\overline{\mathbf{Z}}$ around its mean could be far weaker than the concentration of $\mathbf{Z}$.

For any $\eta \in [0,1]$, we define a smoothed variant[2] of $\overline{p}$ by

$$\overline{p}_\eta(u) \coloneqq \sum_{x\in\Sigma^*} \eta^{|x|} p(ux) = \sum_{n\geq 0} \eta^n p(u\Sigma^n). \tag{4}$$

Note that $\overline{p}_1 = \overline{p}$, $\overline{p}_0 = p$ and that $p(u) \leq \overline{p}_\eta(u) \leq \overline{p}(u)$ for every string u: the functions $\overline{p}_\eta$ are natural intermediates between p and $\overline{p}$.

Any function $\overline{p}_\eta$ can be used to compute p:

$$p(u) = \overline{p}_\eta(u) - \eta\overline{p}_\eta(u\Sigma). \tag{5}$$

Moreover, when p is rational, each $\overline{p}_\eta$ is also rational and a linear representation of p can be derived from any linear representation of $\overline{p}_\eta$. More precisely,

2. Note that our *smoothed variant* can also be interpreted as a *discounted variant* since the parameter η helps to reduce the impact of long strings.

Proposition 8 *Let p be a rational stochastic language, let $\langle I, (\mathbf{M}_x)_{x\in\Sigma}, T\rangle$ be a minimal linear representation of p and let $\overline{T}_\eta = (\mathbf{I}_d - \eta\mathbf{M}_\Sigma)^{-1}T$. Then, $\langle I, (\mathbf{M}_x)_{x\in\Sigma}, \overline{T}_\eta\rangle$ is a linear representation of $\overline{p}_\eta$. Hence, $S_{\overline{p}_\eta}^{(k)} = I^T(\mathbf{I}_d - \mathbf{M}_\Sigma)^{-k}(\mathbf{I}_d - \eta\mathbf{M}_\Sigma)^{-1}T$. In particular, $S_{\overline{p}}^{(k)} = S_p^{(k+1)}$.*

Therefore, it is a consistent learning strategy to learn $\overline{p}_\eta$ from the data, for some η, and next, to derive p. A theoretical study that would guide the choice of the parameter η remains to be done. In the absence of such indications, its value can be set by cross validation.

For any $0 \leq \eta \leq 1$, let $\overline{\mathbf{Z}}_\eta(\xi)$ be the random matrix defined by

$$\overline{\mathbf{Z}}_\eta[u,v] \coloneqq \sum_{x\in\Sigma^*} \eta^{|x|}\mathbf{1}_{\xi=uvx} - \overline{p}_\eta(uv) = \sum_{x\in\Sigma^*} \eta^{|x|}(\mathbf{1}_{\xi=uvx} - p(uvx)).$$

for any $(u,v) \in U\times V$. It is clear that $\mathbb{E}\,\overline{\mathbf{Z}}_\eta = 0$. We show that $\left\|\overline{\mathbf{Z}}_\eta\right\|$ is bounded by $\frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}$ if $\eta < 1$ (Lemma 15), that $\left\|\mathbb{E}\,\overline{\mathbf{X}}_\eta^2\right\| \leq S_{\overline{p}_\eta}^{(2)}$ and $\mathbb{E}\,Tr(\overline{\mathbf{X}}_\eta^2) \leq 2S_{\overline{p}_\eta}^{(2)}$. (Lemma 17).

Therefore, we can apply Theorem 5 with $b = \frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}, \sigma^2 = S_{\overline{p}_\eta}^{(2)}$ and $k = 2$.

Theorem 9 *Let p be a rational stochastic language, let S be a sample of N strings drawn i.i.d. from p and let $0 \leq \eta < 1$. For all $t > 0$,*

$$Pr\left[\left\|\overline{\mathbf{H}}_{\eta,S}^{U,V} - \mathbf{H}_{\overline{p}_\eta}^{U,V}\right\|_2 > \sqrt{\frac{2S_{\overline{p}_\eta}^{(2)}t}{N}} + \frac{t}{3N}\left[\frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}\right]\right] \leq 2t(e^t - t - 1)^{-1}.$$

Remark that when $\eta = 0$ we find back the concentration bound of Theorem 7. When U and V are finite, a careful examination of the proof of Lemma 15 shows that $S_{\overline{p}_\eta}^{(2)}$ can be replaced with $\sum_{(u,v)\in U\times V}\overline{p}_\eta(uv)$, which may provide a significant better bound if U and V are small. Moreover, Inequality 8 can be used to provide a finite bound depending on the sizes of U and V, when $\eta = 1$.

As for Theorem 7, the moment $S_{\overline{p}_\eta}^{(2)}$ is generally unknown but it can be estimated from S, with controled accuracy, providing a reformulation of the theorem that would not depend on this parameter.

4.3 Concentration Bound for the prefix Hankel Matrix $H_{\overline{p}}^{U,V}$

Theorem 9 does not provide a dimension-free bound for the prefix Hankel matrices $\overline{\mathbf{H}}_S^{U,V}$. However, we show that $\overline{\mathbf{Z}}(\xi)$ is a subexponential random matrix (Lemma 21), and that Theorem 6 can be used to provide a bound in this case.

More precisely, let $\mathbf{X}(\xi)$ be the dilation of the matrix $\overline{\mathbf{Z}}(\xi)$, let $C_p > 0$ and $0 < \rho_p < 1$ be such that $p(\Sigma^n) \leq C_p\rho_p^n$ for any integer n.

For any $0 < \beta < -\ln\rho_p$, we show in Lemma 21 the existence of a symmetric matrix $\mathbf{A}^2$ satisfying $\left\|\mathbf{A}^2\right\| \leq K(1-\rho_p e^\beta)^{-1}, \mathrm{Tr}(\mathbf{A}^2) \leq 2K(1-\rho_p e^\beta)^{-2}$and such that for any $k \geq 0$,

$\mathbb{E}\mathbf{X}^k \preccurlyeq \frac{k!}{2} R^{k-2}\mathbf{A}^2$ where $R = e^{1/e}\beta^{-1}$ and $K = 2e^{3/e}\beta^{-3}C_p S_p^{(3)} e^{\beta S_p^{(2)}}$ is a constant that only depends on p and β.

Hence, we can apply Theorem 6 to obtain the following dimension free bound:

Theorem 10 *Let p be a rational stochastic language and let S be a sample of N strings drawn i.i.d. from p. For all $t > 0$,*

$$Pr\left[\left\|\overline{\mathbf{H}}_S^{U,V} - \mathbf{H}_{\overline{p}}^{U,V}\right\|_2 > \frac{e^{1/e}t}{N\beta} + \sqrt{\frac{2\sigma^2 t}{N}}\right] \leq \frac{2}{1-\rho_p e^{\beta}} \cdot t(e^t - t - 1)^{-1}$$

where $\sigma^2 = K(1-\rho_p e^{\beta})^{-2}$, $K = \beta^{-2}C_p S_p^{(3)} e^{\beta S_p^{(2)}}$ and $0 < \beta < -\ln \rho_p$.

Note that β can be set to $1-\rho_p$ but depending on the particular values of the terms, β can be adjusted to obtain better bounds.

Thus, Theorem 10 describes a dimension free concentration bound for the prefix Hankel matrix. However, the constants K and σ that occur in this bound can be very large, which makes it impossible to use it in practical cases, such as those we consider in Section 6.

4.4 Concentration Bound for the smoothed factor Hankel Matrix $H_{\widehat{p}_\eta}^{U,V}$

The random matrix $\widehat{\mathbf{Z}}(\xi) = \widehat{\mathbf{H}}_{\xi}^{U,V} - \mathbf{H}_{\widehat{p}^{U,V}}$ is defined by

$$\widehat{\mathbf{Z}}[u,v] = \sum_{x,y\in\Sigma^*} \mathbf{1}_{\xi=xuvy} - \widehat{p}(uv),$$

where references to U and V are omitted for the sake of readability. $\left\|\widehat{\mathbf{Z}}\right\|$ is unbounded if the support of p is unbounded. Indeed, $\widehat{\mathbf{Z}}[\epsilon,\epsilon] = |\xi| + 1 - \widehat{p}(\epsilon)$. Hence, Theorem 5 cannot be directly applied either.

We can also define smoothed variants $\widehat{p}_\eta$ of $\widehat{p}$, for any $\eta \in [0,1]$ by

$$\widehat{p}_\eta(u) = \sum_{x,y\in\Sigma^*} \eta^{|xy|} p(xuy) = \sum_{m,n\geq 0} \eta^{m+n} p(\Sigma^m u \Sigma^n)$$

which have properties similar to functions $\overline{p}_\eta$:

- $p \leq \widehat{p}_\eta \leq \widehat{p}$, $\widehat{p}_1 = \widehat{p}$ and $\widehat{p}_0 = p$,
- when p is rational, each $\widehat{p}_\eta$ is also rational,
- if $\langle I, (\mathbf{M}_x)_{x\in\Sigma}, T\rangle$ be a minimal linear representation of p then $\langle \widehat{I}_\eta, (\mathbf{M}_x)_{x\in\Sigma}, \overline{T}_\eta\rangle$ is a linear representation of $\hat{p}_\eta$, where $\widehat{I}_\eta = (I_d - \eta\mathbf{M}_\Sigma^{\mathrm{T}})^{-1} I$,
- I and T can be computed from $\widehat{I}_\eta$ and $\widehat{T}_\eta$ when η and $\mathbf{M}_\Sigma$ are known:

$$T = (\mathbf{I}_d - \eta\mathbf{M}_\Sigma)\widehat{T}_\eta, I = (\mathbf{I}_d - \eta\mathbf{M}_\Sigma)\widehat{I}_\eta \text{ and} \tag{6}$$

$$p(u) = \widehat{p}_\eta(u) - \eta\widehat{p}_\eta(u\Sigma) - \eta\widehat{p}_\eta(\Sigma u) + \eta^2\widehat{p}_\eta(\Sigma u\Sigma) \tag{7}$$

- therefore, it is a consistent learning strategy to learn $\widehat{p}_\eta$ from the data, for some η, and next, to derive p.

For $\eta \in [0,1]$, let $\widehat{\mathbf{Z}}_\eta(\xi)$ be the random matrix defined by

$$\widehat{\mathbf{Z}}_\eta[u,v] = \sum_{x,y\in\Sigma^*} \eta^{|xy|}\mathbf{1}_{\xi=xuvy} - \widehat{p}_\eta(uv) = \sum_{x,y\in\Sigma^*} \eta^{|xy|}(\mathbf{1}_{\xi=xuvy} - p(xuvy))$$

for any $(u,v) \in U\times V$. Clearly, $\mathbb{E}\,\widehat{\mathbf{Z}}_\eta = 0$. We show that $\left\|\widehat{\mathbf{Z}}_\eta\right\|$ is bounded by $(1-\eta)^{-2} + S_{\widehat{p}_\eta}^{(1)}$ if $\eta < 1$ (lemma 24).

While $\overline{p}$ is bounded by 1, a property which is often used in the proofs, $\widehat{p}$ is unbounded when η converges to 1. Let us introduce a new constant K_η defined by

$$K_\eta = \begin{cases} 1 & \text{if } \eta \leq e^{-1} \\ (-e\eta\ln\eta)^{-1} & \text{otherwise.} \end{cases}$$

We show in Lemma 26 that

$$\left\|\mathbb{E}\widehat{\mathbf{X}}_\eta^2\right\| \leq K_\eta S_{\widehat{p}_\eta}^{(2)} \text{ and } Tr(\mathbb{E}(\widehat{\mathbf{X}}_\eta^2)) \leq 2K_\eta S_{\widehat{p}_\eta}^{(2)}.$$

Eventually, we can apply Theorem 5 with $b = (1-\eta)^{-2} + S_{\widehat{p}_\eta}^{(1)}, \sigma^2 = K_\eta S_{\widehat{p}_\eta}^{(2)}$ and $k = 2$.

Theorem 11 *Let p be a rational stochastic language, let S be a sample of N strings drawn i.i.d. from p and let $0 \leq \eta < 1$. For all $t > 0$,*

$$Pr\left[\|\widehat{\mathbf{H}}_{\eta,S}^{U,V} - \mathbf{H}_{\widehat{p}_\eta}^{U,V}\|_2 > \sqrt{\frac{2K_\eta S_{\widehat{p}_\eta}^{(2)} t}{N}} + \frac{t}{3N}\left[\frac{1}{(1-\eta)^2} + S_{\widehat{p}_\eta}^{(1)}\right]\right] \leq 2t(e^t - t - 1)^{-1}.$$

Remark that when $\eta = 0$ we find back the concentration bound of Theorem 7. As in the previous cases, $S_{\widehat{p}_\eta}^{(2)}$ can be replaced with $\sum_{(u,v)\in U\times V}\widehat{p}_\eta(uv)$, which may provide a significant better bound if U and V are small. However, it not possible to use these results to obtain a bound, even depending on U and V, when $\eta = 1$.

As for the previous theorems, the moment $S_{\widehat{p}_\eta}^{(2)}$ can be estimated from S in order to provide a reformulation of the theorem that would not depend on this parameter.

4.5 Concentration Bound for the factor Hankel Matrix $H_{\widehat{p}}^{U,V}$

$\widehat{\mathbf{Z}}(\xi)$ is not a subexponential random matrix, and therefore, Theorem 6 cannot be used to provide a bound in this case. This suggests that the concentration of $\widehat{\mathbf{Z}}(\xi)$ around its mean is quite loose.

5. Proofs of all concentration bounds results

In this section, we detail the proofs of all the results stated in Section 4, keeping the titles of all subsections and the notations that have been introduced in the corresponding subsection.

5.1 Concentration Bound for the Hankel Matrix $\mathbf{H}_p^{U,V}$: proofs

Recall that $\mathbf{X}$ is the dilation of the random matrix $\mathbf{Z}(\xi) = \mathbf{H}_\xi^{U,V} - \mathbf{H}_p^{U,V}$.

Clearly, $\mathbb{E}\,\mathbf{X} = 0$. We need technical lemmas in order to obtains bound on $\mathbb{E}\,\mathbf{X}^2$ and $\mathbb{E}\,Tr(\mathbf{X}^2)$ and apply Theorem 5.

Lemma 12 *Let X and Y be two random variables such that $0 \leq X, Y \leq M$. Then,*

$$|\mathbb{E}\,(X - \mathbb{E}\,X)(Y - \mathbb{E}\,Y)| \leq M \min\{\mathbb{E}\,X, \mathbb{E}\,Y\}.$$

Proof Indeed, $0 \leq \mathbb{E}\,XY \leq M \min\{\mathbb{E}\,X, \mathbb{E}\,Y\}$ and $0 \leq \mathbb{E}\,X\mathbb{E}\,Y \leq M \min\{\mathbb{E}\,X, \mathbb{E}\,Y\}$, which entails the lemma. ■

Lemma 13 *For any $u, u' \in U$, $v, v' \in V$,*

$$|\mathbb{E}\,\mathbf{Z}[u,v]\mathbf{Z}[u',v']| \leq \min\{p(uv), p(u'v')\}.$$

Proof This a corollary of Lemma 12 with $X \coloneqq \mathbf{1}_{\xi=uv}$, $Y \coloneqq \mathbf{1}_{\xi=u'v'}$ and $M = 1$. ■

Lemma 14 *$\|\mathbf{X}\| \leq 2$, $\mathbb{E}\,Tr(\mathbf{X}^2) \leq 2S_p^{(2)}$ and $\left\|\mathbb{E}\,\mathbf{X}^2\right\| \leq S_p^{(2)}$.*

Proof

1. $\forall u \in U$, $\sum_{v\in V} |\mathbf{Z}[u,v]| = \sum_{v\in V} |\mathbf{1}_{\xi=uv} - p(uv)| \leq 1 + p(u\Sigma^*) \leq 2$. Therefore, $\|\mathbf{Z}\|_\infty \leq 2$. In a similar way, it can be shown that $\|\mathbf{Z}\|_1 \leq 2$. Hence,

$$\|\mathbf{X}\| = \|\mathbf{Z}\| \leq \sqrt{\|\mathbf{Z}\|_\infty \|\mathbf{Z}\|_1} \leq 2.$$

2. For all $(u,u') \in U^2$: $\mathbf{Z}\mathbf{Z}^T[u,u'] = \sum_{v\in V} \mathbf{Z}[u,v]\mathbf{Z}_{u',v}$. Therefore,

$$\mathbb{E}\,Tr(\mathbf{Z}\mathbf{Z}^T) = \mathbb{E}\sum_{u\in U} \mathbf{Z}\mathbf{Z}^T[u,u] = \mathbb{E}\sum_{u\in U, v\in V} \mathbf{Z}[u,v]\mathbf{Z}[u,v] \leq \sum_{u,v\in\Sigma^*} \mathbb{E}\,\mathbf{Z}[u,v]^2 \leq \sum_{u,v\in\Sigma^*} p(uv) \leq S_p^{(2)}.$$

In a similar way, it can be proved that $\mathbb{E}\,Tr(\mathbf{Z}^T\mathbf{Z}) \leq S_p^{(2)}$. Therefore, $\mathbb{E}\,Tr(\mathbf{X}^2) \leq 2S_p^{(2)}$.

3. For any $u \in U$,

$$\sum_{u'\in U} |\mathbb{E}\,\mathbf{Z}\mathbf{Z}^T[u,u']| \leq \sum_{u'\in U, v\in V} |\mathbb{E}\,\mathbf{Z}[u,v]\mathbf{Z}[u',v]| \leq \sum_{u'\in U, v\in V} p(u'v) \leq S_p^{(2)}.$$

Hence, $\left\|\mathbf{Z}\mathbf{Z}^T\right\|_\infty \leq S_p^{(2)}$. It can be proved, in a similar way, that $\left\|\mathbf{Z}^T\mathbf{Z}\right\|_\infty \leq S_p^{(2)}$, $\left\|\mathbf{Z}\mathbf{Z}^T\right\|_1 \leq S_p^{(2)}$ and $\left\|\mathbf{Z}^T\mathbf{Z}\right\|_1 \leq S_p^{(2)}$. Therefore, $\left\|\mathbf{X}^2\right\| \leq S_p^{(2)}$. ■

5.2 Concentration Bound for the smoothed prefix Hankel Matrix $H_{\overline{p}_\eta}^{U,V}$: proofs

Proof (Proposition 8.)

For every string u, $\overline{p}_\eta(u) = \sum_{n\geq 0} I^T\mathbf{M}_u\eta^n\mathbf{M}_\Sigma^n T = I^T\mathbf{M}_u(\sum_{n\geq 0}\eta^n\mathbf{M}_\Sigma^n)T = I^T\mathbf{M}_u\overline{T}_\eta$. The expression of $S_{\overline{p}_\eta}^{(k)}$ comes directly from Equation 2. ■

Lemma 15 *For any $U, V \subseteq \Sigma^*$,*

$$\left\|\overline{\mathbf{Z}}_\eta\right\| \leq \frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}.$$

Proof Indeed, let $u \in U$.

$$\begin{aligned}\sum_{v\in V} |\overline{\mathbf{Z}}_\eta[u,v]| &\leq \sum_{v,x\in\Sigma^*} \eta^{|x|}\mathbf{1}_{\xi=uvx} + \sum_{v\in\Sigma^*} \overline{p}_\eta(uv)\\ &\leq (1 + \eta + \ldots + \eta^{|\xi|-|u|}) + S_{\overline{p}_\eta}^{(1)}\\ &\leq \frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}.\end{aligned}$$

Hence, $\left\|\overline{\mathbf{Z}}_\eta\right\|_\infty \leq \frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}$. Similarly, $\left\|\overline{\mathbf{Z}}_\eta\right\|_1 \leq \frac{1}{1-\eta} + S_{\overline{p}_\eta}^{(1)}$, which completes the proof. ■

When U and V are bounded, let l be the maximal length of a string in $U \cup V$. It can easily be shown that $\left\|\overline{\mathbf{Z}}_\eta\right\| \leq l + 1 + S_{\overline{p}_\eta}^{(1)}$ and therefore, in that case,

$$\left\|\overline{\mathbf{Z}}_\eta\right\| \leq Min(l+1, \frac{1}{1-\eta}) + S_{\overline{p}_\eta}^{(1)} \tag{8}$$

which holds even if $\eta = 1$.

Lemma 16 *$|\mathbb{E}(\overline{\mathbf{Z}}_\eta[u,v]\overline{\mathbf{Z}}_\eta[u',v'])| \leq \min\{\overline{p}_\eta(uv), \overline{p}_\eta(u'v')\}$, for any $u, u' \in U$ and $v, v' \in V$.*

Proof This a corollary of Lemma 12 with $X \coloneqq \sum_{x\in\Sigma^*} \eta^{|x|}\mathbf{1}_{\xi=uvx}$, $Y \coloneqq \sum_{x\in\Sigma^*} \eta^{|x|}\mathbf{1}_{\xi=u'v'x}$ and $M = 1$. ■

Let $\overline{\mathbf{X}}_\eta$ be the dilation of $\overline{\mathbf{Z}}_\eta$. We can now propose bounds for $\mathbb{E}\,\overline{\mathbf{X}}_\eta^2$ and $\mathbb{E}\,Tr(\overline{\mathbf{X}}_\eta^2)$.

Lemma 17

$$\left\|\mathbb{E}\,\overline{\mathbf{X}}_\eta^2\right\| \leq S_{\overline{p}_\eta}^{(2)} \text{ and } \mathbb{E}\,Tr(\overline{\mathbf{X}}_\eta^2) \leq 2S_{\overline{p}_\eta}^{(2)}.$$

Proof Indeed,

$$\left\|\mathbb{E}\,\overline{\mathbf{Z}}_\eta\overline{\mathbf{Z}}_\eta^T\right\|_\infty \leq \max_{u\in\Sigma^*} \sum_{u',v\in\Sigma^*} |\mathbb{E}\,\overline{\mathbf{Z}}_\eta[u,v]\overline{\mathbf{Z}}_\eta[u',v]| \leq \sum_{u',v} \overline{p}_\eta(u'v) \leq S_{\overline{p}_\eta}^{(2)}.$$

We have also

$$\left\|\mathbb{E}\,\overline{\mathbf{Z}}_\eta\overline{\mathbf{Z}}_\eta^T\right\|_1 \leq S_{\overline{p}_\eta}^{(2)} \text{ and therefore } \left\|\mathbb{E}\,\overline{\mathbf{Z}}_\eta\ \overline{\mathbf{Z}}_\eta^T\right\| \leq S_{\overline{p}_\eta}^{(2)}.$$

Similar computations provide all the inequalities. ■

5.3 Concentration Bound for the prefix Hankel Matrix $H_{\overline{p}}^{U,V}$: proofs

Let $\mathbf{X}(\xi)$ be the dilation of the matrix $\overline{\mathbf{Z}}(\xi)$. It can easily be shown that

$$\mathbf{X}^{2k+1} = \begin{bmatrix} 0 & \left(\overline{\mathbf{Z}}\cdot\overline{\mathbf{Z}}^\top\right)^k\overline{\mathbf{Z}} \\ \overline{\mathbf{Z}}^\top\left(\overline{\mathbf{Z}}\cdot\overline{\mathbf{Z}}^\top\right)^k & 0 \end{bmatrix} \text{ and } \mathbf{X}^{2k} = \begin{bmatrix} \left(\overline{\mathbf{Z}}\cdot\overline{\mathbf{Z}}^\top\right)^k & 0 \\ 0 & \left(\overline{\mathbf{Z}}^\top\cdot\overline{\mathbf{Z}}\right)^k \end{bmatrix}.$$

Let $t \in \Sigma^*$ be a realization of ξ.

Lemma 18 *For any strings $u, v, t \in \Sigma^*$ and any stochastic language p,*

$$\sum_{v\in V}|\overline{\mathbf{Z}}[u,v]| \le |t| + \overline{p}(u\Sigma^*),\ \sum_{u\in U}|\overline{\mathbf{Z}}[u,v]| \le |t| + \overline{p}(\Sigma^* u)$$

and

$$\sum_{u\in U, v\in V}|\overline{\mathbf{Z}}[u,v]| \le \frac{|t|(|t|+3)}{2} + S_p^{(3)}.$$

Proof We have $\sum_{v\in V}|\overline{\mathbf{Z}}[u,v]| \le \sum_{w\in\Sigma^*, uw\in \mathrm{Pref}(t)}|1-\overline{p}(uw)| + \overline{p}(u\Sigma^*)$. Moreover, if $u = w = \epsilon$, $\overline{p}(uw) = 1$. Hence, there are at most $|t|$ strings w such that $uw \in \mathrm{Pref}(t)$ and $|1-\overline{p}(uw)| \neq 0$, which proves the first inequality. The second one is proved in a similar way. A similar argument proves that there are at most $(|t|+1)(|t|+2)/2 - 1 = |t|(|t|+3)/2$ pairs of words u, w such that $uw \in \mathrm{Pref}(t)$ and $|1-\overline{p}(uw)| \neq 0$, which entails the third inequality. ■

Lemma 19 *Let $\mathbf{M}$ be a matrix of the form $(\overline{\mathbf{Z}}^\top)^e(\overline{\mathbf{Z}}\cdot\overline{\mathbf{Z}}^\top)^k\overline{\mathbf{Z}}^f$, where $k \in \mathbb{N}$ and $e, f \in \{0,1\}$. Then, for any strings $u, v \in \Sigma^*$, $|\mathbf{M}[u,v]| \le (|t| + S_p^{(2)})^h$ where $h = e + 2k + f$. Moreover, $\sum_{u\in U}|\mathbf{M}[u,v]|$ and $\sum_{v\in V}|\mathbf{M}[u,v]|$ are bounded by $S_p^{(3)}(|t|+1)(|t|+S_p^{(2)})^h$.*

Proof By induction on $h = e + 2k + f$. The inequality is obvious if $h = 0$. Let $\mathbf{M} = \overline{\mathbf{Z}}\mathbf{N}$.

$$|\mathbf{M}[u,v]| \le \sum_{w\in V}|\overline{\mathbf{Z}}[u,w]\mathbf{N}[w,v]| \le (|t| + S_p^{(2)})^{h-1}\sum_{w\in V}|\overline{\mathbf{Z}}[u,w]| \text{ by induction hypothesis.}$$

By Lemma 18, we have

$$|\mathbf{M}[u,v]| \le (|t| + S_p^{(2)})^h \text{ since } \overline{p}(u\Sigma^*) \le S_p^{(2)}$$

and

$$\sum_{u\in U}|\mathbf{M}[u,v]| \le (|t| + S_p^{(2)})^{h-1}\left(\frac{|t|(|t|+3)}{2} + S_p^{(3)}\right) \le S_p^{(3)}(|t|+1)(|t|+S_p^{(2)})^h$$

since $1 \le S_p^{(3)}$ and $(\frac{|t|(|t|+3)}{2} + 1) \le (|t|+1)(|t|+S_p^{(2)})$.

The other cases are proved in a similar way. ■

Corollary 20 *For any integer k, $\left\|\mathbf{X}^k\right\| \le S_p^{(3)}(|t|+1)(|t|+S_p^{(2)})^k$.*

Proof Indeed, it can easily be shown that for any $k \in \mathbb{N}$ and $e \in \{0,1\}$, $\left\|\mathbf{X}^{2k+e}\right\| = \left\|\left(\overline{\mathbf{Z}}\cdot\overline{\mathbf{Z}}^\top\right)^k\overline{\mathbf{Z}}^e\right\|$. The result is a consequence of Lemma 19. ■

Let $C_p > 0$ and $0 < \rho_p < 1$ be such that $p(\Sigma^n) \le C_p\rho_p^n$.

Let $\mathbf{X}^k(t) = \mathbf{U}_t^\top diag(\lambda_1, \ldots, \lambda_r, 0, \ldots, 0)\mathbf{U}_t$ be an eigenvalue decomposition of $\mathbf{X}^k$, where $\lambda_1, \lambda_2, \ldots, \lambda_r$ are the non zero eigenvalues of $\mathbf{X}^k(t)$, and let $\mathbf{J}_t = diag(1, \ldots, 1, 0, \ldots, 0)$ the matrix whose coefficients are all equal to 0 but the r upper diagonal elements which are equal to 1. We have

$$\mathbf{X}^k \preccurlyeq \mathbf{U}^\top diag(|\lambda_1|, \ldots, |\lambda_r|, 0, \ldots, 0)\mathbf{U} \preccurlyeq \left\|\mathbf{X}_t^k\right\|\mathbf{U}_t^\top\mathbf{J}_t\mathbf{U}_t. \tag{9}$$

For any $n \in \mathbb{N}$, let

$$\mathbf{M}_n = \sum_{t\in\Sigma^n}\frac{p(t)}{p(\Sigma^n)}\mathbf{U}_t^\top\mathbf{J}_t\mathbf{U}_t \text{ if } p(\Sigma^n) \neq 0 \text{ and } \mathbf{M}_n = 0 \text{ otherwise.}$$

We can remark that $\|\mathbf{M}_n\| \le 1$ and $0 \preccurlyeq \mathbf{M}_n$.

Let $\mathbf{A}$ be the symmetric matrix such that

$$\mathbf{A}^2 = K\sum_{n\ge 0}\rho_p^n e^{\beta n}\mathbf{M}_n$$

where $K = 2e^{3/e}\beta^{-3}C_pS_p^{(3)}e^{\beta S_p^{(2)}}$ and $0 < \beta < -\ln\rho_p$. For example, β can be taken equal to $1-\rho_p$.

Since $\rho_p e^\beta < 1$ and $\|\mathbf{M}_n\| \le 1$, $\mathbf{A}$ is well defined. Moreover, $0 \preccurlyeq \mathbf{A}^2$.

Lemma 21 *We have $\left\|\mathbf{A}^2\right\| \le K(1-\rho_p e^\beta)^{-1}, \mathrm{Tr}(\mathbf{A}^2) \le 2K(1-\rho_p e^\beta)^{-2}$ and for any $k \ge 0$, $\mathbb{E}\mathbf{X}^k \preccurlyeq \frac{k!}{2}R^{k-2}\mathbf{A}^2$ where $R = e^{1/e}\beta^{-1}$.*

Proof The bound on $\|\mathbf{A}\|$ is straightforward.

The rank of $\mathbf{X}^k$, equal to the rank of $\mathbf{J}_t$, is bounded by $2(|t|+1)$ and hence, $\mathrm{Tr}(\mathbf{M}_n) \le 2(n+1)$. The bound on $\mathrm{Tr}(\mathbf{A}^2)$ comes from the following classical equality: if $|x| < 1$ then, $\sum_{n\ge 0}(n+1)x^n = (1-x)^{-2}$.

We have

$$\begin{aligned}
\mathbb{E}\mathbf{X}^k &\preccurlyeq \sum_t p(t) S_p^{(3)} (|t|+1)(|t|+S_p^{(2)})^k \mathbf{U}_t^\top \mathbf{J}_t \mathbf{U}_t && \text{from } Eq\ 9 \text{ and } Cor.\ 20 \\
&\preccurlyeq \sum_t p(t) S_p^{(3)} (|t|+S_p^{(2)})^{k+1} \mathbf{U}_t^\top \mathbf{J}_t \mathbf{U}_t && \text{since } S_p^{(2)} \geq 1 \\
&= S_p^{(3)} \sum_{n\geq 0} p(\Sigma^n)(n+S_p^{(2)})^{k+1} \mathbf{M}_n \\
&\preccurlyeq C_p S_p^{(3)} \sum_{n\geq 0} \rho_p^n \frac{[(n+S_p^{(2)})\beta]^{k+1}}{(k+1)!} \frac{(k+1)!}{\beta^{k+1}} \mathbf{M}_n && \text{since } p(\Sigma^n) \leq C_p \rho_p^n \\
&\preccurlyeq C_p S_p^{(3)} \sum_{n\geq 0} \rho_p^n e^{(n+S_p^{(2)})\beta} \frac{(k+1)!}{\beta^{k+1}} \mathbf{M}_n && \text{since } x^k/k! \leq e^x \\
&\preccurlyeq k! \left(\frac{e^{1/e}}{\beta}\right)^k \frac{e^{1/e}}{\beta} C_p S_p^{(3)} e^{S_p^{(2)}\beta} \sum_{n\geq 0} (\rho_p e^\beta)^n \mathbf{M}_n && \text{since } k+1 \leq e^{(k+1)/e} \\
&= \frac{k!}{2} R^{k-2} \mathbf{A}^2.
\end{aligned}$$

■

5.4 Concentration Bound for the smoothed factor Hankel Matrix $H_{\widehat{p}_\eta}^{U,V}$: proofs

Lemma 22 *Let $0 < \eta \leq 1$. For any integer n, $(n+1)\eta^n \leq K_\eta$.*

Proof Let $f(x) = (x+1)\eta^x$. We have $f'(x) = \eta^x(1+(x+1)\ln\eta)$ and f takes its maximum for $x_M = -1 - 1/\ln\eta$, which is positive if and only if $\eta > 1/e$. We have $f(x_M) = (-e\eta\ln\eta)^{-1}$. ■

Lemma 23 *Let $w, u \in \Sigma^*$. Then,*

$$\sum_{x,y\in\Sigma^*} \eta^{|xy|} \mathbf{1}_{w=xuy} \leq K_\eta \text{ and } \widehat{p}(u) \leq K_\eta p(\Sigma^* u \Sigma^*).$$

Proof Indeed, if $w = xuy$, then $|xy| = |w| - |u|$ and u appears at most $|w| - |u| + 1$ times as a factor of w. Therefore, $\sum_{x,y\in\Sigma^*} \eta^{|xy|}\mathbf{1}_{w=xuy} \leq (|w|-|u|+1)\eta^{|w|-|u|} \leq K_\eta$. Moreover,

$$\widehat{p}(u) = \sum_{x,y\in\Sigma^*} \eta^{|xy|} p(xuy) = \sum_{w\in\Sigma^* u \Sigma^*} p(w) \sum_{x,y\in\Sigma^*} \eta^{|xy|} \mathbf{1}_{w=xuvy} \leq K_\eta p(\Sigma^* u \Sigma^*).$$

■

For $\eta \in [0,1]$, let $\widehat{\mathbf{Z}}_\eta(\xi)$ be the random matrix defined by

$$\widehat{\mathbf{Z}}_\eta[u,v] = \sum_{x,y\in\Sigma^*} \eta^{|xy|} \mathbf{1}_{\xi=xuvy} - \widehat{p}_\eta(uv) = \sum_{x,y\epsilon\Sigma^*} \eta^{|xy|} (\mathbf{1}_{\xi=xuvy} - p(xuvy))$$

for any $(u,v) \in U \times V$. Clearly, $\mathbb{E}\,\widehat{\mathbf{Z}}_\eta = 0$. We show below that $\left\|\widehat{\mathbf{Z}}_\eta\right\|$ is bounded if $\eta < 1$.

The moments $S_{\widehat{p}_\eta}^{(k)}$ satisfy $S_{\widehat{p}_\eta}^{(k)} = I^\top (I_d - \eta\mathbf{M}_\Sigma)^{-1}(I_d - \mathbf{M}_\Sigma)^{-k}(I_d - \eta\mathbf{M}_\Sigma)^{-1} T$, $S_{\widehat{p}_0}^{(k)} = S_p^{(k)}$ and $S_{\widehat{p}_1}^{(k)} = S_p^{(k+2)}$.

Lemma 24

$$\left\|\widehat{\mathbf{Z}}_\eta\right\| \leq (1-\eta)^{-2} + S_{\widehat{p}_\eta}^{(1)}.$$

Proof Indeed, for all u,

$$\sum_{v\in V} |\widehat{\mathbf{Z}}_\eta[u,v]| \leq \sum_{v,x,y\in\Sigma^*} [\eta^{|xy|}\mathbf{1}_{\xi=xuvy} + \hat{p}_\eta(uv)] \leq \sum_{x,y\in\Sigma^*} \eta^{|xy|}\mathbf{1}_{\xi\epsilon x\Sigma^* y} + S_{\hat{p}_\eta}^{(1)} \leq (1-\eta)^{-2} + S_{\widehat{p}_\eta}^{(1)}.$$

Hence, $\left\|\widehat{\mathbf{Z}}_\eta\right\|_\infty \leq (1-\eta)^{-2} + S_{\hat{p}_\eta}^{(1)}$. Similarly, $\left\|\overline{\mathbf{Z}}_\eta\right\|_1 \leq (1-\eta)^{-2} + S_{\hat{p}_\eta}^{(1)}$, which completes the proof. ■

Lemma 25 *For any $u, u', v, v' \in \Sigma^*$, $|\mathbb{E}(\widehat{\mathbf{Z}}_\eta[u,v]\widehat{\mathbf{Z}}_\eta[u',v'])| \leq K_\eta \min\{\widehat{p}(uv), \widehat{p}(u'v')\}$.*

Proof This a corollary of Lemmas 12 and 23 with $X \coloneqq \sum_{x,y\in\Sigma^*} \eta^{|xy|}\mathbf{1}_{\xi=xuvy}$, $Y \coloneqq \sum_{x,y\in\Sigma^*} \eta^{|xy|}\mathbf{1}_{\xi=xu'v'y}$ and $M = K_\eta$. ■

Let $\widehat{\mathbf{X}}_\eta$ be the dilation of $\widehat{\mathbf{Z}}_\eta$. We can now propose bounds for $\mathbb{E}\,\widehat{\mathbf{X}}_\eta^2$ and $\mathbb{E}\,Tr(\widehat{\mathbf{X}}_\eta^2)$.

Lemma 26

$$\left\|\mathbb{E}\widehat{\mathbf{X}}_\eta^2\right\| \leq K_\eta S_{\widehat{p}_\eta}^{(2)} \text{ and } Tr(\mathbb{E}(\widehat{\mathbf{X}}_\eta^2)) \leq 2K_\eta S_{\widehat{p}_\eta}^{(2)}.$$

Proof Indeed,

$$||\mathbb{E}(\widehat{\mathbf{Z}}\widehat{\mathbf{Z}}^\top)||_\infty \leq \max_u \sum_{u',v} |\mathbb{E}(\widehat{\mathbf{Z}}_\eta[u,v]\widehat{\mathbf{Z}}_\eta[u',v])| \leq K_\eta \sum_{u',v} \widehat{p}(u'v) \leq K_\eta S_{\widehat{p}_\eta}^{(2)}.$$

We have also

$$||\mathbb{E}(\widehat{\mathbf{Z}}\widehat{\mathbf{Z}}^\top)||_1 \leq K_\eta S_{\widehat{p}_\eta}^{(2)} \text{ and therefore } ||\mathbb{E}(\widehat{\mathbf{Z}}\widehat{\mathbf{Z}}^\top)|| \leq K_\eta S_{\widehat{p}_\eta}^{(2)}.$$

Similar computations provide all inequalities. ■

5.5 The factor Hankel Matrix $H_{\widehat{p}}^{U,V}$ is not subexponential: proof

$\widehat{\mathbf{Z}}(\xi)$ can be not a subexponential random matrix. Let $\Sigma = \{a\}$ be a one-letter alphabet and let p be the rational stochastic language defined by $p(a^n) = 2^{-(n+1)}$. When $\xi = a^n$, $\widehat{\mathbf{H}}_\xi$ is the matrix defined by $\widehat{\mathbf{H}}_\xi[i,j] = n+1-(i+j)$ if $i+j \leq n$ and 0 otherwise. Let $\widehat{\mathbf{H}}_n \in \mathbb{R}^{n\times n}$ be the nonnegative symmetric matrix defined by

$$\widehat{\mathbf{H}}_n = \begin{pmatrix} n & n-1 & \cdots & \cdots & 1 \\ n-1 & n-2 & \cdots & 1 & 0 \\ \vdots & \vdots & \iddots & & \vdots \\ \vdots & 1 & & & \\ 1 & 0 & \cdots & & 0 \end{pmatrix}$$

It can easily be deduced from the definition of a subexponential random matrix that if $\widehat{\mathbf{Z}}(\xi)$ were subexponential then, there would exist constants $C, R > 0$ such that for every integer k,

$$\max_{n\geq 0} 2^{-n}\|\widehat{\mathbf{H}}_n^k\| \leq \|\sum_{n\geq 0} 2^{-n}\widehat{\mathbf{H}}_n^k\| \leq Ck!R^k.$$

Proposition 27 *$\widehat{\mathbf{Z}}(\xi)$ is not subexponential.*

We need the following Lemma.

Lemma 28 *Bo (2000) Let A be a $n \times n$ nonnegative symmetric matrix with positive row sums $d_1, \ldots, d_n$. Then, the spectral radius of A satisfies $\rho(A) \geq n^{-1/2}\sqrt{\sum_{i=1}^n d_i^2}$.*

Proof (of Proposition 27) Lemma 28 applied to the matrix $\widehat{\mathbf{H}}_n$ gives that $\rho(\widehat{\mathbf{H}}_n) = \Omega(n^2)$. Indeed, we have the row sums d_i satisfy $d_i \geq (n+1-i)^2/2$ and $\sum_{i=1}^n d_i^2 = \Theta(n^5)$. Hence, for every integer k, $\|\widehat{\mathbf{H}}_n^k\| \geq n^{2k}$. Now, taking $n = k$, we should have $2^{-k}k^{2k} \leq CR^k k! \leq CR^k k^k$ for every integer k, which is false. ■

6. Experiments

The theoretical bounds described in the previous Sections have been evaluated on the benchmark of PAutomaC (Verwer et al., 2012).

6.1 Presentation of the benchmark

The benchmark of PAutomaC provides samples of strings generated from probabilistic automata and designed to evaluate probabilistic automata learning. We have selected eleven problems from that benchmark, for which the sparsity of the Hankel matrices makes the use of standard SVD algorithms available from `NumPy` or `SciPy` possible. Table 1 provides some information about the selected problems.

- Target models are of different types: non deterministic probabilistic automata (PA), deterministic probabilistic automata (DPA) and hidden Markov models (HMM). Each target is a rational stochastic language. We display the size of the corresponding alphabet, its 2nd and 3rd moments and the spectral radius ρ of $\mathbf{M}_\Sigma$[3], for a minimal representation $\langle I, \mathbf{M}, T\rangle$ of p. We display the number of states of the target automaton and the rank of the corresponding Hankel matrix computed using `NumPy`: the true rank of the target lies between these two values. We also provide constants C_p and ρ_p satisfying $p(\Sigma^n) \leq C_p\rho_p^n$ for any integer n[4].

3. Since the matrices $\mathbf{M}_\Sigma$ corresponding to two minimal representations are similar, the spectral radius ρ only depends on the underlying rational series.
4. From Gelfand's formula, $\|\mathbf{M}_\Sigma^k\|^{1/k}$ converges to ρ when $k \to \infty$. For any k satisfying $\|\mathbf{M}_\Sigma^k\|^{1/k} < 1$, we can take $\rho_p = \|\mathbf{M}_\Sigma^k\|^{1/k}$ and $C_p = \max_{0\leq r<k} \min_{0\leq s\leq r} \|I^T\mathbf{M}^s\| \cdot \|\mathbf{M}^{r-s}T\|$. We have noted in practice that when k increases, ρ_p decreases to ρ while C_p increases very slowly. We have uniformly taken the values computed for $k = 100$.

- Each problem comprises a sample S of strings independently drawn from the target. We provide the cardinal of S, the maximal length and the average length of strings in S.
- The empirical Hankel matrices are built on the prefixes, suffixes or factors of elements of S. We provide their size computed as the product of the number of non null rows by the number of non null columns. Almost all their cells are null: we provide the sparsity ratio.

Table 1: The 11 selected problems. The size of the Hankel matrices matrices is expressed in billions, where 1 g stands for 10^9. Sparsity indicates the ratio of non zero entries in the matrix: for example, there are $5.3 \times 1.9\ 10^4$ non empty cells in $\mathbf{H}_S$ for pb 3.

Pautomac ID	**3**	**4**	**7**	**15**	**25**	**29**	**31**	**38**	**39**	**40**	**42**
Target											
Type	PA	PA	DPA	PA	HMM	PA	PA	HMM	PA	DPA	DPA
$\|\Sigma\|$	4	4	13	14	10	6	5	10	14	14	9
NbStates	25	12	12	26	40	36	12	14	6	65	6
Rank	25	10	12	26	28	36	12	13	6	65	6
$S_p^{(2)}$	8.23	6.25	6.52	13.40	10.65	6.35	6.97	8.09	8.82	9.74	7.39
$S_p^{(3)}$	57.84	31.06	29.61	160.92	93.34	38.11	43.53	65.87	90.81	111.84	62.11
$\rho(M_\Sigma)$	0.85	0.77	0.72	0.92	0.88	0.83	0.84	0.88	0.90	0.91	0.88
ρ_p	0.87	0.79	0.73	0.95	0.92	0.87	0.86	0.91	0.92	0.96	0.90
C_p	0.24	0.42	0.80	0.09	0.21	0.37	0.23	0.29	0.29	0.26	0.25
Sample											
$\|S\|$	20 **k**	100 **k**	20 **k**	20 **k**	20 **k**	20 **k**	20 **k**	20 **k**	20 **k**	20 **k**	20 **k**
Avg $\|w\|$	7.22	5.26	5.52	12.46	9.72	5.29	6.00	7.18	7.74	8.72	6.36
$\max\|w\|$	67	55	36	110	90	59	59	84	106	106	70
Hankel mat.											
$\|Pref\| \times \|Suff\|$	1.9g	0.6g	0.2g	28g	13g	0.5g	1.4g	8.0g	7.7g	15g	3.4g
H_S *sparsity*	5.3e-5	1.9e-4	2.1e-4	9.0e-6	1.5e-5	1.2e-4	6.1e-5	1.8e-5	1.9e-5	1.1e-5	3.3e-5
$\|Pref\| \times \|Fact\|$	11g	1.8g	0.7g	291g	99g	2.4g	7.6g	60g	76g	165g	25g
$\overline{H}_S$ *sparsity*	5.8e-5	1.9e-4	2.1e-4	1.0e-6	1.6e-5	1.2e-4	6.6e-5	1.9e-5	2.0e-5	1.2e-5	3.5e-5
$\|Fact\| \times \|Fact\|$	73g	6.4g	3g	3363g	797g	15.7g	44g	460g	761g	1925g	202g
$\widehat{H}_S$ *sparsity*	5.8e-5	2.0e-4	2.0e-4	1.0e-6	1.6e-5	1.2e-4	6.9e-5	2.0e-5	2.0e-5	1.2e-5	3.6e-5

Figure 1 shows the typical behavior of $S_{\overline{p}_\eta}^{(1)}$ and $S_{\widehat{p}_\eta}^{(1)}$, similar for all the problems.

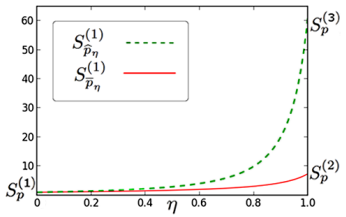


Figure 1: Behavior of $S_{\overline{p}_\eta}^{(1)}$ and $S_{\widehat{p}_\eta}^{(1)}$ for $\eta \in [0;1]$.

6.2 Accuracy of the bounds

For each problem, the exact value of $\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|_2$ is computed for sets U and V of the form $\Sigma^{\leq l}$, where we have maximized l according to our computing resources. This value is compared to the bounds provided by Theorem 7 and Equation (3), with $\delta = 0.05$ (Table 2). The optimized bound ("opt."), refers to the case where σ^2 has been calculated over $U \times V$ rather than $\Sigma^* \times \Sigma^*$ (see the remark at the end of Section 4.1). Tables 3 and 4 show analog comparisons for the prefix and the factor cases with different values of η. We can remark that our dimension-free bounds are significantly more accurate than the one provided by Equation (3). Similar results have been obtained for all the problems of PautomaC.

We have not reported experimental results based on Theorem 10, as for all the problems we consider, the constant σ is extremely large. For example, on Problem 3, with $\beta = 1 - \rho_p$, and using the parameters of Table 1, we have $\sigma \simeq 5308$, which would provide non significant accuracy values.

Table 2: Concentration bounds for $\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|$ where $U = V = \Sigma^{\leq l}$.

Problem number	3	4	7	15	25	29	31	38	39	40	42
l	8	9	8	5	5	9	7	4	6	4	7
$\|\mathbf{H}_S^{U,V} - \mathbf{H}_p^{U,V}\|$	0.005	0.003	0.006	0.004	0.003	0.005	0.005	0.006	0.005	0.004	0.005
Eq. (3)	0.100	0.039	0.088	0.127	0.115	0.088	0.092	0.097	0.103	0.105	0.095
Th. 7	0.067	0.026	0.060	0.085	0.076	0.059	0.062	0.066	0.069	0.073	0.063
Th. 7 (opt.)	0.048	0.023	0.053	0.028	0.032	0.047	0.044	0.028	0.033	0.024	0.038

Table 3: Concentration bounds for $\|\overline{\mathbf{H}}_S^{U,V} - \mathbf{H}_{\overline{p},\eta}^{U,V}\|$ (prefix case) where $U = V = \Sigma^{\leq l}$. The first part of the array is computed for $\eta = 1/2$, the second part for $\eta = 1$. The limiting case $\eta = 1$ (Th. 9 (opt.)) uses the remark at the end of Section 4.3

Problem number	3	4	7	15	25	29	31	38	39	40	42
l	8	9	8	5	5	9	7	4	6	4	7
$\|\overline{\mathbf{H}}_S^{U,V} - \mathbf{H}_{\overline{p},1/2}^{U,V}\|$	0.007	0.004	0.009	0.004	0.004	0.006	0.007	0.006	0.006	0.004	0.006
Eq. (3)	0.140	0.052	0.121	0.186	0.164	0.121	0.126	0.136	0.148	0.154	0.134
Th. 9	0.089	0.034	0.078	0.116	0.103	0.077	0.081	0.088	0.093	0.098	0.084
Th. 9 (opt.)	0.064	0.030	0.070	0.040	0.046	0.062	0.058	0.037	0.043	0.032	0.050
$\|\overline{\mathbf{H}}_S^{U,V} - \mathbf{H}_{\overline{p},1}^{U,V}\|$	0.014	0.006	0.022	0.012	0.015	0.012	0.018	0.013	0.014	0.009	0.013
Eq. (3)	0.281	0.089	0.200	0.476	0.361	0.229	0.241	0.291	0.355	0.387	0.293
Th. 9 (opt.)	0.128	0.052	0.117	0.106	0.106	0.118	0.110	0.078	0.102	0.0761	0.108

6.3 Implication for learning

The theoretical results of the last sections show that $\left\|\mathbf{H}_S^{U,V} - \mathbf{H}^{U,V}\right\|$, and similar expressions for other variants of the Hankel matrices, are bounded by a term that converges to 0 as the size of S increases, and is independent from U and V. This entails that the spectral learning

Table 4: Concentration bounds for $\|\widehat{\mathbf{H}}_S^{U,V} - \mathbf{H}_{\widehat{p},\eta}^{U,V}\|$ (factor case) where $U = V = \Sigma^{\leq l}$ and $\eta = 1/e$.

Problem number	3	4	7	15	25	29	31	38	39	40	42
l	6	7	5	4	4	6	6	4	4	4	5
$\|\widehat{\mathbf{H}}_S^{U,V} - \mathbf{H}_{\widehat{p},1/e}^{U,V}\|$	0.007	0.003	0.007	0.004	0.003	0.005	0.007	0.006	0.007	0.005	0.006
Eq. (3)	0.148	0.056	0.127	0.206	0.177	0.130	0.138	0.152	0.155	0.177	0.142
Th. 11	0.099	0.037	0.086	0.129	0.114	0.0845	0.090	0.098	0.103	0.109	0.093
Th. 11 (opt.)	0.060	0.030	0.062	0.036	0.041	0.056	0.059	0.0401	0.036	0.035	0.044

algorithm is consistent, whatever sets U and V are chosen, as soon as the rank of $\mathbf{H}^{U,V}$ is equal to the rank of the target, and even if we set $U = V = \Sigma^*$. But these concentration bounds give no indication of what should be done in pratical cases.

The spectral learning algorithm first computes the r-first right singular vectors $R_S^{U,V}$ of $\mathbf{H}_S^{U,V}$ and then build a linear representation from $R_S^{U,V}$. Since an exact linear representation of the target can be computed from the r-first right singular vectors $R^{U,V}$ of $\mathbf{H}^{U,V}$, where r is the rank of the target, the distance between the linear spaces spanned by $R_S^{U,V}$ and $R^{U,V}$ seems to be a relevant measure to evaluate the impact on learning of the choice of U and V.

There are several ways to measure the distance between two linear spaces. Most of them are based on the principal angles $\theta_1 \geq \theta_2 \geq \ldots \geq \theta_r$ between them. The largest principal angle θ_1 is a harsh measure since, even if the two spaces coincide along the last $r-1$ principal angles, the distance between the two spaces can be large. We have considered the following measure

$$d(span(R^{U,V}), span(R_S^{U,V})) = 1 - \frac{1}{r}\sum_{i=1}^{r} \cos\theta_i \tag{10}$$

which is equal to 0 if the spaces coincide and 1 if they are completely orthogonal, and which takes into account all the principal angles.

The tables 5 to 9 show the distance between $span(R^{U,V})$ and $span(R_S^{U,V})$ for p, $\overline{p}_{1/2}$, $\overline{p}$, $\widehat{p}_{1/2}$ and $\widehat{p}$, and for matrices having 100 columns and a variable number of rows, from 100 to 20,000 (i.e. $|V| = 100$ and $100 \leq |U| \leq 20,000$).

These tables show that

- the distance between the empirical and true singular vector spaces is smaller for the factor variant than for the prefix variant, and smaller for the prefix variant than for the classical Hankel matrices
- for both the prefix and factor variants, the distance is smaller for $\eta = 1$ than for $\eta = 1/2$ (and $\eta = 0$)
- in most cases, the distance computed for $|U| = 20,000$ is either minimal or not very far from the minimum.

We have run similar experiences for increasing values of $|V|$, from 100 to 20,000. The tables are very similar but the distances systematically increase with V. Table 10 shows the

Table 5: Distance between the spaces spanned by the r first right singular vectors of $\mathbf{H}^{U,V}$ and $\mathbf{H}_S^{U,V}$ for $|V| = 100$ and $100 \leq |U| \leq 20,000$. Entries must be scaled by 10^{-1}.

	3	4	7	15	25	29	31	38	39	40	42
100	2.096	0.011	0.841	1.643	4.171	1.495	0.985	3.375	0.132	1.789	0.031
200	2.079	0.011	0.005	1.466	3.988	1.512	0.902	3.304	0.091	1.609	0.031
500	1.934	0.011	0.004	1.417	3.901	1.457	0.917	2.915	0.104	1.593	0.029
1000	1.883	0.010	0.004	1.421	3.530	1.363	0.908	2.578	0.108	1.501	0.029
2000	1.813	0.010	0.004	1.382	3.529	1.358	0.919	2.511	0.125	1.512	0.029
5000	1.766	0.010	0.004	1.423	3.442	1.134	0.940	2.380	0.172	1.485	0.029
10000	1.755	0.010	0.004	1.424	3.431	1.136	0.961	2.284	0.269	1.390	0.029
20000	1.739	0.011	0.004	1.401	3.257	1.283	0.986	2.051	0.728	1.457	0.029

Table 6: Distance between the spaces spanned by the r first right singular vectors of $\mathbf{H}_{\overline{p}_{0.5}}^{U,V}$ and $\overline{\mathbf{H}}_{0.5,S}^{U,V}$ for $|V| = 100$ and $100 \leq |U| \leq 20,000$. Entries must be scaled by 10^{-1}.

	3	4	7	15	25	29	31	38	39	40	42
100	1.880	0.010	0.535	1.683	4.211	1.342	0.304	2.931	0.077	1.782	0.015
200	1.717	0.009	0.004	1.631	3.928	1.281	0.251	2.867	0.049	1.730	0.015
500	1.646	0.009	0.004	1.500	3.847	1.161	0.284	2.590	0.047	1.578	0.013
1000	1.623	0.009	0.003	1.558	3.537	1.137	0.297	2.443	0.047	1.529	0.013
2000	1.549	0.009	0.003	1.504	3.514	1.107	0.329	2.391	0.047	1.524	0.013
5000	1.491	0.009	0.003	1.501	3.317	1.098	0.439	2.129	0.048	1.486	0.013
10000	1.459	0.009	0.003	1.495	3.284	1.059	0.783	1.785	0.049	1.450	0.013
20000	1.425	0.009	0.003	1.548	3.203	0.950	0.955	1.690	0.052	1.424	0.013

Table 7: Distance between the spaces spanned by the r first right singular vectors of $\mathbf{H}_{\overline{p}}^{U,V}$ and $\overline{\mathbf{H}}_S^{U,V}$ for $|V| = 100$ and $100 \leq |U| \leq 20,000$. Entries must be scaled by 10^{-1}.

	3	4	7	15	25	29	31	38	39	40	42
100	1.281	0.010	0.554	1.729	4.095	1.221	0.224	2.778	0.009	1.662	0.005
200	1.185	0.008	0.007	1.472	4.016	1.237	0.204	2.732	0.006	1.497	0.005
500	1.074	0.007	0.006	1.388	3.886	1.194	0.238	2.619	0.006	1.352	0.005
1000	1.060	0.007	0.006	1.358	3.686	1.227	0.255	2.391	0.006	1.313	0.005
2000	1.054	0.007	0.006	1.179	3.682	1.174	0.288	2.349	0.006	1.341	0.005
5000	1.063	0.007	0.006	1.169	3.347	1.164	0.313	2.263	0.006	1.289	0.005
10000	1.073	0.007	0.006	1.181	3.244	1.165	0.332	2.156	0.006	1.347	0.005
20000	1.088	0.007	0.006	1.213	3.100	1.165	0.357	1.825	0.006	1.356	0.005

Table 8: Distance between the spaces spanned by the r first right singular vectors of $\mathbf{H}_{\widehat{p}_{0.5}}^{U,V}$ and $\widehat{\mathbf{H}}_{0.5,S}^{U,V}$ for $|V| = 100$ and $100 \leq |U| \leq 20,000$. Entries must be scaled by 10^{-1}.

	3	4	7	15	25	29	31	38	39	40	42
100	1.917	0.009	0.004	1.317	3.507	1.229	0.273	2.814	0.082	1.780	0.013
200	1.832	0.008	0.005	1.328	3.445	1.234	0.266	2.740	0.072	1.683	0.013
500	1.796	0.008	0.004	1.344	3.410	1.231	0.284	2.515	0.069	1.600	0.009
1000	1.781	0.008	0.003	1.363	3.312	1.164	0.288	2.344	0.048	1.585	0.009
2000	1.738	0.008	0.003	1.324	3.281	1.221	0.311	2.319	0.048	1.596	0.009
5000	1.727	0.007	0.003	1.323	3.262	1.137	0.315	2.247	0.047	1.475	0.009
10000	1.718	0.007	0.003	1.321	3.156	1.094	0.319	2.164	0.047	1.480	0.009
20000	1.652	0.007	0.003	1.351	3.110	1.065	0.324	2.119	0.047	1.452	0.009

Table 9: Distance between the spaces spanned by the r first right singular vectors of $\mathbf{H}_{\widehat{p}}^{U,V}$ and $\widehat{\mathbf{H}}_S^{U,V}$ for $|V| = 100$ and $100 \leq |U| \leq 20,000$. Entries must be scaled by 10^{-1}.

	3	4	7	15	25	29	31	38	39	40	42
100	1.185	0.004	0.005	0.557	2.264	0.901	0.237	1.977	0.004	1.512	0.002
200	1.065	0.004	0.005	0.480	2.153	0.893	0.244	1.922	0.003	1.307	0.002
500	1.064	0.004	0.004	0.524	2.162	0.797	0.254	1.783	0.003	1.290	0.002
1000	1.048	0.004	0.004	0.529	2.098	0.784	0.260	1.613	0.002	1.294	0.002
2000	1.029	0.004	0.004	0.537	2.075	0.803	0.267	1.597	0.002	1.268	0.002
5000	1.011	0.004	0.004	0.570	2.085	0.822	0.280	1.560	0.002	1.278	0.002
10000	1.012	0.005	0.004	0.584	2.072	0.882	0.287	1.513	0.002	1.278	0.002
20000	1.011	0.005	0.004	0.620	2.072	0.914	0.297	1.499	0.002	1.279	0.002

Table 10: Distance between the spaces spanned by the r first right singular vectors of $\mathbf{H}^{U,V}$ and $\mathbf{H}_S^{U,V}$ for $100 \leq |U| \leq 20,000$ and $100 \leq |V| \leq 20,000$ for problem 3. Entries must be scaled by 10^{-1}.

	100	300	1,000	2,000	5,000	20,000
100	2.096	2.399	2.660	2.747	2.887	3.116
200	2.079	2.292	2.512	2.608	2.752	2.974
500	1.934	2.196	2.350	2.431	2.575	2.808
1000	1.883	2.134	2.306	2.401	2.543	2.767
2,000	1.813	2.052	2.239	2.331	2.475	2.711
5,000	1.766	1.981	2.183	2.274	2.410	2.664
10,000	1.755	1.924	2.132	2.221	2.349	2.615
20,000	1.739	1.876	2.077	2.159	2.276	2.543

results for $d(span(\mathbf{H}^{U,V}), span(\mathbf{H}_S^{U,V}))$ computed on problem 3 - the behavior is similar for all the problems and all other variants.

These experiments call for the following recommendations that remain to be confirmed by further theoretical studies.

- use the data to infer first the factor variant of p, rather than the prefix variant or p itself,
- use a small number of columns,
- use as many rows as available, unless a specific information on the domain indicate to bound their number.

7. Conclusion

We have provided dimension-free concentration inequalities for Hankel matrices in the context of spectral learning of rational stochastic languages. These bounds cover 3 cases, each one corresponding to a specific way to exploit the strings under observation, paying attention to the strings themselves, to their prefixes or to their factors. For the last two cases, we introduced parametrized variants which allow a trade-off between the rate of the concentration and the exploitation of the information contained in data.

Experiments on the PAutomaC benchmark show that our dimension-free bounds are quite tight (except the subexponential bound for the prefix variant) and significantly more accurate than the bounds provided by classically used dimension-dependent bounds.

A consequence of these dimension-free inequalities is that the spectral learning algorithm is consistent, even if the whole empirical Hankel matrix is used, suggesting that the choice of relevant sets of rows and columns is maybe not critical. However, they do not provide any indication on what should be done in pratical cases. Experiments indicate that the singular vector spaces computed from the empirical Hankel matrices converge more quickly to the true singular vector spaces for the factor variant of the Hankel matrix - which is consistent with the experiments in (Balle et al., 2014), a small number of columns and a large number of rows. It would be interesting to obtain concentration results which confirm these practice. Another research direction would be to link up the prefix and factor cases to concentration bounds for sum of random tensors and to generalize the results to the case where a fixed number ≥ 1 of factors is considered for each string.

Acknowledgments

This work has been carried out thanks to the support of the ARCHIMEDE Labex (ANR-11-LABX- 0033) and the A*MIDEX project (ANR-11-IDEX-0001-02) funded by the "Investissements d'Avenir" French government program managed by the ANR.

References

R. Ahlswede and A. Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.

A. Anandkumar, D.P. Foster, D. Hsu, S. Kakade, and Y.-K. Liu. A spectral algorithm for latent dirichlet allocation. In *Proceedings of NIPS*, pages 926–934, 2012a.

A. Anandkumar, D. Hsu, F. Huang, and S. Kakade. Learning mixtures of tree graphical models. In *Proceedings of NIPS*, pages 1061–1069, 2012b.

A. Anandkumar, D. Hsu, and S.M. Kakade. A method of moments for mixture models and hidden markov models. *Proceedings of COLT - Journal of Machine Learning Research - Proceedings Track*, 23:33.1–33.34, 2012c.

R. Bailly. *Méthodes spectrales pour l'inférence grammaticale probabiliste de langages stochastiques rationnels*. PhD thesis, Aix-Marseille Université, 2011.

R. Bailly, F. Denis, and L. Ralaivola. Grammatical inference as a principal component analysis problem. In *Proceedings of ICML*, page 5, 2009.

R. Bailly, A. Habrard, and F. Denis. A spectral approach for probabilistic grammatical inference on trees. In *Proceedings of ALT*, pages 74–88, 2010.

B. Balle and M. Mohri. Spectral learning of general weighted automata via constrained matrix completion. In *Proceedings of NIPS*, pages 2168–2176, 2012.

B. Balle, A. Quattoni, and X. Carreras. A spectral learning algorithm for finite state transducers. In *Proceedings of ECML/PKDD (1)*, pages 156–171, 2011.

B. Balle, A. Quattoni, and X. Carreras. Local loss optimization in operator models: A new insight into spectral learning. In *Proceedings of ICML*, 2012.

B. Balle, X. Carreras, F. M. Luque, and A. Quattoni. Spectral learning of weighted automata: A forward-backward perspective. To appear in Machine Learning, 2013.

B. Balle, W. L. Hamilton, and J. Pineau. Methods of moments for learning stochastic languages: Unified presentation and empirical comparison. In *Proceedings of ICML*, 2014.

J. Berstel and C. Reutenauer. *Rational series and their languages*. EATCS monographs on theoretical computer science. Springer-Verlag, Berlin, New York, 1988. ISBN 0-387-18626-3. URL `http://opac.inria.fr/record=b1086956`. Translation of: Les sries rationnelles et leurs langages.

Z. Bo. On the spectral radius of nonnegative matrices. *Australasian Journal of Combinatorics*, 22:301–306, 2000.

S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. H. Ungar. Spectral learning of Latent-Variable PCFGs. In *ACL (1)*, pages 223–231. The Association for Computer Linguistics, 2012. ISBN 978-1-937284-24-4.

M. Droste, W. Kuich, and H. Vogler, editors. *Handbook of Weighted Automata*. Springer, 2009.

D. Hsu, S.M. Kakade, and T. Zhang. A spectral algorithm for learning hidden markov models. In *Proceedings of COLT*, 2009.

D. Hsu, S. M. Kakade, and T. Zhang. Dimension-free tail inequalities for sums of random matrices. *ArXiv e-prints*, 2011.

D. Hsu, S. Kakade, and T. Zhang. Tail inequalities for sums of random matrices that depend on the intrinsic dimension. *Electron. Commun. Probab.*, 17:no. 14, 1–13, 2012.

F.M. Luque, A. Quattoni, B. Balle, and X. Carreras. Spectral learning for non-deterministic dependency parsing. In *Proceedings of EACL*, pages 409–419, 2012.

A.P. Parikh, L. Song, and E.P. Xing. A spectral algorithm for latent tree graphical models. In *Proceedings of ICML*, pages 1065–1072, 2011.

A. Salomaa and M. Soittola. *Automata-theoretic aspects of formal power series*. Texts and monographs in computer science. Springer, 1978. ISBN 978-0-387-90282-1.

S. Siddiqi, B. Boots, and G.J. Gordon. Reduced-rank hidden Markov models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS-2010)*, 2010.

L. Song, B. Boots, S.M. Siddiqi, G.J. Gordon, and A.J. Smola. Hilbert space embeddings of hidden markov models. In *Proceedings of ICML*, pages 991–998, 2010.

E. M. Stein and R. Shakarchi. *Real analysis : measure theory, integration, and Hilbert spaces*. Princeton lectures in analysis. Princeton University press, Princeton (N.J.), Oxford, 2005. ISBN 0-691-11386-6. URL `http://opac.inria.fr/record=b1133853`.

G. W. Stewart. Perturbation theory for the singular value decomposition. In *SVD and Signal Processing II: Algorithms, Analysis and Applications*, pages 99–109. Elsevier, 1990.

J.A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

R. Vershynin. *Compressed Sensing*, chapter 5. Introduction to the non-asymptotic analysis of random matrices, pages 210–268. Cambridge University Press, 2012.

S. Verwer, R. Eyraud, and C. de la Higuera. Results of the PAutomaC probabilistic automaton learning competition. *Journal of Machine Learning Research - Proceedings Track*, 21: 243–248, 2012.

8. Appendix

8.1 Proof of Proposition 1

On a one-letter alphabet, for any non negative rational convergent series r, the series $u \mapsto r(\Sigma^* u \Sigma^*)$ is rational. Indeed, $r(\Sigma^* u \Sigma^*) = r(u\Sigma^*) = \overline{r}(u)$ and $\overline{r}$ is rational. On the other hand, this property may be false as soon as the alphabet contains at least two letters.

Proposition 1 *There exists a rational stochastic language p of rank 1 and built over a two-letter alphabet such that the series $u \mapsto p(\Sigma^* u \Sigma^*)$ is not rational.*

Proof Let $\Sigma = \{a, b\}$ and p be the rational stochastic language defined by $p(u) \coloneqq \alpha^{|u|_a}\beta^{|u|_b}\gamma$ where $\alpha, \beta, \gamma > 0$, $\alpha + \beta + \gamma = 1$ and where $|u|_x$ denotes the number of occurrences of the letter $x \in \Sigma$ in u. We have

$$p(\Sigma^*) = \sum_{u \in \Sigma^*} \alpha^{|u|_a}\beta^{|u|_b}\gamma = \gamma \sum_{n=0}^{\infty} \sum_{m=0}^{n} \binom{n}{m} \alpha^m \beta^{n-m} = \gamma \sum_{n=0}^{\infty} (\alpha+\beta)^n = \frac{\gamma}{1-\alpha-\beta} = 1.$$

Let f be the series defined by $f(u) \coloneqq p(\Sigma^* u \Sigma^*)$. Let us compute $f(a^n)$ for any integer n. Clearly, $f(\varepsilon) = 1$. Let $n \geq 1$. We can write

$$\Sigma^* = \bigcup_{m=0}^{n-1} \{a^m\} \cup a^n\Sigma^* \cup \bigcup_{m=0}^{n-1} a^m b \Sigma^*$$

and

$$\begin{aligned} f(a^n) &= p(a^n\Sigma^*) + \sum_{m=0}^{n-1} p(a^m b \Sigma^* a^n \Sigma^*) \\ &= \alpha^n + \sum_{m=0}^{n-1} \alpha^m \beta p(\Sigma^* a^n \Sigma^*) \\ &= \alpha^n + \frac{1-\alpha^n}{1-\alpha}\beta f(a^n) \end{aligned}$$

and therefore,

$$f(a^n) = (1-\alpha)\frac{\alpha^n}{\gamma + \beta\alpha^n}.$$

Suppose that f is rational. Then, every submatrix of the Hankel matrix of f is of finite rank. In particular, there exists an index k and real coefficients $\lambda_0, \lambda_1, \ldots, \lambda_{k-1}$ such that for any integer p,

$$f(a^{k+p}) = \sum_{i=0}^{k-1} \lambda_i f(a^{i+p})$$

which is equivalent to

$$\sum_{i=0}^{k-1} \lambda_i \alpha^{i-k} \frac{\gamma + \beta\alpha^{k+p}}{\gamma + \beta\alpha^{i+p}} = 1.$$

Let $g(z)$ be the complex function defined by

$$g(z) \coloneqq \left(\sum_{i=0}^{k-1} \mu_i \frac{\delta + \alpha^k z}{\delta + \alpha^i z}\right) - 1$$

where $\delta = \gamma/\beta$ and $\mu_i = \lambda_i \alpha^{i-k}$ for $0 \leq i \leq k-1$.

The function g as poles at $-\delta\alpha^{-i}$ and hence, is analytic on a neighborhood V of 0. Since $g(\alpha^p) = 0$ for any integer p, the principle of permanence shows that g is uniformly equal to 0 on V, i.e.

$$\sum_{i=0}^{k-1} \mu_i (\delta + \alpha^i z)^{-1} = (\delta + \alpha^k z)^{-1}, \forall z \in V.$$

In particular, these two functions and all their derivatives are equal for $z = 0$: we obtain the system

$$\sum_{i=0}^{k-1} \mu_i \alpha^{ih} = \alpha^{kh} \text{ for every } h \geq 0. \tag{11}$$

The Vandermonde matrix

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \alpha & \alpha^2 & \dots & \alpha^k \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^k & \alpha^{2k} & \dots & \alpha^{k^2} \end{pmatrix}$$

has a non zero determinant since $\alpha^i \neq \alpha^j$ for $i \neq j$ and therefore, the unique solution of the system $\sum_{i=0}^{k} \mu_i \alpha^{ih} = 0$ for $0 \leq h \leq k$ is $\mu_0 = \mu_1 = \dots = \mu_k = 0$ and the system (11) has no solution. ■

8.2 Proof of Proposition 2

From the definition of $\mathbf{T}_s$, it can easily be shown that the mapping $s \mapsto \mathbf{T}_s$ is a morphism: $\mathbf{T}_{s_1}\mathbf{T}_{s_2}[u,v] = \sum_{w\in\Sigma^*} \mathbf{T}_{s_1}[u,w]\mathbf{T}_{s_2}[w,v] = 1$ iff $v = us_1s_2$ and 0 otherwise.
If $\mathbf{X}$ is a matrix whose rows are indexed by Σ^*, we have $\mathbf{T}_s\mathbf{X}[u,v] = \sum_w \mathbf{T}_s[u,w]\mathbf{X}[w,v] = \mathbf{X}[us,v]$: i.e. the rows of $\mathbf{T}_S\mathbf{X}$ are included in the set of rows of $\mathbf{X}$. Then, it follows from the definition of E that $E^\top\mathbf{T}_s$ is equal to the first row of $\mathbf{T}_s$ (indexed by ϵ) with all coordinates equal to zero except the one indexed by s which equal 1.

Now, from the reduced singular value decomposition of $\mathbf{H} = \mathbf{L}\mathbf{D}\mathbf{R}^\top$ at rank d, $\mathbf{R}$ is a matrix of dimension $\infty \times d$ whose columns form a set of orthonormal vectors - the right singular vectors of $\mathbf{H}$ - such that $\mathbf{R}^\top\mathbf{R} = \mathbf{I}_d$ and $\mathbf{R}\mathbf{R}^\top\mathbf{H}^\top = \mathbf{H}^\top$ ($\mathbf{R}\mathbf{R}^\top$ is the orthogonal projection on the subspace spanned by the rows of $\mathbf{H}$).
One can easily deduce, by a recurrence over n, that for every string $u = x_1 \dots x_n$,

$$(\mathbf{R}^\top\mathbf{T}_{x_1}\mathbf{R}) \circ \dots \circ (\mathbf{R}^\top\mathbf{T}_{x_n}\mathbf{R})\mathbf{R}^\top\mathbf{H}^\top = \mathbf{R}^\top\mathbf{T}_u\mathbf{H}^\top.$$

Indeed, the inequality is trivially true for $n = 0$ since $\mathbf{T}_\epsilon = \mathbf{I}_d$. Then, we have that $\mathbf{R}^\top\mathbf{T}_x\mathbf{R}\mathbf{R}^\top\mathbf{T}_u\mathbf{H}^\top = \mathbf{R}^\top\mathbf{T}_x\mathbf{T}_u\mathbf{H}^\top = \mathbf{R}^\top\mathbf{T}_{xu}\mathbf{H}^\top$ since the columns of $\mathbf{T}_u\mathbf{H}^\top$ are rows of $\mathbf{H}$ and $\mathbf{T}$ is a morphism.

If $P^\top$ is the first row of $\mathbf{H}$ then: $E^\top\mathbf{R}(\mathbf{R}^\top\mathbf{T}_{x_1}\mathbf{R}) \circ \dots \circ (\mathbf{R}^\top\mathbf{T}_{x_n}\mathbf{R})\mathbf{R}^\top P = E^\top\mathbf{T}_u P = r(u)$. Thus, $\langle \mathbf{R}^\top E, (\mathbf{R}^\top\mathbf{T}_x\mathbf{R})_{x\in\Sigma}, \mathbf{R}^\top P\rangle$ is a linear representation of r of dimension d.

8.3 Monotonicity in Stewart formula.

Let us first recall the min – max characterization of singular values derived from the Courant-Fischer Theorem: for any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$,

$$\sigma_k(\mathbf{A}) = \min_{u_1,..,u_{k-1}\in\mathbb{R}^n} \max_{x\in\mathbb{R}^n, \|x\|=1, x\in[u_1,..,u_{k-1}]^\perp} \|\mathbf{A}x\|.$$

Let $\mathbf{B}$ be the result obtained by replacing all elements in the last column of $\mathbf{A}$ with 0 and let $u_k \in \mathbb{R}^n$ be defined by $u_k[i] = \mathbf{1}_{i=n}$.

$$\begin{aligned}\sigma_k(\mathbf{A}) &\geq \min_{u_1,..,u_{k-1}\in\mathbb{R}^n} \max_{x\in\mathbb{R}^n, \|x\|=1, x\in[u_1,..,u_k]^\perp} \|\mathbf{A}x\| \\ &= \min_{u_1,..,u_{k-1}\in\mathbb{R}^n} \max_{x\in\mathbb{R}^n, \|x\|=1, x\in[u_1,..,u_{k-1}]^\perp} \|\mathbf{B}x\| = \sigma_k(\mathbf{B}).\end{aligned}$$

A similar argument holds if we delete a row of $\mathbf{A}$. Then, it can easily be shown by induction that if $\mathbf{B}$ is obtained by deleting some rows and columns in $\mathbf{A}$, then $\sigma_k(\mathbf{A}) \geq \sigma_k(\mathbf{B})$ (as far as $\sigma_k(\mathbf{B})$ is defined).

Therefore, if $U \subseteq U'$ and $V \subseteq V'$, then $\left\|\mathbf{H}_S^{U\times V} - \mathbf{H}_r^{U\times V}\right\| \leq \left\|\mathbf{H}_S^{U'\times V'} - \mathbf{H}_r^{U'\times V'}\right\|$ and $\sigma_{\min}(\mathbf{H}_r^{U\times V}) \leq \sigma_{\min}(\mathbf{H}_r^{U'\times V'})$.

Journal of Machine Learning Research 17 (2016) 1-102 Submitted 12/14; Revised 12/15; Published 4/16

Distinguishing Cause from Effect Using Observational Data: Methods and Benchmarks

Joris M. Mooij[*] J.M.MOOIJ@UVA.NL
Institute for Informatics, University of Amsterdam
Postbox 94323, 1090 GH Amsterdam, The Netherlands

Jonas Peters JONAS.PETERS@TUEBINGEN.MPG.DE
Max Planck Institute for Intelligent Systems
Spemannstraße 38, 72076 Tübingen, Germany

Dominik Janzing JANZING@TUEBINGEN.MPG.DE
Max Planck Institute for Intelligent Systems
Spemannstraße 38, 72076 Tübingen, Germany

Jakob Zscheischler JAKOB.ZSCHEISCHLER@ENV.ETHZ.CH
Institute for Atmospheric and Climate Science, ETH Zürich
Universitätstrasse 16, 8092 Zürich, Switzerland

Bernhard Schölkopf BS@TUEBINGEN.MPG.DE
Max Planck Institute for Intelligent Systems
Spemannstraße 38, 72076 Tübingen, Germany

Editor: Isabelle Guyon and Alexander Statnikov

Abstract

The discovery of causal relationships from purely observational data is a fundamental problem in science. The most elementary form of such a causal discovery problem is to decide whether X causes Y or, alternatively, Y causes X, given joint observations of two variables X, Y. An example is to decide whether altitude causes temperature, or vice versa, given only joint measurements of both variables. Even under the simplifying assumptions of no confounding, no feedback loops, and no selection bias, such bivariate causal discovery problems are challenging. Nevertheless, several approaches for addressing those problems have been proposed in recent years. We review two families of such methods: methods based on Additive Noise Models (ANMs) and Information Geometric Causal Inference (IGCI). We present the benchmark CauseEffectPairs that consists of data for 100 different cause-effect pairs selected from 37 data sets from various domains (e.g., meteorology, biology, medicine, engineering, economy, etc.) and motivate our decisions regarding the "ground truth" causal directions of all pairs. We evaluate the performance of several bivariate causal discovery methods on these real-world benchmark data and in addition on artificially simulated data. Our empirical results on real-world data indicate that certain methods are indeed able to distinguish cause from effect using only purely observational data, although more benchmark data would be needed to obtain statistically significant conclusions. One of the best performing methods overall is the method based on Additive Noise Models that has originally been proposed by Hoyer et al. (2009), which obtains an accuracy of 63 ± 10 % and an AUC of 0.74 ± 0.05 on the real-world benchmark. As the main theoretical contribution of this work we prove the consistency of that method.

Keywords: Causal discovery, additive noise, information-geometric causal inference, cause-effect pairs, benchmarks

∗. Part of this work was done while JMM and JZ were with the MPI Tübingen, and JP with ETH Zürich.

1. Introduction

An advantage of having knowledge about causal relationships rather than statistical associations is that the former enables prediction of the effects of actions that perturb the observed system. Knowledge of cause and effect can also have implications on the applicability of semi-supervised learning and covariate shift adaptation (Schölkopf et al., 2012). While the gold standard for identifying causal relationships is controlled experimentation, in many cases, the required experiments are too expensive, unethical, or technically impossible to perform. The development of methods to identify causal relationships from purely observational data therefore constitutes an important field of research.

An observed statistical dependence between two variables X, Y can be explained by a causal influence from X to Y, a causal influence from Y to X, a possibly unobserved common cause that influences both X and Y ("confounding", see e.g., Pearl, 2000), a possibly unobserved common effect of X and Y that is conditioned upon in data acquisition ("selection bias", see e.g., Pearl, 2000), or combinations of these. Most state-of-the-art causal discovery algorithms that attempt to distinguish these cases based on observational data require that X and Y are part of a larger set of observed random variables influencing each other. For example, in that case, and under a genericity condition called "faithfulness", conditional independences between subsets of observed variables allow one to draw partial conclusions regarding their causal relationships (Spirtes et al., 2000; Pearl, 2000; Richardson and Spirtes, 2002; Zhang, 2008).

In this article, we focus on the *bivariate* case, assuming that only two variables, say X and Y, have been observed. We simplify the causal discovery problem considerably by assuming no confounding, no selection bias and no feedback. We study how to distinguish X causing Y from Y causing X using only purely observational data, i.e., a finite i.i.d. sample drawn from the joint distribution $\mathbb{P}_{X,Y}$.[1] As an example, consider the data visualized in Figure 1. The question is: does X cause Y, or does Y cause X? The true answer is "X causes Y", as here X is the altitude of weather stations and Y is the mean temperature measured at these weather stations (both in arbitrary units). In the absence of knowledge about the measurement procedures that the variables correspond with, one can try to exploit the subtle statistical patterns in the data in order to find the causal direction. This challenge of distinguishing cause from effect using only observational data has attracted increasing interest recently (Mooij and Janzing, 2010; Guyon et al., 2010, 2016). Approaches to causal discovery based on conditional independences do not work here, as X and Y are typically dependent, and there are no other observed variables to condition on.

1. We denote probability distributions by $\mathbb{P}$ and probability densities (typically with respect to Lebesgue measure on $\mathbb{R}^d$) by p.

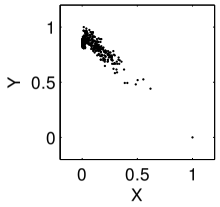


Figure 1: Example of a bivariate causal discovery task: decide whether X causes Y, or Y causes X, using only the observed data (visualized here as a scatter plot).

A variety of causal discovery methods has been proposed in recent years (Friedman and Nachman, 2000; Kano and Shimizu, 2003; Shimizu et al., 2006; Sun et al., 2006, 2008; Hoyer et al., 2009; Mooij et al., 2009; Zhang and Hyvärinen, 2009; Janzing et al., 2010; Mooij et al., 2010; Daniušis et al., 2010; Mooij et al., 2011; Shimizu et al., 2011; Janzing et al., 2012; Hyvärinen and Smith, 2013; Peters and Bühlmann, 2014; Kpotufe et al., 2014; Nowzohour and Bühlmann, 2015; Sgouritsa et al., 2015) that were claimed to be able to solve this task under certain assumptions. All these approaches exploit the *complexity* of the marginal and conditional probability distributions, in one way or the other. On an intuitive level, the idea is that the factorization of the joint density $p_{C,E}(c,e)$ of cause C and effect E into $p_C(c)p_{E\,|\,C}(e\,|\,c)$ typically yields models of lower total complexity than the alternative factorization into $p_E(e)p_{C\,|\,E}(c\,|\,e)$. Although this idea is intuitively appealing, it is not clear how to define complexity. If "complexity" and "information" are measured by Kolmogorov complexity and algorithmic information, respectively (Janzing and Schölkopf, 2010; Lemeire and Janzing, 2013), one can show that the statement "p_C contains no information about $p_{E\,|\,C}$" implies that the sum of the complexities of p_C and $p_{E\,|\,C}$ cannot be greater than the sum of the complexities of p_E and $p_{C\,|\,E}$. Some approaches, instead, define certain classes of "simple" conditionals, e.g., Additive Noise Models (Hoyer et al., 2009) and second-order exponential models (Sun et al., 2006; Janzing et al., 2009), and infer X to be the cause of Y whenever $\mathbb{P}_{Y\,|\,X}$ is from this class (and $\mathbb{P}_{X\,|\,Y}$ is not). Another approach that employs complexity in a more implicit way postulates that $\mathbb{P}_C$ contains no information about $\mathbb{P}_{E\,|\,C}$ (Janzing et al., 2012).

Despite the large number of methods for bivariate causal discovery that has been proposed over the last few years, their practical performance has not been studied very systematically, although some domain-specific studies have been performed (Smith et al., 2011; Statnikov et al., 2012). The present work attempts to address this by presenting benchmark data and reporting extensive empirical results on the performance of various bivariate causal discovery methods. Our main contributions are fourfold:

- We review two families of bivariate causal discovery methods, methods based on *Additive Noise Models (ANMs)* (originally proposed by Hoyer et al., 2009), and *Information Geometric Causal Inference (IGCI)* (originally proposed by Daniušis et al., 2010).

- We present a detailed description of the benchmark CauseEffectPairs that we collected over the years for the purpose of evaluating bivariate causal discovery methods. It currently consists of data for 100 different cause-effect pairs selected from 37 data sets from various domains (e.g., meteorology, biology, medicine, engineering, economy, etc.).

- We report the results of extensive empirical evaluations of the performance of several members of the ANM and IGCI families, both on artificially simulated data as well as on the CauseEffectPairs benchmark.

- We prove the consistency of the original implementation of ANM that was proposed by Hoyer et al. (2009).

The CauseEffectPairs benchmark data are provided on our website (Mooij et al., 2014). The synthetic benchmark data are provided as an online appendix for reproducibility purposes. In addition, all the code (including the code to run the experiments and create the figures) is provided both as an online appendix and on the first author's homepage[2] under an open source license to allow others to reproduce and build on our work.

The structure of this article is somewhat unconventional, as it partially consists of a review of existing methods, but it also contains new theoretical and empirical results. We will start in the next subsection by giving a more rigorous definition of the causal discovery task we consider in this article. In Section 2 we give a review of ANM, an approach based on the assumed additivity of the noise, and describe various ways of implementing this idea for bivariate causal discovery. In Appendix A we provide a proof for the consistency of the original ANM implementation that was proposed by Hoyer et al. (2009). In Section 3, we review IGCI, a method that exploits the independence of the distribution of the cause and the functional relationship between cause and effect. This method is designed for the deterministic (noise-free) case, but has been reported to work on noisy data as well. Section 4 gives more details on the experiments that we have performed, the results of which are reported in Section 5. Appendix D describes the CauseEffectPairs benchmark data set that we used for assessing the accuracy of various methods. We conclude in Section 6.

1.1 Problem Setting

In this subsection, we formulate the problem of interest central to this work. We tried to make this section as self-contained as possible and hope that it also appeals to readers who are not familiar with the terminology in the field of causality. For more details, we refer the reader to Pearl (2000).

Suppose that X, Y are two random variables with joint distribution $\mathbb{P}_{X,Y}$. This observational distribution corresponds to measurements of X and Y in an experiment in which X and Y are both (passively) observed. If an external intervention (i.e., from outside the system under consideration) changes some aspect of the system, then in general, this may lead to a change in the joint distribution of X and Y. In particular, we will consider a

2. http://www.jorismooij.nl/

perfect intervention[3] "do(x)" (or more explicitly: "do($X = x$)") that forces the variable X to have the value x, and leaves the rest of the system untouched. We denote the resulting interventional distribution of Y as $\mathbb{P}_{Y\,|\,\mathrm{do}(x)}$, a notation inspired by Pearl (2000). This interventional distribution corresponds to the distribution of Y in an experiment in which X has been set to the value x by the experimenter, after which Y is measured. Similarly, we may consider a perfect intervention do(y) that forces Y to have the value y, leading to the interventional distribution $\mathbb{P}_{X\,|\,\mathrm{do}(y)}$ of X.

For example, X and Y could be binary variables corresponding to whether the battery of a car is empty, and whether the start engine of the car is broken. Measuring these variables in many cars, we get an estimate of the joint distribution $\mathbb{P}_{X,Y}$. The marginal distribution $\mathbb{P}_X$, which only considers the distribution of X, can be obtained by integrating the joint distribution over Y. The conditional distribution $\mathbb{P}_{X\,|\,Y=0}$ corresponds with the distribution of X for the cars with a broken start engine (i.e., those cars for which we observe that $Y = 0$). The interventional distribution $\mathbb{P}_{X\,|\,\mathrm{do}(Y=0)}$, on the other hand, corresponds with the distribution of X after destroying the start engines of all cars (i.e., after actively setting $Y = 0$). Note that the distributions $\mathbb{P}_X, \mathbb{P}_{X\,|\,Y=0}, \mathbb{P}_{X\,|\,\mathrm{do}(Y=0)}$ may all be different.

In the absence of selection bias, we define:[4]

Definition 1 *We say that X **causes** Y if $\mathbb{P}_{Y\,|\,\mathrm{do}(x)} \neq \mathbb{P}_{Y\,|\,\mathrm{do}(x')}$ for some x, x'.*

Causal relations can be *cyclic*, i.e., X causes Y and Y also causes X. For example, an increase of the global temperature causes sea ice to melt, which causes the temperature to rise further (because ice reflects more sun light).

In the context of multiple variables $X_1, \ldots, X_p$ with $p \geq 2$, we define *direct* causation in the absence of selection bias as follows:

Definition 2 *X_i is a **direct cause** of X_j with respect to $X_1, \ldots, X_p$ if*

$$\mathbb{P}_{X_j\,|\,\mathrm{do}(X_i=x, \boldsymbol{X}_{\setminus ij}=\boldsymbol{c})} \neq \mathbb{P}_{X_j\,|\,\mathrm{do}(X_i=x', \boldsymbol{X}_{\setminus ij}=\boldsymbol{c})}$$

for some x, x' and some $\boldsymbol{c}$, where $\boldsymbol{X}_{\setminus ij} := X_{\{1,\ldots,p\}\setminus\{i,j\}}$ are all other variables besides X_i, X_j.

In words: X is a direct cause of Y with respect to a set of variables under consideration if Y depends on the value we force X to have in a perfect intervention, while fixing all other variables. The intuition is that a direct causal relation of X on Y is not mediated via the other variables. The more variables one considers, the harder it becomes experimentally to distinguish direct from indirect causation, as one has to keep more variables fixed.[5]

We may visualize direct causal relations in a *causal graph*:

3. Different types of "imperfect" interventions can be considered as well, see e.g., Eberhardt and Scheines (2007); Eaton and Murphy (2007); Mooij and Heskes (2013). In this paper we only consider perfect interventions.
4. In the presence of selection bias, one has to be careful when linking causal relations to interventional distributions. Indeed, if one would (incorrectly) apply Definition 1 to the conditional interventional distributions $\mathbb{P}_{Y\,|\,\mathrm{do}(X=x),S=s}$ instead of to the unconditional interventional distributions $\mathbb{P}_{Y\,|\,\mathrm{do}(X=x)}$ (e.g., because one is not aware of the fact that the data has been conditioned on S), one may obtain incorrect conclusions regarding causal relations.
5. For the special case $p = 2$ that is of interest in this work, we do not need to distinguish indirect from direct causality, as they are equivalent in that special case. However, we introduce this concept in order to define causal graphs on more than two variables, which we use to explain the concepts of confounding and selection bias.

Definition 3 *The **causal graph** $\mathcal{G}$ has variables $X_1, \ldots, X_p$ as nodes, and a directed edge from X_i to X_j if and only if X_i is a direct cause of X_j with respect to $X_1, \ldots, X_p$.*

Note that this definition allows for cyclic causal relations. In contrast with the typical assumption in the causal discovery literature, we do not assume here that the causal graph is necessarily a Directed Acyclic Graph (DAG).

If X causes Y, we generically have that $\mathbb{P}_{Y\,|\,\mathrm{do}(x)} \neq \mathbb{P}_Y$. Figure 2 illustrates how various causal relationships between X and Y (and at most one other variable) generically give rise to different (in)equalities between marginal, conditional, and interventional distributions involving X and Y. Note that the list of possibilities in Figure 2 is not exhaustive, as (i) feedback relationships with a latent variable were not considered; (ii) combinations of the cases shown are possible as well, e.g., (d) can be considered to be the combination of (a) and (b), and both (e) and (f) can be combined with all other cases; (iii) more than one latent variable could be present.

Returning to the example of the empty batteries (X) and broken start engines (Y), it seems reasonable to assume that these two variables are not causally related and case (c) in Figure 2 would apply, and therefore X and Y must be statistically independent.

In order to illustrate case (f), let us introduce a third binary variable, S, which measures whether the car starts or not. If the data acquisition is done by a car mechanic who only considers cars that do not start ($S = 0$), then we are in case (f): conditioning on the common effect S of X and Y leads to selection bias, i.e., X and Y are statistically dependent when conditioning on S (even though they are not directly causally related). Indeed, if we know that a car doesn't start, then learning that the battery is not empty makes it much more likely that the start engine is broken.

Another way in which two variables that are not directly causally related can still be statistically dependent is case (e), i.e., if they have a common cause. As an example, take for X the number of stork breeding pairs (per year) and for Y the number of human births (per year) in a country. Data has been collected for different countries and shows a significant correlation between X and Y (Matthews, 2000). Few people nowadays believe that storks deliver babies, or the other way around, and therefore it seems reasonable to assume that X and Y are not directly causally related. One obvious confounder (Z in Figure 2e) that may explain the observed dependence between X and Y is land area.

When data from all (observational and interventional) distributions are available, it becomes straightforward in principle to distinguish the six cases in Figure 2 simply by checking which (in)equalities in Figure 2 hold. In practice, however, we often only have data from the observational distribution $\mathbb{P}_{X,Y}$ (for example, because intervening on stork population or human birth rate is impractical). Can we then still infer the causal relationship between X and Y? If, under certain assumptions, we can decide upon the causal direction, we say that the causal direction is *identifiable* from the observational distribution (and our assumptions).

In this work, we will simplify matters considerably by considering only (a) and (b) in Figure 2 as possibilities. In other words, we assume that X and Y are dependent (i.e.,

6. Here, we assume that the intervention is performed *before* the conditioning. Since conditioning and intervening do not commute in general, one has to be careful when modeling causal processes in the presence of selection bias to take into account the actual ordering of these events.

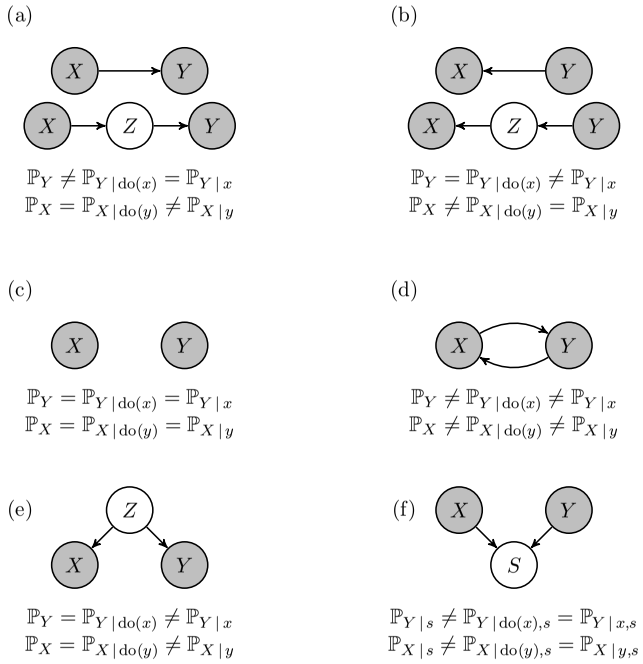


Figure 2: Several possible causal relationships between two observed variables X, Y and a single latent variable: (a) X causes Y; (b) Y causes X; (c) X, Y are not causally related; (d) feedback relationship, i.e., X causes Y and Y causes X; (e) a hidden confounder Z explains the observed dependence; (f) conditioning on a hidden selection variable S explains the observed dependence.[6] We used shorthand notation regarding quantifiers: equalities are generally valid, inequalities not necessarily. For example, "$\mathbb{P}_X = \mathbb{P}_{X\,|\,y}$" means "$\forall y : \mathbb{P}_X = \mathbb{P}_{X\,|\,y}$", whereas "$\mathbb{P}_X \neq \mathbb{P}_{X\,|\,y}$" means "$\exists y : \mathbb{P}_X \neq \mathbb{P}_{X\,|\,y}$". In all situations except (c), X and Y are (generically) dependent, i.e., $\mathbb{P}_{X,Y} \neq \mathbb{P}_X \mathbb{P}_Y$. The basic task we consider in this article is deciding between (a) and (b), using only data from $\mathbb{P}_{X,Y}$.

$\mathbb{P}_{X,Y} \neq \mathbb{P}_X \mathbb{P}_Y$), there is no confounding (common cause of X and Y), no selection bias (common effect of X and Y that is implicitly conditioned on), and no feedback between X and Y (a two-way causal relationship between X and Y). Inferring the causal direction between X and Y, i.e., deciding which of the two cases (a) and (b) holds, using *only the observational distribution* $\mathbb{P}_{X,Y}$ is the challenging task that we consider in this work.[7]

7. Note that this is a different question from the one often faced in problems in epidemiology, economics and other disciplines where causal considerations play an important role. There, the causal direction is often known *a priori*, i.e., one can exclude case (b), but the challenge is to distinguish case (a) from case (e) or

2. Additive Noise Models

In this section, we review a family of causal discovery methods that exploits *additivity* of the noise. We only consider the bivariate case here. More details and extensions to the multivariate case can be found in Hoyer et al. (2009); Peters et al. (2014).

2.1 Theory

There is an extensive body of literature on causal modeling and causal discovery that assumes that effects are linear functions of their causes plus independent, Gaussian noise. These models are known as *Structural Equation Models* (SEM) (Wright, 1921; Bollen, 1989) and are popular in econometrics, sociology, psychology and other fields. Although the assumptions of linearity and Gaussianity are mathematically convenient, they are not always realistic. More generally, one can define *Functional Models*, also known as *Structural Causal Models* (SCM) or *Non-Parametric Structural Equation Models* (NP-SEM), in which effects are modeled as (possibly nonlinear) functions of their causes and latent noise variables (Pearl, 2000).

2.1.1 Bivariate Structural Causal Models

In general, if $Y \in \mathbb{R}$ is a direct effect of a cause $X \in \mathbb{R}$ and m latent causes $\boldsymbol{U} = (U_1, \ldots, U_m) \in \mathbb{R}^m$, then it is intuitively reasonable to model this relationship as

$$\begin{cases} Y = f(X, U_1, \ldots, U_m), \\ X \perp\!\!\!\perp \boldsymbol{U}, \quad X \sim p_X(x), \quad \boldsymbol{U} \sim p_{\boldsymbol{U}}(u_1, \ldots, u_m), \end{cases} \tag{1}$$

where $f : \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}$ is a possibly nonlinear function (measurable with respect to the Borel sets of $\mathbb{R} \times \mathbb{R}^m$ and $\mathbb{R}$), and $p_X(x)$ and $p_{\boldsymbol{U}}(u_1, \ldots, u_m)$ are the joint densities of the observed cause X and latent causes $\boldsymbol{U}$ (with respect to Lebesgue measure on $\mathbb{R}$ and $\mathbb{R}^m$, respectively). The assumption that X and $\boldsymbol{U}$ are independent ("$X \perp\!\!\!\perp \boldsymbol{U}$") is justified by the assumption that there is no confounding, no selection bias, and no feedback between X and Y.[8] We will denote the observational distribution corresponding to (1) by $\mathbb{P}^{(1)}_{X,Y}$. By making use of the semantics of SCMs (Pearl, 2000), (1) also induces interventional distributions $\mathbb{P}^{(1)}_{X\,|\,\mathrm{do}(y)} = \mathbb{P}^{(1)}_X$ and $\mathbb{P}^{(1)}_{Y\,|\,\mathrm{do}(x)} = \mathbb{P}^{(1)}_{Y\,|\,x}$.

As the latent causes $\boldsymbol{U}$ are unobserved anyway, we can summarize their influence by a single "effective" *noise* variable $E_Y \in \mathbb{R}$ (also known as "disturbance term"):

$$\begin{cases} Y = f_Y(X, E_Y) \\ X \perp\!\!\!\perp E_Y, \quad X \sim p_X(x), \quad E_Y \sim p_{E_Y}(e_Y). \end{cases} \tag{2}$$

This simpler model can be constructed in such a way that it induces the same (observational and interventional) distributions as (1):

a combination of both. Even though our empirical results indicate that some methods for distinguishing case (a) from case (b) still perform reasonably well when their assumption of no confounding is violated by adding a latent confounder as in (e), we do not claim that these methods can be used to distinguish case (e) from case (a).

8. Another assumption that we have made here is that there is no *measurement noise*, i.e., noise added by the measurement apparatus. Measurement noise would mean that instead of measuring X itself, we observe a noisy version $\tilde{X}$, but Y is still a function of X, the (latent) variable X that is not corrupted by measurement noise.

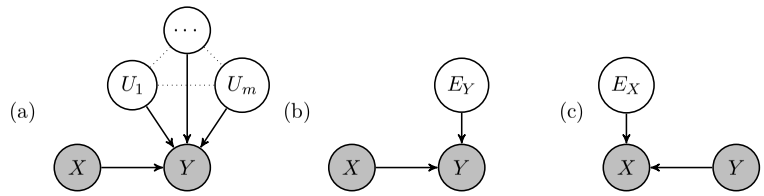


Figure 3: Causal graphs of Structural Causal Models (1), (2) and (3), respectively. (a) and (b) are interventionally equivalent, (b) and (c) are only observationally equivalent in general.

Proposition 4 *Given a model of the form (1) for which the observational distribution has a positive density with respect to Lebesgue measure, there exists a model of the form (2) that is* ***interventionally equivalent****, i.e., it induces the same observational distribution* $\mathbb{P}_{X,Y}$ *and the same interventional distributions* $\mathbb{P}_{X\,|\,\mathrm{do}(y)}$, $\mathbb{P}_{Y\,|\,\mathrm{do}(x)}$.

Proof Denote by $\mathbb{P}^{(1)}_{X,Y}$ the observational distribution induced by model (1). One possible way to construct E_Y and f_Y is to define the conditional cumulative density function $F_{Y\,|\,x}(y) := \mathbb{P}^{(1)}(Y \leq y\,|\,X = x)$ and its inverse with respect to y for fixed x, $F^{-1}_{Y\,|\,x}$. Then, one can define E_Y as the random variable

$$E_Y := F_{Y\,|\,X}(Y),$$

(where now the fixed value x is substituted with the random variable X) and the function f_Y by[9]

$$f_Y(x, e) := F^{-1}_{Y\,|\,x}(e).$$

Now consider the change-of-variables $(X, Y) \mapsto (X, E_Y)$. The corresponding joint densities transform as

$$p^{(1)}_{X,Y}(x, y) = p_{X,E_Y}\left(x, F_{Y\,|\,x}(y)\right)\left|\frac{\partial F_{Y\,|\,x}}{\partial y}(x, y)\right| = p_{X,E_Y}\left(x, F_{Y\,|\,x}(y)\right)p^{(1)}_{Y\,|\,X}(y\,|\,x),$$

and therefore

$$p^{(1)}_X(x) = p_{X,E_Y}\left(x, F_{Y\,|\,x}(y)\right)$$

for all x, y. This implies that $E_Y \perp\!\!\!\perp X$ and that $p_{E_Y} = \mathbf{1}_{(0,1)}$.

This establishes that $\mathbb{P}^{(2)}_{X,Y} = \mathbb{P}^{(1)}_{X,Y}$. The identity of the interventional distributions follows directly, because

$$\mathbb{P}^{(1)}_{X\,|\,\mathrm{do}(y)} = \mathbb{P}^{(1)}_X = \mathbb{P}^{(2)}_X = \mathbb{P}^{(2)}_{X\,|\,\mathrm{do}(y)}$$

and

$$\mathbb{P}^{(1)}_{Y\,|\,\mathrm{do}(x)} = \mathbb{P}^{(1)}_{Y\,|\,x} = \mathbb{P}^{(2)}_{Y\,|\,x} = \mathbb{P}^{(2)}_{Y\,|\,\mathrm{do}(x)}.$$

■

9. Note that we denote probability densities with the symbol p, so we can safely use the symbol f for a function without risking any confusion.

A similar construction of an effective noise variable can be performed in the other direction as well, at least to obtain a model that induces the same observational distribution. More precisely, we can construct a function f_X and a random variable E_X such that

$$\begin{cases} X = f_X(Y, E_X) \\ Y \perp\!\!\!\perp E_X, \quad Y \sim p_Y(y), \quad E_X \sim p_{E_X}(e_X) \end{cases} \tag{3}$$

induces the same observational distribution $\mathbb{P}^{(3)}_{X,Y} = \mathbb{P}^{(2)}_{X,Y}$ as (2) and the original (1). A well-known example is the linear-Gaussian case:

Example 1 *Let*

$$\begin{cases} Y = \alpha X + E_Y & X \sim \mathcal{N}(\mu_X, \sigma^2_X) \\ E_Y \perp\!\!\!\perp X & E_Y \sim \mathcal{N}(\mu_{E_Y}, \sigma^2_{E_Y}). \end{cases}$$

The model

$$\begin{cases} X = \beta Y + E_X & Y \sim \mathcal{N}(\mu_Y, \sigma^2_Y) \\ E_X \perp\!\!\!\perp Y & E_X \sim \mathcal{N}(\mu_{E_X}, \sigma^2_{E_X}) \end{cases}$$

with

$$\beta = \frac{\alpha\sigma^2_X}{\alpha^2\sigma^2_X + \sigma^2_{E_Y}},$$
$$\mu_Y = \alpha\mu_X + \mu_{E_Y}, \quad \sigma^2_Y = \alpha^2\sigma^2_X + \sigma^2_{E_Y},$$
$$\mu_{E_X} = (1 - \alpha\beta)\mu_X - \beta\mu_{E_Y}, \quad \sigma^2_{E_X} = (1 - \alpha\beta)^2\sigma^2_X + \beta^2\sigma^2_{E_Y}$$

induces the same joint distribution on X, Y.

However, in general the interventional distributions induced by (3) will be different from those of (2) and the original model (1). For example, in general

$$\mathbb{P}^{(3)}_{X\,|\,\mathrm{do}(y)} = \mathbb{P}^{(3)}_{X\,|\,y} = \mathbb{P}^{(2)}_{X\,|\,y} \neq \mathbb{P}^{(2)}_X = \mathbb{P}^{(2)}_{X\,|\,\mathrm{do}(y)}.$$

This means that whenever we can model an observational distribution $\mathbb{P}_{X,Y}$ with a model of the form (3), we can also model it using (2), and therefore the causal relationship between X and Y is not identifiable from the observational distribution without making additional assumptions. In other words: (1) and (2) are interventionally equivalent, but (2) and (3) are only observationally equivalent. Without having access to the interventional distributions, this symmetry prevents us from drawing any conclusions regarding the direction of the causal relationship between X and Y if we only have access to the observational distribution $\mathbb{P}_{X,Y}$.

2.1.2 Breaking the Symmetry

By *restricting* the models (2) and (3) to have lower complexity, asymmetries can be introduced. The work of Kano and Shimizu (2003); Shimizu et al. (2006) showed that for *linear* models (i.e., where the functions f_X and f_Y are restricted to be linear), *non-Gaussianity* of the input and noise distributions actually allows one to distinguish the directionality of such functional models. Peters and Bühlmann (2014) recently proved that for linear

models, Gaussian noise variables with *equal variances* also lead to identifiability. For high-dimensional variables, the structure of the covariance matrices can be exploited to achieve asymmetries (Janzing et al., 2010; Zscheischler et al., 2011).

Hoyer et al. (2009) showed that also *nonlinearity* of the functional relationships aids in identifying the causal direction, as long as the influence of the noise is additive. More precisely, they consider the following class of models:

Definition 5 *A tuple (p_X, p_{E_Y}, f_Y) consisting of a density p_X, a density p_{E_Y} with finite mean, and a Borel-measurable function $f_Y : \mathbb{R} \to \mathbb{R}$, defines a* ***bivariate Additive Noise Model (ANM)*** *$X \to Y$*

$$\begin{cases} Y = f_Y(X) + E_Y \\ X \perp\!\!\!\perp E_Y, \quad X \sim p_X, \quad E_Y \sim p_{E_Y}. \end{cases}$$

If the induced distribution $\mathbb{P}_{X,Y}$ has a density with respect to Lebesgue measure, the induced density $p(x, y)$ is said to satisfy an Additive Noise Model $X \to Y$.

Note that an ANM is a special case of model (2) where the influence of the noise on Y is restricted to be additive.

We are especially interested in cases for which the additivity requirement introduces an asymmetry between X and Y:

Definition 6 *If the joint density $p(x, y)$ satisfies an Additive Noise Model $X \to Y$, but does not satisfy any Additive Noise Model $Y \to X$, then we call the ANM $X \to Y$* ***identifiable*** *(from the observational distribution).*

Hoyer et al. (2009) proved that Additive Noise Models are generically identifiable. The intuition behind this result is that if $p(x, y)$ satisfies an Additive Noise Model $X \to Y$, then $p(y \,|\, x)$ depends on x only through its mean, and all other aspects of this conditional distribution do not depend on x. On the other hand, $p(x \,|\, y)$ will typically depend in a more complicated way on y (see also Figure 4). Only for very specific choices of the parameters of an ANM one obtains a non-identifiable ANM. We have already seen an example of such a non-identifiable ANM: the linear-Gaussian case (Example 1). A more exotic example with non-Gaussian distributions is described in Peters et al. (2014, Example 25). Zhang and Hyvärinen (2009) proved that non-identifiable ANMs necessarily fall into one out of five classes. In particular, their result implies something that we might expect intuitively: if f is not injective,[10] the ANM is identifiable.

Mooij et al. (2011) showed that bivariate identifiability even holds generically when feedback is allowed (i.e., if both $X \to Y$ *and* $Y \to X$), at least when assuming noise and input distributions to be Gaussian. Peters et al. (2011) provide an extension of the acyclic model for discrete variables. Zhang and Hyvärinen (2009) give an extension of the identifiability results allowing for an additional bijective[11] transformation of the data, i.e., using a functional model of the form $Y = \phi\big(f_Y(X) + E_Y\big)$, with $E_Y \perp\!\!\!\perp X$, and $\phi : \mathbb{R} \to \mathbb{R}$

10. A mapping is said to be *injective* if it does not map distinct elements of its domain to the same element of its codomain.
11. A mapping is said to be *surjective* if every element in its codomain is mapped to by at least one element of its domain. It is called *bijective* if it is surjective and injective.

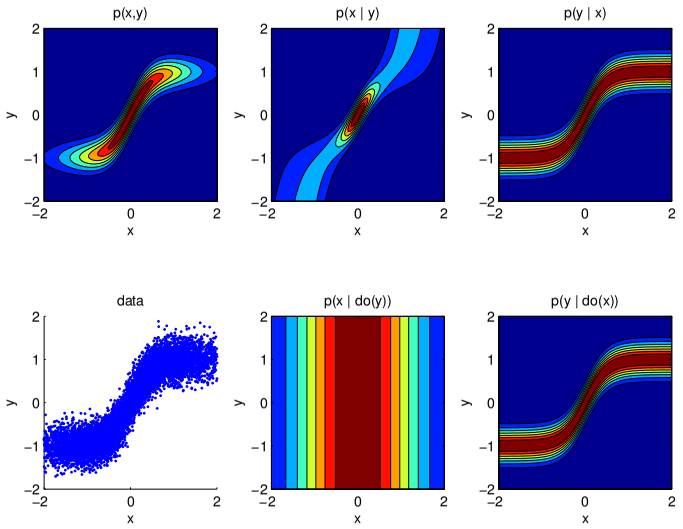


Figure 4: Identifiable ANM with $Y = \tanh(X) + E$, where $X \sim \mathcal{N}(0, 1)$ and $E \sim \mathcal{N}(0, 0.5^2)$. Shown are contours of the joint and conditional distributions, and a scatter plot of data sampled from the model distribution. Note that the contour lines of $p(y \,|\, x)$ only shift as x changes. On the other hand, $p(x \,|\, y)$ differs by more than just its mean for different values of y.

bijective, which they call the Post-NonLinear (PNL) model. The results on identifiability of Additive Noise Models can be extended to the multivariate case if there are no hidden variables and no feedback loops (Peters et al., 2014). This extension can be applied to nonlinear ANMs (Hoyer et al., 2009; Bühlmann et al., 2014), linear non-Gaussian models (Shimizu et al., 2011), the model of equal error variances (Peters and Bühlmann, 2014) or to the case of discrete variables (Peters et al., 2011). Full identifiability in the presence of hidden variables for the acyclic case has only been established for linear non-Gaussian models (Hoyer et al., 2008).

2.1.3 Additive Noise Principle

Following Hoyer et al. (2009), we hypothesize that:

Principle 1 *Suppose we are given a joint density $p(x, y)$ and we know that the causal structure is either that of (a) or (b) in Figure 2. If $p(x, y)$ satisfies an* identifiable *Additive Noise Model $X \to Y$, then it is likely that we are in case (a), i.e., X causes Y.*

This principle should not be regarded as a rigorous statement, but rather as an empirical assumption: we cannot exactly quantify *how likely* the conclusion that X causes Y is, as there is always a possibility that Y causes X while $p_{X,Y}$ happens to satisfy an identifiable

Additive Noise Model $X \to Y$. In general, that would require a special choice of the distribution of X and the conditional distribution of Y given X, which is unlikely. In this sense, we can regard this principle as a special case of Occam's Razor.

In the next subsection, we will discuss various ways of operationalizing this principle. In Section 4, we provide empirical evidence supporting this principle.

2.2 Estimation Methods

The following Lemma is helpful to test whether a density satisfies a bivariate Additive Noise Model:

Lemma 7 *Given a joint density $p(x,y)$ of two random variables X,Y such that the conditional expectation $\mathbb{E}(Y \mid X = x)$ is well-defined for all x and measurable. Then, $p(x,y)$ satisfies a bivariate Additive Noise Model $X \to Y$ if and only if $E_Y := Y - \mathbb{E}(Y \mid X)$ has finite mean and is independent of X.*

Proof Suppose that $p(x,y)$ is induced by (p_X, p_U, f), say $Y = f(X) + U$ with $X \perp\!\!\!\perp U$, $X \sim p_X$, $U \sim p_U$. Then $\mathbb{E}(Y \mid X = x) = f(x) + \nu$, with $\nu = \mathbb{E}(U)$. Therefore, $E_Y = Y - \mathbb{E}(Y \mid X) = Y - (f(X) + \nu) = U - \nu$ is independent of X. Conversely, if E_Y is independent of X, $p(x,y)$ is induced by the bivariate Additive Noise Model $(p_X, p_{E_Y}, x \mapsto \mathbb{E}(Y \mid X = x))$. ■

In practice, we usually do not have the density $p(x,y)$, but rather a finite sample of it. In that case, we can use the same idea for testing whether this sample comes from a density that satisfies an Additive Noise Model: we estimate the conditional expectation $\mathbb{E}(Y \mid X)$ by regression, and then test the independence of the residuals $Y - \mathbb{E}(Y \mid X)$ and X.

Suppose we have two data sets, a *training* data set $\mathcal{D}_N := \{(x_n, y_n)\}_{n=1}^N$ (for estimating the function) and a *test* data set $\mathcal{D}'_N := \{(x'_n, y'_n)\}_{n=1}^N$ (for testing independence of residuals), both consisting of i.i.d. samples distributed according to $p(x,y)$. We will write $\boldsymbol{x} = (x_1, \ldots, x_N)$, $\boldsymbol{y} = (y_1, \ldots, y_N)$, $\boldsymbol{x}' = (x'_1, \ldots, x'_N)$ and $\boldsymbol{y}' = (y'_1, \ldots, y'_N)$. We will consider two scenarios: the "data splitting" scenario where training and test set are independent (typically achieved by splitting a bigger data set into two parts), and the "data recycling" scenario in which the training and test data are identical (where we use the same data twice for different purposes: regression and independence testing).[12]

Hoyer et al. (2009) suggested the following procedure to test whether the data come from a density that satisfies an Additive Noise Model.[13] By regressing Y on X using the training data $\mathcal{D}_N$, an estimate $\hat{f}_Y$ for the regression function $x \mapsto \mathbb{E}(Y \mid X = x)$ is obtained. Then, an independence test is used to estimate whether the predicted residuals are independent of the input, i.e., whether $(Y - \hat{f}_Y(X)) \perp\!\!\!\perp X$, using test data $(\boldsymbol{x}', \boldsymbol{y}')$. If the null hypothesis of independence is not rejected, one concludes that $p(x,y)$ satisfies an Additive Noise Model $X \to Y$. The regression procedure and the independence test can be freely chosen.

There is a caveat, however: under the null hypothesis that $p(x,y)$ indeed satisfies an ANM, the error in the estimated residuals may introduce a dependence between the predicted residuals $\hat{\boldsymbol{e}}'_Y := \boldsymbol{y}' - \hat{f}_Y(\boldsymbol{x}')$ and $\boldsymbol{x}'$ even if the true residuals $\boldsymbol{y}' - \mathbb{E}(Y \mid X = \boldsymbol{x}')$ are

12. Kpotufe et al. (2014) refer to these scenarios as "decoupled estimation" and "coupled estimation", respectively.
13. They only considered the data recycling scenario, but the same idea can be applied to the data splitting scenario.

independent of $\boldsymbol{x}'$. Therefore, the threshold for the independence test statistic has to be chosen with care: the standard threshold that would ensure consistency of the independence test on its own may be too tight. As far as we know, there are no theoretical results on the choice of that threshold that would lead to a consistent way to test whether $p(x,y)$ satisfies an ANM $X \to Y$.

We circumvent this problem by assuming *a priori* that $p(x,y)$ either satisfies an ANM $X \to Y$, or an ANM $Y \to X$, but not both. In that sense, the test statistics of the independence test can be directly compared, and no threshold needs to be chosen. This leads us to Algorithm 1 as a general scheme for identifying the direction of the ANM. In order to decide whether $p(x,y)$ satisfies an Additive Noise Model $X \to Y$, or an Additive Noise Model $Y \to X$, we simply estimate the regression functions in both directions, calculate the corresponding residuals, estimate the dependence of the residuals with respect to the input by some dependence measure $\hat{C}$, and output the direction that has the lowest dependence.

In principle, any consistent regression method can be used in Algorithm 1. Likewise, in principle any consistent measure of dependence can be used in Algorithm 1 as score function. In the next subsections, we will consider in more detail some possible choices for the score function. Originally, Hoyer et al. (2009) proposed to use the p-value of the Hilbert Schmidt Independence Criterion (HSIC), a kernel-based non-parametric independence test. Alternatively, one can also use the HSIC statistic itself as a score, and we will show that this leads to a consistent procedure. Other dependence measures could be used instead, e.g., the measure proposed by Reshef et al. (2011). Kpotufe et al. (2014); Nowzohour and Bühlmann (2015) proposed to use as a score the sum of the estimated differential entropies of inputs and residuals and proved consistency of that procedure. For the Gaussian case, that is equivalent to the score considered in a high-dimensional context that was shown to be consistent by Bühlmann et al. (2014). This Gaussian score is also strongly related to an empirical-Bayes score originally proposed by Friedman and Nachman (2000). Finally, we will briefly discuss a Minimum Message Length score that was considered by Mooij et al. (2010) and another idea (based on minimizing a dependence measure directly) proposed by Mooij et al. (2009).

2.2.1 HSIC-based Scores

One possibility, first considered by Hoyer et al. (2009), is to use the Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005) for testing the independence of the estimated residuals with the inputs. See Appendix A.1 for a definition and basic properties of the HSIC independence test.

As proposed by Hoyer et al. (2009), one can use the p-value of the HSIC statistic under the null hypothesis of independence. This amounts to the following score function for measuring dependence:

$$\hat{C}(\boldsymbol{u}, \boldsymbol{v}) := -\log \hat{p}_{\mathrm{HSIC}_{\kappa_{\hat{\ell}(\boldsymbol{u})}, \kappa_{\hat{\ell}(\boldsymbol{v})}}}(\boldsymbol{u}, \boldsymbol{v}). \tag{4}$$

Here, κ_ℓ is a kernel with parameters ℓ, that are estimated from the data. $\boldsymbol{u}$ and $\boldsymbol{v}$ are either inputs or estimated residuals (see also Algorithm 1). A low HSIC p-value indicates that we should reject the null hypothesis of independence. Another possibility is to use the HSIC

Algorithm 1 General procedure to decide whether $p(x, y)$ satisfies an Additive Noise Model $X \to Y$ or $Y \to X$.

Input:

1. I.i.d. sample $\mathcal{D}_N := \{(x_i, y_i)\}_{i=1}^N$ of X and Y ("training data");
2. I.i.d. sample $\mathcal{D}'_N := \{(x'_i, y'_i)\}_{i=1}^N$ of X and Y ("test data");
3. Regression method;
4. Score estimator $\hat{C} : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$.

Output: $\hat{C}_{X\to Y}$, $\hat{C}_{Y\to X}$, `dir`.

1. Use the regression method to obtain estimates:
   (a) $\hat{f}_Y$ of the regression function $x \mapsto \mathbb{E}(Y \,|\, X = x)$,
   (b) $\hat{f}_X$ of the regression function $y \mapsto \mathbb{E}(X \,|\, Y = y)$
   using the training data $\mathcal{D}_N$;
2. Use the estimated regression functions to predict residuals:
   (a) $\hat{\boldsymbol{e}}'_Y := \boldsymbol{y}' - \hat{f}_Y(\boldsymbol{x}')$
   (b) $\hat{\boldsymbol{e}}'_X := \boldsymbol{x}' - \hat{f}_X(\boldsymbol{y}')$
   from the test data $\mathcal{D}'_N$.
3. Calculate the scores to measure dependence of inputs and estimated residuals on the test data $\mathcal{D}'_N$:
   (a) $\hat{C}_{X\to Y} := \hat{C}(\boldsymbol{x}', \hat{\boldsymbol{e}}'_Y)$;
   (b) $\hat{C}_{Y\to X} := \hat{C}(\boldsymbol{y}', \hat{\boldsymbol{e}}'_X)$;
4. Output $\hat{C}_{X\to Y}, \hat{C}_{Y\to X}$ and

$$\texttt{dir} := \begin{cases} X \to Y & \text{if } \hat{C}_{X\to Y} < \hat{C}_{Y\to X}, \\ Y \to X & \text{if } \hat{C}_{X\to Y} > \hat{C}_{Y\to X}, \\ ? & \text{if } \hat{C}_{X\to Y} = \hat{C}_{Y\to X}. \end{cases}$$

value itself (instead of its p-value):

$$\hat{C}(\boldsymbol{u}, \boldsymbol{v}) := \widehat{\mathrm{HSIC}}_{\kappa_{\hat{\ell}(\boldsymbol{u})}, \kappa_{\hat{\ell}(\boldsymbol{v})}}(\boldsymbol{u}, \boldsymbol{v}). \tag{5}$$

An even simpler option is to use a fixed kernel k:

$$\hat{C}(\boldsymbol{u}, \boldsymbol{v}) := \widehat{\mathrm{HSIC}}_{k,k}(\boldsymbol{u}, \boldsymbol{v}). \tag{6}$$

In Appendix A, we prove that under certain technical assumptions, Algorithm 1 with score function (6) is a consistent procedure for inferring the direction of the ANM. In particular, the product kernel $k \cdot k$ should be characteristic in order for HSIC to detect all possible independences, and the regression method should satisfy the following condition:

Definition 8 *Let X, Y be two real-valued random variables with joint distribution $\mathbb{P}_{X,Y}$. Suppose we are given sequences of training data sets $\mathcal{D}_N = \{X_1, X_2, \ldots, X_N\}$ and test data sets $\mathcal{D}'_N = \{X'_1, X'_2, \ldots, X'_N\}$ (in either the data splitting or the data recycling scenario). We call a regression method **suitable** for regressing Y on X if the mean squared error between true and estimated regression function, evaluated on the test data, vanishes asymptotically in expectation, i.e.,*

$$\lim_{N\to\infty} \mathbb{E}_{\mathcal{D}_N, \mathcal{D}'_N} \left(\frac{1}{N} \sum_{n=1}^{N} \left| \hat{f}_Y(X'_n; \mathcal{D}_N) - \mathbb{E}(Y \,|\, X = X'_n) \right|^2 \right) = 0. \tag{7}$$

Here, the expectation is taken over both training data $\mathcal{D}_N$ and test data $\mathcal{D}'_N$.

The consistency result then reads as follows:

Theorem 9 *Let X, Y be two real-valued random variables with joint distribution $\mathbb{P}_{X,Y}$ that either satisfies an Additive Noise Model $X \to Y$, or $Y \to X$, but not both. Suppose we are given sequences of training data sets $\mathcal{D}_N$ and test data sets $\mathcal{D}'_N$ (in either the data splitting or the data recycling scenario). Let $k : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be a bounded non-negative Lipschitz-continuous kernel such that the product $k \cdot k$ is characteristic. If the regression procedure used in Algorithm 1 is suitable for both $\mathbb{P}_{X,Y}$ and $\mathbb{P}_{Y,X}$, then Algorithm 1 with score (6) is a consistent procedure for estimating the direction of the Additive Noise Model.*

Proof See Appendix A (where a slightly more general result is shown, allowing for two different kernels k, l to be used). The main technical difficulty consists of the fact that the error in the estimated regression function introduces a dependency between the cause and the estimated residuals. We overcome this difficulty by showing that the dependence is so weak that its influence on the test statistic vanishes asymptotically. ■

In the data splitting case, weakly universally consistent regression methods (Györfi et al., 2002) are suitable. In the data recycling scenario, any regression method that satisfies (7) is suitable. An example of a kernel k that satisfies the conditions of Theorem 9 is the Gaussian kernel.

2.2.2 Entropy-based Scores

Instead of explicitly testing for independence of residuals and inputs, one can use the sum of their differential entropies as a score function (e.g., Kpotufe et al., 2014; Nowzohour and Bühlmann, 2015). This can easily be seen using Lemma 1 of Kpotufe et al. (2014), which we reproduce here because it is very instructive:

Lemma 10 *Consider a joint distribution of X, Y with density $p(x, y)$. For arbitrary functions $f, g : \mathbb{R} \to \mathbb{R}$ we have:*

$$H(X) + H\big(Y - f(X)\big) = H(Y) + H\big(X - g(Y)\big) - I\big(X - g(Y), Y\big) + I\big(Y - f(X), X\big),$$

where $H(\cdot)$ denotes differential Shannon entropy, and $I(\cdot,\cdot)$ denotes differential mutual information (Cover and Thomas, 2006). ■

The proof is a simple application of the chain rule of differential entropy. If $p(x,y)$ satisfies an identifiable Additive Noise Model $X \to Y$, then there exists a function f with $I(Y - f(X), X) = 0$ (e.g., the regression function $x \mapsto \mathbb{E}(Y \mid X = x)$), but $I(X - g(Y), Y) > 0$ for any function g. Therefore, one can use Algorithm 1 with score function

$$\hat{C}(\boldsymbol{u}, \boldsymbol{v}) := \hat{H}(\boldsymbol{u}) + \hat{H}(\boldsymbol{v}) \tag{8}$$

in order to estimate the causal direction, using any estimator $\hat{H}(\cdot)$ of the differential Shannon entropy. Kpotufe et al. (2014); Nowzohour and Bühlmann (2015) prove that this approach to estimating the direction of Additive Noise Models is consistent under certain technical assumptions.

Kpotufe et al. (2014) note that the advantage of score (8) (based on marginal entropies) over score (5) (based on dependence) is that marginal entropies are cheaper to estimate than dependence (or mutual information). This is certainly true when considering computation time. However, as we will see later, a disadvantage of relying on differential entropy estimators is that these can be quite sensitive to discretization effects.

2.2.3 Gaussian Score

The differential entropy of a random variable X can be upper bounded in terms of its variance (see e.g., Cover and Thomas, 2006, Theorem 8.6.6):

$$H(X) \leq \frac{1}{2}\log(2\pi e) + \frac{1}{2}\log \mathbb{V}\mathrm{ar}(X), \tag{9}$$

where identity holds in case X has a Gaussian distribution. Assuming that $p(x,y)$ satisfies an identifiable Gaussian Additive Noise Model $X \to Y$ with Gaussian input and Gaussian noise distributions, we therefore conclude from Lemma 10 that

$$\begin{aligned}\log \mathbb{V}\mathrm{ar}(X) + \log \mathbb{V}\mathrm{ar}(Y - \hat{f}(X)) &= 2H(X) + 2H(Y - \hat{f}(X)) - 2\log(2\pi e)\\ &< 2H(Y) + 2H(X - \hat{g}(Y)) - 2\log(2\pi e)\\ &\leq \log \mathbb{V}\mathrm{ar}Y + \log \mathbb{V}\mathrm{ar}(X - \hat{g}(Y))\end{aligned}$$

for any function g. In that case, we can therefore use Algorithm 1 with score function

$$\hat{C}(\boldsymbol{u}, \boldsymbol{v}) := \log \widehat{\mathbb{V}\mathrm{ar}}(\boldsymbol{u}) + \log \widehat{\mathbb{V}\mathrm{ar}}(\boldsymbol{v}). \tag{10}$$

This score was also considered recently by Bühlmann et al. (2014) and shown to lead to a consistent estimation procedure under certain assumptions.

2.2.4 Empirical-Bayes Scores

Deciding the direction of the ANM can also be done by applying model selection using empirical Bayes. As an example, for the ANM $X \to Y$, one can consider a generative model that models X as a Gaussian, and Y as a Gaussian Process (Rasmussen and Williams, 2006) conditional on X. For the ANM $Y \to X$, one considers a similar model with the roles

Algorithm 2 Procedure to decide whether $p(x,y)$ satisfies an Additive Noise Model $X \to Y$ or $Y \to X$ suitable for empirical-Bayes or MML model selection.

Input:

1. I.i.d. sample $\mathcal{D}_N := \{(x_i, y_i)\}_{i=1}^N$ of X and Y ("data");
2. Score function $\hat{C} : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ for measuring model fit and model complexity.

Output: $\hat{C}_{X\to Y}$, $\hat{C}_{Y\to X}$, `dir`.

1. (a) calculate $\hat{C}_{X\to Y} = \hat{C}(\boldsymbol{x}, \boldsymbol{y})$
   (b) calculate $\hat{C}_{Y\to X} = \hat{C}(\boldsymbol{y}, \boldsymbol{x})$
2. Output $\hat{C}_{X\to Y}, \hat{C}_{Y\to X}$ and

$$\texttt{dir} := \begin{cases} X \to Y & \text{if } \hat{C}_{X\to Y} < \hat{C}_{Y\to X},\\ Y \to X & \text{if } \hat{C}_{X\to Y} > \hat{C}_{Y\to X},\\ ? & \text{if } \hat{C}_{X\to Y} = \hat{C}_{Y\to X}. \end{cases}$$

of X and Y reversed. Empirical-Bayes model selection is performed by calculating the maximum evidences (marginal likelihoods) of these two models when optimizing over the hyperparameters, and preferring the model with larger maximum evidence. This is actually a special case (the bivariate case) of an approach proposed by Friedman and Nachman (2000).[14] Considering the negative log marginal likelihoods leads to the following score for the ANM $X \to Y$:

$$\hat{C}_{X\to Y}(\boldsymbol{x}, \boldsymbol{y}) := \min_{\mu,\tau^2,\boldsymbol{\theta},\sigma^2} \left(-\log \mathcal{N}(\boldsymbol{x} \mid \mu\mathbf{1}, \tau^2\boldsymbol{I}) - \log \mathcal{N}(\boldsymbol{y} \mid \mathbf{0}, \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}) + \sigma^2\boldsymbol{I})\right), \tag{11}$$

and a similar expression for $\hat{C}_{Y\to X}$, the score of the ANM $Y \to X$. Here, $\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x})$ is the $N \times N$ kernel matrix $K_{ij} = k_{\boldsymbol{\theta}}(x_i, x_j)$ for a kernel with parameters $\boldsymbol{\theta}$ and $\mathcal{N}(\cdot \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the density of a multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. If one would put a prior distribution on the hyperparameters and integrate them out, this would correspond to Bayesian model selection. In practice, one typically uses "empirical Bayes", which means that the hyperparameters $(\mu, \tau, \boldsymbol{\theta}, \sigma)$ are optimized over instead for computational reasons. Note that this method skips the explicit regression step, instead it (implicitly) integrates over all possible regression functions (Rasmussen and Williams, 2006). Also, it does not distinguish the data splitting and data recycling scenarios, instead it uses the data directly to calculate the (maximum) marginal likelihood. Therefore, the structure of the algorithm is slightly different, see Algorithm 2. In Appendix B we show that this score is actually closely related to the Gaussian score considered in Section 2.2.3.

14. Friedman and Nachman (2000) even hint at using this method for inferring causal relationships, although it seems that they only thought of cases where the functional dependence of the effect on the cause was not injective.

2.2.5 Minimum Message Length Scores

In a similar vein as (empirical) Bayesian marginal likelihoods can be interpreted as measuring likelihood in combination with a complexity penalty, Minimum Message Length (MML) techniques can be used to construct scores that incorporate a trade-off between model fit (likelihood) and model complexity (Grünwald, 2007). Asymptotically, as the number of data points tends to infinity, one would expect the model fit to outweigh the model complexity, and therefore by Lemma 10, simple comparison of MML scores should be enough to identify the direction of an identifiable Additive Noise Model.

A particular MML score was considered by Mooij et al. (2010). This is a special case (referred to in Mooij et al. (2010) as "AN-MML") of their more general framework that allows for non-additive noise. Like (11), the score is a sum of two terms, one corresponding to the marginal density $p(x)$ and the other to the conditional density $p(y\,|\,x)$:

$$\hat{C}_{X\to Y}(\boldsymbol{x},\boldsymbol{y}) := \mathcal{L}(\boldsymbol{x}) + \min_{\boldsymbol{\theta},\sigma^2}\left(-\log \mathcal{N}(\boldsymbol{y}\,|\,0, \boldsymbol{K}_\theta(\boldsymbol{x}) + \sigma^2 \boldsymbol{I})\right). \tag{12}$$

The second term is an MML score for the conditional density $p(y\,|\,x)$, and is identical to the conditional density term in (11). The MML score $\mathcal{L}(\boldsymbol{x})$ for the marginal density $p(x)$ is derived as an asymptotic expansion based on the Minimum Message Length principle for a mixture-of-Gaussians model (Figueiredo and Jain, 2002):

$$\mathcal{L}(\boldsymbol{x}) = \min_{\boldsymbol{\eta}}\left(\sum_{j=1}^{k}\log\left(\frac{N\alpha_j}{12}\right) + \frac{k}{2}\log\frac{N}{12} + \frac{3k}{2} - \log p(\boldsymbol{x}\,|\,\boldsymbol{\eta})\right), \tag{13}$$

where $p(\boldsymbol{x}\,|\,\boldsymbol{\eta})$ is a Gaussian mixture model: $p(x_i\,|\,\boldsymbol{\eta}) = \sum_{j=1}^{k}\alpha_j \mathcal{N}(x_i\,|\,\mu_j,\sigma_j^2)$ with $\boldsymbol{\eta} = (\alpha_i,\mu_i,\sigma_i^2)_{i=1}^k$. The optimization problem (13) is solved numerically by means of the algorithm proposed by Figueiredo and Jain (2002), using a small but nonzero value (10^{-4}) of the regularization parameter.

Comparing this score with the empirical-Bayes score (11), the main conceptual difference is that the former uses a more complicated mixture-of-Gaussians model for the marginal density, whereas (11) uses a simple Gaussian model. We can use (12) in combination with Algorithm 2 in order to estimate the direction of an identifiable Additive Noise Model.

2.2.6 Minimizing HSIC Directly

One can try to apply the idea of combining regression and independence testing into a single procedure (as achieved with the empirical-Bayes score described in Section 2.2.4, for example) more generally. Indeed, a score that measures the dependence between the residuals $\boldsymbol{y}' - f_Y(\boldsymbol{x}')$ and the inputs $\boldsymbol{x}'$ can be minimized with respect to the function f_Y. Mooij et al. (2009) proposed to minimize $\widehat{\mathrm{HSIC}}\big(\boldsymbol{x},\boldsymbol{y}-f(\boldsymbol{x})\big)$ with respect to the function f. However, the optimization problem with respect to f turns out to be a challenging non-convex optimization problem with multiple local minima, and there are no guarantees to find the global minimum. In addition, the performance depends strongly on the selection of suitable kernel bandwidths, for which no suitable automatic procedure is known in this context. Finally, proving consistency of such a method might be challenging, as the minimization may introduce strong dependences between the residuals. Therefore, we do not discuss or evaluate this method in more detail here.

3. Information-Geometric Causal Inference

In this section, we review a class of causal discovery methods that exploits independence of the distribution of the cause and the conditional distribution of the effect given the cause. It nicely complements causal inference based on additive noise by employing asymmetries between cause and effect that have nothing to do with noise.

3.1 Theory

Information-Geometric Causal Inference (IGCI) is an approach that builds upon the assumption that for $X \to Y$ the marginal distribution $\mathbb{P}_X$ contains no information about the conditional[15] $\mathbb{P}_{Y\,|\,X}$ and vice versa, since they represent independent mechanisms. As Janzing and Schölkopf (2010) illustrated for several toy examples, the conditional and marginal distributions $\mathbb{P}_Y, \mathbb{P}_{X\,|\,Y}$ may then contain information about each other, but it is hard to formalize in what sense this is the case for scenarios that go beyond simple toy models. IGCI is based on the strong assumption that X and Y are deterministically related by a bijective function f, that is, $Y = f(X)$ and $X = f^{-1}(Y)$. Although its practical applicability is limited to causal relations with sufficiently small noise and sufficiently high non-linearity, IGCI provides a setting in which the independence of $\mathbb{P}_X$ and $\mathbb{P}_{Y\,|\,X}$ provably implies well-defined dependences between $\mathbb{P}_Y$ and $\mathbb{P}_{X\,|\,Y}$ in a sense described below.

To introduce IGCI, note that the deterministic relation $Y = f(X)$ implies that the conditional $\mathbb{P}_{Y\,|\,X}$ has no density $p(y\,|\,x)$, but it can be represented using f via

$$\mathbb{P}(Y = y\,|\,X = x) = \begin{cases} 1 & \text{if } y = f(x) \\ 0 & \text{otherwise.} \end{cases}$$

The fact that $\mathbb{P}_X$ and $\mathbb{P}_{Y\,|\,X}$ contain no information about each other then translates into the statement that $\mathbb{P}_X$ and f contain no information about each other.

Before sketching a more general formulation of IGCI (Daniušis et al., 2010; Janzing et al., 2012), we begin with the most intuitive case where f is a strictly monotonically increasing differentiable bijection of $[0,1]$. We then assume that the following equality is approximately satisfied:

$$\int_0^1 \log f'(x) p(x)\,dx = \int_0^1 \log f'(x)\,dx, \tag{14}$$

where f' is the derivative of f. To see why (14) is an independence between function f and input density p_X, we interpret $x \mapsto \log f'(x)$ and $x \mapsto p(x)$ as random variables[16] on the probability space $[0,1]$. Then the difference between the two sides of (14) is the covariance of these two random variables with respect to the uniform distribution on $[0,1]$:

$$\begin{aligned}\mathbb{C}\mathrm{ov}(\log f', p_X) &= \mathbb{E}(\log f' \cdot p_X) - \mathbb{E}(\log f')\mathbb{E}(p_X) \\ &= \int \log f'(x)\cdot p(x)dx - \int \log f'(x)dx \int p(x)dx\,.\end{aligned}$$

15. Note that $\mathbb{P}_{Y\,|\,X}$ represents the whole family of distributions $x \mapsto \mathbb{P}_{Y\,|\,X=x}$.
16. Note that random variables are formally defined as maps from a probability space to the real numbers.

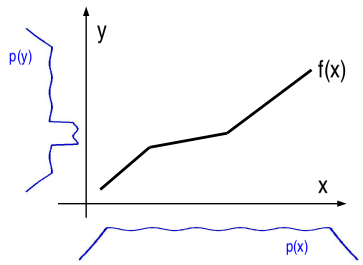


Figure 5: Illustration of the basic intuition behind IGCI. If the density p_X of the cause X is not correlated with the slope of f, then the density p_Y tends to be high in regions where f is flat (and f^{-1} is steep). Source: Janzing et al. (2012).

As shown in Section 2 in Daniušis et al. (2010), p_Y is then related to the inverse function f^{-1} in the sense that

$$\int_0^1 \log(f^{-1})'(y) \cdot p(y)\, dy \geq \int_0^1 \log(f^{-1})'(y)\, dy\,,$$

with equality if and only if f' is constant. Hence, $\log(f^{-1})'$ and p_Y are positively correlated due to

$$\mathbb{E}\big(\log(f^{-1})' \cdot p_Y\big) - \mathbb{E}\big(\log(f^{-1})'\big)\mathbb{E}(p_Y) > 0\,.$$

Intuitively, this is because the density p_Y tends to be high in regions where f is flat, or equivalently, f^{-1} is steep (see also Figure 5). Hence, we have shown that $\mathbb{P}_Y$ contains information about f^{-1} and hence about $\mathbb{P}_{X\,|\,Y}$ whenever $\mathbb{P}_X$ does not contain information about $\mathbb{P}_{Y\,|\,X}$ (in the sense that (14) is satisfied), except for the trivial case where f is linear.

To employ this asymmetry, Daniušis et al. (2010) introduce the expressions

$$C_{X\to Y} := \int_0^1 \log f'(x) p(x) dx \tag{15}$$

$$C_{Y\to X} := \int_0^1 \log(f^{-1})'(y) p(y) dy = -C_{X\to Y}\,. \tag{16}$$

Since the right hand side of (14) is smaller than zero due to $\int_0^1 \log f'(x) dx \leq \log \int_0^1 f'(x) dx = 0$ by concavity of the logarithm (exactly zero only for constant f), IGCI infers $X \to Y$ whenever $C_{X\to Y}$ is negative. Section 3.5 in Daniušis et al. (2010) also shows that

$$C_{X\to Y} = H(Y) - H(X)\,,$$

i.e., the decision rule considers the variable with lower differential entropy as the effect. The idea is that the function introduces new irregularities to a distribution rather than smoothing the irregularities of the distribution of the cause.

Generalization to other reference measures: In the above version of IGCI the uniform distribution on $[0,1]$ plays a special role because it is the distribution with respect to which uncorrelatedness between p_X and $\log f'$ is defined. The idea can be generalized to other reference distributions. How to choose the right one for a particular inference problem is a difficult question which goes beyond the scope of this article. From a high-level perspective, it is comparable to the question of choosing the right kernel for kernel-based machine learning algorithms; it also is an *a priori* structure of the range of X and Y without which the inference problem is not well-defined.

Let u_X and u_Y be densities of X and Y, respectively, that we call "reference densities". For example, uniform or Gaussian distributions would be reasonable choices. Let u_f be the image of u_X under f and $u_{f^{-1}}$ be the image of u_Y under f^{-1}. Then we hypothesize the following generalization of (14):

Principle 2 *If X causes Y via a deterministic bijective function f such that $u_{f^{-1}}$ has a density with respect to Lebesgue measure, then*

$$\int \log \frac{u_{f^{-1}}(x)}{u_X(x)} p(x)\, dx \approx \int \log \frac{u_{f^{-1}}(x)}{u_X(x)} u_X(x)\, dx\,. \tag{17}$$

In analogy to the remarks above, this can also be interpreted as uncorrelatedness of the functions $\log(u_{f^{-1}}/u_X)$ and p_X/u_X with respect to the measure given by the density of u_X with respect to the Lebesgue measure. Again, we hypothesize this because the former expression is a property of the function f alone (and the reference densities) and should thus be unrelated to the marginal density p_X. The special case (14) can be obtained by taking the uniform distribution on $[0,1]$ for u_X and u_Y.

As generalization of (15,16) we define:[17]

$$\begin{aligned} C_{X\to Y} &:= \int \log \frac{u_{f^{-1}}(x)}{u_X(x)} p(x) dx \\ C_{Y\to X} &:= \int \log \frac{u_f(y)}{u_Y(y)} p(y) dy = \int \log \frac{u_X(x)}{u_{f^{-1}}(x)} p(x) dx = -C_{Y\to X}\,, \end{aligned} \tag{18}$$

where the second equality in (18) follows by substitution of variables. Again, the hypothesized independence implies $C_{X\to Y} \leq 0$ since the right hand side of (17) coincides with $-D(u_X \| u_{f^{-1}})$ where $D(\cdot\|\cdot)$ denotes Kullback-Leibler divergence. Hence, we also infer $X \to Y$ whenever $C_{X\to Y} < 0$. Note also that

$$C_{X\to Y} = D(p_X \| u_X) - D(p_X \| u_{f^{-1}}) = D(p_X \| u_X) - D(p_Y \| u_Y)\,,$$

where we have only used the fact that relative entropy is preserved under bijections. Hence, our decision rule amounts to inferring that the density of the cause is closer to its reference density. This decision rule gets quite simple, for instance, if u_X and u_Y are Gaussians with the same mean and variance as p_X and p_Y, respectively. Then it again amounts to inferring $X \to Y$ whenever X has larger entropy than Y after rescaling both X and Y to have the same variance.

17. Note that the formulation in Section 2.3 in Daniušis et al. (2010) is more general because it uses *manifolds* of reference densities instead of a single density.

3.2 Estimation Methods

The specification of the reference measure is essential for IGCI. We describe the implementation for two different choices:

1. *Uniform distribution:* scale and shift X and Y such that extrema are mapped onto 0 and 1.
2. *Gaussian distribution:* scale X and Y to variance 1.

Given this preprocessing step, there are different options for estimating $C_{X\to Y}$ and $C_{Y\to X}$ from empirical data (see Section 3.5 in Daniušis et al., 2010):

1. *Slope-based estimator:*

$$\hat{C}_{X\to Y} := \frac{1}{N-1}\sum_{j=1}^{N-1} \log \frac{|y_{j+1}-y_j|}{x_{j+1}-x_j}\,, \tag{19}$$

   where we assumed the pairs $\{(x_i,y_i)\}$ to be ordered ascendingly according to x_i. Since empirical data are noisy, the y-values need not be in the same order. $\hat{C}_{Y\to X}$ is given by exchanging the roles of X and Y.

2. *Entropy-based estimator:*

$$\hat{C}_{X\to Y} := \hat{H}(Y) - \hat{H}(X)\,, \tag{20}$$

   where $\hat{H}(\cdot)$ denotes some differential entropy estimator.

The theoretical equivalence between these estimators breaks down on empirical data not only due to finite sample effects but also because of noise. For the slope based estimator, we even have

$$\hat{C}_{X\to Y} \neq -\hat{C}_{Y\to X}\,,$$

and thus need to compute both terms separately.

Note that the IGCI implementations discussed here make sense only for continuous variables with a density with respect to Lebesgue measure. This is because the difference quotients are undefined if a value occurs twice. In many empirical data sets, however, the discretization (e.g., due to rounding to some number of digits) is not fine enough to guarantee this. A very preliminary heuristic that was employed in earlier work (Daniušis et al., 2010) removes repeated occurrences by removing data points, but a conceptually cleaner solution would be, for instance, the following procedure: Let $\tilde{x}_j$ with $1\le j\le \tilde{N}$ be the ordered values after removing repetitions and let $\tilde{y}_j$ denote the corresponding y-values. Then we replace (19) with

$$\hat{C}_{X\to Y} := \frac{1}{\sum_{j=1}^{\tilde{N}-1} n_j}\sum_{j=1}^{\tilde{N}-1} n_j \log \frac{|\tilde{y}_{j+1}-\tilde{y}_j|}{\tilde{x}_{j+1}-\tilde{x}_j}\,, \tag{21}$$

where n_j denotes the number of occurrences of $\tilde{x}_j$ in the original data set. Here we have ignored the problem of repetitions of y-values since they are less likely, because they are not

Algorithm 3 General procedure to decide whether $\mathbb{P}_{X,Y}$ is generated by a deterministic monotonic bijective function from X to Y or from Y to X.

Input:

1. I.i.d. sample $\mathcal{D}_N := \{(x_i,y_i)\}_{i=1}^N$ of X and Y ("data");
2. Normalization procedure $\nu : \mathbb{R}^N \to \mathbb{R}^N$;
3. IGCI score estimator $\hat{C} : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$.

Output: $\hat{C}_{X\to Y}$, $\hat{C}_{Y\to X}$, `dir`.

1. Normalization:
   (a) calculate $\tilde{\boldsymbol{x}} = \nu(\boldsymbol{x})$
   (b) calculate $\tilde{\boldsymbol{y}} = \nu(\boldsymbol{y})$
2. Estimation of scores:
   (a) calculate $\hat{C}_{X\to Y} = \hat{C}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}})$
   (b) calculate $\hat{C}_{Y\to X} = \hat{C}(\tilde{\boldsymbol{y}},\tilde{\boldsymbol{x}})$
3. Output $\hat{C}_{X\to Y}, \hat{C}_{Y\to X}$ and

$$\texttt{dir} := \begin{cases} X\to Y & \text{if } \hat{C}_{X\to Y} < \hat{C}_{Y\to X}, \\ Y\to X & \text{if } \hat{C}_{X\to Y} > \hat{C}_{Y\to X}, \\ ? & \text{if } \hat{C}_{X\to Y} = \hat{C}_{Y\to X}. \end{cases}$$

ordered if the relation between X and Y is noisy (and for bijective deterministic relations, they only occur together with repetitions of x anyway).

Finally, let us mention one simple case where IGCI with estimator (19) provably works asymptotically, even though its assumptions are violated. This happens if the effect is a non-injective function of the cause. More precisely, assume $Y = f(X)$ where $f : [0,1] \to [0,1]$ is continuously differentiable and non-injective, and moreover, that p_X is strictly positive and bounded away from zero. To argue that $\hat{C}_{Y\to X} > \hat{C}_{X\to Y}$ asymptotically for $N\to\infty$ we first observe that the mean value theorem implies

$$\frac{|f(x_2)-f(x_1)|}{|x_2-x_1|} \le \max_{x\in[0,1]} |f'(x)| =: s_{\max} \tag{22}$$

for any pair x_1, x_2. Thus, for any sample size N we have $\hat{C}_{X\to Y} \le \log s_{\max}$. On the other hand, $\hat{C}_{Y\to X} \to \infty$ for $N\to\infty$. To see this, note that all terms in the sum (19) are bounded from below by $-\log s_{\max}$ due to (22), while there is no *upper* bound for the summands because adjacent y-values may be from different branches of the non-injective function f and then the corresponding x-values may not be close. Indeed, this will happen for a constant fraction of adjacent pairs. For those, the gaps between the y-values decrease

with $\mathcal{O}(1/N)$ while the distances of the corresponding x-values remain of $\mathcal{O}(1)$. Thus, the overall sum (19) diverges. It should be emphasized, however, that one can have opposite effects for any finite N. To see this, consider the function $x \mapsto 2|x - 1/2|$ and modify it locally around $x = 1/2$ to obtain a continuously differentiable function f. Assume that the probability density in $[1/2, 1]$ is so low that almost all points are contained in $[0, 1/2]$. Then, $\hat{C}_{X\to Y} \approx \log 2$ while $\hat{C}_{Y\to X} \approx -\log 2$ and IGCI with estimator (19) decides (incorrectly) on $Y \to X$. For sufficiently large N, however, a constant (though possibly very small) fraction of y-values come from different branches of f and thus $\hat{C}_{Y\to X}$ diverges (while $\hat{C}_{X\to Y}$ remains bounded from above).

4. Experiments

In this section we describe the data that we used for evaluation, implementation details for various methods, and our evaluation criteria. The results of the empirical study will be presented in Section 5.

4.1 Implementation Details

The complete source code to reproduce our experiments is available online as open source under the FreeBSD license, both as an online appendix and on the homepage of the first author.[18] We used `MatLab` on a `Linux` platform, and made use of external libraries `GPML` v3.5 (2014-12-08) (Rasmussen and Nickisch, 2010) for GP regression and `ITE` v0.61 (Szabó, 2014) for entropy estimation. For parallelization, we used the convenient command line tool `GNU parallel` (Tange, 2011).

4.1.1 Regression

We used standard Gaussian Process (GP) Regression (Rasmussen and Williams, 2006) for nonparametric regression, using the `GPML` implementation (Rasmussen and Nickisch, 2010). We used a squared exponential covariance function, constant mean function, and an additive Gaussian noise likelihood. We used the FITC approximation (Quiñonero-Candela and Rasmussen, 2005) as an approximation for exact GP regression in order to reduce computation time. We found that 100 FITC points distributed on a linearly spaced grid greatly reduce computation time without introducing a noticeable approximation error. Therefore, we used this setting as a default for the GP regression. The computation time of this method scales as $\mathcal{O}(Nm^2T)$, where N is the number of data points, $m = 100$ is the number of FITC points, and T is the number of iterations necessary to optimize the marginal likelihood with respect to the hyperparameters. In practice, this yields considerable speedups compared with exact GP inference, which scales as $\mathcal{O}(N^3T)$.

18. http://www.jorismooij.nl/

Name	Implementation	References
1sp	based on (23)	(Kraskov et al., 2004)
3NN	ITE: Shannon_kNN_k	(Kozachenko and Leonenko, 1987)
sp1	ITE: Shannon_spacing_V	(Vasicek, 1976)
sp2	ITE: Shannon_spacing_Vb	(Van Es, 1992)
sp3	ITE: Shannon_spacing_Vpconst	(Ebrahimi et al., 1994)
sp4	ITE: Shannon_spacing_Vplin	(Ebrahimi et al., 1994)
sp5	ITE: Shannon_spacing_Vplin2	(Ebrahimi et al., 1994)
sp6	ITE: Shannon_spacing_VKDE	(Noughabi and Noughabi, 2013)
KDP	ITE: Shannon_KDP	(Stowell and Plumbley, 2009)
PSD	ITE: Shannon_PSD_SzegoT	(Ramirez et al., 2009; Gray, 2006) (Grenander and Szego, 1958)
EdE	ITE: Shannon_Edgeworth	(van Hulle, 2005)
Gau	based on (9)	
ME1	ITE: Shannon_MaxEnt1	(Hyvärinen, 1997)
ME2	ITE: Shannon_MaxEnt2	(Hyvärinen, 1997)

Table 1: Entropy estimation methods. "ITE" refers to the Information Theoretical Estimators Toolbox (Szabó, 2014). The first group of entropy estimators is nonparametric, the second group makes additional parametric assumptions on the distribution of the data.

4.1.2 Entropy Estimation

We tried many different empirical entropy estimators, see Table 1. The first method, `1sp`, uses a so-called "1-spacing" estimate (see e.g., Kraskov et al., 2004):

$$\hat{H}(\boldsymbol{x}) := \psi(N) - \psi(1) + \frac{1}{N-1}\sum_{i=1}^{N-1} \log|x_{i+1} - x_i| \,, \tag{23}$$

where the x-values should be ordered ascendingly, i.e., $x_i \leq x_{i+1}$, and ψ is the digamma function (i.e., the logarithmic derivative of the gamma function: $\psi(x) = d/dx \log\Gamma(x)$, which behaves as $\log x$ asymptotically for $x \to \infty$). As this estimator would become $-\infty$ if a value occurs more than once, we first remove duplicate values from the data before applying (23). There should be better ways of dealing with discretization effects, but we nevertheless include this particular estimator for comparison, as it was also used in previous implementations of the entropy-based IGCI method (Daniušis et al., 2010; Janzing et al., 2012). These estimators can be implemented in $\mathcal{O}(N \ln N)$ complexity, as they only need to sort the data and then calculate a sum over data points.

We also made use of various entropy estimators implemented in the Information Theoretical Estimators (ITE) Toolbox (Szabó, 2014). The method `3NN` is based on k-nearest neighbors with $k = 3$, all `sp*` methods use Vasicek's spacing method with various corrections, `KDP` uses k-d partitioning, `PSD` uses the power spectral density representation and Szego's theorem, `ME1` and `ME2` use the maximum entropy distribution method, and `EdE` uses the Edgeworth expansion. For more details, see the documentation of the ITE toolbox (Szabó, 2014).

4.1.3 Independence Testing: HSIC

As covariance function for HSIC, we use the popular Gaussian kernel:

$$\kappa_\ell : (x, x') \mapsto \exp\left(-\frac{(x-x')^2}{\ell^2}\right),$$

with bandwidths selected by the median heuristic (Schölkopf and Smola, 2002), i.e., we take

$$\hat{\ell}(\boldsymbol{u}) := \text{median}\{\|u_i - u_j\| : 1 \le i < j \le N, \|u_i - u_j\| \neq 0\},$$

and similarly for $\hat{\ell}(\boldsymbol{v})$. We also compare with a fixed bandwidth of 0.5. As the product of two Gaussian kernels is characteristic, HSIC with such kernels will detect any dependence asymptotically (see also Lemma 12 in Appendix A), at least when the bandwidths are fixed.

The p-value can either be estimated by using permutation, or can be approximated by a Gamma approximation, as the mean and variance of the HSIC value under the null hypothesis can also be estimated in closed form (Gretton et al., 2008). In this work, we use the Gamma approximation for the HSIC p-value.

The computation time of our naïve implementation of HSIC scales as $\mathcal{O}(N^2)$. Using incomplete Cholesky decompositions, one can obtain an accurate approximation in only $\mathcal{O}(N)$ (Jegelka and Gretton, 2007). However, the naïve implementation was fast enough for our purpose.

4.2 Data Sets

We will use both real-world and simulated data in order to evaluate the methods. Here we give short descriptions and refer the reader to Appendix C and Appendix D for details.

4.2.1 Real-world Benchmark Data

The CauseEffectPairs (CEP) benchmark data set that we propose in this work consists of different "cause-effect pairs", each one consisting of samples of a pair of statistically dependent random variables, where one variable is known to cause the other one. It is an extension of the collection of the eight data sets that formed the "CauseEffectPairs" task in the *Causality Challenge #2: Pot-Luck* competition (Mooij and Janzing, 2010) which was performed as part of the NIPS 2008 Workshop on Causality (Guyon et al., 2010). Version 1.0 of the CauseEffectPairs collection that we present here consists of 100 pairs, taken from 37 different data sets from various domains. The CEP data are publicly available at Mooij et al. (2014). Appendix D contains a detailed description of each cause-effect pair and a justification of what we believe to be the ground truth causal relations. Scatter plots of the pairs are shown in Figure 6. In our experiments, we only considered the 95 out of 100 pairs that have one-dimensional variables, i.e., we left out pairs 52–55 and 71.

4.2.2 Simulated Data

As collecting real-world benchmark data is a tedious process (mostly because the ground truths are unknown, and acquiring the necessary understanding of the data-generating process in order to decide about the ground truth is not straightforward), we also studied

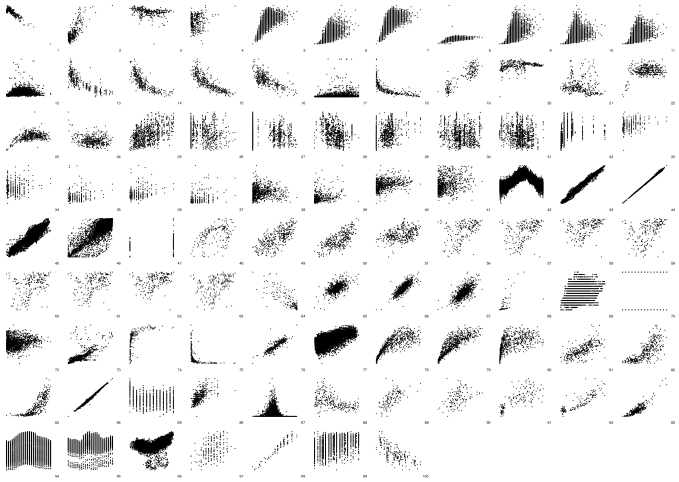


Figure 6: Scatter plots of the cause-effect pairs in the CauseEffectPairs benchmark data. We only show the pairs for which both variables are one-dimensional.

the performance of methods on simulated data where we can control the data-generating process, and therefore can be certain about the ground truth.

Simulating data can be done in many ways. It is not straightforward to simulate data in a "realistic" way, e.g., in such a way that scatter plots of simulated data look similar to those of the real-world data (see Figure 6). For reproducibility, we describe in Appendix C in detail how the simulations were done. Here, we will just sketch the main ideas.

We sample data from the following structural equation models. If we do not want to model a confounder, we use:

$$\begin{aligned} &E_X \sim p_{E_X},\, E_Y \sim p_{E_Y} \\ &X = f_X(E_X) \\ &Y = f_Y(X, E_Y), \end{aligned}$$

and if we do want to include a confounder Z, we use:

$$\begin{aligned} &E_X \sim p_{E_X},\, E_Y \sim p_{E_Y},\, E_Z \sim p_{E_Z} \\ &Z = f_Z(E_Z) \\ &X = f_X(E_X, E_Z) \\ &Y = f_Y(X, E_Y, E_Z). \end{aligned}$$

Here, the noise distributions $p_{E_X}, p_{E_Y}, p_{E_Z}$ are randomly generated distributions, and the causal mechanisms f_Z, f_X, f_Y are randomly generated functions. Sampling the random

distributions for a noise variable E_X (and similarly for E_Y and E_Z) is done by mapping a standard-normal distribution through a random function, which we sample from a Gaussian Process. The causal mechanism f_X (and similarly f_Y and f_Z) is drawn from a Gaussian Process as well. After sampling the noise distributions and the functional relations, we generate data for X, Y, Z. Finally, Gaussian measurement noise is added to both X and Y.

By controlling various hyperparameters, we can control certain aspects of the data generation process. We considered four different scenarios. `SIM` is the default scenario without confounders. `SIM-c` includes a one-dimensional confounder, whose influences on X and Y are typically equally strong as the influence of X on Y. The setting `SIM-ln` has low noise levels, and we would expect IGCI to work well in this scenario. Finally, `SIM-G` has approximate Gaussian distributions for the cause X and approximately additive Gaussian noise (on top of a nonlinear relationship between cause and effect); we expect that methods which make these Gaussianity assumptions will work well in this scenario. Scatter plots of the simulated data are shown in Figures 7–10.

4.3 Preprocessing and Perturbations

The following preprocessing was applied to each pair (X, Y). Both variables X and Y were standardized (i.e., an affine transformation is applied on both variables such that their empirical mean becomes 0, and their empirical standard deviation becomes 1). In order to study the effect of discretization and other small perturbations of the data, one of these four perturbations was applied:

unperturbed : No perturbation is applied.

discretized : Discretize the variable that has the most unique values such that after discretization, it has as many unique values as the other variable. The discretization procedure repeatedly merges those values for which the sum of the absolute error that would be caused by the merge is minimized.

undiscretized : "Undiscretize" both variables X and Y. The undiscretization procedure adds noise to each data point z, drawn uniformly from the interval $[0, z' - z]$, where z' is the smallest value $z' > z$ that occurs in the data.

small noise : Add tiny independent Gaussian noise to both X and Y (with mean 0 and standard deviation 10^{-9}).

Ideally, a causal discovery method should be robust against these and other small perturbations of the data.

4.4 Evaluation Measures

We evaluate the performance of the methods in two different ways:

forced-decision : given a sample of a pair (X, Y) the methods have to decide either $X \to Y$ or $Y \to X$; in this setting we evaluate the accuracy of these decisions;

ranked-decision : we used the scores $\hat{C}_{X\to Y}$ and $\hat{C}_{Y\to X}$ to construct heuristic confidence estimates that are used to rank the decisions; we then produced receiver-operating

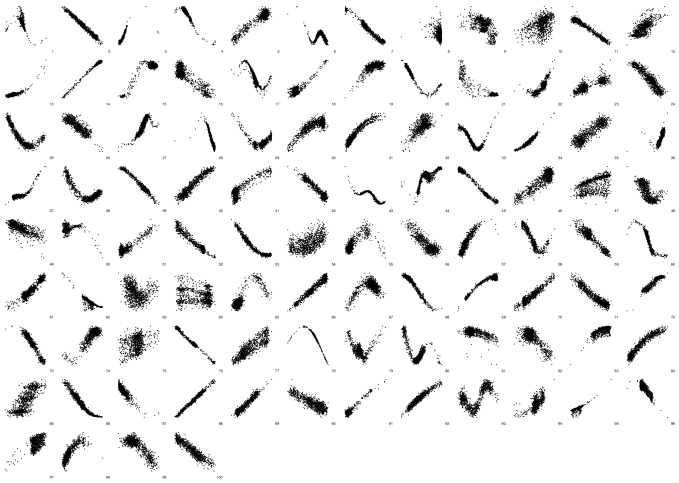


Figure 7: Scatter plots of the cause-effect pairs in simulation scenario `SIM`.

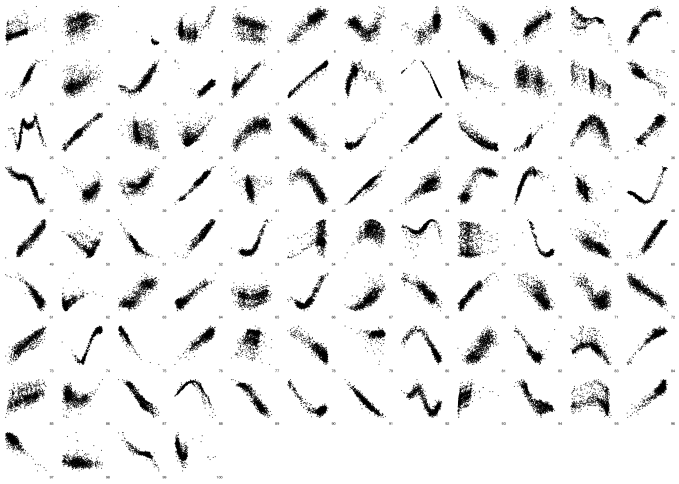


Figure 8: Scatter plots of the cause-effect pairs in simulation scenario `SIM-c`.

Figure 9: Scatter plots of the cause-effect pairs in simulation scenario `SIM-ln`.

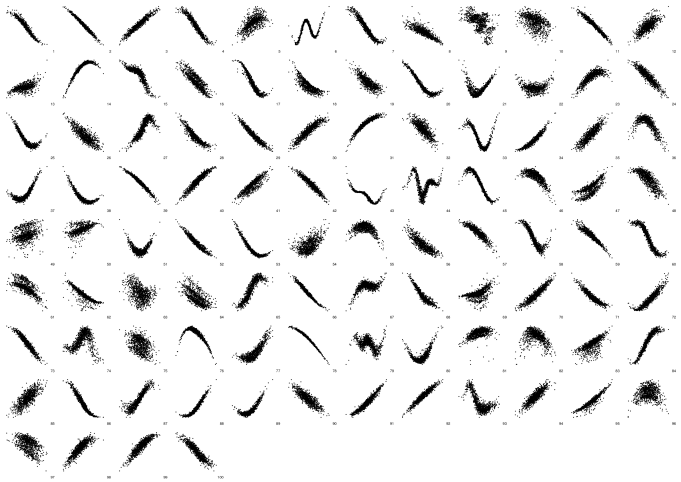


Figure 10: Scatter plots of the cause-effect pairs in simulation scenario `SIM-G`.

characteristic (ROC) curves and used the area under the curve (AUC) as performance measure.

Some methods have an advantage in the second setting, as the scores on which their decisions are based yield a reasonably accurate ranking of the decisions. By only taking the most confident (highest ranked) decisions, the accuracy of these decisions increases, and this leads to a higher AUC than for random confidence estimates. Which of the two evaluation measures (accuracy or AUC) is the most relevant depends on the application.[19]

4.4.1 Weights

For the CEP data, we cannot always consider pairs that come from the same data set as independent. For example, in the case of the `Abalone` data set (Bache and Lichman, 2013; Nash et al., 1994), the variables "whole weight", "shucked weight", "viscera weight", "shell weight" are strongly correlated. Considering the four pairs (age, whole weight), (age, shucked weight), etc., as independent could introduce a bias. We (conservatively) correct for that bias by downweighting these pairs. In general, we chose the weights such that the weights of all pairs from the same data set are equal and sum to one. For the real-world cause-effect pairs, the weights are specified in Table 4. For the simulated pairs, we do not use weighting.

4.4.2 Forced-decision: Evaluation of Accuracy

In the "forced-decision" setting, we calculate the weighted accuracy of a method in the following way:

$$\text{accuracy} = \frac{\sum_{m=1}^{M} w_m \delta_{\hat{d}_m, d_m}}{\sum_{m=1}^{M} w_m},$$

where d_m is the true causal direction for the m'th pair (either "$\leftarrow$" or "$\rightarrow$"), $\hat{d}_m$ is the estimated direction (one of "$\leftarrow$", "$\rightarrow$", and "?"), and w_m is the *weight* of the pair. Note that we are only awarding correct decisions, i.e., if no estimate is given (d_m = "?"), this will negatively affect the accuracy. We calculate confidence intervals assuming a binomial distribution using the method by Clopper and Pearson (1934).

4.4.3 Ranked-decision: Evaluation of AUC

To construct an ROC curve, we need to rank the decisions based on some heuristic estimate of confidence. For most methods we simply use

$$\hat{S} := -\hat{C}_{X \to Y} + \hat{C}_{Y \to X}. \tag{24}$$

19. In earlier work, we have reported accuracy-decision rate curves instead of ROC curves. However, it is easy to visually overinterpret the significance of such a curve in the low decision-rate region. In addition, AUC was used as the evaluation measure by Guyon et al. (2016). A slight disadvantage of ROC curves is that they introduce an asymmetry between "positives" and "negatives", whereas for our task, there is no such asymmetry: we can easily transform a positive into a negative and vice versa by swapping the variables X and Y. Therefore, "accuracy" is a more natural measure than "precision" in our setting. We mitigate this problem by balancing the class labels by swapping X and Y variables for a subset of the pairs.

The interpretation is that the higher $\hat{S}$, the more likely $X \to Y$, and the lower $\hat{S}$, the more likely $Y \to X$. For `ANM-pHSIC`, we use a different heuristic:

$$\hat{S} := \begin{cases} \frac{-1}{\min\{\hat{C}_{X\to Y}, \hat{C}_{Y\to X}\}} & \text{if } \hat{C}_{X\to Y} < \hat{C}_{Y\to X} \\ \frac{1}{\min\{\hat{C}_{X\to Y}, \hat{C}_{Y\to X}\}} & \text{if } \hat{C}_{X\to Y} > \hat{C}_{Y\to X}, \end{cases} \tag{25}$$

and for `ANM-HSIC`, we use:

$$\hat{S} := \begin{cases} \frac{-1}{1-\min\{\hat{C}_{X\to Y}, \hat{C}_{Y\to X}\}} & \text{if } \hat{C}_{X\to Y} < \hat{C}_{Y\to X} \\ \frac{1}{1-\min\{\hat{C}_{X\to Y}, \hat{C}_{Y\to X}\}} & \text{if } \hat{C}_{X\to Y} > \hat{C}_{Y\to X}. \end{cases} \tag{26}$$

In the "ranked-decision" setting, we also use weights to calculate weighted recall (depending on a threshold θ):

$$\text{recall}(\theta) = \frac{\sum_{m=1}^{M} w_m \mathbf{1}_{\hat{S}_m > \theta} \delta_{d_m,\to}}{\sum_{m=1}^{M} w_m \delta_{d_m,\to}},$$

where $\hat{S}_m$ is the heuristic score of the m'th pair (high values indicating high likelihood that $d_m = \ \to$, low values indicating high likelihood that $d_m = \ \leftarrow$), and the weighted precision (also depending on θ):

$$\text{precision}(\theta) = \frac{\sum_{m=1}^{M} w_m \mathbf{1}_{\hat{S}_m > \theta} \delta_{d_m,\to}}{\sum_{m=1}^{M} w_m \mathbf{1}_{\hat{S}_m > \theta}}.$$

We use the `MatLab` routine `perfcurve` to produce (weighted) ROC curves and to estimate weighted AUC and confidence intervals for the weighted AUC by bootstrapping.[20]

5. Results

In this section, we report the results of the experiments that we carried out in order to evaluate the performance of various methods. We plot the accuracies and AUCs as box plots, indicating the estimated (weighted) accuracy or AUC, the corresponding 68% confidence interval, and the 95% confidence interval. If there were pairs for which no decision was taken because of some failure, the number of nondecisions is indicated on the corresponding box plot. The methods that we evaluated are listed in Table 2. Computation times are reported in Appendix E.

5.1 Additive Noise Models

We start by reporting the results for methods that exploit additivity of the noise. Figure 11 shows the performance of all ANM methods on different unperturbed data sets, i.e., the CEP benchmark and various simulated data sets. Figure 12 shows the performance of the same methods on different perturbations of the CEP benchmark data. The six variants `sp1`,...,`sp6` of the spacing estimators perform very similarly, so we show only the results for `ANM-ent-sp1`. For the "undiscretized" perturbed version of the CEP benchmark data, GP regression failed in one case because of a numerical problem, which explains the failures across all methods in Figure 12 for that case.

20. We used the "percentile method" (`BootType = 'per'`) as the default method ("bias corrected and accelerated percentile method") sometimes yielded an estimated AUC that fell outside the estimated 95% confidence interval of the AUC.

Name	Algorithm	Score	Heuristic	Details
`ANM-pHSIC`	1	(4)	(25)	DR, adaptive kernel bandwidth
`ANM-HSIC`	1	(5)	(26)	DR, adaptive kernel bandwidth
`ANM-HSIC-ds`	1	(5)	(26)	DS, adaptive kernel bandwidth
`ANM-HSIC-fk`	1	(5)	(26)	DR, fixed kernel bandwidth (0.5)
`ANM-HSIC-ds-fk`	1	(5)	(26)	DS, fixed kernel bandwidth (0.5)
`ANM-ent-...`	1	(8)	(24)	DR, entropy estimators from Table 1
`ANM-Gauss`	1	(10)	(24)	DR
`ANM-FN`	2	(11)	(24)	
`ANM-MML`	2	(12)	(24)	
`IGCI-slope`	3	(19)	(24)	
`IGCI-slope++`	3	(21)	(24)	
`IGCI-ent-...`	3	(20)	(24)	Entropy estimators from Table 1

Table 2: The methods that are evaluated in this work. DS = Data Splitting, DR = Data Recycling.

5.1.1 HSIC-based Scores

As we see in Figure 11 and Figure 12, the ANM methods that use HSIC perform reasonably well on all data sets, obtaining accuracies between 63% and 85%. Note that the simulated data (and also the real-world data) deviate in at least three ways from the assumptions made by the additive noise method: (i) the noise is not additive, (ii) a confounder can be present, and (iii) additional measurement noise was added to both cause and effect. Moreover, the results turn out to be robust against small perturbations of the data. This shows that the additive noise method can perform reasonably well, even in case of model misspecification.

The results of `ANM-pHSIC` and `ANM-HSIC` are very similar. The influence of various implementation details on performance is small. On the CEP benchmark, data-splitting (`ANM-HSIC-ds`) slightly increases accuracy, whereas using a fixed kernel (`ANM-HSIC-fk`, `ANM-HSIC-ds-fk`) slightly lowers AUC. Generally, the differences in performance are small and not statistically significant. The variant `ANM-HSIC-ds-fk` is proved to be consistent in Appendix A. If standard GP regression satisfies the property in (32), then `ANM-HSIC-fk` is also consistent.

5.1.2 Entropy-based Scores

For the entropy-based score (8), we see in Figure 11 and Figure 12 that the results depend strongly on which entropy estimator is used.

All (nonparametric) entropy estimators (`1sp`, `3NN`, `spi`, `KDP`, `PSD`) perform well on simulated data, with the exception of `EdE`. On the CEP benchmark on the other hand, the performance varies greatly over estimators. One of the reasons for this are discretization

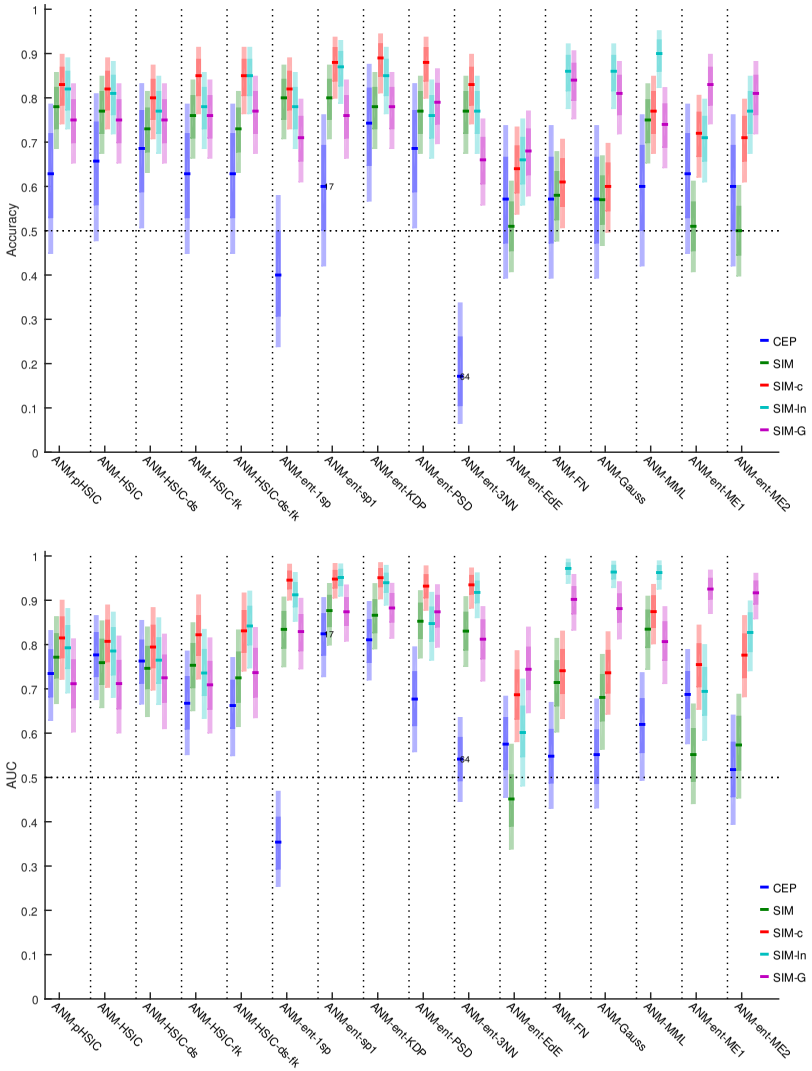


Figure 11: Accuracies (top) and AUCs (bottom) of various ANM methods on different (unperturbed) data sets. For the variants of the spacing estimator, only the results for `sp1` are shown, as results for `sp2`,...,`sp6` were similar.

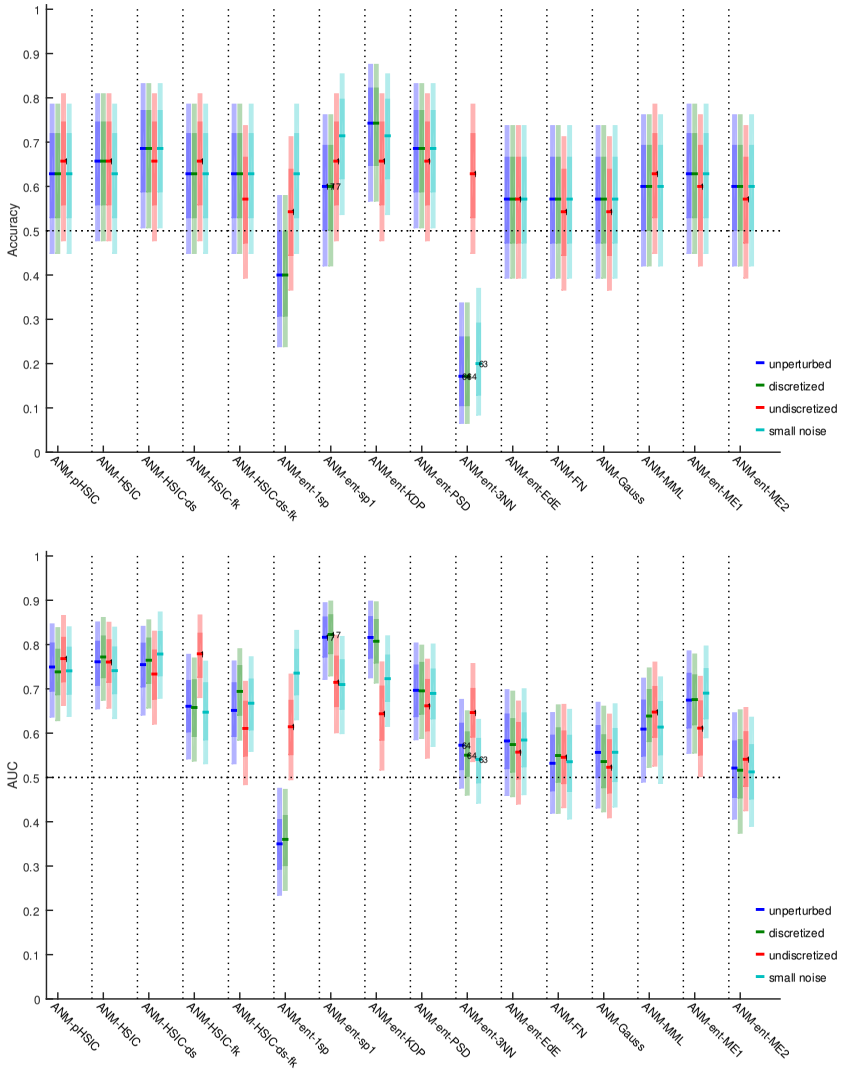


Figure 12: Accuracies (top) and AUCs (bottom) of various ANM methods on different perturbations of the CEP benchmark data. For the variants of the spacing estimator, only the results for `sp1` are shown, as results for `sp2`,...,`sp6` were similar.

effects. Indeed, the differential entropy of a variable that can take only a finite number of values is $-\infty$. The way in which differential entropy estimators treat values that occur multiple times differs, and this can have a large influence on the estimated entropy. For example, `1sp` simply ignores values that occur more than once, which leads to a performance that is below chance level on the CEP data. `3NN` returns $-\infty$ (for both $\hat{C}_{X\to Y}$ and $\hat{C}_{Y\to X}$) in the majority of the pairs in the CEP benchmark and therefore often cannot decide. The spacing estimators `sp`i also return $-\infty$ in quite a few cases. The only (non-parametric) entropy-based ANM methods that perform well on both the CEP benchmark data and the simulated data are `ANM-ent-KDP` and `ANM-ent-PSD`. Of these two methods, `ANM-ent-PSD` seems more robust under perturbations than `ANM-ent-KDP`, and can compete with the HSIC-based methods.

5.1.3 Other Scores

Consider now the results for the parametric entropy estimators (`ANM-Gauss`, `ANM-ent-ME1`, `ANM-ent-ME2`), the empirical-Bayes method `ANM-FN`, and the MML method `ANM-MML`.

First, note that `ANM-Gauss` and `ANM-FN` perform very similarly. This means that the difference between these two scores (i.e., the complexity measure of the regression function, see also Appendix B) does not outweigh the common part (the likelihood) of these two scores. Both these scores do not perform much better than chance on the CEP data, probably because the Gaussianity assumption is typically violated in real data. They do obtain high accuracies and AUCs for the `SIM-ln` and `SIM-G` scenarios. For `SIM-G` this is to be expected, as the assumption that the cause has a Gaussian distribution is satisfied in that scenario. For `SIM-ln` it is not evident why these scores perform so well—it could be that the noise is close to additive and Gaussian in that scenario.

The related score `ANM-MML`, which employs a more sophisticated complexity measure for the distribution of the cause, performs better on the two simulation settings `SIM` and `SIM-c`. However, `ANM-MML` performs worse in the `SIM-G` scenario, which is probably due to a higher variance of the MML complexity measure compared with the simple Gaussian entropy measure. This is in line with expectations. However, performance of `ANM-MML` is hardly better than chance on the CEP data. In particular, the AUC of `ANM-MML` is worse than that of `ANM-pHSIC`.

The parametric entropy estimators `ME1` and `ME2` do not perform very well on the `SIM` data, although their performance on the other simulated data sets (in particular `SIM-G`) is good. The reasons for this behaviour are not understood; we speculate that the parametric assumptions made by these estimators match the actual distribution of the data in these particular simulation settings quite well. The accuracy and AUC of `ANM-ent-ME1` and `ANM-ent-ME2` on the CEP data are lower than those of `ANM-pHSIC`.

5.2 Information Geometric Causal Inference

Here we report the results of the evaluation of different IGCI variants. Figure 13 shows the performance of all the IGCI variants on different (unperturbed) data sets, the CEP benchmark and four different simulation settings, using the uniform base measure. Figure 14 shows the same for the Gaussian base measure. Figure 15 shows the performance of the IGCI methods on different perturbations of the CEP benchmark, using the uniform base measure,

and Figure 16 for the Gaussian base measure. Again, the six variants `sp1`,...,`sp6` of the spacing estimators perform very similarly, so we show only the results for `IGCI-ent-sp1`.

Let us first look at the performance on simulated data. Note that none of the IGCI methods performs well on the simulated data when using the uniform base measure. A very different picture emerges when using the Gaussian base measure: here the performance covers a wide spectrum, from lower than chance level on the `SIM` data to accuracies higher than 90% on `SIM-G`. The choice of the base measure clearly has a larger influence on the performance than the choice of the estimation method.

As IGCI was designed for the bijective deterministic case, one would expect that IGCI would work best on `SIM-ln` (without depending too strongly on the reference measure), because in that scenario the noise is relatively small. Surprisingly, this does not turn out to be the case. To understand this unexpected behavior, we inspect the scatter plots in Figure 9 and observe that the functions in `SIM-ln` are either non-injective or relatively close to linear. Both can spoil the performance despite having low noise (see also the remarks at the end of Subsection 3.2 on finite sample effects).

For the more noisy settings, earlier experiments showed that `IGCI-slope` and `IGCI-1sp` can perform surprisingly well on simulated data (Janzing et al., 2012). Here, however, we see that the performance of all IGCI variants on noisy data depends strongly on characteristics of the data generation process and on the chosen base measure. IGCI seems to pick up certain features in the data that turn out to be correlated with the causal direction in some settings, but can be anticorrelated with the causal direction in other settings. In addition, our results suggest that if the distribution of the cause is close to the base measure used in IGCI, then also for noisy data the method may work well (as in the `SIM-G` setting). However, for causal relations that are not sufficiently non-linear, performance can drop significantly (even below chance level) in case of a discrepancy between the actual distribution of the cause and the base measure assumed by IGCI.

Even though the performance of all IGCI variants with uniform base measure is close to chance level on the simulated data, most methods perform better than chance on the CEP data (with the exception of `IGCI-ent-sp1` and `IGCI-ent-3NN`). When using the Gaussian base measure, performance of IGCI methods on CEP data varies considerably depending on implementation details. For some IGCI variants the performance on CEP data is robust to small perturbations (most notably the parametric entropy estimators), but for most non-parametric entropy estimators and for `IGCI-slope`, there is a strong dependence and sometimes even an inversion of the accuracy when perturbing the data slightly. We do not have a good explanation for these observations.

5.2.1 Original Implementations

Let us now take a closer look at the accuracies of the original methods `IGCI-slope` and `IGCI-ent-1sp` that were proposed by Daniušis et al. (2010); Janzing et al. (2012), and at the newly introduced `IGCI-slope++` that is closely related to `IGCI-slope`. The IGCI variants `slope`, `slope++` and `ent-1sp` perform very similar on all data sets. For both uniform and Gaussian base measures, the performance is better than chance level on the CEP benchmark, but not as much as in previous evaluations on earlier versions of the benchmark. The discrepancy with the accuracies of around 80% reported by Janzing et al.

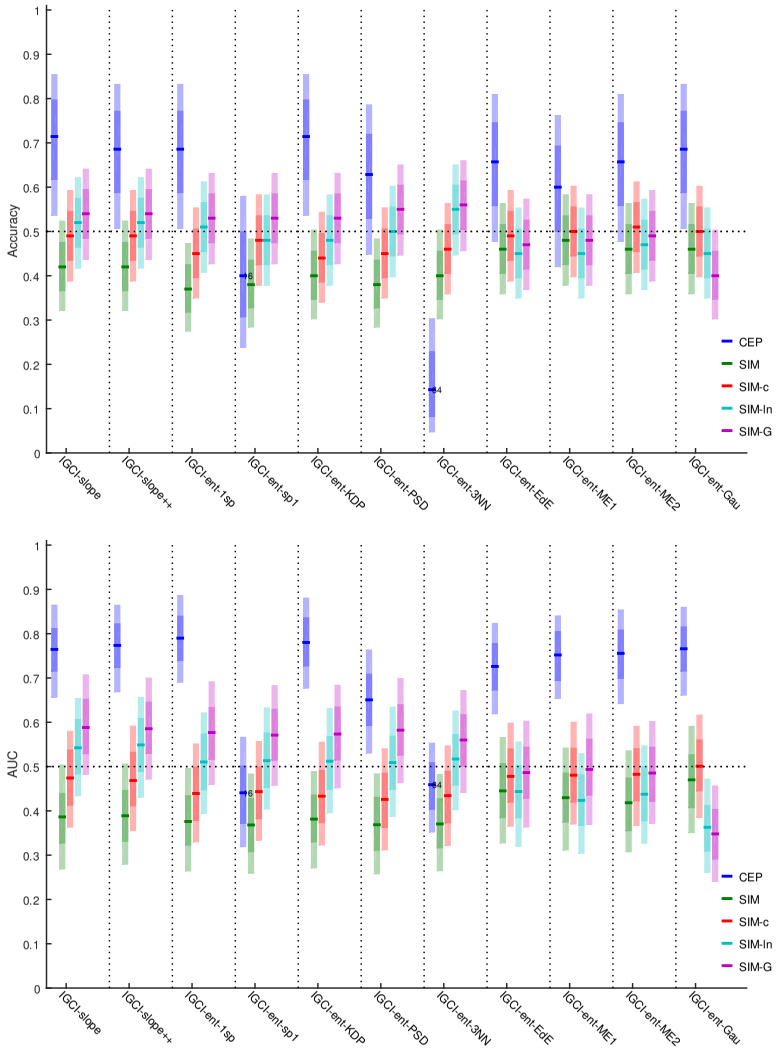


Figure 13: Accuracies (top) and AUCs (bottom) for various IGCI methods using the uniform base measure on different (unperturbed) data sets.

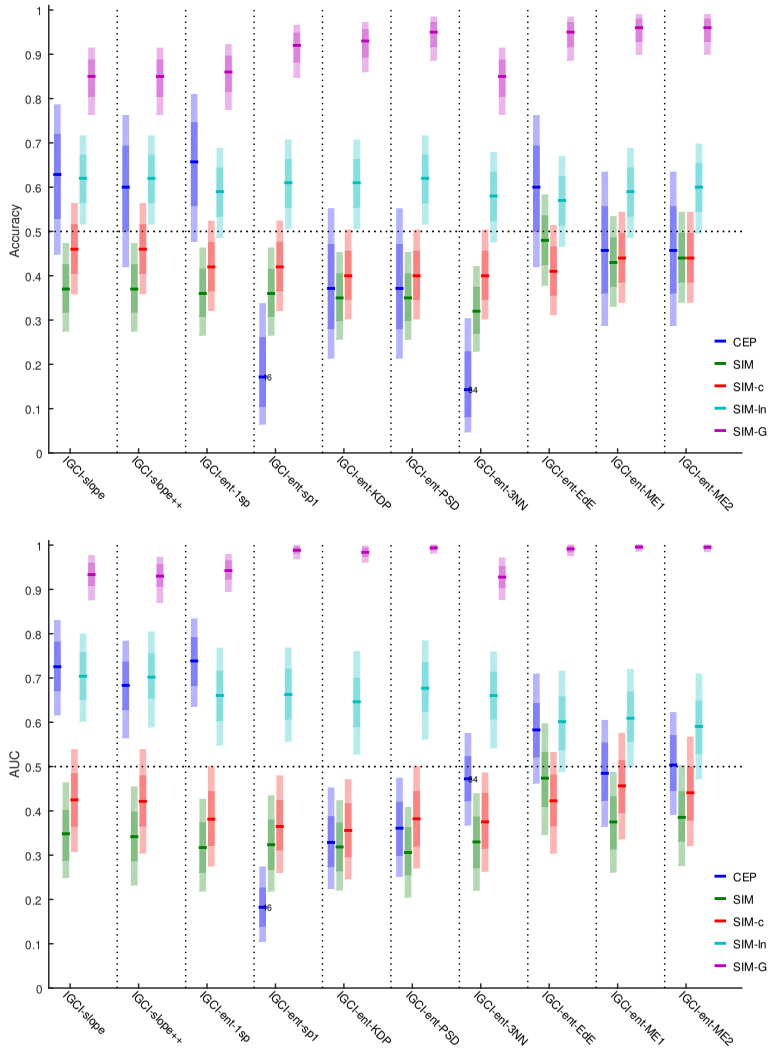


Figure 14: Accuracies (top) and AUCs (bottom) for various IGCI methods using the Gaussian base measure on different (unperturbed) data sets.

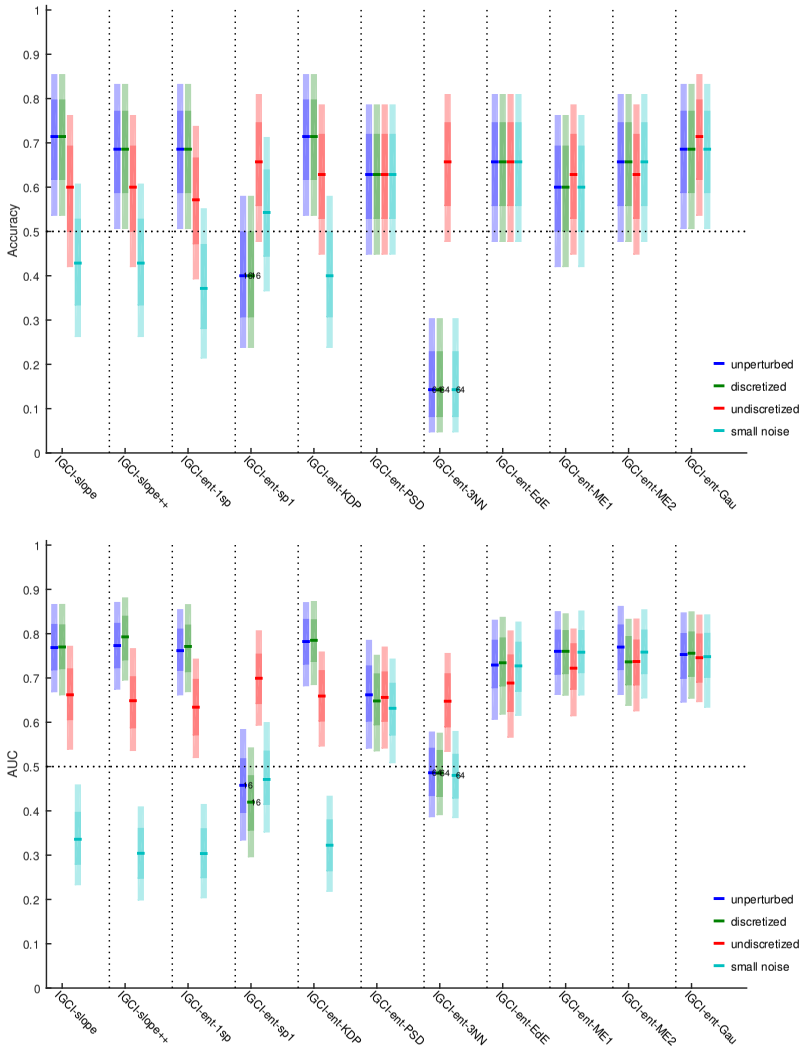


Figure 15: Accuracies (top) and AUCs (bottom) for various IGCI methods using the uniform base measure on different perturbations of the CEP benchmark data.

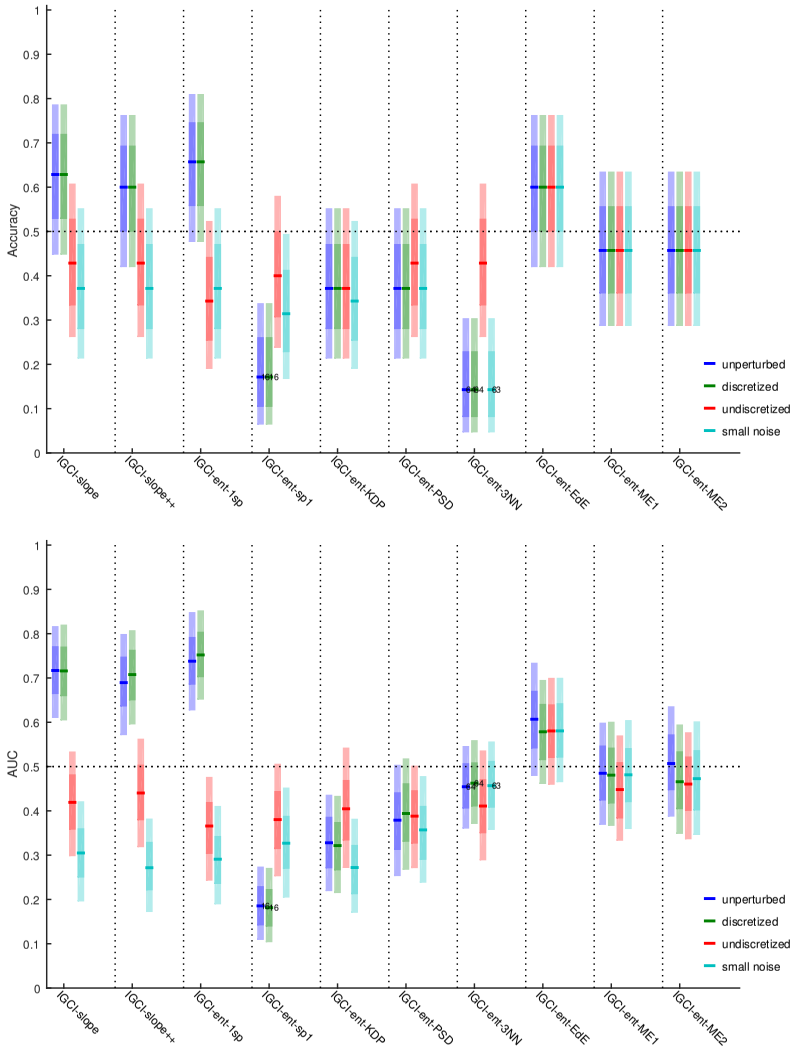


Figure 16: Accuracies (top) and AUCs (bottom) for various IGCI methods using the Gaussian base measure on different perturbations of the CEP benchmark data.

(2012) could be explained by the fact that here we evaluate on a larger set of cause-effect pairs, and we chose the weights more conservatively.

It is also interesting to look at the behavior under perturbations of the CEP data. When using the uniform base measure, the accuracy of both `IGCI-slope` and `IGCI-ent-1sp` drops back to chance level if small noise is added, whereas AUC even becomes worse than chance level. For the Gaussian base measure, both accuracy and AUC become worse than random guessing on certain perturbations of the CEP data, although discretization does not affect performance. This observation motivated the introduction of the slope-based estimator `IGCI-slope++` that uses (21) instead of (19) in order to deal better with repetitions of values. However, as we can see, this estimator does not perform better in practice than the original estimator `IGCI-slope`.

5.2.2 Nonparametric Entropy Estimators

It is clear that discretization effects play an important role in the performance of the nonparametric entropy estimators. For example, the closely related estimators `1sp` and `sp`i perform comparably on simulated data, but on the CEP data, the `sp`i estimators perform worse because of nondecisions due to repeated values. Similarly, the bad performance of `IGCI-ent-3NN` on the CEP data is related to discretization effects. This is in line with our observations on the behavior of these entropy estimators when using them for entropy-based ANM methods.

Further, note that the performance of `IGCI-ent-KDP` is qualitatively similar to that of `IGCI-ent-PSD`, but in contrast with the `PSD` estimator, the results of the `KDP` estimator are not robust under perturbations when using the uniform base measure. The only nonparametric entropy estimators that give results that are robust to small perturbations of the data (for both base measures) are `PSD` and `EdE`. The performance of `IGCI-ent-PSD` on the CEP benchmark depends on the chosen base measure: for the uniform base it is better than chance level, for the Gaussian base measure it is worse than chance level. Interestingly, the `EdE` estimator that performed poorly for ANM gives consistently good results on the CEP benchmark when used for IGCI: it is the only nonparametric entropy estimator that yields results that are better than chance for both base measures and irrespective of whether the data were perturbed or not.

Apparently, implementation details of entropy estimators can result in huge differences in performance, often in ways that we do not understand well.

5.2.3 Parametric Entropy Estimators

Let us finally consider the performance of entropy-based IGCI methods that use parametric entropy estimators, which make additional assumptions on the distribution. As expected, these estimators are robust to small perturbations of the data.

Interestingly, `IGCI-ent-Gau` with uniform base measure turns out to be one of the best IGCI methods on the CEP benchmark, in the sense that it obtains good accuracy and AUC and in addition is robust to perturbations. Note that the performance of `IGCI-ent-Gau` on the CEP benchmark is comparable with that of the original implementation `IGCI-slope` and the newer version `IGCI-slope++`, but that only `IGCI-ent-Gau` is robust to small perturbations of the data. This estimator simply estimates entropy by assuming a Gaussian

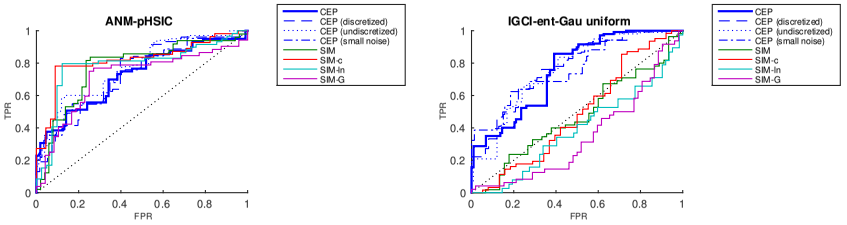


Figure 17: ROC curves for two of the best-performing methods (`ANM-pHSIC` and `IGCI-ent-Gau`). Both methods work well on the CEP benchmark and keep performing well under small perturbations of the data, but only `ANM-pHSIC` also performs well on the simulated data.

distribution. In other words, it uses:

$$\hat{C}_{X\to Y} := \frac{1}{2}\log\widehat{\mathbb{V}\mathrm{ar}}(\tilde{\boldsymbol{y}}) - \frac{1}{2}\log\widehat{\mathbb{V}\mathrm{ar}}(\tilde{\boldsymbol{x}}) = \log\left(\frac{\sqrt{\widehat{\mathbb{V}\mathrm{ar}}(\boldsymbol{y})}}{\max(\boldsymbol{y})-\min(\boldsymbol{y})}\Big/\frac{\sqrt{\widehat{\mathbb{V}\mathrm{ar}}(\boldsymbol{x})}}{\max(\boldsymbol{x})-\min(\boldsymbol{x})}\right).$$

Apparently, the ratio of the size of the support of the distribution and its standard deviation is already quite informative on the causal direction for the CEP data. This might also explain the relatively good performance on this benchmark of `IGCI-ent-ME1` and `IGCI-ent-ME2` when using the uniform base measure, as these estimate entropy by fitting a parametric distribution to the data (which includes Gaussian distributions as a special case). On the other hand, these methods do not work better than chance on the simulated data.

Now let us look at the results when using the Gaussian base measure. `IGCI-ent-Gau` makes no sense in combination with the Gaussian base measure. `IGCI-ent-ME1` and `IGCI-ent-ME2` on the CEP data do not perform better than chance. On simulated data, they only do (extremely) well for the `SIM-G` scenario which uses a Gaussian distribution of the cause measure, i.e., for which the chosen base measure exactly corresponds with the distribution of the cause.

6. Discussion and Conclusion

In this work, we considered a challenging bivariate causal discovery problem, where the task is to decide whether X causes Y or vice versa, using only a sample of purely observational data. We reviewed two families of methods that can be applied to this task: methods based on Additive Noise Models (ANMs) and Information Geometric Causal Inference (IGCI) methods. We discussed various possible implementations of these methods and how they are related.

In addition, we have proposed the CauseEffectPairs benchmark data set consisting of 100 real-world cause-effect pairs and we provided our justifications for the ground truths. We have used this benchmark data in combination with several simulated data sets in order

to evaluate various bivariate causal discovery methods. Our main conclusions (illustrated in Figure 17) are twofold:

1. The ANM methods that use HSIC perform reasonably well on all data sets (including the perturbed versions of the CEP benchmark and all simulation settings), obtaining accuracies between 63% and 85% (see Figures 11 and 12). In particular, the original `ANM-pHSIC` method obtains an accuracy of 63 ± 10 % and an AUC of 0.74 ± 0.05 on the CEP benchmark. The only other ANM method that performs well on all data sets is `ANM-ent-PSD`. It obtains a higher accuracy ($69 \pm 10\%$) than `ANM-pHSIC` on the CEP benchmark, but a lower AUC (0.68 ± 0.06), but these differences are not statistically significant.

2. The performance of IGCI-based methods varies greatly depending on implementation details, perturbations of the data and certain characteristics of the data, in ways that we do not fully understand (see Figures 13, 14, 15, 16). In many cases, causal relations seem to be too linear for IGCI to work well. None of the IGCI implementations performed well on *all* data sets that we considered, and the apparent better-than-chance performance of some of these methods on the CEP benchmark remains somewhat of a mystery.

The former conclusion about the performance of `ANM-pHSIC` is in line with earlier reports, but the latter conclusion is surprising, considering that good performance of `IGCI-slope` and `IGCI-ent-1sp` has been reported on several occasions in earlier work (Daniušis et al., 2010; Mooij et al., 2010; Janzing et al., 2012; Statnikov et al., 2012; Sgouritsa et al., 2015). One possible explanation that the performance of IGCI on simulated data here differs from earlier reports is that earlier simulations used considerably smoother distributions of the cause variable.

Ironically, the original ANM method `ANM-pHSIC` proposed by Hoyer et al. (2009) turned out to be one of the best methods overall, despite the recent research efforts aimed at developing better methods. This observation motivated the consistency proof of HSIC-based ANM methods, the major theoretical contribution of this work. We expect that extending this consistency result to the multivariate case (see also Peters et al., 2014) should be straightforward.

One reason for the disappointing performance of several methods (in particular, the slope-based IGCI estimators and methods that make use of certain nonparametric differential entropy estimators) is discretization. When dealing with real-world data on a computer, variables of a continuous nature are usually discretized because they have to be represented as floating point numbers. Often, additional rounding is applied, for example because only the most significant digits are recorded. We found that for many methods, especially for those that use differential entropy estimators, (coarse) discretization of the data causes problems. This suggests that performance of several methods can still be improved, e.g., by using entropy estimators that are more robust to such discretization effects. The HSIC independence measure (and its p-value) and the `PSD` entropy estimator were found to be robust against small perturbations of the data, including discretization.

Since we compared many different implementations (which turned out to have quite different performance characteristics), we need to use a strong correction for multiple testing

if we would like to conclude that one of these methods performs significantly better than chance. Although it seems unlikely that the good performance of `ANM-pHSIC` on both CEP data *and* all simulated data is due to chance alone, eventually we are most interested in the performance on real-world data alone. Unfortunately, the CEP benchmark turned out to be too small to warrant significant conclusions for any of the tested methods.

A rough estimate how large the CauseEffectPairs benchmark should have been in order to obtain significant results can easily be made. Using a standard (conservative) Bonferroni correction, taking into account that we compared 37 methods, we would need about 120 (weighted) pairs for an accuracy of 65% to be considered significant (with two-sided testing and 5% significance threshold). This is about four times as much as the current number of 37 (weighted) pairs in the CauseEffectPairs benchmark. Therefore, we suggest that at this point, the highest priority regarding future work should be to obtain more validation data, rather than developing additional methods or optimizing computation time of existing methods. We hope that our publication of the CauseEffectPairs benchmark data inspires researchers to collaborate on this important task and we invite everybody to contribute pairs to the CauseEffectPairs benchmark data.

Concluding, our results provide some evidence that distinguishing cause from effect is indeed possible from purely observational real-world data by exploiting certain statistical patterns in the data. However, the performance of current state-of-the-art bivariate causal discovery methods still has to be improved further in order to enable practical applications, and more validation data are needed in order to obtain statistically significant conclusions. Furthermore, it is not clear at this stage under what assumptions current methods could be extended to deal with possible confounding variables, an important issue in practice.

Acknowledgments

JMM was supported by NWO, the Netherlands Organization for Scientific Research (VIDI grant 639.072.410). JP received funding from the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement no 326496. The authors thank Stefan Harmeling for fruitful discussions and providing the code to create Figure 6. We also thank S. Armagan Tarim and Steve Prestwich for contributing cause-effect pairs `pair0094`, `pair0095`, and `pair0096`. Finally, we thank several anonymous reviewers for their comments that helped us to improve the drafts.

Appendix A. Consistency Proof of `ANM-HSIC`

In this Appendix, we prove the consistency of Algorithm 1 with score (6), which is closely related to the algorithm originally proposed by Hoyer et al. (2009) that uses score (4). The main difference is that the original implementation uses the HSIC p-value, whereas here, we use the HSIC value itself as a score. Also, we consider the option of splitting the data set into one part for regression and another part for independence testing. Finally, we fix the HSIC kernel instead of letting its bandwidth be chosen by a heuristic that depends on

the data. The reason that we make these small modifications is that they lead to an easier proof of consistency of the method.

We start with recapitulating the definition and basic properties of the Hilbert Schmidt Independence Criterion (HSIC) in Section A.1. Then, we discuss asymptotic properties of non-parametric regression methods in Section A.2. Finally, we combine these ingredients in Section A.3.

A.1 Consistency of HSIC

We recapitulate the definitions and some asymptotic properties of the Hilbert Schmidt Independence Criterion (HSIC), following mostly the notations and terminology in Gretton et al. (2005). The HSIC estimator that we use here is the original biased estimator proposed by Gretton et al. (2005).

Definition 11 *Given two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ with joint distribution $\mathbb{P}_{X,Y}$, and bounded kernels $k : \mathcal{X}^2 \to \mathbb{R}$ and $l : \mathcal{Y}^2 \to \mathbb{R}$, we define the* ***population HSIC*** *of X and Y as*

$$\begin{aligned}\mathrm{HSIC}_{k,l}(X,Y) :=& \mathbb{E}\big(k(X,X')l(Y,Y')\big) + \mathbb{E}\big(k(X,X')\big)\mathbb{E}\big(l(Y,Y')\big) \\ &- 2\mathbb{E}\Big(\mathbb{E}\big(k(X,X')\,|\,X\big)\mathbb{E}\big(l(Y,Y')\,|\,Y\big)\Big).\end{aligned}$$

Here, (X,Y) and (X',Y') are two independent random variables distributed according to $\mathbb{P}_{X,Y}$.

When k and l are clear from the context, we will typically suppress the dependence of the population HSIC on the choice of the kernels k and l, simply writing $\mathrm{HSIC}(X,Y)$ instead. The justification for the name "independence criterion" stems from the following important result (Fukumizu et al., 2008, Theorem 3):

Lemma 12 *Whenever the product kernel $k \cdot l$ is characteristic (in the sense of Fukumizu et al. (2008); Sriperumbudur et al. (2010)): $\mathrm{HSIC}_{k,l}(X,Y) = 0$ if and only if $X \perp\!\!\!\perp Y$ (i.e., X and Y are independent).* ■

A special case of this lemma, assuming that X and Y have compact domain, was proved originally in Gretton et al. (2005). Recently, Gretton (2015) showed that a similar result also holds if both kernels k and l are characteristic and satisfy some other conditions as well. Intuitively, a characteristic kernel leads to an injective embedding of probability measures into the corresponding Reproducible Kernel Hilbert Space (RKHS). The HSIC is the squared RKHS distance between the embedded joint distribution and the embedded product of the marginals. Given that the embedding is injective, this distance is zero if and only if the variables are independent. Examples of characteristic kernels are Gaussian RBF kernels and Laplace kernels. For more details on the notion of characteristic kernel (see Sriperumbudur et al., 2010). We will use the following (biased) estimator of the population HSIC (Gretton et al., 2005):

Definition 13 *Given two N-tuples (with $N \geq 2$) $\boldsymbol{x} = (x_1, \ldots, x_N) \in \mathcal{X}^N$ and $\boldsymbol{y} = (y_1, \ldots, y_N) \in \mathcal{Y}^N$, and bounded kernels $k : \mathcal{X}^2 \to \mathbb{R}$ and $l : \mathcal{Y}^2 \to \mathbb{R}$, we define*

$$\widehat{\mathrm{HSIC}}_{k,l}(\boldsymbol{x},\boldsymbol{y}) := (N-1)^{-2}\mathrm{tr}(KHLH) = (N-1)^{-2}\sum_{i,j=1}^{N} \bar{K}_{ij}L_{ij}, \tag{27}$$

where $K_{ij} = k(x_i, x_j)$, $L_{ij} = l(y_i, y_j)$ are Gram matrices and $H_{ij} = \delta_{ij} - N^{-1}$ is a centering matrix, and we write $\bar{K} := HKH$ for the centered Gram matrix K. Given an i.i.d. sample $\mathcal{D}_N = \{(x_n, y_n)\}_{n=1}^N$ from $\mathbb{P}_{X,Y}$, we define the ***empirical HSIC*** *of X and Y estimated from $\mathcal{D}_N$ as*

$$\widehat{\mathrm{HSIC}}_{k,l}(X,Y;\mathcal{D}_N) := \widehat{\mathrm{HSIC}}_{k,l}(\boldsymbol{x},\boldsymbol{y}).$$

Again, when k and l are clear from the context, we will typically suppress the dependence of the empirical HSIC on the choice of the kernels k and l. Unbiased estimators of the population HSIC were proposed in later work (Song et al., 2012), but we will not consider those here. A large deviation result for this empirical HSIC estimator is given by Gretton et al. (2005, Theorem 3):

Lemma 14 *Assume that kernels k and l are bounded almost everywhere by 1, and are non-negative. Suppose that the data set $\mathcal{D}_N$ consists of N i.i.d. samples from some joint probability distribution $\mathbb{P}_{X,Y}$. Then, for $N \geq 2$ and all $\delta > 0$, with probability at least $1 - \delta$:*

$$\left|\mathrm{HSIC}_{k,l}(X,Y) - \widehat{\mathrm{HSIC}}_{k,l}(X,Y;\mathcal{D}_N)\right| \leq \sqrt{\frac{\log(6/\delta)}{\alpha^2 N}} + \frac{c}{N},$$

where $\alpha^2 > 0.24$ and c are constants. ■

This directly implies the consistency of the empirical HSIC estimator:[21]

Corollary 15 *Let $(X_1,Y_1),(X_2,Y_2),\ldots$ be i.i.d. according to $\mathbb{P}_{X,Y}$. Defining the sequence of data sets $\mathcal{D}_N = \{(X_n,Y_n)\}_{n=1}^N$ for $N = 2,3,\ldots$, we have for non-negative bounded kernels k,l that, as $N \to \infty$:*

$$\widehat{\mathrm{HSIC}}_{k,l}(X,Y;\mathcal{D}_N) \xrightarrow{P} \mathrm{HSIC}_{k,l}(X,Y).$$

■

We do not know of any results for consistency of HSIC when using adaptive kernel parameters (i.e., when estimating the kernel from the data). This is why we only present a consistency result for fixed kernels here.

For the special case that $\mathcal{Y} = \mathbb{R}$, we will use the following continuity property of the empirical HSIC estimator. It shows that for a Lipschitz-continuous kernel l, the empirical HSIC is also Lipschitz-continuous in the corresponding argument, but with a Lipschitz constant that scales at least as $N^{-1/2}$ for $N \to \infty$. This novel technical result will be the key to our consistency proof of Algorithm 1 with score (6).

Lemma 16 *For all $N \geq 2$, for all $\boldsymbol{x} \in \mathcal{X}^N$, for all $\boldsymbol{y}, \boldsymbol{y}' \in \mathbb{R}^N$, for all bounded kernels $k : \mathcal{X}^2 \to \mathbb{R}$, and for all bounded and Lipschitz-continuous kernels $l : \mathbb{R}^2 \to \mathbb{R}$:*

$$\left|\widehat{\mathrm{HSIC}}(\boldsymbol{x},\boldsymbol{y}) - \widehat{\mathrm{HSIC}}(\boldsymbol{x},\boldsymbol{y}')\right| \leq \frac{32\lambda C}{\sqrt{N}}\left\|\boldsymbol{y} - \boldsymbol{y}'\right\|,$$

where $|k(\xi,\xi')| \leq C$ for all $\xi,\xi' \in \mathcal{X}$ and λ is the Lipschitz constant of l.

21. Let $X_1, X_2, \ldots$ be a sequence of random variables and let X be another random variable. We say that X_n converges to X **in probability**, written $X_n \xrightarrow{P} X$, if

$$\forall \epsilon > 0 : \qquad \lim_{n\to\infty} \mathbb{P}(|X_n - X| > \epsilon) = 0.$$

Proof From (27) it follows that:

$$\left|\widehat{\mathrm{HSIC}}(\boldsymbol{x},\boldsymbol{y}) - \widehat{\mathrm{HSIC}}(\boldsymbol{x},\boldsymbol{y}')\right| = (N-1)^{-2}\left|\sum_{i,j=1}^{N}\bar{K}_{ij}(L_{ij}-L'_{ij})\right|,$$

where $K_{ij} = k(x_i, x_j)$, $L_{ij} = l(y_i, y_j)$, $L'_{ij} = l(y'_i, y'_j)$ and $\bar{K} := HKH$ with $H_{ij} = \delta_{ij} - N^{-1}$. First, note that $|K_{ij}| \leq C$ implies that $\left|\bar{K}_{ij}\right| \leq 4C$:

$$\begin{aligned}\left|\bar{K}_{ij}\right| &= \left|K_{ij} - \frac{1}{N}\sum_{i'=1}^{N}K_{i'j} - \frac{1}{N}\sum_{j'=1}^{N}K_{ij'} + \frac{1}{N^2}\sum_{i',j'=1}^{N}K_{i'j'}\right| \\ &\leq |K_{ij}| + \frac{1}{N}\sum_{i'=1}^{N}\left|K_{i'j}\right| + \frac{1}{N}\sum_{j'=1}^{N}\left|K_{ij'}\right| + \frac{1}{N^2}\sum_{i',j'=1}^{N}\left|K_{i'j'}\right| \leq 4C.\end{aligned}$$

Now starting from the definition and using the triangle inequality:

$$\left|\sum_{i,j=1}^{N}\bar{K}_{ij}(L_{ij}-L'_{ij})\right| \leq \left|\sum_{i,j=1}^{N}\bar{K}_{ij}\big(l(y'_i,y'_j)-l(y'_i,y_j)\big)\right| + \left|\sum_{i,j=1}^{N}\bar{K}_{ij}\big(l(y'_i,y_j)-l(y_i,y_j)\big)\right|.$$

For the first term, using Cauchy-Schwartz (in $\mathbb{R}^{N^2}$) and the Lipschitz property of l:

$$\begin{aligned}\left|\sum_{i,j=1}^{N}\bar{K}_{ij}\big(l(y'_i,y'_j)-l(y'_i,y_j)\big)\right|^2 &\leq \left(\sum_{i,j=1}^{N}\left|\bar{K}_{ij}\right|^2\right)\left(\sum_{i,j=1}^{N}\left|l(y'_i,y'_j)-l(y'_i,y_j)\right|^2\right) \\ &\leq 16N^2C^2\cdot\lambda^2 N\sum_{j=1}^{N}\left|y'_j-y_j\right|^2 \\ &= 16N^3C^2\lambda^2\left\|\boldsymbol{y}'-\boldsymbol{y}\right\|^2.\end{aligned}$$

The second term is similar. The result now follows (using that $\frac{N}{N-1} \leq 2$ for $N \geq 2$). ■

A.2 Consistency of Nonparametric Regression

From now on, we will assume that both X and Y take values in $\mathbb{R}$. Györfi et al. (2002) provide consistency results for several nonparametric regression methods. Here we briefly discuss the main property ("weak universal consistency") that is of particular interest in our setting.

Given a distribution $\mathbb{P}_{X,Y}$, one defines the **regression function** of Y on X as the conditional expectation

$$f(x) := \mathbb{E}(Y \mid X = x).$$

Given an i.i.d. sample of data points $\mathcal{D}_N = \{(x_n, y_n)\}_{n=1}^N$ (the "training" data), a regression method provides an estimate of the regression function $\hat{f}(\,\cdot\,;\mathcal{D}_N)$. The **mean squared error on the training data** (also called "training error") is defined as:

$$\frac{1}{N}\sum_{n=1}^{N}\left|f(x_n)-\hat{f}(x_n;\mathcal{D}_N)\right|^2.$$

The **risk** (also called "generalization error"), i.e., the expected L_2 error on an independent test datum, is defined as:

$$\mathbb{E}_X\left|f(X)-\hat{f}(X;\mathcal{D}_N)\right|^2 = \int\left|f(x)-\hat{f}(x;\mathcal{D}_N)\right|^2 d\mathbb{P}_X(x).$$

Note that the risk is a random variable that depends on the training data $\mathcal{D}_N$.

If the expected risk converges to zero as the number of training points increases, the regression method is called "weakly consistent". More precisely, following Györfi et al. (2002):

Definition 17 *Let $(X_1,Y_1),(X_2,Y_2),\ldots$ be i.i.d. according to $\mathbb{P}_{X,Y}$. Defining training data sets $\mathcal{D}_N = \{(X_n,Y_n)\}_{n=1}^N$ for $N = 2,3,\ldots$, and writing $\mathbb{E}_{\mathcal{D}_N}$ for the expectation value when averaging over $\mathcal{D}_N$, a sequence of estimated regression functions $\hat{f}(\cdot;\mathcal{D}_N)$ is called **weakly consistent for a certain distribution** $\mathbb{P}_{X,Y}$ if*

$$\lim_{N\to\infty}\mathbb{E}_{\mathcal{D}_N}\left(\mathbb{E}_X\left|f(X)-\hat{f}(X;\mathcal{D}_N)\right|^2\right) = 0. \tag{28}$$

*A regression method is called **weakly universally consistent** if it is weakly consistent for all distributions $\mathbb{P}_{X,Y}$ with finite second moment of Y, i.e., with $\mathbb{E}_Y(Y^2) < \infty$.*

Many popular nonparametric regression methods have been shown to be weakly universally consistent (see e.g., Györfi et al., 2002). One might expect naïvely that if the expected risk goes to zero, then also the expected training error should vanish asymptotically:

$$\lim_{N\to\infty}\mathbb{E}_{\mathcal{D}_N}\left(\frac{1}{N}\sum_{n=1}^{N}\left|f(X_n)-\hat{f}(X_n;\mathcal{D}_N)\right|^2\right) = 0. \tag{29}$$

However, property (29) does not necessarily follow from (28).[22] One would expect that asymptotic results on the training error would actually be easier to obtain than results on

22. Here is a counterexample. Suppose that regression method $\hat{f}$ satisfies properties (29) and (28) and that X has bounded density. Given a smooth function $\phi : \mathbb{R} \to \mathbb{R}$ with support $\subset [-1,1]$, $0 \leq \phi(x) \leq 1$, and $\phi(0) = 1$, we now construct a modified sequence of estimated regression functions $\tilde{f}(\,\cdot\,;\mathcal{D}_N)$ that is defined as follows:

$$\tilde{f}(x;\mathcal{D}_N) := \hat{f}(x;\mathcal{D}_N) + \sum_{i=1}^{N}\phi\left(\frac{x-X_i}{\Delta_i^{(N)}}\right),$$

where the $\Delta_i^{(N)}$ should be chosen such that $\Delta_i^{(N)} \leq N^{-2}$ and that the intervals $[X_i - \Delta_i^{(N)}, X_i + \Delta_i^{(N)}]$ are disjoint. Then, we have that

$$\lim_{N\to\infty}\mathbb{E}_{\mathcal{D}_N}\left(\mathbb{E}_X\left|f(X)-\tilde{f}(X;\mathcal{D}_N)\right|^2\right) = 0,$$

but on the other hand,

$$\lim_{N\to\infty}\mathbb{E}_{\mathcal{D}_N}\frac{1}{N}\sum_{n=1}^{N}\left|f(X_n)-\tilde{f}(X_n;\mathcal{D}_N)\right|^2 = 1.$$

generalization error. One result that we found in the literature is Lemma 5 in Kpotufe et al. (2014), which states that a certain box kernel regression method satisfies (29) under certain assumptions on the distribution $\mathbb{P}_{X,Y}$. The reason that we bring this up at this point is that property (29) allows to prove consistency even when one uses the same data for both regression and independence testing (see also Lemma 19).

From now on, we will always consider the following setting. Let $(\tilde{X}_1, \tilde{Y}_1)$, $(\tilde{X}_2, \tilde{Y}_2)$, ... be i.i.d. according to some joint distribution $\mathbb{P}_{X,Y}$. We distinguish two different scenarios:

- "Data splitting": using half of the data for training, and the other half of the data for testing. In particular, we define $X_n := \tilde{X}_{2n-1}$, $Y_n := \tilde{Y}_{2n-1}$, $X'_n := \tilde{X}_{2n}$ and $Y'_n := \tilde{Y}_{2n}$ for $n = 1, 2, \ldots$.

- "Data recycling": using the same data both for regression and for testing. In particular, we define $X_n := \tilde{X}_n$, $Y_n := \tilde{Y}_n$, $X'_n := \tilde{X}_n$ and $Y'_n := \tilde{Y}_n$ for $n = 1, 2, \ldots$.

In both scenarios, for $N = 1, 2, \ldots$, we define a sequence of training data sets $\mathcal{D}_N := \{(X_n, Y_n)\}_{n=1}^N$ (for the regression) and a sequence of test data sets $\mathcal{D}'_N := \{(X'_n, Y'_n)\}_{n=1}^N$ (for testing independence of residuals). Note that in the data recycling scenario, training and test data are identical, whereas in the data splitting scenario, training and test data are independent.

Define a random variable (the "residual")

$$E := Y - f(X) = Y - \mathbb{E}(Y \mid X), \tag{30}$$

and its vector-valued versions on the test data:

$$\boldsymbol{E}'_{\ldots N} := \big(Y'_1 - f(X'_1), \ldots, Y'_N - f(X'_N)\big),$$

called the **true residuals**. Using a regression method, we obtain an estimate $\hat{f}(x; \mathcal{D}_N)$ for the regression function $f(x) = \mathbb{E}(Y \mid X = x)$ from the training data $\mathcal{D}_N$. We then define an estimate of the vector-valued version of E on the test data:

$$\hat{\boldsymbol{E}}'_{\ldots N} := \big(Y'_1 - \hat{f}(X'_1; \mathcal{D}_N), \ldots, Y'_N - \hat{f}(X'_N; \mathcal{D}_N)\big), \tag{31}$$

called the **predicted residuals**.

Definition 18 *We call the regression method **suitable** for regressing Y on X if the mean squared error between true and predicted residuals vanishes asymptotically in expectation:*

$$\lim_{N\to\infty} \mathbb{E}_{\mathcal{D}_N, \mathcal{D}'_N} \left(\frac{1}{N} \left\| \hat{\boldsymbol{E}}'_{\ldots N} - \boldsymbol{E}'_{\ldots N} \right\|^2 \right) = 0. \tag{32}$$

Here, the expectation is taken over both training data $\mathcal{D}_N$ and test data $\mathcal{D}'_N$.

Lemma 19 *In the data splitting case, any regression method that is weakly consistent for $\mathbb{P}_{X,Y}$ is suitable. In the data recycling case, any regression method satisfying property (29) is suitable.*

Proof Simply rewriting:

$$\lim_{N\to\infty} \mathbb{E} \left(\frac{1}{N} \left\| \hat{\boldsymbol{E}}'_{\ldots N} - \boldsymbol{E}'_{\ldots N} \right\|^2 \right) =$$
$$\lim_{N\to\infty} \mathbb{E} \left(\frac{1}{N} \sum_{n=1}^N \left| \big(Y'_n - \hat{f}(X'_n; \mathcal{D}_N)\big) - \big(Y'_n - f(X'_n)\big) \right|^2 \right) =$$
$$\lim_{N\to\infty} \mathbb{E}_{\mathcal{D}_N, \mathcal{D}'_N} \left(\frac{1}{N} \sum_{n=1}^N \left| \hat{f}(X'_n; \mathcal{D}_N) - f(X'_n) \right|^2 \right).$$

Therefore, (32) reduces to (28) in the data splitting scenario (where each X'_n is an independent copy of X), and reduces to (29) in the data recycling scenario (where $X'_n = X_n$). ■

In particular, if $\mathbb{E}(X^2) < \infty$ and $\mathbb{E}(Y^2) < \infty$, any weakly universally consistent regression method is suitable both for regressing X on Y and Y on X in the data splitting scenario.

A.3 Consistency of `ANM-HSIC`

We can now prove our main result, stating that the empirical HSIC calculated from the test set inputs and the predicted residuals on the test set (using the regression function estimated from the training set) converges in probability to the population HSIC of the true inputs and the true residuals:

Theorem 20 *Let $X, Y \in \mathbb{R}$ be two random variables with joint distribution $\mathbb{P}_{X,Y}$. Let $k, l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be two bounded non-negative kernels and assume that l is Lipschitz continuous. Suppose we are given sequences of training data sets $\mathcal{D}_N$ and test data sets $\mathcal{D}'_N$ (in either the data splitting or the data recycling scenario described above). Suppose we use a suitable regression procedure (c.f. Lemma 19), to obtain a sequence $\hat{f}(x; \mathcal{D}_N)$ of estimates of the regression function $\mathbb{E}(Y \mid X = x)$ from the training data. Defining the true residual E by (30), and the predicted residuals $\hat{\boldsymbol{E}}'_{\ldots N}$ on the test data as in (31), then, for $N \to \infty$:*

$$\widehat{\mathrm{HSIC}}_{k,l}(\boldsymbol{X}'_{\ldots N}, \hat{\boldsymbol{E}}'_{\ldots N}) \xrightarrow{P} \mathrm{HSIC}_{k,l}(X, E).$$

Proof We start by applying Lemma 16:

$$\left| \widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \hat{\boldsymbol{E}}'_{\ldots N}) - \widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \boldsymbol{E}'_{\ldots N}) \right|^2 \leq \left(\frac{32\lambda C}{\sqrt{N}} \right)^2 \left\| \hat{\boldsymbol{E}}'_{\ldots N} - \boldsymbol{E}'_{\ldots N} \right\|^2,$$

where λ and C are constants. From the suitability of the regression method, (32), it therefore follows that

$$\lim_{N\to\infty} \mathbb{E}_{\mathcal{D}_N, \mathcal{D}'_N} \left| \widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \hat{\boldsymbol{E}}'_{\ldots N}) - \widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \boldsymbol{E}'_{\ldots N}) \right|^2 = 0,$$

i.e.,

$$\widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \hat{\boldsymbol{E}}'_{\ldots N}) - \widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \boldsymbol{E}'_{\ldots N}) \xrightarrow{L_2} 0.$$

As convergence in L_2 implies convergence in probability (see, e.g, Wasserman, 2004),

$$\widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \hat{\boldsymbol{E}}'_{\ldots N}) - \widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \boldsymbol{E}'_{\ldots N}) \xrightarrow{P} 0.$$

From the consistency of the empirical HSIC, Corollary 15:

$$\widehat{\mathrm{HSIC}}(\boldsymbol{X}'_{\ldots N}, \boldsymbol{E}'_{\ldots N}) \xrightarrow{P} \mathrm{HSIC}(X, E).$$

Hence, by taking sums (see e.g., Theorem 5.5 in Wasserman, 2004), we arrive at the desired statement. ■

We are now ready to show that Algorithm 1 with score (6) (which is the special case $k = l$) is consistent.

Corollary 21 *Let X, Y be two real-valued random variables with joint distribution $\mathbb{P}_{X,Y}$ that either satisfies an Additive Noise Model $X \to Y$, or $Y \to X$, but not both. Suppose we are given sequences of training data sets $\mathcal{D}_N$ and test data sets $\mathcal{D}'_N$ (in either the data splitting or the data recycling scenario). Let $k, l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be two bounded non-negative Lipschitz-continuous kernels such that their product $k \cdot l$ is characteristic. If the regression procedure used in Algorithm 1 is suitable for both $\mathbb{P}_{X,Y}$ and $\mathbb{P}_{Y,X}$, then Algorithm 1 with score (6) is a consistent procedure for estimating the direction of the Additive Noise Model.*

Proof Define "population residuals" $E_Y := Y - \mathbb{E}(Y \,|\, X)$ and $E_X := X - \mathbb{E}(X \,|\, Y)$. Note that $\mathbb{P}_{X,Y}$ satisfies a bivariate Additive Noise Model $X \to Y$ if and only if $E_Y \perp\!\!\!\perp X$ (c.f. Lemma 7). Further, by Lemma 12, we have $\mathrm{HSIC}_{k,l}(X, E_Y) = 0$ if and only if $X \perp\!\!\!\perp E_Y$. Similarly, $\mathbb{P}_{X,Y}$ satisfies a bivariate Additive Noise Model $Y \to X$ if and only if $\mathrm{HSIC}_{l,k}(Y, E_X) = 0$.

Now, by Theorem 20,

$$\hat{C}_{X\to Y} := \widehat{\mathrm{HSIC}}_{k,l}(\boldsymbol{X}'_{\ldots N}, \hat{\boldsymbol{E}}_Y(\mathcal{D}'_N; \mathcal{D}_N)) \xrightarrow{P} \mathrm{HSIC}_{k,l}(X, E_Y),$$

and similarly

$$\hat{C}_{Y\to X} := \widehat{\mathrm{HSIC}}_{l,k}(\boldsymbol{Y}'_{\ldots N}, \hat{\boldsymbol{E}}_X(\mathcal{D}'_N; \mathcal{D}_N)) \xrightarrow{P} \mathrm{HSIC}_{l,k}(Y, E_X),$$

where the predicted residuals are defined by

$$\hat{\boldsymbol{E}}_Y(\mathcal{D}'_N; \mathcal{D}_N) := \big(Y'_1 - \hat{f}_Y(X'_1; \mathcal{D}_N), \ldots, Y'_N - \hat{f}_Y(X'_N; \mathcal{D}_N)\big),$$
$$\hat{\boldsymbol{E}}_X(\mathcal{D}'_N; \mathcal{D}_N) := \big(X'_1 - \hat{f}_X(Y'_1; \mathcal{D}_N), \ldots, X'_N - \hat{f}_X(Y'_N; \mathcal{D}_N)\big),$$

with estimates $\hat{f}_Y(x; \mathcal{D}_N), \hat{f}_X(y; \mathcal{D}_N)$ of the regression functions $\mathbb{E}(Y \,|\, X = x), \mathbb{E}(X \,|\, Y = y)$ from the training data $\mathcal{D}_N$.

Because $\mathbb{P}_{X,Y}$ satisfies an Additive Noise Model only in one of the two directions, this implies that either $\mathrm{HSIC}_{k,l}(X, E_Y) = 0$ and $\mathrm{HSIC}_{l,k}(Y, E_X) > 0$ (corresponding with $X \to Y$), or $\mathrm{HSIC}_{k,l}(X, E_Y) > 0$ and $\mathrm{HSIC}_{l,k}(Y, E_X) = 0$ (corresponding with $Y \to X$). Therefore the test procedure is consistent. ■

Appendix B. Relationship Between Scores (10) and (11)

For the special case of an Additive Noise Model $X \to Y$, the empirical-Bayes score proposed by Friedman and Nachman (2000) is given in (11):

$$\hat{C}_{X\to Y} = \min_{\mu,\tau^2,\boldsymbol{\theta},\sigma^2} \big(-\log \mathcal{N}(\boldsymbol{x} \,|\, \mu\mathbf{1}, \tau^2\boldsymbol{I}) - \log \mathcal{N}(\boldsymbol{y} \,|\, \mathbf{0}, \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}) + \sigma^2\boldsymbol{I})\big).$$

It is a sum of the negative log likelihood of a Gaussian model for the inputs:

$$\begin{aligned} &\min_{\mu,\tau^2} \big(-\log \mathcal{N}(\boldsymbol{x} \,|\, \mu\mathbf{1}, \tau^2\boldsymbol{I})\big) \\ &= \min_{\mu,\tau^2} \left(\frac{N}{2}\log(2\pi\tau^2) + \frac{1}{2\tau^2}\sum_{i=1}^N (x_i - \mu)^2\right) \\ &= \frac{N}{2}\log(2\pi e) + \frac{N}{2}\log\left(\frac{1}{N}\sum_{i=1}^N (x_i - \bar{x})^2\right) \end{aligned} \tag{33}$$

with $\bar{x} := \frac{1}{N}\sum_{i=1}^N x_i$, and the negative log marginal likelihood of a GP model for the outputs, given the inputs:

$$\begin{aligned} &\min_{\boldsymbol{\theta},\sigma^2} \big(-\log \mathcal{N}(\boldsymbol{y} \,|\, 0, \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}) + \sigma^2\boldsymbol{I})\big) \\ &= \min_{\boldsymbol{\theta},\sigma^2} \left(\frac{N}{2}\log(2\pi) + \frac{1}{2}\log|\det(\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}) + \sigma^2\boldsymbol{I})| + \boldsymbol{y}^T(\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}) + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}\right). \end{aligned} \tag{34}$$

Note that (33) is an empirical estimator of the entropy of a Gaussian with variance $\mathbb{V}\mathrm{ar}(X)$, up to a factor N:

$$H(X) = \frac{1}{2}\log(2\pi e) + \frac{1}{2}\log \mathbb{V}\mathrm{ar}(X).$$

We will show that (34) is closely related to an empirical estimator of the entropy of the residuals $Y - \mathbb{E}(Y \,|\, X)$:

$$H(Y - \mathbb{E}(Y \,|\, X)) = \frac{1}{2}\log(2\pi e) + \frac{1}{2}\log \mathbb{V}\mathrm{ar}(Y - \mathbb{E}(Y \,|\, X)).$$

This means that the score (11) considered by Friedman and Nachman (2000) is closely related to the Gaussian score (10) for $X \to Y$:

$$\hat{C}_{X\to Y} = \log \mathbb{V}\mathrm{ar}(X) + \log \mathbb{V}\mathrm{ar}(Y - \hat{f}_Y(X)).$$

The following Lemma shows that standard Gaussian Process regression can be interpreted as a penalized maximum likelihood optimization.

Lemma 22 *Let $\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x})$ be the kernel matrix (abbreviated as $\boldsymbol{K}$) and define a negative penalized log-likelihood as:*

$$-\log \mathcal{L}(\boldsymbol{f}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}) := \underbrace{\frac{N}{2}\log(2\pi\sigma^2) + \frac{1}{2\sigma^2}\sum_{i=1}^N (y_i - f_i)^2}_{\textit{Likelihood}} + \underbrace{\frac{1}{2}\boldsymbol{f}^T\boldsymbol{K}^{-1}\boldsymbol{f} + \frac{1}{2}\log\left|\det\left(\boldsymbol{I} + \sigma^{-2}\boldsymbol{K}\right)\right|}_{\textit{Penalty}}. \tag{35}$$

Minimizing with respect to $\boldsymbol{f}$ yields a minimum at

$$\hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}} = \operatorname*{argmin}_{\boldsymbol{f}} \left(-\log \mathcal{L}(\boldsymbol{f}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}_{\boldsymbol{\theta}})\right) = \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2 \boldsymbol{I})^{-1}\boldsymbol{y}, \tag{36}$$

and the value at the minimum is given by

$$\min_{\boldsymbol{f}} \left(-\log \mathcal{L}(\boldsymbol{f}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}_{\boldsymbol{\theta}})\right) = -\log \mathcal{L}(\hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}_{\boldsymbol{\theta}}) = -\log \mathcal{N}(\boldsymbol{y} \,|\, \boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2 \boldsymbol{I}). \tag{37}$$

Proof Because $\boldsymbol{B}(\boldsymbol{A}^{-1} + \boldsymbol{B}^{-1})\boldsymbol{A} = \boldsymbol{A} + \boldsymbol{B}$ for invertible (equally-sized square) matrices $\boldsymbol{A}, \boldsymbol{B}$, the following identity holds:

$$(\boldsymbol{A}^{-1} + \boldsymbol{B}^{-1})^{-1} = \boldsymbol{A}(\boldsymbol{A} + \boldsymbol{B})^{-1}\boldsymbol{B}.$$

Substituting $\boldsymbol{A} = \boldsymbol{K}$ and $\boldsymbol{B} = \sigma^2 \boldsymbol{I}$, we obtain directly that

$$(\boldsymbol{K}^{-1} + \sigma^{-2}\boldsymbol{I})^{-1} = \boldsymbol{K}(\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1}\sigma^2. \tag{38}$$

By taking log-determinants, it also follows that

$$\log|\det \boldsymbol{K}| + \log|\det(\boldsymbol{K}^{-1} + \sigma^{-2}\boldsymbol{I})| = \log|\det\left(\boldsymbol{I} + \sigma^{-2}\boldsymbol{K}\right)|.$$

Therefore, we can rewrite (35) as follows:

$$\mathcal{L}(\boldsymbol{f}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K})\mathcal{N}(\boldsymbol{y} - \boldsymbol{f}|\boldsymbol{0}, \sigma^2\boldsymbol{I}) \left|\det\left(\boldsymbol{K}^{-1} + \sigma^{-2}\boldsymbol{I}\right)\right|^{-1/2} (2\pi)^{N/2}. \tag{39}$$

Equation (A.7) in Rasmussen and Williams (2006) for the product of two Gaussians states that

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{a}, \boldsymbol{A})\mathcal{N}(\boldsymbol{x}|\boldsymbol{b}, \boldsymbol{B}) = \mathcal{N}(\boldsymbol{a}|\boldsymbol{b}, \boldsymbol{A} + \boldsymbol{B})\mathcal{N}(\boldsymbol{x}|\boldsymbol{c}, \boldsymbol{C}),$$

where $\boldsymbol{C} = (\boldsymbol{A}^{-1} + \boldsymbol{B}^{-1})^{-1}$ and $\boldsymbol{c} = \boldsymbol{C}(\boldsymbol{A}^{-1}\boldsymbol{a} + \boldsymbol{B}^{-1}\boldsymbol{b})$. Substituting $\boldsymbol{x} = \boldsymbol{f}$, $\boldsymbol{a} = \boldsymbol{0}$, $\boldsymbol{A} = \boldsymbol{K}$, $\boldsymbol{b} = \boldsymbol{y}$, and $\boldsymbol{B} = \sigma^2\boldsymbol{I}$, and using (38), this gives:

$$\mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K})\mathcal{N}(\boldsymbol{y} - \boldsymbol{f}|\boldsymbol{0}, \sigma^2\boldsymbol{I}) = \mathcal{N}(\boldsymbol{y} \,|\, \boldsymbol{0}, \boldsymbol{K} + \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}}, (\boldsymbol{K}^{-1} + \sigma^{-2}\boldsymbol{I})^{-1}),$$

where

$$\hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}} := \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}.$$

Therefore, we can rewrite (39) as:

$$\begin{aligned}&\mathcal{L}(\boldsymbol{f}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}) \\ &\quad = \mathcal{N}(\boldsymbol{y} \,|\, \boldsymbol{0}, \boldsymbol{K} + \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}}, (\boldsymbol{K}^{-1} + \sigma^{-2}\boldsymbol{I})^{-1}) \left|\det\left(\boldsymbol{K}^{-1} + \sigma^{-2}\boldsymbol{I}\right)\right|^{-1/2} (2\pi)^{N/2}.\end{aligned}$$

It is now obvious that the penalized likelihood is maximized for $\boldsymbol{f} = \hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}}$ (for fixed hyperparameters $\sigma, \boldsymbol{\theta}$) and that at the maximum, it has the value

$$\mathcal{L}(\hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}}, \sigma^2; \boldsymbol{y}, \boldsymbol{K}_{\boldsymbol{\theta}}) = \mathcal{N}(\boldsymbol{y} \,|\, \boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2\boldsymbol{I}).$$

■

Note that the estimated function (36) is identical to the mean posterior GP, and the value (37) is identical to the negative logarithm of the marginal likelihood (evidence) of the data according to the GP model (Rasmussen and Williams, 2006).

Making use of Lemma 22, the conditional part (34) in score (11) can be rewritten as:

$$\begin{aligned}&\min_{\sigma^2,\boldsymbol{\theta}} \left(-\log \mathcal{N}(\boldsymbol{y} \,|\, \boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2\boldsymbol{I})\right) \\ &= \min_{\sigma^2,\boldsymbol{\theta}} \frac{1}{2}\left(N\log(2\pi\sigma^2) + \frac{1}{\sigma^2}\sum_{i=1}^{N}(y_i - (\hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}})_i)^2 + \hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}}^T \boldsymbol{K}_{\boldsymbol{\theta}}^{-1}\hat{\boldsymbol{f}}_{\sigma,\boldsymbol{\theta}} + \log|\det\left(\boldsymbol{I} + \sigma^{-2}\boldsymbol{K}_{\boldsymbol{\theta}}\right)|\right) \\ &= \underbrace{\frac{N}{2}\log(2\pi\hat{\sigma}^2) + \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{N}(y_i - (\hat{\boldsymbol{f}})_i)^2}_{\text{Likelihood term}} + \underbrace{\frac{1}{2}\hat{\boldsymbol{f}}^T\boldsymbol{K}_{\hat{\boldsymbol{\theta}}}^{-1}\hat{\boldsymbol{f}} + \frac{1}{2}\log|\det\left(\boldsymbol{I} + \hat{\sigma}^{-2}\boldsymbol{K}_{\hat{\boldsymbol{\theta}}}\right)|}_{\text{Complexity penalty}},\end{aligned}$$

where $\hat{\boldsymbol{f}} := \hat{\boldsymbol{f}}_{\hat{\sigma},\hat{\boldsymbol{\theta}}}$ for the minimizing $(\hat{\sigma}, \hat{\boldsymbol{\theta}})$. If the complexity penalty is small compared to the likelihood term around the optimal values $(\hat{\sigma}, \hat{\boldsymbol{\theta}})$, we can approximate:

$$\begin{aligned}&\min_{\sigma^2,\boldsymbol{\theta}} \left(-\log \mathcal{N}(\boldsymbol{y} \,|\, \boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2\boldsymbol{I})\right) \\ &\approx \frac{N}{2}\log(2\pi\hat{\sigma}^2) + \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{N}(y_i - \hat{f}_i)^2 \\ &\approx \min_{\sigma^2}\left(\frac{N}{2}\log(2\pi\sigma^2) + \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \hat{f}_i)^2\right) \\ &= \frac{N}{2}\log(2\pi e) + \frac{N}{2}\log\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}_i)^2\right).\end{aligned}$$

This shows that there is a close relationship between the two scores (11) and (10).

Appendix C. Details on the Simulated Data

Here we give more details on how the data were simulated. The simulated data itself is provided as supplementary material on the first author's website.

C.1 Sampling from a Random Density

We first describe how we sample from a random density. First, we sample $\boldsymbol{X} \in \mathbb{R}^N$ from a standard-normal distribution:

$$\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}_N, \boldsymbol{I}_N),$$

and define $\overrightarrow{\boldsymbol{X}}$ to be the vector that is obtained by sorting $\boldsymbol{X}$ in ascending order. Then, we sample a realization $\boldsymbol{F}$ of a Gaussian Process with inputs $\overrightarrow{\boldsymbol{X}}$, using a kernel with hyperparameters $\boldsymbol{\theta}$ and white noise with standard deviation σ:

$$\boldsymbol{F} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{\theta}}(\overrightarrow{\boldsymbol{X}}) + \sigma^2\boldsymbol{I}),$$

where $\boldsymbol{K}_{\boldsymbol{\theta}}(\overrightarrow{\boldsymbol{X}})$ is the Gram matrix for $\overrightarrow{\boldsymbol{X}}$ using kernel $k_{\boldsymbol{\theta}}$. We use the trapezoidal rule to calculate the cumulative integral of the function $e^F : \mathbb{R} \to \mathbb{R}$ that linearly interpolates the points $(\overrightarrow{\boldsymbol{X}}, \exp(\boldsymbol{F}))$. In this way, we obtain a vector $\boldsymbol{G} \in \mathbb{R}^N$ where each element G_i corresponds to $\int_{\overrightarrow{X}_1}^{\overrightarrow{X}_i} e^F(x)\,dx$. As covariance function, we used the Gaussian kernel:

$$k_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\sum_{i=1}^{D} \frac{(x_i - x_i')^2}{\theta_i^2}\right).$$

We will denote this whole sampling procedure by:

$$\boldsymbol{G} \sim \mathcal{RD}(\boldsymbol{\theta}, \sigma).$$

C.2 Sampling Cause-Effect Pairs

We simulate cause-effect pairs as follows. First, we sample three noise variables:

$$\begin{aligned} W_{E_X} &\sim \Gamma(a_{W_{E_X}}, b_{W_{E_X}}) & \boldsymbol{E}_X &\sim \mathcal{RD}(W_{E_X}, \tau) \\ W_{E_Y} &\sim \Gamma(a_{W_{E_Y}}, b_{W_{E_Y}}) & \boldsymbol{E}_Y &\sim \mathcal{RD}(W_{E_X}, \tau) \\ W_{E_Z} &\sim \Gamma(a_{W_{E_Z}}, b_{W_{E_Z}}) & \boldsymbol{E}_Z &\sim \mathcal{RD}(W_{E_Z}, \tau) \end{aligned}$$

where each noise variable has a random characteristic length scale. We then standardize each noise sample $\boldsymbol{E}_X$, $\boldsymbol{E}_Y$ and $\boldsymbol{E}_Z$.

If there is no confounder, we sample $\boldsymbol{X}$ from a GP with inputs $\boldsymbol{E}_X$:

$$\begin{aligned} S_{E_X} &\sim \Gamma(a_{S_{E_X}}, b_{S_{E_X}}) \\ \boldsymbol{X} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{S_{E_X}}(\boldsymbol{E}_X) + \tau^2 \boldsymbol{I}) \end{aligned}$$

and then we standardize $\boldsymbol{X}$. Then, we sample $\boldsymbol{Y}$ from a GP with inputs $(\boldsymbol{X}, \boldsymbol{E}_Y) \in \mathbb{R}^{N \times 2}$:

$$\begin{aligned} S_X &\sim \Gamma(a_{S_X}, b_{S_X}) \\ S_{E_Y} &\sim \Gamma(a_{S_{E_Y}}, b_{S_{E_Y}}) \\ \boldsymbol{Y} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{(S_X, S_{E_Y})}((\boldsymbol{X}, \boldsymbol{E}_Y)) + \tau^2 \boldsymbol{I}) \end{aligned}$$

and then we standardize $\boldsymbol{Y}$.

If there is a confounder, we sample $\boldsymbol{X}$ from a GP with inputs $(\boldsymbol{E}_X, \boldsymbol{E}_Z) \in \mathbb{R}^{N \times 2}$:

$$\begin{aligned} S_{E_X} &\sim \Gamma(a_{S_{E_X}}, b_{S_{E_X}}) \\ S_{E_Z} &\sim \Gamma(a_{S_{E_Z}}, b_{S_{E_Z}}) \\ \boldsymbol{X} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{(S_{E_X}, S_{E_Z})}((\boldsymbol{E}_X, \boldsymbol{E}_Z)) + \tau^2 \boldsymbol{I}) \end{aligned}$$

and then we standardize $\boldsymbol{X}$. Then, we sample $\boldsymbol{Y}$ from a GP with inputs $(\boldsymbol{X}, \boldsymbol{E}_Y, \boldsymbol{E}_Z) \in \mathbb{R}^{N \times 3}$:

$$\begin{aligned} S_X &\sim \Gamma(a_{S_X}, b_{S_X}) \\ S_{E_Y} &\sim \Gamma(a_{S_{E_Y}}, b_{S_{E_Y}}) \\ S_{E_Z} &\sim \Gamma(a_{S_{E_Z}}, b_{S_{E_Z}}) \\ \boldsymbol{Y} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_{(S_X, S_{E_Y}, S_{E_Z})}((\boldsymbol{X}, \boldsymbol{E}_Y, \boldsymbol{E}_Z)) + \tau^2 \boldsymbol{I}) \end{aligned}$$

Scenario	$(a_{W_{E_X}}, b_{W_{E_X}})$	$(a_{S_{E_X}}, b_{S_{E_X}})$	$(a_{S_{E_Y}}, b_{S_{E_Y}})$	$(a_{S_{M_X}}, b_{S_{M_X}})$	$(a_{S_{M_Y}}, b_{S_{M_Y}})$
`SIM`	$(5, 0.1)$	$(2, 1.5)$	$(2, 15)$	$(2, 0.1)$	$(2, 0.1)$
`SIM-c`	$(5, 0.1)$	$(2, 1.5)$	$(2, 15)$	$(2, 0.1)$	$(2, 0.1)$
`SIM-ln`	$(5, 0.1)$	$(2, 1.5)$	$(2, 300)$	$(2, 0.01)$	$(2, 0.01)$
`SIM-G`	$(10^6, 10^{-3})$	$(10^6, 10^{-3})$	$(2, 15)$	$(2, 0.1)$	$(2, 0.1)$

Table 3: Parameter settings used to simulate cause-effect pairs for four scenarios. `SIM-c` has a confounder, the other scenarios have no confounders. The common parameters for the four scenarios are: $\tau = 10^{-4}$, $(a_{WnE_Y}, b_{W_{E_Y}}) = (5, 0.1)$, $(a_{W_{E_Z}}, b_{W_{E_Z}}) = (5, 0.1)$, $(a_{S_{E_Z}}, b_{S_{E_Z}}) = (2, 15)$, $(a_{S_X}, b_{S_X}) = (2, 15)$.

and then we standardize $\boldsymbol{Y}$.

Finally, we add measurement noise:

$$\begin{aligned} S_{M_X} &\sim \Gamma(a_{S_{M_X}}, b_{S_{M_X}}) \\ \boldsymbol{M}_X &\sim \mathcal{N}(\boldsymbol{0}, S_{M_X}^2 \boldsymbol{I}) \\ \boldsymbol{X} &\leftarrow \boldsymbol{X} + \boldsymbol{M}_X \\ S_{M_Y} &\sim \Gamma(a_{S_{M_Y}}, b_{S_{M_Y}}) \\ \boldsymbol{M}_Y &\sim \mathcal{N}(\boldsymbol{0}, S_{M_Y}^2 \boldsymbol{I}) \\ \boldsymbol{Y} &\leftarrow \boldsymbol{Y} + \boldsymbol{M}_Y \end{aligned}$$

We considered the four scenarios in Table 3: `SIM`, a scenario without confounders; `SIM-c`, a similar scenario but with one confounder; `SIM-ln`, a scenario with low noise levels (for which we expect IGCI to perform well); `SIM-G`, a scenario with a distribution of X that is almost Gaussian. We used $N = 1000$ samples for each pair, and simulated 100 cause-effect pairs for each scenario.

Appendix D. Description of the CauseEffectPairs Benchmark

The CauseEffectPairs benchmark set described here is an extension of the collection of the eight data sets that formed the CauseEffectPairs task in the *Causality Challenge #2: Pot-Luck* competition (Mooij and Janzing, 2010) that was performed as part of the NIPS 2008 Workshop on Causality (Guyon et al., 2010).[23] Here we describe version 1.0 of the CauseEffectPairs benchmark, which consists of 100 "cause-effect pairs", each one consisting of samples of a pair of statistically dependent random variables, where one variable is known to cause the other one. The task is to identify for each pair which of the two variables is the cause and which one the effect, using the observed samples only. The data are publicly available at Mooij et al. (2014).

The data sets were selected such that we expect common agreement on the ground truth. For example, the first pair consists of measurements of altitude and mean annual temperature of more than 300 weather stations in Germany. It should be obvious that altitude causes temperature rather than the other way around. Even though part of the

23. The introduction of this section and the descriptions of these 8 pairs are heavily based on Mooij and Janzing (2010).

statistical dependences may also be due to hidden common causes and selection bias, we expect that there is a significant cause-effect relation between the two variables in each pair, based on our understanding of the data generating process.

The best way to decide upon the ground truth of the causal relationships in the systems that generated the data would be by performing interventions on one of the variables and observing whether the intervention changes the distribution of the other variable. Unfortunately, these interventions cannot be performed in practice for many of the existing pairs because the original data-generating system is no longer available, or because of other practical reasons. Therefore, we have selected data sets in which the causal direction should be clear from the meanings of the variables and the way in which the data were generated. Unfortunately, for many data sets that are publicly available, it is not always clearly documented exactly how the variables are defined and measured.

In selecting the cause-effect pair data sets, we applied the following criteria:

- The minimum number of samples per pair should be a few hundred;
- The variables should have values in $\mathbb{R}^d$ for some $d = 1, 2, 3, \ldots$;
- There should be a significant cause–effect relationship between the two variables;
- The direction of the causal relationship should be known or obvious from the meaning of the variables.

Version 1.0 of the CauseEffectPairs collection consists of 100 pairs satisfying these criteria, taken from 37 different data sets from different domains. We refer to these pairs as `pair0001`, ..., `pair00100`. Table 4 gives an overview of the cause-effect pairs. In the following subsections, we describe the cause-effect pairs in detail, and motivate our decisions on the causal relationships present in the pairs. We provide a scatter plot for each pair, where the horizontal axis corresponds with the cause, and the vertical axis with the effect. For completeness, we describe all the pairs in the data set, including those that have been described before in Mooij and Janzing (2010).

Pair	Variable 1	Variable 2	Data Set	Ground Truth	Weight
pair0001	Altitude	Temperature	D.1	→	1/6
pair0002	Altitude	Precipitation	D.1	→	1/6
pair0003	Longitude	Temperature	D.1	→	1/6
pair0004	Altitude	Sunshine hours	D.1	→	1/6
pair0005	Age	Length	D.2	→	1/7
pair0006	Age	Shell weight	D.2	→	1/7
pair0007	Age	Diameter	D.2	→	1/7
pair0008	Age	Height	D.2	→	1/7
pair0009	Age	Whole weight	D.2	→	1/7
pair0010	Age	Shucked weight	D.2	→	1/7
pair0011	Age	Viscera weight	D.2	→	1/7
pair0012	Age	Wage per hour	D.3	→	1/2
pair0013	Displacement	Fuel consumption	D.4	→	1/4
pair0014	Horse power	Fuel consumption	D.4	→	1/4
pair0015	Weight	Fuel consumption	D.4	→	1/4
pair0016	Horsepower	Acceleration	D.4	→	1/4
pair0017	Age	Dividends from stocks	D.3	→	1/2
pair0018	Age	Concentration GAG	D.5	→	1
pair0019	Current duration	Next interval	D.6	→	1
pair0020	Latitude	Temperature	D.1	→	1/6
pair0021	Longitude	Precipitation	D.1	→	1/6
pair0022	Age	Height	D.7	→	1/3
pair0023	Age	Weight	D.7	→	1/3
pair0024	Age	Heart rate	D.7	→	1/3

(Table continues on next page)

Pair	Variable 1	Variable 2	Data Set	Ground Truth	Weight
pair0025	Cement	Compressive strength	D.8	→	1/8
pair0026	Blast furnace slag	Compressive strength	D.8	→	1/8
pair0027	Fly ash	Compressive strength	D.8	→	1/8
pair0028	Water	Compressive strength	D.8	→	1/8
pair0029	Superplasticizer	Compressive strength	D.8	→	1/8
pair0030	Coarse aggregate	Compressive strength	D.8	→	1/8
pair0031	Fine aggregate	Compressive strength	D.8	→	1/8
pair0032	Age	Compressive strength	D.8	→	1/8
pair0033	Alcohol consumption	Mean corpuscular volume	D.9	→	1/5
pair0034	Alcohol consumption	Alkaline phosphotase	D.9	→	1/5
pair0035	Alcohol consumption	Alanine aminotransferase	D.9	→	1/5
pair0036	Alcohol consumption	Aspartate aminotransferase	D.9	→	1/5
pair0037	Alcohol consumption	Gamma-glutamyl transpeptidase	D.9	→	1/5
pair0038	Age	Body mass index	D.10	→	1/4
pair0039	Age	Serum insulin	D.10	→	1/4
pair0040	Age	Diastolic blood pressure	D.10	→	1/4
pair0041	Age	Plasma glucose concentration	D.10	→	1/4
pair0042	Day of the year	Temperature	D.11	→	1/2
pair0043	Temperature at t	Temperature at $t+1$	D.12	→	1/4
pair0044	Surface pressure at t	Surface pressure at $t+1$	D.12	→	1/4
pair0045	Sea level pressure at t	Sea level pressure at $t+1$	D.12	→	1/4
pair0046	Relative humidity at t	Relative humidity at $t+1$	D.12	→	1/4
pair0047	Number of cars	Type of day	D.13	←	1
pair0048	Indoor temperature	Outdoor temperature	D.14	←	1
pair0049	Ozone concentration	Temperature	D.15	←	1/3
pair0050	Ozone concentration	Temperature	D.15	←	1/3
pair0051	Ozone concentration	Temperature	D.15	←	1/3
pair0052	(Temp, Press, SLP, Rh)	(Temp, Press, SLP, RH)	D.12	←	0
pair0053	Ozone concentration	(Wind speed, Radiation, Temp.)	D.16	←	0
pair0054	(Displ., Horsepower, Weight)	(Fuel consumption, Acceleration)	D.4	→	0
pair0055	Ozone concentration (16-dim.)	Radiation (16-dim.)	D.15	←	0
pair0056	Fem. life expectancy, 2000–2005	Latitude of capital	D.17	←	1/12
pair0057	Fem. life expectancy, 1995–2000	Latitude of capital	D.17	←	1/12
pair0058	Fem. life expectancy, 1990–1995	Latitude of capital	D.17	←	1/12
pair0059	Fem. life expectancy, 1985–1990	Latitude of capital	D.17	←	1/12
pair0060	Male life expectancy, 2000–2005	Latitude of capital	D.17	←	1/12
pair0061	Male life expectancy, 1995–2000	Latitude of capital	D.17	←	1/12
pair0062	Male life expectancy, 1990–1995	Latitude of capital	D.17	←	1/12
pair0063	Male life expectancy, 1985–1990	Latitude of capital	D.17	←	1/12
pair0064	Drinking water access	Infant mortality	D.17	→	1/12
pair0065	Stock return of Hang Seng Bank	Stock return of HSBC Hldgs	D.18	→	1/3
pair0066	Stock return of Hutchison	Stock return of Cheung kong	D.18	→	1/3
pair0067	Stock return of Cheung kong	Stock return of Sun Hung Kai Prop.	D.18	→	1/3
pair0068	Bytes sent	Open http connections	D.19	←	1
pair0069	Inside temperature	Outside temperature	D.20	←	1
pair0070	Parameter	Answer	D.21	→	1
pair0071	Symptoms (6-dim.)	Classification of disease (2-dim.)	D.22	→	0
pair0072	Sunspots	Global mean temperature	D.23	→	1
pair0073	CO_2 emissions	Energy use	D.17	←	1/12
pair0074	GNI per capita	Life expectancy	D.17	→	1/12
pair0075	Under-5 mortality rate	GNI per capita	D.17	←	1/12
pair0076	Population growth	Food consumption growth	D.24	→	1
pair0077	Temperature	Solar radiation	D.11	←	1/2
pair0078	PPFD	Net Ecosystem Productivity	D.25	→	1/3
pair0079	Net Ecosystem Productivity	Diffuse PPFD	D.25	←	1/3
pair0080	Net Ecosystem Productivity	Direct PPFD	D.25	←	1/3
pair0081	Temperature	Local CO_2 flux, BE-Bra	D.26	→	1/3
pair0082	Temperature	Local CO_2 flux, DE-Har	D.26	→	1/3
pair0083	Temperature	Local CO_2 flux, US-PFa	D.26	→	1/3
pair0084	Employment	Population	D.27	←	1
pair0085	Time of measurement	Protein content of milk	D.28	→	1
pair0086	Size of apartment	Monthly rent	D.29	→	1
pair0087	Temperature	Total snow	D.30	→	1
pair0088	Age	Relative spinal bone mineral density	D.31	→	1
pair0089	Root decomposition in Oct	Root decomposition in Apr	D.32	←	1/4
pair0090	Root decomposition in Oct	Root decomposition in Apr	D.32	←	1/4
pair0091	Clay content in soil	soil moisture	D.32	→	1/4
pair0092	Organic carbon in soil	Clay content in soil	D.32	←	1/4
pair0093	Precipitation	Runoff	D.33	→	1
pair0094	Hour of the day	Temperature	D.34	→	1/3
pair0095	Hour of the day	Electricity consumption	D.34	→	1/3
pair0096	Temperature	Electricity consumption	D.34	→	1/3
pair0097	Initial speed	Final speed	D.35	→	1/2
pair0098	Initial speed	Final speed	D.35	→	1/2
pair0099	Language test score	Social-economic status family	D.36	←	1
pair0100	Cycle time of CPU	Performance	D.37	→	1

Table 4: Overview of the pairs in version 1.0 of the CauseEffectPairs benchmark.

D.1 DWD

The DWD climate data were provided by the Deutscher Wetterdienst (DWD). We downloaded the data from http://www.dwd.de and merged several of the original data sets to obtain data for 349 weather stations in Germany, selecting only those weather stations without missing data. After merging the data sets, we selected the following six variables: altitude, latitude, longitude, and annual mean values (over the years 1961–1990) of sunshine duration, temperature and precipitation. We converted the latitude and longitude variables from sexagesimal to decimal notation. Out of these six variables, we selected six different pairs with "obvious" causal relationships: altitude–temperature (pair0001), altitude–precipitation (pair0002), longitude–temperature (pair0003), altitude–sunshine hours (pair0004), latitude–temperature (pair0020), longitude–precipitation (pair0021).

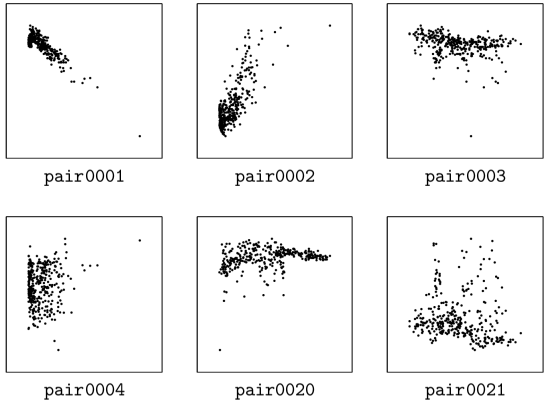


Figure 18: Scatter plots of pairs from D.1. pair0001: altitude → temperature, pair0002: altitude → precipitation, pair0003: longitude → temperature, pair0004: altitude → sunshine hours, pair0020: latitude → temperature, pair0021: longitude → precipitation.

pair0001: Altitude → Temperature

As an elementary fact of meteorology, places with higher altitude tend to be colder than those that are closer to sea level (roughly 1 centigrade per 100 meter). There is no doubt that altitude is the cause and temperature the effect: one could easily think of an intervention where the thermometer is lifted (e.g., by using a balloon) to measure the temperature at a higher point of the same longitude and latitude. On the other hand, heating or cooling a location usually does not change its altitude (except perhaps if the location happens to be the space enclosed by a hot air balloon, but let us assume that the thermometers used to gather this data were fixed to the ground). The altitudes in the DWD data set range from 0 m to 2960 m, which is sufficiently large to detect significant statistical dependences.

One potential confounder is latitude, since all mountains are in the south and far from the sea, which is also an important factor for the local climate. The places with the highest average temperatures are therefore those with low altitude but lying far in the south. Hence this confounder should induce positive correlations between altitude and temperature as opposed to the negative correlation between altitude and temperature that is observed empirically. This suggests that the direct causal relation between altitude and temperature dominates over the confounder.

pair0002: Altitude → Precipitation

It is known that altitude is also an important factor for precipitation since rain often occurs when air is forced to rise over a mountain range and the air becomes over-saturated with water due to the lower temperature (orographic rainfall). This effect defines an indirect causal influence of altitude on precipitation via temperature. These causal relations are, however, less simple than the causal influence from altitude to temperature because gradients of the altitude with respect to the main direction of the wind are more relevant than the altitude itself. A hypothetical intervention that would allow us to validate the causal relation could be to build artificial mountains and observe orographic rainfall.

pair0003: Longitude → Temperature

To detect the causal relation between longitude and temperature, a hypothetical intervention could be to move a thermometer between West and East. Even if one would adjust for altitude and latitude, it is unlikely that temperature would remain the same since the climate in the West is more oceanic and less continental than in the East of Germany. Therefore, longitude causes temperature.

pair0004: Altitude → Sunshine Hours

Sunshine duration and altitude are slightly positively correlated. Possible explanations are that higher weather stations are sometimes above low-hanging clouds. Cities in valleys, especially if they are close to rivers or lakes, typically have more misty days. Moving a sunshine sensor above the clouds clearly increases the sunshine duration whereas installing an artificial sun would not change the altitude. The causal influence from altitude to sunshine duration can be confounded, for instance, by the fact that there is a simple statistical dependence between altitude and longitude in Germany as explained earlier.

pair0020: Latitude → Temperature

Moving a thermometer towards the equator will generally result in an increased mean annual temperature. Changing the temperature, on the other hand, does not necessarily result in a north-south movement of the thermometer. The obvious ground truth of latitude causing temperature might be somewhat "confounded" by longitude, in combination with the selection bias that arises from only including weather stations in Germany.

pair0021: Longitude → Precipitation

As the climate in the West is more oceanic and less continental than in the East of Germany, we expect there to be a relationship between longitude and precipitation. Changing longitude by moving in East-West direction may therefore change precipitation, even if one would adjust for altitude and latitude. On the other hand, making it rain locally (e.g., by cloud seeding) will not result in a change in longitude.

D.2 Abalone

The Abalone data set (Nash et al., 1994) in the UCI Machine Learning Repository (Bache and Lichman, 2013) contains 4177 measurements of several variables concerning the sea snail *Abalone*. We downloaded the data from https://archive.ics.uci.edu/ml/datasets/Abalone. The original data set contains the nine variables sex, length, diameter, height, whole weight, shucked weight, viscera weight, shell weight and number of rings. The number of rings in the shell is directly related to the age of the snail: adding 1.5 to the number of rings gives the age in years. Of these variables, we selected six pairs with obvious cause-effect relationships: age–length (pair0005), age–shell weight (pair0006), age–diameter (pair0007), age–height (pair0008), age–whole weight (pair0009), age–shucked weight (pair0010), age–viscera weight (pair0011).

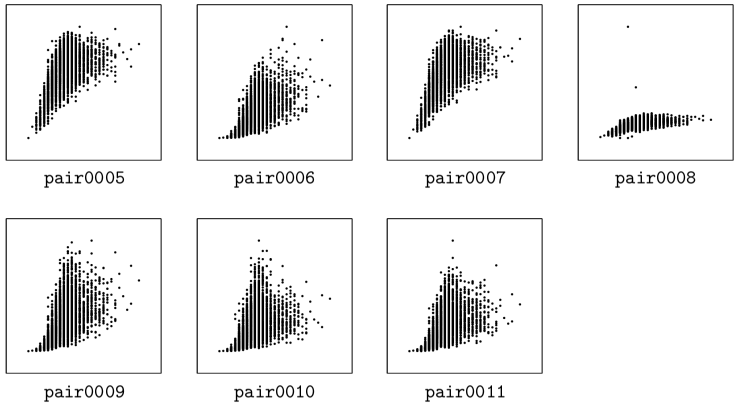


Figure 19: Scatter plots of pairs from D.2. pair0005: age → length, pair0006: age → shell weight, pair0007: age → diameter, pair0008: age → height, pair0009: age → whole weight, pair0010: age → shucked weight, pair0011: age → viscera weight.

pair0005–pair0011: Age → {Length, Shell Weight, Diameter, Height, Whole/Shucked/Viscera Weight}

For the variable "age" it is not obvious what a reasonable intervention would be since there is no possibility to change the time. However, waiting and observing how variables

change over time can be considered as equivalent to the hypothetical intervention on age (provided that the relevant background conditions do not change too much). Clearly, this "intervention" would change the probability distribution of the length, whereas changing the length of snails (by surgery) would not change the distribution of age (assuming that the surgery does not take years). Regardless of the difficulties of defining interventions, we expect common agreement on the ground truth: age causes all the other variables related to length, diameter height and weight.

There is one subtlety that has to do with how age is measured for these shells: this is done by counting the rings. For the variable "number of rings" however, changing the length of the snail may actually change the number of rings. We here presume that all snails have undergone their natural growing process so that the number of rings is a good proxy for the variable age.

D.3 Census Income KDD

The Census Income (KDD) data set (U.S. Department of Commerce, 1994) in the UCI Machine Learning Repository (Bache and Lichman, 2013) has been extracted from the 1984 and 1985 U.S. Census studies. We downloaded the data from https://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD). We have selected the following variables: AAGE (age), AHRSPAY (wage per hour) and DIVVAL (dividends from stocks).

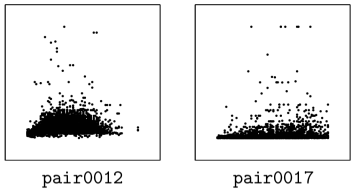


Figure 20: Scatter plots of pairs from D.3. pair0012: age → wage per hour, pair0017: age → dividends from stocks.

pair0012: Age → Wage Per Hour

We only used the first 5000 instances for which wage per hour was not equal to zero. The data clearly shows an increase of wage up to about 45 and a decrease for higher age.

As already argued for the Abalone data, interventions on the variable "age" are difficult to define. Compared to the discussion in the context of the Abalone data set, it seems more problematic to consider waiting as a reasonable "intervention" here, since the relevant (economical) background conditions change rapidly compared to the length of the human life: If someone's salary is higher than the salary of a 20 year younger colleague *because* of his/her longer job experience, we cannot conclude that the younger colleague will earn the same money 20 years later as the older colleague earns now. Possibly, the factory or even the branch of industry he/she was working in does not exist any more and his/her job experience is no longer appreciated. However, we know that employees sometimes indeed do get a higher income because of their longer job experience. Pretending longer job experience

by a fake certificate of employment would be a possible intervention. On the other hand, changing the wage per hour is an intervention that is easy to imagine (though difficult for us to perform) and this would certainly not change the age.

pair0017: Age → Dividends from Stocks

We only used the first 5000 instances for which dividends from stocks was not equal to zero. Similar considerations apply as for age vs. wage per hour. Doing an intervention on age is not practical, but companies could theoretically intervene on the dividends from stocks, and that would not result in a change of age, obviously. On the other hand, age influences income, and thereby over time, the amount of money that people can invest in stocks, and thereby, the amount of dividends they earn from stocks. This causal relation is a very indirect one, though, and the dependence between age and dividends from stock is less pronounced than that between age and wage per hour.

D.4 Auto-MPG

The Auto-MPG data set in the UCI Machine Learning Repository (Bache and Lichman, 2013) concerns city-cycle fuel consumption in miles per gallon (MPG), i.e., the number of miles a car can drive on one gallon of gasoline, and contains several other attributes, like displacement, horsepower, weight, and acceleration. The original data set comes from the StatLib library (Meyer and Vlachos, 2014) and was used in the 1983 American Statistical Association Exposition. We downloaded the data from http://archive.ics.uci.edu/ml/datasets/Auto+MPG and selected only instances without missing data, thereby obtaining 392 samples.

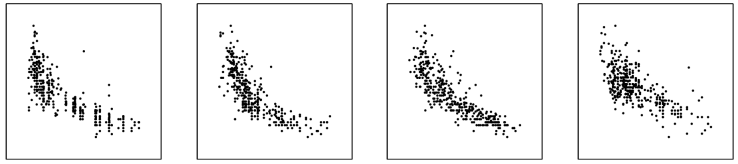


Figure 21: Scatter plots of pairs from D.4. pair0013: displacement → fuel consumption, pair0014: horse power → fuel consumption, pair0015: weight → fuel consumption, pair0016: horsepower → acceleration, pair0054: (displacement,horsepower,weight) → (MPG,acceleration)

pair0013: Displacement → Fuel Consumption

Displacement is the total volume of air/fuel mixture an engine can draw in during one complete engine cycle. The larger the displacement, the more fuel the engine can consume with every turn. Intervening on displacement (e.g., by increasing the cylinder bore) changes the fuel consumption. Changing the fuel consumption (e.g., by increasing the weight of the

car, or changing its air resistance, or by using another gear) will not change the displacement, though.

pair0014: Horse power → Fuel Consumption

Horse power measures the amount of power an engine can deliver. There are various ways to define horsepower and different standards to measure horse power of vehicles. In general, though, it should be obvious that fuel consumption depends on various factors, including horse power. Changing horsepower (e.g., by adding more cylinders to an engine, or adding a second engine to the car) would lead to a change in fuel consumption. On the other hand, changing fuel consumption does not necessarily change horse power.

pair0015: Weight → Fuel Consumption

There is a strong selection bias here, as car designers use a more powerful motor (with higher fuel consumption) for a heavier car. Nevertheless, the causal relationship between weight and fuel consumption should be obvious: if we intervene on weight, then fuel consumption will change, but not necessarily vice versa.

pair0016: Horsepower → Acceleration

Horsepower is one one of the factors that cause acceleration. Other factors are wheel size, the gear used, and air resistance. Intervening on acceleration does not necessarily change horsepower.

pair0054: (Displacement,Horsepower,Weight) → (MPG,Acceleration)

This pair consists of two multivariate variables that are combinations of the variables we have considered before. The multivariate variable consisting of the three components displacement, horsepower and weight can be considered to cause the multivariate variable comprised of fuel consumption and acceleration.

D.5 GAGurine

This data concerns the concentration of the chemical compound Glycosaminoglycan (GAG) in the urine of 314 children aged from zero to seventeen years. This is the GAGurine data set supplied with the MASS package of the computing language R (Venables and Ripley, 2002).

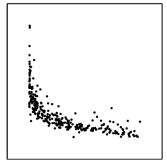


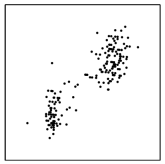
Figure 22: Scatter plots of pairs from D.5. pair0018: age → concentration GAG.

pair0018: Age → Concentration GAG

Obviously, GAG concentration does not cause age, but it could be the other way around, considering the strong dependence between the two variables.

D.6 Old Faithful

This is the geyser data set supplied with the MASS package of the computing language R (Venables and Ripley, 2002). It is originally described in (Azzalini and Bowman, 1990) and contains data about the duration of an eruption and the time interval between subsequent eruptions of the Old Faithful geyser in Yellowstone National Park, USA. The data consists of 194 samples and was collected in a single continuous measurement from August 1 to August 15, 1985.



pair0019

Figure 23: Scatter plots of pairs from D.6. pair0019: current duration → next interval.

pair0019: Current Duration → Next Interval

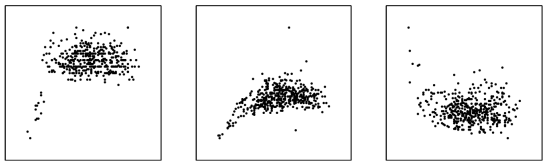
The chronological ordering of events implicates that the time interval between the current and the next eruption is an effect of the duration of the current eruption.

D.7 Arrhythmia

The Arrhythmia data set (Guvenir et al., 1997) from the UCI Machine Learning Repository (Bache and Lichman, 2013) concerns cardiac arrhythmia. It consists of 452 patient records and contains many different variables. We downloaded the data from https://archive.ics.uci.edu/ml/datasets/Arrhythmia and only used the variables for which the causal relationships should be evident. We removed two instances from the data set, corresponding with patient lengths of 680 and 780 cm, respectively.

pair0022–pair0024: Age → {Height, Weight, Heart Rate}

As discussed before, "interventions" on age (for example, waiting a few years) may affect height of persons. On the other hand, we know that height does not cause age. The same holds for age and weight and for age and heart rate. It is important to note here that age is simply measured in years since the birth of a person. Indeed, weight, height and also heart rate might influence "biological aging", the gradual deterioration of function of the human body.



pair0022 pair0023 pair0024

Figure 24: Scatter plots of pairs from D.7. pair0022: age → height, pair0023: age → weight, pair0024: age → heart rate.

D.8 Concrete Compressive Strength

This data set, available at the UCI Machine Learning Repository (Bache and Lichman, 2013), concerns a systematic study (Yeh, 1998) regarding concrete compressive strength as a function of ingredients and age. Citing Yeh (1998): "High-performance concrete (HPC) is a new terminology used in the concrete construction industry. In addition to the three basic ingredients in conventional concrete, i.e., Portland cement, fine and coarse aggregates, and water, the making of HPC needs to incorporate supplementary cementitious materials, such as fly ash and blast furnace slag, and chemical admixture, such as superplasticizer 1 and 2. Several studies independently have shown that concrete strength development is determined not only by the water-to-cement ratio, but that it also is influenced by the content of other concrete ingredients." Compressive strength is measured in units of MPa, age in days, and the other variables are measured in kilograms per cubic metre of concrete mixture. The data set was downloaded from https://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength and contains 1030 measurements.

pair0025–pair0032: {Cement, Blast Furnace Slag, Fly Ash, Water, Superplasticizer, Coarse Aggregate, Fine Aggregate, Age} → Compressive Strength

It should be obvious that compressive strength is the effect, and the other variables are its causes. Note, however, that in practice one cannot easily intervene on the mixture components without simultaneously changing the other mixture components. For example, if one adds more water to the mixture, then as a result, all other components will decrease, as they are measured in kilograms per cubic metre of concrete mixture. Nevertheless, we expect that we can see these interventions as reasonable approximations of "perfect interventions" on a single variable.

D.9 Liver Disorders

This data set, available at the UCI Machine Learning Repository (Bache and Lichman, 2013), was collected by BUPA Medical Research Ltd. It consists of several blood test results, which are all thought to be indicative for liver disorders that may arise from excessive alcohol consumption. Each of the 345 instances constitutes the record of a single male individual. Daily alcohol consumption is measured in number of half-pint equivalents of

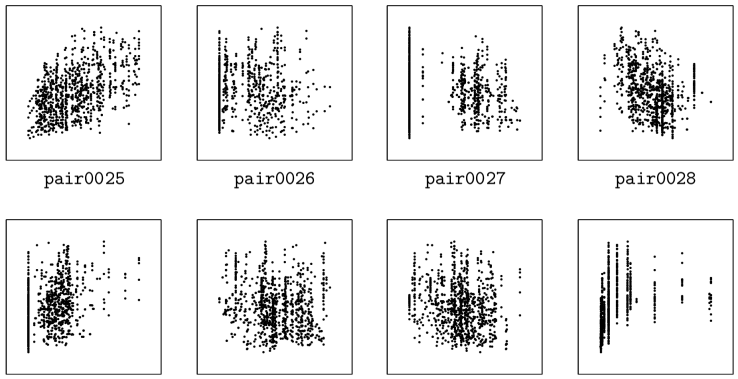


Figure 25: Scatter plots of pairs from D.8. `pair0025`: cement → compressive strength, `pair0026`: blast furnace slag → compressive strength, `pair0027`: fly ash → compressive strength, `pair0028`: water → compressive strength, `pair0029`: superplasticizer → compressive strength, `pair0030`: coarse aggregate → compressive strength, `pair0031`: fine aggregate → compressive strength, `pair0032`: age → compressive strength.

alcoholic beverages drunk per day. The blood test results are mean corpuscular volume (MCV), alkaline phosphotase (ALP), alanine aminotransferase (ALT), aspartate aminotransferase (AST), and gamma-glutamyl transpeptidase (GGT). The data is available at `https://archive.ics.uci.edu/ml/datasets/Liver+Disorders`.

Although one would expect that daily alcohol consumption is the cause, and the blood test results are the effects, this is not necessarily the case. Indeed, citing Baynes and Dominiczak (1999): "[...] increased plasma concentrations of acetaldehyde after the ingestion of alcohol [...] causes the individual to experience unpleasant flushing and sweating, which discourages alcohol abuse. Disulfiram, a drug that inhibits ALDH, also leads to these symptoms when alcohol is taken, and may be given to reinforce abstinence from alcohol." This means that *a priori*, a reverse causation of the chemical whose concentration is measured in one of these blood tests on daily alcohol consumption cannot be excluded with certainty. Nevertheless, we consider this to be unlikely, as the medical literature describes how these particular blood tests can be used to diagnose liver disorders, but we did not find any evidence that these chemicals can be used to *treat* excessive alcohol consumption.

`pair0033`: Alcohol Consumption → Mean Corpuscular Volume

The mean corpuscular volume (MCV) is the average volume of a red blood cell. An elevated MCV has been associated with alcoholism (Tønnesen et al., 1986), but there are many other factors also associated with MCV.

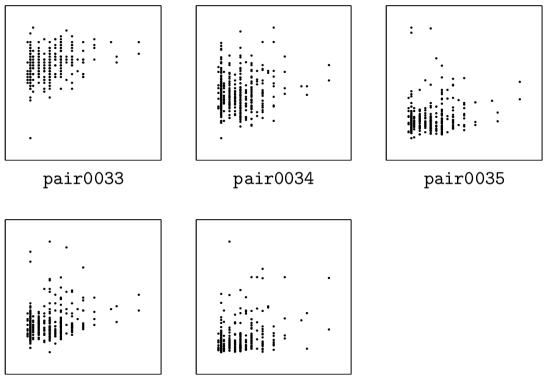


Figure 26: Scatter plots of pairs from D.9. `pair0033`: alcohol consumption → mean corpuscular volume, `pair0034`: alcohol consumption → alkaline phosphotase, `pair0035`: alcohol consumption → alanine aminotransferase, `pair0036`: alcohol consumption → aspartate aminotransferase, `pair0037`: alcohol consumption → gamma-glutamyl transpeptidase.

`pair0034`: Alcohol Consumption → Alkaline Phosphotase

Alkaline phosphatase (ALP) is an enzyme that is predominantly abundant in liver cells, but is also present in bone and placental tissue. Elevated ALP levels in blood can be due to many different liver diseases and also bone diseases, but also occur during pregnancy (Braunwald et al., 2001).

`pair0035`: Alcohol Consumption → Alanine Aminotransferase

Alanine Aminotransferase (ALT) is an enzyme that is found primarily in the liver cells. It is released into the blood in greater amounts when there is damage to the liver cells, for example due to a viral hepatitis or bile duct problems. ALT levels are often normal in alcoholic liver disease (Braunwald et al., 2001).

`pair0036`: Alcohol Consumption → Aspartate Aminotransferase

Aspartate aminotransferase (AST) is an enzyme that is found in the liver, but also in many other bodily tissues, for example the heart and skeletal muscles. Similar to ALT, the AST levels raise in acute liver damage. Elevated AST levels are not specific to the liver, but can also be caused by other diseases, for example by pancreatitis. An AST:ALT ratio of more than 3:1 is highly suggestive of alcoholic liver disease (Braunwald et al., 2001).

pair0037: Alcohol Consumption → Gamma-Glutamyl Transpeptidase

Gamma-Glutamyl Transpeptidase (GGT) GGT is another enzyme that is primarily found in liver cells. It is rarely elevated in conditions other than liver disease. High GGT levels have been associated with alcohol use (Braunwald et al., 2001).

D.10 Pima Indians Diabetes

This data set, available at the UCI Machine Learning Repository (Bache and Lichman, 2013), was collected by the National Institute of Diabetes and Digestive and Kidney Diseases in the USA to forecast the onset of diabetes mellitus in a high risk population of Pima Indians near Phoenix, Arizona. Cases in this data set were selected according to several criteria, in particular being female, at least 21 years of age and of Pima Indian heritage. This means that there could be selection bias on age.

We downloaded the data from `https://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes`. We only selected the instances with nonzero values, as it seems likely that zero values encode missing data. This yielded 768 samples.

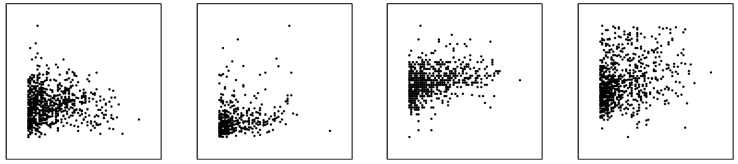


Figure 27: Scatter plots of pairs from D.10. `pair0038`: age → body mass index, `pair0039`: age → serum insulin, `pair0040`: age → diastolic blood pressure, `pair0041`: age → plasma glucose concentration.

pair0038: Age → Body Mass Index

Body mass index (BMI) is defined as the ratio between weight (kg) and the square of height (m). Obviously, age is not caused by body mass index, but as age is a cause of both height and weight, age causes BMI.

pair0039: Age → Serum Insulin

2-Hour serum insulin (μU/ml), measured 2 hours after the ingestion of a standard dose of glucose, in an oral glucose tolerance test. We can exclude that serum insulin causes age, and there could be an effect of age on serum insulin. Another explanation for the observed dependence could be the selection bias.

pair0040: Age → Diastolic Blood Pressure

Diastolic blood pressure (mm Hg). It seems obvious that blood pressure does not cause age. The other causal direction seems plausible, but again, an alternative explanation for the dependence could be selection bias.

pair0041: Age → Plasma Glucose Concentration

Plasma glucose concentration, measured 2 hours after the ingestion of a standard dose of glucose, in an oral glucose tolerance test. Similar reasoning as before: we do not believe that plasma glucose concentration causes ages, but it could be the other way around, and there may be selection bias.

D.11 B. Janzing's Meteo Data

This data set is from a private weather station, owned by Bernward Janzing, located in Furtwangen (Black Forest), Germany at an altitude of 956 m. The measurements include temperature, precipitation, and snow height (since 1979), as well as solar radiation (since 1986). The data have been archived by Bernward Janzing, statistical evaluations have been published in Janzing (2004), monthly summaries of the weather are published in local newspapers since 1981.

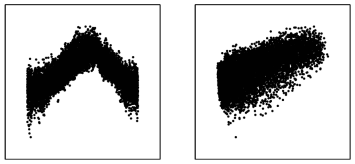


Figure 28: Scatter plots of pairs from D.11. `pair0042`: day of the year → temperature, `pair0077`: solar radiation → temperature.

pair0042: Day of the Year → Temperature

This data set shows the dependence between season and temperature over 25 years plus one month, namely the time range 01/01/1979–01/31/2004. It consists of 9162 measurements.

One variable is the day of the year, represented by an integer from 1 to 365 (or 366 for leap years). The information about the year has been dropped. Y is the mean temperature of the respective day, calculated according to the following definition:

$$T_{mean} := \frac{T_{morning} + T_{midday} + 2T_{evening}}{4},$$

where morning, midday, and evening are measured at 7:00 am, 14:00 pm, and 21:00 pm (MEZ), respectively (without daylight saving time). Double counting of the evening value is official standard of the German authority "Deutscher Wetterdienst". It has been defined at a time where no electronic data loggers were available and thermometers had to be read

out by humans. Weighting the evening value twice has been considered a useful heuristics to account for the missing values at night.

We consider day of the year as the cause, since it can be seen as expressing the angular position on its orbit around the sun. Although true interventions are infeasible, it is commonly agreed that changing the position of the earth would result in temperature changes at a fixed location due to the different solar incidence angle.

`pair0077`: Solar Radiation → Temperature

This data set shows the relation between solar radiation and temperature over 23 years, namely the interval 01/01/1986–12/31/2008. It consists of 8401 measurements.

Solar radiation is measured per area in W/m^2 averaged over one day on a horizontal surface. Temperature is the averaged daily, as in `pair0042`. The original data has been processed by us to extract the common time interval. We assume that radiation causes temperature. High solar radiation increases the temperature of the air already at a scale of hours. Interventions are easy to implement: Creating artificial shade on a large enough surface would decrease the air temperature. On longer time scales there might also be an influence from temperature to radiation via the generation of clouds through evaporation in more humid environments. This should, however, not play a role for daily averages.

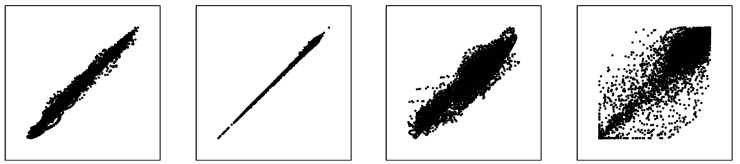
D.12 NCEP-NCAR Reanalysis

This data set, available from the NOAA (National Oceanic and Atmospheric Administration) Earth System Research Laboratory website at `http://www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis.surface.html`, is a subset of a reanalysis data set, incorporating observations and numerical weather prediction model output from 1948 to date (Kalnay et al., 1996). The reanalysis data set was produced by the National Center for Environmental Prediction (NCEP) and the National Center for Atmospheric Research (NCAR). Reanalysis data products aim for a realistic representation of all relevant climatological variables on a spatiotemporal grid. We collected four variables from a global grid of 144×73 cells: air temperature (in K, `pair0043`), surface pressure (in Pascal, `pair0044`), sea level pressure (in Pascal, `pair0045`) and relative humidity (in %, `pair0045`) on two consecutive days, day 50 and day 51 of the year 2000 (i.e., Feb 19th and 20th). Each data pair consists of $144 \times 73 - 143 = 10369$ data points, distributed across the globe. 143 data points were subtracted because at the north pole values are repeated across all longitudes.

Each data point is the daily average over an area that covers $2.5^\circ \times 2.5^\circ$ (approximately 250 km × 250 km at the equator). Because causal influence cannot propagate backwards in time, temperature, pressure and humidity in a certain area are partly affected by their value the day before in the same area.

`pair0043`: Temperature at t → Temperature at t+1

Due to heat storage, mean daily air temperature near surface at any day largely impact daily air temperature at the following day. We assume there is no causation backwards in time, hence the correlation between temperatures at two consecutive days must be driven by confounders (such as large-scale weather patterns) or a causal influence from the first day to the second.



pair0043 pair0044 pair0045 pair0046

Figure 29: Scatter plots of pairs from D.12. `pair0043`: temperature at t → temperature at t+1, `pair0044`: surface pressure at t → surface pressure at t+1, `pair0045`: sea level pressure at t → sea level pressure at t+1, `pair0046`: relative humidity at t → relative humidity at t+1, `pair0052`: (temp, press, slp, rh) at t → (temp, press, slp, rh) at t+1.

`pair0044`: Surface Pressure at t → Surface Pressure at t+1

Pressure patterns near the earth's surface are mostly driven by large-scale weather patterns. However, large-scale weather patterns are also driven by local pressure gradients and hence, some of the correlation between surface pressure at two consecutive days stems from a direct causal link between the first and the second day, as we assume there is no causation in time.

`pair0045`: Sea Level Pressure at t → Sea Level Pressure at t+1

Similar reasoning as in `pair0044`.

`pair0046`: Relative Humidity at t → Relative Humidity at t+1

Humidity of the air at one day affects the humidity of the following day because if no air movement takes place and no drying or moistening occurs, it will approximately stay the same. Furthermore, as reasoned above, because there is no causation backwards in time, relative humidity at day $t+1$ cannot affect humidity at day t. Note that relative humidity has values between 0 and 100. Values can be saturated in very humid places such as tropical rainforest and approach 0 in deserts. For this reason, the scatter plot looks as if the data were clipped.

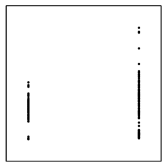
`pair0052`: (Temp, Press, SLP, RH) at t → (Temp, Press, SLP, RH) at t+1

The pairs `pair0043`–`pair0046` were combined to a 4-dimensional vector. From the reasoning above it follows that the vector of temperature, near surface pressure, sea level pressure and relative humidity at day t has a causal influence on the vector of the same variables at time $t+1$.

D.13 Traffic

This data set has been extracted from `http://www.b30-oberschwaben.de/html/tabelle.html`, a website containing various kinds of information about the national highway B30. This is a road in the federal state Baden-Württemberg, Germany, which provides an impor-

tant connection of the region around Ulm (in the North) with the Lake Constance region (in the South). After extraction, the data set contains 254 samples.



pair0047

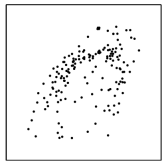
Figure 30: Scatter plots of pairs from D.13. pair0047: type of day → number of cars.

pair0047: Type of Day → Number of Cars

One variable is the number of cars per day, the other denotes the type of the respective day, with "1" indicating Sundays and holidays and "2" indicating working days. The type of day causes the number of cars per day. Indeed, introducing an additional holiday by a political decision would certainly change the amount of traffic on that day, while changing the amount of traffic by instructing a large number of drivers to drive or not to drive at a certain day would certainly not change the type of that day.

D.14 Hipel & McLeod

This data set contains 168 measurements of indoor and outdoor temperatures. It was taken from a book by Hipel and McLeod (1994) and can be downloaded from http://www.stats.uwo.ca/faculty/mcleod/epubs/mhsets/readme-mhsets.html.



pair0048

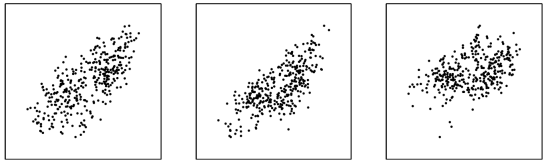
Figure 31: Scatter plots of pairs from D.14. pair0048: outdoor temperature → indoor temperature.

pair0048: Outdoor Temperature → Indoor Temperature

Outdoor temperatures can have a strong impact on indoor temperatures, in particular when indoor temperatures are not adjusted by air conditioning or heating. Contrarily, indoor temperatures will have little or no effect on outdoor temperatures, because the outside environment has a much larger heat capacity.

D.15 Bafu

This data set deals with the relationship between daily ozone concentration in the air and temperature. It was downloaded from http://www.bafu.admin.ch/luft/luftbelastung/blick_zurueck/datenabfrage/index.html. Lower atmosphere ozone (O_3) is a secondary pollutant that is produced by the photochemical oxidation of carbon monoxide (CO), methane (CH_4), and non-methane volatile organic compounds (NMVOCs) by OH in the presence of nitrogen oxides (NO_x, NO + NO_2) (Rasmussen et al., 2012). It is known that ozone concentration strongly correlates with surface temperature (Bloomer et al., 2009). Several explanations are given in the literature (see e.g., Rasmussen et al., 2012). Without going into details of the complex underlying chemical processes, we mention that the crucial chemical reactions are stronger at higher temperatures. For instance, isoprene emissions of plants increase with increasing temperature and isoprene can play a similar role in the generation of O_3 as NO_x (Rasmussen et al., 2012). Apart from this, air pollution may be influenced indirectly by temperature, e.g., via increasing traffic at 'good' weather conditions or an increased occurrence rate of wildfires. All these explanations state a causal path from temperature to ozone. Note that the phenomenon of ozone pollution in the lower atmosphere discussed here should not be confused with the 'ozone hole', which is a lack of ozone in the higher atmosphere. Close to the surface, ozone concentration does not have an impact on temperatures. For all three data sets, ozone is measured in $\mu g/m^3$ and temperature in °C.



pair0049 pair0050 pair0051

Figure 32: Scatter plots of pairs from D.15. pair0049: temperature → ozone concentration, pair0050: temperature → ozone concentration, pair0051: temperature → ozone concentration, pair0055: radiation → ozone concentration.

pair0049: Temperature → Ozone Concentration

365 daily mean values of ozone and temperature of year 2009 in Lausanne-César-Roux, Switzerland.

pair0050: Temperature → Ozone Concentration

365 daily mean values of ozone and temperature of year 2009 in Chaumont, Switzerland.

pair0051: Temperature → Ozone Concentration

365 daily mean values of ozone and temperature of year 2009 in Davos-See, Switzerland.

pair0055: Radiation → Ozone Concentration

72 daily mean values of ozone concentrations and radiation in the last 83 days of 2009 at 16 different places in Switzerland (11 days were deleted due to missing data). Solar radiation and surface ozone concentration are correlated (Feister and Balzer, 1991). The deposition of ozone is driven by complex micro-meteorological processes including wind direction, air temperature, and global radiation (Stockwell et al., 1997). For instance, solar radiation affects the height of the planetary boundary layer and cloud formation and thus indirectly influences ozone concentrations. In contrast, global radiation is not driven by ozone concentrations close to the surface.

Ozone is given in $\mu g/m^3$, radiation in W/m^2. The 16 different places are: 1: Bern-Bollwerk, 2: Magadino-Cadenazzo, 3: Lausanne-César-Roux, 4: Payerne, 5: Lugano-Universita, 6: Taenikon, 7: Zuerich-Kaserne, 8: Laegeren, 9: Basel-Binningen, 10: Chaumont, 11: Duebendorf, 12: Rigi-Seebodenalp, 13: Haerkingen, 14: Davos-See, 15: Sion-Aéroport, 16: Jungfraujoch.

D.16 Environmental

We downloaded ozone concentration, wind speed, radiation and temperature from http://www.mathe.tu-freiberg.de/Stoyan/umwdat.html, discussed in Stoyan et al. (1997). The data consist of 989 daily values over the time period from 05/01/1989 to 10/31/1994 observed in Heilbronn, Germany.

pair0053: (Wind Speed, Radiation, Temperature) → Ozone Concentration

As we have argued above in Section D.15, wind direction (and speed), air temperature, and global radiation influence local ozone concentrations. Wind can influence ozone concentrations for example in the following way. No wind will keep the the concentration of ozone in a given air parcel constant if no lateral or vertical sources or sinks are prevalent. In contrast, winds can move and disperse and hence mix air with different ozone concentrations. Ozone concentration is given in $\mu g/m^3$, wind speed in m/s, global radiation in W/m^2 and temperature in °C.

D.17 UNdata

The following data were taken from the "UNdata" database of the United Nations Statistics Division at http://data.un.org.

pair0056–pair0059: Latitude of Capital → Female Life Expectancy

Pairs pair0056–pair0059 consist of female life expectancy (in years) at birth versus latitude of the country's capital, for various countries (China, Russia and Canada were removed). The four pairs correspond with measurements over the periods 2000–2005, 1995–2000, 1990–1995, 1985–1990, respectively. The data were downloaded from http://data.un.org/Data.aspx?d=GenderStat&f=inID%3a37.

The location of a country (encoded in the latitude of its capital) has an influence on how poor or rich a country is, hence affecting the quality of the health care system and ultimately life expectancy. This influence could stem from abundance of natural resources within

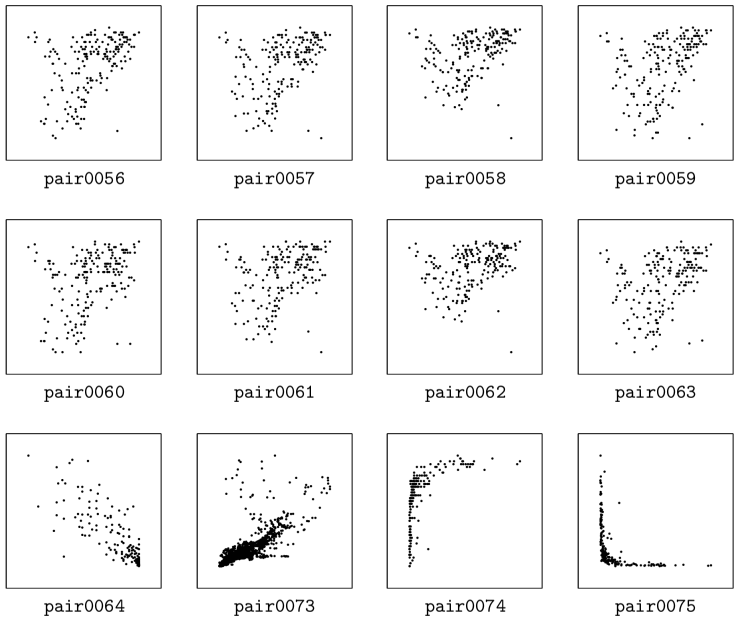


Figure 33: Scatter plots of pairs from D.17. pair0056–pair0059: latitude of capital → female life expectancy, pair0060–pair0063: latitude of capital → male life expectancy, pair0064: drinking water access → infant mortality, pair0073: energy use → CO_2 emissions, pair0074: GNI per capita → life expectancy, pair0075: GNI per capita → under-5 mortality rate.

the country's borders or the influence neighboring countries have on its economic welfare. Furthermore, the latitude can influence life expectancy via climatic factors. For instance, life expectancy might be smaller if a country frequently experiences climatic extremes. In contrast, it is clear that life expectancy does not have any effect on latitude.

pair0060–pair0063: Latitude of Capital → Male Life Expectancy

Pairs pair0060–pair0063 are similar, but concern male life expectancy. The same reasoning as for female life expectancy applies here.

pair0064: Drinking Water Access → Infant Mortality

Here, one variable describes the percentage of population with sustainable access to improved drinking water sources in 2006, whereas the other variable denotes the infant mortality rate (per 1000 live births) for both sexes. The data were downloaded from http://

data.un.org/Data.aspx?d=WHO&f=inID%3aMBD10 and http://data.un.org/Data.aspx?d=WHO&f=inID%3aRF03, respectively, and consist of 163 samples.

Clean drinking water is a primary requirement for health, in particular for infants (Esrey et al., 1991). Changing the percentage of people with access to clean water will directly change the mortality rate of infants, since infants are particularly susceptible to diseases (Lee et al., 1997). There may be some feedback, because if infant mortality is high in a poor country, development aid may be directed towards increasing the access to clean drinking water.

pair0073: Energy Use → CO_2 Emissions

This data set contains energy use (in kg of oil equivalent per capita) and CO_2 emission data from 152 countries between 1960 and 2005, yielding together 5084 samples. Considering the current energy mix across the world, the use of energy clearly results in CO_2 emissions (although in varying amounts across energy sources). Contrarily, a hypothetical change in CO_2 emissions will not affect the energy use of a country on the short term. On the longer term, if CO_2 emissions increase, this may cause energy use to decrease because of fear for climate change.

pair0074: GNI per Capita → Life Expectancy

We collected the Gross National Income (GNI, in USD) per capita and the life expectancy at birth (in years) for 194 different countries. GNI can be seen as an index of wealth of a country. In general, richer countries have a better health care system than poor countries an thus can take better care of their citizens when they are ill. Reversely, we believe that the life expectancy of humans has a smaller impact on how wealthy a country is than vice versa.

pair0075: GNI per Capita → Under-5 Mortality Rate

Here we collected the Gross National Income (GNI, in USD) per capita and the under-5 mortality rate (deaths per 1000 live births) for 205 different countries. The reasoning is similar as in pair0074. GNI as an index of wealth influences the quality of the health care system, which in turn determines whether young children will or will not die from minor diseases. As children typically do not contribute much to GNI per capita, we do not expect the reverse causal relation to be very strong.

D.18 Yahoo database

These data denote stock return values and were downloaded from http://finance.yahoo.com. We collected 1331 samples from the following stocks between January 4th, 2000 and June 17, 2005: Hang Seng Bank (0011.HK), HSBC Hldgs (0005.HK), Hutchison (0013.HK), Cheung kong (0001.HK), and Sun Hung Kai Prop. (0016.HK). Subsequently, the following preprocessing was applied, which is common in financial data processing:

1. Extract the dividend/split adjusted closing price data from the Yahoo Finance data base.

2. For the few days when the price is not available, we use simple linear interpolation to estimate the price.

3. For each stock, denote the closing price on day t by P_t, and the corresponding return is calculated as $X_t = (P_t - P_{t-1})/P_{t-1}$.

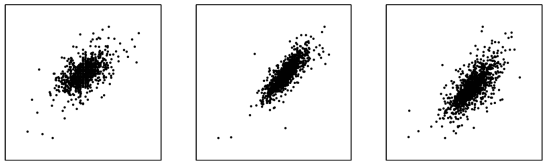


Figure 34: Scatter plots of pairs from D.18. pair0065: Stock Return of Hang Seng Bank → Stock Return of HSBC Hldgs, pair0066: Stock Return of Hutchison → Stock Return of Cheung kong, pair0067: Stock Return of Cheung kong → Stock Return of Sun Hung Kai Prop.

pair0065: Stock Return of Hang Seng Bank → Stock Return of HSBC Hldgs

HSBC owns 60% of Hang Seng Bank. Consequently, if stock returns of Hang Seng Bank change, this should have an influence on stock returns of HSBC Hldgs, whereas causation in the other direction would be expected to be less strong.

pair0066: Stock Return of Hutchison → Stock Return of Cheung kong

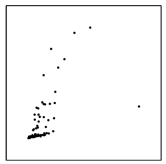
Cheung kong owns about 50% of Hutchison. Same reasoning as in pair0065.

pair0067: Stock Return of Cheung kong → Stock Return of Sun Hung Kai Prop.

Sun Hung Kai Prop. is a typical stock in the Hang Seng Property subindex, and is believed to depend on other major stocks, including Cheung kong.

D.19 Internet Traffic Data

This data set has been created from the log-files of a http-server of the Max Planck Institute for Intelligent Systems in Tübingen, Germany. The variable Internet connections counts the number of times an internal website of the institute has been accessed during a time interval of 1 minute (more precisely, it counts the number of URL requests). Requests for non-existing websites are not counted. The variable Byte transferred counts the total number of bytes sent for all those accesses during the same time interval. The values $(x_1, y_1), \ldots, (x_{498}, y_{498})$ refer to 498 time intervals. To avoid too strong dependence between the measurements, the time intervals are not adjacent but have a distance of 20 minutes.



pair0068

Figure 35: Scatter plots of pairs from D.19. pair0068: internet connections → bytes transferred.

pair0068: Internet Connections → Bytes Transferred

Internet connections causes Bytes transferred because an additional access of the website raises the transfer of data, while transferring more data does not create an additional website access. Note that not every access yields data transfer because the website may still be cached. However, this fact does not spoil the causal relation, it only makes it less deterministic.

D.20 Inside and Outside Temperature

This bivariate time-series data consists of measurements of inside room temperature (°C) and outside temperature (°C), where measurements were taken every 5 minutes for a period of about 56 days, yielding a total of 16382 measurements. The outside thermometer was located on a spot that was exposed to direct sunlight, which explains the large fluctuations. The data were collected by Joris M. Mooij.

pair0069

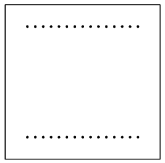
Figure 36: Scatter plots of pairs from D.20. pair0069: outside temperature → inside temperature.

pair0069: Outside Temperature → Inside Temperature

Although there is a causal relationship in both directions, we expect that the strongest effect is from outside temperature on inside temperature, as the heat capacity of the inside of a house is much smaller than that of its surroundings. See also the reasoning for pair0048.

D.21 Armann & Bülthoff

This data set is taken from a psychological experiment that artificially generates images of human faces that interpolate between male and female, taking real faces as basis (Armann and Bülthoff, 2012). The interpolation is done via principal component analysis after representing true face images as vectors in an appropriate high-dimensional space. Human subjects are instructed to label the faces as male or female. The variable "parameter" runs between 0 and 14 and describes the transition from female to male. It is chosen by the experimenter. The binary variable "answer" indicates the answers 'female' and 'male', respectively. The data set consists of 4499 samples.



pair0070

Figure 37: Scatter plots of pairs from D.21. pair0070: parameter → answer.

pair0070: Parameter → Answer

Certainly parameter causes answer. We do not have to talk about *hypothetical* interventions. Instead, we have a true intervention, since "parameter" has been set by the experimenter.

D.22 Acute Inflammations

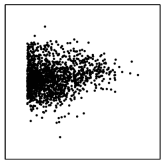
This data set is part of the UCI Machine Learning Repository (Bache and Lichman, 2013) and is available at https://archive.ics.uci.edu/ml/datasets/Acute+Inflammations. It was collected in order to create a computer expert system that decides whether a patient suffers from two different diseases of urinary system (Czerniak and Zarzycki, 2003). The two possible diseases are acute inflammations of urinary bladder and acute nephritises of renal pelvis origin. As it is also possible to chose none of those, the class variable takes values in $\{0,1\}^2$. The decision is based on six symptoms: temperature of patient (e.g. 35.9), occurrence of nausea ("yes" or "no"), lumbar pain ("yes" or "no"), urine pushing ("yes" or "no"), micturition pains ("yes" or "no") and burning of urethra, itch, swelling of urethra outlet ("yes" or "no"). These are grouped together in a six-dimensional vector "symptoms".

pair0071: Symptoms → Classification of Disease

One would think that the disease is causing the symptoms but this data set was created artificially. The description on the UCI homepage says: "The data was created by a medical expert as a data set to test the expert system, which will perform the presumptive diagnosis of two diseases of urinary system. (...) Each instance represents an potential patient." We thus consider the symptoms as the cause for the expert's decision.

D.23 Sunspots

The data set consists of 1632 monthly values between May 1874 and April 2010 and therefore contains 1632 data points. The temperature data have been taken from http://www.cru.uea.ac.uk/cru/data/temperature/ and have been collected by Climatic Research Unit (University of East Anglia) in conjunction with the Hadley Centre (at the UK Met Office) (Morice et al., 2012). The temperature data is expressed in deviations from the 1961–90 mean global temperature of the Earth (i.e., monthly anomalies). The sunspot data (Hathaway, 2010) are taken from the National Aeronautics and Space Administration and were downloaded from http://solarscience.msfc.nasa.gov/SunspotCycle.shtml. According to the description on that website, "sunspot number is calculated by first counting the number of sunspot groups and then the number of individual sunspots. The sunspot number is then given by the sum of the number of individual sunspots and ten times the number of groups. Since most sunspot groups have, on average, about ten spots, this formula for counting sunspots gives reliable numbers even when the observing conditions are less than ideal and small spots are hard to see."



pair0072

Figure 38: Scatter plots of pairs from D.23. pair0072: sunspots → global mean temperature.

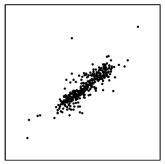
pair0072: Sunspots → Global Mean Temperature

Sunspots are phenomena that appear temporarily on the sun's surface. Although the causes of sunspots are not entirely understood, there is a significant dependence between the number of sunspots and the global mean temperature anomalies (p-value for zero correlation is less than 10^{-4}). There is evidence that the Earth's climate heats and cools as solar activity rises and falls (Haigh, 2007), and the sunspot number can be seen as a proxy for solar activity. Also, we do not believe that the Earth's surface temperature (or changes of the Earth's atmosphere) has an influence on the activity of the sun. We therefore consider number of sunspots causing temperature as the ground truth.

D.24 Food and Agriculture Organization of the UN

The data set has been collected by Food and Agriculture Organization of the UN (http://www.fao.org/economic/ess/ess-fs/en/) and is accessible at http://www.docstoc.com/docs/102679223/Food-consumption-and-population-growth---FAO. It covers 174 countries or areas during the period from 1990–92 to 1995–97 and the period from 1995–97 to 2000–02. As one entry is missing, this gives 347 data points. We selected two variables:

population growth and food consumption. The first variable indicates the average annual rate of change of population (in %), the second one describes the average annual rate of change of total dietary consumption for total population (kcal/day) (also in %).



pair0076

Figure 39: Scatter plots of pairs from D.24. pair0076: population growth → food consumption growth.

pair0076: Population Growth → Food Consumption Growth

We regard population growth to cause food consumption growth, mainly because more people eat more. Both variables are most likely also confounded by the availability of food, driven for instance by advances in agriculture and subsequently increasing yields, but also by national and international conflicts, the global food market and other economic factors. However, for the short time period considered here, confounders which mainly influence the variables on a temporal scale can probably be neglected. Their might also be a causal link from food consumption growth to population growth, for instance one could imagine that if people are well fed, they also reproduce more. However, we assume this link only plays a minor role here.

D.25 Light Response

The filtered version of the light response data was obtained from Moffat (2012). It consists of 721 measurements of Net Ecosystem Productivity (NEP) and three different measures of the Photosynthetic Photon Flux Density (PPFD): the direct, diffuse, and total PPFD. NEP is a measure of the net CO_2 flux between the biosphere and the atmosphere, mainly driven by biotic activity. It is defined as the photosynthetic carbon uptake minus the carbon release by respiration, and depends on the available light. NEP is measured in units of μmol CO_2 m^{-2} s^{-1}. PPFD measures light intensity in terms of photons that are available for photosynthesis, i.e., with wavelength between 400 nm and 700 nm (visible light). More precisely, PPFD is defined as the number of photons with wavelength of 400–700 nm falling on a certain area per time interval, measured in units of μmol photons m^{-2} s^{-1}. The total PPFD is the sum of PPFDdif, which measures only diffusive photons, and PPFDdir, which measures only direct (solar light) photons. The data was measured over several hectare of a forest in Hainich, Germany (site name DE-Hai, latitude: 51.08°N, longitude: 10.45°E), and is available from http://fluxnet.ornl.gov.

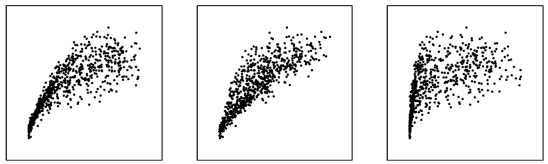


Figure 40: Scatter plots of pairs from D.25. pair0078: PPFD → NEP, pair0079: PPFDdif → NEP, pair0080: PPFDdir → NEP.

pair0078–pair0080: {PPFD,PPFDdif,PPFDdir} → NEP

Net Ecosystem Productivity is known to be driven by both the direct and the diffuse Photosynthetic Photon Flux Density, and hence also by their sum, the total PPFD.

D.26 FLUXNET

The data set contains measurements of net CO_2 exchanges between atmosphere and biosphere aggregated over night, and the corresponding temperature. It is taken from the FLUXNET network (Baldocchi et al., 2001), available at http://fluxnet.ornl.gov (see also Section D.25). The data have been collected at a 10 Hz rate and was aggregated to one value per day over one year (365 values) and at three different sites (BE-Bra, DE-Har, US-PFa). CO_2 exchange measurements typically have a footprint of about 1km^2. The data set contains further information on the quality of the data ("1" means that the value is credible, "NaN" means that the data point has been filled in).

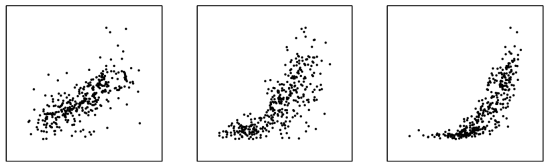


Figure 41: Scatter plots of pairs from D.26. pair0081 (BE-Bra): temperature → local CO_2 flux, pair0082 (DE-Har): temperature → local CO_2 flux, pair0083 (US-PFa): temperature → local CO_2 flux.

pair0081–pair0083: Temperature → Local CO_2 Flux

Because of lack of sunlight, CO_2 exchange at night approximates ecosystem respiration (carbon release from the biosphere to the atmosphere), which is largely dependent on temperature (see, e.g., Mahecha et al., 2010). The CO_2 flux is mostly generated by microbial decomposition in soils and maintenance respiration from plants and does not have a direct effect on temperature. We thus consider temperature causing CO_2 flux as the ground truth.

The three pairs pair0081–pair0083 correspond with sites BE-Bra, DE-Har, US-PFa, respectively.

D.27 US County-Level Growth

The data set from Wheeler (2003) is available at http://www.spatial-econometrics.com/data/contents.html. It contains both employment and population information for 3102 counties in the US in 1980. We selected columns eight and nine in the file "countyg.dat". Column eight contains the natural logarithm of the number of employed people, while column nine contains the natural logarithm of the total number of people living in this county, and is therefore always larger than the number in column eight.

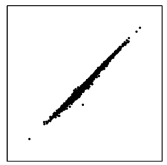


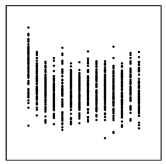
Figure 42: Scatter plots of pairs from D.27. pair0084: population → employment.

pair0084: Population → Employment

It seems reasonable that the total population causes the employment and not vice versa. If we increase the number of people living in an area, this has a direct effect on the number of employed people. We believe that the decision to move into an economically strong area is rather based on the employment rate rather than the absolute number of employed people. There might be an effect that the employment status influences the decision to get children but we regard this effect to be less relevant.

D.28 Milk Protein Trial

This data set is extracted from that for the milk protein trial used by Verbyla and Cullis (1990). The original data set consists of assayed protein content of milk samples taken weekly from each of 79 cows. The cows were randomly allocated to one of three diets: barley, mixed barley-lupins, and lupins, with 25, 27 and 27 cows in the three groups, respectively. Measurements were taken for up to 19 weeks but there were 38 drop-outs from week 15 onwards, corresponding to cows who stopped producing milk before the end of the experiment. We removed the missing values (drop-outs) in the data set: we did not consider the measurements from week 15 onwards, which contain many drop-outs, and we discarded the cows with drop-outs before week 15. Finally, the data set contains 71 cows and 14 weeks, i.e., 994 samples in total. Furthermore, we re-organized the data set to see the relationship between the milk protein and the time to take the measurement. We selected two variables: the time to take weekly measurements (from 1 to 14), and the protein content of the milk produced by each cow at that time.



pair0085

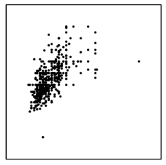
Figure 43: Scatter plots of pairs from D.28. pair0085: time of measurement → protein content of milk.

pair0085: Time of Measurement → Protein Content of Milk

Clearly, the time of the measurement causes the protein content and not vice versa. We do not consider the effect of the diets on the protein content.

D.29 kamernet.nl

This data was collected by Joris M. Mooij from http://www.kamernet.nl, a Dutch website for matching supply and demand of rooms and apartments for students, in 2007. The variables of interest are the size of the apartment or room (in m^2) and the monthly rent in EUR. Two outliers (one with size $0\,m^2$, the other with rent of 1 EUR per month) were removed, after which 666 samples remained.



pair0086

Figure 44: Scatter plots of pairs from D.29. pair0086: size of apartment → monthly rent.

pair0086: Size of Apartment → Monthly Rent

Obviously, the size causes the rent, and not vice versa.

D.30 Whistler Daily Snowfall

The Whistler daily snowfall data is one of the data sets on http://www.mldata.org, and was originally obtained from http://www.climate.weatheroffice.ec.gc.ca/ (Whistler Roundhouse station, identifier 1108906). We downloaded it from http://www.mldata.org/repository/data/viewslug/whistler-daily-snowfall. It concerns historical daily snowfall data in Whistler, BC, Canada, over the period July 1, 1972 to December 31, 2009. It was measured at the top of the Whistler Gondola (Latitude: 50°04′04.000″ N, Longitude:

122°56′50.000″ W, Elevation: 1835 m). We selected two attributes, mean temperature (°C) and total snow (cm). The data consists of 7753 measurements of these two attributes.

pair0087

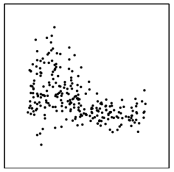
Figure 45: Scatter plots of pairs from D.30. pair0087: temperature → total snow.

pair0087: Temperature → Total Snow

Common sense tells us that the mean temperature is one of the causes of the total amount of snow, although there may be a small feedback effect of the amount of snow on temperature. Confounders are expected to be present (e.g., whether there are clouds).

D.31 Bone Mineral Density

This data set comes from the R package ElemStatLearn, and contains measurements of the age and the relative change of the bone mineral density of 261 adolescents. Each value is the difference in the spinal bone mineral density taken on two consecutive visits, divided by the average. The age is the average age over the two visits. We preprocessed the data by taking only the first measurement for each adolescent, as each adolescent has 1–3 measurements.



pair0088

Figure 46: Scatter plots of pairs from D.31. pair0088: age → relative bone mineral density.

pair0088: Age → Bone Mineral Density

Age must be the cause, bone mineral density the effect.

D.32 Soil Properties

These data were collected within the Biodiversity Exploratories project, see http://www.biodiversity-exploratories.de. We used data set 14686 (soil texture) and 16666 (root decomposition). With the goal to study fine root decomposition rates, Solly et al. (2014)

placed litterbags containing fine roots in 150 forest and 150 grassland sites along a climate gradient across Germany. Besides the decomposition rates, a range of other relevant variables were measured, including soil properties such as clay content, soil organic carbon content and soil moisture. We deleted sites with missing values and separated grasslands and forests.

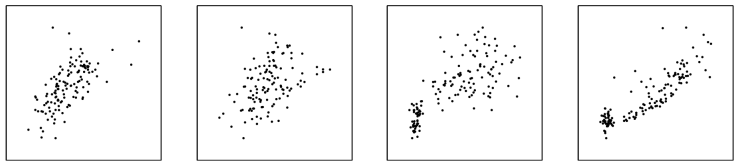


Figure 47: Scatter plots of pairs from D.32. `pair0089`: Root decomposition in April → Root decomposition in October (Forests), `pair0090`: Root decomposition in April → Root decomposition in October (Grasslands), `pair0091`: Clay content in soil → Soil moisture (forests), `pair0092`: Clay content in soil → Organic carbon content (forests).

`pair0089`–`pair0090`: Root Decomposition in April → Root Decomposition in October

Root decomposition happens monotonously in time. Hence the amount decomposed in April directly affects the amount decomposed in October in the same year.

`pair0091`: Clay Content in Soil → Soil Moisture

The amount of water that can be stored in soils depends on its texture. The clay content of a soil influences whether precipitation is stored longer in soils or runs off immediately. In contrast, it is clear that wetness of a soil does not affect its clay content.

`pair0092`: Clay Content in Soil → Organic Carbon Content

How much carbon an ecosystem stores in its soil depends on multiple factors, including the land cover type, climate and soil texture. Higher amounts of clay are favorable for storage of organic carbon (Solly et al., 2014). Soil organic carbon, on the other hand, does not alter the texture of a soil.

D.33 Runoff

This data set comes from the MOPEX data base (`http://www.nws.noaa.gov/ohd/mopex/mo_datasets.htm` and can be downloaded directly from `ftp://hydrology.nws.noaa.gov/pub/gcip/mopex/US_Data/Us_438_Daily/`. It contains precipitation and runoff data from over 400 river catchments in the USA on a daily resolution from 1948 to 2004. We computed yearly averages of precipitation and runoff for each catchment.

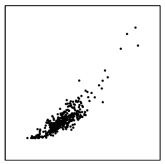


Figure 48: Scatter plots of pairs from D.33. `pair0093`: Precipitation → Runoff.

`pair0093`: Precipitation → Runoff

Precipitation is by far the largest driver for runoff in a given river catchment. There might be a very small feedback from runoff that evaporates and generates new precipitation. This is, however, negligible if the catchment does not span over full continents.

D.34 Electricity Load

This data set comes from a regional energy distributor in Turkey. It contains three variables, the hour of the day, temperature in degree Celsius and electricity consumption (load) in MW per hour. We thank S. Armagan Tarim and Steve Prestwich for providing the data.

Figure 49: Scatter plots of pairs from D.34. `pair0094`: Hour of the day → Temperature, `pair0095`: Hour of the day → Electricity consumption, `pair0096`: Temperature → Electricity consumption.

`pair0094`: Hour of the Day → Temperature

We consider hour of the day as the cause, since it can be seen as expressing the angular position of the sun. Although true interventions are unfeasible, it is commonly agreed that changing the position of the sun would result in temperature changes at a fixed location due to the different solar incidence angle.

`pair0095`: Hour of the Day → Electricity Consumption

The hour of the day constrains in many ways what people do and thus also their use of electricity. Consequently, we consider hour of the day as cause and electricity consumption as effect.

pair0096: Temperature → Electricity Consumption

Changes in temperature can prompt people to use certain electric devices, e.g., an electric heating when it is gets very cold or the usage of a fan or air conditioning when it gets very hot. Furthermore, certain machines such as computers have to be cooled more if temperatures rise. Hence we consider temperature as cause and electricity consumption as effect.

D.35 Ball Track

The data has been recorded by D. Janzing using a ball track that has been equipped with two pairs of light barriers. The first pair measures the initial speed and the second pair the speed of a ball at some later position of the track. The units are arbitrary and differ for both measurements since they are obtained by inverting the time the ball needed to pass the distance between two light barriers of one pair.

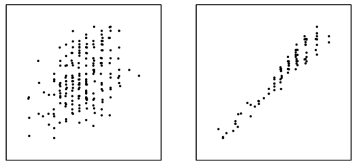


Figure 50: Scatter plots of pairs from D.35. pair0097: Initial speed → Final speed, pair0098: Initial speed → Final speed.

The initial part of the track has large slope. The initial speed is strongly determined by the exact position where the ball is put on the track. For part of the runs, the position of the ball has been chosen by D. Janzing, the other part by a 4-year old child. This should avoid that the variation of the initial position is done in a too systematic way.

Two similar experiments have been performed, using different ball track setups. For pair0098 the ball track had a longer acceleration zone than for pair0097, which allows for larger variations in initial speed.

pair0097: Initial Speed → Final Speed

These data consists of 202 measurements. Obviously, the initial speed of the ball causes the final speed.

pair0098: Initial Speed → Final Speed

These data consist of 94 measurements. Again, the initial speed of the ball causes the final speed.

D.36 Nlschools

This is data set nlschools from the R package MASS. The data were used by Snijders and Bosker (1999) as a running example and are about a study of 2287 eighth-grade pupils (aged about 11) in 132 classes in 131 schools in the Netherlands. We used two variables: lang, a language test score, and SES, the social-economic status of the pupil's family.

Figure 51: Scatter plots of pairs from D.36. pair0099: Social-economic status of family → Language test score.

pair0099: Social-Economic Status of Family → Language Test Score

We consider the social-economic status of the pupil's family to be the cause of the language test score of the pupil. However, note that selection bias may be present via the choice of the schools to include in the study.

D.37 CPUs

This is data set cpus from the R package MASS, and concerns characteristics of 209 CPUs (Ein-Dor and Feldmesser, 1987). We used two variables: syct, cycle time in nanoseconds, and perf, the published performance on a benchmark mix relative to an IBM 370/158-3, and took the logarithms of the original values.

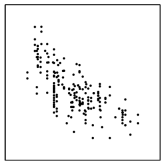


Figure 52: Scatter plots of pairs from D.37. pair0100: CPU cycle time → Performance.

pair0100: CPU Cycle Time → Performance

It should be obvious that CPU cycle time causes its performance.

Appendix E. Computation Time

We report the total computation time for each benchmark set and for each of our implementations of various methods in Figures 53 and 54. We used a machine with `Intel Xeon CPU E5-2680 v2 @ 2.80GHz` processors, 40 cores, and 125 GB of RAM. The measured computation time measures the total time spent (i.e., the sum of the computation times of individual cores). We did not spend much effort on optimizing the implementations, so the reported computation times should be seen as upper bounds on what is achievable. We only report results for the unperturbed data, as the preprocessing does not affect computation time significantly.

In general, for the ANM implementations, most time is taken by the Gaussian Process regression. The HSIC test and entropy estimators are relatively quick compared to that. A notable outlier is `ANM-MML` which spends much time on estimating the MML of the marginal distribution using the algorithm by Figueiredo and Jain (2002). IGCI implementations are much faster than ANM (about two orders of magnitude in our setting), as non-parametric regression is not required. One notable outlier for the IGCI implementations is `IGCI-ent-PSD`, which shows that the `ent-PSD` estimator is slower than the other entropy estimators in the `ITE` toolbox. Interestingly, this is also the only non-parametric entropy estimator that turned out to be robust to perturbations of the data.

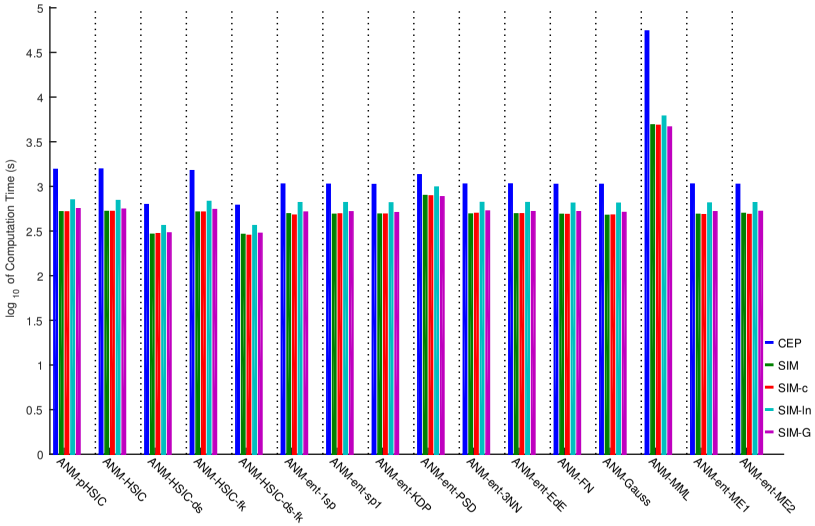


Figure 53: Computation times of various ANM methods on different (unperturbed) data sets. For the variants of the spacing estimator, only the results for `sp1` are shown, as results for `sp2`,...,`sp6` were similar.

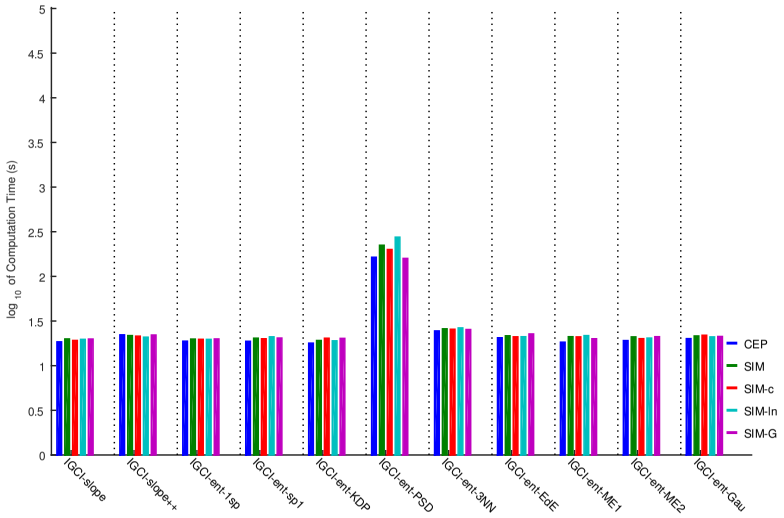


Figure 54: Computation times of various IGCI methods on different (unperturbed) data sets. For the variants of the spacing estimator, only the results for `sp1` are shown, as results for `sp2`,...,`sp6` were similar. We only show results for the uniform base measure as those for the Gaussian base measure are similar.

References

R. Armann and I. Bülthoff. Male and female faces are only perceived categorically when linked to familiar identities – and when in doubt, he is a male. *Vision Research*, 63:69–80, 2012.

A. Azzalini and A. W. Bowman. A look at some data on the Old Faithful Geyser. *Applied Statistics*, 39(3):357–365, 1990.

K. Bache and M. Lichman. UCI Machine Learning Repository, 2013. URL `http://archive.ics.uci.edu/ml`.

D. Baldocchi, E. Falge, L. Gu, R. Olson, D. Hollinger, S. Running, P. Anthoni, C. Bernhofer, K. Davis, R. Evans, J. Fuentes, A. Goldstein, G. Katul, B. Law, X. Lee, Y. Malhi, T. Meyers, W. Munger, W. Oechel, K. T. Paw, K. Pelegaard, H. P. Schmid, R. Valentini, S. Verma, T. Vesala, K. Wilson, and S. Wofsy. FLUXNET: A new tool to study the temporal and spatial variability of ecosystem-scale carbon dioxide, water vapor, and energy flux densities. *Bulletin of the American Meteorological Society*, 82(11):2415–2434, 2001.

J. Baynes and M. H. Dominiczak. *Medical Biochemistry*. Mosby, 1999.

J. Bloomer, J. W. Stehr, C. A. Piety, R. J. Salawitch, and R. R. Dickerson. Observed relationships of ozone air pollution with temperature and emissions. *Geophysical Letters*, 36(9), 2009.

K. A. Bollen. *Structural Equations with Latent Variables*. John Wiley & Sons, 1989.

E. Braunwald, A. S. Fauci, D. L. Kasper, S. L. Hauser, D. L. Long, and J. L. Jameson, editors. *Principles of Internal Medicine: Volume 2*. McGraw-Hill, 15th international edition, 2001.

P. Bühlmann, J. Peters, and J. Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6):2526–2556, 2014.

C. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26:404–413, 1934.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2006.

J. Czerniak and H. Zarzycki. Application of rough sets in the presumptive diagnosis of urinary system diseases. In J. Soldek and L. Drobiazgiewicz, editors, *Artificial Intelligence and Security in Computing Systems*, pages 41–51. Kluwer Academic Publishers, 2003.

P. Daniušis, D. Janzing, J. M. Mooij, J. Zscheischler, B. Steudel, K. Zhang, and B. Schölkopf. Inferring deterministic causal relations. In *Proceedings of the 26th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, pages 143–150, 2010.

D. Eaton and K. Murphy. Exact Bayesian structure learning from uncertain interventions. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 107–114, 2007.

F. Eberhardt and R. Scheines. Interventions and causal inference. *Philosophy of Science*, 74(5): 981–995, 2007.

N. Ebrahimi, K. Pflughoeft, and E. S. Soofi. Two measures of sample entropy. *Statistics and Probability Letters*, 20:225–234, 1994.

P. Ein-Dor and J. Feldmesser. Attributes of the performance of central processing units: a relative performance prediction model. *Communications of the ACM*, 30:308–317, 1987.

S. A. Esrey, J. B. Potash, L. Roberts, and C. Shiff. Effects of improved water supply and sanitation on ascariasis, diarrhoea, dracunculiasis, hookworm infection, schistosomiasis, and trachoma. *Bulletin of the World Health Organization*, 69(5):609, 1991.

U. Feister and K. Balzer. Surface ozone and meteorological predictors on a subregional scale. *Atmospheric Environment. Part A. General Topics*, 25(9):1781–1790, 1991.

M. A. T. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381–396, March 2002.

N. Friedman and I. Nachman. Gaussian process networks. In *Proceedings of the 16th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2000)*, pages 211–219, 2000.

K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems 20 (NIPS*2007)*, pages 489–496, 2008.

R. M. Gray. Toeplitz and circulant matrices: A review. *Foundations and Trends in Communications and Information Theory*, 2:155–239, 2006.

U. Grenander and G. Szego. *Toeplitz forms and their applications*. University of California Press, 1958.

A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Algorithmic Learning Theory*, pages 63–78. Springer-Verlag, 2005.

A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20 (NIPS*2007)*, pages 585–592, 2008.

A. Gretton. A simpler condition for consistency of a kernel independence test. *arXiv.org preprint*, arXiv:1501.06103v1 [stat.ML], January 2015. URL http://arxiv.org/abs/1501.06103v1.

P. D. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

H. A. Guvenir, B. Acar, G. Demiroz, and A. Cekin. A supervised machine learning algorithm for arrhythmia analysis. In *Proceedings of the Computers in Cardiology Conference*, 1997.

I. Guyon, D. Janzing, and B. Schölkopf. Causality: Objectives and assessment. In *JMLR Workshop and Conference Proceedings*, volume 6, pages 1–38, 2010.

I. Guyon et al. Results and analysis of 2013-2014 ChaLearn Cause-Effect Pair Challenges. Forthcoming, 2016.

L. Györfi, M. Kohler, A. Krzyżak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, 2002.

J. D. Haigh. The sun and the earths climate. *Living Reviews in Solar Physics*, 4(2):2298, 2007.

D. H. Hathaway. The solar cycle. *Living Reviews in Solar Physics*, 7:1, 2010.

K. W. Hipel and A. I. McLeod. *Time Series Modelling of Water Resources and Environmental Systems*. Elsevier, 1994.

P. O. Hoyer, S. Shimizu, A. J. Kerminen, and M. Palviainen. Estimation of causal effects using linear non-Gaussian causal models with hidden variables. *International Journal of Approximate Reasoning*, 49:362–378, 2008.

P. O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems 21 (NIPS*2008)*, pages 689–696, 2009.

A. Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In *Advances in Neural Information Processing Systems 9 (NIPS*1996)*, pages 273–279, 1997.

A. Hyvärinen and S. M. Smith. Pairwise likelihood ratios for estimation of non-Gaussian structural equation models. *Journal of Machine Learning Research*, 14:111–152, 2013.

B. Janzing. *Sonne, Wind und Schneerekorde: Wetter und Klima in Furtwangen im Schwarzwald, zum 25-jährigen Bestehen der Wetterstation*. Self-published, in German, 2004.

D. Janzing and B. Schölkopf. Causal inference using the algorithmic Markov condition. *IEEE Transactions on Information Theory*, 56(10):5168–5194, 2010.

D. Janzing, X. Sun, and B. Schölkopf. Distinguishing cause and effect via second order exponential models. *arXiv.org preprint*, arXiv:0910.5561v1 [stat.ML], October 2009. URL http://arxiv.org/abs/0910.5561v1.

D. Janzing, P. Hoyer, and B. Schölkopf. Telling cause from effect based on high-dimensional observations. In *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, pages 479–486, 2010.

D. Janzing, J. M. Mooij, K. Zhang, J. Lemeire, J. Zscheischler, P. Daniušis, B. Steudel, and B. Schölkopf. Information-geometric approach to inferring causal directions. *Artificial Intelligence*, 182–183:1–31, 2012.

S. Jegelka and A. Gretton. Brisk kernel ICA. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*, pages 225–250. MIT Press, 2007.

E. Kalnay, M. Kanamitsu, R. Kistler, W. Collins, D. Deaven, L. Gandin, M. Iredell, S. Saha, G. White, J. Woollen, Y. Zhu, A. Leetmaa, R. Reynolds, M. Chelliah, W. Ebisuzaki, W. Higgins, J. Janowiak, K. C. Mo, C. Ropelewski, J. Wang, R. Jenne, and D. Joseph. The NCEP/NCAR 40-year reanalysis project. *Bulletin of the American Meteorological Society*, 77(3):437–471, 1996.

Y. Kano and S. Shimizu. Causal inference using nonnormality. In *Proceedings of the International Symposium on Science of Modeling, the 30th Anniversary of the Information Criterion*, pages 261–270, 2003.

L. F. Kozachenko and N. N. Leonenko. A statistical estimate for the entropy of a random vector. *Problems of Information Transmission*, 23:9–16, 1987.

S. Kpotufe, E. Sgouritsa, D. Janzing, and B. Schölkopf. Consistency of causal inference under the additive noise model. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, pages 478–486, 2014.

A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physical Review E*, 69:066138, 2004.

L. Lee, M. R. Rosenzweig, and M. M. Pitt. The effects of improved nutrition, sanitation, and water quality on child health in high-mortality populations. *Journal of Econometrics*, 77(1):209–235, 1997.

J. Lemeire and D. Janzing. Replacing causal faithfulness with algorithmic independence of conditionals. *Minds and Machines*, 23(2):227–249, May 2013.

M. D. Mahecha, M. Reichstein, N. Carvalhais, G. Lasslop, H. Lange, S. I. Seneviratne, R. Vargas, C. Ammann, M. A. Arain, A. Cescatti, I. A. Janssens, M. Migliavacca, L. Montagnani, and A. D. Richardson. Global convergence in the temperature sensitivity of respiration at ecosystem level. *Science*, 329(5993):838–840, 2010.

R. Matthews. Storks deliver babies ($p = 0.008$). *Teaching Statistics*, 22(2):36–38, 2000.

M. Meyer and P. Vlachos. Statlib: Data, software and news from the statistics community, 2014. URL http://lib.stat.cmu.edu/.

A. M. Moffat. *A New Methodology to Interpret High Resolution Measurements of Net Carbon Fluxes between Terrestrial Ecosystems and the Atmosphere*. PhD thesis, Friedrich Schiller University, Jena, 2012.

J. M. Mooij and T. Heskes. Cyclic causal discovery from continuous equilibrium data. In *Proceedings of the 29th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2013)*, pages 431–439, 2013.

J. M. Mooij and D. Janzing. Distinguishing between cause and effect. In *JMLR Workshop and Conference Proceedings*, volume 6, pages 147–156, 2010.

J. M. Mooij, D. Janzing, J. Peters, and B. Schölkopf. Regression by dependence minimization and its application to causal inference. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, pages 745–52, 2009.

J. M. Mooij, O. Stegle, D. Janzing, K. Zhang, and B. Schölkopf. Probabilistic latent variable models for distinguishing between cause and effect. In *Advances in Neural Information Processing Systems 23 (NIPS*2010)*, pages 1687–1695, 2010.

J. M. Mooij, D. Janzing, T. Heskes, and B. Schölkopf. On causal discovery with cyclic additive noise models. In *Advances in Neural Information Processing Systems 24 (NIPS*2011)*, pages 639–647, 2011.

J. M. Mooij, D. Janzing, J. Zscheischler, and B. Schölkopf. CauseEffectPairs repository, 2014. URL http://webdav.tuebingen.mpg.de/cause-effect/.

C. P. Morice, J. J. Kennedy, N. A. Rayner, and P. D. Jones. Quantifying uncertainties in global and regional temperature change using an ensemble of observational estimates: The hadcrut4 data set. *Journal of Geophysical Research: Atmospheres (1984–2012)*, 117(D8), 2012.

W. Nash, T. Sellers, S. Talbot, A. Cawthorn, and W. Ford. The population biology of Abalone (Haliotis species) in Tasmania. I. Blacklip Abalone (H. rubra) from the North Coast and Islands of Bass Strait. Sea Fisheries Division, Technical Report No. 48 (ISSN 1034-3288), 1994.

H. A. Noughabi and R. A. Noughabi. On the entropy estimators. *Journal of Statistical Computation and Simulation*, 83:784–792, 2013.

C. Nowzohour and P. Bühlmann. Score-based causal learning in additive noise models. *Statistics*, 2015. doi: 10.1080/02331888.2015.1060237.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

J. Peters and P. Bühlmann. Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, 101:219–228, 2014.

J. Peters, D. Janzing, and B. Schölkopf. Causal inference on discrete data using additive noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:2436–2450, 2011.

J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15:2009–2053, 2014.

J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

D. Ramirez, J. Via, I. Santamaria, and P. Crespo. Entropy and Kullback-Leibler divergence estimation based on Szego's theorem. In *European Signal Processing Conference (EUSIPCO)*, pages 2470–2474, 2009.

C. E. Rasmussen and H. Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 2010.

C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

D. J. Rasmussen, A. M. Fiore, V. Naik, L. W. Horowitz, S. J. McGinnis, and M. G. Schlutz. Surface ozone-temperature relationships in the eastern US: A monthly climatology for evaluating chemistry-climate models. *Atmospheric Environment*, 47:142–153, 2012.

D. N. Reshef, Y. A. Reshef, H. K. Finucane, S. R. Grossman, G. McVean, P. J. Turnbaugh, E. S. Lander, M. Mitzenmacher, and P. C. Sabeti. Detecting novel associations in large data sets. *Science*, 334(6062):1518–1524, 2011.

T. Richardson and P. Spirtes. Ancestral graph Markov models. *The Annals of Statistics*, 30(4): 962–1030, 2002.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. M. Mooij. On causal and anticausal learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, pages 1255–1262, 2012.

E. Sgouritsa, D. Janzing, P. Hennig, and B. Schölkopf. Inference of cause and effect with unsupervised inverse regression. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 38, pages 847–855, 2015.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. J. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

S. Shimizu, T. Inazumi, Y. Sogawa, A. Hyvärinen, Y. Kawahara, T. Washio, P. Hoyer, and K. Bollen. DirectLiNGAM: A direct method for learning a linear non-Gaussian structural equation model. *Journal of Machine Learning Research*, 12:1225–1248, 2011.

S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich. Network modelling methods for FMRI. *NeuroImage*, 54(2):875–891, 2011.

T. A. B. Snijders and R. J. Bosker. *Multilevel Analysis. An Introduction to Basic and Advanced Multilevel Modelling*. Sage, 1999.

E. F. Solly, I. Schöning, S. Boch, E. Kandeler, S. Marhan, B. Michalzik, J. Müller, J. Zscheischler, S. E. Trumbore, and M. Schrumpf. Factors controlling decomposition rates of fine root litter in temperate forests and grasslands. *Plant and Soil*, 382(1-2):203–218, 2014.

L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13:1393–1434, May 2012.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT Press, 2nd edition, 2000.

B. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

A. Statnikov, M. Henaff, N. I. Lytkin, and C. F. Aliferis. New methods for separating causes from effects in genomics data. *BMC Genomics*, 13:S22, 2012.

W. R. Stockwell, G. Kramm, H.-E. Scheel, V. A. Mohnen, and W. Seiler. *Forest Decline and Ozone*. Springer, 1997.

D. Stowell and M. D. Plumbley. Fast multidimensional entropy estimation by k-d partitioning. *IEEE Signal Processing Letters*, 16:537–540, 2009.

D. Stoyan, H. Stoyan, and U. Jansen. *Umwelstatistik: Statistische Verarbeitung und Analyse von Umweltdaten*. Springer, 1997.

X. Sun, D. Janzing, and B. Schölkopf. Causal inference by choosing graphs with most plausible Markov kernels. In *Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics*, pages 1–11, 2006.

X. Sun, D. Janzing, and B. Schölkopf. Causal reasoning by evaluating the complexity of conditional densities with kernel methods. *Neurocomputing*, 71:1248–1256, 2008.

Z. Szabó. Information theoretical estimators toolbox. *Journal of Machine Learning Research*, 15: 283–287, 2014.

O. Tange. GNU Parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1): 42–47, Feb 2011. URL http://www.gnu.org/s/parallel.

H. Tønnesen, L. Hejberg, S. Frobenius, and J. Andersen. Erythrocyte mean cell volume–correlation to drinking pattern in heavy alcoholics. *Acta Medica Scandinavica*, 219:515–518, 1986.

U.S. Department of Commerce. Website of the U.S. Census Bureau, 1994. URL http://www.census.gov/.

B. van Es. Estimating functionals related to a density by a class of statistics based on spacings. *Scandinavian Journal of Statistics*, 19:61–72, 1992.

M. van Hulle. Edgeworth approximation of multivariate differential entropy. *Neural Computation*, 17:1903–1910, 2005.

O. Vasicek. A test for normality based on sample entropy. *Journal of the Royal Statistical Society*, 38:54–59, 1976.

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, 4th edition, 2002.

A. P. Verbyla and B. R. Cullis. Modelling in repeated measures experiments. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 39(3):341–356, 1990.

L. Wasserman. *All of Statistics*. Springer, 2004.

C. H. Wheeler. Evidence on agglomeration economies, diseconomies, and growth. *Journal of Applied Econometrics*, 18(1):79–104, 2003.

S. Wright. Correlation and causation. *Journal of Agricultural Research*, 20:557–585, 1921.

I.-C. Yeh. Modeling of strength of high performance concrete using artificial neural networks. *Cement and Concrete Research*, 28:1797–1808, 1998.

J. Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16-17):1873–1896, 2008.

K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2009)*, pages 647–655, 2009.

J. Zscheischler, D. Janzing, and K. Zhang. Testing whether linear equations are causal: A free probability theory approach. In *Proceedings of the 27th Annual Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, pages 839–847, 2011.

Journal of Machine Learning Research 17 (2016) 1-30 Submitted 5/15; Revised 10/15; Published 4/16

Multi-task Sparse Structure Learning with Gaussian Copula Models

André R. Gonçalves ANDRERIC@DCA.FEE.UNICAMP.BR
Fernando J. Von Zuben VONZUBEN@DCA.FEE.UNICAMP.BR
School of Electrical and Computer Engineering
University of Campinas
São Paulo, Brazil

Arindam Banerjee BANERJEE@CS.UMN.EDU
Computer Science Department
University of Minnesota - Twin Cities
Minneapolis, USA

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

Multi-task learning (MTL) aims to improve generalization performance by learning multiple related tasks simultaneously. While sometimes the underlying task relationship structure is known, often the structure needs to be estimated from data at hand. In this paper, we present a novel family of models for MTL, applicable to regression and classification problems, capable of learning the structure of tasks relationship. In particular, we consider a joint estimation problem of the tasks relationship structure and the individual task parameters, which is solved using alternating minimization. The task relationship revealed by structure learning is founded on recent advances in Gaussian graphical models endowed with sparse estimators of the precision (inverse covariance) matrix. An extension to include flexible Gaussian copula models that relaxes the Gaussian marginal assumption is also proposed. We illustrate the effectiveness of the proposed model on a variety of synthetic and benchmark data sets for regression and classification. We also consider the problem of combining Earth System Model (ESM) outputs for better projections of future climate, with focus on projections of temperature by combining ESMs in South and North America, and show that the proposed model outperforms several existing methods for the problem.

Keywords: multi-task learning, structure learning, Gaussian copula, probabilistic graphical model, sparse modeling

1. Introduction

In multi-task learning (MTL) one can benefit from the knowledge of the underlying structure relating the learning tasks while carrying them out simultaneously. In situations where some tasks might be highly dependent on each other, the strategy of isolating each task will not be helpful in exploiting the potential information one might acquire from other related tasks. The last few years experienced an increase of activity in this area where new methods and applications have been proposed. From the methods perspective, there have been contributions devoted to novel formulations to describe task structure and to incorporate them into the learning framework (Evgeniou and Pontil, 2004; Ji and Ye, 2009; Kim and Xing, 2010; Kumar and Daume III, 2012; Yang et al., 2013). Meanwhile MTL has been applied to problems ranging from object detection in computer vision, going through web image and video search (Wang et al., 2009), and achieving multiple microarray data set integration in computational biology (Widmer and Rätsch, 2012).

Much of the existing work in MTL assumes the existence of a priori knowledge about the task relationship structure (see Section 2). However, in many problems there is only a high level understanding of those relationships, and hence the structure of the task relationship needs to be estimated from the data. Recently, there have been attempts to explicitly model the relationship and incorporate it into the learning process (Zhang and Yeung, 2010; Zhang and Schneider, 2010; Yang et al., 2013). In the majority of these methods, the tasks dependencies are represented as unknown hyper-parameters in hierarchical Bayesian models and are estimated from the data. As will be discussed in Section 2, many of these methods are either computationally expensive or restrictive on dependence structure complexity.

In *structure learning*, we estimate the (conditional) dependence structures between random variables in a high-dimensional distribution, and major advances have been achieved in the past few years (Banerjee et al., 2008; Friedman et al., 2008; Cai et al., 2011; Wang et al., 2013). In particular, assuming sparsity in the conditional dependence structure, i.e., each variable is dependent only on a few others, there are estimators based on convex (sparse) optimization which are guaranteed to recover the correct dependence structure with high probability, even when the number of samples is small compared to the number of variables.

In this paper, we present a family of models for MTL, for regression and classification problems, which are capable of learning the structure of task relationships and parameters for individual tasks. The problem is posed as a joint estimation where parameters of the tasks and relationship structure are learned using alternating minimization. This paper is an extension of our early work (Gonçalves et al., 2014), as it further includes improvements on the task relationship modeling and can now handle a wider spectrum of problems.

The relationship structure is modeled by either imposing a prior over the features across tasks (Section 3.3) or assuming correlated residuals (Section 3.7). We can use of a variety of methods from the structure learning literature to estimate the relationships. The formulation can be extended to Gaussian copula models (Liu et al., 2009; Xue and Zou, 2012), which are more flexible as it does not rely on strict Gaussian assumptions and has shown to be more robust to outliers. The resulting estimation problems are solved using suitable first order methods, including proximal updates (Beck and Teboulle, 2009) and alternating direction method of multipliers (Boyd et al., 2011). Based on our modeling, we show that MTL can benefit from advances in the structure learning area. Moreover, any future development in the area can be readily used in the context of MTL.

The proposed Multi-task Sparse Structure Learning (MSSL) approach has important practical implications: given a set of tasks, one can just feed the data from all the tasks without any knowledge or guidance on task relationship, and MSSL will figure out which tasks are related and will also estimate task specific parameters. Through experiments on a wide variety of data sets for multi-task regression and classification, we illustrate that MSSL is competitive with and usually outperforms several baselines from the existing MTL literature. Furthermore, the task relationships learned by MSSL are found to be accurate and consistent with domain knowledge on the problem.

In addition to evaluation on synthetic and benchmark data sets, we consider the problem of predicting air surface temperature in South and North America. The goal here is to combine outputs from Earth System Models (ESMs) reported by various countries to the Intergovernmental Panel on Climate Change (IPCC), where the regression problem at each geographical location forms a task. The weight on each model at each location forms the "skill" of that model, and the hope is that outputs from skillful models in each region can be more reliable for future projections of temperature. MSSL is able to identify geographically nearby regions as related tasks, which is meaningful for temperature prediction, without any previous knowledge of the spatial location of the tasks, and outperforms baseline approaches.

The remainder of the paper is structured as follows. Section 2 briefly discusses the related work in multi-task learning. Section 3 presents an overview and gentle introduction to the proposed multi-task sparse structure learning (MSSL) approach. Section 3.3 discusses a specific form of MSSL where the task structure dependence is learned based on the task coefficients. The MSSL is extended to the Gaussian copula MSSL in Section 3.6. Section 3.7 discusses another specific form of MSSL where the task structure dependence is learned based on the task residuals. Section 4 presents experimental results on regression and classification using synthetic, benchmark, and climate data sets. We conclude in Section 5.

Notation. We denote by m the number of tasks, d the problem dimension, supposed to be the same for all learning tasks, and n_k the number of samples for the k-th task. $\mathbf{X}_k \in \mathbb{R}^{n_k \times d}$ and $\mathbf{y}_k \in \mathbb{R}^{n_k \times 1}$ are the input and output data for the k-th task. Let $\mathbf{W} \in \mathbb{R}^{d \times m}$ be the parameter matrix, where columns are vector parameters $\mathbf{w}_k \in \mathbb{R}^d$, $k = 1, ..., m$, for the tasks. $(x)_+ = \max(0, x)$. Let $\mathcal{S}_+^p$ be the set of $p \times p$ positive semidefinite matrices. For any matrix $\mathbf{A}$, $\mathrm{tr}(\mathbf{A})$ is the trace operator, $\|\mathbf{A}\|_1$ and $\|\mathbf{A}\|_F$ are the ℓ_1-norm and Frobenius norm of $\mathbf{A}$, respectively. $\mathbf{A} \circ \mathbf{B}$ denotes the Hadamard (element-wise) product of the matrices $\mathbf{A}$ and $\mathbf{B}$. $\mathbf{I}_p$ is the $p \times p$ identity matrix and $\mathbf{0}_{p \times p}$ is a matrix full of zeros. For an m-variate random variable $V = (V_1, ..., V_m)$, we denote by $V_{\setminus\{i,j\}}$ the set of marginals except i and j.

2. Related Work

MTL has attracted a great deal of attention in the past few years and consequently many algorithms have been proposed (Evgeniou and Pontil, 2004; Argyriou et al., 2007; Xue et al., 2007; Jacob et al., 2008; Obozinski et al., 2010; Zhou et al., 2011b; Zhang and Yeung, 2010; Yang et al., 2013; Gonçalves et al., 2014). We will present a general view of the methods and discuss in more details those that are more related to ours.

The majority of the proposed methods fall into the class of regularized multi-task learning, which has the form

$$\min_{\mathbf{W}} \sum_{k=1}^{m} \left(\sum_{i=1}^{n_k} \ell\left(f(\mathbf{x}_k^i, \mathbf{w}_k), y_k^i\right) \right) + \mathcal{R}(\mathbf{W}),$$

where $\ell(\cdot)$ is the loss function such as squared, logistic, and hinge loss; $\mathcal{R}(\mathbf{W})$ is a regularization function for $\mathbf{W}$ that can be designed to enforce some sharing of information between tasks. In such a context, the goal of MTL is to estimate the tasks parameters $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_m]$, while taking into account the underlying relationship among tasks.

The existing methods basically differ in the way the regularization $\mathcal{R}(\mathbf{W})$ is designed, including the structural constraints imposed to matrix $\mathbf{W}$ and the relationship among the

tasks. Some methods assume a fixed structure a priori, while others try to estimate it from the data. In the following we present a representative set of methods from these categories.

2.1 MTL with All Tasks Related

One class of MTL methods assumes that all tasks are related and the information about tasks are selectively shared among all tasks, with the hypothesized structure of the parameter matrix $\mathbf{W}$ controlling how the information is shared.

Evgeniou and Pontil (2004) considered the scenario that all tasks are related in a way that the model parameters are close to some mean model. Motivated by the sparsity inducing property of the ℓ_1-norm (Tibshirani, 1996), the idea of structured sparsity has been widely explored in MTL algorithms. Argyriou et al. (2007) assumed that there exists a subset of features that is shared for all the tasks and imposed an $\ell_{2,1}-$norm penalization on the matrix $\mathbf{W}$ to select such set of features. In the dirty-model proposed in Jalali et al. (2010) the matrix $\mathbf{W}$ is modeled as the sum of a group sparse and an element-wise sparse matrix. The sparsity pattern is imposed by ℓ_q and ℓ_1-norm regularizations. Similar decomposition was assumed in Chen et al. (2010), but there $\mathbf{W}$ is a sum of an element-wise sparse (ℓ_1) and a low-rank (nuclear norm) matrix. The assumption that a low-dimensional subspace is shared by all tasks is explored in Ando et al. (2005), Chen et al. (2009), and Obozinski et al. (2010). For example, in Obozinski et al. (2010) a trace norm regularization on $\mathbf{W}$ was used to select the common low-dimensional subspace.

2.2 MTL with Cluster Assumption

Another class of MTL methods assumes that not all tasks are related, but instead the relatedness is in a group (cluster) structure, that is, mutually related tasks are in the same cluster, while unrelated tasks belong to different clusters. Information is shared only by those tasks belonging to the same cluster. The problem then involves estimating the number of clusters and the matrix encoding the assignment cluster information.

In Bakker and Heskes (2003) task clustering was enforced by considering a mixture of Gaussians as a prior over task parameters. Evgeniou et al. (2005) proposed a task clustering regularization to encode cluster information in the MTL formulation. Xue et al. (2007) employed a Dirichlet process prior over the task coefficients to encourage task clustering and the number of clusters was someway automatically determined by the prior.

2.3 MTL with Dependence Structure Learning

Recently, there have been some proposals to estimate and incorporate the dependence among the tasks into the learning process. These methods are the most related to ours.

A matrix-variate normal distribution was used as a prior for $\mathbf{W}$ matrix in Zhang and Yeung (2010). The hyper-parameter for such a prior distribution captures the covariance matrix ($\mathbf{\Sigma}$) among all task coefficients. The resulting non-convex maximum a posteriori problem is relaxed by restricting the model complexity. It has a positive side of making the whole problem convex, but has the downside of significantly restricting the flexibility of the task relatedness structure. Also, in Zhang and Yeung (2010), the task relationship is modeled by the covariance among tasks, but uses the inverse (precision matrix, $\mathbf{\Sigma}^{-1} = \mathbf{\Omega}$)

in the task parameter learning step, therefore, the inverse of the covariance matrix needed to be computed at every iteration. We, on the other hand, do not constrain the complexity of our model and also learn the inverse of the covariance matrix directly, which tends to be more stable than computing covariance and then inverting it.

Zhang and Schneider (2010) also used a matrix-variate normal prior over $\mathbf{W}$. The two matrix hyper-parameters explicitly represent the covariance among the features (assuming the same feature relationships in all tasks) and covariance among the tasks, respectively. Sparse inducing penalization on the inverse covariance $\mathbf{\Omega}$ of both is added into the formulation. Unlike Zhang and Yeung (2010), both matrices are learned in an alternating minimization algorithm and can be computationally prohibitive in high dimensional problems due to the cost of modeling and estimating the feature covariance.

Yang et al. (2013) also assumed a matrix normal prior for $\mathbf{W}$. However, the row and column covariance hyperparameters have a Matrix Generalized Inverse Gaussian (MGIG) prior distribution. The mean of matrix $\mathbf{W}$ is factorized as the product of two matrices that also has matrix-variate normal distribution as a prior. The model inference is done via a variational Expectation Maximization (EM) algorithm. Due to the lack of a closed form expression to compute statistics of the MGIG distribution, the method resort to the use of sampling techniques, which can be slow for high-dimensional problems.

Rothman et al. (2010) also enforced sparsity on both $\mathbf{W}$ and $\mathbf{\Omega}$. Similar to our residual-based MSSL formulation, it differs in two aspects: (i) our formulation allows a richer class of conditional distribution $p(y|\mathbf{x})$, namely distributions in the exponential family, rather than simply Gaussian; and (ii) we employ a semiparametric Gaussian copula model to capture task relationship, which does not rely of Gaussian assumption on the marginals and have shown to be more robust to outliers (Liu et al., 2012), then traditional Gaussian model used in Rothman et al. (2010). As will be seen in the experiments, the MSSL method with copula models produced more accurate predictions. Rai et al. (2012) extended the formulation in Rothman et al. (2010) to model feature dependence, additionally to the task dependence modeling. However, it is computationally prohibitive for high-dimensional problems, due to the cost of estimating another precision matrix for feature dependence.

Zhou and Tao (2014) used copula as a richer class of conditional marginal distributions $p(y_k|\mathbf{x})$. As copula models express the joint distribution $p(\mathbf{y}|\mathbf{x})$ from the set of marginal distributions, this formulation allows marginals to have arbitrary continuous distributions. Output correlation is exploited via the sparse inverse covariance in the copula function, which is estimated by a procedure based on proximal algorithms. Our method also covers a rich class of conditional distributions, the exponential family that includes Gaussian, Bernoulli, Multinomial, Poisson, and Dirichlet, among others. We use Gaussian copula models to capture tasks dependence, instead of explicitly modeling marginal distributions.

3. Multi-task Sparse Structure Learning

In this section we describe our multi-task Sparse Structure Learning (MSSL) method. As our modeling is founded on structure estimation in Gaussian graphical models, we first introduce the associated problem before presenting the proposed method.

3.1 Structure Estimation in Gaussian Graphical Models

Here we describe the undirected graphical model used to capture the underlying linear dependence structure of our multi-task learning framework.

Let $V = (V_1, \ldots, V_m)$ be an m-variate random vector with joint distribution $p(V)$. Such distribution can be characterized by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the vertex set $\mathcal{V}$ represents the m covariates of V and edge set $\mathcal{E}$ represents the conditional dependence relations between the covariates of V. If V_i is conditionally independent of V_j given the other variables, then the edge (i, j) is not in $\mathcal{E}$. Assuming $V \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, the missing edges correspond to zeros in the inverse covariance matrix or *precision* matrix given by $\mathbf{\Sigma}^{-1} = \mathbf{\Omega}$, i.e., $(\mathbf{\Sigma}^{-1})_{ij} = 0 \; \forall (i, j) \notin E$ (Lauritzen, 1996).

Classical estimation approaches (Dempster, 1972) work well when m is small. Given, that we have n i.i.d. samples $v_1, \ldots, v_n$ from the distribution, the empirical covariance matrix is $\hat{\mathbf{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(v_i - \bar{v})^\top(v_i - \bar{v})$, where $\bar{v} = \frac{1}{n}\sum_{i=1}^{n} v_i$. However, when $m > n$, $\hat{\mathbf{\Sigma}}$ is rank-deficient and its inverse cannot be used to estimate the precision matrix $\mathbf{\Omega}$. Nonetheless, for a sparse graph, i.e. most of the entries in the precision matrix are zero, several methods exist to estimate $\mathbf{\Omega}$ (Friedman et al., 2008; Boyd et al., 2011).

3.2 MSSL Formulation

For ease of exposition, let us consider a simple linear model for each task: $\mathbf{y}_k = \mathbf{X}_k\mathbf{w}_k + \boldsymbol{\xi}_k$ where $\mathbf{w}_k$ is the parameter vector for task k and $\boldsymbol{\xi}_k$ denotes the residual error. The proposed MSSL method estimates both the task parameters $\mathbf{w}_k$ for all tasks and the structure dependence, based on some information from each task. Further, the dependence structure is used as inductive bias in the $\mathbf{w}_k$ learning process, aiming at improving the generalization capability of the tasks.

We investigate and formalize two ways of learning the relationship structure (a graph indicating the relationship among the tasks), represented by $\mathbf{\Omega}$: (a) modeling $\mathbf{\Omega}$ from the task specific parameters $\mathbf{w}_k, \forall k = 1, ..., m$ and (b) modeling $\mathbf{\Omega}$ from the residual errors $\boldsymbol{\xi}_k, \forall k = 1, ..., m$. Based on how we model $\mathbf{\Omega}$, we propose p-MSSL (from tasks parameters) and r-MSSL (from residual error). Both models are discussed in the following sections.

At a high level, the estimation problem in such MSSL approaches takes the form:

$$\min_{\mathbf{W}, \mathbf{\Omega} \succ 0} \; \mathcal{L}((\mathbf{Y}, \mathbf{X}), \mathbf{W}) + \mathcal{B}(\mathbf{W}, \mathbf{\Omega}) + \mathcal{R}_1(\mathbf{W}) + \mathcal{R}_2(\mathbf{\Omega}), \tag{1}$$

where $\mathcal{L}(\cdot)$ denotes suitable task specific loss function, $\mathcal{B}(\cdot)$ is the inductive bias term, and $\mathcal{R}_1(\cdot)$ and $\mathcal{R}_2(\cdot)$ are suitable sparsity inducing regularization terms. The interaction between parameters $\mathbf{w}_k$ and the relationship matrix $\mathbf{\Omega}$ is captured by the $\mathcal{B}(\cdot)$ term. Notably, when $\mathbf{\Omega}_{k,k'} = 0$, the parameters $\mathbf{w}_k$ and $\mathbf{w}_{k'}$ have no influence on each other. Sections 3.3 to 3.7 delineate the modeling details behind MSSL algorithms and how it leads to the solution of the optimization problem in (1).

3.3 Parameter Precision Structure

If the tasks are unrelated, one can learn the columns of the coefficient matrix $\mathbf{W}$ independently for each of the m tasks. However, when there exist relationships among the m tasks, learning the columns of $\mathbf{W}$ independently fails to capture these dependencies. In such a

scenario, we propose to use the precision matrix $\mathbf{\Omega} \in \mathbb{R}^{m \times m}$ in order to capture pairwise partial correlations between tasks.

In the parameter precision structure based MSSL (p-MSSL) model we assume that features across tasks (*rows* $\hat{\mathbf{w}}_j$ of the matrix $\mathbf{W}$) follows a multivariate Gaussian distribution with zero mean and covariance matrix $\mathbf{\Sigma}$, i.e., $\hat{\mathbf{w}}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})\ \forall j = 1, ..., d$, where $\mathbf{\Sigma}^{-1}$=$\mathbf{\Omega}$. The problem of interest is to estimate both the parameters $\mathbf{w}_1, \ldots, \mathbf{w}_m$ and the precision matrix $\mathbf{\Omega}$. By imposing such a prior over the *rows* of $\mathbf{W}$, we are capable of explicitly estimating the dependency structure among the tasks via the precision matrix $\mathbf{\Omega}$.

With a multivariate Gaussian prior over the *rows* of $\mathbf{W}$, its posterior can be written as

$$p\left(\mathbf{W}|(\mathbf{X}, \mathbf{Y}), \mathbf{\Omega}\right) \propto \prod_{k=1}^{m} \prod_{i=1}^{n_k} p\left(y_k^i | \mathbf{x}_k^i, \mathbf{w}_k^\top\right) \prod_{j=1}^{d} p\left(\hat{\mathbf{w}}_j | \mathbf{\Omega}\right), \tag{2}$$

where the first term in the right hand side denotes the conditional distribution of the response given the input and parameters, and the second term denotes the prior over *rows* of $\mathbf{W}$. In this paper, we consider the *penalized* maximization of (2), assuming that the parameter matrix $\mathbf{W}$ and the precision matrix $\mathbf{\Omega}$ are sparse, i.e., contain few non-zero elements. In the following, we provide two specific instantiations of this model. First, we consider a Gaussian conditional distribution, wherein we obtain the well known least squares regression problem (Section 3.3.1). Second, for discrete labeled data, choosing a Bernoulli conditional distribution leads to a logistic regression problem (Section 3.3.2).

3.3.1 Least Squares Regression

Assume that

$$p\left(y_k^i \middle| \mathbf{x}_k^i, \mathbf{w}_k\right) = \mathcal{N}_1\left(y_k^i \middle| \mathbf{w}_k^\top \mathbf{x}_k^i, \sigma_k^2\right),$$

where it is considered for ease of exposition that the variance of the residuals $\sigma_k^2 = 1, \forall k = 1, ..., m$, though it can be incorporated in the model and learned from the data. We can write this optimization problem as minimization of the negative logarithm of (2), which corresponds to a regularized linear regression problem

$$\min_{\mathbf{W}, \mathbf{\Omega} \succ 0} \quad \frac{1}{2} \sum_{k=1}^{m} \sum_{i=1}^{n_k} \left(\mathbf{w}_k^\top \mathbf{x}_k^i - y_k^i\right)^2 - \frac{d}{2} \log |\mathbf{\Omega}| + \frac{1}{2} \mathrm{tr}(\mathbf{W} \mathbf{\Omega} \mathbf{W}^\top).$$

Further, assuming that $\mathbf{\Omega}$ and $\mathbf{W}$ are sparse, we add ℓ_1-norm regularizers over both parameters. In the case one task has a much larger number of samples compared to the others, it may dominate the empirical loss term. To avoid such bias we modify the cost function and compute the weighted average of the empirical losses of the form

$$\min_{\mathbf{W}, \mathbf{\Omega} \succ 0} \quad \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} \left(\mathbf{w}_k^\top \mathbf{x}_k^i - y_k^i\right)^2 - d \log |\mathbf{\Omega}| + \lambda_0 \mathrm{tr}(\mathbf{W} \mathbf{\Omega} \mathbf{W}^\top) + \lambda_1 \|\mathbf{W}\|_1 + \lambda_2 \|\mathbf{\Omega}\|_1, \tag{3}$$

where λ_0, λ_1, and $\lambda_2 > 0$ are penalty parameters. The sparsity assumption on $\mathbf{W}$ is motivated by the fact that maybe some features are not relevant for discriminative purposes and can then be dropped out from the model. Precision matrix $\mathbf{\Omega}$ plays an important role in Gaussian graphical models because its zero entries precisely capture the conditional

independence, that is, $\mathbf{\Omega}_{ij} = 0$ if and only if $\mathbf{w}_i \perp\!\!\!\perp \mathbf{w}_j | \mathbf{W}_{\setminus\{i,j\}}$. Then, enforcing sparsity on $\mathbf{\Omega}$ will highlight the conditional independence among tasks parameters.

In this formulation, the term involving the trace of the outer product $\mathrm{tr}(\mathbf{W} \mathbf{\Omega} \mathbf{W}^\top)$ affects the *rows* of $\mathbf{W}$, such that if $\mathbf{\Omega}_{ij} \neq 0$, then $\mathbf{w}_i$ and $\mathbf{w}_j$ are constrained to be similar.

Although the problem is not jointly convex on $\mathbf{W}$ and $\mathbf{\Omega}$, it is in fact biconvex, that is, fixing $\mathbf{\Omega}$ the problem is convex on $\mathbf{W}$, and vice-versa. So, the associated biconvex function in problem (3) is split into two convex functions exhibited in (4a) and (4b). Then, one can use an alternating optimization procedure that updates $\mathbf{W}$ and $\mathbf{\Omega}$ by fixing one of them and solving the corresponding convex optimization problem (Gorski et al., 2007), given by

$$f_{\mathbf{\Omega}}(\mathbf{W}; \mathbf{X}, \mathbf{Y}, \lambda_0, \lambda_1) = \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} (\mathbf{w}_k^\top \mathbf{x}_k^i - y_k^i)^2 + \lambda_0 \mathrm{tr}(\mathbf{W} \mathbf{\Omega} \mathbf{W}^\top) + \lambda_1 \|\mathbf{W}\|_1, \tag{4a}$$

$$f_{\mathbf{W}}(\mathbf{\Omega}; \mathbf{X}, \mathbf{Y}, \lambda_0, \lambda_2) = \lambda_0 \mathrm{tr}(\mathbf{W} \mathbf{\Omega} \mathbf{W}^\top) - d \log |\mathbf{\Omega}| + \lambda_2 \|\mathbf{\Omega}\|_1. \tag{4b}$$

The alternating minimization algorithm proceeds as described in Algorithm 1. The procedure is guaranteed to converge to a *partial optimum* Gorski et al. (2007), since the original problem (3) is biconvex and convex in each argument $\mathbf{\Omega}$ and $\mathbf{W}$.

Algorithm 1: Multitask Sparse Structure Learning (MSSL) algorithm

```
Data: {X_k, y_k}_{k=1}^m.                          // training data for all tasks
Input: λ0, λ1, λ2 > 0.   // penalty parameters chosen by cross-validation
Result: W, Ω.                                      // estimated parameters
begin
    /* Ω^0 is initialized with identity matrix and                      */
    /* W^0 with random numbers in [-0.5,0.5].                           */
    Initialize Ω^0 and W^0
    t = 1
    repeat
        W^(t+1) = argmin_W f_Ω(t)(W)                // optimize W with Ω fixed
        Ω^(t+1) = argmin_Ω f_W(t+1)(Ω)              // optimize Ω with W fixed
        t = t + 1
    until stopping condition met
end
```

Update for W: The update step involving (4a) is an ℓ_1-regularized quadratic problem. Thus the problem is an ℓ_1-penalized quadratic optimization program, which we solve using established proximal gradient descent methods such as FISTA (Beck and Teboulle, 2009). The $\mathbf{W}$-step can be seen as a general case of the formulation in Subbian and Banerjee (2013) in the context of climate model combination, where in our proposal $\mathbf{\Omega}$ is any positive definite precision matrix, rather than a fixed Laplacian matrix as in Subbian and Banerjee (2013).

In the class of proximal gradient methods the cost function $h(x)$ is decomposed as $h(x) = f(x) + g(x)$, where $f(x)$ is a convex and smooth function and $g(x)$ is convex and

typically non-smooth. The accelerated proximal gradient iterates as follows

$$\begin{aligned} \mathbf{z}^{t+1} &:= \mathbf{w}_k^t + \omega^t \left(\mathbf{w}_k^t - \mathbf{w}_k^{t-1}\right) \\ \mathbf{w}_k^{t+1} &:= \mathbf{prox}_{\rho^t g}\left(\mathbf{z}^{t+1} - \rho^t \nabla f\left(\mathbf{z}^{t+1}\right)\right), \end{aligned} \tag{5}$$

where $\omega^t \in [0, 1)$ is an extrapolation parameter and ρ^t is the step size. The ω^t parameter is chosen as $\omega^t = (\eta_t - 1)/\eta_{t+1}$, with $\eta_{t+1} = (1 + \sqrt{1 + 4\eta_t^2})/2$ as done in Beck and Teboulle (2009) and ρ^t can be computed by a line search. The proximal operator associated with the ℓ_1-norm is the soft-thresholding operator

$$\mathbf{prox}_{\rho^t}(\mathbf{x})_i = (|x_i| - \rho^t)_+ \text{sign}(x_i) \tag{6}$$

The convergence rate of the algorithm is $\mathcal{O}(1/t^2)$ (Beck and Teboulle, 2009). Considering the squared loss, the gradient for the weights of the k-th task is computed as

$$\nabla f\left(\mathbf{w}_k\right) = \frac{1}{n_k}(\mathbf{X}_k^\top \mathbf{X}_k \mathbf{w}_k - \mathbf{X}_k^\top \mathbf{y}_k) + \lambda_0 \boldsymbol{\psi}_k, \tag{7}$$

where $\boldsymbol{\psi}_k$ is the k-th column of matrix $\boldsymbol{\Psi} = 2\mathbf{W}\boldsymbol{\Omega} = \frac{\partial}{\partial \mathbf{W}} \text{tr}(\mathbf{W}\boldsymbol{\Omega}\mathbf{W}^\top)$. Note that the first two terms of the gradient, which come from the loss function, are independent for each task and then can be computed in parallel.

Update for $\boldsymbol{\Omega}$: The update step for $\boldsymbol{\Omega}$ involving (4b) is known as the *sparse inverse covariance selection problem* and efficient methods have been proposed recently (Banerjee et al., 2008; Friedman et al., 2008; Boyd et al., 2011; Cai et al., 2011; Wang et al., 2013). Re-writing (4b) in terms of the sample covariance matrix $\mathbf{S}$, the minimization problem is

$$\min_{\boldsymbol{\Omega} \succ 0} \ \lambda_0 \text{tr}(\mathbf{S}\boldsymbol{\Omega}) - \log|\boldsymbol{\Omega}| + \frac{\lambda_2}{d}\|\boldsymbol{\Omega}\|_1, \tag{8}$$

where $\mathbf{S} = \frac{1}{d}\mathbf{W}^\top \mathbf{W}$. This formulation will be useful to connect to the Gaussian copula extension in the next section. As λ_2 is a user defined parameter, the factor $\frac{1}{d}$ can be incorporated into λ_2.

To solve the minimization problem (8) we use an efficient Alternating Direction Method of Multiplies (ADMM) algorithm (Boyd et al., 2011). ADMM is a strategy that is intended to blend the benefits of dual decomposition and augmented Lagrangian methods for constrained optimization. It takes the form of a *decomposition-coordination* procedure, in which the solutions to small local problems are coordinated to find a solution to a large global problem. We refer interested readers to Boyd et al. (2011) in its Section 6.5 for details on the derivation of the updates.

In ADMM, we start by forming the augmented Lagrangian function of the problem (8)

$$L_\rho(\boldsymbol{\Theta}, \mathbf{Z}, \mathbf{U}) = \lambda_0 \text{tr}(\mathbf{S}\boldsymbol{\Theta}) - \log|\boldsymbol{\Theta}| + \lambda_2\|\mathbf{Z}\|_1 + \frac{\rho}{2}\|\boldsymbol{\Theta} - \mathbf{Z} + \mathbf{U}\|_F^2 - \frac{\rho}{2}\|\mathbf{U}\|_F^2, \tag{9}$$

where $\mathbf{U}$ is the scaled dual variable. Note that the non-smooth convex function (8) is split in two functions by adding an auxiliary variable $\mathbf{Z}$, besides a linear constraint $\boldsymbol{\Theta} - \mathbf{Z} = 0$.

Given the matrix $\mathbf{S}^{(t+1)} = \frac{1}{d}(\mathbf{W}^{(t+1)})^\top \mathbf{W}^{(t+1)}$ and setting $\boldsymbol{\Theta}^0 = \boldsymbol{\Omega}^{(t)}$, $\mathbf{Z}^0 = \mathbf{0}_{m\times m}$, and $\mathbf{U}^0 = \mathbf{0}_{m\times m}$, the ADMM for the problem (8) consists of the iterations:

$$\boldsymbol{\Theta}^{l+1} = \underset{\boldsymbol{\Theta}\succ 0}{\text{argmin}} \ \ \lambda_0 \text{tr}(\mathbf{S}^{l+1}\boldsymbol{\Theta}) - \log|\boldsymbol{\Theta}| + \frac{\rho}{2}\|\boldsymbol{\Theta} - \mathbf{Z}^l + \mathbf{U}^l\|_F^2 \tag{10a}$$

$$\mathbf{Z}^{l+1} = \underset{\mathbf{Z}}{\text{argmin}} \ \ \lambda_2\|\mathbf{Z}\|_1 + \frac{\rho}{2}\|\boldsymbol{\Theta}^{l+1} - \mathbf{Z} + \mathbf{U}^l\|_F^2 \tag{10b}$$

$$\mathbf{U}^{l+1} = \mathbf{U}^l + \boldsymbol{\Theta}^{l+1} - \mathbf{Z}^{l+1}. \tag{10c}$$

The output of the ADMM is $\boldsymbol{\Omega}^{t+1} = \boldsymbol{\Theta}^L$, where L is the number of steps for convergence.

Each ADMM step can be solved efficiently. For the $\boldsymbol{\Theta}$-update, we can observe, from the first order optimality condition of (10a) and the implicit constraint $\boldsymbol{\Theta} \succ 0$, that the solution consists basically of a singular value decomposion.

The $\mathbf{Z}$-update (10b) can be computed in closed form, as follows

$$\mathbf{Z}^{l+1} = S_{\lambda_2/\rho}\Big(\boldsymbol{\Theta}^{l+1} + \mathbf{U}^l\Big), \tag{11}$$

where $S_{\lambda_2/\rho}(\cdot)$ is an element-wise soft-thresholding operator (Boyd et al., 2011). Finally, the updates for $\mathbf{U}$ in (10c) are already in closed form.

3.3.2 Log Linear Models

As described previously, our model can also be applied to classification. Let us assume that

$$p\left(y_k^i \middle| \mathbf{x}_k^i, \mathbf{w}_k\right) = \text{Be}\left(y_k^i \middle| h\left(\mathbf{w}_k^\top \mathbf{x}_k^i\right)\right),$$

where $h(\cdot)$ is the sigmoid function, and $\text{Be}(p)$ is a Bernoulli distribution. Therefore, following the same construction as in Section 3.3.1, parameters $\mathbf{W}$ and $\boldsymbol{\Omega}$ can be obtained by solving the following minimization problem:

$$\min_{\mathbf{W},\boldsymbol{\Omega}\succ 0} \sum_{k=1}^m \frac{1}{n_k} \sum_{i=1}^{n_k} \left(y_k^i \mathbf{w}_k^\top \mathbf{x}_k^i - \log(1 + e^{\mathbf{w}_k^\top \mathbf{x}_k^i})\right) + \lambda_0 \text{tr}(\mathbf{W}\boldsymbol{\Omega}\mathbf{W}^\top) - d\log|\boldsymbol{\Omega}| + \lambda_1\|\mathbf{W}\|_1 + \lambda_2\|\boldsymbol{\Omega}\|_1. \tag{12}$$

The loss function is the logistic loss, where we have considered a 2-class classification setting. In general, we can consider any generalized linear model (GLM) (Nelder and Baker, 1972), with different link functions $h(\cdot)$, and therefore different probability densities, such as Poisson, Multinomial, and Gamma, for the conditional distribution. For any such model, our framework requires the optimization of an objective function of the form

$$\min_{\mathbf{W},\boldsymbol{\Omega}\succ 0} \ \sum_{k=1}^m \mathcal{L}(\mathbf{y}_k, \mathbf{X}_k\mathbf{w}_k) + \lambda_0 \text{tr}(\mathbf{W}\boldsymbol{\Omega}\mathbf{W}^\top) - d\log|\boldsymbol{\Omega}| + \lambda_1\|\mathbf{W}\|_1 + \lambda_2\|\boldsymbol{\Omega}\|_1, \tag{13}$$

where $\mathcal{L}(\cdot)$ is a convex loss function obtained from a GLM.

Note that the objective function in (12) is similar to the one obtained for multi-task learning with linear regression in (3) in Section 3.3.1. Therefore, we use the same alternating minimization algorithm described in Section 3.3.1 to solve the problem in (12).

3.4 p-MSSL Interpretation as Using a Product of Distributions as Prior

From a probabilistic perspective, sparsity can be enforced using the so-called sparsity promoting priors, such as the Laplacian-like (double exponential) prior (Park and Casella, 2008). Accordingly, instead of exclusively assuming a multivariate Gaussian distribution as a prior for the rows of tasks parameter matrix $\mathbf{W}$, we can consider an improper prior which consists of the product of multivariate Gaussian and Laplacian distributions, of the form

$$p_{GL}\Big(\hat{\mathbf{w}}_j|\boldsymbol{\mu},\boldsymbol{\Omega},\lambda_0,\lambda_1\Big) \propto |\boldsymbol{\Omega}|^{1/2}\exp\left\{-\frac{\lambda_0}{2}(\hat{\mathbf{w}}_j-\boldsymbol{\mu})^\top\boldsymbol{\Omega}(\hat{\mathbf{w}}_j-\boldsymbol{\mu})\right\}\exp\left\{-\frac{\lambda_1}{2}\|\hat{\mathbf{w}}_j\|_1\right\}, \quad (14)$$

where we introduced the λ_0 parameter to control the strength of the Gaussian prior. By changing λ_0 and λ_1, we alter the relative effect of the two component priors in the product. Setting λ_0 to one and λ_1 to zero, we return to the exclusive Gaussian prior as in (2). Hence, p-MSSL formulation in (3) can be seen exactly (assuming sparse precision matrix in Gaussian prior) as a MAP inference of the conditional posterior distribution (with $\boldsymbol{\mu}=0$)

$$p\Big(\mathbf{W}|(\mathbf{X},\mathbf{Y}),\boldsymbol{\Omega}\Big) \propto \prod_{k=1}^{K}\prod_{i=1}^{N_k}\mathcal{N}\Big(y_k^i|\mathbf{w}_k^\top\mathbf{x}_k^i,\sigma^2\Big)\prod_{j=1}^{D}p_{GL}\Big(\hat{\mathbf{w}}_j|\boldsymbol{\Omega},\lambda_0,\lambda_1\Big). \quad (15)$$

Equivalently, the p-MSSL with GLM formulation as in (13) can be obtained by replacing the conditional Gaussian in (15) by another distribution in the exponential family.

3.5 Adding New Tasks

Suppose now that, after estimating all the tasks parameters and the precision matrix, a new task arrives and needs to be trained. This is known as the *asymmetric* MTL problem (Xue et al., 2007). Clearly, it will be computationally prohibitive in real applications to re-run the MSSL every time a new task arrives. Fortunately, MSSL can easily incorporate the new learning task into the framework using the information from the previous trained tasks.

After the arrival of the new task $\tilde{m}$, where $\tilde{m}=m+1$, the extended sample covariance matrix $\tilde{\mathbf{S}}$, computed from the parameter matrix $\mathbf{W}$, and the precision matrix $\tilde{\boldsymbol{\Omega}}$ are partitioned in the following form

$$\tilde{\boldsymbol{\Omega}}=\begin{pmatrix}\boldsymbol{\Omega}_{11} & \boldsymbol{\omega}_{12}\\ \boldsymbol{\omega}_{12}^\top & \omega_{22}\end{pmatrix} \qquad \tilde{\mathbf{S}}=\begin{pmatrix}\mathbf{S}_{11} & \mathbf{s}_{12}\\ \mathbf{s}_{12}^\top & s_{22}\end{pmatrix}$$

where $\mathbf{S}_{11}$ and $\boldsymbol{\Omega}_{11}$ are the sample covariance and precision matrix, respectively, corresponding to the previous tasks, which have already been trained and will be kept fixed during the estimation of the parameters associated with the new task.

Let $\mathbf{w}_{\tilde{m}}$ be the set of parameters associated with the new task $\tilde{m}$ and $\tilde{\mathbf{W}}=[\mathbf{W}_m\ \mathbf{w}_{\tilde{m}}]_{d\times\tilde{m}}$, where $\mathbf{W}_m$ is the matrix with the task parameters of all previous m tasks. For the learning of $\mathbf{w}_{\tilde{m}}$, we modify problem (4a) to include only those terms on which $\mathbf{w}_{\tilde{m}}$ depends

$$f_{\tilde{\boldsymbol{\Omega}}}(\mathbf{w}_{\tilde{m}};\mathbf{X}_{\tilde{m}},\mathbf{y}_{\tilde{m}},\lambda_0,\lambda_1)=\frac{1}{n_{\tilde{m}}}\sum_{i=1}^{n_{\tilde{m}}}(\mathbf{w}_{\tilde{m}}^\top\mathbf{x}_{\tilde{m}}^i-y_{\tilde{m}}^i)^2+\lambda_0\mathrm{tr}(\tilde{\mathbf{W}}\tilde{\boldsymbol{\Omega}}\tilde{\mathbf{W}}^\top)+\lambda_1\|\mathbf{w}_{\tilde{m}}\|_1, \quad (16)$$

and the same optimization methods for (4a) can be applied.

Recall that the task dependence learning problem (8) is equivalent to solving a graphical Lasso problem. Based on Banerjee et al. (2008), Friedman et al. (2008) proposed a block coordinate descent method which updates one column (and the corresponding row) of the matrix $\tilde{\boldsymbol{\Omega}}$ per iteration. They show that if $\tilde{\boldsymbol{\Omega}}$ is initialized with a positive semidefinite matrix, then the final (estimated) $\hat{\tilde{\boldsymbol{\Omega}}}$ matrix will be positive semidefinite, even if $d>m$. Setting initial values of $\boldsymbol{\omega}_{12}$ as zero and ω_{22} as one (the new task is supposed to be conditionally independent on all other previous tasks), the extended precision matrix $\tilde{\boldsymbol{\Omega}}$ is assured to be positive semidefinite. From Friedman et al. (2008), $\boldsymbol{\omega}_{12}$ and ω_{22} are obtained as:

$$\boldsymbol{\omega}_{12}=-\hat{\boldsymbol{\beta}}\theta_{22} \quad (17a)$$

$$\omega_{22}=1/(\theta_{22}-\boldsymbol{\theta}_{12}^\top\hat{\boldsymbol{\beta}}), \quad (17b)$$

where $\hat{\boldsymbol{\beta}}$ is computed from

$$\hat{\boldsymbol{\beta}}:=\arg\min_{\boldsymbol{\alpha}}\ \Big\{\frac{1}{2}\|\tilde{\boldsymbol{\Omega}}_m^{1/2}\boldsymbol{\alpha}-\tilde{\boldsymbol{\Omega}}_m^{-1/2}\mathbf{s}_{12}\|_2^2+\delta\|\boldsymbol{\alpha}\|_1\Big\}, \quad (18)$$

where $\eta>0$ and $\delta>0$ are sparsity regularization parameters; and $\boldsymbol{\theta}_{12}^\top=\tilde{\boldsymbol{\Omega}}_{11}^{-1}\hat{\boldsymbol{\beta}}$ and $\theta_{22}=s_{22}+\delta$. See Friedman et al. (2008) for further details. The problem (18) is a simple Lasso formulation for which efficient algorithms have been proposed (Beck and Teboulle, 2009; Boyd et al., 2011). Then to learn the coefficients for the new task $\tilde{m}$ and its relationship with the previous tasks, we iterate over solving (16) and (17) until convergence.

3.6 MSSL with Gaussian Copula Models

In the Gaussian graphical model associated with the problem (4b) the rows of the weight matrix $\mathbf{W}$ are assumed to be normally distributed. As such assumption may not hold in some cases, we need a more flexible model. A promising candidate is the copula model.

Copulas are class of flexible multivariate distributions that are expressed by its univariate marginals and a copula function that describes the dependence structure between the variables. Consequently, copulas decompose a multivariate distribution into its marginal distributions and the copula function connecting them. Copulas are founded on Sklar (1959) theorem which states that: *any m-variate distribution $f(V_1,...,V_m)$ with continuous marginal functions $f_1,...,f_m$ can be expressed as its copula function $C(\cdot)$ evaluated at its marginals, that is, $f(V_1,...,V_m)=C(f_1(V_1),...,f_m(V_m))$* and, conversely, *any copula function $C(\cdot)$ with marginal distributions $f_1,...,f_m$ defines a multivariate distribution.* Several copulas have been described, which typically exhibit different dependence properties. Here, we focus on the Gaussian copula that adopts a balanced combination of flexibility and interpretability that has attracted a lot of attention (Xue and Zou, 2012).

3.6.1 Gaussian Copula Distributions

The Gaussian copula $C_{\boldsymbol{\Sigma}^0}$ is the copula of an m-variate Gaussian distribution $\mathcal{N}_m(0,\ \boldsymbol{\Sigma}^0)$ with $m\times m$ positive definite correlation matrix $\boldsymbol{\Sigma}^0$

$$C(V_1,...,V_m;\boldsymbol{\Sigma}^0)=\Phi_{\boldsymbol{\Sigma}^0}\Big(\Phi^{-1}(V_1),...,\Phi^{-1}(V_m)\Big), \quad (19)$$

where Φ^{-1} is the inverse of a standard normal distribution function and $\Phi_{\mathbf{\Sigma}^0}$ is the joint distribution function of a multivariate normal distribution with mean vector zero and covariance matrix equal to the correlation matrix $\mathbf{\Sigma}^0$. Note that without loss of generality, the covariance matrix $\mathbf{\Sigma}^0$ can be viewed as a correlation matrix, as observations can be replaced by their normal-scores. Therefore, Sklar's theorem allows to construct a multivariate distribution with non-Normal marginal distributions and the Gaussian copula.

A more general formulation of the Gaussian copula is the semiparametric Gaussian copulas (Liu et al., 2009; Xue and Zou, 2012), which allows the marginals to follow any non-parametric distribution.

Definition 1 (Semiparametric Gaussian copula models) *Let $f = \{f_1, ..., f_m\}$ be a set of continuous monotone and differentiable univariate functions. An m-dimensional random variable $V = (V_1, ..., V_m)$ has a semiparametric Gaussian Copula distribution if the joint distribution of the transformed variable $f(V)$ follows a multivariate Gaussian distribution with correlation matrix $\mathbf{\Sigma}^0$, that is, $f(V) = (f_1(V_1), ..., f_m(V_m))^\top \sim \mathcal{N}_m(0, \mathbf{\Sigma}^0)$.*

From the definition we notice that the copula does not have requirements on the marginal distributions as long the monotone continuous functions $f_1, ..., f_m$ exist.

The semiparametric Gaussian copula model is completely characterized by two unknown parameters: the correlation matrix $\mathbf{\Sigma}^0$ (or its inverse, the precision matrix $\mathbf{\Omega}^0 = (\mathbf{\Sigma}^0)^{-1}$) and the marginal transformation functions $f_1, ..., f_m$. The unknown marginal distributions can be estimated by existing nonparametric methods. However, as will be seen next, when estimating the dependence parameter is the ultimate aim, one can directly estimate $\mathbf{\Omega}^0$ without explicitly computing the functions.

Let $Z = (Z_1, ..., Z_m) = (f(V_1), ..., f(V_m))$ be a set of latent variables. By the assumption of joint normality of Z, we know that $\mathbf{\Omega}^0_{ij} = 0 \iff Z_i \perp\!\!\!\perp Z_j | Z_{\setminus\{i,j\}}$. Interestingly, Liu et al. (2009) showed that $Z_i \perp\!\!\!\perp Z_j | Z_{\setminus\{i,j\}} \iff V_i \perp\!\!\!\perp V_j | V_{\setminus\{i,j\}}$, that is, variables V and Z share exactly the same conditional dependence graph. As we focus on sparse precision matrix, to estimate the parameter $\mathbf{\Omega}^0$ we can resort to the ℓ_1-penalized maximum likelihood method, the graphical Lasso problem (8).

Let $r_{1i}, ..., r_{ni}$ be the rank of the samples from variable V_i and the sample mean $\bar{r}_j = \frac{1}{n}\sum_{i=1}^n r_{ij} = \frac{n+1}{2}$. We start by reviewing the Spearman's ρ and Kendal's τ statistics:

$$\text{(Spearman's } \rho) \quad \hat{\rho}_{ij} = \frac{\sum_{t=1}^n (r_{ti} - \bar{r}_i)(r_{tj} - \bar{r}_j)}{\sqrt{\sum_{t=1}^n (r_{ti} - \bar{r}_i)^2 \cdot \sum_{t=1}^n (r_{tj} - \bar{r}_j)^2}}, \tag{20a}$$

$$\text{(Kendall's } \tau) \quad \hat{\tau}_{ij} = \frac{2}{n(n-1)} \sum_{1 \le t \le t' \le n} \text{sign}\Big((v_{ti} - v_{t'i})(v_{tj} - v_{t'j})\Big). \tag{20b}$$

We observe that Spearman's rho is computed from the ranks of the samples and Kendall's correlation is based on the concept of concordance of pairs, which in turn is also computed from the ranks r_i. Therefore, both measures are invariant to monotone transformation of the original samples and rank-based correlations such as Spearman's ρ and Kendal's τ of the observed variables V and the latent variables Z are identical. In other words, if we are only interested in estimating the precision matrix $\mathbf{\Omega}^0$, we can treat the observed variable V as the unknown variable Z, thus avoiding estimating the transformation functions $f_1, ..., f_m$.

To connect Spearman's ρ and Kendal's τ rank-based correlation to the underlying Pearson correlation in the graphical Lasso formulation (8) of the inverse covariance selection problem, for Gaussian random variables a result due to Kendall (1948) is used:

$$\hat{\mathbf{S}}^\rho_{ij} = \begin{cases} 2\sin\left(\frac{\pi}{6}\hat{\rho}_{ij}\right), & i \neq j \\ 1 & , \ i = j \end{cases}, \qquad \hat{\mathbf{S}}^\tau_{ij} = \begin{cases} \sin\left(\frac{\pi}{2}\hat{\tau}_{ij}\right), & i \neq j \\ 1 & , \ i = j. \end{cases}$$

We then replace $\mathbf{S}$ in (8) by $\hat{\mathbf{S}}^\rho$ or $\hat{\mathbf{S}}^\tau$ and the same ADMM proposed in Section 3.3.1 is applied. The MSSL algorithms with Gaussian copula models are called p-MSSL$_{\text{cop}}$ and r-MSSL$_{\text{cop}}$, for the parameter and residual-based versions, respectively.

Liu et al. (2012) suggested that the SGC models can be used as a safe replacement of the popular Gaussian graphical models, even when the data are truly Gaussian. Compared with the Gaussian graphical model (8), the only additional cost of the SGC model is the computation of the $m(m-1)/2$ pairs of Spearman's ρ or Kendal's τ statistics, for which efficient algorithms have complexity $O(m \log m)$.

Other copula distributions also exist, such as the Archimedean class of copulas (McNeil and Nešlehová, 2009), which are useful to model tail dependence. Nevertheless, Gaussian copula is a compelling distribution for expressing the intricate dependency graph structure.

3.7 Residual Precision Structure

In the residual structure based MSSL, called r-MSSL, the relationship among tasks will be modeled in terms of partial correlations among the errors $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_m)^\top$, instead of considering explicit dependencies between the coefficients $\mathbf{w}_1, \ldots, \mathbf{w}_m$ for the different tasks. To illustrate this idea, let us consider the regression scenario where $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_m)$ is a vector of desired outputs for each task, and $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_m)^\top$ are the covariates for the m tasks. The assumed linear model can be denoted by

$$\mathbf{Y} = \mathbf{X}\mathbf{W} + \boldsymbol{\xi}, \tag{21}$$

where $\boldsymbol{\xi} = \mathbf{Y} - \mathbf{X}\mathbf{W} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}^0)$. In this model, the errors are not assumed to be i.i.d., but vary jointly over the tasks following a Gaussian distribution with precision matrix $\mathbf{\Omega} = (\mathbf{\Sigma}^0)^{-1}$. Finding the dependence structure among the tasks now amounts to estimating the precision matrix $\mathbf{\Omega}$. Such models are commonly used in spatial statistics (Mardia and Marshall, 1984) in order to capture spatial autocorrelation between geographical locations. We adopt the framework in order to capture "loose coupling" between the tasks by means of a dependence in the error distribution. For example, in domains such as climate or remote sensing, there often exist noise autocorrelations over the spatial domain under consideration. Incorporating this dependence by means of the residual precision matrix is therefore more interpretable than the explicit dependence among the coefficients in $\mathbf{W}$.

Following the above definition, the multi-task learning framework can be modified to incorporate the relationship between the errors $\boldsymbol{\xi}$. We assume that the coefficient matrix $\mathbf{W}$ is fixed, but unknown. Since $\boldsymbol{\xi}$ follows a Gaussian distribution, maximizing the likelihood of the data, penalized with a sparse regularizer over $\mathbf{\Omega}$, reduces to the optimization problem

$$\min_{\mathbf{W}, \mathbf{\Omega} \succ 0} \left(\sum_{k=1}^m \frac{1}{n_k} \|\mathbf{y}_k - \mathbf{X}_k \mathbf{w}_k\|_2^2 \right) - d \log |\mathbf{\Omega}| + \lambda_0 \text{tr}\Big((\mathbf{Y} - \mathbf{X}\mathbf{W})\mathbf{\Omega}(\mathbf{Y} - \mathbf{X}\mathbf{W})^\top\Big) + \lambda_1 \|\mathbf{W}\|_1 + \lambda_2 \|\mathbf{\Omega}\|_1. \tag{22}$$

We use the alternating minimization scheme illustrated in previous sections to solve the problem in (22). Since the cost function is biconvex and convex in each of its arguments $\mathbf{W}$

and $\mathbf{\Omega}$, thus a *partial optimum* will be found (Gorski et al., 2007). Fixing $\mathbf{W}$, the problem of estimating $\mathbf{\Omega}$ is exactly the same as (8), but with the interpretation of capturing the conditional dependence among the residuals instead of the coefficients. The problem of estimating the tasks coefficients $\mathbf{W}$ will be slightly modified due to the change in the trace term, but the algorithms presented in Section 3.3.1 can still be used. Further, the model can be extended to losses other than the squared loss, used here due to the fact that $\boldsymbol{\xi}$ follows a Gaussian distribution.

Two instances of MSSL have been provided, p-MSSL and r-MSSL, along with their Gaussian copula versions, p-MSSL$_{\text{cop}}$ and r-MSSL$_{\text{cop}}$. In summary, p-MSSL and p-MSSL$_{\text{cop}}$ can be applied to both regression and classification problems. On the other hand, r-MSSL and r-MSSL$_{\text{cop}}$, can only be applied to regression problems, as the residual error of a classification problem is clearly non-Gaussian.

3.8 Complexity Analysis

The complexity of an iteration of the MSSL algorithms can be measured in terms of the complexity of its $\mathbf{W}$-step and $\mathbf{\Omega}$-step. Each iteration of the FISTA algorithm in the $\mathbf{W}$-step involves the element-wise operations, for both the $\mathbf{z}$-update and the proximal operator, which takes $\mathcal{O}(md)$ operations each. Gradient computation of the squared loss with trace penalization involves matrices multiplication which costs $\mathcal{O}(\max(mn^2d, dm^2))$ operations for dense matrix $\mathbf{W}$ and $\mathbf{\Omega}$, but can be reduced as both matrices are sparse. We are assuming that all tasks have the same number of samples n.

In an ADMM iteration, the dominating operation is clearly the SVD decomposition when solving the subproblem (10a). It costs $\mathcal{O}(m^3)$ operations. The other two steps amount to element-wise operations which costs $\mathcal{O}(m^2)$ operations. As mentioned previously, the copula-based MSSL algorithms have the additional cost of $\mathcal{O}(m\log(m))$ for computing Kendal's τ or Spearman's ρ statistics.

The memory requirements include $\mathcal{O}(md)$ for the $\mathbf{z}$ and previous weight matrix $\mathbf{W}^{(t-1)}$ in the $\mathbf{W}$-step and $\mathcal{O}(m^2)$ for the dual variable $\mathbf{U}$ and the auxiliary matrix $\mathbf{Z}$ in the ADMM for the $\mathbf{\Omega}$-step. We should mention that the complexity is evidently associated with the optimization algorithms used for solving problems 4a and 4b.

4. Experimental Results

In this section we provide experimental results to show the effectiveness of the proposed framework for both regression and classification problems.

4.1 Regression

We start with experiments on synthetic data and then move to the problem of predicting land air temperature in South and North America by the use of multi-model ensemble.

To select the penalty parameters λ_1 and λ_2 we use a stability selection procedure described in Meinshausen and Bühlmann (2010). It is a sub-sampling approach that provides a way to find stable structures and hence a principle to choose a proper amount of regularization for structure estimation. The parameter λ_0 was set to one in all experiments.

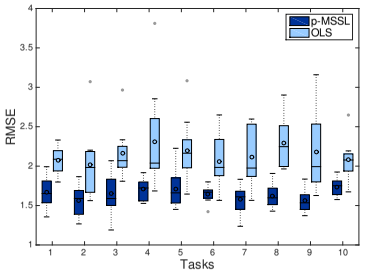


Figure 1: RMSE per task comparison between p-MSSL and Ordinary Least Square over 30 independent runs. p-MSSL gives better performance on related tasks (1-4 and 5-10).

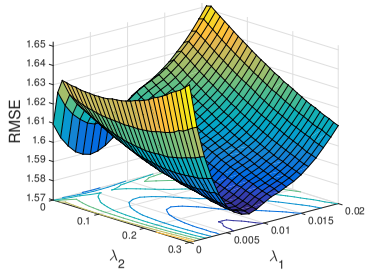


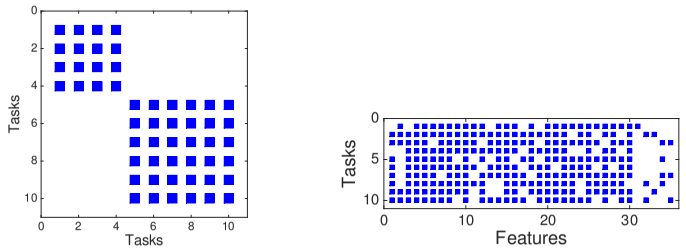
Figure 2: Average RMSE error on the test set of synthetic data for all tasks varying parameters λ_2 (controls sparsity on $\mathbf{\Omega}$) and λ_1 (controls sparsity on $\mathbf{W}$).

4.1.1 Synthetic Data set

We created a synthetic data set with 10 linear regression tasks of dimension $D = D_r + D_u$, where D_r and D_u are the number of relevant and non-relevant (unnecessary) variables, respectively. This is to evaluate the ability of the algorithm to discard non-relevant features. We used $D_r = 30$ and $D_u = 5$. For each task, the relevant input variables $\mathbf{X}'_k$ are generated i.i.d. from a multivariate normal distribution, $\mathbf{X}'_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D_r})$. The corresponding output variable is generated as $\mathbf{y}_k = \mathbf{X}'_k\mathbf{w}_k + \boldsymbol{\xi}$ where $\xi_i \sim \mathcal{N}(0,1), \forall i = 1, ..., n_k$. Unnecessary variables are generated as $\mathbf{X}''_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D_u})$. Hence, the total synthetic input data of the k-th task is formed as the concatenation of both set of variables, $\mathbf{X}_k = [\mathbf{X}'_k\ \mathbf{X}''_k]$. Note that only the relevant variables are used to produce the output variable $\mathbf{y}_k$. The parameter vectors for all tasks are chosen so that tasks 1 to 4 and 5 to 10 form two groups. Parameters for tasks 1-4 were generated as: $\mathbf{w}_k = \mathbf{w}_a \odot \mathbf{b}_k + \boldsymbol{\xi}$, where $\odot$ is the element-wise Hadamard product; and for tasks 5-10: $\mathbf{w}_k = \mathbf{w}_b \odot \mathbf{b}_k + \boldsymbol{\xi}$, where $\boldsymbol{\xi} = \mathcal{N}(\mathbf{0}, 0.2\mathbf{I}_{D_r})$. Vectors $\mathbf{w}_a$ and $\mathbf{w}_b$ are generated from $\mathcal{N}(\mathbf{0}, \mathbf{I}_{D_r})$, while $\mathbf{b}_k \sim \mathcal{U}(0,1)$ are uniformly distributed D_r-dimensional random vectors. In summary, we have two clusters of mutually related tasks. We train the p-MSSL model with 50 data instances and test it on 100 data instances.

Figure 1 shows the RMSE error for p-MSSL and for the case where Ordinary Least Squares (OLS) was applied individually for each task. As expected, sharing information among related tasks improves prediction accuracy. p-MSSL does well on related tasks 1 to 4 and 5 to 10. Figures 3a and 3b depict the sparsity pattern of the task parameters $\mathbf{W}$ and the precision matrix $\mathbf{\Omega}$ estimated by the p-MSSL algorithm. As can be seen, our model is able to recover the true dependence structure among tasks. The two clusters of tasks were clearly revealed, indicated by the filled squares, meaning non-zero entries in the precision matrix, and then, relationship among tasks. Additionally, p-MSSL was able to discard most of the irrelevant features (last five) intentionally added into the synthetic data set.

Sensitivity analysis of p-MSSL sparsity parameters λ_1 (controls sparsity on $\mathbf{W}$) and λ_2 (controls sparsity on $\mathbf{\Omega}$) on the synthetic data is presented in Figure 2. We observe that the smallest RMSE was found with a value of $\lambda_1 > 0$, which implies that a reduced set of variables is more representative than the full set, as it is indeed the case for the synthetic



(a) (b)

Figure 3: Sparsity pattern of the p-MSSL estimated parameters on the synthetic data set: (a) precision matrix $\mathbf{\Omega}$; (b) weight matrix $\mathbf{W}$. The algorithm identified the true task relationship in (a) and removed most of the non-relevant features (last five columns) in (b).

data set. The best solution is with a sparse precision matrix, as we can see in Figure 2 (smallest RMSE with $\lambda_2 > 0$). We should mention that as we increase λ_1 we encourage sparsity on $\mathbf{W}$ and, as a consequence, it becomes harder for p-MSSL to capture the true relationship among the column vectors (tasks parameters), since it learns $\mathbf{\Omega}$ from $\mathbf{W}$. This drawback is overcome in the r-MSSL algorithm, in which the precision matrix is estimated from the residuals instead of being estimated from the task parameters directly.

4.1.2 Combining Earth System Models

An Earth System Model (ESM) is a complex mathematical representation of the major climate system components (atmosphere, land surface, ocean, and sea ice), and their interactions. They are run as computer simulations, to predict climate variables such as temperature, pressure, and precipitation over multiple centuries. Several ESMs have been proposed by climate science institutes from different countries around the world.

The forecasts of future climate variables as predicted by these models have high variability, which in turn introduces uncertainty in analysis based on these predictions. One of the reasons for such uncertainty in the response of the ESMs comes from the model variability due to the fact that ESMs implements certain climatic process in different ways. Then, suitably combining outputs from multiple ESMs can greatly reduce the variability. Another equally important source of uncertainty is due to initial conditions. As ESMs are non-linear dynamic systems, changes in initial conditions can lead to different realizations of climate. In this work we focus only on the model variability. Modeling uncertainty from initial conditions is an ongoing work.

We consider the problem of combining ESM outputs for land surface temperature prediction in both South and North America, which are the world's fourth and third-largest continents, respectively, and jointly cover approximately one third of the Earth's land area. The climate is very diversified in those areas. In South America, the Amazon River basin in the north has the typical hot wet climate suitable for the growth of rain forests. The Andes Mountains, on the other hand, remain cold throughout the year. The desert regions of Chile are the driest part of South America. As for North America, the subartic climate in North

ESM	Origin	Refs.
BCC_CSM1.1	Beijing Climate Center, China	Zhang et al. (2012)
CCSM4	National Center for Atmospheric Research, USA	Washington et al. (2008)
CESM1	National Science Foundation, NCAR, USA	Subin et al. (2012)
CSIRO	Commonwealth Scient. and Ind. Res. Org., Australia	Gordon et al. (2002)
HadGEM2	Met Office Hadley Centre, UK	Collins et al. (2011)
IPSL	Institut Pierre-Simon Laplace, France	Dufresne et al. (2012)
MIROC5	Atmosphere and Ocean Research Institute, Japan	Watanabe et al. (2010)
MPI-ESM	Max Planck Inst. for Meteorology, Germany	Brovkin et al. (2013)
MRI-CGCM3	Meteorological Research Institute, Japan	Yukimoto et al. (2012)
NorESM	Norwegian Climate Centre, Norway	Bentsen et al. (2012)

Table 1: Description of the Earth System Models used in the experiments.

Canada contrasts with the semi-arid climate in western United States and Mexico's central area. The Rocky Mountains have a large impact in land's climate, and temperature significantly varies due to topographic effects (elevation and slope) (Kinel et al., 2002). Southeast of the United States is characterized by its subtropical humid climate with relatively high temperatures and an evenly distributed precipitation throughout the year.

For the experiments we use 10 ESMs from the CMIP5 data set (Taylor et al., 2012). Details about the ESMs data sets are listed in Table 1. The global observation data for surface temperature is obtained from the Climate Research Unit (CRU) at the University of East Anglia. Both, ESM outputs and observed data are the raw temperatures (not anomalies) measured in degree Celsius. We align the data from the ESMs and CRU observations to have the same spatial and temporal resolution, using publicly available climate data operators (CDO). For all the experiments, we used a $2.5^o \times 2.5^o$ grid over latitudes and longitudes in South and North America, and monthly mean temperature data for 100 years, 1901-2000, with records starting from January 16, 1901. In other words, we have two data sets: (1) South America with 250 spatial locations; and (2) North America with 490 spatial locations over land. Data sets and code are available at: `bitbucket.org/andreric/mssl-code`. For the MTL framework, each geographical location represents a task (regression problem).

From an MTL perspective, the two data sets have different levels of difficulty. North America data set has almost twice the number of tasks as compared to South America, so that we discuss the performance of MSSL in problems with high number of tasks. It brings new challenges to MTL methods. On the other hand, South America has a more diverse climate, which makes task dependence structure more complex. Preliminary results on South America were published in Gonçalves et al. (2015) employing a high-level description format.

Baselines and Evaluation: We consider the following eight baselines for comparison and evaluation of MSSL performance for the ESM combination problem. The first two baselines (MMA and Best-ESM) are commonly used in climate sciences due to their stability and simple interpretation. We will refer to these baselines and MSSL as the "models" in the sequel and the constituent ESMs as "submodels". Four well known MTL methods were also added in the comparison. The eight baselines are:

1. **Multi-model Average (MMA)**: is the current technique used by Intergovernmental Panel on Climate Change (IPCC). It gives equal weight to all ESMs at every location.

2. **Best-ESM**: uses the predicted outputs of the best ESM in the training phase (lowest RMSE). This baseline is not a combination of submodels, but a single ESM instead.

3. **Ordinary Least Squares (OLS)**: performs an ordinary least squares regression for each geographic location, independently of the others.

4. **Spatial Smoothing Multi Model Regression (S^2M^2R)**: proposed by Subbian and Banerjee (2013) to deal with ESM outputs combination, can be seen as a special case of MSSL with the pre-defined dependence matrix $\mathbf{\Omega}$ equal to the Laplacian matrix.

5. **MTL-FEAT** (Argyriou et al., 2007): all the tasks are assumed to be related and share a low-dimensional feature subspace. The following two methods, 6 and 7, can be seen as relaxations of this assumption. We used the code provided in MALSAR package (Zhou et al., 2011a).

6. **Group-MTL** (Kang et al., 2011): groups of related tasks are assumed and tasks belonging to the same group share a common feature representation. The code was taken from the author's homepage: `http://www-scf.usc.edu/~zkang/GoupMTLCode.zip`.

7. **GO-MTL** (Kumar and Daume III, 2012): founded on a relaxation of the group idea in Kang et al. (2011) by allowing subspaces shared by each group to overlap between them. We obtained the code directly from the authors.

8. **MTRL** (Zhang and Yeung, 2010): the covariance matrix among tasks coefficients is captured by imposing a matrix-variate normal prior over the coefficient matrix $\mathbf{W}$. The non-convex MAP problem is relaxed and an alternating minimization procedure is proposed to solve the convex problem. The code was taken from author's homepage: `http://www.comp.hkbu.edu.hk/~yuzhang/codes/MTRL.zip`.

Methodology

We assume here that sub-models skills are stationary, that is, the coefficient associated with each sub-model does not change over time. To have an overall measure of the capability of the method, we considered scenarios with different amount of data available for training. For each scenario, the same number of training data (columns of Table 2) are used for all tasks, and the remaining data is used for test. Starting from one year of temperature measures (12 samples), we increase till ten years of data for training. The remained data was used as test set. For each scenario 30 independent runs are performed. Therefore, the results are reported as the average and standard deviation of RMSE for all scenarios.

Results

Table 2 report the average and standard deviation RMSE for all locations in South and North America. In South America, except for the smallest training sample (12 months) the average model (MMA) has the highest RMSE for all training sample size. Best-ESM presented a better temperature future projection compared to MMA. Generally speaking, the MTL methods performed significantly better than non-MTL ones, particularly when a small number of samples are available for training. As the spatial smoothness assumption is true for temperature, S^2M^2R obtained results comparable with those yielded by MTL methods. However, this assumption does not hold for other climate variables, such as

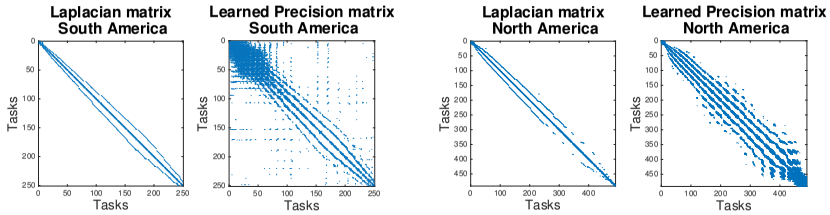


Figure 4: Laplacian matrix (on grid graph) assumed by S^2M^2R and the precision matrix learned by r-MSSL$_{\text{cop}}$ on both South and North America. r-MSSL$_{\text{cop}}$ can capture spatial relations beyond immediate neighbors. While South America is densely connected in the Amazon forest area (corresponding to top left corner) along with many spurious connections, North America is more spatially smooth.

precipitation and S^2M^2R may not succeed in those problems. On the other hand, MTL methods are general enough and in principle can be used for any climate variable. In the realm of MTL methods, all the four MSSL instantiations outperform the four other MTL contenders. It is worth observing that the two MSSL methods based on Gaussian Copula models provided smaller RMSE than the two with Gaussian models, particularly for problems with small training sample size. As Gaussian Copula models are more flexible, it is able to capture a wider range of task dependences than ordinary Gaussian models.

Similar behavior is observed in North America data set, except for the fact that MMA does much better than Best-ESM for all training sample settings. Again, all the four MSSL instantiations provided better future temperature projection. We also note that the residual-based structure dependence modeling with Gaussian Copula, r-MSSL$_{\text{cop}}$, produced slightly better results than the other three MSSL instantiations. As will be left clear in Figures 4 and 6, residual-based MSSL coherently captures related geographical locations, indicating that it can be used as an alternative to parameter-based task dependence modeling.

Figure 4 shows the precision matrix estimated by the r-MSSL$_{\text{cop}}$ algorithm and the Laplacian matrix assumed by S^2M^2R in both South and North America. Blue dots means negative entries in the matrix, while red, positive. Interpreting the entries of the matrix in terms of partial correlation, $\mathbf{\Omega}_{ij} < 0$ means positive partial correlation between $\mathbf{w}_i$ and $\mathbf{w}_j$, while $\mathbf{\Omega}_{ij} > 0$ means negative partial correlation. Not only is the precision matrix for r-MSSL$_{\text{cop}}$ able to capture the relationship among a geographical locations' immediate neighbors (as in a grid graph) but it also recovers relationships between locations that are not immediate neighbors. The plots also provide an information of the range of neighborhood influence, which can be useful in spatial statistics analysis.

The RMSE per geographical location for r-MSSL$_{\text{cop}}$ and three common approaches used in climate sciences, MMA, Best-ESM, and OLS, are shown in Figures 5. As previously mentioned, South and North America have a diverse climate and not all of the ESMs are designed to take into account and capture this scenario. Hence, averaging the model outputs, as done by MMA, reduces prediction accuracy. On the other hand r-MSSL$_{\text{cop}}$

Algorithms	Months									
	12	24	36	48	60	72	84	96	108	120
South America										
Best-ESM	1.61 (0.02)	1.56 (0.01)	1.54 (0.01)	1.53 (0.01)	1.53 (0.01)	1.53 (0.01)	1.52 (0.01)	1.52 (0.01)	1.52 (0.01)	1.52 (0.00)
MMA	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)	1.68 (0.00)
OLS	3.53 (0.45)	1.16 (0.04)	1.03 (0.02)	0.97 (0.01)	0.94 (0.01)	0.92 (0.01)	0.91 (0.01)	0.90 (0.01)	0.89 (0.01)	0.88 (0.00)
S^2M^2R	1.06 (0.03)	0.98 (0.03)	0.94 (0.01)	0.92 (0.01)	0.91 (0.01)	0.90 (0.01)	0.89 (0.01)	0.88 (0.01)	0.88 (0.01)	0.88 (0.00)
Group-MTL	1.09 (0.04)	1.01 (0.04)	0.96 (0.01)	0.93 (0.01)	0.92 (0.01)	0.91 (0.01)	0.90 (0.01)	0.89 (0.01)	0.89 (0.01)	0.88 (0.00)
GO-MTL	1.11 (0.04)	0.98 (0.03)	0.94 (0.01)	0.92 (0.01)	0.92 (0.01)	0.91 (0.01)	0.90 (0.01)	0.90 (0.01)	0.89 (0.01)	0.89 (0.00)
MTL-FEAT	1.05 (0.04)	0.99 (0.04)	0.94 (0.01)	0.92 (0.01)	0.91 (0.01)	0.90 (0.01)	0.89 (0.01)	0.88 (0.01)	0.88 (0.01)	0.88 (0.00)
MTRL	1.01 (0.04)	0.97 (0.03)	0.95 (0.02)	0.95 (0.02)	0.94 (0.01)	0.94 (0.01)	0.94 (0.01)	0.94 (0.01)	0.94 (0.01)	0.93 (0.01)
p-MSSL	1.02 (0.03)	0.94* (0.03)	0.90* (0.01)	0.89* (0.01)	0.88* (0.01)	0.88* (0.01)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	**0.86*** (**0.00**)
p-MSSL$_{\text{cop}}$	**0.98*** (**0.03**)	0.93* (0.03)	**0.90*** (**0.01**)	0.89* (0.01)	**0.88*** (**0.01**)	**0.88*** (**0.01**)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	0.87* (0.00)
r-MSSL	1.02 (0.03)	0.94* (0.03)	0.91* (0.01)	0.89* (0.01)	0.89* (0.01)	0.88* (0.01)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	**0.86*** (**0.00**)
r-MSSL$_{\text{cop}}$	1.00 (0.03)	**0.93*** (**0.03**)	**0.90*** (**0.01**)	**0.89*** (**0.01**)	**0.88*** (**0.01**)	0.88* (0.01)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	**0.87*** (**0.01**)	0.87 (0.00)
North America										
Best-ESM	3.85 (0.07)	3.75 (0.05)	3.70 (0.04)	3.68 (0.04)	3.64 (0.03)	3.64 (0.03)	3.61 (0.02)	3.60 (0.02)	3.60 (0.02)	3.58 (0.02)
MMA	2.94 (0.00)	2.94 (0.00)	2.94 (0.01)	2.94 (0.01)	2.94 (0.01)	2.94 (0.01)	2.94 (0.01)	2.94 (0.01)	2.94 (0.01)	2.94 (0.01)
OLS	10.06 (1.16)	3.37 (0.09)	2.96 (0.07)	2.79 (0.05)	2.69 (0.03)	2.64 (0.04)	2.59 (0.02)	2.56 (0.02)	2.54 (0.02)	2.53 (0.03)
S^2M^2R	3.14 (0.17)	2.79 (0.05)	2.70 (0.05)	2.64 (0.03)	2.59 (0.03)	2.56 (0.03)	2.54 (0.02)	2.52 (0.02)	2.51 (0.02)	2.50 (0.02)
Group-MTL	2.83 (0.13)	2.69 (0.04)	2.64 (0.04)	2.60 (0.03)	2.57 (0.02)	2.54 (0.03)	2.52 (0.02)	2.51 (0.01)	2.50 (0.02)	2.50 (0.02)
GO-MTL	3.02 (0.15)	2.73 (0.05)	2.63 (0.05)	2.58 (0.04)	2.53 (0.03)	2.51 (0.03)	2.49 (0.02)	2.49 (0.02)	2.48 (0.02)	2.48 (0.02)
MTL-FEAT	2.76 (0.12)	2.62 (0.04)	2.59 (0.04)	2.57 (0.03)	2.53 (0.02)	2.52 (0.02)	2.50 (0.02)	2.49 (0.01)	2.49 (0.01)	2.48 (0.02)
MTRL	2.93 (0.17)	2.83 (0.10)	2.78 (0.09)	2.81 (0.09)	2.75 (0.04)	2.77 (0.05)	2.75 (0.04)	2.76 (0.04)	2.75 (0.05)	2.77 (0.04)
p-MSSL	**2.71*** (**0.11**)	2.58* (0.05)	2.53* (0.04)	2.53* (0.04)	2.49* (0.02)	2.50* (0.02)	2.49 (0.02)	2.49 (0.01)	2.48 (0.02)	2.49 (0.01)
p-MSSL$_{\text{cop}}$	**2.71*** (**0.11**)	**2.57*** (**0.05**)	**2.52*** (**0.04**)	2.52* (0.04)	2.49* (0.02)	2.49* (0.02)	2.48* (0.02)	2.48* (0.01)	2.47 (0.02)	2.48 (0.01)
r-MSSL	2.71* (0.11)	2.58* (0.05)	2.53* (0.04)	2.53* (0.04)	2.49* (0.02)	2.49* (0.02)	2.49 (0.02)	2.48 (0.01)	2.48 (0.02)	2.49 (0.01)
r-MSSL$_{\text{cop}}$	2.71* (0.11)	**2.57*** (**0.05**)	**2.52*** (**0.04**)	**2.52*** (**0.04**)	**2.48*** (**0.02**)	**2.49*** (**0.02**)	**2.48*** (**0.02**)	**2.48*** (**0.01**)	**2.47*** (**0.02**)	**2.48** (**0.01**)

Table 2: Mean and standard deviation over 30 independent runs for several amounts of monthly data used for training. The symbol "*" indicates statistically significant (t-test, P < 0.05) improvement when compared to the best contender. MSSL with Gaussian copula provides better prediction accuracy.

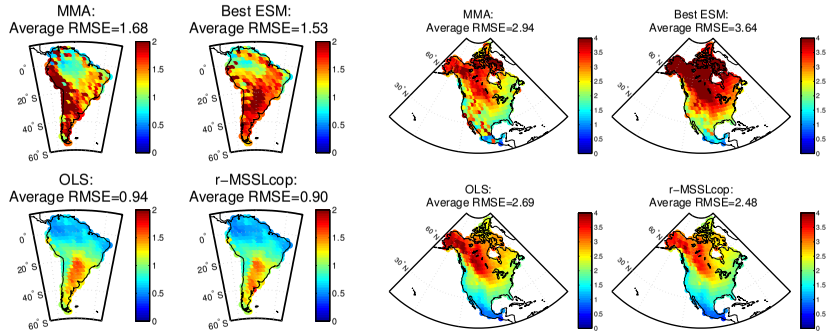


Figure 5: [Best viewed in color] RMSE per location for r-MSSL$_{\text{cop}}$ and three common methods in climate sciences, computed using 60 monthly temperature measures for training. It shows that r-MSSL$_{\text{cop}}$ substantially reduces RMSE, particularly in Northern South America and Northwestern North America.

performs better because it learns a more appropriate weight combination on the model outputs and incorporates spatial smoothing by learning the task relationship.

Figure 6 presents the dependence structure estimated by r-MSSL$_{\text{cop}}$ for South and North America data sets. Blue connections indicate dependent regions.

We immediately observe that locations in the northwest part of South America are densely connected. This area has a typical tropical climate and comprises the Amazon rainforest which is known for having hot and humid climate throughout the year with low temperature variation (Ramos, 2014). The cold climates which occur in the southernmost parts of Argentina and Chile are clearly highlighted. Such areas have low temperatures throughout the year, but there are large daily variations (Ramos, 2014). An important observation can be made about South America west cost, ranging from central Chile to Venezuela passing through Peru which has one of the driest deserts in the world. These areas are located to the left side of Andes Mountains and are known for arid climate. The average model is not performing well on this region compared to r-MSSL$_{\text{cop}}$. We can see the long lines connecting these coastal regions, which probably explains the improvement in terms of RMSE reduction achieved by r-MSSL$_{\text{cop}}$. The algorithm uses information from related locations to enhance its performance on these areas.

In the structure learned for North America, a densely connected area is also observed in the northeast part of North America, particularly the regions of Nunavut and North Quebec, which are characterized by their polar climate, with extremely severe winters and cold summers. Long connections between Alaska and regions from Northwestern Canada, which share similar climate patterns, can also be seen. Again, the r-MSSL$_{\text{cop}}$ algorithm had no access to the latitude and longitude of the locations during the training phase. r-MSSL$_{\text{cop}}$ also identified related regions, in terms of model skills, in the Gulf of Mexico. We hypothesize that no more connections were seen due to the high variability in air and sea

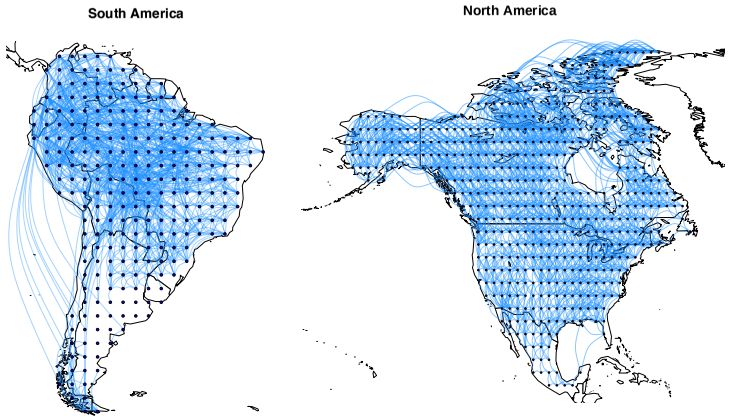


Figure 6: Relationships between geographical locations estimated by the r-MSSL$_{\text{cop}}$ algorithm using 120 months of data for training. The blue lines indicate that connected locations are conditionally dependent on each other. As expected, temperature is very spatially smooth, as we can see by the high neighborhood connectivity, although some long range connections are also observed.

surface temperature in these area (Twilley, 2001), that in turn has a strong impact on Gulf of Mexico costal regions.

4.1.3 MSSL sensitivity to initial values of $\mathbf{W}$

As discussed earlier, the MSSL algorithms may be susceptible to the choice of initial values of the parameters $\mathbf{\Omega}$ and $\mathbf{W}$, as the optimization function (3) is not jointly convex on $\mathbf{\Omega}$ and $\mathbf{W}$. In this section we analyze the impact of different parameter initializations on the RMSE and the number of non-zero entries in the estimated $\mathbf{\Omega}$ and $\mathbf{W}$ parameters.

Table 3 shows the mean and standard deviation over 10 independent runs with random initialization of $\mathbf{W}$ in the interval [-0.5,0.5] for the South America data set. For the $\mathbf{\Omega}$ matrix we started with an identity matrix, as it is reasonable to assume tasks independence beforehand. The results showed that the solutions are not sensitive to initial values of $\mathbf{W}$. The largest variation was found in the number of non-zero entries in the $\mathbf{\Omega}$ matrix for North America data set. However, it corresponds to 0.07% of the average number of non-zero entries and was not enough to significantly alter the RMSE of the solutions. Figure 7 shows the convergence of p-MSSL for several random initializations of $\mathbf{W}$. We note that in all runs the cost function decreases smoothly and similarly to each other, showing the stability of the method.

	Synt.	South America	North America
RMSE	1.14 (2e-6)	0.86 (0)	2.46 (1.6e-4)
# nz $\mathbf{W}$	345 (0)	2341 (0.32)	4758 (2.87)
# nz $\mathbf{\Omega}$	55(0)	4954 (0.63)	73520 (504.4)

Table 3: p-MSSL sensitivity to initial values of $\mathbf{W}$ in terms of RMSE, mean and standard deviation, and number of non-zero entries in $\mathbf{W}$ and $\mathbf{\Omega}$.

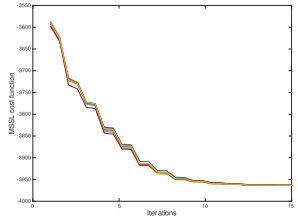


Figure 7: Convergence behavior of p-MSSL for distinct initializations of the weight matrix $\mathbf{W}$.

4.2 Classification

We test the performance of the p-MSSL on five data sets (six problems) described below. Recall that r-MSSL can not be applied for classification problems, once it relies on a Gaussian assumption of the residuals. This is currently the subject of an ongoing work. All data sets were standardized, then all features have zero mean and standard deviation one.

- **Landmine Detection:** Data from 19 landmine fields were collected, which have distinct characteristics. Each object in a given data set is represented by a 9-dimensional feature vector and the corresponding binary label (1 for landmine and 0 for clutter) (Xue et al., 2007). The feature vectors are extracted from radar images, concatenating four moment-based features, three correlation-based features, one energy ratio feature and one spatial variance feature. The goal is to classify between mine or clutter.

- **Spam Detection:** E-mail spam data set from ECML 2006 discovery challenge. It consists of two problems: Problem A with e-mails from 3 different users (2500 e-mails per user); and Problem B with e-mails from 15 distinct users (400 e-mails per user). We performed feature selection to get the 500 most informative variables using the Laplacian score feature selection algorithm (He et al., 2006). The goal is to classify between spam vs. ham. For both problems, we create different tasks for different users. This data set can be found at `http://www.ecmlpkdd2006.org/challenge.html`.

- **MNIST** data set consists of 28×28-size images of hand-written digits from 0 through 9. We transform this multiclass classification problem by applying the all-versus-all decomposition, leading to 45 binary classification problems (tasks). When a new test sample arrives, a voting scheme is performed among the classifiers. The number of samples for each classification problem is about 15000. This data set can be found at `http://yann.lecun.com/exdb/mnist/`.

- **Letter:** The handwritten letter data set consists of eight tasks, with each one being a binary classification of two letters: a/g, a/o, c/e, f/t, g/y, h/n, m/n and i/j. The input for each data point consists of 128 features representing the pixel values of the handwritten letter. The number of data points for each task varies from 3057 to 7931. This data set can be found at `http://ai.stanford.edu/~btaskar/ocr/`.

- **Yale-faces**: The face recognition data set contains 165 grayscale images with dimension 32x32 pixels of 15 individuals. Similar to MNIST, the problem is also transformed by all-versus-all decomposition, totalling 105 binary classification problems (tasks). For each task only 22 samples are available. This data set can be found at `http://vision.ucsd.edu/content/yale-face-database`.

Baseline algorithms: Four baseline algorithms were considered in the experiments and the regularization parameters for all algorithms were selected using cross-validation from the set $\{0.01, 0.1, 1, 10, 100\}$. The algorithms are:

1. **Logistic Regression (LR)**: learns separate logistic regression models for each task.
2. **MTL-FEAT** (Argyriou et al., 2007): employs an $\ell_{2,1}$-norm regularization term to capture the task relationship from multiple related tasks constraining all models to share a common set of features.
3. **CMTL** (Zhou et al., 2011b): incorporates a regularization term to induce clustering between tasks and then share information only to tasks belonging to the same cluster.
4. **Low rank MTL** (Abernethy et al., 2006): assumes that related tasks share a low dimensional subspace and applies a trace regularization norm to capture that relation.

Results: Table 4 shows the results of each algorithm for all data sets. It was obtained over 10 independent runs using a holdout cross-validation (2/3 for training and 1/3 for test). The performance of each run is measured by the average of the performance of all tasks.

Algorithms	Landmine	Spam 3-users	Spam 15-users	MNIST	Letter	Yale faces
LR	6.01 (0.37)	6.62 (0.99)	16.46 (0.67)	9.80 (0.19)	5.56 (0.19)	26.04 (1.26)
CMTL	5.98 (0.32)	3.93 (0.45)	8.01 (0.75)	2.06 (0.14)	8.22 (0.25)	9.43 (0.78)
MTL-FEAT	6.16 (0.31)	3.33 (0.43)	7.03 (0.67)	2.61 (0.08)	11.66 (0.29)	7.15 (1.60)
Trace	5.75 (0.28)	2.65 (0.32)	5.40 (0.54)	2.27 (0.09)	5.90 (0.21)	7.49 (1.72)
p-MSSL	**5.68** (**0.37**)	1.90* (0.27)	6.55 (0.68)	1.96* (0.08)	5.34* (0.19)	9.58 (0.91)
p-MSSL$_{\text{cop}}$	**5.68** (**0.35**)	**1.77*** (**0.29**)	**5.32** (**0.45**)	**1.95*** (**0.08**)	**5.29*** (**0.19**)	**5.28*** (**0.45**)

Table 4: Average classification error rates and standard deviation over 10 independent runs for all methods and data sets considered. Bold values indicate the best value and the symbol "*" means significant statistical improvement of the MSSL algorithm in relation to the contenders determined by t-test with $\mathrm{P} < 0.05$.

For all data sets p-MSSL$_{\text{cop}}$ presented statistically significant better results than the contenders for the most of the data sets. The three MTL methods presume the structure of the matrix $\mathbf{W}$, which may not be a proper choice for some problems. Such disagreement in the structure of the problem might explains the poor results in some data sets.

Focusing the analysis on p-MSSL and the copula version, p-MSSL$_{\text{cop}}$, their results are relatively similar for most of the data set, except for *Yale-faces*, where the difference is quite large. The two algorithms differ only in the way the inverse covariance matrix $\mathbf{\Omega}$ is computed. One hyphotesis for p-MSSL$_{\text{cop}}$ superiority on *Yale-faces* data set is that it may have captured hidden important dependencies among tasks, as the Copula Gaussian model can capture a wider class of dependencies than traditional Gaussian graphical models.

For the *Yale-faces* data set, which contains the smallest number of data available for training, all the MTL algorithms obtained considerable improvement compared to the traditional single task learning approach (LR), corroborating with the assertion that MTL approaches are particularly suitable for problems with few data samples.

5. Conclusion

We proposed a framework for multi-task structure learning. Our framework simultaneously learns the tasks and their relationship, with the task dependences defined as edges in an undirected graphical model. The formulation allows the direct extension of the graphical model to the recently developed semiparametric Gaussian copula models. As such model does not rely on the Gaussian assumption of task parameters, it gives more flexibility to capture hidden task conditional independence, thus helping to improve prediction accuracy. The problem formulation leads to a bi-convex optimization problem which can be efficiently solved using alternating minimization. We show that the proposed framework is general enough to be specialized to Gaussian models and generalized linear models. Extensive experiments on benchmark and climate data sets for regression tasks and real-world data sets for classification tasks illustrate that structure learning not only improves multi-task prediction performance, but also captures relevant qualitative behaviors among tasks.

Acknowledgments

We thank the AE and the three anonymous reviewers for their valuable comments and suggestions. The research was supported by NSF grants IIS-1029711, IIS-0916750, IIS-0953274, CNS-1314560, IIS-1422557, CCF-1451986, IIS-1447566, and by NASA grant NNX12AQ39A. AB acknowledges support from IBM and Yahoo. FJVZ thanks to CNPq for the financial support. ARG was supported by Science without Borders grant from CNPq, Brazil. Computing facilities were provided by University of Minnesota Supercomputing Institute (MSI).

References

J. Abernethy, F. Bach, T. Evgeniou, and J.P. Vert. Low-rank matrix factorization with attributes. Technical Report N-24/06/MM, Ecole des mines de Paris, France, 2006.

R. Ando, T. Zhang, and P. Bartlett. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. of Machine Learning Research*, 6:1817–1853, 2005.

A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 41–50, 2007.

B. Bakker and T. Heskes. Task clustering and gating for bayesian multitask learning. *J. of Machine Learning Research*, 4:83–99, 2003.

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *J. of Machine Learning Research*, 9:485–516, 2008.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

M. Bentsen et al. The Norwegian Earth System Model, NorESM1-M-Part 1: Description and basic evaluation. *Geoscience Model Development Discussion*, 5:2843–2931, 2012.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundation and Trends in Machine Learning*, pages 1–122, 2011.

V. Brovkin, L. Boysen, T. Raddatz, V. Gayler, A. Loew, and M. Claussen. Evaluation of vegetation cover and land-surface albedo in MPI-ESM CMIP5 simulations. *Journal of Advances in Modeling Earth Systems*, pages 48–57, 2013.

T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *J. of the American Statistical Association*, 106(494):594–607, 2011.

J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning shared structures from multiple tasks. In *International Conference on Machine Learning*, pages 137–144, 2009.

J. Chen, J. Liu, and J. Ye. Learning incoherent sparse and low-rank patterns from multiple tasks. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 1179–1188, 2010.

W.J. Collins et al. Development and evaluation of an Earth-system model–HadGEM2. *Geoscience Model Development Discussion*, 4:997–1062, 2011.

A.P. Dempster. Covariance selection. *Biometrics*, pages 157–175, 1972.

J.L. Dufresne et al. Climate change projections using the IPSL-CM5 Earth System Model: from CMIP3 to CMIP5. *Climate Dynamics*, pages 2123–2165, 2012.

T. Evgeniou and M. Pontil. Regularized multi–task learning. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 109–117, 2004.

T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *J. of Machine Learning Research*, 6:615–637, 2005.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9 3:432–441, 2008.

A.R. Gonçalves, P. Das, S. Chatterjee, V. Sivakumar, F.J. Von Zuben, and A. Banerjee. Multi-task Sparse Structure Learning. In *ACM International Conference on Information and Knowledge Management*, pages 451–460, 2014.

A.R. Gonçalves, F.J. Von Zuben, and A. Banerjee. A Multi-Task Learning View on Earth System Model Ensemble. *Computing in Science & Engineering*, pages 35–42, Dec. 2015.

H.B. Gordon et al. *The CSIRO Mk3 climate system model*, volume 130. CSIRO Atmospheric Research, 2002.

J. Gorski, F. Pfeuffer, and K. Klamroth. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, 66 (3):373–407, 2007.

X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, pages 507–514, 2006.

L. Jacob, F. Bach, and J.P. Vert. Clustered Multi-Task Learning: A Convex Formulation. In *Advances in Neural Information Processing Systems*, pages 745–752, 2008.

A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A Dirty Model for Multi-task Learning. *Advances in Neural Information Processing Systems*, pages 964–972, 2010.

S. Ji and J. Ye. An Accelerated Gradient Method for Trace Norm Minimization. In *International Conference on Machine Learning*, pages 457–464, 2009.

Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *International Conference on Machine Learning*, pages 521–528, 2011.

M.G. Kendall. *Rank correlation methods*. Charles Griffin & Company, 1948.

S. Kim and E.P. Xing. Tree-Guided Group Lasso for MultiTask Regression with Structured Sparsity. In *International Conference on Machine Learning*, pages 543–550, 2010.

T.G.F. Kinel, P.E. Thornton, J. A. Royle, and T.N. Chase. Climates of the Rocky Mountains: historical and future patterns. *Rocky Mountain futures: an ecological perspective*, page 59, 2002.

A. Kumar and H. Daume III. Learning task grouping and overlap in multi-task learning. In *International Conference on Machine Learning*, pages 1383–1390, 2012.

S. L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford, 1996.

H. Liu, J. Lafferty, and L. Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *J. of Machine Learning Research*, 10:2295–2328, 2009.

H. Liu, F. Han, M. Yuan, J. Lafferty, and L. Wasserman. High Dimensional Semiparametric Gaussian Copula Graphical Models. *The Annals of Statistics*, 40(40):2293–2326, 2012.

K.V. Mardia and R.J. Marshall. Maximum likelihood estimation of models for residual covariance in spatial regression. *Biometrika*, 71(1):135–146, 1984.

A. J. McNeil and J. Nešlehová. Multivariate Archimedean copulas, $d-$monotone functions and ℓ_1-norm symmetric distributions. *The Annals of Statistics*, pages 3059–3097, 2009.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society B*, 72(4):417–473, 2010.

J.A. Nelder and R.J. Baker. *Generalized linear models*. Wiley Online Library, 1972.

G. Obozinski, B. Taskar, and M. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, pages 231–252, 2010.

T. Park and G. Casella. The Bayesian lasso. *J. of the American Statistical Association*, 103 (482):681–686, 2008.

P. Rai, A. Kumar, and H. Daume III. Simultaneously leveraging output and task structures for multiple-output regression. In *Advances in Neural Information Processing Systems*, pages 3185–3193, 2012.

V.A. Ramos. South America. In *Encyclopaedia Britannica Online Academic Edition*. Encyclopaedia Britannica, Inc., 2014. URL `http://www.britannica.com/EBchecked/topic/555844/South-America`.

A.J. Rothman, E. Levina, and J. Zhu. Sparse multivariate regression with covariance estimation. *J. of Computational and Graphical Statistics*, 19(4):947–962, 2010.

A. Sklar. *Fonctions de répartition à n dimensions et leurs marges*. Publ. Inst. Statis. Univ. Paris, 1959.

K. Subbian and A. Banerjee. Climate Multi-model Regression using Spatial Smoothing. In *SIAM International Conference on Data Mining*, pages 324–332, 2013.

Z.M. Subin, L.N. Murphy, F. Li, C. Bonfils, and W.J. Riley. Boreal lakes moderate seasonal and diurnal temperature variation and perturb atmospheric circulation: analyses in the Community Earth System Model 1 (CESM1). *Tellus A*, 64, 2012.

K.E. Taylor, R.J. Stouffer, and G.A. Meehl. An overview of CMIP5 and the experiment design. *Bulletin of the American Meteorological Society*, 93(4):485, 2012.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58:267–288, 1996.

R.R. Twilley. Confronting climate change in the Gulf Coast region: Prospects for sustaining our ecological heritage, 2001.

H. Wang, A. Banerjee, C.J. Hsieh, P. Ravikumar, and I. Dhillon. Large scale distributed sparse precision estimation. In *Advances in Neural Information Processing Systems*, pages 584–592, 2013.

X. Wang, Zhang. C., and Z. Zhang. Boosted multi-task learning for face verification with applications to web image and video search. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 142–149, 2009.

W.M. Washington et al. The use of the Climate-science Computational End Station (CCES) development and grand challenge team for the next IPCC assessment: an operational plan. *Journal of Physics*, 125(1), 2008.

M. Watanabe et al. Improved climate simulation by MIROC5: Mean states, variability, and climate sensitivity. *Journal of Climate*, 23(23):6312–6335, 2010.

C. Widmer and G. Rätsch. Multitask learning in computational biology. *International Conference on Machine Learning - Work. on Unsupervised and Transfer Learning*, 27: 207–216, 2012.

L. Xue and H. Zou. Regularized rank-based estimation of high-dimensional nonparanormal graphical models. *The Annals of Statistics*, pages 2541–2571, 2012.

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with Dirichlet process priors. *J. of Machine Learning Research*, 8:35–63, 2007.

M. Yang, Y. Li, and Z.M. Zhang. Multi-task learning with gaussian matrix generalized inverse gaussian model. In *International Conference on Machine Learning*, pages 423–431, 2013.

S. Yukimoto, Y. Adachi, and M. Hosaka. A new global climate model of the meteorological research institute: MRI-CGCM3: model description and basic performance. *Journal of the Meteorological Society of Japan*, 90:23–64, 2012.

L. Zhang, T. Wu, X. Xin, M. Dong, and Z. Wang. Projections of annual mean air temperature and precipitation over the globe and in China during the 21st century by the BCC Climate System Model BCC_CSM1. 0. *Acta Met. Sinica*, 26(3):362–375, 2012.

Y. Zhang and J.G. Schneider. Learning multiple tasks with sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems*, pages 2550–2558, 2010.

Y. Zhang and D.-Y. Yeung. A convex formulation for learning task relationships in multi-task learning. In *Conference on Uncertainty in Artificial Intelligence*, pages 733–742, 2010.

J. Zhou, J. Chen, and J. Ye. *MALSAR: Multi-tAsk Learning via StructurAl Regularization*. Arizona State University, 2011a. URL `www.malsar.org`.

J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. In *Advances in Neural Information Processing Systems*, pages 702–710, 2011b.

T. Zhou and D. Tao. Multi-task copula by sparse graph regression. In *ACM Conference on Knowledge Discovery and Data Mining*, pages 771–780, 2014.

Journal of Machine Learning Research 17 (2016) 1-7 Submitted 5/15; Published 4/16

MLlib: Machine Learning in Apache Spark

Xiangrui Meng† MENG@DATABRICKS.COM
Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

Joseph Bradley JOSEPH@DATABRICKS.COM
Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

Burak Yavuz BURAK@DATABRICKS.COM
Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

Evan Sparks SPARKS@CS.BERKELEY.EDU
UC Berkeley, 465 Soda Hall, Berkeley, CA 94720

Shivaram Venkataraman SHIVARAM@EECS.BERKELEY.EDU
UC Berkeley, 465 Soda Hall, Berkeley, CA 94720

Davies Liu DAVIES@DATABRICKS.COM
Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

Jeremy Freeman FREEMANJ11@JANELIA.HHMI.ORG
HHMI Janelia Research Campus, 19805 Helix Dr, Ashburn, VA 20147

DB Tsai DBT@NETFLIX.COM
Netflix, 970 University Ave, Los Gatos, CA 95032

Manish Amde MANISH@ORIGAMILOGIC.COM
Origami Logic, 1134 Crane Street, Menlo Park, CA 94025

Sean Owen SOWEN@CLOUDERA.COM
Cloudera UK, 33 Creechurch Lane, London EC3A 5EB United Kingdom

Doris Xin DORX0@ILLINOIS.EDU
UIUC, 201 N Goodwin Ave, Urbana, IL 61801

Reynold Xin RXIN@DATABRICKS.COM
Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

Michael J. Franklin FRANKLIN@CS.BERKELEY.EDU
UC Berkeley, 465 Soda Hall, Berkeley, CA 94720

Reza Zadeh REZAB@STANFORD.EDU
Stanford and Databricks, 475 Via Ortega, Stanford, CA 94305

Matei Zaharia MATEI@MIT.EDU
MIT and Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

Ameet Talwalkar† AMEET@CS.UCLA.EDU
UCLA and Databricks, 4732 Boelter Hall, Los Angeles, CA 90095

Editor: Mark Reid
† Corresponding authors.

Abstract

Apache Spark is a popular open-source platform for large-scale data processing that is well-suited for iterative machine learning tasks. In this paper we present MLlib, Spark's open-source distributed machine learning library. MLlib provides efficient functionality for a wide range of learning settings and includes several underlying statistical, optimization, and linear algebra primitives. Shipped with Spark, MLlib supports several languages and provides a high-level API that leverages Spark's rich ecosystem to simplify the development of end-to-end machine learning pipelines. MLlib has experienced a rapid growth due to its vibrant open-source community of over 140 contributors, and includes extensive documentation to support further growth and to let users quickly get up to speed.

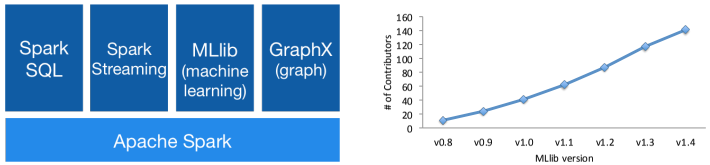
Keywords: scalable machine learning, distributed algorithms, apache spark

1. Introduction

Modern datasets are rapidly growing in size and complexity, and there is a pressing need to develop solutions to harness this wealth of data using statistical methods. Several 'next generation' data flow engines that generalize MapReduce (Dean and Ghemawat, 2004) have been developed for large-scale data processing, and building machine learning functionality on these engines is a problem of great interest. In particular, Apache Spark (Zaharia et al., 2012) has emerged as a widely used open-source engine. Spark is a fault-tolerant and general-purpose cluster computing system providing APIs in Java, Scala, Python, and R, along with an optimized engine that supports general execution graphs. Moreover, Spark is efficient at iterative computations and is thus well-suited for the development of large-scale machine learning applications.

In this work we present MLlib, Spark's distributed machine learning library, and the largest such library. The library targets large-scale learning settings that benefit from data-parallelism or model-parallelism to store and operate on data or models. MLlib consists of fast and scalable implementations of standard learning algorithms for common learning settings including classification, regression, collaborative filtering, clustering, and dimensionality reduction. It also provides a variety of underlying statistics, linear algebra, and optimization primitives. Written in Scala and using native (`C++` based) linear algebra libraries on each node, MLlib includes Java, Scala, and Python APIs, and is released as part of the Spark project under the Apache 2.0 license.

MLlib's tight integration with Spark results in several benefits. First, since Spark is designed with iterative computation in mind, it enables the development of efficient implementations of large-scale machine learning algorithms since they are typically iterative in nature. Improvements in low-level components of Spark often translate into performance gains in MLlib, without any direct changes to the library itself. Second, Spark's vibrant open-source community has led to rapid growth and adoption of MLlib, including contributions from over 140 people. Third, MLlib is one of several high-level libraries built on top of Spark, as shown in Figure 1(a). As part of Spark's rich ecosystem, and in part due to MLlib's `spark.ml` API for pipeline development, MLlib provides developers with a wide range of tools to simplify the development of machine learning pipelines in practice.



(a) (b)

Figure 1: (a) Apache Spark ecosytem. (b). Growth in MLlib contributors.

2. History and Growth

Spark was started in the UC Berkeley AMPLab and open-sourced in 2010. Spark is designed for efficient iterative computation and starting with early releases has been packaged with example machine learning algorithms. However, it lacked a suite of robust and scalable learning algorithms until the creation of MLlib. Development of MLlib began in 2012 as part of the MLbase project (Kraska et al., 2013), and MLlib was open-sourced in September 2013. From its inception, MLlib has been packaged with Spark, with the initial release of MLlib included in the Spark 0.8 release. As an Apache project, Spark (and consequently MLlib) is open-sourced under the Apache 2.0 license. Moreover, as of Spark version 1.0, Spark and MLlib are on a 3-month release cycle.

The original version of MLlib was developed at UC Berkeley by 11 contributors, and provided a limited set of standard machine learning methods. Since this original release, MLlib has experienced dramatic growth in terms of contributors. Less than two years later, as of the Spark 1.4 release, MLlib has over 140 contributors from over 50 organizations. Figure 1(b) demonstrates the growth in MLlib's open source community as a function of release version. The strength of this open-source community has spurred the development of a wide range of additional functionality.

3. Core Features

In this section we highlight the core features of MLlib; we refer the reader to the MLlib user guide for additional details (MLlib, 2015).

Supported Methods and Utilities. MLlib provides fast, distributed implementations of common learning algorithms, including (but not limited to): various linear models, naive Bayes, and ensembles of decision trees for classification and regression problems; alternating least squares with explicit and implicit feedback for collaborative filtering; and k-means clustering and principal component analysis for clustering and dimensionality reduction. The library also provides a number of low-level primitives and basic utilities for convex optimization, distributed linear algebra, statistical analysis, and feature extraction, and supports various I/O formats, including native support for LIBSVM format, data integration via Spark SQL (Armbrust et al., 2015), as well as PMML (Guazzelli et al., 2009) and MLlib's internal format for model export.

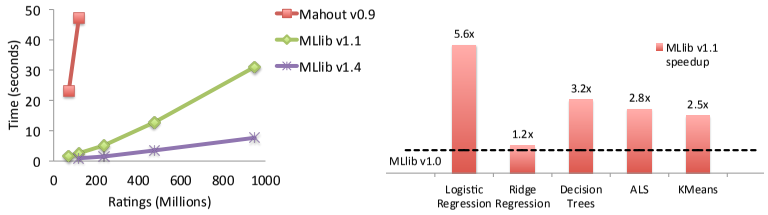
Algorithmic Optimizations. MLlib includes many optimizations to support efficient distributed learning and prediction. We highlight a few cases here. The ALS algorithm for

recommendation makes careful use of blocking to reduce JVM garbage collection overhead and to leverage higher-level linear algebra operations. Decision trees use many ideas from the PLANET project (Panda et al., 2009), such as data-dependent feature discretization to reduce communication costs, and tree ensembles parallelize learning both within trees and across trees. Generalized linear models are learned via optimization algorithms which parallelize gradient computation, using fast `C++`-based linear algebra libraries for worker computations. Many algorithms benefit from efficient communication primitives; in particular tree-structured aggregation prevents the driver from being a bottleneck, and Spark broadcast quickly distributes large models to workers.

Pipeline API. Practical machine learning pipelines often involve a sequence of data preprocessing, feature extraction, model fitting, and validation stages. Most machine learning libraries do not provide native support for the diverse set of functionality required for pipeline construction. Especially when dealing with large-scale datasets, the process of cobbling together an end-to-end pipeline is both labor-intensive and expensive in terms of network overhead. Leveraging Spark's rich ecosystem and inspired by previous work (Pedregosa et al., 2011; Buitinck et al., 2013; Sparks et al., 2013, 2015), MLlib includes a package aimed to address these concerns. This package, called `spark.ml`, simplifies the development and tuning of multi-stage learning pipelines by providing a uniform set of high-level APIs (Meng et al., 2015), including APIs that enable users to swap out a standard learning approach in place of their own specialized algorithms.

Spark Integration. MLlib benefits from the various components within the Spark ecosystem. At the lowest level, Spark core provides a general execution engine with over 80 operators for transforming data, e.g., for data cleaning and featurization. MLlib also leverages the other high-level libraries packaged with Spark. Spark SQL provides data integration functionality, SQL and structured data processing which can simplify data cleaning and preprocessing, and also supports the DataFrame abstraction which is fundamental to the `spark.ml` package. GraphX (Gonzalez et al., 2014) supports large-scale graph processing and provides a powerful API for implementing learning algorithms that can naturally be viewed as large, sparse graph problems, e.g., LDA (Blei et al., 2003; Bradley, 2015). Additionally, Spark Streaming (Zaharia et al., 2013) allows users to process live data streams and thus enables the development of online learning algorithms, as in Freeman (2015). Moreover, performance improvements in Spark core and these high-level libraries lead to corresponding improvements in MLlib.

Documentation, Community, and Dependencies. The MLlib user guide provides extensive documentation; it describes all supported methods and utilities and includes several code examples along with API docs for all supported languages (MLlib, 2015). The user guide also lists MLlib's code dependencies, which as of version 1.4 are the following open-source libraries: Breeze, netlib-java, and (in Python) NumPy (Breeze, 2015; Halliday, 2015; Braun, 2015; NumPy, 2015). Moreover, as part of the Spark ecosystem, MLlib has active community mailing lists, frequent meetup events, and JIRA issue tracking to facilitate open-source contributions (Community, 2015). To further encourage community contributions, Spark Packages (Packages, 2015) provides a community package index to track the growing number of open source packages and libraries that work with Spark. To date, several of the contributed packages consist of machine learning functionality that builds on MLlib.



(a) (b)

Figure 2: (a) Benchmarking results for ALS. (b) MLlib speedup between versions.

Finally, a massive open online course has been created to describe the core algorithmic concepts used to develop the distributed implementations within MLlib (Talwalkar, 2015).

4. Performance and Scalability

In this section we briefly demonstrate the speed, scalability, and continued improvements in MLlib over time. We first look at scalability by considering ALS, a commonly used collaborative filtering approach. For this benchmark, we worked with scaled copies of the Amazon Reviews dataset (McAuley and Leskovec, 2013), where we duplicated user information as necessary to increase the size of the data. We ran 5 iterations of MLlib's ALS for various scaled copies of the dataset, running on a 16 node EC2 cluster with m3.2xlarge instances using MLlib versions 1.1 and 1.4. For comparison purposes, we ran the same experiment using Apache Mahout version 0.9 (Mahout, 2014), which runs on Hadoop MapReduce. Benchmarking results, presented in Figure 2(a), demonstrate that MapReduce's scheduling overhead and lack of support for iterative computation substantially slow down its performance on moderately sized datasets. In contrast, MLlib exhibits excellent performance and scalability, and in fact can scale to much larger problems.

Next, we compare MLlib versions 1.0 and 1.1 to evaluate improvement over time. We measure the performance of common machine learning methods in MLlib, with all experiments performed on EC2 using m3.2xlarge instances with 16 worker nodes and synthetic datasets from the spark-perf package (`https://github.com/databricks/spark-perf`). The results are presented in Figure 2(b), and show a 3× speedup on average across all algorithms. These results are due to specific algorithmic improvements (as in the case of ALS and decision trees) as well as to general improvements to communication protocols in Spark and MLlib in version 1.1 (Yavuz and Meng, 2014).

5. Conclusion

MLlib is in active development, and the following link provides details on how to contribute: `https://cwiki.apache.org/confluence/display/SPARK/Contributing+to+Spark`. Moreover, we would like to acknowledge all MLlib contributors. The list of Spark contributors can be found at `https://github.com/apache/spark`, and the `git log` command can be used to identify MLlib contributors.

References

Michael Armbrust, Reynold Xin, Cheng Lian, Yin Yuai, Davies Liu, Joseph Bradley, Xiangrui Meng, Tomer Kaftan, Michael Franklin, Ali Ghodsi, and Matei Zaharia. Spark SQL: Relational data processing in spark. In *ACM Special Interest Group on Management of Data*, 2015.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3, 2003.

Joseph Bradley. Topic modeling with LDA: MLlib meets GraphX. `https://databricks.com/?p=3135`, 2015.

Mikio Braun. jblas. `http://jblas.org/`, 2015.

Breeze. Breeze. `https://github.com/scalanlp/breeze/wiki`, 2015.

Lars Buitinck et al. API design for machine learning software: experiences from the scikit-learn project. *arXiv:1309.0238*, 2013.

Spark Community. Spark community. `https://spark.apache.org/community.html`, 2015.

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *USENIX Symposium on Operating Systems Design and Implementation*, 2004.

Jeremy Freeman. Introducing streaming k-means in spark 1.2. `https://databricks.com/?p=2382`, 2015.

Joseph E. Gonzalez, Reynold S. Xin, Ankur Dave, Daniel Crankshaw, Michael J. Franklin, and Ion Stoica. Graphx: Graph processing in a distributed dataflow framework. In *Conference on Operating Systems Design and Implementation*, 2014.

Alex Guazzelli, Michael Zeller, Wen-Ching Lin, and Graham Williams. PMML: An open standard for sharing models. *The R Journal*, 1(1), 2009.

Sam Halliday. netlib-java. `https://github.com/fommil/netlib-java`, 2015.

Tim Kraska, Ameet Talwalkar, John Duchi, Rean Griffith, Michael Franklin, and Michael Jordan. MLbase: A Distributed Machine-learning System. In *Conference on Innovative Data Systems Research*, 2013.

Mahout. Apache mahout. `http://mahout.apache.org/`, 2014.

Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *ACM Recommender Systems Conference*, 2013.

Xiangrui Meng, Joseph Bradley, Evan Sparks, and Shivaram Venkataraman. Ml pipelines: A new high-level api for MLlib. `https://databricks.com/?p=2473`, 2015.

MLlib. MLlib user guide. `https://spark.apache.org/docs/latest/mllib-guide.html`, 2015.

NumPy. Numpy. http://www.numpy.org/, 2015.

Spark Packages. Spark packages. https://spark-packages.org, 2015.

Biswanath Panda, Joshua S. Herbach, Sugato Basu, and Roberto J. Bayardo. Planet: Massively parallel learning of tree ensembles with mapreduce. In *International Conference on Very Large Databases*, 2009.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011.

Evan R. Sparks, Ameet Talwalkar, Virginia Smith, Jey Kottalam, Xinghao Pan, Joseph E. Gonzalez, Michael J. Franklin, Michael I. Jordan, and Tim Kraska. MLI: An API for Distributed Machine Learning. In *International Conference on Data Mining*, 2013.

Evan R Sparks, Ameet Talwalkar, Daniel Haas, Michael J. Franklin, Michael I. Jordan, and Tim Kraska. Automating model search for large scale machine learning. *Symposium on Cloud Computing*, 2015.

Ameet Talwalkar. BerkeleyX CS190-1x: Scalable machine learning. https://www.edx.org/course/scalable-machine-learning-uc-berkeleyx-cs190-1x, 2015.

Burak Yavuz and Xiangrui Meng. Spark 1.1: MLlib performance improvements. https://databricks.com/?p=1393, 2014.

Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. In *USENIX Symposium on Networked Systems Design and Implementation*, 2012.

Matei Zaharia, Tathagata Das, Haoyuan Li, Timothy Hunter, Scott Shenker, and Ion Stoica. Discretized streams: Fault-tolerant streaming computation at scale. In *Symposium on Operating Systems Principles*, 2013.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 6/15; Published 4/16

OLPS: A Toolbox for On-Line Portfolio Selection

Bin Li BINLI.WHU@WHU.EDU.CN
Economics and Management School, Wuhan University, Wuhan, P.R. China 430072
Doyen Sahoo DOYEN.2014@PHDIS.SMU.EDU.SG
Steven C. H. Hoi CHHOI@SMU.EDU.SG
School of Information Systems, Singapore Management University, Singapore 178902

Editor: Geoff Holmes

Abstract

On-line portfolio selection is a practical financial engineering problem, which aims to sequentially allocate capital among a set of assets in order to maximize long-term return. In recent years, a variety of machine learning algorithms have been proposed to address this challenging problem, but no comprehensive open-source toolbox has been released for various reasons. This article presents the first open-source toolbox for "On-Line Portfolio Selection" (OLPS), which implements a collection of classical and state-of-the-art strategies powered by machine learning algorithms. We hope that OLPS can facilitate the development of new learning methods and enable the performance benchmarking and comparisons of different strategies. OLPS is an open-source project released under Apache License (version 2.0), which is available at `https://github.com/OLPS/` or `http://OLPS.stevenhoi.org/`.

Keywords: On-line portfolio selection, online learning, trading system, simulation.

1. Introduction

In recent years, machine learning has been applied to various applications in finance (Györfi et al., 2012), including On-line Portfolio Selection, which aims to sequentially allocate capital among a set of assets, such that the investment return can be maximized in the long run (Kelly, 1956). It has attracted increasing attention from both academia and industry, and several machine learning algorithms have been proposed (Li and Hoi, 2014), including traditional algorithms (Cover, 1991; Helmbold et al., 1998; Agarwal et al., 2006; Borodin et al., 2004; Györfi et al., 2006, 2008), and recent state-of-the-art online learning algorithms (Li et al., 2011, 2012, 2013, 2015). Unlike other application domains in machine learning where various open-source packages are available, very few open-source toolkits[1] exist for on-line portfolio selection, primarily due to the confidential nature of financial industry. Consequently, it is difficult for researchers to evaluate new algorithms for comprehensive comparisons with existing ones.

This article introduces an open-source toolkit named "On-Line Portfolio Selection" (OLPS), which consists of a family of classical and state-of-the-art on-line portfolio selection algorithms. To the best of our knowledge, OLPS is the first comprehensive open-source package for this problem, which includes various strategies, and a set of preprocessing, postprocessing and visualization tools in an integrated platform. The open-source nature of OLPS makes it easy for a third party to develop new algorithms, and facilitates the comparisons with many built-in algorithms on real datasets.

1. Some repositories contain public datasets, such as `http://www.cs.technion.ac.il/~rani/portfolios` and `http://www.szit.bme.hu/~oti/portfolio`, but no public code was released.

2. Overview and Implementation

OLPS implements a framework for backtesting various algorithms for on-line portfolio selection in both Matlab and Octave (the GUI is only available in Matlab) under Windows, Linux, and Mac OS. Figure 1 gives an overview of the OLPS toolkit with three main modules: (i) data preprocessing: it loads a dataset and initializes the backtesting environments, e.g., preparing log file handles, etc; (ii) algorithms: it calls selected algorithms, and simulates the trading process based on the chosen data from the first module; and (iii) post-processing: it analyzes statistical significance of the results from the second module, e.g., some risk-adjusted returns.

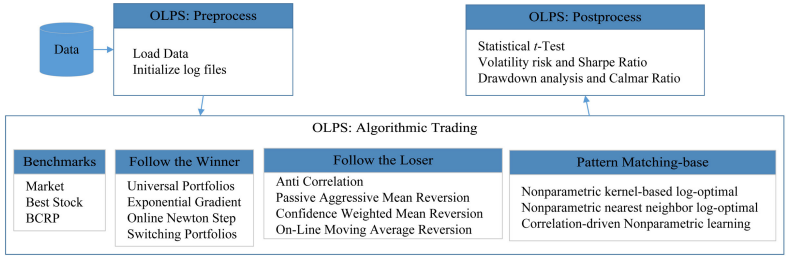


Figure 1: Structure of the OLPS toolbox.

2.1 Pre-processing

Datasets	Markets	Regions	Time Frames	# Periods	# Assets	File Names (.mat)
NYSE (O)	Stock	US	07/03/1962 - 12/31/1962	5651	36	`nyse-o`
NYSE (N)	Stock	US	01/01/1985 - 06/30/2010	6431	23	`nyse-n`
TSE	Stock	CA	01/04/1994 - 12/31/1998	1259	88	`tse`
SP500	Stock	US	01/02/1998 - 01/31/2003	1276	25	`sp500`
MSCI	Index	Global	04/01/2006 - 03/31/2010	1043	24	`msci`
DJIA	Stock	US	01/14/2001 - 01/14/2003	507	30	`djia`

Table 1: Summary of the six datasets from real markets.

The main functionality of this step is to initialize the trading environment. OLPS supports all types of datasets that Matlab/Octave accepts, and all existing datasets are in MAT-file. A typical dataset contains an $n \times m$ matrix of price relatives, where n denotes the number of trading periods, and m refers to the number of assets. It can be further adapted to incorporate real market data feeds, such that the toolkit can handle real time data and conduct paper trading or real trading[2]. Table 1

2. For example, Interactive Brokers (`http://www.interactivebrokers.com`) provides free APIs. Paper and real trading both require to implement order submission, while back test does not.

summarizes several representative public datasets included in the toolbox[3]. Users are free to collect up-to-date data from various sources, such as the CRSP database, Quandl.com, etc.

2.2 Algorithmic Trading

This step conducts simulation based on historical market data. Table 2 summarizes the families of algorithms implemented in the OLPS toolbox. In our framework, implementing a new strategy generally requires four files, i.e., an entry file, a run file, a kernel file and an expert file. The entry file extracts parameters and call the corresponding run file. The run file simulates a whole trading process, and calls its kernel file to construct a portfolio for each period, which is used for rebalancing. The kernel file outputs a portfolio, while it facilitates the development of meta algorithms, which combine multiple experts' portfolios outputted by expert files that output one portfolio. In case of only one expert, the kernel file is not necessary and directly enters the expert file. Developing new strategies involves writing a kernel file, which aims to output portfolio.

Categories	Algorithms	Entry Files	References
Baselines	Uniform BAH	`ubah.m`	Li and Hoi (2014)
	Best Stock	`best.m`	Li and Hoi (2014)
	Uniform CRP	`ucrp.m`	Li and Hoi (2014)
	BCRP	`bcrp.m`	Cover (1991)
Follow the Winner	UP	`up.m`	Cover (1991)
	EG	`eg.m`	Helmbold et al. (1998)
	ONS	`ons.m`	Agarwal et al. (2006)
	SP	`sp.m`	Singer (1997)
Follow the Loser	Anticor	`anticor/anticor_anticor.m`	Borodin et al. (2004)
	PAMR	`pamr/pamr_1/pamr_2.m`	Li et al. (2012)
	CWMR	`cwmr_stdev/cwmr_var.m`	Li et al. (2013)
	OLMAR	`olmar1/olmar2.m`	Li et al. (2015)
Pattern Matching based Algorithms	B^K	`bk.m`	Györfi et al. (2006)
	B^{NN}	`bnn.m`	Györfi et al. (2008)
	CORN	`corn/cornu/cornk.m`	Li et al. (2011)
Others	M0	`m0.m`	Borodin et al. (2000)
	T0	`t0.m`	Borodin et al. (2000)

Table 2: Summary of the implemented algorithms.

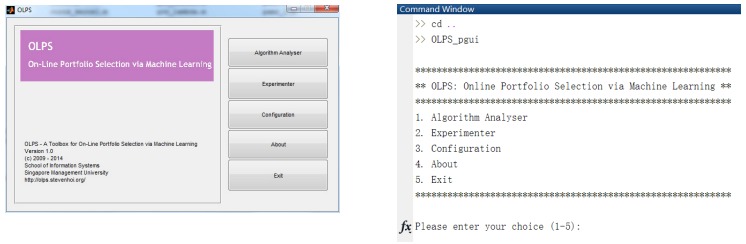
2.3 Post-processing

After the algorithmic trading simulation, the last step is to post-process the portfolio's return series for performance analysis by summarizing the following performance metrics:

- Cumulative Return: the most widely used performance metric in related studies;
- Volatility and Sharpe Ratio: measures volatility risk and related risk-adjusted return;
- Drawdown and Calmar Ratio: measures downside risk and risk-adjusted return;
- *T*-test statistics (Grinold and Kahn, 1999): tests if a strategy's return is significantly different from the market.

3. More datasets and their details, including the components, can be found at the project website.

3. Usage



(a) GUI mode (b) Pseudo GUI mode

Figure 2: GUI mode and Pseudo GUI mode.

We provide three interfaces to run simulations in the toolbox. As shown in Figure 2, the toolbox has two types of Graphical User Interfaces (GUIs), including a GUI implemented in Matlab and a Pseudo GUI in command line, which is thus available in both Matlab and Octave. The other interface is a Command Line Interface (CLI), and is also available in both Matlab and Octave.

While a GUI is straightforward to use, in the CLI mode, we generalize a *manager* function so as to control the running of all algorithms.Suppose we are going to simulate the PAMR algorithm on the NYSE (O) dataset. The commands can be listed as follows:

```
1: >> opts.quiet_mode=1; opts.display_interval=500; opts.progress=0;
2: >> opts.analyze_mode=1; opts.log_mode=1; opts.mat_mode=1;
3: >> manager('pamr', 'nyse-o', {0.5, 0}, opts);
```

Line 1 and 2 set some variables that are used to control the display and file storage. Line 3 calls the manager function to execute the "pamr" strategy on the "nyse-o" dataset with the parameters equaling "{0.5, 0}". During the simulation, the toolbox will output information periodically. After the simulation, the algorithm will analyze the returns and output the cumulative return, risk adjusted returns and statistical test statistics. Corresponding running details and results will be automatically stored in the Log folder.

4. Summary

This article presented OLPS — an open-source On-Line Portfolio Selection toolbox to facilitate the related research in machine learning and computational finance. This is the first open-source project in the area, which not only facilitates researchers to develop new strategies, but also allows them to easily benchmark their performance with existing strategies. Besides, the toolbox supports a large collection of classical and state-of-the-art on-line portfolio selection strategies. The toolbox offers a user-friendly GUI in Matlab, a Pseudo GUI and a CLI mode in both Octave and Matlab. We hope that the open-source nature of the software would encourage researchers to extend the toolkit and share their algorithms through the OLPS platform.

Acknowledgments

This work was partially supported by the Project of NSFC (No. 71401128 and 71471142), the Project of SRF for ROCS, SEM, and Singapore MOE tier 1 project (C220/MSS14C003).

References

A. Agarwal, E. Hazan, S. Kale, and R. E. Schapire. Algorithms for portfolio management based on the newton method. In *Proceedings of ICML*, 2006.

A. Borodin, R. El-Yaniv, and V. Gogan. On the competitive theory and practice of portfolio selection (extended abstract). In *Proceedings of the Latin American Symposium on Theoretical Informatics*, 2000.

A. Borodin, R. El-Yaniv, and V. Gogan. Can we learn to beat the best stock. *JAIR*, 21:579–594, 2004.

T. M. Cover. Universal portfolios. *Mathematical Finance*, 1(1):1–29, 1991.

R. Grinold and R. Kahn. *Active Portfolio Management: A Quantitative Approach for Producing Superior Returns and Controlling Risk*. New York : McGraw-Hill, New York, 1999.

L. Györfi, G. Lugosi, and F. Udina. Nonparametric kernel-based sequential investment strategies. *Mathematical Finance*, 16(2):337–357, 2006.

L. Györfi, F. Udina, and H. Walk. Nonparametric nearest neighbor based empirical portfolio selection strategies. *Statistics and Decisions*, 26(2):145–157, 2008.

L. Györfi, G. Ottucsák, and H. Walk. *Machine Learning for Financial Engineering*. Singapore : World Scientific, 2012.

D. P. Helmbold, R. E. Schapire, Y. Singer, and M. K. Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998.

Jr. Kelly, J. A new interpretation of information rate. *Bell Systems Technical Journal*, 35:917–926, 1956.

B. Li and S. C. H. Hoi. Online portfolio selection: A survey. *ACM CSUR*, 46(3):35, 2014.

B. Li, S. C.H. Hoi, and V. Gopalkrishnan. Corn: Correlation-driven nonparametric learning approach for portfolio selection. *ACM TIST*, 2(3):21:1–21:29, 2011.

B. Li, P. Zhao, S. Hoi, and V. Gopalkrishnan. Pamr: Passive aggressive mean reversion strategy for portfolio selection. *Machine Learning*, 87(2):221–258, 2012.

B. Li, S. C.H. Hoi, P. Zhao, and V. Gopalkrishnan. Confidence weighted mean reversion strategy for online portfolio selection. *ACM TKDD*, 7(1):4:1 – 4:38, 2013.

B. Li, S. C.H. Hoi, D. Sahoo, and Z. Liu. Moving average reversion strategy for on-line portfolio selection. *Artificial Intelligence*, 222:104 – 123, 2015.

Y. Singer. Switching portfolios. *International Journal of Neural Systems*, 8(4):488–495, 1997.

Journal of Machine Learning Research 17 (2016) 1-39 Submitted 6/15; Revised 12/15; Published 4/16

A Bounded p-norm Approximation of Max-Convolution for Sub-Quadratic Bayesian Inference on Additive Factors

Julianus Pfeuffer
Eberhard Karls Universität Tübingen

Oliver Serang ORSERANG@UW.EDU
Freie Universität Berlin
Department of Informatik
Takustr. 9, 14195 Berlin, Germany
and
The Leibniz-Institute of Freshwater Ecology and Inland Fisheries (IGB)
Müggelsee 310, 12587 Berlin, Germany

Editor: Amos Storkey

Abstract

Max-convolution is an important problem closely resembling standard convolution; as such, max-convolution occurs frequently across many fields. Here we extend the method with fastest known worst-case runtime, which can be applied to nonnegative vectors by numerically approximating the Chebyshev norm $\|\cdot\|_\infty$, and use this approach to derive two numerically stable methods based on the idea of computing p-norms via fast convolution: The first method proposed, with runtime in $O(k\log(k)\log(\log(k)))$ (which is less than $18k\log(k)$ for any vectors that can be practically realized), uses the p-norm as a direct approximation of the Chebyshev norm. The second approach proposed, with runtime in $O(k\log(k))$ (although in practice both perform similarly), uses a novel null space projection method, which extracts information from a sequence of p-norms to estimate the maximum value in the vector (this is equivalent to querying a small number of moments from a distribution of bounded support in order to estimate the maximum). The p-norm approaches are compared to one another and are shown to compute an approximation of the Viterbi path in a hidden Markov model where the transition matrix is a Toeplitz matrix; the runtime of approximating the Viterbi path is thus reduced from $O(nk^2)$ steps to $O(nk\log(k))$ steps in practice, and is demonstrated by inferring the U.S. unemployment rate from the S&P 500 stock index.

Keywords: Bayesian inference, *maximum a posteriori*, fast Fourier transform, max-convolution, p-norm, L_p space, hidden Markov model, null space projection, polynomial matrix

1. Introduction

Max-convolution occurs frequently in signal processing and Bayesian inference: it is used in image analysis (Ritter and Wilson, 2000), in network calculus (Boyer et al., 2013), in economic equilibrium analysis (Sun and Yang, 2002), and in a probabilistic variant of combinatoric generating functions, wherein information on a sum of values into their most probable constituent parts (*e.g.*, identifying proteins from mass spectrometry (Serang et al., 2010; Serang, 2014)). Max-convolution operates on the semi-ring $(max, \times)$, meaning that it behaves identically to a standard convolution, except it employs a max operation in lieu of the + operation in standard convolution (max-convolution is also equivalent to min-convolution, also called infimal convolution, which operates on the tropical semi-ring $(min, +)$). Due to the importance and ubiquity of max-convolution, substantial effort has been invested into highly optimized implementations (*e.g.*, implementations of the quadratic method on GPUs; Zach et al., 2008).

Max-convolution can be defined using vectors (or discrete random variables, whose probability mass functions are analogous to nonnegative vectors) with the relationship $M = L + R$. Given the target sum $M = m$, the max-convolution finds the largest value $L[\ell] \times R[r]$ for which $m = \ell + r$.

$$\begin{aligned} M[m] &= \max_{\ell, r:\, m=\ell+r} L[\ell]R[r] \\ &= \max_{\ell} L[\ell]R[m-\ell] \\ &= (L \ *_{\max} \ R)[m] \end{aligned}$$

where $*_{\max}$ denotes the max-convolution operator. In probabilistic terms, this is equivalent to finding the highest probability of the joint events $\Pr(L = \ell, R = r)$ that would produce each possible value of the sum $M = L + R$ (note that in the probabilistic version, the vector M would subsequently need to be normalized so that its sum is 1).

Although applications of max-convolution are numerous, only a small number of methods exist for solving it (Serang, 2015). These methods fall into two main categories, each with their own drawbacks: The first category consists of very accurate methods that are have worst-case runtimes either quadratic (Bussieck et al., 1994) or slightly more efficient than quadratic in the worst-case (Bremner et al., 2006). Conversely, the second type of method computes a numerical approximation to the desired result, but in $O(k\log_2(k))$ steps; however, no bound for the numerical accuracy of this method has been derived (Serang, 2015).

While the two approaches from the first category of methods for solving max-convolution do so by either using complicated sorting routines or by creating a bijection to an optimization problem, the numerical approach solves max-convolution by showing an equivalence between $*_{\max}$ and the process of first generating a vector $u^{(m)}$ for each index m of the result (where $u^{(m)}[\ell] = L[\ell]R[m-\ell]$ for all in-bounds indices) and subsequently computing the maximum $M[m] = \max_\ell u^{(m)}[\ell]$. When L and R are nonnegative, the maximization over the vector $u^{(m)}$ can be computed exactly via the Chebyshev norm

$$\begin{aligned} M[m] &= \max_{\ell} u^{(m)}[\ell] \\ &= \lim_{p\to\infty} \|u^{(m)}\|_p \end{aligned}$$

but requires $O(k^2)$ steps (where k is the length of vectors L and R). However, once a fixed p^*-norm is chosen, the approximation corresponding to that p^* can be computed by expanding the p^*-norm to yield

$$\begin{aligned} \lim_{p\to\infty} \|u^{(m)}\|_p &= \lim_{p\to\infty} \left(\sum_\ell \left(u^{(m)}[\ell] \right)^p \right)^{\frac{1}{p}} \\ &\approx \left(\sum_\ell \left(u^{(m)}[\ell] \right)^{p^*} \right)^{\frac{1}{p^*}} \\ &= \left(\sum_\ell L[\ell]^{p^*} \ R[m-\ell]^{p^*} \right)^{\frac{1}{p^*}} \\ &= \left(\sum_\ell \left(L^{p^*} \right)[\ell] \ \left(R^{p^*} \right)[m-\ell] \right)^{\frac{1}{p^*}} \\ &= \left(L^{p^*} \ * \ R^{p^*} \right)^{\frac{1}{p^*}}[m] \end{aligned}$$

where $L^{p^*} = \langle (L[0])^{p^*}, (L[1])^{p^*}, \ldots, (L[k-1])^{p^*} \rangle$ and $*$ denotes standard convolution. The standard convolution can be done via fast Fourier transform (FFT) in $O(k \log_2(k))$ steps, which is substantially more efficient than the $O(k^2)$ required by the naive method (**Algorithm 1**).

To date, the numerical method has demonstrated the best speed-accuracy trade-off on Bayesian inference tasks, and can be generalized to multiple dimensions (*i.e.*, tensors). In particular, they have been used with probabilistic convolution trees (Serang, 2014) to efficiently compute the most probable values of discrete random variables $X_0, X_1, \ldots X_{n-1}$ for which the sum is known $X_0 + X_1 + \ldots X_{n-1} = y$ (Serang, 2014). The one-dimensional variant of this problem (*i.e.*, where each X_i is a one-dimensional vector) solves the probabilistic generalization of the subset sum problem, while the two-dimensional variant (*i.e.*, where each X_i is a one-dimensional matrix) solves the generalization of the knapsack problem (note that these problems are not NP-hard in this specific case, because we assume an evenly-spaced discretization of the possible values of the random variables).

However, despite the practical performance that has been demonstrated by the numerical method, only cursory analysis has been performed to formalize the influence of the value of p^* on the accuracy of the result and to bound the error of the p^*-norm approximation. Optimizing the choice of p^* is non-trivial: Larger values of p^* more closely resemble a true maximization under the p^*-norm, but result in underflow (note that in **Algorithm 1**, the maximum values of both L and R can be divided out and then multiplied back in after max-convolution so that overflow is not an issue). Conversely, smaller values of p^* suffer less underflow, but compute a norm with less resemblance to maximization. Here we perform an in-depth analysis of the influence of p^* on the accuracy of numerical max-convolution, and from that analysis we construct a modified piecewise algorithm, on which we demonstrate bounds on the worst-case absolute error. This modified algorithm, which runs in $O(k \log(k) \log(\log(k)))$ steps, is demonstrated using a hidden Markov model describing the relationship between U.S. unemployment and the S&P 500 stock index.

We then extend the modified algorithm and introduce a second modified algorithm, which not only uses a single p-norm as a means of approximating the Chebyshev norm, but instead uses a sequence of p-norms and assembles them using a projection as a means to approximate the Chebyshev norm. Using numerical simulations as evidence, we make a conjecture regarding the relative error of the null space projection method. In practice, this null space projection algorithm is shown to have similar runtime and higher accuracy when compared with the piecewise algorithm.

2. Methods

We begin by outlining and comparing three numerical methods for max-convolution. By analyzing the benefits and deficits of each of these methods, we create improved variants. All of these methods will make use of the basic numerical max-convolution idea summarized in the introduction, and as such we first declare a method for computing the numerical max-convolution estimate for a given p^* as `numericalMaxConvolveGivenPStar` (**Algorithm 1**).

Algorithm 1 Numerical max-convolution given a fixed p^*, a numerical method to estimate the max-convolution of two PMFs or nonnegative vectors. The parameters are two nonnegative vectors L' and R' (both scaled so that they have maximal element 1) and the numerical value p^* used for computation. The return value is a numerical estimate of the max-convolution $L' *_{\max} R'$.

procedure numericalMaxConvolveGivenPStar(L', R', p^*)
  $\forall \ell,\ vL[\ell] \leftarrow L[\ell]^{p^*}$
  $\forall r,\ vR[r] \leftarrow R[r]^{p^*}$
  $vM \leftarrow vL * vR$ ▷ Standard FFT convolution is used here
  $\forall m,\ M'[m] \leftarrow vM[m]^{\frac{1}{p^*}}$
  return M'
end procedure

2.1 Fixed Low-Value $p^* = 8$ Method:

The effects of underflow will be minimal (as it is not very far from standard FFT convolution, an operation with high numerical stability), but it can still be imprecise due to numerical "bleed-in" (*i.e.*, error due to contributions from non-maximal terms for a given $u^{(m)}$ because the p^*-norm is not identical to the Chebyshev norm). Overall, this will perform well on indices where the exact value of the result is small, but perform poorly when the exact value of the result is large.

2.2 Fixed High-Value $p^* = 64$ Method:

As noted above, will offer the converse pros and cons compared to using a low p^*: numerical artifacts due to bleed-in will be smaller (thus achieving greater performance on indices where the exact values of the result are larger), but underflow may be significant (and therefore, indices where the exact results of the max-convolution are small will be inaccurate).

2.3 Higher-Order Piecewise Method:

The higher-order piecewise method formalizes the empirical cutoff values found in Serang 2015; previously, numerical stability boundaries were found for each p^* by computing both the exact max-convolution (via the naive $O(k^2)$ method) and via the numerical method using the ascribed value of p^*, and finding the value below which the numerical values experienced a high increase in relative absolute error.

Those previously observed empirical numerical stability boundaries can be formalized by using the fact that the employed `numpy` implementation of FFT convolution has high accuracy on indices where the result has a value $\geq \tau$ relative to the maximum value; therefore, if the arguments L and R are both normalized so that each has a maximum value of 1, the fast max-convolution approximation is numerically stable for any index m where the result of the FFT convolution, *i.e.*, $vM[m]$, is $\geq \tau$. The `numpy` documentation defines a conservative numeric tolerance for underflow $\tau = 10^{-12}$, which is a conservative estimate of the numerical stability boundary demonstrated in **Figure 1** (those boundary points occur very close to the true machine precision $\epsilon \approx 10^{-15}$).

Because Cooley-Tukey implementations of FFT-based convolution (*e.g.*, the `numpy` implementation) are widely applied to large problems with extremely small error, we will make a simplification and assume that, when constraining the FFT result to reach a value higher than machine epsilon (+ tolerance threshold), the error from the FFT is negligible in comparison to the error introduced by the p^*-norm approximation. This is firstly because the only source of numerical error during FFT (assuming an FFT implementation with numerically precise twiddle factors) on vectors in $[0, 1]^k$ will be the result of underflow from repeated addition and subtraction (neglecting the non-influencing multiplication with twiddle factors, which each have magnitude 1). The numerically imprecise routines are thus limited to $(x + y) - x$; when $x >> y$ (*i.e.*, $\frac{y}{x} < \epsilon \approx 10^{-15}$, the machine precision), then $(x + y) - x$ will return 0 instead of y. To recover at least one bit of the significand, the intermediate results of the FFT must surpass machine precision ϵ (since the worst case addition initially happens with the maximum $x = 1.0$).

The maximum sum of any values from a list of k such elements can never exceed k; for this reason, a conservative estimate of the numerical tolerance (with regard to underflow) of a DFT (discrete Fourier transform) will be the smallest value of y for which $\frac{y}{k} > \epsilon$; thus, $y > \epsilon k$. This yields a conservative estimate of the minimum value in one index at the result of an DFT convolution: when the result at some index m is $> \epsilon k$, then the result should be numerically stable. We emphasize on the very conservative nature of this estimate for a minimum stable value that is derived under the assumption of a simple, naive implementation of a DFT. In general however, FFT (a special case of DFT algorithms) implementations (especially well-proven implementations of the Cooley-Tukey algorithm) tend to follow error bounds logarithmic in k (Schatzman, 1996), due to their divide-and-conquer strategies, which perform fewer arithmetic operations. In practice, even for longer vectors

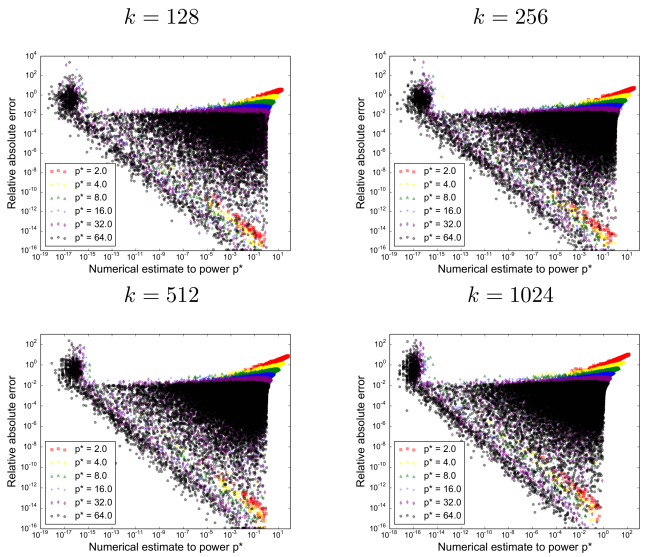


Figure 1: **Empirical estimate of τ to construct a piecewise method.** For each $k \in \{128, 256, 512, 1024\}$, 32 replicate max-convolutions (on vectors filled with uniform values) are performed. Error from two sources can be seen: error due to underflow is depicted in the sharp left mode, whereas error due to imperfect approximation, where $\|\cdot\|_{p^*} > \|\cdot\|_\infty$ can be seen in the gradual mode on the right. Error due to p^*-norm approximation is significantly smaller when p^* is larger (thereby flattening the right mode), but larger p^* values are more susceptible to underflow, pushing more indices into the left mode. Regardless of the value of k, error due to underflow occurs when $(\|\cdot\|_{p^*})^{p^*}$ goes below $\approx 10^{-15}$; this is approximately the numerical tolerance for τ described by the `numpy` documentation. Therefore, at each index m we can construct a piecewise method that uses the largest value of p^* for which the FFT convolution result is not close to the machine precision (*i.e.*, $(\|u^{(m)}\|_{p^*})^{p^*} \geq \tau$ for some $\tau > 10^{-15}$).

(*e.g.*, $k = 1024$ as shown in **Figure 1**), it is empirically demonstrated that the influence of the length on the τ threshold is negligible. By using a numerical tolerance $\tau = 10^{-12}$, we ensure that the vast majority of numerical error for the numerical max-convolution algorithm is due to the p^*-norm approximation (*i.e.*, employing $\|u^{(m)}\|_{p^*}$ instead of $\|u^{(m)}\|_\infty$) and not due to the long-used and numerically performant FFT result. Furthermore, in practice the mean squared error due to FFT will be much smaller than the conservative worst-case outlined here, because it is difficult for the largest intermediate summed value (in this case x) to be consistently large when many such very small values (in this case y) are encountered in the same list. Although τ could be chosen specifically for a problem of size k, note that this simple derivation is very conservative and thus it would be better to use a tighter bound for choosing τ. Regardless, for an FFT implementation that isn't as performant (*e.g.*, because it uses `float` types instead of `double`), increasing τ slightly would suffice.

Therefore, from this point forward we consider that the dominant cause of error to come from the max-convolution approximation. Using larger p^* values will provide a closer approximation; however, using a larger value of p^* may also drive values to zero (because the inputs L and R will be normalized within **Algorithm 1** so that the maximum of each is 1 when convolved via FFT), limiting the applicability of large p^* to indices m for which $vM[m] \geq \tau$.

Through this lens, the choice of p^* can be characterized by two opposing sources of error: higher p^* values better approximate $\|u^{(m)}\|_{p^*}$ but will be numerically unstable for many indices; lower p^* values provide worse approximations of $\|u^{(m)}\|_{p^*}$ but will be numerically unstable for only few indices. These opposing sources of error pose a natural method for improving the accuracy of this max-convolution approximation. By considering a small collection of p^* values, we can compute the full numerical estimate (at all indices) with each p^* using **Algorithm 1**; computing the full result at a given p^* is $\in O(k \log_2(k))$, so doing so on some small number c of p^* values considered, then the overall runtime will be $\in O(ck \log_2(k))$. Then, a final estimate is computed at each index by using the largest p^* that is stable (with respect to underflow) at that index. Choosing the largest p^* (of those that are stable with respect to underflow) corresponds to minimizing the bleed-in error, because the larger p^* becomes, the more muted the non-maximal terms in the norm become (and thus the closer the p^*-norm becomes to the true maximum).

Here we introduce this piecewise method and compare it to the simpler low-value $p^* = 8$ and high-value $p^* = 64$ methods and analyze the worst-case error of the piecewise method.

3. Results

This section derives theoretical error bounds as well as a practical comparison on an example for the standard piecewise method. Furthermore the development of an improvement with affine scaling is shown. Eventually, an evaluation of the latter is performed on a larger problem. Therefore we applied our technique to compute the Viterbi path for a hidden Markov model (HMM) to assess runtime and the level of error propagation.

3.1 Error and Runtime Analysis of the Piecewise Method

We first analyze the error for a particular underflow-stable p^* and then use that to generalize to the piecewise method, which seeks to use the highest underflow-stable p^*.

3.1.1 Error Analysis for a Fixed Underflow-Stable p^*:

We first scale L and R into L' and R' respectively, where the maximum elements of both L' and R' are 1; the absolute error can be found by unscaling the absolute error of the scaled problem:

$$|exact(L, R)[m] - numeric(L', R')[m]| = \max_\ell L[\ell] \max_r R[r] \, |exact(L', R')[m] - numeric(L', R')[m]|.$$

Algorithm 2 Piecewise numerical max-convolution , a numerical method to estimate the max-convolution of nonnegative vectors (revised to reduce bleed-in error). This procedure uses a p^* close to the largest possible stable value at each result index. The return value is a numerical estimate of the max-convolution $L *_{\max} R$. The runtime is in $O(k \log_2(k) \log_2(p^*_{\max}))$.

procedure numericalMaxConvolvePiecewise(L, R, $p^*_{\max}$)
  $\ell_{\max} \leftarrow \text{argmax}_\ell\, L[\ell]$
  $r_{\max} \leftarrow \text{argmax}_r\, R[r]$
  $L' \leftarrow \frac{L}{L[\ell_{\max}]}$
  $R' \leftarrow \frac{R}{R[r_{\max}]}$ ▷ Scale to a proportional problem on L', R'
  $allPStar \leftarrow [2^0, 2^1, \ldots, 2^{\lceil \log_2(p^*_{\max}) \rceil}]$
  for $i \in \{0, 1, \ldots len(allPStar)\}$ **do**
    $resForAllPStar[i] \leftarrow$ `fftNonnegMaxConvolveGivenPStar`(L', R', $allPStar[i]$)
  end for
  for $m \in \{0, 1, \ldots len(L) + len(R) - 1\}$ **do**
    $maxStablePStarIndex[m] \leftarrow \max\{i : (resForAllPStar[i][m])^{\text{allPStar}[i]} \geq \tau)\}$
  end for
  for $m \in \{0, 1, \ldots len(L) + len(R) - 1\}$ **do**
    $i \leftarrow maxStablePStarIndex[m]$
    $result[m] \leftarrow resForAllPStar[i][m]$
  end for
  return $L[\ell_{\max}] \times R[r_{\max}] \times result$ ▷ Undo previous scaling
end procedure

We first derive an error bound for the scaled problem on L', R' (any mention of a vector $u^{(m)}$ refers to the scaled problem), and then reverse the scaling to demonstrate the error bound on the original problem on L, R.

For any particular "underflow-stable" p^* (*i.e.*, any value of p^* for which $\left(\|u^{(m)}\|_{p^*}\right)^{p^*} \geq \tau$), the absolute error for the numerical method for fast max-convolution can be bound fairly easily by factoring out the maximum element of $u^{(m)}$ (this maximum element is equivalent to the Chebyshev norm) from the p^*-norm:

$$\begin{aligned}
& |exact(L', R')[m] - numeric(L', R')[m]| \\
= \;& |\|u^{(m)}\|_{p^*} - \|u^{(m)}\|_\infty| \\
= \;& \|u^{(m)}\|_{p^*} - \|u^{(m)}\|_\infty \\
= \;& \|u^{(m)}\|_\infty \left(\frac{\|u^{(m)}\|_{p^*}}{\|u^{(m)}\|_\infty} - 1 \right) \\
= \;& \|u^{(m)}\|_\infty \left(\| \frac{u^{(m)}}{\|u^{(m)}\|_\infty} \|_{p^*} - 1 \right) \\
= \;& \|u^{(m)}\|_\infty \left(\|v^{(m)}\|_{p^*} - 1 \right)
\end{aligned}$$

where $v^{(m)}$ is a nonnegative vector of the same length as $u^{(m)}$ (this length is denoted k_m) where $v^{(m)}$ contains one element equal to 1 (because the maximum element of $u^{(m)}$ must, by definition, be contained within $u^{(m)}$) and where no element of $v^{(m)}$ is greater than 1 (also provided by the definition of the maximum). These observations result in the equation

$$\begin{aligned}
\|v^{(m)}\|_{p^*} &\leq \|(1, 1, \ldots 1)\|_{p^*} \\
&= \left(\sum_i^{k_m} 1^{p^*} \right)^{\frac{1}{p^*}} \\
&= k_m{}^{\frac{1}{p^*}}.
\end{aligned}$$

Thus, since $\|v^{(m)}\|_{p^*} \geq 1$, the error is bounded:

$$\begin{aligned}
& |exact(L', R')[m] - numeric(L', R')[m]| \\
= \;& \|u^{(m)}\|_\infty \left(\|v^{(m)}\|_{p^*} - 1 \right) \\
\leq \;& \|v^{(m)}\|_{p^*} - 1 \\
\leq \;& k_m^{\frac{1}{p^*}} - 1,
\end{aligned}$$

because $\forall m,\ \|u^{(m)}\|_\infty \leq 1$ for a scaled problem on L', R'.

3.1.2 Error Analysis of Piecewise Method

However, the bounds derived above are only applicable for p^* where $\|u^{(m)}\|_{p^*}^{p^*} \geq \tau$. The piecewise method is slightly more complicated, and can be partitioned into two cases: In the first case, the top contour is used (*i.e.*, when $p^*_{\max}$ is underflow-stable). Conversely, in the second case, a middle contour is used (*i.e.*, when $p^*_{\max}$ is not underflow-stable). In this context, in general a contour comprises of a set of indices m with the same maximum stable p^*.

In the first case, when we use the top contour $p^* = p^*_{\max}$, we know that $p^*_{\max}$ must be underflow-stable, and thus we can reuse the bound given an underflow-stable p^*.

In the second case, because the p^* used is $< p^*_{\max}$, it follows that the next higher contour (using $2p^*$) must not be underflow-stable (because the highest underflow-stable p^* is used and because the p^* are searched in log-space). The bound derived above that demonstrated

$$\|u^{(m)}\|_{p^*} \leq \|u^{(m)}\|_\infty k_m^{\frac{1}{p^*}}$$

can be combined with the property that $\|\cdot\|_{p^*} \geq \|\cdot\|_\infty$ for any $p^* \geq 1$ to show that

$$\|u^{(m)}\|_\infty \in \left[\frac{\|u^{(m)}\|_{p^*}}{k_m^{\frac{1}{p^*}}}, \|u^{(m)}\|_{p^*} \right].$$

Thus the absolute error can also be shown to be bounded using the fact that we are in a middle contour:

$$\begin{aligned}
&= \|u^{(m)}\|_{p^*} - \|u^{(m)}\|_\infty \\
&= \|u^{(m)}\|_{p^*} \left(1 - \frac{\|u^{(m)}\|_\infty}{\|u^{(m)}\|_{p^*}} \right) \\
&\leq \|u^{(m)}\|_{p^*} \left(1 - k_m^{\frac{-1}{p^*}} \right) \\
&< \tau^{\frac{1}{2p^*}} \left(1 - k_m^{\frac{-1}{p^*}} \right).
\end{aligned}$$

The absolute error from middle contours will be quite small when $p^* = 1$ is the maximum underflow-stable value of p^* at index m, because $\tau^{\frac{1}{2p^*}}$, the first factor in the error bound, will become $\sqrt{\tau} \approx 10^{-6}$, and $1 - k_m^{\frac{-1}{p^*}} < 1$ (qualitatively, this indicates that a small p^* is only used when the result is very close to zero, leaving little room for absolute error). Likewise, when a very large p^* is used, then $1 - k_m^{\frac{-1}{p^*}}$ becomes very small, while $\tau^{\frac{1}{2p^*}} < 1$ (qualitatively, this indicates that when a large p^* is used, the $\|\cdot\|_{p^*} \approx \|\cdot\|_\infty$, and thus there is little absolute error). Thus for the extreme values of p^*, middle contours will produce fairly small absolute errors. The unique mode p^*_{mode} can be found by finding the value that solves

$$\frac{\partial}{\partial p^*_{mode}} \left(\tau^{\frac{1}{2p^*_{mode}}} \left(1 - k_m^{\frac{-1}{p^*_{mode}}} \right) \right) = 0,$$

which yields

$$p^*_{mode} = \frac{\log_2(k_m)}{\log_2(-\frac{2\log_2(k) - \log_2(\tau)}{\log_2(\tau)})}.$$

An appropriate choice of $p^*_{\max}$ should be $> p^*_{mode}$ so that the error for any contour (both middle contours and the top contour) is smaller than the error achieved at p^*_{mode}, allowing us to use a single bound for both. Choosing $p^*_{\max} = p^*_{mode}$ would guarantee that all contours are no worse than the middle-contour error at p^*_{mode}; however, using $p^*_{\max} = p^*_{mode}$ is still quite liberal, because it would mean that for indices in the highest contour (there must be a nonempty set of such indices, because the scaling on L' and R' guarantees that the maximum index will have an exact value of 1, meaning that the approximation endures no underflow and is underflow-stable for every p^*), a better error *could* be achieved by increasing $p^*_{\max}$. For this reason, we choose $p^*_{\max}$ so that the top-contour error produced at $p^*_{\max}$ is not substantially larger than all errors produced for p^* before the mode (*i.e.*, for $p^* < p^*_{mode}$).

Choosing any value of $p^*_{\max} > p^*_{mode}$ guarantees the worst-case absolute error bound derived here; however, increasing $p^*_{\max}$ further over p^*_{mode} may possibly improve the mean squared error in practice (because it is possible that many indices in the result would be numerically stable with p^* values substantially larger than p^*_{mode}). However, increasing $p^*_{\max} >> p^*_{mode}$ will produce diminishing returns and generally benefit only a very small number of indices in the result, which have exact values very close to 1. In order to balance these two aims (increasing $p^*_{\max}$ enough over p^*_{mode} but not excessively so), we make a qualitative assumption that a non-trivial number of indices require us to use a p^* below p^*_{mode}; therefore, increasing $p^*_{\max}$ to produce an error significantly smaller than the lowest worst-case error for contours below the mode (*i.e.*, $p^* < p^*_{mode}$) will increase the runtime without significantly decreasing the mean squared error (which will become dominated by the errors from indices that use $p^* < p^*_{mode}$). The lowest worst-case error contour below the mode is $p^* = 1$ (because the absolute error function is unimodal, and thus must be increasing until p^*_{mode} and decreasing afterward); therefore, we heuristically specify that $p^*_{\max}$ should produce a worst-case error on a similar order of magnitude to the worst-case error produced with $p^* = 1$. In practice, specifying the errors at $p^*_{\max}$ and $p^* = 1$ should be equal is very conservative (it produces very large estimates of $p^*_{\max}$, which sometimes benefit only one or two indices in the result); for this reason, we heuristically choose that the worst-case error at $p^*_{\max}$ should be no worse than square root of the worst case error at $p^* = 1$ (this makes the choice of $p^*_{\max}$ less conservative because the errors at $p^* = 1$ are very close to zero, and thus their square root is larger). The square root was chosen because it produced, for the applications described in this paper, the smallest value of $p^*_{\max}$ for which the mean squared error was significantly lower than using $p^*_{\max} = p^*_{mode}$ (the lowest value of $p^*_{\max}$ guaranteed to produce the absolute error bound). This heuristic does satisfy the worst-case bound outlined here (because, again, $p^*_{\max} > p^*_{mode}$), but it could be substantially improved if an expected distribution of magnitudes in the result vector were known ahead of time: prior knowledge

regarding the number of points stable at each p^* considered would enable a well-motivated choice of $p^*_{\max}$ that truly optimizes the expected mean squared error.

From this heuristic choice of $p^*_{\max}$, solving

$$\sqrt{\sqrt{\tau}\left(1 - \frac{1}{k}\right)} = k^{\frac{1}{p^*_{\max}}} - 1$$

(with the square root of the worst-case at $p^* = 1$ on the left and the worst-case error at $p^*_{\max}$ on the right) yields

$$\begin{aligned} p^*_{\max} &= \frac{\log_2(k)}{\log_2(1 + \sqrt{\sqrt{\tau}\left(1 - \frac{1}{k}\right)})} \\ &\approx \frac{\log_2(k)}{\log_2(1 + \sqrt{\sqrt{\tau}})} \end{aligned}$$

for any non-trivial problem (*i.e.*, when $k >> 1$), and thus

$$p^*_{\max} \approx \log_{1+\tau^{\frac{1}{4}}}(k),$$

indicating that the absolute error at the top contour will be roughly equal to the fourth root of τ.

3.1.3 Worst-case Absolute Error

By setting $p^*_{\max}$ in this manner, we guarantee that the absolute error at any index of any unscaled problem on L, R is less than

$$\max_\ell L[\ell] \ \max_r R[r] \ \tau^{\frac{1}{2p^*_{mode}}} \left(1 - k_m^{\frac{-1}{p^*_{mode}}}\right)$$

where p^*_{mode} is defined above. The full formula for the middle-contour error at this value of p^*_{mode} does not simplify and is therefore quite large; for this reason, it is not reported here, but this gives a numeric bound of the worst case middle-contour error that is bound in terms of the variable k (and with no other free variables).

3.1.4 Runtime Analysis

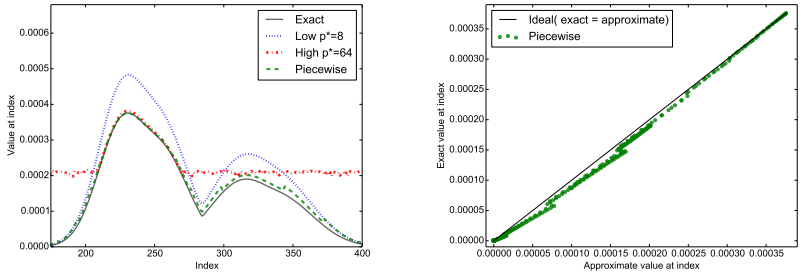
The piecewise method clearly performs $\log_2(p^*_{\max})$ FFTs (each requiring $O(k\log_2(k))$ steps); therefore, since $p^*_{\max}$ is chosen to be $\log_{1+\tau^{\frac{1}{4}}}(k)$ (to achieve the desired error bound), the total runtime is thus

$$O(k\log_2(k)\log_2(\log_{1+\tau^{\frac{1}{4}}}(k)).$$

For any practically sized problem, the $\log_2(\log_{1+\tau^{\frac{1}{4}}}(k))$ factor is essentially a constant; even when k is chosen to be the number of particles in the observable universe ($\approx 2^{270}$; Eddington, 1923), the $\log_2(\log_{1+\tau^{\frac{1}{4}}}(k))$ is ≈ 18, meaning that for any problem of practical size, the full piecewise method is no more expensive than computing between 1 and 18 FFTs.

3.2 Comparison of Low-Value $p^* = 8$, High-value $p^* = 64$, and Piecewise Method

We first use an example max-convolution problem to compare the results from the low-value $p^* = 8$, the high-value $p^* = 64$ and piecewise methods. At every index, these various approximation results are compared to the exact values, as computed by the naive quadratic method (**Figure 2a**).



(a) (b)

Figure 2: **The accuracy of numerical fast max-convolution methods. (a)** Different approximations for a sample max-convolution problem. The low-p^* method is underflow-stable, but overestimates the result. The high-p^* method is accurate when underflow-stable, but experiences underflow at many indices. The piecewise method stitches together approximations from different p^* to maintain underflow-stability. **(b)** Exact vs. piecewise approximation at various indices of the same problem. A clear banding pattern is observed with one tight, elliptical cluster for each contour. The slope of the clusters deviates more for the contours using lower p^* values.

3.3 Improved Affine Piecewise Method

Figure 2b depicts a scatter plot of the exact result vs. the piecewise approximation at every index (using the same problem from **Figure 2a**). It shows a clear banding pattern: the exact and approximate results are clearly correlated, but each contour (*i.e.*, each collection of indices that use a specific p^*) has a different average slope between the exact and approximate values, with higher p^* contours showing a generally larger slope and smaller p^* contours showing greater spread and lower slopes. This intuitively makes sense, because the bounds on $\|u^{(m)}\|_\infty \in [\|u^{(m)}\|_{p^*} k_m^{\frac{-1}{p^*}}, \|u^{(m)}\|_{p^*}]$ derived above constrain the scatter plot points inside a quadrilateral envelope (**Figure 3**).

The correlations within each contour can be exploited to correct biases that emerge for smaller p^* values. In order to do this, $\|u^{(m)}\|_\infty$ must be computed for at least two points m_1 and m_2 within the contour, so that a mapping $\|u^{(m)}\|_{p^*} \approx f(\|u^{(m)}\|_{p^*}) = a\|u^{(m)}\|_{p^*} + b$ can be constructed. Fortunately, a single $\|u^{(m)}\|_\infty$ can be computed exactly in $O(k)$ (by actually computing a single $u^{(m)}$ and computing its max, which is equivalent to computing a single index result via the naive quadratic method). As long as the exact value $\|u^{(m)}\|_\infty$ is computed for only a small number of indices, the order of the runtime will not change (each contour already costs $O(k \log_2(k))$, so adding a small number of $O(k)$ steps for each contour will not change the asymptotic runtime).

If the two indices chosen are

$$m_{\min} = \operatorname*{argmin}_{m \in contour(p^*)} \|u^{(m)}\|_{p^*}$$

and

$$m_{\max} = \operatorname*{argmax}_{m \in contour(p^*)} \|u^{(m)}\|_{p^*},$$

then we are guaranteed that the affine function f can be written as a convex combination of the exact values at those extreme points (using barycentric coordinates):

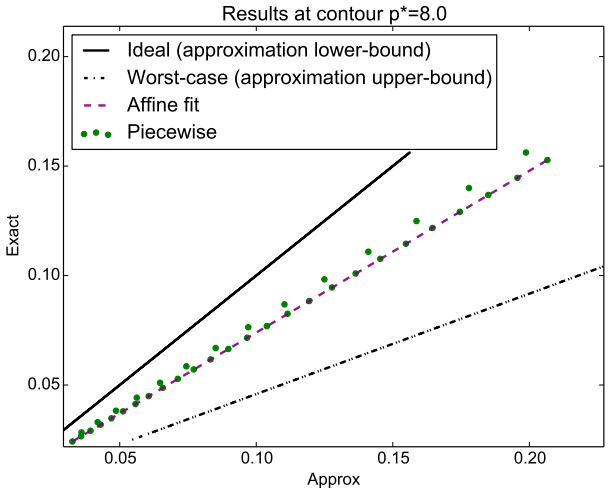
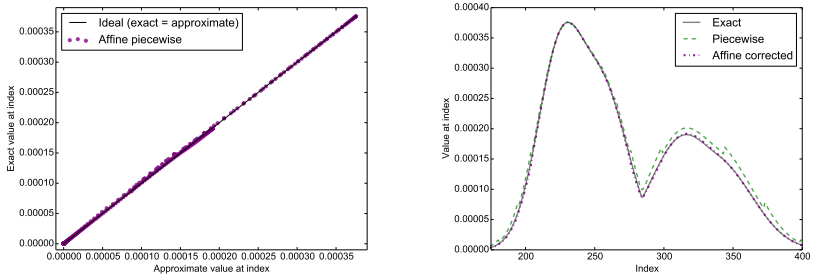


Figure 3: **A single contour from the piecewise approximation.** The cluster of points (one point for each index in the previous figure) is bounded by the exact value (ideal approximation) and the approximation upper-bound for $p^* = 8$ (worst-case approximation). The points are well described by an affine function fit using the left-most and right-most points.



(a)

(b)

Figure 4: **Piecewise method with affine contour fitting.** The approximate values at each index of the max-convolution problem are almost identical to the exact result at the same index.

Algorithm 3 Improved affine piecewise numerical max-convolution, a numerical method to estimate the max-convolution nonnegative vectors (further revised to reduce numerical error). This procedure uses a p^* close to the largest possible stable value at each result index. The return value is a numerical estimate of the max-convolution $L *_{\max} R$. The runtime is in $O(k \log_2(k) \log_2(p^*_{\max}))$.

procedure NUMERICALMAXCONVOLVEPIECEWISEAFFINE(L, R, $p^*_{\max}$)
  $\ell_{\max} \leftarrow \operatorname{argmax}_\ell L[\ell]$
  $r_{\max} \leftarrow \operatorname{argmax}_r R[r]$
  $L' \leftarrow \frac{L}{L[\ell_{\max}]}$
  $R' \leftarrow \frac{R}{R[r_{\max}]}$ ▷ Scale to a proportional problem on L', R'
  $allPStar \leftarrow [2^0, 2^1, \ldots, 2^{\lceil \log_2(p^*_{\max}) \rceil}]$
  for $i \in \{0, 1, \ldots len(allPStar)\}$ **do**
    $resForAllPStar[i] \leftarrow$ `fftNonnegMaxConvolveGivenPStar`$(L', R', allPStar[i])$
  end for
  for $m \in \{0, 1, \ldots len(L) + len(R) - 1\}$ **do**
    $maxStablePStarIndex[m] \leftarrow \max\{i : (resForAllPStar[i][m])^{\text{allPStar}[i]} \geq \tau)\}$
  end for
  $result \leftarrow$ `affineCorrect`$(resForAllPStar, maxStablePStarIndex)$
  return $L[\ell_{\max}] \times R[r_{\max}] \times result$ ▷ Undo previous scaling
end procedure

$$f(\|u^{(m)}\|_{p^*}) = \lambda_m \|u^{(m_{\max})}\|_\infty + (1 - \lambda_m) \|u^{(m_{\min})}\|_\infty$$

$$\lambda_m = \frac{\|u^{(m)}\|_{p^*} - \|u^{(m_{\min})}\|_{p^*}}{\|u^{(m_{\max})}\|_{p^*} - \|u^{(m_{\min})}\|_{p^*}} \in [0, 1]$$

Thus, by computing $\|u^{(m_{\min})}\|_\infty$ and $\|u^{(m_{\max})}\|_\infty$ (each in $O(k)$ steps), we can compute an affine function f to correct contour-specific trends (**Algorithm 3**).

3.3.1 Error Analysis of Improved Affine Piecewise Method

By exploiting the convex combination used to define f, the absolute error of the affine piecewise method can also be bound. Qualitatively, this is because, by fitting on the extrema in the contour, we

Algorithm 4 Subroutine for correcting errors in a contour, with an affine transformation based on exact boundary points. It needs the results of the evaluation of the different p-norms as well as the (index of the) maximum stable values of p^* at every index.

procedure AFFINECORRECT($resForAllPStar$, $maxStablePStarIndex$)
  $\forall i, slope[i] \leftarrow 1$
  $\forall i, bias[i] \leftarrow 0$
  $usedPStar \leftarrow set(maxStablePStarIndex)$
  for $i \in usedPStar$ **do**
    $contour \leftarrow \{m : maxStablePStarIndex[m] = i\}$
    $mMin \leftarrow argmin_{m \in contour} resForAllPStar[i][m]$
    $mMax \leftarrow argmax_{m \in contour} resForAllPStar[i][m]$
    $xMin \leftarrow resForAllPStar[i][mMin]$
    $xMax \leftarrow resForAllPStar[i][mMax]$
    $yMin \leftarrow$ `maxConvolutionAtIndex`$(mMin)$
    $yMax \leftarrow$ `maxConvolutionAtIndex`$(mMax)$
    if $xMax > xMin$ **then**
      $slope[i] \leftarrow \frac{yMax - yMin}{xMax - xMin}$
      $bias[i] \leftarrow yMin - slope[i] \times xMin$
    else
      $slope[i] \leftarrow \frac{yMax}{xMax}$
    end if
  end for
  for $m \in \{0, 1, \ldots len(L) + len(R) - 1\}$ **do**
    $i \leftarrow maxStablePStarIndex[m]$
    $result[m] \leftarrow resForAllPStar[i][m] \times slope[i] + bias[i]$
  end for
  return $result$
end procedure

are now interpolating. If the two points used to determine the parameters of the affine function were not chosen in this manner to fit the affine function, then it would be possible to choose two points with very close x-values (*i.e.*, similar approximate values) and disparate y-values (*i.e.*, different exact values), and extrapolating to other points could propagate a large slope over a large distance; using the extreme points forces the affine function to be a convex combination of the extrema, thereby avoiding this problem.

$$
\begin{aligned}
& f(\|u^{(m)}\|_{p^*}) = \lambda_m \|u^{(m_{\max})}\|_\infty + (1-\lambda_m)\,\|u^{(m_{\min})}\|_\infty \\
\in & \left[\lambda_m \frac{\|u^{(m_{\max})}\|_{p^*}}{k_{m_{\max}}^{\frac{1}{p^*}}} + (1-\lambda_m)\,\frac{\|u^{(m_{\min})}\|_{p^*}}{k_{m_{\min}}^{\frac{1}{p^*}}},\right. \\
& \qquad \left. \lambda_m \|u^{(m_{\max})}\|_{p^*} + (1-\lambda_m)\,\|u^{(m_{\min})}\|_{p^*}\right] \\
\subseteq & \left[\lambda_m \frac{\|u^{(m_{\max})}\|_{p^*}}{k^{\frac{1}{p^*}}} + (1-\lambda_m)\,\frac{\|u^{(m_{\min})}\|_{p^*}}{k^{\frac{1}{p^*}}},\right. \\
& \qquad \left. \lambda_m \|u^{(m_{\max})}\|_{p^*} + (1-\lambda_m)\,\|u^{(m_{\min})}\|_{p^*}\right] \\
= & \left[k^{\frac{-1}{p^*}}\left(\lambda_m \|u^{(m_{\max})}\|_{p^*} + (1-\lambda_m)\,\|u^{(m_{\min})}\|_{p^*}\right),\right. \\
& \qquad \left. \lambda_m \|u^{(m_{\max})}\|_{p^*} + (1-\lambda_m)\,\|u^{(m_{\min})}\|_{p^*}\right] \\
= & \left[k^{\frac{-1}{p^*}}\|u^{(m)}\|_{p^*}, \|u^{(m)}\|_{p^*}\right]
\end{aligned}
$$

The worst-case absolute error of the scaled problem on L', R' can be defined as

$$\max_m |\ f(\|u^{(m)}\|_{p^*}) - \|u^{(m)}\|_\infty\ |.$$

Because the function $f(\|u^{(m)}\|_{p^*}) - \|u^{(m)}\|_\infty$ is affine, it's derivative can never be zero, and thus Lagrangian theory states that the maximum must occur at a boundary point. Therefore, the worst-case absolute error is

$$
\begin{aligned}
\leq & \quad \max\{\|u^{(m)}\|_{p^*} - \|u^{(m)}\|_\infty, \|u^{(m)}\|_\infty - \|u^{(m)}\|_{p^*} k^{\frac{-1}{p^*}}\} \\
= & \quad \|u^{(m)}\|_{p^*} - \|u^{(m)}\|_\infty,
\end{aligned}
$$

which is identical to the worst-case error bound before applying the affine transformation f. Thus applying the affine transformation can dramatically improve error, but will not make it worse than the original worst-case.

3.4 An Improved Approximation of the Chebyshev Norm

In order to improve the error of the piecewise numerical method, we consider the worst-case, when $\frac{u^{(m)}}{\|u^{(m)}\|_\infty} = (1, 1, \ldots 1)$. In this case, computing any two norms of $u^{(m)}$ (at p_1^* and p_2^*) would be sufficient to solve exactly for $\|u^{(m)}\|_\infty$, because

$$
\begin{aligned}
\|u^{(m)}\|_{p_1^*}^{p_1^*} & \propto \|u^{(m)}\|_\infty^{p_1^*} \\
\|u^{(m)}\|_{p_2^*}^{p_2^*} & \propto \|u^{(m)}\|_\infty^{p_2^*},
\end{aligned}
$$

where the proportionality constant is $k_m = len(u^{(m)})$.

Thus we see that although the p^*-norm approximation of the Chebyshev norm is a good approximation, the curve of the norms (at various different p^*) values holds far more information than a simple point estimate would. Therefore, we derive an improved algorithm by proceeding as follows: First, we note that, rather than computing the p^*-norm of a vector $u^{(m)}$ by summing all elements of $u^{(m)}$ taken to the power p^*, it is possible to equivalently sum over only the unique values of $u^{(m)}$ (here denoted in $\beta_1, \beta_2, \ldots$) taken to the p^* where each term in the sum is weighted by the number of occurrences of each (denoted $h_1, h_2, \ldots$, respectively). For this reason, we can then use a sequence of norms to compute a (potentially smaller) collection of approximate unique values $\alpha_1, \alpha_2, \ldots \alpha_r$, where once again, the p^*-norm is equal to the sum over those unique values to the p^*, where each term in the sum is weighted by numbers of occurrences $n_1, n_2, \ldots n_r$. Therefore, given $2r$ different norms of $u^{(m)}$, it is possible to project and estimate r unique α_i values. The maximum of those $\alpha_1, \alpha_2, \ldots \alpha_r$ values, (the $\alpha_1, \alpha_2, \ldots \alpha_r$ values can be thought of as projections of the true unique values $\beta_1, \beta_2, \ldots$) can then be used as an estimate of the true maximum element in $u^{(m)}$, which we will now demonstrate.

Specifically, where k_m is the length of $u^{(m)}$ and where there are $e_m \leq k_m$ unique values (β_i) in $u^{(m)}$, we can model the norms perfectly with

$$\|u^{(m)}\|_{p^*}^{p^*} = \sum_i^{e_m} h_i \beta_i^{p^*}$$

where h_i is an integer that indicates the number of times β_i occurs in $u^{(m)}$ (and where $\sum_i h_i = k_m = len(u^{(m)})$). This multi-set view of the vector $u^{(m)}$ can be used to project it down to a dimension r:

$$
\begin{bmatrix}
\alpha_1^{p^*} & \alpha_2^{p^*} & \cdots & \alpha_r^{p^*} \\
\alpha_1^{2p^*} & \alpha_2^{2p^*} & \cdots & \alpha_r^{2p^*} \\
\alpha_1^{3p^*} & \alpha_2^{3p^*} & \cdots & \alpha_r^{3p^*} \\
\vdots & \vdots & & \vdots \\
\alpha_1^{\ell p^*} & \alpha_2^{\ell p^*} & \cdots & \alpha_r^{\ell p^*}
\end{bmatrix}
\cdot
\begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ \vdots \\ n_r \end{bmatrix}
=
\begin{bmatrix}
\|u^{(m)}\|_{p^*}^{p^*} \\
\|u^{(m)}\|_{2p^*}^{2p^*} \\
\|u^{(m)}\|_{3p^*}^{3p^*} \\
\vdots \\
\|u^{(m)}\|_{\ell p^*}^{\ell p^*}
\end{bmatrix}.
$$

By solving the above system of equations for all α_i, the maximum $\hat{\alpha} = \max_i \alpha_i$ can be used to approximate the true maximum $\max_i \beta_i = \|u^{(m)}\|_\infty$. This projection can be thought of as querying distinct moments of the distribution $\text{pmf}_{U^{(m)}}$ that corresponds to some unknown vector $u^{(m)}$, and then assembling the moments into a model in order to predict the unknown maximum value in $u^{(m)}$. Of course, when r, the number of terms in our model, is sufficiently large, then computing r norms of $u^{(m)}$ will result in an exact result, but it could result in $O(k_m)$ execution time, meaning that our numerical max-convolution algorithm becomes quadratic; therefore, we must consider that a small number of distinct moments are queried in order to estimate the maximum value in $u^{(m)}$. Regardless, the system of equations above is quite difficult to solve directly via elimination for even very small values of r, because the symbolic expressions become quite large and because symbolic polynomial roots cannot be reliably computed when the degree of the polynomial is > 5. Even in cases when it can be solved directly, it will be far too inefficient.

For this reason, we solve for the α_i values using an exact, alternative approach: If we define a polynomial $\gamma(x) = \left(x - \alpha_1^{p^*}\right)\left(x - \alpha_2^{p^*}\right)\cdots\left(x - \alpha_r^{p^*}\right)$, then $x \in \{\alpha_1^{p^*}, \alpha_2^{p^*}, \ldots \alpha_r^{p^*}\} \Leftrightarrow \gamma(x) = 0$. We can expand $\gamma(x) = \gamma_0 + \gamma_1 x + \gamma_2 x^2 + \cdots + \gamma_r x^r$, and then write

$$\begin{bmatrix} \gamma_0 & \gamma_1 & \gamma_2 & \cdots & \gamma_r \end{bmatrix} \cdot \begin{bmatrix} \alpha_1^{p^*} & \alpha_2^{p^*} & & \alpha_r^{p^*} \\ \alpha_1^{2p^*} & \alpha_2^{2p^*} & \cdots & \alpha_r^{2p^*} \\ \alpha_1^{3p^*} & \alpha_2^{3p^*} & & \alpha_r^{3p^*} \\ \vdots & \vdots & & \vdots \\ \alpha_1^{\ell p^*} & \alpha_2^{\ell p^*} & & \alpha_r^{\ell p^*} \end{bmatrix} \cdot \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ \vdots \\ n_r \end{bmatrix} =$$

$$\begin{bmatrix} \alpha_1^{p^*}\gamma(\alpha_1^{p^*}) & \alpha_2^{p^*}\gamma(\alpha_2^{p^*}) & \alpha_3^{p^*}\gamma(\alpha_3^{p^*}) & \cdots & \alpha_r^{p^*}\gamma(\alpha_r^{p^*}) \end{bmatrix} \cdot \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ \vdots \\ n_r \end{bmatrix} =$$

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \cdot \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ \vdots \\ n_r \end{bmatrix} = 0,$$

which indicates that

$$\begin{bmatrix} \gamma_0 & \gamma_1 & \gamma_2 & \cdots & \gamma_r \end{bmatrix} \cdot \begin{bmatrix} \|u^{(m)}\|_{p^*}^{p^*} \\ \|u^{(m)}\|_{2p^*}^{2p^*} \\ \|u^{(m)}\|_{3p^*}^{3p^*} \\ \vdots \\ \|u^{(m)}\|_{\ell p^*}^{\ell p^*} \end{bmatrix} = 0.$$

Furthermore, $\gamma(x) = 0, x \neq 0 \Leftrightarrow x^i\gamma(x) = 0, i \in \mathbb{N}$; therefore we can write

$$\begin{bmatrix} \gamma_0 & \gamma_1 & \gamma_2 & \cdots & \gamma_r & 0 & 0 & \cdots & 0 \\ 0 & \gamma_0 & \gamma_1 & \gamma_2 & \cdots & \gamma_r & 0 & \cdots & 0 \\ 0 & 0 & \gamma_0 & \gamma_1 & \gamma_2 & \cdots & \gamma_r & \cdots & 0 \\ & & & & \vdots & & & & \\ 0 & 0 & \cdots & 0 & \gamma_0 & \gamma_1 & \gamma_2 & \cdots & \gamma_r \end{bmatrix} \cdot \begin{bmatrix} \|u^{(m)}\|_{p^*}^{p^*} \\ \|u^{(m)}\|_{2p^*}^{2p^*} \\ \|u^{(m)}\|_{3p^*}^{3p^*} \\ \vdots \\ \|u^{(m)}\|_{\ell p^*}^{\ell p^*} \end{bmatrix} =$$

$$\begin{bmatrix} \|u^{(m)}\|_{p^*}^{p^*} & \|u^{(m)}\|_{2p^*}^{2p^*} & \|u^{(m)}\|_{3p^*}^{3p^*} & \cdots & \|u^{(m)}\|_{(r+1)p^*}^{(r+1)p^*} \\ \|u^{(m)}\|_{2p^*}^{2p^*} & \|u^{(m)}\|_{3p^*}^{3p^*} & \|u^{(m)}\|_{4p^*}^{4p^*} & \cdots & \|u^{(m)}\|_{(r+2)p^*}^{(r+2)p^*} \\ \|u^{(m)}\|_{3p^*}^{3p^*} & \|u^{(m)}\|_{4p^*}^{4p^*} & \|u^{(m)}\|_{5p^*}^{5p^*} & \cdots & \|u^{(m)}\|_{(r+3)p^*}^{(r+3)p^*} \\ \vdots & \vdots & \vdots & & \vdots \\ \|u^{(m)}\|_{(\ell-r)p^*}^{(\ell-r)p^*} & \|u^{(m)}\|_{(\ell-r+1)p^*}^{(\ell-r+1)p^*} & \|u^{(m)}\|_{(\ell-r+2)p^*}^{(\ell-r+2)p^*} & \cdots & \|u^{(m)}\|_{\ell p^*}^{\ell p^*} \end{bmatrix} \cdot \begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_r \end{bmatrix} = 0.$$

Therefore,

$$\begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_r \end{bmatrix} \in null\left(\begin{bmatrix} \|u^{(m)}\|_{p^*}^{p^*} & \|u^{(m)}\|_{2p^*}^{2p^*} & \|u^{(m)}\|_{3p^*}^{3p^*} & \cdots & \|u^{(m)}\|_{(r+1)p^*}^{(r+1)p^*} \\ \|u^{(m)}\|_{2p^*}^{2p^*} & \|u^{(m)}\|_{3p^*}^{3p^*} & \|u^{(m)}\|_{4p^*}^{4p^*} & \cdots & \|u^{(m)}\|_{(r+2)p^*}^{(r+2)p^*} \\ \|u^{(m)}\|_{3p^*}^{3p^*} & \|u^{(m)}\|_{4p^*}^{4p^*} & \|u^{(m)}\|_{5p^*}^{5p^*} & \cdots & \|u^{(m)}\|_{(r+3)p^*}^{(r+3)p^*} \\ \vdots & \vdots & \vdots & & \vdots \\ \|u^{(m)}\|_{(\ell-r)p^*}^{(\ell-r)p^*} & \|u^{(m)}\|_{(\ell-r+1)p^*}^{(\ell-r+1)p^*} & \|u^{(m)}\|_{(\ell-r+2)p^*}^{(\ell-r+2)p^*} & \cdots & \|u^{(m)}\|_{\ell p^*}^{\ell p^*} \end{bmatrix}\right).$$

Because the columns of

$$\begin{bmatrix} \alpha_1^{p^*} & \alpha_2^{p^*} & & \alpha_r^{p^*} \\ \alpha_1^{2p^*} & \alpha_2^{2p^*} & \cdots & \alpha_r^{2p^*} \\ \alpha_1^{3p^*} & \alpha_2^{3p^*} & & \alpha_r^{3p^*} \\ \vdots & \vdots & & \vdots \\ \alpha_1^{\ell p^*} & \alpha_2^{\ell p^*} & & \alpha_r^{\ell p^*} \end{bmatrix}$$

must be linearly independent when $\alpha_1, \alpha_2, \ldots$ are distinct (which is the case by the definition of our multiset formulation of the norm), then $r = \frac{\ell}{2}$ will determine a unique solution; thus the null space above is computed from a matrix with $r + 1$ columns and r rows, yielding a single vector for $(\gamma_0, \gamma_1, \ldots \gamma_r)$. This vector can then be used to compute the roots of the polynomial $\gamma_0 + \gamma_1 x + \gamma_2 x^2 + \cdots + \gamma_r x^r$, which will determine the values $\{\alpha_1^{p^*}, \alpha_2^{p^*}, \ldots \alpha_r^{p^*}\}$, which can each be taken to the $\frac{1}{p^*}$ power to compute $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$; the largest of those α_i values is used as the estimate of the maximum element in $u^{(m)}$. When $u^{(m)}$ contains at least r distinct values (*i.e.*, $e_m \geq r$), then the problem will be well-defined; thus, if the roots of the null space spanning vector are not well-defined, then a smaller r can be used (and should be able to compute an exact estimate of the maximum, since $u^{(m)}$ can be projected exactly when r is the precise number of unique elements found in $u^{(m)}$).

To summarize, the null space projection method is performed as follows: First, standard FFT-based convolution is used for each p^* to compute the different norms at every index m. Then those norms are used to populate and compute the null space of a matrix. Finally the roots of the polynomial whose coefficients are given by the null space vector are computed to estimate the different $\alpha_1, \alpha_2, \ldots$. Note that this projection method is valid for any sequence of norms with even spacing: $\|u^{(m)}\|_{p_0+p^*}^{p_0+p^*}, \|u^{(m)}\|_{p_0+2p^*}^{p_0+2p^*}, \|u^{(m)}\|_{p_0+3p^*}^{p_0+3p^*}, \ldots \|u^{(m)}\|_{p_0+\ell p^*}^{p_0+\ell p^*}$.

The null space projection method can therefore be computed for an arbitrary r (*i.e.*, it can be used to project to an arbitrary number of unique elements in $u^{(m)}$) by using linear algebra to compute the null space and to compute the roots of the polynomial (for instance, using the `numpy.roots` command in Python). For greater efficiency, setting r to a small constant allows us to precompute closed-form solutions of both the null space and the polynomial roots. For instance, using $r = 2$ (which is equivalent to projecting each vector $u^{(m)}$ to two unique values) results in a $\mathbb{R}^{3\times 2}$ null space computation and computing roots of a quadratic polynomial, both of which can be done in closed form **Algorithm 5**. Details of this $r = 2$ case can be found in **Appendix 5**. In this case, it is possible to construct a series of powers of two with interleaved points, which guarantees that 4 evenly spaced p^* values exist (when the highest numerically stable p^* is higher than 2), meaning that the number of FFT calls is only $2\times$ what would be used by the original piecewise method (rather than $4\times$ the calls, which would be necessary if 4 evenly spaced points were placed at each considered p^* in a naive scheme). From algebraic and empirical evidence, we conjecture that the relative error of this $r = 2$ null space projection is bounded above by $1 - 0.7^{\frac{4}{p^*}}$, where p^* is the highest numerically stable p^* value.

This relative error is superior to the worst-case relative error when using a single p^*-norm estimate of the maximum. The relative error using the null space projection decreases rapidly as p^* is increased, meaning that the same procedure can be used to achieve an absolute error bound:

$$\hat{\alpha} - \|u^{(m)}\|_\infty < \tau^{\frac{1}{2p^*}}\left(1 - 0.7^{\frac{4}{p^*}}\right),$$

which achieves a unique maximum at

$$p^*_{mode} = \frac{1.4267 * \log(\tau) - 4.07094}{(\log(\tau) - 2.8534)\left(\log(1 - \frac{2.8534}{\log(\tau)})\right)} \approx 14.52.$$

Algorithm 5 Piecewise numerical max-convolution with projection, a numerical method to estimate the max-convolution of two PMFs or nonnegative vectors. This method uses a nullspace projection to achieve a closer estimate of the true maximum. Depending on the number of stable estimates, linear or quadratic projection is used. The parameters are two nonnegative vectors L' and R' (both scaled so that they have maximal element 1). The return value is a numerical estimate of the max-convolution $L' *_{\max} R'$.

procedure numericalMaxConvolvePiecewiseProjectionAffine(L', R', p^*)
  $\ell_{\max} \leftarrow \text{argmax}_\ell L[\ell]$
  $r_{\max} \leftarrow \text{argmax}_r R[r]$
  $L' \leftarrow \frac{L}{L[\ell_{\max}]}$
  $R' \leftarrow \frac{R}{R[r_{\max}]}$ ▷ Scale to a proportional problem on L', R'
  $allPStar \leftarrow [2^{-1}, 2^0, 2^1, \ldots, 2 + 2^{\left\lfloor \log_2(p^*_{\max}) \right\rfloor}]$
  for $h \in \{0, 1, \ldots len(allPStar)\}$ **do**
    $allPStarInterleaved[2i] \leftarrow allPStar[i]$
    $allPStarInterleaved[2i+1] \leftarrow 0.5 \times (allPStar[i] + allPStar[i+1])$
  end for
  for $i \in \{0, 1, \ldots len(allPStar)\}$ **do**
    $resForAllPStar[i] \leftarrow$ `fftNonnegMaxConvolveGivenPStar`(L', R', $allPStarInterleaved[i]$)
  end for
  for $m \in \{0, 1, \ldots len(L) + len(R) - 1\}$ **do**
    $maxStablePStarIndex[m] \leftarrow \max\{i : (resForAllPStar[i][m])^{\text{allPStarInterleaved}[i]} \geq \tau)\}$
  end for
  for $o \in \{0, 1, \ldots len(maxStablePStarIndex)\}$ **do**
    $maxStablePStarIndex[o] -= maxStablePStarIndex[o] \% 2$ ▷ Restrict to powers of 2
  end for
  for $p \in \{0, 1, \ldots len(maxStablePStarIndex)\}$ **do**
    $maxP \leftarrow allPStarInterleaved[maxStablePStarIndex[p]]$
    $spacing \leftarrow 0.25 * maxP$
    $est_4 \leftarrow resForAllPStar[maxStablePStarIndex[p]]$
    $est_3 \leftarrow resForAllPStar[maxStablePStarIndex[p] - 1]$
    if $maxStablePStarIndex[p] < 5$ **then** ▷ Need 5 p^* in sequence to get 4 evenly spaced
      $resForAllPStar[p] \leftarrow$ `maxLin`(est_3, est_4)
    else
      $est_2 \leftarrow resForAllPStar[maxStablePStarIndex[p] - 2]$
      $est_1 \leftarrow resForAllPStar[maxStablePStarIndex[p] - 4]$ ▷ Index - 4 is the next evenly spaced point
      $resForAllPStar[p] \leftarrow$ `maxQuad`$(est_1, est_2, est_3, est_4, spacing)$
    end if
  end for
  $result \leftarrow$ `affineCorrect`$(resForAllPStar, maxStablePStarIndex)$
  return $L[\ell_{\max}] \times R[r_{\max}] \times result$ ▷ Undo previous scaling
end procedure

As before, the worst-case absolute error of the unscaled problem will be found by simply scaling the absolute error at p^*_{mode}:

$$\max_\ell L[\ell] \max_r R[r] \tau^{\frac{1}{2p^*_{mode}}} \left(1 - 0.7^{\frac{4}{p^*_{mode}}}\right).$$

Because p^*_{mode} (the value of p^* producing the worst-case absolute error) for the null space projection method it is invariant to the length of the list k (enabling us to compute a numeric value), and because its numeric value is so small, even a fairly small choice of $p^*_{\max}$ will suffice (now $p^*_{\max} \in O(1)$ rather than in $O(\log(k))$ as it was with the original piecewise method).

The one caveat of this worst-case absolute error bound is that it presumes at least four evenly spaced, stable p^* can be found (which may not be the case by choosing p^* from the sequence 2^i in cases when $\|u^{(m)}\|_\infty \approx 0$); however, assuming standard fast convolution can be performed (a reasonable assumption given it is one of the essential numeric algorithms), then four evenly spaced p^* values could be chosen very close to 1; therefore, these values of p^* could be added to the sequence so that the algorithm is slightly slower, but essentially always yield this worst-case absolute error bound.

In practice, we can demonstrate that the null space projection method is very accurate. First we show the impact of using the quadratic (*i.e.*, $r = 2$) projection method on unscaled single $u^{(m)}$ vectors. The projection method was tested on vectors of different lengths drawn from different types of Beta distributions and are compared with the results of the p-norms with the highest stable p (**Figure 5**). The relative errors between the original piecewise method and the null space projection method are compared using a max-convolution on two randomly created input PMFs of lengths 1024 (**Figure 6**). Note that the null space projection can also be paired with affine scaling on the back end, just as the original piecewise method can be. In practice, the null space projection increases the accuracy demonstrably on a variety of different problems, although the original piecewise method also performs well.

Although the worst-case runtime of the null space projection method is roughly 2× that of the original piecewise method, the error bound no longer depends on the length of the result k. Thus, for a given relative error bound on the top contour (*i.e.*, the equivalent of the derivation of $p^*_{\max}$ in the original piecewise algorithm), the value of $p^*_{\max}$ is fixed and no longer $\in O(\log(k))$. For example, achieving a 0.5% relative error in the top contour would require

$$1 - 0.7^{\frac{4}{p^*_{\max}}} \leq 0.005 \rightarrow p^*_{\max} \geq 4\frac{\log(0.7)}{\log(0.995)} \approx 284.62,$$

meaning that choosing $p^*_{\max} = 512$ would achieve a very high accuracy, but while only performing 2×9 FFTs. For very large vectors, this will not be substantially more expensive than the original piecewise algorithm, which uses a higher value of $p^*_{\max}$ (in this case, $p^*_{\max} = \log_{1.005}(k)$, which continues to grow as k does) to keep the error lower in practice. As a result, the runtime of the null space projection approximation is $\in O(k \log(k))$ rather than $O(k \log(k) \log(\log(k)))$, despite the similar runtime in practice to the original piecewise method (the null space projection method uses 2× as many FFTs performed per p^* value, but requires slightly fewer p^* values).

3.5 Practical Runtime Comparison

To compare the actual runtimes of the final algorithm developed in this manuscript with a naive max-convolution and a previously proposed method from Bussieck et al. (1994), all methods were run on vectors of different random (uniform in $[0, 1]$) composition and length (k). The first and second input vector were generated seperately but are always of same length. **Table 1** shows the result of this experiment. All methods were implemented in Python, using `numpy` where applicable (*e.g.*, to vectorize). A non-vectorized version of naive max-convolution was included to estimate the effects of vectorization. The approach from Bussieck et al. ran as a reimplementation based on the

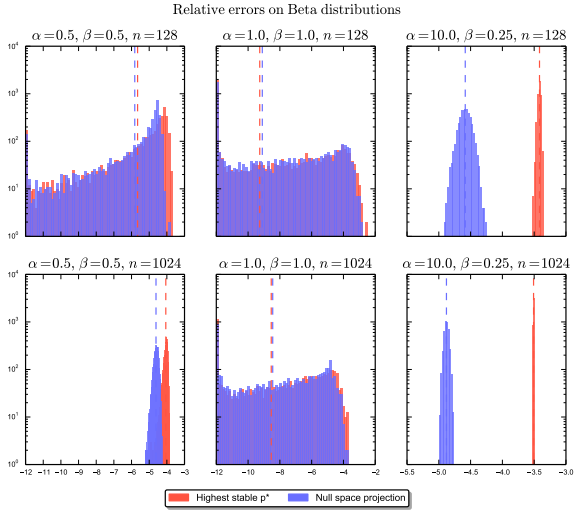


Figure 5: **Relative errors on random vectors with and without null space projection.** For the two approximation methods (using the highest stable p^*-norm with the heuristically chosen $p^*_{\max}$ or using the null space projection with $p^*_{\max} = 64$), vectors of different lengths are sampled (2^{12} repetitions) from a variety of Beta distributions. The settings for the parameters (α, β) of the Beta distribution that were used, as well as the lengths of the generated vectors are shown in the titles of the subplots: $\alpha = 0.5, \beta = 0.5$ (bimodal with modes near zero and one); $\alpha = 0.1, \beta = 0.1$ (uniform distribution); $\alpha = 10, \beta = 0.25$ (with a strong mode near one). The red area depicts the frequencies (y-axis) of the different magnitudes of (relative) error (x-axis) when using the highest stable p^*-norm is used as an approximation of the Chebyshev norm ($p = \infty$). The blue area shows the errors with the method that performs a projection (either quadratic or linear depending on how many numerically stable p^* are available) to estimate the Chebyshev norm.

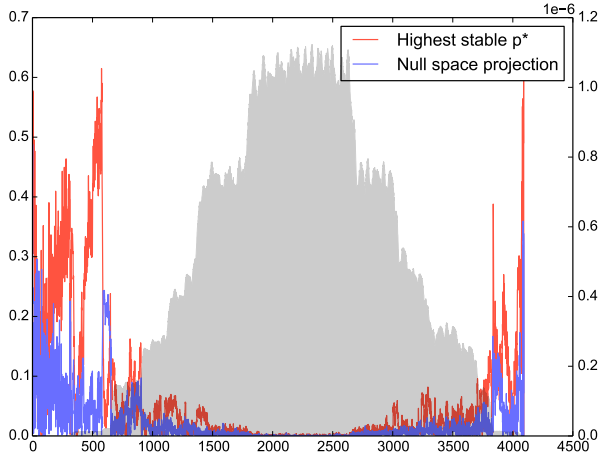


Figure 6: **Relative errors on large max-convolution with and without null space projection.** Max-convolution between two randomly generated vectors (both uniform vectors convolved with narrow Gaussians with uniform noise added afterward), performed with the highest stable p^*-norm (using the heuristic choice of $p^*_{\max}$ for problems of this size) and with null space projection (using $p^*_{\max} = 64$). The left y-axis shows the relative error at index m. Associated with that, you can see the red and blue curve depicting the errors from the two different methods: Red describes the max-norm estimation using only the highest stable p^* while purple was generated using quadratic projection at the four highest stable p^* values (when at least four evenly spaced values are numerically stable) and linear projection at the two highest stable p^* values (when only two p^* are numerically stable). The results of both approaches are corrected with the affine transformation method proposed in this manuscript. In the background the gray shaded curve shows the exact result of the max-convolution at every index (to be used with the second y-axis on the right).

k	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}
Naive	**0.0142**	0.0530	0.192	0.767	3.03	12.1	48.2
Naive (vectorized)	0.0175	0.0381	0.0908	0.251	0.790	2.75	10.1
FILL1 (Bussieck et al., 1994)	0.0866	1.09	7.21	19.4	457	—	—
Max. stable p^*, affine corrected	0.0277	0.0353	0.0533	0.0848	0.149	0.274	0.537
Projection, affine corrected	0.0236	**0.0307**	**0.0467**	**0.0760**	**0.137**	**0.258**	**0.520**

Table 1: **Runtimes of different methods for max-convolution on uniform vectors of length k.** The runtimes were gathered using the `timeit` package in Python. They include all preprocessing steps necessary for the algorithm (*e.g.*, sorting prior to the FILL1 approach). The values are total runtimes (in seconds) to run 5 repetitions on different, randomly generated vectors. FILL1 was not run on larger problems, because it ran substantially longer than the non-vectorized naive approach. On the two approximation methods presented in this manuscript, the highest stable p^*-norm approximation was run with the heuristically chosen $p^*_{\max}$ for problems of the appropriate size and the null space projection was run with $p^*_{\max} = 64$.

pseudocode in their manuscript. From their variants of proposed methods, FILL1 was chosen because of its use in their corresponding benchmark and its recommendation by the authors for having a lower runtime constant in practice compared to other methods they proposed. The method is based on sorting the input vectors and traversing the (implicitly) resulting partially ordered matrix of products in a way that not all entries need to be evaluated, while only keeping track of the so-called cover of maximal elements. FILL1 already includes some more sophisticated checks to keep the cover small and thereby reducing the overhead per iteration. Unfortunately, although we observed that the FILL1 method requires between $O(n \log(n))$ and $O(n^2)$ iterations in practice, this per-iteration overhead results in a worst-case cost of $\log(n)$ per iteration, yielding an overall runtime in practice between $O(n \log(n) \log(n))$ and $O(n^2 \log(n))$. As the authors state, this overhead is due to the expense of storing the cover, which can be implemented *e.g.*, using a binary heap (recommended by the authors and used in this reimplementation). Additionally, due to the fairly sophisticated datastructures needed for this algorithm it had a higher runtime constant than the other methods presented here, and furthermore we saw no means to vectorize it to improve the efficiency. For this reason, it is not truly fair to compare the raw runtimes to the other vectorized algorithms (and it is not likely that this Python reimplementation is as efficient as the original version, which Bussieck et al., 1994 implemented in ANSI-C); however, comparing a non-vectorized implementation of the naive $O(n^2)$ approach with its vectorized counterpart gives an estimated $\approx 5\times$ speedup from vectorization, suggesting that it is not substantially faster than the naive approach on these problems (it should be noted that whereas the methods presented here have tight runtime bounds but produce approximate results, the FILL1 algorithm is exact, but its runtime depends on the data processed). During investigation of these runtimes, we found that on the given problems, the proposed average case of $O(n \log(n))$ iterations was rarely reached. A reason might be an unrecognized violation of the assumptions of the theory behind this theoretical average runtime in how the input vectors were generated.

In contrast to the exact method from Bussieck et al. (1994), the herein proposed approximate procedure are faster whenever the input vectors are at least 128 elements long (shorter vectors are most efficiently processed with the naive approach). The null space projection method is the fastest method presented here (because it can use a lower $p^*_{\max}$), although the higher density of p^* values it uses (and thus, additional FFTs) make the runtimes nearly identical for both approximation methods.

3.6 Demonstration on Hidden Markov Model With Toeplitz Transition Matrix

One example that profits from fast max-convolution of non-negative vectors is computing the Viterbi path using a hidden Markov model (HMM) (*i.e.*, the *maximum a posteriori* states) with an additive transition function satisfying $\Pr(X_{i+1} = a | X_i = b) \propto \delta(a - b)$ for some arbitrary function δ (δ can be represented as a table, because we are considering all possible discrete functions). This additivity constraint is equivalent to the transition matrix being a "Toeplitz matrix": the transition matrix $T_{a,b} = \Pr(X_{i+1} = a | X_i = b)$ is a Toeplitz matrix when all cells diagonal from each other (to the upper left and lower right) have identical values (*i.e.*, $\forall a, \forall b,\ T_{a,b} = T_{a+1,b+1}$). Because of the Markov property of the chain, we only need to max-marginalize out the latent variable at time i to compute the distribution for the next latent variable X_{i+1} and all observed values of the data variables $D_0 \dots D_{i+1}$. This procedure, called the Viterbi algorithm, is continued inductively:

$$\max_{x_0, x_1, \dots x_{i-1}} \Pr(D_0, D_1, \dots D_{i-1}, X_0 = x_0, X_1 = x_1, \dots, X_i = x_i) =$$

$$\max_{x_{i-1}} \max_{x_0, x_1, \dots x_{i-2}} \Pr(D_0, D_1, \dots D_{i-2}, X_0 = x_0, X_1 = x_1, \dots, X_{i-1} = x_{i-1}) \Pr(D_{i-1} | X_{i-1} = x_{i-1}) \Pr(X_i = x_i | X_{i-1} = x_{i-1})$$

and continuing by exploiting the self-similarity on a smaller problem to proceed inductively with variable $fromLeft$, revealing a max-convolution (for this specialized HMM with additive transitions):

$$\max_{x_0, x_1, \dots x_{i-1}} \Pr(D_0, D_1, \dots D_{i-1}, X_0 = x_0, X_1 = x_1, \dots, X_i = x_i) =$$
$$\max_{x_{i-1}} fromLeft[i-1] \Pr(D_{i-1} | X_{i-1} = x_{i-1}) \delta[x_i - x_{i-1}] =$$
$$(fromLeft[i-1]\ likelihood[D_{i-1}])\ *_{\max}\ \delta[x_i - x_{i-1}].$$

After computing this left-to-right pass (which consisted of $n-1$ max-convolutions and vector multiplications), we can find the *maximum a posteriori* configuration of the latent variables $X_0, \dots X_{n-1} = x^*_0, \dots x^*_{n-1}$ backtracking right-to-left, which can be done by finding the variable value x_i that maximizes $fromLeft[i][x_i] \times \delta[x^*_{i+1} - x_i]$ (thus defining x^*_i and enabling induction on the right-to-left pass). The right-to-left pass thus requires $O(nk)$ steps (**Algorithm 6**). Note that the full max-marginal distributions on each latent variable X_i can be computed via a small modification, which would perform a more complex right-to-left pass that is nearly identical to the left-to-right pass, but which performs subtraction instead of addition (*i.e.*, by reversing the vector representation of the PMF of the subtracted argument before it is max-convolved; Serang, 2014).

We apply this HMM with additive transition probabilities to a data analysis problem from economics. It is known for example, that the current figures of unemployment in a country have (among others) impact on prices of commodities like oil. If one could predict unemployment figures before the usual weekly or monthly release by the responsible government bureaus, this would lead to an information advantage and an opportunity for short-term arbitrage. The close relation of economic indicators like market prices and stock market indices (especially of indices combining several stocks of different industries) to unemployment statistics can be used to tackle this problem.

In the following demonstration of our method, we create a simple HMM with additive transitions and use it to infer the *maximum a posteriori* unemployment statistics given past history (*i.e.*, how often unemployment is low and high, as well as how often unemployment goes down or up in a

Algorithm 6 Viterbi for models with additive transitions, which accepts the length k vector *prior*, a list of n binned observations *data*, a $a \times k$ matrix of likelihoods (where a is the number of bins used to discretize the data) *likelihoods*, and a length $2k-1$ vector δ that describes the transition probabilities. The algorithm returns a Viterbi path of length n, where each element in the path is a valid state $\in \{0, 1, \ldots k-1\}$.

procedure ViterbiForAdditiveTransitions($prior, data, likelihood, \delta$)
  $fromLeft[0] \leftarrow prior$
  for $i = 0$ to $n-2$ **do**
    $fromLeft[i] \leftarrow fromLeft[i] \times likelihood[data[i]]$
    $fromLeft[i+1] \leftarrow fromLeft[i] *_{\max} \delta$
  end for
  $fromLeft[n] \leftarrow fromLeft[n] \times likelihood[data[n]]$

  $path[n-1] \leftarrow \text{argmax}_j\, fromLeft[n-1][j]$
  for $i = n-2$ to 0 **do**
    $maxProdPosterior \leftarrow -1$
    $argmaxProdPosterior \leftarrow -1$
    for $l = k$ to 1 **do**
      $currProdPosterior \leftarrow fromLeft[i] \times \delta[l - path[i+1]]$
      if $currProdPosterior > maxProdPosterior$ **then**
        $maxProdPosterior \leftarrow currProdPosterior$
        $argmaxProdPosterior \leftarrow l$
      end if
    end for
    $path[i] \leftarrow argmaxProdPosterior$
  end for
  return *path*
end procedure

short amount of time) and current stock market prices (the observed data). We discretized random variables for the observed data (S&P 500, adjusted closing prices ; retrieved from YAHOO! historical stock prices: `http://data.bls.gov/cgi-bin/surveymost?blsseriesCUUR0000SA0`), and "latent" variables (unemployment insurance claims, seasonally adjusted, were retrieved from the U.S. Department of Labor: `https://www.oui.doleta.gov/unemploy/claims.asp`). Stock prices were additionally inflation adjusted by (*i.e.*, divided by) the consumer price index (CPI) (retrieved from the U.S. Bureau of Labor Statistics: `https://finance.yahoo.com/q?s=^GSPC`). The intersection of both "latent" and observed data was available weekly from week 4 in 1967 to week 52 in 2014, resulting in 2500 data points for each type of variable.

To investigate the influence of overfitting, we partition the data in two parts, before June 2005 and after June 2005, so that we are effectively training on $\frac{2000 \times 100}{2500} = 80\%$ of the data points, and then demonstrate the Viterbi path on the entirety of the data (both the 80% training data and the 20% of the data withheld from empirical parameter estimation). Unemployment insurance claims were discretized into 512 and stock prices were discretized into 128 bins. Simple empirical models of the prior distribution for unemployment, the likelihood of unemployment given stock prices, and the transition probability of unemployment were built as follows: The initial or prior distribution for unemployment claims at $i = 0$ was calculated by marginalizing the time series of training data for the claims (*i.e.*, counting the number of times any particular unemployment value was reached over all possible bins). Our transition function (the conditional probability $\Pr(X_{i+1}|X_i)$) similarly counts the number of times each possible change $X_{i+1} - X_i \in \{-511, -510, \ldots 511\}$ occurred over all available time points. Interestingly, the resulting transition distribution roughly resembles a Gaussian (but is not an exact Gaussian). This underscores a great quality of working with discrete distributions: while continuous distributions may have closed-forms for max-convolution (which can be computed quickly), discrete distributions have the distinct advantage that they can accurately approximate any smooth distribution. Lastly, the likelihoods of observing a stock price given the unemployment at the same time were trained using an empirical joint distribution (essentially a heatmap), which is displayed in **Figure 7**.

We compute the Viterbi path two times: First we use naive, exact max-convolution, which requires a total of $O(nk^2)$ steps. Second, we use fast numerical max-convolution, which requires $O(n\, k \log(k) \log(\log(k))$ steps. Despite the simplicity of the model, the exact Viterbi path (computed via exact max-convolution) is highly informative for predicting the value of unemployment, even for the 20% of the data that were not used to estimate the empirical prior, likelihood, and transition distributions. Also, the numerical max-convolution method is nearly identical to the exact max-convolution method at every index (**Figure 8**). Even with a fairly rough discretization (*i.e.*, $k = 512$), the fast numerical method (via the original, simpler piecewise algorithm with $p^*_{\max} = 8192$) used 141.4 seconds compared to the 292.3 seconds required by the naive approach. The higher-precision projection algorithm (which uses a smaller $p^*_{\max}$, but calls FFT twice for each power of two p^*) computes nearly identical result using $p^*_{\max} = 64$ for this problem in 136.6 seconds. The speedup of both fast numerical algorithms relative to the naive quadratic max-convolution algorithm will increase dramatically as k is increased.

4. Discussion

Both piecewise numerical max-convolution methods are highly accurate in practice and achieve a substantial speedup over both the naive approach and the approach proposed by Bussieck et al. (1994). This is particularly true for large problems: For the original piecewise method presented here, the $\log_2(\log_{1+\tau^{\frac{1}{4}}}(k))$ multiplier may never be small, but it grows so slowly with k that it will be < 18 even when k is on the same order of magnitude as the number of particles in the observable universe. This means that, for all practical purposes, the method behaves asymptotically as a slightly slower $O(k \log_2(k))$ method, which means the speedup relative to the naive method

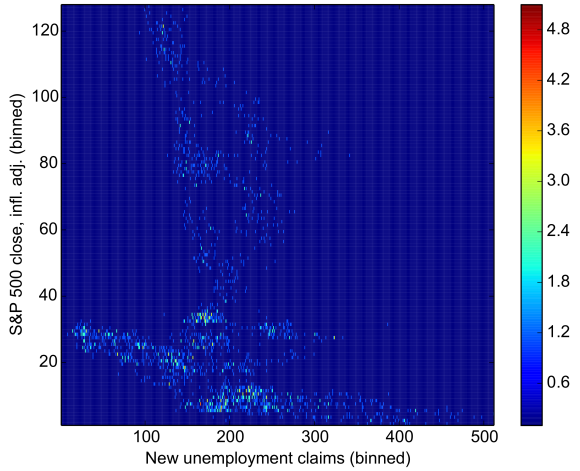


Figure 7: **Heatmap for trained likelihood matrix.** This heatmap depicts a joint empirical distribution between the S&P 500 index and new unemployment claims, which share a tenuous inverse relationship. Given D_i, the discretized stock index value at time i, row D_i contains the likelihood table $\Pr(D_i|X_i)$, which is denoted $likelihood[data[i]]$ in the code.

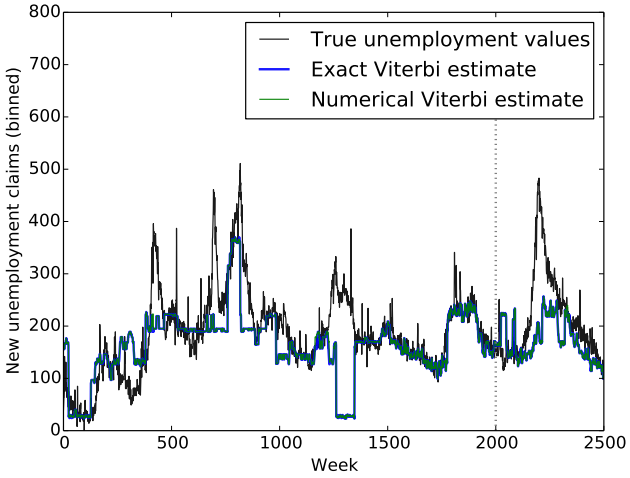


Figure 8: **Viterbi analysis of employment given stock index values.** The Viterbi path corresponding to the *maximum a posteriori* prediction of the number of new unemployment insurance claims is produced for a model where the state transition probabilities are additive. The exact Viterbi estimate tracks well with the true unemployment values. Training parameters were taken from only the true unemployment data to the left of the vertical dotted line; however, the Viterbi paths to the right of the dotted line (where unemployment data were withheld from the likelihood, prior, and transition parameters) also track well with the true unemployment statistics. The Viterbi path computed with fast numerical max-convolution (via the null space projection piecewise approach) is nearly identical to the result computed with the slower exact approach. Note that for this problem, the original, simpler piecewise approach also produces very similar results.

becomes more pronounced as k becomes large. For the second method presented (the null space projection), the runtime for a given relative error bound will be in $O(k \log_2(k))$. In practice, both methods have similar runtime on moderate or large problems.

The basic motivation of the first approach described—*i.e.*, the idea of approximating the Chebyshev norm with the largest p^*-norm that can be computed accurately, and then convolving according to this norm using FFT—also suggests further possible avenues of research. For instance, it may be possible to compute a single FFT (rather than an FFT at each of several contours) on a more precise implementation of complex numbers. Such an implementation of complex values could store not only the real and imaginary components, but also other much smaller real and imaginary components that have been accumulated through + operations, even those which have small enough magnitudes that they are dwarfed by other summands. With such an approach it would be possible to numerically approximate the max-convolution result in the same overall runtime as long as only a bounded "history" of such summands was recorded (*i.e.*, if the top few magnitude summands—whether that be the top 7 or the top $\log_2(\log_{1+\tau^{\frac{1}{4}}}(k))$—was stored and operated on). In a similar vein, it would be interesting to investigate the utility of complex values that use rational numbers (rather than fixed-precision floating point values), which will be highly precise, but will increase in precision (and therefore, computational complexity of each arithmetic operation) as the dynamic range between the smallest and largest nonzero values in L and R increases (because taking L' to a large power p^* may produce a very small value). Other simpler improvements could include optimizing the error vs. runtime trade-off between the log-base of the contour search: the method currently searches $\log_2(p^*_{\max})$ contours, but a smaller or larger log-base could be used in order to optimize the trade-off between error and runtime.

It is likely that the best trade-off will occur by performing the fast p^*-norm convolution with a number type that sums values over vast dynamic ranges by appending them in a short (*i.e.*, bounded or constant size) list or tree and sums values within the same dynamic range by querying the list or tree and then summing in at the appropriate magnitude. This is reminiscent of the fast multipole algorithm (Rokhlin, 1985). This would permit the method to use a single large p^* rather than a piecewise approach, by moving the complexity into operations on a single number rather than by performing multiple FFTs with simple floating-point numbers.

The basic motivation of the second approach described—*i.e.*, using the *sequence* of p^*-norms (each computed via FFT) to estimate the maximum value—generalizes the p^*-norm fast convolution numerical approach into an interesting theoretical problem in its own right: given an oracle that delivers a small number of norms (the number of norms retrieved must be $c \in o(k)$ to significantly outperform the naive quadratic approach) about each vector $u^{(m)}$, amalgamate these norms in an efficient manner to estimate the maximum value in each $u^{(m)}$. This method may be applicable to other problems, such as databases where the maximum values of some combinatorial operation (in this case the *maximum a posteriori* distribution of the sum of two random variables $X+Y$) is desired but where caching all possible queries and their maxima would be time or space prohibitive. In a manner reminiscent of how we employ FFT, it may be possible to retrieve moments of the result of some combinatoric combination between distributions on the fly, and then use these moments to approximate true maximum (or, in general, other sought quantities describing the distribution of interest).

In practice, the worst-case relative error of our quadratic approximation is quite low. For example, when $p^* = 8$ is stable, then the relative error is less than 2.3%, regardless of the lengths of the vectors being max-convolved. In contrast, the worst-case relative error using the original piecewise method would be $\leq k^{\frac{1}{16}} - 1$, where k is the length of the max-convolution result (when $n = 1024$, the relative error of the original piecewise method would be $\approx 54\%$).

Of course, the use of the null space projection method is predicated on the existence of at least four sequential p^* points, but it would be possible to use finer spacing between p^* values (*e.g.*, $p^* \in (1, 1.01, 1.02, 1.03)$ to guarantee that this will essentially be the case as long as FFT (*i.e.*, $p^* = 1$) is stable. But more generally, the problem of estimating extrema from p^*-norms (or,

equivalently, from the p^*-th roots of the p^*-th moments of a distribution with bounded support), will undoubtedly permit many more possible approaches that we have not yet considered. One that would be compelling is to relate the Fourier transform of the sequential moments to the maximum value in the distribution; such an approach could permit all stable p^* at any index m to be used to efficiently approximate the maximum value (by computing the FFT of the sequence of norms). Such new adaptations of the method could permit low worst-case error without any noticable runtime increase.

4.1 Multidimensional Numerical Max-Convolution

The fast numerical piecewise method for max-convolution (and the affine piecewise modification) are both applicable to matrices as well as vectors (and, most generally, to tensors of any dimension). This is because the p^*-norm (as well as the derived error bounds as an approximation of the Chebyshev norm) can likewise approximate the maximum element in the tensor $u^{(m_1,m_2,\ldots)}$ generated to find the max-convolution result at index $m_1, m_2, \ldots$ of a multidimensional problem, because the sum

$$\sum_{i_1,i_2,\ldots} \left(u^{(m_1,m_2,\ldots)}_{i_1,i_2,\ldots}\right)^{p^*}$$

computed by convolution corresponds to the Frobenius norm (*i.e.*, the "entrywise norm") of the tensor, and after taking the result of the sum to the power $\frac{1}{p^*}$, will converge to the maximum value in the tensor (if p^* is large enough).

This means that the fast numerical approximation, including the affine piecewise modification, can be used without modification by invoking standard multidimensional convolution (*i.e.*, $*$). Matrix (and, in general, tensor) convolution is likewise possible for any dimension via the row-column algorithm, which transforms the FFT of a matrix into sequential FFTs on each row and column. The accompanying Python code demonstrates the fast numerical max-convolution method on matrices, and the code can be run on tensors of any dimension (without requiring any modification).

The speedup of FFT tensor convolution (relative to naive convolution) becomes considerably higher as the dimension of the tensors increases; for this reason, the speedup of fast numerical max-convolution becomes even more pronounced as the dimension increases. For a tensor of dimension d and width k (*i.e.*, where the index bounds of every dimension are $\in \{0, 1, \ldots k-1\}$), the cost of naive max-convolution will be in $O(k^{2d})$, whereas the cost of numerical max-convolution is $O(k^d \log_2(k))$ (ignoring the $\log_2(\log_{1+\tau^{\frac{1}{4}}}(k)) \leq 18$ multiplier), meaning that there is an $O(\frac{k^d}{d \log_2(k)})$ speedup from the numerical approach. Examples of such tensor problems include graph theory, where adjacency matrix representation can be used to describe respective distances between nodes in a network.

As a concrete example, the demonstration Python code computes the max-convolution between two 256×256 matrices. The naive method required 494 seconds, but the numerical result with the original piecewise method was computed in 3.18 seconds (yielding a maximum absolute error of 0.0173 and a maximum relative error of 0.0511) and the numerical result with the null space projection method was computed in 3.99 seconds (using $p^*_{\max} = 512$, which corresponds to a relative error of $< 0.1\%$ in the top contour, yielding a maximum absolute error of 0.0141 and a maximum relative error of 0.0227) and in 3.05 seconds (using $p^*_{\max} = 64$, which corresponds to a relative error of $< 2.5\%$ in the top contour, yielding a maximum absolute error of 0.0667 and a maximum relative error of 0.067). Not only does the speedup of the proposed methods relative to naive max-convolution increase significantly as the dimension of the tensor is increased, no other faster-than-naive algorithms exist for max-convolution of matrices or tensors.

Multidimensional max-convolution can likewise be applied to hidden Markov models with additive transitions over multidimensional variables (*e.g.*, allowing the latent variable to be a two-dimensional joint distribution of American and German unemployment with a two-dimensional joint transition probability).

4.2 Max-Deconvolution

The same p^*-norm approximation can also be applied to the problem of max-deconvolution (*i.e.*, solving $M = L *_{\max} R$ for R when given M and L). This can be accomplished by computing the ratio of $FFT(M^{p^*})$ to $FFT(L^{p^*})$ (assuming L has already been properly zero-padded), and then computing the inverse FFT of the result to approximate R^{p^*}; however, it should be noted that deconvolution methods are typically less stable than the corresponding convolution methods, computing a ratio is less stable than computing a product (particularly when the denominator is close to zero).

4.3 Amortized Argument for Low MSE of the Affine Piecewise Method

Although the largest absolute error of the affine piecewise method is the same as the largest absolute error of the original piecewise method, the mean squared error (MSE) of the affine piecewise method will be lower than the square of the worst-case absolute error.

To achieve the worst-case absolute error for a given contour the affine correction must be negligible; therefore, there must be two nearly vertical points on the scatter plot of $\|u^{(m_1)}\|_\infty$ vs. $\|u^{(m_1)}\|_{p^*}$, which are both extremes of the bounding envelope from **Figure 3**. Thus, there must exist two different indices m_1 and m_2 with vectors where $\|u^{(m_1)}\|_{p^*} \approx \|u^{(m_1)}\|_\infty$ and where

$$\|u^{(m_2)}\|_{p^*} \approx \|u^{(m_2)}\|_\infty k_{m_2}^{\frac{1}{p^*}}$$

(creating two vertical points on the scatter plot, and forcing that both cannot simultaneously be corrected by a single affine mapping). In order to do this, it is required to have $u^{(m_1)}$ filled with a single nonzero value and for the remaining elements of $u^{(m_1)}$ to equal zero. Conversely, $u^{(m_2)}$ must be filled entirely with large, nonzero values (the largest values possible that would still use the same contour p^*). Together, these two arguments place strong constraints on the vectors L' and R' (and transitively, also constrains the unscaled vectors L and R): On one hand, filling $u^{(m_1)}$ with $k_{m_1} - 1$ zeros requires that $k_{m_1} - 1$ elements from either L or R must be zero (because at least one factor must be zero to achieve a product of zero). On the other hand, filling $u^{(m_2)}$ with all large-value nonzeros requires that k_{m_2} elements of *both* L and R are nonzero. Together, these requirements stipulate that both $k_{m_1} - 1 + k_{m_2} \leq k$, because entries of L and R cannot simultaneously be zero and nonzero. Therefore, in order to have many such vertical points, constrains the lengths of the $u^{(m_1)}, u^{(m_2)}, u^{(m_3)}, \ldots$ vectors corresponding to those points. While the worst-case absolute error bound presumes that an individual vector $u^{(m)}$ may have length k, this will not be possible for many vectors corresponding to vertical points on the scatter plot. For this reason, the MSE will be significantly lower than the square of the worst-case absolute error, because making a high affine-corrected absolute error on one index necessitates that the absolute errors at another index cannot be the worst-case absolute error (if the sizes of L and R are fixed).

5. Availability

Code for exact max-convolution and the fast numerical method (which includes both $\|\cdot\|_{p^*}$ and null space projection methods) is implemented in Python and available at `https://bitbucket.org/orserang/fast-numerical-max-convolution`. All included code works for `numpy` arrays of any dimension, *i.e.*, tensors).

Acknowledgments

We would like to thank Mattias Frånberg, Knut Reinert, and Oliver Kohlbacher for the interesting discussions and suggestions. J.P. acknowledges funding from BMBF (Center for Integrative Bioinformatics, grant no. 031A367). O.S. acknowledges generous start-up funds from Freie Universität Berlin and the Leibniz-Institute for Freshwater Ecology and Inland Fisheries.

Appendix A. Details on the Projection Method

This appendix establishes a closed-form equation for finding the maximum via the projection method as it is used in the implementation of the algorithm and proves/conjectures an upper/lower bound of its relative error.

A.1 Closed-Form Projection Method for $r = 2$

In general, the computation of both the null space spanning vector $(\gamma_0, \gamma_1, \ldots \gamma_r)$ and of machine-precision approximations for the roots of the polynomial $\gamma_0 + \gamma_1 x + \gamma_2 x^2 + \cdots + \gamma_r x^r$ (which can be approximated by constructing a matrix with that characteristic polynomial and performing eigen-decomposition; Horn and Johnson, 1999) are both in $O(r^3)$ for each index m in the result; however, by using a small $r = 2$, we can compute a closed form solution of both the null space spanning vector and of the resulting quadratic roots. This enables faster exploitation of the curve of norms for estimating the maximum value of $u^{(m)}$ (although it doesn't achieve the high accuracy possible with a much larger $r \approx e$). This is equivalent to approximating $\|u^{(m)}\|_{p^*}^{p^*} \approx h_1 \alpha_1^{p^*} + h_2 \alpha_2^{p^*}$, where $h_1 + h_2 = k_m = len(u^{(m)})$.

In this case, the single spanning vector of the null space of

$$\begin{bmatrix} \|u^{(m)}\|_{p^*}^{p^*} & \|u^{(m)}\|_{2p^*}^{2p^*} & \|u^{(m)}\|_{3p^*}^{3p^*} \\ \|u^{(m)}\|_{2p^*}^{2p^*} & \|u^{(m)}\|_{3p^*}^{3p^*} & \|u^{(m)}\|_{4p^*}^{4p^*} \end{bmatrix}$$

will be

$$\begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \gamma_2 \end{bmatrix} = \begin{bmatrix} \|u^{(m)}\|_{2p^*}^{2p^*} \|u^{(m)}\|_{4p^*}^{4p^*} - \left(\|u^{(m)}\|_{3p^*}^{3p^*}\right)^2 \\ \|u^{(m)}\|_{2p^*}^{2p^*} \|u^{(m)}\|_{3p^*}^{3p^*} - \|u^{(m)}\|_{p^*}^{p^*} \|u^{(m)}\|_{4p^*}^{4p^*} \\ \|u^{(m)}\|_{p^*}^{p^*} \|u^{(m)}\|_{3p^*}^{3p^*} - \left(\|u^{(m)}\|_{2p^*}^{2p^*}\right)^2 \end{bmatrix}$$

and thus $\hat{\alpha} \approx \|u^{(m)}\|_\infty$ can be computed by using the quadratic formula to solve $\gamma_0 + \gamma_1 x + \gamma_2 x^2 = 0$ for x, and computing $\hat{\alpha}$ using the maximum of those zeros: $\hat{\alpha} = x_{\max}^{\frac{1}{p^*}}$. When the quadratic is not well defined, then this indicates that the number of unique elements in $u^{(m)}$ is less than 2, and thus cannot be projected uniquely (*i.e.*, $e_m < r$); in this case, the closed-form linear solution can be used rather than a closed-form quadratic solution:

$$\hat{\alpha} = \left(\frac{\|u^{(m)}\|_{4p^*}^{4p^*}}{\|u^{(m)}\|_{3p^*}^{3p^*}}\right)^{\frac{1}{p^*}}.$$

When the closed-form linear solution is not numerically stable (due to division by a value close to zero), then the p^*-norm approximation can likewise be used.

A.2 Adapted Piecewise Algorithm Using Interleaved p^* Points

Because the norms must have evenly spaced p^* values in order to use the projection method described above, the exponential sequence of p^* values used in the original piecewise algorithm will not contain four evenly spaced points (which are necessary to solve the quadratic formulation, *i.e.*, $r = 2$). One possible solution would be to take the maximal stable value of p^* for any index (which will be a power of two found using the original piecewise method), and then also computing norms (via the FFT, as before) for $p^* - 3\delta, p^* - 2\delta, p^* - \delta, p^*$; however, this will result in a $4\times$ slowdown in the

algorithm, because for every p^*-norm computed via FFT before, now four must be computed. An alternative approach reuses existing values in the 2^i sequence of p^*: for p^* sufficiently large, then the exponential sequence is guaranteed to include these stable p^* values: $\frac{p^*}{4}, \frac{p^*}{2}, p^*$. By considering $\frac{3p^*}{4}$ in p^* candidates, then we can be guaranteed to have four evenly spaced and stable p^* values. This can be achieved easily by noting that

$$\frac{3p^*}{4} = \frac{\frac{p^*}{2} + p^*}{2},$$

meaning that we can insert all possible necessary p^* values for evenly spaced sequences of length four by first computing the exponential sequence of p^* values and then inserting the averages between every pair of adjacent powers of two (and inserting them in a way that maintains the sorted order): $1, 2, 4, 8, 16, \ldots$ becomes $1, 1.5, 2, 3, 4, 6, 8, 12, 16, \ldots$. Thus, if (for some index m) 16 is the highest stable p^* that is a power of two (*i.e.*, the p^* value that would be used by the original piecewise algorithm), then we are guaranteed to use the evenly spaced sequence $4, 8, 12, 16$. By interleaving the powers of two with the averages from the following powers of two, we reduce the number of FFTs to $2\times$ that used by the original piecewise algorithm. For small values of r (such as the $r = 2$ used here), the estimation of the maximum from each sequence of four norms is in $O(4k)$, meaning the total time will still be $k\log(k)\log(\log(k) + 4k \in O(k\log(k)\log(\log(k)))$, which is the same as before. Because the spacing in this formulation is $\frac{p^*}{4}$, and given the maximal root of the quadratic polynomial $\gamma(x_{\max}) = 0$, then $\hat{\alpha} = x_{\max}^{\frac{4}{p^*}}$ (taking the maximal root $x_{\max}$ to the power $\frac{4}{p^*}$ instead of $\frac{1}{p^*}$, which had been the spacing used in the description of the projection method). The null space projection method is shown in **Algorithm 5**.

Algorithm 7 Linear projection of the maximum, using previously computed values est_3, est_4 for two p^* with a difference of *spacing* (estimates given in ascending order of their corresponding p's used). The naming of the variables follows the scheme $est_i = \|u^{(m)}\|_{\frac{i}{4}maxP}$. To prevent numeric instabilities, the algorithm checks for division by zero within a tolerance $\tau_{\text{Div}} = 10^{-10}$ (again, a conservative estimate of the machine precision). The return value is a new estimate of the real maximum.

```
1: procedure MAXLIN(est_3, est_4, spacing)
2:     if |est_3| > τ_Div then
3:         result ← est_4 / est_3
4:     else
5:         result ← est_4
6:     end if
7:     return result^(1.0/spacing)
8: end procedure
```

A.3 Accuracy of the $r = 2$ Projection-Based Method

The full closed-form of the quadratic roots used above (which solve the projection when $r = 2$) will be

Algorithm 8 Quadratic projection of the maximum, using previously computed estimates $est_1, est_2, est_3, est_4$ for four equally spaced p in steps of *spacing* (estimates given in ascending order of their corresponding p's used). The naming of the variables follows the scheme $est_i = \|u^{(m)}\|_{\frac{i}{4}maxP}$. To prevent numeric instabilities, the algorithm checks for division by zero within a tolerance $\tau_{\text{Div}} = 10^{-10}$. The return value is a new estimate of the real maximum.

```
1: procedure MAXQUAD(est_1, est_2, est_3, est_4, spacing)
2:     γ_2 ← est_1 * est_3 − est_2^2
3:     γ_1 ← est_2 * est_3 − est_1 * est_4
4:     γ_0 ← est_2 * est_4 − est_3^2
5:     preRootValue ← γ_1^2 − 4 * γ_2 * γ_0
6:     stableQuadratic ← (γ_0 > τ_Div) & (preRootValue >= 0.0)
7:     if stableQuadratic then
8:         result ← (−γ_1 + √preRootValue)/(2 * γ_2)
9:     else                                          ▷ Resort to linear projection
10:        result ← maxLin(est_3, est_4)
11:    end if
12:    return result^(1.0/spacing)
13: end procedure
```

$$
\begin{aligned}
\hat{\alpha} &= \max\left(\left(\frac{-\gamma_1 \pm \sqrt{\gamma_1^2 - 4\gamma_2\gamma_0}}{2\gamma_2}\right)^{\frac{1}{p^*}}\right) \\
&= \max\Big(\Big(\Big(\|u^{(m)}\|_{2p^*}^{2p^*}\|u^{(m)}\|_{3p^*}^{3p^*} - \|u^{(m)}\|_{1p^*}^{1p^*}\|u^{(m)}\|_{4p^*}^{4p^*} \\
&\quad \pm \Big((\|u^{(m)}\|_{2p^*}^{2p^*}\|u^{(m)}\|_{3p^*}^{3p^*} - \|u^{(m)}\|_{1p^*}^{1p^*}\|u^{(m)}\|_{4p^*}^{4p^*})^2 \\
&\quad - 4\ (\|u^{(m)}\|_{1p^*}^{1p^*}\|u^{(m)}\|_{3p^*}^{3p^*} - \|u^{(m)}\|_{2p^*}^{2p^*{}^2})(\|u^{(m)}\|_{2p^*}^{2p^*}\|u^{(m)}\|_{4p^*}^{4p^*} - \|u^{(m)}\|_{3p^*}^{3p^*{}^2})\Big)^{0.5}\Big) \\
&\quad \div 2(\|u^{(m)}\|_{2p^*}^{2p^*{}^2} - \|u^{(m)}\|_{1p^*}^{1p^*}\|u^{(m)}\|_{3p^*}^{3p^*})\Big)^{\frac{1}{p^*}}\Big) \\
&= \max\Big(\Big(\|u^{(m)}\|_{\infty}^{p^*}\Big(\|v^{(m)}\|_{2p^*}^{2p^*}\|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{4p^*}^{4p^*} \\
&\quad \pm \Big((\|v^{(m)}\|_{2p^*}^{2p^*}\|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{4p^*}^{4p^*})^2 \\
&\quad - 4\ (\|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{2p^*}^{2p^*{}^2})(\|v^{(m)}\|_{2p^*}^{2p^*}\|v^{(m)}\|_{4p^*}^{4p^*} - \|v^{(m)}\|_{3p^*}^{3p^*{}^2})\Big)^{0.5}\Big) \\
&\quad \div 2(\|v^{(m)}\|_{2p^*}^{2p^*{}^2} - \|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{3p^*}^{3p^*})\Big)^{\frac{1}{p^*}}\Big) \\
&= \|u^{(m)}\|_{\infty}\max\Big(\Big(\Big(\|v^{(m)}\|_{2p^*}^{2p^*}\|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{4p^*}^{4p^*} \\
&\quad \pm \Big((\|v^{(m)}\|_{2p^*}^{2p^*}\|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{4p^*}^{4p^*})^2 \\
&\quad - 4\ (\|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{2p^*}^{2p^*{}^2})(\|v^{(m)}\|_{2p^*}^{2p^*}\|v^{(m)}\|_{4p^*}^{4p^*} - \|v^{(m)}\|_{3p^*}^{3p^*{}^2})\Big)^{0.5}\Big) \\
&\quad \div 2(\|v^{(m)}\|_{2p^*}^{2p^*{}^2} - \|v^{(m)}\|_{1p^*}^{1p^*}\|v^{(m)}\|_{3p^*}^{3p^*})\Big)^{\frac{1}{p^*}}\Big)
\end{aligned}
$$

where $p^* = \frac{maxP}{4}$ in the pseudocode (*i.e.*, the maximum numerically stable p^* used by the piecewise algorithm at that index). Note that $\|u^{(m)}\|_{\infty}^{p^*}$ can be factored out because the exponents in every term

in the numerator will be $5p^*$ (*i.e.*, $10p^*$ in the square root). Similarly the terms in the denominator each contain $\|u^{(m)}\|_\infty^{4p^*}$. Factoring out the maximum value is then the same as operating on scaled vectors v (instead of u) with the maximum entry being 1, and at least one element of value 1.

Furthermore, the denominator $2\gamma_2 \geq 0$; even though the terms summed to compute γ_2 are not exclusively nonnegative, symmetry can be used to demonstrate that every negative term is outweighed by a unique corresponding term:

$$\begin{aligned}
\gamma_2 &= \|v^{(m)}\|_1\|v^{(m)}\|_3^3 - \left(\|u^{(m)}\|_{2p^*}^{2p^*}\right)^2 \\
&= \left(\sum_i v_i^{(m)}\right)\left(\sum_i {v_i^{(m)}}^3\right) - \left(\sum_i {v_i^{(m)}}^2\right)^2 \\
&= \sum_{i,j} v_i^{(m)} {v_j^{(m)}}^3 - \sum_{i,j} {v_i^{(m)}}^2 {v_j^{(m)}}^2 \\
&= \sum_{i,j} v_i^{(m)} {v_j^{(m)}}^2 \left(v_j^{(m)} - v_i^{(m)}\right) \\
&= \sum_i v_i^{(m)} {v_i^{(m)}}^2 \left(v_i^{(m)} - v_i^{(m)}\right) + \sum_{i<j} v_i^{(m)} {v_j^{(m)}}^2 \left(v_j^{(m)} - v_i^{(m)}\right) + v_j^{(m)} {v_i^{(m)}}^2 \left(v_i^{(m)} - v_j^{(m)}\right) \\
&= 0 + \sum_{i<j} v_i^{(m)} v_j^{(m)} \left(v_j^{(m)} - v_i^{(m)}\right)\left(v_j^{(m)} - v_i^{(m)}\right) \\
&= \sum_{i<j} v_i^{(m)} v_j^{(m)} \left(v_j^{(m)} - v_i^{(m)}\right)^2 \\
&\geq 0.
\end{aligned}$$

Thus, for well-defined problems (*i.e.*, when $\gamma_2 \neq 0$), the denominator $2\gamma_2 > 0$, and therefore, the maximum root of the quadratic polynomial will correspond to the term that adds (rather than subtracts) the square root term:

$$\begin{aligned}
\hat{\alpha} = \|u^{(m)}\|_\infty \Big(&\Big(\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{4p^*}^{4p^*} \\
&+ \Big((\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{4p^*}^{4p^*})^2 \\
&- 4 \ (\|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{2p^*}^{2p^{*2}}) (\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{4p^*}^{4p^*} - \|v^{(m)}\|_{3p^*}^{3p^{*2}}) \Big)^{0.5} \Big) \\
&\div 2(\|v^{(m)}\|_{2p^*}^{2p^{*2}} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{3p^*}^{3p^*}) \Big)^{\frac{1}{p^*}} \Big).
\end{aligned}$$

The relative absolute error is defined as $|\frac{\hat{\alpha} - \|u^{(m)}\|_\infty}{\|u^{(m)}\|_\infty}| = |\frac{\hat{\alpha}}{\|u^{(m)}\|_\infty} - 1|$; therefore, a bound on the relative error of the projection method can be established by bounding

$$\begin{aligned}
s(p^*, k_m) = &\frac{\hat{\alpha}}{\|u^{(m)}\|_\infty} \\
= &\Big(\Big(\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{4p^*}^{4p^*} \\
&+ \Big((\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{4p^*}^{4p^*})^2 \\
&- 4 \ (\|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{2p^*}^{2p^{*2}}) (\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{4p^*}^{4p^*} - \|v^{(m)}\|_{3p^*}^{3p^{*2}}) \Big)^{0.5} \Big) \\
&\div 2(\|v^{(m)}\|_{2p^*}^{2p^{*2}} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{3p^*}^{3p^*}) \Big)^{\frac{1}{p^*}}.
\end{aligned}$$

where the length of the $u^{(m)}$ (respectively $v^{(m)}$) is k_m. Using this reformulation, $s = 1$ indicates a zero-error approximation. This can be rewritten to bound its value before taking to the power $\frac{1}{p^*}$:

$$s(p^*, k_m) = t(p^*, k_m)^{\frac{1}{p^*}}$$

where

$$\begin{aligned}
t(p^*, k_m) = &\Big(\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{4p^*}^{4p^*} \\
&+ \Big((\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{4p^*}^{4p^*})^2 \\
&- 4 \ (\|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{3p^*}^{3p^*} - \|v^{(m)}\|_{2p^*}^{2p^{*2}}) (\|v^{(m)}\|_{2p^*}^{2p^*} \|v^{(m)}\|_{4p^*}^{4p^*} - \|v^{(m)}\|_{3p^*}^{3p^{*2}}) \Big)^{0.5} \Big) \\
&\div 2(\|v^{(m)}\|_{2p^*}^{2p^{*2}} - \|v^{(m)}\|_{1p^*}^{1p^*} \|v^{(m)}\|_{3p^*}^{3p^*}).
\end{aligned}$$

The extreme values of $t(p^*, k_m)$ can be found by minimizing and maximizing over the possible values of $v^{(m)} \in V = \{v : [0,1]^{k_m} : \exists i, v_i = 1, \exists j, v_j \in (0,1)\}$. The final constraint on v_j in (0,1) is because any v containing only one unique value (which must be 1 in this case since dividing by the maximum element in $u^{(m)}$ to compute $v^{(m)}$ has divided the value at that index by itself ($\exists i, v_i = 1$) will lead to instabilities. When values in v are identical to one another, using $r = 1$ yields an exact solution, and thus solving with $r = 2$ is not well-defined because $\gamma_2 = 0$. Because all elements ${v^{(m)}}^{p^*} \in [0,1]$ and $p^* \geq 1$, we can perform a change of variables $v_i^{(m)} = {v^{(m)}}_i^{p^*}$, thereby eliminating references to p^*:

$$\begin{aligned}
t(k_m) \geq \min_{v \in \mathbb{R}^{k_m} : \exists i, v_i = 1, \exists j, v_j \in (0,1)} &\Big(\|v^{(m)}\|_2^2 \|v^{(m)}\|_3^3 - \|v^{(m)}\|_1^1 \|v^{(m)}\|_4^4 \\
&+ \Big((\|v^{(m)}\|_2^2 \|v^{(m)}\|_3^3 - \|v^{(m)}\|_1^1 \|v^{(m)}\|_4^4)^2 \\
&- 4 \ (\|v^{(m)}\|_1^1 \|v^{(m)}\|_3^3 - \|v^{(m)}\|_2^{2^2}) (\|v^{(m)}\|_2^2 \|v^{(m)}\|_4^4 - \|v^{(m)}\|_3^{3^2}) \Big)^{0.5} \Big) \\
&\div 2(\|v^{(m)}\|_2^{2^2} - \|v^{(m)}\|_1^1 \|v^{(m)}\|_3^3).
\end{aligned}$$

k_m	3	4	5	6	7
Minimum	0.935537	0.902161	0.895671	0.880487	0.85343
Maximum	1	1	1	1	1

Table 2: **Exact bounds of $t(k_m)$ for short vectors of length k_m.** This table shows the results of numerical minimization techniques performed on the symbolic closed-form of $t(k_m)$ in Mathematica (using `NMinimize`). All $k_m - 1$ entries (excluding the first that was set to 1.0) were left symbolic and constrained to $[0, 1]$, with restriction that the denominator of $t(k_m)$ was nonzero.

k_m	4	64	1024
Minimum ($k_m - 1$ d.o.f.)	0.90221268	0.74942834	0.81858283
Maximum ($k_m - 1$ d.o.f.)	0.99999986	0.92482416	0.86795636
Minimum (vectors of form $(1, a, b, \dots b)$, 2 d.o.f.)	0.90216688	0.75455478	0.71695386
Maximum (vectors of form $(1, a, b, \dots b)$, 2 d.o.f.)	1.00000000	1.0000000	1.00000000

Table 3: **Bounds via random sampling for vectors different in size and type.** This table shows the minimal and maximal values resulting from the evaluation of $t(k_m)$ on 10^5 randomly generated vectors (uniform distribution in $[0, 1]$). The first part shows the result for vectors of potentially unconstrained composition, besides one (w.l.o.g. the first) being set to 1.0. The values in the second half were obtained based on vectors of (supposedly) worst-case composition (*i.e.*, of form $(1, a, b, \dots b)$).

$$
\begin{aligned}
t(k_m) \leq \max_{v \in \mathbb{R}^{k_m} : \exists i, v_i = 1, \exists j, v_j \in (0,1)} & \Big(\|v^{(m)}\|_2^2 \|v^{(m)}\|_3^3 - \|v^{(m)}\|_1^1 \|v^{(m)}\|_4^4 \\
& + \Big((\|v^{(m)}\|_2^2 \|v^{(m)}\|_3^3 - \|v^{(m)}\|_1^1 \|v^{(m)}\|_4^4)^2 \\
& - 4 \ (\|v^{(m)}\|_1^1 \|v^{(m)}\|_3^3 - \|v^{(m)}\|_2^{2^2}) (\|v^{(m)}\|_2^2 \|v^{(m)}\|_4^4 - \|v^{(m)}\|_3^{3^2}) \Big)^{0.5} \Big) \\
& \div 2(\|v^{(m)}\|_2^{2^2} - \|v^{(m)}\|_1^1 \|v^{(m)}\|_3^3).
\end{aligned}
$$

For small vector lengths, the exact bounds of $t(k_m)$ are shown in **Table 2**. Notice that the upper bound is fixed, but the lower bound grows monotonically smaller as k_m, the length of the vector considered, increases. For larger vectors, Mathematica does not find optima in a matter of hours, and for arbitrary-length vectors, the Karush-Kuhn-Tucker criteria do not easily yield minima or maxima; however, we do observe that all maxima are achieved by vectors that are permutations (order does not influence the result) of $v = (1, 1, \dots 1, b, b, \dots b)$ (again, when only two unique values are found in v, the approximation is exact and thus $\frac{\hat{\alpha}}{\|u^{(m)}\|_\infty} = 1$). Likewise, the minima are achieved by permutations of $v = (1, a, b, b, \dots b)$. For this reason, we perform further empirical estimation of the bound by randomly sampling vectors of the form $(1, v_2, v_3, \dots v_{k_m})$ with $k_m - 1$ degrees of freedom (d.o.f.) and sampling vectors of the form $v = (1, a, b, b, \dots b)$ (with 2 d.o.f.), whose extrema are shown in **Table 3**.

At length 64 we see that due to an extreme value scenario, an unconstrained vector scores slightly lower than a vector holding the worst-case pattern $(1, a, b, \dots b)$, because both forms of sampling approach the true lower bound, but one of the unconstrained $k_m - 2$ d.o.f. is slightly closer.

From these results, we conjecture that t is bounded above ≤ 1 (this is achievable at any length k_m by letting v contain exactly two distinct values). In this manner, we achieve our predicted upper bound of 1 regardless of the length k_m. Likewise, we conjecture that at any k_m (not simply the lengths investigated, where this principle is true), the lower bound is given by vectors of the form $(1, a, b, b, \dots b)$. Qualitatively, this conjecture stems from the fact that since the estimate is perfect when v contains exactly two distinct elements, then the worst-case lower bound when v contains three distinct values will concentrate the points at some value far from the other two distinct values. When four distinct values are permitted, then we conjecture that the optimal choice (for minimizing t) of the fourth value will equal the choice for the third distinct value, since that was already determined to be the best point for deceiving the quadratic approximation. From this conjecture, we can then use the fact that the bounds should only grow more extreme as k_m increases, since $\mathbb{R}^1 \subset \mathbb{R}^2 \subset \cdots$ (*i.e.*, lower-dimensional solutions can always be reached in a higher dimension by setting some of the values to 0). Thus the minimum for any possible vector should be conservatively bounded below on vectors of the form $(1, a, b, b, \dots b)$ and is achieved by letting k_m approach ∞:

$$
\lim_{k_m \to \infty} t(k_m) = \frac{a^4 b - a^3 b^2 - a^2 b^3 + \sqrt{b^2 \left(-a^4 + 3a^3 b - 3a^2 b^2 + ab^3 + (b-1)^3\right)^2} + ab^4 + b^4 - b^3 - b^2 + b}{2b\left(a^3 - 2a^2 b + ab^2 + (b-1)^2\right)}.
$$

The minimum value of this expression over all $a \in [0, 1], b \in [0, 1]$ is 0.704 (computed again with Mathematica). Overall, assuming our conjecture regarding the forms of the vectors achieving the minima and maxima, then it follows that $t \in (0.7, 1]$, and the worst-case relative error at the $p^*_{\max}$ contour will be bounded by

$$
|t^{\frac{1}{p^*_{\max}}} - 1| < 1 - 0.7^{\frac{4}{p^*_{\max}}}.
$$

References

Marc Boyer, Guillaume Dufour, and Luca Santinelli. Continuity for network calculus. In *21st International Conference on Real-Time Networks and Systems*, page 235. ACM Press, October 2013.

David Bremner, Timothy M Chan, Erik D Demaine, Jeff Erickson, Ferran Hurtado, John Iacono, Stefan Langerman, and Perouz Taslakian. Necklaces, convolutions, and x+y. In *Annual European Symposium on Algorithms*, pages 160–171. Springer, 2006.

Michael Bussieck, Hannes Hassler, Gerhard J Woeginger, and Uwe T Zimmermann. Fast algorithms for the maximum convolution problem. *Operations Research Letters*, 15(3):133–141, April 1994.

Arthur Stanley Eddington. *Mathematical Theory of Relativity*. Cambridge University Press, London, 1923.

Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 1999.

Gerhard X Ritter and Joseph N Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, Inc., August 2000.

Vladimir Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of Computational Physics*, 60(2):187–207, 1985.

James C Schatzman. Accuracy of the discrete Fourier transform and the fast Fourier transform. *SIAM Journal on Scientific Computing*, 17(5):1150–1166, September 1996.

Oliver Serang. The probabilistic convolution tree: efficient exact Bayesian inference for faster LC-MS/MS protein inference. *PLOS ONE*, 9(3):e91507, January 2014.

Oliver Serang. A fast numerical method for max-convolution and the application to efficient max-product inference in bayesian networks. *Journal of Computational Biology*, 22:770–783, 2015.

Oliver Serang, Michael J MacCoss, and William Stafford Noble. Efficient marginalization to compute protein posterior probabilities from shotgun mass spectrometry data. *Journal of Proteome Research*, 9(10):5346–57, October 2010.

Ning Sun and Zaifu Yang. The max-convolution approach to equilibrium analysis. Technical report, Bielefeld University, Center for Mathematical Economics, December 2002.

Christopher Zach, David Gallup, and Jan-Michael Frahm. Fast gain-adaptive KLT tracking on the GPU. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–7. IEEE, June 2008.

Journal of Machine Learning Research 17 (2016) 1-33 Submitted 7/15; Revised 11/15; Published 4/16

Hybrid Orthogonal Projection and Estimation (HOPE): A New Framework to Learn Neural Networks

Shiliang Zhang ZSL2008@MAIL.USTC.EDU.CN
National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, Anhui, China

Hui Jiang HJ@CSE.YORKU.CA
Department of Electrical Engineering and Computer Science
York University, 4700 Keele Street, Toronto, Ontario, M3J 1P3, Canada

Lirong Dai LRDAI@USTC.EDU.CN
National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, Anhui, China

Editor: Yoshua Bengio

Abstract

In this paper, we propose a novel model for high-dimensional data, called the *Hybrid Orthogonal Projection and Estimation (HOPE)* model, which combines a linear orthogonal projection and a finite mixture model under a unified generative modeling framework. The HOPE model itself can be learned unsupervised from unlabelled data based on the maximum likelihood estimation as well as discriminatively from labelled data. More interestingly, we have shown the proposed HOPE models are closely related to neural networks (NNs) in a sense that each hidden layer can be reformulated as a HOPE model. As a result, the HOPE framework can be used as a novel tool to probe why and how NNs work, more importantly, to learn NNs in either supervised or unsupervised ways. In this work, we have investigated the HOPE framework to learn NNs for several standard tasks, including image recognition on MNIST and speech recognition on TIMIT. Experimental results have shown that the HOPE framework yields significant performance gains over the current state-of-the-art methods in various types of NN learning problems, including unsupervised feature learning, supervised or semi-supervised learning.

Keywords: Orthogonal Projection, PCA, Mixture Models, Neural Networks, Unsupervised Learning, von Mises-Fisher (vMF) model

1. Introduction

Machine learning systems normally consist of several distinct steps in design, namely feature extraction and data modeling. In feature extraction, some engineering tricks are used to pre-process raw data to extract useful and representative features for the underlying data sets. As a result, this stage is sometimes called *feature engineering*. For high-dimensional data, the feature extraction stage needs to distill "good" features that are representative enough to discriminate different data samples but also it has to perform effective dimensionality reduction to generate less correlated features that can be easily modeled in a lower dimensional space. In data modeling, an appropriate model is selected to model data in the lower-dimensional feature space. There are a wide range of models available for this purpose, such as k-Nearest-Neighbours (kNN) methods, decision trees, linear discriminant models, neural networks, statistical models from the exponential family, or mixtures of the exponential family distributions, and so on. Subsequently, all unknown model parameters are estimated from available training samples based on certain learning criterion, such as maximum likelihood estimation (MLE) or discriminative learning.

In many traditional machine learning methods, feature extraction and data modeling are normally conducted independently in two loosely-coupled stages, where feature extraction parameters and model parameters are separately optimized based on rather different criteria. Particularly, feature extraction is normally regarded as a pre-processing stage, where feature extraction parameters are estimated under some quite loose conditions, such as the assumption that data is normally distributed in a high-dimensional space as implied in linear discriminant analysis (LDA) and principal component analysis (PCA). On the other hand, neural networks (NNs) favor an end-to-end learning process, which is normally considered as one exception to the above paradigm. In practice, it has been widely observed that NNs are capable of dealing with almost any type of raw data directly without any explicit feature engineering. In the recent resurgence of NNs in "deep learning", more and more empirical results have demonstrated that deep neural networks (DNNs) can effectively de-correlate high-dimensional raw data and automatically learn useful features from large training sets, without being disturbed by "the curse of dimensionality". However, it still remains as an open question why NNs can handle it and what mechanism is used by NNs to de-correlate high-dimensional raw data to learn good feature representations for many complicated real-world tasks.

In this paper, we first propose a novel data modeling framework for high-dimensional data, namely Hybrid Orthogonal Projection and Estimation (HOPE)(Zhang and Jiang, 2015; Zhang et al., 2015). The key argument for the HOPE framework is that feature extraction and data modeling should not be decoupled into two separate stages in learning since a good feature extraction module can not be learned based on some over-simplified and unrealistic modeling assumptions. The feature extraction and data modeling must be jointly learned and optimized by considering the complex nature of data distributions. This is particularly important in coping with high-dimensional data arising from most real-world applications, such as image, video and speech signals. In the HOPE framework, we propose to model high-dimensional data by combining a relatively simple feature extraction model, namely a linear orthogonal projection, with a powerful statistical model for data modeling, namely a finite mixture model of the exponential family distributions, under a unified generative modeling framework. In this paper, we consider two possible choices for the mixture models, namely Gaussian mixture models (GMMs) and mixtures of the von Mises-Fisher (movMFs) distributions. First of all, an orthogonal linear projection is used in feature extraction to ensure that the highly-correlated high-dimensional raw data is first projected onto a lower-dimensional latent feature space, where all feature dimensions are largely de-correlated. This will facilitate data modeling in this feature space over the original data space. Secondly, in the HOPE framework, we propose to use a powerful model to represent data in the lower-dimensional feature space, rather than using any over-simplified models for computational convenience. This is very important since any real-world data tend to follow a rather complex distribution, which can always be approximated by a finite mixture model

up to any arbitrary precision. Thirdly, the most important argument in HOPE is that both the orthogonal projection and the mixture model must be learned jointly according to a single unified criterion. In this paper, we first study how to learn HOPE in an unsupervised manner based on the conventional maximum likelihood (ML) criterion and also explain that the HOPE models can also be learned in a supervised way based on any discriminative learning criterion.

Another important finding in this work is that the proposed HOPE models are closely related to neural networks (NNs) currently widely used in deep learning. As we will show, any single hidden layer in the most popular rectified linear (ReLU) NNs can always be reformulated as a HOPE model consisting of a linear orthogonal projection and a mixture of von Mises-Fisher distributions (movMFs). This formulation helps to explain how NNs actually deal with high-dimensional data and why NNs can de-correlate almost any types of high-dimensional data to generate good feature representations. More importantly, this formulation may open up new possibilities to learn NNs more effectively. For example, both supervised and unsupervised learning algorithms for the HOPE models can be easily applied to learning NNs. By imposing an explicit orthogonal constraint on the feature extraction layer, we will show that the HOPE methods are very effective in learning NNs for both supervised and unsupervised learning. In unsupervised learning, we have shown that the maximum likelihood (ML) based HOPE learning algorithms can serve as a very effective unsupervised learning method to learn NNs from unlabelled data. Our experimental results have shown that the ML-based HOPE learning algorithm can learn good feature representations in an unsupervised way without using any data labels. These unsupervised learned features may be fed to some simple post-stage classifiers, such as linear SVM, to yield comparable performance as deep NNs supervised learned end-to-end using data labels. Our proposed unsupervised learning algorithms significantly outperform the previous methods based on the Restricted Boltzmann Machine (RBM) (Hinton et al., 2006) and the autoencoder variants (Bengio et al., 2007; Vincent et al., 2008). Moreover, in supervised learning, relying on the HOPE models, we have managed to learn some shallow NNs from scratch, which perform comparably with the state-of-the-art deep neural networks (DNNs), as opposed to learning shallow NNs to mimic a pre-trained deep neural network as in Ba and Caruana (2014). Finally, the HOPE models can also be used to train deep neural networks and it normally provides significant performance gain over the standard NN learning methods. These results have suggested that the orthogonal constraint in HOPE may serve as a good model regularization in learning of NNs.

2. Related Work

Dimensionality reduction in feature space is a well-studied topic in machine learning. Among many, PCA is the most popular technique in this category. PCA is defined as the orthogonal projection of the high-dimensional data onto a lower dimensional linear space, known as the principal subspace, such that the variance of the projected data is maximized (Bishop, 2006). The nice property of PCA is that it can be formulated as an eigenvector problem of the data covariance matrix, where a simple closed-form solution exists. Moreover, in the probabilistic PCA (Tipping and Bishop, 1999a; Roweis, 1998), PCA can be expressed as the maximum likelihood solution to a probabilistic latent variable model. In this case, if

the projected data are assumed to follow a zero-mean unit-covariance Gaussian distribution in the principal subspace, the probabilistic PCA can also be solved by an exact closed-form solution related to the eigenvectors of the data covariance matrix. The major limitation of PCA is that it is constrained to learn a linear subspace. Many approaches have been proposed to perform nonlinear dimension reduction to learn possible nonlinear manifolds embedded within a high dimensional data space. One way to model the nonlinear structure is through a combination of linear models, so that we make a piece-wise linear approximation to the manifold. This can be obtained by a clustering technique to partition the data set into local groups with standard PCA applied to each group. (see Hinton et al., 1997; Kambhatla and Leen, 1997; Tipping and Bishop, 1999b). In mixtures of probabilistic PCA (Tipping and Bishop, 1999b), the high-dimensional raw data is assumed to follow a mixture model, where each component may perform its own maximum likelihood PCA in a local region. However, in these methods, it may be quite challenging to perform effective clustering or estimate good mixture models for high-dimensional data due to the strong correlation in various data dimensions. Alternatively, a flexible nonlinear method is proposed to reduce feature dimension based on a deep auto-associative neural network (Hinton and Salakhutdinov, 2006).

Similar to PCA, the Fisher's linear discriminant analysis (LDA) can also be viewed as a linear dimensionality reduction technique. However, PCA is unsupervised in the sense that PCA depends only on the data while Fisher's LDA is supervised since it uses both data and class-label information. The high-dimensional data are linearly projected to a subspace where various classes are best distinguished as measured by the Fisher criterion. Moreover, the so-called heteroscedastic discriminant analysis (HLDA) (Kumar and Andreou, 1998) is proposed to extend LDA to deal with high-dimensional data with heteroscedastic covariance, where a linear projection can be learned from data and class labels based on the maximum likelihood criterion. On the other hand, in the independent component analysis (ICA) methods (Hyvarinen and Oja, 2000), it assumes the latent features follow a fully factorized non-Gaussian distribution, where non-Gaussianity is critical to distinguish independent components in the latent space based on the mixed observations from the original data space. In this paper, we make a more general assumption on the distributions in the latent feature space, which are only factorized between the signal and noise components.

Recently, many deep learning approaches have been proposed to capture complex data distributions, such as Restricted Boltzmann Machine (RBM) (Hinton et al., 2006) and the autoencoder variants (Bengio et al., 2007; Vincent et al., 2008). In the so-called independent factor analysis (Attias, 1999) and variational auto-encoders (Kingma and Welling, 2014; Rezende et al., 2014), a variational Bayes approach is adopted to model the joint distribution of the observed data and latent features, leading to a variational lower-bound of the data likelihood function, which can be maximized in model learning. On the contrary, in this paper, we take a rather different approach to model the data distribution, where the observed data and the latent features are viewed as two sets of random variables that are linked by a deterministic transformation. In this way, we may derive a much simpler formulation that allows us to directly maximize the data likelihood function without using sampling and variational approximation. A simple case of two classes was studied for a one-dimension feature space in an earlier work (Hinton and Nowlan, 1990). Moreover, relying on an orthogonal constraint, we may further simplify the computation of the underlying

Jacobian matrix. Note that a different trick, namely the triangular structure constraint, is used to simplify the way to compute the determinants of the Jacobian matrices (see Dinh et al., 2015; Oord and Dambre, 2015).

3. Hybrid Orthogonal Projection and Estimation (HOPE)

Consider a standard PCA setting, where each data sample is represented as a high-dimensional vector $\mathbf{x}$ with dimensionality D. Our goal is to learn a linear projection, represented as a matrix $\mathbf{U}$, to map each data sample onto a space having dimensionality $M < D$, which is called the latent feature space hereafter in this paper. Our proposed HOPE model is essentially a generative model in nature but it may also be viewed as a generalization to extend the probabilistic PCA in Tipping and Bishop (1999a) to consider a complex data distribution that has to be modeled by a finite mixture model in the latent feature space. This setting is very different from Tipping and Bishop (1999b), where the original data $\mathbf{x}$ is modeled by mixture models in the original higher D-dimensional raw data space.

3.1 HOPE: Combining Generalized PCA with Generative Model

Assume we have a full-size $D \times D$ orthogonal matrix $\hat{\mathbf{U}}$, satisfying $\hat{\mathbf{U}}^T\hat{\mathbf{U}} = \hat{\mathbf{U}}\hat{\mathbf{U}}^T = \mathbf{I}$, each data sample $\mathbf{x}$ in the original D-dimensional data space can be decomposed based on all orthogonal row vectors of $\hat{\mathbf{U}}$, denoted as $\mathbf{u}_i$ with $i = 1, \cdots, D$, as follows:

$$\mathbf{x} = \sum_{i=1}^{D} (\mathbf{x} \cdot \mathbf{u}_i)\, \mathbf{u}_i. \tag{1}$$

As shown in PCA, each high-dimensional data $\mathbf{x}$ can normally be represented fairly precisely in a lower-dimensional principal subspace and the contributions from the remaining dimensions may be viewed as random residual noises that have sufficiently small variances. Therefore, we have

$$\mathbf{x} = \underbrace{(\mathbf{x} \cdot \mathbf{u}_1)\, \mathbf{u}_1 + \cdots + (\mathbf{x} \cdot \mathbf{u}_M)\, \mathbf{u}_M}_{signal\ \ component\ \ \tilde{\mathbf{x}}} + \underbrace{(\mathbf{x} \cdot \mathbf{u}_{M+1})\, \mathbf{u}_{M+1} + \cdots + (\mathbf{x} \cdot \mathbf{u}_D)\, \mathbf{u}_D}_{noise\ \ component\ \ \bar{\mathbf{x}}} \tag{2}$$

Here we are interested in learning an $M \times D$ projection matrix, denoted as $\mathbf{U}$, to extract the signal component $\tilde{\mathbf{x}}$. First of all, if M ($M < D$) is selected properly, the projection may serve as an effective feature extraction for signals as well as a mechanism to eliminate unwanted noises from the higher dimensional raw data. This may make the subsequent learning process more robust and less prone to overfitting. Secondly, all M row vectors $\mathbf{u}_i$ with $i = 1, \cdots, M$ are learned to represent signals well in a lower M-dimension space. Furthermore, since all $\mathbf{u}_i$ are orthogonal, it implies the latent features are largely de-correlated. This will significantly simplify the following learning problem as well.

In this case, each D-dimension data sample, $\mathbf{x}$, is linearly projected onto an M-dimension vector $\mathbf{z}$ as $\mathbf{z} = \mathbf{U}\mathbf{x}$, where $\mathbf{U}$ is an orthogonal matrix, satisfying $\mathbf{U}\mathbf{U}^T = \mathbf{I}$. Meanwhile, we denote the projection of the unwanted noise component $\bar{\mathbf{x}}$ as $\mathbf{n}$, and $\mathbf{n}$ can be similarly computed as $\mathbf{n} = \mathbf{V}\mathbf{x}$, where $\mathbf{V}$ is another orthogonal matrix corresponding to all noise dimensions, satisfying $\mathbf{V}\mathbf{V}^T = \mathbf{I}$. Moreover, $\mathbf{V}$ is orthogonal to $\mathbf{U}$, i.e. $\mathbf{V}\mathbf{U}^T = \mathbf{0}$. In overall, we may represent the above projection as follows:

$$\big[\mathbf{z};\ \mathbf{n}\big] = \big[\mathbf{U};\ \mathbf{V}\big]\mathbf{x} = \hat{\mathbf{U}}\mathbf{x} \tag{3}$$

where $\hat{\mathbf{U}}$ is the above-mentioned $D \times D$ orthogonal projection matrix.

Moreover, it is straightforward to show that the signal component $\tilde{\mathbf{x}}$ and the residual noise $\bar{\mathbf{x}}$ in the original data space can be easily computed from the above projections as follows:

$$\tilde{\mathbf{x}} = \mathbf{U}^T\mathbf{z} = \mathbf{U}^T\mathbf{U}\mathbf{x} \tag{4}$$

$$\bar{\mathbf{x}} = \mathbf{x} - \tilde{\mathbf{x}} = (\mathbf{I} - \mathbf{U}^T\mathbf{U})\mathbf{x} \tag{5}$$

In the following, we consider how to learn the projection matrix $\mathbf{U}$ to represent D-dimensional data well in a lower M-dimension feature space. If this projection is learned properly, we may assume the above signal projection, $\mathbf{z}$, and the residual noise projection, $\mathbf{n}$, are independent in the latent feature space. Therefore, we may derive the probability distribution of the original data as follows:

$$p(\mathbf{x}) = |\hat{\mathbf{U}}^{-1}| \cdot p(\mathbf{z}) \cdot p(\mathbf{n}) \tag{6}$$

where $\hat{\mathbf{U}}^{-1}$ denotes the Jacobian matrix to linearly map the data from the projected space back to the original data space. If $\hat{\mathbf{U}}$ is orthonormal, the above Jacobian term equals to one. In this work, we follow Tipping and Bishop (1999a) to assume the residual noise projection $\mathbf{n}$ follows an isotropic covariance Gaussian distribution in the (D-M)-dimensional space, i.e. $p(\mathbf{n}) \sim \mathcal{N}(\mathbf{n} \mid \mathbf{0}, \sigma^2\mathbf{I})$, [1] where σ^2 is a variance parameter to be learned from data. As for the signal projection $\mathbf{z}$, we adopt a quite different approach, as described below in detail.

In all previous works, the signal projection $\mathbf{z}$ is assumed to follow a simple distribution in the M-dimension space. For example, $\mathbf{z}$ is assumed to follow a zero-mean unit-covariance Gaussian distribution in probabilistic PCA (Tipping and Bishop, 1999a; Roweis, 1998). The advantage of this assumption is that an exact closed-form solution may be derived to calculate the projection matrix $\mathbf{U}$ using the spectral methods based on the data covariance matrix.

However, in most real-world applications, $\mathbf{z}$ is still located in a very high-dimensional space even after the linear projection, it does not make sense to assume $\mathbf{z}$ follows a simple unimodal distribution. As widely reported in the literature, it is empirically observed that real-world data normally do not follow a unimodal distribution in a high-dimensional space and they usually appear only in multiple concentrated regions in the high-dimensional space. More realistically, it is better to assume that $\mathbf{z}$ follows a finite mixture model in the M-dimension feature space because a finite mixture model may theoretically approximate any arbitrary statistical distribution as long as a sufficiently large number of mixture components are used. For simplicity, we may assume $\mathbf{z}$ follows a finite mixture of some exponential family distributions:

$$p(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \cdot f_k(\mathbf{z}|\boldsymbol{\theta}_k) \tag{7}$$

1. Without losing generality, we may simply normalize the training data to ensure that the residual noises have zero mean.

where π_k denotes mixture weights with $\sum_{k=1}^{K} \pi_k = 1$, and $f_k(\mathbf{z}|\boldsymbol{\theta}_k)$ stands for a unimodal distribution from the exponential family with model parameters $\boldsymbol{\theta}_k$. We use $\boldsymbol{\Theta}$ to denote all model parameters in the mixture model, i.e., $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_k, \pi_k \mid k = 1, \cdots, K\}$. In practice, $f_k(\mathbf{z}|\boldsymbol{\theta}_k)$ is chosen from the exponential family based on the nature of data. In this paper, we consider two possible choices for high-dimensional continuous data, namely the multivariate Gaussian distributions and the von Mises-Fisher (vMF) distributions. The learning algorithm can be easily extended to other models in the exponential family.

For example, if we choose the multivariate Gaussian distribution, then $\mathbf{z}$ follows a Gaussian mixture model (GMM) as follows:

$$p(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \cdot f_k(\mathbf{z}|\boldsymbol{\theta}_k) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}_k, \Sigma_k) \tag{8}$$

where $\mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}_k, \Sigma_k)$ denotes a multivariate Gaussian distribution with the mean vector $\boldsymbol{\mu}_k$ and the covariance matrix Σ_k. Since the projection matrix $\mathbf{U}$ is orthogonal, all dimensions in $\mathbf{z}$ are highly de-correlated. Therefore, it is reasonable to assume each Gaussian component has a diagonal covariance matrix Σ_k. This may significantly simplify the model estimation of GMMs.

Alternatively, we may select a less popular model, i.e., the von Mises-Fisher (vMF) distribution.[2] The vMF distribution may be viewed as a generalized normal distribution defined on a high-dimensional spherical surface. In this case, $\mathbf{z}$ follows a mixture of the von Mises-Fisher (movMF) distributions as follows:

$$p(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \cdot f_k(\mathbf{z}|\boldsymbol{\theta}_k) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{C}_M(|\boldsymbol{\mu}_k|) \cdot e^{\mathbf{z} \cdot \boldsymbol{\mu}_k} \tag{9}$$

where $\mathbf{z}$ is located on the surface of an M-dimensional sphere, i.e., $|\mathbf{z}| = 1$, $\boldsymbol{\mu}_k$ denotes all model parameters of the k-th vMF component and it is an M-dimensional vector in this case, and $\mathcal{C}_M(\kappa)$ is the probability normalization term of the k-th vMF component defined as:

$$\mathcal{C}_M(\kappa) = \frac{\kappa^{M/2-1}}{(2\pi)^{M/2} I_{M/2-1}(\kappa)} \tag{10}$$

where $I_v(\cdot)$ denotes the modified Bessel function of the first kind at order v.

4. Unsupervised Learning of HOPE Models

Obviously, the HOPE model is essentially a generative model that combines feature extraction and data modelling together, and thus its model parameters, including both the projection matrix and the mixture model, can be estimated based on the maximum likelihood (ML) criterion, just like normal generative models as well as the probabilistic PCA in Tipping and Bishop (1999a). However, since $\mathbf{z}$ follows a mixture distribution, no closed-form solution is available to derive either the projection matrix or the mixture model. In

2. The main reason why we are interested in the von Mises-Fisher (vMF) distributions is that the choice of the vMF model can strictly link our HOPE model to regular neural networks in deep learning. We will elucidate this later in this paper.

this case, some iterative optimization algorithms, such as stochastic gradient descent (SGD) (Bottou, 2004), may be used to jointly estimate both the projection matrix $\mathbf{U}$ and the underlying mixture model altogether to maximize a joint likelihood function. In this section, we assume the projection matrices, not only $\mathbf{U}$ but also the whole $\hat{\mathbf{U}}$, are all orthonormal. As a result, the Jacobian term in eq.(6) disappears since it equals to one. Refer to Appendix A for the case where $\mathbf{U}$ is not orthonormal.

Given a training set as $\mathbf{X} = \{\mathbf{x}_n \mid n = 1, \cdots, N\}$, assume that all $\mathbf{x}_n$ are normalized to be of unit length as $|\mathbf{x}_n| = 1$, the joint log-likelihood function related to all HOPE parameters, including the projection matrix $\mathbf{U}$, the mixture model $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_k | k = 1, \cdots, K\}$ and residual noise variance σ, can be expressed as follows:

$$\begin{aligned} \mathcal{L}(\mathbf{U}, \boldsymbol{\Theta}, \sigma \mid \mathbf{X}) &= \sum_{n=1}^{N} \ln \Pr(\mathbf{x}_n) = \sum_{n=1}^{N} \Big[\ln \Pr(\mathbf{z}_n) + \ln \Pr(\mathbf{n}_n) \Big] \\ &= \underbrace{\sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \cdot f_k(\mathbf{U}\mathbf{x}_n|\boldsymbol{\theta}_k) \right)}_{\mathcal{L}_1(\mathbf{U},\boldsymbol{\Theta})} + \underbrace{\sum_{n=1}^{N} \ln \Big(\mathcal{N}\big(\mathbf{n}_n \,|\, \mathbf{0}, \sigma^2 \mathbf{I}\big) \Big)}_{\mathcal{L}_2(\mathbf{U},\sigma)} \end{aligned} \tag{11}$$

The HOPE parameters, including $\mathbf{U}$, $\boldsymbol{\Theta}$ and σ, can all be estimated by maximizing the above likelihood function as:

$$\{\mathbf{U}^*, \boldsymbol{\Theta}^*, \sigma^*\} = \arg\max_{\mathbf{U},\boldsymbol{\Theta},\sigma} \quad \mathcal{L}(\mathbf{U}, \boldsymbol{\Theta}, \sigma \mid \mathbf{X}) \tag{12}$$

subject to the orthogonal constraint:

$$\mathbf{U}\mathbf{U}^T = \mathbf{I}. \tag{13}$$

There are many methods to enforce the above orthogonal constraint in the optimization. For example, we may periodically apply the Gram-Schmidt process in linear algebra to orthogonalize $\mathbf{U}$ during the optimization process. In this work, for computational efficiency, we follow Bao et al. (2013, 2012) to cast the orthogonal constraint condition in eq.(13) as a penalty term in the objective function to convert the above constrained optimization problem into an unconstrained one as follows:

$$\{\mathbf{U}^*, \boldsymbol{\Theta}^*, \sigma^*\} = \arg\max_{\mathbf{U},\boldsymbol{\Theta},\sigma} \quad \Big[\mathcal{L}(\mathbf{U}, \boldsymbol{\Theta}, \sigma \mid \mathbf{X}) - \beta \cdot \mathcal{D}(\mathbf{U}) \Big] \tag{14}$$

where β ($\beta > 0$) is a control parameter to balance the contribution of the penalty term, and the penalty term $\mathcal{D}(\mathbf{U})$ is a differentiable function as:

$$\mathcal{D}(\mathbf{U}) = \sum_{i=1}^{M} \sum_{j=i+1}^{M} \frac{|\mathbf{u}_i \cdot \mathbf{u}_j|}{|\mathbf{u}_i| \cdot |\mathbf{u}_j|} \tag{15}$$

with $\mathbf{u}_i$ denoting the i-th row vector of the projection matrix $\mathbf{U}$, and $\mathbf{u}_i \cdot \mathbf{u}_j$ representing the inner product of $\mathbf{u}_i$ and $\mathbf{u}_j$. The norms of all row vectors of $\mathbf{U}$ need to be normalized to one after each update.

In this work, we propose to use the stochastic gradient descent (SGD) method to optimize the objective function in eq.(14). In this case, given any training data or a mini-batch of them, we calculate the gradients of the objective function with respect to the projection matrix, $\mathbf{U}$, and the parameters of the mixture model, $\mathbf{\Theta}$, and then update them iteratively until the objective function converges. The gradients of $\mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})$ depends on the mixture model to be used. In the following, we first consider how to compute the derivatives of $\mathcal{D}(\mathbf{U})$ and $\mathcal{L}_2(\mathbf{U}, \sigma)$, which are general for all HOPE models. After that, as two examples, we will show how to calculate the derivatives of $\mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})$ for GMMs and movMFs.

4.1 Dealing with the Penalty Term $\mathcal{D}(\mathbf{U})$

Following Bao et al. (2013), the gradients of the penalty term $\mathcal{D}(\mathbf{U})$ with respect to each row vector, $\mathbf{u}_i$ $(i = 1, \cdots, M)$, can be easily derived as follows:

$$\frac{\partial \mathcal{D}(\mathbf{U})}{\partial \mathbf{u}_i} = \sum_{j=1}^{M} \; g_{ij} \cdot \left[\frac{\mathbf{u}_j}{\mathbf{u}_i \cdot \mathbf{u}_j} - \frac{\mathbf{u}_i}{\mathbf{u}_i \cdot \mathbf{u}_i} \right] \tag{16}$$

where g_{ij} denotes the absolute cosine value of the angle between two row vectors, $\mathbf{u}_i$ and $\mathbf{u}_j$, computed as follows:

$$g_{ij} = \frac{|\mathbf{u}_i \cdot \mathbf{u}_j|}{|\mathbf{u}_i| \cdot |\mathbf{u}_j|}. \tag{17}$$

The above derivatives can be equivalently represented as the following matrix form:

$$\frac{\partial \mathcal{D}(\mathbf{U})}{\partial \mathbf{U}} = (\mathbf{D} - \mathbf{B})\mathbf{U} \tag{18}$$

where $\mathbf{D}$ is an $M \times M$ matrix, with its elements computed as $d_{ij} = \frac{\text{sign}(\mathbf{u}_i \cdot \mathbf{u}_j)}{|\mathbf{u}_i| \cdot |\mathbf{u}_j|}$ $(1 \leq i, j \leq M)$, and $\mathbf{B}$ is an $M \times M$ diagonal matrix, with its diagonal elements computed as $b_{ii} = \frac{\sum_{j=1} g_{ij}}{\mathbf{u}_i \cdot \mathbf{u}_i}$ $(1 \leq i \leq M)$.

4.2 Dealing with the Noise Model Term $\mathcal{L}_2$

The log-likelihood function related to the noise model, $\mathcal{L}_2(\mathbf{U}, \sigma)$, can be expressed as:

$$\mathcal{L}_2(\mathbf{U}, \sigma) = -\frac{N(D-M)}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^{N} \mathbf{n}_n^T \mathbf{n}_n. \tag{19}$$

And we have:

$$\begin{aligned} \mathbf{n}_n^T \mathbf{n}_n &= (\mathbf{x}_n - \mathbf{U}^T \mathbf{z}_n)^T (\mathbf{x}_n - \mathbf{U}^T \mathbf{z}_n) = (\mathbf{x}_n - \mathbf{U}^T \mathbf{U} \mathbf{x}_n)^T (\mathbf{x}_n - \mathbf{U}^T \mathbf{U} \mathbf{x}_n) \\ &= \mathbf{x}_n^T \mathbf{x}_n - 2\mathbf{x}_n^T \mathbf{U}^T \mathbf{U} \mathbf{x}_n + \mathbf{x}_n^T \mathbf{U}^T \mathbf{U} \mathbf{U}^T \mathbf{U} \mathbf{x}_n \end{aligned} \tag{20}$$

Therefore, $\mathcal{L}_2(\mathbf{U}, \sigma)$ can be expressed as:

$$\mathcal{L}_2(\mathbf{U}, \sigma) = -\frac{N(D-M)}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^{N} \left[\mathbf{x}_n^T \mathbf{x}_n - 2\mathbf{x}_n^T \mathbf{U}^T \mathbf{U} \mathbf{x}_n + \mathbf{x}_n^T \mathbf{U}^T \mathbf{U} \mathbf{U}^T \mathbf{U} \mathbf{x}_n \right] \tag{21}$$

The gradient with respect to $\mathbf{U}$ [3] can be derived as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}_2(\mathbf{U}, \sigma)}{\partial \mathbf{U}} &= \frac{1}{\sigma^2} \sum_{n=1}^{N} \left[2\mathbf{U}\mathbf{x}_n\mathbf{x}_n^T - \mathbf{U}\mathbf{U}^T\mathbf{U}\mathbf{x}_n\mathbf{x}_n^T - \mathbf{U}\mathbf{x}_n\mathbf{x}_n^T\mathbf{U}^T\mathbf{U} \right] \\ &= \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{U} \left[\mathbf{x}_n(\bar{\mathbf{x}}_n)^T + \bar{\mathbf{x}}_n(\mathbf{x}_n)^T \right]. \end{aligned} \tag{22}$$

For the noise variance σ^2, we can easily derive the following closed-form update formula by vanishing its derivative to zero:

$$\sigma^2 = \frac{1}{N(D-M)} \sum_{n=1}^{N} \mathbf{n}_n^T \mathbf{n}_n \tag{23}$$

As long as the learned noise variance σ^2 is small enough, maximizing the above term $\mathcal{L}_2$ will force all signal dimensions into the projection matrix $\mathbf{U}$ and only the residual noises will be modelled by $\mathcal{L}_2$.

4.3 Computing $\mathcal{L}_1$ for GMMs

In this section, we consider how to compute the partial derivatives of $\mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})$ for GMMs. Assume each mini-batch $\mathbf{X}$ consists of a small subset of randomly selected training samples, $\mathbf{X} = \{\mathbf{x}_n \mid n = 1, \cdots, N\}$, the log likelihood function of HOPE models with GMMs can be represented as follows:

$$\mathcal{L}_1(\mathbf{U}, \mathbf{\Theta}) = \sum_{n=1}^{N} \; \ln \left[\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\mathbf{U}\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k) \right] \tag{24}$$

The partial derivative of $\mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})$ w.r.t the mean vector, $\boldsymbol{\mu}_k$, of the k-th Gaussian component can be calculated as follows:

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})}{\partial \boldsymbol{\mu}_k} = \sum_{n=1}^{N} \gamma_k(\mathbf{z}_n) \cdot \Sigma_k^{-1}(\mathbf{z}_n - \boldsymbol{\mu}_k) \tag{25}$$

where $\mathbf{z}_n = \mathbf{U}\mathbf{x}_n$, and $\gamma_k(\mathbf{z}_n)$ denotes the so-called occupancy statistics of the k-th Gaussian component, computed as $\gamma_k(\mathbf{z}_n) = \frac{\pi_k \mathcal{N}(\mathbf{z}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{z}_n | \boldsymbol{\mu}_j, \Sigma_j)}$.

The partial derivative of $\mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})$ w.r.t π_k can be simply derived as follows:

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \mathbf{\Theta})}{\partial \pi_k} = \sum_{n=1}^{N} \; \frac{\gamma_k(\mathbf{z}_n)}{\pi_k} \tag{26}$$

3. We may use the constraint $\mathbf{U}\mathbf{U}^T = \mathbf{I}$ to significantly simplify the above derivation. However, that leads to a gradient computation strongly relying on the orthogonal constraint. Since we use SGD to iteratively optimize all model parameters, including $\mathbf{U}$. We can not ensure $\mathbf{U}\mathbf{U}^T = \mathbf{I}$ strictly holds anytime in the SGD process. Therefore, the simplified gradient usually yields poor convergence performance.

The partial derivative of $\mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})$ w.r.t the Σ_k is computed as follows:

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \Sigma_k} = -\frac{1}{2} \sum_{n=1}^{N} \gamma_k(\mathbf{z}_n) \Big[\Sigma_k^{-1} - \Sigma_k^{-1} (\mathbf{z}_n - \boldsymbol{\mu}_k)(\mathbf{z}_n - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} \Big]. \tag{27}$$

When we use the above gradients to update Gaussian covariance matrices in SGD, we have to impose additional constraints to ensure all covariance matrices are positive semidefinite. However, if we adopt diagonal covariance matrices for all Gaussian components, these constraints can be implemented in a fairly simple way.

Finally, the partial derivative of $\mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})$ w.r.t the projection matrix $\mathbf{U}$ is computed as:

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{U}} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_k(\mathbf{z}_n) \cdot \Sigma_k^{-1} (\boldsymbol{\mu}_k - \mathbf{z}_n) \mathbf{x}_n^T. \tag{28}$$

4.4 Computing $\mathcal{L}_1$ for movMFs

Similarly, we derive all partial derivatives of $\mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})$ for a mixture of vMFs (movMFs). In this case, given a mini-batch of training samples, $\mathbf{X} = \{\mathbf{x}_n \mid n = 1, \cdots, N\}$, the log-likelihood function of the HOPE model with movMFs can be expressed as follows:

$$\mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta}) = \sum_{n=1}^{N} \ \ln \left[\sum_{k=1}^{K} \pi_k \cdot \mathcal{C}_M(|\boldsymbol{\mu}_k|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_k} \right] \tag{29}$$

where each $\mathbf{z}_n$ must be normalized to be of unit length [4] as required by the vMF distribution as:

$$\tilde{\mathbf{z}}_n = \mathbf{U}\mathbf{x}_n, \quad \mathbf{z}_n = \frac{\tilde{\mathbf{z}}_n}{|\tilde{\mathbf{z}}_n|}. \tag{30}$$

Similar to the HOPE models with GMMs, we first define an occupancy statistic for k-th vMF component as:

$$\gamma_k(\mathbf{z}_n) = \frac{\pi_k \cdot \mathcal{C}_M(|\boldsymbol{\mu}_k|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_k}}{\sum_{j=1}^{K} \pi_j \cdot \mathcal{C}_M(|\boldsymbol{\mu}_j|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_j}}. \tag{31}$$

In a similar way, we can derive the partial derivatives of $\mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})$ with respect to π_k, $\boldsymbol{\mu}_k$ and $\mathbf{U}$ as follows:

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \pi_k} = \sum_{n=1}^{N} \frac{\gamma_k(\mathbf{z}_n)}{\pi_k} \tag{32}$$

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \boldsymbol{\mu}_k} = \sum_{n=1}^{N} \ \gamma_k(\mathbf{z}_n) \cdot \left[\mathbf{z}_n - \frac{\boldsymbol{\mu}_k}{|\boldsymbol{\mu}_k|} \cdot \frac{I_{M/2}(|\boldsymbol{\mu}_k|)}{I_{M/2-1}(|\boldsymbol{\mu}_k|)} \right] \tag{33}$$

4. In practice, we usually normalize all original data, $\mathbf{x}_n$, to be of unit length: $|\mathbf{x}_n| = 1$, prior to the HOPE model. In this case, as long as M is properly chosen (e.g., to be large enough), the projection matrix $\mathbf{U}$ is always learned to extract from $\mathbf{x}_n$ as much energy as possible. Therefore, this normalization step may be skipped because the norm of the projected $\mathbf{z}_n$ is always very close to one even without normalization in this stage, i.e., $|\mathbf{z}_n| = |\tilde{\mathbf{z}}_n| \approx 1$.

Algorithm 1 SGD-based Maximum Likelihood Learning Algorithm for HOPE

randomly initialize $\mathbf{u}_i$ $(i = 1, \cdots, M)$, π_k and $\boldsymbol{\theta}_k$ $(k = 1, \cdots, K)$
for *epoch* $= 1$ **to** T **do**
  for *minibatch* $\mathbf{X}$ **in** training set **do**
    $\mathbf{U} \leftarrow \mathbf{U} + \epsilon \cdot \Big(\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{U}} + \frac{\partial \mathcal{L}_2(\mathbf{U}, \sigma)}{\partial \mathbf{U}} - \beta \cdot \frac{\partial \mathcal{D}(\mathbf{U})}{\partial \mathbf{U}} \Big)$
    $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_k + \epsilon \cdot \frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \boldsymbol{\theta}_k}$ $(\forall k)$
    $\pi_k \leftarrow \pi_k + \epsilon \cdot \frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \pi_k}$ $(\forall k)$
    $\sigma^2 \leftarrow \frac{1}{N(D-M)} \sum_{n=1}^{N} \mathbf{n}_n^T \mathbf{n}_n$
    $\pi_k \leftarrow \frac{\pi_k}{\sum_j \pi_j}$ $(\forall k)$ and $\mathbf{u}_i \leftarrow \frac{\mathbf{u}_i}{|\mathbf{u}_i|}$ $(\forall i)$
  end for
end for

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{U}} = \sum_{n=1}^{N} \sum_{k=1}^{K} \ \frac{\gamma_k(\mathbf{z}_n)}{|\tilde{\mathbf{z}}_n|} \cdot (\mathbf{I} - \mathbf{z}_n \mathbf{z}_n^T) \boldsymbol{\mu}_k \mathbf{x}_n^T \tag{34}$$

Refer to Appendix B for all details on how to derive the above derivatives for the movMF distributions. Moreover, when movMFs are used, we need some special treatments to compute the Bessel functions in vMFs, i.e, $I_v(\cdot)$, as shown in eqs.(31) and (33). In this work, we adopt the numerical method in Abramowitz and Stegun (2006) to approximate the Bessel functions, refer to the Appendix C for the numerical details on this.

4.5 The SGD-based Learning Algorithm

Because all mixture weights, π_k $(k = 1, \cdots K)$, and all row vectors, $\mathbf{u}_i$ $(i = 1, \cdots, M)$ of the projection matrix satisfy the constraints: $\sum_k^K \pi_k = 1$ and $|\mathbf{u}_j| = 1$ $(\forall j)$. During the SGD learning process, π_k and $\mathbf{u}_i$ must be normalized after each update as follows:

$$\pi_k \leftarrow \frac{\pi_k}{\sum_j \pi_j} \tag{35}$$

$$\mathbf{u}_i \leftarrow \frac{\mathbf{u}_i}{|\mathbf{u}_i|}. \tag{36}$$

Finally, we summarize the SGD algorithm to learn the HOPE models based on the maximum likelihood (ML) criterion in Algorithm 1.

5. Learning Neural Networks as HOPE

As described above, the HOPE model may be used as a novel model for high-dimensional data. The HOPE model itself can be efficiently learned in an unsupervised manner from unlabelled data based on the above-mentioned maximum likelihood criterion. Moreover, if data labels are available, a variety of discriminative training methods (see Jiang, 2010; Jiang and Li, 2010; Jiang et al., 2015) may be used to learn the HOPE model in a supervised way based on some other discriminative learning criteria.

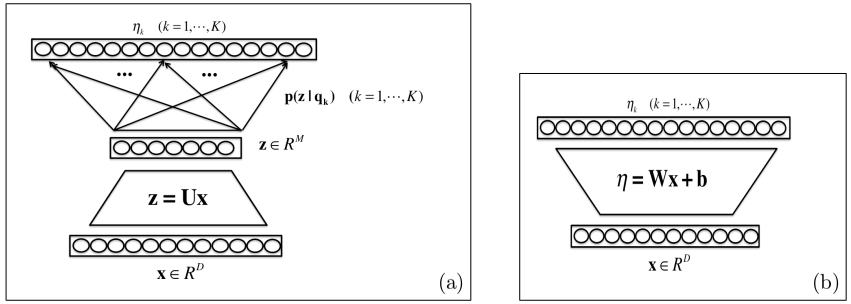


Figure 1: Illustration of a HOPE model as a layered network structure in (a). It may be equivalently reformulated as a hidden layer in neural nets shown in (b).

More interestingly, as we will elucidate here, there exists strong relationship between the HOPE models and neural networks (NNs). First of all, we will show that the HOPE models may be used as a new tool to probe the mechanism why NNs work so well in practice. Under the new HOPE framework, we may explain why NNs can almost universally excel on a variety of data types and how NNs handle various types of highly-correlated high-dimensional data, which may be quite challenging to many other machine learning models. Secondly, more importantly, the HOPE framework provides us with some new approaches to learn NNs: (i) Unsupervised learning: the maximum likelihood estimation of HOPE may be directly applied to learn NNs from unlabelled data; (ii) Supervised learning: the HOPE framework can be incorporated into the normal supervised learning of NNs by explicitly imposing some orthogonal constraints in learning. This may improve the learning of NNs and yield better and more compact models.

5.1 Linking HOPE to Neural Networks

A HOPE model normally consists of two stages: i) a linear orthogonal projection from the raw data space to the latent feature space; ii) a generative model defined as a finite mixture model in the latent feature. As a result, we may depict every HOPE model as a two-layer network: a linear projection layer and a nonlinear model layer, as shown in Figure 1 (a). The first layer represents the linear orthogonal projection from $\mathbf{x}$ ($\mathbf{x} \in \mathbf{R}^D$) to $\mathbf{z}$ ($\mathbf{z} \in \mathbf{R}^M$): $\mathbf{z} = \mathbf{U}\mathbf{x}$. The second layer represents the underlying finite mixture model in the feature space. Taking movMFs as an example, each node in the model layer represents the log-likelihood contribution from one mixture component as follows:

$$\phi_k = \ln\left(\pi_k \cdot f_k(\mathbf{z}|\boldsymbol{\theta}_k)\right) = \ln \pi_k + \ln \mathcal{C}_M(|\boldsymbol{\mu}_k|) + \mathbf{z} \cdot \boldsymbol{\mu}_k. \tag{37}$$

Given an input $\mathbf{x}$ (assuming $\mathbf{x}$ is projected to $\mathbf{z}$ in the latent feature space), if we know all ϕ_k ($1 \le k \le K$) in the model layer, we can easily compute the log-likelihood value of $\mathbf{x}$

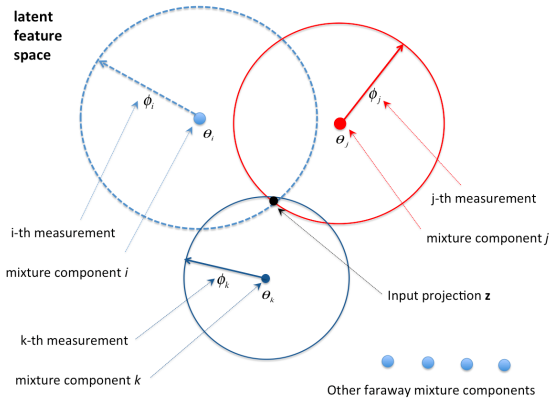


Figure 2: Illustration of the HOPE features as trilateration in the latent feature space.

from the HOPE model in eq.(9) as follows:

$$\ln p(\mathbf{z}) = \ln \sum_{k=1}^{K} \exp(\phi_k). \tag{38}$$

Moreover, all ϕ_k ($1 \le k \le K$) may be used as a set of distance measurements to locate the input projection, $\mathbf{z}$, in the latent space as *trilateration*, shown in Figure 2. Therefore, all ϕ_k ($1 \le k \le K$) may be viewed as a set of features to distinctly represent the original input $\mathbf{x}$.

Furthermore, we may use a preset threshold ε to prune the raw measurements, ϕ_k ($1 \le k \le K$), as follows:

$$\eta_k = \max(0, \phi_k - \varepsilon) \tag{39}$$

to eliminate those small log likelihood values from some faraway mixture components. Pruning small log-likelihood values as above may result in several benefits. Firstly, this pruning operation does not affect the total likelihood from the mixture model because it is always dominated by only a few top components. Therefore, we have:

$$\ln p(\mathbf{z}) \approx \varepsilon + \ln \sum_{k=1}^{K} \exp(\eta_k).$$

Secondly, this pruning step does not affect the trilateration problem in Figure 2. This is similar to the Global Positioning System (GPS) where the weak signals from faraway satellites are not used for localization. More importantly, the above pruning operation may improve robustness of the features since these small log-likelihood values may become very noisy. Note that the above pruning operation is similar to the rectified linear nonlinearity

in regular ReLU neural networks. Here we give a more intuitive explanation for the popular ReLU operation under the HOPE framework.

In this way, as shown in Figure 1 (a), all rectified log likelihood values in the model layer, i.e., η_k $(1 \leq k \leq K)$, may be viewed as a sensory map to measure each input, $\mathbf{x}$, in the latent feature space using all mixture components as the probers. Each pixel in the map, i.e., a pruned measurement η_k, roughly tells the distance between the centroid of a mixture component and the input projection $\mathbf{z}$ in the latent feature space. Under some condition (e.g., $K \gg M$), the input projection can be precisely located based on these pruned η_k values as a trilateration problem in the M-dimensional feature space, as shown in Figure 2. Therefore, this sensory map may be viewed as a feature representation learned to represent the input $\mathbf{x}$, which may be fed to a *softmax* classifier to form a normal shallow neural network, or to another HOPE model to form a deep neural network.

Moreover, since the projection layer is linear, it can be mathematically combined with the upper model layer to generate a single layer structure, as shown in Figure 1 (b). If movMFs are used in HOPE, it is equivalent to a hidden layer in normal rectified linear (ReLU) neural networks.[5] And the weight matrix in the merged layer can be simply derived from the HOPE model parameters, $\mathbf{U}$ and $\mathbf{\Theta}$. It is simple to show that the weight vectors for each hidden node k $(1 \leq k \leq K)$ in Figure 1 (b) may be derived as

$$\mathbf{w}_k = \mathbf{U}^T \boldsymbol{\mu}_k$$

and its bias is computed as

$$b_k = \ln \pi_k + \ln \mathcal{C}_M(|\boldsymbol{\mu}_k|) - \varepsilon.$$

Even though the linear projection layer may be merged with the model layer after all model parameters are learned, it may be beneficial to keep them separate during the model learning process. In this way, the model capacity may be controlled by two distinct control parameters: i) M can be selected properly to filter out noise components as in eq.(2) to prevent overfitting in learning; ii) K may be chosen independently to ensure the model is complex enough to model very big data sets for more difficult tasks. Moreover, we may enforce the orthogonal constraint, i.e., $\mathbf{U}\mathbf{U}^T = \mathbf{I}$, during the model learning to ensure that all dimensions of $\mathbf{z}$ are mostly de-correlated in the latent feature space, which may significantly simplify the density estimation in the feature space using a finite mixture model.

The formulation in Figure 1 (a) helps to explain the underlying mechanism of how neural networks work. Under the HOPE framework, it becomes clear that each hidden layer in neural networks may actually perform two different tasks implicitly, namely feature extraction and data modeling. This may shed some light on why neural nets can directly deal with various types of highly-correlated high-dimensional data (Pan et al., 2012) without any explicit dimensionality reduction and feature de-correlation steps.

Based on the above discussion, a HOPE model is mathematically equivalent to a hidden layer in neural networks. For example, each movMF HOPE model can be collapsed into a single weight matrix, same as a hidden layer of ReLU NNs. On the contrary, any weight

5. On the other hand, if GMMs are used in HOPE, it can be similarly shown that it is equivalent to a hidden layer in Radial basis function (RBF) networks (Park and Sandberg, 1991).

matrix in ReLU NNs may be decomposed as a product of two matrices as in Figure 1 (a) as long as M is not less than the rank of the weight matrix. In practice, we may deliberately choose a smaller value for M to regularize the models. Under this formulation, it is clear that neural networks may be trained under the HOPE framework. There are several advantages to learn neural networks under the HOPE framework. First of all, the modelling capacity of neural networks may be explicitly controlled by selecting proper values for M and K, each of which is chosen for a different purpose. Secondly, we can easily apply the maximum likelihood estimation of HOPE models in section 4 to unsupervised or semi-supervised learning to learn NNs from unlabelled data. Thirdly, the useful orthogonal constraints may be incorporated into the normal back-propagation process to learn better NN models in supervised learning as well.

5.2 Unsupervised Learning of Neural Networks as HOPE

The maximum likelihood estimation method for HOPE in section 4 can be used to learn neural networks layer by layer in an unsupervised learning mode. All HOPE model parameters in Figure 1 (a) are first estimated based on the maximum likelihood criterion as in section 4. Next, the two layers in the HOPE model are merged to form a regular NN hidden layer as in Figure 1 (b). In this case, class labels are not required to learn all network weights and neural networks can be learned from unlabelled data under a theoretically solid framework. This is similar to the Hebbian style learning (Rolls and Treves, 1998) but it has a well-founded and converging objective function in learning. As described above, the rectified log-likelihood values from the HOPE model, i.e., η_k $(1 \leq k \leq K)$, may be viewed as a sensory map using all mixture components as the probers in the latent feature space, which may serve as a good feature representation for the original data. At the end, a small amount of labelled data may be used to learn a simple classifier, either a *softmax* layer or a linear support vector machine (SVM), on the top of the HOPE layers, which takes the sensory map as input for final classification or prediction.

In unsupervised learning, the learned orthogonal projection matrix $\mathbf{U}$ may be viewed as a generalized PCA, which performs dimensionality reduction by considering the complex distribution modeled by a finite mixture model in the latent feature space.

5.3 Supervised Learning of Neural Networks as HOPE

The HOPE framework can also be applied to the supervised learning of neural networks when data labels are available. Let us take ReLU neural networks as example, each hidden layer in a ReLU neural network, as shown in Figure 1 (b), may be viewed as a HOPE model and thus it can be decomposed as a combination of a projection layer and a model layer, as shown in Figure 1 (a). In this case, M needs to be chosen properly to avoid overfitting. In other words, each hidden layer in ReLU NNs is represented as two layers during learning, namely a linear projection layer and a nonlinear model layer. If data labels are available, as in Jiang and Li (2010); Jiang et al. (2015), instead of using the maximum likelihood criterion, we may use other discriminative training criteria (Jiang, 2010) to form the objective function to learn all network parameters. Let us take the popular minimum cross-entropy error criterion as an example, given a training set of the input data and the class labels, i.e., $\mathbf{X} = \{\mathbf{x}_t, l_t \mid 1 \leq t \leq T\}$, we may use the HOPE outputs, i.e., all η_k in

eq.(39) to form the cross-entropy objective function as follows:

$$\mathcal{F}_{CE}(\mathbf{U}, \boldsymbol{\mu}_k, b_k \mid \mathbf{X}) = -\sum_{t=1}^{T} \log \left[\frac{\exp\left(\eta_{l_t}(\mathbf{x}_t)\right)}{\sum_{k=1}^{K} \exp\left(\eta_k(\mathbf{x}_t)\right)} \right] \tag{40}$$

Obviously, the standard back-propagation algorithm may be used to optimize the above objective function to learn all decomposed HOPE model parameters end-to-end, including $\mathbf{U}$, $\{\boldsymbol{\mu}_k, b_k \mid 1 \leq k \leq K\}$ from all HOPE layers. The only difference is that the orthogonal constraints, as in eq.(13), must be imposed for all projection layers during training, where the derivatives in eq.(18) must be incorporated in the standard back-propagation process to update each projection matrix $\mathbf{U}$ to ensure it is orthonormal. Note that in the supervised learning based on a discriminative training criterion, the unit-length normalization of the data projections in eq.(30) can be relaxed for computational simplicity since this normalization is only important for computing the likelihood for pure probabilistic models. After the learning, each pair of projection and model layers may be merged into a single hidden layer. After merging, the resultant network remains the exactly same network structure as normal ReLU neural networks. This learning method is related to the well-known low-rank matrix factorization method used for training deep NNs in speech recognition (Sainath et al., 2013; Xue et al., 2013). However, since we impose the orthogonal constraints for all projection matrices during the training process, it may lead to more compact models and/or better performance.

In supervised learning, the learned orthogonal projection matrix $\mathbf{U}$ may be viewed as a generalized LDA or HLDA (Kumar and Andreou, 1998), which optimizes the data projection to maximize (or minimize) the underlying discriminative learning criterion.

5.4 HOPE for Deep Learning

As above, the HOPE framework can be used to learn rather strong shallow NN models. However, this does not hinder HOPE from building deeper models for deep learning. As shown in Figure 3, we may have two different structures to learn very deep neural networks under the HOPE framework. In Figure 3 (a), one HOPE model is used as the first layer primarily for feature extraction and a deep neural network is concatenated on top of it as a powerful classifier to form a deep structure. The deep model in Figure 3 (a) may be learned in either supervised or semi-unsupervised mode. In semi-unsupervised learning, the HOPE model is learned based on the maximum likelihood estimation and the upper deep NN is learned in a supervised way. Alternatively, if we have enough labelled data, we may jointly learn both HOPE and DNN in a supervised mode. In Figure 3 (b), we may even stack multiple HOPE models to form another deep model structure. In this case, each HOPE model generates a sensory feature map in each HOPE layer. Just like all pixels in a normal image, all thresholded log-likelihood values in the sensory feature map are also highly correlated, especially for those mixture components locating relatively close by in the feature space. Thus, it makes sense to add another HOPE model on top of it to de-correlate features and perform data modeling at a finer granularity. The deep HOPE model structures in Figure 3 (b) can also be learned in an either supervised or unsupervised mode. In unsupervised learning, these HOPE layers are learned layer-wise using the maximum likelihood estimation. In supervised learning, all HOPE layers are

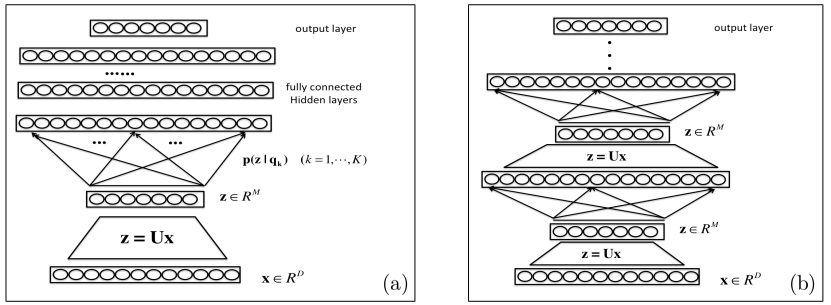


Figure 3: Two structures to learn deep networks with HOPE: (a) Stacking a DNN on top of one HOPE layer; (b) Stacking multiple HOPE layers.

learned by back-propagation with orthonormal constraints being imposed on all projection layers.

6. Experiments

In this section, we will investigate the proposed HOPE framework in learning neural networks for several standard image and speech recognition tasks under several different learning conditions: i) unsupervised feature learning; ii) supervised learning; iii) semi-supervised learning. The examined tasks include the image recognition tasks using the MNIST data set, and the speech recognition task using the TIMIT data set.

6.1 MNIST: Image Recognition

The MNIST data set (LeCun et al., 1998) consists of 28×28 pixel greyscale images of handwritten digits 0-9, with 60,000 training and 10,000 test examples. In our experiments, we first evaluate the performance of unsupervised feature learning using the HOPE model with movMFs. Secondly, we investigate the performance of supervised learning of DNNs under the HOPE framework, and further study the effect of the orthogonal constraint in the HOPE framework. Finally, we consider a semi-supervised learning scenario with the HOPE models, where all training samples (without labels) are used to learn feature representation unsupervised and then a portion of training data (along with labels) is used to learn post-stage classification models in a supervised way. [6]

6.1.1 Unsupervised Feature Learning on MNIST

In this experiment, we first randomly extract many small patches from the original unlabelled training images on MNIST. Each patch is of 6-by-6 in dimension, represented as a vector in $\mathbb{R}^D$, with $D = 36$. In this work, we randomly extract 400,000 patches in total

6. Matlab codes are available at `https://wiki.eecs.yorku.ca/lab/MLL/projects:hope:start` for readers to reproduce all MNIST results reported in this paper.

from the MNIST training set for unsupervised feature learning. Moreover, every patch is normalized by subtracting the mean and being divided by the standard deviation of its elements.

In unsupervised feature learning, we follow the same experimental setting in Coates et al. (2011), where an unsupervised learning algorithm is used to learn a "black box" feature extractor to map each input vector in $\mathbb{R}^D$ to another K-dimension feature vector. In this work, we have examined several different unsupervised learning algorithms for feature learning: (i) kmeans clustering; (ii) spherical kmeans (spkmeans) clustering; (iii) mixture of vMF (movMF), (iv) PCA based dimension reduction plus movMF (PCA-movMF); and (v) the HOPE model with movMFs (HOPE-movMF). As for *kmeans* and *spkmeans*, the only difference is that different distance measures are used in clustering: *kmeans* uses the Euclidean distance while *spkmeans* uses the cosine distance. As for the movMF model, we can use the expectation maximization (EM) algorithm for estimation, as described in Banerjee et al. (2005). In the following, we briefly summarize the experimental details for these feature extractors.

1. **kmeans**: We first apply the k-means clustering method to learn K centroids $\boldsymbol{\mu}_k$ from all extracted patch input vectors. For each learned centroid $\boldsymbol{\mu}_k$, we use a soft threshold function to compute each feature as: $f_k(x) = \max(0, |\mathbf{x} - \boldsymbol{\mu}^k| - \varepsilon)$, where ε is a pre-set threshold. In this way, we may generate a K-dimension feature vector for each patch.

2. **spkmeans**: As for the spk-means clustering, we need to normalize all input patch vectors to be of unit length before clustering them into K different centroids based on the cosine distance measure. Given each learned centroid $\boldsymbol{\mu}_k$, we can compute each feature as $f_k(x) = \max(0, \mathbf{x}^T \boldsymbol{\mu}_k - \varepsilon)$.

3. **movMF**: We also need to normalize all input patch vectors to be of unit length. We use the EM algorithm to learn all model parameters $\boldsymbol{\mu}_k$. For each learned centroid $\boldsymbol{\mu}_k$ of the movMF model, we compute one feature as $f_k(x) = \max\big(0, \ln(\pi_k) + \ln(\mathcal{C}_N(|\boldsymbol{\mu}_k|)) + \mathbf{x}^T \boldsymbol{\mu}_k - \varepsilon\big)$.

4. **PCA-movMF**: Comparing to movMF, the only difference is that we first use PCA for dimensionality reduction, reducing all input patch vectors from $\mathbb{R}^D$ to $\mathbb{R}^M$. Then, we use the same method to estimate an movMF model for the reduced D-dimension feature vectors. In this experiment, we set $M = 20$ to reserve 99.5% of the total sum of all eigenvalues in PCA. For each learned vMF model $\boldsymbol{\mu}_k$, we compute one feature as $f_k(x) = \max\big(0, \ln(\pi_k) + \ln(\mathcal{C}_N(|\boldsymbol{\mu}_k|)) + \mathbf{z}^T \boldsymbol{\mu}_k - \varepsilon\big)$.

5. **HOPE-movMF**: We use the maximum likelihood estimation method described in section 4 to learn a HOPE model with movMFs. For each learned vMF component, we can compute one feature as $f_k(x) = \max(0, \ln(\pi_k) + \ln(\mathcal{C}_M(|\boldsymbol{\mu}_k|)) + \tilde{\mathbf{z}} \cdot \boldsymbol{\mu}_k - \varepsilon)$. Similar to PCA-movMF, we also set $M = 20$ here.

Furthermore, since HOPE-movMF is learned using SGD, we need to tune some hyper-parameters for HOPE, such as the learning rate, mini-batch size, β and σ^2. In this work, the learning rate is set to 0.002, minibatch size is set to 100, we set $\beta = 1.0$, and the noise variance is manually set to $\sigma^2 = 0.1$ for convenience.

model / K=	400	800	1200	1600
kmeans	1.41	1.31	1.16	1.13
spkmeans	1.09	0.90	0.86	0.81
movMF	0.89	0.82	0.81	0.84
PCA-movMF	0.87	0.75	0.73	0.74
HOPE-movMF	0.76	0.71	**0.64**	0.67

Table 1: Classification error rates (in %) on the MNIST test set using supervised learned linear SVMs as classifiers and unsupervised learned features (4K-dimension) from different models.

After learning the above models, they are used as feature extractors. We use the same method as described in Coates et al. (2011) to generate a feature representation for each MNIST image, where each feature extractor is convolving over an MNIST image to obtain the feature representations for all local patches in the image. Next, we split the image into four equally-sized quadrants and the feature vectors in each quadrant are all summed up to pool as one feature vector. In this way, we can get a $4K$-dimensional feature vector for each MNIST image, where K is the number of all learned features for each patch. Finally, we use these pooled $4K$-dimension feature vectors for all training images, along with the labels, to estimate a simple linear SVM as a post-stage classifier for image classification. The experimental results are shown in Table 1. We can see that *spkmeans* and *movMF* can achieve much better performance than *kmeans*. The PCA-based dimension reduction leads to further performance gain. Finally, the jointly trained HOPE model with movMFs yields the best performance, e.g., 0.64% in classification error rate when $K = 1200$. This is a very strong performance for unsupervised feature learning on MNIST.

6.1.2 Supervised Learning of Neural Networks as HOPE on MNIST

In this experiment, we use the MNIST data set to examine the supervised learning of rectified linear (ReLU) neural networks under the HOPE framework, as discussed in section 5.3.

Here we follow the normalized initialization in Glorot and Bengio (2010) to randomly initialize all NN weights, without using pre-training. We adopt a small modification to the method in Glorot and Bengio (2010) by adding a factor to control the dynamic range of initial weights as follows:

$$W_i \sim \left[-\gamma \cdot \frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}, \gamma \cdot \frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}} \right] \tag{41}$$

where n_i denotes the number of units in the i-th layer. For the ReLU units, due to the unbounded behavior of the activation function, activations of ReLU units might grow unbounded. To handle this numerical problem, we shrink the dynamic range of initial weights by using a small factor γ ($\gamma = 0.5$), which is equivalent to scaling the activations.

We use SGD to train ReLU neural networks using the following learning schedule:

$$\epsilon^t = \epsilon_0 \cdot \alpha^t \tag{42}$$

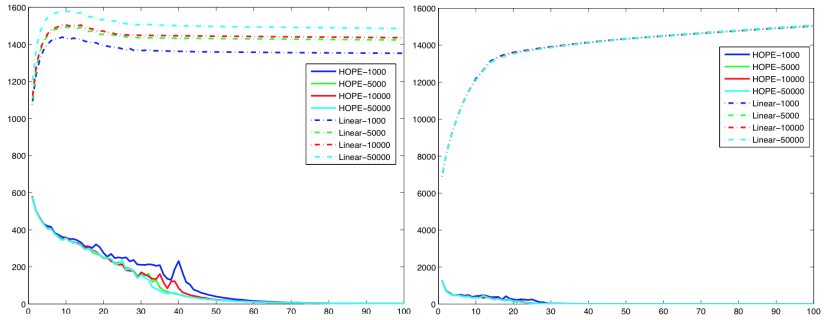


Figure 4: The learning curves of the total sum of correlation coefficients of the linear projection and orthogonal HOPE projection matrices, respectively. left: M=200, K=1k,5k,10k,50k; right: M=400, K=1k,5k,10k,50k.

$$m^t = \begin{cases} \frac{t}{T} m_f + (1 - \frac{t}{T}) m_i & t < T \\ m_f & t \geq T \end{cases} \tag{43}$$

where ϵ^t and m^t denote the learning rate and momentum for the t-th epoch, and we set all control parameters as $m_i = 0.5, m_f = 0.99, \alpha = 0.998$. In total, we run $T = 50$ epochs for learning without dropout and run $T = 500$ epochs for learning with dropout. Moreover, the weight decay is used here and it is set to 0.00001. Furthermore, for the HOPE model, the control parameter for the orthogonal constraint, β, is set to 0.01 in all experiments. In this work, we do not use any data augmentation method.

Under the HOPE framework, we decompose each ReLU hidden layer into two layers as in Figure 1 (a). In this experiment, we first examine the supervised learning of NNs with or without imposing the orthogonal constraints to all projection layers. Firstly, we investigate the performance of a neural network containing only a single hidden layer, decomposed into a pair of a linear projection layer and a nonlinear model layer. Here we evaluate neural networks with a different number of hidden units (K) and a varying size of the projection layer (M). From the experimental results shown in Table. 2, we can see that the HOPE-trained NN can achieve much better performance than the baseline NN, especially when smaller values are used for M. This supports that the projection layer may eliminate residual noises in data to avoid over-fitting when M is properly set. However, after we relax the orthogonal constraint in the HOPE model, as shown in Table 2, the performance of the models using only linear projection layers gets much worse than those of the HOPE models as well as that of the baseline NN. These results verify that orthogonal projection layers are critical in the HOPE models. Furthermore, in Figure. 4, we have plotted the learning curves of the total sum of all correlation coefficients among all row vectors in the learned projection matrix $\mathbf{U}$, i.e., $\sum_{i \neq j} \frac{|\mathbf{u}_i \cdot \mathbf{u}_j|}{|\mathbf{u}_i||\mathbf{u}_j|}$. We can see that all projection vectors tend

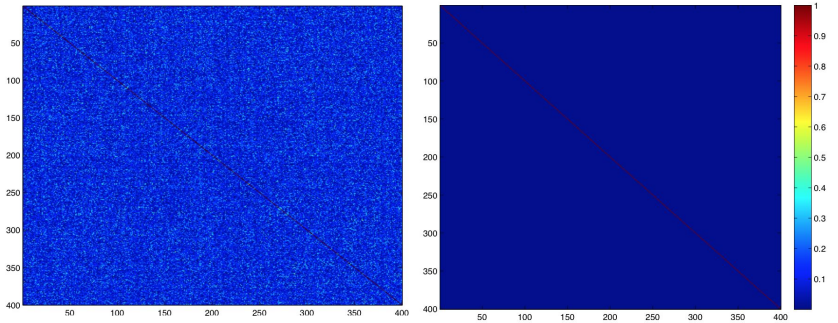


Figure 5: The correlation coefficients of the linear projection matrix (left) and the orthogonal HOPE projection matrix (right) are shown as two images. Here $M = 400$ and $K = 1000$.

Net Architecture / K=	1k	2k	5k	10k	50k
Baseline: 784-K-10	1.49	1.35	1.28	1.30	1.32
HOPE1: 784-[100-K]-10	1.18	1.20	1.17	1.18	1.19
HOPE2: 784-[200-K]-10	1.21	1.20	1.17	1.19	1.18
HOPE3: 784-[400-K]-10	1.19	1.23	1.25	1.25	1.25
Linear1: 784-(100-K)-10	1.45	1.49	1.43	1.45	1.48
Linear2: 784-(200-K)-10	1.52	1.50	1.54	1.55	1.54
Linear3: 784-(400-K)-10	1.53	1.52	1.49	1.52	1.49

Table 2: Classification error rates (in %) on the MNIST test set using a shallow NN with one hidden layer of different hidden units. Neither dropout nor data augmentation is used here for a quick turnaround. Two numbers in brackets, $[M, K]$, indicate a HOPE model with the orthogonal constraint in the projection, and two number in parentheses, (M, K), indicate the same model structure without imposing the orthogonal constraint in the projection.

to get strongly correlated (especially when M is large) in the linear projection matrix as the learning proceeds. On the other hand, the orthogonal constraint can effectively de-correlate all the projection vectors. Moreover, we show all correlation coefficients, i.e., $\frac{|\mathbf{u}_i \cdot \mathbf{u}_j|}{|\mathbf{u}_i||\mathbf{u}_j|}$, of the linear projection matrix and the HOPE orthogonal projection matrix as two images in Figure 5, which clearly shows that the linear projection matrix has many strongly correlated dimensions and the HOPE projection matrix is (as expected) orthogonal.

As the MNIST training set is very small, we use the dropout technique (Hinton et al., 2012; Srivastava et al., 2014) to improve the model learning on the MNIST task. In this experiment, the visible dropout probability is set to 20% and the hidden layer dropout

Net Architecture / K=	1k	2k	5k
NN baseline: 784-K-10	1.05	1.01	1.01
HOPE1: 784-[200-K]-10	0.99	**0.85**	0.89
HOPE2: 784-[400-K]-10	0.86	0.86	**0.85**

Table 3: Comparison of classification error rates (in %) on the MNIST test set between a shallow NN and two HOPE-trained NNs with the same model structure. Dropout is used in this experiment.

model	Net Architecture	without dropout	with dropout
DNN baseline	784-1200-1200-10	1.25	0.92
HOPE + NN	784-[400-1200]-1200-10	0.99	0.82
HOPE*2	784-[400-1200]-[400-1200]-10	0.97	**0.81**

Table 4: Comparison of classification error rates (in %) on the MNIST test set between a 2-hidden-layer DNN and two HOPE-trained DNNs with similar model structures, with or without using dropout.

probability is set to 30%. In Table 3, we compare a 1-hidden-layer shallow NN with two HOPE models (M=200,400). The results show that the HOPE framework can significantly improve supervised learning of NNs. Under the HOPE framework, we can train very simple shallow neural networks from scratch, which can yield comparable performance as deep models. For example, on the MNIST task, as shown in Table 3, we may achieve 0.85% in classification error rate using a shallow neural network (with only one hidden layer of 2000 nodes) trained under the HOPE framework. Furthermore, we consider building even deeper models (two-hidden-layer NNs) under the HOPE framework. Using the two different structures in Figure 3, we can further improve the classification error rate to 0.81%, as shown in Table. 4. This is a very strong result reported on MNIST without using CNNs and data augmentation.

6.1.3 Semi-supervised Learning on MNIST

In this experiment, we combine the unsupervised feature learning with supervised model learning and examine the classification performance when only limited labelled data is available. Here we also list the results using convolutional deep belief networks (CDBN) (Lee et al., 2009) and ladder networks (Rasmus et al., 2015) as two baseline systems for comparison. In our experiments, we use the raw pixel features and unsupervised learned (USL) features in section 6.1.1. As example, we choose the unsupervised learned features from the HOPE-movMF model ($K = 800$) in Table 1.[7] Next, we concatenate the features to a post-stage classifier, which is supervised trained using only a portion of the training data, ranging from 1000 to 60000 (all). We consider many different types of classifiers here,

7. For the HOPE-movMF model with $K = 800$, there are 115 empty clusters. Thus, the unsupervised learned features are of 2740 in dimension.

labeled training samples	1000	2000	5000	10000	20000	60000(ALL)
CDBN(Lee et al., 2009)	2.62	2.13	1.59	-	-	0.82
Ladder network (Rasmus et al., 2015)	1.07	-	-	-	-	0.61
Raw feature+DNN1	8.32	4.71	3.2	2.15	1.48	0.92
Raw feature+HOPE-DNN1	8.22	4.53	2.92	2.04	1.34	0.86
Raw feature+HOPE-DNN2	7.21	4.02	2.60	1.83	1.30	0.82
USL feature+linear SVM	2.91	2.38	1.47	1.13	0.90	0.71
USL feature+DNN2	2.83	1.99	1.03	0.88	0.70	0.43
USL feature+HOPE-DNN3	2.50	1.78	0.95	0.87	0.67	0.42
USL feature+HOPE-DNN4	2.46	1.70	0.90	0.79	0.66	**0.40**

Table 5: Classification error rates (in %) on the MNIST test set using raw pixel features or unsupervised learned features, along with different post-stage classifiers trained separately by limited labeled data. Here USL denotes the unsupervised learned features from HOPE-movMF ($K = 800$). All used classifiers include: DNN1 (784-1200-1200-10); HOPE-DNN1 (784-[400-1200]-10); HOPE-DNN2 (784-[400-1200]-1200-10); DNN2 (2740-1200-1200-10); HOPE-DNN3 (2740-[400-1200]-10); HOPE-DNN4 (784-[400-1200]-1200-10).

including linear SVM, regular DNNs and HOPE-trained DNNs. Note that all classifiers are trained separately from the feature learning. All results are summarized in Table 5. It shows that we can achieve the best performance when we combine the HOPE-trained USL features with HOPE-trained post-stage classifiers. The gains are quite dramatic no matter how much labelled data is used. For example, when only 5000 labelled samples are used, our method can achieve 0.90% in error rate, which significantly outperforms all other methods including CDBN in Lee et al. (2009). The ladder network in (Rasmus et al., 2015) yields better performance when only 1000 labels are used. However, Rasmus et al. (2015) adopts a much larger model than ours. It involves three different modules of distortion, denoising and classification. Each module uses a 6-hidden-layer structure. Moreover, in our semi-supervised experiments, we do not fine-tune the feature extraction module with any supervision signals from labels. Finally, as we use all training data for the HOPE model, we can achieve 0.40% in error rate, which significantly outperforms both CDBN and ladder networks. To the best of our knowledge, this is one of the best results reported on MNIST without using CNNs and data augmentation. Furthermore, our best system uses a quite simple model structure, consisting of a HOPE-trained feature extraction layer of 800 nodes and a HOPE-trained NN of two hidden layers (1200 node in each layer), which is much smaller and simpler than those top-performing systems on MNIST.

6.2 TIMIT: Speech Recognition

In this experiment, we examine the supervised learning of shallow and deep neural networks under the HOPE framework for a standard speech recognition task using the TIMIT data set. The HOPE-based supervised learning method is compared with the regular back-propagation training method. We use the minimum cross-entropy learning criterion here.

The TIMIT speech corpus consists of a training set of 462 speakers, a separate development set of 50 speakers for cross-validation, and a core test set of 24 speakers. All results are reported on the 24-speaker core test set. The speech waveform data is analyzed using a 25-ms Hamming window with a 10-ms fixed frame rate. The speech feature vector is generated by a Fourier-transform-based filter-banks that include 40 coefficients distributed on the Mel scale and energy, along with their first and second temporal derivatives. This leads to a 123-dimension feature vector per speech frame. We further concatenate 15 consecutive frames within a long context window of (7+1+7) to feed to the models, as 1845-dimension input vectors (Xue et al., 2014). All speech data are normalized by subtracting the mean of the training set and being divided by the standard deviation of the training set on each dimension so that all input vectors have zero mean and unit variance. We use 183 target class labels (i.e., 3 states for each of the 61 phones) for the DNN training. After decoding, these 61 phone classes are mapped to a set of 39 classes for the final scoring as in Lee and Hon (1989). In our experiments, a bi-gram language model at phone level, estimated from all transcripts in the training set, is used for speech recognition.

We first train ReLUs based shallow and deep neural networks as our baseline systems. The networks are trained using the back-propagation algorithm, with a mini-batch size of 100. The initial learn rate is set to 0.004 and it is kept unchanged if the error rate on the development set is still decreasing. Afterwards, the learning rate is halved after every epoch, and the whole training procedure is stopped when the error reduction on the development set is less than 0.1% in two consecutive iterations. In our experiments, we also use momentum and weight decay, which are set to 0.9 and 0.0001, respectively. When we use the mini-batch SGD to train neural networks under the HOPE framework, the control parameter for the orthogonal constraints, i.e. β, is set to be 0.01.

In our experiments, we compare the standard NNs with the HOPE-trained NNs for two fully-connected network architectures, one shallow network with one hidden layer of 10240 hidden nodes and one deep network with 3 hidden layers of 2048 nodes. The performance comparison between them is shown in Table. 6. Our results are comparable with Xue et al. (2014) and another recent work (Ba and Caruana, 2014), using deep neural networks on TIMIT. From the results, we can see that the HOPE-trained NNs can consistently outperform the regular NNs by an about 0.8% absolute reduction in phone recognition error rates. Moreover, the HOPE-trained neural networks are much smaller than their counterpart DNNs in number of model parameters if the HOPE layers are not merged. After merging, they have exactly the same model structure as their counterpart NNs.

7. Conclusions

In this paper, we have proposed a novel model, called hybrid orthogonal projection and estimation (HOPE), for high-dimensional data. The HOPE model combines feature extraction and data modeling under a unified generative modeling framework so that both feature extractor and data model can be jointly learned in either supervised or unsupervised ways. More interestingly, we have shown that the HOPE models are closely related to neural networks in a way that each hidden layer in NNs can be reformulated as a HOPE model. Therefore, the proposed HOPE related learning algorithms can be easily applied to perform either supervised or unsupervised learning for neural networks. We have evaluated the proposed HOPE models in learning NNs on several standard tasks, including image recognition on MNIST and speech recognition on TIMIT. Experimental results have strongly supported that the HOPE models can provide a very effective unsupervised learning method for NNs. Meanwhile, the supervised learning of NNs can also be conducted under the HOPE framework, which normally yields better performance and more compact models.

As the future work, we will investigate the HOPE model to learn convolution neural networks (CNNs) for more challenging image recognition tasks, such as CIFAR and ImageNet. Moreover, we are also examining the HOPE-based unsupervised learning for various natural language processing (NLP) tasks.

model	Net Architecture	FACC (%)	PER (%)
NN	1845-10240-183	61.45	23.85
HOPE-NN	1845-[256-10240]-183	62.11	**23.04**
DNN	1845-3*2048-183	63.13	22.37
HOPE-DNN	1845-[512-2048]-2*2048-183	63.55	**21.59**

Table 6: Supervised learning of neural networks on TIMIT with and without using the HOPE framework. The two numbers in a bracket, $[M, K]$, indicate a HOPE model with M-dimension features and K mixture components. FACC: frame classification accuracy from neural networks. PER: phone error rate in speech recognition.

Acknowledgments

We acknowledge the support of the following organizations for research funding: National Nature Science Foundation of China (Grant No. 61273264), Science and Technology Department of Anhui Province (Grant No. 15CZZ02007), Chinese Academy of Sciences (Grant No. XDB02070006), an NSERC discovery grant from Canadian Federal Government and a research donation from iFLYTEK Co., Hefei, China.

Appendix A. Learning HOPE when $\hat{\mathbf{U}}$ is not orthonormal

In some tasks, in additional to dimension reduction, we may want to use the projection matrix $\mathbf{U}$ to perform signal whitening to ensure the signal projection $\mathbf{z}$ has roughly the same variance along all dimensions. This has been shown to be quite important for many image recognition and speech recognition tasks. In this case, we may still want to impose the orthogonal constraints among all row vectors of $\mathbf{U}$, i.e., $\mathbf{u}_i \cdot \mathbf{u}_j = 0 \ \ (i, j = 1, \cdots, M, \ i \neq j)$, but these row vectors may not be of unit length, i.e., $|\mathbf{u}_i| \geq 1 \ \ (i = 1, \cdots, M)$. Moreover, it is better not to whiten the residual noises in the remaining $D - M$ dimensions to unnecessarily amplify them. Therefore, we still enforce the unit-length constraints for the matrix $\mathbf{V}$, i.e., $|\mathbf{u}_i| = 1$ $(i = M + 1, \cdots, D)$. Because $\mathbf{U}$ is not orthonormal anymore, when we compute the likelihood function of the original data in eq.(11) for HOPE, we have to include the Jacobian term as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{U},\mathbf{\Theta},\sigma \mid \mathbf{X}) &= \sum_{n=1}^{N} \ln \Pr(\mathbf{x}_n) = \sum_{n=1}^{N} \Big[\ln|\hat{\mathbf{U}}^{-1}| + \ln \Pr(\mathbf{z}_n) + \ln \Pr(\mathbf{n}_n) \Big] \\ = \underbrace{N \cdot \ln|\hat{\mathbf{U}}^{-1}|}_{\mathcal{J}(\mathbf{U})} &+ \underbrace{\sum_{n=1}^{N} \ln\left(\sum_{k=1}^{K} \pi_k \cdot f_k(\mathbf{U}\mathbf{x}_n|\boldsymbol{\theta}_k) \right)}_{\mathcal{L}_1(\mathbf{U},\mathbf{\Theta})} + \underbrace{\sum_{n=1}^{N} \ln\Big(\mathcal{N}\big(\mathbf{n}_n \,|\, \mathbf{0}, \sigma^2\mathbf{I}\big) \Big)}_{\mathcal{L}_2(\mathbf{U},\sigma)}\end{aligned} \tag{44}$$

Because $\mathbf{U}\mathbf{U}^T \neq \mathbf{I}$ here, we have

$$\mathbf{x}_n = \mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{z}_n + \mathbf{V}^T\mathbf{n}_n \tag{45}$$

and then we have

$$\begin{aligned}\mathbf{x}_n^T\mathbf{x}_n &= \Big(\mathbf{n}_n^T\mathbf{V} + \mathbf{z}_n^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\Big)\Big(\mathbf{V}^T\mathbf{n}_n + \mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{z}_n\Big) \\ &= \mathbf{n}_n^T\mathbf{n}_n + \mathbf{z}_n^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{z}_n = \mathbf{n}_n^T\mathbf{n}_n + \mathbf{x}_n^T\mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\mathbf{x}_n\end{aligned} \tag{46}$$

Therefore, we may derive the residual noise energy as:

$$\begin{aligned}\mathbf{n}_n^T\mathbf{n}_n &= \mathbf{x}_n^T\mathbf{x}_n - \mathbf{x}_n^T\mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\mathbf{x}_n \\ &= \mathbf{x}_n^T\Big[\mathbf{I} - \mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\Big]\mathbf{x}_n\end{aligned} \tag{47}$$

In this case, $\mathcal{L}_2(\mathbf{U},\sigma)$ can be expressed as:

$$\mathcal{L}_2(\mathbf{U},\sigma) = -\frac{N}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}\Big[\mathbf{x}_n^T\mathbf{x}_n - \mathbf{x}_n^T\mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\mathbf{x}_n\Big] \tag{48}$$

Therefore, its gradient with respect to $\mathbf{U}$, can be derived as follows:

$$\begin{aligned}\frac{\partial\mathcal{L}_2(\mathbf{U},\sigma)}{\partial\mathbf{U}} &= \frac{1}{\sigma^2}\sum_{n=1}^{N}\Big[(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\mathbf{x}_n\mathbf{x}_n^T - (\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\mathbf{x}_n\mathbf{x}_n^T\mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\Big] \\ &= \frac{1}{\sigma^2}\sum_{n=1}^{N}(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\mathbf{x}_n\mathbf{x}_n^T\Big[\mathbf{I} - \mathbf{U}^T(\mathbf{U}\mathbf{U}^T)^{-1}\mathbf{U}\Big]\end{aligned} \tag{49}$$

Next, we consider the Jacobian term, $\mathcal{J}(\mathbf{U})$, which can be computed as follows:

$$\mathcal{J}(\mathbf{U}) = N\cdot\ln|\hat{\mathbf{U}}^{-1}| = -N\sum_{i=1}^{M}\ln|\mathbf{u}_i| \tag{50}$$

Because of $\frac{\partial\mathcal{J}(\mathbf{U})}{\partial\mathbf{u}_i} = -\frac{N\cdot\mathbf{u}_i}{|\mathbf{u}_i|^2}$ $(i=1,\cdots,M)$, it is easy to show that its derivative with respect to $\mathbf{U}$ can be derived as:

$$\frac{\partial\mathcal{J}(\mathbf{U})}{\partial\mathbf{U}} = -N\cdot\big(\mathbf{U}\mathbf{U}^T\big)^{-1}\mathbf{U}. \tag{51}$$

Similarly, the HOPE parameters, i.e., $\mathbf{U}$, $\mathbf{\Theta}$ and σ, can be estimated by maximizing the above likelihood function as follows:

$$\{\mathbf{U}^*,\mathbf{\Theta}^*,\sigma^*\} = \arg\max_{\mathbf{U},\mathbf{\Theta},\sigma}\ \ \mathcal{L}(\mathbf{U},\mathbf{\Theta},\sigma \mid \mathbf{X}) \tag{52}$$

subject to an orthogonal constraint:

$$\mathbf{U}\mathbf{U}^T = \mathbf{\Phi}, \tag{53}$$

where $\mathbf{\Phi}$ is a diagonal matrix. The above constraint can also be implemented as an penalty term similar to eq.(15). However, the norm of each row vector is relaxed as follows:

$$|\mathbf{u}_i| \geq 1 \quad (i=1,\cdots,M) \tag{54}$$

The log-likelihood function related to the signal model, $\mathcal{L}_1(\mathbf{U},\mathbf{\Theta})$, and signal variance, σ^2, are calculated in the same way as before.

Appendix B. Derivatives of movMFs

The partial derivatives of the objective function in eq.(29) w.r.t all $\boldsymbol{\mu}_k$ can be computed as follows:

$$\frac{\partial\mathcal{L}_1(\mathbf{U},\mathbf{\Theta})}{\partial\boldsymbol{\mu}_k} = \sum_{n=1}^{N}\frac{\pi_k\cdot\Big(\mathcal{C}_M'(|\boldsymbol{\mu}_k|)\frac{\partial\kappa_k}{\boldsymbol{\mu}_k}\cdot e^{\mathbf{z}_n^T\cdot\boldsymbol{\mu}_k} + \mathcal{C}_M(|\boldsymbol{\mu}_k|)\cdot e^{\mathbf{z}_n^T\cdot\boldsymbol{\mu}_k}\cdot\mathbf{z}_n\Big)}{\sum_{j=1}^{K}\pi_j\cdot\mathcal{C}_M(|\boldsymbol{\mu}_j|)\cdot e^{\mathbf{z}\cdot\boldsymbol{\mu}_j}} \tag{55}$$

where we have

$$\frac{\partial\,|\boldsymbol{\mu}_k|}{\partial\boldsymbol{\mu}_k} = \frac{\boldsymbol{\mu}_k}{|\boldsymbol{\mu}_k|}. \tag{56}$$

As to $\mathcal{C}_M'(\kappa)$, for brevity, let us denote $s=\frac{M}{2}-1$, and $\xi=(2\pi)^{s+1}$.

$$\begin{aligned}\mathcal{C}_M'(\kappa) &= \frac{s\cdot\kappa^{s-1}}{\xi I_s(\kappa)} - \frac{\kappa^s\cdot I'_s(\kappa)}{\xi {I_s}^2(\kappa)} = \frac{\kappa^s}{\xi I_s(\kappa)}\left(\frac{s}{\kappa} - \frac{I'_s(\kappa)}{I_s(\kappa)}\right) \\ &= \mathcal{C}_M(\kappa)\cdot\left(\frac{s}{\kappa} - \frac{I'_s(\kappa)}{I_s(\kappa)}\right)\end{aligned} \tag{57}$$

where $I_v(\cdot)$ denotes the Bessel function of the first kind at the order v. Because we have

$$\kappa I_{s+1}(\kappa) = \kappa I'_s(\kappa) - sI_s(\kappa) \Rightarrow \frac{s}{\kappa} - \frac{I'_s(\kappa)}{I_s(\kappa)} = -\frac{I_{s+1}(\kappa)}{I_s(\kappa)}$$

Thus, we may derive

$$\mathcal{C}_M'(\kappa) = -\mathcal{C}_M(\kappa)\cdot\frac{I_{s+1}(\kappa)}{I_s(\kappa)} \tag{58}$$

Substituting eq. (56) and eq. (58) into eq. (55), we obtain the partial derivative of the objective function in eq. (29) w.r.t $\boldsymbol{\mu}_k$ as follows:

$$\begin{aligned}\frac{\partial\mathcal{L}_1(\mathbf{U},\mathbf{\Theta})}{\partial\boldsymbol{\mu}_k} &= \sum_{n=1}^{N}\frac{\pi_k\cdot\mathcal{C}_M(|\boldsymbol{\mu}_k|)\cdot e^{\mathbf{z}_n\cdot\boldsymbol{\mu}_k}\left(\mathbf{z}_n - \frac{\boldsymbol{\mu}_k}{|\boldsymbol{\mu}_k|}\cdot\frac{I_{M/2}(|\boldsymbol{\mu}_k|)}{I_{M/2-1}(|\boldsymbol{\mu}_k|)}\right)}{\sum_{j=1}^{K}\pi_j\cdot\mathcal{C}_{\mathcal{M}}(|\boldsymbol{\mu}_j|)\cdot e^{\mathbf{z}_n\cdot\boldsymbol{\mu}_j}} \\ &= \sum_{n=1}^{N}\gamma(z_{nk})\cdot\left(\mathbf{z}_n - \frac{\boldsymbol{\mu}_k}{|\boldsymbol{\mu}_k|}\cdot\frac{I_{M/2}(|\boldsymbol{\mu}_k|)}{I_{M/2-1}(|\boldsymbol{\mu}_k|)}\right)\end{aligned} \tag{59}$$

where $\gamma(z_{nk}) = \frac{\pi_k \cdot \mathcal{C}_M(|\boldsymbol{\mu}_k|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_k}}{\sum_{j=1}^K \pi_j \cdot \mathcal{C}_M(|\boldsymbol{\mu}_j|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_j}}$ is the occupancy statistics of k-th component of $\mathbf{z}_n$.

Next, let us consider the partial derivative of the objective function in eq.(29) w.r.t $\mathbf{U}$. Based on the chain rule, we have

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{U}} = \frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \widetilde{\mathbf{z}}_n} \cdot \frac{\partial \widetilde{\mathbf{z}}_n}{\partial \mathbf{U}} = \left(\frac{\partial \mathbf{z}_n^T}{\partial \widetilde{\mathbf{z}}_n} \cdot \frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{z}_n} \right) \cdot \frac{\partial \widetilde{\mathbf{z}}_n}{\partial \mathbf{U}} \tag{60}$$

Furthermore, we may derive

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{z}_n} = \sum_{n=1}^N \sum_{k=1}^K \frac{\pi_k \cdot \mathcal{C}_M(|\boldsymbol{\mu}_k|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_k}}{\sum_{j=1}^K \pi_j \cdot \mathcal{C}_M(|\boldsymbol{\mu}_j|) \cdot e^{\mathbf{z}_n \cdot \boldsymbol{\mu}_j}} \cdot \boldsymbol{\mu}_k = \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) \cdot \boldsymbol{\mu}_k \tag{61}$$

$$\begin{aligned} \frac{\partial \mathbf{z}_n^T}{\partial \tilde{\mathbf{z}}_n} &= \frac{\partial (\tilde{\mathbf{z}}_n^T / |\tilde{\mathbf{z}}_n|)}{\partial \tilde{\mathbf{z}}_n} = \frac{1}{|\tilde{\mathbf{z}}_n|^2} \left(\frac{\partial \tilde{\mathbf{z}}_n^T}{\partial \tilde{\mathbf{z}}_n} |\tilde{\mathbf{z}}_n| - \frac{\partial |\tilde{\mathbf{z}}_n|}{\partial \tilde{\mathbf{z}}_n} \tilde{\mathbf{z}}_n^T \right) \\ &= \frac{1}{|\tilde{\mathbf{z}}_n|} \left(\mathbf{I} - \frac{\tilde{\mathbf{z}}_n \tilde{\mathbf{z}}_n^T}{|\tilde{\mathbf{z}}_n|^2} \right) = \frac{1}{|\tilde{\mathbf{z}}_n|} \left(\mathbf{I} - \mathbf{z}_n \mathbf{z}_n^T \right) \end{aligned} \tag{62}$$

Substituting eq.(61) and eq. (62) into eq. (60), we can obtain

$$\frac{\partial \mathcal{L}_1(\mathbf{U}, \boldsymbol{\Theta})}{\partial \mathbf{U}} = \sum_{n=1}^N \sum_{k=1}^K \frac{\gamma(z_{nk})}{|\tilde{\mathbf{z}}_n|} (\mathbf{I} - \mathbf{z}_n \mathbf{z}_n^T) \boldsymbol{\mu}_k \mathbf{x}_n^T \tag{63}$$

Appendix C. Numerical Methods for $I_v(\cdot)$

In the learning algorithm for movMFs, we may need to compute the Bessel functions, $I_v(\cdot)$, in several places. First of all, we need to compute the normalization term $\mathcal{C}_M(\kappa)$ when calculating the likelihood function of a vMF distribution as in eq.(9). Secondly, we need to calculate the rations of the modified Bessel functions, $A_d(\kappa) = \frac{I_{M/2}(\kappa)}{I_{M/2-1}(\kappa)}$, in eq.(59). As we know, the modified Bessel functions of the first kind take the following form:

$$I_d(\kappa) = \sum_{k \geq 0} \frac{1}{\Gamma(d+1+k)k!} (\frac{\kappa}{2})^{2k+d}, \tag{64}$$

From eq. (64), we can see that when $\kappa \gg d$, $I_d(\kappa)$ overflows quite rapidly. Meanwhile, when $\kappa = o(d)$ and $d \to \infty$, $I_d(\kappa)$ underflows quite rapidly. In this work, we use the approximation strategy in eq. (9.7.7) on page 378 of Abramowitz and Stegun (2006) as follows:

$$I_d(\kappa) \sim \frac{1}{\sqrt{2\pi d}} \cdot \frac{e^{d\eta}}{(1 + (\frac{\kappa}{d})^2)^{1/4}} \cdot \left[1 + \sum_{k=1}^{\infty} \frac{u_k(t)}{d^k} \right] \tag{65}$$

where we have

$$t = \frac{1}{\sqrt{1 + (\kappa/d)^2}} \tag{66}$$

$$\eta = \sqrt{1 + (\kappa/d)^2} + \ln \frac{\kappa/d}{1 + \sqrt{1 + (\kappa/d)^2}} \tag{67}$$

with the functions $u_k(t)$ taking the following forms:

$$\begin{aligned} u_0(t) &= 1 \\ u_1(t) &= (3t - 5t^3)/24 \\ u_2(t) &= (81t^2 - 462t^4 + 385t^6)/1152 \end{aligned}$$

Refer to page 366 in Abramowitz and Stegun (2006) for other higher orders $u_k(t)$.

Usually, the sum of the term $[1 + \sum_{k=1}^{\infty} \frac{u_k(t)}{d^k}]$ in eq.(65) is very small and it is safe to eliminate it from evaluation in most cases. Then, after substituting eq.(66) and eq.(67) into eq.(65), the logarithm of the approximated modified Bessel function is finally computed as follows:

$$\ln I_d(\kappa) = -\ln \sqrt{2\pi d} + d \cdot \left(\sqrt{1 + (\kappa/d)^2} + \ln \frac{\kappa}{d} - \ln \left(1 + \sqrt{1 + (\kappa/d)^2} \right) \right) - \frac{1}{4} \ln \left(1 + (\frac{\kappa}{d})^2 \right) \tag{68}$$

In this work, the approximation in eq.(68) is used to compute all Bessel functions in the learning algorithms for movMFs.

References

M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Washington : U.S. Govt. Print., 2006.

H. Attias. Independent factor analysis. *Neural Computation*, 11(4):803–851, 1999.

J. Ba and R. Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems (NIPS) 27*, pages 2654–2662, 2014.

A. Banerjee, I. S. Dhillon, and J. Ghosh. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.

Y. Bao, H. Jiang, C. Liu, Y. Hu, and L. Dai. Investigation on dimensionality reduction of concatenated features with deep neural network for lvcsr systems. In *IEEE International Conference on Signal Processing (ICSP)*, volume 1, pages 562–566, 2012.

Y. Bao, H. Jiang, L. Dai, and C. Liu. Incoherent training of deep neural networks to decorrelate bottleneck features for speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems (NIPS) 19*. MIT Press, 2007.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning (edited by O. Bousquet and U. von Luxburg)*, pages 146–168. Springer Verlag, 2004.

A. Coates, A. Y. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

L. Dinh, D. Krueger, and Y. Bengio. NICE: Non-linear independent components estimation. In *International Conference on Learning Representations (ICLR)*, 2015.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

G. E. Hinton and S. J. Nowlan. The bootstrap Widrow-Hoff rule as a cluster-formation algorithm. *Neural Computation*, 2(3):355–362, 1990.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

G. E. Hinton, P. Dayan, and M. Revow. Modelling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, 8(1):65–74, 1997.

G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18:1527–1554, 2006.

G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. In *preprint arXiv 1207.0580*, 2012.

A. Hyvarinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13:411–430, 2000.

H. Jiang. Discriminative training for automatic speech recognition: A survey. *Computer and Speech, Language*, 24(4):589–608, 2010.

H. Jiang and X. Li. Parameter estimation of statistical models using convex optimization: An advanced method of discriminative training for speech and language processing. *IEEE Signal Processing Magazine*, 27(3):115–127, 2010.

H. Jiang, Z. Pan, and P. Hu. Discriminative learning of generative models: large margin multinomial mixture models for document classification. *Pattern Analysis and Applications*, 18(3):535–551, 2015.

N. Kambhatla and T. K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

N. Kumar and A. G. Andreou. Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition. *Speech Communication*, 26(4):283–297, 1998.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

H. Lee, R. Grosse, R. Ranganath, and A. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning (ICML)*, pages 609–616, 2009.

K.-L. Lee and H.-W. Hon. Speaker-independent phone recognition using hidden Markov models. *IEEE Trans. Acoust., Speech, Signal Process.*, 37(11):1641–1648, 1989.

A. Oord and J. Dambre. Locally-connected transformations for deep GMMs. In *Deep Learning Workshop at ICML*, 2015.

J. Pan, C. Liu, Z. Wang, Y. Hu, and H. Jiang. Investigations of deep neural networks for large vocabulary continuous speech recognition: Why DNN surpasses GMMs in acoustic modelling. In *International Symposium on Chinese Spoken Language Processing*, 2012.

J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.

A. Rasmus, H. Valpola, M. Honkala, M. Berglund, and T. Raiko. Semi-supervised learning with ladder network. In *preprint arXiv 1507.02672*, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, pages 1278–1286, 2014.

E. T. Rolls and A. Treves. *Neural Networks and Brain Function*. Oxford University Press, 1998.

S. Roweis. EM algorithms for PCA and SPCA. In *Advances in Neural Information Processing Systems (NIPS) 10*, pages 626–632, 1998.

T. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *IEEE International Conference on Acoustic, Speech, Signal Processing (ICASSP)*, 2013.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

M. E. Tipping and C. M. Bishop. Probabilistic principle component analysis. *Journal of the Royal Statistical Society*, Series B 21(3):611–622, 1999a.

M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482, 1999b.

P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning (ICML)*, pages 1096–1103, 2008.

J. Xue, J. Li, and Y. Gong. Restructuring of deep neural network acoustic models with singular value decomposition. In *Interspeech*, 2013.

S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu. Fast adaptation of deep neural network based on discriminant codes for speech recognition. *IEEE/ACM Trans. on Audio, Speech and Language Processing*, 22(12):1713–1725, 2014.

S. Zhang and H. Jiang. Hybrid orthogonal projection and estimation (HOPE): A new framework to probe and learn neural networks. In *preprint arXiv:1502.00702*, 2015.

S. Zhang, L. Dai, and H. Jiang. The new HOPE way to learn neural networks. In *Deep Learning Workshop at ICML*, 2015.

Journal of Machine Learning Research 17 (2016) 1-15 Submitted 7/15; Revised 1/16; Published 4/16

The Optimal Sample Complexity of PAC Learning

Steve Hanneke STEVE.HANNEKE@GMAIL.COM

Editor: John Shawe-Taylor

Abstract

This work establishes a new upper bound on the number of samples sufficient for PAC learning in the realizable case. The bound matches known lower bounds up to numerical constant factors. This solves a long-standing open problem on the sample complexity of PAC learning. The technique and analysis build on a recent breakthrough by Hans Simon.

Keywords: sample complexity, PAC learning, statistical learning theory, minimax analysis, learning algorithm

1. Introduction

Probably approximately correct learning (or *PAC* learning; Valiant, 1984) is a classic criterion for supervised learning, which has been the focus of much research in the past three decades. The objective in PAC learning is to produce a classifier that, with probability at least $1-\delta$, has error rate at most ε. To qualify as a PAC learning algorithm, it must satisfy this guarantee for all possible target concepts in a given family, under all possible data distributions. To achieve this objective, the learning algorithm is supplied with a number m of i.i.d. training samples (data points), along with the corresponding correct classifications. One of the central questions in the study of PAC learning is determining the minimum number $\mathcal{M}(\varepsilon,\delta)$ of training samples necessary and sufficient such that there exists a PAC learning algorithm requiring at most $\mathcal{M}(\varepsilon,\delta)$ samples (for any given ε and δ). This quantity $\mathcal{M}(\varepsilon,\delta)$ is known as the *sample complexity*.

Determining the sample complexity of PAC learning is a long-standing open problem. There have been upper and lower bounds established for decades, but they differ by a logarithmic factor. It has been widely believed that this logarithmic factor can be removed for certain well-designed learning algorithms, and attempting to prove this has been the subject of much effort. Simon (2015) has very recently made an enormous leap forward toward resolving this issue. That work proposed an algorithm that classifies points based on a majority vote among classifiers trained on independent data sets. Simon proves that this algorithm achieves a sample complexity that reduces the logarithmic factor in the upper bound down to a very slowly-growing function. However, that work does not quite completely resolve the gap, so that determining the optimal sample complexity remains open.

The present work resolves this problem by completely eliminating the logarithmic factor. The algorithm achieving this new bound is also based on a majority vote of classifiers. However, unlike Simon's algorithm, here the voting classifiers are trained on data subsets specified by a recursive algorithm, with substantial overlaps among the data subsets the classifiers are trained on.

2. Notation

We begin by introducing some basic notation essential to the discussion. Fix a nonempty set $\mathcal{X}$, called the *instance space*; we suppose $\mathcal{X}$ is equipped with a σ-algebra, defining the measurable subsets of $\mathcal{X}$. Also denote $\mathcal{Y}=\{-1,+1\}$, called the *label space*. A *classifier* is any measurable function $h:\mathcal{X}\to\mathcal{Y}$. Fix a nonempty set $\mathbb{C}$ of classifiers, called the *concept space*. To focus the discussion on nontrivial cases,[1] we suppose $|\mathbb{C}|\geq 3$; other than this, the results in this article will be valid for *any* choice of $\mathbb{C}$.

In the learning problem, there is a probability measure $\mathcal{P}$ over $\mathcal{X}$, called the *data distribution*, and a sequence $X_1(\mathcal{P}), X_2(\mathcal{P}),\ldots$ of independent $\mathcal{P}$-distributed random variables, called the *unlabeled data*; for $m\in\mathbb{N}$, also define $\mathbb{X}_{1:m}(\mathcal{P})=\{X_1(\mathcal{P}),\ldots,X_m(\mathcal{P})\}$, and for completeness denote $\mathbb{X}_{1:0}(\mathcal{P})=\{\}$. There is also a special element of $\mathbb{C}$, denoted $f^\star$, called the *target function*. For any sequence $S_x=\{x_1,\ldots,x_k\}$ in $\mathcal{X}$, denote by $(S_x,f^\star(S_x))=\{(x_1,f^\star(x_1)),\ldots,(x_k,f^\star(x_k))\}$. For any probability measure P over $\mathcal{X}$, and any classifier h, denote by $\mathrm{er}_P(h;f^\star)=P(x:h(x)\neq f^\star(x))$. A *learning algorithm* $\mathcal{A}$ is a map,[2] mapping any sequence $\{(x_1,y_1),\ldots,(x_m,y_m)\}$ in $\mathcal{X}\times\mathcal{Y}$ (called a *data set*), of any length $m\in\mathbb{N}\cup\{0\}$, to a classifier $h:\mathcal{X}\to\mathcal{Y}$ (not necessarily in $\mathbb{C}$).

Definition 1 *For any $\varepsilon,\delta\in(0,1)$, the* sample complexity of (ε,δ)-PAC learning, *denoted $\mathcal{M}(\varepsilon,\delta)$, is defined as the smallest $m\in\mathbb{N}\cup\{0\}$ for which there exists a learning algorithm $\mathcal{A}$ such that, for every possible data distribution $\mathcal{P}$, $\forall f^\star\in\mathbb{C}$, denoting $\hat{h}=\mathcal{A}((\mathbb{X}_{1:m}(\mathcal{P}),f^\star(\mathbb{X}_{1:m}(\mathcal{P}))))$,*

$$\mathbb{P}\left(\mathrm{er}_{\mathcal{P}}\left(\hat{h};f^\star\right)\leq\varepsilon\right)\geq 1-\delta.$$

If no such m exists, define $\mathcal{M}(\varepsilon,\delta)=\infty$.

The sample complexity is our primary object of study in this work. We require a few additional definitions before proceeding. Throughout, we use a natural extension of set notation to sequences: for any finite sequences $\{a_i\}_{i=1}^k,\{b_i\}_{i=1}^{k'}$, we denote by $\{a_i\}_{i=1}^k\cup\{b_i\}_{i=1}^{k'}$ the concatenated sequence $\{a_1,\ldots,a_k,b_1,\ldots,b_{k'}\}$. For any set A, we denote by $\{a_i\}_{i=1}^k\cap A$ the subsequence comprised of all a_i for which $a_i\in A$. Additionally, we write $b\in\{b_i\}_{i=1}^{k'}$ to indicate $\exists i\leq k'$ s.t. $b_i=b$, and we write $\{a_i\}_{i=1}^k\subseteq\{b_i\}_{i=1}^{k'}$ or $\{b_i\}_{i=1}^{k'}\supseteq\{a_i\}_{i=1}^k$ to express that $a_j\in\{b_i\}_{i=1}^{k'}$ for every $j\leq k$. We also denote $|\{a_i\}_{i=1}^k|=k$ (the length of the sequence). For any $k\in\mathbb{N}\cup\{0\}$ and any sequence $S=\{(x_1,y_1),\ldots,(x_k,y_k)\}$ of points in $\mathcal{X}\times\mathcal{Y}$, denote $\mathbb{C}[S]=\{h\in\mathbb{C}:\forall(x,y)\in S,h(x)=y\}$, referred to as the set of classifiers *consistent* with S.

Following Vapnik and Chervonenkis (1971), we say a sequence $\{x_1,\ldots,x_k\}$ of points in $\mathcal{X}$ is *shattered* by $\mathbb{C}$ if $\forall y_1,\ldots,y_k\in\mathcal{Y}$, $\exists h\in\mathbb{C}$ such that $\forall i\in\{1,\ldots,k\}$, $h(x_i)=y_i$: that is, there are 2^k distinct classifications of $\{x_1,\ldots,x_k\}$ realized by classifiers in $\mathbb{C}$. The Vapnik-Chervonenkis dimension (or *VC dimension*) of $\mathbb{C}$ is then defined as the largest

1. The sample complexities for $|\mathbb{C}|=1$ and $|\mathbb{C}|=2$ are already quite well understood in the literature, the former having sample complexity 0, and the latter having sample complexity either 1 or $\Theta(\frac{1}{\varepsilon}\ln\frac{1}{\delta})$ (depending on whether the two classifiers are exact complements or not).
2. We also admit randomized algorithms $\mathcal{A}$, where the "internal randomness" of $\mathcal{A}$ is assumed to be independent of the data. Formally, there is a random variable R independent of $\{X_i(P)\}_{i,P}$ such that the value $\mathcal{A}(S)$ is determined by the input data S and the value of R.

integer k for which there exists a sequence $\{x_1, \ldots, x_k\}$ in $\mathcal{X}$ shattered by $\mathbb{C}$; if no such largest k exists, the VC dimension is said to be infinite. We denote by d the VC dimension of $\mathbb{C}$. This quantity is of fundamental importance in characterizing the sample complexity of PAC learning. In particular, it is well known that the sample complexity is finite for any $\varepsilon, \delta \in (0,1)$ if and only if $d < \infty$ (Vapnik, 1982; Blumer, Ehrenfeucht, Haussler, and Warmuth, 1989; Ehrenfeucht, Haussler, Kearns, and Valiant, 1989). For simplicity of notation, for the remainder of this article we suppose $d < \infty$; furthermore, note that our assumption of $|\mathbb{C}| \geq 3$ implies $d \geq 1$.

We adopt a common variation on big-O asymptotic notation, used in much of the learning theory literature. Specifically, for functions $f, g : (0,1)^2 \to [0,\infty)$, we let $f(\varepsilon,\delta) = O(g(\varepsilon,\delta))$ denote the assertion that $\exists \varepsilon_0, \delta_0 \in (0,1)$ and $c_0 \in (0,\infty)$ such that, $\forall \varepsilon \in (0,\varepsilon_0)$, $\forall \delta \in (0,\delta_0)$, $f(\varepsilon,\delta) \leq c_0 g(\varepsilon,\delta)$; however, we also require that the values $\varepsilon_0, \delta_0, c_0$ in this definition be *numerical constants*, meaning that they are *independent of* $\mathbb{C}$ *and* $\mathcal{X}$. For instance, this means c_0 cannot depend on d. We equivalently write $f(\varepsilon,\delta) = \Omega(g(\varepsilon,\delta))$ to assert that $g(\varepsilon,\delta) = O(f(\varepsilon,\delta))$. Finally, we write $f(\varepsilon,\delta) = \Theta(g(\varepsilon,\delta))$ to assert that both $f(\varepsilon,\delta) = O(g(\varepsilon,\delta))$ and $f(\varepsilon,\delta) = \Omega(g(\varepsilon,\delta))$ hold. We also sometimes write $O(g(\varepsilon,\delta))$ in an expression, as a place-holder for some function $f(\varepsilon,\delta)$ satisfying $f(\varepsilon,\delta) = O(g(\varepsilon,\delta))$: for instance, the statement $N(\varepsilon,\delta) \leq d + O(\log(1/\delta))$ expresses that $\exists f(\varepsilon,\delta) = O(\log(1/\delta))$ for which $N(\varepsilon,\delta) \leq d + f(\varepsilon,\delta)$. Also, for any value $z \geq 0$, define $\mathrm{Log}(z) = \ln(\max\{z, e\})$ and similarly $\mathrm{Log}_2(z) = \log_2(\max\{z, 2\})$.

As is commonly required in the learning theory literature, we adopt the assumption that the events appearing in probability claims below are indeed measurable. For our purposes, this comes into effect only in the application of classic generalization bounds for sample-consistent classifiers (Lemma 4 below). See Blumer, Ehrenfeucht, Haussler, and Warmuth (1989) and van der Vaart and Wellner (1996) for discussion of conditions on $\mathbb{C}$ sufficient for this measurability assumption to hold.

3. Background

Our objective in this work is to establish *sharp* sample complexity bounds. As such, we should first review the known *lower bounds* on $\mathcal{M}(\varepsilon,\delta)$. A basic lower bound of $\frac{1-\varepsilon}{\varepsilon}\ln\left(\frac{1}{\delta}\right)$ was established by Blumer, Ehrenfeucht, Haussler, and Warmuth (1989) for $0 < \varepsilon < 1/2$ and $0 < \delta < 1$. A second lower bound of $\frac{d-1}{32\varepsilon}$ was supplied by Ehrenfeucht, Haussler, Kearns, and Valiant (1989), for $0 < \varepsilon \leq 1/8$ and $0 < \delta \leq 1/100$. Taken together, these results imply that, for any $\varepsilon \in (0, 1/8]$ and $\delta \in (0, 1/100]$,

$$\mathcal{M}(\varepsilon,\delta) \geq \max\left\{\frac{d-1}{32\varepsilon}, \frac{1-\varepsilon}{\varepsilon}\ln\left(\frac{1}{\delta}\right)\right\} = \Omega\left(\frac{1}{\varepsilon}\left(d + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)\right). \tag{1}$$

This lower bound is complemented by classic *upper bounds* on the sample complexity. In particular, Vapnik (1982) and Blumer, Ehrenfeucht, Haussler, and Warmuth (1989) established an upper bound of

$$\mathcal{M}(\varepsilon,\delta) = O\left(\frac{1}{\varepsilon}\left(d\mathrm{Log}\left(\frac{1}{\varepsilon}\right) + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)\right). \tag{2}$$

They proved that this sample complexity bound is in fact achieved by any algorithm that returns a classifier $h \in \mathbb{C}[(\mathbb{X}_{1:m}(\mathcal{P}), f^\star(\mathbb{X}_{1:m}(\mathcal{P})))]$, also known as a *sample-consistent learning algorithm* (or *empirical risk minimization* algorithm). A sometimes-better upper bound was established by Haussler, Littlestone, and Warmuth (1994):

$$\mathcal{M}(\varepsilon,\delta) = O\left(\frac{d}{\varepsilon}\mathrm{Log}\left(\frac{1}{\delta}\right)\right). \tag{3}$$

This bound is achieved by a modified variant of the *one-inclusion graph prediction algorithm*, a learning algorithm also proposed by Haussler, Littlestone, and Warmuth (1994), which has been conjectured to achieve the optimal sample complexity (Warmuth, 2004).

In very recent work, Simon (2015) produced a breakthrough insight. Specifically, by analyzing a learning algorithm based on a simple majority vote among classifiers consistent with distinct subsets of the training data, Simon (2015) established that, for any $K \in \mathbb{N}$,

$$\mathcal{M}(\varepsilon,\delta) = O\left(\frac{2^{2K}\sqrt{K}}{\varepsilon}\left(d\log^{(K)}\left(\frac{1}{\varepsilon}\right) + K + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)\right), \tag{4}$$

where $\log^{(K)}(x)$ is the K-times iterated logarithm: $\log^{(0)}(x) = \max\{x, 1\}$ and $\log^{(K)}(x) = \max\{\log_2(\log^{(K-1)}(x)), 1\}$. In particular, one natural choice would be $K \approx \log^*\left(\frac{1}{\varepsilon}\right)$,[3] which (one can show) optimizes the asymptotic dependence on ε in the bound, yielding

$$\mathcal{M}(\varepsilon,\delta) = O\left(\frac{1}{\varepsilon}2^{O(\log^*(1/\varepsilon))}\left(d + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)\right).$$

In general, the entire form of the bound (4) is optimized (up to numerical constant factors) by choosing $K = \max\left\{\log^*\left(\frac{1}{\varepsilon}\right) - \log^*\left(\frac{1}{d}\mathrm{Log}\left(\frac{1}{\delta}\right)\right) + 1, 1\right\}$. Note that, with either of these choices of K, there is a range of ε, δ, and d values for which the bound (4) is strictly smaller than both (2) and (3): for instance, for small ε, it suffices to have $\mathrm{Log}(1/\delta) \ll d\mathrm{Log}(1/\varepsilon)/(2^{2\log^*(1/\varepsilon)}\sqrt{\log^*(1/\varepsilon)})$ while $2^{2\log^*(1/\varepsilon)}\sqrt{\log^*(1/\varepsilon)} \ll \min\{\mathrm{Log}(1/\delta), d\}$. However, this bound still does not quite match the form of the lower bound (1).

There have also been many special-case analyses, studying restricted types of concept spaces $\mathbb{C}$ for which the above gaps can be closed (e.g., Auer and Ortner, 2007; Darnstädt, 2015; Hanneke, 2015). However, these special conditions do not include many of the most-commonly studied concept spaces, such as linear separators and multilayer neural networks. There have also been a variety of studies that, in addition to restricting to specific concept spaces $\mathbb{C}$, also introduce strong restrictions on the data distribution $\mathcal{P}$, and establish an upper bound of the same form as the lower bound (1) under these restrictions (e.g., Long, 2003; Giné and Koltchinskii, 2006; Bshouty, Li, and Long, 2009; Hanneke, 2009, 2015; Balcan and Long, 2013). However, there are many interesting classes $\mathbb{C}$ and distributions $\mathcal{P}$ for which these results do not imply any improvements over (2). Thus, in the present literature, there persists a gap between the lower bound (1) and the minimum of all of the known upper bounds (2), (3), and (4) applicable to the *general* case of an arbitrary concept space of a given VC dimension d (under arbitrary data distributions).

In the present work, we establish a new upper bound for a novel learning algorithm, which holds for *any* concept space $\mathbb{C}$, and which improves over all of the above general

3. The function $\log^*(x)$ is the iterated logarithm: the smallest $K \in \mathbb{N} \cup \{0\}$ for which $\log^{(K)}(x) \leq 1$. It is an extremely slowly growing function of x.

upper bounds in its joint dependence on ε, δ, and d. In particular, it is *optimal*, in the sense that it matches the lower bound (1) up to numerical constant factors. This work thus solves a long-standing open problem, by determining the precise form of the optimal sample complexity, up to numerical constant factors.

4. Main Result

This section presents the main contributions of this work: a novel learning algorithm, and a proof that it achieves the optimal sample complexity.

4.1 Sketch of the Approach

The general approach used here builds on an argument of Simon (2015), which itself has roots in the analysis of sample-consistent learning algorithms by Hanneke (2009, Section 2.9.1). The essential idea from Simon (2015) is that, if we have two classifiers, $\hat{h}$ and $\hat{g}$, the latter of which is an element of $\mathbb{C}$ consistent with an i.i.d. data set $\tilde{S}$ *independent* from $\hat{h}$, then we can analyze the probability that they *both* make a mistake on a random point by bounding the error rate of $\hat{h}$ under the distribution $\mathcal{P}$, and bounding the error rate of $\hat{g}$ under the *conditional* distribution given that $\hat{h}$ makes a mistake. In particular, it will either be the case that $\hat{h}$ itself has small error rate, or else (if $\hat{h}$ has error rate larger than our desired bound) with high probability, the number of points in $\tilde{S}$ contained in the error region of $\hat{h}$ will be at least some number $\propto \mathrm{er}_{\mathcal{P}}(\hat{h}; f^\star)|\tilde{S}|$; in the latter case, we can bound the conditional error rate of $\hat{g}$ in terms of the number of such points via a classic generalization bound for sample-consistent classifiers (Lemma 4 below). Multiplying this bound on the conditional error rate of $\hat{g}$ by the error rate of $\hat{h}$ results in a bound on the probability they both make a mistake. More specifically, this argument yields a bound of the following form: for an appropriate numerical constant $\tilde{c} \in (0, \infty)$, with probability at least $1-\delta$, $\forall \hat{g} \in \mathbb{C}[\tilde{S}]$,

$$\mathcal{P}\left(x : \hat{h}(x) = \hat{g}(x) \neq f^\star(x)\right) \leq \frac{\tilde{c}}{|\tilde{S}|}\left(d\mathrm{Log}\left(\frac{\mathrm{er}_{\mathcal{P}}(\hat{h}; f^\star)|\tilde{S}|}{d}\right) + \mathrm{Log}\left(\frac{1}{\delta}\right)\right).$$

The original analysis of Simon (2015) applied this reasoning repeatedly, in an inductive argument, thereby bounding the probability that K classifiers, each consistent with one of K independent training sets, all make a mistake on a random point. He then reasoned that the error rate of the majority vote of $2K-1$ such classifiers can be bounded by the sum of these bounds for all subsets of K of these classifiers, since the majority vote classifier agrees with at least K of the constituent classifiers.

In the present work, we also consider a simple majority vote of a number of classifiers, but we alter the way the data is split up, allowing significant overlaps among the subsamples. In particular, each classifier is trained on considerably more data this way. We construct these subsamples recursively, motivated by an inductive analysis of the sample complexity. At each stage, we have a working set S of i.i.d. data points, and another sequence T of data points, referred to as the *partially-constructed subsample*. As a terminal case, if $|S|$ is smaller than a certain cutoff size, we generate a subsample $S \cup T$, on which we will train a classifier $\hat{g} \in \mathbb{C}[S \cup T]$. Otherwise (for the nonterminal case), we use (roughly) a constant fraction of the points in S to form a subsequence S_0, and make three recursive calls to the

algorithm, using S_0 as the working set in each call. By an inductive hypothesis, for each of these three recursive calls, with probability $1-\delta'$, the majority vote of the classifiers trained on subsamples generated by that call has error rate at most $\frac{c}{|S_0|}(d + \mathrm{Log}(1/\delta'))$, for an appropriate numerical constant c. These three majority vote classifiers, denoted h_1, h_2, h_3, will each play the role of $\hat{h}$ in the argument above.

With the remaining constant fraction of data points in S (i.e., those not used to form S_0), we divide them into three independent subsequences S_1, S_2, S_3. Then for each of the three recursive calls, we provide as its partially-constructed subsample (i.e., the "T" argument) a sequence $S_i \cup S_j \cup T$ with $i \neq j$; specifically, for the k^{th} recursive call ($k \in \{1,2,3\}$), we take $\{i,j\} = \{1,2,3\} \setminus \{k\}$. Since the argument T is retained within the partially-constructed subsample passed to each recursive call, a simple inductive argument reveals that, for each $i \in \{1,2,3\}$, $\forall k \in \{1,2,3\} \setminus \{i\}$, all of the classifiers $\hat{g}$ trained on subsamples generated in the k^{th} recursive call are contained in $\mathbb{C}[S_i]$. Furthermore, since S_i is not included in the argument to the i^{th} recursive call, h_i and S_i are independent. Thus, by the argument discussed above, applied with $\hat{h} = h_i$ and $\tilde{S} = S_i$, we have that with probability at least $1-\delta'$, for any $\hat{g}$ trained on a subsample generated in recursive calls $k \in \{1,2,3\} \setminus \{i\}$, the probability that *both* h_i *and* $\hat{g}$ make a mistake on a random point is at most $\frac{\tilde{c}}{|S_i|}\left(d\mathrm{Log}\left(\frac{\mathrm{er}_{\mathcal{P}}(h_i; f^\star)|S_i|}{d}\right) + \mathrm{Log}\left(\frac{1}{\delta'}\right)\right)$. Composing this with the aforementioned inductive hypothesis, recalling that $|S_i| \propto |S|$ and $|S_0| \propto |S|$, and simplifying by a bit of calculus, this is at most $\frac{c'}{|S|}\left(d\mathrm{Log}(c) + \mathrm{Log}\left(\frac{1}{\delta'}\right)\right)$, for an appropriate numerical constant c'. By choosing $\delta' \propto \delta$ appropriately, the union bound implies that, with probability at least $1-\delta$, this holds for all choices of $i \in \{1,2,3\}$. Furthermore, by choosing c sufficiently large, this bound is at most $\frac{c}{12|S|}\left(d + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)$.

To complete the inductive argument, we then note that on any point x, the majority vote of all of the classifiers (from all three recursive calls) must agree with at least one of the three classifiers h_i, and must agree with at least $1/4$ of the classifiers $\hat{g}$ trained on subsamples generated in recursive calls $k \in \{1,2,3\} \setminus \{i\}$. Therefore, on any point x for which the majority vote makes a mistake, with probability at least $1/12$, a uniform random choice of $i \in \{1,2,3\}$, and of $\hat{g}$ from recursive calls $k \in \{1,2,3\} \setminus \{i\}$, results in h_i and $\hat{g}$ that both make a mistake on x. Applying this fact to a *random* point $X \sim \mathcal{P}$ (and invoking Fubini's theorem), this implies that the error rate of the majority vote is at most 12 times the average (over choices of i and $\hat{g}$) of the probabilities that h_i and $\hat{g}$ both make a mistake on X. Combined with the above bound, this is at most $\frac{c}{|S|}\left(d + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)$. The formal details are provided below.

4.2 Formal Details

For any $k \in \mathbb{N} \cup \{0\}$, and any $S \in (\mathcal{X} \times \mathcal{Y})^k$ with $\mathbb{C}[S] \neq \emptyset$, let $L(S)$ denote an arbitrary classifier h in $\mathbb{C}[S]$, entirely determined by S: that is, $L(\cdot)$ is a fixed sample-consistent learning algorithm (i.e., empirical risk minimizer). For any $k \in \mathbb{N}$ and sequence of data sets $\{S_1, \ldots, S_k\}$, denote $L(\{S_1, \ldots, S_k\}) = \{L(S_1), \ldots, L(S_k)\}$. Also, for any values $y_1, \ldots, y_k \in \mathcal{Y}$, define the majority function: $\mathrm{Majority}(y_1, \ldots, y_k) = \mathrm{sign}\left(\sum_{i=1}^k y_i\right) = 2\mathbb{1}\left[\sum_{i=1}^k y_i \geq 0\right] - 1$. We also overload this notation, defining the *majority classifier* $\mathrm{Majority}(h_1, \ldots, h_k)(x) = \mathrm{Majority}(h_1(x), \ldots, h_k(x))$, for any classifiers $h_1, \ldots, h_k$.

Now consider the following recursive algorithm, which takes as input two finite data sets, S and T, satisfying $\mathbb{C}[S \cup T] \neq \emptyset$, and returns a *finite sequence of data sets* (referred to as *subsamples* of $S \cup T$). The classifier used to achieve the new sample complexity bound below is simply the majority vote of the classifiers obtained by applying L to these subsamples.

Algorithm: $\mathbb{A}(S;T)$
0. If $|S| \leq 3$
1. Return $\{S \cup T\}$
2. Let S_0 denote the first $|S| - 3\lfloor |S|/4 \rfloor$ elements of S, S_1 the next $\lfloor |S|/4 \rfloor$ elements, S_2 the next $\lfloor |S|/4 \rfloor$ elements, and S_3 the remaining $\lfloor |S|/4 \rfloor$ elements after that
3. Return $\mathbb{A}(S_0; S_2 \cup S_3 \cup T) \cup \mathbb{A}(S_0; S_1 \cup S_3 \cup T) \cup \mathbb{A}(S_0; S_1 \cup S_2 \cup T)$

Theorem 2

$$\mathcal{M}(\varepsilon, \delta) = O\left(\frac{1}{\varepsilon}\left(d + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)\right).$$

In particular, a sample complexity of the form expressed on the right hand side is achieved by the algorithm that returns the classifier $\mathrm{Majority}(L(\mathbb{A}(S;\emptyset)))$*, given any data set* S*.*

Combined with (1), this immediately implies the following corollary.

Corollary 3

$$\mathcal{M}(\varepsilon, \delta) = \Theta\left(\frac{1}{\varepsilon}\left(d + \mathrm{Log}\left(\frac{1}{\delta}\right)\right)\right).$$

The algorithm $\mathbb{A}$ is expressed above as a recursive method for constructing a sequence of subsamples, as this form is most suitable for the arguments in the proof below. However, it should be noted that one can equivalently describe these constructed subsamples *directly*, as the selection of which data points should be included in which subsamples can be expressed as a simple function of the indices. To illustrate this, consider the simplest case in which $S = \{(x_0, y_0), \ldots, (x_{m-1}, y_{m-1})\}$ with $m = 4^\ell$ for some $\ell \in \mathbb{N}$: that is, $|S|$ is a power of 4. In this case, let $\{T_0, \ldots, T_{n-1}\}$ denote the sequence of labeled data sets returned by $\mathbb{A}(S;\emptyset)$, and note that since each recursive call reduces $|S|$ by a factor of 4 while making 3 recursive calls, we have $n = 3^\ell$. First, note that (x_0, y_0) is contained in *every* subsample T_i. For the rest, consider any $i \in \{1, \ldots, m-1\}$ and $j \in \{0, \ldots, n-1\}$, and let us express i in its base-4 representation as $i = \sum_{t=0}^{\ell-1} i_t 4^t$, where each $i_t \in \{0,1,2,3\}$, and express j in its base-3 representation as $j = \sum_{t=0}^{\ell-1} j_t 3^t$, where each $j_t \in \{0,1,2\}$. Then it holds that $(x_i, y_i) \in T_j$ if and only if the largest $t \in \{0, \ldots, \ell-1\}$ with $i_t \neq 0$ satisfies $i_t - 1 \neq j_t$. This kind of direct description of the subsamples is also possible when $|S|$ is not a power of 4, though a bit more complicated to express.

4.3 Proof of Theorem 2

The following classic result will be needed in the proof. A bound of this type is implied by a theorem of Vapnik (1982); the version stated here features slightly smaller constant factors, obtained by Blumer, Ehrenfeucht, Haussler, and Warmuth (1989).[4]

4. Specifically, it follows by combining their Theorem A2.1 and Proposition A2.1, setting the resulting expression equal to δ and solving for ε.

Lemma 4 *For any* $\delta \in (0,1)$*,* $m \in \mathbb{N}$*,* $f^\star \in \mathbb{C}$*, and any probability measure* P *over* $\mathcal{X}$*, letting* $Z_1, \ldots, Z_m$ *be independent* P*-distributed random variables, with probability at least* $1 - \delta$*, every* $h \in \mathbb{C}[\{(Z_i, f^\star(Z_i))\}_{i=1}^m]$ *satisfies*

$$\mathrm{er}_P(h; f^\star) \leq \frac{2}{m}\left(d\mathrm{Log}_2\left(\frac{2em}{d}\right) + \mathrm{Log}_2\left(\frac{2}{\delta}\right)\right).$$

We are now ready for the proof of Theorem 2.

Proof of Theorem 2 Fix any $f^\star \in \mathbb{C}$ and probability measure $\mathcal{P}$ over $\mathcal{X}$, and for brevity, denote $\mathbb{S}_{1:m} = (\mathbb{X}_{1:m}(\mathcal{P}), f^\star(\mathbb{X}_{1:m}(\mathcal{P})))$, for each $m \in \mathbb{N}$. Also, for any classifier h, define $\mathrm{ER}(h) = \{x \in \mathcal{X} : h(x) \neq f^\star(x)\}$.

We begin by noting that, for any finite sequences S and T of points in $\mathcal{X} \times \mathcal{Y}$, a straightforward inductive argument reveals that all of the subsamples $\hat{S}$ in the sequence returned by $\mathbb{A}(S;T)$ satisfy $\hat{S} \subseteq S \cup T$ (since no additional data points are ever introduced in any step). Thus, if $f^\star \in \mathbb{C}[S]$ and $f^\star \in \mathbb{C}[T]$, then $f^\star \in \mathbb{C}[S] \cap \mathbb{C}[T] = \mathbb{C}[S \cup T] \subseteq \mathbb{C}[\hat{S}]$, so that $\mathbb{C}[\hat{S}] \neq \emptyset$. In particular, this means that, in this case, each of these subsamples $\hat{S}$ is a valid input to $L(\cdot)$, and thus $L(\mathbb{A}(S;T))$ is a well-defined sequence of classifiers. Furthermore, since the recursive calls all have T as a subsequence of their second arguments, and the terminal case (i.e., Step 1) includes this second argument in the constructed subsample, another straightforward inductive argument implies that every subsample $\hat{S}$ returned by $\mathbb{A}(S;T)$ satisfies $\hat{S} \supseteq T$. Thus, in the case that $f^\star \in \mathbb{C}[S]$ and $f^\star \in \mathbb{C}[T]$, by definition of L, we also have that every classifier h in the sequence $L(\mathbb{A}(S;T))$ satisfies $h \in \mathbb{C}[T]$.

Fix a numerical constant $c = 1800$. We will prove by induction that, for any $m' \in \mathbb{N}$, for every $\delta' \in (0,1)$, and every finite sequence T' of points in $\mathcal{X} \times \mathcal{Y}$ with $f^\star \in \mathbb{C}[T']$, with probability at least $1 - \delta'$, the classifier $\hat{h}_{m',T'} = \mathrm{Majority}\left(L\left(\mathbb{A}(\mathbb{S}_{1:m'}; T')\right)\right)$ satisfies

$$\mathrm{er}_{\mathcal{P}}(\hat{h}_{m',T'}; f^\star) \leq \frac{c}{m'+1}\left(d + \ln\left(\frac{18}{\delta'}\right)\right). \tag{5}$$

First, as a base case, consider any $m' \in \mathbb{N}$ with $m' \leq c\ln(18e) - 1$. In this case, fix any $\delta' \in (0,1)$ and any sequence T' with $f^\star \in \mathbb{C}[T']$. Also note that $f^\star \in \mathbb{C}[\mathbb{S}_{1:m'}]$. Thus, as discussed above, $\hat{h}_{m',T'}$ is a well-defined classifier. We then trivially have

$$\mathrm{er}_{\mathcal{P}}(\hat{h}_{m',T'}; f^\star) \leq 1 \leq \frac{c}{m'+1}(1 + \ln(18)) < \frac{c}{m'+1}\left(d + \ln\left(\frac{18}{\delta'}\right)\right),$$

so that (5) holds.

Now take as an inductive hypothesis that, for some $m \in \mathbb{N}$ with $m > c\ln(18e) - 1$, for every $m' \in \mathbb{N}$ with $m' < m$, we have that for every $\delta' \in (0,1)$ and every finite sequence T' in $\mathcal{X} \times \mathcal{Y}$ with $f^\star \in \mathbb{C}[T']$, with probability at least $1 - \delta'$, (5) is satisfied. To complete the inductive proof, we aim to establish that this remains the case with $m' = m$ as well. Fix any $\delta \in (0,1)$ and any finite sequence T of points in $\mathcal{X} \times \mathcal{Y}$ with $f^\star \in \mathbb{C}[T]$. Note that $c\ln(18e) - 1 \geq 3$, so that (since $|\mathbb{S}_{1:m}| = m > c\ln(18e) - 1$) we have $|\mathbb{S}_{1:m}| \geq 4$, and hence the execution of $\mathbb{A}(\mathbb{S}_{1:m}; T)$ returns in Step 3 (not Step 1). Let S_0, S_1, S_2, S_3 be as in the definition of $\mathbb{A}(S;T)$, with $S = \mathbb{S}_{1:m}$. Also denote $T_1 = S_2 \cup S_3 \cup T$, $T_2 = S_1 \cup S_3 \cup T$, $T_3 = S_1 \cup S_2 \cup T$, and for each $i \in \{1,2,3\}$, denote $h_i = \mathrm{Majority}\left(L\left(\mathbb{A}(S_0; T_i)\right)\right)$, corresponding to

the majority votes of classifiers trained on the subsamples from each of the three recursive calls in the algorithm.

Note that $S_0 = \mathbb{S}_{1:(m-3\lfloor m/4 \rfloor)}$. Furthermore, since $m \geq 4$, we have $1 \leq m - 3\lfloor m/4 \rfloor < m$. Also note that $f^\star \in \mathbb{C}[S_i]$ for each $i \in \{1,2,3\}$, which, together with the fact that $f^\star \in \mathbb{C}[T]$, implies $f^\star \in \mathbb{C}[T] \cap \bigcap_{j \in \{1,2,3\} \setminus \{i\}} \mathbb{C}[S_j] = \mathbb{C}[T_i]$ for each $i \in \{1,2,3\}$. Thus, since $f^\star \in \mathbb{C}[S_0]$ as well, for each $i \in \{1,2,3\}$, $L\left(\mathbb{A}(S_0;T_i)\right)$ is a well-defined sequence of classifiers (as discussed above), so that h_i is also well-defined. In particular, note that $h_i = \hat{h}_{(m-3\lfloor m/4 \rfloor), T_i}$. Therefore, by the inductive hypothesis (applied under the conditional distribution given S_1, S_2, S_3, which are independent of S_0), combined with the law of total probability, for each $i \in \{1,2,3\}$, there is an event E_i of probability at least $1 - \delta/9$, on which

$$\mathcal{P}(\mathrm{ER}(h_i)) \leq \frac{c}{|S_0|+1}\left(d + \ln\left(\frac{9 \cdot 18}{\delta}\right)\right) \leq \frac{4c}{m}\left(d + \ln\left(\frac{9 \cdot 18}{\delta}\right)\right). \tag{6}$$

Next, fix any $i \in \{1,2,3\}$, and denote by $\{(Z_{i,1}, f^\star(Z_{i,1})), \ldots, (Z_{i,N_i}, f^\star(Z_{i,N_i}))\} = S_i \cap (\mathrm{ER}(h_i) \times \mathcal{Y})$, where $N_i = |S_i \cap (\mathrm{ER}(h_i) \times \mathcal{Y})|$: that is, $\{(Z_{i,t}, f^\star(Z_{i,t}))\}_{t=1}^{N_i}$ is the subsequence of elements (x,y) in S_i for which $x \in \mathrm{ER}(h_i)$. Note that, since h_i and S_i are independent, $Z_{i,1}, \ldots, Z_{i,N_i}$ are conditionally independent given h_i and N_i, each with conditional distribution $\mathcal{P}(\cdot|\mathrm{ER}(h_i))$ (if $N_i > 0$). Thus, applying Lemma 4 under the conditional distribution given h_i and N_i, combined with the law of total probability, we have that on an event E_i' of probability at least $1 - \delta/9$, if $N_i > 0$, then every $h \in \mathbb{C}\left[\{(Z_{i,t}, f^\star(Z_{i,t}))\}_{t=1}^{N_i}\right]$ satisfies

$$\mathrm{er}_{\mathcal{P}(\cdot|\mathrm{ER}(h_i))}(h; f^\star) \leq \frac{2}{N_i}\left(d\mathrm{Log}_2\left(\frac{2eN_i}{d}\right) + \mathrm{Log}_2\left(\frac{18}{\delta}\right)\right).$$

Furthermore, as discussed above, each $j \in \{1,2,3\} \setminus \{i\}$ and $h \in L\left(\mathbb{A}(S_0;T_j)\right)$ have $h \in \mathbb{C}[T_j]$, and $T_j \supseteq S_i \supseteq \{(Z_{i,t}, f^\star(Z_{i,t}))\}_{t=1}^{N_i}$, so that $\mathbb{C}[T_j] \subseteq \mathbb{C}\left[\{(Z_{i,t}, f^\star(Z_{i,t}))\}_{t=1}^{N_i}\right]$. It follows that every $h \in \bigcup_{j \in \{1,2,3\} \setminus \{i\}} L\left(\mathbb{A}(S_0;T_j)\right)$ has $h \in \mathbb{C}\left[\{(Z_{i,t}, f^\star(Z_{i,t}))\}_{t=1}^{N_i}\right]$. Thus, on the event E_i', if $N_i > 0$, $\forall h \in \bigcup_{j \in \{1,2,3\} \setminus \{i\}} L(\mathbb{A}(S_0;T_j))$,

$$\begin{aligned}
&\mathcal{P}(\mathrm{ER}(h_i) \cap \mathrm{ER}(h)) = \mathcal{P}(\mathrm{ER}(h_i))\mathcal{P}(\mathrm{ER}(h)|\mathrm{ER}(h_i)) \\
&= \mathcal{P}(\mathrm{ER}(h_i))\mathrm{er}_{\mathcal{P}(\cdot|\mathrm{ER}(h_i))}(h; f^\star) \leq \mathcal{P}(\mathrm{ER}(h_i))\frac{2}{N_i}\left(d\mathrm{Log}_2\left(\frac{2eN_i}{d}\right) + \mathrm{Log}_2\left(\frac{18}{\delta}\right)\right).
\end{aligned} \tag{7}$$

Additionally, since h_i and S_i are independent, by a Chernoff bound (applied under the conditional distribution given h_i) and the law of total probability, there is an event E_i'' of probability at least $1 - \delta/9$, on which, if $\mathcal{P}(\mathrm{ER}(h_i)) \geq \frac{23}{\lfloor m/4 \rfloor}\ln\left(\frac{9}{\delta}\right) \geq \frac{2(10/3)^2}{\lfloor m/4 \rfloor}\ln\left(\frac{9}{\delta}\right)$, then

$$N_i \geq (7/10)\mathcal{P}(\mathrm{ER}(h_i))|S_i| = (7/10)\mathcal{P}(\mathrm{ER}(h_i))\lfloor m/4 \rfloor.$$

In particular, on E_i'', if $\mathcal{P}(\mathrm{ER}(h_i)) \geq \frac{23}{\lfloor m/4 \rfloor}\ln\left(\frac{9}{\delta}\right)$, then the above inequality implies $N_i > 0$.

Combining this with (6) and (7), and noting that $\mathrm{Log}_2(x) \leq \mathrm{Log}(x)/\ln(2)$ and $x \mapsto \frac{1}{x}\mathrm{Log}(c'x)$ is nonincreasing on $(0,\infty)$ (for any fixed $c' > 0$), we have that on $E_i \cap E_i' \cap E_i''$,

if $\mathcal{P}(\mathrm{ER}(h_i)) \geq \frac{23}{\lfloor m/4 \rfloor}\ln\left(\frac{9}{\delta}\right)$, then every $h \in \bigcup_{j \in \{1,2,3\} \setminus \{i\}} L(\mathbb{A}(S_0;T_j))$ has

$$\begin{aligned}
\mathcal{P}(\mathrm{ER}(h_i) \cap \mathrm{ER}(h)) &\leq \frac{20}{7\ln(2)\lfloor m/4 \rfloor}\left(d\mathrm{Log}\left(\frac{2e(7/10)\mathcal{P}(\mathrm{ER}(h_i))\lfloor m/4 \rfloor}{d}\right) + \mathrm{Log}\left(\frac{18}{\delta}\right)\right) \\
&\leq \frac{20}{7\ln(2)\lfloor m/4 \rfloor}\left(d\mathrm{Log}\left(\frac{(7/5)ec\left(d + \ln\left(\frac{9 \cdot 18}{\delta}\right)\right)}{d}\right) + \mathrm{Log}\left(\frac{18}{\delta}\right)\right) \\
&\leq \frac{20}{7\ln(2)\lfloor m/4 \rfloor}\left(d\mathrm{Log}\left((2/5)c\left((7/2)e + \frac{7e}{d}\ln\left(\frac{18}{\delta}\right)\right)\right) + \mathrm{Log}\left(\frac{18}{\delta}\right)\right) \\
&\leq \frac{20}{7\ln(2)\lfloor m/4 \rfloor}\left(d\ln((9/5)ec) + 8\ln\left(\frac{18}{\delta}\right)\right),
\end{aligned} \tag{8}$$

where this last inequality is due to Lemma 5 in Appendix A. Since $m > c\ln(18e) - 1 > 3200$, we have $\lfloor m/4 \rfloor > (m-4)/4 > \frac{799}{800}\frac{m}{4} > \frac{799}{800}\frac{3200}{3201}\frac{m+1}{4}$. Plugging this relaxation into the above bound, combined with numerical calculation of the logarithmic factor (with c as defined above), we find that the expression in (8) is less than

$$\frac{150}{m+1}\left(d + \ln\left(\frac{18}{\delta}\right)\right).$$

Additionally, if $\mathcal{P}(\mathrm{ER}(h_i)) < \frac{23}{\lfloor m/4 \rfloor}\ln\left(\frac{9}{\delta}\right)$, then monotonicity of measures implies

$$\mathcal{P}(\mathrm{ER}(h_i) \cap \mathrm{ER}(h)) \leq \mathcal{P}(\mathrm{ER}(h_i)) < \frac{23}{\lfloor m/4 \rfloor}\ln\left(\frac{9}{\delta}\right) < \frac{150}{m+1}\left(d + \ln\left(\frac{18}{\delta}\right)\right),$$

again using the above lower bound on $\lfloor m/4 \rfloor$ for this last inequality. Thus, regardless of the value of $\mathcal{P}(\mathrm{ER}(h_i))$, on the event $E_i \cap E_i' \cap E_i''$, we have $\forall h \in \bigcup_{j \in \{1,2,3\} \setminus \{i\}} L(\mathbb{A}(S_0;T_j))$,

$$\mathcal{P}(\mathrm{ER}(h_i) \cap \mathrm{ER}(h)) < \frac{150}{m+1}\left(d + \ln\left(\frac{18}{\delta}\right)\right).$$

Now denote $h_{\mathrm{maj}} = \hat{h}_{m,T} = \mathrm{Majority}\left(L(\mathbb{A}(S;T))\right)$, again with $S = \mathbb{S}_{1:m}$. By definition of Majority$(\cdot)$, for any $x \in \mathcal{X}$, at least $1/2$ of the classifiers h in the sequence $L(\mathbb{A}(S;T))$ have $h(x) = h_{\mathrm{maj}}(x)$. From this fact, the strong form of the pigeonhole principle implies that at least one $i \in \{1,2,3\}$ has $h_i(x) = h_{\mathrm{maj}}(x)$ (i.e., the majority vote must agree with the majority of classifiers in at least one of the three subsequences of classifiers). Furthermore, since each $\mathbb{A}(S_0;T_j)$ (with $j \in \{1,2,3\}$) supplies an equal number of entries to the sequence $\mathbb{A}(S;T)$ (by a straightforward inductive argument), for each $i \in \{1,2,3\}$, at least $1/4$ of the classifiers h in $\cup_{j \in \{1,2,3\} \setminus \{i\}} L(\mathbb{A}(S_0;T_j))$ have $h(x) = h_{\mathrm{maj}}(x)$: that is, since $|L(\mathbb{A}(S_0;T_i))| = (1/3)|L(\mathbb{A}(S;T))|$, we must have at least $(1/6)|L(\mathbb{A}(S;T))| = (1/4)\left|\bigcup_{j \in \{1,2,3\} \setminus \{i\}} L(\mathbb{A}(S_0;T_j))\right|$ classifiers h in $\bigcup_{j \in \{1,2,3\} \setminus \{i\}} L(S_0;T_j)$ with $h(x) = h_{\mathrm{maj}}(x)$ in order to meet the total of at least $(1/2)|L(\mathbb{A}(S;T))|$ classifiers $h \in L(\mathbb{A}(S;T))$ with $h(x) = h_{\mathrm{maj}}(x)$. In particular, letting I be a random variable uniformly distributed on $\{1,2,3\}$ (independent of the data), and letting $\tilde{h}$ be a random variable conditionally (given I and S) uniformly distributed on the classifiers $\cup_{j \in \{1,2,3\} \setminus \{I\}} L(\mathbb{A}(S_0;T_j))$, this implies that for any fixed $x \in \mathrm{ER}(h_{\mathrm{maj}})$, with conditional (given S) probability at least $1/12$,

$h_I(x) = \tilde{h}(x) = h_{\text{maj}}(x)$, so that $x \in \text{ER}(h_I) \cap \text{ER}(\tilde{h})$ as well. Thus, for a random variable $X \sim \mathcal{P}$ (independent of the data and I,$\tilde{h}$), the law of total probability and monotonicity of conditional expectations imply

$$\begin{aligned}
&\mathbb{E}\left[\mathcal{P}\left(\text{ER}(h_I) \cap \text{ER}(\tilde{h})\right)\middle|S\right] = \mathbb{E}\left[\mathbb{P}\left(X \in \text{ER}(h_I) \cap \text{ER}(\tilde{h})\middle|I, \tilde{h}, S\right)\middle|S\right] \\
&= \mathbb{E}\left[\mathbb{1}\left[X \in \text{ER}(h_I) \cap \text{ER}(\tilde{h})\right]\middle|S\right] = \mathbb{E}\left[\mathbb{P}\left(X \in \text{ER}(h_I) \cap \text{ER}(\tilde{h})\middle|S, X\right)\middle|S\right] \\
&\geq \mathbb{E}\left[\mathbb{P}\left(X \in \text{ER}(h_I) \cap \text{ER}(\tilde{h})\middle|S, X\right)\mathbb{1}\left[X \in \text{ER}(h_{\text{maj}})\right]\middle|S\right] \\
&\geq \mathbb{E}\left[(1/12)\mathbb{1}\left[X \in \text{ER}(h_{\text{maj}})\right]|S\right] = (1/12)\text{er}_{\mathcal{P}}(h_{\text{maj}}; f^\star).
\end{aligned}$$

Thus, on the event $\bigcap_{i\in\{1,2,3\}} E_i \cap E_i' \cap E_i''$,

$$\begin{aligned}
\text{er}_{\mathcal{P}}(h_{\text{maj}}; f^\star) &\leq 12\mathbb{E}\left[\mathcal{P}\left(\text{ER}(h_I) \cap \text{ER}(\tilde{h})\right)\middle|S\right] \\
&\leq 12 \max_{i\in\{1,2,3\}} \max_{j\in\{1,2,3\}\setminus\{i\}} \max_{h\in L(\mathbb{A}(S_0;T_j))} \mathcal{P}(\text{ER}(h_i) \cap \text{ER}(h)) \\
&< \frac{1800}{m+1}\left(d + \ln\left(\frac{18}{\delta}\right)\right) = \frac{c}{m+1}\left(d + \ln\left(\frac{18}{\delta}\right)\right).
\end{aligned}$$

Furthermore, by the union bound, the event $\bigcap_{i\in\{1,2,3\}} E_i \cap E_i' \cap E_i''$ has probability at least $1-\delta$. Thus, since this argument holds for any $\delta \in (0,1)$ and any finite sequence T with $f^\star \in \mathbb{C}[T]$, we have succeeded in extending the inductive hypothesis to include $m' = m$.

By the principle of induction, we have established the claim that, $\forall m \in \mathbb{N}$, $\forall \delta \in (0,1)$, for every finite sequence T of points in $\mathcal{X} \times \mathcal{Y}$ with $f^\star \in \mathbb{C}[T]$, with probability at least $1-\delta$,

$$\text{er}_{\mathcal{P}}(\hat{h}_{m,T}; f^\star) \leq \frac{c}{m+1}\left(d + \ln\left(\frac{18}{\delta}\right)\right). \tag{9}$$

To complete the proof, we simply take $T = \emptyset$ (the empty sequence), and note that, for any $\varepsilon, \delta \in (0,1)$, for any value $m \in \mathbb{N}$ of size at least

$$\left\lfloor \frac{c}{\varepsilon}\left(d + \ln\left(\frac{18}{\delta}\right)\right)\right\rfloor, \tag{10}$$

the right hand side of (9) is less than ε, so that Majority$(L(\mathbb{A}(\cdot;\emptyset)))$ achieves a sample complexity equal the expression in (10). In particular, this implies

$$\mathcal{M}(\varepsilon, \delta) \leq \frac{c}{\varepsilon}\left(d + \ln\left(\frac{18}{\delta}\right)\right) = O\left(\frac{1}{\varepsilon}\left(d + \text{Log}\left(\frac{1}{\delta}\right)\right)\right).$$

■

5. Remarks

On the issue of computational complexity, we note that the construction of subsamples by $\mathbb{A}$ can be quite efficient. Since the branching factor is 3, while $|S_0|$ is reduced by roughly

a factor of 4 with each recursive call, the total number of subsamples returned by $\mathbb{A}(S;\emptyset)$ is a sublinear function of $|S|$. Furthermore, with appropriate data structures, the operations within each node of the recursion tree can be performed in constant time. Indeed, as discussed above, one can directly determine which data points to include in each subsample via a simple function of the indices, so that construction of these subsamples truly is computationally easy.

The only remaining significant computational issue in the learning algorithm is then the efficiency of the sample-consistent base learner L. The existence of such an algorithm L, with running time polynomial in the size of the input sequence and d, has been the subject of much investigation for a variety of concept spaces $\mathbb{C}$ (e.g., Khachiyan, 1979; Karmarkar, 1984; Valiant, 1984; Pitt and Valiant, 1988; Helmbold, Sloan, and Warmuth, 1990). For instance, the commonly-used concept space of *linear separators* admits such an algorithm (where $L(S)$ may be expressed as a solution of a system of linear inequalities). One can easily extend Theorem 2 to admit base learners L that are *improper* (i.e., which may return classifiers not contained in $\mathbb{C}$), as long as they are guaranteed to return a sample-consistent classifier in *some* hypothesis space $\mathcal{H}$ of VC dimension $O(d)$. Furthermore, as discussed by Pitt and Valiant (1988) and Haussler, Kearns, Littlestone, and Warmuth (1991), there is a simple technique for efficiently converting *any* efficient PAC learning algorithm for $\mathbb{C}$, returning classifiers in $\mathcal{H}$, into an efficient algorithm L for finding (with probability $1-\delta'$) a classifier in $\mathcal{H}$ consistent with a given data set S with $\mathbb{C}[S] \neq \emptyset$. Additionally, though the analysis above takes L to be deterministic, this merely serves to simplify the notation in the proof, and it is straightforward to generalize the proof to allow randomized base learners L, including those that fail to return a sample-consistent classifier with some probability δ' taken sufficiently small (e.g., $\delta' = \delta/(2|\mathbb{A}(S;\emptyset)|)$). Composing these facts, we may conclude that, for any concept space $\mathbb{C}$ that is efficiently PAC learnable using a hypothesis space $\mathcal{H}$ of VC dimension $O(d)$, there exists an efficient PAC learning algorithm for $\mathbb{C}$ with optimal sample complexity (up to numerical constant factors).

We conclude by noting that the constant factors obtained in the above proof are quite large. Some small refinements are possible within the current approach: for instance, by choosing the S_i subsequences slightly larger (e.g., $(3/10)|S|$), or using a tighter form of the Chernoff bound when lower-bounding N_i. However, there are inherent limitations to the approach used here, so that reducing the constant factors by more than, say, one order of magnitude, may require significant changes to some part of the analysis, and perhaps the algorithm itself. For this reason, it seems the next step in the study of $\mathcal{M}(\varepsilon,\delta)$ should be to search for strategies yielding refined constant factors. In particular, Warmuth (2004) has conjectured that the one-inclusion graph prediction algorithm also achieves a sample complexity of the optimal form. This conjecture remains open at this time. The one-inclusion graph predictor is known to achieve the optimal sample complexity in the closely-related *prediction model* of learning (where the objective is to achieve *expected* error rate at most ε), with a numerical constant factor very close to optimal (Haussler, Littlestone, and Warmuth, 1994). It therefore seems likely that a (positive) resolution of Warmuth's one-inclusion graph conjecture may also lead to improvements in constant factors compared to the bound on $\mathcal{M}(\varepsilon,\delta)$ established in the present work.

Acknowledgments

I would like to express my sincere thanks to Hans Simon and Amit Daniely for helpful comments on a preliminary attempt at a solution.

Appendix A. A Technical Lemma

The following basic lemma is useful in the proof of Theorem 2.[5]

Lemma 5 *For any $a, b, c_1 \in [1, \infty)$ and $c_2 \in [0, \infty)$,*

$$a \ln\left(c_1\left(c_2 + \frac{b}{a}\right)\right) \leq a \ln\left(c_1(c_2 + e)\right) + \frac{1}{e}b.$$

Proof If $\frac{b}{a} \leq e$, then monotonicity of $\ln(\cdot)$ implies

$$a \ln\left(c_1\left(c_2 + \frac{b}{a}\right)\right) \leq a \ln(c_1(c_2 + e)) \leq a \ln(c_1(c_2 + e)) + \frac{1}{e}b.$$

On the other hand, if $\frac{b}{a} > e$, then

$$a \ln\left(c_1\left(c_2 + \frac{b}{a}\right)\right) \leq a \ln\left(c_1 \max\{c_2, 2\}\frac{b}{a}\right) = a \ln\left(c_1 \max\{c_2, 2\}\right) + a \ln\left(\frac{b}{a}\right).$$

The first term in the rightmost expression is at most $a \ln(c_1(c_2+2)) \leq a \ln(c_1(c_2+e))$. The second term in the rightmost expression can be rewritten as $b\frac{\ln(b/a)}{b/a}$. Since $x \mapsto \ln(x)/x$ is nonincreasing on (e, ∞), in the case $\frac{b}{a} > e$, this is at most $\frac{1}{e}b$. Together, we have that

$$a \ln\left(c_1\left(c_2 + \frac{b}{a}\right)\right) \leq a \ln(c_1(c_2 + e)) + \frac{1}{e}b$$

in this case as well. ■

References

P. Auer and R. Ortner. A new PAC bound for intersection-closed concept classes. *Machine Learning*, 66(2-3):151–163, 2007. 3

M.-F. Balcan and P. M. Long. Active and passive learning of linear separators under log-concave distributions. In *Proceedings of the* 26th *Conference on Learning Theory*, 2013. 3

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4): 929–965, 1989. 2, 3, 3, 4.3

5. This lemma and proof also appear in a sibling paper (Hanneke, 2015).

N. H. Bshouty, Y. Li, and P. M. Long. Using the doubling dimension to analyze the generalization of learning algorithms. *Journal of Computer and System Sciences*, 75(6): 323–335, 2009. 3

M. Darnstädt. The optimal PAC bound for intersection-closed concept classes. *Information Processing Letters*, 115(4):458–461, 2015. 3

A. Ehrenfeucht, D. Haussler, M. Kearns, and L. G. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82:247–261, 1989. 2, 3

E. Giné and V. Koltchinskii. Concentration inequalities and asymptotic results for ratio type empirical processes. *The Annals of Probability*, 34(3):1143–1216, 2006. 3

S. Hanneke. *Theoretical Foundations of Active Learning*. PhD thesis, Machine Learning Department, School of Computer Science, Carnegie Mellon University, 2009. 3, 4.1

S. Hanneke. Refined error bounds for several learning algorithms. *arXiv:1512.07146*, 2015. 3, 5

D. Haussler, M. Kearns, N. Littlestone, and M. K. Warmuth. Equivalence of models of polynomial learnability. *Information and Computation*, 95(2):129–161, 1991. 5

D. Haussler, N. Littlestone, and M. K. Warmuth. Predicting $\{0,1\}$-functions on randomly drawn points. *Information and Computation*, 115:248–292, 1994. 3, 3, 5

D. Helmbold, R. Sloan, and M. K. Warmuth. Learning nested differences of intersection-closed concept classes. *Machine Learning*, 5(2):165–196, 1990. 5

N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984. 5

L. G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979. 5

P. M. Long. An upper bound on the sample complexity of PAC learning halfspaces with respect to the uniform distribution. *Information Processing Letters*, 87(5):229–234, 2003. 3

L. Pitt and L. G. Valiant. Computational limitations on learning from examples. *Journal of the Association for Computing Machinery*, 35(4):965–984, 1988. 5

H. Simon. An almost optimal PAC algorithm. In *Proceedings of the* 28th *Conference on Learning Theory*, 2015. 1, 3, 4.1

L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. 1, 5

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer, 1996. 2

V. Vapnik. *Estimation of Dependencies Based on Empirical Data*. Springer-Verlag, New York, 1982. 2, 3, 4.3

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971. 2

M. K. Warmuth. The optimal PAC algorithm. In *Proceedings of the* 17^{th} *Conference on Learning Theory*, 2004. 3, 5

Journal of Machine Learning Research 17 (2016) 1-40 Submitted 10/15; Revised 4/16; Published 4/16

End-to-End Training of Deep Visuomotor Policies

Sergey Levine[†] SVLEVINE@EECS.BERKELEY.EDU
Chelsea Finn[†] CBFINN@EECS.BERKELEY.EDU
Trevor Darrell TREVOR@EECS.BERKELEY.EDU
Pieter Abbeel PABBEEL@EECS.BERKELEY.EDU
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA
[†]These authors contributed equally.

Editor: Jan Peters

Abstract

Policy search methods can allow robots to learn control policies for a wide range of tasks, but practical applications of policy search often require hand-engineered components for perception, state estimation, and low-level control. In this paper, we aim to answer the following question: does training the perception and control systems jointly end-to-end provide better performance than training each component separately? To this end, we develop a method that can be used to learn policies that map raw image observations directly to torques at the robot's motors. The policies are represented by deep convolutional neural networks (CNNs) with 92,000 parameters, and are trained using a guided policy search method, which transforms policy search into supervised learning, with supervision provided by a simple trajectory-centric reinforcement learning method. We evaluate our method on a range of real-world manipulation tasks that require close coordination between vision and control, such as screwing a cap onto a bottle, and present simulated comparisons to a range of prior policy search methods.

Keywords: Reinforcement Learning, Optimal Control, Vision, Neural Networks

1. Introduction

Robots can perform impressive tasks under human control, including surgery (Lanfranco et al., 2004) and household chores (Wyrobek et al., 2008). However, designing the perception and control software for autonomous operation remains a major challenge, even for basic tasks. Policy search methods hold the promise of allowing robots to automatically learn new behaviors through experience (Kober et al., 2010b; Deisenroth et al., 2011; Kalakrishnan et al., 2011; Deisenroth et al., 2013). However, policies learned using such methods often rely on a number of hand-engineered components for perception and control, so as to present the policy with a more manageable and low-dimensional representation of observations and actions. The vision system in particular can be complex and prone to errors, and it is typically not improved during policy training, nor adapted to the goal of the task.

In this article, we aim to answer the following question: can we acquire more effective policies for sensorimotor control if the perception system is trained jointly with the control policy, rather than separately? In order to represent a policy that performs both

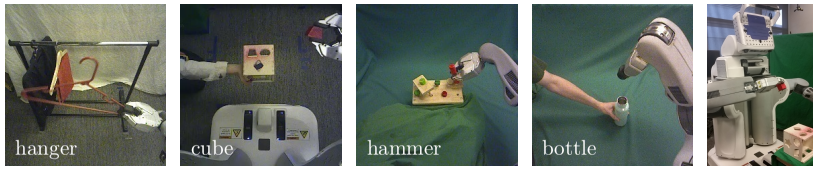


Figure 1: Our method learns visuomotor policies that directly use camera image observations (left) to set motor torques on a PR2 robot (right).

perception and control, we use deep neural networks. Deep neural network representations have recently seen widespread success in a variety of domains, such as computer vision and speech recognition, and even playing video games. However, using deep neural networks for real-world sensorimotor policies, such as robotic controllers that map image pixels and joint angles to motor torques, presents a number of unique challenges. Successful applications of deep neural networks typically rely on large amounts of data and direct supervision of the output, neither of which is available in robotic control. Real-world robot interaction data is scarce, and task completion is defined at a high level by means of a cost function, which means that the learning algorithm must determine on its own which action to take at each point. From the control perspective, a further complication is that observations from the robot's sensors do not provide us with the full state of the system. Instead, important state information, such as the positions of task-relevant objects, must be inferred from inputs such as camera images.

We address these challenges by developing a guided policy search algorithm for sensorimotor deep learning, as well as a novel CNN architecture designed for robotic control. Guided policy search converts policy search into supervised learning, by iteratively constructing the training data using an efficient model-free trajectory optimization procedure. We show that this can be formalized as an instance of Bregman ADMM (BADMM) (Wang and Banerjee, 2014), which can be used to show that the algorithm converges to a locally optimal solution. In our method, the full state of the system is observable at training time, but not at test time. For most tasks, providing the full state simply requires positioning objects in one of several known positions for each trial during training. At test time, the learned CNN policy can handle novel, unknown configurations, and no longer requires full state information. Since the policy is optimized with supervised learning, we can use standard methods like stochastic gradient descent for training. Our CNNs have 92,000 parameters and 7 layers, including a novel spatial feature point transformation that provides accurate spatial reasoning and reduces overfitting. This allows us to train our policies with relatively modest amounts of data and only tens of minutes of real-world interaction time.

We evaluate our method by learning policies for inserting a block into a shape sorting cube, screwing a cap onto a bottle, fitting the claw of a toy hammer under a nail with various grasps, and placing a coat hanger on a rack with a PR2 robot (see Figure 1). These tasks require localization, visual tracking, and handling complex contact dynamics. Our results demonstrate improvements in consistency and generalization from training visuomotor policies end-to-end, when compared to training the vision and control components separately. We also present simulated comparisons that show that guided policy search outperforms a

number of prior methods when training high-dimensional neural network policies. Some of the material in this article has previously appeared in two conference papers (Levine and Abbeel, 2014; Levine et al., 2015), which we extend to introduce visual input into the policy.

2. Related Work

Reinforcement learning and policy search methods (Gullapalli, 1990; Williams, 1992) have been applied in robotics for playing games such as table tennis (Kober et al., 2010b), object manipulation (Gullapalli, 1995; Peters and Schaal, 2008; Kober et al., 2010a; Deisenroth et al., 2011; Kalakrishnan et al., 2011), locomotion (Benbrahim and Franklin, 1997; Kohl and Stone, 2004; Tedrake et al., 2004; Geng et al., 2006; Endo et al., 2008), and flight (Ng et al., 2004). Several recent papers provide surveys of policy search in robotics (Deisenroth et al., 2013; Kober et al., 2013). Such methods are typically applied to one component of the robot control pipeline, which often sits on top of a hand-designed controller, such as a PD controller, and accepts processed input, for example from an existing vision pipeline (Kalakrishnan et al., 2011). Our method learns policies that map visual input and joint encoder readings directly to the torques at the robot's joints. By learning the entire mapping from perception to control, the perception layers can be adapted to optimize task performance, and the motor control layers can be adapted to imperfect perception.

We represent our policies with convolutional neural networks (CNNs). CNNs have a long history in computer vision and deep learning (Fukushima, 1980; LeCun et al., 1989; Schmidhuber, 2015), and have recently gained prominence due to excellent results on a number of vision benchmarks (Ciresan et al., 2011; Krizhevsky et al., 2012; Ciresan et al., 2012; Girshick et al., 2014a; Tompson et al., 2014; LeCun et al., 2015; He et al., 2015). Most applications of CNNs focus on classification, where locational information is discarded by means of successive pooling layers to provide for invariance (Lee et al., 2009). Applications to localization typically either use a sliding window (Girshick et al., 2014a) or object proposals (Endres and Hoiem, 2010; Uijlings et al., 2013; Girshick et al., 2014b) to localize the object, reducing the task to classification, perform regression to a heatmap of manually labeled keypoints (Tompson et al., 2014), requiring precise knowledge of the object position in the image and camera calibration, or use 3D models to localize previously scanned objects (Pepik et al., 2012; Savarese and Fei-Fei, 2007). Many prior robotic applications of CNNs do not directly consider control, but employ CNNs for the perception component of a larger robotic system (Hadsell et al., 2009; Sung et al., 2015; Lenz et al., 2015b; Pinto and Gupta, 2015). We use a novel CNN architecture for our policies that automatically learn feature points that capture spatial information about the scene, without any supervision beyond the information from the robot's encoders and camera.

Applications of deep learning in robotic control have been less prevalent in recent years than in visual recognition. Backpropagation through the dynamics and the image formation process is typically impractical, since they are often non-differentiable, and such long-range backpropagation can lead to extreme numerical instability, since the linearization of a suboptimal policy is likely to be unstable. This issue has also been observed in the related context of recurrent neural networks (Hochreiter et al., 2001; Pascanu and Bengio, 2012). The high dimensionality of the network also makes reinforcement learning difficult (Deisenroth et al., 2013). Pioneering early work on neural network control used

small, simple networks (Pomerleau, 1989; Hunt et al., 1992; Bekey and Goldberg, 1992; Lewis et al., 1998; Bakker et al., 2003; Mayer et al., 2006), and has largely been supplanted by methods that use carefully designed policies that can be learned efficiently with reinforcement learning (Kober et al., 2013). More recent work on sensorimotor deep learning has tackled simple task-space motions (Lenz et al., 2015a; Lampe and Riedmiller, 2013) and used unsupervised learning to obtain low-dimensional state spaces from images (Lange et al., 2012). Such methods have been demonstrated on tasks with a low-dimensional underlying structure: Lenz et al. (2015a) controls the end-effector in 2D space, while Lange et al. (2012) controls a 2-dimensional slot car with 1-dimensional actions. Our experiments include full torque control of 7-DoF robotic arms interacting with objects, with 30-40 state dimensions. In simple synthetic environments, control from images has been addressed with image features (Jodogne and Piater, 2007), nonparametric methods (van Hoof et al., 2015), and unsupervised state-space learning (Böhmer et al., 2013; Jonschkowski and Brock, 2014). CNNs have also been trained to play video games with Q-learning, Monte Carlo tree search, and stochastic search (Mnih et al., 2013; Koutník et al., 2013; Guo et al., 2014), and have been applied to simple simulated control tasks (Watter et al., 2015; Lillicrap et al., 2015). However, such methods have only been demonstrated on synthetic domains that lack the visual complexity of the real world, and require an impractical number of samples for real-world robotic learning. Our method is sample efficient, requiring only minutes of interaction time. To the best of our knowledge, this is the first method that can train deep visuomotor policies for complex, high-dimensional manipulation skills with direct torque control.

Learning visuomotor policies on a real robot requires handling complex observations and high dimensional policy representations. We tackle these challenges using guided policy search. In guided policy search, the policy is optimized using supervised learning, which scales gracefully with the dimensionality of the policy. The training set for supervised learning can be constructed using trajectory optimization under known dynamics (Levine and Koltun, 2013a,b, 2014; Mordatch and Todorov, 2014) and trajectory-centric reinforcement learning methods that operate under unknown dynamics (Levine and Abbeel, 2014; Levine et al., 2015), which is the approach taken in this work. In both cases, the supervision is adapted to the policy, to ensure that the final policy can reproduce the training data. The use of supervised learning in the inner loop of iterative policy search has also been proposed in the context of imitation learning (Ross et al., 2011, 2013). However, such methods typically do not address the question of how the supervision should be adapted to the policy.

The goal of our approach is also similar to visual servoing, which performs feedback control on feature points in a camera image (Espiau et al., 1992; Mohta et al., 2014; Wilson et al., 1996). However, our visuomotor policies are entirely learned from real-world data, and do not require feature points or feedback controllers to be specified by hand. This allows our method much more flexibility in choosing how to use the visual signal. Our approach also does not require any sort of camera calibration, in contrast to many visual servoing methods (though not all – see e.g. Jägersand et al. (1997); Yoshimi and Allen (1994)).

3. Background and Overview

In this section, we define the visuomotor policy learning problem and present an overview of our approach. The core component of our approach is a guided policy search algorithm

that separates the problem of learning visuomotor policies into separate supervised learning and trajectory learning phases, each of which is easier than optimizing the policy directly. We also discuss a policy architecture suitable for end-to-end learning of vision and control, and a training setup that allows our method to be applied to real robotic platforms.

3.1 Definitions and Problem Formulation

In policy search, the goal is to learn a policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ that allows an agent to choose actions $\mathbf{u}_t$ in response to observations $\mathbf{o}_t$ to control a dynamical system, such as a robot. The policy comes from some parametric class parameterized by θ, which could be, for example, the weights of a neural network. The system is defined by states $\mathbf{x}_t$, actions $\mathbf{u}_t$, and observations $\mathbf{o}_t$. For example, $\mathbf{x}_t$ might include the joint angles of the robot, the positions of objects in the world, and their time derivatives, $\mathbf{u}_t$ might consist of motor torque commands, and $\mathbf{o}_t$ might include an image from the robot's onboard camera. In this paper, we address finite horizon episodic tasks with $t \in [1, \ldots, T]$. The states evolve in time according to the system dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$, and the observations are, in general, a stochastic consequence of the states, according to $p(\mathbf{o}_t|\mathbf{x}_t)$. Neither the dynamics nor the observation distribution are assumed to be known in general. For notational convenience, we will use $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ to denote the distribution over actions under the policy conditioned on the state. However, since the policy is conditioned on the observation $\mathbf{o}_t$, this distribution is in fact given by $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t) = \int \pi_\theta(\mathbf{u}_t|\mathbf{o}_t)p(\mathbf{o}_t|\mathbf{x}_t)d\mathbf{o}_t$. The dynamics and $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ together induce a distribution over trajectories $\tau = \{\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2, \mathbf{u}_2, \ldots, \mathbf{x}_T, \mathbf{u}_T\}$:

$$\pi_\theta(\tau) = p(\mathbf{x}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{u}_t|\mathbf{x}_t) p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t).$$

The goal of a task is given by a cost function $\ell(\mathbf{x}_t, \mathbf{u}_t)$, and the objective in policy search is to minimize the expectation $E_{\pi_\theta(\tau)}[\sum_{t=1}^{T} \ell(\mathbf{x}_t, \mathbf{u}_t)]$, which we will abbreviate as $E_{\pi_\theta(\tau)}[\ell(\tau)]$. A summary of the notation used in the paper is provided in Table 1.

3.2 Approach Summary

Our methods consists of two main components, which are illustrated in Figure 3. The first is a supervised learning algorithm that trains policies of the form $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t) = \mathcal{N}(\mu^\pi(\mathbf{o}_t), \Sigma^\pi(\mathbf{o}_t))$, where both $\mu^\pi(\mathbf{o}_t)$ and $\Sigma^\pi(\mathbf{o}_t)$ are general nonlinear functions. In our implementation, $\mu^\pi(\mathbf{o}_t)$ is a deep convolutional neural network, while $\Sigma^\pi(\mathbf{o}_t)$ is an observation-independent learned covariance, though other representations are possible. The second component is a trajectory-centric reinforcement learning (RL) algorithm that generates guiding distributions $p_i(\mathbf{u}_t|\mathbf{x}_t)$ that provide the supervision used to train the policy. These two components form a policy search algorithm that can be used to learn complex robotic tasks using only a high-level cost function $\ell(\mathbf{x}_t, \mathbf{u}_t)$. During training, only samples from the guiding distributions $p_i(\mathbf{u}_t|\mathbf{x}_t)$ are generated by running rollouts on the physical system, which avoids the need to execute partially trained neural network policies on physical hardware.

Supervised learning will not, in general, produce a policy with good long-horizon performance, since a small mistake on the part of the policy will place the system into states that are outside the distribution in the training data, causing compounding errors. To

symbol	definition	example/details
$\mathbf{x}_t$	Markovian system state at time step $t \in [1, T]$	joint angles, end-effector pose, object positions, and their velocities; dimensionality: 14 to 32
$\mathbf{u}_t$	control or action at time step $t \in [1, T]$	joint motor torque commands; dimensionality: 7 (for the PR2 robot)
$\mathbf{o}_t$	observation at time step $t \in [1, T]$	RGB camera image, joint encoder readings & velocities, end-effector pose; dimensionality: around 200,000
τ	trajectory: $\tau = \{\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2, \mathbf{u}_2, \ldots, \mathbf{x}_T, \mathbf{u}_T\}$	notational shorthand for a sequence of states and actions
$\ell(\mathbf{x}_t, \mathbf{x}_t)$	cost function that defines the goal of the task	distance between an object in the gripper and the target
$p(\mathbf{x}_{t+1}\|\mathbf{x}_t, \mathbf{u}_t)$	unknown system dynamics	physics that govern the robot and any objects it interacts with
$p(\mathbf{o}_t\|\mathbf{x}_t)$	unknown observation distribution	stochastic process that produces camera images from system state
$\pi_\theta(\mathbf{u}_t\|\mathbf{o}_t)$	learned nonlinear global policy parameterized by weights θ	convolutional neural network, such as the one in Figure 2
$\pi_\theta(\mathbf{u}_t\|\mathbf{x}_t)$	$\int \pi_\theta(\mathbf{u}_t\|\mathbf{o}_t)p(\mathbf{o}_t\|\mathbf{x}_t)d\mathbf{o}_t$	notational shorthand for observation-based policy conditioned on state
$p_i(\mathbf{u}_t\|\mathbf{x}_t)$	learned local time-varying linear-Gaussian controller for initial state $\mathbf{x}_1^i$	time-varying linear-Gaussian controller has form $\mathcal{N}(\mathbf{K}_{ti}\mathbf{x}_t + \mathbf{k}_{ti}, \mathbf{C}_{ti})$
$\pi_\theta(\tau)$	trajectory distribution for $\pi_\theta(\mathbf{u}_t\|\mathbf{x}_t)$: $p(\mathbf{x}_1)\prod_{t=1}^{T} \pi_\theta(\mathbf{u}_t\|\mathbf{x}_t)p(\mathbf{x}_{t+1}\|\mathbf{x}_t, \mathbf{u}_t)$	notational shorthand for trajectory distribution induced by a policy

Table 1: Summary of the notation frequently used in this article.

avoid this issue, the training data must come from the policy's own state distribution (Ross et al., 2011). We achieve this by alternating between trajectory-centric RL and supervised learning. The RL stage adapts to the current policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$, providing supervision at states that are iteratively brought closer to the states visited by the policy. This is formalized as a variant of the BADMM algorithm (Wang and Banerjee, 2014) for constrained optimization, which can be used to show that, at convergence, the policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ and the guiding distributions $p_i(\mathbf{u}_t|\mathbf{x}_t)$ will exhibit the same behavior. This algorithm is derived in Section 4. The guiding distributions are substantially easier to optimize than learning the policy parameters directly (e.g., using model-free reinforcement learning), because they use the full state of the system $\mathbf{x}_t$, while the policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ only uses the observations. This means that the method requires the full state to be known during training, but not at test time. This makes it possible to efficiently learn complex visuomotor policies, but imposes additional assumptions on the observability of $\mathbf{x}_t$ during training that we discuss in Section 4.

When learning visuomotor tasks, the policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ is represented by a novel convolutional neural network (CNN) architecture, which we describe in Section 5.2. CNNs have enjoyed considerable success in computer vision (LeCun et al., 2015), but the most popular

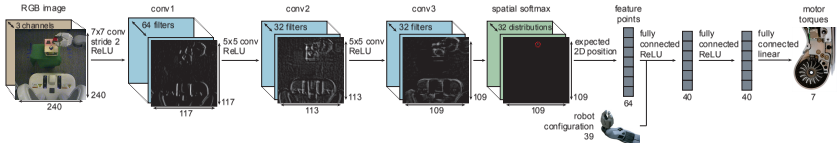


Figure 2: Visuomotor policy architecture. The network contains three convolutional layers, followed by a spatial softmax and an expected position layer that converts pixel-wise features to feature points, which are better suited for spatial computations. The points are concatenated with the robot configuration, then passed through three fully connected layers to produce the torques.

architectures rely on large datasets and focus on semantic tasks such as classification, often intentionally discarding spatial information. Our architecture, illustrated in Figure 2, uses a fixed transformation from the last convolutional layer to a set of spatial feature points, which form a concise representation of the visual scene suitable for feedback control. Our network has 7 layers and around 92,000 parameters, which presents a major challenge for standard policy search methods (Deisenroth et al., 2013).

To reduce the amount of experience needed to train visuomotor policies, we also introduce a pretraining scheme that allows us to train effective policies with a relatively small number of iterations. The pretraining steps are illustrated in Figure 3. The intuition behind our pretraining is that, although we ultimately seek to obtain sensorimotor policies that combine both vision and control, low-level aspects of vision can be initialized independently. To that end, we pretrain the convolutional layers of our network by predicting elements of $\mathbf{x}_t$ that are not provided in the observation $\mathbf{o}_t$, such as the positions of objects in the scene. We also initially train the guiding trajectory distributions $p_i(\mathbf{u}_t|\mathbf{x}_t)$ independently of the convolutional network until the trajectories achieve a basic level of competence at the task, and then switch to full guided policy search with end-to-end training of $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$. In our implementation, we also initialize the first layer filters from the model of Szegedy et al. (2014), which is trained on ImageNet (Deng et al., 2009) classification. The initialization and pretraining scheme is described in Section 5.2.

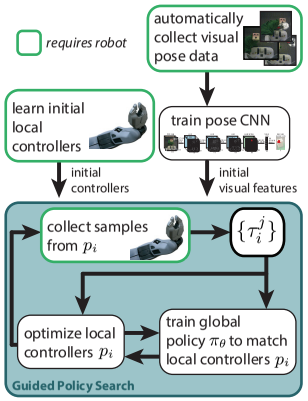


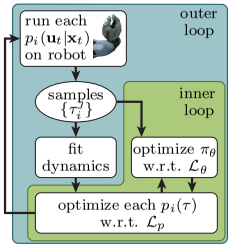
Figure 3: Diagram of our approach, including the main guided policy search phase and initialization phases.

4. Guided Policy Search with BADMM

Guided policy search transforms policy search into a supervised learning problem, where the training set is generated by a simple trajectory-centric RL algorithm. This algorithm

optimizes linear-Gaussian controllers $p_i(\mathbf{u}_t|\mathbf{x}_t)$, and is described in Section 4.2. We refer to the trajectory distribution induced by $p_i(\mathbf{u}_t|\mathbf{x}_t)$ as $p_i(\tau)$. Each $p_i(\mathbf{u}_t|\mathbf{x}_t)$ succeeds from different initial states. For example, in the task of placing a cap on a bottle, these initial states correspond to different positions of the bottle. By training on trajectories for multiple bottle positions, the final CNN policy can succeed from all initial states, and can generalize to other states from the same distribution.

The final policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ learned with guided policy search is only provided with observations $\mathbf{o}_t$ of the full state $\mathbf{x}_t$, and the dynamics are assumed to be unknown. A diagram of this method, which corresponds to an expanded version of the guided policy search box in Figure 3, is shown on the right. In the outer loop, we draw sample trajectories $\{\tau_i^j\}$ for each initial state on the physical system by running the corresponding controller $p_i(\mathbf{u}_t|\mathbf{x}_t)$. The samples are used to fit the dynamics $p_i(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ that are used to improve $p_i(\mathbf{u}_t|\mathbf{x}_t)$, and serve as training data for the policy. The inner loop alternates between optimizing each $p_i(\tau)$ and optimizing the policy to match these trajectory distributions. The policy is trained to predict the actions along each trajectory from the observations $\mathbf{o}_t$, rather than the full state $\mathbf{x}_t$. This allows the policy to directly use raw observations at test time. This alternating optimization can be framed as an instance of the BADMM algorithm (Wang and Banerjee, 2014), which converges to a solution where the trajectory distributions and the policy have the same state distribution. This allows greedy supervised training of the policy to produce a policy with good long-horizon performance.



4.1 Algorithm Derivation

Policy search methods minimize the expected cost $E_{\pi_\theta}[\ell(\tau)]$, where $\tau = \{\mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_T, \mathbf{u}_T\}$ is a trajectory, and $\ell(\tau) = \sum_{t=1}^{T} \ell(\mathbf{x}_t, \mathbf{u}_t)$ is the cost of an episode. In the fully observed case, the expectation is taken under $\pi_\theta(\tau) = p(\mathbf{x}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{u}_t|\mathbf{x}_t) p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$. The final policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ is conditioned on the observations $\mathbf{o}_t$, but $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ can be recovered as $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t) = \int \pi_\theta(\mathbf{u}_t|\mathbf{o}_t) p(\mathbf{o}_t|\mathbf{x}_t) d\mathbf{o}_t$. We will present the derivation in this section for $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$, but we do not require knowledge of $p(\mathbf{o}_t|\mathbf{x}_t)$ in the final algorithm. As discussed in Section 4.3, the integral will be evaluated with samples from the real system, which include both $\mathbf{x}_t$ and $\mathbf{o}_t$. We begin by rewriting the expected cost minimization as a constrained problem:

$$\min_{p, \pi_\theta} E_p[\ell(\tau)] \text{ s.t. } p(\mathbf{u}_t|\mathbf{x}_t) = \pi_\theta(\mathbf{u}_t|\mathbf{x}_t) \; \forall \, \mathbf{x}_t, \mathbf{u}_t, t, \tag{1}$$

where we will refer to $p(\tau)$ as a guiding distribution. This formulation is equivalent to the original problem, since the constraint forces the two distributions to be identical. However, if we approximate the initial state distribution $p(\mathbf{x}_1)$ with samples $\mathbf{x}_1^i$, we can choose $p(\tau)$ to be a class of distributions that is much easier to optimize than π_θ, as we will show later. This will allow us to use simple local learning methods for $p(\tau)$, without needing to train the complex neural network policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ directly with reinforcement learning, which would require a prohibitive amount of experience on real physical systems.

The constrained problem can be solved by a dual descent method, which alternates between minimizing the Lagrangian with respect to the primal variables, and incrementing

the Lagrange multipliers by their subgradient. Minimization of the Lagrangian with respect to $p(\tau)$ and θ is done in alternating fashion: minimizing with respect to θ corresponds to supervised learning (making π_θ match $p(\tau)$), and minimizing with respect to $p(\tau)$ consists of one or more trajectory optimization problems. The dual descent method we use is based on BADMM (Wang and Banerjee, 2014), a variant of ADMM (Boyd et al., 2011) that augments the Lagrangian with a Bregman divergence between the constrained variables. We use the KL-divergence as the Bregman constraint, which is particularly convenient for working with probability distributions. We will also modify the constraint $p(\mathbf{u}_t|\mathbf{x}_t) = \pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ by multiplying both sides by $p(\mathbf{x}_t)$, to get $p(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t) = \pi_\theta(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)$. This constraint is equivalent, but has the convenient property that we can express the Lagrangian in terms of expectations. The BADMM augmented Lagrangians for θ and p are therefore given by

$$\mathcal{L}_\theta(\theta, p) = \sum_{t=1}^{T} E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t)] + E_{p(\mathbf{x}_t)\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)}[\lambda_{\mathbf{x}_t,\mathbf{u}_t}] - E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\lambda_{\mathbf{x}_t,\mathbf{u}_t}] + \nu_t \phi_t^\theta(\theta, p)$$
$$\mathcal{L}_p(p, \theta) = \sum_{t=1}^{T} E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t)] + E_{p(\mathbf{x}_t)\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)}[\lambda_{\mathbf{x}_t,\mathbf{u}_t}] - E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\lambda_{\mathbf{x}_t,\mathbf{u}_t}] + \nu_t \phi_t^p(\theta, p),$$

where $\lambda_{\mathbf{x}_t,\mathbf{u}_t}$ is the Lagrange multiplier for state $\mathbf{x}_t$ and action $\mathbf{u}_t$ at time t, and $\phi_t^\theta(\theta, p)$ are $\phi_t^p(\theta, p)$ are expectations of the KL-divergences:

$$\phi_t^p(p, \theta) = E_{p(\mathbf{x}_t)}[D_{\mathrm{KL}}(p(\mathbf{u}_t|\mathbf{x}_t)\|\pi_\theta(\mathbf{u}_t|\mathbf{x}_t))]$$
$$\phi_t^\theta(\theta, p) = E_{p(\mathbf{x}_t)}[D_{\mathrm{KL}}(\pi_\theta(\mathbf{u}_t|\mathbf{x}_t))\|p(\mathbf{u}_t|\mathbf{x}_t)].$$

Dual descent with alternating primal minimization is then described by the following steps:

$$\theta \leftarrow \arg\min_\theta \sum_{t=1}^{T} E_{p(\mathbf{x}_t)\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)}[\lambda_{\mathbf{x}_t,\mathbf{u}_t}] + \nu_t \phi_t^\theta(\theta, p)$$
$$p \leftarrow \arg\min_p \sum_{t=1}^{T} E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t) - \lambda_{\mathbf{x}_t,\mathbf{u}_t}] + \nu_t \phi_t^p(p, \theta)$$
$$\lambda_{\mathbf{x}_t,\mathbf{u}_t} \leftarrow \lambda_{\mathbf{x}_t,\mathbf{u}_t} + \alpha\nu_t(\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t) - p(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)).$$

This procedure is an instance of BADMM, and therefore inherits its convergence guarantees. Note that we drop terms that are independent of the optimization variables on each line. The parameter α is a step size. As with most augmented Lagrangian methods, the weight ν_t is set heuristically, as described in Appendix A.1.

The dynamics only affect the optimization with respect to $p(\tau)$. In order to make this optimization efficient, we choose $p(\tau)$ to be a mixture of N Gaussians $p_i(\tau)$, one for each initial state sample $\mathbf{x}_1^i$. This makes the action conditionals $p_i(\mathbf{u}_t|\mathbf{x}_t)$ and the dynamics $p_i(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ linear-Gaussian, as discussed in Section 4.2. This is a reasonable choice when the system is deterministic, or the noise is Gaussian or small, and we found that this approach is sufficiently tolerant to noise for use on real physical systems. Our choice of p also assumes that the policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ is conditionally Gaussian. This is also reasonable, since the mean and covariance of $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ can be any nonlinear function of the observations

$\mathbf{o}_t$, which themselves are a function of the unobserved state $\mathbf{x}_t$. In Section 4.2, we show how these assumptions enable each $p_i(\tau)$ to be optimized very efficiently. We will refer to $p_i(\tau)$ as guiding distributions, since they serve to focus the policy on good, low-cost behaviors.

Aside from learning $p_i(\tau)$, we must choose a tractable way to represent the infinite set of constraints $p(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t) = \pi_\theta(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)$. One approximate approach proposed in prior work is to replace the exact constraints with expectations of features (Peters et al., 2010). When the features consist of linear, quadratic, or higher order monomial functions of the random variable, this can be viewed as a constraint on the moments of the distributions. If we only use the first moment, we get a constraint on the expected action: $E_{p(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)}[\mathbf{u}_t] = E_{\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)}[\mathbf{u}_t]$. If the stochasticity in the dynamics is low, as we assumed previously, the optimal solution for each $p_i(\tau)$ will have low entropy, making this first moment constraint a reasonable approximation. The KL-divergence terms in the augmented Lagrangians will still serve to softly enforce agreement between the higher moments. While this simplification is quite drastic, we found that it was more stable in practice than including higher moments, likely because these higher moments are harder to estimate accurately with a limited number of samples. The alternating optimization is now given by

$$\theta \leftarrow \arg\min_\theta \sum_{t=1}^{T} E_{p(\mathbf{x}_t)\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)}[\mathbf{u}_t^{\mathrm{T}}\lambda_{\mu t}] + \nu_t \phi_t^\theta(\theta, p) \tag{2}$$
$$p \leftarrow \arg\min_p \sum_{t=1}^{T} E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t) - \mathbf{u}_t^{\mathrm{T}}\lambda_{\mu t}] + \nu_t \phi_t^p(p, \theta) \tag{3}$$
$$\lambda_{\mu t} \leftarrow \lambda_{\mu t} + \alpha\nu_t(E_{\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)}[\mathbf{u}_t] - E_{p(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)}[\mathbf{u}_t]),$$

where $\lambda_{\mu t}$ is the Lagrange multiplier on the expected action at time t. In the rest of the paper, we will use $\mathcal{L}_\theta(\theta, p)$ and $\mathcal{L}_p(p, \theta)$ to denote the two augmented Lagrangians in Equations (2) and (3), respectively. In the next two sections, we will describe how $\mathcal{L}_p(p, \theta)$ can be optimized with respect to p under unknown dynamics, and how $\mathcal{L}_\theta(\theta, p)$ can be optimized for complex, high-dimensional policies. Implementation details of the BADMM optimization are presented in Appendix A.1.

4.2 Trajectory Optimization under Unknown Dynamics

Since the Lagrangian $\mathcal{L}_p(p, \theta)$ in the previous section factorizes over the mixture elements in $p(\tau) = \sum_i p_i(\tau)$, we describe the trajectory optimization method for a single Gaussian $p(\tau)$. When there are multiple mixture elements, this procedure is applied in parallel to each $p_i(\tau)$. Since $p(\tau)$ is Gaussian, the conditionals $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ and $p(\mathbf{u}_t|\mathbf{x}_t)$, which correspond to the dynamics and the controller, are time-varying linear-Gaussian, and given by

$$p(\mathbf{u}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{K}_t\mathbf{x}_t + \mathbf{k}_t, \mathbf{C}_t) \qquad p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(f_{\mathbf{x}t}\mathbf{x}_t + f_{\mathbf{u}t}\mathbf{u}_t + f_{ct}, \mathbf{F}_t).$$

This type of controller can be learned efficiently with a small number of real-world samples, making it a good choice for optimizing the guiding distributions. Since a different set of time-varying linear-Gaussian dynamics is fitted for each initial state, this dynamics representation can model any continuous deterministic system that can be locally linearized. Stochastic dynamics can violate the local linearity assumption in principle, but we found that in practice this representation was well suited for a wide variety of noisy real-world tasks.

The dynamics are determined by the environment. If they are known, $p(\mathbf{u}_t|\mathbf{x}_t)$ can be optimized with a variant of the iterative linear-quadratic-Gaussian regulator (iLQG) (Li and Todorov, 2004; Levine and Koltun, 2013a), which is a variant of DDP (Jacobson and Mayne, 1970). In the case of unknown dynamics, we can fit $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ to sample trajectories sampled from the trajectory distribution at the previous iteration, denoted $\hat{p}(\tau)$. If $\hat{p}(\tau)$ is too different from $p(\tau)$, these samples will not give a good estimate of $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$, and the optimization will diverge. To avoid this, we can bound the change from $\hat{p}(\tau)$ to $p(\tau)$ in terms of their KL-divergence by a step size ϵ, producing the following constrained problem:

$$\min_{p(\tau)\in\mathcal{N}(\tau)} \mathcal{L}_p(p, \theta) \text{ s.t. } D_{\mathrm{KL}}(p(\tau)\|\hat{p}(\tau)) \leq \epsilon.$$

This type of policy update has previously been proposed by several authors in the context of policy search (Bagnell and Schneider, 2003; Peters and Schaal, 2008; Peters et al., 2010; Levine and Abbeel, 2014). In the case when $p(\tau)$ is Gaussian, this problem can be solved efficiently using dual gradient descent, while the dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ are fitted to samples gathered by running the previous controller $\hat{p}(\mathbf{u}_t|\mathbf{x}_t)$ on the robot. Fitting a global Gaussian mixture model to tuples $(\mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t+1})$ and using it as a prior for fitting the dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ serves to greatly reduce the sample complexity. We describe the dynamics fitting procedure in detail in Appendix A.3.

Note that the trajectory optimization cost function $\mathcal{L}_p(p, \theta)$ also depends on the policy $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$, while we only have access to $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$. In order to compute a local quadratic expansion of the KL-divergence term $D_{\mathrm{KL}}(p(\mathbf{u}_t|\mathbf{x}_t)\|\pi_\theta(\mathbf{u}_t|\mathbf{x}_t))$ inside $\mathcal{L}_p(p, \theta)$ for iLQG, we also estimate a linearization of the mean of the conditionally Gaussian policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ with respect to the state $\mathbf{x}_t$, using the same procedure that we use to linearize the dynamics. The data for this estimation consists of tuples $\{\mathbf{x}_t^i, E_{\pi_\theta(\mathbf{u}_t|\mathbf{o}_t^i)}[\mathbf{u}_t]\}$, which we can obtain because both the states $\mathbf{x}_t^i$ and the observations $\mathbf{o}_t^i$ are available for all of the samples evaluated on the real physical system.

This constrained optimization is performed in the "inner loop" of the optimization described in the previous section, and the KL-divergence constraint $D_{\mathrm{KL}}(p(\tau)\|\hat{p}(\tau)) \leq \epsilon$ imposes a step size on the trajectory update. The overall algorithm then becomes an instance of generalized BADMM (Wang and Banerjee, 2014). Note that the augmented Lagrangian $\mathcal{L}_p(p, \theta)$ consists of an expectation under $p(\tau)$ of a quantity that is independent of p. We can locally approximate this quantity with a quadratic by using a quadratic expansion of $\ell(\mathbf{x}_t, \mathbf{u}_t)$, and fitting a linear-Gaussian to $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ with the same method we used for the dynamics. We can then solve the primal optimization in the dual gradient descent procedure with a standard LQR backward pass. This is significantly simpler and much faster than the forward-backward dynamic programming procedure employed in previous work (Levine and Abbeel, 2014; Levine and Koltun, 2014). This improvement is enabled by the use of BADMM, which allows us to always formulate the KL-divergence term in the Lagrangian with the distribution being optimized as the first argument. Since the KL-divergence is convex in its first argument, this makes the corresponding optimization significantly easier. The details of this LQR-based dual gradient descent algorithm are derived in Appendix A.4.

We can further improve the efficiency of the method by allowing samples from multiple trajectories $p_i(\tau)$ to be used to fit a shared dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$, while the controllers $p_i(\mathbf{u}_t|\mathbf{x}_t)$ are allowed to vary. This makes sense when the initial states of these trajectories

are similar, and they therefore visit similar regions. This allows us to draw just a single sample from each $p_i(\tau)$ at each iteration, allowing us to handle many more initial states.

4.3 Supervised Policy Optimization

Since the policy parameters θ participate only in the constraints of the optimization problem in Equation (1), optimizing the policy corresponds to minimizing the KL-divergence between the policy and trajectory distribution, as well as the expectation of $\lambda_{\mu t}^{\mathrm{T}}\mathbf{u}_t$. For a conditional Gaussian policy of the form $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t) = \mathcal{N}(\mu^\pi(\mathbf{o}_t), \Sigma^\pi(\mathbf{o}_t))$, the objective is

$$\begin{aligned}\mathcal{L}_\theta(\theta, p) = &\frac{1}{2N}\sum_{i=1}^{N}\sum_{t=1}^{T} E_{p_i(\mathbf{x}_t,\mathbf{o}_t)}\big[\mathrm{tr}[\mathbf{C}_{ti}^{-1}\Sigma^\pi(\mathbf{o}_t)] - \log|\Sigma^\pi(\mathbf{o}_t)| \\ &+(\mu^\pi(\mathbf{o}_t) - \mu_{ti}^p(\mathbf{x}_t))\mathbf{C}_{ti}^{-1}(\mu^\pi(\mathbf{o}_t) - \mu_{ti}^p(\mathbf{x}_t)) + 2\lambda_{\mu t}^{\mathrm{T}}\mu^\pi(\mathbf{o}_t)\big],\end{aligned}$$

where $\mu_{ti}^p(\mathbf{x}_t)$ is the mean of $p_i(\mathbf{u}_t|\mathbf{x}_t)$ and $\mathbf{C}_{ti}$ is the covariance, and the expectation is evaluated using samples from each $p_i(\tau)$ with corresponding observations $\mathbf{o}_t$. The observations are sampled from $p(\mathbf{o}_t|\mathbf{x}_t)$ by recording camera images on the real system. Since the input to $\mu^\pi(\mathbf{o}_t)$ and $\Sigma^\pi(\mathbf{o}_t)$ is not the state $\mathbf{x}_t$, but only an observation $\mathbf{o}_t$, we can train the policy to directly use raw observations. Note that $\mathcal{L}_\theta(\theta, p)$ is simply a weighted quadratic loss on the difference between the policy mean and the mean action of the trajectory distribution, offset by the Lagrange multiplier. The weighting is the precision matrix of the conditional in the trajectory distribution, which is equal to the curvature of its cost-to-go function (Levine and Koltun, 2013a). This has an intuitive interpretation: $\mathcal{L}_\theta(\theta, p)$ penalizes deviation from the trajectory distribution, with a penalty that is locally proportional to its cost-to-go. At convergence, when the policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ takes the same actions as $p_i(\mathbf{u}_t|\mathbf{x}_t)$, their Q-functions are equal, and the supervised policy objective becomes equivalent to the policy iteration objective (Levine and Koltun, 2014)

In this work, we optimize $\mathcal{L}_\theta(\theta, p)$ with respect to θ using stochastic gradient descent (SGD), a standard method for neural network training. The covariance of the Gaussian policy does not depend on the observation in our implementation, though adding this dependence would be straightforward. Since training complex neural networks requires a substantial number of samples, we found it beneficial to include sampled observations from previous iterations into the policy optimization, evaluating the action $\mu_{ti}^p(\mathbf{x}_t)$ at their corresponding states using the current trajectory distributions. Since these samples come from the wrong state distribution, we use importance sampling and weight them according to the ratio of their probability under the current distribution $p(\mathbf{x}_t)$ and the one they were sampled from, which is straightforward to evaluate under the estimated linear-Gaussian dynamics (Levine and Koltun, 2013b).

4.4 Comparison with Prior Guided Policy Search Methods

We presented a guided policy search method where the policy is trained on observations, while the trajectories are trained on the full state. The BADMM formulation of guided policy search is new to this work, though several prior guided policy search methods based on constrained optimization have been proposed. Levine and Koltun (2014) proposed a formulation similar to Equation (1), but with a constraint on the KL-divergence between

$p(\tau)$ and π_θ. This results in a more complex, non-convex forward-backward trajectory optimization phase. Since the BADMM formulation solves a convex problem during the trajectory optimization phase, it is substantially faster and easier to implement and use, especially when the number of trajectories $p_i(\tau)$ is large.

The use of ADMM for guided policy search was also proposed by Mordatch and Todorov (2014) for deterministic policies under known dynamics. This approach requires known, deterministic dynamics and trains deterministic policies. Furthermore, because this approach uses a simple quadratic augmented Lagrangian term, it further requires penalty terms on the gradient of the policy to account for local feedback. Our approach enforces this feedback behavior due to the higher moments included in the KL-divergence term, but does not require computing the second derivative of the policy.

5. End-to-End Visuomotor Policies

Guided policy search allows us to optimize complex, high-dimensional policies with raw observations, such as when the input to the policy consists of images from a robot's onboard camera. However, leveraging this capability to directly learn policies for visuomotor control requires designing a policy representation that is both data-efficient and capable of learning complex control strategies directly from raw visual inputs. In this section, we describe a deep convolutional neural network (CNN) model that is uniquely suited to this task. Our approach combines a novel spatial soft-argmax layer with a pretraining procedure that provides for flexibility and data-efficiency.

5.1 Visuomotor Policy Architecture

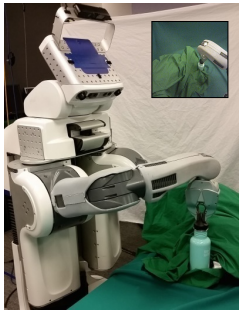
Our visuomotor policy runs at 20 Hz on the robot, mapping monocular RGB images and the robot configurations to joint torques on a 7 DoF arm. The configuration includes the angles of the joints and the pose of the end-effector (defined by 3 points in the space of the end-effector), as well as their velocities, but does not include the position of the target object or goal, which must be determined from the image. CNNs often use pooling to discard the locational information that is necessary to determine positions, since it is an irrelevant distractor for tasks such as object classification (Lee et al., 2009). Because locational information is important for control, our policy does not use pooling. Additionally, CNNs built for spatial tasks such as human pose estimation often also rely on the availability of location labels in image-space, such as hand-labeled keypoints (Tompson et al., 2014). We propose a novel CNN architecture capable of estimating spatial information from an image without direct supervision in image space. Our pose estimation experiments, discussed in Section 5.2, show that this network can learn useful visual features using only 3D position information provided by the robot, and no camera calibration. Further training the network with guided policy search to directly output motor torques causes it to acquire *task-specific* visual features. Our experiments in Section 6.4 show that this improves performance beyond the level achieved with features trained only for pose estimation.

Our network architecture is shown in Figure 2. The visual processing layers of the network consist of three convolutional layers, each of which learns a bank of filters that are applied to patches centered on every pixel of its input. These filters form a hierarchy of local image features. Each convolutional layer is followed by a rectifying nonlinearity of

the form $a_{cij} = \max(0, z_{cij})$ for each channel c and each pixel coordinate (i, j). The third convolutional layer contains 32 response maps with resolution 109×109. These response maps are passed through a spatial softmax function of the form $s_{cij} = e^{a_{cij}} / \sum_{i'j'} e^{a_{ci'j'}}$. Each output channel of the softmax is a probability distribution over the location of a feature in the image. To convert from this distribution to a coordinate representation (f_{cx}, f_{cy}), the network calculates the expected image position of each feature, yielding a 2D coordinate for each channel: $f_{cx} = \sum_{ij} s_{cij} x_{ij}$ and $f_{cy} = \sum_{ij} s_{cij} y_{ij}$, where (x_{ij}, y_{ij}) is the image-space position of the point (i, j) in the response map. Since this is a linear operation, it corresponds to a fixed, sparse fully connected layer with weights $W_{cix} = x_{ij}$ and $W_{cjy} = y_{ij}$. The combination of the spatial softmax and expectation operator implement a kind of soft-argmax. The spatial feature points (f_{cx}, f_{cy}) are concatenated with the robot's configuration and fed into two fully connected layers, each with 40 rectified units, followed by linear connections to the torques. The full network contains about 92,000 parameters, of which 86,000 are in the convolutional layers.

The spatial softmax and the expected position computation serve to convert pixel-wise representations in the convolutional layers to spatial coordinate representations, which can be manipulated by the fully connected layers into 3D positions or motor torques. The softmax also provides lateral inhibition, which suppresses low, erroneous activations, only keeping strong activations that are more likely to be accurate. This makes our policy more robust to distractors, providing generalization to novel visual variation. We compare our architecture with more standard alternatives in Section 6.3 and evaluate robustness to visual distractors in Section 6.4. However, the proposed architecture is also in some sense more specialized for visuomotor control, in contrast to more general standard convolutional networks. For example, not all perception tasks require information that can be coherently summarized by a set of spatial locations.

5.2 Visuomotor Policy Training

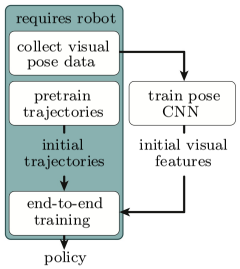


The guided policy search trajectory optimization phase uses the full state of the system, though the final policy only uses the observations. This type of instrumented training is a natural choice for many robotics tasks, where the robot is trained under controlled conditions, but must then act intelligently in uncontrolled, real-world situations. In our tasks, the unobserved variables are the pose of a target object (e.g. the bottle on which a cap must be placed). During training, this target object is typically held in the robot's left gripper, while the robot's right arm performs the task, as shown to the right. This allows the robot to move the target through a range of known positions. The final visuomotor policy does not receive this position as input, but must instead use the camera images. Due to the modest amount of training data, distractors that are correlated with task-relevant variables can hamper generalization. For this reason, the left arm is covered with cloth to prevent the policy from associating its appearance with the object's position.

While we can train the visuomotor policy entirely from scratch, the algorithm would spend a large number of iterations learning basic visual features and arm motions that can more efficiently be learned by themselves, before being incorporated into the policy. To speed up learning, we initialize both the vision layers in the policy and the trajectory distributions for guided policy search by leveraging the fully observed training setup. To initialize the vision layers, the robot moves the target object through a range of random positions, recording camera images and the object's pose, which is computed automatically from the pose of the gripper. This dataset is used to train a pose regression CNN, which consists of the same vision layers as the policy, followed by a fully connected layer that outputs the 3D points that define the target. Since the training set is still small (we use 1000 images collected from random arm motions), we initialize the filters in the first layer with weights from the model of Szegedy et al. (2014), which is trained on ImageNet (Deng et al., 2009) classification. After training on pose regression, the weights in the convolutional layers are transferred to the policy CNN. This enables the robot to learn the appearance of the objects prior to learning the behavior.

To initialize the linear-Gaussian controllers for each of the initial states, we take 15 iterations of guided policy search without optimizing the visuomotor policy. This allows for much faster training in the early iterations, when the trajectories are not yet successful, and optimizing the full visuomotor policy is unnecessarily time consuming. Since we still want the trajectories to arrive at compatible strategies for each target position, we replace the visuomotor policy during these iterations with a small network that receives the full state, which consisted of two layers with 40 rectified linear hidden units in our experiments. This network serves only to constrain the trajectories and avoid divergent behaviors from emerging for similar initial states, which would make subsequent policy learning difficult.

After initialization, we train the full visuomotor policy with guided policy search. During the supervised policy optimization phase, the fully connected motor control layers are first optimized by themselves, since they are not initialized with pretraining. This can be done very quickly because these layers are small. Then, the entire network is further optimized end-to-end. We found that first training the upper layers before end-to-end optimization prevented the convolutional layers from forgetting useful features learning during pretraining, when the error signal due to the untrained upper layers is very large. The entire pretraining scheme is summarized in the diagram on the right. Note that the trajectories can be pretrained in parallel with the vision layer pretraining, which does not require access to the physical system. Furthermore, the entire initialization procedure does not use any additional information that is not already available from the robot.



6. Experimental Evaluation

In this section, we present a series of experiments aimed at evaluating our approach and answering the following questions:

1. How does the guided policy search algorithm compare to other policy search methods for training complex, high-dimensional policies, such as neural networks?
2. Does our trajectory optimization algorithm work on a real robotic platform with unknown dynamics, for a range of different tasks?
3. How does our spatial softmax architecture compare to other, more standard convolutional neural network architectures?
4. Does training the perception and control systems in a visuomotor policy jointly end-to-end provide better performance than training each component separately?

Evaluating a wide range of policy search algorithms on a real robot would be extremely time consuming, particularly for methods that require a large number of samples. We therefore answer question (1) by using a physical simulator and simpler policies that do not use vision. This also allows us to test the generality of guided policy search on tasks that include manipulation, walking, and swimming. To answer question (2), we present a wide range of experiments on a PR2 robot. These experiments allow us to evaluate the sample efficiency of our trajectory optimization algorithm. To address question (3), we compare a range of different policy architectures on the task of localizing a target object (the cube in the shape sorting cube task). Since localizing the target object is a prerequisite for completing the shape sorting cube task, this serves as a good proxy for evaluating different architectures. Finally, we answer the last and most important question (4) by training visuomotor policies for hanging a coat hanger on a clothes rack, inserting a block into a shape sorting cube, fitting the claw of a toy hammer under a nail with various grasps, and screwing on a bottle cap. These tasks are illustrated in Figure 8.

6.1 Simulated Comparisons to Prior Policy Search Methods

In this section, we compare our method against prior policy search techniques on a range of simulated robotic control tasks. These results previously appeared in our conference paper that introduced the trajectory optimization procedure with local linear models (Levine and Abbeel, 2014). In these tasks, the state $\mathbf{x}_t$ consists of the joint angles and velocities of each robot, and the actions $\mathbf{u}_t$ consist of the torques at each joint. The neural network policies used one hidden layer and soft rectifier nonlinearities of the form $a = \log(1+\exp(z))$. Since these policies use the state as input, they only have a few hundred parameters, far fewer than our visuomotor policies. However, even this number of parameters can pose a major challenge for prior policy search methods (Deisenroth et al., 2013).

Experimental tasks. We simulated 2D and 3D peg insertion, octopus arm control, and planar swimming and walking. The difficulty in the peg insertion tasks stems from the need to align the peg with the slot and the complex contacts between the peg and the walls, which result in discontinuous dynamics. Octopus arm control involves moving the tip of a flexible arm to a goal position (Engel et al., 2005). The challenge in this task stems from its high dimensionality: the arm has 25 degrees of freedom, corresponding to 50 state dimensions. The swimming task requires controlling a three-link snake, and the walking task requires a seven-link biped to maintain a target velocity. The challenge in these tasks comes from underactuation. Details of the simulation and cost for each task are in Appendix B.1.

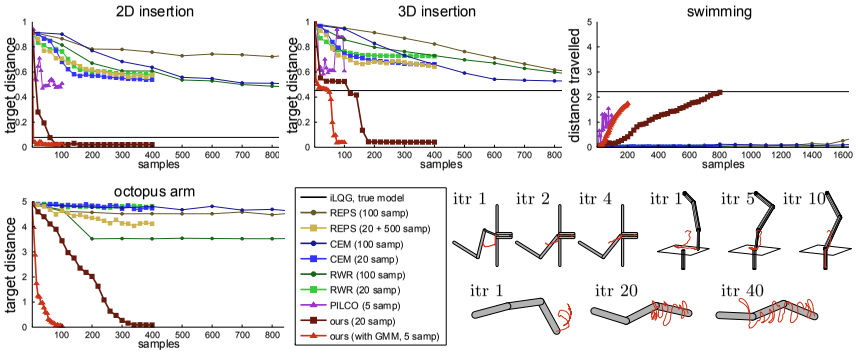


Figure 4: Results for learning linear-Gaussian controllers for 2D and 3D insertion, octopus arm, and swimming. Our approach uses fewer samples and finds better solutions than prior methods, and the GMM further reduces the required sample count. Images in the lower-right show the last time step for each system at several iterations of our method, with red lines indicating end effector trajectories.

Prior methods. We compare to REPS (Peters et al., 2010), reward-weighted regression (RWR) (Peters and Schaal, 2007; Kober and Peters, 2009), the cross-entropy method (CEM) (Rubinstein and Kroese, 2004), and PILCO (Deisenroth and Rasmussen, 2011). We also use iLQG (Li and Todorov, 2004) with a known model as a baseline, shown as a black horizontal line in all plots. REPS is a model-free method that, like our approach, enforces a KL-divergence constraint between the new and old policy. We compare to a variant of REPS that also fits linear dynamics to generate 500 pseudo-samples (Lioutikov et al., 2014), which we label "REPS (20 + 500)." RWR is an EM algorithm that fits the policy to previous samples weighted by the exponential of their reward, and CEM fits the policy to the best samples in each batch. With Gaussian trajectories, CEM and RWR only differ in the weights. These methods represent a class of RL algorithms that fit the policy to weighted samples, including PoWER and PI2 (Kober and Peters, 2009; Theodorou et al., 2010; Stulp and Sigaud, 2012). PILCO is a model-based method that uses a Gaussian process to learn a global dynamics model that is used to optimize the policy. We used the open-source implementation of PILCO provided by the authors. Both REPS and PILCO require solving large nonlinear optimizations at each iteration, while our method does not. Our method used 5 rollouts with the Gaussian mixture model prior, and 20 without. Due to its computational cost, PILCO was provided with 5 rollouts per iteration, while other prior methods used 20 and 100. For all prior methods with free hyperparameters (such as the fraction of elites for CEM), we performed hyperparameter sweeps and chose the most successful settings for the comparison.

Gaussian trajectory distributions. In the first set of comparisons, we evaluate only the trajectory optimization procedure for training linear-Gaussian controllers under unknown dynamics to determine its sample-efficiency and applicability to complex, high-dimensional problems. The results of this comparison for the peg insertion, octopus arm, and swimming

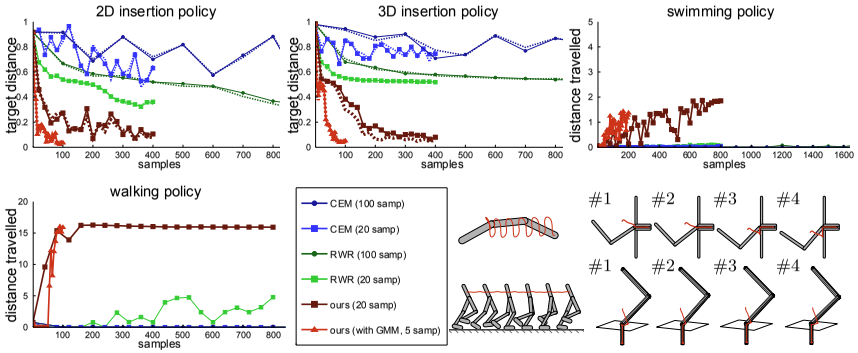


Figure 5: Comparison on neural network policies. For insertion, the policy was trained to search for an unknown slot position on four slot positions (shown above). Generalization to new positions is graphed with dashed lines. Note how the end effector (red) follows the surface to find the slot, and how the swimming gait is smoother due to the stationary policy.

tasks appears in Figure 4. The horizontal axis shows the total number of samples, and the vertical axis shows the minimum distance between the end of the peg and the bottom of the slot, the distance to the target for the octopus arm, or the total distance travelled by the swimmer. Since the peg is 0.5 units long, distances above this amount correspond to controllers that cannot perform an insertion. Our method learned much more effective controllers with fewer samples, especially when using the Gaussian mixture model prior. On 3D insertion, it outperformed the iLQG baseline, which used a known model. Contact discontinuities cause problems for derivative-based methods like iLQG, as well as methods like PILCO that learn a smooth global dynamics model. We use a time-varying local model, which preserves more detail, and fitting the model to samples has a smoothing effect that mitigates discontinuity issues. Prior policy search methods could servo to the hole, but were unable to insert the peg. On the octopus arm, our method succeeded despite the high dimensionality of the state and action spaces.[1] Our method also successfully learned a swimming gait, while prior model-free methods could not initiate forward motion. PILCO also learned an effective gait due to the smooth dynamics of this task, but its GP-based optimization required orders of magnitude more computation time than our method, taking about 50 minutes per iteration. In the case of prior model-free methods, the high dimensionality of the time-varying linear-Gaussian controllers likely caused considerable difficulty (Deisenroth et al., 2013), while our approach exploits the structure of linear-Gaussian controllers for efficient learning.

1. The high dimensionality of the octopus arm made it difficult to run PILCO, though in principle, such methods should perform well on this task given the arm's smooth dynamics.

Neural network policies. In the second set of comparisons, shown in Figure 5, we compare guided policy search to RWR and CEM[2] on the challenging task of training high-dimensional neural network policies for the peg insertion and locomotion tasks. The variant of guided policy search used in this comparison differs somewhat from the version described in Section 4, in that it used a simpler dual gradient descent formulation, rather than BADMM. In practice, we found the performance of these methods to be very similar, though the BADMM variant was substantially faster and easier to implement.

On swimming, our method achieved similar performance to the linear-Gaussian case, but since the neural network policy was stationary, the resulting gait was much smoother. Previous methods could only solve this task with 100 samples per iteration, with RWR eventually obtaining a distance of 0.5m after 4000 samples, and CEM reaching 2.1m after 3000. Our method was able to reach such distances with many fewer samples. Following prior work (Levine and Koltun, 2013a), the walker trajectory was initialized from a demonstration, which was stabilized with simple linear feedback. The RWR and CEM policies were initialized with samples from this controller to provide a fair comparison. The graph shows the average distance travelled on rollouts that did not fall, and shows that only our method was able to learn walking policies that succeeded consistently.

On peg insertion, the neural network was trained to insert the peg without precise knowledge of the position of the hole, resulting in a partially observed problem. The holes were placed in a region of radius 0.2 units in 2D and 0.1 units in 3D. The policies were trained on four different hole positions, and then tested on four new hole positions to evaluate generalization. The hole position was not provided to the neural network, and the policies therefore had to search for the hole, with only joint angles and velocities as input. Only our method could acquire a successful strategy to locate both the training and test holes, although RWR was eventually able to insert the peg into one of the four holes in 2D.

These comparisons show that training even medium-sized neural network policies for continuous control tasks with a limited number of samples is very difficult for many prior policy search algorithms. Indeed, it is generally known that model-free policy search methods struggle with policies that have over 100 parameters (Deisenroth et al., 2013). In subsequent sections, we will evaluate our method on real robotic tasks, showing that it can scale from these simulated tasks all the way up to end-to-end learning of visuomotor control.

6.2 Learning Linear-Gaussian Controllers on a PR2 Robot

In this section, we demonstrate the range of manipulation tasks that can be learned using our trajectory optimization algorithm on a real PR2 robot. These experiments previously appeared in our conference paper on guided policy search (Levine et al., 2015). Since performing trajectory optimization is a prerequisite for guided policy search to learn effective visuomotor policies, it is important to evaluate that our trajectory optimization can learn a wide variety of robotic manipulation tasks under unknown dynamics. The tasks in these experiments are shown in Figure 6, while Figure 7 shows the learning curves for each task. For all robotic experiments in this article, the tasks were learned entirely from scratch,

2. PILCO cannot optimize neural network policies, and we could not obtain reasonable results with REPS. Prior applications of REPS generally focus on simpler, lower-dimensional policy classes (Peters et al., 2010; Lioutikov et al., 2014).

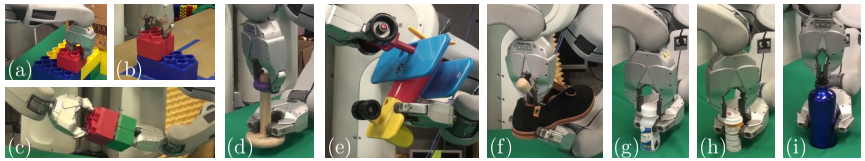


Figure 6: Tasks for linear-Gaussian controller evaluation: (a) stacking lego blocks on a fixed base, (b) onto a free-standing block, (c) held in both gripper; (d) threading wooden rings onto a peg; (e) attaching the wheels to a toy airplane; (f) inserting a shoe tree into a shoe; (g,h) screwing caps onto pill bottles and (i) onto a water bottle.

with the initialization of the controllers $p(\mathbf{u}_t|\mathbf{x}_t)$ described in Appendix B.2. The number of samples required to learn each controller is around 20-25, substantially lower than many prior policy search methods in the literature (Peters and Schaal, 2008; Kober et al., 2010b; Theodorou et al., 2010; Deisenroth et al., 2013). Total learning time was about ten minutes for each task, of which only 3-4 minutes involved system interaction. The rest of the time was spent resetting the robot to the initial state and on computation.

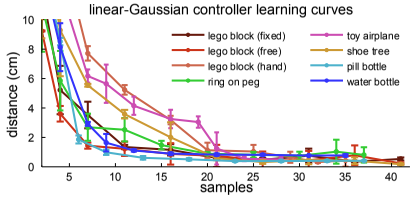


Figure 7: Distance to target point during training of linear-Gaussian controllers. The actual target may differ due to perturbations. Error bars indicate one standard deviation.

The linear-Gaussian controllers are optimized for a specific condition – e.g., a specific position of the target lego block. To evaluate their robustness to errors in the specified target position, we conducted experiments on the lego block and ring tasks where the target object (the lower block and the peg) was perturbed at each trial during training, and then tested with various perturbations. For each task, controllers were trained with Gaussian perturbations with standard deviations of 0, 1, and 2 cm in the position of the target object, and each controller was tested with perturbations of radius 0, 1, 2, and 3 cm. Note that with a radius of 2 cm, the peg would be placed about one ring-width away from the expected position. The results are shown in Table 2. All controllers were robust to perturbations of 1 cm, and would often succeed at 2 cm. Robustness increased slightly when more noise was injected during training, but even controllers trained without noise exhibited considerable robustness, since the linear-Gaussian controllers themselves add noise during sampling. We also evaluated a kinematic baseline for each perturbation level, which planned a straight path from a point 5 cm above the target to the expected (unperturbed) target location. This baseline was only able to place the lego block in the absence of perturbations. The rounded top of the peg provided an easier condition for the baseline, with occasional successes at higher perturbation levels. Our controllers outperformed the baseline by a wide margin.

All of the robotic experiments discussed in this section may be viewed in the corresponding supplementary video, available online: `http://rll.berkeley.edu/icra2015gps`. A video illustration of the visuomotor policies, discussed in the following sections, is also available: `http://sites.google.com/site/visuomotorpolicy`.

		test perturbation							
		lego block				ring on peg			
		0 cm	1 cm	2 cm	3 cm	0 cm	1 cm	2 cm	3 cm
training perturb.	0 cm	5/5	5/5	3/5	2/5	5/5	5/5	0/5	0/5
	1 cm	5/5	5/5	3/5	2/5	5/5	5/5	3/5	0/5
	2 cm	5/5	5/5	5/5	3/5	5/5	5/5	3/5	0/5
	kinematic baseline	5/5	0/5	0/5	0/5	5/5	3/5	0/5	0/5

Table 2: Success rates of linear-Gaussian controllers under target object perturbation.

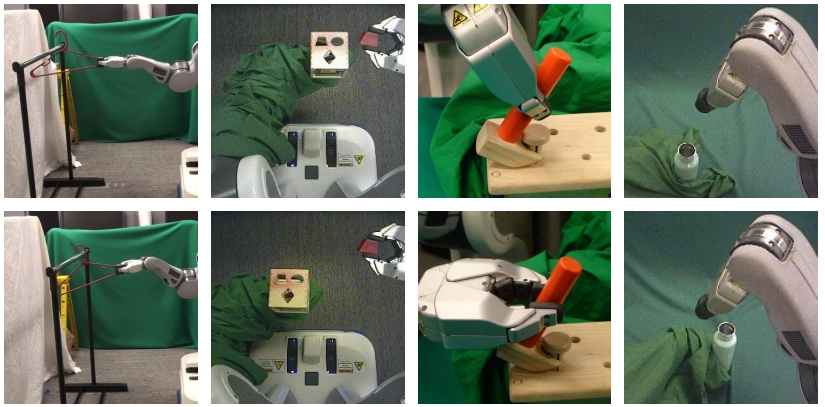
6.3 Spatial Softmax CNN Architecture Evaluation

In this section, we evaluate the neural network architecture that we propose in Section 5.1 in comparison to more standard convolutional networks. To isolate the architectures from other confounding factors, we measure their accuracy on the pose estimation pretraining task described in Section 5.2. This is a reasonable proxy for evaluating how well the network can overcome two major challenges in visuomotor learning: the ability to handle relatively small datasets without overfitting, and the capability to learn tasks that are inherently spatial. We compare to a network where the expectation operator after the softmax is replaced with a learned fully connected layer, as is standard in the literature, a network where both the softmax and the expectation operators are replaced with a fully connected layer, and a version of this network that also uses 3×3 max pooling with stride 2 at the first two layers. These alternative architectures have many more parameters, since the fully connected layer takes the entire bank of response maps from the third convolutional layer as input. Pooling helps to reduce the number of parameters, but not to the same degree as the spatial softmax and expectation operators in our architecture.

The results in Table 3 indicate that using the softmax and expectation operators improves pose estimation accuracy substantially. Our network is able to outperform the more standard architectures because it is forced by the softmax and expectation operators to learn feature points, which provide a concise representation suitable for spatial inference. Since most of the parameters in this architecture are in the convolutional layers, which benefit from extensive weight sharing, overfitting is also greatly reduced. By removing pooling, our network also maintains higher resolution in the convolutional layers, improving spatial accuracy. Although we did attempt to regularize the larger standard architectures with higher weight decay and dropout, we did not observe a significant improvement on this dataset. We also did not extensively optimize the parameters of this network, such as filter size and number of channels, and investigating these design decisions further would be valuable to investigate in future work.

network architecture	test error (cm)
softmax + feature points (**ours**)	**1.30 ± 0.73**
softmax + fully connected layer	2.59 ± 1.19
fully connected layer	4.75 ± 2.29
max-pooling + fully connected	3.71 ± 1.73

Table 3: Average pose estimation accuracy and standard deviation with various architectures, measured as average Euclidean error for the three target points in 3D, with ground truth determined by forward kinematics from the left arm.



(a) hanger (b) cube (c) hammer (d) bottle

Figure 8: Illustration of the tasks in our visuomotor policy experiments, showing the variation in the position of the target for the hanger, cube, and bottle tasks, as well as two of the three grasps for the hammer, which also included variation in position (not shown).

6.4 Deep Visuomotor Policy Evaluation

In this section, we present an evaluation of our full visuomotor policy training algorithm on a PR2 robot. The aim of this evaluation is to answer the following question: does training the perception and control systems in a visuomotor policy jointly end-to-end provide better performance than training each component separately?

Experimental tasks. We trained policies for hanging a coat hanger on a clothes rack, inserting a block into a shape sorting cube, fitting the claw of a toy hammer under a nail with various grasps, and screwing on a bottle cap. The cost function for these tasks encourages low distance between three points on the end-effector and corresponding target points, low torques, and, for the bottle task, spinning the wrist. The equations for these cost functions and the details of each task are presented in Appendix B.2. The tasks are illustrated in Figure 8. Each task involved variation of 10-20 cm in each direction in the position of the target object (the rack, shape sorting cube, nail, and bottle). In addition, the coat hanger and hammer tasks were trained with two and three grasps, respectively. The current angle of the grasp was not provided to the policy, but had to be inferred from observing the robot's gripper in the camera images. All tasks used the same policy architecture and model parameters.

Experimental conditions. We evaluated the visuomotor policies in three conditions: (1) the training target positions and grasps, (2) new target positions not seen during training and, for the hammer, new grasps (spatial test), and (3) training positions with visual distractors (visual test). A selection of these experiments is shown in the supplementary video.[3] For the visual test, the shape sorting cube was placed on a table rather than held in

3. The video can be viewed at `http://sites.google.com/site/visuomotorpolicy`

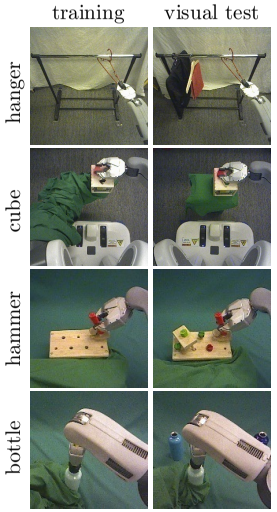
the gripper, the coat hanger was placed on a rack with clothes, and the bottle and hammer tasks were done in the presence of clutter. Illustrations of this test are shown in Figure 9.

Comparison. The success rates for each test are shown in Figure 9. We compared to two baselines, both of which train the vision layers in advance for pose prediction, instead of training the entire policy end-to-end. The features baseline discards the last layer of the pose predictor and uses the feature points, resulting in the same architecture as our policy, while the prediction baseline feeds the predicted pose into the control layers. The pose prediction baseline is analogous to a standard modular approach to policy learning, where the vision system is first trained to localize the target, and the policy is trained on top of it. This variant achieves poor performance. As discussed in Section 6.3, the pose estimate is accurate to about 1 cm. However, unlike the tasks in Section 6.2, where robust controllers could succeed even with inaccurate perception, many of these tasks have tolerances of just a few millimeters. In fact, the pose prediction baseline is only successful on the coat hanger, which requires comparatively little accuracy. Millimeter accuracy is difficult to achieve even with calibrated cameras and checkerboards. Indeed, prior work has reported that the PR2 can maintain a camera to end effector accuracy of about 2 cm during open loop motion (Meeussen et al., 2010). This suggests that the failure of this baseline is not atypical, and that our visuomotor policies are learning visual features and control strategies that improve the robot's accuracy. When provided with pose estimation features, the policy has more freedom in how it uses the visual information, and achieves somewhat higher success rates. However, full end-to-end training performs significantly better, achieving high accuracy even on the challenging bottle task, and successfully adapting to the variety of grasps on the hammer task. This suggests that, although the vision layer pretraining is clearly beneficial for reducing computation time, it is not sufficient by itself for discovering good features for visuomotor policies.

Visual distractors. The policies exhibit moderate tolerance to distractors that are visually separated from the target object. This is enabled in part by the spatial softmax, which has a lateral inhibition effect that suppresses non-maximal activations. Since distractors are unlikely to activate each feature as much as the true object, their activations are therefore suppressed. However, as expected, the learned policies tend to perform poorly under drastic changes to the backdrop, or when the distractors are adjacent to or occluding the manipulated objects, as shown in the supplementary video. A standard solution to this issue to expose the policy to a greater variety of visual situations during training. This issue could also be mitigated by artificially augmenting the image samples with synthetic transformations, as discussed in prior work in computer vision (Simard et al., 2003), or even incorporating ideas from transfer and semi-supervised learning.

6.5 Features Learned with End-to-End Training

The visual processing layers of our architecture automatically learn features points using the spatial softmax and expectation operators. These feature points encapsulate all of the visual information received by the motor layers of the policy. In Figure 10, we show the features points discovered by our visuomotor policy through guided policy search. Each policy learns features on the target object and the robot manipulator, both clearly relevant



coat hanger	training (18)	spatial test (24)	visual test (18)
end-to-end	**100%**	**100%**	**100%**
pose features	88.9%	87.5%	83.3%
pose prediction	55.6%	58.3%	66.7%
shape cube	training (27)	spatial test (36)	visual test (40)
end-to-end	**96.3%**	**91.7%**	**87.5%**
pose features	70.4%	83.3%	40%
pose prediction	0%	0%	n/a
toy hammer	training (45)	spatial test (60)	visual test (60)
end-to-end	**91.1%**	**86.7%**	**78.3%**
pose features	62.2%	75.0%	53.3%
pose prediction	8.9%	18.3%	n/a
bottle cap	training (27)	spatial test (12)	visual test (40)
end-to-end	**88.9%**	**83.3%**	**62.5%**
pose features	55.6%	58.3%	27.5%

Success rates on training positions, on novel test positions, and in the presence of visual distractors. The number of trials per test is shown in parentheses.

Figure 9: Training and visual test scenes as seen by the policy (left), and experimental results (right). The hammer and bottle images were cropped for visualization only.

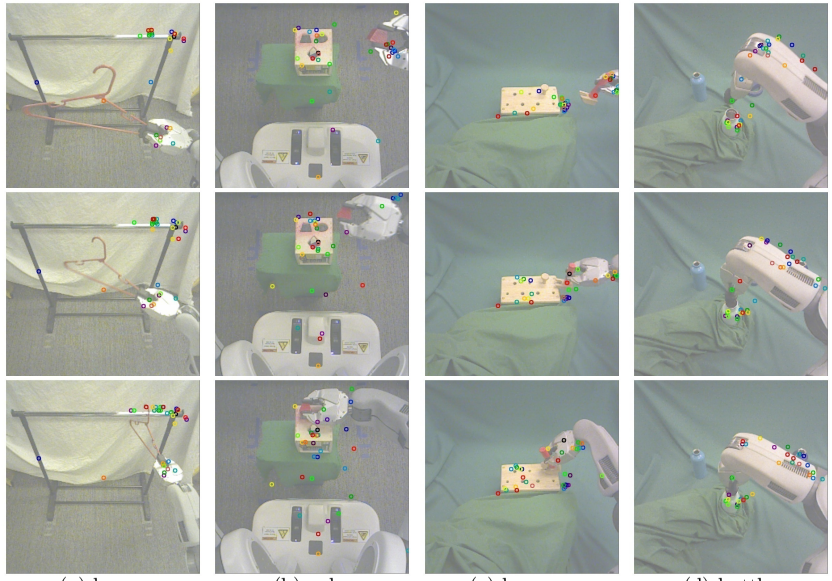
to task execution. The policy tends to pick out robust, distinctive features on the objects, such as the left pole of the clothes rack, the left corners of the shape-sorting cube and the bottom-left corner of the toy tool bench. In the bottle task, the end-to-end trained policy outputs points on both sides of the bottle, including one on the cap, while the pose prediction network only finds points on the right edge of the bottle.

In Figure 11, we compare the feature points learned through guided policy search to those learned by a CNN trained for pose prediction. After end-to-end training, the policy acquired a distinctly different set of feature points compared to the pose prediction CNN used for initialization. The end-to-end trained model finds more feature points on task-relevant objects and fewer points on background objects. This suggests that the policy improves its performance by acquiring *goal-driven* visual features that differ from those learned for object localization.

The feature point representation is very simple, since it assumes that the learned features are present at all times, and only one instance of each feature is ever present in the image. While this is a drastic simplification, both the pose predictor and the policy still achieve good results. A more flexible architecture that still learns a concise feature point representation could further improve policy performance. We hope to explore this in future work.

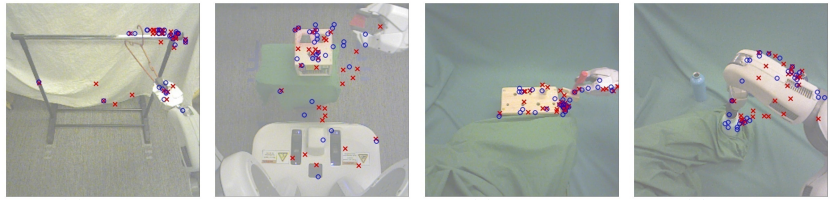
6.6 Computational Performance and Sample Efficiency

We used the Caffe deep learning library (Jia et al., 2014) for CNN training. Each visuomotor policy required a total of 3-4 hours of training time: 20-30 minutes for the pose prediction data collection on the robot, 40-60 minutes for the fully observed trajectory pretraining on



(a) hanger (b) cube (c) hammer (d) bottle

Figure 10: Feature points tracked by the policy during task execution for each of the four tasks. Each feature point is displayed in a different random color, with consistent coloring across images. The policy finds features on the target object and the robot gripper and arm. In the bottle cap task, note that the policy correctly ignores the distractor bottle in the background, even though it was not present during training.



(a) hanger (b) cube (c) hammer (d) bottle

Figure 11: Feature points learned for each task. For each input image, the feature points produced by the policy are shown in blue, while the feature points of the pose prediction network are shown in red. The end-to-end trained policy tends to discover more feature points on the target object and the robot arm than the pose prediction network.

the robot and offline pose pretraining (which can be done in parallel), and between 1.5 and 2.5 hours for end-to-end training with guided policy search. The coat hanger task required two iterations of guided policy search, the shape sorting cube and the hammer required three, and the bottle task required four. Only about 15 minutes of the training time consisted of executing trials on the robot. Since training was dominated by computation, we expect significant speedup from a more efficient implementation. The number of samples for training each policy is shown in Table 4. Each trial was five seconds in length, and the numbers do not include the time needed to collect about 1000 images for pretraining the visual processing layers of the policy.

	number of trials		
task	trajectory pretraining	end-to-end training	total
coat hanger	120	36	156
shape cube	90	81	171
toy hammer	150	90	240
bottle cap	180	108	288

Table 4: Total number of trials used for learning each visuomotor policy.

7. Discussion and Future Work

In this paper, we presented a method for learning robotic control policies that use raw input from a monocular camera. These policies are represented by a novel convolutional neural network architecture, and can be trained end-to-end using our guided policy search algorithm, which decomposes the policy search problem in a trajectory optimization phase that uses full state information and a supervised learning phase that only uses the observations. This decomposition allows us to leverage state-of-the-art tools from supervised learning, making it straightforward to optimize extremely high-dimensional policies. Our experimental results show that our method can execute complex manipulation skills, and that end-to-end training produces significant improvements in policy performance compared to using fixed vision layers trained for pose prediction.

Although we demonstrate moderate generalization over variations in the scene, our current method does not generalize to dramatically different settings, especially when visual distractors occlude the manipulated object or break up its silhouette in ways that differ from the training. The success of CNNs on exceedingly challenging vision tasks suggests that this class of models is capable of learning invariance to irrelevant distractor features (LeCun et al., 2015), and in principle this issue can be addressed by training the policy in a variety of environments, though this poses certain logistical challenges. More practical alternatives that could be explored in future work include simultaneously training the policy on multiple robots, each of which is located in a different environment, developing more sophisticated regularization and pretraining techniques to avoid overfitting, and introducing artificial data augmentation to encourage the policy to be invariant to irrelevant clutter. However, even without these improvements, our method has numerous applications in, for example, an industrial setting where the robot must repeatedly and efficiently perform a task that requires visual feedback under moderate variation in background and clutter conditions.

Our method takes advantage of a known, fully observed state space during training. This is both a weakness and a strength. It allows us to train linear-Gaussian controllers

for guided policy search using a very small number of samples, far more efficiently than standard policy search methods. However, the requirement to observe the full state during training limits the tasks to which the method can be applied. In many cases, this limitation is minor, and the only "instrumentation" required at training is to position the objects in the scene at consistent positions. However, tasks that require, for example, manipulating freely moving objects require more extensive instrumentation, such as motion capture. A promising direction for addressing this limitation is to combine our method with unsupervised state-space learning, as proposed in several recent works, including our own (Lange et al., 2012; Watter et al., 2015; Finn et al., 2015).

In future work, we hope to explore more complex policy architectures, such as recurrent policies that can deal with extensive occlusions by keeping a memory of past observations. We also hope to extend our method to a wider range of tasks that can benefit from visual input, as well as a variety of other rich sensory modalities, including haptic input from pressure sensors and auditory input. With a wider range of sensory modalities, end-to-end training of sensorimotor policies will become increasingly important: while it is often straightforward to imagine how vision might help to localize the position of an object in the scene, it is much less apparent how sound can be integrated into robotic control. A learned sensorimotor policy would be able to naturally integrate a wide range of modalities and utilize them to directly aid in control.

Acknowledgements

This research was funded in part by DARPA through a Young Faculty Award, the Army Research Office through the MAST program, NSF awards IIS-1427425 and IIS-1212798, the Berkeley Vision and Learning Center, and a Berkeley EECS Department Fellowship.

Appendix A. Guided Policy Search Algorithm Details

In this appendix, we describe a number of implementation details of our BADMM-based guided policy search algorithm and our linear-Gaussian controller optimization method.

A.1 BADMM Dual Variables and Weight Adjustment

Recall that the inner loop alternating optimization is given by

$$\theta \leftarrow \arg\min_\theta \sum_{t=1}^{T} E_{p(\mathbf{x}_t)\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)}[\mathbf{u}_t^\mathrm{T}\lambda_{\mu t}] + \nu_t \phi_t^\theta(\theta, p)$$

$$p \leftarrow \arg\min_p \sum_{t=1}^{T} E_{p(\mathbf{x}_t,\mathbf{u}_t)}[\ell(\mathbf{x}_t,\mathbf{u}_t) - \mathbf{u}_t^\mathrm{T}\lambda_{\mu t}] + \nu_t \phi_t^p(p,\theta)$$

$$\lambda_{\mu t} \leftarrow \lambda_{\mu t} + \alpha\nu_t(E_{\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)}[\mathbf{u}_t] - E_{p(\mathbf{u}_t|\mathbf{x}_t)p(\mathbf{x}_t)}[\mathbf{u}_t]).$$

We use a step size of $\alpha = 0.1$ in all of our experiments, which we found to be more stable than $\alpha = 1.0$. The weights ν_t are initialized to 0.01 and incremented based on the following schedule: at every iteration, we compute the average KL-divergence between $p(\mathbf{u}_t|\mathbf{x}_t)$ and $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ at each time step, as well as its standard deviation over time steps.

The weights ν_t corresponding to time steps where the KL-divergence is higher than the average are increased by a factor of 2, and the weights corresponding to time steps where the KL-divergence is two standard deviations or more below the average are decreased by a factor of 2. The rationale behind this schedule is to adjust the KL-divergence penalty to keep the policy and trajectory in agreement by roughly the same amount at all time steps. Increasing ν_t too quickly can lead to the policy and trajectory becoming "locked" together, which makes it difficult for the trajectory to decrease its cost, while leaving it too low requires more iterations for convergence. We found this schedule to work well across all tasks, both during trajectory pretraining and while training the visuomotor policy.

To update the dual variables $\lambda_{\mu t}$, we evaluate the expectations over $p(\mathbf{x}_t)$ by using the latest batch of sampled trajectories. For each state $\{\mathbf{x}_t^i\}$ along these sampled trajectories, we evaluate the expectations over $\mathbf{u}_t$ under $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ and $p(\mathbf{u}_t|\mathbf{x}_t)$, which correspond simply to the means of these conditional Gaussian distributions, in closed form.

A.2 Policy Variance Optimization

As discussed in Section 4, the variance of the Gaussian policy $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ does not depend on the observation, though this dependence would be straightforward to add. Analyzing the objective $\mathcal{L}_\theta(\theta, p)$, we can write out only the terms that depend on Σ^π:

$$\mathcal{L}_\theta(\theta, p) = \frac{1}{2N}\sum_{i=1}^{N}\sum_{t=1}^{T} E_{p_i(\mathbf{x}_t,\mathbf{o}_t)}\left[\mathrm{tr}[\mathbf{C}_{ti}^{-1}\Sigma^\pi] - \log|\Sigma^\pi|\right].$$

Differentiating and setting the derivative to zero, we obtain the following equation for Σ^π:

$$\Sigma^\pi = \left[\frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathbf{C}_{ti}^{-1}\right]^{-1},$$

where the expectation under $p_i(\mathbf{x}_t)$ is omitted, since $\mathbf{C}_{ti}$ does not depend on $\mathbf{x}_t$.

A.3 Dynamics Fitting

Optimizing the linear-Gaussian controllers $p_i(\mathbf{u}_t|\mathbf{x}_t)$ that induce the trajectory distributions $p_i(\tau)$ requires fitting the system dynamics $p_i(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ at each iteration to samples generated on the physical system from the previous controller $\hat{p}_i(\mathbf{u}_t|\mathbf{x}_t)$. In this section, we describe how these dynamics are fitted. As in Section 4, we drop the subscript i, since the dynamics are fitted the same way for all of the trajectory distributions.

The linear-Gaussian dynamics are defined as $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(f_{\mathbf{x}t}\mathbf{x}_t + f_{\mathbf{u}t}\mathbf{u}_t + f_{ct}, \mathbf{F}_t)$, and the data that we obtain from the robot can be viewed as tuples $\{\mathbf{x}_t^i, \mathbf{u}_t^i, \mathbf{x}_{t+1}^i\}$. A simple way to fit these linear-Gaussian dynamics is to use linear regression to determine $f_\mathbf{x}$, $f_\mathbf{u}$, and f_c, and fit $\mathbf{F}_t$ based on the errors. However, the sample complexity of linear regression scales with the dimensionality of $\mathbf{x}_t$. For a high-dimensional robotic system, we might need an impractically large number of samples at each iteration to obtain a good fit. However, we can observe that the dynamics at nearby time steps are strongly correlated, and we can dramatically reduce the sample complexity of the dynamics fitting by bringing in information from other time steps, and even prior iterations. We will bring in this

information by fitting a global model to all of the transitions $\{\mathbf{x}_t^i, \mathbf{u}_t^i, \mathbf{x}_{t+1}^i\}$ for all t and all tuples from several prior iterations (we use three prior iterations in our implementation), and then use this model as a prior for fitting the dynamics at each time step. Note that this global model does not itself need to be a good forward dynamics model – it just needs to serve as a good prior to reduce the sample complexity of linear regression.

To make it more convenient to incorporate a data-driven prior, we will first reformulate this linear regression fit and view it as fitting a Gaussian model to the dataset $\{\mathbf{x}_t^i, \mathbf{u}_t^i, \mathbf{x}_{t+1}^i\}$ at each time step t, and then conditioning this Gaussian to obtain $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$. While this is equivalent to linear regression, it allows us to easily incorporate a normal-inverse-Wishart prior on this Gaussian in order to bring in prior information. Let $\hat{\Sigma}$ be the empirical covariance of our dataset, and let $\hat{\mu}$ be the empirical mean. The normal-inverse-Wishart prior is defined by prior parameters $\mathbf{\Phi}$, μ_0, m, and n_0. Under this prior, the maximum a posteriori estimates for the covariance Σ and mean μ are given by

$$\Sigma = \frac{\mathbf{\Phi} + N\hat{\Sigma} + \frac{Nm}{N+m}(\hat{\mu} - \mu_0)(\hat{\mu} - \mu_0)^{\mathrm{T}}}{N + n_0} \qquad \mu = \frac{m\mu_0 + n_0\hat{\mu}}{m + n_0}.$$

Having obtained Σ and μ, we can obtain an estimate of the dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ by conditioning the distribution $\mathcal{N}(\mu, \Sigma)$ on $[\mathbf{x}_t; \mathbf{u}_t]$, which produces linear-Gaussian dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(f_{\mathbf{x}t}\mathbf{x}_t + f_{\mathbf{u}t}\mathbf{u}_t + f_{ct}, \mathbf{F}_t)$. The parameters of the normal-inverse-Wishart prior are obtained from the global model of the dynamics which, as described previously, is fitted to all available tuples $\{\mathbf{x}_t^i, \mathbf{u}_t^i, \mathbf{x}_{t+1}^i\}$.

The simplest prior can be obtained by fitting a Gaussian distribution to vectors $[\mathbf{x}; \mathbf{u}; \mathbf{x}']$. If the mean and covariance of this data are given by $\bar{\mu}$ and $\bar{\Sigma}$, the prior is given by $\mathbf{\Phi} = n_0\bar{\Sigma}$ and $\mu_0 = \bar{\mu}$, while n_0 and m should be set to the number of data points in the datasets. In practice, settings n_0 and m to 1 tends to produce better results, since the prior is fitted to many more samples than are available for linear regression at each time step. While this prior is simple, we can obtain a better prior by employing a nonlinear model.

The particular global model we use in this work is a Gaussian mixture model over vectors $[\mathbf{x}; \mathbf{u}; \mathbf{x}']$. Systems of articulated rigid bodies undergoing contact dynamics, such as robots interacting with their environment, can be coarsely modeled as having piecewise linear dynamics. The Gaussian mixture model provides a good approximation for such piecewise linear systems, with each mixture element corresponding to a different linear mode (Khansari-Zadeh and Billard, 2010). Under this model, the state transition tuple is assumed to come from a distribution that depends on some hidden state h, which corresponds to the mixture element identity. In practice, this hidden state might correspond to the type of contact profile experienced by a robotic arm at step i. The prior for the dynamics fit at time step t is then obtained by inferring the hidden state distribution for the transition dataset $\{\mathbf{x}_t^i, \mathbf{u}_t^i, \mathbf{x}_{t+1}^i\}$, and using the mean and covariance of the corresponding mixture elements (weighted by their probabilities) to obtain $\bar{\mu}$ and $\bar{\Sigma}$. The prior parameters can then be obtained as described above.

In our experiments, we set the number of mixture elements for the Gaussian mixture model prior such that there were at least 40 samples per mixture element, or 20 total mixture elements, whichever was lower. In general, we did not find the performance of the method to be sensitive to this parameter, though overfitting did tend to occur in the early iterations when the number of samples is low, if the number of mixtures was too high.

A.4 Trajectory Optimization

In this section, we show how the LQR backward pass can be used to optimize the constrained objective in Section 4.2. The constrained trajectory optimization problem is given by

$$\min_{p(\tau)\in\mathcal{N}(\tau)} \mathcal{L}_p(p, \theta) \text{ s.t. } D_{\mathrm{KL}}(p(\tau)\|\hat{p}(\tau)) \leq \epsilon.$$

The augmented Lagrangian $\mathcal{L}_p(p, \theta)$ consists of an entropy term and an expectation under $p(\tau)$ of a quantity that is independent of p. We can locally approximate this quantity with a quadratic by using a quadratic expansion of $\ell(\mathbf{x}_t, \mathbf{u}_t)$, and fitting a linear Gaussian to $\pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$ with the same method we used for the dynamics. We can then solve the primal optimization in the dual gradient descent procedure with a standard LQR backward pass. As discussed in Section 4, $\mathcal{L}_p(p, \theta)$ can be written as the expectation of some function $c(\tau)$ that is independent of p, such that $\mathcal{L}_p(p, \theta) = E_{p(\tau)}[c(\tau)] - \nu_t\mathcal{H}(p(\tau))$. Specifically,

$$c(\mathbf{x}_t, \mathbf{u}_t) = \ell(\mathbf{x}_t, \mathbf{u}_t) - \mathbf{u}_t^{\mathrm{T}}\lambda_{\mu t} - \nu_t \log \pi_\theta(\mathbf{u}_t|\mathbf{x}_t)$$

Writing the Lagrangian of the constrained optimization, we have

$$\mathcal{L}(p) = E_{p(\tau)}[c(\tau) - \eta \log \hat{p}(\tau)] - (\eta + \nu_t)\mathcal{H}(p(\tau)) - \eta\epsilon,$$

where η is the Lagrange multiplier. Note that $\mathcal{L}(p)$ is the Lagrangian of the constrained trajectory optimization, which is not related to the augmented Lagrangian $\mathcal{L}_p(\tau, \theta)$. Grouping the terms in the expectation and omitting constants, we can rewrite the minimization of the Lagrangian with respect to the primal variables as

$$\min_{p(\tau)\in\mathcal{N}(\tau)} E_{p(\tau)}\left[\frac{1}{\eta + \nu_t}c(\tau) - \frac{\eta}{\eta + \nu_t}\log \hat{p}(\tau)\right] - \mathcal{H}(p(\tau)). \tag{4}$$

Let $\tilde{c}(\tau) = \frac{1}{\eta+\nu_t}c(\tau) - \frac{\eta}{\eta+\nu_t}\log \hat{p}(\tau)$. The above optimization corresponds to minimizing $E_{p(\tau)}[\tilde{c}(\tau)] - \mathcal{H}(p(\tau))$. This type of maximum entropy problem can be solved using the LQR algorithm, and the solution is given by

$$p(\mathbf{u}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{K}_t\mathbf{x}_t + \mathbf{k}_t; Q_{\mathbf{u},\mathbf{u}t}^{-1}),$$

where $\mathbf{K}_t$ and $\mathbf{k}_t$ are the feedback and open loop terms of the optimal linear feedback controller corresponding to the cost $\tilde{c}(\mathbf{x}_t, \mathbf{u}_t)$ and the dynamics $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$, and $Q_{\mathbf{u},\mathbf{u}t}$ is the quadratic term in the Q-function at time step t. All of these terms can be obtained from a standard LQR backward pass (Li and Todorov, 2004), which we summarize below.

Recall that the estimated linear-Gaussian dynamics have the form $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(f_{\mathbf{x}t}\mathbf{x}_t + f_{\mathbf{u}t}\mathbf{u}_t + f_{ct}, \mathbf{F}_t)$. The quadratic cost approximation has the form

$$\tilde{c}(\mathbf{x}_t, \mathbf{u}_t) \approx \frac{1}{2}[\mathbf{x}_t; \mathbf{u}_t]^{\mathrm{T}}\tilde{c}_{\mathbf{xu},\mathbf{xu}t}[\mathbf{x}_t; \mathbf{u}_t] + [\mathbf{x}_t; \mathbf{u}_t]^{\mathrm{T}}\tilde{c}_{\mathbf{xu}t} + \text{const},$$

where subscripts denote derivatives, e.g. $\tilde{c}_{\mathbf{xu}t}$ is the gradient of $\tilde{c}$ with respect to $[\mathbf{x}_t; \mathbf{u}_t]$, while $\tilde{c}_{\mathbf{xu},\mathbf{xu}t}$ is the Hessian.[4] Under this model of the dynamics and cost function, the

4. We assume that all Taylor expansions here are recentered around zero. Otherwise, the point around which the derivatives are computed must be subtracted from $\mathbf{x}_t$ and $\mathbf{u}_t$ in all of these equations.

optimal controller can be computed by recursively computing the quadratic Q-function and value function, starting with the last time step. These functions are given by

$$V(\mathbf{x}_t) = \frac{1}{2}\mathbf{x}_t^{\mathrm{T}} V_{\mathbf{x},\mathbf{x}t}\mathbf{x}_t + \mathbf{x}_t^{\mathrm{T}} V_{\mathbf{x}t} + \text{const}$$
$$Q(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2}[\mathbf{x}_t; \mathbf{u}_t]^{\mathrm{T}} Q_{\mathbf{xu},\mathbf{xu}t}[\mathbf{x}_t; \mathbf{u}_t] + [\mathbf{x}_t; \mathbf{u}_t]^{\mathrm{T}} Q_{\mathbf{xu}t} + \text{const}$$

We can express them with the following recurrence, which is computed starting at the last time step $t = T$ and moving backward through time:

$$\begin{aligned}
Q_{\mathbf{xu},\mathbf{xu}t} &= \tilde{c}_{\mathbf{xu},\mathbf{xu}t} + f_{\mathbf{xu}t}^{\mathrm{T}} V_{\mathbf{x},\mathbf{x}t+1} f_{\mathbf{xu}t} \\
Q_{\mathbf{xu}t} &= \tilde{c}_{\mathbf{xu}t} + f_{\mathbf{xu}t}^{\mathrm{T}} V_{\mathbf{x}t+1} + f_{\mathbf{xu}t}^{\mathrm{T}} V_{\mathbf{x},\mathbf{x}t+1} f_{ct} \\
V_{\mathbf{x},\mathbf{x}t} &= Q_{\mathbf{x},\mathbf{x}t} - Q_{\mathbf{u},\mathbf{x}t}^{\mathrm{T}} Q_{\mathbf{u},\mathbf{u}t}^{-1} Q_{\mathbf{u},\mathbf{x}t} \\
V_{\mathbf{x}t} &= Q_{\mathbf{x}t} - Q_{\mathbf{u},\mathbf{x}t}^{\mathrm{T}} Q_{\mathbf{u},\mathbf{u}t}^{-1} Q_{\mathbf{u}t},
\end{aligned}$$

and the optimal control law is then given by $g(\mathbf{x}_t) = \mathbf{K}_t\mathbf{x}_t + \mathbf{k}_t$, where $\mathbf{K}_t = -Q_{\mathbf{u},\mathbf{u}t}^{-1} Q_{\mathbf{u},\mathbf{x}t}$ and $\mathbf{k}_t = -Q_{\mathbf{u},\mathbf{u}t}^{-1} Q_{\mathbf{u}t}$. If, instead of simply minimizing the expected cost, we instead wish to optimize the maximum entropy objective in Equation (4), the optimal controller is instead linear-Gaussian, with the solution given by $p(\mathbf{u}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{K}_t\mathbf{x}_t + \mathbf{k}_t; Q_{\mathbf{u},\mathbf{u}t}^{-1})$, as shown in prior work (Levine and Koltun, 2013a).

Appendix B. Experimental Setup Details

In this appendix, we present a detailed summary of the experimental setup for our simulated and real-world experiments.

B.1 Simulated Experiment Details

All of the simulated experiments used the MuJoCo simulation package (Todorov et al., 2012), with simulated frictional contacts and torque motors at the joints used for actuation. Although no control or state noise was added during simulation, noise was injected naturally by the linear-Gaussian controllers. The linear-Gaussian controllers $p_i(\mathbf{u}_t|\mathbf{x}_t)$ were initialized to stay near the initial state $\mathbf{x}_1$ using linear feedback based on a proportional-derivative control law for all tasks, except for the octopus arm, where $p_i(\mathbf{u}_t|\mathbf{x}_t)$ was initialized to be zero mean with a fixed spherical covariance, and the walker, which was initialized to track a demonstration trajectory with proportional-derivative feedback. The walker was the only task that used a demonstration, as described previously. We describe the details of each system below.

Peg insertion: The 2D peg insertion task has 6 state dimensions (joint angles and angular velocities) and 2 action dimensions. The 3D version of the task has 12 state dimensions, since the arm has 3 degrees of freedom at the shoulder, 1 at the elbow, and 2 at the wrist. Trials were 8 seconds in length and simulated at 100 Hz, resulting in 800 time steps per rollout. The cost function is given by

$$\ell(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2} w_{\mathbf{u}} \|\mathbf{u}_t\|^2 + w_{\mathbf{p}} \ell_{12}(\mathbf{p}_{\mathbf{x}_t} - \mathbf{p}^\star),$$

where $\mathbf{p}_{\mathbf{x}_t}$ is the position of the end effector for state $\mathbf{x}_t$, $\mathbf{p}^\star$ is the desired end effector position at the bottom of the slot, and the norm $\ell_{12}(z)$ is given by $\frac{1}{2}\|z\|^2 + \sqrt{\alpha + z^2}$, which corresponds to the sum of an ℓ_2 and soft ℓ_1 norm. We use this norm to encourage the peg to precisely reach the target position at the bottom of the hole, but to also receive a larger penalty when far away. The task also works well in 2D with a simple ℓ_2 penalty, though we found that the 3D version of the task takes longer to insert the peg all the way into the hole without the ℓ_1-like square root term. The weights were set to $w_{\mathbf{u}} = 10^{-6}$ and $w_{\mathbf{p}} = 1$. Initial states were chosen by moving the shoulder of the arm relative to the hole, with four equally spaced starting states in a 20 cm region for the 2D arm, and four random starting states in a 10 cm radius for the 3D arm.

Octopus arm: The octopus arm consists of six four-sided chambers. Each edge of each chamber is a simulated muscle, and actions correspond to contracting or relaxing the muscle. The state space consists of the positions and velocities of the chamber vertices. The midpoint of one edge of the first chamber is fixed, resulting in a total of 25 degrees of freedom: the 2D positions of the 12 unconstrained points, and the orientation of the first edge. Including velocities, the total dimensionality of the state space is 50. The cost function depends on the activation of the muscles and distance between the tip of the arm and the target point, in the same way as for peg insertion. The weights are set to $w_{\mathbf{u}} = 10^{-3}$ and $w_{\mathbf{p}} = 1$.

Swimmer: The swimmer consists of 3 links and 5 degrees of freedom, including the global position and orientation which, together with the velocities, produces a 10 dimensional state space. The swimmer has 2 action dimensions corresponding to the torques between joints. The simulation applied drag on each link of the swimmer to roughly simulate a fluid, allowing it to propel itself. The rollouts were 20 seconds in length at 20 Hz, resulting in 400 time steps per rollout. The cost function for the swimmer is given by

$$\ell(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2} w_{\mathbf{u}} \|\mathbf{u}_t\|^t + \frac{1}{2} w_v \|v_{x\mathbf{x}_t} - v_x^\star\|^2$$

where $v_{x\mathbf{x}_t}$ is the horizontal velocity, $v_x^\star = 2.0\text{m/s}$, and the weights were $w_{\mathbf{u}} = 2 \cdot 10^{-5}$ and $w_v = 1$.

Walker: The bipedal walker consists of a torso and two legs, each with three links, for a total of 9 degrees of freedom and 18 dimensions, with velocity, and 6 action dimensions. The simulation ran for 5 seconds at 100 Hz, for a total of 500 time steps. The cost function is given by

$$\ell(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2} w_{\mathbf{u}} \|\mathbf{u}_t\|^t + \frac{1}{2} w_v \|v_{x\mathbf{x}_t} - v_x^\star\|^2 + \frac{1}{2} w_h \|p_{y_{\mathbf{x}_t}} - p_y^\star\|^2$$

where $v_{x\mathbf{x}_t}$ is again the horizontal velocity, $p_{y_{\mathbf{x}_t}}$ is the vertical position of the root, $v_x^\star = 2.1\text{m/s}$, $p_y^\star = 1.1\text{m}$, and the weights were set to $w_{\mathbf{u}} = 10^{-4}$, $w_v = 1$, and $w_h = 1$.

B.2 Robotic Experiment Details

All of the robotic experiments were conducted on a PR2 robot. The robot was controlled at 20 Hz via direct effort control,[5] and camera images were recorded using the RGB camera on a PrimeSense Carmine sensor. The images were downsampled to $240 \times 240 \times 3$. The learned policies controlled one 7 DoF arm of the robot, while the other arm was used to move objects in the scene to automatically vary the initial conditions. The camera was kept fixed in each experiment. Each episode was 5 seconds in length. For each task, the cost function required placing the object held in the gripper at a particular location (which might require, for example, to insert a shape into a shape sorting cube). The cost was given by the following equation:

$$\ell(\mathbf{x}_t, \mathbf{u}_t) = w_{\ell_2} d_t^2 + w_{\log} \log(d_t^2 + \alpha) + w_{\mathbf{u}} \|\mathbf{u}_t\|^2,$$

where d_t is the distance between three points in the space of the end-effector and their target positions,[6], and the weights are set to $w_{\ell_2} = 10^{-3}$, $w_{\log} = 1.0$, and $w_{\mathbf{u}} = 10^{-2}$. The quadratic term encourages moving the end-effector toward the target when it is far, while the logarithm term encourages placing it precisely at the target location, as discussed in prior work (Levine et al., 2015). The bottle cap task used an additional cost term consisting of a quadratic penalty on the difference between the wrist angular velocity and a target velocity.

For all of the tasks, we initialized all of the linear-Gaussian controllers $p_i(\mathbf{u}_t|\mathbf{x}_t)$ to stay near the initial state $\mathbf{x}_1$, with a diagonal noise covariance. The covariance of the noise was chosen to be proportional to a diagonal approximation of the inverse effective mass at each joint, as provided by the manufacturer of the PR2 robot, and the feedback controller was constructed using LQR, with an approximate linear model obtained from the same diagonal inverse mass matrix. The role of this initial controller was primarily to avoid dangerous actions during the first iteration. We discuss the particular setup for each experiment below:

Coat hanger: The coat hanger task required the robot to hang a coat hanger on a clothes rack. The coat hanger was grasped at one of two angles, about 35° apart, and the rack was positioned at three different distances from the robot during training, with differences of about 10 cm between each position. The rack was moved manually between these positions during training. A trial was considered successful if, when the coat hanger was released, it remained hanging on the rack rather than dropping to the ground.

Shape sorting cube: The shape sorting cube task required the robot to insert a red trapezoid into a trapezoidal hole on a shape sorting cube. During training, the cube was positioned at nine different positions, situated at the corners, edges, and middle of a rectangular region 16 cm × 10 cm in size. During training, the shape sorting cube was moved through the training positions by using the left arm. A trial was considered successful if the bottom face of the trapezoid was completely inside the shape sorting cube, such that if the robot were to release the trapezoid, it would fall inside the cube.

5. The PR2 robot does not provide for closed loop torque control, but instead supports an effort control interface that directly sets feedforward motor voltages. In practice, these voltages are roughly proportional to feedforward torques, but are also affected by friction and damping.
6. Three points fully define the pose of the end-effector. For the bottle cap task, which is radially symmetric, we use only two points.

Toy hammer: The hammer task required the robot to insert the claw of a toy hammer underneath a toy plastic nail, placing the claw around the base of the nail. The hammer was grasped at one of three angles, each 22.5° apart, for a total variation of 45° degrees, and the nail was positioned at five positions, at the corners and center of a rectangular region 10 cm × 7 cm in size. During training, the toy tool bench containing the nail was moved using the left arm. A trial was considered successful if the tip of the claw of the hammer was at least under the centerline of the nail.

Bottle cap: The bottle cap task required the robot to screw a cap onto a bottle at various positions. The bottle was located at nine different positions, situated at the corners, edges, and middle of a rectangular region 16 cm × 10 cm in size, and the left arm was used to move the bottle through the training positions. A trial was considered successful if, after completion, the cap could not be removed from bottle simply by pulling vertically.

References

J. A. Bagnell and J. Schneider. Covariant policy search. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

B. Bakker, V. Zhumatiy, G. Gruener, and J. Schmidhuber. A robot that reinforcement-learns to identify and memorize important previous observations. In *International Conference on Intelligent Robots and Systems (IROS)*, 2003.

G. Bekey and K. Goldberg. *Neural Networks in Robotics*. Springer US, 1992.

H. Benbrahim and J. A. Franklin. Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems*, 22:283–302, 1997.

W. Böhmer, S. Grünewälder, Y. Shen, M. Musial, and K. Obermayer. Construction of approximation spaces for reinforcement learning. *Journal of Machine Learning Research*, 14(1):2067–2118, January 2013.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1122, 2011.

D. Ciresan, U. Meier, J. Masci, L. Gambardella, and J. Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR)*, 2012.

M. Deisenroth and C. Rasmussen. PILCO: a model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, 2011.

M. Deisenroth, C. Rasmussen, and D. Fox. Learning to control a low-cost manipulator using data-efficient reinforcement learning. In *Robotics: Science and Systems (RSS)*, 2011.

M. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013.

J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot. *International Journal of Robotic Research*, 27(2):213–228, 2008.

I. Endres and D. Hoiem. Category independent object proposals. In *European Conference on Computer Vision (ECCV)*. 2010.

Y. Engel, P. Szabó, and D. Volkinshtein. Learning to control an octopus arm with Gaussian process temporal difference methods. In *Advances in Neural Information Processing Systems (NIPS)*, 2005.

B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3), 1992.

C. Finn, X. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Learning visual feature spaces for robotic manipulation with deep spatial autoencoders. *arXiv preprint arXiv:1509.06113*, 2015.

K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

T. Geng, B. Porr, and F. Wörgötter. Fast biped walking with a reflexive controller and realtime policy searching. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014a.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014b.

V. Gullapalli. A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural Networks*, 3(6):671–692, 1990.

V. Gullapalli. Skillful control under uncertainty via direct reinforcement learning. *Reinforcement Learning and Robotics*, 15(4):237–246, 1995.

X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang. Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

R. Hadsell, P. Sermanet, J. B. A. Erkan, and M. Scoffier. Learning long-range vision for autonomous off-road driving. *Journal of Field Robotics*, pages 120–144, 2009.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In *A Field Guide to Dynamic Recurrent Neural Networks*. IEEE Press, 2001.

K. J. Hunt, D. Sbarbaro, R. Żbikowski, and P. J. Gawthrop. Neural networks for control systems: A survey. *Automatica*, 28(6):1083–1112, November 1992.

D. Jacobson and D. Mayne. *Differential Dynamic Programming*. Elsevier, 1970.

M. Jägersand, O. Fuentes, and R. C. Nelson. Experimental evaluation of uncalibrated visual servoing for precision manipulation. In *International Conference on Robotics and Automation (ICRA)*, 1997.

Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

S. Jodogne and J. H. Piater. Closed-loop learning of visual control policies. *Journal of Artificial Intelligence Research*, 28:349–391, 2007.

R. Jonschkowski and O. Brock. State representation learning in robotics: Using prior knowledge about physical interaction. In *Proceedings of Robotics: Science and Systems*, 2014.

M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal. Learning force control policies for compliant manipulation. In *International Conference on Intelligent Robots and Systems (IROS)*, 2011.

S. M. Khansari-Zadeh and A. Billard. BM: An iterative algorithm to learn stable non-linear dynamical systems with Gaussian mixture models. In *International Conference on Robotics and Automation (ICRA)*, 2010.

J. Kober and J. Peters. Learning motor primitives for robotics. In *International Conference on Robotics and Automation (ICRA)*, 2009.

J. Kober, K. Muelling, O. Kroemer, C.H. Lampert, B. Schoelkopf, and J. Peters. Movement templates for learning of hitting and batting. In *International Conference on Robotics and Automation (ICRA)*, 2010a.

J. Kober, E. Oztop, and J. Peters. Reinforcement learning to adjust robot movements to new situations. In *Robotics: Science and Systems (RSS)*, 2010b.

J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotic Research*, 32(11):1238–1274, 2013.

N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *International Conference on Robotics and Automation (IROS)*, 2004.

J. Koutník, G. Cuccu, J. Schmidhuber, and F. Gomez. Evolving large-scale neural networks for vision-based reinforcement learning. In *Conference on Genetic and Evolutionary Computation*, GECCO '13, 2013.

A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. 2012.

T. Lampe and M. Riedmiller. Acquiring visual servoing reaching and grasping skills using neural reinforcement learning. In *International Joint Conference on Neural Networks (IJCNN)*, 2013.

A. Lanfranco, A. Castellanos, J. Desai, and W. Meyers. Robotic surgery: a current perspective. *Annals of surgery*, 239(1):14, 2004.

S. Lange, M. Riedmiller, and A. Voigtlaender. Autonomous reinforcement learning on raw visual input data in a real world application. In *International Joint Conference on Neural Networks*, 2012.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems (NIPS)*, 1989.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, May 2015.

H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning (ICML)*, 2009.

Ian Lenz, Ross Knepper, and Ashutosh Saxena. DeepMPC: Learning deep latent features for model predictive control. In *RSS*, 2015a.

Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *IJRR*, 2015b.

S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

S. Levine and V. Koltun. Guided policy search. In *International Conference on Machine Learning (ICML)*, 2013a.

S. Levine and V. Koltun. Variational policy search via trajectory optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2013b.

S. Levine and V. Koltun. Learning complex neural network policies with trajectory optimization. In *International Conference on Machine Learning (ICML)*, 2014.

S. Levine, N. Wagener, and P. Abbeel. Learning contact-rich manipulation skills with guided policy search. In *International Conference on Robotics and Automation (ICRA)*, 2015.

F. L. Lewis, A. Yesildirak, and S. Jagannathan. *Neural Network Control of Robot Manipulators and Nonlinear Systems*. Taylor & Francis, Inc., 1998.

W. Li and E. Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)*, pages 222–229, 2004.

T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

R. Lioutikov, A. Paraschos, G. Neumann, and J. Peters. Sample-based information-theoretic stochastic optimal control. In *International Conference on Robotics and Automation*, 2014.

H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber. A system for robotic heart surgery that learns to tie knots using recurrent neural networks. In *International Conference on Intelligent Robots and Systems (IROS)*, 2006.

W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, Victor Eruhimov, T. Foote, J. Hsu, R.B. Rusu, B. Marthi, G. Bradski, K. Konolige, B. Gerkey, and E. Berger. Autonomous door opening and plugging in with a personal robot. In *International Conference on Robotics and Automation (ICRA)*, 2010.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing Atari with deep reinforcement learning. *NIPS '13 Workshop on Deep Learning*, 2013.

K. Mohta, V. Kumar, and K. Daniilidis. Vision based control of a quadrotor for perching on planes and lines. In *International Conference on Robotics and Automation (ICRA)*, 2014.

I. Mordatch and E. Todorov. Combining the benefits of function approximation and trajectory optimization. In *Robotics: Science and Systems (RSS)*, 2014.

A. Y. Ng, H. J. Kim, M. I. Jordan, and S. Sastry. Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*, 2004.

R. Pascanu and Y. Bengio. On the difficulty of training recurrent neural networks. Technical Report arXiv:1211.5063, Universite de Montreal, 2012.

B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3D geometry to deformable part models. In *Computer Vision and Pattern Recognition (CVPR)*, 2012.

J. Peters and S. Schaal. Applying the episodic natural actor-critic architecture to motor primitive learning. In *European Symposium on Artificial Neural Networks (ESANN)*, 2007.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

J. Peters, K. Mülling, and Y. Altün. Relative entropy policy search. In *AAAI Conference on Artificial Intelligence*, 2010.

Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *CoRR*, abs/1509.06825, 2015.

D. Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems (NIPS)*, 1989.

S. Ross, G. Gordon, and A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *Journal of Machine Learning Research*, 15:627–635, 2011.

S. Ross, N. Melik-Barkhudarov, K. Shaurya Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert. Learning monocular reactive UAV control in cluttered natural environments. In *International Conference on Robotics and Automation (ICRA)*, 2013.

R. Rubinstein and D. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004.

S. Savarese and L. Fei-Fei. 3D generic object categorization, localization and pose estimation. In *International Conference on Computer Vision (ICCV)*, 2007.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61: 85–117, 2015.

P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition*, 2003.

F. Stulp and O. Sigaud. Path integral policy improvement with covariance matrix adaptation. In *International Conference on Machine Learning (ICML)*, 2012.

Jaeyong Sung, Seok Hyun Jin, and Ashutosh Saxena. Robobarista: Object part based transfer of manipulation trajectories from crowd-sourcing in 3d pointclouds. *CoRR*, abs/1504.03071, 2015.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

R. Tedrake, T. Zhang, and H. Seung. Stochastic policy gradient reinforcement learning on a simple 3d biped. In *International Conference on Intelligent Robots and Systems (IROS)*, 2004.

E. Theodorou, J. Buchli, and S. Schaal. Reinforcement learning of motor skills in high dimensions. In *International Conference on Robotics and Automation (ICRA)*, 2010.

E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 2013.

H. van Hoof, J. Peters, and G. Neumann. Learning of non-parametric control policies with high-dimensional state features. In *International Conference on Artificial Intelligence and Statistics*, 2015.

H. Wang and A. Banerjee. Bregman alternating direction method of multipliers. In *Advances in Neural Information Processing Systems (NIPS)*. 2014.

M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advanced in Neural Information Processing Systems (NIPS)*, 2015.

R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, May 1992.

W. J. Wilson, C. W. Williams Hulls, and G. S. Bell. Relative end-effector control using cartesian position based visual servoing. *IEEE Transactions on Robotics and Automation*, 12(5), 1996.

K.A. Wyrobek, E.H. Berger, HF M. Van der Loos, and K. Salisbury. Towards a personal robotics development platform: Rationale and design of an intrinsically safe personal robot. In *International Conference on Robotics and Automation (ICRA)*, 2008.

B. H. Yoshimi and P. K. Allen. Active, uncalibrated visual servoing. In *International Conference on Robotics and Automation (ICRA)*, 1994.

Journal of Machine Learning Research 17 (2016) 1-45 Submitted 8/13; Revised 3/15; Published 4/16

On Quantile Regression in Reproducing Kernel Hilbert Spaces with the Data Sparsity Constraint

Chong Zhang CHONG.ZHANG@UWATERLOO.CA
Department of Statistics and Actuarial Science
University of Waterloo
Waterloo, ON N2L 3G1, Canada

Yufeng Liu YFLIU@EMAIL.UNC.EDU
Department of Statistics and Operations Research
Department of Genetics
Department of Biostatistics
Carolina Center for Genome Sciences
The University of North Carolina at Chapel Hill
Chapel Hill, NC 27599, USA

Yichao Wu WU@STAT.NCSU.EDU
Department of Statistics
North Carolina State University
Raleigh, NC 27695, USA

Editor: Saharon Rosset

Abstract

For spline regressions, it is well known that the choice of knots is crucial for the performance of the estimator. As a general learning framework covering the smoothing splines, learning in a Reproducing Kernel Hilbert Space (RKHS) has a similar issue. However, the selection of training data points for kernel functions in the RKHS representation has not been carefully studied in the literature. In this paper we study quantile regression as an example of learning in a RKHS. In this case, the regular squared norm penalty does not perform training data selection. We propose a data sparsity constraint that imposes thresholding on the kernel function coefficients to achieve a sparse kernel function representation. We demonstrate that the proposed data sparsity method can have competitive prediction performance for certain situations, and have comparable performance in other cases compared to that of the traditional squared norm penalty. Therefore, the data sparsity method can serve as a competitive alternative to the squared norm penalty method. Some theoretical properties of our proposed method using the data sparsity constraint are obtained. Both simulated and real data sets are used to demonstrate the usefulness of our data sparsity constraint.

Keywords: kernel learning, Rademacher complexity, regression, smoothing, sparsity

1. Introduction

Regression is one of the most important and commonly used statistical tools. Given a set of data points whose predictors and responses are both available, one builds a regression model to predict the response variable for any new instance with only predictors observed. When solving a regression problem, linear regression can be insufficient. In particular, when the response has highly nonlinear dependence on the predictors, linear models can be suboptimal. To overcome this difficulty, various nonlinear regression models such as kernel smoothers (see Hastie et al., 2009, for a review) and splines (De Boor, 1978) can be used. The main idea is to find a regression function in a nonlinear functional class that best fits the response variable.

In a typical spline regression problem with a univariate predictor, one can use a piecewise nonlinear function as the regression function. The function is smooth everywhere including at the knots, where the nonlinear pieces connect. The knots play a crucial role in spline regression. For the regular smoothing splines (see, for example, Wahba, 1990; Gu, 2002, and the references therein), the knots are located at the observed predictor values automatically. For some other types of spline regressions, one needs to determine the knots. For instance, B-splines (De Boor, 1978) commonly use a set of equally spaced knots, and certain types of P-splines (Eilers and Marx, 1996; Ruppert et al., 2003) take knots based on quantiles of the predictor variable.

For spline regression, it is known that too many knots may lead to overfitting and unnecessary fluctuation in the resulting estimator. For instance, based on Chappell (1989), Koenker et al. (1994) gave an example where the regular smoothing splines perform poorly because of too many change points, and the one-change-point method proposed by Chappell (1989) works much better. Extensive efforts have been devoted on how to choose the knots for B-spline and P-spline methods in the literature (see, for example, Friedman and Silverman, 1989; Eilers and Marx, 1996; Zhou et al., 1998; Ruppert, 2002; Hansen and Kooperberg, 2002; Mao and Zhao, 2003; Miyata and Shen, 2005; Gervini, 2006; Eilers and Marx, 2010, and the references therein).

In this paper, we consider multi-dimensional regression problems with the regression function in a Reproducing Kernel Hilbert Space (RKHS, Aronszajn, 1950; Schölkopf and Smola, 2002). This is a very general setting, which includes many well known regression techniques as special cases, for example penalized linear regressions, additive spline models with or without interactions, and the entire family of smoothing splines. Typically, the optimization of such a RKHS regression can be written in a *loss + penalty* form. Since the regression function is assumed to be in a RKHS, it is common to take the squared norm of the function as the penalty. By the well celebrated representer's theorem (Kimeldorf and Wahba, 1971), the resulting regression function can be represented as a linear combination of kernel functions determined by the training data.

Our motivation for this paper is based on the following observation. The kernel representation of the regression function is similar to the knot structure in smoothing splines, in the sense that each observation in the training data can be regarded as a "knot" in a multi-dimensional space. In particular, when we restrict the RKHS regression to the smoothing splines, the kernel function representation is equivalent to the piecewise nonlinear function representation. With the regular squared norm penalty, the resulting estimator involves all kernel functions on the training data. For large sample size problems, this estimator is known to be consistent with desirable theoretical properties. However, for problems with relatively smaller numbers of observations, using all kernel functions for the representation may introduce a similar issue as using too many knots in spline regressions. Hence it is desirable to have a regularization method that can select the kernel functions.

To this end, we propose a new penalty method to achieve a "data sparsity" model. Through simulation studies, we observe that for some cases, the data sparsity model can perform better than the regular squared norm penalty method, and for other cases, their performance is comparable. See Section 2 for more detailed discussions. Moreover, we provide some theoretical insights on the data sparsity method. In particular, we show that under very mild conditions, the asymptotic convergence rates of the estimation errors for the two methods are the same, and both are close to the "parametric rate". Furthermore, we give finite sample error bounds on the prediction errors for both

methods. We show that for a general RKHS and problems with small sample sizes, the bound for the squared norm penalty method can be large. On the other hand, the data sparsity method can enjoy a better bound because its corresponding functional space is smaller. Hence, we propose the data sparsity method as an alternative approach to the squared norm penalty for RKHS learning. Note that in the literature, Takeuchi et al. (2006) studied kernel based nonparametric quantile regression problems, and mentioned a similar method as a natural extension of their formulation. However, their work focused on nonparametric quantile regression, whereas the possible overfitting of the squared norm penalty wasn't brought to attention. Moreover, Takeuchi et al. (2006) didn't perform detailed theoretical or numerical studies on the data sparsity constraint. Our important contribution in this paper is to explore the similarity and (more importantly) differences between the data sparsity constraint and the regular squared norm penalty, through both numerical and theoretical studies.

In a regression problem, one needs to choose the loss function. The commonly used loss function is the squared error loss, which estimates the conditional mean of the response given the predictors. It is known that compared to the conditional mean estimation, the conditional median estimation is more robust against outliers. Therefore, in this paper, we consider quantile regression, and the loss function we use is the check function (Koenker and Bassett, 1978), although the idea of imposing data sparsity constraint is very general and can be applied to many other settings as well. Note that quantile regression with the check function provides the conditional median estimation as a special case. Another advantage is that it can provide more information on the conditional distribution of the response. Quantile regression has been widely used in many scientific fields, including survival analysis (Koenker and Geling, 2001), microarray study (Wang and He, 2007), economics (Koenker and Hallock, 2001), growth chart (Wei and He, 2006), and many others. Note that quantile regression in RKHS with the regular squared norm penalty was previously studied by Takeuchi et al. (2006) and Li et al. (2007).

In the machine learning literature, the Support Vector Machine (SVM, Boser et al., 1992; Cortes and Vapnik, 1995) and the Support Vector Regression (SVR, Drucker et al., 1997; Vapnik et al., 1997; Smola and Schölkopf, 1998; Stitson et al., 1999; Smola and Schölkopf, 2004) have been well studied and widely used as classification and regression tools. One attractive feature of the SVM and SVR is that even with the regular square norm penalty, due to the choice of the loss functions, the estimated classification function or regression function has a sparse representation in the dual space of the corresponding optimization problem. If the classification or regression function is in a RKHS, sparsity in the dual space representation is equivalent to sparsity in the kernel representation. However, with many other loss functions and the squared norm penalty, the advantage of a sparse representation is lost (Smola and Schölkopf, 2004). Our proposed data sparsity constraint is able to provide such a sparse representation for general loss functions. Note that in the Bayesian learning literature, Tipping (2001) proposed the relevance vector machines to obtain sparse solutions for regression and classification problems.

The rest of this article is organized as follows. In Section 2, we first discuss quantile regression problems under the RKHS learning, then introduce our data sparsity constraint. In Section 3, we derive theoretical results for both asymptotic and finite sample analysis of our data sparsity method. In Section 4, we discuss how to derive the solution path of the involved optimization problem with respect to the tuning parameter, and tackle the problem of tuning parameter selection. In Section 5, we demonstrate the performance of our data sparsity method, using both simulated and real data sets. Some discussions are provided in Section 6. All technical proofs are collected in the appendix.

2. Methodology

We are given the training data (x_i, y_i); $i = 1, \ldots, n$, which are observed according to the model $Y = f_0(X) + \varepsilon(X)$. Let D be the domain of f_0, and let the dimensionality of D be p. We assume that for any given X, $\varepsilon(X)$ has a finite mean. This assumption on ε is very general, in the sense that both the homoscedastic and heteroscedastic cases are covered, along with many commonly used distributions. To estimate the $100\tau\%$ quantile of the conditional distribution of Y given X for some quantile level $\tau \in (0,1)$, Koenker and Bassett (1978) proposed to use the check loss function, which can be written as

$$\rho_\tau(u) = \begin{cases} \tau u & \text{if } u > 0, \\ -(1-\tau)u & \text{if } u \le 0, \end{cases}$$

where $\tau \in (0,1)$ indicates the quantile we are interested in. It is known that for a given X, the population minimizer to the check function is the $100\tau\%$ conditional quantile. For a given τ, suppose that $f_{\text{true}}(X)$ is the population minimizer to the check function. Note that in general $f_0 \neq f_{\text{true}}$. A regular quantile regression problem can be typically formulated in terms of the following optimization

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \rho_\tau\big(y_i - f(x_i)\big) + \lambda J(f), \tag{1}$$

where $\mathcal{F}$ is the functional class we are interested in, $J(\cdot)$ is a penalty on f to prevent overfitting, and λ is a tuning parameter that controls the magnitude of $J(\cdot)$. With $p = 1$, $J(f) = \int |\frac{d^2 f(x)}{dx^2}| dx$, and an appropriately chosen $\mathcal{F}$, Koenker et al. (1994) showed that the solution to (1) is a linear spline with knots at x_i; $i = 1, \ldots, n$.

In this article, we consider the case with $p \ge 1$ and the regression function in a RKHS, which is a more general setting than the regular smoothing splines. To begin with, we introduce some notations. A summary of important notations used in this paper can be found in Tables 11-14. Assume $\mathcal{F} = \{f = f' + b : f' \in \mathcal{H},\ b \in \mathbb{R}\}$, where $\mathcal{H}$ is a RKHS over X with the kernel function $K(\cdot,\cdot)$, and b is the intercept of the regression function. Throughout this paper, we use the notation f' for any function when it belongs to a RHKS without an extra intercept term. This definition of $\mathcal{F}$ allows a more flexible setting than $\mathcal{F} = \mathcal{H}$, because some RKHS's, for example the very popular Gaussian RKHS, do not include non-zero constant functions (Minh, 2010). In this paper, without loss of generality we assume each $f \in \mathcal{F}$ can be uniquely decomposed as $f' + b$. Let the norm in $\mathcal{H}$ be $\|\cdot\|_{\mathcal{H}}$. For more detailed discussions about $\mathcal{H}$ and $\|\cdot\|_{\mathcal{H}}$, we refer the readers to Aronszajn (1950), Wahba (1999), Schölkopf and Smola (2002), Steinwart et al. (2006), Hofmann et al. (2008), Minh (2010), and the references therein. Furthermore, we assume that the RKHS $\mathcal{H}$ is separable, and the kernel function $K(\cdot,\cdot)$ is upper bounded by 1. Our theory can be generalized to the case where $\sup_{X_1, X_2} K(X_1, X_2) < \infty$. Note that a similar assumption was previously used in Steinwart and Scovel (2007) and Blanchard et al. (2008). With a little abuse of notation, we define K to be the n by n matrix $\Big(K(x_i, x_j)\Big)$; $i, j = 1, \ldots, n$, which we call the gram matrix.

The quantile regression with the regular squared norm penalty (Takeuchi et al., 2006; Li et al., 2007) solves

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \rho_\tau\big(y_i - f(x_i)\big) + \lambda \|f'\|_{\mathcal{H}}^2. \tag{2}$$

By the representer's theorem, the solution to (2) can be written as

$$\tilde{f}_n(x) = \tilde{f}_n'(x) + \tilde{b} = \sum_{i=1}^{n} \tilde{\alpha}_i K(x_i, x) + \tilde{b}, \tag{3}$$

where $K(x_i,\cdot)$ is the i^{th} kernel function from the training sample, and $\tilde{\alpha} = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_n)^T$ is the estimated kernel function coefficient vector. In this paper, to clarify notation, we denote the estimated function using the squared norm penalty by $\tilde{f}_n$, and the estimated function using our proposed data sparsity constraint by $\hat{f}_n$. Because $\tilde{f}_n$ possesses such a finite form, (2) can be equivalently written as

$$\min_{\alpha,b} \frac{1}{n} \sum_{i=1}^{n} \rho_\tau\{y_i - \big(\sum_{j=1}^{n} \alpha_j K(x_i, x_j) + b\big)\} + \lambda \alpha^T K \alpha. \tag{4}$$

Li et al. (2007) provided a solution path for (4), with respect to the tuning parameter λ.

For many commonly used kernel functions, we can assume that the gram matrix K is positive definite (Paulsen, 2009). Hence for any α_i; $i = 1, \ldots, n$, the penalty $\alpha^T K \alpha$ in (4) constrains α in an ellipsoid, which does not have any singularity at $\alpha_i = 0$. This is illustrated on the left panel of Figure 1. Note that in the linear learning literature, Fan and Li (2006) discussed the effect of singularity of penalties on variable selection. Similarly, in RKHS regression problems, because the regular squared norm penalty does not have any singularity at $\alpha_i = 0$, it does not perform "kernel function selection". As a result, the estimated $\tilde{\alpha}_i \neq 0$ for all $i = 1, \ldots, n$.

As discussed in Section 1, it is desirable to have a method that can deliver estimators with a sparse kernel function representation. To this end, we propose to penalize directly on the kernel function coefficients α such that some estimated α_i's will be set to 0. The details of our method are as follows. By the representer's theorem (Kimeldorf and Wahba, 1971), the estimated $\tilde{f}_n$ in (4) lies in a space linearly spanned by $K(x_i,\cdot)$; $i = 1, \ldots, n$, and $\tilde{\alpha}$ is constrained in an ellipsoid. To obtain a data-sparsely represented function, we propose to constrain α in an L_1 ball. In other words, we solve the following optimization problem with the data sparsity constraint

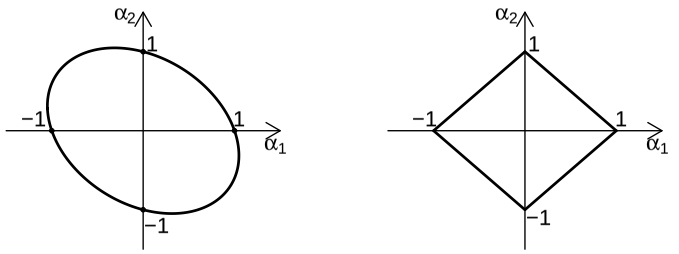
$$\min_{\alpha,b} \frac{1}{n} \sum_{i=1}^{n} \rho_\tau\big(y_i - f(x_i)\big), \text{ subject to } |b| + \sum_{i=1}^{n} |\alpha_i| \leq s, \tag{5}$$

where $s > 0$ is the tuning parameter. Note that Takeuchi et al. (2006) briefly mentioned a possible natural extension of their method that is similar to (5).

The constraint in (5) is an L_1 type regularization and imposes a soft thresholding (Tibshirani, 1996) on α. On the right panel of Figure 1, we illustrate the effect of the data sparsity constraint. For a small s, many of the estimated $\hat{\alpha}_i$ values will be set to 0. Consequently, the regression function $\hat{f}_n$ has a parsimonious representation in (3).

Through simulation studies, we demonstrate that for some settings, the data sparsity method can have a better performance, compared to the regular squared norm penalty method. For other cases, the performance difference between the two approaches is small. In particular,

- when n is small or moderate, and the underlying function can be well approximated by a sparse representation $\sum_{i=1}^{m} \gamma_i K(z_i,\cdot) + c$ for small m's, where z_i are fixed points in D and $c, \gamma_i \in \mathbb{R}$. The data sparsity can then provide a parsimonious model, and the corresponding prediction performance can be better. We demonstrate this issue using an example where $p = 1$ with the Laplacian kernel in Figure 2, and another example with $p = 2$ and the Gaussian kernel in Figure 3;



(a) The regular squared norm penalty. (b) The proposed data sparsity constraint.

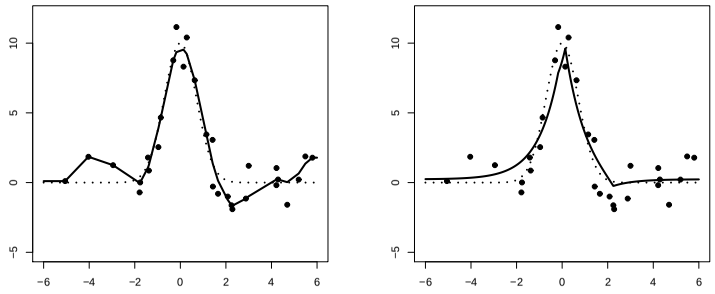
Figure 1: The left panel demonstrates the contour of the regular squared norm penalty $\alpha^T K \alpha = 1$ with $K = \big[(1, 0.3)^T \ (0.3, 1)^T\big]$. The right panel plots the contour of our data sparsity constraint $|\alpha_1| + |\alpha_2| = 1$. For the regular squared norm penalty, there is no singularity at the intersections of the contour and the axes ($\alpha_1 = 0$, $\alpha_2 = 0$), thus it does not encourage sparsity in the estimated kernel function coefficients. In contrast, the data sparsity penalty has singularity at the intersections and is able to achieve sparsity in the estimated kernel function coefficients.

- when n is small or moderate, and the underlying function does not possess such a sparse representation, the data sparsity method tends to choose a large s in (5). As a result, the fitted model is not sparsely represented. In this case, the prediction performance of the data sparsity method and the squared norm penalty method is often comparable. This is illustrated in Figure 4;

- and when n is large, there is enough information to estimate the underlying function accurately, and both methods can perform well in terms of prediction. In particular, we show in Section 3.1 that the estimation errors of (4) and (5) both converge at a rate very close to the "parametric rate" $O_P(n^{-1/2})$. In other words, asymptotically the data sparsity method can perform as well as the squared norm penalty. However, (5) can still provide a data sparse representation model. The advantages of such a parsimonious estimator is that the prediction for new observations can be much faster than the regular method, and a sparser model can be easier to interpret.

Therefore, the data sparsity method can be regarded as an alternative learning technique to the regular squared norm penalty method.

Notice that although we observe that the data sparsity constraint may work well under some settings when n is small or moderate, we still need a certain amount of information from the data

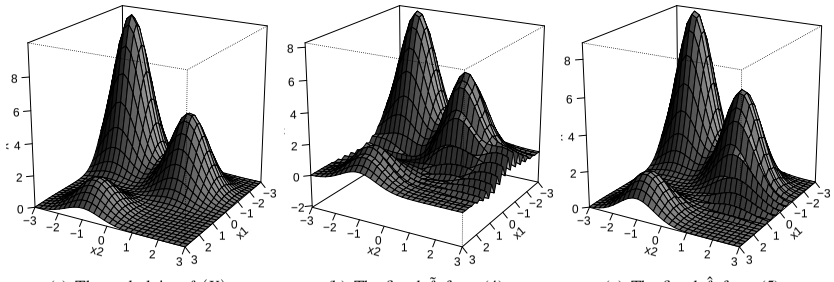


(a) The fitted $\tilde{f}_n$ from (4). (b) The fitted $\hat{f}_n$ from (5).

Figure 2: Plot of the fitted $\tilde{f}_n$ and $\hat{f}_n$ (solid lines) in a simulated example with $n = 30$ and $\tau = 0.5$, using the regular squared norm penalty (left panel) and the proposed data sparsity constraint (right panel). The kernel used is the Laplacian kernel. The error ε follows $U(-2,2)$. The best tuning parameters (λ, s and the kernel parameter) are selected by the GACV criterion (Yuan, 2006). The dotted line is the true regression function. Note that as the regular squared norm does not have sparsity in the estimated kernel function coefficients, the estimated regression function has quite a few wiggles which degrades the prediction performance. On the other hand, our data sparsity method performs remarkably well in this example.

to estimate the underlying function reasonably well. When the dimensionality p is high and the sample size n is small, without variable selection, the curse of dimensionality would prevent most of the kernel methods from working well. Therefore, we focus on the case when p is not large in this paper.

We would like to point out that besides the data sparsity constraint on α, we also impose regularization on b in (5). Although penalizing b may not be standard, some papers, for example Fan et al. (2008), also considered penalizing the intercept. The effect of penalizing on b is two fold. Firstly, it guarantees the uniqueness of the solution path with respect to s, which is discussed in Section 4. Secondly, it prevents b from diverging too fast as $n \to \infty$. This helps to bound the complexity of $\mathcal{F}$ and the functional space we consider in (5), and consequently helps to derive the theoretical properties in Section 3. If we remove b from the constraint, more conditions are needed for the corresponding theorems to be valid. In particular, in the RKHS learning literature, many theoretical results are derived with $f = f' \in \mathcal{H}$ without the intercept term. See, for example, Bousquet and Elisseeff (2002), Chen et al. (2004), and the discussion on Page 17 of Steinwart and Christmann (2008). Our data sparsity constraint can naturally incorporate the intercept in the regularization, and consequently provide desirable theoretical properties without additional assumptions. More discus-



(a) The underlying $f_0(X)$. (b) The fitted $\tilde{f}_n$ from (4). (c) The fitted $\hat{f}_n$ from (5).

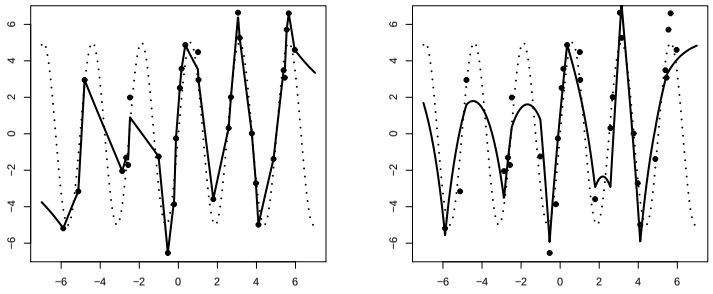
Figure 3: Panel (a) displays the underlying function $f_0(X)$, which has a sparse representation in the Gaussian RKHS. Panel (b) shows the estimated regression function $\tilde{f}_n$ from (4), using a simulated example of size 50 with the Gaussian RKHS and the regular square norm penalty. Panel (c) shows the estimated regression function $\hat{f}_n$ from the same example with our data sparsity constraint in (5). The error ε follows $N(0,1)$, and we use $\tau = 0.5$. We select the best tuning parameters (λ, s and the kernel parameter) over a grid of candidates by the GACV criterion (Yuan, 2006). On Panel (b), one can see that there are fluctuations in $\tilde{f}_n$ which degrade its prediction performance. On the other hand, $\hat{f}_n$ from our data sparsity method has less fluctuations.

sions on this issue are provided in Remark 4 and the proofs of the corresponding theorems in the appendix. In Section 5, we study the numerical performance of our method with and without $|b|$ penalized, respectively. The results demonstrate that the difference between the two settings is small, in terms of their empirical performance. In real data analysis, practitioners can choose whether to penalize b or not based on the nature of the problem and the model.

Next, we derive some theoretical properties of our data sparsity method, as well as the regular squared norm penalty method.

3. Statistical Theory

In this section, we investigate some statistical theory of our data sparsity method. In particular, we study the asymptotic behavior of our data sparsity method and the standard penalized quantile regression with the regular squared norm penalty as $n \to \infty$ in Section 3.1. An example is given in Section 3.2 to calculate the rate of convergence of the approximation error. We discuss the approximation ability of the RKHS in Section 3.3. We also derive some finite sample error bounds in Section 3.4. Note that our main results (Theorems 1-9) require only that the noise $\varepsilon(X)$ has finite mean for all X, therefore they hold in general for both homoscedastic and heteroscedastic cases.



(a) The fitted $\tilde{f}_n$ from (4). (b) The fitted $\hat{f}_n$ from (5).

Figure 4: The left panel shows the fitted $\tilde{f}_n$ from (4) with a simulated example of size 30 and the Laplacian kernel for $\tau = 0.5$. The right panel shows the fitted $\hat{f}_n$ from (5) using the same sample data and τ. The error follows $N(0,1)$. The best tuning parameters (λ, s and the kernel parameter) are selected by minimizing the GACV criterion (Yuan, 2006). Because the underlying function is quite wiggly, a sparse representation in this case cannot perform well. However, by allowing a large s, the data sparsity constraint yields a model that is not "sparse". Therefore one can see that the two methods give comparable performance.

3.1 Asymptotic Results

Before stating our main results, we introduce some additional notations. Let $\mathcal{F}_n(s) = \{f = f' + b : f'(x) = \sum_{i=1}^n \alpha_i K(x,x_i); |b| + \sum_{i=1}^n |\alpha_i| \le s\}$ be the functional space of the optimization problem (5). Note that we can define the functional space of the regular squared norm penalized method in a similar manner. Let $\mathcal{F}(\infty) = \lim_{s\to\infty} \lim_{n\to\infty} \mathcal{F}_n(s)$. Next, we define $f_n^{(s)} = \operatorname{arginf}_{f\in\mathcal{F}_n(s)} E\rho_\tau(Y - f(X))$ and $f^{(\infty)} = \operatorname{arginf}_{f\in\mathcal{F}(\infty)} E\rho_\tau(Y - f(X))$. Here the expectation is taken with respect to the joint distribution of X and the noise ε. Note that the conditional $100\tau\%$ quantile function f_{true} may not belong to $\mathcal{F}(\infty)$. Now let $e(f_1, f_2) = E\rho_\tau(Y - f_1(X)) - E\rho_\tau(Y - f_2(X))$. In the following theorem, we explore the convergence rate of $e(\hat{f}_n, f^{(\infty)})$ by decomposing it into the estimation error $e(\hat{f}_n, f_n^{(s)})$ and the approximation error $e(f_n^{(s)}, f^{(\infty)})$, where $\hat{f}_n = \operatorname{argmin}_{f\in\mathcal{F}_n(s)} \frac{1}{n}\sum_{i=1}^n \rho_\tau(y_i - f(x_i))$. We also study the convergence rate of $e(\tilde{f}_n, f^{(\infty)})$ for the regular method using the squared norm penalty.

Theorem 1 *For the data sparse L_1 method (5), we have $e(\hat{f}_n, f^{(\infty)}) = O_P(\max(sn^{-1/2}\log(n), d_{n,s}))$, where $d_{n,s} = e(f_n^{(s)}, f^{(\infty)})$ is the approximation error between $\mathcal{F}_n(s)$ and $\mathcal{F}(\infty)$.*

For the regular squared norm method with $|b|$ penalized, the estimation error of the solution $\tilde{f}_n$ to

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^n \rho_\tau\big(y_i - f(x_i)\big), \text{ subject to } |b|^2 + \|f'\|_{\mathcal{H}_f}^2 \le s^2 \tag{6}$$

enjoys the same rate as above.

Remark 2 *In Theorem 1, the estimation error converges at the rate $O_P\big(sn^{-1/2}\log(n)\big)$, and $d_{n,s}$ approaches 0 as $s, n \to \infty$. Therefore the optimal value of the tuning parameter s is roughly determined by $sn^{-1/2}\log(n) \approx d_{n,s}$. It can be considered as the trade-off between the approximation error and the estimation error.*

Remark 3 *Theorem 1 is developed for a general separable RKHS such that the kernel function $K(\cdot,\cdot)$ is upper bounded. The results in Section 3 can be refined if one focuses on a smaller set of RKHS's that satisfies additional conditions. For example, the Gaussian RKHS is commonly used in the literature, and the corresponding theoretical properties are well studied (see, for example, Zhou, 2002; Keerthi and Lin, 2003; Steinwart et al., 2006; Steinwart and Scovel, 2007; Minh, 2010, and the references within). In Theorem 1, the width parameter σ of the Gaussian kernel and the dimensionality of X do not affect the convergence rate explicitly. They are both involved in the approximation error $d_{n,s}$. The choice of σ is often data dependent, as described in Section 5. The asymptotic effect of σ is studied in many papers, for example Keerthi and Lin (2003). If f is an element in a Banach space whose norm is defined to be $\|f_1 - f_2\| = |(E\rho_\tau(Y - f_1(X)) - E\rho_\tau(Y - f_2(X)))|$ with an appropriate definition of limits for Cauchy sequences, then the corresponding theory of $d_{n,s}$ can be derived as studied in Cucker and Smale (2002) and Smale and Zhou (2003). In Section 3.2, we give a simple example in which $d_{n,s}$ can be explicitly calculated and vanishes in a rate much faster than the estimation error.*

Remark 4 *The constraint on $|b|$ in (5) and (6) helps to bound the complexity of $\mathcal{F}_n(s)$ in terms of its L_2 entropy number. The definition of the L_2 entropy number is as follows. Let Q be a σ-finite measure on X. One can define the $L_2(Q)$ norm of a square integratable function f on X to be $\|f\|_{L_2(Q)} = (\int f^2 dQ)^{1/2}$. An $\eta-$net on $\mathcal{F}_n(s)$ is defined to be a set of functions $\mathcal{G} = \{g_1, g_2, \ldots\}$ such that for all $f \in \mathcal{F}_n(s)$, there exists a $g \in \mathcal{G}$ satisfying $\|f - g\|_{L_2(Q)} \le \eta$. For any fixed η, the $L_2(Q)$ entropy number of $\mathcal{F}_n(s)$ is defined as the logarithm of the cardinality of an $\eta-$net $\mathcal{G}$ on $\mathcal{F}_n(s)$ whose size is the smallest (Van der Vaart and Wellner, 2000). A bound on $|b|$ helps to control the L_2 entropy number of $\mathcal{F}_n(s)$. See Lemma 14 and its proof in the appendix for more details. Our theory can also be valid with some additional assumptions if $|b|$ is not penalized. The next corollary discusses a natural generalization of our asymptotic results without penalizing $|b|$, when we impose some assumptions on f_0 and ε. First, we define*

$$\mathcal{F}_n^*(s) = \{f = f' + b : f'(x) = \sum_{i=1}^n \alpha_i K(x,x_i); \sum_{i=1}^n |\alpha_i| \le s\},$$

and $f_n^{(*s)}$, $\mathcal{F}^*(\infty)$, and $f^{(*\infty)}$ are defined analogously as in Theorem 1. Note that if a random variable X is sub-Gaussian with the parameter S, then $pr(|X| > t) \le 2\exp(-t^2/S)$ for t large enough.

Corollary 5 *Suppose that f_0 is uniformly bounded, and the error $\varepsilon(X)$ follows a sub-Gaussian distribution with a common finite parameter for any X. Then the solution $\hat{f}_n^*$ to the following optimization*

$$\min_{\alpha,b} \frac{1}{n}\sum_{i=1}^{n} \rho_\tau\big(y_i - f(x_i)\big), \text{ subject to } \sum_{i=1}^{n} |\alpha_i| \le s,$$

satisfies that $e(\hat{f}_n^*, f^{(*\infty)}) = O_P(\max(sn^{-1/2}\log(n), d_{n,s}^*))$, where $d_{n,s}^* = e(f_n^{(*s)}, f^{(*\infty)})$ is the approximation error between $\mathcal{F}_n^*(s)$ and $\mathcal{F}^*(\infty)$.

In Corollary 5, we impose the assumption that the distributions of error terms ε_i; $i = 1,\ldots,n$, are all sub-Gaussian, which covers many commonly used distributions. Consequently, the probability that any observed y_i being significantly away from $f_0(x_i)$ can be well controlled. If the distribution of ε_i has a heavier tail than sub-Gaussian, and we do not penalize b, the convergence rate of the estimation error can be slower than that in Theorem 1. See the proof in the appendix for more discussions. Compared to Corollary 5, our asymptotic theory with $|b|$ in the constraint only requires the noise ε being integrable, hence is more general.

Remark 6 *Note that our results in Theorem 1 also apply to the regular squared norm penalty method. This suggests that asymptotically, the two methods can both perform well. However, for problems with a moderate or small n, asymptotic results may be less useful. In Section 3.4, we give bounds on the prediction errors $E\rho_\tau(Y - \hat{f}_n)$ and $E\rho_\tau(Y - \tilde{f}_n)$. In particular, we give two such bounds. The first bound works for both the regular method with the squared norm penalty and the proposed data sparsity method. The second bound is only for the data sparsity method. We show that for a small n, the second bound can be better than the first one. Therefore, when the true function can be well approximated by functions in $\mathcal{F}_m(s_0)$ for small m and s_0, the data sparsity method can enjoy a smaller prediction error bound.*

In the next section, we give an example where $f_0 \in \mathcal{F}_m(s_0)$ for some fixed s_0 and m, and the approximation error $d_{n,s}$ converges to 0 in a speed much faster than $O_P(n^{-1/2}\log(n))$. In that case, $e(\hat{f}_n, f^{(\infty)}) = O_P(n^{-1/2}\log(n))$.

In the literature, Takeuchi et al. (2006) derived finite sample bounds on the estimation error for general quantile regression problems, using the Rademacher complexity technique (Mohri et al., 2012). They showed that the estimation error can be upper bounded by the Rademacher complexity of the corresponding functional space (plus a small penalty which exists only in finite sample problems). Under various settings where the Rademacher complexity is well studied, one can obtain the asymptotic convergence rate of the estimation error accordingly. For example, when we perform learning with a radial basis function kernel such that $K(\cdot,\cdot)$ is upper bounded, or when the functional space has finite VC dimensions, one can verify that the corresponding Rademacher complexity converges to zero at a rate close or equal to $O_P(n^{-1/2})$. Li et al. (2007) also studied the asymptotic convergence rate of $e(\tilde{f}_n, f^{(\infty)})$ under some assumptions. For example, a bound on the complexity of $\mathcal{F}$ in terms of the L_2 metric entropy was assumed. Our asymptotic theory for quantile regression with the data sparsity constraint is more general, as we only use the assumption that the RKHS is separable and bounded. This is a very weak assumption and can be satisfied by most commonly used kernel spaces. Furthermore, in Section 3.4, we obtain finite sample error bounds on the prediction error for our proposed method with the data sparsity constraint. Our bounds can be directly calculated using the training data and the corresponding tuning parameter.

3.2 An Illustrative Example on the Approximation Error

In this section we give an example to calculate $d_{n,s}$, where we know f_0 and the distribution of X. The Gaussian RKHS is considered.

For simplicity, let $p = 1$, $\tau = 1/2$, $\sigma = 1$ and X be uniformly distributed on $[0,1]$. Moreover, assume that ε is symmetric with respect to 0 for all X. Suppose the underlying model is $f_{\text{true}} = f_0(x) = \exp(-x^2)$. One can verify that when s is fixed at 1, $d_{n,s} \le \frac{1}{2}E(|Y - f_0(X)|) - E(|Y - f_{(1)}(X)|)$, where $f_{(1)}(x) = \exp(-(1-x_{(1)})^2)$ with $x_{(1)}$ being the smallest order statistic of the sample $x = (x_1,\ldots,x_n)$. Note that the probability density function of $x_{(1)}$ is $n(1-x)^{n-1}I_{[0,1]}$ and $d_{n,s} \le \frac{1}{2}E(|f_0(X) - f_{(1)}(X)|)$. Since the largest difference between $f_0(x)$ and $f_{(1)}(x)$ occurs at $x = 0$, $d_{n,s} \le \frac{1}{2}\int_0^1 (1 - \exp(-x^2))n(1-x)^{n-1}dx$. By the Taylor's expansion, one can verify that $(1 - \exp(-x^2)) \le 2x^2$ for all $x \in [0,1]$. Thus, $d_{n,s} \le \frac{1}{2}\int_0^1 2nx^2(1-x)^{n-1}dx = \frac{2}{(n+1)(n+2)}$. Hence in this example, $d_{n,s} = O_P(n^{-2})$, which converges to 0 at a much faster rate than $O_P\big(n^{-1/2}\log(n)\big)$.

In general, when $f_0(x) = \sum_{j=1}^m \gamma_j K(x, z_j) + c$, where $c \in \mathbb{R}$, $\gamma_j \in \mathbb{R}$, and $z_j \in \mathbb{R}^p$ are fixed points (not the observed data points), we can have s fixed and the approximation error vanishes quickly as $n \to \infty$. This is because with growing n, there will be some observed x_i's that are close to z_j; $j = 1,\ldots,m$, and the approximation error $d_{n,s}$ may converge at a rate faster than $O_P(n^{-1/2}\log(n))$ with $s = |c| + \sum_{j=1}^m |\gamma_j|$.

3.3 Approximation Ability of $\mathcal{F}(\infty)$

We have explored the convergence rate of the estimation errors $e(\hat{f}_n, f_n^{(s)})$ and $e(\tilde{f}_n, f_n^{(s)})$ in Section 3.1, and in Section 3.2 we have given an example to illustrate the convergence rate of the approximation error $d_{n,s} = e(f_n^{(s)}, f^{(\infty)})$. For real applications, it is desirable to study the approximation ability of $\mathcal{F}(\infty)$. In other words, how well $f^{(\infty)}$ can approximate f_{true}. However, this approximation ability depends on the properties of f_{true} (i.e., the smoothness, etc.), the richness of the RKHS, and the underlying marginal distribution of X. In the literature of RKHS learning, Steinwart and Scovel (2007) studied the approximation ability of the Gaussian RKHS for support vector machines. In this section, we provide a discussion on this issue for quantile regression with a general RKHS. We measure such approximation ability by $A(\infty) := E\big(\rho_\tau(Y - f^{(\infty)})\big) - E\big(\rho_\tau(Y - f_{\text{true}})\big)$. We first show that if f_{true} is a bounded piecewise step function, an upper bound on $A(\infty)$ for the Gaussian kernel learning can be obtained. This upper bound depends on the marginal distribution of X. In particular, when the marginal distribution of X is absolutely continuous with respect to the Lebesgue measure, the Gaussian RKHS can approximate f_{true} arbitrary well. Then, we extend to the case where f_{true} is a Lipschitz function. Finally, we note that this upper bound can be generalized to other kernels satisfying certain conditions.

We begin with the description of f_{true} and some further notations. Recall the definition of D, and without loss of generality assume $D = \mathbb{R}^p$. As mentioned above, we assume that f_{true} is a bounded step function. In particular, assume $f_{\text{true}} = a_i$ on D_i, where a_i is a constant, D_i is a measurable set in D, and $D = \bigcup D_i$. Let $a > 0$ be the upper bound of $|f_{\text{true}}|$. Next, for any $x \in D_i$, define the distance of x to other D_j; $j \ne i$ as $\psi_x = \min_{j \ne i} \text{dis}(x, D_j)$, where $\text{dis}(x, D_j) = \inf_{x' \in D_j} \|x - x'\|$ and $\|\cdot\|$ is the usual Euclidean norm in D. By this definition of ψ_x, one can verify that $B(x, \psi_x) \in D_i$ for all x, where $B(x, \psi_x)$ is the ball centered at x with the radius ψ_x. Note that $B(x, \psi_x)$ is well defined for all $x \in D$. Recall that σ is the kernel parameter of the Gaussian RKHS.

The next theorem gives an upper bound on $A(\infty)$.

Theorem 7 *Suppose f_{true} is a piecewise step function with $|f_{true}| \le a$ for some $a > 0$. Define ψ_x and $A(\infty)$ as above. One has*

$$A(\infty) \le \max(\tau, 1-\tau) E_{P_X}\left\{8a \exp\left(-\psi_x^2/(2p\sigma^2)\right)\right\}, \tag{7}$$

where P_X is the marginal distribution of X, p is the dimensionality of X and σ is the kernel parameter of the Gaussian RKHS.

For a fixed p, the upper bound in Theorem 7 depends on ψ_x, σ and the distribution P_X. If P_X is absolutely continuous with respect to the Lebesgue measure, one can verify that $A(\infty) \to 0$ with $\sigma \to 0$. This means that $f^{(\infty)}$ can approximate f_{true} arbitrarily well almost everywhere.

Next we consider the case where f_{true} is a general bounded measurable function. All bounded measurable functions can be approximated arbitrarily well by step functions. However, if f_{true} is too wiggly or is discontinuous at too many points in D, it cannot be well approximated by the Gaussian RKHS functions. For example, when f_{true} is discontinuous on a dense subset of D, $\psi_x = 0$ for all x if the step function is close enough to f_{true}. This leads to the right hand side of (7) being large. Therefore, we need some smoothness condition on f_{true}. The next corollary shows that when f_{true} is Lipschitz, $A(\infty) \to 0$ as $\sigma \to 0$.

Corollary 8 *Assume that f_{true} is a bounded Lipschitz function, and P_X is absolutely continuous with respect to the Lebesgue measure on D. Then $A(\infty) \to 0$ as $\sigma \to 0$.*

The discussions above have focused on the Gaussian RKHS. We note that it is possible to generalize the obtained results to more general RKHS's. For example, using similar techniques, one can verify that similar results as in Theorem 7 and Corollary 8 still hold if we consider many other radial kernels such as the Laplacian kernel. Other kernels, for example the polynomial kernel, may not have such guarantee that functions in the kernel space can approximate the underlying function arbitrary well. See the proof of Theorem 7 in the appendix for more discussions.

3.4 Finite Sample Error Bounds

The theory in Section 3.1 gives the asymptotic convergence rate of the estimation error. It is useful when the sample size is large. In this section, we derive some finite sample bounds on the prediction errors $E\rho_\tau(Y - \hat{f}_n)$ and $E\rho_\tau(Y - \tilde{f}_n)$, which can be used to assess the goodness of fit of the resulting model, when n is not large. In this section, we focus on the comparison between $E\rho_\tau(Y - \hat{f}_n)$ and $E\rho_\tau(Y - \tilde{f}_n)$. In particular, we show that the Rademacher complexity for $\hat{f}_n$ can be smaller compared to that of $\tilde{f}_n$. Hence, the prediction error bound for the data sparsity method can be better, and this demonstrates the usefulness of our proposed method. Note that Takeuchi et al. (2006) also used the Rademacher complexity to bound the estimation error. In this paper, we further consider how to bound the Rademacher complexity, especially for the data sparsity method.

To begin with, we introduce the following assumption.

Assumption A: The noise ε is bounded such that $|\varepsilon| \le t$ for some positive t.

Theorem 9 *Suppose Assumption A holds. Then the solution $\hat{f}_n$ to (5) satisfies that, with probability at least $1-\delta$ with a small and positive δ,*

$$E\rho_\tau(Y - \hat{f}_n) \le \frac{1}{n}\sum_{i=1}^{n} \rho_\tau\big(y_i - \hat{f}_n(x_i)\big) + Z_n + \max(\tau, 1-\tau)\mu,$$

where $Z_n = 3\max(\tau, 1-\tau)\Big(n^{-1}(2s^2 + 2t^2)\log(2/\delta)\Big)^{1/2}$ and

$$\mu = s\sqrt{\frac{2\log(2n+2)}{n}}.$$

Moreover, the solution $\tilde{f}_n$ to (6) satisfies that, with probability at least $1-\delta$,

$$E\rho_\tau(Y - \tilde{f}_n) \le \frac{1}{n}\sum_{i=1}^{n} \rho_\tau\big(y_i - \tilde{f}_n(x_i)\big) + Z_n + \max(\tau, 1-\tau)\mu,$$

where Z_n is defined as above and

$$\mu = 2sn^{-1/4} + \sqrt{n^{-1}\big(2^{11}n^{1/4} + 2\log(5) + 0.5\log(n)\big)}.$$

One can see from Theorem 9 that μ for $\hat{f}_n$ is much smaller than $\tilde{f}_n$. This is because the functional space of (5) is smaller than that of (4) (see Lemma 13 in the appendix). The key to the proof for $\tilde{f}_n$ is to control the covering number of the functional space in (4), as in Lemma 14. In particular, we study the covering number of a unit ball in $\mathcal{H}$, i.e., $\{f' : \|f'\|_{\mathcal{H}} \le 1\}$. On the other hand, the key to the proof for $\hat{f}_n$ is that the Rademacher complexity of the functional space of (5) is equivalent to the Rademacher complexity of a convex hull of functions with $2n+2$ vertices, and the latter enjoys a much better bound compared to the Rademacher complexity in (4).

From Theorem 9, one can conclude that when the underlying function can be well approximated by a function that has a sparse representation (in other words, the term $\frac{1}{n}\sum_{i=1}^{n} \rho_\tau\big(y_i - \hat{f}_n(x_i)\big)$ and s are both small), the prediction error bound for the data sparsity method can be better. This observation provides some insight on the usefulness of the data sparsity method, which is illustrated by our numerical examples in Section 5, and discussed in Section 2.

Notice that our theory is for a general RKHS with very weak assumptions. Thus, the first choice of μ, $\mu = 2sn^{-1/4} + \sqrt{n^{-1}\big(2^{11}n^{1/4} + 2\log(5) + 0.5\log(n)\big)}$, can be refined if one considers specific kernels with additional assumptions, such as the decay rate of the eigenvalues of the corresponding Hilbert-Schmidt integral operator. See, for example, Zhou (2002) and Steinwart and Scovel (2007). This can lead to a better bound in Theorem 9. For example, with Gaussian kernel learning (including intercepts), an application of the result in Zhou (2002) gives $\mu = 2sn^{-1/3} + 2^{p+2}p^{1/2}n^{-1/3}\log^{p/2}(n)$, where p is the dimensionality of X. Hence for a small p, our μ on $\tilde{f}_n$ in Theorem 9 can be loose in this case. Nevertheless, one can verify that the bound on the data sparsity method is still better than $\mu = 2sn^{-1/3} + 2^{p+2}p^{1/2}n^{-1/3}\log^{p/2}(n)$. If we use the Gaussian kernel without an intercept, then one can verify that we have $\mu = O_P(n^{-1/2})$ for $\tilde{f}_n$ (Mendelson, 2003). However, without the intercept, the empirical prediction error term might be large for some problems, which can lead to suboptimal results.

As a remark, we note that Assumption A ensures that the response variable y is bounded if we restrict our consideration to the space $\mathcal{F}_n(s)$. Because we want to bound a finite sample error, any large noise in the response data, that is, an observed y_i being significantly away from its expected value given the predictors, can result in the failure of our bound derived in Theorem 9. Assumption A excludes the possibility that this large noise happens. This assumption can be removed if one assumes that the tail of the distribution of $|\varepsilon|$ satisfies certain properties, and similar results can

be obtained by modifying the proof of Theorem 9 accordingly. Corollary 10 gives one possible generalization of the bound in Theorem 9, where we make an assumption of the distribution of the error ε. Note that the results in Theorem 9 and Corollary 10 can be calculated directly from the data and the tuning parameter s we use.

Corollary 10 *Suppose that the errors ε_i; $i = 1,\ldots,n$ follow a common distribution with a continuous cumulative distribution function Φ_ε. For simplicity, assume that the distribution of ε is symmetric with respect to 0. Then $\hat{f}_n$ and $\tilde{f}_n$ are controlled by the same finite sample bounds as in Theorem 9 except that $Z_n = 3\max(\tau,1-\tau)\Big(n^{-1}(2s^2+2t^2)\log(4/\delta)\Big)^{1/2}$ and $t = \Phi_\varepsilon^{-1}\big(0.5+0.5(1-\delta/2)^{1/n}\big)$.*

If ε follows a Gaussian distribution, one can verify that for a fixed δ, $t = O_P\big(\log(n)^{1/2}\big)$ as $n \to \infty$. Hence, t diverges in a slow rate. For other error distributions, one can obtain similar results by studying the corresponding Φ_ε, and we omit the details here.

4. Optimization and Tuning Procedure

The numerical optimization of (5) for fixed s and τ can be done by a simple linear programming. However, it is often desirable to have the entire solution path of $\hat{\alpha}$ and $\hat{b}$ with respect to s. For example, when we need to perform a comprehensive tuning procedure for choosing the optimal s, the solution path can significantly reduce the computational cost. In the literature of penalized quantile regression, Li et al. (2007) developed the solution path for (4) with respect to λ, Li and Zhu (2008) derived the solution path for L_1 penalized quantile regression with linear learning, and Rosset (2009) studied the solution surface with respect to both λ and τ in (4). In this section, we first briefly discuss how to derive the corresponding solution path with respect to s, then consider how to select the tuning parameter s.

Let $\tilde{K}$ be the n by $(n+1)$ matrix $(1\ K)$, where 1 is a vector of 1 of length n, and let $\bar{\alpha} = (b,\alpha_1,\ldots,\alpha_n)$. With b penalized, one can verify that the optimization (5) is equivalent to

$$\min_{\bar{\alpha}} \frac{1}{n}\sum_{i=1}^{n} \rho_\tau\big(y_i - f(x_i)\big), \text{ subject to } |\bar{\alpha}| \le s,$$

where $\big(f(x_1),\ldots,f(x_n)\big) = \tilde{K}\bar{\alpha}$. We note that the solution path of this optimization problem can be obtained in a similar manner as in Li and Zhu (2008) without an intercept in their notation, despite that they only considered linear learning problems. We omit the details here.

To illustrate the algorithm, using the data considered in Figure 2, we plot the piecewise linear solution path $\{b(s),\alpha(s)\}$ in Figure 5. Because the entire set $\{b(s),\alpha(s)\}$ consists of 31 piecewise linear functions, we only report a subset of $\{b(s),\alpha(s)\}$ in Figure 5 to make the plot clear. Moreover, if we plot the solution path on $[0,s_1]$ for large s_1, the lines become less clear on $[0,s_2]$ with $s_2 \ll s_1$. Hence, we only plot the solution path on $[0,20]$ for a demonstration.

Remark 11 *As mentioned in Section 2, penalizing $|b|$ helps to guarantee the uniqueness of the solution path. In particular, if the intercept is not regularized, when s is large (or equivalently, the model is mildly penalized), there exist cases where b is not uniquely determined. See Li et al. (2007) and Li and Zhu (2008) for detailed discussions on this issue.*

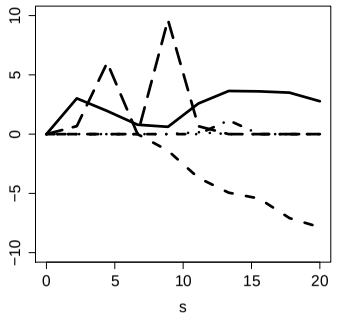


Figure 5: A subset of the solution path $\{b(s),\alpha(s)\}$ as a function of s. Notice that for brevity, we only plot five $\hat{\alpha}_i(s)$'s and the intercept b, and we restrict s such that $s \in [0,20]$ to illustrate the solution path. The solid line corresponds to $b(s)$.

Next, we briefly discuss how to select the optimal tuning parameter s in (5) for a given regression problem. Similar to many other penalized techniques, a proper choice of s is crucial in practice. In particular, s being too large can lead to an overfitted model, and s being too small can lead to an underfitted model. For either cases, the prediction accuracy can be low. Here we discuss two commonly used criteria for kernel quantile regression. The first criterion is the Schwarz Information Criterion (SIC, Schwarz, 1978; Koenker et al., 1994), which can be written as

$$\text{SIC}(s) = \log\left\{\frac{1}{n}\sum_{i=1}^{n}\rho_\tau\big(y_i - \hat{f}(x_i)\big)\right\} + \frac{\log(n)df}{n}.$$

Here df measures the dimensionality of the model, and $\frac{\log(n)df}{n}$ balances the model complexity and goodness-of-fit. The second criterion is the Generalized Approximate Cross-Validation criterion (GACV, Yuan, 2006), defined as

$$\text{GACV}(s) = \frac{1}{n-df}\left\{\frac{1}{n}\sum_{i=1}^{n}\rho_\tau\big(y_i - \hat{f}(x_i)\big)\right\}.$$

Note that GACV was originally proposed as a stable estimator of the generalized comparative Kullback-Leibler distance for the model.

To estimate df, it has been proposed to use the divergence (Nychka et al., 1995; Yuan, 2006)

$$\text{div}(\hat{f}) = \sum_{i=1}^{n}\frac{\partial\hat{f}(x_i)}{\partial y_i}.$$

Li et al. (2007) showed that for the kernel quantile regression with the regular squared norm penalty, $\mathrm{div}(\hat{f})$ coincides with the number of interpolated y_i's, and this makes the estimation of df convenient. The next proposition shows that for the proposed quantile regression with the data sparsity constraint, we have the same estimation formula, given the "one at a time" condition (Efron et al., 2004).

Proposition 12 *Assume the "one at a time" condition holds. For $(y_i;\ i=1,\ldots,n)\in\mathbb{R}^n$ except on a set of Lebesgue measure 0 and any fixed s, we have*

$$\sum_{i=1}^{n}\frac{\partial\hat{f}(x_i)}{\partial y_i}=|\mathcal{E}|,$$

where $\mathcal{E}$ is the set of interpolated y_i's.

5. Numerical Examples

In this section, we examine the performance of our proposed method with the data sparsity constraint using both simulated and real data sets. We demonstrate the effect of the SIC and GACV criteria. As a comparison, we also apply quantile regression using RKHS learning with the regular squared norm penalty, which was previously studied in Takeuchi et al. (2006) and Li et al. (2007).

5.1 Simulated Examples

We use three simulated examples to compare the performance of the proposed method with the data sparsity constraint and the standard method using the regular squared norm penalty. In particular, we study the three examples given in Section 2, which cover two cases. The first case (Examples 1 and 2) considers the situation when $f_0(x)$ can be well approximated by functions of the form $\sum_{j=1}^{m}\gamma_j K(x,z_j)$ for some fixed z_j and γ_j, where m is a small positive integer. For the second case (Example 3), f_0 is constructed to have many fluctuations. In this situation, f_0 cannot be well approximated by functions of the form $\sum_{j=1}^{m}\gamma_j K(x,z_j)$ with a small m.

For each example, we consider two choices of n: moderate ($n\in[30,50]$) and very large ($n=1000$). We show that for Examples 1-2 with moderate n's, the sparsely represented model from our method using the data sparsity constraint can have better prediction performance. On the other hand, for the case with a moderately large n but with the true function being quite wiggly (Example 3), or with a very large n, the performance of the two methods is comparable in terms of prediction accuracy. This is consistent with our theoretical insights.

We explore the performance under various settings of the noise. In particular, for the homoscedastic case, we let ε follow the standard normal distribution $N(0,1)$ (homoscedastic normal distribution, ho-norm. in Tables 2-7), and the t distribution with degrees of freedom 3 (t3). For the case when the noise is heteroscedastic, we let $\varepsilon(x)\sim U[-\|x\|_2/1.5,\|x\|_2/1.5]$ (heteroscedastic uniform distribution, unif. in Tables 2-7), and $\varepsilon(x)\sim N(0,\|x\|_2/3)$ (heteroscedastic normal distribution, he-norm. in Tables 2-7), where $\|\cdot\|_2$ is the usual Euclidian norm. Then we generate a training data set. Prediction models that correspond to different tuning parameters are built on the training data. We select the best tuning parameters by minimizing the SIC and GACV criteria on the training data respectively. The kernel parameters are also selected via the tuning procedure. We predict on a separate testing data set with the size 10000, to compare the performances of the two criteria. For

the choice of quantiles, since ε follows symmetric distributions with respect to 0, we choose $\tau=0.1$, 0.3 and 0.5.

As discussed in Section 3, in this section, we will demonstrate that empirically, the effect of whether to penalize the intercept b or not is not large. In particular, for all the examples, we fit the models of the two compared methods with or without $|b|$ penalized. Note that for the regular squared norm method, we include $|b|$ in the penalty as in (6).

To measure the goodness of fit of the model, we calculate the prediction error

$$\frac{1}{10000}\sum_{i\in\text{test set}}\rho_\tau(y_i-\hat{f}_n(x_i)),$$

and the L_1 norm of $\hat{f}_n-f_{\text{true}}$. This procedure is repeated 1000 times and we report the average prediction error, average L_1 norm and average model complexity in terms of df. For our data sparsity method, we also report the percentage of non-zero α_i's as a measurement of how sparse the model is. Note that this percentage for the regular squared norm method is always 100%.

The simulation results are reported in Tables 2-7. For brevity, we only report the results for certain settings listed in Table 1. The other results are omitted because the general pattern is similar. Notice that for the cases where we compare moderate to large n's, we penalize the intercept of the data sparsity method, but not the intercept of the regular squared norm method, because the numerical difference is small.

Example	τ	Comparison
Ex 1	$\tau=0.1$	Penalize b or not
	$\tau=0.3$	Moderate or large n
Ex 2	$\tau=0.3$	Penalize b or not
	$\tau=0.5$	Moderate or large n
Ex 3	$\tau=0.5$	Penalize b or not
	$\tau=0.1$	Moderate or large n

Table 1: Summary of the settings for the reported simulation results.

Example 1: We generate the data in the same way as in Figure 2. In particular, x is one dimensional and follows the uniform distribution between -6 and 6. The underlying $f_0(x)=10\exp(-x^2)$. We use $n=30$ for the training data. The Laplacian kernel is used.

Example 2: The data are generated in the same way as in Figure 3. In particular, we let x be uniformly distributed in $[-3,3]\times[-3,3]$. The underlying true model is given by $f_0(x)=10\exp(\|x-(-2,-2)^T\|_2^2)+5\exp(\|x-(-1,1)^T\|_2^2)+2\exp(\|x-(2,-1)^T\|_2^2)$. There are 50 observations in the training data set. We apply the Gaussian kernel for this example.

Example 3: The data are generated similarly as in Figure 4. To be specific, x is one dimensional, uniformly distributed in $[-7,7]$. We have the underlying function $f_0(x)=5\sin(2.5x)$. The training data consist of 30 observations, and the Laplacian kernel is employed.

We summarize our findings from the simulation results as follows.

- For Examples 1 and 2 with a moderate n, the data sparsity method tends to choose a simpler model than that of the square norm penalty, with either the GACV or the SIC criterion. Furthermore, the data sparsity constraint performs better than the regular squared norm penalty.

This implies that functions estimated by the squared norm penalty can have potential overfitting because of too many kernel functions, as indicated by Figures 2 and 3. For a large n, the prediction performance of the two methods is comparable, because asymptotically both methods can perform well.

- For Example 3 with a moderate n, the data sparsity method tends to choose a model with a large number of non-zero α_i's. In this case, the model from (5) is not sparse, and the two methods perform similarly.
- For a large n in all the examples, the data sparsity method can choose models with simpler representation than the regular method, while keeping similar prediction performance. Consequently, the data sparsity method yields a parsimonious model that has advantages in terms of computational efficiency and interpretability.
- The SIC always tends to choose a simpler model than the GACV criterion. In these examples, GACV overall works slightly better than SIC, for both the data sparsity constraint and the squared norm penalty.
- As τ gets closer to 0.5, the performances of the two penalties and the two criteria become better, in terms of the L_1 norm.

		$\|b\|$ penalized				$\|b\|$ not penalized			
		Squared norm		Data sparsity		Squared norm		Data sparsity	
	Dist.	GACV	SIC	GACV	SIC	GACV	SIC	GACV	SIC
Pred	ho-norm.	0.760	0.910	0.517	0.596	0.752	0.909	0.523	0.579
	t3	0.893	1.013	0.677	0.699	0.849	1.075	0.711	0.726
	unif.	0.869	0.995	0.636	0.701	0.933	0.959	0.598	0.700
	he-norm.	0.913	0.925	0.708	0.734	0.958	1.033	0.722	0.748
L_1 Norm	ho-norm.	2.061	2.291	0.733	0.759	2.062	2.294	0.715	0.753
	t3	2.233	2.426	1.239	1.297	2.236	2.429	1.244	1.280
	unif.	2.362	2.217	0.665	0.687	2.246	2.199	0.626	0.699
	he-norm.	2.343	2.366	1.290	1.375	2.109	2.256	1.276	1.391
df	ho-norm.	13.24	12.55	6.939	6.703	13.56	12.33	6.899	6.716
	t3	13.19	11.87	6.771	6.529	13.48	11.44	6.784	6.545
	unif.	13.28	11.13	7.044	6.512	13.64	11.23	7.033	6.529
	he-norm.	15.16	14.57	8.182	7.903	15.22	14.91	8.452	8.005
Percent.	ho-norm.	-	-	17.33	16.25	-	-	17.49	16.88
	t3	-	-	17.18	16.10	-	-	17.36	16.42
	unif.	-	-	17.58	16.69	-	-	17.42	17.01
	he-norm.	-	-	19.21	18.34	-	-	18.87	18.55

Table 2: Results of the simulation Example 1 with $\tau = 0.1$ and $|b|$ penalized (left) or not (right). The best mean prediction errors are 0.40 (ho-norm., homoscedastic normal distribution), 0.56 (t3), 0.59 (unif., heteroscedastic uniform distribution), and 0.60 (he-norm., heteroscedastic normal distribution). The standard errors of the prediction errors range from 0.0040 to 0.0053. The standard errors of the L_1 norms range from 0.0068 to 0.0095. The standard errors of df range from 0.050 to 0.066. The standard errors of the percentage of non-zero α_i's for the data sparsity method range from 0.032 to 0.051.

		$n = 30$				$n = 1000$			
		Squared norm		Data sparsity		Squared norm		Data sparsity	
	Dist.	GACV	SIC	GACV	SIC	GACV	SIC	GACV	SIC
Pred	ho-norm.	0.847	0.903	0.711	0.716	0.514	0.510	0.512	0.512
	t3	1.044	1.070	0.928	0.946	0.688	0.691	0.685	0.693
	unif.	0.914	0.916	0.833	0.820	0.655	0.667	0.659	0.671
	he-norm.	0.966	0.989	0.831	0.845	0.711	0.709	0.705	0.713
L_1 Norm	ho-norm.	1.462	1.841	0.517	0.552	0.657	0.644	0.659	0.650
	t3	1.533	1.897	0.583	0.694	1.024	1.058	1.062	0.989
	unif.	1.541	1.882	0.755	0.795	0.578	0.560	0.556	0.581
	he-norm.	1.515	1.798	0.689	0.726	0.913	0.926	0.912	0.904
df	ho-norm.	15.67	12.77	6.523	6.040	20.44	20.16	18.16	18.05
	t3	13.97	12.05	6.164	6.070	21.52	21.09	19.13	18.07
	unif.	15.85	13.71	6.775	6.433	19.89	19.15	18.29	18.14
	he-norm.	16.77	15.84	7.782	7.503	22.10	22.28	21.85	22.34
Percent.	ho-norm.	-	-	17.33	16.25	-	-	0.614	0.603
	t3	-	-	17.18	16.10	-	-	0.610	0.606
	unif.	-	-	17.44	16.96	-	-	0.612	0.606
	he-norm.	-	-	18.92	17.77	-	-	0.693	0.687

Table 3: Results of the simulation Example 1 with $\tau = 0.3$ and moderate n (left) or large (right). The best mean prediction errors are 0.40 (ho-norm., homoscedastic normal distribution), 0.56 (t3), 0.59 (unif., heteroscedastic uniform distribution), and 0.60 (he-norm., heteroscedastic normal distribution). The standard errors of the prediction errors range from 0.0030 to 0.0046. The standard errors of the L_1 norms range from 0.0055 to 0.0079. The standard errors of df range from 0.048 to 0.059. The standard errors of the percentage of non-zero α_i's for the data sparsity method range from 0.031 to 0.055.

5.2 Real Data Analysis

In this section, we apply our proposed method (5) to several real data sets. In particular, we consider 20 data sets studied in Section 5 of Takeuchi et al. (2006), and the well known annual salary of baseball players data studied in He et al. (1998), Yuan (2006) and Li et al. (2007). The description and a summary table of the first 20 data sets can be found in Takeuchi et al. (2006), and we do not repeat it here. For the baseball data, it consists of statistics for 263 North American major league baseball players in the year 1986. The original data set has 22 predictors and the players' 1987 annual salary as the response, and we use the whole data for our analysis reported in Tables 8-10. Furthermore, following He et al. (1998), Yuan (2006) and Li et al. (2007), we use two representative predictors from the baseball data to perform an illustrative analysis, which is reported in Figure 7. In particular, we measure players' performance by the number of home runs in the latest year, and measure player's seniority by the number of years played.

For all real data analysis, the Gaussian kernel is used. For the results reported in Tables 8-10, we first standardize the predictors and response to make the results comparable. We split each data set into 10 parts of roughly the same size. We choose 1 part as the testing data set, and the remaining as the training data. Then we select the best tuning parameter and kernel parameter on the training data, and predict on the testing data. We continue to the next random split once all the 10 parts have served as the testing data. This random split is repeated 100 times for each data set, and we

		$\|b\|$ penalized				$\|b\|$ not penalized			
		Squared norm		Data sparsity		Squared norm		Data sparsity	
	Dist.	GACV	SIC	GACV	SIC	GACV	SIC	GACV	SIC
Pred	ho-norm.	0.885	1.075	0.527	0.673	0.847	1.031	0.557	0.624
	t3	1.068	1.118	0.677	0.894	1.124	1.341	0.710	0.909
	unif.	1.057	1.243	0.644	0.692	1.147	1.286	0.613	0.629
	he-norm.	1.021	1.226	0.697	0.801	1.089	1.146	0.703	0.811
L_1 Norm	ho-norm.	1.826	2.078	0.760	0.911	1.864	2.271	0.796	0.894
	t3	1.873	2.213	0.875	1.244	1.905	2.400	0.892	1.303
	unif.	2.002	2.199	0.841	1.199	1.953	2.168	0.872	1.240
	he-norm.	2.112	2.351	0.873	0.992	1.913	2.325	0.846	1.023
df	ho-norm.	18.16	16.22	12.33	9.885	18.83	16.06	12.51	10.21
	t3	17.91	16.22	11.97	8.678	18.05	15.80	12.03	9.146
	unif.	18.44	15.78	12.91	9.913	18.31	16.03	12.50	10.12
	he-norm.	19.05	18.79	14.41	12.90	19.33	19.03	14.29	12.46
Percent.	ho-norm.	-	-	13.24	12.90	-	-	13.44	13.00
	t3	-	-	14.26	12.92	-	-	14.55	13.22
	unif.	-	-	16.64	15.68	-	-	17.02	15.94
	he-norm.	-	-	17.22	16.78	-	-	16.89	16.59

Table 4: Results of the simulation Example 2 with $\tau = 0.3$ and $|b|$ penalized (left) or not (right). The best mean prediction errors are 0.40 (ho-norm., homoscedastic normal distribution), 0.56 (t3), 0.59 (unif., heteroscedastic uniform distribution), and 0.60 (he-norm., heteroscedastic normal distribution). The standard errors of the prediction errors range from 0.0055 to 0.0068. The standard errors of the L_1 norms range from 0.0071 to 0.0133. The standard errors of df range from 0.066 to 0.072. The standard errors of the percentage of non-zero α_i's for the data sparsity method range from 0.044 to 0.057.

report the average prediction error (Pred) and its sample standard deviation (SSD) in Tables 8-10 for $\tau = 0.1, 0.5, 0.9$. We only report the results where the intercept is penalized for the data sparsity method, but not for the standard method with the squared norm penalty. Similar to the results in Section 5.1, the numerical difference of whether the intercept is penalized or not is not large. For the results reported in Figure 7, we train the model using the entire data set. We then plot the predicted values against the two dimensional input space.

Similar to Takeuchi et al. (2006), we perform a two-sided paired-sample t-test to compare the prediction performance of the two methods. In Tables 8-10, we can see that for the caution, sniffer, GAGurine, topo, CobarOre, and baseball data sets, the performance of the data sparsity method is overall better than that of the squared norm penalty method. For birthwt, engel, gilgais, and mcycle, the data sparsity method is slightly better. For BostonHousing and cpus, the squared norm penalty is slightly better. For the other data sets, their performance is comparable. This demonstrates the usefulness of the data sparsity method. Moreover, we plot the fitted functions $\hat{f}_n$ and $\tilde{f}_n$ with $\tau = 0.5$ for the mcycle data on the left panel of Figure 6. Compared to Figure 3 in Takeuchi et al. (2006), one can see that the data sparsity model has less wiggles, and yields a more interpretable result. We also plot the fitted functions with $\tau = 0.1$ on the right panel of Figure 6. In this case, one can see that $\tilde{f}_n$ is quite wiggly compared to $\hat{f}_n$.

For the results on the illustrative analysis of the baseball data, from the right panels of Figure 7 (our data sparsity constraint), we can see that for all players, the income increases with their perfor-

		$n = 30$				$n = 1000$			
		Squared norm		Data sparsity		Squared norm		Data sparsity	
	Dist.	GACV	SIC	GACV	SIC	GACV	SIC	GACV	SIC
Pred	ho-norm.	0.922	0.984	0.546	0.683	0.487	0.466	0.482	0.478
	t3	1.243	1.309	0.798	0.913	0.634	0.633	0.612	0.629
	unif.	1.155	1.272	0.877	0.916	0.637	0.640	0.642	0.638
	he-norm.	1.133	1.287	0.764	0.787	0.688	0.685	0.681	0.684
L_1 Norm	ho-norm.	1.416	1.813	0.653	0.772	0.714	0.720	0.711	0.715
	t3	1.679	2.002	0.718	1.060	0.922	0.918	0.918	0.927
	unif.	1.744	2.102	0.899	1.132	0.559	0.576	0.565	0.563
	he-norm.	1.553	1.842	0.957	1.086	1.010	1.004	0.989	0.995
df	ho-norm.	18.40	15.66	12.17	10.11	16.14	16.01	15.98	15.79
	t3	18.54	16.11	11.10	9.264	16.48	16.22	16.14	15.88
	unif.	18.25	16.94	11.46	10.72	15.66	15.41	15.36	15.20
	he-norm.	19.91	17.27	14.62	13.78	18.89	18.43	17.67	17.40
Percent.	ho-norm.	-	-	12.28	11.90	-	-	0.591	0.582
	t3	-	-	13.42	13.00	-	-	0.611	0.590
	unif.	-	-	16.54	14.30	-	-	0.622	0.605
	he-norm.	-	-	16.88	15.53	-	-	0.630	0.616

Table 5: Results of the simulation Example 2 with $\tau = 0.5$ and moderate n (left) or large (right). The best mean prediction errors are 0.40 (ho-norm., homoscedastic normal distribution), 0.56 (t3), 0.59 (unif., heteroscedastic uniform distribution), and 0.60 (he-norm., heteroscedastic normal distribution). The standard errors of the prediction errors range from 0.0052 to 0.0068. The standard errors of the L_1 norms range from 0.0083 to 0.0114. The standard errors of df range from 0.044 to 0.078. The standard errors of the percentage of non-zero α_i's for the data sparsity method range from 0.065 to 0.097.

mances. On the other hand, the salary does not necessarily increase with the seniority. For many players, especially the high income ones ($\tau = 0.75$), the salary increases with the seniority until a golden age, then it decreases. This is consistent with our intuition. For the results with the squared norm penalty (the left panels), we can see the same trend. However, for the very senior players, because the estimated salary function has fluctuations from kernel functions, the salary decreases if their performances increase from 20 to 30. This is against our intuition. Therefore, in this data set, our data sparsity constraint performs well and gives a good interpretation of the data.

6. Discussion

In this paper, we study the learning problem in a RKHS. In particular, we propose a data sparsity constraint that can achieve a parsimonious representation of the resulting learning function. Using quantile regression as an example, we numerically show that when the underlying function can be well approximated by functions that have a sparse representation in the corresponding RKHS and n is not large, the data sparsity method can perform better than the regular squared norm penalty method. For other cases, such as when the true function is relatively difficult to be approximated by

		$\|b\|$ penalized				$\|b\|$ not penalized			
		Squared norm		Data sparsity		Squared norm		Data sparsity	
	Dist.	GACV	SIC	GACV	SIC	GACV	SIC	GACV	SIC
Pred	ho-norm.	1.770	1.812	1.749	1.796	1.742	1.814	1.759	1.803
	t3	2.119	2.230	2.138	2.206	2,178	2.267	2.104	2.257
	unif.	1.997	1.989	1.846	2.014	1.884	1.892	1.865	1.911
	he-norm.	1.849	1.897	1.833	1.878	1.845	1.904	1.890	1.926
L_1 Norm	ho-norm.	1.657	1.690	1.664	1.711	1.649	1.698	1.625	1.709
	t3	2.058	2.224	2.060	2.215	2.121	2.269	2.150	2.298
	unif.	1.887	1.845	1.869	1.893	1.853	1.829	1.850	1.837
	he-norm.	1.848	1.820	1.851	1.827	1.857	1.841	1.866	1.836
df	ho-norm.	24.66	24.03	13.76	13.09	24.48	23.19	14.90	13.55
	t3	25.53	24.69	13.96	13.14	25.56	23.92	14.14	14.00
	unif.	25.12	23.55	15.47	14.16	25.09	24.61	14.98	13.58
	he-norm.	25.79	24.15	14.16	13.90	25.12	24.07	13.92	13.23
Percent.	ho-norm.	-	-	66.24	61.32	-	-	68.45	65.37
	t3	-	-	70.18	66.26	-	-	68.91	65.28
	unif.	-	-	68.93	65.21	-	-	70.52	66.42
	he-norm.	-	-	71.29	67.84	-	-	68.35	66.91

Table 6: Results of the simulation Example 3 with $\tau = 0.5$ and $|b|$ penalized (left) or not (right). The best mean prediction errors are 0.40 (ho-norm., homoscedastic normal distribution), 0.56 (t3), 0.59 (unif., heteroscedastic uniform distribution), and 0.60 (he-norm., heteroscedastic normal distribution). The standard errors of the prediction errors range from 0.0091 to 0.0143. The standard errors of the L_1 norms range from 0.0128 to 0.0175. The standard errors of df range from 0.088 to 0.144. The standard errors of the percentage of non-zero α_i's for the data sparsity method range from 0.244 to 0.351.

functions in the RKHS, or when n is large, the data sparsity method can have comparable performance as the regular method. Therefore, the data sparsity method can be regarded as an alternative penalization method to solve learning problems with RKHS learning. Moreover, because of the sparsity in the kernel representation, the prediction for new data sets can be computationally faster. Through theoretical comparisons, we demonstrate that the data sparsity method can achieve the same convergence rate of the estimation error, compared with the squared norm penalty method. Furthermore, we show that for certain cases, the data sparsity method can enjoy a smaller bound on the finite sample prediction error. This helps to shed some light on the usefulness of the data sparsity constraint. We also discuss how to obtain a solution path with respect to the tuning parameter s.

We would like to point out several open problems for the theory developed in Section 3. The technique used to prove Theorems 1 and 9 there does not take into account the fact that the "active functional space" of the estimated function is often smaller than the entire $\mathcal{F}_n(s)$. Therefore, one possible way to obtain better results is to consider the "localized" covering number of the active functional space of (5). See the discussion of localization idea in, for example, Bartlett et al. (2005). In that case, we can expect a faster convergence rate and a tighter bound on the prediction error. Another open problem is to consider a combination of the L_2 and L_1 penalties, which can be a more general form than the pure L_1 or L_2 penalty. In the literature of linear learning, Zou and Hastie (2005) proposed the elastic net penalty as a convex combination of the L_2 and L_1 penalties. In

		$n = 30$				$n = 1000$			
		Squared norm		Data sparsity		Squared norm		Data sparsity	
	Dist.	GACV	SIC	GACV	SIC	GACV	SIC	GACV	SIC
Pred	ho-norm.	1.569	1.658	1.554	1.772	0.745	0.761	0.746	0.753
	t3	2.044	2.168	2.015	2.247	0.810	0.813	0.825	0.839
	unif.	1.717	1.826	1.679	1.774	0.876	0.891	0.859	0.880
	he-norm.	1.946	2.091	1.925	2.083	0.845	0.886	0.839	0.891
L_1 Norm	ho-norm.	1.746	1.766	1.753	1.760	0.924	0.995	0.957	0.986
	t3	2.193	2.324	2.301	2.332	1.103	1.166	1.098	1.156
	unif.	1.986	2.009	2.054	2.013	0.883	0.916	0.849	0.872
	he-norm.	2.094	2.129	1.954	2.058	1.124	1.196	1.049	1.087
df	ho-norm.	25.65	24.88	15.52	14.49	33.14	33.09	32.45	32.16
	t3	25.97	25.12	15.66	14.79	30.26	30.28	31.06	30.73
	unif.	26.36	25.77	16.28	16.11	32.28	32.06	32.44	31.98
	he-norm.	27.48	27.01	16.94	16.59	33.04	32.85	33.30	32.19
Percent.	ho-norm.	-	-	50.53	46.68	-	-	4.221	4.057
	t3	-	-	55.16	53.25	-	-	4.528	4.247
	unif.	-	-	55.23	54.71	-	-	4.567	4.059
	he-norm.	-	-	56.14	54.90	-	-	4.778	4.670

Table 7: Results of the simulation Example 3 with $\tau = 0.1$ and moderate n (left) or large (right). The best mean prediction errors are 0.40 (ho-norm., homoscedastic normal distribution), 0.56 (t3), 0.59 (unif., heteroscedastic uniform distribution), and 0.60 (he-norm., heteroscedastic normal distribution). The standard errors of the prediction errors range from 0.0107 to 0.0177. The standard errors of the L_1 norms range from 0.0109 to 0.0186. The standard errors of df range from 0.099 to 0.158. The standard errors of the percentage of non-zero α_i's for the data sparsity method range from 0.414 to 0.572.

kernel learning, how to perform such a generalization effectively can be an interesting problem to pursue.

Acknowledgments

The authors would like to thank the Action Editor Professor Saharon Rosset and three reviewers for their constructive comments and suggestions, which led to substantial improvements of the presentation of this paper. The authors are supported in part by National Science and Engineering Research Council of Canada (NSERC), National Natural Science Foundation of China (NSFC 61472475), NSF grant DMS-1407241, NSF DMS-1055210, NIH/NCI grant R01 CA-149569, and NIH/NCI P01 CA-142538.

Appendix A. Proof of Theorem 1

In the following proofs, when the technique can be applied to both the proposed data sparsity constraint and the regular squared norm penalty, we omit the difference between $\hat{f}_n$ and $\tilde{f}_n$ for brevity.

Before giving the proof of Theorem 1, we first introduce a lemma.

	Square norm				Data sparsity			
	GACV		SIC		GACV		SIC	
Data	Pred	SSD	Pred	SSD	Pred	SSD	Pred	SSD
baseball	10.09	0.67	10.14	0.52	**9.814**	0.43	10.33	0.49
BigMac2003	6.442	0.64	6.414	0.52	**6.297**	0.57	6.371	0.77
birthwt	18.44	0.84	18.80	0.72	**18.38**	0.69	18.92	0.74
BostonHousing	**5.543**	0.35	5.569	0.29	5.677	0.52	5.712	0.39
*caution	9.782	0.74	9.891	0.56	**8.433**	0.61	8.598	0.59
*CobarOre	15.74	1.16	16.12	1.28	12.19	0.95	**12.10**	0.84
*cpus	4.692	0.16	**4.575**	0.20	5.132	0.13	5.242	0.17
crabs	3.952	0.05	4.127	0.12	**3.893**	0.09	4.016	0.10
engel	5.490	0.42	**5.336**	0.39	5.356	0.51	5.561	0.35
ftcollinssnow	15.99	3.36	16.43	3.28	**15.62**	3.03	15.88	3.00
GAGurine	8.224	0.32	**8.138**	0.24	8.261	0.35	8.210	0.38
geyser	**8.440**	0.49	9.158	0.58	8.923	0.49	8.682	0.65
gilgais	6.731	0.32	6.849	0.29	**6.680**	0.29	7.003	0.33
heights	14.91	0.38	15.56	0.35	**14.69**	0.37	15.27	0.41
highway	8.964	0.57	9.059	0.63	9.112	0.71	**8.877**	0.56
*mcycle	7.732	0.25	8.105	0.31	**7.012**	0.21	7.033	0.29
*sniffer	6.372	0.33	6.457	0.30	**5.416**	0.28	5.608	0.28
snowgeese	5.788	1.01	5.892	0.84	**5.694**	0.70	6.010	0.69
topo	6.514	0.48	6.449	0.42	**6.268**	0.34	6.435	0.35
ufc	**10.11**	1.10	11.06	0.64	10.49	0.89	10.83	0.93
UN3	12.29	1.11	12.04	1.24	**11.92**	0.92	12.09	1.05

Table 8: Results of the real data analysis for $\tau = 0.1$. We reported $100 \times$ prediction error for Pred. Here $*$ means the difference of prediction error between the two methods, both GACV vs. GACV and SIC vs. SIC, is statistically significant at level 0.05 using two-sided paired-sample t-test.

Lemma 13 *Suppose the RKHS is separable and* $\sup_{X_1,X_2} K(X_1,X_2) = 1$. *Then* $\sum_{i=1}^n |\alpha_i| \le s$ *implies* $\|f'\|_{\mathcal{H}}^2 \le s^2$.

Lemma 13 indicates that a bound on $\sum_{i=1}^n |\alpha_i|$ is a stronger constraint than the usual squared norm constraint. Hence the effect of our data sparsity penalty is two fold: control the complexity of f and impose a soft threshold to gain data sparsity. Lemma 13 helps to bound the covering number of the functional class in Lemma 14.

Proof of Lemma 13: For any $f'(x) = \sum_{i=1}^n \alpha_i K(x_i,x)$ with $\sum_{i=1}^n |\alpha_i| \le s$, we have that $\|f'\|_{\mathcal{H}}^2 = \alpha^T K \alpha = \sum_{i=1}^n \sum_{j=1}^n K(x_i,x_j)\alpha_i\alpha_j \le \sum_{i=1}^n |\alpha_i| \cdot \sum_{j=1}^n |\alpha_j| \le s^2$, because $K(\cdot,\cdot) \le 1$. □

To prove Theorem 1, note that $\rho_\tau(y-f) \le |y-f|$. Hence in the following arguments, we can consider the loss $L(a,b) = |a-b|$ instead of the check function. Recall that the definition of $f_n^{(s)}$ is $f_n^{(s)} = \text{argmin}_{f \in \mathcal{F}_n(s)} L(Y,f)$. Define $g_f(\cdot) = (2s)^{-1}(L(\cdot,f) - L(\cdot,f_n^{(s)}))$, and $\mathcal{G} = \{g_f : f \in \mathcal{F}_n(s)\}$.

First we provide a lemma that controls the complexity of $\mathcal{G}$ in terms of its covering number. Notice that this technique can also be applied to the regular squared norm method, therefore, the result in Theorem 1 is also valid for the regular method that penalizes $|b|$ (or make additional assumptions to avoid this penalty on $|b|$). Before that we introduce some further notation. Let T_X be the empirical measure of a training set $((x_1,y_1),\ldots,(x_n,y_n))$, and the L_2 norm be defined as

	Square norm				Data sparsity			
	GACV		SIC		GACV		SIC	
Data	Pred	SSD	Pred	SSD	Pred	SSD	Pred	SSD
*baseball	24.47	0.87	25.61	0.78	22.16	0.69	**21.58**	0.70
BigMac2003	18.81	2.24	19.00	2.51	**18.42**	1.86	19.13	1.72
*birthwt	36.44	1.31	37.68	1.09	**33.20**	1.21	33.97	1.10
*BostonHousing	**10.92**	0.55	11.43	0.59	13.10	0.47	13.55	0.61
*caution	23.17	1.23	23.20	1.08	20.19	0.99	**20.01**	1.04
*CobarOre	41.26	1.84	40.07	1.45	**35.98**	1.76	36.62	1.64
cpus	**3.128**	0.23	3.269	0.30	3.202	0.22	3.294	0.25
crabs	**5.133**	0.24	5.249	0.22	5.227	0.31	5.158	0.28
engel	**14.18**	0.82	14.76	0.86	14.25	0.68	14.40	0.91
ftcollinssnow	40.58	5.14	41.89	5.46	41.43	4.91	**40.29**	3.88
*GAGurine	14.98	0.56	15.25	0.45	**13.07**	0.43	13.56	0.42
geyser	**29.16**	1.57	32.45	1.29	30.08	1.06	31.66	1.11
*gilgais	13.15	0.51	13.59	0.44	**11.02**	0.47	11.27	0.39
heights	36.72	1.13	**35.48**	0.78	36.14	0.81	35.55	0.92
highway	27.21	2.17	27.49	2.24	**26.79**	1.82	27.34	1.70
mcycle	18.24	0.77	19.17	0.61	**17.53**	0.71	18.32	0.64
sniffer	**10.17**	0.67	10.35	0.71	10.68	0.55	10.51	0.60
snowgeese	17.76	1.91	18.14	1.54	**17.42**	1.64	18.08	1.73
*topo	15.28	0.55	16.01	0.62	14.01	0.43	**13.76**	0.50
ufc	22.78	1.27	**21.75**	1.01	23.04	1.21	22.11	1.24
UN3	21.71	1.55	22.94	1.42	**21.25**	1.62	22.18	1.39

Table 9: Results of the real data analysis for $\tau = 0.5$. We reported $100 \times$ prediction error for Pred. Here $*$ means the difference of prediction error between the two methods, both GACV vs. GACV and SIC vs. SIC, is statistically significant at level 0.05 using two-sided paired-sample t-test.

$\|f\|_{L_2(T_X)} = \left(\frac{1}{n}\sum_{i=1}^n |f(x_i,y_i)|^2\right)^{1/2}$. For any $\eta > 0$, define $\mathcal{M}$ to be a η-net of a class of function $\mathcal{F}$ if, for any $f \in \mathcal{F}$, there exists $m \in \mathcal{M}$ such that $\|m-f\|_{L_2(T_X)} \le \eta$. Now let the $L_2(T_X)$ covering number $N(\eta,\mathcal{F},L_2(T_X))$ be the minimal size of all possible η-nets.

Lemma 14 *For* $\eta > 0$ *small enough and* $C_0 = 2^{10}$, *we have that*

$$\sup_{T_X} N\big(\eta,\mathcal{G},L_2(T_X)\big) \le \frac{5\exp(C_0\eta^{-2})}{\eta}.$$

Proof of Lemma 14: The proof consists of two steps. The first step is to bound the entropy number when there is no intercept in the regression function. The second step is to add the intercept into consideration, and bound the corresponding entropy based on the results obtained in the first step.

Here we focus on the covering number of $\mathcal{G}_{\mathcal{H},b} := \{(2s)^{-1}L(\cdot,f) : f \in \mathcal{F}_n(s)\}$, because $\mathcal{G}_{\mathcal{H},b}$ has the same covering number as $\mathcal{G}$. To that end, we first calculate the covering number of $\mathcal{G}_{\mathcal{H}} := \{(2s)^{-1}L(\cdot,f') : f' = \sum_{i=1}^n \alpha_i K(x_i,\cdot), \sum_{i=1}^n |\alpha_i| \le s\}$. Define

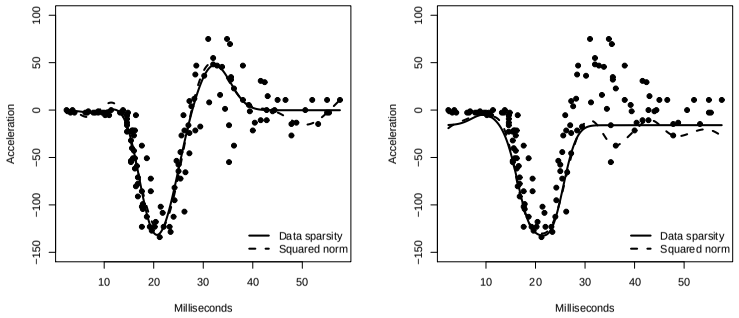
$$\mathcal{G}'_{\mathcal{H}} := \{(2s)^{-1}f' : f' = \sum_{i=1}^n \alpha_i K(x_i,\cdot), \sum_{i=1}^n |\alpha_i| \le s\}.$$

	Square norm				Data sparsity			
	GACV		SIC		GACV		SIC	
Data	Pred	SSD	Pred	SSD	Pred	SSD	Pred	SSD
*baseball	19.61	0.77	19.22	0.64	**15.52**	0.69	15.99	0.59
BigMac2003	11.33	0.92	11.84	0.74	10.92	0.81	**10.44**	0.99
birthwt	16.44	0.90	**15.89**	0.74	16.27	0.72	16.03	0.65
BostonHousing	**7.775**	0.36	8.104	0.40	8.158	0.34	7.960	0.39
*caution	16.76	0.56	16.33	0.49	**12.51**	0.51	13.09	0.62
*CobarOre	14.43	0.88	15.16	0.82	**12.55**	0.75	13.08	0.80
cpus	1.551	0.28	**1.479**	0.24	1.492	0.20	1.505	0.20
crabs	2.461	0.18	**2.447**	0.20	2.569	0.15	2.450	0.14
*engel	6.168	0.42	6.284	0.35	**5.041**	0.38	5.297	0.40
ftcollinssnow	20.18	4.14	22.46	4.51	**19.59**	3.97	22.13	4.22
*GAGurine	10.01	0.42	10.57	0.44	**8.558**	0.53	9.101	0.46
geyser	**12.63**	0.66	13.10	0.71	12.76	0.59	12.91	0.73
gilgais	6.167	0.40	6.228	0.29	**6.109**	0.31	6.334	0.34
heights	15.21	0.58	15.42	0.64	15.63	0.69	**15.02**	0.56
highway	16.96	1.42	16.27	1.59	**15.88**	1.49	16.59	1.34
mcycle	7.162	0.62	7.246	0.41	**6.197**	0.53	6.331	0.49
*sniffer	5.691	0.31	5.743	0.39	**4.409**	0.24	4.553	0.28
snowgeese	8.166	0.74	8.259	0.88	**7.919**	0.80	8.260	0.92
*topo	11.57	0.40	12.18	0.35	**10.51**	0.40	10.86	0.38
ufc	10.52	0.82	11.77	0.62	**10.24**	0.77	10.31	0.65
UN3	**7.774**	0.87	8.126	1.00	7.910	0.94	7.885	0.83

Table 10: Results of the real data analysis for $\tau = 0.9$. We reported $100\times$ prediction error for Pred. Here $*$ means the difference of prediction error between the two methods, both GACV vs. GACV and SIC vs. SIC, is statistically significant at level 0.05 using two-sided paired-sample t-test.

Define T'_X to be the empirical measure of the set $(x_1,\ldots,x_n)$. For any $g_1 := (2s)^{-1}L(\cdot,f'_1) \in \mathcal{G}_{\mathcal{H}}$, $g_2 := (2s)^{-1}L(\cdot,f'_2) \in \mathcal{G}_{\mathcal{H}}$, $|g_1-g_2| \le (2s)^{-1}|f'_1-f'_2|$. Hence, an $L_2(T'_X)$ net on $\mathcal{G}'_{\mathcal{H}}$ naturally introduces an $L_2(T_X)$ net on $\mathcal{G}_{\mathcal{H}}$, and furthermore the $L_2(T_X)$ covering number of $\mathcal{G}_{\mathcal{H}}$ is upper bounded by the $L_2(T'_X)$ covering number of $\mathcal{G}'_{\mathcal{H}}$. Moreover, by Lemma 13, $\|(2s)^{-1}f'\|_{\mathcal{H}} \le 1$. Thus, $\mathcal{G}'_{\mathcal{H}} \subset B_{\mathcal{H}}$, where $B_{\mathcal{H}}$ is the unit ball in $\mathcal{H}$. Hence, we only need to bound $N(\eta,B_{\mathcal{H}},L_2(T'_X))$. This can be done by a similar argument as in Theorem 2.1 of Steinwart and Scovel (2007). In particular, from analogous arguments as those that lead to (21) in Steinwart and Scovel (2007), we have that $\sup_{T'_X} N(\eta,B_{\mathcal{H}},L_2(T'_X)) \le \exp(\frac{C_0\eta^{-2}}{4})$, where one can choose $C_0 = 2^{10}$ (Carl and Stephani, 1990). In one words, we have that $\sup_{T_X} N(\eta,\mathcal{G}_{\mathcal{H}},L_2(T_X)) \le \sup_{T'_X} N(\eta,\mathcal{G}'_{\mathcal{H}},L_2(T'_X)) \le \sup_{T'_X} N(\eta,B_{\mathcal{H}},L_2(T'_X)) \le \exp(\frac{C_0\eta^{-2}}{4})$. Note that Theorem 2.1 of Steinwart and Scovel (2007) considered only the Gaussian RKHS, however the proof of the entropy bound for $p=2$ in their notation only requires that the RKHS is separable.

We proceed to bound the entropy number of $\mathcal{G}_{\mathcal{H},b}$. Define $\mathcal{G}'_{\mathcal{H},b} = \{(2s)^{-1}f : f \in \mathcal{F}_n(s)\}$. By similar arguments as above, $N(\eta,\mathcal{G}_{\mathcal{H},b},L_2(T_X)) \le N(\eta,\mathcal{G}'_{\mathcal{H},b},L_2(T'_X))$. Suppose $\mathcal{G}''$ is a minimal $\frac{\eta}{2}$-net of $\mathcal{G}'_{\mathcal{H}}$. One can verify that the union $\bigcup_{i=-S}^{S}\{\mathcal{G}''+is\frac{\eta}{2}\}$ is an η-net of $\mathcal{G}'_{\mathcal{H},b}$, where S is the smallest integer that is larger than $\frac{2}{\eta}$. To see this, let $g_1^b = (2s)^{-1}(f'_1+b_1)$ be an arbitrary point



(a) The fitted regression functions with $\tau = 0.5$ in the mcycle data.

(b) The fitted regression functions with $\tau = 0.1$ in the mcycle data.

Figure 6: The estimated functions for the mcycle data with $\tau = 0.5$ and $\tau = 0.1$. The dashed lines correspond to $\tilde{f}_n$ using the squared norm penalty method, and the solid lines correspond to $\hat{f}_n$ using the data sparsity method. Note that Takeuchi et al. (2006) also plotted the estimator for the squared norm penalty in their Figure 3. Compared to the dashed lines, the solid lines on both panels have less fluctuations especially when the predictor value is large, and are more interpretable.

in $\mathcal{G}'_{\mathcal{H},b}$, and $g_2^b = (2s)^{-1}(f'_2+b_2)$, with $(2s)^{-1}f'_2$ being the corresponding point in $\mathcal{G}''$ that has an $L_2(T'_X)$ distance to $(2s)^{-1}f'_1$ smaller than $\frac{\eta}{2}$, and $b_2 = is\frac{\eta}{2}$ for some integer $i \in [-S,S]$, such that $|b_2-b_1| \le s\frac{\eta}{2}$. Now the $L_2(T'_X)$ distance between g_1^b and g_2^b is

$$\begin{aligned}
&\Big(\int (g_1^b-g_2^b)^2\Big)^{1/2}\\
&\le (2s)^{-1}\Big(\int (2(f'_1-f'_2)^2+2(b_1-b_2)^2)^2\Big)^{1/2}\\
&\le \eta,
\end{aligned}$$

where the integral is taken with respect to the counting measure on T'_X. Therefore, the covering number of $\mathcal{G}'_{\mathcal{H},b}$ is less than $(\frac{4}{\eta}+2)\exp(C_0\eta^{-2})$. Consequently, the covering number of $\mathcal{G}_{\mathcal{H},b}$ is upper bounded by $(\frac{4}{\eta}+2)\exp(C_0\eta^{-2})$. The desired result follows when η is small enough. □

Proof of Theorem 1: The outline of the proof is as follows. First, we define $M = \sqrt{2}n^{-1/2}\log(n)$. Notice the difference between M (a number) and $\mathcal{M}$ (a functional space used for the definition of the entropy number introduced just before Lemma 14). Then we bound the probability $P(e(\hat{f}_n,f^{(\infty)}) \ge 8sM+d_{n,s})$. In particular, we show that $P(e(\hat{f}_n,f^{(\infty)}) \ge 8sM+d_{n,s}) \le 6(1-\frac{1}{16nM^2})^{-1}\exp(-nM^2)$, then apply the Borel-Cantelli Lemma to obtain the result.

For $M=\sqrt{2}n^{-1/2}\log(n)$, we first verify that for a large n, this M satisfies

$$(\log_2 \frac{16\sqrt{6}\eta_{n,0}}{M}+1)^2(\frac{256C_0}{n}) \le \frac{M^2}{256}, \tag{8}$$

where $\eta_{n,0}>0$ is chosen to satisfy

$$\frac{C_0}{\eta_{n,0}^2}+\log\frac{5}{\eta_{n,0}}=\frac{1}{4}nM^2, \tag{9}$$

and $C_0=2^{10}$ is a constant as in Lemma 14. From (9), one can verify that $\eta_{n,0}$ goes to 0. Now (8) is equivalent to $(\log_2 \frac{16\sqrt{6}\eta_{n,0}}{M}+1)^2 \le \frac{nM^2}{2^{16}C_0}$. Note that $\frac{nM^2}{2^{16}C_0}$ is of the order $O_P(\log(n))^2$. The order of $(\log_2 \frac{16\sqrt{6}\eta_{n,0}}{M}+1)^2$ is less than that of $(\log\frac{1}{M})^2$, which is $O_P(\log\frac{n^{1/2}}{\log(n)})^2$, and $O_P(\log\frac{n^{1/2}}{\log(n)})^2 < O_P(\log(n))^2$. Thus with n large enough, (8) holds.

Now we prove $P(e(\hat{f}_n,f^{(\infty)})\ge 8sM+d_{n,s}) \le 6(1-\frac{1}{16nM^2})^{-1}\exp(-nM^2)$. We have, by definition of $d_{n,s}$,

$$P(e(\hat{f}_n,f^{(\infty)})>8sM+d_{n,s}) \le P(e(\hat{f}_n,f_n^{(s)})(2s)^{-1}>4M).$$

Then because $\frac{1}{n}\sum_{i=1}^n(L(f_n^{(s)},y_i)-L(\hat{f},y_i))>0$, we have

$$\begin{aligned}
&P(e(\hat{f}_n,f^{(\infty)})>8sM+d_{n,s})\\
&\le P^*\Big(\sup_{f\in\mathcal{F}_n(s):e(f,f_n^{(s)})(2s)^{-1}>4M}\frac{1}{n}\sum_{i=1}^n(L(f_n^{(s)},y_i)-L(f,y_i))>0\Big)\\
&\le P^*\Big(\sup_{f\in\mathcal{F}_n(s):e(f,f_n^{(s)})(2s)^{-1}>4M}-\frac{1}{n}\sum_{i=1}^n[g_f(y_i)-Eg_f(y)]>(2s)^{-1}E(L(f,Y)-L(f_n^{(s)},Y))\Big)\\
&\le P^*(\sup_{g_f\in\mathcal{G}}|P_ng_f-Pg_f|>4M).
\end{aligned}$$

Here P^* denotes the outer probability, and the expectation is taken jointly with respect to the distribution of X and the noise. In the last inequality, for brevity we have the empirical process $g_f\to P_ng_f-Pg_f$, where $g_f\in\mathcal{G}$, $Pg_f=\int g_f$ and $P_ng_f=\frac{1}{n}\sum_{i=1}^n g_f(y_i)$.

Now we show that $P^*(\sup_{g_f\in\mathcal{G}}|P_ng_f-Pg_f|>4M)\le 6(1-\frac{1}{16nM^2})^{-1}\exp(-nM^2)$.

This part of proof follows a similar line as Theorem A.2 in Wang and Shen (2007). There are two steps involved. The first step is to sample n observations without replacement from $N=2n$ instances, which are $i.i.d.$ samples from P, and let $(W_1,\ldots,W_N)$ be uniformly distributed on the set of all permutations of $1,\ldots,N$. Define $\tilde{P}_{n,N}=\frac{1}{n}\sum_{i=1}^n\delta_{(X)_{W_i}}$, and $P_N=\frac{1}{N}\sum_{i=1}^N\delta_{(X)_i}$, where $\delta_{(X)_i}$ is the Dirac measure at the observation X_i. Then we can bound the LHS of the required inequality by Lemma 2.14.18 in Van der Vaart and Wellner (2000),

$$P^*(\sup_{g_f\in\mathcal{G}}|P_ng_f-Pg_f|>4M)\le(1-\frac{1}{16nM^2})^{-1}P^*_{|N}(\sup_{\mathcal{G}}|\tilde{P}_{n,N}g_f-P_Ng_f|>M), \tag{10}$$

where $P_{|N}$ is the conditional probability given N observations.

The second step is to bound $P^*_{|N}(\sup_{\mathcal{G}}|\tilde{P}_{n,N}g_f-P_Ng_f|>M)$. We apply the chaining technique here. Let $\eta_{n,0}>\eta_1>\cdots>\eta_T>0$ be a sequence of positive numbers to be determined later

on. Let $\mathcal{G}_q$ be the minimal η_q-net for $\mathcal{G}$ with respect to the $L_2(T_X)$ norm. For each q, let $\pi_qg=\text{argmin}_{h\in\mathcal{G}_q}\|g-h\|_{L_2(T_X)}$. That means, π_qg is the closest point to g within $\mathcal{G}_q$. By definition, $|\mathcal{G}_q|=N(\eta_q,\mathcal{G},L_2(T_X))$, and $\|\pi_qg-g\|_{L_2(T_X)}\le\eta_q$. Hence, decompose $P^*_{|N}(\sup_{\mathcal{G}}|\tilde{P}_{n,N}g_f-P_Ng_f|>M)$ into

$$\begin{aligned}
&P^*_{|N}(\sup_{\mathcal{G}}|\tilde{P}_{n,N}g_f-P_Ng_f|>M)\\
&\le P^*_{|N}(\sup_{\mathcal{G}}|(\tilde{P}_{n,N}-P_N)(\pi_0g_f)|>7M/8)\\
&\quad+P^*_{|N}(\sup_{\mathcal{G}}|(\tilde{P}_{n,N}-P_N)(\pi_0g_f-\pi_Tg_f)|>M/16)\\
&\quad+P^*_{|N}(\sup_{\mathcal{G}}|(\tilde{P}_{n,N}-P_N)(\pi_Tg_f-g_f)|>M/16)\\
&\le|\mathcal{G}_0|\sup_{\mathcal{G}}P^*_{|N}(|(\tilde{P}_{n,N}-P_N)(\pi_0g_f)|>7M/8)\\
&\quad+\sum_{q=1}^T|\mathcal{G}_q||\mathcal{G}_{q-1}|\sup_{\mathcal{G}}P^*_{|N}(|(\tilde{P}_{n,N}-P_N)(\pi_0g_f-\pi_Tg_f)|>\chi)\\
&\quad+P^*_{|N}(\sup_{\mathcal{G}}|(\tilde{P}_{n,N}-P_N)(\pi_Tg_f-g_f)|>M/16)\\
&:=P_1+P_2+P_3,
\end{aligned}$$

where $T\chi\le M/16$. Next, we bound P_1, P_2 and P_3 individually.

For P_3, one can verify that $\tilde{P}_{n,N}f\le 2P_Nf$ for any non-negative f. Note that we have $((\tilde{P}_{n,N}-P_N)z)^2\le 2(\tilde{P}_{n,N}z^2+P_Nz^2)$ for any z. Thus, $|(\tilde{P}_{n,N}-P_N)(\pi_Tg_f-g_f)|^2\le 2(\tilde{P}_{n,N}+P_N)(\pi_Tg_f-g_f)^2\le 6\eta_T^2$. This yields $P_3=0$ if we choose $\eta_T=\frac{M}{16\sqrt{6}}$.

For P_1, note that $0\le\pi_0g_f\le 1$ for any $g_f\in\mathcal{G}$ because $\mathcal{G}$ is scaled. By Hoeffding's inequalities for sums of bounded random variables (Hoeffding, 1963), $P^*_{|N}(|(\tilde{P}_{n,N}-P_N)(\pi_0g_f)|>7M/8)\le 2\exp(-2n(7/8)^2M^2)$. Thus by assumption (9) and Lemma 14,

$$P_1\le 2N(\eta_{n,0},\mathcal{G},L_2(T_X))\exp(-2n(7/8)^2M^2)\le 2\exp(-nM^2).$$

For P_2, if $\eta_{n,0}\le\frac{M}{16\sqrt{6}}$, then let $\eta_q=\eta_{n,0};q=1,\ldots,T$, and we have $P_2=0$ by a similar argument as in the P_3 part discussed above. So suppose $\eta_{n,0}>\frac{M}{16\sqrt{6}}>\eta_T$. Note that $P_N(\pi_qg_f-\pi_{q-1}g_f)^2\le 2(P_N(\pi_qg_f-g_f)^2+P_N(\pi_{q-1}g_f-g_f)^2)\le 4\eta_{q-1}^2$. By Massart's inequality from Lemma 2.14.19 in Van der Vaart and Wellner (2000), we have $P^*_{|N}(|(\tilde{P}_{n,N}-P_N)(\pi_0g_f-\pi_Tg_f)|>\chi)\le 2\exp(-\frac{n\chi^2}{2\sigma_N^2})$ with $\sigma_N^2=P_N(\pi_qg_f-\pi_{q-1}g_f)^2\le 4\eta_{q-1}^2$. So,

$$\begin{aligned}
P_2&\le 2\sum_{q=1}^T|\mathcal{G}_q|^2\exp(-\frac{n\chi^2}{2\sigma_N^2})\\
&\le 2\sum_{q=1}^T\exp(2C_0\eta_q^{-2}+2\log\frac{5}{\eta_q}-\frac{n\chi^2}{8\eta_{q-1}^2}).
\end{aligned}$$

Let $\eta_q=2^{-q}\eta;q=1,\ldots,T$, and let T be the greatest integer that does not exceed $\log_2\frac{16\sqrt{6}\eta_{n,0}}{M}$. Then let $\chi=(\frac{256C_0}{n})^{1/2}$. By assumption (8), we can verify that $T\chi\le M/16$ as satisfied. Because

when η is small enough, $\frac{C_0}{\eta^2} > \log\frac{5}{\eta}$, and this leads to

$$\begin{aligned} P_2 &\le 2\sum_{q=1}^{T}\exp(\frac{-4}{\eta_q^2}C_0) \\ &\le 2\sum_{q=1}^{T}\exp(-2^{2q-1}(nM^2)) \\ &\le 4\exp(-nM^2). \end{aligned}$$

Thus, $P^*_{|N}(\sup_{\mathcal{G}}|\tilde{P}_{n,N}g_f - P_N g_f| > M) < 6\exp(-nM^2)$. The desired result follows after we take expectation with respect to the distribution of N observations. We have proved $P^*(\sup_{g_f\in\mathcal{G}}|P_n g_f - Pg_f| > 4M) \le 6(1-\frac{1}{16nM^2})^{-1}\exp(-nM^2)$.

Finally, observe that $nM^2 = 2(\log(n))^2 > 2\log(n)$. We have $\exp(-nM^2) \le \exp(-2\log(n)) = \frac{1}{n^2}$. The desired result in Theorem 1 then follows from Borel-Cantelli Lemma. ■

Appendix B. Proof of Corollary 5

The key to the proof is to show that with a high probability, the estimated b would be bounded in a range. Then we can apply the same technique as that in the proof of Theorem 1 to prove the desired result.

Without loss of generality, assume $|f_0(X)| < \zeta$ for a fixed $\zeta > 0$. Moreover, for simplicity, we assume that $\varepsilon(X)$ follows a common sub-Gaussian distribution with c.d.f. Φ_ε. The generalization to heteroscedastic cases is straightforward, because we are only concerned with the tail probability $\mathrm{pr}(|\varepsilon(X)| > t)$. Next, for a small positive number δ, define $t^* = \Phi_\varepsilon^{-1}\big(0.5+0.5(1-\delta/2)^{1/n}\big)$. One can verify that with probability at least $1-\delta/2$, all the errors ε_i; $i = 1,\ldots,n$ are in $[-t^*,t^*]$. Therefore, for any τ, we have that with probability at least $1-\delta/2$, $|b| \le \zeta + t^*$. This is because the estimated function cannot be smaller (or larger) than all the observations. Hence, letting $s^* = s+\zeta+t^*$, $M^* = \sqrt{2}n^{-1/2}\log(n)/t^*$ and using similar techniques as that in the proof of Theorem 1, we have $P(e(\hat{f}_n^*, f^{(\infty)}) > 8s^*M^* + d_{n,s^*}) \le 6(1-\frac{1}{16nM^{*2}})^{-1}\exp(-nM^{*2}) + \delta/2$. Let δ converge to zero at the rate $O_P(n^{-2}\log(n))$. The last step is to check that (8) and (9) are both true with our new choice of M^*. Because we assume that Φ_ε is the c.d.f. of a sub-Gaussian distribution with a fixed parameter, one can verify that t^* diverges at a rate slower than $O_P(\log(n))$. Hence, (8) and (9) remain valid, and the Borel-Cantelli Lemma as in the final step of the proof of Theorem 1 holds. This completes the proof. ■

Appendix C. Proof of Theorem 7

The proof uses a similar technique as Theorem 2.7 in Steinwart and Scovel (2007). Consider the function

$$V(x) = C\int_D \exp(\frac{|x-x'|^2}{2\sigma^2})f_{\text{true}}(x')dx', \tag{11}$$

where C is a constant that depends only on σ and p and is to be determined later on. One can verify that $V(x) \in \mathcal{F}(\infty)$ (Steinwart et al., 2006). Hence,

$$\begin{aligned} A(\infty) &\le E\big(\rho_\tau(Y-V)\big) - E\big(\rho_\tau(Y-f_{\text{true}})\big) \\ &\le E_{P_X}\big(\rho_\tau(V-f_{\text{true}})\big). \end{aligned}$$

Therefore, one only needs to bound $E_{P_X}\big(\rho_\tau(V-f_{\text{true}})\big)$. We have

$$\begin{aligned} V(x) &= C\int_D \exp(\frac{|x-x'|^2}{2\sigma^2})\big(f_{\text{true}}(x')+a\big)dx' - a \\ &\ge C\int_{B(x,\psi_x)} \exp(\frac{|x-x'|^2}{2\sigma^2})\big(f_{\text{true}}(x')+a\big)dx' - a \\ &= a_i - (a_i+a)P'(|u| \ge \psi_x), \end{aligned}$$

where the first inequality is because $f_{\text{true}}(x')+a$ is lower bounded by 0, and on $B(x,\psi_x)$ the function f_{true} is constant. Here one chooses C such that P' is the measure of a spherical Gaussian in D (see the definition of spherical Gaussian in, for example, Steinwart, 2002). By the inequality (3.5) on page 59 of Ledoux and Talagrand (1991), $P'(|u| \ge \psi_x) \le 4\exp\big(-\psi_x^2/(2p\sigma^2)\big)$, and consequently we have that on D_i,

$$V - f_{\text{true}} \ge -8a\exp\big(-\psi_x^2/(2p\sigma^2)\big).$$

An analogous derivation on $-f_{\text{true}}$ and $-V$ gives

$$V - f_{\text{true}} \le 8a\exp\big(-\psi_x^2/(2p\sigma^2)\big).$$

Therefore

$$E\big(\rho_\tau(V-f_{\text{true}})\big) \le \max(\tau,1-\tau)E_{P_X}\big\{8a\exp\big(-\psi_x^2/(2p\sigma^2)\big)\big\}.$$

This completes the proof.

Notice that the core part of the proof is at the inequality (3.5) on page 59 of Ledoux and Talagrand (1991). For the Gaussian kernel (and other radial kernels), the probability $P'(|u| \ge \psi_x)$ vanishes as $\sigma \to 0$. However, for other kernels this may not be true. Take the polynomial kernel as an example. One can verify that when $|x_1 - x_2|$ is large, $K(x_1,x_2)$ is large, and this leads to $P'(|u| \ge \psi_x)$ being large. Therefore, the result here does not hold true for general RKHS's. ■

Appendix D. Proof of Corollary 8

Without loss of generality, let the Lipschitz constant of f_{true} be 1. In this proof, let ε be a small positive number, instead of the noise as $Y = f_0 + \varepsilon$. We first consider the approximation of V to f_{true} on $[0,1]^p$, where V is defined as in the proof of Theorem 7. Because $[0,1]^p$ is a compact set, there exists a finite set of $B(x_j,\varepsilon)$; $j = 1,\ldots,J$ that covers $[0,1]^p$. Here $B(x_j,\varepsilon)$ is a ball with center at x_j and radius ε, and J is a positive integer. Based on $B(x_j,\varepsilon)$; $j = 1,\ldots,J$, one can construct sets $S_{x_j} \subset B(x_j,\varepsilon)$; $j = 1,\ldots,J$ such that S_{x_i} and S_{x_j} are non-overlapping for $i \ne j$, and $\bigcup_{j=1}^J S_{x_j} = [0,1]^p$. On each S_{x_j}, one can verify that

$$f_{\text{true}}(x_j) - \varepsilon \le f_{\text{true}} \le f_{\text{true}}(x_j) + \varepsilon.$$

Now on S_{x_j}, define $\bar{f}_{\text{true}} = f_{\text{true}}(x_j)$. We have that

$$E_{[0,1]^p}\rho_\tau(V - f_{\text{true}}) \le E_{[0,1]^p}\rho_\tau(V - \bar{f}_{\text{true}}) + E_{[0,1]^p}\rho_\tau(\bar{f}_{\text{true}} - f_{\text{true}}). \tag{12}$$

By Theorem 7 and the discussion thereafter, the first part on the right hand side of (12) goes to 0 with $\sigma \to 0$, and the second part is upper bounded by ε. Let $\varepsilon \to 0$ and this proves that V can approximate f_{true} arbitrarily well on $[0,1]^p$. For the approximation on D, notice that D can be decomposed into countably many sets such as $[0,1]^p$, and a similar argument as above proves the corollary. ■

Appendix E. Proof of Theorem 9

To prove Theorem 9, we need to introduce the Rademacher variables (see, for example, Bartlett and Mendelson (2002), Koltchinskii and Panchenko (2002), Shawe-Taylor and Cristianini (2004), Bartlett et al. (2005), Koltchinskii (2006), Mohri et al. (2012) and the references therein). With a little abusing of notations, let σ_i; $i = 1,\dots,n$ be *i.i.d.* random variables that take 1 with probability $1/2$, and -1 with probability $1/2$. Denote by S a sample of (x_i, y_i); $i = 1,\dots,n$, *i.i.d.* from the joint distribution of X and Y. Recall the definition of $\mathcal{F}_n(s)$ in Section 3.1. With S fixed, we define the empirical Rademacher complexity of the function class $\mathcal{F}_n(s)$ as

$$\hat{R}_n(\mathcal{F}_n(s)) = E_\sigma[\sup_{f\in\mathcal{F}_n(s)} \frac{1}{n}\sum_{i=1}^n \sigma_i \rho_\tau\big(y_i - f(x_i)\big)],$$

where E_σ represents the expectation with respect to $\sigma = (\sigma_1,\dots,\sigma_n)$. Moreover, let the Rademacher complexity of $\mathcal{F}_n(s)$ be

$$R_n(\mathcal{F}_n(s)) = E_S \hat{R}_n(\mathcal{F}_n(s)),$$

where E_S is the expectation with respect to the distribution of S.

The proof of Theorem 9 follows directly from Lemmas 15, 17 and 18. Lemma 15 bounds $E\rho_\tau(Y - \hat{f}_n)$ or $E\rho_\tau(Y - \tilde{f}_n)$ in terms of the sum of its empirical measurement, the Rademacher complexity of the function class $\mathcal{F}_n(s)$, and a penalty term on δ, where δ is the small probability that the bound fails. Lemma 17 and Lemma 18 bound the Rademacher complexity. In particular, Lemma 17 provides the bound with $\mu = \sqrt{n^{-1}\big(2^{11}n^{1/4} + 2\log(5) + 0.5\log(n)\big)}$ that works for both the regular squared norm method and the data sparsity method. This is because the complexity bound of $\mathcal{F}_n(s)$ is from Lemma 14, which holds for both methods. As discussed in the main text, we provide another bound that only works for the data sparsity method in Lemma 18, which leads to $\mu = s\sqrt{\frac{2\log(2n+2)}{n}}$.

Lemma 15 *Define $R_n(\mathcal{F}_n(s))$ and $\hat{R}_n(\mathcal{F}_n(s))$ as above. Suppose Assumption A holds. With probability at least $1-\delta$,*

$$E\rho_\tau(Y - \hat{f}_n) \le \frac{1}{n}\sum_{i=1}^n \rho_\tau(y_i - \hat{f}_n(x_i)) + 2R_n(\mathcal{F}_n(s)) + T_n(\delta), \tag{13}$$

where $T_n(\delta) = \max(\tau, 1-\tau)\Big(n^{-1}(2s^2 + 2t^2)\log(1/\delta)\Big)^{1/2}$. In addition, with probability at least $1-\delta$,

$$E\rho_\tau(Y - \hat{f}_n) \le \frac{1}{n}\sum_{i=1}^n \rho_\tau(y_i - \hat{f}_n(x_i)) + 2\hat{R}_n(\mathcal{F}_n(s)) + 3T_n(\delta/2). \tag{14}$$

Moreover, (13) and (14) hold for $\tilde{f}_n$.

Proof of Lemma 15: We divide the proof into three parts. In the first part, we bound the left hand side of (13) in terms of its empirical estimation and the expectation of their supremum difference, by the McDiarmid inequality (McDiarmid, 1989). The second part bounds the expectation of the supremum difference from the first step by the previously defined Rademacher complexity with a symmetrization technique. In the third part, we bound the Rademacher complexity by its empirical version. In this proof, we focus on $\hat{f}_n$, as the proof for $\tilde{f}_n$ is the same.

We begin the proof by introducing some notation. For a given sample S, let

$$\phi(S) = \sup_{f\in\mathcal{F}_n(s)} [E\rho_\tau(Y - f(X)) - \frac{1}{n}\sum_{i=1}^n \rho_\tau(y_i - f(x_i))].$$

Define $S^{(i,x)} = \{(x_1,y_1),\dots,(x_i',y_i),\dots,(x_n,y_n)\}$ to be another sample from the joint distribution of X and Y. Notice that the difference between S and $S^{(i,x)}$ is only on the x value of their i^{th} pair. Similarly, define $S^{(i,y)} = \{(x_1,y_1),\dots,(x_i,y_i'),\dots,(x_n,y_n)\}$ with the y values in the i^{th} pair being different. Then we have

$$\begin{aligned}|\phi(S) - \phi(S^{(i,x)})| = |&\sup_{f\in\mathcal{F}_n(s)} [E\rho_\tau(Y - f(X)) - \frac{1}{n}\sum_{S} \rho_\tau(y_i - f(x_i))] \\ &- \sup_{f\in\mathcal{F}_n(s)} [E\rho_\tau(Y - f(X)) - \frac{1}{n}\sum_{S^{(i,x)}} \rho_\tau(y_i - f(x_i))]|.\end{aligned} \tag{15}$$

For simplicity, we consider only the case where there exists a measurable function $f^S \in \mathcal{F}_n(s)$ that achieves the supremum of $\phi(S)$. Note that the case of no function achieving the supremum can be treated similarly, with only minor modification on the proof, and the details are omitted. Substitute f^S in (15) and after some calculation, we have that

$$|\phi(S) - \phi(S^{(i,x)})| \le \frac{2}{n}\max(\tau, 1-\tau)s.$$

Similarly, one can verify that $|\phi(S) - \phi(S^{(i,y)})| \le \frac{2}{n}\max(\tau,1-\tau)t$. Hence, by the McDiarmid inequality, for any $z > 0$, $P(\phi(S) - E\phi(S) \ge z) \le \exp\Big(-\frac{2z^2}{\frac{1}{n}\left(\max(\tau,1-\tau)\right)^2(4s^2+4t^2)}\Big)$. Therefore, with probability at least $1-\delta$, $\phi(S) - E\phi(S) \le T_n(\delta)$. This proves the first part of the lemma.

In the second step, we bound $E\{\phi(S)\}$ by the Rademacher complexity $R_n(\mathcal{F}_n(s))$ using a symmetrization technique. To this end, define $S' = \{(x_i', y_i')\ (i = 1,\dots,n)\}$ as a duplicate sample of S with size n, and assume the distribution of S' is the same as S. Recall the definition of E_S. Moreover, notice that

$$E_{S'}[\frac{1}{n}\sum_{S}\{\rho_\tau(y_i' - \hat{f}_n(x_i')) \mid S\}] = \frac{1}{n}\sum_{S}\{\rho_\tau(y_i' - \hat{f}_n(x_i')),$$

and

$$E_{S'}[\frac{1}{n}\sum_{S'}\{\rho_\tau(y_i' - \hat{f}_n(x_i')) \mid S\}] = E\rho_\tau(Y - \hat{f}_n(X)).$$

Hence, with the Jensen's inequality and the definition of σ, we have

$$\begin{aligned}
E\{\phi(S)\} &= E_S\big(\sup_{f\in\mathcal{F}_n(s)} E_{S'}\big[\frac{1}{n}\sum_{S'}\{\rho_\tau(y_i'-\hat{f}_n(x_i'))\}-\frac{1}{n}\sum_{S}\{\rho_\tau(y_i-\hat{f}_n(x_i))\}\big]\mid S\big)\\
&\le E_{S,S'}\big[\frac{1}{n}\sum_{S'}\{\rho_\tau(y_i'-\hat{f}_n(x_i'))\}-\frac{1}{n}\sum_{S}\{\rho_\tau(y_i-\hat{f}_n(x_i))\}\big]\\
&= E_{S,S',\sigma}\big[\sup_{f\in\mathcal{F}_n(s)}\frac{1}{n}\sum_{S'}\{\rho_\tau(y_i'-\hat{f}_n(x_i'))\}-\frac{1}{n}\sum_{S}\{\rho_\tau(y_i-\hat{f}_n(x_i))\}\big]\\
&\le 2R_n(\mathcal{F}_n(s)).
\end{aligned}$$

This completes the proof of the second step.

The third step bounds $R_n(\mathcal{F}_n(s))$ by the empirical counterpart $\hat{R}_n(\mathcal{F}_n(s))$. This part of the proof is similar to that of the first part, in the sense that we apply the McDiarmid inequality on $\hat{R}_n(\mathcal{F}_n(s))$ and its expectation $R_n(\mathcal{F}_n(s))$. One can then verify that with probability at least $1-\delta$, $R_n(\mathcal{F}_n(s)) \le \hat{R}_n(\mathcal{F}_n(s))+T_n(\delta)$.

The proof of Lemma 15 is thus completed, after combining the results in Steps 1-3 and replacing δ by $\delta/2$. □

The next lemma, Lemma 17, bounds $\hat{R}_n(\mathcal{F}_n(s))$ with the data and tuning parameter that we use. It employs the result obtained in Lemma 14, and is a direct application of the "η-net" idea (Van der Vaart and Wellner, 2000). Because the result in Lemma 14 can be applied to both the regular squared norm method and the data sparsity method, the result in Lemma 17 holds for both methods as well. Before discussing Lemma 17 and its proof, we first introduce the Hoeffding's Inequality.

Proposition 16 *(Hoeffding's Inequality). Let X be a random variable with mean 0 and range in $[a,b]$. Then for any fixed $z>0$, $E(\exp(zX)) \le \exp(z^2(b-a)^2/8)$.*

Lemma 17 *The empirical Rademacher complexity $\hat{R}_n(\mathcal{F}_n(s))$ for $\tilde{f}_n$ satisfies that*

$$\hat{R}_n(\mathcal{F}_n(s)) \le 2sn^{-1/4}+\max(\tau,1-\tau)(s+t)\sqrt{2^{11}n^{-1/2}+(\log(5)/n)+(\log(n)/4n)}.$$

Proof of Lemma 17: Let $R=\frac{\hat{R}_n\mathcal{F}_n(s)}{2s}$. We consider the following function $h_f(\cdot)=(2s)^{-1}\rho_\tau(f-\cdot)$, and the corresponding class $\mathcal{H}_R=\{h_f : f\in\mathcal{F}_n(s)\}$. From the proof of Lemma 14, one can verify that the entropy number of $\mathcal{H}_R$ is bounded by that of $\mathcal{G}$, because the check function is upper bounded by $L(f,\cdot)$ defined after the proof of Lemma 13. Next, we construct the smallest η-net of $\mathcal{H}_R$, $\mathcal{Y}$, such that for all $f\in\mathcal{F}_n(s)$, there exists an element $g_\mathcal{Y}\in\mathcal{Y}$ with the $L_2(T_X)$ distance between f and $g_\mathcal{Y}$ smaller than η, for any arbitrary empirical measure T_X. Therefore, one can verify that

$$\begin{aligned}
R \le& \frac{1}{2s}E_\sigma[\sup_{g_\mathcal{Y}}\frac{1}{n}\sum_{i=1}^n\sigma_i\rho_\tau(g_\mathcal{Y}(x_i)-y_i)]\\
&+\frac{1}{2s}E_\sigma[\sup_{f,g_\mathcal{Y}\ \text{close}}\frac{1}{n}\sum_{i=1}^n\sigma_i(\rho_\tau(f(x_i)-y_i)-\rho_\tau(g_\mathcal{Y}(x_i)-y_i))]\\
:=&R_1+R_2.
\end{aligned}$$

Here "$f, g_\mathcal{Y}$ close" means that the $L_2(T_X)$ distance between f and $g_\mathcal{Y}$ is smaller than η. Consequently, we have that R_2 is bounded by η, by the Hölder's Inequality.

Now we bound R_1. To this end, let z be a positive number to be determined later. From the Jensen's Inequality, we have

$$\begin{aligned}
\exp(2sznR_1) \le& E_\sigma\exp[z\sup_{g_\mathcal{Y}}\sum_{i=1}^n\sigma_i\rho_\tau(g_\mathcal{Y}(x_i)-y_i)]\\
\le&\sum_{g_\mathcal{Y}}E_\sigma[\exp(z\sum_{i=1}^n\sigma_i\rho_\tau(g_\mathcal{Y}(x_i)-y_i))]\\
\le&\sum_{g_\mathcal{Y}}\prod_{i=1}^n E_{\sigma_i}[\exp(z\sigma_i\rho_\tau(g_\mathcal{Y}(x_i)-y_i)].
\end{aligned}$$

Observe that $E_{\sigma_i}[\sigma_i\rho_\tau(g_\mathcal{Y}(x_i)-y_i)]=0$, and

$$-\max(\tau,1-\tau)(s+t) \le \rho_\tau(g_\mathcal{Y}(x_i)-y_i) \le \max(\tau,1-\tau)(s+t).$$

Therefore, by the Hoeffding's Inequality, we have

$$\begin{aligned}
\exp(2sznR_1) \le&\sum_{g_\mathcal{Y}}\prod_{i=1}^n\exp(\frac{z^2(2\max(\tau,1-\tau)(s+t))^2}{8})\\
\le&|\mathcal{Y}|\exp(\frac{nz^2(\max(\tau,1-\tau)(s+t))^2}{2}).
\end{aligned}$$

Equivalently, we have $2snR_1 \le \frac{\log|\mathcal{Y}|}{z}+\frac{nz(\max(\tau,1-\tau)(s+t))^2}{2}$. Choose $z=\frac{\sqrt{2\log|\mathcal{Y}|}}{\max(\tau,1-\tau)(s+t)\sqrt{n}}$, and we have

$$2snR_1 \le \max(\tau,1-\tau)(s+t)\sqrt{2n\log|\mathcal{Y}|},$$

or equivalently,

$$R_1 \le \frac{\max(\tau,1-\tau)(s+t)\sqrt{2\log|\mathcal{Y}|}}{2s\sqrt{n}} \le \frac{\max(\tau,1-\tau)(s+t)}{2s\sqrt{n}}\sqrt{2(C_0\eta^{-2}+\log(5/\eta))}.$$

After we combine the bounds of R_1 and R_2, choose $\eta=n^{-1/4}$, the results then follows. □

Next, we focus on the data sparsity method. In Lemma 18 we would prove that the empirical Rademacher complexity of the functional space in (5) can be smaller than that of (4). As we will see, the technique we use only works for the data sparsity constraint. This bound then leads to another finite sample bound on the prediction error, namely, $\mu=s\sqrt{\frac{2\log(2n+2)}{n}}$. As discussed in the main text, when n is small or moderate, this bound can be much better than the one derived from Lemma 14.

Lemma 18 *The empirical Rademacher complexity $\hat{R}_n(\mathcal{F}_n(s))$ for $\hat{f}_n$ in (5) satisfies that*

$$\hat{R}_n(\mathcal{F}_n(s)) \le s\max(\tau,1-\tau)\sqrt{\frac{2\log(2n+2)}{n}}.$$

Proof of Lemma 18: We first consider the empirical Rademacher complexity of the functional space $\mathcal{F}_n(s)$ without taking the check loss function into consideration. To this end, let us define $\hat{R}_f(\mathcal{F}_n(s)) = E_\sigma[\sup_{f\in\mathcal{F}_n(s)} \frac{1}{n}\sum_{i=1}^n \sigma_i f(x_i)]$. Recall the definition of $\bar{\alpha}$ from Section 4, and define the augmented vector $\tilde{K}_i = (1, K_i)$ for $i = 1,\ldots,n$, where K_i is the i^{th} row of the gram matrix K. We can now rewrite $\hat{R}_f(\mathcal{F}_n(s))$ as

$$\begin{aligned}\hat{R}_f(\mathcal{F}_n(s)) &= E_\sigma[\sup_{\|\bar{\alpha}\|_1\le s} \frac{1}{n}\sum_{i=1}^n \sigma_i \bar{\alpha}^T \tilde{K}_i] \\ &= \frac{s}{n} E_\sigma \|\sum_{i=1}^n \sigma_i \tilde{K}_i\|_\infty \\ &= \frac{s}{n} E_\sigma[\max_{\sigma', j=1,\ldots,n+1} \sigma' \sum_{i=1}^n \sigma_i \tilde{K}_i(j)],\end{aligned}$$

where $\|\cdot\|_\infty$ is the L_∞ norm, $\tilde{K}_i(j)$ is the j^{th} element of $\tilde{K}_i$, and σ' is an independent Rademacher variable. Notice that the new functional space defined by

$$\{\sigma'(\tilde{K}_1(j), \tilde{K}_2(j), \ldots, \tilde{K}_n(j))^T;\ j = 1,\ldots,n+1, \sigma' \in \{\pm 1\}\}$$

consists of $2n+2$ elements.

Next, by applying the same technique as we used in the proof of Lemma 17 to bound R_1, we can show that

$$\hat{R}_f(\mathcal{F}_n(s)) \le s\sqrt{\frac{2\log(2n+2)}{n}}.$$

The rest of the proof is to apply the Talagrand's lemma (Lemma 4.2 on page 78 in Mohri et al., 2012). In particular, as the check loss function is $\max(\tau, 1-\tau)$-Lipschitz, we have

$$\hat{R}_n(\mathcal{F}_n(s)) \le \max(\tau, 1-\tau)\hat{R}_f(\mathcal{F}_n(s)) \le s\max(\tau, 1-\tau)\sqrt{\frac{2\log(2n+2)}{n}}.$$

This completes the proof. ■

Appendix F. Proof of Corollary 10

With the t defined in Corollary 10, one can verify that with probability at least $1-\delta/2$, all the errors ε_i; $i = 1,\ldots,n$ are in $[-t, t]$. Conditioning on this, the claim follows from Theorem 9. ■

Appendix G. Proof of Proposition 12

The proof follows directly from that of Theorem 1 in Li et al. (2007) and is omitted. ■

Section 2	**Methodology**
x, X	Predictor variable
y, Y	Response
n	Number of observation
p	Dimensionality of x
τ	Quantile level
$\varepsilon(\cdot)$	Noise, may depend on x
$f_0(\cdot)$	Defined as $Y = f_0 + \varepsilon$
D	Domain of f_0
$\rho_\tau(\cdot)$	The check function
f_{true}	The population minimizer of the check function
$J(\cdot)$	Penalty on the regression function
λ, s	Tuning parameters
$\mathcal{F}$	Functional class
$\mathbb{R}$	The real line
$\mathcal{H}$	A RKHS
$\|\cdot\|_{\mathcal{H}}$	The norm in the RKHS $\mathcal{H}$
b	Intercept
f'	Regression function in $\mathcal{H}$ without an explicit intercept
f	Regression function in $\mathcal{H}\oplus\mathbb{R}$
$K(\cdot,\cdot)$	The kernel function
K	The gram matrix
$\alpha = (\alpha_1,\ldots,\alpha_n)$	The kernel function coefficients
$\hat{f}_n$	The estimated regression function using the proposed data sparsity constraint
$\tilde{f}_n$	The estimated regression function using the squared norm penalty

Table 11: Important notation introduced in Section 2.

References

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

G. Blanchard, O. Bousquet, and P. Massart. Statistical performance of support vector machines. *Annals of Statistics*, 36(2):489–531, 2008.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X. doi: http://doi.acm.org/10.1145/130385.130401. URL http://doi.acm.org/10.1145/130385.130401.

O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Section 3	**Statistical Theory**
$\mathcal{F}_n(s)$	$\{f = f' + b : f'(x) = \sum_{i=1}^n \alpha_i K(x,x_i); \lvert b\rvert + \sum_{i=1}^n \lvert\alpha_i\rvert \le s\}$
$\mathcal{F}(\infty)$	$\lim_{s\to\infty}\lim_{n\to\infty} \mathcal{F}_n(s)$
$f_n^{(s)}$	$\operatorname{arginf}_{f\in\mathcal{F}_n(s)} E\rho_\tau(Y - f(X))$
$f^{(\infty)}$	$\operatorname{arginf}_{f\in\mathcal{F}(\infty)} E\rho_\tau(Y - f(X))$
$e(f_1, f_2)$	$E\rho_\tau(Y - f_1(X)) - E\rho_\tau(Y - f_2(X))$
$d_{n,s}$	$e(f_n^{(s)}, f^{(\infty)})$, the approximation error between $\mathcal{F}_n(s)$ and $\mathcal{F}(\infty)$
$\hat{f}_n^*$	The estimated function using the data sparsity constraint, without penalty on $\lvert b\rvert$
$\lVert f\rVert_{L_2(Q)}$	$(\int f^2 dQ)^{1/2}$, the $L_2(Q)$ norm of f
$A(\infty)$	$E\big(\rho_\tau(Y - f^{(\infty)})\big) - E\big(\rho_\tau(Y - f_{\text{true}})\big)$
a_i	Constants
D_i	A partition of D, and $f_{\text{true}} = a_i$ on D_i
a	Upper bound on $\lvert f_{\text{true}}\rvert$
$\text{dis}(x, D_j)$	The distance between the point x and the set D_j
ψ_x	$\min_{j\neq i}\text{dis}(x, D_j)$
$B(x, \psi_x)$	The ball centered at x with radius ψ_x
σ	Kernel parameter for Gaussian/Laplacian kernels
P_X	The marginal distribution of X
t	The upper bound of $\lvert\varepsilon\rvert$ in Assumption A
δ	A small probability
μ	$\min\Big(2\sqrt{n^{-1/2}\big(\log(n)+1\big)}, \sqrt{n^{-1}\big(2^{11}n^{1/4} + 2\log(5) + 0.5\log(n)\big)}\Big)$
Φ_ε	The common cumulative distribution function of ε

Table 12: Important notation introduced in Section 3.

B. Carl and I. Stephani. *Entropy, Compactness and the Approximation of Operators*, volume 98. Cambridge University Press, 1990.

R. Chappell. Fitting bent lines to data, with applications to allometry. *Journal of Theoretical Biology*, 138:235–256, 1989.

D.-R. Chen, Q. Wu, Y. Ying, and D.-X. Zhou. Support vector machine soft margin classifiers: error analysis. *Journal of Machine Learning Research*, 5:1143–1175, 2004.

C. Cortes and V. N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

F. Cucker and S. Smale. On the mathematical foundations of learning. *American Mathematical Society*, 39(1):1–49, 2002.

C. De Boor. *A Practical Guide to Splines*. Springer-Verlag, 1978.

H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. Support vector regression machines. *Advances in Neural Information Processing Systems*, pages 155–161, 1997.

B. Efron, T. J. Hastie, I. Johnstone, and R. J. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–451, 2004.

P. H. C. Eilers and B. D. Marx. Flexible smoothing using B-splines and penalized likelihood. *Statistical Science*, 11(2):89–121, 1996.

Appendix	
$L(a,b)$	$\lvert a-b\rvert$
$g_f(\cdot)$	$(2s)^{-1}(L(\cdot,f) - L(\cdot,f_n^{(s)}))$, a scaled empirical process
$\mathcal{G}$	$\{g_f : f \in \mathcal{F}_n(s)\}$
T_X	The empirical measure of a training set
$\lVert f\rVert_{L_2(T_X)}$	$\big(\frac{1}{n}\sum_{i=1}^n \lvert f(x_i,y_i)\rvert^2\big)^{1/2}$
$\mathcal{M}$	A η-net with respect to the $\lVert\cdot\rVert_{L_2(T_X)}$ distance
$N(\eta, \mathcal{F}, L_2(T_X))$	The η-covering number of $\mathcal{F}$ with respect to the $\lVert\cdot\rVert_{L_2(T_X)}$ distance
M	$\sqrt{2}n^{-1/2}\log(n)$
$\eta_{n,0}$	A number depending on n, chosen to satisfy (9)
C_0	2^{10}, a large constant
$P_n g_f$	Defined as $P_n g_f = \frac{1}{n}\sum_{i=1}^n g_f(y_i)$
Pg_f	$\int g_f$
P^*	The outer probability
N	$N = 2n$
$(W_1, \ldots, W_N)$	A permutation of $1, \ldots, N$ whose distribution is uniform
$\delta_{(X)_i}$	Dirac measure at the observation X_i
$\tilde{P}_{n,N}$	$\frac{1}{n}\sum_{i=1}^n \delta_{(X)_{W_i}}$
P_N	$\frac{1}{N}\sum_{i=1}^N \delta_{(X)_i}$
$P_{\lvert N}$	The conditional probability given N observations
T	A positive integer
$\eta_1, \ldots, \eta_T$	A sequence of T positive numbers
$\mathcal{G}_q$	η_q-net of $\mathcal{G}$
$\pi_q g$	The projection of g on $\mathcal{G}_q$
P_1, P_2, P_3	Three probabilities to be bounded
χ	A number such that $T\chi \le M/16$
σ_N^2	Defined as $\sigma_N^2 = P_N(\pi_q g_f - \pi_{q-1} g_f)^2$
$\mathcal{G}_{\mathcal{H},b}$	$\{(2s)^{-1}L(\cdot,f) : f \in \mathcal{F}_n(s)\}$
$\mathcal{G}_{\mathcal{H}}$	$\{(2s)^{-1}L(\cdot,f') : f' = \sum_{i=1}^n \alpha_i K(x_i,\cdot), \sum_{i=1}^n \lvert\alpha_i\rvert \le s\}$
$\mathcal{G}'_{\mathcal{H}}$	$\{(2s)^{-1}f' : f' = \sum_{i=1}^n \alpha_i K(x_i,\cdot), \sum_{i=1}^n \lvert\alpha_i\rvert \le s\}$
T'_X	The empirical measure of the set $(x_1, \ldots, x_n)$
$B_{\mathcal{H}}$	The unit ball in $\mathcal{H}$
$\mathcal{G}'_{\mathcal{H},b}$	$\{(2s)^{-1}f : f \in \mathcal{F}_n(s)\}$
$\mathcal{G}''$	A minimal $\eta/2$-net of $\mathcal{G}'_{\mathcal{H}}$

Table 13: Important notation introduced in the Appendix (Part 1).

P. H. C. Eilers and B. D. Marx. Splines, knots, and penalties. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(6):637–653, 2010.

J. Fan and R. Li. Statistical challenges with high dimensionality: feature selection in knowledge discovery. *Proceedings of the International Congress of Mathematicians*, 3:595–622, 2006.

R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

J. H. Friedman and B. W. Silverman. Flexible parsimonious smoothing and additive modeling. *Technometrics*, 31:3–21, 1989.

Appendix	
ζ	Upper bound on f_0. Assumed in the proof of Corollary 5.
t^*	$t^* = \Phi_\varepsilon^{-1}\left(0.5+0.5(1-\delta/2)^{1/n}\right)$
s^*	$s^* = s+\zeta+t^*$
M^*	$M^* = \sqrt{2}n^{-1/2}\log(n)/t^*$
$V(x)$	$C\int_D \exp(\frac{\lvert x-x'\rvert^2}{2\sigma^2}) f_{\text{true}}(x')dx'$
C	A constant such that $V(x)$ can be used to estimate f_{true}
P'	The measure of a spherical Gaussian in D
$\bar{f}_{\text{true}}$	A piecewise constant function used to approximate f_{true}
$\sigma = (\sigma_i;\ i = 1,\ldots,n)$	A set of n Rademacher random variables, where $P(\sigma_i = 1) = 1/2$ and $P(\sigma_i = -1) = 1/2$
S	A sample of $(x_1,y_1),\ldots,(x_n,y_n)$ *i.i.d.* from the joint distribution of X and Y
$\hat{R}_n(\mathcal{F}_n(s))$	$E_\sigma[\sup_{f\in\mathcal{F}_n(s)} \frac{1}{n}\sum_{i=1}^n \sigma_i\rho_\tau(Y-f(X))]$
$R_n(\mathcal{F}_n(s))$	$E_S\hat{R}_n(\mathcal{F}_n(s))$
$T_n(\delta)$	$\max(\tau,1-\tau)\left(n^{-1}(2s^2+2t^2)\log(1/\delta)\right)^{1/2}$
$\phi(S)$	$\sup_{f\in\mathcal{F}_n(s)}[E\rho_\tau(Y-f(X)) - \frac{1}{n}\sum_{i=1}^n \rho_\tau(y_i-f(x_i))]$
$S^{(i,x)}$	$\{(x_1,y_1),\ldots,(x_i',y_i),\ldots,(x_n,y_n)\}$, the difference between $S^{(i,x)}$ and S is only on the x value of their i^{th} pair
$S^{(i,y)}$	$\{(x_1,y_1),\ldots,(x_i,y_i'),\ldots,(x_n,y_n)\}$
R	$\frac{\hat{R}_n\mathcal{F}_n(s)}{2s}$
$h_f(\cdot)$	$(2s)^{-1}\rho_\tau(f-\cdot)$
$\mathcal{H}_R$	$\{h_f : f\in\mathcal{F}_n(s)\}$
$\mathcal{Y}$	The smallest η-net on $\mathcal{H}_R$
$g_{\mathcal{Y}}$	The projection of f on $\mathcal{Y}$
R_1	$\frac{1}{2s}E_\sigma[\sup_{g_{\mathcal{Y}}}\frac{1}{n}\sum_{i=1}^n \sigma_i\rho_\tau(g_{\mathcal{Y}}(x_i)-y_i)]$
R_2	$\frac{1}{2s}E_\sigma[\sup_{f,g_{\mathcal{Y}}\text{ close}}\frac{1}{n}\sum_{i=1}^n \sigma_i(\rho_\tau(f(x_i)-y_i)-\rho_\tau(g_{\mathcal{Y}}(x_i)-y_i))]$
Λ	$(\Lambda_1,\ldots,\Lambda_{2n})$, convex combination parameters of any element in $\mathcal{G}'_{\mathcal{H}}$
$F_1,\ldots,F_k$	*i.i.d.* random elements with $\text{pr}(F_1 = e_i) = \Lambda_i;\ i = 1,\ldots,2n$
$\bar{F}$	The average of $F_1,\ldots,F_k$

Table 14: Important notation introduced in the Appendix (Part 2)

D. Gervini. Free-knot spline smoothing for functional data. *Journal of the Royal Statistical Society: Series B*, 68(4):671–687, 2006.

C. Gu. *Smoothing Spline ANOVA Models*. Springer-Verlag, 2002.

M. H. Hansen and C. Kooperberg. Spline adaptation in extended linear models. *Statistical Science*, 17(1):2–51, 2002.

T. J. Hastie, R. J. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, 2009.

X. He, P. Ng, and S. Portnoy. Bivariate quantile smoothing splines. *Journal of the Royal Statistical Society: Series B*, 60(3):537–550, 1998.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *Annals of Statistics*, 36(3):1171–1220, 2008.

S. S. Keerthi and C.-J. Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, 15(7):1667–1689, 2003.

G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971.

R. Koenker and G. Bassett. Regression quantiles. *Econometrica*, 46:33–50, 1978.

R. Koenker and O. Geling. Reappraising medfly longevity: a quantile regression survival analysis. *Journal of the American Statistical Association*, 96(454):458–468, 2001.

R. Koenker and K. Hallock. Quantile regression: an introduction. *Journal of Economic Perspectives*, 15(4):43–56, 2001.

R. Koenker, P. Ng, and S. Portnoy. Quantile smoothing splines. *Biometrika*, 81:673–680, 1994.

V. Koltchinskii. Local Rademacher complexities and oracle inequalities in risk minimization. *Annals of Statistics*, 34(6):2593–2656, 2006.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 2002.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*, volume 23. Springer, 1991.

Y. Li and J. Zhu. L1-norm quantile regression. *Journal of Computational and Graphical Statistics*, 17:163–185, 2008.

Y. Li, Y. Liu, and J. Zhu. Quantile regression in reproducing kernel Hilbert spaces. *Journal of the American Statistical Association*, 102(477):255–268, 2007.

W. Mao and L. H. Zhao. Free-knot polynomial splines with confidence intervals. *Journal of the Royal Statistical Society: Series B*, 65(4):901–919, 2003.

C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

S. Mendelson. A few notes on statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 1–40. Springer, 2003.

H. Q. Minh. Some properties of Gaussian reproducing kernel Hilbert spaces and their implications for function approximation and learning theory. *Constructive Approximation*, 32(2):307–338, 2010.

S. Miyata and X. Shen. Free-knot splines and adaptive knot selection. *Journal of the Japan Statistical Society*, 35(2):303–324, 2005.

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. MIT press, 2012.

D. Nychka, G. Gray, P. Haaland, D. Martin, and M. OConnell. A nonparametric regression approach to syringe grading for quality improvement. *Journal of the American Statistical Association*, 90: 1171–1178, 1995.

V. I. Paulsen. An introduction to the theory of reproducing kernel Hilbert spaces. 2009. Technical report.

S. Rosset. Bi-level path following for cross validated solution of kernel quantile regression. *Journal of Machine Learning Research*, 10:2473–2505, 2009.

D. Ruppert. Selecting the number of knots for penalized splines. *Journal of Computational and Graphical Statistics*, 11:735–757, 2002.

D. Ruppert, M. P. Wand, and R. J. Carroll. *Semiparametric Regression*. Cambridge University Press, 2003.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. MIT Press, 2002.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

S. Smale and D.-X. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1(1):1–25, 2003.

A. J. Smola and B. Schölkopf. On a kernel-based method for pattern recognition, regression, approximation, and operator inversion. *Algorithmica*, 22(1-2):211–231, 1998.

A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2002.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, 2008.

I. Steinwart and C. Scovel. Fast rates for support vector machines using Gaussian kernels. *Annals of Statistics*, 35(2):575–607, 2007.

I. Steinwart, D. Hush, and C. Scovel. An explicit description of the reproducing kernel Hilbert spaces of Gaussian RBF kernels. *IEEE Trans. Inform. Theory*, 52:4635–4643, 2006.

M. Stitson, A. Gammerman, V. Vapnik, V. Vovk, C. Watkins, and J. Weston. Support vector regression with ANOVA decomposition kernels. *Advances in Kernel Methods–Support Vector Learning*, pages 285–292, 1999.

I. Takeuchi, Q. V. Le, T. D. Sears, and A. J. Smola. Nonparametric quantile estimation. *Journal of Machine Learning Research*, 7:1231–1264, 2006.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

A. W. Van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes with Application to Statistics*. Springer, 2000.

V. Vapnik, S. E. Golowich, and A. J. Smola. Support vector method for function approximation, regression estimation, and signal processing. *Advances in Neural Information Processing Systems*, pages 281–287, 1997.

G. Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990.

G. Wahba. Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. In *Advances in Kernel Methods Support Vector Learning*, pages 69–88. MIT Press, 1999.

H. Wang and X. He. Detecting differential expressions in genechip microarray studies: a quantile approach. *Journal of the American Statistical Association*, 102(477):104–112, 2007.

L. Wang and X. Shen. On l1-norm multi-class support vector machines: methodology and theory. *Journal of the American Statistical Association*, 102:595–602, 2007.

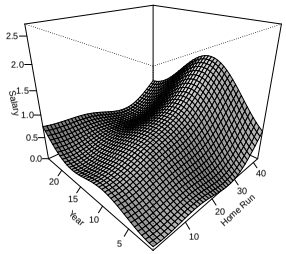
Y. Wei and X. He. Conditional growth charts. *Annals of Statistics*, 34(5):2069–2097, 2006.

M. Yuan. GACV for quantile smoothing splines. *Computational Statistics and Data Analysis*, 5: 813–829, 2006.

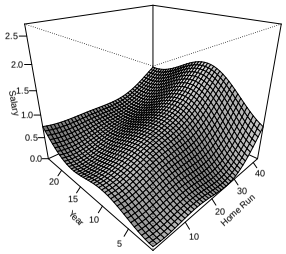
D.-X. Zhou. The covering number in learning theory. *Journal of Complexity*, 18(3):739–767, 2002.

S. Zhou, X. Shen, and D. A. Wolfe. Local asymptotics for regression splines and confidence regions. *Annals of Statistics*, 26(5):1760–1782, 1998.

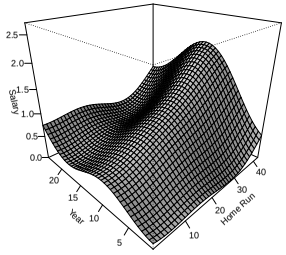
H. Zou and T. J. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67(2):301–320, 2005.



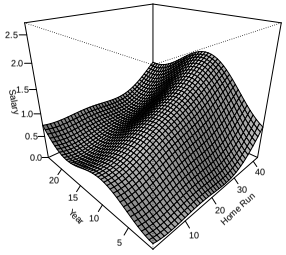
(a) Squared norm penalty with $\tau = 0.25$.



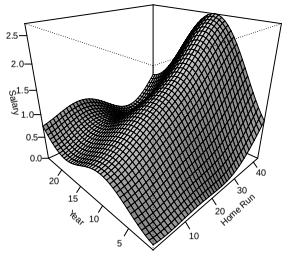
(b) Data sparsity constraint with $\tau = 0.25$.



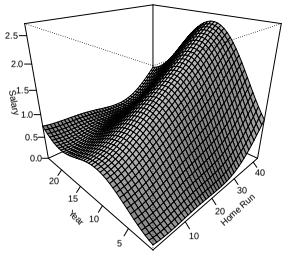
(c) Squared norm penalty with $\tau = 0.5$.



(d) Data sparsity constraint with $\tau = 0.5$.



(e) Squared norm penalty with $\tau = 0.75$.



(f) Data sparsity constraint with $\tau = 0.75$.

Figure 7: Estimated salary for the Baseball data using the number of home runs and the number of years played as the predictors.

Journal of Machine Learning Research 17 (2016) 1-6 Submitted 8/14; Revised 6/15; Published 4/16

BayesPy: Variational Bayesian Inference in Python

Jaakko Luttinen jaakko.luttinen@iki.fi
Department of Computer Science
Aalto University, Finland

Editor: Geoff Holmes

Abstract

BayesPy is an open-source Python software package for performing variational Bayesian inference. It is based on the variational message passing framework and supports conjugate exponential family models. By removing the tedious task of implementing the variational Bayesian update equations, the user can construct models faster and in a less error-prone way. Simple syntax, flexible model construction and efficient inference make BayesPy suitable for both average and expert Bayesian users. It also supports some advanced methods such as stochastic and collapsed variational inference.

Keywords: variational Bayes, probabilistic programming, Python

1. Introduction

Bayesian framework provides a theoretically solid and consistent way to construct models and perform inference. In practice, however, the inference is usually analytically intractable and is therefore based on approximation methods such as variational Bayes (VB), expectation propagation (EP) and Markov chain Monte Carlo (MCMC) (Bishop, 2006). Deriving and implementing the formulas for an approximation method is often straightforward but tedious, time consuming and error prone.

BayesPy is a Python 3 package providing tools for constructing conjugate exponential family models and performing VB inference easily and efficiently. It is based on the variational message passing (VMP) framework which defines a simple message passing protocol (Winn and Bishop, 2005). This enables implementation of small general nodes that can be used as building blocks for large complex models. BayesPy offers a comprehensive collection of built-in nodes that can be used to build a wide range of models and a simple interface for implementing custom nodes. The package is released under the MIT license.

Several other projects have similar goals for making Bayesian inference easier and faster to apply. VB inference is available in Bayes Blocks (Raiko et al., 2007), VIBES (Bishop et al., 2002) and Infer.NET (Minka et al., 2014). Bayes Blocks is an open-source C++/Python package but limited to scalar Gaussian nodes and a few deterministic functions, thus making it very limited. VIBES is an old software package for Java, released under the revised BSD license, but it is no longer actively developed. VIBES has been replaced by Infer.NET, which is partly closed source and licensed for non-commercial use only. Instead of VB inference, mainly MCMC is supported by other projects such as PyMC (Patil et al., 2010), OpenBUGS (Thomas et al., 2006), Dimple (Hershey et al., 2012) and Stan (Stan Development Team, 2014). Thus, there is a need for an open-source and maintained VB software package.

2. Features

BayesPy can be used to construct conjugate exponential family models. The documentation provides detailed examples of how to construct a variety of models, including principal component analysis models, linear state-space models, mixture models and hidden Markov models. BayesPy has also been used in two publications about parameter expansion and time-varying dynamics for linear state-space models (Luttinen, 2013; Luttinen et al., 2014).

Using BayesPy for Bayesian inference consists of four main steps: constructing the model, providing data, finding the posterior approximation and examining the results. The user constructs the model from small modular blocks called nodes. Roughly, each node corresponds to a latent variable, a set of observations or a deterministic function. The inference engine is used to run the message passing algorithm in order to obtain the posterior approximation. The resulting posterior can be examined, for instance, by using a few built-in plotting functions or printing the parameters of the posterior distributions.

Nodes are the primary building blocks for models in BayesPy. There are two types of nodes: stochastic and deterministic. Stochastic nodes correspond to probability distributions and deterministic nodes correspond to functions. Built-in stochastic nodes include all common exponential family distributions (e.g., Gaussian, gamma and Dirichlet distributions), a general mixture distribution and a few complex nodes for dynamic variables (e.g., discrete and Gaussian Markov chains). Built-in deterministic nodes include a gating node and a general sum-product node. In case a model cannot be constructed using the built-in nodes, the documentation provides instructions for implementing new nodes.

BayesPy is designed to be simple enough for non-expert users but flexible and efficient enough for expert users. One goal is to keep the syntax easy and intuitive to read and write by making it similar to the mathematical formulation of the model. Missing values are easy to handle and the variables can be monitored by plotting the posterior distributions during the learning. BayesPy has also preliminary support for some advanced VB methods such as stochastic variational inference (Hoffman et al., 2013), deterministic annealing (Katahira et al., 2008), collapsed inference (Hensman et al., 2012), Riemannian conjugate gradient learning (Honkela et al., 2010), parameter expansions (Qi and Jaakkola, 2007) and pattern searches (Honkela et al., 2003). For developers, the unit testing framework helps in finding bugs, making changes and implementing new features in a robust manner.

BayesPy can be installed similarly to other Python packages. It requires Python 3 and a few popular packages: NumPy, SciPy, matplotlib and h5py. The latest release can be installed from Python Package Index (PyPI) and detailed instructions can be found from the comprehensive online documentation[1]. The latest development version is available at GitHub[2], which is also the platform used for reporting bugs and making pull requests.

3. Example

This section demonstrates the key steps in using BayesPy. An artificial Gaussian mixture dataset is created by drawing 500 samples from two 2-dimensional Gaussian distributions. 200 samples have mean $[2, 2]$ and 300 samples have mean $[0, 0]$:

1. `http://bayespy.org`
2. `https://github.com/bayespy/bayespy`

```
import numpy as np
N = 500; D = 2
data = np.random.randn(N, D)
data[:200,:] += 2*np.ones(D)
```

We construct a mixture model for the data and assume that the parameters, the cluster assignments and the true number of clusters are unknown. The model uses a maximum number of five clusters but the effective number of clusters will be determined automatically:

```
K = 5
```

The unknown cluster means and precision matrices are given Gaussian and Wishart prior distributions:

```
from bayespy import nodes
mu = nodes.Gaussian(np.zeros(D), 0.01*np.identity(D), plates=(K,))
Lambda = nodes.Wishart(D, D*np.identity(D), plates=(K,))
```

The `plates` keyword argument is used to define repetitions similarly to the plate notation in graphical models. The cluster assignments are categorical variables, and the cluster probabilities are given a Dirichlet prior distribution:

```
alpha = nodes.Dirichlet(0.01*np.ones(K))
z = nodes.Categorical(alpha, plates=(N,))
```

The observations are from a Gaussian mixture distribution:

```
y = nodes.Mixture(z, nodes.Gaussian, mu, Lambda)
```

The second argument for the `Mixture` node defines the type of the mixture distribution, in this case Gaussian. The variable is marked as observed by providing the data:

```
y.observe(data)
```

Next, we want to find the posterior approximation for our latent variables. We create the variational Bayesian inference engine:

```
from bayespy.inference import VB
Q = VB(y, mu, z, Lambda, alpha)
```

Before running the VMP algorithm, the symmetry in the model is broken by a random initialization of the cluster assignments:

```
z.initialize_from_random()
```

Without the random initialization, the clusters would not be separated. The VMP algorithm updates the variables in turns and is run for 200 iterations or until convergence:

```
Q.update(repeat=200)
```

The results can be examined visually by using `bayespy.plot` module:

```
import bayespy.plot as bpplt
bpplt.gaussian_mixture_2d(y, alpha=alpha)
bpplt.pyplot.show()
```

It is also possible to print the parameters of the approximate posterior distributions:

```
print(alpha)
```

The `bayespy.plot` module contains also other functions for visually examining the distributions.

	Small GMM	Large GMM	Small PCA	Large PCA
BayesPy	6	90	7 (60)	400 (1 500)
Infer.NET	25	4 600	37 (350)	2 200 (210 000)

Table 1: The average CPU time in milliseconds per iteration for each dataset. The results in parentheses have the following meanings: a) BayesPy without using broadcasting. b) Infer.NET using the same factorization as BayesPy.

4. Comparison with Infer.NET

This section provides a brief comparison with Infer.NET because it is another active project implementing the variational message passing algorithm, whereas the other related active projects implement MCMC. The main advantages of Infer.NET over BayesPy are its support for non-conjugate models (e.g., logistic regression) and other inference engines (EP and Gibbs sampling). On the other hand, BayesPy is open-source software and supports several advanced VB methods (e.g., stochastic variational inference and collapsed inference).

The speed of the packages were compared by using two widely used models: a Gaussian mixture model (GMM) and principal component analysis (PCA).[3] Both models were run for small and large artificial datasets. For GMM, the small model used 10 clusters for 200 observations with 2 dimensions, and the large model used 40 clusters for 2000 observations with 10 dimensions. For PCA, the small model used 10-dimensional latent space for 500 observations with 20 dimensions, and the large model used 40-dimensional latent space for 2000 observations with 100 dimensions. For the PCA model, Infer.NET used both a fully factorizing approximation and also the same approximation as BayesPy which did not factorize with respect to the latent space dimensions. The experiments were run on a quad-core (i7-4702MQ) Linux computer. Because the packages run practically identical algorithms and thus converged to similar solutions in a similar number of iterations (50–100 iterations depending on the dataset), we compared the average CPU time per iteration.

The results are summarized in Table 1. For all datasets, BayesPy is faster probably because it uses highly optimized numerical libraries (BLAS and LAPACK). For PCA, BayesPy also automatically avoids redundant computations which arise because latent variables have equal posterior covariance matrices. However, such broadcasting is not always possible (e.g., if there are missing values in the PCA data). Thus, the table also presents the results when BayesPy is forced to not use broadcasting: BayesPy is slower than Infer.NET on the smaller PCA dataset but faster on the larger PCA dataset. If Infer.NET used the same factorization for PCA as BayesPy, Infer.NET may be orders of magnitude slower.

5. Conclusions

BayesPy provides a simple and efficient way to construct conjugate exponential family models and to find the variational Bayesian posterior approximation in Python. In addition to the standard variational message passing, it supports several advanced methods such

3. The scripts for running the experiments are available as supplementary material.

as stochastic and collapsed variational inference. Future plans include support for non-conjugate models and non-parametric models (e.g., Gaussian and Dirichlet processes).

References

C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, 2nd edition, 2006.

C. M. Bishop, D. Spiegelhalter, and J. Winn. VIBES: a variational inference engine for Bayesian networks. In *Advances in Neural Information Processing Systems 15*, pages 777–784, 2002.

J. Hensman, M. Rattray, and N. D. Lawrence. Fast variational inference in the conjugate exponential family. In *Advances in Neural Information Processing Systems 25*, pages 2888–2896, 2012.

S. Hershey, J. Bernstein, B. Bradley, A. Schweitzer, N. Stein, T. Weber, and B. Vigoda. Accelerating inference: towards a full language, compiler and hardware stack. *Computing Research Repository*, 2012.

M. D. Hoffman, D. M. Blei Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–47, 2013.

A. Honkela, H. Valpola, and J. Karhunen. Accelerating cyclic update algorithms for parameter estimation by pattern searches. *Neural Processing Letters*, 17(2):191–203, 2003.

A. Honkela, T. Raiko, M. Kuusela, M. Tornio, and J. Karhunen. Approximate Riemannian conjugate gradient learning for fixed-form variational Bayes. *Journal of Machine Learning Research*, 11: 3235–3268, 2010.

K. Katahira, K. Watanabe, and M. Okada. Deterministic annealing variant of variational Bayes method. *Journal of Physics: Conference Series*, 95(1), 2008.

J. Luttinen. Fast variational Bayesian linear state-space model. In *Machine Learning and Knowledge Discovery in Databases*, volume 8188 of *Lecture Notes in Computer Science*, pages 305–320. Springer, 2013.

J. Luttinen, T. Raiko, and A. Ilin. Linear state-space model with time-varying dynamics. In *Machine Learning and Knowledge Discovery in Databases*, volume 8725 of *Lecture Notes in Computer Science*, pages 338–353. Springer, 2014.

T. Minka, J. Winn, J. Guiver, and D. A. Knowles. Infer.NET 2.6, 2014. URL `http://research.microsoft.com/infernet`. Microsoft Research Cambridge.

A. Patil, D. Huard, and C. J. Fonnesbeck. PyMC: Bayesian stochastic modelling in Python. *Journal of Statistical Software*, 35(4):1–81, 2010.

Y. Qi and T. S. Jaakkola. Parameter expanded variational Bayesian methods. In *Advances in Neural Information Processing Systems 19*, pages 1097–1104, Vancouver, Canada, 2007.

T. Raiko, H. Valpola, M. Harva, and J. Karhunen. Building blocks for variational Bayesian learning of latent variable models. *Journal of Machine Learning Research*, 2007.

Stan Development Team. Stan: A C++ library for probability and sampling, version 2.4, 2014. URL `http://mc-stan.org/`.

A. Thomas, B. O'Hara, U. Ligges, and S. Sibylle. Making BUGS open. *R News*, 6(1):12–17, 2006.

J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6: 661–694, 2005.

Journal of Machine Learning Research 17 (2016) 1-62 Submitted 9/14; Revised 7/15; Published 4/16

Variational Inference for Latent Variables and Uncertain Inputs in Gaussian Processes

Andreas C. Damianou* ANDREAS.DAMIANOU@SHEFFIELD.AC.UK
Dept. of Computer Science and Sheffield Institute for Translational Neuroscience
University of Sheffield
UK

Michalis K. Titsias* MTITSIAS@AUEB.GR
Department of Informatics
Athens University of Economics and Business
Greece

Neil D. Lawrence N.LAWRENCE@DCS.SHEFFIELD.AC.UK
Dept. of Computer Science and Sheffield Institute for Translational Neuroscience
University of Sheffield
UK

Editor: Amos Storkey

Abstract

The Gaussian process latent variable model (GP-LVM) provides a flexible approach for non-linear dimensionality reduction that has been widely applied. However, the current approach for training GP-LVMs is based on maximum likelihood, where the latent projection variables are maximised over rather than integrated out. In this paper we present a Bayesian method for training GP-LVMs by introducing a non-standard variational inference framework that allows to approximately integrate out the latent variables and subsequently train a GP-LVM by maximising an analytic lower bound on the exact marginal likelihood. We apply this method for learning a GP-LVM from i.i.d. observations and for learning non-linear dynamical systems where the observations are temporally correlated. We show that a benefit of the variational Bayesian procedure is its robustness to overfitting and its ability to automatically select the dimensionality of the non-linear latent space. The resulting framework is generic, flexible and easy to extend for other purposes, such as Gaussian process regression with uncertain or partially missing inputs. We demonstrate our method on synthetic data and standard machine learning benchmarks, as well as challenging real world datasets, including high resolution video data.

Keywords: Gaussian processes, variational inference, latent variable models, dynamical systems, uncertain inputs

1. Introduction

Consider a non linear function, $f(x)$. A very general class of probability densities can be recovered by mapping a simpler density through the non linear function. For example, we might decide that x should be drawn from a Gaussian density

$$x \sim \mathcal{N}(0, 1)$$

and we observe y, which is given by passing samples from x through a non linear function, perhaps with some corrupting noise

$$y = f(x) + \epsilon, \tag{1}$$

where ϵ could also be drawn from a Gaussian density

$$\epsilon \sim \mathcal{N}\left(0, \sigma^2\right),$$

this time with variance σ^2. Whilst the resulting density for y, denoted by $p(y)$, can now have a very general form, these models present particular problems in terms of tractability.

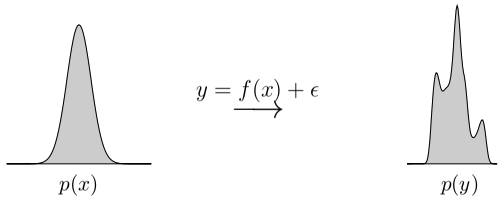


Figure 1: A Gaussian distribution propagated through a non-linear mapping. $y_i = f(x_i) + \epsilon_i$. $\epsilon \sim \mathcal{N}\left(0, 0.2^2\right)$ and $f(\cdot)$ uses RBF basis, 100 centres between -4 and 4 and $\ell = 0.1$. The new distribution over y (right) is multimodal and difficult to normalize.

Models of this form appear in several domains. They can be used for nonlinear dimensionality reduction (MacKay, 1995; Bishop et al., 1998) where several latent variables, $\mathbf{x} = \{x_j\}_{j=1}^q$ are used to represent a high dimensional vector $\mathbf{y} = \{y_j\}_{j=1}^p$ and we normally have $p > q$,

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) + \boldsymbol{\epsilon}.$$

They can also be used for prediction of a regression model output when the input is uncertain (see e.g. Oakley and O'Hagan, 2002) or for autoregressive prediction in time series (see e.g. Girard et al., 2003). Further, by adding a dynamical autoregressive component to the non-linear dimensionality reduction approaches leads to non-linear state space models (Särkkä, 2013), where the states often have a physical interpretation and are propagated through time in an autoregressive manner:

$$\mathbf{x}_t = \mathbf{g}(\mathbf{x}_{t-1}),$$

where $\mathbf{g}(\cdot)$ is a vector valued function. The observations are then observed through a separate nonlinear vector valued function,

$$\mathbf{y}_t = \mathbf{f}(\mathbf{x}_t) + \boldsymbol{\epsilon}.$$

∗. These authors contributed equally to this work.

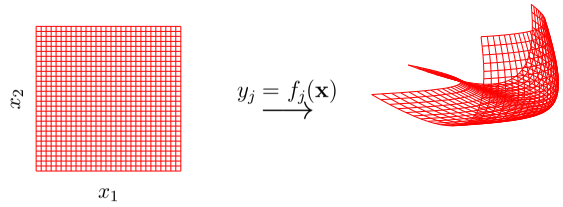


Figure 2: A three dimensional manifold formed by mapping from a two dimensional space to a three dimensional space.

The intractabilities of mapping a distribution through a non-linear function have resulted in a range of different approaches. In density networks sampling was proposed; in particular, in (MacKay, 1995) *importance sampling* was used. When extending importance samplers dynamically, the degeneracy in the weights needs to be avoided, thus leading to the resampling approach suggested for the bootstrap particle filter of Gordon et al. (1993). Other approaches in non-linear state space models include the Laplace approximation as used in extended Kalman filters and unscented and ensemble transforms (see Särkkä, 2013). In dimensionality reduction the generative topographic mapping (GTM Bishop et al., 1998) reinterpreted the importance sampling approach of MacKay (1995) as a mixture model, using a discrete representation of the latent space.

In this paper we suggest a variational approach to dealing with latent variables and input uncertainty that can be applied to Gaussian process models. Gaussian processes provide a probabilistic framework for performing inference over functions. A Gaussian process prior can be combined with a data set (through an appropriate likelihood) to obtain a posterior process that represents all functions that are consistent with the data and our prior.

Our initial focus will be application of Gaussian process models in the context of dimensionality reduction. In dimensionality reduction we assume that our high dimensional data set is really the result of some low dimensional control signals which are, perhaps, non-linearly related to our observed functions. In other words we assume that our data, $\mathbf{Y} \in \Re^{n\times p}$, can be generated by a lower dimensional matrix, $\mathbf{X} \in \Re^{n\times q}$ through a vector valued function where each row, $\mathbf{y}_{i,:}$ of $\mathbf{Y}$ represents an observed data point and is generated through

$$\mathbf{y}_{i,:} = \mathbf{f}(\mathbf{x}_{i,:}) + \boldsymbol{\epsilon}_{i,:},$$

so that the data is a lower dimensional subspace immersed in the original, high dimensional space. If the mapping is linear, e.g. $\mathbf{f}(\mathbf{x}_{i,:}) = \mathbf{W}\mathbf{x}_{i,:}$ with $\mathbf{W} \in \Re^{q\times p}$, methods like principal component analysis, factor analysis and (for non-Gaussian $p(\mathbf{x}_{i,:})$) independent component analysis (Hyvärinen et al., 2001) follow. For Gaussian $p(\mathbf{x}_{i,:})$ the marginalisation of the latent variable is tractable because placing a Gaussian density through an affine transformation retains the Gaussianity of the data density, $p(\mathbf{y}_{i,:})$. However, the linear assumption is very restrictive so it is natural to aim to go beyond it through a non linear mapping.

In the context of dimensionality reduction a range of approaches have been suggested that consider neighbourhood structures or the preservation of local distances to find a low dimensional representation. In the machine learning community, spectral methods such as isomap (Tenenbaum et al., 2000), locally linear embeddings (LLE, Roweis and Saul, 2000) and Laplacian eigenmaps (Belkin and Niyogi, 2003) have attracted a lot of attention. These spectral approaches are all closely related to kernel PCA (Schölkopf et al., 1998) and classical multi-dimensional scaling (MDS) (see e.g. Mardia et al., 1979). These methods do have a probabilistic interpretation as described by Lawrence (2012) which, however, does not explicitly include an assumption of underlying reduced data dimensionality. Other iterative methods such as metric and non-metric approaches to MDS (Mardia et al., 1979), Sammon mappings (Sammon, 1969) and t-SNE (van der Maaten and Hinton, 2008) also lack an underlying generative model.

Probabilistic approaches, such as the generative topographic mapping (GTM, Bishop et al., 1998) and density networks (MacKay, 1995), view the dimensionality reduction problem from a different perspective, since they seek a mapping from a low-dimensional latent space to the observed data space (as illustrated in Figure 2), and come with certain advantages. More precisely, their generative nature and the forward mapping that they define, allows them to be extended more easily in various ways (e.g. with additional dynamics modelling), to be incorporated into a Bayesian framework for parameter learning and to handle missing data. This approach to dimensionality reduction provides a useful archetype for the algorithmic solutions we are providing in this paper, as they require approximations that allow latent variables to be propagated through a non-linear function.

Our framework takes the generative approach prescribed by density networks and the non-linear variants of Kalman filters one step further. Because, rather than considering a specific function, $f(\cdot)$, to map from the latent variables to the data space, we will consider an entire family of functions. One that subsumes the more restricted class of either Gauss Markov processes (such as the *linear* Kalman filter/smoother) and Bayesian basis function models (such as the RBF network used in the GTM, with a Gaussian prior over the basis function weightings). These models can all be cast within the framework of Gaussian processes (Rasmussen and Williams, 2006). Gaussian processes are probabilistic kernel methods, where the kernel has an interpretation of a covariance associated with a prior density. This covariance specifies a distribution over functions that subsumes the special cases mentioned above.

The Gaussian process latent variable model (GP-LVM, Lawrence, 2005) is a more recent probabilistic dimensionality reduction method which has been proven to be very useful for high dimensional problems (Lawrence, 2007; Damianou et al., 2011). GP-LVM can be seen as a non-linear generalisation of probabilistic PCA (PPCA, Tipping and Bishop, 1999; Roweis, 1998), which also has a Bayesian interpretation (Bishop, 1999). In contrast to PPCA, the non-linear mapping of GP-LVM makes a Bayesian treatment much more challenging. Therefore, GP-LVM itself and all of its extensions, rely on a maximum a posteriori (MAP) training procedure. However, a principled Bayesian formulation is highly desirable, since it would allow for robust training of the model, automatic selection of the latent space's dimensionality as well as more intuitive exploration of the latent space's structure.

In this paper we formulate a variational inference framework which allows us to propagate uncertainty through a Gaussian process and obtain a rigorous lower bound on the marginal likelihood of the resulting model. The procedure followed here is non-standard, as computation of a closed-form Jensen's lower bound on the true log marginal likelihood of the data is infeasible with classical approaches to variational inference. Instead, we build on, and significantly extend, the variational GP method of Titsias (2009), where the GP prior is augmented to include auxiliary inducing variables so that the approximation is applied on an expanded probability model. The resulting framework defines a bound on the evidence of the GP-LVM which, when optimised, gives as a by-product an approximation to the true posterior distribution of the latent variables given the data.

Considering a posterior distribution rather than point estimates for the latent points means that our framework is generic and can be easily extended for multiple practical scenarios. For example, if we treat the latent points as noisy measurements of given inputs we obtain a method for Gaussian process regression with uncertain inputs (Girard et al., 2003) or, in the limit, with partially observed inputs. On the other hand, considering a latent space prior that depends on a time vector, allows us to obtain a Bayesian model for dynamical systems (Damianou et al., 2011) that significantly extends classical Kalman filter models with a non-linear relationship between the state space, $\mathbf{X}$, and the observed data $\mathbf{Y}$, along with non-Markov assumptions in the latent space which can be based on continuous time observations. This is achieved by placing a Gaussian process prior on the latent space, $\mathbf{X}$ which is itself a function of time, t. This approach can itself be trivially further extended by replacing the time dependency of the prior for the latent space with a spatial dependency, or a dependency over an arbitrary number of high dimensional inputs. As long as a valid *covariance function*[1] can be derived (this is also possible for strings and graphs). This leads to a Bayesian approach for warped Gaussian process regression (Snelson et al., 2004; Lázaro-Gredilla, 2012).

In the next subsection we summarise the notation and conventions used in this paper. In Section 2 we review the main prior work on dealing with latent variables in the context of Gaussian processes and describe how the model was extended with a dynamical component. We then introduce the variational framework and Bayesian training procedure in Section 3. In Section 4 we describe how the variational approach is applied to a range of predictive tasks and this is demonstrated with experiments conducted on simulated and real world datasets in Section 5. In Section 6 we discuss and experimentally demonstrate natural but important extensions of our model, motivated by situations where the inputs to the GP are not fully unobserved. These extensions give rise to an auto-regressive variant for performing iterative future predictions, as well as a GP regression variant which can handle missing inputs. Finally, based on the theoretical and experimental results of our work, we present our final conclusions in Section 7. This article builds on and extends the previously published conference papers in (Titsias and Lawrence, 2010; Damianou et al., 2011).

1. The constraints for a valid covariance function are the same as those for a Mercer kernel. It must be a positive (semi) definite function over the space of all possible input pairs.

1.1 Notation

Throughout this paper we use capital boldface letters to denote matrices, lower-case boldface letters to denote vectors and lower-case letters to denote scalar quantities. We denote the ith row of the matrix $\mathbf{Y}$ as $\mathbf{y}_{i,:}$ and its jth column as $\mathbf{y}_{:,j}$, whereas $y_{i,j}$ denotes the scalar element found in the ith row and jth column of $\mathbf{Y}$. We assume that data points are stored by rows, so that the $p-$dimensional vector $\mathbf{y}_{i,:}$ corresponds to the ith data point. The collection of test variables (i.e. quantities given at test time for making predictions) is denoted using an asterisk, e.g. $\mathbf{Y}_*$ which has columns $\{\mathbf{y}_{*,j}\}_{j=1}^{p}$.

Concerning variables of interest, $\mathbf{Y}$ is the collection of observed outputs, $\mathbf{F}$ is the collection of latent GP function instantiations and $\mathbf{X}$ is the collection of latent inputs. Further on we will introduce auxiliary inputs denoted by $\mathbf{X}_u$, auxiliary function instantiations denoted by $\mathbf{U}$, a time vector denoted by $\mathbf{t}$, and arbitrary (potentially partially) observed inputs denoted by $\mathbf{Z}$.

If a function f follows a Gaussian process, we use k_f to denote its covariance function and $\mathbf{K}_{ff}$ to denote the covariance matrix obtained by evaluating k_f on all available training inputs. The notation $\boldsymbol{\theta}_f$ then refers to the hyperparameters of k_f.

2. Gaussian Processes with Latent Variables as Inputs

This section provides background material on current approaches for learning using Gaussian process latent variable models (GP-LVMs). Specifically, Section 2.1 specifies the general structure of such models, Section 2.2 reviews the standard GP-LVM for i.i.d. data as well as dynamic extensions suitable for sequence data. Finally, Section 2.3 discusses the drawbacks of MAP estimation over the latent variables which is currently the standard way to train GP-LVMs.

2.1 Gaussian Processes for Latent Mappings

The unified characteristic of all GP-LVM algorithms, as they were first introduced by Lawrence (2005, 2004), is the consideration of a Gaussian Process as a prior distribution for the mapping function $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_p(\mathbf{x}))$ so that

$$f_j(\mathbf{x}) \sim \mathcal{GP}(0, k_f(\mathbf{x}, \mathbf{x}')), \quad j = 1, \dots, p. \tag{2}$$

Here, the individual components of $\mathbf{f}(\mathbf{x})$ are taken to be independent draws from a Gaussian process with kernel or covariance function $k_f(\mathbf{x}, \mathbf{x}')$, which determines the properties of the latent mapping. As shown in (Lawrence, 2005) the use of a linear covariance function makes GP-LVM equivalent to traditional PPCA. On the the other hand, when non-linear covariance functions are considered the model is able to perfom non-linear dimensionality reduction. The non-linear covariance function considered in (Lawrence, 2005) is the exponentiated quadratic (RBF),

$$k_{f(\text{rbf})}(\mathbf{x}_{i,:}, \mathbf{x}_{k,:}) = \sigma^2_{\text{rbf}} \exp\left(-\frac{1}{2\ell^2}\sum_{j=1}^{q}(x_{i,j} - x_{k,j})^2\right), \tag{3}$$

which is infinitely many times differentiable and it uses a common lengthscale parameter for all latent dimensions. The above covariance function results in a non-linear but smooth

mapping from the latent to the data space. Parameters that appear in a covariance function, such as σ^2_{rbf} and ℓ^2, are often referred to as kernel hyperparameters and will be denoted by $\boldsymbol{\theta}_f$.

Given the independence assumption across dimensions in equation (2), the latent variables $\mathbf{F} \in \Re^{n\times p}$ (with columns $\{\mathbf{f}_{:,j}\}_{j=1}^p$), which have one-to-one correspondence with the data points $\mathbf{Y}$, follow the prior distribution $p(\mathbf{F}|\mathbf{X},\boldsymbol{\theta}_f) = \prod_{j=1}^p p(\mathbf{f}_{:,j}|\mathbf{X},\boldsymbol{\theta}_f)$, where $p(\mathbf{f}_{:,j}|\mathbf{X},\boldsymbol{\theta}_f)$ is given by

$$p(\mathbf{f}_{:,j}|\mathbf{X},\boldsymbol{\theta}_f) = \mathcal{N}\left(\mathbf{f}_{:,j}|\mathbf{0},\mathbf{K}_{ff}\right) = |2\pi\mathbf{K}_{ff}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\mathbf{f}_{:,j}^\top \mathbf{K}_{ff}^{-1}\mathbf{f}_{:,j}\right), \tag{4}$$

and where $\mathbf{K}_{ff} = k_f(\mathbf{X},\mathbf{X})$ is the covariance matrix defined by the kernel function k_f. The inputs $\mathbf{X}$ in this kernel matrix are latent random variables following a prior distribution $p(\mathbf{X}|\boldsymbol{\theta}_x)$ with hyperparameters $\boldsymbol{\theta}_x$. The structure of this prior can depend on the application at hand, such as on whether the observed data are i.i.d. or have a sequential dependence. For the remaining of this section we shall leave $p(\mathbf{X}|\boldsymbol{\theta}_x)$ unspecified so that to keep our discussion general while specific forms for it will be given in the next section.

Given the construction outlined above, the joint probability density over the observed data and all latent variables is written as follows:

$$\begin{aligned} p(\mathbf{Y},\mathbf{F},\mathbf{X}|\boldsymbol{\theta}_f,\boldsymbol{\theta}_x,\sigma^2) &= p(\mathbf{Y}|\mathbf{F},\sigma^2)p(\mathbf{F}|\mathbf{X},\boldsymbol{\theta}_f)p(\mathbf{X}|\boldsymbol{\theta}_x) \\ &= \prod_{j=1}^p p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j},\sigma^2)p(\mathbf{f}_{:,j}|\mathbf{X},\boldsymbol{\theta}_f)p(\mathbf{X}|\boldsymbol{\theta}_x), \end{aligned} \tag{5}$$

where the term

$$p(\mathbf{Y}|\mathbf{F},\sigma^2) = \prod_{j=1}^p \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{f}_{:,j},\sigma^2\mathbf{I}_n\right) \tag{6}$$

comes directly from the assumed noise model of equation (1) while $p(\mathbf{F}|\mathbf{X},\boldsymbol{\theta}_f)$ and $p(\mathbf{X}|\boldsymbol{\theta}_x)$ come from the GP and the latent space. As discussed in detail in Section 3.1, the interplay of the latent variables (i.e. the latent matrix $\mathbf{X}$ that is passed as input in the latent matrix $\mathbf{F}$) makes inference very challenging. However, when fixing $\mathbf{X}$ we can treat $\mathbf{F}$ analytically and marginalise it out as follows:

$$p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}) = \left(\int p\left(\mathbf{Y}|\mathbf{F}\right)p(\mathbf{F}|\mathbf{X})\mathrm{d}\mathbf{F}\right)p(\mathbf{X}),$$

where

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^p \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{0},\mathbf{K}_{ff}+\sigma^2\mathbf{I}_n\right).$$

Here (and for the remaining of the paper), we omit reference to the parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_f,\boldsymbol{\theta}_x,\sigma^2\}$ in order to simplify our notation. The above partial tractability of the model gives rise to a straightforward MAP training procedure where the latent inputs $\mathbf{X}$ are selected according to

$$\mathbf{X}_{\text{MAP}} = \arg\max_{\mathbf{X}} p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}).$$

This is the approach suggested by Lawrence (2005, 2006) and subsequently followed by other authors (Urtasun and Darrell, 2007; Ek et al., 2008; Ferris et al., 2007; Wang et al., 2008; Ko and Fox, 2009c; Fusi et al., 2013; Lu and Tang, 2014). Finally, notice that point estimates over the hyperparameters $\boldsymbol{\theta}$ can also be found by maximising the same objective function.

2.2 Different Latent Space Priors and GP-LVM Variants

Different GP-LVM algorithms can result by varying the structure of the prior distribution $p(\mathbf{X})$ over the latent inputs. The simplest case, which is suitable for i.i.d. observations, is obtained by selecting a fully factorised (across data points and dimensions) latent space prior:

$$p(\mathbf{X}) = \prod_{i=1}^n \mathcal{N}\left(\mathbf{x}_{i,:}|\mathbf{0},\mathbf{I}_q\right) = \prod_{i=1}^n\prod_{j=1}^q \mathcal{N}\left(x_{i,j}|0,1\right). \tag{7}$$

More structured latent space priors can also be used that could incorporate available information about the problem at hand. For example, Urtasun and Darrell (2007) add discriminative properties to the GP-LVM by considering priors which encapsulate class-label information. Other existing approaches in the literature seek to constrain the latent space via a smooth dynamical prior $p(\mathbf{X})$ so as to obtain a model for dynamical systems. For example, Wang et al. (2006, 2008) extend GP-LVM with a temporal prior which encapsulates the Markov property, resulting in an auto-regressive model. Ko and Fox (2009b, 2011) further extend these models for Bayesian filtering in a robotics setting, whereas Urtasun et al. (2006) consider this idea for tracking. In a similar direction, Lawrence and Moore (2007) consider an additional temporal model which employs a GP prior that is able to generate smooth paths in the latent space.

In this paper we shall focus on dynamical variants where the dynamics are regressive, as in (Lawrence and Moore, 2007). In this setting, the data are assumed to be a multivariate timeseries $\{\mathbf{y}_{i,:},t_i\}_{i=1}^n$ where $t_i \in \Re_+$ is the time at which the datapoint $\mathbf{y}_{i,:}$ is observed. A GP-LVM dynamical model is obtained by defining a temporal latent function $\mathbf{x}(t) = (x_1(t),\ldots,x_q(t))$ where the individual components are taken to be independent draws from a Gaussian process,

$$x_j(t) \sim \mathcal{GP}(0,k_x(t,t')), \quad j=1,\ldots,q,$$

where $k_x(t,t')$ is the covariance function. The datapoint $\mathbf{y}_{i,:}$ is assumed to be produced via the latent vector $\mathbf{x}_{i,:} = \mathbf{x}(t_i)$, as shown in Figure 3(c). All these latent vectors can be stored in the matrix $\mathbf{X}$ (exactly as in the i.i.d. data case) which now follows the correlated prior distribution,

$$p(\mathbf{X}|\mathbf{t}) = \prod_{j=1}^q p(\mathbf{x}_{:,j}|\mathbf{t}) = \prod_{j=1}^q \mathcal{N}\left(\mathbf{x}_{:,j}|\mathbf{0},\mathbf{K}_x\right),$$

where $\mathbf{K}_x = k_x(\mathbf{t},\mathbf{t})$ is the covariance matrix obtained by evaluating the covariance function k_x on the observed times $\mathbf{t}$. In contrast to the fully factorised prior in (7), the above prior couples all elements in each column of $\mathbf{X}$. The covariance function k_x has parameters $\boldsymbol{\theta}_x$ and determines the properties of each temporal function $x_j(t)$. For instance, the use of

an Ornstein-Uhlenbeck covariance function (Uhlenbeck and Ornstein, 1930) yields a Gauss-Markov process for $x_j(t)$, while the exponentiated quadratic covariance function gives rise to very smooth and non-Markovian process. The specific choices and forms of the covariance functions used in our experiments are discussed in Section 5.1.

2.3 Drawbacks of the MAP Training Procedure

Current GP-LVM based models found in the literature rely on MAP training procedures, discussed in Section 2.1, for optimising the latent inputs and the hyperparameters. However, this approach has several drawbacks. Firstly, the fact that it does not marginalise out the latent inputs implies that it could be sensitive to overfitting. Further, the MAP objective function cannot provide any insight for selecting the optimal number of latent dimensions, since it typically increases when more dimensions are added. This is why most existing GP-LVM algorithms require the latent dimensionality to be either set by hand or selected with cross-validation. The latter case renders the whole training computationally slow and, in practice, only a very limited subset of models can be explored in a reasonable time.

As another consequence of the above, the current GP-LVMs employ simple covariance functions (typically having a common lengthscale over the latent input dimensions as the one in equation (3)) while more complex covariance functions, that could help to automatically select the latent dimensionality, are not popular. Such a latter covariance function can be an exponentiated quadratic, as in (3), but with different lengthscale per input dimension,

$$k_{f(\text{ard})}\left(\mathbf{x}_{i,:}, \mathbf{x}_{k,:}\right) = \sigma^2_{\text{ard}} \exp\left(-\frac{1}{2}\sum_{j=1}^{q}\frac{\left(x_{i,j}-x_{k,j}\right)^2}{l_j^2}\right), \tag{8}$$

where the squared inverse lengthscale per dimension can be seen as a weight, i.e. $\frac{1}{l_j^2} = w_j$. This covariance function could thus allow an automatic relevance determination (ARD) procedure to take place, during which unnecessary dimensions of the latent space $\mathbf{X}$ are assigned a weight w_j with value almost zero. However, with the standard MAP training approach the benefits of Bayesian shrinkage using the ARD covariance function cannot be realised, as typically overfitting will occur; this is later demonstrated in Figure 5. This is the reason why standard GP-LVM approaches in the literature avoid the ARD covariance function and are sensitive to the selection of q.

On the other hand, the fully Bayesian framework allows for a "soft" model selection mechanism (Tipping, 2000; Bishop, 1999), stemming from the different role played by q. Specifically, in such an approach q can be seen as an "initial conservative guess" for the effective dimensionality of the latent space; subsequent optimisation renders unnecessary dimensions almost irrelevant by driving the corresponding inverse lengthscales close to zero. Notice, however, that no threshold needs to be employed. Indeed, in the predictive equations all q latent dimensions are used, but the lengthscales automatically weight the contribution of each. In fact, typically the selection for q is not crucial, as long as it is large enough to capture the effective dimensionality of the data. That is, if $r > q$ is used instead, then the extra $r - q$ dimensions will only slightly affect any predictions, given that they will be assigned an almost zero weight. This was indeed observed in our initial experiments. An alternative to the ARD shrinkage principle employed in this paper is the spike and slab

principle (Mitchell and Beauchamp, 1988), which provides "hard" shrinkage so that unnecessary dimensions are assigned a weight exactly equal to zero. This alternative constitutes a promising direction for future research in the context of GP-LVMs.

Given the above, it is clear that the development of fully Bayesian approaches for training GP-LVMs could make these models more reliable and provide rigorous solutions to the limitations of MAP training. The variational method presented in the next section is such an approach that, as demonstrated in the experiments, shows great ability in avoiding overfitting and permits automatic soft selection of the latent dimensionality.

3. Variational Gaussian Process Latent Variable Models

In this section we describe in detail our proposed method which is based on a non-standard variational approximation that uses auxiliary variables. The resulting class of algorithms will be referred to as *Variational Gaussian Process Latent Variable Models*, or simply *variational GP-LVMs*.

We start with Section 3.1 where we explain the obstacles we need to overcome when applying variational methods to the GP-LVM and specifically why the standard mean field approach is not immediately tractable. In Section 3.2, we show how the use of auxiliary variables together with a certain variational distribution results in a tractable approximation. In Section 3.3 we give specific details about how to apply our framework to the two different GP-LVM variants that this paper is concerned with: the standard GP-LVM and the dynamical/warped one. Finally, we outline two extensions of our variational method that enable its application in more specific modelling scenarios. In the end of Section 3.3.2 we explain how multiple independent time-series can be accommodated within the same dynamical model and in Section 3.4 we describe a simple trick that makes the model (and, in fact, any GP-LVM model) applicable to vast dimensionalities.

3.1 Standard Mean Field is Challenging for GP-LVM

A Bayesian treatment of the GP-LVM requires the computation of the log marginal likelihood associated with the joint distribution of equation (5). Both sets of unknown random variables have to be marginalised out: the mapping values $\mathbf{F}$ (as in the standard model) and the latent space $\mathbf{X}$. Thus, the required integral is written as

$$\log p(\mathbf{Y}) = \log \int p(\mathbf{Y},\mathbf{F},\mathbf{X})\mathrm{d}\mathbf{X}\mathrm{d}\mathbf{F} = \log \int p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{X})p(\mathbf{X})\mathrm{d}\mathbf{X}\mathrm{d}\mathbf{F} \tag{9}$$

$$= \log \int p(\mathbf{Y}|\mathbf{F})\left(\int p(\mathbf{F}|\mathbf{X})p(\mathbf{X})\mathrm{d}\mathbf{X}\right)\mathrm{d}\mathbf{F}. \tag{10}$$

The key difficulty with this Bayesian approach is propagating the prior density $p(\mathbf{X})$ through the non-linear mapping. Indeed, the nested integral in equation (10) can be written as

$$\int p(\mathbf{X}) \prod_{j=1}^{p} p(\mathbf{f}_{:,j}|\mathbf{X})\mathrm{d}\mathbf{X}$$

where each term $p(\mathbf{f}_{:,j}|\mathbf{X})$, given by (4), is proportional to $|\mathbf{K}_{ff}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\mathbf{f}_{:,j}^\top \mathbf{K}_{ff}^{-1}\mathbf{f}_{:,j}\right)$. Clearly, this term contains $\mathbf{X}$, which are the inputs of the kernel matrix $\mathbf{K}_{ff}$, in a complex, non-linear manner and therefore analytical integration over $\mathbf{X}$ is infeasible.

To make progress we can invoke the standard variational Bayesian methodology (Bishop, 2006) to approximate the marginal likelihood of equation (9) with a variational lower bound. We introduce a factorised variational distribution over the unknown random variables,

$$q(\mathbf{F}, \mathbf{X}) = q(\mathbf{F})q(\mathbf{X}),$$

which aims at approximating the true posterior $p(\mathbf{F}|\mathbf{Y}, \mathbf{X})p(\mathbf{X}|\mathbf{Y})$. Based on Jensen's inequality, we can obtain the standard variational lower bound on the log marginal likelihood

$$\log p(\mathbf{Y}) \geq \int q(\mathbf{F})q(\mathbf{X}) \log \frac{p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{X})p(\mathbf{X})}{q(\mathbf{F})q(\mathbf{X})} \mathrm{d}\mathbf{F}\mathrm{d}\mathbf{X}. \tag{11}$$

Nevertheless, this standard *mean field* approach remains problematic because the lower bound above is still intractable to compute. To isolate the intractable term, observe that (11) can be written as

$$\log p(\mathbf{Y}) \geq \int q(\mathbf{F})q(\mathbf{X}) \log p(\mathbf{F}|\mathbf{X}) \mathrm{d}\mathbf{F}\mathrm{d}\mathbf{X} + \int q(\mathbf{F})q(\mathbf{X}) \log \frac{p(\mathbf{Y}|\mathbf{F})p(\mathbf{X})}{q(\mathbf{F})q(\mathbf{X})} \mathrm{d}\mathbf{F}\mathrm{d}\mathbf{X},$$

where the first term of the above equation contains the expectation of $\log p(\mathbf{F}|\mathbf{X})$ under the distribution $q(\mathbf{X})$. This requires an integration over $\mathbf{X}$ which appears non-linearly in $\mathbf{K}_{ff}^{-1}$ and $\log|\mathbf{K}_{ff}|$ and cannot be done analytically. Therefore, standard mean field variational methodologies do not lead to an analytically tractable variational lower bound.

3.2 Tractable Lower Bound by Introducing Auxiliary Variables

In contrast, our framework allows us to compute a closed-form lower bound on the marginal likelihood by applying variational inference after expanding the GP prior so as to include auxiliary inducing variables. Originally, inducing variables were introduced for computational speed ups in GP regression models (Csató and Opper, 2002; Seeger et al., 2003; Csató, 2002; Snelson and Ghahramani, 2006; Quiñonero Candela and Rasmussen, 2005; Titsias, 2009). In our approach, these extra variables will be used within the variational sparse GP framework of Titsias (2009).

More specifically, we expand the joint probability model in (5) by including m extra samples (inducing points) of the GP latent mapping $\mathbf{f}(\mathbf{x})$, so that $\mathbf{u}_{i,:} \in \mathbb{R}^p$ is such a sample. The inducing points are collected in a matrix $\mathbf{U} \in \mathbb{R}^{m\times p}$ and constitute latent function evaluations at a set of pseudo-inputs $\mathbf{X}_u \in \mathbb{R}^{m\times q}$. The augmented joint probability density takes the form

$$\begin{aligned} p(\mathbf{Y}, \mathbf{F}, \mathbf{U}, \mathbf{X}|\mathbf{X}_u) =& p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{U}, \mathbf{X}, \mathbf{X}_u)p(\mathbf{U}|\mathbf{X}_u)p(\mathbf{X}) \\ =& \left(\prod_{j=1}^{p} p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j}, \mathbf{X}, \mathbf{X}_u)p(\mathbf{u}_{:,j}|\mathbf{X}_u)\right) p(\mathbf{X}), \end{aligned} \tag{12}$$

where

$$p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j}, \mathbf{X}, \mathbf{X}_u) = \mathcal{N}\left(\mathbf{f}_{:,j}|\mathbf{a}_j, \mathbf{\Sigma}_f\right) \tag{13}$$

is the conditional GP prior (see e.g. Rasmussen and Williams (2006)), with

$$\mathbf{a}_j = \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}_{:,j} (\text{with } a_{i,j} = k_f(\mathbf{x}_{i,:}, \mathbf{X}_u)\mathbf{K}_{uu}^{-1}\mathbf{u}_{:,j}) \quad \text{and} \quad \mathbf{\Sigma}_f = \mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{uf}. \tag{14}$$

Further,

$$p(\mathbf{u}_{:,j}|\mathbf{X}_u) = \mathcal{N}(\mathbf{u}_{:,j}|\mathbf{0}, \mathbf{K}_{uu}) \tag{15}$$

is the marginal GP prior over the inducing variables. In the above expressions, $\mathbf{K}_{uu}$ denotes the covariance matrix constructed by evaluating the covariance function on the inducing points, $\mathbf{K}_{uf}$ is the cross-covariance between the inducing and the latent points and $\mathbf{K}_{fu} = \mathbf{K}_{uf}^\top$. Figure 3(b) graphically illustrates the augmented probability model.

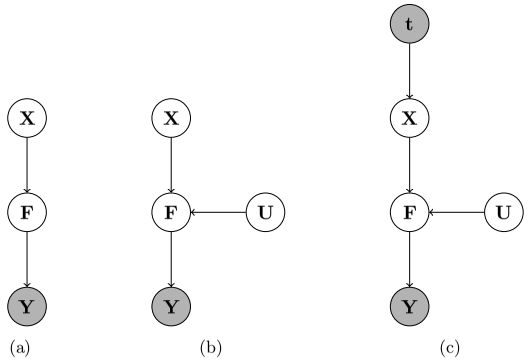


Figure 3: The graphical model for the GP-LVM (a) is augmented with auxiliary variables to obtain the variational GP-LVM model (b) and its dynamical version (c). Shaded nodes represent observed variables. In general, the top level input in (c) can be arbitrary, depending on the application.

Notice that the likelihood $p(\mathbf{Y}|\mathbf{X})$ can be equivalently computed from the above augmented model by marginalizing out $(\mathbf{F}, \mathbf{U})$ and crucially this is true for any value of the inducing inputs $\mathbf{X}_u$. This means that, unlike $\mathbf{X}$, the inducing inputs $\mathbf{X}_u$ are not random variables and neither are they model hyperparameters; they are variational parameters. This interpretation of the inducing inputs is key in developing our approximation and it arises from the variational approach of Titsias (2009). Taking advantage of this observation we now simplify our notation by dropping $\mathbf{X}_u$ from our expressions.

We can now apply variational inference to approximate the true posterior, $p(\mathbf{F}, \mathbf{U}, \mathbf{X}|\mathbf{Y}) = p(\mathbf{F}|\mathbf{U}, \mathbf{Y}, \mathbf{X})\ p(\mathbf{U}|\mathbf{Y}, \mathbf{X})p(\mathbf{X}|\mathbf{Y})$ with a variational distribution of the form

$$q(\mathbf{F}, \mathbf{U}, \mathbf{X}) = p(\mathbf{F}|\mathbf{U}, \mathbf{X})q(\mathbf{U})q(\mathbf{X}) = \left(\prod_{j=1}^{p} p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j}, \mathbf{X})q(\mathbf{u}_{:,j})\right) q(\mathbf{X}), \tag{16}$$

where a key ingredient of this distribution is that the conditional GP prior term $p(\mathbf{F}|\mathbf{U},\mathbf{X})$ that appears in the joint density in (12) is also part of the variational distribution. As shown below this crucially leads to cancellation of difficult terms (involving inverses and determinants over kernel matrices on $\mathbf{X}$) and allows us to compute a closed-form variational lower bound. Furthermore, under this choice the conditional GP prior term $p(\mathbf{F}|\mathbf{U},\mathbf{X})$ attempts to approximate the corresponding exact posterior term $p(\mathbf{F}|\mathbf{U},\mathbf{Y},\mathbf{X})$. This promotes the inducing variables $\mathbf{U}$ to become *sufficient statistics* so that the optimisation of the variational distribution over the inducing inputs $\mathbf{X}_u$ attempts to construct $\mathbf{U}$ so that $\mathbf{F}$ approximately becomes conditionally independent from the data $\mathbf{Y}$ given $\mathbf{U}$. To achieve exact conditional independence we might need to use a large number of inducing variables so that $p(\mathbf{F}|\mathbf{U},\mathbf{X})$ becomes very sharply picked (a delta function). In practice, however, the number of inducing variables must be chosen so that to balance between computational complexity, which is cubic over the number m of inducing variables (see Section 3.4), and approximation accuracy where the latter deteriorates as m becomes smaller.

Moreover, the distribution $q(\mathbf{X})$ in (16) is constrained to be Gaussian,

$$q(\mathbf{X}) = \mathcal{N}\left(\mathbf{X}|\mathcal{M},\mathcal{S}\right), \tag{17}$$

while $q(\mathbf{U})$ is an arbitrary (i.e. unrestricted) variational distribution. We can choose the Gaussian $q(\mathbf{X})$ to factorise across latent dimensions or datapoints and, as will be discussed in Section 3.3, this choice will depend on the form of the prior distribution $p(\mathbf{X})$. For the time being, however, we shall proceed assuming a general form for this Gaussian.

The particular choice for the variational distribution allows us to analytically compute a lower bound. The key reason behind this is that the conditional GP prior term is part of the variational distribution which promotes the cancellation of the intractable $\log p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})$ term. Indeed, by making use of equations (12) and (16) the derivation of the lower bound is as follows:

$$\begin{aligned}
\mathcal{F}\left(q(\mathbf{X}),q(\mathbf{U})\right) &= \int q(\mathbf{F},\mathbf{U},\mathbf{X})\log\frac{p(\mathbf{Y},\mathbf{F},\mathbf{U},\mathbf{X})}{q(\mathbf{F},\mathbf{U},\mathbf{X})}\mathrm{d}\mathbf{X}\mathrm{d}\mathbf{F}\mathrm{d}\mathbf{U}\\
&= \int\prod_{j=1}^{p} p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})q(\mathbf{u}_{:,j})q(\mathbf{X})\log\frac{\prod_{j=1}^{p} p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\cancel{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})}p(\mathbf{u}_{:,j})p(\mathbf{X})}{\prod_{j=1}^{p}\cancel{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})}q(\mathbf{u}_{:,j})q(\mathbf{X})}\mathrm{d}\mathbf{X}\mathrm{d}\mathbf{F}\mathrm{d}\mathbf{U}\\
&= \int\prod_{j=1}^{p} p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})q(\mathbf{u}_{:,j})q(\mathbf{X})\log\frac{\prod_{j=1}^{p} p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})p(\mathbf{u}_{:,j})}{\prod_{j=1}^{p} q(\mathbf{u}_{:,j})}\mathrm{d}\mathbf{X}\mathrm{d}\mathbf{F}\mathrm{d}\mathbf{U}\\
&- \int q(\mathbf{X})\log\frac{q(\mathbf{X})}{p(\mathbf{X})}\mathrm{d}\mathbf{X}\\
&= \hat{\mathcal{F}}\left(q(\mathbf{X}),q(\mathbf{U})\right) - \mathrm{KL}\left(q(\mathbf{X})\,\|\,p(\mathbf{X})\right),
\end{aligned} \tag{18}$$

with

$$\begin{aligned}
\hat{\mathcal{F}}\left(q(\mathbf{X}),q(\mathbf{U})\right) &= \sum_{j=1}^{p}\int q(\mathbf{u}_{:,j})\left(\left\langle\log p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\right\rangle_{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})q(\mathbf{X})} + \log\frac{p(\mathbf{u}_{:,j})}{q(\mathbf{u}_{:,j})}\right)\mathrm{d}\mathbf{u}_{:,j}\\
&= \sum_{j=1}^{p}\hat{\mathcal{F}}_j\left(q(\mathbf{X}),q(\mathbf{u}_{:,j})\right),
\end{aligned} \tag{19}$$

where $\langle\cdot\rangle$ is a shorthand for expectation. Clearly, the second KL term can be easily calculated since both $p(\mathbf{X})$ and $q(\mathbf{X})$ are Gaussians; explicit expressions are given in Section 3.3. To compute $\hat{\mathcal{F}}_j\left(q(\mathbf{X}),q(\mathbf{u}_{:,j})\right)$, first note that (see Appendix A for details)

$$\left\langle\log p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\right\rangle_{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})} = \log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right) - \frac{1}{2\sigma^2}\mathrm{tr}\left(\mathbf{K}_{ff}\right) + \frac{1}{2\sigma^2}\mathrm{tr}\left(\mathbf{K}_{uu}^{-1}\mathbf{K}_{uf}\mathbf{K}_{fu}\right), \tag{20}$$

where $\mathbf{a}_j$ is given by equation (14), based on which we can write

$$\hat{\mathcal{F}}_j\left(q(\mathbf{X}),q(\mathbf{u}_{:,j})\right) = \int q(\mathbf{u}_{:,j})\log\frac{e^{\left\langle\log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})}}p(\mathbf{u}_{:,j})}{q(\mathbf{u}_{:,j})}\mathrm{d}\mathbf{u}_{:,j} - \mathcal{A}, \tag{21}$$

where $\mathcal{A} = \frac{1}{2\sigma^2}\mathrm{tr}\left(\left\langle\mathbf{K}_{ff}\right\rangle_{q(\mathbf{X})}\right) - \frac{1}{2\sigma^2}\mathrm{tr}\left(\mathbf{K}_{uu}^{-1}\left\langle\mathbf{K}_{uf}\mathbf{K}_{fu}\right\rangle_{q(\mathbf{X})}\right)$. The expression in (21) is a KL-like quantity and, therefore, $q(\mathbf{u}_{:,j})$ is optimally set to be proportional to the numerator inside the logarithm of the above equation, i.e.

$$q(\mathbf{u}_{:,j}) = \frac{e^{\left\langle\log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})}}p(\mathbf{u}_{:,j})}{\int e^{\left\langle\log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})}}p(\mathbf{u}_{:,j})\mathrm{d}\mathbf{u}_{:,j}}, \tag{22}$$

which is just a Gaussian distribution (see Appendix A for an explicit form). We can now re-insert the optimal value for $q(\mathbf{u}_{:,j})$ back into $\hat{\mathcal{F}}_j\left(q(\mathbf{X}),q(\mathbf{u}_{:,j})\right)$ in (21) to obtain:

$$\hat{\mathcal{F}}_j\left(q(\mathbf{X})\right) = \log\int e^{\left\langle\log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})}}p(\mathbf{u}_{:,j})\mathrm{d}\mathbf{u}_{:,j} - \mathcal{A}, \tag{23}$$

$$= \log\int\prod_{i=1}^{n} e^{\left\langle\log\mathcal{N}\left(y_{i,j}|a_{i,j},\sigma^2\right) - \frac{1}{2\sigma^2}\left(k_f(\mathbf{x}_{i,:},\mathbf{x}_{i,:}) - k_f(\mathbf{x}_{i,:},\mathbf{X}_u)\mathbf{K}_{uu}^{-1}k_f(\mathbf{X}_u,\mathbf{x}_{i,:})\right)\right\rangle_{q(\mathbf{x}_{i,:})}}p(\mathbf{u}_{:,j})\mathrm{d}\mathbf{u}_{:,j}, \tag{24}$$

where the second expression uses the factorisation of the Gaussian likelihood across data points and it implies that independently of how complex the overall variational distribution $q(\mathbf{X})$ could be, $\hat{\mathcal{F}}_j$ will depend only on the marginals $q(\mathbf{x}_{i,:})$ over the latent variables associated with different observations. Notice that the above trick of finding the optimal factor $q(\mathbf{u}_{:,j})$ and placing it back into the bound (firstly proposed in (King and Lawrence, 2006)) can be informally explained as *reversing Jensen's inequality* (i.e. moving the log outside of the integral) in the initial bound from (21) as pointed out by Titsias (2009).

Furthermore, by optimally eliminating $q(\mathbf{u}_{:,j})$ we obtain a tighter bound which no longer depends on this distribution, i.e. $\hat{\mathcal{F}}_j\left(q(\mathbf{X})\right) \geq \hat{\mathcal{F}}_j\left(q(\mathbf{X}),q(\mathbf{u}_{:,j})\right)$. Also notice that the expectation appearing in equation (23) is a standard Gaussian integral and (23) can be calculated in closed form, which turns out to be (see Appendix A.3 for details)

$$\hat{\mathcal{F}}_j\left(q(\mathbf{X})\right) = \log\left[\frac{\sigma^{-n}|\mathbf{K}_{uu}|^{\frac{1}{2}}}{(2\pi)^{\frac{n}{2}}|\sigma^{-2}\boldsymbol{\Psi}_2 + \mathbf{K}_{uu}|^{\frac{1}{2}}}e^{-\frac{1}{2}\mathbf{y}_{:,j}^{\top}\mathbf{W}\mathbf{y}_{:,j}}\right] - \frac{\psi_0}{2\sigma^2} + \frac{1}{2\sigma^2}\mathrm{tr}\left(\mathbf{K}_{uu}^{-1}\boldsymbol{\Psi}_2\right) \tag{25}$$

where the quantities

$$\psi_0 = \mathrm{tr}\left(\left\langle\mathbf{K}_{\tilde{f}\tilde{f}}\right\rangle_{q(\mathbf{X})}\right)\ ,\quad \boldsymbol{\Psi}_1 = \left\langle\mathbf{K}_{fu}\right\rangle_{q(\mathbf{X})}\ ,\quad \boldsymbol{\Psi}_2 = \left\langle\mathbf{K}_{uf}\mathbf{K}_{fu}\right\rangle_{q(\mathbf{X})} \tag{26}$$

are referred to as Ψ statistics and $\mathbf{W} = \sigma^{-2}\mathbf{I}_n - \sigma^{-4}\mathbf{\Psi}_1(\sigma^{-2}\mathbf{\Psi}_2 + \mathbf{K}_{uu})^{-1}\mathbf{\Psi}_1^\top$.

The computation of $\hat{\mathcal{F}}_j\left(q(\mathbf{X})\right)$ only requires us to compute matrix inverses and determinants which involve $\mathbf{K}_{uu}$ instead of $\mathbf{K}_{ff}$, something which is tractable since $\mathbf{K}_{uu}$ does not depend on $\mathbf{X}$. Therefore, this expression is straightforward to compute, as long as the covariance function k_f is selected so that the Ψ quantities of equation (26) can be computed analytically.

It is worth noticing that the Ψ statistics are computed in a decomposable way across the latent variables of different observations which is due to the factorisation in (24). In particular, the statistics ψ_0 and $\mathbf{\Psi}_2$ are written as sums of independent terms where each term is associated with a data point and similarly each column of the matrix $\mathbf{\Psi}_1$ is associated with only one data point. This decomposition is useful when a new data vector is inserted into the model and can also help to speed up computations during test time as discussed in Section 4. It can also allow for parallelisation in the computations as suggested firstly by Gal et al. (2014) and then by Dai et al. (2014). Therefore, the averages of the covariance matrices over $q(\mathbf{X})$ in equation (26) of the Ψ statistics can be computed separately for each marginal $q(\mathbf{x}_{i,:}) = \mathcal{N}\left(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:}, \mathbf{S}_i\right)$ taken from the full $q(\mathbf{X})$ of equation (17). We can, thus, write that $\psi_0 = \sum_{i=1}^n \psi_0^i$ where

$$\psi_0^i = \int k_f(\mathbf{x}_{i,:}, \mathbf{x}_{i,:})\mathcal{N}\left(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:}, \mathbf{S}_i\right) \mathrm{d}\mathbf{x}_{i,:}. \tag{27}$$

Further, $\mathbf{\Psi}_1$ is an $n \times m$ matrix such that

$$(\Psi_1)_{i,k} = \int k_f(\mathbf{x}_{i,:}, (\mathbf{x}_u)_{k,:})\mathcal{N}\left(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:}, \mathbf{S}_i\right) \mathrm{d}\mathbf{x}_{i,:}, \tag{28}$$

where $(\mathbf{x}_u)_{k,:}$ denotes the kth row of $\mathbf{X}_u$. Finally, $\mathbf{\Psi}_2$ is an $m \times m$ matrix which is written as $\mathbf{\Psi}_2 = \sum_{i=1}^n \Psi_2^i$ where Ψ_2^i is such that

$$(\Psi_2^i)_{k,k'} = \int k_f(\mathbf{x}_{i,:}, (\mathbf{x}_u)_{k,:})k_f((\mathbf{x}_u)_{k',:}, \mathbf{x}_{i,:})\mathcal{N}\left(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:}, \mathbf{S}_i\right) \mathrm{d}\mathbf{x}_{i,:}. \tag{29}$$

Notice that these statistics constitute convolutions of the covariance function k_f with Gaussian densities and are tractable for many standard covariance functions, such as the ARD exponentiated quadratic or the linear one. The analytic forms of the Ψ statistics for the aforementioned covariance functions are given in Appendix B.

To summarize, the final form of the variational lower bound on the marginal likelihood $p(\mathbf{Y})$ is written as

$$\mathcal{F}\left(q(\mathbf{X})\right) = \hat{\mathcal{F}}\left(q(\mathbf{X})\right) - \mathrm{KL}\left(q(\mathbf{X}) \,\|\, p(\mathbf{X})\right), \tag{30}$$

where $\hat{\mathcal{F}}\left(q(\mathbf{X})\right)$ can be obtained by summing both sides of (25) over the p outputs,

$$\hat{\mathcal{F}}\left(q(\mathbf{X})\right) = \sum_{j=1}^{p} \hat{\mathcal{F}}_j\left(q(\mathbf{X})\right).$$

We note that the above framework is, in essence, computing the following approximation analytically,

$$\hat{\mathcal{F}}\left(q(\mathbf{X})\right) \leq \int q(\mathbf{X}) \log p(\mathbf{Y}|\mathbf{X}) \mathrm{d}\mathbf{X}. \tag{31}$$

The lower bound (18) can be jointly maximised over the model parameters $\boldsymbol{\theta}$ and variational parameters $\{\mathcal{M}, \mathcal{S}, \mathbf{X}_u\}$ by applying a gradient-based optimisation algorithm. This approach is similar to the optimisation of the MAP objective function employed in the standard GP-LVM (Lawrence, 2005) with the main difference being that instead of optimising the random variables $\mathbf{X}$, we now optimise a set of *variational parameters* which govern the approximate posterior mean and variance for $\mathbf{X}$. Furthermore, the inducing inputs $\mathbf{X}_u$ are variational parameters and the optimisation over them simply improves the approximation similarly to variational sparse GP regression (Titsias, 2009).

By investigating more carefully the resulting expression of the bound allows us to observe that each term $\hat{\mathcal{F}}_j\left(q(\mathbf{X})\right)$ from (25), that depends on the single column of data $\mathbf{y}_{:,j}$, closely resembles the corresponding variational lower bound obtained by applying the method of Titsias (2009) in standard sparse GP regression. The difference in variational GP-LVM is that now $\mathbf{X}$ is marginalized out so that the terms containing $\mathbf{X}$, i.e. the kernel quantities $\mathrm{tr}\left(\mathbf{K}_{ff}\right)$, $\mathbf{K}_{fu}$ and $\mathbf{K}_{fu}\mathbf{K}_{uf}$, are transformed into averages (i.e. the Ψ quantities in (26)) with respect to the variational distribution $q(\mathbf{X})$.

Similarly to the standard GP-LVM, the non-convexity of the optimisation surface means that local optima can pose a problem and, therefore, sensible initialisations have to be made. In contrast to the standard GP-LVM, in the *variational* GP-LVM the choice of a covariance function is limited to the class of kernels that render the Ψ statistics tractable. Throughout this paper we employ the ARD exponentiated quadratic covariance function. Improving on these areas (non-convex optimisation and choice of covariance function) is, thus, an interesting direction for future research.

Finally, notice that the application of the variational method developed in this paper is not restricted to the set of latent points. As in (Titsias and Lázaro-Gredilla, 2013), a fully Bayesian approach can be obtained by additionally placing priors on the kernel parameters and, subsequently, integrating them out variationally with the methodology that we described in this section.

3.3 Applying the Variational Framework to Different GP-LVM Variants

Different variational GP-LVM algorithms can be obtained by varying the form of the latent space prior $p(\mathbf{X})$ which so far has been left unspecified. One useful property of the variational lower bound is that $p(\mathbf{X})$ appears only in the separate KL divergence term, as can be seen by equation (18), which can be computed analytically when $p(\mathbf{X})$ is Gaussian. This allows our framework to easily accommodate different Gaussian forms for the latent space prior which give rise to different GP-LVM variants. In particular, incorporating a specific prior mainly requires us to specify a suitable factorisation for $q(\mathbf{X})$ and compute the corresponding KL term. In contrast, the general structure of the more complicated $\hat{\mathcal{F}}\left(q(\mathbf{X})\right)$ term remains unaffected. Next we demonstrate these ideas by giving further details about how to apply the variational method to the two GP-LVM variants discussed in Section 2.2. For both cases we follow the recipe that the factorisation of the variational distribution $q(\mathbf{X})$ resembles the factorisation of the prior $p(\mathbf{X})$.

3.3.1 The Standard Variational GP-LVM for I.i.d. Data

In the simplest case, the latent space prior is just a standard normal density, fully factorised across datapoints and latent dimensions, as shown in (7). This is the typical assumption in latent variable models, such as factor analysis and PPCA (Bartholomew, 1987; Basilevsky, 1994; Tipping and Bishop, 1999). We choose a variational distribution $q(\mathbf{X})$ that follows the factorisation of the prior,

$$q(\mathbf{X}) = \prod_{i=1}^{n} \mathcal{N}\left(\mathbf{x}_{i,:} | \boldsymbol{\mu}_{i,:}, \mathbf{S}_i\right), \tag{32}$$

where each covariance matrix $\mathbf{S}_i$ is diagonal. Notice that this variational distribution depends on $2nq$ free parameters. The corresponding KL quantity appearing in (30) takes the explicit form

$$\mathrm{KL}\left(q(\mathbf{X}) \,\|\, p(\mathbf{X})\right) = \frac{1}{2}\sum_{i=1}^{n} \mathrm{tr}\left(\boldsymbol{\mu}_{i,:}\boldsymbol{\mu}_{i,:}^{\top} + \mathbf{S}_i - \log \mathbf{S}_i\right) - \frac{nq}{2},$$

where $\log \mathbf{S}_i$ denotes the diagonal matrix resulting from $\mathbf{S}_i$ by taking the logarithm of its diagonal elements. To train the model we simply need to substitute the above term in the final form of the variational lower bound in equation (30) and follow the gradient-based optimisation procedure.

The resulting variational GP-LVM can be seen as a non-linear version of Bayesian probabilistic PCA (Bishop, 1999; Minka, 2001). In the experiments, we consider this model for non-linear dimensionality reduction and demonstrate its ability to automatically estimate the effective latent dimensionality.

3.3.2 The Dynamical Variational GP-LVM for Sequence Data

We now turn into the second model discussed in Section 2.2, which is suitable for sequence data. Again we define a variational distribution $q(\mathbf{X})$ so that it resembles fully the factorisation of the prior, i.e.

$$q(\mathbf{X}) = \prod_{j=1}^{q} \mathcal{N}\left(\mathbf{x}_{:,j} | \boldsymbol{\mu}_{:,j}, \mathbf{S}_j\right),$$

where $\mathbf{S}_j$ is a $n \times n$ full covariance matrix. The corresponding KL term takes the form

$$\mathrm{KL}\left(q(\mathbf{X}) \,\|\, p(\mathbf{X}|\mathbf{t})\right) = \frac{1}{2}\sum_{j=1}^{q}\left[\mathrm{tr}\left(\mathbf{K}_x^{-1}\mathbf{S}_j + \mathbf{K}_x^{-1}\boldsymbol{\mu}_{:,j}\boldsymbol{\mu}_{:,j}^{\top}\right) + \log|\mathbf{K}_x| - \log|\mathbf{S}_j|\right] - \frac{nq}{2}.$$

This term can be substituted into the final form of the variational lower bound in (30) and allow training using a gradient-based optimisation procedure. If implemented naively, such a procedure, will require too many parameters to tune since the variational distribution depends on $nq + \frac{n(n+1)}{2}q$ free parameters. However, by applying the reparametrisation trick suggested by Opper and Archambeau (2009) we can reduce the number of parameters in the variational distribution to just $2nq$. Specifically, the stationary conditions obtained by setting to zero the first derivatives of the variational bound w.r.t. $\mathbf{S}_j$ and $\boldsymbol{\mu}_{:,j}$ take the form

$$\mathbf{S}_j = \left(\mathbf{K}_x^{-1} + \boldsymbol{\Lambda}_j\right)^{-1} \quad \text{and} \quad \boldsymbol{\mu}_{:,j} = \mathbf{K}_x\bar{\boldsymbol{\mu}}_{:,j}, \tag{33}$$

where

$$\boldsymbol{\Lambda}_j = -2\frac{\partial \hat{\mathcal{F}}\left(q(\mathbf{X})\right)}{\partial \mathbf{S}_j} \quad \text{and} \quad \bar{\boldsymbol{\mu}}_{:,j} = \frac{\partial \hat{\mathcal{F}}\left(q(\mathbf{X})\right)}{\partial \boldsymbol{\mu}_{:,j}}. \tag{34}$$

Here, $\boldsymbol{\Lambda}_j$ is a $n \times n$ diagonal positive definite matrix and $\bar{\boldsymbol{\mu}}_{:,j}$ is a $n-$dimensional vector. Notice that the fact that the gradients of $\hat{\mathcal{F}}\left(q(\mathbf{X})\right)$ with respect to a full (coupled across data points) matrix $\mathbf{S}_j$ reduce to a diagonal matrix is because only the diagonal elements of $\mathbf{S}_j$ appear in $\hat{\mathcal{F}}\left(q(\mathbf{X})\right)$. This fact is a consequence of the factorisation of the likelihood across data points, which makes the term $\hat{\mathcal{F}}\left(q(\mathbf{X})\right)$ to depend only on marginals of the full variational distribution, as it was pointed by the general expression in equation (24).

The above stationary conditions tell us that, since $\mathbf{S}_j$ depends on a diagonal matrix $\boldsymbol{\Lambda}_j$, we can re-parametrise it using only the diagonal elements of that matrix, denoted by the $n-$dimensional vector $\boldsymbol{\lambda}_j$ where all elements of $\boldsymbol{\lambda}_j$ are restricted to be non-negative. Notice that with this re-parameterisation, and if we consider the pair $(\boldsymbol{\lambda}_j,\ \bar{\boldsymbol{\mu}}_{:,j})$ as the set of free parameters, the bound property is retained because any such pair defines a valid Gaussian distribution $q(\mathbf{X})$ based on which the corresponding (always valid) lower bound is computed. Therefore, if we optimise the $2qn$ parameters $(\boldsymbol{\lambda}_j,\ \bar{\boldsymbol{\mu}}_{:,j})$ and find some final values for those parameters, then we can obtain the mean and covariance of $q(\mathbf{X})$ using the transformation in equation (33).

There are two optimisation strategies, depending on the way we choose to treat the newly introduced parameters $\boldsymbol{\lambda}_j$ and $\bar{\boldsymbol{\mu}}_{:,j}$. Firstly, inspired by Opper and Archambeau (2009) we can construct an iterative optimisation scheme. More precisely, the variational bound $\mathcal{F}$ in equation (30) depends on the *actual* variational parameters $\boldsymbol{\mu}_{:,j}$ and $\mathbf{S}_j$ of $q(\mathbf{X})$, which through equation (33) depend on the newly introduced quantities $\bar{\boldsymbol{\mu}}_{:,j}$ and $\boldsymbol{\lambda}_j$ which, in turn, are associated with $\mathcal{F}$ through equation (34). These observations can lead to an EM-style algorithm which alternates between estimating one of the parameter sets $\{\boldsymbol{\theta}, \mathbf{X}_u\}$ and $\{\mathcal{M}, \mathcal{S}\}$ by keeping the other set fixed. An alternative approach, which is the one we use in our implementation, is to treat the new parameters $\boldsymbol{\lambda}_j$ and $\bar{\boldsymbol{\mu}}_{:,j}$ as completely free ones so that equation (34) is never used. In this case, the variational parameters are optimised directly with a gradient based optimiser, jointly with the model hyperparameters and the inducing inputs.

Overall, the above reparameterisation is appealing not only because of improved complexity, but also because of optimisation robustness. Indeed, equation (33) confirms that the original variational parameters are coupled via $\mathbf{K}_x$, which is a full-rank covariance matrix. By reparametrising according to equation (33) and treating the new parameters as free ones, we manage to approximately break this coupling and apply our optimisation algorithm on a set of less correlated parameters.

Furthermore, the methodology described above can be readily applied to model dependencies of a different nature (e.g. spatial rather than temporal), as any kind of high dimensional input variable can replace the temporal inputs of the graphical model in figure 3(c). Therefore, by simply replacing the input $\mathbf{t}$ with any other kind of observed input $\mathbf{Z}$ we trivially obtain a Bayesian framework for warped GP regression (Snelson et al., 2004; Lázaro-Gredilla, 2012) for which we can predict the latent function values in new inputs $\mathbf{Z}_*$ through a non-linear, latent warping layer, using exactly the same architecture and equations described in this section and in Section 4.2. Similarly, if the observed inputs of the top

layer are taken to be the outputs themselves, then we obtain a probabilistic auto-encoder (e.g. Kingma and Welling (2013)) which is non-parametric and based on Gaussian processes.

Finally, the above dynamical variational GP-LVM algorithm can be easily extended to deal with datasets consisting of multiple independent sequences (probably of different length) such as those arising in human motion capture applications. Let, for example, the dataset be a group of s independent sequences $\left(\mathbf{Y}^{(1)}, ..., \mathbf{Y}^{(s)}\right)$. We would like the dynamical version of our model to capture the underlying commonality of these data. We handle this by allowing a different *temporal* latent function for each of the independent sequences, so that $\mathbf{X}^{(i)}$ is the set of latent variables corresponding to the sequence i. These sets are a priori assumed to be independent since they correspond to separate sequences, i.e. $p\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, ..., \mathbf{X}^{(s)}\right) = \prod_{i=1}^{s} p(\mathbf{X}^{(i)})$. This factorisation leads to a block-diagonal structure for the time covariance matrix $\mathbf{K}_x$, where each block corresponds to one sequence. In this setting, each block of observations $\mathbf{Y}^{(i)}$ is generated from its corresponding $\mathbf{X}^{(i)}$ according to $\mathbf{Y}^{(i)} = \mathbf{F}^{(i)} + \boldsymbol{\epsilon}$, where the latent function which governs this mapping is shared across all sequences and $\boldsymbol{\epsilon}$ is Gaussian noise.

3.4 Time Complexity and Handling Very High Dimensional Datasets

Our variational framework makes use of inducing point representations which provide low-rank approximations to the covariance $\mathbf{K}_{ff}$. For the standard variational GP-LVM, this allows us to avoid the typical cubic complexity of Gaussian processes. Specifically, the computational cost is $O(m^3 + nm^2)$ which reduces to $O(nm^2)$, since we typically select a small set of inducing points $m \ll n$, which allows the variational GP-LVM to handle relatively large training sets (thousands of points, n). The *dynamical* variational GP-LVM, however, still requires the inversion of the covariance matrix $\mathbf{K}_x$ of size $n \times n$, as can be seen in equation (33), thereby inducing a computational cost of $O(n^3)$. Further, the models scale only linearly with the number of dimensions p, since the variational lower bound is a sum of p terms (see equation (19)). Specifically, the number of dimensions only matters when performing calculations involving the data matrix $\mathbf{Y}$. In the final form of the lower bound (and consequently in all of the derived quantities, such as gradients) this matrix only appears in the form $\mathbf{Y}\mathbf{Y}^\top$ which can be precomputed. This means that, when $n \ll p$, we can calculate $\mathbf{Y}\mathbf{Y}^\top$ only once and then substitute $\mathbf{Y}$ with the SVD (or Cholesky decomposition) of $\mathbf{Y}\mathbf{Y}^\top$. In this way, we can work with an $n \times n$ instead of an $n \times p$ matrix. Practically speaking, this allows us to work with data sets involving millions of features. In our experiments we model directly the pixels of HD quality video, exploiting this trick.

4. Predictions with the Variational GP-LVM

In this section, we explain how the proposed Bayesian models can accomplish various kinds of prediction tasks. We will use a star ($*$) to denote test quantities, e.g. a test data matrix will be denoted by $\mathbf{Y}_* \in \Re^{n_* \times p}$ while test row and column vectors of such a matrix will be denoted by $\mathbf{y}_{i,*}$ and $\mathbf{y}_{*,j}$ respectively.

The first type of inference we are interested in is the calculation of the probability density $p(\mathbf{Y}_*|\mathbf{Y})$. The computation of this quantity can allow us to use the model as a density estimator which, for instance, can represent the class conditional distribution in a generative based classification system. We will exploit such a use in Section 5.5. Secondly,

we discuss how from a test data matrix $\mathbf{Y}_* = (\mathbf{Y}_*^u, \mathbf{Y}_*^o)$, we can probabilistically reconstruct the unobserved part $\mathbf{Y}_*^u$ based on the observed part $\mathbf{Y}_*^o$ and where u and o denote non-overlapping sets of indices such that their union is $\{1, \ldots, p\}$. For this second problem the missing dimensions are reconstructed by approximating the mean and the covariance of the Bayesian predictive density $p(\mathbf{Y}_*^u|\mathbf{Y}_*^o, \mathbf{Y})$.

Section 4.1 discusses how to solve the above tasks in the standard variational GP-LVM case while Section 4.2 discusses the dynamical case. Furthermore, for the dynamical case the test points $\mathbf{Y}_*$ are accompanied by their corresponding timestamps $\mathbf{t}_*$ based on which we can perform an additional *forecasting* prediction task, where we are given only a test time vector $\mathbf{t}_*$ and we wish to predict the corresponding outputs.

4.1 Predictions with the Standard Variational GP-LVM

We first discuss how to approximate the density $p(\mathbf{Y}_*|\mathbf{Y})$. By introducing the latent variables $\mathbf{X}$ (corresponding to the training outputs $\mathbf{Y}$) and the new test latent variables $\mathbf{X}_* \in \Re^{n_* \times q}$, we can write the density of interest as the ratio of two marginal likelihoods,

$$p(\mathbf{Y}_*|\mathbf{Y}) = \frac{p(\mathbf{Y}_*, \mathbf{Y})}{p(\mathbf{Y})} = \frac{\int p(\mathbf{Y}_*, \mathbf{Y}|\mathbf{X}, \mathbf{X}_*) p(\mathbf{X}, \mathbf{X}_*) \mathrm{d}\mathbf{X} \mathrm{d}\mathbf{X}_*}{\int p(\mathbf{Y}|\mathbf{X}) p(\mathbf{X}) \mathrm{d}\mathbf{X}}. \tag{35}$$

In the denominator we have the marginal likelihood of the GP-LVM for which we have already computed a variational lower bound. The numerator is another marginal likelihood that is obtained by augmenting the training data $\mathbf{Y}$ with the test points $\mathbf{Y}_*$ and integrating out both $\mathbf{X}$ and the newly inserted latent variable $\mathbf{X}_*$. In the following, we explain in more detail how to approximate the density $p(\mathbf{Y}_*|\mathbf{Y})$ of equation (35) through constructing a ratio of lower bounds.

The quantity $\int p(\mathbf{Y}|\mathbf{X}) p(\mathbf{X}) \mathrm{d}\mathbf{X}$ appearing in the denominator of equation (35) is approximated by the lower bound $e^{\mathcal{F}(q(\mathbf{X}))}$ where $\mathcal{F}(q(\mathbf{X}))$ is the variational lower bound as computed in Section 3.2 and is given in equation (30). The maximisation of this lower bound specifies the variational distribution $q(\mathbf{X})$ over the latent variables in the training data. Then, this distribution remains fixed during test time. The quantity $\int p(\mathbf{Y}_*, \mathbf{Y}|\mathbf{X}, \mathbf{X}_*) p(\mathbf{X}, \mathbf{X}_*) \mathrm{d}\mathbf{X} \mathrm{d}\mathbf{X}_*$ appearing in the numerator of equation (35) is approximated by the lower bound $e^{\mathcal{F}(q(\mathbf{X}, \mathbf{X}_*))}$ which has exactly analogous form to (30). This optimisation is fast, because the factorisation imposed for the variational distribution in equation (32) means that $q(\mathbf{X}, \mathbf{X}_*)$ is also a fully factorised distribution so that we can write $q(\mathbf{X}, \mathbf{X}_*) = q(\mathbf{X}) q(\mathbf{X}_*)$. Then, if $q(\mathbf{X})$ is held fixed[2] during test time, we only need to optimise with respect to the $2n_*q$ parameters of the variational Gaussian distribution $q(\mathbf{X}_*) = \prod_{i=1}^{n_*} q(\mathbf{x}_{i,*}) = \prod_{i=1}^{n_*} \mathcal{N}(\boldsymbol{\mu}_{i,*}, \mathbf{S}_{i,*})$ (where $\mathbf{S}_{i,*}$ is a diagonal matrix). Further, since the Ψ statistics decompose across data, during test time we can re-use the already estimated Ψ statistics corresponding to the averages over $q(\mathbf{X})$ and only need to compute the extra average terms associated with $q(\mathbf{X}_*)$. Note that optimisation of the parameters $(\boldsymbol{\mu}_{i,*}, \mathbf{S}_{i,*})$ of $q(\mathbf{x}_{i,*})$ are subject to local minima. However, sensible initialisations of $\boldsymbol{\mu}_*$ can be employed based on the mean of the variational distributions associated with the nearest neighbours of each test point $\mathbf{y}_{i,*}$ in the training data $\mathbf{Y}$. Given the above, the approximation of $p(\mathbf{Y}_*|\mathbf{Y})$

2. Ideally $q(\mathbf{X})$ would be optimised during test time as well.

is given by rewriting equation (35) as

$$p(\mathbf{Y}_*|\mathbf{Y}) \approx e^{\mathcal{F}(q(\mathbf{X},\mathbf{X}_*))-\mathcal{F}(q(\mathbf{X}))}. \tag{36}$$

Notice that the above quantity does not constitute a bound, but only an approximation to the predictive density.

We now discuss the second prediction problem where a set of partially observed test points $\mathbf{Y}_* = (\mathbf{Y}_*^u, \mathbf{Y}_*^o)$ are given and we wish to reconstruct the missing part $\mathbf{Y}_*^u$. The predictive density is, thus, $p(\mathbf{Y}_*^u|\mathbf{Y}_*^o, \mathbf{Y})$. Notice that $\mathbf{Y}_*^u$ is totally unobserved and, therefore, we cannot apply the methodology described previously. Instead, our objective now is to just approximate the moments of the predictive density. To achieve this, we will first need to introduce the underlying latent function values $\mathbf{F}_*^u$ (the noise-free version of $\mathbf{Y}_*^u$) and the latent variables $\mathbf{X}_*$ so that we can decompose the exact predictive density as follows:

$$p(\mathbf{Y}_*^u|\mathbf{Y}_*^o, \mathbf{Y}) = \int p(\mathbf{Y}_*^u|\mathbf{F}_*^u)p(\mathbf{F}_*^u|\mathbf{X}_*, \mathbf{Y}_*^o, \mathbf{Y})p(\mathbf{X}_*|\mathbf{Y}_*^o, \mathbf{Y})\mathrm{d}\mathbf{F}_*^u\mathrm{d}\mathbf{X}_*.$$

Then, we can introduce the approximation coming from the variational distribution so that

$$p(\mathbf{Y}_*^u|\mathbf{Y}_*^o, \mathbf{Y}) \approx q(\mathbf{Y}_*^u|\mathbf{Y}_*^o, \mathbf{Y}) = \int p(\mathbf{Y}_*^u|\mathbf{F}_*^u)q(\mathbf{F}_*^u|\mathbf{X}_*)q(\mathbf{X}_*)\mathrm{d}\mathbf{F}_*^u\mathrm{d}\mathbf{X}_*, \tag{37}$$

based on which we wish to predict $\mathbf{Y}_*^u$ by estimating its mean $\mathbb{E}(\mathbf{Y}_*^u)$ and covariance $\mathrm{Cov}(\mathbf{Y}_*^u)$. This problem takes the form of GP prediction with uncertain inputs similar to (Oakley and O'Hagan, 2002; Quiñonero-Candela et al., 2003; Girard et al., 2003), where the distribution $q(\mathbf{X}_*)$ expresses the uncertainty over these inputs. The first term of the above integral comes from the Gaussian likelihood so $\mathbf{Y}_*^u$ is just a noisy version of $\mathbf{F}_*^u$, as shown in equation (6). The remaining two terms together $q(\mathbf{F}_*^u|\mathbf{X}_*)q(\mathbf{X}_*)$ are obtained by applying the variational methodology in order to optimise a variational lower bound on the following log marginal likelihood

$$\log p(\mathbf{Y}_*^o, \mathbf{Y}) = \log \int p(\mathbf{Y}_*^o, \mathbf{Y}|\mathbf{X}_*, \mathbf{X})p(\mathbf{X}_*, \mathbf{X})\mathrm{d}\mathbf{X}_*\mathrm{d}\mathbf{X} \tag{38}$$

which is associated with the total set of observations $(\mathbf{Y}_*^o, \mathbf{Y})$. By following exactly Section 3, we can construct and optimise a lower bound $\mathcal{F}(q(\mathbf{X}, \mathbf{X}_*))$ on the above quantity, which along the way it allows us to compute a Gaussian variational distribution $q(\mathbf{F}, \mathbf{F}_*^u, \mathbf{X}, \mathbf{X}_*)$ from which $q(\mathbf{F}_*^u|\mathbf{X}_*)q(\mathbf{X}_*)$ is just a marginal. Further details about the form of the variational lower bound and how $q(\mathbf{F}_*^u|\mathbf{X}_*)$ is computed are given in the Appendix D. In fact, the explicit form of $q(\mathbf{F}_*^u|\mathbf{X}_*)$ takes the form of the projected process predictive distribution from sparse GPs (Csató and Opper, 2002; Smola and Bartlett, 2001; Seeger et al., 2003; Rasmussen and Williams, 2006):

$$q(\mathbf{F}_*^u|\mathbf{X}_*) = \mathcal{N}\left(\mathbf{F}_*^u|\mathbf{K}_{*u}\mathbf{B}, \mathbf{K}_{**} - \mathbf{K}_{*u}\left[\mathbf{K}_{uu}^{-1} - (\mathbf{K}_{uu} + \sigma^{-2}\mathbf{\Psi}_2)^{-1}\right]\mathbf{K}_{*u}^\top\right), \tag{39}$$

where $\mathbf{B} = \sigma^{-2}\left(\mathbf{K}_{uu} + \sigma^{-2}\mathbf{\Psi}_2\right)^{-1}\mathbf{\Psi}_1^\top\mathbf{Y}$, $\mathbf{K}_{**} = k_f(\mathbf{X}_*, \mathbf{X}_*)$ and $\mathbf{K}_{*u} = k_f(\mathbf{X}_*, \mathbf{X}_u)$. By substituting now the above Gaussian $q(\mathbf{F}_*^u|\mathbf{X}_*)$ in equation (37) and using the fact that

$q(\mathbf{X}_*)$ is also a Gaussian, we can analytically compute the mean and covariance of the predictive density which, based on the results of Girard et al. (2003), take the form

$$\mathbb{E}(\mathbf{F}_*^u) = \mathbf{B}^\top\mathbf{\Psi}_1^* \tag{40}$$

$$\mathrm{Cov}(\mathbf{F}_*^u) = \mathbf{B}^\top\left(\mathbf{\Psi}_2^* - \mathbf{\Psi}_1^*(\mathbf{\Psi}_1^*)^\top\right)\mathbf{B} + \psi_0^*\mathbf{I} - \mathrm{tr}\left(\left(\mathbf{K}_{uu}^{-1} - \left(\mathbf{K}_{uu} + \sigma^{-2}\mathbf{\Psi}_2\right)^{-1}\right)\mathbf{\Psi}_2^*\right)\mathbf{I}, \tag{41}$$

where $\psi_0^* = \mathrm{tr}\left(\langle\mathbf{K}_{**}\rangle\right)$, $\mathbf{\Psi}_1^* = \langle\mathbf{K}_{u*}\rangle$ and $\mathbf{\Psi}_2^* = \left\langle\mathbf{K}_{u*}\mathbf{K}_{u*}^\top\right\rangle$. All expectations are taken w.r.t. $q(\mathbf{X}_*)$ and can be calculated analytically for several kernel functions as explained in Section 3.2 and Appendix B. Using the above expressions and the Gaussian noise model of equation (6), the predicted mean of $\mathbf{Y}_*^u$ is equal to $\mathbb{E}\left[\mathbf{F}_*^u\right]$ and the predicted covariance (for each column of $\mathbf{Y}_*^u$) is equal to $\mathrm{Cov}(\mathbf{F}_*^u) + \sigma^2\mathbf{I}_{n_*}$.

4.2 Predictions in the Dynamical Model

The two prediction tasks described in the previous section for the standard variational GP-LVM can also be solved for the dynamical variant in a very similar fashion. Specifically, the two predictive approximate densities take exactly the same form as those in equations (36) and (37) while again the whole approximation relies on the maximisation of a variational lower bound $\mathcal{F}(q(\mathbf{X}, \mathbf{X}_*))$. However, in the dynamical case where the inputs $(\mathbf{X}, \mathbf{X}_*)$ are a priori correlated, the variational distribution $q(\mathbf{X}, \mathbf{X}_*)$ does not factorise across $\mathbf{X}$ and $\mathbf{X}_*$. This makes the optimisation of this distribution computationally more challenging, as it has to be optimised with respect to its all $2(n + n_*)q$ parameters. This issue is further explained in Appendix D.1.

Finally, we shall discuss how to solve the forecasting problem with our dynamical model. This problem is similar to the second predictive task described in Section 4.1, but now the observed set is empty. We can therefore write the predictive density similarly to equation (37) as follows:

$$p(\mathbf{Y}_*|\mathbf{Y}) \approx \int p(\mathbf{Y}_*|\mathbf{F}_*)q(\mathbf{F}_*|\mathbf{X}_*)q(\mathbf{X}_*)\mathrm{d}\mathbf{X}_*\mathrm{d}\mathbf{F}_*.$$

The inference procedure then follows exactly as before, using equations (37), (40) and (41). The only difference is that the computation of $q(\mathbf{X}_*)$ (associated with a fully unobserved $\mathbf{Y}_*$) is obtained from standard GP prediction and does not require optimisation, i.e.

$$q(\mathbf{X}_*) = \int p(\mathbf{X}_*|\mathbf{X})q(\mathbf{X})\mathrm{d}\mathbf{X} = \prod_{j=1}^{q}\int p(\mathbf{x}_{*,j}|\mathbf{x}_{:,j})q(\mathbf{x}_{:,j})\mathrm{d}\mathbf{x}_{:,j},$$

where $p(\mathbf{x}_{*,j}|\mathbf{x}_{:,j})$ is a Gaussian found from the conditional GP prior (see Rasmussen and Williams (2006)). Since $q(\mathbf{X})$ is Gaussian, the above is also a Gaussian with mean and variance given by

$$\boldsymbol{\mu}_{x_{*,j}} = \mathbf{K}_{*n}\bar{\boldsymbol{\mu}}_{:,j}$$
$$\mathrm{var}(\mathbf{x}_{*,j}) = \mathbf{K}_{**} - \mathbf{K}_{*n}(\mathbf{K}_x + \mathbf{\Lambda}_j^{-1})^{-1}\mathbf{K}_{n*},$$

where $\mathbf{K}_{*n} = k_x(\mathbf{t}_*, \mathbf{t})$, $\mathbf{K}_{*n} = \mathbf{K}_{n*}^\top$ and $\mathbf{K}_{**} = k_x(\mathbf{t}_*, \mathbf{t}_*)$. Notice that these equations have exactly the same form as found in standard GP regression problems.

5. Demonstration of the Variational Framework

In this section we investigate the performance of the variational GP-LVM and its dynamical extension. The variational GP-LVM allows us to handle very high dimensional data and, using ARD, to automatically infer the importance of each latent dimension. The generative construction allows us to impute missing values when presented with only a partial observation.

In the experiments, a latent space variational distribution is required as initialisation. We use PCA to initialise the $q-$dimensional means. The variances are initialised to values around 0.5, which are considered neutral given that the prior is a standard normal. The selection of q can be almost arbitrary and does not affect critically the end result, since the inverse lengthscales then switch off unnecessary dimensions. The only requirement is for q to be reasonably large in the first place, but an upper bound is $q = n$. In practice, in ad-hoc experiments we never observed any advantage in using $q > 40$, considering the dataset sizes employed. Inducing points are initialised as a random subset of the initial latent space. ARD inverse lengthscales are initialised based on a heuristic that takes into account the scale of each dimension. Specifically, the inverse squared lengthscale w_j is set as the inverse of the squared difference between the maximum and the minimum value of the initial latent mean in direction j. Following initialisation, the model is trained by optimising jointly all (hyper)parameters using the scaled conjugate gradients method. The optimisation is stopped until the change in the objective (variational lower bound) is very small.

We evaluate the models' performance in a variety of tasks, namely visualisation, prediction, reconstruction, generation of data or timeseries and class-conditional density estimation. Matlab source code for repeating the following experiments is available on-line from: `http://git.io/A3TN` and supplementary videos from: `http://git.io/A3t5`.

The experiments section is structured as follows; in Section 5.1 we outline the covariance functions used for the experiments. In Section 5.2 we demonstrate our method in a standard visualisation benchmark. In Section 5.3 we test both, the standard and dynamical variant of our method in a real-world motion capture dataset. In Section 5.4 we illustrate how our proposed model is able to handle a very large number of dimensions by working directly with the raw pixel values of high resolution videos. Additionally, we show how the dynamical model can interpolate but also extrapolate in certain scenarios. In Section 5.5 we consider a classification task on a standard benchmark, exploiting the fact that our framework gives access to the model evidence, thus enabling Bayesian classification.

5.1 Covariance Functions

Before proceeding to the actual evaluation of our method, we first review and give the forms of the covariance functions that will be used for our experiments. The mapping between the input and output spaces $\mathbf{X}$ and $\mathbf{Y}$ is nonlinear and, thus, we use the covariance function of equation (8) which also allows simultaneous model selection within our framework. In experiments where we use our method to also model dynamics, apart from the infinitely differentiable exponentiated quadratic covariance function defined in equation (3), we will also consider for the dynamical component the Matérn 3/2 covariance function which is only once differentiable, and a periodic one (Rasmussen and Williams, 2006; MacKay, 1998) which can be used when data exhibit strong periodicity. These covariance functions take

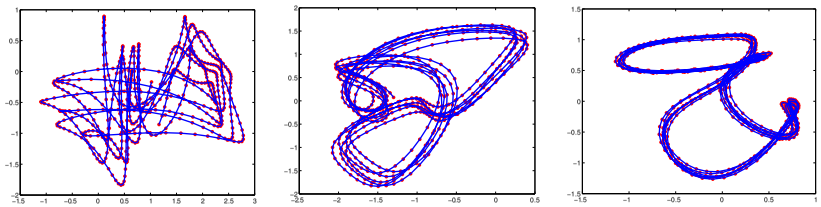
the form

$$k_{x(\text{mat})}\left(t_i, t_j\right) = \sigma^2_{\text{mat}}\left(1 + \frac{\sqrt{3}|t_i - t_j|}{\ell}\right)\exp\left(\frac{-\sqrt{3}|t_i - t_j|}{\ell}\right),$$

$$k_{x(\text{per})}\left(t_i, t_j\right) = \sigma^2_{\text{per}}\exp\left(-\frac{1}{2}\frac{\sin^2\left(\frac{2\pi}{T}\left(t_i - t_j\right)\right)}{\ell}\right),$$

where ℓ denotes the characteristic lengthscale and T denotes the period of the periodic covariance function.

Introducing a separate GP model for the dynamics is a very convenient way of incorporating any prior information we may have about the nature of the data in a nonparametric and flexible manner. In particular, more sophisticated covariance functions can be constructed by combining or modifying existing ones. For example, in our experiments we consider a compound covariance function, $k_{x(\text{per})} + k_{x(\text{rbf})}$ which is suitable for dynamical systems that are known to be only approximately periodic. The first term captures the periodicity of the dynamics whereas the second one corrects for the divergence from the periodic pattern by enforcing the datapoints to form smooth trajectories in time. By fixing the two variances, σ^2_{per} and σ^2_{rbf} to particular ratios, we are able to control the relative effect of each kernel. Example sample paths drawn from this compound covariance function are shown in Figure 4.



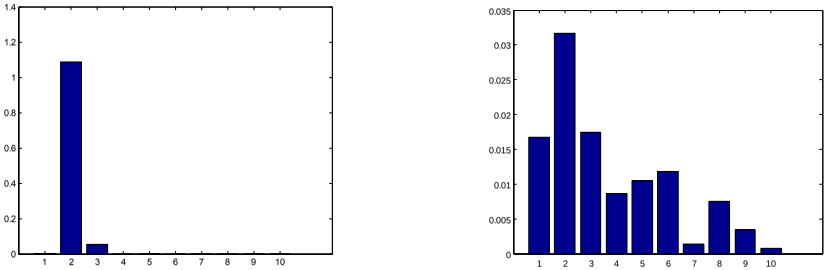
(a) (b) (c)

Figure 4: Typical sample paths drawn from the $k_{x(\text{per})} + k_{x(\text{rbf})}$ covariance function. The variances are fixed for the two terms, controlling their relative effect. In Figures (a), (b) and (c), the ratio $\sigma^2_{\text{rbf}}/\sigma^2_{\text{per}}$ of the two variances was large, intermediate and small respectively, causing the periodic pattern to be shifted accordingly each period.

For our experiments we additionally include a noise covariance function

$$k_{\text{white}}(\mathbf{x}_{i,:}, \mathbf{x}_{k,:}) = \theta_{\text{white}}\delta_{i,k},$$

where $\delta_{i,k}$ is the Kronecker delta. We can then define a compound kernel $k + k_{\text{white}}$, so that the noise level θ_{white} is jointly optimised along with the rest of the kernel hyperparameters. Similarly, we also include a bias term $\theta_{\text{bias}}\mathbf{1}$, where $\mathbf{1}$ denotes a vector of 1s.



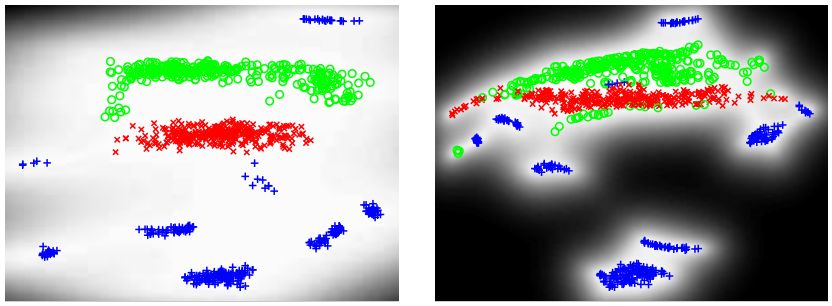
(a) Variational GP-LVM (b) GP-LVM

Figure 5: Left: The squared inverse lengthscales found by applying the variational GP-LVM with ARD EQ kernel on the oil flow data. Right: Results obtained for the standard GP-LVM with $q = 10$. These results demonstrate the ability of the variational GP-LVM to perform a "soft" automatic dimensionality selection. The inverse lengthscale for each dimension is associated with the expected number of the function's upcrossings in that particular direction; small values denote a more linear behaviour, whereas values close to zero denote an irrelevant dimension. For the variational GP-LVM, plot (a) suggests that the non-linearity is captured by dimension 2, as also confirmed by plot 6(a). On the other hand, plot (b) demonstrates the overfitting problem of the GP-LVM which is trained with MAP.

5.2 Visualisation Tasks

Given a dataset with known structure, we can apply our algorithm and evaluate its performance in a simple and intuitive way, by checking if the form of the discovered low dimensional manifold agrees with our prior knowledge.

We illustrate the method in the multi-phase oil flow data (Bishop and James, 1993) that consists of 1000, 12 dimensional observations belonging to three known classes corresponding to different phases of oil flow. This dataset is generated through simulation, and we know that the intrinsic dimensionality is 2 and the number of classes is 3. Figure 6 shows the results for these data obtained by applying the variational GP-LVM with 10 latent dimensions using the exponentiated quadratic ARD kernel. As shown in Figure 5(a), the algorithm switches off 8 out of 10 latent dimensions by making their inverse lengthscales almost zero. Therefore, the two-dimensional nature of this dataset is automatically revealed. Figure 6(a) shows the visualisation obtained by keeping only the dominant latent directions which are the dimensions 2 and 3. This is a remarkably high quality two dimensional visualisation of this data; to the best of our knowledge, our method is the only one that correctly picks up the true dimensionality and class separation at the same time, in a completely unsupervised manner. For comparison, Figure 6(b) shows the visualisation provided by the standard sparse GP-LVM that runs by a priori assuming only 2 latent dimensions. Both models use 50 inducing variables, while the latent variables $\mathbf{X}$ optimised in the standard



(a) Variational GP-LVM, $q = 10$ (2D projection) (b) GP-LVM, $q = 2$

Figure 6: Panel 6(a) shows the means of the variational posterior $q(\mathbf{X})$ for the variational GP-LVM, projected on the two dominant latent dimensions: dimension 2, plotted on the y-axis, and dimension 3 plotted on the x-axis. The plotted projection of a latent point $\mathbf{x}_{i,:}$ is assigned a colour according to the label of the corresponding output vector $\mathbf{x}_{i,:}$. The greyscale background intensities are proportional to the predicted variance of the GP mapping, if the corresponding locations were given as inputs. Plot 6(b) shows the visualisation found by standard sparse GP-LVM initialised with a two dimensional latent space.

GP-LVM are initialised based on PCA. Note that if we were to run the standard GP-LVM with 10 latent dimensions, the model would overfit the data and it would not reduce the dimensionality in the manner achieved by the variational GP-LVM, as illustrated in Figure 5(b). The quality of the class separation in the two-dimensional space can also be quantified in terms of the nearest neighbour error; the total error equals the number of training points whose closest neighbour in the latent space corresponds to a data point of a different class (phase of oil flow). The number of nearest neighbour errors made when finding the latent embedding for the variational GP-LVM is one. For the standard sparse GP-LVM it is 26, for the full GP-LVM with ARD kernel it is 8 and for the full GP-LVM with EQ kernel it is 2. Notice that all standard GP-LVMs were given the true dimensionality ($q = 2$) a priori.

5.3 Human Motion Capture Data

In this section we consider a data set associated with temporal information, as the primary focus of this experiment is on evaluating the dynamical version of the variational GP-LVM. We followed Taylor et al. (2007); Lawrence (2007) in considering motion capture data of walks and runs taken from subject 35 in the CMU motion capture database. We used the dynamical version of our model and treated each motion as an independent sequence. The data set was constructed and preprocessed as described in (Lawrence, 2007). This results in 2613 separate 59-dimensional frames split into 31 training sequences with an average length

of 84 frames each. Our model does not require explicit timestamp information, since we know a priori that there is a constant time delay between poses and the model can construct equivalent covariance matrices given any vector of equidistant time points.

The model is jointly trained, as explained in the last paragraph of Section 3.3.2, on both walks and runs, i.e. the algorithm learns a common latent space for these motions. As in (Lawrence, 2007), we used 100 inducing points. At test time we investigate the ability of the model to reconstruct test data from a previously unseen sequence given partial information for the test targets. This is tested once by providing only the dimensions which correspond to the body of the subject and once by providing those that correspond to the legs. We compare with results in (Lawrence, 2007), which used MAP approximations for the dynamical models, and against nearest neighbour. We can also indirectly compare with the binary latent variable model (BLV) of Taylor et al. (2007) which used a slightly different data preprocessing. Furthermore, we additionally tested the non-dynamical version of our model, in order to explore the structure of the distribution found for the latent space. In this case, the notion of sequences or sub-motions is not modelled explicitly, as the non-dynamical approach does not model correlations between datapoints. However, as will be shown below, the model manages to discover the dynamical nature of the data and this is reflected in both, the structure of the latent space and the results obtained on test data.

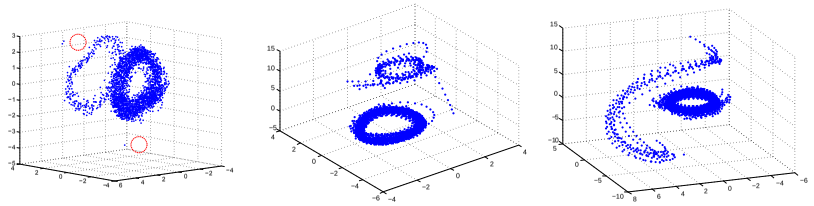
The performance of each method is assessed by using the cumulative error per joint in the scaled space defined in (Taylor et al., 2007) and by the root mean square error in the angle space suggested by Lawrence (2007). Our models were initialised with nine latent dimensions. For the dynamical version, we performed two runs, once using the Matérn covariance function for the dynamical prior and once using the exponentiated quadratic.

The appropriate latent space dimensionality for the data was automatically inferred by our models. The non-dynamical model selected a 5-dimensional latent space. The model which employed the Matérn covariance to govern the dynamics retained four dimensions, whereas the model that used the exponentiated quadratic kept only three. The other latent dimensions were completely switched off by the ARD parameters.

From Table 1 we see that the dynamical variational GP-LVM considerably outperforms the other approaches. The best performance for the legs and the body reconstruction was achieved by our dynamical model that used the Matérn and the exponentiated quadratic covariance function respectively. This is an intuitive result, since the smoother body movements are expected to be better modelled using the infinitely differentiable exponentiated quadratic covariance function, whereas the Matérn one can easier fit the rougher leg motion. Although it is not always obvious how to choose the best covariance function (without expensive cross-validation), the fact that both models outperform significantly other approaches shows that the Bayesian training manages successfully to fit the covariance function parameters to the data in any case. Furthermore, the non-dynamical variational GP-LVM, not only manages to discover a latent space with a dynamical structure, as can be seen in Figure 7(a), but is also proven to be quite robust when making predictions. Indeed, Table 1 shows that the non-dynamical variational GP-LVM performs comparably to nearest neighbor. However, the standard GP-LVM which explicitly models dynamics using MAP approximations performs slightly better than the non-dynamical variational GP-LVM; this suggests that temporal information is crucial in this dataset. Finally, it is worth highlighting the intuition gained by investigating Figure 7. As can be seen, all models split the

Data	CL	CB	L	L	B	B
Error Type	SC	SC	SC	RA	SC	RA
BLV	11.7	**8.8**	-	-	-	-
NN sc.	22.2	**20.5**	-	-	-	-
GP-LVM (q= 3)	-	-	11.4	3.40	16.9	2.49
GP-LVM (q= 4)	-	-	9.7	3.38	20.7	2.72
GP-LVM (q= 5)	-	-	13.4	4.25	23.4	2.78
NN sc.	-	-	13.5	4.44	20.8	2.62
NN	-	-	14.0	4.11	30.9	3.20
VGP-LVM	-	-	14.22	5.09	18.79	2.79
Dyn. VGP-LVM (Exp. Quadr.)	-	-	7.76	3.28	**11.95**	**1.90**
Dyn. VGP-LVM (Matérn 3/2)	-	-	**6.84**	**2.94**	13.93	2.24

Table 1: Errors obtained for the motion capture dataset. The format of this table follows Lawrence (2007), where differently processed datasets were used for the first two columns, as opposed to the last four columns. Specifically, CL / CB are the leg and body data sets as preprocessed in (Taylor et al., 2007), L and B the corresponding datasets from Lawrence. Taylor et al. also used a different scaling for the data, which can be applied to the predictions of the models trained in the L/B datasets to obtain indirect comparisons with the models which were trained in the CL/CB datasets. Specifically, SC corresponds to the error in the scaled space, as in Taylor et al. while RA is the error in the angle space. The methods shown in the table are: nearest neighbour in the angle space (NN) and in the scaled space (NN sc.), GP-LVM (with different pre-selected latent dimensionality q), binary latent variable model (BLV), variational GP-LVM (VGP-LVM) and Dynamical variational GP-LVM (Dyn. VGP-LVM). Notice that NN was run once in the CL/CB dataset and once in the L/B dataset, so as to provide a "link" between the two first and the four last columns. The best error per column is in bold.



(a) (b) (c)

Figure 7: The latent space discovered by our models for the human motion capture data, projected into its three principal dimensions. The latent space found by the non-dynamical variational GP-LVM is shown in (a), by the dynamical model which uses the Matérn in (b) and by the dynamical model which uses the exponentiated quadratic in (c). The red, dotted circles highlight three "outliers".

encoding for the "walk" and "run" regimes into two subspaces. Further, we notice that the smoother the latent space is constrained to be, the less "circular" is the shape of the "run" regime latent space encoding. This can be explained by noticing the "outliers" in the top left and bottom positions of plot (a), highlighted with a red, dotted circle. These latent points correspond to training positions that are very dissimilar to the rest of the training set but, nevertheless, a temporally constrained model is forced to accommodate them in a smooth path. The above intuitions can be confirmed by interacting with the model in real time graphically, as is presented in the supplementary video.

5.4 Modeling Raw High Dimensional Video Sequences

For this set of experiments we considered video sequences (which are included in the supplementary videos available on-line). Such sequences are typically preprocessed before modelling to extract informative features and reduce the dimensionality of the problem. Here we work directly with the raw pixel values to demonstrate the ability of the dynamical variational GP-LVM to model data with a vast number of features. This also allows us to directly sample video from the learned model.

Firstly, we used the model to reconstruct partially observed frames from test video sequences.[3] For the first video discussed here we gave as partial information approximately 50% of the pixels while for the other two we gave approximately 40% of the pixels on each frame. The mean squared error per pixel was measured to compare with the $k-$nearest neighbour (NN) method, for $k \in (1, .., 5)$ (we only present the error achieved for the best choice of k in each case). The datasets considered are the following: firstly, the 'Missa' dataset, a standard benchmark used in image processing. This is a 103680-dimensional

3. 'Missa' dataset: cipr.rpi.edu. 'Ocean': cogfilms.com. 'Dog': fitfurlife.com. See details in supplementary on-line videos. The logo appearing in the 'dog' images in the experiments that follow, has been added with post-processing.

video, showing a woman talking for 150 frames. The data is challenging as there are translations in the pixel space. We also considered an HD video of dimensionality 9×10^5 that shows an artificially created scene of ocean waves as well as a $230400-$dimensional video showing a dog running for 60 frames. The latter is approximately periodic in nature, containing several paces from the dog. For all video datasets, the GPs were trained with $m = n$. For the first two videos we used the Matérn and exponentiated quadratic covariance functions respectively to model the dynamics and interpolated to reconstruct blocks of frames chosen from the whole sequence. For the 'dog' dataset we constructed a compound kernel $k_x = k_{x(\text{rbf})} + k_{x(\text{per})}$ presented in Section 5.1, where the exponentiated quadratic (RBF) term is employed to capture any divergence from the approximately periodic pattern. The variance of the periodic component was fixed to 1 and the variance of the RBF component to 1/150. This is to make sure that the RBF does not dominate before learning some periodicity (in this case periodicity is more difficult to discover as a pattern). The selection of the variances' ratio does not need to be exact and here was made in an ad-hoc manner, aiming at getting samples like the one shown in Figure 4(b). We then used our model to reconstruct the last 7 frames extrapolating beyond the original video. As can be seen in Table 2, our method outperformed NN in all cases. The results are also demonstrated visually in Figures 8, 9, 10 and 11 and the reconstructed videos are available in the supplementary on-line videos.

	Missa	Ocean	Dog
Dyn. VGP-LVM	2.52	9.36	4.01
NN	2.63	9.53	4.15

Table 2: The mean squared error per pixel for Dyn. VGP-LVM and NN for the three datasets (measured only in the missing inputs).

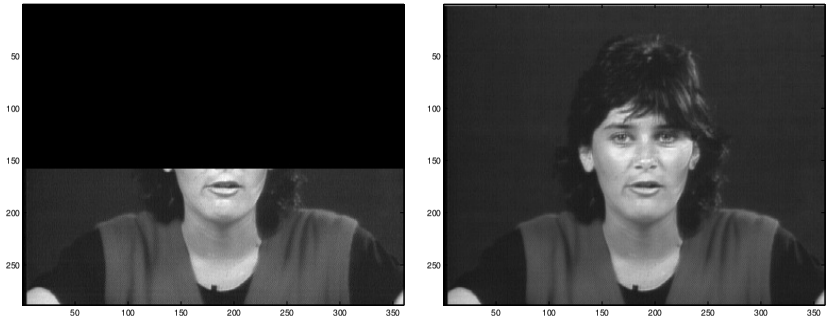
As can be seen in Figures 8, 9 and 10, the dynamical variational GP-LVM predicts pixels which are smoothly connected with the observed part of the image, whereas the NN method cannot fit the predicted pixels in the overall context. Figure 8(c) focuses on this specific problem with NN, but it can be seen more evidently in the corresponding video files.

As a second task, we used our generative model to create new samples and generate a new video sequence. This is most effective for the 'dog' video as the training examples were approximately periodic in nature. The model was trained on 60 frames (time-stamps $[t_1, t_{60}]$) and we generated new frames which correspond to the next 40 time points in the future. The only input given for this generation of future frames was the time-stamp vector, $[t_{61}, t_{100}]$. The results show a smooth transition from training to test and amongst the test video frames. The resulting video of the dog continuing to run is sharp and high quality. This experiment demonstrates the ability of the model to reconstruct massively high dimensional images without blurring. Frames from the result are shown in Figure 13. The full sequence is available in the supplementary on-line videos.

(a) (b)

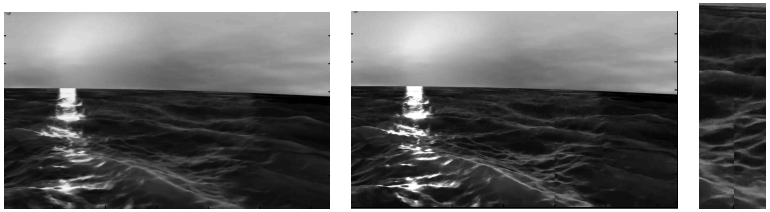
(c)

Figure 8: (a) and (c) demonstrate the reconstruction achieved by dynamical variational GP-LVM and NN respectively for one of the most challenging frames (b) of the 'missa' video, i.e. when translation occurs. In contrast to the NN method, which works in the whole high dimensional pixel space, our method reconstructed the images using a "compressed" latent space. The ARD scales for this example revealed an effectively 12−dimensional latent space.



(a) (b)

Figure 9: Another example of the reconstruction achieved by the dynamical variational GP-LVM given the partially observed image.



(a) (b) (c)

Figure 10: (a) (Dynamical variational GP-LVM) and (b) (NN) depict the reconstruction achieved for a frame of the 'ocean' dataset. Notice that in both of the aforementioned datasets, our method recovers a smooth image, in contrast to the simple NN (a close up of this problem with NN for the 'ocean' video is shown in Figure (c)). The dynamical var. GP-LVM reconstructed the ocean images using a latent space compression of the video defined by 9 effective dimensions (the rest of the inverse lengthscales approached zero).

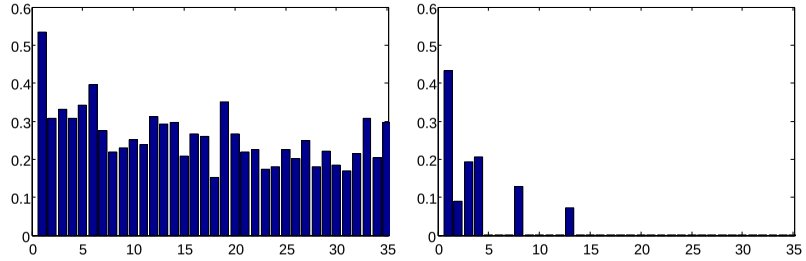
5.5 Class Conditional Density Estimation

In this experiment we use the variational GP-LVM to build a generative classifier for handwritten digit recognition. We consider the well known USPS digits dataset. This dataset consists of 16×16 images for all 10 digits and it is divided into 7291 training examples and 2007 test examples. We ran 10 variational GP-LVMs, one for each digit, on the USPS data base. We used 10 latent dimensions and 50 inducing variables for each model. This allowed us to build a probabilistic generative model for each digit so that we can compute

(a)

(b)

Figure 11: An example for the reconstruction achieved for the 'dog' dataset. 40% of the test image's pixels (Figure (a)) were presented to the model, which was able to successfully reconstruct them, as can be seen in (b).



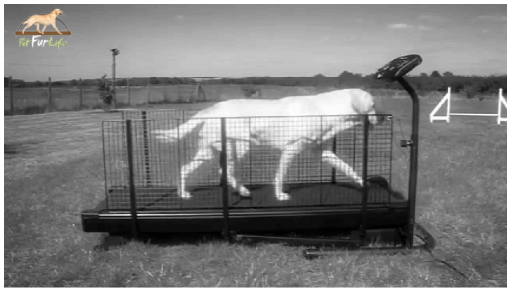
(a) Initialised squared inverse lengthscales (b) Optimised squared inverse lengthscales

Figure 12: The figure demonstrates the ability of the model to automatically estimate an effective latent dimensionality by showing the initial squared inverse lengthscales (fig: (a)) of the ARD covariance function and the values obtained after training (fig: (b)) on the 'dog' data set.

Bayesian class conditional densities in the test data having the form $p(\mathbf{Y}_*|\mathbf{Y}, \text{digit})$. These class conditional densities are approximated through the ratio of lower bounds in equation (36) as described in Section 4. The whole approach allows us to classify new digits by determining the class labels for test data based on the highest class conditional density value and using a uniform prior over class labels. We used the following comparisons: firstly, a logistic classification approach. Secondly, a vanilla SVM from scikit-learn (Pedregosa et al., 2011), for which the error parameter C was selected with $5-$fold cross validation. Thirdly, a GP classification approach with EP approximation from GPy (authors, 2014). Lastly, the recent variational inference based GP classification approach of Hensman et al. (2014), referred to as "GP classification with VI" and taken from the GPy (authors, 2014) implementation. All of these methods operated in a 1-vs-all setting. The results of our experiments are shown in Table 3. In addition to the standard baselines reported here, more sophisticated schemes (many of which result in better performance) have been tried in this dataset by other researchers; a summary of previously published results can be found in (Keysers et al., 2002).

6. Extensions for Different Kinds of Inputs

So far we considered the typical dimensionality reduction scenario where, given high-dimensional output data we seek to find a low-dimensional latent representation in a completely unsupervised manner. For the dynamical variational GP-LVM we have additional temporal information, but the input space $\mathbf{X}$ from where we wish to propagate the uncertainty is still treated as fully unobserved. However, our framework for propagating the input uncertainty through the GP mapping is applicable to the full spectrum of cases, ranging from fully unobserved to fully observed inputs with known or unknown amount of uncertainty per input. In this section we discuss these cases and, further, show how they give



(a)

(b)

(c)

Figure 13: The last frame of the training video (a) is smoothly followed by the first frame (b) of the generated video. A subsequent generated frame can be seen in (c).

	# misclassified	error (%)
variational GP-LVM ($m = 50$)	**95**	**4.73** %
1-vs-all Logistic Regression	283	14.10 %
1-vs-all GP classification with VI ($m = 50$)	100	4.98 %
1-vs-all GP classification with VI ($m = 150$)	100	4.98 %
1-vs-all GP classification with VI ($m = 250$)	99	4.93 %
1-vs-all GP classification with EP ($m = 50$)	139	6.93 %
1-vs-all GP classification with EP ($m = 150$)	128	6.38 %
1-vs-all GP classification with EP ($m = 250$)	126	6.28 %
1-vs-all SVM	119	5.92 %

Table 3: The test error made for classifying the whole set of 2007 test points (USPS digits) by the variational GP-LVM, 1-vs-all Logistic Regression, SVM classification and two types of GP classification.

rise to an auto-regressive model (Section 6.1) and a GP regression variant which can handle missing inputs (Section 6.2).

6.1 Gaussian Process Inference with Uncertain Inputs

Gaussian processes have been used extensively and with great success in a variety of regression tasks. In the most common setting, we are given a dataset of observed input-output pairs, denoted as $\mathbf{Z} \in \Re^{n\times q}$ and $\mathbf{Y} \in \Re^{n\times p}$ respectively, and we wish to infer the unknown outputs $\mathbf{Y}^* \in \Re^{n^*\times p}$ corresponding to some novel given inputs $\mathbf{Z}^* \in \Re^{n^*\times q}$. However, in many real-world applications the inputs are uncertain, for example when measurements come from noisy sensors. In this case, the GP methodology cannot be trivially extended to account for the variance associated with the input space (Girard et al., 2003; McHutchon and Rasmussen, 2011). The aforementioned problem is also closely related to the field of heteroscedastic Gaussian process regression, where the uncertainty in the noise levels is modelled in the output space as a function of the inputs (Kersting et al., 2007; Goldberg et al., 1998; Lázaro-Gredilla and Titsias, 2011).

In this section we show that our variational framework can be used to explicitly model the input uncertainty in the GP regression setting. The assumption made is that the inputs $\mathbf{X}$ are not observed directly but, rather, we only have access to their noisy versions $\{\mathbf{z}_{i,:}\}_{i=1}^n = \mathbf{Z} \in \Re^{n\times q}$. The relationship between the noisy and true inputs is given by assuming the noise to be Gaussian,

$$\mathbf{x}_{i,:} = \mathbf{z}_{i,:} + (\boldsymbol{\epsilon}_x)_{i,:},$$

where $\boldsymbol{\epsilon}_x \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_x)$, as in (McHutchon and Rasmussen, 2011). Since $\mathbf{Z}$ is observed and $\mathbf{X}$ unobserved the above equation essentially induces a Gaussian prior distribution over $\mathbf{X}$ that has the form

$$p(\mathbf{X}|\mathbf{Z}) = \prod_{i=1}^{n} \mathcal{N}(\mathbf{x}_{i,:}|\mathbf{z}_{i,:}, \boldsymbol{\Sigma}_x),$$

where $\mathbf{\Sigma}_x$ is typically an unknown parameter. Given that $\mathbf{X}$ are really the inputs that eventually are passed through the GP latent function (to subsequently generate the outputs) the whole probabilistic model becomes a GP-LVM with the above special form for the prior distribution over the latent inputs, making thus our variational framework easily applicable. More precisely, using the above prior, we can define a variational bound on $p(\mathbf{Y})$ as well as an associated approximation $q(\mathbf{X})$ to the true posterior $p(\mathbf{X}|\mathbf{Y},\mathbf{Z})$. This variational distribution $q(\mathbf{X})$ can be used as a probability estimate of the noisy input locations $\mathbf{X}$. During optimisation of the lower bound we can also learn the parameter $\mathbf{\Sigma}_x$. Furthermore, if we wish to reduce the number of parameters in the variational distribution $q(\mathbf{X}) = \mathcal{N}(\mathcal{M},\mathcal{S})$ a sensible choice would be to set $\mathcal{M} = \mathbf{Z}$, although such a choice may not be optimal. However, this choice also allows us to incorporate $\mathbf{Z}$ directly in the approximate posterior and, hence, we may also remove the coupling in the prior (coming from $\mathbf{\Sigma}_x$) by instead considering a standard normal for $p(\mathbf{X})$. This is the approach taken in this paper.

Having a method which implicitly models the uncertainty in the inputs also allows for doing predictions in an autoregressive manner while propagating the uncertainty through the predictive sequence (Girard et al., 2003). To demonstrate this in the context of our framework, we will take the simple case where the process of interest is a multivariate time-series given as pairs of time points $\mathbf{t} = \{t\}_{i=1}^n$ and corresponding output locations $\mathbf{Y} = \{\mathbf{y}_{i,:}\}_{i=1}^n$, $\mathbf{y}_{i,:} \in \Re^p$. Here, we take the time locations to be deterministic and equally spaced, so that they can be simply denoted by the subscript of the output points $\mathbf{y}_{i,:}$; we thus simply denote with $\mathbf{y}_k$ the output point $\mathbf{y}_{k,:}$ which corresponds to t_k.

We can now reformat the given data $\mathbf{Y}$ into input-output pairs $\hat{\mathbf{Z}}$ and $\hat{\mathbf{Y}}$, where

$$[\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2, ..., \hat{\mathbf{z}}_{n-\tau}] = [[\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_\tau], [\mathbf{y}_2, \mathbf{y}_3, ..., \mathbf{y}_{\tau+1}], ..., [\mathbf{y}_{n-\tau}, \mathbf{y}_{n-\tau+1}, ..., \mathbf{y}_{n-1}]],$$
$$[\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, ..., \hat{\mathbf{y}}_{n-\tau}] = [\mathbf{y}_{\tau+1}, \mathbf{y}_{\tau+2}, ..., \mathbf{y}_n]$$

and τ is the size of the dynamics' "memory". In other words, we define a window of size τ which shifts in time so that the output in time t becomes an input in time $t+1$. Therefore, the uncertain inputs method described earlier in this section can be applied to the new dataset $[\hat{\mathbf{Z}}, \hat{\mathbf{Y}}]$. In particular, although the *training* inputs $\hat{\mathbf{Z}}$ are not necessarily uncertain in this case, the aforementioned way of performing inference is particularly advantageous when the task is extrapolation.

In more detail, consider the simplest case described in this section where the posterior $q(\mathbf{X})$ is centered in the given noisy inputs and we allow for variable noise around the centers. To perform extrapolation one firstly needs to train the model on the dataset $[\hat{\mathbf{Z}}, \hat{\mathbf{Y}}]$. Then, we can perform iterative $k-$step ahead prediction in order to find a future sequence $[\mathbf{y}_{n+1}, \mathbf{y}_{n+2}, ...]$ where, similarly to the approach taken by Girard et al. (2003), the predictive variance in each step is accounted for and propagated in the subsequent predictions. For example, if $k = 1$ the algorithm will make iterative 1-step predictions in the future; in the beginning, the output $\mathbf{y}_{n+1}$ will be predicted given the training set. In the next step, the training set will be augmented to include the previously predicted $\mathbf{y}_{n+1}$ as part of the input set, where the predictive variance is now encoded as the uncertainty of this point.

The advantage of the above method, which resembles a state-space model, is that the future predictions do not almost immediately revert to the mean, as in standard station-

ary GP regression, neither do they underestimate the uncertainty, as would happen if the predictive variance was not propagated through the inputs in a principled way.

6.1.1 Demonstration: iterative $k-$step ahead forecasting

Here we demonstrate our framework in the simulation of a state space model, as was described previously. More specifically, we consider the Mackey-Glass chaotic time series, a standard benchmark which was also considered in (Girard et al., 2003). The data is one-dimensional so that the timeseries can be represented as pairs of values $\{\mathbf{y}, \mathbf{t}\}, t = 1, 2, \cdots, n$ and simulates:

$$\frac{\mathrm{d}\zeta(t)}{dt} = -b\zeta(t) + \alpha\frac{\zeta(t-T)}{1+\zeta(t-T)^{10}}, \quad \text{with} \quad \alpha = 0.2, b = 0.1, T = 17.$$

As can be seen, the generating process is very non-linear, something which makes this dataset particularly challenging. The created dataset is in uniform time-steps.

The model trained on this dataset was the one described previously, where the modified dataset $\{\hat{\mathbf{y}}, \hat{\mathbf{z}}\}$ was created with $\tau = 18$ and we used the first $4\tau = 72$ points for training and predicted the subsequent 1110 points in the future. 30 inducing points were used. We firstly compared to a standard GP model where the input - output pairs were given by the modified dataset $\{\hat{\mathbf{z}}, \hat{\mathbf{y}}\}$ that was mentioned previously; this model is here referred to as the "naive autoregressive GP" model $\mathcal{GP}_{\hat{\mathbf{z}},\hat{\mathbf{y}}}$. For this model, the predictions are made in the $k-$step ahead manner, according to which the predicted values for iteration k are added to the training set. However, this standard GP model has no straight forward way of propagating the uncertainty, and therefore the input uncertainty is zero for every step of the iterative predictions. We also compared against a special case of the variational GP-LVM which implements the functionality developed by Girard et al. (2003). In this version, predictions at every step are performed on a noisy location, i.e. by incorporating the predictive variance of the previous step. In contrast to our algorithm, however, the predictive point is not incorporated as noisy input after the prediction but, rather, discarded. This method is here referred to as $\mathcal{GP}_{\text{uncert}}$. Although we use $\mathcal{GP}_{\text{uncert}}$ as an informative baseline, we note that in the original paper of Girard et al. (2003), additional approximations were implemented, by performing Taylor expansion around the predictive mean and variance in each step. The predictions obtained for all competing methods can be seen in Figure 14.

As shown in the last plot, both the variational GP-LVM and $\mathcal{GP}_{\text{uncert}}$ are robust in handling the uncertainty throughout the predictions; $\mathcal{GP}_{\hat{\mathbf{z}},\hat{\mathbf{y}}}$ underestimates the uncertainty. Consequently, as can be seen from the top three plots, in the first few predictions all methods give the same answer. However, once the predictions of $\mathcal{GP}_{\hat{\mathbf{z}},\hat{\mathbf{y}}}$ diverge a little by the true values, the error is carried on and amplified due to underestimating the uncertainty. On the other hand, $\mathcal{GP}_{\text{uncert}}$ perhaps overestimates the uncertainty and, therefore, is more conservative in its predictions, resulting in higher errors. Quantification of the error is shown in Table 4 and further validates the above discussion. The negative log. probability density for the predicted sequence was also computed for each method. The obtained values were then divided by the length of the predicted sequence to obtain an average per point. The result is 22447 for our method, 30013 for that of Girard et al. (2003) and 36583 for the "naive" GP method.

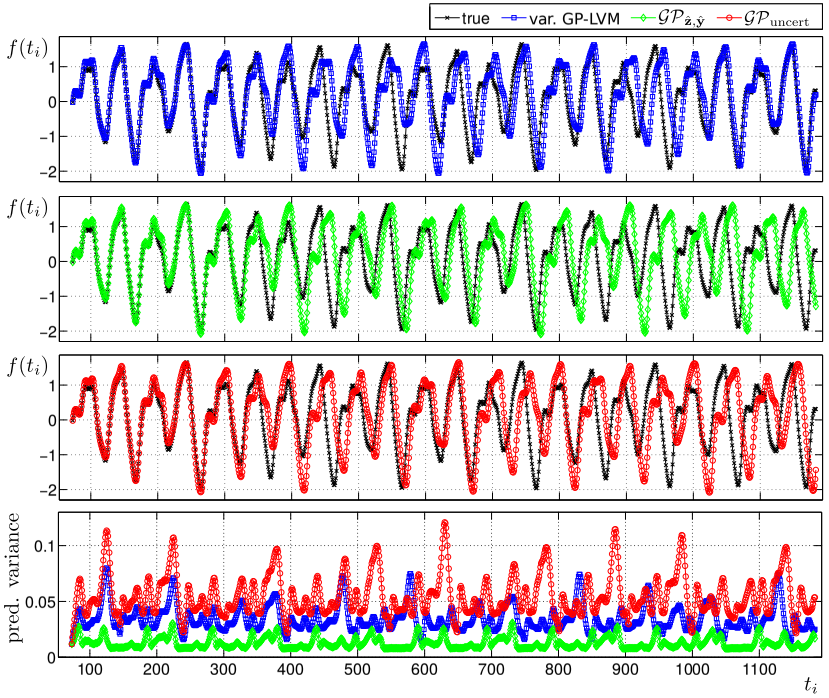


Figure 14: Iterative 1−step ahead prediction for a chaotic timeseries. From top to bottom, the following methods are compared: the variational GP-LVM, a "naive" autoregressive GP approach which does not propagate uncertainties ($\mathcal{GP}_{\hat{\mathbf{z}},\hat{\mathbf{y}}}$) and the approach of Girard et al. (2003) ($\mathcal{GP}_{\text{uncert}}$) implemented as a specific case of the variational GP-LVM. The plot at the bottom shows the predictive variances. The $x-$axis is common for all plots, representing the extrapolation step.

6.2 GP Regression with Missing Inputs and Data Imputation

In standard GP regression we assume that the inputs and the outputs are fully observed. However, in many realistic scenarios missing values can occur. The case where the missing values occur in the outputs (known as semi-supervised learning) has been addressed in the past in the context of GPs, e.g. by Lawrence and Jordan (2005); Sindhwani et al. (2007). However, handling partially observed *input* data in a fully probabilistic way is challenging, since propagating the uncertainty from the input to the output space is intractable. Girard et al. (2003); Quiñonero-Candela et al. (2003) provide an approximation, but their solution does not support uncertain *training* inputs.

Method	MAE	MSE
var. GP-LVM	**0.529**	**0.550**
$\mathcal{GP}_{\text{uncert}}$	0.700	0.914
$\mathcal{GP}_{\hat{\mathbf{z}},\hat{\mathbf{y}}}$	0.799	1.157

Table 4: Mean squared and mean absolute error obtained when extrapolating in the chaotic time-series data. $\mathcal{GP}_{\text{uncert}}$ refers to the method of Girard et al. (2003) implemented as a specific case of our framework and $\mathcal{GP}_{\hat{\mathbf{z}},\hat{\mathbf{y}}}$ refers to the "naive" autoregressive GP approach which does not propagate uncertainties. The lowest errors (achieved by our method) are in bold.

In this section, we describe how our proposed model can be used in a data imputation problem where part of the training inputs are missing. This scenario is here treated as a special case of the uncertain input modelling discussed above. Although a more general setting can be defined, here we consider the case where we have a fully and a partially observed set of inputs, i.e. $\mathbf{Z} = (\mathbf{Z}^o, \mathbf{Z}^u)$, where o and u denote set of rows of $(\mathbf{Z}, \mathbf{Y})$ that contain fully and partially observed inputs respectively[4]. This is a realistic scenario; it is often the case that certain input features are more difficult to obtain (e.g. human specified tags) than others, but we would nevertheless wish to model all available information within the same model. The features missing in $\mathbf{Z}^u$ can be different in number / location for each individual point $\mathbf{z}^u_{i,:}$.

A standard GP regression model cannot straightforwardly model jointly $\mathbf{Z}^o$ and $\mathbf{Z}^u$. In contrast, in our framework the inputs are replaced by distributions $q(\mathbf{X}^o)$ and $q(\mathbf{X}^u)$, so that $\mathbf{Z}^u$ can be taken into account naturally by simply initialising the uncertainty of $q(\mathbf{X}^u)$ in the missing locations to 1 (assuming normalized inputs) and the mean to the empirical mean and then, optionally, optimising $q(\mathbf{X}^u)$. In our experiments we use a slightly more sophisticated approach which resulted in better results. Specifically, we can use the fully observed data subset $(\mathbf{Z}^o, \mathbf{Y}^o)$ to train an initial model for which we fix $q(\mathbf{X}^o) = \mathcal{N}(\mathbf{X}^o|\mathbf{Z}^o, \varepsilon \rightarrow \mathbf{0})$. Given this model, we can then use $\mathbf{Y}^u$ to estimate the predictive posterior $q(\mathbf{X}^u)$ in the missing locations of $\mathbf{Z}^u$ (for the observed locations we match the mean with the observations, as for $\mathbf{Z}^o$). After initialising $q(\mathbf{X}) = q(\mathbf{X}^o, \mathbf{X}^u)$ in this way, we can proceed by training our model on the full (extended) training set $((\mathbf{Z}^o, \mathbf{Z}^u), (\mathbf{Y}^o, \mathbf{Y}^u))$, which contains fully and partially observed inputs. During this training phase, the variational distribution $q(\mathbf{X})$ is held fixed in the locations corresponding to observed values and is optimised in the locations of missing inputs. Considering a distribution $q(\mathbf{X})$ factorised w.r.t data points and constrained with $\mathbf{Z}$ as explained above might not be an optimal choice with respect to the true posterior. However, this approach allows us to incorporate knowledge of the observed locations without adding extra computational cost to the framework.

Given the above formulation, we define a new type of GP model referred to as "*missing inputs GP*". This model naturally incorporates fully and partially observed examples by

4. In section 4, the superscript u denoted the set of missing *columns from test outputs*. Here it refers to *rows of training inputs* that are *partially* observed, i.e. the union of o and u is now $\{1, \cdots, n\}$.

communicating the uncertainty throughout the relevant parts of the model in a principled way. Specifically, the predictive uncertainty obtained by the initial model trained on the fully observed data is incorporated as input uncertainty via $q(\mathbf{X}^u)$ in the model trained on the extended dataset, similarly to how extrapolation was achieved for our auto-regressive approach in Section 6.1. In extreme cases resulting in very non-confident predictions, for example presence of outliers, the corresponding locations will simply be ignored automatically due to the large uncertainty. This mechanism, together with the subsequent optimisation of $q(\mathbf{X}^u)$, guards against reinforcing bad predictions when imputing missing values after learning from small training sets. Details of the algorithm for this approach are given in Appendix E.

Although the focus of this section was on handling missing inputs, the algorithm developed above has conceptual similarities with procedures followed to solve the missing outputs (semi-supervised learning) problem. Specifically, our generative method treats the missing values task as a data imputation problem, similarly to (Kingma et al., 2014). Furthermore, to perform data imputation our algorithm trains an initial model on the fully observed portion of the data, used to predict the missing values. This draws inspiration from self-training methods used for incorporating unlabelled examples in classification tasks (Rosenberg et al., 2005). In a *bootstrap*-based self-training approach this incorporation is achieved by predicting the missing labels using the initial model and, subsequently, augmenting the training set using only the confident predictions subset. However, our approach differs from bootstrap-based self-training methods in two key points: firstly, the partially unobserved set is in the input rather than the output space; secondly, the predictions obtained from the "self-training" step of our method only constitute initialisations which are later optimised along with model parameters. Therefore, we refer to this step of our algorithm as *partial* self-training. Further, in our framework the predictive uncertainty is not used as a hard measure of discarding unconfident predictions but, instead, we allow all values to contribute according to an optimised uncertainty measure. Therefore, the way in which uncertainty is handled makes the "self-training" part of our algorithm principled compared to many bootstrap-based approaches.

6.2.1 Demonstration

In this section we consider simulated and real-world data to demonstrate our missing inputs GP algorithm, which was discussed in Section 6.2. The simulated data were created by sampling inputs $\mathbf{Z}$ from an unknown to the competing models GP and gave this as input to another (again, unknown) GP to obtain the corresponding outputs $\mathbf{Y}$. For the real-world data demonstration we considered a subset of the same motion capture dataset discussed in Section 5.3, which corresponds to a walking motion of a human body represented as a set of 59 joint locations. We formulated a regression problem where the first 20 dimensions of the original data are used as targets and the rest 39 as inputs. In other words, given a partial joint representation of the human body, the task is to infer the rest of the representation; that is, given fully observed test inputs $\mathbf{Z}_*$ we wish to reconstruct test outputs $\mathbf{Y}_*$. For both datasets, simulated and motion capture, we selected a portion of the training inputs, denoted as $\mathbf{Z}^u$, to have randomly missing features. The extended dataset $((\mathbf{Z}^o, \mathbf{Z}^u), (\mathbf{Y}^o, \mathbf{Y}^u))$ was used to train our method as well as a nearest neighbour (NN) method. The NN method

compares a test instance $\mathbf{z}_*$ to each training instance by only taking into account the dimensions that are observed in the training point. This gives a noisy similarity measure in the input space. The predicted output $\mathbf{y}_*$ is then taken to be the training output for which the corresponding input has the largest similarity (according to the above noisy measure). We further compared to a standard GP, which was trained using only the observed data $(\mathbf{Z}^o, \mathbf{Y}^o)$, since it cannot handle missing inputs straightforwardly.

For the simulated data we used the following sizes: $|\mathbf{Z}^o| = 40$, $|\mathbf{Z}^u| = 60$, $|\mathbf{Z}_*| = 100$ and $m = 30$. The dimensionality of the inputs is 15 and of the outputs is 5. For the motion capture data we used $|\mathbf{Z}^o| = 50$, $|\mathbf{Z}^u| = 80$, $|\mathbf{Z}_*| = 200$ and $m = 35$. In Figure 15 we plot the MSE obtained by the competing methods for a varying percentage of missing features in $\mathbf{Z}^u$. For the simulated data experiment, each of the points in the plot is an average of 4 runs which considered different random seeds. As can be seen, the missing inputs GP is able to handle the extra data and make better predictions, even if a very large portion is missing. Indeed, its performance starts to converge to that of a standard GP when there are 90% missing values in $\mathbf{Z}^u$ and performs identically to the standard GP when 100% of the values are missing. In Appendix F.2 we also provide a comparison to multiple linear regression (MLR) (Chatterjee and Hadi, 1986) and to the mean predictor. These methods gave very bad results, and for clarity they were not included in the main Figure 15.

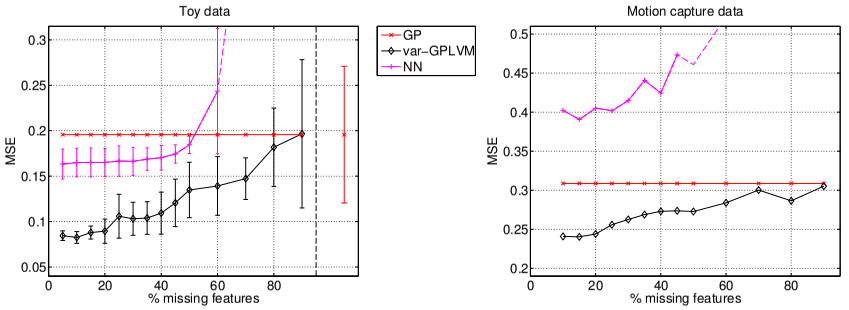


Figure 15: Mean squared error for predictions obtained by different methods in simulated (left) and motion capture data (right). The results for simulated data are obtained from 4 trials and, hence, errorbars are also plotted. For the GP, errorbars do not change with x-axis and, for clarity, they are plotted separately on the right of the dashed vertical line (for nonsensical x values). For clarity, the error for NN is not plotted when it grows too large; the full figure and comparison with other methods can be seen in Figure 20 of the Appendix.

7. Conclusion

We have introduced an approximation to the marginal likelihood of the Gaussian process latent variable model in the form of a variational lower bound. This provides a Bayesian

training procedure which is robust to overfitting and allows for the appropriate dimensionality of the latent space to be automatically determined. Our framework is extended for the case where the observed data constitute multivariate timeseries and, therefore, we obtain a very generic method for dynamical systems modelling able to capture complex, non-linear correlations. We demonstrated the advantages of the rigorous lower bound defined in our framework on a range of disparate real world data sets. This also emphasised the ability of the model to handle vast dimensionalities.

Our approach was easily extended to be applied to training Gaussian processes with uncertain inputs where these inputs have Gaussian prior densities. This further gave rise to two variants of our model: an auto-regressive GP as well as a GP regression model which can handle partially missing inputs. For future research, we envisage several other extensions that become computationally feasible using the same set of methodologies we espouse. In particular, propagation of uncertain inputs through the Gaussian process allows Bayes filtering (Ko and Fox, 2009a; Deisenroth et al., 2012; Frigola et al., 2014) applications to be carried out through variational bounds. Bayes filters are non-linear dynamical systems where time is discrete and the observed data $\mathbf{y}_t$ at time point t is non-linearly related to some unobserved latent state $\mathbf{x}_t$ via

$$\mathbf{y}_t = f(\mathbf{x}_t),$$

which itself has a non-linear autoregressive relationship with past latent states,

$$\mathbf{x}_t = g(\mathbf{x}_{t-1}),$$

where both $g(\cdot)$ and $f(\cdot)$ are assumed to be Gaussian processes. Propagation of the uncertainty through both processes can be achieved through our variational lower bound allowing fast efficient approximations to Gaussian process dynamical models.

The bound also allows for a promising new direction of research, that of *deep Gaussian processes*. In a deep Gaussian process (Lawrence and Moore, 2007; Damianou and Lawrence, 2013) the idea of placing a temporal prior over the inputs to a GP is further extended by hierarchical application. This formalism leads to a powerful class of models where Gaussian process priors are placed over function compositions (Damianou, 2015). For example, in a five layer model we have

$$f(\mathbf{X}) = g_5(g_4(g_3(g_2(g_1(\mathbf{X}))))),$$

where each $g_i(\cdot)$ is a draw from a Gaussian process. By combining such models with structure learning (Damianou et al., 2012) we can develop the potential to learn very complex non linear interactions between data. In contrast to other deep models *all* the uncertainty in parameters and latent variables is marginalised out.

Acknowledgments

This research was partially funded by the European research project EU FP7-ICT (Project Ref 612139 "WYSIWYD"), the Greek State Scholarships Foundation (IKY) and the University of Sheffield Moody endowment fund. We also thank Colin Litster and "Fit Fur Life" for allowing us to use their video files as datasets.

Appendix A. Further Details About the Variational Bound

This appendix contains supplementary details for deriving some mathematical formulae related to the calculation of the final expression of the variational lower bound for the training phase.

Since many derivations require completing the square to recognize a Gaussian, we will use the following notation throughout the Appendix:

$$\mathcal{Z} = \text{the collection of all constants for the specific line in equation,}$$

where the definition of a constant depends on the derivation at hand.

A.1 Calculation of: $\langle \log p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\rangle_{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})}$

First, we show in detail how to obtain the r.h.s of equation (20) for the following quantity: $\langle \log p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\rangle_{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})}$ which appears in the variational bound of equation (19). We now compute the above quantity analytically while temporarily using the notation $\langle\cdot\rangle = \langle\cdot\rangle_{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})}$:

$$\begin{aligned}
\langle \log p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\rangle &\overset{\text{eq. (6)}}{=} \left\langle \log \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{f}_{:,j}, \sigma^2\mathbf{I}_n\right)\right\rangle \\
&= -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\sigma^2\mathbf{I}_n| \\
&\quad -\frac{1}{2}\mathrm{tr}\left(\sigma^{-2}\mathbf{I}_n\left(\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\mathbf{y}_{:,j}\left\langle \mathbf{f}_{:,j}^\top\right\rangle + \left\langle \mathbf{f}_{:,j}\mathbf{f}_{:,j}^\top\right\rangle\right)\right) \\
&\overset{\text{eq. (13)}}{=} \mathcal{Z} - \frac{1}{2}\mathrm{tr}\left(\sigma^{-2}\mathbf{I}_n\left(\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\mathbf{y}_{:,j}\mathbf{a}_j^\top + \mathbf{a}_j\mathbf{a}_j^\top + \boldsymbol{\Sigma}_f\right)\right).
\end{aligned}$$

By completing the square we find:

$$\begin{aligned}
\langle \log p(\mathbf{y}_{:,j}|\mathbf{f}_{:,j})\rangle_{p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j},\mathbf{X})} &= \log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j, \sigma^2\mathbf{I}_n\right) - \frac{1}{2}\mathrm{tr}\left(\sigma^{-2}\boldsymbol{\Sigma}_f\right) \\
&\overset{\text{eq. (14)}}{=} \log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j, \sigma^2\mathbf{I}_n\right) - \frac{1}{2\sigma^2}\mathrm{tr}\left(\mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{uf}\right).
\end{aligned}$$

A.2 Calculating the Explicit Form of $q(\mathbf{u}_{:,j})$

From equation (22), we have:

$$\log q(\mathbf{u}_{:,j}) \propto \left\langle \log\mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j, \sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})} + \log p(\mathbf{u}_{:,j}). \tag{42}$$

All the involved distributions are Gaussian and, hence, we only need to compute the r.h.s of the above equation and complete the square in order to get the posterior Gaussian distribution for $q(\mathbf{u}_{:,j})$. The expectation appearing in the above equation is easily computed

as:

$$\begin{aligned}\left\langle \log \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})} =& \mathcal{Z} - \frac{1}{2\sigma^2}\mathrm{tr}\left(\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\mathbf{y}_{:,j}\left\langle \mathbf{a}_j^\top\right\rangle_{q(\mathbf{X})} + \left\langle \mathbf{a}_j\mathbf{a}_j^\top\right\rangle_{q(\mathbf{X})}\right)\\ \overset{\text{eq. (14)}}{=}& \mathcal{Z} - \frac{1}{2\sigma^2}\mathrm{tr}\Big(\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\mathbf{y}_{:,j}\mathbf{u}_{:,j}^\top\mathbf{K}_{uu}^{-1}\left\langle \mathbf{K}_{fu}^\top\right\rangle_{q(\mathbf{X})}\\ &+ \mathbf{u}_{:,j}^\top\mathbf{K}_{uu}^{-1}\left\langle \mathbf{K}_{fu}\mathbf{K}_{fu}\right\rangle_{q(\mathbf{X})}\mathbf{K}_{uu}^{-1}\mathbf{u}_{:,j}\Big)\\ \overset{\text{eq. (26)}}{=}& \mathcal{Z} - \frac{1}{2\sigma^2}\mathrm{tr}\Big(\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\mathbf{y}_{:,j}\mathbf{u}_{:,j}^\top\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_1^\top\\ &+ \mathbf{u}_{:,j}^\top\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_2\mathbf{K}_{uu}^{-1}\mathbf{u}_{:,j}\Big). \end{aligned} \tag{43}$$

We can now easily find equation (42) by combining equations (43) and (15):

$$\begin{aligned}\log q(\mathbf{u}_{:,j}) &\propto \left\langle \log \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})} + \log p(\mathbf{u}_{:,j})\\ &= \mathcal{Z} - \frac{1}{2\sigma^2}\mathrm{tr}\Big(\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\mathbf{y}_{:,j}\mathbf{u}_{:,j}^\top\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_1^\top + \mathbf{u}_{:,j}^\top\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_2\mathbf{K}_{uu}^{-1}\mathbf{u}_{:,j}\Big)\\ &\quad - \frac{1}{2}\mathrm{tr}\Big(\mathbf{K}_{uu}^{-1}\mathbf{u}_{:,j}\mathbf{u}_{:,j}^\top\Big)\\ &= \mathcal{Z} - \frac{1}{2}\mathrm{tr}\Big(\mathbf{u}_{:,j}^\top\left(\sigma^{-2}\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_2\mathbf{K}_{uu}^{-1} + \mathbf{K}_{uu}^{-1}\right)\mathbf{u}_{:,j} + \sigma^{-2}\mathbf{y}_{:,j}\mathbf{y}_{:,j}^\top - 2\sigma^{-2}\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_1^\top\mathbf{y}_{:,j}\mathbf{u}_{:,j}^\top\Big).\end{aligned} \tag{44}$$

We can now complete the square again and recognize that $q(\mathbf{u}_{:,j}) = \mathcal{N}\left(\mathbf{u}_{:,j}|\boldsymbol{\mu}_u,\mathbf{\Sigma}_u\right)$, where:

$$\begin{aligned}\mathbf{\Sigma}_u &= \left(\sigma^{-2}\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_2\mathbf{K}_{uu}^{-1} + \mathbf{K}_{uu}^{-1}\right)^{-1} \quad \text{and}\\ \boldsymbol{\mu}_u &= \sigma^{-2}\mathbf{\Sigma}_u\mathbf{K}_{uu}^{-1}\mathbf{\Psi}_1^\top\mathbf{y}_{:,j}.\end{aligned}$$

By "pulling" the $\mathbf{K}_{uu}$ matrices out of the inverse and after simple manipulations we get the final form of $q(\mathbf{u}_{:,j})$:

$$\begin{aligned}q(\mathbf{u}_{:,j}) &= \mathcal{N}\left(\mathbf{u}_{:,j}|\boldsymbol{\mu}_u,\mathbf{\Sigma}_u\right) \quad \text{where}\\ \boldsymbol{\mu}_u &= \mathbf{K}_{uu}\left(\sigma^2\mathbf{K}_{uu} + \mathbf{\Psi}_2\right)^{-1}\mathbf{\Psi}_1^\top\mathbf{y}_{:,j}\\ \mathbf{\Sigma}_u &= \sigma^2\mathbf{K}_{uu}\left(\sigma^2\mathbf{K}_{uu} + \mathbf{\Psi}_2\right)^{-1}\mathbf{K}_{uu}.\end{aligned} \tag{45}$$

A.3 Detailed Derivation of $\hat{\mathcal{F}}_j(q(\mathbf{X}))$

The quantity $\hat{\mathcal{F}}_j(q(\mathbf{X}))$ appears in equation (23). Based on the derivations of the previous section, we can rewrite equation (44) as a function of the optimal $q(\mathbf{u}_{:,j})$ found in equation (45) by completing the constant terms:

$$\left\langle \log \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_j,\sigma^2\mathbf{I}_n\right)\right\rangle_{q(\mathbf{X})} + \log p(\mathbf{u}_{:,j}) = \mathcal{B} + \log \mathcal{N}\left(\mathbf{u}_{:,j}|\boldsymbol{\mu}_u,\mathbf{\Sigma}_u\right) \tag{46}$$

where we have defined:

$$\mathcal{B} = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\sigma^2\mathbf{I}_n| - \frac{1}{2}\log|\mathbf{K}_{uu}| - \frac{1}{2\sigma^2}\mathbf{y}_{:,j}^\top\mathbf{y}_{:,j} + \frac{1}{2}\boldsymbol{\mu}_u^\top\mathbf{\Sigma}_u^{-1}\boldsymbol{\mu}_u + \frac{1}{2}\log|\mathbf{\Sigma}_u|. \tag{47}$$

We can now obtain the final expression for (23) by simply putting the quantity of (46) on the exponent and integrating. By doing so, we get:

$$\begin{aligned}&\int e^{\left\langle \log \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_d,\sigma^2 I_d\right)\right\rangle_{q(\mathbf{X})}}p(\mathbf{u}_{:,j})\mathrm{d}\mathbf{u}_{:,j} = \int e^{\mathcal{B}}e^{\log \mathcal{N}\left(\mathbf{u}_{:,j}|\boldsymbol{\mu}_u,\mathbf{\Sigma}_u\right)}\mathrm{d}\mathbf{u}_{:,j} = e^{\mathcal{B}}\\ \overset{\text{eq. (47)}}{=}& (2\pi)^{-\frac{N}{2}}\sigma^{-n}|\mathbf{K}_{uu}|^{-\frac{1}{2}}e^{-\frac{1}{2\sigma^2}\mathbf{y}_{:,j}^\top\mathbf{y}_{:,j}}|\mathbf{\Sigma}_u|^{\frac{1}{2}}e^{\frac{1}{2}\boldsymbol{\mu}_u^\top\mathbf{\Sigma}_u^{-1}\boldsymbol{\mu}_u}.\end{aligned} \tag{48}$$

By using equation (45) and some straightforward algebraic manipulations, we can replace in the above $\boldsymbol{\mu}_u^\top\mathbf{\Sigma}_u^{-1}\boldsymbol{\mu}_u$ with:

$$\boldsymbol{\mu}_u^\top\mathbf{\Sigma}_u^{-1}\boldsymbol{\mu}_u = \mathbf{y}_{:,j}^\top\underbrace{\sigma^{-4}\mathbf{\Psi}_1(\sigma^{-2}\mathbf{\Psi}_2 + \mathbf{K}_{uu})^{-1}\mathbf{\Psi}_1^\top}_{\mathbf{W}'}\mathbf{y}_{:,j}. \tag{49}$$

Finally, using equation (45) to replace $\mathbf{\Sigma}_u$ with its equal, as well as equation (49), we can write the integral of equation (48) as:

$$\int e^{\left\langle \log \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{a}_d,\sigma^2 I_d\right)\right\rangle_{q(\mathbf{X})}}p(\mathbf{u}_{:,j})\mathrm{d}\mathbf{u}_{:,j} = \frac{\sigma^{-n}|\mathbf{K}_{uu}|^{-\frac{1}{2}}|\mathbf{K}_{uu}|e^{-\frac{1}{2\sigma^2}\mathbf{y}_{:,j}^\top\mathbf{y}_{:,j}}}{(2\pi)^{N/2}|\sigma^{-2}\mathbf{\Psi}_2 + \mathbf{K}_{uu}|^{\frac{1}{2}}}e^{\frac{1}{2}\mathbf{y}_{:,j}^\top\mathbf{W}'\mathbf{y}_{:,j}}. \tag{50}$$

We can now obtain the final form for the variational bound by replacing equation (50) in equation (23), as well as replacing the term $\mathcal{A}$ with its equal and defining $\mathbf{W} = \sigma^{-2}\mathbf{I}_n - \mathbf{W}'$. By doing the above, we get exactly the final form of the bound of equation (25).

Appendix B. Calculating the Ψ Quantities

Here we explain how one can compute the Ψ quantities (introduced in Section 3.2) for two standard choices for the GP prior covariance. For completeness, we start by rewriting the equations (27), (28) and (29):

$$\psi_0 = \sum_{i=1}^{n}\psi_0^i, \;\; \text{with} \;\; \psi_0^i = \int k(\mathbf{x}_{i,:},\mathbf{x}_{i,:})\mathcal{N}(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:},\mathbf{S}_i)\mathrm{d}\mathbf{x}_{i,:}.$$

$$(\mathbf{\Psi}_1)_{i,k} = \int k\left(\mathbf{x}_{i,:},(\mathbf{x}_u)_{k,:}\right)\mathcal{N}(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:},\mathbf{S}_i)\mathrm{d}\mathbf{x}_{i,:}.$$

$$\mathbf{\Psi}_2 = \sum_{i=1}^{n}\mathbf{\Psi}_2^i \;\; \text{where} \;\; (\Psi_2^i)_{k,k'} = \int k(\mathbf{x}_{i,:},(\mathbf{x}_u)_{k,:})k((\mathbf{x}_u)_{k',:},\mathbf{x}_{i,:})\mathcal{N}(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:},\mathbf{S}_i)\mathrm{d}\mathbf{x}_{i,:}.$$

The above computations involve convolutions of the covariance function with a Gaussian density. For some standard kernels such the ARD exponentiated quadratic (RBF) covariance and the linear covariance function these statistics are obtained analytically. In particular for the ARD exponentiated quadratic kernel of equation (8) we have:

$$\begin{aligned}\psi_0 &= n\sigma_f^2\\ (\mathbf{\Psi}_1)_{i,k} &= \sigma_f^2\prod_{j=1}^{q}\frac{\exp\left(-\frac{1}{2}\frac{w_j(\mu_{i,j}-(x_u)_{k,j})^2}{w_jS_{i,j}+1}\right)}{(w_jS_{i,j}+1)^{\frac{1}{2}}}\\ (\Psi_2^i)_{k,k'} &= \sigma_f^4\prod_{j=1}^{q}\frac{\exp\left(-\frac{w_j((x_u)_{k,j}-(x_u)_{k',j})^2}{4} - \frac{w_j(\mu_{i,j}-\bar{x}_{:,j})^2}{2w_jS_{i,j}+1}\right)}{(2w_jS_{i,j}+1)^{\frac{1}{2}}},\end{aligned}$$

where $\bar{x}_{:,j} = \frac{((x_u)_{k,j}+(x_u)_{k',j})}{2}$. This gives us all the components we need to compute the variational lower bound for the ARD exponentiated quadratic kernel.

The linear ARD covariance function $k_{f(\text{lin})}(\mathbf{x}_{i,:}, \mathbf{x}_{k,:}) = \sigma^2_{\text{lin}}\mathbf{x}_{i,:}^\top \mathbf{C}\mathbf{x}_{k,:}$ depends on a diagonal matrix $\mathbf{C}$ containing the ARD weights. For this covariance function, the integrals required for the Ψ statistics are also tractable, such that

$$\begin{aligned}\psi_0^i &= \text{tr}\left(\mathbf{C}(\boldsymbol{\mu}_{i,:}\boldsymbol{\mu}_{i,:}^\top + \mathbf{S}_i)\right)\\ (\boldsymbol{\Psi}_1)_{i,k} &= \boldsymbol{\mu}_{i,:}^\top \mathbf{C}(\mathbf{x}_u)_{k,:}\\ (\boldsymbol{\Psi}_2^i)_{k,k'} &= (\mathbf{x}_u)_{k,:}^\top \mathbf{C}\left(\boldsymbol{\mu}_{i,:}\boldsymbol{\mu}_{i,:}^\top + \mathbf{S}_i\right)\mathbf{C}(\mathbf{x}_u)_{k',:}.\end{aligned}$$

Appendix C. Derivatives of the Variational Bound for the Dynamical Version

Before giving the expressions for the derivatives of the variational bound (11), it should be recalled that the variational parameters $\boldsymbol{\mu}_j$ and $\mathbf{S}_j$ (for all qs) have been reparametrised as

$$\mathbf{S}_j = \left(\mathbf{K}_x^{-1} + \text{diag}(\boldsymbol{\lambda}_j)\right)^{-1} \text{ and } \boldsymbol{\mu}_{:,j} = \mathbf{K}_x\bar{\boldsymbol{\mu}}_{:,j},$$

where the function diag$(\cdot)$ transforms a vector into a square diagonal matrix and vice versa. Given the above, the set of the parameters to be optimised is $(\boldsymbol{\theta}_f, \boldsymbol{\theta}_x, \{\bar{\boldsymbol{\mu}}_{:,j}, \boldsymbol{\lambda}_j\}_{j=1}^q, \tilde{\mathbf{X}})$. The gradient w.r.t the inducing points $\tilde{\mathbf{X}}$, however, has exactly the same form as for $\boldsymbol{\theta}_f$ and, therefore, is not presented here.

Some more notation:

1. λ_j is a scalar, an element of the vector $\boldsymbol{\lambda}_j$ which, in turn, is the main diagonal of the diagonal matrix $\boldsymbol{\Lambda}_j$.
2. $(S_j)_{k,l} \triangleq S_{j;kl}$ the element of $\mathbf{S}_j$ found in the k-th row and l-th column.
3. $\mathbf{s}_j \triangleq \{(S_j)_{i,i}\}_{i=1}^n$, i.e. it is a vector with the diagonal of $\mathbf{S}_j$.

C.1 Derivatives w.r.t the Variational Parameters

$$\frac{\partial \mathcal{F}}{\partial \bar{\boldsymbol{\mu}}_j} = \mathbf{K}_x\left(\frac{\partial \hat{\mathcal{F}}}{\partial \boldsymbol{\mu}_{:,j}} - \bar{\boldsymbol{\mu}}_{:,j}\right) \text{ and } \frac{\partial \mathcal{F}}{\partial \boldsymbol{\lambda}_j} = -(\mathbf{S}_j \circ \mathbf{S}_j)\left(\frac{\partial \hat{\mathcal{F}}}{\partial \mathbf{s}_j} + \frac{1}{2}\boldsymbol{\lambda}_j\right).$$

where for each single dimensional element we have:

$$\begin{aligned}\frac{\partial \hat{\mathcal{F}}}{\partial \mu_j} &= -\frac{p}{2\sigma^2}\frac{\partial \psi_0}{\partial \mu_j} + \sigma^{-2}\text{tr}\left(\frac{\partial \boldsymbol{\Psi}_1^\top}{\partial \mu_j}\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\\ &\quad + \frac{1}{2\sigma^2}\text{tr}\left(\frac{\partial \boldsymbol{\Psi}_2}{\partial \mu_j}\left(p\mathbf{K}_{uu}^{-1} - \sigma^2 p\mathbf{A}^{-1} - \mathbf{A}^{-1}\boldsymbol{\Psi}_1^\top\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\right)\\ \frac{\partial \hat{\mathcal{F}}}{\partial (S_j)_{k,l}} &= -\frac{p}{2\sigma^2}\frac{\partial \Psi_0}{\partial (S_j)_{k,l}} + \sigma^{-2}\text{tr}\left(\frac{\partial \boldsymbol{\Psi}_1^\top}{\partial (S_j)_{k,l}}\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\\ &\quad + \frac{1}{2\sigma^2}\text{tr}\left(\frac{\partial \boldsymbol{\Psi}_2}{\partial (S_j)_{k,l}}\left(p\mathbf{K}_{uu}^{-1} - \sigma^2 p\mathbf{A}^{-1} - \mathbf{A}^{-1}\boldsymbol{\Psi}_1^\top\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\right)\end{aligned}$$

with $\mathbf{A} = \sigma^2\mathbf{K}_{uu} + \boldsymbol{\Psi}_2$.

C.2 Derivatives w.r.t $\boldsymbol{\theta} = (\boldsymbol{\theta}_f, \boldsymbol{\theta}_x)$ and $\beta = \sigma^{-2}$

In our implementation, we prefer to parametrise the software with the data precision β, rather than the data variance, σ^2. Therefore, here we will give directly the derivatives for the precision. Obviously, through the use of the chain rule and the relationship $\sigma^2 = \beta^{-1}$ one can obtain the derivatives for the variance. Further, when it comes to model parameters, we will write the gradients with respect to each single element θ_f or θ_x.

Given that the KL term involves only the temporal prior, its gradient w.r.t the parameters $\boldsymbol{\theta}_f$ is zero. Therefore:

$$\frac{\partial \mathcal{F}}{\partial \theta_f} = \frac{\partial \hat{\mathcal{F}}}{\partial \theta_f}$$

with:

$$\begin{aligned}\frac{\partial \hat{\mathcal{F}}}{\partial \theta_f} &= \text{const} - \frac{\beta p}{2}\frac{\partial \psi_0}{\partial \theta_f} + \beta\text{tr}\left(\frac{\partial \boldsymbol{\Psi}_1^\top}{\partial \theta_f}\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\\ &\quad + \frac{1}{2}\text{tr}\left(\frac{\partial \mathbf{K}_{uu}}{\partial \theta_f}\left(p\mathbf{K}_{uu}^{-1} - \beta^{-1}p\mathbf{A}^{-1} - \mathbf{A}^{-1}\boldsymbol{\Psi}_1^\top\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1} - \beta p\mathbf{K}_{uu}^{-1}\boldsymbol{\Psi}_2\mathbf{K}_{uu}^{-1}\right)\right)\\ &\quad + \frac{\beta}{2}\text{tr}\left(\frac{\partial \boldsymbol{\Psi}_2}{\partial \theta_f}\quad\left(p\mathbf{K}_{uu}^{-1} - \beta^{-1}p\mathbf{A}^{-1} - \mathbf{A}^{-1}\boldsymbol{\Psi}_1^\top\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\right)\end{aligned}$$

The expression above is identical for the derivatives w.r.t the inducing points. For the gradients w.r.t the β term, we have a similar expression:

$$\begin{aligned}\frac{\partial \hat{\mathcal{F}}}{\partial \beta} &= \frac{1}{2}\Big[p\left(\text{tr}\left(\mathbf{K}_{uu}^{-1}\boldsymbol{\Psi}_2\right) + (n-m)\beta^{-1} - \psi_0\right) - \text{tr}\left(\mathbf{Y}\mathbf{Y}^\top\right) + \text{tr}\left(\mathbf{A}^{-1}\boldsymbol{\Psi}_1^\top\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\right)\\ &\quad + \beta^{-2}p\ \text{tr}\left(\mathbf{K}_{uu}\mathbf{A}^{-1}\right) + \beta^{-1}\text{tr}\left(\mathbf{K}_{uu}\mathbf{A}^{-1}\boldsymbol{\Psi}_1^\top\mathbf{Y}\mathbf{Y}^\top\boldsymbol{\Psi}_1\mathbf{A}^{-1}\right)\Big].\end{aligned}$$

In contrast to the above, the term $\hat{\mathcal{F}}$ does involve parameters $\boldsymbol{\theta}_x$, because it involves the variational parameters that are now reparametrised with $\mathbf{K}_x$, which in turn depends on $\boldsymbol{\theta}_x$. To demonstrate that, we will forget for a moment the reparametrisation of $\mathbf{S}_j$ and we will express the bound as $\mathcal{F}(\boldsymbol{\theta}_x, \mu_j(\boldsymbol{\theta}_x))$ (where $\mu_j(\boldsymbol{\theta}_x) = K_t\bar{\boldsymbol{\mu}}_{:,j}$) so as to show explicitly the dependency on the variational mean which is now a function of $\boldsymbol{\theta}_x$. Our calculations must now take into account the term $\left(\frac{\partial \hat{\mathcal{F}}(\boldsymbol{\mu}_{:,j})}{\partial \boldsymbol{\mu}_{:,j}}\right)^\top \frac{\partial \mu_j(\boldsymbol{\theta}_x)}{\partial \boldsymbol{\theta}_x}$ that is what we "miss" when we consider $\mu_j(\boldsymbol{\theta}_x) = \boldsymbol{\mu}_{:,j}$:

$$\begin{aligned}\frac{\partial \mathcal{F}(\boldsymbol{\theta}_x, \mu_j(\boldsymbol{\theta}_x))}{\partial \theta_x} &= \frac{\partial \mathcal{F}(\boldsymbol{\theta}_x, \boldsymbol{\mu}_{:,j})}{\partial \theta_x} + \left(\frac{\partial \hat{\mathcal{F}}(\boldsymbol{\mu}_{:,j})}{\partial \boldsymbol{\mu}_{:,j}}\right)^\top \frac{\partial \mu_j(\boldsymbol{\theta}_x)}{\partial \theta_x}\\ &= \cancel{\frac{\partial \hat{\mathcal{F}}(\boldsymbol{\mu}_{:,j})}{\partial \theta_x}} + \frac{\partial(-\text{KL})(\boldsymbol{\theta}_x, \mu_j(\boldsymbol{\theta}_x))}{\partial \theta_x} + \left(\frac{\partial \hat{\mathcal{F}}(\boldsymbol{\mu}_{:,j})}{\partial \boldsymbol{\mu}_{:,j}}\right)^\top \frac{\partial \mu_j(\boldsymbol{\theta}_x)}{\partial \theta_x}.\end{aligned}$$

We do the same for $\mathbf{S}_j$ and then we can take the resulting equations and replace $\boldsymbol{\mu}_j$ and $\mathbf{S}_j$ with their equals so as to take the final expression which only contains $\bar{\boldsymbol{\mu}}_{:,j}$ and $\boldsymbol{\lambda}_j$:

$$\frac{\partial \mathcal{F}(\boldsymbol{\theta}_x, \mu_j(\boldsymbol{\theta}_x), \mathbf{S}_j(\boldsymbol{\theta}_x))}{\partial \theta_x} = \text{tr}\Bigg[\Big[-\frac{1}{2}\left(\hat{\mathbf{B}}_j \mathbf{K}_x \hat{\mathbf{B}}_j + \bar{\boldsymbol{\mu}}_{:,j}\bar{\boldsymbol{\mu}}_{:,j}^\top\right) + \left(\mathbf{I} - \hat{\mathbf{B}}_j \mathbf{K}_x\right) \text{diag}\left(\frac{\partial \hat{\mathcal{F}}}{\partial \mathbf{s}_j}\right)\left(\mathbf{I} - \hat{\mathbf{B}}_j \mathbf{K}_x\right)^\top \Big] \frac{\partial \mathbf{K}_x}{\partial \theta_x}\Bigg] + \left(\frac{\partial \hat{\mathcal{F}}(\boldsymbol{\mu}_{:,j})}{\partial \boldsymbol{\mu}_{:,j}}\right)^\top \frac{\partial \mathbf{K}_x}{\partial \theta_x}\bar{\boldsymbol{\mu}}_{:,j}$$

where $\hat{\mathbf{B}}_j = \boldsymbol{\Lambda}_j^{\frac{1}{2}} \widetilde{\mathbf{B}}_j^{-1} \boldsymbol{\Lambda}_j^{\frac{1}{2}}$. and $\tilde{\mathbf{B}}_j = \mathbf{I} + \boldsymbol{\Lambda}_j^{\frac{1}{2}} \mathbf{K}_x \boldsymbol{\Lambda}_j^{\frac{1}{2}}$. Note that by using this $\tilde{\mathbf{B}}_j$ matrix (which has eigenvalues bounded below by one) we have an expression which, when implemented, leads to more numerically stable computations, as explained in Rasmussen and Williams (2006) page 45-46.

Appendix D. Variational Lower Bound for Partially Observed Test Data

This section provides some more details related to the task of doing predictions based on partially observed test data $\mathbf{Y}_*^u$. Specifically, section D.1 explains in more detail the form of the variational lower bound for the aforementioned prediction scenario and illustrates how this gives rise to certain computational differences for the standard and the dynamical GP-LVM. Section D.2 gives some more details for the mathematical formulae associated with the above prediction task.

D.1 The Variational Bound in the Test Phase and Computational Issues

As discussed in Section 4.1, when doing predictions based on partially observed outputs with the variational GP-LVM, one needs to construct a variational lower bound as for the training phase. However, this now needs to be associated with the full set of observations $(\mathbf{Y}, \mathbf{Y}_*^o)$. Specifically, we need to lower bound the marginal likelihood given in equation (38). To achieve this, we start from equation (38) and then separate $\mathbf{Y}$ into $(\mathbf{Y}^o, \mathbf{Y}^u)$ while factorising the terms according to the conditional independencies, i.e. :

$$\begin{aligned}\log p(\mathbf{Y}_*^o, \mathbf{Y}) &= \log \int p(\mathbf{Y}_*^o, \mathbf{Y}|\mathbf{X}_*, \mathbf{X}) p(\mathbf{X}_*, \mathbf{X}) \mathrm{d}\mathbf{X}_* \mathrm{d}\mathbf{X} \\ &= \log \int p(\mathbf{Y}^u|\mathbf{X}) p(\mathbf{Y}_*^o, \mathbf{Y}^o|\mathbf{X}_*, \mathbf{X}) p(\mathbf{X}_*, \mathbf{X}) \mathrm{d}\mathbf{X}_* \mathrm{d}\mathbf{X}.\end{aligned}$$

Exactly analogously to the training phase, the variational bound to the above quantity takes the form:

$$\log p(\mathbf{Y}_*^o, \mathbf{Y}) \geq \int q(\mathbf{X}_*, \mathbf{X}) \log \frac{p(\mathbf{Y}^u|\mathbf{X}) p(\mathbf{Y}_*^o, \mathbf{Y}^o|\mathbf{X}_*, \mathbf{X}) p(\mathbf{X}_*, \mathbf{X})}{q(\mathbf{X}_*, \mathbf{X})} \mathrm{d}\mathbf{X}_* \mathrm{d}\mathbf{X}. \tag{51}$$

For the standard variational GP-LVM, we can further expand the above equation by noticing that the distributions $q(\mathbf{X}, \mathbf{X}_*)$ and $p(\mathbf{X}, \mathbf{X}_*)$ are fully factorised as $q(\mathbf{X}, \mathbf{X}_*) = \prod_{i=1}^n q(\mathbf{x}_{i,:}) \prod_{i=1}^{n_*} q(\mathbf{x}_{i,*})$. Therefore, equation (51) can be written as:

$$\begin{aligned}\log p(\mathbf{Y}_*^o, \mathbf{Y}) \geq &\int q(\mathbf{X}) \log p(\mathbf{Y}^u|\mathbf{X}) \mathrm{d}\mathbf{X} + \int q(\mathbf{X}_*, \mathbf{X}) \log p(\mathbf{Y}_*^o, \mathbf{Y}^o|\mathbf{X}_*, \mathbf{X}) \mathrm{d}\mathbf{X}_* \mathrm{d}\mathbf{X} \\ &- \mathrm{KL}\left(q(\mathbf{X}) \,\|\, p(\mathbf{X})\right) - \mathrm{KL}\left(q(\mathbf{X}_*) \,\|\, p(\mathbf{X}_*)\right).\end{aligned} \tag{52}$$

Recalling equation (31), we see the first term above can be obtained as the sum $\sum_{j \in u} \hat{\mathcal{F}}_j \left(q(\mathbf{X})\right)$ where each of the involved terms is given by equation (25) and is already computed during the training phase and, therefore, can be held fixed during test time. Similarly, the third term of equation (52) is also held fixed during test time. As for the second and fourth term, they can be optimised exactly as the bound computed for the training phase with the difference that now the data are augmented with test observations and only the observed dimensions are accounted for.

In contrast, the dynamical version of our model requires the full set of latent variables $(\mathbf{X}, \mathbf{X}_*)$ to be fully coupled in the variational distribution $q(\mathbf{X}, \mathbf{X}_*)$, as they together form a timeseries. Consequently, the expansion of equation (52) cannot be applied here, meaning that in this case no precomputations can be used from the training phase. However, one could apply the approximation $q(\mathbf{X}, \mathbf{X}_*) = q(\mathbf{X})q(\mathbf{X}_*)$ to speed up the test phase. In this case, each set of latent variables is still correlated, but the two sets are not. However, this approximation was not used in our implementation as it is only expected to speed up the predictions phase if the training set is very big, which is not the case for our experiments.

D.2 Calculation of the Posterior $q(\mathbf{F}_*^u|\mathbf{X})$

Optimisation based on the variational bound constructed for the test phase with partially observed outputs, as explained in Section 4.1, gives rise to the posterior $q(\mathbf{F}_*^u, \mathbf{U}, \mathbf{X}_*)$, as exactly happens in the training phase. Therefore, according to equation (16) we can write:

$$q(\mathbf{F}_*^u, \mathbf{U}, \mathbf{X}_*) = \left(\prod_{j=1}^p p(\mathbf{f}_{*,j}^u|\mathbf{u}_{:,j}, \mathbf{X}_*) q(\mathbf{u}_{:,j})\right) q(\mathbf{X}_*).$$

The marginal $q(\mathbf{F}_*^u|\mathbf{X}_*)$ (of equation (39)) is then simply found as:

$$\prod_{j \in u} \int p(\mathbf{f}_{*,j}^u|\mathbf{u}_{:,j}, \mathbf{X}_*) q(\mathbf{u}_{:,j}) \mathrm{d}\mathbf{u}_{:,j}.$$

The integrals inside the product are easy to compute since both types of densities appearing there are Gaussian, according to equations (13) and (45). In fact, each factor takes the form of a projected process predictive distribution from sparse GPs (Csató and Opper, 2002; Seeger et al., 2003; Rasmussen and Williams, 2006).

We will show the analytic derivation for the general case where we do not distinguish between training or test variables and all dimensions are observed. In specific, we want to compute:

$$p(\mathbf{f}_{:,j}|\mathbf{X}) = \int p(\mathbf{f}_{:,j}|\mathbf{u}_{:,j}, \mathbf{X}) q(\mathbf{u}_{:,j}) \mathrm{d}\mathbf{u}_{:,j}.$$

For this calculation we simply use the following identity for Gaussians:

$$\int \mathcal{N}\left(\mathbf{f}_{:,j}|\mathbf{M}\mathbf{u}_{:,j} + \mathbf{m}, \boldsymbol{\Sigma}_f\right) \mathcal{N}\left(\mathbf{u}_{:,j}|\boldsymbol{\mu}_u, \boldsymbol{\Sigma}_u\right) \mathrm{d}\mathbf{u}_{:,j} = \mathcal{N}\left(\mathbf{f}_{:,j}|\mathbf{M}\boldsymbol{\mu}_u + \mathbf{m}, \boldsymbol{\Sigma}_f + \mathbf{M}\boldsymbol{\Sigma}_u\mathbf{M}^\top\right).$$

From equations (14) and (45) we recognise:

$$\mathbf{M} = \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\ ,\ \mathbf{m} = \mathbf{0} \qquad \boldsymbol{\mu}_u = \mathbf{K}_{uu}(\sigma^2\mathbf{K}_{uu} + \boldsymbol{\Psi}_2)^{-1}\boldsymbol{\Psi}_1^\top \mathbf{y}_{:,j}$$
$$\boldsymbol{\Sigma}_f = \mathbf{K}_{fu} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{uf} \quad \boldsymbol{\Sigma}_u = \sigma^2\mathbf{K}_{uu}(\sigma^2\mathbf{K}_{uu} + \boldsymbol{\Psi}_2)^{-1}\mathbf{K}_{uu}$$

from where we easily find:

$$p(\mathbf{f}_{:,j}|\mathbf{X}) = \mathcal{N}\left(\mathbf{f}_{:,j}|\mathbf{K}_{fu}\mathbf{B}, \mathbf{K}_{ff} - \mathbf{K}_{fu}\left(\mathbf{K}_{uu}^{-1} + \left(\mathbf{K}_{uu} + \sigma^{-2}\boldsymbol{\Psi}_2\right)^{-1}\mathbf{K}_{uf}\right)\right)$$

with $\mathbf{B} = \sigma^{-2}(\mathbf{K}_{uu} + \sigma^{-2}\boldsymbol{\Psi}_2)^{-1}\boldsymbol{\Psi}_1^\top \mathbf{y}_{:,j}$.

Appendix E. Algorithm for GP Regression with Missing Inputs

Consider a fully and a partially observed set of inputs, i.e. $\mathbf{Z} = (\mathbf{Z}^o, \mathbf{Z}^u)$, where o and u denote set of rows of $(\mathbf{Z}, \mathbf{Y})$ that contain fully and partially observed inputs respectively. The features missing in $\mathbf{Z}^u$ can be different in number / location for each individual point $\mathbf{z}^u_{i,:}$. We can train the model in all of these observations jointly, by replacing the inputs $\mathbf{Z}^o$ and $\mathbf{Z}^u$ with distributions $q(\mathbf{X}^o)$ and $q(\mathbf{X}^u)$ respectively, and using Algorithm 1. Since the posterior distribution is factorised, the algorithm constrains it to be close to a delta function in regions where we have observations, i.e. in areas corresponding to $\mathbf{Z}^o$ and in areas corresponding to non-missing locations of $\mathbf{Z}^u$. The rest of the posterior area's parameters (means and variances of Gaussian marginals) are initialised according to a prediction model $\mathcal{M}^o$ and are subsequently optimised (along with model parameters) in an augmented model $\mathcal{M}^{o,u}$. Notice that the initial model $\mathcal{M}^o$ is obtained by training a variational GP-LVM model with a posterior $q(\mathbf{X}^o)$ whose mean is fully constrained to match the observations $\mathbf{Z}^o$ with very small uncertainty and, thus, the model $\mathcal{M}^o$ behaves almost as a standard GP regression model.

Appendix F. Additional Results from the Experiments

In this section we present additional figures obtained from the experiments.

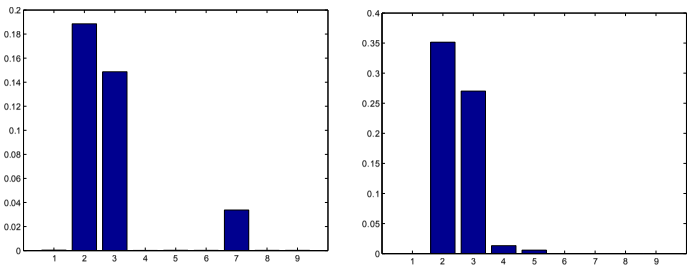
F.1 Motion Capture Data

We start by presenting additional results for the experiment described in Section 5.3 (motion capture data). Figure 16 depicts the optimised ARD weights (squared inverse lengthscales) for each of the dynamical models employed in the experiment. Figure 17 illustrates examples of the predictive performance of the models by plotting the true and predicted curves in the angle space.

As was explained in Section 5.3, all employed models encode the "walk" and "run" regime as two separate subspaces in the latent space. To illustrate this more clearly we sampled points from the learned latent space $\mathbf{X}$ of a trained dynamical variational GP-LVM model and generated the corresponding outputs, so as to investigate the kind of information that is encoded in each subspace of $\mathbf{X}$. Specifically, we considered the model that employed a Matérn $\frac{3}{2}$ covariance function to constrain the latent space and, based on the ARD weights of Figure 16(b), we projected the latent space on dimensions $(2, 3)$ and $(2, 4)$. Interacting with the model revealed that dimension 4 separates the "walk" from the "run" regime. This is an intuitive result, since the two kinds of motions are represented as separate clusters in the latent space. In other words, moving between two well separated

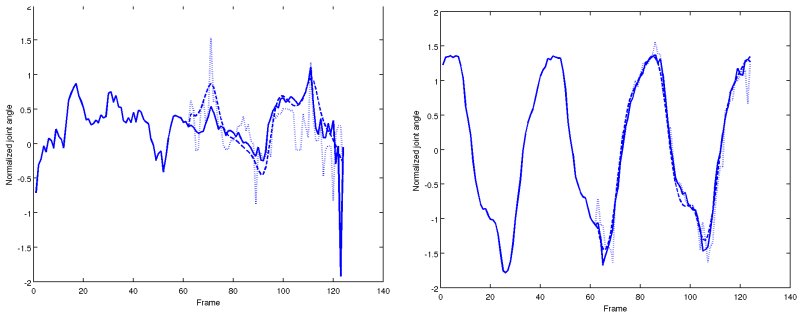
Algorithm 1 GP Regression with Missing Inputs Model: Training and predictions

Given: fully observed data $(\mathbf{Z}^o, \mathbf{Y}^o)$ and partially observed data $(\mathbf{Z}^u, \mathbf{Y}^u)$
Define a small value, e.g. $\varepsilon = 10^{-9}$
Initialize $q(\mathbf{X}^o) = \prod_{i=1}^n \mathcal{N}\left(\mathbf{x}^o_{i,:}|\mathbf{z}^o_{i,:}, \varepsilon\mathbf{I}\right)$
Fix $q(\mathbf{X}^o)$ in the optimiser *# (i.e. not be optimised)*
Train a variational GP-LVM model $\mathcal{M}^o$ given the above $q(\mathbf{X}^o)$ and $\mathbf{Y}^o$
for $i = 1, \cdots, |\mathbf{Y}^u|$ **do**
  Predict $p(\hat{\mathbf{x}}^u_{i,:}|\mathbf{y}^u_i, \mathcal{M}^o) \approx q(\hat{\mathbf{x}}^u_{i,:}) = \mathcal{N}\left(\hat{\mathbf{x}}^u_{i,:}|\hat{\boldsymbol{\mu}}^u_{i,:}, \hat{\mathbf{S}}^u_i\right)$
  Initialize $q(\mathbf{x}^u_{i,:}) = \mathcal{N}\left(\mathbf{x}^u_{i,:}|\boldsymbol{\mu}^u_{i,:}, \mathbf{S}^u_i\right)$ as follows:
  for $j = 1, \cdots, q$ **do**
    if $z^u_{i,j}$ is observed **then**
      $\mu^u_{i,j} = z^u_{i,j}$ and $(S^u_i)_{j,j} = \varepsilon$ *# $(S^u_i)_{j,j}$ denotes the j-th diagonal element of $\mathbf{S}^u_i$*
      Fix $\mu^u_{i,j}, (S^u_i)_{j,j}$ in the optimiser # *(i.e. will not be optimised)*
    else
      $\mu^u_{i,j} = \hat{\mu}^u_{i,j}$ and $(S^u_i)_{j,j} = (\hat{S}^u_i)_{j,j}$
Train a variational GP-LVM model $\mathcal{M}^{o,u}$ using the initial $q(\mathbf{X}^o)$ and $q(\mathbf{X}^u)$ defined above and data $\mathbf{Y}^o, \mathbf{Y}^u$ (the locations that were fixed for the variational distributions will not be optimised).
All subsequent predictions can be made using model $\mathcal{M}^{o,u}$.



(a) Model with exponentiated quadratic (b) Model with Matérn

Figure 16: The values of the weights (squared inverse lengthscales) of the ARD kernel after training on the motion capture dataset using the exponentiated quadratic (fig: (a)) and the Matérn (fig: (b)) kernel to model the dynamics for the dynamical variational GP-LVM. The weights that have zero value "switch off" the corresponding dimension of the latent space. The latent space is, therefore, 3-D for (a) and 4-D for (b). Note that the weights were initialised with very similar values.



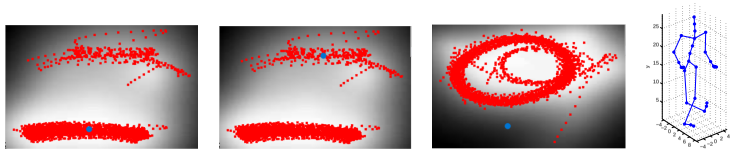
(a) (b)

Figure 17: The prediction for two of the test angles for the body (fig: 17(a)) and for the legs part (fig: 17(b)). Continuous line is the original test data, dotted line is nearest neighbour in scaled space, dashed line is dynamical variational GP-LVM (using the exponentiated quadratic kernel for the body reconstruction and the Matérn for the legs).

clusters needs to be encoded with a close to linear signal, which is in accordance with the small inverse lengthscale for dimension 4 (see also discussion in the caption of Figure 5). More specifically, to interact with the model we first fixed dimension 4 on a value belonging to the region encoding the walk, as can be seen in Figure 18(a), and then sampled multiple latent points by varying the other two dominant dimensions, namely 2 and 3, as can be seen in the top row of Figure 19. The corresponding outputs are shown in the second row of Figure 19. When dimension 4 was fixed on a value belonging to the region encoding the run (Figure 18(b)) the outputs obtained by varying dimensions 2 and 3 as before produced a smooth running motion, as can be seen in the third row of Figure 19. Finally, Figure 18(d) illustrates a motion which clearly is very different from the training set and was obtained by sampling a latent position far from the training data, as can be seen in Figure 18(c). This is indicative of a generative model's ability of producing novel data.

F.2 Gaussian Process Learning With Missing Inputs

In this section we present some more plots for the experiment presented in Section 6.2.1, where a set of fully observed outputs, $\mathbf{Y}$, corresponded to a set of fully observed inputs, $\mathbf{Z}^o$, and a set of inputs with randomly missing components, $\mathbf{Z}^u$. Even if $\mathbf{Y}$ is fully observed, it can be split according to the inputs, so that $\mathbf{Y} = (\mathbf{Y}^o, \mathbf{Y}^u)$. The variational GP-LVM, a nearest neighbour (NN) approach and multiple linear regression (MLR) (Chatterjee and Hadi, 1986) were trained on the full dataset $((\mathbf{Z}^o, \mathbf{Z}^u), (\mathbf{Y}^o, \mathbf{Y}^u))$. The standard GP model could only take into account the fully observed data, $(\mathbf{Z}^o, \mathbf{Y}^o)$. We also compared against predicting with the mean of $\mathbf{Y}$. The results are shown in Figure 20, which is an extension of the Figure 15 presented in the main paper.



(a) (b) (c) (d)

Figure 18: Plots (a) and (b) depict the projection of the latent space on dimensions 2 ($x-$axis) and 4 ($y-$axis), with the blue dot corresponding to the value on which these dimensions were fixed for the sampled latent points and red crosses represent latent points corresponding to training outputs. The intensity of the grayscale background represents the posterior uncertainty at each region (white corresponds to low predictive variance). Plot (c) depicts a latent space projection on dimensions 2 ($x-$axis) and 3 ($y-$axis), with the fixed latent positions corresponding to the generated output depicted in plot (d).

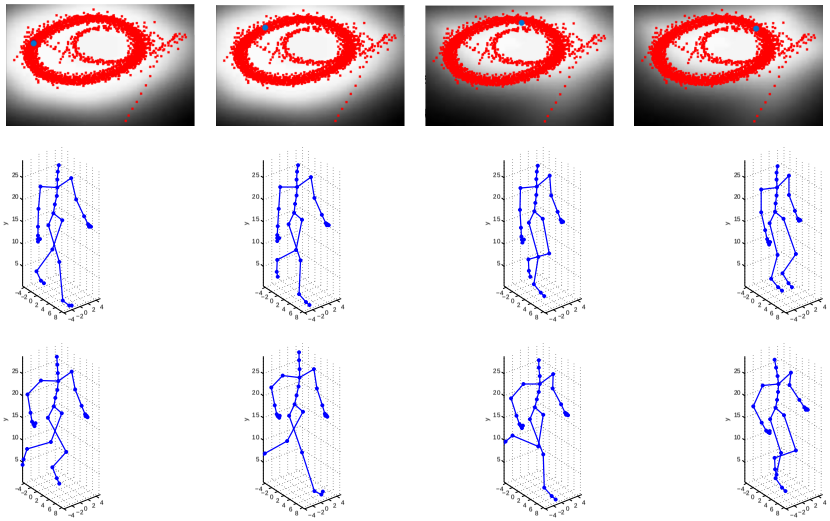


Figure 19: The first row depicts a projection of the latent space on dimensions 2 and 3 with the blue dot showing the value at which these dimensions were fixed for the sampled latent points. The corresponding outputs are depicted in the second row (for the walk regime) and third row (for the run regime).

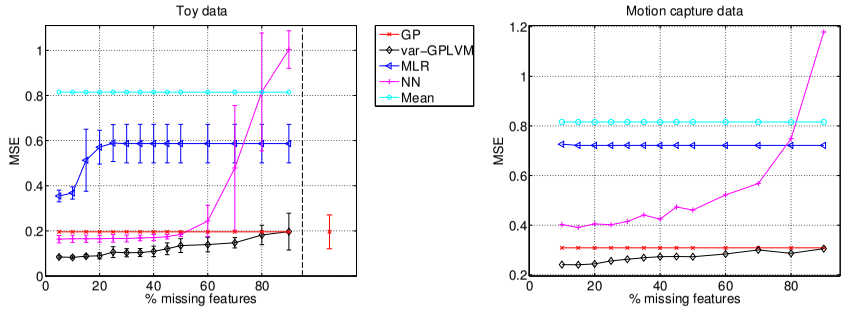


Figure 20: An augmented version of Figure 15. The above figure includes more results for the task of performing regression by learning from incomplete inputs. The results refer to the mean squared error for predictions obtained by different methods in simulated (left) and motion capture data (right). The results for simulated data are obtained from 4 trials and, hence, errorbars are also plotted. Lines without errorbars correspond to methods that cannot take into account partially observed inputs. For the GP, errorbars do not change with x-axis and, for clarity, they are plotted separately on the right of the dashed vertical line (for nonsensical x values).

References

The GPy authors. GPy: A Gaussian process framework in Python. 2014. URL `https://github.com/SheffieldML/GPy`.

David J. Bartholomew. *Latent Variable Models and Factor Analysis*. Charles Griffin & Co. Ltd, London, 1987.

Alexander Basilevsky. *Statistical Factor Analysis and Related Methods*. Wiley, New York, 1994.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003. doi: 10.1162/089976603321780317.

Christopher M. Bishop. Bayesian PCA. In Michael J. Kearns, Sara A. Solla, and David A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 482–388, Cambridge, MA, 1999. MIT Press.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006. ISBN 0387310738.

Christopher M. Bishop and Gwilym D. James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research*, A327:580–593, 1993. doi: 10.1016/0168-9002(93)90728-Z.

Christopher M. Bishop, Marcus Svensén, and Christopher K. I. Williams. GTM: the Generative Topographic Mapping. *Neural Computation*, 10(1):215–234, 1998. doi: 10.1162/089976698300017953.

Samprit Chatterjee and Ali S Hadi. Influential observations, high leverage points, and outliers in linear regression. *Statistical Science*, pages 379–393, 1986.

Lehel Csató. *Gaussian Processes — Iterative Sparse Approximations*. PhD thesis, Aston University, 2002.

Lehel Csató and Manfred Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

Zhenwen Dai, Andreas Damianou, James Hensman, and Neil Lawrence. Gaussian process models with parallelization and GPU acceleration. *arXiv preprint arXiv:1410.4984*, 2014.

Andreas Damianou. Deep Gaussian processes and variational propagation of uncertainty. *PhD Thesis, University of Sheffield*, 2015.

Andreas Damianou and Neil D. Lawrence. Deep Gaussian processes. In Carlos Carvalho and Pradeep Ravikumar, editors, *Proceedings of the Sixteenth International Workshop on Artificial Intelligence and Statistics*, volume 31, AZ, USA, 2013. JMLR W&CP 31.

Andreas Damianou, Michalis K. Titsias, and Neil D. Lawrence. Variational Gaussian process dynamical systems. In Peter Bartlett, Fernando Peirrera, Chris Williams, and John Lafferty, editors, *Advances in Neural Information Processing Systems*, volume 24, Cambridge, MA, 2011. MIT Press.

Andreas Damianou, Carl Henrik Ek, Michalis K. Titsias, and Neil D. Lawrence. Manifold relevance determination. In John Langford and Joelle Pineau, editors, *Proceedings of the International Conference in Machine Learning*, volume 29, San Francisco, CA, 2012. Morgan Kauffman.

Marc Peter Deisenroth, Ryan Darby Turner, Marco F Huber, Uwe D Hanebeck, and Carl Edward Rasmussen. Robust filtering and smoothing with Gaussian processes. *Automatic Control, IEEE Transactions on*, 57(7):1865–1871, 2012.

Carl Henrik Ek, Philip H.S. Torr, and Neil D. Lawrence. Gaussian process latent variable models for human pose estimation. In Andrei Popescu-Belis, Steve Renals, and Hervé Bourlard, editors, *Machine Learning for Multimodal Interaction (MLMI 2007)*, volume 4892 of *LNCS*, pages 132–143, Brno, Czech Republic, 2008. Springer-Verlag. doi: 10.1007/978-3-540-78155-4_12.

Brian D. Ferris, Dieter Fox, and Neil D. Lawrence. WiFi-SLAM using Gaussian process latent variable models. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2480–2485, 2007.

Roger Frigola, Fredrik Lindsten, Thomas B Schön, and Carl E Rasmussen. Identification of Gaussian process state-space models with particle stochastic approximation em. In *19th World Congress of the International Federation of Automatic Control (IFAC), Cape Town, South Africa*, 2014.

Nicoló Fusi, Christoph Lippert, Karsten Borgwardt, Neil D. Lawrence, and Oliver Stegle. Detecting regulatory gene-environment interactions with unmeasured environmental factors. *Bioinformatics*, 2013. doi: 10.1093/bioinformatics/btt148.

Yarin Gal, Mark van der Wilk, and Carl E. Rasmussen. Distributed variational inference in sparse Gaussian process regression and latent variable models. *arXiv:1402.1389*, 2014.

Zoubin Ghahramani, editor. *Proceedings of the International Conference in Machine Learning*, volume 24, 2007. Omnipress. ISBN 1-59593-793-3.

Agathe Girard, Carl Edward Rasmussen, Joaquin Quiñonero Candela, and Roderick Murray-Smith. Gaussian process priors with uncertain inputs—application to multiple-step ahead time series forecasting. In Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 529–536, Cambridge, MA, 2003. MIT Press.

Paul W. Goldberg, Christopher K. I. Williams, and Christopher M. Bishop. Regression with input-dependent noise: A Gaussian process treatment. In Jordan et al. (1998), pages 493–499.

Neil J. Gordon, David J. Salmond, and Adrian F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F Radar and Signal Processing*, 140(2), 1993.

James Hensman, Alex Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. *arXiv preprint arXiv:1411.2005*, 2014.

Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent Component Analysis*. John Wiley and Sons, 2001. ISBN 978-0-471-40540-5.

Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors. *Advances in Neural Information Processing Systems*, volume 10, Cambridge, MA, 1998. MIT Press.

Kristian Kersting, Christian Plagemann, Patrick Pfaff, and Wolfram Burgard. Most likely heteroscedastic Gaussian process regression. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 393–400, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273546.

Daniel Keysers, Roberto Paredes, Hermann Ney, and Enrique Vidal. Combination of tangent vectors and local representations for handwritten digit recognition. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 538–547. Springer, 2002.

Nathaniel J. King and Neil D. Lawrence. Fast variational inference for Gaussian Process models through KL-correction. In *ECML, Berlin, 2006*, Lecture Notes in Computer Science, pages 270–281, Berlin, 2006. Springer-Verlag.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. Technical report, 2013.

Diederik P. Kingma, Danilo Jimenez Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models. *CoRR*, abs/1406.5298, 2014.

Jonathan Ko and Dieter Fox. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Auton. Robots*, 27:75–90, July 2009a. ISSN 0929-5593. doi: 10.1007/s10514-009-9119-x. URL `http://portal.acm.org/citation.cfm?id=1569248.1569255`.

Jonathan Ko and Dieter Fox. Learning GP-BayesFilters via Gaussian process latent variable models. In *Robotics: Science and Systems*, 2009b.

Jonathan Ko and Dieter Fox. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots*, 27:75–90, July 2009c. ISSN 0929-5593. doi: 10.1007/s10514-009-9119-x.

Jonathan Ko and Dieter Fox. Learning GP-Bayesfilters via Gaussian process latent variable models. *Autonomous Robots*, 30:3–23, 2011. ISSN 0929-5593. URL `http://dx.doi.org/10.1007/s10514-010-9213-0`. 10.1007/s10514-010-9213-0.

Neil D. Lawrence. Gaussian process models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

Neil D. Lawrence. The Gaussian process latent variable model. Technical Report CS-06-03, The University of Sheffield, Department of Computer Science, 2006.

Neil D. Lawrence. Learning for larger datasets with the Gaussian process latent variable model. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Workshop on Artificial Intelligence and Statistics*, pages 243–250, San Juan, Puerto Rico, 21-24 March 2007. Omnipress.

Neil D. Lawrence. A unifying probabilistic perspective for spectral dimensionality reduction: Insights and new models. *Journal of Machine Learning Research*, 13, 2012. URL `http://jmlr.csail.mit.edu/papers/v13/lawrence12a.html`.

Neil D. Lawrence and Michael I. Jordan. Semi-supervised learning via Gaussian processes. In Lawrence Saul, Yair Weiss, and Léon Bouttou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 753–760, Cambridge, MA, 2005. MIT Press.

Neil D. Lawrence and Andrew J. Moore. Hierarchical Gaussian process latent variable models. In Ghahramani (2007), pages 481–488. ISBN 1-59593-793-3.

Miguel Lázaro-Gredilla. Bayesian warped Gaussian processes. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, Cambridge, MA, 2012.

Miguel Lázaro-Gredilla and Michalis K. Titsias. Variational heteroscedastic Gaussian process regression. In *In 28th International Conference on Machine Learning*, pages 841–848. ACM, 2011.

Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors. *Advances in Neural Information Processing Systems*, volume 13, Cambridge, MA, 2001. MIT Press.

Chaochao Lu and Xiaoou Tang. Surpassing human-level face verification performance on LFW with GaussianFace. *CoRR*, abs/1404.3840, 2014.

D. J. C. MacKay. Introduction to Gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, NATO ASI Series, pages 133–166. Kluwer Academic Press, 1998.

David J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, A*, 354(1):73–80, 1995. doi: 10.1016/0168-9002(94)00931-7.

Kantilal V. Mardia, John T. Kent, and John M. Bibby. *Multivariate analysis*. Academic Press, London, 1979. ISBN 0-12-471252-5.

Andrew McHutchon and Carl Edward Rasmussen. Gaussian process training with input noise. In *Advances in Neural Information Processing Systems*, pages 1341–1349, 2011.

Thomas P. Minka. Automatic choice of dimensionality for PCA. In Leen et al. (2001), pages 598–604.

Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

Jeremey Oakley and Anthony O'Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.

Manfred Opper and Cédric Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

Joaquin Quiñonero Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

Joaquin Quiñonero-Candela, Agathe Girard, Jan Larsen, and Carl Edward Rasmussen. Propagation of uncertainty in bayesian kernel models-application to multiple-step ahead forecasting. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 2, pages II–701. IEEE, 2003.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.

C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on*, volume 1, pages 29–36, Jan 2005. doi: 10.1109/ACVMOT.2005.107.

Sam T. Roweis. EM algorithms for PCA and SPCA. In Jordan et al. (1998), pages 626–632.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. doi: 10.1126/science.290.5500.2323.

John W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18(5):401–409, 1969. doi: 10.1109/T-C.1969.222678.

Simo Särkkä. *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013. ISBN 9781107619289.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998. doi: 10.1162/089976698300017467.

Matthias Seeger, Christopher K. I. Williams, and Neil D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In Christopher M. Bishop and Brendan J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, 3–6 Jan 2003.

Vikas Sindhwani, Wei Chu, and S Sathiya Keerthi. Semi-supervised Gaussian process classifiers. In *IJCAI*, pages 1059–1064, 2007.

Alexander J. Smola and Peter L. Bartlett. Sparse greedy Gaussian process regression. In Leen et al. (2001), pages 619–625.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Weiss et al. (2006).

Edward Snelson, Carl Edward Rasmussen, and Zoubin Ghahramani. Warped Gaussian processes. *Advances in Neural Information Processing Systems*, 16:337–344, 2004.

Graham W. Taylor, Geoffrey E. Hinton, and Sam Roweis. Modeling human motion using binary latent variables. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems*, volume 19, Cambridge, MA, 2007. MIT Press.

Joshua B. Tenenbaum, Virginia de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319.

Michael E. Tipping. The relevance vector machine. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 652–658, Cambridge, MA, 2000. MIT Press.

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3):611–622, 1999. doi: doi:10.1111/1467-9868.00196.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Workshop on Artificial Intelligence and Statistics*, volume 5, pages 567–574, Clearwater Beach, FL, 16-18 April 2009. JMLR W&CP 5.

Michalis K. Titsias and Neil D. Lawrence. Bayesian Gaussian process latent variable model. *Journal of Machine Learning Research - Proceedings Track*, 9:844–851, 2010.

Michalis K. Titsias and Miguel Lázaro-Gredilla. Variational inference for mahalanobis distance metrics in Gaussian process regression. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 279–287. Curran Associates, Inc., 2013.

G. E. Uhlenbeck and L. S. Ornstein. On the theory of the Brownian motion. *Phys. Rev.*, 36(5):823–841, Sep 1930. doi: 10.1103/PhysRev.36.823.

Raquel Urtasun and Trevor Darrell. Discriminative Gaussian process latent variable model for classification. In Ghahramani (2007). ISBN 1-59593-793-3.

Raquel Urtasun, David J. Fleet, and Pascal Fua. 3D people tracking with Gaussian process dynamical models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 238–245, New York, U.S.A., 17–22 Jun. 2006. IEEE Computer Society Press. doi: 10.1109/CVPR.2006.15.

Larens J. P. van der Maaten and Geoffrey E. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models. In Weiss et al. (2006).

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30 (2):283–298, 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1167.

Yair Weiss, Bernhard Schölkopf, and John C. Platt, editors. *Advances in Neural Information Processing Systems*, volume 18, Cambridge, MA, 2006. MIT Press.

Journal of Machine Learning Research 17 (2016) 1-28 Submitted 11/13; Revised 1/15; Published 4/16

On the Estimation of the Gradient Lines of a Density and the Consistency of the Mean-Shift Algorithm

Ery Arias-Castro EARIASCA@MATH.UCSD.EDU
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093, USA

David Mason DAVIDM@UDEL.EDU
Department of Applied Economics and Statistics
University of Delaware
Newark, DE 19717, USA

Bruno Pelletier BRUNO.PELLETIER@UNIV-RENNES2.FR
Département de Mathématiques
IRMAR – UMR CNRS 6625
Université Rennes II, France

Editor: Mikhail Belkin

Abstract

We consider the problem of estimating the gradient lines of a density, which can be used to cluster points sampled from that density, for example via the mean-shift algorithm of Fukunaga and Hostetler (1975). We prove general convergence bounds that we then specialize to kernel density estimation.

Keywords: mean-shift, gradient lines, density estimation, nonparametric clustering

1. Introduction

Fukunaga and Hostetler (1975) propose clustering points in space according to the gradient ascent flows of the underlying density. Let f be a differentiable density on $\mathbb{R}^d$. Assuming for now that f is known, consider the following scheme. Fix $a > 0$ and, starting at $x_0 \in \mathbb{R}^d$, iteratively define

$$x_\ell = x_{\ell-1} + a\frac{\nabla f(x_{\ell-1})}{f(x_{\ell-1})}, \quad \text{for } \ell \geq 1. \tag{1}$$

When it exists, define $x_\infty = \lim_{\ell\to\infty} x_\ell$. The rationale behind the iterative gradient ascent scheme (1) is to have the sequence $(x_\ell : t \geq 0)$ converge to a local mode of f — representing a cluster center, close in the spirit to Hartigan (1975) — without going through a valley. See Figure 1 and Figure 2 for simple illustrations involving the mixture of two Gaussians in dimensions $d = 1$ and $d = 2$. Now, a sample from f, say $X_1, \dots, X_n$, can be clustered by applying the iteration (1) to each X_i's, obtaining a sequence $(X_{i,\ell} : \ell \geq 0)$, and grouping according to the limit $X_{i,\infty}$, meaning that X_i and X_j are grouped together if $X_{i,\infty} = X_{j,\infty}$.

In the same spirit, Cheng et al. (2004) propose to use the gradient ascent lines of f, which form gradient trees, to perform a kind of hierarchical clustering of points on the plane. Clustering points according to the local maxima of the underlying density is also advocated

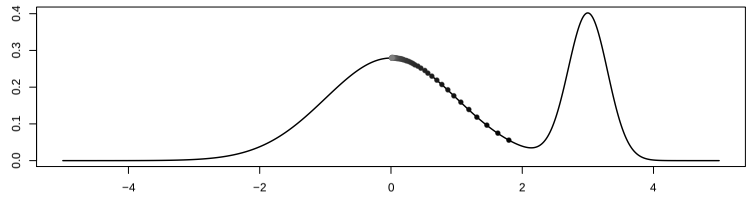


Figure 1: A mixture of two Gaussians in dimension $d = 1$: $f(x) = qg_{0,1}(x) + (1-q)g_{\mu,\sigma}(x)$ where $g_{\mu,\sigma}(x) := e^{-(x-\mu)^2/2\sigma^2}/\sqrt{2\pi}\sigma$, and $q = 0.7$, $\mu = 3$ and $\sigma = 0.3$. The starting point is at $x = 1.8$, and the 50 successive points in the iteration (1) are also plotted. Although the starting point is closer to the peak at $x = 3$, the sequence converges to the peak at $x = 0$.

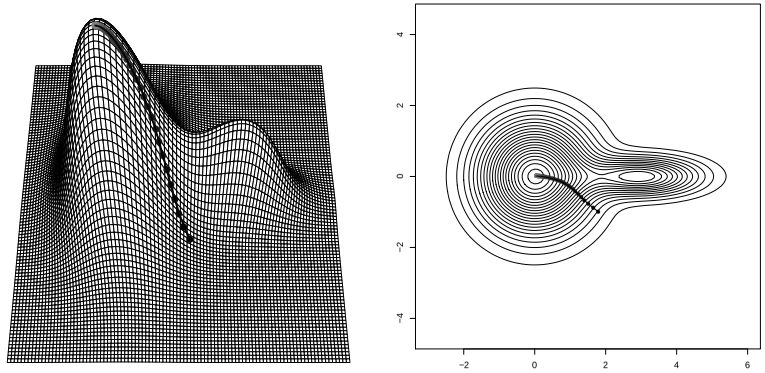


Figure 2: A mixture of two Gaussians in dimension $d = 2$: $f(x, y) = qg_{0,1}(x)g_{0,1}(y) + (1-q)g_{\mu_1,\sigma_1}(x)g_{0,\sigma_2}(y)$ with $q = 0.7$, $\mu_1 = 3$, $\sigma_1 = 1.5$ and $\sigma_2 = 0.5$. The starting point is at $(x, y) = (1.8, -1)$, and the 50 successive points in the iteration (1) are also plotted. Although the starting point is closer to the peak at $(x, y) = (3, 0)$, the sequence converges to the peak at $(x, y) = (0, 0)$.

by Comaniciu and Meer (2002), while an EM-type algorithm for finding the local maxima of the density f is suggested in Carreira-Perpinan and Williams (2003); Carreira-Perpinan (2007); Li et al. (2007).

In practice, the underlying density f is rarely known and has to be estimated. A kernel estimate is used in Fukunaga and Hostetler (1975); Cheng et al. (2004); Li et al.

(2007); Comaniciu and Meer (2002). Let $\Phi : \mathbb{R}^d \to \mathbb{R}$ be a kernel function — an integrable function with $\int_{\mathbb{R}^d} \Phi(x)\mathrm{d}x = 1$ — and for a bandwidth $h > 0$, let $\Phi_h(u) = h^{-d}\Phi(u/h)$. The corresponding kernel estimate for f based on a sample $X_1, \ldots, X_n$ is

$$f^{\phi}_{n,h}(x) := \frac{1}{n}\sum_{i=1}^{n} \Phi_h(x - X_i), \tag{2}$$

and if Φ is differentiable, then we may estimate the gradient of f by

$$\nabla f^{\phi}_{n,h}(x) := \frac{1}{nh}\sum_{i=1}^{n} \nabla\Phi_h(x - X_i).$$

Fukunaga and Hostetler (1975) introduce the term 'mean-shift' when describing the resulting estimate based on the Epanechnikov kernel $\Phi(u) \propto (1 - \|u\|^2)_+$, where $t_+ = \max(t, 0)$ is the positive part of $t \in \mathbb{R}$. Indeed, they show that, in that case,

$$\frac{\nabla f^{\phi}_{n,h}(x)}{f^{\phi}_{n,h}(x)} \propto \frac{1}{|I_{x,h}|}\sum_{i\in I_{x,h}} X_i - x, \quad I_{x,h} := \{i : \|X_i - x\| \le h\}.$$

Cheng (1995) further argues that the gradient ascent algorithm in (1) can be interpreted as a mean-shift when using a spherically symmetric kernel. Indeed, let Φ be a spherically symmetric kernel on $\mathbb{R}^d$, by which we mean a function $\Phi : \mathbb{R}^d \to \mathbb{R}$ of the form[1] $\Phi(u) = \phi(\|u\|)$, where $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ is a non-negative function, called the *profile function* in Cheng (1995), that satisfies the following *unit integral* condition

$$\int_{\mathbb{R}^d} \Phi(u)\mathrm{d}u = \omega_d \int_0^{\infty} \phi(r) r^{d-1}\mathrm{d}r = 1, \tag{3}$$

where ω_d is the surface area of the unit sphere of $\mathbb{R}^d$, and

$$\int_{\mathbb{R}^d} u_i u_j \Phi(u)\mathrm{d}u = \begin{cases} 1 & \text{if } i = j; \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The local average at x is

$$M_{n,h}(x) = \frac{\sum_{i=1}^n X_i \Phi_h(x - X_i)}{\sum_{i=1}^n \Phi_h(x - X_i)} = \frac{1}{n f^{\phi}_{n,h}(x)}\sum_{i=1}^{n} X_i \Phi_h(x - X_i).$$

The *mean shift* at x is defined by

$$T_{n,h}(x) = M_{n,h}(x) - x.$$

This is intimately related to the gradient of another kernel estimate of f. To see this, following Cheng (1995), we consider a *shadow kernel* Ψ of Φ, with profile function ψ defined by

$$\Psi(u) = \psi(\|u\|), \qquad \psi(r) = \int_r^{\infty} s\phi(s)\mathrm{d}s. \tag{5}$$

1. Note that Cheng (1995) uses a kernel of the form $\phi(\|u\|^2)$, so the presentation here is little different.

By construction and (3)-(4), Ψ integrates to 1, and is therefore a kernel function; it is also continuously differentiable. Let

$$f^{\psi}_{n,h}(x) = \frac{1}{n}\sum_{i=1}^{n} \Psi_h(x - X_i),$$

which is the kernel estimate of f with kernel Ψ and bandwidth h.

Lemma 1 *(Cheng, 1995) At any point x of $\mathbb{R}^d$, we have*

$$T_{n,h}(x) = h^2 \frac{\nabla f^{\psi}_{n,h}(x)}{f^{\phi}_{n,h}(x)}.$$

Assume that $\nabla\Psi$ is bounded in $\mathbb{R}^d$. Then by the Law of Large Numbers, for each fixed $x \in \mathbb{R}^d$, $f^{\phi}_{n,h}(x) \to f^{\phi}_h(x)$ and $\nabla f^{\psi}_{n,h}(x) \to \nabla f^{\psi}_h(x)$, almost surely as $n \to \infty$, where

$$f^{\phi}_h(x) = \int f(y)\Phi_h(x - y)\mathrm{d}y,$$

and f^{ψ}_h is defined similarly. Furthermore, if f is bounded and continuously differentiable on $\mathbb{R}^d$ with bounded gradient, then $f^{\phi}_h(x) \to f(x)$ and $\nabla f^{\psi}_h(x) \to \nabla f(x)$ as $h \to 0$. Hence, for any x fixed such that $f(x) > 0$,

$$T_{n,h}(x) \to T_h(x) \sim h^2 \nabla \log f(x),$$

as $n \to \infty$ first, followed by $h \to 0$. Following this line of thought, the mean-shift algorithm appears to approximate the gradient ascent scheme (1), with $a = h^2$. The convergence results in Cheng (1995) and Comaniciu and Meer (2002) provide only a very partial mathematical backing to this intuition.

Our contribution is a mathematical proof of consistency for the estimation of gradient ascent lines by the original mean-shift algorithm of Fukunaga and Hostetler (1975). We note that the same approach also applies to the more general mean-shift algorithm of Cheng (1995), and applies directly to the algorithm suggested by Cheng et al. (2004). In detail, let $f : \mathbb{R}^d \to \mathbb{R}$ be differentiable. Starting at $x_0 \in \mathbb{R}^d$, we study the convergence as $a \to 0$ of the sequence

$$x_\ell = x_{\ell-1} + a\nabla f(x_{\ell-1}), \quad \text{for } \ell \ge 1, \tag{6}$$

towards the gradient ascent line of f starting at x_0. In particular, we characterize the limit x_∞, providing a consistency result for the clustering algorithm based on the local maxima of f. Note that (6) includes (1) by replacing f with $\log f$. We note that such convergence results are available in the rich literature on dynamic systems — see, e.g., Stetter (1973, Sec 3.5), Beyn (1987) and Merlet and Pierre (2010, Sec 2) — and in the literature on convex optimization (where f is convex) — see, e.g., Boyd and Vandenberghe (2004, Sec. 9.3) and Bolte et al. (2010). However, for the general case, we could not find a specific rate of convergence as the one we obtain in (14). Although higher-order discretization schemes can be designed (Stetter, 1973), we focus entirely on the first-order scheme (6). We further elaborate on the literature after stating our main results in Section 2.

Then, given another differentiable function $\hat{f}$, meant to approximate f, we compare the sequence $(\hat{x}_\ell)$ to (x_ℓ), where

$$\hat{x}_\ell = \hat{x}_{\ell-1} + a\nabla\hat{f}(\hat{x}_{\ell-1}), \quad \text{for } \ell \geq 1, \tag{7}$$

starting at the same point $\hat{x}_0 = x_0$. In particular, when estimating the gradient ascent lines of a density f based on a sample $X_1, \ldots, X_n$, $\hat{f}$ can be taken to be some estimate of f, and the gradient ascent sequence defined by $\hat{f} = \log \hat{f}$ (starting at some x_0) is compared to that of $f = \log f$. Such approximation results are often called perturbation or stability results in the literature on dynamical systems. See, for example, Hirsch and Smale (1974, Chap 6) or Teschl (2012, Sec 2.5). Most of these results are qualitative (e.g., pertaining to the topology of the gradient flow lines), while the bound we obtain in (15) is quantitative.

Finally, we provide an explicit convergence rate for the case where the density is estimated by kernel convolution. This seems to be new in the literature on the mean-shift algorithm and, more generally, on the estimation of the gradient lines of a density.

The rest of the paper is organized as follows. In Section 2, we establish our main results, one on the convergence of the gradient ascent scheme (6), and another on the stability of smooth flows, relating the gradient flows of f and $\hat{f}$ when these functions are close as C^2 functions. In Section 3, we deduce convergence rates for the algorithm of Fukunaga and Hostetler (1975) defined in (1). The technical arguments are given in Section 4.

2. Main Results

Before stating our main results, we introduce some notations. For a function $f : \mathbb{R}^d \to \mathbb{R}$, we let $f^{(\ell)}(x)$ denote the differential form of f of order ℓ at a point $x \in \mathbb{R}^d$, and let $H_f(x)$ denote the Hessian matrix of f, when they exist. The differential form $f^{(\ell)}(x)$ of f at x is the multilinear map from $\mathbb{R}^d \times \cdots \times \mathbb{R}^d$ (ℓ times) to $\mathbb{R}$ defined by

$$f^{(\ell)}(x)[u_1, \ldots, u_\ell] = \sum_{i_1,\ldots,i_\ell=1}^{d} \frac{\partial^\ell f(x)}{\partial x_{i_1} \ldots \partial x_{i_\ell}} u_{1,i_1} \ldots u_{\ell,i_\ell},$$

where, for each $1 \leq i \leq \ell$, u_i has components $u_i = (u_{i,1}, \ldots, u_{i,d})$. Given a multilinear map L of order ℓ from $\mathbb{R}^d \times \cdots \times \mathbb{R}^d$ to $\mathbb{R}$, we denote by $\|L\|$ its operator norm defined by

$$\|L\| = \sup\left\{|L[u_1, \ldots, u_\ell]| : \|u_1\| = \cdots = \|u_\ell\| = 1\right\}, \tag{8}$$

and writing L as

$$L[u_1, \ldots, u_\ell] = \sum_{i_1,\ldots,i_\ell=1}^{d} L_{i_1,\ldots,i_\ell} u_{1,i_1} \ldots u_{\ell,i_\ell},$$

we denote by $\|L\|_{\max}$ the norm defined by

$$\|L\|_{\max} = \max\{|L_{i_1\ldots i_\ell}| : 1 \leq i_1, \ldots, i_\ell \leq d\}. \tag{9}$$

We note for future reference that

$$\|L\|_{\max} \leq \|L\| \leq d^{\frac{\ell}{2}} \|L\|_{\max}. \tag{10}$$

For a set $S \subset \mathbb{R}^d$, we also define

$$\kappa_\ell(f, S) = \sup_{x \in S} \|f^{(\ell)}(x)\|. \tag{11}$$

Note that $\kappa_\ell(f, S)$ is well-defined and is finite when f is of class C^ℓ and S is compact. The *upper level set* of a function $f : \mathbb{R}^d \to \mathbb{R}$ at $b \in \mathbb{R}$ is defined as

$$\mathcal{L}_f(b) = \{x \in \mathbb{R}^d : f(x) \geq b\}. \tag{12}$$

We suppress the dependence on f whenever no confusion is possible.

Recall that a *critical point* of f is a point x at which the gradient of f vanishes, that is, such that $\nabla f(x) = 0$. A *flow line* or *integral curve* of the positive gradient flow of f is a curve x such that $x'(t) = \nabla f(x(t))$. Note that, along any flow line, the value of f increases, that is, the function $t \mapsto f(x(t))$ is increasing with t. By the theory of ordinary differential equations, through any point $x_0 \in \mathbb{R}^d$ passes a unique flow line $x(t)$ defined for $t \in [0, t_0)$, where $t_0 > 0$, such that $x(0) = x_0$ (Hirsch et al., 2004, Section 7.2); we say that $x(t)$ is the flow line starting at x_0. Let $x^\star$ be a critical point of f. We say that x_0 is in the attraction basin of $x^\star$ if the flow line $x(t)$ starting at x_0 is defined for all $t \geq 0$ and $\lim_{t\to\infty} x(t) = x^\star$. An accumulation point of a sequence of points through an integral curve x, i.e., a sequence of the form $\{x(t_n) : t_1 < t_2 < \ldots\}$, is called a limit point of x. Any limit point of a gradient flow line of f is necessarily a critical point of f; see Hirsch et al. (2004, Section 9.3, Proposition, p. 206) and Hirsch et al. (2004, Section 9.3, Theorem, p. 205).

We start by establishing the convergence of the gradient ascent scheme (6) towards the flow lines of the underlying function f. Starting from a point x_0 in the attraction basin of the location of a stable local maximum $x^\star$, under some conditions stated below, the iteration (6) converges to $x^\star$. In fact, the polygonal line defined by the sequence (x_ℓ) is uniformly close to the flow line starting at x_0 and ending at $x^\star$. For the definition of a stable equilibrium of a dynamical system, we refer to Hirsch et al. (2004, Section 8.4).

Theorem 1 (Convergence of gradient ascent) *Let f be a function of class C^3. Let $(x(t) : t \geq 0)$ denote the flow line of f starting at x_0 and ending at a local maxima $x^\star$ of f. Let (x_ℓ) be the sequence defined in (6) starting at x_0. Then there exists $A = A(x_0, f) > 0$ such that, whenever $0 < a < A$,*

$$\lim_{\ell\to+\infty} x_\ell = x^\star. \tag{13}$$

Denote by $x_a(t)$ the following polygonal line

$$x_a(t) = x_{\ell-1} + (t/a - \ell + 1)(x_\ell - x_{\ell-1}), \quad \forall t \in [(\ell-1)a, \ell a).$$

Assume $H_f(x^\star)$ has all eigenvalues in $(-\overline{\nu}, -\underline{\nu})$ for some $0 < \underline{\nu} < \overline{\nu}$. Then, there exists a $C = C(x_0, f, \underline{\nu}, \overline{\nu}) > 0$ such that, for any $0 < a < A$,

$$\sup_{t\geq 0} \|x_a(t) - x(t)\| \leq Ca^\delta, \quad \delta := \frac{\underline{\nu}}{\underline{\nu} + \overline{\nu}}. \tag{14}$$

We mention the convergence result (Comaniciu and Meer, 2002, Th 1), which essentially says that, when f is a kernel density estimator with bandwidth h as in (2), the sequence (x_ℓ)

in (6) with choice $a = h^2$ converges and $(f(x_\ell))$ is monotone nondecreasing. In the literature on dynamical systems, the convergence result (13) is proved in (Merlet and Pierre, 2010, Sec 2), together with convergence rates, but under slightly different conditions; in particular, f is assumed to have compact upper level sets. Beyn (1987) compares the discrete and continuous trajectories under milder conditions, but only at a discrete grid of time points, and does so assuming that the starting point x_0 is sufficiently close to the corresponding stationary point $x^\star$. Moreover, the starting point of the discrete and continuous trajectories in Beyn (1987) are potentially different. In fact, Beyn (1987) refers the reader to (Stetter, 1973) — which we mentioned earlier — for the case where the starting points may be taken to be the same.

Next, we establish a stability result for flows of smooth functions. In words, under some conditions made precise below, when f and $\hat{f}$ are close as C^2 functions, then their flow lines are also close. Denote by $B(x, r)$ the open ball of radius r centered at x and by $\bar{B}(x, r)$ its closure.

Theorem 2 (Stability of smooth flows) *Suppose f and $\hat{f}$ are of class C^3. Let $(x(t) : t \geq 0)$ be a flow line of f starting at x_0 and ending at $x^\star$ where $H_f(x^\star)$ has all eigenvalues in $(-\overline{\nu}, -\underline{\nu})$ for some $0 < \underline{\nu} < \overline{\nu}$. Let $\hat{x}(t)$ be the flow line of $\hat{f}$ starting at x_0. Let $S = \mathcal{L}(f(x_0)/2) \cap \bar{B}(x_0, 3r_0)$ where $r_0 = \max_t \|x(t) - x_0\|$, and define*

$$\eta_m = \sup_{x \in S} \|f^{(m)}(x) - \hat{f}^{(m)}(x)\|.$$

Then there is a constant $C = C(f, x_0, \underline{\nu}, \overline{\nu}) \geq 1$ such that, when $\max(\eta_0, \eta_1, \eta_2) \leq 1/C$ and $\eta_3 \leq C$, $\hat{x}(t)$ is defined for all $t \geq 0$ and

$$\sup_{t \geq 0} \|x(t) - \hat{x}(t)\| \leq C \max\left\{\sqrt{\eta_0}, \eta_1^\delta\right\}, \tag{15}$$

where δ is defined in (14).

Stability results tend to be qualitative in the literature on dynamical systems. However, to establish the bound above, we do use a well-known quantitative result. See Lemma 7, which we took from Hirsch et al. (2004, Sec 17.5).

Combining Theorems 1 and 2, we arrive at the following bound for approximating the flow lines of a function f by the polygonal line obtained from the gradient ascent algorithm (7) based on an approximation $\hat{f}$ to f.

Corollary 1 *In the context of Theorem 2, for $a > 0$, define*

$$\hat{x}_a(t) = \hat{x}_{\ell-1} + (t/a - \ell + 1)(\hat{x}_\ell - \hat{x}_{\ell-1}), \quad \forall t \in [(\ell-1)a, \ell a), \tag{16}$$

where $(\hat{x}_\ell)$ is defined in (7). *Then there is a constant $C = C(f, x_0, \underline{\nu}, \overline{\nu}) \geq 1$ such that, when $\max(\eta_0, \eta_1, \eta_2) \leq 1/C$ and $\eta_3 \leq C$,*

$$\sup_{t \geq 0} \|\hat{x}_a(t) - x(t)\| \leq C \left[a^\delta + \max\left\{\sqrt{\eta_0}, \eta_1^\delta\right\}\right], \tag{17}$$

where δ is defined in (14).

Note that the exponent δ which appears in these results only depends on the ratio $\overline{\nu}/\underline{\nu}$ which is a lower bound on the condition number of $H_f(x^\star)$. But the constants in Theorems 1 and 2 depend on $\underline{\nu}$ and $\overline{\nu}$ not only through their ratio.

We note that Beyn (1987) establishes a result similar to Corollary 1 under milder assumptions. Indeed, just as we do here, he studies how the discrete system (7) approximates the continuous system

$$x'(t) = \nabla f(x(t)),$$

when the functions $\hat{f}$ and f may differ. He bounds the difference between the discrete and continuous trajectories, with possibly different starting points, over a discrete grid of time points, assuming the starting point x_0 is close enough to $x^\star$. He also assumes that $\nabla \hat{f}(x^\star) = 0$, which simplifies the analysis a fair amount. With these working assumptions, his bound is in $\kappa_2 a + \eta_1$ — see Equation (3.5) there. His method of proof is based on the theory of stable (solution) manifolds (Irwin, 1980, Chap 4). Our approach is more elementary and we do not know whether this more sophisticated approach has the potential to improve on ours.

We emphasize that Theorems 1 and 2, and their combined fruit in Corollary 1, are designed to establish our result on the uniform consistency of gradient line estimators based on kernel density estimators as stated in Theorem 3 in the next section.

3. The Estimation of Gradient Lines of a Density

Let $\hat{f}_{n,h}$ be the kernel density estimate of f in (2) with kernel Φ and bandwidth h. Sharp almost-sure convergence rates in the uniform norm of kernel density estimates have been obtained by several authors, for example Einmahl and Mason (2000); Giné and Guillou (2002); Einmahl and Mason (2005). Using the recent results of Mason and Swanepoel (2011) and Mason (2012), we derive strong uniform norm convergence rates for $\hat{f}_{n,k}$ and its derivatives.

We first control the bias component.

Lemma 2 *Assume Φ is nonnegative, C^3 on $\mathbb{R}^d$ with all partial derivatives up to order 3 vanishing at infinity, and satisfies*

$$\int_{\mathbb{R}^d} \Phi(x)\mathrm{d}x = 1, \quad \int_{\mathbb{R}^d} x\Phi(x)\mathrm{d}x = 0 \quad \text{and} \quad \int_{\mathbb{R}^d} \|x\|^2 \Phi(x)\mathrm{d}x < \infty. \tag{18}$$

Then for any C^3 density f on $\mathbb{R}^d$ with bounded derivatives up to order 3, there is a constant $C > 0$ such that

$$\sup_{x \in \mathbb{R}^d} \left\| \mathbb{E}\big[\hat{f}_{n,h}^{(\ell)}(x)\big] - f^{(\ell)}(x) \right\| \leq C h^{(3-\ell)\wedge 2}, \quad \forall 0 \leq \ell \leq 3. \tag{19}$$

Next, we control the variance component. For this, we apply the main result of Mason and Swanepoel (2011). See also Theorem 4.1 with Remark 4.2 in Mason (2012).

Lemma 3 *Suppose that Φ is of the form $\Phi : (x_1, \ldots, x_d) \mapsto \prod_{k=1}^d \phi_k(x_k)$, and that each ϕ_k is nonnegative, integrates to 1, and is C^3 on $\mathbb{R}$ with derivatives up to order 3 being of*

bounded variation and in $L_1(\mathbb{R}^d)$. *Then, for any bounded density* f *on* $\mathbb{R}^d$, *there exists a* $0 < b_0 < 1$ *such that*

$$\limsup_{n\to\infty} \sup_{\frac{\log n}{n} \le h^d \le b_0} \sup_{x\in\mathbb{R}^d} \sqrt{\frac{nh^{d+2\ell}}{\log n}} \left\| \hat{f}_{n,h}^{(\ell)}(x) - \mathbb{E}\big[\hat{f}_{n,h}^{(\ell)}(x)\big] \right\| < \infty, \quad \forall 0 \le \ell \le 3, \quad a.s. \quad (20)$$

It is straightforward to design a kernel that satisfies the conditions of Lemmas 2 and 3. In fact, the Gaussian kernel $\Phi(x) = (2\pi)^{-d/2} \exp(-\|x\|^2/2)$ is such a kernel.

Assuming that (20) holds and applying Corollary 1, we deduce a convergence result for the mean-shift algorithm of Fukunaga and Hostetler (1975). We note that a similar result holds for the simpler gradient ascent method of Cheng et al. (2004).

Theorem 3 *Consider a density* f *satisfying the conditions of Lemma 2. Suppose* $\hat{f}_{n,h}$ *is a kernel estimator of* f *of the form* (2), *where* Φ *satisfies the conditions of Lemmas 2 and 3. Let* $(x(t) : t \ge 0)$ *be the flow line of* f *starting at a point* x_0 *with* $f(x_0) > 0$, *ending at a point* $x^\star$ *where* $H_f(x^\star)$ *has all eigenvalues in* $(-\overline{\nu}, -\underline{\nu})$ *for some* $0 < \underline{\nu} < \overline{\nu}$. *For* $a > 0$, *define* $(\hat{x}_a(t) : t \ge 0)$ *by*

$$\hat{x}_a(t) = \hat{x}_{\ell-1} + (t/a - \ell + 1)(\hat{x}_\ell - \hat{x}_{\ell-1}), \quad \forall t \in [(\ell-1)a, \ell a),$$

where

$$\hat{x}_\ell = \hat{x}_{\ell-1} + a\frac{\nabla \hat{f}_{n,h}(\hat{x}_{\ell-1})}{\hat{f}_{n,h}(\hat{x}_{\ell-1})}, \quad for\ \ell \ge 1.$$

Suppose that $h \to 0$ *and* $\frac{nh^{d+6}}{\log n} \to \infty$. *Then there exists a constant* $C > 0$ *such that, with probability one, for all* n *large enough,*

$$\sup_{t\ge 0} \|\hat{x}_a(t) - x(t)\| \le C\left[a + h^2\right]^\delta, \quad \delta := \frac{\underline{\nu}}{\underline{\nu} + \overline{\nu}}. \quad (21)$$

The approximation error decreases as the discretization step a gets smaller, simply because it controls the precision of the (discrete) gradient ascent scheme (7). We made this precise in Theorem 1. However, as a gets smaller, the computational burden of running this gradient ascent scheme to its limit becomes heavier. So there is a compromise between (statistical and numerical) estimation and computational complexity. That said, choosing a smaller (in order of magnitude) than h^2 does not improve our bound (21). When a is that small, the main source of error comes from estimating the density, rather than the accuracy of the gradient ascent scheme, and the resulting rate is

$$\sup_{t\ge 0} \|\hat{x}_a(t) - x(t)\| \le C\gamma_n \left(\frac{\log n}{n}\right)^{2\delta/(d+6)},$$

for any choice of sequence (γ_n) with $\gamma_n \to \infty$. We note that faster rates are possible for densities that are C^k for $k > 3$, since they can be estimated more accurately by a higher order kernel (Devroye and Gyorfi, 1985). We also mention that the curse of dimensionality is at play here since we are estimating a nonparametric density.

4. Proofs

We start in Section 4.1 with some auxiliary results that will be used in the proofs of our main results. Theorem 1 and Theorem 2 are proved in Sections 4.2 and 4.3 respectively. We prove Lemma 2 and Lemma 3 in Sections 4.4 and 4.5, and then Theorem 3 in Section 4.6.

4.1 Preliminary Results

The following is a discrete version of Gronwall's lemma. The proof is straightforward and left to the reader.

Lemma 4 *Let* $(y_\ell : \ell \ge 0)$ *be a sequence of non-negative real numbers such that*

$$y_{\ell+1} \le Q_1 + (1 + Q_2)y_\ell.$$

Then

$$y_\ell \le y_0 e^{Q_2 \ell} + \frac{e^{Q_2\ell} - 1}{Q_2} Q_1.$$

The result below is on the behavior of the upper level set near a stable local maximum.

Lemma 5 *Suppose that* f *is of class* C^3. *Let* $x^\star$ *be the location of a stable local maxima of* f *where* $H_f(x^\star)$ *has all eigenvalues in* $(-\overline{\nu}, -\underline{\nu})$ *with* $\overline{\nu} > \underline{\nu} > 0$. *For* $\epsilon > 0$, *let* $\mathcal{C}(\epsilon)$ *be the connected component of* $\mathcal{L}_f(f(x^\star) - \epsilon)$ *that contains* $x^\star$. *Then there is a constant* $C_5 = C_5(f, x^\star)$ *such that*

$$\bar{B}(x^\star, \sqrt{2\epsilon/\overline{\nu}}) \subset \mathcal{C}(\epsilon) \subset \bar{B}(x^\star, \sqrt{2\epsilon/\underline{\nu}}), \quad for\ all\ \epsilon \le C_5, \quad (22)$$

and

$$f(x^\star) - f(x) \le \frac{\overline{\nu}}{2}\|x - x^\star\|^2, \quad for\ all\ x\ such\ that\ \|x - x^\star\| \le \sqrt{2C_5/\overline{\nu}}. \quad (23)$$

Proof Fix $r > 0$. Let $\mathbf{H}$ and κ_3 be short for $H_f(x^\star)$ and $\kappa_3(f, \bar{B}(x^\star, r))$, respectively. Let $\underline{\nu} < \underline{\nu}' < \overline{\nu}' < \overline{\nu}$ be such that $H_f(x^\star)$ has all eigenvalues in $[-\overline{\nu}', -\underline{\nu}']$. First, we prove (22). A Taylor development of f at $x \in \bar{B}(x^\star, r)$ gives

$$f(x) = f(x^\star) + \frac{1}{2}\mathbf{H}[x - x^\star, x - x^\star] + R(x, x^\star), \quad \text{with } |R(x, x^\star)| \le \frac{\kappa_3}{6}\|x - x^\star\|^3. \quad (24)$$

When $x \in \bar{B}(x^\star, r)$, using the Taylor expansion (24), we get that

$$\begin{aligned} f(x) &\le f(x^\star) - \frac{\underline{\nu}'}{2}\|x^\star - x\|^2 + \frac{\kappa_3}{6}\|x^\star - x\|^3 \\ &\le f(x^\star) - \frac{\underline{\nu}}{2}\|x^\star - x\|^2 \end{aligned}$$

when $\|x^\star - x\| \le \xi_1 := \frac{3(\underline{\nu}' - \underline{\nu})}{\kappa_3} \wedge r$. Fix $0 < \epsilon < \frac{\underline{\nu}\xi_1^2}{2}$ so that $\sqrt{\left(\frac{2\epsilon}{\underline{\nu}}\right)} < \xi_1$. We then have $f(x) < f(x^\star) - \epsilon$ when $\sqrt{\left(\frac{2\epsilon}{\underline{\nu}}\right)} < \|x^\star - x\| \le \xi_1$. This implies that

$$\mathcal{L}_f(f(x^\star) - \epsilon) \subset \bar{B}\big(x^\star, \sqrt{\left(\tfrac{2\epsilon}{\underline{\nu}}\right)}\big) \cup \bar{B}\big(x^\star, \xi_1\big)^{\mathsf{c}},$$

and since the two sets on the right-hand side are disconnected, while $\mathcal{C}(\epsilon)$ is connected and contains $x^\star$, necessarily, $\mathcal{C}(\epsilon) \subset B\big(x^\star, \sqrt{\big(\frac{2\epsilon}{\underline{\nu}}\big)}\big)$.

We also get using (24) that

$$\begin{aligned} f(x) &\geq f(x^\star) - \frac{\overline{\nu}'}{2}\|x^\star - x\|^2 - \frac{\kappa_3}{6}\|x^\star - x\|^3 \\ &\geq f(x^\star) - \frac{\overline{\nu}}{2}\|x^\star - x\|^2 \end{aligned}$$

when $\|x^\star - x\| \leq \xi_2 := \frac{3(\overline{\nu}-\overline{\nu}')}{\kappa_3} \wedge r$. Fix $0 < \epsilon < \frac{\overline{\nu}\xi_2^2}{2}$ so that $\sqrt{\big(\frac{2\epsilon}{\overline{\nu}}\big)} < \xi_2$. Then whenever $\|x^\star - x\| \leq \sqrt{\big(\frac{2\epsilon}{\overline{\nu}}\big)}$, we have $f(x) \geq f(x^\star) - \epsilon$. Reasoning as above, we obtain $B\big(x^\star, \sqrt{\big(\frac{2\epsilon}{\overline{\nu}}\big)}\big) \subset \mathcal{C}(\epsilon)$.

Therefore, by choosing $C_5 < \xi_1 \wedge \xi_2$, we see that (22) holds. Note that ξ_1 and ξ_2 depend on r. Since we do not need an explicit value for the constant C_5, we leave $r > 0$ arbitrarily fixed.

The bound (23) is a direct consequence of (22). ■

Next is a result establishing exponential convergence rates for the gradient flow of a smooth function ending at a stable local maximum.

Lemma 6 *Suppose that f is of class C^3. Let $\{\gamma(t) : t \geq 0\}$ be the flow line of f starting at x_0 and ending at $x^\star$ where $H_f(x^\star)$ has all its eigenvalues in $(-\infty, -\underline{\nu})$, with $\underline{\nu} > 0$. Then, there is $C_6 = C_6(f, x_0)$ such that, for all $t \geq 0$,*

$$\|\gamma(t) - x^\star\| \leq C_6 e^{-\underline{\nu}t}, \tag{25}$$

and

$$f(x^\star) - f(\gamma(t)) \leq C_6 e^{-2\underline{\nu}t}. \tag{26}$$

Proof Note that since γ has beginning and ending points, $\{\gamma(t) : t \geq 0\}$ is bounded. Let $r_0 > 0$ be such that $\{\gamma(t) : t \geq 0\}$ is contained in the ball $\bar{B}(x^\star, r_0)$. Let $\mathbf{H}$ and κ_3 be short for $H_f(x^\star)$ and $\kappa_3(f, \bar{B}(x^\star, r_0))$, respectively. A Taylor development of ∇f at $x \in \bar{B}(x^\star, r_0)$ gives

$$\nabla f(x) = \mathbf{H}(x - x^\star) + R(x, x^\star),$$

with

$$\|R(x, x^\star)\| \leq \kappa_3 \tfrac{\sqrt{d}}{2}\|x - x^\star\|^2.$$

Therefore, we have,

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\gamma(t) - x^\star\right) - \mathbf{H}\left(\gamma(t) - x^\star\right) = R\left(\gamma(t), x^\star\right),$$

and so, since $\gamma(0) = x_0$, γ satisfies the relation

$$\gamma(t) - x^\star = e^{t\mathbf{H}}(x_0 - x^\star) + \int_0^t e^{(t-s)\mathbf{H}} R\left(\gamma(s), x^\star\right) \mathrm{d}s.$$

Since all the eigenvalues of $\mathbf{H}$ are in $(-\infty, -\underline{\nu})$, there is $\nu > \underline{\nu}$ such that we have

$$\left\|e^{\alpha\mathbf{H}}\right\| \leq e^{-\nu\alpha}, \quad \text{for all } \alpha > 0.$$

Then,

$$\|\gamma(t) - x^\star\| \leq e^{-\nu t}\|x_0 - x^\star\| + \kappa_3 \tfrac{\sqrt{d}}{2}\int_0^t e^{-\nu(t-s)}\|\gamma(s) - x^\star\|^2 \mathrm{d}s.$$

Set $u(t) = e^{\nu t}\|\gamma(t) - x^\star\|$ and $U(t) = \|x_0 - x^\star\| + \kappa_3 \frac{\sqrt{d}}{2}\int_0^t e^{\nu s}\|\gamma(s) - x^\star\|^2 \mathrm{d}s$. Then $u(t) \leq U(t)$ and $U'(t) = \kappa_3 \frac{\sqrt{d}}{2} e^{-\nu t} u^2(t)$, so

$$\frac{U'(t)}{U(t)} = \kappa_3 \tfrac{\sqrt{d}}{2} e^{-\nu t} u(t) \frac{u(t)}{U(t)} \leq \kappa_3 \tfrac{\sqrt{d}}{2} e^{-\nu t} u(t) = \kappa_3 \tfrac{\sqrt{d}}{2}\|\gamma(t) - x^\star\|.$$

But since $\gamma(t) \to x^\star$ as $t \to \infty$, there exists $t_0 > 0$ such that $\|\gamma(t) - x^\star\| \leq \frac{2(\nu - \underline{\nu})}{\kappa_3\sqrt{d}}$ for all $t \geq t_0$. By integrating between t_0 and t, we deduce that

$$\log U(t) \leq \log U(t_0) + (\nu - \underline{\nu})(t - t_0),$$

and so

$$\|\gamma(t) - x^\star\| = e^{-\nu t} u(t) \leq e^{-\nu t} U(t) \leq Q_0 e^{-\underline{\nu}t}, \quad \text{for all } t \geq t_0,$$

with $Q_0 := U(t_0)e^{-(\nu - \underline{\nu})t_0}$. For $t < t_0$, we simply have $\|\gamma(t) - x^\star\| \leq Q_1 e^{-\underline{\nu}t}$, where $Q_1 = \max_{0 \leq t \leq t_0}\|\gamma(t) - x^\star\| e^{\underline{\nu}t}$. Therefore (25) holds with the constant $Q_2 = \max\{Q_0, Q_1\}$.

We now turn to proving (26). For any x in $\bar{B}(x^\star, r_0)$, we have

$$f(x) = f(x^\star) + \frac{1}{2}\mathbf{H}[x - x^\star, x - x^\star] + R(x, x^\star),$$

for all x in $\bar{B}(x^\star, r_0)$, where R is a different function (now real valued) satisfying

$$|R(x, x^\star)| \leq \tfrac{\kappa_3}{6}\|x - x^\star\|^3.$$

Then

$$\begin{aligned} f(x^\star) - f(\gamma(t)) &\leq \tfrac{1}{2}\|\mathbf{H}\|\,\|\gamma(t) - x^\star\|^2 + \tfrac{\kappa_3}{6}\|\gamma(t) - x^\star\|^3 \\ &\leq (\tfrac{1}{2}\|\mathbf{H}\| + Q_3)Q_2^2 e^{-2\underline{\nu}t}, \end{aligned}$$

where $Q_3 = \frac{\kappa_3}{6}\max_{t \geq 0}\|\gamma(t) - x^\star\|$ and we applied (25) in the second line with Q_2 defined above. Therefore, (26) holds with the constant $Q_4 := (\|\mathbf{H}\|/2 + Q_3)Q_2^2$.

We then take $C_6 = \max(Q_2, Q_4)$. ■

The following, adapted from Hirsch et al. (2004, Sec 17.5), is a stability result for autonomous gradient flows.

Lemma 7 *Suppose f and g are of class C^3. Let $x_0 \in \mathbb{R}^d$, and suppose that*

$$\|\nabla f(x) - \nabla g(x)\| < \eta, \quad \forall x \in \mathcal{S} := \mathcal{L}_f(f(x_0)) \cup \mathcal{L}_g(g(x_0)).$$

Let κ be a Lipschitz constant for ∇f on $\mathcal{S}$. Let $(x(t) : t \geq 0)$ and $(y(t) : t \geq 0)$ be the flow lines of f and g starting at x_0, supposed to be defined on $[0, \infty)$. Then,

$$\|x(t) - y(t)\| \leq \frac{\eta}{\kappa}\big[e^{\kappa t} - 1\big], \quad \forall t \geq 0.$$

Next is a result on the stability of local maxima.

Lemma 8 *Suppose f and g are of class C^3, and have local maxima at x and y, respectively, with $H_f(x)$ having all eigenvalues in $(-\infty,-\nu]$ for some $\nu>0$. Then for any $C_8 \geq \max\{1, \frac{2}{\sqrt{\nu}}, \frac{4\kappa}{3\nu}\}$, where $\kappa = \max\big(\kappa_3(f,\bar{B}(x,1)),\kappa_3(g,\bar{B}(y,1))\big)$,*

$$\|x-y\| \leq 1/C_8 \quad \Rightarrow \quad \|x-y\| \leq C_8\big(\|f(x)-g(x)\|+\|f(y)-g(y)\|\big)^{1/2}. \tag{27}$$

Proof Let $\mathbf{H}_f$ and $\mathbf{H}_g$ be short for $H_f(x)$ and $H_g(y)$, respectively. We develop f and g around x and y, respectively. Assuming $\|x-y\|\leq 1$, we have

$$\begin{aligned} f(y) &= f(x) + \tfrac{1}{2}\mathbf{H}_f[x-y,x-y] + R_f(x,y), \qquad \text{with} \quad |R_f(x,y)| \leq \tfrac{\kappa}{6}\|x-y\|^3; \\ g(x) &= g(y) + \tfrac{1}{2}\mathbf{H}_g[x-y,x-y] + R_g(x,y), \qquad \text{with} \quad |R_g(x,y)| \leq \tfrac{\kappa}{6}\|x-y\|^3. \end{aligned}$$

Summing these two equalities, we obtain

$$\frac{1}{2}(\mathbf{H}_f+\mathbf{H}_g)[x-y,x-y] = f(y)-g(y)+g(x)-f(x)-R_f(x,y)-R_g(x,y).$$

By the triangle inequality and the fact that $\mathbf{H}_g$ is negative semidefinite,

$$\nu\|x-y\|^2 \leq \|(\mathbf{H}_f+\mathbf{H}_g)[x-y,x-y]\| \leq 2\|f(x)-g(x)\|+2\|f(y)-g(y)\|+\tfrac{2\kappa}{3}\|x-y\|^3.$$

When $\|x-y\| \leq \min\big(\frac{3\nu}{4\kappa},1\big)$, we have $\nu\|x-y\|^2 - \frac{2\kappa}{3}\|x-y\|^3 \geq \frac{\nu}{2}\|x-y\|^2$, and therefore

$$\|x-y\|^2 \leq \tfrac{4}{\nu}\left(\|f(x)-g(x)\|+\|f(y)-g(y)\|\right),$$

and from this we conclude that (27) holds with $C_8 = \max(\sqrt{(\frac{4}{\nu})}, \frac{4\kappa}{3\nu}, 1)$. ■

4.2 Proof of Theorem 1

Below, C_m refers to the constant defined in Lemma m.

We assume that x_0 is not a critical point of f, for otherwise $x_0 = x^\star$ and there is nothing to prove. Let $t_\ell = a\ell$, which is the time at which the polygonal line $x_a(t)$ passes through x_ℓ. Let $\mathcal{L}_0$ be short for $\mathcal{L}_f(f(x_0))$. Note that $(x(t) : t \geq 0)$ is bounded since x is a continuous flow line with a beginning and ending points. Let r_0 be large enough that $(x(t) : t \geq 0) \subset \bar{B}(x_0, r_0)$.

Claim. *Without loss of generality, we may assume that $\mathcal{L}_0$ is bounded.* To see this, suppose the result is true when $\mathcal{L}_0 \subset \bar{B}(x_0, 3r_0)$. We shall prove that it remains true when $\mathcal{L}_0 \not\subseteq \bar{B}(x_0,3r_0)$. Given such a situation, build another function $\tilde{f}$ in such a way that $\tilde{f}$ is C^3 on $\mathbb{R}^d$ with $\tilde{f}(x) = f(x)$ for all $x \in \bar{B}(x_0,2r_0)$ and $\tilde{f}(x) < f(x_0)$ for $x \notin \bar{B}(x_0,3r_0)$, so that $\mathcal{L}_{\tilde{f}}(\tilde{f}(x_0)) \subset \bar{B}(x_0,3r_0)$. To verify that such a function exists, consider the smoothing function $s:\mathbb{R}^d \to \mathbb{R}$ defined by

$$s(x) = \frac{1}{\int_{\bar{B}(0,1)} e^{-1/(1-\|x\|^2)}\mathrm{d}x} e^{-1/(1-\|x\|^2)}\mathbf{1}_{B(0,1)}(x), \quad x \in \mathbb{R}^d,$$

and its dilated versions s_a defined by $s_a(x) = a^{-d}s(x/a)$ for $a>0$, where $\mathbf{1}_{\bar{B}(0,1)}(x) = 1$ if $x \in \bar{B}(0,1)$ and 0 otherwise. Define the function g by $g(x) = \mathbf{1}_{\bar{B}(0,5r_0/2)} \star s_{r_0/2}(x-x_0)$. Then g is of class C^∞, $g(x)=1$ for $x \in \bar{B}(x_0,2r_0)$, $g(x)=0$ if $x \notin \bar{B}(x_0,3r_0)$, and $0<g(x)<1$ when $2r_0 < \|x-x_0\| < 3r_0$. Then we may take $\tilde{f} = fg$.

Therefore, (13) and (14) hold for $\tilde{f}$, for constants $\tilde{A}$ and $\tilde{C}$, with the same exponent δ as given in (14). Denote by $\tilde{x}$ and $\tilde{x}_a$ the flow line and polygonal curve constructed from $\tilde{f}$ in the same way x and x_a are from f. Then, assuming $\tilde{C}a^\delta \leq r_0$, we see by the triangle inequality that $\tilde{x}(t)$ and $\tilde{x}_a(t)$ are determined by $\tilde{f}$ restricted to $\bar{B}(x_0,2r_0)$, and since $\tilde{f}$ coincides with f there, $x(t) = \tilde{x}(t)$ and $x_a(t) = \tilde{x}_a(t)$, so that (13) and (14) are valid for f if $a \leq \min\{\tilde{A}, (r_0/\tilde{C})^{1/\delta}\}$.

From now on, we assume that $\mathcal{L}_0$ is bounded. Note that $\mathcal{L}_0$ is also closed since f is continuous, so in fact $\mathcal{L}_0$ is compact. Let

$$S = \mathcal{L}_0 \oplus \bar{B}(0,\kappa_1(f,\mathcal{L}_0)) =: \{x \in \mathbb{R}^d : \operatorname{dist}(x,\mathcal{L}_0) \leq \kappa_1(f,\mathcal{L}_0)\}, \tag{28}$$

where $\operatorname{dist}(x,\mathcal{L}_0) = \inf\{\|x-y\| : y \in \mathcal{L}_0\}$. For any $0 \leq \ell \leq 3$, let $\kappa_\ell = \kappa_\ell(f,S)$, where S is defined in (28). For any $x \in \mathbb{R}^d$, let

$$\kappa_2(x) = \kappa_2(f,\bar{B}(x,\|\nabla f(x)\|)) = \sup\{\|f^{(2)}(y)\| : y \in \bar{B}(x,\|\nabla f(x)\|)\}. \tag{29}$$

Notice that $\mathcal{L}_0 \subset S$ and that, by construction, $\bar{B}(x,\|\nabla f(x)\|) \subset S$ for any $x \in \mathcal{L}_0$. Hence, $\kappa_2(x) \leq \kappa_2$ for all x in $\mathcal{L}_0$.

Claim. *For any $x \in \mathbb{R}^d$ with $\nabla f(x) \neq 0$ and any $0 \leq b \leq 1 \wedge (2\sqrt{d}\kappa_2(x))^{-1}$, we have $f(x+b\nabla f(x)) > f(x)$ and f is increasing along the line segment $[x, x+b\nabla f(x)]$.* Using a Taylor expansion of f at x, we have

$$f(x+b\nabla f(x)) = f(x) + b\|\nabla f(x)\|^2 + R(x,b),$$

where $|R(x,b)| \leq \frac{1}{2}b^2\kappa_2(x)\|\nabla f(x)\|^2 \leq \frac{b}{2}\|\nabla f(x)\|^2$, since $b \leq (2\sqrt{d}\kappa_2(x))^{-1} \leq \kappa_2^{-1}(x)$. Then

$$\zeta(b) := f(x+b\nabla f(x)) \geq f(x) + \frac{b}{2}\|\nabla f(x)\|^2 > f(x). \tag{30}$$

Now for any $0<\beta<b$,

$$\zeta'(\beta) = \nabla f\left(x+\beta\nabla f(x)\right)\cdot\nabla f(x),$$

and by a Taylor expansion of the components of ∇f

$$\nabla f\left(x+\beta\nabla f(x)\right) = \nabla f(x) + R_2(x,\beta),$$

where $\|R_2(x,\beta)\| \leq \beta\sqrt{d}\kappa_2(x)\|\nabla f(x)\|$. Hence, for any $0<\beta<b$

$$\zeta'(\beta) = \|\nabla f(x)\|^2 + R(x,\beta)\cdot\nabla f(x) \geq \frac{1}{2}\|\nabla f(x)\| > 0$$

since $\beta < b \leq (2\sqrt{d}\kappa_2(x))^{-1}$ and so f is increasing along the line segment $[x, x+b\nabla f(x)]$.

Claim. *For a sufficiently small, $x_a(t) \in \mathcal{L}_0$ for all $t \geq 0$.* Indeed, since $\kappa_2(x) \leq \kappa_2$ for all x in $\mathcal{L}_0$, we have $1 \wedge (2\sqrt{d}\kappa_2(x))^{-1} \geq 1 \wedge (2\sqrt{d}\kappa_2)^{-1}$ for all x in $\mathcal{L}_0$. Consequently, by the previous claim, for any x in $\mathcal{L}_0$ and $a \leq 1 \wedge (2\sqrt{d}\kappa_2)^{-1}$, we have $f(x+a\nabla f(x)) > f(x)$ and

the values of f are increasing along the line segment $[x, x + a\nabla f(x)]$. In particular, since x_a starts at $x_0 \in \mathcal{L}_0$, we have $f(x_1) = f(x_0 + \nabla f(x_0)) > f(x_0)$, and the segment $[x_0, x_1]$ belongs to $\mathcal{L}_0$. By recursion, we deduce that $x_a(t)$ belongs to $\mathcal{L}_0$ for all $t \geq 0$.

From now on, we assume that

$$a \leq A_1 := 1 \wedge (2\sqrt{d}\kappa_2)^{-1}. \tag{31}$$

Claim. *f is increasing along the polygonal curve x_a.* By the previous arguments, the values of f are increasing along the line segment $[x_\ell, x_{\ell+1}]$, for all $\ell \geq 0$.

Claim. *(x_ℓ) converges to a critical point of f.* We just showed that the sequence $(f(x_\ell) : \ell \geq 0)$ is increasing, and since it is bounded by κ_0, it converges. By the first inequality in (30) and the fact that $\|x_{\ell+1} - x_\ell\| = a\|\nabla f(x_\ell)\|$ by construction, we have

$$f(x_{\ell+1}) - f(x_\ell) \geq \frac{1}{2}a\|\nabla f(x_\ell)\|^2 = \frac{1}{2a}\|x_{\ell+1} - x_\ell\|^2, \tag{32}$$

for all $\ell \geq 1$. Hence, for all $\ell \geq 1$, and all $k \geq 1$, we have

$$f(x_{\ell+k}) - f(x_\ell) \geq \frac{1}{2a}\sum_{i=1}^{k}\|x_{\ell+i} - x_\ell\|^2 \geq \frac{1}{2a}\|x_{\ell+k} - x_\ell\|^2,$$

by the triangle inequality. Since $(f(x_\ell))$ is convergent, it is a Cauchy sequence, and consequently, so is (x_ℓ), so that $\tilde{x} := \lim_{\ell\to\infty} x_\ell$ exists. And by (32) and the fact that f is C^1, we have

$$\nabla f(\tilde{x}) = \lim_{\ell\to\infty} \nabla f(x_\ell) = 0,$$

so that $\tilde{x}$ is a critical point of f.

Claim. *We have*

$$\|x(t_\ell) - x_\ell\| \leq \left[e^{\ell a\kappa_2\sqrt{d}} - 1\right]\kappa_1 a, \quad \forall \ell \geq 0. \tag{33}$$

Indeed, let $e_\ell = x(t_\ell) - x_\ell$. Using (6), we have

$$\begin{aligned} e_{\ell+1} &= x(t_{\ell+1}) - x_{\ell+1} \\ &= e_\ell + [x(t_{\ell+1}) - x(t_\ell) - a\nabla f(x(t_\ell))] + a\left[\nabla f(x(t_\ell)) - \nabla f(x_\ell)\right]. \end{aligned} \tag{34}$$

By the definition of κ_2, and a Taylor expansion,

$$\|\nabla f(x(t_\ell)) - \nabla f(x_\ell)\| \leq \sqrt{d}\kappa_2\|x(t_\ell) - x_\ell\| = \sqrt{d}\kappa_2\|e_\ell\|. \tag{35}$$

We also have

$$\begin{aligned} x(t_{\ell+1}) - x(t_\ell) - a\nabla f(x(t_\ell)) &= \int_{t_\ell}^{t_{\ell+1}} x'(s)\mathrm{d}s - \frac{a}{t_{\ell+1} - t_\ell}\int_{t_\ell}^{t_{\ell+1}} x'(t_\ell)\mathrm{d}s \\ &= \int_{t_\ell}^{t_{\ell+1}} (x'(s) - x'(t_\ell))\mathrm{d}s, \end{aligned}$$

by the definitions of $x(t)$ and t_ℓ. Consequently,

$$\|x(t_{\ell+1}) - x(t_\ell) - a\nabla f(x(t_\ell))\| \leq \int_{t_\ell}^{t_{\ell+1}} \|x'(s) - x'(t_\ell)\|\mathrm{d}s.$$

For $s \in [t_\ell, t_{\ell+1}]$, we have

$$\|x'(s) - x'(t_\ell)\| = \|\nabla f(x(s)) - \nabla f(x(t_\ell))\| \leq \kappa_2\sqrt{d}\|x(s) - x(t_\ell)\|,$$

and

$$\|x(s) - x(t_\ell)\| = \left\|\int_{t_\ell}^{s} x'(t)\mathrm{d}t\right\| \leq \int_{t_\ell}^{s}\|x'(t)\|\mathrm{d}t = \int_{t_\ell}^{s}\|\nabla f(x(t))\|\mathrm{d}t \leq \kappa_1(s - t_\ell).$$

Hence

$$\|x'(s) - x'(t_\ell)\| \leq \sqrt{d}\kappa_2\kappa_1(s - t_\ell),$$

and, recalling that $t_\ell = a\ell$,

$$\|x(t_{\ell+1}) - x(t_\ell) - a\nabla f(x(t_\ell))\| \leq \sqrt{d}\kappa_2\kappa_1(t_{\ell+1} - t_\ell)^2 = \sqrt{d}\kappa_2\kappa_1 a^2. \tag{36}$$

Plugging (36) and (35) into (34), we deduce that

$$\|e_{\ell+1}\| \leq \sqrt{d}\kappa_2\kappa_1 a^2 + (1 + \sqrt{d}\kappa_2 a)\|e_\ell\|.$$

The inequality (33) is now a direct consequence of Lemma 4. (Recall that $x(t_0) = x_0$.)

Claim. *(x_ℓ) converges to $x^\star$.* By this we mean that $\tilde{x}$ coincides with $x^\star$. Indeed, for any $\eta > 0$, denote by $\mathcal{C}(\eta)$ the connected component of $\mathcal{L}_f(f(x^\star) - \eta)$ that contains $x^\star$. Let $\mathbf{H}$ be a shorthand for $H_f(x^\star)$. Suppose all the eigenvalues of $\mathbf{H}$ are in $(-\overline{\nu}, -\underline{\nu})$ for some $\overline{\nu} > \underline{\nu} > 0$. Because $\mathbf{H}$ is negative definite, when $\epsilon > 0$ is small enough $\bar{B}(x^\star, \epsilon)$ contains no critical point of f other than $x^\star$. Let ℓ_ϵ be such that $\|x_\ell - \tilde{x}\| \leq \epsilon/3$ when $\ell \geq \ell_\epsilon$, which is well-defined since (x_ℓ) converges to $\tilde{x}$. Using the triangle inequality, and then Lemma 6 and (33), for $\ell = \ell_{\epsilon,a} := \max\left\{\ell_\epsilon, \left\lceil \frac{1}{a\underline{\nu}}\log(3/(C_6\epsilon))\right\rceil\right\}$, we have

$$\begin{aligned} \|x^\star - \tilde{x}\| &\leq \|x^\star - x(t_\ell)\| + \|x(t_\ell) - x_\ell\| + \|x_\ell - \tilde{x}\| \\ &\leq \epsilon/3 + \left[e^{\sqrt{d}\kappa_2 a\ell_{\epsilon,a}} - 1\right]\kappa_1 a + \epsilon/3 \\ &\leq \epsilon, \end{aligned}$$

when $a \leq A_2$ for some $A_2 > 0$ (depending on $\epsilon > 0$) sufficiently small. Hence, $\tilde{x} \in \bar{B}(x^\star, \epsilon)$. Since $\tilde{x}$ is a critical point, and the only critical point in $\bar{B}(x^\star, \epsilon)$ is $x^\star$, necessarily $\tilde{x} = x^\star$. This proves (13) for $a \leq A := \min(1, A_1, A_2)$, where A_1 is defined in (31).

Henceforth, we assume that $a \leq A$, so that $x_\ell \to x^\star$ as $\ell \to \infty$, and focus on proving (14).

Bound for large t. A Taylor expansion gives

$$\nabla f(x) = \mathbf{H}(x - x^\star) + R(x, x^\star), \quad \text{where } \|R(x, x^\star)\| \leq \tfrac{\sqrt{d}}{2}\kappa_3\|x - x^\star\|^2.$$

We then have

$$\begin{aligned} x_{\ell+1} - x^\star &= x_\ell - x^\star + a\nabla f(x_\ell) \\ &= (\mathbf{I} + a\mathbf{H})(x_\ell - x^\star) + aR(x_\ell, x^\star), \end{aligned}$$

so that

$$\|x_{\ell+1} - x^\star\| \leq (1 - a\nu)\|x_\ell - x^\star\| + a\tfrac{\sqrt{d}}{2}\kappa_3\|x_\ell - x^\star\|^2,$$

for some $\nu > \underline{\nu}$. As $x_\ell \to x^\star$, there is ℓ_0 such that, for $\ell \geq \ell_0$, $\nu - \frac{\sqrt{d}}{2}\kappa_3\|x_\ell - x^\star\| > \underline{\nu}$, implying

$$\|x_{\ell+1} - x^\star\| \leq (1 - a\underline{\nu})\|x_\ell - x^\star\|, \quad \forall \ell \geq \ell_0.$$

By recursion, we deduce that there is a constant $Q_1 > 0$ such that

$$\|x_\ell - x^\star\| \leq Q_1(1 - a\underline{\nu})^\ell \leq Q_1 e^{-\underline{\nu}\ell a}, \quad \forall \ell \geq 0. \tag{37}$$

Fix $t \in [t_\ell, t_{\ell+1}]$. Starting with the triangle inequality, we have

$$\begin{aligned}\|x(t) - x_a(t)\| &\leq \|x(t) - x_\star\| + \|x_\star - x_\ell\| + \|x_\ell - x_a(t)\| \\ &\leq C_6 e^{-\underline{\nu}t} + Q_1 e^{-\underline{\nu}\ell a} + (t - t_\ell)\|\nabla f(x_\ell)\| \\ &\leq Q_2 e^{-\underline{\nu}t} + \kappa_1 a. \end{aligned} \tag{38}$$

In the first line, we applied (25), (37), and used the definition of x_a. In the second line, we let $Q_2 = C_6 + Q_1 e^{\underline{\nu}A}$ and used the definition of κ_1 in (11).

Bound for small t. On the other hand, we also have

$$\begin{aligned}\|x(t) - x_a(t)\| &\leq \|x(t) - x(t_\ell)\| + \|x(t_\ell) - x_\ell\| + \|x_\ell - x_a(t)\| \\ &\leq \kappa_1(t_{\ell+1} - t_\ell) + \|x(t_\ell) - x_\ell\| + \|x_\ell - x_{\ell+1}\| \\ &= \kappa_1 a + \|x(t_\ell) - x_\ell\| + a\|\nabla f(x_\ell)\| \\ &\leq 2\kappa_1 a + \|x(t_\ell) - x_\ell\|. \end{aligned}$$

Because f is C^3, there is $\epsilon > 0$ such that all the eigenvalues of $H_f(x)$ exceed $-\overline{\nu}$ when $x \in \bar{B}(x^\star, \epsilon)$. Let ℓ_ϵ be such that $x(t), x_\ell \in \bar{B}(x^\star, \epsilon)$ for all $t \geq a\ell_\epsilon$ and $\ell \geq \ell_\epsilon$, which implies

$$\|\nabla f(x(t)) - \nabla f(x_\ell)\| \leq \overline{\nu}\|x(t) - x_\ell\|.$$

Using this inequality instead of (35), we can refine (33) into

$$\|x(t_\ell) - x_\ell\| \leq \left[e^{\ell a\overline{\nu}} - 1\right]\kappa_1 a, \quad \forall \ell \geq \ell_\epsilon,$$

and since ϵ is fixed, we can combine this with (33) to get

$$\|x(t_\ell) - x_\ell\| \leq \left[e^{\ell a\overline{\nu}} - 1\right]\kappa_1 a + Q_3 a, \quad \forall \ell \geq 0, \tag{39}$$

for some constant Q_3. We thus have

$$\|x(t) - x_a(t)\| \leq \left[2\kappa_1 + (e^{\overline{\nu}t} - 1)\kappa_1 + Q_3\right] a, \tag{40}$$

using the fact that $t \geq t_\ell = a\ell$.

Combining (38) and (40), we have

$$\|x(t) - x_a(t)\| \leq (\kappa_1 + Q_3)a + \min\left\{\kappa_1 a e^{\overline{\nu}t}, Q_2 e^{-\underline{\nu}t}\right\}.$$

From this, we deduce (14) from elementary calculations.

4.3 Proof of Theorem 2

Below, C_m refers to the constant defined in Lemma m.

Arguing as in the proof of Theorem 1, we may assume, without any loss of generality, that $\mathcal{L}_f(f(x_0)/2) \subset \bar{B}(x_0, 3r_0)$. So from now on, we assume that $\mathcal{L}_f(f(x_0)/2)$ is compact and we set $S = \mathcal{L}_f(f(x_0)/2)$. For any $0 \leq \ell \leq 3$, we also let κ_ℓ be short for $\kappa_\ell(f, S)$.

Claim. *For η_0 sufficiently small, $\hat{x}(t) \in S$.* Indeed, suppose there is $t \geq 0$ such that $\hat{x}(t) \notin S$. Fix $\epsilon = f(x_0)/2$. Then, by continuity, there is $0 \leq t' < t$ such that $f(\hat{x}(t')) = f(x_0) - \epsilon$. Since $\hat{x}(t'), x_0 \in S$, we have

$$\begin{aligned}\hat{f}(\hat{x}(t')) &= \hat{f}(\hat{x}(t')) - f(\hat{x}(t')) + f(\hat{x}(t')) \\ &\leq \eta_0 + f(x_0) - \epsilon \\ &= \eta_0 + \hat{f}(x_0) + f(x_0) - \hat{f}(x_0) - \epsilon \\ &\leq \hat{f}(x_0) + 2\eta_0 - \epsilon, \end{aligned}$$

by the triangle inequality, applied twice. Since $\hat{f}(\hat{x}(t')) \geq \hat{f}(x_0)$, we see that this situation does not arise when $\eta_0 < \epsilon/2$.

Claim. *$\hat{x}^\star = \lim_{t\to\infty}\hat{x}(t)$ is well defined and is close to $x^\star$.* Since $\hat{f}$ is of class C^3 by assumption, the map $x \mapsto \nabla\hat{f}(x)$ is C^1, and since $\hat{x}(t)$ stays in S and S is compact, $\hat{x}(t)$ is defined for all $t \geq 0$ by the first corollary to the first theorem in (Hirsch et al., 2004, Sec. 17.4). For any $\epsilon \in (0, C_5)$, with $\epsilon < f(x^\star) - f(x_0)/2$, let t_ϵ be such that $x(t) \in \bar{B}(x^\star, \sqrt{(2\epsilon/\overline{\nu})})$ for all $t \geq t_\epsilon$, which is well-defined since $x(t) \to x^\star$ as $t \to \infty$. By Lemma 7, we have

$$\|\hat{x}(t) - x(t)\| \leq \frac{\eta_1}{\sqrt{d}\kappa_2}e^{\sqrt{d}\kappa_2 t}, \quad \forall t \geq 0. \tag{41}$$

Hence

$$\|\hat{x}(t_\epsilon) - x^\star\| \leq \|\hat{x}(t_\epsilon) - x(t_\epsilon)\| + \|x(t_\epsilon) - x^\star\| \leq \frac{\eta_1}{\sqrt{d}\kappa_2}e^{\sqrt{d}\kappa_2 t_\epsilon} + \sqrt{\frac{2\epsilon}{\overline{\nu}}} =: \delta_1. \tag{42}$$

Assume that η_1 and ϵ are small enough that $\delta_1 < \sqrt{(2C_5/\overline{\nu})}$. Letting $\mathcal{C}(\epsilon)$ be as in Lemma 5, by (22) we have

$$\bar{B}(x^\star, \delta_1) \subset \mathcal{C}(\epsilon_1),$$

with $\epsilon_1 := \frac{\overline{\nu}}{2}\delta_1^2$. Thus $\hat{x}(t_\epsilon)$ belongs to $\mathcal{C}(\epsilon_1)$ and in particular $f(\hat{x}(t_\epsilon)) \geq f(x^\star) - \epsilon_1$. Using this last inequality, we deduce by the triangle inequality and the fact that $t \mapsto \hat{f}(\hat{x}(t))$ is increasing that for all $t \geq t_\epsilon$,

$$f(\hat{x}(t)) \geq \hat{f}(\hat{x}(t)) - \eta_0 \geq \hat{f}(\hat{x}(t_\epsilon)) - \eta_0 \geq f(\hat{x}(t_\epsilon)) - 2\eta_0 \geq f(x^\star) - \epsilon_2,$$

where $\epsilon_2 := \epsilon_1 + 2\eta_0$. Since $\hat{x}(t_\epsilon) \in \mathcal{C}(\epsilon_1) \subset \mathcal{C}(\epsilon_2)$ and $\{\hat{x}(t) : t \geq t_\epsilon\}$ is connected and in $\mathcal{L}_f(f(x^\star) - \epsilon_2)$, we necessarily have $\{\hat{x}(t) : t \geq t_\epsilon\} \subset \mathcal{C}(\epsilon_2)$. Assume ϵ, η_0, η_1 are small enough that $\epsilon_2 \leq C_5$. Then, by Lemma 5, $\mathcal{C}(\epsilon_2) \subset \bar{B}(x^\star, \sqrt{2\epsilon_2/\underline{\nu}})$, and so

$$\|\hat{x}(t) - x^\star\| \leq \epsilon_3 := \sqrt{2\epsilon_2/\underline{\nu}}, \quad \text{for all } t \geq t_\epsilon. \tag{43}$$

Assume ϵ, η_0, η_1 are small enough that $\bar{B}(x^\star, \epsilon_3) \subset S$. For any x and y in $\bar{B}(x^\star, \epsilon_3)$, we then have

$$\|H_f(x) - H_f(y)\| \leq d\|H_f(x) - H_f(y)\|_{\max} \leq d^{\frac{3}{2}}\kappa_3\|x - y\|. \tag{44}$$

Using (44), for any x in $\bar{B}(x^\star, \epsilon_3)$

$$\begin{aligned}\|H_{\hat{f}}(x) - H_f(x^\star)\| &\leq \|H_{\hat{f}}(x) - H_f(x)\| + \|H_f(x) - H_f(x^\star)\| \\ &\leq \eta_2 + d^{\frac{3}{2}}\kappa_3\|x - x^\star\| \\ &\leq \eta_2 + d^{\frac{3}{2}}\kappa_3\epsilon_3. \end{aligned} \tag{45}$$

We then apply Weyl's inequality (Stewart and Sun, 1990, Cor. IV.4.9) to conclude that, when η_2 and ϵ_3 are small enough, for all x in $\bar{B}(x^\star, \epsilon_3)$, the eigenvalues of $H_{\hat{f}}(x)$ are all in $(-\infty, -\underline{\nu})$. We assume that $\epsilon, \eta_0, \eta_1, \eta_2$ are small enough that this is the case. This implies that any critical point of $\hat{f}$ in $\bar{B}(x^\star, \epsilon_3)$ is isolated and a local maximum of $\hat{f}$. Using (43) and the compactness of $\bar{B}(x^\star, \epsilon_3)$, by Cantor's intersection theorem $K := \cap_{t \geq t_\epsilon} \overline{\{\hat{x}(u) : u \geq t\}}$ is nonempty. In addition, K is composed of critical points of $\hat{f}$; see Hirsch et al. (2004, Section 9.3, Proposition, p. 206 and Theorem, p. 205) or Absil and Kurdyka (2006, Lemma 5). Therefore we conclude that K is a singleton, which we denote by $\hat{x}^\star$. This is a critical point of $\hat{f}$ in $\bar{B}(x^\star, \epsilon_3)$ and is the limit of $\hat{x}(t)$ as $t \to \infty$. Moreover, $\hat{x}^\star$ is a local maximum of $\hat{f}$.

Since our assumptions imply that $x^\star$ is also a local maximum, we can apply Lemma 8 to bound $\|\hat{x}^\star - x^\star\|$. In our setting, applying the triangle inequality, we may take $C_8 = \max\left\{1, \frac{2}{\sqrt{\underline{\nu}}}, \frac{4\kappa}{3\underline{\nu}}\right\}$, where $\kappa = \kappa_3 + \eta_3$. Assume ϵ, η_0, η_1 are small enough that $\epsilon_3 \leq 1/C_8$. Then, by (43) and Lemma 8, we conclude that $\|\hat{x}^\star - x^\star\| \leq C_8\sqrt{2\eta_0}$. Hence we have shown that there exists a constant $Q_0 := Q_0(f, \underline{\nu}) \geq 1$ such that, whenever $\max\{\eta_0, \eta_1, \eta_2\} \leq 1/Q_0$ and $\eta_3 \leq Q_0$,

$$\|\hat{x}^\star - x^\star\| \leq Q_0\sqrt{\eta_0}. \tag{46}$$

Let $\mathbf{H}$ and $\hat{\mathbf{H}}$ be short for $H_f(x^\star)$ and $H_{\hat{f}}(\hat{x}^\star)$, respectively. We now bound $\|\hat{x}(t) - x(t)\|$ in two ways.

Bound for large t. We proceed with a linearization of the flows near the critical points. Let $\nu > \underline{\nu}$, but close enough that all the eigenvalues of $\mathbf{H}$ are still in $(-\infty, -\nu)$. Note first that $x^\star$ is an interior point of S. Suppose that $\max\{\eta_0, \eta_1, \eta_2\} \leq 1/Q_0$ and $\eta_3 \leq Q_0$ so that (46) holds. By combining (45) and (46)

$$\|\hat{\mathbf{H}} - \mathbf{H}\| \leq \eta_2 + d^{\frac{3}{2}}\kappa_3 Q_0\sqrt{\eta_0}. \tag{47}$$

Suppose in addition that η_0 is small enough that $\hat{x}^\star$ is also an interior point to S, which is possible by (46), and that $\|\hat{\mathbf{H}} - \mathbf{H}\|$ is small enough that $\hat{\mathbf{H}}$ also has all its eigenvalues in $(-\infty, -\nu)$, which is possible by (47) and Weyl's inequality for η_0 and η_2 small enough. Then there exists $r_\ddagger > 0$ such that

$$\bar{B}(x^\star, r_\ddagger) \subset S \quad \text{and} \quad \bar{B}(\hat{x}^\star, r_\ddagger) \subset S,$$

and since $x(t) \to x^\star$ and $\hat{x}(t) \to \hat{x}^\star$ as $t \to \infty$, there exists a time $t_\ddagger > 0$ such that

$$x(t) \in \bar{B}(x^\star, r_\ddagger) \subset S \quad \text{and} \quad \hat{x}(t) \in \bar{B}(\hat{x}^\star, r_\ddagger), \quad \text{for any } t \geq t_\ddagger.$$

Letting $x_\ddagger(t) = x(t) - x^\star$ and $\hat{x}_\ddagger(t) = \hat{x}(t) - \hat{x}^\star$, by a Taylor expansion, for all $t \geq t_\ddagger$ we have

$$x'_\ddagger(t) = \nabla f(x(t)) = \mathbf{H}\, x_\ddagger(t) + R(t), \qquad \text{with} \quad \|R(t)\| \leq \tfrac{\sqrt{d}\kappa_3}{2}\|x_\ddagger(t)\|^2\,; \tag{48}$$

$$\hat{x}'_\ddagger(t) = \nabla \hat{f}(\hat{x}(t)) = \hat{\mathbf{H}}\,\hat{x}_\ddagger(t) + \hat{R}(t), \qquad \text{with} \quad \|\hat{R}(t)\| \leq \tfrac{\sqrt{d}(\kappa_3+\eta_3)}{2}\|\hat{x}_\ddagger(t)\|^2\,. \tag{49}$$

The difference gives

$$\begin{aligned} x'_\ddagger(t) - \hat{x}'_\ddagger(t) &= \mathbf{H}x_\ddagger(t) - \hat{\mathbf{H}}\hat{x}_\ddagger(t) + R(t) - \hat{R}(t) \\ &= \mathbf{H}(x_\ddagger(t) - \hat{x}_\ddagger(t)) + (\mathbf{H} - \hat{\mathbf{H}})\hat{x}_\ddagger(t) + R(t) - \hat{R}(t), \end{aligned} \tag{50}$$

and after integration between 0 and $t > 0$, we get

$$x_\ddagger(t) - \hat{x}_\ddagger(t) = -e^{t\mathbf{H}}(x^\star - \hat{x}^\star) + \int_0^t e^{(t-s)\mathbf{H}}\big[(\mathbf{H} - \hat{\mathbf{H}})\hat{x}_\ddagger(s) + R(s) - \hat{R}(s)\big]\mathrm{d}s. \tag{51}$$

To check that, note that $x_\ddagger(0) - \hat{x}_\ddagger(0) = x^\star - \hat{x}^\star$, and by differentiating (51), we get

$$\begin{aligned} x'_\ddagger(t) - \hat{x}'_\ddagger(t) = &-\mathbf{H}e^{t\mathbf{H}}(x^\star - \hat{x}^\star) + \mathbf{H}e^{t\mathbf{H}}\int_0^t e^{-s\mathbf{H}}\big[(\mathbf{H} - \hat{\mathbf{H}})\hat{x}_\ddagger(s) + R(s) - \hat{R}(s)\big]\mathrm{d}s \\ &+ (\mathbf{H} - \hat{\mathbf{H}})\hat{x}_\ddagger(t) + R(t) - \hat{R}(t). \end{aligned} \tag{52}$$

From (51), $e^{t\mathbf{H}}(x^\star - \hat{x}^\star)$ may be expressed as

$$e^{t\mathbf{H}}(x^\star - \hat{x}^\star) = -(x_\ddagger(t) - \hat{x}_\ddagger(t)) + \int_0^t e^{(t-s)\mathbf{H}}\big[(\mathbf{H} - \hat{\mathbf{H}})\hat{x}_\ddagger(s) + R(s) - \hat{R}(s)\big]\mathrm{d}s. \tag{53}$$

By reporting (53) in (52) we indeed obtain (50).

Using the triangle inequality in (51), and the fact that all the eigenvalues of $\mathbf{H}$ and $\hat{\mathbf{H}}$ are in $(-\infty, -\nu)$ we then get by (48) and (49) that

$$\begin{aligned} \|x_\ddagger(t) - \hat{x}_\ddagger(t)\| \leq\ & e^{-\nu t}\|x^\star - \hat{x}^\star\| \\ &+ \sqrt{d}\int_0^t e^{-\nu(t-s)}\left[\eta_2\|\hat{x}_\ddagger(s)\| + \tfrac{\kappa_3}{2}\|x_\ddagger(s)\|^2 + \tfrac{\kappa_3+\eta_3}{2}\|\hat{x}_\ddagger(s)\|^2\right]\mathrm{d}s. \end{aligned}$$

By Lemma 6, $\max(\|x_\ddagger(t)\|, \|\hat{x}_\ddagger(t)\|) \leq C_6 e^{-\nu t}$ for all $t \geq 0$. We use this to bound the integral above. We have

$$\begin{aligned} &\int_0^t e^{-\nu(t-s)}\left[\eta_2\|\hat{x}_\ddagger(s)\| + \tfrac{\kappa_3}{2}\|x_\ddagger(s)\|^2 + \tfrac{\kappa_3+\eta_3}{2}\|\hat{x}_\ddagger(s)\|^2\right]\mathrm{d}s \\ &\quad\leq \int_0^t e^{-\nu(t-s)}\left[\eta_2 C_6 e^{-\nu s} + \tfrac{\kappa_3}{2}C_6^2 e^{-2\nu s} + \tfrac{\kappa_3+\eta_3}{2}C_6^2 e^{-2\nu s}\right]\mathrm{d}s \\ &\quad\leq C_6 e^{-\nu t}\left[\eta_2 t + (\kappa_3 + \eta_3)C_6\frac{1 - e^{-\nu t}}{\nu}\right]. \end{aligned}$$

Hence

$$\|x_\ddagger(t) - \hat{x}_\ddagger(t)\| \leq e^{-\nu t}\|x^\star - \hat{x}^\star\| + \sqrt{d}C_6 e^{-\nu t}\left[\eta_2 t + (\kappa_3 + \eta_3)C_6\frac{1 - e^{-\nu t}}{\nu}\right]. \tag{54}$$

By the triangle inequality, $\|x(t)-\hat{x}(t)\| \le \|x^\star - \hat{x}^\star\| + \|x_\ddagger(t) - \hat{x}_\ddagger(t)\|$, and using (46) and (54), we deduce that

$$\|x(t)-\hat{x}(t)\| \le (1+e^{-\nu t})Q_0\sqrt{\eta_0} + \sqrt{d}C_6 e^{-\nu t}\left[\eta_2 t + (\kappa_3+\eta_3)C_6\frac{1-e^{-\nu t}}{\nu}\right], \quad \text{for all } t \ge t_\ddagger.$$

By increasing the constant factors as needed, we arrive at

$$\|x(t)-\hat{x}(t)\| \le Q_1\big(\sqrt{\eta_0} + e^{-\nu t}\left[\eta_2 t + \kappa_3 + \eta_3\right]\big), \quad \text{for all } t \ge 0, \tag{55}$$

for some constant $Q_1 > 0$.

Bound for small t. We also have the following refinement of (41). Since f is C^3, there exists $\epsilon > 0$ such that all the eigenvalues of $H_f(x)$ exceed $-\overline{\nu}$ when $x \in \bar{B}(x^\star, \epsilon)$. Note that this implies that ∇f is Lipschitz on $\bar{B}(x^\star,\epsilon)$ with constant $\overline{\nu}$.

Keeping $\epsilon > 0$ fixed, let t_ϵ be such that $x(t) \in \bar{B}(x^\star,\epsilon)$ and $\hat{x}(t) \in \bar{B}(\hat{x}^\star, \epsilon/2)$, for all $t \ge t_\epsilon$. Assume that η_0 is small enough that $\|\hat{x}^\star - x^\star\| \le \epsilon/2$, which is possible by (46). Then we also have $\hat{x}(t) \in \bar{B}(x^\star,\epsilon)$

We may now apply Lemma 7 to get

$$\|x(t)-\hat{x}(t)\| \le \frac{\eta_1}{\overline{\nu}}e^{\overline{\nu}t}, \quad \forall t \ge t_\epsilon. \tag{56}$$

Since ϵ is fixed, by (41), for any $0 \le t \le t_\epsilon$, we have

$$\|x(t)-\hat{x}(t)\| \le \frac{\eta_1}{\sqrt{d}\kappa_2}e^{\sqrt{d}\kappa_2 t} \le \frac{e^{|\sqrt{d}\kappa_2 - \overline{\nu}|t_\epsilon}}{\sqrt{d}\kappa_2}\eta_1 e^{\overline{\nu}t}. \tag{57}$$

Combining (56) and (57) we deduce that

$$\|x(t)-\hat{x}(t)\| \le Q_2\eta_1 e^{\overline{\nu}t}, \quad \forall t \ge 0, \tag{58}$$

for some constant Q_2.

We now combine (55) and (58), and use the fact that $te^{-\nu t} \le \frac{1}{\nu-\underline{\nu}}e^{-\underline{\nu}t}$ for all $t \ge 0$, to arrive at

$$\|x(t)-\hat{x}(t)\| \le Q_3 \min\left[\sqrt{\eta_0} + e^{-\underline{\nu}t}, \eta_1 e^{\overline{\nu}t}\right], \quad \forall t \ge 0, \tag{59}$$

for some constant Q_3. We shall show that the bound (15) follows from (59). To verify this, we start with

$$\min\left[\sqrt{\eta_0} + e^{-\underline{\nu}t}, \eta_1 e^{\overline{\nu}t}\right] \le 2B(t), \quad B(t) := \min\left[\max\{\sqrt{\eta_0}, e^{-\underline{\nu}t}\}, \eta_1 e^{\overline{\nu}t}\right].$$

Set $t_0 = \frac{1}{2\underline{\nu}}\log(1/\eta_0)$ and note that

$$\max\{\sqrt{\eta_0}, e^{-\underline{\nu}t}\} = \begin{cases} e^{-\underline{\nu}t} & \text{when } t \le t_0 \\ \sqrt{\eta_0} & \text{when } t \ge t_0. \end{cases}$$

- When $t \ge t_0$, then we simply observe that $B(t) \le \eta_0^{1/2}$.

- When $t \le t_0$, we have $B(t) = \min\left\{e^{-\underline{\nu}t}, \eta_1 e^{\overline{\nu}t}\right\}$. Let $t_1 = \frac{1}{\underline{\nu}+\overline{\nu}}\log(1/\eta_1)$. Note that the map defined on $[0,\infty)$ by $t \mapsto \min\left\{e^{-\underline{\nu}t}, \eta_1 e^{\overline{\nu}t}\right\}$ is increasing over $[0,t_1]$, decreasing over $[t_1,\infty)$, and that

$$\min\left\{e^{-\underline{\nu}t}, \eta_1 e^{\overline{\nu}t}\right\} = \begin{cases} \eta_1 e^{\overline{\nu}t} & \text{when } t \le t_1 \\ e^{-\underline{\nu}t} & \text{when } t \ge t_1. \end{cases}$$

  - When $t_1 \ge t_0$, $B(t) \le B(t_0) = \eta_1 e^{\overline{\nu}t_0} = \eta_1 \eta_0^{-\frac{\overline{\nu}}{2\underline{\nu}}}$.
  - When $t_1 < t_0$, then $B(t) \le B(t_1) = e^{-\underline{\nu}t_1} = \eta_1^{\frac{\underline{\nu}}{\underline{\nu}+\overline{\nu}}}$

  Since $t_0 \le t_1$ if, and only if, $\eta_1\eta_0^{-\frac{\overline{\nu}}{2\underline{\nu}}} \le \eta_1^{\frac{\underline{\nu}}{\underline{\nu}+\overline{\nu}}}$, we conclude that $B(t) \le \min\left\{\eta_1^{\frac{\underline{\nu}}{\underline{\nu}+\overline{\nu}}}, \eta_1\eta_0^{-\frac{\overline{\nu}}{2\underline{\nu}}}\right\}$ for all $t \le t_0$.

Hence, we worked (59) into

$$\sup_{t\ge 0}\|x(t)-\hat{x}(t)\| \le 2Q_3 \max\left\{\sqrt{\eta_0}, \min\left[\eta_1^\delta, \eta_0^{\frac{\delta-1}{2\delta}}\eta_1\right]\right\},$$

where $\delta = \frac{\underline{\nu}}{\underline{\nu}+\overline{\nu}}$. We note that

$$\sqrt{\eta_0} \le \eta_1^\delta \iff \eta_0^{\frac{1}{2\delta}} \le \eta_1 \iff \sqrt{\eta_0} \le \eta_1\eta_0^{\frac{1}{2}-\frac{1}{2\delta}} \iff \sqrt{\eta_0} \le \eta_0^{\frac{\delta-1}{2\delta}}\eta_1$$

and that

$$\eta_1^\delta \le \eta_0^{\frac{\delta-1}{2\delta}}\eta_1 \iff \eta_0^{\frac{1-\delta}{2\delta}} \le \eta_1^{1-\delta} \iff \sqrt{\eta_0} \le \eta_1^\delta.$$

Using these equivalences we deduce that

$$\max\left\{\sqrt{\eta_0}, \min\left[\eta_1^\delta, \eta_0^{\frac{\delta-1}{2\delta}}\eta_1\right]\right\} = \max\left\{\sqrt{\eta_0}, \eta_1^\delta\right\}.$$

4.4 Proof of Lemma 2

For any d-tuple $\beta = (\beta_1, \dots, \beta_d) \in \mathbb{N}^d$, let $|\beta| = \beta_1 + \cdots + \beta_d$, and let

$$\partial^\beta g(x) = \frac{\partial^{|\beta|}}{\partial x_1^{\beta_1}\dots\partial x_d^{\beta_d}}g(x) \tag{60}$$

denote the β-th partial derivative of a function $g : \mathbb{R}^d \to \mathbb{R}$. Let C be such that $|\partial^\beta f(x)| \le C$ for all $x \in \mathbb{R}^d$ and all β such that $|\beta| \le 3$.

Fix $\beta \in \mathbb{N}^d$ with $|\beta| = \ell \le 3$. Since the partial derivatives of Φ up to the order 3 vanish at infinity, and those of f are bounded, we obtain by integrating by parts

$$\begin{aligned}
\mathbb{E}\big[\partial^\beta \hat{f}(x)\big] &= \frac{1}{h^{d+\ell}}\mathbb{E}\left[\partial^\beta\Phi\left(\frac{x-X}{h}\right)\right] \\
&= \frac{1}{h^d}\int_{\mathbb{R}^d}\Phi\left(\frac{x-u}{h}\right)\partial^\beta f(u)\mathrm{d}u \\
&= \int_{\mathbb{R}^d}\Phi(u)\,\partial^\beta f(x-hu)\mathrm{d}u.
\end{aligned}$$

When $\ell = 3$, we simply deduce that

$$\left|\mathbb{E}\big[\partial^\beta \hat{f}(x)\big] - \partial^\beta f(x)\right| \le \left|\mathbb{E}\big[\partial^\beta \hat{f}(x)\big]\right| + C \le 2C,$$

using Jensen's inequality.

When $\ell = 2$, we use a Taylor expansion of order 1, to get

$$\left|\partial^\beta f(x - hu) - \partial^\beta f(x)\right| \le \sqrt{d}Ch\|u\|, \quad \forall x, u \in \mathbb{R}^d,$$

and deduce that

$$\left|\mathbb{E}\big[\partial^\beta \hat{f}(x)\big] - \partial^\beta f(x)\right| \le h\sqrt{d}C \int_{\mathbb{R}^d} \|u\|\Phi(u)\mathrm{d}u,$$

using the fact that Φ integrates to 1.

When $\ell \le 1$, we use a Taylor expansion of order 2, to get

$$\left|\partial^\beta f(x - hu) - \partial^\beta f(x) + h\big(\partial^\beta f\big)^{(1)}(x)[u]\right| \le dCh^2\|u\|^2, \quad \forall x, u \in \mathbb{R}^d,$$

and deduce that

$$\left|\mathbb{E}\big[\partial^\beta \hat{f}(x)\big] - \partial^\beta f(x)\right| \le h^2\, dC \int_{\mathbb{R}^d} \|u\|^2\Phi(u)\mathrm{d}u,$$

using the fact that Φ integrates to 1 and kills moments of order 1 by assumption (18).

4.5 Proof of Lemma 3

From Theorem 4.1 in Mason (2012), we immediately deduce the following. (Note that in the statement of condition (G.iii) of Theorem 4.1 in Mason (2012), $\mathcal{G}$ should be corrected to be $\mathcal{G}_0$).

Lemma 9 *Let f be a density on $\mathbb{R}^d$ and let $X \sim f$. Let $\mathcal{G}$ be a class of uniformly bounded measurable functions $\mathbb{R}^d \times (0,1] \to \mathbb{R}$, such that*

$$\sup_{g\in\mathcal{G}} \sup_{h\in(0,1]} \frac{1}{h^d}\mathbb{E}\left[g(X,h)^2\right] < \infty, \tag{61}$$

and such that the class

$$\mathcal{G}_0 = \{x \mapsto g(x,h) : g \in \mathcal{G}, h \in (0,1)\} \tag{62}$$

is pointwise measurable and of VC-type. Then there exists a $0 < b_0 < 1$ such that if $X_1, X_2, \ldots$ is an iid sequence from f,

$$\limsup_{n\to\infty} \sup_{g\in\mathcal{G}} \sup_{\frac{\log n}{n} \le h^d \le b_0} \sqrt{\frac{n}{h^d \log n}}\left|\frac{1}{n}\sum_{i=1}^n g(X_i,h) - \mathbb{E}[g(X,h)]\right| < \infty, \quad \textit{almost surely.} \tag{63}$$

For the definitions of VC-type and pointwise measurable, we refer to Mason (2012, Sec. 4.2) or van der Vaart and Wellner (1996).

Remark 1 *The assumption that the class $\mathcal{G}_0$ be pointwise measurable insures that the supremum of functionals defined on $\mathcal{G}_0$ be measurable. Another condition that is often imposed on a class of functions is image-Suslin measurable. For details see page 138 of de la Peña and Giné (1999).*

Let Φ be a kernel and f be a density as in Lemma 3, and let $X \sim f$. Fixing $\beta \in \mathbb{N}^d$ such that $|\beta| \le 3$, we apply this lemma to

$$\mathcal{G} = \big\{(x,h) \mapsto \partial^\beta\Phi(\tfrac{u-x}{h}) : u \in \mathbb{R}^d\big\}.$$

For any $x, u \in \mathbb{R}^d$ and $h \in (0,1]$,

$$\left|\partial^\beta\Phi(\tfrac{u-x}{h})\right| \le \|\partial^\beta\Phi\|_\infty,$$

so that $\mathcal{G}$ is uniformly bounded, and

$$\mathbb{E}\left[\partial^\beta\Phi\left(\frac{u-X}{h}\right)^2\right] = \int_{\mathbb{R}^d} \partial^\beta\Phi\left(\frac{u-x}{h}\right)^2 f(x)\mathrm{d}x,$$

which by the change of variables $v = \frac{u-x}{h}$ equals

$$h^d \int_{\mathbb{R}^d} \partial^\beta\Phi(v)^2 f(u-hv)\mathrm{d}v \le h^d\|f\|_\infty\|\partial^\beta\Phi\|_\infty \int_{\mathbb{R}^d}\left|\partial^\beta\Phi(v)\right|\mathrm{d}v, \tag{64}$$

where $\|f\|_\infty$, $\|\partial^\beta\Phi\|_\infty$, and $\int_{\mathbb{R}^d}\left|\partial^\beta\Phi(v)\right|\mathrm{d}v$ are finite by assumption. Hence $\mathcal{G}$ satisfies (61). In addition, $\mathcal{G}_0$ is seen to be pointwise measurable by consideration of the subclass

$$\big\{x \mapsto \partial^\beta\Phi(\tfrac{u-x}{h}) : u \in \mathbb{Q}^d, h \in (0,1] \cap \mathbb{Q}\big\}.$$

To see that $\mathcal{G}_0$ is of VC-type, notice that for any $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$, $\partial^\beta\Phi(x) = \prod_{k=1}^d \phi_k^{(\beta_k)}(x_k)$. By assumption, $\phi_k^{(\beta_k)}$ is of bounded variation on $\mathbb{R}$, so that by Nolan and Pollard (1987, Lem 22) the class of functions given by

$$\mathrm{g}_{0,k} := \left\{s \in \mathbb{R} \mapsto \phi_k^{(\beta_k)}(\tfrac{s-t}{h}) : s \in \mathbb{R},\, 0 < h \le 1\right\}$$

is of VC-type. Then an application of Einmahl and Mason (2000, Lem A1) shows that the class of functions $\mathcal{G}_0$, which is equivalently expressed as

$$\mathcal{G}_0 := \{(u_1, \ldots, u_d) \mapsto g_1(u_1)\ldots g_d(u_d) : g_k \in \mathrm{g}_{0,k},\, k = 1, \ldots, d\},$$

is of VC-type.

Therefore, the conditions of Lemma 9 are met, so that we can assert that (63) holds. Noticing that

$$\frac{1}{n}\sum_{i=1}^n \partial^\beta\Phi(\tfrac{u-X_i}{h}) = h^{\ell+d}\partial^\beta f_{n,h}(u),$$

and consequently

$$\mathbb{E}\left[\partial^\beta\Phi(\tfrac{u-X}{h})\right] = h^{\ell+d}\,\mathbb{E}\left[\partial^\beta f_{n,h}(u)\right],$$

we see that (63) yields

$$\limsup_{n\to\infty} \sup_{u\in\mathbb{R}^d} \sup_{\frac{\log n}{n} \le h^d \le b_0} \sqrt{\frac{n}{h^d \log n}}h^{\ell+d}\left|\partial^\beta f_{n,h}(u) - \mathbb{E}\left[\partial^\beta f_{n,h}(u)\right]\right| < \infty, \quad \text{almost surely,}$$

which is exactly (20).

4.6 Proof of Theorem 3

As in the proofs of Theorems 1 and 2, we may assume without loss of generality that $\mathcal{L}_f(f(x_0/2)) \subset \bar{B}(x_0, 3r_0)$, with $r_0 = \sup_{t\geq 0} \|x(t) - x_0\|$, which implies that $\mathcal{L}_f(f(x_0/2)$ is compact. In this subsection,

$$S = \mathcal{L}_f(f(x_0)/2), \quad \kappa_\ell = \kappa_\ell(f, S), \quad \hat{f} = \hat{f}_{n,h}, \tag{65}$$

for short.

For any integer $0 \leq \ell \leq 2$, we let

$$\eta_\ell^\star = \sup_{x\in S} \left\| \hat{f}^{(\ell)}(x) - f^\ell(x) \right\|, \quad \eta_\ell = \sup_{x\in S} \left\| \left(\log \hat{f}\right)^{(\ell)}(x) - \left(\log f\right)^{(\ell)}(x) \right\|,$$

where the norm used is defined in (8). (Keep in mind that we are suppressing in the notation $\hat{f}^{(\ell)}$ and η_ℓ the dependence on n and h.) From (19) and (20), we see that, since $\frac{nh^{d+6}}{\log n} \to \infty$, for any $0 \leq \ell \leq 2$, $\eta_\ell^\star \to 0$ almost surely as $n \to \infty$ while $\eta_3^\star = O(1)$ almost surely. Since $f(x) \geq f(x_0)/2 > 0$ for all x in S, and since $\eta_0^\star \to 0$ almost surely, then almost surely, for all n large enough, $\log \hat{f}(x)$ is well-defined for all x in S. We have

$$\frac{\partial}{\partial x_i} \log f(x) = \frac{1}{f} \frac{\partial}{\partial x_i} f(x), \qquad \frac{\partial}{\partial x_i} f^{-k}(x) = -k f^{-(k+1)}(x) \frac{\partial}{\partial x_i} f(x), \tag{66}$$

and similarly for $\hat{f}$ almost surely for all n large enough, using the fact that $f(x) \geq f(x_0)/2$ for all x in S once again. We see using (66) that for each $0 \leq \ell \leq 3$ and $\beta \in \mathbb{N}^d$ with $|\beta| = \ell$ there is a continuously differentiable function $F_{\ell,\beta}$ defined on $(0,\infty) \times \mathbb{R}^d \times \cdots \times \mathbb{R}^{d\ell}$, where $\mathbb{R}^{d\ell}$ is suppressed if $\ell = 0$, such that for all $x \in S$

$$\partial^\beta \log f(x) = F_{\ell,\beta}\left(f(x), \partial^\alpha f(x),\, \alpha \in \mathbb{N}^d \text{ with } |\alpha| = k,\, k = 1, \ldots, \ell \right),$$

where with some abuse of notation, $\partial^\alpha f(x)$, $\alpha \in \mathbb{N}^d$ with $|\alpha| = k$, $k \geq 1$, represents a dk-vector in $\mathbb{R}^{dk}$, and, similarly, almost surely, for all large enough n

$$\partial^\beta \log \hat{f}(x) = F_{\ell,\beta}\left(\hat{f}(x), \partial^\alpha \hat{f}(x),\, \alpha \in \mathbb{N}^d \text{ with } |\alpha| = k,\, k = 1, \ldots, \ell \right).$$

Observe that the set of points

$$\left\{ \left(f(x), \partial^\alpha f(x),\, \alpha \in \mathbb{N}^d \text{ with } |\alpha| = k,\, k = 1, \ldots, \ell \right) : x \in S \right\} \tag{67}$$

lies in a compact subset of $(0,\infty) \times \mathbb{R}^d \times \cdots \times \mathbb{R}^{d\ell}$, and almost surely for all large enough n the same is true for the set of points formed as in (67) with f replaced by $\hat{f}$. Since a compact subset of $(0,\infty) \times \mathbb{R}^d \times \cdots \times \mathbb{R}^{d\ell}$ can be chosen to include both of these sets, using the mean value theorem we see that for some constant $C(\ell, \beta) > 0$

$$\sup_{x\in S} \left| \partial^\beta \log f(x) - \partial^\beta \log \hat{f}(x) \right|$$
$$\leq C(\ell,\beta) \max \left\{ \sup_{x\in S} \left| \partial^\alpha f(x) - \partial^\alpha \hat{f}(x) \right| : \alpha \in \mathbb{N}^d \text{ with } |\alpha| = k,\, k = 0, \ldots, \ell \right\}.$$

Using (10) this proves that there exists a constant $C > 0$ such that almost surely for all n large enough

$$\eta_\ell \leq C(\eta_0^\star + \cdots + \eta_\ell^\star), \quad 0 \leq \ell \leq 3. \tag{68}$$

Hence, almost surely, $\eta_\ell \to 0$ for all $\ell = 0, 1, 2$ and $\limsup \eta_3 < \infty$. We are then in a position to apply Corollary 1. Noting that $\sqrt{\left(\frac{\log n}{nh^{d+2}}\right)} = o(h^2)$ under the condition $\frac{nh^{d+6}}{\log n} \to \infty$, and using the inequalities in (68), almost surely for all n large enough, $\eta_0 \leq Ch^2$ and $\eta_1 \leq Ch^2$ for some constant $C > 1$, and since $\delta < 1/2$, almost surely, for all n large enough, $\max\{\sqrt{\eta_0}, \eta_1^\delta\} \leq Ch^{2\delta}$. We conclude by applying Corollary 1.

Acknowledgments

The authors are grateful to Jacob Sterbenz for pointers and stimulating discussions, and to three anonymous referees for comments and for bringing to their attention some important references that were missing, in particular, (Merlet and Pierre, 2010; Stetter, 1973; Comaniciu and Meer, 2002). EAC was supported by a grant from the US National Science Foundation (DMS-1513465). BP was supported by a grant from the French National Research Agency (ANR 09-BLAN-0051-01).

References

P.-A. Absil and K. Kurdyka. On the stable equilibrium points of gradient systems. *Systems & Control Letters*, 55:573–577, 2006.

W.-J. Beyn. On the numerical approximation of phase portraits near stationary points. *SIAM J. Numer. Anal.*, 24(5):1095–1113, 1987. ISSN 0036-1429. doi: 10.1137/0724072. URL `http://dx.doi.org/10.1137/0724072`.

J. Bolte, A. Daniilidis, O. Ley, and L. Mazet. Characterizations of Łojasiewicz inequalities: subgradient flows, talweg, convexity. *Trans. Amer. Math. Soc.*, 362(6):3319–3363, 2010.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004.

M.A. Carreira-Perpinan. Gaussian mean-shift is an em algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):767–776, 2007.

M.A. Carreira-Perpinan and C.K.I. Williams. On the number of modes of a gaussian mixture. In *Scale Space Methods in Computer Vision*, volume 2695, pages 625–640. Lecture Notes in Computer Science, 2003.

M.-Y. Cheng, P. Hall, and J.A. Hartigan. Estimating gradient trees. In *A festschrift for Herman Rubin*, volume 45 of *IMS Lecture Notes Monogr. Ser.*, pages 237–249. Inst. Math. Statist., Beachwood, OH, 2004.

Y. Cheng. Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):790–799, 1995.

D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):1–18, 2002.

V.H. de la Peña and E. Giné. *Decoupling. From Dependence to Independence. Randomly Stopped Processes. U-statistics and Processes. Martingales and Beyond.* Probability and its Applications (New York). Springer-Verlag, New York, 1999.

L. Devroye and L. Gyorfi. *Nonparametric Density Estimation: The L_1 View.* John Wiley & Sons, New-York, 1985.

U. Einmahl and D.M. Mason. An empirical process approach to the uniform consistency of kernel-type function estimators. *Journal of Theoretical Probability*, 13:1–37, 2000.

U. Einmahl and D.M. Mason. Uniform in bandwidth consistency of kernel-type function estimators. *Annals of Statistics*, 33:1380–1403, 2005.

K. Fukunaga and L. D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21 (1):32–40, 1975.

E. Giné and A. Guillou. Rates of strong uniform consistency for multivariate kernel density estimators. *Annals of the Institute Henri Poincaré: Probability and Statistics*, 38:907–921, 2002.

J.A. Hartigan. *Clustering Algorithms.* Wiley, New York, 1975.

M.W. Hirsch and S. Smale. *Differential Equations, Dynamical Systems, and Linear Algebra.* Academic Press, 1974.

M.W. Hirsch, S. Smale, and R.L. Devaney. *Differential Equations, Dynamical Systems & An Introduction to Chaos.* Academic Press, second edition, 2004.

M. C. Irwin. *Smooth Dynamical Systems.* Academic Press, New York - London, 1980.

J. Li, S. Ray, and B.G. Lindsay. A nonparametric statistical approach to clustering via mode identification. *Journal of Machine Learning Research*, 8:1687–1723, 2007.

D.M. Mason. Proving consistency of non-standard kernel estimators. *Stochastic Inference for Stochastic Processes*, 15:151–176, 2012.

D.M. Mason and J. Swanepoel. A general result on the uniform in bandwidth consistency of kernel-type function estimators. *Test*, 20:72–94, 2011.

B. Merlet and M. Pierre. Convergence to equilibrium for the backward Euler scheme and applications. *Commun. Pure Appl. Anal.*, 9(3):685–702, 2010.

D. Nolan and D. Pollard. U-processes: rates of convergence. *Annals of Statistics*, 15: 780–799, 1987.

H. J. Stetter. *Analysis of Discretization Methods for Ordinary Differential Equations.* Springer-Verlag, New York-Heidelberg, 1973. Springer Tracts in Natural Philosophy, Vol. 23.

G.W. Stewart and J.G. Sun. *Matrix Perturbation Theory.* Computer Science and Scientific Computing. Academic Press Inc., Boston, MA, 1990.

G. Teschl. *Ordinary Differential Equations and Dynamical Systems*, volume 140. American Mathematical Soc., 2012.

A.W. van der Vaart and J.A. Wellner. *Weak Convergence and Empirical Processes: With Applications to Statistics.* Springer Series in Statistics. Springer-Verlag, New-York, 1996.

Journal of Machine Learning Research 17 (2016) 1-35 Submitted 7/13; Revised 5/15; Published 4/16

Scalable Learning of Bayesian Network Classifiers

Ana M. Martínez ANAM.MARTINEZF@GMAIL.COM
Geoffrey I. Webb GEOFF.WEBB@MONASH.EDU
Faculty of Information Technology
Monash University
VIC 3800, Australia

Shenglei Chen TRISTAN_CHEN@126.COM
College of Information Science/Faculty of Information Technology
Nanjing Audit University/Monash University
China/Australia

Nayyar A. Zaidi NAYYAR.ZAIDI@MONASH.EDU
Faculty of Information Technology
Monash University
VIC 3800, Australia

Editor: Russ Greiner

Abstract

Ever increasing data quantity makes ever more urgent the need for highly scalable learners that have good classification performance. Therefore, an out-of-core learner with excellent time and space complexity, along with high expressivity (that is, capacity to learn very complex multivariate probability distributions) is extremely desirable. This paper presents such a learner. We propose an extension to the k-dependence Bayesian classifier (KDB) that discriminatively selects a sub-model of a full KDB classifier. It requires only one additional pass through the training data, making it a three-pass learner. Our extensive experimental evaluation on 16 large data sets reveals that this out-of-core algorithm achieves competitive classification performance, and substantially better training and classification time than state-of-the-art in-core learners such as random forest and linear and non-linear logistic regression.

Keywords: scalable Bayesian classification, feature selection, out-of-core learning, big data

1. Introduction

Until very recently most machine learning research has been conducted in the context of relatively small datasets, that is, no more than 50,000 instances and a few hundred attributes. It is only in the last five years that larger datasets have started to appear in the literature (Erhan et al., 2010; Sonnenburg and Franc, 2010; Agarwal et al., 2014; Sariyar et al., 2011) on a regular basis. We argue that larger data quantities do not simply call for scaling-up of existing learning algorithms. Rather, we believe, as first argued by Brain and Webb (1999), that the most accurate learners for large data will have much lower bias than the most accurate learners for small data. Thus we need a new generation of computationally efficient low bias learners. To achieve the necessary efficiency, a learning algorithm for very large data should ideally require only a few passes through the training data. Further, to remove memory size as a bottleneck, the algorithm should be able to process data out-of-core.

In this paper, we extend the KDB classifier. KDB is a form of restricted Bayesian network classifier (BNC). It has numerous desirable properties in the context of learning from large quantities of data. These include:

- the capacity to control its bias/variance trade-off with a single parameter, k,
- it does not require data to be held in-core,
- the capacity to learn in just two passes through the training data.

The contributions of this paper are as follows:

- We extend KDB to perform discriminative model selection in a single additional pass through the training data. In this single pass, our algorithm selects between both attribute subsets and network structures. The resulting highly scalable algorithm combines the computational efficiency of classical generative learning with the low bias of discriminative learning.
- We compare the performance of our algorithm with other state-of-the-art classifiers on several large datasets, ranging in size from 165 thousand to 54 million instances and 5 to 784 attributes. We show that our highly scalable algorithms achieve comparable or lower error on large data than a range of out-of-core and in-core state-of-the-art classification learning algorithms.

To illustrate our motivations, in Figure 1 we present learning curves for the poker-hand dataset (which is described in Table 7 in Appendix A). As can be seen, for lower quantities of data the lower variance delivered by low values of k results in lower error for KBD, while for larger quantities of data the lower bias of higher values of k results in lower error. While selective KDB sometimes selects a k that overfits this data with smaller training set sizes, when the training set size exceeds 550,000 it successfully selects the best k and by selecting a subset of the attributes it can substantially reduce error across the entire range of training set sizes relative to the KDB classifier using the same k.

Section 2 reviews the state-of-the-art in out-of-core BNCs and introduces our proposal: selective KDB (SKDB, Section 2.1). In Section 3 we present a set of comparisons for our proposed algorithm on 16 big datasets with *out-of-core* BNCs (Section 3.1), *out-of-core* linear classifiers using Stochastic Gradient Descent (Section 3.2), and with *in-core* (batch) learners BayesNet and Random Forest (Section 3.3). To finalize, Section 4 shows the main conclusions and outlines future work.

2. Scalable Bayesian Networks for Classification

We define the classification learning task as the assignment of a label $y \in \Omega_Y$, from a set of c labels of the class variable Y, to a given example $\mathbf{x} = (x_1, \ldots, x_d)$, with values for the d attributes $\mathcal{A} = \{X_1, \ldots, X_d\}$. Bayesian networks (BN) (Pearl, 1988) provide a framework for computing the joint probability of these d random variables. A BN is

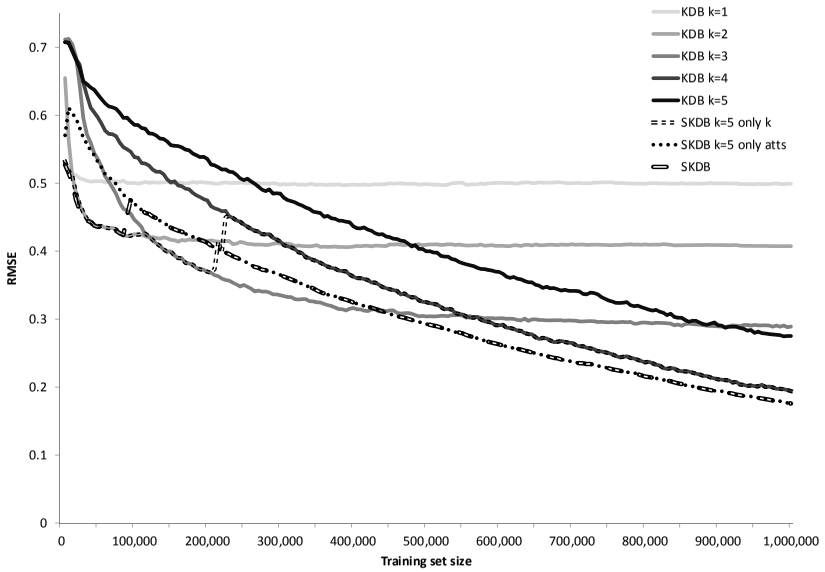


Figure 1: Learning curves for KDB and Selective KDB on the poker-hand dataset. *The values plotted are averages over 10 runs. For each run 1,000 examples are selected as a test set and samples of successive sizes are selected from the remaining data for training. As can be seen, for lower quantities of data the lower variance delivered by low values of k results in lower error for KBD, while for larger quantities of data the lower bias of higher values of k results in lower error, although there is not sufficient data for $k = 5$ to asymptote on this dataset. While only selecting a k (SKDB k=5 only k) causes selective KDB to overfit this data with smaller training set sizes (between 20,000 and 550,000), when the training set size is large enough it successfully selects the best k. Only selecting attributes (SKDB k=5 only atts) always reduces error (relative to KDB k=5). By selecting both k and attributes (SKDB k=5) selective KDB substantially reduces error relative to KDB with any value of k once there are more than approximately 450,000 training examples.*

a directed acyclic graph where the the joint probability of all attributes can be written as the product of the individual probabilities of all attributes given their parents, that is $p(y, \mathbf{x}) = p(y|\pi_y) \prod_{i=1}^{d} p(x_i|\pi_{x_i})$, where π_{x_i} denotes the parents of attribute X_i. If the structure is known, the likelihood of the BN given the data can be maximized by estimating $p(x_i|\pi_{x_i})$ using the empirical estimates from the training data $\mathcal{T}$, composed of t training instances.

While unrestricted BNs are the least biased, training such a model on even moderate size data sets can be extremely challenging, as the search-space that needs to be explored grows exponentially with the number of attributes. This has led to restricted BNCs, including naive Bayes (NB) (Duda and Hart, 1973; Minsky, 1961; Lewis, 1998), tree-augmented naive Bayes (TAN) (Friedman et al., 1997), averaged n-dependence estimators (ANDE) (Webb et al., 2012) and k-dependence estimators (KDB) (Sahami, 1996). These classifiers have in common either no structural learning or minimal structural learning that requires only one or two passes through the data. They have been referred to as semi-naive BNCs (Zheng and Webb, 2005). Even though highly biased due to their inherent conditional attribute independence assumption, they have been shown to be very effective classification methods.

NB is the simplest of the BNCs, assuming that all attributes are independent given the class. It estimates the joint probability using $\hat{p}(y) \prod_{i=1}^{d} \hat{p}(x_i|y)$, where $\hat{p}(\cdot)$ denotes an estimate of $p(\cdot)$. TAN is a structural augmentation of NB where every attribute has as parents the class and at most one other attribute. The structure is determined by using an extension of the Chow-Liu tree (Chow and Liu, 1968), that utilizes conditional mutual information to find a maximum spanning tree. This alleviates some of NB's independence assumption and therefore reduces its bias at the expense of increasing its variance. This results in better performance on larger data sets. TAN, provides an intermediate bias-variance trade-off, standing between NB on one hand and unrestricted BNCs on the other.

There has been a lot of prior work that has explored approaches to alleviate NB's independence assumption. One approach is to add a bounded number of additional interdependencies (Friedman et al., 1997; Zheng and Webb, 2000; Webb et al., 2005; Zhang et al., 2005; Yang et al., 2007; Flores et al., 2009a,b; Zheng et al., 2012; Zaidi et al., 2013). At the other extreme, Su and Zhang (2006) propose the use of a full Bayesian network. Of these algorithms, only two approaches allow the number of interdependencies to be adjusted to best accommodate the best trade-offs between bias and variance for differing data quantities. ANDE (Webb et al., 2012) averages many BNCs, where a parameter n controls the number of interdependencies to be modeled. KDB uses a parameter k to control the number of interdependencies modeled.

KDB relaxes NB's independence assumption by allowing every attribute to be conditioned on the class and, at most, k other attributes. Like TAN, KDB requires two passes over the training data for learning: the first pass collects the statistics to perform calculations of mutual information for structure learning, and the second performs the parameter learning based on the structure created in the former step (Algorithm 1). An advantage over the ANDE model is that KDB does not need to collect all statistics for all combinations of n or k attributes, allowing it to scale to higher order dependencies. It also has the capacity to select a model and hence more closely fit the data. This is a disadvantage for small data, which it may overfit, but KDB will often lead to higher accuracy than ANDE on large data which are more difficult to overfit.

Recently, some techniques have been proposed that address the classification problem with Bayesian networks from a relatively efficient discriminative perspective. Carvalho et al. (2011) propose a decomposable approximation to the conditional log-likelihood scoring criteria. Pernkopf and Bilmes (2010) propose a BNC learning algorithm with some resemblance to KDB but using discriminative evaluation for parent assignment. Neither

Algorithm 1: The KDB algorithm

Input: Training set $\mathcal{T}$ with attributes $\{X_1, \ldots, X_a, Y\}$ and k.
Output: KDB model.

Let $\mathcal{G}$ be a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of links.
Let Θ be a Bayesian network with structure $\mathcal{G}$.
First pass begin
  $\mathcal{G}$ =**learnStructure**($\mathcal{T}$) ; `/* Algorithm 2 */`
end
Second pass begin
  Θ = **learnParameters**($\mathcal{T}$, $\mathcal{G}$) ; `/* Algorithm 3 */`
end
Let KDB be a BN with structure $\mathcal{G}$ and conditional probability distributions in Θ.
return *KDB*

Algorithm 2: learnStructure($\mathcal{T}$)

Input: Training set $\mathcal{T}$
Output: $\mathcal{G}$, network structure.

Calculate $MI(X_i; Y)$ from $\mathcal{T}$ for all attributes.
Calculate $MI(X_i; X_j|Y)$ from $\mathcal{T}$ for each pair of attributes $(i \neq j)$.
Let $\mathcal{L}$ be a list of all X_i in decreasing order of $MI(X_i; Y)$.
$\mathcal{V} = \{Y\}$; $\mathcal{E} = \emptyset$;
for $i = 1 \rightarrow \mathcal{L}.size$ **do**
  $\mathcal{V} = \mathcal{V} \cup \mathcal{L}_i$;
  $\mathcal{E} = \mathcal{E} \cup (Y, \mathcal{L}_i)$;
  $vk = min(i - 1, k)$;
  while *($vk > 0$)* **do**
    $m = \text{argmax}_j\{MI(\mathcal{L}_i; \mathcal{L}_j|Y) \mid 1 \leq j < i \wedge (\mathcal{L}_j, \mathcal{L}_i) \notin \mathcal{E}\}$;
    $\mathcal{E} = \mathcal{E} \cup (\mathcal{L}_j, \mathcal{L}_i)$;
    $vk = vk - 1$;
  end
end
return $\mathcal{G}$

Algorithm 3: learnParameters($\mathcal{T}$, $\mathcal{G}$)

Input: Training set $\mathcal{T}$ and $\mathcal{G}$.
Output: Θ.

Initialize Θ to structure $\mathcal{G}$.
Compute the CPTs for Θ from $\mathcal{T}$.
return Θ

algorithm is suited to learning a BNC in a limited number of passes through the training data.

There have also been some important refinements that improve KDB's performance. Bouckaert (2006) proposes averaging all possible network structures for a fixed value of k (including lower orders). Time complexity is significantly reduced by multiplying the sum of every attribute given all possible parents in the order. For $k = 2$ the authors propose a simplification that identifies one attribute as super parent and consider for every attribute of the class, possibly the super parent and possibly one other lower ordered attribute. The sum of class probabilities is taken as selecting criteria for the superparent. However they agree that this approach is not easy to implement incrementally for large datasets. Rubio and Gámez (2011) present a variant of KDB that employs a hill-climbing search to incrementally build a KDB classifier. This approach requires at least one pass through the data for each attribute.

2.1 Selective KDB

The parameter k controls KDB's bias-variance trade-off. Higher k results in higher variance and lower bias. Unfortunately there is no apriori means to preselect a value of k for which this trade-off will result in the lowest error for a given training set as this is a complex interplay between the data quantity and the complexity and strength of the interactions between the attributes. A further factor that can increase KDB's error is that attributes that carry no useful information about the class must be treated as if they do, which invariably introduces some noise into the estimates. Discarding these attributes can both reduce error and classification time. Our proposed algorithm, Selective KDB (SKDB), extends KDB to select between attribute subsets and values of k in a single additional pass through the training data.

In the general case, attribute selection is a complex combinatorial task. For d attributes there are 2^d alternative attribute subsets that could be explored. Many attribute selection algorithms utilize either forward selection or backwards elimination hill-climbing strategies (Langley and Sage, 1994; Koller and Sahami, 1996). These start with either no attributes or all attributes and then iteratively add or remove one attribute at a time. This requires that all attribute subsets resulting from adding/removing any one attribute are considered at each step. There are at most d such steps resulting in $\mathcal{O}(d^2)$ attribute subset evaluations. If all candidate modifications to a current subset are considered simultaneously in a single pass through the data, this implies up to d passes for the attribute selection stage.

We seek to perform both attribute and best-k selection in a single pass. For attribute selection, we take advantage of the attribute ordering selected by KDB based on mutual information with the class. Given that the attributes are sorted on this order, we consider only the attribute sets $\{x_1, \ldots x_i\}, 1 \leq i \leq d$, that is attribute sets that each contain the i attributes that have the greatest mutual information with the class. These d attribute subsets are evaluated simultaneously in a single pass through the data using leave-one-out cross validation (LOOCV). For big data, LOOCV can be expected to provide an unbiased low-variance estimate of the out-of-sample error. It is possible to efficiently evaluate all d subsets simultaneously, because a KDB classifier using subset $\{x_1, \ldots x_i\}$ is a minor addition to a KDB classifier using subset $\{x_1, \ldots x_{i-1}\}$. Calculating a KDB classifier for all attributes

already incorporates all the calculations for all the KDB classifiers using subsets that we consider.

We use incremental cross validation (Kohavi, 1995), which performs LOOCV on BNCs very efficiently. A naive approach to LOOCV will for each holdout example recalculate from scratch all of the joint frequency counts required by a BNC. Incremental cross-validation first gathers the counts for all training examples in a single pass through the data. Then, when classifying a holdout example, its counts are removed from the table. SKDB collects the complete counts in its second pass through the training data and performs LOOCV in a third pass.

Using the given order, the number of attributes with the best leave-one-out error is selected. In case of a draw, preference is given to the smallest number of attributes. Any loss function can be used that is a function, $\{\langle y^{\mathbf{x}}, \hat{p}(Y, \mathbf{x})\rangle \mid \mathbf{x} \in \mathcal{T}\} \rightarrow \mathbb{R}$ over all objects $\mathbf{x} \in \mathcal{T}$ of a predicted class distribution $\hat{p}(Y, \mathbf{x})$ and the true class for that object, $y^{\mathbf{x}}$. Such loss functions include 0-1 loss, root mean square error, log-loss, Matthews correlation coefficient (Matthews, 1975) or Brier score (Brier, 1950) to name a few. Because we believe it to be an effective measure of the calibration of a classifier's class probability estimates, we use root mean squared error (RMSE) as follows:

$$RMSE = \sqrt{\frac{1}{t}\sum_{\mathbf{x}\in\mathcal{T}} (1 - \hat{p}(y^{\mathbf{x}}|\mathbf{x}))^2} \tag{1}$$

where $\hat{p}(y^{\mathbf{x}}|\mathbf{x})$ is the estimated posterior probability of the true class given $\mathbf{x}$, $y^{\mathbf{x}}$ the class label for the example $\mathbf{x}$.

In order to make full use of available computational resources while reducing the risk of overfitting, we propose to select algorithmically the best value of k for a particular dataset. Just as the KDB classifiers built on successive attribute subsets are embedded one inside the other, a KDB classifier with $k = i$, can be evaluated with little additional computational effort during the process of evaluating a KDB classifier with $k = i + 1$. We leverage this capacity to simultaneously evaluate KDB classifiers with all attribute subsets $\{x_1, \ldots x_{i-1}\}$ and values of k up to the maximum capacity available, taking advantage of the above mentioned LOOCV. The combination of attribute subset and k with the best LOOCV error is selected. In case of a draw, the smaller attribute subset and smaller value of k are selected. The resulting SKDB algorithm is presented in Algorithm 4.

In the first pass our implementation of SKDB generates a three-dimensional table of co-occurrence counts for each pair of attribute values and each class value. This is required to calculate the mutual information of each attribute with the class and the conditional mutual information of every pair of attributes given the class. The resulting space complexity is $\mathcal{O}(c(dv)^2)$. The time complexity of forming the three dimensional probability table is $\mathcal{O}(td^2)$, as an entry must be updated for every training case and every combination of two attribute-values for that case. To calculate the conditional mutual informations, SKDB must consider every pairwise combination of their respective values in conjunction with each class value $\mathcal{O}(c(dv)^2)$. Attribute ordering and parent assignment are $\mathcal{O}(d \log d)$ and $\mathcal{O}(d^2 \log d)$ respectively.

The second pass requires d tables of $k+2$ dimensions (one for each of k parents, one for the child and one for the class), with $\mathcal{O}(cdv^{k+1})$. Updating the probability tables is $\mathcal{O}(tdk)$.

Algorithm 4: The SKDB algorithm

Input: Training set $\mathcal{T}$ with attributes $\{X_1, \ldots, X_a, Y\}$ and $k^{\max}$.
Output: SKDB model.

First pass begin
  $\mathcal{G}$ =**learnStructure**($\mathcal{T}$) ; /* Algorithm 2 */
end
Second pass begin
  Θ = **learnParameters**($\mathcal{T}$, $\mathcal{G}$) ; /* Algorithm 3 */
end
Let $\mathcal{L}$ **be a list of all** X_i **in decreasing order of** $MI(X_i; Y)$**.**
Let $\mathcal{P}$ **be a** $k \times a$ **matrix of posterior probabilities;**
Let $\mathcal{LF}$ **be a matrix of LOOCV results (of length** $k \times a$**); Initialize it with zeros;**
Let $\Theta^{\downarrow \mathbf{x}}$ **be the BN** Θ **with example** $\mathbf{x}$ ***discounted*** **from its CPTs;**
Third pass ($\forall \mathbf{x} \in \mathcal{T}$) **begin**
  for $k' = 1$ **to** $k^{\max}$ **do**
    $\mathcal{P}[k'][y^*] = \hat{p}_{\Theta^{\downarrow \mathbf{x}}}(y^*|\mathbf{x})$, $\forall y^* \in Y$;
    for $l = 1 \rightarrow l = \mathcal{L}.size$ **do**
      $X_{\max} = \mathcal{L}.nextElement$;
      $\mathcal{P}[k'][y^*] = \mathcal{P}[k'][y^*] \cdot \hat{p}_{\Theta^{\downarrow \mathbf{x}}}(x_{\max}|pa^{k'}_{x_{\max}}, y^*)$, $\forall y^* \in \Omega_Y$; **where** $pa^{k'}_{x_{\max}}$ **are the** k' **first parent-values of** $X_{\max}$ **in** $\mathbf{x}$**.**
      $\mathcal{LF}[k'][l] = \mathcal{LF}[k'][l] + LossFunction(\mathcal{P}[k'], y^{\mathbf{x}})$; **where** $\mathcal{P}[k']$ **is the vector of posterior probabilities considering the top** l **attributes by MI.**
    end
  end
end
Select b **and** k **indexes with best** $\mathcal{LF}$**;**
Truncate Θ **to attribute subset** $\{1 \ldots b\}$ **and maximum number of parents** k**;**
return Θ*;*

The algorithm to this stage is simply KDB, with time complexity $\mathcal{O}(td^2 + c(dv)^2 + tdk)$, where v is the maximum number of values per attribute, and space complexity $\mathcal{O}(cdv^{k+1})$.

SKDB then extends KDB with an extra pass through the training data to perform leave-one-out cross validation for the different (ordered) attributes $\mathcal{O}(tcdk)$.

At classification time, SKDB need only store the conditional probability distribution for the d_b selected attributes and the k_b selected, $\mathcal{O}(cd_bv_b^k)$. The time complexity of classifying a single example is $\mathcal{O}(cd_bk_b)$.

In summary, SKDB requires an extra pass over the data compared with KDB, and the complexity of this extra pass is linearly dependent on the number of training examples, classes, attributes and k so the order of the time complexity increases from $\mathcal{O}(td^2 + c(dv)^2 + tdk)$ to $\mathcal{O}(td^2 + c(dv)^2 + tcdk)$. This is not a great increase in practice, since the number of classes is generally small, and indeed for high dimensional data the complexity of the first

pass can dominate that of the subsequent passes. Classification time and space complexity are reduced compared to KDB if the number of selected attributes $d_b < d$ and/or $k_b < k_{\max}$.

For the cost of this modest increase in computation, KDB is extended so as to discriminatively choose between a large class of alternative BNC models. Note that while it is possible to use cross validation to select between any collection of alternative models, it is only possible to do so in the extremely efficient manner we have developed for a very restricted class of learners, those that can be evaluated using incremental cross-validation and can be decomposed into nested models. The only types of learners of which we are aware that belong to this class are KDB and ANDE. We leave applying the approach to ANDE and variants of KDB as a topic for future research.

It may be possible to gain further efficiencies through sampling. As discussed in Hulten and Domingos (2002), it is sometimes possible to obtain sufficiently accurate estimates of the parameters for a learning program from a subsample of the full data. For very large datasets we could potentially utilize a random sample for the first pass of our algorithm, which only needs to estimate the parameters for three-way interactions between each pair of attributes and the class. We believe that the best value of k will usually be the one for which there is only just enough data in the dataset to obtain sufficiently accurate estimates of the required parameters. Thus we do not believe that sampling will usually be useful for the second pass. A technique such as Racing (Maron and Moore, 1994) could be used to select the model in the final pass. However, we suspect that it will usually require extremely large samples to confidently distinguish between the large number of very similar models that are assessed in the third pass. Given that the computational overheads of scanning a large dataset in random order are considerable, we do not believe that sampling will often be useful. However, this also remains a potential direction for future research.

3. Experimental Methodology

We undertook an extensive online search to gather a group of large datasets, all of which have more than 100K instances. These datasets are described in Table 7 in Appendix A, in ascending order of number of samples. Those below the double line have more than 1 million instances. All datasets except for `poker-hand`, `uscensus1990` and `splice` contain one or more numeric attributes. 7 datasets contain only numeric attributes: `MITFaceSetA`, `MITFaceSetB`, `MITFaceSetC`, `Mnist`, `USPSExtended`, `MSDYearPrediction` and `satellite`. For the Bayesian network classifiers we discretize numeric attributes using 5-bin equal frequency discretization. To avoid loading the whole data into memory, only a random sample of 100K points is used to define the bins for discretization.

As described in Section 2.1, we use SKBD with RMSE as the objective function for the third pass that selects between structures because we believe it to be a good measure of the calibration of a classifier's class probability predictions. However, this raises the issue that we might be reporting the measure that best favors our approach. To guard against this, we also present in the Appendices 0-1 loss results of the relative performance of all algorithms, and discuss these results in the appropriate sections below. Table 9 shows that when evaluated on RMSE, SKDB trained to optimize RMSE and SKDB trained to optimize 0-1 loss (SKDB$^{0\text{-}1}$) attain identical 0-1 loss on 12 out of 16 datasets, suggesting that SKDB's performance is similar when selection uses different but closely related loss functions.

	Train		Test	
	time	space	time	space
NB	$\mathcal{O}(td)$	$\mathcal{O}(cdv)$	$\mathcal{O}(cd)$	$\mathcal{O}(cdv)$
TAN	$\mathcal{O}(td^2 + c(dv)^2 + d^2 \log a)$	$\mathcal{O}(c(dv)^2)$	$\mathcal{O}(cd)$	$\mathcal{O}(cdv^2)$
AODE	$\mathcal{O}(td^2)$	$\mathcal{O}(c(dv)^2)$	$\mathcal{O}(cd^2)$	$\mathcal{O}(c(dv)^2)$
KDB	$\mathcal{O}(td^2 + c(dv)^2 + tdk)$	$\mathcal{O}(c(dv)^2 + cdv^{k+1})$	$\mathcal{O}(cdk)$	$\mathcal{O}(cdv^{k+1})$
SKDB$_k$	$\mathcal{O}(td^2 + c(dv)^2 + tcdk)$	$\mathcal{O}(c(dv)^2 + cdv^{k+1})$	$\mathcal{O}(cd_bk)$	$\mathcal{O}(cd_bv^{k+1})$
SKDB	$\mathcal{O}(td^2 + c(dv)^2 + tcdk)$	$\mathcal{O}(c(dv)^2 + cdv^{k+1})$	$\mathcal{O}(cd_bk_b)$	$\mathcal{O}(cd_bv^{k_b+1})$

Table 1: Orders of complexity for the different semi-naive BNCs where: t is the number of training examples, d is the number of attributes, v is the maximum number of values per attribute, c the number of classes, d_b (the best d) is the remaining attributes after applying SKDB, and k_b (the best k) is the k value selected in SKDB.

The structure of the experimental Section is as follows: Section 3.1 compares our proposed SKDB algorithm's results with different out-of-core semi-naive BNCs. Section 3.2 compares our SKDB algorithm with several *online* (out-of-core) linear classifiers. Section 3.3 includes comparisons with two state-of-the-art in-core machine learning algorithms, BayesNet (Augmented Bayesian network) and Random Forest. Section 3.4 presents a global comparison of all learners considered, along with empirical time comparisons for the different out-of-core classifiers. Each of these sections provides a summary of the relevant results. Detailed RMSE error outcomes are presented in Table 8 in Appendix A and a more detailed analysis of results is provided in in Appendix C.

All the experiments for the Bayesian out-of-core algorithms use C++ software specially designed to deal with out-of-core classification methods[1]. Appendix B specifies the implementation details.

Note that both RMSE and 0-1 Loss are assessed using 10-fold cross-validation. This should not be confused with the leave-one-out cross-validation used to select the number of attributes and parents in SKDB. For each fold of the 10-fold cross-validation, SKDB performs leave-one-out cross-validation on the training set to select the parameters of the model that is then tested on the holdout test fold.

3.1 SKDB vs Bayesian Out-of-core Algorithms

The first set of experiments compare SKDB with the out-of-core semi-naive BNCs described in Section 2: NB, TAN, AODE and KDB. We also consider a version of SKDB which fixes the value of k and just selects among the attributes, referred to as SKDB$_k$. Table 1 shows a summary of the orders of complexity for all these BNCs.

Tables 2 and 3 show win-draw-loss records summarizing the relative RMSE and 0-1 loss of the different approaches. Cell $[i, j]$ of each table contains the number of wins/draw/losses

1. A minimum functional part of the software containing SKDB can be found in the (two first) authors' academic web pages, for example `http://www.csse.monash.edu.au/~webb/`.

	KDB				
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
TAN	6-3-7	0-0-16	-	-	-
AODE	3-0-13	0-0-16	-	-	-
SKDB$_k$	8-8-0	6-10-0	6-10-0	7-9-0	6-9-1
SKDB	16-0-0	16-0-0	14-2-0	14-2-0	8-8-0
SKDB $k_{\max}=5$	6-10-0 (vs best k)*				

* The result is 6-9-1 (shift in uscensus) if SKDB is optimized with 0-1 loss.

Table 2: Win-draw-loss records when comparing the RMSE of different out-of-core BNCs.

	KDB				
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
TAN	7-2-7	0-0-16	-	-	-
AODE	4-1-11	0-0-16	-	-	-
SKDB$_k$	5-10-1	5-11-0	6-10-0	6-9-1	6-9-1
SKDB	16-0-0	14-2-0	13-3-0	10-5-1	6-9-1
SKDB $k_{\max}=5$	5-11-0 (vs best k)*				

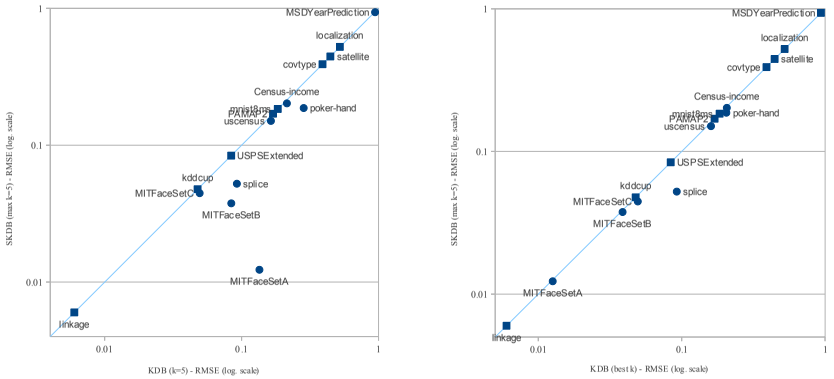
* The result is 6-10-0 if SKDB is optimized with 0-1 loss.

Table 3: Win-draw-loss records when comparing the 0-1 loss of different out-of-core BNCs.

for the classifier on row i against the classifier on column j. A win indicates that the algorithm has significantly lower error than the comparator. A draw indicates that the differences in error are not significant. Statistical significance has been assessed using Wilcoxon signed-rank tests to compare the 10-fold outputs. SKDB$_k$ and SKDB are compared against standard KDB with the same value of k. Additionally, SKDB with $k=5$ is compared against the lowest error of any standard KDB with k between 1 and 5 (which is a theoretical result, since the value of k that will minimize out of sample error can only be determined a posteriori).

The results indicate that TAN is competitive with KDB $k=1$, whereas AODE only achieves lower error on three or four of the datasets, depending on the loss function. We believe the reason AODE performs so poorly is that the advantage it gets in reduced variance from ensembling the d models is less useful for larger data. Both TAN and AODE consistently show higher error than higher orders of KDB on these large datasets. SKDB$_k$ only increases the RMSE of KDB relative to one value of k and only on one dataset, MITFaceSetB, and then only from 0.0841 to 0.0844. For many datasets it does not affect error (but nonetheless speeds up classification time). For many datasets it substantially reduces error, such as the reduction from 0.2048 to 0.1868 for poker-hand.

The average best k for SKDB across the 16 datasets is 4.08. On average, $78.46 \pm 29\%$ of the attributes are selected, which explains some of the ties, although in other cases SKDB



(a) KDB with $k=5$ and SKDB ($k_{\max}=5$).

(b) KDB with the best k for each dataset and SKDB ($k_{\max}=5$).

Figure 2: Scatter plot of RMSE comparisons for KDB and SKDB.

obtains the same results as KDB with the best value of k, but reducing the number of attributes. Indeed, reducing the number of attributes without affecting error is the worst case encountered by SKDB, because it never loses in terms of RMSE compared to the best KDB).

One could think that KDB with $k=5$ would be the fairest comparison with SKDB ($k_{\max}=5$). This comparison is shown in Figure 2a, where the X-axis represents the RMSE results with KDB with $k=5$ for the 16 datasets and the Y-axis the RMSE with SKDB. We have used a logarithmic scale because almost half of the points fall below the 0.1 range. Note that there is not a single point above the diagonal line, which indicates that SKDB is never worse than KDB $k=5$ or even KDB with the best k value considered, as shown in Figure 2b.

If only the best number of parents is selected by SKDB (best k) and no feature selection is performed, then the RMSE is always equivalent to the best RMSE obtained by KDB with any value of $k \in \{1,5\}$. That is, for all datasets SKDB without attribute selection is successful at selecting the best value of k with respect to the specified loss function. However, for five datasets the value of k that attains the best RMSE is different to that which attains the best 0-1 loss, demonstrating the value of optimizing with respect to the relevant loss function. These results are summarized on row SKDB$_{only_k}$ in Tables 8 and 9. The non-shaded cells on these rows correspond to those datasets for which feature selection contributes to further reduce the error of SKDB. There are 9 datasets in total for which the feature selection part of the algorithm does not reduce RMSE and 10 for which it does not reduce 0-1 loss. Note, however, that the extra computational complexity added by the feature selection evaluation is only related to the complexity of the loss function utilized. In the case of considering RMSE-like functions, this extra time is negligible, but if more

(a) Training Times (b) Classification Times

Figure 3: Training and classification time comparisons for KDB and SKDB$_k$.

complex evaluation functions were to be used in a highly time-constrained scenario, then only a random sample of instances could be considered at this step.

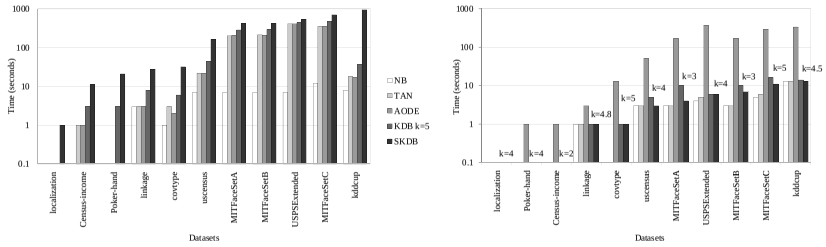
Figures 3a and 3b show the training and classification time comparisons for KDB and SKDB$_k$. Only the 11 smallest out of the 16 datasets have been considered for time measurement, since these experiments have been conducted on a desktop computer with an Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz, 3101 MHz, 64 bits and 7866 MiB of memory, whereas the remaining experiments have been conducted in a heterogeneous grid environment for which CPU times are not commensurable. Each bar represents the sum of all 11 datasets[2] in a 10-fold cross validation experiment. These graphs reinforce what the orders of complexity for the two algorithms indicated, that is, SKDB$_k$ requires a bit more time for training (extra pass on the data), while the classification time is less in practice, since the number of attributes selected is generally smaller than d.

Figures 4a and 4b show the training and classification empirical time comparisons of the different out-of-core BNCs relative to SKDB (with $k_{\max=5}$). The datasets are ordered at increasing time for SKDB. Note that SKDB always takes a bit more time for training, taking the largest time for `kddcup`, due to its large number of classes. Nevertheless, for the datasets with many attributes: `MITFaceSetA`, `MITFaceSetB`, `USPSExtended` and `MITFaceSetC`; the time differences with other BNCs become smaller. At classification time some gain can be appreciated for SKDB compared to KDB $k = 5$ if the selected k is less than 5. AODE's classification time is quadratic in the number of attributes, and hence much higher for all datasets.

3.2 SKDB vs Non-Bayesian Out-of-core Algorithms

In this section we compare SKDB with the state-of-the-art in online learning: Logistic Regression with Stochastic Gradient Descent (LRSGD). We use LRSGD's implementation in Vowpal Wabbit (VW) (Agarwal et al., 2014), an open source out-of-core linear learning system. VW provides a scalable implementation of LRSGD, which can be used in combination with several loss functions, different parameters for the learning rate and regularization, and quadratic or cubic combination of features.

2. census-income, covtype, donation, kddcup, localization, MITFaceSetA, MITFaceSetB, MITFaceSetC, USPSExtended, poker-hand, uscensus1990.



(a) Training Times (b) Classification Times

Figure 4: Time comparisons per dataset for NB, TAN, AODE, KDB against SKDB.

We have performed experiments with varying numbers of passes for LRSGD, varying loss functions and combinations of features. We use the default advanced (step size) updates provided by VW, which are:

1. Importance weight invariance (Karampatziakis and Langford, 2011): this update is meant to be useful specially for cost-sensitive learning, that is, when the misclassification penalty varies for the different class labels. However, it is also claimed to reduce classification error even when no matrix of weights is provided. The following equation is used to compute the weights:

$$w_i \leftarrow w_i - s(\eta I)\frac{\partial L(\hat{y}_w(x), y)}{\partial w_i}, \tag{2}$$

where y is the true class; $\hat{y}_w(x)$ is the predicted value for example $x \in \mathbb{R}^d$ with parameters $w \in \mathbb{R}^d$; $L(\hat{y}_w(x), y)$ is the loss function; η is the learning rate; $I \in \mathbb{N}$ the number of times an example is more important than a typical example, in our case $I = 1$; and $s(\eta I)$ is a scaling factor, for which a closed form solution can be found for each loss function (Karampatziakis and Langford, 2011) .

2. Adaptive updates (Duchi et al., 2011; McMahan and Streeter, 2010): the learning rate must decay to converge. Instead of using $\eta_t = \frac{1}{\sqrt{t}}$ or $\eta_t = \frac{1}{t}$ at iteration t for all features, we use a per-feature learning rate decay:

$$w_i \leftarrow w_i - \eta\frac{g_{it}}{\sqrt{\sum_{t'=1}^{t} g_{it'}^2}}, g_{it} = \frac{\partial L(\hat{y}_w(x_t), y_t)}{\partial w_i}, \tag{3}$$

3. Normalized updates (Ross et al., 2013): instead of using Gaussian sphering standard solution, which would imply in-core processing and an increase in the size of the dataset, a scale-free update is used that renormalizes w_i at each timestep and keeps track of a global scale.

A fair comparison of LRSGD with SKDB should perform the same number of passes on the data. Since LRSGD requires data to be pre-shuffled, an n-pass LRSGD learner

		VWLF	VWSF
SKDB	RMSE	$7^{(+2)}$-0-7	14-0-2
	0-1 Loss	$8^{(+1)}$-1-$5^{(+1)}$	8-1-7

Table 4: Win-draw-loss records for VW and SKDB in terms of RMSE and 0-1 Loss

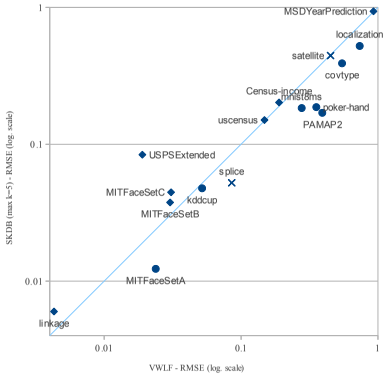
actually results in $n+1$ passes through the data. This is also true for SKDB on numeric data, as an extra pass is required for discretization. We have run experiments with 3, 10 and 20 passes for LRSGD. Each 10-fold cross validation experiment is repeated 10 times. Both the quadratic and logistic loss functions are considered. The one-against-all reduction from multiclass classification to binary classification has been used for the datasets with multiple class labels, which creates one binary problem for each of the classes. Table 10 in Appendix A shows the results for all of the different combinations for which we have conducted comprehensive experiments. The overall best results for LRSGD are obtained with 3 passes and using quadratic features. When the squared loss function is optimized we refer to it as VWSF, and VWLF if the logistic function is used. In all cases, SKDB is highly competitive compared to VW.

Note that while LRSGD has a number of parameters that could potentially be tuned using cross validation, doing so does not make a reasonable comparator to SKBD as such tuning would require multiple passes through the data for each cross-validation fold.

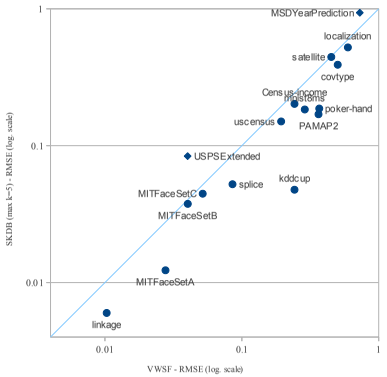
A win-draw-loss record summary is displayed on Table 4. The $^{(+1)}$ and $^{(+2)}$ for SKDB vs VWLF aim at distinguishing the results for `satellite` and `splice`, for which more than 400 hours are required to get the 10cv results using VWLF, and hence VWSF is considered instead. Irrespective of the fact that the SKDB model is optimized with respect to RMSE, the win-draw-loss records only vary for three datasets between evaluation with respect to RMSE or 0-1 loss. The win-draw-loss records remain exactly the same whichever loss function is optimized in VW (see Table 10 for more details).

Since the output of the VW algorithms is not probabilistic, we are using the class prediction, and hence computing 0-1 loss for comparisons with SKDB as well. By using, for example, the following sigmoid function $\frac{1}{1+e^{-\hat{y}w(x)}}$, probability estimates can be estimated from VW. Using this approach, the RMSE results for VWSF are only better than SKDB for two of the datasets (MSDYearPrediction and USPSExtended), both of them originally sparse and containing exclusively numeric attributes. These records improve when using a logistic function for optimization, that is VWLF (see Table 10 in Appendix A for more details). Nevertheless, computation time for VWLF is much larger than for VWSF as shown below.

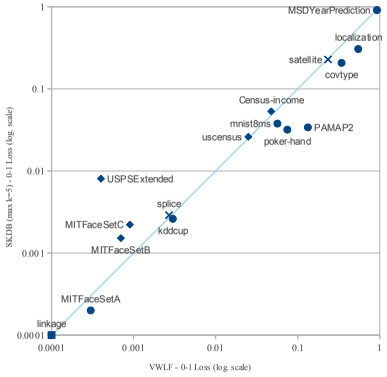
Figures 5a and 5b show the comparisons per dataset with SKDB in terms of RMSE, whereas Figures 5c and 5d show comparisons per dataset for 0-1 Loss. Figures on the left (5a and 5c) show VW optimizing a logistic function, while the two figures on the right (5b and 5d) show VW optimizing RMSE. A diamond symbol is used for those points above the diagonal line, indicating better results for SKDB, as opposed to the bullets. A squared symbol is used to show the ties. We can see that VW achieves lower error using a logistic function (VWLF), with the same number of datasets above and below the diagonal line, as opposed to only 2 wins for VW with the squared function (VWSF). Figures 5c and 5d shows



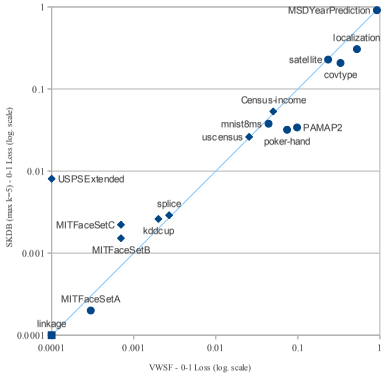
(a) RMSE for VWLF and SKDB.



(b) RMSE for VWSF and SKDB.



(c) 0-1 loss for VWLF and SKDB.



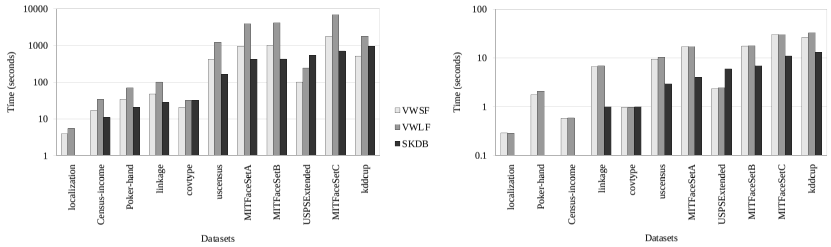
(d) 0-1 loss for VWSF and SKDB.

Figure 5: Scatter plot of RMSE and 0-1 loss comparisons for VW and SKDB ($k_{\max} = 5$).

the comparisons of VW with SKDB in terms of 0-1 Loss. In this case, VWSF performs slightly better than VWLF compared to SKDB.

Figure 5 shows the relative training and test times for VW and SKDB across all datasets. SKDB has substantially lower training and test time on most datasets. One exception is the sparse numeric USPSExtended dataset. Another is covtype, which has a large number of sparse boolean attributes.

While neither SKDB nor VW dominates the other in terms of error, SKDB is substantially more efficient, as shown in Figure 6a, where again training and classification times per dataset for VWLF, VWSF and SKDB are displayed. There is only one dataset for which



(a) Training Times (b) Classification Times

Figure 6: Time comparisons per dataset for VW with squared and logistic functions against SKDB.

SKDB takes more than the alternatives, which is `USPSExtended`. We believe that for this dataset VW is able to exploit the sparseness of the data while SKDB cannot. Some of the reasons why VW is taking in general more time than SKDB both for training and classifying is the consideration of quadratic features, the need to binarize discrete attributes (creating many binary attributes) and the need also to use one-against-all for multiclass classification.

3.3 SKDB vs In-core Algorithms

In order to understand how much predictive capacity, if any, is sacrificed by using our out-of-core classifiers, it is useful to compare them to the state-of-the-art in in-core classification. To this end, we have replicated our set of experiments with two in-core algorithms: a more general BNC and Random Forest, a powerful exemplar of the state-of-the-art. For some datasets it was infeasible to get results, even with high-performance computers, due either to time or memory limitations (indicated in Table 8 in Appendix A with $>$ 600h, when more than 600 CPU hours are required; or by >138G, when more than 138GB of RAM memory are required).

Note that while each of these algorithms has a number of parameters that could potentially be tuned using cross validation, doing so would not make reasonable comparators to SKBD. First, SKDB uses cross-validation to choose between multiple models rather than to choose between parameterizations. Cross validation could also be used to tune SKBD parameters such as the smoothing technique. Second, SKDB uses cross-validation in a constrained manner that requires only a single additional pass through the data while parameter tuning of these algorithms would increase compute time proportionally to the number of folds employed.

3.3.1 Augmented Bayesian Network

We compare performance against a state-of-the-art generic in-core Bayesian network classifier, that uses a hill-climbing algorithm for adding, deleting and reversing arcs (starting with a NB structure, this hill climber also considers arrows as part of the NB structure for

	BayesNet
SKDB	$6^{(+4)}$-3-3

Table 5: Win-draw-loss records for BayesNet and SKDB in terms of RMSE.

deletion, that is, arcs from the class to the attributes can be removed). It optimizes the K2 metric (Cooper and Herskovits, 1992). The maximum number of parents allowed for a node in the Bayes net is set to 5. The win-draw-loss comparison is presented in Table 5.

The detailed results can be found in Table 8 (Appendix A), row BayesNet. The darker grey background corresponds to those cases where BayesNet significantly outperforms SKDB. Note that there are only records for 12 out of the 16 datasets, because there are four cases for which results could not be computed for BayesNet due to memory constraints, since the network cannot be learned incrementally.

Even though BayesNet performs a more exhaustive search than SKDB of the model space of Bayesian networks with up to 5 parents, SKDB obtains lower error than BayesNet more often than the reverse. We hypothesize that this is due to SKDB's use of both generative and discriminative measures for model selection in contrast to BayesNet's use exclusively of generative measures.

3.3.2 Random Forest

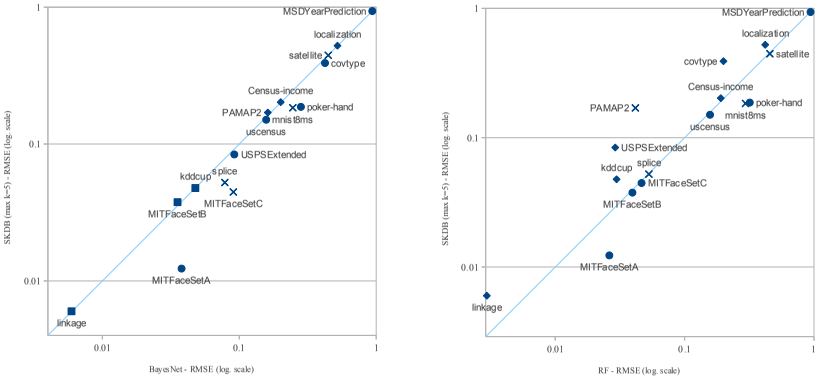
Random Forest (RF) is a state-of-the-art learning algorithm introduced in Breiman (2001). It uses bagging (Breiman, 1996) to aggregate an ensemble of trees that are each grown using a process that involves a stochastic element to increase diversity.

We have conducted experiments with RF selecting 100 trees. We present results both with pre-discretized data, to show relative performance on categorical data, and with undiscretized data. The detailed results can be found on the fifth and sixth from last rows in Table 8 in Appendix A. Table 6 shows the win-draw-loss records when compared with SKDB. The superscripts compute those datasets for which results cannot be obtained for RF due to main memory constrains ($>$ 138G are required). The results show that SKDB is competitive with RF in terms of RMSE, even when RF is performing its own discretization (5 wins for SKDB and 6 wins for RF), for which the computational cost in terms of memory and time is much higher than SKDB's. When 0-1 Loss is considered, out-of-core RF wins more frequently than SKDB, even when 0-1 Loss is used as the objective function during structure selection. When MDL discretization is used instead of 5EF for SKDB, then the results improve slightly for SKDB compared to RF performing its own discretization. When interpreting these results it is important to keep in mind that SKDB is a 3-pass out-of-core algorithm (4-pass with discretization) in contrast to the inherently computationally intensive in-core processing of RF.

Note that the sum of some cells is not exactly 16 but smaller, due to computational constraints for RF. RF is an in-core algorithm, and hence we have not been able to learn classification models for the largest datasets. Interestingly, RF is more negatively affected (in terms of computation) by a large number of classes than the BNCs, for example no 100 tree model can be learnt for `MSDYearPrediction`, which has 90 class labels. Precisely, there are three cases where RF with 10 trees cannot be run, and four in the case of RF with 100

		RF (5EF)	RF (Num)
RMSE	SKDB	$5^{(+4)}$-1-6	$5^{(+5)}$-0-6
0-1 Loss	SKDB	$2^{(+4)}$-3-7	$2^{(+5)}$-2-7
	SKDB$^{0-1}$	$3^{(+4)}$-2-7	$3^{(+5)}$-1-7
	SKDB^{MDL}	$8^{(+4)}$-2-2	$3^{(+5)}$-2-6

Table 6: Win-draw-loss records for RF and SKDB in terms of RMSE and 0-1 Loss.



(a) BayesNet and SKDB. (b) RF and SKDB.

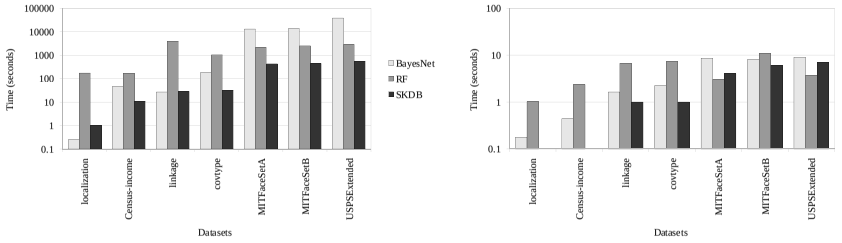
Figure 7: Scatter plot of RMSE comparisons for BayesNet, RF and SKDB.

trees. These four datasets are `MSDYearPrediction`, `satellite`, `mnist8ms` and `splice`, so the largest possible samples able to be computed have been considered.

RMSE comparisons for all datasets for BayesNet and RF when compared with SKDB are shown in Figures 7a and 7b, where the points with an X symbol indicate those datasets for which sampling is required. While BayesNet is not competitive with SKDB, RF provides competitive results for some datasets. `PAMAP2` provides a notable case for which only a sample of 2 millions out of 3.5 suffices for RF to provide better classification performance than SKDB using all the training data. This is however the only dataset for which RF learning from a sample can achieve lower RMSE than SKDB learning from all the data, the reverse being true for `splice`, `mnist` and `satellite`.

Figures 8a and 8b display the training and classification times per dataset for these classifiers compared to SKDB. For most datasets the in-core learners require substantially more time for learning and classifying[3].

3. RF requires more than 8GB of main memory to complete for poker-hand and uscensus1990 since these experiments to measure time has been conducted in a desktop computer no time measurements can be provided to compare with the rest of experiments. Weka's implementation has been considered.



(a) Training Times (b) Classification Times

Figure 8: Time comparisons per dataset for BayesNet and RF against SKDB.

3.4 Global Comparison of All Classifiers

In this section we analyze how all the classifiers explored in this paper perform when they are compared as a set. Figure 9 plots the average rankings across all datasets, along with the standard deviation for each learner. This ranking corresponds to the ranking obtained when a Friedman test is conducted. In this analysis we present the 0-1 Loss performance of SKDB when it is optimized for RMSE, its average rank on 0-1 Loss improves marginally from 2.94 to 2.84 when it is instead optimized for 0-1 Loss.

When assessing the calibration of the probability estimates using RMSE, SKDB obtains the lowest average rank of 2.53, followed by RF with 2.91 and KDB with 3.34 (very close to those for VW and BayesNet). When assessing performance using 0-1 Loss, RF using its own internal discretization of numeric attributes enjoys an average advantage of 1 in rank over SKDB, and SKDB enjoys an average advantage of almost one third over VWLF.

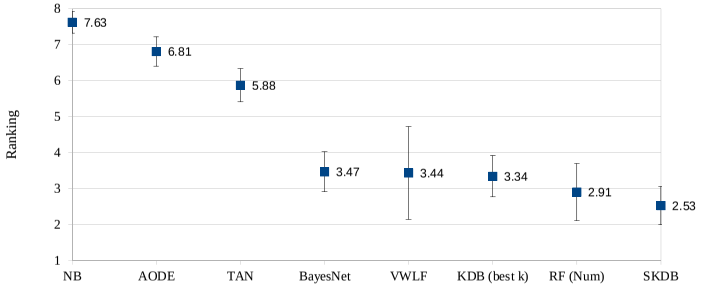
We find NB at the other extreme on both measures, with average ranks 7.63 and 7.72 out of a total of 8 classifiers.

SKDB obtains in the worst case the 4.5th position (this half is due to a tie with KDB-best-k) for the *linkage* dataset and the 4th position for *census-income* when ranked on RMSE and the 4th position for *MITFaceSetB* and *kddcup* when ranked on 0-1 Loss.

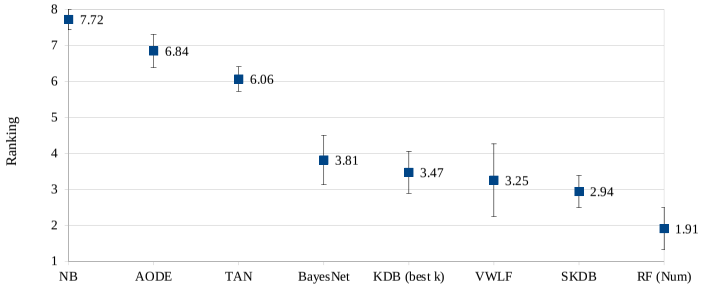
On both measures the largest standard deviation is observed for VW, which usually ranks either very well or rather poorly among the datasets. Still, VW obtains the largest number of top 1 datasets for RMSE, with a total number of 7, against 5 for RF and 4 for SKDB. On 0-1 Loss, RF ranks first 8 times, VW 4 times and SKDB twice (one of those times being a tie with KDB-best-k).

A Friedman test (Demšar, 2006) was performed for the set of 8 classifiers, yielding statistical difference for both measures. To identify where exactly these differences are found, we ran a set of posterior Nemenyi tests. These tests confirm the existence of two sets of algorithms in terms of performance on both RMSE and 0-1 Loss for these large datasets: on one hand NB, AODE and TAN; and on the other hand BayesNet, VW, KDB, RF and SKDB. The Nemenyi tests[4] showed significant differences for the second set of algorithms over NB and AODE, and only RF and SKDB over TAN.

4. Similarly for Holms and Shaffer post-hoc tests.



(a) RMSE

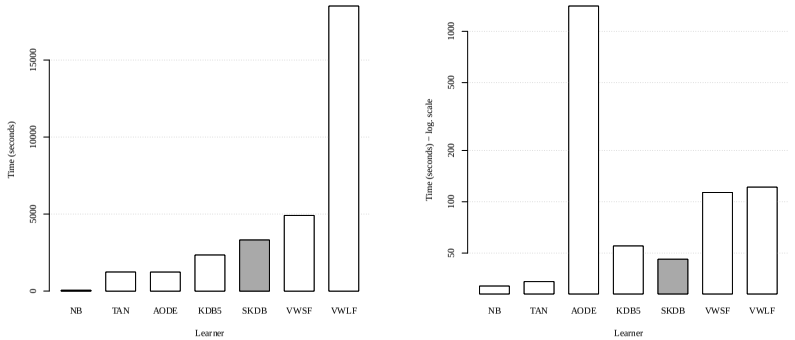


(b) 0-1 Loss

Figure 9: Ranking in terms of RMSE and 0-1 Loss for NB, TAN, AODE, KDB (with best k), SKDB, RF using numeric attributes, BayesNet and VW with a logistic function.

Even though the small number of datasets only allow tentative conclusions to be drawn, the following dataset characteristics appear to indicate a preference for one or another learning scheme.

- RF performs better on datasets with a small number of attributes.
- RF derives an advantage with respect to 0-1 Loss due to its internal discretization of numeric values. It does not achieve the lowest RMSE or 0-1 Loss on any of the three categorical only datasets *poker-hand*, *PAMAP2* or *splice*.
- SKDB seems to cope well with high-dimensional datasets and datasets with more than 1 million points. However its complexity is quadratic with respect to the number of attributes so, like most existing BNCs, it is not feasible for *very* high-dimensionality.



(a) Training Times

(b) Classification Times

Figure 10: Training and classification time comparisons for the out-of-core classifiers.

- VW appears to have an advantage for sparse numeric datasets.

A more detailed study of the domain of competence of these classifiers remains an important area for future work.

We can also analyze the performance of these classifiers with respect to the complexity of the decision function they provide. NB and linear regression, represented in this work by VW with linear features, are classifiers that learn linear decision boundaries. They have low variance and are particularly useful for small data quantity and sparse data. An extra level of expressiveness is offered by AODE, KDB ($k = 1$), TAN and VWSF/VWLF (with quadratic features); which produce quadratic decision functions. A2DE, KDB ($k = 2$) and VWSF/VWLF (with cubic features) produce cubic decision functions. More generally, AnDE and KDB can separate order-k(n) polynomially separable concepts (Jaeger, 2003). These two families of classifiers provide a set of a spectrum of models of increasing complexity, allowing a fine-tuned trade-off between expressivity on the one hand, and efficiency of learning and guarding against overfitting on the other hand, that is, the well-known bias-variance trade off (Brain and Webb, 1999).

Figures 10a and 10b show the time comparison for the out-of-core classifiers considered: namely NB, TAN, AODE, KDB with $k = 5$, SKDB and LRSGD with quadratic features in VW (using either a squared and a logistic function). The time required to pre-randomize the data for VW has not been included. No parallelization techniques have been used in any case, although most algorithms could be parallelized. The average of the 10-fold CV is shown. It is important to note that logarithmic scale has been used for classification times in order to appreciate the smaller differences in the faster learners. Again, only the 11 smallest out of the 16 datasets have been considered for time measurement, since these experiments have been conducted in a personal computer for a controlled environment.

Note that training SKDB with $k = 5$ (maximum) takes only a bit more time than training KDB with $k = 5$. The classification time for KDB is slightly higher than SKDB's (since all attributes are considered). Training time for LRSGD with quadratic features and 3 passes (VWSF and VWLF), which are those that obtain comparable results with SKDB, are also higher, presumably because they have to compute all pairs of quadratic features as indicated above. Similar reasoning holds for classification time. We believe that the extra time for VW is also due to the need to binarize discrete features, that is, since VW cannot deal with discrete attributes directly, v binary attributes are created for each originally discrete attribute with v values; and also because of the one-against-all approach required to handle classes with multiple labels. This overhead may be noticeable since we are using quadratic features. The time required for VWLF is much larger than VWSF, and in any case much larger both for training and testing than that for SKDB (specially note the logarithmic scale in Figure 10b).

4. Summary

We have developed an extension to KDB which performs discriminative attribute and best-k selection using leave-one-out cross validation.

SKDB only requires a single extra pass over the training data compared to KDB. In return for this extra pass during training, it often increases prediction accuracy relative to KDB with the best possible k, and always improves classification time (and space) by reducing the number of attributes considered, which can be quite significant for some datasets (for example for `MITFaceSetC` 60%, 215.8 out of 361, of the attributes are selected improving the classification performance at the same time). It is embarrassingly parallelizable. Furthermore, it can optimize any loss function that is a function over each training example $\mathbf{x}$ of $\hat{p}(Y, \mathbf{x})$ and $y^{\mathbf{x}}$, making it extremely flexible.

Due to its low bias, when datasets are large enough to avoid overfitting, SKDB attains lower error than the other in and out-of-core Bayesian network algorithms considered. It even provides competitive error compared to Random Forest, one of the most powerful in-core classification methods; and online linear and non-linear classifiers with stochastic gradient descent (that is, linear regression with quadratic and cubic features). Compared to the latter, it is in most cases substantially more efficient both at training and classification time. Further, it provides well-calibrated posterior class probability estimates.

Future directions for research include:

1. using only a sample of the data in the leave-one-out cross validation, in order to use more complex loss functions efficiently;

2. tackling SKDB's tendency to overfit small training sets;

3. studying a customized selection of the different links and attributes in a discriminative manner, that is, different numbers of parents for different attributes and also discarded in an order that does not follow the conditional mutual information rank;

4. consideration of other means of generating alternative classifiers to be selected in the final LOOCV pass; and

5. study of more appropriate loss functions for imbalanced datasets.

SKDB provides an efficient and effective solution to the problem of scalable low-bias learning. Its low bias allows it to capture and take advantage of the additional fine-detail that is inherent in very large data while its efficiency makes it feasible to deploy. SKDB thus sets new standards for classification learning from big data.

Acknowledgments

This research has been supported by the Australian Research Council grant DP140100087 and Asian Office of Aerospace Research and Development, Air Force Office of Scientific Research contract FA2386-15-1-4007. It has also been supported in part by the Monash eResearch Center and eSolutions-Research Support Services through the use of the Monash Campus HPC Cluster and the LIEF grant. This research was also undertaken on the NCI National Facility in Canberra, Australia, which is supported by the Australian Commonwealth Government. The authors would also like to thank the researchers from CESBIO (D. Ducrot, C. Marais-Sicre, O. Hagolle and M. Huc) for providing the land cover maps and the geometrically and radiometrically corrected FORMOSAT-2 images (satellite dataset). We also want to thank the anonymous reviewers for their insightful comments which have helped to greatly improve the revised version of the paper.

Appendix A. Tables of the Experimental Section

Table 7: Very large datasets for classification. From left to right showing the identification number; name of the dataset; number of instances, attributes and classes; presence of missing values (and if positive, then maximum percentage per attribute and the average), source paper of the dataset; relevant papers and description.

id	name	t	a	c	? (average$\pm$sd) (max)	Source	Relevant papers	description	size
1	localization	0.165	5	11	N	UCI Kaliza et al. (2010)		Recordings of 5 people performing different activities. Each person wore 3 sensors while performing the same scenario 5 times.	11MB
2	census-income	0.299	41	2	Y (50%) (4.9 $\pm$ 14.6%)	UCI Bache and Lichman (2013)	Ozа and Russell (2001)	Weighted census data extracted from the 1994 and 1995 current population surveys conducted by the U.S. Census Bureau.	136MB
3	USPSExtended	0.341	676	2	N	CVM Tsang et al. (2005)		0/1 digit classification (extended version of the USPS data set[a]).	603MB[s]
4	MITFaceSetA	0.474	361	2	N	CVM Tsang et al. (2005)		Face detection using an extended version of the MIT face database[c]. By adding nonfaces to the original training set.	584MB
5	MITFaceSetB	0.489	361	2	N	CVM Tsang et al. (2005)		Each training face is blurred and added to set A. They are then flipped laterally.	603MB
6	MSDYear-Prediction	0.515	90	90	N	UCI Bertin-Mahieux et al. (2011)		Prediction of the release year of a song from audio features. Songs are mostly western, commercial tracks ranging from 1922 to 2011, with a peak in the year 2000s[c].	601MB[s]
7	covtype	0.581	54	7	N	UCI Blackard (1998)	Gama et al. (2003); Oza and Russell (2001)	Predicting forest cover type from cartographic attributes only (no remotely sensed data).	72MB
8	MITFaceSetC	0.839	361	2	N	CVM Tsang et al. (2005)		Each face in set B is rotated.	1.1GB
9	poker-hand	1.025	10	10	N	UCI	Cattral et al. (2002)	Each record is an example of a hand consisting of 5 cards drawn from a deck of 52. Each card is described using 2 attributes (suit and rank), thus 10 predictive attributes in total. The class label describes the "Poker Hand". The order of cards is important.	24MB
10	uscensus1990	2.458	67	4	N	UCI		Discretized version of the USCensus1990raw dataset, a 1% sample from the full 1990 census. "Temp. Absence From Work' has been selected as class.	325MB
11	PAMAP2	3.850	54	19	Y (91%) (1.6 $\pm$ 12.2%)	UCI Reiss and Stricker (2012)		Data of 18 different physical activities (such as walking, cycling, playing soccer, etc.), performed by 9 subjects wearing 3 inertial measurement units and a heart rate monitor.	1.7GB
12	Kddcup	5.209	41	40	N	UCI (KDD Cup 1999)		Contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment. "bad" connections, called intrusions or attacks, and "good" normal connections.	754MB
13	linkage	5.749	11	2	Y (100%) (16.5 $\pm$ 36.9%)	Schmidtmann et al. (2009)	Sariyar et al. (2011)	Element-wise comparison of records with personal data from a record linkage setting. The task is to decide from a comparison pattern whether the underlying records belong to one person.	251MB
14	mnist8m	8.100	784	10	N	CVM Loosli et al. (2007)	Erhan et al. (2010); Bottou et al. (1994)	Handwritten digit classification. Extended version by performing random elastic deformations of the original mnist digits.	19GB[s]
15	satellite	8.705	138	24	Y (93%) (26.4 $\pm$ 33%)	Petitjean et al. (2012)		Satellite image time series to predict land cover.	3.6GB
16	splice	54.627	141	2	N	Sonnenburg and Franc (2010)	Agarwal et al. (2014)	Recognising a human acceptor splice site (target public data for which subsampling is not an effective learning strategy).	7.3GB

[a] http://c2inet.sce.ntu.edu.sg/Ivor/cvm.html.
[b] http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html.
[c] Subset of the Million Song Dataset (http://labrosa.ee.columbia.edu/millionsong/). Songs from a given artist may end up in both the train and test set.
[s] The superscript *s* stands for sparse format.

Appendix B. Implementation Details

All the experiments for the out-of-core algorithms have been carried out in a C++ software specially designed to deal with out-of-core classification methods. The following details in terms of implementation and configuration should be considered:

- 10-fold cross validation has been used.
- Equal frequency pre-discretization with 5 bins (EF5) has been utilized to discretize all numeric attributes (except if otherwise specified[5]). We have observed that EF5 and MDL (Minimum Description Length) (Fayyad and Irani, 1993) discretization provide the best results in approximately half of the datasets each, EF5 has been chosen because it is faster to perform than MDL, and also because it is not supervised and hence does not potentially provide the classifier with class information from the hold-out data when used for pre-discretization. Using a pre-fixed number of bins gives us the added advantage of not having to deal with a huge number of values per attribute created by MDL discretization in some cases.
- Missing values have been considered as a distinct value in all cases.
- The root mean square error is calculated exclusively on the true class label (different from Weka's implementation (Hall et al., 2009), where all class labels are considered).
- KDB's implementation benefits from a tree structure that resembles an ADTree (Moore and Lee, 1998), sparseness is achieved by storing a NULL instead of a node for any query that matches zero records. In our case, to save the probability distribution of the different attributes once the structure has been determined, we build a (distribution) tree for each attribute. Given, for example, attribute X_4 that has X_0 and X_3 as parents (in this order, and apart from the class), the counts (X_4, Y) are stored in the root node. The following level expands according to the number of values of X_0, which is the first parent, and store the counts for (X_4, Y, X_0). The following level expands as for the number of values of X_3, which is the second parent, and store the counts for (X_4, Y, X_0, X_2), and so forth if there were more parents. Note that as opposed to the ADTree, all the parameters are stored, that is, more than strictly required, since a branch is created for each attribute value, when strictly one less branch would be sufficient to represent the model parameters. The reason to do that is not to overload classification time by having to explore and combine branches.
- Two combined techniques are considered for smoothing. In the first place, we use m-estimates[6] (Mitchell, 1997) as follows:

$$\hat{p}(x_i|\pi_{x_i}) = \frac{counts(x_i, \pi_{x_i}) + \frac{m}{|X_i|}}{counts(\pi_{x_i}) + m} \tag{4}$$

5. Only for the experiments on the antepenultimate row on Table 8 as specified.
6. Also known as Schurmann-Grassberger's Law when $m = 1$, which is a particular case of Lidstone's Law (Lidstone, 1920; Hardy, 1920) with $\lambda = \frac{1}{|X_i|}$, also based on a Dirichlet prior.

learner	args	localiz.	cens-inc	USPS	MITA	MITB	MSDY	covtype	MITC	poker	uscensus	PAMAP	kddcup	linkage	mnist8	satellite	splice
NB		0.7106± 0.0007	0.4660± 0.0018	0.2256± 0.0026	0.0982± 0.0029	0.1394± 0.0014	0.9600± 0.0002	0.5103± 0.0013	0.2367± 0.0021	0.5801± 0.0006	0.2911± 0.0006	0.4647± 0.0006	0.1849± 0.0008	0.0125± 0.0006	0.4957± 0.0005	0.6540± 0.0002	0.0971± 0.0002
TAN		0.6321± 0.0014	0.2247± 0.0025	0.1153± 0.0022	0.0202± 0.0025	0.1213± 0.0020	0.9436± 0.0002	0.4862± 0.0008	0.2068± 0.0017	0.4987± 0.0006	0.1845± 0.0009	0.3232± 0.0007	0.0572± 0.0006	0.0081± 0.0009	0.3817± 0.0006	0.5424± 0.0003	0.1020± 0.0003
AODE		0.6520± 0.0010	0.2932± 0.0020	0.1538± 0.0028	0.1001± 0.0036	0.1682± 0.0025	0.9459± 0.0002	0.4587± 0.0013	0.1564± 0.0014	0.5392± 0.0006	0.2154± 0.0010	0.3881± 0.0008	0.0979± 0.0007	0.0120± 0.0005	0.3497± 0.0005	0.5783± 0.0003	0.1034± 0.0003
KDB	$k=1$	0.6501± 0.0012	0.2219± 0.0024	0.1125± 0.0025	0.0209± 0.0021	0.1291± 0.0019	0.9447± 0.0003	0.4709± 0.0026	0.1940± 0.0021	0.4987± 0.0005	0.1814± 0.0008	0.3276± 0.0007	0.0606± 0.0005	0.0082± 0.0009	0.3751± 0.0005	0.5506± 0.0003	0.0940± 0.0003
	$k=2$	0.5840± 0.0020	0.2064± 0.0021	0.0990± 0.0029	0.0126± 0.0019	0.0633± 0.0018	0.9393± 0.0002	0.4576± 0.0013	0.0906± 0.0012	0.4055± 0.0007	0.1677± 0.0008	0.2721± 0.0005	0.0497± 0.0008	0.0062± 0.0007	0.2922± 0.0004	0.5072± 0.0006	0.0953± 0.0002
	$k=3$	0.5485± 0.0020	0.2102± 0.0015	0.0928± 0.0025	0.0142± 0.0018	0.0468± 0.0018	0.9361± 0.0004	0.4399± 0.0015	0.0705± 0.0018	0.2859± 0.0011	0.1664± 0.0008	0.2290± 0.0004	0.0485± 0.0008	0.0060± 0.0006	0.2500± 0.0005	0.4769± 0.0004	0.1008± 0.0005
	$k=4$	0.5225± 0.0023	0.2107± 0.0012	0.0877± 0.0026	0.0514± 0.0031	0.0387± 0.0021	0.9383± 0.0006	0.4126± 0.0019	0.0616± 0.0016	0.2048± 0.0012	0.1601± 0.0007	0.1937± 0.0004	0.0478± 0.0008	0.0060± 0.0006	0.2169± 0.0005	0.4583± 0.0005	0.1084± 0.0003
	$k=5$	0.5225± 0.0023	0.2143± 0.0022	0.0839± 0.0034	0.1350± 0.0035	0.0841± 0.0023	0.9447± 0.0003	0.3903± 0.0020	0.0494± 0.0015	0.2842± 0.0011	0.1638± 0.0006	0.1697± 0.0004	0.0477± 0.0008	0.0060± 0.0006	0.1839± 0.0005	0.4448± 0.0004	0.0924± 0.0002
SKDB_{only_k}	$k_{\max}=5$	0.5225± 0.0023	0.2064± 0.0021	0.0839± 0.0034	0.0126± 0.0019	0.0387± 0.0021	0.9361± 0.0004	0.3903± 0.0020	0.0494± 0.0015	0.2048± 0.0012	0.1601± 0.0007	0.1697± 0.0004	0.0477± 0.0008	0.0060± 0.0006	0.1839± 0.0005	0.4448± 0.0004	0.0924± 0.0002
	bestk	4	2	5	2	4	3	5	5	4	4	5	5	3.8	5	5	5
SKDB_k	$k=1$	0.6501± 0.0012	0.2054± 0.0014	0.1125± 0.0025	0.0207± 0.0020	0.0746± 0.0012	0.9446± 0.0003	0.4704± 0.0016	0.1227± 0.0014	0.4987± 0.0005	0.1540± 0.0007	0.3276± 0.0007	0.0606± 0.0005	0.0082± 0.0009	0.3751± 0.0005	0.5485± 0.0002	0.0527± 0.0002
	$k=2$	0.5840± 0.0020	0.2021± 0.0021	0.0990± 0.0029	0.0123± 0.0018	0.0574± 0.0011	0.9393± 0.0002	0.4576± 0.0013	0.0782± 0.0017	0.4055± 0.0007	0.1510± 0.0007	0.2721± 0.0005	0.0497± 0.0008	0.0062± 0.0007	0.2922± 0.0004	0.5065± 0.0004	0.0523± 0.0002
	$k=3$	0.5485± 0.0020	0.2056± 0.0016	0.0928± 0.0026	0.0144± 0.0021	0.0439± 0.0021	0.9361± 0.0004	0.4399± 0.0015	0.0596± 0.0020	0.2847± 0.0011	0.1508± 0.0007	0.2290± 0.0004	0.0485± 0.0008	0.0060± 0.0006	0.2500± 0.0005	0.4769± 0.0004	0.0523± 0.0002
	$k=4$	0.5225± 0.0023	0.2041± 0.0015	0.0877± 0.0026	0.0386± 0.0019	0.0376± 0.0026	0.9382± 0.0005	0.4126± 0.0019	0.0517± 0.0020	0.1868± 0.0012	0.1504± 0.0007	0.1937± 0.0004	0.0478± 0.0008	0.0060± 0.0006	0.2169± 0.0005	0.4583± 0.0005	0.0524± 0.0002
	$k=5$	0.5225± 0.0023	0.2070± 0.0015	0.0839± 0.0033	0.0552± 0.0032	0.0844± 0.0025	0.9447± 0.0004	0.3903± 0.0020	0.0446± 0.0020	0.1868± 0.0012	0.1505± 0.0007	0.1697± 0.0004	0.0477± 0.0008	0.0060± 0.0006	0.1839± 0.0005	0.4448± 0.0004	0.0526± 0.0002
SKDB	$k_{\max}=5$	0.5225± 0.0023	0.2021± 0.0021	0.0839± 0.0033	0.0123± 0.0018	0.0376± 0.0026	0.9361± 0.0004	0.3903± 0.0020	0.0446± 0.0020	0.1868± 0.0012	0.1504± 0.0007	0.1697± 0.0004	0.0477± 0.0008	0.0060± 0.0006	0.1839± 0.0005	0.4448± 0.0004	0.0523± 0.0002
	bestk	4	2	5	2	4	3	5	5	4	4	5	5	3.8	5	5	3
	# of atts.	5	19.2	647.5	337.9	275.6	90	54	215.8	5	22	54	37	11	773.6	138	16
SKDB_k (# of attributes selected)	$k=1$	5	7	584.9	354.9	17	80.5	45.8	59	10	11	54	38.2	11	772.2	75	13
	$k=2$	5	19.2	646.5	337.9	87	90	54	206.7	10	12	54	38.1	11	773.1	125.8	14.9
	$k=3$	5	18	648.1	356.1	271.3	90	54	209.6	5	22	54	37.6	11	774.2	138	16
	$k=4$	5	18	640.9	21.3	275.6	78.8	54	214.3	5	22	54	37.9	11	774.2	138	14
	$k=5$	5	18	647.5	18	359.3	81.2	54	215.8	5	22	54	37	11	773.6	138	13
n		5	41	676	361	361	90	54	361	10	67	54	41	11	784	138	141
BayesNet	$k=5$	0.5217± 0.0023	0.2009± 0.0016	0.0920± 0.0023	0.0379± 0.0136	0.0354± 0.0063	0.9369± 0.0004	0.4231± 0.0016	>138G	0.2821± 0.0011	0.1573± 0.0008	0.1615± 0.0006	0.0477± 0.0007	0.0059± 0.0005	>138G	>138G	>138G
Random Forest	100 trees	0.5194± 0.0024	0.1992± 0.0013	0.0338± 0.0010	0.0353± 0.0017	0.0566± 0.0012	>138G	0.3478± 0.0018	0.0888± 0.0006	0.3190± 0.0028	0.1573± 0.0008	0.0962± 0.0012	0.0466± 0.0008	0.0060± 0.0007	>138G	>138G	>138G
Random Forest (Num)	100 trees	0.4199± 0.0017	0.1906± 0.0014	0.0291± 0.0007	0.0262± 0.0016	0.0395± 0.0010	>138G	0.1998± 0.0011	0.0465± 0.0004	0.3190± 0.0028	0.1573± 0.0008	>600h	0.0298± 0.0003	0.0029± 0.0006	>138G	>138G	>138G
$\text{SKDB}^{0\text{-}1}$	$k_{\max}=5$	0.5225± 0.0023	0.2045± 0.0015	0.0839± 0.0033	0.0123± 0.0018	0.0376± 0.0026	0.9447± 0.0003	0.3903± 0.0020	0.0446± 0.0020	0.1868± 0.0012	0.1509± 0.0007	0.1697± 0.0004	0.0477± 0.0008	0.0060± 0.0006	0.1839± 0.0005	0.4448± 0.0004	0.0523± 0.0002
SKDB	MDL disc.	0.5252± 0.0036	0.1923± 0.0025	0.0837± 0.0023	0.0147± 0.0020	0.0370± 0.0026	0.9430± 0.0005	0.2595± 0.0029	0.1810± 0.0048	0.1868± 0.0012	0.1504± 0.0007	0.0377± 0.0029	0.0326± 0.0005	0.0037± 0.0009	0.1449± 0.0007	0.4460± 0.0006	0.0523± 0.0002
m (millions)		0.1649	0.2993	0.3415	0.4741	0.4894	0.5153	0.5811	0.8393	1.025	2.4583	3.8505	5.2095	5.7491	8.1	8.7052	50
c		11	2	9	2	2	90	7	2	10	4	19	40	2	10	24	2

Table 8: Results in terms of RMSE for NB, TAN, AODE, KDB, BayesNet, RF, SKDB_k and SKDB classifiers.

learner	args	localiz.	Census-income	USPS	MITA	MITB	MSDY	covtype	MITC	poker	uscensus	PAMAP	kddcup	linkage	mnist8	satellite	splice
NB		0.5449± 0.0026	0.2410± 0.0017	0.0532± 0.0012	0.0100± 0.0006	0.0199± 0.0004	0.9514± 0.0005	0.3536± 0.0025	0.0582± 0.0010	0.4988± 0.0018	0.0896± 0.0003	0.2365± 0.0007	0.0361± 0.0005	0.0002± 0.0000	0.2521± 0.0005	0.4425± 0.0002	0.0121± 0.0001
TAN		0.4367± 0.0033	0.0675± 0.0016	0.0149± 0.0006	0.0008± 0.0001	0.0161± 0.0005	0.9268± 0.0010	0.3151± 0.0016	0.0455± 0.0008	0.3295± 0.0015	0.0390± 0.0005	0.1171± 0.0005	0.0034± 0.0001	0.0001± 0.0000	0.1327± 0.0007	0.3240± 0.0004	0.0133± 0.0001
AODE		0.433± 0.0027	0.1106± 0.0015	0.0244± 0.0008	0.0104± 0.0007	0.0294± 0.0008	0.9281± 0.0013	0.2859± 0.0016	0.0254± 0.0005	0.4812± 0.0028	0.0532± 0.0004±	0.1654± 0.0007	0.0154± 0.0002	0.0002± 0.0000	0.1271± 0.0004	0.3537± 0.0004	0.0134± 0.0001
KDB	$k=1$	0.4642± 0.0040	0.0667± 0.0014	0.0142± 0.0006	0.0005± 0.0001	0.0183± 0.0009	0.9279± 0.0008	0.2968± 0.0034	0.0406± 0.0009	0.3291± 0.0012	0.0383± 0.0004	0.1209± 0.0006	0.0048± 0.0001	0.0001± 0.0000	0.1500± 0.0004	0.3321± 0.0005	0.0114± 0.0001
	$k=2$	0.3710± 0.0037	0.0562± 0.0013	0.0110± 0.0006	0.0002± 0.0001	0.0043± 0.0002	0.9239± 0.0008	0.2799± 0.0025	0.0088± 0.0003	0.1961± 0.0009	0.0333± 0.0004	0.0850± 0.0003	0.0028± 0.0001	0.0000± 0.0000	0.0919± 0.0001	0.2903± 0.0006	0.0116± 0.0001
	$k=3$	0.3338± 0.0023	0.0556± 0.0010	0.0097± 0.0005	0.0002± 0.0001	0.0023± 0.0002	0.9187± 0.0009	0.2641± 0.0020	0.0053± 0.0003	0.0838± 0.0007	0.0331± 0.0004	0.0609± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0682± 0.0002	0.2613± 0.0003	0.0127± 0.0001
	$k=4$	0.3064± 0.0036	0.0547± 0.0008	0.0087± 0.0005	0.0028± 0.0003	0.0016± 0.0002	0.9131± 0.0012	0.2337± 0.0023	0.0040± 0.0002	0.0486± 0.0007	0.0297± 0.0003	0.0438± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0519± 0.0002	0.2426± 0.0005	0.0145± 0.0001
	$k=5$	0.3064± 0.0036	0.0547± 0.0010	0.0080± 0.0006	0.0188± 0.0010	0.0073± 0.0004	0.9124± 0.0006	0.2077± 0.0023	0.0026± 0.0002	0.0877± 0.0008	0.0313± 0.0002	0.0340± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0104± 0.0001
SKDB_{only_k}	$k_{\max}=5$	0.3064± 0.0036	0.0562± 0.0013	0.0080± 0.0006	0.0028± 0.0003	0.0199± 0.0004	0.9187± 0.0009	0.2077± 0.0023	0.0582± 0.0010	0.0486± 0.0007	0.0297± 0.0003	0.0340± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0029± 0.0000
SKDB_k	$k=1$	0.4642± 0.0040	0.0553± 0.0009	0.0142± 0.0007	0.0005± 0.0001	0.0071± 0.0002	0.9276± 0.0010	0.2961± 0.0021	0.0185± 0.0004	0.3291± 0.0012	0.0293± 0.0004	0.1209± 0.0006	0.0048± 0.0001	0.0001± 0.0000	0.1500± 0.0004	0.3391± 0.0003	0.0029± 0.0000
	$k=2$	0.3710± 0.0037	0.0542± 0.0013	0.0110± 0.0006	0.0002± 0.0000	0.0040± 0.0002	0.9239± 0.0008	0.2799± 0.0025	0.0069± 0.0003	0.1961± 0.0009	0.0271± 0.0003	0.0850± 0.0003	0.0028± 0.0001	0.0000± 0.0000	0.0919± 0.0002	0.2907± 0.0022	0.0029± 0.0000
	$k=3$	0.3338± 0.0023	0.0547± 0.0009	0.0097± 0.0005	0.0002± 0.0001	0.0021± 0.0002	0.9187± 0.0009	0.2641± 0.0020	0.0040± 0.0003	0.0835± 0.0008	0.0268± 0.0003	0.0609± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0682± 0.0002	0.2613± 0.0003	0.0029± 0.0000
	$k=4$	0.3064± 0.0036	0.0532± 0.0008	0.0087± 0.0005	0.0018± 0.0002	0.0015± 0.0002	0.9144± 0.0011	0.2337± 0.0023	0.0030± 0.0002	0.0318± 0.0008	0.0266± 0.0003	0.0438± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0519± 0.0002	0.2426± 0.0005	0.0029± 0.0000
	$k=5$	0.3064± 0.0036	0.0536± 0.0009	0.0080± 0.0006	0.0038± 0.0005	0.0074± 0.0004	0.9135± 0.0008	0.2077± 0.0023	0.0022± 0.0002	0.0318± 0.0008	0.0266± 0.0003	0.0340± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0029± 0.0000
SKDB	$k_{\max}=5$	0.3064± 0.0036	0.0542± 0.0013	0.0080± 0.0006	0.0002± 0.0000	0.0015± 0.0002	0.9187± 0.0009	0.2077± 0.0023	0.0022± 0.0002	0.0318± 0.0008	0.0266± 0.0003	0.0340± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0029± 0.0000
$\text{SKDB}^{0\text{-}1}$	$k_{\max}=5$	0.3064± 0.0036	0.0533± 0.0011	0.0080± 0.0006	0.0002± 0.0000	0.0015± 0.0002	0.9124± 0.0006	0.2077± 0.0023	0.0022± 0.0002	0.0318± 0.0008	0.0261± 0.0002	0.0340± 0.0002	0.0026± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0029± 0.0000
BayesNet	$k=5$	0.3053± 0.0039	0.0533± 0.0012	0.0097± 0.0006	0.0017± 0.0006	0.0014± 0.0004	0.9204± 0.0015	0.2442± 0.0020	>138G	0.0831± 0.0010	0.0288± 0.0003	0.0308± 0.0003	0.0026± 0.0001	0.0000± 0.0000	>138G	>138G	>138G
Random Forest	100 trees	0.3061± 0.0036	0.0527± 0.0009	0.0009± 0.0001	0.0018± 0.0002	0.0031± 0.0003	>138G	0.1157± 0.0013	0.0015± 0.0002	0.0527± 0.0013	0.0266± 0.0003	0.0046± 0.0001	0.0025± 0.0001	0.0000± 0.0000	>138G	>138G	>138G
Random Forest (Num)	100 trees	0.2030± 0.0014	0.0481± 0.0011	0.0005± 0.0001	0.0009± 0.0002	0.0013± 0.0001	>138G	0.0405± 0.0006	0.0004± 0.0000	0.0527± 0.0013	0.0266± 0.0003	>138G	0.0014± 0.0000	0.0000± 0.0000	>138G	>138G	>138G
SKDB	MDL disc.	0.3013± 0.0048	0.0486± 0.0015	0.0078± 0.0005	0.0002± 0.0000	0.0015± 0.0002	0.9129± 0.0006	0.0824± 0.0019	0.0035± 0.0002	0.0318± 0.0008	0.0266± 0.0003	0.0016± 0.0001	0.0017± 0.0001	0.0000± 0.0000	0.0229± 0.0002	0.2175± 0.0005	0.0029± 0.0000
n		5	41	676	361	361	90	54	361	10	67	54	41	11	784	138	141
m (millions)		0.1649	0.2993	0.3415	0.4741	0.4894	0.5153	0.5811	0.8393	1.025	2.4583	3.8505	5.2095	5.7491	8.1	8.7052	50
c		11	2	9	2	2	90	7	2	10	4	19	40	2	10	24	2

Table 9: Results in terms of 0-1 Loss for NB, TAN, AODE, KDB, BayesNet, RF, SKDB_k and SKDB classifiers.

learner	args	localiz.	Census-income	USPS	MITA	MITB	MSDY	covtype	MITC	poker	uscensus	PAMAP	kddcup	linkage	mnist8	satellite	splice	
RMSE	SKDB	$R_{\max=5}$	0.5225± 0.0023	0.2021± 0.0021	0.0839± 0.0033	0.0123± 0.0018	0.0376± 0.0026	0.9361± 0.0004	0.3903± 0.0020	0.0446± 0.0020	0.1868± 0.0012	0.1504± 0.0007	0.1697± 0.0004	0.0477± 0.0008	0.0060± 0.0006	0.4448± 0.0007	0.0523± 0.0004	0.0002
	VW	-p3	0.6249± 0.0006	0.2709± 0.0048	0.2156± 0.0011	0.0326± 0.0076	0.0696± 0.0069	0.7101± 0.0001	0.5227± 0.0010	0.1361± 0.0013	0.5355± 0.0006	0.2038± 0.0014	0.5309± 0.0006	0.1240± 0.0020	0.0184± 0.0038	0.4176± 0.0009	0.5107± 0.0006	0.2085± 0.0045
		-p20	0.6243± 0.0006	0.2662± 0.0028	0.2157± 0.0008	0.0266± 0.0035	0.0714± 0.0047	0.7093± 0.0001	0.5116± 0.0009	0.1366± 0.0008	0.5355± 0.0005	0.2018± 0.0011	0.5317± 0.0006	0.1144± 0.0025	0.0151± 0.0032	0.4173± 0.0008	0.5024± 0.0006	0.2115± 0.0038
		-p3 -c	0.6217± 0.0595± 0.0009	0.3681± 0.3426± 0.0080	0.1894± 0.0402± 0.0006	0.0319± 0.0277± 0.0058	0.0453± 0.0403± 0.0058	0.7088± 0.7276± 0.0005	0.5177± 0.5018± 0.0018	0.1232± 0.0518± 0.0042	0.5342± 0.3679± 0.0019	0.2023± 0.1938± 0.0040	0.4689± 0.3633± 0.0010	0.1123± 0.2426± 0.0080	0.0079± 0.0103± 0.0029	0.4173± 0.2884± 0.0063	0.5031± 0.4505± 0.0015	0.1964± 0.0855± 0.0082
		-p3 -q	0.6217± 0.0006	0.3681± 0.0047	0.1894± 0.0014	0.0319± 0.0358)	0.0453± 0.0065	0.7088± 0.0001	0.5177± 0.0012	0.1232± 0.0022	0.5342± 0.0006	0.2023± 0.0015	0.4689± 0.0007	0.1123± 0.0026	0.0079± 0.0017	0.4173± 0.0009	0.5031± 0.0007	0.1964± 0.0045
		-p3 -q -l	0.7357± 0.0021	0.1896± 0.0020	0.0190± 0.0074	0.0238± 0.0021*	0.0303± 0.0016*	0 ** 0.0001	0.5467± 0.0020	0.0308± 0.0016*	0.3545± 0.0018	0.1482± 0.0009*	0.3920± 0.0017*	0.0519± 0.0007	0.0043± 0.0006	0.2773± 0.0039	>400h	>400h
0-1 Loss	$SKDB_{0\text{-}1}$	$R_{\max=5}$	0.3064± 0.0036	0.0533± 0.0011	0.0080± 0.0006	0.0015± 0.0000	0.0015± 0.0002	0.9124± 0.0006	0.2077± 0.0023	0.0022± 0.0002	0.0318± 0.0008	0.0261± 0.0003	0.0340± 0.0002	0.0036± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0029± 0.0000
	SKDB	$R_{\max=5}$	0.3064± 0.0036	0.0542± 0.0013	0.0080± 0.0006	0.0002± 0.0000	0.0015± 0.0009	0.9187± 0.0009	0.2077± 0.0023	0.0022± 0.0002	0.0318± 0.0008	0.0266± 0.0003	0.0340± 0.0002	0.0036± 0.0001	0.0000± 0.0000	0.0378± 0.0002	0.2284± 0.0004	0.0029± 0.0000
	VW	-p3	0.6056± 0.0041	0.0496± 0.0012	0.0311± 0.0011	0.0002± 0.0000	0.0006± 0.0003	0.9192± 0.0012	0.3688± 0.0029	0.0117± 0.0006	0.4989± 0.0015	0.0290± 0.0003	0.3488± 0.0013	0.0040± 0.0002	0.0000± 0.0000	0.1344± 0.0006	0.3053± 0.0013	0.0029± 0.0000
		-p20	0.6025± 0.0036	0.0490± 0.0011	0.0307± 0.0009	0.0002± 0.0000	0.0028± 0.0003	0.9173± 0.0013	0.3330± 0.0023	0.0115± 0.0004	0.4988± 0.0015	0.0283± 0.0003	0.3316± 0.0009	0.0035± 0.0001	0.0000± 0.0000	0.1341± 0.0006	0.2970± 0.0010	0.0029± 0
		-p3 -q	0.5524± 0.0050	0.0502± 0.0020	0.0000± 0.0000	0.0003± 0.0001	0.0003± 0.0004	0.9198± 0.0012	0.3313± 0.0032	0.0007± 0.0003	0.0740± 0.0007	0.0255± 0.0004	0.0980± 0.0006	0.0020± 0.0001	0.0000± 0.0000	0.0440± 0.0010	0.2341± 0.0015	0.0027± 0.0001
		-p3 -c	0.5932± 0.0039	0.0493± 0.0011	0.0193± 0.0009	0.0005± 0.0002	0.0028± 0.0005	0.9162± 0.0013	0.3619± 0.0029	0.0091± 0.0005	0.5015± 0.0026	0.0285± 0.0003	0.2286± 0.0011	0.0024± 0.0001	0.0000± 0.0000	0.1342± 0.0006	0.2947± 0.0010	0.0029± 0.0000
		-p3 -q -l	0.5452± 0.0051	0.0475± 0.0011	0.0004± 0.0005	0.0003± 0.0001*	0.0028± 0.0005	0.9205± 0.0013	0.3318± 0.0030	0.0009± 0.0001*	0.0746± 0.0008	0.0250± 0.0003*	0.1328± 0.0005*	0.0030± 0.0001	0.0000± 0.0000	0.0565± 0.0005	>400h	>400h
0-1 Loss		m	0.1649	0.2993	0.3415	0.4741	0.4894	0.5153	0.5811	0.8393	1.025	2.4583	3.8505	5.2095	5.7491	8.1	8.7053	24
		c	11	2	9	2	2	90	7	2	10	4	19	40	2	10	24	2

Table 10: Results in terms of RMSE and 0-1 loss for SKDB and VW with different options: -p indicates the number of passes (3 or 20), -q indicates quadratic features (all possible combinations), -c refers to cubic features (all possible combinations), and -l the use of a logistic function for optimization. $SKDB_{0\text{-}1}$ is SKDB optimized for 0-1 loss.

* Results shown for 1 experiment only due to time limitations when using the logistic function in VW.

** Results shown for 1 fold only due to time limitations when using the logistic function in VW.

where π_{x_i} are the parent-values of X_i and $m = 1$. Secondly, if zero counts are found, we back off as many levels in the tree as necessary to find at least one count, that is, if $counts(x_4, x_0, x_3)$ is equal to zero, then $\hat{p}(x_4|x_0)$ is considered instead of $\hat{p}(x_4|x_0, x_3)$.

- In principle, any loss function that can be computed incrementally is suitable for SKDB. We have chosen RMSE (as defined above) for our first set of experiments.
- We use Averaged One Dependence Estimators (AODE) as a representative of the ANDE family of algorithms where $n = 1$.

  In order to avoid an extensive number of passes on each of the datasets, no dataset-tailored parameter tuning has been carried out for any of the algorithms considered. However as detailed in Sections 3.2, for logistic regression we have pre-conducted experiments with varying numbers of passes for stochastic gradient descent, varying loss functions and combinations of features; and used the default advanced (step size) updates provided by vowpal wabbit. In the case of random forest, we have made experiments with and without pre-discretising continuous attributes, as well as learning a variable number of trees (see Section 3.3.2 for more details).

Appendix C. Detailed Analysis of Results

The upper part of Table 8 (in the Appendix) shows the results in terms of RMSE ± standard deviation on the 16 datasets described above. We have decided to stop at $k = 5$, since some of the datasets already start to show worse results for $k = 5$ (due to variance increase), so increasing the complexity further is not productive. The results for NB, TAN and AODE are displayed as baseline orientations. The dark gray background on certain outputs for $SKDB_k$, corresponds to experiments that outperform KDB for the same value of k; the light grey background on others indicate a tie; whereas the absence of the two indicates that plain KDB outperforms $SKDB_k$. As for SKDB, the background coloring has the same meaning, but each entry is compared with the best KDB value for all possible k. Note that only the average of the 10 folds is shown for each experiment, and the background colors indicate statistical difference (Wilcoxon signed-rank tests), dark grey or white (in favor or against the algorithm in the row respectively); or lack of statistical difference, in light grey.

Table 8 also shows the number of selected attributes by $SKDB_k$. There are a few cases for which the number of selected attributes for all values of k is the total number of attributes in the datasets, such as `localization`, `PAMAP` and `linkage`. For some other datasets, for example `MSDYearPrediction`, `USPSExtended` or `mnist8ms`, the performance between selected KDB and KDB is similar, but fewer attributes are considered, so classification time (and space) will be reduced. In most of the cases the number of selected attributes is smaller than the original, sometimes even less than half, reducing the RMSE compared to KDB, for example `MITFaceSetC` with 215.8 attributes selected (out of 361) results in 0.0446 vs 0.0494 for KDB ($k = 5$), or `uscensus` with 22 attributes selected (out of 67) results in 0.1504 vs 0.1601 for KDB ($k = 4$), that in both cases results in statistical improvement.

The results for SKDB using MDL discretization are also shown in Table 8 (antepenultimate row). Note that there are some datasets, such as `linkage`, for which the discretization method used, in this case EF5, is not an adequate one; whereas a technique based on entropy minimization provides better results. Just the opposite situation occurs for datasets

such as MITFaceSetC. Apriori selection of the most appropriate discretization method is an open question, even more for very large datasets where we should consider the same bias/variance trade-off in terms of discretization bias and variance (Yang and Webb, 2009), that is, on the one hand we want that the number of intervals is sufficiently large to provide low discretization bias, but not too large to avoid an increase in discretization variance. The latter should not be a problem in large datasets, but considering a large number of intervals implies bigger datasets, which will have an impact on the classifier complexity.

References

Alekh Agarwal, Olivier Chapelle, Miroslav Dudík, and John Langford. A Reliable Effective Terascale Linear Learning System. *J. Mach. Learn. Res.*, 15(1):1111–1133, January 2014. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=2627435.2638571`.

Kevin Bache and Moshe Lichman. UCI Machine Learning Repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

Jock A. Blackard. *Comparison of Neural Networks and Discriminant Analysis in Predicting Forest Cover Types*. PhD thesis, Fort Collins, CO, USA, 1998. AAI9921979.

Léon Bottou, Corinna Cortes, John S. Denker, Harris Drucker, Isabelle Guyon, Lawrence D. Jackel, Yann Le Cun, Urs A. Muller, Eduard Säckinger, Patrice Simard, and Vladimir Vapnik. Comparison of Classifier Methods: A Case Study in Handwritten Digit Recognition. In *Proceedings of the 12th International Conference on Pattern Recognition, Los Alamitos, CA.*, volume 2, pages 77–82, 1994.

Remco R. Bouckaert. Voting Massive Collections of Bayesian Network Classifiers for Data Streams. In *Proceedings of the 19th Australian joint conference on Artificial Intelligence: advances in Artificial Intelligence*, AI'06, pages 243–252, Berlin, Heidelberg, 2006. Springer-Verlag.

Damien Brain and Geoffrey I. Webb. On The Effect of Data Set Size on Bias and Variance in Classification Learning. In D. Richards, G. Beydoun, A. Hoffmann, and P. Compton, editors, *Proceedings of the Fourth Australian Knowledge Acquisition Workshop (AKAW '99)*, pages 117–128, Sydney, 1999. The University of New South Wales.

Leo Breiman. Bagging Predictors. *Mach. Learn.*, 24(2):123–140, 1996.

Leo Breiman. Random Forests. *Mach. Learn.*, 45(1):5–32, 2001.

Glenn W. Brier. Verification of Forecasts Expressed in Terms of Probability. *Monthly Weather Review*, 75:1–3, 1950.

Alexandra M. Carvalho, Teemu Roos, Arlindo L. Oliveira, and Petri Myllymäki. Discriminative Learning of Bayesian Networks via Factorized Conditional Log-Likelihood. *J. Mach. Learn. Res.*, 12:2181–2210, 2011.

Robert Cattral, Franz Oppacher, and Dwight Deugo. Evolutionary Data Mining with Automatic Rule Generalization. In *Recent Advances in Computers, Computing and Communications*, pages 296–300, 2002.

C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

Gregory F. Cooper and Edward Herskovits. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Mach. Learn.*, 9(4):309–347, 1992.

Janez Demšar. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.*, 7:1–30, 2006.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011.

Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc, 1 edition, 1973.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why Does Unsupervised Pre-training Help Deep Learning? *J. Mach. Learn. Res.*, 11:625–660, 2010.

Usama M. Fayyad and Keki B. Irani. Multi-interval Discretization of Continuous-valued Attributes for Classification Learning. In Ruzena Bajcsy, editor, *IJCAI*, pages 1022–1029. Morgan Kaufmann, 1993.

M. Julia Flores, José A. Gámez, Ana M. Martínez, and José M. Puerta. GAODE and HAODE: Two Proposals Based on AODE to Deal with Continuous Variables. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 313–320. ACM, 2009a.

M. Julia Flores, Josi A. Gamez, Ana M. Martmnez, and Jose Miguel Puerta. HODE: Hidden One-Dependence Estimator. In Claudio Sossai and Gaetano Chemello, editors, *ECSQARU*, volume 5590 of *Lecture Notes in Computer Science*, pages 481–492. Springer, 2009b. ISBN 978-3-642-02905-9.

Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Mach. Learn.*, 29(2-3):131–163, 1997.

João Gama, Ricardo Rocha, and Pedro Medas. Accurate Decision Trees for Mining High-speed Data Streams. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 523–528, New York, NY, USA, 2003. ACM.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: an Update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

G. Hardy. Correspondence. Insurance Record (1889). *Transactions of the Faculty Actuaries*, 8, 1920.

Geoff Hulten and Pedro Domingos. Mining Complex Models from Arbitrarily Large Databases in Constant Time. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 525–531, New York, NY, USA, 2002. ACM.

Manfred Jaeger. Probabilistic Classifiers and the Concepts They Recognize. In Tom Fawcett and Nina Mishra, editors, *ICML*, pages 266–273. AAAI Press, 2003.

Boštjan Kaluža, Violeta Mirchevska, Erik Dovgan, Mitja Luštrek, and Matjaž Gams. An Agent-based Approach to Care in Independent Living. In *Proceedings of the First international joint conference on Ambient intelligence*, AmI'10, pages 177–186, Berlin, Heidelberg, 2010. Springer-Verlag.

Nikos Karampatziakis and John Langford. Online Importance Weight Aware Updates. In Fabio Gagliardi Cozman and Avi Pfeffer, editors, *UAI*, pages 392–399. AUAI Press, 2011.

Ron Kohavi. The Power of Decision Tables. In Nada Lavrac and Stefan Wrobel, editors, *Proceedings of the European Conference on Machine Learning ECML-95*, volume 912 of *Lecture Notes in Computer Science*, pages 174–189. Springer Berlin Heidelberg, 1995.

Daphne Koller and Mehran Sahami. Toward Optimal Feature Selection. In Lorenza Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning (ICML)*, pages 284–292. Morgan Kaufmann Publishers, 1996.

Pat Langley and Stephanie Sage. Induction of Selective Bayesian Classifiers. In *Proceedings of the Tenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 399–406, San Francisco, CA, 1994. Morgan Kaufmann.

David D. Lewis. Naive Bayes at Forty: The Independence Assumption in Information Retrieval. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 4–15, London, UK, UK, 1998. Springer-Verlag.

George James Lidstone. Note on the General Case of the Bayes-Laplace Formula for Inductive or a Posteriori Probabilities. *Transactions of the Faculty Actuaries*, 8:182–192, 1920.

Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training Invariant Support Vector Machines using Selective Sampling. In Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors, *Large Scale Kernel Machines*, pages 301–320. MIT Press, Cambridge, MA., 2007.

Oded Maron and Andrew Moore. Hoeffding Races: Accelerating Model Selection Search for Classification and Function Approximation. In Jack D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 59–66, San Francisco, CA, 1994. Morgan Kaufmann.

B.W. Matthews. Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure* , 405(2):442 – 451, 1975.

H. Brendan McMahan and Matthew Streeter. Adaptive Bound Optimization for Online Convex Optimization. In *Proceedings of the 23rd Annual Conference on Learning Theory (COLT)*, 2010.

Marvin Minsky. Steps Toward Artificial Intelligence. *Proceedings of the Institute of Radio Engineers*, 49:8–30, 1961.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

Andrew Moore and Mary Soon Lee. Cached Sufficient Statistics for Efficient Machine Learning with Large Datasets. *J. Artif. Int. Res.*, 8(1):67–91, 1998.

Nikunj C. Oza and Stuart Russell. Experimental Comparisons of Online and Batch Versions of Bagging and Boosting. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 359–364, New York, NY, USA, 2001. ACM.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Franz Pernkopf and Jeff A. Bilmes. Efficient Heuristics for Discriminative Structure Learning of Bayesian Network Classifiers. *J. Mach. Learn. Res.*, 11:2323–2360, 2010.

François Petitjean, Jordi Inglada, and Pierre Gançarski. Satellite Image Time Series Analysis under Time Warping. *IEEE Transactions on Geoscience and Remote Sensing*, 50(8): 3081–3095, 2012.

Attila Reiss and Didier Stricker. Creating and Benchmarking a New Dataset for Physical Activity Monitoring. In *Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '12, pages 40:1–40:8, New York, NY, USA, 2012. ACM.

Stéphane Ross, Paul Mineiro, and John Langford. Normalized Online Learning. arXiv preprint arXiv:1305.6646, 2013.

Arcadio Rubio and José Antonio Gámez. Flexible Learning of K-dependence Bayesian Network Classifiers. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 1219–1226, New York, NY, USA, 2011. ACM.

Mehran Sahami. Learning Limited Dependence Bayesian Classifiers. In *In KDD-96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 335–338. AAAI Press, 1996.

Murat Sariyar, Andreas Borg, and Klaus Pommerening. Controlling False Match Rates in Record Linkage Using Extreme Value Theory. *J. of Biomedical Informatics*, 44(4): 648–654, 2011.

I Schmidtmann, G Hammer, M Sariyar, and A. Gerhold-Ay. Evaluation des Krebsregisters NRW - Schwerpunkt Record Linkage - Abschlussbericht. Technical report, Institut für medizinische Biometrie, Epidemiologie und Informatik, Universitätsmedizin Mainz, 2009.

Sören Sonnenburg and Vojtech Franc. COFFIN : A Computational Framework for Linear SVMs. In *Proc. ICML 2010*, 2010.

Jiang Su and Harry Zhang. Full Bayesian Network Classifiers. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 897–904, New York, NY, USA, 2006. ACM.

Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. Core Vector Machines: Fast SVM Training on Very Large Data Sets. *J. Mach. Learn. Res.*, 6:363–392, 2005.

Geoffrey I. Webb, Janice R. Boughton, and Zhihai Wang. Not So Naive Bayes: Aggregating One-Dependence Estimators. *Mach. Learn.*, 58(1):5–24, 2005.

Geoffrey I. Webb, Janice R. Boughton, Fei Zheng, Kai Ming Ting, and Houssam Salem. Learning by Extrapolation from Marginal to Full-multivariate Probability Distributions: Decreasingly Naive Bayesian Classification. *Mach. Learn.*, 86(2):233–272, 2012.

Y. Yang, G.I. Webb, J. Cerquides, K. Korb, J. Boughton, and K-M. Ting. To Select or To Weigh: A Comparative Study of Linear Combination Schemes for SuperParent-One-Dependence Estimators. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 19(12):1652–1665, 2007.

Ying Yang and Geoffrey I. Webb. Discretization for Naive-Bayes Learning: Managing Discretization Bias and Variance. *Mach. Learn.*, 74(1):39–74, 2009.

Nayyar A. Zaidi, Jesús Cerquides, Mark James Carman, and Geoffrey I. Webb. Alleviating Naive Bayes Attribute Independence Assumption by Attribute Weighting. *J. Mach. Learn. Res.*, 14(1):1947–1988, 2013.

Harry Zhang, Liangxiao Jiang, and Jiang Su. Hidden Naive Bayes. In *Proceedings of the 20th National Conference on Artificial intelligence - Volume 2*, pages 919–924. AAAI Press, 2005.

F. Zheng, G.I. Webb, P. Suraweera, and L. Zhu. Subsumption Resolution: An Efficient and Effective Technique for Semi-Naive Bayesian Learning. *Mach. Learn.*, 87(1):93–125, 2012.

Fei Zheng and Geoffrey I Webb. A Comparative Study of Semi-naive Bayes Methods in Classification Learning. In *Proceedings of the Fourth Australasian Data Mining Conference (AusDM05)*, pages 141–156, 2005.

Zijian Zheng and Geoffrey I. Webb. Lazy Learning of Bayesian Rules. *Mach. Learn.*, 41(1): 53–84, 2000.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 8/11; Revised 8/15; Published 4/16

A Unified View on Multi-class Support Vector Classification

Ürün Doğan udogan@microsoft.com
Microsoft Research

Tobias Glasmachers tobias.glasmachers@ini.rub.de
Institut für Neuroinformatik
Ruhr-Universität Bochum, Germany

Christian Igel igel@diku.dk
Department of Computer Science
University of Copenhagen, Denmark

Editor: Ingo Steinwart

Abstract

A unified view on multi-class support vector machines (SVMs) is presented, covering most prominent variants including the one-vs-all approach and the algorithms proposed by Weston & Watkins, Crammer & Singer, Lee, Lin, & Wahba, and Liu & Yuan. The unification leads to a template for the quadratic training problems and new multi-class SVM formulations. Within our framework, we provide a comparative analysis of the various notions of multi-class margin and margin-based loss. In particular, we demonstrate limitations of the loss function considered, for instance, in the Crammer & Singer machine.

We analyze Fisher consistency of multi-class loss functions and universal consistency of the various machines. On the one hand, we give examples of SVMs that are, in a particular hyperparameter regime, universally consistent without being based on a Fisher consistent loss. These include the canonical extension of SVMs to multiple classes as proposed by Weston & Watkins and Vapnik as well as the one-vs-all approach. On the other hand, it is demonstrated that machines based on Fisher consistent loss functions can fail to identify proper decision boundaries in low-dimensional feature spaces.

We compared the performance of nine different multi-class SVMs in a thorough empirical study. Our results suggest to use the Weston & Watkins SVM, which can be trained comparatively fast and gives good accuracies on benchmark functions. If training time is a major concern, the one-vs-all approach is the method of choice.

Keywords: support vector machines, multi-class classification, consistency

1. Introduction

Support vector machines (SVMs, Boser et al., 1992; Cortes and Vapnik, 1995) are founded on the intuitive geometric concepts of large margin separation and regularized risk minimization, and they are well embedded into statistical learning theory. However, there is no unique canonical extension to the case of multiple classes, neither from a geometric nor from a learning theoretical point of view. Instead, several variants relying on slightly different notions of margin and margin-based loss have been proposed. We present a unified view on multi-class SVMs. This view allows to identify similarities and differences between existing approaches. It provides a thorough framework for analyzing, designing, and efficiently training multi-class SVMs.

There exist generic ways of doing multi-category classification based on arbitrary binary learning machines by combining hypotheses from independently trained binary classifiers. In contrast, the focus of this study is on multi-class extensions of SVMs that aim at generalizing the notions of margin and large margin loss. These so-called all-in-one machines cast the learning problem into a single quadratic program, which allows them to take all class relationships into account simultaneously. The first of these all-in-one formulations was independently proposed by Weston and Watkins (1999), Vapnik (1998), and Bredensteiner and Bennett (1999). Crammer and Singer (2002) modified this machine, and their approach is frequently used, in particular when dealing with structured output. In addition to these popular methods, we consider the machine by Lee et al. (2004), a variant proposed by Liu and Yuan (2011), as well as multi-class maximum margin regression (Szedmak et al., 2006). The latter is equivalent to a multi-class SVM suggested by Zou et al. (2008). From a geometric point of view, the various approaches differ in the way they extend the concepts of margin and margin violation (or, to be more precise, margin-based loss) to multiple classes. Differentiating the machines form a learning-theoretical perspective is more difficult. Albeit there have been extensions of generalization bounds derived for binary SVMs to the multi-class case (e.g., Guermeur, 2007; Doğan et al., 2012), these do not allow to deduce relative advantages or disadvantages of certain multi-class SVM formulations. But there are hints from theoretical analyzes of consistency. In particular, the loss function proposed by Lee et al. was the first multi-class large margin loss known to be Fisher consistent (Lee et al., 2004; Tewari and Bartlett, 2007; Liu, 2007). The only other loss functions we are aware of sharing this property are those proposed in the work of Liu and Yuan (2011). In practice, however, the decision of which multi-class SVM to use is most often not governed by theoretical arguments, but by training-time considerations and the availability of efficient implementations. For instance, the canonical extension of binary SVMs to multiple classes proposed by Weston and Watkins and the machine by Lee et al. are rarely used. These approaches are theoretically sound and experiments indicate that they lead to well-generalizing hypotheses, but efficient training algorithms have been lacking so far.

Against this background, we analyze large-margin multi-category classification theoretically and derive fast training algorithms for established as well as for new learning machines, which we then evaluate empirically. Several researchers investigated multi-class SVM classification methods before, either experimentally (Hsu and Lin, 2002; Rifkin and Klautau, 2004) or conceptually (Allwein et al., 2001; Hill and Doucet, 2007; Liu, 2007; Guermeur, 2007). The experimental papers by Hsu and Lin (2002) and Rifkin and Klautau (2004) focus on the empirical comparison of methods. Some well established multi-class SVMs are missing in these studies. This also holds true for the conceptually oriented work by Allwein et al. (2001). The inspiring theoretical studies by Hill and Doucet (2007) and Liu (2007) completely lack any empirical comparison. The work most closely related to ours is perhaps the framework proposed by Hill and Doucet. They analyze several multi-class SVMs and present a solver for machine training, however, our study goes beyond their important achievements in several aspects. While Hill and Doucet's framework unifies already existing methods, one can neither easily develop new machines within the framework nor clearly identify the conceptual differences between machines. Further, their analysis

does not provide hints on how loss functions may affect the generalization performance of different machines. Finally, it is important to provide an extensive and unbiased empirical comparison.

This article is organized as follows. The next section presents our unifying framework, which categorizes multi-class SVMs according to their choice of multi-class loss function, margin function, and aggregation operator in subsections 2.1–2.3. Subsection 2.4 shows how the existing multi-class machines fit into our scheme. Then we derive new SVM variants suggested by the framework. Subsection 2.6 states a template for quadratic programs describing the learning problems induced by machines falling into our framework. Section 3 analyzes how different design choices affect the Fisher and universal consistency of the loss functions and the classifiers, respectively. The section closes with showcase experiments on synthetic test problems designed to highlight consequences of certain design choices. Section 4 presents an extensive experimental evaluation of various SVM variants, both with linear as well as Gaussian kernels. The theoretical as well as empirical findings finally lead us to conclusions including recommendations for the use of multi-class SVMs in practice.

2. A Framework for Multi-category SVM Classification

Multi-class SVMs construct hypotheses $h : X \to Y$ from training data $\big((x_1, y_1), \ldots, (x_\ell, y_\ell)\big) \in \big(X \times Y\big)^\ell$, where X and Y are the input and label space, respectively. The data points are assumed to be sampled i.i.d. from a distribution P on $X \times Y$. We restrict our considerations to the standard case of a finite label space and set $Y = \{1, \ldots, d\}$ (there exist extensions of multi-class SVMs to infinite label spaces, e.g., Bordes et al. e.g., 2008). The goal of training a multi-category SVM is to map the training data to a hypothesis minimizing the risk (generalization error) $\mathcal{R}(h) = \int\limits_{X \times Y} L(h(x), y) \ \mathrm{d}P(x, y)$, where the loss function L is typically the 0-1-loss, which is zero if both arguments are identical and one otherwise.

For multi-category classification the binary SVM concept of thresholding a real-valued decision function $f : X \to \mathbb{R}$ at zero is insufficient. The canonical extension to $d > 2$ classes is to employ a vector-valued linear decision function $f : X \to \mathbb{R}^d$, where the maximal component indicates the decision. Using a kernel-induced feature space, the corresponding hypothesis takes the form

$$h : X \to Y = \{1, \ldots, d\}\,; \quad x \mapsto \arg\max_{c \in Y} \big\{f_c(x)\big\} = \arg\max_{c \in Y} \big\{\langle w_c, \phi(x)\rangle + b_c\big\} \ , \qquad (1)$$

where $\phi : X \to \mathcal{H}$ is a feature map into an inner product space $\mathcal{H}$, $w_1, \ldots, w_d \in \mathcal{H}$ are class-wise weight vectors, and $b_1, \ldots, b_d \in \mathbb{R}$ are class-wise bias/offset values. The feature map is defined by a positive definite kernel function $k : X \times X \to \mathbb{R}$ with the property $k(x, x') = \langle \phi(x), \phi(x')\rangle$. We presume that the arg max operator in Equation (1) returns a single class index (ties may, for example, be broken at random). To ease the notation, we define the risk $\mathcal{R}(f)$ to be equal to the corresponding $\mathcal{R}(h)$.

Adding the same function $g : X \to \mathbb{R}$ to all components f_c of the decision function does not change the decision as $\arg\max_{c \in Y} \Big\{f_c(x)\Big\} = \arg\max_{c \in Y} \Big\{f_c(x) + g(x)\Big\}$. This is often undesired, for example, for the sake of uniqueness of a solution. This problem can be fixed

by enforcing the so-called *sum-to-zero* constraint

$$\sum_{c=1}^{d} f_c(x) = 0 \ . \qquad (2)$$

At least for universal kernels (Steinwart, 2002a) the constraint is equivalent to $\sum_{c=1}^{d} w_c = 0$ and $\sum_{c=1}^{d} b_c = 0$. With this constraint we have $f_2 = -f_1$ for the special case of binary classification, and the hypothesis $h(x) = \mathrm{sign}(f_1(x))$ maps the first class to $+1$ and the second to -1. Thus, the binary SVM can be recovered as a special case of the multi-class SVM, which is an important design criterion.

2.1 Multi-class Loss Functions

Support vector machines select a hypothesis by minimizing the regularized empirical risk functional

$$\frac{1}{2}\|f\|^2 + C \cdot \sum_{i=1}^{\ell} L(f(x_i), y_i) \ , \qquad (3)$$

where L is a large-margin loss function and $C > 0$ is a user-defined regularization constant. The squared norm is defined as $\|f\|^2 = \sum_{c \in Y} \|f_c\|^2 = \sum_{c \in Y} \|w_c\|^2$. Most existing approaches to multi-class large margin classification differ only in the type of loss function, which should typically be an upper bound on the 0-1-loss, lead to an optimization problem that can be solved efficiently, and vanish on an open set to allow for sparse solutions. The key observation for our unifying framework is that multi-class SVM loss functions can be decomposed into meaningful components, namely a set of margin functions for the different classes, a large-margin loss for binary problems, and an aggregation operator, combining the various target margin violations into a single loss value. This decomposition is found in all existing approaches to large margin multi-class classification. Given a labeled point $(x, y) \in X \times Y$ and the value $f(x) \in \mathbb{R}^d$ of the decision function, the general form of most multi-class loss functions is

$$L\big(f(x), y\big) = \Delta\Big(\widehat{L}^{\mathrm{bin}}\big(\mu(f(x), y), y\big)\Big) \ ,$$

or conceptually "$L = \Delta \circ \widehat{L}^{\mathrm{bin}} \circ \mu$". Here μ is a *margin function*, Δ is an *aggregation operator*, and L^{bin} is a loss function for binary classification, such as the hinge loss $L^{\mathrm{hinge}}(\mu) = \max\{0, 1 - \mu\}$, the squared hinge loss, or the Huber loss. With $\widehat{L}^{\mathrm{bin}}$ we denote the component-wise application of this loss function to the margin values.

In the next two subsections, we formalize the concepts of margin functions and aggregation operators and give examples of how existing machines can be expressed in our framework.

2.2 Margin Functions

The following definition extends the concept of a scalar margin of a training example to a vector-valued margin function.

Definition 1 (margin function) *A* margin function *is a non-constant function*

$$\mu : \mathbb{R}^d \times Y \to \mathbb{R}^d; \quad \mu = (\mu_1, \ldots, \mu_d)$$

of a prediction $f(x) \in \mathbb{R}^d$ *and a label* $y \in Y$ *that is linear in its first argument and fulfills the properties:*

1. $\forall y \in Y : \mu_y(f(x), y)$ *is non-decreasing in* $f_y(x)$,
2. $\forall c, y \in Y, c \neq y : \mu_c(f(x), y)$ *is non-increasing in* $f_c(x)$,
3. $\Big[\forall c \in Y : \mu_c(f(x), y) \geq 0$ *and* $\exists c \in Y : \mu_c(f(x), y) > 0\Big] \Rightarrow \big[\arg\max_{c \in Y} \big\{f_c(x)\big\} = y\big]$

Due to linearity in $f(x)$ margin functions are of the form

$$\mu_c\big(f(x), y\big) = \sum_{m=1}^{d} \nu_{y,c,m} \cdot f_m(x_i)$$

and can thus be expressed in terms of coefficients $\nu_{y,c,m}$. Property 3 in the above definition ensures that positive margins imply correct classification. Note that the reverse statement is not guaranteed, so, correct classification does not necessarily imply that all margin functions are non-negative. The definition is not as general as it may seem. If we assume an equal treatment of all classes, then the degrees of freedom in $\nu_{y,c,m}$ are drastically reduced.

The following types of margins are of primary interest:

Definition 2 (relative and absolute margins) *The* relative margin function *is given by*

$$\mu_c^{rel}(f(x), y) = \frac{1}{2}\big(f_y(x) - f_c(x)\big)$$

and the absolute margin function *by*

$$\mu_c^{abs}(f(x), y) = \begin{cases} +f_c(x) & \text{for } c = y \\ -f_c(x) & \text{for } c \in Y \setminus \{y\} \end{cases}.$$

The corresponding coefficients are

$$\nu_{y,c,m} = \frac{1}{2}\left(\delta_{y,m} - \delta_{c,m}\right) \qquad \text{(relative)}$$

and

$$\nu_{y,c,m} = \delta_{m,c} \cdot (2\delta_{c,y} - 1) = -(-1)^{\delta_{c,y}} \cdot \delta_{m,c} \qquad \text{(absolute)}$$

for the relative and the absolute margin, respectively, where $\delta_{a,b} = 1$ if $a = b$ and $\delta_{a,b} = 0$ otherwise denotes the Kronecker symbol. Both margin functions can be combined, as in the work of Liu and Yuan (2011). They have a number of interesting properties:

- The coefficients $\nu_{y,c,m}$ are sparse, so that the computation of a margin value μ_c from $f(x)$ is a constant time operation (independent of the number of classes).

- The "own-class-margin" component $\mu_y^{\text{rel}}(f(x), y)$ of relative margins is always zero. For a two-class problem, the other component coincides with the margin $y \cdot f(x)$ of the binary SVM (with labels $y \in \{\pm 1\}$ and real-valued decision function f).
- If the classification according to Equation (1) is correct, then no component of the relative margin function is negative. This implication does not necessarily hold for absolute margin functions regardless of whether the sum-to-zero constraint is enforced or not.
- It holds $\sum_m \nu_{y,c,m} = 0$ for relative but not for absolute margins. Adding the same function to all components f_c of the decision function does not change the decision (1). Hence it is reasonable to demand that adding the same value to all $\nu_{y,c,m}$ does not change the margins. This property is violated for absolute margins. This renders both the sign and the absolute value of a single margin (i.e., a component of the margin function) meaningless. This issue could be addressed by enforcing $\sum_m \nu_{y,c,m} = 0$ also for absolute margins.
- In principle, the sum-to-zero constraint (2) can be avoided at the level of the decision function by shifting the coefficients ν such that they fulfill $\sum_m \nu_{y,c,m} = 0$. However, this would destroy the sparsity of the coefficients. Thus, it can be argued that the sum-to-zero constraint $\sum_c f_c = 0$ is a preferable solution.
- A regularizer of the form $\sum_{c=1}^{d} \|w_c\|^2$ has the effect that the optimal solution of a relative margin SVM always fulfills the sum-to-zero constraint. This has been observed for particular machines by Guermeur (2007) and by Wu and Liu (2007), but can be shown to be true for all relative margin SVMs. Adding the same vector w to all weight vectors w_c (or the same function $g(x) = \langle w, \phi(x)\rangle$ to all components f_c) does not affect relative margins and thus the empirical error term in the primal problem. Thus, to find the optimal w we have to minimize the complexity term

$$\sum_{c=1}^{d} \langle w_c + w, w_c + w\rangle = \sum_{c=1}^{d} \langle w_c, w_c\rangle + 2 \cdot \sum_{c=1}^{d} \langle w_c, w\rangle + d \cdot \langle w, w\rangle$$

  w.r.t. w. Setting the derivative $2 \cdot \left(\sum_{c=1}^{d} w_c + d \cdot w\right)$ to zero gives $w = -\frac{1}{d}\sum_{c=1}^{d} w_c$, which results in $\sum_{c=1}^{d}(w_c + w) = 0$.
- Undesired situations can occur if the absolute margin function is applied without the sum-to-zero constraint. Consider, for instance, the case of all components of f being negative. In this case the "own class" margin μ_y is negative while all "other class" margins μ_c, $c \neq y$, are positive, and the example could still be classified correctly. This seems counter-intuitive, although formally allowed by Definition 1. Thus, we argue that the sum-to-zero constraint should be demanded for any multi-class SVM. The one-versus-all machine (OVA, see below) relies on absolute margins, but the sum-to-zero constraint is not enforced. However, in OVA it is not possible that all components of the margin function are negative.

2.3 Aggregation Operators

Definition 1 ensures that positive margins μ_c indicate correct classification. Therefore it is natural to construct a large margin loss from target margin violations, defined as

$$v_c(f(x), y) = \max\left\{0, \gamma_{y,c} - \mu_c\big(f(x), y\big)\right\} \ .$$

The target margins $\gamma_{y,c}$ are typically chosen to be $\gamma_{y,c} = 1$, which we assume in the following if not stated otherwise. The margin violations $v_c = L^{\text{hinge}}(\mu_c(f(x), y)$ correspond to the hinge loss applied to the different margin components. As indicated above, other binary SVM loss functions can be applied at this point. In this case the resulting multi-class SVMs will be compatible with the corresponding binary SVM based on that surrogate loss (e.g., the squared hinge loss is considered by Guermeur, 2012).

The d margin violations need to be combined into a single cost value:

Definition 3 (aggregation operator) *An* aggregation operator *is a non-constant function*

$$\Delta : (\mathbb{R}_0^+)^d \times Y \to \mathbb{R}_0^+$$

of margin violations $v = (v_1, \ldots, v_d)$ and a label $y \in Y$ with the properties

- $\Delta\big((0, \ldots, 0), y\big) = 0$, *and*
- $\Delta((v_1, \ldots, v_d), y)$ *is monotonically increasing in all components v_i of the first argument.*

The following aggregation operators are employed in machines that have been proposed in the literature:

Definition 4 *We define the following aggregation operators:*

$$\begin{aligned}
&\Delta^{\textit{self}}\big((v_1, \ldots, v_d), y\big) = v_y && (\textit{self} \text{ operator})\\
&\Delta^{\textit{t-max}}\big((v_1, \ldots, v_d), y\big) = \max_{c \in Y}\left\{v_c\right\} && (\textit{total-max} \text{ operator})\\
&\Delta^{\textit{o-max}}\big((v_1, \ldots, v_d), y\big) = \max_{c \in Y \setminus \{y\}}\left\{v_c\right\} && (\textit{max-over-others} \text{ operator})\\
&\Delta^{\textit{t-sum}}\big((v_1, \ldots, v_d), y\big) = \sum_{c \in Y} v_c \ , && (\textit{total-sum} \text{ operator})\\
&\Delta^{\textit{o-sum}}\big((v_1, \ldots, v_d), y\big) = \sum_{c \in Y \setminus \{y\}} v_c \ . && (\textit{sum-over-others} \text{ operator})
\end{aligned}$$

These operators can be understood as computing the value of a linear program, as outlined in the supplementary material (Section A).

2.4 Unification of Existing Machines

This study considers all-in-one approaches to SVM multi-class classification. These methods extend the SVM concept of large-margin classification to the multi-class case and cast the learning task as a single optimization problem. The first all-in-one methods proposed independently by Weston and Watkins (1999), Vapnik (1998, Section 10.10), and Bredensteiner and Bennett (1999) turned out to be equivalent, up to rescaling of the decision functions and the regularization parameter $C > 0$. We refer to this method as WW or as WW-SVM. It is expressed within our framework as the machine relying on the sum operator[1] applied to relative margins.

An alternative multi-class SVM was proposed by Crammer and Singer (2002). Like the WW-SVM, the CS machine takes all class relations into account simultaneously and solves a single optimization problem, however, with fewer slack variables in its primal optimization problem. It corresponds to combining relative margins with the max-over-others operator.

There are two key differences between the WW- and CS-SVMs. First, the aggregation operators are different. Second, the CS-SVM has originally been defined only for hypotheses without bias term. The machine can be extended to hypotheses with bias (Hsu and Lin, 2002), but efficient training algorithms have been missing for this extension. In our framework we do not distinguish between hypothesis classes with and without bias; all machines are defined for both classes.

Lee, Lin and Wahba (2004) proposed a distinct approach to multi-class SVM classification. We refer to this approach as the LLW-SVM. It was the first machine to use absolute margins in an all-in-one approach. The absolute margin violations are combined under the sum-to-zero constraint by means of the sum-over-others operator.

The LLW machine was the first multi-class SVM with a classification calibrated loss function, which guarantees its Fisher consistency (Lee et al., 2004; Tewari and Bartlett, 2007; Liu, 2007; Liu and Yuan, 2011). That is, in the limit of infinite data minimal risk in the LLW sense implies minimal 0-1-risk and thus a Bayes-optimal decision rule.

The multi-class maximum margin regression (MMR) method proposed by Szedmak et al. (2006) is driven by the observation that the quadratic term

$$\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

in the Wolfe dual of the binary SVM (Cortes and Vapnik, 1995; Bottou and Lin, 2007) can be interpreted as the squared norm with respect to the product kernel $k \otimes k_y$, where the "label kernel" $k_y : \{\pm 1\} \times \{\pm 1\} \to \mathbb{R}$ is defined as $k_y(y_i, y_j) = y_i y_j$. This is the standard inner product on the label space $Y = \{-1, +1\} \subset \mathbb{R}$. This idea is generalized to multiple classes by considering the standard inner product on $\mathbb{R}^d$ applied to label prototype vectors $\{\hat{y}_1, \ldots, \hat{y}_d\} = Y \subset \mathbb{R}^d$ representing the classes. This corresponds one-to-one to the definition of a positive definite kernel $k_y(c, c') = \langle \hat{y}_c, \hat{y}_{c'} \rangle$ on the standardized label space $Y = \{1, \ldots, d\}$. Two different sets of prototypes have been proposed, namely

$$\hat{y}_c = \sqrt{\frac{d-1}{d}}\left(\frac{-1}{d-1}, \ldots, 1, \ldots, \frac{-1}{d-1}\right)$$

1. For relative margins, the sum-over-others and the total-sum operator coincide.

and $\hat{y}_c = (0, \ldots, 1, \ldots, 0)$, where the c-th component equals 1. The first set corresponds to a symmetric setup of unit vectors with maximal angles between prototypes, while the second setup relies on an orthonormal basis. We refer to these variants as MMR and MMR$^\perp$, respectively. They correspond to the label kernels

$$k_y(y_i, y_j) = \begin{cases} 1 & \text{for } y_i = y_j \\ -\frac{1}{d-1} & \text{otherwise} \end{cases} \qquad \text{and} \qquad k_y^\perp(y_i, y_j) = \delta_{y_i, y_i} \ .$$

Both of these machines are obtained by applying the self aggregation operator to absolute margins, where the MMR machine enforces the sum-to-zero constraint, while the MMR$^\perp$ machine does not.

The MMR$^\perp$ method applied to a two-class problem does typically not give the same decision function as the standard binary SVM. With its orthogonal label kernel, it explicitly ignores all interactions between classes, so that the machine effectively trains d independent machines, similar to one-class SVMs for the different classes.

The unique selling proposition of the MMR and MMR$^\perp$ machines is their applicability to problems with large numbers of classes. By design, the optimization problem of the MMR machine has only ℓ dual variables. Even better, in case of the MMR$^\perp$ machine the problem decomposes into d independent sub-problems. However, this means that the MMR$^\perp$-SVM does not take any inter-class relationships in the data into account.

Our unification also captures the *reinforced multicategory SVM* (RM-SVM) recently proposed by Liu and Yuan (2011). This machine, or family of machines, weights two types of margin violations using a parameter $\gamma \in [0, 1]$. In our framework, it can be viewed as relying on a *reinforced margin function* defined as

$$\mu_c^{\mathrm{r}}(f(x), y) = \begin{cases} \gamma f_c(x) & \text{for } c = y \\ -(1-\gamma) f_c(x) & \text{for } c \in Y \setminus \{y\} \end{cases}$$

and using the total-sum operator and the coefficients

$$\gamma_{y,c} = \begin{cases} \gamma(d-1) & \text{for } c = y \\ (1-\gamma) & \text{for } c \in Y \setminus \{y\} \end{cases} \tag{4}$$

for combining the margin violations. The sum-to-zero constraint is enforced. For $\gamma = 0$, RM equals LLW and for $\gamma = 1$ it equals MMR with a scaled target margin.

There exist several general techniques to build a multi-category classifier based on some binary classifiers, which can be chosen to be binary SVMs. The most prominent of these so-called sequential approaches are the one-versus-one (OVO) and the one-versus-all (OVA) scheme. As an OVO approach inevitably requires some additional non-canonical decision making procedure different from (1) (e.g., Hastie and Tibshirani, 1998; Platt et al., 2000; Chang and Lin, 2011) and the optimal choice of this procedure is highly task-dependent, OVO-type approaches are out of the scope of this study (error-correcting-output-codes, ECOC, provide a general framework covering OVA and many OVO methods, see Dietterich and Bakiri, 1995; Allwein et al., 2001; Passerini et al., 2004).

However, the OVA scheme fits into our framework. It relies on only d different binary decision functions, each separating one class from all the rest (Vapnik, 1998, Section 10.10).

Usually this class is considered the "positive" class ($y = +1$) in the binary problem, and all other classes are mapped to the "negative" class ($y = -1$). The resulting decision function can be thought of as voting for the class under consideration. Let $y_i^c = +1$ if $c = y_i$ and $y_i^c = -1$ if $c \neq y_i$ denote the binary label of the i-th example when training the machine separating class c from the rest. Then OVA constructs decision functions f_c according to

$$\min_{f_c} \quad \frac{1}{2}\|f_c\|^2 + C \cdot \sum_{i=1}^{\ell} L^{\mathrm{hinge}}\big(y_i^c \cdot f_c(x_i)\big) \ .$$

and plugs them into Equation (1). Solving these d independent optimization problems at once results in a problem of type (3). This proceeding amounts to summing up the margin violations of all binary problems. Thus, OVA can be viewed as total-sum aggregation applied to absolute margins, without enforcing the sum-to-zero constraint.

Our framework highlights the conceptual similarities and differences of the above-mentioned machines, summarized in Table 1.

machine	margin μ	aggregation operator Δ	sum-to-zero constraint	bias term
WW	relative	sum-over-others	optional	yes
CS	relative	maximum-over-others	optional	no
LLW	absolute	sum-over-others	yes	yes
MMR	absolute	self	yes	yes
MMR$^\perp$	absolute	self	no	yes
RM	absolute	total-sum (affine combination of self and sum-over-others)	yes	yes
OVA	absolute	total-sum	no	yes

Table 1: Properties of the various multi-category SVMs in terms of the unifying framework. The last column indicates the presence of the bias or offset term in the original formulation found in the literature; all machines can be formulated with or without bias term.

2.5 Completing the Picture: New Multi-class SVMs

Our unification makes it easy to construct novel loss functions based on the principles extracted from existing approaches. From Table 1 it becomes obvious that some combinations of established margin functions and aggregation operators have not yet been explored. In the following, we define the corresponding machines and name them by a three-letter code, abbreviating the margin and loss types as indicated in bold in the following. The first of these new SVMs is the AMO machine, combining the **a**bsolute margin concept with the **m**ax-over-**o**thers operator. The machine can be understood as a combination of the LLW-SVM, from which it takes the margin concept, and the CS-SVM with its max-over-others aggregation.

The other two machines, ATM and ATS, use **a**bsolute margins with the **t**otal **m**aximum and the **t**otal **s**um operators. The ATS-SVM resembles the RM-SVM with $\gamma = 0.5$, however,

ATS uses target margins of $\gamma_{y,c} = 1$ for $y, c \in \{1, \ldots, d\}$, while RM-SVM uses the target margins defined by (4) which differ depending on whether c equals y or not. The only difference between the OVA scheme and the ATS machine is the presence of the sum-to-zero constraint in the latter. Still, the ATS machine is an all-in-one machine, because its training problem is not a composition of independent (binary) problems.

2.6 Unified Dual Problem

Training SVMs amounts to solving convex quadratic optimization problems. The traditional approach to solving these problems when using non-linear kernels is considering the corresponding dual formulations. In the following, we derive the Wolfe dual of the unified primal problem (3). The structure of the dual is often well-suited for the application of decomposition algorithms, which have proven to be a powerful choice for training SVMs with non-linear kernels.

For the case without sum-to-zero constraint, we arrive at the following dual problem (see supplement Section B for details):

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i,p} \gamma_{y_i,p} \cdot \alpha_{i,p} - \frac{1}{2} \sum_{i,p} \sum_{j,q} M_{y_i,p,y_j,q} \cdot k(x_i, x_j) \cdot \alpha_{i,p} \cdot \alpha_{j,q} && (5) \\ \text{s.t.} \quad & \forall i, p: \quad 0 \leq \alpha_{i,p} \\ & \forall i, r: \quad \sum_{p \in P_y^r} \alpha_{i,p} \leq C \\ & \forall c: \sum_{i,p} \alpha_{i,p} \nu_{y_i,p,c} = 0 && (6) \end{aligned}$$

with

$$M_{y_i,p,y_j,q} = \sum_c \nu_{y_i,p,c} \nu_{y_j,q,c} \ .$$

The bias parameters b_c can be obtained from the KKT optimality conditions.

If the sum-to-zero constraint is enforced, we obtain the same structure of the dual problem, but this time with

$$M_{y_i,p,y_j,q} = \sum_{m,n} \left(\delta_{m,n} - \frac{1}{d} \right) \nu_{y_i,p,m} \nu_{y_j,q,n} \ .$$

We call these auxiliary constants $M_{y_i,p,y_j,q}$ used in the formulation of the dual problems *kernel modifiers*, because they appear as multipliers of the kernel in the dual problem. Let the map $I : \{1, \ldots, \ell\} \times P_y \to \{1, \ldots, \ell |P_y|\}$ assign a unique index to each pair of training pattern and constraint. We define $Q \in \mathbb{R}^{\ell |P_y| \times \ell |P_y|}$ by

$$Q_{I(i,p),I(j,q)} = M_{y_i,p,y_j,q} \cdot k(x_i, x_j) \ .$$

This matrix, together with the vector $V \in \mathbb{R}^{\ell |P_y|}$, $V_{I(i,p)} = \gamma_{y_i,p}$, defines the quadratic objective function

$$W(\alpha) = V^T \alpha - \frac{1}{2} \alpha^T Q \alpha \qquad (7)$$

of the dual problem, written in vector notation.

The weight vectors are given by

$$w_c = \sum_{i,p} \alpha_{i,p} \nu_{y_i,p,c} \phi(x_i) - \eta \qquad (8)$$

with $\eta = \sum_{i,p} \alpha_{i,p} \left(\frac{1}{d} \sum_c \nu_{y_i,p,c} \right) \phi(x_i)$ in case of a sum-to-zero constraint and zero otherwise.

For solving the problem with bias parameters, which lead to d equality constraints, we adopt the *iterative problem alternation* approach by Kienzle and Schölkopf (2005).

2.7 Training Complexity

Table 2 gives the asymptotic training times of various approaches. Under the assumption that the runtime bounds are tight, the table suggests that training MMR$^\perp$ is faster than training OVO, which is slightly faster than MMR, which is again faster than training OVA, which is finally followed by the various all-on-one SVMs with $O(d)$ dual variables per training example. The asymptotic nature of the runtime bounds hides factors considerably influencing training times in practice: Different machines often perform best for different hyper-parameters, and different concepts of separability (as discussed in Section 2 and Section 3) have an impact.

MMR$^\perp$	OVO	MMR	OVA	WW / CS / LLW AMO / ATM / ATS / RM
$\sum_{c=1}^{d} \mathcal{C}(\ell_c)$ $\widehat{=}\ d \cdot \mathcal{C}(\ell/d)$	$\sum_{1 \leq c < e \leq d} \mathcal{C}(\ell_c + \ell_e)$ $\widehat{=}\ d^2 \cdot \mathcal{C}(\ell/d)$	$\mathcal{C}(\ell)$	$d \cdot \mathcal{C}(\ell)$	$\mathcal{C}(d\ell)$

Table 2: Asymptotic runtime of the training algorithms under the assumption that solving the different n-dimensional quadratic programs is in the complexity class $\mathcal{C}(n)$. There are good arguments to assume $\mathcal{C}(n) = O(n^q)$ for some q between 2 and 3 (Joachims, 1998; Bottou and Lin, 2007). The number of training patterns is denoted by ℓ, the number of examples per class c by ℓ_c, and the number of different classes by d. In the second row the complexity is given under the assumption of (roughly) balanced classes, that is, $\ell_c \in \Theta(\ell/d)$.

3. A Closer Look at Aggregation Operators and Margin Concepts

In this section, we analyze the concepts underlying the various multi-class SVM variants.

First, we will consider the asymptotic behavior in the limit of infinite data by studying consistency. This analysis is closely related to two streams of research. First, we will focus on the Fisher consistency of the loss functions employed for SVM training. This type of consistency, also known as classification calibration, has already been studied for a number of multi-class losses (Lee et al., 2004; Tewari and Bartlett, 2007; Liu, 2007; Liu and Yuan, 2011). It is only indirectly concerned with the SVM itself, since the effect of the regularizer is ignored. Second, we will consider universal consistency building on the well-known result for binary SVMs by Steinwart (2002b). It states that with a suitable

dependency $C(\ell)$ of the regularization parameter C on the data set size ℓ, for any universal kernel, the SVM approaches the Bayes-optimal decision in the limit of infinite data. In addition to the theoretical analysis, we will empirically study the performance of the SVMs on artificial learning tasks that have been designed to highlight crucial differences between margin concepts and aggregation operators.

3.1 Fisher Consistency of Multi-class Loss Functions

In the following, we will analyze the Fisher consistency of the loss functions considered in this study. Let $L(f(x), y)$ denote the loss function applied by a multi-class SVM for training. Let $f^* : X \to \mathbb{R}^d$ denote a minimizer (among all measurable functions) of the risk under the data generating distribution P. A loss function is Fisher consistent if

$$\arg\max \left\{ f_c^*(x) \,\middle|\, c \in Y \right\} \subset \arg\max \left\{ P(y \,|\, x) \,\middle|\, y \in Y \right\} \ .$$

In the following, we use the statements that a multi-class SVM variant and its training loss are Fisher consistent interchangeably.

The consistency of the LLW loss was established by Lee et al. (2004), and it was confirmed by Liu (2007) that this is the only example of a Fisher consistent multi-class loss among the popular WW-, CS-, MMR-, and LLW-SVMs. The ATS machine can be viewed as belonging to the family of reinforced multicategory SVMs. Thus, the Fisher consistency of the ATS loss follows from the analysis by Liu and Yuan (2011).

We will investigate the consistency of the new machines AMO and ATM following the proceeding by Liu (2007). We adopt the short notation $P_y = P(y \,|\, x)$ for the probability of observing class y given x and define $[t]_+ = \max\{0, t\}$. It holds:

Theorem 5 *Let $L(f(x), y)$ denote either the loss function used by the AMO machine or the loss function used by the ATM machine, that is, the loss resulting from application of either the max-over-others or the total-max operator to absolute margins:*

$$L(f(x), y) = \max_{c \in Y \setminus \{y\}} \left\{ v_c^{abs}(f(x), y) \right\} = \left[1 + \max_{c \in Y \setminus \{y\}} \left\{ f_c(x) \right\} \right]_+ \qquad \text{(AMO)}$$

or

$$L(f(x), y) = \max_{c \in Y} \left\{ v_c^{abs}(f(x), y) \right\} = \max \left\{ \left[1 + \max_{c \in Y \setminus \{y\}} \left\{ f_c(x) \right\} \right]_+ , \left[1 - f_y(x) \right]_+ \right\} \ . \qquad \text{(ATM)}$$

Then the minimizer f^* of the corresponding risk $\mathcal{R} = \mathbb{E}[L(f(x), y)]$, subject to the sum-to-zero constraint $\sum_{c \in Y} f_c(x) = 0$, satisfies:

- *If there exists a* majority class *$y \in Y$ such that $P_y > (d-1)/d$, then $f_y^*(x) = d - 1$ and $f_c^*(x) = -1$ for all $c \in Y \setminus \{y\}$.*
- *If $P_y < (d-1)/d$ for all $y \in Y$, then $f^*(x) = 0$.*

The proof, which is given in the supplementary material (Section C), works by analyzing the point-wise risk $\mathcal{R}_x = \sum_{y \in Y} P(y \,|\, x) L(f(x), y)$ and computing the function values $f_c(x)$ minimizing $\mathcal{R}_x$. Thus, the Fisher consistency of the loss functions used in the all-in-one machines considering absolute margins can be summarized as follows:

Corollary 1 *It follows directly from the previous results that the AMO-loss and the ATM-loss are not Fisher consistent while the LLW-loss and the ATS-loss are Fisher consistent.*

In light of the conceptual similarity between the ATS machine and the LLW-SVM, it is not surprising that the ATS-loss is Fisher consistent. However, combining margin violations by means of the maximum-operator appears to be problematic, since neither the AMO nor the ATM machine (the two "max-loss siblings" of the Fisher consistent LLW and ATS machines) share this property.

The question arises how Fisher consistency is related to (a) the margin type, (b) the aggregation of margins in the loss, and (c) the sum-to-zero constraint. Part (a) is easy to answer, since Liu (2007) has proved inconsistency of the relative margin machines (WW and CS). Thus, we focus on absolute margins in the following. We investigate the impact of the sum-to-zero constraint on the maximum and the sum of margin violations. It turns out that dropping the constraint destroys Fisher consistency:

Theorem 6 *Let $L(f(x), y)$ denote the maximum or the sum over absolute margin violations, and assume $P_y < 1/2$ for all $y \in Y$. Then the unconstrained minimizer f^* of the corresponding risk $\mathcal{R} = \mathbb{E}[L(f(x), y)]$ satisfies $\forall c \in Y : f_c^*(x) = -1$ for the sum and $\forall c \in Y : f_c^*(x) = 0$ for the maximum.*

We do not prove this statement here; the proof is analog to the one of Theorem 5 using the techniques developed by Liu (2007). The optimal solution for the maximum operator is all zeros and satisfies the sum-to-zero constraint (cf. Theorem 5), and therefore dropping the constraint does not make any difference. However, the constraint is relevant when the sum operator is employed. The resulting solution does not sum to zero, and it is not Bayes-optimal. This result confirms the importance of the constraint for Fisher consistency.

The ATS machine can be interpreted as the OVA machine with sum-to-zero constraint, and Theorem 6 covers the OVA machine, which is inconsistent in the absence of a majority class. Thus, adding the sum-to-zero constraint makes the OVA machine consistent, at the cost that the different weight vectors w_c cannot be obtained independently anymore.

3.2 Universal Consistency

While Fisher consistency is just a property of the loss function, universal consistency considers the whole machine including the regularizer and its impact relative to the data set size. Steinwart's universal consistency analysis for binary SVMs builds on the fact that the hinge loss is classification calibrated for binary problems. It can be extended to multi-class SVMs with Fisher consistent losses under the assumption of the same training set size dependent choice $C(\ell)$ of the regularization parameter as assumed for the binary SVMs. Thus, from Corollary 1 we get:

Theorem 7 *The LLW-SVM and the ATS-SVM with universal kernel and regularization parameter chosen according to Theorem 2 by Steinwart (2002b) are universally consistent classifiers.*

We omit the proof of this statement, because is a straight-forward extension of the involved proof for binary machines.

It can be shown that using the same rule for $C(\ell)$ as in Theorem 2 from Steinwart (2002b) with a non-Fisher consistent loss does not yield a universally consistent classifier. However, this negative result does not exclude the existence of a different dependency of C on ℓ so that the resulting classifier is universally consistent. Thus, the question arises whether for an SVM surrogate loss function L there exists a dependency $C(\ell)$ of the regularization parameter on the training set size such that the resulting classifier is universally consistent. In the following, we answer the question for all SVM variants covered in this study—with the surprising result that also the MMR-, WW-, and OVA-SVMs can be made consistent when operated with a completely different trade-off $C(\ell)$:

Theorem 8 *For the Gaussian kernel $k(x,x') = \exp(-\gamma\|x-x'\|^2)$ on an input space $X \subset \mathbb{R}^p$, the machines MMR-SVM, WW-SVM, LLW-SVM, ATS-SVM, and OVA-SVM with regularization parameter $C(\ell) \le 1/(d\cdot\ell)$ and kernel parameter $\gamma(\ell)$ fulfilling $\lim_{\ell\to\infty}\gamma(\ell) = \infty$ and $\lim_{\ell\to\infty}\ell\cdot\gamma(\ell)^{-p/2} = \infty$ are universally consistent classifiers.*

Proof Let us first consider the all-in-one machines. The absolute values of all components of the quadratic matrix Q in the dual problem (7) are upper bounded by one. All components of the dual solution α are bounded by $C(\ell) = 1/(d\cdot\ell)$. The sum in the derivative

$$\frac{\partial W(\alpha)}{\partial \alpha_{i,p}} = 1 - \sum_{j,q} Q_{I(i,p),I(j,q)}\cdot\alpha_{j,q}$$

runs over at most $d\cdot\ell$ summands, each of which is bounded by $\frac{1}{d\cdot\ell}$. Thus all gradient components are non-negative in the whole box-shaped feasible region. It follows that all dual variables $\alpha_{i,p}$ end up at the upper bound C. From Equation (8) and the definition of the machines it can be seen that for the primal solution it holds $w_c = \tilde{w}_c - \tilde{\eta}$ with

- $\tilde{\eta} = \eta$ and $\tilde{w}_c = C\cdot\sum_{i|y_i=c}\phi(x_i)$ for the MMR machine,
- $\tilde{\eta} = \eta - C\cdot\sum_i\phi(x_i)$ and $\tilde{w}_c = C\cdot d/2\cdot\sum_{i|y_i=c}\phi(x_i)$ for the WW machine,
- $\tilde{\eta} = \eta - C\cdot\sum_i\phi(x_i)$ and $\tilde{w}_c = C\cdot\sum_{i|y_i=c}\phi(x_i)$ for the LLW machine,
- $\tilde{\eta} = \eta - C\cdot\sum_i\phi(x_i)$ and $\tilde{w}_c = 2C\cdot\sum_{i|y_i=c}\phi(x_i)$ for the ATS machine.

The weight vectors $\tilde{w}_c$ are a scaled version of the kernel density estimator (KDE), which is well known to be universally consistent. The above conditions on $\gamma(\ell)$ translate into the preconditions of Theorem 10.1 by Devroye et al. (1996, p. 150) for the kernel width $h = \sqrt{1/\gamma(\ell)}$. The KDE output fed into the arg max decision rule (1) ignores the additive constant $\tilde{\eta}$ and yields consistent classification predictions.

The OVA result can be proven in the same way. Again, all dual variables take the value C. It is easy to see that the primal SVM solution is then a scaled KDE plus a constant vector: $w_c = \tilde{w}_c - \tilde{\eta}$, with $\tilde{\eta} = \sum_i\phi(x_i)$ and $\tilde{w}_c = 2\cdot\sum_{i|y_i=c}\phi(x_i)$. Thus, the resulting classifier is universally consistent. ■

The above statement is a straight-forward reduction of the SVM to a simple kernel density estimator. The same result holds for binary SVMs. It is a much simpler statement with

a much simpler proof than the famous universal consistency result by Steinwart (2002b). The core idea of Steinwart's analysis is to carefully increase the modeling power of the SVM with growing number of training points so that the impact of regularizer decays to zero, while at the same time overfitting is avoided. In contrast, we keep the impact of the regularizer constantly high. When operating the SVM in this regime it loses its most important features, namely the large margin approach and the sparsity (Steinwart and Christmann, 2008). This may seem highly undesirable, but from the viewpoint of classification accuracy it does not matter much whether an SVM with a specific value of C works well because of a large margin or because of the consistency of KDE. Of course, KDE estimation is less sample and memory efficient than large-margin prediction. Sample efficiency is one of the primary arguments brought forward by Vapnik (1998) for preferring direct estimation of the decision boundary over a plug-in classifier based on KDE.

Theorem 8 establishes the consistency of the MMR, WW, and OVA machines, despite the Fisher inconsistency of their losses. This implies that an analysis of the loss function in isolation in terms of Fisher consistency is not sufficient for fully understanding the predictions made by these machines.

The simple trick of reverting to KDE-based prediction is not trivial, since it does not work for all large margin losses:

Theorem 9 *There exists no function $C(\ell)$ so that CS-SVM, AMO-SVM, or ATM-SVM become universally consistent.*

Proof Consider a single-point input space $X = \{x_0\}$ with any kernel $k(x_0,x_0) > 0$ (any positive kernel is universal on this space). Consider a problem with $d = 3$ classes. It is completely described by the class probabilities; w.l.o.g. they fulfill $p_1 \ge p_2 \ge p_3$. Lemma 4 by Liu (2007) for the CS machine, and Theorem 5 for the AMO-SVM and the ATM-SVM, respectively, offer choices of these probabilities so that the minimizer of the empirical risk is $f_1(x_0) = f_2(x_0) = f_3(x_0) = 0$, with corresponding weight vectors $w_1 = w_2 = w_3 = 0$. This inconsistent solution also minimizes the regularizer, and thus the regularized risk for all $C > 0$.[2] ■

The proof lifts results on Fisher-(in)consistency from the level of the loss function to inconsistency of the actual SVM, taking the regularizer into account. The case of the MMR, WW and OVA machines treated above makes clear that this extension is non-trivial, since for these machines the regularizer makes the difference between consistency and inconsistency. We regard this type of analysis as highly relevant, since SVMs do not actually minimize the empirical risk, but instead a combination of empirical risk and a specific regularizer.

The present analysis covers the three losses involving the max-aggregation completely, since for no C (or $C(\ell)$) they result in a universally consistent SVM. On the other hand, there is a gap in our analysis of machines employing sum-aggregation losses: For $\ell\cdot C(\ell)\to\infty$ Steinwart's analysis applies, while the trivial reduction to a KDE works only for $\ell\cdot C(\ell)$ (asymptotically) bounded by a rather small constant, so that all target margins are always

2. An alternative proof strategy is to construct a non-zero dual optimal solution that corresponds to the primal zero solution.

violated. The question whether an SVM can be consistent for parameters in the gap in between these regimes is left open.

3.3 Artificial Benchmark Problems

In the following, we will show experimental results on highly controlled problems with analytically defined distributions. This investigation complements and enriches the previous theoretical analysis. Some of the theoretical results on Fisher consistency of loss functions can be demonstrated in a non-asymptotic setting. The theoretical results on universal consistency require a universal kernel and thus an "arbitrarily rich" feature space. This is, however, not always the case in practice, where restricted and low-dimensional feature spaces can occur. We visualize the behavior of nine different multi-class SVMs over a range of values of the regularization parameter. The simple test problems serve as "counter examples" that help to understand when and why some of the machines deliver substantially sub-optimal solutions.

We define a basic test problem with two variants, one noise-free and one with label noise. The domain $X = S^1 = \{x \in \mathbb{R}^2 \,|\, \|x\| = 1\}$ of these test problems is the unit circle. The circle is parameterized by $t \in [0, 20[$ through the curve $\beta(t) = \big(\cos(t \cdot \pi/10), \sin(t \cdot \pi/10)\big)$. The problem involves three classes $Y = \{1, 2, 3\}$, see Figure 1 and Figure 2. For the noiseless problem data points $(x, y) \in X \times Y$ are generated as follows. First the label $y \in Y$ is drawn from the uniform distribution on Y. Then x is drawn uniformly at random from the sector X_y. The sectors have different sizes. They are defined as $X_1 = \beta([0, 5))$, $X_2 = \beta([5, 11))$, and $X_3 = \beta([11, 20))$. The only change for the noisy case is that 90% of the labels are reassigned uniformly at random. In other words the distribution of the x-component remains unchanged, and conditioned on $x \in X_z$ the event $y = z$ has probability 40%, while the other two cases have probabilities of 30%. In both variants of the problem it is Bayes-optimal to predict label y on sector X_y. This solution can be realized by a linear model without offset term.

These problems are rather elementary. They are low-dimensional and thus easy to visualize. Their only difficulty lies in the uneven sector sizes. This also results in different densities of the x-components in the different sectors. This design avoids the exploitation of artificial symmetries. The construction is so that in the noisy variant there is no majority class at any point.

We have conducted a parameter study with all nine machines (with linear kernel and without offset term). In the noise-free case we have drawn $\ell = 100$ samples. Since the problem is linearly separable and noise-free, we have tested rather large values of $C = 10^n$ with $n \in \{0, 1, 2, 3, 4\}$ (there are no interesting effects outside this range). The resulting classifiers are shown in Figure 1. The information content of noisy samples is significantly reduced, which is why we use $\ell = 500$ and $n \in \{-4, -3, -2, -1, 0\}$ here (again, results do not change outside this range). The classifier predictions are visualized in Figure 2.

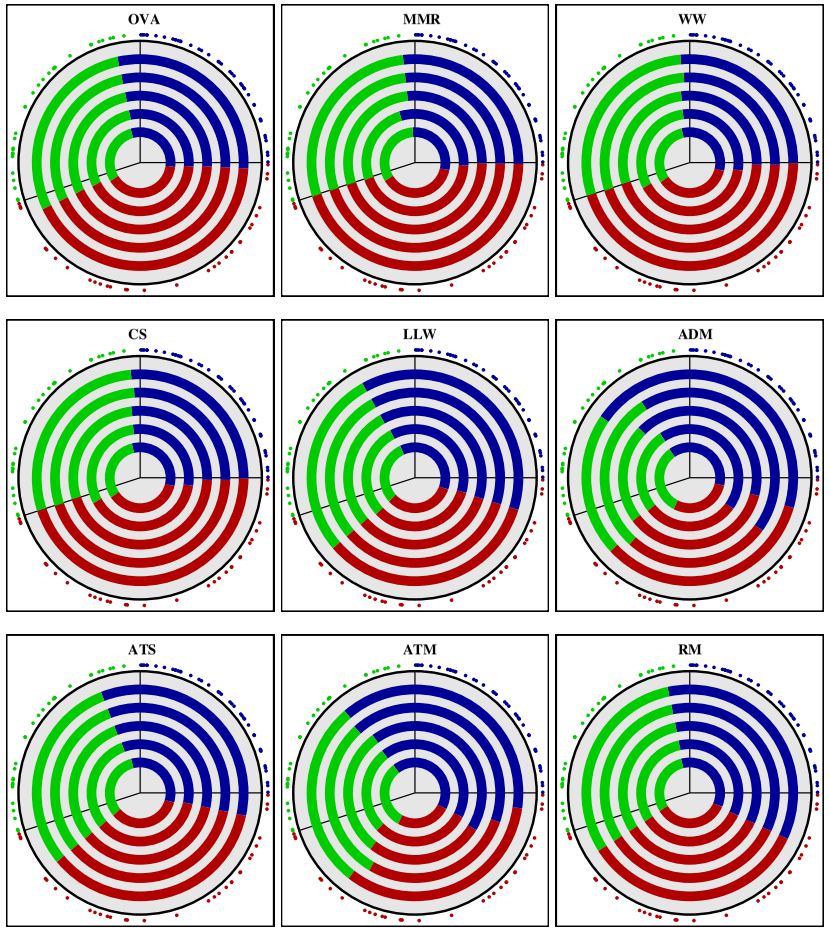


Figure 1: Noiseless circle problem. The sector separators are the decision boundaries of the Bayes-optimal predictor. Classes 1, 2, and 3 are indicated by colors blue, green, and red, respectively. The points on the outside of each graph are the 100 training samples. The colored circles indicate the classifier predictions for $C = 10^n$, $n \in \{0, 1, 2, 3, 4\}$, increasing from inner to outer circles.

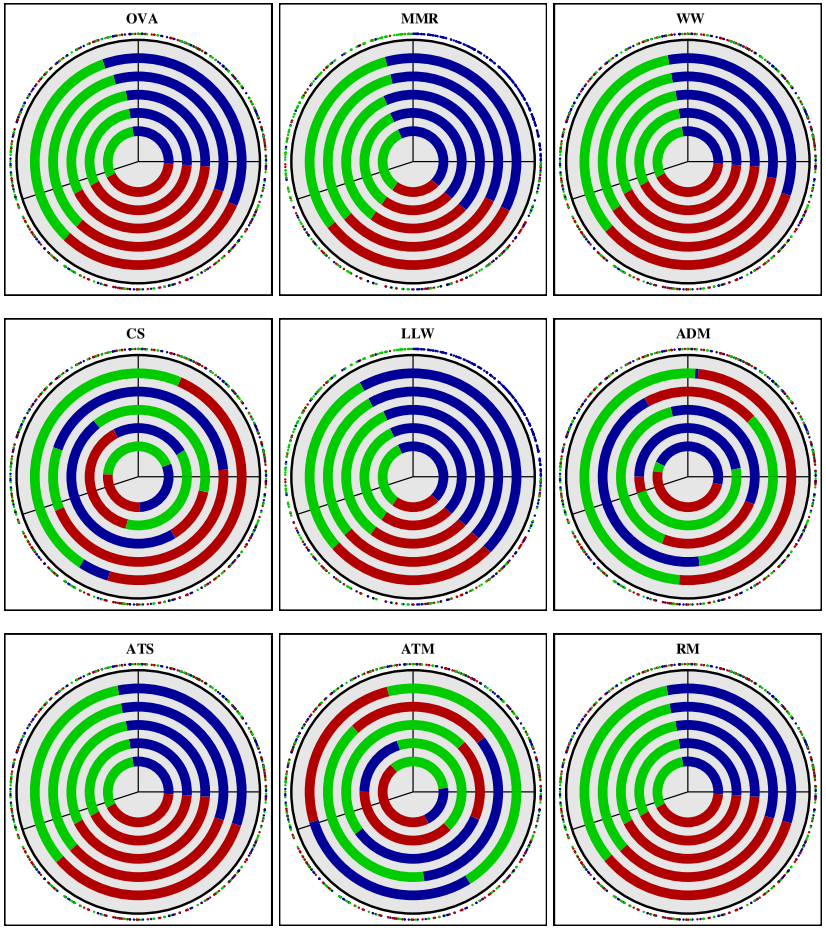


Figure 2: Noisy circle problem. The sector separators are the decision boundaries of the Bayes-optimal predictor. Classes 1, 2, and 3 are indicated by colors blue, green, and red, respectively. The points on the outside of each graph are the 500 training samples. The colored circles indicate the classifier predictions for $C = 10^n$, $n \in \{-4, -3, -2, -1, 0\}$, increasing from inner to outer circles.

Figure 1 shows that the relative margin machines WW and CS excel, in particular for high values of C. Also the MMR machine relying on the self-margin component performs well. In contrast, the absolute margin machines LLW, AMO, ATS, RM, and ATM all fail, over the whole range of parameters (despite the universal consistency of some of the approaches in rich enough feature spaces). The same defect can be observed for OVA, but in weaker form.

The reason for this failure is that the definition of absolute margins does not allow to solve the given problem without violation of the target margin. This holds even for the Bayes-optimal classifier. Thus, absolute margins are not compatible with the form of the decision function. This is obvious for the OVA approach on the circle problem, because one class cannot be separated from the others by a single linear hyperplane through the origin. It turns out that this effect is not a direct consequence of the OVA approach, but of absolute margins in general.

Figure 2 displays the performance in the presence of massive label noise. The performance of the WW machine remains good, but as expected it needs a much smaller value of C. The performance of MMR degrades a bit, while OVA does well in this case. The sub-optimal performance of the LLW machine is largely unchanged, while the closely related ATS and RM profit from the self-margin component (which by itself works well, as shown by the MMR results). Most strikingly, the CS machine with its relative margin as well as the AMO and ATM machines with their absolute margins give random predictions. All three approaches rely on maximum-based aggregation operators. This behavior is perfectly explained by the Fisher consistency results (also reflected in the proof of Theorem 5): Since there is no majority class, all losses favor the trivial solution $w_1 = w_2 = w_3 = 0$. This solution is also optimal for the regularizer. Thus, it is attained for all values of C, and the figures show nothing but numerical noise. In the noisy case the combination of loss components with maximum-based aggregation operators results in guessing performance. Notably this includes the well established CS machine.

In low dimensional feature spaces we see absolute margin machines fail in the noiseless case and maximum-based aggregation operators in the noisy case.[3] Despite the simplicity of the synthetic problems, only a single machine solves both cases, namely the WW approach. The simplistic OVA and MMR machines perform worse, but still satisfactory.

4. Empirical Comparison

This section empirically compares nine approaches to multi-category SVM learning, namely the sequential OVA scheme, the established all-in-one methods WW, CS, LLW, RM, and MMR, as well as the new methods called AMO, ATS, and ATM. All algorithms were implemented in the Shark open source machine learning library (Igel et al., 2008). Source code for reproducing the results can be found in the supplementary material. First, twelve standard benchmark data sets were considered for non-linear SVM learning, and careful model selection was conducted. This already makes our experiments the most extensive comparison of multi-class SVMs so far. Second, additional experiments for linear SVMs are presented later in this section.

3. This result does of course not exclude the existence of a similar problem where WW also fails. However, we did not manage to construct such an instance.

4.1 Multi-class Benchmark Problems

The descriptive statistics of the twelve data sets used to evaluate the non-linear multi-class SVM methods are given in Section D of the supplementary material. All features of all data sets were pre-processed by rescaling to unit variance. This rescaling was done for each split into training and test data individually based on the statistics of the particular training set.

4.2 Model Selection

In all non-linear SVM experiments, Gaussian kernels $k_\gamma(x_1, x_2) = \exp(-\gamma||x_1 - x_2||^2)$ were used. The bandwidth γ of the Gaussian kernel and the regularization parameter C of the machine were determined by nested grid search. Repeated cross-validation was employed as model selection criterion. Five-fold cross-validation was repeated ten times using ten independent random splits into five folds. This stabilizes the model selection procedure especially for small data sets. Candidate parameters were evaluated on the 5×10 validation subsets and the configuration yielding the best average performance was chosen. If any of the selected model parameters was at the grid boundary, then the grid was extended accordingly.

We set the initial grid to $\gamma \in \{2^{-12+3i} \,|\, i = 0, 1, \ldots, 4\}$ and $C \in \{2^{3i} \,|\, i = 0, 1, \ldots, 4\}$. Let (γ_0, C_0) denote the parameter configuration picked in the first stage. Then in the second stage the parameters were further refined on the grid $\gamma \in \{2^i \cdot \gamma_0 \,|\, i = -2, -1, 0, 1, 2\}$ and $C \in \{2^i \cdot C_0 \,|\, i = -2, -1, 0, 1, 2\}$. For linear SVMs we applied the same approach restricted to the complexity control parameter C. The hyperparameters as determined by the grid-searches are given in Section E of the supplementary material.

Table 3 provides a comparison reflecting the computational demand of parameter tuning. It lists the total training time for computing the 5-fold cross validation error on a 5×5 grid of the outer grid search loop, centered on the optimal parameters. Training times differ by several orders of magnitude depending on the SVM variant.

Data set	OVA	MMR	WW	CS	LLW	AMO	ATS	ADS	RM
Car	36.45	12.89	131.7	1027	6112	40231	2399	34046	4106
Glass	1.10	0.32	2.05	318.6	48.50	369.2	247.1	186.1	69.87
Iris	1.41	0.13	13.02	1.30	25.58	4.63	647.7	35.01	243.1
Red wine	53.87	10.51	57.68	2574	7354	75159	6486	67000	2235

Table 3: Time in seconds for computing the 5-fold cross validation error on a single core over a 5×5 grid around the optimal parameter values.

4.3 Evaluation

The multi-class SVMs were evaluated as follows. Using the best parameters found during model selection, 100 machines were trained on 100 random splits into training and test data (preserving the original set sizes).[4] In this way properties of the test error distribution

4. Doing the full model selection 100 times would be the better setup, however, it was computationally too demanding.

can be estimated. We report empirical mean and standard deviation. Furthermore, the 100 repetitions allow to test results for significant differences. For each problem we have compared each machine to the best performing one with a paired U-test (at significance level $\alpha = 0.01$). We are not interested in the test results per se; instead, the tests provide thresholds for judging performance as competitive or inferior. It must be noted that the 100 repetitions share the same data and are thus not independent. This means that the confidence level needs adjustment. However, the U-test still provides a meaningful thresholding mechanism.

4.4 Stopping Condition

For a fair comparison of training times of different SVMs, it is of importance to choose comparable stopping criteria for the quadratic programming. Unfortunately, this is hardly possible in the experiments presented in this study, because the quadratic programs differ. However, in the case of just two classes all machines solve the same problem. Therefore the stopping condition was selected such that for a binary problem these machines would give the same solution. The common threshold of $\varepsilon = 10^{-3}$ on violations of the Karush-Kuhn-Tucker (KKT) conditions of optimality was used as stopping criterion (Bottou and Lin, 2007).

To rule out artifacts of this choice several experiments were repeated with an accuracy of $\varepsilon = 10^{-5}$. These experiments did not result in improved performance. Thus, the choice of the stopping criterion may in the worst case influence training times, but not the test accuracies reported in Section 4.5 and Section 4.6.

For all non-linear machines, the maximum number of SMO iterations was limited to 10.000 times the number of dual variables. The limit for linear machines was 10 times higher.[5] This was necessary to keep the grid searches computationally tractable. However, the few parameter configurations that had hit this limit corresponded to "degenerated" machines (i.e., bad solutions), typically with far too small γ and too large C. Thus, this proceeding did not influence the outcome of the model selection process.

4.5 Non-linear SVM Results

The test accuracies of all nine machines are listed in Table 4. The table also indicates whether a paired U-test judges the 100 test accuracies as significantly worse than the machine with the best performance. The relative accuracies of the 9 machines are represented graphically in Figure 3.

A similar plot in Figure 4 displays training times. The training times varied vastly with the choice of the hyper-parameters, that is, they were strongly dependent on the outcome of the model selection procedure. In some cases there exist configurations with similar prediction performance but different optimization times. Thus, the timing results were subject to non-negligible noise and should thus be interpreted with care.

5. The higher limit accounts for the fact that single iterations are less efficient, because in the algorithm proposed by Hsieh et al. (2008) and Fan et al. (2008) and refined by Glasmachers and Doğan (2013, 2014) the choice of the active variable does not take gradient and gain information into account, see Section 4.6.

	OVA	MMR	WW	CS	LLW	AMO	ATS	ATM	RM
Abalone	26.89	26.51	**27.51**	21.33	26.65	20.42	26.64	20.58	22.02
	± 1.26	± 1.31	± 1.19	± 0.95	± 1.25	± 1.13	± 1.24	± 1.43	± 1.09
Car	98.37	97.58	**98.62**	**98.61**	98.14	98.11	98.14	98.09	98.24
	± 0.72	± 0.82	± 0.72	± 0.71	± 0.79	± 0.80	± 0.77	± 0.74	± 0.62
Glass	**68.69**	**68.03**	**68.78**	**69.03**	68.03	**69.52**	68.38	**69.53**	**69.09**
	± 4.61	± 5.33	± 5.24	± 4.48	± 4.80	± 5.01	± 4.89	± 4.95	± 5.68
Iris	**96.66**	94.71	**96.35**	95.26	**96.22**	95.08	**95.86**	95.11	**96.02**
	± 2.43	± 2.99	± 2.59	± 2.88	± 2.51	± 2.57	± 2.77	± 2.52	± 2.64
Opt. digits	**98.80**	98.25	**98.77**	98.76	**98.85**	97.87	**98.84**	**98.84**	**98.90**
	± 0.25	± 0.26	± 0.24	± 0.24	± 0.22	± 0.29	± 0.22	± 0.22	± 0.23
Page blocks	96.73	96.65	**96.83**	**96.78**	96.69	96.63	96.69	96.63	**96.75**
	± 0.46	± 0.46	± 0.42	± 0.44	± 0.47	± 0.43	± 0.45	± 0.43	± 0.37
Sat	**92.19**	91.75	**92.19**	**92.19**	**92.13**	**92.14**	**92.15**	**92.14**	**92.20**
	± 0.53	± 0.54	± 0.53	± 0.54	± 0.53	± 0.49	± 0.52	± 0.48	± 0.50
Segment	**96.41**	96.22	**96.39**	**96.36**	**96.43**	**96.41**	**96.41**	**96.42**	**96.60**
	± 0.73	± 0.71	± 0.73	± 0.60	± 0.73	± 0.68	± 0.73	± 0.75	± 0.65
Soy bean	89.58	87.24	89.13	89.88	89.81	88.86	**90.41**	88.72	**91.16**
	± 3.04	± 3.40	± 2.96	± 3.11	± 3.32	± 3.00	± 3.23	± 3.21	± 2.52
Vehicle	**84.90**	73.94	84.28	83.99	**84.86**	83.74	**84.83**	83.84	**84.71**
	± 1.88	± 2.17	± 2.00	± 1.90	± 1.99	± 2.30	± 2.13	± 2.30	± 1.81
Red wine	**63.93**	**64.13**	**63.87**	**63.90**	**63.91**	**64.06**	**63.93**	**64.06**	**63.99**
	± 2.07	± 1.83	± 1.91	± 2.02	± 2.13	± 2.03	± 2.07	± 2.03	± 1.87
White wine	64.03	63.96	**64.86**	64.04	64.02	**64.83**	64.39	**64.83**	**64.27**
	± 1.12	± 1.18	± 1.17	± 1.15	± 1.11	± 1.17	± 1.09	± 1.17	± 1.05

Table 4: Classification accuracies in percent of correctly classified test examples. The table lists mean values and standard deviations. In each row bold numbers indicate that the result is not significantly worse than the best one (paired U-test, $\alpha = 0.01$).

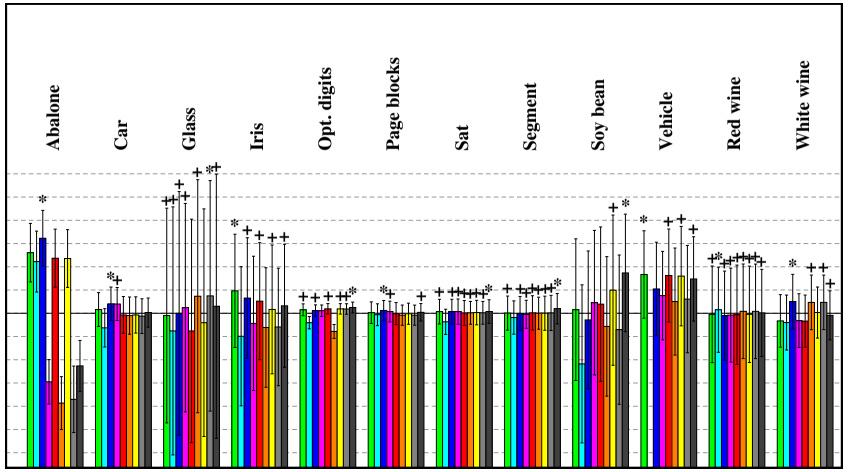


Figure 3: Relative accuracies of all nine machines. The average performance per data set defines the mid-line. Each dashed line marks 1% of deviation. The colored bars denote (relative) classification accuracy (higher is better). The error bars indicate the standard deviation across multiple training/test splits. The best result is indicated with a star (∗). Statistically insignificant differences to the best solution (U-test, $\alpha = 0.01$) are marked with a plus sign (+). Colors represent models as follows: green (OVA), cyan (MMR), blue (WW), pink (CS), red (LLW), orange (AMO), yellow (ATS), light gray (ATM), and dark gray (RM).

In many applications not only training times but also testing times are of relevance. The time it takes to compute the prediction of an SVM classifier is usually dominated by the kernel computations. In the multi-class case this means that sparsity in the dual variables is not enough: only variables that do not appear in any of the expansions of the weight vectors $w_1, \ldots, w_d$ can be dropped in the evaluation of the classifier. We count every training example that is used in the construction of any of the weight vectors as a support vector. The fraction of support vectors is reported in Figure 5.

4.6 Linear SVM Results

In recent years there has been increased interest in linear SVM classifiers. These are a viable alternative to non-linear SVMs in particular in high-dimensional input spaces commonly found, for instance, in text mining and bioinformatics problems. A powerful coordinate descent solver for the dual linear SVM problem was proposed by Hsieh et al. (2008) and Fan et al. (2008) and has been recently extended by Glasmachers and Doğan (2013, 2014). All linear SVM results reported in this section were obtained with an extension of the latter technique to the multi-class case.

For the linear kernel case we have extended the testbed beyond the twelve problems to data sets with different characteristics, which are typically used to evaluate linear multi-class models, see supplementary material (Section E). These data sets are available from the libsvm data collection.[6] We have included problems with rather low-dimenional as well as extremely high-dimensional input spaces. The test errors of the nine linear multi-class SVMs are summarized in Table 5.

6. http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

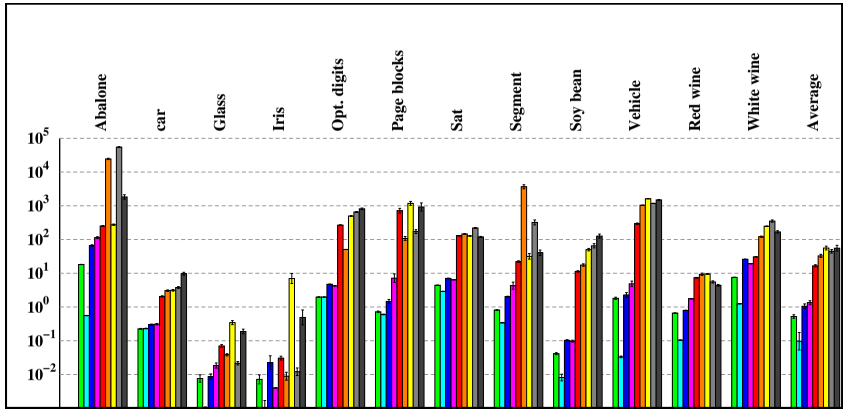


Figure 4: Average training times (logarithmic scale) of all nine machines, in seconds (lower is better). Colors represent models as follows: green (OVA), cyan (MMR), blue (WW), pink (CS), red (LLW), orange (AMO), yellow (ATS), light gray (ATM), and dark gray (RM). The (geometric) average training time across all 12 problems is found on the right.

4.7 Discussion

Our results allow to compare the nine different machines from different angles. We generally view accuracy as a primary goal and training speed and prediction speed (related to sparsity of the solution) as secondary criteria.

The accuracy results of the non-linear SVMs presented in Table 4 and Figure 3 were mixed. The prediction performances of most machines were rather close to each other. This is in accordance with the findings of Hsu and Lin (2002) and Rifkin and Klautau (2004). Larger deviations could be observed on the problems Abalone and Vehicle. The maximum combination operator absolute margin machines AMO and ATM fell behind on the Abalone problem, while the self-margin MMR machine did not give satisfactory performance on the Vehicle problem. Overall the WW and RM machines performed best, for example, when counting the number of data sets where the solution was either best or not significantly worse than the best result. The MMR machines showed the weakest performance. However, it appears than most machines give rather competitive results; OVA, CS, LLW and ATS are hard to put in a clear order, and they are only slightly worse than WW and RM (if at all).

This picture changes completely for linear SVMs. The results presented in Table 5 reveal drastic differences between machines, as already found by Hsu and Lin (2002). There were only two problems where all machines achieved roughly comparable performance, namely News-20 and Sector. These problems come with extremely high-dimensional feature spaces and are thus similar in character to the non-linear SVM problems. Here the classification

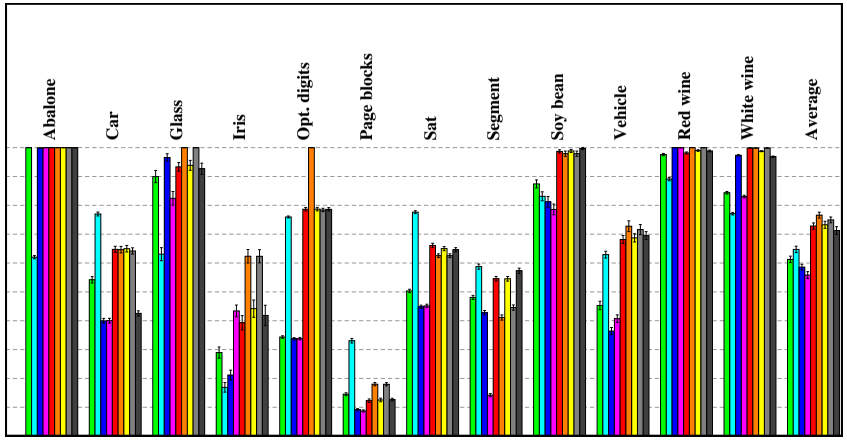


Figure 5: Average fraction of support vectors for all nine machines (lower is better). A bar hitting the top indicates that all points are support vectors, while a bar vanishing at the bottom indicates that no points were used as support vectors. Colors represent models as follows: green (OVA), cyan (MMR), blue (WW), pink (CS), red (LLW), orange (AMO), yellow (ATS), light gray (ATM), and dark gray (RM). The average fraction of support vectors across all 12 problems is found on the right.

calibrated LLW and RM approaches and the other absolute margin machines worked about as well as in the non-linear case. Most other problems have rather low-dimensional feature spaces. Thus, the setting comes conceptually closer to the synthetic circle problems presented in Section 3. This explains the complete breakdown of some methods on some of the problems, for example, all absolute margin machines on the letter data. In accordance with the findings in Section 3, WW and CS outperformed the classification calibrated LLW, ATS and RM machines. The linear SVM results also clearly display the problems of the OVA and MMR approaches. While OVA works well at least in some cases, we have to dismiss the MMR machine completely for its consistently inferior performance. The WW and CS machines clearly performed best, again with a slight but insignificant edge for WW.

The secondary goals of training and test times give a yet completely different picture. The rightmost column in Figure 4 summarizes the training times. The measurements are in accordance with the asymptotical results in Table 2. Not surprisingly, the MMR machine was clearly fastest to train. It is followed by OVA, which is, however, closely followed by WW and CS. It is clear from the asymptotical considerations that the differences between OVA and the all-in-one machines will get more pronounced with increasing number of classes. The WW machine was not slower to train than the popular CS approach, actually

	OVA	MMR	WW	CS	LLW	AMO	ATS	ATM	RM
Covertype	50.59	23.55	**70.55**	45.73	21.87	47.31	18.44	51.19	69.61
	± 5.49	± 0.19	± 0.09	± 5.88	± 23.19	± 6.79	± 17.71	± 7.52	± 0.40
Letter	63.69	21.66	69.39	**76.59**	12.78	6.54	17.44	6.78	21.39
	± 0.48	± 1.21	± 0.63	± 0.61	± 0.40	± 1.88	± 0.79	± 2.04	± 1.84
News-20	85.36	78.10	85.13	85.17	**86.71**	85.65	**86.69**	85.61	**86.61**
	± 0.32	± 0.53	± 0.15	± 0.32	± 0.39	± 0.32	± 0.37	± 0.35	± 0.37
Sector	94.53	91.97	94.10	94.80	94.82	**95.09**	94.82	**95.09**	94.74
	± 0.22	± 0.30	± 0.33	± 0.29	± 0.28	± 0.30	± 0.28	± 0.30	± 0.37
Usps	94.50	87.84	94.46	**95.26**	78.18	40.57	80.60	40.27	88.08
	± 0.39	± 0.75	± 0.57	± 0.46	± 5.27	± 2.15	± 0.65	± 3.27	± 0.69
Abalone	18.95	15.80	**21.70**	14.12	16.56	8.36	18.33	8.78	18.86
	± 0.86	± 0.90	± 1.30	± 1.64	± 1.17	± 2.34	± 0.76	± 2.31	± 1.38
Car	71.69	70.47	**73.76**	**73.15**	65.34	70.43	72.14	70.49	72.52
	± 1.73	± 1.86	± 1.68	± 2.02	± 12.17	± 1.83	± 1.71	± 1.94	± 1.58
Glass	**56.98**	43.87	**61.93**	**61.93**	46.78	28.43	49.51	32.13	51.50
	± 6.44	± 7.74	± 6.63	± 6.04	± 6.77	± 11.98	± 5.78	± 12.04	± 6.77
Iris	91.11	65.34	**95.88**	**91.76**	74.65	67.32	82.05	67.32	85.14
	± 4.85	± 6.07	± 1.71	± 7.18	± 7.52	± 6.66	± 5.94	± 6.66	± 4.96
Opt. Digits	95.98	89.08	96.03	**96.42**	73.56	18.45	81.32	19.11	92.10
	± 0.60	± 1.08	± 0.37	± 0.37	± 2.11	± 4.77	± 1.62	± 4.41	± 0.66
Page Blocks	70.44	48.99	91.14	**94.20**	93.22	**93.87**	**93.81**	**93.74**	**93.58**
	± 21.20	± 24.39	± 5.41	± 2.34	± 1.02	± 1.58	± 1.44	± 1.71	± 0.75
Sat	75.04	31.04	**77.40**	66.87	51.47	47.59	61.21	45.49	63.47
	± 0.96	± 0.95	± 3.00	± 9.90	± 9.01	± 6.89	± 0.95	± 6.97	± 1.29
Segment	**92.54**	50.90	**92.43**	**92.43**	74.50	67.58	76.30	67.58	81.94
	± 0.75	± 2.52	± 2.13	± 2.13	± 1.32	± 12.21	± 4.26	± 12.21	± 2.48
Soy Bean	**90.65**	61.41	87.75	83.49	77.95	38.87	66.74	39.86	79.20
	± 3.03	± 7.80	± 3.16	± 5.80	± 9.97	± 6.42	± 4.52	± 5.84	± 4.23
Vehicle	52.02	52.33	72.75	72.75	63.21	63.21	63.21	63.21	**76.42**
	± 11.98	± 2.98	± 4.13	± 4.13	± 10.63	± 10.63	± 10.63	± 10.63	± 2.38
Red wine	53.38	23.63	**58.37**	55.61	57.26	36.58	56.29	36.27	**57.81**
	± 2.63	± 5.68	± 1.69	± 2.47	± 2.02	± 10.95	± 1.64	± 9.67	± 1.68
White wine	50.73	19.20	**51.78**	50.85	46.44	30.03	51.16	27.06	51.41
	± 1.27	± 9.68	± 1.24	± 1.12	± 1.74	± 8.21	± 0.98	± 7.21	± 1.27

Table 5: Classification accuracies of the linear machines in percent of correctly classified test examples. The table lists mean values and standard deviations. In each row, bold numbers indicate that the result is not significantly worse than the best one (paired U-test, $\alpha = 0.01$).

it was slightly faster on average in our experiments.[7] Finally, the various absolute margin machines took at least an order of magnitude longer to train than WW and CS.

7. Here we only present our results for training without bias parameters. However, this statement also holds true for training with bias parameters.

At least for expensive kernel functions prediction speed is governed by the sparsity of the resulting model. Figure 5 reveals that sparsity is primarily a property of the problem, while the chosen loss function has only a minor impact. Overall, multi-class models are expected to be less sparse than, for instance, binary SVMs since a training point can be dropped only if it does not appear in any of the d weight vectors. We observed significant sparsity only for the page blocks data. In comparison relative margin machines gave slightly more sparse models than absolute margin machines. The MMR self-margin machine did not stand out in this category. However, only few problems had sparse solutions at all, and inter-problem spread was much higher than differences between methods. Hence we do not see good reasons to reject any machine based on these data.

In summary, the WW and CS machines showed robust performance across all disciplines, with WW slightly ahead of CS. The MMR machine performed worst in terms of accuracy, but was fastest to train. In most cases there are no good reasons for absolute margin machines.

The results have an interesting interpretation in the unified view presented in Section 2. Differences between self, relative and absolute machine machines were much more pronounced than differences between aggregation operators. Relying only on the single self-margin component (MMR) is computationally attractive but results in rather poor prediction accuracy. Relative margins give strong machines. Consistent absolute margin machines with sum-aggregation are a viable alternative in sufficiently high-dimensional feature spaces, although they are usually costly to train.

It is questionable whether a connection between asymptotic consistency and practical performance exists. This is partially observable in sum-aggregation absolute margin machines performing well only in high-dimensional feature spaces. However, this condition does not seem to be necessary, since the other absolute margin machines show the same trend and also the inconsistent CS machine performs well, except for the noisy circle problem (see Section 3.3). Overall, prediction performance seems to be more closely related to the applied margin concept than to uniform consistency.

Based on our empirical findings, we can recommend relative margin machines for almost all applications. We prefer the WW machine over the CS approach for its slightly more stable performance and its theoretical properties (consistency of WW with Gaussian kernel, see Theorem 8, and breakdown of CS on the noisy circle problem).

5. Conclusion

This article provides a novel unified view on multi-class SVMs, showing that the various approaches are not as different as they seem. All popular algorithms reduce to the standard SVM for binary classification problems, however, they differ along several dimensions when applied to more than two classes. The machines can be formulated based on a vector-valued decision function with as many components as classes. We have highlighted margin concepts, named relative and absolute margins (including the self-margin component), that describe whether the components of the decision function are optimized relative to each other or not. Independent of the margin concept, the machines can employ either sum- or maximum-based aggregation operators for combining margin violations into a scalar loss. A further distinction can be made based on whether a machine fulfills (or even enforces) that

the components of the decision function sum to zero or not. All machines can be formulated with and without bias term (i.e., for linear and affine hypotheses in feature space), and we derived a unified formulation of the corresponding dual optimization problems in terms of margin concepts and aggregation operators.

Our unifying view pointed at three combinations of these features that had not been investigated. These missing machines were derived and evaluated. The new classifiers, named AMO-, ATS- and ATM-SVM, all use the absolute margin concept but consider different aggregates of the margin violations. The ATS machine, which can be viewed as one-vs-all classification with imposed sum-to-zero constraint and is closely related to RM SVMs, turns out to be Fisher consistent.

The LLW, RM, and ATS SVMs rely on a classification calibrated loss function. However, we have shown that a non-classification calibrated training loss function does not need to be a principal problem for an SVM, since the overall *regularized* empirical risk minimizer can still be consistent. This is the case for the maximum margin regression (MMR), one-vs-all (OVA), and Weston & Watkins (WW) machines. One may argue that for large enough amounts of data one should prefer consistent models over inconsistent large margins. It is provably impossible to "repair" the inconsistency of the Crammer & Singer (CS) multi-class SVM, AMO and ATM losses by means of SVM-style regularization.[8] The MMR and WW machines, and even the OVA scheme (lacking the sum-to-zero property), turn out to be consistent classifiers when regularized properly. We argue that these results are best understood in the context of our unified view. All inconsistent machines apply the maximum operator for aggregation, while none of the consistent machines relies on this operator. Thus, our results hint at a fundamental difference between the sum and maximum aggregation operators.

Our unified learning problems lend themselves to efficient optimization algorithms. These allowed us to perform a thorough empirical comparison of the various multi-class SVMs and to draw conclusions about training times and generalization performance. We considered the LLW and RM machines in our experiments, which are typically omitted in comparison studies for training time reasons (e.g., Ding and Dubchak, 2001; Rifkin and Klautau, 2004; Statnikov et al., 2005), and we could perform proper model selection. We compared nine different multi-class SVM with Gaussian and linear kernel on a large collection of benchmark problems. The unified view enables a meaningful interpretation of the results by relating them to margin concepts and aggregation operators. We find that relative margin machines are superior for the linear kernel, and that sum-based aggregation generally outperforms the maximum approach. Our empirical and theoretical results support the following recommendations:

The standard all-in-one multi-class WW SVM should be used as the default since it gives robust performance at moderate training times. It can be trained at least as fast as the popular CS SVM while giving at least as good generalization results and having nicer theoretical properties (which can matter in practice as shown on artificial test problems and when using a linear kernel). The simplistic OVA method can be a viable alternative; it delivers slightly worse results on average while being a little faster to train. If training time is a major concern then the MMR machine may seem like the best option. However,

8. In practice, the performance of the CS machine is very similar to the WW machine. Thus, more often than not this effect does not seem to be dominant.

its performance can be so poor that even sub-sampling the data in combination with WW or OVA may be a better approach.

With universal kernels and in general in extremely high-dimensional feature spaces the LLW and RM machines with their classification calibrated losses are usually on par with WW and CS and thus promising candidates. However, training may be infeasible for large data sets, for its more than ten times longer training times compared to WW and CS.

Acknowledgments

Tobias Glasmachers acknowledges support by the Mercator Research Center Ruhr (MERCUR), project Pr-2013-0015, and Christian Igel from The Danish Council for Independent Research (DFF) through the project Surveying the sky using machine learning (SkyML).

References

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2001.

A. Bordes, N. Usunier, and L. Bottou. Sequence labelling SVMs trained in one pass. In W. Daelemans, B. Goethals, and K. Morik, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5211 of *LNCS*, pages 146–161. Springer, 2008.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT 1992)*, pages 144–152. ACM, 1992.

L. Bottou and C. J. Lin. Support vector machine solvers. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*, pages 1–28. MIT Press, 2007.

E. J. Bredensteiner and K. P. Bennett. Multicategory classification by support vector machines. *Computational Optimization and Applications*, 12(1):53–79, 1999.

C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2002.

L. Devroye, L. Gyorfi, and G. Lugosi. *A probabilistic theory of pattern recognition*. Springer Verlag, 1996.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2(263):286, 1995.

C. H. Q. Ding and I. Dubchak. Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, 17(4):349, 2001.

Ü. Doğan, T. Glasmachers, and C. Igel. A note on extending generalization bounds for binary large-margin classifiers to multiple classes. In P. A. Flach, T. De Bie, and N. Cristianini, editors, *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD 2012)*, volume 7523 of *LNCS*, pages 122–129. Springer, 2012.

R. E. Fan, K.W. Chang, C. J. Hsieh, X.R. Wang, and C. J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

T. Glasmachers and Ü. Doğan. Accelerated coordinate descent with adaptive coordinate frequencies. In *Proceedings of the Fifth Asian Conference on Machine Learning (ACML)*, JMLR W&CP, 2013.

T. Glasmachers and Ü. Doğan. Coordinate Descent with Online Adaptation of Coordinate Frequencies. Technical Report arXiv:1401.3737, arxiv.org, 2014.

Y. Guermeur. VC theory for large margin multi-category classifiers. *Journal of Machine Learning Research*, 8:2551–2594, 2007.

Y. Guermeur. A generic model of multi-class support vector machine. *International Journal of Intelligent Information and Database Systems (IJIIDS)*, 6:555–577, 2012.

T. Hastie and R. Tibshirani. Classification by pairwise coupling. *Annals of Statistics*, 26 (2):451–471, 1998.

S. I. Hill and A. Doucet. A framework for kernel-based multi-category classification. *Journal of Artificial Intelligence Research*, 30(1):525–564, 2007.

C. J. Hsieh, K.W. Chang, C. J. Lin, S.S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, pages 408–415. ACM, 2008.

C. W. Hsu and C. J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

C. Igel, T. Glasmachers, and V. Heidrich-Meisner. Shark. *Journal of Machine Learning Research*, 9:993–996, 2008.

T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, chapter 11, pages 169–184. MIT Press, 1998.

W. Kienzle and B. Schölkopf. Training support vector machines with multiple equality constraints. In *Machine Learning: ECML 2005*, volume 3720 of *LNCS*, pages 182–193. Springer, 2005.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–82, 2004.

Y. Liu. Fisher consistency of multicategory support vector machines. In M. Meila and X. Shen, editors, *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 2 of *JMLR W&P*, pages 289–296, 2007.

Y. Liu and M. Yuan. Reinforced multicategory support vector machines. *Journal of Computational and Graphical Statistics*, 20(4):901–919, 2011.

A. Passerini, M. Pontil, and P. Frasconi. New results on error correcting output codes of kernel machines. *IEEE Transations on Neural Networks*, 15(1):45–54, 2004.

J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 547–553. MIT Press, 2000.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, 21(5):631, 2005.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2002a.

I. Steinwart. Support vector machines are universally consistent. *Journal of Complexity*, pages 768–791, 2002b.

I. Steinwart and A. Christmann. *Support vector machines*. Springer, 2008.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: Maximum margin regression. Technical report, PASCAL, Southampton, UK, 2006.

A. Tewari and P. L. Bartlett. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 8:1007–1025, 2007.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, 1998.

J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In M. Verleysen, editor, *Proceedings of the Seventh European Symposium On Artificial Neural Networks (ESANN)*, pages 219–224. Evere, Belgium: d-side publications, 1999.

Y. Wu and Y. Liu. Robust truncated hinge loss support vector machines. *Journal of the American Statistical Association*, 102(479):974–983, 2007.

H. Zou, J. Zhu, and T. Hastie. The margin vector, admissible loss and multi-class margin-based classifiers. *Annals of Applied Statistics*, 2:1290–1306, 2008.

Journal of Machine Learning Research 17 (2016) 1-31 Submitted 1/14; Revised 8/15; Published 4/16

Addressing Environment Non-Stationarity by Repeating Q-learning Updates*

Sherief Abdallah SHARIO@IEEE.ORG
The British University in Dubai, P.O.Box 345015, Block 11, DIAC, Dubai, United Arab Emirates
University of Edinburgh, United Kingdom

Michael Kaisers KAISERS@CWI.NL
Centrum Wiskunde & Informatica, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
Maastricht University, The Netherlands

Editor: Jan Peters

Abstract

Q-learning (QL) is a popular reinforcement learning algorithm that is guaranteed to converge to optimal policies in Markov decision processes. However, QL exhibits an artifact: in expectation, the effective rate of updating the value of an action depends on the probability of choosing that action. In other words, there is a tight coupling between the learning dynamics and underlying execution policy. This coupling can cause performance degradation in noisy *non-stationary* environments.

Here, we introduce Repeated Update Q-learning (RUQL), a learning algorithm that resolves the undesirable artifact of Q-learning while maintaining simplicity. We analyze the similarities and differences between RUQL, QL, and the closest state-of-the-art algorithms theoretically. Our analysis shows that RUQL maintains the convergence guarantee of QL in stationary environments, while relaxing the coupling between the execution policy and the learning dynamics. Experimental results confirm the theoretical insights and show how RUQL outperforms both QL and the closest state-of-the-art algorithms in noisy non-stationary environments.

Keywords: reinforcement learning, Q-learning, multi-agent learning, non-stationary environments

1. Introduction

Q-Learning (Watkins and Dayan, 1992), QL, is one of the most widely-used and widely-studied learning algorithms due to its ease of implementation, intuitiveness, and effectiveness over a wide-range of problems (Sutton and Barto, 1999). Although QL was originally designed for single-agent domains, it has also been used in multi-agent settings with reasonable success (Claus and Boutilier, 1998). In addition, several gradient-based learning algorithms, which were invented specifically for multi-agent interactions, rely on Q-learning as an internal component to estimate the values of different actions (Abdallah and Lesser, 2008; Bowling, 2005; Zhang and Lesser, 2010). In this paper, we propose a novel algorithm that addresses important limitations of the Q-learning algorithm while retaining Q-learning's attractive simplicity. We show that our algorithm outperforms Q-learning and closely related state-of-the-art algorithms in several noisy and *non-stationary* environments.

*. An earlier version of this paper was presented at the International conference on Autonomous Agents and Multi-Agent Systems (Abdallah and Kaisers, 2013).

The main problem that our algorithm addresses is what we refer to as the *policy-bias* of the action value update. The policy-bias problem appears in Q-learning because the value of an action is only updated when the action is executed. Consequently, the *effective* rate of updating an action value directly depends on the probability of choosing the action for execution. If, hypothetically, the reward information for all actions (at every state) were available at every time step, then the update rule could be applied to all actions and the policy-bias would disappear. Let us refer to this hypothetical update as the *QL-ideal-update*.[1]

It is important to note that the policy-bias problem is different from the exploration-exploitation problem. To solve the exploration-exploitation problem, researchers optimize the execution policy in order to balance exploration vs. exploitation. However, the policy-bias refers to the dependency of the *rate of updating* an action's value on the *probability of selecting* the corresponding action; this dependency is not resolved by changing the policy. As previous research showed, the policy bias may cause a temporary decrease in the probability of choosing optimal actions with higher expected payoffs (Leslie and Collins, 2005; Kaisers and Tuyls, 2010; Wunder et al., 2010). This effect can be more severe, as we show later, in noisy non-stationary environments where the environment dynamics change over time.

Figure 1 illustrates the policy-bias problem using a very simple multi-armed-bandit domain. The domain is stateless with only two actions, one giving the reward of -1 and the other giving the reward of 1. For simplicity, let us assume the best value for the learning rate α, is 0.01 (as we highlight in our experiments later, the best value of the learning rate depends on how noisy the specific domain is). To measure the deviation between a learning algorithm and the QL-ideal-update, we use RMSE: the root mean square error of the Q values (for all actions) compared to the Q values of the QL-ideal-update over 1000 time steps. Figure 1 plots the RMSE of QL for different learning rates and over different execution policies (the probability of choosing the first action), which is fixed throughout the 1000 time steps.[2] We can observe from the figure that the RMSE of QL is minimal when the probability of choosing both actions are equal (pure exploration). However, as the execution policy deviates from the uniform distribution (toward exploitation), the RMSE of QL grows rapidly. It is also important to note that higher learning rate (α) will not lower the RMSE of QL across different execution policies. The RMSE of our proposed algorithm, RUQL, is lower across the different execution policies. In other words, even though our algorithm RUQL, similar to QL, updates only the selected action, the dynamics of learning are very similar to QL-ideal-update and therefore the Q values are close to the ideal values (as if all the actions are updated at every time step).

The algorithm we propose here addresses the policy bias problem. The main idea of the Repeated Update Q-learning (RUQL) is to repeat the update of an action inversely proportional to the probability of choosing that action. For example, if an action is to be chosen for execution only 10% of the time, then the update of that action (when finally chosen) shall be repeated 1/0.1 or 10 times. Consequently, every action will, in expectation, be updated an equal number of times. RUQL can be approximated by a closed-form expression that removes the need to actually repeat the updates,

1. In practice, QL-ideal-update may lead to horrible performance (since we are executing suboptimal actions every time step) and may be even impossible to achieve (in multi-state domains executing an action would change the state), but hypothetically QL-ideal-update would lead to ideal estimates of the action values (Q).
2. More specifically, QL-ideal-update was run for 1000 time steps. Then QL with each of the various execution policies was run for 1000 time steps. Then the squared error of Q-values were averaged over all actions, the root computed, and then averaged again over all time steps.

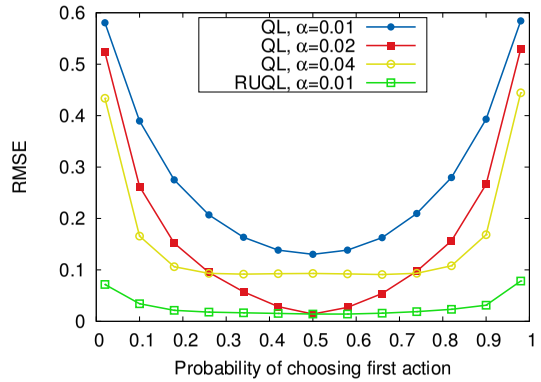


Figure 1: RMSE of QL for different values of α, and RUQL plotted against the probability of choosing the first action $\pi(1)$. RUQL, approximated as in Theorem 1, has the lowest RMSE across different policies.

as shown in Section 3. We show both theoretically and experimentally that the dynamics of RUQL approximate the dynamics of QL-ideal-update (as if all the actions were updated every time step). More importantly, RUQL outperforms both QL and the closest state-of-the-art, FAQL (Kaisers and Tuyls, 2010), in noisy non-stationary settings.

In summary, the main contributions of this paper are:

- A new reinforcement learning algorithm, RUQL, which builds on the Q-learning algorithm but does not suffer from the policy-bias problem.
- A mathematical derivation of a closed form of RUQL that makes the RUQL algorithm computationally feasible.
- Theoretical analysis that shows how RUQL maintains the convergence guarantee of QL (in stationary environments, and for tabular Q values), while having learning dynamics that are better suited for non-stationary environments.
- Theoretical analysis that shows the similarities and differences between RUQL and the closely-related state-of-the-art algorithm, FAQL (Kaisers and Tuyls, 2010).
- An extensive experimental study that confirms the theoretical analysis and compares the performance of RUQL to QL, FAQL, and two other learning algorithms.

The following section presents the required background concepts covering Q-learning and related algorithms. We then introduce the algorithm RUQL, followed by theoretical and experimental analysis. Finally, a summary and a discussion of the contributions conclude the article.

2. Background

The Q-learning algorithm (Watkins and Dayan, 1992) is a model-free[3] reinforcement learning algorithm that is guaranteed to find the optimal policy for a given Markov Decision Process (MDP). An MDP is defined by the tuple $\langle S, A, P, R \rangle$, where S is the set of states representing the system and $A(s)$ is the set of actions available to the agent at a given state s. The function $P(s, a, s')$ is the transition probability function and quantifies the probability of reaching state s' after executing action a at state s. Finally the reward function, $R(s, a, s')$, gives the average reward an agent acquires if the agent executes action a at state s and reaches state s'. The goal is then to find the optimal policy $\pi^*(s, a)$ which gives the probability of choosing every action a at every state s in order to maximize the expected total discounted rewards. In other words, $\pi^*(s, a) = \arg\max_{\pi(s,a)} E\{\sum_t \gamma^t r_t | s_{t=0} = s, a_{t=0} = a\}$, where the parameter γ denotes the discount factor that controls the relative value of future rewards. To solve the model, the action value function, Q, is introduced where $Q(s, a) = \sum_{s'} P(s, a, s') \left(R(s, a, s') + \gamma \max_{a'} Q(s', a')\right)$. The Q function represents the expected total discounted reward if the agent starts at state s, executes action a, and then follows the optimal policy thereafter. Intuitively, the function $Q^t(s, a)$ represents what the agent believes, at time t, to be the worth of action a at state s. The optimal policy can then be defined as $\pi^*(s) = \arg\max_a Q(s, a)$. The Q-learning algorithm incrementally improves the action-value function Q using the update equation

$$Q^{t+1}(s, a) \leftarrow Q^t(s, a) + \alpha \left(r + \gamma \max_{a'} Q^t(s', a') - Q^t(s, a) \right). \quad (1)$$

The parameter α is called the learning rate and controls the speed of learning. The variables r and s' refer to the immediate reward and the next state, both of which are observed after executing action a at state s. Algorithm 1 illustrates how the Q-learning update equation is typically used in combination with an exploration strategy that generates a policy from the Q-values.

Algorithm 1: Q-learning Algorithm

```
begin
    Initialize function Q arbitrarily.
    Observe the current state s.
    repeat
        Compute the policy π(s, a) from Q(s, a), balancing exploration and exploitation.
        Choose an action a according to agent policy π(s, a).
        Execute action a and observe the resulting reward r and the next state s'.
        Update Q(s, a) using Equation 1.
        Set s ← s'.
    until done
end
```

The agent needs to choose the next action considering that its current Q-values may still be erroneous. The belief of an action to be inferior may be based on a crude estimate, and there may be a chance that updating that estimate reveals the action's superiority. Therefore, the function Q^t

3. Q-learning does not require knowing the underlying stochastic reward or transition function of the MDP model.

in itself does not dictate which action an agent should choose in a given state (Step 5 in Algorithm 1). The policy $\pi(s,a)$ of an agent is a function that specifies the probability of choosing action a at state s. A greedy policy that only selects the action with the highest estimated expected value[4] can result in an arbitrarily bad policy, because if the optimal action initially has a low value of Q it might never be explored. To avoid such premature convergence on suboptimal actions, several definitions of π have been proposed that ensure all actions (including actions that may appear sub-optimal) are selected with non-zero probability. These definitions of π are often called *exploration* strategies (Sutton and Barto, 1999). The two most common exploration strategies are ϵ-greedy exploration and Boltzmann exploration. The ϵ-greedy exploration simply chooses a random action with probability ϵ and otherwise (with probability 1-ϵ) chooses the action with the highest Q-value greedily. The Boltzmann exploration strategy defines the policy $\pi(s,a)$ as a function of Q-values and τ

$$\pi^t(s,a) = \frac{e^{\frac{Q^t(s,a)}{\tau}}}{\sum_{a'} e^{\frac{Q^t(s,a')}{\tau}}}, \tag{2}$$

where the tunable parameter τ is called the temperature.

We use Boltzmann exploration in our experiments, but with an additional parameter π_{min}, which defines a threshold below which the probability of choosing an action can not fall. In a sense, this combines the nice continuity of the Boltzmann exploration with the exploration guarantee of ϵ-greedy exploration. This is implemented using a projection function (Zinkevich, 2003; Abdallah and Lesser, 2008), which projects the policy to the closest valid policy with minimum exploration.

Regardless of the exploration strategy, the agent needs to gain information about the value of each action at every state in order to become more certain over time that the action with the highest estimate truly is the optimal action. However, an agent can only execute one action at a time and the agent only receives feedback (reward) for the action that was actually executed. As a result, in Q-learning, the rate of updating an action relies on the probability of choosing that action. The theoretical comparison of updating one vs. all actions at each time step reveals that Q-learning counter-intuitively decreases the probability of optimal actions under some circumstances, which leads to drawbacks in non-stationary environments (Kaisers and Tuyls, 2010; Wunder et al., 2010).

Scaling the learning rate inversely proportional to the policy has been proposed to overcome this limitation, thereby approximating the simultaneous updating of all actions every time a state is visited. This concept has been initially studied to modify fictitious play (Fudenberg and Levine, 1998), and inspired two equivalent modifications of Q-learning named Individual Q-learning (Leslie and Collins, 2005) and Frequency Adjusted Q-learning (FAQL) (Kaisers and Tuyls, 2010). FAQL used the modified Q-learning update rule

$$Q^{t+1}(s,a) \leftarrow Q^t(s,a) + \frac{1}{\pi(s,a)}\alpha\Big(r + \gamma \max_{a'} Q^t(s',a') - Q^t(s,a)\Big).$$

Thus, FAQL scales the learning rate for an action a to be inversely proportional to the probability of choosing action a at a given state. This simple modification to the Q-learning update rule suffers from a practical concern: the update rate becomes unbounded (approaches ∞) as $\pi(s,a)$ approaches zero. Therefore, a safe-guard condition has to be added in practice (Kaisers and Tuyls, 2010)

$$Q^{t+1}(s,a) \leftarrow Q^t(s,a) + \min(1, \frac{\beta}{\pi(s,a)})\alpha\Big(r + \gamma \max_{a'} Q^t(s',a') - Q^t(s,a)\Big). \tag{3}$$

4. A greedy policy can be formally defined as $\pi^t(s,a) = 1$ iff $a = \arg\max_{a'} Q^t(s,a')$ and $\pi^t(s,a) = 0$ otherwise.

where β is a tuning parameter that safeguards against the cases where $\pi(s,a)$ is close to zero. The resulting algorithm is similar to Algorithm 1 but with Line 8 modified to use Equation 3 instead of Equation 1.

While introducing parameter β does make FAQL applicable in practical domains, the introduction of β also results in two undesirable properties. The first undesirable property is reducing the effective learning rate (instead of α it is now $\alpha\beta$) and therefore reducing the responsiveness of FAQL.[5] The second and more important undesirable property is what we refer to as the β-limitation: Once the probability of choosing an action, $\pi(s,a)$, goes below β, FAQL behaves identical to the original QL. In other words, FAQL suffers from the same policy-bias problem when the probability of choosing an action becomes lower than β. This is problematic because the lower the probability of choosing an action, the more important it is to adjust the learning rate (to account for the infrequency of choosing that action). Our proposed algorithm RUQL does not suffer from these limitations and we show in the experiments how this can improve performance in noisy and non-stationary settings.

3. Repeated-Update Q-learning, RUQL

Our proposed algorithm is based on a simple intuitive idea. If an action is chosen with low probability $\pi(s,a)$ then instead of updating the corresponding action value $Q(s,a)$ once, we repeat the update $\frac{1}{\pi(s,a)}$ times. Algorithm 2 shows the naive implementation of this idea. Line 8 is the only difference between Algorithm 2 and Algorithm 1.

Algorithm 2: RUQL (Impractical Implementation)

```
1  begin
2      Initialize function Q arbitrarily.
3      Observe the current state s.
4      for each time step do
5          Compute the policy π using Q.
6          Choose an action a according to agent policy π.
7          Execute action a and observe the resulting reward r and the next state s'.
8          for i : 1 ≤ i ≤ ⌊1/π(s,a)⌋ do
9              Update Q(s,a) using Equation 1.
10         end
11         Set s ← s'.
12     end
13 end
```

This simple modification to the QL algorithm addresses the policy-bias, but it has two limitations: (1) The repetition only works with $\frac{1}{\pi(s,a)}$ an integer, and (2) as $\pi(s,a)$ gets lower, the number of repetitions increases and quickly becomes unbounded as $\pi(s,a)$ approaches zero. Unlike FAQL, here the computation *time* (not the *value* of the update) is what becomes unbounded as $\pi(s,a)$ approaches zero. In the remainder of this section we will derive a closed-form expression for the

5. In our experiments we take the effective learning rate into account when comparing FAQL to our algorithm RUQL, setting $\alpha_{FAQL} = \beta_{FAQL} = \sqrt{\alpha_{RUQL}}$.

RUQL algorithm. The closed-form expression removes the need for actually repeating any update and therefore makes RUQL computationally feasible.

Theorem 1 *The RUQL update rule can be approximated with an error of $O(\alpha^2)$ as an instance of QL with learning rate (step size) z given by the equation*

$$z_{\pi^t(s,a)} = 1 - [1-\alpha]^{\lfloor \frac{1}{\pi(s,a)} \rfloor},$$

where α is a constant, denoting the learning rate of the underlying QL update equation being repeated.

Proof

For convenience, let $Q_i(s,a)$ refer to the intermediary value of the Q function after i iterations of the loop in Steps 8-10 in Algorithm 2. Before starting the loop, we have $Q_0(s,a) \leftarrow Q^t(s,a)$, i.e. before executing any iterations the value of $Q(s,a)$ is set to the value of the previous time step t. Upon finishing the loop we get $Q^{t+1}(s,a) = Q_{\lfloor \frac{1}{\pi(s,a)} \rfloor}(s,a)$, or the new value of $Q(s,a)$ at time step $t+1$. Let us now trace the computation of $Q^{t+1}(s,a) = Q_{\lfloor \frac{1}{\pi(s,a)} \rfloor}$. After one iteration we have

$$Q_1(s,a) \quad = \quad [1-\alpha]Q_0(s,a) + \alpha[r + \gamma \max_{a'} Q_0(s',a')].$$

The second iteration results in

$$\begin{aligned} Q_2(s,a) &= [1-\alpha]\left([1-\alpha]Q_0(s,a) + \alpha[r+\gamma \max_{a'} Q_0(s',a')]\right) + \alpha[r+\gamma \max_{a'} Q_1(s',a')] \\ &= [1-\alpha]^2 Q_0(s,a) + [1-\alpha]\alpha[r+\gamma \max_{a'} Q_0(s',a')] + \alpha[r+\gamma \max_{a'} Q_1(s',a')]. \end{aligned}$$

Similarly, after $\lfloor \frac{1}{\pi(s,a)} \rfloor$ iterations

$$\begin{aligned} Q_{\lfloor \frac{1}{\pi(s,a)} \rfloor}(s,a) \quad = \quad & [1-\alpha]^{\lfloor \frac{1}{\pi(s,a)} \rfloor} Q_0(s,a) + \\ & [1-\alpha]^{\lfloor \frac{1}{\pi(s,a)} \rfloor - 1} \alpha[r+\gamma \max_{a'} Q_0(s',a')] + \\ & [1-\alpha]^{\lfloor \frac{1}{\pi(s,a)} \rfloor - 2} \alpha[r+\gamma \max_{a'} Q_1(s',a')] + \\ & \ldots \\ & \alpha[r+\gamma \max_{a'} Q_{\lfloor \frac{1}{\pi(s,a)} \rfloor - 1}(s',a')]. \end{aligned}$$

It is important to keep in mind that all the above repeated updates occur at the *same time step*. It is also worth noting that the updates are monotonic (i.e., either increasing or decreasing) between t and $t+1$. In the following, the floor notation is dropped for notational convenience. There are four cases that we need to consider (derivations are given in the Appendix):

Case 1, $s' \neq s$, or $\left(s' = s$ and $a \neq \arg\max_{a'} Q^t(s,a')\right)$ throughout the $\frac{1}{\pi(s,a)}$ iterations:

$$Q^{t+1}(s,a) = [1-\alpha]^{\frac{1}{\pi(s,a)}} Q^t(s,a) + \left[1 - (1-\alpha)^{\frac{1}{\pi(s,a)}}\right][r + \gamma \max_{a'} Q^t(s',a')] \quad (4)$$

Case 2, $s' = s$ and $a = \arg\max_{a'} Q^t(s,a')$ throughout the $\frac{1}{\pi(s,a)}$ iterations:

$$Q^{t+1}(s,a) = [1-\alpha(1-\gamma)]^{\frac{1}{\pi(s,a)}} \left(Q^t(s,a) - \frac{r}{1-\gamma}\right) + \frac{r}{1-\gamma} \quad (5)$$

Case 3, $s' = s$ and $a \neq \arg\max_{a'} Q^t(s,a')$ until iteration $i_\top$: Since the update is monotonic, iteration $i_\top$ can be determined, at which $Q_i(s,a)$ becomes larger than $\max_{a'} Q^t(s,a')$. Subsequently, we can compute $Q^{t+1}(s,a)$ by applying the remaining updates $\frac{1}{\pi(s,a)} - i_\top$ to the value of $\max_{a'} Q^t(s,a')$ according to Equation 5, since the updated action bears the maximum value for the remaining updates, i.e.,

$$Q^{t+1}(s,a) = [1-\alpha+\alpha\gamma]^{\frac{1}{\pi(s,a)} - i_\top} \left(\max_{a'} Q^t(s,a') - \frac{r}{1-\gamma}\right) + \frac{r}{1-\gamma}.$$

Case 4, $s' = s$ and $a = \arg\max_{a'} Q^t(s,a')$ until iteration $i_\perp$: Similar to Case 3, let $i_\perp$ be the iteration at which $Q_{i_\perp}(s,a)$ drops below $\max_{a'} Q^t(s,a')$. As a result, we can compute $Q^{t+1}(s,a)$ by applying the remaining updates $\frac{1}{\pi(s,a)} - i_\perp$ to the value of $\max_{a'} Q^t(s,a')$ according to Equation 4, i.e.

$$Q^{t+1}(s,a) = [1-\alpha]^{\frac{1}{\pi(s,a)} - i_\perp} \max_{a'} Q^t(s,a') + \left[1 - (1-\alpha)^{\frac{1}{\pi(s,a)} - i_\perp}\right][r + \gamma \max_{a'} Q^t(s',a')].$$

The computation time complexity of $i_\top$, $i_\perp$ and the update rule of RUQ-Learning is $O(1)$, a significant improvement over the naive implementation in Algorithm 2 with unbounded time complexity. However, the four cases and the corresponding update equations are not as simple as the original Q-learning update equation (Equation 1) or even the FAQL update equation (Equation 3). The following derivation will elucidate the difference between Equation 4 and Equation 5 and show that it is insignificant for small learning rates, such that solely Equation 4 can be used as an approximation in all cases. Consider the Taylor series expansion of both equations at $\alpha = 0$, using $(1-\alpha)^c = 1 - c\alpha + O(\alpha^2)$. In Case 2, when $Q^t(s,a) = \max_{a'} Q(s,a')$, Equation 4 becomes

$$\begin{aligned} Q^{t+1}(s,a) &= \left(1 - \frac{\alpha}{\pi(s,a)}\right) Q^t(s,a) + \frac{\alpha}{\pi(s,a)}[r + \gamma Q^t(s,a)] + O(\alpha^2) \\ &= \left(1 - \frac{\alpha(1-\gamma)}{\pi(s,a)}\right) Q^t(s,a) + \frac{\alpha}{\pi(s,a)} r + O(\alpha^2), \end{aligned}$$

while Equation 5 becomes

$$\begin{aligned} Q^{t+1}(s,a) &= \left(1 - \frac{\alpha(1-\gamma)}{\pi(s,a)}\right)\left(Q^t(s,a) - \frac{r}{1-\gamma}\right) + \frac{r}{1-\gamma} + O(\alpha^2) \\ &= \left(1 - \frac{\alpha(1-\gamma)}{\pi(s,a)}\right) Q^t(s,a) + \frac{\alpha}{\pi(s,a)} r + O(\alpha^2). \end{aligned}$$

In brief, Case 1 requires to apply Equation 4 anyway, and other cases only require to change to Equation 5 if $Q^t(s,a) = \max_{a'} Q(s,a')$. However, given $Q^t(s,a) = \max_{a'} Q(s,a')$ the lines above show that the difference between the equations is $O(\alpha^2)$, and hence negligible for small α,

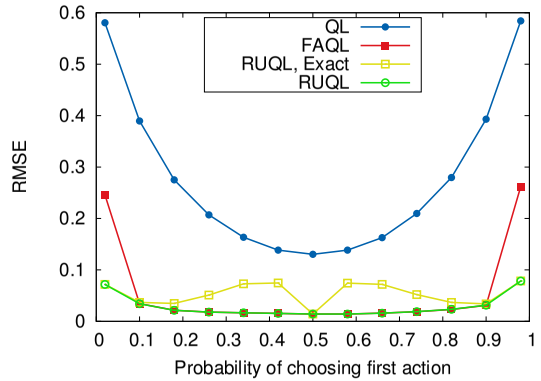


Figure 2: RMSE of QL, FAQL, RUQL (exact) and RUQL (approximate) plotted against the probability of choosing the first action $\pi(1)$. RUQL, approximated as in Theorem 1, has the lowest RMSE across different policies. The learning parameters are set as follows: $\alpha = 0.01$ for QL and RUQL and $\alpha_{FAQL} = \beta = \sqrt{\alpha} = 0.1$.

since higher order terms become insignificant. Therefore, Equation 4 can be applied in all four cases. In other words, RUQL can be approximated by

$$Q^{t+1}(s,a) = [1-\alpha]^{\frac{1}{\pi(s,a)}} Q^t(s,a) + \left[1-(1-\alpha)^{\frac{1}{\pi(s,a)}}\right][r + \gamma \max_{a'} Q^t(s',a')].$$

This equation can then replace Lines 8-10 in Algorithm 2 to produce a simple and efficient implementation of RUQL as an instance of QL with substitute learning rate $z_{\pi^t(s,a)} = 1-[1-\alpha]^{\frac{1}{\pi^t(s,a)}}$. ■

Algorithm 2 uses the floor notation, since fractional repetitions are not possible. However, the approximation alleviates this limitation, hence we adopt $z_{\pi^t(s,a)} = 1-[1-\alpha]^{\frac{1}{\pi(s,a)}}$ without flooring for both our experimental and theoretical analyses. The continuous and smooth definition of z_π will simplify the theoretical analysis we conduct later. More importantly, such smooth definition will handle fractions of $1/\pi$ more accurately. To illustrate this point we recall here the simple domain we used in the introduction section. The RUQL update equation is trying to imitate the QL update equation if (hypothetically) the agent can execute and update *every action at every time step*, or what we referred to as QL-ideal-update in the introduction. Figure 2 plots the RMSE of QL, FAQL, RUQL (exact) and RUQL (approximate) when compared to QL-ideal-update and using the same settings of the multi-armed bandit problem we explained in the introduction section.

It is clear from the figure that RUQL (approximate) has the lowest RMSE across different values of $\pi(1)$. In other words, independent of the underlying execution (exploration) strategy, RUQL results in action values that are the closest to the QL-ideal-update. Furthermore, as $\pi(1)$ goes beyond β in either direction the β limitation hits FAQL. Another interesting observation is that the RMSE of RUQL (exact) is higher than the RMSE of RUQL (approximate). The reason is the

inability of RUQL (exact) to handle partial iterations (i.e. if the ratio $\frac{1}{\pi}$ is not an integer). This is also why the performance of RUQL (exact) is best when $\pi(1) = 0.02, 0.1$, 0.5, 0.9, and 0.98 (with an almost perfect match with RUQL-approximate) in a clear verification of the validity of our approximation. At these points the boost in the learning rate that is given to the less probable action (with probability 0.02,0.1, and 0.5) is based on $\frac{1}{\pi}$ which is an integer in these cases. The further the probabilities are from these values, the further the RMSE of RUQL-exact from the RMSE of RUQL-approximate.

4. Theoretical Analysis

This section analyzes RUQ-Learning by comparing it with two closely-related algorithms: the original Q-learning and FAQL, the closest state of the art.

4.1 RUQL and QL

As stated by Theorem 1, RUQL can be approximated as an instance of QL with an effective learning rate $z_{\pi^t(s,a)} = 1-[1-\alpha]^{\frac{1}{\pi^t(s,a)}}$. However, while (traditionally) the learning rate of Q-learning, α, is independent of the policy, the *effective* learning rate of RUQL, z, is a function of the policy at the particular state s and action a at the given time t. For example, a common definition of the learning rate $\alpha(s,a,t) = \frac{1}{visits(s,a,t)}$, where $visits(s,a,t)$ is the number of times the learning agent visited state s and executed action a until time t. This is different from RUQL where the effective learning rate z includes the policy. The inclusion of the policy in z creates a feedback loop: the effective learning rate affects the learning which affects the policy which affects the effective learning rate.[6] As we show later in the experimental analysis, this results in significantly different learning dynamics (between RUQL and traditional QL) in non-stationary environments. In particular, we show that increasing the learning rate α even by 100 folds does not change the dynamics of QL significantly. On the other hand, using RUQL results in significantly different dynamics.

The dependence of the effective learning rate on the policy raises concerns of whether the convergence of Q-learning in stationary environments still holds for RUQL. We revisit here the main conditions for Q-learning convergence (Sutton and Barto, 1999), and consider whether the conditions still hold for RUQL and under what assumptions. For the original Q-learning two conditions concerning the learning rate need to be satisfied for convergence in MDPs: $\sum_t \alpha_t = \infty$ and $\sum_t (\alpha_t)^2 < \infty$. The second condition requires that the learning rate decays over time, while the first condition ensures that such decay is slow enough to allow the agent to learn the optimal Q values. One possible definition of α that satisfies both conditions is $\alpha_t = \frac{1}{t}$.

Extending the two conditions to RUQL, we substitute z for α_t in the learning rate conditions to get $\sum_t z_{\pi^t} = \infty$ and $\sum_t z_{\pi^t}^2 < \infty$. Expanding the first condition yields

$$\sum_t \left(1-[1-\alpha_t]^{\frac{1}{\pi^t(s,a)}}\right) = \infty.$$

Using polynomial expansion of z

$$z_{\pi^t} = 1-[1-\alpha_t]^{\frac{1}{\pi^t(s,a)}} = 1-\left(1+\frac{1}{\pi^t(s,a)}(-\alpha_t)+O(\alpha_t^2)\right) = \frac{\alpha_t}{\pi^t(s,a)} - O(\alpha_t^2).$$

6. In the case of a multi-agent system, this feedback loop is affected by other agents (because other agents affect the individually learned policy).

Subsequently, omitting the higher order terms[7] denoted by $O(\alpha_t^2)$ and substituting in the first condition leads to

$$\sum_t \left[\alpha_t \frac{1}{\pi^t(s,a)}\right] = \infty.$$

It is worth noting that $\pi^t(s,a)$ is bound away from zero since the sequence of updates proceeds over solely those actions that are selected with positive probability. This implies that for any sequence α_t for which $\sum_t \alpha_t = \infty$, the first condition also holds for the effective learning rate z_π of RUQL, since each summand is at least as large as before. We now move to the second condition:

$$\sum_t \left(1 - [1-\alpha_t]^{\frac{1}{\pi^t(s,a)}}\right)^2 < \infty.$$

Again using the polynomial expansion yields

$$\sum_t \left[\alpha_t \frac{1}{\pi^t(s,a)}\right]^2 < \infty.$$

Here we observe a clear distinction between RUQL and the original QL. Unlike the original Q-learning, we need to impose restrictions on the policy to ensure the above condition. Let us denote the minimum probability of choosing an action at a given time with $\epsilon^t = \min_{s,a} \pi^t(s,a)$ (which reflects the exploration rate). We then need to ensure that

$$\sum_t \left[\frac{\alpha_t}{\epsilon^t}\right]^2 < \infty.$$

To clarify how the above conditions can be used, consider the following example. Suppose we are using ϵ-greedy exploration strategy and $\alpha_t = \frac{1}{t}$, then one possible exploration rate that satisfies the second condition is $\epsilon^t = \frac{1}{\lg t}$.[8] Similarly, we can ensure minimum exploration using Boltzmann exploration through updating the temperature τ using the learned action values (Achbany et al., 2008).

It is worth noting that in practice, and particularly in non-stationary environments, the exploration rate parameter (which is ϵ in ϵ-greedy and τ in Boltzmann) is usually held to small constant values. In such cases, the second condition is trivially satisfied,[9] which means RUQL is guaranteed to converge to the optimal Q values. Our experiments confirm that RUQL works well in these environments.

4.2 RUQL and FAQL

Both RUQL and FAQL provide an algorithmic implementation to address the policy-bias of Q-learning. The two algorithms resolve this problem in different ways: FAQL normalizes the learning rate of the Q-learning update while RUQL repeats the Q-learning update rule. Despite their difference in implementation, RUQL learning dynamics are similar to FAQL learning dynamics when the

7. Omitting $O(\alpha_t^2)$ is justified because α_t is assumed to decay over time to establish convergence in stationary environments, and for small α_t the first order term dominates diminishing higher order terms.
8. The series $\sum_t \left[\frac{\lg t}{t}\right]^2$ is convergent using the Cauchy condensation test.
9. For constant exploration rates, ϵ^t is bound away from zero and can be replaced by a constant that represents the minimum positive value it assumes. RUQL thus satisfies the condition for the same α^t as QL.

probability of choosing actions are larger than β, otherwise FAQL learning dynamics deviate from RUQL learning dynamics. This can be shown through the following arguments.

Let $\Delta Q^t_{\text{algorithm}}(s,a) = Q^{t+1}(s,a) - Q^t(s,a)$ denote the update step for each of the algorithms FAQL and RUQL. Then the difference between the updates of FAQL and RUQL is $d_{\text{FAQL,RUQL}} = \Delta Q^t_{\text{FAQL}}(s,a) - \Delta Q^t_{\text{RUQL}}(s,a) = (w_{\pi^t(s,a)} - z_{\pi^t(s,a)})\mathbf{e}^t(s,a)$, where the Q-value error is denoted by $\mathbf{e}^t(s,a) = \left[r + \gamma Q^t(s', a_{\max}) - Q^t(s,a)\right]$, $w_{\pi^t(s,a)} = \min(1, \frac{\beta}{\pi(s,a)})\alpha_{FAQL}$ represents FAQL's effective learning rate, and $z_{\pi^t(s,a)} = 1 - [1-\alpha_{RUQL}]^{\frac{1}{\pi^t(s,a)}}$ represents RUQL's effective learning rate. Further working out the Taylor expansion at $\alpha = 0$ of the term $(1-\alpha_{RUQL})^{\frac{1}{\pi(s,a)}}$ from RUQL's update equation yields

$$[1-\alpha_{RUQL}]^{\frac{1}{\pi(s,a)}} = 1 - \frac{\alpha_{RUQL}}{\pi(s,a)} + O(\alpha^2_{RUQL}).$$

Therefore, the learning rate of RUQL can be simplified to.

$$\begin{aligned} z_{\pi^t(s,a)} &= 1 - \left(1 - \frac{\alpha_{RUQL}}{\pi(s,a)} + O(\alpha^2_{RUQL})\right) \\ &= \frac{\alpha_{RUQL}}{\pi(s,a)} + O(\alpha^2_{RUQL}). \end{aligned}$$

If $\mathbf{e}^t(s,a) = 0$, then both algorithms keep the Q-values unchanged, i.e.,

$$\Delta Q^t_{\text{FAQL}}(s,a) = \Delta Q^t_{\text{RUQL}}(s,a) = 0.$$

Otherwise, the difference in updates can be rewritten for small learning rates α_{RUQL} as

$$\frac{d_{\text{FAQL,RUQL}}}{\mathbf{e}^t(s,a)} = w_{\pi^t(s,a)} - z_{\pi^t(s,a)} = \min(1, \frac{\beta}{\pi(s,a)})\alpha_{FAQL} - \frac{\alpha_{RUQL}}{\pi(s,a)}.$$

For $\beta \cdot \alpha_{FAQL} = \alpha_{RUQL}$ the two algorithms yield equal updates if $\pi(s,a) \geq \beta$. However, when $\pi(s,a) < \beta$ the learning rates differ by $\frac{\alpha_{RUQL}}{\beta} - \frac{\alpha_{RUQL}}{\pi(s,a)}$, which is due to the β limitation of FAQL as we mentioned before.

5. Related Work

Several modifications and extensions to Q-learning were proposed, such as individual Q-learning (Leslie and Collins, 2005), the CoLF and CK heuristics (de Cote et al., 2006), and the utility-based Q-learning (Moriyama et al., 2011). Some of the proposed extensions aimed at improving the performance of Q-learning in specific domains (such as promoting cooperation in public good games (de Cote et al., 2006; Moriyama et al., 2011)), rather than addressing the policy-bias limitation of Q-learning. There also have been some works on modifying the learning rate α to speed up convergence (George and Powell, 2006; Dabney and Barto, 2012; Mahmood et al., 2012). Most of the proposed approaches defined different decaying functions for the learning rate (for example, as a function of time or the number of visits to a state) that are *independent* of the underlying policy (George and Powell, 2006). More recent approaches proposed more sophisticated methods for

adapting the learning rate based on the error (Dabney and Barto, 2012; Mahmood et al., 2012). None of the proposed approaches addressed the policy-bias problem.

Some learning algorithms specifically targeted non-stationary environments (da Silva et al., 2006; Noda, 2010; Kaisers and Tuyls, 2010). One technique devised a mechanism for updating the learning rate based on gradient descent (to minimize error) (Noda, 2010). Aside from being more complex than our simple update equation, the approach could only handle stateless domains. The RL-Context-Detection (RL-CD) algorithm (da Silva et al., 2006) was model-based and assumed finite set of stationary contexts that the environment switches among. For each stationary context a model was learned. This can be contrasted to our approach, which is model-free and tracks the non-stationarity of the environment without assuming a finite set of contexts. This is important when the non-stationarity is a result of an adaptive opponent (with potentially infinite contexts). The more recently proposed Frequency Adjusted Q-Learning (Kaisers and Tuyls, 2010) aimed to address the policy-bias limitation of Q-learning, but as we have shown, FAQL suffers from the β-limitation.

Q-learning was originally designed for single-agent environment, but yet performed adequately in multi-agent environments. HyperQ-learning was proposed as an extension to Q-learning in order to support multi-agent contexts (Tesauro, 2004). Unlike our approach, HyperQ attempts to learn the opponents' strategies without adapting the learning rate. We believe it is possible to integrate RUQL with HyperQ, but it is beyond the scope of this paper and remains an interesting future direction. Studying the dynamics of learning algorithms in multi-agent contexts has been an active area of research (Abdallah and Lesser, 2008; Babes et al., 2008; Bowling and Veloso, 2002; Claus and Boutilier, 1998; de Cote et al., 2006; Eduardo and Kowalczyk, 2009; Lazaric et al., 2007; Leslie and Collins, 2005; Moriyama, 2009; Tuyls et al., 2006; Vrancx et al., 2008; Wunder et al., 2010; Galstyan, 2013). The dynamics of Q-learning with Boltzmann exploration received attention earlier due to the continuous nature of the Boltzmann exploration. One of the earliest analyses of Q-learning assumed agents keep record of previous interactions and used Boltzmann exploration strategies (Sandholm and Crites, 1996). More recent analysis of Q-learning with Boltzmann exploration used evolutionary (replicator) dynamics (Kaisers and Tuyls, 2010; Lazaric et al., 2007; Tuyls et al., 2006). Similarly, Q-learning with ϵ-greedy exploration has been analyzed theoretically in stateless games (Eduardo and Kowalczyk, 2009; Wunder et al., 2010).

Another relevant and somewhat recent area of research is adversarial MDPs, where an oblivious adversary may choose a new reward function (and possibly the transition probability function as well) at each time step (Yu et al., 2009; Abbasi et al., 2013; Neu et al., 2010). A central assumption in that work is that the agent observes the full reward function (not only the current reward sample), and possibly the transition probability function once a change in the environment occurred. Furthermore, some of the approaches could not handle deterministic environments, as a mixed policy is assumed (Abbasi et al., 2013). Our approach (similar to Q-learning) does not make such strong assumptions.

Aside from Q-learning and its variations and extensions, several reinforcement learning algorithms have been proposed to optimize the exploration and achieve more efficient learning (Strehl et al., 2009; Jaksch et al., 2010). Aside from being more complex than Q-learning and RUQL, the algorithms that optimize exploration do not address the policy-bias problem. We view such algorithms as complementary to RUQL, which reduces the coupling between the learning dynamics and the underlying exploration strategy. Similar to the possible integration between RUQL and gradient-based multi-agent learning algorithms, RUQL can also be integrated with algorithms that optimize the exploration strategy.

The remainder of this section explains the two algorithms DynaQ and Double Q in more detail. These algorithms are more relevant to RUQL and serve as a benchmark comparison in our experiments.

5.1 Dyna-Q (DynaQ)

The Dyna-Q learning algorithm (Sutton, 1990) integrated the use of a model with the traditional Q-learning update. In its simplest form (which was used in the original paper experiments, as well as our experiments), Dyna-Q can be expressed as shown in Algorithm 3. Note that we intentionally framed DynaQ in such a way that highlights the similarities and differences with RUQL. From the algorithm, we note that similar to the original RUQL, DynaQ repeats the Q-learning update using the learned model. However, there are crucial differences. DynaQ repeats the updates *independent* of the underlying policy, in clear contrast to RUQL which repeats the update inversely proportional to the policy. Also there is no closed form equation that approximates DynaQ, and therefore it can be orders of magnitudes slower than QL (depending on the value of parameter k). Our intuition is that DynaQ is equivalent to uniformly increasing the effective learning rate, but we are not aware of any thorough theoretical analysis of DynaQ that established such connection (unlike our theoretical analysis of RUQL, which highlighted how RUQL affects the effective learning rate).

Algorithm 3: Dyna-Q

begin
  Initialize function Q arbitrarily and list L to empty.
  Observe the current state s.
  for *each time step* **do**
    Compute the policy π using Q.
    Choose an action a according to agent policy π.
    Execute action a and observe the resulting reward r and the next state s'.
    Update Q using Equation 1 and the information $\langle s, a, s', r\rangle$.
    Add the experience tuple $\langle s, a, s', r\rangle$ to L
    for $i : 1 \leq i \leq k$ **do**
      retrieve an experience tuple $\langle s, a, s', r\rangle$ uniformly at random from list L
      Update Q using Equation 1 and the information $\langle s, a, s', r\rangle$.
    end
    Set $s \leftarrow s'$.
  end
end

5.2 Double Q-Learning (DoubleQ)

An interesting recent variation of Q-learning is Double Q-learning (DQL) (Hasselt, 2010). DQL attempts to address how Q-learning may overestimate action values due to the max operator in the Q-learning equation. DQL counters overestimation of action values by maintaining two separate action values estimates, Q_A and Q_B, for each state-action pair. Whenever an action value is to be updated, either Q_A or Q_B is chosen uniformly at random to be updated. The update is done

using the traditional QL update equation, Equation 1, with only one difference: the max operator is applied to the action values of the other table. For example, suppose we chose Q_A is to be updated, then we apply Equation 6 as follows:

$$Q_A^{t+1}(s,a) \leftarrow Q_A^t(s,a) + \alpha \left(r + \gamma\, Q_B^t(s', a^*) - Q_A^t(s,a)\right), \tag{6}$$

where $a^* = \arg\max_{a'} Q_A^t(s', a')$. In other words, for the next state s', the best action is determined using the action value table to be updated. However, the *value* of the best action is determined using the action value in the other table. It is important to note that DoubleQ is *identical* to QL in stateless domains, where $\gamma = 0$. Therefore, the curve corresponding to DoubleQ in Figure 2 would be identical to the QL curve (DoubleQ suffers from the same policy bias as QL, at least for stateless domains).

6. Experimental Analysis

To ensure thorough evaluation of the RUQL algorithm, our analysis spans over several domains. The first two domains focus on verifying the theoretical arguments that we have made in Section 4 using simple, single-state, and non-stationary domains. The first domain (Section 6.1) uses a variation of the multi-armed-bandit problem to expose the effect of QL limitations in dynamic environments. The second domain (Section 6.2) aims at verifying the equivalence of RUQL and FAQL for sufficiently small values of the learning rate α. The subsequent three domains evaluate RUQL in multi-state, non-stationary, and noisy domains.

6.1 Multi-Armed Bandit With Switching Means

In order to tease out the differences between RUQL and FAQL, we propose a simple variation of the multi-armed bandit (MAB) problem (Robbins, 1952). Consider the basic MAB problem: an agent can choose between two actions. Each action i has a mean payoff $\mu_i \in \{0, 1\}$ with some random noise. Since the agent is not aware which action is the best (with expected payoff of 1), the agent needs to explore the two actions while learning the expected returns for each action.

We made two modifications to the traditional MAB problem. The first modification is *switching* the value of μ for the two actions every D time steps. In other words, having $D = 500$ means that from time 0 to time 499 $\mu_0 = 0$ and $\mu_1 = 1$ then from time 500 to time 999 $\mu_0 = 1$ and $\mu_1 = 0$ and so on. Allowing μ to switch models a non-stationary environment. The second modification is, instead of the usual Gaussian noise, we use an asymmetric noise signal. Let r_0 be the sample reward of the action with mean payoff of 0, and r_1 be the sample reward of the action with mean payoff of 1. The sample rewards r_0 and r_1 are defined as

$$r_0 = \begin{cases} \frac{-1}{1-p_0}, & \text{with probability } 1 - p_0. \\ \frac{1}{p_0}, & \text{otherwise (with probability } p_0), \end{cases}$$

$$r_1 = \begin{cases} \frac{2}{1-p_1}, & \text{with probability } 1 - p_1. \\ \frac{-1}{p_1}, & \text{otherwise (with probability } p_1), \end{cases}$$

where both p_0 and p_1 are parameters for controlling noise. The first thing to note here is that the expected payoffs remain 0 and 1, respectively. The value of p_1 determines how misleading the optimal action can be. For instance, with $p_1 = 0.01$ the optimal action provides (with very low probability) a payoff of -100, a payoff that is much lower than the expected payoff of the sub-optimal action. This type of noise targets the policy bias weakness of Q-learning. Once the optimal action receives such negative sample reward, and unless the learning rate is very small, the optimal action will not be tried for a long time. However, if the learning rate is very small, then Q-learning will respond poorly to changes in the environment. On the other hand, the value of p_0 determines how misleading the sub-optimal action can be. For example, with $p_0 = 0.01$ the sub-optimal action provides (with very low probability) a payoff of 100, which is much higher than the expected payoff of the actual optimal action. This type of noise targets the weakness of the RUQL algorithm, which gives a boost in the learning rate for infrequent actions.

The modified domain allows us to study two desired properties of learning algorithms: responding (quickly) to genuine change in the payoff *mean* of a particular action while resisting hasty response to stochastic payoff samples. Our definition of noise also resembles realistic situations. For example, the action of investing one's own money in gambling resembles a sub optimal action that has negative expected reward, while (very rarely) producing very high reward samples (very low p_0). Similarly, the action of investing money in stocks has positive expected reward with rare significant losses. It is worth noting that several techniques were proposed specifically to solve the non-stationary multi-armed bandit problem (Garivier and Moulines, 2011; Besbes et al., 2014). We are using the multi-armed bandit problem as a simple domain to understand the benefits and limitations of our approach. Our evaluation, therefore, include domains other than the multi-armed bandit problem (some of which are multi-state domains).

We conducted the experiments over a range of parameter values, including $D \in \{250, 500, \infty\}$, $p_0, p_1 \in \{0, 0.003, 0.01, 0.03, 0.1\}$, $\tau \in \{0.1, 0.3, 1\}$, $\alpha \in \{0.001, 0.01, 0.1, 1\}$,[10] and $\gamma = 0$. For each combination we collected the average payoff over 2000 consecutive time steps. The initial Q values are individually initialized randomly to either 0 or 1. Finally, statistical significance of the difference in the mean is computed over 100 independent simulation runs, using two-tail two-sample t-tests, where the null hypothesis is rejected at p-value 0.05.

In the absence of noise ($p_0 = p_1 = 0$), the only reason for change in a sampled payoff is the switch of μ. Therefore, setting $\alpha = 1$ achieves the best performance across the three algorithms, rendering all of them equivalent. Interestingly, increasing the value of p_0 (while the value of p_1 remains zero) does not cause the setting $\alpha = 1$ to be inferior. The reason is to be found in the QL policy bias. Increasing p_0 means that the suboptimal action will produce exceptionally high reward very rarely. With $\alpha = 1$, QL will naively, and from only one sample, think that the suboptimal action is the optimal one. As a result, QL will almost certainly choose the suboptimal action the following time step, which will most likely result in a disappointment. Therefore, QL will almost immediately correct its expectation. The same goes for RUQL and FAQL with $\alpha = 1$.

Table 1 shows the results for $p_1 = 0.01$ and $p_0 = 0.01$, for different values of D and α. For each algorithm, we report the best average payoff it achieved across the different values of the exploration parameter τ. Bold entries confirm the statistical significance of the outperforming algorithm(s). For example, if only RUQL is bold, then this means RUQL outperforms both QL and FAQL with statistical significance. If both RUQL and FAQL are bold, then this means both algorithms outperform QL with statistical significance but there is no statistical significance regarding the difference between FAQL and RUQL. Notice here the significant drop in performance for high α. The drop is more severe for stationary environments, which may appear counter intuitive.

10. Note that these values of α are for QL and RUQL. For FAQL, we used the square root of these α values for both α_{FAQ} and β_{FAQ}, so that FAQL has an effective learning rate $\alpha_{FAQ} \cdot \beta_{FAQ} = \alpha$.

D \ α	0.001	0.01	0.1	1
∞: QL	0.98	0.69	0.09	0.06
RUQL	0.97	**0.89**	0.09	0.06
FAQL	0.97	0.82	0.08	0.06
500: QL	0.51	0.58	0.37	0.53
RUQL	0.54	**0.72**	0.37	0.54
FAQL	0.53	**0.70**	0.37	0.53
250: QL	0.52	0.55	0.48	0.53
RUQL	0.52	**0.62**	0.48	0.53
FAQL	0.51	0.57	0.48	0.53

Table 1: The mean payoff for the MAB-switching game, with $p_0 = p_1 = 0.01$, and for different values of D and α. The reported value in each cell is the best value we obtained across the different values of the exploration rate τ. The first observation is the significant drop in performance when α increases beyond 0.01. The policy bias results in the optimal action being rarely tried after the optimal action receives a big penalty. For stationary environments ($D = \infty$), RUQL's performance is comparable to QL with no statistically significant difference for $\alpha = 0.001$. As alpha increases to 0.01, the policy bias affects QL significantly, even in stationary environments. RUQL is affected as well, but to a lesser degree. As alpha increases to 0.1 and above, the effect of noise becomes so significant that all algorithms perform poorly. Furthermore, RUQL achieves superior performance in non-stationary environments when $D < \infty$.

What happens here is that when the optimal action receives the rare negative reward, if α is high, then the corresponding Q value becomes much smaller than the Q-value of the suboptimal action, making it very unlikely that the optimal action will ever be tried. When the environment becomes non-stationary, QL's performance improves simply because the suboptimal action becomes optimal after the switch. RUQL reduces the severity of policy bias, which results in better performance in non-stationary cases ($D = 500$ and $D = 250$). It is important to note also that in the presence of noise, and for non-stationary environments, α should neither be too small nor too large. In this experiment, $\alpha = 0.01$ resulted in the most robust performance.

To have a closer look at the learning dynamics, Figure 3 illustrates a sample run with $\alpha = 0.01, D = 500, p_1 = p_0 = 0.01$ and $\tau = 0.3$. The figure compares the accumulated payoff for the three algorithms, using the same random seed (hence the synchronized noise). The three algorithms quickly learn to select Action 1, with payoff given by r_1 (hence the steady increase in the accumulated payoff). The first sudden drop in the accumulated payoff (around Time 50) is due to the noise defined by r_1. Notice slight gain of RUQL as a result of this noise. The reason is that the sudden drop reduces the probability of choosing the optimal action. The reduction in the probability results in subsequent small boost in the learning rate for RUQL (and FAQL), hence the quicker recovery of both RUQL and FAQL against QL. Now consider Time 500, when the first switch of the actions' expected payoff occurs and now Action 1 generates payoff according to r_0. We can see that both RUQL and FAQL initially continue to choose Action 1 (initial drop in the accumulated payoff). The probability of choosing Action 0, which is now the optimal action, is very low. RUQL quickly recovers while FAQL (due to the β-limitation) takes longer time to recover. This results in a bigger gain for RUQL.

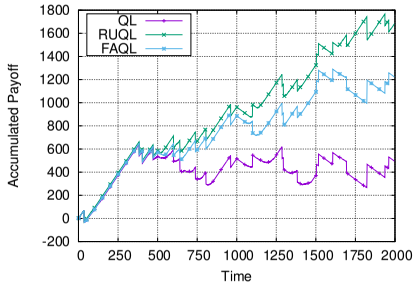


Figure 3: Comparison of the accumulated payoff for the three algorithms in a sample run (with the same random seed), using $\alpha = 0.01, D = 500, p_1 = p_0 = 0.01$ and $\tau = 0.3$.

D	$p_0 = p_1$ \ Alg.	0.003	0.01	0.03	0.1
∞	QL	0.96	0.98	0.95	1
	RUQL	0.98	0.97	0.97	1
	FAQL	0.98	0.97	0.97	1
500	QL	0.56	0.58	0.55	0.72
	RUQL	**0.65**	**0.72**	**0.6**	0.71
	FAQL	**0.62**	**0.70**	**0.6**	0.71
250	QL	0.51	0.55	0.52	0.65
	RUQL	**0.58**	**0.62**	**0.56**	0.64
	FAQL	**0.57**	0.57	0.55	0.66

Table 2: The mean payoff for the MAB-switch game, for different values of D, p_0 and p_1. Each cell represents the average payoff, maximized over α and τ.

Table 2 compares the performance of the three algorithms for different levels of noise (p_i) and non-stationary environments (D). The performance reported for each algorithm is the maximum over α and τ.[11] We can observe here that for stationary environment, RUQL performs as well as QL, with no statistical significance for the difference between the three algorithms. As the environment becomes stationary and noisy, RUQL outperforms QL and FAQL with statistical significance for several settings. Even for the cases where RUQL does not outperform QL (or FAQL), RUQL has comparable performance with no-statistical significance of the difference.

6.2 Prisoner's Dilemma Game

This section compares the (experimental) dynamics of Q-learning, FAQL, and RUQL in the Prisoner's Dilemma game (PD). We are using the PD domain primarily for benchmarking, as it was

11. It is worth noting that for the majority of the cells, $\alpha = 0.01$ and $\tau = 0.3$ resulted in best performance for all three algorithms.

used before in evaluating FAQL (Kaisers and Tuyls, 2010). Each agent is oblivious to the existence of the other agent, therefore the domain can be viewed as a non-stationary environment from the perspective of each individual agent. For all the three algorithms (QL, FAQL, RUQL) the temperature for Boltzmann exploration was set to $\tau = 0.1$. The discount factor γ was set to 0. The learning rate α was set at 10^{-6} for both Q-learning and RUQL, while for FAQL both α and β were set to 10^{-3}. Table 3 shows the payoff values for the PD game. The above setting was used before in the literature (Kaisers and Tuyls, 2010) and we use the same setting here for comparability.

	c	d
c	3,3	0,5
d	5,0	1,1

Table 3: The prisoner's dilemma game with actions cooperate (c) and defect (d).

Figure 4 shows the dynamics of the three algorithms for three different initial value levels of Q. The dynamics of Q-learning and FAQL are consistent with the results reported before (Kaisers and Tuyls, 2010). It is worth noting that under pessimistic initial Q-values (centered around zero, top row), Q-learning approaches the Pareto optimal joint action, and hence a strictly dominated action, simply because it is initialized near this joint action, updates are self-reinforcing and low initial values lead to extremely low exploration probabilities. Given infinite time, the algorithms would converge to the Nash equilibrium, since the defection action would eventually catch up due to its factual higher value against cooperation. In contrast, the dynamics of both FAQL and RUQL show consistent behavior across different initializations, and they are very similar and close to the dynamical system of the infinitesimal limit (approximating that all actions were updated at every iteration), despite the conceptually different update equations. A plot of the infinitesimal limit is omitted since it would be indistinguishable from FAQL and RUQL, but illustrations can be found in related work (Kaisers and Tuyls, 2010).

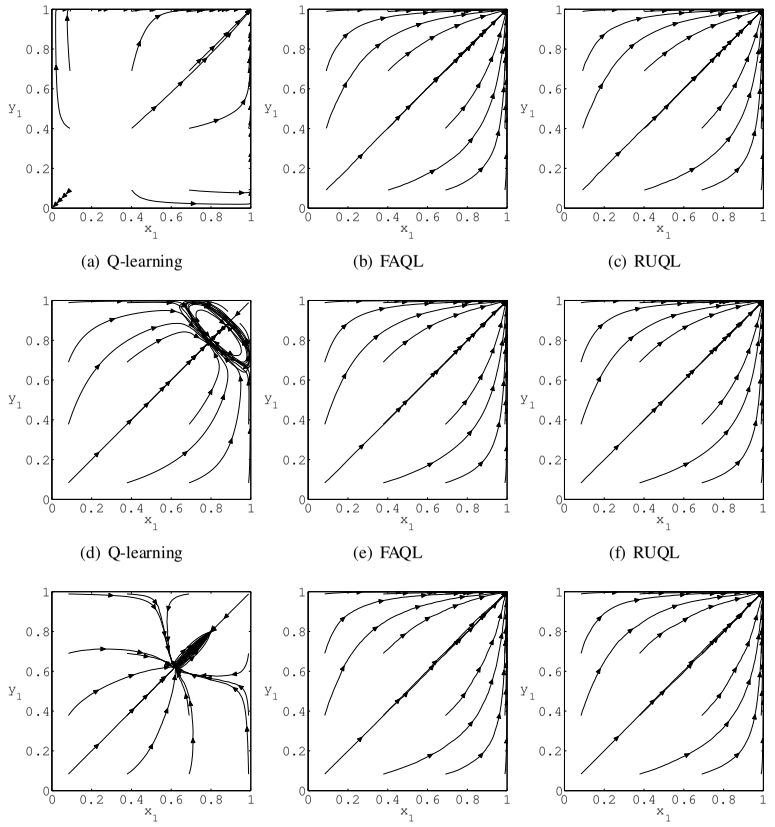
To demonstrate that the effect of RUQL is not equivalent to simply scaling the learning rate of Q-learning, Figure 5 shows the behavior dynamics of Q-learning when increasing the learning rate up to 100 times. As we can see, the general gross features of the dynamics remain the same. Increasing the learning rate only increases the step size in policy space without changing the expected direction.

6.3 1-Row Domain

This section presents a simple multi-state domain to test RUQL's multi-state performance. The domain is depicted in Figure 6 [a]. There are 10 total states, which are organized in one row (hence the name). There are two actions: left and right. The agent starts at a random state (which is not goal) and uses the two actions to reach the goal state depicted by 'G'. Each step yields a reward of -1, and the expected reward for reaching the goal is

$$r_g = \begin{cases} \frac{11}{1-P}, & \text{with probability } 1-P \\ \frac{-1}{P}, & \text{otherwise (with probability } P\text{).} \end{cases}$$

The domain is episodic. An episode starts from any random state that is not a terminal state. A terminal state is either the goal, or the cell opposite from the goal. To study non-stationarity in the

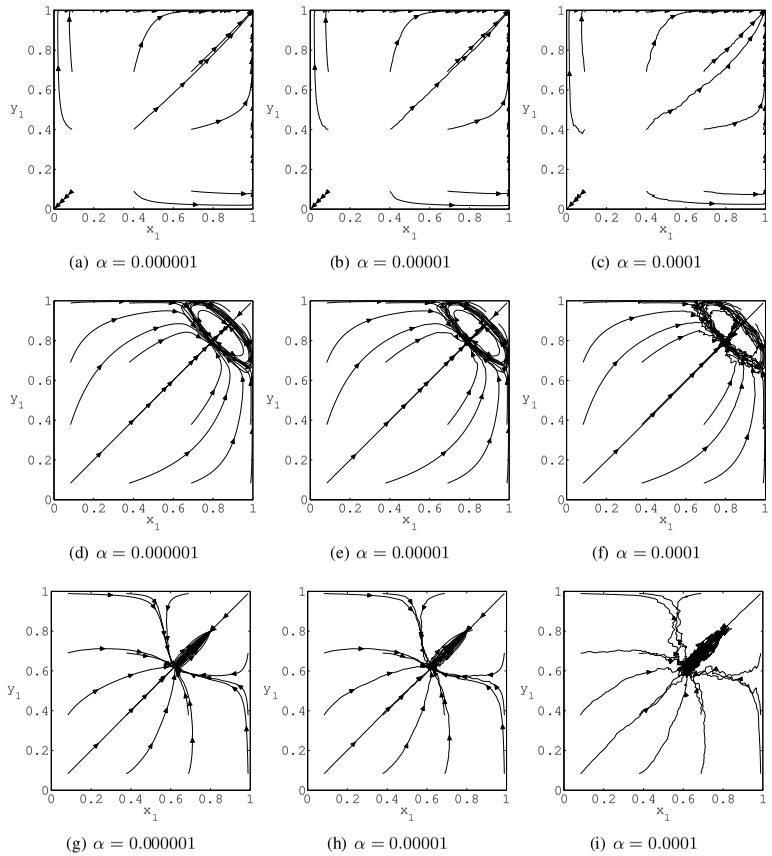


(a) Q-learning (b) FAQL (c) RUQL

(d) Q-learning (e) FAQL (f) RUQL

(g) Q-learning (h) FAQL (i) RUQL

Figure 4: The learning dynamics of Q-learning (left), FAQL (middle), and RUQL (right) for the PD game. Each figure plots the probability of defecting for Player 1 (x-axis) against the probability of defecting for Player 2 (y-axis). The rows show the dynamics when the initial Q-values are centered around 0, 2.5 and 5 (top, middle, bottom). For all figures, the algorithms were allowed to run for 5 million time steps. The learning rate α was set at 10^{-6} for both Q-learning and RUQL, while for FAQL both α and β were set to 10^{-3}. RUQL has dynamics very similar to FAQL, while Q-learning has erratic dynamics due to its policy bias.



(a) $\alpha = 0.000001$ (b) $\alpha = 0.00001$ (c) $\alpha = 0.0001$

(d) $\alpha = 0.000001$ (e) $\alpha = 0.00001$ (f) $\alpha = 0.0001$

(g) $\alpha = 0.000001$ (h) $\alpha = 0.00001$ (i) $\alpha = 0.0001$

Figure 5: The learning dynamics of Q-learning for different values of the learning rate α: 0.000001 (left), 0.00001 (middle), and 0.0001 (right) for the PD game. Each figure plots the probability of defecting for Player 1 (x-axis) against the probability of defecting for Player 2 (y-axis). The top row shows the dynamics when the initial Q-values are around 0, the middle row shows the dynamics when the initial Q-values are around 2.5, and the bottom row shows the dynamics when the initial Q-values are around 5. For all figures, the algorithms were allowed to run for $\frac{5}{\alpha}$ time steps.

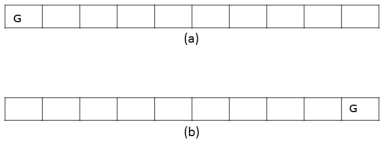


Figure 6: The two settings of the 1-row domain. G indicates the goal state.

domain, there are two settings (shown as [a] and [b] in Figure 6), where parameter D controls the duration before the environment switches from one setting to another.

We conducted the experiments over a range of parameter values, including $D \in \{5000, 10000, \infty\}$, $P \in \{0, 0.01, 0.1\}$, $\tau \in \{0.008, 0.04, 0.2, 1, 5\}$, $\pi_{min} \in \{0, 0.01, 0.001\}$, $\gamma = 1$, and $\alpha \in \{0.01, 0.04, 0.16, 0.64\}$.[12] For each combination we collected the average payoff over 20000 consecutive time steps. The initial Q values are initialized to 0. Finally, statistical significance of the difference in the mean is computed over 10 independent simulation runs, using two-tail two-sample t-test, where the null hypothesis is rejected at p-value 0.05.

Table 4 compares the performance of the three algorithms for different levels of noise (P) and non-stationary environments (D). The performance reported for each algorithm is the best over the different values of π_{min}, α and τ. We can observe similar results to the stateless domain. For noise-free and stationary environments, RUQL performs as well as the other algorithms, with no statistical significance for the difference. For noise-free non-stationary environments, DynaQ performs significantly better. The reason is that by repeating the updates, DynaQ is effectively increasing the learning rate for all actions, which is beneficial in noise-free environments. As the environment becomes non-stationary and noisy, RUQL consistently outperforms other algorithms, with FAQL and DoubleQ as close seconds. Dyna-Q fails in noisy environments because Dyna-Q effectively boosts the learning rate for *all* the actions (therefore, the Q-value of an optimal action may drop significantly upon receiving noisy negative reward).

6.4 Taxi Domain with Switching Stations

To investigate how RUQL works in multi-state environments we use a variation of the taxi domain (Dietterich, 2000). Figure 7 illustrates the domain. The world is a 5x5 grid that has 4 stations (represented by the letters) and walls (represented by the thick lines). A customer appears at a source station and is to be delivered to a destination station. Both the source and the destination stations are picked uniformly at random and can be the same. A taxi can pick a customer up, drop a customer, or navigate through the world, which results in a total of 6 actions: UP, DOWN, LEFT, RIGHT, PICK, and DROP. The state here is the taxi location (25 values) the customer location (5 values: 4 stations and in taxi) and destination (4 values) for a total of 500 states. Each navigation action receives a reward of -1. Hitting the wall results in no change in the state and the same reward of -1. Picking up or dropping off a customer receives a reward of 20 if at the right location, otherwise receives a reward of -10. We made two main modifications:

12. Note that these values of α are for QL, DoubleQ, DynaQ, and RUQL. For FAQL, we used the square root of these α values for both α_{FAQ} and β_{FAQ}, so that FAQL has an effective learning rate $\alpha_{FAQ}.\beta_{FAQ} = \alpha$.

D	Alg. \ P	0.0	0.01	0.1
5000	QL	0.77	0.74	0.83
	DynaQ	**1.31**	-0.01	0.02
	DoubleQ	0.85	0.75	0.80
	FAQL	0.81	**0.80**	**0.85**
	RUQL	0.84	**0.81**	**0.90**
10000	QL	0.81	0.74	0.84
	DynaQ	**1.34**	-0.16	0
	DoubleQ	0.95	0.74	**0.95**
	FAQL	0.88	**0.80**	0.86
	RUQL	0.90	**0.85**	**0.95**
∞	QL	1.18	**1.16**	**1.19**
	DynaQ	1.2	-0.17	0.36
	DoubleQ	1.20	**1.15**	**1.2**
	FAQL	1.18	**1.16**	**1.18**
	RUQL	1.18	**1.16**	**1.18**

Table 4: The mean payoff for the 1-Row Domain, for different values of D and P. Each cell represents the average payoff, maximized over α and τ.

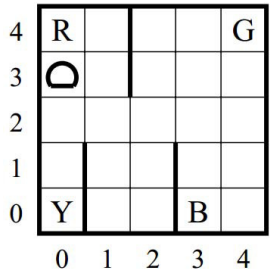


Figure 7: The Taxi domain (Dietterich, 2000).

- Rare but severe noise: PICK and DROP actions fail with probability p and result in penalty $\frac{R_p}{p}$.[13]
- Non-stationary domain: every duration D the stations (R,G,Y,B) rotate their locations (station R takes the location of station G which in turn takes the location of station Y, etc.).

We conducted the experiments over range of parameter values, including $D \in \{300000, 600000, \infty\}$, $\tau \in \{0.03, 0.1, 0.3, 1\}$, $\pi_{min} \in \{0, 0.001, 0.01\}$, $\alpha \in \{0.001, 0.01, 0.1\}$,[14] $p \in \{0, 0.01\}$, $R_p =$

13. For the navigation actions, the noise remains similar to the original Taxi domain: a navigation action fails with probability p and results in skidding to one of the sides with reward of -1.
14. Again the values of α are for QL and RUQL. For FAQL, we used the square root of these α values for both α_{FAQ} and β_{FAQ}, so that FAQL has an effective learning rate $\alpha_{FAQ}.\beta_{FAQ} = \alpha$.

D	Alg. \ p	0	0.01
8	QL	**2.35**	-0.61
	DynaQ	**2.34**	0.19
	DoubleQ	2.27	-1.23
	RUQL	**2.37**	**0.45**
	FAQL	**2.38**	0.29
600000	QL	**1.79**	-1.31
	DynaQ	**1.83**	**-0.42**
	DoubleQ	1.17	-1.91
	RUQL	**1.77**	**-0.47**
	FAQL	**1.80**	-0.58
300000	QL	0.64	-1.46
	DynaQ	**1.38**	-0.79
	DoubleQ	0.11	-1.92
	RUQL	1.19	**-0.70**
	FAQL	0.77	-0.94

Table 5: The mean payoff for the Taxi domain, for different values of p and D.

-6, and $\gamma = 1$. For each combination we collected the average payoff over 1200000 consecutive time steps. All Q-values are initialized to zeros. Finally, statistical significance of the difference in the mean is computed over 100 independent simulation runs, using two-tail two-sample t-test, where the null hypothesis is rejected at p-value 0.05.

Table 5 shows the results for different values of D and p. For each algorithm, we report the best average payoff it achieved across the different parameter values. Bold entries confirm the statistical significance of the outperforming algorithm, similar to the results shown in Section 6.1. We notice similarities to the previous domain. In noise-free environments, again DynaQ outperforms other algorithms. As the environment becomes noisy and non-stationary, the performance of DynaQ drops significantly, while RUQL outperforms all other algorithms. Interestingly, DoubleQ performs the worst in this domain, across the different settings. This is perhaps due to the reported underestimation of DoubleQ for the Q values (Hasselt, 2010). It is also worth noting that even in stationary but noisy setting, RUQL outperforms other algorithms in the taxi domain. The reason for this is that RUQL can boost the learning rate more effectively than Q-learning and other algorithms. For example, suppose the best action in a particular state is to drop the customer. However, due to noise, the payoff will be highly negative. This will result in sudden drop in the Q-value for the DROP action in this particular state, and consequently the probability of choosing DROP. However, RUQL will boost the learning rate for the DROP action once it is tried and succeeds, therefore recovering from the noise effect faster. This results in a cascading effect across states. Also because RUQL's boost diminishes as the action is chosen more frequently, the learning becomes stable in the face of the noise (which can not be achieved via simple increase of the learning rate for QL, or using DynaQ). This is consistent with previous work, which experimentally showed that converging to a policy is in some cases inferior to continuous learning (tracking) even in stationary environments (Sutton et al., 2007). As the environment becomes more dynamic ($D = 600000$ and $D = 300000$) the performance of all algorithms drop (as expected) but RUQL maintains the superior performance.

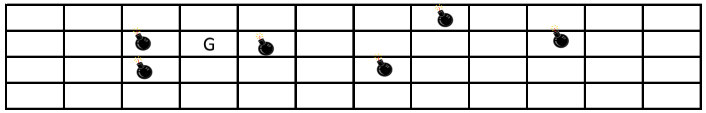


Figure 8: The Mines domain. The agent can navigate through the 4x16 grid, while avoiding the scattered mines and trying to reach the goal.

6.5 Mines Domain

The mines domain is illustrated by Figure 8 and is adopted from the RL-GLUE software (Tanner and White, 2009). The world consists of 4x16 grid, where the agent can navigate up, down, left and right (4 actions) and receives a reward of -1 unless it encounters a mine, in which case the reward is -100, or it reaches the goal state and receives the reward of 10. In this simple domain there are only 64 states. 6 mines in addition to the goal are scattered randomly in the grid. We modified the domain slightly from the original as follows:

- Rare but severe noise: reaching the goal state will fail with probability p and result in reward of $-1/p$, otherise will succeed and produce reward of $11/(1-p)$ (expected reward remains 10).
- Non-stationary domain: every duration D the goal state is randomly re-assigned.

We conducted the experiments over a range of parameter values, including $D \in \{25 \cdot 10^4, 50 \cdot 10^4, \infty\}$, $\tau \in \{0.008, 0.04, 0.2, 1, 5\}$, $p \in \{0, 10^{-2}\}$, $\alpha \in \{0.0025, 0.01, 0.04, 0.16\}$, $\pi_{min} \in \{0, 10^{-2}, 10^{-3}\}$, and $\gamma = 1$.[15] For each combination, we collected the average payoff over 10^6 consecutive time steps, and reported the best performance for each algorithm over the tested parameter values. All Q-values are initialized to zeros. Finally, statistical significance of the difference in the mean is computed over 20 independent simulation runs, using two-tail two-sample t-test, where the null hypothesis is rejected at p-value 0.05. Table 6 summarizes the results for different values of p and the duration D. We observe similar qualitative results to the previous domains. DynaQ is better in noise-free settings, with RUQL outperforming other algorithms as the environment becomes noisy and non-stationary. FAQL is close second in noisy and non-stationary environments as well, albeit RUQL is clearly superior in the most frequently changing benchmark environment with D=250,000.

7. Discussion of RUQL

Looking at the previous experimental results we can identify a few interesting observations. The main benefit of RUQL is reducing the policy-bias, which is reflected in better performance in non-stationary environments. In noise-free environments, however, the benefit of RUQL is diminishing, since for noise-free environments the learning rate can be as high as 1. As RUQL reduces the policy bias by boosting the learning rate for actions with low probability of being chosen, such

15. Again the values of α are for QL and RUQL. For FAQL, we used the square root of these α values for both α_{FAQ} and β_{FAQ}, so that FAQL has an effective learning rate $\alpha_{FAQ}.\beta_{FAQ} = \alpha$.

D	Alg. \ P	0	0.01
∞	QL	0.79	0.6
	DynaQ	0.78	0.35
	DoubleQ	0.78	0.54
	RUQL	0.79	**0.63**
	FAQL	0.79	**0.63**
500000	QL	0.70	0.43
	DynaQ	**0.77**	0.34
	DoubleQ	0.69	0.40
	RUQL	0.70	**0.49**
	FAQL	0.72	**0.48**
250000	QL	0.67	0.27
	DynaQ	**0.75**	0.29
	DoubleQ	0.63	0.39
	RUQL	0.65	**0.43**
	FAQL	0.66	0.33

Table 6: The mean payoff for the Mines domain, for different values of p and D.

benefit fades as the basic learning rate increases. In contrast, if the environment is noisier and changes more frequently, the benefits of RUQL become more apparent. This is consistent across the domains we have studied. Overall, RUQL retains the simplicity of QL, while showing robust performance across various experimental settings.

8. Conclusions

In this paper we proposed and evaluated the Repeated Update Q-learning algorithm, RUQL, which addresses the policy bias of Q-learning. Unlike the closest state-of-the-art algorithm, RUQL's behavior remains consistent even for policies with arbitrarily small probabilities of choosing actions. We show theoretically that our algorithm is guaranteed to converge to the optimal Q-values in single-agent stationary environments, while having better response to changes in non-stationary environments. We show experimentally how this results in superior performance in non-stationary and noisy environments, compared to 4 other algorithms.

Given its merits in non-stationary environments, RUQL may find interesting applications in multi-agent environments. There is a growing collection of algorithms that were designed specifically for multi-agent environments. The Win-or-Learn-Fast heuristic (Bowling and Veloso, 2002; Bowling, 2005) resulted in the first gradient-ascend-based (GAB) multi-agent learning algorithms that successfully converged to mixed Nash Equilibrium in small general-sum games with minimum knowledge of the underlying game (only the player's own payoff). This was followed by the more recent GAB algorithms (Abdallah and Lesser, 2008; Zhang and Lesser, 2010). Other multi-agent algorithms that assumed knowledge of the underlying game were also proposed and were able to converge in larger games (Hu and Wellman, 2003; Conitzer and Sandholm, 2007). All of these algorithms could benefit from improved value approximation techniques like the one presented in this paper. The integration of RUQL in specialized multi-agent learning algorithms is a promising avenue of future research.

Finally, it is important to note that RUQL is *not* intended as a replacement of Q-learning. In stationary environments with stochastic outcomes, QL may outperform RUQL, since the latter may slightly boost the learning rate of rarely chosen action. However, if the environment is actually changing over time (non-stationary and thus violating the core assumption of QL), then RUQL's boost may yield superior performance.

Acknowledgments

We would like to acknowledge support for this project from the Emirates Foundation Ref No: 2010-107 Sience & Engineering Research grant and British University in Dubai INF009 Grant.

Appendix A. Theoretical Derivations of Exact RUQL

Here we derive the different quantities that are needed for each of the four cases of the exact RUQL.

Case 1:

$$\begin{aligned}
Q_{\lfloor\frac{1}{\pi(s,a)}\rfloor}(s,a) &= [1-\alpha]^{\lfloor\frac{1}{\pi(s,a)}\rfloor}Q_0(s,a)+\alpha[r+\gamma\max_{a'}Q_0(s',a')] \\
&\quad \left(1+(1-\alpha)+(1-\alpha)^2+...+(1-\alpha)^{\lfloor\frac{1}{\pi(s,a)}\rfloor-1}\right) \\
&= [1-\alpha]^{\lfloor\frac{1}{\pi(s,a)}\rfloor}Q_0(s,a)+\alpha[r+\gamma\max_{a'}Q_0(s',a')]\frac{[1-(1-\alpha)^{\lfloor\frac{1}{\pi(s,a)}\rfloor}]}{1-(1-\alpha)} \\
&= [1-\alpha]^{\lfloor\frac{1}{\pi(s,a)}\rfloor}Q_0(s,a)+[1-(1-\alpha)^{\lfloor\frac{1}{\pi(s,a)}\rfloor}][r+\gamma\max_{a'}Q_0(s',a')]
\end{aligned}$$

We can remove the floor notation for a better generalization, and using the equalities $Q_0(s,a) = Q^t(s,a)$ and $Q_{\lfloor\frac{1}{\pi(s,a)}\rfloor}(s,a) = Q^{t+1}(s,a)$ we get (RUQL's main update rule):

$$Q^{t+1}(s,a) = [1-\alpha]^{\frac{1}{\pi(s,a)}}Q^t(s,a)+\left[1-(1-\alpha)^{\frac{1}{\pi(s,a)}}\right][r+\gamma Q^t(s',a_{max})]$$

Case 2:

$$\begin{aligned}
Q_1(s,a) &= [1-\alpha]Q^t(s,a)+\alpha\left(r+\gamma Q^t(s,a)\right) \\
&= [1-\alpha+\alpha\gamma]Q^t(s,a)+\alpha r \\
Q_2(s,a) &= [1-\alpha+\alpha\gamma]Q_1(s,a)+\alpha r \\
&= [1-\alpha+\alpha\gamma]\left([1-\alpha+\alpha\gamma]Q^t(s,a)+\alpha r\right)+\alpha r \\
Q_n(s,a) &= [1-\alpha+\alpha\gamma]^nQ^t(s,a)+\alpha r\left(1+[1-\alpha+\alpha\gamma]+\ldots+[1-\alpha+\alpha\gamma]^{n-1}\right) \\
&= [1-\alpha+\alpha\gamma]^nQ^t(s,a)+\alpha r\frac{1-[1-\alpha+\alpha\gamma]^n}{1-[1-\alpha+\alpha\gamma]} \\
&= [1-\alpha+\alpha\gamma]^nQ^t(s,a)+r\frac{1-[1-\alpha+\alpha\gamma]^n}{1-\gamma} \\
&= [1-\alpha+\alpha\gamma]^nQ^t(s,a)+(1-[1-\alpha+\alpha\gamma]^n)\frac{r}{1-\gamma}
\end{aligned}$$

$$\begin{aligned}
&= [1-\alpha+\alpha\gamma]^n\left(Q^t(s,a)-\frac{r}{1-\gamma}\right)+\frac{r}{1-\gamma} \\
Q^{t+1}(s,a) &= [1-\alpha+\alpha\gamma]^{\frac{1}{\pi(s,a)}}\left(Q^t(s,a)-\frac{r}{1-\gamma}\right)+\frac{r}{1-\gamma}
\end{aligned}$$

Case 3 using $Q^t(s,a'') = Q^t(s,a)$, where $a'' = \arg\max_{a'\in A\setminus a}Q^t(s,a')$:

$$\begin{aligned}
Q^t(s,a'') &= [1-\alpha+\alpha\gamma]^{i_\top}\left(Q^t(s,a)-\frac{r}{1-\gamma}\right)+\frac{r}{1-\gamma} \\
Q^t(s,a'')-\frac{r}{1-\gamma} &= [1-\alpha+\alpha\gamma]^{i_\top}\left(Q^t(s,a)-\frac{r}{1-\gamma}\right) \\
\frac{Q^t(s,a'')-\frac{r}{1-\gamma}}{Q^t(s,a)-\frac{r}{1-\gamma}} &= [1-\alpha+\alpha\gamma]^{i_\top} \\
i_\top &= \frac{\log\left(\frac{Q^t(s,a'')-\frac{r}{1-\gamma}}{Q^t(s,a)-\frac{r}{1-\gamma}}\right)}{\log(1-\alpha+\alpha\gamma)}
\end{aligned}$$

Case 4 using $Q^t(s,a'') = Q^t(s,a)$, where $a'' = \arg\max_{a'\in A}Q^t(s,a')$:

$$\begin{aligned}
Q^t(s,a'') &= [1-\alpha]^{i_\perp}Q^t(s,a)+\left[1-(1-\alpha)^{i_\perp}\right][r+\gamma Q^t(s,a'')] \\
Q^t(s,a'')-r-\gamma Q^t(s,a'') &= [1-\alpha]^{i_\perp}Q^t(s,a)-(1-\alpha)^{i_\perp}[r+\gamma Q^t(s,a'')] \\
(1-\gamma)Q^t(s,a'')-r &= [1-\alpha]^{i_\perp}\left[Q^t(s,a)-r-\gamma Q^t(s,a'')\right] \\
\frac{(1-\gamma)Q^t(s,a'')-r}{Q^t(s,a)-r-\gamma Q^t(s,a'')} &= [1-\alpha]^{i_\perp} \\
\frac{(1-\gamma)Q^t(s,a'')-r}{Q^t(s,a)-\gamma Q^t(s,a'')-r} &= [1-\alpha]^{i_\perp} \\
i_\perp &= \frac{\log\left(\frac{(1-\gamma)Q^t(s,a'')-r}{Q^t(s,a)-\gamma Q^t(s,a'')-r}\right)}{\log(1-\alpha)}
\end{aligned}$$

References

Yasin Abbasi, Peter L Bartlett, Varun Kanade, Yevgeny Seldin, and Csaba Szepesvari. Online learning in markov decision processes with adversarially chosen transition probability distributions. In *Annual Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 2508–2516. Curran Associates, Inc., 2013.

Sherief Abdallah and Michael Kaisers. Addressing the policy-bias of q-learning by repeating updates. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1045–1052, 2013.

Sherief Abdallah and Victor Lesser. A multiagent reinforcement learning algorithm with non-linear dynamics. *Journal of Artificial Intelligence Research*, 33:521–549, 2008.

Youssef Achbany, François Fouss, Luh Yen, Alain Pirotte, and Marco Saerens. Tuning continual exploration in reinforcement learning: An optimality property of the boltzmann strategy. *Neurocomputing*, 71(13):2507–2520, 2008.

Monica Babes, Enrique Munoz de Cote, and Michael L. Littman. Social reward shaping in the prisoner's dilemma. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1389–1392, 2008.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Optimal exploration-exploitation in a multi-armed-bandit problem with non-stationary rewards. *Available at SSRN 2436629*, 2014.

Michael Bowling. Convergence and no-regret in multiagent learning. In *Annual Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 209–216, 2005.

Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002. ISSN 0004-3702.

Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *National Conference on Artificial intelligence/Innovative Applications of Artificial Intelligence*, pages 746–752, 1998.

Vincent Conitzer and Tuomas Sandholm. AWESOME: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1-2):23–43, 2007. ISSN 0885-6125.

Bruno C. da Silva, Eduardo W. Basso, Ana L. C. Bazzan, and Paulo M. Engel. Dealing with non-stationary environments using context detection. In *International Conference on Machine Learning (ICML)*, pages 217–224. ACM, 2006.

William Dabney and Andrew G Barto. Adaptive step-size for online temporal difference learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

Jose Enrique Munoz de Cote, Alessandro Lazaric, and Marcello Restelli. Learning to cooperate in multi-agent social dilemmas. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 783–785, 2006.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

Rodrigues Gomes Eduardo and Ryszard Kowalczyk. Dynamic analysis of multiagent q-learning with ϵ-greedy exploration. In *International Conference on Machine Learning (ICML)*, pages 369–376. ACM, 2009.

Drew Fudenberg and David K. Levine. *The Theory of Learning in Games*. MIT press, 1998.

Aram Galstyan. Continuous strategy replicator dynamics for multi-agent q-learning. *Autonomous Agents and Multi-Agent Systems*, 26(1):37–53, 2013. ISSN 1387-2532.

Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems. In *International Conference on Algorithmic Learning Theory*, pages 174–188, 2011.

Abraham P George and Warren B Powell. Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. *Machine learning*, 65(1):167–198, 2006.

Hado V Hasselt. Double q-learning. In *Annual Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 2613–2621, 2010.

Junling Hu and Michael P. Wellman. Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003. ISSN 1533-7928.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 99:1563–1600, 2010.

Michael Kaisers and Karl Tuyls. Frequency adjusted multi-agent q-learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 309–316, 2010. ISBN 978-0-9826571-1-9.

Alessandro Lazaric, Jose Enrique Munoz de Cote, Fabio Dercole, and Marcello Restelli. Bifurcation analysis of reinforcement learning agents in the selten's horse game. In *Workshop on Adaptive Agents and Multi-Agents Systems*, pages 129–144, 2007.

David S. Leslie and Edmund J. Collins. Individual q-learning in normal form games. *SIAM J. Control and Optimization*, 44(2):495–514, 2005.

Ashique Rupam Mahmood, Richard S Sutton, Thomas Degris, and Patrick M Pilarski. Tuning-free step-size adaptation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2121–2124. IEEE, 2012.

Koichi Moriyama. Utility based q-learning to facilitate cooperation in prisoner's dilemma games. *Web Intelligence and Agent Systems*, 7:233–242, August 2009.

Koichi Moriyama, Satoshi Kurihara, and Masayuki Numao. Evolving subjective utilities: Prisoner's dilemma game examples. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 233–240, Richland, SC, 2011.

Gergely Neu, Andras Antos, András György, and Csaba Szepesvári. Online markov decision processes under bandit feedback. In *Annual Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 1804–1812, 2010.

Itsuki Noda. Recursive adaptation of stepsize parameter for non-stationary environments. In MatthewE. Taylor and Karl Tuyls, editors, *Adaptive and Learning Agents*, volume 5924 of *Lecture Notes in Computer Science*, pages 74–90. Springer Berlin Heidelberg, 2010.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

Tuomas W. Sandholm and Robert H. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems*, 37(1–2):147 – 166, 1996.

Alexander L Strehl, Lihong Li, and Michael L Littman. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 10:2413–2444, 2009.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *International Conference on Machine Learning (ICML)*, pages 216–224, 1990.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1999.

Richard S Sutton, Anna Koop, and David Silver. On the role of tracking in stationary environments. In *International Conference on Machine learning (ICML)*, pages 871–878. ACM, 2007.

Brian Tanner and Adam White. RL-Glue: language-independent software for reinforcement-learning experiments. *Journal of Machine Learning Research*, 10:2133–2136, September 2009.

Gerald Tesauro. Extending q-learning to general adaptive multi-agent systems. In *Annual Conference on Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2004.

Karl Tuyls, Pieter Jan 't Hoen, and Bram Vanschoenwinkel. An evolutionary dynamical analysis of multi-agent learning in iterated games. *Autonomous Agents and Multi-Agent Systems*, 12(1): 115–153, 2006.

Peter Vrancx, Karl Tuyls, and Ronald Westra. Switching dynamics of multi-agent learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 307–313, 2008. ISBN 978-0-9817381-0-9.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

Michael Wunder, Michael L. Littman, and Monica Babes. Classes of multiagent q-learning dynamics with ϵ-greedy exploration. In *International Conference on Machine Learning (ICML)*, pages 1167–1174, 2010.

Jia Yuan Yu, Shie Mannor, and Nahum Shimkin. Markov decision processes with arbitrary reward processes. *Mathematics of Operations Research*, 34(3):737–757, 2009.

Chongjie Zhang and Victor Lesser. Multi-agent learning with policy prediction. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 927–934, 2010.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*, pages 928–936, 2003.

Journal of Machine Learning Research 17 (2016) 1-43 Submitted 4/14; Revised 4/15; Published 4/16

Large Scale Online Kernel Learning

Jing Lu JING.LU.2014@PHDIS.SMU.EDU.SG
Steven C.H. Hoi * CHHOI@SMU.EDU.SG
School of Information Systems, Singapore Management University
80 Stamford Road, Singapore, 178902

Jialei Wang JIALEI@UCHICAGO.EDU
Department of Computer Science, University of Chicago
5050 S Lake Shore Drive Apt S2009 Chicago IL, USA, 60637

Peilin Zhao ZHAOP@I2R.A-STAR.EDU.SG
*Institute for Infocomm Research, A*STAR*
1 Fusionopolis Way, 21-01 Connexis, Singapore, 138632

Zhi-Yong Liu ZHIYONG.LIU@IA.AC.CN
State Key Lab of Management and Control for Complex System, Chinese Academy of Sciences
No. 95 Zhongguancun East Road, Haidian District, Beijing, China, 100190

Editor: John Shawe-Taylor

Abstract

In this paper, we present a new framework for large scale online kernel learning, making kernel methods efficient and scalable for large-scale online learning applications. Unlike the regular budget online kernel learning scheme that usually uses some budget maintenance strategies to bound the number of support vectors, our framework explores a completely different approach of kernel functional approximation techniques to make the subsequent online learning task efficient and scalable. Specifically, we present two different online kernel machine learning algorithms: (i) Fourier Online Gradient Descent (FOGD) algorithm that applies the random Fourier features for approximating kernel functions; and (ii) Nyström Online Gradient Descent (NOGD) algorithm that applies the Nyström method to approximate large kernel matrices. We explore these two approaches to tackle three online learning tasks: binary classification, multi-class classification, and regression. The encouraging results of our experiments on large-scale datasets validate the effectiveness and efficiency of the proposed algorithms, making them potentially more practical than the family of existing budget online kernel learning approaches.

Keywords: online learning, kernel approximation, large scale machine learning

1. Introduction

In machine learning, *online learning* represents a family of efficient and scalable learning algorithms for building a predictive model incrementally from a sequence of data examples (Rosenblatt, 1958). Unlike regular batch machine learning methods (Shawe-Taylor and Cristianini, 2004; Vapnik, 1995) which usually suffer from a high re-training cost whenever new training data arrive, online learning algorithms are often very efficient and highly scalable, making them more suitable for large-scale online applications where data usually arrive sequentially and evolve dynamically and rapidly. Online learning techniques can be applied to many real-world applications, such as online spam detection (Ma et al., 2009), online advertising, multimedia retrieval (Xia et al., 2013), and computational finance (Li et al., 2012). In this paper, we first present an online learning methodology to tackle *online binary classification* tasks and then extend the technique to solve the tasks of *online multi-class classification* and *online regression* in the following sections.

Recently, a wide variety of online learning algorithms have been proposed to tackle online classification tasks. One popular family of online learning algorithms, which are referred to as the "linear online learning" (Rosenblatt, 1958; Crammer et al., 2006; Dredze et al., 2008), learn a linear predictive model on the input feature space. The key limitation of these algorithms lies in that the linear model sometimes is restricted to make effective classification if training data are linearly separable in the input feature space, which is not a common scenario for many real-world classification tasks especially when dealing with noisy training data in relatively low dimensional space. This has motivated the studies of "kernel based online learning" or referred to as "online kernel learning" (Kivinen et al., 2001; Freund and Schapire, 1999), which aims to learn kernel-based predictive models for resolving the challenging tasks of classifying instances that are non-separable in the input space.

One key challenge of conventional online kernel learning methods is that an online learner usually has to maintain a set of support vectors (SV's) in memory for representing the kernel-based predictive model. During the online learning process, whenever a new incoming training instance is misclassified, it typically will be added to the SV set, making the size of support vector set unbounded and potentially causing memory overflow for a large-scale online learning task. To address this challenge, a promising research direction is to explore "budget online kernel learning" (Crammer et al., 2003), which attempts to bound the number of SV's with a fixed budget size using different budget maintenance strategies whenever the budget overflows. Despite being studied actively, the existing budget online kernel methods have some limitations. Some efficient algorithms are too simple to achieve satisfactory approximation accuracy; some other algorithms are, despite more effective, too computationally intensive to run for large datasets, making them harm the crucial merit of high efficiency of online learning techniques for large-scale applications. In addition, when dealing with extremely large-scale databases in distributed machine learning environments (Low et al., 2012; Dean and Ghemawat, 2008), the growing large size of SV's would be a significant overhead for communication between different nodes. It is thus very important to investigate effective budget online kernel learning techniques to reduce the size of SV's so as to minimize the overall communication cost.

Unlike the existing budget online kernel learning methods, in this paper, we present a novel framework of large scale online kernel learning by exploring a completely different strategy. In particular, the key idea of our framework is to explore functional approximation techniques to approximate a kernel by transforming data from the input space to a new feature space, and then apply existing linear online learning algorithms on the new feature space. This allows to inherit the power of kernel learning while being able to take advantages of existing efficient linear online learning algorithms for large-scale online learning tasks. Specifically, we propose two different new algorithms: (i) Fourier Online Gradient

*. Corresponding Author

Descent (FOGD) algorithm which adopts the random Fourier features for approximating shift-invariant kernels and learns the subsequent model by online gradient descent; and (ii) Nyström Online Gradient Descent (NOGD) algorithm which employs the Nyström method for large kernel matrix approximation followed by online gradient descent learning. We explore the applications of the proposed algorithms for three different online learning tasks: binary classification, multi-class classification, and regression. We give theoretical analysis of our proposed algorithms, and conduct an extensive set of empirical studies to examine their efficacy.

The rest of the paper is organized as follows. Section 2 reviews the background and related work. Section 3 proposes the FOGD and NOGD algorithms for binary classification task and Section 4 analyze their theoretical properties. Section 5 and Section 6 further extends the two techniques for tackling online multi-class classification and online regression tasks, respectively. Section 7 presents our experimental results for three different tasks and Section 8 concludes our work.

2. Related Work

Our work is related to two major categories of machine learning research work: *online learning* and *kernel methods*. Below we briefly review some representative related work in each category.

First of all, our work is closely related to online learning methods for classification (Rosenblatt, 1958; Freund and Schapire, 1999; Crammer et al., 2006; Zhao and Hoi, 2010; Zhao et al., 2011; Wang et al., 2012a; Hoi et al., 2013), particularly for budget online kernel learning where various algorithms have been proposed to address the critical drawback of unbounded SV size and computational cost in online kernel learning. Most existing budget online kernel learning algorithms attempt to achieve a bounded number of SV's through the following major ways:

- *SV Removal*. Some well-known examples include Randomized Budget Perceptron (RBP) (Cavallanti et al., 2007) that randomly removes one existing SV when the number of SV's overflows the budget, Forgetron (Dekel et al., 2005) that simply discards the oldest SV, Budget Online Gradient Descent (BOGD) that basically also discards some old SV, and Budget Perceptron (Crammer et al., 2003) and Budgeted Passive Aggressive (BPA-S) algorithm (Wang and Vucetic, 2010) which attempt to discard the most redundant SV.

- *SV Projection*. By projecting the discarded SV's onto the remaining ones, these algorithms attempt to bound the SV size while reducing the loss due to budget maintenance. Examples include Projectron (Orabona et al., 2008), Budgeted Passive Aggressive Projectron (BPA-P), and Budgeted Passive Aggressive Nearest Neighbor (BPA-NN) (Wang and Vucetic, 2010). Despite achieving better accuracy, they often suffer extremely high computational costs.

- *SV Merging*. These methods attempt to maintain the budget by merging two existing SV's into a new one, such as the Twin Support Vector Machine (TVM) algorithm (Wang and Vucetic, 2009). The similar idea of SV merging was also explored to bound the number of SV's in Budget Stochastic Gradient Descent (BSGD-M) algorithm in (Wang et al., 2012b).

In contrast to the above approaches, our work explores a completely different approach, i.e., kernel functional approximation techniques, for resolving budget online kernel learning tasks. As a summary, Table 1 compares the properties of different budget online kernel learning algorithms, including the proposed FOGD and NOGD algorithms, where B is the budget on the desired SV size, D is the number of Fourier components, and k is the matrix approximation rank of Nyström.

Algorithms	Budget Strategy	Update Time	Space
Budget Perceptron	Removal	$O(B^2)$	$O(B)$
RBP	Removal	$O(B)$	$O(B)$
Forgetron	Removal	$O(B)$	$O(B)$
BOGD	Removal	$O(B)$	$O(B)$
BPA-S	Removal	$O(B)$	$O(B)$
Projectron	Projection	$O(B^2)$	$O(B^2)$
BPA-P	Projection	$O(B^3)$	$O(B^2)$
BPA-NN	Projection	$O(B)$	$O(B)$
TVM	Merging	$O(B^2)$	$O(B^2)$
FOGD	Functional Approximation	$O(D)$	$O(D)$
NOGD	Functional Approximation	$O(kB)$	$O(kB)$

Table 1: Comparison on different budget online kernel learning algorithms.

Moreover, our work is also related to kernel methods for classification tasks (Shawe-Taylor and Cristianini, 2004; Hoi et al., 2006, 2007), especially for some studies on large-scale kernel methods (Williams and Seeger, 2000; Rahimi and Recht, 2007). Our approach shares the similar idea with the Low-rank Linearization SVM (LLSVM) (Zhang et al., 2012), where the non-linear SVM is transformed into a linear problem via kernel approximation methods. Unlike their approach, we employ the technique of random Fourier features (Rahimi and Recht, 2007), which have been successfully explored for speeding up batch kernelized SVMs (Rahimi and Recht, 2007; Yang et al., 2012) and kernel-based clustering (Chitta et al., 2011, 2012) tasks. Besides, another kernel approximation technique used in our approach is the well-known Nyström method (Williams and Seeger, 2000), which has been widely applied in machine learning tasks, including Gaussian Processes (Williams and Seeger, 2000), Kernelized SVMs (Zhang et al., 2012), Kernel PCA, Spectral Clustering (Zhang and Kwok, 2009), and manifold learning (Talwalkar et al., 2008). Although these techniques have been applied for batch machine learning tasks, to the best of our knowledge, they have been seldom explored for online kernel learning tasks as studied in this paper. Finally, we note that the short version of this work had been published in the Proceedings of the Twenty-Third international joint conference on Artificial Intelligence (IJCAI2013) (Wang et al., 2013). The journal manuscript has made significant extension by including substantial amount of new contents and more extensive empirical studies.

3. Large Scale Online Kernel Learning for Binary Classification

In this section, we introduce the problem formulation of online kernel binary classification and the detailed steps of our proposed algorithms.

3.1 Problem Formulation

We consider the problem of online learning for binary classification by following online convex optimization settings. Our goal is to learn a function $f : \mathbb{R}^d \to \mathbb{R}$ from a sequence of training examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}$, where instance $\mathbf{x}_t \in \mathbb{R}^d$ and class label $y_t \in \mathcal{Y} = \{+1, -1\}$. We refer to the output f of the learning algorithm as a *hypothesis* and denote the set of all possible hypotheses by $\mathcal{H} = \{f | f : \mathbb{R}^d \to \mathbb{R}\}$. We will use $\ell(f(\mathbf{x}); y) : \mathbb{R}^2 \to \mathbb{R}$ as the loss function that penalizes the deviation of estimating $f(\mathbf{x})$ from observed labels y. Further, we consider $\mathcal{H}$ a Reproducing Kernel Hilbert Space (**RKHS**) endowed with a kernel function $\kappa(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ (Vapnik, 1998) implementing the inner product $\langle \cdot, \cdot \rangle$ such that: 1) κ has the reproducing property $\langle f, \kappa(\mathbf{x}, \cdot) \rangle = f(\mathbf{x})$ for $\mathbf{x} \in \mathbb{R}^d$; 2) $\mathcal{H}$ is the closure of the span of all $\kappa(\mathbf{x}, \cdot)$ with $\mathbf{x} \in \mathbb{R}^d$, that is, $\kappa(\mathbf{x}, \cdot) \in \mathcal{H}$ $\forall \mathbf{x} \in \mathcal{X}$. The inner product $\langle \cdot, \cdot \rangle$ induces a norm on $f \in \mathcal{H}$ in the usual way: $\|f\|_{\mathcal{H}} := \langle f, f \rangle^{\frac{1}{2}}$. To make it clear, we denote by $\mathcal{H}_\kappa$ an RKHS with explicit dependence on kernel κ. Throughout the analysis, we assume $\kappa(\mathbf{x}_i, \mathbf{x}_j) \leq 1$, $\forall \mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$.

Training an SVM classifier $f(\mathbf{x})$ can be formulated as the following optimization problem

$$\min_{f \in \mathcal{H}_\kappa} P(f) = \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 + \frac{1}{T}\sum_{t=1}^{T} \ell(f(\mathbf{x}_t); y_t),$$

where $\lambda > 0$ is a regularization parameter used to control model complexity. While in an pure online setting, the regularized loss in the t-th iteration is

$$\mathcal{L}_t(f) = \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 + \ell(f(\mathbf{x}_t); y_t).$$

The goal of an online learning algorithm is to find a sequence of functions f_t, $t \in [T]$ that achieve the minimum *Regret* along the whole learning process. The regret is defined as,

$$Regret = \sum_{t=1}^{T} \mathcal{L}_t(f_t) - \sum_{t=1}^{T} \mathcal{L}_t(f^*),$$

where $f^* = \arg\min_f \sum_{t=1}^{T} \mathcal{L}_t(f)$ is the optimal classifier assuming that we had foresight in all the training instances. In a typical online budgeted kernel learning algorithm, the algorithm learns the kernel-based predictive model $f(\mathbf{x})$ for classifying a new instance $\mathbf{x} \in \mathbb{R}^d$ as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{B} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}),$$

where B is the number of Support Vectors (SV's), α_i denotes the coefficient of the i-th SV, and $\kappa(\cdot, \cdot)$ denotes the kernel function. The existing budget online kernel classification

approach aims to bound the number of SV's by a budget constant B using different budget maintenance strategies. Unlike the existing budget online kernel learning methods using the budget maintenance strategies, we propose to tackle the challenge by exploring a completely different strategy, i.e., the kernel functional approximation approach that construct a kernel-induced new representation $\mathbf{z}(\mathbf{x}) \in \mathbb{R}^D$ such that the inner product of instances in the new space is able to approximate the kernel function:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) \approx \mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_j).$$

By the above approximation, the predictive model can be rewritten:

$$f(\mathbf{x}) = \sum_{i=1}^{B} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) \approx \sum_{i=1}^{B} \alpha_i \mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}) = \mathbf{w}^\top \mathbf{z}(\mathbf{x}),$$

where $\mathbf{w} = \sum_{i=1}^{B} \alpha_i \mathbf{z}(\mathbf{x}_i)$ denotes the weight vector to be learned in the new feature space. As a consequence, solving the regular online kernel classification task can be turned into a problem of an linear online classification task on the new feature space derived from the kernel approximation. In the following, we will present two online kernel learning algorithms for classification by applying two different kernel approximation methods: (i) Fourier Online Gradient Descent (FOGD) and (ii) Nyström Online Gradient Descent (NOGD) methods.

3.2 Fourier Online Gradient Descent

Random Fourier features can be used in shift-invariant kernels (Rahimi and Recht, 2007). A shift-invariant kernel is the kernel that can be written as $\kappa(\mathbf{x}_1, \mathbf{x}_2) = k(\Delta\mathbf{x})$, where k is some function and $\Delta\mathbf{x} = \mathbf{x}_1 - \mathbf{x}_2$ is the shift between the two instances. Examples of shift-invariant kernels include some widely used kernels, such as Gaussian and Laplace kernels. By performing an inverse Fourier transform of the shift-invariant kernel function, one can obtain:

$$\kappa(\mathbf{x}_1, \mathbf{x}_2) = k(\mathbf{x}_1 - \mathbf{x}_2) = \int p(\mathbf{u}) e^{i\mathbf{u}^\top(\mathbf{x}_1 - \mathbf{x}_2)} d\mathbf{u}, \tag{1}$$

where $p(\mathbf{u})$ is a proper probability density function calculated from the Fourier transform of function $k(\Delta\mathbf{x})$,

$$p(\mathbf{u}) = (\frac{1}{2\pi})^d \int e^{-i\mathbf{u}^\top(\Delta\mathbf{x})} k(\Delta\mathbf{x}) d(\Delta\mathbf{x}). \tag{2}$$

More specifically, for a Gaussian kernel $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2}{2\sigma^2})$, we have the corresponding random Fourier component $\mathbf{u}$ with the distribution $p(\mathbf{u}) = \mathcal{N}(0, \sigma^{-2}I)$. And for a Laplacian kernel $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|_1}{\sigma})$, we have $p(\mathbf{u}) = \sigma\Pi_d \frac{1}{\pi(1+\sigma^2 u_d^2)}$. Given a kernel function that is continuous and positive-definite, according to the Bochner's theorem (Rudin, 1990), the kernel function can be expressed as an expectation of function with a random variable $\mathbf{u}$:

$$\begin{aligned} &\int p(\mathbf{u}) e^{i\mathbf{u}^\top(\mathbf{x}_1 - \mathbf{x}_2)} d\mathbf{u} = \mathrm{E}_{\mathbf{u}}[e^{i\mathbf{u}^\top\mathbf{x}_1} \cdot e^{-i\mathbf{u}^\top\mathbf{x}_2}] \\ &= \mathrm{E}_{\mathbf{u}}[\cos(\mathbf{u}^\top\mathbf{x}_1)\cos(\mathbf{u}^\top\mathbf{x}_2) + \sin(\mathbf{u}^\top\mathbf{x}_1)\sin(\mathbf{u}^\top\mathbf{x}_2)] \\ &= \mathrm{E}_{\mathbf{u}}[[\sin(\mathbf{u}^\top\mathbf{x}_1), \cos(\mathbf{u}^\top\mathbf{x}_1)] \cdot [\sin(\mathbf{u}^\top\mathbf{x}_2), \cos(\mathbf{u}^\top\mathbf{x}_2)]]. \end{aligned} \tag{3}$$

The equality (1) can be obtained by only keeping the real part of the complex function. From (3), we can see any shift-invariant kernel function can be expressed by the expectation of the inner product of the new representation of original data, where the new data representation is $\mathbf{z}(\mathbf{x}) = [\sin(\mathbf{u}^\top\mathbf{x}), \cos(\mathbf{u}^\top\mathbf{x})]^\top$. As a consequence, we can sample D number of random Fourier components $\mathbf{u}_1, ...\mathbf{u}_D$ independently for constructing the new representation as follows:

$$\mathbf{z}(\mathbf{x}) = (\sin(\mathbf{u}_1^\top\mathbf{x}), \cos(\mathbf{u}_1^\top\mathbf{x}), ..., \sin(\mathbf{u}_D^\top\mathbf{x}), \cos(\mathbf{u}_D^\top\mathbf{x}))^\top.$$

The online kernel learning task in the original input space can thus be approximated by solving a linear online learning task in the new feature space. For data arriving sequentially, we can construct the new representation of a data instance on-the-fly, and then perform online learning in the new feature space using the online gradient descent algorithm. We refer to the proposed algorithm as the Fourier Online Gradient Descent (FOGD), as summarized in Algorithm 1.

3.3 Nyström Online Gradient Descent

The above random Fourier feature based approach attempts to approximate the kernel function explicitly, which is in general data independent for the given dataset and thus may not fully exploit the potential of data distribution for kernel approximation. To address this, we propose to explore the Nyström method (Williams and Seeger, 2000) to approximate a large kernel matrix by a data-dependent approach.

Before presenting the method, we first introduce some notations. We denote a kernel matrix by $\mathbf{K} \in \mathbb{R}^{T\times T}$ with rank r, the Singular Value Decomposition (SVD) of $\mathbf{K}$ as $\mathbf{K} = \mathbf{V}\mathbf{D}\mathbf{V}^\top$, where the columns of $\mathbf{V}$ are orthogonal and $\mathbf{D} = \text{diag}(\sigma_1, \ldots, \sigma_r,)$ is diagonal. For $k < r$, $\mathbf{K}_k = \sum_{i=1}^k \sigma_i V_i V_i^\top = \mathbf{V}_k\mathbf{D}_k\mathbf{V}_k^\top$ is the best rank-k approximation of $\mathbf{K}$, where V_i is the i-th column of matrix $\mathbf{V}$.

Given a large kernel matrix $\mathbf{K} \in \mathbb{R}^{T\times T}$, the Nyström method randomly samples $B \ll T$ columns to form a matrix $\mathbf{C} \in \mathbb{R}^{T\times B}$, and then derive a much smaller kernel matrix $\mathbf{W} \in \mathbb{R}^{B\times B}$ based on the sampled B instances. We can in turn approximate the original large kernel matrix by

$$\hat{\mathbf{K}} = \mathbf{C}\mathbf{W}_k^+\mathbf{C}^\top \approx \mathbf{K}, \tag{4}$$

where $\mathbf{W}_k$ is the best rank-k approximation of $\mathbf{W}$, $\mathbf{W}^+$ denotes the pseudo inverse of matrix $\mathbf{W}$.

We now apply the above Nyström based kernel approximation to tackle large-scale online kernel classification task. Similar to the previous approach, the key idea is to construct the new representation for every newly arrived data instance based on the kernel approximation principle. In particular, we propose the following scheme: (i) at the very early stage of the online classification task, we simply run any existing online kernel learning methods (e.g., kernel-based online gradient descent in our approach) whenever the number of SV's is smaller than the predefined budget B; (ii) once the budget is reached, we then use the stored B SV's to approximate the kernel value of any new instances (which is equivalent to using the first B columns to approximate the whole kernel matrix). From the approximated kernel matrix in (4), we could see the kernel value between i-th instance $\mathbf{x}_i$ and j-th instance

Algorithm 1 FOGD — Fourier Online Gradient Descent for Binary Classification

Input: the number of Fourier components D, step size η, kernel function k;
Initialize $\mathbf{w}_1 = 0$.
Calculate $p(\mathbf{u})$ for kernel k as (2).
Generate random Fourier components: $\mathbf{u}_1, ..., \mathbf{u}_D$ sampled from distribution $p(\mathbf{u})$
for $t = 1, 2, \ldots, T$ **do**
  Receive $\mathbf{x}_t$;
  Construct new representation:
  $\mathbf{z}_t(\mathbf{x}_t) = (\sin(\mathbf{u}_1^\top\mathbf{x}_t), \cos(\mathbf{u}_1^\top\mathbf{x}_t), ..., \sin(\mathbf{u}_D^\top\mathbf{x}_t), \cos(\mathbf{u}_D^\top\mathbf{x}_t))^\top$
  Predict $\widehat{y}_t = \text{sgn}(\mathbf{w}_t^\top\mathbf{z}(\mathbf{x}_t))$;
  Receive y_t and suffer loss $\ell\big(\mathbf{w}_t^\top\mathbf{z}(\mathbf{x}_t); y_t\big)$;
  if $\ell\big(\mathbf{w}_t^\top\mathbf{z}(\mathbf{x}_t); y_t\big) > 0$ **then**
    $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\nabla\ell\big(\mathbf{w}_t^\top\mathbf{z}(\mathbf{x}_t); y_t\big)$.
  end if
end for

Algorithm 2 NOGD — Nyström Online Gradient Descent for Binary Classification

Input: the budget B, step size η, rank approximation k.
Initialize support vector set $\mathcal{S}_1 = \emptyset$, and model $f_1 = 0$.
while $|\mathcal{S}_t| < B$ **do**
  Receive new instance $\mathbf{x}_t$;
  Predict $\widehat{y}_t = \text{sgn}(f_t(\mathbf{x}_t))$;
  Update f_t by regular Online Gradient Descent (OGD);
  Update $\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{t\}$ whenever loss is nonzero;
  $t = t + 1$;
end while
Construct the kernel matrix $\hat{\mathbf{K}}_t$ from $\mathcal{S}_t$.
$[\mathbf{V}_k, \mathbf{D}_k] = \mathsf{eigs}(\hat{\mathbf{K}}_t, k)$, where $\mathbf{V}_k$ and $\mathbf{D}_k$ are Eigenvectors and Eigenvalues of $\hat{\mathbf{K}}_t$.
Initialize $\mathbf{w}_t^\top = [\alpha_1, ..., \alpha_B](\mathbf{D}_k^{-0.5}\mathbf{V}_k^\top)^{-1}$.
Initialize the instance index $T_0 = t$;
for $t = T_0, \ldots, T$ **do**
  Receive new instance $\mathbf{x}_t$;
  Construct the new representation of $\mathbf{x}_t$:
  $\mathbf{z}(\mathbf{x}_t) = \mathbf{D}_k^{-0.5}\mathbf{V}_k^\top(\kappa(\mathbf{x}_t, \hat{\mathbf{x}}_1), ..., \kappa(\mathbf{x}_t, \hat{\mathbf{x}}_B))^\top$.
  Predict $\widehat{y}_t = \text{sgn}(\mathbf{w}_t^\top\mathbf{z}(\mathbf{x}_t))$;
  Update $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\nabla\ell\big(\mathbf{w}_t^\top\mathbf{z}(\mathbf{x}_t); y_t\big)$.
end for

$\mathbf{x}_j$ is approximated by the following

$$\begin{aligned}\hat{\kappa}(\mathbf{x}_i,\mathbf{x}_j) &= (\mathbf{C}_i\mathbf{V}_k\mathbf{D}_k^{-\frac{1}{2}})(\mathbf{C}_j\mathbf{V}_k\mathbf{D}_k^{-\frac{1}{2}})^\top \\ &= ([\kappa(\hat{\mathbf{x}}_1,\mathbf{x}_i),...,\kappa(\hat{\mathbf{x}}_B,\mathbf{x}_i)]\mathbf{V}_k\mathbf{D}_k^{-\frac{1}{2}})(\kappa(\hat{\mathbf{x}}_1,\mathbf{x}_j),...,\kappa(\hat{\mathbf{x}}_B,\mathbf{x}_j)\mathbf{V}_k\mathbf{D}_k^{-\frac{1}{2}})^\top,\end{aligned}$$

where $\hat{\mathbf{x}}_a, a \in \{1,...,B\}$ is the a-th support vector.

For a new instance, we construct the new representation as follows:

$$\mathbf{z}(\mathbf{x}) = ([\kappa(\hat{\mathbf{x}}_1,\mathbf{x}),...,\kappa(\hat{\mathbf{x}}_B,\mathbf{x})]\mathbf{V}_k\mathbf{D}_k^{-\frac{1}{2}})^\top.$$

Similarly, we can then apply the existing online gradient descent algorithm to learn the linear predictive model on the new feature space induced from the kernel. We denote the proposed algorithm the Nyström Online Gradient Descent (NOGD), as summarized in Algorithm 2. Different from the FOGD algorithm, the algorithm follows the kernelized online gradient descent until the number of SV's reaches B. To initialize the linear classifier $\mathbf{w}$, we aim to achieve

$$\mathbf{w}^\top\mathbf{z}(\mathbf{x}) = [\alpha_1,...,\alpha_B][\kappa(\hat{\mathbf{x}}_1,\mathbf{x}),...,\kappa(\hat{\mathbf{x}}_B,\mathbf{x})]^\top,$$

thus

$$\mathbf{w}^\top\mathbf{D}_k^{-\frac{1}{2}}\mathbf{V}_k^\top[\kappa(\hat{\mathbf{x}}_1,\mathbf{x}),...,\kappa(\hat{\mathbf{x}}_B,\mathbf{x})]^\top = [\alpha_1,...,\alpha_B][\kappa(\hat{\mathbf{x}}_1,\mathbf{x}),...,\kappa(\hat{\mathbf{x}}_B,\mathbf{x})]^\top.$$

The solution is

$$\mathbf{w}^\top\mathbf{D}_k^{-\frac{1}{2}}\mathbf{V}_k^\top = [\alpha_1,...,\alpha_B];$$

$$\mathbf{w}^\top = [\alpha_1,...,\alpha_B](\mathbf{D}_k^{-\frac{1}{2}}\mathbf{V}_k^\top)^{-1}.$$

4. Theoretical Analysis

In this section, we analyze the theoretical properties of the two proposed algorithms. We may use $\ell_t(f)$ instead of $\ell(f(\mathbf{x}_t);y_t)$ for simplicity.

Theorem 1 *Assume we have a shift-invariant kernel $\kappa(\mathbf{x}_1,\mathbf{x}_2) = k(\mathbf{x}_1-\mathbf{x}_2)$, where k is some function and the original data is contained by a ball $\mathbb{R}^d$ of diameter R. Let $\ell(f(\mathbf{x});y):\mathbb{R}^2\to\mathbb{R}$ be a convex loss function that is Lipschitz continuous with Lipschitz constant L. Let $\mathbf{w}_t, t\in[T]$ be the sequence of classifiers generated by FOGD in Algorithm 1. Then, for any $f^*(\mathbf{x}) = \sum_{t=1}^T \alpha_t^*\kappa(\mathbf{x},\mathbf{x}_t)$, we have the following with probability at least $1-2^8(\frac{\sigma_p R}{\epsilon})^2\exp(\frac{-D\epsilon^2}{4(d+2)})$,*

$$\sum_{t=1}^T \ell_t(\mathbf{w}_t) - \sum_{t=1}^T \ell_t(f^*) \le \frac{(1+\epsilon)\|f^*\|_1^2}{2\eta} + \frac{\eta}{2}L^2T + \epsilon LT\|f^*\|_1,$$

where $\|f^*\|_1 = \sum_{t=1}^T|\alpha_t^*|$, $\sigma_p^2 = E_p[\mathbf{u}^\top\mathbf{u}]$ is the second moment of the Fourier transform of the kernel function k given that $p(\mathbf{u})$ is the probability density function calculated by the Fourier transform of function k.

Proof Given $f^*(\mathbf{x}) = \sum_{t=1}^T \alpha_t^*\kappa(\mathbf{x},\mathbf{x}_t)$, according to the representer theorem (Schölkopf et al., 2001), we have a corresponding linear model: $\mathbf{w}^* = \sum_{t=1}^T \alpha_t^*\mathbf{z}(\mathbf{x}_t)$, where

$$\mathbf{z}(\mathbf{x}) = (\sin(\mathbf{u}_1^\top\mathbf{x}),\cos(\mathbf{u}_1^\top\mathbf{x}),...,\sin(\mathbf{u}_D^\top\mathbf{x}),\cos(\mathbf{u}_D^\top\mathbf{x}))^\top.$$

The first step to prove our theorem is to bound the regret of our sequence of linear model $\mathbf{w}_t$ learned by our online learner with respect to the linear model $\mathbf{w}^*$ in the new feature space. First of all, we have the following:

$$\begin{aligned}&\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2 = \|\mathbf{w}_t - \eta\nabla\ell_t(\mathbf{w}_t) - \mathbf{w}^*\|^2 \\ =&\|\mathbf{w}_t-\mathbf{w}^*\|^2 + \eta^2\|\nabla\ell_t(\mathbf{w}_t)\|^2 - 2\eta\nabla\ell_t(\mathbf{w}_t)(\mathbf{w}_t-\mathbf{w}^*).\end{aligned}$$

Combining the above and the convexity of the loss function, i.e.,

$$\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*) \le \nabla\ell_t(\mathbf{w}_t)(\mathbf{w}_t-\mathbf{w}^*),$$

we then have the following

$$\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*) \le \frac{\|\mathbf{w}_t-\mathbf{w}^*\|^2 - \|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}\|\nabla\ell_t(\mathbf{w}_t)\|^2.$$

Summing the above over $t=1,...,T$ leads to:

$$\begin{aligned}\sum_{t=1}^T(\ell_t(\mathbf{w}_t)-\ell_t(\mathbf{w}^*)) &\le \frac{\|\mathbf{w}_1-\mathbf{w}^*\|^2 - \|\mathbf{w}_{T+1}-\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^T\|\nabla\ell_t(\mathbf{w}_t)\|^2 \\ &\le \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2}L^2T. \qquad (5)\end{aligned}$$

Next we further examine the relationship between $\sum_{t=1}^T\ell_t(\mathbf{w}^*)$ and $\sum_{t=1}^T\ell_t(f^*)$. According to the uniform convergence of Fourier features (Claim 1 in Rahimi and Recht (2007)), we have the high probability bound for the difference between the approximated kernel value and the true kernel value, i.e., with probability at least $1-2^8(\frac{\sigma_p R}{\epsilon})^2\exp(\frac{-D\epsilon^2}{4(d+2)})$, where $\sigma_p^2 = E_p[\mathbf{u}^\top\mathbf{u}]$ is the second moment of the Fourier transform of the kernel function k given that $p(\mathbf{u})$ is the probability density function calculated by the Fourier transform of function k. We have $\forall i,j$

$$|\mathbf{z}(\mathbf{x}_i)^\top\mathbf{z}(\mathbf{x}_j) - \kappa(\mathbf{x}_i,\mathbf{x}_j)| < \epsilon.$$

Since we assume $\kappa(\mathbf{x}_i,\mathbf{x}_j)\le 1$, we can assume $\mathbf{z}(\mathbf{x}_i)^\top\mathbf{z}(\mathbf{x}_j)\le 1+\epsilon$, which lead to:

$$\|\mathbf{w}^*\|^2 \le (1+\epsilon)\|f^*\|_1^2. \qquad (6)$$

When $|\mathbf{z}(\mathbf{x}_i)^\top\mathbf{z}(\mathbf{x}_j) - \kappa(\mathbf{x}_i,\mathbf{x}_j)| < \epsilon$, we have

$$\begin{aligned}|\sum_{t=1}^T\ell_t(\mathbf{w}^*) - \sum_{t=1}^T\ell_t(f^*)| &\le \sum_{t=1}^T|\ell_t(\mathbf{w}^*)-\ell_t(f^*)| \\ &\le \sum_{t=1}^T L\sum_{i=1}^T|\alpha_i^*||\mathbf{z}(\mathbf{x}_i)^\top\mathbf{z}(\mathbf{x}_t) - \kappa(\mathbf{x}_i,\mathbf{x}_t)| \\ &\le \sum_{t=1}^T L\epsilon\sum_{i=1}^T|\alpha_i^*| = \epsilon LT\|f^*\|_1. \qquad (7)\end{aligned}$$

Combining (5), (6) and (7) leads to complete the proof. ■

Remark 1. In general, the larger the dimensionality D, the higher the probability of the bound to be achieved. This means that by sampling more random Fourier components, one can approximate the kernel function more accurately and effectively. From the above theorem, it is not difficult to show that, by setting $\eta = \frac{1}{\sqrt{T}}$ and $\epsilon = \frac{1}{\sqrt{T}}$, we have

$$\sum_{t=1}^{T} \ell_t(\mathbf{w}_t) - \sum_{t=1}^{T} \ell_t(f^*) \quad \leq \quad (\frac{2\|f^*\|_1 + L^2}{2} + L\|f^*\|_1)\sqrt{T},$$

which leads to a sub-linear regret $O(\sqrt{T})$. However, setting $\epsilon = \frac{1}{\sqrt{T}}$ requires to sample $D = O(T)$ random components in order to achieve a high probability, which seems unsatisfactory since we will have to solve a very high-dimensional linear online learning problem. However, even in this case, for our FOGD algorithm, the learning time cost for each instance is $O(c_1 T)$, while the time cost for classifying an instance by regular online kernel classification is $O(c_2 T)$, here c_1 is the time for a scalar product by FOGD, while c_2 is the time for computing the kernel function. Since $c_2 \gg c_1$, our method is still much faster than the regular online kernel classification methods.

Remark 2. This theorem bounds the regret for any shift-invariant kernel. Specially, we can get the bound for a Gaussian kernel $k(\mathbf{x}_1 - \mathbf{x}_2) = \exp(-\gamma||\mathbf{x}_1 - \mathbf{x}_2||^2)$, by setting $\sigma_p^2 = 2d\gamma$ (Rahimi and Recht, 2007).

The theoretical analysis for the NOGD algorithm follows the similar procedure as used by the FOGD algorithm. We first introduce a lemma to facility our regret bound analysis.

Lemma 1 *$P(f) = \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 + \frac{1}{T}\sum_{t=1}^{T} \ell_t(f)$ is the objective function of an SVM problem, where $\ell_t(f)$ is the hinge loss function of the t-th iteration. Define f^* to be the optimal solution when using the exact kernel matrix $\mathbf{K}$ and f_N as the optimal when adopting the Nyström approximated kernel matrix $\hat{\mathbf{K}}$. We have*

$$P(f_N) - P(f^*) \leq \frac{1}{2T\lambda}||\mathbf{K} - \hat{\mathbf{K}}||_2,$$

where $\|\mathbf{K} - \hat{\mathbf{K}}\|_2$ is the spectral norm of the kernel approximation gap.

This lemma mainly follows the Lemma 1 in (Yang et al., 2012), we omit the proof for conciseness.

Theorem 2 *Assume we learn with kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) \leq 1$, $\forall i, j \in [T]$. Let $\ell(f(\mathbf{x}); y) : \mathbb{R}^2 \to \mathbb{R}$ be the hinge loss function. Let the sequence of T instances $\mathbf{x}_1, ..., \mathbf{x}_T$ form a kernel matrix $\mathbf{K} \in \mathbb{R}^{T \times T}$, and $\hat{\mathbf{K}}$ is the approximation of $\mathbf{K}$ using Nyström method. Let $f_t(\mathbf{x}) = \mathbf{w}_t^\top \mathbf{z}(\mathbf{x}), t \in [T]$ be the sequence of classifiers generated by NOGD in Algorithm 2. We assume the norm of the gradients in all iterations are always bounded by a constant L, which is easy to achieve by a few projection steps when necessary. In addition, define $f_N(\mathbf{x}) = \mathbf{w}_N^\top \mathbf{z}(\mathbf{x})$ be the optimal classifier when using Nyström kernel approximation and assuming we had foresight for all instances. By defining f^* as the optimal classifier in the original kernel space with the assumption of the foresight for all the instances, we have the following:*

$$\sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_t) - \sum_{t=1}^{T} \mathcal{L}_t(f^*) \leq \frac{\|\mathbf{w}_N\|^2}{2\eta} + \frac{\eta}{2}L^2 T + \frac{1}{2\lambda}||\mathbf{K} - \hat{\mathbf{K}}||_2.$$

Proof Following the similar analysis as in Theorem 1, the regularized loss function in the t-th iteration, $\mathcal{L}_t(\mathbf{w}) = \frac{\lambda}{2}||\mathbf{w}||_2^2 + \ell_t(\mathbf{w})$ satisfies the following bound,

$$\sum_{t=1}^{T} (\mathcal{L}_t(\mathbf{w}_t) - \mathcal{L}_t(\mathbf{w}_N)) \leq \frac{\|\mathbf{w}_N\|^2}{2\eta} + \frac{\eta}{2}L^2 T.$$

As proven in (Yang et al., 2012), the linear optimization problem in $\mathbf{z}(\mathbf{x})$ space, i.e., $P(\mathbf{w}) = \frac{\lambda}{2}||\mathbf{w}||_2^2 + \frac{1}{T}\sum_{t=1}^{T} \ell_t(\mathbf{w})$ is equivalent to the approximated SVM $P(f) = \frac{\lambda}{2}||f||_{\mathcal{H}}^2 + \frac{1}{T}\sum_{t=1}^{T} \ell_t(f)$ when $f \in \mathcal{H}_N$ is the functional space of Nyström method. From Lemma 1, we have,

$$\sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_N) = \sum_{t=1}^{T} \mathcal{L}_t(f_N) = TP(f_N) \leq TP(f^*) + \frac{1}{2\lambda}||\mathbf{K} - \hat{\mathbf{K}}||_2 = \sum_{t=1}^{T} \mathcal{L}(f^*) + \frac{1}{2\lambda}||\mathbf{K} - \hat{\mathbf{K}}||_2.$$

We complete the proof by combining the above two formulas. ■

Remark. As shown in Theorem 5 of (Yang et al., 2012), the kernel approximation gap $||\mathbf{K} - \hat{\mathbf{K}}||_2 \leq O(\frac{T}{B})$. Consequently when setting $B = \sqrt{T}$ and $\eta = \frac{1}{\sqrt{T}}$, we have

$$\sum_{t=1}^{T} \mathcal{L}_t(\mathbf{w}_t) - \sum_{t=1}^{T} \mathcal{L}(f^*) \leq O(\sqrt{T}).$$

This theorem bounds the regret of the NOGD algorithm when using all singular values of matrix $\mathbf{W}$ but only $O(\sqrt{T})$ support vectors. However, when the rank of the Nyström approximated matrix $\hat{\mathbf{K}}$ is only k, the bound should be slightly worse. As (Cortes et al., 2010)(Theorem 1) shows, the following holds with probability at least $1 - \epsilon$,

$$\|\hat{\mathbf{K}} - \mathbf{K}\|_2 \leq \|\mathbf{K} - \mathbf{K}_k\|_2 + \frac{T}{\sqrt{B}}\mathbf{K}_{\max}(2 + \log\frac{1}{\epsilon}),$$

where $\|\mathbf{K} - \mathbf{K}_k\|_2$ is the spectral norm of the best rank-k approximated gap and $\mathbf{K}_{\max}$ is the maximum diagonal entry of $\mathbf{K}$. This indicates that to get the $O(\sqrt{T})$ regret bound, we should set the budget size $B = T/c$, where c is some constant. This seems suboptimal, but we will demonstrate that only a small budget size is needed for satisfactory performance in our experimental results. In addition, the time cost of using k-rank approximation is only k/B times of that when using all singular values, which makes the algorithm extremely efficient.

5. Large Scale Online Kernel Learning for Multi-class Classification

In this section, we extend the proposed Fourier Online Gradient Descent and Nyström Online Gradient Descent methods, which are originally designed for binary classification, to online multi-class classification task. We also give theoretical analysis of the two approaches.

5.1 Problem Settings

Similar to online binary classification tasks, online multi-class classification is performed over a sequence of training examples $(\mathbf{x}_t, y_t)$, $t = 1, \ldots, T$, where $\mathbf{x}_t \in \mathbb{R}^d$ is the observed features of the t-th training instance. Unlike binary classification where class label $y_t \in \mathcal{Y} = \{+1, -1\}$, in a multi-class classification task, each label belongs to a finite set $\mathcal{Y}$ of size $m > 2$, i.e., $y_t \in \mathcal{Y} = \{1, \ldots, m\}$. The true class label y_t is only revealed after the prediction $\hat{y}_t \in \mathcal{Y}$ is made.

We follow the protocol of multi-prototype classification for deriving multi-class online learning algorithm (Crammer et al., 2006). Specifically, it learns a function $f^r : \mathbb{R}^d \to \mathbb{R}$ for each of the classes $r \in \mathcal{Y}$. During the t-th iteration, the algorithm predicts a sequence of scores for the classes:

$$\big(f_t^1(\mathbf{x}_t), \ldots, f_t^m(\mathbf{x}_t)\big).$$

The predicted class is set to be the class with the highest prediction score:

$$\hat{y}_t = \arg\max_{r \in \mathcal{Y}} f_t^r(\mathbf{x}_t). \tag{8}$$

We then define s_t as the irrelevant class with the highest prediction score:

$$s_t = \arg\max_{r \in \mathcal{Y}, r \neq y_t} f_t^r(\mathbf{x}_t).$$

The margin with respect to the hypothesis in the t-th iteration is defined to be the gap between the prediction score of class y_t and s_t:

$$\gamma_t = f_t^{y_t}(\mathbf{x}_t) - f_t^{s_t}(\mathbf{x}_t).$$

Obviously, in a correct prediction, the margin $\gamma_t > 0$. However, as stated in (Crammer et al., 2006), we are not satisfied by a positive margin value and thus we define a *hinge-loss* function:

$$\ell\big(\overline{f}_t, \mathbf{x}_t, y_t\big) = \max(0, 1 - \gamma_t),$$

where $\overline{f}_t$ denotes the set of all m functions for all classes.

5.2 Multi-class Fourier Online Gradient Descent

As introduced in the previous section for binary classification task, the online multi-class kernel classification task in the original input space can also be approximated by solving a linear online learning task in the new feature space. First, we use the same Fourier feature mapping approach as in the binary case to map each input instance $\mathbf{x}_t$ to $\mathbf{z}(\mathbf{x}_t)$. Then Online Gradient Descent method is applied to learn a linear classifier.

Following the multi-class problem setting, we learn a weight vector $\mathbf{w}^r \in \mathbb{R}^d$ for each of the classes $r \in \mathcal{Y}$. And use the linear classifier $f^r(\mathbf{x}_t) = \mathbf{w}^r \cdot \mathbf{z}(\mathbf{x}_t)$ to approximate to kernel prediction score. During the t-th iteration, the algorithm predicts a sequence of scores for the m classes:

$$\big(\mathbf{w}_t^1 \cdot \mathbf{z}(\mathbf{x}_t), \ldots, \mathbf{w}_t^m \cdot \mathbf{z}(\mathbf{x}_t)\big).$$

We define the *hinge-loss* function as following:

$$\ell\big(\overline{\mathbf{w}}_t, \mathbf{x}_t, y_t\big) = \max(0, 1 - \gamma_t) = \max(0, 1 - \mathbf{w}_t^{y_t} \cdot \mathbf{z}(\mathbf{x}_t) + \mathbf{w}_t^{s_t} \cdot \mathbf{z}(\mathbf{x}_t)), \tag{9}$$

where $\overline{\mathbf{w}}_t$ denotes the set of all m weight vectors.

Following the Online Gradient Descent approach, the update strategy of $\overline{\mathbf{w}_t}$ when $\ell > 0$ is:

$$\mathbf{w}_{t+1}^r = \mathbf{w}_t^r - \eta \nabla \ell\big(\overline{\mathbf{w}}_t, \mathbf{x}_t, y_t\big),$$

where η is a positive learning rate parameter and the gradient is taken with regards to $\mathbf{w}_t^r$. By rewriting the loss function explicitly, we can rewrite the above as follows:

$$\mathbf{w}_{t+1}^{y_t} = \mathbf{w}_t^{y_t} + \eta \mathbf{z}(\mathbf{x}_t); \tag{10}$$

$$\mathbf{w}_{t+1}^{s_t} = \mathbf{w}_t^{s_t} - \eta \mathbf{z}(\mathbf{x}_t). \tag{11}$$

Only two of the weight vectors are updated during each iteration. Thus, we can derive the FOGD algorithm for multi-class versions, as summarized in Algorithm 3.

5.3 Multi-class Nyström Online Gradient Descent

Similar to the binary task, in the first a few iterations of multi-class online Nyström algorithm (before the size of SV set reaches the predetermined budget size B), the algorithm performs a regular Online Gradient Descent update to the m kernel classifiers when $\ell > 0$:

$$f_{t+1}^r = f_t^r - \eta \nabla \ell\big(\overline{f}_t, \mathbf{x}_t, y_t\big),$$

where $\eta > 0$ is the gradient descent step size and the gradient is taken with regard to f_t^r. By rewriting the loss function explicitly, we have

$$f_{t+1}^{y_t} = f_t^{y_t} + \eta \kappa(\mathbf{x}_t, \cdot); \tag{12}$$

$$f_{t+1}^{s_t} = f_t^{s_t} - \eta \kappa(\mathbf{x}_t, \cdot). \tag{13}$$

Therefore, we need to store a SV set $\mathcal{S}$ and update it when necessary: $\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{t\}$. Note that all the m classifiers share the same SV set, which is the same setting as that of binary case. However, the storage strategy for α_i, i.e., the coefficient of the i-th SV, is different from that of binary case. In multi-class task, a vector $\alpha_i \in \mathbb{R}^m$ is used to represent the coefficients of the i-th SV. Each of its element α_i^r is the coefficient of the i-th SV for the kernel classifier f^r. Obviously, $\alpha_i^r = 0$ if $r \neq y_t$ and $r \neq s_t$.

After the size of SV set reaches the budget B, we do a Nyström feature mapping as the approach discussed in the binary section to map each input instance $\mathbf{x}_t$ to $\mathbf{z}(\mathbf{x}_t)$. The following linear update steps follow that in the multi-class Fourier Gradient Descent algorithm, as equation (10) and (11).

We derive the NOGD algorithm for multi-class version, as summarized in Algorithm 4.

Algorithm 3 MFOGD — Multi-class Fourier Online Gradient Descent

Input: the number of Fourier components D, step size η, kernel function k;
Initialize $\mathbf{w}_1^r = 0$, $r = 1, ..., m$.
Calculate $p(\mathbf{u})$ for kernel k as (2).
Generate random Fourier components: $\mathbf{u}_1, ..., \mathbf{u}_D$ sampled from distribution $p(\mathbf{u})$.
for $t = 1, 2, \ldots, T$ **do**
  Receive $\mathbf{x}_t$;
  Construct new representation:
  $\mathbf{z}(\mathbf{x}_t) = (\sin(\mathbf{u}_1^\top \mathbf{x}_t), \cos(\mathbf{u}_1^\top \mathbf{x}_t), ..., \sin(\mathbf{u}_D^\top \mathbf{x}_t), \cos(\mathbf{u}_D^\top \mathbf{x}_t))^\top$
  Predict as in (8);
  Receive y_t and suffer loss $\ell\big(\overline{\mathbf{w}}_t, \mathbf{x}_t, y_t\big)$ (9);
  if $\ell\big(\overline{\mathbf{w}}_t, \mathbf{x}_t, y_t\big) > 0$ **then**
    update $\overline{\mathbf{w}}_t$ as (10) and (11)
  end if
end for

Algorithm 4 MNOGD — Multi-class Nyström Online Gradient Descent

Input: the budget B, step size η, rank approximation k.
Initialize support vector set $\mathcal{S}_1 = \emptyset$, and model $f_1^r = 0$, $r = 1, ..., m$.
while $|\mathcal{S}_t| < B$ **do**
  Receive $\mathbf{x}_t$;
  Predict as in (8);
  Update $\overline{f}_t$ by regular Online Gradient Descent (OGD), as (12) and (13);
  Update $\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{t\}$ whenever loss is nonzero;
  $t = t + 1$;
end while
Construct the kernel matrix $\hat{\mathbf{K}}_t$ from $\mathcal{S}_t$.
$[\mathbf{V}_k, \mathbf{D}_k] = \mathsf{eigs}(\hat{\mathbf{K}}_t, k)$, where $\mathbf{V}_k$ and $\mathbf{D}_k$ are Eigenvectors and Eigenvalues of $\hat{\mathbf{K}}_t$.
Initialize $\mathbf{w}_t^{r\top} = [\alpha_1^r, ..., \alpha_B^r](\mathbf{D}_k^{-0.5}\mathbf{V}_k^\top)^{-1}$, $r = 1, ..., m$.
Initialize the instance index $T_0 = t$;
for $t = T_0, \ldots, T$ **do**
  Receive $\mathbf{x}_t$;
  Construct the new representation of $\mathbf{x}_t$:
  $\mathbf{z}(\mathbf{x}_t) = \mathbf{D}_k^{-0.5}\mathbf{V}_k^\top(\kappa(\mathbf{x}_t, \hat{\mathbf{x}}_1), ..., \kappa(\mathbf{x}_t, \hat{\mathbf{x}}_B))^\top$.
  Predict as in (8);
  Update $\overline{\mathbf{w}}_t$ as (10) and (11)
end for

5.4 Theoretical Analysis

In this section, we analyze the theoretical properties of the two proposed multi-class algorithms. We may use $\ell_t(\overline{f})$ instead of $\ell(\overline{f}, \mathbf{x}_t, y_t)$ for simplicity.

Theorem 3 *Assume we have a shift-invariant kernel $\kappa(\mathbf{x}_1, \mathbf{x}_2) = k(\mathbf{x}_1 - \mathbf{x}_2)$, where k is some function and the original data is contained by a ball $\mathbb{R}^d$ of diameter R. Let $\ell\big(\overline{f}, \mathbf{x}, y\big)$ be the hinge-loss we talked above, where $\overline{f}$ denotes the set of m classifiers for the m classes and its output is a sequence of prediction scores. Let $\overline{\mathbf{w}}_t, t \in [T]$ be the sequence of classifiers generated by MFOGD in Algorithm 3. Then, for any $f_*^r(\mathbf{x}) = \sum_{t=1}^T \alpha_t^{r*}\kappa(\mathbf{x}, \mathbf{x}_t)$, $r \in \{1, ..., m\}$, we have the following with probability at least $1 - 2^8(\frac{\sigma_p R}{\epsilon})^2 \exp(\frac{-D\epsilon^2}{4(d+2)})$*

$$\sum_{t=1}^T \ell_t(\overline{\mathbf{w}}_t) - \sum_{t=1}^T \ell_t(\overline{f}_*) \le \frac{m(1+\epsilon)\|f_*\|_1^2}{2\eta} + \eta T D + 2T\epsilon\|f_*\|_1,$$

where $\|f_*\|_1 = \sum_{t=1}^T \max_{r\in\mathcal{Y}}|\alpha_t^{r*}|$, and $\sigma_p^2 = E_p[\mathbf{u}^\top\mathbf{u}]$ is the second moment of the Fourier transform of the kernel function k given that $p(\mathbf{u})$ is the probability density function calculated by the Fourier transform of function k.

Proof Given $f_*^r(\mathbf{x}) = \sum_{t=1}^T \alpha_t^{r*}\kappa(\mathbf{x}, \mathbf{x}_t)$, $r = 1, ...m$, according to the Representer Theorem (Schölkopf et al., 2001), we have a corresponding linear model: $\mathbf{w}_*^r = \sum_{t=1}^T \alpha_t^{r*}\mathbf{z}(\mathbf{x}_t)$, where

$$\mathbf{z}(\mathbf{x}) = (\sin(\mathbf{u}_1^\top\mathbf{x}), \cos(\mathbf{u}_1^\top\mathbf{x}), ..., \sin(\mathbf{u}_D^\top\mathbf{x}), \cos(\mathbf{u}_D^\top\mathbf{x}))^\top.$$

The first step to prove our theorem is to bound the regret of the sequence of linear models $\mathbf{w}_t$ learned by online learner with respect to the linear model $\mathbf{w}^*$ in the new feature space. We first have

$$\begin{aligned}\|\mathbf{w}_{t+1}^r - \mathbf{w}_*^r\|^2 - \|\mathbf{w}_t^r - \mathbf{w}_*^r\|^2 &= \|\mathbf{w}_t^r - \mathbf{w}_*^r + \beta_t^r\mathbf{z}(\mathbf{x}_t)\|^2 - \|\mathbf{w}_t^r - \mathbf{w}_*^r\|^2 \\ &= (\beta_t^r)^2\|\mathbf{z}(\mathbf{x}_t)\|^2 + 2\beta_t^r(\mathbf{w}_t^r \cdot \mathbf{z}(\mathbf{x}_t) - \mathbf{w}_*^r \cdot \mathbf{z}(\mathbf{x}_t)),\end{aligned}$$

where β_t^r is the parameter used to update $\mathbf{w}_t^r$ as (10) and (11). Thus, it may be η, $-\eta$, or 0. Obviously, $\|\mathbf{z}(\mathbf{x}_t)\|^2 = D$. We first assume that $\ell_t(\overline{\mathbf{w}}_t) > 0$ and there are updates in the t-th iteration. Summing the above over $r = 1, ..., m$, we have

$$\begin{aligned}&\sum_{r=1}^m \left(\|\mathbf{w}_{t+1}^r - \mathbf{w}_*^r\|^2 - \|\mathbf{w}_t^r - \mathbf{w}_*^r\|^2\right) \\ &= 2\eta^2 D + 2\eta(\mathbf{w}_t^{y_t} \cdot \mathbf{z}(\mathbf{x}_t) - \mathbf{w}_t^{s_t} \cdot \mathbf{z}(\mathbf{x}_t)) - 2\eta(\mathbf{w}_*^{y_t} \cdot \mathbf{z}(\mathbf{x}_t) - \mathbf{w}_*^{s_t} \cdot \mathbf{z}(\mathbf{x}_t)),\end{aligned}$$

where y_t is the true label in the t-th iteration and s_t is the label of the largest scored irrelevant label classified by the classifier set $\overline{\mathbf{w}}_t$, not by the classifier set $\overline{\mathbf{w}}_*$. Thus we have:

$$\mathbf{w}_t^{y_t} \cdot \mathbf{z}(\mathbf{x}_t) - \mathbf{w}_t^{s_t} \cdot \mathbf{z}(\mathbf{x}_t) = \gamma_{\mathbf{w},t} \qquad \mathbf{w}_*^{y_t} \cdot \mathbf{z}(\mathbf{x}_t) - \mathbf{w}_*^{s_t} \cdot \mathbf{z}(\mathbf{x}_t) \ge \gamma_{\mathbf{w}*,t}.$$

As we have assumed $\ell_t(\overline{\mathbf{w}}_t) > 0$, we have $\ell_t(\overline{\mathbf{w}}_t) = 1 - \gamma_{\mathbf{w},t}$. And from the fact that $\ell_t(\overline{\mathbf{w}}_*) \ge 1 - \gamma_{\mathbf{w}*,t}$, we have:

$$\gamma_{\mathbf{w},t} - \gamma_{\mathbf{w}*,t} \le 1 - \ell_t(\overline{\mathbf{w}}_t) - (1 - \ell_t(\overline{\mathbf{w}}_*)) = \ell_t(\overline{\mathbf{w}}_*) - \ell_t(\overline{\mathbf{w}}_t).$$

Combining the three formula above, we get

$$\ell_t(\overline{\mathbf{w}}_t) - \ell_t(\overline{\mathbf{w}}_*) \leq \eta D + \frac{\sum_{r=1}^m \|\mathbf{w}_t^r - \mathbf{w}_*^r\|^2 - \sum_{r=1}^m \|\mathbf{w}_{t+1}^r - \mathbf{w}_*^r\|^2}{2\eta}.$$

Note that in some iterations, $\ell_t(\overline{\mathbf{w}}_t) = 0$ and there is no update in the t-th iteration, i.e., $\mathbf{w}_t^r = \mathbf{w}_{t+1}^r$. The above formula still holds. Summing it over $t = 1, ...T$ and assuming $\mathbf{w}_1^r = 0$ for all $r = 1, ..., m$ leads to:

$$\sum_{t=1}^T \ell_t(\overline{\mathbf{w}}_t) - \sum_{t=1}^T \ell_t(\overline{\mathbf{w}}_*) \leq \eta T D + \frac{\sum_{r=1}^m \|\mathbf{w}_*^r\|^2}{2\eta}. \tag{14}$$

Next we further examine the relationship between $\sum_{t=1}^T \ell_t(\overline{\mathbf{w}}_*)$ and $\sum_{t=1}^T \ell_t(\overline{f_*})$. According to the uniform convergence of Fourier features (Claim 1 in Rahimi and Recht (2007)), we have the high probability bound for the difference between the approximated kernel value and the true kernel value, i.e., with probability at least $1 - 2^8(\frac{\sigma_p R}{\epsilon})^2 \exp(\frac{-D\epsilon^2}{4(d+2)})$, where $\sigma_p^2 = E_p[\mathbf{u}^\top \mathbf{u}]$ is the second moment of the Fourier transform of the kernel function k given that $p(\mathbf{u})$ is the probability density function calculated by the Fourier transform of function k. We have $\forall i, j$

$$|\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_j) - \kappa(\mathbf{x}_i, \mathbf{x}_j)| < \epsilon.$$

Similar to the binary case:

$$\sum_{r=1}^m \|\mathbf{w}_*^r\|^2 \leq m\|f_*\|_1^2(1+\epsilon). \tag{15}$$

When $|\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_j) - \kappa(\mathbf{x}_i, \mathbf{x}_j)| < \epsilon$, we have

$$\begin{aligned}
&|\sum_{t=1}^T \ell_t(\overline{\mathbf{w}}_*) - \sum_{t=1}^T \ell_t(\overline{f_*})| \leq \sum_{t=1}^T |\ell_t(\overline{\mathbf{w}}_*) - \ell_t(\overline{f_*})| \leq \sum_{t=1}^T L|\gamma_{\mathbf{w}^*,t} - \gamma_{\mathbf{f}^*,t}| \\
&= \sum_{t=1}^T |\sum_{i=1}^T (\alpha_i^{y_t *}\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t) - \alpha_i^{s_t *}\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t)) - \sum_{i=1}^T (\alpha_i^{y_t *}\kappa(\mathbf{x}_i, \mathbf{x}_t) - \alpha_i^{s_t' *}\kappa(\mathbf{x}_i, \mathbf{x}_t))| \\
&\leq \sum_{t=1}^T \big(\sum_{i=1}^T |\alpha_i^{y_t *}\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t) - \alpha_i^{y_t *}\kappa(\mathbf{x}_i, \mathbf{x}_t)| + \sum_{i=1}^T |\alpha_i^{s_t *}\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t) - \alpha_i^{s_t' *}\kappa(\mathbf{x}_i, \mathbf{x}_t)| \big),
\end{aligned}$$

where L in the first line is the Lipschitz constant and can be set to 1 in hinge loss case, and s_t is the largest scored irrelevant label with regards to $\overline{\mathbf{w}}_*$ and s_t' with regard to $\overline{f_*}$. Thus the bound of the first term is easy to find and we will focus on the second term. Without loss of generality, we assume $\alpha_i^{s_t *}\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t) \geq \alpha_i^{s_t' *}\kappa(\mathbf{x}_i, \mathbf{x}_t)$. According to the definition of s_t and s_t', $\alpha_i^{s_t' *}\kappa(\mathbf{x}_i, \mathbf{x}_t) > \alpha_i^{s_t *}\kappa(\mathbf{x}_i, \mathbf{x}_t)$, leading to:

$$|\alpha_i^{s_t *}\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t) - \alpha_i^{s_t' *}\kappa(\mathbf{x}_i, \mathbf{x}_t)| \leq |\alpha_i^{s_t *}\big(\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_t) - \kappa(\mathbf{x}_i, \mathbf{x}_t)\big)| \leq |\alpha_i^{s_t *}|\epsilon.$$

Consequently, we have

$$|\sum_{t=1}^T \ell_t(\overline{\mathbf{w}}_*) - \sum_{t=1}^T \ell_t(\overline{f_*})| \leq 2T\epsilon\|f_*\|_1. \tag{16}$$

Combining (14), (15) and (16) leads to complete the proof. ■

Similar to the analysis of the binary classification case, it is not difficult to observe that the proposed MFOGD algorithm also achieves sub-linear regret $O(\sqrt{T})$. We will continue the analysis of MNOGD method in the following. As in the binary analysis, we first propose a lemma that bounds the gap of averaged loss between the exact kernel SVM and approximated kernel SVM. Unlike the binary classification problem, there are many different problem settings for the multi-class SVM problem, as surveyed by Hsu and Lin (2002). For consistency, in the following analysis, we adopt the common slack variables for all classes setting (Crammer and Singer, 2001, 2002).

Lemma 2 *$P(\overline{f}) = \frac{\lambda}{2}\sum_{r=1}^m \|f^r\|_{\mathcal{H}}^2 + \frac{1}{T}\sum_{t=1}^T \ell_t(\overline{f})$ is the objective function of an multi-class SVM problem, where $\ell_t(\overline{f})$ is the multi-class hinge loss function of the t-th iteration. Define $\overline{f^*}$ as the optimal solution of $P(\overline{f})$ when using the exact kernel matrix $\mathbf{K}$ and $\overline{f_N}$ as the optimal solution of SVM algorithm when adopting the Nyström approximated kernel matrix $\hat{\mathbf{K}}$. We have*

$$P(\overline{f_N}) - P(\overline{f^*}) \leq \frac{m}{2T\lambda}||\mathbf{K} - \hat{\mathbf{K}}||_2,$$

where $\|\mathbf{K} - \hat{\mathbf{K}}\|_2$ is the spectral norm of the kernel approximation gap.

Proof The dual problem of multi-class SVM is

$$\begin{aligned}
\max P(\boldsymbol{\alpha}) &= -\frac{1}{2\lambda}\sum_{r=1}^m \boldsymbol{\alpha}_r^\top \mathbf{K}\boldsymbol{\alpha}_r - \sum_{r=1}^m \boldsymbol{\alpha}_r^\top \mathbf{e}_r, \\
s.t. \quad &\sum_{r=1}^m \alpha_{t,r} = 0, \ \forall t \in \{1, 2, ..., T\}; \\
\alpha_{t,r} \leq 0, \ &\text{if } y_t \neq r, \quad \alpha_{t,r} \leq \frac{1}{T}, \text{if } y_t = r, \quad \forall t \in \{1, 2, ..., T\}, \quad \forall r \in \{1, 2, ...m\}
\end{aligned}$$

where $\boldsymbol{\alpha}_r = [\alpha_{1,r}, \alpha_{2,r}, ..., \alpha_{T,r}]^\top$, $\mathbf{e}_r = [e_{1,r}, e_{2,r}, ..., e_{T,r}]^\top$ and $e_{t,r} = 0$ if $y_t = r$, otherwise $e_{t,r} = 1$.

We can get the dual problem of the Nyström approximated multi-class SVM by replacing the kernel matrix $\mathbf{K}$ with $\hat{\mathbf{K}}$.

$$\begin{aligned}
P(\overline{f_N}) &= \max\left[-\frac{1}{2\lambda}\sum_{r=1}^m \boldsymbol{\alpha}_r^\top \hat{\mathbf{K}}\boldsymbol{\alpha}_r - \sum_{r=1}^m \boldsymbol{\alpha}_r^\top \mathbf{e}_r\right] \\
&= \max\left[-\frac{1}{2\lambda}\sum_{r=1}^m \boldsymbol{\alpha}_r^\top (\hat{\mathbf{K}} - \mathbf{K} + \mathbf{K})\boldsymbol{\alpha}_r - \sum_{r=1}^m \boldsymbol{\alpha}_r^\top \mathbf{e}_r\right] \\
&= \max\left[\frac{1}{2\lambda}\sum_{r=1}^m \boldsymbol{\alpha}_r^\top (\mathbf{K} - \hat{\mathbf{K}})\boldsymbol{\alpha}_r\right] + \max\left[-\frac{1}{2\lambda}\sum_{r=1}^m \boldsymbol{\alpha}_r^\top \mathbf{K}\boldsymbol{\alpha}_r - \sum_{r=1}^m \boldsymbol{\alpha}_r^\top \mathbf{e}_r\right].
\end{aligned}$$

Consequently,

$$P(\overline{f_N}) - P(\overline{f^*}) \leq \max \frac{1}{2\lambda} \sum_{r=1}^{m} \boldsymbol{\alpha}_{\boldsymbol{r}}^{\top} (\mathbf{K} - \hat{\mathbf{K}}) \boldsymbol{\alpha}_{\boldsymbol{r}} \leq \max \sum_{r=1}^{m} \frac{1}{2\lambda} ||\boldsymbol{\alpha}_{\boldsymbol{r}}||_2^2 ||\mathbf{K} - \hat{\mathbf{K}}||_2.$$

We complete the proof by considering $|\alpha_{t,r}| \leq \frac{1}{T}$. ■

Theorem 4 *Assume we learn with kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) \leq 1, \forall i, j \in [T]$. Let $\ell(f(\mathbf{x}); y) : \mathbb{R}^2 \to \mathbb{R}$ be the multi-class hinge loss function. Let the sequence of T instances $\mathbf{x}_1, ..., \mathbf{x}_T$ form a kernel matrix $\mathbf{K} \in \mathbb{R}^{T \times T}$, and $\hat{\mathbf{K}}$ is the approximation of $\mathbf{K}$ using Nyström method . Let $f_t^r(\mathbf{x}) = \mathbf{w}_{t,r}^{\top} \mathbf{z}(\mathbf{x}), t \in [T], r \in \{1, 2, ..., m\}$ be the sequence of classifiers generated by NOGD in Algorithm 4. We assume the norm of the gradients in all iterations are always bounded by a constant L, which is easy to achieve by a few projection steps when necessary. In addition, define $f_N^r(\mathbf{x}) = \mathbf{w}_{N,r}^{\top} \mathbf{z}(\mathbf{x}), r \in \{1, 2, ..., m\}$ be the optimal classifier when using Nyström kernel approximation and assuming we had foresight for all instances. By defining $\overline{f^*}$ as the optimal classifier in the original kernel space with the assumption of the foresight for all the instances, we have the following:*

$$\sum_{t=1}^{T} \mathcal{L}_t(\overline{\mathbf{w}_t}) - \sum_{t=1}^{T} \mathcal{L}_t(\overline{f^*}) \leq \sum_{r=1}^{m} \frac{\|\mathbf{w}_{N,r}\|^2}{2\eta} + \frac{\eta}{2} L^2 T + \frac{m}{2\lambda} ||\mathbf{K} - \hat{\mathbf{K}}||_2.$$

This theorem is a combination of standard online gradient descent analysis and Lemma 2. The proof is omitted since it is similar to the binary analysis and straightforward. It's easy to find that the multi-class NOGD algorithm enjoys the similar regret bound as the binary algorithm.

6. Large Scale Online Kernel Learning for Regression

In this section, we extend the proposed FOGD and NOGD algorithms to tackle online regression tasks. Consider a typical online regression task with a sequence of instances $(\mathbf{x}_t, y_t), t = 1, ..., T$, where $\mathbf{x}_t \in \mathbb{R}^d$ is the feature vector of the t-th instance and $y_t \in \mathbb{R}$ is the real target value, which is only revealed after the prediction is made at each iteration. The goal of online kernel regression task is to learn a model $f(\mathbf{x})$ that maps a new input instance $\mathbf{x} \in \mathbb{R}^d$ to a real value prediction:

$$f(\mathbf{x}) = \sum_{i=1}^{B} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}).$$

We apply the squared loss as the evaluation metric of regression accuracy:

$$\ell(f(\mathbf{x}_t); y_t) = (f(\mathbf{x}_t) - y_t)^2.$$

As the same approximation strategy with the previous task, with a feature mapping function $\mathbf{z}(\mathbf{x})$, the kernel regression task can be tackled by solving its approximated problem: to find a linear model $f(\mathbf{x}) = \mathbf{w}^{\top} \mathbf{z}(\mathbf{x})$ that minimizes the accumulated squared loss of all the

training instances. We use online gradient descent algorithm in this new feature space. In order to reduce the frequency of update, we define ϵ as the threshold. Update is only performed when the loss exceeds threshold ϵ. We denote the proposed algorithms the FOGD for Regression (FOGD-R) and NOGD (NOGD-R) for regression tasks, as summarized in Algorithm 5 and Algorithm 6.

We omit the theoretical analysis of regression since it is similar to the binary case.

7. Experimental Results

In this section, we conduct an extensive set of experiments to examine the efficacy of the proposed algorithms for several kinds of learning tasks in varied settings. Specifically, our first experiment is to evaluate the empirical performance of the proposed FOGD and NOGD algorithms for regular binary classification tasks by following a standard batch learning setting where each dataset is divided into two parts: training set and test set. This experiment aims to make a direct comparison of the proposed algorithms with some state-of-the-art approaches for solving batch classification tasks.

Our second major set of experiments is to evaluate the effectiveness and efficiency of the proposed FOGD and NOGD algorithms for online learning tasks by following a purely online learning setting, where the performance measures are based on average mistake rate and time cost accumulated in the online learning process on the entire dataset (there is no split of training and test sets). In particular, we conduct such experiments for three different online learning tasks: binary classification, multi-class classification, and regression, by comparing the proposed algorithms with a variety of state-of-the-art budget online kernel learning algorithms.

All the source code and datasets for our experiments in this work can be downloaded from our project web page:`http://LSOKL.stevenhoi.org/`. We are planning to release our algorithms in the future release of the LIBOL library (Hoi et al., 2014).

7.1 Experiment for Binary Classification Task in Batch Setting

In this section, we compare our proposed algorithms with many state-of-the-art batch classification algorithms. Different from online learning, the aim of a batch learning task is to train a classifier on the training dataset so that it achieves the best generalized accuracy on the test dataset.

7.1.1 Experimental Test bed and Setups

Table 2 summarizes the details of the datasets used in this experiment. All of them can be downloaded from LIBSVM website [1] or KDDCUP competition site [2]. We follow the original splits of training and test sets in LIBSVM. For KDD datasets, a random split of 4/1 is used.

We compare the proposed algorithms with the following widely used algorithms for training kernel SVM for batch classification tasks:

1. `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/`
2. `http://www.sigkdd.org/kddcup/`

Algorithm 5 FOGD-R — Fourier Online Gradient Descent for Regression

Input: the number of Fourier components D, step size η, threshold ϵ;
Initialize $\mathbf{w}_1 = 0$.
Calculate $p(\mathbf{u})$ as (2). Generate random Fourier components:$\mathbf{u}_1, ..., \mathbf{u}_D$ sampled from distribution $p(\mathbf{u})$.
for $t = 1, 2, \ldots, T$ **do**
  Receive $\mathbf{x}_t$;
  Construct new representation:
  $\mathbf{z}(\mathbf{x}_t) = (\sin(\mathbf{u}_1^\top \mathbf{x}_t), \cos(\mathbf{u}_1^\top \mathbf{x}_t), ..., \sin(\mathbf{u}_D^\top \mathbf{x}_t), \cos(\mathbf{u}_D^\top \mathbf{x}_t))^\top$
  Predict $\widehat{y}_t = \mathbf{w}_t^\top \mathbf{z}(\mathbf{x}_t)$;
  Receive y_t and suffer loss $\ell\big(\mathbf{w}_t^\top \mathbf{z}(\mathbf{x}_t); y_t\big)$;
  if $\ell\big(\mathbf{w}_t^\top \mathbf{z}(\mathbf{x}_t); y_t\big) > \epsilon$ **then**
    $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla \ell\big(\mathbf{w}_t^\top \mathbf{z}(\mathbf{x}_t); y_t\big)$.
  end if
end for

Algorithm 6 NOGD-R — Nyström Online Gradient Descent for Regression

Input: the budget B, step size η, rank approximation k, threshold ϵ.
Initialize support vector set $\mathcal{S}_1 = \emptyset$, and model $f_1 = 0$.
while $|\mathcal{S}_t| < B$ **do**
  Receive new instance $\mathbf{x}_t$;
  Predict $\widehat{y}_t = f_t(\mathbf{x}_t)$;
  Update f_t by regular Online Gradient Descent (OGD);
  Update $\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{t\}$ whenever loss exceeds threshold;
  $t = t + 1$;
end while
Construct the kernel matrix $\hat{\mathbf{K}}_t$ from $\mathcal{S}_t$.
$[\mathbf{V}_k, \mathbf{D}_k] = \mathsf{eigs}(\hat{\mathbf{K}}_t, k)$, where $\mathbf{V}_k$ and $\mathbf{D}_k$ are Eigenvectors and Eigenvalues of $\hat{\mathbf{K}}_t$, respectively.
Initialize $\mathbf{w}_t^\top = [\alpha_1, ..., \alpha_B](\mathbf{D}_k^{-0.5}\mathbf{V}_k^\top)^{-1}$.
Initialize the instance index $T_0 = t$;
for $t = T_0, \ldots, T$ **do**
  Receive new instance $\mathbf{x}_t$;
  Construct the new representation of $\mathbf{x}_t$:
  $\mathbf{z}(\mathbf{x}_t) = \mathbf{D}_k^{-0.5}\mathbf{V}_k^\top(\kappa(\mathbf{x}_t, \hat{\mathbf{x}}_1), ..., \kappa(\mathbf{x}_t, \hat{\mathbf{x}}_B))^\top$.
  Predict $\widehat{y}_t = \mathbf{w}_t^\top \mathbf{z}(\mathbf{x}_t)$;
  Update when loss exceeds ϵ: $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla \ell\big(\mathbf{w}_t^\top \mathbf{z}(\mathbf{x}_t); y_t\big)$.
end for

Dataset	# training instances	# testing instances	# features
codrna	59,535	271,617	8
w7a	24,692	25,057	300
w8a	49,749	14,951	300
a9a	32,561	16,281	123
KDDCUP08	81,835	20,459	117
KDDCUP99	905,257	226,314	127

Table 2: Details of Binary Classification Datasets.

- "LIBSVM": one of state-of-the-art implementation for batch kernel SVM available at the LIBSVM website Chang and Lin (2011);
- "LLSVM": Low-rank Linearization SVM algorithm that transfers kernel classification to a linear problem using low-rank decomposition of the kernel matrix (Zhang et al., 2012);
- "BSGD-M": The Budgeted Stochastic Gradient Descent algorithm which extends the Pegasos algorithm (Shalev-Shwartz et al., 2011) by exploring the SV Merging strategy for budget maintenance (Wang et al., 2012b);
- "BSGD-R": The Budgeted Stochastic Gradient Descent algorithm which extends the Pegasos algorithm (Shalev-Shwartz et al., 2011) by exploring the SV Random Removal strategy for budget maintenance (Wang et al., 2012b).

To make a fair comparison of algorithms with different parameters, all the parameters, including regularization parameter (C in LIBSVM, λ in pegasos), the learning rate (η in FOGD and NOGD) and the RBF kernel width (σ) are optimized by following a standard 5-fold cross validation on the training datasets. The budget size B in NOGD and pegasos algorithms and the feature dimension D in FOGD are set individually for different datasets, as indicated in the tables of experimental results. In general, these parameters are chosen such that they are roughly proportional to the size of support vectors output by the batch SVM algorithm in LIBSVM, since we would expect a relatively larger budget size for tackling more challenging classification tasks in order to achieve competitive accuracy. The rank k in NOGD is set to $0.2B$ for all datasets. For the online learning algorithms, all models are trained by a single pass through the training sets and the reported accuracy and time cost are averaged over the five experiments conducted on different random permutations of the training instances. All the algorithms were implemented in C++, and conducted on a Windows machine with CPU of 3.0GHz. For the existing algorithms, all the codes can be downloaded from LIBSVM website and BudgetedSVM website [3].

7.1.2 Performance Evaluation Results

Table 3 shows the experimental results of batch binary classification tasks. We can drawn several observations from the results.

3. http://www.dabi.temple.edu/budgetedsvm/algorithms.html

Algorithm	codrna, B=400, D=1600			w7a, B=400, D=1600		
	Train (s)	Test (s)	Accuracy	Train (s)	Test (s)	Accuracy
LIBSVM	80.20	126.02	96.67%	54.91	20.21	98.33%
LLSVM	19.04	37.34	95.93%	25.96	4.20	97.92%
BSGD-M	132.50	12.77	94.89%± 0.41	37.41	3.91	97.50%± 0.02
BSGD-R	3.32	12.77	67.16%± 0.68	1.79	1.71	97.50%± 0.01
FOGD	5.47	24.49	94.24%± 0.22	1.58	1.46	97.59%± 0.28
NOGD	2.55	9.90	**95.92%± 0.18**	1.57	1.44	**97.71%± 0.09**
Algorithm	w8a, B=1000, D=4000			a9a, B=1000, D=4000		
	Train (s)	Test (s)	Accuracy	Train (s)	Test (s)	Accuracy
LIBSVM	254.03	40.42	99.40%	97.81	24.80	85.04%
LLSVM	146.97	13.02	98.51%	70.85	14.43	84.89%
BSGD-M	230.51	2.42	97.52%± 0.02	150.16	3.32	84.21%± 0.18
BSGD-R	8.72	2.34	97.06%± 0.04	7.03	3.28	81.96%± 0.27
FOGD	13.04	3.85	97.93%± 0.08	9.35	4.57	84.93%± 0.03
NOGD	14.10	2.94	**98.36%± 0.10**	16.94	3.37	**84.99%± 0.07**
Algorithm	KDD08, B=200, D=800			KDD99, B=200, D=400		
	Train (s)	Test (s)	Accuracy	Train (s)	Test (s)	Accuracy
LIBSVM	1052.12	124.22	99.43%	20772.10	48.91	99.996%
LLSVM	24.31	3.53	99.38%	86.71	17.49	99.995%
BSGD-M	197.78	3.07	99.37%± 0.01	948.21	12.11	99.994%± 0.001
BSGD-R	12.34	3.03	99.12%± 0.23	52.82	12.20	99.980%± 0.031
FOGD	17.22	4.33	98.95%± 3.34	26.81	6.80	**99.996%± 0.001**
NOGD	7.60	2.14	**99.43%± 0.04**	20.71	9.03	99.993%± 0.001

Table 3: Performance Evaluation Results on Batch Binary Classification Tasks.

First of all, by comparing the four online algorithms against the two batch algorithms, we found that the online algorithms in general enjoy significant advantage in terms of computational efficiency especially for large scale datasets. By further examining the learning accuracy, we found that some online algorithms, especially the proposed FOGD and NGOD algorithms, are able to achieve slightly lower but fairly competitive learning accuracy compared with the state-of-the-art batch SVM algorithm. This demonstrates that the proposed online kernel learning algorithms could be potentially a good alternative solution of the existing SVM solvers when solving large scale batch kernel classification tasks in real-world applications due to their significant advantage of much lower learning time and memory costs.

Further, by comparing the four different online algorithms, we found that, in terms of learning accuracy, despite running faster, the BSGD-R ("pegasos+remove") algorithm suffers from very high mistake rate in most of the datasets. This is due to its naive budget maintenance strategy that simply discards the oldest support vector that may contain important information. While for BSGD-M ("pegasos+merging") algorithm, the main drawback is its relatively high computational cost. This can be easily observed in some datasets

(e.g., a9a, w7a and codrna), in which the difference between the number of support vectors of LIBSVM and the budget size is relatively larger than that of the other datasets. Thus, we can conclude that the high time cost of the BSGD-M("pegasos+merging") is due to the complex computation in the merging steps. Compared with the other online learning algorithms, the proposed NOGD algorithm achieves the highest accuracy for most cases while spending almost the lowest learning time cost. Similarly, FOGD algorithm also obtains more accurate result than the two budget Pegasos algorithms on most of the datasets with comparable or sometimes ever lower learning time cost. These facts indicate that the two proposed budget online kernel learning algorithms are both efficient and effective in solving large scale kernel classification problems.

Finally, by comparing the two proposed algorithms, we found that the performance of NOGD is better than that of FOGD. This reflects that the Nyström kernel approximation tends to have a better approximation of the original RBF kernel than the Fourier feature based approximation.

7.2 Experiments for Online Binary Classification Tasks

In this section, we test the performance of our proposed algorithms on the online binary classification task.

7.2.1 Experimental Test beds and Setup

Table 4 shows the details of 9 publicly available datasets of diverse sizes for online binary classification tasks. All of them can be downloaded from LIBSVM website, UCI machine learning repository [4] and KDDCUP competition site. As a yardstick for evaluation, we

Dataset	# instances	# features
german	1,000	24
spambase	4,601	57
w7a	24,692	300
w8a	64,700	300
a9a	48,842	123
KDDCUP08	102,294	117
ijcnn1	141,691	22
codrna	271,617	8
KDDCUP99	1,131,571	127

Table 4: Details of Online Binary Classification Datasets.

include the following two popular algorithms for regular online kernel classification without concerning budget:

- "Perceptron": the kernelized Perceptron (Freund and Schapire, 1999) without budget;
- "OGD": the kernelized online gradient descent (Kivinen et al., 2001) without budget.

4. http://www.ics.uci.edu/~mlearn/

Further, we compare the proposed budget online kernel learning algorithms with the following state-of-the-art budget online kernel learning algorithms:

- "RBP": the random budget perceptron by random removal strategy (Cavallanti et al., 2007);
- "Forgetron": the Forgetron by discarding oldest support vectors (Dekel et al., 2005);
- "Projectron": the Projectron algorithm using the projection strategy (Orabona et al., 2009);
- "Projectron++": the aggressive version of Projectron algorithm (Orabona et al., 2008, 2009);
- "BPA-S": the Budget Passive-Aggressive algorithm with simple SV removal strategy in (Wang and Vucetic, 2010);
- "BOGD": the Budget Online Gradient Descent algorithm by SV removal strategy (Zhao et al., 2012);

To make fair comparisons, all the algorithms follow the same setups. We adopt the hinge loss as the loss function ℓ. Note that hinge loss is not a smooth function, whose gradient is undefined at the point that the classification confidence $yf(\mathbf{x}) = 1$. Following the sub-gradient definition, in our experiment, gradient is only computed under the condition that $yf(\mathbf{x}) < 1$, and set to 0 otherwise. The Gaussian kernel bandwidth is set to 8. The step size η in the all online gradient descent based algorithms is chosen through a random search in range {2, 0.2, ..., 0.0002}. We adopt the same budget size $B = 100$ for NOGD and other budget algorithms. In the setting of FOGD algorithm, $D = \rho_f B$, where $0 < \rho_f < \infty$ is a predefined parameter that controls the number of random Fourier components. For NOGD algorithm, $k = \rho_n B$, where $0 < \rho_n < 1$ is a predefined parameter that controls the accuracy of matrix approximation. We set $\rho_f = 4$ and $\rho_n = 0.2$ and will evaluate their influence on the algorithm performance in the following discussion. For each data set, all the experiments were repeated 20 times using different random permutation of instances in the dataset. All the results were obtained by averaging over these 20 runs. For performance metrics, we evaluate the online classification performance by standard mistake rates and running time (seconds). All algorithms are implemented in Matlab R2013b, on a Windows machine with 3.0 GHZ CPU,6 cores.

7.2.2 Performance Evaluation Results

Table 5 summarizes the empirical evaluation results on the nine diverse data sets. From the results, we can draw the following observations.

First of all, in terms of time efficiency, we found that the budget online classification algorithms in general run much faster than the regular online kernel classification algorithms (Perceptron and OGD) especially on the large datasets, validating the importance of studying scalable online kernel methods. By further examining their results of mistake rates, we found that the budget online classification algorithms are generally worse than the two non-budget algorithms, validating the motivation of exploring effective techniques for budget online kernel classification.

Algorithm	german		spambase		w7a	
	Mistake Rate	Time(s)	Mistake Rate	Time(s)	Mistake Rate	Time(s)
Perceptron	35.2 %± 0.9	0.112	24.5 %± 0.1	1.606	4.01 %± 0.10	74.0
OGD	29.5 %± 0.5	0.130	22.0 %± 0.1	4.444	2.96 %± 0.10	119.9
RBP	37.5 %± 1.1	0.086	33.3 %± 0.4	0.613	5.07 %± 0.13	11.20
Forgetron	38.1 %± 0.9	0.105	34.6 %± 0.5	0.743	5.28 %± 0.06	11.77
Projectron	35.6 %± 1.5	0.101	30.8 %± 1.2	0.644	5.38 %± 1.15	11.22
Projectron++	35.1 %± 1.1	0.299	30.4 %± 1.0	1.865	4.79 %± 1.87	13.43
BPA-S	33.9 %± 0.9	0.092	30.8 %± 0.8	0.604	2.99 %± 0.06	11.60
BOGD	31.6 %± 1.5	0.114	32.2 %± 0.6	0.720	3.49 %± 0.16	11.56
FOGD	**29.9 %± 0.7**	**0.045**	**26.9 %± 1.0**	**0.263**	**2.75 %± 0.03**	**1.474**
NOGD	30.4 %± 0.8	0.109	29.1 %± 0.4	0.633	2.98 %± 0.01	11.58

Algorithm	w8a		a9a		ijcnn1	
	Mistake Rate	Time(s)	Mistake Rate	Time(s)	Mistake Rate	Time(s)
Perceptron	3.47 %± 0.01	642.8	20.9 %± 0.1	948.7	12.27 %± 0.01	812.6
OGD	2.81 %± 0.01	1008.5	16.3 %± 0.1	1549.5	9.52 %± 0.01	1269.0
RBP	5.10 %± 0.08	37.8	27.1 %± 0.2	15.4	16.40 %± 0.10	18.5
Forgetron	5.28 %± 0.07	40.0	27.8 %± 0.4	19.3	16.99 %± 0.32	21.2
Projectron	5.42 %± 1.10	38.1	21.6 %± 1.9	15.3	12.38 %± 0.09	19.2
Projectron++	5.41 %± 3.30	38.7	18.6 %± 0.5	23.4	9.52 %± 0.03	30.3
BPA-S	2.84 %± 0.03	39.2	21.1 %± 0.2	15.4	11.33 %± 0.04	18.3
BOGD	3.43 %± 0.08	38.9	27.9 %± 0.2	15.9	11.67 %± 0.13	19.2
FOGD	**2.43 %± 0.03**	**3.0**	**17.4 %± 0.1**	**1.8**	**9.06 %± 0.05**	**3.3**
NOGD	2.92 %± 0.03	38.9	**17.4 %± 0.2**	15.6	9.55 %± 0.01	19.1

Algorithm	codrna		KDDCUP08		KDDCUP99	
	Mistake Rate	Time(s)	Mistake Rate	Time(s)	Mistake Rate	Time(s)
Perceptron	14.0 %± 0.1	1015.7	0.90 %± 0.01	72.6	0.02 %± 0.00	1136
OGD	9.7 %± 0.1	1676.7	0.52 %± 0.01	421.7	0.01 %± 0.00	8281
RBP	20.3 %± 0.1	24.9	1.06 %± 0.03	34.3	0.02 %± 0.00	682
Forgetron	19.9 %± 0.1	28.9	1.07 %± 0.03	34.8	0.03 %± 0.00	684
Projectron	15.8 %± 0.5	26.0	0.94 %± 0.02	34.1	0.02 %± 0.00	642
Projectron++	13.6 %± 1.2	83.0	0.84 %± 0.03	77.2	**0.01 %± 0.00**	520
BPA-S	15.4 %± 0.3	26.5	0.62 %± 0.01	37.8	**0.01 %± 0.00**	796
BOGD	15.2 %± 0.1	32.5	0.61 %± 0.01	38.2	0.81 %± 0.06	805
FOGD	**10.3 %± 0.1**	**10.3**	0.71 %± 0.01	**4.1**	**0.01 %± 0.00**	**45**
NOGD	13.8 %± 2.1	27.2	**0.59 %± 0.01**	38.9	**0.01 %± 0.00**	511

Table 5: Evaluation of Large-scale Online Kernel Learning on Binary Classification Task .

Second, by comparing the proposed algorithms (FOGD and NOGD) with the budget online classification algorithms, we found that they generally achieve the best classification performance for most cases using fairly comparable or even lower time cost. While other algorithms, are either too slow because of their extremely complex updating methods or of low accuracy because of their simply SV removal steps. Similarly to the batch setting, this demonstrates the effectiveness and efficiency of the proposed algorithms.

Third, it might seem surprising to find that the FOGD algorithm achieves extremely low mistake rate and even outperforms the OGD algorithm in some datasets ($w7a, w8a, ijcnn1$). Ideally, FOGD should perform nearly the same as the kernel-based OGD approach if the number of Fourier components D is extremely large. However, choosing a too large value of D will result in underfitting for a relatively small data set, meanwhile choosing a too small value of D will result in overfitting. In our experiments, we choose an appropriate value of D ($D = 4B$) , which not only could save computational cost, but also may prevent both underfitting and overfitting. In contrast, the kernel OGD always add a new support vector whenever the loss is nonzero. Thus, the predicted model learned by the kernel OGD will become more and more complicated as time goes, and thus would likely suffer from overfitting for noisy examples.

Finally, we note that there are several differences in this result compared with the previous section in batch setting. To begin with, FOGD achieves extremely low time cost in all datasets. While in batch setting using C++ implementation, its time cost is comparable with that of NOGD. This can be explained by the different settings of the two implementation methods. In C++ setting, the most time consuming step in FOGD is to compute the large number of random features, while in Matlab setting, it is automatic transformed to a matrix calculation and parallelized on all cores of CPU. In addition, FOGD tends to performance better than NOGD in terms of accuracy. This is the result of different budget size. For NOGD, it is difficult to approximate the whole kernel matrix with small number of support vectors (such as the setting in this section $B = 100$). But with larger budget size, as in the batch case, the approximation accuracy is better than that of FOGD.

7.3 Experiments for Multi-class Classification Tasks

This section tests the performance of our proposed algorithms on online multi-class classification task.

7.3.1 Experimental Test beds and Setup

In this section, we evaluate the multi-class versions of FOGD and NOGD algorithms on 9 real-world datasets for multi-class classification tasks from the LIBSVM website. Table 6 summarizes the details of these datasets.

We adopt the same set of compared algorithms and similar parameter settings in multi-class task as that of binary case. Larger the budget size parameter B is used for multiclass classification than binary case since we should ensure enough support vectors for each class label. We set $B = 200$ for the first 3 datasets and $B = 100$ for the last 6 large scale datasets. For time efficiency, we omit the experiments of non-budget algorithms on extremely large datasets.

Dataset	# instances	# features	# classes
dna	2,000	180	3
satimage	4,435	36	6
usps	7,291	256	10
mnist	10,000	780	10
letter	15,000	16	26
shuttle	43,500	9	7
acoustic	78,823	50	3
covtype	581,012	54	7
poker	1,000,000	10	10

Table 6: Details of Multi-class Classification Datasets.

7.3.2 Performance Evaluation Results

Table 7 summarizes the average performance evaluation results for the compared algorithms on multi-class classification task. To further inspect more details of online multi-class classification performance, Figure 1 and Figure 2 also show the online performance convergence of all the compared algorithms in the entire online learning process. From these results, we can draw some observations as follows.

First of all, similar to the binary case, budget online kernel learning algorithms are much more efficient than the regular online kernel learning algorithms without budget, which is more obvious for larger scale datasets. For the three largest datasets (*acoustic*, *covtype* and *poker*), some of which consists of nearly one million instances, we have to exclude the non-budget online learning algorithms due to their extremely expensive costs in both time and memory. This again demonstrates the importance of exploring budget online kernel learning algorithms. Among the two non-budget online kernel learning algorithms, we found that OGD often achieves the highest accuracy, which is much better than Perceptron. However, its high-accuracy performance is paid by spending significantly higher computational time cost in comparison to the Perceptron algorithm. This is because OGD performs much more aggressive updates than Perceptron in the online learning process, which thus results in a significantly larger number of support vectors.

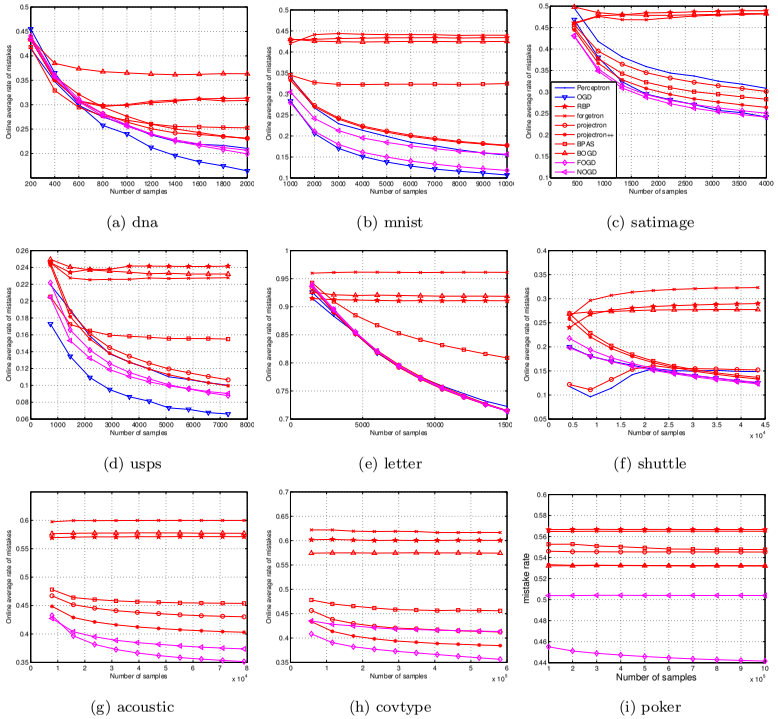
Second, when comparing the performance of different existing budget online kernel learning algorithms, it is clear to observe that the algorithms based on support vector projection strategy (projectron and projectron++) achieve significantly higher accuracy than the algorithms using simple support vector removal strategy. However, the gain of accuracy is paid by the sacrifice of efficiency, as shown by the time cost results in the table. Furthermore, one might be surprised to observe that BPA-S, which is relatively efficient in binary case, is extremely slow in multi-class case. This is due to the different updating approach of BPA-S for multi-class classification. In particular, for other budget multi-class algorithms, their time complexity of each prediction is $O(2B)$, i.e., only 2 out of the m classes (y and s) are updated when adding a new support vector. By contrast, during the update of BPA-S at each iteration, every class has to be updated, leading to the overall time complexity of $O(mB)$. Consequently, the BPA-S is much more expensive than the other algorithms.

Algorithm	dna		mnist		satimage	
	Mistake Rate	Time(s)	Mistake Rate	Time(s)	Mistake Rate	Time(s)
Perceptron	20.4 %±0.7	4.891	15.4 %±0.1	456.76	29.6 %±0.5	4.675
OGD	16.1 %±0.4	25.068	10.7 %±0.2	1004.65	23.6 %±0.3	6.917
RBP	31.1 %±1.4	2.707	43.4 %±0.5	59.98	49.3 %±0.8	2.409
Forgetron	30.9 %±2.1	2.949	43.9 %±0.6	64.32	48.2 %±1.4	2.503
Projectron	22.7 %±4.4	4.254	17.7 %±0.1	434.53	29.6 %±0.5	3.685
Projectron++	23.1 %±6.1	4.330	17.7 %±0.3	430.38	25.9 %±0.4	3.771
BPA-S	25.3 %±1.2	11.055	32.4 %±1.2	91.25	27.9 %±0.3	17.337
BOGD	36.3 %±0.9	3.103	42.5 %±0.4	62.51	48.2 %±0.6	2.670
FOGD	20.8 %±0.7	**0.887**	**11.8 %±0.2**	**1.792**	29.5 %±0.4	**0.915**
NOGD	**20.7 %±0.9**	2.292	15.6 %±0.6	46.90	**23.7 %±0.3**	1.869

Algorithm	usps		letter		shuttle	
	Mistake Rate	Time(s)	Mistake Rate	Time(s)	Mistake Rate	Time(s)
Perceptron	10.0 %±0.2	89.790	71.5 %±0.5	80.901	15.2 %±0.3	137.886
OGD	6.7 %±0.2	310.072	71.2 %±0.3	94.222	12.3 %±0.1	377.328
RBP	24.0 %±0.4	30.180	91.7 %±0.2	18.783	29.3 %±0.5	12.989
Forgetron	22.7 %±0.5	30.478	96.1 %±0.1	19.034	34.1 %±0.4	12.776
Projectron	10.6 %±0.1	192.347	71.5 %±0.5	26.921	15.3 %±0.3	20.034
Projectron++	9.9 %±0.2	194.366	**71.4 %±0.3**	27.584	16.8 %±0.5	20.879
BPA-S	15.4 %±0.6	54.457	84.6 %±0.5	47.459	14.1 %±0.2	58.856
BOGD	23.2 %±0.4	30.966	92.7 %±0.2	18.510	27.9 %±0.2	14.399
FOGD	10.1 %±0.2	**9.589**	71.5 %±0.6	**2.322**	15.6 %±0.5	**4.039**
NOGD	**9.0 %±0.2**	24.332	71.5 %±0.2	3.380	**12.3 %±0.1**	8.484

Algorithm	acoustic		covtype		poker	
	Mistake Rate	Time(s)	Mistake Rate	Time(s)	Mistake Rate	Time(s)
RBP	57.4 %±0.2	40.469	60.1 %±0.1	445.238	56.6 %±0.0	398.423
Forgetron	61.2 %±0.4	46.865	60.4 %±0.4	491.403	56.5 %±0.0	413.807
Projectron	43.0 %±0.1	46.469	41.1 %±0.2	930.646	54.5 %±0.1	810.421
Projectron++	40.3 %±0.1	47.400	38.3 %±0.2	937.860	53.2 %±0.1	825.526
BPA-S	46.2 %±0.3	210.916	45.2 %±0.2	1569.255	54.7 %±0.4	2566.812
BOGD	58.0 %±0.1	40.266	57.4 %±0.0	456.692	53.2 %±0.0	465.020
FOGD	43.0 %±0.2	**12.637**	**40.4 %±0.1**	**80.774**	52.6 %±0.1	**190.102**
NOGD	**37.8 %±0.1**	25.883	41.0 %±0.6	211.354	**50.3 %±0.2**	216.717

Table 7: Evaluation of Large-scale Online Kernel Learning on Multi-class Classification Task .



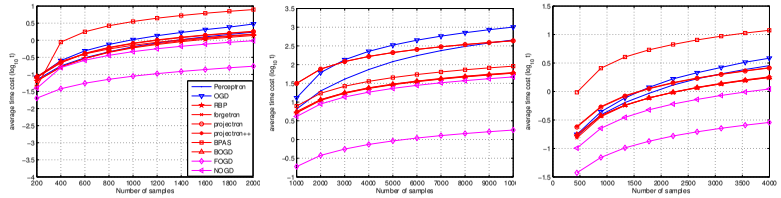
(a) dna (b) mnist (c) satimage

(d) usps (e) letter (f) shuttle

(g) acoustic (h) covtype (i) poker

Figure 1: Convergence evaluation of multi-class datasets: mistake rate (best viewed in color).

Furthermore, by comparing the two proposed algorithms, FOGD and NOGD, with the existing budget online kernel learning algorithms, we observe that the proposed algorithms achieve the highest accuracy for most cases, and meanwhile run significantly faster than the other algorithms, which again validates the effectiveness and efficiency of our proposed technique. Thus, we can conclude that the proposed functional approximation approach for budget online kernel learning is a promising technique for building scalable online kernel learning algorithms for large scale learning tasks. Finally, by comparing FOGD and NOGD, we found that their accuracy performance is nearly comparable while FOGD is relatively faster. As mentioned in the binary section, this indicates that FOGD is easier for parallelization.



(a) dna (b) mnist (c) satimage

(d) usps (e) letter (f) shuttle

(g) acoustic (h) covtype (i) poker

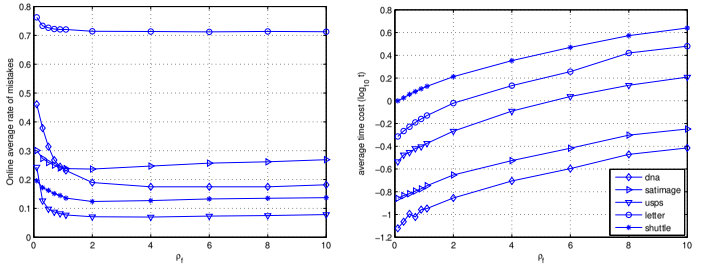
Figure 2: Convergence evaluation of multi-class datasets: time cost (best viewed in color).

7.3.3 Evaluation for the ρ_f and ρ_n on Multi-class Tasks

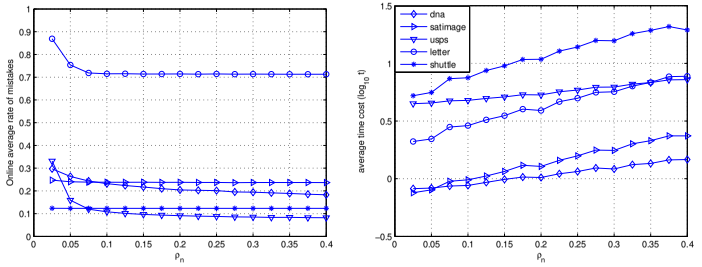
As mentioned in the previous experiments, we set the parameter for the number of Fourier components $D = \rho_f \times B$ and the rank of Nyström matrix approximation $k = \rho_n B$, where different choices of parameters ρ_f and ρ_n could affect the resulting performance of FOGD and NOGD, respectively. In this section, we evaluate the sensitivity of these two parameters and examine their influence to both learning accuracy and time cost of multi-class classification tasks.

Specifically, we fix the budget size B to 200 for all datasets, and set the other parameters (except B, ρ_f, and ρ_n) by following the same settings as the previous multi-class classification tasks. Figure 3 summarizes the performance evaluation results, including average mistake rates and average time costs. From the results, we can draw some observations as follows.

First of all, we observe that increasing the value of ρ_f or ρ_n leads to better classification accuracy but higher running time cost. This is not difficult to understand since increasing



(a) The effect of ρ_f on the mistake rate of FOGD (b) The effect of ρ_f on the time cost of FOGD

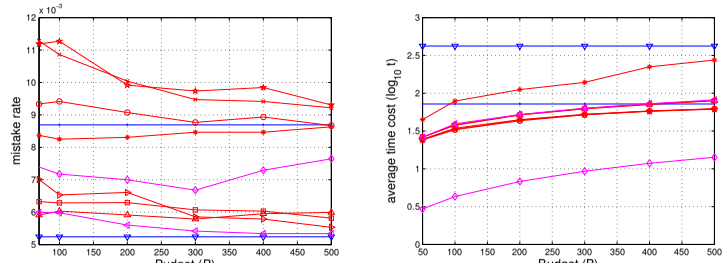


(c) The effect of ρ_n on the mistake rate of NOGD (d) The effect of ρ_n on the time cost of NOGD

Figure 3: Performance evaluation on different values of ρ_f and ρ_n.

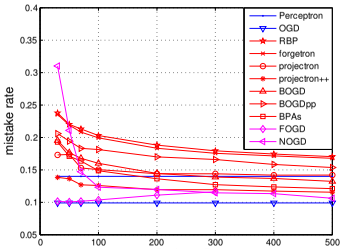
the value ρ_f is essential to increasing the number of Fourier components, leading to a better approximation of lower variance and thus higher classification accuracy. Meanwhile the computational time cost of FOGD is proportional to the number of Fourier components, and thus is proportional to the value of ρ_f. Similarly, for NOGD, the large the value of ρ_n, the more accurate approximation achieved by the Nystrom kernel matrix approximation, and meanwhile the more computational cost required. Thus, the choice of parameter ρ_f or ρ_n for FOGD or NOGD is essentially a trade off between learning effectiveness and computational efficiency.

Second, we found there is some common tendency of the impact on the learning accuracy by the two parameters, although different datasets may have slightly different results. In particular, we observe that when the value of ρ_f or ρ_n is large enough (e.g., $\rho_f > 5$ or $\rho_n > 0.1$), increasing their value has limited impact on the improvement of the learning accuracy while the time cost keeps growing linearly. This gives an important guideline for one to choose the two parameters properly in order to gain computational efficiency without sacrificing learning accuracy. Specifically, as shown in the figure, by choosing the two parameters roughly in the ranges of $\rho_f \in (4, 6)$ and $\rho_n \in (0.2, 0.4)$, we are able to achieve satisfactory tradeoff for most cases.

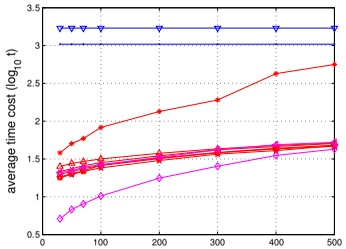


(a) KDDCUP08 (binary) mistake rate

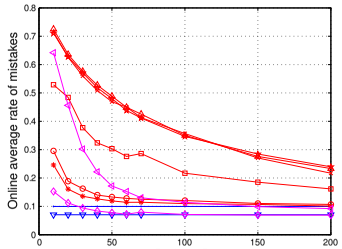
(b) KDDCUP08 (binary) time cost



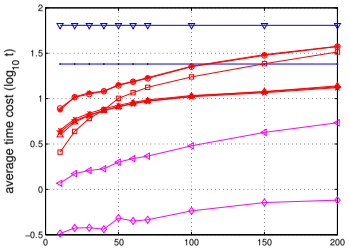
(c) codrna (binary) mistake rate



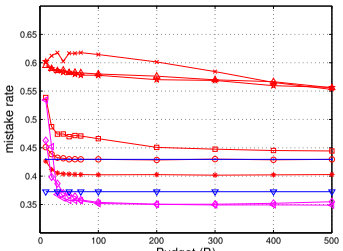
(d) codrna (binary) time cost



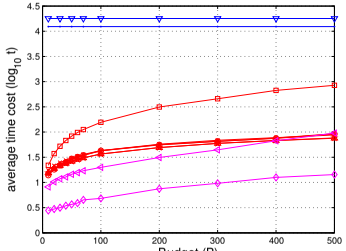
(e) usps (multi-class) mistake rate



(f) usps (multi-class) time cost



(g) acoustic (multi-class) mistake rate



(h) acoustic (multi-class) time cost

Figure 4: The effect of different budget sizes (B) (best viewed in color).

7.3.4 Evaluation for the Selection of Budget Size on Multi-class Classification Task

For all the budget online kernel learning algorithms, the choice of budget size parameter B can have a considerable impact on the resulting performance. In our previous experiments, we simply fix the budget size parameter B to some constants for the compared budget online kernel learning algorithms to enable a fair and simplified comparison. In this section, we aim to evaluate the sensitivity of the budget size parameter B and examine if the proposed algorithms are consistently better than the other budget online kernel learning algorithms under varied settings of the budget size parameter. Specifically, in this experiment, we follow the same experimental setups as the previous experiments, except that we evaluate the influence of varied values of budget size parameter B.

Figure 4 shows the experimental results of both average mistake rates and average time costs of different algorithms during the online learning processes under different values of the budget size parameter B on four randomly chosen datasets, including two binary classification datasets and two multi-class classification datasets. From the experimental results, we can draw several observations on the impact of the budget size parameter to the performances as follows.

First of all, we observe that increasing the budget size generally results in better classification accuracy and higher learning time cost for all the budget online kernel learning algorithms. This is not difficult to understand since a larger budget size potentially leads to a more precise approximation to their non-budget original algorithm, and thus a better prediction accuracy. Second, similar to the previous experiments, we notice that when the budget size is large enough, further increasing the budget size has limited gain on the improvement of classification accuracy. This observation indicates that selecting a proper budget size parameter B is a tradeoff between classification accuracy and learning time cost. Moreover, by comparing different budget learning algorithms under varied values of B, we found that the projectron algorithms and the proposed two algorithms (FOGD and NOGD) tend to achieve the best classification accuracy results for most cases, particularly when the value of budget size B is small. By further examining the time costs, we found that the proposed algorithms (especially FOGD) are significantly more efficient than the Projectron for varied values of B. These encouraging results again validate that the proposed algorithms not only achieve consistently better accuracy results than the existing budget online kernel learning methods for most cases, but also have a significant advantage in computational efficiency for large-scale online kernel learning tasks.

7.4 Experiments for Online Regression Tasks

This section tests the performance of our proposed algorithms on online regression tasks.

7.4.1 Experimental Test beds and Setup

Table 8 summarizes the details of the 9 datasets of diverse sizes in our online regression experiments. All of them are publicly available at the LIBSVM and UCI websites.

For comparison schemes, we compare the proposed FOGD-R and NOGD-R algorithms with three non-budget online regression algorithms including OGD, Perceptron, and Norma

Dataset	# instances	# features
housing	506	13
mg	1,385	6
abalone	4,177	8
parkinsons	5,875	20
cpusmall	8,192	12
cadata	20,640	8
casp	45,730	9
slice	53,500	385
year	463,715	90

Table 8: Details of Regression Datasets.

(Kivinen et al., 2001), and four other existing budget online kernel learning algorithms including RBP, Forgetron, Projectron, and BOGD.

For parameter setting, we follow the same setup as the previous experiments for most of the parameters. For the Norma algorithm, the adaptable threshold parameter ϵ is learned and updated at each iteration. For all the other algorithms, this parameter ϵ is simply fixed to 0.1. We set $\rho_f = 15$ and $B = 30$ for all the regression datasets. According to our empirical experience on online regression tasks, the regular perceptron based algorithms that simply use the default step size 1 would perform extremely poor because of the inappropriate learning rate. In order to have a stronger baseline for comparison, we conduct a validation experiment by searching for the best learning rate parameter (about 0.1) for all the perceptron-based algorithms.

7.4.2 Performance Evaluation Results

Table 9 shows the summary of empirical evaluation results on the nine datasets, and Figures 6 and 6 show the detailed regressions results in terms of both regression errors and time cost in the online learning processes. From these results, we can draw several observations as follows.

First of all, by examining the running time costs of different algorithms, it is clear to see that the budget online kernel learning algorithms are more efficient than the non-budget algorithms, particularly on larger scale datasets. This observation is consistent to the previous classification experiments, again validating the importance of studying budget online kernel learning methods. By examining the non-budget algorithms, we found that NORMA runs faster than the other two algorithms (OGD and Perceptron) which is primarily because of it the adaptive threshold which reduces the frequency of update and thus obtains a relatively smaller support vector set size. Among all the budget algorithms, the proposed FOGD algorithm is able to achieve the smallest time cost for all cases.

Second, in terms of regression accuracy, among the existing budget algorithms, the projectron algorithm outperforms the other existing budget online learning algorithms due to its sophisticated projection strategy. By further comparing the proposed FOGD and NOGD algorithms with the existing ones, we found that our algorithms achieve the lowest squared

Algorithm	housing		mg		abalone	
	Squared Loss	Time	Squared Loss	Time	Squared Loss	Time
OGD	0.04017±0.00043	0.028	0.05341±0.00071	0.103	0.01137±0.00007	0.388
Perceptron	0.04018±0.00080	0.029	0.05682±0.00084	0.103	0.01280±0.00010	0.388
Norma	0.04329±0.00065	0.028	0.06446±0.00073	0.086	0.01224±0.00006	0.448
RBP	0.05837±0.00140	0.028	0.09652±0.00253	0.075	0.02498±0.00034	0.200
Forgetron	0.05848±0.00216	0.037	0.09742±0.00334	0.106	0.02483±0.00042	0.269
Projectron	0.04023±0.00080	0.027	0.05683±0.00084	0.070	0.01280±0.00010	0.183
BOGD	0.05270±0.00134	0.024	0.08936±0.00198	0.064	0.01558±0.00017	0.175
FOGD	**0.04009±0.00071**	**0.016**	0.05590±0.00073	**0.037**	0.01169±0.00005	**0.104**
NOGD	0.04063±0.00043	0.031	**0.05356±0.00076**	0.073	**0.01138±0.00007**	0.202
Algorithm	parkinsons		cpusmall		cadata	
	Squared Loss	Time	Squared Loss	Time	Squared Loss	Time
OGD	0.04835±0.00018	2.025	0.02508±0.00009	1.905	0.03976 ±0.00018	11.63
Perceptron	0.05306±0.00045	2.116	0.02660±0.00015	1.257	0.04155±0.00019	11.50
Norma	0.05084±0.00018	1.385	0.03403±0.00014	2.060	0.05739±0.00008	8.45
RBP	0.07540±0.00102	0.349	0.04895±0.00058	0.449	0.08115±0.00029	1.09
Forgetron	0.07488±0.00114	0.496	0.04905±0.00062	0.581	0.08128±0.00061	1.54
Projectron	0.05306±0.00046	0.320	0.02660±0.00015	0.375	0.04155±0.00020	1.00
BOGD	0.06159±0.00037	0.295	0.04972±0.00048	0.406	0.07259±0.00031	0.94
FOGD	0.04909±0.00020	**0.187**	0.02577±0.00050	**0.217**	0.04097±0.00015	**0.55**
NOGD	**0.04896±0.00068**	0.336	**0.02559±0.00024**	0.427	**0.03983±0.00018**	1.05
Algorithm	casp		slice		year	
	Squared Loss	Time	Squared Loss	Time	Squared Loss	Time
RBP	0.12425±0.00048	2.56	0.04799±0.00025	22.13	0.03151±0.00007	89.7
Forgetron	0.12455±0.00046	3.76	0.04843±0.00024	35.43	0.03148±0.00005	139.7
Projectron	0.08709±0.00021	2.40	0.01493±0.00142	21.84	0.01627±0.00013	87.1
BOGD	0.09683±0.00012	2.23	0.04730±0.00011	21.61	0.05430±0.00002	88.3
FOGD	0.08021±0.00031	**1.37**	**0.00726±0.00019**	**4.65**	**0.01427±0.00004**	**19.3**
NOGD	**0.07844±0.00008**	2.51	0.02636±0.00460	22.05	0.01519±0.00021	89.1

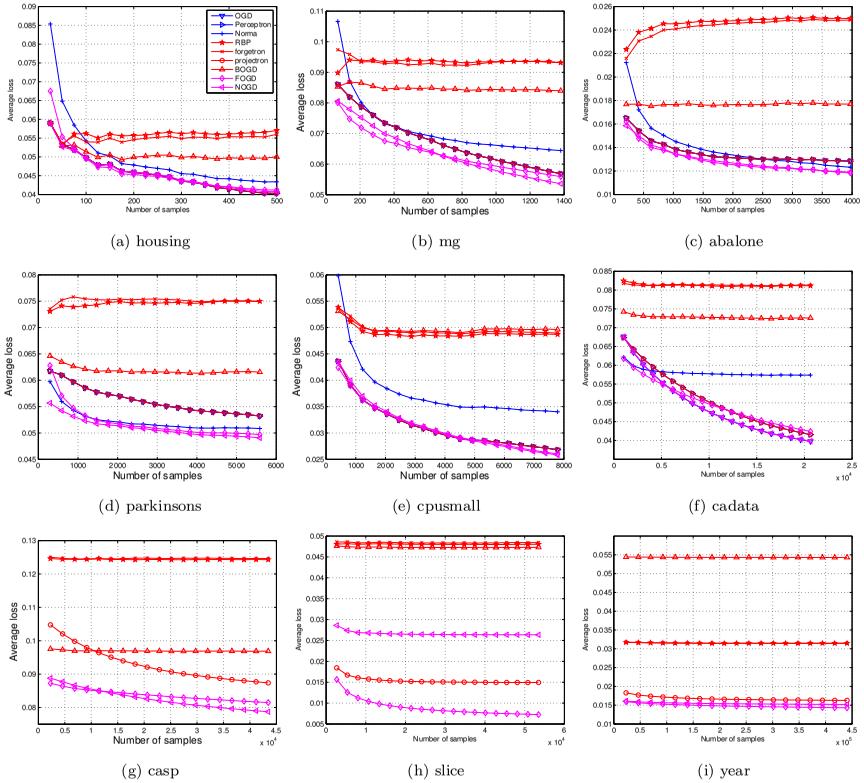
Table 9: Evaluation of Large-scale Online Kernel Learning on Regression Task (Time in Seconds).

loss for most cases while spending comparable or even lower time cost. This encouraging results again validate the advantages of the proposed technique for online kernel regression tasks.

Finally, by examining the two proposed algorithms, FOGD and NOGD, we found that they in general achieve fairly comparable regression accuracy, while FOGD tends to perform more efficiently than NOGD in terms of running time cost. This is primarily because NOGD has to involve the Nystrom matrix approximation which could be computationally intensive.

7.5 Comparison between FOGD and NOGD

In the previous experiments, we have made some comparisons of different budget online kernel learning algorithms for different learning tasks, in which the proposed algorithms

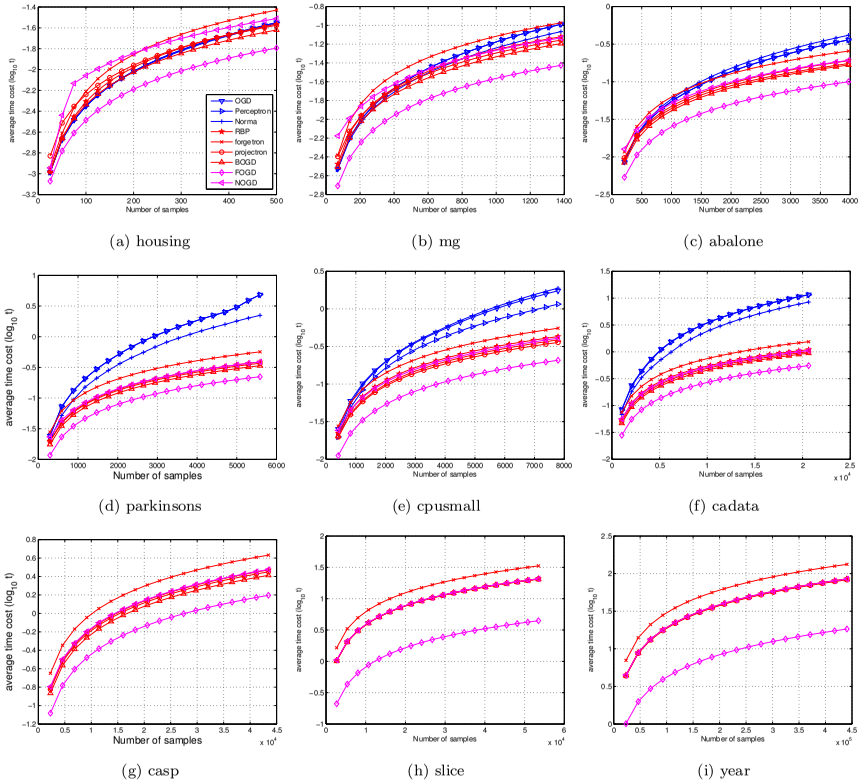


(a) housing (b) mg (c) abalone

(d) parkinsons (e) cpusmall (f) cadata

(g) casp (h) slice (i) year

Figure 5: Evaluation of online average squared loss on the regression tasks (best viewed in color).



(a) housing (b) mg (c) abalone

(d) parkinsons (e) cpusmall (f) cadata

(g) casp (h) slice (i) year

Figure 6: Evaluation of online average time cost on online regression tasks (best viewed in color).

show promising performance. In this section, we conduct both quantitative comparison and in-depth qualitative analysis of the two proposed algorithms in order to better understand their strengths and weaknesses in different scenarios. Specifically, we summarize the comparison of the two algorithms as follows.

First of all, as observed in the previous experiments, the two proposed algorithms in general tends to achieve comparable learning accuracy for most cases. However, NOGD outperforms FOGD in batch setting while FOGD is more accurate in online setting. In terms of running time costs, the result seems relatively implementation dependent. Specifically, when comparing the Matlab implementations of both algorithms, FOGD is faster, while NOGD is faster when comparing their C++ implementations. We conjecture that the reason is primarily because the FOGD algorithm is naturally easier for parallelization than NOGD. When running the Matlab implementations, FOGD may take advantages of Matlab-embedded speedup with implicit multi-core parallelization. While running the C++ implementations, we do not explicitly engage any parallelization, and thus NOGD is faster than FOGD when no parallelization is exploited.

Second, the efficiency performance of the two proposed algorithms also depends on the dataset size. For small-sized datasets, FOGD tends to be more efficient, while NOGD tends to be more efficient on larger-sized datasets. The main reason is that a key step of

NOGD is the Nystrom approximation that involves the eigen-decomposition. The eigen-decomposition computation could be potentially very computationally intensive a small-scale dataset, but relatively small or even negligible for a large-scale dataset. By contrast, FOGD does not involve eigen-decomposition and thus does not suffer from such issue for small-scale datasets.

Third, FOGD suffers from high space complexity (high memory cost) when handling datasets with relatively high dimensionality. This is because FOGD has to maintain a $\rho_f B \times d$ matrix in memory for the random Fourier features computation, while NOGD only needs to keep $B \times \rho_n B$ matrix for storing the feature mapping matrix. For a large-scale high-dimensional learning task where $d \gg B$ and $\rho_f > 1$, FOGD will clearly suffer a much higher memory cost than NOGD.

Forth, FOGD has some restrictions in terms of applicable kernels, e.g., shift-invariant kernels as mentioned before. It may be difficult to be generalized for other divers kernels. By contrast, NOGD is based on the the Nyström approximation which only requires the computation of kernel matrix and does not have a restriction on the applicable kernel type as long as it is a valid kernel.

Finally, FOGD may suffer from some practical limitations and implementation challenges for novel feature extension in some real-world applications. For example, consider learning tasks with stream data where novel features may arrive at different time periods in the online learning process. At the beginning of the online learning task, it is impossible to know the full set of features. During the online learning process, whenever a novel feature appears, FOGD has to update the list of D random Fourier components by expanding their dimensionality. Such kind of updating process usually involves a series of memory operations, such as new memory space allocation, copying existing vectors, and freeing memory space of obsolete data, which could be quite expensive if novel feature appears frequently. By contrast, NOGD suffers less for the novel feature extension issue in that we can simply treat the value of a novel feature as zero when computing kernel value between an existing support vector and a new example with the novel feature.

8. Conclusions

This paper presented a novel framework of large-scale online kernel learning via functional approximation, going beyond conventional online kernel methods that often adopt the budget maintenance strategy for ensuring the size of support vector is bounded. The basic idea of our framework is to approximate a kernel function or kernel matrix by exploring functional approximation techniques, which transforms the online kernel learning task into an approximate linear online learning problem in a new kernel-inducing feature space that can be further resolved by applying existing efficient and scalable online algorithms. We presented two new algorithms for large-scale online kernel learning tasks: Fourier Online Gradient Descent (FOGD) and Nyström Online Gradient Descent (NOGD), and applied them to tackle different tasks, including binary classification, multi-class classification, and regression tasks. Our promising results on large-scale datasets show the proposed new algorithms are able to achieve the state-of-the-art performance in both learning efficacy and efficiency in comparison to a variety of existing techniques. By comparing the two proposed algorithms, we found that they in general achieve quite comparable learning performance

for most cases, while have different advantages and disadvantages under different scenarios. As the first comprehensive work of exploring functional approximation for large-scale online kernel learning, our framework is generic and can be extended to tackle different learning tasks in other settings (e.g., structured prediction). To facilitate other researchers to re-produce our results, we have released the source code of our implementations. In our future work, we plan to extend our work by exploring parallel computing techniques to make kernel methods practical for massive-scale data analytics tasks.

Acknowledgments

This work was supported by Singapore MOE tier 1 research grant (C220/MSS14C003). This work was done when Jialei Wang visited Dr Hoi's group.

References

Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

Radha Chitta, Rong Jin, Timothy C Havens, and Anil K Jain. Approximate kernel k-means: Solution to large scale kernel clustering. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 895–903. ACM, 2011.

Radha Chitta, Rong Jin, and Anil Jain. Efficient kernel clustering using random fourier features. In *IEEE International Conference on Data Mining*, 2012.

Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *International Conference on Artificial Intelligence and Statistics*, 2010.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

Koby Crammer and Yoram Singer. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2-3):201–233, 2002.

Koby Crammer, Jaz S Kandola, and Yoram Singer. Online classification on a budget. In *Neural Information Processing Systems*, volume 2, page 5, 2003.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a fixed budget. In *Neural Information Processing Systems*, 2005.

Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of the International Conference on Machine Learning*, pages 264–271, 2008.

Yoav Freund and Robert E. Schapire. Large margin classification using the perceptron algorithm. *Maching Learning.*, 37(3):277–296, 1999.

Steven C. H. Hoi, Rong Jin, and Michael R. Lyu. Learning nonparametric kernel matrices from pairwise constraints. In *Proceedings of the International Conference on Machine Learning*, pages 361–368, Corvalis, Oregon, 2007. ISBN 978-1-59593-793-3.

Steven C. H. Hoi, Rong Jin, Peilin Zhao, and Tianbao Yang. Online multiple kernel classification. *Machine Learning*, 90(2):289–316, 2013.

Steven CH Hoi, Michael R Lyu, and Edward Y Chang. Learning the unified kernel machines for classification. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 187–196. ACM, 2006.

Steven C.H. Hoi, Jialei Wang, and Peilin Zhao. Libol: A library for online learning algorithms. *Journal of Machine Learning Research*, 15:495–499, 2014. URL `http://jmlr.org/papers/v15/hoi14a.html`.

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, 2002.

Jyrki Kivinen, Alex J. Smola, and Robert C. Williamson. Online learning with kernels. In *Neural Information Processing Systems*, pages 785–792, 2001.

Bin Li, Peilin Zhao, Steven CH Hoi, and Vivekanand Gopalkrishnan. Pamr: Passive aggressive mean reversion strategy for portfolio selection. *Machine learning*, 87(2):221–258, 2012.

Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, and Joseph M Hellerstein. Distributed graphlab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment*, 5(8):716–727, 2012.

Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the International Conference on Machine Learning*, pages 681–688. ACM, 2009.

Francesco Orabona, Joseph Keshet, and Barbara Caputo. The projectron: a bounded kernel-based perceptron. In *Proceedings of the International Conference on Machine Learning*, pages 720–727, 2008.

Francesco Orabona, Joseph Keshet, and Barbara Caputo. Bounded kernel-based online learning. *Journal of Machine Learning Research*, 10:2643–2666, 2009.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems*, 2007.

Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

W. Rudin. *Fourier Analysis on Groups*. Wiley-Interscience, 1990.

Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *Conference on Learning Theory*, pages 416–426, 2001.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: primal estimated sub-gradient solver for svm. *Math. Program.*, 127(1):3–30, 2011.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge university press, 2004.

Ameet Talwalkar, Sanjiv Kumar, and Henry A. Rowley. Large-scale manifold learning. In *Computer Vision and Pattern Recognition*, 2008.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

Vladimir N Vapnik. *Statistical Learning Theory*. Wiley, 1998.

Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. Cost-sensitive online classification. In *IEEE International Conference on Data Mining*, pages 1140–1145, 2012a.

Jialei Wang, Steven CH Hoi, Peilin Zhao, Jinfeng Zhuang, and Zhi-yong Liu. Large scale online kernel classification. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 1750–1756. AAAI Press, 2013.

Zhuang Wang and Slobodan Vucetic. Twin vector machines for online learning on a budget. In *International Conference on Data Mining*, pages 906–917. SIAM, 2009.

Zhuang Wang and Slobodan Vucetic. Online passive-aggressive algorithms on a budget. In *International Conference on Artificial Intelligence and Statistics*, pages 908–915, 2010.

Zhuang Wang, Koby Crammer, and Slobodan Vucetic. Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale svm training. *Journal of Machine Learning Research*, 13:3103–3131, 2012b.

Christopher K. I. Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Neural Information Processing Systems*, pages 682–688, 2000.

Hao Xia, Pengcheng Wu, and Steven C. H. Hoi. Online multi-modal distance learning for scalable multimedia retrieval. In *ACM International Conference on Web Search and Data Mining*, pages 455–464, 2013.

Tianbao Yang, Yufeng Li, Mehrdad Mahdavi, Rong Jin, and Zhi hua Zhou. Nystrom method vs random fourier features: A theoretical and empirical comparison. In *Neural Information Processing Systems*, 2012.

Kai Zhang and James T. Kwok. Density-weighted nyström method for computing large kernel eigensystems. *Neural Computation*, 21(1):121–146, 2009.

Kai Zhang, Liang Lan, Zhuang Wang, and Fabian Moerchen. Scaling up kernel svm on limited resources: A low-rank linearization approach. In *International Conference on Artificial Intelligence and Statistics*, pages 1425–1434, 2012.

Peilin Zhao and Steven CH Hoi. Otl: A framework of online transfer learning. In *Proceedings of the International Conference on Machine Learning*, 2010.

Peilin Zhao, Steven C. H. Hoi, and Rong Jin. Double updating online learning. *Journal of Machine Learning Research*, 12:1587–1615, 2011.

Peilin Zhao, Jialei Wang, Pengcheng Wu, Rong Jin, and Steven C. H. Hoi. Fast bounded online gradient descent algorithms for scalable kernel-based online learning. In *Proceedings of the International Conference on Machine Learning*, 2012.

Journal of Machine Learning Research 17 (2016) 1-41 Submitted 5/14; Revised 9/15; Published 4/16

Kernel Mean Shrinkage Estimators

Krikamol Muandet[*] KRIKAMOL@TUEBINGEN.MPG.DE
Empirical Inference Department, Max Planck Institute for Intelligent Systems
Spemannstraße 38, Tübingen 72076, Germany

Bharath Sriperumbudur[*] BKS18@PSU.EDU
Department of Statistics, Pennsylvania State University
University Park, PA 16802, USA

Kenji Fukumizu FUKUMIZU@ISM.AC.JP
The Institute of Statistical Mathematics
10-3 Midoricho, Tachikawa, Tokyo 190-8562 Japan

Arthur Gretton ARTHUR.GRETTON@GMAIL.COM
Gatsby Computational Neuroscience Unit, CSML, University College London
Alexandra House, 17 Queen Square, London - WC1N 3AR, United Kingdom
ORCID 0000-0003-3169-7624

Bernhard Schölkopf BS@TUEBINGEN.MPG.DE
Empirical Inference Department, Max Planck Institute for Intelligent Systems
Spemannstraße 38, Tübingen 72076, Germany

Editor: Ingo Steinwart

Abstract

A mean function in a reproducing kernel Hilbert space (RKHS), or a kernel mean, is central to kernel methods in that it is used by many classical algorithms such as kernel principal component analysis, and it also forms the core inference step of modern kernel methods that rely on embedding probability distributions in RKHSs. Given a finite sample, an empirical average has been used commonly as a standard estimator of the true kernel mean. Despite a widespread use of this estimator, we show that it can be improved thanks to the well-known Stein phenomenon. We propose a new family of estimators called kernel mean shrinkage estimators (KMSEs), which benefit from both theoretical justifications and good empirical performance. The results demonstrate that the proposed estimators outperform the standard one, especially in a "large d, small n" paradigm.

Keywords: covariance operator, James-Stein estimators, kernel methods, kernel mean, shrinkage estimators, Stein effect, Tikhonov regularization

1. Introduction

This paper aims to improve the estimation of the mean function in a reproducing kernel Hilbert space (RKHS), or a kernel mean, from a finite sample. A kernel mean is defined with respect to a probability distribution $\mathbb{P}$ over a measurable space $\mathcal{X}$ by

$$\mu_{\mathbb{P}} \triangleq \int_{\mathcal{X}} k(x,\cdot)\ \mathrm{d}\mathbb{P}(x) \in \mathcal{H}, \tag{1}$$

∗. Contributed equally

where $\mu_{\mathbb{P}}$ is a Bochner integral (see, *e.g.*, Diestel and Uhl (1977, Chapter 2) and Dinculeanu (2000, Chapter 1) for a definition of Bochner integral) and $\mathcal{H}$ is a separable RKHS endowed with a measurable reproducing kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that $\int_{\mathcal{X}} \sqrt{k(x,x)}\ \mathrm{d}\mathbb{P}(x) < \infty$.[1] Given an i.i.d sample $x_1, x_2, \ldots, x_n$ from $\mathbb{P}$, the most natural estimate of the true kernel mean is empirical average

$$\hat{\mu}_{\mathbb{P}} \triangleq \frac{1}{n}\sum_{i=1}^{n} k(x_i,\cdot)\,. \tag{2}$$

We refer to this estimator as a *kernel mean estimator* (KME). Though it is the most commonly used estimator of the true kernel mean, the key contribution of this work is to show that there exist estimators that can improve upon this standard estimator.

The kernel mean has recently gained attention in the machine learning community, thanks to the introduction of Hilbert space embedding for distributions (Berlinet and Thomas-Agnan, 2004; Smola et al., 2007). Representing the distribution as a mean function in the RKHS has several advantages. First, if the kernel k is *characteristic*, the map $\mathbb{P} \mapsto \mu_{\mathbb{P}}$ is injective.[2] That is, it preserves all information about the distribution (Fukumizu et al., 2004; Sriperumbudur et al., 2008). Second, basic operations on the distribution can be carried out by means of inner products in RKHS, *e.g.*, $\mathbb{E}_{\mathbb{P}}[f(x)] = \langle f, \mu_{\mathbb{P}}\rangle_{\mathcal{H}}$ for all $f \in \mathcal{H}$, which is an essential step in probabilistic inference (see, e.g., Song et al., 2011). Lastly, no intermediate density estimation is required, for example, when testing for homogeneity from finite samples. Thus, the algorithms become less susceptible to the curse of dimensionality; see, *e.g.*, Wasserman (2006, Section 6.5) and Sriperumbudur et al. (2012).

The aforementioned properties make Hilbert space embedding of distributions appealing to many algorithms in modern kernel methods, namely, two-sample testing via maximum mean discrepancy (MMD) (Gretton et al., 2007, 2012), kernel independence tests (Gretton et al., 2008), Hilbert space embedding of HMMs (Song et al., 2010), and kernel Bayes rule (Fukumizu et al., 2011). The performance of these algorithms relies directly on the quality of the empirical estimate $\hat{\mu}_{\mathbb{P}}$.

In addition, the kernel mean has played much more fundamental role as a basic building block of many kernel-based learning algorithms (Vapnik, 1998; Schölkopf et al., 1998). For instance, nonlinear component analyses, such as kernel principal component analysis (KPCA), kernel Fisher discriminant analysis (KFDA), and kernel canonical correlation analysis (KCCA), rely heavily on mean functions and covariance operators in RKHS (Schölkopf et al., 1998; Fukumizu et al., 2007). The kernel K-means algorithm performs clustering in feature space using mean functions as representatives of the clusters (Dhillon et al., 2004). Moreover, the kernel mean also served as a basis in early development of algorithms for classification, density estimation, and anomaly detection (Shawe-Taylor and Cristianini, 2004, Chapter 5). All of these employ the empirical average in (2) as an estimate of the true kernel mean.

1. The separability of $\mathcal{H}$ and measurability of k ensures that $k(\cdot, x)$ is a $\mathcal{H}$-valued measurable function for all $x \in \mathcal{X}$ (Steinwart and Christmann, 2008, Lemma A.5.18). The separability of $\mathcal{H}$ is guaranteed by choosing $\mathcal{X}$ to be a separable topological space and k to be continuous (Steinwart and Christmann, 2008, Lemma 4.33).
2. The notion of characteristic kernel is closely related to the notion of universal kernel. In brief, if the kernel is universal, it is also characteristic, but the reverse direction is not necessarily the case. See, *e.g.*, Sriperumbudur et al. (2011), for more detailed accounts on this topic.

We show in this work that the empirical estimator in (2) is, in a certain sense, not optimal, i.e., there exist "better" estimators (more below), and then propose simple estimators that outperform the empirical estimator. While it is reasonable to argue that $\hat{\mu}_{\mathbb{P}}$ is the "best" possible estimator of $\mu_{\mathbb{P}}$ if nothing is known about $\mathbb{P}$ (in fact $\hat{\mu}_{\mathbb{P}}$ is minimax in the sense of van der Vaart (1998, Theorem 25.21, Example 25.24)), in this paper we show that "better" estimators of $\mu_{\mathbb{P}}$ can be constructed if mild assumptions are made on $\mathbb{P}$. This work is to some extent inspired by Stein's seminal work in 1955, which showed that the maximum likelihood estimator (MLE) of the mean, θ of a multivariate Gaussian distribution $\mathcal{N}(\theta, \sigma^2\mathbf{I})$ is "inadmissible" (Stein, 1955)—i.e., there exists a better estimator—though it is minimax optimal. In particular, Stein showed that there exists an estimator that always achieves smaller total mean squared error regardless of the true $\theta \in \mathbb{R}^d$, when $d \geq 3$. Perhaps the best known estimator of such kind is James-Steins estimator (James and Stein, 1961). Formally, if $X \sim \mathcal{N}(\theta, \sigma^2\mathbf{I})$ with $d \geq 3$, the estimator $\delta(X) = X$ for θ is inadmissible in mean squared sense and is dominated by the following estimator

$$\delta_{\mathrm{JS}}(X) = \left(1 - \frac{(d-2)\sigma^2}{\|X\|^2}\right) X, \tag{3}$$

i.e., $\mathbb{E}\|\delta_{\mathrm{JS}}(X) - \theta\|^2 \leq \mathbb{E}\|\delta(X) - \theta\|^2$ for all θ and there exists at least one θ for which $\mathbb{E}\|\delta_{\mathrm{JS}}(X) - \theta\|^2 < \mathbb{E}\|\delta(X) - \theta\|^2$.

Interestingly, the James-Stein estimator is itself inadmissible, and there exists a wide class of estimators that outperform the MLE, see, *e.g.*, Berger (1976). Ultimately, Stein's result suggests that one can construct estimators better than the usual empirical estimator if the relevant parameters are estimated jointly and if the definition of risk ultimately looks at all of these parameters (or coordinates) together. This finding is quite remarkable as it is counter-intuitive as to why joint estimation should yield better estimators when all parameters are mutually independent (Efron and Morris, 1977). Although the Stein phenomenon has been extensively studied in the statistics community, it has not received much attention in the machine learning community.

The James-Stein estimator is a special case of a larger class of estimators known as *shrinkage estimators* (Gruber, 1998). In its most general form, the shrinkage estimator is a combination of a model with low bias and high variance, and a model with high bias but low variance. For example, one might consider the following estimator:

$$\hat{\theta}_{\mathrm{shrink}} \triangleq \lambda\tilde{\theta} + (1-\lambda)\hat{\theta}_{\mathrm{ML}},$$

where $\lambda \in [0,1]$, $\hat{\theta}_{\mathrm{ML}}$ denotes the usual maximum likelihood estimate of θ, and $\tilde{\theta}$ is an arbitrary point in the input space. In the case of James-Stein estimator, we have $\tilde{\theta} = 0$. Our proposal of shrinkage estimator to estimate $\mu_{\mathbb{P}}$ will rely on the same principle, but will differ fundamentally from the Stein's seminal works and those along this line in two aspects. First, our setting is "non-parametric" in the sense that we do not assume any parametric form for the distribution, whereas most of traditional works focus on some specific distributions, *e.g.*, the Gaussian distribution. The non-parametric setting is very important in most applications of kernel means because it allows us to perform statistical inference without making any assumption on the parametric form of the true distribution $\mathbb{P}$. Second, our setting involves a "non-linear feature map" into a high-dimensional space.

For example, if we use the Gaussian RBF kernel (see (6)), the mean function $\mu_{\mathbb{P}}$ lives in an infinite-dimensional space. As a result, higher moments of the distribution come into play and therefore one cannot adopt Stein's setting straightforwardly as it involves only the first moment. A direct generalization of James-Stein estimator to infinite-dimensional Hilbert space has been considered, for example, in Berger and Wolpert (1983); Mandelbaum and Shepp (1987); Privault and Rveillac (2008). In those works, the parameter to be estimated is assumed to be the mean of a Gaussian measure on the Hilbert space from which samples are drawn. In contrast, our setting involves samples that are drawn from $\mathbb{P}$ defined on an arbitrary measurable space, and not from a Gaussian measure defined on a Hilbert space.

1.1 Contributions

In the following, we present the main contributions of this work.

1. In Section 2.2, we propose kernel mean shrinkage estimators and show that these estimators can theoretically improve upon the standard empirical estimator, $\hat{\mu}_{\mathbb{P}}$ in terms of the mean squared error (see Theorem 1 and Proposition 4), however, requiring the knowledge of the true kernel mean. We relax this condition in Section 2.3 (see Theorem 5) where without requiring the knowledge of the true kernel mean, we construct shrinkage estimators that are *uniformly* better (in mean squared error) than the empirical estimator over a class of distributions $\mathcal{P}$. For bounded continuous translation invariant kernels, we show that $\mathcal{P}$ reduces to a class of distributions whose characteristic functions have an L^2-norm bounded by a given constant. Through concrete choices for $\mathcal{P}$ in Examples 1 and 2, we discuss the implications of the proposed estimator.

2. While the proposed estimators in Section 2.2 and 2.3 are theoretically interesting, they are not useful in practice as they require the knowledge of the true data generating distribution. In Section 2.4 (see Theorem 7), we present a completely data-dependent estimator (say $\check{\mu}_{\mathbb{P}}$)—referred to as B-KMSE—that is $\sqrt{n}$-consistent and satisfies

$$\mathbb{E}\|\check{\mu}_{\mathbb{P}} - \mu_{\mathbb{P}}\|^2_{\mathcal{H}} < \mathbb{E}\|\hat{\mu}_{\mathbb{P}} - \mu_{\mathbb{P}}\|^2_{\mathcal{H}} + O(n^{-3/2}) \text{ as } n \to \infty. \tag{4}$$

3. In Section 3, we present a regularization interpretation for the proposed shrinkage estimator, wherein the shrinkage parameter is shown to be directly related to the regularization parameter. Based on this relation, we present an alternative approach to choosing the shrinkage parameter (different from the one proposed in Section 2.4) through leave-one-out cross-validation, and show that the corresponding shrinkage estimator (we refer to it as R-KMSE) is also $\sqrt{n}$-consistent and satisfies (4).

4. The regularization perspective also sheds light on constructing new shrinkage estimators that incorporate specific information about the RKHS, based on which we present a new $\sqrt{n}$-consistent shrinkage estimator—referred to as S-KMSE—in Section 4 (see Theorem 13 and Remark 14) that takes into account spectral information of the covariance operator in RKHS. We establish the relation of S-KMSE to the problem of learning smooth operators (Grünewälder et al., 2013) on $\mathcal{H}$, and propose a leave-one-out cross-validation method to obtain a data-dependent shrinkage parameter. However, unlike B-KMSE and R-KMSE, it remains an open question as

to whether S-KMSE with a data-dependent shrinkage parameter is consistent and satisfies an inequality similar to (4). The difficulty in answering these questions lies with the complex form of the estimator, $\tilde{\mu}_{\mathbb{P}}$ which is constructed so as to capture the spectral information of the covariance operator.

5. In Section 6, we empirically evaluate the proposed shrinkage estimators of kernel mean on both synthetic data and several real-world scenarios including Parzen window classification, density estimation and discriminative learning on distributions. The experimental results demonstrate the benefits of our shrinkage estimators over the standard one.

While a shorter version of this work already appeared in Muandet et al. (2014a,b)—particularly, the ideas in Sections 2.2, 3 and 4—, this extended version provides a rigorous theoretical treatment (through Theorems 5, 7, 10, 13 and Proposition 15 which are new) for the proposed estimators and also contains additional experimental results.

2. Kernel Mean Shrinkage Estimators

In this section, we first provide some definitions and notation that are used throughout the paper, following which we present a shrinkage estimator of $\mu_{\mathbb{P}}$. The rest of the section presents various properties including the inadmissibility of the empirical estimator.

2.1 Definitions & Notation

For $a \triangleq (a_1, \ldots, a_d) \in \mathbb{R}^d$, $\|a\|_2 \triangleq \sqrt{\sum_{i=1}^d a_i^2}$. For a topological space $\mathcal{X}$, $C(\mathcal{X})$ (*resp.* $C_b(\mathcal{X})$) denotes the space of all continuous (*resp.* bounded continuous) functions on $\mathcal{X}$. For a locally compact Hausdorff space $\mathcal{X}$, $f \in C(\mathcal{X})$ is said to *vanish at infinity* if for every $\epsilon > 0$ the set $\{x : |f(x)| \geq \epsilon\}$ is compact. The class of all continuous f on $\mathcal{X}$ which vanish at infinity is denoted as $C_0(\mathcal{X})$. $M_b(\mathcal{X})$ (*resp.* $M_+^1(\mathcal{X})$) denotes the set of all finite Borel (*resp.* probability) measures defined on $\mathcal{X}$. For $\mathcal{X} \subset \mathbb{R}^d$, $L^r(\mathcal{X})$ denotes the Banach space of r-power ($r \geq 1$) Lebesgue integrable functions. For $f \in L^r(\mathcal{X})$, $\|f\|_{L^r} \triangleq \left(\int_{\mathcal{X}} |f(x)|^r \, \mathrm{d}x\right)^{1/r}$ denotes the L^r-norm of f for $1 \leq r < \infty$. The Fourier transform of $f \in L^1(\mathbb{R}^d)$ is defined as $f^\wedge(\omega) \triangleq (2\pi)^{-d/2} \int_{\mathbb{R}^d} f(x) e^{-\sqrt{-1}\omega^\top x} \, \mathrm{d}x$, $\omega \in \mathbb{R}^d$. The characteristic function of $\mathbb{P} \in M_+^1(\mathbb{R}^d)$ is defined as $\phi_{\mathbb{P}}(\omega) \triangleq \int e^{\sqrt{-1}\omega^\top x} \, \mathrm{d}\mathbb{P}(x)$, $\omega \in \mathbb{R}^d$.

An RKHS over a set $\mathcal{X}$ is a Hilbert space $\mathcal{H}$ consisting of functions on $\mathcal{X}$ such that for each $x \in \mathcal{X}$ there is a function $k_x \in \mathcal{H}$ with the property

$$\langle f, k_x \rangle_{\mathcal{H}} = f(x), \quad \forall f \in \mathcal{H}. \tag{5}$$

The function $k_x(\cdot) \triangleq k(x, \cdot)$ is called the *reproducing kernel* of $\mathcal{H}$ and the equality (5) is called the *reproducing property* of $\mathcal{H}$. The space $\mathcal{H}$ is endowed with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and norm $\|\cdot\|_{\mathcal{H}}$. Any symmetric and positive semi-definite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ uniquely determines an RKHS (Aronszajn, 1950). One of the most popular kernel functions is the Gaussian radial basis function (RBF) kernel on $\mathcal{X} = \mathbb{R}^d$,

$$k(x, y) = \exp\left(-\frac{\|x - y\|_2^2}{2\sigma^2}\right), \quad x, y \in \mathcal{X}, \tag{6}$$

where $\|\cdot\|_2$ denotes the Euclidean norm and $\sigma > 0$ is the bandwidth. For $x \in \mathcal{H}_1$ and $y \in \mathcal{H}_2$, $x \otimes y$ denotes the tensor product of x and y, and can be seen as an operator from $\mathcal{H}_2$ to $\mathcal{H}_1$ as $(x \otimes y)z = x\langle y, z \rangle_{\mathcal{H}_2}$ for any $z \in \mathcal{H}_2$, where $\mathcal{H}_1$ and $\mathcal{H}_2$ are Hilbert spaces.

We assume throughout the paper that we observe a sample $x_1, x_2, \ldots, x_n \in \mathcal{X}$ of size n drawn independently and identically (i.i.d.) from some unknown distribution $\mathbb{P}$ defined over a separable topological space $\mathcal{X}$. Denote by μ and $\hat{\mu}$ the true kernel mean (1) and its empirical estimate (2) respectively. We remove the subscript for ease of notation, but we will use $\mu_{\mathbb{P}}$ (resp. $\hat{\mu}_{\mathbb{P}}$) and μ (resp. $\hat{\mu}$) interchangeably. For the well-definedness of μ as a Bochner integral, throughout the paper we assume that k is continuous and $\int_{\mathcal{X}} k(x, x) \, \mathrm{d}\mathbb{P}(x) < \infty$ (see Footnote 1). We measure the quality of an estimator $\tilde{\mu} \in \mathcal{H}$ of μ by the risk function, $R : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$, $R(\mu, \tilde{\mu}) = \mathbb{E}\|\mu - \tilde{\mu}\|_{\mathcal{H}}^2$, where $\mathbb{E}$ denotes the expectation over the choice of random sample of size n drawn i.i.d. from the distribution $\mathbb{P}$. When $\tilde{\mu} = \hat{\mu}$, for the ease of notation, we will use Δ to denote $R(\mu, \hat{\mu})$, which can be rewritten as

$$\begin{aligned}
\Delta = \mathbb{E}\|\hat{\mu} - \mu\|_{\mathcal{H}}^2 &= \mathbb{E}\|\hat{\mu}\|_{\mathcal{H}}^2 - \|\mu\|_{\mathcal{H}}^2 = \frac{1}{n^2} \sum_{i,j=1}^n \mathbb{E}_{x_i, x_j} k(x_i, x_j) - \|\mu\|_{\mathcal{H}}^2 \\
&= \frac{1}{n^2} \sum_{i=1}^n \mathbb{E}_{x_i} k(x_i, x_i) + \frac{1}{n^2} \sum_{i \neq j}^n \mathbb{E}_{x_i, x_j} k(x_i, x_j) - \|\mu\|_{\mathcal{H}}^2 \\
&= \frac{1}{n} \left(\mathbb{E}_x k(x, x) - \mathbb{E}_{x, \tilde{x}} k(x, \tilde{x}) \right),
\end{aligned} \tag{7}$$

where $\|\mu\|_{\mathcal{H}}^2 = \mathbb{E}_{x, \tilde{x}}[k(x, \tilde{x})] \triangleq \mathbb{E}_{x \sim \mathbb{P}}[\mathbb{E}_{\tilde{x} \sim \mathbb{P}}[k(x, \tilde{x})]]$ with x and $\tilde{x}$ being independent copies. An estimator $\hat{\mu}_1$ is said to be *as good as* $\hat{\mu}_2$ if $R(\mu, \hat{\mu}_1) \leq R(\mu, \hat{\mu}_2)$ for any $\mathbb{P}$, and is *better than* $\hat{\mu}_2$ if it is as good as $\hat{\mu}_2$ and $R(\mu, \hat{\mu}_1) < R(\mu, \hat{\mu}_2)$ for at least one $\mathbb{P}$. An estimator is said to be *inadmissible* if there exists a better estimator.

2.2 Shrinkage Estimation of $\mu_{\mathbb{P}}$

We propose the following kernel mean estimator

$$\hat{\mu}_\alpha \triangleq \alpha f^* + (1 - \alpha)\hat{\mu} \tag{8}$$

where $\alpha \geq 0$ and f^* is a fixed, but arbitrary function in $\mathcal{H}$. Basically, it is a shrinkage estimator that shrinks the empirical estimator toward a function f^* by an amount specified by α. The choice of f^* can be arbitrary, but we will assume that f^* is chosen independent of the sample. If $\alpha = 0$, the estimator $\hat{\mu}_\alpha$ reduces to the empirical estimator $\hat{\mu}$. We denote by Δ_α the risk of the shrinkage estimator in (8), *i.e.*, $\Delta_\alpha \triangleq R(\mu, \hat{\mu}_\alpha)$.

Our first theorem asserts that the shrinkage estimator $\hat{\mu}_\alpha$ achieves smaller risk than that of the empirical estimator $\hat{\mu}$ given an appropriate choice of α, regardless of the function f^*.

Theorem 1 *Let $\mathcal{X}$ be a separable topological space. Then for all distributions $\mathbb{P}$ and continuous kernel k satisfying $\int k(x, x) \, \mathrm{d}\mathbb{P}(x) < \infty$, $\Delta_\alpha < \Delta$ if and only if*

$$\alpha \in \left(0, \frac{2\Delta}{\Delta + \|f^* - \mu\|_{\mathcal{H}}^2}\right). \tag{9}$$

In particular, $\arg\min_{\alpha \in \mathbb{R}} (\Delta_\alpha - \Delta)$ *is unique and is given by* $\alpha_* \triangleq \frac{\Delta}{\Delta + \|f^* - \mu\|_{\mathcal{H}}^2}$.

Proof Note that

$$\Delta_\alpha = \mathbb{E}\|\hat{\mu}_\alpha - \mu\|_{\mathcal{H}}^2 = \|\mathbb{E}[\hat{\mu}_\alpha] - \mu\|_{\mathcal{H}}^2 + \mathbb{E}\,\|\hat{\mu}_\alpha - \mathbb{E}\hat{\mu}_\alpha\|_{\mathcal{H}}^2 = \|\mathrm{Bias}(\hat{\mu}_\alpha)\|_{\mathcal{H}}^2 + \mathrm{Var}(\hat{\mu}_\alpha),$$

where

$$\mathrm{Bias}(\hat{\mu}_\alpha) = \mathbb{E}[\hat{\mu}_\alpha] - \mu = \mathbb{E}[\alpha f^* + (1-\alpha)\hat{\mu}] - \mu = \alpha(f^* - \mu)$$

and

$$\mathrm{Var}(\hat{\mu}_\alpha) = (1-\alpha)^2 \mathbb{E}\,\|\hat{\mu} - \mu\|_{\mathcal{H}}^2 = (1-\alpha)^2 \Delta.$$

Therefore,

$$\Delta_\alpha = \alpha^2\,\|f^* - \mu\|_{\mathcal{H}}^2 + (1-\alpha)^2\Delta, \tag{10}$$

i.e., $\Delta_\alpha - \Delta = \alpha^2\left[\Delta + \|f^* - \mu\|_{\mathcal{H}}^2\right] - 2\alpha\Delta$. This is clearly negative if and only if (9) holds and is uniquely minimized at $\alpha_* \triangleq \frac{\Delta}{\Delta + \|f^* - \mu\|_{\mathcal{H}}^2}$. ■

Remark 2 *(i) The shrinkage estimator always improves upon the standard one regardless of the direction of shrinkage, as specified by the choice of f^*. In other words, there exists a wide class of kernel mean estimators that achieve smaller risk than the standard one.*

(ii) The range of α depends on the choice of f^*. The further f^* is from μ, the smaller the range of α becomes. Thus, the shrinkage gets smaller if f^* is chosen such that it is far from the true kernel mean. This effect is akin to James-Stein estimator.

(iii) From (9), since $0 < \alpha < 2$, i.e., *$0 < (1-\alpha)^2 < 1$, it follows that $\mathrm{Var}(\hat{\mu}_\alpha) < \mathrm{Var}(\hat{\mu}) = \Delta$,* i.e., *the shrinkage estimator always improves upon the empirical estimator in terms of the variance. Further improvement can be gained by reducing the bias by incorporating the prior knowledge about the location of μ via f^*. This implies that we can potentially gain "twice" by adopting the shrinkage estimator: by reducing variance of the estimator and by incorporating prior knowledge in choosing f^* such that it is close to the true kernel mean.*

While Theorem 1 shows $\hat{\mu}$ to be inadmissible by providing a family of estimators that are better than $\hat{\mu}$, the result is not useful as all these estimators require the knowledge of μ (which is the parameter of interest) through the range of α given in (9). In Section 2.3, we investigate Theorem 1 and show that $\hat{\mu}_\alpha$ can be constructed under some weak assumptions on $\mathbb{P}$, without requiring the knowledge of μ. From (9), the existence of positive α is guaranteed if and only if the risk of the empirical estimator is non-zero. Under some assumptions on k, the following result shows that $\Delta = 0$ if and only if the distribution $\mathbb{P}$ is a Dirac distribution, *i.e.*, the distribution $\mathbb{P}$ is a point mass. This result ensures, in many non-trivial cases, a non-empty range of α for which $\Delta_\alpha - \Delta < 0$.

Proposition 3 *Let $k(x,y) = \psi(x-y)$, $x, y \in \mathbb{R}^d$ be a characteristic kernel where $\psi \in C_b(\mathbb{R}^d)$ is positive definite. Then $\Delta = 0$ if and only if $\mathbb{P} = \delta_x$ for some $x \in \mathbb{R}^d$.*

Proof See Section 5.1. ■

2.2.1 Positive-part Shrinkage Estimator

Similar to James-Stein estimator, we can show that the positive-part version of $\hat{\mu}_\alpha$ also outperforms $\hat{\mu}$, where the positive-part estimator is defined by

$$\hat{\mu}_\alpha^+ \triangleq \alpha f^* + (1-\alpha)_+ \hat{\mu} \tag{11}$$

with $(a)_+ \triangleq a$ if $a > 0$ and zero otherwise. Equation (11) can be rewritten as

$$\hat{\mu}_\alpha^+ = \begin{cases} \alpha f^* + (1-\alpha)\hat{\mu}, & 0 \le \alpha \le 1 \\ \alpha f^* & 1 < \alpha < 2. \end{cases} \tag{12}$$

Let $\Delta_\alpha^+ \triangleq \mathbb{E}\|\hat{\mu}_\alpha^+ - \mu\|_{\mathcal{H}}^2$ be the risk of the positive-part estimator. Then, the following result shows that $\Delta_\alpha^+ \le \Delta_\alpha$, given that α satisfies (9).

Proposition 4 *For any α satisfying (9), we have that $\Delta_\alpha^+ \le \Delta_\alpha < \Delta$.*

Proof According to (12), we decompose the proof into two parts. First, if $0 \le \alpha \le 1$, $\hat{\mu}_\alpha$ and $\hat{\mu}_\alpha^+$ behave exactly the same. Thus, $\Delta_\alpha^+ = \Delta_\alpha$. On the other hand, when $1 < \alpha < 2$, the bias-variance decomposition of these estimators yields

$$\Delta_\alpha = \alpha^2\|f^* - \mu\|_{\mathcal{H}}^2 + (1-\alpha)^2\mathbb{E}\|\hat{\mu} - \mu\|_{\mathcal{H}}^2 \quad \text{and} \quad \Delta_\alpha^+ = \alpha^2\|f^* - \mu\|_{\mathcal{H}}^2.$$

It is easy to see that $\Delta_\alpha^+ < \Delta_\alpha$ when $1 < \alpha < 2$. This concludes the proof. ■

Proposition 4 implies that, when estimating α, it is better to restrict the value of α to be smaller than 1, although it can be greater than 1, as suggested by Theorem 1. The reason is that if $0 \le \alpha \le 1$, the bias is an increasing function of α, whereas the variance is a decreasing function of α. On the other hand, if $\alpha > 1$, both bias and variance become increasing functions of α. We will see later in Section 3 that $\hat{\mu}_\alpha$ and $\hat{\mu}_\alpha^+$ can be obtained naturally as a solution to a regularized regression problem.

2.3 Consequences of Theorem 1

As mentioned before, while Theorem 1 is interesting from the perspective of showing that the shrinkage estimator, $\hat{\mu}_\alpha$ performs better—in the mean squared sense—than the empirical estimator, it unfortunately relies on the fact that $\mu_{\mathbb{P}}$ (*i.e.*, the object of interest) is known, which makes $\hat{\mu}_\alpha$ uninteresting. Instead of knowing $\mu_{\mathbb{P}}$, which requires the knowledge of $\mathbb{P}$, in this section, we show that a shrinkage estimator can be constructed that performs better than the empirical estimator, uniformly over a class of probability distributions. To this end, we introduce the notion of an oracle upper bound.

Let $\mathcal{P}$ be a class of probability distributions $\mathbb{P}$ defined on a measurable space $\mathcal{X}$. We define an oracle upper bound as

$$U_{k,\mathcal{P}} \triangleq \inf_{\mathbb{P}\in\mathcal{P}} \frac{2\Delta}{\Delta + \|f^* - \mu\|_{\mathcal{H}}^2}.$$

It follows immediately from Theorem 1 and the definition of $U_{k,\mathcal{P}}$ that if $U_{k,\mathcal{P}} \neq 0$, then for any $\alpha \in (0, U_{k,\mathcal{P}})$, $\Delta_\alpha - \Delta < 0$ holds "uniformly" for all $\mathbb{P} \in \mathcal{P}$. Note that by virtue

of Proposition 3, the class $\mathcal{P}$ cannot contain the Dirac measure δ_x (for any $x \in \mathbb{R}^d$) if the kernel k is translation invariant and characteristic on $\mathbb{R}^d$. Below we give concrete examples of $\mathcal{P}$ for which $U_{k,\mathcal{P}} \neq 0$ so that the above uniformity statement holds. In particular, we show in Theorem 5 below that for $\mathcal{X} = \mathbb{R}^d$, if a non-trivial bound on the L^2-norm of the characteristic function of $\mathbb{P}$ is known, it is possible to construct shrinkage estimators that are better (in mean squared error) than the empirical average. In such a case, unlike in Theorem 1, α does not depend on the individual distribution $\mathbb{P}$, but only on an upper bound associated with a class $\mathcal{P}$.

Theorem 5 *Let $k(x,y) = \psi(x-y)$, $x, y \in \mathbb{R}^d$ with $\psi \in C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$ and ψ is a positive definite function with $\psi(0) > 0$. For a given constant $A \in (0,1)$, let $A_\psi := \frac{A(2\pi)^{d/2}\psi(0)}{\|\psi\|_{L_1}}$ and*

$$\mathcal{P}_{k,A} \triangleq \left\{ \mathbb{P} \in M_+^1(\mathbb{R}^d) : \|\phi_{\mathbb{P}}\|_{L^2} \leq \sqrt{A_\psi} \right\},$$

where $\phi_{\mathbb{P}}$ denotes the characteristic function of $\mathbb{P}$. Then for all $\mathbb{P} \in \mathcal{P}_{k,A}$, $\Delta_\alpha < \Delta$ if

$$\alpha \in \left(0, \frac{2(1-A)}{1 + (n-1)A + \frac{n\|f^*\|_{\mathcal{H}}^2}{\psi(0)} + \frac{2n\sqrt{A}\|f^*\|_{\mathcal{H}}}{\sqrt{\psi(0)}}} \right].$$

Proof By Theorem 1, we have that

$$\Delta_\alpha < \Delta, \ \forall\ \alpha \in \left(0, \frac{2\Delta}{\Delta + \|f^* - \mu\|_{\mathcal{H}}^2} \right). \tag{13}$$

Consider

$$\begin{aligned} \frac{\Delta}{\Delta + \|f^* - \mu\|_{\mathcal{H}}^2} &= \frac{\mathbb{E}_x k(x,x) - \mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x) - \mathbb{E}_{x,\tilde{x}} k(x,\tilde{x}) + n\|f^* - \mu\|_{\mathcal{H}}^2} \\ &\overset{(\dagger)}{=} \frac{1 - \frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)}}{1 + (n-1)\frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)} + \frac{n\|f^*\|_{\mathcal{H}}^2}{\mathbb{E}_x k(x,x)} - \frac{2n\langle f^*, \mu\rangle_{\mathcal{H}}}{\mathbb{E}_x k(x,x)}} \\ &\geq \frac{1 - \frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)}}{1 + (n-1)\frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)} + \frac{n\|f^*\|_{\mathcal{H}}^2}{\mathbb{E}_x k(x,x)} + \frac{2n\|f^*\|_{\mathcal{H}}\sqrt{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}}{\mathbb{E}_x k(x,x)}}, \end{aligned} \tag{14}$$

where the division by $\mathbb{E}_x k(x,x)$ in $(\dagger)$ is valid since $\mathbb{E}_x k(x,x) = \psi(0) > 0$. Note that the numerator in the r.h.s. of (14) is non-negative since

$$\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x}) \leq \mathbb{E}_x \sqrt{k(x,x)} \mathbb{E}_{\tilde{x}} \sqrt{k(\tilde{x},\tilde{x})} \leq \mathbb{E}_x k(x,x)$$

with equality holding if and only if $\mathbb{P} = \delta_y$ for some $y \in \mathbb{R}^d$ (see Proposition 3). However, for any $A \in (0,1)$ and $y \in \mathbb{R}^d$, it is easy to verify that $\delta_y \notin \mathcal{P}_{k,A}$, which implies the numerator in fact positive. The denominator is clearly positive since $\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x}) \geq 0$ and therefore the r.h.s. of (14) is positive. Also note that

$$\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x}) = \int\int \psi(x-y)\, \mathrm{d}\mathbb{P}(x)\, \mathrm{d}\mathbb{P}(y) \overset{(*)}{=} \int |\phi_{\mathbb{P}}(\omega)|^2 \psi^\wedge(\omega)\, \mathrm{d}\omega$$

$$\leq \sup_{\omega \in \mathbb{R}^d} \psi^\wedge(\omega) \|\phi_{\mathbb{P}}\|_{L_2}^2 \leq (2\pi)^{-d/2} \|\psi\|_{L_1} \|\phi_{\mathbb{P}}\|_{L_2}^2, \tag{15}$$

where $\psi^\wedge$ is the Fourier transform of ψ and $(*)$ follows—see (16) in the proof of Proposition 5 in Sriperumbudur et al. (2011)—by invoking Bochner's theorem (Wendland, 2005, Theorem 6.6), which states that ψ is Fourier transform of a non-negative finite Borel measure with density $(2\pi)^{-d/2}\psi^\wedge$, *i.e.*, $\psi(x) = (2\pi)^{-d/2} \int e^{-ix^\top \omega} \psi^\wedge(\omega)\, \mathrm{d}\omega$, $x \in \mathbb{R}^d$. As $\mathbb{E}_x k(x,x) = \psi(0)$, we have that

$$\frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)} \leq \frac{A\|\phi_{\mathbb{P}}\|_{L^2}^2}{A_\psi}$$

and therefore for any $\mathbb{P} \in \mathcal{P}_{k,A}$, $\frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)} \leq A$. Using this in (14) and combining it with (13) yields the result. ∎

Remark 6 *(i) Theorem 5 shows that for any $\mathbb{P} \in \mathcal{P}_{k,A}$, it is possible to construct a shrinkage estimator that dominates the empirical estimator,* i.e., *the shrinkage estimator has a strictly smaller risk than that of the empirical estimator.*

(ii) Suppose that $\mathbb{P}$ has a density, denoted by p, with respect to the Lebesgue measure and $\phi_{\mathbb{P}} \in L^2(\mathbb{R}^d)$. By Plancherel's theorem, $p \in L^2(\mathbb{R}^d)$ as $\|p\|_{L_2} = \|\phi_{\mathbb{P}}\|_{L_2}$, which means that $\mathcal{P}_{k,A}$ includes distributions with square integrable densities (note that in general not every p is square integrable). Since

$$\|\phi_{\mathbb{P}}\|_{L_2}^2 = \int |\phi_{\mathbb{P}}(\omega)|^2\, \mathrm{d}\omega \leq \sup_{\omega \in \mathbb{R}^d} |\phi_{\mathbb{P}}(\omega)| \int |\phi_{\mathbb{P}}(\omega)|\, \mathrm{d}\omega = \|\phi_{\mathbb{P}}\|_{L_1},$$

where we used the fact that $\sup_{\omega \in \mathbb{R}^d} |\phi_{\mathbb{P}}(\omega)| = 1$, it is easy to check that

$$\left\{ \mathbb{P} \in M_+^1(\mathbb{R}^d) : \|\phi_{\mathbb{P}}\|_{L^1} \leq \frac{A(2\pi)^{d/2}\psi(0)}{\|\psi\|_{L_1}} \right\} \subset \mathcal{P}_{k,A}.$$

This means bounded densities belong to $\mathcal{P}_{k,A}$ as $\phi_{\mathbb{P}} \in L^1(\mathbb{R}^d)$ implies that $\mathbb{P}$ has a density, $p \in C_0(\mathbb{R}^d)$. Moreover, it is easy to check that larger the value of A, larger is the class $\mathcal{P}_{k,A}$ and smaller is the range of α for which $\Delta_\alpha < \Delta$ and vice-versa.

In the following, we present some concrete examples to elucidate Theorem 5.

Example 1 (Gaussian kernel and Gaussian distribution) *Define*

$$\mathcal{N} \triangleq \left\{ \mathbb{P} \in M_+^1(\mathbb{R}^d) \,\middle|\, \mathrm{d}\mathbb{P}(x) = \frac{1}{(2\pi\sigma^2)^{d/2}} e^{-\frac{\|x-\theta\|_2^2}{2\sigma^2}}\, \mathrm{d}x,\ \theta \in \mathbb{R}^d, \sigma > 0 \right\},$$

where $\psi(x) = e^{-\|x\|_2^2/2\tau^2}$, $x \in \mathbb{R}^d$ and $\tau > 0$. For $\mathbb{P} \in \mathcal{N}$, it is easy to verify that

$$\phi_{\mathbb{P}}(\omega) = e^{\sqrt{-1}\theta^\top \omega - \frac{1}{2}\sigma^2\|\omega\|_2^2},\ \omega \in \mathbb{R}^d \ \text{ and } \ \|\phi_{\mathbb{P}}\|_{L_2}^2 = \int e^{-\sigma^2\|\omega\|_2^2}\, \mathrm{d}\omega = (\pi/\sigma^2)^{d/2}.$$

Also, $\|\psi\|_{L_1} = (2\pi\tau^2)^{d/2}$. Therefore, for $\mathcal{P}_{k,A} \triangleq \{\mathbb{P} \in \mathcal{N} : \sigma^2 \geq \pi\tau^2/A^{2/d}\}$, assuming $f^* = 0$, we obtain the result in Theorem 5, i.e., *the result in Theorem 5 holds for all Gaussian distributions that are smoother (having larger variance) than that of the kernel.*

Example 2 (Linear kernel) *Suppose $f^* = 0$ and $k(x,y) = x^\top y$. While the setting of Theorem 5 does not fit this choice of k, an inspection of its proof shows that it is possible to construct a shrinkage estimator that improves upon $\mu_{\mathbb{P}}$ for an appropriate class of distributions. To this end, let ϑ and Σ represent the mean vector and covariance matrix of a distribution $\mathbb{P}$ defined on $\mathbb{R}^d$. Then it is easy to check that $\frac{\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x)} = \frac{\|\vartheta\|_2^2}{\mathrm{trace}(\Sigma)+\|\vartheta\|_2^2}$ and therefore for a given $A \in (0,1)$, define*

$$\mathcal{P}_{k,A} \triangleq \left\{ \mathbb{P} \in M_+^1(\mathbb{R}^d) \,\middle|\, \frac{\|\vartheta\|_2^2}{\mathrm{trace}(\Sigma)} \leq \frac{A}{1-A} \right\}.$$

*From (13) and (14), it is clear that for any $\mathbb{P} \in \mathcal{P}_{k,A}$, $\Delta_\alpha < \Delta$ if $\alpha \in \left(0, \frac{2(1-A)}{1+(n-1)A}\right]$. Note that this choice of kernel yields the setting similar to classical James-Stein estimation. In James-Stein estimation, $\mathbb{P} \in \mathcal{N}$ (see Example 1 for the definition of $\mathcal{N}$) and ϑ is estimated as $(1-\tilde{\alpha})\hat{\vartheta}$—which improves upon $\hat{\vartheta}$—where $\tilde{\alpha}$ depends on the sample $(x_i)_{i=1}^n$ and $\hat{\vartheta}$ is the sample mean. In our case, for all $\mathbb{P} \in \mathcal{P}_{k,A} = \left\{\mathbb{P} \in \mathcal{N} : \|\vartheta\|_2 \leq \sigma\sqrt{\frac{dA}{1-A}}\right\}$, $\Delta_\alpha < \Delta$ if $\alpha \in \left(0, \frac{2(1-A)}{1+(n-1)A}\right]$. In addition, in contrast to the James-stein estimator which improves upon the empirical estimator (*i.e.*, sample mean) for only $d \geq 3$, we note here that the proposed estimator improves for any d as long as $\mathbb{P} \in \mathcal{P}_{k,A}$. On the other hand, the proposed estimator requires some knowledge about the distribution (particularly a bound on $\|\vartheta\|_2$), which the James-Stein estimator does not (see Section 2.5 for more details).*

2.4 Data-Dependent Shrinkage Parameter

The discussion so far showed that the shrinkage estimator in (8) performs better than the empirical estimator if the data generating distribution satisfies a certain mild condition (see Theorem 5; Examples 1 and 2). However, since this condition is usually not checkable in practice, the shrinkage estimator lacks applicability. In this section, we present a completely data driven shrinkage estimator by estimating the shrinkage parameter α from data so that the estimator does not require any knowledge of the data generating distribution.

Since the maximal difference between Δ_α and Δ occurs at α_* (see Theorem 1), given an i.i.d. sample $X = \{x_1, x_2, \ldots, x_n\}$ from $\mathbb{P}$, we propose to estimate μ using $\hat{\mu}_{\tilde{\alpha}} = (1-\tilde{\alpha})\hat{\mu}$ (*i.e.*, assuming $f^* = 0$) where $\tilde{\alpha}$ is an estimator of $\alpha_* = \Delta/(\Delta + \|\mu\|_{\mathcal{H}}^2)$ given by

$$\tilde{\alpha} = \frac{\hat{\Delta}}{\hat{\Delta} + \|\hat{\mu}\|_{\mathcal{H}}^2}, \tag{16}$$

with $\hat{\Delta}$ and $\hat{\mu}$ being the empirical versions of Δ and μ, respectively (see Theorem 7 for precise definitions). The following result shows that $\tilde{\alpha}$ is a $n\sqrt{n}$-consistent estimator of α_* and $\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}$ concentrates around $\|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}$. In addition, we show that

$$\Delta_{\alpha_*} \leq \Delta_{\tilde{\alpha}} \leq \Delta_{\alpha_*} + O(n^{-3/2}) \text{ as } n \to \infty,$$

which means the performance of $\hat{\mu}_{\tilde{\alpha}}$ is similar to that of the best estimator (in mean squared sense) of the form $\hat{\mu}_\alpha$. In what follows, we will call the estimator $\hat{\mu}_{\tilde{\alpha}}$ an *empirical-bound kernel mean shrinkage estimator (B-KMSE).*

Theorem 7 *Suppose $n \geq 2$ and $f^* = 0$. Let k be a continuous kernel on a separable topological space $\mathcal{X}$ satisfying $\int_{\mathcal{X}} k(x,x)\,\mathrm{d}\mathbb{P}(x) < \infty$. Define*

$$\hat{\Delta} \triangleq \frac{\hat{\mathbb{E}}k(x,x) - \hat{\mathbb{E}}k(x,\tilde{x})}{n} \quad \text{and} \quad \|\hat{\mu}\|_{\mathcal{H}}^2 \triangleq \frac{1}{n^2}\sum_{i,j=1}^n k(x_i,x_j)$$

where $\hat{\mathbb{E}}k(x,x) \triangleq \frac{1}{n}\sum_{i=1}^n k(x_i,x_i)$ and $\hat{\mathbb{E}}k(x,\tilde{x}) \triangleq \frac{1}{n(n-1)}\sum_{i\neq j}^n k(x_i,x_j)$ are the empirical estimators of $\mathbb{E}_x k(x,x)$ and $\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})$ respectively. Assume there exist finite constants $\kappa_1 > 0$, $\kappa_2 > 0$, $\sigma_1 > 0$ and $\sigma_2 > 0$ such that

$$\mathbb{E}\|k(\cdot,x) - \mu\|_{\mathcal{H}}^m \leq \frac{m!}{2}\sigma_1^2\kappa_1^{m-2}, \quad \forall\, m \geq 2. \tag{17}$$

and

$$\mathbb{E}|k(x,x) - \mathbb{E}_x k(x,x)|^m \leq \frac{m!}{2}\sigma_2^2\kappa_2^{m-2}, \quad \forall\, m \geq 2. \tag{18}$$

Then

$$|\tilde{\alpha} - \alpha_*| = O_{\mathbb{P}}(n^{-3/2}) \text{ and } \left|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}\right| = O_{\mathbb{P}}(n^{-3/2})$$

as $n \to \infty$. In particular,

$$\min_\alpha \mathbb{E}\|\hat{\mu}_\alpha - \mu\|_{\mathcal{H}}^2 \leq \mathbb{E}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 \leq \min_\alpha \mathbb{E}\|\hat{\mu}_\alpha - \mu\|_{\mathcal{H}}^2 + O(n^{-3/2}) \tag{19}$$

as $n \to \infty$.

Proof See Section 5.2. ∎

Remark 8 *(i) $\hat{\mu}_{\tilde{\alpha}}$ is a $\sqrt{n}$-consistent estimator of μ. This follows from*

$$\begin{aligned}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} &\leq \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}} + O_{\mathbb{P}}(n^{-3/2}) \\ &\leq (1-\alpha_*)\|\hat{\mu} - \mu\|_{\mathcal{H}} + \alpha_*\|\mu\|_{\mathcal{H}} + O_{\mathbb{P}}(n^{-3/2})\end{aligned}$$

with

$$\alpha_* = \frac{\Delta}{\Delta + \|\mu\|_{\mathcal{H}}^2} = \frac{\mathbb{E}_x k(x,x) - \mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})}{\mathbb{E}_x k(x,x) + (n-1)\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x})} = O(n^{-1})$$

as $n \to \infty$. Using (38), we obtain $\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} = O_{\mathbb{P}}(n^{-1/2})$ as $n \to \infty$, which implies that $\hat{\mu}_{\tilde{\alpha}}$ is a $\sqrt{n}$-consistent estimator of μ.

(ii) Equation (19) shows that $\Delta_{\tilde{\alpha}} \leq \Delta_{\alpha_*} + O(n^{-3/2})$ where $\Delta_{\alpha_*} < \Delta$ (see Theorem 1) and therefore for any $\mathbb{P}$ satisfying (17) and (18), $\Delta_{\tilde{\alpha}} < \Delta + O(n^{-3/2})$ as $n \to \infty$.

(iii) Suppose the kernel is bounded, i.e., *$\sup_{x,y\in\mathcal{X}} |k(x,y)| \leq \kappa < \infty$. Then it is easy to verify that (17) and (18) hold with $\sigma_1 = \sqrt{\kappa}$, $\kappa_1 = 2\sqrt{\kappa}$, $\sigma_2 = \kappa$ and $\kappa_2 = 2\kappa$ and therefore the claims in Theorem 7 hold for bounded kernels.*

(iv) For $k(x,y) = x^\top y$, we have

$$\mathbb{E}\|k(\cdot,x) - \mu\|_{\mathcal{H}}^m = \mathbb{E}\left(\|k(\cdot,x) - \mu\|_{\mathcal{H}}^2\right)^{m/2} = \mathbb{E}\left(\|x - \mathbb{E}_x x\|_2^2\right)^{m/2} = \mathbb{E}\|x - \mathbb{E}_x x\|_2^m$$

and

$$\mathbb{E}|k(x,x) - \mathbb{E}_x k(x,x)|^m = \mathbb{E}|\|x\|_2^2 - \mathbb{E}_x\|x\|_2^2|^m.$$

The conditions in (17) *and* (18) *hold for* $\mathbb{P} \in \mathcal{N}$ *where* $\mathcal{N}$ *is defined in Example 1. With* $\mathbb{P} \in \mathcal{N}$ *and* $k(x,y) = x^\top y$*, the problem of estimating* μ *reduces to estimating* θ*, for which we have presented a James-Stein-like estimator,* $\hat{\mu}_{\tilde{\alpha}}$ *that satisfies the oracle inequality in* (19)*.*

(v) While the moment conditions in (17) *and* (18) *are obviously satisfied by bounded kernels, for unbounded kernels, these conditions are quite stringent as they require all the higher moments to exist. These conditions can be weakened and the proof of Theorem 7 can be carried out using Chebyshev inequality instead of Bernstein's inequality but at the cost of a slow rate in* (19)*.*

2.5 Connection to James-Stein Estimator

In this section, we explore the connection of our proposed estimator in (8) to the James-Stein estimator. Recall that Stein's setting deals with estimating the mean of the Gaussian distribution $\mathcal{N}(\theta, \sigma^2\mathbf{I}_d)$, which can be viewed as a special case of kernel mean estimation when we restrict to the class of distributions $\mathcal{P} \triangleq \{\mathcal{N}(\theta, \sigma^2\mathbf{I}_d) \,|\, \theta \in \mathbb{R}^d\}$ and a linear kernel $k(x,y) = x^\top y$, $x, y \in \mathbb{R}^d$ (see Example 2). In this case, it is easy to verify that $\Delta = d\sigma^2/n$ and $\Delta_\alpha < \Delta$ for

$$\alpha \in \left(0, \frac{2d\sigma^2}{d\sigma^2 + n\|\theta\|^2}\right).$$

Let us assume that $n = 1$, in which case, we obtain $\Delta_\alpha < \Delta$ for $\alpha \in \left(0, \frac{2d\sigma^2}{\mathbb{E}_x\|x\|^2}\right)$ as $\mathbb{E}_x\|x\|^2 = \|\theta\|^2 + d\sigma^2$. Note that the choice of α is dependent on $\mathbb{P}$ through $\mathbb{E}_x\|x\|^2$ which is not known in practice. To this end, we replace it with the empirical version $\|x\|^2$ that depends only on the sample x. For an arbitrary constant $c \in (0, 2d)$, the shrinkage estimator (assuming $f^* = 0$) can thus be written as

$$\hat{\mu}_\alpha = (1-\alpha)\hat{\mu} = \left(1 - \frac{c\sigma^2}{\|x\|^2}\right)x = x - \frac{c\sigma^2 x}{\|x\|^2},$$

which is exactly the James-Stein estimator in (3). This particular way of estimating the shrinkage parameter α has an intriguing consequence, as shown in Stein's seminal works (Stein, 1955; James and Stein, 1961), that the shrinkage estimator $\hat{\mu}_\alpha$ can be shown to dominate the maximum likelihood estimator $\hat{\mu}$ *uniformly* over all θ.

While it is compelling to see that there is seemingly a fundamental principle underlying both these settings, this connection also reveals crucial difference between our approach and classical setting of Stein—notably, original James-Stein estimator improves upon the sample mean even when α is data-dependent (see $\hat{\mu}_\alpha$ above), however, with the crucial assumption that x is normally distributed.

3. Kernel Mean Estimation as Regression Problem

In Section 2, we have shown that James-Stein-like shrinkage estimator, *i.e.*, Equation (8), improves upon the empirical estimator in estimating the kernel mean. In this section,

we provide a regression perspective to shrinkage estimation. The starting point of the connection between regression and shrinkage estimation is the observation that the kernel mean $\mu_{\mathbb{P}}$ and its empirical estimate $\hat{\mu}_{\mathbb{P}}$ can be obtained as minimizers of the following risk functionals,

$$\mathcal{E}(g) \triangleq \int_{\mathcal{X}} \|k(\cdot,x) - g\|_{\mathcal{H}}^2 \, \mathrm{d}\mathbb{P}(x) \;\text{ and }\; \widehat{\mathcal{E}}(g) \triangleq \frac{1}{n}\sum_{i=1}^n \|k(\cdot,x_i) - g\|_{\mathcal{H}}^2,$$

respectively (Kim and Scott, 2012). Given these formulations, it is natural to ask if minimizing the regularized version of $\widehat{\mathcal{E}}(g)$ will give a "better" estimator. While this question is interesting, it has to be noted that in principle, there is really no need to consider a regularized formulation as the problem of minimizing $\widehat{\mathcal{E}}$ is not ill-posed, unlike in function estimation or regression problems. To investigate this question, we consider the minimization of the following regularized empirical risk functional,

$$\widehat{\mathcal{E}}_\lambda(g) \triangleq \widehat{\mathcal{E}}(g) + \lambda\Omega(\|g\|_{\mathcal{H}}) = \frac{1}{n}\sum_{i=1}^n \|k(\cdot,x_i) - g\|_{\mathcal{H}}^2 + \lambda\Omega(\|g\|_{\mathcal{H}}), \tag{20}$$

where $\Omega : \mathbb{R}_+ \to \mathbb{R}_+$ denotes a monotonically increasing function and $\lambda > 0$ is the regularization parameter. By representer theorem (Schölkopf et al., 2001), any function $g \in \mathcal{H}$ that is a minimizer of (20) lies in a subspace spanned by $\{k(\cdot,x_1), \ldots, k(\cdot,x_n)\}$, *i.e.*, $g = \sum_{j=1}^n \beta_j k(\cdot,x_j)$ for some $\boldsymbol{\beta} \triangleq [\beta_1, \ldots, \beta_n]^\top \in \mathbb{R}^n$. Hence, by setting $\Omega(\|g\|_{\mathcal{H}}) = \|g\|_{\mathcal{H}}^2$, we can rewrite (20) in terms of $\boldsymbol{\beta}$ as

$$\widehat{\mathcal{E}}(g) + \lambda\Omega(\|g\|_{\mathcal{H}}) = \boldsymbol{\beta}^\top\mathbf{K}\boldsymbol{\beta} - 2\boldsymbol{\beta}^\top\mathbf{K}\mathbf{1}_n + \lambda\boldsymbol{\beta}^\top\mathbf{K}\boldsymbol{\beta} + c, \tag{21}$$

where $\mathbf{K}$ is an $n \times n$ Gram matrix such that $\mathbf{K}_{ij} = k(x_i,x_j)$, c is a constant that does not depend on $\boldsymbol{\beta}$, and $\mathbf{1}_n = [1/n, 1/n, \ldots, 1/n]^\top$. Differentiating (21) with respect to $\boldsymbol{\beta}$ and setting it to zero yields an optimal weight vector $\boldsymbol{\beta} = \left(\frac{1}{1+\lambda}\right)\mathbf{1}_n$ and so the minimizer of (20) is given by

$$\hat{\mu}_\lambda = \frac{1}{1+\lambda}\hat{\mu} = \left(1 - \frac{\lambda}{1+\lambda}\right)\hat{\mu} \triangleq (1-\alpha)\hat{\mu}, \tag{22}$$

which is nothing but the shrinkage estimator in (8) with $\alpha = \frac{\lambda}{1+\lambda}$ and $f^* = 0$. This provides a nice relation between shrinkage estimation and regularized risk minimization, wherein the regularization helps in shrinking the estimator $\hat{\mu}$ towards zero although it is not required from the point of view of ill-posedness. In particular, since $0 < 1 - \alpha < 1$, $\hat{\mu}_\lambda$ corresponds to a *positive-part* estimator proposed in Section 2.2.1 when $f^* = 0$.

Note that $\hat{\mu}_\lambda$ is a consistent estimator of μ as $\lambda \to 0$ and $n \to \infty$, which follows from

$$\|\hat{\mu}_\lambda - \mu\|_{\mathcal{H}} \leq \frac{1}{1+\lambda}\|\hat{\mu} - \mu\|_{\mathcal{H}} + \frac{\lambda}{1+\lambda}\|\mu\|_{\mathcal{H}} \leq O_{\mathbb{P}}(n^{-1/2}) + O(\lambda).$$

In particular $\lambda = \tau n^{-1/2}$ (for some constant $\tau > 0$) yields the slowest possible rate for $\lambda \to 0$ such that the best possible rate of $n^{-1/2}$ is obtained for $\|\hat{\mu}_\lambda - \mu\|_{\mathcal{H}} \to 0$ as $n \to \infty$. In addition, following the idea in Theorem 5, it is easy to show that $\mathbb{E}\|\hat{\mu}_\lambda - \mu\|_{\mathcal{H}}^2 < \Delta$ if

$\tau \in \left(0, \frac{2\sqrt{n}\Delta}{\|\mu\|^2_{\mathcal{H}} - \Delta}\right)$. Note that $\hat{\mu}_\lambda$ is not useful in practice as λ is not known *a priori*. However, by choosing

$$\lambda = \frac{\hat{\Delta}}{\|\hat{\mu}\|^2_{\mathcal{H}}},$$

it is easy to verify (see Theorem 7 and Remark 8) that

$$\mathbb{E}\|\hat{\mu}_\lambda - \mu\|^2_{\mathcal{H}} < \mathbb{E}\|\hat{\mu} - \mu\|^2_{\mathcal{H}} + O(n^{-3/2}) \tag{23}$$

as $n \to \infty$. Owing to the connection of $\hat{\mu}_\lambda$ to a regression problem, in the following, we present an alternate data-dependent choice of λ obtained from leave-one-out cross validation (LOOCV) that also satisfies (23), and we refer to the corresponding estimator as *regularized kernel mean shrinkage estimator (R-KMSE)*.

To this end, for a given shrinkage parameter λ, denote by $\hat{\mu}^{(-i)}_\lambda$ as the kernel mean estimated from $\{x_j\}_{j=1}^n \backslash \{x_i\}$. We will measure the quality of $\hat{\mu}^{(-i)}_\lambda$ by how well it approximates $k(\cdot, x_i)$ with the overall quality being quantified by the cross-validation score,

$$LOOCV(\lambda) = \frac{1}{n}\sum_{i=1}^n \left\| k(\cdot, x_i) - \hat{\mu}^{(-i)}_\lambda \right\|^2_{\mathcal{H}}. \tag{24}$$

The LOOCV formulation in (24) differs from the one used in regression, wherein instead of measuring the deviation of the prediction made by the function on the omitted observation, we measure the deviation between the feature map of the omitted observation and the function itself. The following result shows that the shrinkage parameter in $\hat{\mu}_\lambda$ (see (22)) can be obtained analytically by minimizing (24) and requires $O(n^2)$ operations to compute.

Proposition 9 *Let $n \geq 2$, $\rho := \frac{1}{n^2}\sum_{i,j=1}^n k(x_i, x_j)$ and $\varrho := \frac{1}{n}\sum_{i=1}^n k(x_i, x_i)$. Assuming $n\rho > \varrho$, the unique minimizer of $LOOCV(\lambda)$ is given by*

$$\lambda_r = \frac{n(\varrho - \rho)}{(n-1)(n\rho - \varrho)}. \tag{25}$$

Proof See Section 5.3. ∎

It is instructive to compare

$$\alpha_r = \frac{\lambda_r}{\lambda_r + 1} = \frac{\varrho - \rho}{(n-2)\rho + \varrho/n} \tag{26}$$

to the one in (16), where the latter can be shown to be $\frac{\varrho - \rho}{\varrho + (n-2)\rho}$, by noting that $\hat{\mathbb{E}}k(x,x) = \varrho$ and $\hat{\mathbb{E}}k(x,\tilde{x}) = \frac{n\rho - \varrho}{n-1}$ (in Theorem 7, we employ the U-statistic estimator of $\mathbb{E}_{x,\tilde{x}}k(x,\tilde{x})$, whereas ρ in Proposition 9 can be seen as a V-statistic counterpart). This means α_r obtained from LOOCV will be relatively larger than the one obtained from (16). Like in Theorem 7, the requirement that $n \geq 2$ in Theorem 9 stems from the fact that at least two data points are needed to evaluate the LOOCV score. Note that $n\rho > \varrho$ if and only if $\hat{\mathbb{E}}k(x,\tilde{x}) > 0$, which is guaranteed if the kernel is positive valued. We refer to $\hat{\mu}_{\lambda_r}$ as R-KMSE, whose $\sqrt{n}$-consistency is established by the following result, which also shows that $\hat{\mu}_{\lambda_r}$ satisfies (23).

Theorem 10 *Let $n \geq 2$, $n\rho > \varrho$ where ρ and ϱ are defined in Proposition 9 and k satisfies the assumptions in Theorem 7. Then $\|\hat{\mu}_{\lambda_r} - \mu\|_{\mathcal{H}} = O_{\mathbb{P}}(n^{-1/2})$,*

$$\min_\alpha \mathbb{E}\|\hat{\mu}_\alpha - \mu\|^2_{\mathcal{H}} \leq \mathbb{E}\|\hat{\mu}_{\lambda_r} - \mu\|^2_{\mathcal{H}} \leq \min_\alpha \mathbb{E}\|\hat{\mu}_\alpha - \mu\|^2_{\mathcal{H}} + O(n^{-3/2}) \tag{27}$$

where $\hat{\mu}_\alpha = (1-\alpha)\hat{\mu}$ and therefore

$$\mathbb{E}\|\hat{\mu}_{\lambda_r} - \mu\|^2_{\mathcal{H}} < \mathbb{E}\|\hat{\mu} - \mu\|^2_{\mathcal{H}} + O(n^{-3/2}) \tag{28}$$

as $n \to \infty$.

Proof See Section 5.4. ∎

4. Spectral Shrinkage Estimators

Consider the following regularized risk minimization problem

$$\arg\inf_{\mathbf{F} \in \mathcal{H}\otimes\mathcal{H}} \quad \mathbb{E}_{x\sim\mathbb{P}} \|k(x,\cdot) - \mathbf{F}[k(x,\cdot)]\|^2_{\mathcal{H}} + \lambda\|\mathbf{F}\|^2_{\mathrm{HS}}, \tag{29}$$

where the minimization is carried over the space of Hilbert-Schmidt operators, $\mathbf{F}$ on $\mathcal{H}$ with $\|\mathbf{F}\|_{\mathrm{HS}}$ being the Hilbert-Schmidt norm of $\mathbf{F}$. As an interpretation, we are finding a smooth operator $\mathbf{F}$ that maps $k(x,\cdot)$ to itself (see Grünewälder et al. (2013) for more details on this smooth operator framework). It is not difficult to show that the solution to (29) is given by $\mathbf{F} = \Sigma_{XX}(\Sigma_{XX} + \lambda I)^{-1}$ where $\Sigma_{XX} = \int k(\cdot.x) \otimes k(\cdot, x)\, \mathrm{d}\mathbb{P}(x)$ is a covariance operator defined on $\mathcal{H}$ (see, e.g., Grünewälder et al., 2012). Note that Σ_{XX} is a Bochner integral, which is well-defined as a Hilbert-Schmidt operator if $\mathcal{X}$ is a separable topological space and k is a continuous kernel satisfying $\int k(x,x)\, \mathrm{d}\mathbb{P}(x) < \infty$. Consequently, let us define

$$\mu_\lambda = \mathbf{F}\mu = \Sigma_{XX}(\Sigma_{XX} + \lambda I)^{-1}\mu,$$

which is an approximation to μ as it can be shown that $\|\mu_\lambda - \mu\|_{\mathcal{H}} \to 0$ as $\lambda \to 0$ (see the proof of Theorem 13). Given an i.i.d. sample $x_1, \ldots, x_n$ from $\mathbb{P}$, the empirical counterpart of (29) is given by

$$\arg\min_{\mathbf{F} \in \mathcal{H}\otimes\mathcal{H}} \quad \frac{1}{n}\sum_{i=1}^n \|k(x_i,\cdot) - \mathbf{F}[k(x_i,\cdot)]\|^2_{\mathcal{H}} + \lambda\|\mathbf{F}\|^2_{\mathrm{HS}} \tag{30}$$

resulting in

$$\check{\mu}_\lambda \triangleq \mathbf{F}\hat{\mu} = \hat{\Sigma}_{XX}(\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\mu} \tag{31}$$

where $\hat{\Sigma}_{XX}$ is the empirical covariance operator on $\mathcal{H}$ given by

$$\hat{\Sigma}_{XX} = \frac{1}{n}\sum_{i=1}^n k(\cdot.x_i) \otimes k(\cdot, x_i).$$

Unlike $\hat{\mu}_\lambda$ in (22), $\check{\mu}_\lambda$ shrinks $\hat{\mu}$ differently in each coordinate by taking the eigenspectrum of $\hat{\Sigma}_{XX}$ into account (see Proposition 11) and so we refer to it as the *spectral kernel mean shrinkage estimator (S-KMSE)*.

Proposition 11 *Let $\{(\gamma_i, \phi_i)\}_{i=1}^n$ be eigenvalue and eigenfunction pairs of $\hat{\Sigma}_{XX}$. Then*

$$\check{\mu}_\lambda = \sum_{i=1}^{n} \frac{\gamma_i}{\gamma_i + \lambda} \langle \hat{\mu}, \phi_i \rangle_{\mathcal{H}} \phi_i.$$

Proof Since $\hat{\Sigma}_{XX}$ is a finite rank operator, it is compact. Since it is also a self-adjoint operator on $\mathcal{H}$, by Hilbert-Schmidt theorem (Reed and Simon, 1972, Theorems VI.16, VI.17), we have $\hat{\Sigma}_{XX} = \sum_{i=1}^n \gamma_i \langle \phi_i, \cdot \rangle_{\mathcal{H}} \phi_i$. The result follows by using this in (31). ■

As shown in Proposition 11, the effect of S-KMSE is to reduce the contribution of high frequency components of $\hat{\mu}$ (*i.e.*, contribution of $\hat{\mu}$ along the directions corresponding to smaller γ_i) when $\hat{\mu}$ is expanded in terms of the eigenfunctions of the empirical covariance operator, which are nothing but the kernel PCA basis (Rasmussen and Williams, 2006, Section 4.3). This means, similar to R-KMSE, S-KMSE also shrinks $\hat{\mu}$ towards zero, however, the difference being that while R-KMSE shrinks equally in all coordinates, S-KMSE controls the amount of shrinkage by the information contained in each coordinate. In particular, S-KMSE takes into account more information about the kernel by allowing for different amount of shrinkage in each coordinate direction according to the value of γ_i, wherein the shrinkage is small in the coordinates whose γ_i are large. Moreover, Proposition 11 reveals that the effect of shrinkage is akin to *spectral filtering* (Bauer et al., 2007)—which in our case corresponds to Tikhonov regularization—wherein S-KMSE filters out the high-frequency components of the spectral representation of the kernel mean. Muandet et al. (2014b) leverages this observation and generalizes S-KMSE to a family of shrinkage estimators via spectral filtering algorithms.

The following result presents an alternate representation for $\check{\mu}_\lambda$, using which we relate the smooth operator formulation in (30) to the regularization formulation in (20).

Proposition 12 *Let $\Phi : \mathbb{R}^n \to \mathcal{H}$, $\mathbf{a} \mapsto \sum_{i=1}^n a_i k(\cdot, x_i)$ where $\mathbf{a} \triangleq (a_1, \ldots, a_n)$. Then*

$$\check{\mu}_\lambda = \hat{\Sigma}_{XX} (\hat{\Sigma}_{XX} + \lambda I)^{-1} \hat{\mu} = \Phi (\mathbf{K} + n\lambda \mathbf{I})^{-1} \mathbf{K} \mathbf{1}_n,$$

where $\mathbf{K}$ is the Gram matrix, I is an identity operator on $\mathcal{H}$, $\mathbf{I}$ is an $n \times n$ identity matrix and $\mathbf{1}_n \triangleq [1/n, \ldots, 1/n]^\top$.

Proof See Section 5.5. ■

From Proposition 12, it is clear that

$$\check{\mu}_\lambda = \frac{1}{\sqrt{n}} \sum_{j=1}^{n} (\boldsymbol{\beta}_s)_j k(\cdot, x_j) \tag{32}$$

where $\boldsymbol{\beta}_s \triangleq \sqrt{n} (\mathbf{K} + n\lambda \mathbf{I})^{-1} \mathbf{K} \mathbf{1}_n$. Given the form of $\check{\mu}_\lambda$ in (32), it is easy to verify that $\boldsymbol{\beta}_s$ is the minimizer of (20) when $\widehat{\mathcal{E}}_\lambda$ is minimized over $\{g = \frac{1}{\sqrt{n}} \sum_{j=1}^n (\boldsymbol{\beta})_j k(\cdot, x_j) : \boldsymbol{\beta} \in \mathbb{R}^n\}$ with $\Omega(\|g\|_{\mathcal{H}}) \triangleq \|\boldsymbol{\beta}\|_2^2$.

The following result, discussed in Remark 14, establishes the consistency and convergence rate of S-KMSE, $\check{\mu}_\lambda$.

Theorem 13 *Suppose $\mathcal{X}$ is a separable topological space and k is a continuous, bounded kernel on $\mathcal{X}$. Then the following hold.*

(i) If $\mu \in \overline{\mathcal{R}(\Sigma_{XX})}$, then $\|\check{\mu}_\lambda - \mu\|_{\mathcal{H}} \to 0$ as $\lambda\sqrt{n} \to \infty$, $\lambda \to 0$ and $n \to \infty$.

(ii) If $\mu \in \mathcal{R}(\Sigma_{XX})$, then $\|\check{\mu}_\lambda - \mu\|_{\mathcal{H}} = O_{\mathbb{P}}(n^{-1/2})$ for $\lambda = cn^{-1/2}$ with $c > 0$ being a constant independent of n.

Proof See Section 5.6. ■

Remark 14 *While Theorem 13(i) shows that S-KMSE, $\check{\mu}_\lambda$ is not universally consistent,* i.e., *S-KMSE is not consistent for all $\mathbb{P}$ but only for those $\mathbb{P}$ that satisfies $\mu \in \overline{\mathcal{R}(\Sigma_{XX})}$, under some additional conditions on the kernel, the universal consistency of S-KMSE can be guaranteed. This is achieved by assuming that constant functions are included in $\mathcal{H}$,* i.e., *$1 \in \mathcal{H}$. Note that if $1 \in \mathcal{H}$, then it is easy to check that there exists $g \in \mathcal{H}$ (choose $g = 1$) such that $\mu = \Sigma_{XX} g = \int k(\cdot, x) g(x)\, \mathrm{d}\mathbb{P}(x)$,* i.e., *$\mu \in \mathcal{R}(\Sigma_{XX})$, and, therefore, by Theorem 13, $\check{\mu}_\lambda$ is not only universally consistent but also achieves a rate of $n^{-1/2}$. Choosing $k(x, y) = \tilde{k}(x, y) + b$, $x, y \in \mathcal{X}$, $b > 0$ where $\tilde{k}$ is any bounded, continuous positive definite kernel ensures that $1 \in \mathcal{H}$.*

Note that the estimator $\check{\mu}_\lambda$ requires the knowledge of the shrinkage or regularization parameter, λ. Similar to R-KMSE, below, we present a data dependent approach to select λ using leave-one-out cross validation. While the shrinkage parameter for R-KMSE can be obtained in a simple closed form (see Proposition 9), we will see below that finding the corresponding parameter for S-KMSE is more involved. Evaluating the score function (*i.e.*, (24)) naïvely requires one to solve for $\hat{\mu}_\lambda^{(-i)}$ explicitly for every i, which is computationally expensive. The following result provides an alternate expression for the score, which can be evaluated efficiently. We would like to point out that a variation of Proposition 15 already appeared in Muandet et al. (2014a, Theorem 4). However, Theorem 4 in Muandet et al. (2014a) uses an inappropriate choice of $\hat{\mu}_\lambda^{(-i)}$, which we fixed in the following result.

Proposition 15 *The LOOCV score of S-KMSE is given by*

$$LOOCV(\lambda) = \frac{1}{n} \mathrm{tr}\left((\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{K} (\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{A}_\lambda \right) - \frac{2}{n} \mathrm{tr}\left((\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{B}_\lambda \right) + \frac{1}{n} \sum_{i=1}^{n} k(x_i, x_i),$$

where $\lambda_n \triangleq (n-1)\lambda$, $\mathbf{A}_\lambda \triangleq \frac{1}{(n-1)^2} \sum_{i=1}^n \mathbf{c}_{i,\lambda} \mathbf{c}_{i,\lambda}^\top$, $\mathbf{B}_\lambda \triangleq \frac{1}{n-1} \sum_{i=1}^n \mathbf{c}_{i,\lambda} \mathbf{k}_i^\top$, $d_{i,\lambda} \triangleq \mathbf{k}_i^\top (\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{e}_i$,

$$\mathbf{c}_{i,\lambda} \triangleq \mathbf{K}\mathbf{1} - \mathbf{k}_i - \mathbf{e}_i \mathbf{k}_i^\top \mathbf{1} + \mathbf{e}_i k(x_i, x_i) + \frac{\mathbf{e}_i \mathbf{k}_i^\top (\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{K}\mathbf{1}}{1 - d_{i,\lambda}} - \frac{\mathbf{e}_i \mathbf{k}_i^\top (\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{k}_i}{1 - d_{i,\lambda}} - \frac{\mathbf{e}_i \mathbf{k}_i^\top (\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{e}_i \mathbf{k}_i^\top \mathbf{1}}{1 - d_{i,\lambda}} + \frac{\mathbf{e}_i \mathbf{k}_i^\top (\mathbf{K} + \lambda_n \mathbf{I})^{-1} \mathbf{e}_i k(x_i, x_i)}{1 - d_{i,\lambda}},$$

$\mathbf{k}_i$ is the i^{th} column of $\mathbf{K}$, $\mathbf{1} \triangleq (1, \ldots, 1)^\top$ and $\mathbf{e}_i \triangleq (0, 0, \ldots, 1, \ldots, 0)^\top$ with 1 being in the i^{th} position. Here $\mathrm{tr}(\mathbf{A})$ denotes the trace of a square matrix $\mathbf{A}$.

Proof See Section 5.7. ■

Unlike R-KMSE, a closed form expression for the minimizer of $LOOCV(\lambda)$ in Proposition 15 is not possible and so proving the consistency of S-KMSE along with results similar to those in Theorem 10 are highly non-trivial. Hence, we are not able to provide any theoretical comparison of $\tilde{\mu}_\lambda$ (with λ being chosen as a minimizer of $LOOCV(\lambda)$ in Proposition 15) with $\hat{\mu}$. However, in the next section, we provide an empirical comparison through simulations where we show that the S-KMSE outperforms the empirical estimator.

5. Proofs

In this section, we present the missing proofs of the results of Sections 2–4.

5.1 Proof of Proposition 3

($\Rightarrow$) If $\mathbb{P} = \delta_x$ for some $x \in \mathcal{X}$, then $\hat{\mu} = \mu = k(\cdot, x)$ and thus $\Delta = 0$.
($\Leftarrow$) Suppose $\Delta = 0$. It follows from (7) that $\iint (k(x,x) - k(x,y))\, \mathrm{d}\mathbb{P}(x)\, \mathrm{d}\mathbb{P}(y) = 0$. Since k is translation invariant, this reduces to

$$\iint (\psi(0) - \psi(x-y))\, \mathrm{d}\mathbb{P}(x)\, \mathrm{d}\mathbb{P}(y) = 0.$$

By invoking Bochner's theorem (Wendland, 2005, Theorem 6.6), which states that ψ is the Fourier transform of a non-negative finite Borel measure Λ, *i.e.*, $\psi(x) = \int e^{-ix^\top \omega}\, \mathrm{d}\Lambda(\omega)$, $x \in \mathbb{R}^d$, we obtain (see (16) in the proof of Proposition 5 in Sriperumbudur et al. (2011))

$$\iint \psi(x-y)\, \mathrm{d}\mathbb{P}(x)\, \mathrm{d}\mathbb{P}(y) = \int |\phi_{\mathbb{P}}(\omega)|^2\, \mathrm{d}\Lambda(\omega),$$

thereby yielding

$$\int (|\phi_{\mathbb{P}}(\omega)|^2 - 1)\, \mathrm{d}\Lambda(\omega) = 0, \tag{33}$$

where $\phi_{\mathbb{P}}$ is the characteristic function of $\mathbb{P}$. Note that $\phi_{\mathbb{P}}$ is uniformly continuous and $|\phi_{\mathbb{P}}| \leq 1$. Since k is characteristic, Theorem 9 in Sriperumbudur et al. (2010) implies that $\mathrm{supp}(\Lambda) = \mathbb{R}^d$, using which in (33) yields $|\phi_{\mathbb{P}}(\omega)| = 1$ for all $\omega \in \mathbb{R}^d$. Since $\phi_{\mathbb{P}}$ is positive definite on $\mathbb{R}^d$, it follows from Sasvári (2013, Lemma 1.5.1) that $\phi_{\mathbb{P}}(\omega) = e^{\sqrt{-1}\omega^\top x}$ for some $x \in \mathbb{R}^d$ and thus $\mathbb{P} = \delta_x$.

5.2 Proof of Theorem 7

Before we prove Theorem 7, we present Bernstein's inequality in separable Hilbert spaces, quoted from Yurinsky (1995, Theorem 3.3.4), which will be used to prove Theorem 7.

Theorem 16 (Bernstein's inequality) *Let $(\Omega, \mathcal{A}, P)$ be a probability space, H be a separable Hilbert space, $B > 0$ and $\theta > 0$. Furthermore, let $\xi_1, \ldots, \xi_n : \Omega \to H$ be zero mean independent random variables satisfying*

$$\sum_{i=1}^n \mathbb{E}\|\xi_i\|_H^m \leq \frac{m!}{2}\theta^2 B^{m-2}. \tag{34}$$

Then for any $\tau > 0$,

$$P^n \left\{ (\xi_1, \ldots, \xi_n) : \left\| \sum_{i=1}^n \xi_i \right\|_H \geq 2B\tau + \sqrt{2\theta^2\tau} \right\} \leq 2e^{-\tau}.$$

Proof (of Theorem 7) Consider

$$\begin{aligned}
\tilde{\alpha} - \alpha_* &= \frac{\hat{\Delta}}{\hat{\Delta} + \|\hat{\mu}\|_{\mathcal{H}}^2} - \frac{\Delta}{\Delta + \|\mu\|_{\mathcal{H}}^2} = \frac{\hat{\Delta}\|\mu\|_{\mathcal{H}}^2 - \Delta\|\hat{\mu}\|_{\mathcal{H}}^2}{(\hat{\Delta} + \|\hat{\mu}\|_{\mathcal{H}}^2)(\Delta + \|\mu\|_{\mathcal{H}}^2)} \\
&= \frac{\hat{\Delta}(\|\mu\|_{\mathcal{H}}^2 - \|\hat{\mu}\|_{\mathcal{H}}^2)}{(\Delta + \|\mu\|_{\mathcal{H}}^2)(\hat{\Delta} + \|\hat{\mu}\|_{\mathcal{H}}^2)} + \frac{(\hat{\Delta} - \Delta)\|\hat{\mu}\|_{\mathcal{H}}^2}{(\Delta + \|\mu\|_{\mathcal{H}}^2)(\hat{\Delta} + \|\hat{\mu}\|_{\mathcal{H}}^2)} \\
&= \frac{\tilde{\alpha}(\|\mu\|_{\mathcal{H}}^2 - \|\hat{\mu}\|_{\mathcal{H}}^2)}{(\Delta + \|\mu\|_{\mathcal{H}}^2)} + \frac{(\hat{\Delta} - \Delta)(1 - \tilde{\alpha})}{(\Delta + \|\mu\|_{\mathcal{H}}^2)}.
\end{aligned}$$

Rearranging $\tilde{\alpha}$, we obtain

$$\tilde{\alpha} - \alpha_* = \frac{\alpha_*(\|\mu\|_{\mathcal{H}}^2 - \|\hat{\mu}\|_{\mathcal{H}}^2) + (1 - \alpha_*)(\hat{\Delta} - \Delta)}{\hat{\Delta} + \|\hat{\mu}\|_{\mathcal{H}}^2}.$$

Therefore,

$$|\tilde{\alpha} - \alpha_*| \leq \frac{\alpha_* |\|\mu\|_{\mathcal{H}}^2 - \|\hat{\mu}\|_{\mathcal{H}}^2| + (1 + \alpha_*)|\hat{\Delta} - \Delta|}{(\Delta + \|\mu\|_{\mathcal{H}}^2) - (\|\mu\|_{\mathcal{H}}^2 - \|\hat{\mu}\|_{\mathcal{H}}^2) + (\hat{\Delta} - \Delta)}, \tag{35}$$

where it is easy to verify that

$$|\hat{\Delta} - \Delta| \leq \frac{|\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x}) - \hat{\mathbb{E}} k(x,\tilde{x})|}{n} + \frac{|\hat{\mathbb{E}} k(x,x) - \mathbb{E}_x k(x,x)|}{n}. \tag{36}$$

In the following we obtain bounds on $|\hat{\mathbb{E}} k(x,x) - \mathbb{E}_x k(x,x)|$, $|\mathbb{E}_{x,\tilde{x}} k(x,\tilde{x}) - \hat{\mathbb{E}} k(x,\tilde{x})|$ and $|\|\mu\|_{\mathcal{H}}^2 - \|\hat{\mu}\|_{\mathcal{H}}^2|$ when the kernel satisfies (17) and (18).

Bound on $|\hat{\mathbb{E}} k(x,x) - \mathbb{E}_x k(x,x)|$:

Since k is a continuous kernel on a separable topological space $\mathcal{X}$, it follows from Lemma 4.33 of Steinwart and Christmann (2008) that $\mathcal{H}$ is separable. By defining $\xi_i \triangleq k(x_i, x_i) - \mathbb{E}_x k(x.x)$, it follows from (18) that $\theta = \sqrt{n}\sigma_2$ and $B = \kappa_2$ and so by Theorem 16, for any $\tau > 0$, with probability at least $1 - 2e^{-\tau}$,

$$|\hat{\mathbb{E}} k(x,x) - \mathbb{E}_x k(x,x)| \leq \sqrt{\frac{2\sigma_2^2\tau}{n}} + \frac{2\kappa_2\tau}{n}. \tag{37}$$

Bound on $\|\hat{\mu} - \mu\|_{\mathcal{H}}$:

By defining $\xi_i \triangleq k(\cdot, x_i) - \mu$ and using (17), we have $\theta = \sqrt{n}\sigma_1$ and $B = \kappa_1$. Therefore, by Theorem 16, for any $\tau > 0$, with probability at least $1 - 2e^{-\tau}$,

$$\|\hat{\mu} - \mu\|_{\mathcal{H}} \leq \sqrt{\frac{2\sigma_1^2\tau}{n}} + \frac{2\kappa_1\tau}{n}. \tag{38}$$

Bound on $|\|\hat{\mu}\|_{\mathcal{H}}^2 - \|\mu\|_{\mathcal{H}}^2|$:

Since

$$\left|\|\hat{\mu}\|_{\mathcal{H}}^2 - \|\mu\|_{\mathcal{H}}^2\right| \leq (\|\hat{\mu}\|_{\mathcal{H}} + \|\mu\|_{\mathcal{H}})\|\hat{\mu} - \mu\|_{\mathcal{H}} \leq (\|\hat{\mu} - \mu\|_{\mathcal{H}} + 2\|\mu\|_{\mathcal{H}})\|\hat{\mu} - \mu\|_{\mathcal{H}},$$

it follows from (38) that for any $\tau > 0$, with probability at least $1 - 2e^{-\tau}$,

$$\left|\|\hat{\mu}\|_{\mathcal{H}}^2 - \|\mu\|_{\mathcal{H}}^2\right| \leq D_1\sqrt{\frac{\tau}{n}} + D_2\left(\frac{\tau}{n}\right) + D_3\left(\frac{\tau}{n}\right)^{3/2} + D_4\left(\frac{\tau}{n}\right)^2, \tag{39}$$

where $(D_i)_{i=1}^4$ are positive constants that depend only on σ_1^2, κ and $\|\mu\|_{\mathcal{H}}$, and not on n and τ.

Bound on $|\hat{\mathbb{E}}k(x,\tilde{x}) - \mathbb{E}_{x,\tilde{x}}k(x,\tilde{x})|$:
Since

$$\hat{\mathbb{E}}k(x,\tilde{x}) - \mathbb{E}_{x,\tilde{x}}k(x,\tilde{x}) = \frac{n^2(\|\hat{\mu}\|_{\mathcal{H}}^2 - \|\mu\|_{\mathcal{H}}^2) + n(\mathbb{E}_x k(x,x) - \hat{\mathbb{E}}k(x,x)) + n(\|\mu\|_{\mathcal{H}}^2 - \mathbb{E}_x k(x,x))}{n(n-1)},$$

it follows from (37) and (39) that for any $\tau > 0$, with probability at least $1 - 4e^{-\tau}$,

$$\begin{aligned}|\hat{\mathbb{E}}k(x,\tilde{x}) - \mathbb{E}_{x,\tilde{x}}k(x,\tilde{x})| &\leq F_1\sqrt{\frac{\tau}{n}} + F_2\left(\frac{\tau}{n}\right) + F_3\left(\frac{\tau}{n}\right)^{3/2} + F_4\left(\frac{\tau}{n}\right)^2 + \frac{F_5}{n} \\ &\leq F_1'\sqrt{\frac{1+\tau}{n}} + F_2'\left(\frac{1+\tau}{n}\right) + F_3'\left(\frac{1+\tau}{n}\right)^{3/2} + F_4'\left(\frac{1+\tau}{n}\right)^2 \end{aligned} \tag{40}$$

where $(F_i)_{i=1}^5$ and $(F_i')_{i=1}^4$ are positive constants that do not depend on n and τ.

Bound on $|\tilde{\alpha} - \alpha_*|$:
Using (37) and (40) in (36), for any $\tau > 0$, with probability at least $1 - 4e^{-\tau}$,

$$|\hat{\Delta} - \Delta| \leq \frac{F_1''}{n}\sqrt{\frac{1+\tau}{n}} + \frac{F_2''}{n}\left(\frac{1+\tau}{n}\right) + \frac{F_3''}{n}\left(\frac{1+\tau}{n}\right)^{3/2} + \frac{F_4''}{n}\left(\frac{1+\tau}{n}\right)^2,$$

using which in (35) along with (39), we obtain that for any $\tau > 0$, with probability at least $1 - 4e^{-\tau}$,

$$|\tilde{\alpha} - \alpha_*| \leq \frac{\sum_{i=1}^4 \left(G_{i1}\alpha_* + \frac{G_{i2}}{n}(1+\alpha_*)\right)\left(\frac{1+\tau}{n}\right)^{i/2}}{\left|\theta_n - \sum_{i=1}^4 \left(G_{i1} + \frac{G_{i2}}{n}\right)\left(\frac{1+\tau}{n}\right)^{i/2}\right|}, \tag{41}$$

where $\theta_n \triangleq \Delta + \|\mu\|_{\mathcal{H}}^2$ and $(G_{i1})_{i=1}^4$, $(G_{i2})_{i=1}^4$ are positive constants that do not depend on n and τ. Since $\alpha_* = \frac{\Delta}{\Delta + \|\mu\|_{\mathcal{H}}^2} = \frac{\mathbb{E}_x k(x,x) - \mathbb{E}_{x,\tilde{x}}k(x,\tilde{x})}{\mathbb{E}_x k(x,x) + (n-1)\mathbb{E}_{x,\tilde{x}}k(x,\tilde{x})} = O(n^{-1})$ and $\theta_n = \frac{\mathbb{E}_x k(x,x) + (n-1)\|\mu\|_{\mathcal{H}}^2}{n} = O(1)$ as $n \to \infty$, it follows from (41) that $|\tilde{\alpha} - \alpha_*| = O_{\mathbb{P}}(n^{-3/2})$ as $n \to \infty$.

Bound on $|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}|$:
Using (38) and (41) in

$$|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| \leq \|\hat{\mu}_{\tilde{\alpha}} - \hat{\mu}_{\alpha_*}\|_{\mathcal{H}} \leq |\tilde{\alpha} - \alpha_*|\|\hat{\mu} - \mu\|_{\mathcal{H}} + |\tilde{\alpha} - \alpha_*|\|\mu\|_{\mathcal{H}},$$

for any $\tau > 0$, with probability at least $1 - 4e^{-\tau}$, we have

$$|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| \leq \frac{\sum_{i=1}^6 \left(G_{i1}'\alpha_* + \frac{G_{i2}'}{n}(1+\alpha_*)\right)\left(\frac{1+\tau}{n}\right)^{i/2}}{\left|\theta_n - \sum_{i=1}^4 \left(G_{i1} + \frac{G_{i2}}{n}\right)\left(\frac{1+\tau}{n}\right)^{i/2}\right|}, \tag{42}$$

where $(G_{i1}')_{i=1}^6$ and $(G_{i2}')_{i=1}^6$ are positive constants that do not depend on n and τ. From (42), it is easy to see that $|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| = O_{\mathbb{P}}(n^{-3/2})$ as $n \to \infty$.

Bound on $\mathbb{E}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \mathbb{E}\|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2$:
Since

$$\begin{aligned}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2 &\leq (\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} + \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}})\,|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| \\ &\leq 2(\|\hat{\mu}\|_{\mathcal{H}} + \|\mu\|_{\mathcal{H}})\,|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| \\ &\leq 2(\|\hat{\mu} - \mu\|_{\mathcal{H}} + 2\|\mu\|_{\mathcal{H}})\,|\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}|\,,\end{aligned}$$

for any $\tau > 0$, with probability at least $1 - 4e^{-\tau}$,

$$\begin{aligned}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2 &\leq \frac{\sum_{i=1}^8 \left(G_{i1}''\alpha_* + \frac{G_{i2}''}{n}(1+\alpha_*)\right)\left(\frac{1+\tau}{n}\right)^{i/2}}{\left|\theta_n - \sum_{i=1}^4 \left(G_{i1} + \frac{G_{i2}}{n}\right)\left(\frac{1+\tau}{n}\right)^{i/2}\right|}, \\ &\leq \frac{\sum_{i=1}^8 \left(G_{i1}''\alpha_* + \frac{G_{i2}''}{n}(1+\alpha_*)\right)\left(\frac{1+\tau}{n}\right)^{i/2}}{\left|\theta_n - \sum_{i=1}^4 \left(G_{i1} + \frac{G_{i2}}{n}\right)\left(\frac{1}{n}\right)^{i/2}\right|}, \\ &\leq \begin{cases} \frac{\gamma_n}{\phi_n}\sqrt{\frac{1+\tau}{n}}, & 0 < \tau \leq n-1 \\ \frac{\gamma_n}{\phi_n}\left(\frac{1+\tau}{n}\right)^4, & \tau \geq n-1 \end{cases},\end{aligned}$$

where $\gamma_n \triangleq H_1\alpha_* + \frac{H_2}{n}(1+\alpha_*)$, $\phi_n \triangleq \left|\theta_n - \sum_{i=1}^4 \left(G_{i1} + \frac{G_{i2}}{n}\right)\left(\frac{1}{n}\right)^{i/2}\right|$ and $(H_i)_{i=1}^2$ are positive constants that do not depend on n and τ. In other words,

$$\mathbb{P}\left(\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2 > \epsilon\right) \leq \begin{cases} 4\exp\left(1 - n\left(\frac{\epsilon\phi_n}{\gamma_n}\right)^2\right), & \frac{\gamma_n}{\phi_n\sqrt{n}} \leq \epsilon \leq \frac{\gamma_n}{\phi_n} \\ 4\exp\left(1 - n\left(\frac{\epsilon\phi_n}{\gamma_n}\right)^{1/4}\right), & \epsilon \geq \frac{\gamma_n}{\phi_n} \end{cases}.$$

Therefore,

$$\begin{aligned}\mathbb{E}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \mathbb{E}\|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2 &= \int_0^\infty \mathbb{P}\left(\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2 > \epsilon\right)\,\mathrm{d}\epsilon \\ &\leq \frac{\gamma_n}{\phi_n\sqrt{n}} + 4\int_{\frac{\gamma_n}{\phi_n\sqrt{n}}}^{\frac{\gamma_n}{\phi_n}} \exp\left(1 - n\left(\frac{\epsilon\phi_n}{\gamma_n}\right)^2\right)\,\mathrm{d}\epsilon \\ &\quad + 4\int_{\frac{\gamma_n}{\phi_n}}^\infty \exp\left(1 - n\left(\frac{\epsilon\phi_n}{\gamma_n}\right)^{1/4}\right)\,\mathrm{d}\epsilon \\ &= \frac{\gamma_n}{\phi_n\sqrt{n}} + \frac{2\gamma_n}{\phi_n\sqrt{n}}\int_0^{n-1} \frac{e^{-t}}{\sqrt{t+1}}\,\mathrm{d}t + \frac{16e\gamma_n}{n^4\phi_n}\int_n^\infty t^3 e^{-t}\,\mathrm{d}t.\end{aligned}$$

Since $\int_0^{n-1} \frac{e^{-t}}{\sqrt{t+1}}\,\mathrm{d}t \leq \int_0^\infty e^{-t}\,\mathrm{d}t = 1$ and $\int_n^\infty t^3 e^{-t}\,\mathrm{d}t \leq \int_0^\infty t^3 e^{-t}\,\mathrm{d}t = 6$, we have

$$\mathbb{E}\|\hat{\mu}_{\tilde{\alpha}} - \mu\|_{\mathcal{H}}^2 - \mathbb{E}\|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}^2 \leq \frac{3\gamma_n}{\phi_n\sqrt{n}} + \frac{96e\gamma_n}{n^4\phi_n}.$$

The claim in (19) follows by noting that $\gamma_n = O(n^{-1})$ and $\phi_n = O(1)$ as $n \to \infty$. ■

5.3 Proof of Proposition 9

Define $\alpha \triangleq \frac{\lambda}{\lambda+1}$ and $\phi(x_i) \triangleq k(\cdot, x_i)$. Note that

$$\begin{aligned}
LOOCV(\lambda) &\triangleq \frac{1}{n}\sum_{i=1}^n \left\| \frac{(1-\alpha)}{n-1}\sum_{j\neq i}\phi(x_j) - \phi(x_i)\right\|_{\mathcal{H}}^2 \\
&= \frac{1}{n}\sum_{i=1}^n \left\| \frac{n(1-\alpha)}{n-1}\hat{\mu} - \frac{1-\alpha}{n-1}\phi(x_i) - \phi(x_i)\right\|_{\mathcal{H}}^2 \\
&= \left\|\frac{n(1-\alpha)}{n-1}\hat{\mu}\right\|_{\mathcal{H}}^2 - \frac{2}{n}\left\langle \sum_{i=1}^n \frac{n-\alpha}{n-1}\phi(x_i), \frac{n(1-\alpha)}{n-1}\hat{\mu}\right\rangle_{\mathcal{H}} + \frac{1}{n}\sum_{i=1}^n \left\|\frac{n-\alpha}{n-1}\phi(x_i)\right\|_{\mathcal{H}}^2 \\
&= \left(\frac{n^2(1-\alpha)^2}{(n-1)^2} - \frac{2n(n-\alpha)(1-\alpha)}{(n-1)^2}\right)\|\hat{\mu}\|_{\mathcal{H}}^2 + \frac{(n-\alpha)^2}{n(n-1)^2}\sum_{i=1}^n k(x_i,x_i) \\
&= \frac{1}{(n-1)^2}\left\{\alpha^2(n^2\rho - 2n\rho + \varrho) + 2n\alpha(\rho-\varrho) + n^2(\varrho-\rho)\right\} \triangleq \frac{F(\alpha)}{(n-1)^2}.
\end{aligned}$$

Since $\frac{d}{d\lambda}LOOCV(\lambda) = (n-1)^{-2}\frac{d}{d\alpha}F(\alpha)\frac{d\alpha}{d\lambda} = (n-1)^{-2}(1+\lambda)^{-2}\frac{d}{d\alpha}F(\alpha)$, equating it zero yields (25). It is easy to show that the second derivative of $LOOCV(\lambda)$ is positive implying that $LOOCV(\lambda)$ is strictly convex and so λ_r is unique.

5.4 Proof of Theorem 10

Since $\hat{\mu}_{\lambda_r} = \frac{\hat{\mu}}{1+\lambda_r} = (1-\alpha_r)\hat{\mu}$, we have $\|\hat{\mu}_{\lambda_r} - \mu\|_{\mathcal{H}} \leq \alpha_r\|\hat{\mu}\|_{\mathcal{H}} + \|\hat{\mu} - \mu\|_{\mathcal{H}}$. Note that

$$\alpha_r = \frac{n(\varrho-\rho)}{n(n-2)\rho + \varrho} = \frac{n\hat{\Delta}}{\hat{\Delta} + (n-1)\|\hat{\mu}\|_{\mathcal{H}}^2} = \frac{\hat{\mathbb{E}}k(x,x) - \hat{\mathbb{E}}k(x,\tilde{x})}{\hat{\mathbb{E}}k(x,x) + (n-2)\hat{\mathbb{E}}k(x,\tilde{x})},$$

where $\hat{\Delta}$, $\|\hat{\mu}\|_{\mathcal{H}}^2$, $\hat{\mathbb{E}}k(x,x)$ and $\hat{\mathbb{E}}k(x,\tilde{x})$ are defined in Theorem 7. Consider $|\alpha_r - \alpha_*| \leq |\alpha_r - \tilde{\alpha}| + |\tilde{\alpha} - \alpha_*|$ where $\tilde{\alpha}$ is defined in (16). From Theorem 7, we have $|\tilde{\alpha} - \alpha_*| = O_{\mathbb{P}}(n^{-3/2})$ as $n \to \infty$ and

$$\begin{aligned}
\alpha_r - \tilde{\alpha} &= \frac{\hat{\mathbb{E}}k(x,x) - \hat{\mathbb{E}}k(x,\tilde{x})}{\hat{\mathbb{E}}k(x,x) + (n-2)\hat{\mathbb{E}}k(x,\tilde{x})} - \frac{\hat{\mathbb{E}}k(x,x) - \hat{\mathbb{E}}k(x,\tilde{x})}{2\hat{\mathbb{E}}k(x,x) + (n-2)\hat{\mathbb{E}}k(x,\tilde{x})} \\
&= \frac{\tilde{\alpha}\hat{\mathbb{E}}k(x,x)}{\hat{\mathbb{E}}k(x,x) + (n-2)\hat{\mathbb{E}}k(x,\tilde{x})} = (\tilde{\alpha} - \alpha_*)\beta + \alpha_*\beta,
\end{aligned}$$

where $\beta \triangleq \frac{\hat{\mathbb{E}}k(x,x)}{\hat{\mathbb{E}}k(x,x) + (n-2)\hat{\mathbb{E}}k(x,\tilde{x})}$. Therefore, $|\alpha_r - \tilde{\alpha}| \leq |\tilde{\alpha} - \alpha_*||\beta| + \alpha_*|\beta|$, where $\alpha_* = O(n^{-1})$ as $n \to \infty$, which follows from Remark 8(i). Since $|\hat{\mathbb{E}}k(x,x) - \mathbb{E}_x k(x,x)| = O_{\mathbb{P}}(n^{-1/2})$ and

$|\hat{\mathbb{E}}k(x,\tilde{x}) - \mathbb{E}_{x,\tilde{x}}k(x,\tilde{x})| = O_{\mathbb{P}}(n^{-1/2})$, which follow from (37) and (40) respectively, we have $|\beta| = O_{\mathbb{P}}(n^{-1})$ as $n \to \infty$. Combining the above, we have $|\alpha_r - \tilde{\alpha}| = O_{\mathbb{P}}(n^{-2})$, thereby yielding $|\alpha_r - \alpha_*| = O_{\mathbb{P}}(n^{-3/2})$. Proceeding as in Theorem 7, we have

$$|\|\hat{\mu}_{\lambda_r} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| \leq \|\hat{\mu}_{\lambda_r} - \mu_{\alpha_*}\|_{\mathcal{H}} \leq |\alpha_r - \alpha_*|\|\hat{\mu} - \mu\|_{\mathcal{H}} + |\alpha_r - \alpha_*|\|\mu\|_{\mathcal{H}},$$

which from the above follows that $|\|\hat{\mu}_{\lambda_r} - \mu\|_{\mathcal{H}} - \|\hat{\mu}_{\alpha_*} - \mu\|_{\mathcal{H}}| = O_{\mathbb{P}}(n^{-3/2})$ as $n \to \infty$. By arguing as in Remark 8(i), it is easy to show that $\hat{\mu}_{\lambda_r}$ is a $\sqrt{n}$-consistent estimator of μ. (27) follows by carrying out the analysis as in the proof of Theorem 7 verbatim by replacing $\tilde{\alpha}$ with α_r, while (28) follows by appealing to Remark 8(ii).

5.5 Proof of Proposition 12

First note that for any $i \in \{1, \ldots, n\}$,

$$\hat{\Sigma}_{XX}k(\cdot, x_i) = \frac{1}{n}\sum_{j=1}^n k(\cdot, x_j)k(x_i, x_j) = \frac{1}{n}\boldsymbol{\Phi}\mathbf{k}_i^\top$$

with $\mathbf{k}_i$ being the i^{th} row of $\mathbf{K}$. This implies for any $\mathbf{a} \in \mathbb{R}^n$,

$$\hat{\Sigma}_{XX}\boldsymbol{\Phi}\mathbf{a} = \hat{\Sigma}_{XX}\left(\sum_{i=1}^n a_i k(\cdot, x_i)\right) \overset{(*)}{=} \sum_{i=1}^n a_i \hat{\Sigma}_{XX}k(\cdot, x_i) = \frac{1}{n}\sum_{i=1}^n a_i \boldsymbol{\Phi}\mathbf{k}_i^\top,$$

where $(*)$ holds since $\hat{\Sigma}_{XX}$ is a linear operator. Also, since Φ is a linear operator, we obtain

$$\hat{\Sigma}_{XX}\boldsymbol{\Phi}\mathbf{a} = \frac{1}{n}\boldsymbol{\Phi}\left(\sum_{i=1}^n a_i \mathbf{k}_i^\top\right) = \frac{1}{n}\boldsymbol{\Phi}\mathbf{K}\mathbf{a}. \tag{43}$$

To prove the result, let us define $\mathbf{a} \triangleq (\mathbf{K} + n\lambda\mathbf{I})^{-1}\mathbf{K}\mathbf{1}_n$ and consider

$$(\hat{\Sigma}_{XX} + \lambda I)\boldsymbol{\Phi}\mathbf{a} \overset{(43)}{=} n^{-1}\boldsymbol{\Phi}\mathbf{K}\mathbf{a} + \lambda\boldsymbol{\Phi}\mathbf{a} = \boldsymbol{\Phi}(n^{-1}\mathbf{K} + \lambda\mathbf{I})\mathbf{a} = \frac{1}{n}\boldsymbol{\Phi}\mathbf{K}\mathbf{1}_n \overset{(43)}{=} \hat{\Sigma}_{XX}\boldsymbol{\Phi}\mathbf{1}_n = \hat{\Sigma}_{XX}\hat{\mu}.$$

Multiplying to the left on both sides of the above equation by $(\hat{\Sigma}_{XX} + \lambda I)^{-1}$, we obtain $\boldsymbol{\Phi}(\mathbf{K} + n\lambda\mathbf{I})^{-1}\mathbf{K}\mathbf{1}_n = (\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}\hat{\mu}$ and the result follows by noting that $(\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX} = \hat{\Sigma}_{XX}(\hat{\Sigma}_{XX} + \lambda I)^{-1}$.

5.6 Proof of Theorem 13

By Proposition 12, we have $\check{\mu}_\lambda = (\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}\hat{\mu}$. Define $\mu_\lambda \triangleq (\Sigma_{XX} + \lambda I)^{-1}\Sigma_{XX}\mu$. Let us consider the decomposition $\check{\mu}_\lambda - \mu = (\check{\mu}_\lambda - \mu_\lambda) + (\mu_\lambda - \mu)$ with

$$\begin{aligned}
\check{\mu}_\lambda - \mu_\lambda &= (\hat{\Sigma}_{XX} + \lambda I)^{-1}(\hat{\Sigma}_{XX}\hat{\mu} - \hat{\Sigma}_{XX}\mu_\lambda - \lambda\mu_\lambda) \\
&\overset{(*)}{=} (\hat{\Sigma}_{XX} + \lambda I)^{-1}(\hat{\Sigma}_{XX}\hat{\mu} - \hat{\Sigma}_{XX}\mu_\lambda - \Sigma_{XX}\mu + \Sigma_{XX}\mu_\lambda) \\
&= (\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}(\hat{\mu} - \mu) - (\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}(\mu_\lambda - \mu) \\
&\qquad + (\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX}(\mu_\lambda - \mu),
\end{aligned}$$

where we used $\lambda\mu_\lambda = \Sigma_{XX}\mu - \Sigma_{XX}\mu_\lambda$ in $(*)$. By defining $\mathcal{A}(\lambda) \triangleq \|\mu_\lambda - \mu\|_{\mathcal{H}}$, we have

$$\begin{aligned}\|\check{\mu}_\lambda - \mu\|_{\mathcal{H}} &\leq \|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}(\hat{\mu} - \mu)\|_{\mathcal{H}} + \|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}(\mu_\lambda - \mu)\|_{\mathcal{H}} \\ &\quad + \|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX}(\mu_\lambda - \mu)\|_{\mathcal{H}} + \mathcal{A}(\lambda) \\ &\leq \|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}\| \left(\|\hat{\mu} - \mu\|_{\mathcal{H}} + \mathcal{A}(\lambda)\right) + \|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX}\|\mathcal{A}(\lambda) \\ &\quad + \mathcal{A}(\lambda), \end{aligned} \tag{44}$$

where for any bounded linear operator B, $\|B\|$ denotes its operator norm. We now bound $\|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX}\|$ as follows. It is easy to show that

$$\begin{aligned}(\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX} &= \left(I - (\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \hat{\Sigma}_{XX})\right)^{-1}(\Sigma_{XX} + \lambda I)^{-1}\Sigma_{XX} \\ &= \left(\sum_{j=0}^{\infty}\left((\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \hat{\Sigma}_{XX})\right)^j\right)(\Sigma_{XX} + \lambda I)^{-1}\Sigma_{XX},\end{aligned}$$

where the last line denotes the Neumann series and therefore

$$\begin{aligned}\|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX}\| &\leq \sum_{j=0}^{\infty}\left\|(\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \hat{\Sigma}_{XX})\right\|^j \|(\Sigma_{XX} + \lambda I)^{-1}\Sigma_{XX}\| \\ &\leq \sum_{j=0}^{\infty}\left\|(\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \hat{\Sigma}_{XX})\right\|_{\mathrm{HS}}^j,\end{aligned}$$

where we used $\|(\Sigma_{XX} + \lambda I)^{-1}\Sigma_{XX}\| \leq 1$ and the fact that Σ_{XX} and $\hat{\Sigma}_{XX}$ are Hilbert-Schmidt operators on $\mathcal{H}$ as $\|\Sigma_{XX}\|_{\mathrm{HS}} \leq \kappa < \infty$ and $\|\hat{\Sigma}_{XX}\|_{\mathrm{HS}} \leq \kappa < \infty$ with κ being the bound on the kernel. Define $\eta : \mathcal{X} \to \mathrm{HS}(\mathcal{H})$, $\eta(x) = (\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \Sigma_x)$, where $\mathrm{HS}(\mathcal{H})$ is the space of Hilbert-Schmidt operators on $\mathcal{H}$ and $\Sigma_x \triangleq k(\cdot, x) \otimes k(\cdot, x)$. Observe that $\mathbb{E}\frac{1}{n}\sum_{i=1}^n \eta(x_i) = 0$. Also, for all $i \in \{1, \ldots, n\}$, $\|\eta(x_i)\|_{\mathrm{HS}} \leq \|(\Sigma_{XX} + \lambda I)^{-1}\|\|\Sigma_{XX} - \Sigma_x\|_{\mathrm{HS}} \leq \frac{2\kappa}{\lambda}$ and $\mathbb{E}\|\eta(x_i)\|_{\mathrm{HS}}^2 \leq \frac{4\kappa^2}{\lambda^2}$. Therefore, by Bernstein's inequality (see Theorem 16), for any $\tau > 0$, with probability at least $1 - 2e^{-\tau}$ over the choice of $\{x_i\}_{i=1}^n$,

$$\|(\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \hat{\Sigma}_{XX})\|_{\mathrm{HS}} \leq \frac{\kappa\sqrt{2\tau}}{\lambda\sqrt{n}} + \frac{2\kappa\tau}{\lambda n} \leq \frac{\kappa\sqrt{2\tau}(\sqrt{2\tau} + 1)}{\lambda\sqrt{n}}.$$

For $\lambda \geq \frac{\kappa\sqrt{8\tau}(\sqrt{2\tau}+1)}{\sqrt{n}}$, we obtain that $\|(\Sigma_{XX} + \lambda I)^{-1}(\Sigma_{XX} - \hat{\Sigma}_{XX})\|_{\mathrm{HS}} \leq \frac{1}{2}$ and therefore $\|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\Sigma_{XX}\| \leq 2$. Using this along with $\|(\hat{\Sigma}_{XX} + \lambda I)^{-1}\hat{\Sigma}_{XX}\| \leq 1$ and (38) in (44), we obtain that for any $\tau > 0$ and $\lambda \geq \frac{\kappa\sqrt{8\tau}(\sqrt{2\tau}+1)}{\sqrt{n}}$, with probability at least $1 - 2e^{-\tau}$ over the choice of $\{x_i\}_{i=1}^n$,

$$\|\check{\mu}_\lambda - \mu\|_{\mathcal{H}} \leq \frac{\sqrt{2\kappa\tau} + 4\tau\sqrt{\kappa}}{\sqrt{n}} + 4\mathcal{A}(\lambda). \tag{45}$$

We now analyze $\mathcal{A}(\lambda)$. Since k is continuous and $\mathcal{X}$ is separable, $\mathcal{H}$ is separable (Steinwart and Christmann, 2008, Lemma 4.33). Also Σ_{XX} is compact since it is Hilbert-Schmidt. The consistency result therefore follows from Sriperumbudur et al. (2013, Proposition A.2) which ensures $\mathcal{A}(\lambda) \to 0$ as $\lambda \to 0$. The rate also follows from Sriperumbudur et al. (2013, Proposition A.2) which yields $\mathcal{A}(\lambda) \leq \|\Sigma_{XX}^{-1}\mu\|_{\mathcal{H}}\lambda$, thereby obtaining $\|\check{\mu}_\lambda - \mu\|_{\mathcal{H}} = O_{\mathbb{P}}(n^{-1/2})$ for $\lambda = cn^{-1/2}$ with $c > 0$ being a constant independent of n.

5.7 Proof of Proposition 15

From Proposition 12, we have $\check{\mu}_\lambda^{(-i)} = (\hat{\Sigma}^{(-i)} + \lambda I)^{-1}\hat{\Sigma}^{(-i)}\hat{\mu}^{(-i)}$ where

$$\hat{\Sigma}^{(-i)} \triangleq \frac{1}{n-1}\sum_{j\neq i} k(\cdot, x_j) \otimes k(\cdot, x_j).$$

and $\hat{\mu}^{(-i)} \triangleq \frac{1}{n-1}\sum_{j\neq i} k(\cdot, x_j)$. Define $a \triangleq k(\cdot, x_i)$. It is easy to verify that

$$\hat{\Sigma}^{(-i)} = \frac{n}{n-1}\left(\hat{\Sigma} - \frac{a \otimes a}{n}\right) \text{ and } \hat{\mu}^{(-i)} = \frac{n}{n-1}\left(\hat{\mu} - \frac{a}{n}\right).$$

Therefore,

$$\check{\mu}_\lambda^{(-i)} = \frac{n}{n-1}\left((\hat{\Sigma} + \lambda_n' I) - \frac{a \otimes a}{n}\right)^{-1}\left(\hat{\Sigma} - \frac{a \otimes a}{n}\right)\left(\hat{\mu} - \frac{a}{n}\right),$$

which after using Sherman-Morrison formula[3] reduces to

$$\check{\mu}_\lambda^{(-i)} = \frac{n}{n-1}\left((\hat{\Sigma} + \lambda_n' I)^{-1} + \frac{(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)(\hat{\Sigma} + \lambda_n' I)^{-1}}{n - \langle a, (\hat{\Sigma} + \lambda_n' I)^{-1}a\rangle_{\mathcal{H}}}\right)\left(\hat{\Sigma} - \frac{a \otimes a}{n}\right)\left(\hat{\mu} - \frac{a}{n}\right),$$

where $\lambda_n' \triangleq \frac{n-1}{n}\lambda$. Using the notation in the proof of Proposition 12, the following can be proved:

(i) $(\hat{\Sigma} + \lambda_n' I)^{-1}\hat{\Sigma}\hat{\mu} = n^{-1}\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{K}\mathbf{1}$.

(ii) $(\hat{\Sigma} + \lambda_n' I)^{-1}\hat{\Sigma}a = \Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{k}_i$.

(iii) $(\hat{\Sigma} + \lambda_n' I)^{-1}a = n\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i$.

Based on the above, it is easy to show that

(iv) $(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)\hat{\mu} = (\hat{\Sigma} + \lambda_n' I)^{-1}a\langle a, \hat{\mu}\rangle_{\mathcal{H}} = \Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i\mathbf{k}_i^\top\mathbf{1}$.

(v) $(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)a = (\hat{\Sigma} + \lambda_n' I)^{-1}a\langle a, a\rangle_{\mathcal{H}} = n\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i k(x_i, x_i)$.

(vi) $(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)(\hat{\Sigma} + \lambda_n' I)^{-1}\hat{\Sigma}\hat{\mu} = \Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i\mathbf{k}_i^\top(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{K}\mathbf{1}$.

(vii) $(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)(\hat{\Sigma} + \lambda_n' I)^{-1}\hat{\Sigma}a = n\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i\mathbf{k}_i^\top(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{k}_i$.

(viii) $(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)\hat{\mu} = n\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i\mathbf{k}_i^\top(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i\mathbf{k}_i^\top\mathbf{1}$.

(ix) $(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)(\hat{\Sigma} + \lambda_n' I)^{-1}(a \otimes a)a = n^2\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i\mathbf{k}_i^\top(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i k(x_i, x_i)$.

(x) $\langle a, (\hat{\Sigma} + \lambda_n' I)^{-1}a\rangle_{\mathcal{H}} = n\mathbf{k}_i^\top(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{e}_i$.

Using the above in $\check{\mu}_\lambda^{(-i)}$, we obtain

$$\check{\mu}_\lambda^{(-i)} = \frac{1}{n-1}\Phi(\mathbf{K} + \lambda_n\mathbf{I})^{-1}\mathbf{c}_{i,\lambda}.$$

Substituting the above in (24) yields the result.

3. The Sherman-Morrison formula states that $(\mathbf{A} + \mathbf{u}\mathbf{v}^\top)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{u}\mathbf{v}^\top\mathbf{A}^{-1}}{1+\mathbf{v}^\top\mathbf{A}^{-1}\mathbf{u}}$ where $\mathbf{A}$ is an invertible square matrix, $\mathbf{u}$ and $\mathbf{v}$ are column vectors such that $1 + \mathbf{v}^\top\mathbf{A}^{-1}\mathbf{u} \neq 0$.

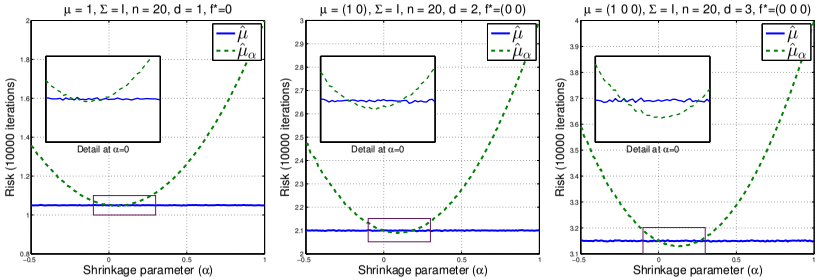


Figure 1: The comparison between standard estimator, $\hat{\mu}$ and shrinkage estimator, $\hat{\mu}_\alpha$ (with $f^* = 0$) of the mean of the Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ on $\mathbb{R}^d$ where $d = 1, 2, 3$.

6. Experiments

In this section, we empirically compare the proposed shrinkage estimators to the standard estimator of the kernel mean on both synthetic and real-world datasets. Specifically, we consider the following estimators: *i*) empirical/standard kernel mean estimator (KME), *ii*) KMSE whose parameter is obtained via empirical bound (B-KMSE), *iii*) regularized KMSE whose parameter is obtained via Proposition 9 (R-KMSE), and *iv*) spectral KMSE whose parameter is obtained via Proposition 15 (S-KMSE).

6.1 Synthetic Data

Given the true data-generating distribution $\mathbb{P}$ and the i.i.d. sample $X = \{x_1, x_2, \ldots, x_n\}$ from $\mathbb{P}$, we evaluate different estimators using the loss function

$$L(\boldsymbol{\beta}, X, \mathbb{P}) \triangleq \left\| \sum_{i=1}^{n} \beta_i k(x_i, \cdot) - \mathbb{E}_{x\sim\mathbb{P}}[k(x, \cdot)] \right\|_{\mathcal{H}}^2,$$

where $\boldsymbol{\beta}$ is the weight vector associated with different estimators. Then, we can estimate the risk of the estimator by averaging over m independent copies of X, *i.e.*, $\widehat{R} = \frac{1}{m}\sum_{j=1}^{m} L(\boldsymbol{\beta}_j, X_j, \mathbb{P})$.

To allow for an exact calculation of $L(\boldsymbol{\beta}, X, \mathbb{P})$, we consider $\mathbb{P}$ to be a mixture-of-Gaussians distribution and k being one of the following kernel functions: *i*) linear kernel $k(x, x') = x^\top x'$, *ii*) polynomial degree-2 kernel $k(x, x') = (x^\top x' + 1)^2$, *iii*) polynomial degree-3 kernel $k(x, x') = (x^\top x' + 1)^3$ and *iv*) Gaussian RBF kernel $k(x, x') = \exp\left(-\|x - x'\|^2/2\sigma^2\right)$. We refer to them as LIN, POLY2, POLY3, and RBF, respectively. The analytic forms of $\mathbb{E}_{x\sim\mathbb{P}}[k(x, \cdot)]$ for Gaussian distribution are given in Song et al. (2008) and Muandet et al. (2012). Unless otherwise stated, we set the bandwidth parameter of the Gaussian kernel as $\sigma^2 = \text{median}\left\{\|x_i - x_j\|^2 : i, j = 1, \ldots, n\right\}$, *i.e.*, the median heuristic.

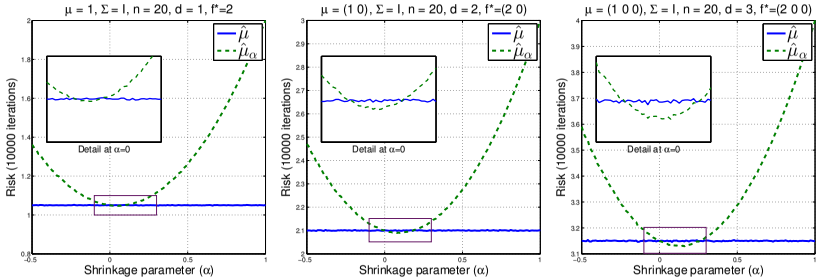


Figure 2: The risk comparison between standard estimator, $\hat{\mu}$ and shrinkage estimator, $\hat{\mu}_\alpha$ (with $f^* \in \{2, (2, 0)^\top, (2, 0, 0)^\top\}$) of the mean of the Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ on $\mathbb{R}^d$ where $d = 1, 2, 3$.

6.1.1 Gaussian Distribution

We begin our empirical studies by considering the simplest case in which the distribution $\mathbb{P}$ is a Gaussian distribution $\mathcal{N}(\mu, \mathbf{I})$ on $\mathbb{R}^d$ where $d = 1, 2, 3$ and k is a linear kernel. In this case, the problem of kernel mean estimation reduces to just estimating the mean μ of the Gaussian distribution $\mathcal{N}(\mu, \mathbf{I})$. We consider only shrinkage estimators of form $\hat{\mu}_\alpha = \alpha f^* + (1-\alpha)\hat{\mu}$. The true mean μ of the distribution is chosen to be 1, $(1, 0)^\top$, and $(1, 0, 0)^\top$, respectively. Figure 1 depicts the comparison between the standard estimator and the shrinkage estimator, $\hat{\mu}_\alpha$ when the target f^* is the origin. We can clearly see that even in this simple case, an improvement can be gained by applying a small shrinkage. Furthermore, the improvement becomes more substantial as we increase the dimensionality of the underlying space. Figure 2 illustrates similar results when $f^* \neq 0$ but $f^* \in \{2, (2, 0)^\top, (2, 0, 0)^\top\}$. Interestingly, we can still observe similar improvement, which demonstrates that the choice of target f^* can be arbitrary when no prior knowledge about $\mu_\mathbb{P}$ is available.

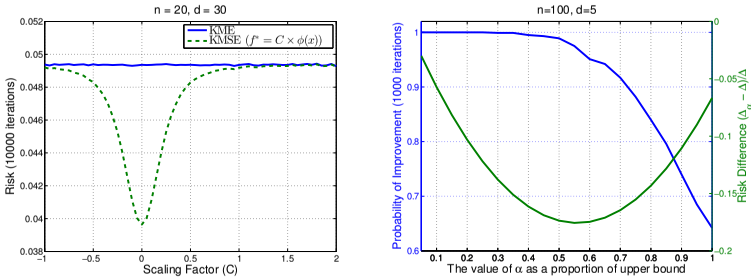
6.1.2 Mixture of Gaussians Distributions

To simulate a more realistic case, let y be a sample from $\mathbb{P} \triangleq \sum_{i=1}^{4} \pi_i \mathcal{N}(\boldsymbol{\theta}_i, \Sigma_i)$. In the following experiments, the sample x is generated from the following generative process:

$$x = y + \varepsilon, \quad \theta_{ij} \sim \mathcal{U}(-10, 10), \quad \Sigma_i \sim \mathcal{W}(2 \times \mathbf{I}_d, 7), \quad \varepsilon \sim \mathcal{N}(0, 0.2 \times \mathbf{I}_d),$$

where $\mathcal{U}(a, b)$ and $\mathcal{W}(\Sigma_0, df)$ represent the uniform distribution and Wishart distribution, respectively. We set $\boldsymbol{\pi} = (0.05, 0.3, 0.4, 0.25)^\top$. The choice of parameters here is quite arbitrary; we have experimented using various parameter settings and the results are similar to those presented here.

Figure 3(a) depicts the comparison between the standard kernel mean estimator and the shrinkage estimator, $\hat{\mu}_\alpha$ when the kernel k is the Gaussian RBF kernel. For shrinkage estimator $\hat{\mu}_\alpha$, we consider $f^* = C \times k(x, \cdot)$ where C is a scaling factor and each element of x is a realization of uniform random variable on $(0, 1)$. That is, we allow the target f^* to change



(a) (b)

Figure 3: (a) The risk comparison between $\hat{\mu}$ (KME) and $\hat{\mu}_{\tilde{\alpha}}$ (KMSE) where $\tilde{\alpha} = \hat{\Delta}/(\hat{\Delta} + \|f^* - \hat{\mu}\|^2_{\mathcal{H}})$. We consider when $f^* = C \times k(x, \cdot)$ where x is drawn uniformly from a pre-specified range and C is a scaling factor. (b) The probability of improvement and the risk difference as a function of shrinkage parameter α averaged over 1,000 iterations. As the value of α increases, we get more improvement in term of the risk, whereas the probability of improvement decreases as a function of α.

depending on the value of C. As the absolute value of C increases, the target function f^* will move further away from the origin. The shrinkage parameter α is determined using the empirical bound, *i.e.*, $\tilde{\alpha} = \hat{\Delta}/(\hat{\Delta} + \|f^* - \hat{\mu}\|^2_{\mathcal{H}})$. As we can see in Figure 3(a), the results reveal how important the choice of f^* is. That is, we may get substantial improvement over the empirical estimator if appropriate prior knowledge is incorporated through f^*, which in this case suggests that f^* should lie close to the origin. We intend to investigate the topic of prior knowledge in more detail in our future work.

Previous comparisons between standard estimator and shrinkage estimator is based entirely on the notion of a risk, which is in fact not useful in practice as we only observe a single copy of sample from the probability distribution. Instead, one should also look at the probability that, given a single copy of sample, the shrinkage estimator outperforms the standard one in term of a loss. To this end, we conduct an experiment comparing the standard estimator and shrinkage estimator using the Gaussian RBF kernel. In addition to the risk comparison, we also compare the probability that the shrinkage estimator gives smaller loss than that of the standard estimator. To be more precise, the probability is defined as a proportion of the samples drawn from the same distribution whose shrinkage loss is smaller than the loss of the standard estimator. Figure 3(b) illustrates the risk difference ($\Delta_\alpha - \Delta$) and the probability of improvement (*i.e.*, the fraction of times $\Delta_\alpha < \Delta$) as a function of shrinkage parameter α. In this case, the value of α is specified as a proportion of empirical upper bound $2\hat{\Delta}/(\hat{\Delta} + \|\hat{\mu}\|^2_{\mathcal{H}})$. The results suggest that the shrinkage parameter α controls the trade-off between the amount of improvement in terms of risk and the probability that the shrinkage estimator will improve upon the standard one. However, this trade-off only holds up to a certain value of α. As α becomes too large, both the probability of improvement and the amount of improvement itself decrease, which coincides with the intuition given for the positive-part shrinkage estimators (cf. Section 2.2.1).

6.1.3 Shrinkage Estimators via Leave-One-Out Cross-Validation

In addition to the empirical upper bound, one can alternatively compute the shrinkage parameter using leave-one-out cross-validation proposed in Section 3. Our goal here is to compare the B-KMSE, R-KMSE and S-KMSE on synthetic data when the shrinkage parameter λ is chosen via leave-one-out cross-validation procedure. Note that the only difference between B-KMSE and R-KMSE is the way we compute the shrinkage parameter.

Figure 4 shows the empirical risk of different estimators using different kernels as we increase the value of shrinkage parameter λ (note that R-KMSE and S-KMSE in Figure 4 refer to those in (22) and (31) respectively). Here we scale the shrinkage parameter by the smallest non-zero eigenvalue γ_0 of the kernel matrix $\mathbf{K}$. In general, we find that R-KMSE and S-KMSE outperforms KME. Nevertheless, as the shrinkage parameter λ becomes large, there is a tendency that the specific shrinkage estimate might actually perform worse than the KME, *e.g.*, see LIN kernel and outliers in Figure 4. The result also supports our previous observation regarding Figure 3(b), which suggests that it is very important to choose the parameter λ appropriately.

To demonstrate the leave-one-out cross-validation procedure, we conduct similar experiments in which the parameter λ is chosen by the proposed LOOCV procedure. Figure 5 depicts the percentage of improvement (with respect to the empirical risk of the KME[4]) as we vary the sample size and dimension of the data. Clearly, B-KMSE, R-KMSE and S-KMSE outperform the standard estimator. Moreover, both R-KMSE and S-KMSE tend to outperform the B-KMSE. We can also see that the performance of S-KMSE depends on the choice of kernel. This makes sense intuitively because S-KMSE also incorporates the eigen-spectrum of $\mathbf{K}$, whereas R-KMSE does not. The effects of both sample size and data dimensionality are also transparent from Figure 5. While it is intuitive to see that the improvement gets smaller with increase in sample size, it is a bit surprising to see that we can gain much more in high-dimensional input space, especially when the kernel function is non-linear, because the estimation happens in the feature space associated with the kernel function rather than in the input space. Lastly, we note that the improvement is more substantial in the "large d, small n" paradigm.

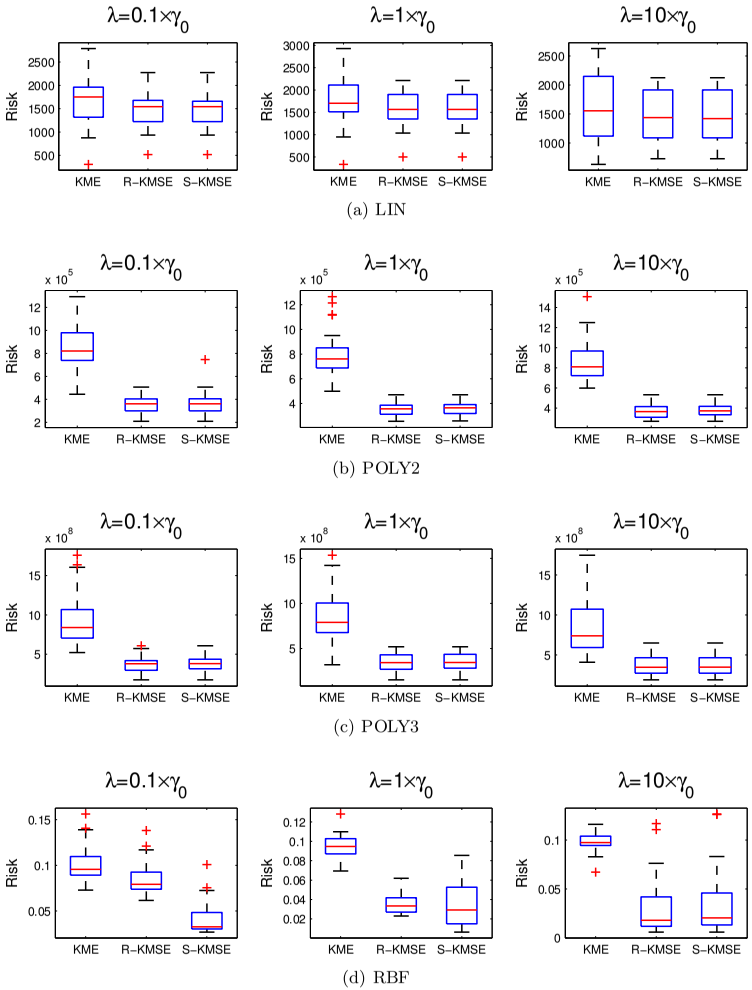
6.2 Real Data

To evaluate the proposed estimators on real-world data, we consider several benchmark applications, namely, classification via Parzen window classifier, density estimation via kernel mean matching (Song et al., 2008), and discriminative learning on distributions (Muandet et al., 2012; Muandet and Schölkopf, 2013). For some of these tasks we employ datasets from the UCI repositories. We use only real-valued features, each of which is normalized to have zero mean and unit variance.

6.2.1 Parzen Window Classifiers

One of the oldest and best-known classification algorithms is the *Parzen window classifier* (Duda et al., 2000). It is easy to implement and is one of the powerful non-linear supervised

4. If we denote the loss of KME and KMSE as ℓ_{KME} and ℓ_{KMSE}, respectively, the percentage of improvement is calculated as $100 \times (\ell_{KME} - \ell_{KMSE})/\ell_{KME}$.



(a) LIN

(b) POLY2

(c) POLY3

(d) RBF

Figure 4: The average loss of KME (left), R-KMSE (middle) and S-KMSE (right) estimators with different values of shrinkage parameter. We repeat the experiments over 30 different distributions with $n = 10$ and $d = 30$.

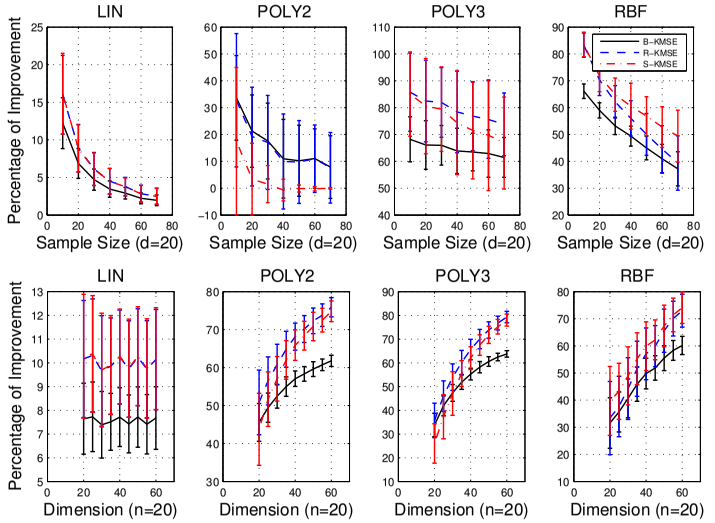


Figure 5: The percentage of improvement compared to KME over 30 different distributions of B-KMSE, R-KMSE and S-KMSE with varying sample size (n) and dimension (d). For B-KMSE, we calculate α using (16), whereas R-KMSE and S-KMSE use LOOCV to choose λ.

learning techniques. Suppose we have data points from two classes, namely, positive class and negative class. For positive class, we observe $\mathfrak{X} \triangleq \{x_1, x_2, \ldots, x_n\} \subset \mathcal{X}$, while for negative class we have $\mathfrak{Y} \triangleq \{y_1, y_2, \ldots, y_m\} \subset \mathcal{X}$. Following Shawe-Taylor and Cristianini (2004, Sec. 5.1.2), the Parzen window classifier is given by

$$f(z) = \operatorname{sgn}\left(\frac{1}{n}\sum_{i=1}^{n} k(z, x_i) - \frac{1}{m}\sum_{j=1}^{m} k(z, y_j) + b\right) = \operatorname{sgn}\left(\hat{\mu}_{\mathfrak{X}}(z) - \hat{\mu}_{\mathfrak{Y}}(z) + b\right), \tag{46}$$

where b is a bias term given by $b = \frac{1}{2}(\|\hat{\mu}_{\mathfrak{Y}}\|^2_{\mathcal{H}} - \|\hat{\mu}_{\mathfrak{X}}\|^2_{\mathcal{H}})$. Note that $f(z)$ is a threshold linear function in $\mathcal{H}$ with weight vector $\mathbf{w} = (1/n)\sum_{i=1}^{n}\phi(x_i) - (1/m)\sum_{j=1}^{m}\phi(y_j)$ (see Shawe-Taylor and Cristianini (2004, Sec. 5.1.2) for more detail). This algorithm is often referred to as the lazy algorithm as it does not require training.

In brief, the classifier (46) assigns the data point z to the class whose empirical kernel mean $\hat{\mu}$ is closer to the feature map $k(z, \cdot)$ of the data point in the RKHS. On the other hand, we may view the empirical kernel mean $\hat{\mu}_{\mathfrak{X}} \triangleq \frac{1}{n}\sum_{i=1}^{n} k(x_i, \cdot)$ (resp. $\hat{\mu}_{\mathfrak{Y}} \triangleq \frac{1}{m}\sum_{j=1}^{m} k(y_j, \cdot)$) as a standard empirical estimate, *i.e.*, KME, of the true kernel mean representation of the class-conditional distribution $\mathbb{P}(X|Y = +1)$ (resp. $\mathbb{P}(X|Y = -1)$). Given the improvement of shrinkage estimators over the empirical estimator of kernel mean, it is natural to expect

Dataset	Classification Error Rate			
	KME	B-KMSE	R-KMSE	S-KMSE
Climate Model	0.0348±0.0118	0.0348±0.0118	0.0348±0.0118	0.0348±0.0118
Ionosphere	0.2873±0.0343	**0.2768±0.0359**	**0.2749±0.0341**	0.2800±0.0367
Parkinsons	0.1318±0.0441	0.1250±0.0366	**0.1157±0.0395**	0.1309±0.0396
Pima	0.2951±0.0462	0.2921±0.0442	0.2937±0.0458	0.2943±0.0471
SPECTF	0.2583±0.0829	0.2597±0.0817	**0.2263±0.0626**	0.2417±0.0651
Iris	0.1079±0.0379	0.1071±0.0389	0.1055±0.0389	0.1040±0.0383
Wine	0.1301±0.0381	**0.1183±0.0445**	**0.1161±0.0414**	**0.1183±0.0431**

Table 1: The classification error rate of Parzen window classifier via different kernel mean estimators. The boldface represents the result whose difference from the baseline, *i.e.*, KME, is statistically significant.

that the performance of Parzen window classifier can be improved by employing shrinkage estimators of the true mean representation.

Our goal in this experiment is to compare the performance of Parzen window classifier using different kernel mean estimators. That is, we replace $\hat{\mu}_{\mathcal{X}}$ and $\hat{\mu}_{\mathcal{Y}}$ by their shrinkage counterparts and evaluate the resulting classifiers across several datasets taken from the UCI machine learning repository. In this experiment, we only consider the Gaussian RBF kernel whose bandwidth parameter is chosen by cross-validation procedure over a uniform grid $\sigma \in [0.1, 2]$. We use 30% of each dataset as a test set and the rest as a training set. We employ a simple pairwise coupling and majority vote for multi-class classification. We repeat the experiments 100 times and perform the paired-sample t-test on the results at 5% significance level. Table 1 reports the classification error rates of the Parzen window classifiers with different kernel mean estimators. Although the improvement is not substantial, we can see that the shrinkage estimators consistently give better performance than the standard estimator.

6.2.2 Density Estimation

We perform density estimation via kernel mean matching (Song et al., 2008), wherein we fit the density $Q = \sum_{j=1}^{m} \pi_j \mathcal{N}(\boldsymbol{\theta}_j, \sigma_j^2 \mathbf{I})$ to each dataset by the following minimization problem:

$$\min_{\boldsymbol{\pi}, \boldsymbol{\theta}, \boldsymbol{\sigma}} \|\hat{\mu} - \mu_Q\|_{\mathcal{H}}^2 \quad \text{subject to} \quad \sum_{j=1}^{m} \pi_j = 1,\ \pi_j \geq 0\,. \tag{47}$$

The empirical mean map $\hat{\mu}$ is obtained from samples using different estimators, whereas μ_Q is the kernel mean embedding of the density Q. Unlike experiments in Song et al. (2008), our goal is to compare different estimators of $\mu_{\mathbb{P}}$ (where $\mathbb{P}$ is the true data distribution), by replacing $\hat{\mu}$ in (47) with different shrinkage estimators. A better estimate of $\mu_{\mathbb{P}}$ should lead to better density estimation, as measured by the negative log-likelihood of Q on the test set, which we choose to be 30% of the dataset. For each dataset, we set the number of mixture components m to be 10. The model is initialized by running 50 random initializations using

the k-means algorithm and returning the best. We repeat the experiments 30 times and perform the paired sign test on the results at 5% significance level.[5]

The average negative log-likelihood of the model Q, optimized via different estimators, is reported in Table 2. In most cases, both R-KMSE and S-KMSE consistently achieve smaller negative log-likelihood when compared to KME. B-KMSE also tends to outperform the KME. However, in few cases the KMSEs achieve larger negative log-likelihood, especially when we use linear and degree-2 polynomial kernels. This highlight the potential of our estimators in a non-linear setting.

6.2.3 Discriminative Learning on Probability Distributions

The last experiment involves the discriminative learning on a collection of probability distributions via the kernel mean representation. A positive semi-definite kernel between distributions can be defined via their kernel mean embeddings. That is, given a training sample $(\widehat{\mathbb{P}}_1, y_1), \ldots, (\widehat{\mathbb{P}}_m, y_m) \in \mathcal{P} \times \{-1, +1\}$ where $\widehat{\mathbb{P}}_i := \frac{1}{n_i} \sum_{p=1}^{n_i} \delta_{x_p^i}$ and $x_p^i \sim \mathbb{P}_i$, the linear kernel between two distributions is approximated by

$$\langle \hat{\mu}_{\mathbb{P}_i}, \hat{\mu}_{\mathbb{P}_j} \rangle_{\mathcal{H}} = \left\langle \sum_{p=1}^{n_i} \beta_p^i \phi(x_p^i), \sum_{q=1}^{n_j} \beta_q^j \phi(x_q^j) \right\rangle_{\mathcal{H}} = \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} \beta_p^i \beta_q^j k(x_p^i, x_q^j),$$

where the weight vectors $\boldsymbol{\beta}^i$ and $\boldsymbol{\beta}^j$ come from the kernel mean estimates of $\mu_{\mathbb{P}_i}$ and $\mu_{\mathbb{P}_j}$, respectively. The non-linear kernel can then be defined accordingly, *e.g.*, $\kappa(\mathbb{P}_i, \mathbb{P}_j) = \exp(\|\hat{\mu}_{\mathbb{P}_i} - \hat{\mu}_{\mathbb{P}_j}\|_{\mathcal{H}}^2 / 2\sigma^2)$, see Christmann and Steinwart (2010). Our goal in this experiment is to investigate if the shrinkage estimators of the kernel mean improve the performance of discriminative learning on distributions. To this end, we conduct experiments on natural scene categorization using support measure machine (SMM) (Muandet et al., 2012) and group anomaly detection on a high-energy physics dataset using one-class SMM (OC-SMM) (Muandet and Schölkopf, 2013). We use both linear and non-linear kernels where the Gaussian RBF kernel is employed as an embedding kernel (Muandet et al., 2012). All hyper-parameters are chosen by 10-fold cross-validation.[6] For our unsupervised problem, we repeat the experiments using several parameter settings and report the best results. Table 3 reports the classification accuracy of SMM and the area under ROC curve (AUC) of OCSMM using different kernel mean estimators. All shrinkage estimators consistently lead to better performance on both SMM and OCSMM when compared to KME.

In summary, the proposed shrinkage estimators outperform the standard KME. While B-KMSE and R-KMSE are very competitive compared to KME, S-KMSE tends to outperform both B-KMSE and R-KMSE, however, sometimes leading to poor estimates depending on the dataset and the kernel function.

5. The paired sign test is a nonparametric test that can be used to examine whether two paired samples have the same distribution. In our case, we compare B-KMSE, R-KMSE and S-KMSE against KME.
6. In principle one can incorporate the shrinkage parameter into the cross-validation procedure. In this work we are only interested in the value of λ returned by the proposed LOOCV procedure.

	Dataset	LIN				POLY2				POLY3				RBF			
		KME	B-KMSE	R-KMSE	S-KMSE	KME	B-KMSE	R-KMSE	S-KMSE	KME	B-KMSE	R-KMSE	S-KMSE	KME	B-KMSE	R-KMSE	S-KMSE
1.	ionosphere	39.878	40.038	39.859	39.823	34.651	**34.352**	34.390	**34.009**	35.943	**35.575**	35.543	**34.617**	41.601	40.976	**40.817**	41.229
2.	sonar	72.240	**72.044**	72.198	**72.157**	100.420	**99.573**	**97.844**	**97.783**	72.294	**71.933**	72.003	71.835	98.540	**95.815**	**93.458**	**93.010**
3.	Australian	18.277	18.280	18.294	18.293	18.357	18.381	18.391	18.429	18.611	18.463	18.466	18.495	19.428	19.325	19.418	19.393
4.	specft	57.444	**57.2808**	57.218	**57.224**	67.018	66.979	**66.431**	**66.391**	59.585	**58.969**	60.006	60.616	65.674	65.138	65.039	**64.699**
5.	wdbc	31.801	31.759	31.776	31.781	32.421	32.310	32.373	32.316	31.183	**31.167**	**31.127**	31.110	36.471	36.453	36.335	35.898
6.	wine	16.019	16.000	16.039	16.009	17.070	16.920	**16.886**	16.960	16.393	16.300	16.309	16.202	17.569	17.546	17.498	17.498
7.	satimage	25.258	25.317	25.219	25.186	24.214	24.111	24.132	24.259	25.284	25.276	25.239	25.263	23.741	23.753	23.728	**24.384**
8.	segment	18.326	**17.868**	18.055	**18.124**	18.571	18.292	18.277	18.631	19.642	19.549	19.404	19.628	21.946	**21.598**	**21.580**	**21.822**
9.	vehicle	16.633	16.519	16.521	16.499	16.096	**15.998**	16.031	16.041	16.288	16.278	**16.281**	**16.263**	18.260	18.056	18.119	**17.911**
10.	svmguide2	27.298	27.273	27.281	27.276	27.812	**28.030**	27.985	**27.975**	28.014	**28.177**	**28.321**	28.250	28.132	28.122	28.119	28.020
11.	vowel	12.632	12.626	12.629	12.656	12.532	12.471	12.479	12.472	13.069	13.061	13.056	**13.054**	13.526	13.486	13.462	**13.453**
12.	housing	14.637	14.441	14.469	**14.296**	15.543	15.467	15.414	15.390	15.592	**15.543**	**15.509**	**15.408**	16.487	16.239	16.424	**16.019**
13.	bodyfat	17.527	17.362	**17.348**	17.396	17.386	17.358	17.356	17.329	16.418	16.393	**16.305**	**16.194**	17.875	17.652	17.607	**17.651**
14.	abalone	5.706	5.665	5.708	5.722	7.281	7.116	7.185	7.025	5.864	5.847	5.853	5.832	6.068	6.039	6.049	5.910
15.	glass	9.245	9.211	9.198	9.217	8.571	8.473	8.457	8.414	9.050	8.991	9.012	**8.737**	9.606	**9.605**	**9.575**	**9.573**

Table 2: Average negative log-likelihood of the model Q on test points over 30 randomizations. The boldface represents the result whose difference from the baseline, *i.e.*, KME, is statistically significant.

Estimator	Linear Kernel		Non-linear Kernel	
	SMM	OCSMM	SMM	OCSMM
KME	0.5432	0.6955	0.6017	0.9085
B-KMSE	0.5455	0.6964	0.6106	0.9088
R-KMSE	0.5521	0.6970	0.6303	0.9105
S-KMSE	0.5606	0.6970	0.6412	0.9063

Table 3: The classification accuracy of SMM and the area under ROC curve (AUC) of OCSMM using different estimators to construct the kernel on distributions.

7. Conclusion and Discussion

Motivated by the classical James-Stein phenomenon, in this paper, we proposed a shrinkage estimator for the kernel mean μ in a reproducing kernel Hilbert space $\mathcal{H}$ and showed they improve upon the empirical estimator $\hat{\mu}$ in the mean squared sense. We showed the proposed shrinkage estimator $\tilde{\mu}$ (with the shrinkage parameter being learned from data) to be $\sqrt{n}$-consistent and satisfies $\mathbb{E}\|\tilde{\mu} - \mu\|_{\mathcal{H}}^2 < \mathbb{E}\|\hat{\mu} - \mu\|_{\mathcal{H}}^2 + O(n^{-3/2})$ as $n \to \infty$. We also provided a regularization interpretation to shrinkage estimation, using which we also presented two shrinkage estimators, namely regularized shrinkage estimator and spectral shrinkage estimator, wherein the first one is closely related to $\hat{\mu}$ while the latter exploits the spectral decay of the covariance operator in $\mathcal{H}$. We showed through numerical experiments that the proposed estimators outperform the empirical estimator in various scenarios. Most importantly, the shrinkage estimators not only provide more accurate estimation, but also lead to superior performance on many real-world applications.

In this work, while we focused mainly on an estimation of the mean function in RKHS, it is quite straightforward to extend the shrinkage idea to estimate covariance (and cross-covariance) operators and tensors in RKHS (see Appendix A for a brief description). The key observation is that the covariance operator can be viewed as a mean function in a tensor RKHS. Covariance operators in RKHS are ubiquitous in many classical learning algorithms such as kernel PCA, kernel FDA, and kernel CCA. Recently, a preliminary investigation with some numerical results on shrinkage estimation of covariance operators is carried out in Muandet et al. (2014a) and Wehbe and Ramdas (2015). In the future, we intend to carry out a detailed study on the shrinkage estimation of covariance (and cross-covariance) operators.

Acknowledgments

The authors thanks the reviewers and the action editor for their detailed comments that significantly improved the manuscript. This work was partly done while Krikamol Muandet was visiting the Institute of Statistical Mathematics, Tokyo, and New York University, New York; and while Bharath Sriperumbudur was visiting the Max Planck Institute for Intelligent Systems, Germany. The authors wish to thank David Hogg and Ross Fedely for reading the first draft and giving valuable comments. We also thank Motonobu Kanagawa, Yu Nishiyama, and Ingo Steinwart for fruitful discussions. Kenji Fukumizu has been supported in part by MEXT Grant-in-Aid for Scientific Research on Innovative Areas 25120012.

Appendix A. Shrinkage Estimation of Covariance Operator

Let $(\mathcal{H}_X, k_X)$ and $(\mathcal{H}_Y, k_Y)$ be separable RKHSs of functions on measurable spaces $\mathcal{X}$ and $\mathcal{Y}$, with measurable reproducing kernels k_X and k_Y (with corresponding feature maps ϕ and φ), respectively. We consider a random vector $(X, Y) : \Omega \to \mathcal{X} \times \mathcal{Y}$ with distribution $\mathbb{P}_{XY}$. The marginal distributions of X and Y are denoted by $\mathbb{P}_X$ and $\mathbb{P}_Y$, respectively. If $\mathbb{E}_X k_X(X, X) < \infty$ and $\mathbb{E}_Y k_Y(Y, Y) < \infty$, then there exists a unique *cross-covariance*

operator $\Sigma_{YX} : \mathcal{H}_X \to \mathcal{H}_Y$ such that

$$\langle g, \Sigma_{YX} f \rangle_{\mathcal{H}_Y} = \mathbb{E}_{XY}[(f(X) - \mathbb{E}_X[f(X)])(g(Y) - \mathbb{E}_Y[g(Y)])] = Cov(f(X), g(Y))$$

holds for all $f \in \mathcal{H}_X$ and $g \in \mathcal{H}_Y$ (Baker, 1973; Fukumizu et al., 2004). If X is equal to Y, we obtain the self-adjoint operator Σ_{XX} called the *covariance operator*. Given i.i.d sample $\{(x_i, y_i)\}_{i=1}^n$ from $\mathbb{P}_{XY}$, we can write the empirical cross-covariance operator $\widehat{\Sigma}_{YX}$ as

$$\widehat{\Sigma}_{YX} \triangleq \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \otimes \varphi(y_i) - \hat{\mu}_X \otimes \hat{\mu}_Y \tag{48}$$

where $\hat{\mu}_X = \frac{1}{n} \sum_{i=1}^n \phi(x_i)$ and $\hat{\mu}_Y = \frac{1}{n} \sum_{i=1}^n \varphi(y_i)$.[7] Let $\tilde{\phi}$ and $\tilde{\varphi}$ be the centered version of the feature map ϕ and φ defined as $\tilde{\phi}(x) = \phi(x) - \hat{\mu}_X$ and $\tilde{\varphi}(y) = \varphi(y) - \hat{\mu}_Y$, respectively. Then, the empirical cross-covariance operator in (48) can be rewritten as

$$\widehat{\Sigma}_{YX} = \frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \otimes \tilde{\varphi}(y_i),$$

and therefore a shrinkage estimator of Σ_{YX}, *e.g.*, an equivalent of B-KMSE, can be constructed based on the ideas presented in this paper. That is, by the inner product property in product space, we have

$$\begin{aligned} \langle \tilde{\phi}(x) \otimes \tilde{\varphi}(y), \tilde{\phi}(x') \otimes \tilde{\varphi}(y') \rangle_{\mathcal{H}_X \otimes \mathcal{H}_Y} &= \langle \tilde{\phi}(x), \tilde{\phi}(x') \rangle_{\mathcal{H}_X} \langle \tilde{\varphi}(y), \tilde{\varphi}(y') \rangle_{\mathcal{H}_Y} \\ &= \tilde{k}_X(x, x') \tilde{k}_Y(y, y'). \end{aligned}$$

where $\tilde{k}_X$ and $\tilde{k}_Y$ denote the centered kernel functions. As a result, we can obtain the shrinkage estimators for Σ_{YX} by plugging the above kernel into the KMSEs.

References

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

Charles R. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:pp. 273–289, 1973.

Frank Bauer, Sergei Pereverzev, and Lorenzo Rosasco. On regularization algorithms in learning theory. *Journal of Complexity*, 23(1):52 – 72, 2007. ISSN 0885-064X.

James Berger and Robert Wolpert. Estimating the mean function of a Gaussian process and the Stein effect. *Journal of Multivariate Analysis*, 13(3):401–424, 1983.

James O. Berger. Admissible minimax estimation of a multivariate normal mean with arbitrary quadratic loss. *Annals of Statistics*, 4(1):223–226, 1976.

7. Although it is possible to estimate $\hat{\mu}_X$ and $\hat{\mu}_Y$ using our shrinkage estimators, the key novelty here is to directly shrink the *centered* covariance operator.

Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, 2004.

Andreas Christmann and Ingo Steinwart. Universal kernels on Non-Standard input spaces. In *Advances in Neural Information Processing Systems (NIPS)*, pages 406–414. 2010.

Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: Spectral clustering and normalized cuts. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 551–556, New York, NY, USA, 2004.

Joseph Diestel and John J. Uhl. *Vector Measures*. American Mathematical Society, Providence, 1977.

Nicolae Dinculeanu. *Vector Integration and Stochastic Integration in Banach Spaces*. Wiley, 2000.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

Bradley Efron and Carl N. Morris. Stein's paradox in statistics. *Scientific American*, 236 (5):119–127, 1977.

Kenji Fukumizu, Francis R. Bach, and Michael I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

Kenji Fukumizu, Francis R. Bach, and Arthur Gretton. Statistical consistency of kernel canonical correlation analysis. *Journal of Machine Learning Research*, 8:361–383, 2007.

Kenji Fukumizu, Le Song, and Arthur Gretton. Kernel Bayes' rule. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1737–1745. 2011.

Arthur Gretton, Karsten M. Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

Arthur Gretton, Kenji Fukumizu, Choon Hui Teo, Le Song, Bernhard Schölkopf, and Alexander J. Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*, pages 585–592. MIT Press, 2008.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

Marvin Gruber. *Improving Efficiency by Shrinkage: The James-Stein and Ridge Regression Estimators*. Statistics Textbooks and Monographs. Marcel Dekker, 1998.

Steffen Grünewälder, Guy Lever, Arthur Gretton, Luca Baldassarre, Sam Patterson, and Massimiliano Pontil. Conditional mean embeddings as regressors. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

Steffen Grünewälder, Arthur Gretton, and John Shawe-Taylor. Smooth operators. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

W. James and James Stein. Estimation with quadratic loss. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, pages 361–379. University of California Press, 1961.

JooSeuk Kim and Clayton D. Scott. Robust kernel density estimation. *Journal of Machine Learning Research*, 13:2529–2565, Sep 2012.

Avi Mandelbaum and L. A. Shepp. Admissibility as a touchstone. *Annals of Statistics*, 15 (1):252–268, 1987.

Krikamol Muandet and Bernhard Schölkopf. One-class support measure machines for group anomaly detection. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2013.

Krikamol Muandet, Kenji Fukumizu, Francesco Dinuzzo, and Bernhard Schölkopf. Learning from distributions via support measure machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 10–18. 2012.

Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Arthur Gretton, and Bernhard Schölkopf. Kernel mean estimation and Stein effect. In *ICML*, pages 10–18, 2014a.

Krikamol Muandet, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean estimation via spectral filtering. In *Advances in Neural Information Processing Systems 27*, pages 1–9. Curran Associates, Inc., 2014b.

Nicolas Privault and Anthony Rveillac. Stein estimation for the drift of Gaussian processes using the Malliavin calculus. *Annals of Statistics*, 36(5):2531–2550, 2008.

Carl E. Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Michael Reed and Barry Simon. *Functional Analysis*. Academic Press, New York, 1972.

Zoltán Sasvári. *Multivariate Characteristic and Correlation Functions*. De Gruyter, Berlin, Germany, 2013.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, July 1998.

Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*, COLT '01/EuroCOLT '01, pages 416–426, London, UK, UK, 2001. Springer-Verlag.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK, 2004.

Alexander J. Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A Hilbert space embedding for distributions. In *Proceedings of the 18th International Conference on Algorithmic Learning Theory (ALT)*, pages 13–31. Springer-Verlag, 2007.

Le Song, Xinhua Zhang, Alex Smola, Arthur Gretton, and Bernhard Schölkopf. Tailoring density estimation via reproducing kernel moment matching. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 992–999, 2008.

Le Song, Byron Boots, Sajid M. Siddiqi, Geoffrey Gordon, and Alexander J. Smola. Hilbert space embeddings of hidden Markov models. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

Le Song, Ankur P. Parikh, and Eric P. Xing. Kernel embeddings of latent tree graphical models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2708–2716, 2011.

Bharath Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Gert Lanckriet, and Bernhard Schölkopf. Injective Hilbert space embeddings of probability measures. In *The 21st Annual Conference on Learning Theory (COLT)*, 2008.

Bharath Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 99:1517–1561, 2010.

Bharath Sriperumbudur, Kenji Fukumizu, and Gert Lanckriet. Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research*, 12: 2389–2410, 2011.

Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert R. G. Lanckriet. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599, 2012. doi: 10.1214/12-EJS722.

Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Aapo Hyvärinen, and Revant Kumar. Density estimation in infinite dimensional exponential families. 2013. http://arxiv.org/pdf/1312.3516.

Charles Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the 3rd Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 197–206. University of California Press, 1955.

Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer, New York, 2008.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, Cambridge, 1998.

Vladimir Vapnik. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

Larry Wasserman. *All of Nonparametric Statistics*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

Leila Wehbe and Aaditya Ramdas. Nonparametric independence testing for small sample sizes. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3777–3783, July 2015.

Holger Wendland. *Scattered Data Approximation*. Cambridge University Press, Cambridge, UK, 2005.

Vadim Yurinsky. *Sums and Gaussian Vectors*, volume 1617 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1995.

Journal of Machine Learning Research 17 (2016) 1-49 Submitted 5/14; Revised 1/16; Published 5/16

SPSD Matrix Approximation vis Column Selection: Theories, Algorithms, and Extensions

Shusen Wang WSSATZJU@GMAIL.COM
Department of Statistics
University of California at Berkeley
Berkeley, CA 94720

Luo Luo RICKY@SJTU.EDU.CN
Zhihua Zhang* ZHIHUA@SJTU.EDU.CN
Department of Computer Science and Engineering
Shanghai Jiao Tong University
800 Dong Chuan Road, Shanghai, China 200240

Editor: Inderjit Dhillon

Abstract

Symmetric positive semidefinite (SPSD) matrix approximation is an important problem with applications in kernel methods. However, existing SPSD matrix approximation methods such as the Nyström method only have weak error bounds. In this paper we conduct in-depth studies of an SPSD matrix approximation model and establish strong relative-error bounds. We call it the prototype model for it has more efficient and effective extensions, and some of its extensions have high scalability. Though the prototype model itself is not suitable for large-scale data, it is still useful to study its properties, on which the analysis of its extensions relies.

This paper offers novel theoretical analysis, efficient algorithms, and a highly accurate extension. First, we establish a lower error bound for the prototype model and improve the error bound of an existing column selection algorithm to match the lower bound. In this way, we obtain the first optimal column selection algorithm for the prototype model. We also prove that the prototype model is exact under certain conditions. Second, we develop a simple column selection algorithm with a provable error bound. Third, we propose a so-called spectral shifting model to make the approximation more accurate when the eigenvalues of the matrix decay slowly, and the improvement is theoretically quantified. The spectral shifting method can also be applied to improve other SPSD matrix approximation models.

Keywords: Matrix approximation, matrix factorization, kernel methods, the Nyström method, spectral shifting

1. Introduction

The kernel methods are important tools in machine learning, computer vision, and data mining (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004). However, for two reasons, most kernel methods have scalability difficulties. First, given n data points of d dimension, generally we need $\mathcal{O}(n^2d)$ time to form the $n \times n$ kernel matrix $\mathbf{K}$. Second, most kernel methods require expensive matrix computations. For example, Gaussian process regression and classification require inverting some $n \times n$ matrices which costs $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ memory; kernel PCA and spectral clustering perform the truncated eigenvalue decomposition which takes $\tilde{\mathcal{O}}(n^2k)$ time[1] and $\mathcal{O}(n^2)$ memory, where k is the target rank of the decomposition.

Besides high time complexities, these matrix operations also have high memory cost and are difficult to implement in distributed computing facilities. The matrix decomposition and (pseudo) inverse operations are generally solved by numerical iterative algorithms, which go many passes through the matrix until convergence. Thus, the whole matrix had better been placed in main memory, otherwise in each iteration there would be a swap between memory and disk, which incurs high I/O costs and can be more expensive than CPU time. Unless the algorithm is pass-efficient, that is, it goes constant passes through the data matrix, the main memory should be at least the size of the data matrix. For two reasons, such iterative algorithms are expensive even if they are performed in distributed computing facilities such as MapReduce. First, the memory cost is too expensive for each individual machine to stand. Second, communication and synchronization must be performed in each iteration of the numerical algorithms, so the cost of each iteration is high.

Many matrix approximation methods have been proposed to make kernel machines scalable. Among them the Nyström method (Nyström, 1930; Williams and Seeger, 2001) and random features (Rahimi and Recht, 2008) are the most efficient and widely applied. However, only weak results are known (Drineas and Mahoney, 2005; Gittens and Mahoney, 2013; Lopez-Paz et al., 2014). Yang et al. (2012) showed that the Nyström method is likely a better choice than random features, both theoretically and empirically. However, even the Nyström method cannot attain high accuracy. The lower bound in (Wang and Zhang, 2013) indicates that the Nyström method costs at least $\Omega(n^2k/\epsilon)$ time and $\Omega(n^{1.5}k^{0.5}\epsilon^{-0.5})$ memory to attain $1+\epsilon$ Frobenius norm error bound relative to the best rank k approximation.

In this paper we investigate a more accurate low-rank approximation model proposed by Halko et al. (2011); Wang and Zhang (2013), which we refer to as the prototype model. For any symmetric positive semidefinite (SPSD) matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$, the prototype model first draws a random matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$ and forms a sketch $\mathbf{C} = \mathbf{KP}$, and then computes the intersection matrix

$$\mathbf{U}^\star \;=\; \operatorname*{argmin}_{\mathbf{U}} \|\mathbf{K} - \mathbf{CUC}^T\|_F^2 \;=\; \mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T \;\in\; \mathbb{R}^{c\times c}. \tag{1}$$

Finally, the model approximates $\mathbf{K}$ by $\mathbf{CU}^\star\mathbf{C}^T$. With this low-rank approximation at hand, it takes time $\mathcal{O}(nc^2)$ to compute the approximate matrix inversion and eigenvalue decomposition. In the following we discuss how to form $\mathbf{C}$ and compute $\mathbf{U}^\star$.

Column Selection vs. Random Projection. Although the sketch $\mathbf{C} = \mathbf{KP}$ can be formed by either random projection or column selection, when applied to the kernel methods, column selection is preferable to random projection. As aforementioned, suppose we are given n data points of d dimension. It takes time $\mathcal{O}(n^2d)$ to compute the whole of the kernel matrix $\mathbf{K}$, which is prohibitive when n is in million scale. Unfortunately, whatever

∗. Corresponding author.

1. The $\tilde{\mathcal{O}}$ notation hides the logarithm terms and the data-dependent spectral gap parameter.

existing random projection technique is employed to form the sketch $\mathbf{C}$, every entry of $\mathbf{K}$ must be visited. In contrast, by applying data independent column selection algorithms such as uniform sampling, we can form $\mathbf{C}$ by observing only $\mathcal{O}(nc)$ entries of $\mathbf{K}$. At present all the existing column selection algorithms, including our proposed uniform+adaptive^2 algorithm, cannot avoid observing the whole of $\mathbf{K}$ while keeping constant-factor bound. Nevertheless, we conjecture that our uniform+adaptive^2 algorithm can be adapted to satisfy these two properties simultaneously (see Section 5.4 for discussions in detail).

The Intersection Matrix. With the sketch $\mathbf{C}$ at hand, it remains to compute the intersection matrix. The most straightforward way is (1), which minimizes the Frobenius norm approximation error. However, this approach has two drawbacks. First, it again requires the full observation of $\mathbf{K}$. Second, the matrix product $\mathbf{C}^\dagger\mathbf{K}$ costs $\mathcal{O}(n^2c)$ time. The prototype model is therefore time-inefficient. Fortunately, Wang et al. (2015) recently overcame the two drawbacks by solving (1) approximately rather than optimally. Wang et al. (2015) obtained the approximate intersection matrix $\tilde{\mathbf{U}}$ in $\mathcal{O}(nc^3/\epsilon)$ time while keeping

$$\|\mathbf{K}-\mathbf{C}\tilde{\mathbf{U}}\mathbf{C}^T\|_F^2 \;\le\; (1+\epsilon)\min_{\mathbf{U}}\|\mathbf{K}-\mathbf{C}\mathbf{U}\mathbf{C}^T\|_F^2 \tag{2}$$

with high probability.

With the more efficient solution, why is it useful to study the exact solution to the prototype model (1)? On the one hand, from (2) we can see that the quality of the approximation depends on the prototype model, thus improvement of the prototype model directly applies to the more efficient model. On the other hand, for medium-scale problems where $\mathbf{K}$ does not fit in memory, the prototype model can produce high quality approximation with reasonable time expense. The experiment on kernel PCA in Section 6.3 shows that the prototype model is far more accurate than the Nyström method. For the above two reasons, we believe the study of the prototype model is useful.

1.1 Contributions

Our contributions mainly include three aspects: theoretical analysis, column selection algorithms, and extensions. They are summarized as follows.

1.1.1 Contributions: Theories

Kumar et al. (2009); Talwalkar and Rostamizadeh (2010) previously showed that the Nyström method is exact when the original kernel matrix is low-rank. In Section 4.1 we show that the prototype model exactly recovers the original SPSD matrix under the same conditions.

The prototype model with the near-optimal+adaptive column sampling algorithm satisfies $1+\epsilon$ relative-error bound when $c=\mathcal{O}(k/\epsilon^2)$ (Wang and Zhang, 2013). It was unknown whether this upper bound is optimal. In Section 4.2 we establish a lower error bound for the prototype model. We show that at least $2k/\epsilon$ columns must be chosen to attain $1+\epsilon$ bound. In Theorem 3 we improve the upper error bound of the near-optimal+adaptive algorithm to $\mathcal{O}(k/\epsilon)$, which matches the lower bound up to a constant factor.

1.1.2 Contributions: Algorithms

In Section 5 we devise a simple column selection algorithm which we call *the uniform+adaptive^2 algorithm*. The uniform+adaptive^2 algorithm is more efficiently and more easily implemented than the near-optimal+adaptive algorithm of Wang and Zhang (2013), yet its error bound is comparable with the near-optimal+adaptive algorithm. It is worth mentioning that our uniform+adaptive^2 algorithm is the adaptive-full algorithm of (Figure 3, Kumar et al., 2012) with two rounds of adaptive sampling, and thus our results theoretically justify the adaptive-full algorithm.

1.1.3 Contributions: Extension

When the spectrum of a matrix decays slowly (that is, the $c+1$ to n largest eigenvalues are not small enough), all of the low-rank approximations are far from the original kernel matrix. Inspired by Zhang (2014), we propose a new method called *spectral shifting (SS)* to make the approximation still effective even when the spectrum decays slowly. Unlike the low-rank approximation $\mathbf{K}\approx\mathbf{C}\mathbf{U}\mathbf{C}^T$, the spectral shifting model approximates $\mathbf{K}$ by $\mathbf{K}\approx\tilde{\mathbf{C}}\mathbf{U}^{\mathrm{ss}}\tilde{\mathbf{C}}^T+\delta^{\mathrm{ss}}\mathbf{I}_n$, where $\mathbf{C},\tilde{\mathbf{C}}\in\mathbb{R}^{n\times c}$, $\mathbf{U},\mathbf{U}^{\mathrm{ss}}\in\mathbb{R}^{c\times c}$, and $\delta^{\mathrm{ss}}\ge 0$. When the spectrum of $\mathbf{K}$ decays slowly, the term $\delta^{\mathrm{ss}}\mathbf{I}_n$ helps to improve the approximation accuracy. In Section 7 we describe the spectral shifting method in detail.

We highlight that the spectral shifting method can naturally apply to improve other kernel approximation models such as the memory efficient kernel approximation (MEKA) model (Si et al., 2014). Experiments demonstrate that MEKA can be significantly improved by spectral shifting.

1.2 Paper Organization

The remainder of this paper is organized as follows. In Section 2 we define the notation. In Section 3 we introduce the motivations of SPSD matrix approximation and define the SPSD matrix approximation models. Then we present our work—theories, algorithms, and extension—respectively in Sections 4, 5, and 7. In Section 6 we conduct experiments to compare among the column sampling algorithms. In Section 8 we empirically evaluate the proposed spectral shifting model. All the proofs are deferred to the appendix.

2. Notation

Let $[n]=\{1,\dots,n\}$, and $\mathbf{I}_n$ be the $n\times n$ identity matrix. For an $m\times n$ matrix $\mathbf{A}=[a_{ij}]$, we let $\mathbf{a}_{i:}$ be its i-th row, $\mathbf{a}_{:i}$ be its i-th column, and use $\mathbf{a}_i$ to denote either row or column when there is no ambiguity. Let $\mathbf{A}_1\oplus\mathbf{A}_2\oplus\cdots\oplus\mathbf{A}_q$ be the block diagonal matrix whose the i-th diagonal block is $\mathbf{A}_i$. Let $\|\mathbf{A}\|_F=(\sum_{i,j}a_{ij}^2)^{1/2}$ be the Frobenius norm and $\|\mathbf{A}\|_2=\max_{\mathbf{x}\neq\mathbf{0}}\|\mathbf{A}\mathbf{x}\|_2/\|\mathbf{x}\|_2$ be the spectral norm.

Letting $\rho=\mathrm{rank}(\mathbf{A})$, we write the condensed singular value decomposition (SVD) of $\mathbf{A}$ as $\mathbf{A}=\mathbf{U}_\mathbf{A}\mathbf{\Sigma}_\mathbf{A}\mathbf{V}_\mathbf{A}^T$, where the (i,i)-th entry of $\mathbf{\Sigma}_\mathbf{A}\in\mathbb{R}^{\rho\times\rho}$ is the i-th largest singular value of $\mathbf{A}$ (denoted $\sigma_i(\mathbf{A})$). Unless otherwise specified, in this paper "SVD" means the condensed SVD. We also let $\mathbf{U}_{\mathbf{A},k}$ and $\mathbf{V}_{\mathbf{A},k}$ be the first k ($<\rho$) columns of $\mathbf{U}_\mathbf{A}$ and $\mathbf{V}_\mathbf{A}$, respectively, and $\mathbf{\Sigma}_{\mathbf{A},k}$ be the $k\times k$ top sub-block of $\mathbf{\Sigma}_\mathbf{A}$. Then the $m\times n$ matrix $\mathbf{A}_k=\mathbf{U}_{\mathbf{A},k}\mathbf{\Sigma}_{\mathbf{A},k}\mathbf{V}_{\mathbf{A},k}^T$ is the "closest" rank-k approximation to $\mathbf{A}$.

If $\mathbf{A}$ is normal, we let $\mathbf{A}=\mathbf{U}_\mathbf{A}\mathbf{\Lambda}_\mathbf{A}\mathbf{U}_\mathbf{A}^T$ be the eigenvalue decomposition, and denote the i-th diagonal entry of $\mathbf{\Lambda}_\mathbf{A}$ by $\lambda_i(\mathbf{A})$, where $|\lambda_1(\mathbf{A})|\ge\cdots\ge|\lambda_n(\mathbf{A})|$. When $\mathbf{A}$ is SPSD, the SVD and the eigenvalue decomposition of $\mathbf{A}$ are identical.

Table 1: Comparisons among the matrix approximation models in Section 3.2 and Section 3.3. Here "#Entries" denotes the number of entries of $\mathbf{K}$ required to observe. The costs of column selection is not counted; they are listed separately in Table 2.

	Time	Memory	#Entries	Theory
Prototype	$\mathcal{O}(n^2c)$	$\mathcal{O}(nc)$	n^2	$1+\epsilon$ relative-error
Faster	$\mathcal{O}(nc^3/\epsilon)$	$\mathcal{O}(nc)$	nc^2/ϵ	$1+\epsilon$ relative-error
Nyström	$\mathcal{O}(nc^2)$	$\mathcal{O}(nc)$	nc	weak
SS	the same to "prototype"			stronger than "prototype"
Faster SS	the same to "faster"			unknown

Based on SVD, the *matrix coherence* of the columns of $\mathbf{A}$ relative to the best rank-k approximation is defined as $\mu_k = \frac{n}{k}\max_j \left\|(\mathbf{V}_{\mathbf{A},k})_{j:}\right\|_2^2$. Let $\mathbf{A}^\dagger = \mathbf{V}_\mathbf{A}\mathbf{\Sigma}_\mathbf{A}^{-1}\mathbf{U}_\mathbf{A}^T$ be the *Moore-Penrose inverse* of $\mathbf{A}$. When $\mathbf{A}$ is nonsingular, the Moore-Penrose inverse is identical to the matrix inverse. Given another $n\times c$ matrix $\mathbf{C}$, we define $\mathcal{P}_\mathbf{C}(\mathbf{A}) = \mathbf{C}\mathbf{C}^\dagger\mathbf{A}$ as the projection of $\mathbf{A}$ onto the column space of $\mathbf{C}$ and $\mathcal{P}_{\mathbf{C},k}(\mathbf{A}) = \mathbf{C}\cdot\mathrm{argmin}_{\mathrm{rank}(\mathbf{X})\le k}\|\mathbf{A}-\mathbf{C}\mathbf{X}\|_F$ as the rank restricted projection. It is obvious that $\|\mathbf{A}-\mathcal{P}_\mathbf{C}(\mathbf{A})\|_F \le \|\mathbf{A}-\mathcal{P}_{\mathbf{C},k}(\mathbf{A})\|_F$.

3. SPSD Matrix Approximation Models

In Section 3.1 we provide motivating examples to show why SPSD matrix approximation is useful. In Section 3.2 we formally describe low-rank approximation models. In Section 3.3 we describe the spectral shifting model. In Table 1 we compare the matrix approximation models defined in Section 3.2 and Section 3.3.

3.1 Motivations

Let $\mathbf{K}$ be an $n\times n$ kernel matrix. Many kernel methods require the eigenvalue decomposition of $\mathbf{K}$ or solving certain linear systems involving $\mathbf{K}$.

- Spectral clustering, kernel PCA, and manifold learning need to perform the rank k eigenvalue decomposition which costs $\tilde{\mathcal{O}}(n^2k)$ time and $\mathcal{O}(n^2)$ memory.
- Gaussian process regression and classification both require solving this kind of linear systems:
$$(\mathbf{K}+\alpha\mathbf{I}_n)\mathbf{b} = \mathbf{y}, \tag{3}$$
whose solution is $\mathbf{b}^\star = (\mathbf{K}+\alpha\mathbf{I}_n)^{-1}\mathbf{y}$. Here α is a constant. This costs $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ memory.

Fortunately, if we can efficiently find an approximation in the form
$$\tilde{\mathbf{K}} \;=\; \mathbf{L}\mathbf{L}^T+\delta\mathbf{I}_n \;\approx\; \mathbf{K},$$
where $\delta\ge 0$ and $\mathbf{L}\in\mathbb{R}^{n\times l}$ with $l\ll n$, then the eigenvalue decomposition and linear systems can be approximately solved in $\mathcal{O}(nl^2)$ time and $\mathcal{O}(nl)$ space in the following way.

Algorithm 1 Computing the Prototype Model in $\mathcal{O}(nc+nd)$ Memory.

Input: data points $\mathbf{x}_1,\cdots,\mathbf{x}_n\in\mathbb{R}^d$, kernel function $\kappa(\cdot,\cdot)$.
Compute $\mathbf{C}$ and $\mathbf{C}^\dagger$; *// In $\mathcal{O}(nc+nd)$ memory and $\mathcal{O}(ncd+nc^2)$ time*
Form a $c\times n$ all-zero matrix $\mathbf{D}$; *// In $\mathcal{O}(nc)$ memory and $\mathcal{O}(nc)$ time*
for $j=1$ to n **do**
    Form the j-th column of $\mathbf{K}$ by $\mathbf{k}_j=[\kappa(\mathbf{x}_1,\mathbf{x}_j),\cdots,\kappa(\mathbf{x}_n,\mathbf{x}_j)]^T$;
    Compute the j-th column of $\mathbf{D}$ by $\mathbf{d}_j=\mathbf{C}^\dagger\mathbf{k}_j$;
    Delete $\mathbf{k}_j$ from memory;
end for
// Now the matrix $\mathbf{D}$ is $\mathbf{C}^\dagger\mathbf{K}$
// The loop totaly costs $\mathcal{O}(nc+nd)$ memory and $\mathcal{O}(n^2d+n^2c)$ time
Compute $\mathbf{U}=\mathbf{D}(\mathbf{C}^\dagger)^T$; *// In $\mathcal{O}(nc)$ memory and $\mathcal{O}(nc^2)$ time*
return $\mathbf{C}$ and $\mathbf{U}$ $(=\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T)$.

- Approximate Eigenvalue Decomposition. Let $\mathbf{L}=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the SVD and $\mathbf{U}_\perp$ be the orthogonal complement of $\mathbf{U}$. Then the full eigenvalue decomposition of $\tilde{\mathbf{K}}$ is
$$\tilde{\mathbf{K}} \;=\; \mathbf{U}(\mathbf{\Sigma}^2+\delta\mathbf{I}_l)\mathbf{U}^T+\mathbf{U}_\perp(\delta\mathbf{I}_{n-l})\mathbf{U}_\perp^T.$$
- Approximately Solving the Linear Systems. Here we use a more general form: $\tilde{\mathbf{K}}=\mathbf{L}\mathbf{L}^T+\mathbf{\Delta}$, where $\mathbf{\Delta}$ is a diagonal matrix with positive diagonal entries. Then
$$\begin{aligned}\mathbf{b}^\star &= (\mathbf{K}+\alpha\mathbf{I}_n)^{-1}\mathbf{y} \;\approx\; (\mathbf{L}\mathbf{L}^T+\mathbf{\Delta}+\alpha\mathbf{I}_n)^{-1}\mathbf{y} \;=\; (\mathbf{L}\mathbf{L}^T+\mathbf{\Delta}')^{-1}\mathbf{y}\\ &= \mathbf{\Delta}'^{-1}\mathbf{y}-\underbrace{\mathbf{\Delta}'^{-1}\mathbf{L}}_{n\times l}\underbrace{(\mathbf{I}_l+\mathbf{L}^T\mathbf{\Delta}'^{-1}\mathbf{L})^{-1}}_{l\times l}\underbrace{\mathbf{L}^T\mathbf{\Delta}'^{-1}}_{l\times n}\mathbf{y}.\end{aligned}$$
Here the second equality is obtained by letting $\mathbf{\Delta}'=\mathbf{\Delta}+\alpha\mathbf{I}_n$, and the third equality follows by the Sherman-Morrison-Woodbury matrix identity.

The remaining problem is to find such matrix approximation efficiently while keeping $\tilde{\mathbf{K}}$ close to $\mathbf{K}$.

3.2 Low-Rank Matrix Approximation Models

We first recall the prototype model introduced previously and then discuss its approximate solutions. In fact, the famous Nyström method (Nyström, 1930; Williams and Seeger, 2001) is an approximation to the prototype model. Throughout this paper, we let $\mathbf{P}\in\mathbb{R}^{n\times c}$ be random projection or column selection matrix and $\mathbf{C}=\mathbf{K}\mathbf{P}$ be a sketch of $\mathbf{K}$. The only difference among the discussed models is their intersection matrices.

The Prototype Model. Suppose we have $\mathbf{C}\in\mathbb{R}^{n\times c}$ at hand. It remains to find an intersection matrix $\mathbf{U}\in\mathbb{R}^{c\times c}$. Since our objective is to make $\mathbf{C}\mathbf{U}\mathbf{C}^T$ close to $\mathbf{K}$, it is straightforward to optimize their difference. The prototype model computes the intersection matrix by
$$\mathbf{U}^\star \;=\; \mathop{\mathrm{argmin}}_{\mathbf{U}}\|\mathbf{K}-\mathbf{C}\mathbf{U}\mathbf{C}^T\|_F^2 \;=\; \mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger \;\in\; \mathbb{R}^{c\times c}. \tag{4}$$

With $\mathbf{C}$ at hand, the prototype model still needs one pass through the data, and it costs $\mathcal{O}(n^2c)$ time. When applied to kernel methods, the memory cost is $\mathcal{O}(nc+nd)$ (see Algorithm 1), where n is the number of data points and d is the dimension. The prototype model has the same time complexity as the exact rank k eigenvalue decomposition, but it is more memory-efficient and pass-efficient.

Halko et al. (2011) showed that when $\mathbf{P}$ is a standard Gaussian matrix and $c=\mathcal{O}(k/\epsilon)$, the prototype model attains $2+\epsilon$ error relative to $\|\mathbf{K}-\mathbf{K}_k\|_F^2$. Wang and Zhang (2013) showed that when $\mathbf{C}$ contains $c=\mathcal{O}(k/\epsilon^2)$ columns selected by the near-optimal+adaptive sampling algorithm, the prototype model attains $1+\epsilon$ relative error. In Section 5.2 we improve the result to $c=\mathcal{O}(k/\epsilon)$, which is near optimal.

Faster SPSD matrix Approximation Model. Wang et al. (2015) noticed that (4) is a strongly over-determined linear system, and thus proposed to solve (4) by randomized approximations. They proposed to sample $s=\mathcal{O}(c\sqrt{n/\epsilon}) \ll n$ columns according to the row leverage scores of $\mathbf{C}$, which costs $\mathcal{O}(nc^2)$ time. Let $\mathbf{S}\in\mathbb{R}^{n\times s}$ be the corresponding column selection matrix. They proposed the faster SPSD matrix approximation model which computes the intersection matrix by

$$\tilde{\mathbf{U}} \;=\; \operatorname*{argmin}_{\mathbf{U}} \left\|\mathbf{S}^T(\mathbf{K}-\mathbf{C}\mathbf{U}\mathbf{C}^T)\mathbf{S}\right\|_F^2 \;=\; \underbrace{(\mathbf{S}^T\mathbf{C})^\dagger}_{c\times s}\underbrace{(\mathbf{S}^T\mathbf{K}\mathbf{S})}_{s\times s}\underbrace{(\mathbf{C}^T\mathbf{S})^\dagger}_{s\times c} \;\in\; \mathbb{R}^{c\times c}. \tag{5}$$

The faster model visits only $s^2=\mathcal{O}(nc^2/\epsilon)=o(n^2)$ entries of $\mathbf{K}$, and the time complexity is $\mathcal{O}(nc^2+s^2c)=\mathcal{O}(nc^3/\epsilon)$. The following error bound is satisfied with high probability:

$$\|\mathbf{K}-\mathbf{C}\tilde{\mathbf{U}}\mathbf{C}^T\|_F^2 \;\le\; (1+\epsilon)\min_{\mathbf{U}}\|\mathbf{K}-\mathbf{C}\mathbf{U}\mathbf{C}^T\|_F^2.$$

This implies that if $\mathbf{C}$ is such a high quality sketch that the prototype model satisfies $1+\epsilon$ relative-error bound, then the faster SPSD matrix approximation model also satisfies $1+\epsilon$ relative-error bound.

The Nyström Method. The Nyström method is a special case of the faster SPSD matrix approximation model, and therefore it is also an approximate solution to (4). If we let the two column selection matrices $\mathbf{S}$ and $\mathbf{P}$ be the same, then (5) becomes

$$\tilde{\mathbf{U}} \;=\; \operatorname*{argmin}_{\mathbf{U}} \left\|\mathbf{P}^T(\mathbf{K}-\mathbf{C}\mathbf{U}\mathbf{C}^T)\mathbf{P}\right\|_F^2 \;=\; \underbrace{(\mathbf{P}^T\mathbf{K}\mathbf{P})^\dagger}_{c\times c}\underbrace{(\mathbf{P}^T\mathbf{K}\mathbf{P})}_{c\times c}\underbrace{(\mathbf{P}^T\mathbf{K}\mathbf{P})^\dagger}_{c\times c} \;=\; \underbrace{(\mathbf{P}^T\mathbf{K}\mathbf{P})^\dagger}_{c\times c}.$$

The matrix $(\mathbf{P}^T\mathbf{K}\mathbf{P})^\dagger$ is exactly the intersection matrix of the Nyström method. The Nyström method costs only $\mathcal{O}(nc^2)$ time, and it can be applied to million-scale problems (Talwalkar et al., 2013). However, its accuracy is low. Much work in the literature has analyzed the error bound of the Nyström method, but only weak results are known (Drineas and Mahoney, 2005; Shawe-Taylor et al., 2005; Kumar et al., 2012; Jin et al., 2013; Gittens and Mahoney, 2013). Wang and Zhang (2013) even showed that the Nyström method cannot attain $1+\epsilon$ relative-error bound unless $c\ge\Omega(\sqrt{nk/\epsilon})$. Equivalently, to attain $1+\epsilon$ bound, the Nyström would take $\Omega(n^2k/\epsilon)$ time and $\Omega(n^{1.5}k^{0.5}\epsilon^{-0.5})$ memory.

3.3 Spectral Shifting Models

We propose a more accurate SPSD matrix approximation method called the spectral shifting model. Here we briefly describe the model and its fast solution. The theoretical analysis is left to Section 7.

The Spectral Shifting (SS) Model. As before, we let $\mathbf{P}\in\mathbb{R}^{n\times c}$ be a column selection matrix and $\bar{\mathbf{C}}=\bar{\mathbf{K}}\mathbf{P}$, where $\bar{\mathbf{K}}=\mathbf{K}$ or $\bar{\mathbf{K}}=\mathbf{K}-\bar{\delta}\mathbf{I}_n$ for some parameter $\bar{\delta}\ge 0$. We approximate $\mathbf{K}$ by $\bar{\mathbf{C}}\mathbf{U}^{\mathrm{ss}}\bar{\mathbf{C}}^T+\delta^{\mathrm{ss}}\mathbf{I}_n$, where

$$\left(\mathbf{U}^{\mathrm{ss}},\delta^{\mathrm{ss}}\right) \;=\; \operatorname*{argmin}_{\mathbf{U},\delta}\left\|\mathbf{K}-\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T-\delta\mathbf{I}_n\right\|_F^2. \tag{6}$$

This optimization problem has closed-form solution (see Theorem 6)

$$\begin{aligned}\delta^{\mathrm{ss}} &= \frac{1}{n-\mathrm{rank}(\bar{\mathbf{C}})}\Big(\mathrm{tr}(\mathbf{K})-\mathrm{tr}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}\bar{\mathbf{C}}\big)\Big),\\ \mathbf{U}^{\mathrm{ss}} &= \bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T-\delta^{\mathrm{ss}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger,\end{aligned}$$

which can be computed in $\mathcal{O}(n^2c)$ time and $\mathcal{O}(nc)$ memory. Later we will show that the SS model is more accurate than the prototype model.

Faster Spectral Shifting Model. The same idea of Wang et al. (2015) also applies to the SS model (14). Specifically, we can draw another column selection matrix $\mathbf{S}\in\mathbb{R}^{n\times s}$ and solve

$$\begin{aligned}\left(\tilde{\mathbf{U}}^{\mathrm{ss}},\tilde{\delta}^{\mathrm{ss}}\right) &= \operatorname*{argmin}_{\mathbf{U},\delta}\left\|\mathbf{S}^T\big(\mathbf{K}-\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T-\delta\mathbf{I}_n\big)\mathbf{S}\right\|_F^2\\ &= \operatorname*{argmin}_{\mathbf{U},\delta}\left\|\mathbf{S}^T\mathbf{K}\mathbf{S}-(\mathbf{S}^T\bar{\mathbf{C}})\mathbf{U}(\mathbf{S}^T\bar{\mathbf{C}})^T-\delta\mathbf{I}_s\right\|_F^2.\end{aligned}$$

Similarly, it has closed-form solution

$$\begin{aligned}\tilde{\delta}^{\mathrm{ss}} &= \frac{1}{s-\mathrm{rank}(\mathbf{S}^T\bar{\mathbf{C}})}\Big[\mathrm{tr}\big(\mathbf{S}^T\mathbf{K}\mathbf{S}\big)-\mathrm{tr}\Big((\mathbf{S}^T\bar{\mathbf{C}})^\dagger(\mathbf{S}^T\mathbf{K}\mathbf{S})(\mathbf{S}^T\bar{\mathbf{C}})\Big)\Big],\\ \tilde{\mathbf{U}}^{\mathrm{ss}} &= (\mathbf{S}^T\bar{\mathbf{C}})^\dagger(\mathbf{S}^T\mathbf{K}\mathbf{S})(\bar{\mathbf{C}}^T\mathbf{S})^\dagger-\tilde{\delta}^{\mathrm{ss}}(\bar{\mathbf{C}}^T\mathbf{S}\mathbf{S}^T\bar{\mathbf{C}})^\dagger.\end{aligned}$$

In this way, the time cost is merely $\mathcal{O}(s^2c)$. However, the theoretical properties of this model are yet unknown. We do not conduct theoretical or empirical study of this model; we leave it as a future work,

4. Theories

In Section 4.1 we show that the prototype model is exact when $\mathbf{K}$ is low-rank. In Section 4.2 we provide a lower error bound of the prototype model.

4.1 Theoretical Justifications

Let $\mathbf{P}$ be a column selection matrix, $\mathbf{C}=\mathbf{K}\mathbf{P}$ be a sketch, and $\mathbf{W}=\mathbf{P}^T\mathbf{K}\mathbf{P}$ be the corresponding submatrix. Kumar et al. (2009); Talwalkar and Rostamizadeh (2010) showed that the Nyström method is exact when $\mathrm{rank}(\mathbf{W})=\mathrm{rank}(\mathbf{K})$. We present a similar result in Theorem 1.

Theorem 1 *The following three statements are equivalent:* (i) $\mathrm{rank}(\mathbf{W})=\mathrm{rank}(\mathbf{K})$, (ii) $\mathbf{K}=\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T$, (iii) $\mathbf{K}=\mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{C}^T$.

Theorem 1 implies that the prototype model and the Nyström method are equivalent when $\mathrm{rank}(\mathbf{W})=\mathrm{rank}(\mathbf{K})$; that is, the kernel matrix $\mathbf{K}$ is low rank. However, it holds in general that $\mathrm{rank}(\mathbf{K})\gg c\ge\mathrm{rank}(\mathbf{W})$, where the two models are not equivalent.

4.2 Lower Bound

Wang and Zhang (2013) showed that with $c = \mathcal{O}(k/\epsilon^2)$ columns chosen by the near-optimal+adaptive sampling algorithm, the prototype model satisfies $1+\epsilon$ relative-error bound. We establish a lower error bound in Theorem 2, which shows that at least $c \geq 2k\epsilon^{-1}$ columns must be chosen to attain the $1+\epsilon$ bound. This indicates there exists a gap between the upper bound in (Wang and Zhang, 2013) and our lower bound, and thus there is room of improvement. The proof of Theorem 2 is left to Appendix B.

Theorem 2 (Lower Bound of the Prototype Model) *Whatever column sampling algorithm is used, there exists an $n \times n$ SPSD matrix $\mathbf{K}$ such that the error incurred by the prototype model obeys:*

$$\big\|\mathbf{K} - \mathbf{C}\mathbf{U}^\star\mathbf{C}^T\big\|_F^2 \;\geq\; \frac{n-c}{n-k}\Big(1+\frac{2k}{c}\Big)\|\mathbf{K}-\mathbf{K}_k\|_F^2.$$

Here k is an arbitrary target rank, c is the number of selected columns, and $\mathbf{U}^\star = \mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T$.

5. Column Sampling Algorithms

In Section 5.1 we introduce the column sampling algorithms in the literature. In Section 5.2 we improve the bound of the near-optimal+adaptive sampling algorithm (Wang and Zhang, 2013), and the obtained upper bound matches the lower bound up to a constant factor. In Section 5.3 we develop a more efficient column sampling algorithm which we call the uniform+adaptive^2 algorithm. In Section 5.4 we discuss the possibility of making uniform+adaptive^2 more scalable.

5.1 Related Work

Column selection is an important matrix sketching approach that enables expensive matrix computations to be performed on much a smaller matrix. The column selection problem has been widely studied in the theoretical computer science community (Boutsidis et al., 2014; Mahoney, 2011; Guruswami and Sinop, 2012; Woodruff, 2014) and the numerical linear algebra community (Gu and Eisenstat, 1996; Stewart, 1999), and numerous algorithms have been devised and analyzed. Here we focus on some provable algorithms studied in the theoretical computer science community.

The adaptive sampling algorithm devised by Deshpande et al. (2006) (see Algorithm 2) is the most relevant to this paper. The adaptive sampling algorithm has strong error bound (Deshpande et al., 2006; Wang and Zhang, 2013; Boutsidis et al., 2014) and good empirical performance (Kumar et al., 2012). Particularly, Wang and Zhang (2013) proposed an algorithm that combines the near-optimal column sampling algorithm (Boutsidis et al., 2014) and the adaptive sampling algorithm (Deshpande et al., 2006). They showed that by selecting $c = \mathcal{O}(k\epsilon^{-2})$ columns of $\mathbf{K}$ to form $\mathbf{C}$, it holds that

$$\mathbb{E}\big\|\mathbf{K} - \mathbf{C}\big(\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\big)\mathbf{C}^T\big\|_F \;\leq\; (1+\epsilon)\|\mathbf{K}-\mathbf{K}_k\|_F.$$

This error bound was the tightest among all the feasible algorithms for SPSD matrix approximation.

Algorithm 2 The Adaptive Sampling Algorithm.

1: **Input:** a residual matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ and number of selected columns c ($< n$).
2: Compute sampling probabilities $p_j = \|\mathbf{b}_{:j}\|_2^2/\|\mathbf{B}\|_F^2$ for $j = 1, \cdots, n$;
3: Select c indices in c i.i.d. trials, in each trial the index j is chosen with probability p_j;
4: **return** an index set containing the indices of the selected columns.

5.2 Near Optimal Column Selection for SPSD Matrix Approximation

The error bound of near-optimal+adaptive can be improved by a factor of ϵ by exploiting the latest results of Boutsidis and Woodruff (2014). Using the same algorithm except for different c_2 (i.e. the number of columns selected by adaptive sampling), we obtain the following stronger theorem. Recall from Theorem 2 that the lower bound is $c \geq \Omega\big(2k\epsilon^{-1}(1+o(1))\big)$. Thus the near-optimal+adaptive algorithm is optimal up to a constant factor. The proof of the theorem is left to Appendix 3.

Theorem 3 (Near-Optimal+Adaptive) *Given a symmetric matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ and a target rank k, the algorithm samples totally $c = 3k\epsilon^{-1}\big(1+o(1)\big)$ columns of $\mathbf{K}$ to construct the approximation. Then*

$$\mathbb{E}\big\|\mathbf{K} - \mathbf{C}\big(\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\big)\mathbf{C}^T\big\|_F \;\leq\; (1+\epsilon)\|\mathbf{K}-\mathbf{K}_k\|_F.$$

The algorithm costs $\mathcal{O}\big(n^2c + nk^3\epsilon^{-2/3}\big)$ time and $\mathcal{O}(nc)$ memory in computing $\mathbf{C}$.

Despite its optimal error bound, the near-optimal+adaptive algorithm lacks of practicality. The implementation is complicated and difficult. Its main component—the near-optimal algorithm (Boutsidis et al., 2014)—is highly iterative and therefore not suitable for parallel computing. Every step of the near-optimal algorithm requires the full observation of $\mathbf{K}$ and there is no hope to avoid this. Thus we propose to use uniform sampling to replace the near-optimal algorithm. Although the obtained uniform+adaptive^2 algorithm also has quadratic time complexity and requires the full observation of $\mathbf{K}$, there may be some way to making it more efficient. See the discussions in Section 5.4.

5.3 The Uniform+Adaptive^2 Column Sampling Algorithm

In this paper we propose a column sampling algorithm which is efficient, effective, and very easy to be implemented. The algorithm consists of a uniform sampling step and two adaptive sampling steps, so we call it *the uniform+adaptive^2 algorithm*. The algorithm is described in Algorithm 3 and analyzed in Theorem 4. The proof is left to Appendix D.

It is worth mentioning that our uniform+adaptive^2 algorithm is a special instance of the adaptive-full algorithm of (Kumar et al., 2012, Figure 3). The adaptive-full algorithm consists of random initialization and multiple adaptive sampling steps. Using multiple adaptive sampling steps can surely reduce the approximation error. However, the update of sampling probability in each step is expensive, so we choose to do only two steps. The adaptive-full algorithm of (Kumar et al., 2012, Figure 3) is merely a heuristic scheme without theoretical guarantee; our result provides theoretical justification for the adaptive-full algorithm.

Algorithm 3 The Uniform+Adaptive^2 Algorithm.

1: **Input:** an $n \times n$ symmetric matrix $\mathbf{K}$, target rank k, error parameter $\epsilon \in (0, 1]$, matrix coherence μ.
2: **Uniform Sampling.** Uniformly sample
$$c_1 = 20\mu k \log\big(20k\big)$$
columns of $\mathbf{K}$ without replacement to construct $\mathbf{C}_1$;
3: **Adaptive Sampling.** Sample
$$c_2 = 17.5k/\epsilon$$
columns of $\mathbf{K}$ to construct $\mathbf{C}_2$ using the adaptive sampling algorithm (Algorithm 2) according to the residual $\mathbf{K} - \mathcal{P}_{\mathbf{C}_1}(\mathbf{K})$;
4: **Adaptive Sampling.** Sample
$$c_3 = 10k/\epsilon$$
columns of $\mathbf{K}$ to construct $\mathbf{C}_3$ using the adaptive sampling algorithm (Algorithm 2) according to the residual $\mathbf{K} - \mathcal{P}_{[\mathbf{C}_1,\, \mathbf{C}_2]}(\mathbf{K})$;
5: **return** $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]$.

Theorem 4 (Uniform+Adaptive^2) *Given an $n \times n$ symmetric matrix $\mathbf{K}$ and a target rank k, let μ_k denote the matrix coherence of $\mathbf{K}$. Algorithm 3 samples totally*

$$c = \mathcal{O}\big(k\epsilon^{-1} + \mu_k k \log k\big)$$

columns of $\mathbf{K}$ to construct the approximation. The error bound

$$\big\|\mathbf{K} - \mathbf{C}\big(\mathbf{C}^\dagger \mathbf{K} (\mathbf{C}^\dagger)^T\big)\mathbf{C}^T\big\|_F \;\leq\; \big(1+\epsilon\big)\big\|\mathbf{K} - \mathbf{K}_k\big\|_F$$

holds with probability at least 0.7. The algorithm costs $\mathcal{O}(n^2 c)$ time and $\mathcal{O}(nc)$ space in computing $\mathbf{C}$.

Remark 5 *Theoretically, Algorithm 3 requires computing the matrix coherence of $\mathbf{K}$ in order to determine c_1 and c_2. However, computing the matrix coherence is as hard as computing the truncated SVD; even the fast approximation approach of Drineas et al. (2012) is not feasible here because $\mathbf{K}$ is a square matrix. The use of the matrix coherence here is merely for theoretical analysis; setting the parameter μ in Algorithm 3 to be exactly the matrix coherence does not certainly result in the highest accuracy. Empirically, the resulting approximation accuracy is not sensitive to the value of μ. Thus we suggest setting μ in Algorithm 3 as a constant (e.g. 1), rather than actually computing the matrix coherence.*

Table 2 presents comparisons between the near-optimal+adaptive algorithm and our uniform+adaptive^2 algorithm over the time cost, memory cost, number of passes through $\mathbf{K}$, the number of columns required to attain $1+\epsilon$ relative-error bound, and the hardness of implementation. Our algorithm is more time-efficient and pass-efficient than the near-optimal+adaptive algorithm, and the memory costs of the two algorithms are the same. To attain the same error bound, our algorithm needs to select $c = \mathcal{O}\big(k\epsilon^{-1} + \mu_k k \log k\big)$ columns, which is a little larger than that of the near-optimal+adaptive algorithm.

Table 2: Comparisons between the two sampling algorithms.

	Uniform+Adaptive^2	Near-Optimal+Adaptive
Time	$\mathcal{O}(n^2 c)$	$\mathcal{O}(n^2 c + nk^3\epsilon^{-2/3})$
Memory	$\mathcal{O}(nc)$	$\mathcal{O}(nc)$
#Passes	2	4
#Columns	$\mathcal{O}\big(k\epsilon^{-1} + \mu_k k \log k\big)$	$\mathcal{O}\big(k\epsilon^{-1}\big)$
Implement	Easy to implement	Hard to implement

Algorithm 4 The Incomplete Uniform+Adaptive^2 Algorithm.

1: **Input:** part of an $n \times n$ symmetric matrix $\mathbf{K}$.
2: **Uniform Sampling.** Uniformly sample c_1 columns of $\mathbf{K}$ without replacement to construct $\mathbf{C}_1$;
3: **Adaptive Sampling.** Uniformly sample $o(n)$ columns of $\mathbf{K}$ to form $\mathbf{K}'$; then sample c_2 columns of $\mathbf{K}'$ to construct $\mathbf{C}_2$ using the adaptive sampling algorithm (Algorithm 2) according to the residual $\mathbf{K}' - \mathcal{P}_{\mathbf{C}_1}(\mathbf{K}')$;
4: **Adaptive Sampling.** Uniformly sample $o(n)$ columns of $\mathbf{K}$ to form $\mathbf{K}''$; then sample c_3 columns of $\mathbf{K}''$ to construct $\mathbf{C}_3$ using the adaptive sampling algorithm (Algorithm 2) according to the residual $\mathbf{K}'' - \mathcal{P}_{[\mathbf{C}_1,\, \mathbf{C}_2]}(\mathbf{K}'')$;
5: **return** $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3]$.

5.4 Discussions

The two algorithms discussed in the above have strong theoretical guarantees, but they are not efficient enough for large-scale applications. First, their time complexities are quadratic in n. Second, they require the full observation of $\mathbf{K}$. In fact, at present no existing column selection algorithm satisfies the three properties simultaneously:

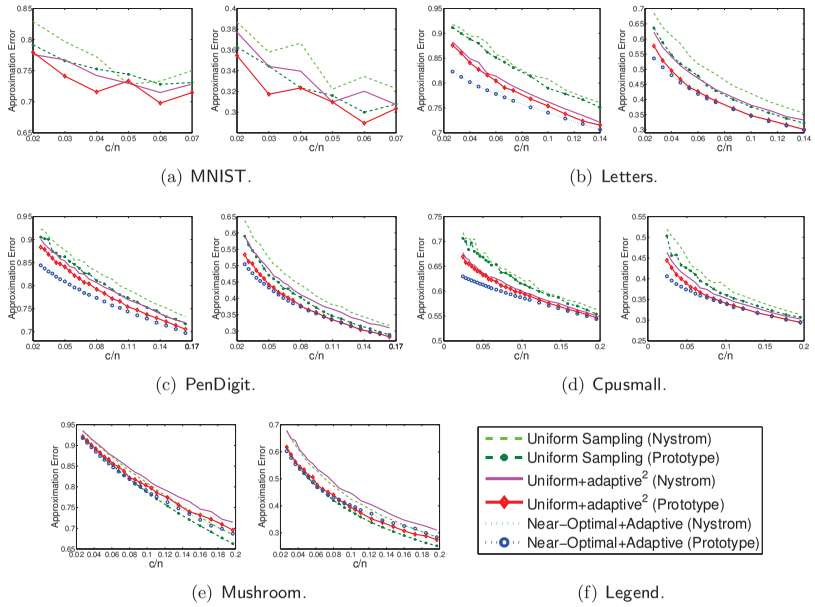
1. the time and memory costs are $\mathcal{O}(n)$;
2. only $\mathcal{O}(n)$ entries of $\mathbf{K}$ need to be observed;
3. relative-error bound holds in expectation or with high probability.

It is interesting to find such an algorithm, and it remains an open problem.

Nevertheless, uniform+adaptive^2 is a promising column selection algorithm for it may be adapted to satisfy the above three properties. The drawback of uniform+adaptive^2 is that computing the adaptive sampling probability costs quadratic time and requires the full observation of $\mathbf{K}$. There may be remedies for this problem. The adaptive-partial algorithm in (Kumar et al., 2012) satisfies the first two properties, but it lacks theoretical analysis. Another possibility is to first uniformly sample $o(n)$ columns and then down-sample to $\mathcal{O}(k/\epsilon)$ columns by adaptive sampling, which we describe in Algorithm 4. In this way, the first two properties can be satisfied, and it may be theoretically explained under the incoherent matrix assumption. We do not implement such heuristics for their theoretical property is completely unknown and they are beyond the scope of this paper.

Table 3: A summary of the datasets for kernel approximation.

Dataset	MNIST	Letters	PenDigit	Cpusmall	Mushrooms
#Instance	60,000	15,000	10,992	8,192	8,124
#Attribute	780	16	16	12	112
γ ($\eta = 0.5$)	≈ 1.50	0.155	0.045	0.031	0.850
γ ($\eta = 0.9$)	≈ 2.30	0.290	0.073	0.057	1.140



(a) MNIST. (b) Letters.

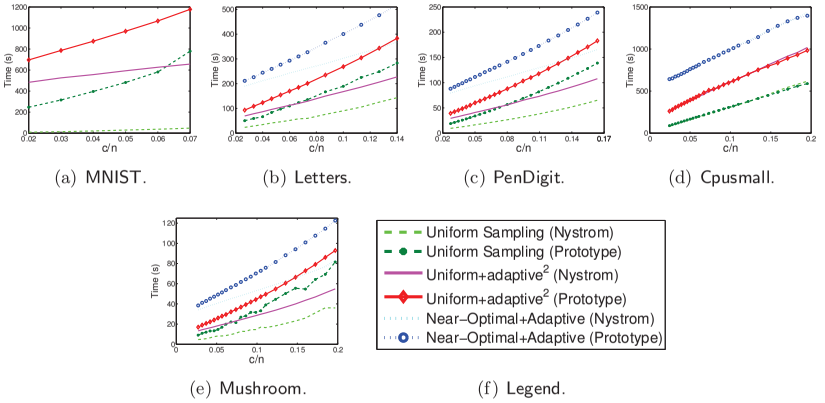
(c) PenDigit. (d) Cpusmall.

(e) Mushroom. (f) Legend.

Figure 1: The ratio $\frac{c}{n}$ against the error ratio defined in (8). In each subfigure, the left corresponds to the RBF kernel matrix with $\eta = 0.5$, and the right corresponds to $\eta = 0.9$, where η is defined in (7).

6. Experiments on the Column Sampling Algorithms

We empirically conduct comparison among three column selection algorithms—uniform sampling, uniform + adaptive^2, and the near-optimal + adaptive sampling algorithm.



(a) MNIST. (b) Letters. (c) PenDigit. (d) Cpusmall.

(e) Mushroom. (f) Legend.

Figure 2: The growth of the average elapsed time in $\frac{c}{n}$.

6.1 Experiment Setting

We perform experiments on several datasets collected on the LIBSVM website[2] where the data are scaled to [0,1]. We summarize the datasets in Table 3.

For each dataset, we generate a radial basis function (RBF) kernel matrix $\mathbf{K}$ defined by $k_{ij} = \exp(-\frac{1}{2\gamma^2}\|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$. Here $\gamma > 0$ is the scaling parameter; the larger the scaling parameter γ is, the faster the spectrum of the kernel matrix decays (Gittens and Mahoney, 2013). The previous work has shown that for the same dataset, with different settings of γ, the sampling algorithms have very different performances. Instead of setting γ arbitrarily, we set γ in the following way.

Letting $p = \lceil 0.05n \rceil$, we define

$$\eta \triangleq \frac{\sum_{i=1}^{p} \lambda_i^2(\mathbf{K})}{\sum_{i=1}^{n} \lambda_i^2(\mathbf{K})} = \frac{\|\mathbf{K}_p\|_F^2}{\|\mathbf{K}\|_F^2}, \tag{7}$$

which denotes the ratio of the top 5% eigenvalues of the kernel matrix $\mathbf{K}$ to the all eigenvalues. In general, a large γ results in a large η. For each dataset, we use two different settings of γ such that $\eta = 0.5$ or $\eta = 0.9$.

The models and algorithms are all implemented in MATLAB. We run the algorithms on a workstation with Intel Xeon 2.40GHz CPUs, 24GB memory, and 64bit Windows Server 2008 system. To compare the running time, we set MATLAB in single thread mode by the command "maxNumCompThreads(1)". In the experiments we do not keep $\mathbf{K}$ in memory. We use a variant of Algorithm 1—we compute and store one block, instead of one column, of $\mathbf{K}$ at a time. We keep at most 1,000 columns of $\mathbf{K}$ in memory at a time.

2. http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/

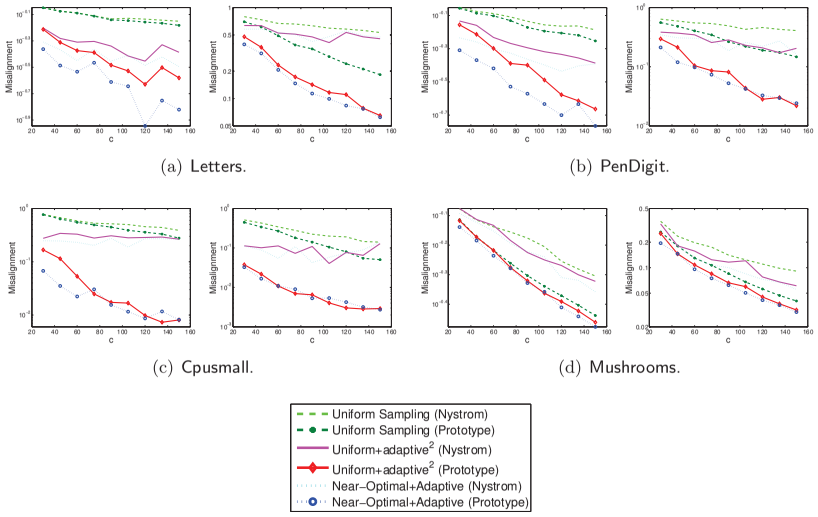
We set the target rank k to be $k = \lceil n/100 \rceil$ in all the experiments unless otherwise specified. We evaluate the performance by

$$\text{Approximation Error} \;=\; \|\mathbf{K} - \tilde{\mathbf{K}}\|_F / \|\mathbf{K}\|_F, \tag{8}$$

where $\tilde{\mathbf{K}}$ is the approximation generated by each method.

To evaluate the quality of the approximate rank-k eigenvalue decomposition, we use *misalignment* to indicate the distance between the true eigenvectors $\mathbf{U}_k$ $(n \times k)$ and the approximate eigenvectors $\tilde{\mathbf{V}}_k$ $(n \times k)$:

$$\text{Misalignment} \;=\; \frac{1}{k}\left\|\mathbf{U}_k - \tilde{\mathbf{V}}_k \tilde{\mathbf{V}}_k^T \mathbf{U}_k\right\|_F^2 \;\in\; [0,1]. \tag{9}$$



(a) Letters. (b) PenDigit.

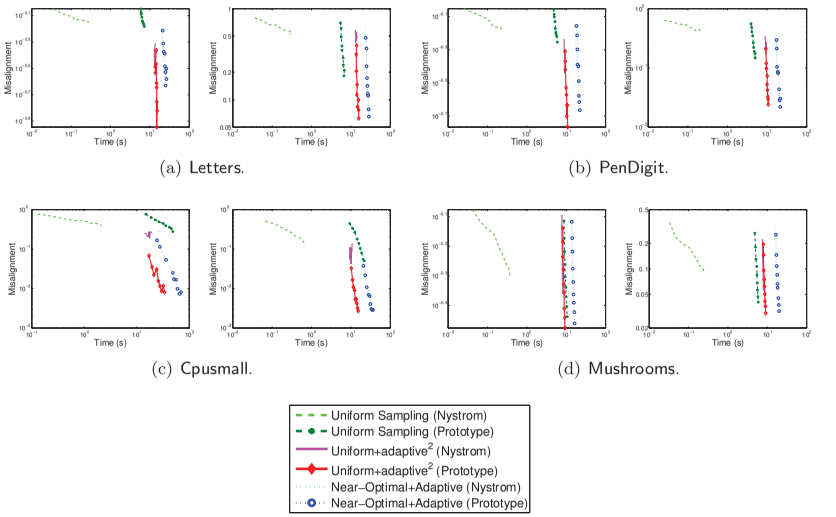
(c) Cpusmall. (d) Mushrooms.

(e) Legend.

Figure 3: The number of selected columns c against the misalignment (log-scale) defined in (9). In each subfigure, the left corresponds to the RBF kernel matrix with $\eta = 0.5$, and the right corresponds to $\eta = 0.9$, where η is defined in (7).

6.2 Matrix Approximation Accuracy

In the first set of experiments, we compare the matrix approximation quality using the Frobenius norm approximation error defined in (8) as the metric.



(a) Letters. (b) PenDigit.

(c) Cpusmall. (d) Mushrooms.

(e) Legend.

Figure 4: The elapsed time (log-scale) against the misalignment (log-scale) defined in (9). In each subfigure, the left corresponds to the RBF kernel matrix with $\eta = 0.5$, and the right corresponds to $\eta = 0.9$, where η is defined in (7).

Every time when we do column sampling, we repeat each sampling algorithm 10 times and record the minimal approximation error of the 10 repeats. We report the average elapsed time of the 10 repeat rather than the total elapsed time because the 10 repeats can be done in parallel on 10 machines. We plot c against the approximation error in Figure 1. For the kernel matrices with $\eta = 0.9$, we plot c against the average elapsed time in Figure 2; for $\eta = 0.5$, the curve of the running time is very similar, so we do not show it.

The results show that our uniform+adaptive^2 algorithm achieves accuracy comparable with the near-optimal+adaptive algorithm. Especially, when c is large, these two algorithms have virtually the same accuracy, which agrees with our analysis: a large c implies a small error term ϵ, and the error bounds of the two algorithms coincide when ϵ is small. As for the running time, we can see that our uniform+adaptive^2 algorithm is much more efficient than the near-optimal+adaptive algorithm.

Particularly, the MNIST dataset has $60,000$ instances, and the $60,000 \times 60,000$ kernel matrix $\mathbf{K}$ does not fit in memory. The experiment shows that neither the prototype model nor the uniform+adaptive^2 algorithm require keeping $\mathbf{K}$ in memory.

6.3 Kernel Principal Component Analysis

In the second set of experiment, we apply the kernel approximation methods to approximately compute the rank $k = 3$ eigenvalue decomposition of the RBF kernel matrix. We use the *misalignment* defined in (9) as the metric, which reflects the distance between the true and the approximate eigenvectors. We report the average misalignment of the 20 repeats. We do not conduct experiments on the MNIST dataset because the true eigenvectors are too expensive to compute.

To evaluate the memory efficiency, we plot c against the misalignment in Figure 3. The results show that the two non-uniform sampling algorithms are significantly better than uniform sampling. The performance of our uniform+adaptive^2 algorithm is nearly the same to the near-optimal+adaptive algorithm.

To evaluate the time efficiency, we plot the elapsed time against the misalignment in Figure 4. Though the uniform sampling algorithm is the most efficient in most cases, its accuracy is unsatisfactory. In terms of time efficiency, the uniform+adaptive^2 algorithm is better than the near-optimal+adaptive algorithm.

The experiment on the kernel PCA shows that the prototype model with the uniform + adaptive^2 column sampling algorithm achieves the best performance. Though the Nyström method with uniform sampling is the most efficient, its resulting misalignment is worse by an order of magnitude. Therefore, when applied to speedup eigenvalue decomposition, the Nyström method may not be a good choice, especially when high accuracy is required.

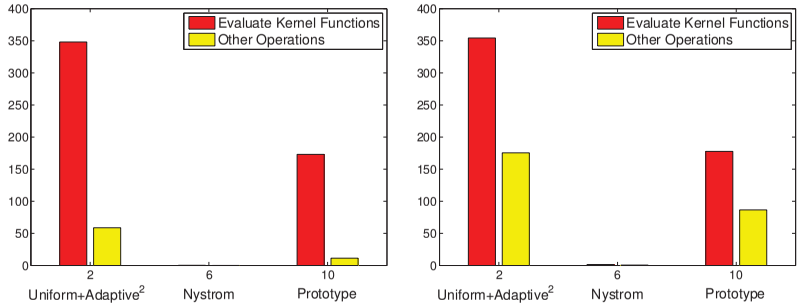
6.4 Separating the Time Costs

Besides the total elapsed time, the readers may be interested in the time cost of each step, especially when the data do no fit in memory. We run the Nyström method and the prototype model, each with uniform+adaptive^2 column sampling algorithm, on the MNIST dataset with $\gamma = 2.3$ (see Table 3). Notice that the $60,000 \times 60,000$ kernel matrix does not fit in memory, so we keep at most $1,000$ columns of the kernel matrix in memory at a time. We set $c = 50$ or 500 and repeat the procedure 20 times and record the average elapsed time. In Figure 5 we separately show the time costs of the uniform+adaptive^2 algorithm and the computation of the intersection matrices. In addition, we separate the time costs of evaluating the kernel functions and all the other computations (e.g. SVD of $\mathbf{C}$ and matrix multiplications).

We can see from Figure 5 that when $\mathbf{K}$ does not fit in memory, the computation of the kernel matrix contributes to the most of the computations. By comparing the two subfigures in Figure 5, we can see that as c increases, the costs of computing the kernel matrix barely change, but the costs of other matrix operations significantly increase.

7. The Spectral Shifting Model

All the low-rank approximation methods work well only when the bottom eigenvalues of $\mathbf{K}$ are near zero. In this section we develop extensions of the three SPSD matrix approximation models to tackle matrices with relatively big bottom eigenvalues. We call the proposed method the *spectral shifting (SS) model* and describe it in Algorithm 5. We show that the SS model has stronger error bound than the prototype model.



(a) $c = 50$ (b) $c = 500$

Figure 5: The time costs (s) of the uniform+adaptive^2 algorithm and the computation of the intersection matrices.

Algorithm 5 The Spectral Shifting Method.

Input: an $n \times n$ SPSD matrix $\mathbf{K}$, a target rank k, the number of sampled columns c, the oversampling parameter l.
// (optional) approximately do the initial spectral shifting
$\mathbf{\Omega} \longleftarrow n \times l$ standard Gaussian matrix;
$\mathbf{Q} \longleftarrow$ the l orthonormal basis of $\mathbf{K\Omega} \in \mathbb{R}^{n \times l}$;
$s \longleftarrow$ sum of the top k singular values of $\mathbf{Q}^T\mathbf{K} \in \mathbb{R}^{l \times n}$;
$\tilde{\delta} = \frac{1}{n-k}\big(\mathrm{tr}(\mathbf{K}) - s\big) \approx \bar{\delta}$;
$\bar{\mathbf{K}} \leftarrow \mathbf{K} - \tilde{\delta}\mathbf{I}_n \in \mathbb{R}^{n \times n}$;
// perform sketching, e.g. random projection or column selection
$\bar{\mathbf{C}} = \bar{\mathbf{K}}\mathbf{P}$, where $\mathbf{P}$ is an $n \times c$ random projection or selection matrix;
Optional: replace $\bar{\mathbf{C}}$ by its orthonormal bases;
// compute the spectral shifting parameter and the intersection matrix
$\delta^{\mathrm{ss}} \longleftarrow \frac{1}{n-\mathrm{rank}(\bar{\mathbf{C}})}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}\bar{\mathbf{C}}\big)\Big)$;
$\mathbf{U}^{\mathrm{ss}} \longleftarrow \bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T - \delta^{\mathrm{ss}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger$;
return the approximation $\tilde{\mathbf{K}}_c^{\mathrm{ss}} = \bar{\mathbf{C}}\mathbf{U}^{\mathrm{ss}}\bar{\mathbf{C}}^T + \delta^{\mathrm{ss}}\mathbf{I}_n$.

In Section 7.1 we formulate the SS model. In Section 7.2 we study SS from an optimization perspective. In Section 7.3 we show that SS has better error bound than the prototype model. Especially, with the near-optimal+adaptive column sampling algorithm, SS demonstrates much stronger error bound than the existing matrix approximation methods. In Section 7.4 we provide an efficient algorithm for computing the initial spectral shifting term. In Section 7.5 we discuss how to combine spectral shifting with other kernel approximation methods.

7.1 Model Formulation

The spectral shifting (SS) model is defined by

$$\tilde{\mathbf{K}}_c^{\mathrm{ss}} \;=\; \bar{\mathbf{C}}\mathbf{U}^{\mathrm{ss}}\bar{\mathbf{C}}^T + \delta^{\mathrm{ss}}\mathbf{I}_n. \tag{10}$$

Here $\delta^{\mathrm{ss}} \geq 0$ is called the spectral shifting term. This approximation is computed in three steps. Firstly, (approximately) compute the initial spectral shifting term

$$\bar{\delta} \;=\; \frac{1}{n-k}\Big(\mathrm{tr}(\mathbf{K}) - \sum_{j=1}^{k}\sigma_j(\mathbf{K})\Big), \tag{11}$$

and then perform spectral shifting $\bar{\mathbf{K}} = \mathbf{K} - \bar{\delta}\mathbf{I}_n$, where $k \leq c$ is the target rank. This step is optional. Due to Theorem 6 and Remark 7, SS is better than the prototype model even if $\bar{\delta} = 0$; however, without this step, the quantity of the improvement contributed by SS is unknown. Secondly, draw a column selection matrix $\mathbf{P}$ and form the sketch $\bar{\mathbf{C}} = \bar{\mathbf{K}}\mathbf{P}$. Finally, with $\bar{\mathbf{C}}$ at hand, compute δ^{ss} and $\mathbf{U}^{\mathrm{ss}}$ by

$$\begin{aligned}\delta^{\mathrm{ss}} \;&=\; \frac{1}{n-\mathrm{rank}(\bar{\mathbf{C}})}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}\bar{\mathbf{C}}\big)\Big),\\ \mathbf{U}^{\mathrm{ss}} \;&=\; \bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T - \delta^{\mathrm{ss}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger.\end{aligned} \tag{12}$$

We will show that $\tilde{\mathbf{K}}_c^{\mathrm{ss}}$ is positive (semi)definite if $\mathbf{K}$ is positive (semi)definite.

However, when the bottom eigenvalues of $\mathbf{K}$ are small, the computed spectral shifting term is small, where there is little difference between SS and the prototype model, and the spectral shifting operation is not advised.

7.2 Optimization Perspective

The SS model is an extension of the prototype model from the optimization perspective. Given an SPSD matrix $\mathbf{K}$, the prototype model computes the sketch $\mathbf{C} = \mathbf{K}\mathbf{P}$ and the intersection matrix

$$\mathbf{U}^\star \;=\; \mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T \;=\; \operatorname*{argmin}_{\mathbf{U}} \|\mathbf{K} - \mathbf{C}\mathbf{U}\mathbf{C}^T\|_F^2. \tag{13}$$

Analogously, with the sketch $\bar{\mathbf{C}} = \bar{\mathbf{K}}\mathbf{P} = \mathbf{K}\mathbf{P} - \bar{\delta}\mathbf{P}$ at hand, SS is obtained by solving

$$\big(\mathbf{U}^{\mathrm{ss}}, \delta^{\mathrm{ss}}\big) \;=\; \operatorname*{argmin}_{\mathbf{U},\delta} \big\|\mathbf{K} - \bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T - \delta\mathbf{I}_n\big\|_F^2, \tag{14}$$

obtaining the intersection matrix $\mathbf{U}^{\mathrm{ss}}$ and the spectral shifting term δ^{ss}. By analyzing the optimization problem (14), we obtain the following theorem. Its proof is in Appendix E.

Theorem 6 *The pair $(\delta^{ss}, \mathbf{U}^{ss})$ defined in (12) is the global minimizer of problem (14), which indicates that using any other $(\delta, \mathbf{U})$ to replace $(\delta^{ss}, \mathbf{U}^{ss})$ results in a larger approximation error. Furthermore, if $\mathbf{K}$ is positive (semi)definite, then the approximation $\bar{\mathbf{C}}\mathbf{U}^{ss}\bar{\mathbf{C}}^T + \delta^{ss}\mathbf{I}_n$ is also positive (semi)definite.*

Remark 7 *The optimization perspective indicates the superiority of SS. Suppose we skip the initial spectral shifting step and simply set $\bar{\delta} = 0$. Then $\bar{\mathbf{C}} = \mathbf{C}$. If the constraint $\delta = 0$ is to the optimization problem (14), then (14) will become identical to the prototype model (13). Obviously, adding this constraint will make the optimal objective function value get worse, so the optimal objective function value of (14) is always less than or equal to (13). Hence, without the initial spectral shifting step, SS is still more accurate than the prototype model.*

7.3 Error Analysis

The following theorem indicates that the SS model with any spectral shifting term $\delta \in (0, \bar{\delta}]$ has a stronger bound than the prototype model. The proof is in Appendix F.

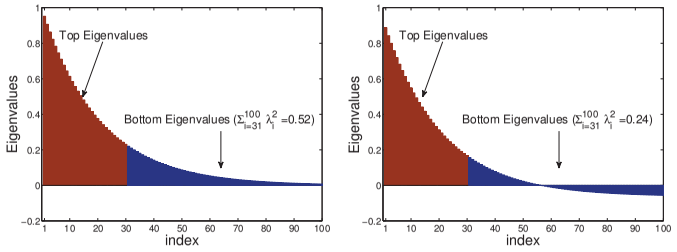
Theorem 8 *Suppose there is a sketching matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$ such that for any $n \times n$ symmetric matrix $\mathbf{A}$ and target rank k ($\ll n$), by forming $\mathbf{C} = \mathbf{A}\mathbf{P}$, the prototype model satisfies the error bound*

$$\big\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T\mathbf{C}^T\big\|_F^2 \;\leq\; \eta\,\big\|\mathbf{A} - \mathbf{A}_k\big\|_F^2$$

for certain $\eta > 0$. Let $\mathbf{K}$ be any $n \times n$ SPSD matrix, $\tilde{\delta} \in (0, \bar{\delta}]$ be the initial spectral shifting term where $\bar{\delta}$ is defined in (11), $\bar{\mathbf{K}} = \mathbf{K} - \tilde{\delta}\mathbf{I}_n$, $\bar{\mathbf{C}} = \bar{\mathbf{K}}\mathbf{P}$, and $\tilde{\mathbf{K}}_c^{ss}$ be the SS model defined in (10). Then

$$\big\|\mathbf{K} - \tilde{\mathbf{K}}_c^{ss}\big\|_F^2 \;\leq\; \eta\,\big\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\big\|_F^2 \;\leq\; \eta\,\big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2.$$

We give an example in Figure 6 to illustrate the intuition of spectral shifting. We use the toy matrix $\mathbf{K}$: an $n\times n$ SPSD matrix whose the t-th eigenvalue is 1.05^{-t}. We set $n = 100$ and $k = 30$, and hence $\bar{\delta} = 0.064$. From the plot of the eigenvalues we can see that the "tail" of the eigenvalues becomes thinner after the spectral shifting. Specifically, $\|\mathbf{K} - \mathbf{K}_k\|_F^2 = 0.52$ and $\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\|_F^2 \leq 0.24$. From Theorem 8 we can see that if $\|\mathbf{K} - \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{C}^T\|_F \leq 0.52\eta$, then $\|\mathbf{K} - \tilde{\mathbf{K}}_c^{\mathrm{ss}}\|_F \leq 0.24\eta$. This indicates that SS has much stronger error bound than the prototype model.



(a) Before spectral shifting. (b) After spectral shifting.

Figure 6: We plot the eigenvalues of $\mathbf{K}$ in Figure 6(a) and $\bar{\mathbf{K}} = \mathbf{K} - \bar{\delta}\mathbf{I}_{100}$ in Figure 6(b).

The following theorem shows an error bound of the SS model, which is stronger than the prototype model, especially when the bottom $n-k$ eigenvalues of $\mathbf{K}$ are big. The proof is in Appendix F.

Theorem 9 *Suppose there is a sketching matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$ such that for any $n \times n$ symmetric matrix $\mathbf{A}$ and target rank k ($\ll n$), by forming the sketch $\mathbf{C} = \mathbf{AP}$, the prototype model satisfies the error bound*

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^\dagger)^T\mathbf{C}^T\right\|_F^2 \;\le\; \eta\left\|\mathbf{A}-\mathbf{A}_k\right\|_F^2$$

for certain $\eta > 0$. Let $\mathbf{K}$ be any $n \times n$ SPSD matrix, $\bar{\delta}$ defined in (11) be the initial spectral shifting term, and $\tilde{\mathbf{K}}_c^{ss}$ be the SS model defined in (10). Then

$$\left\|\mathbf{K} - \tilde{\mathbf{K}}_c^{ss}\right\|_F^2 \;\le\; \eta\left(\left\|\mathbf{K}-\mathbf{K}_k\right\|_F^2 - \frac{\left[\sum_{i=k+1}^n \lambda_i(\mathbf{K})\right]^2}{n-k}\right).$$

If $\bar{\mathbf{C}}$ contains the columns of $\bar{\mathbf{K}}$ sampled by the near-optimal+adaptive algorithm in Theorem 3, which has the strongest bound, then the error bound incurred by SS is given in the following corollary.

Corollary 10 *Suppose we are given any SPSD matrix $\mathbf{K}$ and we sample $c = \mathcal{O}(k/\epsilon)$ columns of $\bar{\mathbf{K}}$ to form $\bar{\mathbf{C}}$ using the near-optimal+adaptive column sampling algorithm (Theorem 3). Then the inequality holds:*

$$\mathbb{E}\left\|\mathbf{K} - \tilde{\mathbf{K}}_c^{ss}\right\|_F^2 \le (1+\epsilon)\left(\left\|\mathbf{K}-\mathbf{K}_k\right\|_F^2 - \frac{\left[\sum_{i=k+1}^n \lambda_i(\mathbf{K})\right]^2}{n-k}\right).$$

Here we give an example to demonstrate the superiority of SS over the prototype model, the Nyström method, and even the truncated SVD of the same scale.

Example 1 *Let $\mathbf{K}$ be an $n \times n$ SPSD matrix such that $\lambda_1(\mathbf{K}) \ge \cdots \ge \lambda_k(\mathbf{K}) > \theta = \lambda_{k+1}(\mathbf{K}) = \cdots = \lambda_n(\mathbf{K}) > 0$. By sampling $c = \mathcal{O}(k)$ columns by the near-optimal+adaptive algorithm (Theorem 3), we have that*

$$\left\|\mathbf{K} - \tilde{\mathbf{K}}_c^{ss}\right\|_F^2 \;=\; 0$$

and that

$$(n-c)\theta^2 \;=\; \left\|\mathbf{K}-\mathbf{K}_c\right\|_F^2 \;\le\; \left\|\mathbf{K}-\tilde{\mathbf{K}}_c^{proto}\right\|_F^2 \;\le\; \left\|\mathbf{K}-\tilde{\mathbf{K}}_c^{nys}\right\|_F^2.$$

Here $\tilde{\mathbf{K}}_c^{proto}$ and $\tilde{\mathbf{K}}_c^{nys}$ respectively denote the approximation formed by the prototype model and the Nyström method. In this example the SS model is far better than the other models if we set θ as a large constant.

7.4 Approximately Computing $\bar{\delta}$

The SS model uses $\bar{\delta}$ as the initial spectral shifting term. However, computing $\bar{\delta}$ according to (11) requires the partial eigenvalue decomposition which costs $\mathcal{O}(n^2k)$ time and $\mathcal{O}(n^2)$ memory. For large-scale data, one can simply set $\bar{\delta} = 0$; Remark 7 shows that SS with this setting still works better than the prototype model. For medium-scale data, one can approximately compute $\bar{\delta}$ by the algorithm devised and analyzed in this subsection.

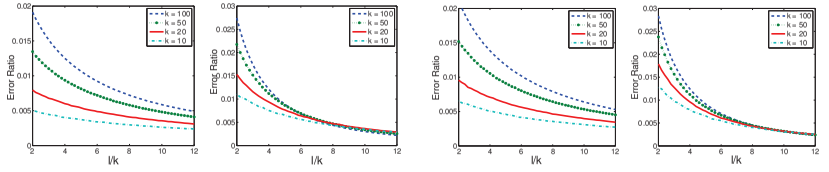
We depict the algorithm in Lines 2–6 of Algorithm 5. The performance of the approximation is analyzed in the following theorem.

Theorem 11 *Let $\bar{\delta}$ be defined in (11) and $\tilde{\delta}$, k, l, n be defined in Algorithm 5. The following inequality holds:*

$$\mathbb{E}\left[\left|\bar{\delta}-\tilde{\delta}\right| / \bar{\delta}\right] \;\le\; k/\sqrt{l},$$

where the expectation is taken w.r.t. the Gaussian random matrix $\mathbf{\Omega}$ in Algorithm 5. Lines 2–6 in Algorithm 5 compute $\tilde{\delta}$ in $\mathcal{O}(n^2l)$ time and $\mathcal{O}(nl)$ memory.

Here we empirically evaluate the accuracy of the approximation to $\bar{\delta}$ (Lines 2–6 in Algorithm 5) proposed in Theorem 11. We use the RBF kernel matrices with the scaling parameter γ listed Table 3. We use the error ratio $|\bar{\delta}-\tilde{\delta}|/\bar{\delta}$ to evaluate the approximation quality. We repeat the experiments 20 times and plot l/k against the average error ratio in Figure 7. Here $\tilde{\delta}$, l, and k are defined in Theorem 11. We can see that the approximation of $\bar{\delta}$ has high quality: when $l = 4k$, the error ratios are less than 0.03 in all cases, no matter whether the spectrum of $\mathbf{K}$ decays fast or slow.



(a) Letters (b) PenDigit

Figure 7: The ratio $\frac{l}{k}$ against the error $|\bar{\delta}-\tilde{\delta}|/\bar{\delta}$. In each subfigure, the left corresponds to the RBF kernel matrix with $\eta = 0.5$, and the right corresponds to $\eta = 0.9$, where η is defined in (7).

7.5 Combining with Other Matrix Approximation Methods

There are many other matrix approximation approaches such as the ensemble Nyström method (Kumar et al., 2012) and MEKA (Si et al., 2014). In fact, the key components of the ensemble Nyström method and MEKA are the Nyström method, which can be straightforwardly replaced by other matrix approximation methods such as the SS model.

The ensemble Nyström method improves the Nyström method by running the Nyström method t times and combine the samples to construct the kernel approximation:

$$\tilde{\mathbf{K}}^{\text{ens}}_{t,c} = \sum_{i=1}^{t} \mu^{(i)} \mathbf{C}^{(i)} \mathbf{W}^{(i)\dagger} \mathbf{C}^{(i)T},$$

where $\mu^{(1)}, \cdots, \mu^{(t)}$ are the weights of the samples, and a simple but effective strategy is to set the weights as $\mu^{(1)} = \cdots = \mu^{(t)} = \frac{1}{t}$. However, the time and memory costs of computing $\mathbf{C}$ and $\mathbf{U}$ are respectively t times as much as that of its base method (e.g. the Nyström method), and the ensemble Nyström method needs to use the Sherman-Morrison-Woodbury formula t times to combine the samples. When executed on a single machine, the accuracy gained by the ensemble may not worth the t times more time and memory costs.

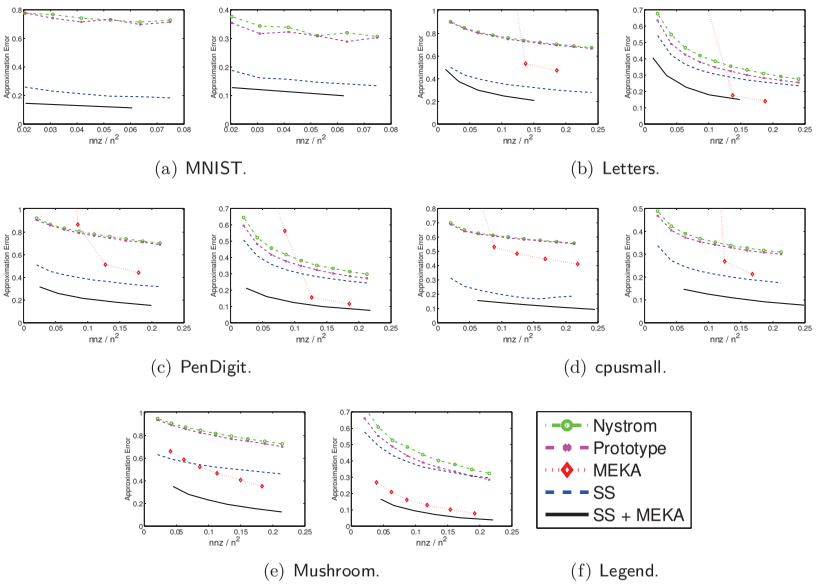
MEKA is reported to be the state-of-the-art kernel approximation method. It exploits the block-diagonal structure of kernel matrices, and outputs an $n \times c$ sparse matrix $\mathbf{C}$ and a $c \times c$ small matrix $\mathbf{U}$ such that $\mathbf{K} \approx \mathbf{C}\mathbf{U}\mathbf{C}^T$. MEKA first finds the blocks by clustering the data into b clusters and permutes the kernel matrix accordingly. It then approximates the diagonal blocks by the Nyström method, *which can be replaced by other methods.* It finally approximates the off-diagonal blocks using the diagonal blocks. If the kernel matrix is partitioned into b^2 blocks, only b blocks among the b^2 blocks of $\mathbf{C}$ are nonzero, and the number of nonzero entries of $\mathbf{C}$ is at most $\text{nnz}(\mathbf{C}) = nc/b$. MEKA is thus much more memory efficient than the Nyström method. If we use the SS model to approximate the diagonal blocks, then the resulting MEKA approximation will be in the form $\mathbf{K} \approx \mathbf{C}\mathbf{U}\mathbf{C}^T + \delta_1 \mathbf{I} \oplus \cdots \oplus \delta_b \mathbf{I}$, where δ_i corresponds to the i-th diagonal block.

8. Empirically Evaluating the Spectral Shifting Model

We empirically evaluate the spectral shifting method by comparing the following kernel approximation models in terms of approximation quality and the generalization performance on Gaussian process regression.

- The Nyström method with the uniform+adaptive^2 algorithm.
- The prototype model with the uniform+adaptive^2 algorithm.
- The memory efficient kernel approximation (MEKA) method (Si et al., 2014), which approximates the diagonal blocks of the kernel matrix by the Nyström method. We use the code released by the authors with default settings.
- The spectral shifting (SS) model with the uniform+adaptive^2 algorithm.
- SS+MEKA: the same to MEKA except for using SS, rather than the Nyström method, to approximate the diagonal blocks.

Since the experiments are all done on a single machine, the time and memory costs of the ensemble Nyström method (Kumar et al., 2012) are t times larger. It would be unfair to directly do comparison with the ensemble Nyström method.



(a) MNIST. (b) Letters. (c) PenDigit. (d) cpusmall. (e) Mushroom. (f) Legend.

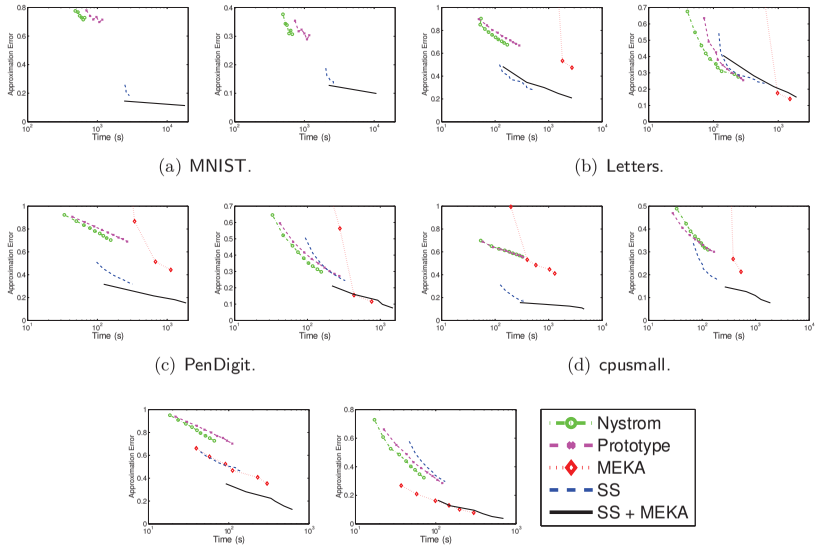
Figure 8: The memory cost against approximation error. Here "nnz" is number of nonzero entries in the sketch, namely, $\text{nnz}(\mathbf{C}) + \text{nnz}(\mathbf{U})$. In each subfigure, the left corresponds to the RBF kernel matrix with $\eta = 0.5$, and the right corresponds to $\eta = 0.9$, where η is defined in (7).

8.1 Experiments on Approximation Accuracy

We conduct experiments using the same setting as in Section 6.1. To demonstrate the effect of spectral shifting, we only consider a kernel matrix with slowly decaying spectrum. Thus we set η (defined in (7)) relatively small: $\eta = 0.5$ and $\eta = 0.9$. When η is near one, the spectral shifting term becomes near zero, and spectral shifting makes no difference.

We present the results in two ways: (1) Figure 8 plots the memory usage against the approximation error, where the memory usage is proportional to the number of nonzero entries of $\mathbf{C}$ and $\mathbf{U}$; (2) Figure 9 plots the elapsed time against the approximation error. The results have the following implications.

- Using the spectral shifting technique is better than without using it: SS and SS+MEKA are more accurate than the prototype model and MEKA, respectively. The advantage



(a) MNIST. (b) Letters.

(c) PenDigit. (d) cpusmall.

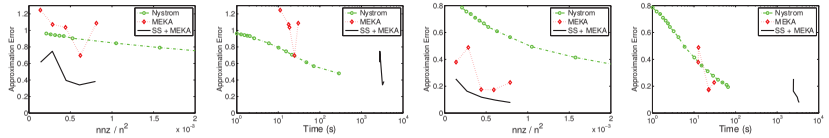
(e) Mushroom. (f) Legend.

Figure 9: The elapsed time (log-scale) against the approximation error. In each subfigure, the left corresponds to the RBF kernel matrix with $\eta = 0.5$, and the right corresponds to $\eta = 0.9$, where η is defined in (7).

of spectral shifting is particularly obvious when the spectrum of $\mathbf{K}$ decays slowly, i.e. when $\eta = 0.5$.

- SS and SS+MEKA are the best among the compared methods according to our experiments. Especially, if a high-quality approximation in limited memory is desired, SS+MEKA should be the best choice.
- The kernel matrix of the MNIST dataset is $60,000 \times 60,000$, which does not fit in the 24GB memory. This shows that the compared methods and sampling algorithm do not require keeping $\mathbf{K}$ in memory.

In addition, we conduct experiments on the large-scale dataset—covtype—which has 581,012 data instances. We compare among the Nyström method, MEKA, and MEKA+SS. The experiment setting is slightly different from the ones in the above. For each of the four methods, we use uniform sampling to select columns. As for the MEKA based methods, we set the number of clusters to be 30 (whereas the default is 10) to increase scalability. As



(a) $\eta \approx 0.5$ (b) $\eta \approx 0.9$

Figure 10: The experiments on the covtype dataset. We plot the memory cost against approximation error and plot the elapsed time (log-scale) against the approximation error.

Table 4: Summary of datasets for Gaussian process regression

	Plant	White Wine	Red Wine	Concrete	Energy (Heat)	Energy (Cool)	Housing
#Instance	9,568	4,898	1,599	1,030	768	768	506
#Attribute	4	11	11	8	8	8	13
γ	0.1	1	1	1	0.5	0.5	1
σ^2	0.1	0.01	0.01	0.0002	0.0005	0.001	0.005

for the spectral shifting method, we do not perform the initial spectral shifting. The results are plotted in Figure 10. The results show the effectiveness and scalability of the spectral shift method.

8.2 Experiments on Gaussian Process Regression

We apply the kernel approximation methods to Gaussian process regression (GPR). We assume that the training set is $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ are input vectors and $y_i \in \mathbb{R}$ are the corresponding outputs. GPR is defined as

$$y = u + f(\mathbf{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2),$$

where $f(\mathbf{x})$ follows a Gaussian process with mean function 0 and kernel function $\kappa(\cdot, \cdot)$. Furthermore, we define the kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$, where the (i, j)-th entry of $\mathbf{K}$ is $\kappa(\mathbf{x}_i, \mathbf{x}_j)$.

For a test input $\mathbf{x}_*$, the prediction is given by

$$\hat{y}_* = \mathbf{k}^T(\mathbf{x}_*)(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y},$$

where $\mathbf{y} = [y_1, \ldots, y_n]^T$ and $\mathbf{k}(\mathbf{x}_*) = [\kappa(\mathbf{x}_*, \mathbf{x}_1), \ldots, \kappa(\mathbf{x}_*, \mathbf{x}_n)]^T$. We apply different kernel approximation methods to approximate $\mathbf{K}$ and approximately compute $(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}$ according to Section 3.1. We evaluate the generalization performance using the mean squared error:

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y_{i*} - \hat{y}_{i*})^2,$$

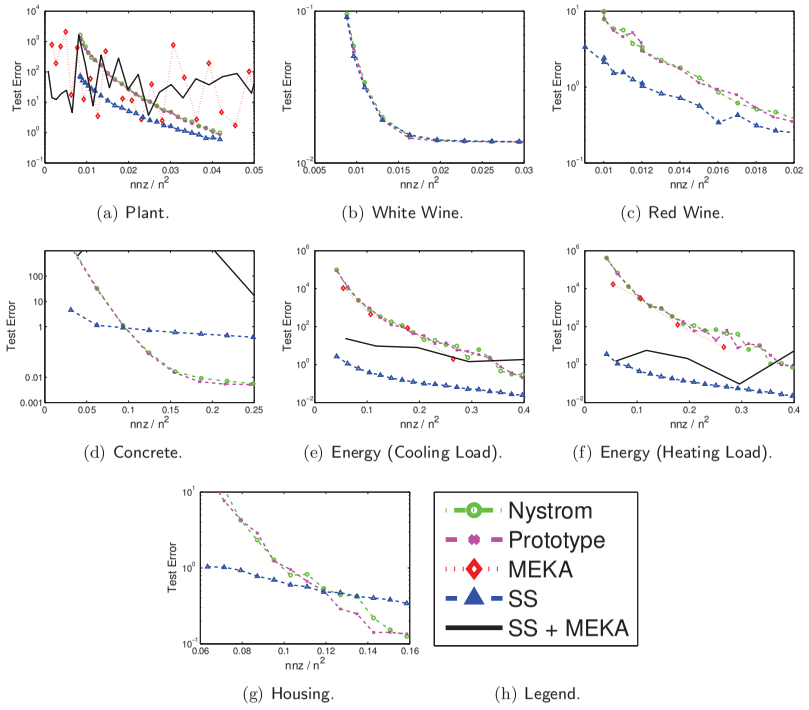


Figure 11: The results on Gaussian process regression. In the figures "nnz" is number of nonzero entries in the sketch, that is, $\mathrm{nnz}(\mathbf{C}) + \mathrm{nnz}(\mathbf{U})$.

where y_{i*} is the real output of the i-th test sample and m is the number of test samples.

We conduct experiment on seven datasets summarized in Table 4. We use the Gaussian RBF kernel and tune two parameters: the variance σ^2 and the kernel scaling parameter γ. Recall that there are seven compared methods and eight datasets, so the time cost would be daunting if we use cross-validation to find σ and γ for each method on each dataset. Thus we perform a five-fold cross-validation without using kernel approximation to predetermine the two parameters σ and γ, and the same parameters are used for all the kernel approximation methods. We list the obtained parameters in Table 4.

For each of the compared methods, we randomly hold 80% samples for training and the rest for test; we repeat this procedure 50 times and record the average MSE, the average

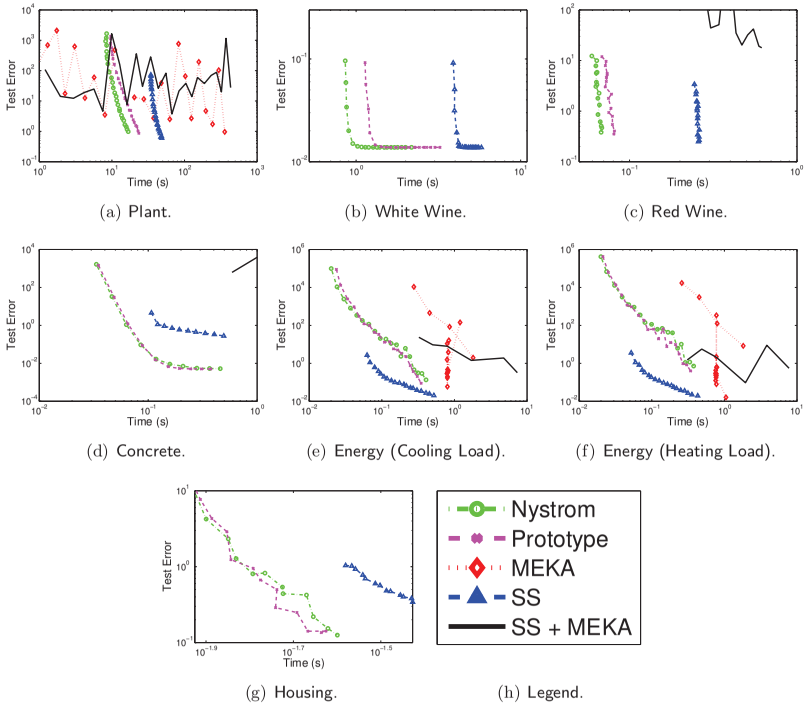


Figure 12: The results on Gaussian process regression.

elapsed time, and the average of the number of nonzero entries in the sketch. We plot $\frac{\mathrm{nnz}(\mathbf{C})+\mathrm{nnz}(\mathbf{U})}{n^2}$ against MSE in Figure 11 and the elapsed time against MSE in Figure 12.

Using the same amount of memory, our SS model achieves the best performance on the Plant, Red Wine, Energy (Cool), and Energy (Heat) datasets. On the Concrete and Housing datasets, using spectral shifting leads better results unless c is unreasonably large (e.g. $c > 0.1n$). However, using the same amount of time, the compared methods have competitive performance.

MEKA and SS+MEKA in general work very well, but they are numerically unstable on some of the randomly partitioned training data. On the White Wine, Housing, and Concrete datasets, MATLAB occasionally reports errors of numerical instability in approximating the off-diagonal blocks of $\mathbf{K}$ by solving some linear systems; when such happens, we do not report the corresponding test errors in the figures. MEKA and SS+MEKA are also sometimes unstable on the Plant and Red Wine datasets, though MATLAB does not report

error. Although MEKA and SS+MEKA have good performance in general, the numerical instability makes MEKA and SS+MEKA perform very poorly on a few among the 50 randomly partitioned training/test data, and consequently the average test errors become large. This problem can get avoided by repeating MEKA multiple times and choose the one that is stable. However, this will significantly increase the time cost.

9. Conclusions

We have provided an in-depth study of the prototype model for SPSD matrix approximation. First, we have shown that with $c = \mathcal{O}(k/\epsilon)$ columns sampled by the near-optimal+adaptive algorithm, the prototype model attains $1+\epsilon$ Frobenius norm relative-error bound. This upper bound matches the lower bound up to a constant factor. Second, we have devised a simple column selection algorithm called *uniform+adaptive*^{2}. The algorithm is efficient and very easy to implement, and it has near-optimal relative-error bound. Third, we have proposed an extension called the spectral shifting (SS) model. We have shown that SS has much stronger error bound than the low-rank approximation models, especially when the bottom eigenvalues are not sufficiently small.

Although the prototype model is not very time-efficient, we can resort to the approximate method provided by Wang et al. (2015) to obtain a faster solution to the prototype model. This faster solution requires only linear time and linear memory. The theoretical analysis of the fast solution heavily relies on that of the prototype model, so our established results are useful even if the prototype model itself is not the working horse in real-world applications. In addition, we have shown that the fast solution can also be naturally incorporated with the spectral shifting method.

Acknowledgments

We thank the anonymous reviewers for their helpful suggestions. Wang has been supported by Baidu Scholarship. Luo and Zhang have been supported by the National Natural Science Foundation of China (No. 61572017), Natural Science Foundation of Shanghai City (No. 15ZR1424200), and Microsoft Research Asia Collaborative Research Award.

Appendix A. Proof of Theorem 1

Proof Suppose that $\mathrm{rank}(\mathbf{W}) = \mathrm{rank}(\mathbf{K})$. We have that $\mathrm{rank}(\mathbf{W}) = \mathrm{rank}(\mathbf{C}) = \mathrm{rank}(\mathbf{K})$ because

$$\mathrm{rank}(\mathbf{K}) \;\geq\; \mathrm{rank}(\mathbf{C}) \;\geq\; \mathrm{rank}(\mathbf{W}). \tag{15}$$

Thus there exists a matrix $\mathbf{X}$ such that

$$\begin{bmatrix} \mathbf{K}_{21}^T \\ \mathbf{K}_{22} \end{bmatrix} \;=\; \mathbf{C}\mathbf{X}^T \;=\; \begin{bmatrix} \mathbf{W}\mathbf{X}^T \\ \mathbf{K}_{21}\mathbf{X}^T \end{bmatrix},$$

and it follows that $\mathbf{K}_{21} = \mathbf{X}\mathbf{W}$ and $\mathbf{K}_{22} = \mathbf{K}_{21}\mathbf{X}^T = \mathbf{X}\mathbf{W}\mathbf{X}^T$. Then we have that

$$\mathbf{K} \;=\; \begin{bmatrix} \mathbf{W} & (\mathbf{X}\mathbf{W})^T \\ \mathbf{X}\mathbf{W} & \mathbf{X}\mathbf{W}\mathbf{X}^T \end{bmatrix} \;=\; \begin{bmatrix} \mathbf{I} \\ \mathbf{X} \end{bmatrix} \mathbf{W} \begin{bmatrix} \mathbf{I} & \mathbf{X}^T \end{bmatrix}, \tag{16}$$

$$\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T \;=\; \begin{bmatrix} \mathbf{W} \\ \mathbf{X}\mathbf{W} \end{bmatrix} \mathbf{W}^\dagger \begin{bmatrix} \mathbf{W} & (\mathbf{X}\mathbf{W})^T \end{bmatrix} \;=\; \begin{bmatrix} \mathbf{I} \\ \mathbf{X} \end{bmatrix} \mathbf{W} \begin{bmatrix} \mathbf{I} & \mathbf{X}^T \end{bmatrix}. \tag{17}$$

Here the second equality in (17) follows from $\mathbf{W}\mathbf{W}^\dagger\mathbf{W} = \mathbf{W}$. We obtain that $\mathbf{K} = \mathbf{C}\mathbf{W}^\dagger\mathbf{C}$. Then we show that $\mathbf{K} = \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{C}^T$.

Since $\mathbf{C}^\dagger = (\mathbf{C}^T\mathbf{C})^\dagger\mathbf{C}^T$, we have that

$$\mathbf{C}^\dagger \;=\; \big(\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}\big)^\dagger\mathbf{W}\,[\mathbf{I},\,\mathbf{X}^T],$$

and thus

$$\begin{aligned} &\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{W} \\ &= \big(\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}\big)^\dagger\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\Big[\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}\big(\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}\big)^\dagger\mathbf{W}\Big] \\ &= \big(\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}\big)^\dagger\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}, \end{aligned}$$

where the second equality follows from Lemma 12 because $(\mathbf{I}+\mathbf{X}^T\mathbf{X})$ is positive definite. Similarly we have

$$\mathbf{W}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{W} \;=\; \mathbf{W}\big(\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W}\big)^\dagger\mathbf{W}(\mathbf{I}+\mathbf{X}^T\mathbf{X})\mathbf{W} \;=\; \mathbf{W}.$$

Thus we have

$$\mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{C} \;=\; \begin{bmatrix} \mathbf{I} \\ \mathbf{X} \end{bmatrix} \mathbf{W}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{W} \begin{bmatrix} \mathbf{I} & \mathbf{X}^T \end{bmatrix} \;=\; \begin{bmatrix} \mathbf{I} \\ \mathbf{X} \end{bmatrix} \mathbf{W} \begin{bmatrix} \mathbf{I} & \mathbf{X}^T \end{bmatrix}. \tag{18}$$

It follows from Equations (16) (17) (18) that $\mathbf{K} = \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T = \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{C}^T$.

Conversely, when $\mathbf{K} = \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T$, it holds that $\mathrm{rank}(\mathbf{K}) \leq \mathrm{rank}(\mathbf{W}^\dagger) = \mathrm{rank}(\mathbf{W})$. It follows from (15) that $\mathrm{rank}(\mathbf{K}) = \mathrm{rank}(\mathbf{W})$.

When $\mathbf{K} = \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^\dagger)^T\mathbf{C}^T$, we have $\mathrm{rank}(\mathbf{K}) \leq \mathrm{rank}(\mathbf{C})$. Thus there exists a matrix $\mathbf{X}$ such that

$$\begin{bmatrix} \mathbf{K}_{21}^T \\ \mathbf{K}_{22} \end{bmatrix} \;=\; \mathbf{C}\mathbf{X}^T \;=\; \begin{bmatrix} \mathbf{W}\mathbf{X}^T \\ \mathbf{K}_{21}\mathbf{X}^T \end{bmatrix},$$

and therefore $\mathbf{K}_{21} = \mathbf{X}\mathbf{W}$. Then we have that

$$\mathbf{C} \;=\; \begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{21} \end{bmatrix} \;=\; \begin{bmatrix} \mathbf{I} \\ \mathbf{X} \end{bmatrix} \mathbf{W},$$

so $\mathrm{rank}(\mathbf{C}) \leq \mathrm{rank}(\mathbf{W})$. Apply (15) again we have $\mathrm{rank}(\mathbf{K}) = \mathrm{rank}(\mathbf{W})$. ■

Lemma 12 $\mathbf{X}^T\mathbf{V}\mathbf{X}\big(\mathbf{X}^T\mathbf{V}\mathbf{X}\big)^\dagger\mathbf{X}^T = \mathbf{X}^T$ *for any positive definite matrix* $\mathbf{V}$.

Proof The positive definite matrix $\mathbf{V}$ have a decomposition $\mathbf{V} = \mathbf{B}^T\mathbf{B}$ for some nonsingular matrix $\mathbf{B}$. It follows that

$$\begin{aligned}\mathbf{X}^T\mathbf{V}\mathbf{X}\big(\mathbf{X}^T\mathbf{V}\mathbf{X}\big)^\dagger\mathbf{X}^T &= (\mathbf{B}\mathbf{X})^T\Big(\mathbf{B}\mathbf{X}\big((\mathbf{B}\mathbf{X})^T(\mathbf{B}\mathbf{X})\big)^\dagger\Big)(\mathbf{B}\mathbf{X})^T\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\\ &= (\mathbf{B}\mathbf{X})^T\big((\mathbf{B}\mathbf{X})^T\big)^\dagger(\mathbf{B}\mathbf{X})^T(\mathbf{B}^T)^{-1} = (\mathbf{B}\mathbf{X})^T(\mathbf{B}^T)^{-1} = \mathbf{X}^T.\end{aligned}$$

■

Appendix B. Proof of Theorem 2

In Section B.1 we provide several key lemmas, and then in Section B.2 we prove Theorem 2 using Lemmas 14 and 15.

B.1 Key Lemmas

Lemma 13 provides a useful tool for expanding the Moore-Penrose inverse of partitioned matrices, and the lemma will be used to prove Lemma 15 and Theorem 2.

Lemma 13 (see Ben-Israel and Greville, 2003, Page 179) *Given a matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ of rank c, let it have a nonsingular $c\times c$ submatrix $\mathbf{X}_{11}$. By rearrangement of columns and rows by permutation matrices $\mathbf{P}$ and $\mathbf{Q}$, the submatrix $\mathbf{X}_{11}$ can be bought to the top left corner of $\mathbf{X}$, that is,*

$$\mathbf{P}\mathbf{X}\mathbf{Q} = \begin{bmatrix}\mathbf{X}_{11} & \mathbf{X}_{12}\\ \mathbf{X}_{21} & \mathbf{X}_{22}\end{bmatrix}.$$

Then the Moore-Penrose inverse of $\mathbf{X}$ is

$$\mathbf{X}^\dagger = \mathbf{Q}\begin{bmatrix}\mathbf{I}_c\\ \mathbf{T}^T\end{bmatrix}\big(\mathbf{I}_c+\mathbf{T}\mathbf{T}^T\big)^{-1}\mathbf{X}_{11}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\begin{bmatrix}\mathbf{I}_c & \mathbf{S}^T\end{bmatrix}\mathbf{P},$$

where $\mathbf{T} = \mathbf{X}_{11}^{-1}\mathbf{X}_{12}$ and $\mathbf{S} = \mathbf{X}_{21}\mathbf{X}_{11}^{-1}$.

Lemmas 14 and 15 will be used to prove Theorem 2.

Lemma 14 (Wang and Zhang, 2013, Lemma 19) *Given n and k, we let $\mathbf{B}$ be an $\frac{n}{k}\times\frac{n}{k}$ matrix whose diagonal entries equal to one and off-diagonal entries equal to $\alpha\in[0,1)$. Let $\mathbf{A}$ be the $n\times n$ block-diagonal matrix*

$$\mathbf{A} = \underbrace{\mathbf{B}\oplus\mathbf{B}\oplus\cdots\oplus\mathbf{B}}_{k \text{ blocks}}. \tag{19}$$

Let $\mathbf{A}_k$ be the best rank-k approximation to $\mathbf{A}$. Then

$$\|\mathbf{A}-\mathbf{A}_k\|_F = (1-\alpha)\sqrt{n-k}.$$

Lemma 15 *Let $\mathbf{B}$ be the $n\times n$ matrix with diagonal entries equal to one and off-diagonal entries equal to α and $\tilde{\mathbf{B}}$ be its rank c approximation formed by the prototype model. Then*

$$\|\mathbf{B}-\tilde{\mathbf{B}}\|_F^2 \geq (1-\alpha)^2(n-c)\left(1+\frac{2}{c}-(1-\alpha)\frac{1+o(1)}{\alpha cn/2}\right).$$

Proof Without loss of generality, we assume the first c column of $\mathbf{B}$ are selected to construct $\mathbf{C}$. We partition $\mathbf{B}$ and $\mathbf{C}$ as:

$$\mathbf{B} = \begin{bmatrix}\mathbf{W} & \mathbf{B}_{21}^T\\ \mathbf{B}_{21} & \mathbf{B}_{22}\end{bmatrix} \quad \text{and} \quad \mathbf{C} = \begin{bmatrix}\mathbf{W}\\ \mathbf{B}_{21}\end{bmatrix}.$$

Here the matrix $\mathbf{W}$ can be expressed by $\mathbf{W} = (1-\alpha)\mathbf{I}_c + \alpha\mathbf{1}_c\mathbf{1}_c^T$. We apply the Sherman-Morrison-Woodbury matrix identity

$$(\mathbf{X}+\mathbf{Y}\mathbf{Z}\mathbf{R})^{-1} = \mathbf{X}^{-1} - \mathbf{X}^{-1}\mathbf{Y}(\mathbf{Z}^{-1}+\mathbf{R}\mathbf{X}^{-1}\mathbf{Y})^{-1}\mathbf{R}\mathbf{X}^{-1}$$

to compute $\mathbf{W}^{-1}$, and it follows that

$$\mathbf{W}^{-1} = \frac{1}{1-\alpha}\mathbf{I}_c - \frac{\alpha}{(1-\alpha)(1-\alpha+c\alpha)}\mathbf{1}_c\mathbf{1}_c^T. \tag{20}$$

We expand the Moore-Penrose inverse of $\mathbf{C}$ by Lemma 13 and obtain

$$\mathbf{C}^\dagger = \mathbf{W}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\begin{bmatrix}\mathbf{I}_c & \mathbf{S}^T\end{bmatrix}$$

where

$$\mathbf{S} = \mathbf{B}_{21}\mathbf{W}^{-1} = \frac{\alpha}{1-\alpha+c\alpha}\mathbf{1}_{n-c}\mathbf{1}_c^T.$$

It is easily verified that $\mathbf{S}^T\mathbf{S} = \big(\frac{\alpha}{1-\alpha+c\alpha}\big)^2(n-c)\mathbf{1}_c\mathbf{1}_c^T$.

Now we express the approximation by the prototype model in the partitioned form:

$$\begin{aligned}\tilde{\mathbf{B}} &= \mathbf{C}\mathbf{C}^\dagger\mathbf{B}\big(\mathbf{C}^\dagger\big)^T\mathbf{C}^T\\ &= \begin{bmatrix}\mathbf{W}\\ \mathbf{B}_{21}\end{bmatrix}\mathbf{W}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\begin{bmatrix}\mathbf{I}_c & \mathbf{S}^T\end{bmatrix}\mathbf{B}\begin{bmatrix}\mathbf{I}_c\\ \mathbf{S}\end{bmatrix}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\mathbf{W}^{-1}\begin{bmatrix}\mathbf{W}\\ \mathbf{B}_{21}\end{bmatrix}^T\\ &= \begin{bmatrix}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\\ \mathbf{B}_{21}\mathbf{W}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{I}_c & \mathbf{S}^T\end{bmatrix}\mathbf{B}\begin{bmatrix}\mathbf{I}_c\\ \mathbf{S}\end{bmatrix}\begin{bmatrix}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\\ \mathbf{B}_{21}\mathbf{W}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}\end{bmatrix}^T.\end{aligned} \tag{21}$$

We then compute the submatrices $\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}$ and $\mathbf{B}_{21}\mathbf{W}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}$ respectively as follows. We apply the Sherman-Morrison-Woodbury matrix identity to compute $\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1}$. We obtain

$$\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1} = \left(\mathbf{I}_c+\Big(\frac{\alpha}{1-\alpha+c\alpha}\Big)^2(n-c)\mathbf{1}_c\mathbf{1}_c^T\right)^{-1} = \mathbf{I}_c-\gamma_1\mathbf{1}_c\mathbf{1}_c^T, \tag{22}$$

where

$$\gamma_1 = \frac{n-c}{nc+\big(\frac{1-\alpha}{\alpha}\big)^2+\frac{2(1-\alpha)c}{\alpha}}.$$

It follows from (20) and (22) that

$$\mathbf{W}^{-1}\big(\mathbf{I}_c+\mathbf{S}^T\mathbf{S}\big)^{-1} = (\gamma_2\mathbf{I}_c-\gamma_3\mathbf{1}_c\mathbf{1}_c^T)(\mathbf{I}_c-\gamma_1\mathbf{1}_c\mathbf{1}_c^T) = \gamma_2\mathbf{I}_c+(\gamma_1\gamma_3c-\gamma_1\gamma_2-\gamma_3)\mathbf{1}_c\mathbf{1}_c^T,$$

where

$$\gamma_2 = \frac{1}{1-\alpha} \quad \text{and} \quad \gamma_3 = \frac{\alpha}{(1-\alpha)(1-\alpha+\alpha c)}.$$

It follows that

$$\mathbf{B}_{21}\mathbf{W}^{-1}\big(\mathbf{I}_c + \mathbf{S}^T\mathbf{S}\big)^{-1} = \alpha\big(\gamma_1\gamma_3 c^2 - \gamma_3 c - \gamma_1\gamma_2 c + \gamma_2\big)\mathbf{1}_{n-c}\mathbf{1}_c^T \triangleq \gamma\mathbf{1}_{n-c}\mathbf{1}_c^T, \tag{23}$$

where

$$\gamma = \alpha\big(\gamma_1\gamma_3 c^2 - \gamma_3 c - \gamma_1\gamma_2 c + \gamma_2\big) = \frac{\alpha(\alpha c - \alpha + 1)}{2\alpha c - 2\alpha - 2\alpha^2 c + \alpha^2 + \alpha^2 c n + 1}. \tag{24}$$

Since $\mathbf{B}_{21} = \alpha\mathbf{1}_{n-c}\mathbf{1}_c^T$ and $\mathbf{B}_{22} = (1-\alpha)\mathbf{I}_{n-c} + \alpha\mathbf{1}_{n-c}\mathbf{1}_{n-c}^T$, it is easily verified that

$$\begin{bmatrix} \mathbf{I}_c & \mathbf{S}^T \end{bmatrix}\mathbf{B}\begin{bmatrix} \mathbf{I}_c \\ \mathbf{S} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_c & \mathbf{S}^T \end{bmatrix}\begin{bmatrix} \mathbf{W} & \mathbf{B}_{21}^T \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}\begin{bmatrix} \mathbf{I}_c \\ \mathbf{S} \end{bmatrix} = (1-\alpha)\mathbf{I}_c + \lambda\mathbf{1}_c\mathbf{1}_c^T, \tag{25}$$

where

$$\lambda = \frac{\alpha(3\alpha n - \alpha c - 2\alpha + \alpha^2 c - 3\alpha^2 n + \alpha^2 + \alpha^2 n^2 + 1)}{(\alpha c - \alpha + 1)^2}$$

It follows from (21), (22), (23), and (25) that

$$\tilde{\mathbf{B}} = \begin{bmatrix} \mathbf{I}_c - \gamma_1\mathbf{1}_c\mathbf{1}_c^T \\ \gamma\mathbf{1}_{n-c}\mathbf{1}_c^T \end{bmatrix}\Big((1-\alpha)\mathbf{I}_c + \lambda\mathbf{1}_c\mathbf{1}_c^T\Big)\begin{bmatrix} \mathbf{I}_c - \gamma_1\mathbf{1}_c\mathbf{1}_c^T \\ \gamma\mathbf{1}_{n-c}\mathbf{1}_c^T \end{bmatrix}^T \triangleq \begin{bmatrix} \tilde{\mathbf{B}}_{11} & \tilde{\mathbf{B}}_{21}^T \\ \tilde{\mathbf{B}}_{21} & \tilde{\mathbf{B}}_{22} \end{bmatrix},$$

where

$$\begin{aligned}
\tilde{\mathbf{B}}_{11} &= (1-\alpha)\mathbf{I}_c + \big[(1-\gamma_1 c)(\lambda - \lambda\gamma_1 c - (1-\alpha)\gamma_1) - (1-\alpha)\gamma_1\big]\mathbf{1}_c\mathbf{1}_c^T \\
&= (1-\alpha)\mathbf{I}_c + \eta_1\mathbf{1}_c\mathbf{1}_c^T, \\
\tilde{\mathbf{B}}_{21} &= \tilde{\mathbf{A}}_{12}^T = \gamma(1-\gamma_1 c)(1-\alpha+\lambda c)\mathbf{1}_{n-c}\mathbf{1}_c^T = \eta_2\mathbf{1}_{n-c}\mathbf{1}_c^T, \\
\tilde{\mathbf{B}}_{22} &= \gamma^2 c(1-\alpha+\lambda c)\mathbf{1}_{n-c}\mathbf{1}_{n-c}^T = \eta_3\mathbf{1}_{n-c}\mathbf{1}_{n-c}^T,
\end{aligned}$$

where

$$\begin{aligned}
\eta_1 &= (1-\gamma_1 c)(\lambda - \lambda\gamma_1 c - (1-\alpha)\gamma_1) - (1-\alpha)\gamma_1, \\
\eta_2 &= \gamma(1-\gamma_1 c)(1-\alpha+\lambda c), \\
\eta_3 &= \gamma^2 c(1-\alpha+\lambda c),
\end{aligned}$$

By dealing with the four blocks of $\tilde{\mathbf{B}}$ respectively, we finally obtain that

$$\begin{aligned}
\|\mathbf{B} - \tilde{\mathbf{B}}\|_F^2 &= \|\mathbf{W} - \tilde{\mathbf{B}}_{11}\|_F^2 + 2\|\mathbf{B}_{21} - \tilde{\mathbf{B}}_{21}\|_F^2 + \|\mathbf{B}_{22} - \tilde{\mathbf{B}}_{22}\|_F^2 \\
&= c^2(\alpha - \eta_1)^2 + 2c(n-c)(\alpha - \eta_2)^2 \\
&\quad + (n-c)(n-c-1)(\alpha - \eta_3)^2 + (n-c)(1-\eta_3)^2 \\
&= (n-c)(\alpha-1)^2\bigg(1 + \frac{2}{c} - \big(1+o(1)\big)\frac{1-\alpha}{\alpha c n/2}\bigg).
\end{aligned}$$

■

B.2 Proof of the Theorem

Now we prove Theorem 2 using Lemma 15 and Lemma 14. Let $\mathbf{C}$ consist of c columns sampled from $\mathbf{A}$ and $\hat{\mathbf{C}}_i$ consist of c_i columns sampled from the i-th block diagonal matrix in $\mathbf{A}$. Without loss of generality, we assume $\hat{\mathbf{C}}_i$ consists of the first c_i columns of $\mathbf{B}$. Then the intersection matrix $\mathbf{U}$ is computed by

$$\begin{aligned}
\mathbf{U} &= \mathbf{C}^\dagger\mathbf{A}(\mathbf{C}^T)^\dagger = \big[\hat{\mathbf{C}}_1 \oplus \cdots \oplus \hat{\mathbf{C}}_k\big]^\dagger\big[\mathbf{B} \oplus \cdots \oplus \mathbf{B}\big]\big[\hat{\mathbf{C}}_1^T \oplus \cdots \oplus \hat{\mathbf{C}}_k^T\big]^\dagger \\
&= \hat{\mathbf{C}}_1^\dagger\mathbf{B}(\hat{\mathbf{C}}_1^\dagger)^T \oplus \cdots \oplus \hat{\mathbf{C}}_k^\dagger\mathbf{B}(\hat{\mathbf{C}}_k^\dagger)^T.
\end{aligned}$$

Let $\tilde{\mathbf{A}}$ be the approximation formed by the prototype model. Then

$$\tilde{\mathbf{A}} = \mathbf{C}\mathbf{U}\mathbf{C}^T = \hat{\mathbf{C}}_1\hat{\mathbf{C}}_1^\dagger\mathbf{B}(\hat{\mathbf{C}}_1^\dagger)^T\hat{\mathbf{C}}_1^T \oplus \cdots \oplus \hat{\mathbf{C}}_k\hat{\mathbf{C}}_k^\dagger\mathbf{B}(\hat{\mathbf{C}}_k^\dagger)^T\hat{\mathbf{C}}_k^T,$$

and thus the approximation error is

$$\begin{aligned}
\|\mathbf{A} - \tilde{\mathbf{A}}\|_F^2 &= \sum_{i=1}^k \Big\|\mathbf{B} - \hat{\mathbf{C}}_i\hat{\mathbf{C}}_i^\dagger\mathbf{B}(\hat{\mathbf{C}}_i^\dagger)^T\hat{\mathbf{C}}_i^T\Big\|_F^2 \\
&\geq (1-\alpha)^2\sum_{i=1}^k(p-c_i)\bigg(1 + \frac{2}{c_i} - (1-\alpha)\Big(\frac{1+o(1)}{\alpha c_i p/2}\Big)\bigg) \\
&= (1-\alpha)^2\bigg(\sum_{i=1}^k(p-c_i) + \sum_{i=1}^k\frac{2(p-c_i)}{c_i}\Big(1 - \frac{(1-\alpha)(1+o(1))}{\alpha p}\Big)\bigg) \\
&\geq (1-\alpha)^2(n-c)\bigg(1 + \frac{2k}{c}\Big(1 - \frac{k(1-\alpha)(1+o(1))}{\alpha n}\Big)\bigg),
\end{aligned}$$

where the former inequality follows from Lemma 15, and the latter inequality follows by minimizing over $c_1, \cdots, c_k$. Finally the theorem follows by setting $\alpha \to 1$ and applying Lemma 14.

Appendix C. Proof of Theorem 3

Proof Sampling $c_1 = 2k\epsilon^{-1}\big(1+o(1)\big)$ columns of $\mathbf{K}$ by the near-optimal algorithm of Boutsidis et al. (2014) to form $\mathbf{C}_1 \in \mathbb{R}^{n\times c_1}$, we have that

$$\mathbb{E}\big\|\mathbf{K} - \mathcal{P}_{\mathbf{C}_1,k}(\mathbf{K})\big\|_F^2 \leq (1+\epsilon)\big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2.$$

Applying Lemma 3.11 of Boutsidis and Woodruff (2014), we can find a much smaller matrix $\mathbf{U}_1 \in \mathbb{R}^{n\times k}$ with orthogonal columns in the column space of $\mathbf{C}_1$ such that

$$\big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}\big\|_F^2 \leq \big\|\mathbf{K} - \mathcal{P}_{\mathbf{C}_1,k}(\mathbf{K})\big\|_F^2.$$

(We do not actually compute $\mathbf{U}_1$ because the adaptive sampling algorithm does not need to know $\mathbf{U}_1$.) Because the columns of $\mathbf{U}_1$ are all in the columns space of $\mathbf{C}_1$, we have that $\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}\big\|_F^2 \leq \big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}\big\|_F^2$. Combining the above inequalities we obtain

$$\mathbb{E}\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}\big\|_F^2 \leq \mathbb{E}\big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}\big\|_F^2 \leq \big\|\mathbf{K} - \mathcal{P}_{\mathbf{C}_1,k}(\mathbf{K})\big\|_F^2.$$

Given $\mathbf{C}_1$, we use adaptive sampling to select $c_2 = k\epsilon^{-1}$ rows of $\mathbf{K}$ to form $\mathbf{C}_2$ and denote $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2]$. Since the columns of $\mathbf{U}_1$ are all in the columns space of $\mathbf{C}_1$, Lemma 17 of Wang and Zhang (2013) can be slightly modified to show

$$\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 \;\leq\; \big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2.$$

By the adaptive sampling theorem of Wang and Zhang (2013) we have

$$\begin{aligned}
\mathbb{E}\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 &\leq \mathbb{E}\big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 \\
&\leq \big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}\big\|_F^2 + \frac{k}{c_2}\big\|\mathbf{K} - \mathbf{K}(\mathbf{C}_1^T)^\dagger\mathbf{C}_1^T\big\|_F^2 \\
&\leq \Big(1 + \frac{k}{c_2}\Big)\big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^\dagger\mathbf{K}\big\|_F^2 \\
&\leq (1+\epsilon)\big\|\mathbf{K} - \mathcal{P}_{\mathbf{C}_1,k}(\mathbf{K})\big\|_F^2, \qquad (26)
\end{aligned}$$

where the expectation is taken w.r.t. $\mathbf{C}_2$, and the last inequality follows by setting $c_2 = k/\epsilon$. Here the trick is bounding $\mathbb{E}\big\|\mathbf{K} - \mathbf{U}_1\mathbf{U}_1^T\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2$ rather than directly bounding $\mathbb{E}\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2$; otherwise the factor $\frac{k}{c_2}$ would be $\frac{c_1}{c_2} = \frac{2k\epsilon^{-1}(1+o(1))}{c_2}$. This is the key to the improvement. It follows that

$$\begin{aligned}
\mathbb{E}\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 &\leq (1+\epsilon)\mathbb{E}\big\|\mathbf{K} - \mathcal{P}_{\mathbf{C}_1,k}(\mathbf{K})\big\|_F^2 \\
&\leq (1+\epsilon)^2\big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2,
\end{aligned}$$

where the first expectation is taken w.r.t. $\mathbf{C}_1$ and $\mathbf{C}_2$, and the second expectation is taken w.r.t. $\mathbf{C}_1$. Applying Lemma 17 of Wang and Zhang (2013) again, we obtain

$$\mathbb{E}\big\|\mathbf{K} - \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 \;\leq\; \mathbb{E}\big\|\mathbf{K} - \mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 \;\leq\; (1+\epsilon)^2\big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2.$$

Hence totally $c = c_1 + c_2 = 3k\epsilon^{-1}\big(1 + o(1)\big)$ columns suffice. ■

Appendix D. Proof of Theorem 4

In this section we first provide a constant factor bound of the uniform sampling, and then prove Theorem 4 in the subsequent subsections.

D.1 Tools for Analyzing Uniform Sampling

This subsection provides several useful tools for analyzing column sampling. The matrix $\mathbf{S} \in \mathbb{R}^{n\times s}$ is a column selection matrix if each column has exactly one nonzero entry; let (i_j, j) be the position of the nonzero entry in the j-th column. Let us add randomness to column selection. Suppose we are given the sampling probabilities $p_1, \cdots, p_n \in (0,1)$ and $\sum_i p_i = 1$. In each round we pick one element in $[n]$ such that the i-th element is sampled with probability p_i. We repeat the procedure s times, either with or without replacement, and let $i_1, \cdots, i_s$ be the selected indices. For $j = 1$ to s, we set

$$S_{i_j,j} = \frac{1}{\sqrt{sp_{i_j}}}. \qquad (27)$$

The following lemma shows an important property of arbitrary column sampling. The proof mirrors the leverage score sampling bound in (Woodruff, 2014).

Lemma 16 *Let $\mathbf{U} \in \mathbb{R}^{n\times k}$ be any fixed matrix with orthonormal columns. The column selection matrix $\mathbf{S} \in \mathbb{R}^{n\times s}$ samples s columns according to arbitrary probabilities $p_1, p_2, \cdots, p_n$. Assume that*

$$\max_{i\in[n]} \frac{\|\mathbf{u}_{i:}\|_2^2}{p_i} \;\leq\; \alpha$$

and $\alpha \geq 1$. When $s \geq \alpha\frac{6+2\eta}{3\eta^2}\log(k/\delta)$, it holds that

$$\mathbb{P}\Big\{\big\|\mathbf{I}_k - \mathbf{U}^T\mathbf{S}\mathbf{S}^T\mathbf{U}\big\|_2 \;\geq\; \eta\Big\} \;\leq\; \delta.$$

Proof We can express $\mathbf{I}_k = \mathbf{U}^T\mathbf{U}$ as the sum of size $k\times k$ and rank one matrices:

$$\mathbf{I}_k \;=\; \mathbf{U}^T\mathbf{U} \;=\; \sum_{i=1}^n \mathbf{u}_{i:}^T\mathbf{u}_{i:}$$

where $\mathbf{u}_{i:} \in \mathbb{R}^{1\times k}$ is the i-th row of $\mathbf{U}$. The approximate matrix product can be expressed as

$$\mathbf{U}^T\mathbf{S}\mathbf{S}^T\mathbf{U} \;=\; \sum_{j=1}^s S_{i_j,j}^2\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:} \;=\; \sum_{j=1}^s \frac{1}{sp_{i_j}}\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:}.$$

We define the symmetric random matrices

$$\mathbf{Z}_{i_j} \;=\; \frac{1}{sp_{i_j}}\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:} - \frac{1}{s}\mathbf{I}_k$$

for $j = 1$ to s. Whatever the sampling distribution is, it always holds that

$$\mathbb{E}\mathbf{Z}_{i_j} \;=\; \sum_{q=1}^n p_q\Big(\frac{1}{\sqrt{sp_q}}\Big)^2\mathbf{u}_{q:}^T\mathbf{u}_{q:} - \sum_{q=1}^n p_q\frac{1}{s}\mathbf{I}_k \;=\; \frac{1}{s}\mathbf{I}_k - \frac{1}{s}\mathbf{I}_k \;=\; \mathbf{0}. \qquad (28)$$

Thus $\mathbf{Z}_{i_j}$ has zero mean. Let $\mathcal{Z}$ be the set

$$\mathcal{Z} \;=\; \Big\{\frac{1}{s}\mathbf{I}_k - \frac{1}{sp_1}\mathbf{u}_{1:}^T\mathbf{u}_{1:},\; \cdots,\; \frac{1}{s}\mathbf{I}_k - \frac{1}{sp_n}\mathbf{u}_{n:}^T\mathbf{u}_{n:}\Big\}.$$

Clearly, $\mathbf{Z}_{i_1}, \cdots, \mathbf{Z}_{i_s}$ are sampled from $\mathcal{Z}$, and

$$\mathbf{U}^T\mathbf{S}\mathbf{S}^T\mathbf{U} - \mathbf{I}_k = \sum_{j=1}^s \mathbf{Z}_{i_j}.$$

Therefore we can bound its spectral norm using the matrix Bernstein. Elementary proof of the matrix Bernstein can be found in Tropp (2015).

Lemma 17 (Matrix Bernstein) *Consider a finite sequence $\{\mathbf{Z}_i\}$ of independent, random, Hermitian matrices with dimension k. Assume that*

$$\mathbb{E}\mathbf{Z}_i = \mathbf{0} \quad \text{and} \quad \max_i \|\mathbf{Z}_i\|_2 \le L$$

for each index i. Introduce the random matrix $\mathbf{Y} = \sum_i \mathbf{Z}_i$. Let $v(\mathbf{Y})$ be the matrix variance statistics of the sum:

$$v(\mathbf{Y}) \;=\; \left\|\mathbb{E}\mathbf{Y}^2\right\|_2 \;=\; \Big\|\sum_i \mathbb{E}\mathbf{Z}_i^2\Big\|_2.$$

Then

$$\mathbb{P}\{\lambda_{\max}(\mathbf{Y}) \ge \eta\} \;\le\; k \cdot \exp\left(\frac{-\eta^2/2}{v(\mathbf{Y}) + L\eta/3}\right).$$

To apply the matrix Bernstein, it remains to bound the variance of $\sum_{j=1}^s \mathbf{Z}_{i_j}$ and to bound $L = \max_{j\in[s]} \|\mathbf{Z}_{i_j}\|_2$. We have that

$$\begin{aligned}
\mathbb{E}\mathbf{Z}_{i_j}^2 &= \mathbb{E}\Big(\frac{1}{sp_{i_j}}\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:} - \frac{1}{s}\mathbf{I}_k\Big)^2 \\
&= \frac{1}{s^2}\left[\mathbb{E}\Big(\frac{1}{p_{i_j}}\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:}\Big)^2 - 2\mathbb{E}\Big(\frac{1}{p_{i_j}}\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:}\Big) + \mathbf{I}_k\right] \\
&= \frac{1}{s^2}\left[\sum_{q=1}^n \Big(p_q\Big(\frac{1}{p_q}\Big)^2\mathbf{u}_{q:}^T\mathbf{u}_{q:}\mathbf{u}_{q:}^T\mathbf{u}_{q:}\Big) - 2\mathbf{I}_k + \mathbf{I}_k\right] \\
&= -\frac{1}{s^2}\mathbf{I}_k + \frac{1}{s^2}\sum_{q=1}^n \frac{\|\mathbf{u}_{q:}\|_2^2}{p_q}\mathbf{u}_{q:}^T\mathbf{u}_{q:}.
\end{aligned}$$

The variance is defined by

$$v \;\triangleq\; \left\|\sum_{j=1}^s \mathbb{E}\mathbf{Z}_{i_j}^2\right\|_2 \;=\; \frac{1}{s}\left\| - \mathbf{I}_k + \sum_{q=1}^n \frac{\|\mathbf{u}_{q:}\|_2^2}{p_q}\mathbf{u}_{q:}^T\mathbf{u}_{q:}\right\|_2 \;\le\; \frac{1}{s}(\alpha - 1).$$

Here the inequality is due to the definition of α and

$$-\mathbf{I}_k + \sum_{q=1}^n \frac{\|\mathbf{u}_{q:}\|_2^2}{p_q}\mathbf{u}_{q:}^T\mathbf{u}_{q:} \;\preceq\; -\mathbf{I}_k + \sum_{q=1}^n \alpha\mathbf{u}_{q:}^T\mathbf{u}_{q:} \;=\; (-1+\alpha)\mathbf{I}_k.$$

In addition, $L = \max_{j\in[s]} \|\mathbf{Z}_{i_j}\|_2$ can be bounded by

$$\begin{aligned}
L \;&=\; \max_{j\in[s]} \|\mathbf{Z}_{i_j}\|_2 \;=\; \max_{j\in[s]} \left\|\frac{1}{s}\mathbf{I}_k - \frac{1}{sp_{i_j}}\mathbf{u}_{i_j:}^T\mathbf{u}_{i_j:}\right\|_2 \;\le\; \max_{i\in[n]} \left\|\frac{1}{s}\mathbf{I}_k - \frac{1}{sp_i}\mathbf{u}_{i:}^T\mathbf{u}_{i:}\right\|_2 \\
&\le\; \frac{1}{s} + \max_{i\in[n]} \left\|\frac{1}{sp_i}\mathbf{u}_{i:}^T\mathbf{u}_{i:}\right\|_2 \;=\; \frac{1}{s} + \max_{i\in[n]} \frac{\|\mathbf{u}_{i:}\|_2^2}{sp_i} \;\le\; \frac{1}{s}(\alpha+1).
\end{aligned}$$

Finally, the lemma follows by plugging v and L in the matrix Bernstein. ■

Theorem 18 shows an important property of uniform sampling. It shows that when the number of sampled columns is large enough, all the singular values of $\mathbf{U}^T\mathbf{S}\mathbf{S}^T\mathbf{U}$ are within $1 \pm \eta$ with high probability.

Theorem 18 *Let $\mathbf{U} \in \mathbb{R}^{n\times k}$ be any fixed matrix with orthonormal columns and $\mu(\mathbf{U})$ be its row coherence. Let $\mathbf{S} \in \mathbb{R}^{n\times s}$ be an uniformly sampling matrix. When $s \ge \mu(\mathbf{U})k\frac{6+2\eta}{3\eta^2}\log(k/\delta)$, it hold that*

$$\mathbb{P}\Big\{\big\|\mathbf{U}^T\mathbf{S}\mathbf{S}^T\mathbf{U} - \mathbf{I}_k\big\|_2 \;\ge\; \eta\Big\} \;\le\; \delta.$$

Proof Since $\frac{\|\mathbf{u}_{q:}\|_2^2}{p_q} = n\|\mathbf{u}_{q:}\|_2^2 \le k\mu(\mathbf{U})$ for all $q \in [n]$, the theorem follows from Lemma 16. ■

The following lemma was established by Drineas et al. (2008). It can be proved by writing the squared Frobenius norm as the sum of scalars and then taking expectation.

Lemma 19 *Let $\mathbf{A} \in \mathbb{R}^{n\times k}$ and $\mathbf{B} \in \mathbb{R}^{n\times d}$ be any fixed matrices and $\mathbf{S} \in \mathbb{R}^{n\times s}$ be the column sampling matrix defined in (27). Assume that the columns are selected randomly and pairwisely independently. Then*

$$\mathbb{E}\Big\|\mathbf{A}^T\mathbf{B} - \mathbf{A}^T\mathbf{S}\mathbf{S}^T\mathbf{B}\Big\|_F^2 \;\le\; \frac{1}{s}\sum_{i=1}^n \frac{1}{p_i}\big\|\mathbf{a}_{i:}\big\|_2^2\big\|\mathbf{b}_{i:}\big\|_2^2.$$

Theorem 20 follows from Lemma 19 and shows another important property of uniform sampling.

Theorem 20 *Let $\mathbf{U} \in \mathbb{R}^{n\times k}$ be any fixed matrix with orthonormal columns and $\mathbf{B} \in \mathbb{R}^{n\times d}$ be any fixed matrix. Use the uniform sampling matrix $\mathbf{S} \in \mathbb{R}^{n\times s}$ and set $s \ge \frac{k\mu(\mathbf{U})}{\epsilon\delta}$. Then it holds that*

$$\mathbb{P}\Big\{\big\|\mathbf{U}^T\mathbf{B} - \mathbf{U}^T\mathbf{S}^T\mathbf{S}\mathbf{B}\big\|_F^2 \;\ge\; \epsilon\|\mathbf{B}\|_F^2\Big\} \;\le\; \delta.$$

Proof We have shown that $\|\mathbf{u}_{q:}\|_2^2/p_q \le k\mu(\mathbf{U})$ for all $q = 1$ to n. It follows from Lemma 19 that

$$\mathbb{E}\Big\|\mathbf{U}^T\mathbf{B} - \mathbf{U}^T\mathbf{S}\mathbf{S}^T\mathbf{B}\Big\|_F^2 \;\le\; \frac{1}{s}\sum_{i=1}^n k\mu(\mathbf{U})\big\|\mathbf{b}_{i:}\big\|_2^2 \;=\; \frac{k\mu(\mathbf{U})}{s}\|\mathbf{B}\|_F^2.$$

The theorem follows from the setting of s and the Markov's inequality. ■

The following lemma is very useful in analyzing randomized SVD.

Lemma 21 *Let $\mathbf{M} \in \mathbb{R}^{m\times n}$ be any matrix. We decompose $\mathbf{M}$ by $\mathbf{M} = \mathbf{M}_1 + \mathbf{M}_2$ such that $\mathrm{rank}(\mathbf{M}_1) = k$. Let the right singular vectors of $\mathbf{M}_1$ be $\mathbf{V}_1 \in \mathbb{R}^{n\times k}$. Let $\mathbf{S} \in \mathbb{R}^{n\times c}$ be any matrix such that $\mathrm{rank}(\mathbf{V}_1^T\mathbf{S}) = k$ and let $\mathbf{C} = \mathbf{M}\mathbf{S} \in \mathbb{R}^{m\times c}$. Then*

$$\big\|\mathbf{M} - \mathbf{C}(\mathbf{V}_1^T\mathbf{S})^\dagger\mathbf{V}_1^T\big\|_\xi^2 \;\le\; \big\|\mathbf{M}_2\big\|_\xi^2 + \sigma_{\min}^{-2}(\mathbf{V}_1^T\mathbf{S}\mathbf{S}^T\mathbf{V}_1)\,\big\|\mathbf{M}_2\mathbf{S}\mathbf{S}^T\mathbf{V}_1\big\|_\xi^2$$

for $\xi = 2$ or F.

Proof Boutsidis et al. (2014) showed that

$$\big\|\mathbf{M}-\mathbf{C}(\mathbf{V}_1^T\mathbf{S})^\dagger\mathbf{V}_1^T\big\|_\xi^2 \;\le\; \big\|\mathbf{M}_2\big\|_\xi^2+\big\|\mathbf{M}_2\mathbf{S}(\mathbf{V}_1^T\mathbf{S})^\dagger\big\|_\xi^2.$$

Since $\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^\dagger=\mathbf{Y}^\dagger$ for any matrix $\mathbf{Y}$, it follows that

$$\begin{aligned}&\big\|\mathbf{M}_2\mathbf{S}(\mathbf{V}_1^T\mathbf{S})^\dagger\big\|_\xi^2 \;=\; \big\|\mathbf{M}_2\mathbf{S}(\mathbf{V}_1^T\mathbf{S})^T(\mathbf{V}_1^T\mathbf{S}\mathbf{S}^T\mathbf{V}_1)^\dagger\big\|_\xi^2\\ &\le\; \big\|\mathbf{M}_2\mathbf{S}\mathbf{S}^T\mathbf{V}_1\big\|_\xi^2\,\big\|(\mathbf{V}_1^T\mathbf{S}\mathbf{S}^T\mathbf{V}_1)^\dagger\big\|_2^2 \;=\; \big\|\mathbf{M}_2\mathbf{S}\mathbf{S}^T\mathbf{V}_1^T\big\|_\xi^2\,\sigma_{\min}^{-2}(\mathbf{V}_1^T\mathbf{S}\mathbf{S}^T\mathbf{V}_1),\end{aligned}$$

by which the lemma follows. ■

D.2 Uniform Sampling Bound

Theorem 22 shows that uniform sampling can be applied to randomized SVD. Specifically, if $\mathbf{C}$ consists of $c=\mathcal{O}(k\mu_k/\epsilon+k\mu_k\log k)$ uniformly sampled columns of $\mathbf{A}$, then $\|\mathbf{A}-\mathcal{P}_{\mathbf{C},k}(\mathbf{A})\|_F^2\le(1+\epsilon)\|\mathbf{A}-\mathbf{A}_k\|_F^2$ holds with high probability, where μ_k is the column coherence of $\mathbf{A}_k$.

Theorem 22 *Let $\mathbf{A}\in\mathbb{R}^{m\times n}$ be any fixed matrix, k ($\ll m,n$) be the target rank, and μ_k be the column coherence of $\mathbf{A}_k$. Let $\mathbf{S}\in\mathbb{R}^{m\times c}$ be a uniform sampling matrix and $\mathbf{C}=\mathbf{A}\mathbf{S}\in\mathbb{R}^{m\times c}$. When $c\ge\mu_k k\cdot\max\{20\log(20k),\,45/\epsilon\}$,*

$$\min_{\mathrm{rank}(\mathbf{X})\le k}\big\|\mathbf{A}-\mathbf{C}\mathbf{X}\big\|_F^2 \;\le\; \big(1+\epsilon\big)\,\|\mathbf{A}-\mathbf{A}_k\|_F^2.$$

holds with probability at least 0.9.

Proof Let $\mathbf{V}_k\in\mathbb{R}^{n\times k}$ contain the top k right singular vectors of $\mathbf{A}$. Obviously μ_k is the row coherence of $\mathbf{V}_k$. We apply Theorem 18 with $\delta=0.05$, $\eta=1/3$, $s=20\mu_k k\log(20k)$ and obtain

$$\mathbb{P}\Big\{\sigma_{\min}(\mathbf{V}_k^T\mathbf{S}\mathbf{S}^T\mathbf{V}_k)\;\le\;2/3\Big\}\;\le\;0.05.$$

We then apply Theorem 20 with $\delta=0.05$ and $s=45\mu_k k/\epsilon$ and obtain

$$\mathbb{P}\Big\{\big\|\mathbf{U}^T\mathbf{S}^T\mathbf{S}(\mathbf{A}-\mathbf{A}_k)\big\|_F^2\;\ge\;\frac{4}{9}\epsilon\|\mathbf{A}-\mathbf{A}_k\|_F^2\Big\}\;\le\;0.05.$$

It follows from Lemma 21 that

$$\begin{aligned}\min_{\mathrm{rank}(\mathbf{X})\le k}\|\mathbf{A}-\mathbf{C}\mathbf{X}\|_F^2 \;&\le\; \|\mathbf{A}-\mathbf{A}_k\|_F^2+\sigma_{\min}^{-2}(\mathbf{V}_k^T\mathbf{S}\mathbf{S}^T\mathbf{V}_k)\big\|\mathbf{U}^T\mathbf{S}^T\mathbf{S}(\mathbf{A}-\mathbf{A}_k)\big\|_F^2\\ &\le\; (1+\epsilon)\|\mathbf{A}-\mathbf{A}_k\|_F^2,\end{aligned}$$

where the latter inequality holds with probability at least 0.9 (due to the union bound). ■

D.3 The Uniform+Adaptive Column Selection Algorithm

In fact, running adaptive sampling only once yields a column sampling algorithm with $1+\epsilon$ bound for any $m\times n$ matrix. We call it the uniform+adaptive column selection algorithm, which is a part of the uniform+adaptive^2 algorithm. Though the algorithm is irrelevant to this work, we describe it in the following for it is of independent interest.

Theorem 23 (The Uniform+Adaptive Algorithm) *Given an $m\times n$ matrix $\mathbf{A}$, we sample $c_1=20\mu_k k\log(20k)$ columns by uniform sampling to form $\mathbf{C}_1$ and sample additional $c_2=17.5k/\epsilon$ columns by adaptive sampling to form $\mathbf{C}_2$. Let $\mathbf{C}=[\mathbf{C}_1,\mathbf{C}_2]$. Then the inequality*

$$\min_{\mathrm{rank}(\mathbf{X})\le k}\big\|\mathbf{A}-\mathbf{C}\mathbf{X}\big\|_F^2\;\le\;(1+\epsilon)\big\|\mathbf{A}-\mathbf{A}_k\big\|_F^2$$

holds with probability at least 0.8.

Proof We apply Theorem 22 with $\epsilon=0.75$ and $c_1=20\mu_k k\log(20k)$ and obtain that

$$\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\|_F^2\;\le\;1.75\|\mathbf{A}-\mathbf{A}_k\|_F^2$$

holds with probability at least 0.9.

Deshpande et al. (2006) showed that

$$\mathbb{E}\Big[\min_{\mathrm{rank}(\mathbf{X})\le k}\big\|\mathbf{A}-\mathbf{C}\mathbf{X}\big\|_F^2-\|\mathbf{A}-\mathbf{A}_k\|_F^2\Big]\;\le\;\frac{k}{c_2}\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\|_F^2.$$

It follows from Markov's inequality that

$$\min_{\mathrm{rank}(\mathbf{X})\le k}\big\|\mathbf{A}-\mathbf{C}\mathbf{X}\big\|_F^2-\|\mathbf{A}-\mathbf{A}_k\|_F^2\;\le\;\frac{k}{\delta_2 c_2}\|\mathbf{A}-\mathbf{C}_1\mathbf{C}_1^\dagger\mathbf{A}\|_F^2$$

holds with probability at least $1-\delta_2$. We let $\delta_2=0.1$ and $c_2=17.5k/\epsilon$, and it follows that

$$\min_{\mathrm{rank}(\mathbf{X})\le k}\big\|\mathbf{A}-\mathbf{C}\mathbf{X}\big\|_F^2\;\le\;(1+\epsilon)\|\mathbf{A}-\mathbf{A}_k\|_F^2$$

holds with probability at least 0.8. ■

D.4 Proof of Theorem 4

We sample $c_1=20\mu_k k\log(20k)$ columns by uniform sampling to form $\mathbf{C}_1$ and sample additional $c_2=17.5k/\epsilon$ columns by adaptive sampling to form $\mathbf{C}_2$. Let $\hat{\mathbf{C}}=[\mathbf{C}_1,\mathbf{C}_2]$. It follows from Theorem 23 that

$$\big\|\mathbf{K}-\mathcal{P}_{\hat{\mathbf{C}},k}(\mathbf{K})\big\|_F^2\;\le\;(1+\epsilon)\big\|\mathbf{K}-\mathbf{K}_k\big\|_F^2$$

holds with probability at least 0.8.

Let $\mathbf{C} = [\hat{\mathbf{C}}, \mathbf{C}_3]$ where $\mathbf{C}_3$ consists of c_3 columns of $\mathbf{K}$ chosen by adaptive sampling. Then

$$\begin{aligned}\mathbb{E}\big\|\mathbf{K} - \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 &\leq \mathbb{E}\big\|\mathbf{K} - \hat{\mathbf{C}}\hat{\mathbf{C}}^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 \\ &\leq \Big(1+\frac{k}{c_3}\Big)\big\|\mathbf{K} - \mathcal{P}_{\hat{\mathbf{C}},k}(\mathbf{K})\big\|_F^2,\end{aligned}$$

where the former inequality follows from Lemma 17 of Wang and Zhang (2013), and the latter inequality follows from (26). It follows from Markov's inequality that

$$\big\|\mathbf{K} - \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 - \big\|\mathbf{K} - \mathcal{P}_{\hat{\mathbf{C}},k}(\mathbf{K})\big\|_F^2 \leq \frac{k}{\delta_3 c_3}\big\|\mathbf{K} - \mathcal{P}_{\hat{\mathbf{C}},k}(\mathbf{K})\big\|_F^2$$

holds with probability at least $1-\delta_3$. We set $\delta_3 = 0.1$ and $c_3 = 10k/\epsilon$, Then

$$\big\|\mathbf{K} - \mathbf{C}\mathbf{C}^\dagger\mathbf{K}(\mathbf{C}^T)^\dagger\mathbf{C}^T\big\|_F^2 \leq (1+\epsilon)\big\|\mathbf{K} - \mathcal{P}_{\hat{\mathbf{C}},k}(\mathbf{K})\big\|_F^2 \leq (1+\epsilon)^2\big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2,$$

where the former inequality holds with probability $1-\delta_3$, and the latter inequality holds with probability $0.8 - \delta_3 = 0.7$.

Appendix E. Proof of Theorem 6

In Section E.1 we derive the solution to the optimization problem (14). In Section E.2 we prove that the solutions are global optimum. In Section E.3 we prove that the resulting solution is positive (semi)definite when $\mathbf{K}$ is positive (semi)definite.

E.1 Solution to the Optimization Problem (14)

We denote the objective function of the optimization problem (14) by

$$f(\mathbf{U},\delta) \;=\; \big\|\mathbf{K} - \bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T - \delta\mathbf{I}_n\big\|_F^2.$$

We take the derivative of $f(\mathbf{U},\delta)$ w.r.t. $\mathbf{U}$ to be zero

$$\begin{aligned}\frac{\partial f(\mathbf{U},\delta)}{\partial \mathbf{U}} &= \frac{\partial}{\partial \mathbf{U}}\mathrm{tr}(\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T - 2\mathbf{K}\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T + 2\delta\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T) \\ &= 2\bar{\mathbf{C}}^T\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T\bar{\mathbf{C}} - 2\bar{\mathbf{C}}^T\mathbf{K}\bar{\mathbf{C}} + 2\delta\bar{\mathbf{C}}^T\bar{\mathbf{C}} \;=\; \mathbf{0},\end{aligned}$$

and obtain the solution

$$\begin{aligned}\mathbf{U}^{\mathrm{ss}} &= (\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger(\bar{\mathbf{C}}^T\mathbf{K}\bar{\mathbf{C}} - \delta^{\mathrm{ss}}\bar{\mathbf{C}}^T\bar{\mathbf{C}})(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger \\ &= (\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger\bar{\mathbf{C}}^T\mathbf{K}\bar{\mathbf{C}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger - \delta^{\mathrm{ss}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger\bar{\mathbf{C}}^T\bar{\mathbf{C}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger \\ &= \bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T - \delta^{\mathrm{ss}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger.\end{aligned}$$

Similarly, we take the derivative of $f(\mathbf{U},\delta)$ w.r.t. δ to be zero

$$\frac{\partial f(\mathbf{U},\delta)}{\partial \delta} \;=\; \frac{\partial}{\partial\delta}\mathrm{tr}(\delta^2\mathbf{I}_n - 2\delta\mathbf{K} + 2\delta\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T) \;=\; 2n\delta - 2\mathrm{tr}(\mathbf{K}) + 2\mathrm{tr}(\bar{\mathbf{C}}\mathbf{U}\bar{\mathbf{C}}^T) = 0,$$

and it follows that

$$\begin{aligned}\delta^{\mathrm{ss}} &= \frac{1}{n}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}(\bar{\mathbf{C}}\mathbf{U}^{\mathrm{ss}}\bar{\mathbf{C}}^T)\Big) \\ &= \frac{1}{n}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}\Big(\bar{\mathbf{C}}\bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T\bar{\mathbf{C}}^T\Big) + \delta^{\mathrm{ss}}\mathrm{tr}\Big(\bar{\mathbf{C}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger\bar{\mathbf{C}}^T\Big)\Big) \\ &= \frac{1}{n}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}\Big(\bar{\mathbf{C}}^T\bar{\mathbf{C}}\bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T\Big) + \delta^{\mathrm{ss}}\mathrm{tr}\Big(\bar{\mathbf{C}}\bar{\mathbf{C}}^\dagger\Big)\Big) \\ &= \frac{1}{n}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}\Big(\bar{\mathbf{C}}^T\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T\Big) + \delta^{\mathrm{ss}}\mathrm{rank}(\bar{\mathbf{C}})\Big),\end{aligned}$$

and thus

$$\delta^{\mathrm{ss}} \;=\; \frac{1}{n-\mathrm{rank}(\bar{\mathbf{C}})}\Big(\mathrm{tr}(\mathbf{K}) - \mathrm{tr}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}\bar{\mathbf{C}}\big)\Big).$$

E.2 Proof of Optimality

The Hessian matrix of $f(\mathbf{U},\delta)$ w.r.t. $(\mathbf{U},\delta)$ is

$$\mathbf{H} \;=\; \begin{bmatrix}\dfrac{\partial^2 f(\mathbf{U},\delta)}{\partial\mathsf{vec}(\mathbf{U})\partial\mathsf{vec}(\mathbf{U})^T} & \dfrac{\partial^2 f(\mathbf{U},\delta)}{\partial\mathsf{vec}(\mathbf{U})\partial\delta} \\ \dfrac{\partial^2 f(\mathbf{U},\delta)}{\partial\delta\partial\mathsf{vec}(\mathbf{U})^T} & \dfrac{\partial^2 f(\mathbf{U},\delta)}{\partial\delta^2}\end{bmatrix} \;=\; 2\begin{bmatrix}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})\otimes(\bar{\mathbf{C}}^T\bar{\mathbf{C}}) & \mathsf{vec}(\bar{\mathbf{C}}^T\bar{\mathbf{C}}) \\ \mathsf{vec}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^T & n\end{bmatrix}.$$

Here $\otimes$ denotes the Kronecker product, and $\mathsf{vec}(\mathbf{A})$ denotes the vectorization of the matrix $\mathbf{A}$ formed by stacking the columns of $\mathbf{A}$ into a single column vector. For any $\mathbf{X}\in\mathbb{R}^{c\times c}$ and $b\in\mathbb{R}$, we let

$$\begin{aligned}q(\mathbf{X},b) &= \begin{bmatrix}\mathsf{vec}(\mathbf{X})^T & b\end{bmatrix}\mathbf{H}\begin{bmatrix}\mathsf{vec}(\mathbf{X}) \\ b\end{bmatrix} \\ &= \mathsf{vec}(\mathbf{X})^T\big((\bar{\mathbf{C}}^T\bar{\mathbf{C}})\otimes(\bar{\mathbf{C}}^T\bar{\mathbf{C}})\big)\mathsf{vec}(\mathbf{X}) + 2b\,\mathsf{vec}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^T\mathsf{vec}(\mathbf{X}) + nb^2 \\ &= \mathsf{vec}(\mathbf{X})^T\mathsf{vec}\big((\bar{\mathbf{C}}^T\bar{\mathbf{C}})\mathbf{X}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})\big) + 2b\,\mathsf{vec}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^T\mathsf{vec}(\mathbf{X}) + nb^2 \\ &= \mathrm{tr}(\mathbf{X}^T\bar{\mathbf{C}}^T\bar{\mathbf{C}}\mathbf{X}\bar{\mathbf{C}}^T\bar{\mathbf{C}}) + 2b\,\mathrm{tr}(\bar{\mathbf{C}}^T\bar{\mathbf{C}}\mathbf{X}) + nb^2.\end{aligned}$$

Let $\bar{\mathbf{C}}\mathbf{X}\bar{\mathbf{C}}^T = \mathbf{Y}\in\mathbb{R}^{n\times n}$. Then

$$\begin{aligned}q(\mathbf{X},b) &= \mathrm{tr}(\bar{\mathbf{C}}\mathbf{X}^T\bar{\mathbf{C}}^T\bar{\mathbf{C}}\mathbf{X}\bar{\mathbf{C}}^T) + 2b\,\mathrm{tr}(\bar{\mathbf{C}}\mathbf{X}\bar{\mathbf{C}}^T) + nb^2 \\ &= \mathrm{tr}(\mathbf{Y}^T\mathbf{Y}) + 2b\,\mathrm{tr}(\mathbf{Y}) + nb^2 \\ &= \sum_{i=1}^n\sum_{j=1}^n y_{ij}^2 + 2b\sum_{l=1}^n y_{ll} + nb^2 \\ &= \sum_{i\neq j} y_{ij}^2 + \sum_{l=1}^n (y_{ll}+b)^2 \\ &\geq 0,\end{aligned}$$

which shows that the Hessian matrix $\mathbf{H}$ is SPSD. Hence $f(\mathbf{U}^{\mathrm{ss}},\delta^{\mathrm{ss}})$ is the global minimum of f.

E.3 Proof of Positive (Semi)Definite

We denote the thin SVD of $\bar{\mathbf{C}}$ by $\bar{\mathbf{C}} = \mathbf{U}_{\bar{\mathbf{C}}}\mathbf{\Sigma}_{\bar{\mathbf{C}}}\mathbf{V}_{\bar{\mathbf{C}}}^T$ and let $\mathbf{U}_{\bar{\mathbf{C}}}^\perp$ be the orthogonal complement of $\mathbf{U}_{\bar{\mathbf{C}}}$. The approximation is

$$\begin{aligned}\tilde{\mathbf{K}} &= \bar{\mathbf{C}}\mathbf{U}^{\text{ss}}\bar{\mathbf{C}}^T + \delta^{\text{ss}}\mathbf{I}_n = \bar{\mathbf{C}}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T - \delta^{\text{ss}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger\big)\bar{\mathbf{C}}^T + \delta^{\text{ss}}\mathbf{I}_n \\ &= \bar{\mathbf{C}}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}(\bar{\mathbf{C}}^\dagger)^T\big)\bar{\mathbf{C}}^T + \delta^{\text{ss}}\big(\mathbf{I}_n - \bar{\mathbf{C}}(\bar{\mathbf{C}}^T\bar{\mathbf{C}})^\dagger\bar{\mathbf{C}}^T\big) \\ &= \mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T + \delta^{\text{ss}}\big(\mathbf{I}_n - \mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T\big) \\ &= \mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T + \delta^{\text{ss}}\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T. \end{aligned} \tag{29}$$

The first term is SPSD because $\mathbf{K}$ is SPSD. The second term is SPSD if δ^{ss} is nonnegative. We have

$$\begin{aligned}\delta^{\text{ss}} &= \text{tr}(\mathbf{K}) - \text{tr}\big(\bar{\mathbf{C}}^\dagger\mathbf{K}\bar{\mathbf{C}}\big) = \text{tr}(\mathbf{K}) - \text{tr}\big(\bar{\mathbf{C}}\bar{\mathbf{C}}^\dagger\mathbf{K}\big) = \text{tr}\big(\mathbf{K} - \mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T\mathbf{K}\big) \\ &= \text{tr}\big[\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\mathbf{K}\big] = \text{tr}\big[\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\mathbf{K}\big] = \text{tr}\big[\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\big] \geq 0.\end{aligned}$$

To this end, we have shown SPSD.

Then we assume $\mathbf{K}$ is positive definite and prove that its approximation formed by SS is also positive definite. Since $\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}}^\perp(\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T$ is SPSD, its trace is zero only when it is all-zeros, which is equivalent to $\mathbf{K} = \mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}}\mathbf{U}_{\bar{\mathbf{C}}}^T$. However, this cannot hold if $\mathbf{K}$ has full rank. Therefore $\delta^{\text{ss}} > 0$ holds when $\tilde{\mathbf{K}}$ is positive definite. When $\mathbf{K}$ is positive definite, the $c \times c$ matrix $\mathbf{U}_{\bar{\mathbf{C}}}^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}}$ is also positive definite. It follows from (29) that

$$\tilde{\mathbf{K}} = \bar{\mathbf{C}}\mathbf{U}^{\text{ss}}\bar{\mathbf{C}}^T + \delta^{\text{ss}}\mathbf{I}_n = \begin{bmatrix}\mathbf{U}_{\bar{\mathbf{C}}} & \mathbf{U}_{\bar{\mathbf{C}}}^\perp\end{bmatrix}\begin{bmatrix}\mathbf{U}_{\bar{\mathbf{C}}}^T\mathbf{K}\mathbf{U}_{\bar{\mathbf{C}}} & \mathbf{0} \\ \mathbf{0} & \delta^{\text{ss}}\mathbf{I}_{n-c}\end{bmatrix}\begin{bmatrix}\mathbf{U}_{\bar{\mathbf{C}}}^T \\ (\mathbf{U}_{\bar{\mathbf{C}}}^\perp)^T\end{bmatrix}.$$

Here the block diagonal matrix is positive definite, and thus $\tilde{\mathbf{K}}$ is positive definite.

Appendix F. Proof of Theorem 8 and Theorem 9

Directly analyzing the theoretical error bound of the SS model is not easy, so we formulate a variant of SS called the inexact spectral shifting (ISS) model and instead analyze the error bound of ISS. We define ISS in Section F.1 and prove Theorem 8 and Theorem 9 in Section F.2 and F.3, respectively. In the following we let $\tilde{\mathbf{K}}_c^{\text{ss}}$ and $\tilde{\mathbf{K}}_c^{\text{iss}}$ be respectively the approximation formed by the two models.

F.1 The ISS Model

ISS is defined by

$$\tilde{\mathbf{K}}_c^{\text{iss}} = \bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T + \delta\mathbf{I}_n, \tag{30}$$

where $\delta > 0$ is the spectral shifting term, and $\bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T$ is the prototype model of $\bar{\mathbf{K}} = \mathbf{K} - \delta\mathbf{I}_n$. It follows from Theorem 6 that ISS is less accurate than SS in that

$$\big\|\mathbf{K} - \tilde{\mathbf{K}}_c^{\text{ss}}\big\|_F \leq \big\|\mathbf{K} - \tilde{\mathbf{K}}_c^{\text{iss}}\big\|_F,$$

thus the error bounds of ISS still hold if $\tilde{\mathbf{K}}_c^{\text{iss}}$ is replaced by $\tilde{\mathbf{K}}_c^{\text{ss}}$.

We first show how to set the spectral shifting term δ. Since $\bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T$ is the approximation to $\bar{\mathbf{K}}$ formed by the prototype model, it can holds with high probability that

$$\big\|\mathbf{K} - \tilde{\mathbf{K}}_c^{\text{iss}}\big\|_F = \big\|\mathbf{K} - \bar{\delta}\mathbf{I}_n - \bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T\big\|_F = \big\|\bar{\mathbf{K}} - \bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T\big\|_F \leq \eta\big\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\big\|_F$$

for some error parameter η. Apparently, for fixed k, the smaller the error $\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\|_F$ is, the tighter error bound the ISS has; if $\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\|_F \leq \|\mathbf{K} - \mathbf{K}_k\|_F$, then ISS has a better error bound than the prototype model. Therefore, our goal is to make $\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\|_F$ as small as possible, so we formulate the following optimization problem to compute δ:

$$\min_{\delta \geq 0} \big\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\big\|_F^2; \qquad \text{s.t. } \bar{\mathbf{K}} = \mathbf{K} - \delta\mathbf{I}_n.$$

However, since $\bar{\mathbf{K}}$ is in general indefinite, it requires all of the eigenvalues of $\mathbf{K}$ to solve the problem exactly. Since computing the full eigenvalue decomposition is expensive, we attempt to relax the problem. Considering that

$$\big\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\big\|_F^2 = \min_{|\mathcal{J}|=n-k}\sum_{j\in\mathcal{J}}\big(\sigma_j(\mathbf{K}) - \delta\big)^2 \leq \sum_{j=k+1}^{n}\big(\sigma_j(\mathbf{K}) - \delta\big)^2, \tag{31}$$

we seek to minimize the upper bound of $\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\|_F^2$, which is the right-hand side of (31), to compute δ, leading to the solution

$$\bar{\delta} = \frac{1}{n-k}\sum_{j=k+1}^{n}\sigma_j(\mathbf{K}) = \frac{1}{n-k}\bigg(\text{tr}(\mathbf{K}) - \sum_{j=1}^{k}\sigma_j(\mathbf{K})\bigg). \tag{32}$$

If we choose $\delta = 0$, then ISS degenerates to the prototype model.

F.2 Proof of Theorem 8

Theorem 6 indicates that ISS is less accurate than SS, thus

$$\big\|\mathbf{K} - \tilde{\mathbf{K}}_c^{\text{ss}}\big\|_F \leq \big\|\mathbf{K} - \tilde{\mathbf{K}}_c^{\text{iss}}\big\|_F = \big\|\mathbf{K} - \bar{\delta}\mathbf{I}_n - \bar{\mathbf{C}}\bar{\mathbf{C}}^\dagger\bar{\mathbf{K}}(\bar{\mathbf{C}}^\dagger)^T\bar{\mathbf{C}}^T\big\|_F = \big\|\bar{\mathbf{K}} - \bar{\mathbf{C}}\bar{\mathbf{C}}^\dagger\bar{\mathbf{K}}(\bar{\mathbf{C}}^\dagger)^T\bar{\mathbf{C}}^T\big\|_F.$$

Theorem 8 follows from the above inequality and the following theorem.

Theorem 24 *Let $\mathbf{K}$ be any $n \times n$ SPSD matrix, $\bar{\delta}$ be defined in (32), and $\bar{\mathbf{K}} = \mathbf{K} - \delta\mathbf{I}_n$. Then for any $\delta \in (0, \bar{\delta}]$, the following inequality holds:*

$$\big\|\bar{\mathbf{K}} - \bar{\mathbf{K}}_k\big\|_F^2 \leq \big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2.$$

Proof Since the righthand side of (31) is convex and $\bar{\delta}$ is the minimizer of the righthand of (31), for any $\delta \in (0, \bar{\delta}]$, it holds that

$$\sum_{j=k+1}^{n}\big(\sigma_j(\mathbf{K}) - \delta\big)^2 \leq \sum_{j=k+1}^{n}\big(\sigma_j(\mathbf{K}) - 0\big)^2 = \big\|\mathbf{K} - \mathbf{K}_k\big\|_F^2.$$

Then the theorem follows by the inequality in (31). ■

F.3 Proof of Theorem 9

Since $\|\mathbf{K}-\tilde{\mathbf{K}}_c^{\mathrm{ss}}\|_F \le \|\mathbf{K}-\tilde{\mathbf{K}}_c^{\mathrm{iss}}\|_F$, Theorem 9 follows from Theorem 25.

Theorem 25 *Suppose there is a sketching matrix $\mathbf{P}\in\mathbb{R}^{n\times c}$ such that for any $n\times n$ symmetric matrix $\mathbf{S}$ and target rank k ($\ll n$), by forming the sketch $\mathbf{C}=\mathbf{SP}$, the prototype model satisfies the error bound*

$$\left\|\mathbf{S}-\mathbf{CC}^{\dagger}\mathbf{S}(\mathbf{C}^{\dagger})^T\mathbf{C}^T\right\|_F^2 \le \eta\left\|\mathbf{S}-\mathbf{S}_k\right\|_F^2.$$

Let $\mathbf{K}$ be any $n\times n$ SPSD matrix, $\bar{\delta}$ defined in (32) be the initial spectral shifting term, and $\tilde{\mathbf{K}}_c^{iss}$ be the ISS approximation defined in (30). Then

$$\left\|\mathbf{K}-\tilde{\mathbf{K}}_c^{iss}\right\|_F^2 \le \eta\left(\left\|\mathbf{K}-\mathbf{K}_k\right\|_F^2-\frac{\left[\sum_{i=k+1}^n\lambda_i(\mathbf{K})\right]^2}{n-k}\right).$$

Proof The error incurred by SS is

$$\begin{aligned}\left\|\mathbf{K}-\tilde{\mathbf{K}}_c^{\mathrm{ss}}\right\|_F^2 &= \left\|\left(\bar{\mathbf{K}}+\bar{\delta}\mathbf{I}_n\right)-\left(\bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T+\bar{\delta}\mathbf{I}_n\right)\right\|_F^2 = \left\|\bar{\mathbf{K}}-\bar{\mathbf{C}}\bar{\mathbf{U}}\bar{\mathbf{C}}^T\right\|_F^2\\ &\le \eta\left\|\bar{\mathbf{K}}-\bar{\mathbf{K}}_k\right\|_F^2 = \eta\sum_{i=k+1}^n\sigma_i^2\left(\bar{\mathbf{K}}\right) = \eta\sum_{i=k+1}^n\lambda_i(\bar{\mathbf{K}}^2).\end{aligned}$$

Here the inequality follows from the assumption. The i-th largest eigenvalue of $\bar{\mathbf{K}}$ is $\lambda_i(\mathbf{K})-\bar{\delta}$, so the n eigenvalues of $\bar{\mathbf{K}}^2$ are all in the set $\{(\lambda_i(\mathbf{K})-\bar{\delta})^2\}_{i=1}^n$. The sum of the smallest $n-k$ of the n eigenvalues of $\bar{\mathbf{K}}^2$ must be less than or equal to the sum of any $n-k$ of the eigenvalues, thus we have

$$\begin{aligned}\sum_{i=k+1}^n\lambda_i(\bar{\mathbf{K}}^2) &\le \sum_{i=k+1}^n\left(\lambda_i(\mathbf{K})-\bar{\delta}\right)^2\\ &= \sum_{i=k+1}^n\lambda_i^2(\mathbf{K})-2\sum_{i=k+1}^n\bar{\delta}\lambda_i(\mathbf{K})+(n-k)(\bar{\delta})^2\\ &= \left\|\mathbf{K}-\mathbf{K}_k\right\|_F^2-\frac{1}{n-k}\left[\sum_{i=k+1}^n\lambda_i(\mathbf{K})\right]^2,\end{aligned}$$

by which the theorem follows. ■

Appendix G. Proof of Theorem 11

Proof Let $\tilde{\mathbf{K}}=\mathbf{Q}(\mathbf{Q}^T\mathbf{K})_k$, where $\mathbf{Q}$ is defined in Line 4 in Algorithm 5. Boutsidis et al. (2014) showed that

$$\mathbb{E}\|\mathbf{K}-\tilde{\mathbf{K}}\|_F^2 \le (1+k/l)\left\|\mathbf{K}-\mathbf{K}_k\right\|_F^2, \tag{33}$$

where the expectation is taken w.r.t. the random Gaussian matrix $\mathbf{\Omega}$.

It follows from Lemma 26 that

$$\|\boldsymbol{\sigma}_{\mathbf{K}}-\boldsymbol{\sigma}_{\tilde{\mathbf{K}}}\|_2^2 \le \|\mathbf{K}-\tilde{\mathbf{K}}\|_F^2,$$

where $\boldsymbol{\sigma}_{\mathbf{K}}$ and $\boldsymbol{\sigma}_{\tilde{\mathbf{K}}}$ contain the singular values in a descending order. Since $\tilde{\mathbf{K}}$ has a rank at most k, the $k+1$ to n entries of $\boldsymbol{\sigma}_{\tilde{\mathbf{K}}}$ are zero. We split $\boldsymbol{\sigma}_{\mathbf{K}}$ and $\boldsymbol{\sigma}_{\tilde{\mathbf{K}}}$ into vectors of length k and $n-k$:

$$\boldsymbol{\sigma}_{\mathbf{K}} = \begin{bmatrix}\boldsymbol{\sigma}_{\mathbf{K},k}\\ \boldsymbol{\sigma}_{\mathbf{K},-k}\end{bmatrix} \quad\text{and}\quad \boldsymbol{\sigma}_{\tilde{\mathbf{K}}} = \begin{bmatrix}\boldsymbol{\sigma}_{\tilde{\mathbf{K}},k}\\ \mathbf{0}\end{bmatrix}$$

and thus

$$\|\boldsymbol{\sigma}_{\mathbf{K},k}-\boldsymbol{\sigma}_{\tilde{\mathbf{K}},k}\|_2^2+\|\boldsymbol{\sigma}_{\mathbf{K},-k}\|_2^2 \le \|\mathbf{K}-\tilde{\mathbf{K}}\|_F^2. \tag{34}$$

Since $\|\boldsymbol{\sigma}_{\mathbf{K},-k}\|_2^2=\|\mathbf{K}-\mathbf{K}_k\|_F^2$, it follows from (33) and (34) that

$$\mathbb{E}\|\boldsymbol{\sigma}_{\mathbf{K},k}-\boldsymbol{\sigma}_{\tilde{\mathbf{K}},k}\|_2^2 \le \frac{k}{l}\|\boldsymbol{\sigma}_{\mathbf{K},-k}\|_2^2.$$

Since $\|\mathbf{x}\|_2\le\|\mathbf{x}\|_1\le\sqrt{k}\|\mathbf{x}\|_2$ for any $\mathbf{x}\in\mathbb{R}^k$, we have that

$$\mathbb{E}\left\|\boldsymbol{\sigma}_{\mathbf{K},k}-\boldsymbol{\sigma}_{\tilde{\mathbf{K}},k}\right\|_1 \le \frac{k}{\sqrt{l}}\left\|\boldsymbol{\sigma}_{\mathbf{K},-k}\right\|_1.$$

Then it follows from (32) and Line 6 in Algorithm 5 that

$$\begin{aligned}\mathbb{E}\left|\bar{\delta}-\tilde{\delta}\right| &= \mathbb{E}\left[\frac{1}{n-k}\left|\sum_{i=1}^k\sigma_i(\mathbf{K})-\sum_{i=1}^k\sigma_i(\tilde{\mathbf{K}})\right|\right]\\ &\le \frac{1}{n-k}\mathbb{E}\left\|\boldsymbol{\sigma}_{\mathbf{K},k}-\boldsymbol{\sigma}_{\tilde{\mathbf{K}},k}\right\|_1 \le \frac{k}{\sqrt{l}}\frac{1}{n-k}\left\|\boldsymbol{\sigma}_{\mathbf{K},-k}\right\|_1 = \frac{k}{\sqrt{l}}\bar{\delta}.\end{aligned}$$

■

Lemma 26 *Let $\mathbf{A}$ and $\mathbf{B}$ be $n\times n$ matrices and $\boldsymbol{\sigma}_{\mathbf{A}}$ and $\boldsymbol{\sigma}_{\mathbf{B}}$ contain the singular values in a descending order. Then we have that*

$$\|\boldsymbol{\sigma}_{\mathbf{A}}-\boldsymbol{\sigma}_{\mathbf{B}}\|_2^2 \le \|\mathbf{A}-\mathbf{B}\|_F^2.$$

Proof It is easy to show that

$$\begin{aligned}\|\mathbf{A}-\mathbf{B}\|_F^2 &= \mathrm{tr}(\mathbf{A}^T\mathbf{A})+\mathrm{tr}(\mathbf{B}^T\mathbf{B})-2\mathrm{tr}(\mathbf{A}^T\mathbf{B})\\ &= \sum_{i=1}^n\sigma_i^2(\mathbf{A})+\sum_{i=1}^n\sigma_i^2(\mathbf{B})-2\mathrm{tr}(\mathbf{A}^T\mathbf{B}).\end{aligned} \tag{35}$$

We also have

$$\mathrm{tr}(\mathbf{A}^T\mathbf{B}) \le \sum_{i=1}^n\sigma_i(\mathbf{A}^T\mathbf{B}) \le \sum_{i=1}^n\sigma_i(\mathbf{A})\sigma_i(\mathbf{B}), \tag{36}$$

where the first inequality follows from Theorem 3.3.13 of Horn and Johnson and the second inequality follows from Theorem 3.3.14 of Horn and Johnson. Combining (35) and (36) we have that

$$\|\mathbf{A}-\mathbf{B}\|_F^2 \;\geq\; \sum_{i=1}^{n}\left(\sigma_i^2(\mathbf{A})+\sigma_i^2(\mathbf{B})-2\sigma_i(\mathbf{A})\sigma_i(\mathbf{B})\right) \;=\; \|\boldsymbol{\sigma}_{\mathbf{A}}-\boldsymbol{\sigma}_{\mathbf{B}}\|_2^2,$$

by which the theorem follows. ■

References

Adi Ben-Israel and Thomas N.E. Greville. *Generalized Inverses: Theory and Applications. Second Edition*. Springer, 2003. 31

Christos Boutsidis and David P. Woodruff. Optimal CUR matrix decompositions. *arXiv preprint arXiv:1405.7910*, 2014. 10, 34

Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014. 9, 10, 34, 39, 45

Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(2006): 225–247, 2006. 9, 40

Petros Drineas and Michael W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005. 2, 7

Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, September 2008. 38

Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13:3441–3472, 2012. 11

Alex Gittens and Michael W. Mahoney. Revisiting the nyström method for improved large-scale machine learning. In *International Conference on Machine Learning (ICML)*, 2013. 2, 7, 14

Ming Gu and Stanley C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996. 9

Venkatesan Guruswami and Ali Kemal Sinop. Optimal column-based low-rank matrix reconstruction. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2012. 9

Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011. 2, 7

Roger A. Horn and Charles R. Johnson. Topics in matrix analysis. 1991. *Cambridge University Presss, Cambridge*. 47

Rong Jin, Tianbao Yang, Mehrdad Mahdavi, Y Li, and Z Zhou. Improved bounds for the Nyström method with application to kernel classification. *IEEE Transactions on Information Theory*, 59(10):6939–6949, 2013. 7

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. On sampling-based approximate spectral decomposition. In *International Conference on Machine Learning (ICML)*, 2009. 3, 8

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012. 4, 7, 9, 10, 12, 22, 23

David Lopez-Paz, Suvrit Sra, Alex Smola, Zoubin Ghahramani, and Bernhard Schölkopf. Randomized nonlinear component analysis. In *International Conference on Machine Learning (ICML)*, 2014. 2

Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011. 9

Evert J. Nyström. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. *Acta Mathematica*, 54(1):185–204, 1930. 2, 6

Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems (NIPS)*. 2008. 2

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002. 1

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004. 1

John Shawe-Taylor, Christopher K. I. Williams, Nello Cristianini, and Jaz Kandola. On the eigenspectrum of the gram matrix and the generalisation error of kernel pca. *IEEE Transactions on Information Theory*, 51:2510–2522, 2005. 7

Si Si, Cho-Jui Hsieh, and Inderjit Dhillon. Memory efficient kernel approximation. In *International Conference on Machine Learning (ICML)*, pages 701–709, 2014. 4, 22, 23

G. W. Stewart. Four algorithms for the efficient computation of truncated pivoted QR approximations to a sparse matrix. *Numerische Mathematik*, 83(2):313–323, 1999. 9

Ameet Talwalkar and Afshin Rostamizadeh. Matrix coherence and the Nyström method. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010. 3, 8

Ameet Talwalkar, Sanjiv Kumar, Mehryar Mohri, and Henry Rowley. Large-scale SVD and manifold learning. *Journal of Machine Learning Research*, 14:3129–3152, 2013. 7

Joel A Tropp. An introduction to matrix concentration inequalities. *arXiv preprint arXiv:1501.01571*, 2015. 36

Shusen Wang and Zhihua Zhang. Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling. *Journal of Machine Learning Research*, 14:2729–2769, 2013. 2, 3, 4, 7, 9, 31, 35, 41

Shusen Wang, Zhihua Zhang, and Tong Zhang. Towards more efficient SPSD matrix approximation and CUR matrix decomposition. *arXiv preprint arXiv:1503.08395*, 2015. 3, 7, 8, 29

Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2001. 2, 6

David P Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014. 9, 36

Tianbao Yang, Yu-Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems (NIPS)*, 2012. 2

Zhihua Zhang. The matrix ridge approximation: algorithms and applications. *Machine Learning*, 97:227–258, 2014. 4

Journal of Machine Learning Research 17 (2016) 1-33 Submitted 7/14; Revised 3/15; Published 4/16

Combinatorial Multi-Armed Bandit and Its Extension to Probabilistically Triggered Arms*

Wei Chen† WEIC@MICROSOFT.COM
Microsoft
Beijing, China

Yajun Wang YAJUNW@MICROSOFT.COM
Microsoft
Sunnyvale, CA, U.S.A.

Yang Yuan YANGYUAN@CS.CORNELL.EDU
Cornell University
Ithaca, NY, U.S.A.

Qinshi Wang WANGQINSHI1995@GMAIL.COM
Tsinghua University
Beijing, China

Editor: Shie Mannor

Abstract

We define a general framework for a large class of combinatorial multi-armed bandit (CMAB) problems, where subsets of base arms with unknown distributions form *super arms*. In each round, a super arm is played and the base arms contained in the super arm are played and their outcomes are observed. We further consider the extension in which more base arms could be probabilistically triggered based on the outcomes of already triggered arms. The reward of the super arm depends on the outcomes of all played arms, and it only needs to satisfy two mild assumptions, which allow a large class of nonlinear reward instances. We assume the availability of an offline (α, β)-approximation oracle that takes the means of the outcome distributions of arms and outputs a super arm that with probability β generates an α fraction of the optimal expected reward. The objective of an online learning algorithm for CMAB is to minimize (α, β)-*approximation regret*, which is the difference in total expected reward between the $\alpha\beta$ fraction of expected reward when always playing the optimal super arm, and the expected reward of playing super arms according to the algorithm. We provide CUCB algorithm that achieves $O(\log n)$ distribution-dependent regret, where n is the number of rounds played, and we further provide distribution-independent bounds for a large class of reward functions. Our regret analysis is tight in that it matches the bound of UCB1 algorithm (up to a constant factor) for the classical MAB problem, and it significantly improves the regret bound in an earlier paper on combinatorial bandits with linear rewards. We apply our CMAB framework to two new applications, probabilistic maximum coverage (PMC) for online advertising and social influence maximization for viral marketing, both having nonlinear reward structures. In particular, application to social influence maximization requires our extension on probabilistically triggered arms.

Keywords: combinatorial multi-armed bandit, online learning, upper confidence bound, social influence maximization, online advertising

*. A preliminary version of this paper has appeared in ICML'2013 (Chen et al., 2013). The current version contains the extension of CMAB to accommodate probabilistically triggered arms and its application to social influence maximization. The research is partially supported by the National Natural Science Foundation of China (Grant No. 61433014).

†. Contact author: Wei Chen, Microsoft Research Asia, Microsoft Asia R&D Headquarters, Building 2 14-171, 5 Dan Ling Street, Haidian District, Beijing, China, 100080.

1. Introduction

Multi-armed bandit (MAB) is a problem extensively studied in statistics and machine learning. The classical version of the problem is formulated as a system of m arms (or machines), each having an unknown distribution of the reward with an unknown mean. The task is to repeatedly play these arms in multiple rounds so that the total expected reward is as close to the reward of the optimal arm as possible. An MAB algorithm needs to decide which arm to play in the next round given the outcomes of the arms played in the previous rounds. The metric for measuring the effectiveness of an MAB algorithm is its *regret*, which is the difference in the total expected reward between always playing the optimal arm (the arm with the largest expected reward) and playing arms according to the algorithm. The MAB problem and its solutions reflect the fundamental tradeoff between exploration and exploitation: whether one should try some arms that have not been played much (exploration) or one should stick to the arms that provide good reward so far (exploitation). Existing results show that one can achieve a regret of $O(\log n)$ when playing arms in n rounds, and this is asymptotically the best possible.

In many real-world applications, the setting is not the simple MAB one, but has a combinatorial nature among multiple arms and possibly non-linear reward functions. For example, consider the following online advertising scenario. A web site contains a set of web pages and has a set of users visiting the web site. An advertiser wants to place an advertisement on a set of selected web pages on the site, and due to his budget constraint, he can select at most k web pages. Each user visits a certain set of pages, and each visited page has one click-through probability for each user clicking the advertisement on the page, but the advertiser does not know these probabilities. The advertiser wants to repeatedly select sets of k web pages, observe the click-through data collected to learn the click-through probabilities, and maximize the number of users clicking his advertisement over time.

There are several new challenges raised by the above example. First, page-user pairs can be viewed as arms, but they are not played in isolation. Instead, these arms form certain combinatorial structures, namely bipartite graphs, and in each round, a set of arms (called a *super arm*) are played together. Second, the reward structure is not a simple linear function of the outcomes of all played arms but takes a more complicated form. In the above example, for all page-user pairs with the same user, the collective reward of these arms is either 1 if the user clicks the advertisement on at least one of the pages, or 0 if the user does not click the advertisement on any page. Third, even the offline optimization problem when the probabilities on all edges of the bipartite graph are known is still an NP-hard problem. Thus, the online learning algorithm needs to deal with combinatorial arm structures, nonlinear reward functions, and computational hardness of the offline optimization task.

Consider another example of viral marketing in online social networks. In an online social network such as Facebook, companies carry out viral marketing campaigns by engaging with

a certain set of seed users (e.g. providing free sample products to seed users), and hoping that these seed users could generate a cascade in the network promoting their products. The cascades follow certain stochastic diffusion model such as the independent cascade model (Kempe et al., 2003), but the influence probabilities on edges are not known in advance and have to be learned over time. Thus, the online learning task is to repeatedly select seed nodes in a social network, observe the cascading behavior of the viral information to learn influence probabilities between individuals in the social network, with the goal of maximizing the overall effectiveness of all viral cascades. Similar to the online advertising example given above, we can treat each edge in the social network as a base arm, and all outgoing edges from a seed set as a super arm, which is the unit of play. Besides sharing the same challenges such as the combinatorial arm structures, nonlinear reward functions, and computational hardness of the offline maximization task, this viral marketing task faces another challenge: in each round after some seed set is selected, the cascade from the seed set may probabilistically trigger more edges (or arms) in the network, and the reward of the cascade depends on all probabilistically or deterministically triggered arms.

A naive way to tackle both examples above is to treat every super arm as an arm and simply apply the classical MAB framework to solve the above combinatorial problems. However, such naive treatment has two issues. First, the number of super arms may be exponential to the problem instance size due to the combinatorial explosion, and thus classical MAB algorithms may need exponential number of steps just to go through all the super arms once. Second, after one super arm is played, in many cases, we can observe some information regarding the outcomes of the underlying arms, which may be shared by other super arms. However, this information is discarded in the classical MAB framework, making it less effective.

In this paper, we define a general framework for the *combinatorial multi-armed bandit (CMAB)* problem to address the above issues and cover a large class of combinatorial online learning problems in the stochastic setting, including the two examples given above. In the CMAB framework, we have a set of m base arms, whose outcomes follow certain unknown joint distribution. A super arm S is a subset of base arms. In each round, one super arm is played and all base arms contained in the super arm are played. To accommodate applications such as viral marketing, we allow that the play of a super arm S may further trigger more base arms probabilistically, and the triggering depends on the outcomes of the already played base arms in the current round. The reward of the round is determined by the outcomes of all triggered arms, which are observed as the feedback to the online learning algorithm. A CMAB algorithm needs to use these feedback information from the past rounds to decide the super arm to play in the next round.

The framework allows an arbitrary combination of arms into super arms. The reward function only needs to satisfy two mild assumptions (referred to as monotonicity and bounded smoothness), and thus covering a large class of nonlinear reward functions. We do not assume the direct knowledge on how super arms are formed from underlying arms or how the reward is computed. Instead, we assume the availability of an offline computation oracle that takes such knowledge as well as the expectations of outcomes of all arms as input and computes the optimal super arm with respect to the input.

Since many combinatorial problems are computationally hard, we further allow (randomized) approximation oracles with failure probabilities. In particular, we relax the oracle

to be an (α, β)-approximation oracle for some $\alpha, \beta \leq 1$, that is, with success probability β, the oracle could output a super arm whose expected reward is at least α fraction of the optimal expected reward. As a result, our regret metric is not comparing against the expected reward of playing the optimal super arm each time, but against the $\alpha\beta$ fraction of the optimal expected reward, since the offline oracle can only guarantee this fraction in expectation. We refer to this as the (α, β)-*approximation regret*.

For the general framework, we provide the CUCB (combinatorial upper confidence bound) algorithm, an extension to the UCB1 algorithm for the classical MAB problem (Auer et al., 2002a). We provide a rigorous analysis on the distribution-dependent regret of CUCB and show that it is still bounded by $O(\log n)$. Our analysis further allows us to provide a distribution-independent regret bound that works for arbitrary distributions of underlying arms, for a large class of CMAB instances. For the extension accommodating probabilistically triggered arms, we also provide distribution-dependent and -independent bounds with triggering probabilities as parameters.

We then apply our framework and provide solutions to two new bandit applications, the probabilistic maximum coverage problem for advertisement placement and social influence maximization for viral marketing. The offline versions of both problems are NP-hard, with constant approximation algorithms available. Both problems have nonlinear reward structures that cannot be handled by any existing work.

We also apply our result to combinatorial bandits with linear rewards, recently studied by Gai et al. (2012). We show that we significantly improve their distribution-dependent regret bound, even though we are covering a much larger class of combinatorial bandit instances. We also provide new distribution-independent bound not available in their paper (Gai et al., 2012).

This paper is an extension to our ICML'2013 paper (Chen et al., 2013), with explicit modeling of probabilistically triggered arms and their regret analysis for the CUCB algorithm. We correct an erroneous claim in that paper (Chen et al., 2013), which states that the original CMAB model and result without probabilistically triggered arms can be applied to the online learning task for social influence maximization. Our correction includes explicit modeling of probabilistically triggered arms in the CMAB framework, and significant reworking of the regret analysis to incorporate triggering probabilities in the analysis and the regret bounds.

In summary, our contributions include: (a) defining a general CMAB framework that encompasses a large class of nonlinear reward functions, (b) providing CUCB algorithm with a rigorous regret analysis as a general solution to this CMAB framework, (c) further generalizing our framework to accommodate probabilistically triggered base arms, and applying this framework to the social influence maximization problem, and (d) demonstrating that our general framework can be effectively applied to a number of practical combinatorial bandit problems, including ones with nonlinear rewards. Moreover, our framework provides a clean separation of the online learning task and the offline computation task: the oracle takes care of the offline computation task, which uses the domain knowledge of the problem instance, while our CMAB algorithm takes care of the online learning task, and is oblivious to the domain knowledge of the problem instance.

1.1 Related Work

Multi-armed bandit problem has been well studied in the literature, in particular in statistics and reinforcement learning (cf. Berry and Fristedt, 1985; Sutton and Barto, 1998). Our work follows the line of research on stochastic MAB problems, which is initiated by Lai and Robbins (1985), who show that under certain conditions on reward distributions, one can achieve a tight asymptotic regret of $\Theta(\log n)$, where n is the number of rounds played. Later, Auer et al. (2002a) demonstrate that $O(\log n)$ regret can be achieved uniformly over time rather than only asymptotically. They propose several MAB algorithms, including the UCB1 algorithm, which has been widely followed and adapted in MAB research.

For combinatorial multi-armed bandits, a few specific instances of the problem has been studied in the literature. A number of studies consider simultaneous plays of k arms among m arms (e.g. Anantharam et al., 1987; Caro and Gallien, 2007; Liu et al., 2011). Other instances include the matching bandit (Gai et al., 2010) and the online shortest path problem (Liu and Zhao, 2012).

The work closest to ours is a recent work by Gai et al. (2012), which also considers a combinatorial bandit framework with an approximation oracle. However, our work differs from theirs in several important aspects. Most importantly, their work only considers linear rewards while our CMAB framework includes a much larger class of linear and nonlinear rewards. Secondly, our regret analysis is much tighter, and as a result we significantly improve their distribution-dependent regret bound when applying our result to the linear reward case, and we are able to derive a distribution-independent regret bound close to the theoretical lower bound while they do not provide distribution-independent bounds. Moreover, we allow the approximation oracle to have a failure probability (i.e., $\beta < 1$), while they do not consider such failure probabilities.

In terms of types of feedbacks in combinatorial bandits (Audibert et al., 2011), our work belongs to the *semi-bandit* type, in which the player observes only the outcomes of played arms in one round of play. Other types include (a) *full information*, in which the player observes the outcomes of all arms, and (b) *bandit*, in which the player only observes the final reward but no outcome of any individual arm. More complicated feedback dependences are also considered by Mannor and Shamir (2011).

Bounded smoothness property in our paper is an extended form of Lipschitz condition, but our model and results differ from the Lipschitz bandit research (Kleinberg et al., 2008) in several aspects. First, Lipschitz bandit considers a continuous metric space where every point is an arm, and the Lipschitz condition is applied to two points (i.e., two arms). Under this assumption, if we know one arm pretty well, we will also know the nearby arms pretty well. In contrast, our bounded smoothness condition is applied to a vector of mean values of the base arms instead of one super arm, and by knowing one super arm well, we cannot directly know how good are the other super arms. Second, the feedback model is different: Lipschitz bandit assumes bandit feedback model while our CMAB assumes semi-bandit feedback. Third, using the Lipschitz condition, they designed a new algorithm called zooming algorithm, which maintains a confidence radius of arms, so that by knowing the center arm well, they are also pretty confident of the arms within the confidence radius of the center. In comparison, our algorithm is basically a direct extension of the classical UCB algorithm, in which the confidence radius is used to get confidence on the estimate of

every base arm. Kleinberg et al. (2008) generalize the setting of continuum bandits, which assumes the strategy set is a compact subset of $\mathbb{R}^d$ and the reward function satisfies the Lipschitz condition (see e.g. Agrawal, 1995; Kleinberg, 2004).

A different line of research considers *adversarial multi-armed bandit*, initiated by Auer et al. (2002b), in which no probabilistic assumptions are made about the rewards, and they can even be chosen by an adversary. In the context of adversarial bandits, several studies also consider combinatorial bandits (Cesa-Bianchi and Lugosi, 2009; Audibert et al., 2011; Bubeck et al., 2012). For linear rewards, Kakade et al. (2009) have shown how to convert an approximation oracle into an online algorithm with sublinear regret both in the full information setting and the bandit setting. For non-linear rewards, various online submodular optimization problems with bandit feedback are studied in the adversarial setting (Streeter and Golovin, 2008; Radlinski et al., 2008; Streeter et al., 2009; Hazan and Kale, 2009). Notice that our framework deals with stochastic instances and we can handle reward functions more general than the submodular ones.

This paper is the full version of our ICML'2013 paper (Chen et al., 2013) with the extension to include probabilistically triggered arms in the model and analysis. We made a mistake previously (Chen et al., 2013) by claiming that the online learning task for social influence maximization is an instance of the original CMAB model (Chen et al., 2013) without explicitly modeling probabilistically triggered arms. In this paper we correct this mistake by allowing probabilistically triggered arms in the CMAB model, and by significantly revising the analysis to include triggering probabilities in the analysis and the regret bounds.

Since our work of CMAB model (Chen et al., 2013), several studies are also related to combinatorial multi-armed bandits or in general combinatorial online learning. Qin et al. (2014) extend CMAB to contextual bandits and apply it to diversified online recommendations. Lin et al. (2014) address combinatorial actions with limited feedbacks. Gopalan et al. (2014) use Thompson sampling method to tackle combinatorial online learning problems. Comparing with our CMAB framework, they allow more feedback models than our semi-bandit feedback model, but they require finite number of actions and observations, their regret contains a large constant term, and it is unclear if their framework supports approximation oracles for hard combinatorial optimization problems. Kveton et al. (2014) study linear matroid bandits, which is a subclass of the linear combinatorial bandits we discussed in Section 4.2, and they provide better regret bounds than our general bounds given in Section 4.2, because their analysis utilizes the matroid combinatorial structure. In a latest paper, Kveton et al. (2015) improve the regret bounds of the linear combinatorial bandits via a more sophisticated non-uniform sufficient sampling condition than the one we used in our paper. However, it is unclear if this technique can be applied to non-linear reward functions satisfying the bounded smoothness condition (see discussions in Section 4.2 for more details).

1.2 Paper Organization

In Section 2 we formally define the CMAB framework. Section 3 provides the CUCB algorithm and the main results on its regret bounds and the proofs. Section 4 shows how to apply the CMAB framework and CUCB algorithm to the online advertising and viral

marketing applications, as well as the class of combinatorial bandits with linear reward functions. We conclude the paper in Section 5.

2. General CMAB Framework

A *combinatorial multi-armed bandit (CMAB)* problem consists of m *base arms* associated with a set of random variables $X_{i,t}$ for $1 \le i \le m$ and $t \ge 1$, with bounded support on $[0,1]$. Variable $X_{i,t}$ indicates the random outcome of the i-th base arm in its t-th trial. The set of random variables $\{X_{i,t} \mid t \ge 1\}$ associated with base arm i are independent and identically distributed according to some unknown distribution with unknown expectation μ_i. Let $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_m)$ be the vector of expectations of all base arms. Random variables of different base arms may be dependent.

The unit of play in CMAB is a *super arm*, which is a set of base arms. Let $\mathcal{S}$ denote the set of all possible super arms that can be played in a CMAB problem instance. For example, $\mathcal{S}$ could be the set of all subsets of base arms containing at most k base arms. In each round, one of the super arms $S \in \mathcal{S}$ is selected and played, and every base arm $i \in S$ are triggered and played as a result. The outcomes of base arms in S may trigger other base arms not in S to be played, and the outcomes of these arms may further trigger more arms to be played, and so on. Therefore, when super arm S is played in round t, a superset of S is triggered and played, and the final reward of this round depends on the outcomes of all triggered base arms. The feedback in the round after playing super arm S is the outcomes of the triggered (played) base arms. The random outcomes of triggered base arms in one round are independent of random outcomes in other rounds, but they may depend on one another in the same round.

For each $i \in [m]$, let p_i^S denote the probability that base arm i is triggered when super arm S is played. Once super arm S is fixed, the event of triggering of base arm i is independent of the history of previous plays of super arms. It is clear that for all $i \in S$, $p_i^S = 1$. Note that probability p_i^S may not be known to the learning algorithm, since the event of triggering base arm i may depend on the random outcomes of other base arms, the distribution of which may be unknown. Moreover, the triggering of base arms may depend on certain combinatorial structure of the problem instance, and triggering of different base arms may not be independent from one another (for an example, see the social influence maximization application in Section 4.3).

Let $\tilde{S} = \{i \in [m] \mid p_i^S > 0\}$ denote the set of possibly triggered base arms by super arm S, also referred to as the *triggering set* of S. Let $p_i \triangleq \min_{S \in \mathcal{S}, i \in \tilde{S}} p_i^S$ denote the minimum nonzero triggering probability of base arm i under all super arms. When $p_i = 1$ for all $i \in [m]$, each super arm S deterministically triggers all base arms in $\tilde{S}$, in which case we treat S and $\tilde{S}$ as the same set. Let $p^* \triangleq \min_{i \in [m]} p_i$.

In our model, it is possible that a base arm i does not belong to any super arm, and thus i can only be probabilistically triggered. In fact, our model is flexible enough to allow that *all* based arms are probabilistically triggered. To do so, we can simply add a set of dummy base arms and dummy super arms for the purpose of probabilistically triggering real base arms. In particular, for each real base arm i, we can add a dummy base arm d_i, which is a Bernoulli random variable with 1 meaning i is triggered and 0 meaning i is not triggered. Then a dummy super arm containing a subset of these Bernoulli dummy base

arms can be used to probabilistically trigger a set of real base arms. If all super arms are such dummy super arms, then all real base arms are only probabilistically triggered.

For each arm $i \in [m]$, let $T_{i,t}$ denote the number of times arm i has been successfully triggered after the first t rounds in which t super arms are played. If an arm $i \in \tilde{S} \setminus S$ is not triggered in round t when super arm S is played, then $T_{i,t} = T_{i,t-1}$. Let $R_t(S)$ be a non-negative random variable denoting the reward of round t when super arm S is played. The reward depends on the actual problem instance definition, the super arm S played, and the outcomes of all triggered arms in round t. The reward $R_t(S)$ might be as simple as a summation of the outcomes of the triggered arms in S: $R_t(S) = \sum_{i \in \tilde{S}, i \text{ is triggered}} X_{i,T_{i,t}}$, but our framework allows more sophisticated nonlinear rewards, as explained below.

In this paper, we consider CMAB problems in which the expected reward of playing any super arm S in any round t, $\mathbb{E}[R_t(S)]$, is a function of S and the expectation vector $\boldsymbol{\mu}$ of all arms. For the linear reward case as given above together with no probabilistic triggering ($S = \tilde{S}$), this is true because linear addition is commutative with the expectation operator. For non-linear reward functions not commutative with the expectation operator, it is still true if we know the type of distributions and only the expectations of arm outcomes are unknown. For example, the distribution of $X_{i,t}$'s are known to be independent 0-1 Bernoulli random variables with unknown mean μ_i.[1] Henceforth, we denote the expected reward of playing S as $r_{\boldsymbol{\mu}}(S) \triangleq \mathbb{E}[R_t(S)]$.

Definition 1 (Assumptions on expected reward function)**.** *To carry out our analysis, we make the following two mild assumptions on the expected reward $r_{\boldsymbol{\mu}}(S)$:*

- **Monotonicity**. *The expected reward of playing any super arm $S \in \mathcal{S}$ is monotonically nondecreasing with respect to the expectation vector, i.e., if for all $i \in [m]$, $\mu_i \le \mu_i'$, we have $r_{\boldsymbol{\mu}}(S) \le r_{\boldsymbol{\mu}'}(S)$ for all $S \in \mathcal{S}$.*
- **Bounded smoothness.** *There exists a continuous, strictly increasing (and thus invertible) function $f(\cdot)$ with $f(0) = 0$, called* bounded smoothness function*, such that for any two expectation vectors $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$ and for any $\Lambda > 0$, we have $|r_{\boldsymbol{\mu}}(S) - r_{\boldsymbol{\mu}'}(S)| \le f(\Lambda)$ if $\max_{i \in \tilde{S}} |\mu_i - \mu_i'| \le \Lambda$, for all $S \in \mathcal{S}$.*

Both assumptions are natural. In particular, they hold true for all the applications we considered. We remark that bounded smoothness is an extended form of Lipschitz condition in that we use a general function f instead of linear or power-law functions typically used in Lipschitz condition definition, and we use infinity norm instead of typically used L_2 norm.

Definition 2 (CMAB algorithm)**.** *A CMAB algorithm A is one that selects the super arm of round t to play based on the outcomes of revealed arms of previous rounds, without knowing the expectation vector $\boldsymbol{\mu}$. Let $S_t^A \in \mathcal{S}$ be the super arm selected by A in round t. Note that S_t^A is a random super arm that depends on the outcomes of arms in previous rounds and potential randomness in the algorithm A itself. The objective of algorithm A is to maximize the expected reward of all rounds up to a round n, that is, $\mathbb{E}_{S,R}[\sum_{t=1}^n R_t(S_t^A)] = \mathbb{E}_S[\sum_{t=1}^n r_{\boldsymbol{\mu}}(S_t^A)]$, where $\mathbb{E}_{S,R}$ denotes taking expectation among all*

1. It is also possible that the Bernoulli random variables are not independent. For example, the joint distribution is determined by sampling a random value $\rho \in [0,1]$ uniformly at random, and then each base arm i takes value 1 if any only if $\rho \le \mu_i$.

random events generating the super arms S_t^A's and generating rewards $R_t(S_t^A)$'s, and $\mathbb{E}_S$ denotes taking expectation only among all random events generating the super arms S_t^A's.

We do not assume that the learning algorithm has the direct knowledge about the problem instance, e.g. how super arms are formed from the base arms, how base arms outside of a super arm are triggered, and how reward is defined. Instead, the algorithm has access to a computation oracle that takes the expectation vector $\boldsymbol{\mu}$ as the input, and together with the knowledge of the problem instance, computes the optimal or near-optimal super arm S. Let $\text{opt}_{\boldsymbol{\mu}} = \max_{S\in\mathcal{S}} r_{\boldsymbol{\mu}}(S)$ and $S^*_{\boldsymbol{\mu}} = \text{argmax}_{S\in\mathcal{S}} r_{\boldsymbol{\mu}}(S)$. We consider the case that exact computation of $S^*_{\boldsymbol{\mu}}$ may be computationally hard, and the algorithm may be randomized with a small failure probability. Thus, we resolve to the following (α,β)-*approximation oracle*:

Definition 3 ((α,β)-Approximation oracle)**.** *For some $\alpha,\beta \le 1$, (α,β)-approximation oracle is an oracle that takes an expectation vector $\boldsymbol{\mu}$ as input, and outputs a super arm $S \in \mathcal{S}$, such that $\Pr[r_{\boldsymbol{\mu}}(S) \ge \alpha \cdot \text{opt}_{\boldsymbol{\mu}}] \ge \beta$. Here β is the success probability of the oracle.*

Many computationally hard problems do admit efficient approximation oracles (Vazirani, 2004). With an (α,β)-approximation oracle, it is no longer fair to compare the performance of a CMAB algorithm against the optimal reward $\text{opt}_{\boldsymbol{\mu}}$ as the regret of the algorithm. Instead, we compare against the $\alpha\cdot\beta$ fraction of the optimal reward, because only a β fraction of oracle computations are successful, and when successful the reward is only an α-approximation of the optimal value.

Definition 4 ((α,β)-approximation regret)**.** *(α,β)-approximation regret of a CMAB algorithm A after n rounds of play using an (α,β)-approximation oracle under the expectation vector $\boldsymbol{\mu}$ is defined as*

$$Reg^A_{\boldsymbol{\mu},\alpha,\beta}(n) = n\cdot\alpha\cdot\beta\cdot\text{opt}_{\boldsymbol{\mu}} - \mathbb{E}_S\left[\sum_{t=1}^{n} r_{\boldsymbol{\mu}}(S_t^A)\right]. \tag{1}$$

Note that the classical MAB problem is a special case of our general CMAB problem, in which (a) the constraint $\mathcal{S} = \{\{i\} \mid i \in [m]\}$ so that each super arm is just a base arm; (b) $S = \tilde{S}$ for all super arm S, that is, playing of a base arm does not trigger any other arms; (c) the reward of a super arm $S = \{i\}$ in its t's trial is its outcome $X_{i,t}$; (d) the monotonicity and bounded smoothness hold trivially with function $f(\cdot)$ being the identity function; and (e) the (α,β)-approximation oracle is simply the argmax function among all expectation vectors, with $\alpha = \beta = 1$.

3. CUCB Algorithm for CMAB

We present our CUCB algorithm in Algorithm 1. We maintain an empirical mean $\hat{\mu}_i$ for each arm i. More precisely, if arm i has been played s times by the end of round n, then the value of $\hat{\mu}_i$ at the end of round n is $(\sum_{j=1}^{s} X_{i,j})/s$. The actual expectation vector $\bar{\boldsymbol{\mu}}$ given to the oracle contains an adjustment term $\sqrt{\frac{3\ln t}{2T_i}}$ for each $\hat{\mu}_i$, which depends on the round number t and the number of times arm i has been played (stored in variable T_i).

1: For each arm i, maintain: (1) variable T_i as the total number of times arm i is played so far, initially 0; (2) variable $\hat{\mu}_i$ as the mean of all outcomes $X_{i,*}$'s of arm i observed so far, initially 1.
2: $t \leftarrow 0$.
3: **while true do**
4: $\quad t \leftarrow t+1$.
5: $\quad$ For each arm i, set $\bar{\mu}_i = \min\left\{\hat{\mu}_i + \sqrt{\frac{3\ln t}{2T_i}}, 1\right\}$.
6: $\quad S = \text{Oracle}(\bar{\mu}_1, \bar{\mu}_2, \ldots, \bar{\mu}_m)$.
7: $\quad$ Play S, observe outcomes of played base arms i, and update all T_i's and $\hat{\mu}_i$'s.
8: **end while**

Algorithm 1: CUCB with computation oracle.

Then we simply play the super arm returned by the oracle and update variables T_i's and $\hat{\mu}_i$'s accordingly. Note that in our model all arms have bounded support on $[0,1]$, but with the adjustment the upper confidence bound $\hat{\mu}_i + \sqrt{\frac{3\ln t}{2T_i}}$ may exceed 1, in which case we simply trim it down to 1 and assign it to $\bar{\mu}_i$ (line 5).

Our algorithm does not have an initialization phase where all base arms are played at least once. This is to accommodate the case where some base arms may only be probabilistically triggered and there is no super arm that can trigger them deterministically. Instead, we simply initialize the counter T_i to 0 and $\hat{\mu}_i$ to 1 for every base arm i. Thus initially $\bar{\mu}_i = 1$ for all i, and the oracle will select a super arm given an all-one vector input. Intuitively, any base arm i that has not been played will have its $\bar{\mu}_i = 1$, which should let the oracle be biased toward playing a super arm that (likely) triggers i. It may be possible that a base arm i is never played, and this only means that i is not important for the optimization task and the oracle decides not to play it (deterministically or probabilistically). Our analysis works correctly without the initialization phase.

We now provide necessary definitions for the main theorems.

Definition 5 (Bad super arm)**.** *A super arm S is* bad *if $r_{\boldsymbol{\mu}}(S) < \alpha\cdot\text{opt}_{\boldsymbol{\mu}}$. The set of bad super arms is defined as $\mathcal{S}_{\text{B}} \triangleq \{S \mid r_{\boldsymbol{\mu}}(S) < \alpha\cdot\text{opt}_{\boldsymbol{\mu}}\}$. For a given base arm $i \in [m]$, let $\mathcal{S}_{i,\text{B}} = \{S \in \mathcal{S}_{\text{B}} \mid i \in \tilde{S}\}$ be the set of bad super arms whose triggering sets contain i. We sort all bad super arms in $\mathcal{S}_{i,\text{B}}$ as $S^1_{i,\text{B}}, S^2_{i,\text{B}}, \ldots, S^{K_i}_{i,\text{B}}$, in increasing order of their expected rewards, where $K_i = |\mathcal{S}_{i,\text{B}}|$. Note that when $K_i = 0$, there is no bad super arm that can trigger base arm i.*

Definition 6 (Δ of bad super arms)**.** *For a bad super arm $S \in \mathcal{S}_{\text{B}}$, we define $\Delta_S \triangleq \alpha\cdot\text{opt}_{\boldsymbol{\mu}} - r_{\boldsymbol{\mu}}(S)$. For a given base arm $i \in [m]$ with $K_i > 0$ and index $j \in [K_i]$, we define*

$$\Delta^{i,j} \triangleq \Delta_{S^j_{i,\text{B}}}.$$

We have special notations for the minimum and the maximum $\Delta^{i,j}$ for a fixed i with $K_i > 0$:

$$\Delta^i_{\max} \triangleq \Delta^{i,1},$$
$$\Delta^i_{\min} \triangleq \Delta^{i,K_i}.$$

Furthermore, define $\Delta_{\max} \triangleq \max_{i\in[m],K_i>0}\Delta^i_{\max}$, $\Delta_{\min} \triangleq \min_{i\in[m],K_i>0}\Delta^i_{\min}$.

Our main theorem below provides the distribution-dependent regret bound of the CUCB algorithm using the Δ notations. We use $\mathbb{I}\{\cdot\}$ to denote the indicator function, and $\mathbb{I}\{\mathcal{E}\} = 1$ if $\mathcal{E}$ is true, and 0 if $\mathcal{E}$ is false.

Theorem 1. *The (α, β)-approximation regret of the CUCB algorithm in n rounds using an (α, β)-approximation oracle is at most*

$$\sum_{i\in[m],K_i>0}\left(\ell_n(\Delta^i_{\min}, p_i)\Delta^i_{\min} + \int_{\Delta^i_{\min}}^{\Delta^i_{\max}} \ell_n(x, p_i)\mathrm{d}x\right) + \left(\frac{(2+\mathbb{I}\{p^*<1\})\pi^2}{6}+1\right)\cdot m\cdot\Delta_{\max}, \tag{2}$$

where $p^* = \min_{i\in[m]} p_i$, and

$$\ell_n(\Delta, p) = \begin{cases} \max\left(\frac{12\cdot\ln n}{(f^{-1}(\Delta))^2\cdot p}, \frac{24\cdot\ln n}{p}\right), & \text{if } 0<p<1,\\ \frac{6\ln n}{(f^{-1}(\Delta))^2}, & \text{if } p=1.\end{cases}$$

and $f(\cdot)$ is the bounded smoothness function.

Note that when $p_i = 1$ for all $i \in [m]$, each super arm S deterministically triggers base arms in S and no probabilistic triggering of other arms. In this case, the above theorem falls back to Theorem 1 of the original CMAB paper (Chen et al., 2013). When $p_i < 1$ for some $i \in [m]$, the regret bound is slightly more complicated, in particular, it has an extra factor of $1/p_i$ appearing in the leading $\ln n$ term.

In Theorem 1, when $\Delta^i_{\min}$ is extremely small, the regret would be approaching infinity. Below we prove a distribution-independent regret for arbitrary distributions with support in $[0, 1]$ on all arms, for a large class of problem instances with a polynomial bounded smoothness function $f(x) = \gamma x^\omega$ for $\gamma > 0$ and $0 < \omega \le 1$. The rough idea of the proof is, if $\Delta^i_{\min} \le 1/\sqrt{n}$, it can only contribute $\sqrt{n}$ regret at time horizon n. The proof of the following theorem relies on the tight regret bound of Theorem 1 on the leading $\ln n$ term.

Theorem 2. *Consider a CMAB problem with an (α, β)-approximation oracle. Let $p^* = \min_{i\in[m]} p_i$. If the bounded smoothness function $f(x) = \gamma\cdot x^\omega$ for some $\gamma > 0$ and $\omega \in (0, 1]$, the regret of CUCB is at most:*

$$\begin{cases} \frac{2\gamma}{2-\omega}\cdot(6m\ln n)^{\omega/2}\cdot n^{1-\omega/2} + \left(\frac{\pi^2}{3}+1\right)\cdot m\cdot\Delta_{\max}, & \text{if } p^*=1,\\ \frac{2\gamma}{2-\omega}\cdot\left(\frac{12m\ln n}{p^*}\right)^{\omega/2}\cdot n^{1-\omega/2} + \left(\frac{\pi^2}{2}+1\right)\cdot m\cdot\Delta_{\max} + \sum_{i\in[m]}\frac{24\ln n}{p_i}\cdot\Delta_{\max}, & \text{if } 0<p^*<1.\end{cases}$$

Note that for all applications discussed in Section 4, we have $\omega = 1$. For the classical MAB setting with $\omega = 1$ and $p^* = 1$, we obtain a distribution-independent bound of $O(\sqrt{mn\ln n})$, which matches (up to a logarithmic factor) the original UCB1 algorithm (Audibert et al., 2009). In the linear combinatorial bandit setting, i.e., semi-bandit with L_∞ assumption (Audibert et al., 2011), our regret is $O(\sqrt{m^3 n\log n})$, which is a factor $\sqrt{m}$ off the optimal bound in the adversarial setting, a more general setting than the stochastic setting (see the discussion in the end of Section 4.2 for a reason of this gap).

3.1 Proof of the Theorems

Below we prove Theorem 1 and Theorem 2.

3.1.1 Proof of Theorem 1

Before getting to the proof of our theorem, we need more definitions and lemmas. First, we have a convenient notation for the case when the oracle outputs non-α-approximation answers.

Definition 7 (Non-α-approximation output)**.** *In the t-th round, let F_t be the event that the oracle fails to produce an α-approximate answer with respect to its input $\bar{\boldsymbol{\mu}} = (\bar\mu_1, \bar\mu_2, \ldots, \bar\mu_m)$. We have $\Pr[F_t] = \mathbb{E}[\mathbb{I}\{F_t\}] \le 1-\beta$.*

Since the value of many variables are changing in different rounds, we also define notations for their value in round t. All of them are random variables.

Definition 8 (Variables in round t)**.** *For variable T_i, let $T_{i,t}$ be the value of T_i at the end of round t, that is, $T_{i,t}$ is the number of times arm i is played in the first t rounds. For variable $\hat\mu_i$, let $\hat\mu_{i,s}$ be the value of $\hat\mu_i$ after arm i is played s times, that is, $\hat\mu_{i,s} = (\sum_{j=1}^s X_{i,j})/s$, where $X_{i,j}$ is the outcome of base arm i in its j-th trial, as defined at the beginning of Section 2. Then, the value of variable $\hat\mu_i$ at the end of round t is $\hat\mu_{i,T_{i,t}}$. For variable $\bar\mu_i$, let $\bar\mu_{i,t}$ be the value of $\bar\mu_i$ at the end of round t.*

Next, we introduce an important parameter in our proof called sampling threshold.

Definition 9 (Sampling threshold)**.** *For a probability value $p \in (0, 1]$ and reward difference value $\Delta \in \mathbb{R}^+$, the value $\ell_n(\Delta, p)$ defined below is called the* sampling threshold *for round n:*

$$\ell_n(\Delta, p) = \begin{cases} \max\left(\frac{12\cdot\ln n}{(f^{-1}(\Delta))^2\cdot p}, \frac{24\cdot\ln n}{p}\right), & \text{if } 0<p<1,\\ \frac{6\ln n}{(f^{-1}(\Delta))^2}, & \text{if } p=1.\end{cases}$$

Informally, base arm $i \in [m]$ at round n is considered as sufficiently sampled if the number of times i has been played by round n, $T_{i,n}$, is above its sampling threshold $\ell_n(\Delta^i_{\min}, p_i)$. When all base arms are sufficiently sampled, with high probability we would obtain accurate estimates of their sample means and would be able to distinguish the α-approximate super arms from bad super arms.

We utilize the following well known tail bounds in our analysis.

Fact 1 (Hoeffding's Inequality (Hoeffding, 1963))**.** *Let $X_1, \cdots, X_n$ be independent and identically distributed random variables with common support $[0, 1]$ and mean μ. Let $Y = X_1 + \cdots + X_n$. Then for all $\delta \ge 0$,*

$$\Pr\{|Y - n\mu| \ge \delta\} \le 2e^{-2\delta^2/n}.$$

Fact 2 (Multiplicative Chernoff Bound (Mitzenmacher and Upfal, 2005)[2])**.** *Let $X_1, \cdots, X_n$ be Bernoulli random variables taking values from $\{0, 1\}$, and $\mathbb{E}[X_t|X_1, \cdots, X_{t-1}] \ge \mu$ for*

2. The result in the book by Mitzenmacher and Upfal (2005) (Theorem 4.5 together with Exercise 4.7) only covers the case where random variables X_i's are independent. However the result can be easily generalized to our case with an almost identical proof. The only main change is to replace $\mathbb{E}\left[e^{t(\sum_{j=1}^{i-1} X_j + X_i)}\right] = \mathbb{E}\left[e^{t\sum_{j=1}^{i-1} X_j}\right]\mathbb{E}\left[e^{tX_i}\right]$ with $\mathbb{E}\left[e^{t(\sum_{j=1}^{i-1} X_j + X_i)}\right] = \mathbb{E}\left[e^{t\sum_{j=1}^{i-1} X_j}\mathbb{E}\left[e^{tX_i} \mid X_1, \ldots, X_{i-1}\right]\right]$.

every $t \le n$. Let $Y = X_1 + \cdots + X_n$. Then for all $0 < \delta < 1$,

$$\Pr\{Y \le (1-\delta)n\mu\} \le e^{-\frac{\delta^2 n\mu}{2}}.$$

Using the above tail bounds, we can prove that with high probability, the empirical mean of a set of independently sampled variables is close to the actual mean. Below we give a definition on the standard difference between the empirical mean and the actual expectation.

Definition 10 (Standard difference). *For the random variable $T_{i,t-1}$, standard difference is defined as a random variable $\Lambda_{i,t} = \min\{\sqrt{\frac{3\ln t}{2T_{i,t-1}}}, 1\}$. The maximum standard difference is defined as a random variable $\Lambda_t = \max\{\Lambda_{i,t} \mid i \in \tilde{S}_t\}$ (be reminded that it is $\tilde{S}_t$, not S_t). The universal difference bound is defined as $\Lambda^{i,l} = \frac{f^{-1}(\Delta^{i,l})}{2}$, which is* not *a random variable.*

If in the round t, the difference between the empirical mean and the actual expectation is below the standard difference, we call the process a "nice process". See the formal definition below.

Definition 11 (Nice run). *The run of Algorithm 1 is* nice *at time t (denoted as the indicator $\mathcal{N}_t$) if:*

$$\forall i \in [m],\ |\hat{\mu}_{i,T_{i,t-1}} - \mu_i| \le \Lambda_{i,t}. \tag{3}$$

Lemma 1. *The probability that the run of Algorithm 1 is* nice *at time t is at least $1 - \frac{2m}{t^2}$.*

Proof. If $T_{i,t-1} = 0$, this is trivially true. If $T_{i,t-1} > 0$, by the Hoeffding's inequality in Fact 1, for any $i \in [m]$,

$$\begin{aligned}
&\Pr\left\{|\ \hat{\mu}_{i,T_{i,t-1}} - \mu_i\ | \ge \Lambda_{i,t}\right\} = \sum_{s=1}^{t-1} \Pr\left\{|\ \hat{\mu}_{i,s} - \mu_i\ | \ge \Lambda_{i,t}, T_{i,t-1} = s\right\} \\
\le &\sum_{s=1}^{t-1} \Pr\left\{|\ \hat{\mu}_{i,s} - \mu_i\ | \ge \sqrt{\frac{3\ln t}{2s}}\right\} \le t \cdot 2e^{-3\ln t} = \frac{2}{t^2}.
\end{aligned} \tag{4}$$

The lemma follows by taking union bound on i. □

Lemma 1 tells us that if at time t, $T_{i,t-1}$ is large, then we can get a good estimation of μ_i. Intuitively, if we estimate all μ_i's pretty well, it is unlikely that we will choose a bad super arm using the approximation oracle. On the other hand, in the case that for some i $T_{i,t-1}$ is small, although we may not have a good estimate of μ_i, it indicates that arm i has not been played for many times, which gives us an upper bound on the number of times that the algorithm plays a bad super arm containing arm i. Based on this idea, it is crucial to find a sampling threshold, which separates these two cases.

Now we need to define the way that we count the sampling times of each arm i.

Definition 12 (Counter for arm i). *We maintain a counter N_i for each arm i. Let $N_{i,t}$ be the value of N_i at the end of round t and $N_{i,0} = 0$. $\{N_i\}$ is updated in the following way.*

For a round $t > 0$, let S_t be the super arm selected in round t by the oracle (line 6 of Algorithm 1). Round t is bad *if the oracle selects a* bad *super arm $S_t \in \mathcal{S}_{\mathrm{B}}$. If round t is bad, let $i = \mathrm{argmin}_{j \in \tilde{S}_t} N_{j,t-1} \cdot p_j$. If the above i is not unique, we pick an arbitrary one. Then we increment the counter N_i, i.e., $N_{i,t} = N_{i,t-1} + 1$ while not changing other counters N_j with $j \ne i$. If round t is not bad, i.e., $S_t \notin \mathcal{S}_{\mathrm{B}}$, no counter N_i is incremented.*

Note that the counter N_i is for the purpose of analysis, and its maintenance is not part of the algorithm. Intuitively, for each round t where a bad super arm S_t is played, we increment exactly one counter N_i, where i is selected among all possibly triggered base arms $\tilde{S}_t$ such that the current value of $N_i \cdot p_i$ is the lowest. In the special case when $p_i = 1$ for some $i \in [m]$, we know that $i \notin \tilde{S} \setminus S$ for any super arm S. Therefore, whenever arm i is selected to increment its counter N_i in a round t, i must have been played in round t, and thus we have $T_{i,t} \ge N_{i,t}$ for any $i \in [m]$ with $p_i = 1$ and all time t. However, this may not holds for $i \in [m]$ with $p_i < 1$, that is, it is possible that in a round t a base arm i is not triggered but its counter N_i is incremented.

In every bad round, exactly one counter in $\{N_i\}$ is incremented, so the total number of bad rounds in the first n rounds is exactly $\sum_i N_{i,n}$. Below we give the definition of refined counters.

Definition 13 (Refined counters). *Each time N_i gets updated, one of the bad super arms that could trigger i is played. We further separate N_i into a set of counters as follows:*

$$\forall l \in [K_i],\ N^l_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{S_t = S^l_{i,\mathrm{B}}, N_{i,t} > N_{i,t-1}\}.$$

That is, each time N_i is updated, we also record which bad super arm is played.

With these counters in hands, we shall define the two stages "sufficiently sampled" and "under-sampled" using the sampling threshold, which further split the counter $N^l_{i,n}$ into two counters.

Definition 14 (Sufficiently sampled and under-sampled). *Consider time horizon n and current time $t \le n$. For the refined counter $N^l_{i,n}$'s, we separate them into sufficiently sampled part and under-sampled part, as defined below. When counter $N^l_{i,t}$ is incremented at time t, i.e, $S_t = S^l_{i,\mathrm{B}}$, we inspect the counter $N_{i,t-1}$. If $N_{i,t-1} > \ell_n(\Delta^{i,l}, p_i)$, we say that the bad super arm $S^l_{i,\mathrm{B}}$ is* sufficiently sampled *(with respect to base arm i); otherwise, it is* under-sampled *(with respect to base arm i). Thus counter $N^l_{i,n}$ is separated into the following sufficiently sampled part and under-sampled part:*

$$\begin{aligned}
N^{l,suf}_{i,n} &= \sum_{t=1}^{n} \mathbb{I}\{S_t = S^l_{i,\mathrm{B}}, N_{i,t} > N_{i,t-1}, N_{i,t-1} > \ell_n(\Delta^{i,l}, p_i)\}, \\
N^{l,und}_{i,n} &= \sum_{t=1}^{n} \mathbb{I}\{S_t = S^l_{i,\mathrm{B}}, N_{i,t} > N_{i,t-1}, N_{i,t-1} \le \ell_n(\Delta^{i,l}, p_i)\}.
\end{aligned}$$

Following the definition, we have $N_{i,n}^{l,und} \le \ell_n(\Delta^{i,l}, p_i)$, and $N_{i,n} = \sum_{l\in[K_i]}(N_{i,n}^{l,suf} + N_{i,n}^{l,und})$. Using this notation, the total reward at time horizon n is at least

$$n \cdot \alpha \cdot \mathrm{opt}_{\boldsymbol{\mu}} - \sum_{i\in[m],K_i>0} \sum_{l\in[K_i]} (N_{i,n}^{l,suf} + N_{i,n}^{l,und}) \cdot \Delta^{i,l}. \tag{5}$$

To get an upper bound on the regret, we want to upper bound $N_{i,n}^{l,suf}$ and $N_{i,n}^{l,und}$ separately. Before doing that, we prove an important connection as follows.

Lemma 2 (Connection between $N_{i,t-1}$ and $T_{i,t-1}$)**.** *Let n be the time horizon. For every round t with $0 < t \le n$, every base arm $i \in [m]$, every $\Delta > 0$, and every integer $k > \ell_n(\Delta, p_i)$, we have,*

$$\Pr\left\{N_{i,t-1} = k, T_{i,t-1} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\} \le \frac{1}{t^3}. \tag{6}$$

Moreover, if p_i=1, we have

$$\Pr\left\{N_{i,t-1} = k, T_{i,t-1} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\} = 0.$$

Proof. Fix a base arm i. The case of $p_i = 1$ is trivial since in this case $T_{i,t-1} \ge N_{i,t-1}$ and $n \ge t$. Now we only consider the case of $0 < p_i < 1$.

In a run of CUCB algorithm (Algorithm 1), let $t^{(j)}$ be the round number at which counter N_i is incremented for the j-th time. Suppose that in round $t^{(j)}$, super arm $S^{(j)}$ is played. Note that both $t^{(j)}$ and $S^{(j)}$ are random, depending on the randomness of the outcomes of base arms and the triggering of base arms from super arms in all historical rounds.

Let $X^{(j)}$ be the Bernoulli random variable indicating whether arm i is triggered by the play of super arm $S^{(j)}$ in round $t^{(j)}$. If in a run of the CUCB algorithm counter N_i is only incremented a finite number of times, let $N_{i,\infty}$ denote the final value of the counter N_i in this run. In this case, we simply define $X^{(j)} = 1$ for all $j > N_{i,\infty}$. For convenience, when $j > N_{i,\infty}$, we denote the corresponding super arm $S^{(j)} = \perp$. Thus, for any $\ell \ge 1$, $\sum_{j=1}^{\ell} X^{(j)}$ is well defined. For all $0 < t \le n$, since $T_{i,t-1}$ is the number of times i is triggered by the end of round $t-1$, we have

$$\sum_{j=1}^{N_{i,t-1}} X^{(j)} \le T_{i,t-1}. \tag{7}$$

We now show that for any $j \ge 1$, $\mathbb{E}[X^{(j)} \mid X^{(1)}, \ldots, X^{(j-1)}] \ge p_i$. Fixing a super arm $A \in \mathcal{S}$, if super arm $S^{(j)}$ played in round $t^{(j)}$ is A, then conditioned on the event $S^{(j)} = A$, in this round whether arm i is triggered or not only depends on the randomness of triggering base arms after playing A, and is independent of randomness in previous rounds. In other words, we have

$$\Pr\left\{X^{(j)} = 1 \mid S^{(j)} = A, X^{(1)}, \ldots, X^{(j-1)}\right\} = \Pr\left\{X^{(j)} = 1 \mid S^{(j)} = A\right\} = p_i^A \ge p_i.$$

By the law of total probability, we have

$$\begin{aligned}
&\mathbb{E}[X^{(j)} \mid X^{(1)}, \ldots, X^{(j-1)}] \\
&= \Pr\left\{X^{(j)} = 1 \mid X^{(1)}, \ldots, X^{(j-1)}\right\} \\
&= \sum_{A\in\mathcal{S}} \Pr\left\{S^{(j)} = A\right\} \cdot \Pr\left\{X^{(j)} = 1 \mid S^{(j)} = A, X^{(1)}, \ldots, X^{(j-1)}\right\} \\
&\quad + \Pr\left\{S^{(j)} = \perp\right\} \cdot \Pr\left\{X^{(j)} = 1 \mid S^{(j)} = \perp, X^{(1)}, \ldots, X^{(j-1)}\right\} \\
&\ge p_i \sum_{A\in\mathcal{S}} \Pr\left\{S^{(j)} = A\right\} + \Pr\left\{S^{(j)} = \perp\right\} \cdot 1 \qquad (8) \\
&\ge p_i,
\end{aligned}$$

where the first part of the Inequality Eq. (8) comes from Eq. (3.1.1), and the second part comes from our definition that when $S^{(j)} = \perp$, it means that the counter N_i stops before reaching j and $X^{(j)} = 1$ in this case.

With the result that for any $j \ge 1$, $\mathbb{E}[X^{(j)} \mid X^{(1)}, \ldots, X^{(j-1)}] \ge p_i$, we apply the multiplicative Chernoff bound (Fact 2) to obtain that for any $\ell \ge 1$, $0 < \delta < 1$,

$$\Pr\left\{\sum_{j=1}^{\ell} X^{(j)} \le \ell \cdot p_i(1-\delta)\right\} \le e^{-\delta^2 \ell p_i/2}. \tag{9}$$

We are now ready to carry out the following derivation for any $0 < t \le n$, $i \in [m]$, $\Delta > 0$, and integer $k > \ell_n(\Delta, p_i)$:

$$\begin{aligned}
&\Pr\left\{N_{i,t-1} = k, T_{i,t-1} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\} \\
&\le \Pr\left\{N_{i,t-1} = k, \sum_{j=1}^{N_{i,t-1}} X^{(j)} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\} && \{\text{by Eq. (7)}\} \\
&\le \Pr\left\{\sum_{j=1}^{k} X^{(j)} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\} \\
&\le \Pr\left\{\sum_{j=1}^{\lceil \ell_n(\Delta,p_i)\rceil} X^{(j)} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\} \\
&\le \Pr\left\{\sum_{j=1}^{\lceil \ell_t(\Delta,p_i)\rceil} X^{(j)} \le \frac{6\cdot\ln t}{f^{-1}(\Delta)^2}\right\}. && \{n \ge t \Rightarrow \ell_n(\Delta,p_i) \ge \ell_t(\Delta,p_i)\} \qquad (10)
\end{aligned}$$

If $f^{-1}(\Delta)^2 \le \frac{1}{2}$, let $\delta = \frac{1}{2}$, we know $\ell_t(\Delta, p_i) = \frac{12 \cdot \ln t}{f^{-1}(\Delta)^2 \cdot p_i}$, so

$$\begin{aligned}
(10) &= \Pr\left\{\sum_{j=1}^{\lceil \ell_t(\Delta,p_i)\rceil} X^{(j)} \le \ell_t(\Delta,p_i)\cdot p_i \cdot \frac{1}{2}\right\} \\
&\le \Pr\left\{\sum_{j=1}^{\lceil \ell_t(\Delta,p_i)\rceil} X^{(j)} \le \lceil \ell_t(\Delta,p_i)\rceil \cdot p_i \cdot \frac{1}{2}\right\} \\
&\le e^{-\lceil \ell_t(\Delta,p_i)\rceil p_i/8} && \{\text{by Eq. (9)}\} \\
&\le e^{-\ell_t(\Delta,p_i)p_i/8} = e^{-\frac{3\ln t}{2f^{-1}(\Delta)^2}} \le e^{-3\ln t} = \frac{1}{t^3}.
\end{aligned}$$

If $f^{-1}(\Delta)^2 > \frac{1}{2}$, we know $\ell_t(\Delta, p_i) = \frac{24 \cdot \ln t}{p_i}$. Now let $\delta = 1 - \frac{1}{4f^{-1}(\Delta)^2} \ge \frac{1}{2}$, which means $1 - \delta = \frac{1}{4f^{-1}(\Delta)^2}$. So we have,

$$\begin{aligned}
(10) &= \Pr\left\{\sum_{j=1}^{\lceil \ell_t(\Delta,p_i)\rceil} X^{(j)} \le \frac{24\ln t}{4f^{-1}(\Delta)^2\cdot p_i}\cdot p_i\right\} \\
&= \Pr\left\{\sum_{j=1}^{\lceil \ell_t(\Delta,p_i)\rceil} X^{(j)} \le \ell_t(\Delta,p_i)\cdot p_i \cdot (1-\delta)\right\} \\
&\le e^{-\ell_t(\Delta,p_i)p_i/8} && \{\text{by Eq. (9)}\} \\
&= e^{-3\ln t} = \frac{1}{t^3}.
\end{aligned}$$

Therefore, Inequality (6) holds. □

Recall that a nice run at time t (Definition 11, denoted as $\mathcal{N}_t$) means that the difference between the empirical mean and the actual mean is bounded by the standard difference $\Lambda_{i,t}$ for every arm $i \in [m]$ ($\forall i \in [m]$, $|\hat{\mu}_{i,T_{i,t-1}} - \mu_i| \le \Lambda_{i,t}$). By Lemma 1, we know that with probability $1 - \frac{2m}{t^2}$, $\mathcal{N}_t$ holds. According to line 5 of Algorithm 1, we have $\bar{\mu}_{i,t} = \min\{\hat{\mu}_{i,T_{i,t-1}} + \Lambda_{i,t}, 1\}$. Thus, we have

$$\mathcal{N}_t \Rightarrow \forall i \in [m],\ \bar{\mu}_{i,t} - \mu_i \ge 0, \tag{11}$$

$$\mathcal{N}_t \Rightarrow \forall i \in \tilde{S}_t,\ \bar{\mu}_{i,t} - \mu_i \le 2\Lambda_t. \tag{12}$$

Meanwhile, by Definition 10, we know that for any $i \in [m]$, $l \in [K_i]$ and any time t:

$$\left\{S_t = S^l_{i,\mathrm{B}}, \forall s \in \tilde{S}_t, T_{s,t-1} > \frac{6\ln t}{f^{-1}(\Delta^{i,l})^2}\right\} \Rightarrow \Lambda^{i,l} > \Lambda_t. \tag{13}$$

With the previous observations, we have the following lemma. Informally, it says that in a nice run in round t, it is impossible that the algorithm would select a bad super arm S_t using the oracle, which outputs a correct α-approximation answer, while every arm in $\tilde{S}_t$ has been tested for enough times.

Lemma 3 (Impossible case)**.** *Let F_t be the indicator defined in Definition 7. For any $i \in [m]$, $l \in [K_i]$ and any time t, the event $\left\{\mathcal{N}_t, \neg F_t, S_t = S^l_{i,\mathrm{B}}, \forall s \in \tilde{S}_t, T_{s,t-1} > \frac{6\ln t}{f^{-1}(\Delta^{i,l})^2}\right\}$ is empty.*

Proof. Indeed, if all the conditions hold, we have:

$$\begin{aligned}
r_{\boldsymbol{\mu}}(S_t) + f(2\Lambda^{i,l}) &> r_{\boldsymbol{\mu}}(S_t) + f(2\Lambda_t) && \{\text{ strict monotonicity of } f(\cdot) \text{ and Eq.(13)}\} \\
&\ge r_{\bar{\boldsymbol{\mu}}_t}(S_t) && \{\text{bounded smoothness property and Eq.(12)}\} \\
&\ge \alpha \cdot \mathrm{opt}_{\bar{\boldsymbol{\mu}}_t} && \{\neg F_t \Rightarrow S_t \text{ is an } \alpha \text{ approximation w.r.t } \bar{\boldsymbol{\mu}}_t\ \} \\
&\ge \alpha \cdot r_{\bar{\boldsymbol{\mu}}_t}(S^*_{\boldsymbol{\mu}}) && \{\text{definition of } \mathrm{opt}_{\bar{\boldsymbol{\mu}}_t}\} \\
&\ge \alpha \cdot r_{\boldsymbol{\mu}}(S^*_{\boldsymbol{\mu}}) = \alpha \cdot \mathrm{opt}_{\boldsymbol{\mu}}. && \{\text{ monotonicity of } r_{\boldsymbol{\mu}}(S) \text{ and Eq.(11)}\}
\end{aligned}$$

So we have

$$r_{\boldsymbol{\mu}}(S^l_{i,\mathrm{B}}) + f(2\Lambda^{i,l}) > \alpha \cdot \mathrm{opt}_{\boldsymbol{\mu}}. \tag{14}$$

However, by Definition 10, $f(2\Lambda^{i,l}) = f(f^{-1}(\Delta^{i,l})) = \Delta^{i,l}$. Thus, Inequality (14) contradicts the definition of $\Delta^{i,l}$ in Definition 6. □

Now we are ready to prove the bound on sufficiently sampled part. Recall that $p^* = \min_{i\in[m]} p_i$.

Lemma 4. *[Bound on sufficiently sampled part] For any time horizon $n > m$,*

$$\mathbb{E}\left[\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]} N^{l,suf}_{i,n}\right] \le (1-\beta)n + \frac{(2+\mathbb{I}\{p^*<1\})m\pi^2}{6}. \tag{15}$$

Proof. From Definition 14 on $N^{l,suf}_{i,n}$, we have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]} N^{l,suf}_{i,n}\right] \\
&= \mathbb{E}\left[\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}\sum_{t=1}^{n} \mathbb{I}\left\{S_t = S^l_{i,\mathrm{B}}, N_{i,t} > N_{i,t-1}, N_{i,t-1} > \ell_n\left(\Delta^{i,l}, p_i\right)\right\}\right] \\
&\le \sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}\sum_{t=1}^{n} \Pr\left\{S_t = S^l_{i,\mathrm{B}}, N_{i,t} > N_{i,t-1}, \forall s \in \tilde{S}_t, N_{s,t-1} > \ell_n\left(\Delta^{i,l}, p_s\right)\right\},
\end{aligned}$$

where the last inequality is due to our way of updating counter N_i by Definition 12: When N_i is incremented in round t such that $N_{i,t} > N_{i,t-1}$, we know that $N_{i,t-1} \cdot p_i$ has the lowest value among all $N_{s,t-1} \cdot p_s$ for $s \in \tilde{S}_t$, and thus $N_{i,t-1} > \ell_n\left(\Delta^{i,l}, p_i\right)$ implies that for $s \in \tilde{S}_t$, $N_{s,t-1} \ge N_{i,t-1} \cdot p_i/p_s > \ell_n\left(\Delta^{i,l}, p_i\right) \cdot p_i/p_s = \ell_n\left(\Delta^{i,l}, p_s\right)$. To prove the lemma, it is sufficient to show that for any $0 < t \le n$,

$$\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]} \Pr\left\{S_t = S^l_{i,\mathrm{B}}, N_{i,t} > N_{i,t-1}, \forall s \in \tilde{S}_t, N_{s,t-1} > \ell_n\left(\Delta^{i,l}, p_s\right)\right\} \tag{16}$$

$$\le (1-\beta) + \frac{(2+\mathbb{I}\{p^*<1\})m}{t^2}. \tag{17}$$

This is because we may then take the union bound on all t's, and get a bound of

$$\sum_{t=1}^{n}\left((1-\beta)+\frac{3m}{t^2}\right)\leq(1-\beta)n+\frac{(2+\mathbb{I}\{p^*<1\})m\pi^2}{6}.$$

Thus, in order to prove our claim, it suffices to prove Inequality (17).

We first split Eq.(16) into two parts:

$$\begin{aligned}
&\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}\Pr\left\{S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t,N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right)\right\}\\
=&\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}\Pr\Bigg\{S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t, &(18)\\
&N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),T_{s,t-1}>\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\Bigg\}\\
&+\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}\Pr\Bigg\{S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t, &(19)\\
&N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),\exists s\in\tilde{S}_t,T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\Bigg\}.\\
=&\Pr\Bigg\{\exists i\in[m],\exists l\in[K_i],S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t, &(20)\\
&N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),T_{s,t-1}>\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\Bigg\}\\
&+\Pr\Bigg\{\exists i\in[m],\exists l\in[K_i],S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1}, &(21)\\
&\forall s\in\tilde{S}_t,N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),\exists s\in\tilde{S}_t,T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\Bigg\}, &(22)
\end{aligned}$$

where the last equality is due to that the events $\{S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1}\}$ for all $i\in[m]$ and $l\in[K_i]$ are mutually exclusive, since $N_{i,t}>N_{i,t-1}$ determines the unique i (at most one N_i is incremented in each round by Definition 12) and then $S_t=S_{i,\mathrm{B}}^l$ determines the unique l.

For the first term in Eq.(22), we apply Lemma 3 and have:

$$\begin{aligned}
&\forall i\in[m]\forall l\in[K_i],\Pr\left\{\mathcal{N}_t,\neg F_t,S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t,\ T_{s,t-1}>\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\right\}=0\Rightarrow\\
&\Pr\left\{\mathcal{N}_t,\neg F_t,\exists i\in[m],\exists l\in[K_i],S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t,\ T_{s,t-1}>\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\right\}=0\Rightarrow\\
&\Pr\left\{\exists i\in[m],\exists l\in[K_i],S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t,\ T_{s,t-1}>\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\right\}\\
&\quad\leq\Pr[F_t\vee\neg\mathcal{N}_t]\leq(1-\beta)+\frac{2m}{t^2}. \qquad (23)
\end{aligned}$$

The inequality in Eq.(23) uses the definition of F_t (Definition 7) and Lemma 1.

For the second term in Eq.(22), we have:

$$\begin{aligned}
&\Pr\Bigg\{\exists i\in[m],\exists l\in[K_i],S_t=S_{i,\mathrm{B}}^l,N_{i,t}>N_{i,t-1},\forall s\in\tilde{S}_t,N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right), &(24)\\
&\exists s\in\tilde{S}_t,T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\Bigg\}\\
&\leq\Pr\left\{\exists i\in[m],\exists l\in[K_i],\exists s\in\tilde{S}_t,N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\right\}\\
&\leq\sum_{s\in\tilde{S}_t}\sum_{k=1}^{t-1}\Pr\left\{\exists i\in[m],\exists l\in[K_i],N_{s,t-1}=k,N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\right\}. \qquad (25)
\end{aligned}$$

Let $\Delta^*(s,k)=\min_{i\in[m],l\in[K_i],\ell_n(\Delta^{i,l},p_s)<k}\Delta^{i,l}$, and $\Delta^*(s,k)=\emptyset$ if the condition of min is not satisfied. Since $f^{-1}(\Delta)$ decreases when Δ decreases, we know that when the event $\{\exists i\in[m],\exists l\in[K_i],N_{s,t-1}=k,N_{s,t-1}>\ell_n\left(\Delta^{i,l},p_s\right),T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^{i,l})^2}\}$ is non-empty, it is included in the event $\{N_{s,t-1}=k,T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^*(s,k))^2}\}$. Therefore, we have

$$\begin{aligned}
(25)&\leq\sum_{s\in\tilde{S}_t}\sum_{k\in[t-1],\Delta^*(s,k)\neq\emptyset}\Pr\left\{N_{s,t-1}=k,T_{s,t-1}\leq\frac{6\cdot\ln t}{f^{-1}(\Delta^*(s,k))^2}\right\}\\
&\leq\sum_{s\in\tilde{S}_t}\sum_{k\in[t-1]}\frac{\mathbb{I}\{p_i<1\}}{t^3} &\{\text{by Lemma 2}\}\\
&\leq\frac{\mathbb{I}\{p^*<1\}m}{t^2}. &(26)
\end{aligned}$$

Combining Eq.(23) and Eq.(26), we obtain Eq.(17). □

Now we consider the bound on under-sampled part, i.e., the number of times that the played bad super arms are under-sampled. For a particular arm i, its counter N_i will increase from 0 to $\ell_n(\Delta^{i,K_i},p_i)$ before it is sufficiently sampled. Assume $N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i)]$ when N_i is incremented at time t with an *under-sampled* super arm $S_{i,B}^l$. We can conclude that $\Delta^{i,l}\leq\Delta^{i,j}$, which will be used as an upper bound for the regret. Otherwise, we must have $\Delta^{i,l}\geq\Delta^{i,j-1}$ and $S_{i,B}^l$ is already *sufficiently sampled*.

To simplify the notation, set $\ell_n(\Delta^{i,0},p_i)=0$. Notice that $N_{i,0}=0$ for all i. For each base arm i, the boundary case of $0=N_{i,t-1}<N_{i,t}$ occurs in only one round in a run, and we treat it separately by using $\Delta_{\max}^i$ as the regret for this case. For the rest, we break the range of the counter $N_{i,t-1}$ with $N_{i,t-1}>0$ into discrete segments, i.e., $(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i)]$ for $j\in[K_i]$. Let us assume that the round t is bad and $N_{i,t}$ is incremented. Assume $N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i)]$ for some j. Notice that we are only interested in the case that S_t is under-sampled. In particular, if this is indeed the case, $S_t=S_{\mathrm{B}}^{i,l}$ for some $l\geq j$. (Otherwise, S_t is sufficiently sampled based on the counter $N_{i,t-1}$.) Therefore, we will suffer a regret of $\Delta^{i,l}\leq\Delta^{i,j}$ (See Definition 6). Consequently, for counter $N_{i,t}$ in range $(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i)]$, we will suffer a total regret for those under-sampled arms at most $(\ell_n(\Delta^{i,j},p_i)-\ell_n(\Delta^{i,j-1},p_i))\cdot\Delta^{i,j}$ in rounds that $N_{i,t}$ is incremented.

Lemma 5 (Bound on under-sampled part). *For any time horizon $n > m$, we have,*

$$\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]} N_{i,n}^{l,und}\cdot\Delta^{i,l} \le \sum_{i\in[m],K_i>0}\left(\ell_n(\Delta^i_{\min},p_i)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,p_i)\mathrm{d}x+\Delta^i_{\max}\right). \tag{27}$$

Proof. It suffices to show that for any arm $i\in[m]$ with $K_i>0$,

$$\sum_{l\in[K_i]} N_{i,n}^{l,und}\cdot\Delta^{i,l} \le \ell_n(\Delta^i_{\min},p_i)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,p_i)\mathrm{d}x+\Delta^i_{\max}.$$

Now, by definition and discussion on the interval that $N_{i,t-1}$ lies in, we have

$$\begin{aligned}
&\sum_{l\in[K_i]} N_{i,n}^{l,und}\cdot\Delta^{i,l}\\
=&\sum_{t=1}^{n}\sum_{l\in[K_i]}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\le\ell_n(\Delta^{i,l},p_i)\}\cdot\Delta^{i,l}\\
=&\sum_{t=1}^{n}\sum_{l\in[K_i]}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},0<N_{i,t-1}\le\ell_n(\Delta^{i,l},p_i)\}\cdot\Delta^{i,l}\\
&+\sum_{t=1}^{n}\sum_{l\in[K_i]}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1}=0\}\cdot\Delta^{i,l}\\
\le&\sum_{t=1}^{n}\sum_{l\in[K_i]}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},0<N_{i,t-1}\le\ell_n(\Delta^{i,l},p_i)\}\cdot\Delta^{i,l}+\Delta^i_{\max}\\
=&\sum_{t=1}^{n}\sum_{l\in[K_i]}\sum_{j=1}^{l}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i))]\}\cdot\Delta^{i,l}+\Delta^i_{\max}\\
\le&\sum_{t=1}^{n}\sum_{l\in[K_i]}\sum_{j=1}^{l}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i))]\}\cdot\Delta^{i,\boldsymbol{j}}+\Delta^i_{\max}
\end{aligned}\tag{28}$$

$$\begin{aligned}
\le&\sum_{t=1}^{n}\sum_{l\in[K_i]}\sum_{j\in[K_i]}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i))]\}\cdot\Delta^{i,j}+\Delta^i_{\max}\\
=&\sum_{t=1}^{n}\sum_{j\in[K_i]}\mathbb{I}\{S_t\in\mathcal{S}_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i))]\}\cdot\Delta^{i,j}+\Delta^i_{\max}.
\end{aligned}\tag{29}$$

The Inequality (28) holds since $\Delta^{i,j}\ge\Delta^{i,l}$ for $j\le l$. Equality (29) is by first switching summations and then merging all $S^l_{i,\mathrm{B}}$ into $\mathcal{S}_{i,\mathrm{B}}$. We may now switch the summations again,

and get

$$\begin{aligned}
(29)=&\sum_{j\in[K_i]}\sum_{t=1}^{n}\mathbb{I}\{S_t\in\mathcal{S}_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},p_i),\ell_n(\Delta^{i,j},p_i))]\}\cdot\Delta^{i,j}+\Delta^i_{\max}\\
\le&\sum_{j\in[K_i]}\left(\lfloor\ell_n(\Delta^{i,j},p_i)\rfloor-\lfloor\ell_n(\Delta^{i,j-1},p_i)\rfloor\right)\cdot\Delta^{i,j}+\Delta^i_{\max} &(30)\\
\le&\sum_{j\in[K_i]}\left(\ell_n(\Delta^{i,j},p_i)-\ell_n(\Delta^{i,j-1},p_i)\right)\cdot\Delta^{i,j}+\Delta^i_{\max}. &(31)
\end{aligned}$$

Inequality (30) uses a relaxation on the indicators. In Inequality (31), for every $j\ge2$, we relax the part of $(\ell_n(\Delta^{i,j-1},p_i)-\lfloor\ell_n(\Delta^{i,j-1},p_i)\rfloor)\cdot\Delta^{i,j}$ to $(\ell_n(\Delta^{i,j-1},p_i)-\lfloor\ell_n(\Delta^{i,j-1},p_i)\rfloor)\cdot\Delta^{i,j-1}$. Now we simply expand the summation, and some terms will be canceled Then, we upper bound the new summation using an integral:

$$\begin{aligned}
(31)=&\ell_n(\Delta^{i,K_i},p_i)\Delta^{i,K_i}+\sum_{j\in[K_i-1]}\ell_n(\Delta^{i,j},p_i)\cdot(\Delta^{i,j}-\Delta^{i,j+1})+\Delta^i_{\max}\\
\le&\ell_n(\Delta^{i,K_i},p_i)\Delta^{i,K_i}+\int_{\Delta^{i,K_i}}^{\Delta^{i,1}}\ell_n(x,p_i)\mathrm{d}x+\Delta^i_{\max} &(32)\\
=&\ell_n(\Delta^i_{\min},p_i)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,p_i)\mathrm{d}x+\Delta^i_{\max}. &(33)
\end{aligned}$$

Inequality (32) comes from the fact that $\ell_n(x,p_i)$ is decreasing in x. □

Finally we are ready to prove our main theorem. We just need to combine the upper bounds from the sufficiently sampled part and the under-sampled part together.

Proof of Theorem 1. Using the counters defined in Definition 13, we may get the expectation of the regret by computing the expectation of the value of the counters after the n-th round. More specifically, according to Definition 4, the expected regret is the difference between $n\cdot\alpha\cdot\beta\cdot\mathrm{opt}_{\boldsymbol{\mu}}$ and the expected reward, which is at least $\alpha\cdot n\cdot\mathrm{opt}_{\boldsymbol{\mu}}$ minus the expected loses from playing bad super arms.

Therefore, combining with Eq.(15) and Eq.(27), the overall regret of our algorithm is

$$
\begin{aligned}
&Reg^{A}_{\boldsymbol{\mu},\alpha,\beta}(n)\\
&\le \mathbb{E}\left[n\cdot\alpha\cdot\beta\cdot \mathrm{opt}_{\boldsymbol{\mu}} - \left(\alpha\cdot n\cdot \mathrm{opt}_{\boldsymbol{\mu}} - \sum_{i\in[m],K_i>0}\left(\sum_{l\in[K_i]}(N^{l,suf}_{i,n}+N^{l,und}_{i,n})\cdot\Delta^{i,l}\right)\right)\right] \quad (34)\\
&\le \Delta_{\max}\cdot\mathbb{E}\left[\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}N^{l,suf}_{i,n}\right]\\
&\quad+\sum_{i\in[m],K_i>0}\left(\ell_n(\Delta^i_{\min},p_i)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,p_i)\mathrm{d}x+\Delta^i_{\max}\right)-(1-\beta)\cdot n\cdot\alpha\cdot\mathrm{opt}_{\boldsymbol{\mu}}\\
&\le \sum_{i\in[m],K_i>0}\left(\ell_n(\Delta^i_{\min},p_i)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,p_i)\mathrm{d}x\right)+\left(\frac{(2+\mathbb{I}\{p^*<1\})\pi^2}{6}+1\right)\cdot m\cdot\Delta_{\max}\\
&\quad+(1-\beta)n\cdot\Delta_{\max}-(1-\beta)\cdot n\cdot\alpha\cdot\mathrm{opt}_{\boldsymbol{\mu}} \quad (35)\\
&\le \sum_{i\in[m],K_i>0}\left(\ell_n(\Delta^i_{\min},p_i)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,p_i)\mathrm{d}x\right)+\left(\frac{(2+\mathbb{I}\{p^*<1\})\pi^2}{6}+1\right)\cdot m\cdot\Delta_{\max}. \quad (36)
\end{aligned}
$$

The last step of derivation from Eq.(35) to Eq.(36) uses the fact that all rewards are nonnegative and thus $\Delta_{\max}\le\alpha\cdot\mathrm{opt}_{\boldsymbol{\mu}}$ by Definition 6. □

3.1.2 Proof of Theorem 2

The proof of Theorem 2 relies on the tight regret bound for the leading $\ln n$ term given by Theorem 1.

Proof of Theorem 2. We first prove the case of $p^*=1$. Following the proof of Theorem 1, we only need to consider the base arms that are played when they are under-sampled. Following the intuition, we need to quantify when Δ is too small. In particular, we measure the threshold for $\Delta^i_{\min}$ based on $N_{i,n}$, i.e., the counter of arm i at time horizon n. Let $\{n_j \mid j\in[m]\}$ be a set of possible counter values at time horizon n. Our analysis will then be conditioned on event $\mathcal{E}=\{\forall j\in[m], N_{j,n}=n_j\}$.

For an arm $i\in[m]$ with $K_i>0$, we have

$$
\begin{aligned}
&\sum_{l\in[K_i]}N^{l,und}_{i,n}\cdot\Delta^{i,l}\mid\mathcal{E}\\
=&\sum_{t=1}^{n}\sum_{l\in[K_i]}\mathbb{I}\{S_t=S^l_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\le\ell_n(\Delta^{i,l},1)\mid\mathcal{E}\}\cdot\Delta^{i,l}
\end{aligned}
$$

With $f(x)=\gamma x^{\omega}$, we have $f^{-1}(x)=\left(\frac{x}{\gamma}\right)^{1/\omega}$. Define $\Delta^*(n_i)=\left(\frac{6\gamma^{2/\omega}\ln n}{n_i}\right)^{\omega/2}$, i.e., $\ell_n(\Delta^*(n_i),1)=n_i$. Now we consider two cases.

Case (1): $\Delta^i_{\min}>\Delta^*(n_i)$. Following the same derivation as in the proof of Lemma 5 (notice that the same derivation still works when conditioned on event $\mathcal{E}$), we have

$$
\begin{aligned}
\sum_{l\in[K_i]}N^{l,und}_{i,n}\cdot\Delta^{i,l}\mid\mathcal{E}&\le\ell_n(\Delta^i_{\min},1)\Delta^i_{\min}+\int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\ell_n(x,1)\mathrm{d}x+\Delta^i_{\max} \quad (37)\\
&=\frac{6\gamma^{\frac{2}{\omega}}\ln n}{(\Delta^i_{\min})^{\frac{2}{\omega}-1}}+\frac{\omega}{2-\omega}6\gamma^{\frac{2}{\omega}}\ln n\left((\Delta^i_{\min})^{1-\frac{2}{\omega}}-(\Delta^i_{\max})^{1-\frac{2}{\omega}}\right)+\Delta^i_{\max}\\
&\le\frac{2}{2-\omega}\cdot\frac{6\cdot\gamma^{\frac{2}{\omega}}\ln n}{(\Delta^i_{\min})^{\frac{2}{\omega}-1}}\le\frac{2\gamma}{2-\omega}\cdot(6\ln n)^{\omega/2}n_i^{1-\omega/2}+\Delta^i_{\max}. \quad (38)
\end{aligned}
$$

The last inequality above is by replacing $\Delta^i_{\min}$ with $\Delta^*(n_i)$.

Case (2): $\Delta^i_{\min}\le\Delta^*(n_i)$. Let $l^*=\min\{l\in[K_i]\mid\Delta^{i,l}\le\Delta^*(n_i)\}$. Notice that $\Delta^{i,l^*}\le\left(\frac{6\gamma^{2/\omega}\ln n}{n_i}\right)^{\omega/2}$. We follow the same derivation as in the proof of Lemma 5, and then we critically use the fact that the counter N_i cannot go beyond n_i (in the first term in Inequality (40)):

$$
\begin{aligned}
&\sum_{l\in[K_i]}N^{l,und}_{i,n}\cdot\Delta^{i,l}\mid\mathcal{E}\\
&\le\sum_{j\in[K_i]}\sum_{t=1}^{n}\mathbb{I}\{S_t\in\mathcal{S}_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},1),\ell_n(\Delta^{i,j},1)]\mid\mathcal{E}\}\cdot\Delta^{i,j}+\Delta^i_{\max} \quad (39)\\
&\le\sum_{j\ge l*}\sum_{t=1}^{n}\mathbb{I}\{S_t\in\mathcal{S}_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},1),\ell_n(\Delta^{i,j},1)]\mid\mathcal{E}\}\cdot\Delta^*(n_i)+\Delta^i_{\max}\\
&\quad+\sum_{j\in[l*-1]}\sum_{t=1}^{n}\mathbb{I}\{S_t\in\mathcal{S}_{i,\mathrm{B}},N_{i,t}>N_{i,t-1},N_{i,t-1}\in(\ell_n(\Delta^{i,j-1},1),\ell_n(\Delta^{i,j},1)]\mid\mathcal{E}\}\cdot\Delta^{i,j}+\Delta^i_{\max}\\
&\le(n_i-\ell_n(\Delta^{i,l^*-1},1))\cdot\Delta^*(n_i)+\sum_{j\in[l^*-1]}(\ell_n(\Delta^{i,j},1)-\ell_n(\Delta^{i,j-1},1))\cdot\Delta^{i,j}+\Delta^i_{\max} \quad (40)\\
&\le n_i\cdot\Delta^*(n_i)+\int_{\Delta^*(n_i)}^{\Delta^{i,1}}\ell_n(x,1)\mathrm{d}x+\Delta^i_{\max}\le\frac{2\gamma}{2-\omega}\cdot(6\ln n)^{\omega/2}n_i^{1-\omega/2}+\Delta^i_{\max}. \quad (41)
\end{aligned}
$$

Therefore, Eq.(41) holds in both cases. We then have

$$
\begin{aligned}
\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]}N^{l,und}_{i,n}\cdot\Delta^{i,l}\mid\mathcal{E}&\le\frac{2\gamma}{2-\omega}\cdot(6\ln n)^{\omega/2}\cdot\sum_{i\in[m],K_i>0}n_i^{1-\omega/2}+\Delta^i_{\max}\\
&\le\frac{2\gamma}{2-\omega}\cdot(6m\ln n)^{\omega/2}\cdot n^{1-\omega/2}+\Delta^i_{\max}. \quad (42)
\end{aligned}
$$

The last inequality comes from Jensen's inequality and $\sum_i n_i\le n$. Since the final inequality does not depend on n_i, we can drop the condition $\mathcal{E}$ above. With the bound on the under-sampled part given in Inequality (42), we combine it with the result on sufficiently

sampled part given in Lemma 4, then we can following the similar derivation as shown from Eq.(34) to Eq.(36) to derive the distribution-independent regret bound given in Theorem 2 for the case of $p^* = 1$.

We now prove the case of $p^* < 1$. The proof is essentially the same, but with a different definition of $\ell_n(\Delta, p)$. For convenience, we relax $\ell_n(\Delta, p) = \max\left(\frac{12\cdot\ln n}{(f^{-1}(\Delta))^2\cdot p}, \frac{24\cdot\ln n}{p}\right)$ to $\frac{12\cdot\ln n}{(f^{-1}(\Delta))^2\cdot p} + \frac{24\cdot\ln n}{p}$. In this case, we define $\Delta_i^*(n_i) = \left(\frac{12\gamma^{2/\omega}\ln n}{p_i n_i}\right)^{\omega/2}$.

For Case (1): $\Delta^i_{\min} > \Delta_i^*(n_i)$, following the same derivation as Eq.(37)—(38) except that we use $\ell_n(\cdot, p_i)$ instead of $\ell_n(\cdot, 1)$ (Definition 9), we have

$$\sum_{l\in[K_i]} N^{l,und}_{i,n}\cdot\Delta^{i,l} \mid \mathcal{E} \le \frac{2\gamma}{2-\omega}\cdot\left(\frac{12\ln n}{p_i}\right)^{\omega/2} n_i^{1-\omega/2} + \frac{24\ln n}{p_i}\cdot\Delta^i_{\max} + \Delta^i_{\max}.$$

For Case (2): $\Delta^i_{\min} \le \Delta_i^*(n_i)$, again following the same derivation Eq.(39)—(41) except that we use $\ell_n(\cdot, p_i)$ instead of $\ell_n(\cdot, 1)$,, we have

$$\sum_{l\in[K_i]} N^{l,und}_{i,n}\cdot\Delta^{i,l} \mid \mathcal{E} \le \frac{2\gamma}{2-\omega}\cdot\left(\frac{12\ln n}{p_i}\right)^{\omega/2} n_i^{1-\omega/2} + \frac{24\ln n}{p_i}\cdot\Delta^i_{\max} + \Delta^i_{\max}.$$

Together, we have

$$\begin{aligned}
&\sum_{i\in[m],K_i>0}\sum_{l\in[K_i]} N^{l,und}_{i,n}\cdot\Delta^{i,l} \mid \mathcal{E}\\
&\le \frac{2\gamma}{2-\omega}\cdot\left(\frac{12\ln n}{p^*}\right)^{\omega/2}\sum_{i\in[m],K_i>0} n_i^{1-\omega/2} + \sum_{i\in[m],K_i>0}\frac{24\ln n}{p_i}\cdot\Delta^i_{\max}\\
&\le \frac{2\gamma}{2-\omega}\cdot\left(\frac{12m\ln n}{p^*}\right)^{\omega/2} n^{1-\omega/2} + \sum_{i\in[m]}\frac{24\ln n}{p_i}\cdot\Delta_{\max} + \Delta^i_{\max}.
\end{aligned}$$

Finally, combining Lemma 4 and the derivation for the regret bound as shown from Eq.(34) to Eq.(36), we obtain the regret bound for the case of $p^* < 1$. □

3.2 Discussions

We may further improve the bound in Theorem 1 as follows, when all the triggering probabilities are 1.

Improving the coefficient of the leading term when $\forall i, p_i = 1$. In general, we can set $\bar{\mu}_i = \hat{\mu}_i + \sqrt{y/(2T_i)}$ for some y in line 6 in the CUCB algorithm. The corresponding regret bound obtained is

$$\sum_{i\in[m],K_i>0}\left(\frac{2\cdot y}{(f^{-1}(\Delta^i_{\min}))^2}\cdot\Delta^i_{\min} + \int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\frac{2\cdot y}{(f^{-1}(x))^2}\mathrm{d}x\right) + \left(1+\sum_{t=1}^{n}\frac{2t}{e^{-y}}\right)\cdot m\cdot\Delta_{\max}.$$

What we need is to make sure the term $\sum_{t=1}^{n}\frac{2t}{e^{-y}}$ in the above regret bound converges. We can thus set y appropriately to guarantee convergence while improving the constant in the

leading term. One way is setting $y = (1+c)\ln t$ with a constant $c > 1$, or equivalently setting $\bar{\mu}_i = \hat{\mu}_i + \sqrt{(1+c)\ln t/(2T_i)}$, so that $\sum_{t=1}^{n}\frac{2t}{e^{-y}} = 2\sum_{t=1}^{n} t^{-c} \le 2\zeta(c)$, where $\zeta(c) = \sum_{t=1}^{\infty}\frac{1}{t^c}$ is the Riemann's zeta function, and has a finite value when $c > 1$. Then the regret bound is

$$\sum_{i\in[m],K_i>0}\left(\frac{2\cdot(1+c)\cdot\ln n}{(f^{-1}(\Delta^i_{\min}))^2}\cdot\Delta^i_{\min} + \int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\frac{2\cdot(1+c)\cdot\ln n}{(f^{-1}(x))^2}\mathrm{d}x\right) + (2\cdot\zeta(c)+1)\cdot m\cdot\Delta_{\max}.$$

We can also further improve the constant factor from $2(1+c)$ to 4 by setting $\bar{\mu}_i = \hat{\mu}_i + \sqrt{\frac{2\ln t+\ln\ln t}{2T_i}}$ at the cost of a second order $\ln\ln n$ term (Garivier and Cappé, 2011), with regret at most

$$\sum_{i\in[m],K_i>0}\left(\frac{2\cdot(2\ln n+\ln\ln n)}{(f^{-1}(\Delta^i_{\min}))^2}\cdot\Delta^i_{\min} + \int_{\Delta^i_{\min}}^{\Delta^i_{\max}}\frac{2\cdot(2\ln n+\ln\ln n)}{(f^{-1}(x))^2}\mathrm{d}x\right) + (1+2\ln\ln n)\cdot m\cdot\Delta_{\max}.$$

This is because $\sum_{t=1}^{n}\frac{1}{t\ln t} \le \int_{m}^{n}\frac{1}{t\ln t}\mathrm{d}t \le \ln\ln n$ when $m > e$.

Comparing to classical MAB. As we discussed earlier, the classical MAB is a special instance of our CMAB framework in which each super arm is a simple arm, $p_i = 1$ for all $i \in [m]$, function $f(\cdot)$ is the identity function, and $\alpha = \beta = 1$. Notice that $\Delta^i_{\max} = \Delta^i_{\min}$. Thus, by Theorem 1, the regret bound of the classical MAB is

$$\sum_{i\in[m],\Delta^i>0}\frac{6\ln n}{\Delta^i} + \left(\frac{\pi^2}{3}+1\right)\cdot m\cdot\Delta_{\max}, \tag{43}$$

where $\Delta^i = \max_{j\in[m]}\mu_j - \mu_i$. Comparing with the regret bound in Theorem 1 of the paper by Auer et al. (2002a), we see that we even have a better coefficient $\sum_{i\in[m],\Delta^i>0} 6/\Delta^i$ in the leading $\ln n$ term than the one $\sum_{i\in[m],\Delta^i>0} 8/\Delta^i$ in the original analysis of UCB1.[3] The improvement is due to a tighter analysis, and is the reason that we obtained improved regret over the regret obtained by Gai et al. (2012). Thus, the regret upper bound of our CUCB algorithm when applying to the classical MAB problem is at the same level (up to a constant factor) as UCB1, which is designed specifically for the MAB problem.

4. Applications

In this section, we describe two applications with non-linear reward functions as well as the class of linear reward applications that fit our CMAB framework. Notice that, the probabilistic maximum coverage bandit and social influence maximization bandit are also instances of the online submodular maximization problem, which can be addressed in the adversarial setting by Streeter and Golovin (2008), but we are not aware of their counterpart in the stochastic setting.

4.1 Probabilistic Maximum Coverage Bandit

The online advertisement placement application discussed in the introduction can be modeled by the bandit version of the probabilistic maximum coverage (PMC) problem. PMC

3. We remark that the constant of UCB1 has been tightened to the optimum (Garivier and Cappé, 2011).

has as input a weighted bipartite graph $G = (L, R, E)$ where each edge (u, v) has a probability $p(u, v)$, and it needs to find a set $S \subseteq L$ of size k that maximizes the expected number of activated nodes in R, where a node $v \in R$ can be activated by a node $u \in S$ with an independent probability of $p(u, v)$. In the advertisement placement scenario, L is the set of web pages, R is the set of users, and $p(u, v)$ is the probability that user v clicks the advertisement on page u. PMC problem is NP-hard, since when all edge probabilities are 1, it becomes the NP-hard Maximum Coverage problem.

Using submodular set function maximization technique (Nemhauser et al., 1978), it can be easily shown that there exists a deterministic $(1 - 1/e)$ approximation algorithm for the PMC problem, which means that we have a $(1 - 1/e, 1)$-approximation oracle for PMC.

The PMC bandit problem is that edge probabilities are unknown, and one repeatedly selects k targets in L in multiple rounds, observes all edge activations and adjusts target selection accordingly in order to maximize the total number of activated nodes over all rounds.

We can formulate this problem as an instance in the CMAB framework. Each edge $(u, v) \in E$ represents an arm, and each play of the arm is a 0-1 Bernoulli random variable with parameter $p_{u,v}$. A super arm is the set of edges E_S incident to a set $S \subseteq L$ of size k. The reward of E_S is the number of activated nodes in R, which is the number of nodes in R that are incident to at least one edge in E_S with outcome 1. Since all arms are independent Bernoulli random variables, we know that the expected reward only depends on the probabilities on all edges. In particular we have that the expected reward $r_{\boldsymbol{\mu}}(E_S) = \sum_{v \in R}(1 - \prod_{u \in L,(u,v) \in E_S}(1 - p(u, v)))$. Note that this expected reward function is not linear in $\boldsymbol{\mu} = \{p(u, v)\}_{(u,v) \in E}$. For all arm $i \in E$, we have $p_i = 1$, that is, we do not have probabilistically triggered arms. The monotonicity property is straightforward. The bounded smoothness function is $f(x) = |E| \cdot x$, i.e., increasing all probabilities of all arms in a super arm by x can increase the expected number of activated nodes in V by at most $|E| \cdot x$. Since $f(\cdot)$ is a linear function, the integral in Eq.(2) has a closed form. In particular, by Theorem 1, we know that the distribution-dependent $(1 - 1/e, 1)$-approximation regret bound of our CUCB algorithm on PMC bandit is

$$\sum_{i \in E, K_i > 0} \frac{12 \cdot |E|^2 \cdot \ln n}{\Delta^i_{\min}} + \left(\frac{\pi^2}{3} + 1\right) \cdot |E| \cdot \Delta_{\max}.$$

Notice that all edges incident to a node $u \in L$ are always played together. In other words, these edges can share one counter. We call these arms (edges) as *clustered arms*. It is possible to exploit this property to improve the coefficient of the $\ln n$ term, so that the summation is not among all edges but only nodes in L. (See Section 4.1 and the supplementary material of Chen et al. (2013) for the regret bound and analysis for the case of clustered arms).

From Theorem 2, we also have the distribution-independent regret bound of

$$\sqrt{24|E|^3 n \ln n} + \left(\frac{\pi^2}{3} + 1\right) \cdot |E| \cdot \Delta_{\max}.$$

Note that for the PMC bandit, $\Delta_{\max}$ is at most the number of vertices covered in R, and thus $\Delta_{\max} \leq |R|$.

4.2 Combinatorial Bandits With Linear Rewards

Gai et al. (2012) studied the *Learning with Linear Reward* policy (LLR). Their formulation is close to ours except that their reward function must be linear. In their setting, there are m underlying arms. There are a finite number of super arms, each of which consists of a set of underlying arms S together with a set of coefficients $\{w_{i,S} \mid i \in S\}$. The reward of playing super arm S is $\sum_{i \in S} w_{i,S} \cdot X_i$, where X_i is the random outcome of arm i. The formulation can model a lot of bandit problems appeared in the literature, e.g., multiple plays, shortest path, minimum spanning tree and maximum weighted matching.

Our framework contains such linear reward problems as special cases.[4] In particular, let $L = \max_S |S|$ and $a_{\max} = \max_{i,S} w_{i,S}$, and we have the bounded smoothness function $f(x) = a_{\max} \cdot L \cdot x$. In this setting we have $p_i = 1$ for all $i \in [m]$. By applying Theorem 1, the regret bound is

$$\left(\sum_{i \in [m], K_i > 0} \frac{12 \cdot a^2_{\max} \cdot L^2 \cdot \ln n}{\Delta^i_{\min}}\right) + \left(\frac{\pi^2}{3} + 1\right) \cdot m \cdot \Delta_{\max}.$$

Our result significantly improves the coefficient of the leading $\ln n$ term comparing to Theorem 2 in the paper by Gai et al. (2012) in two aspects: (a) we remove a factor of $L+1$; and (b) the coefficient $\sum_{i \in [m], \Delta^i_{\min} > 0} 1/\Delta^i_{\min}$ is likely to be much smaller than $m \cdot \Delta_{\max}/(\Delta_{\min})^2$ used by Gai et al. (2012). This demonstrates that while our framework covers a much larger class of problems, we are still able to provide much tighter analysis than the one for linear reward bandits. Moreover, applying Theorem 2 we can obtain distribution-independent bound for combinatorial bandits with linear rewards, which is not provided by Gai et al. (2012):

$$a_{\max} L \sqrt{24 m n \ln n} + \left(\frac{\pi^2}{3} + 1\right) \cdot m \cdot \Delta_{\max}.$$

Note that, for the class of linear bandits, the reward is at most $a_{\max} \cdot L$, and thus $\Delta_{\max} \leq a_{\max} \cdot L$.

We remark that, in a latest paper, Kveton et al. (2015) show that the above regret bounds can be improved to $O(L \log n \sum_i 1/\Delta^i_{\min})$ for distribution-dependent regret and $O(\sqrt{Lmn \log n})$ for distribution-independent regret, respectively, which are tight (up to a factor of $\sqrt{\log n}$ for the distribution-independent bound). The improvement is achieved by a weaker and non-uniform sufficient sampling condition—in our analysis, we require all relevant base arms of a super arm S_t played in round t to be sufficiently sampled to ensure that S_t cannot be a bad super arm (Lemma 3), but Kveton et al. (2015) relax this and show that it is enough to have sufficiently many base arms to be sampled sufficiently many times, while the rest arms only need to satisfy some weaker sufficient sampling condition. The intuition is that due to linear reward summation, as long as many base arms are sufficiently sampled and the rest have a weaker sufficiently sampled condition, the sum of the errors would be still small enough to guarantee that a good super arm is selected by the oracle. However, it is unclear if this technique can be applied to non-linear reward functions

4. To include the linear reward case, we allow two super arms with the same set of underlying arms to have different sets of coefficients. This is fine as long as the oracle could output super arms with appropriate parameters.

satisfying our bounded smoothness assumption, since the estimate error of each base arm may not linearly affect the estimate error in the expected reward.

4.3 Application to Social Influence Maximization

In social influence maximization with the independent cascade model (Kempe et al., 2003), we are given a directed graph $G = (V, E)$, where every edge (u, v) is associated with an unknown *influence probability* $p_{u,v}$. Initially, a seed set $S \subseteq V$ are selected and activated. In each iteration of the diffusion process, each node u activated in the previous iteration has one chance of activating its inactive outgoing neighbor v independently with probability $p_{u,v}$. The reward of S after the diffusion process is the total number of activated nodes in the end. Influence maximization is to find a seed set S of at most k nodes that maximize the expected reward, also referred to as the *influence spread* of seed set S. Kempe et al. (2003) show that the problem is NP-hard and provide an algorithm with approximation ratio $1 - 1/e - \varepsilon$ with success probability $(1 - 1/|E|)$ for any fixed $\varepsilon > 0$. This means that we have a $(1 - 1/e - \varepsilon, 1 - 1/|E|)$-approximation oracle.

In the CMAB framework, we do not know the activation probabilities of edges and want to learn them during repeated seed selections while maximizing overall reward. Each edge in E is considered as a base arm, and a super arm in this setting is the set E_S of edges incident to the seed set S. Note that these edges will be deterministically triggered, but other edges not in E_S may also be triggered, and the reward is related to all the triggered arms. Therefore, this is an instance where arms may be probabilistically triggered, and thus $p_i < 1$ for some $i \in E$.

It is straightforward to see that the expected reward function is still a function of probabilities on all edges, and the monotonicity holds. However, bounded smoothness property is nontrivial to argue, as we will show in the following lemma.

Lemma 6. *The social influence maximization instance satisfies the bounded smoothness property with bounded smoothness function $f(x) = |E||V|x$.*

Proof. For the social influence maximization bandit, the expectation vector $\boldsymbol{\mu}$ is the vector of all probabilities on all edges. For a seed set $S \subseteq V$, the corresponding super arm is the set E_S of edges incident to vertices in S. Without loss of generality, we assume that for any edge $i \in E$, its probability $\mu_i > 0$. Then for super arm E_S, the set of base arms that can be triggered by E_S, denoted as $\tilde{E}_S$, is exactly the set of edges reachable from seed set S (an edge (u, v) reachable from a set S means its starting vertex u is reachable from S). By Definition 1, to show bounded smoothness with bounded smoothness function $f(x) = |E||V|x$, we need to show that for any two expectation vectors $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$ and for any $\Lambda > 0$, we have $|r_{\boldsymbol{\mu}}(E_S) - r_{\boldsymbol{\mu}'}(E_S)| \le f(\Lambda)$ if $\max_{i \in \tilde{E}_S} |\mu_i - \mu'_i| \le \Lambda$.

Since we know that monotonicity holds, it is sufficient to assume that for all $i \in \tilde{E}_S$, $\mu_i = \mu'_i + \Lambda$. This is because without loss of generality, we can assume $r_{\boldsymbol{\mu}}(E_S) \ge r_{\boldsymbol{\mu}'}(E_S)$, and if $\mu_i < \mu'_i + \Lambda$ we can increase μ_i and decrease μ'_i such that $\mu_i = \mu'_i + \Lambda$, and this only increase the gap between $r_{\boldsymbol{\mu}}(E_S)$ and $r_{\boldsymbol{\mu}'}(E_S)$. Thus, henceforth let us assume that $i \in \tilde{E}_S$, $\mu_i = \mu'_i + \Lambda$.

Starting from $\boldsymbol{\mu}'$, we take one edge i_1 in $\tilde{E}_S$, and increase μ'_{i_1} to $\mu'_{i_1} + \Lambda = \mu_{i_1}$ to get a new vector $\boldsymbol{\mu}^{(1)}$. Suppose the edge i_1 is (u_1, v_1). Comparing $\boldsymbol{\mu}'$ with $\boldsymbol{\mu}^{(1)}$, the only

difference is that the probability on edge (u_1, v_1) increases by Λ. For the influence spread of seed set S, the above change increases the activation probability of v_1 and every node reachable from v_1 by at most Λ. Thus the total increase of influence spread is at most $|V|\Lambda$. Then we select the second edge i_2 in $\tilde{E}_S$ and increases its probability by Λ. By the same argument, the influence spread increases at most $|V|\Lambda$. Repeating the above process, after selecting all edges in $\tilde{E}_S$, we obtain probability vector $\boldsymbol{\mu}^{(s)}$ where $s = |\tilde{E}_S|$, and the increase in influence spread is at most $s|V|\Lambda$. Comparing vector $\boldsymbol{\mu}^{(s)}$ with $\boldsymbol{\mu}$, they are the same on all edges in $\tilde{E}_S$, and may only differ in the rest of edges. However, since the rest of edges cannot be reachable from S, their difference will not affect the influence spread of S. Therefore, we know that the difference between influence spread $r_{\boldsymbol{\mu}}(E_S)$ and $r_{\boldsymbol{\mu}'}(E_S)$ is at most $s|V|\Lambda \le |E||V|\Lambda$. This concludes that if we use function $f(x) = |E||V|x$, the bounded smoothness property holds. □

Remark. In Section 4.2 of the original CMAB paper (Chen et al., 2013), we made a claim that social influence maximization bandit satisfies the bounded smoothness property (with function $f(x) = |E||V|x$) that does not consider probabilistically triggered arms, that is, it satisfies the property that for any two expectation vectors $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$ and for any $\Lambda > 0$, $|r_{\boldsymbol{\mu}}(E_S) - r_{\boldsymbol{\mu}'}(E_S)| \le f(\Lambda)$ if $\max_{i \in E_S} |\mu_i - \mu'_i| \le \Lambda$. This claim is incorrect. For example, all edges in E_S could have the same probability (and thus we could have Λ to be arbitrarily small), but other edges reachable from E_S have different probabilities, and thus the gap between $r_{\boldsymbol{\mu}}(E_S)$ and $r_{\boldsymbol{\mu}'}(E_S)$ will not be arbitrarily small and cannot be bounded by $f(\Lambda)$ for any continuous f tending to zero when Λ tends to zero.

With $f(x) = |E||V|x$, we have $\ell_n(\Delta, p) = \max\left(\frac{12 \cdot \ln n}{(f^{-1}(\Delta))^2 \cdot p}, \frac{24 \cdot \ln n}{p}\right) = \max\left(\frac{12|V|^2|E|^2 \ln n}{\Delta^2 \cdot p}, \frac{24 \cdot \ln n}{p}\right)$. Since Δ is at most $\Delta_{\max}$ in the regret bound and $\Delta_{\max} \le |V|$, it is clear that we have $\ell_n(\Delta, p) = \frac{12|V|^2|E|^2 \ln n}{\Delta^2 \cdot p}$. Then applying Theorem 1, we know that the distribution-dependent $(1 - 1/e - \varepsilon, 1 - 1/|E|)$-approximation regret bound of the CUCB algorithm on influence maximization is:

$$\sum_{i \in E, K_i > 0} \frac{24 \cdot |V|^2|E|^2 \cdot \ln n}{\Delta^i_{\min} \cdot p_i} + \left(\frac{\pi^2}{2} + 1\right) \cdot |E| \cdot \Delta_{\max}.$$

With Theorem 2 (and further using $\ell_n(\Delta, p) = \frac{12|V|^2|E|^2 \ln n}{\Delta^2 \cdot p}$ instead of the relaxed $\ell_n(\Delta, p) = \frac{12|V|^2|E|^2 \ln n}{\Delta^2 \cdot p} + \frac{24 \cdot \ln n}{p}$ as in the proof of Theorem 2), we obtain the distribution-independent bound:

$$|V|\sqrt{\frac{48|E|^3 n \ln n}{p^*}} + \left(\frac{\pi^2}{2} + 1\right) \cdot |E| \cdot \Delta_{\max}.$$

5. Conclusion

In this paper, we propose the first general stochastic CMAB framework that accommodates a large class of nonlinear reward functions among combinatorial and stochastic arms, and it even accommodates probabilistically triggered arms such as what occurs in the viral marketing application. We provide CUCB algorithm with tight analysis on its distribution-dependent and distribution-independent regret bounds and applications to new practical combinatorial bandit problems.

There are many possible future directions from this work. One may study the CMAB problems with Markovian outcome distributions on arms, or the restless version of CMAB, in which the states of arms continue to evolve even if they are not played. Another direction is to investigate if some of the results in this paper are tight or can be further improved. For example, for the nonlinear bounded smoothness function $f(x) = \gamma \cdot x^{\omega}$ with $\omega < 1$, if our bound in Theorem 2 is tight or can be improved; and for the case of probabilistic triggering, if the regret bound dependency on $1/p_i$ is necessary. For the latter case, one may also look into improvement specifically for the influence maximization application. For nonlinear reward functions, currently we assume that the expected reward is a function of the expectation vector of base arms. One may also look into the more general cases where the expected reward depends not only on the expected outcomes of base arms.

References

Rajeev Agrawal. The continuum-armed bandit problem. *SIAM J. Control Optim.*, 33(6): 1926–1951, 1995.

Venkatachalam Anantharam, Pravin Varaiya, and Jean Walrand. Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays — Part I: i.i.d. rewards. *IEEE Transactions on Automatic Control*, AC-32(11):968–976, 1987.

Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. Minimax policies for adversarial and stochastic bandits. In *Proceedings of the 22nd Annual Conference on Learning Theory (COLT)*, 2009.

Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. Minimax policies for combinatorial prediction games. In *Proceedings of the 24th Annual Conference on Learning Theory (COLT)*, 2011.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002a.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002b.

Donald A. Berry and Bert Fristedt. *Bandit problems: Sequential Allocation of Experiments*. Chapman and Hall, 1985.

Sébastien Bubeck, Nicolò Cesa-Bianchi, and Sham M. Kakade. Towards minimax policies for online linear optimization with bandit feedback. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, 2012.

Felipe Caro and Jérémie Gallien. Dynamic assortment with demand learning for seasonal consumer goods. *Management Science*, 53:276–292, 2007.

Nicolò Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. In *Proceedings of the 22nd Conference on Learning Theory*, 2009.

Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework, results, and applications. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation. In *Proceedings of IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2010.

Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking*, 20, 2012.

Aurélien Garivier and Olivier Cappé. The KL-UCB algorithm for bounded stochastic bandits and beyond. In *Proceedings of the 24th Annual Conference on Learning Theory (COLT)*, 2011.

Aditya Gopalan, Shie Mannor, and Yishay mansour. Thompson sampling for complex online problems. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

Elad Hazan and Satyen Kale. Online submodular minimization. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS)*, 2009.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Sham M. Kakade, Adam Tauman Kalai, and Katrina Ligett. Playing games with approximation algorithms. *SIAM Journal on Computing*, 39(3):1088–1106, 2009.

David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 137–146, 2003.

Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *ACM Symposium on Theory of Computing (STOC)*, 2008.

Robert D. Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *Proceedings of the 17th Annual Conference on Neural Information Processing Systems (NIPS)*, 2004.

Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Brian Eriksson. Matroid bandits: Fast combinatorial optimization with learning. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.

Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvári. Tight regret bounds for stochastic combinatorial semi-bandits. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, 2015. to appear, with arxiv version arXiv:1410.0949.

Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.

Tian Lin, Bruno Abrahao, Robert Kleinberg, John C. S. Lui, and Wei Chen. Combinatorial partial monitoring game with linear feedback and its applications. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

Haoyang Liu, Keqin Liu, and Qing Zhao. Logarithmic weak regret of non-bayesian restless multi-armed bandit. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2011.

Keqin Liu and Qing Zhao. Adaptive shortest-path routing under unknown and stochastically varying link states. In *Proceedings of the 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2012.

Shie Mannor and Ohad Shamir. From bandits to experts: On the value of side-observations. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

Michael Mitzenmacher and Eli Upfal. *Probability and Computing*. Cambridge University Press, 2005.

G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.

Lijing Qin, Shouyuan Chen, and Xiaoyan Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In *Proceedings of the 2014 SIAM International Conference on Data Mining (SDM)*, 2014.

Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, 2008.

Matthew Streeter and Daniel Golovin. An online algorithm for maximizing submodular functions. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2008.

Matthew Streeter, Daniel Golovin, and Andreas Krause. Online learning of assignments. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS)*, 2009.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2004.

Journal of Machine Learning Research 17 (2016) 1-42 Submitted 9/14; Revised 9/15; Published 4/16

Differentially Private Data Releasing for Smooth Queries

Ziteng Wang WANGZT2012@GMAIL.COM
Key Laboratory of Machine Perception (MOE), School of EECS
Peking University
Beijing, 100871, China

Chi Jin CHIJIN@CS.BERKELEY.EDU
Department of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Kai Fan KAI.FAN@DUKE.EDU
Computational Biology & Bioinformactics
Duke University
Durham, NC 27708, USA

Jiaqi Zhang ZHANGJQ@CIS.PKU.EDU.CN
Key Laboratory of Machine Perception (MOE), School of EECS
Peking University
Beijing, 100871, China

Junliang Huang HUANGJUNLIANG@PKU.EDU.CN
School of Mathematical Sciences
Peking University
Beijing, 100871, China

Yiqiao Zhong YIQIAOZHONG@PKU.EDU.CN
School of Mathematical Sciences
Peking University
Beijing, 100871, China

Liwei Wang WANGLW@CIS.PKU.EDU.CN
Key Laboratory of Machine Perception (MOE), School of EECS
Peking University
Beijing, 100871, China

Editor: Sanjoy Dasgupta

Abstract

In the past few years, differential privacy has become a standard concept in the area of privacy. One of the most important problems in this field is to answer queries while preserving differential privacy. In spite of extensive studies, most existing work on differentially private query answering assumes the data are *discrete* (i.e., in $\{0,1\}^d$) and focuses on queries induced by *Boolean* functions. In real applications however, *continuous* data are at least as common as binary data. Thus, in this work we explore a less studied topic, namely, differential privately query answering for continuous data with continuous function. As a first step towards the continuous case, we study a natural class of linear queries on continuous data which we refer to as *smooth* queries. A linear query is said to be K-smooth if it is specified by a function defined on $[-1,1]^d$ whose partial derivatives up to order K are all bounded. We develop two ϵ-differentially private mechanisms which are able to answer *all* smooth queries. The first mechanism outputs a summary of the database and can then give answers to the queries. The second mechanism is an improvement of the first one and it outputs a synthetic database. The two mechanisms both achieve an accuracy of $O(n^{-\frac{K}{2d+K}}/\epsilon)$. Here we assume that the dimension d is a constant. It turns out that even in this parameter setting (which is almost trivial in the discrete case), using existing discrete mechanisms to answer the smooth queries is difficult and requires more noise. Our mechanisms are based on L_∞-approximation of (transformed) smooth functions by low-degree even trigonometric polynomials with uniformly bounded coefficients. We also develop practically efficient variants of the mechanisms with promising experimental results.[1]

Keywords: differential privacy, smooth queries, synthetic dataset

1. Introduction

Statistical analysis and machine learning are often conducted on data sets containing sensitive information, such as medical records, commercial data, etc. The benefit of mining from such data is tremendous. But when releasing sensitive data, one must take privacy issue into consideration, and has to accept a tradeoff between the accuracy and the amount of privacy loss of the individuals in the database.

In this paper, we consider *differential privacy* (Dwork et al., 2006), which has become a standard concept in the area of privacy. Roughly speaking, a mechanism which releases information about some database is said to preserve differential privacy, if the change of a single database element does not affect the probability distribution of the output significantly. Differential privacy provides strong guarantees against attacks. It ensures that the risk of information leakage for any individual who submits her information to the database is very small, in the sense that an adversary can discover almost nothing new from the database that contains one individual's information compared with that from the database without that individual's information. Recently there have been extensive studies of machine learning, statistical estimation, and data mining under the differential privacy framework (Wasserman and Zhou, 2010; Chaudhuri et al., 2011; Lei, 2011; Kifer and Lin, 2010; Chaudhuri et al., 2012; Williams and McSherry, 2010; Jain et al., 2012; Chaudhuri and Hsu, 2011).

Accurately answering statistical queries is a well studied problem in differential privacy. A simple and efficient method is the Laplace mechanism (Dwork et al., 2006), which adds Laplace noise to the true answers. Laplace mechanism is especially useful for queries with low sensitivity, which is the maximal difference of the query values of two databases that are different in only one item. A typical class of queries that has low sensitivity is *linear* queries, whose sensitivity is $O(1/n)$, where n is the size of the database.

The Laplace mechanism has a limitation. It can answer at most $O(n^2)$ queries. If the number of queries is substantially larger than n^2, Laplace mechanism is not able to provide differentially private answers with nontrivial accuracy. Considering that potentially there are many users and each user may submit a set of queries, limiting the number of total

1. Part of this work has been appeared in (Wang et al., 2013).

1. Source codes available at `http://www.cis.pku.edu.cn/faculty/vision/wangliwei/software.html`

queries to be smaller than n^2 is too restricted in some situations. A remarkable result due to Blum, Ligett and Roth (Blum et al., 2008) (will be referred to as BLR in this paper) shows that: information theoretically it is possible for a mechanism to answer far more than n^2 linear queries while preserving differential privacy and nontrivial accuracy simultaneously.

There is a series of works (Dwork et al., 2009, 2010; Roth and Roughgarden, 2010; Hardt and Rothblum, 2010) improving the result of (Blum et al., 2008). All these mechanisms are very powerful in the sense that they can answer general and adversely chosen queries. On the other hand, even the fastest algorithms for query answering (Hardt and Rothblum, 2010; Hardt et al., 2012a) run in time linear in the size of the data universe. Often the size of the data universe is much larger than that of the database, so these mechanisms are inefficient. Recently, Ullman (2013) shows that there is no polynomial time algorithm that can answer $n^{2+o(1)}$ *general* linear queries while preserving privacy and accuracy (assuming the existence of one-way function).

Recently, there are growing interests in studying differentially private mechanisms for *restricted* classes of queries, in particular queries useful in applications. One class of queries that attracts a lot of attentions are the k-way conjunctions. The data universe for this problem is $\{0,1\}^d$. Thus each individual record has d binary attributes. A k-way conjunction query is specified by k features. The query asks what fraction of the individual records in the database has all these k features being 1. A series of works attack this problem using several different techniques (Barak et al., 2007; Gupta et al., 2011; Cheraghchi et al., 2012; Hardt et al., 2012b; Thaler et al., 2012) . They proposed elegant mechanisms which run in time $\text{poly}(n)$ when k is a constant. Another class of queries that yields efficient mechanisms is the class of sparse query. A query is m-sparse if it takes non-zero values on at most m elements in the data universe. Blum and Roth (2013) developed mechanisms which are efficient when $m = \text{poly}(n)$.

The above differentially private mechanisms can also be categorized into two classes according to the output of the algorithm. The first class of algorithms output answers to the queries. The second class of algorithms output synthetic databases instead; the answers of the queries can then be simply computed from the synthetic database. The Laplace mechanism belongs to the first class. BLR and the offline version of the Private Multiplicative Weight updating (PMW) mechanism (Hardt and Rothblum, 2010; Hardt et al., 2012a) output synthetic database. From a practical point of view, the synthetic dataset output is appealing. In fact, before the notion of differential privacy was proposed, almost all practical techniques developed to preserve privacy against certain types of attacks output a synthetic dataset by modifying the raw dataset (please see the survey Aggarwal and Yu 2008 and the references therein).

Among the studies of differentially private query answering, most existing works focus on binary data and queries induced by Boolean functions[2]. In real applications however, continuous data are at least as common as binary data; and one has to use continuous functions in this scenario instead of Boolean functions. In this paper, we explore this relatively less studied topic, i.e., differentially private query answering with continuous

2. On the other hand, there are many results (e.g., Chaudhuri et al. 2011, Williams and McSherry 2010, Jain et al. 2012, Chaudhuri and Hsu 2011) on differentially private learning that study continuous data and continuous function.

functions on continuous data. We assume that 1) the data universe is $\mathcal{X} = [-1,1]^d$; 2) a linear query q_f is induced by a continuous function $f : \mathcal{X} \to \mathbb{R}$.

As will be clear soon, answering general linear queries in the continuous setting is considerably more difficult than that in the binary case. Therefore we will focus our analysis on a restricted class of linear queries. The first question is: which subclass of linear queries is commonly used in practice? Let $D = \{x_1, \ldots, x_n\}$ ($x_i \in [-1,1]^d$) be a database consisting of continuous data. A linear query q_f on D is defined as $q_f(D) := \frac{1}{n}\sum_{x\in D} f(x)$, where $f : [-1,1]^d \to \mathbb{R}$ is a continuous function. Thus, to find out a natural class of linear queries, one only needs to find out a natural class of continuous functions.

The set of continuous functions considered in this paper is the set of smooth queries. We say a function $f : [-1,1]^d \to \mathbb{R}$ is K-smooth, if all the partial derivatives of f up to the Kth order are bounded. We say a linear query q_f is K-smooth if f is a K-smooth function. We believe smooth function is one of the most natural and useful classes of continuous functions; and therefore the set of smooth queries is a natural class of linear queries worth studying. Our aim is to answer *all* smooth queries while preserving differential privacy and accuracy. See also section 4.2 for an explanation of why answering all smooth queries is useful for differentially private machine learning. As a first step towards differential privacy for continuous query, we study in this work the case where the dimension d of the data universe $[-1,1]^d$ is a constant. It turns out that the smooth query problem is very challenging even for $d = O(1)$ which is almost trivial in the discrete case (see below).

We develop two mechanisms for the smooth query. The two mechanisms both achieve accuracy of $O\left(\left(\frac{1}{n}\right)^{\frac{K}{2d+K}}/\epsilon\right)$ for *all* K-smooth queries. If the order of smoothness K is large compared to the dimension d, then the errors of the mechanisms are close to n^{-1}.

To be more concrete, the former mechanism outputs a summary of the database. To obtain an answer of a smooth query, the user runs a public evaluation procedure which contains no information of the database. Outputting the summary has running time $O\left(n^{1+\frac{d}{2d+K}}\right)$, and the evaluation procedure for answering a query runs in time $\tilde{O}(n^{\frac{d+2+\frac{2d}{K}}{2d+K}})$. It can be seen that if K is large compared to d, then the mechanism runs in almost linear time. For our second mechanism which outputs synthetic database, the running time is $O(n^{\frac{3dK+5d}{4d+2K}})$, polynomial in the size of the database. Theoretically, the second mechanism is far less efficient as the first one. This is reasonable since outputting synthetic database is much harder than outputting answers. We then develop practically efficient variants of this mechanism.

As a comparison, we will consider how to apply existing mechanisms (e.g., PMW, BLR) to solve the smooth query problem. As our goal is to answer all smooth queries while preserving privacy and accuracy, and because there are infinitely many K-smooth functions, one has to discretize the set of smooth functions before using any existing mechanism. It is not clear how to efficiently discretize this set of queries. More importantly, we show that even if one can discretize this query set in an optimal way, there is a lower bound for the number of discretized queries of this set. The lower bound is exponential in $1/\alpha$, where α is the desired error of the query answering mechanism; and as a result, the best existing mechanism has significantly larger error than that of the algorithms proposed in this paper. (See Section 3.2.1 and Proposition 18 in Section 4.1 for details.)

The mechanisms proposed in this paper are based on L_∞-approximation and is motivated by Thaler et al. (2012), which considers approximation of k-way conjunctions by low degree polynomials. Our basic idea is to approximate the whole query class by linear combination of a small set of basis functions. The main technical difficulty lies in that in order that the approximation induces an efficient and differentially private mechanism, all the linear coefficients of the basis functions must be small. To guarantee this, we first transform the query function. Then by using even trigonometric polynomials as basis functions we prove a constant upper bound for the linear coefficients. It is worth pointing out that to guarantee accuracy, we must consider L_∞-approximation. It is completely different from L_2-approximation, which is simply Fourier analysis when using trigonometric polynomial as basis. Another difficulty for the first mechanism is how to efficiently compute the linear coefficients. It turns out that the smoothness of the functions allows us to use an efficient numerical method to compute the coefficients to a precision so that the accuracy of the mechanism is not affected significantly.

We also point out that the basis functions described above have some relation to conjunctions. In fact, conjunctions can also be viewed as smooth functions as they are multilinear polynomials. Conjunctions form a small subset of smooth functions. Moreover, the set of conjunctions is also a strict subset of the basis function used in our mechanism. If we restrict to conjunctions, our algorithm essentially reduces to Laplace mechanism (see Section 3.2.2 for details). But different to all existing works dealing with conjunctions, our aim here is to answer all smooth queries while preserving privacy and accuracy.

Finally we conduct experiments on the efficient variant of the mechanism which outputs synthetic database as it may be more useful in practice. Experimental results demonstrate that the algorithm achieves good accuracy and are practically efficient on datasets of various sizes and of a number of attributes.

This work is a small step towards an understanding of smooth queries. Our mechanisms have obvious limitations: The performances decrease exponentially with respect to the data dimension d. Thus the algorithms cannot handle the case in which d is a super constant. The experimental results also demonstrate that our mechanisms work well mostly when d is no more than a few hundred. Studying the general case $d = \omega(1)$ is our future work.

The rest of the paper is organized as follows. Section 2 gives the background and all the definitions. In Section 3, we propose the query-answering mechanism. In Section 4, we give the mechanism which is able to output synthetic database. In Section 5, we develop a practical variant of the second mechanism and conduct experiments to evaluate its performance. Finally, we conclude in Section 6.

2. Preliminaries

Let D be a database containing n data points in the data universe $\mathcal{X}$. In this paper, we consider the case that $\mathcal{X} \subset \mathbb{R}^d$ where d is a constant. Typically, we assume that the data universe $\mathcal{X} = [-1, 1]^d$. Two databases D and D' are called neighbors if $|D| = |D'| = n$ and they differ in exactly one data point. The following is the formal definition of differential privacy.

Definition 1 ((ϵ, δ)-differential privacy) *A randomized mechanism $\mathcal{M}$ whose output lies in some range $\mathcal{S}$ is said to preserve (ϵ, δ)-differential privacy if for all measurable $S \in \mathcal{S}$, for all pairs of neighbor databases D, D', the following holds:*

$$\mathbb{P}(\mathcal{M}(D) \in S) \leq \mathbb{P}(\mathcal{M}(D') \in S) \cdot e^\epsilon + \delta,$$

where the probability is taken over the randomness of the $\mathcal{M}$. If $\mathcal{M}$ preserves $(\epsilon, 0)$-differential privacy, we say $\mathcal{M}$ is ϵ-differentially private.

We consider *linear queries*. Each linear query q_f is specified by a function f which maps the data universe $[-1, 1]^d$ to $\mathbb{R}$. q_f is defined as

$$q_f(D) := \frac{1}{|D|} \sum_{\mathbf{x} \in D} f(\mathbf{x}).$$

Let Q be a set of queries. The accuracy of a mechanism with respect to Q is defined as follows.

Definition 2 ((α, β)-accuracy) *Let Q be a set of queries. A mechanism $\mathcal{M}$ is said to have (α, β)-accuracy for size n databases with respect to Q, if for every database D with $|D| = n$ the following holds*

$$\mathbb{P}(\exists q \in Q, \quad |\mathcal{M}(D, q) - q(D)| \geq \alpha) \leq \beta,$$

where $\mathcal{M}(D, q)$ is the answer to q given by $\mathcal{M}$, and the probability is over the internal randomness of the mechanism $\mathcal{M}$.

We will make use of the Laplace mechanism (Dwork et al., 2006) in our algorithm. Laplace mechanism adds Laplace noise to the output. We denote by $\mathrm{Lap}(\sigma)$ the random variable distributed according to the Laplace distribution with parameter σ, whose density function is $\frac{1}{2\sigma} \exp(-|x|/\sigma)$.

Next, we formally define smooth queries, which is a special class of linear queries. Since each linear query q_f is specified by a function f, a set of queries Q_F can be specified by a set of functions F. Remember that each $f \in F$ maps $[-1, 1]^d$ to $\mathbb{R}$. For any point $\mathbf{x} = (x_1, \ldots, x_d) \in [-1, 1]^d$, if $\mathbf{k} = (k_1, \ldots, k_d)$ is a d-tuple of nonnegative integers, then we define

$$D^{\mathbf{k}} := D_1^{k_1} \cdots D_d^{k_d} := \frac{\partial^{k_1}}{\partial x_1^{k_1}} \cdots \frac{\partial^{k_d}}{\partial x_d^{k_d}}.$$

Let $|\mathbf{k}| := k_1 + \ldots + k_d$. Define the K-norm as

$$\|f\|_K := \sup_{|\mathbf{k}| \leq K} \sup_{\mathbf{x} \in [-1,1]^d} |D^{\mathbf{k}} f(\mathbf{x})|.$$

We will study the set C_B^K which contains all *smooth* functions whose derivatives up to order K have ∞-norm upper bounded by a constant $B > 0$. Formally,

$$C_B^K := \{f : \quad \|f\|_K \leq B\}.$$

The set of queries specified by C_B^K, denoted as $Q_{C_B^K}$, is our focus. Smooth functions have been studied in depth in machine learning (van der Vart and Wellner, 1996; Wahba et al.,

1999; Smola et al., 1998; Wang, 2011). See Section 4.2 for examples of smooth functions and its relation to differentially private machine learning.

In this work we will frequently use *trigonometric polynomials*. For the univariate case, a function $p(\theta)$ is called a trigonometric polynomial of degree m if

$$p(\theta) = a_0 + \sum_{r=1}^{m} (a_r \cos r\theta + b_r \sin r\theta),$$

where a_r, b_r are constants. If $p(\theta)$ is an even function, we say that it is an even trigonometric polynomial, and

$$p(\theta) = a_0 + \sum_{r=1}^{m} a_r \cos r\theta.$$

For the multivariate case, if

$$p(\theta_1, \ldots, \theta_d) = \sum_{\mathbf{r}=(r_1,\ldots,r_d)} a_{\mathbf{r}} \cos(r_1\theta_1) \ldots \cos(r_d\theta_d),$$

then p is said to be an even trigonometric polynomial (with respect to each variable), and the degree of θ_i is $\max_{i\le d}\{r_i\}$.

3. The First Mechanism: Query Answering

In this section we propose our first mechanism which outputs answers to the queries. Our second mechanism which is able to output synthetic database will be given in the next section.

3.1 Mechanism and Main Results

The following theorem is our first main result. It says that if the query class is specified by smooth functions, then there is an efficient mechanism for query answering which preserves ϵ-differential privacy and good accuracy. The mechanism consists of two parts: One for outputting a summary of the database, the other for answering a query. The two parts are described in Algorithm 1 and Algorithm 2 respectively. The second part of the mechanism contains no private information of the database.

Theorem 3 *Let the query set be*

$$Q_{C_B^K} := \{q_f = \frac{1}{n}\sum_{\mathbf{x}\in D} f(\mathbf{x}) : \quad f \in C_B^K\},$$

where $K \in \mathbb{N}$ and $B > 0$ are constants. Let the data universe be $[-1,1]^d$, where $d \in \mathbb{N}$ is a constant. Then the mechanism $\mathcal{M}$ given in Algorithm 1 and Algorithm 2 satisfies that for any $\epsilon > 0$, the followings hold:

1) The mechanism is ϵ-differentially private.

Algorithm 1 Outputting the summary

Notations: $\mathcal{T}_t^d := \{0, 1, \ldots, t-1\}^d$, $\mathbf{x} := (x_1, \cdots, x_d)$, $\theta_i(\mathbf{x}) := \arccos(x_i)$.
Parameters: Privacy parameters $\epsilon, \delta > 0$, Failure probability $\beta > 0$,
Smoothness order $K \in \mathbb{N}$.
Input: Database $D \in \left([-1,1]^d\right)^n$
Output: A t^d-dimensional vector as the summary $\hat{\mathbf{b}}$.

1: Set $t = \lceil n^{\frac{1}{2d+K}} \rceil$.
2: **for all** $\mathbf{r} = (r_1, \ldots, r_d) \in \mathcal{T}_t^d$ **do**
3: $\quad b_{\mathbf{r}} \leftarrow \frac{1}{n}\sum_{\mathbf{x}\in D} \cos(r_1\theta_1(\mathbf{x})) \ldots \cos(r_d\theta_d(\mathbf{x}))$
4: $\quad \hat{b}_{\mathbf{r}} \leftarrow b_{\mathbf{r}} + \mathrm{Lap}\left(\frac{t^d}{n\epsilon}\right)$
5: **end for**
6: $\hat{\mathbf{b}} \leftarrow (\hat{b}_{\mathbf{r}})_{\|\mathbf{r}\|_\infty \le t-1}$ ($\hat{\mathbf{b}}$ is a t^d dimensional vector)
7: **return:** $\hat{\mathbf{b}}$

Algorithm 2 Answering a query

Input: A query q_f, where $f : [-1,1]^d \to \mathbb{R}$ and $f \in C_B^K$,
Summary $\hat{\mathbf{b}}$ returned by Algorithm 1
Output: Approximate answer to $q_f(D)$.

1: Set $t = \lceil n^{\frac{1}{2d+K}} \rceil$.
2: Let $g_f(\boldsymbol{\theta}) = f(\cos(\theta_1), \ldots, \cos(\theta_d))$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d) \in [-\pi, \pi]^d$.
3: Compute the a trigonometric polynomial approximation $p_t(\boldsymbol{\theta})$ of $g_f(\boldsymbol{\theta})$, where $p_t(\boldsymbol{\theta}) = \sum_{\mathbf{r}=(r_1,\ldots,r_d), \|\mathbf{r}\|_\infty \le t-1} c_{\mathbf{r}} \cos(r_1\theta_1) \ldots \cos(r_d\theta_d)$. (see Section 3.3 for details)
4: $\mathbf{c} \leftarrow (c_{\mathbf{r}})_{\|\mathbf{r}\|_\infty \le t-1}$ ($\mathbf{c}$ is a t^d dimensional vector)
5: **return:** $\mathbf{c} \cdot \hat{\mathbf{b}}$

2) For any $\beta \ge 10 \cdot e^{-\frac{1}{5}(n^{\frac{d}{2d+K}})}$ the mechanism is (α, β)-accurate, where $\alpha = O\left(n^{-\frac{K}{2d+K}}/\epsilon\right)$, and the hidden constant depends only on d, K and B.

3) The running time for $\mathcal{M}$ to output the summary is $O(n^{\frac{3d+K}{2d+K}})$.

4) The running time for $\mathcal{M}$ to answer a query is $O(n^{\frac{d+2+\frac{2d}{K}}{2d+K}} \cdot \mathrm{polylog}(n))$.

The proof of Theorem 3 is essentially based on Theorem 4 given below. The detailed proof of Theorem 3 is given in the Section 3.4.

To have a better idea of how the performances depend on the order of smoothness, let us consider three cases. The first case is $K = 1$, i.e., the query functions only have the first order derivatives. Another extreme case is $K \gg d$, and we assume $d/K = \epsilon_0 \ll 1$. We also consider a case in the middle by assuming $K = 2d$. Table 1 gives simplified upper bounds for the error and running time in these cases. We have the following observations:

1) The accuracy α improves dramatically from roughly $O(n^{-\frac{1}{2d}})$ to nearly $O(n^{-1})$ as K increases. For $K > 2d$, the error is smaller than the sampling error $O(n^{-\frac{1}{2}})$.

Order of smoothness	Accuracy α	Running Time: Outputting summary	Running Time: Answering a query
$K=1$	$O(n^{-\frac{1}{2d+1}})$	$O(n^{\frac{3}{2}})$	$\tilde{O}(n^{\frac{3}{2}+\frac{1}{4d+2}})$
$K=2d$	$O(n^{-\frac{1}{2}})$	$O(n^{\frac{5}{4}})$	$\tilde{O}(n^{\frac{1}{4}+\frac{3/4}{d}})$
$\frac{d}{K}=\epsilon_0 \ll 1$	$O(n^{-(1-2\epsilon_0)})$	$O(n^{1+\epsilon_0})$	$\tilde{O}(n^{\epsilon_0(1+\frac{3}{d})})$

Table 1: Performance vs. Order of Smoothness for Query Answering

2) The running time for outputting the summary does not change too much, because reading through the database requires $\Omega(n)$ time.

3) The running time for answering a query reduces significantly from roughly $O(n^{3/2})$ to nearly $O(n^{\epsilon_0})$ as K getting large. When $K=2d$, it is approximately $n^{1/4}$ if d is not too small.

Conceptually our mechanism is simple. First, by change of variables we have

$$g_f(\theta_1,\ldots,\theta_d)=f(\cos\theta_1,\ldots,\cos\theta_d).$$

It also transforms the data universe from $[-1,1]^d$ to $[-\pi,\pi]^d$. Note that for each variable θ_i, g_f is an even function. To compute the summary, the mechanism just gives noisy answers to queries specified by *even trigonometric monomials* $\cos(r_1\theta_1)\ldots\cos(r_d\theta_d)$. For each trigonometric monomial, the highest degree of any variable is $\max_{1\le i\le d} r_i \le t = O(n^{\frac{1}{2d+K}})$. The summary is an $O(n^{\frac{d}{2d+K}})$-dimensional vector. To answer a query specified by a smooth function f, the mechanism computes a trigonometric polynomial approximation of g_f. The answer to the query q_f is a linear combination of the summary by the coefficients of the approximation trigonometric polynomial.

Our algorithm is an L_∞-approximation based mechanism, which is motivated by Thaler et al. (2012). An approximation based mechanism relies on three conditions:

1) There exists a small set of basis functions such that every query function can be well approximated by a linear combination of them.

2) All the linear coefficients are small.

3) The whole set of the linear coefficients can be computed efficiently.

If these conditions hold, then the mechanism just outputs noisy answers to the set of queries specified by the basis functions as the summary. When answering a query, the mechanism computes the coefficients with which the linear combination of the basis functions approximate the query function. The answer to the query is sidiscretizationmply the inner product of the coefficients and the summary vector.

The following theorem contains the main technical results based on which Theorem 3 holds. It guarantees that by change of variables and using even trigonometric polynomials as the basis functions, the class of smooth functions has all the three properties described above.

Theorem 4 *Let $\gamma>0$. For every $f\in C_B^K$ defined on $[-1,1]^d$, let*

$$g_f(\theta_1,\ldots,\theta_d)=f(\cos\theta_1,\ldots,\cos\theta_d),\qquad \theta_i\in[-\pi,\pi].$$

Then, there is an even trigonometric polynomial p whose degree of each variable is $t(\gamma)=\left(\frac{1}{\gamma}\right)^{1/K}$:

$$p(\theta_1,\ldots,\theta_d)=\sum_{0\le r_1,\ldots,r_d<t(\gamma)} c_{r_1,\ldots,r_d}\cos(r_1\theta_1)\ldots\cos(r_d\theta_d),$$

such that

1) $\|g_f-p\|_\infty\le\gamma$.

2) All the linear coefficients $c_{r_1,\ldots,r_d}$ can be uniformly upper bounded by a constant M independent of $t(\gamma)$ (i.e., M depends only on K, d, and B).

3) The whole set of the linear coefficients can be computed in time $O\left((\frac{1}{\gamma})^{\frac{d+2}{K}+\frac{2d}{K^2}}\cdot\mathrm{polylog}(\frac{1}{\gamma})\right)$.

Theorem 4 is proved in Section 3.3. Based on Theorem 4, the proof of Theorem 3 is mainly the argument for Laplace mechanism together with an optimization of the approximation error γ trading-off with the Laplace noise. (Please see Section 3.4 for more details.)

3.2 Discussions

In this section, we show that our algorithm is more effective than methods based on discretization.

3.2.1 Performance of Existing Mechanisms for Smooth Queries

As mentioned in Introduction, in order to apply existing mechanisms to smooth query problem, one has to conduct discretization, both to the data universe and the set of queries. Here, we show that even if one can discretize this function set and even if the it is implemented in an optimal way, the discretized set is exponentially large, and all existing mechanisms have accuracies significantly worse than that of our algorithm.

Proposition 5 *Let $0<\alpha<1$. Let $C_B^K(\alpha)$ be a subset of C_B^K so that for every $f,g\in C_B^K(\alpha)$, $\|f-g\|_{[-1,1]^d}\ge\alpha$. In other words, $|C_B^K(\alpha)|$ is the packing number of C_B^K. We have*

$$\log\left|C_B^K(\alpha)\right|\ge\Omega\left(\left(\frac{1}{\alpha}\right)^{d/K}\right).$$

If we further define $Q^\alpha_{C_B^K}$ as the corresponding discretization of $Q_{C_B^K}$ with precision α, then

$$\log\left|Q^\alpha_{C_B^K}\right|\ge\Omega\left(\left(\frac{1}{\alpha}\right)^{d/K}\right).$$

Corollary 6 *Suppose the query set $Q_{C_B^K}$ can be discretized in an optimal way. Then the accuracy guarantee of the PMW mechanism is at best $O(n^{-\frac{K}{2(d+K)}}/\epsilon)$, and the running time is $O(n^{\frac{Kd}{2(K+d)}})$ per query.*

Compare to the accuracy of our mechanism $O(n^{-\frac{K}{2d+K}}/\epsilon)$, our algorithm has significantly better accuracy especially when K is relatively large.

3.2.2 Connection with Conjunctions

In a sense, the well studied query class conjunctions defined on $\{0,1\}^d$ can be seen as smooth functions, by simply extending the domain from $\{0,1\}^d$ to $[-1,1]^d$ and extrapolating $x_{i_1} \wedge \ldots \wedge x_{i_J}$ to $x_{i_1} \cdots x_{i_J}$. Clearly $f(\mathbf{x}) := \prod_{j=1}^J x_{i_j}$ is multilinear and smooth.

Note that for multilinear function f induced query, our mechanism will transform it to $g_f(\theta_1, \ldots, \theta_d) := f(\cos\theta_1, \ldots, \cos\theta_d) = \prod_{j=1}^J \cos\theta_{i_j}$. Thus g_f is simply a multilinear trigonometric polynomial and is one of the basis functions used in our algorithm. Recall that the mechanism adds Laplace noise to the basis functions. Therefore, if we focus only on the conjunction queries, our algorithm simply reduces to Laplace mechanism.

3.3 L_∞-approximation of smooth functions: small and efficiently computable coefficients

In this section we prove Theorem 4. That is, for every $f \in C_B^K$ the corresponding g_f can be approximated by a low degree trigonometric polynomial in L_∞-norm. We also require that the linear coefficients of the trigonometric polynomial are all small and can be computed efficiently. These properties are crucial for the differentially private mechanism to be accurate and efficient.

In fact, L_∞-approximation of smooth functions in C_B^K by polynomial (and other basis functions) is an important topic in approximation theory. It is well-known that for every $f \in C_B^K$ there is a low degree polynomial with small approximation error. However, it is not clear whether there is an upper bound for the linear coefficients that is sufficiently good for our purpose. Instead we transform f to g_f and use trigonometric polynomials as the basis functions in the mechanism. Then we are able to give a constant upper bound for the linear coefficients. We also need to compute the coefficients efficiently. But results from approximation theory give the coefficients as complicated integrals. We adopt an algorithm which fully exploits the smoothness of the function and thus can efficiently compute approximations of the coefficients to certain precision so that the errors involved do not affect the accuracy of the differentially private mechanism too much.

Below, Section 3.3.1 describes the classical theory on trigonometric polynomial approximation of smooth functions. Section 3.3.2 shows that the coefficients have a small upper bound and can be efficiently computed. Theorem 4 then follows from these results.

3.3.1 Trigonometric Polynomial Approximation with Generalized Jackson Kernel

This section mainly contains known results of trigonometric polynomial approximation, stated in a way tailored to our problem. For a comprehensive description of univariate

approximation theory, please refer to the excellent book of DeVore and Lorentz (1993) and to Temlyakov (1994) for multivariate approximation theory.

Let g_f be the function obtained from $f \in C_B^K([-1,1]^d)$:

$$g_f(\theta_1, \ldots, \theta_d) = f(\cos\theta_1, \ldots, \cos\theta_d).$$

Note that $g_f \in C_{B'}^K([-\pi,\pi]^d)$ for some constant B' depending only on B, K, d, and g_f is even with respect to each variable. The key tool in trigonometric polynomial approximation of smooth functions is the generalized Jackson kernel.

Definition 7 *Define the generalized Jackson kernel as*

$$J_{t,r}(s) = \frac{1}{\lambda_{t,r}} \left(\frac{\sin(ts/2)}{\sin(s/2)} \right)^{2r},$$

where $\lambda_{t,r}$ is determined by $\int_{-\pi}^{\pi} J_{t,r}(s)\mathrm{d}s = 1$.

$J_{t,r}(s)$ is an even trigonometric polynomial of degree $r(t-1)$. Let $H_{t,r}(s) = J_{t',r}(s)$, where $t' = \lfloor t/r \rfloor + 1$. Then $H_{t,r}$ is an even trigonometric polynomial of degree at most t. We write

$$H_{t,r}(s) = a_0 + \sum_{l=1}^{t} a_l \cos ls. \tag{1}$$

Suppose that g is a univariate function defined on $[-\pi,\pi]$ which satisfies that $g(-\pi) = g(\pi)$. Define the approximation operator $I_{t,K}$ as

$$I_{t,K}(g)(x) = -\int_{-\pi}^{\pi} H_{t,r}(s) \sum_{l=1}^{K+1} (-1)^l \binom{K+1}{l} g(x+ls)\mathrm{d}s, \tag{2}$$

where $r = \lceil \frac{K+3}{2} \rceil$. It is not difficult to see that $I_{t,K}$ maps g to a trigonometric polynomial of degree at most t.

Next suppose that g is a d-variate function defined on $[-\pi,\pi]^d$, and is even with respect to each variable. Define an operator $I_{t,K}^d$ as sequential composition of $I_{t,K,1}, \ldots, I_{t,K,d}$, where $I_{t,K,j}$ is the approximation operator given in (2) with respect to the jth variable of g. Thus $I_{t,K}^d(g)$ is a trigonometric polynomial of d-variables and each variable has degree at most t.

Theorem 8 *Suppose that g is a d-variate function defined on $[-\pi,\pi]^d$, and is even with respect to each variable. Let $D_j^{(K)} g$ be the Kth order partial derivative of g respect to the j-th variable. If $\|D_j^{(K)} g\|_\infty \le M$ for some constant M for all $1 \le j \le d$, then there is a constant C such that*

$$\|g - I_{t,K}^d(g)\|_\infty \le \frac{C}{t^{K+1}},$$

where C depends only on M, d and K.

3.3.2 The Linear Coefficients

In this subsection we study the linear coefficients in the trigonometric polynomial $I^d_{t,K}(g_f)$. The previous subsection established that g_f can be approximated by $I^d_{t,K}(g_f)$ for a small t. Here we consider the upper bound and approximate computation of the coefficients. Since $I^d_{t,K}(g_f)(\theta_1,\ldots,\theta_d)$ is even with respect to each variable, we write

$$I^d_{t,K}(g_f)(\theta_1,\ldots,\theta_d) = \sum_{0\le n_1,\ldots,n_d\le t} c_{n_1,\ldots,n_d}\cos(n_1\theta_1)\ldots\cos(n_d\theta_d). \tag{3}$$

Fact 9 *The coefficients $c_{n_1,\ldots,n_d}$ of $I^d_{t,K}(g_f)$ can be written as*

$$c_{n_1,\ldots,n_d} = (-1)^d \sum_{\substack{1\le k_1,\ldots,k_d\le K+1\\ 0\le l_1,\ldots,l_d\le t\\ l_i=k_i\cdot n_i\,\forall i\in[d]}} m_{l_1,k_1,\ldots,l_d,k_d}, \tag{4}$$

where

$$m_{l_1,k_1,\ldots,l_d,k_d} = \prod_{i=1}^d (-1)^{k_i} a_{l_i}\binom{K+1}{k_i}\left(\int_{[-\pi,\pi]^d}\prod_{i=1}^d\cos\left(\frac{l_i}{k_i}\theta_i\right) g_f(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}\right), \tag{5}$$

and a_{l_i} is the linear coefficient of $\cos(l_i s)$ in $H_{t,r}(s)$ as given in (1).

The following lemma shows that the coefficients $c_{n_1,\ldots,n_d}$ of $I^d_{t,K}(g_f)$ can be uniformly upper bounded by a constant independent of t.

Lemma 10 *There exists a constant M which depends only on K, B, d but independent of t, such that for every $f\in C^K_B$, all the linear coefficients $c_{n_1,\ldots,n_d}$ of $I^d_{t,K}(g_f)$ satisfy*

$$|c_{n_1,\ldots,n_d}|\le M.$$

For clarity, we postpone the proof in Section 3.3.3.

Now we consider the computation of the coefficients $c_{n_1,\ldots,n_d}$ of $I^d_{t,K}(g_f)$. Note that each coefficient involves d-dimensional integrations of smooth functions, so we have to numerically compute approximations of them. For function class C^K_B defined on $[-1,1]^d$, traditional numerical integration methods run in time $O((\frac{1}{\tau})^{d/K})$ in order that the error is less than τ. Here we adopt the sparse grids algorithm due to Gerstner and Griebel (1998) which fully exploits the smoothness of the integrand. By choosing a particular quadrature rule as the algorithm's subroutine, we are able to prove that the running time of the sparse grids is bounded by $O((\frac{1}{\tau})^{2/K})$. The sparse grids algorithm, the theorem giving the bound for the running time and its proof are all given in the follow section 3.3.4 and 3.3.5. Based on these results, we establish the running time for computing the approximate coefficients of the trigonometric polynomial, which is stated in the following Lemma.

Lemma 11 *Let $\hat{c}_{n_1,\ldots,n_d}$ be an approximation of the coefficient $c_{n_1,\ldots,n_d}$ of $I^d_{t,K}(g_f)$ obtained by approximately computing the integral in* (5) *with a version of the sparse grids algorithm (Gerstner and Griebel, 1998) (given in the section 3.3.4). Let*

$$\hat{I}^d_{t,K}(g_f)(\theta_1,\ldots,\theta_d) = \sum_{0\le n_1,\ldots,n_d\le t}\hat{c}_{n_1,\ldots,n_d}\cos(n_1\theta_1)\ldots\cos(n_d\theta_d).$$

Then for every $f\in C^K_B$, in order that

$$\|\hat{I}^d_{t,K}(g_f) - I^d_{t,K}(g_f)\|_\infty \le O\left(t^{-K}\right),$$

it suffices that the computation of all the coefficients $\hat{c}_{n_1,\ldots,n_d}$ runs in time $O\left(t^{(1+\frac{2}{K})d+2}\cdot\mathrm{polylog}(t)\right)$. In addition, $\max_{n_1,\ldots,n_d}|\hat{c}_{n_1,\ldots,n_d} - c_{n_1,\ldots,n_d}| = o(1)$ as $t\to\infty$.

The proof of Lemma 11 is given in the Section 3.3.5. Theorem 4 then follows easily from Lemma 10 and Lemma 11.

Proof of Theorem 4

Setting $t = t(\gamma) = \left(\frac{1}{\gamma}\right)^{1/K}$. Let $p = \hat{I}^d_{m,K}(g_f)$. Combining Lemma 10 and Lemma 11, and note that the coefficients $\hat{c}_{n_1,\ldots,n_d}$ are upper bounded by a constant, the theorem follows. ■

3.3.3 Proof of Lemma 10

We first give a simple lemma.

Lemma 12 *Let*

$$H_{t,r}(s) = \sum_{l=0}^t a_l\cos ls. \tag{6}$$

Then for all $l = 0,1,\ldots,t$

$$|a_l|\le 1/\pi.$$

Proof For any $l\in\{0,1,\ldots,t\}$, multiplying $\cos ls$ on both sides of (6) and integrating from $-\pi$ to π, we obtain that for some $\xi\in[-\pi,\pi]$,

$$a_l = \frac{1}{\pi}\int_{-\pi}^{\pi} H_{t,r}(s)\cos ls\mathrm{d}s = \frac{\cos l\xi}{\pi}\int_{-\pi}^{\pi} H_{t,r}(s)\mathrm{d}s = \frac{\cos l\xi}{\pi}.$$

where in the last equation we use the identity

$$\int_{-\pi}^{\pi} H_{t,r}(s)\mathrm{d}s = 1.$$

This completes the proof. ■

Proof of Lemma 10

We first bound $m_{l_1,k_1,\ldots,l_d,k_d}$. Recall that (see also (5) in Fact 9)

$$m_{l_1,k_1,\ldots,l_d,k_d} = \prod_{i=1}^d (-1)^{k_i} a_{l_i}\binom{K+1}{k_i}\left(\int_{[-\pi,\pi]^d}\prod_{i=1}^d\cos\left(\frac{l_i}{k_i}\theta_i\right) g_f(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}\right).$$

It is not difficult to see that $|m_{l_1,k_1,\ldots,l_d,k_d}|$ can be upper bounded by a constant depending only on d, K and B, but independent of t. This is because that the previous lemma shows $|a_{l_i}| \leq \frac{1}{\pi}$ and g_f is upper bounded by a constant.

Now consider $c_{n_1,\ldots,n_d}$. Recall that

$$c_{n_1,\ldots,n_d} = (-1)^d \sum_{\substack{1\leq k_1,\ldots,k_d\leq K+1 \\ 0\leq l_1,\ldots,l_d\leq t \\ l_i=k_i\cdot n_i \forall i\in[d]}} m_{l_1,k_1,\ldots,l_d,k_d}.$$

We need to show that all $|c_{n_1,\ldots,n_d}|$ are upper bounded by a constant independent of t. Note that although each l_i takes $t+1$ values, l_i and k_i must satisfy the constraint $l_i/k_i = n_i$. Since k_i can take at most $K+1$ values, the number of $m_{l_1,k_1,\ldots,l_d,k_d}$ appeared in the summation is at most $(K+1)^d$. Therefore all $c_{n_1,\ldots,n_d}$ are bounded by a constant depending only on d, K and B, and is independent of t. ■

3.3.4 The Sparse Grids Algorithm

In this section we briefly describe the sparse grids numerical integration algorithm due to Gerstner and Griebel. (Please refer to Gerstner and Griebel 1998 for a complete introduction.) We also specify a subroutine used by this algorithm, which is important for proving the running time.

Numerical integration algorithms dicretize the space and use weighted sum to approximate the integration. Traditional methods for the multidimensional case usually discretize each dimension to the same precision level. In contrast, the sparse grids methods, first proposed by Smolyak (1963), discretize each dimension to carefully chosen and possibly different precision levels, and finally combine many such discretization results. When the integrand has bounded mixed derivatives, as in our case that the integrand is in C_B^K, one can use very few grids in most dimension and still achieve high accuracy.

The sparse grids method is based on one dimensional quadrature (i.e., numerical integration). There are many candidates for one dimensional quadrature. In order to prove an upper bound for the running time, we choose the Clenshaw-Curtis rule (Clenshaw and Curtis, 1960) as the subroutine. This also makes the analysis simpler.

Let $h : [-1,1]^d \to \mathbb{R}$ be the integrand. Let $SG(h)$ be the output of the sparse grids algorithm. Let l be the *level* parameter of the algorithm.

Let $\mathbf{k} = (k_1,\ldots,k_d)$ and $\mathbf{j} = (j_1,\ldots,j_d)$ be d-tuples of positive integers. Then $SG(h)$ is given as a combination of weighted sum:

$$SG(h) := \sum_{|\mathbf{k}|\leq l+d-1} \sum_{j_1=1}^{m(k_1)} \cdots \sum_{j_d=1}^{m(k_d)} u_{\mathbf{k},\mathbf{j}} f(\mathbf{x}_{\mathbf{k},\mathbf{j}}). \tag{7}$$

Below we describe $m(k_i)$, $\mathbf{x}_{\mathbf{k},\mathbf{j}}$ and $u_{\mathbf{k},\mathbf{j}}$ respectively.

1) For any $k \in \mathbb{N}$, $m(k) := 2^k$.

2) For each $\mathbf{k} = (k_1,\ldots,k_d)$ and $\mathbf{j} = (j_1,\ldots,j_d)$, define $\mathbf{x}_{\mathbf{k},\mathbf{j}} := (x_{k_1,j_1},\ldots,x_{k_d,j_d})$, and x_{k_i,j_i} is the j_ith zero of the Chebyshev polynomial with degree $m(k_i)$. Denote by $T_{m(k_i)}$ the

Chebyshev polynomial. Its zeros are given by the following formula.

$$x_{k_i,j_i} = \cos\left(\frac{(2j_i-1)\pi}{2m(k_i)}\right), \qquad j_i = 1,2,\ldots,m(k_i). \tag{8}$$

3) Now we define the weights $u_{\mathbf{k},\mathbf{j}}$. First let $w_{k,j}$ be the weight of $x_{k,j}$ in the one-dimensional Clenshaw-Curtis quadrature rule given by

$$\begin{aligned} w_{k,1} &= \frac{1}{(m(k)+1)(m(k)-1)}, \\ w_{k,j} &= \frac{2}{m(k)}\left(1 + 2\sum_{r=1}^{m(k)/2}{}' \frac{1}{1-4r^2}\cos\left(\frac{2\pi(j-1)r}{m(k)}\right)\right), \quad \text{for } 2\leq j\leq m(k), \end{aligned} \tag{9}$$

where $\sum'$ means that the last term of the summation is halved.

Next, for any fixed k and j, define

$$v_{(k+q),j} = \begin{cases} w_{k,j} & \text{if } q = 1, \\ w_{(k+q-1),r} - w_{(k+q-2),s} & \text{if } q > 1, \text{ and } r,s \text{ with } x_{k,j} = x_{(k+q-1),r} = x_{(k+q-2),s}\ , \end{cases}$$

where $x_{k,j}$ is the zero of Chebyshev polynomial defined above.

Finally, the weight $u_{\mathbf{k},\mathbf{j}}$ is given by

$$u_{\mathbf{k},\mathbf{j}} = \sum_{|\mathbf{k}+\mathbf{q}|\leq l+2d-1} v_{(k_1+q_1),j_1}\cdots v_{(k_d+q_d),j_d},$$

where $\mathbf{k} = (k_1,\ldots,k_d)$ and $\mathbf{q} = (q_1,\cdots,q_d)$. This completes the description of the sparse grids algorithm.

3.3.5 Proof of Lemma 11

We first give the result that characterizes the running time of the Gerstner-Griebel sparse grids algorithm in order to achieve a given accuracy.

Lemma 13 *Let $h \in C_B^K([-\pi,\pi]^d)$ for some constants K and B. Let $SG(h)$ be the numerical integration of h using the sparse grids algorithm described in the previous section. Given any desired accuracy parameter $\tau > 0$, the algorithm achieves*

$$\left|\int_{[\pi,\pi]^d} h(\theta)\mathrm{d}\theta - \mathrm{SG}(h)\right| \leq \tau,$$

with running time at most $O\left(\left(\frac{1}{\tau}\right)^{\frac{2}{K}}(\log\frac{1}{\tau})^{3d+\frac{2d}{K}+1}\right)$.

Proof

Let $L = 2^{l+1}-2$, where l is the level parameter of the sparse grids algorithm. l and L will be determined by the desired accuracy τ later. In fact, L is the maximum number of

grid points of one dimension. By (7) it is easy to see that the total number of grid points, denoted by N_l^d, is given by

$$\begin{aligned} N_l^d &= \sum_{|\mathbf{k}|\le l+d-1} m(k_1)\cdots m(k_d) \\ &= O(l^{d-1}L) \\ &= O(L(\log_2 L)^{d-1}). \end{aligned} \tag{10}$$

In Gerstner and Griebel (1998), it is shown that the approximation error τ can be bounded by the maximal number of grid points per dimension as follows.

$$\tau = O(L^{-K}(\log L)^{(K+1)(d-1)}). \tag{11}$$

Next, let us consider the computational cost per grid point. Since we assume that $h(\mathbf{x})$ can be computed in unit time, and the zeros of Chebyshev polynomials can be computed according to (8), then computing the weights $u_{\mathbf{kj}}$ dominates the running time. Fix $k \in \mathbb{N}$, consider $w_{k,j}$, $1 \le j \le m(k)$. From (9), it is not difficult to see that the set of $w_{k,j}$ can be computed by Fast Fourier Transform (FFT). Therefore the computation cost is $O(m(k)\log m(k))$. Some calculations yield that for a fixed $\mathbf{k},\mathbf{j}$, the computational cost for $u_{\mathbf{kj}}$ is $O(dL\log L)$. Combining this with (10) and (11) the lemma follows.

■

Next we turn to prove Lemma 11. First, we need the following famous result.

Lemma 14 *Let m be a positive integer, let $\sigma(m)$ denotes the number of divisors of m, then for large t*

$$\sum_{m=1}^{t}\sigma(m) = t\ln t + (2c-1)t + O(t^{1/2}),$$

where c is Euler's constant.

To analyze the running time, we also need a result about the normalizing constant of the generalized Jackson kernel (Vyazovskaya and Pupashenko, 2006).

Lemma 15 (Vyazovskaya and Pupashenko 2006) *Let*

$$J_{t,r} = \frac{1}{\lambda_{t,r}}\left(\frac{\sin(ts/2)}{\sin(s/2)}\right)^{2r},$$

be the generalized Jackson kernel as given in Definition 4.1, and the normalizing constant $\lambda_{t,r}$ is determined by

$$\int_{-\pi}^{\pi} J_{t,r}(s)\mathrm{d}s = 1.$$

Then the following identity of the normalizing constant $\lambda_{t,r}$ holds

$$\lambda_{t,r} = 2\pi\sum_{k=0}^{[r-r/t]}(-1)^k\binom{2r}{k}\binom{r(t+1)-tk-1}{r(t-1)-tk}. \tag{12}$$

Now we are ready to prove Lemma 11.

Proof of Lemma 11

Assume that the error induced by the sparse grids algorithm is at most τ per integration. That is, for every $\mathbf{k}=(k_1,\ldots,k_d)$, $\mathbf{l}=(l_1,\ldots,l_d)$

$$\left|\int_{[-\pi,\pi]^d}\prod_{i=1}^{d}\cos\left(\frac{l_i}{k_i}\theta_i\right)g(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} - SG\left(\prod_{i=1}^{d}\cos\left(\frac{l_i}{k_i}\theta_i\right)g(\boldsymbol{\theta})\right)\right| \le \tau.$$

Then

$$\sup_{n_1,\ldots,n_d}|c_{n_1,\ldots,n_d}-\hat{c}_{n_1,\ldots,n_d}| \le \sup_{n_1,\ldots,n_d}\left|\sum_{l_i/k_i=n_i}\prod_{i=1}^{d}(-1)^{k_i}\binom{K+1}{k_i}a_{l_i}\right|\cdot\tau.$$

By Lemma 12, $|a_{l_i}|\le\frac{1}{\pi}$. We obtain that

$$\sup_{n_1,\ldots,n_d}|c_{n_1,\ldots,n_d}-\hat{c}_{n_1,\ldots,n_d}| \le M\cdot\tau, \tag{13}$$

for some constant M independent of t.

Similarly, we have

$$\left\|I_{t,K}^d(g)-\hat{I}_{t,K}^d(g)\right\|_\infty \le O(t^d\tau). \tag{14}$$

Since in the statement of the lemma the desired approximation error is $O(t^{-K})$, we have

$$\tau = t^{-(K+d)}. \tag{15}$$

It is also clear that

$$\max_{n_1,\ldots,n_d}|\hat{c}_{n_1,\ldots,n_d}-c_{n_1,\ldots,n_d}| = o(1), \quad \text{as } t\to\infty.$$

Now let us consider the computation cost. Recall that the kernel $H_{t,r}$ is an even trigonometric of degree at most t:

$$H_{t,r}(s) = a_0 + \sum_{l=1}^{t}a_l\cos ls, \tag{16}$$

where $H_{t,r}(s)=J_{t',r}(s)$ and $J_{t',r}$ is the generalized Jackson kernel given in Definition 4.1. First we need to compute the value of the linear coefficient a_l of $H_{t,r}$. By Lemma 15, one can compute the linear coefficients a_l by solving a system of $t+1$ linear equations. That is, we choose arbitrary $t+1$ points in $[-\pi,\pi]$ and solve (16), since we can compute the value of $H_{t,r}(s)$ directly based on the value of $\lambda_{t,r}$. Clearly, the running time is $O(t^3)$.

Having a_{l_i}, let us consider the computational cost for calculating $\hat{c}_{n_1,\ldots,n_d}$. According to Lemma 13, the running time for the sparse grids algorithm to compute one integration is

$$O\left((\frac{1}{\tau})^{\frac{2}{K}}(\log(1/\tau))^{3d+\frac{2d}{K}+1}\right) = O\left(t^{\frac{2(K+d)}{K}}\mathrm{polylog}(t)\right).$$

Since we only need to compute the integration when $l_i|k_i$ for all $i \in [d]$, by Lemma 14 the number of integrations to compute is at most

$$(K+1+\sigma(1)+\ldots+\sigma(t))^d = O\left((t\log t)^d\right).$$

Thus the total time cost for all numerical integration is $O\left(t^{(1+\frac{2}{K})d+2}\text{polylog}(t)\right)$. Since

$$(1+\frac{2}{K})d+2 \geq 3,$$

the computation time for obtaining the coefficients a_l in $H_{t,r}$ is dominated by the running time of the sparse grids algorithm. It is also easy to see that all other computation costs are dominated by that of the numerical integration. The lemma follows. ■

3.4 Proof of Theorem 3

Here we provide the full proof of our first main theorem (Theorem 3).

Proof of Theorem 3

We prove the four results separately.

3.4.1 Differential Privacy

The summary is a t^d-dimensional vector with sensitivity $\frac{t^d}{n}$. By the standard argument for Laplace mechanism, adding t^d i.i.d. Laplace noise $\text{Lap}(\frac{t^d}{n\epsilon})$ preserves ϵ-differential privacy.

3.4.2 Accuracy

The error of the answer to each query consists of two parts: the approximation error and the noise error. Setting the approximation error γ in Theorem 4 as $\gamma = n^{-\frac{K}{2d+K}}$. Then the degree of each variable in $g(\boldsymbol{\theta})$ is

$$t(\gamma) = \left(\frac{1}{\gamma}\right)^{1/K} = n^{\frac{1}{2d+K}},$$

which is the same as t given in Algorithm 1. Now consider the error induced by the Laplace noise. The noise error is simply the inner product of the t^d linear coefficients $c_{l_1,\ldots,l_d}$ and t^d i.i.d. $\text{Lap}(\frac{t^d}{n\epsilon})$. Since the coefficients are uniformly bounded by a constant, the noise error is bounded by the sum of t^d independent and exponentially distributed random variables (i.e., $|\text{Lap}(\frac{t^d}{n\epsilon})|$). The following lemma gives it an upper bound.

Lemma 16 *Let $X_1,\ldots,X_N$ be i.i.d. random variables with p.d.f. $\mathbb{P}(X_i = x) = \frac{1}{\sigma}e^{-x/\sigma}$ for $x \geq 0$. Then*

$$\mathbb{P}(\sum_{i=1}^{N} X_i \geq 2N\sigma) \leq 10 \cdot e^{-\frac{N}{5}}.$$

Proof Let $Y = \sum_{i=1}^{N} X_i$. It is well-known that Y satisfies the gamma distribution, and for $\forall u > 0$

$$\mathbb{P}(Y \geq u) \leq e^{-\frac{u}{\sigma}} \sum_{n=0}^{N-1} \frac{1}{n!}\left(\frac{u}{\sigma}\right)^n.$$

Thus

$$\mathbb{P}(Y \geq 2N\sigma) \leq e^{-2N} \sum_{n=0}^{N-1} \frac{1}{n!}(2N)^n.$$

Note that for $n < N$

$$\frac{\frac{1}{n!}(2N)^n}{e^{2N}} \leq \frac{\frac{1}{N!}(2N)^N}{\frac{1}{(2N)!}(2N)^{2N}} \leq \prod_{n=1}^{N-1}(1-\frac{n}{2N}) \leq e^{-\frac{N-1}{4}}.$$

Thus

$$\mathbb{P}(Y \geq 2N\sigma) \leq e^{-2N}\left(e^{2N}Ne^{-\frac{N-1}{4}}\right) \leq 10 \cdot e^{-\frac{N}{5}}.$$

■

Part 2) of Theorem 3 then follows from Lemma 16.

3.4.3 Running Time for Outputing Summary

This is straightforward since the summary is a t^d-dimensional vector and for each item the running time is $O(n)$.

3.4.4 Running Time for Answering a Query

According to our setting of t, it is easy to check that the error induced by Laplace noise and that of approximation have the same order. Then by the third part of Theorem 4 we have the running time for computing the coefficients of the trigonometric polynomial is $O\left(n^{\frac{d+2+\frac{2d}{K}}{2d+K}} \cdot \text{polylog}(n)\right)$. The result follows since computing the inner product has running time $O(n^{\frac{d}{2d+K}})$, which is much less than computing the coefficients. ■

4. An Improved Mechanism: Output Synthetic Database

Although the mechanism given in the previous section achieves good accuracy for smooth queries and is efficient, it has a disadvantage: The mechanism can only output answers to given queries, and therefore is not convenient for most machine learning tasks which involve optimizations.

In this section we propose an improved mechanism. The mechanism has the advantage that it is able to output a synthetic database. From a practical point of view this is very appealing, because users can simply use this differentially private database to learn whatever they want. Of course, utility will be guaranteed only for restricted types of tasks

as described below. Also, the price for outputting synthetic database is that theoretically this mechanism is less efficient than the previous one, although it runs in polynomial time. Please see Section 5.2 for practically efficient variations.

4.1 The Mechanism and Theoretical Results

The following theorem is our second main result. It says that if the query class is specified by smooth functions, then there is a polynomial time mechanism which preserves ϵ-differential privacy and achieves good accuracy. The output of the mechanism is a synthetic dataset. A formal description of the mechanism is given in Algorithm 3.

Theorem 17 *Let the query set be*

$$Q_{C_B^K} := \{q_f(D) = \frac{1}{n}\sum_{\mathbf{x}\in D} f(\mathbf{x}) : \quad f \in C_B^K\},$$

where $K \in \mathbb{N}$ and $B > 0$ are constants. Let the data universe be $[-1,1]^d$, where d is a constant. Then the mechanism described in Algorithm 3 satisfies that for any $\epsilon > 0$, the followings hold:

1) *The mechanism preserves ϵ-differential privacy.*

2) *There is an absolute constant c such that for every $\beta \geq c \cdot e^{-n^{\frac{1}{2d+K}}}$ the mechanism is (α, β)-accurate, where $\alpha = O(n^{-\frac{K}{2d+K}}/\epsilon)$, and the hidden constant depends only on d, K and B.*

3) *The running time of the mechanism is $O(n^{\frac{3dK+5d}{4d+2K}})$. (This is dominated by solving the linear programming problem in step 20 of the algorithm.)*

4) *The size of the output synthetic database is $O(n^{1+\frac{K+1}{2d+K}})$.*

The proof of Theorem 17 is given in the Section 4.3. Note that the accuracy of this new mechanism is of the same order as that of our first mechanism.

In Table 2, we illustrate the results with typical parameters as we did in the previous section. From Table 2 we can see, as before, the same accuracy improvement as K/d increases. On the other hand, the running time of the mechanism increases if one wants better accuracy for highly smooth queries. Finally, the size of the output synthetic database also increases in order to have better accuracy: roughly, $O(n^{-1})$ accuracy requires an $O(n^2)$-size synthetic database.

Now we explain the mechanism in detail. Part of Algorithm 3 is the same as Algorithm 1. In particular, Algorithm 3 still employs L_∞ approximation with trigonometric polynomials as basis for transformed smooth functions

$$g_f(\theta_1, \ldots, \theta_d) = f(\cos\theta_1, \ldots, \cos\theta_d).$$

Here we view the trigonometric polynomial functions as a set of basis queries. As in Algorithm 1, we compute the answers to the basis queries and add Laplace noise. So far, if we

Algorithm 3 Private Synthetic DB for Smooth Queries

Notations: $\mathcal{T}_t^d := \{0, 1, \ldots, t-1\}^d$, $\mathbf{x} := (x_1, \cdots, x_d)$, $\theta_i(\mathbf{x}) := \arccos(x_i)$, $a_k := \frac{2k+1-N}{N}$, $\mathcal{A} := \{a_k | k = 0, 1, \ldots, N-1\}$, $\mathcal{L} := \{\frac{i}{L} | i = -L, -L+1, \ldots, L-1, L\}$.
Parameters: Privacy parameters $\epsilon, \delta > 0$, Failure probability $\beta > 0$, Smoothness order $K \in \mathbb{N}$.
Input: Database $D \in \left([-1,1]^d\right)^n$
Output: Synthetic database $\tilde{D} \in \left([-1,1]^d\right)^m$

1: Set $t = \lceil n^{\frac{1}{2d+K}} \rceil$, $N = \lceil n^{\frac{K}{2d+K}} \rceil$, $m = \lceil n^{1+\frac{K+1}{2d+K}} \rceil$, $L = \lceil n^{\frac{d+K}{2d+K}} \rceil$.
2: Initialize: $\underline{D} \leftarrow \emptyset$, $\tilde{D} \leftarrow \emptyset$, $\mathbf{u} \leftarrow \mathbf{0}_{N^d}$
3: **for all** $\mathbf{x} = (x_1, \ldots, x_d) \in D$ **do**
4: $\quad \underline{x_i} \leftarrow \arg\min_{a \in \mathcal{A}} |x_i - a|$, $i = 1, \ldots, d$
5: $\quad$ Add $\underline{\mathbf{x}} = (\underline{x_1}, \ldots, \underline{x_d})$ to $\underline{D}$
6: **end for**
7: **for all** $\mathbf{r} = (r_1, \ldots, r_d) \in \mathcal{T}_t^d$ **do**
8: $\quad b_{\mathbf{r}} \leftarrow \frac{1}{n}\sum_{\underline{\mathbf{x}} \in \underline{D}} \cos(r_1\theta_1(\underline{\mathbf{x}})) \ldots \cos(r_d\theta_d(\underline{\mathbf{x}}))$
9: $\quad \hat{b}_{\mathbf{r}} \leftarrow b_{\mathbf{r}} + \mathrm{Lap}\left(\frac{t^d}{n\epsilon}\right)$
10: $\quad \underline{\hat{b}_{\mathbf{r}}} \leftarrow \arg\min_{l \in \mathcal{L}} |\hat{b}_{\mathbf{r}} - l|$
11: **end for**
12: **for all** $\mathbf{k} = (k_1, \ldots, k_d) \in \mathcal{T}_N^d$ **do**
13: $\quad$ **for all** $\mathbf{r} = (r_1, \ldots, r_d) \in \mathcal{T}_t^d$ **do**
14: $\quad\quad W_{\mathbf{rk}} \leftarrow \cos(r_1 \arccos(a_{k_1})) \ldots \cos(r_d \arccos(a_{k_d}))$
15: $\quad\quad \underline{W_{\mathbf{rk}}} \leftarrow \arg\min_{l \in \mathcal{L}} |W_{\mathbf{rk}} - l|$
16: $\quad$ **end for**
17: **end for**
18: $\underline{\hat{\mathbf{b}}} \leftarrow (\underline{\hat{b}_{\mathbf{r}}})_{\|\mathbf{r}\|_\infty \leq t-1}$ ($\underline{\hat{\mathbf{b}}}$ is a t^d dimensional vector)
19: $\underline{\mathbf{W}} \leftarrow (\underline{W_{\mathbf{rk}}})_{\|\mathbf{r}\|_\infty \leq t-1, \|\mathbf{k}\|_\infty \leq N-1}$ (a $t^d \times N^d$ matrix)
20: Solve the following LP problem: $\min_{\mathbf{u}} \|\underline{\mathbf{W}}\mathbf{u} - \underline{\hat{\mathbf{b}}}\|_1$, subject to $\mathbf{u} \succeq 0$, $\|\mathbf{u}\|_1 = 1$. Obtain the optimal solution $\mathbf{u}^*$.
21: **repeat**
22: $\quad$ Sample $\mathbf{y}$ according to distribution $\mathbf{u}^*$
23: $\quad$ Add $\mathbf{y}$ to $\tilde{D}$
24: **until** $|\tilde{D}| = m$
25: **return:** $\tilde{D}$

ignore the discretization steps (step 3-6), Algorithm 3 is essentially the same as Algorithm 1.

Recall that the next step in our first mechanism is that, for any given smooth query function, we compute the linear coefficients with which the combination of the trigonometric polynomial basis functions is a good approximation of it. The key idea (and the main advantage) of Algorithm 3 is that we do not even need to know the linear coefficients. We merely need to know that there *exist* such coefficients which leads to a good approximation of the smooth function. Now, the goal of the algorithm is to generate a synthetic dataset so that if we evaluate all the basis queries on this synthetic database, all the answers will

Order of smoothness	Accuracy α	Running time	Size of synthetic DB
$K=1$	$O(n^{-\frac{1}{2d+1}})$	$O(n^2)$	$O(n^{1+\frac{2}{2d+1}})$
$K=2d$	$O(n^{-\frac{1}{2}})$	$O(n^{\frac{3}{4}d+\frac{5}{8}})$	$O(n^{\frac{3}{2}+\frac{1}{4d}})$
$\frac{d}{K}=\epsilon_0 \ll 1$	$O(n^{-(1-2\epsilon_0)})$	$O(n^{d(\frac{3}{2}-\frac{\epsilon_0}{2})})$	$O(n^{2-\frac{\epsilon_0}{2}})$

Table 2: Performance vs. Order of Smoothness for Outputing Synthetic Database

be close to the noisy answers obtained from the original dataset. The key observation is that if we have such a synthetic dataset, then the evaluation of any smooth query on this synthetic dataset is an answer both differentially private and accurate. To generate such a dataset, we first learn a probability distribution over $[-1,1]^d$ so that the answers of the basis queries with respect to this distribution are close to the noisy answers. Observe that such a distribution must exist, because the uniform distribution over the original dataset satisfies this requirement. However, learning a continuous distribution is computationally intractable. So we discretize the domain (as well as the original data (step 4)) and consider distributions over the discretized data universe. Because the queries are smooth, the error involved by discretization can be controlled. Learning the distribution can be formulated as a linear programming problem (step 20). Note that in the LP problem we minimize l_1 error instead of l_∞ error because it results in slightly better accuracy. Finally, we randomly draw sufficiently large number of data from this probability distribution, and these data form the output synthetic database.

The running time of the mechanism is dominated by the linear programming step. It is known that the worst-case time complexity of the interior point method is upper bounded in terms of the number of variables, number of constraints, and the number of bits to encode the problem. It is easy to see that there are only poly(n) variables and constraints. To control the number of bits, we round each number in the linear programming problem at a certain precision level (step 10 and 15). Because all the numbers after rounding are bounded uniformly by a constant, the total number of bits to encode the problem is not too large.

We can also analyze the performance of BLR for the smooth query problem and compare to our mechanism proposed in this section as both of them output synthetic database. The analysis is almost identical to that for PMW in Section 3.2.1. Even if one can discretize the query set $Q_{C_B^K}$ in an optimal way, the error of BLR is still considerably larger than that of our algorithm, as described in the following Proposition.

Proposition 18 *Suppose the query set $Q_{C_B^K}$ in an optimal way. The accuracy guarantee of the BLR is at best $O\left(n^{-\frac{K}{d+3K}}\right)$; and the running time is super-exponential in n.*

4.2 Examples of Smooth Queries and Application to Learning

Almost all widely used continuous functions are smooth up to a certain order. Here we list some simple examples. The smoothness of these functions are either obvious or are well known. (See Section 4.4 for proofs.)

1) Linear functions: $f(\mathbf{x}) = \mathbf{w}^T x$ is infinitely smooth if $\|\mathbf{w}\|$ is bounded.

2) Gaussian kernel functions: $f(\mathbf{x}) = \exp(\frac{\|\mathbf{x}-\mathbf{x}_0\|^2}{2\sigma^2})$ (for some $\mathbf{x}_0 \in \mathbb{R}^d$) is in $C_1^{\sigma^2}$, i.e., its derivatives up to order σ^2 is bounded by 1.

3) Logistic function: $f(\mathbf{x}) = \frac{1}{1+e^{-x/\sigma}}$ (for $\sigma > 2$) is K-smooth for any K such that $\max_{k\le K} B_k \le 1$, where B_k is the kth Bernoulli number.

We also point out that many loss functions used in machine learning, composed with smooth functions, are still smooth. Here we just give one example. Suppose the data record in the database are of the form $(\mathbf{x}, y)$. Then the square loss for a smooth regression function f defined as

$$l(f;\mathbf{x},y) = (y - f(x))^2,$$

is smooth.

Now let us discuss applications of our mechanisms to differentially private machine learning. Suppose people want to learn a regression function using the database, where for each data record the first $d-1$ features are independent variables and the dth feature is the dependent variable. In fact, there are excellent algorithms that can do this and guarantee differential privacy. However, consider the following setting: There are many users each wants to learn a regressor and each uses a different type of smooth functions (e.g., linear, polynomial, Ridge, kernel, etc.). Suppose there are totally M users. If we want to guarantee ϵ-differential privacy, we have to require each learner achieve ϵ/M-differential privacy; or equivalently an M-times accuracy decrease in accuracy. As the major goal of differential privacy is to safely release the data and allows everyone to use it, the number of learners M can be very large. Thus a differentially private regression algorithm is not sufficient for this aim. Recently, Ullman (2015) applied PMW to answering multiple convex optimization problems. Using their method, the accuracy decreases only $\log M$-times for M learning problems. However, their algorithm only works for convex loss of linear learning models, while ours work for all smooth functions.

On the other hand, our mechanism can achieve this goal easily: just run the mechanism and output the synthetic database. It is clear that the mechanism guarantees ϵ-differential privacy no matter how many learners use it. The mechanism also guarantees accuracy to *all* learners who use smooth regression functions and the least square criterion, because the accuracy provided by our mechanism hold for *all* smooth functions simultaneously. Therefore, there is no accuracy decrease or privacy loss increase as the number of users grow.

4.3 Proof of Theorem 17

In this section we prove Theorem 17.

Proof of Theorem 17

We first define some notations repeatedly used in this proof.Let the input database be $D = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(n)}\}$. Let the discretized dataset be (please see step 5 in Algorithm 3) $\underline{D} = \{\underline{\mathbf{x}}^{(1)}, \underline{\mathbf{x}}^{(2)}, \cdots, \underline{\mathbf{x}}^{(n)}\}$. Also let the output synthetic dataset be $\tilde{D} = \{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \cdots, \mathbf{y}^{(m)}\}$.

Let

$$\mathbf{b} = (b_{\mathbf{r}})_{\|\mathbf{r}\|_\infty \le t-1}.$$

be a t^d dimensional vector, where $b_{\mathbf{r}}$ is defined in step 8 of the Algorithm 3. Similarly, Let

$$\hat{\mathbf{b}} = (\hat{b}_{\mathbf{r}})_{\|\mathbf{r}\|_\infty \le t-1}$$

and

$$\mathbf{W} = (W_{\mathbf{rk}})_{\|\mathbf{r}\|_\infty \le t-1, \|\mathbf{k}\|_\infty \le N-1},$$

where $\hat{b}_{\mathbf{r}}$ and $W_{\mathbf{rk}}$ are defined as in step 9 and 14 of the algorithm respectively. Let $\mathbf{\Delta} = \hat{\mathbf{b}} - \mathbf{b}$ be the t^d dimensional Laplace noise. Finally let

$$\tilde{\mathbf{b}} = (\tilde{b}_{\mathbf{r}})_{\|\mathbf{r}\|_\infty \le t-1},$$

where

$$\tilde{b}_{\mathbf{r}} = \frac{1}{m} \sum_{\mathbf{y} \in \tilde{D}} \cos(r_1 \theta_1(\mathbf{y})) \dots \cos(r_d \theta_d(\mathbf{y})).$$

(Recall that $\theta_i(\mathbf{y}) = \arccos(y_i)$. Please see also the Notations in Algorithm 3.)

Now we prove the four results in the theorem one by one.

4.3.1 Differential Privacy

That the mechanism preserves ϵ-differential privacy is straightforward. Note that the output synthetic database $\tilde{D}$ contains no private information other than that obtained from $\hat{\mathbf{b}}$. So we only need to show that $\hat{\mathbf{b}}$ is differentially private. But this is immediate from the privacy of Laplace mechanism.

4.3.2 Accuracy

Let $\boldsymbol{\theta} = (\theta_1, \dots, \theta_d)$. For any $f(\mathbf{x}) \in C_B^K$, where $\mathbf{x} \in [-1,1]^d$, let

$$g_f(\boldsymbol{\theta}) := f(\cos\theta_1, \dots, \cos\theta_d).$$

By Theorem 4, we know, there's an even trignometric polynomial:

$$h_f^{M,t}(\boldsymbol{\theta}) := \sum_{\mathbf{r}=(r_1,..,r_d), \|\mathbf{r}\|_\infty \le t-1} c_{\mathbf{r}}^* \cos(r_1\theta_1) \dots \cos(r_d\theta_d) \tag{17}$$

satisfy: (Denote $\mathbf{c}^* = (c_{\mathbf{r}}^*)_{\|\mathbf{r}\|_\infty \le t-1}$)

$$\|g_f - h_f^{M,t}\|_\infty \le O(\frac{1}{t^{K+1}}) \tag{18}$$

$$\|\mathbf{c}^*\|_\infty \le M \tag{19}$$

Where constant M only dependents on K, d, B.

Recall that $\boldsymbol{\theta}(\mathbf{x}) := (\arccos x_1, \dots, \arccos x_d)$. Now we are ready to decompose the error of the mechanism into several terms:

$$\begin{aligned}
&\left|q_f(\tilde{D}) - q_f(D)\right| = \left|\frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} f(\mathbf{y}) - \frac{1}{n}\sum_{\mathbf{x}\in D} f(\mathbf{x})\right| \\
\le &\left|\frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} f(\mathbf{y}) - \frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} h_f^{M,t}(\boldsymbol{\theta}(\mathbf{y}))\right| + \left|\frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} h_f^{M,t}(\boldsymbol{\theta}(\mathbf{y})) - \frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} h_f^{M,t}(\boldsymbol{\theta}(\underline{\mathbf{x}}))\right| \\
&+ \left|\frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} h_f^{M,t}(\boldsymbol{\theta}(\underline{\mathbf{x}})) - \frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} f(\underline{\mathbf{x}})\right| + \left|\frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} f(\underline{\mathbf{x}}) - \frac{1}{n}\sum_{\mathbf{x}\in D} f(\mathbf{x})\right|.
\end{aligned} \tag{20}$$

We further decompose the second term in the last row of the above inequality. We have

$$\begin{aligned}
&\left|\frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} h_f^{M,t}(\boldsymbol{\theta}(\mathbf{y})) - \frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} h_f^{M,t}(\boldsymbol{\theta}(\underline{\mathbf{x}}))\right| \\
= &\left|\mathbf{c}^* \cdot (\tilde{\mathbf{b}} - \mathbf{b})\right| \le \left(\|\tilde{\mathbf{b}} - \hat{\mathbf{b}}\|_1 + \|\mathbf{\Delta}\|_1\right)\|\mathbf{c}^*\|_\infty \\
\le &\left(\|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_1 + \|\mathbf{W}\mathbf{u}^* - \underline{\mathbf{W}}\mathbf{u}^*\|_1 + \|\underline{\mathbf{W}}\mathbf{u}^* - \underline{\hat{\mathbf{b}}}\|_1 + \|\underline{\hat{\mathbf{b}}} - \hat{\mathbf{b}}\|_1 + \|\mathbf{\Delta}\|_1\right)\|\mathbf{c}^*\|_\infty \\
\le &\Big(\|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_1 + \|(\mathbf{W} - \underline{\mathbf{W}})\mathbf{u}^*\|_1 + \|\mathbf{W}\underline{\mathbf{u}} - \hat{\mathbf{b}}\|_1 + \|(\mathbf{W} - \underline{\mathbf{W}})\underline{\mathbf{u}}\|_1 \\
&+ 2\|\underline{\hat{\mathbf{b}}} - \hat{\mathbf{b}}\|_1 + \|\mathbf{\Delta}\|_1\Big)\|\mathbf{c}^*\|_\infty \\
\le &\left(\|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_1 + \frac{4t^d}{L} + 4\|\mathbf{\Delta}\|_1\right)\|\mathbf{c}^*\|_\infty
\end{aligned} \tag{21}$$

where L is the number of grid points in each dimension for rounding, $\underline{\hat{\mathbf{b}}}$ is rounded version of $\hat{\mathbf{b}}$, $\underline{\mathbf{W}}$ is rounded version of $\mathbf{W}$, $\mathbf{u}^*$ is optimal distribution by solving LP in step 20 in Algorithm 3, and $\underline{\mathbf{u}}$ is the uniform distribution on $\underline{D}$. Note that the second last inequality holds because

$$\|\underline{\mathbf{W}}\mathbf{u}^* - \underline{\hat{\mathbf{b}}}\|_1 \le \|\underline{\mathbf{W}} \cdot \underline{\mathbf{u}} - \underline{\hat{\mathbf{b}}}\|_1.$$

Also, the last inequality in (21) follows from

$$\|\underline{\hat{\mathbf{b}}} - \hat{\mathbf{b}}\|_1 \le \frac{t^d}{L} + \|\mathbf{\Delta}\|_1,$$

and

$$\|\mathbf{W}\underline{\mathbf{u}} - \hat{\mathbf{b}}\|_1 \le \|\mathbf{W}\underline{\mathbf{u}} - \mathbf{b}\|_1 + \|\mathbf{\Delta}\|_1 = \|\mathbf{\Delta}\|_1,$$

where the last equality holds since $\mathbf{W}\underline{\mathbf{u}} = \mathbf{b}$.

Define

$$\begin{aligned}
\eta_d &= \left|\frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} f(\underline{\mathbf{x}}) - \frac{1}{n}\sum_{\mathbf{x}\in D} f(\mathbf{x})\right|, \\
\eta_n &= 4\|\mathbf{\Delta}\|_1\|\mathbf{c}^*\|_\infty, \\
\eta_a &= \left|\frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} f(\mathbf{y}) - \frac{1}{m}\sum_{\mathbf{y}\in\tilde{D}} h_f^{M,t}(\boldsymbol{\theta}(\mathbf{y}))\right| + \left|\frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} h_f^{M,t}(\boldsymbol{\theta}(\underline{\mathbf{x}})) - \frac{1}{n}\sum_{\underline{\mathbf{x}}\in\underline{D}} f(\underline{\mathbf{x}})\right|, \\
\eta_s &= \|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_1\|\mathbf{c}^*\|_\infty, \\
\eta_r &= \frac{4t^d}{L}\|\mathbf{c}^*\|_\infty,
\end{aligned}$$

where η_d, η_n, η_a, η_s, η_r correspond to the discretization error, noise error, approximation error, sampling error and rounding error, respectively. Combining (20), (21) and the equations above, we have the error of the mechanism bounded by the sum of these five types of errors:

$$\left|q_f(\tilde{D}) - q_f(D)\right| \le \eta_d + \eta_n + \eta_a + \eta_s + \eta_r.$$

We now bound the five errors separately.

Discretization error η_d: Since $f \in C_B^K$ ($K \geq 1$), the first order derivatives of f are all bounded by B. Also the discretization precision of $[-1,1]^d$ is $\frac{1}{N}$, so the distance between the data in D and the corresponding data in $\underline{D}$ is $O(\frac{1}{N})$. Recall that N is number of grid points in each dimension for discretization. Thus we have

$$\eta_d = \left| \frac{1}{n} \sum_{\underline{\mathbf{x}} \in \underline{D}} f(\underline{\mathbf{x}}) - \frac{1}{n} \sum_{\mathbf{x} \in D} f(\mathbf{x}) \right| \leq \frac{dB}{N} = O\left(n^{-\frac{K}{2d+K}}\right).$$

Noise error η_n: Since M in Equation (19) is a constant only depending on d, K, B, We know $\|\mathbf{c}^*\|_\infty = O(1)$. Thus to bound $\eta_n = \|\mathbf{\Delta}\|_1 \cdot \|\mathbf{c}^*\|_\infty$, we only need to bound the l_1 norm of the t^d-dimensional vector $\mathbf{\Delta}$ which contains i.i.d. random variables Lap $\left(\frac{t^d}{n\epsilon}\right)$; or equivalently bound the sum of t^d i.i.d. random variables with exponential distribution. It is well known that such a sum satisfies gamma distribution. Simple calculations yields

$$\mathbb{P}\left(\|\mathbf{\Delta}\|_1 \leq 2\frac{t^{2d}}{n\epsilon} \right) \geq 1 - 10e^{-\frac{t^d}{5}}.$$

Thus, with probability $1 - 10e^{-\frac{t^d}{5}}$, we have

$$\eta_n = \|\mathbf{\Delta}\|_1 \|\mathbf{c}^*\|_\infty \leq O\left(\frac{t^{2d}}{n\epsilon}\right).$$

Approximation error η_a: Recall that for any $\mathbf{x}$,

$$g_f(\boldsymbol{\theta}(\mathbf{x})) = f(\mathbf{x}).$$

Denote

$$\|g_f - h_f^{M,t}\|_{[-\pi,\pi]^d} := \sup_{\boldsymbol{\theta} \in [-\pi,\pi]^d} \left| g_f(\boldsymbol{\theta}) - h_f^{M,t}(\boldsymbol{\theta}) \right|.$$

We have

$$\begin{aligned}\eta_a &= \left| \frac{1}{m} \sum_{\mathbf{y} \in \tilde{D}} f(\mathbf{y}) - \frac{1}{m} \sum_{\mathbf{y} \in \tilde{D}} h_f^{M,t}(\boldsymbol{\theta}(\mathbf{y})) \right| + \left| \frac{1}{n} \sum_{\underline{\mathbf{x}} \in \underline{D}} h_f^{M,t}(\boldsymbol{\theta}(\underline{\mathbf{x}})) - \frac{1}{n} \sum_{\underline{\mathbf{x}} \in \underline{D}} f(\underline{\mathbf{x}}) \right| \\ &\leq 2\|g_f - h_f^{M,t}\|_\infty \leq O\left(\frac{1}{t^{K+1}}\right).\end{aligned}$$

Sampling error η_s: It is easy to bound sampling error. Let $\mathbf{W}_\mathbf{r}$. be the row vector of matrix $\mathbf{W}$ indexed by $\mathbf{r}$. Recall that $-1 \leq W_{\mathbf{rk}} \leq 1$. Thus for each $\mathbf{r}$, by Chernoff bound we have that for any $\tau > 0$:

$$\mathbb{P}\left(|\tilde{b}_\mathbf{r} - \mathbf{W}_\mathbf{r}.\mathbf{u}^*| \geq \tau \right) \leq 2e^{-\frac{m\tau^2}{2}},$$

since $\tilde{b}_\mathbf{r}$ is just the average of m i.i.d. samples and $\mathbf{W}\mathbf{u}^*$ is its expectation. Next by union bound

$$\mathbb{P}\left(\|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_\infty \geq \tau \right) \leq 2t^d e^{-\frac{m\tau^2}{2}},$$

and therefore

$$\mathbb{P}\left(\|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_1 \geq t^d\tau \right) \leq 2t^d e^{-\frac{m\tau^2}{2}}.$$

Setting τ such that $2t^d e^{-\frac{m\tau^2}{2}} = e^{-t}$, we have that with probability $1 - e^{-t}$,

$$\|\tilde{\mathbf{b}} - \mathbf{W}\mathbf{u}^*\|_1 \leq O\left(\frac{t^{d+1/2}}{\sqrt{m}}\right).$$

Rounding error η_r: Since $\|\mathbf{c}^*\|_\infty$ is upper bounded by a constant, we have

$$\eta_r \leq O\left(\frac{t^d}{L}\right).$$

Putting it together: Combining the five types of errors, we have that with probability $1 - e^{-t} - 10e^{-\frac{t^d}{5}}$, the error of the mechanism satisfies

$$\left| \frac{1}{m} \sum_{\mathbf{y} \in \tilde{D}} f(\mathbf{y}) - \frac{1}{n} \sum_{\mathbf{x} \in D} f(\mathbf{x}) \right| \leq O\left(\frac{1}{N} + \frac{1}{t^{K+1}} + \frac{t^{2d}}{n\epsilon} + \frac{t^{d+\frac{1}{2}}}{\sqrt{m}} + \frac{t^d}{L} \right). \tag{22}$$

Recall that the mechanism sets

$$\begin{aligned} t = \lceil n^{\frac{1}{2d+K}} \rceil,\ N = \lceil n^{\frac{K}{2d+K}} \rceil, \\ m = \lceil n^{1+\frac{K+1}{2d+K}} \rceil,\ L = \lceil n^{\frac{d+K}{2d+K}} \rceil. \end{aligned}$$

The theorem follows after some simple calculation.

4.3.3 Running Time

It is not difficult to see that in this case the running time of the mechanism is dominated by solving the Linear Programming problem in step 20. (Because the time complexity of linear programming is with respect to arithmetic operations, all running time discussed here should be understand in this way.) To analyze the running time of the LP problem, observe that it could be rewritten in following standard form:

$$\begin{aligned} \max_{\bar{\mathbf{x}}} \quad & \bar{\mathbf{c}}^T \bar{\mathbf{x}} \\ \text{s.t.} \quad & \bar{\mathbf{A}}\bar{\mathbf{x}} = \bar{\mathbf{b}} \\ & \bar{\mathbf{x}} \succeq \mathbf{0} \end{aligned} \tag{23}$$

where

$$\bar{\mathbf{A}} = \begin{pmatrix} L \cdot \underline{\mathbf{W}} & L \cdot \mathbf{I}_{t^d} & -L \cdot \mathbf{I}_{t^d} \\ \mathbf{1}_{N^d}^T & 0 & 0 \end{pmatrix},$$

$$\bar{\mathbf{b}} = \begin{pmatrix} L \cdot \hat{\underline{\mathbf{b}}} \\ 1 \end{pmatrix}, \quad \bar{\mathbf{c}} = \begin{pmatrix} \mathbf{0} \\ \mathbf{1}_{t^d} \\ \mathbf{1}_{t^d} \end{pmatrix}, \quad \bar{\mathbf{x}} = \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{w} \end{pmatrix}.$$

$\bar{\mathbf{A}}$ is a $\bar{m} \times \bar{n}$ matrix where $\bar{m} = t^d + 1$ and $\bar{n} = N^d + 2t^d$. Note that 1) each element of $\underline{\mathbf{W}}$ is in $[-1, 1]$; 2) each element of $\underline{\hat{\mathbf{b}}}$ is in $[-1, 1]$; and 3) each element of $\underline{\mathbf{W}}$ and $\underline{\hat{b}}$ is rounded to precision $1/L$. So actually we have reduce to a LP problem (23), with elements of $\bar{\mathbf{A}}$, $\bar{b}$, $\bar{c}$ are all integers and bounded by L.

The most well-known worst-case complexity of the interior point algorithm for linear programming with integer parameters is $O(\bar{n}^3\tilde{L})$, where $\bar{n}$ is the number of variables and $\tilde{L}$ is the number of bits to encode the linear programming problem. Here we use a more refined bound given in Anstreicher (1999). By using this bound, we are able to prove a much better time complexity for our algorithm; because in the linear programming problem (23), the number of constraints is often much smaller than the number of variables. The bound we make use of for the complexity of linear programming is $O(\frac{\bar{n}^{1.5}\bar{m}^{1.5}}{\ln \bar{m}}\bar{L})$ (Anstreicher, 1999). Here, $\bar{L}$ is the size of LP problem in standard form defined as follows (Monteiro and Adler, 1989):

$$\begin{aligned}\bar{L} =& \lceil \log(1 + |\det(\bar{\mathbf{A}}_{max})|) \rceil + \lceil \log(1 + \|\bar{\mathbf{c}}\|_\infty) \rceil \\ &+ \lceil \log(1 + \|\bar{\mathbf{b}}\|_\infty) \rceil + \lceil \log(\bar{m} + \bar{n}) \rceil,\end{aligned}$$

where

$$\bar{\mathbf{A}}_{max} = \mathop{\arg\max}_{\mathbf{X} \text{is a square submatrix of} \bar{\mathbf{A}}} |\det(\mathbf{X})|.$$

Note that $\bar{m} < \bar{n}$, so the size of $\bar{\mathbf{A}}_{max}$ is at most $\bar{m} \times \bar{m}$. Therefore, we have

$$|\det(\bar{\mathbf{A}}_{max})| \leq \bar{m}! L^{\bar{m}},$$

and

$$\bar{L} = O(\bar{m}(\log \bar{m} + \log L) + \log \bar{n}).$$

Given $\bar{m} = O(t^d)$ and $\bar{n} = O(N^d)$, simple calculation shows that the total time complexity is

$$O\left(\frac{\bar{n}^{1.5}\bar{m}^{1.5}}{\ln \bar{m}}\bar{L}\right) = O\left(N^{1.5d}t^{2.5d}\right) = O\left(n^{\frac{3dK+5d}{4d+2K}}\right).$$

4.3.4 Size of the Output Synthetic Database

The size of synthetic dataset m is set in step 1 of the algorithm. ■

4.4 Proof of Smoothness

Here we prove the smoothness of the functions listed in Section 4.2. We only give the proof for Gaussian kernel function. The smoothness for logistic function follows directly from the derivative of hyperbolic tangent.

Proposition 19 *Let*

$$f(\mathbf{x}) = \sum_{j=1}^{J} \alpha_j \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_j\|^2}{2\sigma^2}\right),$$

where $\mathbf{x} \in \mathbb{R}^d$. *Let* $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_J)$. *Suppose* $\|\boldsymbol{\alpha}\|_1 \leq 1$. *Then for every* $K \leq \sigma^2$, $\|f\|_K \leq 1$.

Proof We first give a well-known inequality for Hermite polynomial. Proposition 19 follows immediately from this lemma.

Lemma 20 (Indritz 1961) *For Hermite polynomial of degree* k *defined as*

$$H_k(x) = (-1)^k e^{x^2} \frac{\mathrm{d}^k}{\mathrm{d}x^k} e^{-x^2},$$

where $k \in \mathbb{N}$ *and* $x \in (-\infty, \infty)$, *it satisfies following inequality:*

$$|H_k(x)| \leq (2^k k!)^{\frac{1}{2}} e^{\frac{1}{2}x^2}.$$

To prove Proposition 19, we only need to show that the K-norm of the Gaussian kernel function is bounded by 1 since $\|\boldsymbol{\alpha}\|_1 \leq 1$.

Let $g(x) = e^{-x^2}$. From Lemma 20 we directly have:

$$|\frac{\mathrm{d}^k}{\mathrm{d}x^k} g(x)| = |H_k(x) e^{-x^2}| \leq (2^k k!)^{\frac{1}{2}}.$$

Let $\mathbf{k} = (k_1, \ldots, k_d)$, and $|\mathbf{k}| = K$. Therefore, for $f(\mathbf{x})$ defined in Proposition 19, we have:

$$|D^{\mathbf{k}} f(\mathbf{x})| = \prod_{j=1}^{d} \frac{\mathrm{d}^{k_j}}{\mathrm{d}x_j^{k_j}} g\left(\frac{x_j - y_j}{\sqrt{2}\sigma}\right) \leq \left(\frac{1}{\sqrt{2}\sigma}\right)^K \left(\prod_{j=1}^{d} (2^{k_j} k_j!)\right)^{\frac{1}{2}} \leq \frac{(K!)^{\frac{1}{2}}}{\sigma^K}.$$

Obviously, when $K \leq \sigma^2$,

$$|D^{\mathbf{k}} f(\mathbf{x})| \leq \frac{K^{\frac{K}{2}}}{\sigma^K} \leq 1.$$

The proposition follows. ■

5. Additional Results

In this section we provide some additional results. In Section 5.1 we show that the two mechanisms described in Section 3 and Section 4 respectively can be modified to achieve slightly better accuracy and preserve (ϵ, δ)-differential privacy. In Section 5.2 we give a variant of our second mechanism which outputs synthetic database. The goal is to make the algorithm practically efficient, since as stated in Theorem 17 the running time of that mechanism is approximately $O(n^{3d/2})$, which is not acceptable in real applications.

5.1 (ϵ, δ)-Differentially Private Mechanisms

The two mechanisms given in Section 3 and Section 4 respectively preserve ϵ-differential privacy. It is easy to generalize them to preserve (ϵ, δ)-differential privacy and have slightly better accuracy.

For the first mechanism, simply setting

$$t = \lceil n^{\frac{2}{3d+2K}} (\log \frac{1}{\delta})^{-\frac{1}{3d+2K}} \rceil,$$

in step 1 of Algorithm 1 and Algorithm 2 respectively we have the following result.

Theorem 21 *Let the query set be*

$$Q_{C_B^K} = \{q_f = \frac{1}{n}\sum_{\mathbf{x}\in D} f(\mathbf{x}) : \quad f \in C_B^K\},$$

where $K \in \mathbb{N}$ and $B > 0$ are constants. Let the data universe be $[-1,1]^d$, where $d \in \mathbb{N}$ is a constant. Then the new mechanism related to Algorithm 1 and Algorithm 2 satisfies that for any $\epsilon > 0$, $\delta > 0$, the following hold:

1) *The mechanism is (ϵ, δ)-differentially private.*

2) *There is an absolute constant c such that for any $\beta \geq c \cdot e^{-n^{\frac{2}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{1}{3d+2K}}}$, the mechanism is (α, β)-accurate, where $\alpha = O\left(n^{-\frac{2K}{3d+2K}}(\log\frac{1}{\delta})^{\frac{K}{3d+2K}}/\epsilon\right)$, and the hidden constant depends only on d, K and B.*

3) *The running time of the mechanism is $O\left(n^{\frac{5d+2K}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{d}{3d+2K}}\right)$.*

4) *The running time for $\mathcal{S}$ to answer a query is $O\left(n^{\frac{2d+4+\frac{2d}{K}}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{d+2+\frac{d}{K}}{3d+2K}}\log(n)\right)$.*

The proof of Theorem 21 is along the same line as the proof of Theorem 3 plus standard use of the composition theorem (Dwork et al., 2010). We omit the details.

For the second mechanism, replacing step 1 and step 9 of Algorithm 3 by the following we obtain an (ϵ, δ)-differentially private mechanism.

1) Step 1. Set $t = \lceil n^{\frac{2}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{1}{3d+2K}}\rceil$, $N = \lceil n^{\frac{2K}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{K}{3d+2K}}\rceil$,
$m = \lceil n^{\frac{4d+4K+2}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{2d+2K+1}{3d+2K}}\rceil$, and $L = \lceil n^{\frac{2d+2K}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{d+K}{3d+2K}}\rceil$.

2) Step 9. $\hat{b}_{\mathbf{r}} = b_{\mathbf{r}} + \text{Lap}\left(\frac{(t^d\log\frac{1}{\delta})^{\frac{1}{2}}}{n\epsilon}\right)$.

Theorem 22 *Let the query set $Q_{C_B^K}$ be defined as in Theorem 17. Let the data universe be $[-1,1]^d$, where $d \in \mathbb{N}$ is a constant. Then the new mechanism related to Algorithm 3 satisfies that for any $\epsilon > 0$, $\delta > 0$, the followings hold:*

1) *The mechanism is (ϵ, δ)-differentially private.*

2) *There is an absolute constant c such that for any $\beta \geq c \cdot e^{-n^{\frac{2}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{1}{3d+2K}}}$, the mechanism is (α, β)-accurate, where $\alpha = O\left(n^{-\frac{2K}{3d+2K}}(\log\frac{1}{\delta})^{\frac{K}{3d+2K}}/\epsilon\right)$, and the hidden constant depends only on d, K and B.*

3) *The running time of the mechanism is $O\left(n^{\frac{3dK+5d}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{3dK+5d}{6d+4K}}\right)$.*

4) *The size of synthetic database is $O\left(n^{\frac{4d+4K+2}{3d+2K}}(\log\frac{1}{\delta})^{-\frac{2d+2K+1}{3d+2K}}\right)$.*

The proof of Theorem 22 is omitted for the same reason as above.

5.2 A Practical Variant of the Synthetic-Database-Output Mechanism

The time complexity of our second mechanism, which outputs synthetic database, can be nearly $n^{\frac{3d}{2}}$ to achieve n^{-1} accuracy for highly smooth queries, where d is the dimension of the data. In real application such a running time is unacceptable. We thus consider a variant of Algorithm 3 which turns out to be very efficient and suffers only from minor loss in accuracy in our experiments. (On the other hand we do not have theoretical guarantee for the utility of this algorithm any more.) Note that the running time of Algorithm 3 is dominated by the linear programming step. This LP problem has $O(N^d)$ variables and $O(t^d)$ constraints, where N^d is the number of discretized grids in $[-1,1]^d$ and t^d is the number of trigonometric polynomial basis functions. To make our algorithm practical, we consider a subset $\mathcal{S}$ of the N^d grids with size $C := |\mathcal{S}| \ll N^d$ and restrict the probability distribution $\mathbf{u}$ on this subset of grids (see step 20 in Algorithm 3). Similarly, we may use a subset of size R of the t^d trigonometric polynomial basis functions preferred to lower degrees. By doing this, the LP problem has C variables and R constraints.

The simplest approach to obtain a small subset $\mathcal{S}$ is sampling from the N^d grids in $[-1,1]^d$ uniformly. However, this approach suffers from substantial loss in accuracy (see the supplementary material for experimental results of this method), because $|\mathcal{S}|$ is extremely small compared to N^d, the probability that $\mathcal{S}$ contains data points in D (or close to D) is very small. In order to reduce the size of the LP problem and preserve the accuracy, we need a better approach to obtain $\mathcal{S}$. Formally, the problem of choosing a subset $\mathcal{S}$ for our purpose can be formulated as follows: We want a subset $\mathcal{S}$ so that

1) $\mathcal{S}$ is differentially private;

2) $|\mathcal{S}|$ is small;

3) For almost every data point x in D, there is a point in $\mathcal{S}$ close to x.

Note that without the privacy concern, one can simply let $\mathcal{S} = D$. But under the requirement of privacy, this problem is non-trivial. Here we adopt private PCA to obtain a low dimensional ellipsoid. The ellipsoid is spanned by the (private) top eigenvectors of the data covariance matrix with the square root of the (private) eigenvalues as the radius. In particular, we use a slightly modified version of the Private Subspace Iteration (PSI) mechanism (Hardt, 2013) to compute the private eigenpairs. The mechanism is described in Algorithm 4. Finally, we uniformly sample C points from the ellipsoid to form $\mathcal{S}$.

In the following three results, we show that the PSI mechanism is differentially private and accurate for the top eigenvectors and eigenvalues respectively. Note that Hardt (2013) shows that the principal angle between the space spanned by the top-k leading eigenvectors of the true data covariance matrix and the the space spanned by the output column vectors of $\mathbf{X}^{(T)}$ is small. However, it does not directly imply that the output private ellipsoid converges to the true PCA ellipsoid. Our results slightly strengthen the results in Hardt (2013). We show each private top-k eigenvalue is close to the true eigenvalue and each private top-k eigenvector is close to the true eigenvector, provided the true eigenvalues are well separated.

Theorem 23 (Accuracy of the eigenvectors) *Given a database D with $|D| = n$, let*

$$\mathbf{A} = \frac{1}{n}\sum_{\mathbf{y}\in D}(\mathbf{y} - \bar{\mathbf{y}})(\mathbf{y} - \bar{\mathbf{y}})^T$$

Algorithm 4 Private Subspace Iteration (Hardt, 2013)

Input: Database $D \in ([-1,1]^d)^n$

Output: $\boldsymbol{\lambda}$ (as private top-k eigenvalues), $\mathbf{X}^{(T)}$ (columns as private top-k eigenvectors).

Parameters: Number of iterations $T \in \mathbb{N}$, dimension k, privacy parameters $\epsilon, \delta > 0$, Denote GS as the Gram-Schmidt orthonormalization algorithm.

1: Set $\sigma = \frac{5d\sqrt{4k(T+1)\log(1/\delta)}}{n\epsilon}$, $\mathbf{A} = \frac{1}{n}\sum_{\mathbf{y}\in D}(\mathbf{y}-\bar{\mathbf{y}})(\mathbf{y}-\bar{\mathbf{y}})^T$, where $\bar{\mathbf{y}} = \frac{1}{n}\sum_{\mathbf{y}\in D}\mathbf{y}$. Thus $\mathbf{A}$ is the data covariance matrix.
2: Initialize: $\mathbf{G}^{(0)} \sim N(0,1)^{d\times k}$ (i.i.d. Gaussian distribution), $\mathbf{X}^{(0)} \leftarrow \mathrm{GS}(\mathbf{G}^0)$
3: **for all** $l = 1, 2, \ldots, T$ **do**
4: Sample $\mathbf{G}^{(l)} \sim N(0,\sigma^2)^{n\times k}$.
5: $\mathbf{W}^{(l)} = \mathbf{A}\mathbf{X}^{(l-1)} + \mathbf{G}^{(l)}$
6: $\mathbf{X}^{(l)} \leftarrow \mathrm{GS}(\mathbf{W}^{(l)})$
7: **end for**
8: Set $\hat{\lambda}_s = \|\mathbf{w}_s^{(T)}\|_2$ for $s = [k]$, where $\mathbf{w}_s^{(T)}$ is the s-th column of $\mathbf{W}^{(T)}$. Set $\boldsymbol{\lambda} = (\hat{\lambda}_s)_{s\le k}$.

be the data covariance matrix. Also let $\lambda_1 \ge \cdots \ge \lambda_d$ be the eigenvalues of A and $\gamma_k = \lambda_k/\lambda_{k+1} - 1$. Let

$$\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_k) \in \mathbb{R}^{d\times k},$$

where $\mathbf{u}_1, \ldots, \mathbf{u}_k$ are the top k eigenvectors of $\mathbf{A}$. Denote $\theta(\mathbf{u},\mathbf{v})$ as the angle between two vectors $\mathbf{u}$ and $\mathbf{v}$. Then, with parameters $k \le d/2$ and $T \ge C(\min_{s\le k}\gamma_s)^{-1}\log d$ for some sufficiently large constant C, the matrix $\mathbf{X}^{(T)} = (\mathbf{x}_1^{(T)}, \ldots, \mathbf{x}_k^{(T)}) \in \mathbb{R}^{d\times k}$ returned by Algorithm 4 satisfies that: with probability $1 - o(1)$, for all $s \le k$

$$\sin\theta(\mathbf{u}_s, \mathbf{x}_s^{(T)}) \le O\Big(\frac{\omega_s d^{\frac{3}{2}}\sqrt{kT\log T\log\left(\frac{1}{\delta}\right)}}{n\epsilon}\Big),$$

where

$$\omega_s = \begin{cases} \frac{1}{\gamma_1\lambda_1} & s = 1, \\ \max\{\frac{1}{\gamma_s\lambda_s}, \frac{1}{\gamma_{s-1}\lambda_{s-1}}\} & 2 \le s \le k. \end{cases}$$

Theorem 24 (Accuracy of the eigenvalues) *Using the same notions as in Theorem 23, let $\hat{\lambda}_s = \|\mathbf{w}_s^{(T)}\|_2$ for $s \le k$, where $\mathbf{w}_s^{(T)}$ is the sth column of $\mathbf{W}^{(T)}$. Then with probability $1 - o(1)$, we have for all $s \le k$*

$$|\hat{\lambda}_s - \lambda_s| \le O\Big(\frac{d^3kT\log T\log\left(\frac{1}{\delta}\right)}{n^2\epsilon^2\gamma_s^2\lambda_s^2} + \frac{kd\sqrt{T\log T\log\left(\frac{1}{\delta}\right)}}{n\epsilon}\Big).$$

Theorem 25 (Privacy) *If Algorithm 4 is executed with each $\mathbf{G}^{(l)}$ independently sampled as*

$$\mathbf{G}^{(l)} \sim N(0,\sigma^2)^{d\times k},$$

with

$$\sigma = \frac{5d\sqrt{4kT\log(1/\delta)}}{n\epsilon},$$

then Algorithm 4 satisfies (ϵ,δ)-differential privacy.

If Algorithm 4 is executed with each $\mathbf{G}^{(l)}$ independently sampled as

$$\mathbf{G}^{(l)} \sim Lap(\sigma)^{d\times k},$$

with

$$\sigma = \frac{50d^{3/2}kT}{n\epsilon},$$

then Algorithm 4 satisfies ϵ-differential privacy.

The proof of Theorem 23, Theorem 24 and Theorem 25 are given below.

Before we state it formally, let us take a closer look at the Theorem 3.3 in Hardt (2013): With high probability, the tangent of the angle between the space spanned by the top-k leading eigenvectors, namely eigenspace, and the space spanned by the output columns, namely output-space, is *small*, given regularity conditions. Our goal is the column-wise convergence between eigenvectors and output columns, which can be concluded from the simultaneous convergence between the increasing sequence of eigenspaces and the increasing sequence of output-spaces, given that they shared the same dimension. This constraint leads us to utilize a weaker version of Theorem 3.3 by specifying $r = k$, but the favored column-wise convergence at least compensated for the loss of tuning parameter r. Note that simply applying Theorem 3.3 consecutively for the sequence will not assure the high convergence probability $1 - o(1)$ and our analysis can be extended to the case $k = O(d)$, where the dimension d can grow as the size of database given the aptitude of added noise is adequate.

Our results generalize Theorem 3.3 in Hardt (2013) which proves that the principal angle between the subspace spanned by the private top-k eigenvectors and the subspace spanned by the true eigenvectors is small. But what we need in this paper is that each private eigenvector (and eigenvalue) converges to the true eigenvector (and eigenvalue). It is worth pointing out that simply applying Theorem 3.3 in Hardt (2013) consecutively for each k does not obtain our result, although our proofs rely on some results in Hardt (2013).

We first give three lemmas. The proof of the first two lemmas are straightforward and are omitted. For simplicity, below we denote $\|\mathbf{X}\|$ the spectral norm of a matrix $\mathbf{X}$.

Lemma 26 *Assuming the data universe $\mathcal{X} = [-1,1]^d$, for all pairs of neighbor databases D, D' with $|D| = |D'| = n$, let*

$$\mathbf{A}(D) = \frac{1}{n}\sum_{\mathbf{y}\in D}(\mathbf{y}-\bar{\mathbf{y}})(\mathbf{y}-\bar{\mathbf{y}})^T,$$

be the data covariance matrix. It holds that

$$\|\mathbf{A}(D) - \mathbf{A}(D')\| \le \frac{5d}{n}.$$

Lemma 27 *Let $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{n\times d}$, denote $\mathbf{A}_{kl} = (a_{ij})_{i\le k, j\le l}$ the (k,l)-sub matrix of $\mathbf{A}$ for any $k \le n$ and $l \le d$, then $\|\mathbf{A}_{kl}\| \le \|\mathbf{A}\|$.*

Lemma 28 *Let $\mathbf{U} \in \mathbb{R}^{d\times k}$ be a matrix with orthonormal columns. Let*

$$\mathbf{G}^{(1)}, ..., \mathbf{G}^{(T)} \sim N(0, \sigma^2)^{d\times k},$$

with $k \leq d$ and assume that $T \leq d$. Let $\mathbf{G}_s^{(l)}$ and $\mathbf{U}_s$ be the (d, s)-sub matrix of $\mathbf{G}^{(l)}$ and $\mathbf{U}$ respectively for $s \in [k]$. Then, with probability $1 - o(1)$,

$$\max_{l\in[T]} \|\mathbf{U}_s^T \mathbf{G}_s^{(l)}\| \leq O(\sigma\sqrt{k \log T}), \quad \forall\, s \in [k].$$

Proof

By Lemma A.3 in Hardt (2013),

$$\|\mathbf{U}_k^T \mathbf{G}_k^{(l)}\| \leq O(\sigma\sqrt{k \log T}).$$

The desired result follows from Lemma 27 since $\mathbf{U}_s^T \mathbf{G}_s^{(l)}$ is the (s, s)-sub matrix of $\mathbf{U}_k^T \mathbf{G}_k^{(l)}$. ■

Proof of Theorem 23 Consider the spectral decomposition $\mathbf{A} = \mathbf{Z}\mathbf{\Lambda}\mathbf{Z}^{-1}$, and denote $\mathbf{\Lambda} = \begin{pmatrix} \mathbf{\Lambda}_1 & \\ & \mathbf{\Lambda}_2 \end{pmatrix}$, and $\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_1 & \mathbf{Z}_2 \end{pmatrix}$, where $\mathbf{\Lambda}_1 \in \mathbb{R}^{s\times s}$ and $\mathbf{Z}_1 \in \mathbb{R}^{d\times s}$. Next we denote $\mathbf{U}_s = \mathbf{Z}_1\mathbf{\Lambda}_1\mathbf{Z}_1^T$ and $\mathbf{V}_s = \mathbf{Z}_2\mathbf{\Lambda}_2\mathbf{Z}_2^T$. Obviously we have

$$\mathbf{A} = \mathbf{U}_s\mathbf{\Lambda}_1\mathbf{U}_s^T + \mathbf{V}_s\mathbf{\Lambda}_2\mathbf{V}_s^T.$$

Let

$$\Delta(\mathbf{U}_s) = \max_l \|\mathbf{U}_s^T \mathbf{G}_s^{(l)}\|,$$

and

$$\Delta(\mathbf{V}_s) = \max_l \|\mathbf{V}_s^T \mathbf{G}_s^{(l)}\|,$$

where $\mathbf{G}_s^{(l)}$ is the (d, s)-sub matrix of $\mathbf{G}^{(l)}$. By Lemma 28, we concludes that with probability $1 - o(1)$, the following events occur simultaneously:

1. $\forall\, s \in [k], \Delta(\mathbf{U}_s) \leq O(\sigma\sqrt{k \log T})$,
2. $\forall\, s \in [k], \Delta(\mathbf{V}_s) \leq O(\sigma\sqrt{d \log T})$.

Notice that for all $s \leq k$, we have with probability $1 - o(1)$ that $\Delta(\mathbf{U}_s) \leq \Delta(\mathbf{V}_s)$ as we set $s \leq k \leq d/2$. Since $\arccos\theta(\mathbf{U}_s, \mathbf{X}_s^{(0)})$ is bounded, where $\mathbf{X}_s^{(0)}$ is the (d, s)-sub matrix of $\mathbf{X}^{(0)}$, we have for all $s \leq k$

$$\Delta(\mathbf{U}_s) \arccos\theta(\mathbf{U}_s, \mathbf{X}_s^{(0)}) \leq O(\sigma\sqrt{k \log T}).$$

Applying Theorem 2.9 in Hardt (2013), we have with probability of $1 - o(1)$, for all $s \in [k]$

$$\tan\theta(\mathbf{U}_s, \mathbf{X}_s^{(T)}) \leq O\Big(\frac{\sigma}{\gamma_s\lambda_s}\sqrt{d \log T}\Big). \tag{24}$$

For the case $s = 1$, the theorem is proved. Now for any fixed $s \in [k]$, notice that $\mathbf{u}_s$ is in the space spanned by $(\mathbf{u}_1, \ldots, \mathbf{u}_s)$ as well as the orthogonal complement of the space spanned by $(\mathbf{u}_1, \ldots, \mathbf{u}_{s-1})$, we have

$$\begin{aligned}
&\sin^2\theta(\mathbf{u}_s, \mathbf{x}_s^{(T)}) \\
&= \|\mathbf{U}_{s-1}\mathbf{U}_{s-1}^T\mathbf{x}_s^{(T)} + (\mathbf{I} - \mathbf{U}_s\mathbf{U}_s^T)\mathbf{x}_s^{(T)}\|^2 \\
&= \|\mathbf{U}_{s-1}\mathbf{U}_{s-1}^T\mathbf{x}_s^{(T)}\|^2 + \|(\mathbf{I} - \mathbf{U}_s\mathbf{U}_s^T)\mathbf{x}_s^{(T)}\|^2 \\
&\leq \sin^2\theta(\mathbf{U}_{s-1}, \mathbf{X}_{s-1}^{(T)}) + \sin^2\theta(\mathbf{U}_s, \mathbf{X}_s^{(T)}) \\
&\leq 2(\max\{\sin^2\theta(\mathbf{U}_{s-1}, \mathbf{X}_{s-1}^{(T)}), \sin^2\theta(\mathbf{U}_s, \mathbf{X}_s^{(T)})\}) \\
&\leq 2(\max\{\tan^2\theta(\mathbf{U}_{s-1}, \mathbf{X}_{s-1}^{(T)}), \tan^2\theta(\mathbf{U}_s, \mathbf{X}_s^{(T)})\}).
\end{aligned} \tag{25}$$

The theorem, for the case $s \geq 2$, is proved by substituting (24) into (25). ■

Proof of Theorem 24 Denote $\mathbf{x}_s = \mathbf{x}_s^{(T-1)}$, $\mathbf{w}_s = \mathbf{w}_s^{(T)}$, $\mathbf{g}_s$ as the s-column of $\mathbf{G}^{(T)}$, and $\theta^{(T)} = \theta(\mathbf{U}_s, \mathbf{X}_s^{(T)})$ for short. Let $\mathbf{x}_s = \mathbf{u} + \mathbf{u}^\perp$, where $\mathbf{u}$ is the eigenvector corresponding to λ_s. Then, since $\mathbf{u} = \mathbf{x}_s\cos\phi$ and $\mathbf{u}^\perp = \mathbf{x}_s\sin\phi$ for a $\phi \leq \theta^{(T)}$, we have

$$\begin{aligned}
\hat{\lambda}_s^2 &= \mathbf{w}_s^T\mathbf{w}_s \\
&= (\mathbf{A}\mathbf{x}_s + \mathbf{g}_s)^T(\mathbf{A}\mathbf{x}_s + \mathbf{g}_s) \quad , \\
&= \mathbf{x}_s^T\mathbf{A}^2\mathbf{x}_s + 2\mathbf{x}_s^T\mathbf{A}\mathbf{g}_s + \mathbf{g}_s^T\mathbf{g}_s
\end{aligned}$$

Let $R = 2\mathbf{x}_s^T\mathbf{A}\mathbf{g}_s + \mathbf{g}_s^T\mathbf{g}_s$, then by Lemma 28, with probability $1 - o(1)$, $R \leq O(\sigma\sqrt{k \log T}) + O(d\sigma^2)$.

$$\begin{aligned}
\mathbf{x}_s^T\mathbf{A}^2\mathbf{x}_s &= \mathbf{u}^T\mathbf{A}^2\mathbf{u} + \mathbf{u}^{\perp T}\mathbf{A}^2\mathbf{u}^\perp \\
&= \lambda_s^2\mathbf{u}^T\mathbf{u} + \mathbf{u}^{\perp T}\mathbf{A}^2\mathbf{u}^\perp \\
&\leq \lambda_s^2\|\mathbf{u}\|^2 + \lambda_1^2\|\mathbf{u}^\perp\|^2 \\
&\leq \lambda_s^2\cos^2\theta^{(T)} + \lambda_1^2\sin^2\theta^{(T)} \\
&= \lambda_s^2(1 - \sin^2\theta^{(T)}) + \lambda_1^2\sin^2\theta^{(T)} \\
&= \lambda_s^2 + (\lambda_1^2 - \lambda_s^2)\sin^2\theta^{(T)}.
\end{aligned}$$

Thus,

$$\begin{aligned}
|\hat{\lambda}_s - \lambda_s| &\leq \frac{(\lambda_1^2 - \lambda_s^2)}{\hat{\lambda}_s + \lambda_s}\sin^2\theta^{(T)} + O(\sigma\sqrt{k \log T}) + O(d\sigma^2) \\
&= O(\frac{\sigma^2 d \log T}{\gamma_s^2\lambda_s^2}) + O(\sigma\sqrt{k \log T}).
\end{aligned}$$

Plug in the setting of σ the corollary follows. ■

Dataset	Size (n)	# Attributes (d)
NOM	20643	116
NUR	12958	28
CTG	2126	42

Table 3: Summary of the dataset

Remark 29 *The accuracy of both the private eigenvectors and the private eigenvalues can be improved by considering the matrix coherence of A, as analyzed in Hardt (2013). Typically, the accuracy can be improved by a factor of $O\left(\frac{1}{d}\cdot \text{polylog}(d)\right)$.*

Proof of Theorem 25 The theorem follows immediately from Lemma 26 and Lemma 3.6 in Hardt (2013). ■

5.2.1 Experimental Results

Here we provide experimental results for the practical variant mechanism described above. It turns out that this algorithm is also far more efficient than our first mechanism given in Section 3 as answering a query is still time consuming in practice.

We adopt three datasets all from the UCI repository. A summary of the size and the number of attributes of these datasets is given in Table 3. Since the data universe considered in this paper is $[-1,1]^d$, we normalize each attribute to $[-1,1]$.

We conduct two sets of experiments in order to have a relatively comprehensive understanding of the *utility* of the differentially private synthetic database generated by our mechanism. In both sets of experiments we first output the synthetic database. In one set of experiments, we test the accuracy of query answering, as described in the previous sections. In the other set of experiments, we consider a very different task. We learn a SVM classifier from the synthetic database, and evaluate its accuracy (on the original data). We think that this task may be more useful from a practical point of view. It is worth pointing out that in the scenario of classification, the classification error ($0-1$ loss) is not a smooth function. Therefore the second set of experiments might be viewed, in a sense, as a test of how well our mechanism generalize to non-smooth functions.

The queries employed in the first set of experiments are linear combinations of Gaussian kernel functions. We use this type of functions because 1) These functions possess good smoothness property as stated in Section 2, and 2) linear combinations of Gaussian are universal approximators.

Detailed parameter setting of the query functions is as follows. We consider

$$f(\mathbf{x}) = \sum_{j=1}^{J} \alpha_j \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_j\|^2}{2\sigma^2}\right).$$

In all experiments, we set $J = 10$; α_j is randomly chosen from $[0,1]$, and $\mathbf{x}_j$ is randomly chosen from $[-1,1]^d$. We test various values of σ and various ϵ to see how the smoothness

Dataset	ϵ	Error	σ					Time (s)
			2	4	6	8	10	
NOM	0.1	Abs	0.0276	0.0859	0.0750	0.0520	0.0374	8.5
		Rel	0.6800	0.2192	0.1138	0.0657	0.0433	
	1	Abs	0.0155	0.0506	0.0330	0.0286	0.0202	8.2
		Rel	0.4600	0.1286	0.0521	0.0373	0.0237	
	10	Abs	0.0047	0.0199	0.0070	0.0096	0.0065	8.5
		Rel	0.1660	0.0594	0.0109	0.0128	0.0077	
NUR	0.1	Abs	0.0050	0.0018	0.0015	0.0004	0.0001	2.3
		Rel	0.0057	0.0018	0.0015	0.0004	0.0001	
	1	Abs	0.0015	0.0007	0.0003	0.0003	0.0002	3.1
		Rel	0.0018	0.0008	0.0003	0.0004	0.0001	
	10	Abs	0.0037	0.0001	0.0001	0.0005	0.0001	2.7
		Rel	0.0042	0.0001	0.0001	0.0006	0.0001	
CTG	0.1	Abs	0.1022	0.2113	0.1479	0.0984	0.0693	0.7
		Rel	0.8590	0.3871	0.1981	0.1139	0.0770	
	1	Abs	0.0970	0.1605	0.1193	0.0770	0.0505	0.7
		Rel	0.8007	0.2923	0.1597	0.0891	0.0560	
	10	Abs	0.0462	0.0632	0.0430	0.0353	0.0147	0.8
		Rel	0.3881	0.1184	0.0602	0.0415	0.0165	

Table 4: Worst-case error for the variant of the database-outputting mechanism

of the query function and privacy parameter affect the performance of the algorithm (see below for detailed results).

We use different performance measures to evaluate the algorithm. The goal is to have a comprehensive understanding of the performance of the mechanism. We consider the worst-case error of the mechanism over the set of queries. Because our query set, i.e., linear combination of Gaussian Kernels, contains infinitely many functions, we randomly choose 10^4 queries in each experiment. The worst-case error is over these 10^4 queries.

We give both absolute error and relative error for all experiments. The absolute error of a query q_f is defined as $|q_f(D) - q_f(\tilde{D})|$; and relative error is defined as $|\frac{q_f(D)-q_f(\tilde{D})}{q_f(D)}|$. We present relative error because in certain cases (e.g. when σ is small) $f(\mathbf{x})$ is very small for most $\mathbf{x} \in D$. Therefore in this case a small absolute error does not necessarily imply good performance, and relative error is more informative[3].

We present the running time of the mechanism for outputting the synthetic database in each experiment. The computer used in all the experiments is a workstation with 2 Intel Xeon X5650 processors of 2.67GHz and 32GB RAM.

We present the performance of the ϵ-differentially private algorithm in Table 4. For each dataset, both absolute error and relative error, as average of 20 rounds, are reported

3. One also needs to be careful when using relative error. In our experiments, we deliberately set $\alpha_j \in [0,1]$. So $f(\mathbf{x}) \geq 0$ for all $\mathbf{x}$. If instead we set $\alpha_j \in [-1,1]$, then $f(\mathbf{x})$ can be either positive or negative, and it is possible that $q_f(D)$ is close to zero while $f(\mathbf{x})$ is not small for most $\mathbf{x} \in D$. In such a case, a large relative error does not necessarily imply a bad performance.

Dataset	Original	ϵ		
		0.1	1	10
NOM	0.9353	0.6328	0.7959	0.8491
NUR	0.9081	0.5361	0.7748	0.8164
CTG	0.9755	0.5309	0.5853	0.6116

Table 5: AUC for the variant of the database-outputting mechanism

sequentially. We make use of linear combination of Gaussian with different values of σ as the query functions. The last column of the table lists the running time with respect to the worst σ of the algorithm for outputting the synthetic database.

Now we analyze the experimental results in Table 4 in greater detail. First, the algorithm is quite efficient. On all datasets, the mechanism outputs the synthetic databases in a few seconds. Next consider the accuracy. As explained earlier, the relative error is more meaningful in our experiments. It can be seen that except for the case $\sigma = 2$ (recall that $K = \sigma^2$), the accuracy is reasonably good. The relative errors decrease monotonically as the the order of smoothness of the queries increases.

We then turn to describe the second set of experiments. We randomly partition each dataset into two subsets of equal size. One subset is used as *training* dataset, the other as *test* dataset. By running our mechanism, we generate a differentially private synthetic database using the training dataset as input. Since our task is classification, and the attribute values in the synthetic dataset are continuous ($[-1, 1]$), we round the label attribute to $\{-1, 1\}$. We then learn a SVM classifier from the synthetic dataset. Finally we test the classifier's performance on the *test* dataset. As a comparison, we also list the performance of the SVM classifier learned from the non-private training data, as shown in the second column of Table 5. Here, the performance measures are Area Under ROC Curve (AUC).

We test three values of ϵ, and the performances are given in Table 5. For each dataset, the AUC is an average of 10 rounds. It can be seen that when ϵ is small, there is usually a relatively big utility gap. But as ϵ getting large, the performances improve quickly.

6. Conclusion

In this paper, we study differentially private mechanisms with respect to the K-smooth queries. We first propose a differentially private mechanism for efficiently answering smooth queries. The running time for outputting the summary is $O(n^{1\sim1.5})$. For queries of high order smoothness, the running time for answering a query is sublinear, and nearly $O(n^{\epsilon_0})$. Furthermore, the mechanisms achieve an accuracy nearly $O(n^{-1})$ in highly smooth case, much better than the sampling error $O(n^{-1/2})$ which is inherent to differentially private mechanisms answering general queries.

From a practical viewpoint, outputting a synthetic database while preserving differential privacy is more appealing. In this paper, we also propose a differentially private mechanism which output synthetic database. The user can obtain accurate answers to all smooth queries from the synthetic database. Our mechanisms run in polynomial time, while existing

algorithms run in super-exponential time. This synthetic dataset outputing mechanism achieves almost the same level of accuracy as the query-answering mechanism.

There is a future direction we think worth exploring. As mentioned in Introduction, there exists an efficient and differentially private algorithm which outputs a synthetic database and is accurate to the class of rectangle queries defined on $[-1, 1]^d$ (Blum et al., 2008). Rectangle queries are not smooth. They are specified by indicator functions which are not even continuous. The mechanism proposed in Blum et al. (2008) is completely different to the mechanism for smooth queries given in this paper. Thus an immediate question is: can we develop efficient mechanisms which output synthetic database and preserve differential privacy for a natural class of queries containing both smooth and important non-smooth functions?

Acknowledgments

This work was supported by National Basic Research Program of China (973 Program)(grant no. 2015CB352502), NSFC(61573026, 61222307), IBM Faculty Award and a grant from Microsoft Research Asia.

References

C. Aggarwal and P. Yu. A general survey of privacy preserving data mining models and algorithms. In *Privacy-Preserving Data Mining*, chapter 2, pages 11–52. Springer, 2008.

K. M. Anstreicher. Linear programming in $O(\frac{n^3}{\ln n}L)$ operations. *SIAM J. on Optimization*, 9(4):803–812, April 1999.

B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *PODS*, pages 273–282. ACM, 2007.

A. Blum and A. Roth. Fast private data release algorithms for sparse queries. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 395–410. Springer, 2013.

A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, pages 609–618. ACM, 2008.

K. Chaudhuri and D. Hsu. Sample complexity bounds for differentially private learning. In *COLT*, 2011.

K. Chaudhuri, C. Monteleoni, and A.D. Sarwate. Differentially private empirical risk minimization. *JMLR*, 12:1069–1109, 2011.

K. Chaudhuri, A. Sarwate, and K. Sinha. Near-optimal differentially private principal components. In *NIPS*, pages 998–1006, 2012.

M. Cheraghchi, A. Klivans, P. Kothari, and H.K. Lee. Submodular functions are noise stable. In *SODA*, pages 1586–1592. SIAM, 2012.

C.W. Clenshaw and A.R. Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, 2(1):197–205, 1960.

R.A. DeVore and G. G. Lorentz. *Constructive approximation*, volume 303. Springer Verlag, 1993.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. *TCC*, pages 265–284, 2006.

C. Dwork, M. Naor, O. Reingold, G.N. Rothblum, and S. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC*, pages 381–390. ACM, 2009.

C. Dwork, G.N. Rothblum, and S. Vadhan. Boosting and differential privacy. In *FOCS*, pages 51–60. IEEE, 2010.

T. Gerstner and M. Griebel. Numerical integration using sparse grids. *Numerical algorithms*, 18(3-4):209–232, 1998.

A. Gupta, M. Hardt, A. Roth, and J. Ullman. Privately releasing conjunctions and the statistical query barrier. In *STOC*, pages 803–812. ACM, 2011.

M. Hardt. Robust subspace iteration and privacy-preserving spectral analysis. *arXiv preprint arXiv:1311.2495*, 2013.

M. Hardt and G.N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70. IEEE Computer Society, 2010.

M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. In *NIPS*, 2012a.

M. Hardt, G. N. Rothblum, and R. A. Servedio. Private data release via learning thresholds. In *SODA*, pages 168–187. SIAM, 2012b.

J. Indritz. An inequality for hermite polynomials. *Proceedings of the American Mathematical Society*, 12(6):981–983, 1961.

P. Jain, P. Kothari, and A. Thakurta. Differentially private online learning. In *COLT*, 2012.

D. Kifer and B.R. Lin. Towards an axiomatization of statistical privacy and utility. In *PODS*, pages 147–158. ACM, 2010.

J. Lei. Differentially private M-estimators. In *NIPS*, 2011.

R. D.C. Monteiro and I. Adler. Interior path following primal-dual algorithms. Part I: Linear programming. *Math. Program.*, 44(1):27–41, June 1989.

A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *STOC*, pages 765–774. ACM, 2010.

A. Smola, B. Schölkopf, and K. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11(4):637–649, 1998.

S. A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Dokl. Akad. Nauk SSSR*, volume 4, page 123, 1963.

V.N. Temlyakov. *Approximation of periodic functions*. Nova Science Pub Inc, 1994.

J. Thaler, J. Ullman, and S. Vadhan. Faster algorithms for privately releasing marginals. In *ICALP*, pages 810–821. Springer, 2012.

J. Ullman. Answering $n^{2+o(1)}$ counting queries with differential privacy is hard. In *STOC*. ACM, 2013.

J. Ullman. Private multiplicative weights beyond linear queries. In *PODS*, 2015.

A. van der Vart and J.A. Wellner. *Weak Convergence and Empirical Processes*. Springer, 1996.

M. Vyazovskaya and N. Pupashenko. On the normalizing multiplier of the generalized jackson kernel. *Mathematical Notes*, 80(1-2):19–26, 2006.

G. Wahba et al. Support vector machines, reproducing kernel Hilbert spaces and the randomized gacv. *Advances in Kernel Methods-Support Vector Learning*, 6:69–87, 1999.

L. Wang. Smoothness, disagreement coefficient, and the label complexity of agnostic active learning. *JMLR*, 12:2269–2292, 2011.

Z. Wang, K. Fan, J. Zhang, and L. Wang. Efficient algorithm for privately releasing smooth queries. In *NIPS*, 2013.

L. Wasserman and S. Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010.

O. Williams and F. McSherry. Probabilistic inference and differential privacy. In *NIPS*, 2010.

Journal of Machine Learning Research 17 (2016) 1-21 Submitted 10/14; Revised 11/15; Published 4/16

Subspace Learning with Partial Information

Alon Gonen ALONGNN@CS.HUJI.AC.IL
School of Computer Science and Engineering
The Hebrew University
Jerusalem, Israel

Dan Rosenbaum DANRSM@CS.HUJI.AC.IL
School of Computer Science and Engineering
The Hebrew University
Jerusalem, Israel

Yonina C. Eldar YONINA@EE.TECHNION.AC.IL
Department of Electrical Engineering
Technion, Israel Institute of Technology
Haifa, Israel

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
School of Computer Science and Engineering
The Hebrew University
Jerusalem, Israel

Editor: Kevin Murphy

Abstract

The goal of subspace learning is to find a k-dimensional subspace of $\mathbb{R}^d$, such that the expected squared distance between instance vectors and the subspace is as small as possible. In this paper we study subspace learning in a *partial information* setting, in which the learner can only observe $r \leq d$ attributes from each instance vector. We propose several efficient algorithms for this task, and analyze their sample complexity.

Keywords: principal components analysis, budgeted learning, statistical learning, learning with partial information, learning theory

1. Introduction

Subspace learning is a dimensionality reduction technique in a variety of applications such as face recognition (Yang et al., 2004), image compression (Du and Fowler, 2007), and document classification (Papadimitriou et al., 1998). Recently, there has been growing interest in subspace learning from partially observed data (e.g., (Chen et al., 2013; Wang and Poor, 1998; Chi et al., 2013)). As motivation, consider the scenario of subspace learning from corrupted data. As discussed in Chen et al. (2013), data corruption may cause some (or even most) of the attributes to be missing. Another typical scenario is subspace learning from multiple sources. Many applications (e.g., wireless sensor networks (Chi et al., 2013)) rely on data which is collected from multiple sources (e.g., sensors). When the data dimension is high, it may be impossible or prohibitively expensive to collect every data entry from every source. Note that in the first scenario, we have no control over which attributes are missing, while in the second one a learner may actively choose which attributes to observe.

The subspace learning problem is formally defined as follows. Let $\mathcal{X}$ be a subset of the Euclidean unit ball in $\mathbb{R}^d$, and let P be some unknown distribution over $\mathcal{X}$. Our goal is to find a rank-k projection matrix $\Pi \in \mathbb{R}^{d\times d}$ such that the expected squared distance, $\mathbb{E}_{x\sim P}[\|x - \Pi x\|_2^2]$, is as small as possible.

When $\mathcal{X}$ is a finite set and P is the uniform distribution over $\mathcal{X}$, the optimal solution to the subspace learning problem is given by the Principal Component Analysis (PCA) algorithm, which returns the projection matrix that corresponds to the k leading eigenvectors of the matrix $\frac{1}{|\mathcal{X}|}\sum_{x\in\mathcal{X}} xx^\top$. In the more general stochastic optimization setting of subspace learning, $\mathcal{X}$ is not restricted to be a finite set, P is an arbitrary distribution over $\mathcal{X}$, and the information given to the learner has the form of an i.i.d. training sequence $(x_1, \ldots, x_m) \sim P^m$.

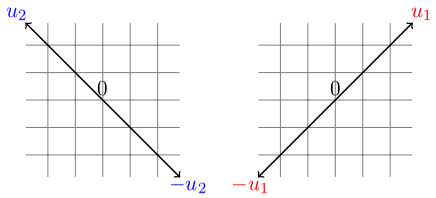
In the usual full information setting of subspace learning, the learner has access to all attributes of the sampled vectors. In this paper, we consider subspace learning in a partial information setting, in which only a subset of indices from each vector can be observed. Inspired by the two applications presented above, we study two variants of this problem, which will be named the *passive* setting, and the *active* setting, respectively. In the passive setting, we assume that each attribute is observed with probability $p = r/d$ ($r \leq d$). Therefore, the expected number of observed attributes from each vector is r. In the active setting, the learner can choose (possibly at random) r attributes to be revealed. The *sample complexity* of a subspace learning algorithm is defined as the number of samples that are needed by the algorithm in order to find a projection matrix Π with expected squared distance, $\mathbb{E}_{x\sim P}[\|x - \Pi x\|_2^2]$, of at most ϵ more than the optimal expected squared distance. The sample complexity for each of the models is defined as the minimal sample complexity attained by any algorithm. In this paper we propose efficient algorithms for both settings and analyze their sample complexity. We also provide several lower bounds on the sample complexity that can be attained by any algorithm.

1.1 Our Contribution

Our first observation is that subspace learning (in both the passive and active models) using $r = 1$ attributes is impossible. Consider the task of learning a 1-dimensional subspace (a line) in $\mathbb{R}^2$ (see Figure 1). Let $u_1 = (\alpha, \alpha)$, $u_2 = (-\alpha, \alpha)$. Denote by P_1 the uniform distribution over $\{u_1, -u_1\}$, and by P_2 be the uniform distribution over $\{u_2, -u_2\}$. Suppose that the actual distribution P is chosen uniformly at random from $\{P_1, P_2\}$. The task of subspace learning in this case amounts to distinguishing between P_1 and P_2. However, since each single coordinate is distributed uniformly over $\{-\alpha, \alpha\}$, the learner does not obtain any information from a single observation, and hence cannot identify the right subspace.

For the case $2 \leq r \leq d$ we propose two efficient algorithms, named Partially Observed PCA (Section 2.1) and Matrix Bandit Exponentiated Gradient (Section 3.1), which are designed for the passive and active models, respectively.

Partially Observed PCA (POPCA) is an extension of the PCA algorithm. Denote the population covariance matrix $\mathbb{E}[xx^\top]$ by C. The optimal projection matrix, denoted $\Pi^\star$, is the projection onto the subspace that is spanned by the k leading eigenvectors of C. An ERM (empirical risk minimizer, i.e., an algorithm that minimizes the empirical loss) for the full information setting is given by PCA which returns the projection matrix corresponding to the k leading eigenvectors of



(a) (b)

Figure 1: Impossibility of subspace learning using $r = 1$ observed attributes. The distribution of the observed attribute is identical for (a) and for (b), therefore they are indistinguishable.

the matrix $m^{-1}\sum_{i=1}^{m} x_i x_i^\top$ (whose expected value is C). Similarly, the POPCA algorithm uses the random observations in order to construct an estimate $\hat{C}$ of C, and approximates $\Pi^\star$ using the projection matrix onto the k leading eigenvectors of $\hat{C}$. We analyze the sample complexity of this algorithm, showing (see Corollary 3) that it is bounded from above by $(d/r)^2\frac{k}{\epsilon^2}$.

In the full information setting, the sample complexity is known to be $O(k/\epsilon^2)$ (This bounds coincides with our bound when $r = d$). Hence, the (multiplicative) price of partial information, that is, how many more examples we need in order to compensate for the lack of full information on each individual example, is $O((d/r)^2)$. It is interesting to understand whether a lower price can be achieved. In Theorem 4 we prove that the sample complexity of every algorithm in the passive model is $\Omega\left((d/r)^2\frac{k}{\epsilon^2}\right)$. The optimality of POPCA in the passive mode is thus established. Another appealing property of POPCA is that, in terms of computational complexity, the challenge of partial information does not incur any additional cost; While the sample complexity grows as r decreases, the runtime per iteration decreases by the same order.

Next, we investigate the active model and ask whether the price of partial information can be reduced due to the ability of the learner to actively choose the observed attributes. Intuitively, one may hope that the learning process would reveal some useful information that can be utilized while choosing which coordinates to observe. Our second algorithm, called Matrix Bandit Exponentiated Gradient (MBEG), exploits the active setting by maintaining a "weight matrix" which is updated with every observation, and induces a non-uniform attribute sampling distribution. For MBEG, we derive an upper bound of $\max\left\{8k\cdot\frac{d+r}{r}\cdot\frac{k}{\epsilon^2}\,,\,\frac{d^4 r}{2(d+r)}\right\}\cdot\log(d)$ (see Theorem 7) on the sample complexity. We note that if ϵ is small enough, then the right term in the bound becomes irrelevant, and thus a linear dependence on d/r is obtained (for a detailed comparison, see Section 3.2). The (almost trivial) fact that every subspace learner, even in the active model, must have a sample complexity that grows linearly with d/r is proved in Appendix C. Hence, the dependence of MBEG on d/r is optimal, in the regime where ϵ is small.

The results in the active model immediately lead to the following question: Can we attain a linear price for partial information in the active model, independently of the required accuracy? We discuss possible directions for tackling this question in Section 4.

1.2 Related Work

In the full information setting, it has been shown by Shawe-Taylor et al. (2005) and Blanchard et al. (2007) that the optimal sample complexity of subspace learning is at most $O(k/\epsilon^2)$, and this upper bound is achievable by applying PCA on i.i.d. samples according to the distribution P. A similar result is obtained by applying the Stochastic Gradient Descent algorithm (Arora et al., 2013).

Subspace learning in the partial information setting has been studied in Chi et al. (2013), where an algorithm named PETRELS is proposed. However, no formal guarantees are derived for this method. The setting in Mitliagkas et al. (2014) is similar to our passive setting, but they assume that the distribution that generates the instance vectors is Gaussian.

A closely related problem to the task of subspace learning (in the passive setting) is the approximation of the covariance matrix using partially observed attributes. We discusses this relation in Section 2.2.

One possible way to tackle the challenge of subspace learning with partial information is based on the matrix completion method. For example, we may think of the partially observed examples as a data matrix with unobserved entries. Then, one could first fill in the missing entries using a matrix completion technique (e.g., as described in Candès and Recht (2009)), and then apply PCA to the data matrix. We note that this approach may work[1] provided that the average number of observed attributes per example is sufficiently large. More precisely, according to the main result of Candès and Recht (2009), the number of observed attributes per example (column of the data matrix) should scale with the rank of the data matrix (which may be large as d). In contrast, our focus in this paper is on methods that work even when the number of observed attributes per example is much smaller (two attributes per instance suffice).

The active setting resembles the setting of the multi-armed bandit problem (Auer et al., 2002), in which the learner obtains limited feedback at each time, namely, it receives only the reward of the chosen arm. The challenge of learning linear predictors in $\mathbb{R}^d$ with partially observed attributes (e.g., as in Cesa-Bianchi et al. (2011)) may be seen as an extension of this problem. One of the most significant challenges in this work is to adapt the technique used in the vector setting (e.g., those employed by the Exp3 algorithm of Auer et al. (2002)) to the corresponding matrix setting. We already observed one difference: While a single arm suffices in the vector case, subspace learning with $r = 1$ attributes is impossible.

Our MBEG algorithm can be seen as an extension of the Online PCA algorithm of Warmuth and Kuzmin (2008) (see also Nie et al. (2013)) to the partial information setting. Similarly to their approach, MBEG maintains a weight matrix which induces a probability distribution over projection matrices. In MBEG, this weight matrix also induces a distribution over which attributes to observe.

2. The Passive Setting

We begin by investigating the passive setting. We start by describing an algorithm for this case. We then analyze its sample complexity, and discuss its implementation.

2.1 Partially Observed Principal Component Analysis (POPCA)

In this section we describe the POPCA algorithm. We start by reviewing the definition of the loss function and characterizing the minimizer of the loss.

1. Under some additional assumptions on the data such as the incoherence assumption made in Candès and Recht (2009)

Denote the set of projection matrices from $\mathbb{R}^d$ onto $\mathbb{R}^k$ by $\mathcal{P}_k^d$. Since $\Pi^2 = \Pi$ for any $\Pi \in \mathcal{P}_k^d$, the loss of a projection matrix $\Pi \in \mathcal{P}_k^d$ can be expressed as

$$L(\Pi) = \mathbb{E}\|x - \Pi x\|_2^2 = \mathbb{E}\left[\|x\|_2^2 - 2x^\top \Pi x + x^\top \Pi^\top \Pi x\right] = \mathbb{E}\left[\|x\|_2^2 - x^\top \Pi x\right] , \quad (1)$$

Define the inner product of matrices A, B by $\langle A, B\rangle = \mathrm{tr}(A^\top B)$. We can further rewrite the loss as

$$L(\Pi) = \mathbb{E}\left[\|x\|_2^2 - \langle \Pi, xx^\top\rangle\right] = \mathbb{E}\left[\|x\|_2^2\right] - \mathbb{E}\left[\langle \Pi, xx^\top\rangle\right] = \mathbb{E}\left[\|x\|_2^2\right] - \langle \Pi, \mathbb{E}[xx^\top]\rangle .$$

Denote the covariance matrix $\mathbb{E}[xx^\top]$ by C. Since $\|x\|_2^2$ does not depend on Π, the goal of subspace learning is equivalent to finding a projection matrix Π such that

$$\langle \Pi, -C\rangle \le \min_{\Pi' \in \mathcal{P}_k^d} \langle \Pi', -C\rangle + \epsilon . \quad (2)$$

It is well-known that the rank-k matrix which minimizes the expression $\langle \Pi', -C\rangle$ is the matrix $\sum_{i=1}^k v_i v_i^\top$, where $v_1, \ldots, v_k$ are the leading eigenvectors of C. The PCA approach for subspace learning in the full information setting replaces C with $C(S) = \frac{1}{m}\sum_{i=1}^m x_i x_i^\top$, where $S = (x_1, \ldots, x_m)$ is an i.i.d. training sequence drawn according to the distribution P. Clearly, $\mathbb{E}[C(S)] = C$. In our case, we will construct an unbiased estimate of C based on partially observed examples, as detailed below.

Consider an instance vector $x \sim P$ and let $\hat{x}$ be the observed vector. According to our assumptions, for each $i \in [d]$, the i-th coordinate of $\hat{x}$ satisfies

$$\hat{x}_i = \begin{cases} x_i & \text{w.p. } r/d \\ 0 & \text{w.p. } 1 - r/d . \end{cases} \quad (3)$$

Denote $p = r/d$. Similarly to Mitliagkas et al. (2014), we form the estimate

$$\hat{C}_{\hat{x}} = \frac{1}{p^2}\hat{x}\hat{x}^\top + \left(\frac{1}{p} - \frac{1}{p^2}\right)\mathrm{diag}(\hat{x}\hat{x}^\top) . \quad (4)$$

Indeed, it is easy to verify that $\hat{C}_{\hat{x}}$ forms an unbiased estimate of C:

$$\mathbb{E}\left[\left(\frac{1}{p^2}\hat{x}\hat{x}^\top\right)_{i,j}\right] = \begin{cases} \frac{1}{p}\mathbb{E}[x_i^2] & i = j \\ \mathbb{E}[x_i x_j] & i \ne j , \end{cases}$$

and

$$\mathbb{E}\left[\left(\frac{1}{p} - \frac{1}{p^2}\right)\mathrm{diag}(\hat{x}\hat{x}^\top)_{i,j}\right] = \begin{cases} \mathbb{E}[x_i^2] - \frac{1}{p}\mathbb{E}[x_i^2] & i = j \\ 0 & i \ne j . \end{cases}$$

Summing the corresponding entries, we see that $\mathbb{E}[(\hat{C}_{\hat{x}})_{i,j}] = \mathbb{E}_{x\sim P}[x_i x_j] = C_{i,j}$. Therefore, $\mathbb{E}[\hat{C}_{\hat{x}}] = C$.

The POPCA algorithm (see Algorithm 1) assumes access to m i.i.d. vectors sampled according to P. For each instance vector, it forms an estimate according to 4. By averaging these m estimates we obtain $\hat{C}$. The returned projection corresponds to the k leading eigenvectors of $\hat{C}$.

Algorithm 1 POPCA

Input: $r, k \le d$
$\hat{C} = 0 \in \mathbb{R}^{d\times d}$
for $i = 1$ **to** m **do**
  Let $x_i \sim P$ and let $\hat{x}_i$ be the observed vector
  $\hat{C}_{\hat{x}_i} = \frac{d^2}{r^2}\hat{x}_i\hat{x}_i^\top + \left(\frac{d}{r} - \frac{d^2}{r^2}\right)\mathrm{diag}(\hat{x}_i\hat{x}_i^\top)$
  $\hat{C} = \hat{C} + \frac{1}{m}\hat{C}_{\hat{x}_i}$
end for
Compute the eigendecomposition $\hat{C} = \sum_{j=1}^d \lambda_j v_j v_j^\top$
Assuming $\lambda_1 \ge \ldots \ge \lambda_d$, return $\hat{\Pi} = \sum_{j=1}^k v_j v_j^\top$

2.2 Analysis of POPCA

The following lemma relates the success of Algorithm 1 to the quality of the estimation $\hat{C}$ of the covariance matrix C.

Lemma 1 *Suppose that the final estimate $\hat{C}$ of POPCA satisfies*

$$\mathbb{E}[\|C - \hat{C}\|_F] \le \epsilon/\sqrt{k} . \quad (5)$$

Then, the resulting projection matrix $\hat{\Pi}$ satisfies the desired bound

$$\mathbb{E}[\langle \hat{\Pi}, -C\rangle] \le \min_{\Pi' \in \mathcal{P}_k^d} \langle \Pi', -C\rangle + \epsilon .$$

Proof Using the Cauchy-Schwarz inequality, we get

$$\begin{aligned} \mathbb{E}[\sup_{\Pi\in\mathcal{P}_k^d} \langle \Pi, -C\rangle - \langle \Pi, -\hat{C}\rangle] &\le \mathbb{E}[\sup_{\Pi\in\mathcal{P}_k^d} \|\Pi\|_F \|\hat{C} - C\|_F] \\ &\le \sup_{\Pi\in\mathcal{P}_k^d} \|\Pi\|_F \, \mathbb{E}[\|\hat{C} - C\|_F] \\ &\le \sqrt{k}\cdot\epsilon/\sqrt{k} \\ &= \epsilon . \end{aligned}$$

Consequently, since $\hat{\Pi} = \mathrm{argmin}_{\Pi\in\mathcal{P}_k^d}\langle \Pi, -\hat{C}\rangle$, we have

$$\begin{aligned} \mathbb{E}[\langle \hat{\Pi}, -C\rangle - \langle \Pi^\star, -C\rangle] &= \mathbb{E}[\langle \hat{\Pi}, -C\rangle - \langle \Pi^\star, -\hat{C}\rangle] \\ &\le \mathbb{E}[\langle \hat{\Pi}, -\hat{C}\rangle - \langle \Pi^\star, -\hat{C}\rangle] + \epsilon \\ &\le \epsilon , \end{aligned}$$

which completes the proof. ■

In view of Lemma 1, obtaining upper bound on the sample complexity of POPCA reduces to analyzing the quality of the covariance estimation. A fairly vast body of literature exists on the latter task in the full-information scenario, and several results are known in the case of missing entries. For example, Lounici (2014) considered a setting similar to our passive setting. Corollary 1 in Lounici

(2014) gives a bound on the Frobenius error that depends on the spectral decay of C. We will derive a slightly different (worst-case) bound under the assumption that the instances are bounded in the Euclidean unit ball. A comparison between the two bounds is given Appendix E.

Lemma 2 *Let $\epsilon \in (0,1)$. If $m \geq \left\lceil (d/r)^2 \cdot \frac{k}{\epsilon^2} \right\rceil$, then*

$$\mathbb{E}[\|C - \hat{C}\|_F] \leq \epsilon/\sqrt{k} \,.$$

Proof Using Jensen's inequality, we have

$$\mathbb{E}[\|\hat{C} - C\|_F] = \mathbb{E}[(\|\hat{C} - C\|_F^2)^{1/2}] \leq (\,\mathbb{E}[\|\hat{C} - C\|_F^2])^{1/2} \,.$$

Since the observations are i.i.d.,

$$\begin{aligned}\mathbb{E}[\|\hat{C} - C\|_F^2] &= \sum_{i,j} \mathrm{Var}[\hat{C}_{i,j}] = \sum_{i,j} \mathrm{Var}\left[(1/m)\sum_{q=1}^{m} \hat{C}_{\hat{x}_q,i,j}\right] \\ &= \frac{1}{m}\sum_{i,j} \mathrm{Var}[\hat{C}_{\hat{x}_1,i,j}] \leq \frac{1}{m}\sum_{i,j} \mathbb{E}[\hat{C}^2_{\hat{x}_1,i,j}] \,,\end{aligned}$$

where $\hat{C}_{\hat{x}_q,i,j}$ is the $[i,j]$-th entry of $\hat{C}_q$. Denote $\hat{x} = \hat{x}_1$ and $x = x_1$ (subscript indices will now correspond to the entries of these vectors). According to 4,

$$\mathbb{E}[\hat{C}^2_{\hat{x},i,j}|x] = \begin{cases} p^{-4}\,\mathbb{E}[\hat{x}_i^2\hat{x}_j^2|x] = p^{-2}x_i^2x_j^2 & i \neq j \\ p^{-2}\,\mathbb{E}[\hat{x}_i^4|x] = p^{-1}x_i^4 \leq p^{-2}x_i^4 & i = j \end{cases} .$$

Therefore, since the ℓ_2 norm of the instances is at most 1, we have

$$\sum_{i,j} \mathbb{E}[\hat{C}^2_{\hat{x}_1,i,j}|x] \leq p^{-2}\sum_{i,j} x_i^2x_j^2 = p^{-2}\|x\|^4 \leq p^{-2} = \frac{d^2}{r^2} \,,$$

and $\sum_{i,j} \mathbb{E}[\hat{C}^2_{\hat{x}_1,i,j}] \leq \frac{d^2}{r^2}$. Combining the above bounds, we obtain

$$\mathbb{E}[\|\hat{C} - C\|_F] \leq \frac{1}{\sqrt{m}} \cdot \frac{d}{r} \,.$$

For $m \geq \left\lceil \frac{d^2}{r^2} \cdot \frac{k}{\epsilon^2} \right\rceil$, we arrive at the claimed bound. ■

Let $m_p(d,k,r,\epsilon)$ be the sample complexity of subspace learning in the passive partial information model, namely, how many examples are needed (for the optimal learner) to guarantee that (2) holds. Based on Lemma 2 and Lemma 1, we now conclude the following bound on the sample complexity.

Corollary 3 *Using POPCA (Algorithm 1), we have the following bound on the sample complexity for any integer $r \geq 2$:*

$$m_p(d,k,r,\epsilon) \leq \left\lceil (d/r)^2 \cdot \frac{k}{\epsilon^2} \right\rceil \,.$$

2.3 Optimality of POPCA

In this section we prove the following lower bound on the sample complexity of subspace learning with partial information in the passive model.

Theorem 4 *Assume that $k \leq d/2$ and $\epsilon \in (0, 1/128)$. The sample complexity in the passive model is at least $\Omega\left((d/r)^2 \cdot \frac{k}{\epsilon^2}\right)$. Therefore, we have*

$$m_p(d,k,r,\epsilon) = \Theta\left((d/r)^2 \cdot \frac{k}{\epsilon^2}\right) \,.$$

Note that up to a constant factor, our lower bound coincides with the upper bound obtained by POPCA (Corollary 3). Therefore, Theorem 4 establishes the optimality of POPCA in the passive model.

The proof of Theorem 4 is divided into two parts. First, in Theorem 5 we prove that the sample complexity in the full-information setting is at least $\Omega(k/\epsilon^2)$. Then, we complete the proof of Theorem 4 by showing that the multiplicative price of partial information is at least $\Omega((d/r)^2)$.

Theorem 5 *Assume that $k \leq d/2$ and let $\epsilon \in (0, 1/128)$. The sample complexity of subspace learning with full information is bounded below by*

$$m(d,k,r=d,\epsilon) = \Omega(k/\epsilon^2) \,.$$

We now sketch the proof of Theorem 5. A detailed proof is provided in Appendix A.

Proof (sketch)
The idea is to reduce the problem of coin identification (see Section 5.2 in Anthony and Bartlett (2009)) to that of subspace learning. Assume that $k \leq d/2$ and $\epsilon \in (0, 1/128)$. Let $\mathcal{U} = \{u_j\}_{j=1}^{2k} \subseteq \mathbb{R}^d$ be a set of $2k$ orthonormal vectors. A distribution over $\mathcal{U}$ is defined as follows. First, we draw a sequence $b = (b_1, \ldots, b_k) \in \{-1,1\}^k$ uniformly at random. We associate the pair $\{u_i, u_{i+k}\}$ with a Bernoulli random variable B_i with parameter $p_i = \frac{1+b_i\alpha}{2}$, where $\alpha = 16\epsilon$. To define a distribution $P := P_b$, we now describe the process of drawing an instance $x \sim P$:

1. An integer $i \in [k]$ is chosen uniformly at random.
2. The i-th coin is flipped (according to p_i). Denote the corresponding random variable by Z.
3. If $Z = 1$, then x is chosen uniformly at random from the set $\{u_i, -u_i\}$. Otherwise ($Z = 0$), x is chosen uniformly at random from the set $\{u_{i+k}, -u_{i+k}\}$.

In Lemma 10 we show that a successful subspace learner must identify the bias of "most" of the coins. Thus, we can reduce k independent tasks of coin identification to the task of subspace learning. A well-known result in statistics (Anthony and Bartlett, 2009)[Lemma 5.1] tells us that $\Omega(1/\alpha^2)$ samples are needed to identify a coin with bias α. Hence, each of the pairs must be observed $\Omega(1/\epsilon^2)$ times. ■

Proof (of Theorem 4) Consider the construction presented in the proof (sketch) of Theorem 5. We next specify the set $\mathcal{U}$ and prove that the price of partial information in the passive model is $\Omega(d^2/r^2)$. Consequently, this will conclude the proof of Theorem 4.

For every $i \in [k]$, let $u_i = \frac{\sqrt{2}}{2}(e_i + e_{i+k})$, $u_{i+k} = \frac{\sqrt{2}}{2}(-e_i + e_{i+k})$. To specify a distribution P, fix a vector $b = (b_1, \ldots, b_k) \in \{-1, 1\}^k$. Consider now a single interaction between the learner and the environment. A vector x is drawn according to P as described in the proof (sketch) of Theorem 5. Let $i \in [k]$ denote the index of the coin which is associated with x. As we observed in Section 1, each of the coordinates i, $i + k$ is distributed uniformly over $\{-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\}$. Hence, to obtain any information, the learner must observe both of the coordinates i and $i + k$. The probability that both coordinates are observed is at most $O(r^2/d^2)$. Hence, in expectation, (only) a single "meaningful" observation is obtained every $\Omega(d^2/r^2)$ iterations. Therefore, the price of partial information is $\Omega(d^2/r^2)$. ■

2.4 Implementation of POPCA

As we discussed above, when the data is fully visible, the sample complexity of the PCA algorithm (which computes the k leading eigenvectors of the empirical covariance matrix, $m^{-1}\sum_{i=1}^m x_i x_i^\top$, and returns the corresponding projection matrix) is $m_f := O(k/\epsilon^2)$. When the number of samples, m_f, is larger than the dimension, a standard implementation of this algorithm costs $O(m_f d^2)$. We next show that POPCA has a similar runtime.

We established above that POPCA requires a training set of size $O((d/r)^2 m_f)$. Consider a single iteration of POPCA. Note that the construction of $\hat{C}_{\hat{x}_i}$ (see 4) costs $O(r^2)$. Therefore, it costs $O(m_f d^2)$ to obtain the average of the estimates, $\hat{C}$. The computation of the SVD of $\hat{C}$ costs $O(d^3)$. Therefore, when $m_f \geq d$, the overall runtime of POPCA is indeed $O(m_f d^2)$.

3. The Active Setting

We now consider the active model of subspace learning with partial information. In particular, we will present and analyze the Matrix Bandit Exponentiated Gradient (MBEG) algorithm.

Before describing MBEG, it should be noticed that POPCA (along with its analysis) can be modified to fit the active model; An active version of POPCA simply selects the r observed attributes uniformly at random (with replacement) and then proceeds similarly to POPCA[2]. It is not hard to verify that the sample complexity of this algorithm is asymptotically equivalent to the sample complexity of POPCA. In particular, the price of partial information is quadratic in d/r.

The main differences between MBEG and POPCA are as follows:

1. MBEG is designed for the active model. It employs a non-uniform sampling method which gives higher priority to "stronger" directions.

2. MBEG is an iterative algorithm which maintains a *weight matrix* that belongs to the convex hull of the set $\mathcal{P}_k^d$ of projection matrices. It can be thought of as a Bandit version of the extension of the Exponentiated Gradient (EG) algorithm to matrices. The EG algorithm, and its extension to matrices are due to Kivinen and Warmuth (1997), and Tsuda et al. (2005), respectively.

2. Note that here the attributes are no longer independent and therefore we should replace the weights in the definition of $\hat{C}_{\hat{x}}$.

3.1 Matrix Bandit Exponentiated Gradient (MBEG)

3.1.1 CONVEXIFICATION

In order to be able to apply the Matrix EG algorithm, we first need to formulate our task as a convex optimization problem. Recall that our problem is equivalent to approximately minimizing the objective $\operatorname{argmin}_{\Pi \in \mathcal{P}_k^d} \langle \Pi, -C \rangle$, where $C = \mathbb{E}[xx^\top]$. The objective is linear and thus convex. We will replace the non-convex set $\mathcal{P}_k^d$ with its convex hull, $\mathcal{C}_k^d := \text{conv}(\mathcal{P}_k^d)$. Note that for every W in $\mathcal{C}_k^d$, the gradient is given by $-C$. Therefore, the EG procedure would start with some $W_1 \in \mathcal{C}_k^d$, and, at iteration i, would update according to

$$\begin{aligned} &1.\quad U_{i+1} = \exp(\log(W_i) + \eta C) \\ &2.\quad W_{i+1} = \operatorname*{argmin}_{W \in \mathcal{C}_k^d} D_R(W, U_{i+1}) \,, \end{aligned} \tag{6}$$

where $D(R, U) = \text{tr}(W \log W - W \log U - W + U)$ is the Bregman divergence induced by the *quantum entropy* regularizer, $R(W) = \text{tr}(W \log W - W)$. Additional details regarding this regualarizer can be found in Tsuda et al. (2005) and Warmuth and Kuzmin (2008). As in POPCA, the gradient C is unknown but can be estimated. We would like to exploit the active setting, and therefore, we will employ a non-uniform sampling method which relies on the current weight matrix W_t (see Section 3.1.2).

Next, we recall that the output of the algorithm should be a projection matrix, while our algorithm maintains weight matrices, which may not belong to the set $\mathcal{P}_k^d$. Therefore, the final step of the algorithm is to construct an element from $\mathcal{P}_k^d$ that performs "similarly" to the average of the weight matrices maintained during the run of the algorithm. For this purpose, we rely on the following lemma, due to Warmuth and Kuzmin (2008):

Lemma 6 *Every matrix $W \in \mathcal{C}_k^d$ can be decomposed in time $O(d^3)$ into a convex combination of at most d elements from $\mathcal{P}_k^d$.*

For completeness, we recall the decomposition procedure of Warmuth and Kuzmin (2008) in Appendix D. Getting back to our algorithm, let $\hat{W} = \frac{1}{m}\sum_{i=1}^m W_i$ be the average weight matrix, and denote by $\hat{W} = \sum_{j=1}^d \beta_j \Pi_j$ a decomposition of $\hat{W}$ into a convex combination of elements from $\mathcal{P}_k^d$. The final step of MBEG sets the output matrix $\hat{\Pi}$ to be Π_j with probability β_j. This guarantees that the expected performance of $\hat{\Pi}$ is the same as the performance of $\hat{W}$.

3.1.2 PRIORITIZING "STRONGER" DIRECTIONS

In this part we present the non-uniform sampling mechanism which is employed by MBEG for attribute sampling. We first consider the case $r = 2$. Let $W := W_i$ be a weight matrix obtained during the run of MBEG. Recall[3] that $\sum_i W_{i,i} = k$, and for every $i \in [d]$, $W_{i,i} \in [0, 1]$. Therefore, a natural distribution for attribute sampling is to choose each pair $(s, q) \in [d]^2$ with probability $p_{s,q} = \frac{W_{s,s}}{k} \cdot \frac{W_{q,q}}{k}$. Unfortunately, we were not able to obtain a good bound using this sampling technique. Instead, we pick attributes according to

$$p_{s,q} = (1 - \alpha)\frac{W_{s,s} + W_{q,q}}{2dk} + \frac{\alpha}{d^2} \,, \tag{7}$$

3. These properties clearly hold for projection matrices, and thus hold for any convex combination of projection matrices.

for some parameter $\alpha \in (0, 1/2)$, which is tuned below. That is, we mix a uniform distribution over $[d]^2$, with a distribution which gives higher sampling probability to pairs for which $W_{s,s}, W_{q,q}$ are high, reflecting a bias toward sampling from "stronger" directions. Mixing with the uniform distribution guarantees that every pair has large enough probability to be sampled, which will later help us ensure that we perform enough "exploration".

Based on this probability distribution over pairs, we define an unbiased estimate of the matrix C by

$$\hat{C} = \frac{1}{2p_{s,q}} x_s x_q (E_{s,q} + E_{q,s}) \,, \tag{8}$$

where $E_{s,q}$ is the all zeros matrix except 1 in the i, j coordinate.

The extension of MBEG to any (even) $r > 2$ is straightforward. We simply pick $r/2$ independent estimates $\hat{C}_1, \ldots, \hat{C}_{r/2}$, each of which is constructed as in the case $r = 2$, and set $\hat{C} = \frac{2}{r} \sum_{j=1}^{r/2} \hat{C}_j$.

The algorithm is summarized in Algorithm 2. As explained in Tsuda et al. (2005), the projection step w.r.t. the Bregman divergence, $W_{i+1} = \operatorname{argmin}_{W \in \mathcal{C}_k^d} D_R(W, U_{i+1})$ can be performed in time $O(d^3)$. Overall, the running time per iteration is $O(d^3)$.

Algorithm 2 Matrix Bandit Exponentiated Gradient

Input: $r, k < d$ ($r \mod 2 = 0$)
$\eta = \sqrt{\frac{r \log(d)}{2m(d+r)}}$
$\alpha = \eta d^2$ (we assume that m is large enough so that $\alpha \leq 1/2$)
$W_1 = \frac{k}{d} I$
for $i = 1$ **to** m **do**
  denote $W = W_i$
  for $j = 1$ **to** $r/2$ **do**
    pick $(s, q) \in [d]^2$ with probability $p_{s,q} = (1 - \alpha) \frac{W_{s,s} + W_{q,q}}{2dk} + \frac{\alpha}{d^2}$
    for $x \sim P$, let (x_s, x_q) be the corresponding attributes
    $\hat{C}_{i,j} = \frac{1}{2p_{s,q}} x_s x_q (E_{s,q} + E_{q,s})$
  end for
  $\hat{C}_i = \frac{2}{r} \sum_{j=1}^{r/2} \hat{C}_{i,j}$
  $U_{i+1} = \exp(\log(W) + \eta \hat{C}_i)$
  $W_{i+1} = \operatorname{argmin}_{W \in \mathcal{C}_k^d} D_R(W, U_{i+1})$
end for
$\hat{W} = \frac{1}{m} \sum_{i=1}^{m} W_i$
decompose $\hat{W}$ into $\hat{W} = \sum_{j=1}^{d} \beta_j \Pi_j$ using Algorithm 3
return $\hat{\Pi} = \Pi_j$ with probability β_j

3.1.3 ANALYSIS OF MBEG

Let $m_a(d, k, r, \epsilon)$ be the sample complexity of subspace learning in the active partial information model. We denote the Frobenius norm, the spectral norm, and the trace norm by $\|\cdot\|_F$, $\|\cdot\|_{\mathrm{sp}}$, and $\|\cdot\|_{\mathrm{tr}}$, respectively. Throughout this section we prove the following result.

Theorem 7 *Using MBEG (Algorithm 2), we have the following bound on the sample complexity for any (even) integer $r \geq 2$:*

$$m_a(d, k, r, \epsilon) \leq \max \left\{ 8k \cdot \frac{d + r}{r} \cdot \frac{k}{\epsilon^2} \,,\ \frac{2d^4 r}{d + r} \right\} \cdot \log(d) \,.$$

In order to prove Theorem 7, we next apply the general analysis of Matrix EG (Hazan et al., 2012)[Theorem 13] to our case.

Theorem 8 *Assume that the sequence $\hat{C}_1, \ldots, \hat{C}_m$ obtained during the run of MBEG satisfies $\|\hat{C}_i\|_{\mathrm{sp}} \leq 1/\eta$ for every $i \in [m]$. Then, for every $\Pi^\star \in \mathcal{P}_k^d$,*

$$\sum_{i=1}^{m} \langle W_i - \Pi^\star, -\hat{C}_i \rangle \leq \frac{k \log(d)}{\eta} + \eta \sum_{i=1}^{m} \langle W_i, \hat{C}_i^2 \rangle \,. \tag{9}$$

The right-most term in 9 can be thought as the variance which is associated with the estimation process of MBEG. We now show that in contrast to POPCA, the variance scales only linearly with d/r.

Lemma 9 *For any matrix $W_i \in \mathcal{C}_k^d$ maintained by MBEG at time i, denote the conditional expectation given W_i by $\mathbb{E}_i$. Then,*

$$\mathbb{E}_i \langle W_i, \hat{C}_i^2 \rangle \leq \frac{2k(d + r)}{r} \,.$$

We now prove the lemma while assuming that $r = 2$. The extension to any $r > 2$ is detailed in Appendix B.

Proof Let $W = W_i, \hat{C} = \hat{C}_i$. Then,

$$\begin{aligned} \mathbb{E}_i \langle W, \hat{C}^2 \rangle &= \sum_{(s,q) \in [d]^2} p_{s,q} (W_{s,s} + W_{q,q}) \frac{1}{4p_{s,q}^2} x_s^2 x_q^2 \\ &= \sum_{(s,q) \in [d]^2} \frac{(W_{s,s} + W_{q,q})}{4p_{s,q}} x_s^2 x_q^2 \,. \end{aligned}$$

Since $\alpha \in (0, 1/2)$, we have

$$\frac{W_{s,s} + W_{q,q}}{p_{s,q}} = \frac{W_{s,s} + W_{q,q}}{(1 - \alpha) \frac{W_{s,s} + W_{q,q}}{2dk} + \alpha/d^2} \leq \frac{W_{s,s} + W_{q,q}}{(1 - \alpha) \frac{W_{s,s} + W_{q,q}}{2dk}} \leq 4dk \,.$$

Therefore, since $r = 2$, we obtain

$$\mathbb{E}_i \langle W, \hat{C}^2 \rangle \leq dk \sum_{(s,q) \in [d]^2} x_s^2 x_q^2 \leq dk < \frac{2k(d + r)}{r} \,,$$

completing the proof of the lemma. ■

Proof (of Theorem 7) By definition of $\hat{C}_{i,j}$ we have that $\hat{C}_{i,j}^2 = \hat{C}_{i,j}\hat{C}_{i,j}^\top$ is a diagonal matrix with all elements on the diagonal equal to zero except the (s,s) and (q,q) elements, which are both bounded above by $\frac{x_s^2 x_q^2}{p_{s,q}^2}$. It follows that

$$\|\hat{C}_{i,j}\|_{\mathrm{sp}} \leq \frac{|x_s x_q|}{p_{s,q}} \leq \frac{1}{p_{s,q}} \leq \frac{d^2}{\alpha} = \frac{1}{\eta} \,.$$

Hence,

$$\|\hat{C}_i\|_{\mathrm{sp}} \leq \frac{2}{r}\sum_{j=1}^{r/2} \|\hat{C}_{i,j}\|_{\mathrm{sp}} \leq \frac{2}{r}\frac{r}{2}\frac{1}{\eta} = \frac{1}{\eta} \,.$$

Therefore, the conditions of Theorem 8 hold. Taking expectation over 9, we obtain

$$\sum_{i=1}^{m} \mathbb{E}\left\langle W_i - \Pi^\star, -\hat{C}_i \right\rangle \leq \frac{k\log(d)}{\eta} + \eta \sum_{i=1}^{m} \mathbb{E}\left\langle W_i, \hat{C}_i^2 \right\rangle \,. \tag{10}$$

Let $\mathbb{E}_i$ denote the conditional expectation given W_i. Then, $\mathbb{E}_i[\hat{C}_i] = C$ and therefore, by the law of total expectation,

$$\mathbb{E}\left\langle W_i - \Pi^\star, -\hat{C}_i \right\rangle = \mathbb{E}\langle W_i - \Pi^\star, -C\rangle \,. \tag{11}$$

Combining 11 with Lemma 9, and plugging into 10, we obtain that

$$\mathbb{E}\sum_{i=1}^{m}\langle W_i - \Pi^\star, -C\rangle \leq \frac{k\log(d)}{\eta} + \eta m \frac{2k(d+r)}{r} \,.$$

Dividing by m, denoting $\hat{W} = \frac{1}{m}\sum_{i=1}^{m} W_i$, substituting $\eta = \sqrt{\frac{r\log(d)}{2(d+r)m}}$, rearranging terms, and observing that $\mathbb{E}[\hat{\Pi}|\hat{W}] = \hat{W}$, we have that

$$\mathbb{E}\left\langle \hat{\Pi} - \Pi^\star, -C\right\rangle = \left\langle \mathbb{E}\left[\hat{W}\right] - \Pi^\star, -C\right\rangle \leq \sqrt{\frac{8k^2(d+r)\log(d)}{rm}} \,.$$

The right-hand side of the above is smaller than ϵ if $m \geq 8k\log(d)\cdot\frac{d+r}{r}\cdot\frac{k}{\epsilon^2}$. Note, however, that we also require that m is large enough so that $\alpha \leq 1/2$. Since $\alpha = \eta d^2$, it follows that m should also satisfy $m \geq \frac{2d^4 r\log(d)}{d+r}$. ■

3.2 Comparison between the bounds in the passive and the active models

The left term in the bound of MBEG is smaller than the bound of POPCA if $\frac{d^2}{r^2} > 8k\log(d)\cdot\frac{d+r}{r}$. The right term in the bound of MBEG is smaller than the bound of POPCA if

$$\epsilon < \sqrt{\frac{(d+r)k}{2d^2 r^3\log(d)}} \,.$$

Hence, if both of the conditions hold, then the bound of MBEG is better than the bound of POPCA. Also, if $\epsilon < \frac{2(d+r)k}{d^2 r}$, then the dependence of the sample complexity of MBEG on d/r is linear.

A reasonable regime in which MBEG enjoys a linear price is when r and k are constants, and ϵ is proportional to $1/d$. Note that in this regime, the bound of POPCA scales with d^4, while the bound of MBEG scales only with d^3 (in the full-information setting, the sample complexity for this case scales with d^2). To summarize, ignoring the dependence on k (and logarithmic factors), MBEG attains the desired linear price when ϵ is 'small'.

4. Discussion

We introduced the problem of subspace learning with partial information, and considered both a passive and active model. Our first observation was that looking at a single coordinate does not give any information. Therefore, our algorithms look at the products of attribute pairs. Using the POPCA algorithm for the passive model, we showed that the sample complexity is tightly characterized by $\Theta((d/r)^2 k/\epsilon^2)$. Hence, the price of partial information in this case is quadratic. For the active model we introduced the MBEG algorithm which exploits the gathered information in order to make a better choice over which attributes to observe. We showed that if the desired accuracy ϵ is small, then MBEG achieves a linear price. Since the expected number of observed attributes from each vector is r, we can not hope for a better dependence on d/r. At this point, a natural question arises: can we attain a linear price of partial information in the active model, independently of the required accuracy? We conclude with an observation regarding MBEG that provides a partial answer to this question.

Assume that ϵ, r and k are constants. Examining the implementation of MBEG, we note that as long as the products, $x_s x_q$, between two consecutive observed attributes is zero, MBEG does not change the sampling distribution (which is initially uniform) nor its current estimation of the covariance matrix. Fix some attribute j and consider the distribution P_j which is concentrated on e_j. The probability that the product between two consecutive observed attributes is not zero is $(1/d)^2$. Since r is constant, only zero products are observed for at least $\Omega(d^2)$ iterations, implying the same bound on the sample complexity.

The implication of this result is that in order to achieve a linear price for any value of ϵ, it is necessary to extract more information from the partial observations (e.g., take into account both the products of attribute pairs and the single attributes). Developing such algorithms or alternatively, tightening the lower bounds, is left for future research.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. We also thank Amit Daniely for helpful discussions. This research has been supported by ISF no 1673/14.

Appendix A. Proof of Theorem 5

In this section we prove Theorem 5.

A.1 The adversarial distribution

Let $\mathcal{U} = \{u_j\}_{j=1}^{2k} \subseteq \mathbb{R}^d$ be a set of $2k$ orthonormal vectors. In the proof sketch of Theorem 5 we described a family $\mathcal{F}$ of distributions over the set $\{u, -u : u \in \mathcal{U}\}$. Our lower bounds will be proved to hold in expectation when choosing $P \in \mathcal{F}$ at random. According to Yao's minimax principle, such a result implies that the lower bound holds for some distribution in $\mathcal{F}$.

A.2 A Successful Subspace Learner Is Also a Successful Coin identifier

In this part we formalize the reduction from coin identification to subspace learning. As we sketched before, a key ingredient of our analysis of the lower bounds is the relation between subspace learning and "coin identification". This relation is formalized next. We first need to introduce some additional notation. To specify the distribution $P \in \mathcal{F}$, fix a vector $(b_1, \ldots, b_k) \in \{-1, 1\}^k$. Let $\hat{\Pi} = \sum_{i=1}^k \hat{u}_i \hat{u}_i^\top \in \mathcal{P}_k^d$ (where $\hat{u}_1, \ldots, \hat{u}_k$ are orthonormal). For every $(i, j) \in [k] \times [2k]$, define $\theta_{i,j} = |\langle \hat{u}_i, u_j \rangle|$ to be the *covariance* between $\hat{u}_i$ and u_j. Next, for each $j \in [2k]$, define $\theta_j^2 = \sum_{i=1}^k \theta_{i,j}^2$. Also, define the set

$$J = \{j \in [k] : b_j = 1 \wedge \theta_j^2 > \theta_{j+k}^2\} \cup \{j \in [k] : b_j = -1 \wedge \theta_j^2 < \theta_{j+k}^2\} \,. \tag{12}$$

For reasons that will become apparent shortly, we name J the set of identified coins. The following lemma asserts that a successful subspace learner must identify most of the coins.

Lemma 10 *Let $\epsilon \in (0, 1)$. Assume that $L(\hat{\Pi}) - \min_{\Pi \in \mathcal{P}_k^d} L(\Pi) \leq \epsilon$. Let J be defined as in 12. Then, $|J|/k > 1 - \frac{2\epsilon}{\alpha}$.*

Proof Assume w.l.o.g. that $b_1 = \ldots = b_k = 1$. Note that

$$\mathbb{E}[xx^\top] = \frac{1}{k} \sum_{j=1}^k \left(\frac{1+\alpha}{2} u_j u_j^\top + \frac{1-\alpha}{2} u_{j+k} u_{j+k}^\top \right) \,.$$

The optimal projection is obtained by picking the largest eigenvectors of $\mathbb{E}[xx^\top]$. Precisely, the optimal projection matrix is $\Pi^\star = \sum_{i=1}^k u_i u_i^\top$. We assume the loss function as formulated in 1.

The loss of $\hat{\Pi}$ is calculated as follows:

$$\begin{aligned}
\mathbb{E}[\|x\|_2^2] - L(\hat{\Pi}) &= \langle \sum_{i=1}^k \hat{u}_i \hat{u}_i^\top \,,\, \mathbb{E}[xx^\top] \rangle \\
&= \frac{1}{k} \sum_{i=1}^k \sum_{j=1}^k \left(\frac{1+\alpha}{2} \langle \hat{u}_i \hat{u}_i^\top, u_j u_j^\top \rangle + \frac{1-\alpha}{2} \langle \hat{u}_i \hat{u}_i^\top, u_{j+k} u_{j+k}^\top \rangle \right) \\
&= \frac{1}{k} \sum_{i=1}^k \sum_{j=1}^k \left(\frac{(1+\alpha)\theta_{i,j}^2}{2} + \frac{(1-\alpha)\theta_{i,j+k}^2}{2} \right) \\
&= \frac{1}{k} \sum_{j=1}^k \left(\frac{(1+\alpha)\theta_j^2}{2} + \frac{(1-\alpha)\theta_{j+k}^2}{2} \right) \\
&= \frac{1}{2k} \sum_{j=1}^{2k} \theta_j^2 + \frac{\alpha}{2k} \sum_{j=1}^k (\theta_j^2 - \theta_{j+k}^2) \,.
\end{aligned} \tag{13}$$

Since $\{u_j\}_{j=1}^{2k}$ is orthonormal, for each $i \in [k]$ we have $\sum_{j=1}^{2k} \theta_{i,j}^2 \leq 1$. Hence, $\sum_{j=1}^{2k} \theta_j^2 \leq k$, so that the second-to-last term of 13 is at most $\frac{1}{2}$. This value is attained by $\Pi^\star$, and for simplicity, we will assume that is attained by $\hat{\Pi}$ as well. For the same reasons, for each $i \in [k]$, we have $\sum_{j=1}^k (\theta_{i,j}^2 - \theta_{i,j+k}^2) \leq 1$. Hence, $\sum_{j=1}^k (\theta_j^2 - \theta_{j+k}^2) \leq k$, so that the last term of 13 is at most $\alpha/2$. Once again, this value is attained by $\Pi^\star$. Our last observation is that if the j-th coin is not identified, i.e., $j \notin J$, then $\theta_j^2 - \theta_{j+k}^2 \leq 0$. Combining these observations, we obtain that

$$L(\hat{\Pi}) - L(\Pi^\star) \geq \alpha(1 - |J|/k)/2 \,.$$

The proof is completed by combining the assumption that $L(\hat{\Pi}) - L(\Pi^\star) \leq \epsilon$. ■

A.2.1 LOWER BOUND ON COIN IDENTIFICATION

We previously informally argued that $\Omega(1/\alpha^2)$ samples are needed to identify a coin with bias $\frac{1\pm\alpha}{2}$. If we have k such independent coins, then $\Omega(k/\alpha^2)$ are needed. Let us formalize this result.

A coin identification problem with parameter α is defined as follows. Consider a binary classification problem with a domain $[k]$ and label set $\{0, 1\}$. The hypothesis class is the set $\mathcal{H} = \{0, 1\}^{[k]}$. The underlying distribution over $[k] \times \{0, 1\}$ is chosen at random using the following mechanism. The marginal distribution over $[k]$ is uniform, and the conditional probability over the label (coin) is determined by $P(y = 1 | x = j) = \frac{1+b_j\alpha}{2}$, where each b_j is an independent Rademacher random variable (drawn in advance). We observe that this distribution is identical to the distribution defined (over a shattered set of size k) in (Anthony and Bartlett, 2009, Theorem 5.2).

As usual, an algorithm for coin identification obtains an i.i.d. training sequence S according to P, and has to return a hypothesis $h \in \mathcal{H}$. The generalization error of $h \in \mathcal{H}$, denoted $\text{err}(h)$, is defined to be the probability that it misclassifies a new generated point (x, y), i.e., $\text{err}(h) = P(h(x) \neq y)$. The next theorem follows from (Anthony and Bartlett, 2009, Theorem 5.2).

Theorem 11 *Let $\mathcal{B}$ be an algorithm for coin identification, and let $\tilde{\epsilon} \in (0, 1/64)$. Consider a coin identification problem with $\alpha = 8\tilde{\epsilon}$. If $m < \frac{k}{8\tilde{\epsilon}^2}$, then there exists a distribution P for which*

$$\mathbb{E}_{S\sim P^m} \mathit{err}(\mathcal{B}(S)) - \min_{h\in\mathcal{H}} \mathit{err}(h) > \tilde{\epsilon} \,.$$

A.3 Concluding the Theorem

We are now in position to complete the reduction from the task of coin identification to the task of subspace learning, and consequently conclude Theorem 5.

Let $\mathcal{A}$ be a subspace learner whose sample complexity is $m(d, k, r = d, \epsilon)$. We describe an algorithm $\mathcal{B}$ for coin identification (with parameter α) which uses $\mathcal{A}$ as a subroutine. To this end, we shall provide a (randomized) map between the input of $\mathcal{B}$ to the input of $\mathcal{A}$. These inputs have the form of training sequences, which will be denoted by S_{subspace} and S_{coins}, respectively. For each $j \in [k]$, the pair $(j, 1)$ is associated with u_j or $-u_j$ with equal probability. The pair $(j, 0)$ is associated with u_{j+k} of $-u_{j+k}$ with equal probability. Clearly, the sequence provided to $\mathcal{A}$ is generated according to the distribution discussed in the proof sketch of Theorem 5. Given an accuracy parameter $\tilde{\epsilon} \in (0, 1/64)$ for $\mathcal{B}$, we will require accuracy $\epsilon = \tilde{\epsilon}/2$ from $\mathcal{A}$. To complete the reduction, we will specify how the output of $\mathcal{B}$ is determined using the output of $\mathcal{A}$.

For each $j \in [k]$, the output hypothesis returned by $\mathcal{B}$ is defined by

$$h(j) = \begin{cases} 1 & \theta_j > \theta_{j+k} \\ 0 & \text{otherwise} \end{cases} \,.$$

Denote the set of identified coins by J (as in Lemma 10). Observe that each coin that is not identified, adds α/k to the relative error of $\mathcal{B}$. It follows from Lemma 10 that

$$\begin{aligned} \text{err}(\mathcal{B}(S_{\text{coins}})) - \min_{h\in\mathcal{H}} \text{err}(h) &= \alpha(1 - |J|/k) \\ &\leq 2\left(L(\mathcal{A}(S_{\text{subspace}})) - \min_{\Pi\in\mathcal{P}_k^d} L(\Pi)\right) \,. \end{aligned} \tag{14}$$

If $m \geq m(d, k, r = d, \epsilon)$, then the right-hand side of 14 is at most $2\epsilon = \tilde{\epsilon}$. The proof is completed by applying Theorem 11 (with $\alpha = 8\tilde{\epsilon}$).

Appendix B. Extending Lemma 9 to r > 2

We will now extend the proof of Lemma 9 to any (even) $r > 2$. Fix an iteration i, and denote $\hat{C} = \hat{C}_i, W = W_i$. We have

$$\mathbb{E}_i\langle W, \hat{C}^2\rangle = \frac{4}{r^2}\left[\sum_{j=1}^{r/2} \mathbb{E}_i\langle W, \hat{C}_j^2\rangle + \sum_{j\neq t} \mathbb{E}_i\langle W, \hat{C}_j\hat{C}_t\rangle\right] \,.$$

From the case $r = 2$, we already know that the term $\mathbb{E}_i\langle W, \hat{C}_j^2\rangle$ is at most dk. Fix some pair $j \neq t$. Note that $\hat{C}_j\hat{C}_t \neq 0$ if and only if (at least) one of the indices s_j, q_j is equal to one of the indices

s_t, q_t. Hence,

$$\begin{aligned} \mathbb{E}_i\langle W, \hat{C}_j\hat{C}_t\rangle \leq 2 &\sum_{(s,q)\in[d]^2} p_{s,q}^2(W_{s,s} + W_{q,q})\frac{x_s^2 x_q^2}{4p_{s,q}^2} \\ &+ 4 \sum_{\substack{(s,q,q')\in[d]^3: \\ q\neq q'}} p_{s,q}p_{s,q'}W_{q,q'}\frac{x_s^2|x_q||x'_q|}{4p_{s,q}p_{s,q'}} \,. \end{aligned}$$

The first term can be bounded as follows:

$$\begin{aligned} 2\sum_{(s,q)\in[d]^2} p_{s,q}^2(W_{s,s} + W_{q,q})\frac{x_s^2x_q^2}{4p_{s,q}^2} &= \frac{2}{4}\sum_{(s,q)\in[d]^2}(W_{s,s} + W_{q,q})x_s^2x_q^2 \\ &= \frac{2}{4}\left\langle (W_{s,s} + W_{q,q})_{(s,q)\in[d]^2}, (x_s^2x_q^2)_{(s,q)\in[d]^2}\right\rangle \\ &\leq \frac{2}{4}\|(W_{s,s} + W_{q,q})_{(s,q)\in[d]^2}\|_\infty \|(x_s^2x_q^2)_{(s,q)\in[d]^2}\|_1 \\ &\leq \frac{2}{4}\cdot 2 = 1 \,. \end{aligned}$$

The second term can be bounded as:

$$\begin{aligned} 4\sum_{\substack{(s,q,q')\in[d]^3: \\ q\neq q'}} p_{s,q}p_{s,q'}W_{q,q'}\frac{x_s^2|x_q||x_{q'}|}{4p_{s,q}p_{s,q'}} &\leq 4\cdot\frac{1}{4}\left(\sum_{s\in[d]} x_s^2\right)\left(\sum_{(q,q')\in[d]^2} |W_{q,q'}|\cdot|x_qx_{q'}|\right) \\ &\leq \|W\|_{\text{tr}}\|xx^\top\|_{\text{sp}} \\ &\leq k \,. \end{aligned}$$

Combining the above, we obtain

$$\begin{aligned} \mathbb{E}_i\langle W, \hat{C}^2\rangle &= \frac{4}{r^2}\left[\frac{r}{2}dk + \frac{r}{2}\left(\frac{r}{2} - 1\right)(k+1)\right] \\ &\leq \frac{2dk}{r} + k + 1 \\ &\leq \frac{2dk}{r} + 2k \\ &= \frac{2k(d+r)}{r} \,. \end{aligned}$$

Appendix C. The Price of Partial Information is at Least Linear

Theorem 12 *Let $k = 1$. Then, the sample complexity of subspace learning is bounded below by:*

$$m(d, k = 1, r, \epsilon) \geq \Omega\left(\frac{d}{r\epsilon}\right) \,.$$

Proof Let $\epsilon \in (0, 1/4)$, and let $\mathcal{A}$ be a bandit subspace learner. For each $s \in [d]$, we define a distribution P_s as follows. The zero vector is drawn with probability $1 - c\epsilon$ (where $c > 2$), and the

vector e_s is drawn with probability $c\epsilon$. Note that $\mathbb{E}[xx^\top] = c\epsilon\, e_s e_s^\top$, and thus the optimal projection is given by $\Pi^\star := e_s e_s^\top$.

We next show that a successful subspace learner must identify the distribution. Let P_s be a concrete distribution. Denote by $\hat{\Pi} = \hat{u}\hat{u}^\top$ the output of the learner. Define $\theta_s = u_s^2$. It can be easily seen that if $L(\hat{\Pi}) - L(\Pi^\star) < \epsilon < c\epsilon/2$, then $\theta_s > \theta_q$ for any $q \neq s$. That is, a successful subspace learner must identify the index s.

Next we observe that if the size of the sample is at most $o\left(\frac{d}{r\epsilon}\right)$, then with a non-negligible probability, all the observations made by the learner are equal to zero. It follows that there exists a distribution P_j for which, $\mathbb{E}[L(\hat{\Pi}) - L(\Pi^\star)] > c\epsilon/2 > \epsilon$. ■

Appendix D. Decomposing Elements in $\mathcal{C}_k^d$ into a Convex Combination of Elements From $\mathcal{P}_k^d$

In this part we detail the decomposition procedure mentioned in Lemma 6. For a symmetric $d \times d$ matrix A, we denote its eigenvalues by $\lambda(A)_1 \geq \ldots \geq \lambda(A)_d$. For convenience, we define two subroutines. The procedure $\mathrm{sort}(x, s)$ returns a set of s indices, corresponding to the s largest values of x. Given a vector $x \in \mathbb{R}^d$, the function $\mathrm{diag} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ returns a diagonal matrix X with $X_{j,j} = x_j$.

Algorithm 3 Decomposition Procedure

Input: $W \in \mathcal{C}_k^d = \mathrm{conv}(\mathcal{P}_k^d)$
perform eigendecomposition $W = U\mathrm{diag}(\lambda(A))U^\top$
$\lambda := k^{-1}(\lambda(A)_1, \ldots, \lambda(A)_d)$
$i := 1$
repeat
  $J := \mathrm{sort}(\lambda, k)$
  $c_i := k^{-1}\sum_{j\in J} e_j$
  $s := \min_{j\in J} \lambda_j$
  $l := \max_{j\in[d]\setminus J} \lambda_j$
  $\beta_i := \min\left\{sk, \sum_{j=1}^d \lambda_j - lk\right\}$
  $\lambda := \lambda - \beta_i c_i$
  $\Pi_i := U\mathrm{diag}(kc_i)U^\top$
  $i := i + 1$
until $\lambda = (0, \ldots, 0)$
For each $j \in [i-1]$ choose $A = \Pi_j$ with probability β_j

It has been proved by Warmuth and Kuzmin (2008) that the loop inside Algorithm 3 repeats at most d times, and decomposes W into a convex combination $\sum \beta_i \Pi_i$ of elements from $\mathcal{P}_k^d$.

Appendix E. Covariance Estimation with Missing Entries

Corollary 1 in Lounici (2014) provides bounds under the assumption that the instances are drawn from a sub-gaussian distribution (see assumption 1 in Lounici (2014)). This assumption is weaker than our boundedness assumption. Translating the results to our notation yields:

Lemma 13 *Given m partial observations, let $\hat{C}$ be the unbiased estimation constructed by POPCA. Let $\lambda > 0$ be a parameter.*

$$\tilde{C} := C(\tilde{\lambda}) = \operatorname*{argmin}_{S\in\mathbb{S}_+^d}\{\|S - \hat{C}\|_F^2 + \lambda\|S\|_1\}\,,$$

where $\mathbb{S}_+^d$ is the set of symmetric positive semi-definite $d \times d$ matrices. Then, for a suitable choice of λ, with probability at least $1 - 1/(2d)$, we have

$$\|\tilde{C} - C\|_F \leq \sqrt{\inf_{S\in\mathbb{S}_+^d}\{\|S - C\|_F^2 + c_1\lambda_1^2(C)\cdot\frac{\mathrm{tr}(C)}{\lambda_1(C)}\cdot\frac{d^2}{r^2 m}\cdot \mathit{rank}(S)\cdot\log(2d)\}}\,,$$

where c_1 is a constant and $\lambda_1(C)$ is the leading eigenvalue of C.

Substituting $m = \lceil (d/r)^2 \cdot \frac{k}{\epsilon^2}\rceil$ from Lemma 2 into the above bound, we obtain that the RHS is at most

$$\sqrt{\inf_{S\in\mathbb{S}_+^d}\{\|S - C\|_F^2 + c_1\lambda_1^2(C)\cdot\frac{\mathrm{tr}(C)}{\lambda_1(C)}\cdot\frac{\epsilon^2}{k}\cdot\mathrm{rank}(S)\cdot\log(2d))\}}\,.$$

This bound is always worse than our bound (i.e., larger than $\epsilon/\sqrt{k}$).

References

Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

Raman Arora, Andrew Cotter, and Nathan Srebro. Stochastic optimization of pca with capped msg. *arXiv preprint arXiv:1307.1674*, 2013.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multi-armed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

Gilles Blanchard, Olivier Bousquet, and Laurent Zwald. Statistical properties of kernel principal component analysis. *Machine Learning*, 66(2-3):259–294, 2007.

Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

Nicolo Cesa-Bianchi, Shai Shalev-Shwartz, and Ohad Shamir. Efficient learning with partially observed attributes. *The Journal of Machine Learning Research*, 12:2857–2878, 2011.

Yudong Chen, Ali Jalali, Sujay Sanghavi, and Constantine Caramanis. Low-rank matrix recovery from errors and erasures. *Information Theory, IEEE Transactions on*, 59(7):4324–4337, 2013.

Yuejie Chi, Y.C. Eldar, and Robert Calderbank. Petrels: Parallel subspace estimation and tracking by recursive least squares from partial observations. *arXiv preprint arXiv:1207.6353*, 2013.

Qian Du and James E Fowler. Hyperspectral image compression using jpeg2000 and principal component analysis. *Geoscience and Remote Sensing Letters, IEEE*, 4(2):201–205, 2007.

Elad Hazan, Satyen Kale, and Shai Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. *arXiv preprint arXiv:1204.0136*, 2012.

Jyrki Kivinen and Manfred K Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

Karim Lounici. High-dimensional covariance matrix estimation with missing observations. *Bernoulli*, 20(3):1029–1058, 2014.

Ioannis Mitliagkas, Constantine Caramanis, and Prateek Jain. Streaming pca with many missing entries. *Preprint*, 2014.

Jiazhong Nie, Wojciech Kotłowski, and Manfred K Warmuth. Online pca with optimal regrets. In *Algorithmic Learning Theory*, pages 98–112. Springer, 2013.

Christos H Papadimitriou, Hisao Tamaki, Prabhakar Raghavan, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168. ACM, 1998.

John Shawe-Taylor, Christopher KI Williams, Nello Cristianini, and Jaz Kandola. On the eigenspectrum of the gram matrix and the generalization error of kernel-pca. *Information Theory, IEEE Transactions on*, 51(7):2510–2522, 2005.

Koji Tsuda, Gunnar Rätsch, and Manfred K Warmuth. Matrix exponentiated gradient updates for on-line learning and bregman projection. In *Journal of Machine Learning Research*, pages 995–1018, 2005.

Xiaodong Wang and H Vincent Poor. Blind multiuser detection: A subspace approach. *Information Theory, IEEE Transactions on*, 44(2):677–690, 1998.

M.K. Warmuth and D. Kuzmin. Randomized online pca algorithms with regret bounds that are logarithmic in the dimension. *Journal of Machine Learning Research*, 9:2287–2320, 2008.

Jian Yang, David Zhang, Alejandro F Frangi, and Jing-yu Yang. Two-dimensional pca: a new approach to appearance-based face representation and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(1):131–137, 2004.

Journal of Machine Learning Research 17 (2016) 1-38 Submitted 11/14; Revised 10/15; Published 4/16

Iterative Hessian Sketch: Fast and Accurate Solution Approximation for Constrained Least-Squares

Mert Pilanci MERT@BERKELEY.EDU
Department of Electrical Engineering and Computer Science
University of California
Berkeley, CA 94720-1776, USA

Martin J. Wainwright WAINWRIG@BERKELEY.EDU
Department of Electrical Engineering and Computer Science
Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Editor: Tong Zhang

Abstract

We study randomized sketching methods for approximately solving least-squares problem with a general convex constraint. The quality of a least-squares approximation can be assessed in different ways: either in terms of the value of the quadratic objective function (cost approximation), or in terms of some distance measure between the approximate minimizer and the true minimizer (solution approximation). Focusing on the latter criterion, our first main result provides a general lower bound on any randomized method that sketches both the data matrix and vector in a least-squares problem; as a surprising consequence, the most widely used least-squares sketch is sub-optimal for solution approximation. We then present a new method known as the *iterative Hessian sketch*, and show that it can be used to obtain approximations to the original least-squares problem using a projection dimension proportional to the statistical complexity of the least-squares minimizer, and a logarithmic number of iterations. We illustrate our general theory with simulations for both unconstrained and constrained versions of least-squares, including ℓ_1-regularization and nuclear norm constraints. We also numerically demonstrate the practicality of our approach in a real face expression classification experiment.

Keywords: Convex optimization, Random Projection, Lasso, Low-rank Approximation, Information Theory

1. Introduction

Over the past decade, the explosion of data volume and complexity has led to a surge of interest in fast procedures for approximate forms of matrix multiplication, low-rank approximation, and convex optimization. One interesting class of problems that arise frequently in data analysis and scientific computing are constrained least-squares problems. More specifically, given a data vector $y \in \mathbb{R}^n$, a data matrix $A \in \mathbb{R}^{n \times d}$ and a convex constraint set $\mathcal{C}$, a constrained least-squares problem can be written as follows

$$x^{\mathrm{LS}} := \arg\min_{x \in \mathcal{C}} f(x) \qquad \text{where } f(x) := \tfrac{1}{2n}\|Ax - y\|_2^2. \tag{1}$$

The simplest case is the unconstrained form ($\mathcal{C} = \mathbb{R}^d$), but this class also includes other interesting constrained programs, including those based ℓ_1-norm balls, nuclear norm balls, interval constraints $[-1, 1]^d$ and other types of regularizers designed to enforce structure in the solution.

Randomized sketches are a well-established way of obtaining an approximate solutions to a variety of problems, and there is a long line of work on their uses (e.g., see the books and papers by Vempala (2004); Boutsidis and Drineas (2009); Mahoney (2011); Drineas et al. (2011); Kane and Nelson (2014), as well as references therein). In application to problem (1), sketching methods involving using a random matrix $S \in \mathbb{R}^{m \times n}$ to project the data matrix A and/or data vector y to a lower dimensional space ($m \ll n$), and then solving the approximated least-squares problem. There are many choices of random sketching matrices; see Section 2.1 for discussion of a few possibilities. Given some choice of random sketching matrix S, the most well-studied form of sketched least-squares is based on solving the problem

$$\widetilde{x} := \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2n}\|SAx - Sy\|_2^2 \Big\}, \tag{2}$$

in which the data matrix-vector pair (A, y) are approximated by their sketched versions (SA, Sy). Note that the sketched program is an m-dimensional least-squares problem, involving the new data matrix $SA \in \mathbb{R}^{m \times d}$. Thus, in the regime $n \gg d$, this approach can lead to substantial computational savings as long as the projection dimension m can be chosen substantially less than n. A number of authors (e.g., Sarlos (2006); Boutsidis and Drineas (2009); Drineas et al. (2011); Mahoney (2011); Pilanci and Wainwright (2015a)) have investigated the properties of this sketched solution (2), and accordingly, we refer to to it as the *classical least-squares sketch*.

There are various ways in which the quality of the approximate solution $\widetilde{x}$ can be assessed. One standard way is in terms of the minimizing value of the quadratic cost function f defining the original problem (1), which we refer to as *cost approximation*. In terms of f-cost, the approximate solution $\widetilde{x}$ is said to be ε-optimal if

$$f(x^{\mathrm{LS}}) \;\leq\; f(\widetilde{x}) \;\leq\; (1+\varepsilon)^2 f(x^{\mathrm{LS}}). \tag{3}$$

For example, in the case of unconstrained least-squares ($\mathcal{C} = \mathbb{R}^d$) with $n > d$, it is known that with Gaussian random sketches, a sketch size $m \gtrsim \frac{1}{\varepsilon^2} d$ suffices to guarantee that $\widetilde{x}$ is ε-optimal with high probability (for instance, see the papers by Sarlos (2006) and Mahoney (2011), as well as references therein). Similar guarantees can be established for sketches based on sampling according to the statistical leverage scores (Drineas and Mahoney, 2010; Drineas et al., 2012). Sketching can also be applied to problems with constraints: Boutsidis and Drineas (2009) prove analogous results for the case of non-negative least-squares considering

JMLR 17(53):1-38

the sketch in equation (2), whereas our own past work (Pilanci and Wainwright, 2015a) provides sufficient conditions for ε-accurate cost approximation of least-squares problems over arbitrary convex sets based also on the form in (2).

It should be noted, however, that other notions of "approximation goodness" are possible. In many applications, it is the least-squares minimizer x^{LS} itself—as opposed to the cost value $f(x^{\mathrm{LS}})$—that is of primary interest. In such settings, a more suitable measure of approximation quality would be the ℓ_2-norm $\|\widetilde{x} - x^{\mathrm{LS}}\|_2$, or the prediction (semi)-norm

$$\|\widetilde{x} - x^{\mathrm{LS}}\|_A := \frac{1}{\sqrt{n}}\|A(\widetilde{x} - x^{\mathrm{LS}})\|_2. \tag{4}$$

We refer to these measures as *solution approximation*.

Now of course, a cost approximation bound (3) can be used to derive guarantees on the solution approximation error. However, it is natural to wonder whether or not, for a reasonable sketch size, the resulting guarantees are "good". For instance, using arguments from Drineas et al. (2011), for the problem of unconstrained least-squares, it can be shown that the same conditions ensuring a ε-accurate cost approximation also ensure that

$$\|\widetilde{x} - x^{\mathrm{LS}}\|_A \leq \varepsilon \sqrt{f(x^{\mathrm{LS}})}. \tag{5}$$

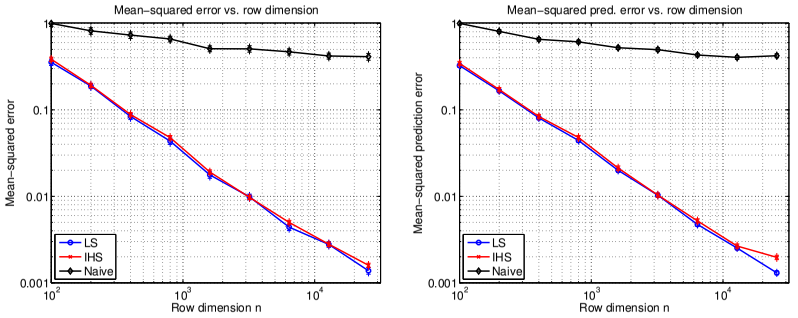
Given lower bounds on the singular values of the data matrix A, this bound also yields control of the ℓ_2-error.

In certain ways, the bound (5) is quite satisfactory: given our normalized definition (1) of the least-squares cost f, the quantity $f(x^{\mathrm{LS}})$ remains an order one quantity as the sample size n grows, and the multiplicative factor ε can be reduced by increasing the sketch dimension m. But how small should ε be chosen? In many applications of least-squares, each element of the response vector $y \in \mathbb{R}^n$ corresponds to an observation, and so as the sample size n increases, we expect that x^{LS} provides a more accurate approximation to some underlying population quantity, say $x^* \in \mathbb{R}^d$. As an illustrative example, in the special case of unconstrained least-squares, the accuracy of the least-squares solution x^{LS} as an estimate of x^* scales as $\|x^{\mathrm{LS}} - x^*\|_A \asymp \frac{\sigma^2 d}{n}$. Consequently, in order for our sketched solution to have an accuracy of the same order as the least-square estimate, we must set $\varepsilon^2 \asymp \frac{\sigma^2 d}{n}$. Combined with our earlier bound on the projection dimension, this calculation suggests that a projection dimension of the order

$$m \gtrsim \frac{d}{\varepsilon^2} \asymp \frac{n}{\sigma^2}$$

is required. This scaling is undesirable in the regime $n \gg d$, where the whole point of sketching is to have the sketch dimension m much lower than n.

Now the alert reader will have observed that the preceding argument was only rough and heuristic. However, the first result of this paper (Theorem 1) provides a rigorous confirmation of the conclusion: whenever $m \ll n$, the classical least-squares sketch (2) is sub-optimal as a method for solution approximation. Figure 1 provides an empirical demonstration of the poor behavior of the classical least-squares sketch for an unconstrained problem.



(a) (b)

Figure 1. Plots of mean-squared error versus the row dimension $n \in \{100, 200, 400, \ldots, 25600\}$ for unconstrained least-squares in dimension $d = 10$. The blue curves correspond to the error $x^{\mathrm{LS}} - x^*$ of the unsketched least-squares estimate. Red curves correspond to the IHS method applied for $N = 1 + \lceil \log(n) \rceil$ rounds using a sketch size $m = 7d$. Black curves correspond to the naive sketch applied using $M = Nm$ projections in total, corresponding to the same number used in all iterations of the IHS algorithm. (a) Error $\|\widetilde{x} - x^*\|_2^2$. (b) Prediction error $\|\widetilde{x} - x^*\|_A^2 = \frac{1}{n}\|A(\widetilde{x} - x^*)\|_2^2$. Each point corresponds to the mean taken over 300 trials with standard errors shown above and below in crosses.

This sub-optimality holds not only for unconstrained least-squares but also more generally for a broad class of constrained problems. Actually, Theorem 1 is a more general claim: *any estimator* based only on the pair (SA, Sy)—an infinite family of methods including the standard sketching algorithm as a particular case—is sub-optimal relative to the original least-squares estimator in the regime $m \ll n$. We are thus led to a natural question: can this sub-optimality be avoided by a different type of sketch that is nonetheless computationally efficient? Motivated by this question, our second main result (Theorem 2) is to propose an alternative method—known as the iterative Hessian sketch—and prove that it yields optimal approximations to the least-squares solution using a projection size that scales with the intrinsic dimension of the underlying problem, along with a logarithmic number of iterations. The main idea underlying iterative Hessian sketch is to obtain multiple sketches of the data $(S^1A, ..., S^NA)$ and iteratively refine the solution where N can be chosen logarithmic in n.

The remainder of this paper is organized as follows. In Section 2, we begin by introducing some background on classes of random sketching matrices, before turning to the statement of our lower bound (Theorem 1) on the classical least-squares sketch (2). We then introduce the Hessian sketch, and show that an iterative version of it can be used to compute ε-accurate solution approximations using $\log(1/\varepsilon)$-steps (Theorem 2). In Section 3, we illustrate the consequences of this general theorem for various specific classes of least-squares problems, and we conclude with a discussion in Section 4. The majority of our proofs are deferred to the appendices.

For the convenience of the reader, we summarize some standard notation used in this paper. For sequences $\{a_t\}_{t=0}^{\infty}$ and $\{b_t\}_{t=0}^{\infty}$, we use the notation $a_t \preceq b_t$ to mean that there is a constant (independent of t) such that $a_t \leq C\, b_t$ for all t. Equivalently, we write $b_t \succeq a_t$. We write $a_t \asymp b_t$ if $a_t \preceq b_t$ and $b_t \preceq a_t$.

2. Main results

In this section, we begin with background on different classes of randomized sketches, including those based on random matrices with sub-Gaussian entries, as well as those based on randomized orthonormal systems and random sampling. In Section 2.2, we prove a general lower bound on the solution approximation accuracy of any method that attempts to approximate the least-squares problem based on observing only the pair (SA, Sy). This negative result motivates the investigation of alternative sketching methods, and we begin this investigation by introducing the Hessian sketch in Section 2.3. It serves as the basic building block of the iterative Hessian sketch (IHS), which can be used to construct an iterative method that is optimal up to logarithmic factors.

2.1 Different types of randomized sketches

Various types of randomized sketches are possible, and we describe a few of them here. Given a sketching matrix S, we use $\{s_i\}_{i=1}^m$ to denote the collection of its n-dimensional rows. We restrict our attention to sketch matrices that are zero-mean, and that are normalized so that $\mathbb{E}[S^T S/m] = I_n$.

2.1.1 Sub-Gaussian sketches:

The most classical sketch is based on a random matrix $S \in \mathbb{R}^{m \times n}$ with i.i.d. standard Gaussian entries. A straightforward generalization is a random sketch with i.i.d. sub-Gaussian rows. In particular, a zero-mean random vector $s \in \mathbb{R}^n$ is 1-sub-Gaussian if for any $u \in \mathbb{R}^n$, we have

$$\mathbb{P}[\langle s,\, u\rangle \geq \varepsilon \|u\|_2] \leq e^{-\varepsilon^2/2} \qquad \text{for all } \varepsilon \geq 0. \tag{6}$$

For instance, a vector with i.i.d. $N(0,1)$ entries is 1-sub-Gaussian, as is a vector with i.i.d. Rademacher entries (uniformly distributed over $\{-1,+1\}$). Suppose that we generate a random matrix $S \in \mathbb{R}^{m \times n}$ with i.i.d. rows that are zero-mean, 1-sub-Gaussian, and with $\mathrm{cov}(s) = I_n$; we refer to any such matrix as a *sub-Gaussian sketch*. As will be clear, such sketches are the most straightforward to control from the probabilistic point of view. However, from a computational perspective, a disadvantage of sub-Gaussian sketches is that they require matrix-vector multiplications with unstructured random matrices. In particular, given an data matrix $A \in \mathbb{R}^{n \times d}$, computing its sketched version SA requires $\mathcal{O}(mnd)$ basic operations in general (using classical matrix multiplication).

2.1.2 Sketches based on randomized orthonormal systems (ROS):

The second type of randomized sketch we consider is *randomized orthonormal system* (ROS), for which matrix multiplication can be performed much more efficiently.

In order to define a ROS sketch, we first let $H \in \mathbb{R}^{n \times n}$ be an orthonormal matrix with entries $H_{ij} \in [-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$. Standard classes of such matrices are the Hadamard or Fourier bases, for which matrix-vector multiplication can be performed in $\mathcal{O}(n \log n)$ time via the fast Hadamard or Fourier transforms, respectively. Based on any such matrix, a sketching matrix $S \in \mathbb{R}^{m \times n}$ from a ROS ensemble is obtained by sampling i.i.d. rows of the form

$$s^T = \sqrt{n} e_j^T H D \qquad \text{with probability } 1/n \text{ for } j = 1, \ldots, n,$$

where the random vector $e_j \in \mathbb{R}^n$ is chosen uniformly at random from the set of all n canonical basis vectors, and $D = \mathrm{diag}(\nu)$ is a diagonal matrix of i.i.d. Rademacher variables $\nu \in \{-1,+1\}^n$. A similar sketching matrix can also be obtained by sampling canonical basis vectors without replacement. Given a fast routine for matrix-vector multiplication, the sketched data (SA, Sy) can be formed in $\mathcal{O}(n\, d \log m)$ time (for instance, see Ailon and Chazelle (2006)).

2.1.3 Sketches based on random row sampling:

Given a probability distribution $\{p_j\}_{j=1}^n$ over $[n] = \{1, \ldots, n\}$, another choice of sketch is to randomly sample the rows of the extended data matrix $\begin{bmatrix} A & y \end{bmatrix}$ a total of m times with replacement from the given probability distribution. Thus, the rows of S are independent and take on the values

$$s^T = \frac{e_j}{\sqrt{p_j}} \qquad \text{with probability } p_j \text{ for } j = 1, \ldots, n$$

where $e_j \in \mathbb{R}^n$ is the j^{th} canonical basis vector. Different choices of the weights $\{p_j\}_{j=1}^n$ are possible, including those based on the leverage values of A—i.e., $p_j \propto \|u_j\|_2$ for $j = 1, \ldots, n$, where $U \in \mathbb{R}^{n \times d}$ is the matrix of left singular vectors of A (e.g., see Drineas and Mahoney (2010)). In our analysis of lower bounds to follow, we assume that the weights are α-balanced, meaning that

$$\max_{j=1,\ldots,n} p_j \leq \frac{\alpha}{n} \tag{7}$$

for some constant α independent of n.

In the following section, we present a lower bound that applies to all the three kinds of sketching matrices described above.

2.2 Sub-optimality of classical least-squares sketch

We begin by proving a lower bound on any estimator that is a function of the pair (SA, Sy). In order to do so, we consider an ensemble of least-squares problems, namely those generated by a noisy observation model of the form

$$y = Ax^* + w, \qquad \text{where } w \sim N(0, \sigma^2 I_n), \tag{8}$$

the data matrix $A \in \mathbb{R}^{n \times d}$ is fixed, and the unknown vector x^* belongs to some set $\mathcal{C}_0$ that is star-shaped around zero.[1] In this case, the constrained least-squares estimate x^{LS} from

1. Explicitly, this star-shaped condition means that for any $x \in \mathcal{C}_0$ and scalar $t \in [0,1]$, the point tx also belongs to $\mathcal{C}_0$.

equation (1) corresponds to a constrained form of maximum-likelihood for estimating the unknown regression vector x^*. In Appendix D, we provide a general upper bound on the error $\mathbb{E}[\|x^{\text{LS}} - x^*\|_A^2]$ in the least-squares solution as an estimate of x^*. This result provides a baseline against which to measure the performance of a sketching method: in particular, our goal is to characterize the minimal projection dimension m required in order to return an estimate $\widetilde{x}$ with an error guarantee $\|\widetilde{x} - x^{\text{LS}}\|_A \approx \|x^{\text{LS}} - x^*\|_A$. The result to follow shows that unless $m \geq n$, then *any method* based on observing *only* the pair (SA, Sy) necessarily has a substantially larger error than the least-squares estimate. In particular, our result applies to an arbitrary measurable function $(SA, Sy) \mapsto x^\dagger$, which we refer to as an *estimator*.

More precisely, our lower bound applies to any random matrix $S \in \mathbb{R}^{m \times n}$ for which

$$\|\mathbb{E}\Big[S^T(SS^T)^{-1}S\Big]\|_{\text{op}} \leq \eta \, \frac{m}{n}, \tag{9}$$

where η is a constant independent of n and m, and $\|A\|_{\text{op}}$ denotes the ℓ_2-operator norm (maximum eigenvalue for a symmetric matrix). In Appendix A.1, we show that these conditions hold for various standard choices, including most of those discussed in the previous section. Letting $\mathbb{B}_A(1)$ denote the unit ball defined by the semi-norm $\|\cdot\|_A$, our lower bound also involves the complexity of the set $\mathcal{C}_0 \cap \mathbb{B}_A(1)$, which we measure in terms of its metric entropy. In particular, for a given tolerance $\delta > 0$, the δ-packing number M_δ of the set $\mathcal{C}_0 \cap \mathbb{B}_A(1)$ with respect to $\|\cdot\|_A$ is the largest number of vectors $\{x^j\}_{j=1}^M \subset \mathcal{C}_0 \cap \mathbb{B}_A(1)$ such that $\|x^j - x^k\|_A > \delta$ for all distinct pairs $j \neq k$.

With this set-up, we have the following result:

Theorem 1 (Sub-optimality) *For any random sketching matrix $S \in \mathbb{R}^{m \times n}$ satisfying condition* (9)*, any estimator $(SA, Sy) \mapsto x^\dagger$ has MSE lower bounded as*

$$\sup_{x^* \in \mathcal{C}_0} \mathbb{E}_{S,w}\big[\|x^\dagger - x^*\|_A^2\big] \geq \frac{\sigma^2}{128\,\eta} \, \frac{\log(\frac{1}{2}M_{1/2})}{\min\{m,n\}} \tag{10}$$

where $M_{1/2}$ is the 1/2-packing number of $\mathcal{C}_0 \cap \mathbb{B}_A(1)$ in the semi-norm $\|\cdot\|_A$.

The proof, given in Appendix A, is based on a reduction from statistical minimax theory combined with information-theoretic bounds. The lower bound is best understood by considering some concrete examples:

Example 1 (Sub-optimality for ordinary least-squares) *We begin with the simplest case—namely, in which $\mathcal{C} = \mathbb{R}^d$. With this choice and for any data matrix A with $\text{rank}(A) = d$, it is straightforward to show that the least-squares solution x^{LS} has its prediction mean-squared error at most*

$$\mathbb{E}\big[\|x^{LS} - x^*\|_A^2\big] \precsim \frac{\sigma^2 d}{n}. \tag{11a}$$

On the other hand, with the choice $\mathcal{C}_0 = \mathbb{B}_2(1)$, we can construct a 1/2-packing with $M = 2^d$ elements, so that Theorem 1 implies that any estimator $x^\dagger$ based on (SA, Sy) has its prediction MSE lower bounded as

$$\mathbb{E}_{S,w}\big[\|\widehat{x} - x^*\|_A^2\big] \succsim \frac{\sigma^2\, d}{\min\{m,n\}}. \tag{11b}$$

Consequently, the sketch dimension m must grow proportionally to n in order for the sketched solution to have a mean-squared error comparable to the original least-squares estimate. This is highly undesirable for least-squares problems in which $n \gg d$, since it should be possible to sketch down to a dimension proportional to $\text{rank}(A) = d$. Thus, Theorem 1 this reveals a surprising gap between the classical least-squares sketch (2) *and the accuracy of the original least-squares estimate.*

In contrast, the sketching method of this paper, known as iterative Hessian sketching (IHS), matches the optimal mean-squared error using a sketch of size $d + \log(n)$ in each round, and a total of $\log(n)$ rounds; see Corollary 2 for a precise statement. The red curves in Figure 1 show that the mean-squared errors ($\|\widehat{x} - x^*\|_2^2$ in panel (a), and $\|\widehat{x} - x^*\|_A^2$ in panel (b)) of the IHS method using this sketch dimension closely track the associated errors of the full least-squares solution (blue curves). Consistent with our previous discussion, both curves drop off at the n^{-1} rate.

Since the IHS method with $\log(n)$ rounds uses a total of $T = \log(n)\big\{d + \log(n)\big\}$ sketches, a fair comparison is to implement the classical method with T sketches in total. The black curves show the MSE of the resulting sketch: as predicted by our theory, these curves are relatively flat as a function of sample size n. Indeed, in this particular case, the lower bound (10)

$$\mathbb{E}_{S,w}\big[\|\widetilde{x} - x^*\|_A^2\big] \succsim \frac{\sigma^2 d}{m} \succsim \frac{\sigma^2}{\log^2(n)},$$

showing we can expect (at best) an inverse logarithmic drop-off. ♢

This sub-optimality can be extended to other forms of constrained least-squares estimates as well, such as those involving sparsity constraints.

Example 2 (Sub-optimality for sparse linear models) *We now consider the sparse variant of the linear regression problem, which involves the ℓ_0-"ball"*

$$\mathbb{B}_0(s) := \big\{x \in \mathbb{R}^d \mid \sum_{j=1}^d \mathbb{I}[x_j \neq 0] \leq s\big\},$$

corresponding to the set of all vectors with at most s non-zero entries. Fixing some radius $R \geq \sqrt{s}$, consider a vector $x^* \in \mathcal{C}_0 := \mathbb{B}_0(s) \cap \{\|x\|_1 = R\}$, and suppose that we make noisy observations of the form $y = Ax^* + w$.

Given this set-up, one way in which to estimate x^* is by by computing the least-squares estimate x^{LS} constrained[2] *to the ℓ_1-ball $\mathcal{C} = \{x \in \mathbb{R}^n \mid \|x\|_1 \leq R\}$. This estimator is a*

2. This set-up is slightly unrealistic, since the estimator is assumed to know the radius $R = \|x^*\|_1$. In practice, one solves the least-squares problem with a Lagrangian constraint, but the underlying arguments are basically the same.

form of the Lasso (Tibshirani, 1996): as shown in Appendix D.2, when the design matrix A satisfies the restricted isometry property (see Candes and Tao (2005) for a definition), then it has MSE at most

$$\mathbb{E}\big[\|x^{LS} - x^*\|_A^2\big] \precsim \frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{n}. \tag{12a}$$

On the other hand, the $\frac{1}{2}$-packing number M of the set $\mathcal{C}_0$ can be lower bounded as $\log M \succsim s \log\left(\frac{ed}{s}\right)$; see Appendix D.2 for the details of this calculation. Consequently, in application to this particular problem, Theorem 1 implies that any estimator $x^\dagger$ based on the pair (SA, Sy) has mean-squared error lower bounded as

$$\mathbb{E}_{w,S}\big[\|x^\dagger - x^*\|_A^2\big] \succsim \frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{\min\{m, n\}}. \tag{12b}$$

Again, we see that the projection dimension m must be of the order of n in order to match the mean-squared error of the constrained least-squares estimate x^{LS} up to constant factors. By contrast, in this special case, the sketching method developed in this paper matches the error $\|x^{LS} - x^*\|_2$ using a sketch dimension that scales only as $s \log\left(\frac{ed}{s}\right) + \log(n)$; see Corollary 3 for the details of a more general result. ♢

Example 3 (Sub-optimality for low-rank matrix estimation) *In the problem of multivariate regression, the goal is to estimate a matrix $X^* \in \mathbb{R}^{d_1 \times d_2}$ model based on observations of the form*

$$Y = AX^* + W, \tag{13}$$

where $Y \in \mathbb{R}^{n \times d_1}$ is a matrix of observed responses, $A \in \mathbb{R}^{n \times d_1}$ is a data matrix, and $W \in \mathbb{R}^{n \times d_2}$ is a matrix of noise variables. One interpretation of this model is as a collection of d_2 regression problems, each involving a d_1-dimensional regression vector, namely a particular column of X^*. In many applications, among them reduced rank regression, multi-task learning and recommender systems (e.g., Srebro et al. (2005); Yuan and Lin (2006); Negahban and Wainwright (2011); Bunea et al. (2011)), it is reasonable to model the matrix X^* as having a low-rank. Note a rank constraint on matrix X be written as an ℓ_0-"norm" constraint on its singular values: in particular, we have

$$\operatorname{rank}(X) \leq r \quad \text{if and only if} \quad \sum_{j=1}^{\min\{d_1, d_2\}} \mathbb{I}[\gamma_j(X) > 0] \leq r,$$

where $\gamma_j(X)$ denotes the j^{th} singular value of X. This observation motivates a standard relaxation of the rank constraint using the nuclear norm $\|X\|_{nuc} := \sum_{j=1}^{\min\{d_1, d_2\}} \gamma_j(X)$.

Accordingly, let us consider the constrained least-squares problem

$$X^{LS} = \arg\min_{X \in \mathbb{R}^{d_1 \times d_2}} \Big\{ \frac{1}{2}\|Y - AX\|_{fro}^2 \Big\} \qquad \text{such that } \|X\|_{nuc} \leq R, \tag{14}$$

where $\|\cdot\|_{fro}$ denotes the Frobenius norm on matrices, or equivalently the Euclidean norm on its vectorized version. Let $\mathcal{C}_0$ denote the set of matrices with rank $r < \frac{1}{2}\min\{d_1, d_2\}$, and Frobenius norm at most one. In this case, we show in Appendix D that the constrained least-squares solution X^{LS} satisfies the bound

$$\mathbb{E}\Big[\|X^{LS} - X^*\|_A^2\Big] \precsim \frac{\sigma^2 r\,(d_1 + d_2)}{n}. \tag{15a}$$

On the other hand, the $\frac{1}{2}$-packing number of the set $\mathcal{C}_0$ is lower bounded as $\log M \succsim r\big(d_1 + d_2\big)$, so that Theorem 1 implies that any estimator $X^\dagger$ based on the pair (SA, SY) has MSE lower bounded as

$$\mathbb{E}_{w,S}\big[\|X^\dagger - X^*\|_A^2\big] \succsim \frac{\sigma^2 r\big(d_1 + d_2\big)}{\min\{m, n\}}. \tag{15b}$$

As with the previous examples, we see the sub-optimality of the sketched approach in the regime $m < n$. In contrast, for this class of problems, our sketching method matches the error $\|X^{LS} - X^*\|_A$ using a sketch dimension that scales only as $\{r(d_1 + d_2) + \log(n)\}\log(n)$. See Corollary 4 for further details.

♢

2.3 Introducing the Hessian sketch

As will be revealed during the proof of Theorem 1, the sub-optimality is in part due to sketching the response vector—i.e., observing Sy instead of y. It is thus natural to consider instead methods that sketch *only* the data matrix A, as opposed to both the data matrix and data vector y. In abstract terms, such methods are based on observing the pair $\big(SA, A^T y\big) \in \mathbb{R}^{m \times d} \times \mathbb{R}^d$. One such approach is what we refer to as the *Hessian sketch*—namely, the sketched least-squares problem

$$\widehat{x} := \arg\min_{x \in \mathcal{C}} \Big\{ \underbrace{\frac{1}{2}\|SAx\|_2^2 - \langle A^T y,\, x\rangle}_{g_S(x)} \Big\}. \tag{16}$$

As with the classical least-squares sketch (2), the quadratic form is defined by the matrix $SA \in \mathbb{R}^{m \times d}$, which leads to computational savings. Although the Hessian sketch on its own does not provide an optimal approximation to the least-squares solution, it serves as the building block for an iterative method that can obtain an ε-accurate solution approximation in $\log(1/\varepsilon)$ iterations.

In controlling the error with respect to the least-squares solution x^{LS} the set of possible descent directions $\{x - x^{\mathrm{LS}} \mid x \in \mathcal{C}\}$ plays an important role. In particular, we define the *transformed tangent cone*

$$\mathcal{K}^{\mathrm{LS}} = \big\{ v \in \mathbb{R}^d \mid v = t\,A(x - x^{\mathrm{LS}}) \quad \text{for some } t \geq 0 \text{ and } x \in \mathcal{C} \big\}. \tag{17}$$

Note that the error vector $\widehat{v} := A(\widehat{x} - x^{\mathrm{LS}})$ of interest belongs to this cone. Our approximation bound is a function of the quantities

$$Z_1(S) := \inf_{v \in \mathcal{K}^{\mathrm{LS}} \cap \mathcal{S}^{n-1}} \frac{1}{m}\|Sv\|_2^2 \quad \text{and} \tag{18a}$$

$$Z_2(S) := \sup_{v \in \mathcal{K}^{\mathrm{LS}} \cap \mathcal{S}^{n-1}} \left| \langle u, \, (\frac{S^T S}{m} - I_n)\, v \rangle \right|, \tag{18b}$$

where u is a fixed unit-norm vector. These variables played an important role in our previous analysis (Pilanci and Wainwright, 2015a) of the classical sketch (2). The following bound applies in a deterministic fashion to any sketching matrix.

Proposition 1 (Bounds on Hessian sketch) *For any convex set $\mathcal{C}$ and any sketching matrix $S \in \mathbb{R}^{m \times n}$, the Hessian sketch solution $\widehat{x}$ satisfies the bound*

$$\|\widehat{x} - x^{LS}\|_A \leq \frac{Z_2}{Z_1} \|x^{LS}\|_A. \tag{19}$$

For random sketching matrices, Proposition 1 can be combined with probabilistic analysis to obtain high probability error bounds. For a given tolerance parameter $\rho \in (0, \frac{1}{2}]$, consider the "good event"

$$\mathcal{E}(\rho) := \left\{ Z_1 \geq 1 - \rho, \text{ and } Z_2 \leq \frac{\rho}{2} \right\}. \tag{20a}$$

Conditioned on this event, Proposition 1 implies that

$$\|\widehat{x} - x^{\mathrm{LS}}\|_A \leq \frac{\rho}{2\,(1-\rho)} \|x^{\mathrm{LS}}\|_A \;\leq\; \rho \|x^{\mathrm{LS}}\|_A, \tag{20b}$$

where the final inequality holds for all $\rho \in (0, 1/2]$.

Thus, for a given family of random sketch matrices, we need to choose the projection dimension m so as to ensure the event $\mathcal{E}\rho$ holds for some ρ. For future reference, let us state some known results for the cases of sub-Gaussian and ROS sketching matrices. We use (c_0, c_1, c_2) to refer to numerical constants, and we let $D = \dim(\mathcal{C})$ denote the dimension of the space $\mathcal{C}$. In particular, we have $D = d$ for vector-valued estimation, and $D = d_1 d_2$ for matrix problems.

Our bounds involve the "size" of the cone $\mathcal{K}^{\mathrm{LS}}$ previously defined (17), as measured in terms of its *Gaussian width*

$$\mathcal{W}(\mathcal{K}^{\mathrm{LS}}) := \mathbb{E}_g\big[\sup_{v \in \mathcal{K}^{\mathrm{LS}} \cap \mathbb{B}_2(1)} |\langle g, \, v \rangle| \big], \tag{21}$$

where $g \sim N(0, I_n)$ is a standard Gaussian vector. With this notation, we have the following:

Lemma 1 (Sufficient conditions on sketch dimension (Pilanci and Wainwright, 2015a))

(a) For sub-Gaussian sketch matrices, given a sketch size $m > \frac{c_0}{\rho^2}\mathcal{W}^2(\mathcal{K}^{LS})$, we have

$$\mathbb{P}\big[\mathcal{E}(\rho)\big] \geq 1 - c_1 e^{-c_2 m \delta^2}. \tag{22a}$$

(b) For randomized orthogonal system (ROS) sketches (sampled with replacement) over the class of self-bounding cones, given a sketch size $m > \frac{c_0 \, \log^4(D)}{\rho^2}\mathcal{W}^2(\mathcal{K}^{LS})$, we have

$$\mathbb{P}\big[\mathcal{E}(\rho)\big] \geq 1 - c_1 e^{-c_2 \frac{m\rho^2}{\log^4(D)}}. \tag{22b}$$

The class of self-bounding cones is described more precisely in Lemma 8 of our earlier paper (Pilanci and Wainwright, 2015a). It includes among other special cases the cones generated by unconstrained least-squares (Example 1), ℓ_1-constrained least squares (Example 2), and least squares with nuclear norm constraints (Example 3). For these cones, given a sketch size $m > \frac{c_0 \, \log^4(D)}{\rho^2}\mathcal{W}^2(\mathcal{K}^{\mathrm{LS}})$, the Hessian sketch applied with ROS matrices is guaranteed to return an estimate $\widehat{x}$ such that

$$\|\widehat{x} - x^{\mathrm{LS}}\|_A \leq \rho \|x^{\mathrm{LS}}\|_A \tag{23}$$

with high probability. More recent work by Bourgain et al. (2015) has established sharp bounds for various forms of sparse Johnson-Lindenstrauss transforms (Kane and Nelson, 2014). As a corollary of their results, a form of the guarantee (23) also holds for such random projections.

Returning to the main thread, the bound (23) is an analogue of our earlier bound (5) for the classical sketch with $\sqrt{f(x^{\mathrm{LS}})}$ replaced by $\|x^{\mathrm{LS}}\|_A$. For this reason, we see that the Hessian sketch alone suffers from the same deficiency as the classical sketch: namely, it will require a sketch size $m \asymp n$ in order to mimic the $\mathcal{O}(n^{-1})$ accuracy of the least-squares solution.

2.4 Iterative Hessian sketch

Despite the deficiency of the Hessian sketch itself, it serves as the building block for an novel scheme—known as the iterative Hessian sketch—that can be used to match the accuracy of the least-squares solution using a reasonable sketch dimension. Let begin by describing the underlying intuition. As summarized by the bound (20b), conditioned on the good event $\mathcal{E}(\rho)$, the Hessian sketch returns an estimate with error within a ρ-factor of $\|x^{\mathrm{LS}}\|_A$, where x^{LS} is the solution to the original unsketched problem. As show by Lemma 1, as long as the projection dimension m is sufficiently large, we can ensure that $\mathcal{E}(\rho)$ holds for some $\rho \in (0, 1/2)$ with high probability. Accordingly, given the current iterate x^t, *suppose that we can construct a new least-squares problem* for which the optimal solution is $x^{\mathrm{LS}} - x^t$. Applying the Hessian sketch to this problem will then produce a new iterate x^{t+1} whose distance to x^{LS} has been reduced by a factor of ρ. Repeating this procedure N times will reduce the initial approximation error by a factor ρ^N.

With this intuition in place, we now turn a precise formulation of the *iterative Hessian sketch*. Consider the optimization problem

$$\widehat{u} = \arg\min_{u \in \mathcal{C} - x^t} \Big\{ \frac{1}{2}\|Au\|_2^2 - \langle A^T(y - Ax^t), u \rangle \Big\}, \tag{24}$$

where x^t is the iterate at step t. By construction, the optimum to this problem is given by $\widehat{u} = x^{\mathrm{LS}} - x^t$. We then apply to Hessian sketch to this optimization problem (24) in order to obtain an approximation $x^{t+1} = x^t + \widehat{u}$ to the original least-squares solution x^{LS} that is more accurate than x^t by a factor $\rho \in (0, 1/2)$. Recursing this procedure yields a sequence of iterates whose error decays geometrically in ρ.

Formally, the iterative Hessian sketch algorithm takes the following form:

Iterative Hessian sketch (IHS): Given an iteration number $N \geq 1$:

(1) Initialize at $x^0 = 0$.

(2) For iterations $t = 0, 1, 2, \ldots, N-1$, generate an independent sketch matrix $S^{t+1} \in \mathbb{R}^{m \times n}$, and perform the update

$$x^{t+1} = \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2m}\|S^{t+1}A(x - x^t)\|_2^2 - \langle A^T(y - Ax^t), x \rangle \Big\}. \tag{25}$$

(3) Return the estimate $\widehat{x} = x^N$.

The following theorem summarizes the key properties of this algorithm. It involves the sequence $\{Z_1(S^t), Z_2(S^t)\}_{t=1}^N$, where the quantities Z_1 and Z_2 were previously defined in equations (18a) and (18b). In addition, as a generalization of the event (20a), we define the sequence of "good" events

$$\mathcal{E}^t(\rho) := \left\{ Z_1(S^t) \geq 1 - \rho, \text{ and } Z_2(S^t) \leq \frac{\rho}{2} \right\} \qquad \text{for } t = 1, \ldots, N. \tag{26}$$

With this notation, we have the following guarantee:

Theorem 2 (Guarantees for iterative Hessian sketch) *The final solution $\widehat{x} = x^N$ satisfies the bound*

$$\|\widehat{x} - x^{LS}\|_A \leq \Big\{ \prod_{t=1}^N \frac{Z_2(S^t)}{Z_1(S^t)} \Big\} \|x^{LS}\|_A. \tag{27a}$$

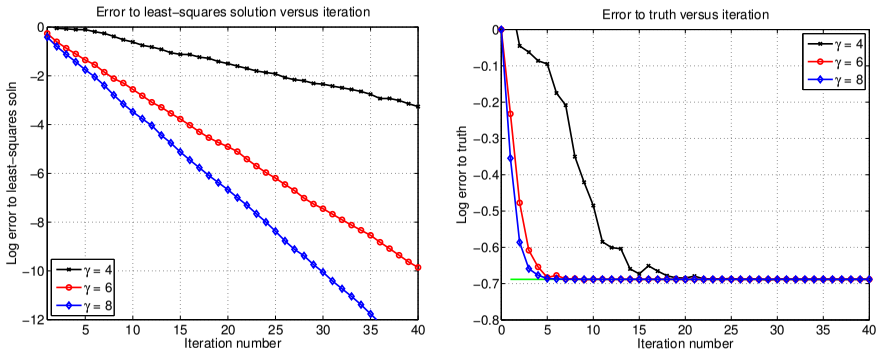
Consequently, conditioned on the event $\cap_{t=1}^N \mathcal{E}^t(\rho)$ for some $\rho \in (0, 1/2)$, we have

$$\|\widehat{x} - x^{LS}\|_A \leq \rho^N \, \|x^{LS}\|_A. \tag{27b}$$

Note that for any $\rho \in (0, 1/2)$, then event $\mathcal{E}^t(\rho)$ implies that $\frac{Z_2(S^t)}{Z_1(S^t)} \leq \rho$, so that the bound (27b) is an immediate consequence of the product bound (27a).

Lemma 1 can be combined with the union bound in order to ensure that the compound event $\cap_{t=1}^N \mathcal{E}^t(\rho)$ holds with high probability over a sequence of N iterates, as long as the sketch size is lower bounded as $m \geq \frac{c_0}{\rho^2}\mathcal{W}^2(\mathcal{K}^{\mathrm{LS}}) \log^4(D) + \log N$. Based on the bound (27b), we then expect to observe geometric convergence of the iterates.

In order to test this prediction, we implemented the IHS algorithm using Gaussian sketch matrices, and applied it to an unconstrained least-squares problem based on a data matrix with dimensions $(d, n) = (200, 6000)$ and noise variance $\sigma^2 = 1$. As shown in Appendix D.2, the Gaussian width of $\mathcal{K}^{\mathrm{LS}}$ is proportional to d, so that Lemma 1 shows that it suffices to choose a projection dimension $m \gtrsim \gamma d$ for a sufficiently large constant γ. Panel (a) of Figure 2 illustrates the resulting convergence rate of the IHS algorithm, measured in terms of the error $\|x^t - x^{\mathrm{LS}}\|_A$, for different values $\gamma \in \{4, 6, 8\}$. As predicted by Theorem 2, the convergence rate is geometric (linear on the log scale shown), with the rate increasing as the parameter γ is increased.



(a) (b)

Figure 2. Simulations of the IHS algorithm for an unconstrained least-squares problem with noise variance $\sigma^2 = 1$, and of dimensions $(d, n) = (200, 6000)$. Simulations based on sketch sizes $m = \gamma d$, for a parameter $\gamma > 0$ to be set. (a) Plots of the log error $\|x^t - x^{\mathrm{LS}}\|_A$ versus the iteration number t. Three different curves for $\gamma \in \{4, 6, 8\}$. Consistent with the theory, the convergence is geometric, with the rate increasing as the sampling factor γ is increased. (b) Plots of the log error $\|x^t - x^*\|_A$ versus the iteration number t. Three different curves for $\gamma \in \{4, 6, 8\}$. As expected, all three curves flatten out at the level of the least-squares error $\|x^{\mathrm{LS}} - x^*\|_A = 0.20 \approx \sqrt{\sigma^2 d/n}$.

Assuming that the sketch dimension has been chosen to ensure geometric convergence, Theorem 2 allows us to specify, for a given target accuracy $\varepsilon \in (0,1)$, the number of iterations required.

Corollary 1 *Fix some $\rho \in (0,1/2)$, and choose a sketch dimension $m > \frac{c_0 \log^4(D)}{\rho^2} \mathcal{W}^2(\mathcal{K}^{LS})$. If we apply the IHS algorithm for $N(\rho,\varepsilon) := 1 + \frac{\log(1/\varepsilon)}{\log(1/\rho)}$ steps, then the output $\widehat{x} = x^N$ satisfies the bound*

$$\frac{\|\widehat{x} - x^{LS}\|_A}{\|x^{LS}\|_A} \leq \varepsilon \tag{28}$$

with probability at least $1 - c_1 N(\rho,\varepsilon) e^{-c_2 \frac{m\rho^2}{\log^4(D)}}$.

This corollary is an immediate consequence of Theorem 2 combined with Lemma 1, and it holds for both ROS and sub-Gaussian sketches. (In the latter case, the additional $\log(D)$ terms may be omitted.) Combined with bounds on the width function $\mathcal{W}(\mathcal{K}^{\text{LS}})$, it leads to a number of concrete consequences for different statistical models, as we illustrate in the following section.

One way to understand the improvement of the IHS algorithm over the classical sketch is as follows. Fix some error tolerance $\varepsilon \in (0,1)$. Disregarding logarithmic factors, our previous results (Pilanci and Wainwright, 2015a) on the classical sketch then imply that a sketch size $m \gtrsim \varepsilon^{-2}\, \mathcal{W}^2(\mathcal{K}^{\text{LS}})$ is sufficient to produce a ε-accurate solution approximation. In contrast, Corollary 1 guarantees that a sketch size $m \gtrsim \log(1/\varepsilon)\, \mathcal{W}^2(\mathcal{K}^{\text{LS}})$ is sufficient. Thus, the benefit is the reduction from ε^{-2} to $\log(1/\varepsilon)$ scaling of the required sketch size.

It is worth noting that in the absence of constraints, the least-squares problem reduces to solving a linear system, so that alternative approaches are available. For instance, one can use a randomized sketch to obtain a preconditioner, which can then be used within the conjugate gradient method. As shown in past work (Rokhlin and Tygert, 2008; Avron et al., 2010), two-step methods of this type can lead to same reduction of ε^{-2} dependence to $\log(1/\varepsilon)$. However, a method of this type is very specific to unconstrained least-squares, whereas the procedure described in this paper is generally applicable to least-squares over any compact, convex constraint set.

2.5 Computational and space complexity

Let us now make a few comments about the computational and space complexity of implementing the IHS algorithm using the fast Johnson-Lindenstrauss (ROS) sketches, such as those based on the fast Hadamard transform. For a given sketch size m, the IHS algorithm requires $\mathcal{O}(nd\log(m))$ basic operations to compute the data sketch $S^{t+1}A$ at iteration t; in addition, it requires $\mathcal{O}(nd)$ operations to compute $A^T(y - Ax^t)$. Consequently, if we run the algorithm for N iterations, then the overall complexity scales as

$$\mathcal{O}\Big(N\,\big(nd\log(m) + C(m,d)\big)\Big), \tag{29}$$

where $C(m,d)$ is the complexity of solving the $m \times d$ dimensional problem in the update (25). Also note that, in problems where the data matrix A is sparse, $S^{t+1}A$ can be computed in time proportional to the number of non-zero elements in A using Gaussian sketching matrices. The space used by the sketches SA scales as $\mathcal{O}(md)$. To be clear, note that the IHS algorithm also requires access to the data via matrix-vector multiplies for forming $A^T(y - Ax^t)$. In limited memory environments, computing matrix-vector multiplies is considerably easier via distributed or interactive computation. For example, they can be efficiently implemented for multiple large datasets which can be loaded to memory only one at a time.

If we want to obtain estimates with accuracy ε, then we need to perform $N \asymp \log(1/\varepsilon)$ iterations in total. Moreover, for ROS sketches, we need to choose $m \gtrsim \mathcal{W}^2(\mathcal{K}^{\text{LS}})\log^4(d)$. Consequently, it only remains to bound the Gaussian width $\mathcal{W}$ in order to specify complexities that depend only on the pair (n,d), and properties of the solution x^{LS}.

For an unconstrained problem with $n > d$, the Gaussian width can be bounded as $\mathcal{W}^2(\mathcal{K}^{\text{LS}}) \precsim d$, and the complexity of the solving the sub-problem (25) can be bounded as d^3. Thus, the overall complexity of computing an ε-accurate solution scales as $\mathcal{O}(nd\log(d) + d^3)\log(1/\varepsilon)$, and the space required is $\mathcal{O}(d^2)$.

As will be shown in Section 3.2, in certain cases, the cone $\mathcal{K}^{\text{LS}}$ can have substantially lower complexity than the unconstrained case. For instance, if the solution is sparse, say with s non-zero entries and the least-squares program involves an ℓ_1-constraint, then we have $\mathcal{W}^2(\mathcal{K}^{\text{LS}}) \precsim s\log d$. Using a standard interior point method to solve the sketched problem, the total complexity for obtaining an ε-accurate solution is upper bounded by $\mathcal{O}((nd\log(s) + s^2 d\log^2(d))\log(1/\varepsilon))$. Although the sparsity s is not known a priori, there are bounds on it that can be computed in $\mathcal{O}(nd)$ time (for instance, see Ghaoui et al. (2011)).

3. Consequences for concrete models

In this section, we derive some consequences of Corollary 1 for particular classes of least-squares problems. Our goal is to provide empirical confirmation of the sharpness of our theoretical predictions, namely the minimal sketch dimension required in order to match the accuracy of the original least-squares solution.

3.1 Unconstrained least squares

We begin with the simplest case, namely the unconstrained least-squares problem ($\mathcal{C} = \mathbb{R}^d$). For a given pair (n,d) with $n > d$, we generated a random ensemble of least-square problems according to the following procedure:

- first, generate a random data matrix $A \in \mathbb{R}^{n\times d}$ with i.i.d. $N(0,1)$ entries
- second, choose a regression vector x^* uniformly at random from the sphere $\mathcal{S}^{d-1}$
- third, form the response vector $y = Ax^* + w$, where $w \sim N(0, \sigma^2 I_n)$ is observation noise with $\sigma = 1$.

As discussed following Lemma 1, for this class of problems, taking a sketch dimension $m \gtrsim \frac{d}{\rho^2}$ guarantees ρ-contractivity of the IHS iterates with high probability. Consequently, we can

obtain a ε-accurate approximation to the original least-squares solution by running roughly $\log(1/\varepsilon)/\log(1/\rho)$ iterations.

Now how should the tolerance ε be chosen? Recall that the underlying reason for solving the least-squares problem is to approximate x^*. Given this goal, it is natural to measure the approximation quality in terms of $\|x^t - x^*\|_A$. Panel (b) of Figure 2 shows the convergence of the iterates to x^*. As would be expected, this measure of error levels off at the ordinary least-squares error

$$\|x^{\mathrm{LS}} - x^*\|_A^2 \asymp \frac{\sigma^2 d}{n} \approx 0.10.$$

Consequently, it is reasonable to set the tolerance parameter proportional to $\sigma^2 \frac{d}{n}$, and then perform roughly $1 + \frac{\log(1/\varepsilon)}{\log(1/\rho)}$ steps. The following corollary summarizes the properties of the resulting procedure:

Corollary 2 *For some given $\rho \in (0, 1/2)$, suppose that we run the IHS algorithm for*

$$N = 1 + \lceil \frac{\log \sqrt{n}\, \frac{\|x^{LS}\|_A}{\sigma}}{\log(1/\rho)} \rceil$$

iterations using $m = \frac{c_0}{\rho^2} d$ projections per round. Then the output $\widehat{x}$ satisfies the bounds

$$\|\widehat{x} - x^{LS}\|_A \leq \sqrt{\frac{\sigma^2 d}{n}}, \qquad \text{and} \qquad \|x^N - x^*\|_A \leq \sqrt{\frac{\sigma^2 d}{n}} + \|x^{LS} - x^*\|_A \tag{30}$$

with probability greater than $1 - c_1 N e^{-c_2 \frac{m\rho^2}{\log^4(d)}}$.

In order to confirm the predicted bound (30) on the error $\|\widehat{x} - x^{\mathrm{LS}}\|_A$, we performed a second experiment. Fixing $n = 100d$, we generated $T = 20$ random least squares problems from the ensemble described above with dimension d ranging over $\{32, 64, 128, 256, 512\}$. By our previous choices, the least-squares estimate should have error $\|x^{\mathrm{LS}} - x^*\|_2 \approx \sqrt{\frac{\sigma^2 d}{n}} = 0.1$ with high probability, independently of the dimension d. This predicted behavior is confirmed by the blue bars in Figure 3; the bar height corresponds to the average over $T = 20$ trials, with the standard errors also marked. On these same problem instances, we also ran the IHS algorithm using $m = 6d$ samples per iteration, and for a total of

$$N = 1 + \lceil \frac{\log\left(\sqrt{\frac{n}{d}}\right)}{\log 2} \rceil \; = 4 \qquad \text{iterations.}$$

Since $\|x^{\mathrm{LS}} - x^*\|_A \asymp \sqrt{\frac{\sigma^2 d}{n}} \approx 0.10$, Corollary 2 implies that with high probability, the sketched solution $\widehat{x} = x^N$ satisfies the error bound

$$\|\widehat{x} - x^*\|_2 \leq c_0' \sqrt{\frac{\sigma^2 d}{n}}$$

for some constant $c_0' > 0$. This prediction is confirmed by the green bars in Figure 3, showing that $\|\widehat{x} - x^*\|_A \approx 0.11$ across all dimensions. Finally, the red bars show the results of running the classical sketch with a sketch dimension of $(6 \times 4)d = 24d$ sketches, corresponding to the total number of sketches used by the IHS algorithm. Note that the error is roughly twice as large.

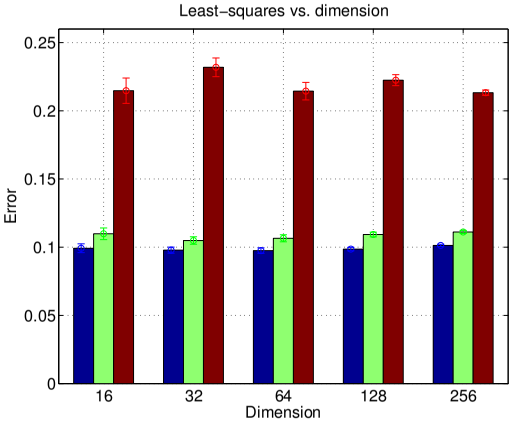


Figure 3. Simulations of the IHS algorithm for unconstrained least-squares. In these experiments, we generated random least-squares problem of dimensions $d \in \{16, 32, 64, 128, 256\}$, on all occasions with a fixed sample size $n = 100d$. The initial least-squares solution has error $\|x^{\mathrm{LS}} - x^*\|_A \approx 0.10$, as shown by the blue bars. We then ran the IHS algorithm for $N = 4$ iterations with a sketch size $m = 6d$. As shown by the green bars, these sketched solutions show an error $\|\widehat{x} - x^*\|_A \approx 0.11$ independently of dimension, consistent with the predictions of Corollary 2. Finally, the red bars show the error in the classical sketch, based on a sketch size $M = Nm = 24d$, corresponding to the total number of projections used in the iterative algorithm. This error is roughly twice as large.

3.2 Sparse least-squares

We now turn to a study of an ℓ_1-constrained form of least-squares, referred to as the Lasso or relaxed basis pursuit program (Chen et al., 1998; Tibshirani, 1996). In particular, consider the convex program

$$x^{\mathrm{LS}} = \arg \min_{\|x\|_1 \leq R} \{\frac{1}{2}\|y - Ax\|_2^2\}, \tag{31}$$

where $R > 0$ is a user-defined radius. This estimator is well-suited to the problem of sparse linear regression, based on the observation model $y = Ax^* + w$, where x^* has at most s non-zero entries, and $A \in \mathbb{R}^{n \times d}$ has i.i.d. $N(0, 1)$ entries. For the purposes of this illustration, we assume[3] that the radius is chosen such that $R = \|x^*\|_1$.

Under these conditions, the proof of Corollary 3 shows that a sketch size $m \geq \gamma\, s \log\left(\frac{ed}{s}\right)$ suffices to guarantee geometric convergence of the IHS updates. Panel (a) of Figure 4 illustrates the accuracy of this prediction, showing the resulting convergence rate of the the IHS

3. In practice, this unrealistic assumption of exactly knowing $\|x^*\|_1$ is avoided by instead considering the ℓ_1-penalized form of least-squares, but we focus on the constrained case to keep this illustration as simple as possible.

algorithm, measured in terms of the error $\|x^t - x^{\mathrm{LS}}\|_A$, for different values $\gamma \in \{2, 5, 25\}$. As predicted by Theorem 2, the convergence rate is geometric (linear on the log scale shown), with the rate increasing as the parameter γ is increased.

As long as $n \gtrsim s \log\left(\frac{ed}{s}\right)$, it also follows as a corollary of Proposition 2 that

$$\|x^{\mathrm{LS}} - x^*\|_A^2 \precsim \frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{n}. \tag{32}$$

with high probability. This bound suggests an appropriate choice for the tolerance parameter ε in Theorem 2, and leads us to the following guarantee.

Corollary 3 *For the stated random ensemble of sparse linear regression problems, suppose that we run the IHS algorithm for* $N = 1 + \lceil \frac{\log \sqrt{n}}{\log(1/\rho)} \frac{\|x^{LS}\|_A}{\sigma} \rceil$ *iterations using* $m = \frac{c_0}{\rho^2} s \log\left(\frac{ed}{s}\right)$ *projections per round. Then with probability greater than* $1 - c_1 N e^{-c_2 \frac{m\rho^2}{\log^4(d)}}$, *the output* $\widehat{x}$ *satisfies the bounds*

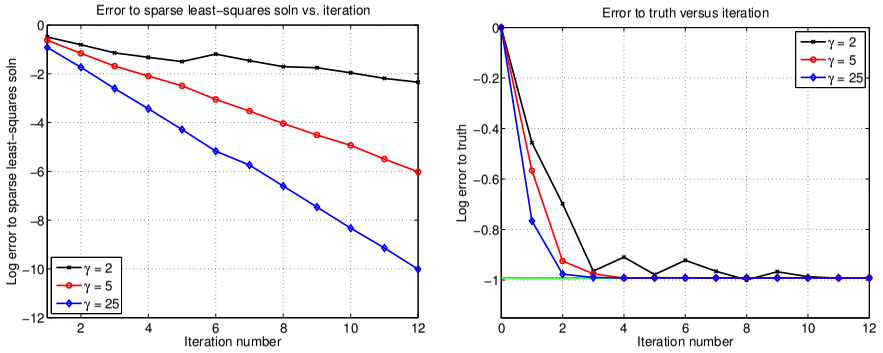
$$\|\widehat{x} - x^{LS}\|_A \leq \sqrt{\frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{n}} \qquad \text{and} \qquad \|x^N - x^*\|_A \leq \sqrt{\frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{n}} + \|x^{LS} - x^*\|_A. \tag{33}$$

In order to verify the predicted bound (33) on the error $\|\widehat{x} - x^{\mathrm{LS}}\|_A$, we performed a second experiment. Fixing $n = 100 s \log\left(\frac{ed}{s}\right)$. we generated $T = 20$ random least squares problems (as described above) with the regression dimension ranging as $d \in \{32, 64, 128, 256\}$, and sparsity $s = \lceil 2\sqrt{d} \rceil$. Based on these choices, the least-squares estimate should have error $\|x^{\mathrm{LS}} - x^*\|_A \approx \sqrt{\frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{n}} = 0.1$ with high probability, independently of the pair (s, d). This predicted behavior is confirmed by the blue bars in Figure 5; the bar height corresponds to the average over $T = 20$ trials, with the standard errors also marked.

On these same problem instances, we also ran the IHS algorithm using $N = 4$ iterations with a sketch size $m = 4 s \log\left(\frac{ed}{s}\right)$. Together with our earlier calculation of $\|x^{\mathrm{LS}} - x^*\|_A$, Corollary 2 implies that with high probability, the sketched solution $\widehat{x} = x^N$ satisfies the error bound

$$\|\widehat{x} - x^*\|_A \leq c_0 \sqrt{\frac{\sigma^2 s \log\left(\frac{ed}{s}\right)}{n}} \tag{34}$$

for some constant $c_0 \in (1, 2]$. This prediction is confirmed by the green bars in Figure 5, showing that $\|\widehat{x} - x^*\|_A \gtrsim 0.11$ across all dimensions. Finally, the green bars in Figure 5 show the error based on using the naive sketch estimate with a total of $M = Nm$ random projections in total; as with the case of ordinary least-squares, the resulting error is roughly twice as large. We also note that a similar bound also applies to problems where a parameter constrained to unit simplex is estimated, such as in portfolio analysis and density estimation (Markowitz, 1959; Pilanci et al., 2012).



(a) (b)

Figure 4. Simulations of the IHS algorithm for a sparse least-squares problem with noise variance $\sigma^2 = 1$, and of dimensions $(d, n, s) = (256, 8872, 32)$. Simulations based on sketch sizes $m = \gamma s \log d$, for a parameter $\gamma > 0$ to be set. (a) Plots of the log error $\|x^t - x^{\mathrm{LS}}\|_2$ versus the iteration number t. Three different curves for $\gamma \in \{2, 5, 25\}$. Consistent with the theory, the convergence is geometric, with the rate increasing as the sampling factor γ is increased. (b) Plots of the log error $\|x^t - x^*\|_2$ versus the iteration number t. Three different curves for $\gamma \in \{2, 5, 25\}$. As expected, all three curves flatten out at the level of the least-squares error $\|x^{\mathrm{LS}} - x^*\|_2 = 0.10 \approx \sqrt{\frac{s \log(ed/s)}{n}}$.

3.3 Some larger-scale experiments

In order to further explore the computational gains guaranteed by IHS, we performed some larger scale experiments on sparse regression problems, with the sample size n ranging over the set $\{2^{12}, 2^{13}, ..., 2^{19}\}$ with a fixed input dimension $d = 500$. As before, we generate observations from the linear model $y = Ax^* + w$, where x^* has at most s non-zero entries, and each row of the data matrix $A \in \mathbb{R}^{n \times d}$ is distributed i.i.d. according to a $N(1_d, \Sigma)$ distribution. Here the d-dimensional covariance matrix Σ has entries $\Sigma_{jk} = 2 \times 0.9^{|j-k|}$, so that the columns of the matrix A will be correlated. Setting a sparsity $s = \lceil 3 \log(d) \rceil$, we chose the unknown regression vector x^* with its support uniformly random with entries $\pm \frac{1}{\sqrt{s}}$ with equal probability.

Baseline: In order to provide a baseline for comparison, we used the homotopy algorithm—that is, the Lasso modification of the LARS updates (Osborne et al., 2000; Efron et al., 2004)—to solve the original ℓ_1 constrained problem with ℓ_1-ball radius $R = \sqrt{s}$. The homotopy algorithm is especially efficient when the Lasso solution x^{LS} is sparse. Since the columns of A are correlated in our ensemble, standard first-order algorithms—among them iterative soft-thresholding, FISTA, spectral projected gradient methods, as well as (block) coordinate descent methods, see, e.g., Beck and Teboulle (2009); Wu and Lange (2008)—performed poorly

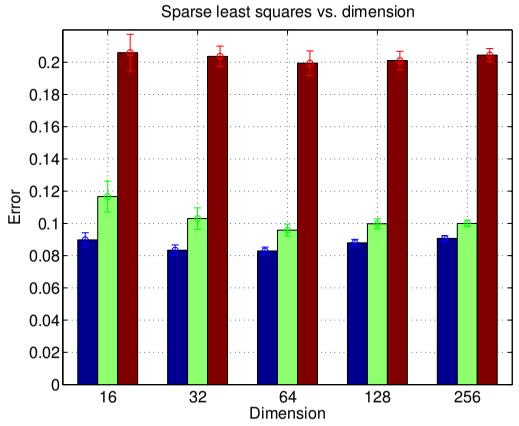


Figure 5. Simulations of the IHS algorithm for ℓ_1-constrained least-squares. In these experiments, we generated random sparse least-squares problem of dimensions $d \in \{16, 32, 64, 128, 256\}$ and sparsity $s = \lceil 2\sqrt{d} \rceil$, on all occasions with a fixed sample size $n = 100 s \log\left(\frac{ed}{s}\right)$. The initial Lasso solution has error $\|x^{\mathrm{LS}} - x^*\|_2 \approx 0.10$, as shown by the blue bars. We then ran the IHS algorithm for $N = 4$ iterations with a sketch size $m = 4s \log\left(\frac{ed}{s}\right)$. These sketched solutions show an error $\|\widehat{x} - x^*\|_A \approx 0.11$ independently of dimension, consistent with the predictions of Corollary 3. Red bars show the error in the naive sketch estimate, using a sketch of size $M = Nm = 16 s \log\left(\frac{ed}{s}\right)$, equal to the total number of random projections used by the IHS algorithm. The resulting error is roughly twice as large.

relative to the homotopy algorithm in terms of computation time; see Bach et al. (2011) for observations of this phenomenon in past work.

IHS implementation: For comparison, we implemented the IHS algorithm with a projection dimension $m = \lfloor 4s \log(d) \rfloor$. After projecting the data, we then used the homotopy method to solve the projected sub-problem at each step. In each trial, we ran the IHS algorithm for $N = \lceil \log n \rceil$ iterations.

Table 1 provides a summary comparison of the running times for the baseline method (homotopy method on the original problem), versus the IHS method (running time for computing the iterates using the homotopy method), and IHS method plus sketching time. Note that with the exception of the smallest problem size ($n = 4096$), the IHS method including sketching time is the fastest, and it is more than two times faster for large problems. The gains are somewhat more significant if we remove the sketching time from the comparison.

One way in which to measure the quality of the least-squares solution x^{LS} as an estimate of x^* is via its mean-squared (in-sample) prediction error $\|x^{\mathrm{LS}} - x^*\|_A^2 = \frac{\|A(x^{\mathrm{LS}} - x^*)\|_2^2}{n}$. For the random ensemble of problems that we have generated, the bound (34) guarantees that the squared error should decay at the rate $1/n$ as the sample size n is increased with the

Samples n	4096	8192	16384	32768	65536	131072	262144	524288
Baseline	0.0840	0.1701	0.3387	0.6779	1.4083	2.9052	6.0163	12.0969
IHS	0.0783	0.0993	0.1468	0.2174	0.3601	0.6846	1.4748	3.1593
IHS+Sketch	0.0877	0.1184	0.1887	0.3222	0.5814	1.1685	2.5967	5.5792

Table 1. Running time comparison in seconds of the Baseline (homotopy method applied to original problem), IHS (homotopy method applied to sketched subproblems), and IHS plus sketching time. Each running time estimate corresponds to an average over 300 independent trials of the random sparse regression model described in the main text.

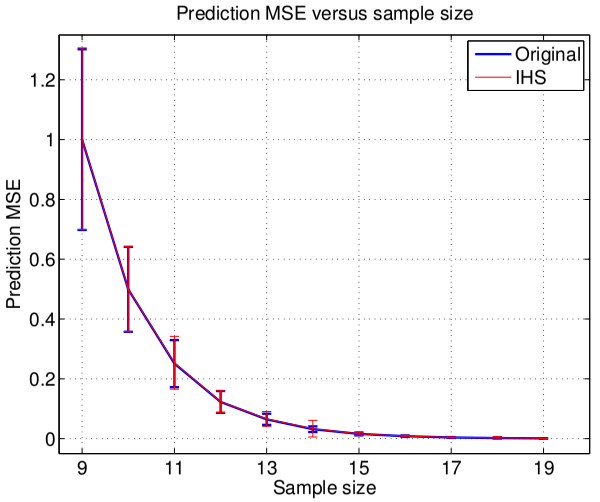


Figure 6. Plots of the mean-squared prediction errors $\frac{\|A(\widetilde{x} - x^*)\|_2^2}{n}$ versus the sample size $n \in 2^{\{9,10,\ldots,19\}}$ for the original least-squares solution ($\widetilde{x} = x^{\mathrm{LS}}$ in blue) versus the sketched solution ($\widehat{x} = x^{\mathrm{LS}}$ in red). Each point on each curve corresponds to the average over 300 independent trials of the same type used to generate the data in Table 1; the error bars correspond to one standard errors. In generating the plots, all errors have been renormalized so that the error for sample size $n = 2^9$ is equal to one. As can be seen, the sketched method generates solutions with prediction MSE that are essentially indistinguishable from the original solution.

dimension d and sparsity s fixed. Figure 6 compares the prediction MSE of x^{LS} versus the analogous quantity $\|\widehat{x} - x^*\|_A^2$ for the sketched solution. Note that the two curves are essentially indistinguishable, showing that the sketched solution provides an estimate of x^* that is as good as the original least-squares estimate.

3.4 Matrix estimation with nuclear norm constraints

We now turn to the study of nuclear-norm constrained form of least-squares matrix regression. This class of problems has proven useful in many different application areas, among them matrix completion, collaborative filtering, multi-task learning and control theory (e.g., (Fazel, 2002; Yuan et al., 2007; Bach, 2008; Recht et al., 2010; Negahban and Wainwright, 2012)). In particular, let us consider the convex program

$$X^{\mathrm{LS}} = \arg \min_{X \in \mathbb{R}^{d_1 \times d_2}} \left\{ \frac{1}{2} \|Y - AX\|_{\mathrm{fro}}^2 \right\} \qquad \text{such that } \|X\|_{\mathrm{nuc}} \leq R, \tag{35}$$

where $R > 0$ is a user-defined radius as a regularization parameter.

3.4.1 Simulated data

Recall the linear observation model previously introduced in Example 3: we observe the pair (Y, A) linked according to the linear $Y = AX^* + W$, where the unknown matrix $X^* \in \mathbb{R}^{d_1 \times d_2}$ is an unknown matrix of rank r. The matrix W is observation noise, formed with i.i.d. $N(0, \sigma^2)$ entries. This model is a special case of the more general class of matrix regression problems (Negahban and Wainwright, 2012). As shown in Appendix D.2, if we solve the nuclear-norm constrained problem with $R = \|X^*\|_{\mathrm{nuc}}$, then it produces a solution such that $\mathbb{E}\big[\|X^{\mathrm{LS}} - X^*\|_{\mathrm{fro}}^2\big] \precsim \sigma^2 \frac{r\,(d_1+d_2)}{n}$. The following corollary characterizes the sketch dimension and iteration number required for the IHS algorithm to match this scaling up to a constant factor.

Corollary 4 (IHS for nuclear-norm constrained least squares) *Suppose that we run the IHS algorithm for* $N = 1 + \lceil \frac{\log \sqrt{n}}{\log(1/\rho)} \frac{\|X^{LS}\|_A}{\sigma} \rceil$ *iterations using* $m = c_0 \rho^2 r\big(d_1 + d_2\big)$ *projections per round. Then with probability greater than* $1 - c_1\, N\, e^{-c_2 \frac{m\rho^2}{\log^4(d_1 d_2)}}$, *the output* X^N *satisfies the bound*

$$\|X^N - X^*\|_A \leq \sqrt{\frac{\sigma^2 r\big(d_1 + d_2\big)}{n}} + \|X^{LS} - X^*\|_A. \tag{36}$$

We have also performed simulations for low-rank matrix estimation, and observed that the IHS algorithm exhibits convergence behavior qualitatively similar to that shown in Figures 3 and 5. Similarly, panel (a) of Figure 8 compares the performance of the IHS and classical methods for sketching the optimal solution over a range of row sizes n. As with the unconstrained least-squares results from Figure 1, the classical sketch is very poor compared to the original solution whereas the IHS algorithm exhibits near optimal performance.

3.4.2 Application to multi-task learning

To conclude, let us illustrate the use of the IHS algorithm in speeding up the training of a classifier for facial expressions. In particular, suppose that our goal is to separate a collection of facial images into different groups, corresponding either to distinct individuals or to different facial expressions. One approach would be to learn a different linear classifier $(a \mapsto \langle a, x \rangle)$

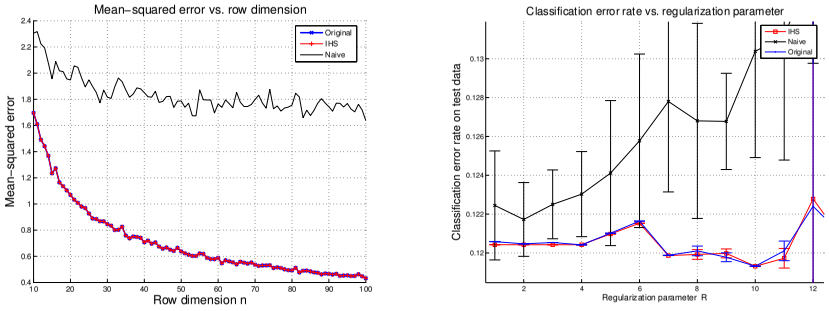
for each separate task, but since the classification problems are so closely related, the optimal classifiers are likely to share structure. One way of capturing this shared structure is by concatenating all the different linear classifiers into a matrix, and then estimating this matrix in conjunction with a nuclear norm penalty (Amit et al., 2007; Argyriou et al., 2008).

Figure 7. Japanese Female Facial Expression (JAFFE) Database: The JAFFE database consists of 213 images of 7 different emotional facial expressions (6 basic facial expressions + 1 neutral) posed by 10 Japanese female models.

In more detail, we performed a simulation study using the The Japanese Female Facial Expression (JAFFE) database (Lyons et al., 1998). It consists of $N = 213$ images of 7 facial expressions (6 basic facial expressions + 1 neutral) posed by 10 different Japanese female models; see Figure 7 for a few example images. We performed an approximately $80:20$ split of the data set into $n_{\mathrm{train}} = 170$ training and $n_{\mathrm{test}} = 43$ test images respectively. Then we consider classifying each facial expression and each female model as a separate task which gives a total of $d_{\mathrm{task}} = 17$ tasks. For each task $j = 1, \ldots, d_{\mathrm{task}}$, we construct a linear classifier of the form $a \mapsto \mathrm{sign}(\langle a, x_j \rangle)$, where $a \in \mathbb{R}^d$ denotes the vectorized image features given by Local Phase Quantization (Ojansivu and Heikkil, 2008). In our implementation, we fixed the number of features $d = 32$. Given this set-up, we train the classifiers in a joint manner, by optimizing simultaneously over the matrix $X \in \mathbb{R}^{d \times d_{\mathrm{task}}}$ with the classifier vector $x_j \in \mathbb{R}^d$ as its j^{th} column. The image data is loaded into the matrix $A \in \mathbb{R}^{n_{\mathrm{train}} \times d}$, with image feature vector $a_i \in \mathbb{R}^d$ in column i for $i = 1, \ldots, n_{\mathrm{train}}$. Finally, the matrix $Y \in \{-1, +1\}^{n_{\mathrm{train}} \times d_{\mathrm{task}}}$ encodes class labels for the different classification problems. These instantiations of the pair (Y, X) give us an optimization problem of the form (35), and we solve it over a range of regularization radii R.

More specifically, in order to verify the classification accuracy of the classifier obtained by IHT algorithm, we solved the original convex program, the classical sketch based on ROS sketches of dimension $m = 100$, and also the corresponding IHS algorithm using ROS sketches of size 20 in each of 5 iterations. In this way, both the classical and IHS procedures use the same total number of sketches, making for a fair comparison. We repeated each of these three procedures for all choices of the radius $R \in \{1, 2, 3, \ldots, 12\}$, and then applied the resulting classifiers to classify images in the test dataset. For each of the three procedures, we calculated the classification error rate, defined as the total number of mis-classified images divided by $n_{\mathrm{test}} \times d_{\mathrm{task}}$. Panel (b) of Figure 8 plots the resulting classification errors versus the regularization parameter. The error bars correspond to one standard deviation calculated

over the randomness in generating sketching matrices. The plots show that the IHS algorithm yields classifiers with performance close to that given by the original solution over a range of regularizer parameters, and is superior to the classification sketch. The error bars also show that the IHS algorithm has less variability in its outputs than the classical sketch.



(a) (b)

Figure 8. Simulations of the IHS algorithm for nuclear-norm constrained problems. The blue curves correspond to the solution of the original (unsketched problem), whereas red curves correspond to the IHS method applied for $N = 1 + \lceil \log(n) \rceil$ rounds using a sketch size of m. Black curves correspond to the naive sketch applied using $M = Nm$ projections in total, corresponding to the same number used in all iterations of the IHS algorithm. (a) Mean-squared error versus the row dimension $n \in [10, 100]$ for recovering a 20×20 matrix of rank $r2$, using a sketch dimension $m = 60$. Note how the accuracy of the IHS algorithm tracks the error of the unsketched solution over a wide range of n, whereas the classical sketch has essentially constant error. (b) Classification error rate versus regularization parameter $R \in \{1, \ldots, 12\}$, with error bars corresponding to one standard deviation over the test set. Sketching algorithms were applied to the JAFFE face expression using a sketch dimension of $M = 100$ for the classical sketch, and $N = 5$ iterations with $m = 20$ sketches per iteration for the IHS algorithm.

4. Discussion

In this paper, we focused on the problem of solution approximation (as opposed to cost approximation) for a broad class of constrained least-squares problem. We began by showing that the classical sketching methods are sub-optimal, from an information-theoretic point of view, for the purposes of solution approximation. We then proposed a novel iterative scheme, known as the iterative Hessian sketch, for deriving ε-accurate solution approximations. We proved a general theorem on the properties of this algorithm, showing that the sketch dimension per iteration need grow only proportionally to the statistical dimension of the optimal solution, as measured by the Gaussian width of the tangent cone at the optimum. By taking $\log(1/\varepsilon)$ iterations, the IHS algorithm is guaranteed to return an ε-accurate solution approximation with exponentially high probability.

In addition to these theoretical results, we also provided empirical evaluations that reveal the sub-optimality of the classical sketch, and show that the IHS algorithm produces near-optimal estimators. Finally, we applied our methods to a problem of facial expression using a multi-task learning model applied to the JAFFE face database. We showed that IHS algorithm applied to a nuclear-norm constrained program produces classifiers with considerably better classification accuracy compared to the naive sketch.

There are many directions for further research, but we only list here some of them. The idea behind iterative sketching can also be applied to problems beyond minimizing a least-squares objective function subject to convex constraints. Examples include penalized forms of regression, e.g., see the recent work (Yang et al., 2015), and various other cost functions. An important class of such problems are ℓ_p-norm forms of regression, based on the convex program

$$\min_{x \in \mathbb{R}^d} \|Ax - y\|_p^p \qquad \text{for some } p \in [1, \infty].$$

The case of ℓ_1-regression ($p = 1$) is an important special case, known as robust regression; it is especially effective for data sets containing outliers (Huber, 2001). Recent work (Clarkson et al., 2013) has proposed to find faster solutions of the ℓ_1-regression problem using the classical sketch (i.e., based on (SA, Sy)) but with sketching matrices based on Cauchy random vectors. Based on the results of the current paper, our iterative technique might be useful in obtaining sharper bounds for solution approximation in this setting as well. Finally, we refer the reader to the more recent work (Pilanci and Wainwright, 2015b) on sketching for general convex objective functions.

Acknowledgments

Both authors were partially supported by Office of Naval Research MURI grant N00014-11-1-0688, Office of Naval Research MURI grant ONR-MURI-DOD-002888, and National Science Foundation Grants CIF-31712-23800 and DMS-1107000. In addition, MP was supported by a Microsoft Research Fellowship.

Appendix A. Proof of lower bounds

This appendix is devoted to the verification of condition (9) for different model classes, followed by the proof of Theorem 1.

A.1 Verification of condition (9)

We verify the condition for three different types of sketches.

A.1.1 Gaussian sketches:

First, let $S \in \mathbb{R}^{m \times n}$ be a random matrix with i.i.d. Gaussian entries. We use the singular value decomposition to write $S = U \Lambda V^T$ where both U and V are orthonormal matrices of left and right singular vectors. By rotation invariance, the columns $\{v_i\}_{i=1}^m$ are uniformly

distributed over the sphere $\mathcal{S}^{n-1}$. Consequently, we have

$$\mathbb{E}_S\big[S^T\big(SS^T\big)^{-1}S\big] = \mathbb{E}\sum_{i=1}^m v_i v_i^T = \frac{m}{n} I_n, \tag{37}$$

showing that condition (9) holds with $\eta = 1$.

A.1.2 ROS sketches (sampled without replacement):

In this case, we have $S = \sqrt{n}PHD$, where $P \in \mathbb{R}^{m\times n}$ is a random picking matrix with each row being a standard basis vector sampled without replacement. We then have $SS^T = nI_m$ and also $\mathbb{E}_P[P^TP] = \frac{m}{n}I_n$, so that

$$\mathbb{E}_S[S^T(SS^T)^{-1}S] = \mathbb{E}_{D,P}[DH^TP^TPHD] = \mathbb{E}_D[DH^T(\frac{m}{n}I_n)HD] = \frac{m}{n}I_n,$$

showing that the condition holds with $\eta = 1$.

A.1.3 Weighted row sampling:

Finally, suppose that we sample m rows independently using a distribution $\{p_j\}_{j=1}^n$ on the rows of the data matrix that is α-balanced (7). Letting $\mathcal{R} \subseteq \{1, 2, \ldots, n\}$ be the subset of rows that are sampled, and let N_j be the number of times each row is sampled. We then have

$$\mathbb{E}\Big[S^T\big(SS^T\big)^{-1}S\Big] = \sum_{j\in\mathcal{R}} \mathbb{E}[e_j e_j^T] \;=\; D,$$

where $D \in \mathbb{R}^{n\times n}$ is a diagonal matrix with entries $D_{jj} = \mathbb{P}[j \in \mathcal{R}]$. Since the trials are independent, the j^{th} row is sampled at least once in m trials with probability $q_j = 1-(1-p_j)^m$, and hence

$$\mathbb{E}_S\big[S^T\big(SS^T\big)^{-1}S\big] = \operatorname{diag}\big(\{1-(1-p_i)^m\}_{i=1}^m\big) \;\preceq\; \big(1-(1-p_\infty)^m\big)I_n \;\preceq mp_\infty,$$

where $p_\infty = \max_{j\in[n]} p_j$. Consequently, as long as the row weights are α-balanced (7) so that $p_\infty \le \frac{\alpha}{n}$, we have

$$\big\|\mathbb{E}_S\big[S^T\big(SS^T\big)^{-1}S\big]\big\|_{\text{op}} \le \alpha\frac{m}{n}$$

showing that condition (9) holds with $\eta = \alpha$, as claimed.

A.2 Proof of Theorem 1

Let $\{z^j\}_{j=1}^M$ be a 1/2-packing of $\mathcal{C}_0 \cap \mathbb{B}_A(1)$ in the semi-norm $\|\cdot\|_A$, and for a fixed $\delta \in (0, 1/4)$, define $x^j = 4\delta z^j$. Sine $4\delta \in (0,1)$, the star-shaped assumption guarantees that each x^j belongs to $\mathcal{C}_0$. We thus obtain a collection of M vectors in $\mathcal{C}_0$ such that

$$2\delta \le \underbrace{\frac{1}{\sqrt{n}}\|A(x^j - x^k)\|_2}_{\|x^j-x^k\|_A} \le 8\delta \qquad \text{for all } j \neq k.$$

Letting J be a random index uniformly distributed over $\{1, \ldots, M\}$, suppose that conditionally on $J = j$, we observe the sketched observation vector $Sy = SAx^j + Sw$, as well as the sketched matrix SA. Conditioned on $J = j$, the random vector Sy follows a $\mathcal{N}(SAx^j, \sigma^2 SS^T)$ distribution, denoted by $\mathbb{P}_{x^j}$. We let $\overline{Y}$ denote the resulting mixture variable, with distribution $\frac{1}{M}\sum_{j=1}^M \mathbb{P}_{x^j}$.

Consider the multiway testing problem of determining the index J based on observing $\overline{Y}$. With this set-up, a standard reduction in statistical minimax (e.g., (Birgé, 1987; Yu, 1997)) implies that, for any estimator $x^\dagger$, the worst-case mean-squared error is lower bounded as

$$\sup_{x^*\in\mathcal{C}} \mathbb{E}_{S,w}\|x^\dagger - x^*\|_A^2 \ge \delta^2 \inf_\psi \mathbb{P}[\psi(\overline{Y}) \neq J], \tag{38}$$

where the infimum ranges over all testing functions ψ. Consequently, it suffices to show that the testing error is lower bounded by 1/2.

In order to do so, we first apply Fano's inequality (Cover and Thomas, 1991) conditionally on the sketching matrix S to see that

$$\mathbb{P}[\psi(\overline{Y}) \neq J] \;=\; \mathbb{E}_S\Big\{\mathbb{P}[\psi(\overline{Y}) \neq J \mid S]\Big\} \ge 1 - \frac{\mathbb{E}_S\big[I_S(\overline{Y}; J)\big] + \log 2}{\log M}, \tag{39}$$

where $I_S(\overline{Y}; J)$ denotes the mutual information between $\overline{Y}$ and J with S fixed. Our next step is to upper bound the expectation $\mathbb{E}_S[I(\overline{Y}; J)]$.

Letting $D(\mathbb{P}_{x^j} \,\|\, \mathbb{P}_{x^k})$ denote the Kullback-Leibler divergence between the distributions $\mathbb{P}_{x^j}$ and $\mathbb{P}_{x^k}$, the convexity of Kullback-Leibler divergence implies that

$$I_S(\overline{Y}; J) = \frac{1}{M}\sum_{j=1}^M D(\mathbb{P}_{x^j} \,\|\, \frac{1}{M}\sum_{k=1}^M \mathbb{P}_{x^k}) \;\le\; \frac{1}{M^2}\sum_{j,k=1}^M D(\mathbb{P}_{x^j} \,\|\, \mathbb{P}_{x^k}).$$

Computing the KL divergence for Gaussian vectors yields

$$I_S(\overline{Y}; J) \le \frac{1}{M^2}\sum_{j,k=1}^M \frac{1}{2\sigma^2}(x^j - x^k)^T A^T\Big[S^T(SS^T)^{-1}S\Big]A(x^j - x^k).$$

Thus, using condition (9), we have

$$\mathbb{E}_S[I(\overline{Y}; J)] \le \frac{1}{M^2}\sum_{j,k=1}^M \frac{m\,\eta}{2\,n\sigma^2}\|A(x^j - x^k)\|_2^2 \;\le \frac{32\,m\,\eta}{\sigma^2}\,\delta^2,$$

where the final inequality uses the fact that $\|x^j - x^k\|_A \le 8\delta$ for all pairs.

Combined with our previous bounds (38) and (39), we find that

$$\sup_{x^*\in\mathcal{C}} \mathbb{E}\|\widehat{x} - x^*\|_2^2 \ge \delta^2\Big\{1 - \frac{32\frac{m\,\eta\,\delta^2}{\sigma^2} + \log 2}{\log M}\Big\}.$$

Setting $\delta = \frac{\sigma^2\log(M/2)}{64\,\eta\,m}$ yields the lower bound (10).

Appendix B. Proof of Proposition 1

Since $\widehat{x}$ and x^{LS} are optimal and feasible, respectively, for the Hessian sketch program (16), we have

$$\langle A^T S^T \big(SA\widehat{x} - y\big),\, x^{\mathrm{LS}} - \widehat{x}\rangle \geq 0 \tag{40a}$$

Similarly, since x^{LS} and $\widehat{x}$ are optimal and feasible, respectively, for the original least squares program

$$\langle A^T(Ax^{\mathrm{LS}} - y),\, \widehat{x} - x^{\mathrm{LS}}\rangle \geq 0. \tag{40b}$$

Adding these two inequalities and performing some algebra yields the basic inequality

$$\frac{1}{m}\|SA\Delta\|_2^2 \leq \Big|(Ax^{\mathrm{LS}})^T\big(I_n - \frac{S^TS}{m}\big)A\Delta\Big|. \tag{41}$$

Since Ax^{LS} is independent of the sketching matrix and $A\Delta \in \mathcal{K}^{\mathrm{LS}}$, we have

$$\frac{1}{m}\|SA\Delta\|_2^2 \geq Z_1\,\|A\Delta\|_2^2, \qquad \text{and} \quad \Big|(Ax^{\mathrm{LS}})^T\big(I_n - S^TS\big)A\Delta\Big| \leq Z_2\|Ax^{\mathrm{LS}}\|_2\,\|A\Delta\|_2,$$

using the definitions (18a) and (18b) of the random variables Z_1 and Z_2 respectively. Combining the pieces yields the claim.

Appendix C. Proof of Theorem 2

It suffices to show that, for each iteration $t = 0, 1, 2, \ldots$, we have

$$\|x^{t+1} - x^{\mathrm{LS}}\|_A \leq \frac{Z_2(S^{t+1})}{Z_1(S^{t+1})}\|x^t - x^{\mathrm{LS}}\|_A. \tag{42}$$

The claimed bounds (27a) and (27b) then follow by applying the bound (42) successively to iterates 1 through N.

For simplicity in notation, we abbreviate S^{t+1} to S and x^{t+1} to $\widehat{x}$. Define the error vector $\Delta = \widehat{x} - x^{\mathrm{LS}}$. With some simple algebra, the optimization problem (25) that underlies the update $t+1$ can be re-written as

$$\widehat{x} = \arg\min_{x\in\mathcal{C}}\Big\{\frac{1}{2m}\|SAx\|_2^2 - \langle A^T\widetilde{y},\, x\rangle\Big\},$$

where $\widetilde{y} := y - \Big[I - \frac{S^TS}{m}\Big]Ax^t$. Since $\widehat{x}$ and x^{LS} are optimal and feasible respectively, the usual first-order optimality conditions imply that

$$\langle A^T\frac{S^TS}{m}Ax - A^T\widetilde{y},\, x^{\mathrm{LS}} - \widehat{x}\rangle \geq 0.$$

As before, since x^{LS} is optimal for the original program, we have

$$\langle A^T(Ax^{\mathrm{LS}} - \widetilde{y} + \Big[I - \frac{S^TS}{m}\Big]Ax^t),\, \widehat{x} - x^{\mathrm{LS}}\rangle \geq 0.$$

Adding together these two inequalities and introducing the shorthand $\Delta = \widehat{x} - x^{\mathrm{LS}}$ yields

$$\frac{1}{m}\|SA\Delta\|_2^2 \leq \Big|(A(x^{\mathrm{LS}} - x^t)^T\big[I - \frac{S^TS}{m}\big]A\Delta\Big| \tag{43}$$

Note that the vector $A(x^{\mathrm{LS}} - x^t)$ is independent of the randomness in the sketch matrix S^{t+1}. Moreover, the vector $A\Delta$ belongs to the cone $\mathcal{K}$, so that by the definition of $Z_2(S^{t+1})$, we have

$$\Big|(A(x^{\mathrm{LS}} - x^t)^T\Big[I - \frac{S^TS}{m}\Big]A\Delta\Big| \leq \|A(x^{\mathrm{LS}} - x^t)\|_2\,\|A\Delta\|_2\,Z_2(S^{t+1}). \tag{44a}$$

Similarly, note the lower bound

$$\frac{1}{m}\|SA\Delta\|_2^2 \geq \|A\Delta\|_2^2\,Z_1(S^{t+1}). \tag{44b}$$

Combining the two bounds (44a) and (44b) with the earlier bound (43) yields the claim (42).

Appendix D. Maximum likelihood estimator and examples

In this section, we a general upper bound on the error of the constrained least-squares estimate. We then use it (and other results) to work through the calculations underlying Examples 1 through 3 from Section 2.2.

D.1 Upper bound on MLE

The accuracy of x^{LS} as an estimate of x^* depends on the "size" of the star-shaped set

$$\mathcal{K}(x^*) = \big\{v \in \mathbb{R}^d \mid v = \frac{t}{\sqrt{n}}A(x - x^*) \quad \text{for some } t \in [0,1] \text{ and } x \in \mathcal{C}\big\}. \tag{45}$$

When the vector x^* is clear from context, we use the shorthand notation $\mathcal{K}^*$ for this set. By taking a union over all possible $x^* \in \mathcal{C}_0$, we obtain the set $\overline{\mathcal{K}} := \bigcup_{x^*\in\mathcal{C}_0}\mathcal{K}(x^*)$, which plays an important role in our bounds. The complexity of these sets can be measured of their *localized Gaussian widths*. For any radius $\varepsilon > 0$ and set $\Theta \subseteq \mathbb{R}^n$, the Gaussian width of the set $\Theta \cap \mathbb{B}_2(\varepsilon)$ is given by

$$\mathcal{W}_\varepsilon(\Theta) := \mathbb{E}_g\Big[\sup_{\substack{\theta\in\Theta\\ \|\theta\|_2\leq\varepsilon}}|\langle w,\, \theta\rangle|\Big], \tag{46a}$$

where $g \sim N(0, I_{n\times n})$ is a standard Gaussian vector. Whenever the set Θ is star-shaped, then it can be shown that, for any $\sigma > 0$ and positive integer ℓ, the inequality

$$\frac{\mathcal{W}_\varepsilon(\Theta)}{\varepsilon\sqrt{\ell}} \leq \frac{\varepsilon}{\sigma} \tag{46b}$$

has a smallest positive solution, which we denote by $\varepsilon_\ell(\Theta;\sigma)$. We refer the reader to Bartlett et al. (2005) for further discussion of such localized complexity measures and their properties. The following result bounds the mean-squared error associated with the constrained least-squares estimate:

Proposition 2 *For any set $\mathcal{C}$ containing x^*, the constrained least-squares estimate* (1) *has mean-squared error upper bounded as*

$$\mathbb{E}_w\big[\|x^{LS} - x^*\|_A^2\big] \leq c_1\big\{\varepsilon_n^2(\mathcal{K}^*) + \frac{\sigma^2}{n}\big\} \;\leq\; c_1\big\{\varepsilon_n^2(\overline{\mathcal{K}}) + \frac{\sigma^2}{n}\big\}. \tag{47}$$

We provide the proof of this claim in Section D.3.

D.2 Detailed calculations for illustrative examples

In this appendix, we collect together the details of calculations used in our illustrative examples from Section 2.2. In all cases, we make use tof the convenient shorthand $\widetilde{A} = A/\sqrt{n}$.

D.2.1 Unconstrained least squares: Example 1

By definition of the Gaussian width, we have

$$\mathcal{W}_\delta(\mathcal{K}^*) = \mathbb{E}_g\big[\sup_{\|\widetilde{A}(x-x^*)\|_2 \leq \delta} |\langle g,\, \widetilde{A}(x - x^*)\rangle|\big] \;\leq\; \delta\sqrt{d}$$

since the vector $\widetilde{A}(x - x^*)$ belongs to a subspace of dimension $\operatorname{rank}(A) = d$. The claimed upper bound (11a) thus follows as a consequence of Proposition 2.

D.2.2 Sparse vectors: Example 2

The RIP property of order $8s$ implies that

$$\frac{\|\Delta\|_2^2}{2} \overset{(i)}{\leq} \|\widetilde{A}\Delta\|_2^2 \overset{(ii)}{\leq} 2\|\Delta\|_2^2 \qquad \text{for all vectors with } \|\Delta\|_0 \leq 8s,$$

a fact which we use throughout the proof. By definition of the Gaussian width, we have

$$\mathcal{W}_\delta(\mathcal{K}^*) = \mathbb{E}_g\big[\sup_{\substack{\|x\|_1 \leq \|x^*\|_1 \\ \|\widetilde{A}(x-x^*)\|_2 \leq \delta}} |\langle g,\, \widetilde{A}(x - x^*)\rangle|\big].$$

Since $x^* \in \mathbb{B}_0(s)$, it can be shown (e.g., see the proof of Corollary 3 in Pilanci and Wainwright (2015a)) that for any vector $\|x\|_1 \leq \|x^*\|_1$, we have $\|x - x^*\|_1 \leq 2\sqrt{s}\|x - x^*\|_2$. Thus, it suffices to bound the quantity

$$F(\delta; s) := \mathbb{E}_g\big[\sup_{\substack{\|\Delta\|_1 \leq 2\sqrt{s}\|\Delta\|_2 \\ \|\widetilde{A}\Delta\|_2 \leq \delta}} |\langle g,\, \widetilde{A}\Delta\rangle|\big].$$

By Lemma 11 in Loh and Wainwright (2012), we have

$$\mathbb{B}_1(\sqrt{s}) \cap \mathbb{B}_2(1) \subseteq 3\,\mathrm{clconv}\Big\{\mathbb{B}_0(s) \cap \mathbb{B}_2(1)\Big\},$$

where clconv denotes the closed convex hull. Applying this lemma with $s = 4s$, we have

$$F(\delta; s) \leq 3\big[\sup_{\substack{\|\Delta\|_0 \leq 4s \\ \|\widetilde{A}\Delta\|_2 \leq \delta}} |\langle g,\, \widetilde{A}\Delta\rangle|\big] \;\leq\; 3\mathbb{E}\big[\sup_{\substack{\|\Delta\|_0 \leq 4s \\ \|\Delta\|_2 \leq 2\delta}} |\langle g,\, \widetilde{A}\Delta\rangle|\big],$$

using the lower RIP property (i). By the upper RIP property, for any pair of vectors Δ, Δ' with ℓ_0-norms at most $4s$, we have

$$\operatorname{var}\big(\langle g,\, \widetilde{A}\Delta\rangle - \langle g,\, \widetilde{A}\Delta'\rangle\big) \leq 2\|\Delta - \Delta'\|_2^2 \;=\; 2\operatorname{var}\big(\langle g,\, \Delta - \Delta'\rangle\big)$$

Consequently, by the Sudakov-Fernique comparison (Ledoux and Talagrand, 1991), we have

$$\mathbb{E}\big[\sup_{\substack{\|\Delta\|_0 \leq 4s \\ \|\Delta\|_2 \leq 2\delta}} |\langle g,\, \widetilde{A}\Delta\rangle|\big] \leq 2\mathbb{E}\big[\sup_{\substack{\|\Delta\|_0 \leq 4s \\ \|\Delta\|_2 \leq 2\delta}} |\langle g,\, \Delta\rangle|\big] \;\leq c\,\delta\sqrt{s\log\big(\frac{ed}{s}\big)},$$

where the final inequality standard results on Gaussian widths (Gordon et al., 2007). All together, we conclude that

$$\varepsilon_n^2(\mathcal{K}^*; \sigma) \leq c_1\sigma^2\frac{s\log\big(\frac{ed}{s}\big)}{n}.$$

Combined with Proposition 2, the claimed upper bound (12a) follows.

In the other direction, a straightforward argument (e.g., Raskutti et al. (2011)) shows that there is a universal constant $c > 0$ such that $\log M_{1/2} \geq c\,s\log\big(\frac{ed}{s}\big)$, so that the stated lower bound follows from Theorem 1.

D.2.3 Low rank matrices: Example 3:

By definition of the Gaussian width, we have width, we have

$$\mathcal{W}_\delta(\mathcal{K}^*) = \mathbb{E}_g\left[\sup_{\substack{\|\widetilde{A}(X-X^*)\|_{\mathrm{fro}} \leq \delta \\ \|X\|_{\mathrm{nuc}} \leq \|X^*\|_{\mathrm{nuc}}}} |\langle\!\langle \widetilde{A}^T G,\; (X - X^*)\rangle\!\rangle|\right],$$

where $G \in \mathbb{R}^{n\times d_2}$ is a Gaussian random matrix, and $\langle\!\langle C,\; D\rangle\!\rangle$ denotes the trace inner product between matrices C and D. Since X^* has rank at most r, it can be shown that $\|X - X^*\|_{\mathrm{nuc}} \leq 2\sqrt{r}\|X - X^*\|_{\mathrm{fro}}$; for instance, see Lemma 1 in Negahban and Wainwright (2011). Recalling that $\gamma_{\min}(\widetilde{A})$ denotes the minimum singular value, we have

$$\|X - X^*\|_{\mathrm{fro}} \leq \frac{1}{\gamma_{\min}(\widetilde{A})}\|\widetilde{A}(X - X^*)\|_{\mathrm{fro}} \leq \frac{\delta}{\gamma_{\min}(\widetilde{A})}.$$

Thus, by duality between the nuclear and operator norms, we have

$$\mathbb{E}_g\left[\sup_{\substack{\|\widetilde{A}(X-X^*)\|_{\mathrm{fro}} \leq \delta \\ \|X\|_{\mathrm{nuc}} \leq \|X^*\|_{\mathrm{nuc}}}} |\langle\!\langle G,\; \widetilde{A}(X - X^*)\rangle\!\rangle|\right] \leq \frac{2\sqrt{r}\,\delta}{\gamma_{\min}(A)}\,\mathbb{E}\big[\|\widetilde{A}^T G\|_{\mathrm{op}}\big].$$

Now consider the matrix $A^T G \in \mathbb{R}^{d_1\times d_2}$. For any fixed pair of vectors $(u, v) \in \mathcal{S}^{d_1-1} \times \mathcal{S}^{d_2-1}$, the random variable $Z = u^T\widetilde{A}^T G v$ is zero-mean Gaussian with variance at most $\gamma_{\max}^2(\widetilde{A})$. Consequently, by a standard covering argument in random matrix theory Vershynin (2012), we have $\mathbb{E}\big[\|\widetilde{A}^T G\|_{\mathrm{op}}\big] \precsim \gamma_{\max}(\widetilde{A})\big(\sqrt{d_1 + d_2}\big)$. Putting together the pieces, we conclude that

$$\varepsilon_n^2 \preceq \sigma^2\,\frac{\gamma_{\max}^2(A)}{\gamma_{\min}^2(A)}\,r\,(d_1 + d_2),$$

so that the upper bound (15a) follows from Proposition 2.

D.3 Proof of Proposition 2

Throughout this proof, we adopt the shorthand $\varepsilon_n = \varepsilon_n(\mathcal{K}^*)$. Our strategy is to prove the following more general claim: for any $t \geq \varepsilon_n$, we have

$$\mathbb{P}_{S,w}\big[\|x^{\mathrm{LS}} - x^*\|_A^2 \geq 16 t\varepsilon_n\big] \leq c_1 e^{-c_2 \frac{nt\varepsilon_n}{\sigma^2}}. \tag{48}$$

A simple integration argument applied to this tail bound implies the claimed bound (47) on the expected mean-squared error.

Since x^* and x^{LS} are feasible and optimal, respectively, for the optimization problem (1), we have the basic inequality

$$\frac{1}{2n}\|y - Ax^{\mathrm{LS}}\|_2^2 \leq \frac{1}{2n}\|y - Ax^*\|_2 \;=\; \frac{1}{2n}\|w\|_2^2.$$

Introducing the shorthand $\Delta = x^{\mathrm{LS}} - x^*$ and re-arranging terms yields

$$\frac{1}{2}\|\Delta\|_A^2 \;=\; \frac{1}{2n}\|A\Delta\|_2^2 \leq \frac{\sigma}{n}\big|\sum_{i=1}^n \langle g,\, A\Delta\rangle\big|, \tag{49}$$

where $g \sim N(0, I_n)$ is a standard normal vector.

For a given $u \geq \varepsilon_n$, define the "bad" event

$$\mathcal{B}(u) := \big\{\exists \quad z \in \mathcal{C} - x^* \quad \text{with } \|z\|_A \geq u, \text{ and } |\tfrac{\sigma}{n}\textstyle\sum_{i=1}^n g_i (Az)_i| \geq 2u\,\|z\|_A\big\}$$

The following lemma controls the probability of this event:

Lemma 2 *For all $u \geq \varepsilon_n$, we have $\mathbb{P}[\mathcal{B}(u)] \leq e^{-\frac{nu^2}{2\sigma^2}}$.*

Returning to prove this lemma momentarily, let us prove the bound (48). For any $t \geq \varepsilon_n$, we can apply Lemma 2 with $u = \sqrt{t\varepsilon_n}$ to find that

$$\mathbb{P}[\mathcal{B}^c(\sqrt{t\varepsilon_n})] \geq 1 - e^{-\frac{nt\varepsilon_n}{2\sigma^2}}.$$

If $\|\Delta\|_A < \sqrt{t\,\varepsilon_n}$, then the claim is immediate. Otherwise, we have $\|\Delta\|_A \geq \sqrt{t\,\varepsilon_n}$. Since $\Delta \in \mathcal{C} - x^*$, we may condition on $\mathcal{B}^c(\sqrt{t\varepsilon_n})$ so as to obtain the bound

$$\big|\frac{\sigma}{n}\sum_{i=1}^n g_i(A\Delta)_i\big| \leq 2\,\|\Delta\|_A\,\sqrt{t\varepsilon_n}.$$

Combined with the basic inequality (49), we see that

$$\frac{1}{2}\|\Delta\|_A^2 \leq 2\,\|\Delta\|_A\,\sqrt{t\varepsilon_n}, \qquad \text{or equivalently } \|\Delta\|_A^2 \leq 16 t\varepsilon_n,$$

a bound that holds with probability greater than $1 - e^{-\frac{nt\varepsilon_n}{2\sigma^2}}$ as claimed.

It remains to prove Lemma 2. Our proof involves the auxiliary random variable

$$V_n(u) \;:= \sup_{\substack{z \in \mathrm{star}(\mathcal{C}-x^*) \\ \|z\|_A \leq u}} |\frac{\sigma}{n}\sum_{i=1}^n g_i\,(Az)_i|,$$

Inclusion of events: We first claim that $\mathcal{B}(u) \subseteq \{V_n(u) \geq 2u^2\}$. Indeed, if $\mathcal{B}(u)$ occurs, then there exists some $z \in \mathcal{C} - x^*$ with $\|z\|_A \geq u$ and

$$|\frac{\sigma}{n}\sum_{i=1}^n g_i\,(Az)_i| \geq 2u\,\|z\|_A. \tag{50}$$

Define the rescaled vector $\widetilde{z} = \frac{u}{\|z\|_A} z$. Since $z \in \mathcal{C} - x^*$ and $\frac{u}{\|z\|_A} \leq 1$, the vector $\widetilde{z} \in \mathrm{star}(\mathcal{C} - x^*)$. Moreover, by construction, we have $\|\widetilde{z}\|_A = u$. When the inequality (50) holds, the vector $\widetilde{z}$ thus satisfies $|\frac{\sigma}{n}\sum_{i=1}^n g_i\,(A\widetilde{z})_i| \geq 2u^2$, which certifies that $V_n(u) \geq 2u^2$, as claimed.

Controlling the tail probability: The final step is to control the probability of the event $\{V_n(u) \geq 2u^2\}$. Viewed as a function of the standard Gaussian vector $(g_1, \ldots, g_n)$, it is easy to see that $V_n(u)$ is Lipschitz with constant $L = \frac{\sigma u}{\sqrt{n}}$. Consequently, by concentration of measure for Lipschitz Gaussian functions, we have

$$\mathbb{P}\big[V_n(u) \geq \mathbb{E}[V_n(u)] + u^2\big] \leq e^{-\frac{nu^2}{2\sigma^2}}. \tag{51}$$

In order to complete the proof, it suffices to show that $\mathbb{E}[V_n(u)] \leq u^2$. By definition, we have $\mathbb{E}[V_n(u)] = \frac{\sigma}{\sqrt{n}}\mathcal{W}_u(\mathcal{K}^*)$. Since $\mathcal{K}^*$ is a star-shaped set, the function $v \mapsto \mathcal{W}_v(\mathcal{K}^*)/v$ is non-increasing (Bartlett et al., 2005). Since $u \geq \varepsilon_n$, we have

$$\sigma\frac{\mathcal{W}_u(\mathcal{K}^*)}{u} \leq \sigma\frac{\mathcal{W}_{\varepsilon_n}(\mathcal{K}^*)}{\varepsilon_n} \;\leq\; \varepsilon_n.$$

where the final step follows from the definition of ε_n. Putting together the pieces, we conclude that $\mathbb{E}[V_n(u)] \leq \varepsilon_n u \leq u^2$ as claimed.

References

N. Ailon and B. Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 557–563. ACM, 2006.

Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 17–24, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273499. URL `http://doi.acm.org/10.1145/1273496.1273499`.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008. ISSN 0885-6125. doi: 10.1007/s10994-007-5040-8. URL `http://dx.doi.org/10.1007/s10994-007-5040-8`.

H. Avron, P. Maymounkov, and S. Toledo. Blendenpik: Supercharging lapack's least-squares solver. *SIAM Journal on Scientific Computing*, 32(3):1217–1236, 2010.

F. Bach. Consistency of trace norm minimization. *Journal of Machine Learning Research*, 9: 1019–1048, June 2008.

F. Bach, R. Jenatton, J. Mairal, G. Obozinski, et al. Convex optimization with sparsity-inducing norms. *Optimization for Machine Learning*, pages 19–53, 2011.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

L. Birgé. Estimating a density under order restrictions: Non-asymptotic minimax risk. *Annals of Statistics*, 15(3):995–1012, March 1987.

J. Bourgain, S. Dirksen, and J. Nelson. Toward a unified theory of sparse dimensionality reduction in euclidean space. *Geometric and Functional Analysis*, 25(4), 2015.

C. Boutsidis and P. Drineas. Random projections for the nonnegative least-squares problem. *Linear Algebra and its Applications*, 431(5–7):760–771, 2009.

F. Bunea, Y. She, and M. Wegkamp. Optimal selection of reduced rank estimators of high-dimensional matrices. *Annals of Statistics*, 39(2):1282–1309, 2011.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. Info Theory*, 51 (12):4203–4215, December 2005.

S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Computing*, 20(1):33–61, 1998.

K. L. Clarkson, P. Drineas, M. Magdon-Ismail, M. W. Majoney, X. Meng, and D. P. Woodruff. The fast cauchy transform and faster robust linear regression. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 466–477. SIAM, 2013.

T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, 1991.

P. Drineas and M. W. Mahoney. Effective resistances, statistical leverage, and applications to linear equation solving. *arXiv preprint arXiv:1005.3097*, 2010.

P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlos. Faster least squares approximation. *Numer. Math*, 117(2):219–249, 2011.

P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13(1): 3475–3506, 2012.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

M. Fazel. *Matrix Rank Minimization with Applications*. PhD thesis, Stanford, 2002. Available online: http://faculty.washington.edu/mfazel/thesis-final.pdf.

L. El Ghaoui, V. Viallon, and T. Rabbani. Safe feature elimination for the lasso. *Submitted, April*, 2011.

Y. Gordon, A. E. Litvak, S. Mendelson, and A. Pajor. Gaussian averages of interpolated bodies and applications to approximate reconstruction. *Journal of Approximation Theory*, 149:59–73, 2007.

P. Huber. Robust regression: Asymptotics, conjectures and Monte Carlo. *Annals of Statistics*, 1:799–821, 2001.

D. M. Kane and J. Nelson. Sparser Johnson-Lindenstrauss transforms. *Journal of the ACM*, 61(1), 2014.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*. Springer-Verlag, New York, NY, 1991.

P. Loh and M. J. Wainwright. High-dimensional regression with noisy and missing data: Provable guarantees with non-convexity. *Annals of Statistics*, 40(3):1637–1664, September 2012.

M.J. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba. Coding facial expressions with gabor wavelets. In *Proc. Int'l Conf. Automatic Face and Gesture Recognition*, pages 200–205, 1998.

M. W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning in Machine Learning*, 3(2), 2011.

H. M. Markowitz. *Portfolio Selection*. Wiley, New York, 1959.

S. Negahban and M. J. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *Annals of Statistics*, 39(2):1069–1097, 2011.

S. Negahban and M. J. Wainwright. Restricted strong convexity and (weighted) matrix completion: Optimal bounds with noise. *Journal of Machine Learning Research*, 13:1665–1697, May 2012.

V. Ojansivu and J. Heikkil. Blur insensitive texture classification using local phase quantization. In *Proc. Image and Signal Processing (ICISP 2008)*, pages 236–243, 2008.

M. R. Osborne, B. Presnell, and B. A. Turlach. On the Lasso and its dual. *Journal of Computational and Graphical Statistics*, 9(2):319–337, 2000.

M. Pilanci and M. J. Wainwright. Randomized sketches of convex programs with sharp guarantees. *IEEE Trans. Info. Theory*, 9(61):5096–5115, September 2015a.

M. Pilanci and M. J. Wainwright. Newton sketch: A linear-time optimization algorithm with linear-quadratic convergence. Technical report, UC Berkeley, 2015b. URL `http://arxiv.org/pdf/1505.02250.pdf`.

M. Pilanci, L. El Ghaoui, and V. Chandrasekaran. Recovery of sparse probability measures via convex programming. In *Advances in Neural Information Processing Systems*, pages 2420–2428, 2012.

G. Raskutti, M. J. Wainwright, and B. Yu. Minimax rates of estimation for high-dimensional linear regression over ℓ_q-balls. *IEEE Trans. Information Theory*, 57(10):6976—6994, October 2011.

B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

V. Rokhlin and M. Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proceedings of the National Academy of Sciences*, 105(36):13212–13217, 2008.

T. Sarlos. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 143–152. IEEE, 2006.

N. Srebro, N. Alon, and T. S. Jaakkola. Generalization error bounds for collaborative prediction with low-rank matrices. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2005.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

S. Vempala. *The Random Projection Method*. Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, Providence, RI, 2004.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *Compressed Sensing: Theory and Applications*, 2012.

T. T. Wu and K. Lange. Coordinate descent algorithms for Lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.

Y. Yang, M. Pilanci, and M. J. Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. Technical report, UC Berkeley, 2015. URL `http://arxiv.org/pdf/1501.06195.pdf`.

B. Yu. Assouad, Fano and Le Cam. In *Festschrift for Lucien Le Cam*, pages 423–435. Springer-Verlag, Berlin, 1997.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society B*, 1(68):49, 2006.

M. Yuan, A. Ekici, Z. Lu, and R. Monteiro. Dimension reduction and coefficient estimation in multivariate linear regression. *Journal Of The Royal Statistical Society Series B*, 69(3): 329–346, 2007.

Journal of Machine Learning Research 17 (2016) 1-23 Submitted 11/14; Revised 7/15; Published 4/16

Estimating Causal Structure Using Conditional DAG Models

Chris. J. Oates CHRISTOPHER.OATES@UTS.EDU.AU
School of Mathematical and Physical Sciences
University of Technology Sydney
NSW 2007, Australia

Jim Q. Smith J.Q.SMITH@WARWICK.AC.UK
Department of Statistics
University of Warwick
Coventry, CV4 7AL, UK

Sach Mukherjee SACH.MUKHERJEE@DZNE.DE
German Center for Neurodegenerative Diseases (DZNE)
53175 Bonn, Germany

Editor: Samuel Kaski

Abstract

This paper considers inference of causal structure in a class of graphical models called conditional DAGs. These are directed acyclic graph (DAG) models with two kinds of variables, primary and secondary. The secondary variables are used to aid in the estimation of the structure of causal relationships between the primary variables. We prove that, under certain assumptions, such causal structure is identifiable from the joint observational distribution of the primary and secondary variables. We give causal semantics for the model class, put forward a score-based approach for estimation and establish consistency results. Empirical results demonstrate gains compared with formulations that treat all variables on an equal footing, or that ignore secondary variables. The methodology is motivated by applications in biology that involve multiple data types and is illustrated here using simulated data and in an analysis of molecular data from the Cancer Genome Atlas.

Keywords: Graphical Models, Causal Inference, Directed Acyclic Graphs, Instrumental Variables, Data Integration

1. Introduction

This paper considers learning causal structure among a set of *primary variables* $(Y_i)_{i\in V}$, using additional *secondary variables* $(X_i)_{i\in W}$ to aid in estimation. The primary variables are those of direct scientific interest while the secondary variables are variables that are known to influence the primary variables, but whose mutual relationships are not of immediate interest and perhaps not amenable to estimation using the available data. As we discuss further below, the primary/secondary distinction is common in biostatistical applications and is often dealt with in an *ad hoc* manner, for example by leaving some relationships or edges implicit in causal diagrams. Our aim is to define a class of graphical models for this setting and to clarify the conditions under which secondary variables can aid in causal estimation. We focus on causal estimation in the sense of estimation of the presence or absence of edges in the causal graph rather than estimation of quantitative causal effects.

The fact that primary variables of direct interest are often part of a wider context, including additional secondary variables, presents challenges for graphical modelling and causal inference, since in general the secondary variables will not be independent and simply marginalising may introduce spurious dependencies (Evans and Richardson, 2014). Motivated by this observation, we define conditional DAG (CDAG) models and discuss their semantics. Nodes in a CDAG are of two kinds, corresponding to primary and secondary variables, and as detailed below the semantics of CDAGs allow causal inferences to be made about the primary variables $(Y_i)_{i\in V}$ whilst accounting for the secondary variables $(X_i)_{i\in W}$. To limit scope, we focus on the setting where each primary variable has a known cause among the secondary variables. Specifically we suppose there is a bijection $\phi : V \to W$, between the primary and secondary index sets V and W, such that for each $i \in V$ a direct[1] causal dependency $X_{\phi(i)} \to Y_i$ exists. Under explicit assumptions we show that such secondary variables can aid in causal inference for the primary variables, because known relationships between secondary and primary variables render "primary-to-primary" causal links of the form $Y_i \to Y_j$ identifiable from joint data on primary and secondary variables. We put forward score-based estimators of CDAG structure that we show are asymptotically consistent under certain conditions; importantly, independence assumptions on the secondary variables are not needed.

This work is motivated by current efforts in biology aimed at exploiting high-throughput data to better understand causal molecular mechanisms, such as those involved in gene regulation or protein signaling. A notable feature of molecular biology is the fact that some causal links are relatively clearly defined by known sequence specificity. For example, the DNA sequence has a causal influence on the level of corresponding mRNA; mRNAs have a causal influence on corresponding total protein levels; and total protein levels have a causal influence on levels of post-translationally modified protein. This means that in a study involving a certain molecular variable (a protein, say), a subset of the causal influences upon it may be clear at the outset (e.g. the corresponding mRNA) and typically it is the unknown influences that are the subject of the study. Then, it is natural to ask whether accounting for the known influences can aid in identification of the unknown influences. For example, if interest focuses on causal relationships between proteins, known mRNA-protein links could be exploited to aid in causal identification at the protein-protein level. We show below an example using protein data.

Our development of the CDAG can be considered dual to the acyclic directed mixed graphs (ADMGs) developed by Evans and Richardson (2014), in the sense that we investigate conditioning as an alternative to marginalisation. In this respect our work mirrors recently developed undirected graphical models called conditional graphical models (CGMs; Li *et al*, 2012; Cai *et al.*, 2013) In CGMs, Gaussian random variables $(Y_k)_{k\in V}$ satisfy

$$Y_i \perp\!\!\!\perp Y_j | (Y_k)_{k\in V\setminus\{i,j\}}, (X_k)_{k\in W} \text{ if and only if } (i,j) \notin G \tag{1}$$

where G is an undirected acyclic graph and $(X_k)_{k\in W}$ are auxiliary random variables that are conditioned upon. CGMs have recently been applied to gene expression data $(Y_i)_{i\in V}$. In that setting Zhang and Kim (2014) used single nucleotide polymorphisms (SNPs) as the

1. Throughout, we use the term "direct" in the sense of Pearl (2009) and note that the causal influence need not be *physically* direct.

$(X_i)_{i\in W}$, while Logsdon and Mezey (2010); Yin and Li (2011); Cai *et al.* (2013) used expression qualitative trait loci (e-QTL) as the $(X_i)_{i\in W}$. The latter work was recently extended to jointly estimate several such graphical models in Chun *et al.* (2013). Also in the context of undirected graphs, van Wieringen and van de Wiel (2014) recently considered encoding a bijection between DNA copy number and mRNA expression levels into inference. Our work complements these efforts by using directed models that are arguably more appropriate for causal inference (Pearl, 2009). Evans (2015) uses a similar directed graphical characterisation and nomenclature. However, secondary variables in Evans (2015) are unobserved and can have multiple children in the set of primary variables, while ours are observed and have exactly one child among the primary variables. CDAGs are also related to instrumental variables and Mendelian randomisation approaches (Didelez and Sheehan, 2007) that we discuss below (Section 2.2).

CDAGs share some similarity with the influence diagrams (IDs) introduced by Dawid (2002) as an extension of DAGs that distinguish between variable nodes and decision nodes. This generalised the augmented DAGs of Spirtes *et al.* (2000); Lauritzen (2000); Pearl (2009) in which each variable node is associated with a decision node that represents an intervention on the corresponding variable. However, the semantics of IDs are not well suited to the scientific contexts that we consider, where secondary nodes represent variables to be observed, not the outcomes of decisions. The notion of a non-atomic intervention (Pearl, 2003), where many variables are intervened upon simultaneously, shares similarity with CDAGs in the sense that the secondary variables are in general non-independent. However again the semantics differ, since our secondary nodes represent random variables rather than interventions. In a different direction, Neto *et al.* (2010) recently observed that the use of e-QTL data $(X_i)_{i\in W}$ can help to identify causal relationships among gene expression levels $(Y_i)_{i\in V}$. However, Neto *et al.* (2010) require independence of the $(X_i)_{i\in W}$; this is too restrictive for general settings, including in molecular biology, since the secondary variables will typically themselves be subject to regulation and far from independent. An important and novel aspect of our approach is that no independence or conditional independence assumptions need to be placed on the secondary variables in order to draw causal conclusions about the primary variables.

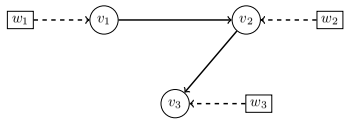
This paper begins in Sec. 2 by defining CDAGs and discussing identifiability of their structure from observational data on primary and secondary variables. Sufficient conditions are then given for consistent estimation of CDAG structure along with an algorithm based on integer linear programming. The methodology is illustrated in Section 3 on simulated data, including data sets that violate CDAG assumptions, and on protein data from cancer samples, the latter from the Cancer Genome Atlas (TCGA).

2. Methodology

Below we define CDAGS, study their theoretical properties and propose consistent estimators for CDAG structure.

2.1 A Statistical Framework for Conditional DAG Models

Consider index sets V, W and a bijection $\phi : V \to W$ between them. We will distinguish between the nodes in graphs and the random variables (RVs) that they represent. Specif-



(a)

type	index	node	variable
primary	$i \in V$	$v_i \in N(V)$	Y_i
secondary	$i \in W$	$w_i \in N(W)$	X_i

(b)

Figure 1: A conditional DAG model with primary nodes $N(V) = \{v_1, v_2, v_3\}$ and secondary nodes $N(W) = \{w_1, w_2, w_3\}$. Here primary nodes represent primary random variables $(Y_i)_{i\in V}$ and solid arrows correspond to a DAG G on these vertices. Square nodes are secondary variables $(X_i)_{i\in W}$ that, in the causal interpretation of CDAGs, represent known direct causes of the corresponding $(Y_i)_{i\in V}$ (dashed arrows represent known relationships; the random variables $(X_i)_{i\in W}$ need not be independent). The name conditional DAG refers to the fact that conditional upon $(X_i)_{i\in W}$, the solid arrows encode conditional independence relationships among the $(Y_i)_{i\in V}$.

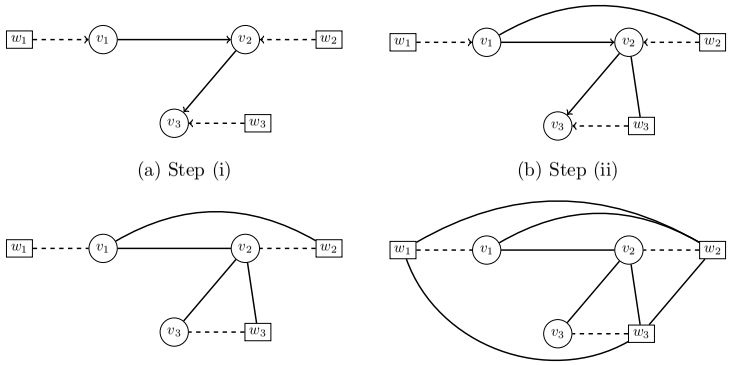
ically, indices correspond to nodes in graphical models; this is signified by the notation $N(V) = \{v_1, \ldots, v_p\}$ and $N(W) = \{w_1, \ldots, w_p\}$. Each node $v_i \in N(V)$ corresponds to a primary RV Y_i and similarly each node $w_i \in N(W)$ corresponds to a secondary RV X_i.

Definition 1 (CDAG) *A* conditional DAG *(CDAG)* $\overline{G}$*, with primary and secondary index sets* V*,* W *respectively and a bijection* ϕ *between them, is a DAG on the primary node set* $N(V)$ *with additional directed edges from each secondary node* $w_{\phi(i)} \in N(W)$ *to its corresponding primary node* $v_i \in N(V)$*.*

In other words, a CDAG $\overline{G}$ has node set $N(V) \cup N(W)$ and an edge set that can be generated by starting with a DAG on the primary nodes $N(V)$ and adding a directed edge from each secondary node in $N(W)$ to its corresponding primary node in $N(V)$, with the correspondence specified by the bijection ϕ. An example of a CDAG is shown in Fig. 1a. To further distinguish V and W in the graphical representation we employ circular and square vertices respectively. In addition we use dashed lines to represent edges that are required by definition and must therefore be present in any CDAG $\overline{G}$.

Since the DAG on the primary nodes $N(V)$ is of particular interest, throughout we use G to denote a DAG on $N(V)$. We use $\mathcal{G}$ to denote the set of all possible DAGs with $|V|$ vertices. For notational clarity, and without loss of generality, below we take the bijection to simply be the identity map $\phi(i) = i$. The parents of node v_i in a DAG G are indexed by $\mathrm{pa}_G(i) \subseteq V \setminus \{i\}$. Write $\mathrm{an}_{\overline{G}}(S)$ for the ancestors of nodes $S \subseteq N(V) \cup N(W)$ in the CDAG $\overline{G}$ (which by definition includes the nodes in S). For disjoint sets of nodes A, B, C in an undirected graph, we say that C *separates* A and B if every path between a node in A and a node in B in the graph contains a node in C.

Definition 2 (c-separation) *Consider disjoint* $A, B, C \subseteq N(V) \cup N(W)$ *and a CDAG* $\overline{G}$*. We say that* A *and* B *are* c-separated *by* C *in* $\overline{G}$*, written* $A \perp\!\!\!\perp B | C\ [\overline{G}]$*, when* C *separates*



(a) Step (i) (b) Step (ii)

(c) Step (iii) (d) Step (iv)

Figure 2: Illustrating c-separation. Here we ask whether v_3 and v_1 are c-separated by $\{v_2\}$ in $\overline{G}$, the CDAG shown in Fig. 1a. [Step (i): Take the subgraph induced by $\mathrm{an}_{\overline{G}}(\{v_1, v_2, v_3\})$. Step (ii): Moralise this subgraph (i.e. join with an undirected edge any parents of a common child that are not already connected by a directed edge). Step (iii): Take the skeleton of the moralised subgraph (i.e. remove the arrowheads). Step (iv): Add an undirected edge between every pair (w_i, w_j). In the final panel (d) we ask whether there exists a path from v_3 to v_1 that does not pass through v_2; the answer is positive (e.g. $v_3 - w_3 - w_1 - v_1$) and hence we conclude that $v_3 \not\perp\!\!\!\perp v_1 | v_2 [\overline{G}]$, i.e. v_3 and v_1 are not c-separated by $\{v_2\}$ in $\overline{G}$.]

A and B in the undirected graph U_4 that is formed as follows: (i) Take the subgraph U_1 induced by $\mathrm{an}_{\overline{G}}(A \cup B \cup C)$. (ii) Moralise U_1 to obtain U_2 (i.e. join with an undirected edge any parents of a common child that are not already connected by a directed edge). (iii) Take the skeleton of the moralised subgraph U_2 to obtain U_3 (i.e. remove the arrowheads). (iv) Add an undirected edge between every pair of nodes in $N(W)$ to obtain U_4.

The c-separation procedure is illustrated in Fig. 2, where we show that v_3 is not c-separated from v_1 by the set $\{v_2\}$ in the CDAG from Fig. 1a.

Remark 3 *The classical notion of d-separation for DAGs is equivalent to omitting step (iv) in c-separation. Notice that v_3* is *d-separated from v_1 by the set $\{v_2\}$ in the DAG G, underlining the need for a custom notion of separation for CDAGs.*

The topology (i.e. the set of edges) of a CDAG carries formal (potentially causal) semantics on the primary RVs, conditional on the secondary RVs, as specified below. Write $T(S)$ for the collection of triples $\langle A, B | C \rangle$ where A, B, C are disjoint subsets of S.

Definition 4 (Independence model) *The CDAG $\overline{G}$, together with c-separation, implies a formal* independence model *(p.12 of Studený, 2005)*

$$\mathcal{M}_G = \{\langle A, B | C \rangle \in T(N(V) \cup N(W)) : A \perp\!\!\!\perp B | C \, [\overline{G}]\} \tag{2}$$

where $\langle A, B | C \rangle \in \mathcal{M}_G$ carries the interpretation that the RVs corresponding to A are conditionally independent of the RVs corresponding to B when given the RVs corresponding to C. We will write $A \perp\!\!\!\perp B | C \, [\mathcal{M}_G]$ as a shorthand for $\langle A, B | C \rangle \in \mathcal{M}_G$.

Remark 5 *An independence model $\mathcal{M}_G$ does not contain any information on the structure of the marginal distribution $\mathbb{P}^{(X_i)}$ of the secondary variables, due to the additional step (iv) in c-separation.*

Lemma 6 (Equivalence classes) *The map $G \mapsto \mathcal{M}_G$ is an injection.*

Proof Consider two distinct DAGs $G, H \in \mathcal{G}$ and suppose that, without loss of generality, the edge $v_i \to v_j$ belongs to G and not to H. It suffices to show that $\mathcal{M}_G \neq \mathcal{M}_H$. First notice that G represents the relations (i) $w_i \not\perp\!\!\!\perp v_j | w_j \, [\overline{G}]$, (ii) $w_i \not\perp\!\!\!\perp v_j | w_j, (v_k)_{k \in V \setminus \{i,j\}} \, [\overline{G}]$, and (iii) $w_j \perp\!\!\!\perp v_i | w_i \, [\overline{G}]$. (These can each be directly verified by c-separation.) We show below that H cannot also represent (i-iii) and hence, from Def. 4, it follows that $\mathcal{M}_G \neq \mathcal{M}_H$. We distinguish between two cases for H, namely (a) $v_i \leftarrow v_j \notin H$, and (b) $v_i \leftarrow v_j \in H$.

Case (a): Suppose (i) also holds for H; that is, $w_i \not\perp\!\!\!\perp v_j | w_j \, [\overline{H}]$. Then since $v_i \to v_j \notin H$, it follows from c-separation that the variable v_i must be connected to v_j by directed path(s) whose interior vertices must only belong to $N(V) \setminus \{v_i, v_j\}$. Thus H implies the relation $w_i \perp\!\!\!\perp v_j | w_j, (v_k)_{k \in V \setminus \{i,j\}} \, [\overline{H}]$, so that (ii) cannot also hold.

Case (b): Since $v_i \leftarrow v_j \in H$ it follows from c-separation that $w_j \not\perp\!\!\!\perp v_i | w_i \, [\overline{H}]$, so that (iii) does not hold for H. ■

Remark 7 *More generally the same argument shows that a DAG $G \in \mathcal{G}$ satisfies $v_i \to v_j \notin G$ if and only if $\exists S \subseteq pa_G(j) \setminus \{i\}$ such that $w_i \perp\!\!\!\perp v_j | w_j, (v_k)_{k \in S} \, [\overline{G}]$. As a consequence, we have the interpretation that conditional upon the (secondary variables) $(X_i)_{i \in W}$, the solid arrows in Fig. 1a encode conditional independence relationships among the (primary variables) $(Y_i)_{i \in V}$, motivating the CDAG nomenclature.*

It is well known that conditional independence (and causal) relationships can usefully be described through a qualitative, graphical representation. However to be able to use a graph for *reasoning* it is necessary for that graph to embody certain assertions that themselves obey a logical calculus. Pearl and Verma (1987) proposed such a set of rules (the semi-graphoid axioms) that any reasonable set of assertions about how one set of variables might be irrelevant to the prediction of a second, given the values of a third, might hold (see also Dawid, 2001; Studený, 2005). This can then be extended to causal assertions (Pearl, 2009), thereby permitting study of causal hypotheses and their consequences without the need to first construct elaborate probability spaces and their extensions under control. Below we establish that the independence models $\mathcal{M}_G$ induced by c-separation on CDAGs are semi-graphoids and thus enable reasoning in the present setting with two kinds of variables:

Lemma 8 (Semi-graphoid) *For any DAG $G \in \mathcal{G}$, the set $\mathcal{M}_G$ is semi-graphoid (Pearl and Paz, 1985). That is to say, for all disjoint $A, B, C, D \subseteq N(V) \cup N(W)$ we have*

(i) (triviality) $A \perp\!\!\!\perp \emptyset | C$ $[\mathcal{M}_G]$

(ii) (symmetry) $A \perp\!\!\!\perp B | C$ $[\mathcal{M}_G]$ implies $B \perp\!\!\!\perp A | C$ $[\mathcal{M}_G]$

(iii) (decomposition) $A \perp\!\!\!\perp B, D | C$ $[\mathcal{M}_G]$ implies $A \perp\!\!\!\perp D | C$ $[\mathcal{M}_G]$

(iv) (weak union) $A \perp\!\!\!\perp B, D | C$ $[\mathcal{M}_G]$ implies $A \perp\!\!\!\perp B | C, D$ $[\mathcal{M}_G]$

(v) (contraction) $A \perp\!\!\!\perp B | C, D$ $[\mathcal{M}_G]$ and $A \perp\!\!\!\perp D | C$ $[\mathcal{M}_G]$ implies $A \perp\!\!\!\perp B, D | C$ $[\mathcal{M}_G]$

Proof The simplest proof is to note that our notion of c-separation is equivalent to classical d-separation applied to an extension $\underline{G}$ of the CDAG $\overline{G}$. The semi-graphoid properties then follow immediately by the facts that (i) d-separation satisfies the semi-graphoid properties (p.48 of Studený, 2005), and (ii) the restriction of a semi-graphoid to a subset of vertices is itself a semi-graphoid (p.14 of Studený, 2005).

Construct an extended graph $\underline{G}$ from the CDAG $\overline{G}$ by the addition of a node z and directed edges from z to each of the secondary vertices $N(W)$. Then for disjoint $A, B, C \subseteq N(V) \cup N(W)$ we have that A and B are c-separated by C in $\overline{G}$ if and only if A and B are d-separated by C in $\underline{G}$. This is because every path in the undirected graph $U_4(\overline{G})$ (recall the definition of c-separation) that contains an edge $w_i \to w_j$ corresponds uniquely to a path in $U_3(\underline{G})$ that contains the sub-path $w_i \to z \to w_j$. ■

2.2 Causal CDAG Models

The previous section defined CDAG models using the framework of formal independence models. However, CDAGs can also be embellished with a causal interpretation, that we make explicit below. In this paper we make a causal sufficiency assumption that the $(X_i)_{i \in W}$ are the only source of confounding for the $(Y_i)_{i \in V}$ and below we talk about direct causes at the level of $(X_i)_{i \in W} \cup (Y_i)_{i \in V}$.

Definition 9 (Causal CDAG) *A CDAG is* causal *when an edge $v_i \to v_j$ exists if and only if Y_i is a direct cause of Y_j. It is further assumed that X_i is a direct cause of Y_i and not a direct cause of Y_j for $j \neq i$. Finally it is assumed that no Y_i is a direct cause of any X_j.*

Remark 10 *Here* direct cause *is understood to mean that the parent variable has a "controlled direct effect" on the child variable in the framework of Pearl (e.g. Def. 4.5.1 of Pearl, 2009) (it is not necessary that the effect is physically direct). No causal assumptions are placed on interaction between the secondary variables $(X_i)_{i \in W}$.*

Remark 11 *In a causal CDAG the secondary variables $(X_i)_{i \in W}$ share some of the properties of instrumental variables (Didelez and Sheehan, 2007). Consider estimating the average causal effect of Y_i on Y_j. Then, conditioning on X_j in the following, X_i can be used as a natural experiment (Greenland, 2000) to determine the size and sign of this causal effect.*

When we are interested in the controlled direct effect, we can repeat this argument with additional conditioning on the $(Y_k)_{k \in V \setminus \{i,j\}}$ (or a smaller subset of conditioning variables if the structure of G is known).

2.3 Identifiability of CDAGs

There exist well-known identifiability results for independence models $\mathcal{M}$ that are induced by Bayesian networks; see for example Spirtes *et al.* (2000); Pearl (2009). These relate the observational distribution $\mathbb{P}^{(Y_i)}$ of the random variables $(Y_i)_{i \in V}$ to an appropriate DAG representation by means of d-separation, Markov and faithfulness assumptions (discussed below). The problem of identification for CDAGs is complicated by the fact that (i) the primary variables $(Y_i)_{i \in V}$ are insufficient for identification, (ii) the joint distribution $\mathbb{P}^{(X_i) \cup (Y_i)}$ of the primary variables $(Y_i)_{i \in V}$ and the secondary variables $(X_i)_{i \in W}$ need not be Markov with respect to the CDAG $\overline{G}$, and (iii) we must work with the alternative notion of c-separation. Below we propose novel "partial" Markov and faithfulness conditions that will permit, in the next section, an identifiability theorem for CDAGs. We make the standard assumption that there is no selection bias (for example by conditioning on common effects). We also assume throughout that there exists a true CDAG $\overline{G}$. In other words, the observational distribution $\mathbb{P}^{(X_i) \cup (Y_i)}$ induces an independence model that can be expressed as $\mathcal{M}_G$ for some DAG $G \in \mathcal{G}$.

Definition 12 (Partial Markov) *Let G denote the true DAG. We say that the observational distribution $\mathbb{P}^{(X_i) \cup (Y_i)}$ is* partially Markov *with respect to G when the following holds: For all disjoint subsets $\{i\}, \{j\}, C \subseteq \{1, \ldots, p\}$ we have $w_i \perp\!\!\!\perp v_j | w_j, (v_k)_{k \in C}$ $[\mathcal{M}_G]$ $\Rightarrow$ $X_i \perp\!\!\!\perp Y_j | X_j, (Y_k)_{k \in C}$.*

Definition 13 (Partial faithfulness) *Let G denote the true DAG. We say that the observational distribution $\mathbb{P}^{(X_i) \cup (Y_i)}$ is* partially faithful *with respect to G when the following holds: For all disjoint subsets $\{i\}, \{j\}, C \subseteq \{1, \ldots, p\}$ we have $w_i \perp\!\!\!\perp v_j | w_j, (v_k)_{k \in C}$ $[\mathcal{M}_G] \Leftarrow$ $X_i \perp\!\!\!\perp Y_j | X_j, (Y_k)_{k \in C}$*

Remark 14 *The partial Markov property implies that $\mathbb{P}^{(X_i) \cup (Y_i)}$ admits the factorisation*

$$p((x_i)_{i=1,\ldots,p}, (y_i)_{i=1,\ldots,p}) = p((x_i)_{i=1,\ldots,p}) \prod_{i=1}^{p} p(y_i | (y_k)_{k \in pa_G(i)}, x_i), \tag{3}$$

while partial faithfulness ensures that none of the terms inside the product in Eqn. 3 can be further decomposed. In this work we will prove only estimation consistency; for more refined convergence analysis the partial faithfulness property must be strengthened. See e.g. Uhler et al. (2013) for a recent discussion of faithfulness in the DAG setting.

Remark 15 *The partial Markov and partial faithfulness properties do not place any constraint on the marginal distribution $\mathbb{P}^{(X_i)}$ of the secondary variables.*

The following is an immediate corollary of Lem. 6:

Theorem 16 (Identifiability) *Suppose that the observational distribution $\mathbb{P}^{(X_i) \cup (Y_i)}$ is partially Markov and partially faithful with respect to the true DAG G. Then*

(i) It is not possible to identify the true DAG G based on the observational distribution $\mathbb{P}^{(Y_i)}$ of the primary variables alone.

(ii) It is possible to identify the true DAG G based on the observational distribution $\mathbb{P}^{(X_i)\cup(Y_i)}$.

Proof (i) We have already seen that $\mathbb{P}^{(Y_i)}$ is not Markov with respect to the DAG G: Indeed a statistical association $Y_i \not\perp\!\!\!\perp Y_j|(Y_k)_{k\in V\setminus\{i,j\}}$ observed in the distribution $\mathbb{P}^{(Y_i)}$ could either be due to a direct interaction $Y_i \to Y_j$ (or $Y_j \to Y_i$), or could be mediated entirely through variation in the secondary variables $(X_k)_{k\in W}$. (ii) It follows immediately from Lemma 6 that observation of both the primary and secondary variables $(Y_i)_{i\in V} \cup (X_i)_{i\in W}$ is sufficient for identification of G. ■

2.4 Estimating CDAGs From Data

In this section we assume that the partial Markov and partial faithfulness properties hold, so that the true DAG G is identifiable from the joint observational distribution of the primary and secondary variables. Below we consider score-based estimation for CDAGs and prove consistency of certain score-based CDAG estimators.

Definition 17 (Score function; Chickering (2003)) *A* score function *is a map $S : \mathcal{G} \to [0,\infty)$ with the interpretation that if two DAGs $G, H \in \mathcal{G}$ satisfy $S(G) < S(H)$ then H is preferred to G.*

We will study the asymptotic behaviour of $\hat{G}_S$, the estimate of graph structure obtained by maximising $S(G)$ over all $G \in \mathcal{G}$ based on observations $(X_i^j, Y_i^j)_{i=1,\dots,p}^{j=1,\dots,n}$. Let $\mathbb{P}_n = \mathbb{P}^{(X_i^j, Y_i^j)}$ denote the finite-dimensional distribution of the n observations.

Definition 18 (Partial local consistency) *We say the score function S is* partially locally consistent *if, whenever H is constructed from G by the addition of one edge $Y_i \to Y_j$, we have*

1. $X_i \not\perp\!\!\!\perp Y_j|X_j, (Y_k)_{k\in pa_G(j)} \Rightarrow \lim_{n\to\infty} \mathbb{P}_n[S(H) > S(G)] = 1$
2. $X_i \perp\!\!\!\perp Y_j|X_j, (Y_k)_{k\in pa_G(j)} \Rightarrow \lim_{n\to\infty} \mathbb{P}_n[S(H) < S(G)] = 1.$

Theorem 19 (Consistency) *If S is partially locally consistent then $\lim_{n\to\infty} \mathbb{P}_n[\hat{G}_S = G] = 1$, so that $\hat{G}_S$ is a consistent estimator of the true DAG G.*

Proof It suffices to show that $\lim_{n\to\infty} \mathbb{P}_n[\hat{G}_S = H] = 0$ whenever $H \neq G$. There are two cases to consider:

Case (a): Suppose $v_i \to v_j \in H$ but $v_i \to v_j \notin G$. Let H' be obtained from H by the removal of $v_i \to v_j$. From c-separation we have $w_i \perp\!\!\!\perp v_j|w_j, (v_k)_{k\in \mathrm{pa}_G(j)}$ $[\overline{G}]$ and hence from the partial Markov property we have $X_i \perp\!\!\!\perp Y_j|X_j, (Y_k)_{k\in \mathrm{pa}_G(j)}$. Therefore if S is partially locally consistent then $\lim_{n\to\infty} \mathbb{P}_n[S(H) < S(H')] = 1$, so that $\lim_{n\to\infty} \mathbb{P}_n[\hat{G}_S = H] = 0$.

Case (b): Suppose $v_i \to v_j \notin H$ but $v_i \to v_j \in G$. Let H' be obtained from H by the addition of $v_i \to v_j$. From c-separation we have $w_i \not\perp\!\!\!\perp v_j|w_j, (v_k)_{k\in \mathrm{pa}_G(j)}$ $[\overline{G}]$ and hence from the partial faithfulness property we have $X_i \not\perp\!\!\!\perp Y_j|X_j, (Y_k)_{k\in \mathrm{pa}_G(j)}$. Therefore if S is partially locally consistent then $\lim_{n\to\infty} \mathbb{P}_n[S(H) < S(H')] = 1$, so that $\lim_{n\to\infty} \mathbb{P}_n[\hat{G}_S = H] = 0$. ■

Remark 20 *In this paper we adopt a maximum a posteriori (MAP) -Bayesian approach and consider score functions given by a posterior probability $p(G|(x_i^l, y_i^l)_{i=1,\dots,p}^{l=1,\dots,n})$ of the DAG G given the data $(x_i^l, y_i^l)_{i=1,\dots,p}^{l=1,\dots,n}$. This requires that a prior $p(G)$ is specified over the space $\mathcal{G}$ of DAGs. From the partial Markov property we have that, for n independent observations, such score functions factorise as*

$$S(G) = p(G) \prod_{l=1}^{n} p((x_i^l)_{i=1,\dots,p}) \prod_{i=1}^{p} p(y_i^l|(y_k^l)_{k\in pa_G(i)}, x_i^l). \tag{4}$$

We further assume that the DAG prior $p(G)$ factorises over parent sets $pa_G(i) \subseteq V \setminus \{i\}$ as

$$p(G) = \prod_{i=1}^{p} p(pa_G(i)). \tag{5}$$

This implies that the score function in Eqn. 4 is decomposable *and the maximiser $\hat{G}_S$, i.e. the MAP estimate, can be obtained via integer linear programming (ILP). In Sec. 2.6 we derive an ILP that targets the CDAG $\hat{G}_S$ and thereby allows exact (i.e. deterministic) estimation in this class of models.*

Lemma 21 *A score function of the form Eqn. 4 is partially locally consistent if and only if, whenever H is constructed from G by the addition of one edge $v_i \to v_j$, we have*

1. $X_i \not\perp\!\!\!\perp Y_j|X_j, (Y_k)_{k\in pa_G(j)} \Rightarrow \lim_{n\to\infty} \mathbb{P}_n[B_{H,G} > 1] = 1$
2. $X_i \perp\!\!\!\perp Y_j|X_j, (Y_k)_{k\in pa_G(j)} \Rightarrow \lim_{n\to\infty} \mathbb{P}_n[B_{H,G} < 1] = 1$

where

$$B_{H,G} = \frac{p((Y_j^l)^{l=1,\dots,n}|(Y_k^l)_{k\in pa_H(j)}^{l=1,\dots,n}, (X_j^l)^{l=1,\dots,n})}{p((Y_j^l)^{l=1,\dots,n}|(Y_k^l)_{k\in pa_G(j)}^{l=1,\dots,n}, (X_j^l)^{l=1,\dots,n})} \tag{6}$$

is the Bayes factor between two competing local models $pa_G(j)$ and $pa_H(j)$.

2.5 Bayes Factors and Common Variables

The characterisation in Lemma 21 justifies the use of any consistent Bayesian variable selection procedure to obtain a score function. To be clear, although our local scores derive from the variable selection literature, we do *not* perform node-wise variable selection, which would not produce a DAG in general. Instead, the local scores form the basis for a global search over DAGs via ILPs; see Sec. 2.6.

The secondary variables $(X_i^l)^{l=1,\ldots,n}$ are included in all models, and parameters relating to these variables should therefore share a common prior. Below we discuss a formulation of the Bayesian linear model that is suitable for CDAGs. Consider variable selection for node j and candidate parent (index) set $\mathrm{pa}_G(j) = \pi \subseteq V \setminus \{j\}$. We construct a linear model for the observations

$$Y_j^l = [1\ X_j^l]\boldsymbol{\beta}_0 + \boldsymbol{Y}_\pi^l \boldsymbol{\beta}_\pi + \epsilon_j^l, \quad \epsilon_j^l \sim N(0, \sigma^2) \tag{7}$$

where $\boldsymbol{Y}_\pi^l = (Y_k^l)_{k\in\pi}$ is used to denote a row vector and the noise ϵ_j^l is assumed independent for $j = 1, \ldots, p$ and $l = 1, \ldots, n$. Although suppressed in the notation, the parameters $\boldsymbol{\beta}_0$, $\boldsymbol{\beta}_\pi$ and σ are specific to node j. This regression model can be written in vectorised form as

$$\boldsymbol{Y}_j = \boldsymbol{M}_0\boldsymbol{\beta}_0 + \boldsymbol{Y}_\pi\boldsymbol{\beta}_\pi + \boldsymbol{\epsilon} \tag{8}$$

where $\boldsymbol{M}_0$ is the $n \times 2$ matrix whose rows are the $[1\ X_j^l]$ for $l = 1, \ldots, n$ and $\boldsymbol{Y}_\pi$ is the matrix whose rows are $\boldsymbol{Y}_\pi^l$ for $l = 1, \ldots, n$.

We orthogonalize the regression problem by defining $\boldsymbol{M}_\pi = (\boldsymbol{I} - \boldsymbol{M}_0(\boldsymbol{M}_0^T\boldsymbol{M}_0)^{-1}\boldsymbol{M}_0^T)\boldsymbol{Y}_\pi$ so that the model can be written as

$$\boldsymbol{Y}_j = \boldsymbol{M}_0\tilde{\boldsymbol{\beta}}_0 + \boldsymbol{M}_\pi\tilde{\boldsymbol{\beta}}_\pi + \boldsymbol{\epsilon} \tag{9}$$

where $\tilde{\boldsymbol{\beta}}_0$ and $\tilde{\boldsymbol{\beta}}_\pi$ are Fisher orthogonal parameters (see Deltell, 2011, for details).

In the conventional approach of Jeffreys (1961), the prior distribution is taken as

$$p_{j,\pi}(\tilde{\boldsymbol{\beta}}_\pi, \tilde{\boldsymbol{\beta}}_0, \sigma) = p_{j,\pi}(\tilde{\boldsymbol{\beta}}_\pi|\tilde{\boldsymbol{\beta}}_0, \sigma)p_j(\tilde{\boldsymbol{\beta}}_0, \sigma) \tag{10}$$

$$p_j(\tilde{\boldsymbol{\beta}}_0, \sigma) \propto \sigma^{-1} \tag{11}$$

where Eqn. 11 is the reference or independent Jeffreys prior. (For simplicity of exposition we leave conditioning upon $\boldsymbol{M}_0$, and $\boldsymbol{M}_\pi$ implicit.) The use of the reference prior here is motivated by the observation that the common parameters $\boldsymbol{\beta}_0, \sigma$ have the same meaning in each model π for variable Y_j and should therefore share a common prior distribution (Jeffreys, 1961). Alternatively, the prior can be motivated by invariance arguments that derive $p(\boldsymbol{\beta}_0, \sigma)$ as a right Haar measure (Bayarri *et al.*, 2012). Note however that σ does not carry the same meaning across $j \in V$ in the application that we consider below, so that the prior is specific to fixed j. For the parameter prior $p_{j,\pi}(\tilde{\boldsymbol{\beta}}_\pi|\tilde{\boldsymbol{\beta}}_0, \sigma)$ we use the g-prior (Zellner, 1986)

$$\tilde{\boldsymbol{\beta}}_\pi|\tilde{\boldsymbol{\beta}}_0, j, \pi \sim N(\boldsymbol{0}, g\sigma^2(\boldsymbol{M}_\pi^T\boldsymbol{M}_\pi)^{-1}) \tag{12}$$

where g is a positive constant to be specified. Due to orthogonalisation, $\mathrm{cov}(\hat{\boldsymbol{\beta}}_\pi) = \sigma^2(\boldsymbol{M}_\pi^T\boldsymbol{M}_\pi)^{-1}$ where $\hat{\boldsymbol{\beta}}_\pi$ is the maximum likelihood estimator for $\tilde{\boldsymbol{\beta}}_\pi$, so that the prior is specified on the correct length scale (Deltell, 2011). We note that many alternatives to Eqn. 12 are available in the literature (including Johnson and Rossell, 2010; Bayarri *et al.*, 2012). For discrete data we mention recent work by Massam and Wesołowski (2014).

Under the prior specification above, the marginal likelihood for a candidate model π has the following closed-form expression:

$$p_j(\boldsymbol{y}_j|\pi) = \frac{1}{2}\Gamma\left(\frac{n-2}{2}\right)\frac{1}{\pi^{(n-2)/2}}\frac{1}{|\boldsymbol{M}_0^T\boldsymbol{M}_0|^{1/2}}\left(\frac{1}{g+1}\right)^{|\pi|/2} b^{-(n-2)/2} \tag{13}$$

$$b = \boldsymbol{y}_j^T\left(\boldsymbol{I} - \boldsymbol{M}_0(\boldsymbol{M}_0^T\boldsymbol{M}_0)^{-1}\boldsymbol{M}_0^T - \frac{g}{g+1}\boldsymbol{M}_\pi(\boldsymbol{M}_\pi^T\boldsymbol{M}_\pi)^{-1}\boldsymbol{M}_\pi^T\right)\boldsymbol{y}_j \tag{14}$$

Following Scott and Berger (2010), we control multiplicity via the prior

$$p(\pi) \propto \binom{p}{|\pi|}^{-1}. \tag{15}$$

Proposition 22 (Consistency) *Let $g = n$. Then the Bayesian score function $S(G)$ defined above is partially locally consistent, and hence the corresponding estimator $\hat{G}_S$ is consistent.*

Proof This result is an immediate consequence of Lemma 21 and the well-known variable selection consistency property for the unit-information g-prior (see e.g. Fernández *et al.*, 2001). ■

2.6 Computation via Integer Linear Programming

Structure learning for DAGs is a well-studied problem, with contributions including Friedman and Koller (2003); Silander and Myllymäkki (2006); Tsamardinos *et al.* (2006); Cowell (2009); Cussens (2011); Yuan and Malone (2013). Discrete optimisation via ILP can be used to allow efficient estimation for graphical models, exploiting the availability of powerful (and exact) ILP solvers (Nemhauser and Wolsey, 1988; Wolsey, 1998; Achterberg, 2009), as discussed in Bartlett and Cussens (2013). Below we extend the ILP approach to CDAG models.

We begin by computing and caching the quantities

$$p((y_i^l)^{l=1,\ldots,n}|(y_k^l)_{k\in\pi}^{l=1,\ldots,n}, (x_i^l)^{l=1,\ldots,n}) \tag{16}$$

that summarise evidence in the data for the local model $\mathrm{pa}_G(i) = \pi \subseteq V \setminus \{i\}$ for variable i. These cached quantities are transformed to obtain "local scores"

$$s(i, \pi) := \log(p((y_i^l)^{l=1,\ldots,n}|(y_k^l)_{k\in\pi}^{l=1,\ldots,n}, (x_i^l)^{l=1,\ldots,n})) + \log(p(\pi)). \tag{17}$$

These are the (log-) evidence from Eqn. 16 with an additional penalty term that provides multiplicity correction over varying $\pi \subseteq V \setminus \{i\}$, arising from Eqn. 5. Then we define binary indicator variables that form the basis of our ILP as follows:

$$\Pi(i, \pi) := [\mathrm{pa}_G(i) = \pi] \quad \forall i = 1, \ldots, p,\ \pi \subseteq V \setminus \{i\}. \tag{18}$$

The information in the variables Π contains all information on the DAG G. However it will be necessary to impose constraints that ensure the Π correspond to a well-defined DAG:

$$\sum_{\pi\subseteq V\setminus\{i\}} \Pi(i, \pi) = 1 \quad \forall i = 1, \ldots, p \qquad \text{(C1; convexity)}$$

Constraint (C1) requires that every node i has exactly one parent set (i.e. there is a well-defined graph G). To ensure G is acyclic we require further constraints:

$$\sum_{i\in C}\sum_{\substack{\pi\subseteq V\setminus\{i\}\\ \pi\cap C=\emptyset}} \Pi(i, \pi) \geq 1 \quad \forall\, C \subseteq V, C \neq \emptyset. \qquad \text{(C2; acyclicity)}$$

(C2) states that for every non-empty set C there must be at least one node in C that does not have a parent in C.

Proposition 23 *The MAP estimate $\hat{G}_S$ is characterised as the solution of the ILP*

$$\hat{G}_S = \arg\max_{G\in\mathcal{G}} \sum_{i=1}^{p} \sum_{\pi\subseteq V\setminus\{i\}} s(i,\pi)\Pi(i,\pi) \tag{19}$$

subject to constraints (C1) and (C2).

Proof It was proven in Jaakola *et al.* (2010) that (C1) and (C2) together exactly characterise the space $\mathcal{G}$ of DAGs. ■

For the applications in this paper, all ILP instances were solved using the GOBNILP software that is freely available to download from `http://www.cs.york.ac.uk/aig/sw/gobnilp/`.

3. Results

Below we present results based on simulated data and data from molecular biology.

3.1 Simulated Data

We simulated data from linear-Gaussian structural equation models (SEMs). Here we summarise the simulation procedure, with full details provided in the supplement. We first sampled a DAG G for the primary variables and a second DAG G' for the secondary variables (independently of G), following a sampling procedure described in the supplement. That is, G is the causal structure of interest, while G' governs dependence between the secondary variables. Data for the secondary variables $(X_i)_{i\in W}$ were generated from an SEM with structure G'. The strength of dependence between secondary variables was controlled by a parameter $\theta \in [0,1]$. Here $\theta = 0$ renders the secondary variables independent and $\theta = 1$ corresponds to a deterministic relationship between secondary variables, with intermediate values of θ giving different degrees of covariation among the secondary variables. Finally, conditional on the $(X_i)_{i\in W}$, we simulated data for the primary variables $(Y_i)_{i\in V}$ from an SEM with structure G. To manage computational burden, for all estimators we considered only models of size $|\pi| \leq 5$. Performance was quantified by the structural Hamming distance (SHD) between the estimated DAG $\hat{G}_S$ and true, data-generating DAG G, taking into account directionality; we report the mean SHD as computed over 10 independent realisations of the data. We emphasise that the secondary variables need not be generated from a DAG model, however this is a convenient and familiar approach. We note that in the biological application below a DAG for the secondary variables might not be appropriate.

In Table 1 we compare the proposed score-based CDAG estimator with the corresponding score-based DAG estimator that uses only the primary variable data $(Y_i^l)_{i=1,\ldots,p}^{l=1,\ldots,n}$. We also considered an alternative (DAG2) where a standard DAG estimator is applied to *all* of the variables $(X_i)_{i\in W}$, $(Y_i)_{i\in V}$, with the subgraph induced on the primary variables giving the estimate for G. We considered values $\theta = 0,\ 0.5,\ 0.99$ corresponding to zero, mild

$\theta = 0$	$n = 10$			$n = 100$			$n = 1000$		
	DAG	DAG2	CDAG	DAG	DAG2	CDAG	DAG	DAG2	CDAG
$p = 5$	2.9 ± 0.48	2.6 ± 0.48	3.4 ± 0.56	3.2 ± 0.81	1.9 ± 0.43	0.8 ± 0.25	3 ± 0.76	1.6 ± 0.48	0.3 ± 0.21
$p = 10$	9.8 ± 0.55	9.2 ± 0.66	8.8 ± 0.81	8.2 ± 0.99	5 ± 0.84	2.8 ± 0.51	5.5 ± 1.2	5.4 ± 1.1	0.3 ± 0.15
$p = 15$	15 ± 1.3	14 ± 1.1	15 ± 1.2	11 ± 1	6.8 ± 0.83	4.4 ± 0.58	6.3 ± 1.5	8.2 ± 0.92	0.8 ± 0.25
$\theta = 0.5$	$n = 10$			$n = 100$			$n = 1000$		
	DAG	DAG2	CDAG	DAG	DAG2	CDAG	DAG	DAG2	CDAG
$p = 5$	5.7 ± 0.56	4.5 ± 0.48	4 ± 0.49	3.9 ± 0.75	2.1 ± 0.62	0.6 ± 0.31	3.8 ± 0.74	1.8 ± 0.81	0 ± 0
$p = 10$	9.8 ± 0.96	8.1 ± 0.95	7.8 ± 1.1	7.2 ± 1.7	4.6 ± 1.2	1.7 ± 0.84	7.9 ± 1.3	3 ± 0.52	0.6 ± 0.31
$p = 15$	14 ± 0.85	13 ± 0.86	13 ± 0.7	14 ± 1.1	6.2 ± 0.55	3.6 ± 0.76	11 ± 0.93	7.5 ± 1.7	1 ± 0.45
$\theta = 0.99$	$n = 10$			$n = 100$			$n = 1000$		
	DAG	DAG2	CDAG	DAG	DAG2	CDAG	DAG	DAG2	CDAG
$p = 5$	5.1 ± 0.72	4 ± 0.56	4.2 ± 0.49	4.7 ± 0.79	2.1 ± 0.5	0.5 ± 0.31	2.7 ± 0.42	1.7 ± 0.56	0.9 ± 0.48
$p = 10$	9.1 ± 0.84	8.1 ± 0.72	7.8 ± 1.3	9.7 ± 1.1	3.7 ± 0.52	2.5 ± 0.45	9.4 ± 1	4.4 ± 0.64	0.3 ± 0.3
$p = 15$	15 ± 1.1	13 ± 0.94	13 ± 1.2	13 ± 1.4	6.6 ± 0.91	4.7 ± 0.86	17 ± 1.8	9.1 ± 0.69	0.7 ± 0.4

Table 1: Simulated data results. Here we display the mean structural Hamming distance from the estimated to the true graph structure, computed over 10 independent realisations, along with corresponding standard errors. [Data were generated using linear-Gaussian structural equations. $\theta \in [0,1]$ quantifies dependence between the secondary variables $(X_i)_{i\in W}$, n is the number of data points and p is the number of primary variables $(Y_i)_{i\in V}$. "DAG" = estimation based only on primary variables $(Y_i)_{i\in V}$, "DAG2" = estimation based on the full data $(X_i)_{i\in W}\cup(Y_i)_{i\in V}$, "CDAG" = estimation based on the full data and enforcing CDAG structure.]

and strong covariation among the secondary variables. In each data-generating regime we found that CDAGs were either competitive with, or (typically) more effective than, the DAG and DAG2 estimators. In the $p = 15$, $n = 1000$ regime (that is closest to the biological application that we present below) the CDAG estimator dramatically outperforms these two alternatives. Inference using DAG2 and CDAG (that both use primary as well as secondary variables) is robust to large values of θ, whereas in the $p = 15$, $n = 1000$ regime, the performance of the DAG estimator based on only the primary variables deteriorates for large θ. This agrees with intuition since association between the secondary X_i's may induce correlations between the primary Y_i's. CDAGs outperformed DAG2 at large n (see also Table 1).

To better understand the limitations of CDAGs we considered three data-generating regimes that violate the CDAG assumptions. Firstly we focused on the $\theta = 0$, $p = 15$, $n = 1000$ regime where the CDAG estimator performs well when data are generated "from the model". We then introduced a number E of edges of the form $X_i \to Y_j$ where $Y_i \to Y_j \in G$. These edges (strongly) violate the structural assumptions implied by the CDAG model because their presence means that X_i is no longer a suitable instrument for $Y_i \to Y_j$ as it is no longer conditionally independent of the variable Y_j given Y_i. We assessed performance of the CDAG, DAG and DAG2 estimators as the number E of such misspecified edges is increased (Fig. 3). We find that whilst CDAG continues to perform well up to a moderate fraction of misspecified edges, for larger fractions performance degrades and eventually coincides with DAG and DAG2. Secondly, we fixed $\theta = 0$, $p = 15$, $n = 1000$ and reduced the dependence of the Y_i on the X_i from 100% (relative to values used in the above simulations)

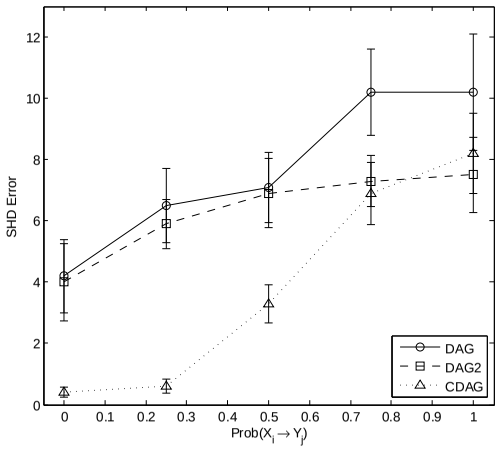


Figure 3: Simulated data results; model misspecification. [Data were generated using linear-Gaussian structural equations. Here we fixed $\theta = 0$, $p = 15$, $n = 1000$ and considered varying the number E of misspecified edges as described in the main text. On the x-axis we display the marginal probability that any given edge $Y_i \to Y_j$ has an associated misspecified edge $X_i \to Y_j$, so that when $\text{Prob}(X_i \to Y_j) = 1$ the number E of misspecified edges is equal to the number of edges in G. "DAG" = estimation based only on primary variables $(Y_i)_{i \in V}$, "DAG2" = estimation based on the full data $(X_i)_{i \in W} \cup (Y_i)_{i \in V}$, "CDAG" = CDAG estimation based on the full data $(X_i)_{i \in W} \cup (Y_i)_{i \in V}$.]

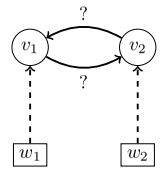
down to 0%. At 0% the partial faithfulness assumption is violated and estimators are no longer consistent. Results (SFig. 1) show that CDAG remains effective over a wide range of data-generating regimes, but eventually degrades when faithfulness is strictly violated. Thirdly, we considered removing the $X_i \to Y_i$ dependence from a random subset of the indices $i \in \{1, \ldots, p\}$, so that faithfulness is violated in a more heterogeneous way across the CDAG, similar to the situation considered by Neto *et al.* (2010). Results (SFig. 2) showed CDAG remained effective when only a handful of deletions occur, but eventually degraded as all the dependencies on secondary variables were removed.

3.2 Molecular Biological Data

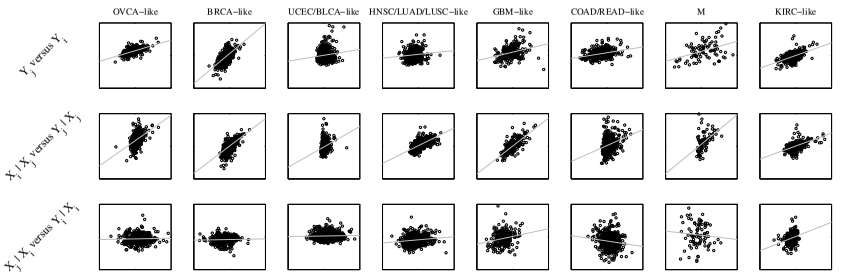
In this section we illustrate the use of CDAGs in an analysis of molecular data from cancer samples. We focus on causal links between post-translationally modified proteins involved in a process called cell signalling. The estimation of signalling networks has been a prominent topic in computational biology for some years (see, among others, Sachs *et al.*, 2005;

protein	node	variable
CHK1	w_i	X_i
p-CHK1	v_i	Y_i
CHK2	w_j	X_j
p-CHK2	v_j	Y_j

(a)



(b)



(c)

Figure 4: CHK1 total protein (t-CHK1) as a natural experiment for phosphorylation of CHK2 (p-CHK2). (a) Description of the variables. (b) A portion of the CDAG relating to these variables. It is desired to estimate whether there is a causal relationship $Y_i \to Y_j$ (possibly mediated by other protein species) or *vice versa*. (c) Top row: Plotting phosphorylated CHK1 (p-CHK1; Y_i) against p-CHK2 (Y_j) we observe weak correlation in some of the cancer subtypes. Middle row: We plot the residuals when t-CHK1 is regressed on total CHK2 (t-CHK2; x-axis) against the residuals when p-CHK2 is regressed on t-CHK2 (y-axis). The plots show a strong (partial) correlation in each subtype that suggests a causal effect in the direction p-CHK1 $\to$ p-CHK2. Bottom row: Reproducing the above but with the roles of CHK1 and CHK2 reversed, we see much reduced and in many cases negligible partial correlation, suggesting lack of a causal effect in the reverse direction, i.e. p-CHK1 $\not\leftarrow$ p-CHK2. [The grey line in each panel is a least-squares linear regression.]

Nelander *et al.*, 2008; Hill *et al.*, 2012; Oates and Mukherjee, 2012). Aberrations to causal signalling networks are central to cancer biology (Weinberg, 2013).

In this application, the primary variables $(Y_i)_{i \in V}$ represent abundance of phosphorylated protein (p-protein) while the secondary variables $(X_i)_{i \in W}$ represent abundance of corresponding total proteins (t-protein). A t-protein can be modified by a process called phosphorylation to form the corresponding p-protein and the p-proteins play a key role in signalling. An edge $v_i \to v_j$ has the biochemical interpretation that the phosphorylated form of protein i acts as a causal influence on phosphorylation of protein j. The DAG2

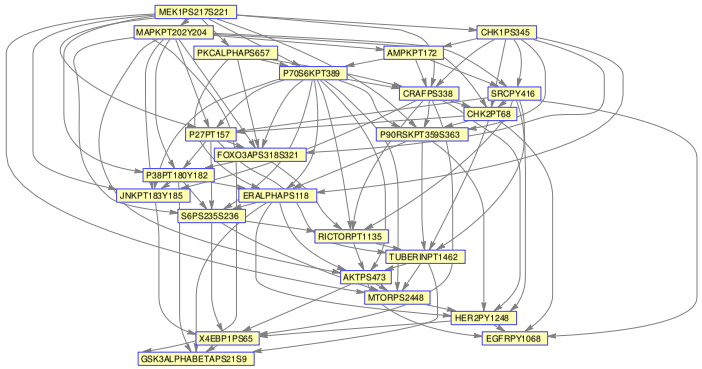


Figure 5: *Maximum a posteriori* conditional DAG, estimated from protein data from cancer samples (from the Cancer Genome Atlas, samples belonging to the BRCA-like group, see text). Here vertices represent phosphorylated proteins (primary variables) and edges have the biochemical interpretation that the parent protein plays a causal role in phosphorylation of the child protein.

method described in Sec. 3.1 may not be suitable for use here since the t-proteins are likely confounded by unobserved variables.

The data we analyse are from the TCGA "pan-cancer" project (Akbani *et al.*, 2014) and comprise measurements of protein levels (including both t- and p-proteins) using a technology called reverse phase protein arrays (RPPAs). We focus on $p = 24$ proteins for which (total, phosphorylated) pairs are available; the data span eight different cancer types (as defined by a clustering analysis due to Städler *et al.*, 2015) with a total sample size of $n = 3,467$ patients. We first illustrate the key idea of using secondary variables to inform causal inference regarding primary variables with an example from these data:

Example 1 (CHK1 t-protein as a natural experiment for CHK2 phosphorylation) *Consider RVs (Y_i, Y_j) corresponding respectively to p-CHK1 and p-CHK2, the phosphorylated forms of CHK1 and CHK2 proteins. Fig. 4c (top row) shows that these variables are weakly correlated in most of the 8 cancer subtypes. There is a strong partial correlation between t-CHK1 (X_i) and p-CHK2 (Y_j) in each of the subtypes when conditioning on t-CHK2 (X_j) (middle row), but there is essentially no partial correlation between t-CHK2 (X_j) and p-CHK1 (Y_i) in the same subtypes when conditioning on t-CHK1 (bottom row). Thus, under the CDAG assumptions, this suggests that there exists a directed causal path from p-CHK1 to p-CHK2, but not vice versa.*

Example 1 provides an example from the TCGA data where controlling for a secondary variable (here, t-protein abundance) may be important for causal inference concerning primary variables (p-protein abundance).

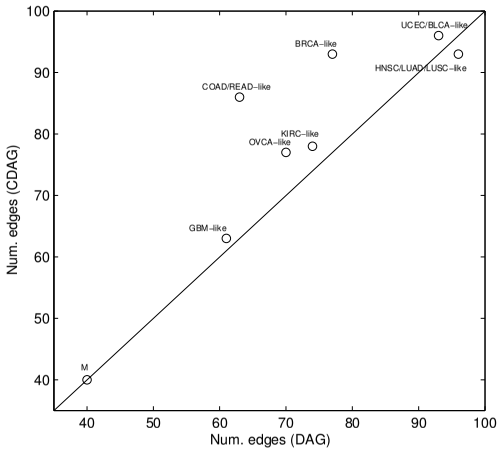


Figure 6: Cancer patient data; relative density of estimated protein networks. Here we plot the number of edges in the networks inferred by estimators based on DAGs (x-axis) and based on CDAGs (y-axis). [Each point corresponds to a cancer subtype (see text). "DAG" = estimation based only on primary variables $(Y_i)_{i \in V}$, "CDAG" = estimation based on the full data and enforcing CDAG structure.]

We now apply the CDAG methodology to all $p = 24$ primary variables that we consider, using data from the largest subtype in the study (namely BRCA-like). The estimated graph is shown in Fig. 5. We note that assessment of the biological validity of this causal graph is a nontrivial matter, and outside the scope of the present paper. However, we observe that several well known edges, such as from p-MEK to p-MAPK, appear in the estimated graph and are oriented in the expected direction. Interestingly, in several of these cases, the edge orientation is different when a standard DAG estimator is applied to the same data, showing that the CDAG formulation can reverse edge orientation with respect to a classical DAG (see supplement). We note also that the CDAG is denser, with more edges, than the DAG (Fig. 6), demonstrating that in many cases, accounting for secondary variables can render candidate edges more salient. These differences support our theoretical results insofar as they demonstrate that in practice CDAG estimation can give quite different results from a DAG analysis of the same primary variables but we note that proper assessment of estimated causal structure in this setting is beyond the scope of this paper.

4. Conclusions

Practitioners of causal inference understand that it is important to distinguish between variables of direct interest and others that can play a supporting role in analysis. In this

work we put forward CDAGs as a simple class of graphical models that make this distinction explicit. Motivated by molecular biological applications, we developed CDAGs that use bijections between primary and secondary index sets. The general approach presented here could be extended to other multivariate settings where variables are in some sense non-exchangeable. Naturally many of the philosophical considerations and practical limitations and caveats of classical DAGs remain relevant for CDAGs and we refer the reader to Dawid (2010) for an illuminating discussion of these issues.

The application to biological data presented above represents a principled approach to integrate different molecular data types (here, total and phosphorylated protein, but the ideas are general) for causal inference. Our results suggest that integration in a causal framework may be useful in some settings. Theoretical and empirical results showed that CDAGs can improve estimation of causal structure relative to classical DAGs when the CDAG assumptions are even approximately satisfied.

We briefly mention three natural extensions of the present work: (i) The CDAGs put forward here allow exactly one secondary variable X_i for each primary variable Y_i. In many settings this may be overly restrictive. In biology there are many examples of known causal relationships that are one-to-many or many-to-one. It would be natural to extend CDAGs in this direction. Examples of a more general formulation along these lines were recently discussed by Neto *et al.* (2010) in the context of eQTL data. Conversely we could extend the recent ideas of Kang *et al.* (2014) by allowing for multiple secondary variables for each primary variable, not all of which may be valid as instruments. (ii) In many applications data may be available from multiple related but causally non-identical groups, for example disease types. It could then be useful to consider *joint* estimation of multiple CDAGs, following recent work on estimation for multiple DAGs (Oates *et al.*, 2015, 2014). (iii) Biotechnological advances now mean that the number p of variables is frequently very large. Estimation for high-dimensional CDAGs may be possible using recent results for high-dimensional DAGs (e.g. Kalisch and Bühlmann, 2007; Loh and Bühlmann, 2014, and others).

Acknowledgments

This work was inspired by discussions with Jonas Peters at the Workshop *Statistics for Complex Networks*, Eindhoven 2013, and Vanessa Didelez at the *UK Causal Inference Meeting*, Cambridge 2014. CJO was supported by the Centre for Research in Statistical Methodology (CRiSM) EPSRC EP/D002060/1 and the ARC Centre of Excellence for Mathematical and Statistical Frontiers. Part of this work was done while SM was at the Medical Research Council Biostatistics Unit and University of Cambridge, Cambridge, UK. SM acknowledges the support of the Medical Research Council and a Wolfson Research Merit Award from the UK Royal Society.

References

Achterberg, T. (2009). SCIP: solving constraint integer programs. *Mathematical Programming Computation* **1**(1), 1-41.

Akbani, R. *et al.* (2014). A pan-cancer proteomic perspective on The Cancer Genome Atlas. *Nature Communications* 5:3887.

Bartlett, M., and Cussens, J. (2013). Advances in Bayesian network learning using integer programming. *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, 182-191.

Bayarri, M. J., Berger, J. O., Forte, A., and García-Donato, G. (2012). Criteria for Bayesian model choice with application to variable selection. *The Annals of Statistics* **40**(3), 1550-1577.

Cai, T. T., Li, H., Liu, W., and Xie, J. (2013). Covariate-adjusted precision matrix estimation with an application in genetical genomics. *Biometrika* **100**(1), 139-156.

Cai, X., Bazerque, J. A., and Giannakis, G. B. (2013). Inference of gene regulatory networks with sparse structural equation models exploiting genetic perturbations. *PLoS Computational Biology* **9**(5), e1003068.

Chickering, D.M. (2003). Optimal structure identification with greedy search. *The Journal of Machine Learning Research* **3**, 507-554.

Chun, H., Chen, M., Li, B., and Zhao, H. (2013). Joint conditional Gaussian graphical models with multiple sources of genomic data. *Frontiers in Genetics* **4**, 294.

Consonni, G., and La Rocca, L. (2010). Moment priors for Bayesian model choice with applications to directed acyclic graphs. *Bayesian Statistics* **9**(9), 119-144.

Cowell, R.G. (2009). Efficient maximum likelihood pedigree reconstruction. *Theoretical Population Biology* **76**, 285-291.

Cussens, J. (2011). Bayesian network learning with cutting planes. *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 153-160.

Dawid, A.P. (2001). Separoids: A mathematical framework for conditional independence and irrelevance. *The Annals of Mathematics and Artificial Intelligence* **32**(1-4), 335-372.

Dawid, A.P. (2002). Influence diagrams for causal modelling and inference. *International Statistical Review* **70**(2), 161-189.

Dawid, A.P. (2010). Beware of the DAG! *Journal of Machine Learning Research-Proceedings Track* **6**, 59-86.

Deltell, A.F. (2011). Objective Bayes criteria for variable selection. *Doctoral dissertation, Universitat de València.*

Didelez, V., and Sheehan, N. (2007). Mendelian randomization as an instrumental variable approach to causal inference. *Statistical Methods in Medical Research* **16**, 309-330.

Evans, R. J., and Richardson, T. S. (2014). Markovian acyclic directed mixed graphs for discrete data. *The Annals of Statistics* **42**(4), 1452-1482.

Evans, R. J. (2015). Margins of discrete Bayesian networks. arXiv:1501.02103.

Fernández, C., Ley, E., and Steel, M. F. (2001). Benchmark priors for Bayesian model averaging. *Journal of Econometrics* **100**(2), 381-427.

Friedman, N., and Koller, D. (2003). Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning* **50**(1-2):95-126.

Greenland, S. (2000). An introduction to instrumental variables for epidemiologists. *International Journal of Epidemiology* **29**(4), 722-729.

Hill, S. M., Lu, Y., Molina, J., Heiser, L. M., Spellman, P. T., Speed, T. P., Gray, J. W., Mills, G. B., and Mukherjee, S. (2012). Bayesian inference of signaling network topology in a cancer cell line. *Bioinformatics* **28**(21), 2804-2810.

Jaakkola, T., Sontag, D., Globerson, A., and Meila, M. (2010). Learning Bayesian network structure using LP relaxations. *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 358-365.

Jeffreys, H. (1961). *Theory of probability*. Oxford University Press, 3rd edition.

Johnson V.E., and Rossell D. (2010). Non-local prior densities for default Bayesian hypothesis tests. *Journal of the Royal Statistical Society B* **72**, 143-170.

Kalisch, M., and Bühlmann, P. (2007). Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research* **8**, 613-636.

Kang, H., Zhang, A., Cai, T. T., and Small, D. S. (2014). Instrumental variables estimation with some invalid instruments and its application to Mendelian randomization. arXiv:1401.5755.

Lauritzen, S.L. (2000). Causal inference from graphical models. In: *Complex Stochastic Systems, Eds. O.E. Barndorff-Nielsen, D.R. Cox and C. Klüppelberg.*CRC Press, London.

Lauritzen, S.L., and Richardson, T.S. (2002). Chain graph models and their causal interpretations. *Journal of the Royal Statistical Society: Series B* **64**(3):321-348.

Li, B., Chun, H., and Zhao, H. (2012). Sparse estimation of conditional graphical models with application to gene networks. *Journal of the American Statistical Association* **107**(497), 152-167.

Logsdon, B. A., and Mezey, J. (2010). Gene expression network reconstruction by convex feature selection when incorporating genetic perturbations. *PLoS Computational Biology* **6**(12), e1001014.

Loh, P. L., and Bühlmann, P. (2014). High-dimensional learning of linear causal networks via inverse covariance estimation. *Journal of Machine Learning Research* **15**, 3065-3105.

Massam, H., and Wesołowski, J. (2014). A new prior for the discrete DAG models with a restricted set of directions. arXiv:1412.0972.

Nelander, S., Wang, W., Nilsson, B., She, Q. B., Pratilas, C., Rosen, N., Gennemark, P., and Sander, C. (2008). Models from experiments: combinatorial drug perturbations of cancer cells. *Molecular Systems Biology* **4**, 216.

Nemhauser, G.L., and Wolsey, L.A. (1988). *Integer and combinatorial optimization*. Wiley, New York.

Neto, E. C., Keller, M. P., Attie, A. D., and Yandell, B. S. (2010). Causal graphical models in systems genetics: a unified framework for joint inference of causal network and genetic architecture for correlated phenotypes. *The Annals of Applied Statistics* **4**(1), 320-339.

Oates, C. J., and Mukherjee, S. (2012). Network inference and biological dynamics. *The Annals of Applied Statistics* **6**(3), 1209-1235.

Oates, C. J., Costa, L., and Nichols, T.E. (2014). Towards a multi-subject analysis of neural connectivity. *Neural Computation* **27**, 1-20.

Oates, C. J., Smith, J. Q., Mukherjee, S., and Cussens, J. (2015). Exact estimation of multiple directed acyclic graphs. *Statistics and Computing*, to appear.

Pearl, J., and Paz, A. (1985). Graphoids: A graph-based logic for reasoning about relevance relations. Computer Science Department, University of California.

Pearl, J., and Verma, T. (1987). The logic of representing dependencies by directed graphs. In: *Proceedings of the AAAI*, Seattle WA, 374-379.

Pearl, J. (2003). Reply to Woodward. *Economics and Philosophy* **19**(2), 341-344.

Pearl, J. (2009). *Causality: models, reasoning and inference (2nd ed.)*. Cambridge University Press.

Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D. A., and Nolan, G. P. (2005). Causal protein-signaling networks derived from multiparameter single-cell data. *Science* **308**(5721), 523-529.

Scott, J. G., and Berger, J. O. (2010). Bayes and empirical-Bayes multiplicity adjustment in the variable-selection problem. *The Annals of Statistics* **38**(5), 2587-2619.

Silander, T., and Myllymäkki, P. (2006). A simple approach to finding the globally optimal Bayesian network structure. *Proceedings of the 22nd Conference on Artificial Intelligence*, 445-452.

Spirtes, P., Glymour, C., and Scheines, R. (2000). *Causation, prediction, and search (Second ed.)*. MIT Press.

Städler, N., Dondelinger, F., Hill, S. M., Kwok, P., Ng, S., Akbani, R., Werner, H. M. J., Shahmoradgoli, M., Lu, Y., Mills, G. B., and Mukherjee, S. (2015). In preparation.

Studený, M. (2005). Probabilistic conditional independence structures. London: Springer.

Tsamardinos, I., Brown, L.E., and Aliferis, C.F. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning* **65**(1), 31-78.

Uhler, C., Raskutti, G., Bhlmann, P., and Yu, B. (2013). Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics* **41**(2), 436-463.

van Wieringen, W. N., and van de Wiel, M. A. (2014). Penalized differential pathway analysis of integrative oncogenomics studies. *Statistical Applications in Genetics and Molecular Biology* **13**(2), 141-158.

Weinberg, R. (2013). *The biology of cancer*. Garland Science.

Wolsey, L.A. (1998). *Integer programming*. Wiley, New York.

Yin, J., and Li, H. (2011). A sparse conditional Gaussian graphical model for analysis of genetical genomics data. *The Annals of Applied Statistics* **5**(4), 2630-2650.

Yuan, C., and Malone, B. (2013). Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research* **48**, 23-65.

Zellner, A. (1986). On assessing prior distributions and Bayesian regression analysis with g-prior distributions. *In A. Zellner, ed., Bayesian Inference and Decision techniques: Essays in Honour of Bruno de Finetti, Edward Elgar Publishing Limited*, 389-399.

Zhang, L., and Kim, S. (2014). Learning gene networks under SNP perturbations using eQTL datasets. *PLoS Computational Biology* **10**(2), e1003420.

Journal of Machine Learning Research 17 (2016) 1-46 Submitted 1/15; Revised 12/15; Published 4/16

Adaptive Lasso and group-Lasso for functional Poisson regression

Stéphane Ivanoff IVANOFF@CEREMADE.DAUPHINE.FR
CEREMADE UMR CNRS 7534
Université Paris Dauphine
F-75775 Paris, France

Franck Picard FRANCK.PICARD@UNIV-LYON1.FR
Laboratoire de Biométrie et Biologie Évolutive,
UMR CNRS 5558 Univ. Lyon 1
F-69622 Villeurbanne, France

Vincent Rivoirard RIVOIRARD@CEREMADE.DAUPHINE.FR
CEREMADE UMR CNRS 7534
Université Paris Dauphine
F-75775 Paris, France

Editor: Zhihua Zhang

Abstract

High dimensional Poisson regression has become a standard framework for the analysis of massive counts datasets. In this work we estimate the intensity function of the Poisson regression model by using a dictionary approach, which generalizes the classical basis approach, combined with a Lasso or a group-Lasso procedure. Selection depends on penalty weights that need to be calibrated. Standard methodologies developed in the Gaussian framework can not be directly applied to Poisson models due to heteroscedasticity. Here we provide data-driven weights for the Lasso and the group-Lasso derived from concentration inequalities adapted to the Poisson case. We show that the associated Lasso and group-Lasso procedures satisfy fast and slow oracle inequalities. Simulations are used to assess the empirical performance of our procedure, and an original application to the analysis of Next Generation Sequencing data is provided.

Keywords: Functional Poisson regression, adaptive lasso, adaptive group-lasso, calibration, concentration

Introduction

Poisson functional regression has become a standard framework for image or spectra analysis, in which case observations are made of n independent couples $(Y_i, X_i)_{i=1,\dots,n}$, and can be modeled as

$$Y_i|X_i \sim \mathcal{P}oisson(f_0(X_i)).$$

The X_i's (random or fixed) are supposed to lie in a known compact support of $\mathbb{R}^d$ ($d \geq 1$), say $[0,1]^d$, and the purpose is to estimate the unknown intensity function f_0 assumed to be positive. Wavelets have been used extensively for intensity estimation, and the statistical challenge has been to propose thresholding procedures in the spirit of Donoho and Johnstone (1994), that were adapted to the variance's spatial variability associated with the Poisson framework. An early method to deal with high dimensional count data has been to apply a variance stabilizing-transform (see Anscombe (1948)) and to treat the transformed data as if they were Gaussian. More recently, the same idea has been applied to the data's decomposition in the Haar-wavelet basis, see Fryzlewicz and Nason (2004) and Fryzlewicz (2008), but these methods rely on asymptotic approximations and tend to show lower performance when the level of counts is low (see Besbeas et al. (2004)). Dedicated wavelet thresholding methods were developed in the Poisson setting by Kolaczyk (1999) and Sardy et al. (2004), and a recurrent challenge has been to define an appropriate threshold like the universal threshold for shrinkage and selection, as the heteroscedasticity of the model calls for component-wise thresholding.

In this work we first propose to enrich the standard wavelet approach by considering the so-called dictionary strategy. We assume that $\log f_0$ can be well approximated by a linear combination of p known functions, and we reduce the estimation of f_0 to the estimation of p coefficients. Dictionaries can be built from classical orthonormal systems such as wavelets, histograms or the Fourier basis, which results in a framework that encompasses wavelet methods. Considering overcomplete (*ie* redundant) dictionaries is efficient to capture different features in the signal, by using sparse representations (see Chen et al. (2001) or Tropp (2004)). For example, if $\log f_0$ shows piece-wise constant trends along with some periodicity, combining both Haar and Fourier bases will be more powerful than separate strategies, and the model will be sparse in the coefficients domain. To ensure sparse estimations, we consider the Lasso and the group-Lasso procedures. Group estimators are particularly well adapted to the dictionary framework, especially if we consider dictionaries based on a wavelet system, for which it is well known that coefficients can be grouped scale-wise for instance (see Chicken and Cai (2005)). Finally, even if we do not make any assumption on p itself, it may be larger than n and methodologies based on ℓ_1-penalties, such as the Lasso and the group-Lasso appear appropriate.

The statistical properties of the Lasso are particularly well understood in the context of regression with *i.i.d.* errors, or for density estimation for which a range of oracle inequalities have been established. These inequalities, now widespread in the literature, provide theoretical error bounds that hold on events with a controllable (large) probability. See for instance Bertin et al. (2011), Bickel et al. (2009), Bunea et al. (2007a,b) and the references therein. For generalized linear models, Park and Hastie (2007) studied ℓ_1-regularization path algorithms and van de Geer (2008) established non-asymptotic oracle inequalities. The sign consistency of the Lasso has been studied by Jia et al. (2013) for a very specific Poisson model. Finally, we also mention than the Lasso has also been extensively consid-

ered in survival analysis. See for instance Gaïffas and Guilloux (2012), Zou (2008), Kong and Nan (2014), Bradic et al. (2011), Lemler (2013) and Hansen et al. (2014).

Here we consider not only the Lasso estimator but also its extension, the group-Lasso proposed by Yuan and Lin (2006), which is relevant when the set of parameters can be partitioned into groups. The analysis of the group-Lasso has been led in different contexts. For instance, consistency has been studied by Bach (2008), Obozinski et al. (2011) and Wei and Huang (2010). In the linear model, Nardi and Rinaldo (2008) derived conditions ensuring various asymptotic properties such as consistency, oracle properties or persistence. Still for the linear model, Lounici et al. (2011) established oracle inequalities and, in the Gaussian setting, pointed out advantages of the group-Lasso with respect to the Lasso, generalizing the results of Chesneau and Hebiri (2008) and Huang and Zhang (2010). We also mention Meier et al. (2008) who studied the group-Lasso for logistic regression, Blazere et al. (2014) for generalized linear model with Poisson regression as a special case and Dalalyan et al. (2013) for other linear heteroscedastic models.

As pointed out by empirical comparative studies (see Besbeas et al. (2004)), the calibration of any thresholding rule is of central importance. Here we consider Lasso and group-Lasso penalties of the form

$$\mathrm{pen}(\boldsymbol{\beta}) = \sum_{j=1}^{p} \lambda_j |\beta_j|$$

and

$$\mathrm{pen}^g(\boldsymbol{\beta}) = \sum_{k=1}^{K} \lambda_k^g \|\boldsymbol{\beta}_{G_k}\|_2,$$

where $G_1 \cup \cdots \cup G_K$ is a partition of $\{1, \ldots, p\}$ into non-overlapping groups (see Section 1 for more details). By calibration we refer to the definition and to the suitable choice of the weights λ_j and λ_k^g, which is intricate in heteroscedastic models, especially for the group-Lasso. For functional Poissonian regression, the ideal shape of these weights is unknown, even if for the group-Lasso, the λ_k^g's should of course depend on the groups size. As for the Lasso, most proposed weights in the literature are non-random and constant such that the penalty is proportional to $\|\boldsymbol{\beta}\|_1$, but when facing variable selection and consistency simultaneously, Zou (2006) showed the interest in considering non-constant data-driven ℓ_1-weights even in the simple case where the noise is Gaussian with constant variance. This issue becomes even more critical in Poisson functional regression in which variance shows spatial heterogeneity. As Zou (2006), our first contribution is to propose here adaptive procedures with weights depending on the data. Weights λ_j for the Lasso are derived by using sharp concentration inequalities, in the same spirit as Bertin et al. (2011), Gaïffas and Guilloux (2012), Lemler (2013) and Hansen et al. (2014), but adapted to the Poissonian setting. To account for heteroscedasticity, weights λ_j are component-specific and depend

on the data (see Theorem 1). We propose a similar procedure for the calibration of the group-Lasso. In most proposed procedures, the analogs of the λ_k^g's are proportional to the $\sqrt{|G_k|}$'s (see Nardi and Rinaldo (2008), Bühlmann and van de Geer (2011) or Blazere et al. (2014)). But to the best of our knowledge, adaptive group-Lasso procedures (with weights depending on the data) have not been proposed yet. This is the purpose of Theorem 2, which is the main result of this work, generalizing Theorem 1 by using sharp concentration inequalities for infinitely divisible vectors. We show the shape relevance of the data-driven weights λ_k^g by comparing them to the weights proposed by Lounici et al. (2011) in the Gaussian framework. In Theorem 2, we do not impose any condition on the groups size. However, whether $|G_k|$ is smaller than $\log p$ or not highly influences the order of magnitude of λ_k^g.

Our second contribution consists in providing the theoretical validity of our approach by establishing slow and fast oracle inequalities under RE-type conditions in the same spirit as Bickel et al. (2009). Closeness between our estimates and f_0 is measured by using the empirical Kullback-Leibler divergence. We show that classical oracle bounds are achieved. We also show the relevance of considering the group-Lasso instead of the Lasso in some situations. Our results, that are non-asymptotic, are valid under very general conditions on the design $(X_i)_{i=1,\ldots,n}$ and on the dictionary. However, to shed some light on our results, we illustrate some of them in the asymptotic setting with classical dictionaries like wavelets, histograms or Fourier bases. Our approach generalizes the classical basis approach and in particular block wavelet thresholding which is equivalent to group-Lasso in that case (see Yuan and Lin (2006)). We refer the reader to Chicken and Cai (2005) for a deep study of block wavelet thresholding in the context of density estimation whose framework shows some similarities with ours in terms of heteroscedasticity. Note that sharp estimation of variance terms proposed in this work can be viewed as an extension of coarse bounds provided by Chicken and Cai (2005). Finally, we emphasize that our procedure differs from Blazere et al. (2014)'s one in several aspects: First, in their Poisson regression setting, they do not consider a dictionary approach. Furthermore, their weights are constant and not data-driven, so are strongly different from ours. Finally, rates of Blazere et al. (2014) are established under much stronger assumptions than ours (see Section 3.1 for more details).

Finally, we explore the empirical properties of our calibration procedures by using simulations. We show that our procedures are very easy to implement, and we compare their performance with variance-stabilizing transforms and cross-validation. The calibrated Lasso and group-Lasso are associated with excellent reconstruction properties, even in the case of low counts. We also propose an original application of functional Poisson regression to the analysis of Next Generation Sequencing data, with the denoising of a Poisson intensity applied to the detection of replication origins in the human genome (see Picard et al. (2014)).

This article is organized as follows. In Section 1, we introduce the Lasso and group-Lasso procedures we propose in the dictionary approach setting. In Section 2, we derive

data-driven weights of our procedures that are extensively commented. Theoretical performance of our estimates are studied in Section 3 in the oracle approach. In Section 4, we investigate the empirical performance of the proposed estimators using simulated data, and an application is provided on next generation sequencing data in Section 5.

1. Penalized log-likelihood estimates for Poisson regression and dictionary approach

We consider the functional Poisson regression model, with n observed counts $Y_i \in \mathbb{N}$ modeled such that:

$$Y_i|X_i \sim \mathcal{P}oisson(f_0(X_i)), \tag{1.1}$$

with the X_i's (random or fixed) supposed to lie in a known compact support, say $[0,1]^d$. Since the goal here is to estimate the function f_0 assumed to be positive on $[0,1]^d$, a natural candidate is a function f of the form $f = \exp(g)$. Then, we consider the so-called dictionary approach which consists in decomposing g as a linear combination of the elements of a given finite dictionary of functions denoted by $\Upsilon = \{\varphi_j\}_{j\in\mathcal{J}}$, with $\|\varphi_j\|_2 = 1$ for all j. Consequently, we choose g of the form:

$$g = \sum_{j\in\mathcal{J}} \beta_j \varphi_j,$$

with $p = \text{card}(\mathcal{J})$ that may depend on n (as well as the elements of Υ). Without loss of generality we will assume in the following that $\mathcal{J} = \{1,\dots,p\}$. In this framework, estimating f_0 is equivalent to estimating the vector of regression coefficients $\boldsymbol{\beta} = (\beta_j)_{j\in\mathcal{J}} \in \mathbb{R}^p$. In the sequel, we write $g_{\boldsymbol{\beta}} = \sum_{j\in\mathcal{J}} \beta_j\varphi_j$, $f_{\boldsymbol{\beta}} = \exp(g_{\boldsymbol{\beta}})$, for all $\boldsymbol{\beta} \in \mathbb{R}^p$. Note that we do not require the model to be true, that is we do not suppose the existence of $\boldsymbol{\beta}_0$ such that $f_0 = f_{\boldsymbol{\beta}_0}$.

The strength of the dictionary approach lies in its ability to capture different features of the function to estimate (smoothness, sparsity, periodicity,...) by sparse combinations of elements of the dictionary so that only few coefficients need to be selected, which limits its estimation errors. Obviously, the dictionary approach encompasses the classical basis approach consisting in decomposing g on an orthonormal system. The richer the dictionary, the sparser the decomposition, so p can be larger than n and the model becomes high-dimensional.

We consider a likelihood-based penalized criterion to select $\boldsymbol{\beta}$, the coefficients of the dictionary decomposition. We denote by $\mathbf{A}$ the $n \times p$-design matrix with $A_{ij} = \varphi_j(X_i)$, $\mathbf{Y} = (Y_1,\dots,Y_n)^T$ and the log-likelihood associated with this model is

$$l(\boldsymbol{\beta}) = \sum_{j\in\mathcal{J}} \beta_j (\mathbf{A}^T\mathbf{Y})_j - \sum_{i=1}^n \exp\Big(\sum_{j\in\mathcal{J}} \beta_j A_{ij}\Big) - \sum_{i=1}^n \log(Y_i!),$$

which is a concave function of $\boldsymbol{\beta}$. Next sections propose two different ways to penalize $-l(\boldsymbol{\beta})$.

1.1 The Lasso estimate

The first penalty we propose is based on the (weighted) ℓ_1-norm and we obtain a Lasso-type estimate by considering

$$\widehat{\boldsymbol{\beta}}^L \in \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\text{argmin}} \left\{ -l(\boldsymbol{\beta}) + \sum_{j=1}^p \lambda_j|\beta_j| \right\}. \tag{1.2}$$

The penalty term $\sum_{j=1}^p \lambda_j|\beta_j|$ depends on positive weights $(\lambda_j)_{j\in\mathcal{J}}$ that vary according to the elements of the dictionary and are chosen in Section 2.1. This choice of varying weights instead of a unique λ stems from heteroscedasticity due to the Poisson regression, and a first part of our work consists in providing theoretical data-driven values for these weights, in the same spirit as Bertin et al. (2011) or Hansen et al. (2014) for instance. From the first order optimality conditions (see Bühlmann and van de Geer (2011)), $\widehat{\boldsymbol{\beta}}^L$ satisfies

$$\begin{cases} \mathbf{A}_j^T(\mathbf{Y} - \exp(\mathbf{A}\widehat{\boldsymbol{\beta}}^L)) = \lambda_j \dfrac{\widehat{\beta}_j^L}{|\widehat{\beta}_j^L|} & \text{if } \widehat{\beta}_j^L \neq 0, \\ |\mathbf{A}_j^T(\mathbf{Y} - \exp(\mathbf{A}\widehat{\boldsymbol{\beta}}^L))| \leq \lambda_j & \text{if } \widehat{\beta}_j^L = 0, \end{cases}$$

where $\exp(\mathbf{A}\boldsymbol{\beta}) = (\exp((\mathbf{A}\boldsymbol{\beta})_1),\dots,\exp((\mathbf{A}\boldsymbol{\beta})_n))^T$ and $\mathbf{A}_j$ is the j-th column of the matrix $\mathbf{A}$. Note that the larger the λ_j's, the sparser the estimates. In particular $\widehat{\boldsymbol{\beta}}^L$ belongs to the set of the vectors $\boldsymbol{\beta} \in \mathbb{R}^p$ that satisfies for any $j \in \mathcal{J}$,

$$|\mathbf{A}_j^T(\mathbf{Y} - \exp(\mathbf{A}\boldsymbol{\beta}))| \leq \lambda_j. \tag{1.3}$$

The Lasso estimator of f_0 is now easily derived.

Definition 1 *The Lasso estimator of f_0 is defined as*

$$\widehat{f}^L(x) := \exp(\widehat{g}^L(x)) := \exp\left(\sum_{j=1}^p \widehat{\beta}_j^L \varphi_j(x)\right).$$

We also propose an alternative to $\widehat{f}^L$ by considering the group-Lasso.

1.2 The group-Lasso estimate

We also consider the grouping of coefficients into non-overlapping blocks. Indeed, group estimates may be better adapted than their single counterparts when there is a natural group structure. The procedure keeps or discards all the coefficients within a block and can increase estimation accuracy by using information about coefficients of the same block. In our setting, we partition the set of indices $\mathcal{J} = \{1,\dots,p\}$ into K non-empty groups:

$$\{1,\dots,p\} = G_1 \cup G_2 \cup \dots \cup G_K.$$

For any $\boldsymbol{\beta} \in \mathbb{R}^p$, $\boldsymbol{\beta}_{G_k}$ stands for the sub-vector of $\boldsymbol{\beta}$ with elements indexed by the elements of G_k, and we define the block ℓ_1-norm on $\mathbb{R}^p$ by

$$\|\boldsymbol{\beta}\|_{1,2} = \sum_{k=1}^{K} \|\boldsymbol{\beta}_{G_k}\|_2.$$

Similarly, $\mathbf{A}_{G_k}$ is the $n \times |G_k|$ submatrix of $\mathbf{A}$ whose columns are indexed by the elements of G_k. Then the group-Lasso $\widehat{\boldsymbol{\beta}}^{gL}$ is a solution to the following convex optimization problem:

$$\widehat{\boldsymbol{\beta}}^{gL} \in \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \Big\{ -l(\boldsymbol{\beta}) + \sum_{k=1}^{K} \lambda_k^g \|\boldsymbol{\beta}_{G_k}\|_2 \Big\},$$

where the λ_k^g's are positive weights for which we also provide a theoretical data-driven expression in Section 2.2. This group-estimator is constructed similarly to the Lasso, with the block ℓ_1-norm being used instead of the ℓ_1-norm. In particular, note that if all groups are of size one then we recover the Lasso estimator. Convex analysis states that $\widehat{\boldsymbol{\beta}}^{gL}$ is a solution of the above optimization problem if the p-dimensional vector $\mathbf{0}$ is in the subdifferential of the objective function. Therefore, $\widehat{\boldsymbol{\beta}}^{gL}$ satisfies:

$$\begin{cases} \mathbf{A}_{G_k}^T(\mathbf{Y} - \exp(\mathbf{A}\widehat{\boldsymbol{\beta}}^{gL})) = \lambda_k^g \dfrac{\widehat{\boldsymbol{\beta}}_{G_k}^{gL}}{\|\widehat{\boldsymbol{\beta}}_{G_k}^{gL}\|_2} & \text{if } \widehat{\boldsymbol{\beta}}_{G_k}^{gL} \neq \mathbf{0}, \\ \|\mathbf{A}_{G_k}^T(\mathbf{Y} - \exp(\mathbf{A}\widehat{\boldsymbol{\beta}}^{gL}))\|_2 \leq \lambda_k^g & \text{if } \widehat{\boldsymbol{\beta}}_{G_k}^{gL} = \mathbf{0}. \end{cases}$$

This procedure naturally enhances group-sparsity as analyzed by Yuan and Lin (2006), Lounici et al. (2011) and references therein.

Obviously, $\widehat{\boldsymbol{\beta}}^{gL}$ belongs to the set of the vectors $\boldsymbol{\beta} \in \mathbb{R}^p$ that satisfy for any $k \in \{1, \ldots, K\}$,

$$\|\mathbf{A}_{G_k}^T(\mathbf{Y} - \exp(\mathbf{A}\boldsymbol{\beta}))\|_2 \leq \lambda_k^g. \tag{1.4}$$

Now, we set

Definition 2 *The group Lasso estimator of f_0 is defined as*

$$\widehat{f}^{gL}(x) := \exp(\widehat{g}^{gL}(x)) := \exp\left(\sum_{j=1}^{p} \widehat{\beta}_j^{gL} \varphi_j(x)\right).$$

In the following our results are given conditionally on the X_i's, and $\mathbb{E}$ (resp. $\mathbb{P}$) stands for the expectation (resp. the probability measure) conditionally on $X_1, \ldots, X_n$. In some situations, to give orders of magnitudes of some expressions, we will use the following definition:

Definition 3 *We say that the design $(X_i)_{i=1,\ldots,n}$ is regular if either the design is deterministic and the X_i's are equispaced in $[0,1]$ or the design is random and the X_i's are i.i.d. with density h, with*

$$0 < \inf_{x\in[0,1]^d} h(x) \leq \sup_{x\in[0,1]^d} h(x) < \infty.$$

2. Weights calibration using concentration inequalities

Our first contribution is to derive theoretical data-driven values of the weights λ_j's and λ_k^g's, specially adapted to the Poisson model. In the classical Gaussian framework with noise variance σ^2, weights for the Lasso are chosen to be proportional to $\sigma\sqrt{\log p}$ (see Bickel et al. (2009) for instance). The Poisson setting is more involved due to heteroscedasticity and such simple tuning procedures cannot be generalized easily. Sections 2.1 and 2.2 give closed forms of parameters λ_j and λ_k^g. They are based on concentration inequalities specific to the Poisson model. In particular, λ_j is used to control the fluctuations of $\mathbf{A}_j^T\mathbf{Y}$ around its mean, which enhances the key role of V_j, a variance term (the analog of σ^2) defined by

$$V_j = \mathrm{Var}(\mathbf{A}_j^T\mathbf{Y}) = \sum_{i=1}^{n} f_0(X_i)\varphi_j^2(X_i). \tag{2.1}$$

2.1 Data-driven weights for the Lasso procedure

For any j, we choose a data-driven value for λ_j as small as possible so that with high probability, for any $j \in \mathcal{J}$,

$$|\mathbf{A}_j^T(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])| \leq \lambda_j. \tag{2.2}$$

Such a control is classical for Lasso estimates (see the references above) and is also a key point of the technical arguments of the proofs. Requiring that the weights are as small as possible is justified, from the theoretical point of view, by oracle bounds depending on the λ_j's (see Corollaries 1 and 2). Furthermore, as discussed in Bertin et al. (2011), choosing theoretical Lasso weights as small as possible is also a suitable guideline for practical purposes. Finally, note that if the model were true, *i.e.* if there existed a true sparse vector $\boldsymbol{\beta}_0$ such that $f_0 = f_{\boldsymbol{\beta}_0}$, then $\mathbb{E}[\mathbf{Y}] = \exp(\mathbf{A}\boldsymbol{\beta}_0)$ and $\boldsymbol{\beta}_0$ would belong to the set defined by (1.3) with large probability. The smaller the λ_j's, the smaller the set within selection of $\widehat{\boldsymbol{\beta}}^L$ is performed. So, with a sharp control in (2.2), we increase the probability to select $\boldsymbol{\beta}_0$. The following theorem provides the data-driven weights λ_j's. The main theoretical ingredient we use to choose the weights λ_j's is a concentration inequality for Poisson processes and to proceed, we link the quantity $\mathbf{A}_j^T\mathbf{Y}$ to a specific Poisson process, as detailed in the proofs Section 7.1.

Theorem 1 *Let j be fixed and $\gamma > 0$ be a constant. Define $\widehat{V}_j = \sum_{i=1}^n \varphi_j^2(X_i)Y_i$ the natural unbiased estimator of V_j and*

$$\widetilde{V}_j = \widehat{V}_j + \sqrt{2\gamma \log p \widehat{V}_j \max_i \varphi_j^2(X_i)} + 3\gamma \log p \max_i \varphi_j^2(X_i).$$

Set

$$\lambda_j = \sqrt{2\gamma \log p \widetilde{V}_j} + \frac{\gamma \log p}{3} \max_i |\varphi_j(X_i)|, \tag{2.3}$$

then

$$\mathbb{P}\Big(|\mathbf{A}_j^T(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])| \geq \lambda_j\Big) \leq \frac{3}{p^\gamma}. \tag{2.4}$$

The first term $\sqrt{2\gamma \log p \widetilde{V}_j}$ in λ_j is the main one, and constitutes a variance term depending on $\widetilde{V}_j$ that slightly overestimates V_j (see Section 7.1 for more details about the derivation of $\widetilde{V}_j$). Its dependence on an estimate of V_j was expected since we aim at controlling fluctuations of $\mathbf{A}_j^T\mathbf{Y}$ around its mean. The second term comes from the heavy tail of the Poisson distribution, and is the price to pay, in the non-asymptotic setting, for the added complexity of the Poisson framework compared to the Gaussian framework.

To shed more lights on the form of the proposed weights from the asymptotic point of view, assume that the design is regular (see Definition 3). In this case, it is easy to see that under mild assumptions on f_0, V_j is asymptotically of order n. If we further assume that

$$\max_i |\varphi_j(X_i)| = o(\sqrt{n/\log p}), \tag{2.5}$$

then, when p is large, with high probability, $\widehat{V}_j$ (and then $\widetilde{V}_j$) is also of order n (using Remark 2 in the proofs Section 7.1), and the second term in λ_j is negligible with respect to the first one. In this case, λ_j is of order $\sqrt{n \log p}$. Note that Assumption (2.5) is quite classical in heteroscedastic settings (see Bertin et al. (2011)). By taking the hyperparameter γ larger than 1, then for large values of p, (2.2) is true for any $j \in \mathcal{J}$, with large probability.

2.2 Data-driven weights for the group Lasso procedure

Current group-Lasso procedures are tuned by choosing the analog of λ_k^g proportional to $\sqrt{|G_k|}$ (see Nardi and Rinaldo (2008), Chapter 4 of Bühlmann and van de Geer (2011) or Blazere et al. (2014)). A more refined version of tuning group-Lasso is provided by Lounici et al. (2011) in the Gaussian setting (see below for a detailed discussion). To the best of our knowledge, data-driven weights (with theoretical validation) for the group-Lasso have not been proposed yet. It is the purpose of Theorem 2. Similarly to the previous section, we propose data-driven theoretical derivations for the weights λ_k^g's that are chosen as small as possible, but satisfying for any $k \in \{1, \dots, K\}$,

$$\|\mathbf{A}_{G_k}^T(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])\|_2 \leq \lambda_k^g \tag{2.6}$$

with high probability (see (1.4)). Choosing the smallest possible weights is also recommended by Lounici et al. (2011) in the Gaussian setting (see in their Section 3 the discussion about weights and comparisons with coarser weights of Nardi and Rinaldo (2008)). Obviously, λ_k^g should depend on sharp estimates of the variance parameters $(V_j)_{j \in G_k}$. The following theorem is the equivalent of Theorem 1 for the group-Lasso. Relying on specific concentration inequalities established for infinitely divisible vectors by Houdré et al. (2008), it requires a known upper bound for f_0, which can be chosen as $\max_i Y_i$ in practice.

Theorem 2 *Let $k \in \{1, \dots, K\}$ be fixed and $\gamma > 0$ be a constant. Assume that there exists $M > 0$ such that for any x, $|f_0(x)| \leq M$. Let*

$$c_k = \sup_{\mathbf{x} \in \mathbb{R}^n} \frac{\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{x}\|_2}{\|\mathbf{A}_{G_k}^T\mathbf{x}\|_2}. \tag{2.7}$$

For all $j \in G_k$, still with $\widehat{V}_j = \sum_{i=1}^n \varphi_j^2(X_i)Y_i$, define

$$\widetilde{V}_j^g = \widehat{V}_j + \sqrt{2(\gamma \log p + \log|G_k|)\widehat{V}_j \max_i \varphi_j^2(X_i)} + 3(\gamma \log p + \log|G_k|) \max_i \varphi_j^2(X_i). \tag{2.8}$$

Let $\gamma > 0$ be fixed. Define $b_k^i = \sqrt{\sum_{j \in G_k} \varphi_j^2(X_i)}$ and $b_k = \max_i b_k^i$. Finally, we set

$$\lambda_k^g = \left(1 + \frac{1}{2\sqrt{2\gamma \log p}}\right) \sqrt{\sum_{j \in G_k} \widetilde{V}_j^g} + 2\sqrt{\gamma \log p \, D_k}, \tag{2.9}$$

where $D_k = 8Mc_k^2 + 16b_k^2\gamma \log p$. Then,

$$\mathbb{P}\left(\|\mathbf{A}_{G_k}^T(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])\|_2 \geq \lambda_k^g\right) \leq \frac{2}{p^\gamma}. \tag{2.10}$$

Similarly to the weights λ_j's of the Lasso, each weight λ_k^g is the sum of two terms. The term $\widetilde{V}_j^g$ is an estimate of V_j so it plays the same role as $\widetilde{V}_j$. In particular, $\widetilde{V}_j^g$ and $\widetilde{V}_j$ are of the same order since $\log|G_k|$ is not larger than $\log p$. The first term in λ_k^g is a variance term, and the leading constant $1 + 1/(2\sqrt{2\gamma \log p})$ is close to 1 when p is large. So, the first term is close to the square root of the sum of sharp estimates of the $(V_j)_{j \in G_k}$, as expected for a grouping strategy (see Chicken and Cai (2005)).

The second term, namely $2\sqrt{\gamma \log p \, D_k}$, is more involved. To shed light on it, since b_k and c_k play a key role, we first state the following proposition controlling values of these terms.

Proposition 1 *Let k be fixed. We have*

$$b_k \leq c_k \leq \sqrt{n} b_k. \tag{2.11}$$

Furthermore,

$$c_k^2 \leq \max_{j\in G_k} \sum_{j'\in G_k} \Big|\sum_{l=1}^n \varphi_j(X_l)\varphi_{j'}(X_l)\Big|. \tag{2.12}$$

The first inequality of Proposition 1 shows that $2\sqrt{\gamma \log p\, D_k}$ is smaller than $c_k\sqrt{\log p}+b_k \log p \leq 2c_k \log p$ up to a constant depending on γ and M. At first glance, the second inequality of Proposition 1 shows that c_k is controlled by the coherence of the dictionary (see Tropp (2004)) and b_k depends on $(\max_i |\varphi_j(X_i)|)_{j\in G_k}$. In particular, if for a given block G_k, the functions $(\varphi_j)_{j\in G_k}$ are orthonormal, then for fixed $j \neq j'$, if the X_i's are deterministic and equispaced on $[0,1]$ or if the X_i's are i.i.d. with a uniform density on $[0,1]^d$, then, when n is large

$$\frac{1}{n}\sum_{l=1}^n \varphi_j(X_l)\varphi_{j'}(X_l) \approx \int \varphi_j(x)\varphi_{j'}(x)dx = 0$$

and we expect

$$c_k^2 \lesssim \max_{j\in G_k} \sum_{l=1}^n \varphi_j^2(X_l).$$

In any case, by using the Cauchy-Schwarz Inequality, Condition (2.12) gives

$$c_k^2 \leq \max_{j\in G_k} \sum_{j'\in G_k} \left(\sum_{l=1}^n \varphi_j^2(X_l)\right)^{1/2} \left(\sum_{l=1}^n \varphi_{j'}^2(X_l)\right)^{1/2}. \tag{2.13}$$

To further discuss orders of magnitude for the c_k's, we consider the following condition

$$\max_{j\in G_k} \sum_{l=1}^n \varphi_j^2(X_l) = O(n), \tag{2.14}$$

which is satisfied for instance for fixed k if the design is regular, since $\|\varphi_j\|_2 = 1$. Under Assumption (2.14), Inequality (2.13) gives

$$c_k^2 = O(|G_k|n).$$

We can say more on b_k and c_k (and then on the order of magnitude of λ_k^g) by considering classical dictionaries of the literature to build the blocks G_k, which is of course realized in practice. In the subsequent discussions, the balance between $|G_k|$ and $\log p$ plays a key role. Note also that $\log p$ is the group size often recommended in the classical setting ($p = n$) for block thresholding (see Theorem 1 of Chicken and Cai (2005)).

2.2.1 Order of magnitude of λ_k^g by considering classical dictionaries.

Let G_k be a given block and assume that it is built by using only one of the subsequent systems. For each example, we discuss the order of magnitude of the term $D_k = 8Mc_k^2 + 16b_k^2\gamma \log p$. For ease of exposition, we assume that f_0 is supported by $[0,1]$ but we could easily generalize the following discussion to the multidimensional setting.

Bounded dictionary. Similarly to Blazere et al. (2014), we assume that there exists a constant L not depending on n and p such that for any $j \in G_k$, $\|\varphi_j\|_\infty \leq L$. For instance, atoms of the Fourier basis satisfy this property. We then have

$$b_k^2 \leq L^2|G_k|.$$

Finally, under Assumption (2.14),

$$D_k = O(|G_k|n + |G_k|\log p). \tag{2.15}$$

Compactly supported wavelets. Consider the one-dimensional Haar dictionary: For $j = (j_1, k_1) \in \mathbb{Z}^2$ we set $\varphi_j(x) = 2^{j_1/2}\psi(2^{j_1}x - k_1)$, $\psi(x) = 1_{[0,0.5]}(x) - 1_{]0.5,1]}(x)$. Assume that the block G_k depends on only one resolution level j_1: $G_k = \{j = (j_1, k_1) : \quad k_1 \in B_{j_1}\}$, where B_{j_1} is a subset of $\{0, 1, \ldots, 2^{j_1} - 1\}$. In this case, since for $j, j' \in G_k$ with $j \neq j'$, for any x, $\varphi_j(x)\varphi_{j'}(x) = 0$,

$$b_k^2 = \max_i \sum_{j\in G_k} \varphi_j^2(X_i) = \max_{i,j\in G_k} \varphi_j^2(X_i) = 2^{j_1}$$

and Inequality (2.12) gives

$$c_k^2 \leq \max_{j\in G_k} \sum_{l=1}^n \varphi_j^2(X_l).$$

If, similarly to Condition (2.5), we assume that $\max_{i,j\in G_k} |\varphi_j(X_i)| = o(\sqrt{n/\log p})$, then

$$b_k^2 = o(n/\log p),$$

and under Assumption (2.14),

$$D_k = O(n),$$

which improves (2.15). This property can be easily extended to general compactly supported wavelets ψ, since, in this case, for any $j = (j_1, k_1)$

$$S_j = \left\{j' = (j_1, k_1') : \quad k_1' \in \mathbb{Z},\ \varphi_j \times \varphi_{j'} \not\equiv 0\right\}$$

is finite with cardinal only depending on the support of ψ.

Regular histograms. Consider a regular grid of the interval $[0,1]$, $\{0,\delta,2\delta,\ldots\}$ with $\delta>0$. Consider then $(\varphi_j)_{j\in G_k}$ such that for any $j\in G_k$, there exists ℓ such that $\varphi_j=\delta^{-1/2}1_{(\delta(\ell-1),\delta\ell]}$. We have $\|\varphi_j\|_2=1$ and $\|\varphi_j\|_\infty=\delta^{-1/2}$. As for the wavelet case, for $j,j'\in G_k$ with $j\neq j'$, for any x, $\varphi_j(x)\varphi_{j'}(x)=0$, then

$$b_k^2=\max_i\sum_{j\in G_k}\varphi_j^2(X_i)=\max_{i,j\in G_k}\varphi_j^2(X_i)=\delta^{-1}.$$

If, similarly to Condition (2.5), we assume that $\max_{i,j\in G_k}|\varphi_j(X_i)|=o(\sqrt{n/\log p})$, then

$$b_k^2=o(n/\log p),$$

and under Assumption (2.14),

$$D_k=O(n).$$

The previous discussion shows that we can exhibit dictionaries such that c_k^2 and D_k are of order n and the term $b_k^2\log p$ is negligible with respect to c_k^2. Then, if similarly to Section 2.1, the terms $(\widetilde{V}_j^g)_{j\in G_k}$ are all of order n, λ_k^g is of order $\sqrt{n\times\max(\log p;|G_k|)}$ and the main term in λ_k^g is the first one as soon as $|G_k|\geq\log p$. In this case, λ_k^g is of order $\sqrt{|G_k|n}$.

2.2.2 Comparison with the Gaussian framework.

Now, let us compare the λ_k^g's to the weights proposed by Lounici et al. (2011) in the Gaussian framework. Adapting their notations to ours, Lounici et al. (2011) estimate the vector $\boldsymbol{\beta}_0$ in the model $\mathbf{Y}\sim\mathcal{N}(\mathbf{A}\boldsymbol{\beta}_0,\sigma^2\mathbf{I}_n)$ by using the group-Lasso estimate with weights equal to

$$\widetilde{\lambda}_k^g=2\sqrt{\sigma^2\Big(Tr(\mathbf{A}_{G_k}^T\mathbf{A}_{G_k})+2|||\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}|||(2\gamma\log p+\sqrt{|G_k|\gamma\log p})\Big)},$$

where $|||\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}|||$ denotes the maximal eigenvalue of $\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}$ (see (3.1) in Lounici et al. (2011)). So, if $|G_k|\leq\log p$, the above expression is of the same order as

$$\sqrt{\sigma^2Tr(\mathbf{A}_{G_k}^T\mathbf{A}_{G_k})}+\sqrt{\sigma^2|||\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}|||\gamma\log p}. \tag{2.16}$$

Neglecting the term $16b_k^2\gamma\log p$ in the definition of D_k (see the discussion in Section 2.2.1), we observe that λ_k^g is of the same order as

$$\sqrt{\sum_{j\in G_k}\widetilde{V}_j^g}+\sqrt{Mc_k^2\gamma\log p}. \tag{2.17}$$

Since M is an upper bound of $\mathrm{Var}(Y_i)=f_0(X_i)$ for any i, strong similarities can be highlighted between the forms of the weights in the Poisson and Gaussian settings:

- For the first terms, $\widetilde{V}_j^g$ is an estimate of V_j and

$$\sum_{j\in G_k}V_j\leq M\sum_{j\in G_k}\sum_{i=1}^n\varphi_j^2(X_i)=M\times Tr(\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}).$$

- For the second terms, in view of (2.7), c_k^2 is related to $|||\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}|||$ since we have

$$c_k^2=\sup_{\mathbf{x}\in\mathbb{R}^n}\frac{\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2}{\|\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2}\leq\sup_{\mathbf{y}\in\mathbb{R}^{|G_k|}}\frac{\|\mathbf{A}_{G_k}\mathbf{y}\|_2^2}{\|\mathbf{y}\|_2^2}=|||\mathbf{A}_{G_k}^T\mathbf{A}_{G_k}|||.$$

These strong similarities between the Gaussian and the Poissonian settings strongly support the shape relevance of the weights we propose.

2.2.3 Suboptimality of the naive procedure

Finally, we show that the naive procedure that considers $\sqrt{\sum_{j\in G_k}\lambda_j^2}$ instead of λ_k^g is suboptimal even if, obviously due to Theorem 1, with high probability,

$$\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2\leq\sqrt{\sum_{j\in G_k}\lambda_j^2}.$$

Suboptimality is justified by following heuristic arguments. Assume that for all j and k, the first terms in (2.3) and (2.9) are the main ones and $\widetilde{V}_j\approx\widetilde{V}_j^g\approx V_j$. Then by considering λ_k^g instead of $\sqrt{\sum_{j\in G_k}\lambda_j^2}$, we improve our weights by the factor $\sqrt{\log p}$, since in this situation,

$$\lambda_k^g\approx\sqrt{\sum_{j\in G_k}V_j}$$

and

$$\sqrt{\sum_{j\in G_k}\lambda_j^2}\approx\sqrt{\log p\sum_{j\in G_k}V_j}\approx\sqrt{\log p}\,\lambda_k^g.$$

Remember that our previous discussion shows the importance to consider weights as small as possible as soon as (2.6) is satisfied with high probability. The next section will confirm this point.

3. Oracle inequalities

In this section, we establish oracle inequalities to study theoretical properties of our estimation procedures. The X_i's are still assumption-free, and the performance of our procedures will be only evaluated at the X_i's. To measure the closeness between f_0 and an estimate,

we use the empirical Kullback-Leibler divergence associated with our model, denoted by $K(\cdot,\cdot)$. Straightforward computations (see for instance Leblanc and Letué (2006)) show that for any positive function f,

$$\begin{aligned} K(f_0,f) &= \mathbb{E}\left[\log\left(\frac{\mathcal{L}(f_0)}{\mathcal{L}(f)}\right)\right] \\ &= \sum_{i=1}^{n}\left[(f_0(X_i)\log f_0(X_i) - f_0(X_i))\right] - \left[(f_0(X_i)\log f(X_i) - f(X_i))\right], \end{aligned}$$

where $\mathcal{L}(f)$ is the likelihood associated with f. We speak about *empirical divergence* to emphasize its dependence on the X_i's. Note that we can write

$$K(f_0,f) = \sum_{i=1}^{n} f_0(X_i)(e^{u_i} - u_i - 1), \tag{3.1}$$

where $u_i = \log\frac{f(X_i)}{f_0(X_i)}$. This expression clearly shows that $K(f_0,f)$ is non-negative and $K(f_0,f) = 0$ if and only if for all $i \in \{1,\dots,n\}$, we have $u_i = 0$, that is $f(X_i) = f_0(X_i)$ for all $i \in \{1,\dots,n\}$.

Remark 1 *To weaken the dependence on n in the asymptotic setting, an alternative, not considered here, would consist in considering $n^{-1}K(\cdot,\cdot)$ instead of $K(\cdot,\cdot)$.*

If the classical $\mathbb{L}_2$-norm is the natural loss-function for penalized least squares criteria, the empirical Kullback-Leibler divergence is a natural alternative for penalized likelihood criteria. In next sections, oracle inequalities will be expressed by using $K(\cdot,\cdot)$.

3.1 Oracle inequalities for the group-Lasso estimate

In this section, we state oracle inequalities for the group-Lasso. These results can be viewed as generalizations of results by Lounici et al. (2011) to the case of the Poisson regression model. They will be established on the set Ω_g where

$$\Omega_g = \left\{\|\mathbf{A}_{G_k}^T(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])\|_2 \le \lambda_k^g \quad \forall\, k \in \{1,\dots,K\}\right\}. \tag{3.2}$$

Under assumptions of Theorem 2, we have $\mathbb{P}(\Omega_g) \ge 1 - \frac{2K}{p^\gamma} \ge 1 - 2p^{1-\gamma}$. By considering $\gamma > 1$, we have that $\mathbb{P}(\Omega_g)$ goes to 1 at a polynomial rate of convergence when p goes to $+\infty$. Even if our procedure is applied with weights defined in Section 2.2, note that subsequent oracle inequalities would hold for any weights $(\lambda_k^g)_{k=1,\dots,K}$ such that the associated set Ω_g has high probability. For any $\boldsymbol{\beta} \in \mathbb{R}^p$, we denote by

$$f_{\boldsymbol{\beta}}(x) = \exp\left(\sum_{j=1}^{p}\beta_j\varphi_j(x)\right),$$

the candidate associated with $\boldsymbol{\beta}$ to estimate f_0. We first give a *slow oracle inequality* (see for instance Bunea et al. (2007a), Gaïffas and Guilloux (2012) or Lounici et al. (2011)) that does not require any assumption.

Theorem 3 *On Ω_g,*

$$K(f_0,\widehat{f}^{gL}) \le \inf_{\boldsymbol{\beta}\in\mathbb{R}^p}\left\{K(f_0,f_\beta) + 2\sum_{k=1}^{K}\lambda_k^g\|\boldsymbol{\beta}_{G_k}\|_2\right\}. \tag{3.3}$$

Note that

$$\sum_{k=1}^{K}\lambda_k^g\|\boldsymbol{\beta}_{G_k}\|_2 \le \max_{k\in\{1,\dots,K\}}\lambda_k^g \times \|\boldsymbol{\beta}\|_{1,2}$$

and (3.3) is then similar to Inequality (3.9) of Lounici et al. (2011). We can improve the rate of (3.3) at the price of stronger assumptions on the matrix $\mathbf{A}$. We consider the following assumptions:

Assumption 1. We assume that

$$m := \max_{i\in\{1,\dots,n\}}|\log f_0(X_i)| < \infty.$$

This assumption is equivalent to assuming that f_0 is bounded from below and from above by positive constants. We do not assume that m is known in the sequel.

Assumption 2. For some integer $s \in \{1,\dots,K\}$ and some constant r, the following condition holds:

$$0 < \kappa_n(s,r) := \min\left\{\frac{(\boldsymbol{\beta}^T\mathbf{G}\boldsymbol{\beta})^{1/2}}{\|\boldsymbol{\beta}_J\|_2} :\ |J| \le s,\ \boldsymbol{\beta}\in\mathbb{R}^p\setminus\{0\},\ \sum_{k\in J^c}\lambda_k^g\|\boldsymbol{\beta}_{G_k}\|_2 \le r\sum_{k\in J}\lambda_k^g\|\boldsymbol{\beta}_{G_k}\|_2\right\},$$

where $\mathbf{G}$ is the Gram matrix defined by $\mathbf{G} = \mathbf{A}^T\mathbf{C}\mathbf{A}$, where $\mathbf{C}$ is the diagonal matrix with $C_{i,i} = f_0(X_i)$. With a slight abuse, $\boldsymbol{\beta}_J$ stands for the sub-vector of $\boldsymbol{\beta}$ with elements indexed by the indices of the groups $(G_k)_{k\in J}$.

This assumption is the natural extension of the classical *Restricted Eigenvalue condition* introduced by Bickel et al. (2009) to study the Lasso estimate where the ℓ_1-norm is replaced with the weighted $\|\cdot\|_{1,2}$-norm. In the Gaussian setting, Lounici et al. (2011) considered similar conditions to establish oracle inequalities for their group-Lasso procedure (see their Assumption (3.1)). RE-type assumptions are among the mildest ones to establish oracle inequalities (see van de Geer and Bühlmann (2009)). In particular, if the Gram matrix G has a positive minimal eigenvalue, say $d_{G,min}$, then Assumption 2 is satisfied with $\kappa_n^2(s,r) \ge d_{G,min}$. Assumption 2 can be connected to stronger coherence type conditions involving

the ratio $\frac{\max_k \lambda_k^g}{\min_k \lambda_k^g}$ (see Appendix B.3. of Lounici et al. (2011)). Furthermore, if c_0 is a positive lower bound for f_0, then for all $\boldsymbol{\beta} \in \mathbb{R}^p$,

$$\boldsymbol{\beta}^T \mathbf{G} \boldsymbol{\beta} = (\mathbf{A}\boldsymbol{\beta})^T \mathbf{C} (\mathbf{A}\boldsymbol{\beta}) \geq c_0 \|\mathbf{A}\boldsymbol{\beta}\|_2^2 = c_0 \sum_{i=1}^n \Big(\sum_{j=1}^p \beta_j \varphi_j(X_i) \Big)^2 = c_0 \sum_{i=1}^n g_{\boldsymbol{\beta}}^2(X_i),$$

with $g_{\boldsymbol{\beta}} = \sum_{j=1}^p \beta_j \varphi_j$. If $(\varphi_j)_{j \in \mathcal{J}}$ is orthonormal on $[0,1]^d$ and if the design is regular, then the last term is the same order as

$$n \int g_{\boldsymbol{\beta}}^2(x) dx = n \|\boldsymbol{\beta}\|_2^2 \geq n \|\boldsymbol{\beta}_J\|_2^2$$

for any subset $J \subset \{1, \ldots, K\}$. Under these assumptions, $\kappa_n^{-2}(s,r) = O(n^{-1})$.

We now introduce for any $\mu > 0$

$$\Gamma(\mu) = \left\{ \boldsymbol{\beta} \in \mathbb{R}^p : \quad \max_{i \in \{1,\ldots,n\}} \left| \sum_{j=1}^p \beta_j \varphi_j(X_i) \right| \leq \mu \right\}.$$

In the sequel, we restrict our attention to estimates belonging to the convex set $\Gamma(\mu)$. Of course, if m were known we would take $\mu = m$ (or μ a bit larger than m). Note that we do not impose any upper bound on μ so this condition is quite mild. The role of $\Gamma(\mu)$ consists in connecting $K(.,.)$ to some empirical quadratic loss functions (see the proof of Theorem 4). Alternative stronger assumptions have been considered by Lemler (2013) relying on van de Geer (2008) and Kong and Nan (2014). The value of μ only influences constants in subsequent oracle inequalities.

We consider the slightly modified group-Lasso estimate. Let $\alpha > 1$ and let us set

$$\widehat{\boldsymbol{\beta}}_{\mu,\alpha}^{gL} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \Gamma(\mu)} \Big\{ -l(\boldsymbol{\beta}) + \alpha \sum_{k=1}^K \lambda_k^g \|\boldsymbol{\beta}_{G_k}\|_2 \Big\}, \quad \widehat{f}_{\mu,\alpha}^{gL}(x) = \exp\left(\sum_{j=1}^p (\widehat{\boldsymbol{\beta}}_{\mu,\alpha}^{gL})_j \varphi_j(x) \right)$$

for which we obtain the following fast oracle inequality.

Theorem 4 *Let $\varepsilon > 0$ and s a positive integer. Let Assumption 2 be satisfied with s and*

$$r = \frac{\alpha + 1 + 2\alpha/\varepsilon}{\alpha - 1}.$$

Then, under Assumption 1, there exists a constant $B(\varepsilon, m, \mu)$ depending on ε, m and μ such that, on Ω_g,

$$K(f_0, \widehat{f}_{\mu,\alpha}^{gL}) \leq (1+\varepsilon) \inf_{\substack{\boldsymbol{\beta} \in \Gamma(\mu) \\ |J(\boldsymbol{\beta})| \leq s}} \left\{ K(f_0, f_{\boldsymbol{\beta}}) + B(\varepsilon, m, \mu) \frac{\alpha^2 |J(\boldsymbol{\beta})|}{\kappa_n^2} \times \left(\max_{k \in \{1,\ldots,K\}} \lambda_k^g \right)^2 \right\}, \quad (3.4)$$

where κ_n stands for $\kappa_n(s,r)$, and $J(\boldsymbol{\beta})$ is the subset of $\{1, \ldots, K\}$ such that $\boldsymbol{\beta}_{G_k} = \mathbf{0}$ if and only if $k \notin J(\boldsymbol{\beta})$.

The estimate $\widehat{\boldsymbol{\beta}}_{\mu,\alpha}^{gL}$ studied in Theorem 4 slightly differs from the estimate $\widehat{\boldsymbol{\beta}}^{gL}$ introduced in Section 1.2 since it depends on α and minimization is only performed on $\Gamma(\mu)$. Both estimates coincide when $\mu = +\infty$ and when $\alpha = 1$. An examination of the proof of Theorem 4 shows that $\lim_{\mu \to +\infty} B(\varepsilon, m, \mu) = +\infty$. In practice, we observe that most of the time, with $\alpha = 1$ and μ large enough, $\widehat{\boldsymbol{\beta}}_{\mu,\alpha}^{gL}$ and $\widehat{\boldsymbol{\beta}}^{gL}$ coincide. So, when α is close to 1 and μ is large, Theorem 4 gives a good flavor of theoretical performances satisfied by our group-Lasso procedure.

Let us comment each term of the right-hand side of (3.4). The first term is an approximation term, which can vanish if f_0 can be decomposed on the dictionary. The second term is a variance term, according to the usual terminology, which is proportional to the size of $J(\boldsymbol{\beta})$. Its shape is classical in the high dimensional setting. See for instance Theorem 3.2 of Lounici et al. (2011) for the group-Lasso in linear models, or Theorem 6.1 of Bickel et al. (2009) and Theorem 3 of Bertin et al. (2011) for the Lasso. If the order of magnitude of λ_k^g is $\sqrt{n \times \max(\log p; |G_k|)}$ (see Section 2.2.1) and if $\kappa_n^{-2} = O(n^{-1})$, the order of magnitude of this variance term is not larger than $|J(\boldsymbol{\beta})| \times \max(\log p; |G_k|)$. Finally, if f_0 can be well approximated (for the empirical Kullback-Leibler divergence) by a group-sparse combination of the functions of the dictionary, then the right hand side of (3.4) will take small values. So, the previous result justifies our group-Lasso procedure from the theoretical point of view. Note that (3.3) and (3.4) also show the interest of considering weights as small as possible.

Blazere et al. (2014) established rates of convergence under stronger assumptions, namely all coordinates of the analog of $\mathbf{A}$ are bounded by a quantity L, where L is viewed as a constant. Rates depend on L in an exponential manner and would highly deteriorate if L depended on n and p. So, this assumption is not reasonable if we consider dictionaries such as wavelets or histograms (see Section 2.2.1).

3.2 Oracle inequalities for the Lasso estimate

For the sake of completeness, we provide oracle inequalities for the Lasso. Theorems 3 and 4 that deal with the group-Lasso estimate can be adapted to the non-grouping strategy when we take groups of size 1. Subsequent results are similar to those established by Lemler (2013) who studied the Lasso estimate for the high-dimensional Aalen multiplicative intensity model. The block ℓ_1-norm $\|\cdot\|_{1,2}$ becomes the usual ℓ_1-norm and the group support $J(\boldsymbol{\beta})$ is simply the support of $\boldsymbol{\beta}$. As previously, we only work on the probability set Ω defined by

$$\Omega = \Big\{ |\mathbf{A}_j^T (\mathbf{Y} - \mathbb{E}[\mathbf{Y}])| \leq \lambda_j \quad \forall j \in \{1, \ldots, p\} \Big\}. \quad (3.5)$$

Theorem 1 asserts that $\mathbb{P}(\Omega) \geq 1 - \frac{3}{p^{\gamma - 1}}$ that goes to 1 as soon as $\gamma > 1$. As previously, subsequent oracle inequalities would hold for any weights $(\lambda_j)_{j=1,\ldots,p}$ such that the asso-

ciated probability set Ω has high probability. We obtain a slow oracle inequality for $\widehat{f}^L$:

Corollary 1 *On Ω,*

$$K(f_0, \widehat{f}^L) \leq \inf_{\boldsymbol{\beta}\in\mathbb{R}^p} \Big\{ K(f_0, f_{\boldsymbol{\beta}}) + 2\sum_{j=1}^{p} \lambda_j |\beta_j| \Big\}.$$

Now, let us consider fast oracle inequalities. In this framework, Assumption 2 is replaced with the following:

Assumption 3. For some integer $s \in \{1, \dots, p\}$ and some constant r, the following condition holds:

$$0 < \kappa_n(s, r) := \min \left\{ \frac{(\boldsymbol{\beta}^T \mathbf{G} \boldsymbol{\beta})^{1/2}}{\|\boldsymbol{\beta}_J\|_2} : \ |J| \leq s, \ \boldsymbol{\beta} \in \mathbb{R}^p \setminus \{0\}, \ \sum_{j\in J^c} \lambda_j |\boldsymbol{\beta}_j| \leq r \sum_{j\in J} \lambda_j |\boldsymbol{\beta}_j| \right\},$$

where $\mathbf{G}$ is the Gram matrix defined by $\mathbf{G} = \mathbf{A}^T \mathbf{C} \mathbf{A}$, where $\mathbf{C}$ is the diagonal matrix with $C_{i,i} = f_0(X_i)$.

As previously, we consider the slightly modified Lasso estimate. Let $\alpha > 1$ and let us set

$$\widehat{\boldsymbol{\beta}}^L_{\mu,\alpha} \in \operatorname*{argmin}_{\boldsymbol{\beta}\in\Gamma(\mu)} \Big\{ -l(\boldsymbol{\beta}) + \alpha \sum_{j=1}^{p} \lambda_j |\boldsymbol{\beta}_j| \Big\}, \quad \widehat{f}^L_{\mu,\alpha}(x) = \exp\left(\sum_{j=1}^{p} (\widehat{\boldsymbol{\beta}}^L_{\mu,\alpha})_j \varphi_j(x) \right)$$

for which we obtain the following fast oracle inequality.

Corollary 2 *Let $\varepsilon > 0$ and s a positive integer. Let Assumption 3 be satisfied with s and*

$$r = \frac{\alpha + 1 + 2\alpha/\varepsilon}{\alpha - 1}.$$

Then, under Assumption 1, there exists a constant $B(\varepsilon, m, \mu)$ depending on ε, m and μ such that, on Ω,

$$K(f_0, \widehat{f}^L_{\mu,\alpha}) \leq (1+\varepsilon) \inf_{\substack{\boldsymbol{\beta}\in\Gamma(\mu) \\ |J(\boldsymbol{\beta})|\leq s}} \left\{ K(f_0, f_{\boldsymbol{\beta}}) + B(\varepsilon, m, \mu) \frac{\alpha^2 |J(\boldsymbol{\beta})|}{\kappa_n^2} (\max_{j\in\{1,\dots,p\}} \lambda_j{}^2) \right\},$$

where κ_n stands for $\kappa_n(s, r)$, and $J(\boldsymbol{\beta})$ is the support of β.

This corollary is derived easily from Theorem 4 by considering all groups of size 1. Comparing Corollary 2 and Theorem 4, we observe that the group-Lasso can improve the Lasso estimate when the function f_0 can be well approximated by a function $f_{\boldsymbol{\beta}}$ so that the number of non-zero groups of $\boldsymbol{\beta}$ is much smaller than the total number of non-zero coefficients. The simulation study of the next section illustrates this comparison from the numerical point of view.

4. Simulation study

Simulation settings. Even if our theoretical results do not concern model selection properties, we propose to start the simulation study by considering a simple toy example such that $\log(f_0) = \mathbf{A}\boldsymbol{\beta}_0$ is generated from a true sparse vector of coefficients of size $p = 2^J$ in the Haar basis, such that $J = 10$ and scales $j = 1, 3, 4$ have all non-null coefficients. More details on this function can be found in our code that is freely available [1]. In this setting, $\log(f_0)$ can be decomposed on the dictionary and we can assess the accuracy of selection of different methods.

We then explore the empirical performance of the Lasso and the group Lasso strategies using simulations. By performance we mean the quality of reconstruction of simulated signals as our theoretical results concern reconstruction properties. We considered different forms for intensity functions by taking the standard functions of Donoho and Johnstone (1994): blocks, bumps, doppler, heavisine, to set g_0. These functions do not have an exact decomposition on any dictionary considered below. The signal to noise ratio was increased by multiplying the intensity functions by a factor α taking values in $\{1, \dots, 7\}$, $\alpha = 7$ corresponding to the most favorable configuration. Observations Y_i were generated such that $Y_i | X_i \sim \mathcal{P}oisson(f_0(X_i))$, with $f_0 = \alpha \exp(g_0)$, and $(X_1, \dots, X_n)$ was set as the regular grid of length $n = 2^{10}$. Each configuration was repeated 20 times. Our method was implemented using the `grpLasso` `R` package of Meier et al. (2008) to which we provide our concentration-based weights (the code is fully available [1]).

The basis and the dictionary frameworks. The dictionary we consider is built on the Haar basis, on the Daubechies basis with 6 vanishing moments, and on the Fourier basis, in order to catch piece-wise constant trends, localized peaks and periodicities. Each orthonormal system has n elements, which makes $p = n$ when systems are considered separately, and $p = 2n$ or $3n$ depending on the considered dictionary. For wavelets, the dyadic structure of the decomposition allows us to group the coefficients scale-wise by forming groups of coefficients of size 2^q. As for the Fourier basis, groups (also of size 2^q) are formed by considering successive coefficients (while keeping their natural ordering). When grouping strategies are considered, we set all groups at the same size.

Weights calibration in practice. First for both the Lasso and the group Lasso introduced in Section 1, following the arguments at the end of Section 2.1 that provide some theoretical guarantees, we take γ larger than 1. More precisely, we take $\gamma = 1.01$. Our theoretical results do not provide any information about values smaller than 1, so we conduct an empirical study on a very simple case, namely we estimate f_0 such that $\log(f_0) = 1_{[0,1]}$ on the Haar basis, so that only one parameter should be selected. Fig. 1 shows that when γ is too small, the risk of the selected models explodes (left panel), because too many coefficients are selected (right panel). Taking $\gamma = 1.01$ actually corresponds to a conservative choice avoiding the explosion of the MSE and of the number of selected

1. `http://pbil.univ-lyon1.fr/members/fpicard/software.html`

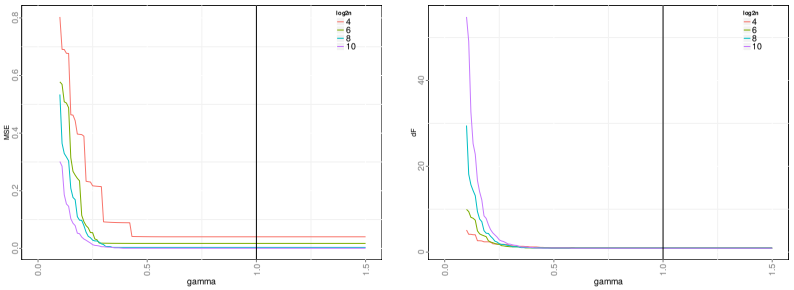


Figure 1: Tuning of the hyperparameter γ for the Lasso penalty. The simulated function is $\log(f_0) = 1_{[0,1]}$, so that only one parameter should be selected on the Haar basis. For different values of n, the left/right panels represent the mean square error / the number of estimated parameters with respect to γ (averaged over 20 simulations). The vertical line corresponds to $\gamma = 1$.

parameters. Furthermore this automatic choice provides good results in practice, as shown by our simulation study that considers many shapes of intensity functions (see below).

Then we estimate V_j (resp V_j^g) by $\widetilde{V}_j$ (resp $\widetilde{V}_j^g$). Note that $\widehat{V}_j$ (resp $\widehat{V}_j^g$) can also be used: these variances are easier to compute in practice, and this slight modification does not significantly change the performance of the procedures (not shown). The parameter γ being fixed, we use the expression (2.3) for Lasso weights. As for the group Lasso weights (Theorem 2), the first term is replaced by $\sqrt{\sum_{j\in G_k} \widetilde{V}_j}$, as it is governed by a quantity that tends to one when p is large. We conducted a preliminary numerical study (not shown) to calibrate the second term, that depends on quantities M, c_k and b_k defined in Theorem 2. The best empirical performance were achieved so that the left- and right-hand terms of (2.9) were approximatively equal. This resumes to group-Lasso weights of the form $2\sqrt{\sum_{j\in G_k} \widetilde{V}_j}$.

Competitors. We compete our Lasso procedure (`Lasso.exact` in the sequel), with the Haar-Fisz transform (using the `haarfisz` package), applied to the same data followed by soft-thresholding. Here we mention that we did not perform cycle-spinning (that is often included in denoising procedures) in order to focus on the effects of thresholding only. We also implemented the half-fold cross-validation proposed by Nason (1996) in the Poisson case to set the weights in the penalty, with the proper scaling ($2^{s/2}\lambda$, with s the scale of the wavelet coefficients) as proposed by Sardy et al. (2004). Briefly, this cross-validation procedure is used to calibrate λ by estimating f_0 on a training set made of even positions (for different values of λ), and by measuring the reconstruction error on the test set made

by odd positions (see Nason (1996)). This procedure is referred to cross-validation in the sequel. Then we compare the performance of the group-Lasso with varying group sizes (2,4,8) to the Lasso, to assess the benefits or grouping wavelet coefficients.

Performance measurement. For any estimate $\hat{f}$, reconstruction performance were measured using the (normalized) mean-squared error $MSE = \|\widehat{f} - f_0\|_2^2/\|f_0\|_2^2$. When there is a true sparse $\boldsymbol{\beta}_0$, model selection performance were measured by the standard indicators: accuracy on support recovery, sensitivity (proportion of true non-null coefficients among selected coefficients) and specificity of detection (proportion of true null coefficients among non-selected coefficients), based on the support of $\boldsymbol{\beta}_0$ and on the support of its estimate.

Model selection performance with a true sparse $\boldsymbol{\beta}_0$. The simple toy example described above for model selection perfectly shows the performance of our Lasso procedure (Figure 2). Even if our theoretical results do not concern support recovery, our theoretically calibrated weights provide the best accuracy in support selection, along with the best sensitivity, specificity and reconstruction errors. This example also shows the interest of the theoretical weights calibration compared with the cross-validation procedure that shows bad performance, and also compared with the Haar-Fisz transform. Also, this toy example illustrates the need of scaling for the cross-validated vanilla Lasso: if the procedure proposed by Sardy et al. (2004) is not used, the cross-validated vanilla Lasso lacks of adaptation to heteroscedasticity, which leads to poor results on selection (lack of accuracy and specificity, or too many selected coefficients), and thus to overfitted reconstruction (Fig. 3). Thus, in further simulations we only considered the scaled version of the cross-validated Lasso, in order to compare methods that account for heteroscedasticity.

Performance in the basis setting. Here we focus on wavelet basis (Haar for blocks and Daubechies for bumps, doppler and heavisine) and not on a dictionary approach (considered in a second step) in order to compare our calibrated weights with other methods that rely on penalized strategy. It appears that, except for the `bumps` function, the Lasso with exact weights shows the lowest reconstruction error whatever the shape of the intensity function (Figure 4). Moreover, better performance of the Lasso with exact weights in cases of low intensity emphasize the interest of theoretically calibrated procedures rather than asymptotic approximations (like the Haar-Fisz transform). In the case of `bumps`, cross-validation seems to perform better than the Lasso, but when looking at reconstructed average function (Figure 5a) this lower reconstruction error of cross-validation is associated with higher local variations around the peaks. Compared with Haar-Fisz, the gain of using exact weights is substantial even when the signal to noise ratio is high, which indicates that even in the validity domain of the Haar-Fisz transform (large intensities), the Lasso combined with exact thresholds is more suitable (Figure 5a). As for the group Lasso, its performance highly depend on the group size: while groups of size 2 show similar performance as the Lasso, groups of size 4 and 8 increase the reconstruction error (Figure

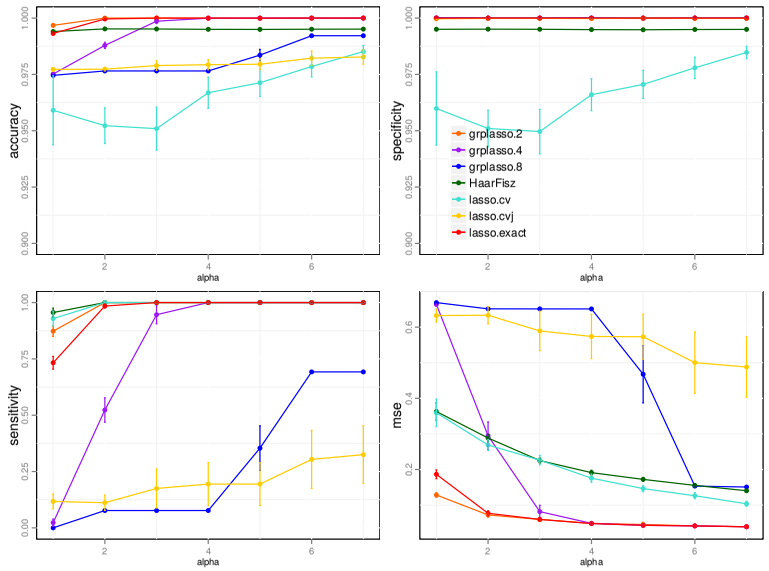


Figure 2: Average (over 20 repetitions) accurracy in support recovery for the toy function f_0 with a true sparse $\boldsymbol{\beta}_0$ and the Haar basis, specificity and sensitivity of selection, and Mean Square Error of reconstruction. `Lasso.exact`: Lasso penalty with our data-driven theoretical weights, `Lasso.cv`: Lasso penalty with weights calibrated by cross validation without scaling, `Lasso.cvj`: Lasso penalty with weights calibrated by cross validation with scaling $2^{s/2}\lambda$, `group.Lasso.2/4/8`: group Lasso penalty with our data-driven theoretical weights with group sizes 2/4/8, `HaarFisz`: Haar-Fisz tranform followed by soft-thresholding. Alpha (α) stands for the signal strength.

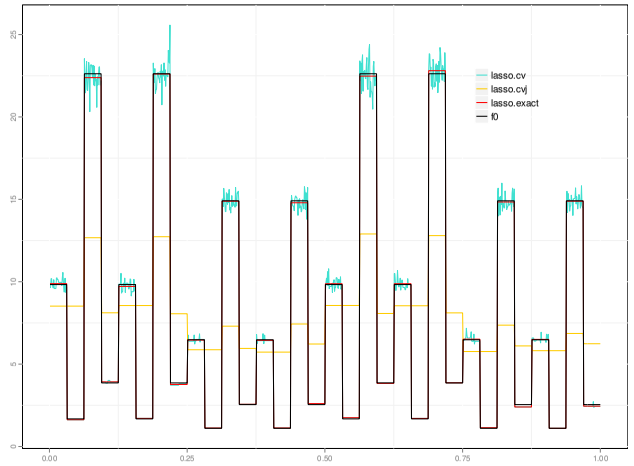


Figure 3: Example of reconstruction for the toy function f_0 with a true sparse $\boldsymbol{\beta}_0$ and the Haar basis. The reconstruction with the vanilla-lasso is based on too many coefficients which leads to over-fitting. `Lasso.exact`: Lasso penalty with our data-driven theoretical weights, `Lasso.cv`: Lasso penalty with weights calibrated by cross validation without scaling, `Lasso.cvj`: Lasso penalty with weights calibrated by cross validation with scaling $2^{s/2}\lambda$

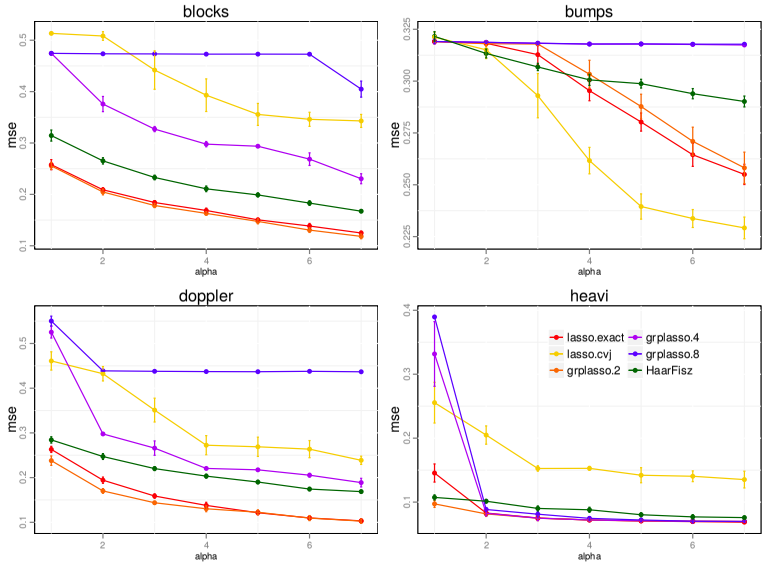


Figure 4: Average (over 20 repetitions) Mean Square Error of reconstruction of different methods for the estimation of simulated intensity functions according to function shapes (blocks, bumps, doppler, heavisine) and signal strength (α). `Lasso.exact`: Lasso penalty with our data-driven theoretical weights, `Lasso.cvj`: Lasso penalty with weights calibrated by cross validation with scaling $2^{s/2}\lambda$, `group.Lasso.2/4/8`: group Lasso penalty with our data-driven theoretical weights with group sizes 2/4/8, `HaarFisz`: Haar-Fisz tranform followed by soft-thresholding.

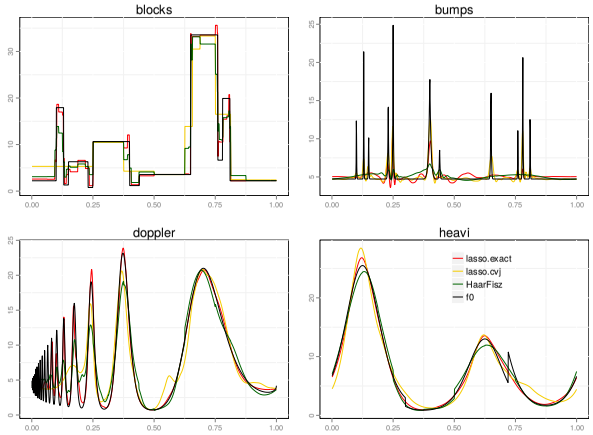
4 and 5b), since they are not scaled to the size of the irregularities in the signal. This trend is not systematic as the group Lasso appears to be adapted to functions that are more regular (Heavisine), and seems to avoid edge effects in some situations.

Performance in the dictionary framework. Lastly, we explored the performance of the dictionary approach, by considering different dictionaries to estimate each function: Daubechies (D), Fourier (F), Haar (H), or their combinations (Figure 6). Rich dictionaries can be very powerful to catch complex shapes in the true intensity function (like the notch in the heavisine case Figure 6b), and the richest dictionary (DFH) often leads to the lowest reconstruction error (MSE) on average. However the richest dictionary (DFH) is not always

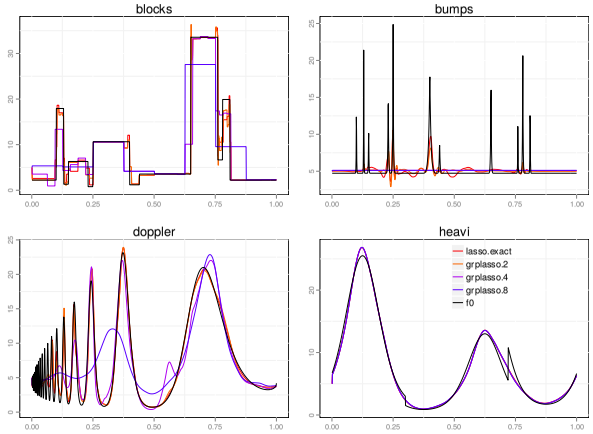
the best choice in terms of reconstruction error, which is stricking in the case of the `blocks` function. In this case the Haar system only would be preferable for the Lasso (Figure 6a). For the group-Lasso and the `blocks` intensity function, the combination of the Daubechies and the Haar systems provides the best MSE, but when looking at the reconstructed intensity (Figure 6b-blocks), the Daubechies system introduces wiggles that are not relevant for `blocks`. Also, richer dictionaries do not necessarily lead to more selected parameters (Figure 6a), which illustrates that selection depends on the redundancies between the systems elements of the dictionary. In practice we often do not have any *prior* knowledge concerning the elements that shape the signal, and these simulations suggest that the blind use of the richest dictionary may not be the best strategy in terms of reconstructed functions. Consequently, in the following application, we propose to adapt the half-fold cross validation of Nason (1996) to choose the best combinations of systems.

5. Applications

The analysis of biological data has faced a new challenge with the extensive use of next generation sequencing (NGS) technologies. NGS experiments are based on the massive parallel sequencing of short sequences (reads). The mapping of these reads onto a reference genome (when available) generates counts data (Y_t) spatially organized (in 1D) along the genome (at position X_t). These technologies have revolutionized the perspectives of many fields in molecular biology, and among many applications, one is to get a local quantification of DNA or of a given DNA-related molecule (like transcription factors for instance with chIP-Seq experiments, Furey (2012)). This technology has recently been applied to the identification of replication origins along the human genome. Replication is the process by which a genome is duplicated into two copies. This process is tightly regulated in time and space so that the duplication process takes place in the highly regulated cell cycle. The human genome is replicated at many different starting points called origins of replication, that are loci along the genome at which the replication starts. Until very recently, the number of such origins remained controversial, and thanks to the application of NGS technologies, first estimates of this number could be obtained. The signal is made of counts along the human genome such that reads accumulations indicate an origin activity (see Picard et al. (2014)). Scan statistics were first applied to these data, to detect significant local enrichments reads accumulation, but there is currently no consensus on the best method to analyze such data. Here we propose to use the Poisson functional regression to estimate the intensity function of the data on a portion of the human chromosomes X and 20. Half-fold cross-validation was used to select the appropriate dictionary between Daubechies, Fourier, Haar (and their combinations), and our theoretical weights were used to calibrate the Lasso (Figure 7). Our results are very promising as the sparse dictionary approach is very efficient for denoising (Chromosome X, Figure 7b) and produces null intensities when the signal is low (higher specificity). Another aspect of our method is that it seems to be more powerful in the identification of peaks that are more precise (Chromosome 20,

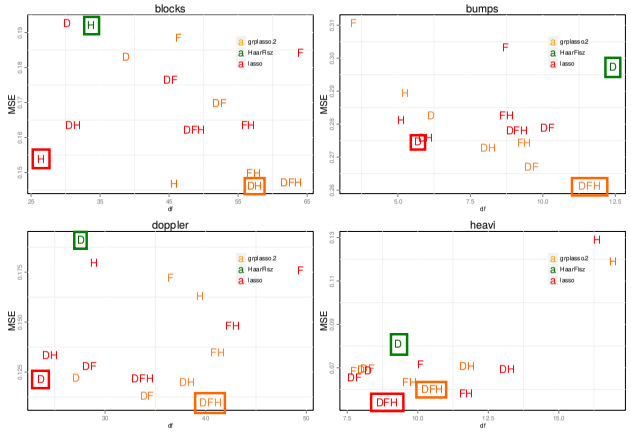


(a) Average reconstructed functions for the Lasso and competitors.

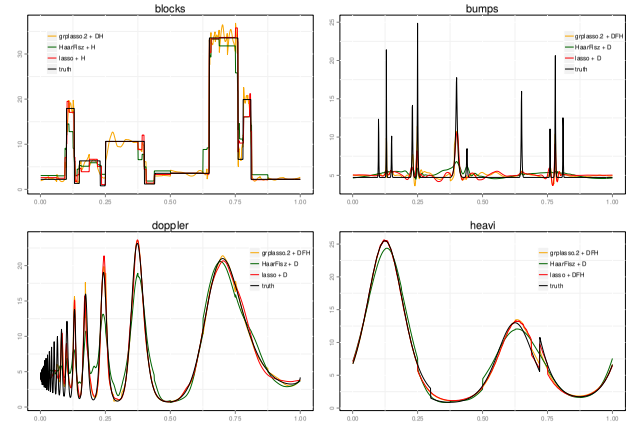


(b) Average reconstructed functions for the group strategies.

Figure 5: Average (over 20 repetitions) reconstructed functions by different methods of estimation according to function shapes (blocks, bumps, doppler, heavisine). Top panel corresponds to non-grouped strategies (5a) and bottom panel compares group-strategies to the Lasso (5b). `Lasso.exact`: Lasso penalty with our data-driven theoretical weights, `Lasso.cvj`: Lasso penalty with weights calibrated by cross validation with scaling $2^{s/2}\lambda$, `group.Lasso.2/4/8`: group Lasso penalty with our data-driven theoretical weights with group sizes 2/4/8, `HaarFisz`: Haar-Fisz tranform followed by soft-thresholding, `f0`: simulated intensity function.



(a) Average Mean Square Error for different dictionaries with respect to the average number of selected coefficients (df).



(b) Reconstructed functions for the dictionaries with the smallest MSE.

Figure 6: Average (over 20 repetitions) Mean Square Errors and number of selected coefficients (df) (6a), and reconstructed functions (6b) for different dictionaries: Daubechies (D), Fourier (F), Haar (H) and their combinations. `Lasso.exact`: Lasso penalty with our data-driven theoretical weights, `group.Lasso.2`: group Lasso penalty with our data-driven theoretical weights with group sizes 2, `HaarFisz`: Haar-Fisz tranform followed by soft-thresholding.

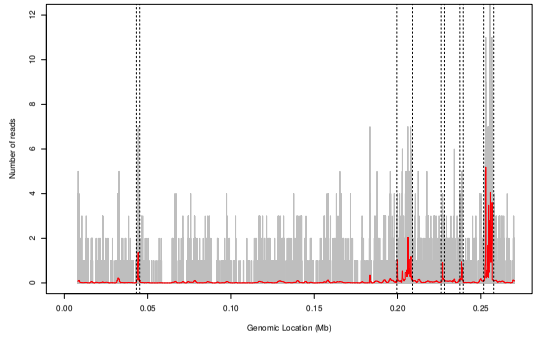
positions 0.20 and 0.25Mb, Figure 7a), which indicates that the dictionary approach may be more sensitive to detect peaks. Given the spread of NGS data and the importance of peak detection in the analysis process, for chIP-Seq Furey (2012), FAIRE-Seq Thurman et al. (2012), OriSeq Picard et al. (2014), our preliminary results suggest that the sparse dictionary approach will be a very promising framework for the analysis of such data.

6. Conclusion

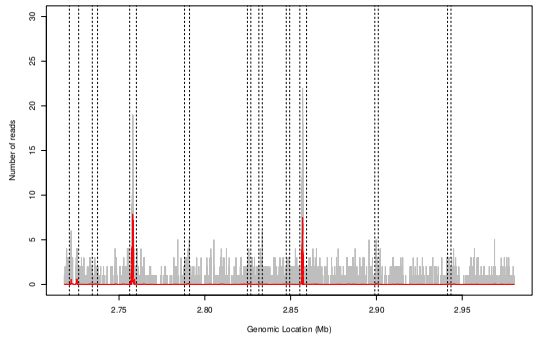
We proposed new adaptive Lasso and group-Lasso procedures to estimate the regression function in high dimensional Poisson regression, with a special focus on the calibration of weights involved in the penalties. Inspired from the adaptive Lasso procedure proposed by Zou (2006) in the context of model selection, we derived data-driven component-specific weights, whose shape appears to be relevant from both theoretical and practical points of view. In the dictionary approach, which extends the classical basis approach, we obtain slow and fast oracle inequalities under RE-type conditions. These theoretical results are enhanced by a numerical study that illustrates the good performance of our procedure on simulated and experimental data. Moreover, even if our main objective is functional reconstruction (that is different from model selection), a numerical toy example shows very promising results of our Lasso and group-Lasso procedures for model selection. The theoretical study of this problem is an exciting challenge we wish to investigate in further works.

The purpose of our data-driven weights is to control the random fluctuations of the normalized observations, namely $(\mathbf{A}_j^T\mathbf{Y})_j$ and $(\mathbf{A}_{G_k}^T\mathbf{Y})_k$, around their expectation. To obtain controls as sharp as possible, we use concentration inequalities for infinitely divisible vectors, which allows us to account for the heteroscedasticity that characterizes the Poisson setting. So, our approach is very different from Zou (2006) who considered weights proportional to the inverse of preliminary estimates (ordinary least squares estimates or maximum likelihood estimates). But both approaches confirm that random weighting schemes can provide suitable procedures for inference in regression models.

Finally, we mention that the constants of our procedure were tuned by using theoretical arguments, and our numerical studies show that they are suitable. Other values for these constants would probably be appropriate for some signals. Even if our empirical results show that the performance with $\gamma = 1.01$ are excellent, whatever the form of f_0 we considered, our procedure could be enriched by an additional calibration step for γ, with associated extra computational time.



(a) Chromosome 20



(b) Chromosome X

Figure 7: Estimation of the intensity function of Ori-Seq data (chromosomes 20 7a and X 7b). Grey bars indicate the number of reads that match genomic positions (x-axis, in MegaBases). The red line corresponds to the estimated intensity function, and vertical dotted lines stand for the detected origins by scanning statistics.

7. Proofs

7.1 Proof of Theorem 1

We denote by μ the Lebesgue measure on $\mathbb{R}^d$ and we introduce a partition of the set $[0,1]^d$ denoted $\cup_{i=1}^n S_i$ so that for any $i=1,\dots,n$, $X_i \in S_i$ and $\mu(S_i)>0$. Let h the function defined for any $t\in[0,1]^d$ by

$$h(t)=\sum_{i=1}^n \frac{f_0(X_i)}{\mu(S_i)} 1_{S_i}(t).$$

Finally, we introduce N the Poisson process on $[0,1]^d$ with intensity h (see Kingman (1993)). Therefore, for any $i=1,\dots,n$, $N(S_i)$ is a Poisson variable with parameter $\int_{S_i} h(t)dt = f_0(X_i)$ and since $\cup_{i=1}^n S_i$ is a partition of $[0,1]^d$, $(N(S_1),\dots,N(S_n))$ has the same distribution as $(Y_1,\dots,Y_n)$. We observe that if for any $j=1,\dots,p$,

$$\widetilde{\varphi}_j(t)=\sum_{i=1}^n \varphi_j(X_i)1_{S_i}(t),$$

then

$$\int \widetilde{\varphi}_j(t)dN(t) \sim \sum_{i=1}^n \varphi_j(X_i)Y_i = \mathbf{A}_j^T\mathbf{Y}.$$

We use the following exponential inequality (see Inequality (5.2) of Reynaud-Bouret (2003)). If g is bounded, for any $u>0$,

$$\mathbb{P}\left(\int g(x)(dN(x)-h(x)dx) \geq \sqrt{2u\int g^2(x)h(x)dx} + \frac{u}{3}\|g\|_\infty\right) \leq \exp(-u). \tag{7.1}$$

By taking successively $g=\widetilde{\varphi}_j$ and $g=-\widetilde{\varphi}_j$, we obtain

$$\mathbb{P}\left(|\mathbf{A}_j^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])| \geq \sqrt{2u\int \widetilde{\varphi}_j^2(x)h(x)dx} + \frac{u}{3}\|\widetilde{\varphi}_j\|_\infty\right) \leq 2e^{-u}.$$

Since

$$\int \widetilde{\varphi}_j^2(x)h(x)dx = \sum_{i=1}^n \varphi_j^2(X_i)f_0(X_i) = V_j,$$

we obtain

$$\mathbb{P}\left(|\mathbf{A}_j^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])| \geq \sqrt{2uV_j} + \frac{u}{3}\|\widetilde{\varphi}_j\|_\infty\right) \leq 2e^{-u}. \tag{7.2}$$

To control V_j, we use (7.1) with $g=-\widetilde{\varphi}_j^2$ and we have:

$$\mathbb{P}\left(V_j - \widehat{V}_j \geq \sqrt{2u\int \widetilde{\varphi}_j^4(t)h(t)dt} + \frac{u}{3}\|\widetilde{\varphi}_j\|_\infty^2\right) \leq e^{-u}.$$

We observe that

$$\int \widetilde{\varphi}_j^4(t)h(t)dt \leq \|\widetilde{\varphi}_j\|_\infty^2 \int \widetilde{\varphi}_j^2(t)h(t)dt = \|\widetilde{\varphi}_j\|_\infty^2 V_j.$$

Setting $v_j = u\|\widetilde{\varphi}_j\|_\infty^2$, we have:

$$\mathbb{P}\left(V_j - \sqrt{2v_jV_j} - \frac{v_j}{3} - \widehat{V}_j \geq 0\right) \leq e^{-u}.$$

Let $\alpha_j = \sqrt{\widehat{V}_j + \frac{5}{6}v_j} + \sqrt{\frac{v_j}{2}}$, such that α_j is the positive solution to $\alpha_j^2 - \sqrt{2v_j}\alpha_j - (\widehat{V}_j + \frac{v_j}{3}) = 0$. Then

$$\mathbb{P}\Big(V_j \geq \alpha_j^2\Big) = \mathbb{P}\Big(\sqrt{V_j} \geq \alpha_j\Big) \leq e^{-u}. \tag{7.3}$$

We choose $u=\gamma\log p$ and observe that $\alpha_j^2 \leq \widetilde{V}_j$. Then, by combining (7.2) and (7.3), we have

$$\mathbb{P}\left(|\mathbf{A}_j^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])| \geq \sqrt{2\gamma\log p\widetilde{V}_j} + \frac{\gamma\log p}{3}\|\widetilde{\varphi}_j\|_\infty\right) \leq \frac{3}{p^\gamma}.$$

As $\|\widetilde{\varphi}_j\|_\infty = \max_i|\varphi_j(X_i)|$, the theorem follows. □

Remark 2 *By slightly extending previous computations, we easily show that for $u>0$,*

$$\mathbb{P}\left(|V_j - \widehat{V}_j| \geq \sqrt{2uV_j\|\widetilde{\varphi}_j\|_\infty^2} + \frac{u}{3}\|\widetilde{\varphi}_j\|_\infty^2\right) \leq 2e^{-u},$$

which leads to

$$\mathbb{P}\left(|V_j - \widehat{V}_j| \geq \frac{V_j}{2} + \frac{4\gamma\log p}{3}\|\widetilde{\varphi}_j\|_\infty^2\right) \leq \frac{2}{p^\gamma}.$$

7.2 Proof of Theorem 2

For each $k\in\{1,\dots,K\}$, we recall that $b_k^i = \sqrt{\sum_{j\in G_k}\varphi_j^2(X_i)}$, so $b_k^i = \|\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2$, where $\mathbf{e}_i$ is the vector whose i-th coordinate is equal to 1 and all others to 0. We first state the following lemma:

Lemma 1 *Let k be fixed. Assume that there exists some $M>0$ such that $\forall\, x, |f_0(x)|\leq M$. Assume further that there exists some $c_k\geq 0$ such that $\forall\,\mathbf{y}\in\mathbb{R}^n, \|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{y}\|_2 \leq c_k\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2$. Then, $\forall\, x>0, \forall\,\varepsilon>0$,*

$$\mathbb{P}\left(\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 \geq (1+\varepsilon)\sqrt{\sum_{j\in G_k} V_j} + x\right) \leq \exp\left(\frac{x}{b_k} - \Big(\frac{x}{b_k} + \frac{D_k^\varepsilon}{b_k^2}\Big)\log\Big(1+\frac{b_k x}{D_k^\varepsilon}\Big)\right),$$

where $D_k^\varepsilon = 8Mc_k^2 + \frac{2}{\varepsilon^2}b_k^2$.

Proof With $k \in \{1,\dots,K\}$ being fixed, we define $f:\mathbb{R}^n \to \mathbb{R}$ by $f(\mathbf{y}) = \Big(\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2 - E\Big)_+$, where $E>0$ is a constant chosen later. We use Corollary 1 from Houdré et al. (2008), applied to the infinitely divisible vector $\mathbf{Y}-\mathbb{E}[\mathbf{Y}] \in \mathbb{R}^n$, whose components are independent, and to f. First note that for any $t>0$,

$$\begin{aligned}\mathbb{E}e^{tb_k^i|Y_i-\mathbb{E}Y_i|} &\leq \mathbb{E}e^{tb_k^i(Y_i+f_0(X_i))}\\ &= \exp\Big(f_0(X_i)(e^{tb_k^i}+tb_k^i-1)\Big) < \infty.\end{aligned}$$

Furthermore, for any $i\in\{1,...,n\}$, any $\mathbf{y}\in\mathbb{R}^n$ and any $u\in\mathbb{R}$,

$$\begin{aligned}|f(\mathbf{y}+u\mathbf{e}_i)-f(\mathbf{y})| &\leq \Big|\|\mathbf{A}_{G_k}^T(\mathbf{y}+u\mathbf{e}_i)\|_2 - \|\mathbf{A}_{G_k}^T\mathbf{y}\|_2\Big|\\ &\leq \|\mathbf{A}_{G_k}^T(u\mathbf{e}_i)\|_2\\ &= |u|b_k^i.\end{aligned}$$

Therefore, for all $x>0$,

$$\mathbb{P}\Big(f(\mathbf{Y}-\mathbb{E}[\mathbf{Y}]) - \mathbb{E}[f(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])] \geq x\Big) \leq \exp\Big(-\int_0^x h_f^{-1}(s)ds\Big),$$

where h_f is defined for all $t>0$ by

$$h_f(t) = \sup_{\mathbf{y}\in\mathbb{R}^n}\sum_{i=1}^n\int_{\mathbb{R}}|f(\mathbf{y}+u\mathbf{e}_i)-f(\mathbf{y})|^2\frac{e^{tb_k^i|u|}-1}{b_k^i|u|}\widetilde{\nu}_i(du)$$

and $\widetilde{\nu}_i$ is the Lévy measure associated with $Y_i-\mathbb{E}[Y_i]$. It is easy to show that $\widetilde{\nu}_i = f_0(X_i)\delta_1$, and so

$$h_f(t) = \sup_{\mathbf{y}\in\mathbb{R}^n}\sum_{i=1}^n f_0(X_i)\Big(f(\mathbf{y}+\mathbf{e}_i)-f(\mathbf{y})\Big)^2\frac{e^{tb_k^i}-1}{b_k^i}.$$

Furthermore, writing $A_i = \Big\{\|\mathbf{A}_{G_k}^T(\mathbf{y}+\mathbf{e}_i)\|_2 \geq E \text{ or } \|\mathbf{A}_{G_k}^T\mathbf{y}\|_2 \geq E\Big\}$, we have

$$\begin{aligned}|f(\mathbf{y}+\mathbf{e}_i)-f(\mathbf{y})| &\leq \Big|\|\mathbf{A}_{G_k}^T(\mathbf{y}+\mathbf{e}_i)\|_2 - \|\mathbf{A}_{G_k}^T\mathbf{y}\|_2\Big|1_{A_i}\\ &= \frac{1_{A_i}\Big|\|\mathbf{A}_{G_k}^T(\mathbf{y}+\mathbf{e}_i)\|_2^2 - \|\mathbf{A}_{G_k}^T\mathbf{y}\|_2^2\Big|}{\|\mathbf{A}_{G_k}^T(\mathbf{y}+\mathbf{e}_i)\|_2 + \|\mathbf{A}_{G_k}^T\mathbf{y}\|_2}\\ &= \frac{1_{A_i}\Big|2<\mathbf{A}_{G_k}^T\mathbf{e}_i,\mathbf{A}_{G_k}^T\mathbf{y}> + \|\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2^2\Big|}{\|\mathbf{A}_{G_k}^T(\mathbf{y}+\mathbf{e}_i)\|_2 + \|\mathbf{A}_{G_k}^T\mathbf{y}\|_2}\\ &\leq 2\frac{\Big|<\mathbf{A}_{G_k}^T\mathbf{e}_i,\mathbf{A}_{G_k}^T\mathbf{y}>\Big|}{\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2} + \frac{\|\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2^2}{E},\end{aligned}$$

with $<\cdot,\cdot>$ the usual scalar product. We now have

$$\Big(f(\mathbf{y}+\mathbf{e}_i)-f(\mathbf{y})\Big)^2 \leq 8\frac{<\mathbf{A}_{G_k}^T\mathbf{e}_i,\mathbf{A}_{G_k}^T\mathbf{y}>^2}{\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2^2} + 2\frac{\|\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2^4}{E^2}.$$

The first term can be rewritten as $8\frac{<\mathbf{e}_i,\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{y}>^2}{\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2^2}$ and the second one is equal to $2\frac{{b_k^i}^4}{E^2}$, so we can now bound $h_f(t)$ as follows.

$$\begin{aligned}h_f(t) &\leq \sup_{\mathbf{y}}\sum_i f_0(X_i)\frac{e^{tb_k^i}-1}{b_k^i}\left(8\frac{<\mathbf{e}_i,\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{y}>^2}{\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2^2} + 2\frac{{b_k^i}^4}{E^2}\right)\\ &\leq \frac{e^{tb_k}-1}{b_k}\sup_{\mathbf{y}}\left(8M\frac{\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{y}\|_2^2}{\|\mathbf{A}_{G_k}^T\mathbf{y}\|_2^2} + \frac{2}{E^2}\sum_i f_0(X_i){b_k^i}^4\right)\\ &\leq \frac{e^{tb_k}-1}{b_k}\left(8Mc_k^2 + \frac{2}{E^2}\sum_i f_0(X_i){b_k^i}^4\right).\end{aligned}$$

Now, we set

$$E = \varepsilon\sqrt{\sum_{j\in G_k}V_j}.$$

So we have:

$$\begin{aligned}E^2 &= \varepsilon^2\sum_{j\in G_k}\sum_{i=1}^n f_0(X_i)\varphi_j^2(X_i)\\ &= \varepsilon^2\sum_{i=1}^n f_0(X_i)\sum_{j\in G_k}\varphi_j^2(X_i)\\ &= \varepsilon^2\sum_{i=1}^n f_0(X_i){b_k^i}^2.\end{aligned}$$

Thus, we can finally bound the function h_f by the increasing function h defined by

$$h(t) = D_k^\varepsilon\frac{e^{tb_k}-1}{b_k},$$

with $D_k^\varepsilon = 8Mc_k^2 + \frac{2b_k^2}{\varepsilon^2}$. Therefore,

$$\begin{aligned}\exp\Big(-\int_0^x h_f^{-1}(s)ds\Big) &\leq \exp\Big(-\int_0^x h^{-1}(s)ds\Big)\\ &= \exp\left(\frac{x}{b_k} - \Big(\frac{x}{b_k}+\frac{D_k^\varepsilon}{b_k^2}\Big)\log\Big(1+\frac{b_k x}{D_k^\varepsilon}\Big)\right).\end{aligned}$$

Now,

$$\begin{aligned} f(\mathbf{Y}-\mathbb{E}[\mathbf{Y}]) - \mathbb{E}[f(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])] &= \Big(\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 - E\Big)_+ - \mathbb{E}\Big(\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 - E\Big)_+ \\ &\geq \|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 - E - \mathbb{E}\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2. \end{aligned}$$

Furthermore, by Jensen's inequality, we have

$$\begin{aligned} \mathbb{E}\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 &\leq \sqrt{\mathbb{E}\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2^2} \\ &= \sqrt{\sum_{j\in G_k}\mathbb{E}[(\mathbf{A}_j^T(\mathbf{Y}-\mathbb{E}\mathbf{Y}))^2]} \\ &= \sqrt{\sum_{j\in G_k}\mathrm{Var}(\mathbf{A}_j^T\mathbf{Y})} \\ &= \sqrt{\sum_{j\in G_k} V_j}. \end{aligned}$$

Recalling that $E = \varepsilon\sqrt{\sum_{j\in G_k} V_j}$, we thus have

$$\mathbb{P}\Big(f(\mathbf{Y}-\mathbb{E}[\mathbf{Y}]) - \mathbb{E}f(\mathbf{Y}-\mathbb{E}[\mathbf{Y}]) \geq x\Big) \geq \mathbb{P}\Big(\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 - (1+\varepsilon)\sqrt{\sum_{j\in G_k} V_j} \geq x\Big),$$

which concludes the proof. ■

We apply Lemma 1 with

$$\varepsilon = \frac{1}{2\sqrt{2\gamma\log p}} \quad \text{and} \quad x = 2\sqrt{\gamma\log p D_k^\varepsilon}.$$

Then,

$$\begin{aligned} \frac{b_k x}{D_k^\varepsilon} &= \frac{2b_k\sqrt{\gamma\log p}}{\sqrt{D_k^\varepsilon}} \\ &= \frac{2b_k\sqrt{\gamma\log p}}{\sqrt{8Mc_k^2 + \frac{2b_k^2}{\varepsilon^2}}} \\ &\leq \varepsilon\sqrt{2\gamma\log p} = \frac{1}{2}. \end{aligned}$$

Finally, using the fact that $\log(1+u) \geq u - \frac{u^2}{2}$, we have:

$$\begin{aligned} \exp\left(\frac{x}{b_k} - \Big(\frac{x}{b_k} + \frac{D_k^\varepsilon}{b_k^2}\Big)\log\Big(1+\frac{b_k x}{D_k^\varepsilon}\Big)\right) &\leq \exp\left(\frac{x}{b_k} - \Big(\frac{x}{b_k} + \frac{D_k^\varepsilon}{b_k^2}\Big)\Big(\frac{b_k x}{D_k^\varepsilon} - \frac{b_k^2 x^2}{2D_k^{\varepsilon 2}}\Big)\right) \\ &= \exp\left(\frac{-x^2}{2D_k^\varepsilon} + \frac{b_k x^3}{2D_k^{\varepsilon 2}}\right) \\ &= \exp\left(\frac{-x^2}{2D_k^\varepsilon}\Big(1 - \frac{b_k x}{D_k^\varepsilon}\Big)\right) \\ &\leq \exp\Big(\frac{-x^2}{4D_k^\varepsilon}\Big) = \frac{1}{p^\gamma}. \end{aligned}$$

We obtain

$$\mathbb{P}\left(\|\mathbf{A}_{G_k}^T(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\|_2 \geq (1+\varepsilon)\sqrt{\sum_{j\in G_k} V_j} + 2\sqrt{\gamma\log p D_k^\varepsilon}\right) \leq \frac{1}{p^\gamma}.$$

We control V_j as in the proof of Theorem 1, but we take $u = \gamma\log p + \log|G_k|$. The analog of (7.3) is

$$\mathbb{P}\Big(V_j > \widetilde{V}_j^g\Big) \leq e^{-u} = \frac{1}{|G_k|p^\gamma}$$

and thus

$$\mathbb{P}\Big(\exists\, j\in G_k,\, V_j > \widetilde{V}_j^g\Big) \leq \frac{1}{p^\gamma}.$$

This concludes the proof of Theorem 2. □

7.3 Proof of Proposition 1

For the first point, we write:

$$\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2 = \sum_{l=1}^n\left(\sum_{j\in G_k}\varphi_j(X_l)\sum_{i=1}^n\varphi_j(X_i)x_i\right)^2.$$

Then, we apply the Cauchy-Schwarz inequality:

$$\begin{aligned}
\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2 &\leq \sum_{l=1}^{n}\left(\sum_{j\in G_k}\varphi_j^2(X_l)\right)\left(\sum_{j\in G_k}\Big(\sum_{i=1}^{n}\varphi_j(X_i)x_i\Big)^2\right)\\
&= \|\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2\sum_{l=1}^{n}\left(\sum_{j\in G_k}\varphi_j^2(X_l)\right)\\
&= \|\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2\sum_{l=1}^{n}(b_k^l)^2\\
&\leq nb_k^2\|\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2,
\end{aligned}$$

which proves the upper bound of (2.11). For the lower bound, we just observe that for any $i = 1, \ldots, n$, with $\mathbf{e}_i$ the vector whose i-th coordinate is equal to 1 and all others to 0,

$$\begin{aligned}
{b_k^i}^2 &= \|\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2^2\\
&= <\mathbf{A}_{G_k}^T\mathbf{e}_i, \mathbf{A}_{G_k}^T\mathbf{e}_i>\\
&= <\mathbf{e}_i, \mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{e}_i>\\
&\leq \|\mathbf{e}_i\|_2\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2\\
&\leq c_k\|\mathbf{A}_{G_k}^T\mathbf{e}_i\|_2\\
&= c_k b_k^i,
\end{aligned}$$

which obviously entails $b_k \leq c_k$. For the last point, we observe that

$$\|\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2 = \sum_{j\in G_k}K_j^2,$$

where $K_j = \sum_{i=1}^n \varphi_j(X_i)x_i$. By expressing $\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2$ with respect to the K_j's, we obtain:

$$\begin{aligned}
\|\mathbf{A}_{G_k}\mathbf{A}_{G_k}^T\mathbf{x}\|_2^2 &= \sum_{l=1}^{n}\left(\sum_{j\in G_k}\varphi_j(X_l)\sum_{i=1}^{n}\varphi_j(X_i)x_i\right)^2\\
&= \sum_{l=1}^{n}\sum_{j\in G_k}\varphi_j(X_l)\sum_{i=1}^{n}\varphi_j(X_i)x_i\sum_{j'\in G_k}\varphi_{j'}(X_l)\sum_{i'=1}^{n}\varphi_{j'}(X_{i'})x_{i'}\\
&= \sum_{j\in G_k}\sum_{j'\in G_k}\sum_{l=1}^{n}\varphi_j(X_l)\varphi_{j'}(X_l)\sum_{i=1}^{n}\varphi_j(X_i)x_i\sum_{i'=1}^{n}\varphi_{j'}(X_{i'})x_{i'}\\
&= \sum_{j\in G_k}\sum_{j'\in G_k}\sum_{l=1}^{n}\varphi_j(X_l)\varphi_{j'}(X_l)K_jK_{j'}\\
&\leq \frac{1}{2}\sum_{j\in G_k}\sum_{j'\in G_k}\Big|\sum_{l=1}^{n}\varphi_j(X_l)\varphi_{j'}(X_l)\Big|(K_j^2+K_{j'}^2)\\
&= \sum_{j\in G_k}\sum_{j'\in G_k}\Big|\sum_{l=1}^{n}\varphi_j(X_l)\varphi_{j'}(X_l)\Big|K_j^2,
\end{aligned}$$

from which we deduce (2.12). □

7.4 Proof of Theorem 3

For any $\boldsymbol{\beta} \in \mathbb{R}^p$, we have

$$\begin{aligned}
K(f_0, f_{\boldsymbol{\beta}}) &= \sum_{i=1}^{n} f_0(X_i)\big(\log f_0(X_i) - \log f_{\boldsymbol{\beta}}(X_i)\big) + f_{\boldsymbol{\beta}}(X_i) - f_0(X_i)\\
&= \sum_{i=1}^{n} Y_i\big(\log f_0(X_i) - \log f_{\boldsymbol{\beta}}(X_i)\big) + f_{\boldsymbol{\beta}}(X_i) - f_0(X_i)\\
&\qquad + \sum_{i=1}^{n}(f_0(X_i) - Y_i)\big(\log f_0(X_i) - \log f_{\boldsymbol{\beta}}(X_i)\big)\\
&= \log\mathcal{L}(f_0) - \log\mathcal{L}(f_{\boldsymbol{\beta}}) + \sum_{i=1}^{n}(f_0(X_i) - Y_i)\big(\log f_0(X_i) - \log f_{\boldsymbol{\beta}}(X_i)\big).
\end{aligned}$$

Therefore, for all $\boldsymbol{\beta} \in \mathbb{R}^p$,

$$\begin{aligned} K(f_0, \widehat{f}^{gL}) - K(f_0, f_{\boldsymbol{\beta}}) &= l(\boldsymbol{\beta}) - l(\widehat{\boldsymbol{\beta}}^{gL}) + \sum_{i=1}^{n} \big(f_0(X_i) - Y_i\big)\big(\log f_{\boldsymbol{\beta}}(X_i) - \log \widehat{f}^{gL}(X_i)\big) \\ &= l(\boldsymbol{\beta}) - l(\widehat{\boldsymbol{\beta}}^{gL}) + \sum_{i=1}^{n} \big(f_0(X_i) - Y_i\big) \sum_{j=1}^{p} (\beta_j - \widehat{\beta}_j^{gL})\varphi_j(X_i) \\ &= l(\boldsymbol{\beta}) - l(\widehat{\boldsymbol{\beta}}^{gL}) + \sum_{j=1}^{p} (\widehat{\beta}_j^{gL} - \beta_j) \sum_{i=1}^{n} \varphi_j(X_i)(Y_i - f_0(X_i)). \end{aligned}$$

Let us write $\eta_j = \sum_{i=1}^n \varphi_j(X_i)(Y_i - f_0(X_i)) = \mathbf{A}_j^T(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])$. We have

$$K(f_0, \widehat{f}^{gL}) = K(f_0, f_{\boldsymbol{\beta}}) + l(\boldsymbol{\beta}) - l(\widehat{\boldsymbol{\beta}}^{gL}) + (\widehat{\boldsymbol{\beta}}^{gL} - \boldsymbol{\beta})^T \boldsymbol{\eta}. \tag{7.4}$$

By definition of $\widehat{\boldsymbol{\beta}}^{gL}$,

$$-l(\widehat{\boldsymbol{\beta}}^{gL}) + \sum_{k=1}^{K} \lambda_k^g \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL}\|_2 \leq -l(\boldsymbol{\beta}) + \sum_{k=1}^{K} \lambda_k^g \|\boldsymbol{\beta}_{G_k}\|_2.$$

Furthermore, on Ω_g,

$$\begin{aligned} |(\widehat{\boldsymbol{\beta}}^{gL} - \boldsymbol{\beta})^T \boldsymbol{\eta}| &= \Big| \sum_{j=1}^{p} (\widehat{\beta}_j^{gL} - \beta_j)(\mathbf{A}_j^T(\mathbf{Y} - \mathbb{E}\mathbf{Y})) \Big| \\ &\leq \sum_{k=1}^{K} \sum_{j \in G_k} |\widehat{\beta}_j^{gL} - \beta_j| |\mathbf{A}_j^T(\mathbf{Y} - \mathbb{E}\mathbf{Y})| \\ &\leq \sum_{k=1}^{K} \Big(\sum_{j \in G_k} (\widehat{\beta}_j^{gL} - \beta_j)^2 \Big)^{1/2} \Big(\sum_{j \in G_k} (\mathbf{A}_j^T(\mathbf{Y} - \mathbb{E}\mathbf{Y}))^2 \Big)^{1/2} \\ &= \sum_{k=1}^{K} \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 \|\mathbf{A}_{G_k}^T(\mathbf{Y} - \mathbb{E}\mathbf{Y})\|_2 \\ &\leq \sum_{k=1}^{K} \lambda_k^g \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2. \end{aligned} \tag{7.5}$$

Therefore, for all $\boldsymbol{\beta} \in \mathbb{R}^p$,

$$K(f_0, \widehat{f}^{gL}) \leq K(f_0, f_{\boldsymbol{\beta}}) + \sum_{k=1}^{K} \lambda_k^g \Big(\|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 - \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL}\|_2 + \|\boldsymbol{\beta}_{G_k}\|_2 \Big),$$

from which we deduce (3.3). □

7.5 Proof of Theorem 4

To avoid too tedious notations, we denote $\widehat{f}^{gL}$ instead of $\widehat{f}_{\mu,\alpha}^{gL}$ and $\widehat{\boldsymbol{\beta}}^{gL}$ instead of $\widehat{\boldsymbol{\beta}}_{\mu,\alpha}^{gL}$. We start from Equality (7.4) combined with Inequality (7.5). Then, we have that on Ω_g, for any $\boldsymbol{\beta} \in \Gamma(\mu)$,

$$K(f_0, \widehat{f}^{gL}) + (\alpha - 1) \sum_{k=1}^{K} \lambda_k^g \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 \leq K(f_0, f_{\boldsymbol{\beta}}) + \sum_{k=1}^{K} \alpha \lambda_k^g \Big(\|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 - \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL}\|_2 + \|\boldsymbol{\beta}_{G_k}\|_2 \Big).$$

On $J(\boldsymbol{\beta})^c$, $\|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 - \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL}\|_2 + \|\boldsymbol{\beta}_{G_k}\|_2 = 0$ and

$$K(f_0, \widehat{f}^{gL}) + (\alpha - 1) \sum_{k=1}^{K} \lambda_k^g \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 \leq K(f_0, f_{\boldsymbol{\beta}}) + 2\alpha \sum_{k \in J(\boldsymbol{\beta})} \lambda_k^g \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2. \tag{7.6}$$

By applying the Cauchy-Schwarz inequality we also have

$$K(f_0, \widehat{f}^{gL}) + (\alpha - 1) \sum_{k=1}^{K} \lambda_k^g \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2 \leq K(f_0, f_{\boldsymbol{\beta}}) + 2\alpha |J(\boldsymbol{\beta})|^{1/2} \Big(\sum_{k \in J(\boldsymbol{\beta})} (\lambda_k^g)^2 \|\widehat{\boldsymbol{\beta}}_{G_k}^{gL} - \boldsymbol{\beta}_{G_k}\|_2^2 \Big)^{1/2}. \tag{7.7}$$

If we write $\boldsymbol{\Delta} = \mathbf{D}(\widehat{\boldsymbol{\beta}}^{gL} - \boldsymbol{\beta})$, where $\mathbf{D}$ is a diagonal matrix with $D_{j,j} = \lambda_k^g$ if $j \in G_k$, then we can rewrite (7.6) as

$$K(f_0, \widehat{f}^{gL}) + (\alpha - 1)\|\boldsymbol{\Delta}\|_{1,2} \leq K(f_0, f_{\boldsymbol{\beta}}) + 2\alpha \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_{1,2} \tag{7.8}$$

and we deduce from (7.7)

$$K(f_0, \widehat{f}^{gL}) \leq K(f_0, f_{\boldsymbol{\beta}}) + 2\alpha (|J(\boldsymbol{\beta})|)^{1/2} \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_2. \tag{7.9}$$

Now, on the event $\big\{2\alpha \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_{1,2} \leq \varepsilon K(f_0, f_{\boldsymbol{\beta}})\big\}$, the theorem follows immediately from (7.8). We now assume that $\varepsilon K(f_0, f_{\boldsymbol{\beta}}) \leq 2\alpha \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_{1,2}$. Since K is non-negative, we deduce from (7.8) that

$$(\alpha - 1)\|\boldsymbol{\Delta}\|_{1,2} \leq 2\alpha \Big(1 + \frac{1}{\varepsilon}\Big) \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_{1,2},$$

$$(\alpha - 1)\|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})^c}\|_{1,2} \leq \left(2\alpha \Big(1 + \frac{1}{\varepsilon}\Big) - (\alpha - 1) \right) \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_{1,2}$$

and

$$\|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})^c}\|_{1,2} \leq \left(\frac{\alpha + 1 + 2\alpha/\varepsilon}{\alpha - 1} \right) \|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_{1,2}.$$

This is equivalent to

$$\sum_{k\in J(\beta)^c}\lambda_k^g\|(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})_{G_k}\|_2\leq\left(\frac{\alpha+1+2\alpha/\varepsilon}{\alpha-1}\right)\sum_{k\in J(\beta)}\lambda_k^g\|(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})_{G_k}\|_2.$$

From Assumption 2 we have that, if $\boldsymbol{\beta}$ is such that $|J(\boldsymbol{\beta})|\leq s$, then

$$\|(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})_{J(\boldsymbol{\beta})}\|_2\leq\frac{1}{\kappa_n}\big((\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})^T\mathbf{G}(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})\big)^{1/2}.$$

Since

$$\mathbf{G}_{j,j'}=\sum_{i=1}^n\varphi_j(X_i)\varphi_{j'}(X_i)f_0(X_i),$$

by setting

$$u_i=\log f_{\boldsymbol{\beta}}(X_i)-\log f_0(X_i)\quad\text{and}\quad\widehat{u}_i^{gL}=\log f_{\widehat{\boldsymbol{\beta}}^{gL}}(X_i)-\log f_0(X_i),$$

we have

$$\begin{aligned}(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})^T\mathbf{G}(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta}) &= \sum_{j=1}^p\sum_{j'=1}^p(\widehat{\beta}_j^{gL}-\beta_j)(\widehat{\beta}_{j'}^{gL}-\beta_{j'})\mathbf{G}_{j,j'}\\ &= \sum_{i=1}^n f_0(X_i)\Big(\sum_{j=1}^p(\widehat{\beta}_j^{gL}-\beta_j)\varphi_j(X_i)\Big)^2\\ &= \sum_{i=1}^n f_0(X_i)(\widehat{u}_i^{gL}-u_i)^2.\end{aligned}$$

We set $h(f_0,f_{\boldsymbol{\beta}})=\sum_{i=1}^n f_0(X_i)u_i^2$ and $h(f_0,\widehat{f}^{gL})=\sum_{i=1}^n f_0(X_i)(\widehat{u}_i^{gL})^2$. From (7.9) and since

$$\begin{aligned}\|\boldsymbol{\Delta}_{J(\boldsymbol{\beta})}\|_2 &\leq (\max_k\lambda_k^g)\|(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})_{J(\boldsymbol{\beta})}\|_2\\ &\leq \frac{\max_k\lambda_k^g}{\kappa_n}\big((\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})^T\mathbf{G}(\widehat{\boldsymbol{\beta}}^{gL}-\boldsymbol{\beta})\big)^{1/2},\end{aligned}$$

we have

$$K(f_0,\widehat{f}^{gL})\leq K(f_0,f_{\boldsymbol{\beta}})+\frac{2\alpha}{\kappa_n}|J(\boldsymbol{\beta})|^{1/2}(\max_k\lambda_k^g)\Big(\sqrt{h(f_0,\widehat{f}^{gL})}+\sqrt{h(f_0,f_{\boldsymbol{\beta}})}\Big).$$

To conclude, we use arguments similar to Lemler (2013). We recall them for the safe of completeness. To connect $h(f_0,f_{\boldsymbol{\beta}})$ to $K(f_0,f_{\boldsymbol{\beta}})$, we use Lemma 1 of Bach (2010) that is recalled now.

Lemma 2 *Let g be a convex three times differentiable function $g:\mathbb{R}\to\mathbb{R}$ such that for all $t\in\mathbb{R}$, $|g'''(t)|\leq Sg''(t)$ for some $S\geq 0$. Then, for all $t\geq 0$,*

$$\frac{g''(0)}{S^2}\phi(-St)\leq g(t)-g(0)-g'(0)t\leq\frac{g''(0)}{S^2}\phi(St),$$

where $\phi(x)=e^x-x-1$.

Let h be a real function. We set

$$G(h)=\sum_{i=1}^n\big(e^{h(X_i)}-f_0(X_i)h(X_i)\big)$$

and

$$g(t)=G(h+tk),$$

where h and k are functions and $t\in\mathbb{R}$. We have :

$$g'(t)=\sum_{i=1}^n\big(k(X_i)e^{h(X_i)+tk(X_i)}-f_0(X_i)k(X_i)\big),$$

$$g''(t)=\sum_{i=1}^n\big(k^2(X_i)e^{h(X_i)+tk(X_i)}\big)$$

and

$$g'''(t)=\sum_{i=1}^n\big(k^3(X_i)e^{h(X_i)+tk(X_i)}\big).$$

Therefore $|g'''(t)|\leq Sg''(t)$ with $S=\max_i|k(X_i)|$. We choose $h(X_i)=\log f_0(X_i)$ and $k(X_i)=u_i=\log f_{\boldsymbol{\beta}}(X_i)-\log f_0(X_i)$ and we apply Lemma 2 to g with $t=1$. Computations yield that $g(1)-g(0)=K(f_0,f_{\boldsymbol{\beta}})$, $g'(0)=0$ and $g''(0)=\sum_{i=1}^n f_0(X_i)u_i^2=h(f_0,f_{\boldsymbol{\beta}})$. Therefore

$$\frac{\phi(-S)}{S^2}h(f_0,f_{\boldsymbol{\beta}})\leq K(f_0,f_{\boldsymbol{\beta}})\leq\frac{\phi(S)}{S^2}h(f_0,f_{\boldsymbol{\beta}}).$$

Finally, for $\boldsymbol{\beta}\in\Gamma(\mu)$, $S=\max_i|u_i|\leq\mu+m$. Furthermore, $x\longrightarrow\frac{\phi(x)}{x^2}$ is a nonnegative increasing function and therefore we have

$$\mu' h(f_0,f_{\boldsymbol{\beta}})\leq K(f_0,f_{\boldsymbol{\beta}})\leq\mu'' h(f_0,f_{\boldsymbol{\beta}}),$$

where $\mu'=\frac{\phi(-\mu-m)}{(\mu+m)^2}$ and $\mu''=\frac{\phi(\mu+m)}{(\mu+m)^2}$. It follows that, for $\boldsymbol{\beta}\in\Gamma(\mu)$ such that $|J(\boldsymbol{\beta})|\leq s$,

$$K(f_0,\widehat{f}^{gL})\leq K(f_0,f_{\boldsymbol{\beta}})+\frac{2\alpha}{\kappa_n\sqrt{\mu'}}|J(\boldsymbol{\beta})|^{1/2}(\max_k\lambda_k^g)\Big(\sqrt{K(f_0,\widehat{f}^{gL})}+\sqrt{K(f_0,f_{\boldsymbol{\beta}})}\Big).$$

We use twice the inequality $2uv \leq bu^2 + \frac{v^2}{b}$ for any $b > 0$, applied to $u = \frac{\alpha}{\kappa_n}\sqrt{|J(\boldsymbol{\beta})|}(\max_k \lambda_k^g)$ and v being either $\sqrt{\frac{1}{\mu'}K(f_0, \widehat{f}^{gL})}$ or $\sqrt{\frac{1}{\mu'}K(f_0, f_{\boldsymbol{\beta}})}$. We have

$$\Big(1 - \frac{1}{\mu' b}\Big)K(f_0, \widehat{f}^{gL}) \leq \Big(1 + \frac{1}{\mu' b}\Big)K(f_0, f_{\boldsymbol{\beta}}) + 2b\frac{\alpha^2|J(\boldsymbol{\beta})|}{\kappa_n^2}(\max_k \lambda_k^g)^2.$$

Finally,

$$K(f_0, \widehat{f}^{gL}) \leq \Big(\frac{\mu' b + 1}{\mu' b - 1}\Big)K(f_0, f_{\boldsymbol{\beta}}) + 2\frac{\mu' b^2}{\mu' b - 1}\frac{\alpha^2|J(\boldsymbol{\beta})|}{\kappa_n^2}(\max_k \lambda_k^g)^2.$$

We choose $b > 1/\mu'$ such that $\frac{\mu' b+1}{\mu' b-1} = 1 + \varepsilon$ and we set $B(\varepsilon, m, \mu) = 2(1+\varepsilon)^{-1}\frac{\mu' b^2}{\mu' b - 1}$. Finally, we have, for any $\boldsymbol{\beta} \in \Gamma(\mu)$ such that $|J(\boldsymbol{\beta})| \leq s$,

$$K(f_0, \widehat{f}^{gL}) \leq (1+\varepsilon)\left(K(f_0, f_{\boldsymbol{\beta}}) + B(\varepsilon, m, \mu)\frac{\alpha^2|J(\boldsymbol{\beta})|}{\kappa_n^2}(\max_k \lambda_k^g)^2\right).$$

This completes the proof of Theorem 4. □

Acknowledgments

The research of Stéphane Ivanoff and Vincent Rivoirard is partly supported by the french Agence Nationale de la Recherche (ANR 2011 BS01 010 01 projet Calibration) and by the Chaire Economie et Gestion des Nouvelles Données, under the auspices of Institut Louis Bachelier, Havas-Media and Paris-Dauphine. The research of Franck Picard is partly supported by the ABS4NGS ANR project ANR-11-BINF-0001-06. The authors wish to thank two anonymous referees who each made helpful suggestions that improved the presentation of the paper.

References

Anscombe, F. J. (1948). The transformation of Poisson, binomial and negative-binomial data. *Biometrika*, 35:246–254.

Bach, F. (2010). Self-concordant analysis for logistic regression. *Electron. J. Stat.*, 4:384–414.

Bach, F. R. (2008). Consistency of the group lasso and multiple kernel learning. *J. Mach. Learn. Res.*, 9:1179–1225.

Bertin, K., Le Pennec, E., and Rivoirard, V. (2011). Adaptive Dantzig density estimation. *Ann. Inst. Henri Poincaré Probab. Stat.*, 47(1):43–74.

Besbeas, P., De Feis, I., and Sapatinas, T. (2004). A comparative simulation study of wavelet shrinkage estimators for Poisson counts. *Intern. Statist. Review*, 72(2):209–237.

Bickel, P. J., Ritov, Y., and Tsybakov, A. B. (2009). Simultaneous analysis of lasso and Dantzig selector. *Ann. Statist.*, 37(4):1705–1732.

Blazere, M., Loubes, J.-M., and Gamboa, F. (2014). Oracle inequalities for a group lasso procedure applied to generalized linear models in high dimension. *IEEE Transactions on Information Theory*, 4(12):2303–2318.

Bradic, J., Fan, J., and Jiang, J. (2011). Regularization for Cox's proportional hazards model with NP-dimensionality. *Ann. Statist.*, 39(6):3092–3120.

Bühlmann, P. and van de Geer, S. (2011). *Statistics for high-dimensional data*. Springer Series in Statistics. Springer, Heidelberg. Methods, theory and applications.

Bunea, F., Tsybakov, A., and Wegkamp, M. (2007a). Sparsity oracle inequalities for the Lasso. *Electron. J. Stat.*, 1:169–194.

Bunea, F., Tsybakov, A. B., and Wegkamp, M. H. (2007b). Aggregation for Gaussian regression. *Ann. Statist.*, 35(4):1674–1697.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM Rev.*, 43(1):129–159. Reprinted from SIAM J. Sci. Comput. **2**0 (1998), no. 1, 33–61 (electronic) [MR1639094 (99h:94013)].

Chesneau, C. and Hebiri, M. (2008). Some theoretical results on the grouped variables Lasso. *Math. Methods Statist.*, 17(4):317–326.

Chicken, E. and Cai, T. (2005). Block thresholding for density estimation: Local and global adaptivity. *Journal of Multivariate Analysis*, 95:76–106.

Dalalyan, A. S., Hebiri, M., Meziani, K., and Salmon, J. (2013). Learning heteroscedastic models by convex programming under group sparsity. In *ICML*.

Donoho, D. and Johnstone, I. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81:425–455.

Fryzlewicz, P. (2008). Data-driven wavelet-Fisz methodology for nonparametric function estimation. *Electron. J. Stat.*, 2:863–896.

Fryzlewicz, P. and Nason, G. P. (2004). A Haar-Fisz algorithm for Poisson intensity estimation. *J. Comput. Graph. Statist.*, 13(3):621–638.

Furey, T. S. (2012). ChIP-seq and beyond: new and improved methodologies to detect and characterize protein-DNA interactions. *Nat. Rev. Genet.*, 13(12):840–852.

Gaïffas, S. and Guilloux, A. (2012). High-dimensional additive hazards models and the Lasso. *Electron. J. Stat.*, 6:522–546.

Hansen, N., Reynaud-Bouret, P., and Rivoirard, V. (2014). Lasso and probabilistic inequalities for multivariate point processes. *To appear in Bernoulli*.

Houdré, C., Marchal, P., and Reynaud-Bouret, P. (2008). Concentration for norms of infinitely divisible vectors with independent components. *Bernoulli*, 14(4):926–948.

Huang, J. and Zhang, T. (2010). The benefit of group sparsity. *Ann. Statist.*, 38(4):1978–2004.

Jia, J., Rohe, K., and Yu, B. (2013). The lasso under Poisson-like heteroscedasticity. *Statist. Sinica*, 23(1):99–118.

Kingman, J. F. C. (1993). *Poisson processes*, volume 3 of *Oxford Studies in Probability*. The Clarendon Press, Oxford University Press, New York. Oxford Science Publications.

Kolaczyk, E. D. (1999). Wavelet shrinkage estimation of certain Poisson intensity signals using corrected thresholds. *Statist. Sinica*, 9(1):119–135.

Kong, S. and Nan, B. (2014). Non-asymptotic oracle inequalities for the high-dimensional cox regression via lasso. *Statistica Sinica*, 24:2542.

Leblanc, F. and Letué, F. (2006). Maximum likelihood estimation in poisson regression via wavelet model selection. Technical report, hal-00079298.

Lemler, S. (2013). Oracle inequalities for the lasso in the high-dimensional multiplicative Aalen intensity model. *Submitted*.

Lounici, K., Pontil, M., van de Geer, S., and Tsybakov, A. B. (2011). Oracle inequalities and optimal inference under group sparsity. *Ann. Statist.*, 39(4):2164–2204.

Meier, L., van de Geer, S., and Bühlmann, P. (2008). The group Lasso for logistic regression. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 70(1):53–71.

Nardi, Y. and Rinaldo, A. (2008). On the asymptotic properties of the group lasso estimator for linear models. *Electron. J. Stat.*, 2:605–633.

Nason, G. P. (1996). Wavelet shrinkage using cross-validation. *J. Roy. Statist. Soc. Ser. B*, 58(2):463–479.

Obozinski, G., Wainwright, M. J., and Jordan, M. I. (2011). Support union recovery in high-dimensional multivariate regression. *Ann. Statist.*, 39(1):1–47.

Park, M. Y. and Hastie, T. (2007). L_1-regularization path algorithm for generalized linear models. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 69(4):659–677.

Picard, F., Cadoret, J. C., Audit, B., Arneodo, A., Alberti, A., Battail, C., Duret, L., and Prioleau, M. N. (2014). The spatiotemporal program of DNA replication is associated with specific combinations of chromatin marks in human cells. *PLoS Genet.*, 10(5):e1004282.

Reynaud-Bouret, P. (2003). Adaptive estimation of the intensity of inhomogeneous Poisson processes via concentration inequalities. *Probab. Theory Related Fields*, 126(1):103–153.

Sardy, S., Antoniadis, A., and Tseng, P. (2004). Automatic smoothing with wavelets for a wide class of distributions. *J. Comput. Graph. Statist.*, 13(2):399–421.

Thurman, R. E., Rynes, E., Humbert, R., Vierstra, J., Maurano, M. T., Haugen, E., Sheffield, N. C., Stergachis, A. B., Wang, H., Vernot, B., Garg, K., John, S., Sandstrom, R., Bates, D., Boatman, L., Canfield, T. K., Diegel, M., Dunn, D., Ebersol, A. K., Frum, T., Giste, E., Johnson, A. K., Johnson, E. M., Kutyavin, T., Lajoie, B., Lee, B. K., Lee, K., London, D., Lotakis, D., Neph, S., Neri, F., Nguyen, E. D., Qu, H., Reynolds, A. P., Roach, V., Safi, A., Sanchez, M. E., Sanyal, A., Shafer, A., Simon, J. M., Song, L., Vong, S., Weaver, M., Yan, Y., Zhang, Z., Zhang, Z., Lenhard, B., Tewari, M., Dorschner, M. O., Hansen, R. S., Navas, P. A., Stamatoyannopoulos, G., Iyer, V. R., Lieb, J. D., Sunyaev, S. R., Akey, J. M., Sabo, P. J., Kaul, R., Furey, T. S., Dekker, J., Crawford, G. E., and Stamatoyannopoulos, J. A. (2012). The accessible chromatin landscape of the human genome. *Nature*, 489(7414):75–82.

Tropp, J. A. (2004). Greed is good: algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50(10):2231–2242.

van de Geer, S. A. (2008). High-dimensional generalized linear models and the lasso. *Ann. Statist.*, 36(2):614–645.

van de Geer, S. A. and Bühlmann, P. (2009). On the conditions used to prove oracle results for the Lasso. *Electron. J. Stat.*, 3:1360–1392.

Wei, F. and Huang, J. (2010). Consistent group selection in high-dimensional linear regression. *Bernoulli*, 16(4):1369–1384.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 68(1):49–67.

Zou, H. (2006). The adaptive lasso and its oracle properties. *J. Amer. Statist. Assoc.*, 101(476):1418–1429.

Zou, H. (2008). A note on path-based variable selection in the penalized proportional hazards model. *Biometrika*, 95(1):241–247.

Journal of Machine Learning Research 17 (2016) 1-53 Submitted 4/15; Revised 8/15; Published 4/16

Causal Inference through a Witness Protection Program

Ricardo Silva RICARDO@STATS.UCL.AC.UK
Department of Statistical Science and CSML
University College London
London WC1E 6BT, UK

Robin Evans EVANS@STATS.OX.AC.UK
Department of Statistics
University of Oxford
Oxford OX1 3TG, UK

Editor: Kevin Murphy

Abstract

One of the most fundamental problems in causal inference is the estimation of a causal effect when treatment and outcome are confounded. This is difficult in an observational study, because one has no direct evidence that all confounders have been adjusted for. We introduce a novel approach for estimating causal effects that exploits observational conditional independencies to suggest "weak" paths in an unknown causal graph. The widely used faithfulness condition of Spirtes et al. is relaxed to allow for varying degrees of "path cancellations" that imply conditional independencies but do not rule out the existence of confounding causal paths. The output is a posterior distribution over bounds on the average causal effect via a linear programming approach and Bayesian inference. We claim this approach should be used in regular practice as a complement to other tools in observational studies.

Keywords: Causal inference, instrumental variables, Bayesian inference, linear programming

1. Contribution

We provide a new methodology for obtaining bounds on the average causal effect (ACE) of a treatment variable X on an outcome variable Y. We introduce methods for binary models and for linear continuous models. For binary variables, the ACE is defined as

$$E[Y \mid do(X = 1)] - E[Y \mid do(X = 0)] = P(Y = 1 \mid do(X = 1)) - P(Y = 1 \mid do(X = 0)), \quad (1)$$

where $do(\cdot)$ is the operator of Pearl (2000), denoting distributions where a set of variables has been intervened on by an external agent.

In this paper, we assume the reader is familiar with the concept of causal graphs, the basics of the *do* operator, and the basics of causal discovery algorithms such as the PC algorithm of Spirtes et al. (2000). We provide a short summary for context in Section 2.

The ACE is in general not identifiable from observational data. We obtain upper and lower bounds on the ACE by exploiting a set of covariates, which we assume are not affected by X or Y as justified by temporal ordering or other background assumptions. Such covariate sets are often found in real-world problems, and form the basis of many of the

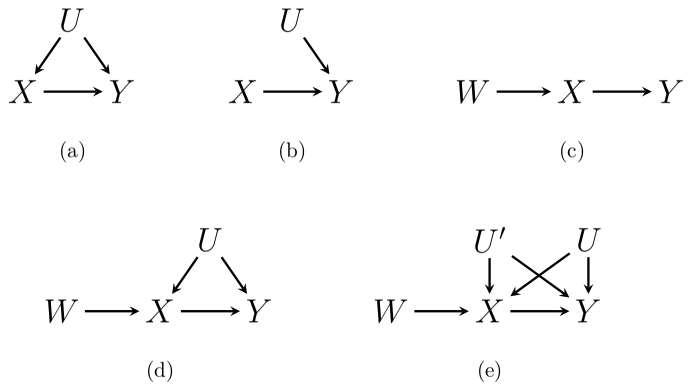


Figure 1: (a) A generic causal graph where X and Y are confounded by some U. (b) The same system in (a) where X is intervened upon by an external agent. (c) A system where W and Y are independent given X. (d) A system where it is possible to use faithfulness to discover that U is sufficient to block all back-door paths between X and Y. (e) Here, U itself is not sufficient.

observational studies done in practice (Rosenbaum, 2002a). However, it is not obvious how to obtain the ACE as a function of the covariates. Our contribution modifies the results of Entner et al. (2013), who exploit conditional independence constraints to obtain point estimates of the ACE but rely on assumptions that might be unstable with finite sample sizes. Our modification provides a different interpretation of their search procedure, which we use to generate candidate *instrumental variables* (Manski, 2007). The linear programming approach of Dawid (2003), inspired by Balke and Pearl (1997) and further refined by Ramsahai (2012), is then modified to generate bounds on the ACE by introducing constraints on some causal paths, motivated as relaxations of Entner et al. (2013). The new setup can be computationally expensive, so we introduce further relaxations to the linear program to generate novel symbolic bounds, and a fast algorithm that sidesteps the full linear programming optimization.

In Section 2, we discuss the background of the problem. In Section 3 we provide an overview of the methodology, which is divided into several subcomponents and described through Sections 4–8. Section 9 contains experiments with synthetic and real data.

2. Background: Instrumental Variables, Witnesses and Admissible Sets

Assuming X is a potential cause of Y, but not the opposite, a cartoon of the possibly complex real-world causal system containing X and Y is shown in Figure 1(a). U represents the universe of common causes of X and Y. In control and policy-making problems, we

would like to know what will happen to the system when the distribution of X is overridden by some external agent (e.g., a doctor, a robot or an economist). The resulting modified system is depicted in Figure 1(b), and represents the family of distributions indexed by $do(X = x)$: the graph in (a) has undergone a "surgery" that removes incoming edges to X. Spirtes et al. (2000) provide an account of the first graphical methods exploiting this idea, which are related to the overriding of structural equations proposed by Haavelmo (1943). Notice that if U is observed in the data set, then we can obtain the distribution $P(Y = y \,|\, do(X = x))$ by simply calculating $\sum_u P(Y = y \,|\, X = x, U = u)P(U = u)$ (Spirtes et al., 2000). This was popularized by Pearl (2000) as the *back-door adjustment*. In general $P(Y = y \,|\, do(X = x))$ can be substantially different from $P(Y = y \,|\, X = x)$.

The ACE can usually be estimated via a trial in which X is randomized: this is equivalent to estimating the conditional distribution of Y given X under data generated as in Figure 1(b). In contrast, in an *observational study* (Rosenbaum, 2002a) we obtain data generated by the system in Figure 1(a). If one believes all relevant confounders U have been recorded in the data then the back-door adjustment can be used, though such completeness is uncommon. By postulating knowledge of the causal graph relating components of U, one can infer whether a measured subset of the causes of X and Y is enough (Pearl, 2000; VanderWeele and Shpitser, 2011; Pearl, 2009). Without knowledge of the causal graph, assumptions such as *faithfulness* (Spirtes et al., 2000) are used to infer it.

The faithfulness assumption states that a conditional independence constraint in the observed distribution exists if and only if a corresponding structural independence exists in the underlying causal graph. For instance, observing the independence $W \perp\!\!\!\perp Y \,|\, X$, and assuming faithfulness and the causal order, we can infer the causal graph Figure 1(c); in all the other graphs this conditional independence in not implied. We deduce that no unmeasured confounders between X and Y exist. This simple procedure for identifying chains $W \to X \to Y$ is useful in exploratory data analysis (Chen et al., 2007; Cooper, 1997), where a large number of possible causal relations $X \to Y$ are unquantified but can be screened off using observational data before experiments are performed. The purpose of using faithfulness is to be able to sometimes identify such quantities.

Entner et al. (2013) generalize the discovery of chain models to situations where a non-empty set of covariates is necessary to block all back-doors. Suppose $\mathbf{W}$ is a set of covariates which are known not to be effects of either X or Y, and we want to find an *admissible set* contained in $\mathbf{W}$: a set of observed variables which we can use for back-door adjustment to obtain $P(Y = y \,|\, do(X = x))$. Entner et al.'s "Rule 1" states the following:

Rule 1: *If there exists a variable $W \in \mathbf{W}$ and a set $\mathbf{Z} \subseteq \mathbf{W} \backslash \{W\}$ such that*

$$(i) \quad W \not\perp\!\!\!\perp Y \,|\, \mathbf{Z} \qquad\qquad (ii) \quad W \perp\!\!\!\perp Y \,|\, \mathbf{Z} \cup \{X\},$$

then infer that $\mathbf{Z}$ is an admissible set.

Entner et al. (2013) also provide ways of identifying zero effects with a "Rule 2." For simplicity of presentation, for now we assume that the effect of interest was already identified as non-zero. Section 7 discusses the case of zero effects.

A point estimate of the ACE can then be found using $\mathbf{Z}$. Given that $(W, \mathbf{Z})$ satisfies Rule 1, we call W a *witness* for the admissible set $\mathbf{Z}$. The model in Figure 1(c) can be

identified with Rule 1, where W is the witness and $\mathbf{Z} = \emptyset$. In this case, for binary models a so-called "naïve" functional $P(Y = 1 \,|\, X = 1) - P(Y = 1 \,|\, X = 0)$ will provide the correct ACE. If U is observable in Figure 1(d), then it can be identified as an admissible set for witness W. Notice that in Figure 1(a), taking U as a scalar, it is not possible to find a witness since there are no remaining variables. Also, if in Figure 1(e) our covariate set $\mathbf{W}$ is $\{W, U\}$, then no witness can be found since U' cannot be blocked. Hence, it is possible for a procedure based on Rule 1 to answer "I don't know," even when a back-door adjustment would be possible *if* one knew the causal graph. However, using the faithfulness assumption alone one cannot do better: Rule 1 is complete for non-zero effects if no further information is available (Entner et al., 2013).

Despite its appeal, the faithfulness assumption is not without difficulties. Even if unfaithful distributions can be ruled out as pathological under seemingly reasonable conditions (Meek, 1995), distributions which lie close to (but not on) an unfaithful model may in practice be indistinguishable from distributions within that unfaithful model at finite sample sizes.

To appreciate these complications, consider the structure in Figure 1(d) with U unobservable and the remaining (observable) variables binary. Here W is randomized but X is not, and we would like to know the ACE of X on Y[1]. W is sometimes known as an *instrumental variable* (IV), and we call Figure 1(d) the *standard IV structure* (SIV); the distinctive features here being the constraints $W \perp\!\!\!\perp U$ and $W \perp\!\!\!\perp Y \mid \{X, U\}$, statements which include latent variables. If this structure is known, optimal bounds

$$\mathcal{L}_{SIV} \leq E[Y \mid do(X = 1)] - E[Y \mid do(X = 0)] \leq \mathcal{U}_{SIV}$$

can be obtained without further assumptions, and estimated using only observational data over the binary variables W, X and Y (Balke and Pearl, 1997). This structure cannot be found using faithfulness, as the only independence constraints involve a latent variable. However, there exist distributions faithful to the IV structure but which at finite sample sizes may appear to satisfy the Markov property for the structure $W \to X \to Y$; in practice this can occur at any finite sample size (Robins et al., 2003). The true average causal effect may lie anywhere in the interval $[\mathcal{L}_{SIV}, \mathcal{U}_{SIV}]$, which can be rather wide even when $W \perp\!\!\!\perp Y \,|\, X$, as shown by the following result:

Proposition 1 *If $W \perp\!\!\!\perp Y \,|\, X$ and the model follows the causal structure of the standard IV graph, then $\mathcal{U}_{SIV} - \mathcal{L}_{SIV} = 1 - |P(X = 1 \,|W = 1) - P(X = 1 \,|\, W = 0)|$.*

All proofs in this manuscript are given in Appendix A. For a fixed joint distribution $P(W, X, Y)$, the length of such an interval cannot be further minimized (Balke and Pearl, 1997). Notice that the length of the interval will depend on how strongly associated W and X are: $W = X$ implies $\mathcal{U}_{IV} - \mathcal{L}_{IV} = 0$ as expected, since this is the scenario of a perfect intervention. The scenario where $W \perp\!\!\!\perp X$ is analogous to not having any instrumental variable, and the length of the corresponding interval is 1.

1. A classical example is in non-compliance: suppose W is the assignment of a patient to either drug or placebo, X is whether the patient actually took the medicine or not, and Y is a measure of health status. The doctor controls W but not X. This problem is discussed by Pearl (2000) and Dawid (2003).

Thus, the true ACE may differ considerably from the "naïve" functional supported by Enter et al.'s Rule 1, appropriate for the simpler structure $W \to X \to Y$ but not for the standard IV structure. While we emphasize that this is a worst-case scenario analysis and by itself should not rule out faithfulness as a useful assumption, it is desirable to provide a method that gives greater control over violations of faithfulness.

3. Outline

In **Section 4**, we introduce the main algorithm, the *Witness Protection Program*. The core idea is (i) to *invert the usage of Entner et al.'s Rule 1*, so that pairs $(W, \mathbf{Z})$ should provide an instrumental variable bounding method instead of a back-door adjustment; (ii) express violations of faithfulness as *bounded violations of local independence*; (iii) find bounds on the ACE using *a linear programming formulation*. Unless stated otherwise, it is assumed that all observed variables are binary.

A simplified version of the algorithm is shown in Algorithm 1. This version assumes we know the distribution of the observed variables, $P(\mathbf{W}, X, Y)$, which simplifies the exposition of the method. The loops in Steps 2 and 3 are a search for pairs $(W, \mathbf{Z})$ of witness-admissible sets that satisfy Enter et al.'s Rule 1, done by verifying independence constraints in the given joint distribution. If we assumed faithfulness, the job would be complete: we either obtain an empty set or the true ACE. This is essentially the contribution of Entner et al. (2013).

However, we assume that faithfulness need not hold, and that all variables can be connected to each other in the causal graph, including a set U of hidden common causes of X and Y. At the same time, we cannot allow for arbitrary violations of faithfulness, as the presence of hidden common causes leads to only very weak constraints on the ACE. Instead, we allow for the expression of a subset of possible violations, expressed as "weak edges" on the fully connected causal graph of $W, \mathbf{Z}, X, Y$ and U. The meaning of a "weak edge" is given in detail in Section 4, and it is fully defined by a set of hyperparameters $\aleph$ that needs to be provided to the algorithm, also explained in Section 4. Given $\aleph$, a generalization of the linear programming problem for instrumental variables described by Ramsahai (2012) can be used to find tight lower and upper bounds on the ACE. As the approach provides each witness a degree of protection against faithfulness violations, using a linear program, we call this framework the *Witness Protection Program* (WPP).

Thus, *this procedure unifies back-door adjustments and (a generalization of) instrumental variable approaches in a single framework, while not requiring knowing the true causal graph and relying on assumptions weaker than faithfulness.* This is the main message of the paper.

The output of the algorithm provides a set of lower/upper bounds on the ACE. If one could assume that $\aleph$ is conservative (that is, the actual edges are "weaker" than the ones implied by the set of causal models $(W, X, Y, \mathbf{Z}, U)$ compatible with $\aleph$), then a tight interval containing the ACE will be given by the largest lower bound and the smallest upper bound. However, there are several practical issues that need to be solved, the main ones being: (i) we do not know $P(\mathbf{W}, X, Y)$ and hence it needs to estimated from data; (ii) once statistical errors are introduced, it is not clear how to combine the different constraints implied by the algorithm; (iii) the computational cost of the procedure can be high, particularly if

input : A distribution $P(\mathbf{W}, X, Y)$;
A set of relaxation parameters $\aleph$;
Covariate index set $\mathbf{W}$ and cause-effect indices X and Y;
output: A set of quadruplets $(W, \mathbf{Z}, \mathcal{L}_{W\mathbf{Z}}, \mathcal{U}_{W\mathbf{Z}})$, where $(W, \mathbf{Z})$ is a witness-admissible set pair and $(\mathcal{L}_{W\mathbf{Z}}, \mathcal{U}_{W\mathbf{Z}})$ are a lower and upper bound on the ACE, respectively;

1 $\mathcal{R} \leftarrow \emptyset$;
2 **for** *each* $W \in \mathbf{W}$ **do**
3 **for** *every admissible set* $\mathbf{Z} \subseteq \mathbf{W} \backslash \{W\}$ *identified by* W **do**
4 $(\mathcal{L}_{W\mathbf{Z}}, \mathcal{U}_{W\mathbf{Z}}) \leftarrow$ bounds on the ACE as given by $P(W, X, Y, \mathbf{Z})$ and $\aleph$;
5 $\mathcal{R} \leftarrow \mathcal{R} \cup \{(W, \mathbf{Z}, \mathcal{L}_{W\mathbf{Z}}, \mathcal{U}_{W\mathbf{Z}})\}$;
6 **end**
7 **end**
8 **return** $\mathcal{R}$

Algorithm 1: A simplified Witness Protection Program algorithm, assuming the observable distribution $P(\mathbf{W}, X, Y)$ is known.

uncertainty estimates are required; (iv) we would like to have some results for continuous data; (v) the set of hyperparameters $\aleph$ needs to be chosen somehow, and some objective criterion to choose them is important in practice.

Section 4.2 addresses points (i) and (ii) using a Bayesian approach. This requires a likelihood function. Since the latent variable model that includes U is not identifiable, we work directly on the marginal observable distribution under the constraints implied by the linear program. Independence constraints can be tested using Bayesian model selection, but optionally can be ignored in the linear programming step to provide a more stringent test of feasibility of $\aleph$, as the feasible region for the ACE might be empty if the tested independence does not fit the data well enough even if it passes the test. An interpretation of this usage of independence tests is given in **Section 4.3**. We criticize naïve uses of Bayesian inference for latent variable models in **Section 4.4**.

A convenient implication of using Bayesian inference is that credible intervals for the ACE bounds can be computed in a conceptually simple way, using Monte Carlo methods. However, numerically running a linear program for each sample is expensive. A fully analytical solution to the linear program is not known, but a solution to a relaxed version of it can be found in a much cheaper and more numerically stable iterative algorithm (compared to a black-box solver) given in **Section 5**. This addresses point (iii), but bounds are looser than those obtained with a numerical solver as a consequence.

Point (iv) is partially addressed by **Section 6**, where we derive bounding methods for linear models. This complements Entner et al. (2012), which relies on non-Gaussianity and faithfulness using conditions weaker than Rule 1. Conceptually the method can be adapted to discrete non-binary data without major modifications, although presentation gets considerably more complicated. Treating continuous $\mathbf{W}$ is not a theoretical problem (at least by discretizing each W on demand while keeping $\mathbf{Z}$ continuous), although different estimation techniques and parametric assumptions would be required. Likewise, it is theoretically

possible to get bounds on the cumulative distribution function of Y by dichotomizing it at different levels $Y \leq y$, but we will not further discuss these generalizations in this paper.

Section 7 is a final note concerning the main procedure, where we discuss the possibility of exploiting Enter et al.'s Rule 2 for detecting zero effects. Although we do not further analyze this modification in our experiments, this section provides further insights on how WPP is related to sensitivity analysis methods for observational studies previously found in the literature.

Finally, **Section 8** is an extensive discussion on point (v), the choice of $\aleph$ and the need to deal with possibly incoherent bounds, which also relates back to point (ii). While this discussion is orthogonal to the main algorithm, which takes $\aleph$ as a given and it is guaranteed to be at least as conservative as Entner et al. (2013), it is a important practical issue. This section also complements the discussion started in **Section 7** on the relation between WPP and sensitivity analysis methods.

4. The Witness Protection Program

Let $(W, \mathbf{Z})$ be any pair found by a search procedure that decides when Rule 1 holds. W will play the role of an instrumental variable, instead of being discarded. Conditional on $\mathbf{Z}$, the lack of an edge $W \to Y$ can be justified by faithfulness (as $W \perp\!\!\!\perp Y \,|\, \{X, \mathbf{Z}\}$). For the same reason, there should not be any (conditional) dependence between W and a possible unmeasured common parent[2] U of X and Y. Hence, $W \perp\!\!\!\perp U$ and $W \perp\!\!\!\perp Y \mid \{U, X\}$ hold given $\mathbf{Z}$. A standard IV bounding procedure such as (Balke and Pearl, 1997) can then be used conditional on each individual value $\mathbf{z}$ of $\mathbf{Z}$, then averaged over $P(\mathbf{Z})$. That is, we can independently obtain lower and upper bounds $\{\mathcal{L}(\mathbf{z}), \mathcal{U}(\mathbf{z})\}$ for each value $\mathbf{z}$, and bound the ACE by

$$\sum_{\mathbf{z}} \mathcal{L}(\mathbf{z}) P(\mathbf{Z} = \mathbf{z}) \leq E[Y \mid do(X = 1)] - E[Y \mid do(X = 0)] \leq \sum_{\mathbf{z}} \mathcal{U}(\mathbf{z}) P(\mathbf{Z} = \mathbf{z}), \qquad (2)$$

since $E[Y \mid do(X = 1)] - E[Y \mid do(X = 0)] = \sum_{\mathbf{z}} (E[Y \mid do(X = 1), \mathbf{Z} = \mathbf{z}] - E[Y \mid do(X = 0), \mathbf{Z} = \mathbf{z}]) P(\mathbf{Z} = \mathbf{z})$.

Under the assumption of faithfulness and the satisfiability of Rule 1, the calculation of the above interval is redundant, as Rule 1 allows the direct use of the back-door adjustment using $\mathbf{Z}$. Our goal is to not enforce faithfulness, but use Rule 1 as a motivation to allow for a subset of violations of faithfulness, but not arbitrary violations.

In what follows, assume $\mathbf{Z}$ is set to a particular value $\mathbf{z}$ and all references to distributions are implicitly assumed to be defined conditioned on the event $\mathbf{Z} = \mathbf{z}$. That is, for simplicity of notation, we will neither represent nor condition on $\mathbf{Z}$ explicitly. The causal ordering where X and Y cannot precede any other variable is also assumed, as well as the causal ordering between X and Y.

Consider a standard parameterization of a directed acyclic graph (DAG) model, not necessarily causal, in terms of conditional probability tables (CPTs): let $\theta^V_{v,\mathbf{p}}$ represent $P(V = v \mid Par(V) = \mathbf{p})$ where $V \in \{W, X, Y, U\}$ denotes both a random variable and a vertex in the corresponding DAG; $Par(V)$ is the corresponding set of parents of V.

2. In this manuscript, we will sometimes refer to U as a *set* of common parents, although we do not change our notation to bold face to reflect that.

Faithfulness violations occur when independence constraints among observables are not *structural*, but due to "path cancellations." This means that parameter values are arranged so that $W \perp\!\!\!\perp Y \mid X$ holds, but paths connecting W and U, or W and Y, may exist so that either $W \not\perp\!\!\!\perp U$ or $W \not\perp\!\!\!\perp Y \mid \{U, X\}$. In this situation, some combination of the following should hold true:

$$\begin{aligned} P(Y = y \mid X = x, W = w, U = u) &\neq P(Y = y \mid X = x, U = u) \\ P(Y = y \mid X = x, W = w, U = u) &\neq P(Y = y \mid X = x, W = w) \\ P(X = x \mid W = w, U = u) &\neq P(X = x \mid W = w) \\ P(U = u \mid W = w) &\neq P(U = u), \end{aligned} \qquad (3)$$

for some $\{w, x, y, u\}$ in the sample space of $P(W, X, Y, U)$.

For instance, if the second and third statements above are true, this implies the existence of an active path into X and Y via U, conditional on W[3], such as $X \leftarrow U \to Y$. If the first statement is true, this corresponds to an active path between W and Y that is not blocked by $\{X, U\}$. If the fourth statement is true, U and W are marginally dependent, with a corresponding active path. Notice that some combinations are still compatible with a model where $W \perp\!\!\!\perp U$ and $W \perp\!\!\!\perp Y \mid \{U, X\}$ hold: if the second statement in (3) is false, this does not mean that U is necessarily a common parent of X and Y. Such a family of models is observationally equivalent[4] to one where U is independent of all variables.

When translating the conditions (3) into parameters $\{\theta^V_{v,\mathbf{p}}\}$, we need to define parent sets for each vertex, which only need to respect the partial causal ordering being assumed; similarly to the Prediction Algorithm of Spirtes et al. (2000), we do not need to fully specify a causal model in order to identify some of its interventional distributions. In our conditional probability table (CPT) factorization, we define $Par(X) = \{W, U\}$ and $Par(Y) = \{W, X, U\}$. The joint distribution of $\{W, U\}$ can be factorized arbitrarily: the causal directionality among W, U (and $\mathbf{Z}$) is not relevant to the only interventional distribution of interest, $do(X)$. In the next subsection, we refine the parameterization of our model by introducing *redundancies*: we provide a parameterization for the latent variable model $P(W, X, Y, U)$, the interventional distribution $P(W, Y, U \mid do(X))$ and the corresponding (latent-free) marginals $P(W, X, Y)$, $P(W, Y \mid do(X))$. These parameters cannot vary fully independently of each other. It is this fact that will allow us to bound the ACE using only $P(W, X, Y)$.

4.1 Encoding Faithfulness Relaxations with Linear Constraints

We define a *relaxation of faithfulness* as any set of assumptions that allows the relations in (3) to be true, but not necessarily in an *arbitrary* way: this means that while the left-hand and right-hand sides of each entry of (3) are indeed different, their difference is bounded by either the absolute difference or by ratios. Without such restrictions, (3) will only imply uninteresting bounds, as discussed in our presentation of Proposition 1.

Consider the following parameterization of the distribution of $\{W, X, Y, U\}$ under the observational and interventional regimes, and their respective marginals obtained by in-

3. That is, a path that d-connects X and Y and includes U, conditional on W; it is "into" X (and Y) because the edge linking X to the path points to X. See Spirtes et al. (2000) and Pearl (2000) for formal definitions and more examples.
4. Meaning a family of models where $P(W, X, Y)$ satisfies the same constraints.

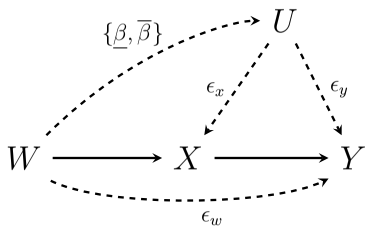


Figure 2: A visual depiction of the family of assumptions introduced in our framework. Dashed edges correspond to conditional dependencies that are constrained according to free parameters, displayed along each corresponding edge. This is motivated by observing $W \perp\!\!\!\perp Y \mid X$.

tegrating U away[5]. Again we condition everywhere on a particular value $\mathbf{z}$ of $\mathbf{Z}$ but, for simplicity of presentation, we suppress this from our notation, since it is not crucial to the developments in this section:

$$\begin{aligned}
\zeta^\star_{yx.w} &\equiv P(Y=y, X=x \,|\, W=w, U) \\
\zeta_{yx.w} &\equiv \textstyle\sum_U P(Y=y, X=x \,|\, W=w, U) P(U \,|\, W=w) \\
&= P(Y=y, X=x \,|\, W=w) \\
\\
\eta^\star_{xw} &\equiv P(Y=1 \,|\, X=x, W=w, U) \\
\eta_{xw} &\equiv \textstyle\sum_U P(Y=1 \,|\, X=x, W=w, U) P(U \,|\, W=w) \\
&= P(Y=1 \,|\, do(X=x), W=w) \\
\\
\delta^\star_w &\equiv P(X=1 \,|\, W=w, U) \\
\delta_w &\equiv \textstyle\sum_U P(X=x \,|\, W=w, U) P(U \,|\, W=w) \\
&= P(X=1 \,|\, W=w).
\end{aligned}$$

Under this encoding, the ACE is given by

$$\eta_{11} P(W=1) + \eta_{10} P(W=0) - \eta_{01} P(W=1) - \eta_{00} P(W=0). \tag{4}$$

Notice that we do not explicitly parameterize the marginal of U, for reasons that will become clear later.

We introduce the following assumptions, as illustrated by Figure 2:

$$|\eta^\star_{x1} - \eta^\star_{x0}| \leq \epsilon_w \tag{5}$$
$$|\eta^\star_{xw} - P(Y=1 \,|\, X=x, W=w)| \leq \epsilon_y \tag{6}$$
$$|\delta^\star_w - P(X=1 \,|\, W=w)| \leq \epsilon_x \tag{7}$$
$$\underline{\beta} P(U) \leq P(U \,|\, W=w) \leq \overline{\beta} P(U). \tag{8}$$

5. Notice from the development in this section that U is not necessarily a scalar, nor discrete.

Setting $\epsilon_w = 0$, $\underline{\beta} = \overline{\beta} = 1$ recovers the standard IV structure. Further assuming $\epsilon_y = \epsilon_x = 0$ recovers the chain structure $W \to X \to Y$. Under this parameterization in the case $\epsilon_y = \epsilon_x = 1$, $\underline{\beta} = \overline{\beta} = 1$, Ramsahai (2012), extending Dawid (2003), used linear programming to obtain bounds on the ACE. We will briefly describe the four main steps of the framework of Dawid (2003), and refer to the cited papers for more details of their implementation.

For now, assume that $\zeta_{yx.w}$ and $P(W=w)$ are known constants—that is, treat $P(W, X, Y)$ as known. This assumption will be dropped later. Dawid's formulation of a bounding procedure for the ACE is as follows.

Step 1 *Notice that parameters $\{\eta^\star_{xw}\}$ take values in a 4-dimensional polytope. Find the extreme points of this polytope. Do the same for $\{\delta^\star_w\}$.*

In particular, for $\epsilon_w = \epsilon_y = 1$, the polytope of feasible values for the four dimensional vector $(\eta^\star_{00}, \eta^\star_{01}, \eta^\star_{10}, \eta^\star_{11})$ is the unit hypercube $[0,1]^4$, a polytope with a total of 16 vertices $(0,0,0,0), (0,0,0,1), \ldots (1,1,1,1)$. Dawid (2003) covered the case $\epsilon_w = 0$, where a two-dimensional vector $\{\eta^\star_x\}$ replaces $\{\eta^\star_{xw}\}$. In Ramsahai (2012), the case $0 \leq \epsilon_w < 1$ is also covered: some of the corners in $[0,1]^4$ disappear and are replaced by others. The case where $\epsilon_w = \epsilon_x = \epsilon_y = 1$ is vacuous, in the sense that the consecutive steps cannot infer non-trivial constraints on the ACE.

Step 2 *Find the extreme points of the joint space $\{\zeta^\star_{yx.w}\} \times \{\eta^\star_{xw}\}$ by mapping them from the extreme points of $\{\delta^\star_w\} \times \{\eta^\star_{xw}\}$, since $\zeta^\star_{yx.w} = (\delta^\star_w)^x (1-\delta^\star_w)^{(1-x)} \eta^\star_{xw}$.*

The extreme points of the joint space $\{\delta^\star_w\} \times \{\eta^\star_{xw}\}$ are just the combination of the extreme points of each space. Some combinations $\delta^\star_w \times \eta^\star_{xw}$ map to the same $\zeta^\star_{yx.w}$, while the mapping from a given $\delta^\star_w \times \eta^\star_{xw}$ to $\eta^\star_{xw}$ is just the trivial projection. At this stage, we obtain all the extreme points of the polytope $\{\zeta^\star_{yx.w}\} \times \{\eta^\star_{xw}\}$ that are entailed by the factorization of $P(W, X, Y, U)$ and our constraints.

Step 3 *Using the extreme points of the joint space $\{\zeta^\star_{yx.w}\} \times \{\eta^\star_{xw}\}$, find the dual polytope of this space in terms of linear inequalities. Points in this polytope are convex combinations of $\{\zeta^\star_{yx.w}\} \times \{\eta^\star_{xw}\}$, shown by Dawid (2003) to correspond to the marginalizations over some $P(U)$, and each $P(U)$ corresponds to some point in the polytope. This results in constraints over $\{\zeta_{yx.w}\} \times \{\eta_{xw}\}$.*

This is the core step in Dawid (2003): points in the polytope $\{\zeta_{yx.w}\} \times \{\eta_{xw}\}$ correspond to different marginalizations of U according to different $P(U)$. Describing the polytope in terms of inequalities provides all feasible distributions that result from marginalizing U according to some $P(U)$. Because we included both $\zeta^\star_{yx.w}$ and $\eta^\star_{xw}$ in the same space, this will tie together $P(Y, X \mid W)$ and $P(Y \mid do(X), W)$.

Step 4 *Finally, maximize/minimize (4) with respect to $\{\eta_{xw}\}$ subject to the constraints found in Step 3 to obtain upper/lower bounds on the ACE.*

Allowing for the case where $\epsilon_x < 1$ or $\epsilon_y < 1$ is just a matter of changing the first step, where box constraints are set on each individual parameter as a function of the known $P(Y = y, X = x \,|\, W = w)$, prior to the mapping in Step 2. The resulting constraints are now implicitly non-linear in $P(Y = y, X = x \,|\, W = w)$, but at this stage this does not matter as the distribution of the observables is treated as a constant. That is, each resulting constraint in Step 3 is a linear function of $\{\eta_{xw}\}$ and a multilinear function on $\{\{\zeta_{yx.w}\}, \epsilon_x, \epsilon_y, \epsilon_w, \overline{\beta}, \underline{\beta}, P(W)\}$, as discussed in Section 5. Within the objective function (4), the only decision variables are $\{\eta_{xw}\}$, and hence Step 4 still sets up a linear programming problem even if there are multiplicative interactions between $\{\zeta_{yx.w}\}$ and other parameters.

To allow for the case $\underline{\beta} < 1 < \overline{\beta}$, we substitute every occurrence of $\zeta_{yx.w}$ due to the dualization in Step 3 above[6] by $\kappa_{yx.w} \equiv \sum_U \zeta^\star_{yx.w} P(U)$; notice the difference between $\kappa_{yx.w}$ and $\zeta_{yx.w}$. Likewise, we substitute every occurrence of η_{xw} in the constraints by $\omega_{xw} \equiv \sum_U \eta^\star_{xw} P(U)$. Instead of plugging in constants for the values of $\kappa_{yx.w}$ and turning the crank of a linear programming solver, we treat $\{\kappa_{yx.w}\}$ (and $\{\omega_{xw}\}$) as unknowns, linking them to observables and η_{xw} by the constraints

$$\begin{array}{rcl} \eta_{xw}/\overline{\beta} \leq & \omega_{xw} & \leq \min(1, \eta_{xw}/\underline{\beta}), \\ \zeta_{yx.w}/\overline{\beta} \leq & \kappa_{yx.w} & \leq \zeta_{yx.w}/\underline{\beta}, \end{array} \tag{9}$$

$$\sum_{yx} \kappa_{yx.w} = 1. \tag{10}$$

Finally, the steps requiring finding extreme points and converting between representations of a polytope can be easily implemented using a package such as Polymake[7] or the SCDD package[8] for R. Once bounds are obtained for each particular value of $\mathbf{Z}$, Equation (2) is used to obtain the unconditional bounds assuming $P(\mathbf{Z})$ is known.

In Section 8, we provide some guidance on how to choose the free parameters of the relaxation. However, it is relevant to point out that any choice of $\epsilon_w \geq 0, \epsilon_y \geq 0, \epsilon_x \geq 0, 0 \leq \underline{\beta} \leq 1 \leq \overline{\beta}$ is *guaranteed to provide bounds that are at least as conservative* as the back-door adjusted point estimator of Entner et al. (2013), which is always covered by the bounds. Background knowledge, after a user is suggested a witness and admissible set, can also be used to set relaxation parameters.

So far, the linear programming formulated through Steps 1–4 assumes one has already identified an appropriate witness W and admissible set $\mathbf{Z}$, and that the joint distribution $P(W, X, Y, \mathbf{Z})$ is known. In the next section, we discuss how this procedure is integrated with statistical inference for $P(W, X, Y, \mathbf{Z})$ and the search procedure of Entner et al. (2013).

6. Notice the subtlety: the values of $P(y, x \mid w)$ appear within the extreme points of $\{\zeta^\star_{yx.w}\} \times \{\eta^\star_{xw}\}$, but here we are only concerned about the symbols $\zeta_{yx.w}$ emerging from convex combinations of $\zeta^\star_{yx.w}$. In the original formulation of Dawid (2003), $\kappa_{yx.w} = P(y, x \mid w)$ is satisfied, because $P(U) = P(U \mid W)$ is assumed, but in our case in general this will not be true. Hence, the need for a different symbol. Ramsahai (2012) deals with the $P(U) \neq P(U \mid W)$ relaxation in a different way by conditioning on each value of W, but his ACE intervals always include zero.
7. `http://www.poymake.org`
8. `http://cran.r-project.org/`

input : A binary data matrix $\mathcal{D}$;
A set of relaxation parameters $\aleph$;
A covariate index set $\mathbf{W}$ and cause-effect indices X and Y;
output: A set of triplets $(W, \mathbf{Z}, \mathcal{B})$, where $(W, \mathbf{Z})$ is a witness-admissible set pair contained in $\mathbf{W}$ and $\mathcal{B}$ is a distribution over lower/upper bounds on the ACE implied by the pair

$\mathcal{R} \leftarrow \emptyset$;
for *each* $W \in \mathbf{W}$ **do**
  for *every admissible set* $\mathbf{Z} \subseteq \mathbf{W} \backslash \{W\}$ *identified by* W *given* $\mathcal{D}$ **do**
    $\mathcal{B} \leftarrow$ posterior over lower/upper bounds on the ACE as given by $(W, \mathbf{Z}, X, Y, \mathcal{D}, \aleph)$;
    if *there is no evidence in* $\mathcal{B}$ *to falsify the* $(W, \mathbf{Z}, \aleph)$ *model* **then**
      $\mathcal{R} \leftarrow \mathcal{R} \cup \{(W, \mathbf{Z}, \mathcal{B})\}$;
    end
  end
end
return $\mathcal{R}$

Algorithm 2: The outline of the Witness Protection Program algorithm.

4.2 Bayesian Learning and Result Summarization

In the previous section, we treated (the conditional) $\zeta_{yx.w}$ and $P(W = w)$ as known. A common practice is to replace them by plug-in estimators (and in the case of a non-empty admissible set $\mathbf{Z}$, an estimate of $P(\mathbf{Z})$ is also necessary). Such models can also be falsified, as the constraints generated are typically only supported by a strict subset of the probability simplex. In principle, one could fit parameters without constraints, and test the model by a direct check of satisfiability of the inequalities using the plug-in values. However, this does not take into account the uncertainty in the estimation. For the standard IV model, Ramsahai and Lauritzen (2011) discuss a proper way of testing such models in a frequentist sense.

Our models can be considerably more complicated. Recall that constraints will depend on the extreme points of the $\{\zeta^\star_{yx.w}\}$ parameters. As implied by (6) and (7), extreme points will be functions of $\zeta_{yx.w}$. Writing the constraints fully in terms of the observed distribution will reveal non-linear relationships. We approach the problem in a Bayesian way. We will assume first the dimensionality of $\mathbf{Z}$ is modest (say, 10 or less), as this is the case in most applications of faithfulness to causal discovery. We parameterize $\zeta^{\mathbf{z}}_{yxw} \equiv P(Y = y, X = x, W = w \mid \mathbf{Z} = \mathbf{z})$ as a full $2 \times 2 \times 2$ contingency table[9]. In the context of the linear programming problem of the previous section, for a given $\mathbf{z}$, we have $\zeta_{yx.w} = \zeta_{yxw}/P(W = w)$, $P(W = w) = \sum_{yx} \zeta_{yxw}$.

Given that the dimensionality of the problem is modest, we assign to each three-variate distribution $P(Y, X, W \,|\, \mathbf{Z} = \mathbf{z})$ an independent Dirichlet prior for every possible assignment of $\mathbf{Z}$, constrained by the inequalities implied by the model (and renormalized accordingly).

9. That is, we allow for dependence between W and Y given $\{X, \mathbf{Z}\}$, interpreting the decision of independence used in Rule 1 as being only an indicator of approximate independence.

The posterior is also a 8-dimensional constrained Dirichlet distribution, where we use rejection sampling to obtain a posterior sample by proposing from the unconstrained Dirichlet. A Dirichlet prior is assigned to $P(\mathbf{Z})$. Using a sample from the posterior of $P(\mathbf{Z})$ and a sample (for each possible value $\mathbf{z}$) from the posterior of $P(Y, X, W \mid \mathbf{Z} = \mathbf{z})$, we obtain a sample upper and lower bound for the ACE by just running the linear program for each sample of $\{\eta^{\mathbf{z}}_{yxw}\}$ and $\{P(\mathbf{Z} = \mathbf{z})\}$.

The full algorithm is shown in Algorithm 2, where $\aleph \equiv \{\epsilon_w, \epsilon_x, \epsilon_y, \underline{\beta}, \overline{\beta}\}$. The search procedure is left unspecified, as different existing approaches can be plugged into this step. See Entner et al. (2013) for a discussion. In Section 9 we deal with small dimensional problems only, using the brute-force approach of performing an exhaustive search for $\mathbf{Z}$. In practice, brute-force can be still valuable by using a method such as discrete PCA (Buntine and Jakulin, 2004) to reduce $\mathbf{W}\backslash\{W\}$ to a small set of binary variables. To decide whether the premises in Rule 1 hold, we merely perform Bayesian model selection with the BDeu score (Buntine, 1991) between the full graph $\{W \to X, W \to Y, X \to Y\}$ (conditional on $\mathbf{Z}$) and the graph with the edge $W \to Y$ removed.

Step 5 in Algorithm 2 is a "falsification test." Since the data might provide a bad fit to the constraints entailed by the model[10], we opt not to accept every pair $(W, \mathbf{Z})$ that passes Rule 1. One possibility is to calculate the posterior distribution of the model where constraints are enforced, and compare it against the posteriors of the saturated model given by the unconstrained contingency table. This requires another prior over the constraint hypothesis and the calculation of the corresponding marginal likelihoods. As an alternative approach, we adopt the pragmatic rule of thumb suggested by Richardson et al. (2011): sample M samples from the $\{\zeta^{\mathbf{z}}_{yxw}\}$ posterior given the *unconstrained* model, and check the proportion of values that are rejected. If more than 95% of them are rejected, we take this as an indication that the proposed model provides a bad fit and reject the given choice of $(W, \mathbf{Z})$.

The final result provides a set of posterior distributions over bounds, possibly contradictory, which should be summarized as appropriate. One possibility is to check for the union of all intervals or, as a simpler alternative, report the lowest of the lower bound estimates and the highest of the upper bound estimates using a point estimate for each bound:

1. for each $(W, \mathbf{Z})$ in $\mathcal{R}$, calculate the posterior expected value of the lower and upper bounds[11];
2. report the interval $\mathcal{L} \leq ACE \leq \mathcal{U}$ where $\mathcal{L}$ is the minimum of the lower bounds and $\mathcal{U}$ the maximum of the upper bounds.

In our experiments, we use a different summary. As we calculate the log-marginal posterior M_1, M_2, M_3, M_4 for the hypotheses $W \not\perp\!\!\!\perp Y \mid \mathbf{Z}$, $W \perp\!\!\!\perp Y \mid \mathbf{Z}$, $W \perp\!\!\!\perp Y \mid \mathbf{Z} \cup \{X\}$, $W \not\perp\!\!\!\perp Y \mid \mathbf{Z} \cup \{X\}$, respectively, we use the score

$$(M_1 - M_2) + (M_3 - M_4) \tag{11}$$

10. This is a result of not enforcing $W \perp\!\!\!\perp Y \mid \mathbf{Z} \cup \{X\}$ in $\eta^{\mathbf{z}}_{yxw}$.
11. Alternatively to using the expected posterior estimator for the lower/upper bounds, one can, for instance, report the 0.025 quantile of the marginal lower bound distribution and the 0.975 quantile of the marginal upper bound distribution. Notice, however, this does not give a 0.95 credible interval over ACE intervals as the lower bound and the upper bound are dependent in the posterior.

to assess the quality of the bounds obtained with the corresponding witness-admissible set pair. $M_1 - M_2$ and $M_3 - M_4$ are the log-posterior odds for the models required by the premises of Rule 1 against the respective alternatives. Just reporting the posterior of each $(W, \mathbf{Z})$ model to rank witness-admissible set pairs would not be entirely appropriate, as we are comparing models for different random variables. Adding the log-posteriors instead of a different combination is done for the sake of simplicity, contrasted to the idea of comparing the posterior of $W \to X \to Y$ against the other seven combinations of edges involving $\{W, X, Y\}$.

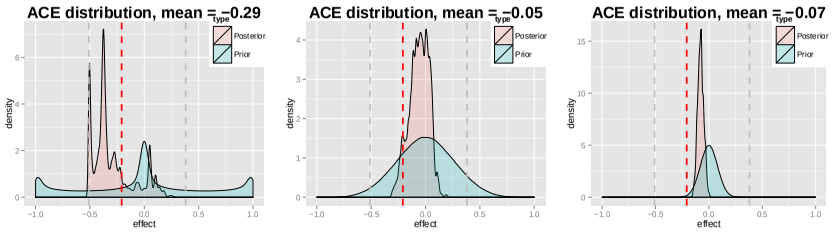
Given the score, we then report the corresponding top-scoring interval and evaluation metric based on this criterion. The reason for reporting a single representative interval is our belief that it is more productive to accommodate most of the (possibly large) discrepancies among estimated ACE intervals in the selection of $\aleph$, as done in Section 8. By selecting a single baseline with a unique lower/upper bound pair, it is simpler to communicate uncertainty, as we can then provide credible intervals or full distributions for the selected lower/upper bounds[12].

4.3 A Note on Weak Dependencies

As we briefly mentioned in the previous section, our parameterization $\{\zeta^{\mathbf{z}}_{yxw}\}$ does not enforce the independence condition $W \perp\!\!\!\perp Y \mid \mathbf{Z} \cup \{X\}$ required by Rule 1. Our general goal is to let WPP accept "near independencies," in which the meaning of the symbol $\perp\!\!\!\perp$ in practice means weak dependence[13]. We do not define what a weak dependence should mean, except for the general guideline that some agreed measure of conditional association should be "small." Our pragmatic view on WPP is that Rule 1, when supported by weak dependencies, should be used as a motivation for the constraints in Section 4.1. That is, one makes the assumption that "weak dependencies are not generated by arbitrary near-path cancellations," reflecting the belief that very weak associations should correspond to weak direct causal effects (and, where this is unacceptable, WPP should either be adapted to exclude relevant cases, or not be used). At the same time, users of WPP do not need to accept this view, as the method does not change under the usual interpretation of $\perp\!\!\!\perp$.

However, it is worthwhile to point out that computational gains can be obtained by using a parameterization that encodes the independence: that is, if we change our parameterization to enforce the independence constraint $W \perp\!\!\!\perp Y \mid \mathbf{Z} \cup \{X\}$, then there is no need to perform the check in line 5 of Algorithm 2, as the model is compatible with the (conditional on $\mathbf{Z}$) chain $W \to X \to Y$ regardless of $\aleph$. One can then generate full posterior distribution bounds only for those witness-admissible sets for which uncertainty estimates are required. The validity of point estimates of a bound is guaranteed by running a single linear programming on a point estimate of the distribution implied by the Bayesian net-

12. One should not confuse credible intervals with ACE intervals, as these are two separate concepts: each lower or upper bound is a function of the unknown $P(W, X, Y, \mathbf{Z})$ and needs to be estimated. There is posterior uncertainty over each lower/upper bound as in any problem where a functional of a distribution needs to be estimated. So the posterior distribution and the corresponding *credible intervals* over the *ACE intervals* are perfectly well-defined as in any standard Bayesian inference problem.
13. The procedure that decides conditional independencies in Section 4.2 is a method for testing exact independencies, although the prior on the independence assumption regulates how strong the evidence in the data should be for independence to be accepted.



(a) (b) (c)

Figure 3: Posterior over the ACE obtained by three different priors conditioned on a synthetic data set of size 1,000,000. Posterior computed by running 1,000,000 iterations of Gibbs sampling. The (independent) priors for $\theta^{Y}_{1.xu}$ and $\theta^{X}_{x.wu}$ are Beta (α, α), while θ^{U}_{u} is given a Dirichlet $(\alpha, \alpha, \alpha, \alpha)$. We set $\alpha = 0.1, 1, 10$ for the cases shown in (a), (b) and (c), respectively. Vertical red line shows the true ACE, while the population IV bounds are shown with gray lines. As the prior gets less informative (moving from (c) to (a)), the erratic shape of the posterior distribution also shows the effect of bad Gibbs sampling mixing. Even with a very large data set, the concentration of the posterior is highly dependent on the concentration of the prior.

work $W \to X \to Y$ (for every instance of $\mathbf{Z}$), as no further constraints in the observable distribution will exist. That is, if one wants to enforce the independence constraints, Line 4 of Algorithm 2 can be modified to directly generate just point estimates of the bounds for any witness-admissible set pair where a full posterior distribution is not required, and the falsification test in Step 5 (with the costly polytope construction) can also be skipped.

4.4 A Note on Unidentifiability

An alternative to bounding the ACE or using back-door adjustments is to put priors directly on the latent variable model for $\{W, X, Y, U\}$. Using the standard IV model as an example, we can define parameters $\theta^{Y}_{y.xu} \equiv P(Y = y \mid X = x, U = u)$, $\theta^{X}_{x.wu} \equiv P(X = x \mid W = w, U = u)$ and $\theta^{U}_{u} \equiv P(U = u)$, on which priors are imposed[14]. No complicated procedure for generating constraints in the observable marginal is necessary, and the approach provides point estimates of the ACE instead of bounds.

This sounds too good to be true, and indeed it is: results strongly depend on the prior, regardless of sample size. To illustrate this, consider a simulation from a standard IV model (Figure 1(c)), with $\mathbf{Z} = \emptyset$ and U an unobservable discrete variable of 4 levels. We generated a model by setting $P(W = w) = 0.5$ and sampling parameters $\theta^{Y}_{1.xu}$ and $\theta^{X}_{1.wu}$ from the uniform $[0, 1]$ distribution, while the 4-dimensional vector θ^{U}_{u} comes from a

14. $P(W = w)$ is not necessary, as the standard IV bounds (Balke and Pearl, 1997) do not depend on it.

Dirichlet $(1, 1, 1, 1)$. The resulting model had an ACE of -0.20, with a wide IV interval $[-0.50, 0.38]$ as given by the method of Balke and Pearl (1997). Narrower intervals can only be obtained by making more assumptions: there is no free lunch. However, as in this case where WPP cannot identify any witness, one might put priors on the latent variable model to get a point estimate, such as the posterior expected value of the ACE.

To illustrate the pitfalls of this approach, we perform Bayesian inference by putting priors directly on the CPT parameters of the latent variable model, assuming we know the correct number of levels for U. Figure 3 shows some results with a few different choices of priors. The sample size is large enough so that the posterior is essentially entirely within the population bounds and the estimation of $P(W, X, Y, Z)$ is itself nearly exact. The posterior over the ACE covers a much narrower area than the IV interval, and its behavior is erratic.

This is not to say that informative priors on a latent variable model cannot produce important results. For instance, Steenland and Greenland (2004) discuss how empirical priors on smoking habits among blue-collar workers were used in their epidemiological question: the causal effect of the occupational hazard of silica exposure on lung cancer incidence among industrial sand workers. Smoking is a confounding factor given the evidence that smoking and occupation are associated. The issue was that smoking was unrecorded among the workers, and so priors on the latent variable relationship to the observables were necessary. Notice, however, that this informative prior is essentially a way of performing a back-door adjustment when the adjustment set $\mathbf{Z}$ and treatment-outcome pair $\{X, Y\}$ are not simultaneously measured within the same subjects. When latent variables are "unknown unknowns," a prior on $P(Y \mid X, U)$ may be hard to justify. Richardson et al. (2011) discuss more issues on priors over latent variable models as a way of obtaining ACE point estimates, one alternative being the separation of identifiable and unidentifiable parameters to make transparent the effect of prior (mis)specification.

5. Algebraic Bounds and the Back-substitution Algorithm

Posterior sampling is expensive within the context of Bayesian WPP: constructing the dual polytope for possibly millions of instantiations of the problem is time consuming, even if each problem is small. Moreover, the numerical procedure described in Section 4 does not provide any insight on how the different free parameters $\{\epsilon_w, \epsilon_x, \epsilon_y, \underline{\beta}, \overline{\beta}\}$ interact to produce bounds, unlike the analytical bounds available in the standard IV case. Ramsahai (2012) derives analytical bounds under (5) given a *fixed*, *numerical* value of ϵ_w. We know of no previous analytical bounds as an algebraic function of ϵ_w.

5.1 Algebraic Bounds

We derive a set of bounds, whose validity are proved by three theorems. The first theorem derives separate upper and lower bounds on ω_{xw} using all the assumptions except Equation (5); this means constraints which do not link distributions under different values of $W = w$. The second theorem derives linear constraints on $\{\omega_{xw}\}$ using (5) and more elementary constraints. Our final result will construct less straightforward bounds, again using Equation (5) as the main assumption. As before, assume we are implicitly conditioning on some $\mathbf{Z} = \mathbf{z}$ everywhere.

We introduce the notation

$$\begin{array}{rcl} L^{YU}_{xw} & \equiv & \max(P(Y=1|X=x,W=w)-\epsilon_y,0) \\ U^{YU}_{xw} & \equiv & \min(P(Y=1|X=x,W=w)+\epsilon_y,1) \\ L^{XU}_{w} & \equiv & \max(P(X=1|W=w)-\epsilon_x,0) \\ U^{XU}_{w} & \equiv & \min(P(X=1|W=w)+\epsilon_x,1) \end{array}$$

and define $\underline{L} \equiv \min\{L^{YU}_{xw}\}, \overline{U} \equiv \max\{U^{YU}_{xw}\}$. Moreover, some further redundant notation is used to simplify the description of the constraints:

$$\begin{array}{rcl} \delta^{\star}_{1.w} & \equiv & \delta^{\star}_{w} \\ \delta^{\star}_{0.w} & \equiv & 1-\delta^{\star}_{w} \\ L^{XU}_{11} & \equiv & L^{XU}_{1} \\ L^{XU}_{01} & \equiv & 1-U^{XU}_{1} \\ U^{XU}_{11} & \equiv & U^{XU}_{1} \\ U^{XU}_{01} & \equiv & 1-L^{XU}_{1} \end{array}$$

and, following Ramsahai (2012), for any $x \in \{0,1\}$, we define x' as the complementary binary value (i.e. $x' = 1-x$). The same convention applies to pairs $\{w, w'\}$. Finally, define $\chi_{x.w} \equiv \sum_U P(X=x \mid W=w,U)P(U) = \kappa_{1x.w} + \kappa_{0x.w}$.

Theorem 2 *The following constraints are entailed by the assumptions expressed in Equations (6), (7) and (8):*

$$\omega_{xw} \leq \min \begin{cases} \kappa_{1x.w} + U^{YU}_{xw}(\kappa_{0x'.w} + \kappa_{1x'.w}) \\ \kappa_{1x.w}/L^{XU}_{xw} \\ 1 - \kappa_{0x.w}/U^{XU}_{xw} \end{cases} \tag{12}$$

$$\omega_{xw} \geq \max \begin{cases} \kappa_{1x.w} + L^{YU}_{xw}(\kappa_{0x'.w} + \kappa_{1x'.w}) \\ \kappa_{1x.w}/U^{XU}_{xw} \\ 1 - \kappa_{0x.w}/L^{XU}_{xw} \end{cases} \tag{13}$$

Theorem 3 *The following constraints are entailed by the assumptions expressed in Equations (5), (6), (7) and (8):*

$$\omega_{xw} \leq \min \begin{cases} (\kappa_{1x.w'} + \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/L^{XU}_{xw'} \\ 1 - (\kappa_{0x.w'} - \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/U^{XU}_{xw'} \end{cases} \tag{14}$$

$$\omega_{xw} \geq \max \begin{cases} (\kappa_{1x.w'} - \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/U^{XU}_{xw'} \\ 1 - (\kappa_{0x.w'} + \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/L^{XU}_{xw'} \end{cases} \tag{15}$$

$$\begin{array}{rcl} \omega_{xw} - \omega_{xw'}U^{XU}_{x'w} & \leq & \kappa_{1x.w} + \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w}) \\ \omega_{xw} - \omega_{xw'}L^{XU}_{x'w} & \geq & \kappa_{1x.w} - \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w}) \\ \omega_{xw} - \omega_{xw'}U^{XU}_{x'w} & \geq & 1 - \kappa_{0x.w} - U^{XU}_{x'w} - \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w}) \\ \omega_{xw} - \omega_{xw'}L^{XU}_{x'w} & \leq & 1 - \kappa_{0x.w} - L^{XU}_{x'w} + \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w}) \\ \omega_{xw} - \omega_{xw'} & \leq & \epsilon_w \\ \omega_{xw} - \omega_{xw'} & \geq & -\epsilon_w \end{array} \tag{16}$$

Theorem 4 *The following constraints are entailed by the assumptions expressed in Equations (5), (6), (7) and (8):*

$$\omega_{xw} \leq \min \begin{cases} \kappa_{1x'.w'} + \kappa_{1x.w'} + \kappa_{1x.w} - \kappa_{1x'.w} + \chi_{x'w}(\overline{U} + \underline{L} + 2\epsilon_w) - \underline{L} \\ \kappa_{1x'.w} + \kappa_{1x.w} + \kappa_{1x.w'} - \kappa_{1x'.w'} + 2\chi_{x'w}\epsilon_w + \chi_{x'w'}(\overline{U} + \underline{L}) - \underline{L} \end{cases} \tag{17}$$

$$\omega_{xw} \geq \max \begin{cases} -\kappa_{1x'.w'} + \kappa_{1x.w'} + \kappa_{1x'.w} + \kappa_{1x.w} + \chi_{x'w'}(\overline{U} + \underline{L}) - 2\epsilon_w\chi_{x'w} - \overline{U} \\ -\kappa_{1x'.w} + \kappa_{1x.w} + \kappa_{1x'.w'} + \kappa_{1x.w'} - \chi_{x'w}(2\epsilon_w - \overline{U} - \underline{L}) - \overline{U} \end{cases} \tag{18}$$

$$\begin{array}{rcl} \omega_{xw} + \omega_{x'w} - \omega_{x'w'} & \geq & \kappa_{1x'.w} + \kappa_{1x.w} - \kappa_{1x'.w'} + \kappa_{1x.w'} - \chi_{xw'}(\overline{U} + \underline{L} + 2\epsilon_w) + \underline{L} \\ \omega_{xw} + \omega_{x'w'} - \omega_{x'w} & \geq & \kappa_{1x'.w'} + \kappa_{1x.w'} - \kappa_{1x'.w} + \kappa_{1x.w} - 2\chi_{xw'}\epsilon_w - \chi_{xw}(\overline{U} + \underline{L}) + \underline{L} \\ \omega_{xw} + \omega_{x'w'} - \omega_{x'w} & \leq & -\kappa_{1x'.w} + \kappa_{1x.w} + \kappa_{1x'.w'} + \kappa_{1x.w'} - \chi_{xw}(\overline{U} + \underline{L}) + 2\epsilon_w\chi_{xw'} + \overline{U} \\ \omega_{xw} + \omega_{x'w} - \omega_{x'w'} & \leq & -\kappa_{1x'.w'} + \kappa_{1x.w'} + \kappa_{1x'.w} + \kappa_{1x.w} + \chi_{xw'}(2\epsilon_w - \overline{U} - \underline{L}) + \overline{U} \end{array} \tag{19}$$

Although at first such relations seem considerably more complex than those given by Ramsahai (2012), on closer inspection they illustrate qualitative aspects of our free parameters. For instance, consider

$$\omega_{xw} \geq \kappa_{1x.w} + L^{YU}_{xw}(\kappa_{0x'.w} + \kappa_{1x'.w}),$$

one of the instances of (13). If $\epsilon_y = 1$ and $\underline{\beta} = \overline{\beta} = 1$, then $L^{YU}_{xw} = 0$ and this relation collapses to $\eta_{xw} \geq \zeta_{1x.w}$, one of the original relations found by Balke and Pearl (1997) for the standard IV model. Decreasing ϵ_y will linearly increase L^{YU}_{xw} only after $\epsilon_y \leq P(Y = 1 \mid X = x, W = w)$, tightening the corresponding lower bound given by this equation.

Consider now

$$\omega_{xw} \leq 1 - (\kappa_{0x.w'} - \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/U^{XU}_{xw'}.$$

If also $\epsilon_w = 0$ and $\epsilon_x = 1$, from this inequality it follows that $\eta_{xw} \leq 1 - \zeta_{0x.w'}$. This is another of the standard IV inequalities (Balke and Pearl, 1997).

Equation (5) implies $|\omega_{x'w} - \omega_{x'w'}| \leq \epsilon_w$, and as such by setting $\epsilon_w = 0$ we have that

$$\omega_{xw} + \omega_{x'w} - \omega_{x'w'} \geq \kappa_{1x'.w} + \kappa_{1x.w} - \kappa_{1x'.w'} + \kappa_{1x.w'} - \chi_{xw'}(\overline{U} + \underline{L} + 2\epsilon_w) + \underline{L} \tag{20}$$

implies $\eta_{xw} \geq \eta_{1x.w} + \eta_{1x.w'} - \eta_{1x'.w'} - \eta_{0x.w'}$, one of the most complex relationships in (Balke and Pearl, 1997). Further geometric intuition about the structure of the binary standard IV model is given by Richardson and Robins (2010).

These bounds are not tight, in the sense that we opt not to fully exploit all possible algebraic combinations for some results, such as (20): there we use $\underline{L} \leq \eta^{\star}_{xw} \leq \overline{U}$ and $0 \leq \delta^{\star}_{w} \leq 1$ instead of all possible combinations resulting from (6) and (7). The proof idea in Appendix A can be further refined, at the expense of clarity. Because our derivation is a further relaxation, our final bounds are more conservative (that is, looser).

5.2 Efficient Optimization and Falsification Tests

Besides providing insight into the structure of the problem, the algebraic bounds give an efficient way of checking whether a proposed parameter vector $\{\zeta_{yxw}\}$ is valid in Step 5 of Algorithm 2, as well as finding the ACE bounds: we can now use back-substitution on the symbolic set of constraints to find box constraints $\mathcal{L}_{xw} \leq \omega_{xw} \leq \mathcal{U}_{xw}$. The proposed parameter will be rejected whenever an upper bound is smaller than a lower bound, and (4) can be trivially optimized conditioning only on the box constraints—this is yet another relaxation, added on top of the ones used to generate the algebraic inequalities. We initialize by intersecting all algebraic box constraints (of which (12) and (14) are examples); next we refine these by scanning relations $\pm\omega_{xw} - a\omega_{xw'} \leq c$ (the family given by (16)) in lexicographical order, and tightening the bounds of ω_{xw} using the current upper and lower bounds on $\omega_{xw'}$ where possible. We then identify constraints $\mathcal{L}_{xww'} \leq \omega_{xw} - \omega_{xw'} \leq \mathcal{U}_{xww'}$ starting from $-\epsilon_w \leq \omega_{xw} - \omega_{xw'} \leq \epsilon_w$ and the existing bounds, and plug them into relations $\pm\omega_{xw} + \omega_{x'w} - \omega_{x'w'} \leq c$ (as exemplified by (20)) to get refined bounds on ω_{xw} as functions of $(\mathcal{L}_{x'ww'}, \mathcal{U}_{x'ww'})$. We iterate this until convergence, which is guaranteed since lower/upper bounds never decrease/increase at any iteration. This back-substitution of inequalities follows the spirit of message-passing, in the sense that we iteratively update quantities of interest (intervals bounding the decision variables of the linear program) based on a small subset of other quantities, and it can be orders of magnitude more efficient than the fully numerical solution, while not increasing the width of the intervals by too much. In Section 9, we provide evidence for this claim. The back-substitution method is used in our experiments, combined with the fully numerical linear programming approach as explained in Section 9. The full method is given in Algorithm 3.

6. Linear Models

Entner et al. (2012) present a variant of their methodology for linear non-Gaussian models. The main difference is that in this case no witness variable W is necessary: it is possible to validate $\mathbf{Z}$ as an admissible set from a regression of X on $\mathbf{Z}$ and Y on $\{X, \mathbf{Z}\}$. Faithfulness is not necessary under some non-Gaussianity assumptions, although not all of these assumptions are testable without faithfulness and assumptions of parameter stability are still necessary for constraints other than independence constraints.

In this section, we adapt WPP to linear models. Vanishing partial correlations are used in the premise of Rule 1 instead of independence constraints, even if variables are non-Gaussian. The computation of the ACE bounds is vastly simplified. It complements Entner et al. (2012) in the sense that, although the method does not provide point estimates of the ACE and it might fail to identify some admissible sets, it does not require either faithfulness or non-Gaussianity[15]. Only second-order moments are necessary in the construction of the bound, although nonparametric linear models for testing partial correlations or sampling from the posterior distribution of covariance matrices might be necessary.

15. Even if variables are clearly non-Gaussian, the residuals of their regression on the admissible set might be close to Gaussian—this is after all the motivation for Gaussian likelihoods in most regression models, parametric or not.

input : Distributions $\{\zeta_{yx.w}\}$ and $\{P(W = w)\}$;
output: Lower and upper bounds $(\mathcal{L}_{xw}, \mathcal{U}_{xw})$ for every ω_{xw}

Find tightest lower and upper bounds $(\mathcal{L}_{xw}, \mathcal{U}_{xw})$ for each ω_{xw} using inequalities (12), (13) (14), (15), (17) and (18);
Let $\mathcal{L}^{\epsilon_w}_{xw}$ and $\mathcal{U}^{\epsilon_w}_{xw}$ be lower/upper bounds of $\omega_{xw} - \omega_{xw'}$;
for *each pair* $(x, w) \in \{0, 1\}^2$ **do**
  $\mathcal{L}^{\epsilon_w}_{xw} \leftarrow -\epsilon_w$;
  $\mathcal{U}^{\epsilon_w}_{xw} \leftarrow \epsilon_w$;
end
while *TRUE* **do**
  for *each relation* $\omega_{xw} - b \times \omega_{xw'} \leq c$ *in (16)* **do**
    $\mathcal{U}^{\epsilon_w}_{xw} \leftarrow \min\{\mathcal{U}^{\epsilon_w}_{xw}, (b-1)\mathcal{L}_{xw} + c\}$
  end
  for *each relation* $\omega_{xw} - b \times \omega_{xw'} \geq c$ *in (16)* **do**
    $\mathcal{L}^{\epsilon_w}_{xw} \leftarrow \max\{\mathcal{L}^{\epsilon_w}_{xw}, (b-1)\mathcal{U}_{xw} + c\}$
  end
  for *each relation* $\omega_{xw} + \omega_{x'w} - \omega_{x'w'} \leq c$ *in (19)* **do**
    $\mathcal{U}_{xw} \leftarrow \min\{\mathcal{U}_{xw}, c - \mathcal{L}^{\epsilon_w}_{xw'}\}$
  end
  for *each relation* $\omega_{xw} - (\omega_{x'w} - \omega_{x'w'}) \leq c$ *in (19)* **do**
    $\mathcal{U}_{xw} \leftarrow \min\{\mathcal{U}_{xw}, c + \mathcal{U}^{\epsilon_w}_{xw'}\}$
  end
  for *each relation* $\omega_{xw} + \omega_{x'w} - \omega_{x'w'} \geq c$ *in (19)* **do**
    $\mathcal{U}_{xw} \leftarrow \max\{\mathcal{U}_{xw}, c - \mathcal{U}^{\epsilon_w}_{xw'}\}$
  end
  for *each relation* $\omega_{xw} - (\omega_{x'w} - \omega_{x'w'}) \geq c$ *in (19)* **do**
    $\mathcal{U}_{xw} \leftarrow \max\{\mathcal{U}_{xw}, c + \mathcal{L}^{\epsilon_w}_{xw'}\}$
  end
  if *no changes in* $\{(\mathcal{L}_{xw}, \mathcal{U}_{xw})\}$ **then**
    break
  end
end
return $(\mathcal{L}_{xw}, \mathcal{U}_{xw})$ *for each* $(x, w) \in \{0, 1\}^2$

Algorithm 3: The iterative back-substitution procedure for bounding $\mathcal{L}_{xw} \leq \omega_{xw} \leq \mathcal{U}_{xw}$ for all combinations of x and w in $\{0, 1\}^2$.

6.1 A Bounding Procedure for Linear Models

Consider for now the linear model case with an empty admissible set $\mathbf{Z}$:

$$\begin{aligned} X &= aW + U_x \\ Y &= bX + cW + U_y, \end{aligned} \tag{21}$$

where $\{W, U_x, U_y\}$ are assumed to be zero mean variables, and $\{U_x, U_y\}$ are unobservable. The case with non-empty $\mathbf{Z}$ is analogous and discussed in the next section. The ACE is

given by b. We denote as s_{ww}, s_{wx}, s_{wy}, ... the corresponding variances/covariances of $\{W, U_x, U_y\}$. Moreover, let the variances of $\{W, U_x, U_y\}$ be set such that each element of $\{W, X, Y\}$ has unit variance, and denote as $\rho_{wx}, \rho_{wy}, \rho_{xy}$ the corresponding correlations of $\{W, X, Y\}$. Notice that $s_{ww} = 1$, and no assumptions about Gaussianity are being made. As before, we assume for now that $\rho_{wx}, \rho_{wy}, \rho_{xy}$ are known constants, and we would like to bound b as a function of this observable correlation matrix. The *implied* correlation matrix of model (21) needs to match the observable correlation matrix:

$$\rho_{wx} = a + s_{wx} \tag{22}$$
$$\rho_{wy} = b\rho_{wx} + c + s_{wy} \tag{23}$$
$$\rho_{xx} = 1 = a^2 + 2as_{wx} + s_{xx} \tag{24}$$
$$\rho_{xy} = b + c\rho_{wx} + as_{wy} + s_{xy} \tag{25}$$
$$\rho_{yy} = 1 = b^2 + 2bc\rho_{wx} + c^2 + s_{yy} + 2[b(as_{wy} + s_{xy}) + cs_{wy}], \tag{26}$$

where the above identities follow directly from (21). The feasible values of the parameters are given by the intersection of the above and

$$-\epsilon_c \leq c \leq \epsilon_c \tag{27}$$

$$-\epsilon_{wx} \leq s_{wx} \leq \epsilon_{wx}, \quad -\epsilon_{wy} \leq s_{wy} \leq \epsilon_{wy}, \quad -\epsilon_{xy} \leq s_{xy} \leq \epsilon_{xy} \tag{28}$$

$$0 \leq s_{xx} \leq 1, \quad 0 \leq s_{yy} \leq 1. \tag{29}$$

We ignore the positive semidefiniteness requirement of the covariance matrix of $\{W, U_X, U_Y\}$ for simplicity.

The set of constraints can be simplified as follows:

Theorem 5 *Assume*[16] $\rho_{wy} = \rho_{wx}\rho_{xy}$. *If an assignment of values to* $\{a, b, c, s_{wx}, s_{xx}, s_{xy}, s_{xx}\}$ *satisfies (27)-(29), then it satisfies (22)-(26) if and only it satisfies the following:*

$$\rho_{wy} = b\rho_{wx} + c + s_{wy} \tag{30}$$

$$\begin{cases} \rho_{xy} - \epsilon_{xy} - \mathcal{U}_a s_{wy} \leq b + c\rho_{wx} \leq \rho_{xy} + \epsilon_{xy} - \mathcal{L}_a s_{wy}, & \text{if } s_{wy} \geq 0 \\ \rho_{xy} - \epsilon_{xy} - \mathcal{L}_a s_{wy} \leq b + c\rho_{wx} \leq \rho_{xy} + \epsilon_{xy} - \mathcal{U}_a s_{wy}, & \text{if } s_{wy} < 0 \end{cases} \tag{31}$$

$$b^2 + 2bc\rho_{wx} + c^2 - 2(b\rho_{xy} + c\rho_{wy}) \leq 0 \tag{32}$$

where $\mathcal{L}_a \equiv \max(\min(0, 2\rho_{wx}), \rho_{wx} - \epsilon_{wx})$ *and* $\mathcal{U}_a \equiv \min(\max(0, 2\rho_{wx}), \rho_{wx} + \epsilon_{wx})$.

This means that optimizing b subject to constraints (27)−(32) is a convex program on $\{b, c, s_{wy}\}$ (conditioned on the sign of s_{wy}). Notice that, because of the assumption $\rho_{wy} = \rho_{wx}\rho_{xy}$, the system is always satisfiable. It can nevertheless rule out some the possible values of b (e.g. $b = \rho_{xy} = \rho_{wy}/\rho_{wx}$ if $\epsilon_{xy} = \epsilon_{wy}$ or $\epsilon_c = \epsilon_{wy} = 0$). Given $\{\rho_{wx}, \rho_{xy}\}$ (and setting $\rho_{wy} = \rho_{wx}\rho_{xy}$), we can find an upper bound for the ACE by maximizing b under the constraints (27)−(32) and $0 \leq s_{wy} \leq \epsilon_{wy}$, followed by maximization under the condition $-\epsilon_{wy} \leq s_{wy} \leq 0$. The upper bound is the maximum of the two conditional maxima. The lower bound is derived in an analogous way.

16. This assumption can be dropped, but the proof of Theorem 5 gets more complicated.

6.2 Algorithm for Gaussian Copula Models

One general model family in which vanishing partial correlations are closely connected to independence is the Gaussian copula (Elidan, 2013; Nelsen, 2007). Consider the following generative model:

$$\begin{aligned} \mathbf{V}^\star &\sim \mathcal{N}(0, R) \\ V_i &= F_i^{-1}(\phi(V_i^\star)), i = 1, 2, \ldots, p, \end{aligned} \tag{33}$$

where $\mathbf{V}^\star$ is a p-dimensional random vector generated according to the Gaussian with $p \times p$ correlation matrix R, $F_i(\cdot)$ is some arbitrary cumulative distribution function (CDF), and $\phi(\cdot)$ is standard Gaussian CDF. In continuous distributions, $F_i(\cdot)$ is invertible, and Markov properties of $\mathbf{V}^\star$ are preserved in the distribution of $\mathbf{V}$. See Harris and Drton (2013) for a discussion of Gaussian copula models in the context of causal inference, in particular for structure learning using the PC algorithm (Spirtes et al., 2000). Causal structure and effects are defined for $\mathbf{V}^*$ as in a typical linear causal system. Conditional independencies can be tested by copula-based measures, such as Spearman's rank correlation, by testing for the corresponding vanishing partial correlations. Given a target treatment $X \in \mathbf{V}$ and outcome $Y \in \mathbf{V}$, we are interested in bounding the ACE of $X^\star$ in $Y^\star$, the Gaussian variables underlying the possibly non-Gaussian X and Y.

For simplicity, we search for admissible sets $\mathbf{Z}$ with corresponding witness W using a Gaussian copula correlation matrix estimate $\hat{R}$. The unobserved data for $\mathbf{V}^\star$ is then for simplicity assumed to have zero empirical mean and empirical covariance matrix $\hat{R}$. We score models entailing independence of some V_i and V_j given $\mathbf{V}_Z$ by scoring two Gaussian networks, $G_1 \equiv \{\mathbf{V}_Z^\star \to V_i^\star, \mathbf{V}_Z^\star \to V_j^\star\}$ against $G_2 \equiv \{\mathbf{V}_Z^\star \to V_i^\star, \mathbf{V}_Z^\star \to V_j^\star, V_i^\star \to V_j^\star\}$. This is analogous to the binary case, where here we use the corrected BGe score (Kuipers et al., 2014). Notice this test is approximate, as $\hat{R}$ is used as a surrogate for the empirical covariance matrix of the unobserved data $\mathbf{V}^\star$, which is required by BGe[17].

For a given $(W, \mathbf{Z})$ accepted by Rule 1, we calculate the empirical residual (rank) correlation matrix obtained by regressing $W^\star$ on $\mathbf{Z}^\star$, $X^\star$ on $W^\star$ and $\mathbf{Z}^\star$, and $Y^\star$ on $X^\star$ and $\mathbf{Z}^\star$, so that the partial (residual) correlation of W and Y given X and $\mathbf{Z}$ is zero. Regression of subsets of $\mathbf{V}^\star$ on other subsets is done by standard regression using $\hat{R}$: let $\{\hat{\sigma}_{ww.\mathbf{z}}, \hat{\sigma}_{xx.\mathbf{z}}, \hat{\sigma}_{yy.\mathbf{z}}\}$ be the residual variances of the regression of $\{W^\star, X^\star, Y^\star\}$ on $\mathbf{Z}^\star$ as defined by $\hat{R}$. The resulting residual covariance matrix is scaled to unit variance, and the method in Section 6.1 is used to generate scaled bounds $\mathcal{L}_{b_s} \leq b_{standardized} \leq \mathcal{U}_{b_s}$, which are converted in bounds on the ACE in the original scale as $[\sqrt{\hat{\sigma}_{xx}\hat{\sigma}_{yy}}\mathcal{L}_{b_s}, \sqrt{\hat{\sigma}_{xx}\hat{\sigma}_{yy}}\mathcal{U}_{b_s}]$.

The algorithm is basically the same as Algorithm 2, except we report only point estimates for the bounds instead of posteriors, and no falsification step is necessary (Step 5 of Algorithm 2) as the model cannot be falsified given the accepted conditional independence.

17. Alternatively, one could test for the corresponding vanishing partial correlations in the empirical Spearman rank correlation matrix, as suggested by Harris and Drton (2013), at a particular significance level α. However, this only provides p-values, which are not ideal to sort witness/admissible sets by a score, as p-values measure only the surprise of seeing the observed data under a constraint. This does not measure strength of dependence nor a posterior over models. A fully Bayesian version of this approach is conceptually simple, although nonparametric modeling of $\{F_i(\cdot)\}$ (the so-called nonparanormal model) might require Markov chain Monte Carlo methods and computing marginal likelihoods is computationally very intensive.

7. Zero Effects

Under faithfulness, the premise of Rule 1 will not be true in a system where X is not a cause of Y. The result is that no conclusion about the ACE can be made, but identifiability can still be achieved by other means. Rule 2 (Entner et al., 2013) covers all identifiable cases where X is not a cause of Y:

Rule 2a: *If there exists a set* $\mathbf{Z} \subseteq \mathbf{W}$ *such that* $X \perp\!\!\!\perp Y \,|\, \mathbf{Z}$, *then infer an ACE of zero.*

Rule 2b: *If there exists a variable* $W \in \mathbf{W}$ *and a set* $\mathbf{Z} \subseteq \mathbf{W} \backslash \{W\}$ *such that:*

$$\textit{(i)} \quad W \not\perp\!\!\!\perp X \,|\, \mathbf{Z} \qquad\qquad \textit{(ii)} \quad W \perp\!\!\!\perp Y \,|\, \mathbf{Z},$$

then infer an ACE of zero.

Rule 2a is a direct application of faithfulness, while Rule 2b essentially corresponds to the "unshielded collider" check in the FCI algorithm of Spirtes et al. (2000). Figure 4 illustrates the paths that can be weakened under Rules 2a and 2b, excluding any a priori restrictions on $X \to Y$, since this is the relation that we want to bound given conditions on other paths. It is clear that for 2a not much of interest can be said beyond this: any association that cancels the causal effect of X and Y should be due to a corresponding association generated by unmeasured confounders. If such a strong contribution due to confounders does not exist, then we should not expect the ACE to be strong[18].

Rule 2b seems more interesting. However, as suggested by Figure 4(b), we are forcing *fewer* structural constraints in the linear program compared to Rule 1, as there is nothing from Rule 2b that motivates weak confounding effects. The reason for that is that Rule 2b concerns the removal of $X \to Y$, which corresponds to the effect we want to bound as a consequence of assumptions elsewhere (instead of assuming a priori, say, $|ACE| \leq \epsilon_\emptyset$ for some new hyperparameter $\epsilon_\emptyset$). One possibility is that for a pair $(W, \mathbf{Z})$ that satisfies Rule 2b, we perform the standard WPP bounding with $\epsilon_x = \epsilon_y = 1$ and, if desired, the added constraint $|ACE| \leq \epsilon_\emptyset$ to be assumed given the firing of Rule 2b.

Another possibility is to exploit Rule 2 to learn something about the possible effects of W on Y: in this case, we condition on constraints $|\eta^\star_{0w} - \eta^\star_{1w}| \leq \epsilon_\emptyset$ to derive bounds on the direct effect of W on Y (Cai et al., 2008). In the context of our ACE problem, it might suggest information about ϵ_w that can be reused in another suggested pair $(W, \mathbf{Z}')$, but this will require further assumptions or tests, as the differences between $\mathbf{Z}$ and $\mathbf{Z}'$ will make this transfer of information not trivial. For the rest of the paper, we will ignore the use of Rule 2 for simplicity. In the next section, however, we will consider the implications of having different pairs of witness/admissible set as a way of learning information about our hyperparameters.

18. This is *not* to say that such an observation has little scientific value. A similar statement is that a strong association between X and Y should be indicative of some causal effect, in the absence of a set of confounders that could fully explain this association. Simple as this is, this type of reasoning has long been explored in observational studies (Cornfield et al., 1959), and it is essentially what is behind Rosenbaum's sensitivity analysis methods (Rosenbaum, 2002a). Our point is that the linear programming approach for this setup is trivial.

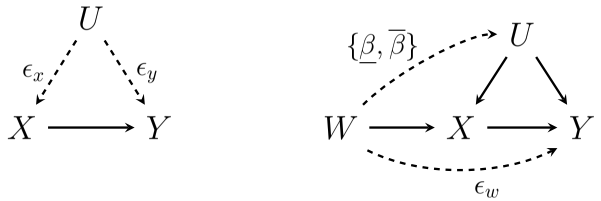


Figure 4: An illustration of conditional dependencies to be weakened under the acceptance of Rule 2. (a) Unmeasured confounding between X and Y is considered when these two variables are (conditionally) independent (given a possibly non-empty set $\mathbf{Z}$). (b) (Conditional) observable independence of W and Y is used to suggest that W and Y have bounded dependence conditioned on U, as well as weak dependence between W and U. Notice no weakening of effects $\{U \to X, U \to Y\}$.

8. Choosing Relaxation Parameters

The free parameters $\aleph \equiv \{\epsilon_w, \epsilon_x, \epsilon_y, \underline{\beta}, \overline{\beta}\}$ do not have a unique, clear-cut, domain-free procedure by which they can be calibrated. However, as we briefly discussed in Section 4, it is useful to state explicitly the following simple guarantee of WPP:

Corollary 6 *Given* $W \not\perp\!\!\!\perp Y \mid \mathbf{Z}$ *and* $W \perp\!\!\!\perp Y \mid \{X, \mathbf{Z}\}$, *the* WPP *population bounds on the ACE will always include the back-door adjusted population ACE based on* $\mathbf{Z}$.

Proof The proof follows directly by plugging in the quantities $\epsilon_w = \epsilon_y = \epsilon_x = 0$, $\underline{\beta} = \overline{\beta} = 1$, into the analytical bounds of Section 5.1, which will give the tightest bounds on the ACE (generalized to accommodate a background set $\mathbf{Z}$): a single point, which is also the functional obtained by the back-door adjustment. ■

The implication is that, regardless of the choice of free parameters, the result is guaranteed to be more conservative than the one obtained using the faithfulness assumption. This does not mean that a judicious choice of relaxation parameters is of secondary importance. Ideally, domain knowledge should be used: given a witness and admissible set, an expert decides which relaxations are reasonable. This is domain dependent, and might not be easier than choosing an admissible set from background knowledge. As an alternative, this section covers more generic methods for choosing relaxation parameters. Two main approaches are discussed:

- $\aleph$ is deduced by the outcome of a sensitivity analysis procedure; given a particular interval length L, we derive a quantification of faithfulness violations (represented by $\aleph$) required to generate causal models compatible with the observational data and an interval of length L containing the ACE. This is covered in Section 8.1;

- exploit the multiplicity of solutions (pairs of candidate witness/admissible sets) usually provided by Rule 1 to learn about the extent of possible faithfulness violations. Combine the multiple solutions with constraints or prior distributions for $\aleph$ to obtain estimates of the relaxation parameters. This is covered in Section 8.2.

8.1 Choice by Grid Search Conditioned on Acceptable Information Loss

One pragmatic default method is to first ask how wide an ACE interval can be so that the result is still useful for the goals of the analysis (e.g., sorting possible control variables X as candidates for a lab experiment based on lower bounds on the ACE). Let L be the interval width the analyst is willing to accept. Set $\epsilon_w = \epsilon_x = \epsilon_y = k_\epsilon$ and $\underline{\beta} = c$, $\overline{\beta} = 1/c$, for some pair (k_ϵ, c) such that $0 \leq k_\epsilon < 1$, $0 < c \leq 1$, and let (k_ϵ, c) vary over a grid of values. For each witness/admissible set candidate pair, pick the (k_ϵ, c) choice(s) entailing interval(s) of length closest to L. In case of more than one solution, summarize them by a criterion such the union of the intervals.

This methodology provides an explicit trade-off between length of the interval and tightness of assumptions. Notice that, starting from the back-door adjusted point estimator of Entner et al. (2013), it is not obvious how the trade-off could be obtained: that is, how to build an interval around the back-door point estimate that can be interpreted as bounds under an acceptable amount of information loss. WPP provides a principled way of building such an interval, with the resulting assumptions on $\aleph$ being explicitly revealed as a by-product. If the analyst believes that the resulting values of $\aleph$ are not strict enough, and no substantive knowledge exists that allows particular parameters to be tightened up, then one either has to concede that wider intervals are necessary or to find other means of identifying the ACE without the faithfulness assumption[19].

In the experiments in Section 9.2, we define a parameter space of $k_\epsilon \in \{0.05, 0.10, \ldots, 0.30\}$ and $c \in \{0.9, 1\}$. More than one interval of approximately the same width is identified. For instance, the configurations $(k_\epsilon = 0.25, c = 1)$ and $(k_\epsilon = 0.05, c = 0.9)$ both produced intervals of approximately length 0.30.

8.2 Linking Selection on the Observables to Selection on the Unobservables

Observational studies cannot be carried out without making assumptions that are untestable given the data at hand. There will always be degrees of freedom that must be chosen, even if such choices are open to criticism. The game is to provide a language to express assumptions in as transparent a manner as possible. Our view on priors for the latent variable model (Section 4.4) is that such prior knowledge is far too difficult to justify when the interpretation of U is unclear. Moreover, putting a prior on a parameter such as $P(Y = 1 \,|X = x, W = w, U = u)$ so that this prior is bounded by the constraint $|P(Y = 1 \,|X = x, W = w, U = u) - P(Y = 1 \mid X = w, W = w)| \leq \epsilon_w$ has no clear advantage over WPP: a specification of the shape of this prior is still necessary and may have undesirable side effects; it has no computational advantages, as constraints will have to be dealt with now within a Markov chain Monte Carlo procedure; it provides no insight on how constraints are related to one another (Section 5); it still suggests a point estimate that should not be trusted lightly, and posterior bounds which cannot be interpreted as data-driven bounds; and it still requires a choice of ϵ_w.

That is not to say that subjective priors on the relationship between U and the observables cannot be exploited, but the level of abstraction at which they need to be specified should have advantages when compared to the latent variable model approach. For instance, Altonji et al. (2005) introduced a framework to deal with violations of the IV assumptions (in the context of linear models). Their main idea is to linearly decompose the (observational) dependence of W and $\mathbf{Z}$, and the (causal) dependence of Y and $\mathbf{Z}$, as two signal-plus-noise decompositions, and assume that dependence among the signals allows one to infer the dependence among the noise terms. In this linear case, the dependence among noise terms gives the association between W and Y through unmeasured confounders. The constraint given by the assumption can then be used to infer bounds on the (differential) ACE. The details are not straightforward, but the justification for the assumption is indirectly derived by assuming $\mathbf{Z}$ is chosen by a sampling mechanism that picks covariates from the space of confounders U, so that $|\mathbf{Z}|$ and $|U|$ are large. The core idea is that the dependence between the covariates which are observed (i.e. $\mathbf{Z}$) and the other variables (W, X, Y) should tell us something about the impact of the unmeasured confounders. Their method is presented for linear models only, and the justification requires a very large $|\mathbf{Z}|$.

We introduce a very different method inspired by the same general principle, but exploiting the special structure of our procedure. Instead of relying on linearity and a fixed set of covariates, consider the following postulate: the variability of back-door adjusted ACE estimators based on different admissible sets, as implied by Rule 1, should provide some information about the extent of faithfulness violations in the given domain. In what follows, let $\aleph$ be simplified so that $\aleph \equiv \{\epsilon_w, \epsilon_{xy}, \beta\}$, where $\epsilon_{xy} \equiv \epsilon_x = \epsilon_y$ and $\beta \equiv \overline{\beta} = 1/\underline{\beta}$. The task then is to choose the three parameters in this set.

8.2.1 Method 1: Tightest ACE Coverage

Let $\{(W_1, \mathbf{Z}_1), \ldots, (W_k, \mathbf{Z}_k)\}$ be the set of all pairs found by WPP. Let A_i be the ACE calculated by the back-door adjustment on $\mathbf{Z}_i$, which will be the true ACE if faithfulness holds (we assume for now the joint distribution of the observables is known, so no statistical uncertainty plays a role yet). Let $(\mathcal{L}_i(\aleph), \mathcal{U}_i(\aleph))$ be the corresponding lower bound and upper bound implied by $(W_i, \mathbf{Z}_i)$ and $\aleph$. Finally, let $i^\star \in \{1, 2, \ldots, k\}$ be the index of a witness/admissible set that will be our reference pair[20] to output the final bounds on the ACE, once we choose $\aleph$.

The idea is simple: minimize $(\epsilon_w, \epsilon_{xy}, \beta)$ subject to $A_j \in [\mathcal{L}_{i^\star}(\aleph), \mathcal{U}_{i^\star}(\aleph)]$ for $1 \leq j \leq k$. This is a multi-objective minimization problem, of which we can return the Pareto frontier. Because this is a small dimensional problem in which high precision is not needed, a simple grid search will suffice, as performed in Section 9. Given that the Pareto frontier is likely to contain multiple points, we can report all intervals implied by each possible choice of $\aleph$.

19. That is, expert knowledge should of course still be invoked to decide whether the resulting relaxation is plausible or not (and hence, whether the resulting interval is believable), although communication by sensitivity analysis might facilitate discussion and criticism of the study. In his rejoinder to the discussion of (Rosenbaum, 2002b), Rosenbaum points out that the sensitivity analysis procedure just states the logical outcome of the structural assumptions: the deviation of (say) $P(Y = 1 \mid X = x, W = w)$ from $P(Y = 1 \mid X = x, W = w, U = u)$, required to explain the given magnitude of variation of plausible ACEs, is not imposed a priori by expert knowledge, but deduced.

20. The score in Equation (11) is used to pick $i^\star$.

Alternatively, we can provide a summary of the resulting bounds, such as the union of the intervals.

The rationale for this is as follows: if faithfulness is true, then all A_i will collapse to the same value, which implies that all bounds will collapse to a single point. Differences among A_i are the result of faithfulness violations, and we explain the contradictions via our $\aleph$ parameters. Contradictions in constraints entailed by faithfulness have been exploited before to achieve robust causal inference, as in the Conservative PC algorithm of Ramsey et al. (2006). To the best of our knowledge, we provide here the first algorithm for accommodating faithfulness contradictions in a space of constraints other than conditional independence constraints among observables.

For the real case where the observable joint distribution needs to be estimated from data, one simple alternative is just to use empirical estimates of the unconstrained joint. In one sense, this provides a conservative choice of $\aleph$, as one could modify the constraints in the minimization of $(\epsilon_w, \epsilon_{xy}, \beta)$ to require instead a less stringent criterion: that (say) the 95% credible interval for each A_j overlaps with credible intervals for $[\mathcal{L}_{i^*}(\aleph), \mathcal{U}_{i^*}(\aleph)]$. Credible intervals can be obtained as a function of the posterior distribution of the parameters of the joint of $\{X, Y, W_1, \ldots, W_k\} \cup \bigcup_{i=1}^{k} \mathbf{Z}_i$, where a prior over the multivariate binary distributions is subject to the WPP constraints for $(W_i, \mathbf{Z}_i)$ and the independence constraints for all other pairs, at each candidate value of $\aleph$. This is very costly, and better sampling procedures than the off-the-shelf rejection sampler will be necessary. We adopt the simple conservative approach with plug-in estimators instead.

8.2.2 Method 2: Bayesian Learning of Relaxation Parameters

A criticism of the tightest ACE coverage method is that it does not take into account the size of k: for $k = 1$, it will return $\epsilon_w = \epsilon_{xy} = 0$, for instance. Judgment is necessary on whether k is large enough in order to trust the results of this analysis. Alternatively, one may cast the problem of learning $\aleph$ as yet another Bayesian learning problem, with $\{(W_1, \mathbf{Z}_1), \ldots, (W_k, \mathbf{Z}_k)\}$ providing evidence for $\aleph$ according to some reasonable definition of "likelihood function." In what follows, again we assume the joint distribution of the observables is given, so that the back-door ACE functionals $A_1, A_2, \ldots, A_k$ given by Rule 1 are observable. As in the previous section, in our implementation we just plug-in the empirical distribution of the observables, but more sophisticated approaches accounting for the uncertainty in this estimate can in principle be constructed.

The principle is: if we allow for many ways in which faithfulness violations might be detected by contradictory results, but contradictions are not found, then this should be evidence that faithfulness violations do not exist. If contradictions are "small" (i.e., ACEs implied by different back-door adjustments are close), faithfulness violations should be small. Our uncertainty should decrease as more opportunities for contradictions are allowed. In particular, we want the posterior to converge to the single values $\epsilon_w = 0, \epsilon_{xy} = 0$ and $\beta = 1$ as the number of witness/admissible set pairs increase and they agree on the same value.

We start by defining

$$d_w \equiv \max_{i,x,\mathbf{z}} |P(Y = 1 \mid X = x, W_i = 1, \mathbf{Z}_i = \mathbf{z}, U) - P(Y = 1 \mid X = x, W_i = 0, \mathbf{Z}_i = \mathbf{z}, U)|,$$

for $i = 1, 2, \ldots, k$. An analogous definition is given for d_{xy} and d_β. Next, we define the "likelihood" function for d_w under "data set" $\{A_1, A_2, \ldots A_k\}$ as follows;

$$P(A_1, A_2, \ldots, A_k \mid d_w, d_{xy}, d_\beta) = \prod_{i=1}^{k} p_{U[\mathcal{L}(d_w, d_{xy}, d_\beta), \mathcal{U}(d_w, d_{xy}, d_\beta)]}(A_i), \tag{34}$$

where $p_{U[a,b]}(\cdot)$ is the uniform distribution in $[a, b]$. Functions $\mathcal{L}(d_w, d_{xy}, d_\beta), \mathcal{U}(d_w, d_{xy}, d_\beta)$ are the respective lower bound and upper bound implied by the WPP constraints parameterized by $\{d_w, d_{xy}, d_\beta\}$, and the (given) joint distribution of the observables. A uniform (discrete) prior for $\{d_w, d_{xy}, d_\beta\}$ is given over a pre-defined grid of values for these parameters[21]. We then choose a set of $\{\epsilon_w, \epsilon_{xy}, \beta\}$ as the high posterior density region defined by sorting all $\{d_w, d_{xy}, d_\beta\}$ in decreasing value of posterior mass, picking the minimum set that adds up to at least 95% of posterior mass. We summarize the implied set of bounds as necessary, see Section 9.

There is no reason why a uniform prior and the uniform likelihood (34) should be the only choices. Our motivation is that the chosen likelihood function penalizes parameters that imply wide intervals, while remaining agnostic about the position of each ACE within bounds. More importantly, the penalization increases as k increases, making the posterior more peaked. It however forces all intervals of equal length to be distinguished based on the prior only. Priors matter in applied work, but in our experiments we choose the uniform prior for its simplicity. We leave the discussion of other choices of likelihood and priors for future work.

A criticism of Equation (34) is that the pairs in set $\{(W_1, \mathbf{Z}_1), \ldots, (W_k, \mathbf{Z}_k)\}$ might have much overlap (in the sense that a same witness may appear in many pairs, and the intersection among $\{\mathbf{Z}_i\}$ may be large). As such, the multiplication in Equation (34) provides overconfident posteriors, as pairs are considered to be independent pieces of information for the relaxation parameters. More free parameters accounting for the dependence of $\{A_i\}$ given $\{d_w, d_{xy}, d_\beta\}$ should be added. However, while we remove a class of irrelevant pairs (any $(W_j, \mathbf{Z}_j)$ such that there is some $i \neq j$ where $\mathbf{Z}_i \subset \mathbf{Z}_j$ and $W_i = W_j$), in this work we ignore more complex adjustments for simplicity.

9. Experiments

In this section, we start with a comparison of the back-substitution algorithm of Section 5.2 against the fully numerical procedure, which generates constraints using standard algorithms for changing between polytope representations. We then perform studies with synthetic data, comparing different back-door estimation algorithms against WPP. Finally, we perform studies with real data sets.

21. See Section 9. Using a pre-defined discretization simplifies computation, as no MCMC is required and we do not need high precision in estimating relaxation parameters. A continuous space would also imply challenges to the MCMC approach, as the posterior can be flat in some regions where different parameter settings imply intervals of same length $\mathcal{U}(d_w, d_{xy}, d_\beta) - \mathcal{L}(d_w, d_{xy}, d_\beta)$.

9.1 Empirical Investigation of the Back-substitution Algorithm

We compare the back-substitution algorithm introduced in Section 5.2 with the fully numerical algorithm. Comparison is done in two ways: (i) computational cost, as measured by the wallclock time taken to generate 100 samples by rejection sampling; (ii) width of the generated intervals. As discussed in Section 5.2, bounds obtained by the back-substitution algorithm are at least as wide as in the numerical algorithm, barring rounding problems[22].

We ran two batches of 1000 trials each, varying the level of the relaxation parameters. In the first batch, we set $\epsilon_x = \epsilon_y = \epsilon_w = 0.2$, and $\underline{\beta} = 0.9$, $\overline{\beta} = 1.1$. In the second batch, we change parameters so that $\underline{\beta} = \overline{\beta} = 1$. Experiments were run on a Intel Xeon E5-1650 at 3.20Ghz. Models were simulated according the the structure $W \to X \to Y$, sampling each conditional distribution of a vertex being equal to 1 given its parent from the uniform $(0, 1)$ distribution. The numerical procedure of converting extreme points to linear inequalities was done using the package RCDD, a R wrapper for the *cddlib* by Komei Fukuda. Inference is done by rejection sampling, requiring 100 samples per trial. We fix the number of iterations of the back-substitution method to 4, which is more than enough to achieve convergence. All code was written in R.

For the first batch, the average time difference between the fully numerical method and the back-substitution algorithm was 1 second, standard deviation (s.d.) 0.34. The ratio between times had a mean of 203 (s.d. 82). Even with a more specialized implementation of the polytope dualization step[23], two orders of magnitude of difference seem hard to remove by better coding. Concerning interval widths, the mean difference was 0.15 (s.d. 0.06), meaning that the back-substitution on average has intervals where the upper bound minus the lower bound difference is 0.15 units more than the numerical method, under this choice of relaxation parameters and averaged over problems generated according to our simulation scheme. There is a correlation between the width difference and the interval width given by the numerical method the gap, implying that differences tend to be larger when bounds are looser: the gap between methods was as small as 0.04 for a fully numerical interval of width 0.19, and as large as 0.23 for a fully numerical interval of width 0.49. For the case where $\overline{\beta} = \underline{\beta} = 1$, the average time difference was 0.92 (s.d. of 0.24), ratio of 152 (s.d. 54.3), interval width difference of 0.09 (s.d. 0.03); The gap was as small as 0.005 for a fully numerical interval of width 0.09, and as large as 0.17 for a fully numerical interval of with 0.23.

9.2 Synthetic Studies

We describe a set of synthetic studies for binary data where, for procedures that estimate ACE intervals, we assess the trade-off between its correctness (that is, how far from the true ACEs the intervals are, for a suitable definition of distance) and its informativeness (how wide the intervals are).

22. We did not use rational arithmetic in the polytope generator in order to speed it up; consequently, about 1% of the time we observed numerical problems. Those were excluded from the statistics reported in this section.
23. One advantage of the analytical bounds, as used by the back substitution method, is that it is easy to express them as matrix operations over all Monte Carlo samples, while the polytope construction requires iterations over the samples.

In the synthetic study setup, we compare our method against NE1 and NE2, two naïve point estimators defined by back-door adjustment on the whole of set of available covariates $\mathbf{W}$ and on the empty set, respectively. The former is widely used in practice, even when there is no causal basis for doing so (Pearl, 2009). The point estimator of Entner et al. (2013), based solely on the faithfulness assumption, is also assessed.

We generate problems where conditioning on the whole set $\mathbf{W}$ is guaranteed to give incorrect estimates. In detail: we generate graphs where $\mathbf{W} \equiv \{Z_1, Z_2, \ldots, Z_8\}$. Four independent latent variables $L_1, \ldots, L_4$ are added as parents of each $\{Z_5, \ldots, Z_8\}$; L_1 is also a parent of X, and L_2 a parent of Y. L_3 and L_4 are each randomly assigned to be a parent of either X or Y, but not both. $\{Z_5, \ldots, Z_8\}$ have no other parents. The graph over $Z_1, \ldots, Z_4$ is chosen by adding edges uniformly at random according to a fixed topological order. As a consequence, using the full set $\mathbf{W}$ for back-door adjustment is always incorrect, as at least four paths $X \leftarrow L_1 \to Z_i \leftarrow L_2 \to Y$ are active for $i = 5, 6, 7, 8$. The conditional probabilities of a vertex given its parents are generated by a logistic regression model with pairwise interactions, where parameters are sampled according to a zero mean Gaussian with standard deviation 20 / number of parents. Parameter values are also further bounded, so that if the generated value if greater than 0.975 or less than 0.025, it is resampled uniformly in $[0.950, 0.975]$ or $[0.025, 0.050]$, respectively.

We analyze two variations: in the first, it is guaranteed that at least one valid pair witness-admissible set exists; in the second, all latent variables in the graph are set also as common parents also of X and Y, so no valid witness exists. We divide each variation into two subcases: in the first, "hard" subcase, parameters are chosen (by rejection sampling, proposing from the model described in the previous paragraph) so that NE1 has a bias of at least 0.1 in the population; in the second, no such a selection is enforced, and as such our exchangeable parameter sampling scheme makes the problem relatively easy. We summarize each WPP interval by the posterior expected value of the lower and upper bounds. In general WPP returns more than one bound: we select the upper/lower bound corresponding to the $(W, \mathbf{Z})$ pair which maximizes the score described at the end of Section 4.2. A BDeu prior with an equivalent sample size of 10 was used.

Our main evaluation metric for an estimate is the Euclidean distance (henceforth, "error") between the true ACE and the closed point in the given estimate, whether the estimate is a point or an interval. For methods that provide point estimates (NE1, NE2, and faithfulness), this means just the absolute value of the difference between the true ACE and the estimated ACE. For WPP, the error of the interval $[\mathcal{L}, \mathcal{U}]$ is zero if the true ACE lies in this interval. We report *error average* and *error tail mass at 0.1*, the latter meaning the proportion of cases where the error exceeds 0.1. Moreover, the faithfulness estimator is defined by averaging over all estimated ACEs as given by the accepted admissible sets in each problem.

As discussed in Section 8.1, WPP can be understood as providing a trade-off between information loss and accuracy. For instance, while the trivial interval $[-1, 1]$ will always have zero error, it is not an interesting solution. We assess the trade-off by running simulations at different levels of k_ϵ, where $\epsilon_w = \epsilon_y = \epsilon_x = k_\epsilon$. We also have two configurations for $\{\underline{\beta}, \overline{\beta}\}$: we set them at either $\underline{\beta} = \overline{\beta} = 1$ or $\underline{\beta} = 0.9, \overline{\beta} = 1.1$.

For the cases where no witness exists, Entner's Rule 1 should theoretically report no solution. Entner et al. (2013) used stringent thresholds for deciding when the two conditions

of Rule 1 held, we refer to that paper for an evaluation on how well Rule 1 can be correctly activated under the more conservative setup. Instead we take a more relaxed approach, using a uniform prior on the hypothesis of independence. As such, due to the nature of our parameter randomization, more often than not it will propose at least one witness. That is, for the problems where no exact solution exists, we assess how sensitive the methods are given conclusions taken from "approximate independencies" instead of exact ones.

The analytical bounds are combined with the numerical procedure as follows. We use the analytical bounds to test each proposed model using the rejection sampling criterion. Under this scheme, we calculate the posterior expected value of the contingency table and, using this single point, calculate the bounds using the fully numerical method. This is not guaranteed to work: the point estimator using the analytical bounds might lie outside the polytope given by the full set of constraints. If this situation is detected, we revert to calculating the bounds using the analytical method. The gains in interval length reduction using the full numerical method are relatively modest (e.g., at $k_\epsilon = 0.20$, the average interval width reduced from 0.30 to 0.24) but depending on the application they might make a sizeable difference.

We simulate 100 data sets for each one of the four cases (hard case/easy case, with theoretical solution/without theoretical solution), 5000 points per data set, 1000 Monte Carlo samples per decision. Results for the point estimators (NE1, NE2, faithfulness) are obtained using the population contingency tables. Results are summarized in Table 1. The first observation is that at very low levels of k_ϵ we increase the ability to reject all witness candidates: this is due mostly not because Rule 1 never fires, but because the falsification rule of WPP (which does not enforce independence constraints) rejects the proposed witnesses found by Rule 1. The trade-off set by WPP is quite stable, where larger intervals are indeed associated with smaller error. The point estimates vary in quality, being particularly bad in the situation where no witness should theoretically exist. The set-up where $\underline{\beta} = 0.9, \overline{\beta} = 1.1$ is especially uninformative compared to $\underline{\beta} = \overline{\beta} = 1$. At $k_\epsilon = 0.2$, we obtain interval widths around 0.50. As Manski (2007) emphasizes, this is the price for making fewer assumptions. Even there, they typically cover only about 25% of the interval $[-1, 1]$ of *a priori* possibilities for the ACE.

9.2.1 Selection of Relaxation Parameters

We performed an automated choice of relaxation parameters applying the methods in Section 8.2 to the same synthetic data sets. For each data set and each parameter choice method, we obtain a set B of intervals defined by a lower/upper bound. We summarize B in two ways: the *tightest* bound, meaning we choose the narrowest interval in B; the *loosest* bound, defined as the interval where the lower (upper) bound is the smallest lower (largest upper) bound in B. We then report results for each of the four synthetic case scenarios and each of the two methods: the Tightest ACE Coverage (TAC) method from Section 8.2.1 and the high posterior density (HPD) method of Section 8.2.2. Each parameter ϵ_w and $\epsilon_{xy} = \epsilon_x = \epsilon_y$ was allowed to assume values in the discretized grid $\{0.01, 0.05, 0.10, \ldots, 0.50\}$. Parameter $\beta = \overline{\beta} = 1/\underline{\beta}$ was allowed to take values in $\{1, 1.05, \ldots, 1.20\}$. Results are summarized in Table 2.

Hard, Solvable: NE1 = $(0.12, 1.00)$, NE2 = $(0.02, 0.03)$									
k_ϵ	Found	Faith.1		WPP1		Width1	WPP2		Width2
0.05	0.74	0.03	0.05	0.02	0.05	0.05	0.00	0.00	0.34
0.10	0.94	0.04	0.05	0.01	0.01	0.11	0.00	0.00	0.41
0.15	0.99	0.04	0.05	0.01	0.02	0.16	0.00	0.00	0.46
0.20	1.00	0.05	0.05	0.01	0.01	0.24	0.00	0.00	0.53
0.25	1.00	0.05	0.07	0.00	0.00	0.32	0.00	0.00	0.60
0.30	1.00	0.05	0.10	0.00	0.00	0.41	0.00	0.00	0.69
Easy, Solvable: NE1 = $(0.01, 0.01)$, NE2 = $(0.07, 0.24)$									
k_ϵ	Found	Faith.1		WPP1		Width1	WPP2		Width2
0.05	0.81	0.03	0.02	0.02	0.04	0.04	0.00	0.01	0.34
0.10	0.99	0.02	0.02	0.01	0.02	0.09	0.00	0.00	0.40
0.15	1.00	0.02	0.01	0.00	0.00	0.17	0.00	0.00	0.46
0.20	1.00	0.02	0.01	0.00	0.00	0.24	0.00	0.00	0.54
0.25	1.00	0.02	0.01	0.00	0.00	0.32	0.00	0.00	0.61
0.30	1.00	0.02	0.01	0.00	0.00	0.41	0.00	0.00	0.67
Hard, Not Solvable: NE1 = $(0.16, 1.00)$, NE2 = $(0.20, 0.88)$									
k_ϵ	Found	Faith.1		WPP1		Width1	WPP2		Width2
0.05	0.67	0.20	0.90	0.17	0.76	0.06	0.04	0.14	0.32
0.10	0.91	0.19	0.91	0.13	0.63	0.10	0.02	0.07	0.39
0.15	0.97	0.19	0.92	0.10	0.41	0.18	0.01	0.03	0.45
0.20	0.99	0.19	0.95	0.07	0.25	0.24	0.01	0.01	0.51
0.25	1.00	0.19	0.96	0.03	0.13	0.31	0.00	0.00	0.58
0.30	1.00	0.19	0.96	0.02	0.06	0.39	0.00	0.00	0.66
Easy, Not Solvable: NE1 = $(0.09, 0.32)$, NE2 = $(0.14, 0.56)$									
k_ϵ	Found	Faith.1		WPP1		Width1	WPP2		Width2
0.05	0.68	0.13	0.51	0.10	0.37	0.05	0.02	0.07	0.33
0.10	0.97	0.12	0.53	0.08	0.28	0.10	0.01	0.05	0.39
0.15	1.00	0.12	0.52	0.05	0.17	0.16	0.01	0.03	0.46
0.20	1.00	0.12	0.53	0.03	0.08	0.23	0.01	0.03	0.52
0.25	1.00	0.12	0.48	0.02	0.05	0.31	0.00	0.02	0.59
0.30	1.00	0.12	0.48	0.01	0.04	0.39	0.00	0.01	0.65

Table 1: Summary of the outcome of the synthetic studies. Columns labeled WPP1 refer to results obtained for $\underline{\beta} = \overline{\beta} = 1$, while WPP2 refers to the case $\underline{\beta} = 0.9, \overline{\beta} = 1.1$. The first column is the level in which we set the remaining parameters, $\epsilon_x = \epsilon_y = \epsilon_w = k_\epsilon$. The second column is the frequency by which a WPP solution has been found among 100 runs. For each particular method (NE1, NE2, Faithfulness and WPP) we report the pair (error average, error tail mass at 0.1), as explained in the main text. The Faithfulness estimator is the back-door adjustment obtained by using as the admissible set the same set found by WPP1. Averages are taken only over the cases where a witness-admissible set pair has been found. The columns following each WPP results are the median width of the respective WPP interval across the 100 runs.

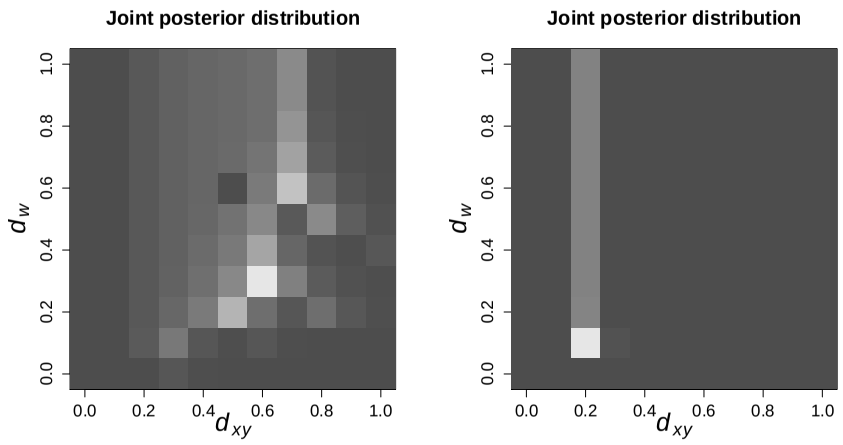
Case	Tightest				Loosest			
	TAC		HPD		TAC		HPD	
	error	width	error	width	error	width	error	width
Hard, Solvable	0.004	0.18	0.004	0.18	0.002	0.24	0.00009	0.40
Easy, Solvable	0.002	0.13	0.002	0.13	0.001	0.20	0.002	0.26
Hard, Not Solvable	0.12	0.14	0.12	0.14	0.10	0.20	0.07	0.33
Easy, Not Solvable	0.07	0.14	0.07	0.14	0.05	0.20	0.04	0.35

Table 2: Applying the criteria for choosing relaxation parameters from Section 8.2 to the four synthetic case scenarios. "Error" is the average error, as formalized for Table 1. "Width" is the average width over all 100 subcases of the respective study. "Tightest" and "Loosest" are the two criteria for summarizing a set of intervals, as explained in the main text.

Comparing it against Table 1, results seem to be slightly worse than WPP1 at the same interval width, but without making prior assumptions on β. Compared to WPP2, overall widths are much smaller. The HPD method agrees with TAC on the tightest interval, as our choice of prior will always imply a posterior mode on the TAC solution. The loosest interval for HPD will always be larger or equal to the loosest in TAC, as the 95% posterior mass that generates intervals will include the Pareto frontier and possibly many other candidates. In our simulations, the reduction in error for HPD with the loosest bound came with a non-trivial increase on the length of the corresponding intervals. While we do not explicitly advocate one method over another, the HPD method can be used to classify problems as harder than others by assessing how much of the posterior mass of hyperparameters is not on the Pareto frontier. In Figure 5, we visualize the marginal posterior distribution of $\{d_{xy}, d_w\}$ for two synthetic problems in the easy/solvable case, where in one problem the tightest interval failed to cover the true ACE, while in the other the ACE was correctly accounted for.

9.3 Influenza Study

Our empirical study concerns the effect of influenza vaccination on a patient being later on hospitalized with chest problems. $X = 1$ means the patient got a flu shot, $Y = 1$ indicates the patient was hospitalized. A negative ACE therefore suggests a desirable vaccine. The study was originally discussed by McDonald et al. (1992). Shots were not randomized, but doctors were randomly assigned to receive a reminder letter to encourage their patients to be inoculated, an event recorded as binary variable *GRP*. This suggests the standard IV model in Figure 1(d), with $W = GRP$ and U unobservable. That is, W and U are independent because W is randomized, and there are reasonable justifications to believe the lack of a direct effect of letter randomization on patient hospitalization. Richardson et al. (2011) and Hirano et al. (2000) provide further discussion.



(a) (b)

Figure 5: Marginal posterior distribution for $\{d_{xy}, d_w\}$ in two problems instances. Darker values represent smaller probabilities. In instance (a), the length of the tightest interval was 0.32 and did not contain the true ACE (but the error was still < 0.01). In instance (b), the length of the tightest interval was 0.11 and did contain the true ACE. Parameter d_w does not seem to be as influential conditional on d_{xy}, and the uniform prior allows for little variability in the d_w posterior away from the mode.

From this randomization, it is possible to directly estimate the ACE[24] of W on Y: -0.01. This is called *intention-to-treat* (ITT) analysis (Rothman et al., 2008), as it is based on the treatment assigned by randomization and not on the variable of interest (X), which is not randomized. While the ITT can be used for policy making, the ACE of X on Y would be a more relevant result, as it reveals features of the vaccine that are not dependent on the encouragement design. X and Y can be confounded, as X is not controlled. For instance, the patient choice of going to be vaccinated might be caused by her general health status, which will be a factor for hospitalization in the future.

The data contains records of 2,681 patients, with some demographic indicators (age, sex and race) and some historical medical data (for instance, whether the patient is diabetic). A total of 9 covariates is available. Using the bounds of Balke and Pearl (1997) and observed

24. Notice that while the ACE might be small, this does not mean that in another scale, such as odd-ratios, the results do not reveal an important effect. This depends on the domain.

frequencies produces an interval of $[-0.23, 0.64]$ for the ACE. WPP could *not* validate *GRP* as a witness for any admissible set.

Instead, when forbidding *GRP* to be included in an admissible set (since the theory says *GRP* cannot be a common direct cause of vaccination and hospitalization), WPP selected as the highest-scoring pair the witness *DM* (patient had history of diabetes prior to vaccination) with admissible set composed of *AGE* (dichotomized as "60 or less years old," and "above 60") and *SEX*. Choosing, as an illustration, $\epsilon_w = \epsilon_y = \epsilon_x = 0.2$ and $\underline{\beta} = 0.9$, $\overline{\beta} = 1.1$, we obtain the posterior expected interval $[-0.10, 0.17]$. This does *not* mean the vaccine is more likely to be bad (positive ACE) than good: the posterior distribution is over bounds, not over points, being completely agnostic about the distribution within the bounds. Notice that even though we allow for full dependence between all of our variables, the bounds are stricter than in the standard IV model due to the weakening of hidden confounder effects postulated by observing conditional independencies. It is also interesting that two demographic variables ended up being chosen by Rule 1, instead of other indicators of past diseases.

When allowing *GRP* to be included in an admissible set, the pair (*DM*, {*AGE*, *SEX*}) is now ranked second among all pairs that satisfy Rule 1, with the first place being given by *RENAL* as the witness (history of renal complications), with the admissible set being *GRP, COPD* (history of pulmonary disease), and *SEX*. In this case, the expected posterior interval was approximately the same, $[-0.08, 0.16]$. It is worthwhile to mention that, even though this pair scored highest by our criterion that measures the posterior probability distribution of each premise of Rule 1, it is clear that the fit of this model is not as good as the one with *DM* as the witness, as measured by the much larger proportion of rejected samples when generating the posterior distribution. This suggests future work on how to rank such models.

In Figure 6 we show a scatter plot of the posterior distribution over lower and upper bounds on the influenza vaccination, where *DM* is the witness. In Figure 7(a) and (b) we show kernel density estimators based on the Monte Carlo samples for the cases where *DM* and *RENAL* are the witnesses, respectively. While the witnesses were tested using the analytical bounds, the final set of samples shown here were generated with the fully numerical optimization procedure, which is quite expensive.

We also analyze how to select $\aleph = \{\epsilon_w, \epsilon_{xy} = \epsilon_x = \epsilon_y, \beta = \overline{\beta} = 1/\underline{\beta}\}$ using the Tightest ACE Coverage (TAC) method of Section 8.2.1. The motivation is that this is a domain with overall weak dependencies among variables. From one point of view, this is bad as instruments will be weak and generate wide intervals (as suggested by Proposition 1). From another perspective, this suggests that the effect of hidden confounders may also be weak.

A total of 48 witness/admissible sets were proposed by WPP via Rule 1. The TAC Pareto frontier, using the same parameter space as in Section 9.2.1, included only two possibilities, $\epsilon_{xy} = 0.05, \epsilon_w = 0.01, \beta = 1$ and $\epsilon_{xy} = 0.01, \epsilon_w = 0.01, \beta = 1.05$. Using the empirical distribution as an estimator of the joint of the observables, the respective ACE intervals were $[-0.01, 0.01]$ and $[-0.02, 0.02]$. Although the sign of the ACE is not determined from the data, the WPP procedure suggests that the magnitude of the ACE is no greater than 0.02, which by itself is of interest.

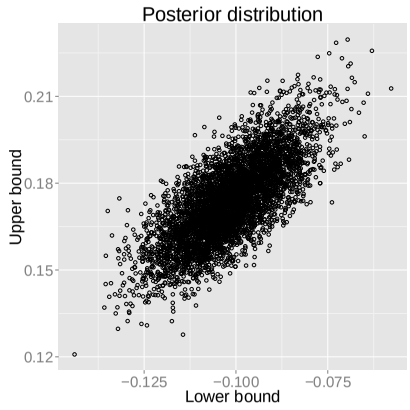


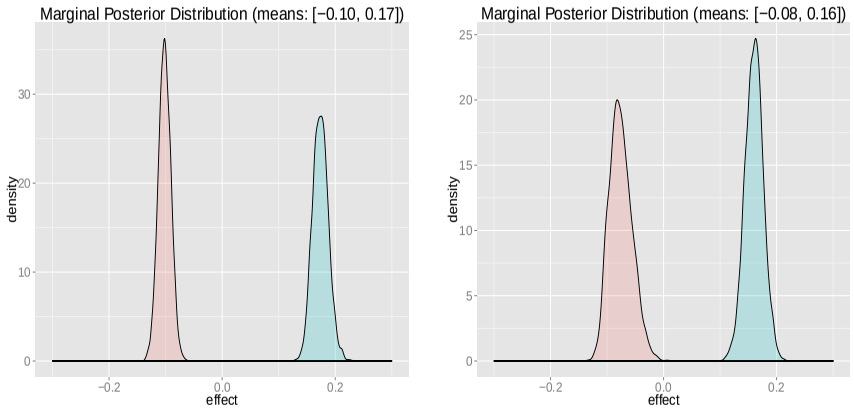
Figure 6: Scatterplot of the joint posterior distribution of lower bounds and upper bounds, Pearson correlation coefficient of 0.71.

9.4 Linear Models

In this section, we assess the usage of the method in the linear case. Independently, we also introduce a complementary way of summarizing the outcome of a WPP analysis.

We first check the performance of the method with a small synthetic study. We generate models following the same pattern of the "hard/solvable" case of Section 9.2, the ACE being the coefficient of $X^\star$ in the equation for $Y^\star$. Each conditional model for a variable $V_i^\star$ given its parents is generated by sampling its coefficients from independent standard Gaussians, sampling the variance of the error term from a uniform $[0, 0.5]$, then rescaling the coefficients such that the marginal variance of $V_i^\star$ is 1. Observable data is then generated by transforming each $V_i^\star$ to follow a gamma distribution with mean and variance equal to 2. We generated 100 data sets with a sample size of 1000 each. We perform experiments[25] setting all hyperparameters $\epsilon_c = \epsilon_{wx} = \epsilon_{wy} = \epsilon_{xy} = 0.2$ and $\epsilon_c = \epsilon_{wx} = \epsilon_{wy} = \epsilon_{xy} = 0.1$. Estimates of the Gaussian copula correlation matrices are obtained using function HUGE.NPN from the R package HUGE to transform the data, of which we compute the empirical correlation matrix. We obtained average errors of 0.04 for the method with parameters set at 0.2, and 0.07 for parameters set at 0.1. The average length of the proposed intervals were 0.5 and 0.26, respectively. For comparison, the population error for the two naïve estimators was 0.23 and 0.18.

25. The test for conditional independencies is done with the corrected BGe score (Kuipers et al., 2014) as discussed in Section 6.2. The hyperparameters are a prior of 0.5 for the independence constraint hypothesis, and a inverse Wishart prior with $\upsilon \equiv p + 2$ degrees of freedom and a scale matrix given by the $p \times p$ identity matrix multiplied by υ, where p is the number of variables in the test.



(a) (b)

Figure 7: In (a), the marginal densities for the lower bound (red) and upper bound (blue) on the ACE, smoothed kernel density estimates based on 5000 Monte Carlo samples. Bounds were derived using DM as the witness. In (b), a similar plot using $RENAL$ as the witness.

We performed an empirical study with the 1976 Panel Study of Income Dynamics. The study uses data from 1975, assessing incoming of couples in 1976. Our outcome variable Y is the wife's reported wage at the time of the 1976 interview, and the treatment X is the number of years of the wife's previous labor market experience. The data was discussed by Mroz (1987) and can be obtained from the R package AER (Kleiber and Zeileis, 2008). Covariate set $\mathbf{W}$ includes a combination of discrete and continuous variables, such as husband's wages, number of children, and whether the wife went to college. We infer a Gaussian copula correlation matrix using the extended rank likelihood method of Hoff (2007) with the R package sbgcop, which can deal with discrete and continuous variables but requires expensive MCMC sampling. Notice that conditional independencies among the discrete observable elements of $\mathbf{V} \equiv \{X, Y\} \cup \mathbf{W}$ do not follow from conditional independencies among the unobservable Gaussian variables $\mathbf{V}^\star$. We nevertheless test Rule 1 among $\mathbf{V}^\star$ using the estimated copula correlation matrix and the relatively high prior of 0.5 for the hypothesis of independence, for any given independence assessment. Sample size is 753, with 17 covariates[26]. We then make the (strong) assumption that work experience in 1975 is not a cause of any other variable in the covariate set.

26. We removed two covariates from the original data set: the indicator of economical participation, which is a deterministic function of other covariates; and the estimated wage of the wife in 1975, which for that year was not available directly via self-report. In order to speed up the search algorithm, for each witness candidate W, the space of variables to test for an admissible set is composed of the 10 covariates mostly

Setting all relaxation parameters $\epsilon_c = \epsilon_{wx} = \epsilon_{wy} = \epsilon_{xy}$ to 0.1, we obtain in Figure 8(a) all corresponding intervals, with black dots representing the corresponding estimated ACE for the chosen admissible set. An explanation of all variables can be found in the documentation of package AER (Kleiber and Zeileis, 2008). Recall that the units here are given in the latent Gaussian space, where each $V_i^\star$ is a non-linear transformation of the corresponding V_i, as explained in Section 6.2. This analysis reveals two clear clusters of behavior, which internally show little variability but are very different from one another, even accounting for a violation of 0.1. This illustrates possible ways of communicating the output of a WPP analysis so that issues with assumptions and data can be raised.

In this case, the two clusters of intervals differ in one variable in the admissible set: variable *HOURS* is present in cases where the intervals are closer to zero. This variable measures the number of work hours of the wife in 1975, and is partially embedded in the definition of the experience level measured at 1975. By removing all admissible sets that include the *HOURS* variable, we obtain the summary given as Figure 8(b). This type of visualization step can be used to flag major contradictions that cannot be easily explained by allowing mild violations of faithfulness, but which might suggest problematic measurements to be reconsidered in the analysis.

10. Conclusion

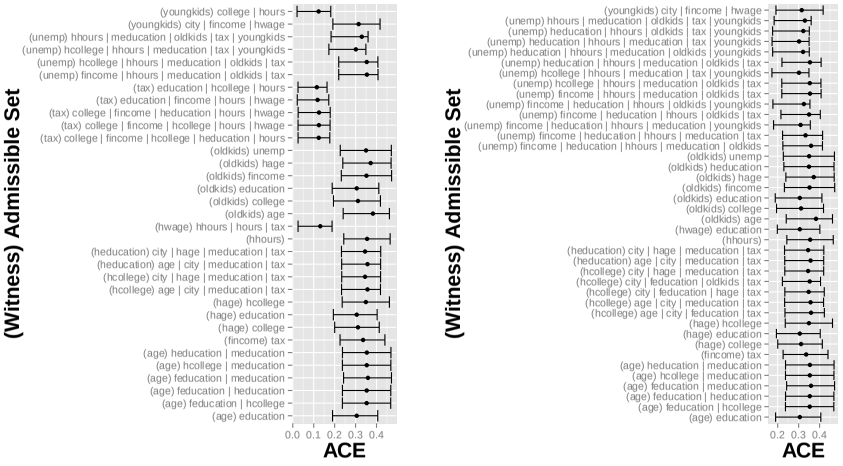
Our model provides a novel compromise between point estimators given by the faithfulness assumption and bounds based on instrumental variables. We believe such an approach should become a standard item in the toolbox of methodologies for observational studies, as it provides means to draw conclusions from a complementary set of assumptions. Ongoing updates of software for WPP is provided as part of the R package CausalFX, available at the Comprehensive R Network[27] and GitHub[28]. A snapshot of the code used in this paper is available at `http://www.homepages.ucl.ac.uk/~ucgtrbd/wpp`.

In particular, unlike Bayesian approaches that put priors directly on the parameters of the unidentifiable latent variable model $P(Y, X, W, U \mid \mathbf{Z})$, the constrained Dirichlet prior on the observed distribution does not suffer from massive sensitivity to the choice of hyperparameters. When a strongly informative prior is lacking, WPP keeps inference more honest by focusing on bounds. While it is tempting to look for an alternative that will provide a point estimate of the ACE, it is also important to have a method that trades-off information for fewer assumptions. WPP provides a framework to express such assumptions.

The brute-force search used in the implementation of Rule 1 can be substituted by other combinatorial search procedures and dimensionality reduction methods. Entner et al. (2013) provide alternatives by borrowing ideas from the PC algorithm, for instance. Package CausalFX implements the idea discussed briefly in Section 9.4, where for each witness candidate W we pre-select a small set of candidates from $\mathbf{W}\backslash\{W\}$ and perform a brute-force search for admissible sets within this candidate set only. Pre-selection in CausalFX 1.0 is done by first sorting all $Z \in \mathbf{W}\backslash\{W\}$ according to the empirical mutual information

strongly associated with W, measured by the absolute value of the corresponding copula correlation matrix entry.

27. `https://cran.r-project.org/web/packages/CausalFX/index.html`
28. `https://github.com/rbas2015/CausalFX`



(a) (b)

Figure 8: The diagrams depict some ACE intervals obtained for the linear model of the impact of work experience up to 1975 of a married woman into her salary in 1976. On the y-axis, we show the witness in brackets, followed by all variables in the admissible set; the x-axis shows the point estimates of the interval for the ACE using $\epsilon_c = \epsilon_{wx} = \epsilon_{wy} = \epsilon_{xy} = 0.1$. Black dots are the corresponding point estimates of the ACE using the back-door method. All variable names are explained in the documentation of package AER (Kleiber and Zeileis, 2008). In (a), we allow all other recorded variables into the covariate set $\mathbf{W}$ from which witnesses and admissible sets are generated. In (b), we remove *HOURS* from the pool of possible covariates.

of Z and W given X and then picking the top K candidates, in descending value of mutual information (the heuristic being that we should look first at paths $W \to X \leftarrow Z$ that are "strong"). K is chosen such that enumerating 2^K candidate admissible sets is possible within the available computer resources. Although this restricted search procedure might miss some admissible sets, it has the advantage of avoiding sensitivity to propagation of statistical mistakes that creates difficulties for the PC algorithm and similar methods.

We emphasize that the credible intervals obtained by the procedure are conditioned on the search results, discarding uncertainty coming from the choices of witnesses, admissible sets and relaxation parameters. Ideally, uncertainty concerning the outcome of the Rule 1 search should also be taken into account. An approach analogous to (Friedman and Koller, 2003) is necessary, which we leave as future work.

As further future work, we will look at a generalization of the procedure beyond relaxations of chain structures $W \to X \to Y$. Much of the machinery here developed, including Entner et al.'s Rules, can be adapted to the case where causal ordering is unknown: starting from the algorithm of Mani et al. (2006) to search for "Y-structures," it is possible to generalize Rule 1 to setups where we have an outcome variable Y that needs to be controlled, but where there is no covariate X known not to be a cause of other covariates. Mooij and Cremers (2015) investigate the robustness of the faithfulness condition in this setup. Finally, the techniques used to derive the symbolic bounds in Section 5 may prove useful in a more general context, and complement other methods to find subsets of useful constraints such as the graphical approach of Evans (2012).

Acknowledgments

We thank McDonald, Hiu and Tierney for their flu vaccine data and the anonymous reviewers for their many suggestions to improve our paper. Much of this work was done while RS was hosted by the Department of Statistics at the University of Oxford. Parts of this work were previously published in the 2014 Neural Information Processing Systems Conference (Silva and Evans, 2014). Section 6, 7 and 8 are completely new, and all remaining sections have new material added, including the appendix.

Appendix A. Proofs

In this Appendix, we prove the results mentioned in the main text.

A.1 Basic Results

We divide the proofs in four main sections. The first section provides the basic methods, including how classical results in instrumental variable bounding can be rederived. The second and third sections are proofs for the most complex types of bounds. Finally, the fourth section covers the linear continuous case.

Proof of Proposition 1 In the standard IV case, simple analytical bounds are known for $P(Y = y \mid do(X = x))$ (Balke and Pearl, 1997; Dawid, 2003):

$$\eta_0 \leq \min \begin{cases} 1 - \zeta_{00.0} \\ 1 - \zeta_{00.1} \\ \zeta_{01.0} + \zeta_{10.0} + \zeta_{10.1} + \zeta_{11.1} \\ \zeta_{10.0} + \zeta_{11.0} + \zeta_{01.1} + \zeta_{10.1} \end{cases} \qquad \eta_0 \geq \max \begin{cases} \zeta_{10.1} \\ \zeta_{10.0} \\ \zeta_{10.0} + \zeta_{11.0} - \zeta_{00.1} - \zeta_{11.1} \\ -\zeta_{00.0} - \zeta_{11.0} + \zeta_{10.1} + \zeta_{11.1} \end{cases}$$

$$\eta_1 \leq \min \begin{cases} 1 - \zeta_{01.1} \\ 1 - \zeta_{01.0} \\ \zeta_{10.0} + \zeta_{11.0} + \zeta_{00.1} + \zeta_{11.1} \\ \zeta_{00.0} + \zeta_{11.0} + \zeta_{10.1} + \zeta_{11.1} \end{cases} \qquad \eta_1 \geq \max \begin{cases} \zeta_{11.1} \\ \zeta_{11.0} \\ -\zeta_{01.0} - \zeta_{10.0} + \zeta_{10.1} + \zeta_{11.1} \\ \zeta_{10.0} + \zeta_{11.0} - \zeta_{01.1} - \zeta_{10.1} \end{cases}$$

where $\eta_x \equiv P(Y=1 \mid do(X=x))$ and $\zeta_{yx.w} \equiv P(Y=y, X=x \mid W=w)$. Define also $\alpha_x \equiv P(Y=1 \mid X=x)$ and $\beta_w \equiv P(X=1 \mid W=w)$ so that

$$\zeta_{yx.w} = \alpha_x^{I(y=1)}(1-\alpha_x)^{I(y=0)}\beta_w^{I(x=1)}(1-\beta_w)^{I(x=0)}, \tag{35}$$

where $I(\cdot)$ is the indicator function returning 1 or 0 depending on whether its argument is true or false, respectively.

Assume for now that $\beta_1 \geq \beta_0$, that is, $P(X=1 \mid W=1) \geq P(X=1 \mid W=0)$. We will first show that $1-\zeta_{00.0} \leq \min\{1-\zeta_{00.1}, \zeta_{01.0}+\zeta_{10.0}+\zeta_{10.1}+\zeta_{11.1}, \zeta_{10.0}+\zeta_{11.0}+\zeta_{01.1}+\zeta_{10.1}\}$.

That $1-\zeta_{00.0} \leq 1-\zeta_{00.1}$ follows directly from the relationship (35) and the assumptions $W \perp\!\!\!\perp Y \mid X$ and $\beta_1 \geq \beta_0$: $(1-\zeta_{00.0})-(1-\zeta_{00.1}) = -(1-\alpha_0)(1-\beta_0)+(1-\alpha_0)(1-\beta_1) = (1-\alpha_0)(\beta_0-\beta_1) \leq 0$.

Now consider $(1-\zeta_{00.0})-(\zeta_{01.0}+\zeta_{10.0}+\zeta_{10.1}+\zeta_{11.1})$. This is equal to

$$\begin{aligned}
&= (1-(1-\alpha_0)(1-\beta_0)) - ((1-\alpha_1)\beta_0 + \alpha_0(1-\beta_0) + \alpha_0(1-\beta_1) + \alpha_1\beta_1) \\
&= (\beta_0 + \alpha_0(1-\beta_0)) - (\beta_0 - \alpha_1\beta_0 + \alpha_0(1-\beta_0) + \alpha_0 - \alpha_0\beta_1 + \alpha_1\beta_1) \\
&= \alpha_1(\beta_0-\beta_1) - \alpha_0(1-\beta_1) \leq 0
\end{aligned}$$

Analogously, we can show that $1-\zeta_{00.0} \leq \zeta_{10.0}+\zeta_{11.0}-\zeta_{01.1}-\zeta_{10.1}$. Tedious but analogous manipulations lead to the overall conclusion

$$1-\zeta_{00.0} = \min\begin{cases} 1-\zeta_{00.0} \\ 1-\zeta_{00.1} \\ \zeta_{01.0}+\zeta_{10.0}+\zeta_{10.1}+\zeta_{11.1} \\ \zeta_{10.0}+\zeta_{11.0}+\zeta_{01.1}+\zeta_{10.1} \end{cases} \qquad \zeta_{10.0} = \max\begin{cases} \zeta_{10.1} \\ \zeta_{10.0} \\ \zeta_{10.0}+\zeta_{11.0}-\zeta_{00.1}-\zeta_{11.1} \\ -\zeta_{00.0}-\zeta_{11.0}+\zeta_{10.1}+\zeta_{11.1} \end{cases}$$

$$1-\zeta_{01.1} = \min\begin{cases} 1-\zeta_{01.1} \\ 1-\zeta_{01.0} \\ \zeta_{10.0}+\zeta_{11.0}+\zeta_{00.1}+\zeta_{11.1} \\ \zeta_{00.0}+\zeta_{11.0}+\zeta_{10.1}+\zeta_{11.1} \end{cases} \qquad \zeta_{11.1} = \max\begin{cases} \zeta_{11.1} \\ \zeta_{11.0} \\ -\zeta_{01.0}-\zeta_{10.0}+\zeta_{10.1}+\zeta_{11.1} \\ \zeta_{10.0}+\zeta_{11.0}-\zeta_{01.1}-\zeta_{10.1} \end{cases}$$

The upper bound on the ACE $\eta_1-\eta_0$ is obtained by subtracting the lower bound on η_0 from the upper bound on η_1. That is, $\eta_1-\eta_0 \leq (1-\zeta_{01.1})-\zeta_{10.0} = \mathcal{U}_{SIV}$. Similarly, $\eta_1-\eta_0 \geq \zeta_{11.1}-(1-\zeta_{00.0}) = \mathcal{L}_{SIV}$. It follows that $\mathcal{U}_{SIV}-\mathcal{L}_{SIV} = 1-(P(X=1 \mid W=1)-P(X=1 \mid W=0))$.

Finally, assuming $\beta_1 \leq \beta_0$ gives by symmetry the interval width $1-(P(X=1 \mid W=0)-P(X=1 \mid W=1))$, implying the width in the general case is given by $1-|P(X=1 \mid W=1)-P(X=1 \mid W=0)|$. ■

Now we will prove the main theorems stated in Section 5. To facilitate reading, we repeat here the notation used in the description of the constraints with a few additions, as well as the identities mapping different parameter spaces and the corresponding assumptions exploited in the derivation.

We start with the basic notation,

$$\begin{aligned}
\zeta^\star_{yx.w} &\equiv P(Y=y, X=x \mid W=w, U) \\
\zeta_{yx.w} &\equiv \textstyle\sum_U P(Y=y, X=x \mid W=w, U)P(U \mid W=w) \\
&= P(Y=y, X=x \mid W=w) \\
\kappa_{yx.w} &\equiv \textstyle\sum_U P(Y=y, X=x \mid W=w, U)P(U)
\end{aligned}$$

$$\begin{aligned}
\eta^\star_{xw} &\equiv P(Y=1 \mid X=x, W=w, U) \\
\eta_{xw} &\equiv \textstyle\sum_U P(Y=1 \mid X=x, W=w, U)P(U \mid W=w) \\
&= P(Y=1 \mid do(X=x), W=w) \\
\omega_{xw} &\equiv \textstyle\sum_U P(Y=1 \mid X=x, W=w, U)P(U)
\end{aligned}$$

$$\begin{aligned}
\delta^\star_w &\equiv P(X=1 \mid W=w, U) \\
\delta_w &\equiv \textstyle\sum_U P(X=1 \mid W=w, U)P(U \mid W) = P(X=1 \mid W=w) \\
&= \zeta_{11.w} + \zeta_{01.w} \\
\chi_{x.w} &\equiv \textstyle\sum_U P(X=x \mid W=w, U)P(U) \\
&= \kappa_{1x.w} + \kappa_{0x.w}
\end{aligned}$$

The explicit relationship between parameters describing the latent variable model is:

$$\begin{aligned}
\zeta^\star_{00.0} &= (1-\eta^\star_{00})(1-\delta^\star_0) \\
\zeta^\star_{01.0} &= (1-\eta^\star_{10})\delta^\star_0 \\
\zeta^\star_{10.0} &= \eta^\star_{00}(1-\delta^\star_0) \\
\zeta^\star_{11.0} &= \eta^\star_{10}\delta^\star_0 \\
\zeta^\star_{00.1} &= (1-\eta^\star_{01})(1-\delta^\star_1) \\
\zeta^\star_{01.1} &= (1-\eta^\star_{11})\delta^\star_1 \\
\zeta^\star_{10.1} &= \eta^\star_{01}(1-\delta^\star_1) \\
\zeta^\star_{11.1} &= \eta^\star_{11}\delta^\star_1
\end{aligned}$$

All upper bound constants $U^{YU}_{\cdot\cdot}$ are assumed to be positive. For $L^{YU}_{\cdot\cdot}=0$, $c \geq 0$, all ratios $c/L^{YU}_{\cdot\cdot}$ are defined to be positive infinite.

In what follows, we define "the standard IV model" as the one which obeys exogeneity of W and exclusion restriction—that is, the model following the directed acyclic graph $\{W \to X \to Y, X \leftarrow U \to Y\}$. All variables are binary, and the goal is to bound the average causal effect (ACE) of X on Y given a non-descendant W and a possible (set of) confounder(s) U of X and Y.

Proof of Theorem 2 Start with the relationship between η_{xw} and its upper bound:

$$\begin{aligned}
\eta^\star_{xw} &\leq U^{YU}_{xw} && \text{(Multiply both sides by } \delta^\star_{x'.w}\text{)} \\
\eta^\star_{xw}(1-(1-\delta^\star_{x'.w})) &\leq U^{YU}_{xw}\delta^\star_{x'.w} && \text{(Marginalize over } P(U)\text{)} \\
\omega_{xw} - \kappa_{1x.w} &\leq U^{YU}_{xw}\chi_{x'.w} \\
\omega_{xw} &\leq \kappa_{1x.w} + U^{YU}_{xw}(\kappa_{0x'.w}+\kappa_{1x'.w})
\end{aligned}$$

and an analogous series of steps gives $\omega_{xw} \geq \kappa_{1x.w} + L^{YU}_{xw}(\kappa_{0x'.w}+\kappa_{1x'.w})$. Notice such bounds above will depend on how tight ϵ_y is. As an illustration of its implications, consider

the derived identity $\zeta^\star_{0x.w} = (1-\eta^\star_{xw})\delta^\star_{x.w} \Rightarrow 1-\eta^\star_{xw} = \zeta^\star_{0x.w}/\delta^\star_{x.w} \Rightarrow 1-\eta^\star_{xw} \geq \zeta^\star_{0x.w} \Rightarrow \eta^\star_{xw} \leq 1-\zeta^\star_{0x.w} = \zeta^\star_{0x.w} + \zeta^\star_{0x'.w} + \zeta^\star_{1x'.w} \Rightarrow \omega_{xw} \leq \kappa_{0x.w} + \kappa_{0x'.w} + \kappa_{1x'.w}$.

It follows from $U^{YU}_{xw} \leq 1$ that that the derived bound $\omega_{xw} \leq \kappa_{1x.w} + U^{YU}_{xw}(\kappa_{0x'.w} + \kappa_{1x'.w})$ is at least as tight as the one obtained via $\eta^\star_{xw} \leq 1-\zeta^\star_{0x.w}$. Notice also that the standard IV bound $\eta_{xw} \leq 1-\zeta_{0x.w}$ (Balke and Pearl, 1997; Dawid, 2003) is a special case for $\epsilon_y = 0$, $\underline{\beta} = \overline{\beta} = 1$.

For the next bounds, consider

$$\begin{array}{rcll} \delta^\star_{x.w} & \leq & U^{XU}_{xw} & \\ \eta^\star_{xw}\delta^\star_{x.w} & \leq & U^{XU}_{xw}\eta^\star_{xw} & \text{(Marginalize over } P(U)) \\ \kappa_{1x.w} & \leq & U^{XU}_{xw}\omega_{xw} & \\ \omega_{xw} & \geq & \kappa_{1x.w}/U^{XU}_{xw} & \end{array}$$

where the bound $\omega_{xw} \leq \kappa_{1x.w}/L^{XU}_{xw}$ can be obtained analogously. The corresponding bound for the standard IV model (with possible direct effect $W \to Y$) is $\eta_{xw} \geq \zeta_{1x.w}$, obtained again by choosing $\epsilon_x = 1$, $\underline{\beta} = \overline{\beta} = 1$. The corresponding bound $\omega_{xw} \geq \kappa_{1x.w}$ is a looser bound for $U^{XU}_{xw} < 1$. Notice that if $L^{XU}_{xw} = 0$, the upper bound is defined as infinite.

Finally, the last bounds are similar to the initial ones, but as a function of ϵ_x instead of ϵ_y:

$$\begin{array}{rcll} \delta^\star_{x.w} & \leq & U^{XU}_{xw} & \\ (1-\eta^\star_{xw})\delta^\star_{x.w} & \leq & U^{XU}_{xw}(1-\eta^\star_{xw}) & \text{(Marginalize over } P(U)) \\ \kappa_{0x.w} & \leq & U^{XU}_{xw}(1-\omega_{xw}) & \\ \omega_{xw} & \leq & 1-\kappa_{0x.w}/U^{XU}_{xw} & \end{array}$$

The lower bound $\omega_{xw} \geq 1-\kappa_{0x.w}/L^{XU}_{xw}$ is obtained analogously, and implied to be minus infinite if $L^{XU}_{xw} = 0$. ■

Proof of Theorem 3 We start with the following derivation,

$$\begin{array}{rcll} \eta^\star_{xw'} - \eta^\star_{xw} & \leq & \epsilon_w & \\ \eta^\star_{xw'}\delta^\star_{x.w'} - \eta^\star_{xw}\delta^\star_{x.w'} & \leq & \epsilon_w\delta^\star_{x.w'} & \text{(Use } -U^{XU}_{xw'} \leq -\delta^\star_{x.w'}) \\ \eta^\star_{xw'}\delta^\star_{x.w'} - \eta^\star_{xw}U^{XU}_{xw'} & \leq & \epsilon_w\delta^\star_{x.w'} & \text{(Marginalize over } P(U)) \\ \kappa_{1x.w'} - \omega_{xw}U^{XI}_{xw} & \leq & \epsilon_w\chi_{x.w'} & \\ \omega_{xw} & \geq & (\kappa_{1x.w'} - \epsilon_w\chi_{x.w'})/U^{XU}_{xw'} & \\ \omega_{xw} & \geq & (\kappa_{1x.w'} - \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/U^{XU}_{xw'} & \end{array}$$

Analogously, starting from $\eta^\star_{xw'} - \eta^\star_{xw} \geq \epsilon_w$, we obtain $\omega_{xw} \leq (\kappa_{1x.w'} + \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/L^{XU}_{xw'}$. Notice that for the special case ϵ_w and $U^{XU}_{xw'} = 1$, we obtain the corresponding lower bound $\omega_{xw} \geq \kappa_{1x.w'}$ that relates ω and κ across different values of W.

The result corresponding to the upper bound $\eta_{xw} \leq 1-\zeta_{0x.w'}$ can be obtained as follows:

$$\begin{array}{rcll} \eta^\star_{xw'} - \eta^\star_{xw} & \geq & -\epsilon_w & \\ 1+\eta^\star_{xw'} - 1 - \eta^\star_{xw} & \geq & -\epsilon_w & \\ (1-\eta^\star_{xw}) - (1-\eta^\star_{xw'}) & \geq & -\epsilon_w & \\ (1-\eta^\star_{xw})\delta^\star_{x.w'} - (1-\eta^\star_{xw'})\delta^\star_{x.w'} & \geq & -\epsilon_w\delta^\star_{x.w'} & \\ (1-\eta^\star_{xw})U^{XU}_{xw'} - (1-\eta^\star_{xw'})\delta^\star_{x.w'} & \geq & -\epsilon_w\delta^\star_{x.w'} & \text{(Marginalize over } P(U)) \\ (1-\omega_{xw})U^{XU}_{xw'} - \kappa_{0x.w'} & \geq & -\epsilon_w\chi_{x.w'} & \\ \omega_{xw} & \leq & 1-(\kappa_{0x.w'} - \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/U^{XU}_{xw'} & \end{array}$$

with the corresponding lower bound (non-trivial for $L^{XU}_{xw'} > 0$) given by $\omega^\star_{xw} \geq 1-(\kappa_{0x.w'} + \epsilon_w(\kappa_{0x.w'} + \kappa_{1x.w'}))/L^{XU}_{xw'}$.

The final block of relationships can be derived as follows:

$$\begin{array}{rcll} \eta^\star_{xw} - \eta^\star_{xw'} & \leq & \epsilon_w & \\ \eta^\star_{xw}\delta^\star_{x'.w} - \eta^\star_{xw'}\delta^\star_{x'.w} & \leq & \epsilon_w\delta^\star_{x'.w} & \\ \eta^\star_{xw}(1-(1-\delta^\star_{x'.w})) - \eta^\star_{xw'}\delta^\star_{x'.w} & \leq & \epsilon_w\delta^\star_{x'.w} & \text{(Use } -U^{XU}_{x'w} \leq -\delta^\star_{x'.w}) \\ \eta^\star_{xw} - \eta^\star_{xw}(1-\delta^\star_{x'.w}) - \eta^\star_{xw'}U^{XU}_{x'w} & \leq & \epsilon_w\delta^\star_{x'.w} & \text{(Marginalize over } P(U)) \\ \omega_{xw} - \kappa_{1x.w} - \omega_{xw'}U^{XU}_{x'w} & \leq & \epsilon_w\chi_{x'.w} & \\ \omega_{xw} - \omega_{xw'}U^{XU}_{x'w} & \leq & \kappa_{1x.w} + \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w}) & \end{array}$$

with the lower bound $\omega_{xw} - \omega_{xw'}L^{XU}_{x'w} \geq \kappa_{1x.w} - \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w})$ derived analogously. Moreover,

$$\begin{array}{rcl} \eta^\star_{xw'} - \eta^\star_{xw} & \leq & \epsilon_w \\ (1-\eta^\star_{xw})\delta^\star_{x'.w} - (1-\eta^\star_{xw'})\delta^\star_{x'.w} & \leq & \epsilon_w\delta^\star_{x'.w} \\ (1-\eta^\star_{xw})(1-(1-\delta^\star_{x'.w})) - (1-\eta^\star_{xw'})U^{XU}_{x'w} & \leq & \epsilon_w\delta^\star_{x'.w} \\ 1-\omega_{xw} - \kappa_{0x.w} - (1-\omega_{xw'})U^{XU}_{x'w} & \leq & \epsilon_w\chi_{x'.w} \\ \omega_{xw} - \omega_{xw'}U^{XU}_{x'w} & \geq & 1-\kappa_{0x.w} - U^{XU}_{x'w} - \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w}) \end{array}$$

and the corresponding $\omega_{xw} - \omega_{xw'}L^{XU}_{x'w} \leq 1-\kappa_{0x.w} - L^{XU}_{x'w} + \epsilon_w(\kappa_{0x'.w} + \kappa_{1x'.w})$. The last two relationships follow immediately from the definition of ϵ_w. ■

Our constraints found so far collapse to some of the constraints found in the standard IV models (Balke and Pearl, 1997; Dawid, 2003) given $\epsilon_w = 0$, $\underline{\beta} = \overline{\beta} = 1$. Namely,

$$\begin{array}{rcl} \eta_{xw} & \leq & 1-\zeta_{0x.w} \\ \eta_{xw} & \leq & 1-\zeta_{0x.w'} \\ \eta_{xw} & \geq & \zeta_{1x.w} \\ \eta_{xw} & \geq & \zeta_{1x.w'} \end{array}$$

However, none of the constraints so far found counterparts in the following:

$$\begin{array}{rcl} \eta_{xw} & \leq & \zeta_{0x.w} + \zeta_{1x.w} + \zeta_{1x.w'} + \zeta_{1x'.w'} \\ \eta_{xw} & \leq & \zeta_{0x.w'} + \zeta_{1x.w'} + \zeta_{1x.w} + \zeta_{1x'.w} \\ \eta_{xw} & \geq & \zeta_{1x.w} + \zeta_{1x'.w} - \zeta_{0x.w'} - \zeta_{1x'.w'} \\ \eta_{xw} & \geq & \zeta_{1x.w'} + \zeta_{1x'.w'} - \zeta_{0x.w} - \zeta_{1x'.w} \end{array}$$

These constraints have the distinctive property of being functions of both $P(Y=x, X=x \mid W=w)$ and $P(Y=x, X=x \mid W=w')$, simultaneously. So far, we have only used the basic identities and constraints, without attempting at deriving constraints that are not a direct application of such identities. In the framework of (Dawid, 2003; Ramsahai, 2012), it is clear that general linear combinations of functions of $\{\delta^\star_{x.w}\eta^\star_{1x.w}, \delta^\star_{x.w}, \eta^\star_{1x.w}\}$ can generate constraints on observable quantities $\zeta_{yx.w}$ and causal quantities of interest, η_{xw}. We need to encompass these possibilities in such a way that we get a framework for generating symbolic constraints as a function of $\{\epsilon_w, \epsilon_y, \epsilon_x, \underline{\beta}, \overline{\beta}\}$.

One of the difficulties on exploiting a black-box polytope package for that is due to the structure of the process, which exploits the constraints in Section 4 by first finding the extreme points of the feasible region of $\{\delta^\star_w\}$, $\{\eta^\star_{xw}\}$. If we use the constraints

$$|\eta^\star_{x1} - \eta^\star_{x0'}| \leq \epsilon_w$$
$$0 \leq \eta^\star_{xw} \leq 1$$

then assuming $0 < \epsilon_w < 1$, we always obtain the following six extreme points,

$$(0,0)$$
$$(0,\epsilon_w)$$
$$(\epsilon_w,0)$$
$$(1-\epsilon_w,1)$$
$$(1,1-\epsilon_w)$$
$$(1,1)$$

In general, however, once we introduce constraints $L^{YU}_{xw} \leq \eta^\star_{xw} \leq U^{XU}_{xw}$, the number of extreme points will vary. Moreover, when multiplied with the extreme points of the space $\delta^\star_1 \times \delta^\star_0$, the resulting extreme points of $\zeta^\star_{yx.w}$ might be included or excluded of the polytope depending on the relationship among $\{\epsilon_w, \epsilon_x, \epsilon_y\}$ and the observable $P(Y, X \mid W)$. Numerically, this is not a problem (barring numerical instabilities, which do occur with a nontrivial frequency). Algebraically, this makes the problem considerably complicated[29]. Instead, in what follows we will define a simpler framework that will not give tight constraints, but will shed light on the relationship between constraints, observable probabilities and the ϵ parameters. This will also be useful to scale up the full Witness Protection Program, as discussed in the main paper.

A.2 Methodology for Cross-W Constraints

Consider the standard IV model again, i.e., where W is exogenous with no direct effect on Y. So far, we have not replicated anything such as e.g. $\eta_1 \leq \zeta_{00.0} + \zeta_{11.0} + \zeta_{10.1} + \zeta_{11.1}$. We call this a "cross-W" constraint, as it relates observables under different values of $W \in \{0,1\}$. These are important when considering weakening the effect $W \to Y$. The recipe for deriving them will be as follows. Consider the template

$$\delta^\star_0 f_1(\eta^\star_0, \eta^\star_1) + \delta^\star_1 f_2(\eta^\star_0, \eta^\star_1) + f_3(\eta^\star_0, \eta^\star_1) \geq 0 \tag{36}$$

such that $f_i(\cdot,\cdot)$ are linear. Linearity is imposed so that this function will correspond to a linear function of $\{\zeta^\star, \eta^\star, \delta^\star\}$, of which expectations will give observed probabilities or interventional probabilities.

We will require that evaluating this expression at each of the four extreme points of the joint space $(\delta^\star_0, \delta^\star_1) \in \{0,1\}^2$ will translate into one of the basic constraints $1 - \eta^\star_i \geq 0$ or $\eta^\star_i \geq 0$, $i \in \{0,1\}$. This implies any combination of $\{\delta^\star_0, \delta^\star_1, \eta^\star_0, \eta^\star_1\}$ will satisfy (36) (more on that later).

29. As a counterpart, imagine we defined a polytope through the matrix inequality $A\mathbf{x} \leq \mathbf{b}$. If we want to obtain its extreme point representation as an algebraic function of the entries of matrix A and vector $\mathbf{b}$, this will be a complicated problem since we cannot assume we know the magnitudes and signs of the entries.

Given a choice of basic constraint (say, $\eta^\star_1 \geq 0$), and setting $\delta^\star_0 = \delta^\star_1 = 0$, this immediately identifies $f_3(\cdot,\cdot)$. We assign the constraint corresponding to $\delta^\star_0 = \delta^\star_1 = 1$ with the "complementary constraint" for η_1 (in this case, $\eta^\star_1 \leq 1$). This leaves two choices for assigning the remaining constraints.

Why do we associate the $\delta^\star_0 = \delta^\star_1 = 1$ case with the complementary constraint? Let us parameterize each function as $f_i(\eta^\star_0, \eta^\star_1) \equiv a_i\eta^\star_0 + b_i\eta^\star_1 + c_i$. Let $a_3 = q$, where either $q = 1$ (case $\eta^\star_0 \geq 0$) or $q = -1$ (case $1 - \eta^\star_0 \geq 0$). Without loss of generality, assume case $(\delta^\star_0 = 1, \delta^\star_1 = 0)$ is associated with the complementary constraint where the coefficient of $\eta^\star_0$ should be $-q$. For the other two cases, the coefficient of $\eta^\star_0$ should be 0 by construction. We get the system

$$\begin{aligned} a_3 &= q \\ a_1 + a_3 &= -q \\ a_2 + a_3 &= 0 \\ a_1 + a_2 + a_3 &= 0 \end{aligned}$$

This system has no solution. Assume instead $\delta^\star_0 = \delta^\star_1 = 1$ is associated with the complementary constraint where the coefficient of $\eta^\star_0$ should be $-q$. The system now is:

$$\begin{aligned} a_3 &= q \\ a_1 + a_3 &= 0 \\ a_2 + a_3 &= 0 \\ a_1 + a_2 + a_3 &= -q \end{aligned}$$

This system always have the solution $a_1 = a_2 = -q$. We do have freedom with b_1, b_2, b_3, which means we can choose to allocate the remaining two cases in two different ways.

Lemma 7 *Consider the constraints derived by the above procedure. Then any choice of* $(\delta^\star_0, \delta^\star_1, \eta^\star_0, \eta^\star_1) \in [0,1]^4$ *will satisfy these constraints.*

Proof Without loss of generality, let $f_3(\eta^\star_0, \eta^\star_1) = q\eta^\star_0 + (1-q)/2$, $q \in \{-1, 1\}$. That is, $a_3 = q, b_3 = 0, c_3 = (1-q)/2$. This implies $a_1 = a_2 = -q$ (as above). Associating $(\delta^\star_0 = 1, \delta^\star_1 = 0)$ with $\eta^\star_1 \geq 0$ gives $\{b_1 = 1, c_1 = (q-1)/2\}$ and consequently associating $(\delta^\star_0 = 0, \delta^\star_0 = 1)$ with $1 - \eta^\star_1 \geq 0$ implies $\{b_2 = -1, c_2 = (1+q)/2\}$. Plugging this into the expression $\delta^\star_0 f_1(\eta^\star_0, \eta^\star_1) + \delta^\star_1 f_2(\eta^\star_0, \eta^\star_1) + f_3(\eta^\star_0, \eta^\star_1)$ we get

$$\begin{aligned} &= \delta^\star_0(-q\eta^\star_0 + \eta^\star_1 + (q-1)/2) + \delta^\star_1(-q\eta^\star_0 - \eta^\star_1 + (1+q)/2) + q\eta^\star_0 + (1-q)/2 \\ &= \eta^\star_0(q - (\delta^\star_0 + \delta^\star_1)q) + \eta^\star_1(\delta^\star_0 - \delta^\star_1) + \delta^\star_0(q-1)/2 + \delta^\star_1(1+q)/2 + (1-q)/2 \\ &= \eta^\star_0(q - (\delta^\star_0 + \delta^\star_1)q) + \eta^\star_1(\delta^\star_0 - \delta^\star_1) + (-q + (\delta^\star_0 + \delta^\star_1)q)/2 + (\delta^\star_1 - \delta^\star_0 + 1)/2 \\ \\ &= q((\delta^\star_1 + \delta^\star_0) - 1)(1 - 2\eta^\star_0)/2 + ((\delta^\star_1 - \delta^\star_0)(1 - 2\eta^\star_1) + 1)/2 \\ &= (\delta^\star_1 + \delta^\star_0 - 1)s/2 + (\delta^\star_1 - \delta^\star_0)t/2 + 1/2 \end{aligned}$$

where $s = q(1 - 2\eta^\star_0) \in [-1, 1]$ and $t = (1 - 2\eta^\star_1) \in [-1, 1]$. Then evaluating at the four extreme points $s, t \in \{-1, +1\}$ we get $\delta_0, \delta_1, 1-\delta_0, 1-\delta_1$, all of which are non-negative. ■

The procedure derives 8 bounds (4 cases that we get by associating f_3 with either $\eta_x \geq 0$ or $1 - \eta_x \geq 0$. For each of these cases, 2 subcases what we get by assigning $(\delta^\star_0 = 1, \delta^\star_1 = 0)$

with either $\eta_{x'} \geq 0$ or $1 - \eta_{x'} \geq 0$). Now, for an illustration of one case:

Deriving a constraint for the standard IV model, example: $f_3(\eta_0^\star, \eta_1^\star) \equiv \eta_0^\star \geq 0$

Associate $\eta_1^\star \geq 0$ with assignment $(\delta_0^\star = 1, \delta_1^\star = 0)$ (implying we associate $\eta_1^\star \leq 1$ with assignment $(\delta_0^\star = 0, \delta_1^\star = 1)$ and $\eta_0^\star \leq 1$ with $(\delta_0^\star = 1, \delta_1^\star = 1)$). This uniquely gives $f_1(\eta_0^\star, \eta_1^\star) = \eta_1^\star - \eta_0^\star$, $f_2(\eta_0^\star, \eta_1^\star) = -\eta_1^\star - \eta_0^\star + 1$. The resulting expression is

$$\delta_0^\star(\eta_1^\star - \eta_0^\star) + \delta_1^\star(-\eta_1^\star - \eta_0^\star + 1) + \eta_0^\star \geq 0$$

from which we can verify that the assignment $(\delta_0^\star = 1, \delta_1^\star = 1)$ gives $\eta_0^\star \leq 1$. Now, we need to take the expectation of the above with respect to U to obtain observables ζ and causal distributions η. However, first we need some rearrangement so that we match $\eta_0^\star$ with corresponding $(1 - \delta_w^\star)$ and so on.

$$\begin{array}{rcl} \eta_1^\star(\delta_0^\star - \delta_1^\star) + \eta_0^\star(1 - \delta_0^\star - \delta_1^\star) + \delta_1^\star & \geq & 0 \\ \eta_1^\star(\delta_0^\star - \delta_1^\star) + \eta_0^\star((1 - \delta_0^\star) + (1 - \delta_1^\star) - 1) + \delta_1^\star & \geq & 0 \\ \zeta_{11.0}^\star - \zeta_{11.1}^\star + \zeta_{10.0}^\star + \zeta_{10.1}^\star - \eta_0^\star + \zeta_{01.1}^\star + \zeta_{11.1}^\star & \geq & 0 \end{array}$$

Taking expectations and rearranging it, we have

$$\eta_0 \leq \zeta_{11.0} + \zeta_{10.0} + \zeta_{10.1} + \zeta_{01.1}$$

rediscovering one of the IV bounds for η_0. Choosing to associate $\eta_1^\star \geq 0$ with assignment $(\delta_0^\star = 0, \delta_1^\star = 1)$ will give instead

$$\eta_0 \leq \zeta_{11.1} + \zeta_{10.1} + \zeta_{10.0} + \zeta_{01.0}$$

Basically the effect of one of the two choices within any case is to switch $\zeta_{yx.w}$ with $\zeta_{yx.w'}$. ■

A.3 Deriving Cross-W Constraints

What is left is a generalization of that under the condition $|\eta_{xw} - \eta_{xw'}| \leq \epsilon_w$, $w \neq w'$, instead of $\eta_{xw} = \eta_{xw'}$. In this situation, we exploit the constraint $\underline{L} \leq \eta_{xw}^\star \leq \overline{U}$ instead of $0 \leq \eta_{xw}^\star \leq 1$ or $L_{xw}^{YU} \leq \eta_{xw}^\star \leq U_{xw}^{YU}$, where $\underline{L} \equiv \min\{L_{xw}^{YU}\}, \overline{U} \equiv \max\{U_{xw}^{YU}\}$. Using $L_{xw}^{YU} \leq \eta_{xw}^\star \leq U_{xw}^{YU}$ complicates things considerably. Also, we will not derive here the analogue proof of Lemma 1 for the case where $(\eta_0^\star, \eta_1^\star) \in [\underline{L}, \overline{U}]^2$, as it is analogous but with a more complicated notation.

Proof of Theorem 4 We demonstrate this through two special cases.
General Model, Special Case 1: $f_3(\eta_{0w}^\star, \eta_{1w}^\star) \equiv \eta_{xw}^\star - \underline{L} \geq 0$

There are two modifications. First, we perform the same associations as before, but with respect to $\underline{L} \leq \eta_{xw}^\star \leq \overline{U}$ instead of $0 \leq \eta_x^\star \leq 1$. Second, before we take expectations, we swap some of the $\eta_{xw}^\star$ with $\eta_{xw'}^\star$ up to some error ϵ_w.

Following the same sequence as in the example for the IV model, we get the resulting expression (where $x' \equiv \{0,1\} \backslash x$):

$$\delta_w^\star(\eta_{x'w}^\star - \eta_{xw}^\star) + \delta_{w'}^\star(-\eta_{x'w}^\star - \eta_{xw}^\star + \overline{U} + \underline{L}) + \eta_{xw}^\star - \underline{L} \geq 0$$

from which we can verify that the assignment $(\delta_w^\star = 1, \delta_{w'}^\star = 1)$ gives $\overline{U} - \eta_{xw}^\star \geq 0$. Now, we need to take the expectation of the above with respect to U to obtain "observables" κ and causal effects ω. However, the difficulty now is that terms $\eta_{xw}^\star \delta_{w'}^\star$ and $\eta_{xw'}^\star \delta_w^\star$ have no observable counterpart under expectation. We get around this transforming $\eta_{xw'}^\star \delta_w^\star$ into $\eta_{xw}^\star \delta_w^\star$ (and $\eta_{xw}^\star \delta_{w'}^\star$ into $\eta_{xw'}^\star \delta_{w'}^\star$) by adding the corresponding correction $-\eta_{xw}^\star \leq -\eta_{xw'}^\star + \epsilon_w$:

$$\begin{array}{rcl} \delta_w^\star(\eta_{x'w}^\star - \eta_{xw}^\star) + \delta_{w'}^\star(-\eta_{x'w}^\star - \eta_{xw}^\star + \overline{U} + \underline{L}) + \eta_{xw}^\star - \underline{L} & \geq & 0 \\ \delta_w^\star(\eta_{x'w}^\star - \eta_{xw}^\star) + \delta_{w'}^\star(-\eta_{x'w'}^\star + \epsilon_w - \eta_{xw'}^\star + \epsilon_w + \overline{U} + \underline{L}) + \eta_{xw}^\star - \underline{L} & \geq & 0 \\ \eta_{x'w}^\star \delta_w^\star + \eta_{xw}^\star(1 - \delta_w^\star) - \eta_{x'w'} \delta_{w'}^\star - \eta_{xw'} \delta_{w'}^\star + \delta_{w'}^\star(\overline{U} + \underline{L} + 2\epsilon_w) - \underline{L} & \geq & 0 \end{array}$$

Now, the case for $x = 1$ gives

$$\begin{array}{rcl} \eta_{0w}^\star \delta_w^\star + \eta_{1w}^\star(1 - \delta_w^\star) - \eta_{0w'} \delta_{w'}^\star - \eta_{1w'} \delta_{w'}^\star + \ldots & \geq & 0 \\ \eta_{0w}^\star(1 - (1 - \delta_w^\star)) + \eta_{1w}^\star(1 - \delta_w^\star) - \eta_{0w'}^\star(1 - (1 - \delta_{w'}^\star)) - \eta_{1w'}^\star \delta_{w'}^\star + \ldots & \geq & 0 \end{array}$$

Taking the expectations:

$$\omega_{0w} - \kappa_{10.w} + \omega_{1w} - \kappa_{11.w} - \omega_{0w'} + \kappa_{10.w'} - \kappa_{11.w'} + \chi_{w'}(\overline{U} + \underline{L} + 2\epsilon_w) - \underline{L} \geq 0 \quad (37)$$

Notice that for $\underline{\beta} = \overline{\beta} = 1$, $\underline{L} = 0$, $\overline{U} = 1$, $\epsilon_w = 0$, this implies $\eta_{xw} = \eta_{xw'}$ and this collapses to

$$\eta_{0w} - \zeta_{10.w} + \eta_{1w} - \zeta_{11.w} - \eta_{0w'} + \zeta_{10.w'} - \zeta_{11.w'} + \delta_{w'} \geq 0$$

$$\eta_{1w} \geq \zeta_{10.w} + \zeta_{11.w} - \zeta_{10.w'} - \zeta_{01.w'}$$

which is one of the lower bounds one obtains under the standard IV model.

The case for $x = 0$ is analogous and gives

$$\omega_{0w'} \leq \kappa_{11.w} + \kappa_{10.w} + \kappa_{10.w'} - \kappa_{11.w'} + \chi_{w'}(\overline{U} + \underline{L} + 2\epsilon_w) - \underline{L} \quad (38)$$

The next subcase is when we exchange the assignment of $(\delta_w^\star, \delta_{w'}^\star)$ to other constraints. We obtain the following inequality:

$$\delta_{w'}^\star(\eta_{x'w}^\star - \eta_{xw}^\star) + \delta_w^\star(-\eta_{x'w}^\star - \eta_{xw}^\star + \overline{U} + \underline{L}) + \eta_{xw}^\star - \underline{L} \geq 0$$

which from an analogous sequence of steps leads to

$$\begin{array}{rcl} \delta_{w'}^\star(\eta_{x'w}^\star - \eta_{xw}^\star) + \delta_w^\star(-\eta_{x'w}^\star - \eta_{xw}^\star + \overline{U} + \underline{L}) + \eta_{xw}^\star - \underline{L} & \geq & 0 \\ \delta_{w'}^\star(\eta_{x'w'}^\star + \epsilon_w - \eta_{xw'}^\star + \epsilon_w) + \delta_w^\star(-\eta_{x'w}^\star - \eta_{xw}^\star + \overline{U} + \underline{L}) + \eta_{xw}^\star - \underline{L} & \geq & 0 \\ \eta_{x'w'}^\star \delta_{w'}^\star - \eta_{xw'}^\star \delta_{w'}^\star + 2\delta_{w'}^\star \epsilon_w - \eta_{x'w}^\star \delta_w^\star + \eta_{xw}^\star(1 - \delta_w^\star) + \delta_w^\star(\overline{U} + \underline{L}) - \underline{L} & \geq & 0 \end{array}$$

For $x = 1$,

$$\begin{array}{rcl} \eta_{0w'}^\star \delta_{w'}^\star - \eta_{1w'}^\star \delta_{w'}^\star + \eta_{0w}^\star \delta_w^\star + \eta_{1w}^\star(1 - \delta_w^\star) + \ldots & \geq & 0 \\ \eta_{0w'}^\star(1 - (1 - \delta_{w'}^\star)) - \eta_{1w'}^\star \delta_{w'}^\star - \eta_{0w}^\star(1 - (1 - \delta_w^\star)) + \eta_{1w}^\star(1 - \delta_w^\star) + \ldots & \geq & 0 \end{array}$$

Taking expectations,

$$\omega_{0w'} - \kappa_{10.w'} - \kappa_{11.w'} - \omega_{0w} + \kappa_{10.w} + \omega_{1w} - \kappa_{11.w} + 2\chi_{w'}\epsilon_w + \chi_w(\overline{U} + \underline{L}) - \underline{L} \geq 0 \quad (39)$$

For $x = 0$,

$$\begin{aligned}
\eta^\star_{1w'}\delta^\star_{w'} - \eta^\star_{0w'}\delta^\star_{w'} + \eta^\star_{1w}\delta^\star_w + \eta^\star_{0w}(1-\delta^\star_w) + \ldots &\geq 0\\
\eta^\star_{1w'}\delta^\star_{w'} - \eta^\star_{0w'}(1-(1-\delta^\star_{w'})) - \eta^\star_{1w}\delta^\star_w + \eta^\star_{0w}(1-\delta^\star_w) + \ldots &\geq 0\\
\kappa_{11.w'} - \omega_{0w'} + \kappa_{10.w'} - \kappa_{11.w} + \kappa_{10.w} + 2\chi_{w'}\epsilon_w + \chi_w(\overline{U}+\underline{L}) - \underline{L} &\geq 0\\
\omega_{0w'} \leq \kappa_{11.w'} + \kappa_{10.w'} - \kappa_{11.w} + \kappa_{10.w} + 2\chi_{w'}\epsilon_w + \chi_w(\overline{U}+\underline{L}) - \underline{L} &\quad (40)
\end{aligned}$$

General Model, Special Case 2: $f_3(\eta^\star_{0w}, \eta^\star_{1w}) \equiv \overline{U} - \eta^\star_{xw} \geq 0$

Associate $\eta^\star_{x'w} \geq \underline{L}$ with assignment $(\delta^\star_w = 1, \delta^\star_{w'} = 0)$ (implying we associate $\eta^\star_{x'w} \leq \overline{U}$ with assignment $(\delta^\star_w = 0, \delta^\star_{w'} = 1)$ and $\eta^\star_{xw} \geq \underline{L}$ with $(\delta^\star_w = 1, \delta^\star_{w'} = 1)$). The resulting expression is

$$\delta^\star_w(\eta^\star_{x'w} + \eta^\star_{xw} - \overline{U} - \underline{L}) + \delta^\star_{w'}(-\eta^\star_{x'w} + \eta^\star_{xw}) + \overline{U} - \eta^\star_{xw} \geq 0$$

Following the same line of reasoning as before, we get this for $x = 1$:

$$\omega_{0w} - \omega_{0w'} - \omega_{1w} - \kappa_{10.w} + \kappa_{11.w} + \kappa_{10.w'} + \kappa_{11.w'} - \chi_w(\overline{U}+\underline{L}) + 2\epsilon_w\chi_{w'} + \overline{U} \geq 0 \quad (41)$$

We get this for $x = 0$:

$$\omega_{0w'} \geq -\kappa_{11.w} + \kappa_{10.w} + \kappa_{11.w'} + \kappa_{10.w'} + \chi_w(\overline{U}+\underline{L}) - 2\epsilon_w\chi_{w'} - \overline{U} \quad (42)$$

With the complementary assignment, we start with the relationship

$$\delta^\star_{w'}(\eta^\star_{x'w} + \eta^\star_{xw} - \overline{U} - \underline{L}) + \delta^\star_w(-\eta^\star_{x'w} + \eta^\star_{xw}) + \overline{U} - \eta^\star_{xw} \geq 0$$

For $x = 1$,

$$\omega_{0w'} - \omega_{0w} - \omega_{1w} - \kappa_{10.w'} + \kappa_{11.w'} + \kappa_{10.w} + \kappa_{11.w} + \chi_{w'}(2\epsilon_w - \overline{U} - \underline{L}) + \overline{U} \geq 0 \quad (43)$$

For $x = 0$,

$$\omega_{0w'} \geq -\kappa_{11.w'} + \kappa_{10.w'} + \kappa_{11.w} + \kappa_{10.w} - \chi_{w'}(2\epsilon_w - \overline{U} - \underline{L}) - \overline{U} \quad (44)$$

Notice that the bounds obtained are asymmetric in x, i.e., we derive different bounds for ω_{0w} and ω_{1w}. Symmetry is readily obtained by the same derivation where $\delta^\star_w$ is interpreted as $P(X = 0 \mid W = w, U)$ and x is swapped with x'. ■

A.4 Linear Case

Our final proof refers to results introduced in Section 6.

Proof of Theorem 5 Variable s_{xx} appears only in Equation (24) and the inequalities $0 \leq s_{xx} \leq 1$. The intersection of these relationships is satisfiable if and only if $0 \leq a^2 + 2as_{wx} \leq 1$ is satisfiable. Moreover, s_{wx} appears only in Equation (22). Solving this equation for s_{wx} and plugging it in $0 \leq a^2 + 2as_{wx} \leq 1$, we obtain $0 \leq -a^2 + 2a\rho_{wx} \leq 1$. The quadratic

expression for a achieves a unique maximum at $a^\star = \rho_{wx}$, implying $-a^{\star 2} + 2a^\star\rho_{wx} = \rho^2_{wx} \leq 1$. We can then drop the inequality $-a^2 + 2a\rho_{wx} \leq 1$ as this is always satisfied. The resulting interval is $a^2 - 2a\rho_{wx} \leq 0$, and the set of values of a satisfying it is either the interval $[2\rho_{wx}, 0]$ or $[0, 2\rho_{wx}]$ depending on the sign of ρ_{wx}. This can be written as

$$\min(0, 2\rho_{wx}) \leq a \leq \max(0, 2\rho_{wx}) \quad (45)$$

The intersection of Equation (22) and $-\epsilon_{wx} \leq s_{wx} \leq \epsilon_{wx}$ is satisfiable only if $\rho_{wx} - \epsilon_{wx} \leq a \leq \rho_{wx} + \epsilon_{wx}$. Combining this interval with (45), we obtain the inequality

$$\max(\min(0, 2\rho_{wx}), \rho_{wx} - \epsilon_{wx}) \leq a \leq \min(\max(0, 2\rho_{wx}), \rho_{wx} + \epsilon_{wx}) \quad (46)$$

which $\mathcal{L}_a$ and $\mathcal{U}_a$ being defined as the lower and upper bounds, respectively, in the interval above.

Since a now only appears in (46) and Equation (25), and assuming $s_{wy} \geq 0$, the intersection of the two equations is satisfiable if and only if

$$\rho_{xy} - \mathcal{U}_a s_{wy} \leq b + c\rho_{wx} + s_{xy} \leq \rho_{xy} - \mathcal{L}_a s_{wy} \quad (47)$$

Equation (31) follows from s_{xy} not appearing anywhere else but in the relationship $-\epsilon_{xy} \leq s_{xy} \leq \epsilon_{xy}$, and also considering the case $s_{wy} < 0$.

Equation (26) and the relationship $0 \leq s_{yy} \leq 1$ is satisfiable if and only if

$$0 \leq b^2 + 2bc\rho_{wx} + c^2 + 2[b(as_{wy} + s_{xy}) + cs_{wy}] \leq 1 \quad (48)$$

is satisfiable. From Equation (25) we have $as_{wy} + s_{xy} = \rho_{xy} - b - c\rho_{wx}$ and from Equation (23) we have $s_{wy} = \rho_{wy} - b\rho_{wx} - c$. Making these substitutions into (48), we get

$$0 \leq -b^2 - 2bc\rho_{wx} - c^2 + 2(b\rho_{xy} + c\rho_{wy}) \leq 1.$$

The quadratic function in (b, c) has a unique maximum. Assuming for now c is unconstrained, taking the derivatives of $-b^2 - 2bc\rho_{wx} - c^2 + 2(b\rho_{xy} + c\rho_{wy})$ with respect to b and c, and setting them to zero, we obtain the stationary point

$$b^\star = \frac{\rho_{xy} - \rho_{wx}\rho_{wy}}{1 - \rho^2_{wx}}, \quad c^\star = \frac{\rho_{wy} - \rho_{wx}\rho_{xy}}{1 - \rho^2_{wx}}, \quad (49)$$

and by assumption it follows that $c^\star = 0$. Plugging $c^\star$ in $-b^2 - 2bc\rho_{wx} - c^2 + 2(b\rho_{xy} + c\rho_{wy})$, we get $b(2\rho_{xy} - b) \leq \rho^2_{xy} \leq 1$. So, it is sufficient to satisfy

$$b^2 + 2bc\rho_{wx} + c^2 - 2(b\rho_{xy} + c\rho_{wy}) \leq 0. \quad (50)$$

■

References

J. Altonji, T. Elder, and C. Taber. Selection on observed and unobserved variables: Assessing the effectiveness of Catholic schools. *Journal of Political Economy*, 113:151–184, 2005.

A. Balke and J. Pearl. Bounds on treatment effects from studies with imperfect compliance. *Journal of the American Statistical Association*, 92:1171–1176, 1997.

W. Buntine. Theory refinement on Bayesian networks. *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence (UAI 1991)*, pages 52–60, 1991.

W. Buntine and A. Jakulin. Applying discrete PCA in data analysis. *Proceedings of 20th Conference on Uncertainty in Artificial Intelligence (UAI 2004)*, pages 59–66, 2004.

Z. Cai, M. Kuroki, J. Pearl, and J. Tian. Bounds on direct effects in the presence of confounded intermediate variables. *Biometrics*, 64:695–701, 2008.

L. Chen, F. Emmert-Streib, and J. D. Storey. Harnessing naturally randomized transcription to infer regulatory relationships among genes. *Genome Biology*, 8:R219, 2007.

G. Cooper. A simple constraint-based algorithm for efficiently mining observational databases for causal relationships. *Data Mining and Knowledge Discovery*, 2, 1997.

J. Cornfield, W. Haenszel, E. Hammond, A. Lilienfeld, M. Shimkin, and E. Wynder. Smoking and lung cancer: recent evidence and a discussion of some questions. *Journal of the National Cancer Institute*, 22:173–203, 1959.

A.P. Dawid. Causal inference using influence diagrams: the problem of partial compliance. In P.J. Green, N.L. Hjort, and S. Richardson, editors, *Highly Structured Stochastic Systems*, pages 45–65. Oxford University Press, 2003.

G. Elidan. Copulas in Machine Learning. *Copulae in Mathematical and Quantitative Finance, Lecture Notes in Statistics*, pages 39–60, 2013.

D. Entner, P.O. Hoyer, and P. Spirtes. Statistical test for consistent estimation of causal effects in linear non-Gaussian models. *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS 2012)*, pages 364–372, 2012.

D. Entner, P. Hoyer, and P. Spirtes. Data-driven covariate selection for nonparametric estimation of causal effects. *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS 2013)*, pages 256–264, 2013.

R. Evans. Graphical methods for inequality constraints in marginalized DAGs. *Proceedings of the 22nd Workshop on Machine Learning and Signal Processing*, 2012.

N. Friedman and D. Koller. Being Bayesian about network structure: a Bayesian approach to structure discovery in Bayesian networks. *Machine Learning Journal*, 50:95–126, 2003.

T. Haavelmo. The statistical implications of a system of simultaneous equations. *Econometrica*, 11:1–12, 1943.

N. Harris and M. Drton. PC algorithm for nonparanormal graphical models. *Journal of Machine Learning Research*, 14:3365–3383, 2013.

K. Hirano, G. Imbens, D. Rubin, and X.-H. Zhou. Assessing the effect of an influenza vaccine in an encouragement design. *Biometrics*, 1:69–88, 2000.

P. Hoff. Extending the rank likelihood for semiparametric copula estimation. *Annals of Applied Statistics*, 1:265–283, 2007.

C. Kleiber and A. Zeileis. *Applied Econometrics with R.* Springer-Verlag, 2008.

J. Kuipers, G. Moffa, and D. Heckerman. Addendum on the scoring of Gaussian directed acyclic graphical models. *Annals of Statistics*, 42:1689–1691, 2014.

S. Mani, G. Cooper, and P. Spirtes. A theoretical study of Y structures for causal discovery. *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI2006)*, pages 314–323, 2006.

C. Manski. *Identification for Prediction and Decision.* Harvard University Press, 2007.

C. McDonald, S. Hiu, and W. Tierney. Effects of computer reminders for influenza vaccination on morbidity during influenza epidemics. *MD Computing*, 9:304–312, 1992.

C. Meek. Strong completeness and faithfulness in Bayesian networks. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI 1995)*, pages 411–418, 1995.

J. Mooij and J. Cremers. An empirical study of one of the simplest causal prediction algorithms. *Proceedings of the Advances in Causal Inference Workshop, UAI 2015*, 2015.

T. Mroz. The sensitivity of an empirical model of married women's hours of work to economic and statistical assumptions. *Econometrica*, 55:765–799, 1987.

R. Nelsen. *An Introduction to Copulas.* Springer-Verlag, 2007.

J. Pearl. *Causality: Models, Reasoning and Inference.* Cambridge University Press, 2000.

J. Pearl. Myth, confusion, and science in causal analysis. *UCLA Cognitive Systems Laboratory, Technical Report (R-348)*, 2009.

R. Ramsahai. Causal bounds and observable constraints for non-deterministic models. *Journal of Machine Learning Research*, 13:829–848, 2012.

R. Ramsahai and S. Lauritzen. Likelihood analysis of the binary instrumental variable model. *Biometrika*, 98:987–994, 2011.

J. Ramsey, P. Spirtes, and J. Zhang. Adjacency-faithfulness and conservative causal inference. *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI 2006)*, pages 401–408, 2006.

T. Richardson and J. Robins. Analysis of the binary instrumental variable model. In R. Dechter, H. Geffner, and J.Y. Halpern, editors, *Heuristics, Probability and Causality: A Tribute to Judea Pearl*, pages 415–444. College Publications, 2010.

T. Richardson, R. Evans, and J. Robins. Transparent parameterizations of models for potential outcomes. In J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 9*, pages 569–610. Oxford University Press, 2011.

J. Robins, R. Scheines, P. Spirtes, and L. Wasserman. Uniform consistency in causal inference. *Biometrika*, 90:491–515, 2003.

P. Rosenbaum. *Observational Studies*. Springer-Verlag, 2002a.

P. Rosenbaum. Covariance adjustment in randomized experiments and observational studies. *Statistical Science*, 17(3):286–327, 2002b.

K. Rothman, S. Greenland, and T. Lash. *Modern Epidemiology*. Wolters Kluwer, 2008.

R. Silva and R. Evans. Causal inference through a witness protection program. *Advances in Neural Information Processing Systems*, 27:298–306, 2014.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search*. Cambridge University Press, 2000.

K. Steenland and S. Greenland. Monte Carlo sensitivity analysis and Bayesian analysis of smoking as an unmeasured confounder in a study of silica and lung cancer. *American Journal of Epidemiology*, 160:384–392, 2004.

T. VanderWeele and I. Shpitser. A new criterion for confounder selection. *Biometrics*, 64: 1406–1413, 2011.

Journal of Machine Learning Research 17 (2016) 1-47 Submitted 4/15; Revised 12/15; Published 4/16

Structure Discovery in Bayesian Networks by Sampling Partial Orders*

Teppo Niinimäki TEPPO.NIINIMAKI@HELSINKI.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 68 (Gustaf Hällströminkatu 2b)
FI-00014 University of Helsinki, Finland

Pekka Parviainen PEKKA.PARVIAINEN@AALTO.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 15400 (Konemiehentie 2)
FI-00076 Aalto, Finland

Mikko Koivisto MIKKO.KOIVISTO@HELSINKI.FI
Helsinki Institute for Information Technology
Department of Computer Science
P.O. Box 68 (Gustaf Hällströminkatu 2b)
FI-00014 University of Helsinki, Finland

Editor: Edo Airoldi

Abstract

We present methods based on Metropolis-coupled Markov chain Monte Carlo (MC^3) and annealed importance sampling (AIS) for estimating the posterior distribution of Bayesian networks. The methods draw samples from an appropriate distribution of partial orders on the nodes, continued by sampling directed acyclic graphs (DAGs) conditionally on the sampled partial orders. We show that the computations needed for the sampling algorithms are feasible as long as the encountered partial orders have relatively few down-sets. While the algorithms assume suitable modularity properties of the priors, arbitrary priors can be handled by dividing the importance weight of each sampled DAG by the number of topological sorts it has—we give a practical dynamic programming algorithm to compute these numbers. Our empirical results demonstrate that the presented partial-order-based samplers are superior to previous Markov chain Monte Carlo methods, which sample DAGs either directly or via linear orders on the nodes. The results also suggest that the convergence rate of the estimators based on AIS are competitive to those of MC^3. Thus AIS is the preferred method, as it enables easier large-scale parallelization and, in addition, supplies good probabilistic lower bound guarantees for the marginal likelihood of the model.

Keywords: annealed importance sampling, directed acyclic graph, fast zeta transform, linear extension, Markov chain Monte Carlo

∗. Preliminary versions of this work have appeared in the proceedings of AISTATS 2010, UAI 2011, and IJCAI 2013 (Parviainen and Koivisto, 2010; Niinimäki et al., 2011; Niinimäki and Koivisto, 2013b). All correspondence should be addressed to the first author.

1. Introduction

The Bayesian paradigm to structure learning in Bayesian networks is concerned with the posterior distribution of the underlying directed acyclic graph (DAG) given data on the variables associated with the nodes of the graph (Buntine, 1991; Cooper and Herskovits, 1992; Madigan and York, 1995). The paradigm is appealing as it offers an explicit way to incorporate prior knowledge as well as full characterization of posterior uncertainty about the quantities of interest, including proper treatment of any non-identifiability issues. However, a major drawback of the Bayesian paradigm is its large computational requirements. Indeed, because the number of DAGs grows very rapidly with the number of nodes, exact computation of the posterior distribution becomes impractical already when there are more than about a dozen of nodes.

There are two major approaches to handle the posterior distribution of DAGs without explicitly computing and representing the entire distribution. One is to summarize the posterior distribution by a relatively small number of summary statistics, of which perhaps the most extensively used is a *mode* of the distribution, that is, a maximum a posteriori DAG (Cooper and Herskovits, 1992). Other useful statistics are the marginal posterior probabilities of so-called *structural features*, such as individual arcs or larger subgraphs (Buntine, 1991; Cooper and Herskovits, 1992; Friedman and Koller, 2003). When the interest is in how well the chosen Bayesian model fits the data, say in comparison to some alternative model, then the key quantity is the *marginal likelihood* of the model—also known as the *integrated likelihood*, *evidence*, or the *normalizing constant*—which is simply the marginal probability (density) of the observed data. Provided that the model satisfies certain modularity conditions, all these statistics can be computed in a dynamic programming fashion, and thereby avoiding exhaustive traversing through individual DAGs. Specifically, assuming a very basic form of modularity one can find a mode over n-node DAGs in $O(2^n n^2)$ time (Koivisto and Sood, 2004; Ott et al., 2004; Singh and Moore, 2005; Silander and Myllymäki, 2006) and the arc posterior probabilities and the marginal likelihood in $O(3^n n)$ time (Tian and He, 2009). Assuming a more convoluted form of modularity, also the latter quantities can be computed in $O(2^n n^2)$ time (Koivisto and Sood, 2004; Koivisto, 2006). In practice, these algorithms scale up to about 25 nodes. For mode finding, there are also algorithms based on the A* search heuristic (Yuan and Malone, 2013) and integer linear programming (Bartlett and Cussens, 2013), which often can solve even larger problem instances.

The other approach is to approximate the posterior distribution by collecting a sample of DAGs, each of which assigned a weight reflecting how representative the DAG is of the posterior distribution. Having such a collection at hand, various quantities, including the aforementioned statistics, can be estimated by appropriate weighted averages. Principled implementations of the approach have used the Markov chain Monte Carlo (MCMC) methodology in various forms: Madigan and York (1995) simulated a Markov chain that moves in the space of DAGs by simple arc changes such that the chain's stationary distribution is the posterior distribution. Friedman and Koller (2003) obtained a significantly faster mixing chain by operating, not directly on DAGs, but in the much smaller and smoother space of node orderings, or *linear orders* on the nodes more formally. The sampler, called *order-MCMC* in the sequel, requires the prior to be of a particular form that favors DAGs that are compatible with many node orderings, thus introducing a "bias." Ellis and Wong

(2008) enhanced order-MCMC by presenting a sophisticated sampler based on tempering techniques, and a heuristic for removing the bias. Also other refinements to Madigan and York's sampler have been presented (Eaton and Murphy, 2007; Grzegorczyk and Husmeier, 2008; Corander et al., 2008), however with somewhat more limited advantages over order-MCMC. More recently, Battle et al. (2010) extended Madigan and York's sampler in yet another direction by applying annealed importance sampling (AIS) (Neal, 2001) to sample fully specified Bayesian networks (i.e., DAGs equipped with the associated conditional distributions).

While the current MCMC methods for structure learning seem to work well in many cases, they also leave the following central questions open:

1. *Smoother sampling spaces.* Can we take the idea of Friedman and Koller (2003) further: are there sampling spaces that yield still a better tradeoff between computation time and accuracy?

2. *Arbitrary structure priors.* Can we efficiently remove the bias due to sampling node orderings? Specifically, is it computationally feasible to estimate posterior expectations under an arbitrary structure prior, yet exploiting the smoother sampling space of node orderings? (The method of Ellis and Wong (2008) relies on heuristic arguments and becomes computationally infeasible when the data set is small.)

3. *Efficient parallel computation.* Can we efficiently and easily exploit parallel computation, that is, to run the algorithm in parallel on thousands of processors, preferably without frequent synchronization or communication between the parallel processes. (Existing MCMC methods are designed rather for a small number of very long runs, and thus do not enable large-scale parallelization.)

4. *Quality guarantees.* Can we measure how accurate the algorithm's output is? For instance, is it a lower bound, an upper bound, or an approximation to within some multiplicative or additive term? (Existing MCMC methods offer such guarantees only in the limit of running the algorithm infinitely many steps.)

In this article, we make a step toward answering these questions in the affirmative. Specifically, we advance the state of the art by presenting three new ideas, published in a preliminary form in three conference proceedings (Parviainen and Koivisto, 2010; Niinimäki et al., 2011; Niinimäki and Koivisto, 2013b), and their combination that has not been investigated prior to the present work. The next paragraphs give an overview of our contributions.

We address the first question by introducing a sampling space, in which each state is a partial order on the nodes. Compared to the space of node orderings (i.e., linear orders on the nodes), the resulting sampling space is smaller and the induced sampling distribution is smoother. We will also show that going from linear orders to partial orders does not increase the computational cost per sampled order, as long as the partial orders are sufficiently thin, that is, they have relatively few so-called downsets. These algorithmic results build on and extend our earlier work on finding an *optimal* DAG subject to a given partial order constraint (Parviainen and Koivisto, 2013).

To address the second question, we take a somewhat straightforward approach: per sampled partial order, we draw one or several DAGs independently from the corresponding conditional distribution, and assign each DAG a weight that compensates the difference of the structure prior of interest and the "proxy prior" we employ to make order-based sampling efficient. Specifically, we show that the number of linear extensions (or, topological sorts) of a given DAG—that we need for the weight—can be computed sufficiently fast in practice for moderate-size DAGs, even though the problem is #P-hard in general (Brightwell and Winkler, 1991).

Our third contribution applies both to the third and the fourth question. Motivated by the desire for accuracy guarantees, we seek a sampler such that we know exactly from which distribution the samples are drawn. Here, the annealed importance sampling (AIS) method of Neal (2001) provides an appealing solution. It enables drawing independent and identically distributed samples and computing the associated importance weights, so that the expected value of each weighted sample matches the quantity of interest. Due to the independence of the samples, already a small number of samples may suffice, not only for producing an accurate estimate, but also for finding a relatively tight, high-confidence lower bound on the true value (Gomes et al., 2007; Gogate et al., 2007; Gogate and Dechter, 2011). Furthermore, the independence of the samples renders the approach embarrassingly parallel, requiring interaction of the parallel computations only at the very end when the independent samples are collected in a Monte Carlo estimator. We note that Battle et al. (2010) adopted AIS for quite different reasons. Namely, due to the structure of their model, they had to sample fully specified Bayesian networks whose posterior distribution is expected to be severely multimodal, in which case AIS is a good alternative to the usual MCMC methods.

Finally, we evaluate the significance of the aforementioned advances empirically. As a benchmark we use Friedman and Koller's (2003) simple Markov chain on the space of node orderings, however, equipped with the Metropolis-coupling technique (Geyer, 1991) to enhance the chain's mixing properties. Our implementation of the *Metropolis-coupled MCMC* (MC^3) method for the space of node orderings also serves as a proxy of a related implementation[1] of Ellis and Wong (2008). Our experimental study aims to answer two main questions: First, does the squeezing of the space of linear orders into a space of partial orders yield a significantly faster mixing Markov chain when we already use the Metropolis coupling technique to help mixing? This question was left unanswered in our preliminary work (Niinimäki et al., 2011) that only considered a single simple Markov chain similar to that of Friedman and Koller (2003). Second, are the Monte Carlo estimators based on AIS competitive to the MC^3-based estimators when we sample partial orders instead of linear orders? This question was left unanswered in our preliminary work (Niinimäki and Koivisto, 2013b) that only considered sampling linear orders.

The remainder of this article is organized as follows. We begin in Section 2 with an introduction to some basic properties of graphs and partial orders. The section also contains some more advanced algorithmic results, which will serve as building blocks of our method for learning Bayesian networks. In Section 3, we review the Bayesian formulation of the structure learning problem in Bayesian networks, and also outline the approach of Friedman and Koller (2003) based on sampling node orderings. In Section 4, we extend the idea of

1. The software used in the work of Ellis and Wong (2008) is not publicly available (W. H. Wong, personal communication, January 29, 2013).

sampling node orderings to partial orders and formulate the two sampling methods, MC^3 and AIS, in that context. We dedicate Section 5 to the description of fast algorithms for computing several quantities needed by the samplers. Experimental results are reported in Section 6. We conclude and discuss directions for future research in Section 7.

2. Partial Orders and DAGs

This section introduces concepts, notation, and algorithms associated with graphs and partial orders. In later sections, we will apply them as building blocks in the context of structure learning in Bayesian networks. The content of the first subsection will be needed already in Section 3. The content of the latter subsections will be needed only in Section 5, and the reader may wish to skip them on the first read.

2.1 Antisymmetric Binary Relations

We use the standard concepts of graphs and partial orders. Our notation is however somewhat nonstandard, which warrants a special attention. Let N be a finite set, and let $R \subseteq N \times N$ be a binary relation on N. We shall denote an element (u, v) of R by uv for short. We say that R is

$$\begin{aligned} \textit{reflexive} &\quad \text{if } u \in N \text{ implies } uu \in R\,; \\ \textit{antisymmetric} &\quad \text{if } uv, vu \in R \text{ implies } u = v\,; \\ \textit{transitive} &\quad \text{if } uv, vw \in R \text{ implies } uw \in R\,; \\ \textit{total} &\quad \text{if } u, v \in N \text{ implies } uv \in R \text{ or } vu \in R\,. \end{aligned}$$

If R has the first three properties, it is called a *partial order*. If it has all the four properties, it is called a *total* or *linear order*. The set N is the *ground set* of the order and the pair (N, R) is called a linearly or partially ordered set.

We will also consider relations that are antisymmetric but that need not have any other of the above the properties. We say that R is

$$\begin{aligned} \textit{acyclic} &\quad \text{if there are no elements } u_1, \ldots, u_l \text{ such that} \\ &\quad u_1 = u_l \text{ and } u_{i-1}u_i \in R \text{ for each } i = 2, \ldots, l\,. \end{aligned}$$

Note that acyclicity implies irreflexivity, that is, that $uu \notin R$ for all $u \in N$. If R is acyclic, we call the pair (N, R) a *directed acyclic graph* (DAG), N its *node set*, and R its *arc set*.

When the set N is clear from the context—as it will often be in our applications—we identify the structure (N, R) with the relation R. We will typically use the letters P, L, and A for a partial order, linear order, and a DAG, respectively.

The following relationships among the three objects are central in later sections of this article. Let R and Q be relations on N. We say that Q is an *extension* of R if simply $R \subseteq Q$. If Q is a linear order, then we may also call it a *linear extension* of R. Note that in the literature, a linear extension of a DAG is sometimes called a topological sort of the DAG. Sometimes we will be interested in the *number of linear extensions* of R, which we denote by $\ell(R)$. Furthermore, we say that R and Q are *compatible* with each other if they have a common linear extension $L \supseteq R, Q$. While this relationship is symmetric, in our applications one of the R, Q will be a partial order and the other one a DAG. Also,

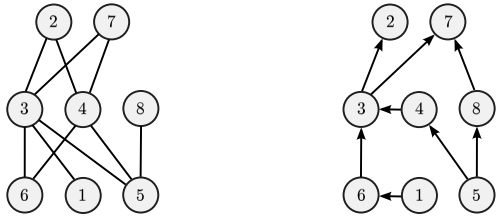


Figure 1: The Hasse diagram of a partial order on eight nodes (left) and a DAG compatible with the partial order (right).

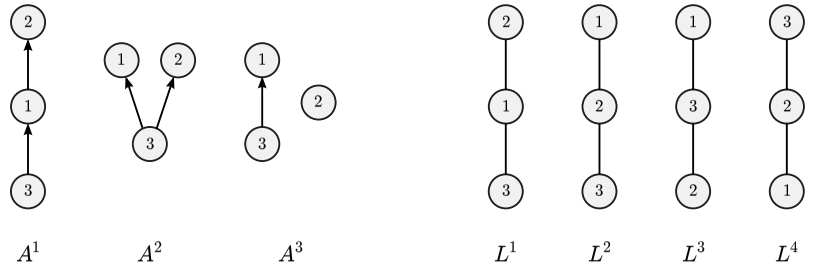


Figure 2: Three DAGs (left) and the Hasse diagrams of four linear orders (right), some of which are compatible with some of the DAGs. DAG A^1 has only one linear extension: L^1. DAG A^2 has two linear extensions: L^1 and L^2. DAG A^3 has three linear extensions: L^1, L^2 and L^3. None of the DAGs is compatible with L^4.

we say that Q is the *transitive closure* of R if Q is the minimal transitive relation that contains R. Finally, we say that R is the *transitive reduction* of Q if R is the minimal relation with the same transitive closure as Q. The transitive reduction of a partial order Q is sometimes called the *covering relation* and visualized graphically by means of the Hasse diagram. Figures 1 and 2 illustrate some of these concepts.

Some of the above described relationships can be characterized locally, in terms of the in-neighbors of each element of N. To this end we let R_v denote the set of elements that precede v in R, formally

$$R_v = \{u : uv \in R, u \neq v\}\,.$$

We will call the elements of R_v the *parents* of v and the set R_v the *parent set* of v. If R is a partial order we may call the parents also the *predecessors* of v. We observe that if Q is reflexive, then Q is an extension of R if and only if $R_v \subseteq Q_v$ for all $v \in N$.

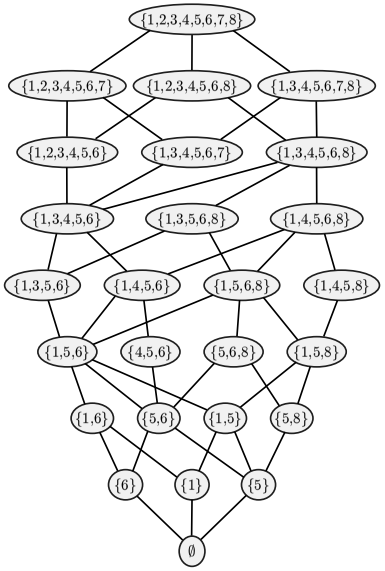


Figure 3: The covering graph of the downset lattice of the partial order shown in Figure 1.

2.2 Downsets and Counting Linear Extensions

Let P be a partial order on a set N. A subset $Y \subseteq N$ is a *downset* of P if $v \in Y$ and $uv \in P$ imply that $u \in Y$. In the literature downsets are sometimes called also *order ideals* or just *ideals*. We denote the set of downsets by $\mathcal{D}(P)$, or shorter $\mathcal{D}$ when there is no ambiguity about the partial order. The *downset lattice* of P is the set of downsets ordered by inclusion, $(\mathcal{D}, \subseteq)$. While we do not define the notion of lattice here, we note that every lattice is a partially ordered set and thus can be represented by its *covering graph*, that is, $\mathcal{D}$ equipped with the covering relation. An example of a covering graph is shown in Figure 3. Observe that in the covering graph a node X is a parent of another node Y if and only if X is obtained from Y by removing some maximal element of Y, that is, an element of

$$\max Y = \big\{u \in Y : uv \notin P \text{ for all } v \in Y \setminus \{u\}\big\}\,.$$

(Later we may use the notation $\max Y$ also for a set Y that is not a downset.)

The following result of Habib et al. (2001) guarantees us an efficient access to the downset lattice of a given partial order:

Theorem 1 *Given a partial order P on an n-element set, the covering graph of the downset lattice of P can be constructed in $O(n|\mathcal{D}|)$ time and space.*

Remark 2 Actually Habib et al. (2001) show a stronger result, namely that the factor n in the time and space complexity can be reduced to the width of the partial order.

As a first use of these concepts we consider the problem of counting the linear extensions of a given partial order P on N. Recall that we denote this count by $\ell(P)$. It is immediate that $\ell(P)$ equals the number of paths from the empty set $\emptyset$ to the ground set N in the covering graph of the downset lattice of P. Thus, letting $F(\emptyset) = 1$ and, recursively for nonempty $Y \in \mathcal{D}$,

$$F(Y) = \sum_{\substack{v \in Y \\ Y \setminus \{v\} \in \mathcal{D}}} F\big(Y \setminus \{v\}\big)\,,$$

we have that $F(N) = \ell(P)$. This result is a special case of Lemma 18 that will be given in Section 5.2. Because the covering graph provides us with an efficient access to the downsets and their parents in the covering graph, we have the following result:

Theorem 3 *Given a partial order P on an n-element set, the number of linear extensions of P can be computed in $O(n|\mathcal{D}|)$ time and space.*

This result extends to counting the linear extensions of a given DAG A. Namely, we observe that $\ell(A)$ equals the number of linear extensions of the partial order $P(A)$ that is obtained by taking the transitive closure of A and adding the self-loop vv for each node $v \in N$. The transitive closure can be computed relatively fast, in $O(n^3)$ time, using the Floyd–Warshall algorithm.

Corollary 4 *Given a DAG A on an n-element set, the number of linear extensions of A can be computed in $O\big(n^3 + n|\mathcal{D}(P(A))|\big)$ time and $O\big(n|\mathcal{D}(P(A))|\big)$ space.*

Remark 5 In the worst case the number of downsets is 2^n. We are not aware of any algorithm that would compute the exact number of linear extensions faster than in $O(n2^n)$ time in the worst case. As the problem is #P-complete (Brightwell and Winkler, 1991), there presumably is no polynomial time exact algorithm for the problem. It is possible to *approximate* the count to within any relative error $\varepsilon > 0$, roughly, in $O(\varepsilon^{-2} n^5 \log^3 n)$ time (see Sect. 4 of Bubley and Dyer, 1999). Unfortunately, the large hidden constants and the large degree of the polynomial render the approximation algorithms impractical even for moderate values of ε. It is also known that the linear extensions of P can be enumerated in constant amortized time, which enables counting in $O(n^2 + \ell(P))$ time (Pruesse and Ruskey, 1994; Ono and Nakano, 2005). However, the enumeration approach to counting is not feasible when $\ell(P)$ is large.

2.3 Fast Zeta Transforms

Denote by 2^N the power set of N and by $\mathbb{R}$ the set of real numbers. For a function $\varphi : 2^N \to \mathbb{R}$ the *zeta transform* of φ over the subset lattice $(2^N, \subseteq)$ is the function $\widehat{\varphi} : 2^N \to \mathbb{R}$ defined by

$$\widehat{\varphi}(Y) = \sum_{X \subseteq Y} \varphi(X)\,, \quad \text{for } Y \subseteq N\,. \tag{1}$$

We shall introduce two computational problems that concern the evaluation of the zeta transform in restricted settings where the input function is sparse (has a small support) and also the output function is evaluated only at some subsets. In the *inflating zeta transform* problem we are given a partial order P on N, a function $\varphi : \mathcal{D} \to \mathbb{R}$ from the downsets of P to real numbers, and an arbitrary collection $\mathcal{C}$ of subsets of N. The task is to compute the zeta transform $\widehat{\varphi}$ restricted to $\mathcal{C}$, that is, to compute $\widehat{\varphi}(Y)$ for every $Y \in \mathcal{C}$. To make the formula (1) applicable, we understand that $\varphi(X) = 0$ for $X \in 2^N \setminus \mathcal{D}$. In the *deflating zeta transform* problem we are given a partial order P on N, a collection $\mathcal{C}$ of subsets of N, and a function $\varphi : \mathcal{C} \to \mathbb{R}$. The task is to compute the zeta transform $\widehat{\varphi}$ restricted to $\mathcal{D}$. Again, we understand that $\varphi(X) = 0$ for $X \in 2^N \setminus \mathcal{C}$. Clearly, the problems coincide if we take $\mathcal{C} = \mathcal{D}$, and the resulting transform is known as the *zeta transform over the downset lattice*. In these problems we assume that the input function is given as a list of argument–value pairs $(X, \varphi(X))$, where X ranges over the domain of the function (i.e., $\mathcal{C}$ or $\mathcal{D}$).

We begin with the problem of computing a zeta transform over the downset lattice:

Theorem 6 *Given a partial order P on an n-element set and a function $\varphi : \mathcal{D} \to \mathbb{R}$, the zeta transform of φ over the downset lattice $(\mathcal{D}, \subseteq)$ can be computed in $O(n|\mathcal{D}|)$ time and space.*

To prove this result we consider the following algorithm. First construct the covering graph of the downset lattice and associate each downset Y with the value $\varphi_0(Y) = \varphi(Y)$. Then find an ordering $v_1, \ldots, v_n$ of the elements of N such that $v_i v_j \in P$ implies $i \leq j$. Next, for each downset Y and for each $i = 1, \ldots, n$, let

$$\varphi_i(Y) = \varphi_{i-1}(Y) + \begin{cases} \varphi_{i-1}(Y \setminus \{v_i\}) & \text{if } v_i \in Y \text{ and } Y \setminus \{v_i\} \in \mathcal{D}\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

Finally return the values $\varphi_n(Y)$ for $Y \in \mathcal{D}$. We can show that the algorithm is correct:

Lemma 7 *It holds that $\varphi_n(Y) = \widehat{\varphi}(Y)$ for all $Y \in \mathcal{D}$.*

The proof of this lemma and Lemmas 9–11 below are given in the appendix.

To complete the proof of Theorem 6, it remains to analyze the time and space requirements of the algorithm. The time and space requirement of constructing the covering graph are within the budget by Theorem 1. Finding a valid ordering takes $O(n^2)$ time and space using standard algorithms for topological sorting. Finally, for each $Y \in \mathcal{D}$, running the n steps and storing the values $\varphi_i(Y)$ takes $O(n)$ time and space, thus $O(n|\mathcal{D}|)$ in total. Note that the covering graph representation enables constant-time accessing of the downsets $Y \setminus \{v_i\}$ for a given downset Y.

Let us then turn to the two more general problems.

Theorem 8 *The deflating zeta transform problem and the inflating zeta transform problem can be solved in $O(n^2|\mathcal{C}| + n|\mathcal{D}|)$ time and $O(n|\mathcal{C}| + n|\mathcal{D}|)$ space.*

We prove first the claim for the deflating zeta transform problem. We develop our algorithm in two stages so as to split the sum in the zeta transform into two nested sums: the outer sum will be only over the downsets X of P, while the inner sum will gather the needed terms for each X.

The key concept is the *tail* of a downset Y, which we define as the set interval

$$\mathcal{T}_Y = \{X : \max Y \subseteq X \subseteq Y\}\,.$$

The following lemma shows that the tails are pairwise disjoint.

Lemma 9 *Let Y and Y' be distinct downsets. Then the tails $\mathcal{T}_Y$ and $\mathcal{T}_{Y'}$ are disjoint.*

We also have that each subset of the ground set belongs to some tail:

Lemma 10 *Let $X \subseteq N$. Let $Y = \{u : uv \in P, v \in X\}$, that is, the downward-closure of X. Then $Y \in \mathcal{D}$ and $X \in \mathcal{T}_Y$.*

Lemmas 9 and 10 imply that for any downset $S \in \mathcal{D}$ the tails $\{\mathcal{T}_Y : Y \in \mathcal{D} \cap 2^S\}$ partition the power set 2^S. This allows us to split the zeta transform into two nested summations. Let

$$\beta(Y) = \sum_{X \in \mathcal{T}_Y} \varphi(X)\,, \quad \text{for } Y \in \mathcal{D}\,.$$

Now

$$\widehat{\varphi}(Y) = \widehat{\beta}(Y) = \sum_{\substack{X \subseteq Y \\ X \in \mathcal{D}}} \beta(X)\,, \quad \text{for } Y \in \mathcal{D}\,.$$

Our algorithm computes $\widehat{\varphi}$ in two stages: first, given φ, it evaluates β at all downsets of P; second, given β, it evaluates $\widehat{\varphi}$ at all downsets of P. We have already seen how the second stage can be computed within the claimed time and space budget (Theorem 6).

The first stage is computationally relatively straightforward, since each $X \in \mathcal{C}$ contributes to exactly one term $\beta(Y)$. Specifically, we can compute the function β by initializing the values $\beta(Y)$ to zero, then considering each $X \in \mathcal{C}$ in turn and incrementing the value $\beta(Y)$ by $\varphi(X)$ for the unique downset Y satisfying $X \in \mathcal{T}_Y$. Using the observation that Y is the downward-closure of X, as given by Lemma 10, the set Y can be found in $O(n^2)$ time. This completes the proof of the first claim of Theorem 8.

Consider then the inflating zeta transform problem. We use the same idea as above, however, reversing the order of the two stages. Specifically, the algorithm first computes the zeta transform over the downset lattice $\mathcal{D}$, resulting in the values $\widehat{\varphi}(Y)$ for all $Y \in \mathcal{D}$. Then, the algorithm extends the output function to the domain $\mathcal{C}$ using the observation that the zeta transform is piecewise constant, that is, for any $Z \subseteq N$ we have $\widehat{\varphi}(Z) = \widehat{\varphi}(Y)$ for a unique downset Y:

Lemma 11 *Let $Z \subseteq N$. Then $\mathcal{D} \cap 2^Z = \mathcal{D} \cap 2^Y$, where the set $Y \in \mathcal{D}$ is given by*

$$Y = \{v \in Z : \text{if } uv \in P, \text{ then } u \in Z\}\,;$$

in words, Y consists of all elements of Z whose predecessors also are in Z.

Clearly the set Y can be constructed in $O(n^2)$ time for a given set Z. This completes the proof of Theorem 8.

Remark 12 The zeta transforms studied in this subsection have natural "upward-variants" where the condition $X \subseteq Y$ is replaced by the condition $X \supseteq Y$. The presented results readily apply to these variants, by complementation. Indeed, letting P be a partial order on N and denoting by $\bar{Y}$ the complement $N \setminus Y$ and by $\bar{P}$ the reversed partial order $\{vu : uv \in P\}$, we have the equivalences

$$X \supseteq Y \iff \bar{X} \subseteq \bar{Y} \quad \text{and} \quad Y \in \mathcal{D}(P) \iff \bar{Y} \in \mathcal{D}(\bar{P})\,.$$

Thus an upward-variant can be solved by first complementing the arguments of the input function, then solving the "downward-variants", and finally complementing back the arguments of the output function.

2.4 Random Sampling Variants

The above described zeta transform algorithms compute recursively large sums of nonnegative weights $\varphi(Z)$ each corresponding to a subset $Z \subseteq N$. Later we will also need a way to draw independent samples of the subsets $Z \subseteq Y$, proportionally to the weights. Coincidentally, the introduced summation algorithms readily provide us an efficient way to do this: for each sample we only need to stochastically backtrack the recursive steps.

We illustrate this generic approach to extend a summation algorithm into a sampling algorithm by considering the deflating zeta transform problem. Suppose we are given a partial order P on N, a collection of $\mathcal{C}$ of subsets of N, and a function $\varphi : \mathcal{C} \mapsto \mathbb{R}$. As an additional input we now also assume a downset $Y \in \mathcal{D}$. We consider the problem of generating a random subset $Z \in \mathcal{C} \cap 2^Y$ proportionally to $\varphi(Z)$. We show next that this problem can be solved in $O(n)$ time, provided that the deflating zeta transform has been precomputed and the intermediate results are stored in memory.

In the first stage the algorithm backtracks the zeta transform over the downset lattice, as follows. Let $Y_n = Y$. For $i = n, n-1, \ldots, 1$,

> if $Y_i \setminus \{v_i\} \in \mathcal{D}$, then with probability $\beta_{i-1}(Y_i \setminus \{v_i\})/\beta_i(Y_i)$ let $Y_{i-1} = Y_i \setminus \{v_i\}$;
> otherwise let $Y_{i-1} = Y_i$.

By the proof of Theorem 6, the resulting set Y_1 is a random draw from the subsets in $\mathcal{D} \cap 2^Y$ proportionally to $\beta(Y_1)$. This first stage takes clearly $O(n)$ time.

In the second stage the algorithm generates a random set $X \in \mathcal{T}_{Y_1}$ proportionally to $\varphi(X)$. In this case, the number of alternative options is not constant, and we need an efficient way to sample from the respective discrete probability distribution. A particularly efficient solution to this subproblem is known as the *Alias method* (Walker, 1977; Vose, 1991):

Theorem 13 (Alias method) *Given positive numbers $q_1, q_2, \ldots, q_r$, one can in $O(r)$ time construct a data structure, which enables drawing a random $i \in \{1, \ldots, r\}$ proportionally to q_i in constant time.*

We can view the list of the terms $\varphi(X)$, for $X \in \mathcal{T}_{Y_1}$, as an intermediate result, which has been precomputed and stored in memory using the Alias method. Thus the second stage takes only $O(1)$ time.

3. Bayesian Learning of Bayesian Networks

In this section we review the Bayesian approach to structure learning in Bayesian networks. Our emphasis will be on the notion of modularity of the various components of the Bayesian model. From the works of Friedman and Koller (2003) and Koivisto and Sood (2004) we also adopt the notion of order-modularity that is central in the partial-order-MCMC method, which we introduce in the next section.

3.1 The Bayesian Approach to Structure Learning

We shall consider probabilistic models for n *attributes*, the vth attribute taking values in a set $\mathcal{X}_v$. We will also assume the availability of a *data set* D consisting of m records over the attributes. We denote by D_v^j the datum for the vth attribute in the jth record. We will denote by N the index set $\{1, \ldots, n\}$, and often apply indexing by subsets. For example, if $S \subseteq N$, we write D_S^j for $\{D_v^j : v \in S\}$ and simply D^j for D_N^j.

We build a Bayesian network model for the data *a priori*, before seeing the actual data. To this end, we treat each datum D_v^j as a random variable with the state space $\mathcal{X}_v$. We model the random vectors $D^j = (D_1^j, \ldots, D_n^j)$ as independent draws from an n-variate distribution θ which is Markov with respect to some directed acyclic graph $G = (N, A)$, that is,

$$\theta(D^j) = \prod_{v \in N} \theta(D_v^j \,|\, D_{A_v}^j)\,. \tag{2}$$

Recall that $A_v = \{u : uv \in A\}$ denotes the set of parents of node v in G. We call the pair (G, θ) a *Bayesian network*, G its *structure*, and θ its *parameter*. As our interest will be in settings where the node set N is considered fixed, whereas the arc set A varies, we will identify the structure with the arc set A and drop G from the notation.

Because our interest is in learning the structure from the data, we include the Bayesian network in the model as a random variable. Thus we compose a joint distribution $p(A, \theta, D)$ as the product of a *structure prior* $p(A)$, a *parameter prior* $p(\theta \mid A)$, and the *likelihood* $p(D \,|\, A, \theta) = \prod_j \theta(D^j)$. Once the data D have been observed, our interest is in the *structure posterior* $p(A \mid D)$, which, using Bayes's rule, is obtained as the ratio $p(A)p(D \mid A)/p(D)$, where $p(D \,|\, A)$ is the *structure likelihood* and $p(D)$ the *marginal likelihood*. Note that as the parameter θ is not of direct interest, it is marginalized out. With suitable choices of the parameter prior, the marginalization can be carried out analytically; we will illustrate this below in Example 1.

Various quantities that are central in structure discovery, as well as in prediction of yet unobserved data, can be cast as the posterior expectation

$$\mathbb{E}(f \,|\, D) = \sum_A f(A) p(A \,|\, D) \tag{3}$$

of some function f that associates each structure A with a real number, or more generally, an element of a vector space. For example, if we let f be the indicator function of the structures where s is a parent of t, then $\mathbb{E}(f \mid D)$ equals the posterior probability that $st \in A$. For another example, define f in relation to the observed data D and to unobserved data D' as the conditional distribution $p(D' \mid D, A)$. Then $\mathbb{E}(f \mid D)$ equals the *posterior predictive*

distribution $p(D' \,|\, D)$. Note that in this instantiation we allowed the function f depend on the observed data D, which is only implicitly enabled in the expression (3). The marginal likelihood $p(D)$ is yet another variant of (3), obtained as the *prior expectation* $\mathbb{E}(f)$ of the function $f(A) = p(D \,|\, A)$. We will generally refer to the function f of interest as the *feature*.

3.2 Modularity and Node Orderings

From a computational point of view, the main challenge is to carry out the summation over all possible structures A, either exactly or approximately. As the number of possible structures grows very rapidly in the number of nodes, the hope is in exploiting the properties of both the posterior distribution and the feature. Here, a central role is played by functions that are modular in the sense that they factorize into a product of "local" factors, one term per pair (v, A_v), in concordance with the factorization (2).

We define the notion of modularity so that it applies equally to features, structure prior, and structure likelihood. Let φ be a mapping that associates each binary relation R on N with a real number. We say that φ is *modular* if for each node v there exists a mapping φ_v from the subsets of $N \setminus \{v\}$ to real numbers such that $\varphi(R) = \prod_{v \in N} \varphi_v(R_v)$. We call the functions φ_v the *factors* of φ.

In what follows, the relation R will typically be the arc set of a DAG. For example, the indicator function for a fixed arc st mentioned above is modular, with the factors f_v satisfying $f_v(A_v) = 0$ if $v = t$ and $s \notin A_t$ and $f_v(A_v) = 1$ otherwise.

Modular structure priors can take various specific forms. For example, a simple prior is obtained by assigning each node pair uv a real number $\kappa_{uv} > 0$ and letting the prior $p(A)$ be proportional to $\kappa(A) = \prod_{uv \in A} \kappa_{uv}$. Such prior allows giving separate weight for each arc according to its prior plausibility. The *uniform prior* is a special case where $\kappa_{uv} = 1$ for all node pairs uv. Note that proportionality guarantees modularity, since the n factors of the prior can absorb the normalizing constant $c = \sum_A \kappa(A)$, for example, by setting the vth factor to $\kappa_v(A_v)c^{-1/n}$. Another example is the prior proposed by Heckerman et al. (1995), that penalizes the distance between A and a user-defined prior DAG A^0. This can be obtained by letting $p(A)$ be proportional to $\prod_{v \in N} \kappa_v(A_v)$ where $\kappa_v(A_v) = \delta^{|(A_v \cup A_v^0) \setminus (A_v \cap A_v^0)|}$ and $0 < \delta \leq 1$ is a user-defined constant that determines the penalization strength. Modular structure priors also enable a straightforward way to bound the indegrees of the nodes. Indeed, we often consider the case where $p(A)$ vanishes if there exists a node v for which $|A_v|$ exceeds some fixed *maximum indegree*, denoted by k. Note, however, that modular structure priors do not allow similar controlling of the outdegrees of the nodes. For a node v we will call a set S a *potential parent set* if the prior does not vanish when S is the parent set of v, that is, $p(A) > 0$ for some structure A with $A_v = S$. Specifically, if p is modular, then S is a potential parent set of v if and only if $p_v(S) > 0$.

The following example describes a frequently used modular structure likelihood (Buntine, 1991; Heckerman et al., 1995):

Example 1 (Dirichlet–multinomial likelihood) Suppose that each set $\mathcal{X}_v$ is finite. Consider a fixed structure $A \subseteq N \times N$. For each node v and its parent set A_v, let the conditional distribution of D_v^j given $D_{A_v}^j = y$ be categorical with parameters $\theta_{v \cdot y} = \{\theta_{vxy} : x \in \mathcal{X}_v\}$. Define a distribution θ of D^j as the product of the conditional distributions and observe that (A, θ) is a Bayesian network. Assign each set of parameters $\theta_{v \cdot y}$ a Dirichlet prior with

parameters $r_{vxy} > 0$, for $x \in \mathcal{X}_v$, and compose the prior $p(\theta \,|\, A)$ by assuming independence of the components. Let m_{vxy} denote the number of data records j where $D_v^j = x$ and $D_{A_v}^j = y$. Then

$$\begin{aligned} p(D \,|\, A) &= \int p(\theta \,|\, A) p(D \,|\, A, \theta) \mathrm{d}\theta \\ &= \prod_{v \in N} \prod_{y \in \mathcal{X}_{A_v}} \frac{\Gamma(r_{v \cdot y})}{\Gamma(r_{v \cdot y} + m_{v \cdot y})} \prod_{x \in \mathcal{X}_v} \frac{\Gamma(r_{vxy} + m_{vxy})}{\Gamma(r_{vxy})}, \end{aligned}$$

where $r_{v \cdot y}$ and $m_{v \cdot y}$ are the sums of the numbers r_{vxy} and m_{vxy}, respectively, and Γ is the gamma function. In an important special case, each parameter r_{vxy} is set to a so-called *equivalent sample size* $\alpha \geq 0$ divided by the number of joint configurations of x and y (i.e., by $|X_v|$ if A_v is empty and by $|\mathcal{X}_v||\mathcal{X}_{A_v}|$ otherwise).

By a *modular model* we will refer to a Bayesian model, where both the structure likelihood and the structure prior are modular. We will assume that such a model is represented in terms of the factors λ_v and κ_v of a likelihood λ and an unnormalized prior κ, respectively. In our empirical study we have used the following simple model:

Example 2 (uniform Dirichlet–multinomial model, UDM) In this modular model, the structure prior is specified by letting $\kappa_v(A_v) = 1$ if $|A_v| \leq k$, and $\kappa_v(A_v) = 0$ otherwise. The likelihood is the Dirichlet–multinomial likelihood with the equivalent sample size parameter set to 1 (see Example 1). The maximum indegree k is a parameter of the model.

For a demonstration of the computational benefit of modularity, let us consider the probability (density) of the data D conditioning on the constraint that the unknown DAG A is compatible with a given linear order L on the nodes. We may express this compatibility constraint by writing simply L, and hence the quantity of interest by $p(D \,|\, L)$. Write first

$$p(D \,|\, L) = \sum_A p(A, D \,|\, L) = \frac{\sum_{A \subseteq L} p(A) p(D \,|\, A)}{\sum_{A \subseteq L} p(A)} = \frac{\sum_{A \subseteq L} \kappa(A) \lambda(A)}{\sum_{A \subseteq L} \kappa(A)}.$$

Here the last equality holds because the normalizing constants of the prior cancel out. Then we use the key implication of the constraint $A \subseteq L$, namely, that the set of possible structures A decomposes into a Cartesian product of the sets of potential parent sets A_v for each node v (Buntine, 1991; Cooper and Herskovits, 1992):

$$p(D \,|\, L) = \prod_{v \in N} \sum_{A_v \subseteq L_v} \kappa_v(A_v) \lambda_v(A_v) \Bigg/ \prod_{v \in N} \sum_{A_v \subseteq L_v} \kappa_v(A_v). \tag{4}$$

This factorization enables independent processing of the parent sets for each node, which amounts to significant computational savings compared to exhaustive enumeration of all possible structures. A similar factorization can be derived for the conditional posterior expectation $\mathbb{E}(f \,|\, D, L)$ of a modular feature f.

Motivated by the savings, Friedman and Koller (2003) addressed the unconstrained setting where no node ordering is given. They proposed averaging $\mathbb{E}(f \,|\, D, L)$ over a sample

of linear orders drawn from a distribution that is proportional to the marginal likelihood $p(D \mid L)$, as we will describe in the next subsection. For *exact* averaging over all linear orders, Koivisto and Sood (2004) gave exponential-time dynamic programming algorithm. We will obtain that algorithm as a special case of the algorithm we give in Section 5.

When the modularity is exploited as described above, the actual joint model of the data and structures does not remain modular. This is essentially because some DAGs are consistent with fewer linear orders than other DAGs. We will discuss this issue further at the end of the next subsection. To characterize the model, for which the node-ordering based methods work correctly, we follow Koivisto and Sood (2004) and call a model *order-modular* if the likelihood function is modular and the structure prior $p(A)$ is proportional to $\kappa(A) \sum_{L \supseteq A} \mu(L)$, for some modular functions κ and μ. Thus an order-modular model can be interpreted as a "modular" joint model for the structure A and the linear order L. Note, that if $\mu(L) > 0$ for all linear orders L, then the support of such a prior contains the same DAGs as the support of the corresponding modular prior determined by κ. We will also use the terms *order-modular prior* and *order-modular posterior* in an obvious manner.

Example 3 (order-modular UDM) In this model, the structure likelihood and the factors κ_v are as in a modular UDM (see Example 2), and the maximum indegree k is a parameter of the model. The difference to the modular UDM is that we specify instead an order-modular prior by letting $\mu(L) = 1$ for all linear orders L on N. Thus the prior $p(A)$ is proportional to $\kappa(A)\ell(A)$.

3.3 Sampling-based Approximations

We next review the basic sampling-based approaches for structure learning in Bayesian networks. For a broader introduction to the subject in the machine learning context, we refer to the survey by Andrieu et al. (2003).

Importance sampling methods provide us with a generic approach to approximate the expectation $\mathbb{E}(f \,|\, D)$ by a sample average

$$\frac{1}{T} \sum_{t=1}^{T} \frac{f(A^t) p(A^t \,|\, D)}{q(A^t)}\,, \qquad A^1, A^2, \ldots, A^T \sim q(A)\,, \tag{5}$$

where $q(A)$ is some appropriate *sampling distribution*. It would be desirable that (i) $q(A)$ is as close to the function $|f(A^t)p(A^t \,|\, D)|$ as possible, up to a multiplicative constant, and (ii) that the samples A^t are independent. In order to compute the average we also need, for any given sample A^t, (iii) an access to the value $q(A^t)$ either exactly or up to a small error. If the function q can be evaluated only up to a constant factor, then one typically uses the *self-normalized importance sampling estimate*, which is obtained by dividing the sample average (5) by the sample average of $1/q(A^t)$. In practice, it is possible to simultaneously satisfy only some of the desiderata (i–iii).

The *structure-MCMC* method of Madigan and York (1995), in particular, draws the samples by simulating a Markov chain whose stationary distribution is $p(A \,|\, D)$. Thus, while the sampling distribution tends to $p(A \,|\, D)$ in the limit, after a finitely many simulation steps there is no guarantee about the quality of the sampling distribution. Furthermore, since the samples are dependent, their number has to be relatively large to obtain the efficiency that

could be obtained with a smaller number of independent samples. Finally, the computation of $q(A^t)$ is avoided by assuming it to be a good approximation to $p(A^t \,|\, D)$ and thus canceling in the estimator. The performance of structure-MCMC depends crucially on the mixing rate of the Markov chain, that is, how fast the chain "forgets" its initial state so that the draws will be (approximately) from $p(A \,|\, D)$. The main shortcoming of structure-MCMC is that the chain may get easily trapped at small regions near local maxima of the posterior.

The *order-MCMC* method of Friedman and Koller (2003), which we already mentioned in the previous subsection, aims at better performance by sampling node orderings from a distribution that is proportional to $p(D \mid L)$. Compared to the space of DAGs, the space of node orderings is not only significantly smaller but it also smoothens the sampling distribution, because for any L the marginal likelihood $p(D \,|\, L)$ is a sum over exponentially many DAGs. The resulting estimator becomes

$$\frac{1}{T} \sum_{t=1}^{T} \mathbb{E}(f \,|\, D, L^t)\,, \qquad L^1, L^2, \ldots, L^T \sim p'(L \,|\, D)\,,$$

where $p'(L \,|\, D) \propto p(D \,|\, L) p'(L)$ is obtained via $p(D \,|\, L)$ by re-modelling the $n!$ possible node ordering constraints L as mutually exclusive events, assigned with a uniform prior, $p'(L)$. In cases where exact computation of the expectation $\mathbb{E}(f \,|\, D, L^t)$ is not feasible, it can, in turn, be approximated by a single evaluation at a sampled DAG,

$$f(A^t)\,, \qquad A^t \sim p(A \,|\, D, L^t)\,,$$

or, more accurately, by an average over several independent samples from $p(A \mid D, L^t)$. Importantly, the latter sampling step, if needed, is again computationally easy thanks to the fixed node ordering. Thus the difficulty of sampling only concerns the sampling of node orderings. Friedman and Koller (2003) showed that simple local changes, namely swaps of two nodes, yield a Markov chain that mixes relatively fast in their sets of experiments. We omit a more detailed description of the order-MCMC method here, as the method is obtained as a special instantiation of the method we will introduce in Section 4.

The innocent-looking treatment of node ordering constraints as mutually exclusive events, however, introduces a "bias" to the estimator. Indeed, it is easy to see that the samples A^t will be generated from the distribution

$$\sum_{L \supseteq A} p'(L \,|\, D) p(A \,|\, D, L) \propto p(A \,|\, D) \ell(A)\,.$$

In other words, compared to sampling from the true posterior $p(A \,|\, D)$, sampling via node orderings favors DAGs that are compatible with larger numbers of linear orders. But also a different interpretation is valid: the DAGs are sampled from the correct posterior under a modified (order-modular) model where the original, modular structure prior $p(A)$ is replaced by the order-modular prior that is proportional to $p(A)\ell(A)$ (as in Example 3). Oftentimes, the modularity of the structure prior is preferred, as it can express priors that are uniform over all DAGs (subject to a maximum indegree constraint). That being said, Friedman and Koller (2003) give supportive arguments also for the other viewpoint.

4. Sampling Partial Orders

Our key idea is to extend the order-MCMC of Friedman and Koller (2003) by replacing the state space of node orderings by an appropriately defined space of partial orders on the nodes. Throughout this section we will denote (perhaps counter to the reader's anticipation) by π' the modular posterior distribution and by π its "biased" order-modular counterpart that arises due to treating node orderings as mutually exclusive events. Our goal is to perform Bayesian inference under the modular posterior π'. Because our approach is to sample from the order-modular posterior π, we obtain, as a by-product, also an inference method for order-modular models. When needed, we will refer to the underlying modular model by p' and to the induced order-modular model by p.

4.1 Sampling Spaces of Partial Orders

We will consider sampling from a state space that consists of partial orders on the node set N. We will denote the state space by $\mathcal{P}$. The idea is that the states in $\mathcal{P}$ partition the set of all linear orders on N. To this end, we require $\mathcal{P}$ to be an *exact cover* on N, that is, every linear order on N must be an extension of exactly one partial order P in $\mathcal{P}$. Examples 4 and 5 below illustrate the concept and also introduce the notion of a bucket order, which is central in our implementation of the proposed methods.

Example 4 (bucket orders) A partial order B is a *bucket order* if the ground set admits a partition into pairwise disjoint sets, $B_1, B_2, \ldots, B_h$, called the *buckets*, such that $uv \in B$ if and only if $u = v$ or $u \in B_i$ and $v \in B_j$ for some $i < j$. Intuitively, the order of elements in different buckets is determined by the buckets' order, while within each bucket the elements are incomparable. We say that the bucket order is of *type* $(b_1, b_2, \ldots, b_h)$, when b_i is the size of B_i. We call the bucket order a *balanced bucket order* with a *maximum bucket size* b if $b = b_1 = b_2 = \cdots = b_{h-1} \geq b_h$. Furthermore, we call two bucket orders *reorderings* of each other if they have the same ground set and they are of the same type. It is immediate that the set (equivalence class) of reorderings of a bucket order P constitute an exact cover on their common ground set (Koivisto and Parviainen, 2010). For a later reference, we also note that the number of downsets of the bucket order is $1 + \sum_i (2^{b_i} - 1)$.

The next example gives a straightforward extension of bucket orders.

Example 5 (parallel bucket orders) A partial order P is a parallel composition of bucket orders, or *parallel bucket order* for short, if P can be partitioned into r bucket orders $B^1, B^2, \ldots, B^r$ on disjoint ground sets. We call two parallel bucket orders P and Q *reorderings* of each other if their bucket orders can be labelled as $P^1, P^2, \ldots, P^r$ and $Q^1, Q^2, \ldots, Q^r$ such that each P^j is a reordering of Q^j. It is known that the set (equivalence class) of reorderings of a parallel bucket order P is an exact cover on their common ground set (Koivisto and Parviainen, 2010). Compared to bucket orders, parallel bucket orders make it possible to obtain better time–space tradeoffs in the context of exact structure discovery. However, our preliminary calculations (Niinimäki et al., 2011) show that parallel bucket orders are unlikely to yield substantive advantages in the sampling context.

Because the methods we shall consider are based on local moves in the sampling space, we equip the sampling space with a neighborhood structure. Formally, a neighborhood

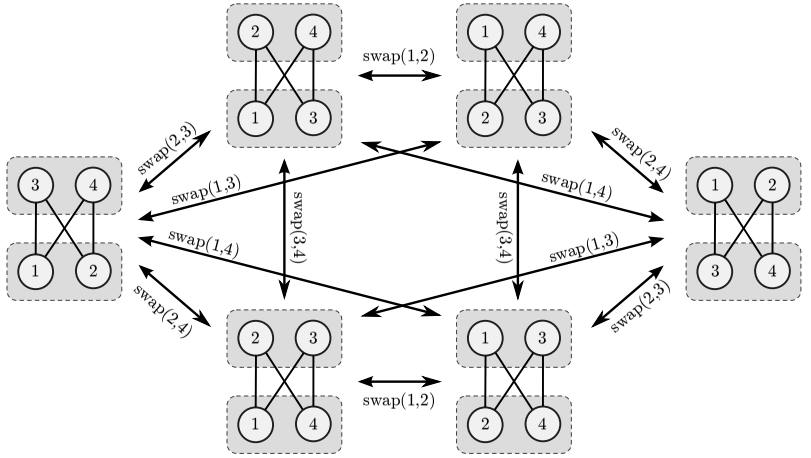


Figure 4: All reorderings of a bucket order of type $(2, 2)$ on the node set $\{1, 2, 3, 4\}$. Adjacency in the swap neighborhood is indicated by arrows labeled by the corresponding swap operation. Each bucket order is visualized using a Hasse diagram, with rectangles indicating individual buckets.

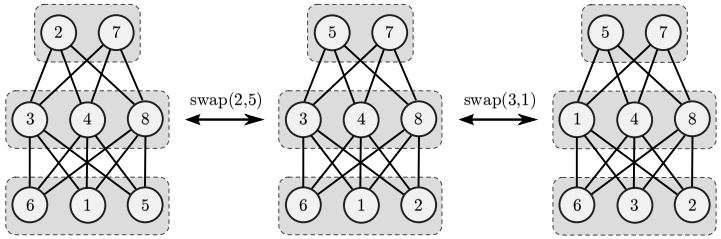


Figure 5: Three adjacent states in the space of reorderings of a bucket order of type $(3, 3, 2)$.

structure on $\mathcal{P}$ is just a graph on $\mathcal{P}$, the adjacency relation specifying the neighborhood relation. Here, we do not make an attempt to give any specific neighborhood structure that would be appropriate for an arbitrary exact cover $\mathcal{P}$. Instead, we continue with an example:

Example 6 (swap neighborhood) Let P be a (parallel) bucket order on N, and let $\mathcal{P}$ the set of reorderings of P. The *swap neighborhood* on $\mathcal{P}$ is a graph in which the vertex set consists of the reorderings $\mathcal{P}$ and two members $Q, R \in \mathcal{P}$ are adjacent if Q is obtained from

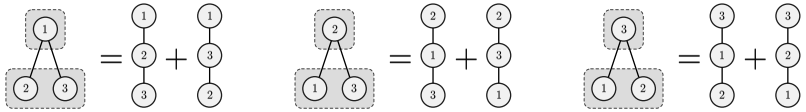


Figure 6: There are in total six linear orders on the node set $\{1, 2, 3\}$ and three reorderings of a bucket order of type (2,1). As shown in the figure, the probability of each bucket order is the sums of the probabilities of its linear extensions. Observe, how the bucket orders form an exact cover and hence partition the set of linear orders into three disjoint subsets.

R by *swapping* two nodes $s, t \in N$, more formally:

$$uv \in R \iff \sigma(u)\sigma(v) \in Q\,,$$

where σ is the transposition $N \to N$ that swaps s and t. Clearly, the swap neighborhood is connected. See Figures 4 and 5 for an illustration of bucket orders and swap neighborhoods.

For each partial order P in an exact cover $\mathcal{P}$, the posterior probability $\pi(P)$ is obtained simply as the sum of the posterior probabilities $\pi(L)$ of all linear extensions L of P. This is illustrated in Figure 6. Note that the choice of the state space $\mathcal{P}$ does not affect the posterior π and consequently leaves the bias untouched. The correction of the bias will be taken care of separately, as shown in the next subsection.

4.2 Partial-Order-MCMC and Bias Correction

The basic *partial-order-MCMC* method has three steps. It first samples states from $\mathcal{P}$ along a Markov chain whose stationary distribution is π. Then it draws a DAG from each sampled partial order. Finally, it estimates the posterior expectation of the feature in interest by taking a weighted average of the samples. In more detail, these steps are as follows:

1. *Sample partial orders along a Markov chain using the Metropolis–Hastings algorithm.* Start from a random partial order $P^1 \in \mathcal{P}$. To move from state P^t to the next state, first draw a candidate state $P^\star$ from a proposal distribution $q(P^\star \,|\, P^t)$. Then accept the candidate with probability

$$\min\left\{1, \frac{\pi(P^\star)q(P^t \,|\, P^\star)}{\pi(P^t)q(P^\star \,|\, P^t)}\right\},$$

   and let $P^{t+1} = P^\star$; otherwise let $P^{t+1} = P^t$. This produces a sample of partial orders $P^1, P^2, \ldots, P^T$. Each move constitutes one *iteration* of the algorithm.

2. *Sample DAGs from the sampled partial orders.* For each sampled partial order P^t, draw a DAG A^t compatible with P^t from the conditional posterior distribution $\pi(A^t \,|\, P^t)$. This produces a sample of DAGs $A^1, A^2, \ldots, A^T$.

3. *Estimate the expected value of the feature by a weighted sample average.* Return

$$f_{\text{MCMC}} = \sum_t \frac{f(A^t)}{\ell(A^t)} \Bigg/ \sum_t \frac{1}{\ell(A^t)}$$

   as an estimate of $\mathbb{E}_{\pi'}(f)$. Recall that $\ell(A^t)$ is the number of linear extensions of A^t.

Several standard MCMC techniques can be applied to enhance the method in practice. Specifically, it is computationally advantageous to thin the set of samples P^t by keeping only, say, every 100th sample. Also, it is often a good idea to start collecting the samples only after a burn-in period that may cover, say, as much as 50% of the total allocated running time. On the other hand, we can compensate these savings in step 2 of the method by drawing *multiple*, independent DAGs per sampled partial order P^t, which can improve considerably the estimate obtained in step 3. We will consider these techniques in more detail in the context of our empirical study in Section 6.

It is worth noting that access to the exact posterior probabilities $\pi(P^t)$ are not needed in step 1. It suffices that we can evaluate a function g that is proportional to π. An appropriate function is given by

$$g(P^t) = \sum_{L \supseteq P^t} \sum_{A \subseteq L} p'(A, D)\,. \tag{6}$$

We will see later that the modularity of the model p' enables fast computation of $g(P^t)$.

We can show that under mild conditions on the proposal distribution q, the Markov chain constructed in step 1 converges to the target distribution $\pi(P)$, and consequently, the estimate f_{MCMC} tends to $\mathbb{E}_{\pi'}(f)$ as the number of samples T grows. In the next paragraphs we justify these two claims.

For the first claim it suffices to show that the chain is irreducible, that is, from any state P the chain can reach any other state $P^\star$ with some number of moves (with a positive probability). In principle, this condition would be easily satisfied by a proposal distribution $q(P^\star \mid P)$ whose support is the entire $\mathcal{P}$ for every P. From a practical point of view, however, it is essential for good mixing of the chain to have a proposal distribution that concentrates the proposes locally to a few *neighbors* of the current state P. In that case it is crucial to show that the induced neighborhood graph on $\mathcal{P}$ is strongly connected, thus implying irreducibility of the chain. In the order-MCMC method, connectedness was obtained because any two node orderings can be reached from each other by some number of swaps of two nodes. It is easy to see that this proposal distribution applies to partial orders as well, only noting that some swaps of nodes may result in partial orders that are outside the fixed state space $\mathcal{P}$ and must thus be rejected (or avoid proposing). Rather than pursuing the issue in full generality, we extend the swap proposal for the sampling space of parallel bucket orders:

Example 7 (swap proposal for parallel bucket orders) Consider the set of reorderings $\mathcal{P}$ and the swap-neighborhood described in Example 6. For any $P \in \mathcal{P}$, let the conditional proposal distribution $q(P^\star \,|\, P)$ be uniform over the neighbors $P^\star$ of P (and vanish otherwise). Note that it is possible to sample directly from this conditional distribution by drawing two nodes s, t that belong to the same part but different buckets in the partition of N, uniformly at random, and proposing the swap of s and t.

We then turn to the second claim that the estimate tends to the posterior expectation of the feature. We investigate the behavior of the estimate f_{MCMC}. By the properties of the Markov chain we may assume that the $P^1, P^2, \ldots, P^T$ are an ergodic sample from $\pi(P)$, that is, any sample average tends to the corresponding expected value as T grows. Consequently, the $A^1, A^2, \ldots, A^T$ are an ergodic sample from the biased posterior

$$\pi(A) = \sum_{P \in \mathcal{P}} \pi(P)\, \pi(A \,|\, P) = \sum_{P \in \mathcal{P}} \sum_{L \supseteq P} \pi(P, L, A) = \sum_{L} \pi(L, A) \propto \pi'(A)\ell(A)\,.$$

Here the last equation holds because $\mathcal{P}$ is an exact cover on N; and the last proportionality holds because $\pi(L, A)$ vanishes if L is not an extension of A, and is otherwise proportional to $\pi'(A)$. The ergodicity now guarantees that the average of the $1/\ell(A^t)$ tends to the expectation $\mathbb{E}_\pi(1/\ell(A^t)) = 1/c$, where

$$c = \sum_{A} \pi'(A)\ell(A)\,, \tag{7}$$

and that the average of the $f(A^t)/\ell(A^t)$ tends to the ratio $\mathbb{E}_{\pi'}(f)/c$. This implies that f_{MCMC} approaches $\mathbb{E}_{\pi'}(f)$ as the number of samples T grows.

The computational complexity of partial-order-MCMC is determined, in addition to the number of samples T, by the complexity of the problems solved for each sampled partial order P^t and DAG A^t. These problems—of which analysis we postpone to Section 5—are:

(a) *Unnormalized posterior:* Compute $g(P^t)$ for a given P^t.

(b) *Sample DAGs:* Draw a DAG A^t from $\pi(A^t \,|\, P^t)$ for a given partial order P^t .

(c) *Number of linear extensions:* Compute $\ell(A^t)$ for a given DAG A^t.

We will see that problems (a) and (b) can be solved in time that scales, roughly, as $C + |\mathcal{D}|$, where C is the total number of potential parent sets and $|\mathcal{D}|$ is the number of downsets of the partial order P^t.

The problem (c) of counting the linear extensions was already discussed in Section 2, Corollary 4. We note that if the target model is order-modular, then there is no need to compute the terms $\ell(A^t)$, as no bias correction is needed. Moreover, for an order-modular model also the DAG sampling step can be avoided if the interest is in a modular feature f. Namely, then the estimate is obtained simply as an average of the conditional expectations $\mathbb{E}_\pi(f \,|\, P^t)$, which gives us one more problem to solve:

(a') *Expectation of a modular feature:* Compute $\mathbb{E}_\pi(f \,|\, P^t)$ for a given partial order P^t.

We will see in Section 5 that this problem can be computed in almost the same way as the unnormalized posterior probability of P^t, which justifies the label (a').

4.3 Metropolis-coupled Markov Chain Monte Carlo (MC^3)

Tempering techniques can enhance mixing of the chain and, as a byproduct, they also offer a good estimator for the marginal likelihood $p(D)$. Here we consider one such technique, *Metropolis-coupled MCMC* (MC^3) (Geyer, 1991). In MC^3, several Markov chains, indexed

by $0, 1, \ldots, K$, are simulated in parallel, each chain i having its own stationary distribution π_i. The idea is to take π_0 as a "hot" distribution, for example, the uniform distribution, and then let the π_i be increasingly "cooler" and closer approximations of the posterior π, putting finally $\pi_K = \pi$. Usually, powering schemes of the form

$$\pi_i \propto \pi^{\beta_i}\,, \quad 0 \leq \beta_0 < \beta_1 < \cdots < \beta_K = 1$$

are used. For instance, Geyer and Thompson (1995) suggest harmonic stepping, $\beta_i = 1/(K + 1 - i)$; in our experiments we have used linear stepping, $\beta_i = i/K$.

In addition to running the chains in parallel, every now and then we propose a swap of the states P_i and P_j of two randomly chosen chains i and $j = i + 1$. The proposal is accepted with probability

$$\min\left\{1, \frac{\pi_i(P_j)\pi_j(P_i)}{\pi_i(P_i)\pi_j(P_j)}\right\}.$$

We note that each π_i needs to be known only up to some constant factor, that is, it suffices that we can efficiently evaluate a function g_i that is proportional to π_i. By using samples from the coolest chain only, an estimate of the expectation $\mathbb{E}_{\pi'}(f)$, which we denote by f_{MC3}, is obtained by following steps 2 and 3 of the partial-order-MCMC method. In Section 4.5 we will show that, by using samples from all chains, we can also get good estimates of the marginal likelihood $p'(D)$. This technique is a straightforward extension of the technique for estimating $p(D)$.

4.4 Annealed Importance Sampling (AIS)

AIS produces independent samples of partial orders $P^1, P^2, \ldots, P^T$ and associated importance weights $w^1, w^2, \ldots, w^T$. Like in MC^3, a sequence of distributions $\pi_0, \pi_1, \ldots, \pi_K$ is introduced, such that sampling from π_0 is easy, and as i increases, the distributions π_i provide gradually improving approximations to the posterior distribution π, until finally π_K equals π. In our experiments we have used the same scheme as for MC^3, however, with a much larger value of K. For each π_i we assume the availability of a corresponding function g_i that is proportional to π_i and that can be evaluated fast at any given point.

To sample P^t, we first sample a sequence of partial orders $P_0, P_1, \ldots, P_{K-1}$ along a Markov chain, starting from π_0 and moving according to suitably defined transition kernels τ_i, as follows:

Generate P_0 from π_0.
Generate P_1 from P_0 using τ_1.
$\vdots$
Generate P_{K-1} from P_{K-2} using τ_{K-1}.

The transition kernels τ_i are constructed by a simple Metropolis–Hastings move: At state P_{i-1} a candidate state $P_\star$ is drawn from a proposal distribution $q(P_\star \,|\, P_{i-1})$; the candidate is accepted as the state P_i with probability

$$\min\left\{1, \frac{g_i(P_\star)q(P_{i-1} \,|\, P_\star)}{g_i(P_{i-1})q(P_\star \,|\, P_{i-1})}\right\},$$

and otherwise P_i is set to P_{i-1}. It follows that the transition kernel τ_i leaves π_i invariant. Finally, we set $P^t = P_{K-1}$ and assign the importance weight as

$$w^t = \frac{g_1(P_0)}{g_0(P_0)} \frac{g_2(P_1)}{g_1(P_1)} \cdots \frac{g_K(P_{K-1})}{g_{K-1}(P_{K-1})} .$$

We then generate a DAG A^t from each P^t as in step 2 of the partial-order-MCMC method. An estimate of $\mathbb{E}_{\pi'}(f)$ is given by

$$f_{\text{AIS}} = \sum_t \frac{w^t f(A^t)}{\ell(A^t)} \Bigg/ \sum_t \frac{w^t}{\ell(A^t)} . \tag{8}$$

To see that this self-normalized importance sampling estimate is consistent, we examine separately the expected values of the numerator and the denominator, and show that their ratio equals $\mathbb{E}_{\pi'}(f)$. To this end, consider a fixed t and any function h of partial orders. Denote by q the joint sampling distribution of the partial orders $P_0^t, P_1^t, \ldots, P_K^t$ and the DAG A^t. The general result of Neal (2001) implies the following: Let ρ denote the ratio of the normalizing constants of g_K and g_0. Then

$$\mathbb{E}_q\big(w^t h(P_K^t)\big) = \rho \cdot \mathbb{E}_\pi\big(h(P_K^t)\big) \quad \text{and} \quad \mathbb{E}_q\big(w^t\big) = \rho .$$

Using the fact that $\mathbb{E}_q\big(w^t\, h'(A^t)\big) = \mathbb{E}_q\big(w^t\, \mathbb{E}_q(h'(A^t) \,|\, P_0^t, \ldots, P_K^t)\big) = \mathbb{E}_q\big(w^t\, \mathbb{E}_\pi(h'(A^t) \,|\, P_K^t)\big)$ and applying the above result with a particular choice of the functions h and h' yields

$$\mathbb{E}_q\left(w^t \cdot \frac{f(A^t)}{\ell(A^t)}\right) = \mathbb{E}_q\left(w^t \cdot \mathbb{E}_\pi\left(\frac{f(A^t)}{\ell(A^t)} \,\middle|\, P_K^t\right)\right) = \rho \cdot \mathbb{E}_\pi\left(\frac{f(A^t)}{\ell(A^t)}\right) = \frac{\rho}{c} \cdot \mathbb{E}_{\pi'}\big(f(A^t)\big) ,$$

where c is, as given before in (7), the normalizing constant of $\pi'(A)\ell(A)$. From this we also see that the expected value of each term in the denominator in (8) equals ρ/c. Thus the ratio of the expectations is $\mathbb{E}_{\pi'}(f)$ as desired.

4.5 Estimating the Marginal Likelihood

We now turn to the estimation of the marginal likelihood $p'(D)$ using samples produced by either MC^3 or AIS. We will view $p'(D)$ as the normalizing constant of the function $g'(A) = p'(A, D)$, and denote the constant by c' for short. We will estimate c' indirectly, by estimating a ratio $\rho' = c'/c_0$, where c_0 is another normalizing constant that we can compute exactly. In fact, c_0 will be the normalizing constant g_0/π_0, and in general, we will denote by c_i the normalizing constant g_i/π_i.

With a sample generated by MC^3, our estimate for the marginal likelihood is obtained, in essence, as a product of estimates of the ratios c_{i+1}/c_i, as given by

$$\rho'_{\text{MC3}} = \prod_{i=0}^{K-1} \left(\frac{1}{T} \sum_t \frac{g_{i+1}(P_i^t)}{g_i(P_i^t)}\right) \left(\frac{1}{T} \sum_t \frac{1}{\ell(A^t)}\right) .$$

To see that the estimate is asymptotically unbiased, observe first that

$$\mathbb{E}_{\pi_i}\left(\frac{g_{i+1}(P_i^t)}{g_i(P_i^t)}\right) = \frac{c_{i+1}}{c_i} \quad \text{and} \quad \mathbb{E}_\pi\left(\frac{1}{\ell(A^t)}\right) = \frac{c'}{c_K} .$$

Here the former equation is easy to verify. For the latter we recall that $\mathbb{E}_\pi\big(1/\ell(A^t)\big) = 1/c$ and write the normalizing constant of $g_K = g$ using (6) as

$$c_K = \sum_P g(P) = \sum_L \sum_{A \subseteq L} p'(A, D) = p'(D) \sum_A \pi'(A)\ell(A) = c'c .$$

Now, if the estimates were independent and, moreover, the samples P_i^t were exactly from π_i, then ρ'_{MC3} would be an unbiased estimate of the marginal likelihood. While neither condition is satisfied in our case, the ergodicity of the chains guarantees that each estimate, and thereby their product, is asymptotically unbiased.

With a sample generated by AIS, our estimate for the marginal likelihood is

$$\rho'_{\text{AIS}} = \frac{1}{T} \sum_t \frac{w^t}{\ell(A^t)} .$$

It is not difficult to see that this estimate is unbiased. Namely, we have already seen that the expected value of this estimate is ρ/c, where $\rho = c_K/c_0$. Because we just showed that $1/c = c'/c_K$, we obtain $\rho/c = c'/c_0 = \rho'$, as desired.

The AIS-based estimate has two main advantages over the MC^3-based estimate. One is that we can use a fairly large number of steps K in AIS, which renders the estimate more accurate. In MC^3 we have to use a much smaller K to reserve time for simulating each chain a large number of steps. A smaller K is expected to yield less accurate estimates. The other advantage of the AIS-based estimate stems from the unbiasedness and independence of the samples. Indeed, these two properties allow us to compute high-confidence *lower bounds* for the marginal likelihood. We will make use the following elementary theorem; for variations and earlier uses in other contexts, we refer to the works of Gomes et al. (2007) and Gogate and Dechter (2011).

Theorem 14 (lower bound) *Let $Z_1, Z_2, \ldots, Z_s$ be independent nonnegative random variables with mean μ. Let $0 < \delta < 1$. Then, with probability at least $1 - \delta$, we have*

$$\delta^{1/s} \min\{Z_1, Z_2, \ldots, Z_s\} \leq \mu .$$

Proof By Markov's inequality, $Z_i > \delta^{-1/s}\mu$ with probability at most $\delta^{1/s}$, for each i. Taking the product gives that $\min\{Z_1, Z_2, \ldots, Z_s\} > \delta^{-1/s}\mu$ with probability at most δ. To complete the proof, multiply both sides by $\delta^{1/s}$ and consider the complement event. ■

We apply this result by dividing our T samples into s bins of equal size and letting Z_i be the estimate of the marginal likelihood based on the samples in the ith bin. There is a tradeoff in choosing a good value of s. Namely, to obtain good individual estimates Z_i, we would like to set s as small as possible. On the other hand, we would like to use a large s in order to have a *slack* factor $\delta^{1/s}$ as close to 1 as possible. The following examples show two different ways to address this tradeoff.

Example 8 (slack-2 lower bound) Put $\delta = 2^{-5} = 0.03125$ and $s = 5$. Then $\delta^{1/s} = 1/2$.

Example 9 (square-root lower-bounding scheme) Put $\delta = 2^{-5}$ and $s = \lfloor \sqrt{T} \rfloor$ for T samples. Then $\delta^{1/s}$ grows with T, being 2^{-1}, $2^{-1/2}$, $2^{-1/4}$ at $T = 25, 100, 400$, respectively.

5. Per-Sample Computations

In the previous section we encountered a number of computational problems associated with each sampled partial order and DAG (see the end of Section 4.2). In this section we give algorithms to solve those problems. We begin by formulating the computational problems and stating the main results in Section 5.1. The proofs are given in Sections 5.2–5.4. Finally, in Section 5.5 we discuss the possibility to reduce the time and space requirements in certain special cases that are relevant for the present applications.

5.1 Problems and Results

We shall derive solutions to the computational problems (a), (b), and (a') of Section 4.2 as specific instantiations of slightly more abstract problems concerning modular functions. Recall that the problem (c) was already discussed in Section 2.

We abstract the core algorithmic problem underlying problems (a) and (a') as what we call the *DAG-extensions* (DAGE) problem, defined as follows. As input we are given a modular function φ that associates each DAG on N with a real number. We assume that each factor φ_v, for $v \in N$, is given explicitly as a list of argument–value pairs $(X, \varphi_v(X))$ where X runs through some collection $\mathcal{C}_v$ of subsets of $N \setminus \{v\}$. We further assume that the factor vanishes outside this collection. As input we are also given a partial order P on N. Our task is to compute the value $\varphi(P)$ defined by

$$\varphi(P) = \sum_{L \supseteq P} \sum_{A \subseteq L} \varphi(A)\,.$$

We denote by C the sum of the sizes $|\mathcal{C}_v|$, and by $\mathcal{D}$ the set of downsets of P.

Problems (a) and (a') reduce to the DAGE problem: We obtain the unnormalized posterior probability $g(P)$ as $\varphi(P)$ by letting $\varphi(A) = \kappa(A)\lambda(A)$. The collections $\mathcal{C}_v$ consists of the potential parent sets of node v. Similarly, we obtain the expectation $\mathbb{E}_\pi(f \mid P)$ of a modular feature f as a ratio $\varphi^f(P)/g(P)$ by letting $\varphi^f(A) = f(A)\kappa(A)\lambda(A)$.

In the next subsection we prove:

Theorem 15 (DAG-extensions) *Given a partial order P on N and a modular function φ over N, we can compute $\varphi(P)$ in $O\big(n^2(C + |\mathcal{D}|)\big)$ time and $O\big(n(C + |\mathcal{D}|)\big)$ space.*

To address problem (b), we define the *DAG sampling* problem as follows. Our input is as in the DAGE problem, except that we are also given a number T. Our task is to sample T independent DAGs from a distribution that is proportional to $\varphi(A)\ell(A \cup P)$. Observe that $\varphi(P)$ can be written as a sum of $\varphi(A)\ell(A \cup P)$ over all DAGs A on N. Problem (b) reduces to the DAG sampling problem in an obvious manner.

In Section 5.3 we prove:

Theorem 16 (DAG sampling) *Given a partial order P on N, a modular nonnegative function φ over N, and a number $T > 0$, we can draw T independent DAGs A on N proportionally to $\varphi(A)\ell(A \cup P)$ in $O\big(n^2(C + |\mathcal{D}| + T)\big)$ time and $O\big(nC + n^2|\mathcal{D}|\big)$ space.*

We also consider the following variant of the DAGE problem, which we call the *arc probabilities* problem. Our input is as in the DAGE problem. For a pair of nodes $s, t \in N$,

define φ^{st} as the modular function obtained from φ by setting each factor as

$$\varphi_v^{st}(A_v) \equiv \begin{cases} 0 & \text{if } t = v \text{ and } s \notin A_v\,, \\ \varphi_v(A_v) & \text{otherwise}\,. \end{cases}$$

Our task is to compute the values $\varphi^{st}(P)$ for all node pairs st. This problem models the task of computing the posterior probabilities of all arcs: we see that $\varphi^{st}(P)/g(P)$ equals $\mathbb{E}_\pi(f)$ when we set $\varphi(A) = \kappa(A)\lambda(A)$, and $f(A) = 1$ if $st \in A$ and $f(A) = 0$ otherwise.

In Section 5.4 we prove:

Theorem 17 (arc probabilities) *Given a partial order P on N and a modular function φ over N, we can compute the values $\varphi^{st}(P)$ for every pair of two nodes $s, t \in N$ simultaneously in $O\big(n^2(C + |\mathcal{D}|)\big)$ time and $O\big(n(C + |\mathcal{D}|)\big)$ space.*

5.2 Proof of Theorem 15

We prove Theorem 15 by giving an algorithm that evaluates $\varphi(P)$ in the claimed time and space. The algorithm consists of two phases, which stem from the sum–product expression

$$\varphi(P) = \sum_{L \supseteq P} \prod_v \alpha_v(L_v)\,, \quad \text{where} \quad \alpha_v(L_v) = \sum_{A_v \subseteq L_v} \varphi_v(A_v)\,. \tag{9}$$

This expression is obtained by applying the same decomposition that was used to obtain factorization (4). In the first phase of the algorithm, we compute the values $\alpha_v(L_v)$ for each $v \in N$ and every relevant subset $L_v \subseteq N \setminus \{v\}$—we will see that only the downsets of P can be relevant. In the second phase, we compute the sum over all linear extensions of P by dynamic programming across the downsets of P, now assuming efficient access to each (precomputed) value $\alpha_v(L_v)$.

Let us first consider the first phase for a fixed node v. We observe that the problem of computing the values $\alpha_v(Y)$ for all downsets $v \notin Y \in \mathcal{D}$ reduces trivially to the deflating zeta transform problem, and can thus, by Theorem 8, be solved in $O(n^2|\mathcal{C}_v| + n|\mathcal{D}|)$ time and $O(n(|\mathcal{C}_v| + |\mathcal{D}|))$ space. Summing the time bounds over the n nodes yields a time bound of $O(n^2(C + |\mathcal{D}|)$ in total. Because the working space can be reused for different nodes v, the space requirement is dominated by the input and the output size, which is $O(n(C + |\mathcal{D}|)$ in total.

Consider then the second phase. Define the function F from $\mathcal{D}$ to real numbers by letting $F(\emptyset) = 1$ and for nonempty $Y \in \mathcal{D}$ recursively:

$$F(Y) = \sum_{\substack{v \in Y \\ Y \setminus \{v\} \in \mathcal{D}}} \alpha_v\big(Y \setminus \{v\}\big)\, F\big(Y \setminus \{v\}\big)\,. \tag{10}$$

Lemma 18 below shows that $F(N) = \varphi(P)$. Thus $\varphi(P)$ can be evaluated in $O(n|\mathcal{D}|)$ time and space using the covering graph of the downset lattice (Theorem 1).

Lemma 18 *We have $F(N) = \varphi(P)$.*

Proof For any subset $Y \subseteq N$ denote by $P[Y]$ the induced partial order $\{xy \in P : x, y \in Y\}$. We show by induction on the size of Y that

$$F(Y) = \sum_{L \supseteq P[Y]} \prod_{v \in Y} \alpha_v(L_v)\,,$$

where the sum is over all linear extensions of $P[Y]$.

For the base case, consider an arbitrary singleton $Y = \{v\} \in \mathcal{D}$. From the definition we get that $F(Y) = \alpha_v(\emptyset)F(\emptyset) = \alpha_v(\emptyset)$. Likewise, the induction claim evaluates to $F(Y) = \alpha_v(\emptyset)$, as $P[Y] = \{vv\}$.

For the induction step, let $\emptyset \neq Y \in \mathcal{D}$. We write the induction claim as

$$\begin{aligned}\sum_{L \supseteq P[Y]} \prod_{v \in Y} \alpha_v(L_v) &= \sum_{u \in \max Y} \alpha_u\big(Y \setminus \{u\}\big) \sum_{L' \supseteq P[Y \setminus \{u\}]} \prod_{v \in Y \setminus \{u\}} \alpha_v(L'_v) \\ &= \sum_{\substack{u \in Y \\ Y \setminus \{u\} \in \mathcal{D}}} \alpha_u\big(Y \setminus \{u\}\big)\, F\big(Y \setminus \{u\}\big)\,,\end{aligned}$$

which equals $F(Y)$ by the recursive definition (10). ■

5.3 Proof of Theorem 16

Consider the following algorithm. First solve the corresponding instance of the DAGE problem as described in the proof of Theorem 15. Store the intermediate results in the way described in Section 2.4. This takes $O\big(n^2(C + |\mathcal{D}|)\big)$ time and $O\big(nC + n^2|\mathcal{D}|\big)$ space. (Now we do not reuse the space.)

Then, to generate one of the T independent samples, do the following:

1. Generate a random linear extension $L \supseteq P$ proportionally to $\prod_v \alpha_v(L_v)$ by stochastically backtracking the recurrence (10). Using the Alias method (Theorem 13) this takes only $O(n)$ time and space, since there are n recursive steps.

2. For each node $v \in N$, generate a random parent set $A_v \in \mathcal{C}_v \cap 2^{L_v}$ by stochastically backtracking the deflating zeta transform as described in Section 2.4. This takes $O(n^2)$ time and space by the arguments given in Section 2.4.

3. Output the obtained DAG A.

Thus the algorithm has the claimed complexity in total.

5.4 Proof of Theorem 17

We will derive an algorithm that solves the problem in the claimed time and space. The algorithm extends the forward–backward algorithm of Koivisto (2006) to accommodate the partial order constraint.

Let the functions α_v, for each node $v \in N$, be as defined in the proof of Theorem 15. Define the "forward" function $F : \mathcal{D} \to \mathbb{R}$ as in definition (10), that is, by letting $F(\emptyset) = 1$

and, recursively,

$$F(Y) = \sum_{\substack{v \in Y \\ Y \setminus \{v\} \in \mathcal{D}}} F\big(Y \setminus \{v\}\big)\, \alpha_v\big(Y \setminus \{v\}\big)\,, \quad \text{for } \emptyset \subset Y \in \mathcal{D}\,.$$

Likewise, define the "backward" function $B : \mathcal{D} \to \mathbb{R}$ by letting $B(N) = 1$ and, recursively,

$$B(Y) = \sum_{\substack{v \in N \setminus Y \\ Y \cup \{v\} \in \mathcal{D}}} \alpha_v\big(Y\big)\, B\big(Y \cup \{v\}\big)\,, \quad \text{for } N \supset Y \in \mathcal{D}\,.$$

Furthermore, for each node $t \in N$, let

$$\gamma_t(A_t) = \sum_{\substack{Y \in \mathcal{D} \\ Y \supseteq A_t}} F\big(Y\big)\, B\big(Y \cup \{t\}\big)\,, \quad \text{for } A_t \in \mathcal{C}_t\,.$$

Lemma 19 *It holds that*

$$\varphi^{st}(P) = \sum_{s \in A_t \in \mathcal{C}_t} \varphi_t(A_t)\, \gamma_t(A_t)\,. \tag{11}$$

Proof Consider a fixed pair of nodes $s, t \in N$. Starting from (9), write

$$\varphi^{st}(P) = \sum_{L \supseteq P} \left(\sum_{A_t \subseteq L_t} \varphi_t^{st}(A_t) \right) \prod_{v \neq t} \alpha_v(L_v) = \sum_{A_t \subseteq N \setminus \{t\}} \varphi_t^{st}(A_t) \overbrace{\sum_{\substack{L \supseteq P \\ L_t \supseteq A_t}} \prod_{v \neq t} \alpha_v(L_v)}^{\text{Denote by } \gamma'_t(A_t).}\,.$$

Note, that $\varphi_t^{st}(A_t)$ vanishes unless $s \in A_t$ and otherwise equals $\varphi_t(A_t)$, which in turn vanishes unless $A_t \in \mathcal{C}_t$. Thus, it remains to show that the just-introduced function γ'_t equals γ_t. To see this, we split the sum over $L \supseteq P$ and $L_t \supseteq A_t$ into two nested sums that first iterate over $L_t \supseteq A_t$ such that $L_t \in \mathcal{D}$ and then over L_v for $v \neq t$. Furthermore, once L_t is fixed in the outer sum, then the inner sum must have $L_v \subset L_t$ for $v \in L_t$ and $L_v \supseteq L_t \cup \{t\}$ for $v \in N \setminus (L_t \cup \{t\})$. The inner sum can thus be split into two independent sums, as follows:

$$\gamma'_t(A_t) = \sum_{\substack{L_t \supseteq A_t \\ L_t \in \mathcal{D}}} \left(\sum_{L' \supseteq P[L_t]} \prod_{v \in L_t} \alpha_v(L'_v) \right) \left(\sum_{L' \supseteq P[N \setminus L_{t+}]} \prod_{v \in N \setminus L_{t+}} \alpha_v(L'_v \cup L_{t+}) \right),$$

where L_{t+} is a shorthand for $L_t \cup \{t\}$. To complete the proof, we have to show that

$$F(L_t) = \sum_{L' \supseteq P[L_t]} \prod_{v \in L_t} \alpha_v(L'_v)$$

and

$$B(L_{t+}) = \sum_{L' \supseteq P[N \setminus L_{t+}]} \prod_{v \in N \setminus L_{t+}} \alpha_v(L'_v \cup L_{t+})\,.$$

The first equation follows directly from the proof of Lemma 18. The proof for the second equation is analogous, and is thus not repeated here. ■

We arrive at the following algorithm:

Algorithm AllArcs
Input: partial order P on N and $\varphi_v(A_v)$ for $(v, A_v) \in N \times \mathcal{C}_v$.
Output: $\varphi^{st}(P)$ for all pairs $s, t \in N$.

1. Compute $\alpha_v(Y)$ for all $(v, Y) \in N \times \mathcal{D}$.
2. Compute $F(Y)$ and $B(Y)$ for all $Y \in \mathcal{D}$.
3. For each $t \in N$:
   (a) Compute $\gamma_t(A_t)$ for all $A_t \in \mathcal{C}_t$.
   (b) For each $s \in N \setminus \{t\}$:
   Compute $\varphi^{st}(P)$ using (11).

The complexity of the first two steps is clearly within the claimed budget—these steps are essentially the same as in our algorithm for the DAGE problem.

Step 3a is the upward-variant of the inflating zeta transform problem and can thus, by Theorem 8 and Remark 12, be solved in $O(n^2|\mathcal{C}_t| + n|\mathcal{D}|)$ time, for each t. This gives $O(n^2(C + |\mathcal{D}|))$ time in total. The space requirement is, again, clearly within the claimed budget, since the same space can be reused for different nodes t.

Step 3b takes only $O(n|\mathcal{C}_t|)$ time for each t, thus $O(nC)$ in total. The additional space requirement is negligible.

5.5 Special Cases: Regular Parent Set Collections and Bucket Orders

We have formulated our results in a very general setting where (i) the collections $\mathcal{C}_v$ of potential parent sets can be arbitrary for each node v, and (ii) the partial order P can be arbitrary. In our bounds for the time and space requirements we have paid a relatively high cost for this generality: in many cases we obtained a running time bound of $O(n^2(C+|\mathcal{D}|))$.

It appears that these bounds can be reduced significantly if we restrict (i') the parent set collections to all sets of size at most some maximum indegree k, and (ii') the partial orders to bucket orders. This restricted setting was, in fact, considered already by Koivisto and Sood (2004, Theorem 12), who showed that (using our terminology) the DAGE problem can be solved in $O(C + n2^b)$ time, when the maximum bucket size is b. We note that this bound hides a factor that is linear in k. For the term that depends on the number of downsets, the improvement is thus from about $n^2(n/b)2^b$ to $n2^b$, assuming a balanced bucket order (see Example 4). We have observed that in this restricted setting similar improved bounds can be obtained also for the DAG sampling problem and for the arc probabilities problem (we omit details).

6. Experimental Results

We have implemented the proposed partial-order-MCMC method, including the extensions based on MC^3 and AIS, for the special case of bucket orders (see Examples 4–7).[2] We will refer to these three variants of the methods simply as MCMC, MC^3, and AIS. This section reports experimental results on a selection of data sets of different characteristics. Details of the data sets and the employed Bayesian models are given in Section 6.1. Implementation details of the computational methods are given in Section 6.2.

We aim to answer four main questions: Does sampling partial orders provide us with a significant advantage over sampling linear orders? How accurate is AIS as compared to MC^3? Does the bias correction approach (i.e., scaling by the number of linear extensions) work in practice? How well can we estimate and lower bound the marginal likelihood of the model? We address these questions in Sections 6.3–6.6, respectively.

6.1 Data Sets, Model Parameters, and Features of Interest

Table 1 lists the data sets used in our experiments. The *Flare*, *German*, *Mushroom*, and *Spambase* data sets are obtained from the UCI Machine Learning Repository (Lichman, 2013). The *Alarm* data set was generated from the Alarm network (Beinlich et al., 1989). Of the *Mushroom* data set we used both the whole data set and a subsample consisting of 1000 randomly selected records of the data set. We will refer to these two versions as *Mushroom-8124* and *Mushroom-1000*. The data set with the fewest attributes, *Flare*, was used only for examining the performance of the bias correction method of Ellis and Wong (2008).

For each data set we employed the modular and the order-modular uniform Dirichlet–multinomial model described in Examples 2 and 3. In these models we set the maximum indegree parameter k to 4 for all data sets, except for *Spambase*, for which we set the value to 3 in order to keep the per-sample computations feasible.

We focus on the estimation of the arc posterior probabilities and the marginal likelihood of the model. For comparison purposes, we also computed exact values of these quantities on the *Flare*, *German*, and *Mushroom* data sets using the algorithms of Koivisto and Sood (2004; 2006) and Tian and He (2009).

6.2 Implementation Details

We made the following implementation choices in the MCMC method:

Sampling space. The sampling space was set to the balanced bucket orders of maximum bucket size b (see Example 4). Separately for each data, we set the parameter b to a value as large as possible, subject to the condition that its impact to the running time is no more than about 2 times the impact of the terms that do not depend on b. Table 1 shows the obtained values. Note that linear orders correspond to the special case of $b = 1$.

Proposal distribution. We employed swap proposals, as described in Example 7.

2. The program BEANDisco, written in C++, is publicly available at `www.cs.helsinki.fi/u/tzniinim/BEANDisco/`.

Name	n	m	k	b	CPU time	Number of iterations: Linear orders	Number of iterations: Bucket orders
Flare	13	1066	4	–	1 d	6.1×10^8	–
German	20	1000	4	7	4 d	2.4×10^8	1.0×10^8
Mushroom	22	8124	4	7	4 d	1.9×10^8	9.6×10^7
Spambase	58	4601	3	9	4 d	1.5×10^7	9.6×10^6
Alarm	37	1000	4	10	4 d	1.0×10^7	6.3×10^6

Table 1: Data sets and basic parameters used in the experiments. Abbreviations: number of attributes n, number of data records m, maximum indegree k, maximum bucket size b.

Burn-in iterations. Always 50% of the samples were treated as burn-in samples that were not included in the estimates of the quantities of interest.

Thinning. We included only every 1024th of the visited states in the final sample.

Number of DAG samples. Per sampled partial order (after thinning), we draw as many independent DAGs as was possible within 25% of the time needed for sampling the partial order (i.e., 1024 partial orders due to thinning). In order to ensure that the varying per-DAG processing time does not cause any bias, the DAGs were drawn in two phases: First, 10% of the time budget is used to get an estimate T' for the number of DAGs that can be drawn within the remaining time budget. Then, exactly T' DAGs are drawn for the use of the algorithm.

Running time. Per configuration we allowed a total running time of 4 days (excluding the additional time used to sample the DAGs), except for the *Flare* data set, for which we only ran some of configurations and at the maximum of 1 day. Table 1 shows the approximate total number of iterations made, both for linear orders and bucket orders.

Independent runs. We ran each configuration 7 times, starting from states drawn independently and uniformly at random.

In addition to the above choices, we made the following additional choices in the MC^3 and AIS methods:

Tempering scheme. We used the linear stepping scheme. For MC^3 we varied the number of temperature levels K in $\{3, 15, 63\}$, and the thinning factor was reduced to $1024/(K+1)$ correspondingly. For AIS we set the number of levels proportionally to the data size, $K = K'mn$ where the factor K' varied in $\{1/4, 1, 4\}$. These values were found by preliminary experiments (results not shown).

Number of chain swap proposals. For MC^3 swaps of adjacent chains were proposed $1000 \times K$ times every time before moving all the chains one step. Here the rationale is that chain

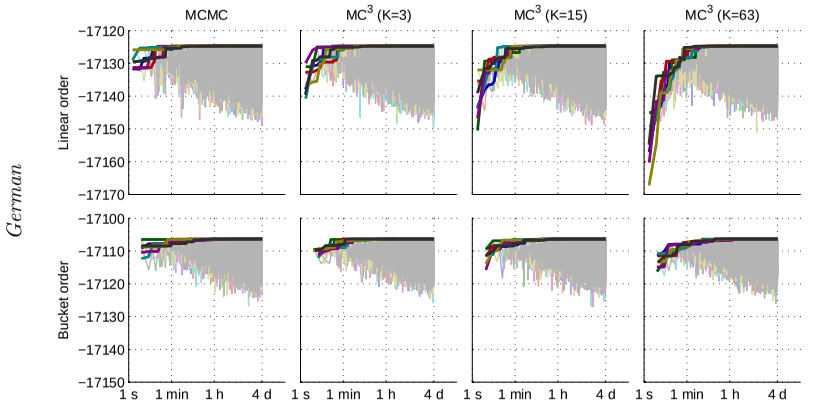


Figure 7: Mixing and convergence of MCMC and MC^3 on the *German* data set, for linear orders (top) and bucket orders (bottom). Each panel shows the traces of 7 independent runs (thin pale lines), that is, the natural logarithm of the unnormalized posterior probability of the visited state (y-axis) as a function of the time elapsed (x-axis). The cumulative maximum of these values are also shown for all the 7 runs (thick dark lines). If mixing is good, then all 7 runs should quickly converge to approximately same posterior probability levels. This seems to be the case in all eight panels. Note that posterior probabilities of linear orders and bucket orders are not directly comparable.

swaps are computationally cheap and improve mixing considerably especially when K is large.

Number of iterations along the coolest chain. For AIS we ran $K/4$ iterations along the coolest chain. Here the rationale is that collecting a large number of samples from the coolest chain is relatively cheap in comparison to the long annealing schedule. Note that thinning concerns only these $K/4$ iterations.

We will mainly examine the methods' performance as functions of running time. For visualization purposes we did a second round of thinning by an additional factor of 10. Note however that this additional thinning does not affect the estimates, which are based on the full set of samples obtained after the first round of thinning.

6.3 Advantage of Partial Orders

We first compared the effect of the sampling space—whether linear orders or bucket orders—to the mixing rate and convergence speed of the Markov chains. We did this for the basic MCMC as well as for MC^3 with varying number of temperature levels K (Figures 7–9). We found that on the *German* and *Alarm* data sets the two sampling spaces perform about

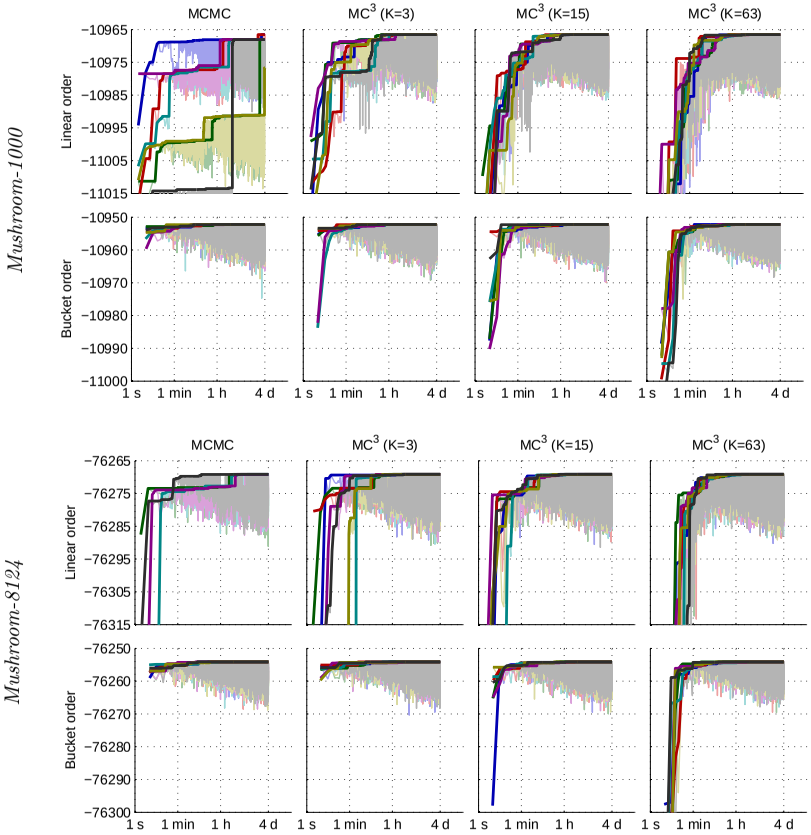


Figure 8: Mixing and convergence of MCMC and MC^3 on the *Mushroom* data sets. See the caption of Figure 7 for further descriptions. The top left panels of both data sets are examples of bad mixing. Note that MCMC with linear orders fails completely on the *Mushroom-8124* data set for 3 out of the 7 runs, and consequently the traces do not achieve the visible range of the y-axis.

equally well. On both these data sets, the chains seem to converge within a couple of minutes. We also observe that that tempering is not particularly beneficial. The results confirm that linear orders can perform very well on some data sets. The harder data sets, *Mushroom* and *Spambase*, on the other hand, separate the two sampling spaces: the performance of bucket orders is superior to linear orders. While the difference is particularly

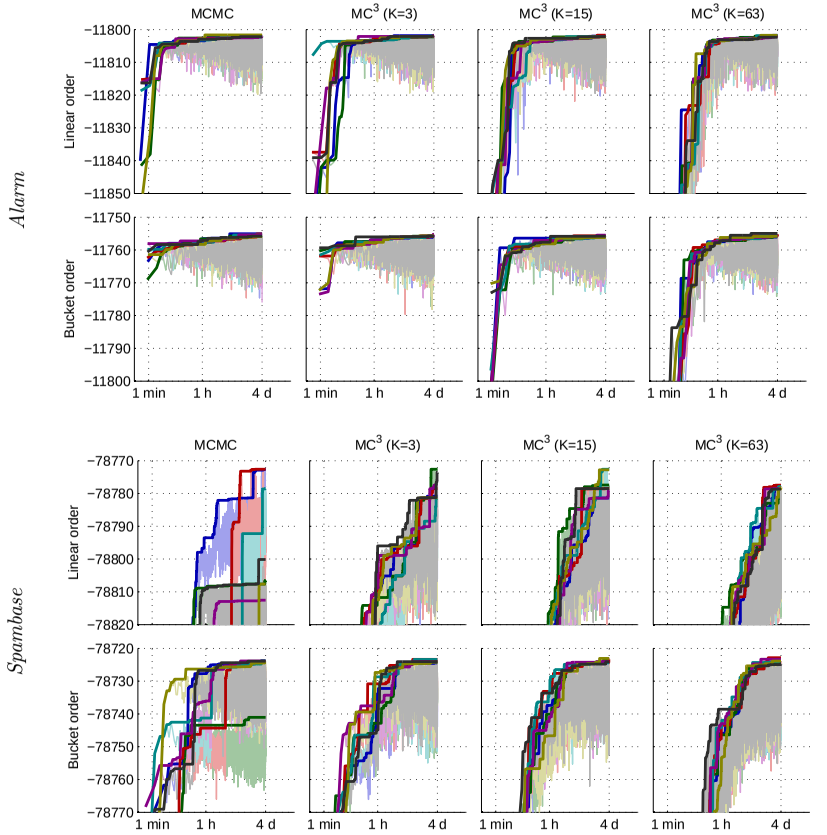


Figure 9: Mixing and convergence of MCMC and MC^3 on the *Alarm* and *Spambase* data sets. See the caption of Figure 7 for further descriptions.

large for the basic MCMC method, the difference remains significant for the MC^3 variants: On *Mushroom*, all MC^3 runs seem to converge, but the convergence is quicker when using bucket orders. On *Spambase*, it is not clear if any of the linear-order-based runs managed to converge within the given time budged, while all the bucket-order-based MC^3 runs appear to converge.

Next we investigated how the differences in mixing and convergence rates translate to differences in the accuracy of the arc posterior probability estimates under the order-modular model (i.e., without bias correction). For each pair of nodes, we measure the

accuracy by the standard deviation of the estimates in the 7 independent runs. When the exact values were available we also gauged the accuracy by the median of the 7 absolute errors. In a worst-case spirit, we report the respective the *largest standard deviation* and the *largest median error*, which we obtain by taking the maximum over the node pairs. We found that, qualitatively, the results follow closely the mixing and convergence behavior of the methods (Figure 10). Specifically, on the *German* and *Alarm* data sets all the methods perform about equally well, whereas on the *Mushroom* data sets the estimates we obtain with bucket orders are significantly more accurate than the estimates we obtain with linear orders, the difference being about one order of magnitude. On the *Spambase* data set none of the methods performs particularly well, which can be probably explained by the insufficiency of the allocated running time for achieving proper convergence. As the exact values are not available for *Spambase*, the small empirical standard deviations might be just due to similar yet insufficient convergence of the 7 runs. On the other hand, we also observe that, in general, the largest standard deviation reflects very well the largest median error.

Based on these results with different number of temperature levels K for MC^3, we fixed $K = 15$ for presenting the remaining results in the next subsections. This value of K appears to make a good compromise between a large number of iterations and fast mixing.

6.4 Accuracy of AIS

To study to performance of AIS, we compared the obtained arc posterior probability estimates to those obtained with MC^3, now with $K = 15$ only (Figure 11, left; Figure 12, left). We found that on the easiest data set, *German*, the methods perform almost identically, regardless of the value of the K' parameter. This holds also on the *Mushroom* data sets, provided that K' is large enough (1 or 4). Furthermore, we observe that bucket orders are superior to linear orders also in the case of AIS. On the *Alarm* data set AIS is slightly behind MC^3 even when sampling bucket orders, and on the *Spambase* data set the difference of the methods is larger.

Based on these results with different values of K', we fixed $K' = 1$ for presenting the remaining results in the next subsections. This value of K' appears to make a good compromise between a long annealing schedule and a relatively large number of independent samples.

6.5 Efficiency of Bias Correction

So far we have only discussed the results obtained under the order-modular model. We next turn to the results under the modular model, which we obtain by applying the bias-corrected estimators. The results are presented graphically in Figures 11–13.

First we studied the subproblem of counting the linear extensions of a given DAG. We generated random DAGs for a varying number of nodes n, setting the maximum indegree to either 3 or 6, and then ran the exact dynamic programming algorithm (from Section 2.2). We found that while both the time and the space complexity of the algorithm grow exponentially in n, the computations are feasible as long as n is at most about 40 for sparse DAGs, or at most about 60 for dense DAGs (Figure 13a). For example, the linear extensions of a 49-node DAG of maximum indegree 6 can typically be counted within a couple of seconds.

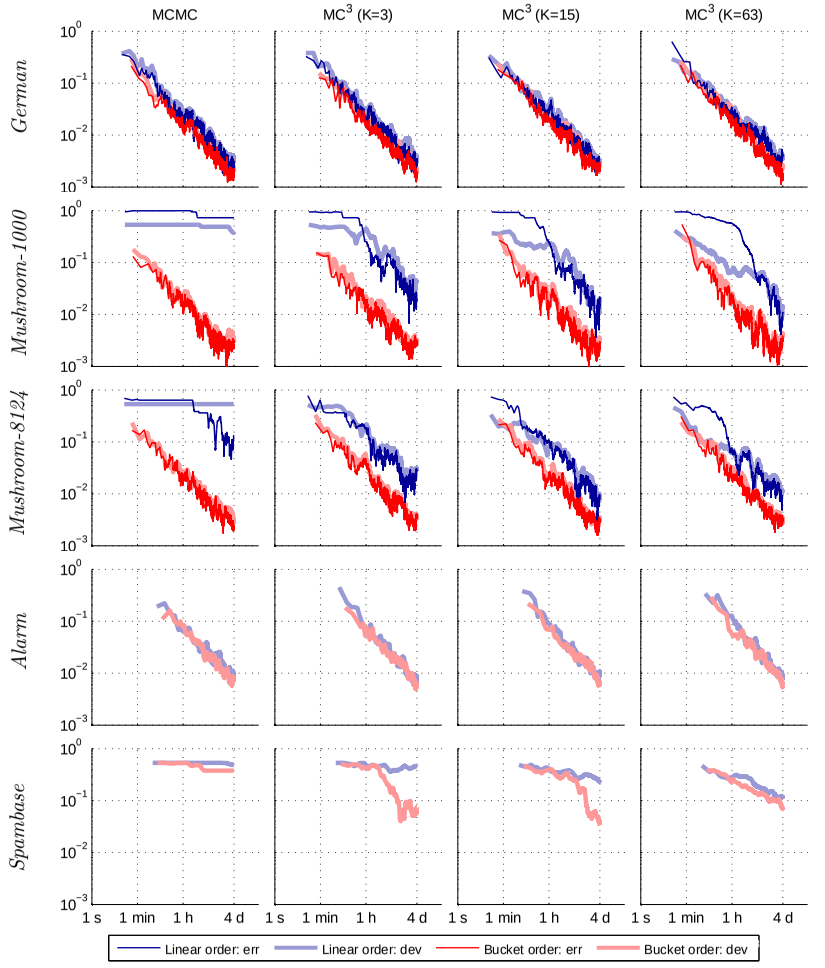


Figure 10: The accuracy of the arc posterior probability estimates under an order-modular model, for MCMC and MC^3. For each method the largest median error (err) and the largest standard deviation (dev) are shown as a function of the time elapsed.

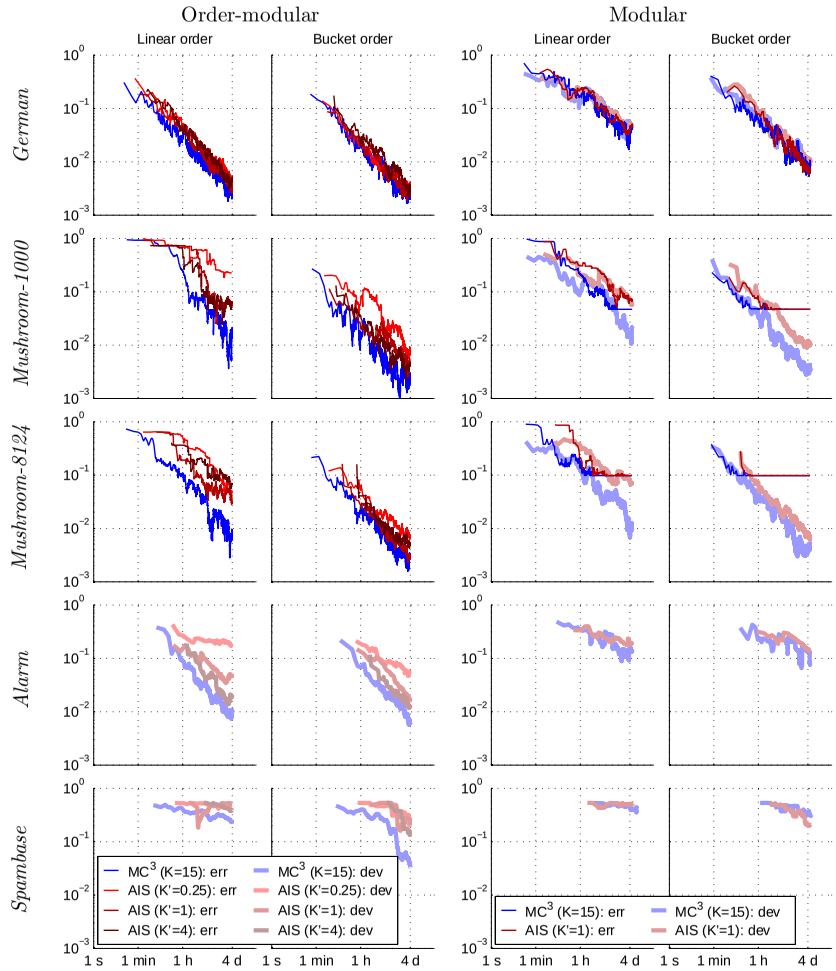


Figure 11: The accuracy of the arc posterior probability estimates under an order-modular model (left) and a modular model (right), for AIS and MC^3. For clarity, only the largest median errors are shown for the *German* and *Mushroom* data sets.

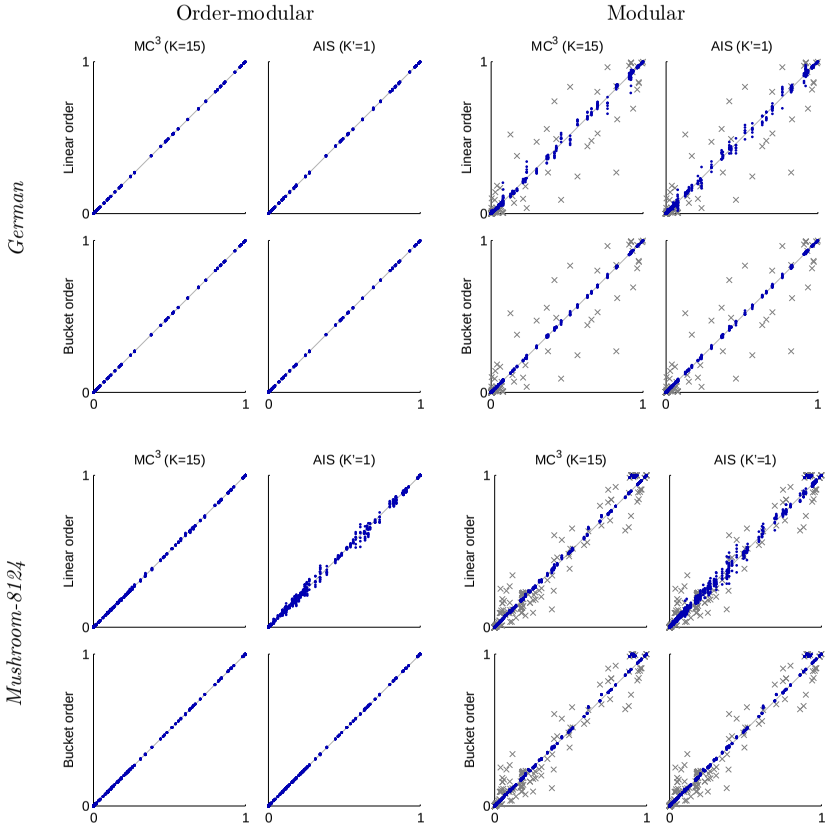
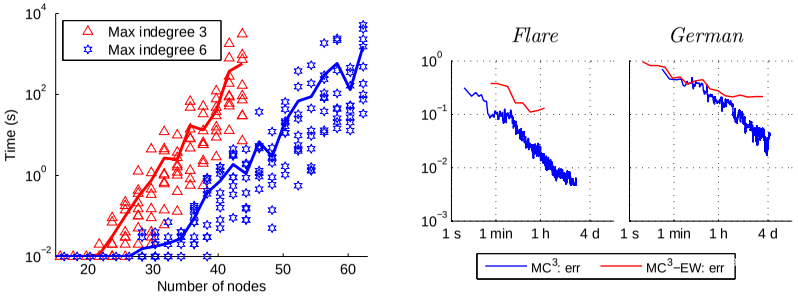


Figure 12: The arc posterior probability estimates for 7 independent runs (y-axis) plotted against the exact values (x-axis), under an order-modular and modular model. For the modular model also the biased values (exact under the order-modular model) are plotted against the unbiased exact values (gray $\times$).

However, the simple dynamic programming algorithm does not exploit sparsity well and so the performance degrades for maximum indegree 3.

Then we compared our bias correction method to that of Ellis and Wong (2008), which we refer to as the *EW method* in the sequel. The EW method works as follows. First it draws a sample of node orderings from an approximate posterior distribution, like the order-MCMC method of Friedman and Koller (2003). Then, from each sampled node ordering, it



(a) The speed of counting linear extensions.

(b) The accuracy of bias correction methods.

Figure 13: (a) The runtime required to count the linear extensions of 9 random DAGs with a varying number of nodes and a varying maximum indegree. Runtimes less than 0.01 seconds are rounded up to 0.01. The mean of the runtimes is shown by a solid line. The available memory was limited to 16 GB. No runtime estimates are shown when the memory requirement exceeded the limit for at least one of the 9 DAGs. (b) The accuracy of the proposed bias correction method (MC^3) compared to the EW method with $\epsilon = 0.05$ (MC^3-EW). Both methods are based on sampling linear orders by MC^3 under an order-modular model. The largest median error of the arc posterior probability estimates over 7 independent runs is shown as a function of the time elapsed. The runs of the EW method terminated as soon as one of the 7 runs ran out the 20 GB of memory allocated for storing the DAGs. For the EW method the estimates were computed at doubling sample sizes $2, 4, 8, \ldots$, to make sliding burn-in periods computationally feasible.

generates a number of independent DAGs from the conditional posterior distribution until the total posterior mass of the unique DAGs obtained is at least $1 - \epsilon$, where $\epsilon > 0$ is a parameter of the method. Finally, the DAGs so obtained for each node ordering are merged into a single set, duplicates are removed, and each DAG is assigned an importance weight proportional to the posterior probability of the DAG.

The results of the comparison on the *Flare* and *German* data sets are shown in Figure 13b. On these data sets the EW method turned out to be inferior to the proposed method. On the other data sets (*Mushroom-1000*, *Mushroom-8124*, *Alarm*, *Spambase*), the EW method either ran out of memory immediately or was able to process only a couple of node ordering samples, yielding poor estimates (the largest media error close to 1; results not shown).

Finally we compared the arc posterior probability estimates under the two models, order-modular and modular (Figure 11, right). The results confirm the expectation that the estimates are less accurate under the modular model. The weaker performance is due to the additional importance sampling step needed for bias correction, which reduces the effective sample size. Otherwise, the earlier conclusions hold also under the modular model:

bucket orders outperform linear orders and MC^3 is slightly superior to AIS. However, the bias correction method also brings a new feature: on the *Mushroom* data sets the largest median error stops decreasing at some point, stagnating at an error of about 0.10 (while the largest standard deviation continues decreasing). This phenomenon is due to a few node pairs for which the arc posterior probability is very close to 1 under the order-modular model, but around 0.90 under the modular model. The culprits can be located in Figure 12 (right) as a cluster of points in the upper-right corner of the panels. Note that for the other node pairs the arc posterior probability estimates are very accurate.

6.6 Estimating the Marginal Likelihood

For estimating the marginal likelihood AIS clearly outperforms MC^3, as expected (Figure 14). The relatively small number of chains (i.e., temperature levels) in MC^3 leads to a large fluctuation in the estimates within a single run and between independent runs. AIS, on the other hand, benefits from the long annealing scheme and produces very accurate estimates on the *German* and *Mushroom* data sets. On the *Alarm* data set the variance of the estimates is larger for both methods. Yet the estimates of AIS seem to converge well. The *Spambase* data set is, again, the hardest instance for both method, the results suggesting insufficient convergence of the samplers.

We also observed that when the estimates of AIS are close to the exact value, then also the high-confidence lower bounds are very good. Indeed, on the *German* and *Mushroom* data sets the lower bounds obtained with the square-root lower-bounding scheme are within an absolute error of about 0.4 or less in the logarithmic scale, hence within a relative error of about $e^{0.4} - 1 \approx 0.4$ or less. Whether the model is order-modular or modular affects the accuracy of the marginal likelihood estimates considerably on the *German* data set but very little on the *Mushroom* data sets.

7. Conclusions and Future Work

We have investigated a sampling-based approach to Bayesian structure learning in Bayesian networks. Our work has been inspired to a large extent by the order-MCMC method of Friedman and Koller (2003), in which the sampling space consists of all possible linear orders of the nodes. Our partial-order-MCMC methods advance the methodology in four dimensions:

1. *Smoother sampling space.* The space of partial orders is smaller still than the space of linear orders. Also, the posterior distribution tends to be smoother, because the posterior probability of each partial order is obtained by summing the posterior probabilities of its linear extensions. Our empirical results agree with these expectations, and show instances where partial-order-MCMC performs significantly better than order-MCMC.

2. *Arbitrary structure priors.* We proposed a method to correct the bias that arises because the samples are drawn (approximately) from an order-modular rather than a modular posterior distribution. The correction amounts to a simple scaling term per sampled DAG. The term only depends on the number of linear extensions of the DAG, and as we showed, it can be computed sufficiently fast when the number of

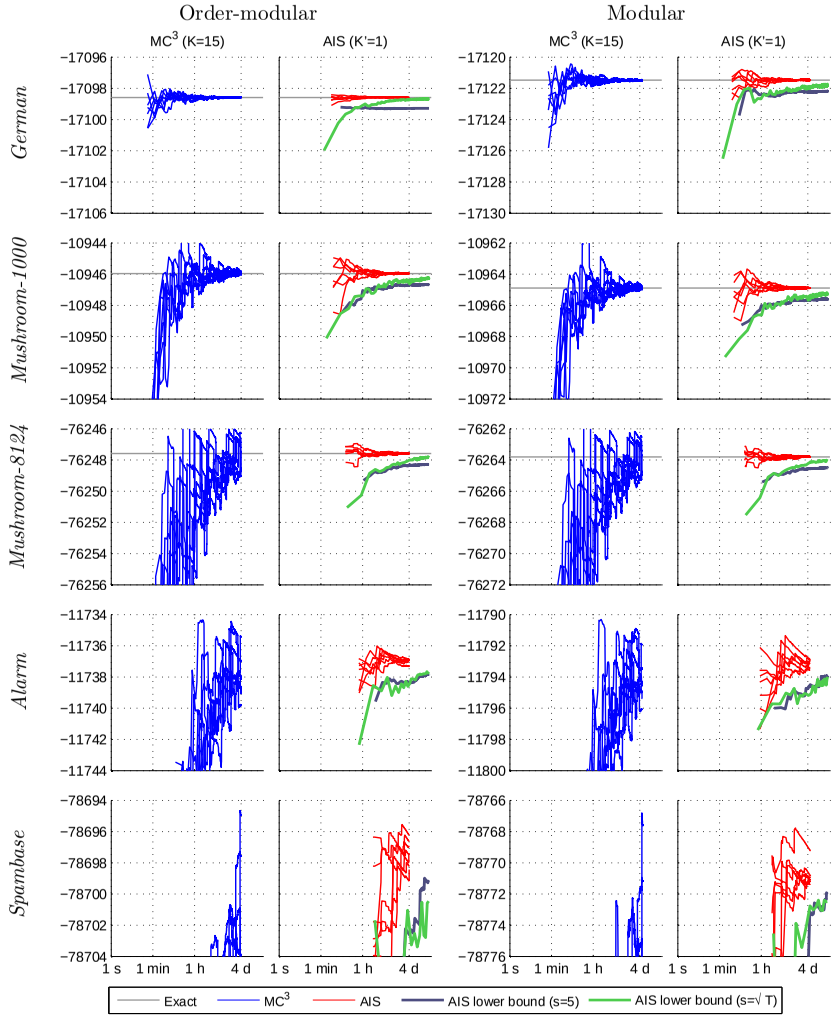


Figure 14: The marginal likelihood of the model estimated by 7 independent runs of AIS and MC^3. For AIS, shown are also lower bound estimates obtained with all the samples pooled together (thus extending the total running). The lower bound schemes are as described in Example 8 and 9. On the y-axis is the natural logarithm of the estimate or exact value.

nodes is moderate, say, at most 40. Our empirical results confirmed that, in general, the correction works well also in practice—some rare cases where the correction fails are discussed below.

For convenience, we focused on the special case where the correct model is the modular counterpart of an order-modular model. In principle, it is straightforward to extend the estimators to accommodate an arbitrary model, as long as the corresponding posterior probability function (a) can be efficiently evaluated for any given DAG (up to a normalizing constant) and (b) has a support that is contained in the support of the order-modular sampling distribution. The statistical efficiency of the estimator may, however, deteriorate if the true distribution is far from the sampling distribution.

3. *Efficient parallel computation.* We observed that the annealed importance sampling method (AIS) is easy to run in parallel, since the method is designed to produce independent samples. We compared AIS empirically to another tempering method, Metropolis-coupled MCMC (MC^3), and found that AIS-based estimates are slightly less accurate for arc posterior probabilities but significantly more accurate for the marginal likelihood of the model.

4. *Quality guarantees.* We also observed that AIS allows us to compute high-confidence lower bounds for the marginal likelihood. We showed that the lower bounds are very good, within a factor of about 1.2 on the data sets for which exact values were available. Admitted, lower bounds on the marginal likelihood is just a small step toward accuracy guarantees more generally.

These advancements (1–4) upon the order-MCMC method come essentially "for free" concerning both computational and statistical efficiency. Indeed, sampling partial orders is not more expensive than sampling linear orders, provided that the partial orders are sufficiently thin (i.e., the number of downsets is small). Namely, the bottleneck in both methods is the need to visit a large number of potential parent sets of the nodes. Likewise, while counting the linear extensions of a sampled DAG can be computationally demanding, the computational cost is compensated by the relatively small number of sampled DAGs (after thinning) as compared to the total number of partial order samples (before thinning).

The proposed methods also have shortcomings that call for further investigations. First, the complexity of the per-sample computations grows rapidly with the number of nodes n. In particular, the complexity of computing the unnormalized posterior probability of a partial order does not scale well with the indegree k, and the computations become impractical when, say, $n \geq 60$ and $k \geq 4$. For example, on the *Spambase* data set ($n = 58$, $k = 3$) we observed that the large number of potential parent sets rendered the allocated running time of 4 days insufficient for proper convergence of the sampling methods. To significantly expedite these computations, a plausible idea is to resort to approximations instead of exact computation. In the sampling space of *linear* node orders, the approach has proved successful, provided that the number of data records is large (Friedman and Koller, 2003; Niinimäki and Koivisto, 2013a). In contrast, the problem of counting the linear extensions of a sampled DAG seems to get harder for sparser DAGs. However, we believe this is rather a feature of our simple algorithm—our preliminary results with a more sophisticated algorithm suggest that sparsity can actually be turned into a computational advantage.

Second, the theoretical accuracy guarantees of the proposed methods are quite modest. For example, currently the methods do not produce good lower or upper bounds for the arc posterior probabilities. Thus the user has to resort to monitoring empirical variance and related statistics that do not always generalize well beyond the obtained sample. We saw a warning example of that on the *Mushroom* data set, where the bias-corrected estimates had small variance over 7 independent runs, but where the estimates were biased and the bias did not decrease as the number of samples grew. While this failure concerned only a few node pairs (the estimates being very accurate for the rest), it shows that sometimes the proposed bias correction method is not efficient. It may happen that all the sampled DAGs contain the arc of interest, because the biased, order-modular posterior probability of the arc is close to 1, and yet the unbiased, modular posterior probability of the arc can be much below 1, say, 0.90. In such cases the present bias correction approach may fail. It is an intriguing open question how situations of this kind can be treated or circumvented.

Acknowledgments

The authors would like to thank Kustaa Kangas for valuable discussions about the problem of counting linear extensions, and Tianqi Li for her contributions to preliminary experiments on the bias correction heuristic of Ellis and Wong (2008). This research was funded in part by the Academy of Finland, Grants 276864 "Supple Exponential Algorithms" and 251170 "Finnish Centre of Excellence in Computational Inference Research COIN".

Appendix A. Proofs

Lemma 7 *It holds that $\varphi_n(Y) = \widehat{\varphi}(Y)$ for all $Y \in \mathcal{D}$.*

Proof For a set X write X_i for $X \cap \{v_i\}$, and for sets X, Y write $X \subseteq_i Y$ as a shorthand for $X \subseteq Y$ and $X_j = Y_j$ for $j > i$. Let $Y \in \mathcal{D}$. We prove by induction on i that

$$\varphi_i(Y) = \sum_{\substack{X \in \mathcal{D} \\ X \subseteq_i Y}} \varphi(X)\,.$$

The claim then follows, for the condition $X \subseteq_n Y$ is equivalent to $X \subseteq Y$. Note also that the base case of $i = 0$ clearly holds, since $X \subseteq_0 Y$ is equivalent to $X = Y$.

Let then $i > 0$. Suppose first that $v_i \notin Y$ or $Y \setminus \{v_i\} \notin \mathcal{D}$. Then $\varphi_i(Y) = \varphi_{i-1}(Y)$. If $v_i \notin Y$, then the conditions $X \subseteq_i Y$ and $X \subseteq_{i-1} Y$ are equivalent, and by the induction hypothesis the claim follows. Assume then that $v_i \in Y$ and $Y \setminus \{v_i\} \notin \mathcal{D}$. By the induction hypothesis it suffices to show that, if $X \in \mathcal{D}$, then the conditions $X \subseteq_i Y$ and $X \subseteq_{i-1} Y$ are equivalent. Because the latter condition implies the former, it remains to consider the other direction. To this end, suppose the contrary, $X \subseteq_i Y$ but $X_i \neq Y_i$. Thus we must have $v_i \notin X$. However, the assumption $Y \setminus \{v_i\} \notin \mathcal{D}$ and the fact that $Y \in \mathcal{D}$ imply that there is an element $v_j \in Y \setminus \{v_i\}$ such that $v_i v_j \in P$, which, together with the ordering of the elements imply that $i < j$. Therefore, as $X \subseteq_i Y$, we have $v_j \in X$, which contradicts our assumption that $X \in \mathcal{D}$.

Suppose then that $v_i \in Y$ and $Y \setminus \{v_i\} \in \mathcal{D}$. Then $\varphi_i(Y) = \varphi_{i-1}(Y) + \varphi_{i-1}(Y \setminus \{v_i\})$. Let $X \in \mathcal{D}$. Denote the conditions of interest by

$$\begin{aligned} E &= X \subseteq_i Y\,, \\ E_1 &= X \subseteq_{i-1} Y\,, \\ E_2 &= X \subseteq_{i-1} Y \setminus \{v_i\}\,. \end{aligned}$$

By the recurrence and the induction hypothesis, it suffices to show that E holds if and only if exactly one the conditions E_1 and E_2 holds. Clearly, if either E_1 or E_2 holds, so does E. Suppose therefore that E holds. Now, either $X_i = \{v_i\} = Y_i$, in which case E_1 holds (but E_2 does not); or $X_i = \emptyset$, in which case E_2 holds (but E_1 does not). ■

Lemma 9 *Let Y and Y' be distinct downsets. Then the tails $\mathcal{T}_Y$ and $\mathcal{T}_{Y'}$ are disjoint.*

Proof Suppose the contrary that there exists a $X \in \mathcal{T}_Y \cap \mathcal{T}_{Y'}$. Since Y and Y' are distinct, by symmetry we may assume $Y \setminus Y'$ contains an element w. Thus $w \notin X$, because $X \subseteq Y'$. Since $\max Y \subseteq X$, we have $w \notin \max Y$. On the other hand, from the definition of $\max Y$ together with transitivity and acyclicity of P it follows that, for every $u \in Y \setminus \max Y$ there exists a $v \in \max Y$ such that $uv \in P$. In particular, there exists a $v \in \max Y$ such that $wv \in P$. Since $w \notin Y'$ and Y' is in $\mathcal{D}$, it follows from the definition of a downset that $v \notin Y'$. But this is a contradiction, because we had $v \in \max Y$ and $\max Y \subseteq X \subseteq Y'$, which imply $v \in Y'$. ■

Lemma 10 *Let $X \subseteq N$. Let $Y = \{u : uv \in P, v \in X\}$, that is, the downward-closure of X. Then $Y \in \mathcal{D}$ and $X \in \mathcal{T}_Y$.*

Proof To see that Y is a downset, suppose the contrary that there exist $v \in Y$ and $uv \in P$ such that $u \notin Y$. By the definition of Y, there must be $vw \in P$ such that $w \in X$. But since P is transitive, $uw \in P$ and thus $u \in Y$ which is a contradiction.

Consider then the second claim, that $\max Y \subseteq X \subseteq Y$. The second inclusion follows from the reflexivity of P. For the first inclusion, note that that $\max Y = \max X$, since Y only contains elements u such that $uv \in P$ for some $v \in X$.

■

Lemma 11 *Let $Z \subseteq N$. Then $\mathcal{D} \cap 2^Z = \mathcal{D} \cap 2^Y$, where the set $Y \in \mathcal{D}$ is given by*

$$Y = \{v \in Z : \textit{if } uv \in P, \textit{ then } u \in Z\}\,;$$

in words, Y consists of all elements of Z whose predecessors also are in Z.

Proof We show first that $Y \in \mathcal{D}$. Suppose the contrary that $v \in Y$ and $uv \in P$ but $u \notin Y$. By the definition of Y we have $u \in Z$. Since $u \notin Y$, there is a $wu \in P$ such that $w \notin Z$. However, as P is transitive, we have $wv \in P$ and thus $w \in Z$ which is a contradiction.

To complete the proof, we show that $\mathcal{D} \cap 2^Z = \mathcal{D} \cap 2^Y$. Clearly $\mathcal{D} \cap 2^Z \supseteq \mathcal{D} \cap 2^Y$. For the other direction, we consider an arbitrary $X \in \mathcal{D} \cap 2^Z$ and show that $X \subseteq Y$. To this end, consider any $v \in X$. Now, if $uv \in P$, then $u \in X$ (because X is a downset)

and consequently $u \in Z$ (because $X \subseteq Z$). Thus, by the definition of Y we get that $v \in Y$. ■

References

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. Introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.

Mark Bartlett and James Cussens. Advances in Bayesian network learning using integer programming. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 182–191, 2013.

Alexis J. Battle, Martin Jonikas, Peter Walter, Jonathan Weissman, and Daphne Koller. Automated identification of pathways from quantitative genetic interaction data. *Molecular Systems Biology*, 6:379–391, 2010.

Ingo Beinlich, Henri J. Suermondt, R. Martin Chavez, and Gregory F. Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine (AIME)*, pages 247–256, 1989.

Graham Brightwell and Peter Winkler. Counting linear extensions. *Order*, 8:225–242, 1991.

Russ Bubley and Martin E. Dyer. Faster random generation of linear extensions. *Discrete Mathematics*, 201:81–88, 1999.

Wray Buntine. Theory refinement on Bayesian networks. In *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 52–60, 1991.

Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

Jukka Corander, Magnus Ekdahl, and Timo Koski. Parallel interacting MCMC for learning of topologies of graphical models. *Data Mining and Knowledge Discovery*, 17:431–456, 2008.

Daniel Eaton and Kevin Murphy. Bayesian structure learning using dynamic programming and MCMC. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 101–108, 2007.

Byron Ellis and Wing Hung Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103:778–789, 2008.

Nir Friedman and Daphne Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–125, 2003.

Charles J. Geyer. Markov chain Monte Carlo maximum likelihood. In *Proceedings of the 23rd Symposium on the Interface*, pages 156–163, 1991.

Charles J. Geyer and Elizabeth A. Thompson. Annealing Markov chain Monte Carlo with application to ancestral inference. *Journal of the American Statistical Association*, 90: 909–920, 1995.

Vibhav Gogate and Rina Dechter. Sampling-based lower bounds for counting queries. *Intelligenza Artificiale*, 5:171–188, 2011.

Vibhav Gogate, Bozhena Bidyuk, and Rina Dechter. Studies in lower bounding probabilities of evidence using the Markov inequality. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 141–148, 2007.

Carla P. Gomes, Joerg Hoffmann, Ashish Sabharwal, and Bart Selman. From sampling to model counting. In *Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI)*, pages 2293–2299, 2007.

Marco Grzegorczyk and Dirk Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71: 265–305, 2008.

Michel Habib, Raoul Medina, Lhouari Nourine, and George Steiner. Efficient algorithms on distributive lattices. *Discrete Applied Mathematics*, 110:169–187, 2001.

David Heckerman, Dan Geiger, and David Maxwell Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

Mikko Koivisto. Advances in exact Bayesian structure discovery in Bayesian networks. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 241–248, 2006.

Mikko Koivisto and Pekka Parviainen. A space–time tradeoff for permutation problems. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 484–492, 2010.

Mikko Koivisto and Kismat Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

Moshe Lichman. UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences, 2013. URL `http://archive.ics.uci.edu/ml`.

David Madigan and Jeremy York. Bayesian graphical models for discrete data. *International Statistical Review*, 63:215–232, 1995.

Radford Neal. Annealed importance sampling. *Statistics and Computing*, 11:125–139, 2001.

Teppo Niinimäki and Mikko Koivisto. Treedy: A heuristic for counting and sampling subsets. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 469–477, 2013a.

Teppo Niinimäki and Mikko Koivisto. Annealed importance sampling for structure learning in Bayesian networks. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1579–1585, 2013b.

Teppo Niinimäki, Pekka Parviainen, and Mikko Koivisto. Partial order MCMC for structure discovery in Bayesian networks. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 557–565, 2011.

Akimitsu Ono and Shin-Ichi Nakano. Constant time generation of linear extensions. In *Fundamentals of Computation Theory*, pages 445–453, 2005.

Sascha Ott, Seiya Imoto, and Satoru Miyano. Finding optimal models for small gene networks. In *Pacific Symposium on Biocomputing*, pages 557–567, 2004.

Pekka Parviainen and Mikko Koivisto. Bayesian structure discovery in Bayesian networks with less space. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistic (AISTATS)*, pages 589–596, 2010.

Pekka Parviainen and Mikko Koivisto. Finding optimal bayesian networks using precedence constraints. *Journal of Machine Learning Research*, 14:1387–1415, 2013.

Gara Pruesse and Frank Ruskey. Generating linear extensions fast. *SIAM Journal on Computing*, 23:373–386, 1994.

Tomi Silander and Petri Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 445–452, 2006.

Ajit Singh and Andrew Moore. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University, 2005.

Jin Tian and Ru He. Computing posterior probabilities of structural features in Bayesian networks. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 538–547, 2009.

Michael Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on Software Engineering*, 17:972–975, 1991.

Alastair Walker. An efficient method for generating discrete random variables with general distributions. *ACM Transactions on Mathematical Software*, 3:253–256, 1977.

Changhe Yuan and Brandon Malone. Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research*, 48:23–65, 2013.

Journal of Machine Learning Research 17 (2016) 1-47 Submitted 5/15; Revised 12/15; Published 2/16

Estimation from Pairwise Comparisons: Sharp Minimax Bounds with Topology Dependence

Nihar B. Shah[†] NIHAR@EECS.BERKELEY.EDU
Sivaraman Balakrishnan[#] SIVA@STAT.CMU.EDU
Joseph Bradley[†] JOSEPH.KURATA.BRADLEY@GMAIL.COM
Abhay Parekh[†] PAREKH@BERKELEY.EDU
Kannan Ramchandran[†] KANNANR@EECS.BERKELEY.EDU
Martin J. Wainwright[† ⋆] WAINWRIG@BERKELEY.EDU

[†] *Department of Electrical Engineering and Computer Sciences*
[⋆] *Department of Statistics,*
University of California, Berkeley
Berkeley, CA-94720, USA

[#] *Department of Statistics,*
Carnegie Mellon University
Pittsburgh, PA-15213, USA

Editor: Moritz Hardt

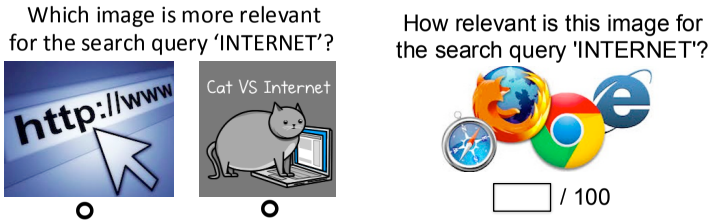
Keywords: Pairwise comparisons, graph, topology, ranking, crowdsourcing

Abstract

Data in the form of pairwise comparisons arises in many domains, including preference elicitation, sporting competitions, and peer grading among others. We consider parametric ordinal models for such pairwise comparison data involving a latent vector $w^* \in \mathbb{R}^d$ that represents the "qualities" of the d items being compared; this class of models includes the two most widely used parametric models—the Bradley-Terry-Luce (BTL) and the Thurstone models. Working within a standard minimax framework, we provide tight upper and lower bounds on the optimal error in estimating the quality score vector w^* under this class of models. The bounds depend on the topology of the comparison graph induced by the subset of pairs being compared, via the spectrum of the Laplacian of the comparison graph. Thus, in settings where the subset of pairs may be chosen, our results provide principled guidelines for making this choice. Finally, we compare these error rates to those under cardinal measurement models and show that the error rates in the ordinal and cardinal settings have identical scalings apart from constant pre-factors.

1. Introduction

In an increasing range of applications, it is of interest to elicit judgments from non-expert humans. For instance, in marketing, elicitation of preferences of consumers about products, either directly or indirectly, is a common practice (Green et al., 1981). The gathering of this and related data types has been greatly facilitated by the emergence of "crowdsourcing"



(a) Asking for a pairwise comparison. (b) Asking for a numeric score.

Figure 1: An example of eliciting judgments from people: rating the relevance of the result of a search query.

platforms such as Amazon Mechanical Turk: they have become powerful, low-cost tools for collecting human judgments (Khatib et al., 2011; Lang and Rio-Ross, 2011; von Ahn et al., 2008). Crowdsourcing is employed not only for collection of consumer preferences, but also for other types of data, including counting the number of malaria parasites in an image of a blood smear (Luengo-Oroz et al., 2012); rating responses of an online search engine to search queries (Kazai, 2011); or for labeling data for training machine learning algorithms (Hinton et al., 2012; Raykar et al., 2010; Deng et al., 2009). In a different domain, competitive sports can be understood as a mechanism for sequentially performing comparisons between individuals or teams (Ross, 2007; Herbrich et al., 2007). Finally, peer-grading in massive open online courses (MOOCs) (Piech et al., 2013) can be viewed as another form of elicitation.

A common method of elicitation is through pairwise comparisons. For instance, the decision of a consumer to choose one product over another constitutes a pairwise comparison between the two products. Workers in a crowdsourcing setup are often asked to compare pairs of items: for instance, they might be asked to identify the better of two possible results of a search engine, as shown in Figure 1a. Competitive sports such as chess or basketball also involve sequences of pairwise comparisons. From a modeling point of view, we can think of pairwise comparisons as a means of estimating the underlying "qualities" or "weights" of the items being compared (e.g., skill levels of chess players, relevance of search engine results, etc.). Each pairwise comparison can be viewed as a noisy sample of some function of the underlying pair of (real-valued) weights. Noise can arise from a variety of sources. When objective questions are posed to human subjects, noise can arise from their differing levels of expertise. In a sports competition, many sources of randomness can influence the outcome of any particular match between a pair of competitors. Thus, one important goal is to estimate the latent qualities based on noisy data in the form of pairwise comparisons. A related problem is that of experimental design: assuming that we can choose the subset of pairs to be compared (e.g., in designing a chess tournament), what choice leads to the most accurate estimation? Characterizing the fundamental difficulty of estimating the weights allow us to make this choice judiciously. These tasks are the primary focus of this paper.

In more detail, the focus of this paper is the aggregation from pairwise comparisons in a fairly broad class of parametric models. This class includes as special cases the two most

popular models for pairwise comparisons—namely, the Thurstone (Case V) (Thurstone, 1927) and the Bradley-Terry-Luce (BTL) (Bradley and Terry, 1952; Luce, 1959) models. The Thurstone (Case V) model has been used in a variety of both applied (Swets, 1973; Ross, 2007; Herbrich et al., 2007) and theoretical papers (Bramley, 2005; Krabbe, 2008; Nosofsky, 1985). Similarly, the BTL model has been popular in both theory and practice (Nosofsky, 1985; Atkinson et al., 1998; Koehler and Ridpath, 1982; Heldsinger and Humphry, 2010; Loewen et al., 2012; Green et al., 1981; Khairullah and Zionts, 1987).

1.1 Related work

There is a vast literature on the Thurstone and BTL models, and we focus on those most closely related to our own work. Negahban et al. (2012) provide minimax bounds for the BTL model in the special case where the comparisons are evenly distributed. They focus on this case in order to complement their analysis of an algorithm based on a random walk. In their analysis, there is a gap between the achievable rate of the MLE and the lower bound. In contrast, our analysis eliminates this discrepancy and shows that MLE is an optimal estimator (up to constant factors) and achieves the minimax rate. In a work concurrent with our initial submission to arXiv (Shah et al., 2014), Hajek et al. (2014) consider the problem of estimation in the Plackett-Luce model, which extends the BTL model to comparisons of two or more items. They derive bounds on the minimax error rates under this model which, under certain conditions on the comparison graphs, are tight up to logarithmic factors. In general, their topology-dependent bounds rely on the degrees of the vertices in the comparison graph which makes the bounds quite loose. In contrast, our results are tight up to constants and, as we detail in the following sections, provide deeper insights into the role of the topology of the comparison graph. We elaborate on these differences in the appropriate sections in the sequel. We also note that the models studied in the present paper as well as in the aforementioned works fall under the broader paradigm of random utility models (Thurstone, 1927; Train, 2009; Azari Soufiani et al., 2013).

In other related works, Jagabathula and Shah (2008) design an algorithm for aggregating ordinal data when the underlying distribution over the permutations is assumed to be sparse. Ammar and Shah (2011) employ a different, maximum entropy approach towards parameterization and inference from partially ranked data. Rajkumar and Agarwal (2014) study the statistical convergence properties of several rank aggregation algorithms.

Our work assumes a fixed design setup. In this setup, the choice of which pairs to compare and the number of times to compare them is chosen ahead of time in a non-adaptive fashion. There is a parallel line of literature on "sorting" or "active ranking" from pairwise comparisons. For instance, Braverman and Mossel (2008) assume a noise model where the outcome of a pairwise comparison depends only on the relative ranks of the items being compared, and not on their actual ranks or values. On the other hand, Jamieson and Nowak (2011) consider the problem of ranking a set of items assuming that items can be embedded into a smaller-dimensional Euclidean space, and that the outcomes of the pairwise comparisons are based on the relative distances of these items from a fixed reference point in the Euclidean space.

A recent line of work considers a variant of the BTL and the Thurstone models where the comparisons may depend on some auxiliary unknown variable in addition to the items being compared; for instance, the accuracy of the individual making the comparison in an objective task. Chen et al. (2013) consider a crowdsourcing setup where the outcome depends on the worker's expertise. They present algorithms for inference under such a model and present empirical evaluations. Yi et al. (2013) consider a problem in the spirit of collaborative filtering where certain unknown preferences of a certain user must be predicted based on the preferences of other users as well as of that user over other items. Lee et al. (2011) consider the inverse problem of measuring the expertise of individuals based on the rankings submitted by them, and the proposed algorithms assume an underlying Thurstone model. Shah et al. (2013) make a case for ordinal evaluations in certain types of MOOC homeworks/exams and study a variant of the BTL model.

1.2 Our contributions

Both the Thurstone (Case V) and BTL models involve an unknown vector $w^* \in \mathbb{R}^d$ corresponding to the underlying qualities of d items, and in a pairwise comparison between items j and k, the probability of j being ranked above k is some function F of the difference $w^*_j - w^*_k$. The Thurstone (Case V) and BTL are based on different choices of F, and both belong to the broader class of models analyzed in this paper, in which F is required only to be strongly log-concave.

With this context, the main contributions of this paper are to provide some answers to the following questions:

- How does the minimax error for estimating the weight vector w^* in various norms scale with the problem dimension (the number of items) and the number of observations?
  - We derive upper and lower bounds on the minimax estimation rates under the model described above. Our upper/lower bounds on the estimation error agree up to constant factors: to the best of our knowledge, despite the voluminous literature on these two models, this provides the first sharp characterization of the associated minimax rates. Moreover, our error guarantees provide guidance to the practitioner in assessing the number of pairwise comparisons to be made in order to guarantee a pre-specified accuracy.
- Given a budget of n comparisons, which pairs of items should be compared?
  - The bounds that we derive depend on the comparison graph induced by the subset of pairs that are compared. Our theoretical analysis reveals that the spectral gap of a certain scaled version of the graph Laplacian plays a fundamental role, and provides guidelines for the practitioner on how to choose the subset of comparisons to be made.
- When is it better to elicit pairwise comparisons versus numeric scores?
  - When eliciting data, one often has the liberty to ask for either cardinal values (Figure 1b) or for pairwise comparisons (Figure 1a) from the human subjects. One would like to adopt the approach that would lead to a better estimate. One may be tempted to think that cardinal elicitation methods are superior, since each cardinal measurement

gives a real-valued number whereas an ordinal measurement provides at most one bit of information. Our bounds show, however, that the scaling of the error in the cardinal and ordinal settings is identical up to constant pre-factors. As we demonstrate, this result allows for a comparison of cardinal and ordinal data elicitation methods in terms of the per-measurement noise alone, *independent* of the number of measurements and the number of items. A priori, there is no obvious reason for the relative performance to be independent of the number of measurements and items.

Notation: For any symmetric matrix M of size $(m \times m)$, we let $\lambda_1(M) \leq \lambda_2(M) \leq \cdots \leq \lambda_m(M)$ denote its ordered eigenvalues. We use the notation $D_{\mathrm{KL}}(\mathbb{P}_1 \| \mathbb{P}_2)$ to denote the Kullback-Leibler divergence between the two distributions $\mathbb{P}_1$ and $\mathbb{P}_2$. For any integer m, we let $[m]$ denote the set $\{1, \ldots, m\}$. For any pair of vectors u and v of the same length, we let $\langle u, v \rangle$ denote their inner product.

2. Problem formulation

We begin with some background followed by a precise formulation of the problem.

2.1 Generative models for ranking

Given a collection of d items to be evaluated, we suppose that each item has a certain numeric *quality score*, and a comparison of any pair of items is generated via a comparison of the two quality scores in the presence of noise. We represent the quality scores as a vector $w^* \in \mathbb{R}^d$, so item $j \in [d]$ has quality score w_j^*. Now suppose that we make n pairwise comparisons: if comparison $i \in [n]$ pertains to comparing item a_i with item b_i, then it can be described by a differencing vector $x_i \in \mathbb{R}^d$, with entry a_i equal to one, entry b_i equal to -1, and the remaining entries set to 0.

With this notation, we study the problem of estimating the weight vector w^* based on observing a collection of n independent samples $y_i \in \{-1, 1\}$ drawn from the distribution

$$\mathbb{P}\big[y_i = 1 | x_i, w^*\big] = F\Big(\frac{\langle x_i,\, w^* \rangle}{\sigma}\Big) \quad \text{for } i \in [n], \tag{Ordinal}$$

where F is a known function taking values in $[0, 1]$. Since the probability of item a_i dominating b_i should be independent of the order of the two items being compared, we require throughout that $F(x) = 1 - F(-x)$.

In any model of the general form (Ordinal), the parameter $\sigma > 0$, assumed to be known, plays the role of a noise parameter, with a higher value of σ leading to more uncertainty in the comparisons. Moreover, we assume that F is *strongly log-concave* in a neighborhood of the origin, meaning that there is some curvature parameter $\gamma > 0$ such that

$$\frac{d^2}{dt^2}(-\log F(t)) \geq \gamma \quad \text{for all } t \in [-2B/\sigma, 2B/\sigma]. \tag{1}$$

Here the known parameter B denotes a bound on the ℓ_∞-norm of the weight vector, namely

$$\|w^*\|_\infty \leq B.$$

As our analysis shows, a bound of this form is fundamental: the minimax error for estimating w^* diverges to infinity if we are allowed to consider models in which B is arbitrarily large (see Proposition 17 in Appendix G). Informally, this behavior is related to the difficulty of estimating very small (or very large) probabilities that can arise in the two models for large $\|w^*\|_\infty$. Note that any model of the form (Ordinal) is invariant to shifts in w^*, that is, it does not differentiate between the vector w^* and the shifted vector $w^* + 1$, where 1 denotes the vector of all ones. Therefore, in order to ensure identifiability of w^*, we assume throughout that $\langle 1,\, w^* \rangle = 0$. We use the notation $\mathcal{W}_B$ to denote the set of permissible quality score vectors

$$\mathcal{W}_B := \big\{ w \in \mathbb{R}^d \mid \|w\|_\infty \leq B, \quad \text{and} \quad \langle 1,\, w \rangle = 0 \big\}. \tag{2}$$

Both the Thurstone (Case V) model with Gaussian noise (Thurstone, 1927) and Bradley-Terry-Luce (BTL) models (Bradley and Terry, 1952; Luce, 1959) are special cases of this general set-up, as we now describe.

Thurstone (Case V): This model is is a special case of the family (Ordinal), obtained by setting

$$F(t) = \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du, \tag{3}$$

corresponding to the CDF of the standard normal distribution. Consequently, the Thurstone model can alternatively be written as making n i.i.d. observations of the form

$$y_i = \operatorname{sign}\Big\{ \langle x_i,\, w^* \rangle + \epsilon_i \Big\}, \quad \text{for } i \in [n], \tag{Thurstone}$$

where $\epsilon_i \sim N(0, \sigma^2)$ is observation noise. It can be verified that the Thurstone model is strongly log-concave over the set $\mathcal{W}_B$ (e.g., see Tsukida and Gupta (2011)).

Bradley-Terry-Luce: The Bradley-Terry-Luce (BTL) model (Bradley and Terry, 1952; Luce, 1959) is another special case in which

$$F(t) = \frac{1}{1 + e^{-t}},$$

and hence

$$\mathbb{P}\big[y_i = 1 | x_i, w^*\big] = \frac{1}{1 + \exp\big(-\frac{\langle x_i, w^* \rangle}{\sigma}\big)} \quad \text{for } i \in [n]. \tag{BTL}$$

It can also be verified that the BTL model is strongly log-concave over the set $\mathcal{W}_B$.

Cardinal observation models: While our primary focus is on the pairwise-comparison setting, for comparison purposes we also analyze analogous cardinal settings where each observation is real valued. In particular, we consider the following two cardinal analogues of the Thurstone model. In the Cardinal model we consider, each observation $i \in [n]$ consists of a numeric evaluation $y_i \in \mathbb{R}$ of a single item,

$$y_i = \langle u_i,\, w^* \rangle + \epsilon_i \qquad \text{for } i \in [n], \tag{Cardinal}$$

where u_i in this case is a coordinate vector with one of its entries equal to 1 and remaining entries equal to 0, and ϵ_i is independent Gaussian noise $N(0, \sigma^2)$. One may alternatively elicit cardinal values of the differences between pairs of items

$$y_i = \langle x_i, w^* \rangle + \epsilon_i \qquad \text{for } i \in [n], \tag{Paired Cardinal}$$

where ϵ_i are i.i.d. $N(0, \sigma^2)$. We term this model the Paired Cardinal model.

2.2 Fixed design and the graph Laplacian

We analyze the estimation error when a fixed subset of pairs is chosen for comparison. Of interest to us is the *comparison graph* defined by these chosen pairs, with each pair inducing an edge in the graph. Edge weights are determined by the fraction of times a given pair is compared. The analysis in the sequel reveals the central role played by the Laplacian of this weighted graph. Note that we are operating in a fixed-design setup where the graph is constructed offline and does not depend on the observations.

In the ordinal models, the i^{th} measurement is related to the difference between the two items being compared, as defined by the measurement vector $x_i \in \mathbb{R}^d$. We let $X \in \mathbb{R}^{n \times d}$ denote the measurement matrix with the vector x_i^T as its i^{th} row. The Laplacian matrix L associated with this differencing matrix is given by

$$L := \frac{1}{n} X^T X \;=\; \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T. \tag{4}$$

By construction, for any vector $v \in \mathbb{R}^d$, we have $v^T L v = \sum_{j \neq k} L_{jk} (v_j - v_k)^2$, where L_{jk} is the fraction of the measurement vectors $\{x_i\}_{i=1}^n$ in which items (j, k) are compared.

The Laplacian matrix is positive semidefinite, and has at least one zero-eigenvalue, corresponding to the all-ones eigenvector. The Laplacian matrix induces a graph on the vertex set $\{1, \ldots, d\}$, in which a given pair (j, k) is included as an edge if and only if $L_{jk} \neq 0$, and the weight on an edge (j, k) equals L_{jk}. We emphasize that throughout our analysis, we assume that the comparison graph is *connected*, since otherwise, the quality score vector w^* is not identifiable. Note that the Laplacian matrix L induces a semi-norm[1] on $\mathbb{R}^d$, given by

$$\|u - v\|_L := \sqrt{(u - v)^T L (u - v)}. \tag{5}$$

We study optimal rates of estimation in this semi-norm, as well as the usual ℓ_2-norm. As will be clearer in the sequel the L semi-norm is a natural metric in our setup, and estimation in this induced metric can be done at a topology independent rate. The estimation error in the L semi-norm is closely related to the prediction risk in generalized linear models. In particular, for an estimate $\widehat{w}$ of w^* from n_{ij} comparisons between each pair of items (i, j), we have $\|\widehat{w} - w^*\|_L^2 = \sum_{i<j} n_{ij} \big((\widehat{w}_i - \widehat{w}_j) - (w_i^* - w_j^*)\big)^2$. It arises naturally when one is interested in predicting the probability of a certain outcome for a new comparison.

1. A semi-norm differs from a norm in that the semi-norm of a non-zero element is allowed to be zero.

3. Bounds on the minimax risk

In this section, we state the main results of the paper, and discuss some of their consequences.

3.1 Minimax rates in the squared L semi-norm

Our first main result provides bounds on the minimax risk under the squared L semi-norm (5) in the pairwise comparison models introduced earlier. In all of the statements, we use c_1, c_2, etc. to denote positive numerical constants, independent of the sample size n, number of items d and other problem-dependent parameters.

Apart from the curvature parameter γ defined earlier in (1), the bounds presented subsequently depend on F through a second parameter ζ, defined as

$$\zeta := \frac{\max\limits_{x \in [0, 2B/\sigma]} F'(x)}{F(2B/\sigma)(1 - F(2B/\sigma))}. \tag{6}$$

In the BTL and the Thurstone models, we have $\zeta := \frac{F'(0)}{F(2B/\sigma)(1-F(2B/\sigma))}$. For instance, when $B = \sigma = 1$, then under the BTL model we have $\gamma = 0.25$ and $\zeta = 1.43$. As observed in Negahban et al. (2015), the reader may consider the parameters B, σ, γ and ζ as having $\mathcal{O}(1)$ values: a fixed value of B is the hardest regime, and furthermore, in situations of practical interest, the problem dependent parameters σ, γ and ζ, are typically independent of d and n. Consequently, in the sequel, we treat these parameters as fixed.

Theorem 1 (Bounds on minimax rates in L semi-norm)

(a) For a sample size $n \geq \frac{c_1 \sigma^2 \operatorname{tr}(L^\dagger)}{\zeta B^2}$, any estimator $\widetilde{w}$ based on n samples from the Ordinal *model has Laplacian squared error lower bounded as*

$$\sup_{w^* \in \mathcal{W}_B} \mathbb{E}\Big[\|\widetilde{w} - w^*\|_L^2\Big] \geq \frac{c_{1\ell}}{\zeta}\, \sigma^2 \frac{d}{n}. \tag{7a}$$

(b) For any instance of the Ordinal *model with γ-strong log-concavity and any $w^* \in \mathcal{W}_B$, the maximum likelihood estimator satisfies the bound*

$$\mathbb{P}\Big[\|\widehat{w}_{ML} - w^*\|_L^2 > t \frac{c \zeta^2 \sigma^2}{\gamma^2} \frac{d}{n}\Big] \leq e^{-t} \quad \text{for all } t \geq 1,$$

and consequently

$$\sup_{w^* \in \mathcal{W}_B} \mathbb{E}\Big[\|\widehat{w}_{ML} - w^*\|_L^2\Big] \leq \frac{c_{1u} \zeta^2}{\gamma^2} \sigma^2 \frac{d}{n}. \tag{7b}$$

The results of Theorem 1 characterize the minimax risk in the squared L semi-norm up to constant factors. The upper bounds follow from an analysis of the maximum likelihood estimator, which turns out to be a convex optimization problem. On the other hand, the lower bounds are based on a combination of information-theoretic techniques and carefully constructed packings of the parameter set $\mathcal{W}_B$. The main technical difficulty is in constructing a packing in the semi-norm induced by the Laplacian L. See Appendix A for the full proof.

3.2 Minimax rates in the squared ℓ_2-norm

Let us now turn to optimizing the minimax risk under the squared Euclidean norm. Theorem 2 below presents upper and lower bounds on this quantity.

Theorem 2 (Bounds on minimax rates in ℓ_2-norm)

(a) For a sample size $n \geq \frac{c_2 \sigma^2 \mathrm{tr}(L^\dagger)}{\zeta B^2}$, any estimator $\widetilde{w}$ based on n samples from the Ordinal *model has squared Euclidean error lower bounded as*

$$\sup_{w^* \in \mathcal{W}_B} \mathbb{E}\Big[\|\widetilde{w} - w^*\|_2^2\Big] \geq c_{2\ell} \frac{\sigma^2}{n} \max\Big\{d^2, \max_{d' \in \{2,\ldots,d\}} \sum_{i=\lceil 0.99 d' \rceil}^{d'} \frac{1}{\lambda_i(L)}\Big\}. \tag{8a}$$

(b) For any instance of the Ordinal *model with γ-strong log-concavity and any $w^* \in \mathcal{W}_B$, the maximum likelihood estimator satisfies the bound*

$$\mathbb{P}\Big[\|\widehat{w}_{ML} - w^*\|_2^2 > t \frac{c\zeta^2\sigma^2}{\gamma^2} \frac{d}{\lambda_2(L)n}\Big] \leq e^{-t} \quad \textit{for all } t \geq 1, \tag{8b}$$

and consequently

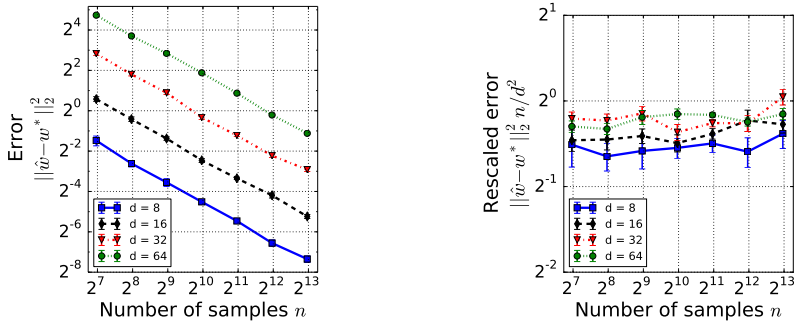
$$\sup_{w^* \in \mathcal{W}_B} \mathbb{E}\Big[\|\widehat{w}_{ML} - w^*\|_2^2\Big] \leq \frac{c_{2u}\zeta^2}{\gamma^2} \sigma^2 \frac{d}{\lambda_2(L)n}. \tag{8c}$$

See Appendix B for the proof of this theorem. As we describe in the next section, the upper and lower bounds on minimax risk from Theorem 2 to identify the comparison graph(s) that lead to the best possible minimax risk over all possible graph topologies.

Figure 2 depicts results from simulations under the Thurstone model, depicting the squared ℓ_2 error for the maximum likelihood estimator for various values of n and d. In the simulations, the true vector w^* is generated by first drawing a d-length vector uniformly at random from $[-1, 1]^d$, followed by a scale and shift to ensure $w^* \in \mathcal{W}_B$. The n pairs are chosen uniformly (with replacement) at random from the set of $\binom{d}{2}$ possible pairs of items. The value of σ and B are both fixed to be 1. Given the n samples, inference is performed via the maximum likelihood estimator for the Thurstone model. Each point in the plots is an average of 20 such trials.

The error in Figure 2 reduces linearly with n, exactly as predicted by our Theorem 2. For the complete graph, $\frac{1}{\lambda_2(L)} = \frac{d-1}{2}$. Theorem 2 thus predicts a quadratic increase in the error with d. As predicted, the error when normalized by $\frac{1}{d^2}$ in Figure 2 converges to the same curve for all values of d.

Detailed comparison with other work: Having stated our main theoretical results we are now in a position to revisit the results of the earlier works of Negahban et al. (2012, 2015); Hajek et al. (2014). The papers by Negahban et al. (2012, 2015) consider the BTL model under a restricted sampling setting in which every pair considered is compared the same number of times. For the problem of recovering the vector w^* (as considered in the present paper), they derive upper bounds when the pairs considered for comparisons arise



(a) Error (b) Rescaled error

Figure 2: Simulation results under the Thurstone model. The comparison topology chosen here is the complete graph.

from an Erdős-Rényi comparison graph; setting $t = \log d$ in equation (8b) in Theorem 2 above recovers their results. The two papers also provide results pertaining to inference of a set of parameters that are an exponentiated form of w^*. Their upper bounds for these parameters are derived in terms of the random-walk normalized Laplacian matrix of the comparison graph, while our bounds for w^* are derived in terms of the (combinatorial) Laplacian matrix. Their associated information-theoretic lower bounds also assume an Erdős-Rényi comparison graph whereas our lower bounds apply to general comparison graphs.

A concurrent paper by Hajek et al. (2014) also considers the specific BTL model, but a general comparison topology. Their high-probability upper bounds can be recovered by setting $t = \log d$ in equation (8b) of Theorem 2, while their upper bounds on the expected error are loose by a logarithmic factor. On the other hand, the lower bounds of Hajek et al. (2014), although dependent on topology, are quite loose due to their reliance on the degrees of the vertices in the comparison graph. On the other hand, our results are derived in terms of the graph Laplacian which better captures the critical aspects of the problem. For instance, considering a disconnected graph where every node has at least one neighbor as a sanity check, the degree-based lower bounds of Hajek et al. (2014) do not reflect the non-identifiability of w^* due to their reliance on only the degrees of the vertices, whereas the presence of the spectral gap in our bound (8a) indicates non-identifiability. As another example, the bounds of Hajek et al. (2014) cannot distinguish between a star graph and a path graph, whereas our results establish the star graph as an optimal comparison graph and the path graph as strictly suboptimal. In Section 4 below, we further describe deeper insights on the graph topology derived from our analytical results.

The Paired Cardinal *model:* Before concluding this section, we also look at the Paired Cardinal model (Section 2.1), the cardinal analogue of the Thurstone model.

Theorem 3 (Bounds on minimax rates in ℓ_2-norm) *For the* Paired Cardinal *model, the minimax risk is sandwiched as*

$$c_{3\ell}\,\sigma^2\frac{\mathrm{tr}(L^\dagger)}{n} \le \inf_{\widehat{w}}\sup_{w^*\in\mathcal{W}_\infty}\mathbb{E}\Big[\|\widehat{w}-w^*\|_2^2\Big] \le c_{3u}\,\sigma^2\frac{\mathrm{tr}(L^\dagger)}{n}. \tag{9}$$

The proof of Theorem 3 is available in Appendix C.

We conjecture that the dependence of the squared ℓ_2 minimax risk under the Ordinal models on the problem parameters n, d and the graph topology is identical to that derived in Theorem 3 for the Paired Cardinal model, i.e., is proportional to $\frac{\mathrm{tr}(L^\dagger)}{n}$. Note that the condition $\mathrm{tr}(L)=2$ implies that $\frac{d^2}{9} \le \mathrm{tr}(L^\dagger) \le \frac{d}{\lambda_2(L)}$.

3.3 Extension to m-ary comparisons

Suppose instead of eliciting pairwise comparisons, one can instead ask the workers to make comparisons between more than two options. In particular, we assume that each sample is a selection of the item with the largest perceived quality among some m presented items. The setting of pairwise comparisons is a special case with $m=2$. Recall from Theorem 2 that the minimum squared ℓ_2 minimax risk in the pairwise comparison setting is of the order $\frac{d^2}{n}$. Our goal in this section is to bring the concept of multiple-item comparison under the same framework as the pairwise case, and via a generalization of our earlier theoretical analysis, understand how the error exponent depends on m.

Consider d items, where every item $j\in[d]$ has a certain underlying quality score $w_j^*\in[-B,B]$. Suppose you have access to n samples, with each sample being a selection of the item with the largest perceived value among some m presented items.

Consider $(d\times m)$ matrices $E_1,\ldots,E_n$ such that for each $i\in[n]$, the m columns of E_i are distinct unit vectors. The positions of the non-zero elements in the m columns of E_i represent the identities of the m items compared in the i^{th} sample. One can visualize the choices of the items compared as a hyper-graph, with d vertices representing the d items and hyper-edge $i\in[n]$ containing the m items compared in observation i.

Let $R_1,\ldots,R_m$ be $(m\times m)$ permutation matrices representing m cyclic shifts in an arbitrary (but fixed) direction. Consider the observation model

$$\mathbb{P}(y_i=j|w^*,E_i) = F((w^*)^TE_iR_j)$$

for all $j\in[m]$, where $F:[-B,B]^m\to[0,1]$ represents the probability of choosing the *first* among the m items presented. We also assume that F does not depend on the *order* of the last $(m-1)$ coordinates in its input, meaning that the likelihood of choosing an item is independent of the ordering of the m items in the argument to F. For every $x\in[-B,B]^m$, $F(x)$ is assumed to satisfy:

- Shift-invariance: the probabilities depend only on the *differences* in the weights of the items presented, i.e, $F(x)$ depends only on $\{x_i-x_j\}_{i,j\in[m]}$.
- Strong log-concavity: $\nabla^2(-\log F(x)) \succeq H$ for some $(m\times m)$ symmetric matrix H with $\lambda_2(H)>0$.

Note that the shift-invariance assumption implies $1\in\mathrm{nullspace}(\nabla^2(-\log F(x)))$, thereby necessitating $\mathrm{nullspace}(H)=\mathrm{span}(1)$ and $\lambda_1(H)=0$. One can also verify that the model proposed here reduces to the Ordinal model of Section 2.1 when $m=2$.

For any hope of inferring the true weights w^*, we must ensure that the comparison hyper-graph is "connected", i.e., for every pair of items $i,j\in[d]$, there must exist a path connecting item i and item j in the comparison hyper-graph. We assume this condition is satisfied. We also continue to assume that $w^*\in\mathcal{W}_B := \{w\in\mathbb{R}^d \mid \|w\|_\infty\le B,\ \langle w,1\rangle=0\}$.

The popular Plackett-Luce model falls in this class, as illustrated below.

Example 1 (Plackett-Luce model (Plackett, 1975; Luce, 1959)) *The Plackett-Luce model concerns the process of choosing an item from a given set. Specifically, given m items with quality scores $w_1^*,\ldots,w_m^*$ respectively, the likelihood of choosing item $\ell\in[m]$ under this model is given by*

$$\frac{e^{w_\ell^*}}{\sum_{j=1}^m e^{w_j^*}} =: F([w_\ell^*,w_1^*,\ldots,w_{\ell-1}^*,w_{\ell+1}^*,\ldots,w_m^*]).$$

Every choice is made independent of all other choices.

It is easy to verify that the Plackett-Luce model satisfies shift invariance. Furthermore, the function F does not depend on the ordering of the last $(m-1)$ coordinates in its argument. We now show that it also satisfies strong log-concavity. A little algebra gives

$$\nabla^2(-\log F(x)) = \frac{\langle e^x,1\rangle\mathrm{diag}(e^x)-e^x(e^x)^T}{(\langle e^x,1\rangle)^2},$$

where $e^x := [e^{x_1}\cdots e^{x_m}]^T$. We now derive a lower bound for the expression above. An application of the Cauchy-Schwarz inequality yields that for any vector $v\in\mathbb{R}^m$,

$$v^T(e^x(e^x)^T)v \le v^T\mathrm{diag}(e^x)\langle e^x,1\rangle v,$$

with equality if and only if $v\in\mathrm{span}(1)$. It follows that $\lambda_2(\nabla^2(-\log F(x)))>0$ for all $x\in[-B,B]^m$. Defining the scalar $\beta := \min_{x\in[-B,B]^m}\lambda_2(\frac{\langle e^x,1\rangle\mathrm{diag}(e^x)-e^x(e^x)^T}{(\langle e^x,1\rangle)^2})$, one can see that setting $H=\beta(I-11^T)$ satisfies the strong log-concavity conditions.

Our goal is to capture the scaling of the minimax error with respect to the number of observations n, the dimension d of the problem, and the choice of the subsets compared $\{E_i\}_{i\in[n]}$. It is well understood (Miller, 1956; Kiger, 1984; Shiffrin and Nosofsky, 1994; Saaty and Ozdemir, 2003) that humans have a limited information storage and processing capacity, which makes it difficult to compare more than a small number of items. For instance, Saaty and Ozdemir (2003) recommend eliciting preferences over *no more than seven options*. Thus in this work we restrict our attention to $m=\mathcal{O}(1)$. Moreover, the amount of noise in the selection process also depends on the number of items m presented at a time: the higher the number, the greater the noise. We thus do not use a "noise parameter σ"in this setting, and assume the noise to be incorporated in the function F, which itself is a function of m.

Our results involve the Laplacian of the comparison graph, defined for the m-wise comparison setting as follows. Let L be an $(d \times d)$ matrix that depends on the choice of the comparison topology as

$$L := \frac{1}{n}\sum_{i=1}^{n} E_i(mI - 11^T)E_i^T. \tag{10}$$

We call L the Laplacian of the comparison hyper-graph. One can verify that when applied to the special case of $m = 2$, the matrix L defined in (10) reduces to the Laplacian of the pairwise-comparison graph defined earlier in (4).

The following theorem presents our main results for the m-wise comparison setting.

Theorem 4 *For the m-*wise *model, the minimax risk is sandwiched as*

$$c_{3\ell}\frac{\inf_z F(z)}{m^2\lambda_m(H)\sup_z \|\nabla F(z)\|_{H^\dagger}^2}\frac{d}{n} \le \inf_{\widehat{w}}\sup_{w^*\in\mathcal{W}_B}\mathbb{E}\left[\|\widehat{w}-w^*\|_L^2\right] \le c_{3u}\frac{m^2\sup_z\|\nabla\log F(z)\|_2^2}{\lambda_2(H)^2}\frac{d}{n},$$

in the squared L semi-norm and as

$$c_{4\ell}\frac{\inf_z F(z)}{m^2\lambda_m(H)\sup_z \|\nabla F(z)\|_{H^\dagger}^2}\frac{d^2}{n} \le \inf_{\widehat{w}}\sup_{w^*\in\mathcal{W}_B}\mathbb{E}\left[\|\widehat{w}-w^*\|_2^2\right] \le c_{4u}\frac{m^2\sup_z\|\nabla\log F(z)\|_2^2}{\lambda_2(H)^2}\frac{d}{\lambda_2(L)n},$$

in the squared ℓ_2 norm. Here we assume $n \ge c_5\frac{\mathrm{tr}(L^\dagger)\inf_z F(z)}{B^2\lambda_m(H)\sup_z\|\nabla F(z)\|_{H^\dagger}^2}$ for both the lower bounds, and where the suprema and infima with respect to the parameter z are taken over the set $[-B,B]^m$.

The proof of Theorem 4 is provided in Appendix D. Our results establish that the dependence of the squared L semi-norm and squared Euclidean minimax error on m occurs only as multiplicative pre-factors, and the error exponent is independent of m. Thus, if one follows the standard recommendation in the psychology literature Miller (1956); Kiger (1984); Shiffrin and Nosofsky (1994); Saaty and Ozdemir (2003)—namely to choose $m = \mathcal{O}(1)$—then the best possible scaling of the squared L semi-norm minimax risk with respect to d and n is always $\frac{d}{n}$, that of the squared Euclidean minimax risk is always $\frac{d^2}{n}$, and evenly spreading the samples across all possible choices of m items is optimal. Nevertheless, a more refined modeling and analysis is required to understand the precise tradeoffs governing the choice of the number m of items presented to the user.

Finally, when specialized to the case of $m = 2$, the upper bounds of Theorem 4 are identical (up to constant factors) to those of Theorem 1 and Theorem 2. The lower bound for the L semi-norm is identical to that of Theorem 1. The additional generality results in a lower bound for the ℓ_2 norm that is weaker in general as compared to Theorem 2, but is tight when the underlying hypergraph forms a complete graph.

4. Role of graph topology

We now return to the setting of pairwise comparisons. In certain applications, one may have the liberty to decide which pairs are compared. The results of the previous section

demonstrated the role played by the Laplacian of the comparison graph in the estimation error. We now employ these results to derive guidelines towards designing the comparison graph. Let us focus on the estimation error in the squared ℓ_2 norm in the ordinal setting. As discussed earlier, we assume that the graph induced by the comparisons is connected. An application of Theorem 2 lets us identify good topologies for pairwise comparisons in the fixed-design setup.

A popular class of comparison topologies is that of evenly distributed samples on an unweighted graph (e.g., Negahban et al. (2012)). Consider any fixed, unweighted graph $G = (V, E)$. We assume that the samples are distributed evenly along the edges E of G, and that the sample size n is sufficiently large. Using standard matrix concentration inequalities, it is straightforward to extend our analysis to the setting of random chosen comparisons from a fixed graph (see, for instance, Oliveira (2009)). Let L' denote the Laplacian of G. We define the *scaled Laplacian* of G as

$$L := \frac{1}{|E|}L'.$$

One can verify that the matrix L defined here is identical to what was defined in (4) in a more general context. In order to differentiate from L, we refer to L' as the *regular Laplacian* of the graph G. For a given budget n on the number of samples, we say that a comparison graph is *optimal* if the error under this graph is the smallest (up to constants) among all graphs.

4.1 Analytical results

Consider the Ordinal model and the squared ℓ_2-norm as the metric of interest. We claim that in order to determine whether a given comparison graph achieves minimax risk (up to a constant pre-factor), it suffices to examine the eigen-spectrum of the scaled Laplacian matrix. In particular, we claim that:

- If the scaled Laplacian has a second smallest eigenvalue that scales as $\frac{1}{\lambda_2(L)} = \Theta(d)$, then the comparison graph is optimal, and leads to the smallest possible minimax risk, in particular one that scales as $\frac{d^2}{n}$.
- Conversely, if the scaled Laplacian matrix has an eigen-spectrum satisfying

$$d^2 = o\left(\max_{d'\in\{2,\ldots,d\}}\sum_{i=\lfloor 0.99d'\rfloor}^{d'}\frac{1}{\lambda_i(L)}\right), \tag{11}$$

  then the associated estimation error is strictly larger than the minimax risk. In particular, this sub-optimality holds whenever $d^2 = o(\frac{1}{\lambda_2(L)})$.

In order to verify these claims, we note that by definition (4) of the Laplacian matrix, we have

$$\mathrm{tr}(L) = \frac{1}{n}\sum_{i=1}^{n}\mathrm{tr}(x_i x_i^T) = 2.$$

It follows that $\lambda_2(L) \le \frac{2}{d-1}$, i.e., that $\frac{1}{\lambda_2(L)} = \Omega(d)$. As we will see shortly, several classes of graphs satisfy $\frac{1}{\lambda_2(L)} = \Theta(d)$. Comparing the lower bound of $\Omega(\frac{d^2}{n})$ on the minimax risk (8a) with the upper bound (8c) gives the sufficient condition of $\frac{1}{\lambda_2(L)} = \Theta(d)$ for optimality, and the smallest minimax risk as $\Theta(\frac{d^2}{n})$. The lower bound (8a) now also gives the claimed condition for strict sub-optimality.

In order to illustrate these claims, let us consider a few canonical classes of graphs, and study how the estimation error under the squared Euclidean norm scales in the Ordinal model. The spectra of the regular Laplacian matrices of these graphs can be found in various standard texts on spectral graph theory (e.g., Brouwer and Haemers (2011)).

- **Complete graph.** A complete graph has one edge between every pair of nodes. The spectrum of the regular Laplacian of the complete graph is $0, d, \ldots, d$, and hence the spectrum of the scaled Laplacian L is $0, \frac{2}{d-1}, \ldots, \frac{2}{d-1}$. Substituting $\lambda_2(L) = \frac{2}{d-1}$ in Theorem 2b gives an upper bound of $\Theta(\frac{d^2}{n})$ on the minimax risk, and Theorem 2 gives a matching lower bound. The sufficiency condition discussed above proves optimality.

- **Constant-degree expander.** The spectrum of the regular Laplacian of a constant-degree expander graph is $0, \Theta(d), \Omega(d), \ldots, \Omega(d)$. Since the number of edges is $\Theta(d)$, the spectrum of the scaled Laplacian equals $0, \Theta(\frac{1}{d}), \Omega(\frac{1}{d}), \ldots, \Omega(\frac{1}{d})$. The evaluation of this class of graphs with respect to the minimax risk is identical to that of complete graphs, giving a lower and upper bound of $\Theta(\frac{d^2}{n})$ on the minimax risk, and guaranteeing optimality.

- **Complete bipartite.** The d nodes are partitioned into two sets comprising, say, m_1 and m_2 nodes. There is an edge between every pair of nodes in different sets, and there are no edges between any two nodes in the same set. The eigenvalues of the regular Laplacian of this graph are $0, \underbrace{m_2, \ldots, m_2}_{m_1-1}, \underbrace{m_1, \ldots, m_1}_{m_2-1}, m_1 + m_2$. Since the total number of edges is $m_1 m_2$, the scaled Laplacian L has a spectrum $0, \underbrace{\frac{1}{m_1}, \ldots, \frac{1}{m_1}}_{m_1-1}, \underbrace{\frac{1}{m_2}, \ldots, \frac{1}{m_2}}_{m_2-1}, \frac{1}{m_1} + \frac{1}{m_2}$. Suppose without loss of generality that $m_1 \ge m_2$. Also suppose that $m_2 > 1$ (the case of $m_2 = 1$ is the star graph discussed below). Then we have $\frac{1}{m_1} \le \frac{1}{m_2} \le \frac{1}{m_1} + \frac{1}{m_2}$ and that $d > m_1 \ge \frac{d}{2}$. Furthermore since $m_2 > 1$, the multiplicity of $\frac{1}{m_1}$ in the spectrum of the scaled Laplacian is at least 1. Thus we have $\lambda_2(L) = \Theta(\frac{1}{d})$. Theorem 2 then gives lower and upper bounds on the minimax risk as $\Theta(\frac{d^2}{n})$ and the sufficiency condition discussed above guarantees its optimality.

- **Star.** A star graph has one central node with edges to every other node. It is a special case of the complete bipartite graph with $m_1 = d-1$ and $m_2 = 1$. The spectrum of the regular Laplacian is $0, 1, \ldots, 1, d$. Since there are $(d-1)$ edges, the spectrum of the scaled Laplacian is $0, \frac{1}{d-1}, \ldots, \frac{1}{d-1}, \frac{d}{d-1}$. Theorem 2 and the sufficiency condition discussed above imply that this class of graphs is optimal and is associated to a minimax risk of $\Theta(\frac{d^2}{n})$.

- **Path.** A path graph is associated to an arbitrary ordering of the d nodes with edges between pairs j and $(j+1)$ for every $j \in \{1, \ldots, d-1\}$. The spectrum of the regular Laplacian is given by $2\big(1 - \cos\big(\frac{\pi i}{d}\big)\big)$, $i \in \{0, \ldots, d-1\}$, and that of the scaled Laplacian is thus $\frac{2}{d-1}\big(1 - \cos\big(\frac{\pi i}{d}\big)\big)$, $i \in \{0, \ldots, d-1\}$. The relation $(1 - \cos x) = \sin^2 \frac{x}{2}$ and the approximation $\sin x \approx x$ for values of x close to zero gives $\lambda_2(L) = \Theta(\frac{1}{d^3})$. The minimax risk is thus upper bounded as $\mathcal{O}(\frac{d^4}{n})$ and lower bounded as $\Omega(\frac{d^3}{n})$. This class of graphs is thus strictly suboptimal.

- **Cycle.** A cycle is identical to a path except for an additional edge between node d and node 1. The spectrum of the regular Laplacian is given by $2\big(1 - \cos\big(\frac{2\pi i}{d}\big)\big)$, $i \in \{0, \ldots, d-1\}$, and that of the scaled Laplacian is thus $\frac{2}{d}\big(1 - \cos\big(\frac{2\pi i}{d}\big)\big)$, $i \in \{0, \ldots, d-1\}$. The relation $(1 - \cos x) = \sin^2 \frac{x}{2}$ and the approximation $\sin x \approx x$ for values of x close to zero gives $\lambda_2(L) = \Theta(\frac{1}{d^3})$. The minimax risk is thus upper bounded as $O(\frac{d^4}{n})$ and lower bounded as $\Omega(\frac{d^3}{n})$. This class of graphs is thus strictly suboptimal.

- **Barbell.** The nodes are partitioned into two sets of $\frac{d}{2}$ nodes each, and there is an edge between every pair of nodes within each set. In addition, there is exactly one edge across the sets. The spectrum of the regular Laplacian can be computed as $0, \Theta(\frac{1}{d}), \Theta(d), \ldots, \Theta(d)$. Since there are $\Theta(d^2)$ edges, the spectrum of the scaled Laplacian turns out to become $0, \Theta(\frac{1}{d^3}), \Theta(\frac{1}{d}), \ldots, \Theta(\frac{1}{d}), \Omega(\frac{1}{d})$. Applying the results derived earlier in the paper, we get that a lower bound of $\Omega(\frac{d^3}{n})$ and an upper bound of $O(\frac{d^4}{n})$ on the minimax risk, thereby also establishing the sub-optimality of this class of graphs.

- **2D Lattice.** An $(m_1 \times m_2)$ lattice has $d = m_1 m_2$ vertices arranged as a $(m_1 \times m_2)$ grid. Assume $m_1 = \Theta(d)$ and $m_2 = \Theta(d)$. This class of graphs can be written as a Cartesian product of a path graph of length m_1 and a second path graph of length m_2. As a result, the spectrum of the scaled Laplacian is $\frac{2}{d}\big(2 - \cos\big(\frac{\pi i}{m_1}\big) - \cos\big(\frac{\pi j}{m_2}\big)\big)$, $i \in \{0, \ldots, m_1 - 1\}, j \in \{0, \ldots, m_2 - 1\}$. Again, using the small angle approximation of the sinusoid, one can compute an upper bound on the minimax risk as $\mathcal{O}(\frac{d^3}{n})$ and a lower bound of $\Omega(\frac{d^2}{n})$. We do not know at this point whether the 2D lattice minimizes the minimax risk.

- **Hypercube.** Assume $d = 2^m$ for some integer m. Representing each node as a distinct m-length binary vector, an edge exists between the nodes corresponding to any pair of vectors within a Hamming distance of one. The hypercube is an m-fold Cartesian product of a path with two nodes, and hence the regular Laplacian has an eigenvalue of $2i$ with multiplicity $\binom{m}{i}$, for $i \in \{0, \ldots, m\}$. The scaled Laplacian has an eigenvalue of $\frac{2i}{d \log d}$ with multiplicity $\binom{m}{i}$, for $i \in \{0, \ldots, m\}$. A lower bound on the minimax risk is $\Omega(\frac{d^2}{n})$ and an upper bound is $O(\frac{d^2 \log d}{n})$. We do not know if the hypercube is optimal, our bounds do tell us that any sub-optimality is bounded by at most a logarithmic factor.

Observe that the degree-k expander requires $n \ge kd$ samples while the complete graph requires $n \ge \binom{d}{2}$ samples, so in practical applications at least for small sample sizes we should prefer a low-degree expander.

Finally, if the conjecture discussed at the end of Section 3.2 were true, namely that the ℓ_2 minimax risk scales as $\sigma^2 \mathrm{tr}(L^\dagger)/n$, then the condition $\mathrm{tr}(L^\dagger) = \Theta(d^2)$ would be necessary and sufficient for optimality of a comparison graph with the scaled Laplacian L. Observe that the graphs designated as 'optimal' in the discussion above indeed satisfy this condition. On the other hand, the graphs established as strictly suboptimal have $\mathrm{tr}(L^\dagger) = \Omega(d^3)$.

4.2 Experiments and simulations

This section evaluates the dependence of the squared ℓ_2-error on the topology of the comparison graph. We consider the following five topologies: path, barbell, complete, expander and 2D-lattice. In order to form an expander graph, we used the construction due to Gabber and Galil (1981). For any chosen graph topology, the n difference vectors are selected as one edge each chosen uniformly at random (with replacement) from the comparison graph. Recall that our theory predicts that the complete and expander graphs yield the best performance, and that the line and dumbbell graphs fare the worst. Also recall that our theory predicts the error scales as $\|w^* - \widehat{w}\|_2^2$ scales with n as $1/n$ in the complete and expander topologies.

4.2.1 Experiments on synthetic data

This section describes simulations using data generated synthetically from the Thurstone model. In the simulations, we first generate a quality score vector $w^* \in \mathcal{W}_B$ using one of the procedures described below. Once w^* is chosen, the n pairwise comparisons for any given topology are generated as follows. An edge is selected uniformly (with replacement) at random from the underlying graph, and the chosen edge determines the pair of items compared. The outcome of the comparison is generated as per the Thurstone model with the chosen w^* as the underlying quality score. Finally, the maximum likelihood estimator for the Thurstone model is employed to estimate w^*. Every point in the plots is an average across 40 trials.

The following six procedures are employed to generated the true quality score vector w^* in the six respective subfigures of Figure 3.

(a) Gaussian: w^* is drawn from the standard normal distribution $\mathcal{N}(0, I)$.

(b) Uniform: w^* is drawn uniformly at random from the set $[-1, 1]^d$.

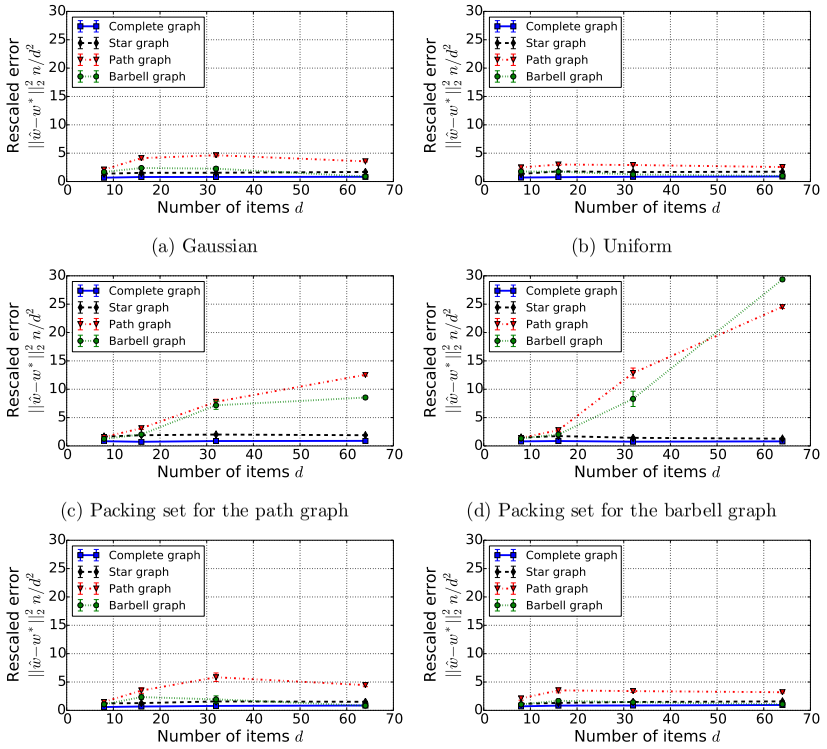
(c) Packing set for the path graph: We first choose a vector z as by setting a value of 0 in the first coordinate, a value -1 in $\frac{d}{2}$ of the other coordinates chosen uniformly at random, and a value 1 in the remaining coordinates. Letting $L = U^T \Lambda U$ denote the eigen-decomposition of the Laplacian matrix of the path graph, w^* is set as $U^T \Lambda^\dagger z$, where $\Lambda^\dagger$ is the Moore-Penrose pseudoinverse of Λ. This generation process mimics a construction used to prove the lower bound in Theorem 2, and tailors the construction for the path graph.

(d) Packing set for the barbell graph: The procedure is identical to that in (c), except that the Laplacian matrix used is that of the barbell graph.

(e) Packing set for the complete graph: The procedure is identical to that in (c), except that the Laplacian matrix used is that of the complete graph.

(f) Packing set for the star graph: The procedure is identical to that in (c), except that the Laplacian matrix used is that of the star graph.

The vector w^* generated in this procedure is then scaled and shifted to ensure $w^* \in \mathcal{W}_B$. The values of B and σ are set as 1.



(a) Gaussian (b) Uniform

(c) Packing set for the path graph (d) Packing set for the barbell graph

(e) Packing set for the complete graph (f) Packing set for the star graph

Figure 3: Estimation error under different topologies for different generative processes in the synthetic simulations.

Figure 3 plots the estimation error under various topologies of the comparison graph. Observe in the figure that the error is the lowest under the complete and the star graphs, and the highest under the barbell and the path graphs. In particular, the error consistently varies as $\Theta(d^2/n)$ for the complete and star graphs – this phenomenon holds even in plots (e) and (f) where the procedure to choose w^* forms the worst case for the complete and star graphs respectively according to the proof of Theorem 2. On the other hand, the minimax error varies as $\Omega(d^3/n)$ in the worst case for the path and the barbell graphs. Finally, observe that in the simulations, the (constant) multiplicative factors to the term $\frac{d^2}{n}$ in the error turn out to be rather small, in the range of 0 to 9.

Although not the primary focus of this paper, we note that our implementation for computing the maximum likelihood estimate under the Thurstone model requires several tens of minutes for modest problem dimensions. Computing the MLE is a convex optimization problem. Our implementation, which is not optimized for speed, employs the sequential least squares programming subroutine of the Scipy package in the Python programming language.

4.2.2 Experiments on MTurk

In this section, we describe the results of experiments conducted on the popular Amazon Mechanical Turk (`https://www.mturk.com/`; henceforth referred to as "MTurk") commercial crowdsourcing platform, evaluating the effects of the choice of the topology. MTurk is an online platform where individuals or businesses can put up a task, and any individual can log in and complete the tasks in exchange for a payment that is specified along with the task. In our experiments, each worker was offered 20 cents per completed task. A worker was allowed to do no more than one task in an experiment. Workers were required to answer all the questions in a task. Only those workers who had 100 or more prior approved works and an approval rate of 95% or higher were allowed. Workers from any country were allowed to participate, except for the task of estimating distances between cities (for which only USA-based workers were permitted since all questions involved American cities).

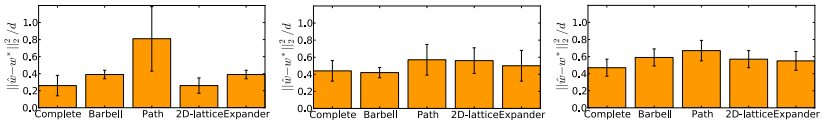
We conducted three experiments that required the workers to make ordinal choices.

(a) *Estimating areas of circles:* In each question, the worker was shown a circle in a bounding box (Figure 5a), and the worker was required to identify the fraction of the box's area that the circle occupied.

(b) *Estimating age of people from photographs:* The worker was shown photographs of people (Figure 5b) and was asked to estimate their ages.

(c) *Estimating distances between pairs of cities:* Pairs of cities were listed (Figure 5c) and for each pair, the worker had to estimate the distance between them.

For each experiment, we recruited 140 workers on MTurk, and assigned them to one of the five topologies uniformly at random. In this experiment and others involving aggregation of ordinal data from MTurk, the aggregation procedure follows maximum likelihood estimation under the Thurstone model, and the estimator is supplied the best-fitting value of σ obtained via 3-fold cross-validation. Each run of the estimation procedure employs the data provided by five randomly chosen workers from the pool of workers who performed



(a) Area of circle (b) Age from photograph (c) City distances

Figure 4: Estimation error under different topologies in the experiments conducted on MTurk. Also shown is the standard deviation across the estimates (The error bars are smaller by a factor of $\sqrt{\text{number of samples}}$).

that task. (The number five is inspired by practical systems as in Wang et al. (2011); Piech et al. (2013).) The complete dataset pertaining to these experiments is available on the first author's website.

Figure 4 plots the squared ℓ_2 estimation error for the three experiments under the five topologies considered, and the associated standard deviation. We see that the relative errors are generally consistent with our theory, with the complete graph exhibiting the best performance and the path graph faring the worst. On real datasets, model misspecification can in some cases cause the outcomes to differ from our theoretical predictions. Understanding the effect of model misspecification, especially on topology considerations, is an important question we hope to address in future work.

5. Cardinal versus ordinal measurements

In this section, we compare two approaches towards eliciting data: a score-based "cardinal" approach and a comparison-based "ordinal" approach. In a cardinal approach, evaluators directly enter numeric scores as their answers (Figure 1b), while an ordinal approach involves comparing (pairs of) items (Figure 1a).

There are obvious advantages and disadvantages associated with either approach. On one hand, the cardinal approach allows for very fine measurements. For instance, the cardinal measurements in Figure 1 can take any value between 0 and 100, whereas an ordinal measurement is binary. One might be tempted to go even further and argue that ordinal measurements necessarily give less information, for one can always convert a set of cardinal measurements into ordinal, simply by ordering the measurements by value. If this conversion were valid, the data processing inequality (Cover and Thomas, 2012), would then guarantee that estimators based on ordinal data can never outperform estimators based on cardinal data. However, this conversion assumes that cardinal and ordinal measurements suffer from the same type of statistical fluctuation. The following set of experiments show this assumption is false.

5.1 Raw data from MTurk

We conducted seven different experiments on MTurk to investigate the possibility of a "data-processing inequality" between the elicited cardinal and ordinal responses: Are responses elicited in ordinal form equivalent to data obtained by first eliciting cardinal responses and then subtracting pairs of items? Our experiments lead us to conclude that this is generally

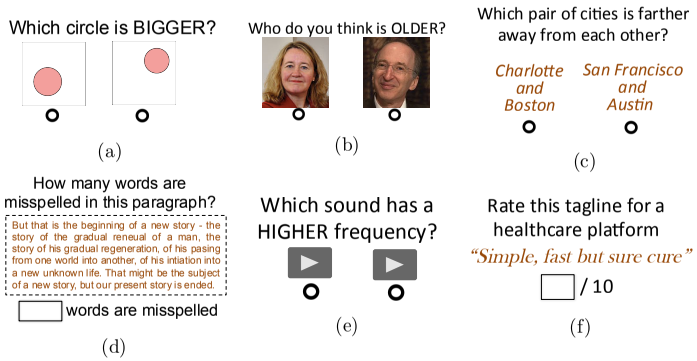


Figure 5: Screenshots of the tasks presented to the subjects. For each task, only one version (cardinal or ordinal) is shown here.

not the case: converting cardinally collected data into ordinal (by subtracting pairs of responses) often leads to a higher amount of noise as compared to that in data that is elicited directly in ordinal form.

The tasks were selected to have a broad coverage of several important subjective judgment paradigms such as preference elicitation, knowledge elicitation, audio and visual perception and skill utilization.

In addition to the three experiments described in Section 4.2.2, we conducted the following four experiments.

(d) *Finding spelling mistakes in text:* The worker had to identify the number of words that were misspelled in each paragraph shown (Figure 5d).

(e) *Identifying sounds:* The worker was presented with audio clips, each of which was the sound of a single key on a piano (which corresponds to a single frequency). The worker had to estimate the frequency of the sound in each audio clip (Figure 5e).

(f) *Rating tag-lines for a product:* A product was described and tag-lines for this product were shown (Figure 5f). The worker had to rate each of these tag-lines in terms of its originality, clarity and relevance to this product.

(g) *Rating relevance of the results of a search query:* Results for the query 'Internet' for an image search were shown (Figure 1) and the worker had to rate the relevance of these results with respect to the given query.

Note that the data collected for experiments (a)–(c) here was different and independent of the data collected for these tasks in Section 4.2.2.

The number of items d in the experiments ranged from 10 to 25. For each of the seven experiments, we recruited 100 workers, and assigned each worker to either the ordinal or the cardinal version of the task at random. Upon obtaining the data, we first reduced the

Task	Circle	Age	Distance	Spelling	Audio	Tagline	Relevance
Error in Ordinal	**6%**	**13%**	**17%**	**40%**	**20%**	44%	**31%**
Std. dev.	.23	.33	.38	.49	.40	.47	.44
Error in Cardinal	17%	17%	20%	42%	29%	**42%**	35%
Std. dev.	.31	.38	.38	.46	.43	.46	.44
Time in Ordinal	**98s**	**31s**	**84s**	**316s**	**66s**	**251s**	**105s**
Std. dev.	21.1	14.3	62.1	33.2	11.1	28.1	13.1
Time in Cardinal	181s	70s	144s	525s	134s	342s	185s
Std. dev.	39.9	33.1	56.2	46.0	12.4	44.6	28.2

Table 1: Comparison of the average amount of error when ordinal data is collected directly versus when cardinal data is collected and converted to ordinal. Also tabulated is the median time (in seconds) taken to complete a task by a subject in either type of task.

cardinal data obtained from the experiments into ordinal form by comparing answers given by the subjects to consecutive questions. For five of the experiments ((a) through (e)), we had access to the "ground truth" solutions, using which we computed the fraction of answers that were incorrect in the ordinal and the cardinal-converted-to-ordinal data (any tie in the latter case was counted as half an error). For the two remaining experiments ((f) and (g)) for which there is no ground truth, we computed the 'error' as the fraction of (ordinal or cardinal-converted-to-ordinal) answers provided by the subjects that disagreed with each other. It is important to note that in the experiments in this section, we did *not* run any estimation procedure on the data: we only measured the noise in the *raw responses*. The entire data pertaining to these experiments, including the interface seen by the workers and the data obtained from their work, is available on the first author's website.

The results are summarized in Table 1. If the cardinal measurements could always be converted to ordinal ones with the same noise level as directly eliciting ordinal responses, then it would be unlikely for the amount of error in the ordinal setting to be smaller than that in the cardinal setting. Table 1 shows that converting cardinal data to an ordinal form very often results in a higher (and sometimes significantly higher) per-sample error in the (raw) responses than direct elicitation of ordinal evaluations. Such an outcome may be explained by the argument that the inherent evaluation process in humans is not the same in the cardinal and ordinal cases: humans do *not* perform an ordinal evaluation by first performing cardinal evaluations and then comparing them (Barnett, 2003; Stewart et al., 2005). One can also see from Table 1 that the amount of time required for cardinal evaluations was typically (much) higher than for ordinal evaluations. One can thus assume that we typically have the per-observation error in the ordinal case lower than that in the cardinal case. In particular, considering the Thurstone and the Cardinal models (introduced in Section 2.1) with σ and σ_c denoting the standard deviations of the noise in the Thurstone and the Cardinal models respectively, the above empirical results imply that $\sigma < \sqrt{2}\sigma_c$.

5.2 Analytical comparison of Cardinal versus Ordinal

As discussed earlier, while cardinal measurements allow more flexibility in the range of responses, ordinal measurements contain a lower per-sample error. Ordinal measurements have additional benefits in that they avoid calibration issues that are frequently encountered in cardinal measurements (Tsukida and Gupta, 2011), such as the evaluators' inherent (and possibly time-varying) biases, or tendencies to give inflated or conservative evaluations. Ordinal measurements are also recognized to be easier or faster for humans to make (Barnett, 2003; Stewart et al., 2005), allowing for more evaluations with the same amount of time, effort and cost.

The lack of clarity regarding when to use a cardinal versus an ordinal approach forms the motivation of this section. Can we make as reliable estimates from paired comparisons as from numeric scores? How much lower does the noise have to be for comparative measurements to be preferred over cardinal measurements? The answers to these questions help to determine how responses should be elicited.

In order to compare the cardinal and ordinal methods of data elicitation, we focus on a setting with evenly budgeted measurements. In accordance with the fixed-design setup assumed throughout the paper, we choose the vectors x_i a priori. Suppose that n is large enough, and that in the ordinal case we compare each pair $n/\binom{d}{2}$ times. In the cardinal case suppose that we evaluate the quality of each item n/d times. We consider the Gaussian-noise models Thurstone and Cardinal introduced earlier in Section 2.1. In order to capture the fact that the amount of noise is different in the cardinal and ordinal settings, we denote the standard deviation of the noise in the cardinal setting as σ_c, and retain our notation of σ for the noise in the ordinal setting. In order to bring the two models on the same footing, we measure the error in terms of the squared ℓ_2-norm.

Let γ_G and ζ_G denote the parameters γ and ζ (defined in (1) and (6) respectively) specialized to the Gaussian distribution. Define $b_\ell(\sigma, B) := \frac{c_{2\ell}}{\zeta_G(B,\sigma)}$, $b_u(\sigma, B) := \frac{c_{2u}\zeta_G(B,\sigma)}{\gamma_G(B,\sigma)}$ and $b(\sigma, B) := \left\lceil \frac{c_2\sigma^2}{\zeta_G B^2} \right\rceil$. Observe that b_ℓ, b_u and b are independent of the parameters n and d.

With these preliminaries in place, we now compare the minimax error in the estimation under the cardinal and ordinal settings.

Proposition 5 *Given a sample size n that is a multiple of $d(d-1)b(\sigma, B)$, suppose that we observe each coordinate n/d times under the* Cardinal *model. Then the minimax risk is given by*

$$\inf_{\widehat{w}} \sup_{w^* \in \mathcal{W}_B} \mathbb{E}\Big[\|\widehat{w} - w^*\|_2^2\Big] = \sigma_c^2 \frac{d}{n}. \tag{12a}$$

Similarly, if we observe each pair $n/\binom{d}{2}$ times in the Thurstone *model, then the minimax risk is sandwiched as*

$$\sigma^2 b_\ell(\sigma, B)\frac{d}{n} \le \inf_{\widehat{w}} \sup_{w^* \in \mathcal{W}_B} \mathbb{E}\Big[\|\widehat{w} - w^*\|_2^2\Big] \le \sigma^2 b_u(\sigma, B)\frac{d}{n}. \tag{12b}$$

In the cardinal case, when each coordinate is measured the same number of times, the Cardinal model reduces to the well-studied normal location model, for which the MLE

is known to be the minimax estimator and its risk is straightforward to characterize (see Lehmann and Casella (1998) for instance). In the ordinal case, the result follows from the general treatment in Section 3.

Let us now return to the question deciding between the cardinal and the ordinal methods of data elicitation. Suppose that we believe the Gaussian-noise models to be reasonably correct, and the per-observation errors σ and σ_c under the two settings are known or can be separately measured. Proposition 5 shows that the scaling of the minimax error in the cardinal and ordinal settings is identical in terms of the problem parameters n and d. As an important consequence, our result thus allows for the choice to be made based *only* on the parameters (σ, σ_c, B), and independent of n and d: the ordinal approach incurs a lower minimax error when $b_u(\sigma, B)\sigma^2 < \sigma_c^2$ while the cardinal approach is better off in terms of minimax error whenever $b_\ell(\sigma, B)\sigma^2 > \sigma_c^2$. Establishing the exact decision boundary would require tightening the constants in the bounds, a task we leave for future work.

5.3 Aggregate estimation error in experiments on MTurk

For the sake of completeness, we also computed the estimation error in the cardinal and ordinal settings. We consider data from the three experiments (c), (d) and (e).[2] We normalize the true vector to have $\|w^*\|_\infty = 1$ and set $B = 1$. For each of the three experiments, we execute 100 iterations of the following procedure. Select five workers from the cardinal and five from the ordinal pool of workers uniformly at random. (The number five is inspired by practical systems as in Wang et al. (2011); Piech et al. (2013).) We run the maximum-likelihood estimator of the Cardinal model on the data from the five workers selected from the cardinal pool, and the maximum-likelihood estimator of the Thurstone model on the data from the five workers of the ordinal pool. Note that unlike Section 5.1, the cardinal data here is *not* converted to ordinal.

Task	Spelling	Distance	Audio
$\frac{\|w^* - \widehat{w}\|_2^2}{d}$ in Ordinal	0.358 ± 0.035	**0.168 ± 0.026**	**0.444 ± 0.055**
$\frac{\|w^* - \widehat{w}\|_2^2}{d}$ in Cardinal	**0.350 ± 0.045**	0.330 ± 0.028	0.508 ± 0.053
Kendall's tau coefficient in Ordinal	**0.277 ± 0.049**	**0.547 ± 0.034**	**0.513 ± 0.047**
Kendall's tau coefficient in Cardinal	0.129 ± 0.046	0.085 ± 0.038	0.304 ± 0.049

Table 2: Evaluation of the inferred solution from the data received from multiple workers (mean and standard deviation shown).

The results are tabulated in Table 2. To put the results in perspective of the rest of the paper, let us also recall the per-sample errors in these experiments from Table 1. Observe that among these three experiments, the per-sample noise in the cardinal data was closest to that in the ordinal data in the experiment on identifying the number of spelling mistakes. The gap was larger in the two remaining experiments. This fact is reflected in

2. We restrict attention to these three experiments for the following reasons. There is no ground truth for experiments (f) and (g). In experiment (a), the size of each circle in each question is chosen independently from a continuous distribution, making all questions different and preventing aggregation. Experiment (b) employs a disconnected topology.

the results of Table 2 where the estimator on the cardinal data incurs a lower ℓ_2-error than the estimator on the ordinal data in the experiment on identifying the number of spelling mistakes, whereas the outcome goes the other way in the two remaining experiments. Our theory needs to tighten the constants in order to address this regime. With respect to the Kendall's tau correlation coefficient—a particular type of ordinal metric—the estimator on the ordinal data consistently gives a higher accuracy as compared to the cardinal case.

6. Conclusions

In this paper, we presented topology-aware minimax error bounds under a broad class of preference-elicitation models. We demonstrated the utility of these results in guiding the selection of comparisons and in guiding the choice of the elicitation paradigm (cardinal versus ordinal) when these options are available. A direction for future work would be to characterize the precise thresholds for making the choice between the cardinal and ordinal approaches. Secondly, the Thurstone and BTL models are parametric idealizations that have proved useful in a wide variety of applications. In future work we would like to investigate more flexible semi-parametric and non-parametric pairwise comparison models (see, for instance, Chatterjee (2014); Braverman and Mossel (2008)).

Acknowledgments

The authors are grateful to the Action Editor Moritz Hardt and anonymous referees for their suggestions that helped improve the manuscript.
This work was partially supported by Office of Naval Research MURI grant N00014-11-1-0688, Air Force Office of Scientific Research Grant AFOSR-FA9550-14-1-0016, and National Science Foundation Grants CIF-31712-23800, DMS-1107000 and CIF-81652-23800. The work of NBS was also partially supported by a Microsoft Research PhD fellowship.

Appendix A. Proof of Theorem 1

The following two sections prove the lower and upper bounds (respectively) on the minimax risk of Ordinal model under the squared L semi-norm.

A.1 Lower bound

Our lower bounds are based on the Fano argument, which is a standard method in minimax analysis (see for instance Tsybakov (2008)). Suppose that our goal is to bound the minimax risk of estimating a parameter w over an indexed class of distributions $\mathcal{P} = \{\mathbb{P}_w \mid w \in \mathcal{W}\}$ in the square of a pseudo-metric ρ. Consider a collection of vectors $\{w^1, \ldots, w^M\}$ contained within $\mathcal{W}$ such that

$$\min_{\substack{j,k\in[M]\\ j\neq k}} \rho\big(w^j, w^k\big) \geq \delta \quad \text{and} \quad \frac{1}{\binom{M}{2}} \sum_{\substack{j,k\in[M]\\ j\neq k}} D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq \beta.$$

We refer to any such subset as an (δ, β)-packing set.

Lemma 6 (Pairwise Fano minimax lower bound) *Suppose that we can construct a (δ, β)-packing with cardinality M. Then the minimax risk is lower bounded as*

$$\inf_{\widehat{w}} \sup_{w^* \in \mathcal{W}} \mathbb{E}\Big[\rho(\widehat{w}, w^*)^2\Big] \geq \frac{\delta^2}{2}\Big(1 - \frac{\beta + \log 2}{\log M}\Big). \tag{13}$$

The main difficulty in deriving the lower bound is the construction of a suitable packing set for application in Lemma 6. To this end, given a scalar $\alpha \in (0, \frac{1}{4})$ whose value will be specified later, define the integer

$$M(\alpha) := \left\lfloor \exp\Big\{\frac{d}{2}\big(\log 2 + 2\alpha \log 2\alpha + (1-2\alpha)\log(1-2\alpha)\big)\Big\}\right\rfloor. \tag{14}$$

The following two lemmas aid in our construction of a packing set. The first lemma is a straightforward consequence of the Gilbert-Varshamov bound (Gilbert, 1952; Varshamov, 1957).

Lemma 7 *For any $\alpha \in (0, \frac{1}{4})$, there exists a set of $M(\alpha)$ binary vectors $\{z^1, \ldots, z^{M(\alpha)}\} \subset \{0,1\}^d$ such that*

$$\alpha d \leq \|z^j - z^k\|_2^2 \leq d \qquad \text{for all } j \neq k \in [M(\alpha)], \text{ and} \tag{15a}$$

$$\langle e_1, z^j \rangle = 0 \qquad \text{for all } j \in [M(\alpha)], \tag{15b}$$

where e_1 denotes the first canonical basis vector.

The next lemma derives an upper bound on the Kullback-Leibler divergence between the probability distributions induced by any pair of quality score vectors.

Lemma 8 *For any pair of quality score vectors w^j and w^k, and for*

$$\zeta := \frac{\max\limits_{x\in[0,2B/\sigma]} F'(x)}{F(2B/\sigma)(1 - F(2B/\sigma))},$$

we have

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq \frac{n\zeta}{\sigma^2}(w^j - w^k)^T L (w^j - w^k). \tag{16}$$

We prove these two lemmas at the end of this section.

Taking these two lemmas as given for the moment, consider the set $\{z^1, \ldots, z^{M(\alpha)}\}$ of d-dimensional binary vectors given by Lemma 7. The Laplacian L of the comparison graph is symmetric and positive-semidefinite, and so has a diagonalization of the form $L = U^T \Lambda U$ where $U \in \mathbb{R}^{d\times d}$ is an orthonormal matrix, and Λ is a diagonal matrix of nonnegative eigenvalues.

Letting matrix $\Lambda^\dagger$ denote the Moore-Penrose pseudo-inverse of Λ, consider the collection $\{w^1, \ldots, w^{M(\alpha)}\}$ of vectors given by $w^j := \frac{\delta}{\sqrt{d}} U^T \sqrt{\Lambda^\dagger} z^j$ for each $j \in [M(\alpha)]$. Since $1 \in$

nullspace(L), we are guaranteed that $\langle 1,\, w^j\rangle = \frac{\delta}{\sqrt{d}} 1^T U^T \sqrt{\Lambda^\dagger} z^j \;= 0$. On the other hand,

$$\begin{aligned}(w^j - w^k)^T L (w^j - w^k) &\le \frac{\delta^2}{d}(z^j - z^k)^T \sqrt{\Lambda^\dagger} U L U^T \sqrt{\Lambda^\dagger}(z^j - z^k)\\ &= \frac{\delta^2}{d}(z^j - z^k)\sqrt{\Lambda^\dagger}\Lambda\sqrt{\Lambda^\dagger}(z^j - z^k)\\ &= \frac{\delta^2}{d}\|z^j - z^k\|_2^2,\end{aligned}$$

Here the last step makes use of the fact that the first coordinate of each vector z^j and z^k is zero. It follows that $\alpha\delta^2 \le \|w^j - w^k\|_L^2 \le \delta^2$.

Setting $\delta^2 := 0.01\frac{\sigma^2 d}{n\zeta}$, we find that

$$\|w^j\|_\infty \;\le\; \frac{\delta}{\sqrt{d}}\|\sqrt{\Lambda^\dagger} z^j\|_2 \;\overset{(i)}{\le}\; \frac{\delta}{\sqrt{d}}\sqrt{\mathrm{tr}(\Lambda^\dagger)} \;\overset{(ii)}{=}\; \frac{\delta}{\sqrt{d}}\sqrt{\mathrm{tr}(L^\dagger)} \;\overset{(iii)}{\le}\; B,$$

where inequality (i) follows from the fact that z^j has entries in $\{0, 1\}$; equation (ii) follows since $L^\dagger = U^T\Lambda^\dagger U$ by definition; and inequality (iii) follows from our choice of δ and our assumption $n \ge \frac{c\sigma^2 \mathrm{tr}(L^\dagger)}{\zeta B^2}$ on the sample size with $c = 0.01$. We have thus verified that each vector w^j also satisfies the boundedness constraint $\|w^j\|_\infty \le B$ required for membership in $\mathcal{W}_B$. Finally, observe that

$$\max_{j\neq k} D_{\mathrm{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) \le \frac{n\zeta\delta^2}{\sigma^2}, \quad \text{and} \quad \min_{j\neq k}\|w^j - w^k\|_L^2 \ge \alpha\delta^2.$$

We have thus constructed a suitable packing set for applying Lemma 6, which yields the lower bound

$$\mathbb{E}[\|\widehat{w} - w^*\|_L^2] \ge \frac{\alpha}{2}\delta^2\Big\{1 - \frac{\frac{\delta^2\zeta n}{\sigma^2} + \log 2}{\log M(\alpha)}\Big\}.$$

Substituting our choice of δ and setting $\alpha = 0.01$ proves the claim for $d > 9$.

In order to handle the case $d \le 9$, we consider the set of the three d-length vectors given by $z^1 = [0 \;\; \cdots \;\; 0 \;\; -1]$, $z^2 = [0 \;\; \cdots \;\; 0 \;\; 1]$ and $z^3 = [0 \;\; \cdots \;\; 0 \;\; 0]$. Construct the packing set $\{w^1, w^2, w^3\}$ from these three vectors $\{z^1, z^2, z^3\}$ as done above for the case of $d > 9$. From the calculations made for the general case above, we have for all pairs $\min_{j\neq k}\|w^j - w^k\|_L^2 \ge \frac{\delta^2}{9}$ and $\max_{j,k}\|w^j - w^k\|_L^2 \le 4\delta^2$, and as a result $\max_{j,k} D_{\mathrm{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) \le \frac{4n\zeta\delta^2}{\sigma^2}$. Choosing $\delta^2 = \frac{\sigma^2\log 2}{8n\zeta}$ and applying Lemma 6 proves the theorem.

The only remaining detail is to prove and Lemma 7 and Lemma 8.

Proof of Lemma 7: The Gilbert-Varshamov bound Gilbert (1952); Varshamov (1957) guarantees the existence of a binary code $\{\widetilde{z}^1, \ldots, \widetilde{z}^N\}$ in dimension $(d-1)$, minimum Hamming distance $\lceil \alpha d\rceil$, and the number of code words N at least

$$N \ge \frac{2^{d-1}}{\sum_{\ell=0}^{\lceil\alpha d\rceil - 1}\binom{d-1}{\ell}}.$$

Since $d \ge 2$ and $\alpha \in (0, \frac{1}{4})$, we have

$$\frac{\lceil \alpha d\rceil - 1}{d-1} \le 2\alpha \le \frac{1}{2}.$$

Applying standard bounds on the tail of the binomial distribution gives

$$\begin{aligned}\frac{1}{2^{d-1}}\sum_{\ell=0}^{\lceil\alpha d\rceil - 1}\binom{d-1}{\ell} &\le \exp\Big(-(d-1)D_{\mathrm{KL}}(\tfrac{\lceil\alpha d\rceil - 1}{d-1}\|\tfrac{1}{2})\Big)\\ &\le \exp\Big(-(d-1)D_{\mathrm{KL}}(2\alpha\|\tfrac{1}{2})\Big),\end{aligned}$$

and hence $N \ge M(\alpha)$.

We now define the set of vectors $\{z^1, \ldots, z^{M(\alpha)}\}$ as $(z^i)^T = [0 \;\; (\widetilde{z}^i)^T]$ for every $i \in [M(\alpha)]$. Given this condition, it is easy to see that $\langle e_1,\, z^j\rangle = 0$ for every vector z^j in this set. Finally, the minimum distance condition gives the desired constraints on the difference between any pair of vectors in this set under the squared ℓ_2 metric.

Proof of Lemma 8: For any pair of quality score vectors w^j and w^k, the KL divergence between the distributions $\mathbb{P}_{w^j}$ and $\mathbb{P}_{w^k}$ is given by

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) = \sum_{i=1}^n F(\langle w^j,\, x_i\rangle/\sigma)\log\frac{F(\langle w^j,\, x_i\rangle/\sigma)}{F(\langle w^k,\, x_i\rangle/\sigma)} + (1 - F(\langle w^j,\, x_i\rangle/\sigma))\log\frac{1 - F(\langle w^j,\, x_i\rangle/\sigma)}{1 - F(\langle w^k,\, x_i\rangle/\sigma)}.$$

For any $a, b \in (0, 1)$, we have the elementary inequality $a\log\frac{a}{b} \le (a-b)\frac{a}{b}$. Applying this inequality to our expression above gives

$$\begin{aligned}D_{\mathrm{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) &\le \sum_{i=1}^n (F(\langle w^j,\, x_i\rangle/\sigma) - F(\langle w^k,\, x_i\rangle/\sigma))\frac{F(\langle w^j,\, x_i\rangle/\sigma)}{F(\langle w^k,\, x_i\rangle/\sigma)}\\ &\qquad - \Big\{F(\langle w^j,\, x_i\rangle/\sigma)) - F(\langle w^k,\, x_i\rangle/\sigma)\Big\}\frac{1 - F(\langle w^j,\, x_i\rangle/\sigma)}{1 - F(\langle w^k,\, x_i\rangle/\sigma)}\\ &\le \sum_{i=1}^n \frac{(F(\langle w^j,\, x_i\rangle/\sigma) - F(\langle w^k,\, x_i\rangle/\sigma))^2}{F(\langle w^k,\, x_i\rangle/\sigma)(1 - F(\langle w^k,\, x_i\rangle/\sigma))}.\end{aligned}$$

Since $\max\{\|w^j\|_\infty, \|w^k\|_\infty\} \le B$, and since F is a non-decreasing function, we have

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) \le \sum_{i=1}^n \frac{(F(\langle w^j,\, x_i\rangle/\sigma) - F(\langle w^k,\, x_i\rangle/\sigma))^2}{F(2B/\sigma)(1 - F(2B/\sigma))}.$$

Finally, applying the mean value theorem and recalling the definition of ζ (from (6)) yields

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) \le \sum_{i=1}^n \zeta(\langle w^j,\, x_i\rangle/\sigma - \langle w^k,\, x_i\rangle/\sigma)^2 \;=\; \frac{n\zeta}{\sigma^2}(w^j - w^k)^T L (w^j - w^k),$$

as claimed.

A.2 Upper bound

For the Ordinal model, the MLE is given by $\widehat{w} \in \arg\min_{w \in \mathcal{W}_B} \ell(w)$, where

$$\ell(w) = -\frac{1}{n}\sum_{i=1}^{n}\Big\{\mathbf{1}[y_i = 1]\log F\big(\frac{\langle x_i,\, w\rangle}{\sigma}\big) + \mathbf{1}[y_i = -1]\log\Big(1 - F\big(\frac{\langle x_i,\, w\rangle}{\sigma}\big)\Big)\Big\}, \qquad \text{and} \tag{17a}$$

$$\mathcal{W}_B := \big\{w \in \mathbb{R}^d \mid \langle 1,\, w\rangle = 0, \quad \text{and} \quad \|w\|_\infty \le B\big\}. \tag{17b}$$

Our goal is to bound the estimation error of the MLE in the squared semi-norm $\|v\|_L^2 = v^T L v$.

For the purposes of this proof (as well as subsequent ones), let us state and prove an auxiliary lemma that applies more generally to M-estimators that are based on minimizing an arbitrary convex and differentiable function over some subset $\mathcal{W}$ of the set $\mathcal{W}_\infty := \{w \in \mathbb{R}^d \mid \langle 1,\, w\rangle = 0\}$. The MLE under consideration here is a special case. This lemma requires that ℓ is differentiable and strongly convex at w^* with respect to the semi-norm $\|\cdot\|_L$, meaning that there is some constant $\kappa > 0$ such that

$$\ell(w^* + \Delta) - \ell(w^*) - \langle \nabla\ell(w^*),\, \Delta\rangle \ge \kappa\|\Delta\|_L^2 \tag{18}$$

for all perturbations $\Delta \in \mathbb{R}^d$ such that $(w^* + \Delta) \in \mathcal{W}$. Finally, it is also convenient to introduce the semi-norm $\|u\|_{L^\dagger} = \sqrt{u^T L^\dagger u}$, where $L^\dagger$ is the Moore-Penrose pseudo-inverse of L.

Lemma 9 (Upper bound for M-estimators) *Consider the M-estimator*

$$\widehat{w} \in \arg\min_{w \in \mathcal{W}} \ell(w), \qquad \textit{where } \mathcal{W} \textit{ is any subset of } \mathcal{W}_\infty, \tag{19}$$

and ℓ is a differentiable cost function satisfying the κ-strong convexity condition (18) *at some $w^* \in \mathcal{W}$. Then*

$$\|\widehat{w} - w^*\|_L \le \frac{1}{\kappa}\|\nabla\ell(w^*)\|_{L^\dagger}. \tag{20}$$

Proof Since $\widehat{w}$ and w^* are optimal and feasible, respectively, for the original optimization problem, we have $\ell(\widehat{w}) \le \ell(w^*)$. Defining the error vector $\Delta = \widehat{w} - w^*$, adding and subtracting the quantity $\langle \nabla\ell(w^*),\, \Delta\rangle$ yields the bound

$$\ell(w^* + \Delta) - \ell(w^*) - \langle \nabla\ell(w^*),\, \Delta\rangle \le -\langle \nabla\ell(w^*),\, \Delta\rangle.$$

By the κ-convexity condition, the left-hand side is lower bounded by $\kappa\|\Delta\|_L^2$. As for the right-hand side, note that Δ satisfies the constraint $\langle 1,\, \Delta\rangle = 0$, and thus is orthogonal to the nullspace of the Laplacian matrix L. Therefore, by Lemma 16 (in Appendix F), we have $|\langle \nabla\ell(w^*),\, \Delta\rangle| \le \|\nabla\ell(w^*)\|_{L^\dagger}\,\|\Delta\|_L$. Combining the pieces yields the claimed inequality (20). ■

In order to apply Lemma 9 to the MLE for the Ordinal model, we need to verify that the negative log likelihood (17a) satisfies the strong convexity condition, and we need to bound the random variable $\|\nabla\ell(w^*)\|_{L^\dagger}$ defined in the dual norm $\|\cdot\|_{L^\dagger}$.

Verifying strong convexity: By chain rule, the Hessian of ℓ is given by

$$\nabla^2\ell(w) = \frac{1}{n\sigma^2}\sum_{i=1}^{n}\Big\{\mathbf{1}[y_i = 1]T_{i1} + \mathbf{1}[y_i = -1]T_{i2}\Big\}\, x_i x_i^T,$$

where

$$T_{i1} := \frac{F'(\frac{\langle w, x_i\rangle}{\sigma})^2 - F(\frac{\langle w, x_i\rangle}{\sigma})F''(\frac{\langle w, x_i\rangle}{\sigma})}{F(\frac{\langle w, x_i\rangle}{\sigma})^2}, \quad \text{and} \quad T_{i2} := \frac{F'(\frac{\langle w, x_i\rangle}{\sigma})^2 + (1 - F(\frac{\langle w, x_i\rangle}{\sigma}))F''(\frac{\langle w, x_i\rangle}{\sigma})}{(1 - F(\frac{\langle w, x_i\rangle}{\sigma}))^2}.$$

Observe that the term T_{i1} is simply the second derivative of $\log F$ evaluated at $\frac{\langle w, x_i\rangle}{\sigma}$, and hence the strong log-concavity of F implies $T_{i1} \ge \gamma$. On the other hand, the term T_{i2} is the second derivative of $\log(1 - F)$. Since $F(-x) = 1 - F(x)$ for all x, it follows that the function $x \mapsto 1 - F(x)$ is also strongly log-concave with parameter γ and hence $T_{i2} \ge \gamma$. Putting together the pieces, we conclude that

$$v^T\nabla^2\ell(w)v \ge \frac{\gamma}{n\sigma^2}\|Xv\|_2^2 \qquad \text{for all } v, w \in \mathcal{W}_B,$$

where $X \in \mathbb{R}^{n\times d}$ has the differencing vector $x_i \in \mathbb{R}^d$ as its i^{th} row.

Thus, if we introduce the error vector $\Delta := \widehat{w} - w^*$, then we may conclude that

$$\ell(w^* + \Delta) - \ell(w^*) - \langle \nabla\ell(w^*),\, \Delta\rangle \ge \frac{\gamma}{n\sigma^2}\|X\Delta\|_2^2 \;=\; \frac{\gamma}{\sigma^2}\|\Delta\|_L^2,$$

showing that ℓ is strongly convex around w^* with parameter $\kappa = \frac{\gamma}{\sigma^2}$. An application of Lemma 9 then gives $\|\Delta\|_L^2 \le \frac{\sigma^4}{\gamma^2}\|\nabla\ell(w^*)\|_{L^\dagger}^2$.

Bounding the dual norm: In order to obtain a concrete bound, it remains to control the quantity $\nabla\ell(w^*)^T L^\dagger \nabla\ell(w^*)$. Observe that the gradient takes the form

$$\nabla\ell(w^*) = \frac{-1}{n\sigma}\sum_{i=1}^{n}\big[\mathbf{1}[y_i = 1]\frac{F'(\langle w^*,\, x_i\rangle/\sigma)}{F(\langle w^*,\, x_i\rangle/\sigma)} - \mathbf{1}[y_i = -1]\frac{F'(\langle w^*,\, x_i\rangle/\sigma)}{1 - F(\langle w^*,\, x_i\rangle/\sigma)}\big]x_i.$$

Define a random vector $V \in \mathbb{R}^n$ with independent components as

$$V_i = \begin{cases} \frac{F'(\langle w^*, x_i\rangle/\sigma)}{F(\langle w^*, x_i\rangle/\sigma)} & \text{w.p.} \quad F(\langle w^*,\, x_i\rangle/\sigma) \\ \frac{-F'(\langle w^*, x_i\rangle/\sigma)}{1-F(\langle w^*, x_i\rangle/\sigma)} & \text{w.p.} \quad 1 - F(\langle w^*,\, x_i\rangle/\sigma). \end{cases}$$

With this notation, we have $\nabla\ell(w^*) = -\frac{1}{n\sigma}X^T V$. One can verify that $\mathbb{E}[V] = 0$ and

$$|V_i| \le \sup_{z\in[-2B/\sigma, 2B/\sigma]} \max\Big\{\frac{F'(z)}{F(z)}, \frac{F'(z)}{1 - F(z)}\Big\} \le \sup_{z\in[-2B/\sigma, 2B/\sigma]} \frac{F'(z)}{F(z)(1 - F(z))} \le \zeta, \tag{21}$$

where ζ is as defined in (6). Defining the n-dimensional square matrix $M := \frac{\sigma^2}{\gamma^2 n^2}XL^\dagger X^T$, our definitions and previous bounds imply that $\|\Delta\|_L^2 \le V^T M V$.

Consequently, our problem has been reduced to controlling the fluctuations of the quadratic form $V^T M V$; in order to do so, we apply the Hanson-Wright inequality (see Lemma 13 in Appendix E). A straightforward calculation yields

$$\|M\|_{\text{fro}}^2 = (d-1)\frac{\sigma^4}{\gamma^4 n^2} \quad \text{and} \quad \|M\|_{\text{op}} = \frac{\sigma^2}{\gamma^2 n},$$

where we have used the fact that $L = \frac{1}{n} X^T X$. Moreover, since the components of V are independent and of zero mean, a straightforward calculation yields that $\mathbb{E}[V^T M V] \leq \mathbb{E}[\|V\|_\infty^2 \text{tr}(M)] \leq \frac{\zeta^2 \sigma^2 d}{\gamma^2 n}$.

Since $|V_i| \leq \zeta$, the variables are ζ-sub-Gaussian, and hence the Hanson-Wright inequality implies that

$$\mathbb{P}\Big[V^T M V - \frac{\zeta^2 \sigma^2 d}{\gamma^2 n} > t\Big] \leq 2\exp\big(-c \min\{\frac{t^2 \gamma^4 n^2}{\zeta^4 (d-1)\sigma^4}, \frac{t\gamma^2 n}{\zeta^2 \sigma^2}\}\big) \quad \text{for all } t > 0.$$

Consequently, after some simple algebra, we conclude that

$$\mathbb{P}\Big(\|\Delta\|_L^2 > t \frac{c\zeta^2\sigma^2}{\gamma^2}\frac{d}{n}\Big) \leq e^{-t} \quad \text{for all } t \geq 1,$$

for some universal constant c. Integrating this tail bound yields the bound on the expectation.

Appendix B. Proof of Theorem 2

The following two sections prove the upper and lower bounds (respectively) on the minimax risk in the squared Euclidean norm for Ordinal model. We prove the lower bound in two parts corresponding to the two components of the "max" in the statement of the theorem.

B.1 Upper bound

The proof of the upper bound under the Euclidean norm follows directly from the upper bound under the L semi-norm proved in Theorem 1. From the setting described in Section 2, we have that the nullspace of the matrix L is given by the span of the all ones vector. Furthermore, we have $\langle w^* - \widehat{w}, 1\rangle = 0$, and $\|w^* - \widehat{w}\|_L^2 \geq \lambda_2(L)\|w^* - \widehat{w}\|_2^2$. Substituting this inequality into the upper bound (7b) gives the desired result.

B.2 Lower bound: Part I

Since the Laplacian L of the comparison graph is symmetric and positive-semidefinite. By diagonalization, we can write $L = U^T \Lambda U$ where $U \in \mathbb{R}^{d\times d}$ is an orthonormal matrix, and Λ is a diagonal matrix of nonnegative eigenvalues with $\Lambda_{jj} = \lambda_j(L)$.

We first use the Fano method (Lemma 6) to prove that the minimax risk is lower bounded as $c\sigma^2 \frac{d^2}{n}$. For scalars $\alpha \in (0, \frac{1}{4})$ and $\delta > 0$ whose values will be specified later, recall the set $\{z^1, \ldots, z^{M(\alpha)}\}$ of vectors in the Boolean hypercube $\{0,1\}^d$ given by Lemma 7. We then define a second set $\{w^j, j \in [M(\alpha)]\}$ via $w^j := \frac{\delta}{\sqrt{d}} U^T P z^j$, where P is a permutation matrix to be specified momentarily. At this point, the only constraint imposed on P is

that it keeps the first coordinate constant. By construction, for each $j \neq k$, we have $\|w^j - w^k\|_2^2 = \frac{\delta^2}{d}\|z^j - z^k\|_2^2 \geq \alpha\delta^2$, where the final inequality follows from the fact that the set $\{z^1, \ldots, z^{M(\alpha)}\}$ comprises binary vectors with a minimum Hamming distance at least αd.

Consider any distinct $j, k \in [M(\alpha)]$. Then, for some $\{i_1, \ldots, i_r\} \subseteq \{2, \ldots, d\}$ with $\alpha d \leq r \leq d$, it must be that

$$\|w^j - w^k\|_L^2 = \frac{\delta^2}{d}\|U^T P z^j - U^T P z^k\|_L^2 = \frac{\delta^2}{d}\|z^j - z^k\|_\Lambda^2 = \frac{\delta^2}{d}\sum_{m=1}^{r} \lambda_{i_m}(L).$$

It follows that for some non-negative numbers $a_2, \ldots, a_d$ such that $\alpha d \leq \sum_{i=2}^d a_i \leq d$,

$$\frac{1}{\binom{M(\alpha)}{2}}\sum_{j\neq k}\|w^j - w^k\|_L^2 = \frac{\delta^2}{d}\sum_{i=2}^{d} a_i \lambda_i(L).$$

We choose the permutation matrix P such that the last $(d-1)$ coordinates are permuted to have $a_2 \geq \cdots \geq a_d$ and the d^{th} coordinate remains fixed. With this choice, we get

$$\frac{1}{\binom{M(\alpha)}{2}}\sum_{j\neq k}\|w^j - w^k\|_L^2 \leq \frac{\delta^2}{d}\frac{d}{d-1}\text{tr}(L) \leq \frac{2\delta^2}{d}\text{tr}(L).$$

Lemma (14) (Appendix F) gives the trace constraint $\text{tr}(L) = 2$, which in turn guarantees that $\frac{1}{\binom{M(\alpha)}{2}}\sum_{j\neq k}\|w^j - w^k\|_L^2 \leq \frac{4\delta^2}{d}$. For the choice of P specified above, we have for every $j \in [M(\alpha)]$,

$$\langle 1, w^j\rangle = \frac{\delta}{\sqrt{d}} e_1^T P z^j = e_1^T z^j = 0,$$

where the final equation employed the property (15b).

Setting $\delta^2 = 0.01\frac{\sigma^2 d^2}{4n\zeta}$, we have $\|w^j\|_\infty \leq \frac{\delta}{\sqrt{d}}\|z^j\|_2 \overset{(i)}{\leq} \delta \overset{(ii)}{\leq} B$, where inequality (i) follows from the fact that z^j has entries in $\{0,1\}$; inequality (ii) follows from our choice of δ and our assumption $n \geq \frac{c\sigma^2 \text{tr}(L^\dagger)}{\zeta B^2}$ on the sample size with $c = 0.002$, where Lemma 14 guarantees $n \geq \frac{c\sigma^2 d^2}{4\zeta B^2}$. We have thus verified that each vector w^j also satisfies the boundedness constraint $\|w^j\|_\infty \leq B$ required for membership in $\mathcal{W}_B$.

From the proof of Theorem 1, we have that for any distinct $D_{\text{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) \leq \frac{n\zeta}{\sigma^2}\|w^j - w^k\|_L^2$, and hence

$$\frac{1}{\binom{M(\alpha)}{2}}\sum_{j\neq k} D_{\text{KL}}(\mathbb{P}_{w^j}\|\mathbb{P}_{w^k}) \leq \frac{n\zeta}{\sigma^2}\frac{4\delta^2}{d} = 0.01\, d,$$

where we have substituted our previous choice of δ.

Applying Lemma 6 with the packing set $\{w^1, \ldots, w^{M(\alpha)}\}$ gives that any estimator $\widetilde{w}$ must incur an error lower bounded as

$$\sup_{w^* \in \mathcal{W}_B} \mathbb{E}\big[\|\widetilde{w} - w^*\|_2^2\big] \geq \frac{\alpha\delta^2}{2}\big(1 - \frac{0.01d + \log 2}{\log M(\alpha)}\big).$$

Substituting our choice of δ and setting $\alpha = 0.01$ proves the claim for $d > 9$.

For the case of $d \le 9$, consider the set of the three d-length vectors $z^1 = [0 \ \cdots \ 0 \ -1]$, $z^2 = [0 \ \cdots \ 0 \ 1]$ and $z^3 = [0 \ \cdots \ 0 \ 0]$. Construct the packing set $\{w^1, w^2, w^3\}$ from these three vectors $\{z^1, z^2, z^3\}$ as done above for the case of $d > 9$. From the calculations made for the general case above, we have for all pairs $\min_{j\neq k} \|w^j - w^k\|_2^2 \ge \frac{\delta^2}{9}$ and $\max_{j,k} \|w^j - w^k\|_L^2 \le 4\delta^2$, and as a result $\max_{j,k} D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \le \frac{4n\zeta\delta^2}{\sigma^2}$. Choosing $\delta^2 = \frac{\sigma^2 \log 2}{8n\zeta}$ and applying Lemma 6 yields the claim.

B.3 Lower bound: Part II

Given an integer $d' \in \{2, \ldots, d\}$, and scalars $\alpha \in (0, \frac{1}{4})$ and $\delta > 0$, define the integer

$$M'(\alpha) := \left\lfloor \exp\left\{ \frac{d'}{2}\big(\log 2 + 2\alpha \log 2\alpha + (1-2\alpha)\log(1-2\alpha) \big) \right\} \right\rfloor. \tag{22}$$

Applying Lemma 7 with d' as the dimension yields a subset $\{z^1, \ldots, z^{M'(\alpha)}\}$ of the Boolean hypercube $\{0,1\}^{d'}$ with the stated properties. We then define a set of d-length vectors $\{\widetilde{w}^1, \ldots, \widetilde{w}^{M'(\alpha)}\}$ via

$$\widetilde{w}^j = [0 \ (z^j)^T \ 0 \ \cdots \ 0]^T \quad \text{for each } j \in [M(\alpha)].$$

For each $j \in [M(\alpha)]$, let us define $w^j := \frac{\delta}{\sqrt{d'}} U^T \sqrt{\Lambda^\dagger}\widetilde{w}^j$. Now, letting $e_1 \in \mathbb{R}^d$ denote the first standard basis vector, we have $\langle 1, w^j \rangle = \frac{\delta}{\sqrt{d'}} 1^T U^T \sqrt{\Lambda^\dagger}\widetilde{w}^j \ = 0$. where we have used the fact that $1 \in \mathrm{nullspace}(L)$. Furthermore, for any $j \neq k$, we have

$$\|w^j - w^k\|_2^2 = \frac{\delta^2}{d'}(\widetilde{w}^j - \widetilde{w}^k)^T \Lambda^\dagger (\widetilde{w}^j - \widetilde{w}^k) \ge \frac{\delta^2}{d'} \sum_{i=\lceil (1-\alpha)d' \rceil}^{d'} \frac{1}{\lambda_i}.$$

Thus, setting $\delta^2 = 0.01 \frac{\sigma^2 d'}{n\zeta}$ yields

$$\|w^j\|_\infty \ \le \ \frac{\delta}{\sqrt{d'}} \|\sqrt{\Lambda^\dagger}\widetilde{w}^j\|_2 \ \overset{(i)}{\le} \ \frac{\delta}{\sqrt{d'}} \sqrt{\mathrm{tr}(\Lambda^\dagger)} \ \overset{(ii)}{=} \ \frac{\delta}{\sqrt{d'}} \sqrt{\mathrm{tr}(L^\dagger)} \ \overset{(iii)}{\le} \ B,$$

where inequality (i) follows from the fact that z^j has entries in $\{0,1\}$; step (ii) follows because the matrices $\sqrt{\Lambda^\dagger}$ and $\sqrt{L^\dagger}$ have the same eigenvalues; and inequality (iii) follows from our choice of δ and our assumption $n \ge \frac{c\sigma^2 \mathrm{tr}(L^\dagger)}{\zeta B^2}$ on the sample size with $c = 0.01$. We have thus verified that each vector w^j also satisfies the boundedness constraint $\|w^j\|_\infty \le B$ required for membership in $\mathcal{W}_B$. Furthermore, for any pair of distinct vectors in this set, we have

$$\|w^j - w^k\|_L^2 = \frac{\delta^2}{d'} \|z^j - z^k\|_2^2 \le \delta^2.$$

From the proof of Theorem 1, we $D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \le \frac{n\zeta}{\sigma^2} \|w^j - w^k\|_L^2 \le 0.01 d'$. Applying Lemma 6 with the packing set $\{w^1, \ldots, w^{M'(\alpha)}\}$ gives that any estimator $\widetilde{w}$ must incur an error lower bounded as

$$\sup_{w^* \in \mathcal{W}_B} \mathbb{E}\big[\|\widetilde{w} - w^*\|_2^2\big] \ge \frac{\frac{\delta^2}{d'} \sum_{i=\lceil (1-\alpha)d' \rceil}^{d'} \frac{1}{\lambda_i}}{2} \big(1 - \frac{0.01 d' + \log 2}{\log M'(\alpha)}\big).$$

Substituting our choice of δ and setting $\alpha = 0.01$ proves the claim for $d' > 9$.

For the case of $d' \le 9$, we now prove a lower bound of $\frac{c\sigma^2}{n} \frac{9}{\lambda_2(L)}$ for a universal constant $c > 0$. This quantity is at least as large as the claimed lower bound. Consider the packing set of three d-length vectors $w^1 = \delta U \sqrt{\Lambda^\dagger}[0 \ 1 \ 0 \ \cdots \ 0]^T$, $w^2 = -w^1$ and $w^3 = [0 \ \cdots \ 0]^T$ for some $\delta > 0$. Then for every $j \neq k$, one can verify that $\|w^j - w^k\|_L^2 \le 4\delta^2$, $\|w^j - w^k\|_2^2 \ge \frac{\delta^2}{\lambda_2(L)}$. Choosing $\delta^2 = \frac{\sigma^2 \log 2}{8n\zeta}$ and applying Lemma 6 proves the claim for $d' \le 9$.

Finally, taking the maximum over all values of $d' \in \{2, \ldots, d\}$ gives the claimed lower bound.

Appendix C. Proof of Theorem 3

We now turn to the proof of Theorem 3 on the minimax rate for the Paired Cardinal model. Recall that this observation model takes the standard linear model, $y = Xw^* + \epsilon$, where $y \in \mathbb{R}^n, w \in \mathbb{R}^d$ and $\epsilon \sim N(0, \sigma^2 I)$.

C.1 Upper bound under the squared L semi-norm

The maximum likelihood estimate in the Paired Cardinal model is a special case of the general M-estimator (19) with $\ell(w) := \frac{1}{2n} \sum_{i=1}^n \big(y_i - \langle x_i, w \rangle\big)^2$. For this quadratic objective function, it is easy to verify that the γ-convexity condition holds with $\gamma = 1$. (In particular, note that the Hessian of ℓ is given by $L = X^T X/n$.)

Given the result of Lemma 9, it remains to upper bound $\|\nabla \ell(w^*)\|_{L^\dagger}$. A straightforward computation yields $\|\nabla \ell(w^*)\|_{L^\dagger}^2 = \frac{\varepsilon}{\sigma}^T Q \frac{\varepsilon}{\sigma}$ where $Q := \frac{\sigma^2}{n^2} X L^\dagger X^T$. Consequently, the random variable $\|\nabla \ell(w^*)\|_{L^\dagger}^2$ is quadratic form in the standard Gaussian random vector $\frac{\varepsilon}{\sigma}$. An application of Lemma 15 (Appendix F) gives $\mathrm{tr}(Q) = \frac{\sigma^2}{n}\big(d-1\big)$ and $|||Q|||_{\mathrm{op}} = \frac{\sigma^2}{n}$, and then applying a known tail bound on Gaussian quadratic forms (see Lemma 12 in Appendix E) yields

$$\mathbb{P}\left[\frac{\|\nabla \ell(w^*)\|_{L^\dagger}^2}{\sigma^2} \ge \Big(\sqrt{\frac{d}{n}} + \frac{\delta}{\sqrt{n}} \Big)^2 \right] \le e^{-\frac{\delta^2}{2}} \qquad \text{for all } \delta \ge 0.$$

Since $d \ge 2$, we have $\big(\sigma\sqrt{\frac{d}{n}} + \frac{\sigma}{\sqrt{n}}\delta\big)^2 \le \frac{2\sigma^2 d\delta^2}{n}$ for all $\delta \ge 4$, which yields

$$\mathbb{P}\Big[\|\nabla \ell(w^*)\|_{L^\dagger}^2 \ge t \, \frac{4\sigma^2 d}{n} \Big] \le e^{-t} \qquad \text{for all } t \ge 8.$$

Integrating this tail bound yields that $\mathbb{E}\Big[\|\nabla \ell(w^*)\|_{L^\dagger}^2\Big] \le c\sigma^2 \frac{d}{n}$, from which the claim follows.

C.2 Lower bound under the squared L semi-norm

Based on the pairwise Fano lower bound previously stated in Lemma 6, we need to construct a suitable (δ, β)-packing, where the semi-norm $\rho(w^j, w^k) = \|w^j - w^k\|_L$ is defined by the Laplacian. Given the additive Gaussian noise observation model, we also have

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) = \frac{n}{2\sigma^2} \|w^j - w^k\|_L^2, \tag{23}$$

The construction of the packing and the remainder of the proof proceeds in a manner identical to the proof of the lower bound in Theorem 1, except for the absence of the requirement of $\|w^j\|_\infty \le B$ on the elements $\{w^j\}$ of the packing set.

C.3 Upper bound under the squared Euclidean norm

The upper bound follows by direct analysis of the (unconstrained) least-squares estimate, which has the explicit form $\widehat{w} = \frac{1}{n} L^\dagger X^T y$, and thus

$$\mathbb{E}\|\widehat{w} - w^*\|_2^2 = \mathbb{E}\|\frac{1}{n} L^\dagger X^T \epsilon\|_2^2 \;=\; \sigma^2 \mathrm{tr}(\frac{1}{n^2} L^\dagger X^T X L^\dagger)$$

where we have used the fact that $\epsilon \sim N(0, \sigma^2 I_n)$. Since $L = X^T X/n$ by definition, we conclude that $\mathbb{E}\|\widehat{w} - w^*\|_2^2 = \frac{\sigma^2 \mathrm{tr}(L^\dagger)}{n}$ as claimed.

C.4 Lower bound under the squared Euclidean norm

We obtain the lower bound by computing the Bayes risk with respect to a suitably defined (proper) prior distribution over the weight vector w^*. In particular, if we impose the prior $w^* \sim N(0, \frac{\sigma^2}{n} L^\dagger)$, Bayes' rule then leads to the posterior distribution

$$\mathbb{P}\big(w \mid y; X\big) \propto \exp\left(\frac{-1}{2\sigma^2}\|y - Xw\|_2^2\right) \exp\left(\frac{-n}{2\sigma^2} w^T L w\right) \mathbf{1}\{\langle w, 1\rangle = 0\}.$$

Thus conditioned on y, w is distributed as $N\left((X^T X + nL)^{-1} X^T y, \frac{\sigma^2}{2} L^\dagger\right)$. By applying iterated expectations, the Bayes risk is given by $\mathbb{E}\|w - \frac{1}{2} L^\dagger X^T y\|_2^2 = \frac{\sigma^2}{2}\mathrm{tr}(L^\dagger)$, which completes the proof.

Appendix D. Proof of Theorem 4

This section presents the proof of Theorem 4 for the setting of m-wise comparisons. We first state some simple properties of the model introduced in Section 3.3, which we use subsequently in the proofs of the results.

Lemma 10 *The Laplacian of the underlying pairwise-comparison graph satisfies the trace constraints* $\mathrm{nullspace}(L) = 1$, $\lambda_2(L) > 0$ *and* $\mathrm{tr}(L) = m(m-1)$.

Lemma 11 *For any $j \in [m]$, $i \in [n]$ and any vector $v \in \mathbb{R}^m$, we have*

$$\frac{\lambda_2(H)}{m} v^T(mI - 11^T)v \le v^T R_j H R_j^T v \le \frac{\lambda_{\max}(H)}{m} v^T(mI - 11^T)v.$$

See Section D.2 for the proof of these auxiliary lemmas.

D.1 Upper bound under the squared L semi-norm

We prove this upper bound by applying Lemma 9. In this case, the rescaled negative log likelihood takes the form

$$\ell(w) = -\frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \mathbf{1}[y_i = j] \log F\big(w^T E_i R_j\big),$$

and the MLE is obtained by constrained minimization over the set $\mathcal{W}_B := \big\{w \in \mathbb{R}^d \mid \langle 1, w\rangle = 0, \quad \text{and} \quad \|w\|_\infty \le B\big\}$. As in our proof of the upper bound in Theorem 1, we need to verify the κ-strong convexity condition, and to control the dual norm $\|\nabla \ell(w^*)\|_{L^\dagger}$.

Verifying strong convexity: The gradient of the negative log likelihood is

$$\nabla \ell(w) = -\frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \mathbf{1}[y_i = j] E_i R_j \nabla \log F(v)\big|_{v = w^T E_i R_j}.$$

The Hessian of the negative log likelihood can be written as

$$\nabla^2 \ell(w) = \frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \mathbf{1}[y_i = j] E_i R_j \nabla^2 \log F(v)\big|_{v = w^T E_i R_j} R_j^T E_i^T.$$

Using our strongly log-concave assumption on F, we have that for any vector $z \in \mathbb{R}^d$,

$$\begin{aligned} z^T \nabla^2 \ell(w) z &= -\frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \mathbf{1}[y_i = j] z^T E_i R_j \nabla^2 \log F(v)\big|_{v = w^T E_i R_j} R_j^T E_i^T z \\ &\ge \frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \mathbf{1}[y_i = j] z^T E_i R_j H R_j^T E_i^T z \\ &\ge \frac{\lambda_2(H)}{m}\frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \mathbf{1}[y_i = j] z^T E_i (mI - 11^T) E_i^T z, \end{aligned}$$

where the last step follows from Lemma 11. The definition (10) of L implies that

$$z^T \nabla^2 \ell(w) z \ge \frac{\lambda_2(H)}{m} z^T L z = \frac{\lambda_2(H)}{m}\|z\|_L^2.$$

Consequently, the κ-convexity condition holds around w^* with $\kappa = \frac{\lambda_2(H)}{m}$. An application of Lemma 9 then yields

$$\|\widehat{w}_{\mathrm{ML}} - w^*\|_L^2 \le \frac{m^2}{\lambda_2(H)^2}\|\nabla \ell(w^*)\|_{L^\dagger}^2 \;=\; \frac{m^2}{\lambda_2(H)^2} \nabla \ell(w^*)^T L^\dagger \nabla \ell(w^*). \tag{24}$$

Controlling the dual norm: The gradient of the negative log likelihood can then be rewritten as $\nabla \ell(w^*) = -\frac{1}{n}\sum_{i=1}^n E_i V_i$, where each index $i \in [n]$, the random vector vector $V_i \in \mathbb{R}^m$ is given by $V_i := \sum_{j=1}^m \mathbf{1}[y_i = j]\; R_j \nabla \log F(\langle w^*, E_i\rangle R_j)$. Now observe that the matrix $M := I - \frac{1}{m} 11^T$ is symmetric and positive semi-definite with rank $(m-1)$, eigenvalues $\{1, \ldots, 1, 0\}$, its nullspace equals the span of the all-ones vector, and that $M^\dagger = M$. Using this matrix, we define the transformed vector $\widetilde{V}_i := (M^\dagger)^{\frac{1}{2}} V_i$ for each $i \in [n]$.

Consider a vector x and its shifted version $x + t1$, where $t \in \mathbb{R}$ and 1 denotes the vector of all ones. By the shift invariance property, the function $g(t) = F(x + t1) - F(x)$ is constant, and hence

$$g'(0) = \langle \nabla F(x), 1\rangle = 0, \quad \text{and} \quad g''(0) = \langle 1, \big(\nabla^2 F(x)\big)1\rangle = 0, \tag{25}$$

which implies that $1 \in \text{nullspace}(\nabla^2 F(x))$. Continuing on, we also have that $\langle \nabla \log F(x),\, 1\rangle = \frac{1}{F(x)}\langle \nabla F(x),\, 1\rangle = 0$. Consequently, $\langle V_i,\, 1\rangle = 0 = \langle V_i,\, \text{nullspace}(M)\rangle$. This allows us to write

$$\nabla \ell(w^*) = -\frac{1}{n}\sum_{i=1}^{n} E_i M^{\frac{1}{2}} \widetilde{V}_i, \quad \text{and} \quad \nabla \ell(w^*)^T L^\dagger \nabla \ell(w^*) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\ell=1}^{n} \widetilde{V}_i^T M^{\frac{1}{2}} E_i^T L^\dagger E_\ell M^{\frac{1}{2}} \widetilde{V}_\ell.$$

By definition, for every pair $i \neq \ell \in [n]$, $\widetilde{V}_i$ is independent of $\widetilde{V}_\ell$. Moreover, for every $i \in [n]$,

$$\begin{aligned}
\mathbb{E}[\widetilde{V}_i] &= \mathbb{E}[(M^\dagger)^{\frac{1}{2}} \sum_{j=1}^{m} \mathbf{1}[y_i = j] R_j \nabla \log F(v)\big|_{v=(w^*)^T E_i R_j}] \\
&= (M^\dagger)^{\frac{1}{2}} \sum_{j=1}^{m} F((w^*)^T E_i R_j) R_j \nabla \log F(v)\big|_{v=(w^*)^T E_i R_j} \\
&= (M^\dagger)^{\frac{1}{2}} \sum_{j=1}^{m} R_j \nabla F(v)\big|_{v=(w^*)^T E_i R_j}.
\end{aligned}$$

In order to further evaluate this expression, define a function $g : \mathbb{R}^m \to \mathbb{R}$ as $g(z) = \sum_{j=1}^m F(z^T R_j)$. Then by definition we have $g(z) = 1$. Taking derivatives, we get $0 = \nabla g(z) = \sum_{j=1}^m R_j \nabla F(z^T R_j)$. It follows that $\mathbb{E}[\widetilde{V}_i] = 0$, and hence that

$$\begin{aligned}
\mathbb{E}[\nabla \ell(w^*)^T L^\dagger \nabla \ell(w^*)] &= \frac{1}{n^2}\mathbb{E}[\sum_{i=1}^{n}\sum_{\ell=1}^{n} \widetilde{V}_i^T M^{\frac{1}{2}} E_i^T L^\dagger E_\ell M^{\frac{1}{2}} \widetilde{V}_\ell] \\
&= \frac{1}{n^2}\mathbb{E}[\sum_{i=1}^{n} \widetilde{V}_i^T M^{\frac{1}{2}} E_i^T L^\dagger E_i M^{\frac{1}{2}} \widetilde{V}_i] \\
&\leq \frac{1}{n}\mathbb{E}[\sup_{\ell \in [n]} \|\widetilde{V}_\ell\|_2^2] \text{tr}(\frac{1}{n}\sum_{i=1}^{n} M^{\frac{1}{2}} E_i^T L^\dagger E_i M^{\frac{1}{2}}).
\end{aligned}$$

Since $L = \frac{m}{n}\sum_{i=1}^n E_i M E_i^T$, we have $\text{tr}(\frac{1}{n}\sum_{i=1}^n M^{\frac{1}{2}} E_i^T L^\dagger E_i M^{\frac{1}{2}}) = \frac{d-1}{m}$, as well as

$$\|\widetilde{V}_\ell\|_2^2 = \sum_{j=1}^{m} \mathbf{1}[y_i = j] (\nabla \log F(v)\big|_{v=(w^*)^T E_i R_j})^T R_j^T M R_j \nabla \log F(v)\big|_{v=(w^*)^T E_i R_j}.$$

Recalling the previously defined matrix M, observe that since R_j is simply a permutation matrix, we have $R_j^T M R_j = M$ for every $j \in [m]$. By chain rule, we have $\langle \nabla \log F(v),\, 1\rangle = \frac{1}{F(v)}\langle \nabla F(v),\, 1\rangle = 0$, where the last step follows from our previous calculation. It follows that

$$\mathbb{E}\big[\langle \nabla \ell(w^*),\, L^\dagger \nabla \ell(w^*)\rangle\big] \leq \frac{d}{n} \sup_{v \in [-B,B]^m} \|\nabla \log F(v)\|_2^2.$$

Substituting this bound into equation (24) yields the claim.

D.1.1 Lower bound under the squared L semi-norm

For any pair of quality score vectors w^j and w^k, the KL divergence between the distributions $\mathbb{P}_{w^j}$ and $\mathbb{P}_{w^k}$ is given by

$$D_{\text{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) = \sum_{i=1}^{n}\sum_{l=1}^{m} F(w^{jT} E_i R_l) \log \frac{F(w^{jT} E_i R_l)}{F(w^{kT} E_i R_l)}.$$

Applying the inequality $\log x \leq x - 1$, valid for $x > 0$, we find that

$$D_{\text{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq \sum_{i=1}^{n}\sum_{l=1}^{m} F(w^{jT} E_i R_l)\Big(\frac{F(w^{jT} E_i R_l)}{F(w^{kT} E_i R_l)} - 1\Big).$$

Now employing the fact that $\sum_{l=1}^m F(w^{jT} E_i R_l) = \sum_{l=1}^m F(w^{kT} E_i R_l) = 1$ gives

$$\begin{aligned}
D_{\text{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) &\leq \sum_{i=1}^{n}\sum_{l=1}^{m} \Big(\frac{F(w^{jT} E_i R_l)^2}{F(w^{kT} E_i R_l)} - 2F(w^{jT} E_i R_l) + F(w^{kT} E_i R_l)\Big). \\
&= \sum_{i=1}^{n}\sum_{l=1}^{m} \frac{(F(w^{jT} E_i R_l) - F(w^{kT} E_i R_l))^2}{F(w^{kT} E_i R_l)} \\
&\leq \frac{1}{F(-B, B, \ldots, B)} \sum_{i=1}^{n}\sum_{l=1}^{m} (F(w^{jT} E_i R_l) - F(w^{kT} E_i R_l))^2 \\
&\leq \frac{1}{F(-B, B, \ldots, B)} \sum_{i=1}^{n}\sum_{l=1}^{m} (\langle \nabla F(z_{il}),\, w^{jT} E_i R_l - w^{kT} E_i R_l\rangle)^2,
\end{aligned}$$

for some $z_{il} \in [-B, B]^m$. Letting $\zeta = \frac{\sup_{z \in [-B,B]^m} \|\nabla F(z)\|_{H^\dagger}^2}{F(-B,B,\ldots,B)}$ and applying Lemma 16 (noting that $\langle w^{jT} E_i R_l,\, \text{nullspace}(H)\rangle = 0$ for all i, j, l) gives

$$\begin{aligned}
D_{\text{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) &\leq \sum_{i=1}^{n}\sum_{l=1}^{m} \zeta \|w^{jT} E_i R_l - w^{kT} E_i R_l\|_H^2 \\
&\leq \zeta (w^j - w^k)^T \Big(\sum_{i=1}^{n}\sum_{l=1}^{m} E_i R_l H R_l^T E_i^T\Big)(w^j - w^k) \\
&\leq \zeta \lambda_m(H) n \|w^j - w^k\|_L^2, \qquad (26)
\end{aligned}$$

where the final step is a result of Lemma 11.

Consider the pair of scalars $\alpha \in (0, \frac{1}{4})$ and $\delta > 0$ whose values will be specified later. Let $M(\alpha)$ be as defined in (14). Consider the packing set $\{w^1, \ldots, w^{M(\alpha)}\}$ constructed in Appendix A.1. Each of these vectors is of length d, satisfies $\langle w^j,\, 1\rangle = 0$, and furthermore, each pair from this set satisfies $\alpha\delta^2 \leq \|w^j - w^k\|_L^2 \leq \delta^2$. Setting $\delta^2 = 0.01\frac{d}{n\zeta\lambda_m(H)}$ yields

$$D_{\text{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq 0.01d.$$

Every element from the packing set also satisfies $\|w^j\|_\infty \leq B$ when $n \geq \frac{0.01\sigma^2 \text{tr}(L^\dagger)}{\zeta B^2 \lambda_m(H)}$, and thus belongs to the class $\mathcal{W}_B$.

Applying Lemma 6 yields the lower bound

$$\|\widehat{w} - w^*\|_L^2 \geq \frac{\alpha}{2} 0.01 \frac{d}{n\zeta\lambda_m(H)} \Big\{ 1 - \frac{0.01d + \log 2}{\log M(\alpha)} \Big\}.$$

Setting $\alpha = 0.01$ proves the claim for $d > 9$.

For the case of $d \leq 9$, consider the set of the three d-length vectors $z^1 = [0 \ \cdots \ 0 \ -1]$, $z^2 = [0 \ \cdots \ 0 \ 1]$ and $z^3 = [0 \ \cdots \ 0 \ 0]$. Construct the packing set $\{w^1, w^2, w^3\}$ from these three vectors $\{z^1, z^2, z^3\}$ as done above for the case of $d > 9$. From the calculations made for the general case above, we have for all pairs $\min_{j \neq k} \|w^j - w^k\|_L^2 \geq \frac{\delta^2}{9}$ and $\max_{j,k} \|w^j - w^k\|_L^2 \leq 4\delta^2$, and as a result $\max_{j,k} D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq 4n\zeta\lambda_m(H)\delta^2$. Choosing $\delta^2 = \frac{\log 2}{8n\zeta\lambda_m(H)}$ and applying Lemma 6 proves the claim.

D.1.2 Upper bound under the squared Euclidean norm

The upper bound under the squared ℓ_2-norm follows directly from the upper bound under the squared L semi-norm in Theorem 4: noting that $(w^* - \widehat{w}) \perp \text{nullspace}(L)$, we get that

$$(w^* - \widehat{w})^T L (w^* - \widehat{w}) \geq \lambda_2(L) \|w^* - \widehat{w}\|_2^2.$$

Substituting this inequality in the upper bound on the minimax risk under the squared L semi-norm in Theorem 4 gives the desired result.

D.1.3 Lower bound under the squared Euclidean norm

Define $\zeta = \frac{\sup_{z \in [-B,B]^m} \|\nabla F(z)\|_{H^\dagger}^2}{F(-B,B,\ldots,B)}$. Equation (26) in Appendix D.1.1 shows that for any vectors $w^j, w^k \in \mathcal{W}_B$,

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq \zeta\lambda_m(H) n \|w^j - w^k\|_L^2,$$

Consider the pair of scalars $\alpha \in (0, \frac{1}{4})$ and $\delta > 0$ whose values will be specified later. Let $M(\alpha)$ be as defined in (14). In Appendix B.2 we constructed a set $\{w^1, \ldots, w^{M(\alpha)}\}$ of vectors of length d that satisfy $\langle w^j, 1\rangle = 0$ for every $j \in [M(\alpha)]$, and for every pair of vectors in this set, $\|w^j - w^k\|_2^2 \geq \alpha\delta^2$ and $\frac{1}{\binom{M(\alpha)}{2}} \sum_{j \neq k} \|\widetilde{w}^j - \widetilde{w}^k\|_L^2 \leq \frac{2\delta^2}{d} \mathrm{tr}(L)$. Applying Lemma 10 gives

$$\frac{1}{\binom{M(\alpha)}{2}} \sum_{j \neq k} \|\widetilde{w}^j - \widetilde{w}^k\|_L^2 \leq \frac{2\delta^2}{d} m(m-1).$$

Setting $\delta^2 = 0.005 \frac{d^2}{n\zeta\lambda_m(H)m(m-1)}$ yields

$$D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq 0.01d.$$

In a manner similar to Lemma 14 in the pairwise comparison case, one can show that in the general setting of this section, $\mathrm{tr}(L^\dagger) \geq \frac{d^2}{4m(m-1)}$. Then, every element from the packing set also satisfies $\|w^j\|_\infty \leq B$ when $\delta \leq B$, which holds true under our assumption

of $n \geq \frac{c\sigma^2 \mathrm{tr}(L^\dagger)}{\zeta B^2 \lambda_m(H)} \geq \frac{c\sigma^2 d^2}{4m(m-1)\zeta B^2 \lambda_m(H)}$ with $c = 0.01$. Each element of our packing set thus belongs to the class $\mathcal{W}_B$. Applying Lemma 6 yields the lower bound

$$\|\widehat{w} - w^*\|_L^2 \geq \frac{\alpha}{2} 0.01 \frac{d^2}{n\zeta\lambda_m(H)m(m-1)} \Big\{ 1 - \frac{0.01d + \log 2}{\log M(\alpha)} \Big\}.$$

Setting $\alpha = 0.01$ proves the claim for $d > 9$.

For the case of $d \leq 9$, consider the set of the three d-length vectors $z^1 = [0 \ \cdots \ 0 \ -1]$, $z^2 = [0 \ \cdots \ 0 \ 1]$ and $z^3 = [0 \ \cdots \ 0 \ 0]$. Construct the packing set $\{w^1, w^2, w^3\}$ from these three vectors $\{z^1, z^2, z^3\}$ as done above for the case of $d > 9$. From the calculations made for the general case above, we have for all pairs $\min_{j \neq k} \|w^j - w^k\|_2^2 \geq \frac{\delta^2}{9}$ and $\max_{j,k} \|w^j - w^k\|_L^2 \leq 4\delta^2$, and as a result $\max_{j,k} D_{\mathrm{KL}}(\mathbb{P}_{w^j} \| \mathbb{P}_{w^k}) \leq 4n\zeta\lambda_m(H)\delta^2$. Choosing $\delta^2 = \frac{\log 2}{8n\zeta\lambda_m(H)}$ and applying Lemma 6 proves the claim.

D.2 Some implied properties of the model

In this section, we prove the two auxiliary lemmas stated at the start of this appendix.

D.2.1 Proof of Lemma 10

From the definition (10) of L, have

$$L1 = \frac{1}{n} \sum_{i=1}^n E_i (mI - 11^T) E_i^T 1 \ = \ \frac{1}{n} \sum_{i=1}^n E_i (mI - 11^T) 1 \ = 0,$$

showing that $1 \in \text{nullspace}(L)$.

Now consider any non-zero vector $v := [v_1, \ldots, v_d]^T \in \mathbb{R}^d$ such that $v \notin \text{span}(1)$. Then there must exist some $i, j \in [d]$ such that $v_i \neq v_j$. We know that there exists some path from item i to j in the comparison hyper-graph. Thus there must exist some hyper-edge in this path with two items, say i', j', such that $v_{i'} \neq v_{j'}$. Suppose that hyper-edge corresponds to sample $\ell \in [n]$. Let $v' := E_\ell^T v$. Then $v' \notin \text{span}(1)$. The Cauchy-Schwarz inequality $\langle v', v'\rangle\langle 1, 1\rangle > (\langle v', 1\rangle)^2$ thus implies

$$v^T E_\ell (mI - 11^T) E_\ell^T v > 0.$$

Furthermore, for any $v'' \in \mathbb{R}^m$, the Cauchy-Schwarz inequality $\langle v'', v''\rangle\langle 1, 1\rangle > (\langle v'', 1\rangle)^2$ implies that for any $i \in [n]$, we have $v^T E_i (mI - 11^T) E_i^T v \geq 0$. Overall we conclude that have $v^T L v > 0$ for every $v \notin \text{span}(1)$, and hence, $\text{nullspace}(L) = 1$ and $\lambda_2(L) > 0$.

Finally, we have

$$\mathrm{tr}(L) = \frac{1}{n} \sum_{i=1}^n \mathrm{tr}(E_i (mI - 11^T) E_i^T) \ = \ \frac{1}{n} \sum_{i=1}^n \Big(m\mathrm{tr}(E_i E_i^T) - \mathrm{tr}(E_i 11^T E_i^T) \Big). \tag{27}$$

By the definition of the matrices $\{E_i\}_{i \in [n]}$, $\mathrm{tr}(E_i E_i^T) = m$ and $\mathrm{tr}(E_i 11^T E_i^T) = m$. Substituting these values in (27) gives the desired result $\mathrm{tr}(L) = m(m-1)$. ■

D.2.2 Proof of Lemma 11

Let $h_1, \ldots, h_m$ denote the m eigenvectors of H, with $h_1 = \frac{1}{\sqrt{m}}1$. Then for any vector $v' \in \mathbb{R}^m$,

$$v'^T H v' = \sum_{i=2}^m \lambda_i(H)\langle v',\, h_i\rangle^2 \geq \lambda_2(H)\sum_{i=2}^m \langle v',\, h_i\rangle^2 = \lambda_2(H)\Big(\sum_{i=1}^m \langle v',\, h_i\rangle^2 - \frac{1}{m}\langle v',\, 1\rangle^2\Big)$$
$$= \lambda_2(H) v'^T (I - \frac{1}{m}11^T)v',$$

where the final step employed the property $\sum_{i=1}^m h_i h_i^T = I$ of the eigenvectors $h_1, \ldots, h_m$ of H. A similar argument gives

$$v'^T H v' = \sum_{i=2}^m \lambda_i(H)\langle v',\, h_i\rangle^2 \leq \lambda_{\max}(H)\sum_{i=2}^m \langle v',\, h_i\rangle^2 = \lambda_{\max}(H)\Big(\sum_{i=1}^m \langle v',\, h_i\rangle^2 - \frac{1}{m}\langle v',\, 1\rangle^2\Big)$$
$$= \lambda_{\max}(H) v'^T (I - \frac{1}{m}11^T)v'.$$

Setting $v' = R_j^T v$ gives

$$\lambda_2(H) v^T R_j (I - \frac{1}{m}11^T) R_j^T v \leq v^T R_j H R_j^T v \leq \lambda_{\max}(H) v^T R_j (I - \frac{1}{m}11^T) R_j^T v.$$

Observe that the matrix $I - \frac{1}{m}11^T$ is invariant to permutation of the coordinates, and hence $R_j(I - \frac{1}{m}11^T)R_j^T = I - \frac{1}{m}11^T$. This gives

$$\frac{\lambda_2(H)}{m} v^T (mI - 11^T) v \leq v^T R_j H R_j^T v \leq \frac{\lambda_{\max}(H)}{m} v^T (mI - 11^T) v.$$

■

Appendix E. Some useful tail bounds

In this appendix, we collect a few useful tail bounds for quadratic forms in Gaussian and sub-Gaussian random variables.

Lemma 12 (Tail bound for Gaussian quadratic form) *For any positive semidefinite matrix Q and standard Gaussian vector $g \sim N(0, I_d)$, we have*

$$\mathbb{P}\big[g^T Q g \geq \big(\sqrt{\mathrm{tr}(Q)} + \sqrt{|||Q|||_{op}}\,\delta\big)^2\big] \leq e^{-\delta/2}. \tag{28}$$

valid for all $\delta \geq 0$.

Proof Note that the function $g \mapsto \|\sqrt{Q}g\|_2$ is Lipschitz with constant $\|\sqrt{Q}\|_{\mathrm{op}}$. Consequently, by concentration for Lipschitz functions of Gaussian vectors (Ledoux, 2001), the random variable $Z = \|\sqrt{Q}g\|_2$ satisfies the upper bound

$$\mathbb{P}\big[Z \geq \mathbb{E}[Z] + t\big] \leq \exp\big(-\frac{t^2}{2\|\sqrt{Q}\|_{\mathrm{op}}^2}\big) = \exp\big(-\frac{t^2}{2\|Q\|_{\mathrm{op}}}\big).$$

By Jensen's inequality, we have $\mathbb{E}[Z] = \mathbb{E}[\|\sqrt{Q}g\|_2] \leq \sqrt{\mathbb{E}[g^T Q g]} = \sqrt{\mathrm{tr}(Q)}$. Setting $t = \sqrt{|||Q|||_{\mathrm{op}}}\,\delta$ completes the proof. ■

Lemma 13 ((Hanson and Wright, 1971; Rudelson and Vershynin, 2013)) *Let $V \in \mathbb{R}^d$ be a random vector with independent zero-mean components that are sub-Gaussian with parameter K, and let $M \in \mathbb{R}^{d\times d}$ be an arbitrary matrix. Then there is a universal constant $c > 0$ such that*

$$\mathbb{P}\left[\left|V^T M V - \mathbb{E}[V^T M V]\right| > t\right) \leq 2\exp\left(-c\min\left\{\frac{t^2}{K^4|||M|||_{fro}^2}, \frac{t}{K^2|||M|||_{op}}\right\}\right) \qquad \text{for all } t > 0. \tag{29}$$

Appendix F. Properties of Laplacian matrices

By construction, the Laplacian L of the comparison graph is symmetric and positive-semidefinite. By the singular value decomposition, we can write $L = U^T \Lambda U$ where $U \in \mathbb{R}^{d\times d}$ is an orthonormal matrix, and Λ is a diagonal matrix of nonnegative eigenvalues with $\Lambda_{jj} = \lambda_j(L)$ for every $j \in [d]$. Given our assumption of $\lambda_1(L) \leq \cdots \leq \lambda_d(L)$, we also have $\Lambda_{11} \leq \cdots \leq \Lambda_{dd}$. Also recall that $L^\dagger$ denotes the Moore-Penrose pseudo-inverse of L. In terms of the notation introduced, the Moore-Penrose pseudo-inverse is then given by $L^\dagger = U^T \Lambda^\dagger U$, where $\Lambda^\dagger$ is a diagonal matrix with entries

$$\Lambda^\dagger_{jj} = \begin{cases} (\Lambda_{jj}^{-1}) & \text{if } \Lambda_{jj} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

The following pair of lemmas establish some useful properties about L.

Lemma 14 *The Laplacian matrix (4) satisfies the trace constraints*

$$\mathrm{tr}(L) = 2, \quad \textit{and} \quad \mathrm{tr}(L^\dagger) \geq \frac{d^2}{4}.$$

Proof From the definition (4) of the matrix L, we have $\mathrm{tr}(L) = \frac{1}{n}\sum_{i=1}^n \mathrm{tr}(x_i x_i^T) = 2$. We also know that $\lambda_1(L) = 0$, and hence $\sum_{j=2}^d \lambda_j(L) = 2$. Given the latter constraint, the sum $\sum_{j=2}^d \frac{1}{\lambda_j(L)}$ is minimized when $\lambda_2(L) = \cdots = \lambda_d(L)$. Some simple algebra now gives the claimed result. ■

Lemma 15 *For the matrix L defined in (4), and for a $(n \times d)$ matrix X with x_i^T as its i^{th} row,*

$$\mathrm{tr}(\frac{1}{n}x^T L^\dagger x) = d - 1, \quad |||\frac{1}{n}x^T L^\dagger x|||_{fro} = d - 1, \quad \textit{and} \quad |||\frac{1}{n}x^T L^\dagger x|||_{op} = 1.$$

Proof Let $Q = \frac{1}{n} x^T L^\dagger x$. Since $L = \frac{1}{n} X^T X = U^T \Lambda U$, the diagonal entries of Λ are the squared singular values of $X/\sqrt{n}$. Consequently, there must exist an orthonormal matrix V such that $X/\sqrt{n} = V\sqrt{\Lambda}U^T$, and thus we can write $Q = V\sqrt{\Lambda}\,\Lambda^\dagger\,\sqrt{\Lambda}\,V^T$. By definition of the Moore-Penrose pseudo-inverse, the matrix $\sqrt{\Lambda}\,\Lambda^\dagger\,\sqrt{\Lambda}$ is a diagonal matrix; since the Laplacian graph is connected, its diagonal contains $(d-1)$ ones and a single zero. Noting that V is an orthonormal matrix gives the claimed result. ■

For future reference, we state and prove a lemma showing that these two semi-norms satisfy a restricted form of the Cauchy-Schwarz inequality:

Lemma 16 *For any two vectors u and v such that $u \perp \text{nullspace}(L)$ or/and $v \perp \text{nullspace}(L)$, we have*

$$|\langle u, v\rangle| \leq \|u\|_{L^\dagger}\, \|v\|_L. \tag{30}$$

Proof Since $L = U^T \Lambda U$ and $L^\dagger = U^T \Lambda^\dagger U$, we have

$$\sqrt{v^T L v}\sqrt{u^T L^\dagger u} = \sqrt{v^T U^T \Lambda U v}\,\sqrt{u^T U^T \Lambda^\dagger U u} = \|\widetilde{v}\|_2 \|\widetilde{u}\|_2 \geq |\langle \widetilde{v}, \widetilde{u}\rangle|,$$

where we have defined $\widetilde{v} := \sqrt{\Lambda} U v$ and $\widetilde{u} := \sqrt{\Lambda^\dagger} U u$. Continuing on,

$$\langle \widetilde{v}, \widetilde{u}\rangle = v^T U^T \sqrt{\Lambda}\sqrt{\Lambda^\dagger} U u \;=\; v^T U U^T u,$$

where we have used the fact that u or/and v are orthogonal to the null space of L. Since U is orthonormal, we conclude that $\langle \widetilde{v}, \widetilde{u}\rangle = \langle v, u\rangle$, which completes the proof. ■

Appendix G. Minimax risk without assumptions on quality scores

The setting considered throughout the paper imposes two restrictions (2) on the quality score vector w^*. The first condition is that of shift invariance, that is, $\langle w^*, 1\rangle = 0$. The necessity of this condition for identifiability under the Ordinal model is easy to verify. The second condition is that the quality score vectors are B-bounded, that is, $\|w^*\|_\infty \leq B$ for some finite B. In this section, for the sake of completeness, we show that the minimax risk is infinite in the absence of this condition.

Proposition 17 *Any estimator $\widetilde{w}$ based on n samples from the* Ordinal *model (with unbounded quality score vectors) has error lower bounded as*

$$\sup_{w^* \in \mathcal{W}_\infty} \mathbb{E}\Big[\|\widetilde{w} - w^*\|_2^2\Big] = \sup_{w^* \in \mathcal{W}_\infty} \mathbb{E}\Big[\|\widetilde{w} - w^*\|_L^2\Big] = \infty.$$

The remainder of this section is devoted to the formal proof of Proposition 17. Consider the event where for every comparison, the item with the higher quality score in w^* wins. For any $w^* \in \mathcal{W}_\infty \backslash \{0\}$, this event occurs with a probability at least $\frac{1}{2^n}$. Under this event, the true w^* is indistinguishable from the quality score vector $cw^* \in \mathcal{W}_\infty$ for every $c \geq 0$, and the error is also unbounded. Since the probability of this event is strictly bounded away from zero, the expected error is also unbounded.

References

Ammar Ammar and Devavrat Shah. Ranking: Compare, don't score. In *Allerton Conference on Communication, Control, and Computing*, pages 776–783, 2011.

Donald R Atkinson, Bruce E Wampold, Susana M Lowe, Linda Matthews, and Hyun-Nie Ahn. Asian American preferences for counselor characteristics: Application of the Bradley-Terry-Luce model to paired comparison data. *The Counseling Psychologist*, 26 (1):101–123, 1998.

Hossein Azari Soufiani, David C Parkes, and Lirong Xia. Preference elicitation for general random utility models. In *Uncertainty in Artificial Intelligence: Proceedings of the 29th Conference*. AUAI Press, 2013.

William Barnett. The modern theory of consumer behavior: Ordinal or cardinal? *The Quarterly Journal of Austrian Economics*, 6(1):41–65, 2003.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, pages 324–345, 1952.

Tom Bramley. A rank-ordering method for equating tests by expert judgment. *Journal of Applied Measurement*, 6(2):202–223, 2005.

Mark Braverman and Elchanan Mossel. Noisy sorting without resampling. In *Symposium on Discrete Algorithms*, pages 268–276, 2008.

Andries E Brouwer and Willem H Haemers. *Spectra of graphs*. Springer, 2011.

Sourav Chatterjee. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2014.

Xi Chen, Paul N Bennett, Kevyn Collins-Thompson, and Eric Horvitz. Pairwise ranking aggregation in a crowdsourced setting. In *International Conference on Web Search and Data Mining*, pages 193–202, 2013.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

Ofer Gabber and Zvi Galil. Explicit constructions of linear-sized superconcentrators. *Journal of Computer and System Sciences*, 22(3):407–420, 1981.

Edgar N Gilbert. A comparison of signalling alphabets. *Bell System Technical Journal*, 31 (3):504–522, 1952.

Paul E Green, J Douglas Carroll, and Wayne S DeSarbo. Estimating choice probabilities in multiattribute decision making. *Journal of Consumer Research*, pages 76–84, 1981.

Bruce Hajek, Sewoong Oh, and Jiaming Xu. Minimax-optimal inference from partial rankings. In *Advances in Neural Information Processing Systems*, pages 1475–1483, 2014.

David Lee Hanson and Farroll Tim Wright. A bound on tail probabilities for quadratic forms in independent random variables. *The Annals of Mathematical Statistics*, pages 1079–1083, 1971.

Sandra Heldsinger and Stephen Humphry. Using the method of pairwise comparison to obtain reliable teacher assessments. *The Australian Educational Researcher*, 37(2):1–19, 2010.

Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill: A Bayesian skill rating system. In *Advances in Neural Information Processing Systems*, volume 19, page 569, 2007.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

Srikanth Jagabathula and Devavrat Shah. Inferring rankings under constrained sensing. In *Advances in Neural Information Processing Systems*, pages 753–760, 2008.

Kevin G Jamieson and Robert Nowak. Active ranking using pairwise comparisons. In *Advances in Neural Information Processing Systems*, pages 2240–2248, 2011.

Gabriella Kazai. In search of quality in crowdsourcing for search engine evaluation. In *Advances in Information Retrieval*, pages 165–176. Springer, 2011.

Zahid Y Khairullah and Stanley Zionts. An approach for preference ranking of alternatives. *European journal of operational research*, 28(3):329–342, 1987.

Firas Khatib, Frank DiMaio, Seth Cooper, Maciej Kazmierczyk, Miroslaw Gilski, Szymon Krzywda, Helena Zabranska, Iva Pichova, James Thompson, Zoran Popović, Mariusz Jaskolski, and David Baker. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature structural & molecular biology*, 18(10):1175–1177, 2011.

John I Kiger. The depth/breadth trade-off in the design of menu-driven user interfaces. *International Journal of Man-Machine Studies*, 20(2):201–213, 1984.

Kenneth J Koehler and Harold Ridpath. An application of a biased version of the Bradley-Terry-Luce model to professional basketball results. *Journal of Mathematical Psychology*, 25(3), 1982.

Paul FM Krabbe. Thurstone scaling as a measurement method to quantify subjective health outcomes. *Medical care*, 46(4):357–365, 2008.

ASID Lang and Joshua Rio-Ross. Using Amazon Mechanical Turk to transcribe historical handwritten documents. *The Code4Lib Journal*, 2011.

M. Ledoux. *The Concentration of Measure Phenomenon*. Mathematical Surveys and Monographs. American Mathematical Society, Providence, RI, 2001.

Michael D Lee, Mark Steyvers, Mindy De Young, and Brent J Miller. A model-based approach to measuring expertise in ranking tasks. In *Proceedings of the 33rd annual conference of the cognitive science society*, 2011.

E.L. Lehmann and G. Casella. *Theory of Point Estimation*. Springer Texts in Statistics, 1998.

Peter John Loewen, Daniel Rubenson, and Arthur Spirling. Testing the power of arguments in referendums: A Bradley–Terry approach. *Electoral Studies*, 31(1):212–221, 2012.

R Duncan Luce. *Individual Choice Behavior: A Theoretical Analysis*. New York: Wiley, 1959.

Miguel Angel Luengo-Oroz, Asier Arranz, and John Frean. Crowdsourcing malaria parasite quantification: an online game for analyzing images of infected thick blood smears. *Journal of medical Internet research*, 14(6), 2012.

George A Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

Sahand Negahban, Sewoong Oh, and Devavrat Shah. Iterative ranking from pair-wise comparisons. In *Advances in Neural Information Processing Systems*, pages 2474–2482, 2012.

Sahand Negahban, Sewoong Oh, and Devavrat Shah. Rank centrality: Ranking from pairwise comparisons. *arXiv preprint arXiv:1209.1688v4*, 2015.

Robert M Nosofsky. Luce's choice model and Thurstone's categorical judgment model compared: Kornbrot's data revisited. *Attention, Perception, & Psychophysics*, 37(1): 89–91, 1985.

Roberto Imbuzeiro Oliveira. Concentration of the adjacency matrix and of the laplacian in random graphs with independent edges. *arXiv preprint arXiv:0911.0600*, 2009.

Chris Piech, Jonathan Huang, Zhenghao Chen, Chuong Do, Andrew Ng, and Daphne Koller. Tuned models of peer assessment in MOOCs. In *International Conference on Educational Data Mining*, 2013.

Robin L Plackett. The analysis of permutations. *Applied Statistics*, pages 193–202, 1975.

Arun Rajkumar and Shivani Agarwal. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *Proceedings of the 31st International Conference on Machine Learning*, pages 118–126, 2014.

Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *The Journal of Machine Learning Research*, 99:1297–1322, 2010.

Daniel Ross. Arpad Elo and the Elo rating system, 2007. `http://en.chessbase.com/post/arpad-elo-and-the-elo-rating-system`.

Mark Rudelson and Roman Vershynin. Hanson-wright inequality and sub-gaussian concentration. *Electronic Communications in Probability*, 18:1–9, 2013.

Thomas L Saaty and Mujgan S Ozdemir. Why the magic number seven plus or minus two. *Mathematical and Computer Modelling*, 38(3):233–244, 2003.

Nihar B Shah, Joseph K Bradley, Abhay Parekh, Martin Wainwright, and Kannan Ramchandran. A case for ordinal peer-evaluation in MOOCs. In *NIPS Workshop on Data Driven Education*, December 2013.

Nihar B Shah, Sivaraman Balakrishnan, Joseph Bradley, Abhay Parekh, Kannan Ramchandran, and Martin J. Wainwright. When is it better to compare than to score? *arXiv preprint arXiv:1406.6618*, 2014.

Richard M Shiffrin and Robert M Nosofsky. Seven plus or minus two: a commentary on capacity limitations. *Psychological Review*, 1994.

Neil Stewart, Gordon DA Brown, and Nick Chater. Absolute identification by relative judgment. *Psychological review*, 112(4):881, 2005.

John Swets. The relative operating characteristic in psychology. *Science*, 182(4116), 1973.

Louis L Thurstone. A law of comparative judgment. *Psychological Review*, 34(4):273, 1927.

Kenneth E Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.

Kristi Tsukida and Maya R Gupta. How to analyze paired comparison data. Technical report, DTIC Document, 2011.

A.B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics, 2008.

RR Varshamov. Estimate of the number of signals in error correcting codes. In *Dokl. Akad. Nauk SSSR*, volume 117, pages 739–741, 1957.

Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. Recaptcha: Human-based character recognition via web security measures. *Science*, 321 (5895):1465–1468, 2008.

Jing Wang, Panagiotis G Ipeirotis, and Foster Provost. Managing crowdsourcing workers. In *Winter Conference on Business Intelligence*, pages 10–12, 2011.

Jinfeng Yi, Rong Jin, Shaili Jain, and Anil Jain. Inferring users' preferences from crowdsourced pairwise comparisons: A matrix completion approach. In *AAAI Conference on Human Computation and Crowdsourcing*, 2013.

Journal of Machine Learning Research 17 (2016) 1-35 Submitted 5/15; Published 4/16

Domain-Adversarial Training of Neural Networks

Yaroslav Ganin ganin@skoltech.ru
Evgeniya Ustinova evgeniya.ustinova@skoltech.ru
Skolkovo Institute of Science and Technology (Skoltech)
Skolkovo, Moscow Region, Russia

Hana Ajakan hana.ajakan.1@ulaval.ca
Pascal Germain Pascal.Germain@ift.ulaval.ca
Département d'informatique et de génie logiciel, Université Laval
Québec, Canada, G1V 0A6

Hugo Larochelle hugo.larochelle@usherbrooke.ca
Département d'informatique, Université de Sherbrooke
Québec, Canada, J1K 2R1

François Laviolette Francois.Laviolette@ift.ulaval.ca
Mario Marchand Mario.Marchand@ift.ulaval.ca
Département d'informatique et de génie logiciel, Université Laval
Québec, Canada, G1V 0A6

Victor Lempitsky lempitsky@skoltech.ru
Skolkovo Institute of Science and Technology (Skoltech)
Skolkovo, Moscow Region, Russia

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

We introduce a new representation learning approach for domain adaptation, in which data at training and test time come from similar but different distributions. Our approach is directly inspired by the theory on domain adaptation suggesting that, for effective domain transfer to be achieved, predictions must be made based on features that cannot discriminate between the training (source) and test (target) domains.

The approach implements this idea in the context of neural network architectures that are trained on labeled data from the source domain and unlabeled data from the target domain (no labeled target-domain data is necessary). As the training progresses, the approach promotes the emergence of features that are (i) discriminative for the main learning task on the source domain and (ii) indiscriminate with respect to the shift between the domains. We show that this adaptation behaviour can be achieved in almost any feed-forward model by augmenting it with few standard layers and a new *gradient reversal* layer. The resulting augmented architecture can be trained using standard backpropagation and stochastic gradient descent, and can thus be implemented with little effort using any of the deep learning packages.

We demonstrate the success of our approach for two distinct classification problems (document sentiment analysis and image classification), where state-of-the-art domain adaptation performance on standard benchmarks is achieved. We also validate the approach for descriptor learning task in the context of person re-identification application.

Keywords: domain adaptation, neural network, representation learning, deep learning, synthetic data, image classification, sentiment analysis, person re-identification

1. Introduction

The cost of generating labeled data for a new machine learning task is often an obstacle for applying machine learning methods. In particular, this is a limiting factor for the further progress of deep neural network architectures, that have already brought impressive advances to the state-of-the-art across a wide variety of machine-learning tasks and applications. For problems lacking labeled data, it may be still possible to obtain training sets that are big enough for training large-scale deep models, but that suffer from the *shift* in data distribution from the actual data encountered at "test time". One important example is training an image classifier on synthetic or semi-synthetic images, which may come in abundance and be fully labeled, but which inevitably have a distribution that is different from real images (Liebelt and Schmid, 2010; Stark et al., 2010; Vázquez et al., 2014; Sun and Saenko, 2014). Another example is in the context of sentiment analysis in written reviews, where one might have labeled data for reviews of one type of product (*e.g.*, movies), while having the need to classify reviews of other products (*e.g.*, books).

Learning a discriminative classifier or other predictor in the presence of a *shift* between training and test distributions is known as *domain adaptation* (DA). The proposed approaches build mappings between the *source* (training-time) and the *target* (test-time) domains, so that the classifier learned for the source domain can also be applied to the target domain, when composed with the learned mapping between domains. The appeal of the domain adaptation approaches is the ability to learn a mapping between domains in the situation when the target domain data are either fully unlabeled (*unsupervised domain annotation*) or have few labeled samples (*semi-supervised domain adaptation*). Below, we focus on the harder unsupervised case, although the proposed approach (*domain-adversarial learning*) can be generalized to the semi-supervised case rather straightforwardly.

Unlike many previous papers on domain adaptation that worked with fixed feature representations, we focus on combining domain adaptation and deep feature learning within one training process. Our goal is to embed domain adaptation into the process of learning representation, so that the final classification decisions are made based on features that are both discriminative and invariant to the change of domains, *i.e.*, have the same or very similar distributions in the source and the target domains. In this way, the obtained feed-forward network can be applicable to the target domain without being hindered by the shift between the two domains. Our approach is motivated by the theory on domain adaptation (Ben-David et al., 2006, 2010), that suggests that a good representation for cross-domain transfer is one for which an algorithm cannot learn to identify the domain of origin of the input observation.

We thus focus on learning features that combine (i) discriminativeness and (ii) domain-invariance. This is achieved by jointly optimizing the underlying features as well as two discriminative classifiers operating on these features: (i) the *label predictor* that predicts class labels and is used both during training and at test time and (ii) the *domain classifier* that discriminates between the source and the target domains during training. While the parameters of the classifiers are optimized in order to minimize their error on the training set, the parameters of the underlying deep feature mapping are optimized in order to *minimize* the loss of the label classifier and to *maximize* the loss of the domain classifier. The latter

update thus works *adversarially* to the domain classifier, and it encourages domain-invariant features to emerge in the course of the optimization.

Crucially, we show that all three training processes can be embedded into an appropriately composed deep feed-forward network, called *domain-adversarial neural network* (DANN) (illustrated by Figure 1, page 12) that uses standard layers and loss functions, and can be trained using standard backpropagation algorithms based on stochastic gradient descent or its modifications (*e.g.*, SGD with momentum). The approach is generic as a DANN version can be created for almost any existing feed-forward architecture that is trainable by backpropagation. In practice, the only non-standard component of the proposed architecture is a rather trivial *gradient reversal* layer that leaves the input unchanged during forward propagation and reverses the gradient by multiplying it by a negative scalar during the backpropagation.

We provide an experimental evaluation of the proposed domain-adversarial learning idea over a range of deep architectures and applications. We first consider the simplest DANN architecture where the three parts (label predictor, domain classifier and feature extractor) are linear, and demonstrate the success of domain-adversarial learning for such architecture. The evaluation is performed for synthetic data as well as for the sentiment analysis problem in natural language processing, where DANN improves the state-of-the-art *marginalized Stacked Autoencoders* (mSDA) of Chen et al. (2012) on the common Amazon reviews benchmark.

We further evaluate the approach extensively for an image classification task, and present results on traditional deep learning image data sets—such as MNIST (LeCun et al., 1998) and SVHN (Netzer et al., 2011)—as well as on Office benchmarks (Saenko et al., 2010), where domain-adversarial learning allows obtaining a deep architecture that considerably improves over previous state-of-the-art accuracy.

Finally, we evaluate domain-adversarial *descriptor* learning in the context of person re-identification application (Gong et al., 2014), where the task is to obtain good pedestrian image descriptors that are suitable for retrieval and verification. We apply domain-adversarial learning, as we consider a *descriptor predictor* trained with a Siamese-like loss instead of the label predictor trained with a classification loss. In a series of experiments, we demonstrate that domain-adversarial learning can improve cross-data-set re-identification considerably.

2. Related work

The general approach of achieving domain adaptation explored under many facets. Over the years, a large part of the literature has focused mainly on linear hypothesis (see for instance Blitzer et al., 2006; Bruzzone and Marconcini, 2010; Germain et al., 2013; Baktashmotlagh et al., 2013; Cortes and Mohri, 2014). More recently, non-linear representations have become increasingly studied, including neural network representations (Glorot et al., 2011; Li et al., 2014) and most notably the state-of-the-art mSDA (Chen et al., 2012). That literature has mostly focused on exploiting the principle of robust representations, based on the denoising autoencoder paradigm (Vincent et al., 2008).

Concurrently, multiple methods of matching the feature distributions in the source and the target domains have been proposed for unsupervised domain adaptation. Some approaches perform this by reweighing or selecting samples from the source domain (Borgwardt et al., 2006; Huang et al., 2006; Gong et al., 2013), while others seek an explicit feature space transformation that would map source distribution into the target one (Pan et al., 2011; Gopalan et al., 2011; Baktashmotlagh et al., 2013). An important aspect of the distribution matching approach is the way the (dis)similarity between distributions is measured. Here, one popular choice is matching the distribution means in the kernel-reproducing Hilbert space (Borgwardt et al., 2006; Huang et al., 2006), whereas Gong et al. (2012) and Fernando et al. (2013) map the principal axes associated with each of the distributions.

Our approach also attempts to match feature space distributions, however this is accomplished by modifying the feature representation itself rather than by reweighing or geometric transformation. Also, our method uses a rather different way to measure the disparity between distributions based on their separability by a deep discriminatively-trained classifier. Note also that several approaches perform transition from the source to the target domain (Gopalan et al., 2011; Gong et al., 2012) by changing gradually the training distribution. Among these methods, Chopra et al. (2013) does this in a "deep" way by the layerwise training of a sequence of deep autoencoders, while gradually replacing source-domain samples with target-domain samples. This improves over a similar approach of Glorot et al. (2011) that simply trains a single deep autoencoder for both domains. In both approaches, the actual classifier/predictor is learned in a separate step using the feature representation learned by autoencoder(s). In contrast to Glorot et al. (2011); Chopra et al. (2013), our approach performs feature learning, domain adaptation and classifier learning jointly, in a unified architecture, and using a single learning algorithm (backpropagation). We therefore argue that our approach is simpler (both conceptually and in terms of its implementation). Our method also achieves considerably better results on the popular Office benchmark.

While the above approaches perform unsupervised domain adaptation, there are approaches that perform *supervised* domain adaptation by exploiting labeled data from the target domain. In the context of deep feed-forward architectures, such data can be used to "fine-tune" the network trained on the source domain (Zeiler and Fergus, 2013; Oquab et al., 2014; Babenko et al., 2014). Our approach does not require labeled target-domain data. At the same time, it can easily incorporate such data when they are available.

An idea related to ours is described in Goodfellow et al. (2014). While their goal is quite different (building generative deep networks that can synthesize samples), the way they measure and minimize the discrepancy between the distribution of the training data and the distribution of the synthesized data is very similar to the way our architecture measures and minimizes the discrepancy between feature distributions for the two domains. Moreover, the authors mention the problem of saturating sigmoids which may arise at the early stages of training due to the significant dissimilarity of the domains. The technique they use to circumvent this issue (the "adversarial" part of the gradient is replaced by a gradient computed with respect to a suitable cost) is directly applicable to our method.

Also, recent and concurrent reports by Tzeng et al. (2014); Long and Wang (2015) focus on domain adaptation in feed-forward networks. Their set of techniques measures and minimizes the distance between the data distribution means across domains (potentially, after embedding distributions into RKHS). Their approach is thus different from our idea of matching distributions by making them indistinguishable for a discriminative classifier.

Below, we compare our approach to Tzeng et al. (2014); Long and Wang (2015) on the Office benchmark. Another approach to deep domain adaptation, which is arguably more different from ours, has been developed in parallel by Chen et al. (2015).

From a theoretical standpoint, our approach is directly derived from the seminal theoretical works of Ben-David et al. (2006, 2010). Indeed, DANN directly optimizes the notion of $\mathcal{H}$-divergence. We do note the work of Huang and Yates (2012), in which HMM representations are learned for word tagging using a posterior regularizer that is also inspired by Ben-David et al.'s work. In addition to the tasks being different—Huang and Yates (2012) focus on word tagging problems—, we would argue that DANN learning objective more closely optimizes the $\mathcal{H}$-divergence, with Huang and Yates (2012) relying on cruder approximations for efficiency reasons.

A part of this paper has been published as a conference paper (Ganin and Lempitsky, 2015). This version extends Ganin and Lempitsky (2015) very considerably by incorporating the report Ajakan et al. (2014) (presented as part of the *Second Workshop on Transfer and Multi-Task Learning*), which brings in new terminology, in-depth theoretical analysis and justification of the approach, extensive experiments with the shallow DANN case on synthetic data as well as on a natural language processing task (sentiment analysis). Furthermore, in this version we go beyond classification and evaluate domain-adversarial learning for descriptor learning setting within the person re-identification application.

3. Domain Adaptation

We consider classification tasks where X is the input space and $Y = \{0, 1, \ldots, L-1\}$ is the set of L possible labels. Moreover, we have two different distributions over $X \times Y$, called the *source domain* $\mathcal{D}_\mathrm{S}$ and the *target domain* $\mathcal{D}_\mathrm{T}$. An *unsupervised domain adaptation* learning algorithm is then provided with a *labeled source sample* S drawn *i.i.d.* from $\mathcal{D}_\mathrm{S}$, and an *unlabeled target sample* T drawn *i.i.d.* from $\mathcal{D}_\mathrm{T}^X$, where $\mathcal{D}_\mathrm{T}^X$ is the marginal distribution of $\mathcal{D}_\mathrm{T}$ over X.

$$S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \sim (\mathcal{D}_\mathrm{S})^n\,; \quad T = \{\mathbf{x}_i\}_{i=n+1}^N \sim (\mathcal{D}_\mathrm{T}^X)^{n'},$$

with $N = n + n'$ being the total number of samples. The goal of the learning algorithm is to build a classifier $\eta : X \to Y$ with a low *target risk*

$$R_{\mathcal{D}_\mathrm{T}}(\eta) \;=\; \Pr_{(\mathbf{x},y)\sim\mathcal{D}_\mathrm{T}} \Big(\eta(\mathbf{x}) \neq y\Big)\,,$$

while having no information about the labels of $\mathcal{D}_\mathrm{T}$.

3.1 Domain Divergence

To tackle the challenging domain adaptation task, many approaches bound the target error by the sum of the source error and a notion of distance between the source and the target distributions. These methods are intuitively justified by a simple assumption: the source risk is expected to be a good indicator of the target risk when both distributions are similar. Several notions of distance have been proposed for domain adaptation (Ben-David et al., 2006, 2010; Mansour et al., 2009a,b; Germain et al., 2013). In this paper, we focus on the $\mathcal{H}$-divergence used by Ben-David et al. (2006, 2010), and based on the earlier work of Kifer et al. (2004). Note that we assume in definition 1 below that the hypothesis class $\mathcal{H}$ is a (discrete or continuous) set of binary classifiers $\eta : X \to \{0, 1\}$.[1]

Definition 1 (Ben-David et al., 2006, 2010; Kifer et al., 2004) *Given two domain distributions $\mathcal{D}_\mathrm{S}^X$ and $\mathcal{D}_\mathrm{T}^X$ over X, and a hypothesis class $\mathcal{H}$, the* $\mathcal{H}$-divergence *between $\mathcal{D}_\mathrm{S}^X$ and $\mathcal{D}_\mathrm{T}^X$ is*

$$d_{\mathcal{H}}(\mathcal{D}_\mathrm{S}^X, \mathcal{D}_\mathrm{T}^X) \;=\; 2 \sup_{\eta\in\mathcal{H}} \left| \Pr_{\mathbf{x}\sim\mathcal{D}_\mathrm{S}^X} \big[\eta(\mathbf{x}) = 1\big] - \Pr_{\mathbf{x}\sim\mathcal{D}_\mathrm{T}^X} \big[\eta(\mathbf{x}) = 1\big] \right|.$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples generated by $\mathcal{D}_\mathrm{S}^X$ from examples generated by $\mathcal{D}_\mathrm{T}^X$. Ben-David et al. (2006, 2010) proved that, for a symmetric hypothesis class $\mathcal{H}$, one can compute the *empirical* $\mathcal{H}$*-divergence* between two samples $S \sim (\mathcal{D}_\mathrm{S}^X)^n$ and $T \sim (\mathcal{D}_\mathrm{T}^X)^{n'}$ by computing

$$\hat{d}_{\mathcal{H}}(S, T) \;=\; 2\left(1 - \min_{\eta\in\mathcal{H}} \left[\frac{1}{n}\sum_{i=1}^{n} I[\eta(\mathbf{x}_i)=0] + \frac{1}{n'}\sum_{i=n+1}^{N} I[\eta(\mathbf{x}_i)=1]\right]\right), \tag{1}$$

where $I[a]$ is the indicator function which is 1 if predicate a is true, and 0 otherwise.

3.2 Proxy Distance

Ben-David et al. (2006) suggested that, even if it is generally hard to compute $\hat{d}_{\mathcal{H}}(S, T)$ exactly (*e.g.*, when $\mathcal{H}$ is the space of linear classifiers on X), we can easily approximate it by running a learning algorithm on the problem of discriminating between source and target examples. To do so, we construct a new data set

$$U \;=\; \{(\mathbf{x}_i, 0)\}_{i=1}^n \cup \{(\mathbf{x}_i, 1)\}_{i=n+1}^N\,, \tag{2}$$

where the examples of the source sample are labeled 0 and the examples of the target sample are labeled 1. Then, the risk of the classifier trained on the new data set U approximates the "min" part of Equation (1). Given a generalization error ϵ on the problem of discriminating between source and target examples, the $\mathcal{H}$-divergence is then approximated by

$$\hat{d}_{\mathcal{A}} \;=\; 2\,(1 - 2\epsilon)\,. \tag{3}$$

In Ben-David et al. (2006), the value $\hat{d}_{\mathcal{A}}$ is called the *Proxy* $\mathcal{A}$*-distance* (PAD). The $\mathcal{A}$*-distance* being defined as $d_{\mathcal{A}}(\mathcal{D}_\mathrm{S}^X, \mathcal{D}_\mathrm{T}^X) = 2 \sup_{A\in\mathcal{A}} \left| \Pr_{\mathcal{D}_\mathrm{S}^X}(A) - \Pr_{\mathcal{D}_\mathrm{T}^X}(A) \right|$, where $\mathcal{A}$ is a subset of X. Note that, by choosing $\mathcal{A} = \{A_\eta | \eta \in \mathcal{H}\}$, with A_η the set represented by the characteristic function η, the $\mathcal{A}$-distance and the $\mathcal{H}$-divergence of Definition 1 are identical.

In the experiments section of this paper, we compute the PAD value following the approach of Glorot et al. (2011); Chen et al. (2012), *i.e.*, we train either a linear SVM or a deeper MLP classifier on a subset of U (Equation 2), and we use the obtained classifier error on the other subset as the value of ϵ in Equation (3). More details and illustrations of the linear SVM case are provided in Section 5.1.5.

1. As mentioned by Ben-David et al. (2006), the same analysis holds for multiclass setting. However, to obtain the same results when $|Y| > 2$, one should assume that $\mathcal{H}$ is a symmetrical hypothesis class. That is, for all $h \in \mathcal{H}$ and any permutation of labels $c : Y \to Y$, we have $c(h) \in \mathcal{H}$. Note that this is the case for most commonly used neural network architectures.

3.3 Generalization Bound on the Target Risk

The work of Ben-David et al. (2006, 2010) also showed that the $\mathcal{H}$-divergence $d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^X, \mathcal{D}_{\mathrm{T}}^X)$ is upper bounded by its empirical estimate $\hat{d}_{\mathcal{H}}(S,T)$ plus a constant complexity term that depends on the *VC dimension* of $\mathcal{H}$ and the size of samples S and T. By combining this result with a similar bound on the source risk, the following theorem is obtained.

Theorem 2 (Ben-David et al., 2006) *Let $\mathcal{H}$ be a hypothesis class of VC dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_{\mathrm{S}})^n$ and $T \sim (\mathcal{D}_{\mathrm{T}}^X)^n$, for every $\eta \in \mathcal{H}$:*

$$R_{\mathcal{D}_{\mathrm{T}}}(\eta) \;\leq\; R_S(\eta) + \sqrt{\frac{4}{n}\left(d\log\tfrac{2en}{d} + \log\tfrac{4}{\delta}\right)} + \hat{d}_{\mathcal{H}}(S,T) + 4\sqrt{\frac{1}{n}\left(d\log\tfrac{2n}{d} + \log\tfrac{4}{\delta}\right)} + \beta\,,$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_{\mathrm{S}}}(\eta^*) + R_{\mathcal{D}_{\mathrm{T}}}(\eta^*)\right]$, and

$$R_S(\eta) \;=\; \frac{1}{n}\sum_{i=1}^{m} I\left[\eta(\mathbf{x}_i) \neq y_i\right]$$

is the empirical source risk.

The previous result tells us that $R_{\mathcal{D}_{\mathrm{T}}}(\eta)$ can be low only when the β term is low, *i.e.*, only when there exists a classifier that can achieve a low risk on both distributions. It also tells us that, to find a classifier with a small $R_{\mathcal{D}_{\mathrm{T}}}(\eta)$ in a given class of fixed VC dimension, the learning algorithm should minimize (in that class) a trade-off between the source risk $R_S(\eta)$ and the empirical $\mathcal{H}$-divergence $\hat{d}_{\mathcal{H}}(S,T)$. As pointed-out by Ben-David et al. (2006), a strategy to control the $\mathcal{H}$-divergence is to find a representation of the examples where both the source and the target domain are as indistinguishable as possible. Under such a representation, a hypothesis with a low source risk will, according to Theorem 2, perform well on the target data. In this paper, we present an algorithm that directly exploits this idea.

4. Domain-Adversarial Neural Networks (DANN)

An original aspect of our approach is to explicitly implement the idea exhibited by Theorem 2 into a neural network classifier. That is, to learn a model that can generalize well from one domain to another, we ensure that the internal representation of the neural network contains no discriminative information about the origin of the input (source or target), while preserving a low risk on the source (labeled) examples.

In this section, we detail the proposed approach for incorporating a "domain adaptation component" to neural networks. In Subsection 4.1, we start by developing the idea for the simplest possible case, *i.e.*, a single hidden layer, fully connected neural network. We then describe how to generalize the approach to arbitrary (deep) network architectures.

4.1 Example Case with a Shallow Neural Network

Let us first consider a standard neural network (NN) architecture with a single hidden layer. For simplicity, we suppose that the input space is formed by m-dimensional real

vectors. Thus, $X = \mathbb{R}^m$. The hidden layer G_f learns a function $G_f : X \to \mathbb{R}^D$ that maps an example into a new D-dimensional representation[2], and is parameterized by a matrix-vector pair $(\mathbf{W}, \mathbf{b}) \in \mathbb{R}^{D\times m} \times \mathbb{R}^D$:

$$G_f(\mathbf{x}; \mathbf{W}, \mathbf{b}) \;=\; \mathrm{sigm}\big(\mathbf{W}\mathbf{x} + \mathbf{b}\big)\,, \tag{4}$$

with $\mathrm{sigm}(\mathbf{a}) = \left[\frac{1}{1+\exp(-a_i)}\right]_{i=1}^{|\mathbf{a}|}$.

Similarly, the prediction layer G_y learns a function $G_y : \mathbb{R}^D \to [0,1]^L$ that is parameterized by a pair $(\mathbf{V}, \mathbf{c}) \in \mathbb{R}^{L\times D} \times \mathbb{R}^L$:

$$G_y(G_f(\mathbf{x}); \mathbf{V}, \mathbf{c}) \;=\; \mathrm{softmax}\big(\mathbf{V}G_f(\mathbf{x}) + \mathbf{c}\big)\,,$$

with $\mathrm{softmax}(\mathbf{a}) = \left[\frac{\exp(a_i)}{\sum_{j=1}^{|\mathbf{a}|}\exp(a_j)}\right]_{i=1}^{|\mathbf{a}|}$.

Here we have $L = |Y|$. By using the softmax function, each component of vector $G_y(G_f(\mathbf{x}))$ denotes the conditional probability that the neural network assigns $\mathbf{x}$ to the class in Y represented by that component. Given a source example $(\mathbf{x}_i, y_i)$, the natural classification loss to use is the negative log-probability of the correct label:

$$\mathcal{L}_y\big(G_y(G_f(\mathbf{x}_i)), y_i\big) \;=\; \log\frac{1}{G_y(G_f(\mathbf{x}))_{y_i}}\,.$$

Training the neural network then leads to the following optimization problem on the source domain:

$$\min_{\mathbf{W},\mathbf{b},\mathbf{V},\mathbf{c}} \left[\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_y^i(\mathbf{W},\mathbf{b},\mathbf{V},\mathbf{c}) + \lambda\cdot R(\mathbf{W},\mathbf{b})\right], \tag{5}$$

where $\mathcal{L}_y^i(\mathbf{W},\mathbf{b},\mathbf{V},\mathbf{c}) = \mathcal{L}_y\big(G_y(G_f(\mathbf{x}_i;\mathbf{W},\mathbf{b});\mathbf{V},\mathbf{c}), y_i\big)$ is a shorthand notation for the prediction loss on the i-th example, and $R(\mathbf{W},\mathbf{b})$ is an optional regularizer that is weighted by hyper-parameter λ.

The heart of our approach is to design a *domain regularizer* directly derived from the $\mathcal{H}$-divergence of Definition 1. To this end, we view the output of the hidden layer $G_f(\cdot)$ (Equation 4) as the internal representation of the neural network. Thus, we denote the source sample representations as

$$S(G_f) \;=\; \big\{G_f(\mathbf{x}) \,\big|\, \mathbf{x} \in S\big\}\,.$$

Similarly, given an unlabeled sample from the target domain we denote the corresponding representations

$$T(G_f) \;=\; \big\{G_f(\mathbf{x}) \,\big|\, \mathbf{x} \in T\big\}\,.$$

Based on Equation (1), the empirical $\mathcal{H}$-divergence of a symmetric hypothesis class $\mathcal{H}$ between samples $S(G_f)$ and $T(G_f)$ is given by

$$\hat{d}_{\mathcal{H}}\big(S(G_f), T(G_f)\big) \;=\; 2\left(1 - \min_{\eta\in\mathcal{H}}\left[\frac{1}{n}\sum_{i=1}^{n} I\big[\eta(G_f(\mathbf{x}_i))=0\big] + \frac{1}{n'}\sum_{i=n+1}^{N} I\big[\eta(G_f(\mathbf{x}_i))=1\big]\right]\right). \tag{6}$$

2. For brevity of notation, we will sometimes drop the dependence of G_f on its parameters $(\mathbf{W},\mathbf{b})$ and shorten $G_f(\mathbf{x};\mathbf{W},\mathbf{b})$ to $G_f(\mathbf{x})$.

Let us consider $\mathcal{H}$ as the class of hyperplanes in the representation space. Inspired by the Proxy $\mathcal{A}$-distance (see Section 3.2), we suggest estimating the "min" part of Equation (6) by a *domain classification layer* G_d that learns a logistic regressor $G_d : \mathbb{R}^D \to [0,1]$, parameterized by a vector-scalar pair $(\mathbf{u}, z) \in \mathbb{R}^D \times \mathbb{R}$, that models the probability that a given input is from the source domain $\mathcal{D}_{\mathrm{S}}^X$ or the target domain $\mathcal{D}_{\mathrm{T}}^X$. Thus,

$$G_d(G_f(\mathbf{x}); \mathbf{u}, z) \;=\; \mathrm{sigm}\big(\mathbf{u}^\top G_f(\mathbf{x}) + z\big)\,. \tag{7}$$

Hence, the function $G_d(\cdot)$ is a *domain regressor*. We define its loss by

$$\mathcal{L}_d\big(G_d(G_f(\mathbf{x}_i)), d_i\big) \;=\; d_i \log \frac{1}{G_d(G_f(\mathbf{x}_i))} + (1-d_i) \log \frac{1}{1-G_d(G_f(\mathbf{x}_i))}\,,$$

where d_i denotes the binary variable (*domain label*) for the i-th example, which indicates whether $\mathbf{x}_i$ come from the source distribution ($\mathbf{x}_i \sim \mathcal{D}_{\mathrm{S}}^X$ if $d_i{=}0$) or from the target distribution ($\mathbf{x}_i \sim \mathcal{D}_{\mathrm{T}}^X$ if $d_i{=}1$).

Recall that for the examples from the source distribution ($d_i{=}0$), the corresponding labels $y_i \in Y$ are known at training time. For the examples from the target domains, we do not know the labels at training time, and we want to predict such labels at test time. This enables us to add a domain adaptation term to the objective of Equation (5), giving the following regularizer:

$$R(\mathbf{W}, \mathbf{b}) \;=\; \max_{\mathbf{u}, z} \left[-\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_d^i(\mathbf{W}, \mathbf{b}, \mathbf{u}, z) - \frac{1}{n'} \sum_{i=n+1}^{N} \mathcal{L}_d^i(\mathbf{W}, \mathbf{b}, \mathbf{u}, z) \right], \tag{8}$$

where $\mathcal{L}_d^i(\mathbf{W}, \mathbf{b}, \mathbf{u}, z) {=} \mathcal{L}_d\big(G_d(G_f(\mathbf{x}_i; \mathbf{W}, \mathbf{b}); \mathbf{u}, z), d_i\big)$. This regularizer seeks to approximate the $\mathcal{H}$-divergence of Equation (6), as $2(1{-}R(\mathbf{W}, \mathbf{b}))$ is a surrogate for $\hat{d}_{\mathcal{H}}\big(S(G_f), T(G_f)\big)$. In line with Theorem 2, the optimization problem given by Equations (5) and (8) implements a trade-off between the minimization of the source risk $R_S(\cdot)$ and the divergence $\hat{d}_{\mathcal{H}}(\cdot, \cdot)$. The hyper-parameter λ is then used to tune the trade-off between these two quantities during the learning process.

For learning, we first note that we can rewrite the complete optimization objective of Equation (5) as follows:

$$\begin{aligned} &E(\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}, \mathbf{u}, z) \qquad\qquad (9)\\ &\quad= \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_y^i(\mathbf{W}, \mathbf{b}, \mathbf{V}, \mathbf{c}) - \lambda \left(\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_d^i(\mathbf{W}, \mathbf{b}, \mathbf{u}, z) + \frac{1}{n'} \sum_{i=n+1}^{N} \mathcal{L}_d^i(\mathbf{W}, \mathbf{b}, \mathbf{u}, z) \right), \end{aligned}$$

where we are seeking the parameters $\hat{\mathbf{W}}, \hat{\mathbf{V}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}, \hat{\mathbf{u}}, \hat{z}$ that deliver a saddle point given by

$$\begin{aligned} (\hat{\mathbf{W}}, \hat{\mathbf{V}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}) &= \operatorname*{argmin}_{\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}} E(\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}, \hat{\mathbf{u}}, \hat{z})\,,\\ (\hat{\mathbf{u}}, \hat{z}) &= \operatorname*{argmax}_{\mathbf{u}, z} E(\hat{\mathbf{W}}, \hat{\mathbf{V}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}, \mathbf{u}, z)\,. \end{aligned}$$

Thus, the optimization problem involves a minimization with respect to some parameters, as well as a maximization with respect to the others.

Algorithm 1 Shallow DANN – Stochastic training update

Input:
— samples $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ and $T = \{\mathbf{x}_i\}_{i=1}^{n'}$,
— hidden layer size D,
— adaptation parameter λ,
— learning rate μ,
Output: neural network $\{\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}\}$

$\mathbf{W}, \mathbf{V} \leftarrow$ random_init(D)
$\mathbf{b}, \mathbf{c}, \mathbf{u}, d \leftarrow 0$
while stopping criterion is not met **do**
  for i from 1 to n **do**
    `# Forward propagation`
    $G_f(\mathbf{x}_i) \leftarrow \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}_i)$
    $G_y(G_f(\mathbf{x}_i)) \leftarrow \mathrm{softmax}(\mathbf{c} + \mathbf{V} G_f(\mathbf{x}_i))$
    `# Backpropagation`
    $\Delta_{\mathbf{c}} \leftarrow -(\mathbf{e}(y_i) - G_y(G_f(\mathbf{x}_i)))$
    $\Delta_{\mathbf{V}} \leftarrow \Delta_{\mathbf{c}}\, G_f(\mathbf{x}_i)^\top$
    $\Delta_{\mathbf{b}} \leftarrow \big(\mathbf{V}^\top \Delta_{\mathbf{c}}\big) \odot G_f(\mathbf{x}_i) \odot (1 - G_f(\mathbf{x}_i))$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{b}} \cdot (\mathbf{x}_i)^\top$
    `# Domain adaptation regularizer...`
    `# ...from current domain`
    $G_d(G_f(\mathbf{x}_i)) \leftarrow \mathrm{sigm}(d + \mathbf{u}^\top G_f(\mathbf{x}_i))$
    $\Delta_d \leftarrow \lambda(1 - G_d(G_f(\mathbf{x}_i)))$
    $\Delta_{\mathbf{u}} \leftarrow \lambda(1 - G_d(G_f(\mathbf{x}_i)))G_f(\mathbf{x}_i)$
    $\mathrm{tmp} \leftarrow \lambda(1 - G_d(G_f(\mathbf{x}_i))) \times \mathbf{u} \odot G_f(\mathbf{x}_i) \odot (1 - G_f(\mathbf{x}_i))$
    $\Delta_{\mathbf{b}} \leftarrow \Delta_{\mathbf{b}} + \mathrm{tmp}$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{W}} + \mathrm{tmp} \cdot (\mathbf{x}_i)^\top$
    `# ...from other domain`
    $j \leftarrow$ uniform_integer$(1, \ldots, n')$
    $G_f(\mathbf{x}_j) \leftarrow \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}_j)$
    $G_d(G_f(\mathbf{x}_j)) \leftarrow \mathrm{sigm}(d + \mathbf{u}^\top G_f(\mathbf{x}_j))$
    $\Delta_d \leftarrow \Delta_d - \lambda G_d(G_f(\mathbf{x}_j))$
    $\Delta_{\mathbf{u}} \leftarrow \Delta_{\mathbf{u}} - \lambda G_d(G_f(\mathbf{x}_j))G_f(\mathbf{x}_j)$
    $\mathrm{tmp} \leftarrow -\lambda G_d(G_f(\mathbf{x}_j)) \times \mathbf{u} \odot G_f(\mathbf{x}_j) \odot (1 - G_f(\mathbf{x}_j))$
    $\Delta_{\mathbf{b}} \leftarrow \Delta_{\mathbf{b}} + \mathrm{tmp}$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{W}} + \mathrm{tmp} \cdot (\mathbf{x}_j)^\top$
    `# Update neural network parameters`
    $\mathbf{W} \leftarrow \mathbf{W} - \mu \Delta_{\mathbf{W}}$
    $\mathbf{V} \leftarrow \mathbf{V} - \mu \Delta_{\mathbf{V}}$
    $\mathbf{b} \leftarrow \mathbf{b} - \mu \Delta_{\mathbf{b}}$
    $\mathbf{c} \leftarrow \mathbf{c} - \mu \Delta_{\mathbf{c}}$
    `# Update domain classifier`
    $\mathbf{u} \leftarrow \mathbf{u} + \mu \Delta_{\mathbf{u}}$
    $d \leftarrow d + \mu \Delta_d$
  end for
end while

Note: In this pseudo-code, $\mathbf{e}(y)$ refers to a "one-hot" vector, consisting of all 0s except for a 1 at position y, and $\odot$ is the element-wise product.

We propose to tackle this problem with a simple stochastic gradient procedure, in which updates are made in the opposite direction of the gradient of Equation (9) for the minimizing parameters, and in the direction of the gradient for the maximizing parameters. Stochastic estimates of the gradient are made, using a subset of the training samples to compute the averages. Algorithm 1 provides the complete pseudo-code of this learning procedure.[3] In words, during training, the neural network (parameterized by $\mathbf{W}, \mathbf{b}, \mathbf{V}, \mathbf{c}$) and the domain regressor (parameterized by $\mathbf{u}, z$) are competing against each other, in an adversarial way, over the objective of Equation (9). For this reason, we refer to networks trained according to this objective as Domain-Adversarial Neural Networks (DANN). DANN will effectively attempt to learn a hidden layer $G_f(\cdot)$ that maps an example (either source or target) into a representation allowing the output layer $G_y(\cdot)$ to accurately classify source samples, but crippling the ability of the domain regressor $G_d(\cdot)$ to detect whether each example belongs to the source or target domains.

3. We provide an implementation of *Shallow DANN* algorithm at `http://graal.ift.ulaval.ca/dann/`

4.2 Generalization to Arbitrary Architectures

For illustration purposes, we've so far focused on the case of a single hidden layer DANN. However, it is straightforward to generalize to other sophisticated architectures, which might be more appropriate for the data at hand. For example, deep convolutional neural networks are well known for being state-of-the-art models for learning discriminative features of images (Krizhevsky et al., 2012).

Let us now use a more general notation for the different components of DANN. Namely, let $G_f(\cdot;\theta_f)$ be the D-dimensional neural network feature extractor, with parameters θ_f. Also, let $G_y(\cdot;\theta_y)$ be the part of DANN that computes the network's label prediction output layer, with parameters θ_y, while $G_d(\cdot;\theta_d)$ now corresponds to the computation of the domain prediction output of the network, with parameters θ_d. Note that for preserving the theoretical guarantees of Theorem 2, the hypothesis class $\mathcal{H}_d$ generated by the domain prediction component G_d should include the hypothesis class $\mathcal{H}_y$ generated by the label prediction component G_y. Thus, $\mathcal{H}_y \subseteq \mathcal{H}_d$.

We will note the prediction loss and the domain loss respectively by

$$\begin{aligned} \mathcal{L}_y^i(\theta_f,\theta_y) &= \mathcal{L}_y\big(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y), y_i\big)\,, \\ \mathcal{L}_d^i(\theta_f,\theta_d) &= \mathcal{L}_d\big(G_d(G_f(\mathbf{x}_i;\theta_f);\theta_d), d_i\big)\,. \end{aligned}$$

Training DANN then parallels the single layer case and consists in optimizing

$$E(\theta_f,\theta_y,\theta_d) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_y^i(\theta_f,\theta_y) - \lambda\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_d^i(\theta_f,\theta_d) + \frac{1}{n'}\sum_{i=n+1}^{N}\mathcal{L}_d^i(\theta_f,\theta_d)\right), \tag{10}$$

by finding the saddle point $\hat{\theta}_f,\hat{\theta}_y,\hat{\theta}_d$ such that

$$(\hat{\theta}_f,\hat{\theta}_y) = \operatorname*{argmin}_{\theta_f,\theta_y} E(\theta_f,\theta_y,\hat{\theta}_d)\,, \tag{11}$$

$$\hat{\theta}_d = \operatorname*{argmax}_{\theta_d} E(\hat{\theta}_f,\hat{\theta}_y,\theta_d)\,. \tag{12}$$

As suggested previously, a saddle point defined by Equations (11-12) can be found as a stationary point of the following gradient updates:

$$\theta_f \longleftarrow \theta_f - \mu\left(\frac{\partial\mathcal{L}_y^i}{\partial\theta_f} - \lambda\frac{\partial\mathcal{L}_d^i}{\partial\theta_f}\right), \tag{13}$$

$$\theta_y \longleftarrow \theta_y - \mu\frac{\partial\mathcal{L}_y^i}{\partial\theta_y}, \tag{14}$$

$$\theta_d \longleftarrow \theta_d - \mu\lambda\frac{\partial\mathcal{L}_d^i}{\partial\theta_d}, \tag{15}$$

where μ is the learning rate. We use stochastic estimates of these gradients, by sampling examples from the data set.

The updates of Equations (13-15) are very similar to stochastic gradient descent (SGD) updates for a feed-forward deep model that comprises feature extractor fed into the label

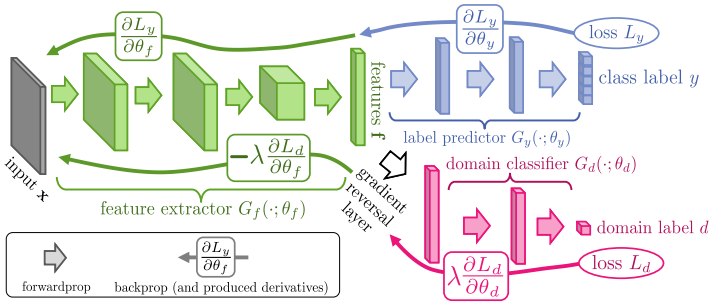


Figure 1: The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds standardly and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

predictor and into the domain classifier (with loss weighted by λ). The only difference is that in (13), the gradients from the class and domain predictors are subtracted, instead of being summed (the difference is important, as otherwise SGD would try to make features dissimilar across domains in order to minimize the domain classification loss). Since SGD—and its many variants, such as ADAGRAD (Duchi et al., 2010) or ADADELTA (Zeiler, 2012)—is the main learning algorithm implemented in most libraries for deep learning, it would be convenient to frame an implementation of our stochastic saddle point procedure as SGD.

Fortunately, such a reduction can be accomplished by introducing a special *gradient reversal layer* (GRL), defined as follows. The gradient reversal layer has no parameters associated with it. During the forward propagation, the GRL acts as an identity transformation. During the backpropagation however, the GRL takes the gradient from the subsequent level and changes its sign, *i.e.*, multiplies it by -1, before passing it to the preceding layer. Implementing such a layer using existing object-oriented packages for deep learning is simple, requiring only to define procedures for the forward propagation (identity transformation), and backpropagation (multiplying by -1). The layer requires no parameter update.

The GRL as defined above is inserted between the feature extractor G_f and the domain classifier G_d, resulting in the architecture depicted in Figure 1. As the backpropagation process passes through the GRL, the partial derivatives of the loss that is downstream

the GRL (*i.e.*, $\mathcal{L}_d$) w.r.t. the layer parameters that are upstream the GRL (*i.e.*, θ_f) get multiplied by -1, *i.e.*, $\frac{\partial \mathcal{L}_d}{\partial \theta_f}$ is effectively replaced with $-\frac{\partial \mathcal{L}_d}{\partial \theta_f}$. Therefore, running SGD in the resulting model implements the updates of Equations (13-15) and converges to a saddle point of Equation (10).

Mathematically, we can formally treat the gradient reversal layer as a "pseudo-function" $\mathcal{R}(\mathbf{x})$ defined by two (incompatible) equations describing its forward and backpropagation behaviour:

$$\mathcal{R}(\mathbf{x}) = \mathbf{x}\,, \tag{16}$$

$$\frac{d\mathcal{R}}{d\mathbf{x}} = -\mathbf{I}\,, \tag{17}$$

where $\mathbf{I}$ is an identity matrix. We can then define the objective "pseudo-function" of $(\theta_f, \theta_y, \theta_d)$ that is being optimized by the stochastic gradient descent within our method:

$$\tilde{E}(\theta_f, \theta_y, \theta_d) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_y\left(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y), y_i\right) \tag{18}$$

$$-\,\lambda\left(\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_d\left(G_d(\mathcal{R}(G_f(\mathbf{x}_i;\theta_f));\theta_d), d_i\right) + \frac{1}{n'}\sum_{i=n+1}^{N} \mathcal{L}_d\left(G_d(\mathcal{R}(G_f(\mathbf{x}_i;\theta_f));\theta_d), d_i\right)\right).$$

Running updates (13-15) can then be implemented as doing SGD for (18) and leads to the emergence of features that are domain-invariant and discriminative at the same time. After the learning, the label predictor $G_y(G_f(\mathbf{x};\theta_f);\theta_y)$ can be used to predict labels for samples from the target domain (as well as from the source domain). Note that we release the source code for the Gradient Reversal layer along with the usage examples as an extension to `Caffe` (Jia et al., 2014).[4]

5. Experiments

In this section, we present a variety of empirical results for both *shallow* domain adversarial neural networks (Subsection 5.1) and *deep* ones (Subsections 5.2 and 5.3).

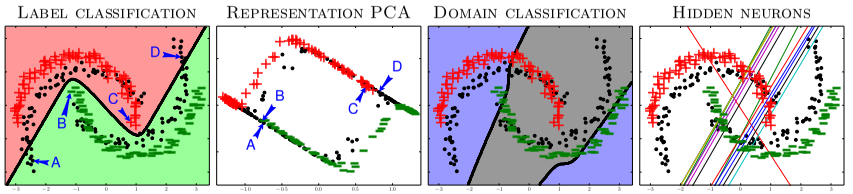
5.1 Experiments with Shallow Neural Networks

In this first experiment section, we evaluate the behavior of the simple version of DANN described by Subsection 4.1. Note that the results reported in the present subsection are obtained using Algorithm 1. Thus, the stochastic gradient descent approach here consists of sampling a pair of source and target examples and performing a gradient step update of all parameters of DANN. Crucially, while the update of the regular parameters follows as usual the opposite direction of the gradient, for the adversarial parameters the step must follow the gradient's direction (since we maximize with respect to them, instead of minimizing).

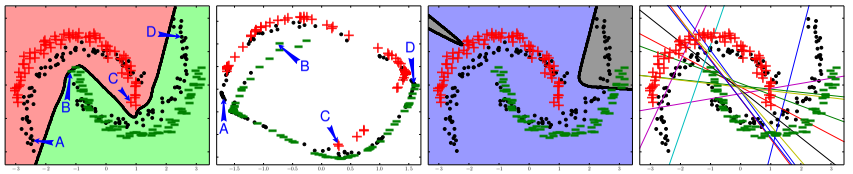
5.1.1 Experiments on a Toy Problem

As a first experiment, we study the behavior of the proposed algorithm on a variant of the *inter-twinning moons* 2D problem, where the target distribution is a rotation of the source

4. `http://sites.skoltech.ru/compvision/projects/grl/`



(a) Standard NN. For the "domain classification", we use a *non adversarial* domain regressor on the hidden neurons learned by the Standard NN. (This is equivalent to run Algorithm 1, without Lines 22 and 31)



(b) DANN (Algorithm 1)

Figure 2: The *inter-twinning moons* toy problem. Examples from the source sample are represented as a "+"(label 1) and a "−"(label 0), while examples from the unlabeled target sample are represented as black dots. See text for the figure discussion.

one. As the source sample S, we generate a lower moon and an upper moon labeled 0 and 1 respectively, each of which containing 150 examples. The target sample T is obtained by the following procedure: (1) we generate a sample S' the same way S has been generated; (2) we rotate each example by 35°; and (3) we remove all the labels. Thus, T contains 300 unlabeled examples. We have represented those examples in Figure 2.

We study the adaptation capability of DANN by comparing it to the standard neural network (NN). In these toy experiments, both algorithms share the same network architecture, with a hidden layer size of 15 neurons. We train the NN using the same procedure as the DANN. That is, we keep updating the domain regressor component using target sample T (with a hyper-parameter $\lambda = 6$; the same value is used for DANN), but we disable the *adversarial* back-propagation into the hidden layer. To do so, we execute Algorithm 1 by omitting the lines numbered 22 and 31. This allows recovering the NN learning algorithm—based on the source risk minimization of Equation (5) without any regularizer—and simultaneously train the domain regressor of Equation (7) to discriminate between source and target domains. With this toy experience, we will first illustrate how DANN adapts its decision boundary when compared to NN. Moreover, we will also illustrate how the representation given by the hidden layer is less adapted to the source domain task with DANN than with NN (this is why we need a domain regressor in the NN experiment). We recall that this is the founding idea behind our proposed algorithm. The analysis of the experiment appears in Figure 2, where upper graphs relate to standard NN, and lower graphs relate to DANN. By looking at the lower and upper graphs pairwise, we compare NN and DANN from four different perspectives, described in details below.

The column "Label Classification" of Figure 2 shows the decision boundaries of DANN and NN on the problem of predicting the labels of both source and the target examples. As expected, NN accurately classifies the two classes of the source sample S, but is *not fully adapted* to the target sample T. On the contrary, the decision boundary of DANN perfectly classifies examples from both source and target samples. In the studied task, DANN clearly adapts to the target distribution.

The column "Representation PCA" studies how the domain adaptation regularizer affects the representation $G_f(\cdot)$ provided by the network hidden layer. The graphs are obtained by applying a Principal component analysis (PCA) on the set of all representation of source and target data points, *i.e.*, $S(G_f) \cup T(G_f)$. Thus, given the trained network (NN or DANN), every point from S and T is mapped into a 15-dimensional feature space through the hidden layer, and projected back into a two-dimensional plane by the PCA transformation. In the DANN-PCA representation, we observe that target points are homogeneously spread out among source points; In the NN-PCA representation, a number of target points belong to clusters containing no source points. Hence, labeling the target points seems an easier task given the DANN-PCA representation.
To push the analysis further, the PCA graphs tag four crucial data points by the letters A, B, C and D, that correspond to the moon extremities in the original space (note that the original point locations are tagged in the first column graphs). We observe that points A and B are very close to each other in the NN-PCA representation, while they clearly belong to different classes. The same happens to points C and D. Conversely, these four points are at the opposite four corners in the DANN-PCA representation. Note also that the target point A (resp. D)—that is difficult to classify in the original space—is located in the "+"cluster (resp. "−"cluster) in the DANN-PCA representation. Therefore, the representation promoted by DANN is better suited to the adaptation problem.

The column "Domain Classification" shows the decision boundary on the domain classification problem, which is given by the domain regressor G_d of Equation (7). More precisely, an example $\mathbf{x}$ is classified as a source example when $G_d(G_f(\mathbf{x})) \geq 0.5$, and is classified as a domain example otherwise. Remember that, during the learning process of DANN, the G_d regressor struggles to discriminate between source and target domains, while the hidden representation $G_f(\cdot)$ is *adversarially* updated to prevent it to succeed. As explained above, we trained a domain regressor during the learning process of NN, but without allowing it to influence the learned representation $G_f(\cdot)$.
On one hand, the DANN domain regressor clearly fails to generalize source and target distribution topologies. On the other hand, the NN domain regressor shows a better (although imperfect) generalization capability. Inter alia, it seems to roughly capture the rotation angle of the target distribution. This again corroborates that the DANN representation does not allow discriminating between domains.

The column "Hidden Neurons" shows the configuration of hidden layer neurons (by Equation 4, we have that each neuron is indeed a linear regressor). In other words, each of the fifteen plot line corresponds to the coordinates $\mathbf{x} \in \mathbb{R}^2$ for which the i-th component of $G_f(\mathbf{x})$ equals $\frac{1}{2}$, for $i \in \{1, \ldots, 15\}$. We observe that the standard NN neurons are grouped in three clusters, each one allowing to generate a straight line of the *zigzag* decision boundary for the label classification problem. However, most of these neurons are also able

to (roughly) capture the rotation angle of the domain classification problem. Hence, we observe that the adaptation regularizer of DANN prevents these kinds of neurons to be produced. It is indeed striking to see that the two predominant patterns in the NN neurons (*i.e.*, the two parallel lines crossing the plane from lower left to upper right) are vanishing in the DANN neurons.

5.1.2 Unsupervised Hyper-Parameter Selection

To perform unsupervised domain adaption, one should provide ways to set hyper-parameters (such as the domain regularization parameter λ, the learning rate, the network architecture for our method) in an unsupervised way, *i.e.*, without referring to labeled data in the target domain. In the following experiments of Sections 5.1.3 and 5.1.4, we select the hyper-parameters of each algorithm by using a variant of *reverse cross-validation* approach proposed by Zhong et al. (2010), that we call *reverse validation*.

To evaluate the *reverse validation risk* associated to a tuple of hyper-parameters, we proceed as follows. Given the labeled source sample S and the unlabeled target sample T, we split each set into training sets (S' and T' respectively, containing 90% of the original examples) and the validation sets (S_V and T_V respectively). We use the labeled set S' and the unlabeled target set T' to learn a classifier η. Then, using the same algorithm, we learn a *reverse* classifier η_r using the *self-labeled* set $\{(\mathbf{x}, \eta(\mathbf{x}))\}_{\mathbf{x} \in T'}$ and the unlabeled part of S' as target sample. Finally, the reverse classifier η_r is evaluated on the validation set S_V of source sample. We then say that the classifier η has a *reverse validation risk* of $R_{S_V}(\eta_r)$. The process is repeated with multiple values of hyper-parameters and the selected parameters are those corresponding to the classifier with the lowest reverse validation risk.

Note that when we train neural network architectures, the validation set S_V is also used as an early stopping criterion during the learning of η, and *self-labeled* validation set $\{(\mathbf{x}, \eta(\mathbf{x}))\}_{\mathbf{x} \in T_V}$ is used as an early stopping criterion during the learning of η_r. We also observed better accuracies when we initialized the learning of the reverse classifier η_r with the configuration learned by the network η.

5.1.3 Experiments on Sentiment Analysis Data Sets

We now compare the performance of our proposed DANN algorithm to a standard neural network with one hidden layer (NN) described by Equation (5), and a Support Vector Machine (SVM) with a linear kernel. We compare the algorithms on the *Amazon reviews* data set, as pre-processed by Chen et al. (2012). This data set includes four domains, each one composed of reviews of a specific kind of product (books, dvd disks, electronics, and kitchen appliances). Reviews are encoded in 5 000 dimensional feature vectors of unigrams and bigrams, and labels are binary: "0" if the product is ranked up to 3 stars, and "1" if the product is ranked 4 or 5 stars.

We perform twelve domain adaptation tasks. All learning algorithms are given 2 000 labeled source examples and 2 000 unlabeled target examples. Then, we evaluate them on separate target test sets (between 3 000 and 6 000 examples). Note that NN and SVM do not use the unlabeled target sample for learning.

Here are more details about the procedure used for each learning algorithms leading to the empirical results of Table 1.

		Original data			mSDA representation		
Source	Target	DANN	NN	SVM	DANN	NN	SVM
books	dvd	.784	.790	**.799**	.829	.824	**.830**
books	electronics	.733	.747	**.748**	**.804**	.770	.766
books	kitchen	**.779**	.778	.769	**.843**	.842	.821
dvd	books	.723	.720	**.743**	.825	.823	**.826**
dvd	electronics	**.754**	.732	.748	**.809**	.768	.739
dvd	kitchen	**.783**	.778	.746	.849	**.853**	.842
electronics	books	**.713**	.709	.705	**.774**	.770	.762
electronics	dvd	**.738**	.733	.726	**.781**	.759	.770
electronics	kitchen	**.854**	**.854**	.847	.881	**.863**	.847
kitchen	books	**.709**	.708	.707	.718	.721	**.769**
kitchen	dvd	**.740**	.739	.736	**.789**	**.789**	.788
kitchen	electronics	**.843**	.841	.842	.856	.850	**.861**

(a) Classification accuracy on the Amazon reviews data set

Original data				mSDA representations			
	DANN	NN	SVM		DANN	NN	SVM
DANN	.50	**.87**	**.83**	DANN	.50	**.92**	**.88**
NN	.13	.50	.63	NN	.08	.50	.62
SVM	.17	.37	.50	SVM	.12	.38	.50

(b) Pairwise Poisson binomial test

Table 1: Classification accuracy on the Amazon reviews data set, and Pairwise Poisson binomial test.

- For the DANN algorithm, the adaptation parameter λ is chosen among 9 values between 10^{-2} and 1 on a logarithmic scale. The hidden layer size l is either 50 or 100. Finally, the learning rate μ is fixed at 10^{-3}.
- For the NN algorithm, we use exactly the same hyper-parameters grid and training procedure as DANN above, except that we do not need an adaptation parameter. Note that one can train NN by using the DANN implementation (Algorithm 1) with $\lambda = 0$.
- For the SVM algorithm, the hyper-parameter C is chosen among 10 values between 10^{-5} and 1 on a logarithmic scale. This range of values is the same as used by Chen et al. (2012) in their experiments.

As presented at Section 5.1.2, we used *reverse cross validation* selecting the hyper-parameters for all three learning algorithms, with *early stopping* as the stopping criterion for DANN and NN.

The "Original data" part of Table 1a shows the target test accuracy of all algorithms, and Table 1b reports the probability that one algorithm is significantly better than the others according to the Poisson binomial test (Lacoste et al., 2012). We note that DANN has a significantly better performance than NN and SVM, with respective probabilities **0.87** and **0.83**. As the only difference between DANN and NN is the domain adaptation regularizer, we conclude that our approach successfully helps to find a representation suitable for the target domain.

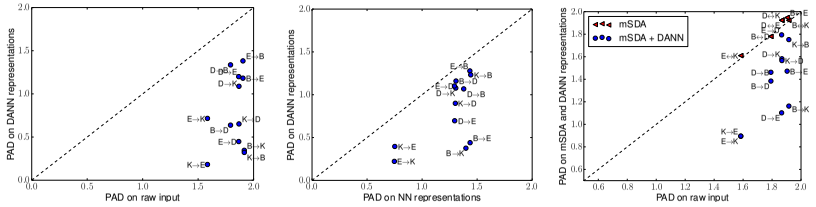
5.1.4 Combining DANN with Denoising Autoencoders

We now investigate on whether the DANN algorithm can improve on the representation learned by the state-of-the-art *Marginalized Stacked Denoising Autoencoders* (mSDA) proposed by Chen et al. (2012). In brief, mSDA is an unsupervised algorithm that learns a new robust feature representation of the training samples. It takes the unlabeled parts of both source and target samples to learn a feature map from input space X to a new representation space. As a *denoising autoencoders* algorithm, it finds a feature representation from which one can (approximately) reconstruct the original features of an example from its noisy counterpart. Chen et al. (2012) showed that using mSDA with a linear SVM classifier reaches state-of-the-art performance on the *Amazon reviews* data sets. As an alternative to the SVM, we propose to apply our Shallow DANN algorithm on the same representations generated by mSDA (using representations of both source and target samples). Note that, even if mSDA and DANN are two representation learning approaches, they optimize different objectives, which can be complementary.

We perform this experiment on the same *Amazon reviews* data set described in the previous subsection. For each source-target domain pair, we generate the mSDA representations using a corruption probability of 50% and a number of layers of 5. We then execute the three learning algorithms (DANN, NN, and SVM) on these representations. More precisely, following the experimental procedure of Chen et al. (2012), we use the concatenation of the output of the 5 layers and the original input as the new representation. Thus, each example is now encoded in a vector of 30 000 dimensions. Note that we use the same grid search as in the previous Subsection 5.1.3, but use a learning rate μ of 10^{-4} for both DANN and the NN. The results of "mSDA representation" columns in Table 1a confirm that combining mSDA and DANN is a sound approach. Indeed, the Poisson binomial test shows that DANN has a better performance than the NN and the SVM, with probabilities **0.92** and **0.88** respectively, as reported in Table 1b. We note however that the standard NN and the SVM find the best solution on respectively the second and the fourth tasks. This suggests that DANN and mSDA adaptation strategies are not fully complementary.

5.1.5 Proxy Distance

The theoretical foundation of the DANN algorithm is the domain adaptation theory of Ben-David et al. (2006, 2010). We claimed that DANN finds a representation in which the source and the target example are hardly distinguishable. Our toy experiment of Section 5.1.1 already points out some evidence for that and here we provide analysis on real data. To do so, we compare the Proxy $\mathcal{A}$-distance (PAD) on various representations of the *Amazon Reviews* data set; these representations are obtained by running either NN, DANN, mSDA,



(a) DANN on *Original data*. (b) DANN & NN with 100 hidden neurons. (c) DANN on *mSDA representations*.

Figure 3: Proxy $\mathcal{A}$-distances (PAD). Note that the PAD values of mSDA representations are symmetric when swapping source and target samples.

or mSDA and DANN combined. Recall that PAD, as described in Section 3.2, is a metric estimating the similarity of the source and the target representations. More precisely, to obtain a PAD value, we use the following procedure: (1) we construct the data set U of Equation (2) using both source and target representations of the training samples; (2) we randomly split U in two subsets of equal size; (3) we train linear SVMs on the first subset of U using a large range of C values; (4) we compute the error of all obtained classifiers on the second subset of U; and (5) we use the lowest error to compute the PAD value of Equation (3).

Firstly, Figure 3a compares the PAD of DANN representations obtained in the experiments of Section 5.1.3 (using the hyper-parameters values leading to the results of Table 1) to the PAD computed on raw data. As expected, the PAD values are driven down by the DANN representations.

Secondly, Figure 3b compares the PAD of DANN representations to the PAD of standard NN representations. As the PAD is influenced by the hidden layer size (the discriminating power tends to increase with the representation length), we fix here the size to 100 neurons for both algorithms. We also fix the adaptation parameter of DANN to $\lambda \simeq 0.31$; it was the value that has been selected most of the time during our preceding experiments on the *Amazon Reviews* data set. Again, DANN is clearly leading to the lowest PAD values.

Lastly, Figure 3c presents two sets of results related to Section 5.1.4 experiments. On one hand, we reproduce the results of Chen et al. (2012), which noticed that the mSDA representations have greater PAD values than original (raw) data. Although the mSDA approach clearly helps to adapt to the target task, it seems to contradict the theory of Ben-David et al.. On the other hand, we observe that, when running DANN on top of mSDA (using the hyper-parameters values leading to the results of Table 1), the obtained representations have much lower PAD values. These observations might explain the improvements provided by DANN when combined with the mSDA procedure.

5.2 Experiments with Deep Networks on Image Classification

We now perform extensive evaluation of a deep version of DANN (see Subsection 4.2) on a number of popular image data sets and their modifications. These include large-scale data sets of small images popular with deep learning methods, and the Office data sets (Saenko et al., 2010), which are a *de facto* standard for domain adaptation in computer vision, but have much fewer images.

5.2.1 Baselines

The following baselines are evaluated in the experiments of this subsection. The *source-only* model is trained without consideration for target-domain data (no domain classifier branch included into the network). The *train-on-target* model is trained on the target domain with class labels revealed. This model serves as an upper bound on DA methods, assuming that target data are abundant and the shift between the domains is considerable.

In addition, we compare our approach against the recently proposed unsupervised DA method based on *subspace alignment (SA)* (Fernando et al., 2013), which is simple to setup and test on new data sets, but has also been shown to perform very well in experimental comparisons with other "shallow" DA methods. To boost the performance of this baseline, we pick its most important free parameter (the number of principal components) from the range $\{2, \ldots, 60\}$, so that the test performance on the target domain is maximized. To apply SA in our setting, we train a source-only model and then consider the activations of the last hidden layer in the label predictor (before the final linear classifier) as descriptors/features, and learn the mapping between the source and the target domains (Fernando et al., 2013).

Since the SA baseline requires training a new classifier after adapting the features, and in order to put all the compared settings on an equal footing, we retrain the last layer of the label predictor using a standard linear SVM (Fan et al., 2008) for all four considered methods (including ours; the performance on the target domain remains approximately the same after the retraining).

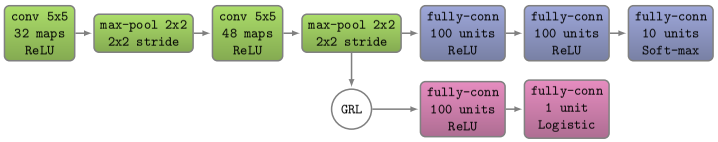
For the Office data set (Saenko et al., 2010), we directly compare the performance of our full network (feature extractor and label predictor) against recent DA approaches using previously published results.

5.2.2 CNN architectures and Training Procedure

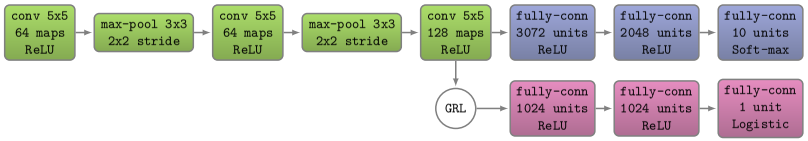
In general, we compose feature extractor from two or three convolutional layers, picking their exact configurations from previous works. More precisely, four different architectures were used in our experiments. The first three are shown in Figure 4. For the Office domains, we use pre-trained `AlexNet` from the `Caffe`-package (Jia et al., 2014). The adaptation architecture is identical to Tzeng et al. (2014).[5]

For the domain adaption component, we use three ($x \to 1024 \to 1024 \to 2$) fully connected layers, except for MNIST where we used a simpler ($x \to 100 \to 2$) architecture to speed up the experiments. Admittedly these choices for domain classifier are arbitrary, and better adaptation performance might be attained if this part of the architecture is tuned.

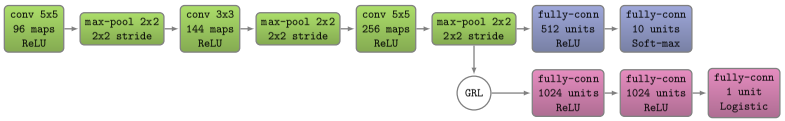
5. A 2-layer domain classifier ($x \to 1024 \to 1024 \to 2$) is attached to the 256-dimensional bottleneck of `fc7`.



(a) MNIST architecture; inspired by the classical LeNet-5 (LeCun et al., 1998).



(b) SVHN architecture; adopted from Srivastava et al. (2014).



(c) GTSRB architecture; we used the single-CNN baseline from Cireşan et al. (2012) as our starting point.

Figure 4: CNN architectures used in the experiments. Boxes correspond to transformations applied to the data. Color-coding is the same as in Figure 1.

For the loss functions, we set $\mathcal{L}_y$ and $\mathcal{L}_d$ to be the logistic regression loss and the binomial cross-entropy respectively. Following Srivastava et al. (2014) we also use dropout and ℓ_2-norm restriction when we train the SVHN architecture.

The other hyper-parameters are not selected through a grid search as in the small scale experiments of Section 5.1, which would be computationally costly. Instead, the learning rate is adjusted during the stochastic gradient descent using the following formula:

$$\mu_p = \frac{\mu_0}{(1+\alpha\cdot p)^\beta}\,,$$

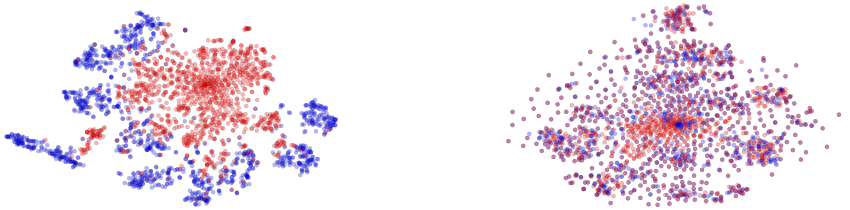
where p is the training progress linearly changing from 0 to 1, $\mu_0 = 0.01$, $\alpha = 10$ and $\beta = 0.75$ (the schedule was optimized to promote convergence and low error on the *source* domain). A momentum term of 0.9 is also used.

The domain adaptation parameter λ is initiated at 0 and is gradually changed to 1 using the following schedule:

$$\lambda_p = \frac{2}{1+\exp(-\gamma\cdot p)} - 1\,,$$

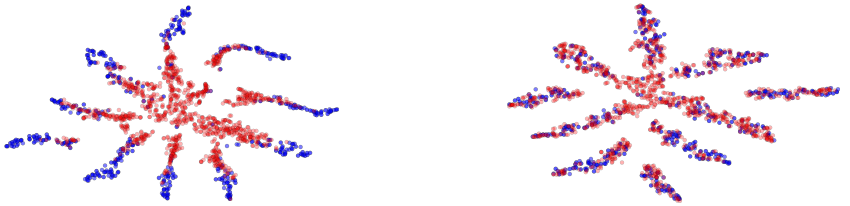
where γ was set to 10 in all experiments (the schedule was not optimized/tweaked). This strategy allows the domain classifier to be less sensitive to noisy signal at the early stages of the training procedure. Note however that these λ_p were used only for updating the *feature*

MNIST → MNIST-M: top feature extractor layer



(a) Non-adapted (b) Adapted

Syn Numbers → SVHN: last hidden layer of the label predictor



(a) Non-adapted (b) Adapted

Figure 5: The effect of adaptation on the distribution of the extracted features (best viewed in color). The figure shows t-SNE (van der Maaten, 2013) visualizations of the CNN's activations **(a)** in case when no adaptation was performed and **(b)** in case when our adaptation procedure was incorporated into training. *Blue* points correspond to the source domain examples, while *red* ones correspond to the target domain. In all cases, the adaptation in our method makes the two distributions of features much closer.

extractor component G_f. For updating the *domain classification* component, we used a fixed $\lambda = 1$, to ensure that the latter trains as fast as the *label predictor* G_y.[6]

Finally, note that the model is trained on 128-sized batches (images are preprocessed by the mean subtraction). A half of each batch is populated by the samples from the source domain (with known labels), the rest constitutes the target domain (with labels not revealed to the algorithms except for the train-on-target baseline).

5.2.3 Visualizations

We use t-SNE (van der Maaten, 2013) projection to visualize feature distributions at different points of the network, while color-coding the domains (Figure 5). As we already observed with the shallow version of DANN (see Figure 2), there is a strong correspondence

6. Equivalently, one can use the same λ_p for both feature extractor and domain classification components, but use a learning rate of μ/λ_p for the latter.

between the success of the adaptation in terms of the classification accuracy for the target domain, and the overlap between the domain distributions in such visualizations.

5.2.4 Results On Image Data Sets

We now discuss the experimental settings and the results. In each case, we train on the source data set and test on a different target domain data set, with considerable shifts between domains (see Figure 6). The results are summarized in Table 2 and Table 3.

MNIST → MNIST-M. Our first experiment deals with the MNIST data set (LeCun et al., 1998) (source). In order to obtain the target domain (MNIST-M) we blend digits from the original set over patches randomly extracted from color photos from BSDS500 (Arbelaez et al., 2011). This operation is formally defined for two images I^1, I^2 as $I^{out}_{ijk} = |I^1_{ijk} - I^2_{ijk}|$, where i, j are the coordinates of a pixel and k is a channel index. In other words, an output sample is produced by taking a patch from a photo and inverting its pixels at positions corresponding to the pixels of a digit. For a human the classification task becomes only slightly harder compared to the original data set (the digits are still clearly distinguishable) whereas for a CNN trained on MNIST this domain is quite distinct, as the background and the strokes are no longer constant. Consequently, the source-only model performs poorly. Our approach succeeded at aligning feature distributions (Figure 5), which led to successful adaptation results (considering that the adaptation is unsupervised). At the same time, the improvement over source-only model achieved by subspace alignment (SA) (Fernando et al., 2013) is quite modest, thus highlighting the difficulty of the adaptation task.

Synthetic numbers → SVHN. To address a common scenario of training on synthetic data and testing on real data, we use Street-View House Number data set SVHN (Netzer et al., 2011) as the target domain and synthetic digits as the source. The latter (Syn Numbers) consists of ≈ 500,000 images generated by ourselves from Windows™ fonts by varying the text (that includes different one-, two-, and three-digit numbers), positioning, orientation, background and stroke colors, and the amount of blur. The degrees of variation were chosen manually to simulate SVHN, however the two data sets are still rather distinct, the biggest difference being the structured clutter in the background of SVHN images.

The proposed backpropagation-based technique works well covering almost 80% of the gap between training with source data only and training on target domain data with known target labels. In contrast, SA (Fernando et al., 2013) results in a slight classification accuracy drop (probably due to the information loss during the dimensionality reduction), indicating that the adaptation task is even more challenging than in the case of the MNIST experiment.

MNIST ↔ SVHN. In this experiment, we further increase the gap between distributions, and test on MNIST and SVHN, which are significantly different in appearance. Training on SVHN even without adaptation is challenging — classification error stays high during the first 150 epochs. In order to avoid ending up in a poor local minimum we, therefore, do not use learning rate annealing here. Obviously, the two directions (MNIST → SVHN and SVHN → MNIST) are not equally difficult. As SVHN is more diverse, a model trained on SVHN is expected to be more generic and to perform reasonably on the MNIST data set. This, indeed, turns out to be the case and is supported by the appearance of the

Figure 6: Examples of domain pairs used in the experiments. See Section 5.2.4 for details.

Method / Source / Target	MNIST / MNIST-M	Syn Numbers / SVHN	SVHN / MNIST	Syn Signs / GTSRB
Source only	.5225	.8674	.5490	.7900
SA (Fernando et al., 2013)	.5690 (4.1%)	.8644 (−5.5%)	.5932 (9.9%)	.8165 (12.7%)
DANN	**.7666** (52.9%)	**.9109** (79.7%)	**.7385** (42.6%)	**.8865** (46.4%)
Train on target	.9596	.9220	.9942	.9980

Table 2: Classification accuracies for digit image classifications for different source and target domains. MNIST-M corresponds to difference-blended digits over non-uniform background. The first row corresponds to the lower performance bound (*i.e.*, if no adaptation is performed). The last row corresponds to training on the target domain data with known class labels (upper bound on the DA performance). For each of the two DA methods (ours and Fernando et al., 2013) we show how much of the gap between the lower and the upper bounds was covered (in brackets). For all five cases, our approach outperforms Fernando et al. (2013) considerably, and covers a big portion of the gap.

Method / Source / Target	Amazon / Webcam	DSLR / Webcam	Webcam / DSLR
GFK(PLS, PCA) (Gong et al., 2012)	.197	.497	.6631
SA* (Fernando et al., 2013)	.450	.648	.699
DLID (Chopra et al., 2013)	.519	.782	.899
DDC (Tzeng et al., 2014)	.618	.950	.985
DAN (Long and Wang, 2015)	.685	.960	.990
Source only	.642	.961	.978
DANN	**.730**	**.964**	**.992**

Table 3: Accuracy evaluation of different DA approaches on the standard Office (Saenko et al., 2010) data set. All methods (except SA) are evaluated in the "fully-transductive" protocol (some results are reproduced from Long and Wang, 2015). Our method (last row) outperforms competitors setting the new state-of-the-art.

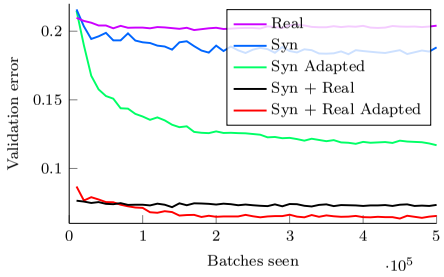


Figure 7: Results for the traffic signs classification in the semi-supervised setting. *Syn* and *Real* denote available labeled data (100,000 synthetic and 430 real images respectively); *Adapted* means that $\approx$ 31,000 unlabeled target domain images were used for adaptation. The best performance is achieved by employing both the labeled samples and the large unlabeled corpus in the target domain.

feature distributions. We observe a quite strong separation between the domains when we feed them into the CNN trained solely on MNIST, whereas for the SVHN-trained network the features are much more intermixed. This difference probably explains why our method succeeded in improving the performance by adaptation in the SVHN $\rightarrow$ MNIST scenario (see Table 2) but not in the opposite direction (SA is not able to perform adaptation in this case either). Unsupervised adaptation from MNIST to SVHN gives a failure example for our approach: it doesn't manage to improve upon the performance of the non-adapted model which achieves $\approx$ 0.25 accuracy (we are unaware of any unsupervised DA methods capable of performing such adaptation).

Synthetic Signs $\rightarrow$ *GTSRB.* Overall, this setting is similar to the Syn Numbers $\rightarrow$ SVHN experiment, except the distribution of the features is more complex due to the significantly larger number of classes (43 instead of 10). For the source domain we obtained 100,000 synthetic images (which we call Syn Signs) simulating various imaging conditions. In the target domain, we use 31,367 random training samples for unsupervised adaptation and the rest for evaluation. Once again, our method achieves a sensible increase in performance proving its suitability for the synthetic-to-real data adaptation.

As an additional experiment, we also evaluate the proposed algorithm for semi-supervised domain adaptation, *i.e.*, when one is additionally provided with a small amount of labeled target data. Here, we reveal 430 labeled examples (10 samples per class) and add them to the training set for the label predictor. Figure 7 shows the change of the validation error throughout the training. While the graph clearly suggests that our method can be beneficial in the semi-supervised setting, thorough verification of semi-supervised setting is left for future work.

Office data set. We finally evaluate our method on Office data set, which is a collection of three distinct domains: Amazon, DSLR, and Webcam. Unlike previously discussed data

sets, Office is rather small-scale with only 2817 labeled images spread across 31 different categories in the largest domain. The amount of available data is crucial for a successful training of a deep model, hence we opted for the fine-tuning of the CNN pre-trained on the ImageNet (`AlexNet` from the `Caffe` package, see Jia et al., 2014) as it is done in some recent DA works (Donahue et al., 2014; Tzeng et al., 2014; Hoffman et al., 2013; Long and Wang, 2015). We make our approach more comparable with Tzeng et al. (2014) by using exactly the same network architecture replacing domain mean-based regularization with the domain classifier.

Following previous works, we assess the performance of our method across three transfer tasks most commonly used for evaluation. Our training protocol is adopted from Gong et al. (2013); Chopra et al. (2013); Long and Wang (2015) as during adaptation we use all available labeled source examples and unlabeled target examples (the premise of our method is the abundance of unlabeled data in the target domain). Also, all source domain data are used for training. Under this "fully-transductive" setting, our method is able to improve previously-reported state-of-the-art accuracy for unsupervised adaptation very considerably (Table 3), especially in the most challenging Amazon $\rightarrow$ Webcam scenario (the two domains with the largest domain shift).

Interestingly, in all three experiments we observe a slight over-fitting (performance on the target domain degrades while accuracy on the source continues to improve) as training progresses, however, it doesn't ruin the validation accuracy. Moreover, switching off the domain classifier branch makes this effect far more apparent, from which we conclude that our technique serves as a regularizer.

5.3 Experiments with Deep Image Descriptors for Re-Identification

In this section we discuss the application of the described adaptation method to person re-identification *(re-id)* problem. The task of person re-identification is to associate people seen from different camera views. More formally, it can be defined as follows: given two sets of images from different cameras (*probe* and *gallery*) such that each person depicted in the probe set has an image in the gallery set, for each image of a person from the probe set find an image of the same person in the gallery set. Disjoint camera views, different illumination conditions, various poses and low quality of data make this problem difficult even for humans (*e.g.*, Liu et al., 2013, reports human performance at Rank1=71.08%).

Unlike classification problems that are discussed above, re-identification problem implies that each image is mapped to a vector descriptor. The distance between descriptors is then used to match images from the probe set and the gallery set. To evaluate results of re-id methods the *Cumulative Match Characteristic* (CMC) curve is commonly used. It is a plot of the identification rate (recall) at rank-k, that is the probability of the matching gallery image to be within the closest k images (in terms of descriptor distance) to the probe image.

Most existing works train descriptor mappings and evaluate them within the same data set containing images from a certain camera network with similar imaging conditions. Several papers, however, observed that the performance of the resulting re-identification systems drops very considerably when descriptors trained on one data set and tested on another. It is therefore natural to handle such cross-domain evaluation as a domain-adaptation problem, where each camera network (data set) constitutes a domain.

Figure 8: Matching and non-matching pairs of probe-gallery images from different person re-identification data sets. The three data sets are treated as different domains in our experiments.

Recently, several papers with significantly improved re-identification performance (Zhang and Saligrama, 2014; Zhao et al., 2014; Paisitkriangkrai et al., 2015) have been presented, with Ma et al. (2015) reporting good results in cross-data-set evaluation scenario. At the moment, deep learning methods (Yi et al., 2014) do not achieve state-of-the-art results probably because of the limited size of the training sets. Domain adaptation thus represents a viable direction for improving deep re-identification descriptors.

5.3.1 Data Sets and Protocols

Following Ma et al. (2015), we use PRID (Hirzer et al., 2011), VIPeR (Gray et al., 2007), CUHK (Li and Wang, 2013) as target data sets for our experiments. The *PRID* data set exists in two versions, and as in Ma et al. (2015) we use a single-shot variant. It contains images of 385 persons viewed from camera A and images of 749 persons viewed from camera B, 200 persons appear in both cameras. The *VIPeR* data set also contains images taken with two cameras, and in total 632 persons are captured, for every person there is one image for each of the two camera views. The *CUHK* data set consists of images from five pairs of cameras, two images for each person from each of the two cameras. We refer to the subset of this data set that includes the first pair of cameras only as *CUHK/p1* (as most papers use this subset). See Figure 8 for samples of these data sets.

We perform extensive experiments for various pairs of data sets, where one data set serves as a source domain, *i.e.*, it is used to train a descriptor mapping in a supervised way with known correspondences between probe and gallery images. The second data set is used as a target domain, so that images from that data set are used without probe-gallery correspondence.

In more detail, CUHK/p1 is used for experiments when CUHK serves as a target domain and two settings ("whole CUHK" and CUHK/p1) are used for experiments when CUHK serves as a source domain. Given PRID as a target data set, we randomly choose 100 persons appearing in both camera views as training set. The images of the other 100 persons from camera A are used as probe, all images from camera B excluding those used in training (649 in total) are used as gallery at test time. For VIPeR, we use random 316 persons for training and all others for testing. For CUHK, 971 persons are split into 485 for training and 486 for testing. Unlike Ma et al. (2015), we use all images in the first pair of cameras of CUHK instead of choosing one image of a person from each camera view. We also performed two

experiments with all images of the whole CUHK data set as source domain and VIPeR and PRID data sets as target domains as in the original paper (Yi et al., 2014).

Following Yi et al. (2014), we augmented our data with mirror images, and during test time we calculate similarity score between two images as the mean of the four scores corresponding to different flips of the two compared images. In case of CUHK, where there are 4 images (including mirror images) for each of the two camera views for each person, all 16 combinations' scores are averaged.

5.3.2 CNN architectures and Training Procedure

In our experiments, we use siamese architecture described in Yi et al. (2014) (*Deep Metric Learning* or *DML*) for learning deep image descriptors on the source data set. This architecture incorporates two convolution layers (with 7×7 and 5×5 filter banks), followed by ReLU and max pooling, and one fully-connected layer, which gives 500-dimensional descriptors as an output. There are three parallel flows within the CNN for processing three part of an image: the upper, the middle, and the lower one. The first convolution layer shares parameters between three parts, and the outputs of the second convolution layers are concatenated. During training, we follow Yi et al. (2014) and calculate pairwise cosine similarities between 500-dimensional features within each batch and backpropagate the loss for all pairs within batch.

To perform domain-adversarial training, we construct a DANN architecture. The feature extractor includes the two convolutional layers (followed by max-pooling and ReLU) discussed above. The label predictor in this case is replaced with *descriptor predictor* that includes one fully-connected layer. The domain classifier includes two fully-connected layers with 500 units in the intermediate representation ($x \to 500 \to 1$).

For the verification loss function in the descriptor predictor we used Binomial Deviance loss, defined in Yi et al. (2014) with similar parameters: $\alpha = 2$, $\beta = 0.5$, $c = 2$ (the asymmetric cost parameter for negative pairs). The domain classifier is trained with logistic loss as in subsection 5.2.2.

We used learning rate fixed to 0.001 and momentum of 0.9. The schedule of adaptation similar to the one described in subsection 5.2.2 was used. We also inserted dropout layer with rate 0.5 after the concatenation of outputs of the second max-pooling layer. 128-sized batches were used for source data and 128-sized batches for target data.

5.3.3 Results on Re-identification data sets

Figure 9 shows results in the form of CMC-curves for eight pairs of data sets. Depending on the hardness of the annotation problem we trained either for 50,000 iterations (CUHK/p1 $\to$ VIPeR, VIPeR $\to$ CUHK/p1, PRID $\to$ VIPeR) or for 20,000 iterations (the other five pairs).

After the sufficient number of iterations, domain-adversarial training consistently improves the performance of re-identification. For the pairs that involve PRID data set, which is more dissimilar to the other two data sets, the improvement is considerable. Overall, this demonstrates the applicability of the domain-adversarial learning beyond classification problems.

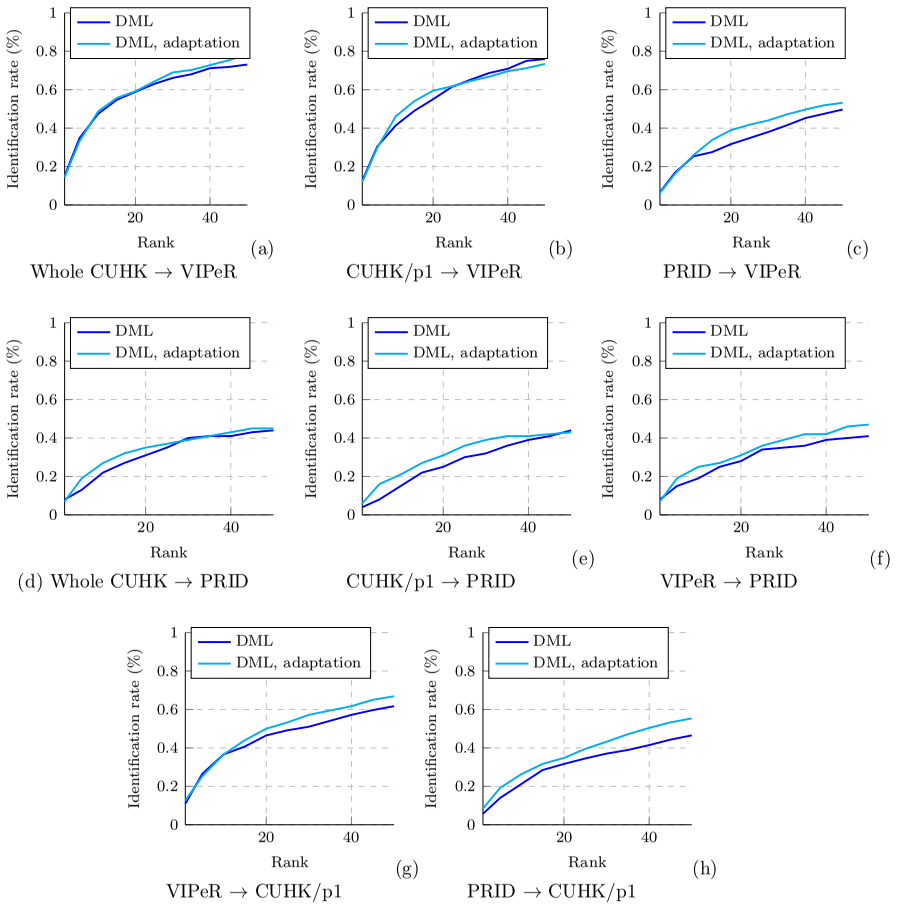
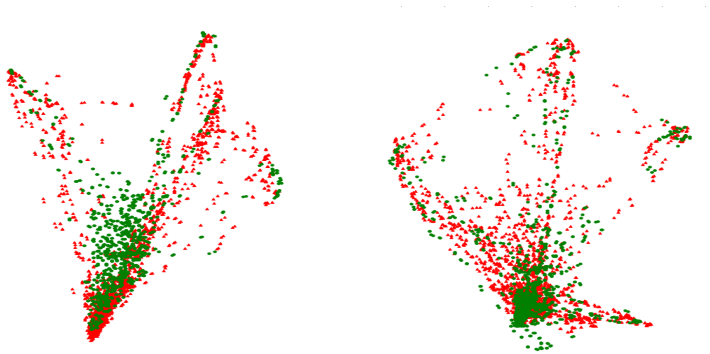


Figure 9: Results on VIPeR, PRID and CUHK/p1 with and without domain-adversarial learning. Across the eight domain pairs domain-adversarial learning improves re-identification accuracy. For some domain pairs the improvement is considerable.



(a) DML (b) DML, adaptation

Figure 10: The effect of adaptation shown by t-SNE visualizations of source and target domains descriptors in a VIPeR → CUHK/p1 experiment pair. VIPeR is depicted with *green* and CUHK/p1 - with *red*. As in the image classification case, domain-adversarial learning ensures a closer match between the source and the target distributions.

Figure 10 further demonstrates the effect of adaptation on the distributions of the learned descriptors in the source and in target sets in VIPeR → CUHK/p1 experiments, where domain adversarial learning once again achieves better intermixing of the two domains.

6. Conclusion

The paper proposes a new approach to domain adaptation of feed-forward neural networks, which allows large-scale training based on large amount of annotated data in the source domain and large amount of unannotated data in the target domain. Similarly to many previous shallow and deep DA techniques, the adaptation is achieved through aligning the distributions of features across the two domains. However, unlike previous approaches, the alignment is accomplished through standard backpropagation training.

The approach is motivated and supported by the domain adaptation theory of Ben-David et al. (2006, 2010). The main idea behind DANN is to enjoin the network hidden layer to learn a representation which is predictive of the source example labels, but uninformative about the domain of the input (source or target). We implement this new approach within both shallow and deep feed-forward architectures. The latter allows simple implementation within virtually any deep learning package through the introduction of a simple gradient reversal layer. We have shown that our approach is flexible and achieves state-of-the-art

results on a variety of benchmark in domain adaptation, namely for sentiment analysis and image classification tasks.

A convenient aspect of our approach is that the domain adaptation component can be added to almost any neural network architecture that is trainable with backpropagation. Towards this end, We have demonstrated experimentally that the approach is not confined to classification tasks but can be used in other feed-forward architectures, *e.g.*, for descriptor learning for person re-identification.

Acknowledgments

This work has been supported by National Science and Engineering Research Council (NSERC) Discovery grants 262067 and 0122405 as well as the Russian Ministry of Science and Education grant RFMEFI57914X0071. Computations were performed on the Colosse supercomputer grid at Université Laval, under the auspices of Calcul Québec and Compute Canada. The operations of Colosse are funded by the NSERC, the Canada Foundation for Innovation (CFI), NanoQuébec, and the Fonds de recherche du Québec – Nature et technologies (FRQNT). We also thank the Graphics & Media Lab, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University for providing the synthetic road signs data set.

References

Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, and Mario Marchand. Domain-adversarial neural networks. *NIPS 2014 Workshop on Transfer and Multi-task learning: Theory Meets Practice*, 2014. URL `http://arxiv.org/abs/1412.4446`.

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transaction Pattern Analysis and Machine Intelligence*, 33, 2011.

Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky. Neural codes for image retrieval. In *ECCV*, pages 584–599, 2014.

Mahsa Baktashmotlagh, Mehrtash Tafazzoli Harandi, Brian C. Lovell, and Mathieu Salzmann. Unsupervised domain adaptation by domain invariant projection. In *ICCV*, pages 769–776, 2013.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *NIPS*, pages 137–144, 2006.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

John Blitzer, Ryan T. McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Conference on Empirical Methods in Natural Language Processing*, pages 120–128, 2006.

Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alexander J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. In *ISMB*, pages 49–57, 2006.

Lorenzo Bruzzone and Mattia Marconcini. Domain adaptation problems: A DASVM classification technique and a circular validation strategy. *IEEE Transaction Pattern Analysis and Machine Intelligence*, 32(5):770–787, 2010.

Minmin Chen, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *ICML*, pages 767–774, 2012.

Qiang Chen, Junshi Huang, Rogerio Feris, Lisa M. Brown, Jian Dong, and Shuicheng Yan. Deep domain adaptation for describing people based on fine-grained clothing attributes. In *CVPR*, June 2015.

S. Chopra, S. Balakrishnan, and R. Gopalan. Dlid: Deep learning for domain adaptation by interpolating between domains. In *ICML Workshop on Challenges in Representation Learning*, 2013.

Dan Cireşan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32:333–338, 2012.

Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theor. Comput. Sci.*, 519:103–126, 2014.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. Technical report, EECS Department, University of California, Berkeley, Mar 2010.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 325–333, 2015. URL `http://jmlr.org/proceedings/papers/v37/ganin15.html`.

Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers. In *ICML*, pages 738–746, 2013.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, pages 513–520, 2011.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, pages 2066–2073, 2012.

Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, pages 222–230, 2013.

Shaogang Gong, Marco Cristani, Shuicheng Yan, and Chen Change Loy. *Person re-identification*. Springer, 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, pages 999–1006, 2011.

Doug Gray, Shane Brennan, and Hai Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *IEEE International Workshop on Performance Evaluation for Tracking and Surveillance, Rio de Janeiro*, 2007.

Martin Hirzer, Csaba Beleznai, Peter M. Roth, and Horst Bischof. Person re-identification by descriptive and discriminative classification. In *SCIA*, 2011.

Judy Hoffman, Eric Tzeng, Jeff Donahue, Yangqing Jia, Kate Saenko, and Trevor Darrell. One-shot adaptation of supervised deep convolutional models. *CoRR*, abs/1312.6204, 2013. URL `http://arxiv.org/abs/1312.6204`.

Fei Huang and Alexander Yates. Biased representation learning for domain adaptation. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1313–1323, 2012.

Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, pages 601–608, 2006.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093, 2014.

Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *Very Large Data Bases*, pages 180–191, 2004.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

Alexandre Lacoste, François Laviolette, and Mario Marchand. Bayesian comparison of machine learning algorithms on single and multiple datasets. In *AISTATS*, pages 665–675, 2012.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

Wei Li and Xiaogang Wang. Locally aligned feature transforms across views. In *CVPR*, pages 3594–3601, 2013.

Yujia Li, Kevin Swersky, and Richard Zemel. Unsupervised domain adaptation by domain invariant projection. In *NIPS 2014 Workshop on Transfer and Multitask Learning*, 2014.

Joerg Liebelt and Cordelia Schmid. Multi-view object class detection with a 3d geometric model. In *CVPR*, 2010.

Chunxiao Liu, Chen Change Loy, Shaogang Gong, and Guijin Wang. POP: person re-identification post-rank optimisation. In *ICCV*, pages 441–448, 2013.

Mingsheng Long and Jianmin Wang. Learning transferable features with deep adaptation networks. *CoRR*, abs/1502.02791, 2015.

Andy Jinhua Ma, Jiawei Li, Pong C. Yuen, and Ping Li. Cross-domain person reidentification using domain adaptation ranking svms. *IEEE Transactions on Image Processing*, 24(5):1599–1613, 2015.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *COLT*, 2009a.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Multiple source adaptation and the rényi divergence. In *UAI*, pages 367–374, 2009b.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

Sakrapee Paisitkriangkrai, Chunhua Shen, and Anton van den Hengel. Learning to rank in person re-identification with metric ensembles. *CoRR*, abs/1503.01543, 2015. URL `http://arxiv.org/abs/1503.01543`.

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226, 2010.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Michael Stark, Michael Goesele, and Bernt Schiele. Back to the future: Learning shape models from 3d CAD data. In *BMVC*, pages 1–11, 2010.

Baochen Sun and Kate Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *BMVC*, 2014.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014. URL `http://arxiv.org/abs/1412.3474`.

Laurens van der Maaten. Barnes-Hut-SNE. *CoRR*, abs/1301.3342, 2013. URL `http://arxiv.org/abs/1301.3342`.

David Vázquez, Antonio Manuel López, Javier Marín, Daniel Ponsa, and David Gerónimo Gomez. Virtual and real world adaptationfor pedestrian detection. *IEEE Transaction Pattern Analysis and Machine Intelligence*, 36(4):797–809, 2014.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103, 2008.

Dong Yi, Zhen Lei, and Stan Z. Li. Deep metric learning for practical person re-identification. *CoRR*, abs/1407.4979, 2014. URL `http://arxiv.org/abs/1407.4979`.

Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. URL `http://arxiv.org/abs/1212.5701`.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. URL `http://arxiv.org/abs/1311.2901`.

Ziming Zhang and Venkatesh Saligrama. Person re-identification via structured prediction. *CoRR*, abs/1406.4444, 2014. URL `http://arxiv.org/abs/1406.4444`.

Rui Zhao, Wanli Ouyang, and Xiaogang Wang. Person re-identification by saliency learning. *CoRR*, abs/1412.1908, 2014. URL `http://arxiv.org/abs/1412.1908`.

Erheng Zhong, Wei Fan, Qiang Yang, Olivier Verscheure, and Jiangtao Ren. Cross validation framework to choose amongst models and datasets for transfer learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 547–562. Springer, 2010.

Journal of Machine Learning Research 17 (2016) 1-34 Submitted 6/15; Revised 12/15; Published 4/16

Probabilistic Low-Rank Matrix Completion from Quantized Measurements

Sonia A. Bhaskar SBHASKAR@STANFORD.EDU
Department of Electrical Engineering
Stanford University
Stanford, CA 94305, USA

Editor: Benjamin Recht

Abstract

We consider the recovery of a low rank real-valued matrix M given a subset of noisy discrete (or quantized) measurements. Such problems arise in several applications such as collaborative filtering, learning and content analytics, and sensor network localization. We consider constrained maximum likelihood estimation of M, under a constraint on the entry-wise infinity-norm of M and an exact rank constraint. We provide upper bounds on the Frobenius norm of matrix estimation error under this model. Previous theoretical investigations have focused on binary (1-bit) quantizers, and been based on convex relaxation of the rank. Compared to the existing binary results, our performance upper bound has faster convergence rate with matrix dimensions when the fraction of revealed observations is fixed. We also propose a globally convergent optimization algorithm based on low rank factorization of M and validate the method on synthetic and real data, with improved performance over previous methods.

Keywords: constrained maximum likelihood, quantization, matrix completion, collaborative filtering, convex optimization

1. Introduction

Recovery of a low-rank matrix from a subset of its entries is known as the matrix completion problem. This problem arises in many applications, including collaborative filtering (Rennie and Srebro, 2005; Koren et al., 2009), sensor network localization (Shang et al., 2004; Karbasi and Oh, 2013), learning and content analytics (Lan et al., 2014c,b), rank aggregation (Gleich and Lim, 2011), and manifold learning (Tenenbaum et al., 2000; Saul and Roweis, 2003). In many of these applications, the entries of the matrix are not real-valued, but discrete or quantized, e.g., binary-valued or multiple-valued. For example, in the Netflix problem where a subset of the users' ratings is observed, the ratings take integer values between 1 and 5. Classical matrix completion has treated these values as real-valued with good results, however, performance improvement can be achieved when the observations are treated as discrete (Davenport et al., 2014; Lan et al., 2014a).

We consider the problem of completing a matrix from a subset of its entries, but instead of assuming the observed entries are real-valued, we observe subset of quantized measurements. These observations are related to the underlying matrix M via a probabilistic model, as follows. Given $M \in \mathbb{R}^{m\times n}$, a subset of indices $\Omega \subseteq [m] \times [n]$, and a twice differentiable function $f_\ell : \mathbb{R} \to [0,1]$, with $\ell \in [K]$, $K \geq 2$, we observe

$$Y_{ij} = \ell \text{ with probability } f_\ell(M_{ij}) \text{ for } (i,j) \in \Omega, \tag{1}$$

where $\sum_{\ell=1}^{K} f_\ell(M_{ij}) = 1$. One important application of this model is the K-level quantization of noisy $M_{ij} + Z_{ij}$, where Y_{ij} is given by (Lan et al., 2014a)

$$Y_{ij} = \mathcal{Q}(M_{ij} + Z_{ij}), \quad (i,j) \in \Omega, \tag{2}$$

where the noise matrix Z has i.i.d. entries with cumulative distribution function (CDF) $\Phi(z)$, and the function $\mathcal{Q} : \mathbb{R} \to [K]$ corresponds to a scalar quantizer that maps a real number to one of the K ordered labels according to

$$\mathcal{Q}(x) = \ell \text{ if } \omega_{\ell-1} < x \leq \omega_\ell, \ \ell \in [K], \tag{3}$$

where $\omega_0 < \omega_1 < \cdots < \omega_K$ are the quantization bin boundaries. We will take $\omega_0 = -\infty$ and $\omega_K = \infty$. This quantization model was first considered in McCullagh (1980) for regression applications.

It then follows that

$$\begin{aligned} f_\ell(M_{ij}) &= P(Y_{ij} = \ell | M_{ij}) \\ &= \Phi(\omega_\ell - M_{ij}) - \Phi(\omega_{\ell-1} - M_{ij}). \end{aligned} \tag{4}$$

This observation model arises in many applications. In connectivity-based sensor network localization (Shang et al., 2004; Karbasi and Oh, 2013), M is a matrix of distances between sensors, and Y_{ij} takes binary values based on whether sensor i and sensor j are within a specified radius of each other. In learning and content analytics (Lan et al., 2014c,b), M governs the learners' responses to questions, and in recommender systems (Rennie and Srebro, 2005; Koren et al., 2009), M can represent the true underlying preferences of users. The matrix Y is then the matrix of ratings $\ell \in [K]$, which may represent quantization of some underlying real-valued user preference. Hence, the model (1)-(3) accounts for finer ordering of users' true preferences which then are quantized to discrete values dictated by the rating system. It is known that Netflix uses such real-valued predictions to order movie recommendations when generating recommendations for a user.

The probabilistic model described in (1)-(3) was first introduced by Davenport et al. (2014) for the case of binary ($K = 2$), or 1-bit, observations and has been studied in depth in the literature. This case corresponds to (2) with $\omega_1 = 0$ when the quantization model is considered. Under the assumption that M is low-rank, Davenport et al. (2014) and Lafond et al. (2014) proposed a convex program using maximum likelihood estimation and a nuclear (or trace) norm to promote a low-rank solution. Both works present theoretical recovery guarantees for the estimate, with the latter improving the convergence rate of the upper bound on the error. In Cai and Zhou (2013), a constrained maximum likelihood estimator was also considered but with the max-norm in place of the nuclear norm. Upper and lower bounds on the error norm of the solution to the resulting convex program were also given of the same order as Davenport et al. (2014). The binary model is also investigated in Soni et al. (2014) for sparse factor models using maximum likelihood estimation with an exact low-rank constraint; their results apply to non-sparse models also.

The theoretical recovery guarantee for the estimate given in Soni et al. (2014) is in the form of an upper bound on the expectation of the error norm, in contrast to Davenport et al. (2014), Lafond et al. (2014) and Cai and Zhou (2013), where the (high probability) upper bounds on the error norm itself are given. The bounds presented in this paper are also on the error norm, not on its expectation.

The extension to multi-level observations ($K \geq 2$) was introduced in Lan et al. (2014a), with a focus on the quantized observation model given in (2). A constrained maximum likelihood estimator, similar to that of Davenport et al. (2014), was proposed and validated through numerical experiments, but no theoretical results were given. An extension to multi-level observations was also proposed in Lafond et al. (2014). In contrast to the quantization observation model of Lan et al. (2014a), which involves just one M, the observation model of Lafond et al. (2014) related the matrix Y of K level observations to a vector of $K-1$ underlying matrices $(M^j)_{j=1}^{K-1}$. An upper bound on the error norm was given for a penalized maximum likelihood estimate of this vector of matrices, of the same order as established for the binary case. Recently Cao and Xie (2015) also investigated matrix completion for categorical data and extended the results of Davenport et al. (2014) to multi-level observations. The error bounds of Cao and Xie (2015) are of the same order as that of Davenport et al. (2014) for the binary case. The multi-level observation model of Cao and Xie (2015) does not include the quantized observation model given in (2).

Generalized performance bounds for a generic regularized convex program with arbitrary regularizer were given in Gunasekar et al. (2014) and Lafond (2015), which can be applied to the observation model (1) for $K \geq 2$ when the the link function f comes from the exponential family. Hence, this theoretical guarantee does not apply when considering a quantization observation model, given in (2) when $K > 2$. In this paper, we will allow for an arbitrary log-concave link function in our observation model and theoretical results, which will allow for applications such as noisy K-level quantization.

Another line of work in the context of collaborative filtering has been concerned with probabilistic matrix factorization (PMF) models (Salakhutdinov and Mnih, 2008; Gopalan et al., 2014), some of which can handle integer-valued observations. In an item ratings context, for an $m \times n$ ratings matrix M, one writes $M = UV^\top$ where the factors $U \in \mathbb{R}^{m\times d}$, $V \in \mathbb{R}^{n\times d}$ represent latent users and item feature matrices. A Gaussian model for the observations, parametrized by these factors, is used in Salakhutdinov and Mnih (2008), and a Poisson model is used in Gopalan et al. (2014) allowing for integer-valued observations. The item and user feature vectors are assigned priors, and hyperparameters and feature vectors are estimated by maximizing the log-posterior in Salakhutdinov and Mnih (2008) and minimizing the Kullback-Leibler divergence in Gopalan et al. (2014). To our knowledge, there are no theoretical recovery guarantees regarding the performance of these PMF models. Also in the context of collaborative filtering, Koren and Sill (2011, 2013) proposed an ordinal model for predicting missing rating distributions from revealed multi-level numerical ratings. The model of Koren and Sill (2011, 2013) is a quantized observation model similar to that in Lan et al. (2014a), and as in Lan et al. (2014a), no theoretical results were given. Modeling of ordinal data with Gaussian restricted Boltzmann machines for both vector-variates and matrix-variates has been investigated in Tran et al. (2012), where a quantized observation model is also considered. No theoretical results were given Tran et al. (2012).

Aside from the PMF models, all prior works have considered convex programs which use a convex relaxation of matrix rank as a surrogate for promoting low rank. While this may be advantageous in cases where the matrix is approximately low rank and also because it results in a convex problem, often in applications the rank is known (as in sensor network localization), or can be reliably estimated. One question is, if we consider an exact rank constraint, can performance guarantees be proved and performance improvement be achieved, and can we find an algorithm to lead us to a global solution?

In this paper, we extend the theory of 1-bit matrix completion to that of multi-level discrete measurements, with an emphasis on the application to quantization. We consider maximum likelihood (ML) estimation of M from multi-level quantized observations, and establish upper bounds on the estimation error norm for this problem, which has a faster rate of convergence than previously established upper bounds for the binary case. In contrast to Gunasekar et al. (2014) and Lafond (2015), we do not restrict the likelihood (i.e., the link function f) to come from an exponential family distribution. We allow the likelihood to come from any strictly log-concave distribution, which includes distributions of bounded discrete random variables from the exponential family. We furthermore focus on the application to a quantization observation model similar to that of Lan et al. (2014a), where the likelihood is not from the exponential family when the number of levels is greater than two (that is, when $K > 2$). Rather than using a convex relaxation to promote a low-rank solution as in previous works, we use an exact rank constraint. We present several algorithms based on matrix factorization for solving our optimization problem, one of which is globally convergent. Our method outperforms some of the existing low-rank matrix completion methods on both synthetic and real world data.

In a preliminary short version of this paper (Bhaskar, 2015), we presented Algorithms 1 and 2 (for known bin boundaries) and stated a preliminary version of our performance upper bound without any proof. The present paper provides a more comprehensive upper bound of the Frobenius norm of the error with the complete proof, an additional algorithm, Algorithm 3, and more extensive numerical experiments which include validation on the MovieLens 1M dataset.

The paper is organized as follows. In Section 2 we discuss the assumptions on the target matrix to make it identifiable and also discuss our sampling model on which we follow Bhojanapalli and Jain (2014). In Section 3, we state our main results regarding theoretical guarantees on matrix recovery. We first describe the proposed constrained ML estimation of the target matrix M from multi-level quantized observations. We then establish upper bounds on the estimation error norm for this problem, which yield a faster rate of convergence than previously established upper bounds for the binary case. In Section 4 we present several algorithms based on matrix factorization for solving our optimization problem, one of which is globally convergent. We corroborate our results with numerical examples in Section 5 where we test our methods on synthetic and real data, and also compare our methods with that of Keshavan et al. (2009, 2010) (OptSpace), Cai et al. (2010) (SVT), Cai and Zhou (2013) and Davenport et al. (2014). Proofs of technical claims are given in the two appendices.

Notation: We use capital letters, such as M, to denote a matrix, and M_{ij} for its (i,j)th entry. We let $\|M\|_2$, $\|M\|_F$, $\|M\|_*$ and $\|M\|_\infty$ denote the operator, Frobenius, nuclear (or trace) and entry-wise infinity norm, respectively, of M. The notation $M^\top$ denotes the

transpose of M, $|\mathcal{S}|$ denotes the cardinality of the set $\mathcal{S}$, $[n]$ denotes the set of integers $\{1, \ldots, n\}$, $\mathbf{1}_n \in \mathbb{R}^n$ is the vector of all ones, $\tilde{\mathbf{1}}_n = \mathbf{1}_n/\sqrt{n}$, and $\mathbf{1}_{[A]}$ denotes the indicator function, i.e. $\mathbf{1}_{[A]} = 1$ when A is true, and $\mathbf{1}_{[A]} = 0$ otherwise. We use $\langle A, B \rangle$ to denote $\mathrm{tr}(A^\top B) = \sum_{ij} A_{ij} B_{ij}$. The abbreviations w.h.p. and w.p.1 stand for with high probability and with probability one, respectively.

2. Preliminaries and Model Assumptions

In this section, we discuss the assumptions on the target matrix M to make it identifiable. We also discuss our sampling model on which we follow Bhojanapalli and Jain (2014).

2.1 Low-rank Matrices

The problem of completing a matrix from a subset of its entries is ill-posed without imposing structural assumptions on the matrix. Hence, some relationship between the entries must be assumed to reconstruct M from a subset of its entries. The majority of the literature on matrix completion assumes that the matrix M to be recovered is low rank, i.e., that it spans a low-dimensional subspace. This assumption is reasonable in many applications.

We assume that M is a low-rank matrix with rank bounded by r. We note that in many applications, such as sensor network localization, where M is known to exist in 2 or 3-dimensional grid, or DNA haplotype assembly, the rank r is known. In examples such as collaborative filtering, where M is a matrix in which rows may represent users and columns may represent their preferences for an item, M is low rank since the users' preferences are believed to be a function of just a few factors. In applications where the rank r is not explicitly known, as in the former example, it can be reliably estimated (Keshavan et al., 2010).

We furthermore assume that the true matrix M satisfies $\|M\|_\infty \leq \alpha$, which helps make the recovery of M well-posed by preventing excessive "spikiness" of the matrix. In classical matrix completion (Cai et al., 2010), the incoherence assumption is used to ensure that the left and right singular vectors are not aligned with the standard basis vectors, and to facilitate establishment of recovery guarantees. This assumption was made less stringent in Negahban and Wainright (2012) by instead restricting the "spikiness" ratio of the matrix. The assumption $\|M\|_\infty \leq \alpha$ follows from this condition (Gunasekar et al., 2014). We refer the reader to Davenport et al. (2014), Cai and Zhou (2013), Klopp (2014) and Negahban and Wainright (2012) for further details.

2.2 Sampling Model

We now discuss our assumptions on the set Ω, on which we follow Bhojanapalli and Jain (2014), where the classical matrix completion problem is considered. Consider a bipartite graph $G = ([m], [n], E)$, where the edge set $E \subseteq [m] \times [n]$ is related to the index set of revealed entries Ω as $(i, j) \in E$ iff $(i, j) \in \Omega$. Abusing the notation, we use G for both the graph and its bi-adjacency matrix where $G_{ij} = 1$ if $(i, j) \in E$, $G_{ij} = 0$ if $(i, j) \notin E$.

We denote the association of G to Ω by $G \backslash \Omega$. Without loss of generality we take $m \geq n$. We assume that each row of G has d nonzero entries (thus $|\Omega| = md$) with the following properties on its singular value decomposition (SVD):

(A1) The left and right top singular vectors of G are $\mathbf{1}_m/\sqrt{m}$ and $\mathbf{1}_n/\sqrt{n}$, respectively.

(A2) We have $\sigma_1(G) \geq d$ and $\sigma_2(G) \leq C\sqrt{d}$, where $C > 0$ is some universal constant. Here $\sigma_1(G)$ and $\sigma_2(G)$ respectively denote the largest and the second largest singular values of G.

Thus we require G to have a large enough spectral gap.

Examples. We now discuss a few examples of graphs families which satisfy assumptions (A1) and (A2).

(1) Ramanujan graphs, a class of regular expander graphs (Hoory et al., 2006).

(2) Erdös-Renyi random graphs with average degree $d \geq c \log(m)$. These graphs satisfy this spectral gap property with high probability (Feige and Ofek, 2005). More explicitly, if G is an Erdös-Renyi bipartite random graph with probability p of an edge being placed, then the ensemble average of the bi-adjacency matrix $\mathbb{E}[G] = p\mathbf{1}_m\mathbf{1}_n^\top = \sqrt{nmp^2}\tilde{\mathbf{1}}_m\tilde{\mathbf{1}}_n^\top$ and $\tilde{G} = G - \mathbb{E}[G]$ is a random matrix with zero-mean i.i.d. entries of variance $p(1-p)$ with the largest singular value having $\mathcal{O}(\sqrt{m+n})$ with high probability (and also with probability 1) (Bolla et al., 2010). Thus, $\sigma_1(G) = \sqrt{nmp^2}$ is the dominant singular value of G, and (A1) and (A2) hold with high probability (and also with probability 1) (Bolla et al., 2010). Note that the uniform sampling assumption used in Davenport et al. (2014), Gunasekar et al. (2014), and Lan et al. (2014c), generates an Erdös-Renyi random graph.

(3) Stochastic block models for certain choices of inter- and intra-cluster edge connection probabilities. Consider the case of two clusters of the left and right vertices, with $m/2$ left vertices and $n/2$ right vertices of graph G belonging to the first cluster and the remaining left and right vertices to second cluster. Suppose that each intra-cluster edge is placed with probability p and an inter-cluster edge is placed with probability q. Then the ensemble average of the bi-adjacency matrix $\mathbb{E}[G]$ consists of elements equal to p for edges with both vertices in the same cluster and q for edges with vertices in different clusters. This can be expressed as (see also Nadakuditi and Newman, 2012)

$$\mathbb{E}[G] = \frac{\sqrt{mn}(p+q)}{2}\tilde{\mathbf{1}}_m\tilde{\mathbf{1}}_n^\top + \frac{\sqrt{mn}|p-q|}{2}\tilde{\mathbf{u}}_m\tilde{\mathbf{u}}_n^\top \tag{5}$$

where $\tilde{\mathbf{u}}_m = \mathbf{u}_m/\sqrt{m}$, the elements of $\mathbf{u}_m \in \mathbb{R}^m$ are ± 1, $+1$ if the left vertex is in the first cluster, -1 otherwise; similarly for $\mathbf{u}_n$ and $\tilde{\mathbf{u}}_n$. For even m and n (clusters of equal size), (5) is an SVD of $\mathbb{E}[G]$ since $\{\tilde{\mathbf{1}}_m, \tilde{\mathbf{u}}_m\}$ and $\{\tilde{\mathbf{1}}_n, \tilde{\mathbf{u}}_n\}$ are sets of orthonormal vectors representing the left and right singular vectors of non-zero singular values of $\mathbb{E}[G]$, which is of rank 2. The matrix $\tilde{G} = G - \mathbb{E}[G]$ is a random matrix with bounded and independent entries, with the largest singular value having $\mathcal{O}(\sqrt{m+n})$ (Bolla et al., 2010). Thus, the two largest singular values of G are $\frac{\sqrt{mn}(p+q)}{2}$ and $\frac{\sqrt{mn}|p-q|}{2}$ (perturbed by random $\tilde{G}$). When $p = q$, we have an Erdös-Renyi random graph with the largest spectral gap. As q becomes smaller, the spectral gap decreases. By (5), (A1) is true, but for (A2) to be true, one should have $|p - q| = \mathcal{O}(1/\sqrt{m})$. For fixed p and q,

$\sigma_2(G) = \frac{\sqrt{mn}|p-q|}{2}$ does not satisfy (A2) although the spectral gap can be made smaller by making $|p-q|$ smaller. As shown in Bhojanapalli and Jain (2014), the stochastic block model is a useful device to study the effect of the spectral gap on the performance of matrix recovery approaches. Such stochastic block models also apply to practical settings such as modeling connectivity subnetworks in the brain (Ghanbari et al., 2013).

3. Main Results

In this section, we describe the rank-constrained ML estimation of the target matrix M from multi-level quantized observations. We then establish upper bounds on the estimation error norm for this problem, which yield a faster rate of convergence than previously established upper bounds for the binary case.

3.1 Rank-Constrained Maximum Likelihood Estimation

We wish to estimate unknown M from the observed entries of Y using a constrained ML approach. We assume Y is related to M via the probabilistic model given in (1)-(4). We use $X \in \mathbb{R}^{m\times n}$ to denote the optimization variable. The negative log-likelihood function is given by

$$F_{\Omega,Y}(X) = -\sum_{(i,j)\in\Omega}\left[\sum_{\ell=1}^{K}\log(f_\ell(X_{ij}))\mathbf{1}_{[Y_{ij}=\ell]}\right] \tag{6}$$

which is a convex function in X when the function f_ℓ is log-concave in X_{ij}, and can be an implicit function of $\boldsymbol{\omega}$, where

$$\boldsymbol{\omega} = [\omega_1\ \omega_2\ \cdots\ \omega_{K-1}]^\top \in \mathbb{R}^{K-1} \tag{7}$$

is the vector of bin boundaries.

Two common choices for which the function f_ℓ, and its associated CDFs and pdfs, are log-concave, are:

(i) Logistic model with logistic CDF $\Phi(x) = \Phi_{log}(x/\sigma) = \frac{1}{1+e^{-x/\sigma}}$, $\sigma > 0$.

(ii) Probit model with $\Phi(x) = \Phi_{norm}(x/\sigma)$ where $\sigma > 0$ and $\Phi_{norm}(x)$ is the CDF of the standard normal distribution $\mathcal{N}(0,1)$.

We consider two classes of problems. In the first, $\boldsymbol{\omega}$ is known, and thus the distribution in (4) is completely specified. We will assume that the bin boundaries ω_ℓ, $\ell \in [K]$, are known for our theoretical results. In the other, $\boldsymbol{\omega}$ is unknown and will be determined as part of the optimization problem. By allowing the bin boundaries to be determined by the optimization, we allow the distribution in (4) to be tuned to real data. For our numerical results, we allow the ω_ℓs to be unknown and estimate them (along with M), as in Lan et al. (2014a).

When $\boldsymbol{\omega}$ is known, we seek the solution to the optimization problem (P1): (s.t. stands for subject to)

$$\begin{aligned}(\text{P1}):\quad &\widehat{M} = \arg\min_{X} F_{\Omega,Y}(X)\\ &\text{s.t.}\quad \|X\|_\infty \le \alpha,\ \text{rank}(X) \le r.\end{aligned} \tag{8}$$

As a result of the rank constraint, (P1) is a nonconvex optimization problem. In Section 4, we will discuss factorization methods for solving this problem which come with guarantees of global convergence. In Section 3.2, we present performance upper bounds for problem (P1).

When $\boldsymbol{\omega}$ is unknown, the constrained ML estimate we wish to obtain is given by the solution to the optimization problem (P2):

$$\begin{aligned}(\text{P2}):\quad &\left(\widehat{M},\widehat{\boldsymbol{\omega}}\right) = \arg\min_{X,\boldsymbol{\omega}} F_{\Omega,Y}(X)\ \text{s.t.}\ \ \|X\|_\infty \le \alpha,\\ &\text{rank}(X) \le r \text{ and } \omega_1 < \omega_2 < \cdots < \omega_{K-1}.\end{aligned} \tag{9}$$

The negative log-likelihood $F_{\Omega,Y}(X)$ is not jointly convex in X and $\boldsymbol{\omega}$. However, we show in Lemma 3 in Appendix A that f_ℓ is log-concave in ω_k for fixed X and ω_is ($i \ne k$), and in Section 3.3, that it is strictly log-concave in X for fixed $\boldsymbol{\omega}$ whenever $\Phi(x)$ is log-concave. Thus, $F_{\Omega,Y}(X)$ is convex in ω_k for fixed X and ω_is ($i \ne k$), and convex in X for fixed $\boldsymbol{\omega}$. Consequently, as seen later in Section 4, it will require an alternating minimization procedure (block-coordinate descent).

3.2 Performance Upper Bounds

We now present performance upper bounds for $\widehat{M}$ in (8), i.e., problem (P1) where $\boldsymbol{\omega}$, the vector of true bin boundaries, is assumed to be known. We first define some constants which appear in the bound, involving functions of $f_\ell(x)$ and its first two derivatives. With $\dot{f}(x) := (\mathrm{d}f(x)/\mathrm{d}x)$, define

$$\gamma_\alpha \le \min_{\ell\in[K]}\inf_{|x|\le\alpha}\left\{\frac{\dot{f}_\ell^2(x)}{f_\ell^2(x)} - \frac{\ddot{f}_\ell(x)}{f_\ell(x)}\right\}, \tag{10}$$

$$L_\alpha \ge \max_{\ell\in[K]}\sup_{|x|\le\alpha}\left\{\left|\dot{f}_\ell(x)\right|/f_\ell(x)\right\}, \tag{11}$$

where α is the bound on the entry-wise infinity-norm of $\widehat{M}$ (see Equation 8). For further reference, define the constraint set

$$\mathcal{C} := \left\{X \in \mathbb{R}^{m\times n}\ :\ \|X\|_\infty \le \alpha,\ \text{rank}(X) \le r\right\}. \tag{12}$$

Theorem 1 *Suppose that $M \in \mathcal{C}$, and $G\backslash\Omega$ satisfies assumptions (A1) and (A2). Without loss of generality, assume $m \ge n$. Further, suppose Y is generated according to (1) where $f_\ell(x)$ is log-concave in x $\forall \ell \in [K]$. Then, provided $\gamma_\alpha > 0$, with probability at least $1 - 2(9\alpha\sqrt{mn})^{-r(m+n+1)} - C_1\exp(-C_2 m)$, any global minimizer $\widehat{M}$ of (8) satisfies*

$$\frac{1}{\sqrt{mn}}\|\widehat{M} - M\|_F \le \min\left(2\alpha,\ U_1,\ U_2\right) \tag{13}$$

where

$$U_1 = \max\left(\frac{C_{1\alpha} r\sigma_2(G)}{\sigma_1(G)}, \frac{C_{2\alpha} m\sqrt{r^3 n}}{\sigma_1^2(G)}\right) \le \max\left(\frac{C_{1\alpha} Cr\sqrt{m}}{\sqrt{|\Omega|}}, \frac{C_{2\alpha} m^3\sqrt{r^3 n}}{|\Omega|^2}\right), \tag{14}$$

$$U_2 = \max\left(\frac{C_{1\alpha} r\sigma_2(G)}{\sigma_1(G)},\ \frac{C_{3\alpha} r^{0.75}\left(|\Omega|(m+n+1)\log(9\alpha\sqrt{mn})\right)^{0.25}}{\sigma_1(G)}\right) \tag{15}$$

$$\leq \max\left(\frac{C_{1\alpha} C r\sqrt{m}}{\sqrt{|\Omega|}},\ C_{3\alpha}\left(\frac{r}{|\Omega|}\right)^{0.75} m\left((m+n+1)\log(9\alpha\sqrt{mn})\right)^{0.25}\right). \tag{16}$$

Here, $C_{1\alpha} = 4\sqrt{2}\alpha$, $C_{2\alpha} = 32.16\sqrt{2}L_\alpha/\gamma_\alpha$, $C_{3\alpha} = 8\sqrt{\frac{(1+\alpha)L_\alpha}{\gamma_\alpha}}$, $C_1, C_2, C > 0$ are universal constants, and γ_α and L_α are given by (10), (11).

A proof of Theorem 1 is given in Appendix B. In the binary case ($K = 2$) the link function f in (4) belongs to the exponential family for a large class of CDFs $\Phi(x)$ (e.g., logistic or Gaussian), but not for $K > 3$. The bounding approaches in Gunasekar et al. (2014), Lafond (2015) and Cao and Xie (2015) for $K > 3$ require f to come from the exponential family whereas our approach based on a Taylor series approximation and some concentration inequalities applies to the quantization model (4). One of the novelties in our proof compared to existing works is how we bound the gradient of the cost function in two different ways (see Lemmas 5 and 6 in Appendix B).

Of some interest is the case where the fraction of revealed entries $p = \frac{|\Omega|}{mn}$ is fixed and we let m and n become large, with $m/n \equiv \delta \geq 1$ fixed. In this case we have the following Corollary.

Corollary 2 *Assume the conditions of Theorem 1. Let $p = \frac{|\Omega|}{mn}$ be fixed independent of m and n. Then*

$$U_1 \leq \mathcal{O}\left(\frac{\delta}{p^2}\sqrt{\frac{r^3}{n}}\right), \quad U_2 \leq \mathcal{O}\left(\left(\frac{r^3\delta^2\log(n)}{p^3 n}\right)^{1/4}\right).$$

Then with probability at least $1 - C_1\exp(-C_2 m) - 2/(9\alpha n\sqrt{\delta})^{2rn}$, any global minimum $\widehat{M}$ to (8) satisfies

$$\frac{1}{\sqrt{mn}}\|\widehat{M} - M\|_F \leq \min\left(\mathcal{O}\left(\frac{\delta}{p^2}\sqrt{\frac{r^3}{n}}\right), \mathcal{O}\left(\left(\frac{r^3\delta^2\log(n)}{p^3 n}\right)^{1/4}\right)\right). \tag{17}$$

Corollary 2 suggests that for "larger" fixed values of p, U_1 dominates, signifying a convergence rate of at least $1/\sqrt{n}$ for $\frac{1}{\sqrt{mn}}\|\widehat{M} - M\|_F$ whereas for "small" values of p, U_2 is likely to dominate signifying convergence rate of at least $(\log(n)/n)^{1/4}$. In our simulation results shown later in Figure 5 for $m = n = 100, 200$, or 400, and $p = 0.2, 0.4$, or 0.6, we find that $\frac{1}{mn}\|\widehat{M} - M\|_F^2$ decays approximately as $\mathcal{O}(1/n)$.

3.3 Constants γ_α and L_α for the logistic and probit models

It is known that $f_\ell(x)$ is log-concave iff $\ddot{f}_\ell(x)f_\ell(x) \leq (\dot{f}_\ell(x))^2$ (Boyd and Vandenberghe, 2004). Thus $\gamma_\alpha \geq 0$ for log-concave $f_\ell(x)$ and $\gamma_\alpha > 0$ for strictly log-concave $f_\ell(x)$. For the

logistic model, i.e., when $\Phi(x) = \Phi_{log}(x/\sigma)$, some tedious calculations show that

$$\frac{\dot{f}_\ell^2(x)}{f_\ell^2(x)} - \frac{\ddot{f}_\ell(x)}{f_\ell(x)} = \frac{1}{\sigma^2}\Big[\Phi_{log}\left(\frac{\omega_\ell - x}{\sigma}\right)\left(1 - \Phi_{log}\left(\frac{\omega_\ell - x}{\sigma}\right)\right) + \Phi_{log}\left(\frac{\omega_{\ell-1} - x}{\sigma}\right)\left(1 - \Phi_{log}\left(\frac{\omega_{\ell-1} - x}{\sigma}\right)\right)\Big]$$
$$> 0 \ \ \forall x \in \mathbb{R},\ \forall \ell \in [K]. \tag{18}$$

Therefore, by (10), $\gamma_\alpha > 0$ for the logistic model. Similarly one can verify numerically that $\gamma_\alpha > 0$ for the probit model when $\Phi(x) = \Phi_{norm}(x/\sigma)$. For the logistic model, it turns out that $L_\alpha = 1/(2\sigma\beta_{\alpha\sigma})$ where

$$\beta_{\alpha\sigma} := \min_{\ell\in[K]} \inf_{|x|\leq\alpha} \left\{\Phi_{log}(\frac{\omega_\ell - x}{\sigma}) - \Phi_{log}(\frac{\omega_{\ell-1} - x}{\sigma})\right\} > 0. \tag{19}$$

For the probit model, we have $L_\alpha = \sqrt{2}/(\sqrt{\pi}\sigma\beta_{n\alpha\sigma})$ where

$$\beta_{n\alpha\sigma} := \min_{\ell\in[K]} \inf_{|x|\leq\alpha} \left\{\Phi_{norm}(\frac{\omega_\ell - x}{\sigma}) - \Phi_{norm}(\frac{\omega_{\ell-1} - x}{\sigma})\right\} > 0. \tag{20}$$

3.4 Comparison of Convergence Rates

We first provide a comparison of our bounds with those of Davenport et al. (2014) and Cai and Zhou (2013), who have established bounds for only the binary ($K = 2$) level case. Consider $M \in \mathbb{R}^{n\times n}$, with p fraction of its entries sampled. Then $m = n$, and $|\Omega| = pn^2$. The bounds proposed in Davenport et al. (2014) and Cai and Zhou (2013) yield (for $|\Omega| \geq 4n\log(n)$)

$$\frac{1}{n^2}\|\widehat{M} - M\|_F^2 \leq \mathcal{O}\left(\sqrt{\frac{r}{pn}}\right). \tag{21}$$

Our bound (Corollary 2) yields

$$\frac{1}{n^2}\|\widehat{M} - M\|_F^2 \leq \min\left(\mathcal{O}\left(\frac{r^3}{p^4 n}\right), \mathcal{O}\left(\sqrt{\frac{r^3\log(n)}{p^3 n}}\right)\right). \tag{22}$$

For $K = 2$, the results of Lafond et al. (2014), Gunasekar et al. (2014) and Lafond (2015) apply to our model but not for $K > 2$. The bound of Lafond et al. (2014) and Lafond (2015) yields

$$\frac{1}{n^2}\|\widehat{M} - M\|_F^2 \leq \mathcal{O}\left(\frac{r\log(n)}{pn}\right) \tag{23}$$

and the bound of (Gunasekar et al., 2014, Corollary 1) yields

$$\frac{1}{n^2}\|\widehat{M} - M\|_F^2 \leq \mathcal{O}\left(\alpha^{*2}\frac{r\log(n)}{pn}\right) = \mathcal{O}\left(\frac{rm\log(n)}{p}\right) \tag{24}$$

since in Gunasekar et al. (2014), $\alpha^* \geq \sqrt{mn}\|M\|_\infty$. The bound in Soni et al. (2014) as applied to non-sparse models and $K = 2$, yields

$$\frac{1}{n^2}\mathbb{E}\left[\|\widehat{M} - M\|_F^2\right] \leq \mathcal{O}\left(\frac{r\log(n)}{pn}\right) \tag{25}$$

where the expectation is over the noise (Z_{ij} in Equation 2) and sampling distributions of the revealed matrix entries. Thus, (25) is similar to (23) but is averaged over the noise and sampling distributions. The multi-level observation model of Cao and Xie (2015) does not include the quantized observation model given in (2) but applies to a multinomial logistic model. The bound in Cao and Xie (2015) yields

$$\frac{1}{n^2}\|\widehat{M} - M\|_F^2 \leq \mathcal{O}\left(\sqrt{\frac{r}{pn}}\right). \tag{26}$$

In (21)-(26) the omitted constants do not depend on r, p or n.

Comparing (21)-(26) for the case $K = 2$, we see our method has faster convergence rate in n for fixed rank r and fraction of revealed entries p, compared to Davenport et al. (2014), Cai and Zhou (2013), Lafond et al. (2014), Gunasekar et al. (2014), Lafond (2015) and Soni et al. (2014); the same comment applies for $K > 2$ when comparing with Cao and Xie (2015). On the other hand, for fixed n, our bound is inferior to these other bounds in p or r. One may notice if the revealed entries scale with n according to $p \sim \mathcal{O}(\log(n)/n)$ then Davenport et al. (2014) and Cao and Xie (2015) yield bounded error while our bound grows with n; in our case we need $p \geq \mathcal{O}(1/n^{1/3})$. We believe this to be an artifact of our proof, as our numerical results in Figure 1 show our method outperforms Cai and Zhou (2013) and Davenport et al. (2014), especially for low values of p and higher values of rank r.

4. Optimization

In this section, we describe the algorithms used to solve problems (P1) and (P2). We use the matrix factorization technique (Bach et al., 2008; Burer and Monteiro, 2003; Lee et al., 2010) where instead of optimizing with respect to X, it is factorized into two matrices $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{n\times k}$ such that $X = UV^\top$. One then optimizes with respect to the factors U, V. This method is non-convex, however, it is known (Bach et al., 2008; Burer and Monteiro, 2003; Lee et al., 2010) that if k is chosen to be large enough, it is guaranteed that the local minimum of the problem is also the global minimum of the non-factorized problem.

4.1 Known Bin Boundaries

We have the following approximate projected gradient method for solving problem (P1) following the algorithm of Cai and Zhou (2013), where the case of $K = 2$ was considered.

4.1.1 Algorithm 1: Approximate Projected Gradient Method

Given initial estimates U^0, V^0, one updates

$$\begin{bmatrix} U^{t+1} \\ V^{t+1} \end{bmatrix} = \mathcal{P}_\alpha\left(\begin{bmatrix} U^t - \frac{\tau}{\sqrt{t}}\nabla_X F_{\Omega,Y}(U^tV^{t\top})V^t \\ V^t - \frac{\tau}{\sqrt{t}}\nabla_X F_{\Omega,Y}(U^tV^{t\top})^\top U^t \end{bmatrix}\right) \tag{27}$$

where U^t, V^t are the estimates at iteration t, and

$$\mathcal{P}_\alpha\left([U^\top\ V^\top]^\top\right) = \begin{cases} \sqrt{\alpha/\|UV^\top\|_\infty}[U^\top\ V^\top]^\top & \text{if} \quad \|UV^\top\|_\infty > \alpha \\ [U^\top\ V^\top]^\top & \text{if} \quad \|UV^\top\|_\infty \leq \alpha. \end{cases} \tag{28}$$

In (27) the stepsize τ is selected via a backtracking line search using Armijo's rule, to minimize the cost $F_{\Omega,Y}(U^{t+1}V^{t+1\top})$.

In addition to the approximate projection $\mathcal{P}_\alpha$ in (28), Cai and Zhou (2013) (for $K = 2$) also uses another projection to enforce a max-norm constraint. In Cai and Zhou (2013), for $K = 2$, the negative log-likelihood cost is minimized w.r.t. X subject to the constraints $\|X\|_\infty \leq \alpha$ and $\|X\|_{\max} \leq R$ for some $\alpha > 0$ and $R > 0$. The operator $\mathcal{P}_\alpha$ enforces the constraint $\|X\|_\infty \leq \alpha$. The factored form definition of the max norm (Lee et al., 2010) is given by $\|X\|_{\max} = \inf\left\{\max(\|\bar{U}\|_{2,\infty}^2, \|\bar{V}\|_{2,\infty}^2) : X = \bar{U}\bar{V}^\top\right\}$ where $\|\bar{U}\|_{2,\infty} = \max_i \sqrt{\sum_j \bar{U}_{ij}^2}$, $\bar{U} \in \mathbb{R}^{m\times k}$, $\bar{V} \in \mathbb{R}^{n\times k}$, $k = 1, 2, \ldots, \min(m, n) = n$. For fixed k and $X = UV^\top$, Cai and Zhou (2013) enforce the constraint set $\|X\|_{\max} \leq R$ by requiring $\max(\|U\|_{2,\infty}^2, \|V\|_{2,\infty}^2) \leq R$. As stated in Cai and Zhou (2013), the global minimum of the cost over the constraints $\|X\|_\infty \leq \alpha$ and $\|X\|_{\max} \leq R$ is the same as that over the constraints $\|X\|_\infty \leq \alpha$, $\|U\|_{2,\infty}^2 \leq R$ and $\|V\|_{2,\infty}^2 \leq R$ if $\text{rank}(X) \leq k$. If a matrix A has rank r and $\|A\|_\infty \leq \alpha$, then $\|A\|_{\max} \leq \sqrt{r}\alpha$ (Cai and Zhou, 2013). Therefore, in our case the max-norm constraint is unnecessary as it is automatically satisfied for any $R \geq \sqrt{r}\alpha$. In this sense, our Algorithm 1 is the same as the approach of Cai and Zhou (2013) when $K = 2$ and one picks $R \geq \sqrt{r}\alpha$.

Remark 1 *The hard rank constraint results in a nonconvex constraint set. Thus,* (8) *is a nonconvex optimization problem; similarly for Algorithm 1 for which the rank constraint is implicit in the factorization of X. However, the following result is shown in (Bach et al., 2008, Proposition 4), based on Burer and Monteiro (2003), for nonconvex problems of this form. If (U^*, V^*) is a local minimum of the reformulated (i.e., factored) problem, then $X^* = U^*V^{*\top}$ is the global minimum of problem* (8)*, so long as U^* and V^* are rank-deficient. (Rank deficiency of (U^*, V^*) is a sufficient condition, not necessary.) This result is invoked in Recht and Re (2013), Lee et al. (2010) and Cai and Zhou (2013) for problems of this form. Thus one would expect to achieve global convergence for the problem of* (8) *provided iterations (27)-(28) converge to a local minimum. These iterations represent a projected gradient method which converges to a stationary point if one has orthogonal projection onto a convex constraint set (Bertsekas, 1999, Prop. 2.3.1). However, the "projection" $\mathcal{P}_\alpha$ in (28) is not an orthogonal projection and the set $\{\|UV^\top\|_\infty \leq \alpha\}$ is not convex in U, V (although $\{\|X\|_\infty \leq \alpha\}$ is convex in X), therefore, convergence to even a local minimum is not ensured. However, numerically, this method has still provided good results (similarly reported in Cai and Zhou (2013)). In Cai and Zhou (2013), for the $K = 2$ case, there are two additional constraints $\|U\|_{2,\infty}^2 \leq R$ and $\|V\|_{2,\infty}^2 \leq R$ which are convex sets in U and V, and the corresponding projections are orthogonal projections. However, the projection corresponding to our $\mathcal{P}_\alpha$ in Cai and Zhou (2013) is not orthogonal.*

Thus, for problems of this form, one can choose $k = r + 1$ to achieve global convergence if an upper bound r on the rank of M is known.

The convergence deficiency discussed in Remark 1 motivates the following log-barrier penalty function approach.

4.1.2 Algorithm 2: Logarithmic Barrier Gradient Method

The constraint $\max_{i,j}|X_{ij}| \leq \alpha$ translates to $X_{ij} - \alpha \leq 0$ and $-X_{ij} - \alpha \leq 0\ \forall (i,j)$, which motivates the log-barrier penalty function $-\log\left(1-(X_{ij}/\alpha)^2\right)$, which is finite for $|X_{ij}| < \alpha$, and is infinite otherwise. This leads to the objective function

$$\tilde{F}_{\Omega,Y}(X) = F_{\Omega,Y}(X) - \lambda \sum_{(i,j)} \log\left(1-(X_{ij}/\alpha)^2\right) \tag{29}$$

and the optimization problem

$$\widehat{M} = \arg\min_{X} \tilde{F}_{\Omega,Y}(X) \text{ s.t. } \text{rank}(X) \leq r. \tag{30}$$

The parameter $\lambda > 0$ in (29) sets the accuracy of approximation of $\max_{i,j}|X_{ij}| \leq \alpha$ via the log-barrier function (which is twice-differentiable and convex in X, hence so is $\tilde{F}_{\Omega,Y}(X)$). Now, however, the factorization approach $X = UV^\top$ is well-justified, per Remark 1, and convergence is guaranteed.

The log-barrier method is ill-conditioned and solving the problem for a fixed value of λ generally only works well for small problems or good choices of initialization (Boyd and Vandenberghe, 2004). The above problem is typically solved via a sequence of central path following solutions (Boyd and Vandenberghe, 2004) where one gradually reduces λ toward 0. In our approach we initialize it with the solution to Algorithm 1, providing a good initialization, and then either use a single "small" value of λ, or select λ via 5-fold cross-validation. One may therefore view augmentation with the log-barrier cost as regularization of $F_{\Omega,Y}(X)$.

We solve the factored version $X = UV^\top$ of problem (30) for a fixed λ using a gradient method as follows. Given initial estimates U^0, V^0, one updates

$$\begin{bmatrix} U^{t+1} \\ V^{t+1} \end{bmatrix} = \begin{bmatrix} U^t - \frac{\tau}{\sqrt{t}} \nabla_X \tilde{F}_{\Omega,Y}(U^t V^{t\top}) V^t \\ V^t - \frac{\tau}{\sqrt{t}} \nabla_X \tilde{F}_{\Omega,Y}(U^t V^{t\top})^\top U^t \end{bmatrix} \tag{31}$$

where U^t, V^t are the estimates at iteration t and $\nabla_X \tilde{F}_{\Omega,Y}(U^t V^{t\top}) = \nabla_X \tilde{F}_{\Omega,Y}(X)\big|_{X=U^tV^{t\top}}$. In (31) the stepsize τ is selected via a backtracking line search using Armijo's rule, to minimize the cost $\tilde{F}_{\Omega,Y}(U^{t+1}V^{t+1\top})$.

Define $Z = [U^\top\ V^\top]^\top \in \mathbb{R}^{(m+n)\times k}$ and let $\bar{F}_{\Omega,Y}(Z^t) = \tilde{F}_{\Omega,Y}(U^t V^{t\top})$. Then (31) can be rewritten as

$$Z^{t+1} = Z^t - \frac{\tau}{\sqrt{t}} \nabla_Z \bar{F}_{\Omega,Y}(Z^t), \quad \nabla_Z \bar{F}_{\Omega,Y}(Z^t) = \begin{bmatrix} \nabla_X \tilde{F}_{\Omega,Y}(U^t V^{t\top}) V^t \\ \nabla_X \tilde{F}_{\Omega,Y}(U^t V^{t\top})^\top U^t \end{bmatrix}. \tag{32}$$

Thus, (31) is a gradient descent method using Armijo's rule for stepsize selection, for unconstrained minimization of the continuously differentiable cost $\tilde{F}_{\Omega,Y}(UV^\top) = \bar{F}_{\Omega,Y}(Z)$ w.r.t. Z. By (Bertsekas, 1999, Prop. 1.2.1), the iterations given in (31) converge to a stationary point of $\tilde{F}_{\Omega,Y}(UV^\top)$. Since the cost is non-increasing at every iteration, the stationary point is either a local minimum or a saddle point. In theory, convergence to saddle points can not be excluded for gradient descent algorithms for continuously differentiable functions. However, we assume that we escape saddle points in practice as saddle points are generally unstable from a numerical point of view, i.e., a perturbation always exists at the saddle point which will decrease the cost function.

4.2 Unknown Bin Boundaries

As noted earlier $F_{\Omega,Y}(X)$ is convex in ω_k for fixed X and ω_is ($i \neq k$), and convex in X for fixed $\boldsymbol{\omega}$, however, $F_{\Omega,Y}(X)$ is not jointly convex in X and $\boldsymbol{\omega}$. Thus the problem (P2) specified in (9) is multi-convex in X and $\omega_1, \omega_2, \cdots, \omega_{K-1}$, and one approach to solve it is via block-coordinate descent (Xu and Yin, 2013), for which there are no convergence guarantees, in general. In order to detail this approach, with an abuse of notation, we now explicitly denote the dependence of $F_{\Omega,Y}(X)$ on the ω_is as $F_{\Omega,Y}(X, \omega_1, \cdots, \omega_{K-1})$, and that of $\tilde{F}_{\Omega,Y}(X)$ as $\tilde{F}_{\Omega,Y}(X, \omega_1, \cdots, \omega_{K-1})$. Following Xu and Yin (2013) and our Algorithms 1 and 2, our optimization algorithm for the case of unknown bin boundaries is given in Algorithm 3 where, in Step 5, δ_0 makes the constraint set for ω_i convex.

Algorithm 3 Block-Coordinate Descent Method for Solving (9)

Input: Set of observed entries Y_{ij} for $(i,j) \in \Omega$, initialization $U^0 \in \mathbb{R}^{m\times k}$, $V^0 \in \mathbb{R}^{m\times k}$, $\omega_1^0, \omega_2^0, \cdots, \omega_{K-1}^0 \in \mathbb{R}$, $\omega_1^0 < \omega_2^0 < \cdots < \omega_{K-1}^0$, parameters α, λ

Output: Solution $X^* = U^* V^{*\top}$, $\boldsymbol{\omega}^*$

1: **for** $\ell = 1, 2, \cdots$, until convergence, **do**
2: $\left(L^\ell, R^\ell\right) \leftarrow \arg\min_{U,V} F_{\Omega,Y}(UV^\top, \omega_1^{\ell-1}, \cdots, \omega_{K-1}^{\ell-1})$ subject to $\|UV^\top\|_\infty \leq \alpha$. Solve using approximate projected gradient method (27) initialized with $U^{\ell-1}, V^{\ell-1}$.
3: $\left(U^\ell, V^\ell\right) \leftarrow \arg\min_{U,V} \tilde{F}_{\Omega,Y}(UV^\top, \omega_1^{\ell-1}, \cdots, \omega_{K-1}^{\ell-1})$. Solve using log-barrier gradient method (31) initialized with L^ℓ, R^ℓ.
4: **for** $i = 1, 2, \cdots, K-1$, **do**
5: $\omega_i^\ell \leftarrow \arg\min_{\omega_i} F_{\Omega,Y}(U^\ell V^{\ell\top}, \omega_1^\ell, \cdots, \omega_{i-1}^\ell, \omega_i, \omega_{i+1}^{\ell-1}, \cdots, \omega_{K-1}^{\ell-1})$ subject to $\omega_{i-1}^\ell + \delta_0 \leq \omega_i \leq \omega_{i+1}^{\ell-1} - \delta_0$ for some "small" $\delta_0 > 0$. Solve using a gradient descent method initialized with $\omega_i = \omega_i^{\ell-1}$.
6: **end for**
7: **end for**
8: **return** $X^* = U^* V^{*\top}$, $\boldsymbol{\omega}^*$

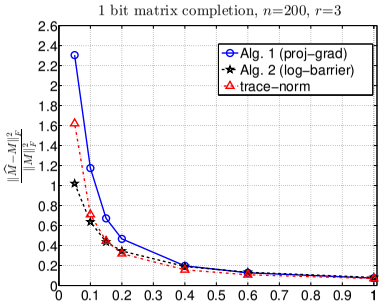
In Step 2 of Algorithm 3, we have used the solution of the approximate projected gradient method (27), to provide a good initialization to the log-barrier algorithm in Step 3, since we are not using central path following, for the reasons aforementioned in Section 4.1.2.

5. Numerical Experiments

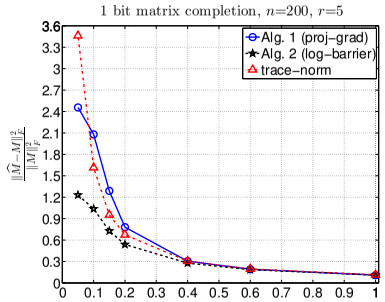
In this section, we test our methods on synthetic and real data, and also compare our methods with that of Keshavan et al. (2009, 2010) (OptSpace), Cai et al. (2010) (SVT), Cai and Zhou (2013) and Davenport et al. (2014).

5.1 Synthetic Data

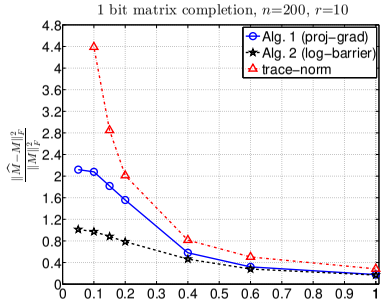
In this section, we report the results of evaluating our method on synthetic data. We set $m = n$ and construct $M \in \mathbb{R}^{n\times n}$ as $M = M_1 M_2^\top$ where M_1 and M_2 are $n \times r$ matrices with



(a) $r = 3$



(b) $r = 5$



(c) $r = 10$

Figure 1: Relative MSE $\|\widehat{M} - M\|_F^2/\|M\|_F^2$ for varied values of $p = q$, $n = 200$, $\alpha = 1$, Gaussian noise with $\sigma = 0.18$, K=2: binary case, $w_1 = 0$, known bin boundaries, "trace-norm" refers to Davenport et al. (2014), the proposed Alg. 1 (proj-grad) coincides with the algorithm of Cai and Zhou (2013) when $K = 2$ and one picks $R \geq \sqrt{r}\alpha$ in Cai and Zhou (2013).

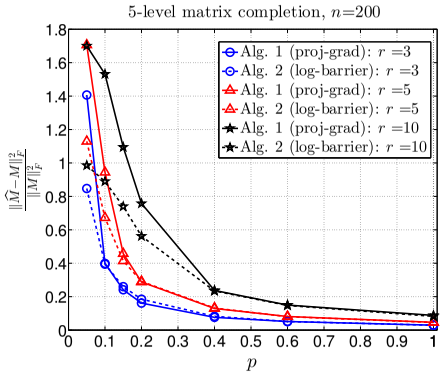


Figure 2: Relative MSE $\|\widehat{M} - M\|_F^2/\|M\|_F^2$ for varied values of $p = q$, $n = 200$, $\alpha = 1$, Gaussian noise with $\sigma = 0.18$, K=5: $w_1 = -0.4$, $w_2 = -0.15$, $w_3 = 0.15$, $w_4 = 0.4$, known bin boundaries.

i.i.d. entries drawn from a uniform distribution on $[-0.5, 0.5]$. Then we scaled M to achieve $\|M\|_\infty = 1 = \alpha$. We pick $r = 3$, 5 or 10, and vary matrix sizes $n = 100, 200$, or 400. We used the model (2) with Z_{ij} as a zero-mean Gaussian with standard deviation $\sigma = 0.18$. These choices follow the numerical experiments of Cai and Zhou (2013) and Davenport et al. (2014), which dealt with the case of binary observations (i.e., in which $K = 2$). We generate the set Ω of revealed indices via a stochastic block model as in Bhojanapalli and Jain (2014). In the basic stochastic block model, we divide the set of nodes $[n]$ into two clusters, where each intra-cluster edge is sampled uniformly with probability p and an inter-cluster edge is sampled with probability q. For our simulations, initially we chose $p = q$ which corresponds to the Bernoulli sampling model of Davenport et al. (2014). Then we change the fraction of revealed 1-bit entries as $p = 0.05, 0.1, 0.15, 0.2, 0.4, 0.6$ or 1. Algorithm 1 was implemented with random initialization and its result was used to initialize Algorithm 2 where we either picked λ via 5-fold cross-validation (how well the label values of revealed Y_{ij} in the test set are matched), or simply used a small fixed λ. We assumed the bin boundaries to be known. The resulting relative mean-square error (MSE) $\|\widehat{M} - M\|_F^2/\|M\|_F^2$, averaged over 20 Monte Carlo runs, is shown for $n = 200$ in Figure 1 for $K = 2$, and Figure 2 for $K = 5$. As expected, the performance improves with increasing n and increasing p. For comparison, Figure 1 also shows the MSE for Davenport et al. (2014) and Cai and Zhou (2013), and it is seen that Algorithm 2 (log-barrier) significantly outperforms Davenport et al. (2014) and Cai and Zhou (2013) for low values of p and high values of r (e.g., $r = 10$ and $p < 0.4$), and the performances are comparable for higher values of p and lower values of r (e.g., $r = 3$ and $p > 0.1$). Note that as aforementioned, Algorithm 1 (proj-grad) coincides with the algorithm of Cai and Zhou (2013) when $K = 2$ and one picks $R \geq \sqrt{r}\alpha$ in Cai and Zhou (2013).

In Figure 3 we show the results for the case of $r = 5$ and $m = n = 200$ for both known and estimated bin boundaries. As before, the results were averaged over 20 Monte Carlo runs, and the missing entries were set via the stochastic block model with $p = q$. For the case of unknown bin boundaries, we used Algorithm 3 with initialization $\omega_1^0 = -0.3$, $\omega_2^0 = -0.1$, $\omega_3^0 = 0.1$ and $\omega_4^0 = 0.3$, and $\alpha = 1$, $\lambda = 0.5$. Also shown are the results of the algorithm OptSpace of Keshavan et al. (2010) which is a matrix completion algorithm that assumes a real-valued low-rank matrix. The results using OptSpace were obtained for two cases: the case labeled "quantized noisy M" refers to the case where OptSpace is provided with revealed Y_{ij}'s, and the case labeled "unquantized noisy M" refers to the case where OptSpace works on revealed noisy M_{ij}s (i.e., $M_{ij} + Z_{ij}$s). For OptSpace, we scaled the estimate $\widehat{M}$ to have the same Frobenius norm as the true M before computing the MSE. It is seen that for $p \geq 0.2$, there is no loss in performance when bin boundaries are estimated, using the proposed approach. The algorithm OptSpace performs poorly for quantized data.

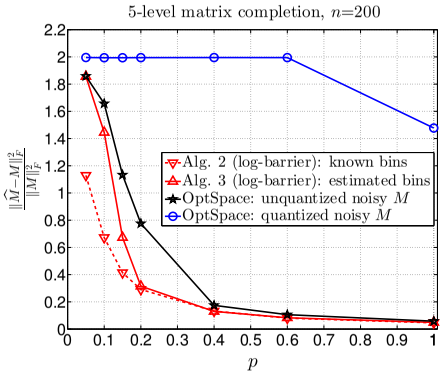


Figure 3: Relative MSE $\|\widehat{M} - M\|_F^2/\|M\|_F^2$ for varied values of $p = q$, $n = 200$, $\alpha = 1$, Gaussian noise with $\sigma = 0.18$, K=5: true bin boundaries $w_1 = -0.4$, $w_2 = -0.15$, $w_3 = 0.15$, $w_4 = 0.4$, known and estimated bin boundaries. OptSpace is the method of Keshavan et al. (2010).

In Figure 4, we show the results for varying number of quantization levels K, with $r = 3, 5, 10$, $p = 0.2$, $m = n = 200$ and known bin boundaries. The matrix M is constructed as for Figure 1. The results were over 20 Monte Carlo runs, and the missing entries were set via the stochastic block model with $p = q$. With $\alpha = 1$, the bin boundaries were picked as $w_1 = 0$ for $K = 2$, $w_1 = -0.2$, $w_2 = 0.2$ for $K = 3$, $w_1 = -0.25$, $w_2 = 0$, $w_3 = 0.25$ for $K = 4$, $w_1 = -0.4$, $w_2 = -0.15$, $w_3 = 0.15$, $w_4 = 0.4$ for $K = 5$ and $w_1 = -0.4$, $w_2 = -0.2$, $w_3 = -0.05$, $w_4 = 0.05$, $w_5 = 0.2$, $w_6 = 0.4$ for $K = 7$. These choices yield a comparable number of entries in each bin. It is seen from Figure 4 that performance improves with increasing K. This is not surprising since with increasing K, bin intervals

shrink and the quantization error becomes smaller. For the considered model there are two sources of error/noise: additive noise and quantization error.

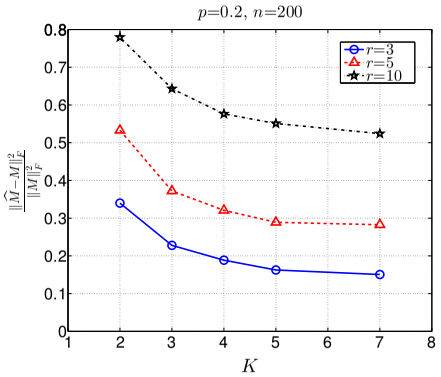
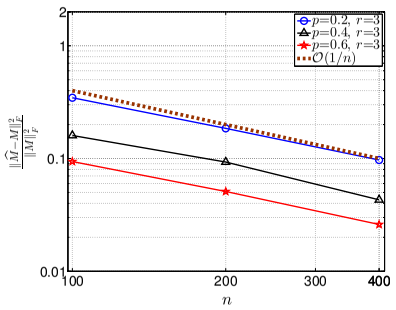


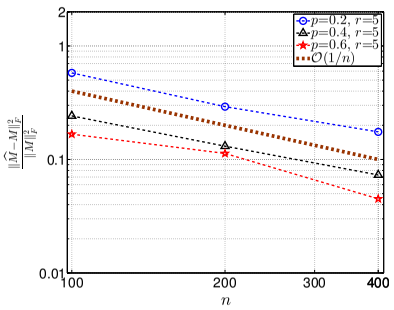
Figure 4: Relative MSE $\|\widehat{M} - M\|_F^2/\|M\|_F^2$ for varied values of K, $p = 0.2$, $n = 200$, $\alpha = 1$, Gaussian noise with $\sigma = 0.18$.

In Figure 5 we show the relative MSE for $r = 3$, 5 and 10, respectively, and $n = 100, 200, 400$, $p = q = 0.2, 0.4, 0.6$. In Section 3.2 (see Equation 22), the upper bound on MSE was established as $\min\left(\mathcal{O}\left(\frac{r^3}{p^4 n}\right), \mathcal{O}\left(\sqrt{\frac{r^3 \log(n)}{p^3 n}}\right)\right)$. Therefore, for fixed r and p, the bound is $\mathcal{O}\left(\frac{1}{n}\right)$, whereas for fixed n and p, the bound is $\mathcal{O}\left(r^{1.5}\right)$, and for fixed n and r, the bound is $\mathcal{O}\left(p^{-1.5}\right)$. We also plot the lines $1/n$ in Figure 5 to show the scale of the upper bound $\mathcal{O}(1/n)$ for fixed r and p. As we can see, the empirical estimation errors follow approximately the same scaling, suggesting that our analysis is tight with respect to n, up to some constant. In Figures 6 and 7 we show the relative MSE as a function of r and p, respectively, for $p = q = 0.2$ and $n = 100, 200, 400$. We also plot the lines $r^{1.5}$ and $1/p^{1.5}$ in Figures 6 and 7, respectively, to show the scale of the upper bound $\mathcal{O}\left(r^{1.5}\right)$ for fixed n and p, and the upper bound $\mathcal{O}\left(1/p^{1.5}\right)$ for fixed n and r. Now we see that these bounds are not tight. The empirical MSE results are approximately $\mathcal{O}(r)$ for fixed n and p and $\mathcal{O}(1/p)$ for fixed n and r.

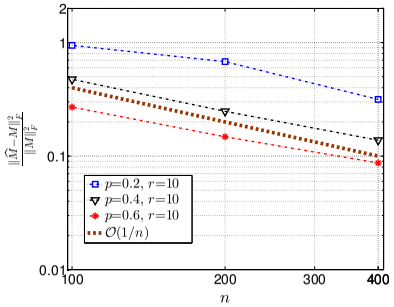
In Figure 8 we additionally plot the relative MSE for $n = 200$ and rank $r = 5$, via the same method described above, but with varying p and keeping $p+q = 0.7$, under the probit model. This enables us to study the performance of the model under nonuniform sampling. Note that when $p = q = 0.35$, then the spectral gap is largest (Bhojanapalli and Jain, 2014) and the MSE is the smallest, and as p gets larger, the spectral gap decreases, leading to larger MSE.



(a) $r = 3$



(b) $r = 5$



(c) $r = 10$

Figure 5: Relative MSE: $K = 5$, $p = q$, $r = 3$, 5 or 10, $n = 100$, 200, 400, known bin boundaries, $\alpha = 1$, Gaussian noise $\sigma = 0.18$.

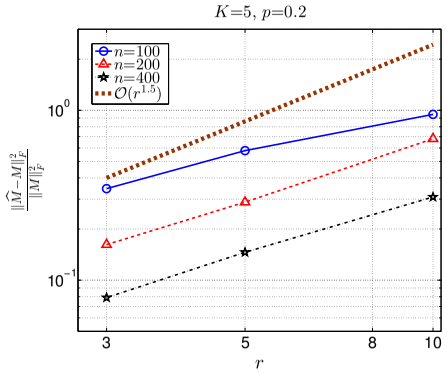


Figure 6: Relative MSE $\|\widehat{M} - M\|_F^2 / \|M\|_F^2$ for varied values of r, $p = 0.2$, $K = 5$, $\alpha = 1$, Gaussian noise with $\sigma = 0.18$.

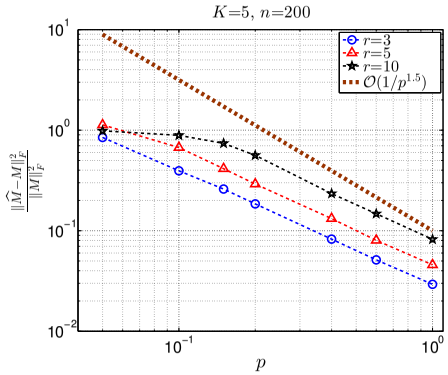


Figure 7: Relative MSE $\|\widehat{M} - M\|_F^2 / \|M\|_F^2$ for varied values of p, $n = 200$, $K = 5$, $\alpha = 1$, Gaussian noise with $\sigma = 0.18$.

5.2 MovieLens Dataset

Now we consider the MovieLens 1M dataset (available from http://www.grouplens.org) consisting of 1,000,000 movie ratings on a scale from 1 to 5, from 6040 users on 3952 movies (95.8% missing entries). A given set of ratings has a matrix representation with rows representing the users and columns representing the movies, and the (i, j)th entry of the matrix is non-zero if user i has given a rating for movie j. Thus estimating the remaining ratings in the matrix corresponds to a matrix completion problem. We consider

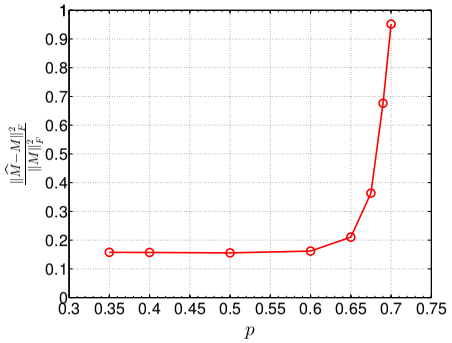


Figure 8: Relative MSE versus p, known bin boundaries, fixed $p+q=0.7$, $K=5$, $n=200$, Gaussian noise $\sigma=0.18$.

20 independent realizations of 80%/20% training/test splits of the 1 million revealed entries. For each data split, we train the proposed Algorithms 1 and 3, and the approaches OptSpace (Keshavan et al., 2009, 2010) and SVT (Cai et al., 2010), on the training set and compare their performance on the corresponding test set in predicting the revealed matrix entries in the test set. For implementation of OptSpace and SVT, we used the code made available by the authors online. Let Ω_T denote the test set, Y_{ij} the original rating in the data set and $\widehat{Y}_{ij}$ the predicted rating of user i for movie j. For performance assessment, we use the normalized mean absolute error (NMAE) and the root mean-square error (RMSE) in prediction on test set, defined as

$$RMSE = \sqrt{\frac{1}{|\Omega_T|}\sum_{(i,j)\in\Omega_T}\left(Y_{ij}-\widehat{Y}_{ij}\right)^2},$$

$$NMAE = \frac{1}{|\Omega_T|(Y_{max}-Y_{min})}\sum_{(i,j)\in\Omega_T}|Y_{ij}-\widehat{Y}_{ij}|,$$

where Y_{max} and Y_{min} are the upper and lower bounds, respectively, on the ratings (5 and 1, respectively). Both metrics are widely used for evaluation of prediction accuracy (Gunawardana and Shani, 2009); for instance, RMSE has been used in Salakhutdinov and Mnih (2008) and NMAE in Keshavan et al. (2009).

Remark 2 *Estimation of missing entries of Y. In many applications such as this one, one may wish to estimate missing discrete Y_{ij}s instead of (or in addition to) the continuous-valued M. We will use the model (1). If one wishes to pick the estimate $\widehat{Y}_{ij}$ of Y_{ij} given M_{ij}, to minimize the MSE $\mathbb{E}\{[\widehat{Y}_{ij}-Y_{ij}]^2\}$, then $\widehat{Y}_{ij}=\mathbb{E}\{Y_{ij}\,|\,M_{ij}\}=\sum_{\ell=1}^{K}\ell\, f_\ell(M_{ij})$. If, on the other hand, the optimality criterion is the MAE $\mathbb{E}\{|\widehat{Y}_{ij}-Y_{ij}|\}$, then given M_{ij}, $\widehat{Y}_{ij}$ is a*

MovieLens 1M		
Approach	RMSE	NMAE
Alg. 1 (proj-grad): logistic	0.8698 ± 0.0029	0.1590 ± 0.0004
Alg. 3 (log-barrier + unknown bins): logistic	**0.8568** ± 0.0014	**0.1559** ± 0.0004
Alg. 3 (log-barrier + unknown bins): probit	0.8580 ± 0.0027	0.1561 ± 0.0005
OptSpace	0.8947 ± 0.0033	0.1767 ± 0.0006
SVT	0.9023 ± 0.0014	0.1754 ± 0.0003

Table 1: Test data RMSE and NMAE (± one standard deviation) averaged over 20 realizations of 80%/20% training/test splits drawn from MovieLens 1M data

median of conditional distribution $f_\ell(M_{ij})$. For the model (1)-(4), if $\Phi(0)=0.5$ (true for the logistic and probit models considered in this paper), then a median of $f_\ell(M_{ij})$ is given by $\mathcal{Q}(M_{ij})$. These estimators are used with M_{ij} replaced with the estimated $\widehat{M}_{ij}$.

For Algorithm 1 we used $\alpha=1$, $\text{rank}(M)=7$, the logistic model with $\sigma=1/16$, and considered fixed bin boundaries $\omega_1^0=-0.6$, $\omega_2^0=-0.2$, $\omega_3^0=0.2$ and $\omega_4^0=0.6$ spaced (arbitrarily) uniformly over $[-\alpha,\alpha]$ to get equal width bins. An alternative initialization would be to pick the bin boundaries to match the distribution of the revealed entries. Let p_ℓ denote the fraction of revealed entries with $Y_{ij}=\ell$. Then a reasonable choice is to satisfy $\Phi(\omega_\ell)-\Phi(\omega_{\ell-1})=p_\ell$, leading to $\omega_0^0=-\infty$, $\omega_L^0=\infty$, and $\omega_\ell^0=\Phi^{-1}(\sum_{i=1}^{\ell}p_i)$ for $\ell=1,\cdots,L-1$. For a given choice of α, this requires a proper choice of σ to ensure that ω_ℓ, $\ell=1,\cdots,L-1$, are within $[-\alpha,\alpha]$. Note that scaling the variables, M, α, σ, and the bin boundaries by the same factor will lead to the same likelihood for observed data. Since we fixed (arbitrarily) $\alpha=1$, different values of σ, the bin boundaries, and X will lead to a different likelihood for the observed data.

For Algorithm 3, based on additional optimization w.r.t. $\boldsymbol{\omega}$, we used $\alpha=1$, $\text{rank}(M)=7$, initialization $\omega_1^0=-0.6$, $\omega_2^0=-0.2$, $\omega_3^0=0.2$ and $\omega_4^0=0.6$, and either the logistic model with $\sigma=1/16$, or the probit model with $\sigma=1/13$. The results (RMSE and NMAE) averaged over 20 runs are shown in Table 1. It is seen that our methods outperform OptSpace and SVT under both performance measures, and fixed bin boundaries also yield useful and improved predictions. Tran et al. (2012) reported the results shown in Table 2 for their Matrix Cumulative RBM (restricted Boltzmann machines) based method and the OrdRec method of Koren and Sill (2011, 2013) when tested on the MovieLens 1M dataset; both these approaches are based on a quantization observation model. Comparing Tables 1 and 2 we see that our methods outperform OrdRec and Matrix Cumulative RBM under both performance measures.

Acknowledgments

I would like to thank Andrea Montanari and Adel Javanmard for helpful discussions.

MovieLens 1M: Results from Tran et al. (2012)						
	r=50		r=100		r=200	
Approach	RMSE	NMAE	RMSE	NMAE	RMSE	NMAE
OrdRec	0.904	0.1705	0.902	0.1705	0.902	0.1700
Matrix Cumulative RBM	0.904	0.1665	0.904	0.1655	0.906	0.1660

Table 2: RMSE and NMAE results from Tran et al. (2012) for MovieLens 1M data for various values of r=rank(M). OrdRec is the method of Koren and Sill (2011, 2013) and Matrix Cumulative RBM is the restricted Boltzman machines based method of Tran et al. (2012)

Appendix A. Proof of a Technical Lemma

Here we provide a proof of the assertion made in Section 3.1 (after Equation 9) that $f_\ell(X_{ij})$ is log-concave in ω_k for fixed X and ω_is ($i \neq k$) for log-concave $\Phi(x)$.

Lemma 3 *The probability $f_\ell(X_{ij})$ defined in (4) is log-concave in ω_k for fixed X and ω_is ($i \neq k$) for log-concave $\Phi(x)$.*

Proof We have $f_\ell(x) = \Phi(\omega_\ell - x) - \Phi(\omega_{\ell-1} - x)$. Therefore, it is obviously log-concave in ω_i, $i \in [K-1]$, $i \neq \ell$ or $\ell - 1$. By p. 121, Problem 3.48, of Boyd and Vandenberghe (2004) $\Phi(\omega_\ell - x) - \Phi(\omega_{\ell-1} - x)$ is log-concave in ω_ℓ for fixed x and $\omega_{\ell-1}$. To show that $\Phi(\omega_\ell - x) - \Phi(\omega_{\ell-1} - x)$ is log-concave in $\omega_{\ell-1}$ for fixed x and ω_ℓ, we will modify a proof given in Prop. 1 of An (1995) to prove log-concavity of $1 - \Phi(x)$ in x. Let $\phi(x) = \frac{d\Phi(x)}{dx} \geq 0$. Then with $y_0 = \omega_\ell - x$ and $y = \omega_{\ell-1} - x$, we have $y_0 > y$ for every x,

$$s(y) := \Phi(\omega_\ell - x) - \Phi(\omega_{\ell-1} - x) = \int_y^{y_0} \phi(u)\, du \geq 0$$

and

$$h(y) := -\frac{d \log s(y)}{dy} = \frac{\phi(y)}{s(y)}.$$

By Prop. 1 of An (1995), $s(y)$ is log-concave iff $h(y)$ is non-decreasing is y. For $y_1 < y_2$, we have

$$\begin{aligned}
& h(y_2) - h(y_1) \geq 0 \\
\iff & \phi(y_2)s(y_1) - \phi(y_1)s(y_2) \\
& = \phi(y_2) \int_{y_1}^{y_0} \phi(u)\, du - \phi(y_1) \int_{y_2}^{y_0} \phi(u)\, du \\
& = \int_0^{y_0 - y_2} [\phi(y_2)\phi(y_1 + v) - \phi(y_1)\phi(y_2 + v)]\, dv + \int_{y_0 - y_2}^{y_0 - y_1} \phi(y_2)\phi(y_1 + v) dv \\
& \geq 0
\end{aligned}$$

where we have used the fact that $\int_{y_0 - y_2}^{y_0 - y_1} \phi(y_2)\phi(y_1 + v) dv \geq 0$ since the integrand is non-negative and $y_1 < y_2$, and since $\phi(x)$ is log-concave, by Lemma 1 of An (1995)

$$\int_0^{y_0 - y_2} [\phi(y_2)\phi(y_1 + v) - \phi(y_1)\phi(y_2 + v)]\, dv \geq 0.$$

Thus, for fixed x and ω_ℓ, $s(y)$ is log-concave in y, hence, in $\omega_{\ell-1}$. ■

Appendix B. Proof of Theorem 1

Our proof is based on a second-order Taylor series expansion and a matrix concentration inequality. Let $\theta = \text{vec}(X) \in \mathbb{R}^{mn}$ and $\tilde{F}_{\Omega,Y}(\theta) = F_{\Omega,Y}(X)$. The objective function $F_{\Omega,Y}(X)$ is continuous in X and the set $\mathcal{C}$ is compact, therefore, $F_{\Omega,Y}(X)$ achieves a minimum in $\mathcal{C}$. If $\hat{\theta} = \text{vec}(\widehat{M})$ minimizes $\tilde{F}_{\Omega,Y}(\theta)$ subject to the constraints, then $\tilde{F}_{\Omega,Y}(\hat{\theta}) \leq \tilde{F}_{\Omega,Y}(\theta^*)$ where $\theta^* = \text{vec}(M)$. By the second-order Taylor's theorem, expanding around θ^* we have

$$\tilde{F}_{\Omega,Y}(\theta) = \tilde{F}_{\Omega,Y}(\theta^*) + \langle \nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*), \theta - \theta^* \rangle + \frac{1}{2} \langle \theta - \theta^*, \left(\nabla^2_{\theta\theta} \tilde{F}_{\Omega,Y}(\tilde{\theta}) \right) (\theta - \theta^*) \rangle \qquad (33)$$

where $\tilde{\theta} = \theta^* + \gamma(\theta - \theta^*)$ for some $\gamma \in [0, 1]$, with corresponding matrices $\tilde{X} = M + \gamma(X - M)$. We need several auxiliary results before we can prove Theorem 1.

We need the following result from Chatterjee (2013) concerning spectral norms of random matrices for Lemma 5.

Lemma 4 *(Theorem 8.4 of Chatterjee (2013)) Take any two numbers m and n such that $1 \leq n \leq m$. Suppose that $A = [a_{ij}]_{1 \leq i \leq m, 1 \leq j \leq n}$ is a matrix whose entries are independent random variables that satisfy, for some $\sigma^2 \in [0, 1]$,*

$$\mathbb{E}[a_{ij}] = 0,\ \mathbb{E}[a_{ij}^2] \leq \sigma^2,\ \text{and } |a_{ij}| \leq 1\ a.s.$$

Suppose that $\sigma^2 \geq m^{-1+\varepsilon}$ for some $\varepsilon > 0$. Then

$$P\left(\|A\|_2 \geq 2.01\sigma\sqrt{m} \right) \leq C_1(\varepsilon) e^{-C_2 \sigma^2 m},$$

where $C_1(\varepsilon)$ is a constant that depends only on ε and C_2 is a positive universal constant. The same result is true when $m = n$ and A is symmetric or skew-symmetric, with independent entries on and above the diagonal, all other assumptions remaining the same. Lastly, all results remain true if the assumption $\sigma^2 \geq m^{-1+\varepsilon}$ is changed to $\sigma^2 \geq m^{-1}(\log(m))^{6+\varepsilon}$.

Using (6), it follows that

$$\frac{\partial F_{\Omega,Y}(X)}{\partial X_{ij}} = -\left(\sum_{\ell=1}^{K} \frac{\dot{f}_\ell(X_{ij})}{f_\ell(X_{ij})} \mathbf{1}_{[Y_{ij} = \ell]} \right) \mathbf{1}_{[(i,j) \in \Omega]}, \qquad (34)$$

$$\frac{\partial^2 F_{\Omega,Y}(X)}{\partial X_{ij}^2} = \sum_{\ell=1}^{K} \left(\frac{\dot{f}_\ell^2(X_{ij})}{f_\ell^2(X_{ij})} - \frac{\ddot{f}_\ell(X_{ij})}{f_\ell(X_{ij})} \right) \mathbf{1}_{[Y_{ij} = \ell]} \mathbf{1}_{[(i,j) \in \Omega]} \qquad (35)$$

and

$$\frac{\partial^2 F_{\Omega,Y}(X)}{\partial X_{i_1 j_1} \partial X_{i_2 j_2}} = 0 \text{ if } (i_1, j_1) \neq (i_2, j_2). \tag{36}$$

Let $w = \mathrm{vec}(X^{(1)} - M) = \theta^{(1)} - \theta^*$; for later use, we would like $\theta^{(1)}$ in w to be not necessarily equal to θ in the gradient or the Hessian of the objective function. Using (34) and the notation

$$\nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*) = \mathrm{vec}\left(\left[\frac{\partial F_{\Omega,Y}(X)}{\partial X_{ij}} \right] \Big|_{X=M} \right) = \mathrm{vec}\left(\left[\frac{\partial F_{\Omega,Y}(M)}{\partial X_{ij}} \right] \right),$$

we have

$$\langle \nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*), w \rangle = \langle \nabla_X F_{\Omega,Y}(M), X^{(1)} - M \rangle \tag{37}$$

where $\langle A, B \rangle := \mathrm{tr}(A^\top B)$, $|\langle A, B \rangle| \leq \|A\|_2 \|B\|_*$, $\|B\|_*$ is the nuclear (or Schatten) norm of B, and

$$[\nabla_X F_{\Omega,Y}(M)]_{ij} =: z_{ij} = -\left(\sum_{\ell=1}^{K} \frac{\dot{f}_\ell(M_{ij})}{f_\ell(M_{ij})} \mathbf{1}_{[Y_{ij}=\ell]} \right) \mathbf{1}_{[(i,j)\in\Omega]}. \tag{38}$$

Using (1), (11), and the fact that $\sum_{\ell=1}^{K} f_\ell(X_{ij}) = 1$, we have

$$\mathbb{E}[z_{ij}] = -\left(\sum_{\ell=1}^{K} \dot{f}_\ell(M_{ij}) \right) \mathbf{1}_{[(i,j)\in\Omega]} = 0, \tag{39}$$

and

$$|z_{ij}| \leq L_\alpha \implies \mathbb{E}[z_{ij}^2] \leq L_\alpha^2. \tag{40}$$

Lemma 5 *Let $w = \mathrm{vec}(X^{(1)} - M) = \theta^{(1)} - \theta^*$ and $X^{(1)}, M \in \mathcal{C}$. Then with probability at least $1 - C_1(\varepsilon)\exp(-C_2 m)$, we have*

$$\left| \langle \nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*), w \rangle \right| \leq 2.01 L_\alpha \sqrt{2rm} \|X^{(1)} - M\|_F,$$

where $\varepsilon \in (0,1)$, $C_1(\varepsilon)$ is a constant that depends only on ε and C_2 is a positive universal constant.

Proof Using (37), we have

$$\begin{aligned} |\langle \nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*), w \rangle| &= |\langle \nabla_X F_{\Omega,Y}(M), X^{(1)} - M \rangle| \\ &\leq \|\nabla_X F_{\Omega,Y}(M)\|_2 \|X^{(1)} - M\|_*. \end{aligned} \tag{41}$$

Consider $\tilde{z}_{ij} := \left[L_\alpha^{-1} \nabla_X F_{\Omega,Y}(M) \right]_{ij}$. Then we have $\mathbb{E}[\tilde{z}_{ij}] = 0$, $|\tilde{z}_{ij}| \leq 1$ and $\mathbb{E}[\tilde{z}_{ij}^2] \leq 1$. We will apply Lemma 4 to $L_\alpha^{-1} \nabla_X F_{\Omega,Y}(M)$, for which we have to ensure that $\mathbb{E}[\tilde{z}_{ij}^2] \leq \sigma^2$ and $m^{-1+\varepsilon} \leq \sigma^2$ for some $\varepsilon > 0$. Therefore, we pick $\sigma^2 = 1 = \max\left(1, \frac{1}{m^{1-\varepsilon}}\right)$. Hence, by Lemma 4, $\|L_\alpha^{-1} \nabla_X F_{\Omega,Y}(M)\|_2 \leq 2.01\sqrt{m}$ with probability at least $1 - C_1(\varepsilon)\exp(-C_2 m)$ for some positive constants $C_1(\varepsilon)$ and C_2. Since for any rank r matrix A, $\|A\|_* \leq \sqrt{r}\|A\|_F$, we have $\|X^{(1)} - M\|_* \leq \sqrt{2r}\|X^{(1)} - M\|_F$, yielding the desired result. ■

Lemma 6 *Let $w = \mathrm{vec}(X - M) = \theta - \theta^*$ and $X, M \in \mathcal{C}$. Then with probability at least $1 - 2(9\alpha\sqrt{mn})^{-r(m+n+1)} - C_1\exp(-C_2 m)$, we have*

$$\left| \langle \nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*), w \rangle \right| \leq 4(1+\alpha) L_\alpha \sqrt{|\Omega| r (m+n+1) \log(9\alpha\sqrt{mn})}.$$

Proof Define the set $S_{rK} = \{X \in \mathbb{R}^{m\times n} : \mathrm{rank}(X) \leq r,\ \|X\|_F \leq K\}$ for some $K > 0$. By Lemma A.2 in the supplementary material of Wang and Xu (2012) (which is based on Lemma 3.1 in Candes and Plan (2011)), there exists a 1-net for the Frobenius norm, $\overline{S}_{rK}(1) = \{Q \in \mathbb{R}^{m\times n} : \mathrm{rank}(Q) \leq r,\ \|X - Q\|_F \leq 1\} \subset S_{rK}$ with its cardinality $|\overline{S}_{rK}(1)| \leq (9K)^{(m+n+1)r}$. That is, given any $X \in S_{rK}$, there exists $Q \in \overline{S}_{rK}(1) \subset S_{rK}$ such that $\|X - Q\|_F \leq 1$. Suppose that $X \in S_{rK}$ is such that $\|X\|_\infty \leq \alpha$. Then for $Q \in \overline{S}_{rK}(1)$, we have

$$\|Q\|_\infty \leq \|Q - X\|_\infty + \|X\|_\infty \leq \|Q - X\|_F + \|X\|_\infty \leq 1 + \alpha. \tag{42}$$

Also, $\|X\|_F \leq \sqrt{mn}\|X\|_\infty \leq \alpha\sqrt{mn}$. For $X, M \in S_{rK}$ and $Q \in \overline{S}_{rK}(1)$, consider

$$\begin{aligned} u_X &:= \langle \nabla_\theta \tilde{F}_{\Omega,Y}(\theta^*), w \rangle = \langle \nabla_X F_{\Omega,Y}(M), X - M \rangle \\ &= \langle \nabla_X F_{\Omega,Y}(M), Q - M \rangle + \langle \nabla_X F_{\Omega,Y}(M), X - Q \rangle = u_Q + \langle \nabla_X F_{\Omega,Y}(M), X - Q \rangle. \end{aligned} \tag{43}$$

Therefore, for any $X \in S_{rK}$ and corresponding $Q \in \overline{S}_{rK}(1)$, we have

$$\begin{aligned} |u_X| &\leq |u_Q| + |\langle \nabla_X F_{\Omega,Y}(M), X - Q \rangle| \\ &\leq |u_Q| + \|\nabla_X F_{\Omega,Y}(M)\|_2 \|X - Q\|_* \\ &\leq |u_Q| + 2.01 L_\alpha \sqrt{2rm} \ \text{ with probability } \geq 1 - C_1\exp(-C_2 m) \end{aligned} \tag{44}$$

where in the last inequality we have used Lemma 5 with $C_1 = C_1(1/2)$, and the fact that $\|X - Q\|_F \leq 1$ and both X and Q are of rank r. Now consider u_Q and rewrite it as

$$u_Q = \sum_{(i,j)\in\Omega} h_{ij} \ \text{ where } \ h_{ij} = \frac{\partial F_{\Omega,Y}(M)}{\partial X_{ij}} (Q_{ij} - M_{ij}).$$

We have $\mathbb{E}[h_{ij}] = 0$ (see Equations 38 and 39), and

$$\begin{aligned} |h_{ij}| &\leq \left| \frac{\partial F_{\Omega,Y}(M)}{\partial X_{ij}} \right| \times |Q_{ij} - M_{ij}| \\ &\leq L_\alpha(|Q_{ij}| + |M_{ij}|) \leq L_\alpha(1 + 2\alpha) =: \beta_\alpha. \end{aligned} \tag{45}$$

Apply the Hoeffding inequality to u_Q to obtain

$$\mathbb{P}\left(|u_Q| > t \right) \leq 2\exp\left(-2\frac{t^2}{|\Omega|\beta_\alpha^2} \right).$$

Set $K = \alpha\sqrt{mn}$ (since $\|X\|_F \leq \alpha\sqrt{mn}$) and apply the union bound over all $Q \in \overline{S}_{rK}(1)$ to obtain

$$\begin{aligned} \mathbb{P}\left(\cup_{Q\in\overline{S}_{rK}(1)} \{|u_Q| > t\} \right) &\leq 2\left(9\alpha\sqrt{mn}\right)^{(m+n+1)r} \exp\left(-2\frac{t^2}{|\Omega|\beta_\alpha^2} \right) \\ &\leq 2\exp\left(-\frac{2t^2}{|\Omega|\beta_\alpha^2} + (m+n+1)r\log\left(9\alpha\sqrt{mn}\right) \right). \end{aligned}$$

We pick

$$t = \beta_\alpha \sqrt{|\Omega|(m+n+1)r\log\left(9\alpha\sqrt{mn}\right)} \tag{46}$$

to achieve

$$\begin{aligned}\mathbb{P}\left(\cup_{Q\in\overline{S}_{rK}(1)}\{|u_Q|>t\}\right) &\le 2\exp\left(-(m+n+1)r\log\left(9\alpha\sqrt{mn}\right)\right)\\ &= \frac{2}{(9\alpha\sqrt{mn})^{r(m+n+1)}}.\end{aligned} \tag{47}$$

Now using (44)-(47), the union bound and the fact that the chosen $t > 2.01L_\alpha\sqrt{2rm}$, we have the desired result. ■

Lemma 7 *Let $w = \mathrm{vec}(X^{(1)} - M) = \theta^{(1)} - \theta^*$ and $X^{(1)}, X, M \in \mathcal{C}$. Then for any $\tilde{\theta} = \theta^* + \gamma(\theta - \theta^*)$ and any $\gamma \in [0,1]$, we have*

$$\langle w, \left[\nabla^2_{\theta\theta}\tilde{F}_{\Omega,Y}(\tilde{\theta})\right] w\rangle \ge \gamma_\alpha \left\|\left(X^{(1)} - M\right)_\Omega\right\|_F^2$$

where $X_\Omega = X_{ij}$ if $(i,j)\in\Omega$, and $= 0$ otherwise.

Proof Using (10), (35) and (36), we have

$$\begin{aligned}\langle w, \left[\nabla^2_{\theta\theta}\tilde{F}_{\Omega,Y}(\tilde{\theta})\right] w\rangle &= \sum_{(i,j)\in\Omega}\left(\frac{\partial^2 F_{\Omega,Y}(\tilde{X})}{\partial X_{ij}^2}\right)(X^{(1)}_{ij} - M_{ij})^2\\ &\ge \gamma_\alpha \sum_{(i,j)\in\Omega}(X^{(1)}_{ij} - M_{ij})^2 = \gamma_\alpha\left\|\left(X^{(1)} - M\right)_\Omega\right\|_F^2.\end{aligned} \tag{48}$$

■

We need a result similar to Theorem 4.1 of Bhojanapalli and Jain (2014) regarding closeness of a fixed matrix to its sampled version, which is proved therein for square matrices M under an incoherence assumption on M. In Lemma 8 we prove a similar result for rectangular Z with bounded $\|Z\|_\infty$. Take $Z \in \mathbb{R}^{m\times n}$, $m \ge n$, and as in Lemma 7, define the operator $\mathcal{R}_\Omega$ as $Z_\Omega := \mathcal{R}_\Omega(Z) = Z_{ij}$ if $(i,j)\in\Omega$, and $= 0$ otherwise.

Lemma 8 *Let $G\backslash\Omega$ satisfy assumptions (A1) and (A2) in Section 2. Let $Z \in \mathbb{R}^{m\times n}$ have rank $\le r$. Then we have*

$$\left\|\left(\frac{\sqrt{mn}}{\sigma_1(G)}R_\Omega - I\right)(Z)\right\|_2 \le \frac{\sqrt{mn}\sigma_2(G)}{\sigma_1(G)}\|Z\|_{\max} \tag{49}$$

$$\le \frac{\sqrt{rmn}\sigma_2(G)}{\sigma_1(G)}\|Z\|_\infty \le Cm\sqrt{\frac{nr}{|\Omega|}}\|Z\|_\infty. \tag{50}$$

Proof Normalize $\mathbf{1}_m$ to unit norm as $\tilde{\mathbf{1}}_m = \mathbf{1}_m/\sqrt{m}$, and similarly for $\tilde{\mathbf{1}}_n$. It then follows from the properties (A1)-(A2) that

$$G = \sigma_1(G)\tilde{\mathbf{1}}_m\tilde{\mathbf{1}}_n^\top + \sum_{i=2}^n \sigma_i(G)U_iV_i^\top \tag{51}$$

where the SVD of G is $G = U\Sigma_G V^\top$ and U_i is the i-th column of U. First some preliminaries. If $\mathrm{rank}(Z) \le r$, then we have $\|Z\|_{\max} \le \sqrt{r}\|Z\|_\infty$. By the factored form definition of the max norm (Lee et al., 2010), we have $\|Z\|_{\max} = \inf\left\{\max(\|\bar{U}\|^2_{2,\infty}, \|\bar{V}\|^2_{2,\infty}) : Z = \bar{U}\bar{V}^\intercal\right\}$ where $\|\bar{U}\|_{2,\infty} = \max_i\sqrt{\sum_j \bar{U}_{ij}^2}$, $\bar{U}\in\mathbb{R}^{m\times k}$, $\bar{V}\in\mathbb{R}^{n\times k}$, $k = 1, 2, \cdots, \min(m,n) = n$. Hence, there exist $U_Z \in \mathbb{R}^{m\times k}$ and $V_Z \in \mathbb{R}^{n\times k}$ for some $1 \le k \le \min(m,n)$ such that $Z = U_Z V_Z^\top$, $\|U_Z\|^2_{2,\infty} \le \|Z\|_{\max}$ and $\|V_Z\|^2_{2,\infty} \le \|Z\|_{\max}$. For Z of rank $\le r$, one should have $k \le r$, but this fact is not needed in our proof. We now follow the proof of Theorem 4.1 of Bhojanapalli and Jain (2014) with $Z = \sum_{i=1}^k U_{Zi}V_{Zi}^\top$. Note that

$$\|\frac{\sqrt{mn}}{\sigma_1(G)}R_\Omega(Z) - Z\|_2 = \max_{x,y:\,\|x\|_2=1=\|y\|_2} y^\top\left(\frac{\sqrt{mn}}{\sigma_1(G)}R_\Omega(Z) - Z\right)x.$$

As in the proof of Theorem 4.1 of Bhojanapalli and Jain (2014), noting that $R_\Omega(Z) = Z\circ G$ where $\circ$ denotes the Hadamard (elementwise) product, we have

$$y^\top\left(\frac{\sqrt{mn}}{\sigma_1(G)}\mathcal{R}_\Omega(Z) - Z\right)x = \sum_{i=1}^k\left(\frac{\sqrt{mn}}{\sigma_1(G)}(y\circ U_{Zi})^\top G(x\circ V_{Zi}) - (y^\top U_{Zi})(x^\top V_{Zi})\right). \tag{52}$$

Let $y\circ U_{Zi} = \alpha_i\tilde{\mathbf{1}}_m + \beta_i\tilde{\mathbf{1}}^i_{m\perp}$ where $\tilde{\mathbf{1}}^i_{m\perp}$ is a unit norm vector orthogonal to $\tilde{\mathbf{1}}_m$. Then $\alpha_i = \tilde{\mathbf{1}}_m^\top(y\circ U_{Zi}) = y^\top U_{Zi}/\sqrt{m}$. Using the fact that $\tilde{\mathbf{1}}_m^\top G = \sigma_1(G)\tilde{\mathbf{1}}_n^\top$, we have

$$\begin{aligned}&y^\top\left(\frac{\sqrt{mn}}{\sigma_1(G)}\mathcal{R}_\Omega(Z) - Z\right)x\\ &= \sum_{i=1}^k\left(\frac{\sqrt{mn}}{\sigma_1(G)}\left[(1/\sqrt{m})y^\top U_{Zi}\tilde{\mathbf{1}}_m^\top G(x\circ V_{Zi}) + \beta_i\tilde{\mathbf{1}}^{i\top}_{m\perp}G(x\circ V_{Zi})\right] - (y^\top U_{Zi})(x^\top V_{Zi})\right)\\ &= \sum_{i=1}^k\left(\frac{\sqrt{mn}}{\sigma_1(G)}\beta_i\tilde{\mathbf{1}}^{i\top}_{m\perp}G(x\circ V_{Zi})\right)\end{aligned} \tag{53}$$

where we have also used $\tilde{\mathbf{1}}_n^\top(x\circ V_{Zi}) = x^\top V_{Zi}/\sqrt{n}$. Using the SVD (51) of G, we have

$$\tilde{\mathbf{1}}^{i\top}_{m\perp}G = \sum_{\ell=2}^n \sigma_\ell(G)(\tilde{\mathbf{1}}^{i\top}_{m\perp}U_\ell)V_\ell^\top$$

$$\implies |\tilde{\mathbf{1}}^{i\top}_{m\perp}Gz| \le \sigma_2(G)\|z\|_2 \text{ for any } z\in\mathbb{R}^n.$$

Using the above inequality in (53) we obtain

$$y^\top \left(\frac{\sqrt{mn}}{\sigma_1(G)} R_\Omega(Z) - Z \right) x \leq \frac{\sqrt{mn}}{\sigma_1(G)} \sigma_2(G) \sum_{i=1}^{k} |\beta_i| \|x \circ V_{Zi}\|_2$$
$$\leq \frac{\sqrt{mn}}{\sigma_1(G)} \sigma_2(G) \sqrt{\sum_{i=1}^{k} \beta_i^2} \sqrt{\sum_{i=1}^{k} \|x \circ V_{Zi}\|_2^2} \,. \tag{54}$$

We have $\beta_i = \tilde{\mathbf{1}}_{m\perp}^{i\top}(y \circ U_{Zi})$, hence, $|\beta_i| \leq \|(y \circ U_{Zi})\|_2$. Therefore,

$$\sum_{i=1}^{k} \beta_i^2 \leq \sum_{i=1}^{k} \|(y \circ U_{Zi})\|_2^2 = \sum_{j=1}^{m} \sum_{i=1}^{k} y_j^2 U_{Zji}^2$$
$$\leq \sum_{j=1}^{m} y_j^2 \|U_Z^j\|_2^2 \leq \|U_Z\|_{2,\infty}^2 \sum_{j=1}^{m} y_j^2 \leq \|Z\|_{\max} \tag{55}$$

where U_Z^j denotes the jth row of U_Z and $\sum_{j=1}^m y_j^2 = 1$. Similarly, we have

$$\sum_{i=1}^{k} \|x \circ V_{Zi}\|_2^2 = \sum_{j=1}^{n} \sum_{i=1}^{k} x_j^2 V_{Zji}^2 \leq \sum_{j=1}^{n} x_j^2 \|\tilde{V}_Z^j\|_2^2$$
$$\leq \|\tilde{V}_Z\|_{2,\infty}^2 \sum_{j=1}^{n} x_j^2 \leq \|Z\|_{\max} \,. \tag{56}$$

It then follows from (54)-(56) that

$$y^\top \left(\frac{\sqrt{mn}}{\sigma_1(G)} R_\Omega(Z) - Z \right) x \leq \frac{\sqrt{mn}\sigma_2(G)}{\sigma_1(G)} \|Z\|_{\max}$$
$$\implies \|\frac{\sqrt{mn}}{\sigma_1(G)} R_\Omega(Z) - Z\|_2 \leq \frac{\sqrt{mn}\sigma_2(G)}{\sigma_1(G)} \|Z\|_{\max}. \tag{57}$$

This establishes (49). Now use the facts $\|Z\|_{\max} \leq \sqrt{r}\|Z\|_\infty$ and $|\Omega| = md$ to establish (50). ■

Lemma 9 *Let $X, M \in \mathcal{C}$. Then we have*

$$\|(X - M)_\Omega\|_F \geq \frac{\sigma_1(G)}{\sqrt{2rmn}} \|X - M\|_F - 2\alpha\sqrt{r}\sigma_2(G).$$

Proof Let $Z = X - M$, $a = \sqrt{mn}/\sigma_1(G)$, and $b = (\sigma_2(G)/\sigma_1(G))\sqrt{rmn}$. Then by Lemma 8 and the fact that $\text{rank}(Z) \leq \text{rank}(X) + \text{rank}(M) \leq 2r$, we have

$$|a\|Z_\Omega\|_2 - \|Z\|_2| \leq \|aZ_\Omega - Z\|_2 \leq b\|Z\|_\infty. \tag{58}$$

Using $\|Z\|_\infty = \|X - M\|_\infty \leq \|X\|_\infty + \|M\|_\infty \leq 2\alpha$, (58) can be expressed as $\|Z\|_2 \leq a\|Z_\Omega\|_2 + 2\alpha b$. Since $\|A\|_2 \leq \|A\|_F \; \forall A$, we then have $\|Z\|_2 \leq a\|Z_\Omega\|_F + 2\alpha b$. Since $\|A\|_F \leq \sqrt{\text{rank}(A)}\|A\|_2 \; \forall A$, we have $\|Z\|_F \leq \sqrt{2r}\|Z\|_2 \leq (\sqrt{2r}a)\|Z_\Omega\|_F + \sqrt{2r}2\alpha b$, leading to the desired result. ■

We now turn to the proof of Theorem 1.

Proof of Theorem 1 The bound 2α follows from the fact that $\widehat{M}, M \in \mathcal{C}$. To establish bound U_1, we will use Lemma 5 and to establish U_2, we will use Lemma 6. We first prove U_1. Consider $\tilde{F}_{\Omega,Y}(\theta) = F_{\Omega,Y}(X)$. The objective function $F_{\Omega,Y}(X)$ is continuous in X and the set $\mathcal{C}$ is compact, therefore, $F_{\Omega,Y}(X)$ achieves a minimum in $\mathcal{C}$. Now suppose that $\widehat{M} \in \mathcal{C}$ minimizes $F_{\Omega,Y}(X)$. Then $F_{\Omega,Y}(\widehat{M}) \leq F_{\Omega,Y}(X) \; \forall X \in \mathcal{C}$, including $X = M$. Define

$$c_g = 2.01 L_\alpha \sqrt{2rm}, \quad c_h = \frac{\sigma_1^2(G)\gamma_\alpha}{4rmn}, \quad \bar{c}_h = \frac{\gamma_\alpha}{2}. \tag{59}$$

Using (33) and Lemmas 5 and 7, we have w.h.p. (specified in Lemma 5)

$$F_{\Omega,Y}(\widehat{M}) \geq F_{\Omega,Y}(M) - c_g\|\widehat{M} - M\|_F + \bar{c}_h \left\| \left(\widehat{M} - M \right)_\Omega \right\|_F^2 \,. \tag{60}$$

Since $\widehat{M}$ minimizes $F_{\Omega,Y}(X)$, we have

$$0 \geq F_{\Omega,Y}(\widehat{M}) - F_{\Omega,Y}(M)$$
$$\geq -c_g\|\widehat{M} - M\|_F + \bar{c}_h \left\| \left(\widehat{M} - M \right)_\Omega \right\|_F^2 \,. \tag{61}$$

Set

$$\eta = 2\alpha r(\sigma_2(G)/\sigma_1(G))\sqrt{2mn} \quad \text{and} \quad a_0 = \sigma_1(G)/\sqrt{2rmn} \,.$$

Then Lemma 9 implies $\|(X - M)_\Omega\|_F \geq a_0 [\|X - M\|_F - \eta]$. Now consider two cases: (i) $\|\widehat{M} - M\|_F < 2\eta$, (ii) $\|\widehat{M} - M\|_F \geq 2\eta$. In case (i), we clearly have an obvious upper bound on $\|\widehat{M} - M\|_F$. Turning to case (ii), we have

$$\|\widehat{M} - M\|_F - \eta \geq \|\widehat{M} - M\|_F - \frac{1}{2}\|\widehat{M} - M\|_F$$
$$= \frac{1}{2}\|\widehat{M} - M\|_F. \tag{62}$$

Using (61), (62) and Lemma 9 with $X = \widehat{M}$, we have

$$0 \geq F_{\Omega,Y}(\widehat{M}) - F_{\Omega,Y}(M)$$
$$\geq -c_g\|\widehat{M} - M\|_F + \frac{c_h}{4}\|\widehat{M} - M\|_F^2$$
$$= \|\widehat{M} - M\|_F \left[-c_g + \frac{c_h}{4}\|\widehat{M} - M\|_F \right]. \tag{63}$$

In order for (63) to be true, we must have $\|\widehat{M} - M\|_F \leq 4c_g/c_h$ otherwise the right-side of (63) is positive violating (63). Combining the two cases, we obtain

$$\begin{aligned}\|\widehat{M} - M\|_F &\leq \max\left(2\eta, \frac{4c_g}{c_h}\right) \\ &= \max\left(4\alpha r\sqrt{2mn}\frac{\sigma_2(G)}{\sigma_1(G)}, \frac{32.16\sqrt{2}L_\alpha(rm)^{1.5}n}{\gamma_\alpha\sigma_1^2(G)}\right). \end{aligned} \tag{64}$$

This is the bound U_1 stated in (13)-14) of the theorem after division by $\sqrt{mn}$. The high probability stated in the theorem follows from Lemma 5 after setting $\varepsilon = 0.5$. Finally, we use $(\sigma_2(G)/\sigma_1(G)) \leq (C/\sqrt{d}) = C\sqrt{m}/\sqrt{|\Omega|}$ and $(1/\sigma_1^2(G)) \leq (1/d^2) = m^2/|\Omega|^2$ to derive (14).

Finally we turn to proving U_2. Define

$$\bar{c}_g = 4(1+\alpha)L_\alpha\sqrt{|\Omega|r(m+n+1)\log(9\alpha\sqrt{mn})}. \tag{65}$$

Using (33) and Lemmas 6 and 7, we have w.h.p. (specified in Lemma 6)

$$F_{\Omega,Y}(\widehat{M}) \geq F_{\Omega,Y}(M) - \bar{c}_g + \bar{c}_h\left\|\left(\widehat{M} - M\right)_\Omega\right\|_F^2. \tag{66}$$

Arguing as earlier, we then have

$$\left\|\left(\widehat{M} - M\right)_\Omega\right\|_F \leq \sqrt{\frac{2\bar{c}_g}{\gamma_\alpha}}. \tag{67}$$

As before, we have either $\|\widehat{M} - M\|_F < 2\eta$ or $\|\widehat{M} - M\|_F \geq 2\eta$; the former yields an obvious upper bound while the latter case yields

$$\left\|\widehat{M} - M\right\|_F \leq \frac{2}{a_0}\left\|\left(\widehat{M} - M\right)_\Omega\right\|_F \leq \frac{2}{a_0}\sqrt{\frac{2\bar{c}_g}{\gamma_\alpha}}. \tag{68}$$

The stated bound U_2 in (15)-16) then follows just as U_1. This completes the proof. ■

References

M.Y. An. Log-concave probability distributions: Theory and statistical testing. Working Paper No. 95-03, Department of Economics, Duke University, Durham, North Carolina, 1995.

F. Bach, J. Mairal, and J. Ponce. Convex sparse matrix factorizations. *arXiv preprint arXiv:0812.1869v1*, 2008.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 2nd edition, 1999.

S.A. Bhaskar. Quantized matrix completion for low rank matrices. In *Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3741–3745, Brisbane, Queensland, Australia, 2015.

S. Bhojanapalli and P. Jain. Universal matrix completion. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

M. Bolla, K. Friedl, and A. Kramli. Singular value decomposition of large random matrices (for two-way classification of microarrays). *Journal Multivariate Analysis*, 101:434–446, 2010.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge Univ Press, 2004.

S. Burer and R.D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming (series B)*, 95:329–357, 2003.

J.F. Cai, E.J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. Optimization*, 20(4):1956–1982, 2010.

T. Cai and W.-X. Zhou. A Max-Norm Constrained Minimization Approach to 1-Bit Matrix Completion. *Journal of Machine Learning Research*, 14:3619–3647, 2013.

E.J. Candes and Y. Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Transactions on Information Theory*, 57:2342–2359, 2011.

Y. Cao and Y. Xie. Categorical matrix completion. *arXiv preprint arXiv:1507.00421v1*, 2015.

S. Chatterjee. Matrix estimation by universal singular value thresholding. *arXiv preprint arXiv:1212.1247v5*, 2013.

M.A. Davenport, Y. Plan, E. van den Berg, and M. Wootters. 1-bit matrix completion. *Information and Inference*, 3:189–223, 2014.

U. Feige and E. Ofek. Spectral techniques applied to sparse random graphs. *Random Structures & Algorithms*, 27:251–275, 2005.

Y. Ghanbari, A.R. Smith, R.T. Schultz, and R. Verma. Connectivity subnetwork learning for pathology and developmental variations. In K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab, editors, *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*, pages 90–97. Springer, 2013.

D.F. Gleich and L.-H. Lim. Rank aggregation via nuclear norm minimization. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 60–68, 2011.

P. Gopalan, F.J.R. Ruiz, R. Ranganath, and D.M. Blei. Bayesian nonparametric poisson factorization for recommendation systems. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, 2014.

S. Gunasekar, P. Ravikumar, and J. Ghosh. Exponential family matrix completion under structural constraints. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1917–1925, 2014.

A. Gunawardana and G. Shani. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, 10:2935–2962, 2009.

S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of American Mathematical Society*, 43(4):439–561, 2006.

A. Karbasi and S. Oh. Robust localization from incomplete local information. *IEEE/ACM Transactions on Networking*, 21:1131–1144, August 2013.

R.H. Keshavan, A. Montanari, and S. Oh. Low-rank matrix completion with noisy observations: a quantitative comparison. In *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing*, Urbana, Illinois, 2009.

R.H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, 2010.

O. Klopp. Noisy low-rank matrix completion with general sampling distribution. *Bernoulli*, 20(1):282–303, 2014.

Y. Koren and J. Sill. OrdRec: An ordinal method for predicting personalized item rating distributions. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, pages 117–124, Chicago, Illinois, 2011.

Y. Koren and J. Sill. Collaborative filtering on ordinal user feedback. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 3022–3026, Beijing, China, 2013.

Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

J. Lafond. Low rank matrix completion with exponential family noise. *arXiv preprint arXiv:1502.06919v2*, 2015.

J. Lafond, O. Klopp, E. Moulines, and J. Salmon. Probabilistic low-rank matrix completion on finite alphabets. In *Advances in Neural Information Processing Systems*, pages 1727–1735, 2014.

A.S. Lan, C. Studer, and R.G. Baraniuk. Matrix recovery from quantized and corrupted measurements. In *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Florence, Italy, 2014a.

A.S. Lan, C. Studer, and R.G. Baraniuk. Quantized matrix completion for personalized learning. In *Proceedings of the 7th International Conference on Educational Data Mining*, London, UK, 2014b.

A.S. Lan, A.E. Waters, C. Studer, and R.G. Baraniuk. Sparse factor analysis for learning and content analytics. *Journal of Machine Learning Research*, 15:1959–2008, 2014c.

J.D. Lee, B. Recht, R. Salakhutdinov, N. Srebro, and J.A. Tropp. Practical large-scale optimization for max-norm regularization. In *Advances in Neural Information Processing Systems*, pages 1297–1305, 2010.

P. McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society, Series B (Methodological)*, 42(2):109–142, 1980.

R.R. Nadakuditi and M.E.J. Newman. Graph spectra and the detectability of community structure in networks. *Physical Review Letters*, 108(18):188701–5, 2012.

S. Negahban and M.J. Wainright. Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. *Journal of Machine Learning Research*, 13:1665–1697, 2012.

B. Recht and C. Re. Parallel stochastic gradient algorithms for large-scale matrix completion. *Math. Program. Comput.*, 5(2):201–226, 2013.

J.D.M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 713–719, 2005.

R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, 2008.

L.K. Saul and S.T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz. Localization from connectivity in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(11):961–974, 2004.

A. Soni, S. Jain, J. Haupt, and S. Gonella. Noisy matrix completion under sparse factor models. *arXiv preprint arXiv:1411.0282v1*, 2014.

J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

T. Tran, D. Phung, and S. Venkatesh. Cumulative restricted Boltzmann machines for ordinal matrix data analysis. In *Proceedings of the 4th Asian Conference on Machine Learning*, volume 25 of *JMLR Workshop and Conference Proceedings*, pages 411–426, 2012.

Y.-X. Wang and H. Xu. Stability of matrix factorization in collaborative filtering. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

Y. Xu and W. Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM J. Imaging Sciences*, 6(3):1758–1789, 2013.

Journal of Machine Learning Research 17 (2016) 1-35 Submitted 6/15; Revised 2/16; Published 3/16

DSA: Decentralized Double Stochastic Averaging Gradient Algorithm

Aryan Mokhtari ARYANM@SEAS.UPENN.EDU
Alejandro Ribeiro ARIBEIRO@SEAS.UPENN.EDU
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104, USA

Editor: Mark Schmidt

Abstract

This paper considers optimization problems where nodes of a network have access to summands of a global objective. Each of these local objectives is further assumed to be an average of a finite set of functions. The motivation for this setup is to solve large scale machine learning problems where elements of the training set are distributed to multiple computational elements. The decentralized double stochastic averaging gradient (DSA) algorithm is proposed as a solution alternative that relies on: (i) The use of local stochastic averaging gradients. (ii) Determination of descent steps as differences of consecutive stochastic averaging gradients. Strong convexity of local functions and Lipschitz continuity of local gradients is shown to guarantee linear convergence of the sequence generated by DSA in expectation. Local iterates are further shown to approach the optimal argument for almost all realizations. The expected linear convergence of DSA is in contrast to the sublinear rate characteristic of existing methods for decentralized stochastic optimization. Numerical experiments on a logistic regression problem illustrate reductions in convergence time and number of feature vectors processed until convergence relative to these other alternatives.

Keywords: decentralized optimization, stochastic optimization, stochastic averaging gradient, linear convergence, large-scale optimization, logistic regression

1. Introduction

We consider machine learning problems with large training sets that are distributed into a network of computing agents so that each of the nodes maintains a moderate number of samples. This leads to decentralized consensus optimization problems where summands of the global objective function are available at different nodes of the network. In this class of problems agents (nodes) try to optimize the global cost function by operating on their local functions and communicating with their neighbors only. Specifically, consider a variable $\mathbf{x} \in \mathbb{R}^p$ and a connected network of size N where each node n has access to a local objective function $f_n : \mathbb{R}^p \to \mathbb{R}$. The local objective function $f_n(\mathbf{x})$ is defined as the average of q_n local instantaneous functions $f_{n,i}(\mathbf{x})$ that can be individually evaluated at node n. Agents cooperate to solve the global optimization problem

$$\tilde{\mathbf{x}}^* := \operatorname*{argmin}_{\mathbf{x}} \sum_{n=1}^{N} f_n(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{x}} \sum_{n=1}^{N} \frac{1}{q_n} \sum_{i=1}^{q_n} f_{n,i}(\mathbf{x}). \tag{1}$$

The formulation in (1) models a training set with a total of $\sum_{n=1}^{N} q_n$ training samples that are distributed among the N agents for parallel processing conducive to the determination of the optimal classifier $\tilde{\mathbf{x}}^*$ (Bekkerman et al. (2011); Tsianos et al. (2012a); Cevher et al. (2014)). Although we make no formal assumption, in cases of practical importance the total number of training samples $\sum_{n=1}^{N} q_n$ is very large, but the number of elements q_n available at a specific node is moderate.

Analogous formulations are also of interest in decentralized control systems (Bullo et al. (2009); Cao et al. (2013); Lopes and Sayed (2008)), wireless systems (Ribeiro (2010, 2012)), and sensor networks (Schizas et al. (2008); Khan et al. (2010); Rabbat and Nowak (2004)). Our interest here is in solving (1) with a method that has the following three properties

- Decentralized; nodes operate on their local functions and communicate with neighbors only.
- Stochastic; nodes determine a descent direction by evaluating only one out of the q_n functions $f_{n,i}$ at each iteration.
- Linear convergence rate; the expected distance to the optimum is scaled by a subunit factor at each iteration.

Decentralized optimization is relatively mature and various methods are known with complementary advantages. These methods include decentralized gradient descent (DGD) (Nedić and Ozdaglar (2009); Jakovetic et al. (2014); Yuan et al. (2013)), network Newton (Mokhtari et al. (2015a,b)), decentralized dual averaging (Duchi et al. (2012); Tsianos et al. (2012b)), the exact first order algorithm (EXTRA) (Shi et al. (2015)), as well as the alternating direction method of multipliers (ADMM) (Boyd et al. (2011); Schizas et al. (2008); Shi et al. (2014); Iutzeler et al. (2013)) and its linearized variants (Ling and Ribeiro (2014); Ling et al. (2015); Mokhtari et al. (2015c)). The ADMM, its variants, and EXTRA converge linearly to the optimal argument but DGD, network Newton, and decentralized dual averaging have sublinear convergence rates. Of particular importance to this paper, is the fact that DGD has (inexact) linear convergence to a neighborhood of the optimal argument when it uses constant stepsizes. It can achieve exact convergence by using diminishing stepsizes, but the convergence rate degrades to sublinear. This lack of linear convergence is solved by EXTRA through the use of iterations that rely on information of two consecutive steps (Shi et al. (2015)).

All of the algorithms mentioned above require the computationally costly evaluation of the local gradients $\nabla f_n(\mathbf{x}) = (1/q_n) \sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{x})$. This cost can be avoided by stochastic decentralized algorithms that reduce computational cost of iterations by substituting all local gradients with their stochastic approximations. This reduces the computational cost per iteration but results in sublinear convergence rates of order $O(1/t)$ even if the corresponding deterministic algorithm exhibits linear convergence. This is a drawback that also exists in centralized stochastic optimization where linear convergence rates in expectation are established by decreasing the variance of the stochastic gradient approximation (Roux et al. (2012); Schmidt et al. (2013); Shalev-Shwartz and Zhang (2013); Johnson and Zhang (2013); Konečnỳ and Richtárik (2013); Defazio et al. (2014)). In this paper we build on the ideas of the stochastic averaging gradient (SAG) algorithm (Schmidt et al. (2013)) and its unbiased version SAGA (Defazio et al. (2014)). Both of these algorithms use the idea of stochastic incremental averaging gradients. At each iteration only one of the stochastic gradients is updated and the average of all of the most recent stochastic gradients is used for estimating the gradient.

The contribution of this paper is to develop the decentralized double stochastic averaging gradient (DSA) method, a novel decentralized stochastic algorithm for solving (1). The method exploits a new interpretation of EXTRA as a saddle point method and uses stochastic averaging gradients in lieu of gradients. DSA is *decentralized* because it is implementable in a network setting where nodes can communicate only with their neighbors. It is *double* because iterations utilize the information of two consecutive iterates. It is *stochastic* because the gradient of only one randomly selected function is evaluated at each iteration and it is an *averaging* method because it uses an average of stochastic gradients to approximate the local gradients. DSA is proven to converge linearly to the optimal argument $\tilde{\mathbf{x}}^*$ in expectation when the local instantaneous functions $f_{n,i}$ are strongly convex, with Lipschitz continuous gradients. This is in contrast to all other decentralized stochastic methods to solve (1) that converge at sublinear rates.

We begin the paper with a discussion of DGD, EXTRA and stochastic averaging gradient. With these definitions in place we define the DSA algorithm by replacing the gradients used in EXTRA

by stochastic averaging gradients (Section 2). We follow with a digression on the limit points of DGD and EXTRA iterations to explain the reason why DGD does not achieve exact convergence but EXTRA is expected to do so (Section 2.1). A reinterpretation of EXTRA as a saddle point method that solves for the critical points of the augmented Lagrangian of a constrained optimization problem equivalent to (1) is then introduced. It follows from this reinterpretation that DSA is a stochastic saddle point method (Section 2.2). The fact that DSA is a stochastic saddle point method is the critical enabler of the subsequent convergence analysis (Section 3). In particular, it is possible to guarantee that strong convexity and gradient Lipschitz continuity of the local instantaneous functions $f_{n,i}$ imply that a Lyapunov function associated with the sequence of iterates generated by DSA converges linearly to its optimal value in expectation (Theorem 7). Linear convergence in expectation of the local iterates to the optimal argument $\tilde{\mathbf{x}}^*$ of (1) follows as a trivial consequence (Corollary 8). We complement this result by showing convergence of all the local variables to the optimal argument $\tilde{\mathbf{x}}^*$ with probability 1 (Theorem 9).

The advantages of DSA relative to a group of stochastic and deterministic alternatives in solving a logistic regression problem are then studied in numerical experiments (Section 4). These results demonstrate that DSA is the only decentralized stochastic algorithm that reaches the optimal solution with a linear convergence rate. We further show that DSA outperforms deterministic algorithms when the metric is the number of times that elements of the training set are evaluated. The behavior of DSA for different network topologies is also evaluated. We close the paper with pertinent remarks (Section 5).

Notation Lowercase boldface $\mathbf{v}$ denotes a vector and uppercase boldface $\mathbf{A}$ a matrix. For column vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ we use the notation $\mathbf{x} = [\mathbf{x}_1; \ldots; \mathbf{x}_N]$ to represent the stack column vector $\mathbf{x}$. We use $\|\mathbf{v}\|$ to denote the Euclidean norm of vector $\mathbf{v}$ and $\|\mathbf{A}\|$ to denote the Euclidean norm of matrix $\mathbf{A}$. For a vector $\mathbf{v}$ and a positive definite matrix $\mathbf{A}$, the $\mathbf{A}$-weighted norm is defined as $\|\mathbf{v}\|_{\mathbf{A}} := \sqrt{\mathbf{v}^T\mathbf{A}\mathbf{v}}$. The null space of matrix $\mathbf{A}$ is denoted by null($\mathbf{A}$) and the span of a vector by span($\mathbf{x}$). The operator $\mathbb{E}_{\mathbf{x}}[\cdot]$ stands for expectation over random variable $\mathbf{x}$ and $\mathbb{E}[\cdot]$ for expectation with respect to the distribution of a stochastic process.

2. Decentralized Double Stochastic Averaging Gradient

Consider a connected network that contains N nodes such that each node n can only communicate with peers in its neighborhood $\mathcal{N}_n$. Define $\mathbf{x}_n \in \mathbb{R}^p$ as a local copy of the variable $\mathbf{x}$ that is kept at node n. In decentralized optimization, agents try to minimize their local functions $f_n(\mathbf{x}_n)$ while ensuring that their local variables $\mathbf{x}_n$ coincide with the variables $\mathbf{x}_m$ of all neighbors $m \in \mathcal{N}_n$ – which, given that the network is connected, ensures that the variables $\mathbf{x}_n$ of all nodes are the same and renders the problem equivalent to (1). DGD is a well known method for decentralized optimization that relies on the introduction of nonnegative weights $w_{ij} \geq 0$ that are not null if and only if $m = n$ or if $m \in \mathcal{N}_n$. Letting $t \in \mathbb{N}$ be a discrete time index and α a given stepsize, DGD is defined by the recursion

$$\mathbf{x}_n^{t+1} = \sum_{m=1}^{N} w_{nm}\mathbf{x}_m^t - \alpha\nabla f_n(\mathbf{x}_n^t), \qquad n = 1, \ldots, N. \tag{2}$$

Since $w_{nm} = 0$ when $m \neq n$ and $m \notin \mathcal{N}_n$, it follows from (2) that node n updates $\mathbf{x}_n$ by performing an average over the variables $\mathbf{x}_m^t$ of its neighbors $m \in \mathcal{N}_n$ and its own $\mathbf{x}_n^t$, followed by descent through the negative local gradient $-\nabla f_n(\mathbf{x}_n^t)$. If a constant stepsize is used, DGD iterates $\mathbf{x}_n^t$ approach a neighborhood of the optimal argument $\tilde{\mathbf{x}}^*$ of (1) but don't converge exactly. To achieve exact convergence diminishing stepsizes are used but the resulting convergence rate is sublinear (Nedić and Ozdaglar (2009)).

EXTRA is a method that resolves either of these issues by mixing two consecutive DGD iterations with different weight matrices and opposite signs. To be precise, introduce a second set of weights

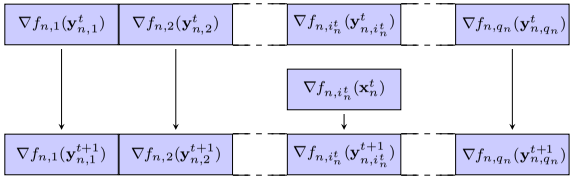


Figure 1: Stochastic averaging gradient table at node n. At each iteration t a random local instantaneous gradient $\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)$ is updated by $\nabla f_{n,i_n^t}(\mathbf{x}_n^t)$. The rest of the local instantaneous gradients remain unchanged, i.e., $\nabla f_{n,i}(\mathbf{y}_{n,i}^{t+1}) = \nabla f_{n,i}(\mathbf{y}_{n,i}^t)$ for $i \neq i_n^t$. This list is used to compute the stochastic averaging gradient in (7).

$\tilde{w}_{nm}$ with the same properties as the weights w_{nm} and define EXTRA through the recursion

$$\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \sum_{m=1}^{N} w_{nm}\mathbf{x}_m^t - \sum_{m=1}^{N} \tilde{w}_{nm}\mathbf{x}_m^{t-1} - \alpha\left[\nabla f_n(\mathbf{x}_n^t) - \nabla f_n(\mathbf{x}_n^{t-1})\right], \quad n = 1, \ldots, N. \tag{3}$$

Observe that (3) is well defined for $t > 0$. For $t = 0$ we utilize the regular DGD iteration in (2). In the nomenclature of this paper we say that EXTRA performs a decentralized double gradient descent step because it operates in a decentralized manner while utilizing a difference of two gradients as descent direction. Minor modification as it is, the use of this gradient difference in lieu of simple gradients, endows EXTRA with exact linear convergence to the optimal argument $\tilde{\mathbf{x}}^*$ under mild assumptions (Shi et al. (2015)).

If we recall the definitions of the local functions $f_n(\mathbf{x}_n)$ and the instantaneous local functions $f_{n,i}(\mathbf{x}_n)$ available at node n, the implementation of EXTRA requires that each node n computes the full gradient of its local objective function f_n at $\mathbf{x}_n^t$ as

$$\nabla f_n(\mathbf{x}_n^t) = \frac{1}{q_n}\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{x}_n^t). \tag{4}$$

This is computationally expensive when the number of instantaneous functions q_n is large. To resolve this issue, local stochastic gradients can be substituted for the local objective functions gradients in (3). These stochastic gradients approximate the gradient $\nabla f_n(\mathbf{x}_n)$ of node n by randomly choosing one of the instantaneous functions gradients $\nabla f_{n,i}(\mathbf{x}_n)$. If we let $i_n^t \in \{1, \ldots q_n\}$ denote a function index that we choose at time t at node n uniformly at random and independently of the history of the process, then the stochastic gradient is defined as

$$\hat{\mathbf{s}}_n(\mathbf{x}_n^t) := \nabla f_{n,i_n^t}(\mathbf{x}_n^t). \tag{5}$$

We can then write a stochastic version of EXTRA by replacing $\nabla f_n(\mathbf{x}_n^t)$ by $\hat{\mathbf{s}}_n(\mathbf{x}_n^t)$ and $\nabla f_n(\mathbf{x}_n^{t-1})$ by $\hat{\mathbf{s}}_n(\mathbf{x}_n^{t-1})$. Such an algorithm would have a small computational cost per iteration. On the negative side, it either has a linear convergence to a neighborhood of the optimal solution $\mathbf{x}^*$ with constant stepsize α, or it would converge sublinearly to the optimal argument when the stepsize diminishes as time passes. Here however, we want to design an algorithm with low computational complexity that converges linearly to the exact solution $\mathbf{x}^*$.

To reduce this noise we propose the use of stochastic averaging gradients instead (Defazio et al. (2014)). The idea is to maintain a list of gradients of all instantaneous functions in which one randomly chosen element is replaced at each iteration and to use an average of the elements of this

list for gradient approximation; see Figure 1. Formally, define the variable $\mathbf{y}_{n,i} \in \mathbb{R}^p$ to represent the iterate value the last time that the instantaneous gradient of function $f_{n,i}$ was evaluated. If we let $i_n^t \in \{1,\dots,q_n\}$ denote the function index chosen at time t at node n, as we did in (5), the variables $\mathbf{y}_{n,i}$ are updated recursively as

$$\mathbf{y}_{n,i}^{t+1} = \mathbf{x}_n^t, \quad \text{if } i = i_n^t, \qquad \mathbf{y}_{n,i}^{t+1} = \mathbf{y}_{n,i}^t, \quad \text{if } i \neq i_n^t. \tag{6}$$

With these definitions in hand we can define the stochastic averaging gradient at node n as

$$\hat{\mathbf{g}}_n^t := \nabla f_{n,i_n^t}(\mathbf{x}_n^t) - \nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t) + \frac{1}{q_n}\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{y}_{n,i}^t). \tag{7}$$

Observe that to implement (7) the gradients $\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$ are stored in the local gradient table shown in Figure 1.

The DSA algorithm is a variation of EXTRA that substitutes the local gradients $\nabla f_n(\mathbf{x}_n^t)$ in (3) for the local stochastic average gradients $\hat{\mathbf{g}}_n^t$ in (7),

$$\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \sum_{m=1}^N w_{nm}\mathbf{x}_m^t - \sum_{m=1}^N \tilde{w}_{nm}\mathbf{x}_m^{t-1} - \alpha\left[\hat{\mathbf{g}}_n^t - \hat{\mathbf{g}}_n^{t-1}\right]. \tag{8}$$

The DSA initial update is given by applying the same substitution for the update of DGD in (2) as

$$\mathbf{x}_n^1 = \sum_{m=1}^N w_{nm}\mathbf{x}_m^0 - \alpha\,\hat{\mathbf{g}}_n^0. \tag{9}$$

DSA is summarized in Algorithm 1 for $t \geq 0$. The DSA update in (8) is implemented in Step 9. This step requires access to the local iterates $\mathbf{x}_m^t$ of neighboring nodes $m \in \mathcal{N}_n$ which are collected in Step 2. Furthermore, implementation of the DSA update also requires access to the stochastic averaging gradients $\hat{\mathbf{g}}_n^{t-1}$ and $\hat{\mathbf{g}}_n^t$. The latter is computed in Step 4 and the former is computed and stored at the same step in the previous iteration. The computation of the stochastic averaging gradients requires the selection of the index i_n^t. This index is chosen uniformly at random in Step 3. Determination of stochastic averaging gradients also necessitates access and maintenance of the gradients table in Figure 1. The i_n^t element of this table is updated in Step 5 by replacing $\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)$ with $\nabla f_{n,i_n^t}(\mathbf{x}_n^t)$, while the other vectors remain unchanged. To implement the first DSA iteration at time $t = 0$ we have to perform the update in (9) instead of the update in (8) as in Step 7. Further observe that the auxiliary variables $\mathbf{y}_{n,i}^0$ are initialized to the initial iterate $\mathbf{x}_n^0$. This implies that the initial values of the stored gradients are $\nabla f_{n,i}(\mathbf{y}_{n,i}^0) = \nabla f_{n,i}(\mathbf{x}_n^0)$.

We point out that the weights w_{nm} and $\tilde{w}_{nm}$ can't be arbitrary. If we define weight matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ with elements w_{nm} and $\tilde{w}_{nm}$, respectively, they have to satisfy conditions that we state as an assumption for future reference.

Assumption 1 *The weight matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ must satisfy the following properties*

(a) Both are symmetric, $\mathbf{W} = \mathbf{W}^T$ and $\tilde{\mathbf{W}} = \tilde{\mathbf{W}}^T$.

(b) The null space of $\mathbf{I} - \tilde{\mathbf{W}}$ includes the span of $\mathbf{1}$, i.e., null$(\mathbf{I} - \tilde{\mathbf{W}}) \supseteq$ span$(\mathbf{1})$, the null space of $\mathbf{I} - \mathbf{W}$ is the span of $\mathbf{1}$, i.e., null$(\mathbf{I} - \mathbf{W}) =$ span$(\mathbf{1})$, and the null space of the difference $\tilde{\mathbf{W}} - \mathbf{W}$ is the span of $\mathbf{1}$, i.e., null$(\tilde{\mathbf{W}} - \mathbf{W}) =$ span$(\mathbf{1})$.

(c) They satisfy the spectral ordering $\mathbf{W} \preceq \tilde{\mathbf{W}} \preceq (\mathbf{I} + \mathbf{W})/2$ and $\mathbf{0} \prec \tilde{\mathbf{W}}$.

Algorithm 1 DSA algorithm at node n

Require: Vectors $\mathbf{x}_n^0$. Gradient table initialized with instantaneous gradients $\nabla f_{n,i}(\mathbf{y}_{n,i}^0)$ with $\mathbf{y}_{n,i}^0 = \mathbf{x}_n^0$.

1: **for** $t = 0, 1, 2, \dots$ **do**
2: Exchange variable $\mathbf{x}_n^t$ with neighboring nodes $m \in \mathcal{N}_n$;
3: Choose i_n^t uniformly at random from the set $\{1,\dots,q_n\}$;
4: Compute and store stochastic averaging gradient as per (7):
$\hat{\mathbf{g}}_n^t = \nabla f_{n,i_n^t}(\mathbf{x}_n^t) - \nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t) + \frac{1}{q_n}\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$;
5: Take $\mathbf{y}_{n,i_n^t}^{t+1} = \mathbf{x}_n^t$ and store $\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^{t+1}) = \nabla f_{n,i_n^t}(\mathbf{x}_n^t)$ in i_n^t gradient table position. All other entries in the table remain unchanged. The vector $\mathbf{y}_{n,i_n^t}^{t+1}$ is not explicitly stored;
6: **if** $t = 0$ **then**
7: Update variable $\mathbf{x}_n^t$ as per (9): $\mathbf{x}_n^{t+1} = \sum_{m=1}^N w_{nm}\mathbf{x}_n^t - \alpha\hat{\mathbf{g}}_n^t$;
8: **else**
9: Update variable $\mathbf{x}_n^t$ as per (8): $\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \sum_{m=1}^N w_{nm}\mathbf{x}_n^t - \sum_{m=1}^N \tilde{w}_{nm}\mathbf{x}_n^{t-1} - \alpha\left[\hat{\mathbf{g}}_n^t - \hat{\mathbf{g}}_n^{t-1}\right]$;
10: **end if**
11: **end for**

Requiring the matrix $\mathbf{W}$ to be symmetric and with specific null space properties is necessary to let all agents converge to the same optimal variable. Analogous properties are necessary in DGD and are not difficult to satisfy. The condition on spectral ordering is specific to EXTRA but is not difficult to satisfy either. E.g., if we have a matrix $\mathbf{W}$ that satisfies all the conditions in Assumption 1, the weight matrix $\tilde{\mathbf{W}} = (\mathbf{I} + \mathbf{W})/2$ makes Assumption 1 valid.

We also point that, as written in (7), computation of local stochastic averaging gradients $\hat{\mathbf{g}}_n^t$ is costly because it requires evaluation of the sum $\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$ at each iteration. To be more precise, if we implement the update in (7) naively, at each iteration we should compute the sum $\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$ which has a computational cost of the order $O(q_n)$. This cost can be avoided by updating the sum at each iteration with the recursive formula

$$\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t) \;=\; \sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^{t-1}) + \nabla f_{n,i_n^{t-1}}(\mathbf{x}_n^{t-1}) - \nabla f_{n,i_n^{t-1}}(\mathbf{y}_{n,i_n^{t-1}}^{t-1}). \tag{10}$$

Using the update in (10), we can update the sum $\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$ required for (7) in a computationally efficient manner. Important properties and interpretations of EXTRA and DSA are presented in the following sections after pertinent remarks.

Remark 1 The local stochastic averaging gradients in (7) are unbiased estimates of the local gradients $\nabla f_n(\mathbf{x}_n^t)$. Indeed, if we let $\mathcal{F}_t$ measure the history of the system up until time t we have that the sum in (7) is deterministic given this sigma-algebra. This observation implies that the conditional expectation $\mathbb{E}\left[(1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)\,|\,\mathcal{F}^t\right]$ can be simplified as $(1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$. Thus, the conditional expectation of the stochastic averaging gradient is,

$$\mathbb{E}\left[\hat{\mathbf{g}}_n^t\,\middle|\,\mathcal{F}^t\right] = \mathbb{E}\Big[\nabla f_{n,i_n^t}(\mathbf{x}_n^t)\,\Big|\,\mathcal{F}^t\Big] - \mathbb{E}\Big[\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)\,\Big|\,\mathcal{F}^t\Big] + \frac{1}{q_n}\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t). \tag{11}$$

With the index i_n^t chosen equiprobably from the set $\{1,\dots,q_n\}$, the expectation of the second term in (11) is the same as the sum in the last term – each of the indexes is chosen with probability $1/q_n$. In other words, we can write $\mathbb{E}\Big[\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)\,\Big|\,\mathcal{F}^t\Big] = (1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$. Therefore,

these two terms cancel out each other and, since the expectation of the first term in (11) is simply $\mathbb{E}\big[\nabla f_{n,i_n^t}(\mathbf{x}_n^t)\,\big|\,\mathcal{F}^t\big] = (1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{x}_n^t) = \nabla f_n(\mathbf{x}_n^t)$, we can simplify (11) to

$$\mathbb{E}\left[\hat{\mathbf{g}}_n^t\,\big|\,\mathcal{F}^t\right] = \nabla f_n(\mathbf{x}_n^t). \tag{12}$$

The expression in (12) means, by definition, that $\hat{\mathbf{g}}_n^t$ is an unbiased estimate of $\nabla f_n(\mathbf{x}_n^t)$ when the history $\mathcal{F}^t$ is given.

Remark 2 The local stochastic averaging gradient $\hat{\mathbf{g}}_n^t$ at node n contains three terms. The first two terms $\nabla f_{n,i_n^t}(\mathbf{x}_n^t)$ and $\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)$ are the new and old gradients of the chosen objective function f_{n,i_n^t} at node n, respectively. The last term $(1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)$ is the average of the average of all the instantaneous gradients available at node n. This update can be considered as a localized version of the stochastic averaging gradient update in the SAGA algorithm (Defazio et al. (2014)). Notice that instead of the difference $\nabla f_{n,i_n^t}(\mathbf{x}_n^t) - \nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)$ in (7) we could use the difference $(\nabla f_{n,i_n^t}(\mathbf{x}_n^t) - \nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t))/q_n$ which would lead to stochastic averaging gradient suggested in the SAG algorithm (Schmidt et al. (2013)). As studied in (Defazio et al. (2014)), both of these approximations lead to a variance reduction method. The one suggested by SAGA is an unbiased estimator of the exact gradient $(1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{x}_n^t)$, while the one suggested by SAG is a biased estimator of the gradient with smaller variance. Since the analysis of the estimator suggested by SAGA is simpler, we use its idea to define the local stochastic averaging gradient $\hat{\mathbf{g}}_n^t$ in (7).

2.1 Limit Points of DGD and EXTRA

The derivation of EXTRA hinges on the observation that the optimal argument of (1) is not a fixed point of the DGD iteration in (2) but is a fixed point of the iteration in (3). To explain this point define $\mathbf{x} := [\mathbf{x}_1;\ldots;\mathbf{x}_N] \in \mathbb{R}^{Np}$ as a vector that concatenates the local iterates $\mathbf{x}_n$ and the aggregate function $f:\mathbb{R}^{Np}\to\mathbb{R}$ as the one that takes values $f(\mathbf{x}) = f(\mathbf{x}_1,\ldots,\mathbf{x}_N) := \sum_{n=1}^N f_n(\mathbf{x}_n)$. Decentralized optimization entails the minimization of $f(\mathbf{x})$ subject to the constraint that all local variables are equal,

$$\begin{aligned}\mathbf{x}^* := \operatorname{argmin}\ & f(\mathbf{x}) = f(\mathbf{x}_1,\ldots,\mathbf{x}_N) = \sum_{n=1}^N f_n(\mathbf{x}_n),\\ \text{s.t.}\quad & \mathbf{x}_n = \mathbf{x}_m, \quad \text{for all } n,m.\end{aligned} \tag{13}$$

The problems in (1) and (13) are equivalent in the sense that the vector $\mathbf{x}^*\in\mathbb{R}^{Np}$ is a solution of (13) if it satisfies $\mathbf{x}_n^* = \tilde{\mathbf{x}}^*$ for all n, or, equivalently, if we can write $\mathbf{x}^* = [\tilde{\mathbf{x}}^*;\ldots;\tilde{\mathbf{x}}^*]$. Regardless of interpretation, the Karush, Kuhn, Tucker (KKT) conditions of (13) dictate that that optimal argument $\mathbf{x}^*$ must sastisfy

$$\mathbf{x}^* \subset \operatorname{span}(\mathbf{1}_N\otimes\mathbf{I}_p), \qquad (\mathbf{1}_N\otimes\mathbf{I}_p)^T\nabla f(\mathbf{x}^*) = \mathbf{0}. \tag{14}$$

The first condition in (14) requires that all the local variables $\mathbf{x}_n^*$ be equal, while the second condition requires the sum of local gradients to vanish at the optimal point. This latter condition is not the same as $\nabla f(\mathbf{x}) = \mathbf{0}$. If we observe that the gradient $\nabla f(\mathbf{x}^t)$ of the aggregate function can be written as $\nabla f(\mathbf{x}) = [\nabla f_1(\mathbf{x}_1);\ldots;\nabla f_N(\mathbf{x}_N)]\in\mathbb{R}^{Np}$, the condition $\nabla f(\mathbf{x}) = \mathbf{0}$ implies that all the local gradients are null, i.e., that $\nabla f_n(\mathbf{x}_n) = \mathbf{0}$ for all n. This is stronger than having their sum being null as required by (14).

Define now the extended weight matrices as the Kronecker products $\mathbf{Z} := \mathbf{W}\otimes\mathbf{I}\in\mathbb{R}^{Np\times Np}$ and $\tilde{\mathbf{Z}} := \tilde{\mathbf{W}}\otimes\mathbf{I}\in\mathbb{R}^{Np\times Np}$. Note that the required conditions for the weight matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ in Assumption 1 enforce some conditions on the extended weight matrices $\mathbf{Z}$ and $\tilde{\mathbf{Z}}$. Based on Assumption 1(a), the matrices $\mathbf{Z}$ and $\tilde{\mathbf{Z}}$ are also symmetric, i.e., $\mathbf{Z} = \mathbf{Z}^T$ and $\tilde{\mathbf{Z}} = \tilde{\mathbf{Z}}^T$. Conditions in

Assumption 1(b) imply that $\operatorname{null}\{\tilde{\mathbf{Z}}-\mathbf{Z}\} = \operatorname{span}\{\mathbf{1}\otimes\mathbf{I}\}$, $\operatorname{null}\{\mathbf{I}-\mathbf{Z}\} = \operatorname{span}\{\mathbf{1}\otimes\mathbf{I}\}$, and $\operatorname{null}\{\mathbf{I}-\tilde{\mathbf{Z}}\}\supseteq \operatorname{span}\{\mathbf{1}\otimes\mathbf{I}\}$. Lastly, the spectral properties of matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ in Assumption 1(c) yield that matrix $\tilde{\mathbf{Z}}$ is positive definite and the expression $\mathbf{Z}\preceq\tilde{\mathbf{Z}}\preceq(\mathbf{I}+\mathbf{Z})/2$ holds.

According to the definition of the extended weight matrix $\mathbf{Z}$, the DGD iteration in (2) is equivalent to

$$\mathbf{x}^{t+1} = \mathbf{Z}\mathbf{x}^t - \alpha\nabla f(\mathbf{x}^t), \tag{15}$$

where, according to (13), the gradient $\nabla f(\mathbf{x}^t)$ of the aggregate function can be written as $\nabla f(\mathbf{x}^t) = [\nabla f_1(\mathbf{x}_1^t);\ldots;\nabla f_N(\mathbf{x}_N^t)]\in\mathbb{R}^{Np}$. Likewise, the EXTRA iteration in (3) can be written as

$$\mathbf{x}^{t+1} = (\mathbf{I}+\mathbf{Z})\mathbf{x}^t - \tilde{\mathbf{Z}}\mathbf{x}^{t-1} - \alpha\left[\nabla f(\mathbf{x}^t) - \nabla f(\mathbf{x}^{t-1})\right]. \tag{16}$$

The fundamental difference between DGD and EXTRA is that a fixed point of (15) does not necessarily satisfy (14), whereas the fixed points of (16) are guaranteed to do so. Indeed, taking limits in (15) we see that the fixed points $\mathbf{x}^\infty$ of DGD must satisfy

$$(\mathbf{I}-\mathbf{Z})\mathbf{x}^\infty + \alpha\nabla f(\mathbf{x}^\infty) = \mathbf{0}, \tag{17}$$

which is incompatible with (14) except in peculiar circumstances – such as, e.g., when all local functions have the same minimum. The limit points of EXTRA, however, satisfy the relationship

$$\mathbf{x}^\infty - \mathbf{x}^\infty = (\mathbf{Z}-\tilde{\mathbf{Z}})\mathbf{x}^\infty - \alpha[\nabla f(\mathbf{x}^\infty) - \nabla f(\mathbf{x}^\infty)]. \tag{18}$$

Canceling out the variables on the left hand side and the gradients in the right hand side it follows that $(\mathbf{Z}-\tilde{\mathbf{Z}})\mathbf{x}^\infty = \mathbf{0}$. Since the null space of of $\mathbf{Z}-\tilde{\mathbf{Z}}$ is $\operatorname{null}(\mathbf{Z}-\tilde{\mathbf{Z}}) = \mathbf{1}_N\otimes\mathbf{I}_p$ by assumption, we must have $\mathbf{x}^\infty\subset\operatorname{span}(\mathbf{1}_N\otimes\mathbf{I}_p)$. This is the first condition in (14). For the second condition in (14) sum the updates in (16) recursively and use the telescopic nature of the sum to write

$$\mathbf{x}^{t+1} = \tilde{\mathbf{Z}}\mathbf{x}^t - \alpha\nabla f(\mathbf{x}^t) - \sum_{s=0}^t(\tilde{\mathbf{Z}}-\mathbf{Z})\mathbf{x}^s. \tag{19}$$

Substituting the limit point in (19) and reordering terms, we see that $\mathbf{x}^\infty$ must satisfy

$$\alpha\nabla f(\mathbf{x}^\infty) = (\mathbf{I}-\tilde{\mathbf{Z}})\mathbf{x}^\infty - \sum_{s=0}^\infty(\tilde{\mathbf{Z}}-\mathbf{Z})\mathbf{x}^s. \tag{20}$$

In (20) we have that $(\mathbf{I}-\tilde{\mathbf{Z}})\mathbf{x}^\infty = \mathbf{0}$ because the null space of $(\mathbf{I}-\tilde{\mathbf{Z}})$ is $\operatorname{null}(\mathbf{Z}-\tilde{\mathbf{Z}}) = \mathbf{1}_N\otimes\mathbf{I}_p$ by assumption and $\mathbf{x}^\infty\subset\operatorname{span}(\mathbf{1}_N\otimes\mathbf{I}_p)$ as already shown. Implementing this simplification and considering the multiplication of the resulting equality by $(\mathbf{1}_N\otimes\mathbf{I}_p)^T$ we obtain

$$(\mathbf{1}_N\otimes\mathbf{I}_p)^T\alpha\nabla f(\mathbf{x}^\infty) = -\sum_{s=0}^\infty(\mathbf{1}_N\otimes\mathbf{I}_p)^T(\mathbf{Z}-\tilde{\mathbf{Z}})\mathbf{x}^s. \tag{21}$$

In (21), the terms $(\mathbf{1}_N\otimes\mathbf{I}_p)^T(\mathbf{Z}-\tilde{\mathbf{Z}}) = \mathbf{0}$ because the matrices $\mathbf{Z}$ and $\tilde{\mathbf{Z}}$ are symmetric and $(\mathbf{1}_N\otimes\mathbf{I}_p)$ is in the null space of the difference $\mathbf{Z}-\tilde{\mathbf{Z}}$. This implies that $(\mathbf{1}_N\otimes\mathbf{I}_p)^T\alpha\nabla f(\mathbf{x}^\infty) = \mathbf{0}$, which is the second condition in (14). Therefore, given the assumption that the sequence of EXTRA iterates $\mathbf{x}^t$ has a limit point $\mathbf{x}^\infty$ it follows that this limit point satisfies both conditions in (14) and for this reason exact convergence with constant stepsize is achievable for EXTRA.

2.2 Stochastic Saddle Point Method Interpretation of DSA

The convergence proofs of DSA build on a reinterpretation of EXTRA as a saddle point method. To introduce this primal-dual interpretation consider the update in (19) and define the sequence of

vectors $\mathbf{v}^t = \sum_{s=0}^{t}(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^s$. The vector $\mathbf{v}^t$ represents the accumulation of variable dissimilarities in different nodes over time. Considering this definition of $\mathbf{v}^t$ we can rewrite (19) as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha\left[\nabla f(\mathbf{x}^t) + \frac{1}{\alpha}(\mathbf{I}-\tilde{\mathbf{Z}})\mathbf{x}^t + \frac{1}{\alpha}(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{v}^t\right]. \tag{22}$$

Furthermore, based on the definition of the sequence $\mathbf{v}^t = \sum_{s=0}^{t}(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^s$ we can write the recursive expression

$$\mathbf{v}^{t+1} = \mathbf{v}^t + \alpha\left[\frac{1}{\alpha}(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^{t+1}\right]. \tag{23}$$

Consider $\mathbf{x}$ as a primal variable and $\mathbf{v}$ as a dual variable. Then, the updates in (22) and (23) are equivalent to the updates of a saddle point method with stepsize α that solves for the critical points of the augmented Lagrangian

$$\mathcal{L}(\mathbf{x},\mathbf{v}) = f(\mathbf{x}) + \frac{1}{\alpha}\mathbf{v}^T(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x} + \frac{1}{2\alpha}\mathbf{x}^T(\mathbf{I}-\tilde{\mathbf{Z}})\mathbf{x}. \tag{24}$$

In the Lagrangian in (24) the factor $(1/\alpha)\mathbf{v}^T(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}$ stems from the linear constraint $(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}=\mathbf{0}$ and the quadratic term $(1/2\alpha)\mathbf{x}^T(\mathbf{I}-\tilde{\mathbf{Z}})\mathbf{x}$ is the augmented term added to the Lagrangian. Therefore, the optimization problem whose augmented Lagrangian is the one given in (24) is

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \; f(\mathbf{x}) \qquad \text{s.t. } \frac{1}{\alpha}(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x} = \mathbf{0}. \tag{25}$$

Observing that the null space of $(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}$ is $\text{null}((\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}) = \text{null}(\tilde{\mathbf{Z}}-\mathbf{Z}) = \text{span}\{\mathbf{1}_N \otimes \mathbf{I}_p\}$, the constraint in (25) is equivalent to the consensus constraint $\mathbf{x}_n = \mathbf{x}_m$ for all n,m that appears in (13). This means that (25) is equivalent to (13), which, as already argued, is equivalent to the original problem in (1). Hence, EXTRA is a saddle point method that solves (25) which, because of their equivalence, is tantamount to solving (1). Considering that saddle point methods converge linearly, it follows that the same is true of EXTRA.

That EXTRA is a saddle point method provides a simple explanation of its convergence properties. For the purposes of this paper, however, the important fact is that if EXTRA is a saddle point method, DSA is a stochastic saddle point method. To write DSA in this form define $\hat{\mathbf{g}}^t := [\hat{\mathbf{g}}_1^t;\ldots;\hat{\mathbf{g}}_N^t] \in \mathbb{R}^{Np}$ as the vector that concatenates all the local stochastic averaging gradients at step t. Then, the DSA update in (8) can be written as

$$\mathbf{x}^{t+1} = (\mathbf{I}+\mathbf{Z})\mathbf{x}^t - \tilde{\mathbf{Z}}\mathbf{x}^{t-1} - \alpha\left[\hat{\mathbf{g}}^t - \hat{\mathbf{g}}^{t-1}\right]. \tag{26}$$

Comparing (16) and (26) we see that they differ in the latter using stochastic averaging gradients $\hat{\mathbf{g}}^t$ in lieu of the full gradients $\nabla f(\mathbf{x}^t)$. Therefore, DSA is a stochastic saddle point method in which the primal variables are updated as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha\hat{\mathbf{g}}^t - (\mathbf{I}-\tilde{\mathbf{Z}})\mathbf{x}^t - (\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{v}^t, \tag{27}$$

and the dual variables $\mathbf{v}^t$ are updated as

$$\mathbf{v}^{t+1} = \mathbf{v}^t + (\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^{t+1}. \tag{28}$$

Notice that the initial primal variable $\mathbf{x}^0$ is an arbitrary vector in $\mathbb{R}^{Np}$, while according to the definition $\mathbf{v}^t = \sum_{s=0}^{t}(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^s$. We then need to set the initial multiplier to $\mathbf{v}^0 = (\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^0$. This is not a problem in practice because (27) and (28) are not used for implementation. In our converge analysis we utilize the (equivalent) stochastic saddle point expressions for DSA shown in (27) and (28). The expression in (8) is used for implementation because it avoids exchanging dual variables – as well as the initialization problem. The convergence analysis is presented in the following section.

3. Convergence Analysis

Our goal here is to show that as time progresses the sequence of iterates $\mathbf{x}^t$ approaches the optimal argument $\mathbf{x}^*$. To do so, in addition to the conditions on the weight matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$ in Assumption 1, we assume the instantaneous local functions $f_{n,i}$ have specific properties that we state next.

Assumption 2 The instantaneous local functions $f_{n,i}(\mathbf{x}_n)$ are differentiable and strongly convex with parameter μ.

Assumption 3 The gradient of instantaneous local functions $\nabla f_{n,i}$ are Lipschitz continuous with parameter L, i.e., for all $n \in \{1,\ldots,N\}$ and $i \in \{1,\ldots,q_n\}$ we can write

$$\|\nabla f_{n,i}(\mathbf{a}) - \nabla f_{n,i}(\mathbf{b})\| \leq L\,\|\mathbf{a}-\mathbf{b}\| \quad \mathbf{a},\mathbf{b} \in \mathbb{R}^p. \tag{29}$$

The condition imposed by Assumption 2 implies that the local functions $f_n(\mathbf{x}_n)$ and the global cost function $f(\mathbf{x}) = \sum_{n=1}^{N} f_n(\mathbf{x}_n)$ are also strongly convex with parameter μ. Likewise, Lipschitz continuity of the local instantaneous gradients considered in Assumption 3 enforces Lipschitz continuity of the local functions gradient $\nabla f_n(\mathbf{x}_n)$ and the aggregate function gradient $\nabla f(\mathbf{x})$ – see, e.g., (Lemma 1 of Mokhtari et al. (2015a)).

3.1 Preliminaries

In this section we study some basic properties of the sequences of primal and dual variables generated by the DSA algorithm. In the following lemma, we study the relation of the iterates $\mathbf{x}^t$ and $\mathbf{v}^t$ with the optimal primal $\mathbf{x}^*$ and dual $\mathbf{v}^*$ arguments.

Lemma 3 *Consider the DSA algorithm as defined in* (6)-(9) *and recall the updates of the primal* $\mathbf{x}^t$ *and dual* $\mathbf{v}^t$ *variables in* (27) *and* (28)*, respectively. Further, define the positive semidefinite matrix* $\mathbf{U} := (\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}$*. If Assumption 1 holds true, then the sequence of primal* $\mathbf{x}^t$ *and dual* $\mathbf{v}^t$ *variables satisfy*

$$\alpha\left[\hat{\mathbf{g}}^t - \nabla f(\mathbf{x}^*)\right] = (\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})(\mathbf{x}^* - \mathbf{x}^{t+1}) + \tilde{\mathbf{Z}}(\mathbf{x}^t - \mathbf{x}^{t+1}) - \mathbf{U}(\mathbf{v}^{t+1} - \mathbf{v}^*). \tag{30}$$

Proof Considering the update rule for the dual variable in (28) and the definition $\mathbf{U} = (\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}$, we can substitute $\mathbf{U}\mathbf{v}^t$ in (27) by $\mathbf{U}\mathbf{v}^{t+1} - \mathbf{U}^2\mathbf{x}^{t+1}$. Applying this substitution into the DSA primal update in (27) yields

$$\alpha\hat{\mathbf{g}}^t = -(\mathbf{I}+\mathbf{Z}-\tilde{\mathbf{Z}})\mathbf{x}^{t+1} + \tilde{\mathbf{Z}}\mathbf{x}^t - \mathbf{U}\mathbf{v}^{t+1}. \tag{31}$$

By adding and subtracting $\tilde{\mathbf{Z}}\mathbf{x}^{t+1}$ to the right hand side of (31) and considering the fact that $(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})\mathbf{x}^* = \mathbf{0}$ we obtain

$$\alpha\hat{\mathbf{g}}^t = (\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})(\mathbf{x}^* - \mathbf{x}^{t+1}) + \tilde{\mathbf{Z}}(\mathbf{x}^t - \mathbf{x}^{t+1}) - \mathbf{U}\mathbf{v}^{t+1}. \tag{32}$$

One of the KKT conditions of problem (25) implies that the optimal variables $\mathbf{x}^*$ and $\mathbf{v}^*$ satisfy $\alpha\nabla f(\mathbf{x}^*) + \mathbf{U}\mathbf{v}^* = \mathbf{0}$ or equivalently $-\alpha\nabla f(\mathbf{x}^*) = \mathbf{U}\mathbf{v}^*$. Adding this equality to both sides of (32) follows the claim in (30). ■

In the subsequent analyses of convergence of DSA, we need an upper bound for the expected value of squared difference between the stochastic averaging gradient $\hat{\mathbf{g}}^t$ and the optimal argument gradient $\nabla f(\mathbf{x}^*)$ given the observations until step t, i.e. $\mathbb{E}\left[\|\hat{\mathbf{g}}^t - \nabla f(\mathbf{x}^*)\|^2 \mid \mathcal{F}^t\right]$. To establish this upper bound first we define the sequence $p^t \in \mathbb{R}$ as

$$p^t := \sum_{n=1}^{N}\left[\frac{1}{q_n}\sum_{i=1}^{q_n}\left(f_{n,i}(\mathbf{y}_{n,i}^t) - f_{n,i}(\tilde{\mathbf{x}}^*) - \nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^t - \tilde{\mathbf{x}}^*)\right)\right]. \tag{33}$$

Notice that based on the strong convexity of the local instantaneous functions $f_{n,i}$, each term $f_{n,i}(\mathbf{y}_{n,i}^t) - f_{n,i}(\tilde{\mathbf{x}}^*) - \nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^t - \tilde{\mathbf{x}}^*)$ is positive and as a result the sequence p^t defined in (33) is always positive. In the following lemma, we use the result in Lemma 3 to guarantee an upper bound for the expectation $\mathbb{E}\left[\|\hat{\mathbf{g}}^t - \nabla f(\mathbf{x}^*)\|^2 \mid \mathcal{F}^t\right]$ in terms of p^t and the optimality gap $f(\mathbf{x}^t) - f(\mathbf{x}^*) - \nabla f(\mathbf{x}^*)^T(\mathbf{x}^t - \mathbf{x}^*)$.

Lemma 4 *Consider the DSA algorithm in* (6)-(9) *and the definition of the sequence* p^t *in* (33)*. If Assumptions 1-3 hold true, then the squared norm of the difference between the stochastic averaging gradient* $\hat{\mathbf{g}}^t$ *and the optimal gradient* $\nabla f(\mathbf{x}^*)$ *in expectation is bounded above by*

$$\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t - \nabla f(\mathbf{x}^*)\right\|^2 | \mathcal{F}^t\right] \leq 4Lp^t + 2\left(2L - \mu\right)\left(f(\mathbf{x}^t) - f(\mathbf{x}^*) - \nabla f(\mathbf{x}^*)^T(\mathbf{x}^t - \mathbf{x}^*)\right). \tag{34}$$

Proof See Appendix A. ■

Observe that as the sequence of iterates $\mathbf{x}^t$ approaches the optimal argument $\mathbf{x}^*$, all the local auxiliary variables $\mathbf{y}_{n,i}^t$ converge to $\tilde{\mathbf{x}}^*$ which follows convergence of p^t to null. This observation in association with the result in (34) implies that the expected value of the difference between the stochastic averaging gradient $\hat{\mathbf{g}}^t$ and the optimal gradient $\nabla f(\mathbf{x}^*)$ vanishes as the sequence of iterates $\mathbf{x}^t$ approaches the optimal argument $\mathbf{x}^*$.

3.2 Convergence

In this section we establish linear convergence of the sequence of iterates $\mathbf{x}^t$ generated by DSA to the optimal argument $\mathbf{x}^*$. To do so, define $0 < \gamma$ and $\Gamma < \infty$ as the smallest and largest eigenvalues of the positive definite matrix $\tilde{\mathbf{Z}}$, respectively. Likewise, define γ' as the smallest non-zero eigenvalue of the matrix $\tilde{\mathbf{Z}} - \mathbf{Z}$ and Γ' as the largest eigenvalue of the matrix $\tilde{\mathbf{Z}} - \mathbf{Z}$. Further, define the vectors $\mathbf{u}^t, \mathbf{u}^* \in \mathbb{R}^{2Np}$ and matrix $\mathbf{G} \in \mathbb{R}^{2Np \times 2Np}$ as

$$\mathbf{u}^* := \begin{bmatrix} \mathbf{x}^* \\ \mathbf{v}^* \end{bmatrix}, \quad \mathbf{u}^t := \begin{bmatrix} \mathbf{x}^t \\ \mathbf{v}^t \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \tilde{\mathbf{Z}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}. \tag{35}$$

Observe that the vector $\mathbf{u}^* \in \mathbb{R}^{2Np}$ concatenates the optimal primal and dual variables and the vector $\mathbf{u}^t \in \mathbb{R}^{2Np}$ contains primal and dual iterates at step t. Further, $\mathbf{G} \in \mathbb{R}^{2Np \times 2Np}$ is a block diagonal positive definite matrix that we introduce since instead of tracking the value of ℓ_2 norm $\|\mathbf{u}^t - \mathbf{u}^*\|_2^2$ we study the convergence properties of $\mathbf{G}$ weighted norm $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$. Notice that the weighted norm $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ is equivalent to $(\mathbf{u}^t - \mathbf{u}^*)^T\mathbf{G}(\mathbf{u}^t - \mathbf{u}^*)$. Our goal is to show that the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ converges linearly to null. To do this we show linear convergence of a Lyapunov function of the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$. The Lyapunov function is defined as $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^t$ where $c > 0$ is a positive constant.

To prove linear convergence of the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^t$ we first show an upper bound for the expected error $\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 \mid \mathcal{F}^t\right]$ in terms of $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2$ and some parameters that capture optimality gap.

Lemma 5 *Consider the DSA algorithm as defined in* (6)-(9)*. Further recall the definitions of* p^t *in* (33) *and* $\mathbf{u}^t$, $\mathbf{u}^*$, *and* $\mathbf{G}$ *in* (35)*. If Assumptions 1-3 hold true, then for any positive constant* $\eta > 0$ *we can write*

$$\begin{aligned}\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 \mid \mathcal{F}^t\right] \leq \|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 - 2\mathbb{E}\left[\|\mathbf{x}^{t+1} - \mathbf{x}^*\|_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}^2 \mid \mathcal{F}^t\right] + \frac{\alpha 4L}{\eta}p^t \\ - \mathbb{E}\left[\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_{\tilde{\mathbf{Z}}-2\alpha\eta\mathbf{I}}^2 \mid \mathcal{F}^t\right] - \mathbb{E}\left[\|\mathbf{v}^{t+1} - \mathbf{v}^t\|^2 \mid \mathcal{F}^t\right] \\ - \left(\frac{4\alpha\mu}{L} - \frac{2\alpha(2L-\mu)}{\eta}\right)\left(f(\mathbf{x}^t) - f(\mathbf{x}^*) - \nabla f(\mathbf{x}^*)^T(\mathbf{x}^t - \mathbf{x}^*)\right).\end{aligned} \tag{36}$$

Proof See Appendix B. ■

Lemma 5 shows an upper bound for the squared norm $\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2$ which is the first part of the Lyapunov function $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + cp^t$ at step $t+1$. Likewise, we provide an upper bound for the second term of the Lyapunov function at time $t+1$ which is p^{t+1} in terms of p^t and some parameters that capture optimality gap. This bound is studied in the following lemma.

Lemma 6 *Consider the DSA algorithm as defined in* (6)-(9) *and the definition of* p^t *in* (33)*. Further, define* $q_{\min}$ *and* $q_{\max}$ *as the smallest and largest values for the number of instantaneous functions at a node, respectively. If Assumptions 1-3 hold true, then for all* $t > 0$ *the sequence* p^t *satisfies*

$$\mathbb{E}\left[p^{t+1} \mid \mathcal{F}^t\right] \leq \left[1 - \frac{1}{q_{\max}}\right]p^t + \frac{1}{q_{\min}}\left[f(\mathbf{x}^t) - f(\mathbf{x}^*) - \nabla f(\mathbf{x}^*)^T(\mathbf{x}^t - \mathbf{x}^*)\right]. \tag{37}$$

Proof See Appendix C. ■

Lemma 6 provides an upper bound for p^{t+1} in terms of its previous value p^t and the optimality error $f(\mathbf{x}^t) - f(\mathbf{x}^*) - \nabla f(\mathbf{x}^*)^T(\mathbf{x}^t - \mathbf{x}^*)$. Combining the results in Lemmata 5 and 6 we can show that in expectation the Lyapunov function $\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^{t+1}$ at step $t+1$ is strictly smaller than its previous value $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^t$ at step t.

Theorem 7 *Consider the DSA algorithm as defined in* (6)-(9)*. Further recall the definition of the sequence* p^t *in* (33)*. Define* η *as an arbitrary positive constant chosen from the interval*

$$\eta \in \left(\frac{L^2 q_{\max}}{\mu q_{\min}} + \frac{L^2}{\mu} - \frac{L}{2}\ ,\ \infty\right). \tag{38}$$

If Assumptions 1-3 hold true and the stepsize α *is chosen from the interval* $\alpha \in (0, \gamma/2\eta)$*, then for arbitrary* c *chosen from the interval*

$$c \in \left(\frac{4\alpha L q_{\max}}{\eta}\ ,\ \frac{4\alpha\mu q_{\min}}{L} - \frac{2\alpha q_{\min}(2L-\mu)}{\eta}\right), \tag{39}$$

there exits a positive constant $0 < \delta < 1$ *such that*

$$\mathbb{E}\left[\|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^{t+1} \mid \mathcal{F}^t\right] \leq (1-\delta)\left(\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^t\right). \tag{40}$$

Proof See Appendix D. ■

We point out that the linear convergence constant δ in (40) is explicitly available – see (100) in Appendix D. It is a function of the strong convexity parameter μ, the Lipschitz continuity constant L, lower and upper bounds on the eigenvalues of the matrices $\tilde{\mathbf{Z}}$, $\tilde{\mathbf{Z}} - \mathbf{Z}$, and $\mathbf{I} + \mathbf{Z} - 2\tilde{\mathbf{Z}}$, the smallest $q_{\min}$ and largest $q_{\max}$ values for the number of instantaneous functions available at a node, and the stepsize α. Insight on the dependence of δ with problem parameters is offered in Section 3.3.

The inequality in (40) shows that the expected value of the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^t$ at time $t+1$ given the observation until step t is strictly smaller than the previous iterate at step t. Note that, it is not hard to verify that if the positive constant η is chosen from the interval in (38), the interval in (39) is non-empty. Computing the expected value with respect to the initial sigma field $\mathbb{E}\left[.\mid \mathcal{F}^0\right] = \mathbb{E}\left[.\right]$ implies that in expectation the sequence $\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^t$ converges linearly to null, i.e.,

$$\mathbb{E}\left[\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^t\right] \leq (1-\delta)^t\left(\|\mathbf{u}^0 - \mathbf{u}^*\|_{\mathbf{G}}^2 + c\,p^0\right). \tag{41}$$

We use the result in (41) to establish linear convergence of the sequence of squared norm error $\|\mathbf{x}^t - \mathbf{x}^*\|^2$ in expectation.

Corollary 8 *Consider the DSA algorithm as defined in* (6)-(9) *and recall* γ *is the minimum eigenvalue of the positive definite matrix* $\tilde{\mathbf{Z}}$*. Suppose the conditions of Theorem 7 hold, then there exits a positive constant* $0 < \delta < 1$ *such that*

$$\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^*\|^2\right] \leq (1-\delta)^t \frac{\left(\|\mathbf{u}^0 - \mathbf{u}^*\|^2_{\mathbf{G}} + c\,p^0\right).}{\gamma} \tag{42}$$

Proof First note that according to the definitions of $\mathbf{u}$ and $\mathbf{G}$ in (35) and the definition of p^t in (33) , we can write $\|\mathbf{x}^t - \mathbf{x}^*\|^2_{\tilde{\mathbf{Z}}} \leq \|\mathbf{u}^t - \mathbf{u}^*\|^2_{\mathbf{G}} + c\,p^t$. Further, note that the weighted norm $\|\mathbf{x}^t - \mathbf{x}^*\|^2_{\tilde{\mathbf{Z}}}$ is lower bounded by $\gamma\|\mathbf{x}^t - \mathbf{x}^*\|^2$, since γ is a lower bound for the eigenvalues of $\tilde{\mathbf{Z}}$. Combine these two observations to obtain $\gamma\|\mathbf{x}^t - \mathbf{x}^*\|^2 \leq \|\mathbf{u}^t - \mathbf{u}^*\|^2_{\mathbf{G}} + c\,p^t$. This inequality in conjunction with the expression in (41) follows the claim in (42). ■

Corollary 8 states that the sequence $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^*\|^2\right]$ linearly converges to null. Note that the sequence $\mathbb{E}\left[\|\mathbf{x}^t - \mathbf{x}^*\|^2\right]$ is not necessarily monotonically decreasing as the sequence $\mathbb{E}\left[\|\mathbf{u}^t - \mathbf{u}^*\|^2_{\mathbf{G}} + c\,p^t\right]$ is. The result in (42) shows linear convergence of the sequence of variables generated by DSA in expectation. In the following Theorem we show that the local variables $\mathbf{x}_n^t$ generated by DSA almost surely converge to the optimal argument of (1).

Theorem 9 *Consider the DSA algorithm as defined in* (6)-(9) *and suppose the conditions of Theorem 7 hold. Then, the sequences of the local variables* $\mathbf{x}_n^t$ *for all* $n = 1, \ldots, N$ *converge almost surely to the optimal argument* $\tilde{\mathbf{x}}^*$*, i.e.,*

$$\lim_{t\to\infty} \mathbf{x}_n^t \;=\; \tilde{\mathbf{x}}^* \quad a.s. \qquad \text{for all } n = 1, \ldots, N. \tag{43}$$

Proof See Appendix E. ■

Theorem 9 provides almost sure convergence of $\mathbf{x}^t$ to the optimal solution $\mathbf{x}^*$ which is stronger result than convergence in expectation as in Corollary 8.

3.3 Convergence Constant

The constant δ that controls the speed of convergence can be simplified by selecting specific values for η, α, and c. This uncovers connections to the properties of the local objective functions and the network topology. To make this clearer recall the definitions of γ and Γ as the smallest and largest eigenvalues of the positive definite matrix $\tilde{\mathbf{Z}}$, respectively, and γ' and Γ' as the smallest and largest *positive* eigenvalues of the positive semi-definite matrix $\tilde{\mathbf{Z}} - \mathbf{Z}$, respectively. Further, recall that the local objective functions are strongly convex with constant μ and their gradients are Lipschitz continuous with constant L. Then, define the condition numbers of the objective function and the graph as

$$\kappa_f = \frac{L}{\mu}, \qquad \kappa_g = \frac{\max\{\Gamma, \Gamma'\}}{\min\{\gamma, \gamma'\}}, \tag{44}$$

respectively. The condition number of the function is a measure of how difficult it is to minimize the local functions using gradient descent directions. The condition number of the graph is a measure of how slow the graph is in propagating a diffusion process. Both are known to control the speed of convergence of distributed optimization methods. The following corollary illustrates that these condition numbers also determine the convergence speed of DSA.

Corollary 10 *Consider the DSA algorithm as defined in* (6)-(9) *and suppose the conditions of Theorem 7 hold. Choose the weight matrices* $\mathbf{W}$ *and* $\tilde{\mathbf{W}}$ *as* $\tilde{\mathbf{W}} = (\mathbf{I}+\mathbf{W})/2$*, assign the same number*

of instantaneous local functions $f_{n,i}$ *to each node, i.e.,* $q_{\min} = q_{\max} = q$*, and set the constants* η*,* α *and* c *as*

$$\eta = \frac{2L^2}{\mu}, \quad \alpha = \frac{\gamma\mu}{8L^2}, \quad c = \frac{q\gamma\mu^2}{4L^3}\left(1 + \frac{\mu}{4L}\right). \tag{45}$$

The linear convergence constant $0 < \delta < 1$ *in* (40) *reduces to*

$$\delta = \min\left[\frac{1}{16\kappa_g^2},\; \frac{1}{q[1 + 4\kappa_f(1+\gamma/\gamma')]},\; \frac{1}{4(\gamma/\gamma')\kappa_f + 32\kappa_g\kappa_f^4}\right]. \tag{46}$$

Proof The given values for η, α, and c satisfy the conditions in Theorem 7. Substitute then these values into the expression for δ in (100). Simplify terms and utilize the condition number definitions in (44). The second term in the minimization in (100) becomes redundant because it is dominated by the first. ■

Observe that while the choices of η, α, and c in (45) satisfy all the required conditions of Theorem 7, they are not necessarily optimal for maximizing the linear convergence constant δ. Nevertheless, the expression in (46) shows that the convergence speed of DSA decreases with increases in the graph condition number κ_g, the local functions condition number κ_f, and the number of functions assigned to each node q. For a cleaner expression observe that both, γ and γ' are the minimum eigenvalues of the weight matrix $\mathbf{W}$ and the weight matrix difference $\tilde{\mathbf{W}} - \mathbf{W}$. They can therefore be chosen to be of similar order. For reference, say that we choose $\gamma = \gamma'$ so that the ratio $\gamma/\gamma' = 1$. In that case, the constant δ in (46) reduces to

$$\delta = \min\left[\frac{1}{16\kappa_g^2},\; \frac{1}{q(1 + 8\kappa_f)},\; \frac{1}{4(\kappa_f + 8\kappa_f^4\kappa_g)}\right]. \tag{47}$$

The three terms in (47) establish separate regimes, problems where the graph condition number is large, problems where the number of functions at each node is large, and problems where the condition number of the local functions are large. In the first regime the first term in (47) dominates and establishes a dependence in terms of the square of the graph's condition number. In the second regime the middle term dominates and results in an inverse dependence with the number of functions available at each node. In the third regime, the third term dominates. The dependence in this case is inversely proportional to κ_f^4.

4. Numerical Experiments

We numerically study the performance of DSA in solving a logistic regression problem. In this problem we are given $Q = \sum_{n=1}^N q_n$ training samples that we distribute across N distinct nodes. Denote q_n as the number of samples that are assigned to node n. We assume that the samples are distributed equally over the nodes, i.e., $q_n = q_{max} = q_{min} = q = Q/N$ for $n = 1, \ldots, N$. The training points at node n are denoted by $\mathbf{s}_{n,i} \in \mathbb{R}^p$ for $i = 1, \ldots, q_n$ with associated labels $l_{n,i} \in \{-1, 1\}$. The goal is to predict the probability $\mathrm{P}\left(l = 1 \mid \mathbf{s}\right)$ of having label $l = 1$ for sample point $\mathbf{s}$. The logistic regression model assumes that this probability can be computed as $\mathrm{P}\left(l = 1 \mid \mathbf{s}\right) = 1/(1+\exp(-\mathbf{s}^T\mathbf{x}))$ given a linear classifier $\mathbf{x}$ that is computed based on the training samples. It follows from this model that the regularized maximum log likelihood estimate of the classifier $\mathbf{x}$ given the training samples $(\mathbf{s}_{n,i}, l_{n,i})$ for $i = 1, \ldots, q_n$ and $n = 1, \ldots, N$ is the solution of the optimization problem

$$\tilde{\mathbf{x}}^* := \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^p} \frac{\lambda}{2}\|\mathbf{x}\|^2 + \sum_{n=1}^N \sum_{i=1}^{q_n} \log\left(1 + \exp(-l_{n,i}\mathbf{s}_{n,i}^T\mathbf{x})\right), \tag{48}$$

where the regularization term $(\lambda/2)\|\mathbf{x}\|^2$ is added to reduce overfitting to the training set. The optimization problem in (48) can be written in the form of (1) by defining the local objective functions f_n as

$$f_n(\mathbf{x}) = \frac{\lambda}{2N}\|\mathbf{x}\|^2 + \sum_{i=1}^{q_n} \log\Big(1+\exp(-l_{n,i}\mathbf{s}_{n,i}^T\mathbf{x})\Big). \tag{49}$$

Observe that the local functions f_n in (49) can be written as the average of a set of instantaneous functions $f_{n,i}$ defined as

$$f_{n,i}(\mathbf{x}) = \frac{\lambda}{2N}\|\mathbf{x}\|^2 + q_n \log\Big(1+\exp\big(-l_{n,i}\mathbf{s}_{n,i}^T\mathbf{x}\big)\Big), \tag{50}$$

for all $i = 1, \ldots, q_n$. Considering the definitions of the instantaneous local functions $f_{n,i}$ in (50) and the local functions f_n in (49), problem (48) can be solved using the DSA algorithm.

In our experiments in Sections 4.1-4.4, we use a synthetic dataset where the components of the feature vectors $\mathbf{s}_{n,i}$ with label $l_{n,i} = 1$ are generated from a normal distribution with mean μ and standard deviation σ_+, while sample points with label $l_{n,i} = -1$ are generated from a normal distribution with mean $-\mu$ and standard deviation σ_-. In Section 4.5, we consider a large-scale real dataset for training the classifier.

We consider a network of size N where the edges between nodes are generated randomly with probability p_c. The weight matrix $\mathbf{W}$ is generated using the Laplacian matrix $\mathbf{L}$ of network as

$$\mathbf{W} = \mathbf{I} - \mathbf{L}/\tau, \tag{51}$$

where τ should satisfy $\tau > (1/2)\lambda_{\max}(\mathbf{L})$. In our experiments we set this parameter as $\tau = (2/3)\lambda_{\max}(\mathbf{L})$. We capture the error of each algorithm by the sum of squared differences of the local iterates $\mathbf{x}_n^t$ from the optimal solution $\tilde{\mathbf{x}}^*$ as

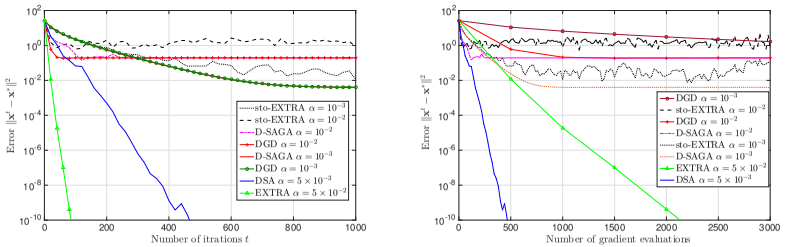
$$e^t = \|\mathbf{x}^t - \mathbf{x}^*\|^2 = \sum_{n=1}^{N} \|\mathbf{x}_n^t - \tilde{\mathbf{x}}^*\|^2. \tag{52}$$

We use a centralized algorithm for computing the optimal argument $\tilde{\mathbf{x}}^*$ in all of our experiments.

4.1 Comparison with Decentralized Methods

We provide a comparison of DSA with respect to DGD, EXTRA, stochastic EXTRA, and decentralized SAGA. The stochastic EXTRA (sto-EXTRA) is defined by using the stochastic gradient in (5) instead of using full gradient as in EXTRA or stochastic averaging gradient as in DSA. The decentralized SAGA (D-SAGA) is a stochastic version of the DGD algorithm that uses stochastic averaging gradient instead of exact gradient which is the naive approach for developing a decentralized version of the SAGA algorithm. In our experiments, the weight matrix $\tilde{\mathbf{W}}$ in EXTRA, stochastic EXTRA, and DSA is chosen as $\tilde{\mathbf{W}} = (\mathbf{I}+\mathbf{W})/2$. We use the total number of sample points $Q = 500$, feature vectors dimension $p = 2$, regularization parameter $\lambda = 10^{-4}$, probability of existence of an edge $p_c = 0.35$. To make the dataset *not* linearly separable we set the mean as $\mu = 2$ and the standard deviations to $\sigma_+ = \sigma_- = 2$. Moreover, the maximum eigenvalue of the Laplacian matrix is $\lambda_{\max}(\mathbf{L}) = 8.017$ which implies that the choice of τ in (51) is $\tau = (2/3)\lambda_{\max}(\mathbf{L}) = 5.345$. We set the total number of nodes $N = 20$ which implies that each node has access to $q = Q/N = 25$ samples.

Fig. 2 illustrates the convergence paths of DSA, EXTRA, DGD, Stochastic EXTRA, and Decentralized SAGA with constant stepsizes for $N = 20$ nodes. For EXTRA and DSA different stepsizes are chosen and the best performance for EXTRA and DSA are achieved by $\alpha = 5 \times 10^{-2}$ and $\alpha = 5 \times 10^{-3}$, respectively. It is worth mentioning that the choice of stepsize α for DSA in practice is larger than the theoretical result in Theorem 6 and Corollary 9 which suggest stepsize of the



(a) (b)

Figure 2: Convergence paths of DSA, EXTRA, DGD, Stochastic EXTRA, and Decentralized SAGA for a logistic regression problem with $Q = 500$ samples and $N = 20$ nodes. Distance to optimality $e^t = \|\mathbf{x}^t - \mathbf{x}^*\|^2$ is shown with respect to number of iterations t and number of gradient evaluations in Fig 2(a) and Fig. 2(b), respectively. DSA and EXTRA converge linearly to the optimal argument $\mathbf{x}^*$, while DGD, Stochastic EXTRA, and Decentralized SAGA with constant step sizes converge to a neighborhood of the optimal solution. Smaller choice of stepsize for DGD, Stochastic EXTRA, and Decentralized SAGA leads to a more accurate convergence, while the speed of convergence becomes slower. DSA outperforms EXTRA in terms of number of gradient evaluations to achieve a target accuracy.

order $O(\mu/L^2)$. As shown in Fig. 2, DSA is the only stochastic algorithm that converges linearly. Decentralized SAGA after a few iterations achieves the performance of DGD and they both cannot converge to the optimal argument. By choosing a smaller stepsize as $\alpha = 10^{-3}$, they reach a more accurate convergence relative to the case that the stepsize is $\alpha = 10^{-2}$; however, the speed of convergence is slower for the smaller stepsize. Stochastic EXTRA also suffers from inexact convergence, but for a different reason. DGD and decentralized SAGA have inexact convergence since they solve a penalty version of the original problem, while stochastic EXTRA can not reach the optimal solution since the noise of stochastic gradient is not vanishing. DSA resolves both issues by combining the idea of stochastic averaging from SAGA to control the noise of stochastic gradient estimation and the double descent idea of EXTRA to solve the correct optimization problem.

Fig. 2(a) illustrates convergence paths of the considered methods in terms of number of iterations t. Notice that the number of iterations t indicates the number of local iterations processed at each node. Convergence rate of EXTRA is faster than DSA in terms of number of iterations or equivalently number of communications as shown in Fig. 2(a); however, the complexity of each iteration for EXTRA is higher than DSA. Therefore, it is reasonable to compare the performances of these algorithms in terms of number of processed feature vectors or equivalently number of gradient evaluations. For instance, DSA requires $t = 380$ iterations or equivalently 380 gradient evaluations to achieve the error $e^t = 10^{-8}$, while to achieve the same accuracy EXTRA requires $t = 69$ iterations which is equivalent to $t \times q_n = 69 \times 25 = 1725$ processed feature vectors or gradient evaluations.

To illustrate this difference better, we compare the convergence paths of DSA, EXTRA, DGD, Stochastic EXTRA, and Decentralized SAGA in terms of number of gradient evaluations in Fig. 2(b). Note that the total number of gradient evaluations at each node for the stochastic methods such as DSA, sto-EXTRA, and D-SAGA is equal to the the number of iterations t, while for EXTRA and DGD – which are deterministic methods – the number of gradient evaluations is equal to the product $t \times q$. This is true since each node in the stochastic methods only evaluates 1 gradient

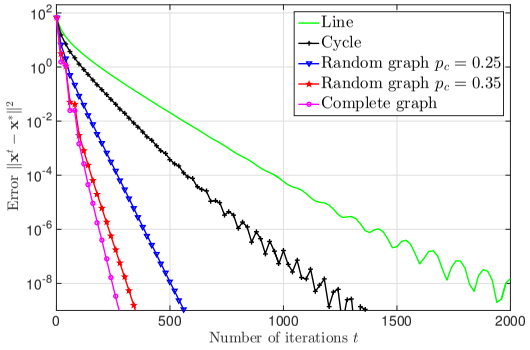


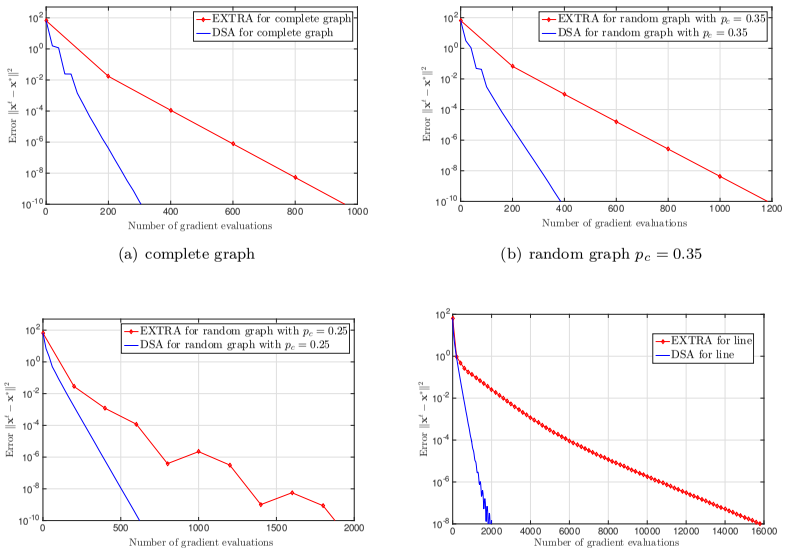
Figure 3: Convergence of DSA for different network topologies when the total number of samples is $Q = 500$ and the size of network is $N = 50$. Distance to optimality $e^t = \|\mathbf{x}_t - \mathbf{x}^*\|^2$ is shown with respect to number of iterations t. As the graph condition number κ_g becomes larger the linear convergence of DSA becomes slower. The best performance belongs to the complete graph which has the smallest condition number and the slowest convergence path belongs to the line graph which has the largest graph condition number.

per iteration, while in the deterministic methods each node requires q gradient evaluations per iteration. The convergence paths in Fig. 2(b) showcase the advantage of DSA relative to EXTRA in requiring less processed feature vectors (or equivalently gradient evaluations) for achieving a specific accuracy. It is important to mention that the initial gradient evaluations for the DSA method is not considered in Fig. 2(b) since the initial decision variable is $\mathbf{x}^0 = \mathbf{0}$ in all experiments and evaluation of the initial gradients $\nabla f_{n,i}(\mathbf{x}^0) = -(1/2)ql_{n,i}\mathbf{s}_{n,i}$ is not computationally expensive relative to the general gradient computation which is given by $\nabla f_{n,i}(\mathbf{x}) = (\lambda\mathbf{x}/N) - (ql_{n,i}\mathbf{s}_{n,i})/(1 + \exp(l_{n,i}\mathbf{x}^T\mathbf{s}_{n,i}))$. However, if we consider this initial processing the plot for DSA in Fig. 2(b) will be shifted by $q = 25$ gradient evaluations which doesn't change the conclusion that DSA outperforms EXTRA in terms of gradient evaluations

4.2 Effect of Graph Condition Number κ_g

In this section we study the effect of the graph condition number κ_g as defined in (44) on the performance of DSA. We keep the parameters in Fig. 2 except for the network size N which we set as $N = 50$. Thus, each node has access to $q = 500/50 = 10$ sample points. The convergence paths of the DSA algorithm for random networks with $p_c = 0.25$ and $p_c = 0.35$, complete graph, cycle, and line are shown in Fig. 3. Notice that the graph condition number of the line graph, cycle graph, random graph with $p_c = 0.25$, random graph with $p_c = 0.35$, and complete graph are $\kappa_g = 1.01 \times 10^3$, $\kappa_g = 2.53 \times 10^2$, $\kappa_g = 17.05$, $\kappa_g = 4.87$, and $\kappa_g = 4$, respectively. For each network topology, we have hand-optimized the stepsize α and the best choice of stepsize for the complete graph, random graph with $p_c = 0.35$, random graph with $p_c = 0.25$, cycle, and line are $\alpha = 2 \times 10^{-2}$, $\alpha = 1.5 \times 10^{-2}$, $\alpha = 10^{-2}$, $\alpha = 5 \times 10^{-3}$, and $\alpha = 3 \times 10^{-3}$, respectively.

As we expect for the topologies that the graph has more edges and the graph condition number κ_g is smaller we observe a faster linear convergence for DSA. The best performance belongs to the



(a) complete graph (b) random graph $p_c = 0.35$

(c) random graph $p_c = 0.25$ (d) line

Figure 4: Convergence paths of DSA and EXTRA for different network topologies when the total number of samples is $Q = 500$ and the size of network is $N = 50$. Distance to optimality $e^t = \|\mathbf{x}^t - \mathbf{x}^*\|^2$ is shown with respect to number of gradient evaluations. DSA converges faster relative to EXTRA in all of the considered networks. The difference between the convergence paths of DSA and EXTRA is more substantial when the graph has a large condition number κ_g. The stepsize α for DSA and EXTRA in all the considered cases is hand-optimized and the results for the best choice of α are reported.

complete graph which requires $t = 247$ iterations to achieve the relative error $e^t = 10^{-8}$. In the random graphs with connectivity probabilities $p_c = 0.35$ and $p_c = 0.25$, DSA achieves the relative error $e^t = 10^{-8}$ after $t = 310$ and $t = 504$ iterations, respectively. For the cycle and line graphs the numbers of required iterations for reaching the relative error $e^t = 10^{-8}$ are $t = 1133$ and $t = 1819$, respectively. These observations match the theoretical result in (47) that DSA converges faster when the graph condition number κ_g is smaller.

We also compare the performances of DSA and EXTRA over different topologies to verify the claim that DSA is more efficient than EXTRA in terms of number of gradient evaluations over different network topologies. The parameters are as in Fig. 3 and the stepsize α for EXTRA in different topologies are optimized separately. In particular, the best stepsize for the complete graph, random graph with $p_c = 0.35$, random graph with $p_c = 0.25$, and line are $\alpha = 6 \times 10^{-2}$, $\alpha = 5 \times 10^{-2}$, $\alpha = 3 \times 10^{-2}$, and $\alpha = 5 \times 10^{-2}$, respectively. Fig. 4 shows the convergence paths of DSA and EXTRA versus number of gradient evaluations for four different network topologies. We observe

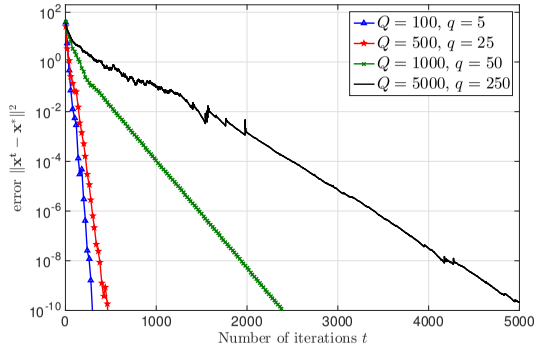


Figure 5: Comparison of convergence paths of DSA for different number of samples Q when the network size is $N = 20$ and the graph is randomly generated with the connectivity ratio $p_c = 0.35$. Convergence time for DSA increases by increasing the total number of sample points Q which is equivalent to increasing the number of samples at each node $q = Q/N$.

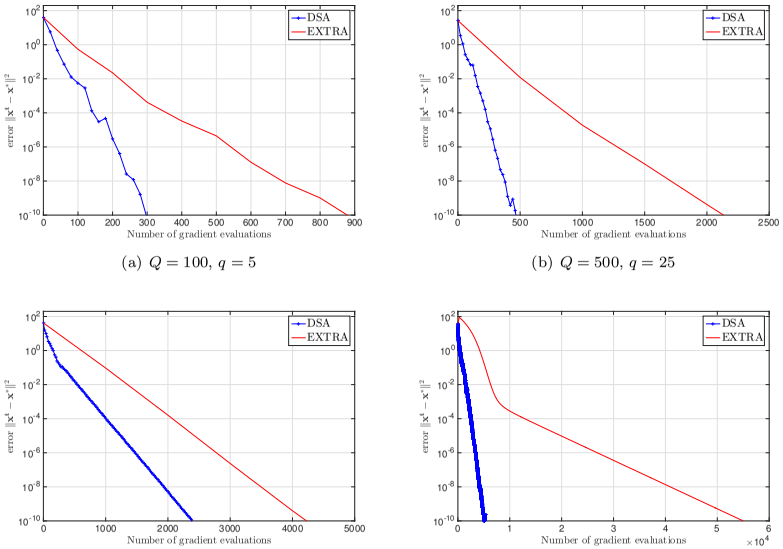
that in the considered graphs, DSA achieves a target accuracy $\|\mathbf{x}^t - \mathbf{x}^*\|^2$ faster than EXTRA. In other words, to achieve a specific accuracy $\|\mathbf{x}^t - \mathbf{x}^*\|^2$ DSA requires less number of local gradient evaluations relative to EXTRA. In addition, the gap between the performance of DSA and EXTRA is more substantial when the graph condition number κ_g is larger. In particular, in the case that we have a complete graph, which has a small graph condition number, the difference between the convergence paths of DSA and EXTRA is less significant comparing to the line graph which has a large graph condition number.

4.3 Effect of Number of Functions (Samples) at Each Node q

To evaluate performance for different number of functions (sample points) available at each node which is indicated by q, we use the same setting as in Fig. 2; however, we consider scenarios with different number of samples Q which leads to different number of samples at each node q. To be more precise, we fix the total number of nodes in the network as $N = 20$ and we consider the cases that the total number of samples are $Q = 100$, $Q = 500$, $Q = 1000$, and $Q = 5000$ where the corresponding number of samples at each node are $q = 5$, $q = 25$, $q = 50$, and $q = 250$, respectively. Similar to the experiment in Fig. 2, the graph is generated randomly with connectivity ratio $p_c = 0.35$.

For each of these scenarios the DSA stepsize α is hand-optimized and the best choice is used for comparison with others. The results are reported for $\alpha = 10^{-4}$, $\alpha = 10^{-3}$, $\alpha = 5 \times 10^{-3}$, and $\alpha = 10^{-1}$ when the total number of samples are $Q = 5000$, $Q = 1000$, $Q = 500$, $Q = 100$, respectively. The resulting convergence paths are shown in Fig. 5.

The convergence paths in Fig. 5 show that as we increase the total number of samples Q and consequently the number of assigned samples to each node q, we observe that DSA converges slower to the optimal argument. This conclusion is expected from the theoretical result in (47) which shows that the linear convergence rate of DSA becomes slower by increasing q. In particular, to achieve the target accuracy of $\|\mathbf{x}^t - \mathbf{x}^*\|^2 = 10^{-8}$ DSA requires $t = 260$, $t = 380$, $t = 1960$, and $t = 4218$



(a) $Q = 100,\ q = 5$ (b) $Q = 500,\ q = 25$

(c) $Q = 1000,\ q = 50$ (d) $Q = 5000,\ q = 250$

Figure 6: Convergence paths of DSA and EXTRA for the cases that $(Q = 100, q = 5)$, $(Q = 500, q = 25)$, $(Q = 1000, q = 50)$, and $(Q = 5000, q = 250)$ are presented. Distance to optimality $e^t = \|\mathbf{x}^t - \mathbf{x}^*\|^2$ is shown with respect to number of gradient evaluations. The total number of nodes in the network is fixed and equal to $N = 20$ and the graph is randomly generated with the connectivity ratio $p_c = 0.35$. DSA converges faster relative to EXTRA and they both converge slower when the total number of samples Q increases.

iterations (or equivalently gradient evaluations) for the cases that $q = 5$, $q = 25$, $q = 50$, $q = 250$, respectively.

To have a more comprehensive comparison of DSA and EXTRA, we also compare their performances under the four different settings considered in Fig. 5. The convergence paths of these methods in terms of number of gradient evaluations for $(Q = 100, q = 5)$, $(Q = 500, q = 25)$, $(Q = 1000, q = 50)$, and $(Q = 5000, q = 250)$ are presented in Fig 6. The optimal stepsizes for EXTRA in the considered settings are $\alpha = 4 \times 10^{-1}$, $\alpha = 5 \times 10^{-2}$, $\alpha = 3 \times 10^{-2}$, and $\alpha = \times 10^{-2}$, respectively. An interesting observation is the effect of q on the convergence rate of EXTRA. We observe that EXTRA converges slower as the number of samples at each node q increases which is identical to the observation for DSA in Fig. 5. Moreover, for all of the settings considered in Fig. 6, DSA outperforms EXTRA in terms of number of required gradient evaluations until convergence. Moreover, by increasing the total number of samples Q and subsequently the number of assigned samples to each node q the advantage of DSA with respect to EXTRA in terms of computational complexity becomes more significant. This observation justifies the use of DSA for large-scale optimization problems as we consider in Section 4.5.

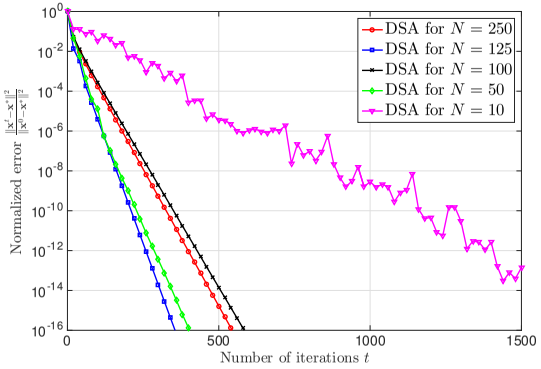
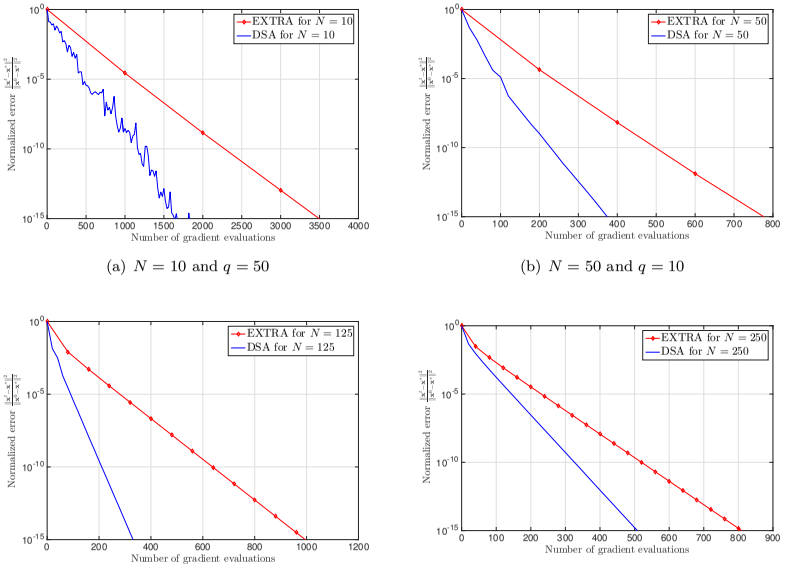


Figure 7: Normalized error $\|\mathbf{x}^t-\mathbf{x}^*\|^2/\|\mathbf{x}^0-\mathbf{x}^*\|^2$ of DSA versus number of iterations t for networks with different number of nodes N when the total number of samples is fixed $Q = 500$. The graphs are randomly generated with the connectivity ratio $p_c = 0.35$. Picking a very small or large value for N which leads to a very large or small value for q, respectively, is not preferable. The best performance belongs to the case that $N = 125$ and $q = 4$.

4.4 Effect of Number of Nodes N

In some settings, we can choose the number of nodes (processors) N for training the dataset. In this section, we study the effect of network size N on the convergence path of DSA when a fixed number of samples Q is given to train the classifier $\mathbf{x}$. Notice that when Q is fixed, by changing the number of nodes N, the number of assigned samples to each node $q = Q/N$ changes proportionally. Then, we may want to pick the number of nodes N or equivalently the number of assigned samples to each node q which leads to the best performance of DSA for training Q samples. Hence, we fix the total number of sample points as $Q = 500$ and assign the same amount of sample points q to each node. We consider 5 different settings with $N = 10$, $N = 50$, $N = 100$, $N = 125$, and $N = 250$ which their corresponding number of assigned samples to each node are $q = 50$, $q = 10$, $q = 5$, $q = 4$, and $q = 2$, respectively. The DSA stepsize for each of the considered settings is hand-optimized. The stepsizes $\alpha = 5 \times 10^{-3}$, $\alpha = 2 \times 10^{-2}$, $\alpha = 6 \times 10^{-2}$, and $\alpha = 8 \times 10^{-2}$ are considered for the cases that the number of assigned samples to each node are $q = 50$, $q = 10$, $q = 5$, $q = 4$, and $q = 2$, respectively.

Fig. 7 shows the convergence paths of DSA for networks with different number of nodes. Notice that the normalized error $\tilde{e}^t = \|\mathbf{x}^t-\mathbf{x}^*\|^2/\|\mathbf{x}^0-\mathbf{x}^*\|^2$ is reported, since the dimension of the vector $\mathbf{x}$ is different for different choices of N. Comparison of the convergence paths in Fig. 7 shows that the best performance belongs to the case that $N = 125$ and each node has access to $q = 4$ sample points. The performance of DSA becomes worse for the case that there are $N = 5$ nodes in the network and each node has $q = 100$ sample points. This observation implies that the DSA algorithm is also preferable to SAGA which corresponds to the case that $N = 1$. Moreover, we observe that when the number of nodes is large as $N = 250$ and each node has access to $q = 2$ samples, DSA doesn't perform well. Thus, increasing the size of network N doesn't always lead to a better performance for DSA. The best performance is observed when a moderate subset of the samples is assigned to each node.



(a) $N = 10$ and $q = 50$

(b) $N = 50$ and $q = 10$

(c) $N = 125$ and $q = 4$

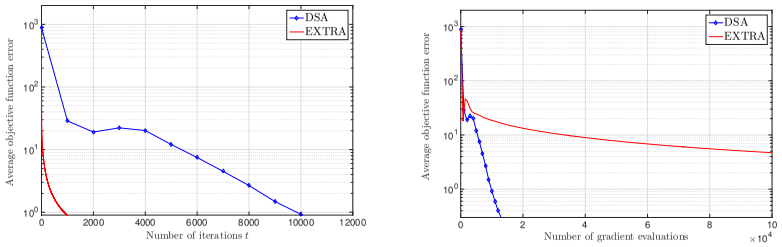
(d) $N = 250$ and $q = 2$

Figure 8: Convergence paths of DSA and EXTRA for different different number of nodes N when the total number of sample points is fixed as $Q = 500$. The graphs are randomly generated with the connectivity ratio $p_c = 0.35$. Normalized distance to optimality $\tilde{e}^t = \|\mathbf{x}^t - \mathbf{x}^*\|^2/\|\mathbf{x}^0 - \mathbf{x}^*\|^2$ is shown with respect to number of gradient evaluations. DSA converges faster relative to EXTRA in all of the considered settings.

We also study the convergence rates of DSA and EXTRA in terms of number of gradient evaluations for networks with different number of nodes N. Fig. 8 demonstrates the convergence paths of DSA and EXTRA for the cases that $N = 10$, $N = 50$, $N = 125$, and $N = 250$. Similar to DSA, we report the best performance of EXTRA for each setting which is achieved by the stepsizes $\alpha = 5 \times 10^{-2}$, $\alpha = 8 \times 10^{-2}$, $\alpha = 8 \times 10^{-2}$, and $\alpha = 10^{-1}$ for $N = 10$, $N = 50$, $N = 125$, and $N = 250$, respectively. Observe that in all settings DSA is more efficient relative to EXTRA and it requires less number of gradient evaluations for convergence.

4.5 Large-scale Classification Application

In this section we solve the logistic regression problem in (48) for the protein homology dataset provided in KDD Cup 2004. The dataset contains $Q = 1.45 \times 10^5$ sample points and each sample point has $p = 74$ features. We consider the case that the sample points are distributed over $N = 200$ nodes which implies that each node has access to $q = 725$ samples. We set the connectivity ratio $p_c = 0.35$ and hand optimize the stepsize α for DSA and EXTRA separately. The best performance of DSA is observed for $\alpha = 2 \times 10^{-7}$ and the best choice of stepize for EXTRA is $\alpha = 6 \times 10^{-7}$. We capture the error in terms of the average objective function error e^t_{avg} of the network which is



(a) (b)

Figure 9: Convergence paths of DSA and EXTRA for the protein homology classification problem with $Q = 1.45 \times 10^5$ samples. The graph has $N = 200$ nodes and it is randomly generated with the connectivity ratio $p_c = 0.35$. The average objective function error is shown with respect to number of iterations t and number of gradient evaluations, respectively.

defined as

$$e_{avg}^t := \frac{1}{N} \sum_{m=1}^{N} \left[\sum_{n=1}^{N} f_n(\mathbf{x}_m^t) - \sum_{n=1}^{N} f_n(\mathbf{x}^*) \right]. \tag{53}$$

Note that the difference $\sum_{n=1}^N f_n(\mathbf{x}_m^t) - \sum_{n=1}^N f_n(\mathbf{x}^*)$ shows the objective function error associated with the decision variable of node m at time t. Thus, the expression in (53) indicates the average objective function error of the network at step t.

The average objective function error for DSA and EXTRA in terms of number of iterations t and number of gradient evaluations are presented in Fig. 9(a) and Fig. 9(b), respectively. As we observe, the results in Fig. 9 for the large-scale classification problem match the observations in Fig. 2 for the classification problem with a synthetic dataset. In particular, both algorithms converge linearly, while EXTRA converges faster than DSA in terms of number of iterations or equivalently in terms of communication cost. On the other hand, DSA outperforms EXTRA in terms of computational complexity or number of required gradients to reach a target accuracy. Moreover, notice that the difference between the performances of DSA and EXTRA in terms of number of gradient evaluations is more significant in Fig. 9(b) relative to the one in Fig. 2(b). Thus, by increasing the problem dimension we obtain more computational complexity benefit by using DSA instead of EXTRA.

5. Conclusions

Decentralized double stochastic averaging gradient (DSA) is proposed as an algorithm for solving decentralized optimization problems where the local functions can be written as an average of a set of local instantaneous functions. DSA exploits stochastic averaging gradients in lieu of gradients and mixes information of two consecutive iterates to determine the descent direction. By assuming strongly convex local instantaneous functions with Lipschitz continuous gradients, the DSA algorithm converges linearly to the optimal arguments in expectation. In addition, the sequence of local iterates $\mathbf{x}_n^t$ for each node in the network almost surely converges to the optimal argument $\tilde{\mathbf{x}}^*$. A comparison between the DSA algorithm and a group of stochastic and deterministic alternatives are provided for solving a logistic regression problem. The numerical results show DSA is the only stochastic decentralized algorithm to reach linear convergence. DSA outperforms decentralized stochastic alternatives in terms of number of required iteration for convergence, and exhibits faster

convergence relative to deterministic alternatives in terms of number feature vectors processed until convergence.

DSA utilizes the idea of stochastic averaging gradient suggested in SAGA to reduce the computational cost of EXTRA. Although, this modification is successful in reducing the computational complexity of EXTRA and remaining the convergence rate linear, it requires stronger assumptions to prove the linear convergence. In DSA, the local instantaneous functions are required to be strongly convex which is a stricter assumption relative to the required condition for EXTRA that the global objective function should be strongly convex. This assumption for the linear convergence of DSA is inherited from the SAGA algorithm and it can be relaxed by using SVRG (Johnson and Zhang (2013)) instead of SAGA for estimating the gradients of the local functions. This modification in the update of DSA is an obvious extension of the current work and can be considered as a future research direction.

Acknowledgments

We acknowledge the support of the National Science Foundation (NSF CAREER CCF-0952867) and the Office of Naval Research (ONR N00014-12-1-0997).

Appendix A. Proof of Lemma 4

According to the definition of $\hat{\mathbf{g}}^t$ which is the concatenation of the local stochastic averaging gradients $\hat{\mathbf{g}}_n^t$ and the fact that the expected value of sum is equal to the sum of expected values, we can write the expected value $\mathbb{E}\left[\|\hat{\mathbf{g}}^t - \nabla f(\mathbf{x}^*)\|^2 \mid \mathcal{F}^t\right]$ as

$$\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t - \nabla f(\mathbf{x}^*)\right\|^2 \mid \mathcal{F}^t\right] = \sum_{n=1}^{N} \mathbb{E}\left[\left\|\hat{\mathbf{g}}_n^t - \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \mid \mathcal{F}^t\right]. \tag{54}$$

We proceed by finding upper bounds for the summands of (54). Observe that using the standard variance decomposition for any random variable vector $\mathbf{a}$ we can write $\mathbb{E}\left[\|\mathbf{a}\|^2\right] = \|\mathbb{E}[\mathbf{a}]\|^2 + \mathbb{E}\left[\|\mathbf{a} - \mathbb{E}[\mathbf{a}]\|^2\right]$. Notice that the same relation holds true when the expectations are computed with respect to a specific field $\mathcal{F}$. By setting $\mathbf{a} = \hat{\mathbf{g}}_n^t - \nabla f_n(\tilde{\mathbf{x}}^*)$ and considering that $\mathbb{E}[\mathbf{a} \mid \mathcal{F}^t] = \nabla f_n(\mathbf{x}_n^t) - \nabla f_n(\tilde{\mathbf{x}}^*)$, the variance decomposition implies

$$\begin{aligned}\mathbb{E}\left[\left\|\hat{\mathbf{g}}_n^t - \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \mid \mathcal{F}^t\right] &= \left\|\nabla f_n(\mathbf{x}_n^t) - \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \\ &\quad + \mathbb{E}\left[\left\|\hat{\mathbf{g}}_n^t - \nabla f_n(\tilde{\mathbf{x}}^*) - \nabla f_n(\mathbf{x}_n^t) + \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \mid \mathcal{F}^t\right]. \end{aligned} \tag{55}$$

The next step is to find an upper bound for the last term in (55). Adding and subtracting $\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)$ and using the inequality $\|\mathbf{a}+\mathbf{b}\|^2 \leq 2\|\mathbf{a}\|^2 + 2\|\mathbf{b}\|^2$ for $\mathbf{a} = \nabla f_{n,i_n^t}(\mathbf{x}_n^t) - \nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*) - \nabla f_n(\mathbf{x}_n^t) + \nabla f_n(\tilde{\mathbf{x}}^*)$ and $\mathbf{b} = -(\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t) - \nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*) - (1/q_n)\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{y}_{n,i}^t) + \nabla f_n(\tilde{\mathbf{x}}^*))$ lead to

$$\begin{aligned}&\mathbb{E}\left[\left\|\hat{\mathbf{g}}_n^t - \nabla f_n(\tilde{\mathbf{x}}^*) - \nabla f_n(\mathbf{x}_n^t) + \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \mid \mathcal{F}^t\right] \\ &\leq 2\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t) - \nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*) - \nabla f_n(\mathbf{x}_n^t) + \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \mid \mathcal{F}^t\right] \\ &\quad + 2\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t) - \nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*) - \frac{1}{q_n}\sum_{i=1}^{q_n} \nabla f_{n,i}(\mathbf{y}_{n,i}^t) + \nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2 \mid \mathcal{F}^t\right]. \end{aligned} \tag{56}$$

In this step we use the standard variance decomposition twice to simplify the two expectations in the right hand side of (56). Based on the standard variance decomposition $\mathbb{E}\left[\|\mathbf{a} - \mathbb{E}[\mathbf{a}]\|^2\right] =$

$\mathbb{E}\left[\|\mathbf{a}\|^2\right]-\|\mathbb{E}\left[\mathbf{a}\right]\|^2$ we obtain $\mathbb{E}\left[\|\mathbf{a}-\mathbb{E}\left[\mathbf{a}\right]\|^2\right]\leq\mathbb{E}\left[\|\mathbf{a}\|^2\right]$. Therefore, by setting $\mathbf{y}=\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)$ and observing that the expected value $\mathbb{E}\left[\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\mid\mathcal{F}^t\right]$ is equal to $(1/q_n)\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)-\nabla f_n(\tilde{\mathbf{x}}^*)$ we obtain that

$$\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)-\frac{1}{q_n}\sum_{i=1}^{q_n}\nabla f_{n,i}(\mathbf{y}_{n,i}^t)+\nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]$$
$$\leq\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right].\tag{57}$$

Moreover, by choosing $\mathbf{a}=\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)$ and noticing the relation for the expected value which is $\mathbb{E}\left[\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\mid\mathcal{F}^t\right]=\nabla f_n(\mathbf{x}_n^t)-\nabla f_n(\tilde{\mathbf{x}}^*)$, the equality $\mathbb{E}\left[\|\mathbf{a}-\mathbb{E}\left[\mathbf{a}\right]\|^2\right]=\mathbb{E}\left[\|\mathbf{a}\|^2\right]-\|\mathbb{E}\left[\mathbf{a}\right]\|^2$ yields

$$\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)-\nabla f_n(\mathbf{x}_n^t)+\nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]$$
$$=\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]-\left\|\nabla f_n(\mathbf{x}_n^t)-\nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2.\tag{58}$$

By substituting the upper bound in (57) and the simplification in (58) into (56), and considering the expression in (55) we obtain that

$$\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]\leq 2\sum_{n=1}^{N}\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]-\sum_{n=1}^{N}\left\|\nabla f_n(\mathbf{x}_n^t)-\nabla f_n(\tilde{\mathbf{x}}^*)\right\|^2$$
$$+2\sum_{n=1}^{N}\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right].\tag{59}$$

We proceed by finding an upper bound for the first sum in the right hand side of (59). Notice that if the gradients of the function g are Lipschitz continuous with parameter L, then for any two vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ we can write $g(\mathbf{a}_1)\geq g(\mathbf{a}_2)+\nabla g(\mathbf{a}_2)^T(\mathbf{a}_1-\mathbf{a}_2)+(1/2L)\|\nabla g(\mathbf{a}_1)-\nabla g(\mathbf{a}_2)\|^2$. According to the Lipschitz continuity of the instantaneous local functions gradient $\nabla f_{n,i}(\mathbf{x}_n)$, we can write the inequality for $g=f_{n,i}$, $\mathbf{a}_1=\mathbf{y}_{n,i}^t$ and $\mathbf{a}_2=\tilde{\mathbf{x}}^*$ which is equivalent to

$$\frac{1}{2L}\left\|\nabla f_{n,i}(\mathbf{y}_{n,i}^t)-\nabla f_{n,i}(\tilde{\mathbf{x}}^*)\right\|^2\leq f_{n,i}(\mathbf{y}_{n,i}^t)-f_{n,i}(\tilde{\mathbf{x}}^*)-\nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^t-\tilde{\mathbf{x}}^*).\tag{60}$$

Summing up both sides of (60) for all $i=1,\ldots,q_n$ and dividing both sides of the implied inequality by q_n yield

$$\frac{1}{q_n}\sum_{i=1}^{q_n}\left\|\nabla f_{n,i}(\mathbf{y}_{n,i}^t)-\nabla f_{n,i}(\tilde{\mathbf{x}}^*)\right\|^2\leq 2L\left[\frac{1}{q_n}\sum_{i=1}^{q_n}f_{n,i}(\mathbf{y}_{n,i}^t)-f_{n,i}(\tilde{\mathbf{x}}^*)-\nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^t-\tilde{\mathbf{x}}^*)\right].\tag{61}$$

Since the random functions f_{n,i_n^t} has a uniform distribution over the set $\{f_{n,1},\ldots,f_{n,q_n}\}$, we can substitute the left hand side of (61) by $\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{y}_{n,i_n^t}^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]$. Apply this substitution and sum up both sides of (61) for $n=1,\ldots,N$. According to the definition of sequence p^t in (33), if we sum up the right hand side of (61) over n it can be simplified as $2Lp^t$. Applying these simplifications we obtain

$$\sum_{n=1}^{N}\mathbb{E}\left[\left\|\nabla f_{n,\boldsymbol{\theta}_n^t}(\mathbf{y}_n^t)-\nabla f_{n,\boldsymbol{\theta}_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]\leq 2Lp^t.\tag{62}$$

Substituting the upper bound in (62) into (59) and simplifying the sum $\sum_{n=1}^{N}\|\nabla f_n(\mathbf{x}_n^t)-\nabla f_n(\tilde{\mathbf{x}}^*)\|^2$ as $\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2$ yield

$$\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]\leq 2\sum_{n=1}^{N}\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]-\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2$$
$$+4Lp^t.\tag{63}$$

To show that the sum in the right hand side of (63) is bounded above we use the Lipschitz continuity of the instantaneous functions gradients $\nabla f_{n,i}$. Using the same argument from (60) to (62) we can write

$$\sum_{n=1}^{N}\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]\tag{64}$$
$$\leq 2L\sum_{n=1}^{N}\frac{1}{q_n}\left[\sum_{i=1}^{q_n}f_{n,i}(\mathbf{x}_n^t)-f_{n,i}(\tilde{\mathbf{x}}^*)-\nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{x}_n^t-\tilde{\mathbf{x}}^*)\right].$$

Considering the definition of the local objective functions $f_n(\mathbf{x}_n)=(1/q_n)\sum_{i=1}^{q_n}f_{n,i}(\mathbf{x}_n)$ and the aggregate function $f(\mathbf{x}):=\sum_{n=1}^{N}f_n(\mathbf{x}_n)$, the right hand side of (64) can be simplified as

$$\sum_{n=1}^{N}\mathbb{E}\left[\left\|\nabla f_{n,i_n^t}(\mathbf{x}_n^t)-\nabla f_{n,i_n^t}(\tilde{\mathbf{x}}^*)\right\|^2\mid\mathcal{F}^t\right]\leq 2L\left(f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right).\tag{65}$$

Replacing the sum in (63) by the upper bound in (65) implies

$$\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]\leq 4Lp^t-\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2+4L\left(f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right).\tag{66}$$

Considering the strong convexity of the global objective function f with constant μ we can write

$$\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2\geq 2\mu\left(f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right).\tag{67}$$

Therefore, we can substitute $\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2$ in (65) by the lower bound in (67) and the claim in (34) follows.

Appendix B. Proof of Lemma 5

According to the Lipschitz continuity of the aggregate function gradients $\nabla f(\mathbf{x})$, we can write $(1/L)\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2\leq(\mathbf{x}^t-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*))$. By adding and subtracting $\mathbf{x}^{t+1}$ to the term $\mathbf{x}^t-\mathbf{x}^*$ and multiplying both sides of the inequality by 2α we obtain

$$\frac{2\alpha}{L}\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2\leq 2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*))+2\alpha(\mathbf{x}^t-\mathbf{x}^{t+1})^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)).\tag{68}$$

Expanding the difference $\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)$ as $\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)+\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t$ for the first inner product in the right hand side of (68) implies

$$\frac{2\alpha}{L}\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2\leq 2\alpha(\mathbf{x}^t-\mathbf{x}^{t+1})^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*))+2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*))$$
$$+2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t).\tag{69}$$

We proceed to simplify the inner product $2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*))$ in the right hand side of (69) by substituting $\alpha(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*))$ with its equivalent as introduced in (30). By applying this substitution the inner product $2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*))$ can be simplified as

$$2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)) = -2\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}+2(\mathbf{x}^{t+1}-\mathbf{x}^*)^T\tilde{\mathbf{Z}}(\mathbf{x}^t-\mathbf{x}^{t+1}) - 2(\mathbf{x}^{t+1}-\mathbf{x}^*)^T\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*). \tag{70}$$

Based on the KKT condition of problem (25), the optimal primal variable satisfies $(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}\mathbf{x}^*=\mathbf{0}$ which by considering the definition of the matrix $\mathbf{U}=(\tilde{\mathbf{Z}}-\mathbf{Z})^{1/2}$ we obtain that $\mathbf{U}\mathbf{x}^*=\mathbf{0}$. This observation in conjunction with the update rule of the dual variable $\mathbf{v}^t$ in (28) implies that we can substitute $\mathbf{U}(\mathbf{x}^{t+1}-\mathbf{x}^*)$ by $\mathbf{v}^{t+1}-\mathbf{v}^t$. Making this substitution into the last summand of the right hand side of (70) and considering the symmetry of the matrix $\mathbf{U}$ yield

$$2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)) = -2\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}+2(\mathbf{x}^{t+1}-\mathbf{x}^*)^T\tilde{\mathbf{Z}}(\mathbf{x}^t-\mathbf{x}^{t+1}) - 2(\mathbf{v}^{t+1}-\mathbf{v}^t)^T(\mathbf{v}^{t+1}-\mathbf{v}^*). \tag{71}$$

According to the definition of the vector $\mathbf{u}$ and matrix $\mathbf{G}$ in (35), the last two summands of (71) can be simplified as $2(\mathbf{u}^{t+1}-\mathbf{u}^t)^T\mathbf{G}(\mathbf{u}^*-\mathbf{u}^{t+1})$. Moreover, observe that the inner product $2(\mathbf{u}^{t+1}-\mathbf{u}^t)^T\mathbf{G}(\mathbf{u}^*-\mathbf{u}^{t+1})$ can be simplified as $\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}}-\|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}$. Applying this simplification into (71) implies

$$2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)) = -2\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}+\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}} - \|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}. \tag{72}$$

The next step is to find an upper bound for the inner product $2\alpha(\mathbf{x}^t-\mathbf{x}^{t+1})^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*))$. Note that for any two vectors $\mathbf{a}$ and $\mathbf{b}$, and any positive scalar η the inequality $2\mathbf{a}^T\mathbf{b}\leq\eta\|\mathbf{a}\|^2+\eta^{-1}\|\mathbf{b}\|^2$ holds. Thus, by setting $\mathbf{a}=\mathbf{x}^t-\mathbf{x}^{t+1}$ and $\mathbf{b}=\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)$ we obtain that

$$2\alpha(\mathbf{x}^t-\mathbf{x}^{t+1})^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*))\leq\frac{\alpha}{\eta}\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2+\alpha\eta\|\mathbf{x}^t-\mathbf{x}^{t+1}\|^2. \tag{73}$$

Now we substitute the terms in the right hand side of (69) by their simplifications or upper bounds. Replacing the inner product $2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*))$ by the simplification in (72), substituting expression $2\alpha(\mathbf{x}^t-\mathbf{x}^{t+1})^T(\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*))$ by the upper bound in (73), and substituting inner product $2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)$ by the sum $2\alpha(\mathbf{x}^t-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)+2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^t)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)$ imply

$$\frac{2\alpha}{L}\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2\leq -2\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}+\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}} - \|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}+\alpha\eta\|\mathbf{x}^t-\mathbf{x}^{t+1}\|^2+\frac{\alpha}{\eta}\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2 + 2\alpha(\mathbf{x}^t-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)+2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^t)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t). \tag{74}$$

Considering that $\mathbf{x}^t-\mathbf{x}^*$ is deterministic given observations until step t and observing the relation $\mathbb{E}\left[\hat{\mathbf{g}}^t\mid\mathcal{F}^t\right]=\nabla f(\mathbf{x}^t)$, we obtain that $\mathbb{E}\left[(\mathbf{x}^t-\mathbf{x}^*)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)\mid\mathcal{F}^t\right]=0$. Therefore, by computing the expected value of both sides of (74) given the observations until step t and regrouping the terms we obtain

$$\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right]\geq\alpha\left(\frac{2}{L}-\frac{1}{\eta}\right)\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2+\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right] + 2\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}\mid\mathcal{F}^t\right]-\alpha\eta\mathbb{E}\left[\|\mathbf{x}^t-\mathbf{x}^{t+1}\|^2\mid\mathcal{F}^t\right] - \mathbb{E}\left[2\alpha(\mathbf{x}^{t+1}-\mathbf{x}^t)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)\mid\mathcal{F}^t\right]. \tag{75}$$

By applying inequality $2\mathbf{a}^T\mathbf{b}\leq\eta\|\mathbf{a}\|^2+\eta^{-1}\|\mathbf{b}\|^2$ for the vectors $\mathbf{a}=\mathbf{x}^{t+1}-\mathbf{x}^t$ and $\mathbf{b}=\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t$, we obtain that the inner product $2(\mathbf{x}^{t+1}-\mathbf{x}^t)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)$ is bounded above by $\eta\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2+(1/\eta)\|\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t\|^2$. Replacing $2(\mathbf{x}^{t+1}-\mathbf{x}^t)^T(\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t)$ in (75) by its upper bound $\eta\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2+(1/\eta)\|\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t\|^2$ yields

$$\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right]\geq\alpha\left(\frac{2}{L}-\frac{1}{\eta}\right)\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2+\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right] + 2\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}\mid\mathcal{F}^t\right]-2\alpha\eta\mathbb{E}\left[\|\mathbf{x}^t-\mathbf{x}^{t+1}\|^2\mid\mathcal{F}^t\right] - \frac{\alpha}{\eta}\mathbb{E}\left[\|\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t\|^2\mid\mathcal{F}^t\right]. \tag{76}$$

Observe that the squared norm $\|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}$ can be expanded as $\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2_{\tilde{\mathbf{Z}}}+\|\mathbf{v}^{t+1}-\mathbf{v}^t\|^2$. Using this simplification for $\|\mathbf{u}^{t+1}-\mathbf{u}^t\|^2_{\mathbf{G}}$ and regrouping the terms in (76) lead to

$$\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right]\geq\alpha\left(\frac{2}{L}-\frac{1}{\eta}\right)\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2 + \mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2_{\mathbf{Z}-2\alpha\eta\mathbf{I}}\mid\mathcal{F}^t\right]+\mathbb{E}\left[\|\mathbf{v}^{t+1}-\mathbf{v}^t\|^2\mid\mathcal{F}^t\right] + 2\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}\big|\mathcal{F}^t\right]-\frac{\alpha}{\eta}\mathbb{E}\left[\|\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t\|^2\big|\mathcal{F}^t\right]. \tag{77}$$

We proceed by simplifying the expectation $\mathbb{E}\left[\|\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t\|^2\big|\mathcal{F}^t\right]$ in (77). Note that by adding and subtracting $\nabla f(\mathbf{x}^*)$, the expectation can be written as $\mathbb{E}\left[\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)+\nabla f(\mathbf{x}^*)-\hat{\mathbf{g}}^t\|^2\mid\mathcal{F}^t\right]$ and by expanding the squared norm and simplifying the terms we obtain

$$\mathbb{E}\left[\left\|\nabla f(\mathbf{x}^t)-\hat{\mathbf{g}}^t\right\|^2\mid\mathcal{F}^t\right]=\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]-\mathbb{E}\left[\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]. \tag{78}$$

Substituting the simplification in (78) into (77) yields

$$\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right]\geq\frac{2\alpha}{L}\left\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\right\|^2 + \mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2_{\mathbf{Z}-2\alpha\eta\mathbf{I}}\mid\mathcal{F}^t\right]+\mathbb{E}\left[\|\mathbf{v}^{t+1}-\mathbf{v}^t\|^2\mid\mathcal{F}^t\right] + 2\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}\big|\mathcal{F}^t\right]-\frac{\alpha}{\eta}\mathbb{E}\left[\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\|^2\mid\mathcal{F}^t\right]. \tag{79}$$

Considering the strong convexity of the global objective function f with constant μ we can write $\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2\geq 2\mu\left(f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right)$. Substituting the squared norm $\|\nabla f(\mathbf{x}^t)-\nabla f(\mathbf{x}^*)\|^2$ by this lower bound in (79) follows

$$\|\mathbf{u}^t-\mathbf{u}^*\|^2_{\mathbf{G}}-\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|^2_{\mathbf{G}}\mid\mathcal{F}^t\right]\geq\frac{4\alpha\mu}{L}\left(f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right) + \mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2_{\mathbf{Z}-2\alpha\eta\mathbf{I}}\mid\mathcal{F}^t\right]+\mathbb{E}\left[\|\mathbf{v}^{t+1}-\mathbf{v}^t\|^2\mid\mathcal{F}^t\right] + 2\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}\big|\mathcal{F}^t\right]-\frac{\alpha}{\eta}\mathbb{E}\left[\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\|^2\mid\mathcal{F}^t\right]. \tag{80}$$

Substituting the upper bound for the expectation $\mathbb{E}\left[\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\|^2\mid\mathcal{F}^t\right]$ in (34) into (80) and regrouping the terms show the validity of the claim in (36).

Appendix C. Proof of Lemma 6

Given the information until time t, each auxiliary vector $\mathbf{y}^{t+1}_{n,i}$ is a random variable that takes values $\mathbf{y}^t_{n,i}$ and $\mathbf{x}^t_n$ with associated probabilities $1-1/q_n$ and $1/q_n$, respectively. This observation holds

since with probability $1/q_n$ node n may choose index i to update at time $t+1$ and with probability $1-(1/q_n)$ choose other indices. Therefore, we can write

$$\mathbb{E}\left[\frac{1}{q_n}\sum_{i=1}^{q_n}\left(\nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^{t+1}-\tilde{\mathbf{x}}^*)\right)\mid\mathcal{F}^t\right]=\left[1-\frac{1}{q_n}\right]\frac{1}{q_n}\sum_{i=1}^{q_n}\nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^t-\tilde{\mathbf{x}}^*)+\frac{1}{q_n}\nabla f_n(\tilde{\mathbf{x}}^*)^T(\mathbf{x}_n^t-\tilde{\mathbf{x}}^*). \tag{81}$$

Likewise, the distribution of random function $f_{n,i}(\mathbf{y}_{n,i}^{t+1})$ given observation until time t has two possibilities $f_{n,i}(\mathbf{y}_{n,i}^t)$ and $f_{n,i}(\mathbf{x}_n^t)$ with associated probabilities $1-1/q_n$ and $1/q_n$, respectively. Hence, we can write $\mathbb{E}\left[f_{n,i}(\mathbf{y}_{n,i}^{t+1})\mid\mathcal{F}^t\right]=(1-1/q_n)f_{n,i}(\mathbf{y}_{n,i}^t)+(1/q_n)f_{n,i}(\mathbf{x}_n^t)$. By summing this relation for all $i\in 1,\ldots,q_n$ and divining both sides of the resulted expression by q_n we obtain

$$\mathbb{E}\left[\frac{1}{q_n}\sum_{i=1}^{q_n}f_{n,i}(\mathbf{y}_{n,i}^{t+1})\mid\mathcal{F}^t\right]=\left[1-\frac{1}{q_n}\right]\frac{1}{q_n}\sum_{i=1}^{q_n}f_{n,i}(\mathbf{y}_{n,i}^t)+\frac{1}{q_n}f_n(\mathbf{x}_n^t). \tag{82}$$

To simplicity equations let us define the sequence p_n^t as

$$p_n^t:=\frac{1}{q_n}\sum_{i=1}^{q_n}f_{n,i}(\mathbf{y}_{n,i}^t)-f_n(\tilde{\mathbf{x}}^*)-\frac{1}{q_n}\sum_{i=1}^{q_n}\nabla f_{n,i}(\tilde{\mathbf{x}}^*)^T(\mathbf{y}_{n,i}^t-\tilde{\mathbf{x}}^*). \tag{83}$$

Subtracting (81) from (82) and adding $-f_n(\tilde{\mathbf{x}}^*)$ to the both sides of equality in association with the definition of the sequence p_n^t in (83) yield

$$\mathbb{E}\left[p_n^{t+1}\mid\mathcal{F}^t\right]=\left[1-\frac{1}{q_n}\right]p_n^t+\frac{1}{q_n}\left[f_n(\mathbf{x}_n^t)-f_n(\tilde{\mathbf{x}}^*)-\nabla f_n(\tilde{\mathbf{x}}^*)^T(\mathbf{x}_n^t-\tilde{\mathbf{x}}^*)\right]. \tag{84}$$

We proceed to find and upper bound for the terms in the right hand side of (84). First note that according to the strong convexity of the local instantaneous functions $f_{n,i}$ and local functions f_n both terms in the right hand side of (84) are non-negative. Observing that the number of instantaneous functions at each node q_n satisfies the condition $q_{\min}\leq q_n\leq q_{\max}$, we obtain

$$1-\frac{1}{q_n}\leq 1-\frac{1}{q_{\max}}\ ,\qquad \frac{1}{q_n}\leq\frac{1}{q_{\min}}. \tag{85}$$

Substituting the upper bounds in (85) into (84), summing both sides of the implied inequality over $n\in\{1,\ldots,N\}$, and considering the definitions of the optimal argument $\mathbf{x}^*=[\tilde{\mathbf{x}}^*;\ldots;\tilde{\mathbf{x}}^*]$ and the aggregate function $f(\mathbf{x})=\sum_{n=1}^N f_n(\mathbf{x}_n)$ lead to

$$\sum_{n=1}^N\mathbb{E}\left[p_n^{t+1}\mid\mathcal{F}^t\right]\leq\left[1-\frac{1}{q_{\max}}\right]\sum_{n=1}^N p_n^t+\frac{1}{q_{\min}}\left[f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right]. \tag{86}$$

Now observe that according to the definitions of the sequences p^t and p_n^t in (33) and (83), respectively, p^t is the sum of p_n^t for all n, i.e. $p^t=\sum_{n=1}^N p_n^t$. Hence, we can rewrite (86) as

$$\mathbb{E}\left[p^{t+1}\mid\mathcal{F}^t\right]\leq\left[1-\frac{1}{q_{\max}}\right]p^t+\frac{1}{q_{\min}}\left[f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right]. \tag{87}$$

Therefore, the claim in (37) is valid.

Appendix D. Proof of Theorem 7

To prove the result in Theorem 7 first we prove the following Lemma to establish an upper bound for the squared error norm $\|\mathbf{v}^t-\mathbf{v}^*\|^2$.

Lemma 11 *Consider the DSA algorithm as defined in* (6)-(9). *Further, recall γ' as the smallest non-zero eigenvalue and Γ' as the largest eigenvalue of the matrix $\tilde{\mathbf{Z}}-\mathbf{Z}$. If Assumptions 1-3 hold, then the squared norm of the difference $\|\mathbf{v}^t-\mathbf{v}^*\|^2$ is bounded above as*

$$\|\mathbf{v}^t-\mathbf{v}^*\|^2\leq\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^{t+1}-\mathbf{x}^*\right\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^2}\mid\mathcal{F}^t\right]+\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^t-\mathbf{x}^{t+1}\right\|^2_{\tilde{\mathbf{Z}}^2}\mid\mathcal{F}^t\right]+\frac{16\alpha^2L}{\gamma'}\ p^t+\frac{2\Gamma'}{\gamma'}\mathbb{E}\left[\|\mathbf{v}^t-\mathbf{v}^{t+1}\|^2\mid\mathcal{F}^t\right]+\frac{8\alpha^2\left(2L-\mu\right)}{\gamma'}\left[f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right]. \tag{88}$$

Proof Consider the inequality $\|\mathbf{a}+\mathbf{b}\|^2\leq 2\|\mathbf{a}\|^2+2\|\mathbf{b}\|^2$ for the case that $\mathbf{a}=\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)$, $\mathbf{b}=\mathbf{U}(\mathbf{v}^t-\mathbf{v}^{t+1})$ which can be written as

$$\|\mathbf{U}(\mathbf{v}^t-\mathbf{v}^*)\|^2\leq 2\|\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)\|^2+2\|\mathbf{U}(\mathbf{v}^t-\mathbf{v}^{t+1})\|^2. \tag{89}$$

We proceed by finding an upper bound for $2\|\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)\|^2$. Based on the result of Lemma 3 in (30), the term $\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)$ is equal to the sum of vectors $\mathbf{a}+\mathbf{b}$ where $\mathbf{a}=(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})(\mathbf{x}^{t+1}-\mathbf{x}^*)-\tilde{\mathbf{Z}}(\mathbf{x}^t-\mathbf{x}^{t+1})$ and $\mathbf{b}=-\alpha\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)$. Therefore, using the inequality $\|\mathbf{a}+\mathbf{b}\|^2\leq 2\|\mathbf{a}\|^2+2\|\mathbf{b}\|^2$ we can write

$$\left\|\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)\right\|^2\leq 2\left\|(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})(\mathbf{x}^{t+1}-\mathbf{x}^*)-\tilde{\mathbf{Z}}(\mathbf{x}^t-\mathbf{x}^{t+1})\right\|^2+2\alpha^2\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2. \tag{90}$$

By using the inequality $\|\mathbf{a}+\mathbf{b}\|^2\leq 2\|\mathbf{a}\|^2+2\|\mathbf{b}\|^2$ one more time for vectors $\mathbf{a}=(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})(\mathbf{x}^{t+1}-\mathbf{x}^*)$ and $\mathbf{b}=-\tilde{\mathbf{Z}}(\mathbf{x}^t-\mathbf{x}^{t+1})$, we obtain an upper bound for the term $\|(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})(\mathbf{x}^{t+1}-\mathbf{x}^*)-\tilde{\mathbf{Z}}(\mathbf{x}^t-\mathbf{x}^{t+1})\|^2$. Substituting this upper bound into (90) yields

$$\left\|\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)\right\|^2\leq 4\left\|\mathbf{x}^{t+1}-\mathbf{x}^*\right\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^2}+4\left\|\mathbf{x}^t-\mathbf{x}^{t+1}\right\|^2_{\tilde{\mathbf{Z}}^2}+2\alpha^2\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2. \tag{91}$$

Inequality (91) shows an upper bound for $2\|\mathbf{U}(\mathbf{v}^{t+1}-\mathbf{v}^*)\|^2$ in (89). Moreover, we know that the second term $\|\mathbf{U}(\mathbf{v}^t-\mathbf{v}^{t+1})\|^2$ is also bounded above by $\Gamma'\|\mathbf{v}^t-\mathbf{v}^{t+1}\|^2$ where Γ' is the largest eigenvalue of matrix $\tilde{\mathbf{Z}}-\mathbf{Z}=\mathbf{U}^2$. Substituting these upper bounds into (89) and computing the expected value of both sides given the information until step t yield

$$\|\mathbf{U}(\mathbf{v}^t-\mathbf{v}^*)\|^2\leq\ 8\mathbb{E}\left[\left\|\mathbf{x}^{t+1}-\mathbf{x}^*\right\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^2}\mid\mathcal{F}^t\right]+8\mathbb{E}\left[\left\|\mathbf{x}^t-\mathbf{x}^{t+1}\right\|^2_{\tilde{\mathbf{Z}}^2}\mid\mathcal{F}^t\right]+4\alpha^2\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]+2\Gamma'\mathbb{E}\left[\|\mathbf{v}^t-\mathbf{v}^{t+1}\|^2\mid\mathcal{F}^t\right]. \tag{92}$$

Note the vectors $\mathbf{v}^t$ and $\mathbf{v}^*$ lie in the column space of the matrix $\mathbf{U}$. Thus, we obtain that $\|\mathbf{U}(\mathbf{v}^t-\mathbf{v}^*)\|^2\geq\gamma'\|\mathbf{v}^t-\mathbf{v}^*\|^2$. Substituting this lower bound for $\|\mathbf{U}(\mathbf{v}^t-\mathbf{v}^*)\|^2$ in (92) and deviding both sides of the imposed inequality by γ' yield

$$\|\mathbf{v}^t-\mathbf{v}^*\|^2\leq\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^{t+1}-\mathbf{x}^*\right\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^2}\mid\mathcal{F}^t\right]+\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^t-\mathbf{x}^{t+1}\right\|^2_{\tilde{\mathbf{Z}}^2}\mid\mathcal{F}^t\right]+\frac{4\alpha^2}{\gamma'}\mathbb{E}\left[\left\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\right\|^2\mid\mathcal{F}^t\right]+\frac{2\Gamma'}{\gamma'}\mathbb{E}\left[\|\mathbf{v}^t-\mathbf{v}^{t+1}\|^2\mid\mathcal{F}^t\right]. \tag{93}$$

By substituting the expectation $\mathbb{E}\left[\|\hat{\mathbf{g}}^t-\nabla f(\mathbf{x}^*)\|^2\mid\mathcal{F}^t\right]$ in the right hand side of (93) with its upper bound in (34), the claim in (88) follows. ■

Using the result in Lemma 11 we show that the sequence $\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+c\,p^t$ converges linearly to zero.

Proof of Theorem 7: Proving the linear convergence claim in (40) is equivalent to showing that

$$\delta\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+\delta c\,p^t\leq\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2-\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|_{\mathbf{G}}^2\mid\mathcal{F}^t\right]+c\,(p^t-\mathbb{E}\left[p^{t+1}\mid\mathcal{F}^t\right]). \tag{94}$$

Substituting the terms $\mathbb{E}\left[\|\mathbf{u}^{t+1}-\mathbf{u}^*\|_{\mathbf{G}}^2\mid\mathcal{F}^t\right]$ and $\mathbb{E}\left[p^{t+1}\mid\mathcal{F}^t\right]$ by their upper bounds as introduced in Lemma 5 and Lemma 6, respectively, yields a sufficient condition for the claim in (94) as

$$\begin{aligned}\delta\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+\delta c\,p^t\leq{}&\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2_{\tilde{\mathbf{Z}}-2\alpha\eta\mathbf{I}}\mid\mathcal{F}^t\right]+\mathbb{E}\left[\|\mathbf{v}^{t+1}-\mathbf{v}^t\|^2\mid\mathcal{F}^t\right]\\&+2\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}}}\mid\mathcal{F}^t\right]+\left(\frac{c}{q_{\max}}-\frac{4\alpha L}{\eta}\right)p^t\\&+\left[\frac{4\alpha\mu}{L}-\frac{2\alpha(2L-\mu)}{\eta}-\frac{c}{q_{\min}}\right]\left[f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right].\end{aligned} \tag{95}$$

We emphasize that if the inequality in (95) holds, then the inequalities in (94) and (40) are valid. Note that the weighted norm $\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2$ in the left hand side of (95) can be simplified as $\|\mathbf{x}^t-\mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2+\|\mathbf{v}^t-\mathbf{v}^*\|^2$. Considering the definition of Γ as the maximum eigenvalue of the matrix $\tilde{\mathbf{Z}}$, we can conclude that $\|\mathbf{x}^t-\mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2$ is bounded above by $\Gamma\|\mathbf{x}^t-\mathbf{x}^*\|^2$. Considering this relation and observing the upper bound for $\|\mathbf{v}^t-\mathbf{v}^*\|^2$ in (88), we obtain that $\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2=\|\mathbf{x}^t-\mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2+\|\mathbf{v}^t-\mathbf{v}^*\|^2$ is bounded above by

$$\begin{aligned}\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2\;\leq\;&\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^{t+1}-\mathbf{x}^*\right\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^2}\mid\mathcal{F}^t\right]+\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^t-\mathbf{x}^{t+1}\right\|^2_{\tilde{\mathbf{Z}}^2}\mid\mathcal{F}^t\right]+\frac{16\alpha^2L}{\gamma'}\,p^t\\&+\frac{2\Gamma'}{\gamma'}\mathbb{E}\left[\|\mathbf{v}^t-\mathbf{v}^{t+1}\|^2\mid\mathcal{F}^t\right]+\Gamma\|\mathbf{x}^t-\mathbf{x}^*\|^2\\&+\frac{8\alpha^2\,(2L-\mu)}{\gamma'}\left[f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right].\end{aligned} \tag{96}$$

Further, the strong convexity of the global objective function f implies that the squared norm $\|\mathbf{x}^t-\mathbf{x}^*\|^2$ is upper bound by $(2/\mu)(f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*))$. Replacing the the squared norm $\|\mathbf{x}^t-\mathbf{x}^*\|^2$ in (96) by its upper bound leads to

$$\begin{aligned}\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2\leq{}&\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^{t+1}-\mathbf{x}^*\right\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^2}\mid\mathcal{F}^t\right]+\frac{8}{\gamma'}\mathbb{E}\left[\left\|\mathbf{x}^t-\mathbf{x}^{t+1}\right\|^2_{\tilde{\mathbf{Z}}^2}\mid\mathcal{F}^t\right]+\frac{16\alpha^2L}{\gamma'}\,p^t\\&+\frac{2\Gamma'}{\gamma'}\mathbb{E}\left[\|\mathbf{v}^t-\mathbf{v}^{t+1}\|^2\mid\mathcal{F}^t\right]\\&+\left(\frac{8\alpha^2\,(2L-\mu)}{\gamma'}+\frac{2\Gamma}{\mu}\right)\left[f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)\right].\end{aligned} \tag{97}$$

Replacing $\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2$ in (95) by the upper bound in (97) and regrouping the terms lead to

$$\begin{aligned}0\leq{}&\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^t\|^2_{\tilde{\mathbf{Z}}-\alpha(\eta+\eta)\mathbf{I}-\frac{8\delta}{\gamma'}\tilde{\mathbf{Z}}^2}\mid\mathcal{F}^t\right]+\mathbb{E}\left[\|\mathbf{x}^{t+1}-\mathbf{x}^*\|^2_{(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^{\frac{1}{2}}\left[2\mathbf{I}-\frac{8\delta}{\gamma'}(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})\right](\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})^{\frac{1}{2}}}\mid\mathcal{F}^t\right]\\&+\mathbb{E}\left[\|\mathbf{v}^{t+1}-\mathbf{v}^t\|^2_{(1-\frac{2\delta\Gamma'}{\gamma'})\mathbf{I}}\mid\mathcal{F}^t\right]+\left[\frac{c}{q_{\max}}-\frac{4\alpha L}{\eta}-\delta c-\frac{16\delta\alpha^2L}{\gamma'}\right]p^t\\&+\left[\frac{4\alpha\mu}{L}-\frac{2\alpha(2L-\mu)}{\eta}-\frac{c}{q_{\min}}-\frac{8\delta\alpha^2\,(2L-\mu)}{\gamma'}-\frac{2\delta\Gamma}{\mu}\right](f(\mathbf{x}^t)-f(\mathbf{x}^*)-\nabla f(\mathbf{x}^*)^T(\mathbf{x}^t-\mathbf{x}^*)).\end{aligned} \tag{98}$$

Notice that if the inequality in (98) holds true, then the relation in (95) is valid and as we mentioned before the claim in (94) holds. To verify the sum in the right hand side of (98) is always positive and the inequality is valid, we enforce each summands in the right hand side of (98) to be non-negative. Therefore, the following conditions should be satisfied

$$\begin{aligned}&\gamma-\alpha(\eta+\eta)-\frac{8\delta}{\gamma'}\Gamma^2\geq0,\quad 2-\frac{8\delta}{\gamma'}\lambda_{max}(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})\geq0,\quad 1-\frac{2\delta\Gamma'}{\gamma'}\geq0,\\&\frac{c}{q_{\max}}-\frac{4\alpha L}{\eta}-\delta c-\frac{16\delta\alpha^2L}{\gamma'}\geq0,\quad \frac{4\alpha\mu}{L}-\frac{2\alpha(2L-\mu)}{\eta}-\frac{c}{q_{\min}}-\frac{8\delta\alpha^2\,(2L-\mu)}{\gamma'}-\frac{2\delta\Gamma}{\mu}\geq0.\end{aligned} \tag{99}$$

Recall that γ is the smallest eigenvalue of the positive definite matrix $\mathbf{Z}$. All the inequalities in (99) are satisfied, if δ is chosen as

$$\begin{aligned}\delta=\min\Bigg\{&\frac{(\gamma-2\alpha\eta)\gamma'}{8\Gamma^2},\frac{\gamma'}{4\lambda_{max}(\mathbf{I}+\mathbf{Z}-2\tilde{\mathbf{Z}})},\frac{\gamma'}{2\Gamma'},\frac{\gamma'(c\eta-4\alpha Lq_{\max})}{\eta q_{\max}(c\gamma'+16\alpha^2L)},\\&\left[\frac{4\alpha\mu}{L}-\frac{2\alpha(2L-\mu)}{\eta}-\frac{c}{q_{\min}}\right]\left[\frac{8\alpha^2\,(2L-\mu)}{\gamma'}+\frac{2\Gamma}{\mu}\right]^{-1}\Bigg\}.\end{aligned} \tag{100}$$

where η, c and α are selected from the intervals

$$\eta\in\left(\frac{L^2q_{\max}}{\mu q_{\min}}+\frac{L^2}{\mu}-\frac{L}{2}\;,\;\infty\right),\;\alpha\in\left(0\;,\;\frac{\gamma}{2\eta}\right),\;c\in\left(\frac{4\alpha Lq_{\max}}{\eta}\;,\;\frac{4\alpha\mu q_{\min}}{L}-\frac{2\alpha q_{\min}(2L-\mu)}{\eta}\right). \tag{101}$$

Notice that considering the conditions for the variables η, α and c in (101), the constant δ in (100) is strictly positive $\delta>0$. Moreover, according to the definition in (100) the constant δ is smaller than $\gamma'/2\Gamma'$ which leads to the conclusion that $\delta\leq1/2<1$. Therefore, we obtain that $0<\delta<1$ and the claim in (40) is valid.

Appendix E. Proof of Theorem 9

The proof uses the relationship in the statement (40) of Theorem 7 to build a supermartingale sequence. To do this define the stochastic processes ζ^t and β^t as

$$\zeta^t:=\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+c\,p^t,\quad \beta^t:=\delta\left(\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+c\,p^t\right). \tag{102}$$

The stochastic processes ζ^t and β^t are alway non-negative. Let now $\mathcal{F}_t$ be a sigma-algebra measuring ζ^t, β^t, and $\mathbf{u}^t$. Considering the definitions of ζ^t and β^t and the relation in (40) we can write

$$\mathbb{E}\left[\zeta^{t+1}\mid\mathcal{F}^t\right]\leq\zeta^t-\beta^t. \tag{103}$$

Since the sequences α^t and β^t are nonnegative it follows from (103) that they satisfy the conditions of the supermartingale convergence theorem – see e.g. theorem E7.4 Solo and Kong (1995) . Therefore, we obtain that: (i) The sequence ζ^t converges almost surely. (ii) The sum $\sum_{t=0}^{\infty}\beta^t<\infty$ is almost surely finite. The definition of β^t in (102) implies that

$$\sum_{t=0}^{\infty}\delta\left(\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+c\,p^t\right)<\infty,\qquad\text{a.s.} \tag{104}$$

Since $\|\mathbf{x}^t-\mathbf{x}^*\|_{\tilde{\mathbf{Z}}}^2\leq\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+c\,p^t$ and the eigenvalues of $\tilde{\mathbf{Z}}$ are lower bounded by γ we can write $\gamma\|\mathbf{x}^t-\mathbf{x}^*\|^2\leq\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{G}}^2+c\,p^t$. This inequality in association with the fact that the sum in (104) is finite leads to

$$\sum_{t=0}^{\infty}\delta\,\gamma\,\|\mathbf{x}^t-\mathbf{x}^*\|^2<\infty,\qquad\text{a.s.} \tag{105}$$

Observing the fact that δ and γ are positive constants, we can conclude from (105) that the sequence $\|\mathbf{x}^t - \mathbf{x}^*\|^2$ is almost surely summable and it converges with probability 1 to 0.

References

Ron Bekkerman, Mikhail Bilenko, and John Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Francesco Bullo, Jorge Cortés, and Sonia Martinez. *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *Industrial Informatics, IEEE Transactions on*, 9 (1):427–438, 2013.

Volkan Cevher, Steffen Becker, and Martin Schmidt. Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics. *Signal Processing Magazine, IEEE*, 31(5):32–43, 2014.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014.

John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: convergence analysis and network scaling. *Automatic control, IEEE Transactions on*, 57(3): 592–606, 2012.

Franck Iutzeler, Pascal Bianchi, Philippe Ciblat, and Walid Hachem. Explicit convergence rate of a distributed alternating direction method of multipliers. *arXiv preprint arXiv:1312.1085*, 2013.

Dusan Jakovetic, Joao Xavier, and Jose MF Moura. Fast distributed gradient methods. *Automatic Control, IEEE Transactions on*, 59(5):1131–1146, 2014.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

Usman A Khan, Soummya Kar, and JoséM F Moura. Diland: An algorithm for distributed sensor localization with noisy distance measurements. *Signal Processing, IEEE Transactions on*, 58(3): 1940–1947, 2010.

Jakub Konečnỳ and Peter Richtárik. Semi-stochastic gradient descent methods. *arXiv preprint arXiv:1312.1666*, 2013.

Qing Ling and Alejandro Ribeiro. Decentralized linearized alternating direction method of multipliers. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 5447–5451. IEEE, 2014.

Qing Ling, Wei Shi, Gang Wu, and Alejandro Ribeiro. Dlm: Decentralized linearized alternating direction method of multipliers. *Signal Processing, IEEE Transactions on*, 63(15):4051–4064, 2015.

Cassio G Lopes and Ali H Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *Signal Processing, IEEE Transactions on*, 56(7):3122–3136, 2008.

Aryan Mokhtari, Qing Ling, and Alejandro Ribeiro. Network newton-part i: Algorithm and convergence. *arXiv preprint arXiv:1504.06017*, 2015a.

Aryan Mokhtari, Qing Ling, and Alejandro Ribeiro. Network newton-part ii: Convergence rate and implementation. *arXiv preprint arXiv:1504.06020*, 2015b.

Aryan Mokhtari, Wei Shi, Qing Ling, and Alejandro Ribeiro. Decentralized quadratically approximated alternating direction method of multipliers. In *Proc. IEEE Global Conf. on Signal and Inform. Process.*, 2015c.

Angelia Nedić and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *Automatic Control, IEEE Transactions on*, 54(1):48–61, 2009.

Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 20–27. ACM, 2004.

Alejandro Ribeiro. Ergodic stochastic optimization algorithms for wireless communication and networking. *Signal Processing, IEEE Transactions on*, 58(12):6369–6386, 2010.

Alejandro Ribeiro. Optimal resource allocation in wireless communication and networking. *EURASIP Journal on Wireless Communications and Networking*, 2012(1):1–19, 2012.

Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in Neural Information Processing Systems*, pages 2663–2671, 2012.

Ioannis D Schizas, Alejandro Ribeiro, and Georgios B Giannakis. Consensus in ad hoc wsns with noisy links–part i: Distributed estimation of deterministic signals. *Signal Processing, IEEE Transactions on*, 56(1):350–364, 2008.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *arXiv preprint arXiv:1309.2388*, 2013.

Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss. *The Journal of Machine Learning Research*, 14(1):567–599, 2013.

Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *Signal Processing, IEEE Transactions on*, 62(7):1750–1761, 2014.

Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Victor Solo and Xuan Kong. *Adaptive Signal Processing Algorithms: Stability and Performance*. NJ: Prentice-Hall, Englewood Cliffs, 1995.

Konstantinos Tsianos, Sean Lawlor, Michael G Rabbat, et al. Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pages 1543–1550. IEEE, 2012a.

Konstantinos I. Tsianos, Sean Lawlor, and Michael G Rabbat. Push-sum distributed dual averaging for convex optimization. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 5453–5458. IEEE, 2012b.

Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. *arXiv preprint arXiv:1310.7063*, 2013.

Journal of Machine Learning Research 17 (2016) 1-29 Submitted 7/15; Revised 1/16; Published 4/16

The Statistical Performance of Collaborative Inference

Gérard Biau GERARD.BIAU@UPMC.FR
Laboratoire de Statistique Théorique et Appliquée, FRE CNRS 3684
Université Pierre et Marie Curie
Boîte 158, 4 place Jussieu
75005, Paris, France

Kevin Bleakley KEVIN.BLEAKLEY@INRIA.FR
INRIA Saclay – Ile-de-France
1 rue Honoré d'Estienne d'Orves
91120, Palaiseau, France

Benoît Cadre BENOIT.CADRE@ENS-RENNES.FR
IRMAR, ENS Rennes
Campus de Ker Lann
Avenue Robert Schuman
35170 Bruz, France

Editor: Gábor Lugosi

Abstract

The statistical analysis of massive and complex data sets will require the development of algorithms that depend on distributed computing and collaborative inference. Inspired by this, we propose a collaborative framework that aims to estimate the unknown mean θ of a random variable X. In the model we present, a certain number of calculation units, distributed across a communication network represented by a graph, participate in the estimation of θ by sequentially receiving independent data from X while exchanging messages via a stochastic matrix A defined over the graph. We give precise conditions on the matrix A under which the statistical precision of the individual units is comparable to that of a (gold standard) virtual centralized estimate, even though each unit does not have access to all of the data. We show in particular the fundamental role played by both the non-trivial eigenvalues of A and the Ramanujan class of expander graphs, which provide remarkable performance for moderate algorithmic cost.

Keywords: distributed computing, collaborative estimation, stochastic matrix, graph theory, complexity, Ramanujan graph

1. Introduction

A promising way to overcome computational problems associated with inference and prediction in large-scale settings is to take advantage of distributed and collaborative algorithms, whereby several processors perform computations and exchange messages with the end-goal of minimizing a certain cost function. For instance, in modern data analysis one is frequently faced with problems where the sample size is too large for a single computer or standard computing resources. Distributed processing of such large data sets is often regarded as a possible solution to data overload, although designing and analyzing algorithms in this setting is challenging. Indeed, good distributed and collaborative architectures should maintain the desired statistical accuracy of their centralized counterpart, while retaining sufficient flexibility and avoiding communication bottlenecks which may excessively slow down computations. The literature is too vast to permit anything like a fair summary within the confines of a short introduction—the papers by Duchi et al. (2012), Jordan (2013), Zhang et al. (2013), and references therein contain a sample of relevant work.

Similarly, the advent of sensor, wireless, and peer-to-peer networks in science and technology necessitates the design of distributed and information-exchange algorithms (Boyd et al., 2006; Predd et al., 2009). Such networks are designed to perform inference and prediction tasks for the environments they are sensing. Nonetheless, they are typically characterized by constraints on energy, bandwidth, and/or privacy, which limit the sensors' ability to share data with each other or with a hub for centralized processing. For example, in a hospital network, the aim is to make safer decisions by sharing information between therapeutic services. However, a simple exchange of database entries containing patient details can pose information privacy risks. At the same time, a large percentage of medical data may require exchanging high-resolution images, the centralized processing of which may be computationally prohibitive. Overall, such constraints call for the design of communication-constrained distributed procedures, where each node exchanges information with only a few of its neighbors at each time instance. The goal in this setting is to distribute the learning task in a computationally efficient way, and make sure that the statistical performance of the network matches that of the centralized version.

The foregoing observations have motivated the development and analysis of many local message-passing algorithms for distributed and collaborative inference, optimization, and learning. Roughly speaking, message-passing procedures are those that use only local communication to approximately achieve the same end as global (i.e., centralized) algorithms, which require sending raw data to a central processing facility. Message-passing algorithms are thought to be efficient by virtue of their exploitation of local communication. They have been successfully involved in kernel linear least-squares regression estimation (Predd et al., 2009), support vector machines (Forero et al., 2010), sparse L_1 regression (Mateos et al., 2010), gradient-type optimization (Tsitsiklis et al., 1986; Bertsekas and Tsitsiklis, 1997), and various online inference and learning tasks (Bianchi et al., 2011a,b, 2013). An important research effort has also been devoted to so-called averaging and consensus problems, where a set of autonomous agents—which may be sensors or nodes of a computer network—compute the average of their opinions in the presence of restricted communication capabilities and try to agree on a collective decision (e.g., Blondel et al., 2005; Olshevsky and Tsitsiklis, 2011).

However, despite their rising success and impact in machine learning, little is known regarding the statistical properties of message-passing algorithms. The statistical performance of collaborative computing has so far been studied in terms of consensus (i.e., whether all nodes give the same result), with perhaps mean convergence rates (e.g., Olshevsky and Tsitsiklis, 2011; Duchi et al., 2012; Zhang et al., 2013). While it is therefore proved that using a network, even sparse (i.e., with few connections), does not degrade the rate of convergence, the problem of whether it is optimal to do this remains unanswered, including for the most basic statistics. For example, which network properties guarantee collaborative calculation performances equal to those of a hypothetical centralized system? The goal

of this article is to give a more precise answer to this fundamental question. In order to present in the clearest way possible the properties such a network must have, we undertake this study for the most simple statistic possible: the mean.

In the model we consider, there are a number of computing agents (also known as nodes or processors) that sequentially estimate the mean of a random variable by regularly updating an estimate stored in their memory. Meanwhile, they exchange messages, thus informing each other about the results of their latest computations. Agents that receive messages use them to directly update the value in their memory by forming a convex combination. We focus primarily on the properties that the communication process must satisfy to ensure that the statistical precision of a single processor—that only sees part of the data—is similar to that of an inaccessible centralized intelligence that could tackle the whole data set at once. The literature is surprisingly quiet on this question, which we believe is of fundamental importance if we want to provide concrete tradeoffs between communication constraints and statistical accuracy.

This paper makes several important contributions. First, in Section 2 we introduce communication network models and define a performance ratio allowing us to quantify the statistical quality of a network. In Section 3 we analyze the asymptotic behavior of this performance ratio as the number of data items t received online sequentially per node becomes large, and give precise conditions on communication matrices A so that this ratio is asymptotically optimal. Section 4 goes one step further, connecting the rate of convergence of the ratio with the behavior of the eigenvalues of A. In Section 5 we present the remarkable Ramanujan expander graphs and analyze the tradeoff between statistical efficiency and communication complexity for these graphs with a series of simulation studies. Lastly, Section 6 provides several elements for analysis of more complicated asynchronous models with delays. A short discussion follows in Section 7. For clarity, proofs are gathered in Section 8.

2. The model

Let X be a square-integrable real-valued random variable, with $\mathbb{E}X = \theta$ and $\mathrm{Var}(X) = \sigma^2$. We consider a set $\{1, \dots, N\}$ of computing entities ($N \geq 2$) that collectively participate in the estimation of θ. In this distributed model, agent i sequentially receives an i.i.d. sequence $X_1^{(i)}, \dots, X_t^{(i)}, \dots$, distributed as the prototype X, and forms, at each time t, an estimate of θ. It is assumed throughout that the $X_t^{(i)}$ are independent when both $t \geq 1$ and $i \in \{1, \dots, N\}$ vary.

In the absence of communication between agents, the natural estimate held by agent i at time t is the empirical mean

$$\bar{X}_t^{(i)} = \frac{1}{t} \sum_{k=1}^{t} X_k^{(i)}.$$

Equivalently, processor i is initialized with $X_1^{(i)}$ and performs its estimation via the iteration

$$\bar{X}_{t+1}^{(i)} = \frac{t\bar{X}_t^{(i)} + X_{t+1}^{(i)}}{t+1}, \quad t \geq 1.$$

Let $\top$ denote transposition and assume that vectors are in column format. Letting $\mathbf{X}_t = (X_t^{(1)}, \dots, X_t^{(N)})^\top$ and $\bar{\mathbf{X}}_t = (\bar{X}_t^{(1)}, \dots, \bar{X}_t^{(N)})^\top$, we see that

$$\bar{\mathbf{X}}_{t+1} = \frac{t\bar{\mathbf{X}}_t + \mathbf{X}_{t+1}}{t+1}, \quad t \geq 1. \tag{1}$$

In a more complicated collaborative setting, besides its own measurements and computations, each agent may also receive messages from other processors and combine this information with its own conclusions. At its core, this message-passing process can be modeled by a *directed* graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ with vertex set $\mathscr{V} = \{1, \dots, N\}$ and edge set $\mathscr{E}$. This graph represents the way agents communicate, with an edge from j to i (in that order) if j sends information to i. Furthermore, we have an $N \times N$ stochastic matrix $A = (a_{ij})_{1 \leq i,j \leq N}$ (i.e., $a_{ij} \geq 0$ and for each i, $\sum_{j=1}^N a_{ij} = 1$) with associated graph $\mathscr{G}$, i.e., $a_{ij} > 0$ if and only if $(j, i) \in \mathscr{E}$. The matrix A accounts for the way agents incorporate information during the collaborative process. Denoting by $\hat{\boldsymbol{\theta}}_t = (\hat{\theta}_t^{(1)}, \dots, \hat{\theta}_t^{(N)})^\top$ the collection of estimates held by the N agents over time, the computation/combining mechanism is assumed to be as follows:

$$\hat{\boldsymbol{\theta}}_{t+1} = \frac{t}{t+1} A\hat{\boldsymbol{\theta}}_t + \frac{1}{t+1}\mathbf{X}_{t+1}, \quad t \geq 1,$$

with $\hat{\boldsymbol{\theta}}_1 = (X_1^{(1)}, \dots, X_1^{(N)})^\top$. Thus, each individual estimate $\hat{\theta}_{t+1}^{(i)}$ is a convex combination of the estimates $\hat{\theta}_t^{(j)}$ held by the agents over the network at time t, augmented by the new observation $X_{t+1}^{(i)}$.

The matrix A models the way processors exchange messages and collaborate, ranging from $A = I_N$ (the $N \times N$ identity matrix, i.e., no communication) to $A = \mathbf{1}\mathbf{1}^\top/N$ (where $\mathbf{1} = (1, \dots, 1)^\top$, i.e., full communication). We note in particular that the choice $A = I_N$ gives back iteration (1) with $\hat{\boldsymbol{\theta}}_t = \bar{\mathbf{X}}_t$. We also note that, given a graph $\mathscr{G}$, various choices are possible for A. Thus, aside from a convenient way to represent a communication channel over which agents can retrieve information from each other, the matrix A can be seen as a "tuning parameter" on $\mathscr{G}$ to improve the statistical performance of $\hat{\boldsymbol{\theta}}_t$, as we shall see later. Important examples for A include the choices

$$A_1 = \frac{1}{2}\begin{pmatrix} 1 & 1 & & & & & & & & \\ 1 & 0 & 1 & & & & & & & \\ & 1 & 0 & 1 & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & & & 1 & 0 & 1 & & & \\ & & & & & 1 & 0 & 1 & & \\ & & & & & & 1 & 1 & & \end{pmatrix} \tag{2}$$

and

$$A_2 = \frac{1}{3}\begin{pmatrix} 2 & 1 & & & & & & & & \\ 1 & 1 & 1 & & & & & & & \\ & 1 & 1 & 1 & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & & & 1 & 1 & 1 & & & \\ & & & & & 1 & 1 & 1 & & \\ & & & & & & 1 & 2 & & \end{pmatrix} \tag{3}$$

(unmarked entries are zero).

It is easy to verify that for all $t \geq 1$,

$$\hat{\boldsymbol{\theta}}_t = \frac{1}{t}\sum_{k=0}^{t-1} A^k \mathbf{X}_{t-k}. \tag{4}$$

Thus, denoting by $\|\cdot\|$ the Euclidean norm (for vector or matrices), we may write, for all $t \geq 1$,

$$\begin{aligned}\mathbb{E}\|\hat{\boldsymbol{\theta}}_t - \theta\mathbf{1}\|^2 &= \frac{1}{t^2}\mathbb{E}\left\|\sum_{k=0}^{t-1} A^k(\mathbf{X}_{t-k} - \theta\mathbf{1})\right\|^2 \\ &\qquad \text{(since } A^k \text{ is a stochastic matrix)} \\ &= \frac{1}{t^2}\sum_{k=1}^{t}\mathbb{E}\left\|A^{t-k}(\mathbf{X}_k - \theta\mathbf{1})\right\|^2,\end{aligned}$$

by independence of $\mathbf{X}_1, \ldots, \mathbf{X}_t$. It follows that

$$\begin{aligned}\mathbb{E}\|\hat{\boldsymbol{\theta}}_t - \theta\mathbf{1}\|^2 &\leq \mathbb{E}\|\mathbf{X}_1 - \theta\mathbf{1}\|^2 \times \frac{1}{t^2}\sum_{k=0}^{t-1}\|A^k\|^2 \\ &\leq \mathbb{E}\|\mathbf{X}_1 - \theta\mathbf{1}\|^2 \times \frac{N}{t}.\end{aligned}$$

In the last inequality, we used the fact that A^k is a stochastic matrix and thus $\|A^k\|^2 \leq N$ for all $k \geq 0$. We can merely conclude that $\mathbb{E}\|\hat{\boldsymbol{\theta}}_t - \theta\mathbf{1}\|^2 \to 0$ as $t \to \infty$ (mean-squared error consistency), and so $\hat{\theta}_t^{(i)} \to \theta$ in probability for each $i \in \{1, \ldots, N\}$. Put differently, the agents asymptotically agree on the (true) value of the parameter, independently of the choice of the (stochastic) matrix A—this property is often called consensus in the distributed optimization literature (see, e.g., Bertsekas and Tsitsiklis, 1997). We insist on the fact that in our framework, consensus is obvious, and is not the question we are looking at here.

The consensus property, although interesting, does not say anything about the positive (or negative) impact of the graph on the comparative performances of estimates with respect to a centralized version. To clarify this remark, assume that there exists a centralized intelligence that could tackle all data $X_1^{(1)}, \ldots, X_t^{(1)}, \ldots, X_1^{(N)}, \ldots, X_t^{(N)}$ at time t, and take advantage of these sample points to assess the value of the parameter θ. In this ideal framework, the natural estimate of θ is the global empirical mean

$$\bar{\mathbb{X}}_{Nt} = \frac{1}{Nt}\sum_{i=1}^{N}\sum_{k=1}^{t} X_k^{(i)},$$

which is clearly the best we can hope for with the data at hand. However, this estimate is to be considered as an unattainable "gold standard" (or oracle), insofar as it uses the whole $(N \times t)$-sample. In other words, its evaluation requires sending all examples to a centralized processing facility, which is precisely what we want to avoid.

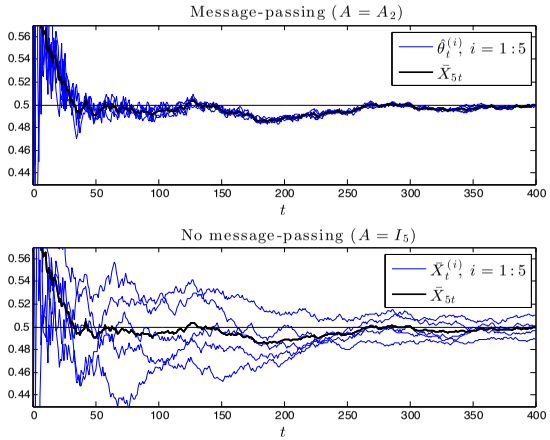


Figure 1: Convergence of individual nodes' estimates with and without message-passing.

Thus, a natural question arises: can the message-passing process be tapped to ensure that the individual estimates $\hat{\theta}_t^{(i)}$ achieve statistical accuracy "close" to that of the gold standard $\bar{\mathbb{X}}_{Nt}$? Figure 1 illustrates this pertinent question.

In the trials shown, i.i.d. uniform random variables on $[0, 1]$ are delivered online to $N = 5$ nodes, one to each at each time t. With message-passing (here, $A = A_2$), each node aggregates the new data point with data it has seen previously and messages received from its nearest neighbors in the network. We see that all of the five nodes' updates seem to converge with a performance comparable to that of the (unseen) global estimate $\bar{\mathbb{X}}_{Nt}$ to the mean 0.5. In contrast, in the absence of message-passing ($A = I_5$), individual nodes' estimates do still converge to 0.5, but at a slower rate.

To deal with this question of statistical accuracy satisfactorily, we first need a criterion to compare the performance of $\hat{\boldsymbol{\theta}}_t$ with that of $\bar{\mathbb{X}}_{Nt}$. Perhaps the most natural one is the following ratio, which depends upon the matrix A:

$$\tau_t(A) = \frac{\mathbb{E}\left\|(\bar{\mathbb{X}}_{Nt} - \theta)\mathbf{1}\right\|^2}{\mathbb{E}\|\hat{\boldsymbol{\theta}}_t - \theta\mathbf{1}\|^2}, \quad t \geq 1.$$

The more this ratio is close to 1, the more the collaborative algorithm is statistically efficient, in the sense that its performance compares favorably to that of the centralized gold standard. In the remainder of the paper, we call $\tau_t(A)$ the *performance ratio* at time t.

Of particular interest in our approach is the stochastic matrix A, which plays a crucial role in the analysis. Roughly, a good choice for A is one for which $\tau_t(A)$ is not too far from 1, while ensuring that communication over the network is not prohibitively expensive. Although there are several ways to measure "complexity" of the message-passing process, we have in mind a setting where the communication load is well-balanced between agents,

in the sense that no node should play a dominant role. To formalize this idea, we define the communication-complexity index $\mathscr{C}(A)$ as the maximal indegree of the edges of the graph $\mathscr{G}$ associated with A, i.e., the maximal number of edges pointing to a node in $\mathscr{G}$ (by convention, self-loops are counted twice when $\mathscr{G}$ is undirected). Essentially, A is communication-efficient when $\mathscr{C}(A)$ is small with respect to N or, more generally, when $\mathscr{C}(A) = \mathrm{O}(1)$ as N becomes large.

To provide some context, $\mathscr{C}(A)$ measures in a certain sense the "local" aspect of message exchanges induced by A. We have in mind node connection set-ups where $\mathscr{C}(A)$ is small, perhaps due to energy or bandwidth constraints in the system's architecture, or when for privacy reasons data must not be sent to a central node. Indeed, a large $\mathscr{C}(A)$ roughly means that one or several nodes play centralized roles—precisely what we are trying to avoid. Furthermore, the decentralized networks we are interested in can be seen as being more autonomous than high-$\mathscr{C}(A)$ ones, in the sense that having few network connections means less things that can potentially break, as well as improved robustness due to the fact that the loss of one node does not lead to destruction of the whole system. As examples, the matrices A_1 and A_2 defined earlier have $\mathscr{C}(A_1) = 3$ and $\mathscr{C}(A_2) = 4$, respectively, while the stochastic matrix A_3 below has $\mathscr{C}(A_3) = N + 1$:

$$A_3 = \frac{1}{N}\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 1 & N-1 & & & & & \\ 1 & & N-1 & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & & & & & & N-1 \end{pmatrix}. \tag{5}$$

Thus, from a network complexity point of view, A_1 and A_2 are preferable to A_3 where node 1 has the flavor of a central command center.

Now, having defined $\tau_t(A)$ and $\mathscr{C}(A)$, it is natural to suspect that there will be some kind of tradeoff between implementing a low-complexity message-passing algorithm (i.e., $\mathscr{C}(A)$ small) and achieving good asymptotic performance (i.e., $\tau_t(A) \approx 1$ for large t). Our main goal in the next few sections is to probe this intuition by analyzing the asymptotic behavior of $\tau_t(A)$ as $t \to \infty$ under various assumptions on A. We start by proving that $\tau_t(A) \leq 1$ for all $t \geq 1$, and give precise conditions on the matrix A under which $\tau_t(A) \to 1$. Thus, thanks to the benefit of inter-agent communication, the statistical accuracy of individual estimates may be asymptotically comparable to that of the gold standard, despite the fact that none of the agents in the network have access to all of the data. Indeed, as we shall see, this stunning result is possible even for low-$\mathscr{C}(A)$ matrices. The take-home message here is that the communication process, once cleverly designed, may "boost" the individual estimates, even in the presence of severe communication constraints. We also provide an asymptotic development of $\tau_t(A)$, which offers valuable information on the optimal way to design the communication network in terms of the eigenvalues of A.

3. Convergence of the performance ratio

Recall that a stochastic square matrix $A = (a_{ij})_{1 \leq i,j \leq N}$ is irreducible if for every pair of indices i and j, there exists a nonnegative integer k such that $(A^k)_{ij}$ is not equal to 0. The matrix is said to be reducible if it is not irreducible.

Proposition 1 *We have $\frac{1}{N} \leq \tau_t(A) \leq 1$ for all $t \geq 1$. In addition, if A is reducible, then*

$$\tau_t(A) \leq 1 - \frac{1}{N+1}, \quad t \geq 1.$$

It is apparent from the proof of the proposition (all proofs are found in Section 8) that the lower bound $1/N$ for $\tau_t(A)$ is achieved by taking $A = I_N$, which is clearly the worst choice in terms of communication. This proposition also shows that the irreducibility of A is a necessary condition for the collaborative algorithm to be statistically efficient, for otherwise there exists $\varepsilon \in (0,1)$ such that $\tau_t(A) \leq 1 - \varepsilon$ for all $t \geq 1$.

We recall from the theory of Markov chains (e.g., Grimmett and Stirzaker, 2001) that for a fixed agent $i \in \{1, \ldots, N\}$, the period of i is the greatest common divisor of all positive integers k such that $(A^k)_{ii} > 0$. When A is irreducible, the period of every state is the same and is called the period of A. The following lemma describes the asymptotic behavior of $\tau_t(A)$ as t tends to infinity.

Lemma 2 *Assume that A is irreducible, and let d be its period. Then there exist projectors $Q_1, \ldots, Q_d$ such that*

$$\tau_t(A) \to \frac{1}{\sum_{\ell=1}^{d} \|Q_\ell\|^2} \quad \text{as } t \to \infty.$$

The projectors $Q_1, \ldots, Q_d$ in Lemma 2 originate from the decomposition

$$A^k = \sum_{\ell=1}^{d} \lambda_\ell^k Q_\ell + \sum_{\gamma \in \Gamma} \gamma^k Q_\gamma(k),$$

where $\lambda_1 = 1, \ldots, \lambda_d$ are the eigenvalues of A (distinct) of unit modulus, Γ the set of eigenvalues of A of modulus strictly smaller than 1, and $Q_\gamma(k)$ certain $N \times N$ matrices (see Theorem 8 in the proofs section). In particular, we see that $\tau_t(A) \to 1$ as $t \to \infty$ if and only if $\sum_{\ell=1}^{d} \|Q_\ell\|^2 = 1$. It turns out that this condition is satisfied if and only if A is irreducible, aperiodic (i.e., $d = 1$), and bistochastic, i.e., $\sum_{i=1}^{N} a_{ij} = \sum_{j=1}^{N} a_{ij} = 1$ for all $(i,j) \in \{1, \ldots, N\}^2$. This important result is encapsulated in the next theorem.

Theorem 3 *We have $\tau_t(A) \to 1$ as $t \to \infty$ if and only if A is irreducible, aperiodic, and bistochastic.*

Theorem 3 offers necessary and sufficient conditions for the communication matrix A to be asymptotically statistically efficient. Put differently, under the conditions of the theorem, the message-passing process conveys sufficient information to local computations to make individual estimates as accurate as the gold standard for large t. We again stress that this theorem is new and different from results obtained in the consensus literature. The theorem shows that one machine, on its own, if it is well-informed, does as good a job as a virtual central machine that has access to all the data.

In the context of multi-agent coordination, an example of such a communication network is the so-called (time-invariant) equal neighbor model (Tsitsiklis et al., 1986; Olshevsky and Tsitsiklis, 2011), in which

$$a_{ij} = \begin{cases} 1/|N^{(i)}| & \text{if } j \in N^{(i)} \\ 0 & \text{otherwise,} \end{cases}$$

where

$$N^{(i)} = \big\{ j \in \{1, \dots, N\} : a_{ij} > 0 \big\}$$

is the set of agents whose value is taken into account by i, and $|N^{(i)}|$ its cardinality. Clearly, the communication matrix A is stochastic, and also bistochastic as soon as A is symmetric (bidirectional model). Assuming in addition that the directed graph $\mathscr{G}$ associated with A is connected means that A is irreducible. Moreover, if $a_{ii} > 0$ for some $i \in \{1, \dots, N\}$, then A is also aperiodic, so the conditions of Theorem 3 are fulfilled.

Another way to choose an irreducible, aperiodic, and bistochastic matrix on an undirected graph is by letting $a_{ij} = 1/\max(1 + d(i), 1 + d(j))$, where $d(i)$ is the degree of node i; following this, a_{ii} is set to that which is needed to make each row sum to 1.

It is also interesting to note that there exist low-$\mathscr{C}(A)$ matrices that meet the requirements of Theorem 3. This is for instance the case of matrices A_1 and A_2 in (2) and (3), which are irreducible, aperiodic, and bistochastic, and satisfy $\mathscr{C}(A) \leq 4$. Also note that the matrix A_3 in (5), though irreducible, aperiodic, and bistochastic, should be avoided because $\mathscr{C}(A_3) = N + 1$.

We stress that the irreducibility and aperiodicity conditions are inherent properties of the graph $\mathscr{G}$, not A, insofar as these conditions do not depend upon the actual values of the nonzero entries of A. This is different for the bistochasticity condition, which requires knowledge of the coefficients of A. In fact, as observed by Sinkhorn and Knopp (1967), it is not always possible to associate such a bistochastic matrix with a given directed graph $\mathscr{G}$. To be more precise, consider $G = (g_{ij})_{1 \leq i,j \leq N}$, the transpose of the adjacency matrix of the graph $\mathscr{G}$—that is, $g_{ij} \in \{0, 1\}$ and $g_{ij} = 1 \Leftrightarrow (j, i) \in \mathscr{E}$. Then G is said to have total support if, for every positive element g_{ij}, there exists a permutation σ of $\{1, \dots, N\}$ such that $j = \sigma(i)$ and $\prod_{k=1}^{N} g_{k\sigma(k)} > 0$. The main theorem of Sinkhorn and Knopp (1967) asserts that there exists a bistochastic matrix A of the form $A = D_1 G D_2$, where D_1 and D_2 are $N \times N$ diagonal matrices with positive diagonals, if and only if G has total support. The algorithm to induce A from G is called the Sinkhorn-Knopp algorithm. It does this by generating a sequence of matrices whose rows and columns are normalized alternately. It is known that the convergence of the algorithm is linear, and upper bounds have been given for its rate of convergence (e.g., Knight, 2008).

Nevertheless, if for some reason we face a situation where it is impossible to associate a bistochastic matrix with the graph $\mathscr{G}$, Proposition 4 below shows that it is still possible to obtain information about the performance ratio, provided A is irreducible and aperiodic.

Proposition 4 *Assume that A is irreducible and aperiodic. Then*

$$\tau_t(A) \to \frac{1}{N\|\boldsymbol{\mu}\|^2} \quad \text{as } t \to \infty,$$

where $\boldsymbol{\mu}$ is the stationary distribution of A.

To illustrate this result, take $N = 2$ and consider the graph $\mathscr{G}$ with (symmetric) adjacency matrix $\mathbf{1}\mathbf{1}^\top$ (i.e., full communication). Various stochastic matrices may be associated with $\mathscr{G}$, each with a certain statistical performance. For $\alpha > 1$ a given parameter, we may choose for example

$$H_\alpha = \frac{1}{\alpha} \begin{pmatrix} 1 & \alpha - 1 \\ 1 & \alpha - 1 \end{pmatrix}.$$

When $\alpha = 2$, we have $\tau_t(H_2) \to 1$ by Theorem 3. More generally, using Proposition 4, it is an easy exercise to prove that, as $t \to \infty$,

$$\tau_t(H_\alpha) \to \frac{\alpha^2}{2 + 2(\alpha - 1)^2}.$$

We see that the statistical performance of the local estimates deteriorates as α becomes large, for in this case $\tau_t(H_\alpha)$ gets closer and closer to $1/2$. This toy model exemplifies the role the stochastic matrix is playing as a "tuning parameter" to improve the performance of the distributed estimate.

4. Convergence rates

Theorem 3 gives precise conditions ensuring $\tau_t(A) = 1 + o(1)$, but does not say anything about the rate (i.e., the behavior of the second-order term) at which this convergence occurs. It turns out that a much more informative limit may be obtained at the price of the mild additional assumption that the stochastic matrix A is symmetric (and hence bistochastic).

Theorem 5 *Assume that A is irreducible, aperiodic, and symmetric. Let $1 > \gamma_2 \geq \cdots \geq \gamma_N > -1$ be the eigenvalues of A different from 1. Then*

$$\tau_t(A) = \frac{1}{1 + \frac{1}{t} \sum_{\ell=2}^{N} \frac{1 - \gamma_\ell^{2t}}{1 - \gamma_\ell^2}}.$$

In addition, setting

$$\mathscr{S}(A) = \sum_{\ell=2}^{N} \frac{1}{1 - \gamma_\ell^2} \quad \text{and} \quad \Gamma(A) = \max_{2 \leq \ell \leq N} |\gamma_\ell|,$$

we have, for all $t \geq 1$,

$$1 - \frac{\mathscr{S}(A)}{t} \leq \tau_t(A) \leq 1 - \frac{\mathscr{S}(A)}{t} + \Gamma^{2t}(A) \frac{\mathscr{S}(A)}{t} + \Big(\frac{\mathscr{S}(A)}{t} \Big)^2.$$

Clearly, we thus have

$$t\big(1 - \tau_t(A)\big) \to \mathscr{S}(A) \quad \text{as } t \to \infty.$$

The take-home message is that the smaller the coefficient $\mathscr{S}(A)$, the better the matrix A performs from a statistical point of view. In this respect, we note that $\mathscr{S}(A) \geq N - 1$ (uniformly over the set of stochastic, irreducible, aperiodic, and symmetric matrices). Consider the full-communication matrix

$$A_0 = \frac{1}{N} \mathbf{1}\mathbf{1}^\top, \tag{6}$$

which models a saturated communication network in which each agent shares its information with all others. The associated communication topology, which has $\mathscr{C}(A_0) = N + 1$, is

roughly equivalent to a centralized algorithm and, as such, is considered inefficient from a computational point of view. On the other hand, intuitively, the amount of statistical information propagating through the network is large so $\mathscr{S}(A_0)$ should be small. Indeed, it is easy to see that in this case, $\gamma_\ell = 0$ for all $\ell \in \{2, \dots, N\}$ and $\mathscr{S}(A_0) = N-1$. Therefore, although complex in terms of communication, A_0 is statistically optimal.

Remark 6 *Interestingly, as pointed out by a referee, $\mathscr{S}(A)$ has a graph theoretic interpretation as the Kemeny constant of the Markov chain A^2, and may be written in terms of hitting times. Consequently, for a number of graphs it is easy to compute—see Jadbabaie and Olshevsky (2015) for example.*

For a comparative study of statistical performance and communication complexity of matrices, let us consider the sparser graph associated with the tridiagonal matrix A_1 defined in (2). With this choice, $\gamma_\ell = \cos\frac{(\ell-1)\pi}{N}$ (Fiedler, 1972), so that

$$\mathscr{S}(A_1) = \sum_{\ell=1}^{N-1} \frac{1}{1-\cos^2\frac{\ell\pi}{N}} = \frac{N^2}{6} + \mathrm{O}(N) \quad \text{as } N \to \infty. \tag{7}$$

Thus, we lose a power of N but now have lower communication complexity $\mathscr{C}(A_1) = 3$.

Let us now consider the tridiagonal matrix A_2 defined in (3). Noticing that $3A_2 = 2A_1 + I_N$, we deduce that for the matrix A_2, $\gamma_\ell = \frac{1}{3} + \frac{2}{3}\cos\frac{(\ell-1)\pi}{N}$, $2 \le \ell \le N$. Thus, as $N \to \infty$,

$$\mathscr{S}(A_2) = \frac{N^2}{9} + \mathrm{O}(N). \tag{8}$$

By comparing (7) and (8), we can conclude that the matrices A_1 and A_2, which are both low-$\mathscr{C}(A)$, are also nearly equivalent from a statistical efficiency point of view. A_2 is nevertheless preferable to A_1, which has a larger constant in front of the N^2. This slight difference may be due to the fact that most of the diagonal elements of A_1 are zero, so that agents $i \in \{2, \dots, N-1\}$ do not integrate their current value in the next iteration, as happens for A_2. Furthermore, for large N, the performance of A_1 and A_2 are expected to dramatically deteriorate in comparison with those of A_0, since $\mathscr{S}(A_1)$ and $\mathscr{S}(A_2)$ are proportional to N^2, while $\mathscr{S}(A_0)$ is proportional to N.

Figure 2 shows the evolution of $\tau_t(A)$ for N fixed and t increasing for the matrices $A = A_0$, A_1, A_2 as well as the identity I_N. As expected, we see convergence of $\tau_t(A_i)$ to 1, with degraded performance as the number of agents N increases. Also, we see that the lack of message-passing for I_N means it is statistically inefficient, with constant $\tau_t(I_N) = 1/N$ for all t.

The discussion and plots above highlight the crucial influence of $\mathscr{S}(A)$ on the performance of the communication network. Indeed, Theorem 5 shows that the optimal order for $\mathscr{S}(A)$ is N, and that this scaling is achieved by the computationally-inefficient choice A_0—see (6). Thus, a natural question to ask is whether there exist communication networks that have $\mathscr{S}(A)$ proportional to N and, simultaneously, $\mathscr{C}(A)$ constant or small with respect to N. These two conditions, which are in a sense contradictory, impose that the absolute values of the non-trivial eigenvalues γ_ℓ stay far from 1, while the maximal indegree of the graph $\mathscr{G}$ remains moderate. It turns out that these requirements are satisfied by so-called Ramanujan graphs, which are presented in the next section.

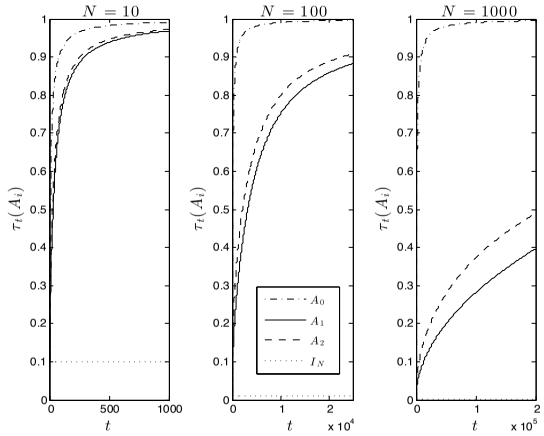


Figure 2: Evolution of $\tau_t(A_i)$ with t for different values of N, for $A = A_0$, A_1, A_2, and I_N.

5. Ramanujan graphs

In this section, we consider *undirected* graphs $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ that are also d-regular, in the sense that all vertices have the same degree d; that is each vertex is incident to exactly d edges. Recall that in this definition, self-loops are counted twice and multiple edges are allowed. However, in what follows, we restrict ourselves to graphs without self-loops and multiple edges. In this setting, the natural (bistochastic) communication matrix A associated with $\mathscr{G}$ is $A = \frac{1}{d}G$, where $G = (g_{ij})_{1 \le i,j \le N}$ is the adjacency matrix of $\mathscr{G}$ ($g_{ij} \in \{0,1\}$ and $g_{ij} = 1 \Leftrightarrow (i,j) \in \mathscr{E}$). Note that $\mathscr{C}(A) = d$.

The matrix G is symmetric and we let $d = \mu_1 \ge \mu_2 \ge \cdots \ge \mu_N \ge -d$ be its (real) eigenvalues. Similarly, we let $1 = \gamma_1 \ge \gamma_2 \ge \cdots \ge \gamma_N \ge -1$ be the eigenvalues of A, with the straightforward correspondence $\gamma_i = \mu_i/d$. We note that A is irreducible (or, equivalently, that $\mathscr{G}$ is connected) if and only if $d > \mu_2$ (see, e.g., Shlomo et al., 2006, Section 2.3). In addition, A is aperiodic as soon as $\mu_N > -d$. According to the Alon-Boppana theorem (Nilli, 1991) one has, for every d-regular graph,

$$\mu_2 \ge 2\sqrt{d-1} - \mathrm{o}_N(1),$$

where the $\mathrm{o}_N(1)$ term is a quantity that tends to zero for every fixed d as $N \to \infty$. Moreover, a d-regular graph $\mathscr{G}$ is called Ramanujan if

$$\max\big(|\mu_\ell| : \mu_\ell < d\big) \le 2\sqrt{d-1}.$$

In view of the above, a Ramanujan graph is optimal, at least as far as the spectral gap measure of expansion is concerned. Ramanujan graphs fall in the category of so-called

expander graphs, which have the apparently contradictory features of being both highly connected and at the same time sparse (for a review, see Shlomo et al., 2006).

Although the existence of Ramanujan graphs for any degree larger than or equal to 3 has been recently established by Marcus et al. (2015), their explicit construction remains difficult to use in practice. However, a conjecture by Alon (1986), proved by Friedman (2008) (see also Bordenave, 2015) asserts that most d-regular graphs are Ramanujan, in the sense that for every $\varepsilon > 0$,

$$\mathbb{P}\Big(\max\big(|\mu_2|, |\mu_N|\big) \geq 2\sqrt{d-1} + \varepsilon \Big) \to 0 \quad \text{as } N \to \infty,$$

or equivalently, in terms of the eigenvalues of A,

$$\mathbb{P}\Big(\max\big(|\gamma_2|, |\gamma_N|\big) \geq \frac{2\sqrt{d-1}}{d} + \varepsilon \Big) \to 0 \quad \text{as } N \to \infty.$$

In both results, the limit is along any sequence going to infinity with Nd even, and the probability is with respect to random graphs uniformly sampled in the family of d-regular graphs with vertex set $\mathscr{V} = \{1, \ldots, N\}$.

In order to generate a random irreducible, aperiodic d-regular Ramanujan graph, we can first generate a random d-regular graph using an improved version of the standard *pairing* algorithm, proposed by Steger and Wormald (1999). We retain it if it passes the tests of being irreducible, aperiodic, and Ramanujan as described above. Otherwise, we continue to generate a d-regular graph until all these conditions are satisfied. Figure 3 gives an example of a 3-regular Ramanujan graph with $N = 16$ vertices, generated in this way.

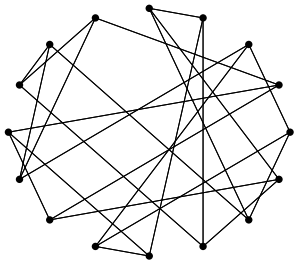


Figure 3: Randomly-generated 3-regular Ramanujan graph with $N = 16$ vertices.

Now, given an irreducible and aperiodic communication matrix A associated with a d-regular Ramanujan graph $\mathscr{G}$, we have, whenever $d \geq 3$,

$$\mathscr{S}(A) \leq \frac{N-1}{1 - \frac{4(d-1)}{d^2}}.$$

Thus, recalling that $\mathscr{S}(A) \geq N - 1$, we see that $\mathscr{S}(A)$ scales optimally as N while having $\mathscr{C}(A) = d$ (fixed). This remarkable super-efficiency property can be compared with the

full-communication matrix A_0, which has $\mathscr{S}(A_0) = N - 1$ but inadmissible complexity $\mathscr{C}(A_0) = N + 1$.

The statistical efficiency of these graphs is further highlighted in Figure 4. It shows results for 3- and 5-regular Ramanujan-type matrices (A_3 and A_5) as well as the previous results for non-Ramanujan-type matrices A_0, A_1, and A_2 (see Figure 2).

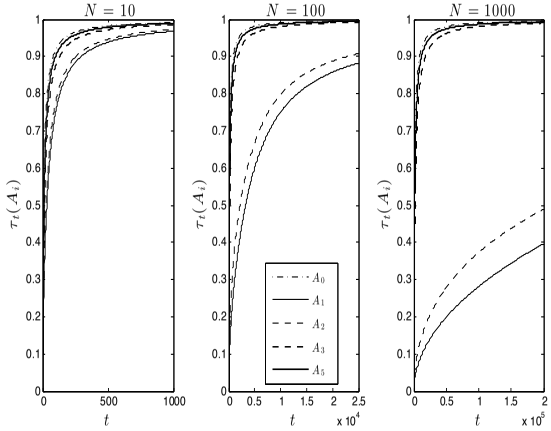


Figure 4: Evolution of $\tau_t(A_i)$ with t for different values of N, for $A = A_0$, A_1, A_2 as before with the addition of 3- and 5-regular Ramanujan-type matrices A_3 and A_5.

We see that A_3 is already close to the statistical performance of A_0, the saturated network, and for all intents and purposes A_5 is essentially as good as A_0, even when there are $N = 1000$ nodes; i.e., the statistical performance of the 5-regular Ramanujan graph is barely distinguishable from that of the totally connected graph! Nevertheless, we must not forget that the possibility of building such efficient networks in real-world situations will ultimately depend on the specific application, and may not always be possible.

Next, assuming that the Ramanujan-type matrix A is irreducible and aperiodic, it is apparent that there is a compromise to be made between the communication complexity of the algorithm (as measured by the degree index $\mathscr{C}(A) = d$) and its statistical performance (as measured by the coefficient $\mathscr{S}(A)$). Clearly, the two are in conflict. Upon this a question arises: is it possible to reach a compromise in the range of statistical performances $\mathscr{S}(A)$ while varying the communication complexity between $d = 3$ and $d = N$? The answer is affirmative, as shown in the following simulation exercise.

We fix $N = 200$ and then for each $d = 3, \ldots, N$:

(i) Generate a matrix A_d associated with a d-regular Ramanujan graph as before.

(*ii*) Compute the (non-unitary) eigenvalues $\gamma_2^{(d)}, \ldots, \gamma_N^{(d)}$ of the matrix A_d and evaluate the sum

$$\mathscr{S}(A_d) = \sum_{\ell=2}^{N} \frac{1}{1 - \left(\gamma_\ell^{(d)}\right)^2}.$$

(*iii*) Plot $\mathscr{S}(A_d)$ and $\beta\mathscr{C}(A_d) = \beta d$ as well as penalized sums $\mathscr{S}(A_d) + \beta\mathscr{C}(A_d)$ for $\beta \in \{1/2, 1, 2, 4\}$, where β represents an explicit cost incurred when increasing the number of connections between nodes.

Results are shown in Figure 5, where $d^\star$ refers to the d for which the penalized sum $\mathscr{S}(A_d) + \beta\mathscr{C}(A_d)$ is minimized.

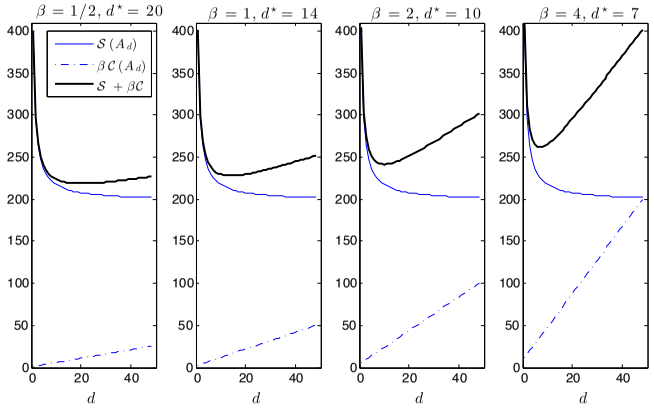


Figure 5: Statistical efficiency vs communication complexity tradeoff for four different node communication penalties β. $d^\star$ is the d which minimizes $\mathscr{S}(A_d) + \beta\mathscr{C}(A_d)$.

We observe that $\mathscr{S}(A_d)$ is decreasing whereas $\mathscr{C}(A_d)$ increases linearly. The tradeoff between statistical efficiency and communication complexity can be seen as minimizing their penalized sum, where β for example represents a monetary cost incurred by adding new network connections between nodes. We see that the optimal $d^\star$ and thus the number of node connections decreases as the cost of adding new ones increases.

Next, let us investigate the tradeoffs involved in the case where we have a large but fixed total number T of data to be streamed to N nodes, each receiving one new data value from time $t = 1$ to time $t = T/N$. In this context, the natural question to ask is how many nodes should we choose, and how much communication should we allow between them in order to get "very good" results for a "low" cost? Here a low cost comes from both limiting the number of nodes as well as the number of connections between them.

In the same set-up for A_d defined above, one way to look at this is to ask, for each N, what is the smallest $d \in \{3, \ldots, N\}$ and therefore the smallest communication cost

$\mathscr{C}(A_d) = d$ for which the performance ratio $\tau_t(A_d)$ is at least 0.99 after receiving all the data, i.e., when $t = T/N$? Then, as there is also a cost associated with increasing N, minimizing $\mathscr{C}(A_{d^\star})/N$ (where $d^\star$ is this smallest d chosen) should help us choose the number of nodes N and the amount of connection $\mathscr{C}(A_{d^\star})$ between them. The result of this is shown in Figure 6 for $T = 100$ million data points.

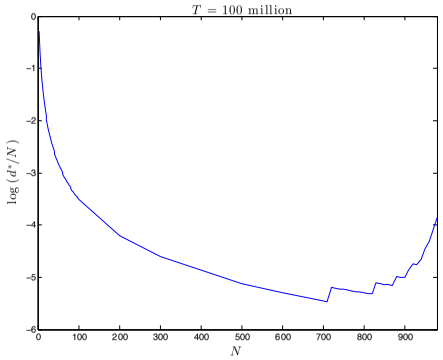


Figure 6: Optimizing the number of nodes N and the level of communication d required between nodes to obtain a performance ratio $\tau_t(A_d) \geq 0.99$ given a large fixed quantity of data T.

The minimum is found at $(N, d^\star) = (710, 3)$, suggesting that with 100 million data points, one can get excellent performance results ($\tau_t(A_{d^\star}) \geq 0.99$) for a low cost with around 700 nodes, each connected only to three other nodes! Increasing N further raises the cost necessary to obtain the same performance, both due to the price of adding more nodes, as well as requiring more connections between them: $d^\star$ must increase to 4, 5, and so on.

6. Asynchronous models

The models considered so far assume that messages from one agent to another are immediately delivered. However, a distributed environment may be subject to communication delays, for instance when some processors compute faster than others or when latency and finite bandwidth issues perturb message transmission. In the presence of such communication delays, it is conceivable that an agent will end up averaging its own value with an outdated value from another processor. Situations of this type fall within the framework of distributed asynchronous computation (Tsitsiklis et al., 1986; Bertsekas and Tsitsiklis, 1997). In the present section, we have in mind a model where agents do not have to wait at predetermined moments for predetermined messages to become available. We thus allow

some agents to compute faster and execute more iterations than others and allow communication delays to be substantial.

Communication delays are incorporated into our model as follows. For B a nonnegative integer, we assume that the last instant before t where agent j sent a message to agent i is $t - B_{ij}$, where $B_{ij} \in \{0, \dots, B\}$. Put differently, recalling that $\hat{\theta}_t^{(i)}$ is the estimate held by agent i at time t, we have

$$\hat{\theta}_{t+1}^{(i)} = \frac{1}{t+1} \sum_{j=1}^{N} a_{ij}(t - B_{ij}) \hat{\theta}_{t-B_{ij}}^{(j)} + \frac{1}{t+1} X_{t+1}^{(i)}, \quad t \geq 1. \tag{9}$$

Thus, at time t, when agent i uses the value of another agent j, this value is not necessarily the most recent one $\hat{\theta}_t^{(j)}$, but rather an outdated one $\hat{\theta}_{t-B_{ij}}^{(j)}$, where B_{ij} represents the communication delay. The time instants $t - B_{ij}$ are deterministic and, in any case, $0 \leq B_{ij} \leq B$, i.e., we assume that delays are bounded. Notice that some of the values $t - B_{ij}$ in (9) may be negative—in this case, by convention we set $\hat{\theta}_{t-B_{ij}}^{(j)} = 0$. Our goal is to establish a counterpart to Theorem 3 in the presence of communication delays. As usual, we set $\hat{\boldsymbol{\theta}}_t = (\hat{\theta}_t^{(1)}, \dots, \hat{\theta}_t^{(N)})^\top$.

Let $\kappa(t)$ be the smallest ℓ such that for all $(k_0, \dots, k_\ell) \in \{1, \dots, N\}^{\ell+1}$ satisfying $\prod_{j=1}^{\ell} a_{k_{j-1}k_j} > 0$, we have

$$t - \ell - \sum_{j=1}^{\ell} B_{k_{j-1}k_j} \leq B.$$

Observe that $t - \ell - \sum_{j=1}^{\ell} B_{k_{j-1}k_j}$ is the last time before t when a message was sent from agent k_0 to agent k_ℓ via $k_1, \dots, k_{\ell-1}$. Accordingly, $\kappa(t)$ is nothing but the smallest number of transitions needed to return at a time instant earlier than B, whatever the path. We note that $\kappa(t)$ is roughly of order t, since

$$\frac{1}{B+1} \leq \liminf_{t\to\infty} \frac{\kappa(t)}{t} \leq \limsup_{t\to\infty} \frac{\kappa(t)}{t} \leq 1.$$

From now on, it is assumed that $A = A_1$, i.e., the irreducible, aperiodic, and symmetric matrix defined in (2). Besides its simplicity, this choice is motivated by the fact that A_1 is communication-efficient while its associated performance obeys

$$\tau_t(A) \approx 1 - \frac{N^2}{6t}$$

for large t and N. The main result of the section now follows.

Theorem 7 *Assume that X is bounded and let $A = A_1$ be defined as in (2). Then, as $t \to \infty$,*

$$\mathbb{E}\left\| \frac{t}{\kappa(t)} \hat{\boldsymbol{\theta}}_t - \theta \mathbf{1} \right\|^2 = \mathrm{O}\left(\frac{1}{t}\right).$$

The advantages one hopes to gain from asynchronism are twofold. First, a reduction of the synchronization penalty and a potential speed advantage over synchronous algorithms,

perhaps at the expense of higher communication complexity. Second, a greater implementation flexibility and tolerance to system failure and uncertainty. On the other hand, the powerful result of Theorem 7 comes at the price of assumptions on the transmission network, which essentially demand that communication delays B_{ij} are time-independent. In fact, we find that the introduction of delays considerably complicates the consistency analysis of $\tau_t(A)$ even for the simple case of the empirical mean. This unexpected mathematical burden is due to the fact that the introduction of delays makes the analysis of the variance of the estimates quite complicated.

7. Conclusions and future work

This article has introduced new ideas which show how units collaborating among themselves can "boost" the statistical properties of the individual estimates by appropriately sharing information. Clearly, calculating the mean is a "simple" task with respect to current applications—our main motivation was to open a new front in this research direction. The obvious next step is to deal with more realistic problems in maximum likelihood, prediction, and learning.

As kindly pointed out by a referee, casting the problem using a single irreducible and aperiodic matrix A is a much more constrained approach than simply asking what are "good" communication schemes on a given graph and letting A be random and depend on t, i.e., A_t. In this more general case, we could have many more zeros than the graph's adjacency matrix, the case of random pairwise gossip being an example. There may be interesting choices of A_t that lead to good convergence properties. This is a promising direction for future research.

8. Proofs

We start this section by recalling the following important theorem, whose proof can be found for example in Foata and Fuchs (2004, Theorems 6.8.3 and 6.8.4). Here and elsewhere, A stands for the stochastic communication matrix.

Theorem 8 *Let $\lambda_1, \dots, \lambda_d$ be the eigenvalues of A of unit modulus (with $\lambda_1 = 1$) and Γ be the set of eigenvalues of A of modulus strictly smaller than 1.*

(*i*) *There exist projectors $Q_1, \dots, Q_d$ such that, for all $k \geq N$,*

$$A^k = \sum_{\ell=1}^{d} \lambda_\ell^k Q_\ell + \sum_{\gamma\in\Gamma} \gamma^k Q_\gamma(k),$$

where the matrices $\{Q_\gamma(k) : k \geq N, \gamma \in \Gamma\}$ satisfy $Q_\gamma(k)Q_{\gamma'}(k') = Q_\gamma(k+k')$ if $\gamma = \gamma'$, and 0 otherwise. In addition, for all $\gamma \in \Gamma$, $\lim_{k\to\infty} \gamma^k Q_\gamma(k) = 0$.

(*ii*) *The sequence $(A^k)_{k\geq 0}$ converges in the Cesàro sense to Q_1, i.e.,*

$$\frac{1}{t} \sum_{k=0}^{t} A^k \to Q_1 \quad \text{as } t \to \infty.$$

8.1 Proof of Proposition 1

According to (4), since A^k is a stochastic matrix, we have

$$\hat{\boldsymbol{\theta}}_t - \theta\mathbf{1} = \frac{1}{t}\sum_{k=0}^{t-1} A^k(\mathbf{X}_{t-k} - \theta\mathbf{1}).$$

Therefore, it may be assumed, without loss of generality, that $\theta = 0$. Thus,

$$\tau_t(A) = \frac{\mathbb{E}\left\|\bar{\mathbb{X}}_{Nt}\mathbf{1}\right\|^2}{\mathbb{E}\|\hat{\boldsymbol{\theta}}_t\|^2}.$$

Next, let $A^k = (a_{ij}^{(k)})_{1\le i,j\le N}$. Then, for each $i \in \{1,\dots,N\}$,

$$\hat{\theta}_t^{(i)} = \frac{1}{t}\sum_{k=0}^{t-1}\sum_{j=1}^{N} a_{ij}^{(k)} X_{t-k}^{(j)}, \quad t \ge 1.$$

By independence of the samples,

$$\mathbb{E}\big(\hat{\theta}_t^{(i)}\big)^2 = \frac{\sigma^2}{t^2}\sum_{k=0}^{t-1}\sum_{j=1}^{N}\big(a_{ij}^{(k)}\big)^2.$$

Upon noting that $\mathbb{E}(\bar{\mathbb{X}}_{Nt})^2 = \frac{\sigma^2}{Nt}$, we get

$$\begin{aligned}\tau_t(A) &= \frac{N\mathbb{E}\big(\bar{\mathbb{X}}_{Nt}\big)^2}{\mathbb{E}\big(\hat{\theta}_t^{(1)}\big)^2 + \cdots + \mathbb{E}\big(\hat{\theta}_t^{(N)}\big)^2}\\ &= \frac{t}{\sum_{k=0}^{t-1}\|A^k\|^2}.\end{aligned}$$

Since each A^k is a stochastic matrix, $\|A^k\|^2 \le N$ and, by the Cauchy-Schwarz inequality, $\|A^k\| \ge 1$. Thus, $\frac{1}{N} \le \tau_t(A) \le 1$, the lower bound being achieved when A is the identity matrix.

Let us now assume that A is reducible, and let $C \subsetneq \{1,\dots,N\}$ be a recurrence class. Arguing as above, we obtain that for all $i \in C$,

$$\mathbb{E}\big(\hat{\theta}_t^{(i)}\big)^2 = \frac{\sigma^2}{t^2}\sum_{k=0}^{t-1}\sum_{j=1}^{N}\big(a_{ij}^{(k)}\big)^2 \ge \frac{\sigma^2}{t^2}\sum_{k=0}^{t-1}\sum_{j\in C}\big(a_{ij}^{(k)}\big)^2.$$

Since C is a recurrence class, the restriction of A to entries in C is a stochastic matrix as well. Thus, setting $N_1 = |C|$, by the Cauchy-Schwarz inequality,

$$\mathbb{E}\big(\hat{\theta}_t^{(i)}\big)^2 \ge \begin{cases} \frac{\sigma^2}{tN_1} & \text{if } i \in C\\ \frac{\sigma^2}{tN} & \text{otherwise.}\end{cases}$$

To conclude,

$$\begin{aligned}\tau_t(A) &= \frac{\sigma^2/t}{\sum_{i\in C}\mathbb{E}\big(\hat{\theta}_t^{(i)}\big)^2 + \sum_{i\notin C}\mathbb{E}\big(\hat{\theta}_t^{(i)}\big)^2}\\ &\le \frac{1}{1+(N-N_1)/N}\\ &\le \frac{N}{N+1},\end{aligned}$$

since $N - N_1 \ge 1$.

8.2 Proof of Lemma 2

As in the previous proof, we assume that $\theta = 0$. Recall that

$$\hat{\boldsymbol{\theta}}_t = \frac{1}{t}\sum_{k=0}^{t-1} A^k\mathbf{X}_{t-k}, \quad t \ge 1.$$

Thus, for all $t \ge 1$,

$$\begin{aligned}\mathbb{E}\|\hat{\boldsymbol{\theta}}_t\|^2 &= \frac{1}{t^2}\mathbb{E}\left\|\sum_{k=0}^{t-1} A^k\mathbf{X}_{t-k}\right\|^2\\ &= \frac{1}{t^2}\sum_{k=0}^{t-1}\mathbb{E}\|A^k\mathbf{X}_{t-k}\|^2\\ &\qquad \text{(by independence of } \mathbf{X}_1,\dots,\mathbf{X}_t)\\ &= \frac{1}{t^2}\mathbb{E}\mathbf{X}_1^\top\Big(\sum_{k=0}^{t-1}(A^k)^\top A^k\Big)\mathbf{X}_1.\end{aligned}$$

Denote by $\lambda_1 = 1,\dots,\lambda_d$ the eigenvalues of A of modulus 1, and let Γ be the set of eigenvalues γ of A of modulus strictly smaller than 1. According to Theorem 8, there exist projectors $Q_1,\dots,Q_d$ and matrices $Q_\gamma(k)$ such that for all $k \ge N$,

$$A^k = \sum_{\ell=1}^{d}\lambda_\ell^k Q_\ell + \sum_{\gamma\in\Gamma}\gamma^k Q_\gamma(k).$$

Therefore,

$$\begin{aligned}\sum_{k=0}^{t-1}(A^k)^\top A^k &= \sum_{k=0}^{t-1}(\bar{A}^k)^\top A^k\\ &= \sum_{k=0}^{t-1}\left(\sum_{\ell=1}^{d}\bar{\lambda}_\ell^k\bar{Q}_\ell + \sum_{\gamma\in\Gamma}\bar{\gamma}^k\bar{Q}_\gamma(k)\right)^\top\left(\sum_{j=1}^{d}\lambda_j^k Q_j + \sum_{\gamma\in\Gamma}\gamma^k Q_\gamma(k)\right)\\ &= \sum_{k=0}^{t-1}\sum_{\ell,j=1}^{d}\bar{\lambda}_\ell^k\lambda_j^k\bar{Q}_\ell^\top Q_j + \mathrm{o}(t).\end{aligned}$$

Here, we have used Cesàro's lemma combined with the fact that, by Theorem 8, for any $\gamma \in \Gamma$, $\lim_{k\to\infty} \gamma^k Q_\gamma(k) = 0$.

Since A is irreducible, according to the Perron-Frobenius theorem (e.g., Grimmett and Stirzaker, 2001, page 240), we have that $\lambda_\ell = e^{\frac{2\pi i(\ell-1)}{d}}$, $1 \le \ell \le d$. Accordingly,

$$\bar{\lambda}_\ell \lambda_j = e^{\frac{2\pi i(j-\ell)}{d}} = 1 \Leftrightarrow j = \ell.$$

Thus,

$$\sum_{k=0}^{t-1} (A^k)^\top A^k = t \sum_{\ell=1}^{d} \bar{Q}_\ell^\top Q_\ell + \mathrm{O}(1) + \mathrm{o}(t).$$

Letting $Q = \sum_{\ell=1}^d \bar{Q}_\ell^\top Q_\ell$, we obtain

$$\begin{aligned} t\mathbb{E}\|\hat{\boldsymbol{\theta}}_t\|^2 &= \mathbb{E}\mathbf{X}_1^\top Q \mathbf{X}_1 + \mathbb{E}\mathbf{X}_1^\top \left(\frac{1}{t} \sum_{k=0}^{t-1} (A^k)^\top A^k - Q \right) \mathbf{X}_1 \qquad (10) \\ &= \mathbb{E}\mathbf{X}_1^\top Q \mathbf{X}_1 + \mathrm{o}(1) \\ &= \sum_{\ell=1}^{d} \mathbb{E}\|Q_\ell \mathbf{X}_1\|^2 + \mathrm{o}(1). \end{aligned}$$

Denoting by $Q_{\ell,ij}$ the (i,j)-entry of Q_ℓ, we conclude

$$\begin{aligned} t\mathbb{E}\|\hat{\boldsymbol{\theta}}_t\|^2 &= \sum_{\ell=1}^{d} \mathbb{E} \sum_{i=1}^{N} \left(\sum_{j=1}^{N} Q_{\ell,ij} X_1^{(j)} \right)^2 + \mathrm{o}(1) \\ &= \sigma^2 \sum_{\ell=1}^{d} \sum_{i,j=1}^{N} Q_{\ell,ij}^2 + \mathrm{o}(1) \\ &\quad \text{(by independence of } X_1^{(1)}, \ldots, X_1^{(N)}) \\ &= \sigma^2 \sum_{\ell=1}^{d} \|Q_\ell\|^2 + \mathrm{o}(1). \end{aligned}$$

Lastly, recalling that $\mathbb{E}\|\bar{\mathbb{X}}_{Nt}\mathbf{1}\|^2 = \frac{\sigma^2}{t}$, we obtain

$$\tau_t(A) = \frac{1}{\sum_{\ell=1}^d \|Q_\ell\|^2 + \mathrm{o}(1)} = \frac{1}{\sum_{\ell=1}^d \|Q_\ell\|^2} + \mathrm{o}(1).$$

8.3 Proof of Theorem 3

Sufficiency. Assume that A is irreducible, aperiodic, and bistochastic. The first two conditions imply that 1 is the unique eigenvalue of A of unit modulus. Therefore, according to Lemma 2, we only need to prove that the projector Q_1 satisfies $\|Q_1\| = 1$.

Since A is bistochastic, its stationary distribution is the uniform distribution on the set $\{1, \ldots, N\}$. Moreover, since A is irreducible and aperiodic, we have, as $k \to \infty$,

$$A^k \to \frac{1}{N} \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \ldots & 1 \end{pmatrix}.$$

By comparing this limit with that of the second statement of Theorem 8, we conclude by Cesàro's lemma that

$$Q_1 = \frac{1}{N} \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \ldots & 1 \end{pmatrix}.$$

This implies in particular that $\|Q_1\| = 1$.

Necessity. Assume that $\tau_t(A)$ tends to 1 as $t \to \infty$. According to Proposition 1, A is irreducible. Thus, by Lemma 2, we have $\sum_{\ell=1}^d \|Q_\ell\|^2 = 1$. Observe, since each Q_ℓ is a projector, that $\|Q_\ell\| \ge 1$. Therefore, the identity $\sum_{\ell=1}^d \|Q_\ell\|^2 = 1$ implies $d = 1$ and $\|Q_1\| = 1$. We conclude that A is aperiodic.

Then, since A is irreducible and aperiodic, we have, as $k \to \infty$,

$$A^k \to \begin{pmatrix} \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{\mu} \end{pmatrix},$$

where $\boldsymbol{\mu}$ is the stationary distribution of A, represented as a row vector. Comparing once again this limit with the second statement of Theorem 8, we see that

$$Q_1 = \begin{pmatrix} \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{\mu} \end{pmatrix}.$$

Thus, $\|Q_1\|^2 = N\|\boldsymbol{\mu}\|^2 = 1$. In particular, letting $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_N)$, we have

$$N \sum_{i=1}^{N} \mu_i^2 = \sum_{i=1}^{N} \mu_i.$$

This is an equality case in the Cauchy-Schwarz inequality, from which we deduce that $\boldsymbol{\mu}$ is the uniform distribution on $\{1, \ldots, N\}$. Since $\boldsymbol{\mu}$ is the stationary distribution of A, this implies that A is bistochastic.

8.4 Proof of Proposition 4

If A is irreducible and aperiodic, then by Lemma 2, $\tau_t(A) \to \frac{1}{\|Q_1\|^2}$ as $t \to \infty$. But, as $k \to \infty$,

$$A^k \to \begin{pmatrix} \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{\mu} \end{pmatrix},$$

where the stationary distribution $\boldsymbol{\mu}$ of A is represented as a row vector. By the second statement of Theorem 8, we conclude that $\|Q_1\|^2 = N\|\boldsymbol{\mu}\|^2$.

8.5 Proof of Theorem 5

Without loss of generality, assume that $\theta = 0$. Since A is irreducible and aperiodic, the matrix Q in the proof of Lemma 2 is $Q = Q_1^\top Q_1$. Moreover, since A is also bistochastic, we have already seen that as $k \to \infty$,

$$A^k \to \frac{1}{N}\begin{pmatrix} 1 & 1 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}. \tag{11}$$

However, by the second statement of Theorem 8, the above matrix is equal to Q_1. Thus, the projector Q_1 is symmetric, which implies $Q = Q_1$.

Next, we deduce from (10) that

$$\begin{aligned} \tau_t(A) &= \frac{\sigma^2}{\mathbb{E}\mathbf{X}_1^\top Q \mathbf{X}_1 + \mathbb{E}\mathbf{X}_1^\top \big(\frac{1}{t}\sum_{k=0}^{t-1}(A^k)^\top A^k - Q\big)\mathbf{X}_1} \\ &= \frac{\sigma^2}{\sigma^2 + \mathbb{E}\mathbf{X}_1^\top \big(\frac{1}{t}\sum_{k=0}^{t-1} A^{2k} - Q\big)\mathbf{X}_1}, \end{aligned} \tag{12}$$

by symmetry of A and the fact that $\mathbb{E}\mathbf{X}_1^\top Q \mathbf{X}_1 = \sigma^2$. The symmetric matrix A can be put into the form

$$A = UDU^\top,$$

where U is a unitary matrix with real entries (so, $U^\top = U^{-1}$) and $D = \mathrm{diag}(1, \gamma_2, \dots, \gamma_N)$, with $1 > \gamma_2 \geq \cdots \geq \gamma_N > -1$. Therefore, as $k \to \infty$,

$$\frac{1}{t}\sum_{k=0}^{t-1} A^{2k} = U\Big(\frac{1}{t}\sum_{k=0}^{t-1} D^{2k}\Big)U^\top \to U\begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}U^\top.$$

However, by (11) and Cesàro's lemma,

$$\frac{1}{t}\sum_{k=0}^{t-1} A^{2k} \to Q \quad \text{as } k \to \infty.$$

It follows that $Q = UMU^\top$, where

$$M = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}.$$

Thus,

$$\begin{aligned} \frac{1}{t}\sum_{k=0}^{t-1} A^{2k} - Q &= U\Big(\frac{1}{t}\sum_{k=0}^{t-1} D^{2k} - M\Big)U^\top \\ &= U\Big(\frac{1}{t}\sum_{k=0}^{t-1} \mathrm{diag}\big(0, \gamma_2^{2k}, \dots, \gamma_N^{2k}\big)\Big)U^\top \\ &= U\mathrm{diag}\Big(0, \frac{1}{t}\frac{1-\gamma_2^{2t}}{1-\gamma_2^2}, \dots, \frac{1}{t}\frac{1-\gamma_N^{2k}}{1-\gamma_N^2}\Big)U^\top. \end{aligned}$$

Next, set

$$\alpha_\ell = \frac{1}{t}\frac{1-\gamma_\ell^{2t}}{1-\gamma_\ell^2}, \quad 2 \leq \ell \leq N,$$

and let $U = (u_{ij})_{1 \leq i,j \leq N}$. With this notation, the (i,j)-entry of the matrix $\frac{1}{t}\sum_{k=0}^{t-1} A^{2k} - Q$ is

$$\sum_{\ell=2}^{N} u_{i\ell}\alpha_\ell u_{j\ell}.$$

Hence,

$$\mathbf{X}_1^\top\Big(\frac{1}{t}\sum_{k=0}^{t-1} A^{2k} - Q\Big)\mathbf{X}_1 = \sum_{i=1}^{N} X_1^{(i)} \sum_{j=1}^{N}\Big(\sum_{\ell=2}^{N} u_{i\ell}\alpha_\ell u_{j\ell}\Big)X_1^{(j)}.$$

Thus,

$$\begin{aligned} \mathbb{E}\mathbf{X}_1^\top\Big(\frac{1}{t}\sum_{k=0}^{t-1} A^{2k} - Q\Big)\mathbf{X}_1 &= \sigma^2 \sum_{i=1}^{N}\sum_{\ell=2}^{N} u_{i\ell}\alpha_\ell u_{i\ell} \\ &= \sigma^2 \sum_{i=1}^{N}\sum_{\ell=2}^{N} \alpha_\ell u_{i\ell}^2 \\ &= \sigma^2 \sum_{\ell=2}^{N} \alpha_\ell \\ &= \frac{\sigma^2}{t}\sum_{\ell=2}^{N} \frac{1-\gamma_\ell^{2t}}{1-\gamma_\ell^2}. \end{aligned}$$

We conclude from (12) that

$$\tau_t(A) = \frac{1}{1 + \frac{1}{t}\sum_{\ell=2}^{N}\frac{1-\gamma_\ell^{2t}}{1-\gamma_\ell^2}}.$$

This shows the first statement of the theorem. Using the inequality $\frac{1}{1+x} \geq 1 - x$, valid for all $x \geq 0$, we have

$$\begin{aligned} \tau_t(A) &\geq 1 - \frac{1}{t}\sum_{\ell=2}^{N}\frac{1-\gamma_\ell^{2t}}{1-\gamma_\ell^2} \\ &\geq 1 - \frac{\mathscr{S}(A)}{t}. \end{aligned}$$

Finally, evoking the inequality $\frac{1}{1+x} \leq 1 - x + x^2$, valid for all $x \geq 0$, we conclude

$$\begin{aligned}\tau_t(A) &\leq 1 - \frac{1}{t}\sum_{\ell=2}^{N} \frac{1-\gamma_\ell^{2t}}{1-\gamma_\ell^2} + \Big(\frac{1}{t}\sum_{\ell=2}^{N} \frac{1-\gamma_\ell^{2t}}{1-\gamma_\ell^2}\Big)^2 \\ &\leq 1 - \frac{\mathscr{S}(A)}{t} + \Gamma^{2t}(A)\frac{\mathscr{S}(A)}{t} + \Big(\frac{\mathscr{S}(A)}{t}\Big)^2.\end{aligned}$$

8.6 Proof of Theorem 7

From now on, we fix $k_0 \in \{1,\dots,N\}$ and let $Z_t^{(i)} = t\hat{\theta}_t^{(i)}$ for any $i \in \{1,\dots,N\}$. Thus, for all $t \geq 1$,

$$Z_t^{(k_0)} = \sum_{k=1}^{N} a_{k_0 k} Z_{t-B_{k_0 k}-1}^{(k)} + X_t^{(k_0)},$$

and

$$Z_t^{(k_0)} = \sum_{k_1,k_2=1}^{N} a_{k_0 k_1} a_{k_1 k_2} Z_{t-B_{k_0 k_1}-B_{k_1 k_2}-2}^{(k_2)} + \sum_{k_1=1}^{N} a_{k_0 k_1} X_{t-B_{k_0 k_1}-1}^{(k_1)} + X_t^{(k_0)}. \tag{13}$$

Our first task is to iterate this formula. To do so, we need additional notation. For ℓ a positive integer and $k \in \{1,\dots,N\}$, let $\underline{K}^\ell(k)$ be the set of vectors in $\{1,\dots,N\}^{\ell+1}$ of the form $(k_0, k_1, \dots, k_{\ell-1}, k)$ such that $w(\underline{K}^\ell(k)) > 0$, where

$$w\big(\underline{K}^\ell(k)\big) = a_{k_0 k_1} a_{k_1 k_2} \dots a_{k_{\ell-2} k_{\ell-1}} a_{k_{\ell-1} k}.$$

In particular, by our choice of A, we have $w(\underline{K}^\ell(k)) = 2^{-\ell}$ for any k. Next, we set

$$\Delta\big(\underline{K}^\ell(k)\big) = \ell + B_{k_0 k_1} + B_{k_1 k_2} + \dots + B_{k_{\ell-2} k_{\ell-1}} + B_{k_{\ell-1} k}.$$

When $\ell = 0$, then by convention $\underline{K}^0(k) = (k_0)$, $w(\underline{K}^0(k)) = 1$ if $k = k_0$ and 0 otherwise, and $\Delta(\underline{K}^0(k)) = 0$.

We are now ready to iterate (13). To do so, observe that

$$\begin{aligned}Z_t^{(k_0)} &= \sum_{k=1}^{N} \sum_{\underline{K}^{\kappa(t)}(k)} w\big(\underline{K}^{\kappa(t)}(k)\big) Z_{t-\Delta(\underline{K}^{\kappa(t)}(k))}^{(k)} \\ &\quad + \sum_{\ell=0}^{\kappa(t)-1} \sum_{k=1}^{N} \sum_{\underline{K}^\ell(k)} w\big(\underline{K}^\ell(k)\big) X_{t-\Delta(\underline{K}^\ell(k))}^{(k)} \\ &\stackrel{\text{def}}{=} R_t^1 + R_t^2.\end{aligned} \tag{14}$$

By the definition of $\kappa(t)$, for all $k \in \{1,\dots,N\}$, $t - \Delta(\underline{K}^{\kappa(t)}(k)) \leq B$. Since X is bounded, we deduce that there exists $C > 0$ such that

$$|R_t^1| \leq C \sum_{k=1}^{N} \sum_{\underline{K}^{\kappa(t)}(k)} w\big(\underline{K}^{\kappa(t)}(k)\big).$$

This implies that $|R_t^1| \leq C$. To see this, note that $A^{\kappa(t)}$ is a stochastic matrix and that for all $k \in \{1,\dots,N\}$,

$$\sum_{\underline{K}^{\kappa(t)}(k)} w\big(\underline{K}^{\kappa(t)}(k)\big) = (A^{\kappa(t)})_{k_0 k}.$$

The analysis of the term R_t^2 is more delicate. The difficulty arises from the fact that this term is *not* a sum of independent random variables, and therefore its components must be grouped. Since each B_{ij} is smaller than B and $\Delta(\underline{K}^\ell(k)) = x$ implies $x \geq \ell$, we obtain

$$\begin{aligned}R_t^2 &= \sum_{\ell=0}^{\kappa(t)-1} \sum_{k=1}^{N} \sum_{x=0}^{(B+1)\ell} \sum_{\underline{K}^\ell(k):\Delta(\underline{K}^\ell(k))=x} w\big(\underline{K}^\ell(k)\big)\, X_{t-x}^{(k)} \\ &= \sum_{x=0}^{(B+1)(\kappa(t)-1)} \sum_{k=1}^{N} \sum_{\ell=\lfloor x/(B+1)\rfloor+1}^{x} \sum_{\underline{K}^\ell(k):\Delta(\underline{K}^\ell(k))=x} w\big(\underline{K}^\ell(k)\big)\, X_{t-x}^{(k)}\end{aligned}$$

($\lfloor\cdot\rfloor$ is the floor function). By independence of the $X_j^{(i)}$, we get

$$\mathrm{Var}(R_t^2) = \sigma^2 \sum_{x=0}^{(B+1)(\kappa(t)-1)} \sum_{k=1}^{N} \Bigg(\sum_{\ell=\lfloor x/(B+1)\rfloor+1}^{x} \sum_{\underline{K}^\ell(k):\Delta(\underline{K}^\ell(k))=x} w\big(\underline{K}^\ell(k)\big) \Bigg)^2.$$

Recalling that $w(\underline{K}^\ell(k)) = 2^{-\ell}$, we obtain

$$\mathrm{Var}(R_t^2) = \sigma^2 \sum_{x=0}^{(B+1)(\kappa(t)-1)} \sum_{k=1}^{N} \Bigg(\sum_{\ell=\lfloor x/(B+1)\rfloor+1}^{x} \frac{1}{2^\ell} \Big|\underline{K}^\ell(k) : \Delta\big(\underline{K}^\ell(k)\big) = x\Big| \Bigg)^2.$$

Next, consider the Markov chain $(Y_n)_{n\geq 0}$ with transition matrix A such that $Y_0 = k_0$. Observe that

$$\mathbb{P}\Big(Y_\ell = k, \sum_{j=1}^{\ell} B_{Y_{j-1}Y_j} = x - \ell\Big) = \frac{1}{2^\ell} \Big|\underline{K}^\ell(k) : \Delta\big(\underline{K}^\ell(k)\big) = x\Big|.$$

Moreover, for fixed x, the events

$$\Big\{ \sum_{j=1}^{\ell} B_{Y_{j-1}Y_j} = x - \ell \Big\}, \quad \Big\lfloor \frac{x}{B+1} \Big\rfloor + 1 \leq \ell \leq x,$$

are disjoint since the B_{ij} are nonnegative. Thus,

$$\sum_{\ell=\lfloor x/(B+1)\rfloor+1}^{x} \frac{1}{2^\ell} \Big|\underline{K}^\ell(k) : \Delta\big(\underline{K}^\ell(k)\big) = x\Big| \leq 1,$$

and so,

$$\mathrm{Var}(R_t^2) \leq \sigma^2 \sum_{x=0}^{(B+1)(\kappa(t)-1)} \sum_{k=1}^{N} 1 = \sigma^2 N\big((B+1)\kappa(t) - B\big). \tag{15}$$

The expectation of R_t^2 is easier to compute. Indeed, since each A^ℓ is a stochastic matrix,

$$\mathbb{E}R_t^2 = \theta \sum_{\ell=0}^{\kappa(t)-1} \sum_{k=1}^{N} \sum_{\underline{K}^\ell(k)} w\big(\underline{K}^\ell(k)\big) = \theta \sum_{\ell=0}^{\kappa(t)-1} \sum_{k=1}^{N} (A^\ell)_{k_0 k} = \theta\kappa(t).$$

Combining (14), (15), and the fact that $|R_t^1| \leq C$, we obtain

$$\begin{aligned}\mathbb{E}\bigg(\frac{t}{\kappa(t)}\hat{\theta}_t^{(k_0)} - \theta\bigg)^2 &= \mathbb{E}\bigg(\frac{R_t^1}{\kappa(t)} + \frac{R_t^2}{\kappa(t)} - \theta\bigg)^2 \\ &= \mathbb{E}\bigg(\frac{R_t^2 - \mathbb{E}R_t^2}{\kappa(t)} + \frac{R_t^1}{\kappa(t)}\bigg)^2 \\ &= \mathrm{O}\bigg(\frac{1}{\kappa(t)}\bigg).\end{aligned}$$

The result follows from the identity $1/\kappa(t) = \mathrm{O}(1/t)$.

Acknowledgments

Gérard Biau would like to acknowledge support for this project from the Institut universitaire de France. The authors also thank the Action Editor and two referees for valuable comments and insightful suggestions, which led to a substantial improvement of the paper.

References

N. Alon. Eigenvalues and expanders. *Combinatorica*, 6:83–96, 1986.

D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, 1997.

P. Bianchi, G. Fort, W. Hachem, and J. Jakubowicz. Convergence of a distributed parameter estimator for sensor networks with local averaging of the estimates. In *Proceedings of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011a.

P. Bianchi, G. Fort, W. Hachem, and J. Jakubowicz. Performance analysis of a distributed Robbins-Monro algorithm for sensor networks. In *Proceedings of the 19th European Signal Processing Conference*, 2011b.

P. Bianchi, S. Clémençon, J. Jakubowicz, and G. Morral. On-line learning gossip algorithm in multi-agent systems with local decision rules. In *Proceedings of the 2013 IEEE International Conference on Big Data*, 2013.

V.D. Blondel, J.M. Hendrickx, A. Olshevsky, and J.N. Tsitsiklis. Convergent in multiagent coordination, consensus, and flocking. In *Proceedings of the Joint 44th IEEE Conference on Decision and Control and European Control Conference*, 2005.

C. Bordenave. A new proof of Friedman's second eigenvalue theorem and its extension to random lifts. *arXiv:1502.04482*, 2015.

S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52:2508–2530, 2006.

J.C. Duchi, A. Agarwal, and M.J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57: 592–606, 2012.

M. Fiedler. Bounds for eigenvalues of doubly stochastic matrices. *Linear Algebra and Its Applications*, 5:299–310, 1972.

D. Foata and A. Fuchs. *Processus Stochastiques : Processus de Poisson, Chaînes de Markov et Martingales*. Dunod, Paris, 2004.

P.A. Forero, A. Cano, and G.B. Giannakis. Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11:1663–1707, 2010.

J. Friedman. *A Proof of Alon's Second Eigenvalue Conjecture and Related Problems*, volume 195 of *Memoirs of the American Mathematical Society*. American Mathematical Society, Providence, 2008.

G.R. Grimmett and D.R. Stirzaker. *Probability and Random Processes. Third Edition*. Oxford University Press, Oxford, 2001.

A. Jadbabaie and A. Olshevsky. On performance of consensus protocols subject to noise: Role of hitting times and network structure. *arXiv:1508.00036*, 2015.

M.I. Jordan. On statistics, computation and scalability. *Bernoulli*, 19:1378–1390, 2013.

P.A. Knight. The Sinkhorn-Knopp algorithm: Convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 30:261–275, 2008.

A.W. Marcus, D.A. Spielman, and N. Srivastava. Interlacing families I: Bipartite Ramanujan graphs of all degrees. *Annals of Mathematics*, 182:307–325, 2015.

G. Mateos, J.A. Bazerques, and G.B. Giannakis. Distributed sparse linear regression. *IEEE Transactions on Signal Processing*, 58:5262–5276, 2010.

A. Nilli. On the second eigenvalue of a graph. *Discrete Mathematics*, 91:207–210, 1991.

A. Olshevsky and J.N. Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM Review*, 53:747–772, 2011.

J.B. Predd, S.R. Kulkarni, and H.V. Poor. A collaborative training algorithm for distributed learning. *IEEE Transactions on Automatic Control*, 55:1856–1871, 2009.

H. Shlomo, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43:439–561, 2006.

R. Sinkhorn and P. Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21:343–348, 1967.

A. Steger and N.C. Wormald. Generating random regular graphs quickly. *Combinatorics, Probability and Computing*, 8:377–396, 1999.

J.N. Tsitsiklis, D.P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31:803–812, 1986.

Y. Zhang, J.C. Duchi, and M.J. Wainwright. Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 14:3321–3363, 2013.

Journal of Machine Learning Research 17 (2016) 1-53 Submitted 7/15; Revised 10/15; Published 4/16

Convergence of an Alternating Maximization Procedure

Andreas Andresen ANDREAS.ANDRESEN@POSTEO.DE

Weierstrass-Institute,
Mohrenstr. 39,
10117 Berlin, Germany

Vladimir Spokoiny SPOKOINY@WIAS-BERLIN.DE
Weierstrass-Institute and Humboldt University Berlin,
Higher School of Economics, IITP RAN, MIPT Moscow,
Mohrenstr. 39, 10117 Berlin, Germany

Editor: Jie Peng

Abstract

We derive two convergence results for a sequential alternating maximization procedure to approximate the maximizer of random functionals such as the realized log likelihood in MLE estimation. We manage to show that the sequence attains the same deviation properties as shown for the profile M-estimator by Andresen and Spokoiny (2013), that means a finite sample Wilks and Fisher theorem. Further under slightly stronger smoothness constraints on the random functional we can show nearly linear convergence to the global maximizer if the starting point for the procedure is well chosen.

Keywords: alternating maximization, alternating minimization, profile maximum likelihood, EM-algorithm, M-estimation, local linear approximation, local concentration, semiparametric

Contents

1. Introduction

This paper presents two convergence results for an alternating maximization procedure to approximate M-estimators. Let $\mathbb{Y} \in \mathcal{Y}$ denote some observed random data, and $\mathbb{P}$ denote the data distribution. In the semiparametric profile M-estimation framework the target of analysis is

$$\boldsymbol{\theta}^* = \Pi_{\boldsymbol{\theta}} \boldsymbol{\upsilon}^* = \Pi_{\boldsymbol{\theta}} \operatorname*{argmax}_{\boldsymbol{\upsilon}} \mathbb{E}_{\mathbb{P}} \mathcal{L}(\boldsymbol{\upsilon}, \mathbb{Y}), \tag{1}$$

where $\mathcal{L} : \Upsilon \times \mathcal{Y} \to \mathbb{R}$ is an appropriate functional, $\Pi_{\boldsymbol{\theta}} : \Upsilon \to \mathbb{R}^p$ is a projection and where Υ is some high dimensional or even infinite dimensional parameter space. A prominent way of estimating $\boldsymbol{\theta}^*$ is the profile M-estimator (pME)

$$\widetilde{\boldsymbol{\theta}} \overset{\text{def}}{=} \Pi_{\boldsymbol{\theta}} \widetilde{\boldsymbol{\upsilon}} \overset{\text{def}}{=} \Pi_{\boldsymbol{\theta}} \operatorname*{argmax}_{(\boldsymbol{\theta},\boldsymbol{\eta})} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}). \tag{2}$$

This paper focuses on finite dimensional parameter spaces $\Upsilon \subseteq \mathbb{R}^{p^*}$ with $p^* = p + m \in \mathbb{N}$ being the full dimension, as infinite dimensional maximization problems are computationally not feasible. This is motivated by the sieve M-estimation technique, which projects the estimation problem to a finite dimensional submodel - see Section 1.2 for details.

The alternating maximization procedure is used in situations where a direct computation of the full maximum estimator (ME) $\widetilde{\boldsymbol{\upsilon}} \in \mathbb{R}^{p^*}$ is not feasible or simply very difficult to implement. Consider for example the task to calculate the pME where with scalar random observations $\mathbb{Y} = (y_i)_{i=1}^n \subset \mathbb{R}$, parameter $\boldsymbol{\upsilon} = (\boldsymbol{\theta}, \boldsymbol{\eta}) \in \mathbb{R}^p \times \mathbb{R}^m$ and a function basis $(\boldsymbol{e}_k) \subset L^2(\mathbb{R})$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}) = -\frac{1}{2} \sum_{i=1}^{n} \Big| y_i - \sum_{k=0}^{m} \boldsymbol{\eta}_k \boldsymbol{e}_k(\boldsymbol{X}_i^\top \boldsymbol{\theta}) \Big|^2.$$

In this case the maximization problem is high dimensional and non-convex (see Section 3 for more details). But for fixed $\boldsymbol{\theta} \in S_1 \subset \mathbb{R}^p$ maximization with respect to $\boldsymbol{\eta} \in \mathbb{R}^m$ is rather simple while for fixed $\boldsymbol{\eta} \in \mathbb{R}^m$ the maximization with respect to $\boldsymbol{\theta} \in \mathbb{R}^p$ can be feasible for low $p \in \mathbb{N}$. This motivates the following iterative procedure. Given some (data dependent) functional $\mathcal{L} : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}$ and an initial guess $\widetilde{\boldsymbol{\upsilon}}_0 \in \mathbb{R}^{p+m}$ set for $k \in \mathbb{N}$

$$\begin{aligned} \widetilde{\boldsymbol{\upsilon}}_{k,k+1} &\overset{\text{def}}{=} (\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_{k+1}) = \left(\widetilde{\boldsymbol{\theta}}_k, \operatorname*{argmax}_{\boldsymbol{\eta} \in \mathbb{R}^m} \mathcal{L}(\widetilde{\boldsymbol{\theta}}_k, \boldsymbol{\eta}) \right), \\ \widetilde{\boldsymbol{\upsilon}}_{k,k} &\overset{\text{def}}{=} (\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k) = \left(\operatorname*{argmax}_{\boldsymbol{\theta} \in \mathbb{R}^p} \mathcal{L}(\boldsymbol{\theta}, \widetilde{\boldsymbol{\eta}}_k), \widetilde{\boldsymbol{\eta}}_k \right). \end{aligned} \tag{3}$$

The so called "alternating maximization procedure" (or minimization) is a widely applied algorithm in many parameter estimation tasks (see Jain, Netrapalli, and Sanghavi, 2013; Netrapalli, Jain, and Sanghavi, 2013; Keshavan, Montanari, and Oh, 2010; Yi, Caramanis, and Sanghavi, 2013). Some natural questions arise: Does the sequence $(\widetilde{\boldsymbol{\theta}}_k)$ converge to

a limit that satisfies the same statistical properties as the profile estimator? And if the answer is yes, after how many steps does the sequence acquire these properties? Under what circumstances does the sequence actually converge to the global maximizer $\widetilde{\boldsymbol{\upsilon}}$? This problem is hard because the behavior of each step of the sequence is determined by the actual finite sample realization of the functional $\mathcal{L}(\cdot, \mathbb{Y})$. To the authors' knowledge no general "convergence" result is available that answers the questions from above except for the treatment of specific models again (see Jain, Netrapalli, and Sanghavi, 2013; Netrapalli, Jain, and Sanghavi, 2013; Keshavan, Montanari, and Oh, 2010; Yi, Caramanis, and Sanghavi, 2013) or variants of the procedure (Cheng, 2013).

We address this difficulty via employing new finite sample techniques (Andresen and Spokoiny, 2014; Spokoiny, 2012), which allow to answer the above questions: with growing iteration number $k \in \mathbb{N}$ the estimators $\widetilde{\boldsymbol{\theta}}_k$ attain the same statistical properties as the profile M-estimator and Theorem 7 provides a choice of the necessary number of steps $K \in \mathbb{N}$. Under slightly stronger conditions on the structure of the model we can give a convergence result to the global maximizier that does not rely on unimodality. Further we can address the important question under which ratio of full dimension $p^* = p + m \in \mathbb{N}$ to sample size $n \in \mathbb{N}$ the sequence behaves as desired. For instance for smooth $\mathcal{L}$ our results become sharp if $p^*/\sqrt{n}$ is small and convergence to the full maximizer already occurs if p^*/n is small.

The alternating maximization procedure can be understood as a special case of the Expectation Maximization algorithm (EM algorithm) as we illustrate in Section 1.1. The EM algorithm itself was derived in a work of Dempster, Laird, and Rubin (1977) where particular versions of this approach are generalized. This paper (Dempster, Laird, and Rubin, 1977) also contains a variety of problems where an application of the EM algorithm can be fruitful; for a brief history of the EM algorithm (see McLachlan and Krishnan, 1997, Section 1.8). We briefly explain the EM algorithm in Section 1.1.

Since the EM algorithm is very popular in applications a lot of research on its behavior has been done. We are only dealing with a special case of this procedure so we restrict our selves to citing the well-known convergence result by Wu Wu (1983), which is still state of the art in most settings. Unfortunately Wu's result - as most convergence results on these iterative procedures - only ensures convergence to some set of local maximizers or fixpoints of the procedure. Only in very special cases like unimodality can actual convergence to the maximizer be ensured.

In a recent work (Balakrishnan, Wainwright, and Yu, 2014) a new way is presented of addressing the properties of the EM sequence in a very general i.i.d. setting, based on concavity of $\mathcal{L}$. They assume that the functional $\mathcal{L}$ is concave and smooth enough (First order stability) and that for a sample $(\boldsymbol{Y}_i)_{i=1,\dots,n}$ with high probability an uniform bound is satisfied of the kind

$$\max_{\boldsymbol{\theta}^\circ \in B_r(\boldsymbol{\theta}^*)} \left| \operatorname*{argmax}_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \widetilde{\boldsymbol{\eta}}_{\boldsymbol{\theta}^\circ}) - \operatorname*{argmax}_{\boldsymbol{\theta}} \mathbb{E}\mathcal{L}(\boldsymbol{\theta}, \widetilde{\boldsymbol{\eta}}_{\boldsymbol{\theta}^\circ}) \right| \leq \epsilon_n. \tag{4}$$

Under these assumptions, with high probability and some $\nu < 1$ they show

$$\|\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*\| \leq \nu^k \|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\| + \mathsf{C}\epsilon_n. \tag{5}$$

Unfortunately this does not answer our two questions to full satisfaction. First the bound (4) is rather high level and has to be checked for each model, while we seek (and find) properties of the functional - such as smoothness and bounds on the moments of its gradient - that lead to comparably desirable behavior. Further with (5) it remains unclear whether for large $k \in \mathbb{N}$ the alternating sequence satisfies a Fisher expansion or whether a Wilks type phenomenon occurs. In particular it remains open which ratio of dimension to sample size ensures good performance of the procedure. Also the actual convergence of $\widetilde{\boldsymbol{\theta}}_k \to \widetilde{\boldsymbol{\theta}}$ is not addressed.

These results apply to our problem if the involved regularity criteria are met. But as noted these results do not tell us if the limit of the sequence $(\widetilde{\boldsymbol{\theta}}_k)$ actually is the profile and the statistical properties of limit points are not clear without too restrictive assumptions on $\mathcal{L}$ and the data.

Another new work (Cheng, 2013) contains the analysis of a slightly altered algorithm in a very general semiparametric asymptotic framework. Instead of alternatingly maximizing the functional $\mathcal{L}$ a kind of gradient decent procedure for the profile likelihood $pl(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta})$ is analyzed, i.e. they define

$$\boldsymbol{\theta}_k \stackrel{\text{def}}{=} \boldsymbol{\theta}_{k-1} + \widehat{\mathrm{D}}(\boldsymbol{\theta}_{k-1})^{-2} \ell(\boldsymbol{\theta}_{k-1}), \tag{6}$$

where $\widehat{\boldsymbol{\eta}}(\cdot)$ is an estimator of $\operatorname{argmax}_{\boldsymbol{\eta}} \mathbb{E}\mathcal{L}(\cdot, \boldsymbol{\eta})$, $\widehat{\mathrm{D}}(\cdot)$ is an estimator of $\nabla^2 \mathbb{E} \max_{\boldsymbol{\eta}} \mathcal{L}(\cdot, \boldsymbol{\eta})$ and $\ell(\cdot)$ is an estimator of $\nabla \max_{\boldsymbol{\eta}} \mathcal{L}(\cdot, \boldsymbol{\eta})$. Under common regularity conditions it is shown, that $\|\boldsymbol{\theta}_k - \widetilde{\boldsymbol{\theta}}\| = o_{\mathbb{P}}(1/\sqrt{n})$ if $k(n) \in \mathbb{N}$ is chosen such that the rate of the initial guess $\boldsymbol{\theta}_0$ - obtained via a stochastic grid search - and the rate of the estimator of the nuisance parameter are addressed. These results resemble very much what is aimed for in this work but it is important to note a series of differences between the results of that work and the present paper. First and most importantly, the treated algorithm in that paper (Cheng, 2013) is similar in virtue to the alternating procedure, but in fact is a different procedure. It is a gradient descend scheme and involves a very careful data driven choice of step sizes when carrying out the estimations necessary in (6) and in that sense differs substantially from the simple and direct alternating maximization. Also the estimating step of the nuisance component is not object of the analysis but assumed to be sufficiently good for the arguments to go through. Finally the results of (Cheng, 2013) are purely asymptotic.

In this work we carry out a finite sample analysis for the alternating maxmimization procedure in (3). Instead of a general semiparametric framework we address sieve profile estimators also called *finite dimensional linear series estimation* (see Chen, 2007; Andresen and Spokoiny, 2014), see Section 1.2 for more details. In this setting the bias of estimation - induced by projection the full model to a finite dimensional sub model - can be treated separately and the model becomes finite dimensional as far as the algorithm is concerned. This allows a very careful and explicit analysis of the behavior of the procedure. In particular the speed of convergence can be linked to characteristics of the information matrix $-\nabla^2 \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}^*)$ - namely to the constant $\nu < 1$ in (20) - and to the the full dimension of the projected parameter space. The resulting number of iterations necessary for efficient estimation can be given in a rather simple and closed form. Finally our results are nonasymptotic which in this context is crucial as a clear comparison of the computational and the estimation error for finite samples is needed for reasonable inference.

Our main result can be summarized as follows: Under a set of regularity conditions on the data and the functional $\mathcal{L}$ points of the sequence $(\widetilde{\boldsymbol{\theta}}_k)$ behave for large iteration number $k \in \mathbb{N}$ like the pME. To be more precise we show in Theorem 7 that if the initial guess $\widetilde{\boldsymbol{\upsilon}}_0 \in \Upsilon$ is good enough the step estimator sequence $(\widetilde{\boldsymbol{\theta}}_k)$ satisfies with high probability

$$\left\| \check{\mathrm{D}}\big(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*\big) - \check{\boldsymbol{\xi}} \right\|^2 \leq \epsilon(p^* + \nu^k \mathrm{R}_0), \tag{7}$$

$$\left| \max_{\boldsymbol{\eta}} \mathcal{L}(\widetilde{\boldsymbol{\theta}}_k, \boldsymbol{\eta}) - \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}^*, \boldsymbol{\eta}) - \|\check{\boldsymbol{\xi}}\|^2/2 \right| \leq (p + \mathtt{x})^{1/2} \epsilon (p^* + \nu^k \mathrm{R}_0), \tag{8}$$

where $\nu < 1$ is introduced in (20) and $\epsilon > 0$ is some small number, for example $\epsilon = \mathtt{C}/\sqrt{n}$ in the smooth i.i.d setting. Further $\mathrm{R}_0 > 0$ is a bound related to the quality of the initial guess. Generally it is proportional to the full dimension and in this way the rate with which the full nuisance can be estimated affects the speed of the convergence of the procedure. The random variable $\check{\boldsymbol{\xi}} \in \mathbb{R}^p$ and the matrix $\check{\mathrm{D}} \in \mathbb{R}^{p \times p}$ are related to the efficient influence function in semiparametric models and its covariance. These are up to $\nu^k \mathrm{R}_0$ the same properties as those proven for the pME by Andresen and Spokoiny (2014) under nearly the same set of conditions. Up to the finite sample bounds on the right hand sides this means that the estimating points of the procedure admit a Fisher expansion - in other words are asymptotical normal - and a Wilks expansion. Consequently the usual inference procedures based on confidence and concentration sets can be applied to these estimators. Further in our second main result we manage to show under slightly stronger smoothness conditions that $(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k)$ approaches the ME $(\widetilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{\eta}}) = \operatorname{argmax} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}^*)$ with nearly linear convergence speed, i.e. $\|\mathcal{D}((\boldsymbol{\theta}_k, \boldsymbol{\eta}_k) - \widetilde{\boldsymbol{\upsilon}})\| \leq \tau^{k/\log(k)}$ with some $0 < \tau < 1$ and $\mathcal{D}^2 = -\mathbb{E}\nabla^2 \mathcal{L}(\boldsymbol{\upsilon}^*)$ (see Theorem 14).

To clarify we want to mention that the term convergence refers to the behavior of the sequence $(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k)$ when the number of iterations $k \in \mathbb{N}$ tends to infinity. We show $(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k) \to (\widetilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{\eta}})$. This has to be distinguished from the usual stochastic convergence results of the M-estimator $(\widetilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{\eta}})$ towards the target $(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*)$ or the weak convergence to a normal distribution as the sample size increases. Our setup is assuming finite sample size such that even with $k \to \infty$ there remains a gap between $(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k)$ and $(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*)$ and between $\widetilde{\boldsymbol{\theta}}_k$ and $\boldsymbol{\theta}^*$ that is related to the parametric and semiparametric Cramer-Rao lower bounds respectively. This is why we can in the finite sample setting only hope to obtain convergence of the alternating procedure to $(\widetilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{\eta}})$ but not to $(\boldsymbol{\theta}^*, \boldsymbol{\eta}^*)$. But for a growing sample size (7) implies the weak convergence results also for the estimator $\widetilde{\boldsymbol{\theta}}_k$ when $k(n) \in \mathbb{N}$ is large enough and ϵp^* vanishes (see Section 1.3).

In the following we write $\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}$ in statements that are true for both $\widetilde{\boldsymbol{\upsilon}}_{k,k+1}$ and $\widetilde{\boldsymbol{\upsilon}}_{k,k}$. Also we do not specify whether the elements of the resulting sequence are sets or single points. All statements made about properties of $\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}$ are to be understood in the sense that they hold for "every point of $\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}$".

1.1 Relation to the EM Algorithm

In the introduction we claimed that the alternating procedure analyzed in this work is related to the EM algorithm. In this section we want to elaborate on that.

First we explain the EM algortihm. Consider data $(\mathbb{X}) \sim \mathbb{P}_{\boldsymbol{\theta}}$ for some parametric family $(\mathbb{P}_{\boldsymbol{\theta}}, \boldsymbol{\theta} \in \Theta)$. Assume that a parameter $\boldsymbol{\theta} \in \Theta$ is to be estimated as maximizer of the functional $\mathcal{L}_c(\boldsymbol{\theta}, \mathbb{X}) \in \mathbb{R}$, but that only $\mathbb{Y} \in \mathcal{Y}$ is observed, where $\mathbb{Y} = f_Y(\mathbb{X})$ is the image of the complete data set $\mathbb{X} \in \mathcal{X}$ under some map $f_Y : \mathcal{X} \to \mathcal{Y}$. Prominent examples for the map f_Y are projections onto subspaces of $\mathcal{X}$ if both $\mathcal{Y}, \mathcal{X}$ are vector spaces. The information lost under the map can be regarded as missing data or latent variables. As a direct maximization of the functional is impossible without knowledge of $\mathbb{X}$ the EM algorithm serves as a workaround. It consists of the iteration of two steps: starting with some initial guess $\widetilde{\boldsymbol{\theta}}_0$ the kth "Expectation step" derives the functional Q via

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_k) = \mathbb{E}_{\boldsymbol{\theta}_k}[\mathcal{L}_c(\boldsymbol{\theta}, \mathbb{X})|\mathbb{Y}],$$

which means that on the right hand side the conditional expectation is calculated under the distribution $\mathbb{P}_{\boldsymbol{\theta}_k}$. The kth "Maximation step" then simply locates the maximizer $\boldsymbol{\theta}_{k+1}$ of Q.

Now we can present the convergence result of Wu (1983) in more detail. Wu presents regularity conditions that ensure that $\mathcal{L}(\boldsymbol{\theta}_{k+1}|\mathbb{Y}) \geq \mathcal{L}(\boldsymbol{\theta}_k|\mathbb{Y})$ where

$$\mathcal{L}(\boldsymbol{\theta}|\mathbb{Y}) \stackrel{\text{def}}{=} \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\theta}, \mathbb{X})|\mathbb{Y} = f_Y(\mathbb{X})\right],$$

such that $\mathcal{L}(\boldsymbol{\theta}_k|\mathbb{Y}) \to \mathcal{L}^*$ for some limit value $\mathcal{L}^*(\mathbb{Y}) > 0$, that may depend on the starting point $\boldsymbol{\theta}_0$. Additionally Wu gives conditions that guarantee that the sequence $\boldsymbol{\theta}_k$ (possibly a sequence of sets) converges to $C(\mathcal{L}^*) \stackrel{\text{def}}{=} \{\boldsymbol{\theta}|\, \mathcal{L}(\boldsymbol{\theta}|\mathbb{Y}) = \mathcal{L}^*(\mathbb{Y})\}$. Dempster, Laird, and Rubin (1977) show that the speed of convergence is linear in the case of point valued $\boldsymbol{\theta}_k$ and of some differentiability criterion being met. A limitation of these results is that it is not clear whether $\mathcal{L}^*(\mathbb{Y}) = \sup \mathcal{L}(\boldsymbol{\theta}|\mathbb{Y})$ and thus it is not guaranteed that $C(\mathcal{L}^*)$ is the desired MLE and not just some local maximum. Of course this problem disappears if $\mathcal{L}(\cdot|\mathbb{Y})$ is unimodal and the regularity conditions are met but this assumption may be too restrictive.

To see that the procedure (3) is a special case of the EM algorithm we have to find the right triplet $(\mathbb{X}, f_Y, \mathcal{L}_c)$. For this we take $\mathbb{X} = (\boldsymbol{Z}, \mathbb{Y})$ with $\boldsymbol{Z} \sim \operatorname{argmax}_{\boldsymbol{\eta}} \mathcal{L}\{(\boldsymbol{\theta}, \boldsymbol{\eta}), \mathbb{Y}\}$ under $\mathbb{P}_{\boldsymbol{\theta}}$. Further we set $f_Y(\mathbb{X}) = \mathbb{Y}$ and $\mathcal{L}_c(\boldsymbol{\theta}, \boldsymbol{X}) \stackrel{\text{def}}{=} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbb{Y})$, where $\boldsymbol{X} = (\boldsymbol{\eta}, \mathbb{Y})$. Then we find

$$\begin{aligned}
Q(\boldsymbol{\theta}, \widetilde{\boldsymbol{\theta}}^{(k-1)}) &= \mathbb{E}_{\widetilde{\boldsymbol{\theta}}^{(k-1)}}[\mathcal{L}_c(\boldsymbol{\theta}, \mathbb{X})|\mathbb{Y}] \\
&= \mathbb{E}_{\widetilde{\boldsymbol{\theta}}^{(k-1)}}\Big[\mathcal{L}_c\Big(\boldsymbol{\theta}, \operatorname*{argmax}_{\boldsymbol{\eta}} \mathcal{L}\{(\boldsymbol{\theta}^{(k-1)}, \boldsymbol{\eta}), \mathbb{Y}\}, \mathbb{Y}\Big)\Big|\mathbb{Y}\Big] \\
&= \mathcal{L}_c\Big(\boldsymbol{\theta}, \operatorname*{argmax}_{\boldsymbol{\eta}} \mathcal{L}\{(\widetilde{\boldsymbol{\theta}}^{(k-1)}, \boldsymbol{\eta}), \mathbb{Y}\}, \mathbb{Y}\Big) \\
&= \mathcal{L}(\boldsymbol{\theta}, \widetilde{\boldsymbol{\eta}}^{(k)}, \mathbb{Y}),
\end{aligned}$$

and thus the resulting sequence is the same as in (3).

1.2 Linear Series Estimators

In semiparametric models the profile M estimator $\widetilde{\boldsymbol{\theta}} \in \mathbb{R}^p$ from Equation (2) cannot be calculated in practice if the full model is infinite dimensional. There are various ways to circumvent this problem. Next to non parametric estimation and plugin of the nuisance $\boldsymbol{\eta} \in \mathcal{X}$ a prominent approach is the so called sieve technique that motivates the setting in this work.

The sieve approach was systematically introduced by Grenander (see Grenander, 1981, Chapter 8) and consists in choosing a suitable sequence of subsets $(\Upsilon_m)_{m=1}^{\infty} \subset \Upsilon$ such that for each $\boldsymbol{\upsilon} \in \Upsilon$ there exists a sequence $\Pi_m(\boldsymbol{\upsilon}) \subset \Upsilon_m$ with $\|\boldsymbol{\upsilon} - \Pi_m(\boldsymbol{\upsilon})\| \to 0$. Furthermore the sets $\Upsilon_m \subset \Upsilon$ have to be such that $\sup_{\boldsymbol{\upsilon} \in \Upsilon_m} \mathcal{L}(\boldsymbol{\upsilon})$ can be calculated in practice. In the setting of semiparametric M-Estimation we assume $\Upsilon = \Upsilon_{\boldsymbol{\theta}} \times \Upsilon_{\boldsymbol{\eta}} \subseteq \mathbb{R}^p \times \mathcal{X}$ with some infinite dimensional separable Hilbert space $\mathcal{X}$ and countable basis $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots\} \subset \mathcal{X}$. We set $\Upsilon_m = \Upsilon_{\boldsymbol{\theta}} \times \Pi_m \Upsilon_{\boldsymbol{\eta}}$, where $\Pi_m : \mathcal{X} \to \mathcal{X}_m$ denotes the orthogonal projection onto $\mathcal{X}_m \stackrel{\text{def}}{=} \operatorname{span}(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\}$. For each $m \in \mathbb{N}$ the sieve profile M-estimator is defined as

$$\widetilde{\boldsymbol{\theta}}_m \stackrel{\text{def}}{=} \Pi_{\boldsymbol{\theta}} \widetilde{\boldsymbol{\upsilon}}_m \stackrel{\text{def}}{=} \Pi_{\boldsymbol{\theta}} \operatorname*{argmax}_{\substack{\boldsymbol{\theta} \in \mathbb{R}^p \\ \boldsymbol{\eta} \in \mathbb{R}^m}} \mathcal{L}\left(\boldsymbol{\theta}, \sum_{k=1}^{m} \eta_k \boldsymbol{e}_k\right).$$

This means that for the calculation of the estimator $\widetilde{\boldsymbol{\theta}}_m$ only a finite dimensional setting has to be considered. In our analysis we will focus on the behavior of the alternating maximization procedure in that case.

But of course the projection onto a finite dimensional submodel induces an approximation bias " $\boldsymbol{\upsilon}^* - \boldsymbol{\upsilon}_m^*$ " where

$$(\boldsymbol{\theta}_m^*, \boldsymbol{\eta}_m^*) \stackrel{\text{def}}{=} \boldsymbol{\upsilon}_m^* \stackrel{\text{def}}{=} \operatorname*{argmax}_{\substack{\boldsymbol{\theta} \in \mathbb{R}^p \\ \boldsymbol{\eta} \in \mathbb{R}^m}} \mathbb{E}\mathcal{L}\left(\boldsymbol{\theta}, \sum_{k=1}^{m} \eta_k \boldsymbol{e}_k\right).$$

In (Andresen and Spokoiny, 2014) it is explained in detail how this bias can be treated. Once the bias is controlled this leads for each $m \in \mathbb{N}$ to bounds of the kind

$$\begin{aligned}
\left\|\breve{\mathrm{D}}(\widetilde{\boldsymbol{\theta}}_m - \boldsymbol{\theta}^*) - \breve{\boldsymbol{\xi}}_m\right\| &\leq \breve{\Diamond}(\mathtt{x}) + \alpha(m), \\
\left|\max_{\boldsymbol{\eta} \in \Pi_m \Upsilon_{\boldsymbol{\eta}}} \mathcal{L}(\widetilde{\boldsymbol{\theta}}_m, \boldsymbol{\eta}) - \max_{\boldsymbol{\eta} \in \Pi_m \Upsilon_{\boldsymbol{\eta}}} \mathcal{L}(\boldsymbol{\theta}^*, \boldsymbol{\eta}) - \|\breve{\boldsymbol{\xi}}_m\|^2\right| &\leq p\breve{\Diamond}(\mathtt{x}) + \alpha(m),
\end{aligned}$$

where $\alpha(m) \geq 0$ quantifies the impact of the bias " $\boldsymbol{\upsilon}^* - \boldsymbol{\upsilon}_m^*$ ". The choice of $m \in \mathbb{N}$ then has to balance the two terms $\breve{\Diamond}(\mathtt{x})$ and $\alpha(m)$, which leads to common optimal choices for the dimension based on the "smoothness" of the nuisance component $\boldsymbol{\eta}^* \in \mathcal{X}$. To ease notation we drop the $\cdot_m$ in the following, as the treatment of the bias can be done separately, (see Andresen and Spokoiny, 2014).

1.3 Finite sample Wilks and Fisher Theorems

Before we present our main results we want to explain what type of results we aim at and how they can be interpreted. Hopefully this will ease the understanding and will make some of the apparently cumbersome notation more intelligible.

Usually in asymptotic treatments of semiparametric M-estimators like $\widetilde{\boldsymbol{\theta}}$ in (2) the aim is to derive statements of the kind

$$\sqrt{n}(\widetilde{\boldsymbol{\theta}} - \boldsymbol{\theta}^*) - \breve{d}^{-1}\breve{\boldsymbol{\xi}} = o_{\mathbb{P}}(1), \tag{9}$$

$$\max_{\boldsymbol{\eta}} \mathcal{L}(\widetilde{\boldsymbol{\theta}}, \boldsymbol{\eta}) - \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}^*, \boldsymbol{\eta}) - \|\breve{\boldsymbol{\xi}}\|^2/2 = o_{\mathbb{P}}(1), \tag{10}$$

$$\breve{\boldsymbol{\xi}} \xrightarrow{w} \mathcal{N}(0, \breve{d}^{-1}\breve{v}^2\breve{d}^{-1}),$$

where $n \in \mathbb{N}$ denotes the sample size. The random variable $\breve{\boldsymbol{\xi}} \in \mathbb{R}^p$ is called semiparametric score. Below we will briefly explain its derivation along with the explanation of the matrices $\breve{v}^2, \breve{d}^2 \in \mathbb{R}^{p\times p}$. But before, we sketch how (9) and (10) can be used for the construction of asymptotic confidence sets that yield statistical tests. Given the matrices $\breve{v}^2, \breve{d}^2$ the construction works as follows. Let $q_\alpha^2 > 0$ be an $\alpha-$ level quantile of a $\chi^2_p(\breve{d}^{-2}\breve{v}^2\breve{d}^{-2})$-distribution. Set

$$\mathcal{E}(q_\alpha) = \left\{\boldsymbol{\theta} : \sqrt{n}\|(\widetilde{\boldsymbol{\theta}} - \boldsymbol{\theta})\| \le q_\alpha\right\}; \tag{11}$$

then one can use (9) to show

$$\mathbb{P}\left\{\boldsymbol{\theta}^* \notin \mathcal{E}\left(q_\alpha\right)\right\} = \mathbb{P}\left\{\sqrt{n}\|(\widetilde{\boldsymbol{\theta}} - \boldsymbol{\theta}^*)\| \ge q_\alpha\right\} \to 1 - \alpha.$$

Similarly one can exploit (10).

Now we explain the definition of $\breve{v}^2$ and $\breve{d}^2$. Introduce

$$n\breve{v}^{-2}(\boldsymbol{\upsilon}) \stackrel{\text{def}}{=} \Pi_{\boldsymbol{\theta}} \operatorname{Cov}(\nabla\mathcal{L}(\boldsymbol{\upsilon}))^{-1}\Pi_{\boldsymbol{\theta}}^\top,$$
$$n\breve{d}^{-2}(\boldsymbol{\upsilon}) \stackrel{\text{def}}{=} -\Pi_{\boldsymbol{\theta}} \left(\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})\right)^{-1}\Pi_{\boldsymbol{\theta}}^\top,$$

where $\Pi_{\boldsymbol{\theta}}$ is the orthogonal projection onto the $\boldsymbol{\theta}$-commponent in $\mathbb{R}^p$ and $\Pi_{\boldsymbol{\theta}}^\top$ is its dual operator. Then $\breve{v}^2 = \breve{v}^2(\boldsymbol{\upsilon}^*)$ and $\breve{d}^2 = \breve{d}^2(\boldsymbol{\upsilon}^*)$.

Remark 1 *Note that these two matrices coincide if the functional $\mathcal{L}$ was the complete loglikelihood of the observations and that then $\breve{d}^2$ would equal the covariance of the efficient influence function (see Kosorok, 2005, for more details).*

For the definition of the semiparametric score $\breve{\boldsymbol{\xi}} \in \mathbb{R}^p$ consider

$$nd^2(\boldsymbol{\upsilon}) = \begin{pmatrix} d^2(\boldsymbol{\upsilon}) & a(\boldsymbol{\upsilon}) \\ a^\top(\boldsymbol{\upsilon}) & h^2(\boldsymbol{\upsilon}) \end{pmatrix} \stackrel{\text{def}}{=} -\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}) \in \mathbb{R}^{p^*\times p^*}.$$

Then

$$\begin{aligned}\breve{\boldsymbol{\xi}} &\stackrel{\text{def}}{=} \frac{1}{\sqrt{n}}\breve{d}(\boldsymbol{\upsilon}^*)(1 - \mathbb{E}_\epsilon)\Pi_{\boldsymbol{\theta}} d^{-2}(\boldsymbol{\upsilon}^*)\nabla\mathcal{L}(\boldsymbol{\upsilon}^*) \\ &= \frac{1}{\sqrt{n}}(1 - \mathbb{E}_\epsilon)\breve{d}^{-1}(\boldsymbol{\upsilon}^*)\left\{\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*) - ah^{-2}(\boldsymbol{\upsilon}^*)\nabla_{\boldsymbol{\eta}}\mathcal{L}(\boldsymbol{\upsilon}^*)\right\},\end{aligned}$$

This random variable is related to the efficient influence function in semiparametric estimation and it plays the role that the usual score $\nabla\mathcal{L}(\boldsymbol{\upsilon}^*)$ plays in the setting of parametric M-estimation.

In this work we derive (9) and (10) for $\widetilde{\boldsymbol{\theta}}_k$ instead of $\widetilde{\boldsymbol{\theta}}$ but with finite sample bounds for the terms on the right-hand sides of (9) and (10). To be more precise we derive statements of the following kind. With probability greater than $1 - \mathtt{C}\mathrm{e}^{-\mathtt{x}}$

$$\left\|\sqrt{n}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) - \breve{d}^{-1}\breve{\boldsymbol{\xi}}\right\| \le \epsilon(p^* + \nu^k\mathrm{R}_0), \tag{12}$$

$$\left|\max_{\boldsymbol{\eta}} \mathcal{L}(\widetilde{\boldsymbol{\theta}}_k, \boldsymbol{\eta}) - \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}^*, \boldsymbol{\eta}) - \|\breve{\boldsymbol{\xi}}\|^2/2\right| = (p + \mathtt{x})^{1/2}\epsilon(p^* + \nu^k\mathrm{R}_0), \tag{13}$$

with some small value $\epsilon > 0$ as in (7) and (8). Note that for vanishing right hand sides these equations imply (9) and (10) when n tends to infty. Using the scheme in (11) the bounds (12) and (13) allow the construction of (conservative) finite sample "confidence sets". Assume that (approximate) quantiles q_α for $\|\breve{\boldsymbol{\xi}}\|$ are available, i.e. that with some small $\epsilon > 0$ and any $\alpha \in [0, 1]$

$$\mathbb{P}(\|\breve{\boldsymbol{\xi}}\| \le q_\alpha) \in (\alpha - \delta, \alpha + \delta),$$

then with some generic constant $\mathtt{C} > 0$ (see Andresen and Spokoiny, 2014, Remark 2.13)

$$\alpha + \delta + \mathtt{C}\mathrm{e}^{-\mathtt{x}} \le \mathbb{P}\left\{\boldsymbol{\theta}^* \in \mathcal{E}(q_\alpha + \epsilon(p^* + \nu^k\mathrm{R}_0)\right\},$$
$$\mathbb{P}\left\{\boldsymbol{\theta}^* \in \mathcal{E}(q_\alpha - \epsilon(p^* + \nu^k\mathrm{R}_0)\right\} \le \alpha - \delta - \mathtt{C}\mathrm{e}^{-\mathtt{x}}.$$

The important achievement is that one can make approximate confidence statements for the estimators of the alternating procedure and this even in the finite sample case, without ignoring "hopefully small enough" terms. As remarked above such approximate quantiles could be attained via an plug-in-estimation of $\breve{d}^{-2}\breve{v}^2\breve{d}^{-2}$ combined with a Gaussian approximation or a bootstrap.

2. Main Results

This Section contains the thorough presentation of our main convergence results. It involves the introduction of various technicalities and may appear a bit cumbersome on first read. We recommend to carefully read Section 1.3 first to ease understanding.

2.1 Conditions

This section collects the conditions imposed on the model. We use the same set of assumptions as Andresen and Spokoiny (2014) and this section closely follows Section 2.1 of that paper.

Let the full dimension of the paramter space be finite, i.e. $p^* < \infty$. Our conditions involve the symmetric positive definite *information matrix* $\mathcal{D}_0^2 \in \mathbb{R}^{p^* \times p^*}$ and a central point $\boldsymbol{\upsilon}^* \in \mathbb{R}^{p^*}$. To ease presentation in this paper we identify $\boldsymbol{\upsilon}^*$ with the "true point" from (1) and define

$$\mathcal{D}_0^2 \stackrel{\text{def}}{=} -\nabla^2 \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}^*),$$

where we assume, that the second derivative exists. In the context of semiparametric estimation, it is convenient to represent the *information matrix* in block form:

$$\mathcal{D}_0^2 = \begin{pmatrix} \mathrm{D}_0^2 & \mathrm{A}_0 \\ \mathrm{A}_0^\top & \mathrm{H}_0^2 \end{pmatrix}.$$

First we state an *identifiability condition*, which basically imposes that $\mathcal{D}^2$ is positive definite. Note that in this work $\|\cdot\|$ allways denotes the spectral norm when its argument is a matrix.

$(\boldsymbol{\mathcal{I}})$ It holds for some $\nu < 1$

$$\|\mathrm{H}_0^{-1}\mathrm{A}_0^\top \mathrm{D}_0^{-1}\| \leq \sqrt{\nu}. \tag{14}$$

The condition $(\mathcal{I})$ allows to introduce the important $p \times p$ efficient information matrix $\breve{\mathrm{D}}_0^2$ which is defined as the inverse of the $\boldsymbol{\theta}$-block of the inverse of the full dimensional matrix $\mathcal{D}_0^2$. The exact formula is given by

$$\breve{\mathrm{D}}_0^2 \stackrel{\text{def}}{=} \mathrm{D}_0^2 - \mathrm{A}_0 \mathrm{H}^{-2} \mathrm{A}_0^\top,$$

and $(\mathcal{I})$ ensures that the matrix $\breve{\mathrm{D}}_0^2$ is positive definite such that $\breve{\mathrm{D}}$ is well defined. At the same time $\nu < 1$ ensures that the alternating sequence actually converges. As can be seen in Theorem 7 the speed of convergence is linear in ν.

Using the matrix $\mathcal{D}_0^2$ and the central point $\boldsymbol{\upsilon}^* \in \mathbb{R}^{p^*}$, we define the local set $\Upsilon_\circ(\mathtt{r}) \subset \Upsilon \subseteq \mathbb{R}^{p^*}$ with some $\mathtt{r} \geq 0$:

$$\Upsilon_\circ(\mathtt{r}) \stackrel{\text{def}}{=} \bigl\{\boldsymbol{\upsilon} = (\boldsymbol{\theta}, \boldsymbol{\eta}) \in \Upsilon \colon \|\mathcal{D}_0(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*)\| \leq \mathtt{r}\bigr\}. \tag{15}$$

2.1.1 Smoothness

Usually in the context of regular M-estimation one assumes local quadraticity, i.e. that

$$pl(\boldsymbol{\theta}) = \nabla pl(\boldsymbol{\theta}^*)(\boldsymbol{\theta} - \boldsymbol{\theta}^*) + \frac{1}{2}(\boldsymbol{\theta}^* - \boldsymbol{\theta}^*)^\perp \nabla^2 pl(\boldsymbol{\theta}^*)(\boldsymbol{\theta}^* - \boldsymbol{\theta}^*) + o_{\mathbb{P}}(1),$$

for $\boldsymbol{\theta} \in \mathbb{R}^p$ close enough to $\boldsymbol{\theta}^*$. The functional $pl(\cdot)$ in this context denotes the profile-functional and is defined as

$$pl(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}),$$

see for instance (Cheng, 2013).

In our setting we need a precise bound for the accuracy of a quadratic approximation of the expected profile functional $\mathbb{E}pl$. We want to bound the error of a local linear approximation of the projected gradient $\breve{\nabla}_{\boldsymbol{\theta}} \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})$ which is defined as

$$\breve{\nabla}_{\boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} - \mathrm{A}_0 \mathrm{H}_0^{-2} \nabla_{\boldsymbol{\eta}}.$$

Instead of local quadraticity of the profile functional we impose that for $(\boldsymbol{\theta}, \boldsymbol{\eta}) = \boldsymbol{\upsilon} \in \Upsilon_\circ(\mathtt{r})$ - with the local set $\Upsilon_\circ(\mathtt{r})$ defined in (15) - with some small $\breve{\epsilon} > 0$

$$\Bigl\| \breve{\mathrm{D}}^{-1} \Bigl(\breve{\nabla} \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}) - \breve{\nabla} \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}^*) \Bigr) - \breve{\mathrm{D}}(\boldsymbol{\theta} - \boldsymbol{\theta}^*) \Bigr\| \leq \breve{\epsilon} \mathtt{r}^2.$$

The following condition serves a less high level way of checking that such an approximation holds (see Andresen and Spokoiny, 2014, Lemma B.1)

$(\boldsymbol{\breve{\mathcal{L}}_0})$ For each $\mathtt{r} \leq \mathtt{r}_0$, there is a constant $\breve{\epsilon} > 0$ such that it holds on the set $\Upsilon_\circ(\mathtt{r})$:

$$\|\mathrm{D}^{-1}\mathrm{D}^2(\boldsymbol{\upsilon})\mathrm{D}^{-1} - I_p\| \leq \breve{\epsilon}\mathtt{r}, \quad \|\mathrm{D}^{-1}(\mathrm{A}(\boldsymbol{\upsilon}) - \mathrm{A})\mathrm{H}^{-1}\| \leq \breve{\epsilon}\mathtt{r},$$
$$\bigl\|\mathrm{D}^{-1}\mathrm{A}\mathrm{H}^{-1}\bigl(I_m - \mathrm{H}^{-1}\mathrm{H}^2(\boldsymbol{\upsilon})\mathrm{H}^{-1}\bigr)\bigr\| \leq \breve{\epsilon}\mathtt{r}.$$

If $\mathbb{E}\mathcal{L}$ is three times continuously differentiable one obtains $\breve{\epsilon} \leq \mathtt{C}\|\mathcal{D}^{-1}\|$. In i.i.d. models one usually has $\|\mathcal{D}^{-1}\| = O(1/\sqrt{n})$.

Remark 2 *Here and in what follows we implicitly assume that the function $\mathcal{L}(\boldsymbol{\upsilon}) \colon \mathbb{R}^{p^*} \to \mathbb{R}$ is sufficiently smooth in $\boldsymbol{\upsilon} \in \mathbb{R}^{p^*}$, $\nabla\mathcal{L}(\boldsymbol{\upsilon}) \in \mathbb{R}^{p^*}$ stands for the gradient and $\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}) \in \mathbb{R}^{p^* \times p^*}$ for the Hessian of the expectation $\mathbb{E}\mathcal{L} : \mathbb{R}^{p^*} \to \mathbb{R}$ at $\boldsymbol{\upsilon} \in \mathbb{R}^{p^*}$. By smooth enough we mean that we can interchange $\nabla\mathbb{E}\mathcal{L} = \mathbb{E}\nabla\mathcal{L}$ on $\Upsilon_\circ(R_0)$, where $\Upsilon_\circ(\mathtt{r})$ is defined in (15) and $R_0 > 0$ in (19). It is worth mentioning that $\mathcal{D}_0^2 = \mathcal{V}_0^2 \stackrel{\text{def}}{=} \mathrm{Cov}(\nabla\mathcal{L}(\boldsymbol{\upsilon}^*))$ if the model $\boldsymbol{Y} \sim \mathbb{P}_{\boldsymbol{\upsilon}^*} \in (\mathbb{P}_{\boldsymbol{\upsilon}})$ is correctly specified and sufficiently regular; see e.g. (Ibragimov and Khas'minskij, 1981).*

2.1.2 Complexity

The usual approach to gain asymptotic control on profile M-estimators would be to assume that the class

$$\{\breve{\nabla}\mathcal{L}(\boldsymbol{\upsilon}), \boldsymbol{\upsilon} \in \Upsilon_\circ(\mathtt{r})\},$$

is $\mathbb{P}$-Donsker (see Cheng, 2013).

To pin down the estimator sequence $(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k)_{k \in \mathbb{N}}$ and to obtain finite sample results we use a more specific approach, which is based on a new finite sample approach (Spokoiny,

2012; Andresen and Spokoiny, 2014). First note that we assume that - as far as the alternating procedure is concerned - the model is finite dimensional, which means that $\Upsilon_\circ(\mathtt{r}) \subset \Upsilon \subset \mathbb{R}^{p^*}$ in (15) is automatically compact for finite radius $\mathtt{r} > 0$. If we can ensure the right smoothness and moment conditions on $\check{\nabla}\mathcal{L}(\boldsymbol{\upsilon})$, we automatically obtain that the above class is $\mathbb{P}$-Donsker. But using the new techniques (Spokoiny, 2012; Andresen and Spokoiny, 2014) we manage to obtain slightly stronger bounds that are useful in a finite sample setting.

To understand the next condition consider first the definition of a subgaussian random vector. A random vector $\boldsymbol{X} \in \mathbb{R}^p$ is called subgaussian, if for any $\mu \in \mathbb{R}$ and some $\boldsymbol{\nu} > 0$

$$\sup_{\|\gamma\|\le 1} \log \mathbb{E} \exp\Big\{\mu \gamma^\top \boldsymbol{X}\Big\} \le \boldsymbol{\nu}^2 \mu^2/2. \tag{16}$$

Obviously this is a strong condition. As it turns out in many situations it is sufficient to assume subexponentiality, which simply relaxes subgaussianity to demanding that (16) is met for any $|\mu| \le \mathtt{g}$ with some $\mathtt{g} > 0$. In this way many distributions that would be excluded by assuming subgaussianity can still be treated.

Our next condition combines subexponentiality with a smoothness constraint on the stochastic component of $\check{\nabla}\mathcal{L}(\boldsymbol{\upsilon})$. It assumes that - with some $\check{\epsilon} > 0$ - the random vector

$$\frac{1}{\check{\epsilon}\|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}')\|} \check{\mathrm{D}}^{-1}\big\{\check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}')\big\} \in \mathbb{R}^p,$$

is uniformly subexponential for $\boldsymbol{\upsilon}, \boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})$, with the stochastic component defined as $\zeta(\boldsymbol{\upsilon}) \stackrel{\text{def}}{=} \mathcal{L}(\boldsymbol{\upsilon}) - \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})$. It reads:

$(\check{\mathcal{E}}\mathcal{D}_1)$ For all $0 < \mathtt{r} < \mathtt{r}_0$, there exists a constant $\check{\epsilon} \le 1/2$ such that for all $|\mu| \le \check{\mathtt{g}}$ and $\boldsymbol{\upsilon}, \boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})$

$$\sup_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})} \sup_{\|\gamma\|\le 1} \log \mathbb{E} \exp\left\{\mu \frac{\gamma^\top \check{\mathrm{D}}^{-1}\big\{\check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}')\big\}}{\check{\epsilon}\|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}')\|}\right\} \le \frac{\check{\nu}_1^2 \mu^2}{2}.$$

To convey more intuition consider for some pair $\boldsymbol{\upsilon}, \boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})$ the function

$$\psi_\gamma(t) \stackrel{\text{def}}{=} \gamma^\top \check{\mathrm{D}}^{-1}\big\{\check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon} + t(\boldsymbol{\upsilon}' - \boldsymbol{\upsilon}))\big\}.$$

Then $(\check{\mathcal{E}}\mathcal{D}_1)$ is met with $\check{\epsilon} \le \|\mathcal{D}^{-1}\|$, if - uniformly in γ - for any pair $\boldsymbol{\upsilon}, \boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})$ the corresponding function $\psi_\gamma : [0,1] \to \mathbb{R}$ is Lipshitz continuous and the Lipshitz constant a subexponential random variable.

2.1.3 Moments

We need another condition that allows to control the deviation behavior of $\|\check{\mathrm{D}}^{-1}\check{\nabla}\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\|$. To present this condition define the covariance matrix $\mathcal{V}_0^2 \in \mathbb{R}^{p^* \times p^*}$ and $\check{V}^2 \in \mathbb{R}^{p \times p}$

$$\mathcal{V}_0^2 \stackrel{\text{def}}{=} \mathrm{Cov}\big\{\nabla\mathcal{L}(\boldsymbol{\upsilon}^*)\big\}, \quad \check{V}^2 = \mathrm{Cov}(\check{\nabla}_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*)).$$

We impose subexponential moments on $\check{V}^{-1}\check{\nabla}_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*)$:

$(\check{\mathcal{E}}\mathcal{D})$ There exist constants $\nu_0 > 0$ and $\check{\mathtt{g}} > 0$ such that for all $|\mu| \le \check{\mathtt{g}}$

$$\sup_{\|\gamma\|\le 1} \log \mathbb{E} \exp\Big\{\mu \gamma^\top \check{V}^{-1}\check{\nabla}_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*)\Big\} \le \frac{\check{\nu}_0^2 \mu^2}{2}.$$

2.1.4 Conditions for the Full Model

So far we only presented conditions that allow to treat the properties of $\widetilde{\boldsymbol{\theta}}_k$ on local sets $\Upsilon_\circ(\mathtt{r}_k)$, for some sequence $(\mathtt{r}_k)_{k\in\mathbb{N}}$. To show that that the sequence of estimators $(\boldsymbol{\upsilon}_k)_{k\in\mathbb{N}}$ satisfies $\boldsymbol{\upsilon}_k \in \Upsilon_\circ(\mathtt{r}_k)$ for an appropriately decreasing sequence $(\mathtt{r}_k)_{k\in\mathbb{N}}$ the following, stronger conditions are employed, which can be interpreted just as the previous ones.

$(\boldsymbol{\mathcal{L}_0})$ For each $\mathtt{r} \le \mathtt{r}_0$, there is a constant $\epsilon > 0$ such that it holds on the set $\Upsilon_\circ(\mathtt{r})$:

$$\big\|\mathcal{D}_0^{-1}\big\{\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})\big\}\mathcal{D}_0^{-1} - I_{p^*}\big\| \le \epsilon\mathtt{r}.$$

$(\mathcal{E}\mathcal{D}_1)$ There exists a constant $\epsilon \le 1/2$, such that for all $|\mu| \le \mathtt{g}$ and all $0 < \mathtt{r} < \mathtt{r}_0$

$$\sup_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})} \sup_{\|\gamma\|=1} \log \mathbb{E} \exp\left\{\frac{\mu\, \gamma^\top \mathcal{D}_0^{-1}\big\{\nabla\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \nabla\boldsymbol{\zeta}(\boldsymbol{\upsilon}')\big\}}{\epsilon\, \|\mathcal{D}_0(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}')\|}\right\} \le \frac{\nu_1^2 \mu^2}{2}.$$

$(\mathcal{E}\mathcal{D})$ There exist constants $\nu_0 > 0$ and $\mathtt{g} > 0$ such that for all $|\mu| \le \mathtt{g}$

$$\sup_{\|\gamma\|\le 1} \log \mathbb{E} \exp\Big\{\mu \gamma^\top \mathcal{V}_0^{-1}\nabla\zeta(\boldsymbol{\upsilon}^*)\Big\} \le \frac{\nu_0^2 \mu^2}{2}.$$

It is important to note, that the constants $\check{\epsilon}, \check{\nu}$ and ϵ, ν in the respective weak and strong version can differ substantially and may depend on the full dimension $p^* \in \mathbb{N}$ in less or more severe ways ($\mathcal{A}\mathrm{H}^{-2}\nabla_{\boldsymbol{\eta}}\mathcal{L}$ might be quite smooth while $\nabla_{\boldsymbol{\eta}}\mathcal{L}$ could be less regular).

For the convergence statement in Theorem 14 we additionally need the following condition, that controls the moments and the smoothness of the process $\nabla^2(\mathcal{L} - \mathbb{E}\mathcal{L})$:

$(\mathcal{E}\mathcal{D}_2)$ There exists a constant $\epsilon_2 \le 1/2$, such that for all $|\mu| \le \mathtt{g}$ and all $0 < \mathtt{r} < \mathtt{r}_0$

$$\sup_{\boldsymbol{\upsilon},\boldsymbol{\upsilon}' \in \Upsilon_\circ(\mathtt{r})} \sup_{\|\gamma_1\|=1} \sup_{\|\gamma_2\|=1} \log \mathbb{E} \exp\left\{\frac{\mu\, \gamma_1^\top \mathcal{D}^{-1}\big\{\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}')\big\}\gamma_2}{\epsilon_2\, \|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}')\|}\right\} \le \frac{\nu_2^2 \mu^2}{2}.$$

2.1.5 Quadratic Drift Beats Linear Fluctuation

Finally we present two conditions that allow to ensure that with a high probability the sequence $(\boldsymbol{\upsilon}_{k,k(+1)})$ stays close to $\boldsymbol{\upsilon}^*$ if the initial guess $\widetilde{\boldsymbol{\upsilon}}_0$ lands close to $\boldsymbol{\upsilon}^*$. These conditions have to be satisfied on the whole set $\Upsilon \subseteq \mathbb{R}^{p^*}$.

The first condition imposes that $\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})$ decreases nearly quadratically as the distance of $\boldsymbol{\upsilon}$ to $\boldsymbol{\upsilon}^*$ grows.

$(\mathcal{L}\mathtt{r})$ For any $\mathtt{r} > \mathtt{r}_0$ there exists a value $\mathtt{b} > 0$, such that

$$\mathbb{E}\left[\mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\boldsymbol{\upsilon}^*)\right] \leq \mathtt{b}\|\mathcal{D}_0(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*)\|^2.$$

The next condition bounds moments of the full gradient $\nabla\mathcal{L}(\boldsymbol{\upsilon})$ - again via subexponentiality.

$(\mathcal{E}\mathtt{r})$ For any $\mathtt{r} \geq \mathtt{r}_0$ there exists a constant $\mathtt{g}(\mathtt{r}) > 0$ such that

$$\sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \sup_{\mu\leq\mathtt{g}(\mathtt{r})} \sup_{\|\boldsymbol{\gamma}\|\leq 1} \log\mathbb{E}\exp\left\{\mu\boldsymbol{\gamma}^\top\mathcal{D}^{-1}\nabla\zeta(\boldsymbol{\upsilon})\right\} \leq \frac{\nu_{\mathtt{r}}^2\mu^2}{2}.$$

We impose one further merely technical condition:

$(\mathbf{B}_1)$ We assume for all $\mathtt{r} \geq \frac{6\nu_{\mathtt{r}}}{\mathtt{b}}\sqrt{\mathtt{x}+4p^*}$

$$1 + \sqrt{\mathtt{x}+4p^*} \leq \frac{3\nu_{\mathtt{r}}^2}{\mathtt{b}}\mathtt{g}(\mathtt{r}).$$

Remark 3 *Without this the calculation of $R_0(\mathtt{x})$ in Section 4.3 would become technically more involved, without that further insight would be gained.*

Remark 4 *The condition $(\mathcal{E}\mathtt{r})$ can be substantially relaxed to $\mathtt{b} = \mathtt{b}(\mathtt{r}) > 0$ that decreases to 0 as $\mathtt{r} \to \infty$. We avoid the resulting technicalities and refer the reader to the original publication for the non constant case (see Spokoiny, 2012, Theorem 4.1).*

2.2 Dependence on Initial Guess

Our main theorem is only valid under the conditions from Section 2.1 and under some constraints on the quality of the initial guess $\widetilde{\boldsymbol{\upsilon}}_0 \in \mathbb{R}^{p^*}$ which we denote by (A_1), (A_2) and (A_3):

$(\mathbf{A}_1)$ With probability greater $1-\beta$ the initial guess satisfies $\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0) - \mathcal{L}(\boldsymbol{\upsilon}^*) \geq -\mathrm{K}_0$ for some $\mathrm{K}_0 \geq 0$.

$(\mathbf{A}_2)$ The conditions $(\breve{\mathcal{E}}\mathcal{D}_1)$, $(\breve{\mathcal{L}}_0)$, $(\mathcal{E}\mathcal{D}_1)$ and $(\mathcal{L}_0)$ from Section 2.1 hold for all $\mathtt{r} \leq \mathrm{R}_0(\mathtt{x})$ where $\mathrm{R}_0(\mathtt{x})$ can be bounded with (see (19))

$$\mathrm{R}_0(\mathtt{x}) \leq \mathtt{C}\sqrt{\mathtt{x}+p^*+\mathrm{K}_0}.$$

$(\mathbf{A}_3)$ $\mathrm{K}_0 \in \mathbb{R}$ and $\epsilon > 0$ are small enough to ensure

$$\epsilon\mathtt{C}(\nu)\mathrm{R}_0 < 1, \tag{17}$$

with

$$\mathtt{C}(\nu) \stackrel{\text{def}}{=} \frac{16\sqrt{2}(1+\sqrt{\nu})}{(1-\nu)(1-\sqrt{\nu})}, \tag{18}$$

where $\nu > 0$ is defined in (14).

Condition (A_1) allows to concentrate the analysis on a local set $\{\|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*)\| \leq \mathrm{R}_0(\mathtt{x})\} \subset \Upsilon$ with dominating probability (see Theorem 28). Conditions (A_2) and (A_3) ensure that this neighborhood is small enough to imply convergence of the procedure. They impose a bound on $\mathrm{R}_0(\mathtt{x})$ and thus on K_0 from (A_1). These conditions boil down to $\epsilon\sqrt{\mathrm{K}_0}$ being significantly smaller than 1, which is a quantification of the quality of the first guess. There are numerous ways to initiate the procedure. In Section 3 we use a grid search and show that for a sufficiently fine grid these conditions can be met in the treated model.

Remark 5 *One way of obtaining condition (A_1) is to show that $\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}} - \boldsymbol{\upsilon}^*)\| \leq R$ with probability greater $1-\beta$ for some finite $R \in \mathbb{R}$ and $0 \leq \beta(R) < 1$. Then one can use - with some constant $\mathtt{C} > 0$ -*

$$K_0 \leq (1/2 + \epsilon(1+12\nu_0))(R + \mathtt{C}\sqrt{\mathtt{x}+p^*})^2,$$

as we show in Lemma 31.

Remark 6 *The precise definition of $R_0(\mathtt{x}) > 0$ reads*

$$R_0(\mathtt{x}) \stackrel{\text{def}}{=} \mathfrak{z}(\mathtt{x}) \vee \frac{6\nu_{\mathtt{r}}}{\mathtt{b}(1-\nu)}\sqrt{\mathtt{x} + 2.4p^* + \frac{\mathtt{b}^2}{9\nu_{\mathtt{r}}^2}K_0}, \tag{19}$$

with the term

$$\mathfrak{z}(\mathtt{x}) \approx \mathtt{C}\sqrt{p^*+\mathtt{x}},$$

which is defined in (30).

2.3 Introduction of Important Objects

In this section we collect the most important objects and bounds that are relevant for Theorem 7. Remember the $p^* \times p^*$ *information matrix* $\mathcal{D}^2$ from Section 2.1, which is defined similarly to the Fisher information matrix:

$$\mathcal{D}^2 \stackrel{\text{def}}{=} -\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}^*) = \begin{pmatrix} \mathrm{D}^2 & \mathcal{A} \\ \mathcal{A}^\top & \mathrm{H}^2 \end{pmatrix}.$$

A crucial object is the constant $0 \leq \nu$ defined by

$$\|\mathrm{D}^{-1}\mathcal{A}\mathrm{H}^{-1}\|^2 \stackrel{\text{def}}{=} \nu, \tag{20}$$

which we assume with condition $(\mathcal{I})$ to be smaller 1. It determines the speed of convergence of the alternating procedure (see Theorem 7).

Further introduce the $p \times p$ matrix $\breve{\mathrm{D}}$ and the p-vectors $\breve{\nabla}_{\boldsymbol{\theta}}$ and $\breve{\boldsymbol{\xi}}$ as

$$\breve{\mathrm{D}}^2 = \mathrm{D}^2 - \mathcal{A}\mathrm{H}^{-2}\mathcal{A}^\top,$$
$$\breve{\boldsymbol{\xi}} = \breve{\mathrm{D}}^{-1}\breve{\nabla}_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*), \quad \breve{\nabla}_{\boldsymbol{\theta}}\mathcal{L} = \nabla_{\boldsymbol{\theta}}\mathcal{L} - \mathcal{A}\mathrm{H}^{-2}\nabla_{\boldsymbol{\eta}}\mathcal{L}.$$

The random variable $\breve{\boldsymbol{\xi}} \in \mathbb{R}^p$ is related to the efficient influence function in semiparametric models. If the model is regular and correctly specified $\breve{\mathtt{D}}^2$ is the covariance of the efficient influence function and its inverse the semiparametric Cramer-Rao lower bound for regular estimators.

We define the *semiparametric uniform spread*

$$\breve{\diamondsuit}_Q(\mathtt{r},\mathtt{x}) \stackrel{\text{def}}{=} \breve{\epsilon}\left\{\frac{16}{(1-\nu^2)^2}\mathtt{r}^2 + \mathfrak{z}_Q(\mathtt{x}, 2p^*+2p)^2\right\}(1+6\breve{\nu}_1^2). \tag{21}$$

This object is central for our analysis as it describes the accuracy of our main results. It is small if $\breve{\epsilon}(\mathtt{r}^2+\mathtt{x}+p^*)$ is small, since $\mathfrak{z}_Q(\mathtt{x},p^*) \approx \sqrt{p^*+\mathtt{x}}$ (see its definition in Equation (42)).

2.4 Statistical Properties of the Alternating Sequence

In this Section we present our main theorem in full rigor, i.e. that the limit of the alternating sequence satisfies a finite sample Wilks Theorem and Fisher expansion.

Theorem 7 *Assume that the conditions* $(\mathcal{L}_\mathtt{r})$, $(\mathcal{E}\mathtt{r})$ *and* (B_1) *of Section 2.1 are met. Further assume* (A_1), (A_2) *and* (A_3) *of Section 2.2. Then it holds with probability greater* $1-8\mathrm{e}^{-\mathtt{x}}-\beta$ *for all* $k \in \mathbb{N}$

$$\left\|\breve{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) - \breve{\boldsymbol{\xi}}\right\| \le \breve{\diamondsuit}_Q(\mathtt{r}_k,\mathtt{x}), \tag{22}$$

$$\left|\max_{\boldsymbol{\eta}} \mathcal{L}(\widetilde{\boldsymbol{\theta}}_k,\boldsymbol{\eta}) - \max_{\boldsymbol{\eta}} \mathcal{L}(\boldsymbol{\theta}^*,\boldsymbol{\eta}) - \|\breve{\boldsymbol{\xi}}\|^2/2\right| \le 5\left(\|\breve{\boldsymbol{\xi}}\| + \breve{\diamondsuit}_Q(\mathtt{r}_k,\mathtt{x})\right)\breve{\diamondsuit}_Q(\mathtt{r}_k,\mathtt{x}), \tag{23}$$

where

$$\mathtt{r}_k \le \mathtt{C}\left(\sqrt{p^*+\mathtt{x}} + \nu^k R_0\right),$$

with a constant $\mathtt{C}$ *that depends on* $\nu<1$ *and* $1-\mathtt{C}(\nu)\epsilon R_0 > 0$. *In particular this means that if*

$$k \ge \frac{\log(p^*+\mathtt{x}) - \log\{R_0\}}{\log(\nu)},$$

we have

$$\breve{\diamondsuit}_Q(\mathtt{r}_k,\mathtt{x}) \approx \breve{\diamondsuit}_Q\left(\mathtt{C}\sqrt{p^*+\mathtt{x}},\mathtt{x}\right).$$

Remark 8 *Note that with linear convergence speed this leads to statements about* $\widetilde{\boldsymbol{\theta}}_k$ *that are very similar to those in (Andresen and Spokoiny, 2014) for the profile M estimator* $\widetilde{\boldsymbol{\theta}}$.

Remark 9 *Concerning the properties of* $\breve{\boldsymbol{\xi}} \in \mathbb{R}^p$ *we repeat remark 2.1 of (Andresen and Spokoiny, 2014). In case of correct model specification the deviation properties of the quadratic form* $\|\breve{\boldsymbol{\xi}}\|^2 = \|\breve{D}^{-1}\breve{\nabla}_{\boldsymbol{\theta}}\|^2$ *are essentially the same as those of a chi-square random variable with* p *degrees of freedom; see Theorem 39 in the appendix. In the case of a possible*

model misspecification the behavior of the quadratic form $\|\breve{\boldsymbol{\xi}}\|^2$ *will depend on the characteristics of the matrix* $\breve{\mathbb{B}} \stackrel{\text{def}}{=} \mathrm{Cov}(\breve{\boldsymbol{\xi}})$; *see again Theorem 39. Moreover, in the asymptotic setup the vector* $\breve{\boldsymbol{\xi}}$ *is asymptotically standard normal; see Section 2.2. of (Andresen and Spokoiny, 2014) for the i.i.d. case.*

Remark 10 *These results allow to derive some important corollaries like concentration and confidence sets (see Section 1.3).*

Remark 11 *In general an exact numerical computation of*

$$\theta(\boldsymbol{\eta}) \stackrel{\text{def}}{=} \operatorname*{argmax}_{\boldsymbol{\theta}\in\mathbb{R}^p} \mathcal{L}(\boldsymbol{\theta},\boldsymbol{\eta}), \quad \text{or} \quad \eta(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \operatorname*{argmax}_{\boldsymbol{\eta}\in\mathbb{R}^m} \mathcal{L}(\boldsymbol{\theta},\boldsymbol{\eta}),$$

is not possible. Define $\widehat{\theta}(\boldsymbol{\eta})$ *and* $\widehat{\eta}(\boldsymbol{\theta})$ *as the numerical approximations to* $\theta(\boldsymbol{\eta})$ *and* $\eta(\boldsymbol{\theta})$ *and assume that - with the local set* $\Upsilon_\circ(\mathtt{r})$ *defined in* (15) *-*

$$\|D(\widehat{\theta}(\boldsymbol{\eta}) - \theta(\boldsymbol{\eta}))\| \le \tau, \ \text{ for all } \ \boldsymbol{\eta} \in \Upsilon_{\circ,\boldsymbol{\eta}}(R_0) \stackrel{\text{def}}{=} \{\boldsymbol{\upsilon} \in \Upsilon_\circ(R_0),\, \Pi_{\boldsymbol{\eta}}\boldsymbol{\upsilon} = \boldsymbol{\eta}\},$$

$$\|H(\widehat{\eta}(\boldsymbol{\theta}) - \eta(\boldsymbol{\theta}))\| \le \tau, \ \text{ for all } \ \boldsymbol{\theta} \in \Upsilon_{\circ,\boldsymbol{\theta}}(R_0) \stackrel{\text{def}}{=} \{\boldsymbol{\upsilon} \in \Upsilon_\circ(R_0),\, \Pi_{\boldsymbol{\theta}}\boldsymbol{\upsilon} = \boldsymbol{\theta}\}.$$

Then we can easily modify the proof of Theorem 7 via adding $\mathtt{C}(\nu)\tau$ *to the error terms and the radii* $\mathtt{r}_k$, *where* $\mathtt{C}(\nu)$ *is some rational function of* ν.

2.5 Convergence to the ME

Even though Theorem 7 tells us that the statistical properties of the alternating sequence resemble those of its target - the profile ME $\widetilde{\boldsymbol{\theta}}$ - it is an interesting question if the underlying approach allows to qualify conditions under which the sequence actually attains the maximizer $\widetilde{\boldsymbol{\upsilon}}$.

Define the radius $\mathtt{r}_0(\mathtt{x}) > 0$ to be the smallest radius $\mathtt{r} > 0$ such that $\mathbb{P}(\widetilde{\boldsymbol{\upsilon}}, \widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \in \Upsilon_\circ(\mathtt{r}_0)) \ge 1-\mathrm{e}^{-\mathtt{x}}$, where

$$\widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \stackrel{\text{def}}{=} \operatorname*{argmax}_{\substack{\boldsymbol{\upsilon}\in\Upsilon \\ \Pi_{\boldsymbol{\theta}}\boldsymbol{\upsilon}=\boldsymbol{\theta}^*}} \mathcal{L}(\boldsymbol{\upsilon}).$$

Remark 12 *This radius can be determined using conditions* $(\mathcal{L}_\mathtt{r})$ *and* $(\mathcal{E}\mathtt{r})$ *of Section 2.1 and Theorem 28 which would yield* $\mathtt{r}_0(\mathtt{x}) = O(\sqrt{\mathtt{x}+p^*})$.

Without further assumptions Theorem 7 yields the following Corollary:

Corollary 13 *Under the assumptions of Theorem 7 it holds with probability greater* $1-8\mathrm{e}^{-\mathtt{x}}-\beta$

$$\|\breve{D}(\widetilde{\boldsymbol{\theta}} - \widetilde{\boldsymbol{\theta}}_k)\| \le \breve{\diamondsuit}_Q(\mathtt{r}_k,\mathtt{x}) + \breve{\diamondsuit}_Q(\mathtt{r}_0,\mathtt{x}).$$

Corollary 13 is a first step in the direction of an actual convergence result but the gap $\breve{\diamondsuit}_Q(\mathtt{r}_k,\mathtt{x}) + \breve{\diamondsuit}_Q(\mathtt{r}_0,\mathtt{x})$ is not a zero sequence in $k \in \mathbb{N}$. It turns out that it is possible to

prove convergence to the ME at the cost of assuming more smoothness of the functional $\mathcal{L}$ and using the right bound for the maximal eigenvalue of the Hessian $\nabla^2\mathcal{L}(\upsilon^*)$.

Define $\mathfrak{z}_2(\mathtt{x}, \nabla^2\mathcal{L}(\upsilon^*))$ via

$$\mathbb{P}\Big(\|\mathcal{D}^{-1}\nabla^2\mathcal{L}(\upsilon^*)\| \geq \mathfrak{z}_2\left(\mathtt{x}, \nabla^2\mathcal{L}(\upsilon^*)\right) \Big) \leq \mathrm{e}^{-\mathtt{x}},$$

and $\kappa(\mathtt{x}, \mathrm{R}_0)$ as

$$\kappa(\mathtt{x}, \mathrm{R}_0) \stackrel{\mathrm{def}}{=} \frac{2\sqrt{2}(1+\sqrt{\nu})}{\sqrt{1-\nu}} \left\{ \epsilon \mathrm{R}_0 + 9\epsilon_2\nu_2 \|\mathcal{D}^{-1}\| \mathfrak{z}_1(\mathtt{x}, 6p^*)\mathrm{R}_0 + \|\mathcal{D}^{-1}\| \mathfrak{z}_2\left(\mathtt{x}, \nabla^2\mathcal{L}(\upsilon^*)\right) \right\},$$

where $\mathfrak{z}_1(\mathtt{x}, \cdot) \approx \sqrt{\mathtt{x} + p^*}$, see (46). With these definitions we can prove the following Theorem:

Theorem 14 *Let the conditions $(\mathcal{L}_\mathtt{r})$, $(\mathcal{E}\mathtt{r})$ and (B_1) be met. Further suppose (A_1) and (A_2), with $(\mathcal{E}\mathcal{D}_2)$ instead of $(\mathcal{E}\mathcal{D}_1)$. Assume that $\kappa(\mathtt{x}, R_0) < (1-\nu)$. Then*

$$\mathbb{P}\left(\bigcap_{k \in \mathbb{N}} \left\{ \|\mathcal{D}(\upsilon_{k,k(+1)} - \widetilde{\upsilon})\| \leq \mathtt{r}_k^* \right\} \right) \geq 1 - 3\mathrm{e}^{-\mathtt{x}} - \beta,$$

where

$$\mathtt{r}_k^* \leq \begin{cases} \nu^k \frac{4\sqrt{2}}{1-\kappa(\mathtt{x},R_0)k} R_0, & \kappa(\mathtt{x}, R_0)k \leq 1, \\ \nu^{\frac{k}{\log(k)}\log\left(\frac{1-\nu}{\kappa(\mathtt{x},R_0)}\right)} c_k R_0, & \textit{otherwise}, \end{cases} \tag{24}$$

with some sequence $(c_k) \in \mathbb{N}$, where $0 < c_k \to 2$.

Remark 15 *This means that we obtain nearly linear convergence to the global maximizer $\widetilde{\upsilon}$.*

Remark 16 *As in Remark 11 if no exact numerical computation of the stepwise maximizers is possible we can easily modify the proof of Theorem 14 via adding $\mathtt{C}(\nu)\tau$ to $\kappa(\mathtt{x}, R_0)$ to address that case.*

Remark 17 *For the case that $\mathcal{L}(\upsilon) = \sum_{i=1}^n \ell_i(\upsilon)$ with a sum of independent marginal functionals $\ell_i : \Upsilon \to \mathbb{R}$ we can use Corollary 3.7 of (Tropp, 2012) to obtain*

$$\mathfrak{z}_2\left(\mathtt{x}, \nabla^2\mathcal{L}(\upsilon^*)\right) = \sqrt{2\tau}\nu\sqrt{\mathtt{x} + \log(p^*)},$$

if for some sequence of matrices $(\boldsymbol{A}_i) \subset \mathbb{R}^{p^* \times p^*}$

$$\log \mathbb{E} \exp \lambda \nabla^2 \ell_i(\upsilon^*) \preceq \nu^2\lambda^2/2\boldsymbol{A}_i, \quad \|\sum_{i=1}^n \boldsymbol{A}_i\| \leq \tau.$$

In the case of smooth i.i.d models this means that

$$\kappa(\mathtt{x}, R_0) \leq \frac{\mathtt{C}}{\sqrt{n}}(\mathtt{x} + R_0 + \log(p^*)),$$

if $p^* + \mathtt{x} = o(n)$.

Remark 18 *It may happen that $\kappa(\mathtt{x}, R_0)/(1-\nu)$ is very close to or even larger than 1. But a close look at the proof of Theorem 14 reveals that this can be improved using Lemma 33. For this purpose bound $\mathtt{r}_k^* \leq \mathtt{C}^*(\mathfrak{z}(\mathtt{x}) + \nu^k R_0)$ with $\mathtt{r}_k^*$ defined in (33) and with some constant $\mathtt{C}^* > 0$. Then the result of Theorem 14 is true with $\kappa(\mathtt{x}, \mathtt{C}^*\sqrt{p^* + \mathtt{x}})$ instead of $\kappa(\mathtt{x}, R_0)$ and with probability greater $1 - 10\mathrm{e}^{-\mathtt{x}}$. See Remark 38 for more details.*

2.6 Critical Dimension

We want to address the issue of *critical parameter dimensions* when the full dimension p^* grows with the sample size n. We write $p^* = p_n$. The results of Theorem 7 are accurate if the spread function $\breve{\Diamond}_Q(\mathtt{r}_k, \mathtt{x})$ from (21) is small. The critical size of p_n then depends on the exact bounds on $\epsilon, \check{\epsilon}$. In the i.i.d setting we have $\epsilon \asymp \check{\epsilon} \asymp 1/\sqrt{n}$ such that $\breve{\Diamond}(\mathtt{r}_k, \mathtt{x}) \asymp p_n/\sqrt{n}$ for large $k \in \mathbb{N}$. In other words, one needs that "${p_n}^2/n$ is small" to obtain an accurate non-asymptotic version of the Wilks phenomenon and the Fisher Theorem for the limit of the alternating sequence. This is not surprising because good performance of the ME itself can only be guaranteed if "${p_n}^2/n$ is small", as is shown by Andresen and Spokoiny (2014). There are examples where the pME only satisfies a Wilks- or Fisher result if "${p_n}^2/n$ is small", such that in any of those settings the alternating sequence started in the global maximizer does not admit an accurate Wilks- or Fisher expansion.

Interesting enough the constrain $\kappa(\mathtt{x}, \mathrm{R}_0) < (1-\nu)$ of Theorem 14 for the convergence of the sequence to the global maximizer means that one needs $p_n/n \ll 1$ in the smooth i.i.d. setting if $\mathrm{R}_0 \leq \mathtt{C}_{\mathrm{R}_0}\sqrt{p_n + \mathtt{x}}$. Further Theorem 14 states a lower bound for the speed of convergence that in the smooth i.i.d. setting decreases if p_n/n grows. Unfortunately we were unable to find an example that meets the conditions of Section 2.1 and where no convergence occurs if p_n/n tends to infinity. So whether this dimension effect on the convergence is an artifact of our proofs or indeed a property of the alternating procedure remains an open question.

3. Application to Single Index Model

We illustrate how the results of Theorem 7 and Theorem 14 can be applied in Single Index modeling. This section is based on (Andresen, 2015). See that paper for a more detailed presentation.

Consider the following model

$$y_i = f(\boldsymbol{X}_i^\top \boldsymbol{\theta}^*) + \varepsilon_i, \qquad i = 1, ..., n,$$

for some $f : \mathbb{R} \to \mathbb{R}$ and $\boldsymbol{\theta}^* \in S_1^{p,+} \subset \mathbb{R}^p$ and with i.i.d errors $\varepsilon_i \in \mathbb{R}$, $\mathrm{Var}(\varepsilon_i) = \sigma^2$ and i.i.d random variables $\boldsymbol{X}_i \in \mathbb{R}^p$ with distribution denoted by $\mathbb{P}^{\boldsymbol{X}}$. The single-index model is widely applied in statistics. For example in econometric studies it serves as a compromise between too restrictive parametric models and flexible but hardly estimable purely non-parametric models. Usually the statistical inference focuses on estimating the index vector $\boldsymbol{\theta}^*$. A lot of research has already been done in this field. For instance, (Delecroix. et al., 1997) show the asymptotic efficiency of the general semiparametric maximum-functional estimator for particular examples and in (Haerdle et al., 1993) the right choice of band-

width for the nonparametric estimation of the link function is analyzed. We want to use this model to illustrate our theoretical results.

To ensure identifiability of $\boldsymbol{\theta}^* \in \mathbb{R}^p$ we assume that it lies in the half sphere $S_1^{p,+} \stackrel{\text{def}}{=} \{\boldsymbol{\theta} \in \mathbb{R}^p : \|\boldsymbol{\theta}\| = 1,\, \theta_1 > 0\} \subset \mathbb{R}^p$. For simplicity we assume that the support of the $\boldsymbol{X}_i \in \mathbb{R}^p$ is contained in the ball of radius $s > 0$. This allows to approximate $f \in \{f : [-s, s] \mapsto \mathbb{R}\}$ by an orthonormal C^3-Daubechies-wavelet basis $(\boldsymbol{e}_k)_{k\in\mathbb{N}}$ on the interval.

A candidate to estimate $\boldsymbol{\theta}^*$ is the sieve profile ME

$$\widetilde{\boldsymbol{\theta}}_m \stackrel{\text{def}}{=} \Pi_{\boldsymbol{\theta}} \operatorname*{argmax}_{(\boldsymbol{\theta},\boldsymbol{\eta})\in\Upsilon_m} \mathcal{L}_m(\boldsymbol{\theta}, \boldsymbol{\eta}),$$

where

$$\mathcal{L}_m(\boldsymbol{\theta}, \boldsymbol{\eta}) = -\frac{1}{2}\sum_{i=1}^{n}\Big|y_i - \sum_{k=0}^{m} \boldsymbol{\eta}_k \boldsymbol{e}_k(\boldsymbol{X}_i^\top \boldsymbol{\theta})\Big|^2,$$

and where $\Upsilon_m = S_1^{p,+} \times \mathbb{R}^m$. Again we will suppress the sub index $\cdot_m$ in the following.

In this setting a direct computation of $\widetilde{\boldsymbol{\upsilon}}$ becomes involved, as the maximization problem is high dimensional and not convex. But as noted in the introduction the maximization with respect to $\boldsymbol{\eta}$ for given $\boldsymbol{\theta}$ is high dimensional but convex and consequently feasible. Further for moderate $p \in \mathbb{N}$ the maximization with respect to $\boldsymbol{\theta}$ for fixed $\boldsymbol{\eta}$ is computationally realistic. So an alternating maximization procedure is applicable. To show that it behaves in a desired way we apply the technique presented above.

For the initial guess $\widetilde{\boldsymbol{\upsilon}}_0 \in \Upsilon$ one can use a simple grid search. For this take a uniform grid $G_N \stackrel{\text{def}}{=} (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N) \subset S_1^+$ and define

$$\widetilde{\boldsymbol{\upsilon}}_0 \stackrel{\text{def}}{=} \operatorname*{argmax}_{\substack{(\boldsymbol{\theta},\boldsymbol{\eta})\in\Upsilon \\ \boldsymbol{\theta}\in G_N}} \mathcal{L}(\boldsymbol{\upsilon}). \tag{25}$$

Note that given the grid the above maximizer is easily obtained. Simply calculate

$$\widetilde{\boldsymbol{\eta}}_{0,k} \stackrel{\text{def}}{=} \operatorname{argmax} \mathcal{L}(\boldsymbol{\theta}_k, \boldsymbol{\eta}) = \left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{e}\boldsymbol{e}^\top(\boldsymbol{X}_i^\top \boldsymbol{\theta}_k)\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} y_i \boldsymbol{e}^\top(\boldsymbol{X}_i^\top \boldsymbol{\theta}_k) \in \mathbb{R}^m, \tag{26}$$

where by abuse of notation $\boldsymbol{e} = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m) \in \mathbb{R}^m$. Now observe that

$$\widetilde{\boldsymbol{\upsilon}}_0 = \operatorname*{argmax}_{k=1,\ldots,N} \mathcal{L}(\boldsymbol{\theta}_k, \widetilde{\boldsymbol{\eta}}_{0,k}).$$

Define the mesh size of the grid $\tau \stackrel{\text{def}}{=} \sup_{\boldsymbol{\theta},\boldsymbol{\theta}^\circ\in G_N} \|\boldsymbol{\theta} - \boldsymbol{\theta}^\circ\|$.

To apply the result presented in Theorem 7 and Theorem 14 we need a list of assumptions denoted by $(\mathcal{A})$.

($\mathbf{Cond}_{\boldsymbol{X}}$) The random variables $(\boldsymbol{X}_i)_{i=1,\ldots,n} \subset \mathbb{R}^p$ are i.i.d and bounded with distribution denoted by $\mathbb{P}^{\boldsymbol{X}}$ and independent of $(\varepsilon_i)_{i=1,\ldots,n} \subset \mathbb{R}$.

The measure $\mathbb{P}^{\boldsymbol{X}}$ is absolutely continuous with respect to the Lebesgue measure. The

Lebesgue density $p_{\boldsymbol{X}}$ of $\mathbb{P}^{\boldsymbol{X}}$ is Lipschitz continuous and positive on $B_s(0) \subset \mathbb{R}^p$. For any pair $\boldsymbol{\theta} \in S_1^{+,p}$ with $\boldsymbol{\theta} \perp \boldsymbol{\theta}^*$ we have almost surely

$$\operatorname{Var}\left(\boldsymbol{X}^\top\boldsymbol{\theta}\middle|\boldsymbol{X}^\top\boldsymbol{\theta}^*\right) > 0.$$

($\mathbf{Cond}_f$) For some $\boldsymbol{\eta}^* \in B_{\mathtt{r}^\circ}(0) \subset l^2 \stackrel{\text{def}}{=} \{(u_k)_{k\in\mathbb{N}} : \sum_{k=1}^{\infty} u_k^2 < \infty\}$

$$f = \sum_{k=1}^{\infty} \eta_k^* \boldsymbol{e}_k,$$

where $\|f'\|_\infty < \infty$ and $\|f''\|_\infty < \infty$ and where with some $\alpha > 2$

$$\sum_{k=0}^{\infty} k^{2\alpha} \eta_k^{*2} < \infty.$$

On some interval $[t_0 - h, t_0 + h] \subseteq [-s + s]$ with $h > 0$ it holds true that

$$|f'(t)| > 0.$$

($\mathbf{Cond}_\varepsilon$) The errors $(\varepsilon_i) \in \mathbb{R}$ are i.i.d. with $\mathbb{E}[\varepsilon_i] = 0$, $\operatorname{Cov}(\varepsilon_i) = \sigma^2$ and satisfy for all $|\mu| \le \widetilde{g}$ for some $\widetilde{g} > 0$ and some $\widetilde{\nu} > 0$

$$\log \mathbb{E}[\exp\{\mu\varepsilon_1\}] \le \widetilde{\nu}^2\mu^2/2.$$

Remark 19 *Note that our assumptions in terms of moments and smoothness are quite common in this model. For instance (Haerdle et al., 1993) assume that the density $p_{\boldsymbol{X}}$ of the regressors $(\boldsymbol{X}_i)$ is twice continuously differentiable, that f has two bounded derivatives and that the errors (ε_i) are centered with bounded polynomial moments of arbitrary degree.*

Remark 20 $\operatorname{Var}\left(\boldsymbol{X}^\top\boldsymbol{\theta}^\circ\middle|\boldsymbol{X}^\top\boldsymbol{\theta}^*\right) = 0$ *would mean that* $\boldsymbol{X}^\top\boldsymbol{\theta}^\circ = a(\boldsymbol{X}^\top\boldsymbol{\theta}^*)$ *for some measurable function* $a : \mathbb{R} \to \mathbb{R}$. *But then we would have for any* $(\alpha, \beta) \in \mathbb{R}^2$ *with* $\alpha^2 + \beta^2 = 1$ *that*

$$f(\boldsymbol{X}^\top(\alpha\boldsymbol{\theta}^* + \beta\boldsymbol{\theta}^\circ)) = f(\alpha\boldsymbol{X}^\top\boldsymbol{\theta}^* + \beta a(\boldsymbol{X}^\top\boldsymbol{\theta}^*)) \stackrel{\text{def}}{=} f^\circ_{\alpha,\beta}(\boldsymbol{X}^\top\boldsymbol{\theta}^*),$$

such that the problem would no longer be identifiable. Also $|f'(t)| > 0$ *on some interval is necessary for identifyability of* $\boldsymbol{\theta}^*$.

Proposition 21 *Let* $\tau = o(p^{*-3/2})$ *and* $p^{*5}/n \to 0$. *With initial guess given by Equation* (25) *and for not too large* $\mathtt{x} > 0$ *the alternating sequence satisfies* (22) *and* (23) *with probability greater* $1 - 9\exp\{-\mathtt{x}\}$ *and where with some constant* $\mathtt{C} \in \mathbb{R}$

$$\overset{\circ}{\diamondsuit}_Q(\mathtt{r}, \mathtt{x}) \le \frac{\mathtt{C}(p^* + \mathtt{x})^{3/2}}{\sqrt{n}}(\mathtt{r}^2 + p^* + \mathtt{x}).$$

Proposition 22 *Take the initial guess given by Equation* (25). *Assume* $(\mathcal{A})$. *Further assume that* $p^{*4}/n \to 0$ *and* $\tau = o(p^{*-3/2})$. *Then we get the claim of Theorem 14 with* $\beta = \mathrm{e}^{-\mathtt{x}}$ *and*

$$\kappa(\mathtt{x}, R_0) = O(\tau p^{*3/2} + \sqrt{\tau \mathtt{x}} p^{*3/2}/n^{1/4}) + O(p^{*2}/\sqrt{n}) \to 0,$$

for moderate choice of $\mathtt{x} > 0$.

Remark 23 *The constraint* $\tau = o(p^{*-3/2})$ *implies that for the calculation of the initial guess the vector* $\widetilde{\boldsymbol{\eta}}_{0,l}$ *of* (26) *and the functional* $\mathcal{L}(\cdot)$ *have to be evaluated* $N = p^{*3(p-1)/2}$ *times.*

For details and proofs see (Andresen, 2015).

4. Proof of Theorem 7

In this section we present the proof of Theorem 7. As the proof is quite technical and complex we want to first explain the basic ideas of the proof. In a second section we will outline more clearly the steps of the proof. Finally we carry out each of these steps which combine to yield the proof.

4.1 Idea of the Proof

To ease the understanding of what follows in the subsequent sections we want to illustrate the central ideas with a simple model. Consider for some positive definite matrix $\mathcal{D} \in \mathbb{R}^{p^* \times p^*}$ and some vector $\boldsymbol{\upsilon}^* = (\boldsymbol{\theta}^*, \boldsymbol{\eta}^*) \in \mathbb{R}^{p+m} = \mathbb{R}^{p^*}$ the model

$$\mathbb{Y} = \boldsymbol{\upsilon}^* + \boldsymbol{\varepsilon} \in \mathbb{R}^{p^*}, \text{ where } \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathcal{D}^{-2}), \quad \mathcal{D}^2 = \begin{pmatrix} \mathrm{D}^2 & A \\ A^\top & \mathrm{H}^2 \end{pmatrix} \in \mathbb{R}^{p^* \times p^*}.$$

Set $\mathcal{L}$ to be the true log likelihood of the observations, i.e.

$$\mathcal{L}(\boldsymbol{\upsilon}, \mathbb{Y}) = -\|\mathcal{D}(\boldsymbol{\upsilon} - \mathbb{Y})\|^2/2.$$

With any starting initial guess $\widetilde{\boldsymbol{\upsilon}}_0 \in \mathbb{R}^{p+m}$ we obtain from (3) for $k \in \mathbb{N}$ and the usual first order criterion of maximality the following two equations

$$\begin{aligned} \mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) &= \mathrm{D}\boldsymbol{\varepsilon}_{\boldsymbol{\theta}} + \mathrm{D}^{-1} A (\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*), \\ \mathrm{H}(\widetilde{\boldsymbol{\eta}}_{k+1} - \boldsymbol{\eta}^*) &= \mathrm{H}\boldsymbol{\varepsilon}_{\boldsymbol{\eta}} + \mathrm{H}^{-1} A^\top (\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*). \end{aligned}$$

Combining these two equations we derive, assuming $\|\mathrm{D}^{-1} A \mathrm{H}^{-2} A^\top \mathrm{D}^{-1}\| \stackrel{\text{def}}{=} \|\boldsymbol{M}_0\| = \nu < 1$

$$\begin{aligned} \mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) &= \mathrm{D}^{-1}(\mathrm{D}^2 \boldsymbol{\varepsilon}_{\boldsymbol{\theta}} - A \boldsymbol{\varepsilon}_{\boldsymbol{\eta}}) + \mathrm{D}^{-1} A \mathrm{H}^{-1} A^\top \mathrm{D}^{-1} \mathrm{D}(\widetilde{\boldsymbol{\theta}}_{k-1} - \boldsymbol{\theta}^*) \\ &= \sum_{l=1}^{k} \boldsymbol{M}_0^{k-l} \mathrm{D}^{-1}(\mathrm{D}^2 \boldsymbol{\varepsilon}_{\boldsymbol{\theta}} - A \boldsymbol{\varepsilon}_{\boldsymbol{\eta}}) + \boldsymbol{M}_0^k \mathrm{D}(\widetilde{\boldsymbol{\theta}}_0 - \boldsymbol{\theta}^*) \\ &\to \mathbf{X} \stackrel{\text{def}}{=} \mathrm{D}(\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*). \end{aligned}$$

Because the limit $\widehat{\boldsymbol{\theta}}$ is independent of the initial point $\widetilde{\boldsymbol{\upsilon}}_0$ and because the profile $\widetilde{\boldsymbol{\theta}}$ is a fix-point of the procedure the unique limit satisfies $\widehat{\boldsymbol{\theta}} = \widetilde{\boldsymbol{\theta}}$. This argument is based on the fact that in this setting the functional is quadratic such that the gradient satisfies

$$\nabla \mathcal{L}(\boldsymbol{\upsilon}) = \mathcal{D}^2_{\boldsymbol{\upsilon}^*}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*) + \mathcal{D}^2_{\boldsymbol{\upsilon}^*} \boldsymbol{\varepsilon}.$$

Any smooth function is quadratic around its maximizer which motivates a local linear approximation of the gradient of the functional $\mathcal{L}$ to derive our results with similar arguments. This is done in the proof of Theorem 7 .

First it is ensured that the whole sequence $(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)})_{k \in \mathbb{N}_0}$ satisfies for some $\mathrm{R}_0(\mathtt{x}) > 0$ and with probability greater that $1 - \mathrm{e}^{-\mathtt{x}}$

$$\{\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)},\, k \in \mathbb{N}_0\} \subset \{\|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*)\| \le \mathrm{R}_0(\mathtt{x})\}, \tag{27}$$

where $\mathcal{D}^2 \stackrel{\text{def}}{=} -\nabla^2 \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}^*)$ (see Theorem 28). In the second step we approximate with $\zeta = \mathcal{L} - \mathbb{E}\mathcal{L}$

$$\mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\boldsymbol{\upsilon}^*) = \nabla \zeta(\boldsymbol{\upsilon}^*)(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*) - \|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*)\|^2/2 + \alpha(\boldsymbol{\upsilon}, \boldsymbol{\upsilon}^*), \tag{28}$$

where $\alpha(\boldsymbol{\upsilon}, \boldsymbol{\upsilon}^*)$ is defined by (28). Similar to the toy case above this allows using the first order criterion of maximality and (27) to obtain a bound of the kind

$$\begin{aligned} \|\mathcal{D}(\boldsymbol{\upsilon}_{k,k} - \boldsymbol{\upsilon}^*)\| &\le \mathtt{C} \sum_{l=0}^{k} \nu^l \left(\|\mathcal{D}^{-1} \nabla \zeta(\boldsymbol{\upsilon}^*)\| + |\alpha(\boldsymbol{\upsilon}_{l,l}, \boldsymbol{\upsilon}^*)| \right) \\ &\le \mathtt{C}_1 \left(\|\mathcal{D}^{-1} \nabla \zeta(\boldsymbol{\upsilon}^*)\| + \epsilon(\mathrm{R}_0) \right) + \nu^k \mathrm{R}_0 \stackrel{\text{def}}{=} \mathtt{r}_k. \end{aligned}$$

This is done in Lemma 32 using results from (Andresen and Spokoiny, 2014) to show that $\epsilon(\mathrm{R}_0)$ is small. Finally the same arguments as in (Andresen and Spokoiny, 2014) allow to obtain our main result using that with high probability for all $k \in \mathbb{N}_0$ $\widetilde{\boldsymbol{\upsilon}}_{k,k} \in \{\|\mathcal{D}(\boldsymbol{\upsilon} - \boldsymbol{\upsilon}^*)\| \le \mathtt{r}_k\}$. For the convergence result similar arguments are used. The only difference is that instead of (28) we use the approximation

$$\mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\widetilde{\boldsymbol{\upsilon}}) = -\|\mathcal{D}(\boldsymbol{\upsilon} - \widetilde{\boldsymbol{\upsilon}})\|^2/2 + \alpha'(\boldsymbol{\upsilon}, \widetilde{\boldsymbol{\upsilon}}),$$

exploiting that $\nabla \mathcal{L}(\widetilde{\boldsymbol{\upsilon}}) \equiv 0$, which allows to obtain actual convergence to the ME.

4.2 A Desirable Set

In this section we will explain the agenda of the proof. The first step of the proof is to find a desirable set $\Omega(\mathtt{x}) \subset \Omega$ of high probability, on which a linear approximation of the gradient of the functional $\mathcal{L}(\boldsymbol{\upsilon})$ can be carried out with sufficient accuracy. Once this set is found all subsequent analysis concerns events in $\Omega(\mathtt{x}) \subset \Omega$.

For this purpose define - with the local set $\Upsilon_\circ(\mathtt{r})$ defined in (15) - for some $K \in \mathbb{N}$ the set

$$\Omega(\mathtt{x}) = \bigcap_{k=0}^{K} (C_{k,k} \cap C_{k,k+1}) \cap C(\nabla) \cap \{\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0) - \mathcal{L}(\boldsymbol{\upsilon}^*) \geq -\mathrm{K}_0\}, \text{ where} \tag{29}$$

$$C_{k,k(+1)} = \Big\{ \|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \boldsymbol{\upsilon}^*)\| \leq \mathrm{R}_0(\mathtt{x}),\ \|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*)\| \leq \mathrm{R}_0(\mathtt{x}),$$
$$\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_{k(+1)} - \boldsymbol{\eta}^*)\| \leq \mathrm{R}_0(\mathtt{x}) \Big\},$$

$$C(\nabla) = \bigcap_{\mathtt{r} \leq \mathrm{R}_0(\mathtt{x})} \left\{ \sup_{\boldsymbol{\upsilon} \in \Upsilon_\circ(\mathtt{r})} \left\{ \frac{1}{6\epsilon\nu_1} \|\mathcal{Y}(\boldsymbol{\upsilon})\| - 2\mathtt{r}^2 \right\} \leq \mathfrak{z}_Q(\mathtt{x}, 4p^*)^2 \right\}$$
$$\bigcap_{\mathtt{r} \leq 4\mathrm{R}_0(\mathtt{x})} \left\{ \sup_{\boldsymbol{\upsilon} \in \Upsilon_\circ(\mathtt{r})} \left\{ \frac{1}{6\check{\epsilon}\check{\nu}_1} \|\check{\mathcal{Y}}(\boldsymbol{\upsilon})\| - 2\mathtt{r}^2 \right\} \leq \mathfrak{z}_Q(\mathtt{x}, 2p^* + 2p)^2 \right\}$$
$$\cap \Big\{ \max\{\|\mathcal{D}^{-1}\nabla\mathcal{L}\|, \|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}\|, \|\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\mathcal{L}\|\} \leq \mathfrak{z}(\mathtt{x}) \Big\}$$
$$\cap \{\widetilde{\boldsymbol{\upsilon}}, \widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \in \Upsilon_\circ(\mathtt{r}_0(\mathtt{x}))\}.$$

For $\boldsymbol{\zeta}(\boldsymbol{\upsilon}) = \mathcal{L}(\boldsymbol{\upsilon}) - \mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})$ the semiparametric normalized stochastic gradient gap is defined as

$$\check{\mathcal{Y}}(\boldsymbol{\upsilon}) = \check{\mathrm{D}}^{-1}\Big(\check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \check{\nabla}_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\Big).$$

the parametric normalized stochastic gradient gap $\mathcal{Y}(\boldsymbol{\upsilon})$ is defined as

$$\mathcal{Y}(\boldsymbol{\upsilon}) = \mathcal{D}^{-1}\Big(\nabla\zeta(\boldsymbol{\upsilon}) - \nabla\zeta(\boldsymbol{\upsilon}^*)\Big),$$

and $\mathtt{r}_0(\mathtt{x}) > 0$ is chosen such that $\mathbb{P}(\widetilde{\boldsymbol{\upsilon}}, \widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \in \Upsilon_\circ(\mathtt{r}_0)) \geq 1 - \mathrm{e}^{-\mathtt{x}}$, where

$$\widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \stackrel{\text{def}}{=} \operatorname*{argmax}_{\substack{\boldsymbol{\upsilon} \in \Upsilon \\ \Pi_{\boldsymbol{\theta}}\boldsymbol{\upsilon} = \boldsymbol{\theta}^*}} \mathcal{L}(\boldsymbol{\upsilon}).$$

The constant $\mathfrak{z}(\mathtt{x})$ in the definition of $C(\nabla)$ is only introduced for ease of notation. This makes some bounds less sharp but allows to address all terms that are of order $\sqrt{p^* + \mathtt{x}}$ with one symbol. It is defined as

$$\mathfrak{z}(\mathtt{x}) \stackrel{\text{def}}{=} \mathfrak{z}(\mathtt{x}, \mathcal{D}^{-1}\mathcal{V}^2\mathcal{D}^{-1}) \vee \mathfrak{z}_Q(\mathtt{x}, 4p^*) \approx \sqrt{p^* + \mathtt{x}}, \tag{30}$$

where $\mathcal{V}^2 \in \mathbb{R}^{p^* \times p^*}$ denotes the *covariance matrix* from Section 2.1

$$\mathcal{V}^2 \stackrel{\text{def}}{=} \mathrm{Cov}(\nabla\mathcal{L}(\boldsymbol{\upsilon}^*)).$$

Remark 24 *$\mathfrak{z}(\mathtt{x}, \cdot)$ is explained in more detail in Section A and $\mathfrak{z}_Q(\mathtt{x}, \cdot)$ is defined in Equation (42). The constant $\mathfrak{z}(\mathtt{x}, I\!B)$ is comparable to the "$1 - \mathrm{e}^{-\mathtt{x}}$"-quantile of the norm of $\boldsymbol{X}$, where $\boldsymbol{X} \sim \mathcal{N}(0, I\!B)$, i.e. it is of order of the trace of $I\!B$. The constant $\mathfrak{z}_Q(\mathtt{x}, \mathbb{Q})$ arises as an exponential deviation bound for the supremum of a smooth process over a set with complexity described by $\mathbb{Q}$.*

Remark 25 *We intersect the set with the event $\{\widetilde{\boldsymbol{\upsilon}}, \widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \in \Upsilon_\circ(\mathtt{r}_0)\}$ where we a priory demand $\mathtt{r}_0(\mathtt{x}) > 0$ to be chosen such that $\mathbb{P}(\widetilde{\boldsymbol{\upsilon}}, \widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \in \Upsilon_\circ(\mathtt{r}_0)) \geq 1 - \mathrm{e}^{-\mathtt{x}}$. Note that condition $(\mathcal{E}\mathtt{r})$ together with $(\mathcal{L}\mathtt{r})$ allow to set $\sqrt{p^* + \mathtt{x}} \approx \mathtt{r}_0 \leq R_0$ (see Theorem 28).*

In Section 4.3 we show that this set is of probability greater $1 - 8\mathrm{e}^{-\mathtt{x}} - \beta_{(\boldsymbol{A})}$. We want to explain the purpose of this set along the architecture of the proof of our main theorem.

$\{\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0, \boldsymbol{\upsilon}^*) \geq -\mathbf{K}_0\}$: This set ensures, that the first guess satisfies $\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0, \boldsymbol{\upsilon}^*) \geq -\mathrm{K}_0$, which means that it is close enough to the target $\boldsymbol{\upsilon}^* \in \mathbb{R}^{p^*}$. This fact allows us to obtain an a priori bound for the deviation of the sequence $(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}) \subset \Upsilon$ from $\boldsymbol{\upsilon}^* \in \Upsilon$ with Theorem 28.

$\{\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \boldsymbol{\upsilon}^*) \leq \mathbf{R}_0(\mathtt{x})\}$: As just mentioned this event is of high probability due to $\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0, \boldsymbol{\upsilon}^*) \geq -\mathrm{K}_0$ and Theorem 28. This allows to concentrate the analysis on the set $\Upsilon_\circ(\mathrm{R}_0)$ on which Taylor expansions of the functional $\mathcal{L} : \mathbb{R}^{p^*} \to \mathbb{R}$ become accurate.

$C(\nabla)$: This set ensures that on $\Omega(\mathtt{x}) \subset \Omega$ all occurring random quadratic forms and stochastic errors are controlled by $\mathfrak{z}(\mathtt{x}) \in \mathbb{R}$. Consequently we can derive in the proof of Lemma 32 an a priori bound of the form $\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \boldsymbol{\upsilon}^*)\| \leq \mathtt{r}_k$ for a decreasing sequence of radii $(\mathtt{r}_k) \subset \mathbb{R}_+$ satisfying $\limsup_{k\to\infty} \mathtt{r}_k = \mathtt{C}\mathfrak{z}(\mathtt{x})$. Further this set allows to obtain in Lemma 34 the bounds for all $k \in \mathbb{N}$.

On $\Omega(\mathtt{x}) \subset \Omega$ we find $\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} \in \Upsilon_\circ(\mathtt{r}_k)$ such that we can follow the arguments of Theorem 2.2 of (Andresen and Spokoiny, 2014) to obtain the desired result with accuracy measured by $\check{\Diamond}_Q(\mathtt{r}_k, \mathtt{x})$.

The sketch in figure 4.2 illustrates the behavior of the first steps of the procedure. The axes correspond to the $\boldsymbol{\theta}$- or $\boldsymbol{\eta}$-subspaces respectively. The two ellipsoids with center $\boldsymbol{\upsilon}^*$ and solid frame represent the local sets $\Upsilon_\circ(R_K) \subset \Upsilon_\circ(\mathrm{R}_0)$, with $R_K > 0$ from Remark 5. We see that the initial guess $\widetilde{\boldsymbol{\upsilon}}_0$ lies in $\Upsilon_\circ(R)$. The elements $(\boldsymbol{\upsilon}_{k,k(+1)})$ of the alternating sequence all land inside of the respective $\Upsilon_\circ(\mathtt{r}_k)$, which are represented by shrinking ellipsoids centered in $\boldsymbol{\upsilon}^*$ with doted frames. Note that not the set $\Upsilon_\circ(R_K)$ but $\Upsilon_\circ(\mathrm{R}_0)$ contains all points of the sequence.

4.3 Probability of Desirable Set

Here we show that the set $\Omega(\mathtt{x})$ actually is of probability greater $1 - 8\mathrm{e}^{-\mathtt{x}} - \beta$. We prove the following two Lemmas, which together yield the claim.

Lemma 26 *The set $C(\nabla)$ satisfies*

$$\mathbb{P}(C(\nabla)) \geq 1 - 7\mathrm{e}^{-\mathtt{x}}.$$

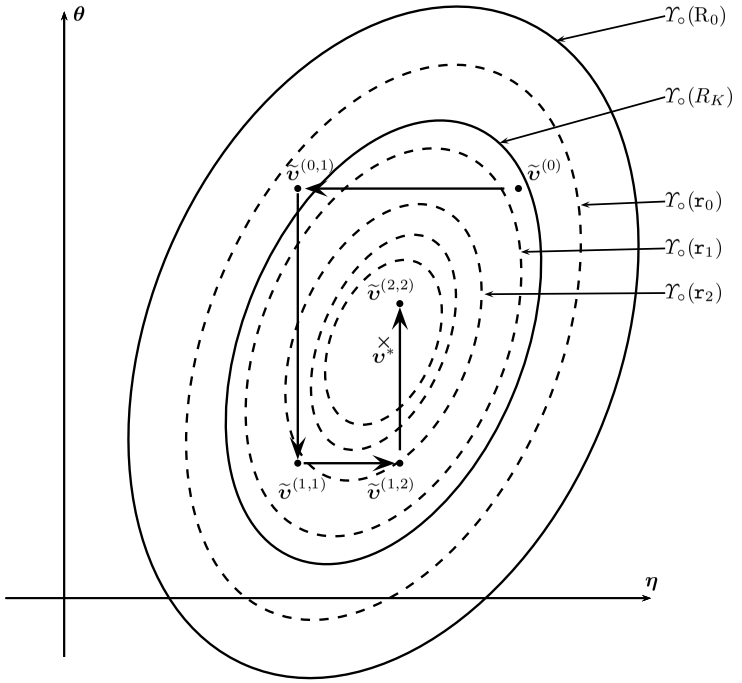


Figure 1: The behavior of the procedure for the first 4 steps of the alternating algorithm.

Proof The proof is similar to the proof of Theorem 3.1 in (Spokoiny, 2012). Denote

$$\mathcal{A} \stackrel{\text{def}}{=} \bigcap_{\mathtt{r}\le \mathrm{R}_0(\mathtt{x})} \left\{ \sup_{\upsilon\in\Upsilon_\circ(\mathtt{r})} \left\{ \frac{1}{6\epsilon\nu_1}\|\mathcal{Y}(\upsilon)\| - 2\mathtt{r}^2 \right\} \le \mathfrak{z}_Q(\mathtt{x}, 4p^*)^2 \right\}$$

$$\mathcal{B} \stackrel{\text{def}}{=} \bigcap_{\mathtt{r}\le 4\mathrm{R}_0(\mathtt{x})} \left\{ \sup_{\upsilon\in\Upsilon_\circ(\mathtt{r})} \left\{ \frac{1}{6\check{\epsilon}\check{\nu}_1}\|\check{\mathcal{Y}}(\upsilon)\| - 2\mathtt{r}^2 \right\} \le \mathfrak{z}_Q(\mathtt{x}, 2p^*+2p)^2 \right\}$$

$$\mathcal{C} \stackrel{\text{def}}{=} \Big\{ \max\{\|\mathcal{D}^{-1}\nabla\mathcal{L}\|, \|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}\|, \|\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\mathcal{L}\|\} \le \mathfrak{z}(\mathtt{x}) \Big\}.$$

We estimate

$$\begin{aligned}\mathbb{P}(C(\nabla)) \ge 1 &- \mathbb{P}\left(\mathcal{A}^c\right) - \mathbb{P}\left(\mathcal{B}^c\right) - \mathbb{P}\left(\mathcal{C}^c\right) \\ &- \mathbb{P}\left(\widetilde{\upsilon}, \widetilde{\upsilon}_{\boldsymbol{\theta}^*} \notin \Upsilon_\circ(\mathtt{r}_0)\right) - \mathbb{P}\left(\|\check{\boldsymbol{\xi}}\| > \mathfrak{z}(\mathtt{x}, \mathrm{Cov}(\check{\boldsymbol{\xi}}))\right).\end{aligned}$$

We bound using for both terms Theorem 42 which is applicable due to $(\mathcal{ED}_1)$ and $(\check{\mathcal{E}}\mathcal{D}_1)$:

$$\mathbb{P}(\mathcal{A}^c) \le \mathrm{e}^{-\mathtt{x}}, \;\; \mathbb{P}\left(\mathcal{B}^c\right) \le \mathrm{e}^{-\mathtt{x}}.$$

For the set $\mathcal{C} \subset \Omega$ observe that we can use $(\boldsymbol{\mathcal{I}})$ and Lemma 27 to find

$$\|\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\| \vee \|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\| \le \|\mathcal{D}^{-1}\nabla\|.$$

This implies that

$$\begin{aligned}&\{\|\mathcal{D}^{-1}\nabla\| \le \mathfrak{z}(\mathtt{x}, I\!\!B)\} \\ &\quad\subseteq \{\|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\| \vee \|\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\| \le \mathfrak{z}(\mathtt{x}, I\!\!B)\}.\end{aligned}$$

Using the deviation properties of quadratic forms as sketched in Section A we find

$$\mathbb{P}\left(\|\mathcal{D}^{-1}\nabla\| > \mathfrak{z}(\mathtt{x}, I\!\!B)\right) \le 2\mathrm{e}^{-\mathtt{x}}, \;\; \mathbb{P}\big(\|\check{\mathrm{D}}^{-1}\check{\nabla}\| > \mathfrak{z}(\mathtt{x}, \mathrm{Cov}(\check{\boldsymbol{\xi}}))\big) \le 2\mathrm{e}^{-\mathtt{x}}.$$

By the choice of $\mathfrak{z}(\mathtt{x}) > 0$ and $\mathtt{r}_0 > 0$ this gives the claim. ■

We cite Lemma B.2 of (Andresen and Spokoiny, 2014):

Lemma 27 *Let*

$$\mathcal{D}^2 = \begin{pmatrix} D^2 & A \\ A^\top & H^2 \end{pmatrix} \in \mathbb{R}^{(p+p)\times(p+p)}, \quad D \in \mathbb{R}^{p\times p},\ H \in \mathbb{R}^{m\times m} \text{ invertible},$$

$$\|D^{-1}AH^{-1}\| < 1.$$

Then for any $\boldsymbol{\upsilon} = (\boldsymbol{\theta}, \boldsymbol{\eta}) \in \mathbb{R}^{p+m}$ *we have* $\|H^{-1}\boldsymbol{\eta}\| \vee \|D^{-1}\boldsymbol{\theta}\| \le \|\mathcal{D}^{-1}\boldsymbol{\upsilon}\|$.

The next step is to show that the set $\bigcap_{k=1}^{K}(C_{k,k}\cap C_{k,k+1})$ has high probability, that is independent of the number of necessary steps. A close look at the proof of Theorem 4.1 of (Spokoiny, 2012) shows that it actually yields the following modified version:

Theorem 28 ((Spokoiny, 2012), Theorem 4.1) *Suppose* $(\mathcal{E}\mathtt{r})$ *and* $(\mathcal{L}\mathtt{r})$ *with* $\mathtt{b}(\mathtt{r})\equiv\mathtt{b}$. *Further define the following random set*

$$\Upsilon(K)\stackrel{\text{def}}{=}\{\boldsymbol{\upsilon}\in\Upsilon:\mathcal{L}(\boldsymbol{\upsilon},\boldsymbol{\upsilon}^*)\geq -K\}.$$

If for a fixed $\mathtt{r}_0$ *and any* $\mathtt{r}\geq\mathtt{r}_0$, *the following conditions are fulfilled:*

$$1+\sqrt{\mathtt{x}+2p^*}\leq 3\nu_{\mathtt{r}}^2\mathtt{g}(\mathtt{r})/\mathtt{b},$$
$$6\nu_{\mathtt{r}}\sqrt{\mathtt{x}+2p^*+\frac{\mathtt{b}}{9\nu_{\mathtt{r}}^2}K}\leq\mathtt{r}\mathtt{b},$$

then

$$\mathbb{P}(\Upsilon(K)\subseteq\Upsilon_\circ(\mathtt{r}_0))\geq 1-\mathrm{e}^{-\mathtt{x}}.$$

Note that with $(\mathcal{I})$

$$\|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k-\boldsymbol{\theta}^*)\|\vee\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_{k(+1)}-\boldsymbol{\eta}^*)\|\leq\frac{1}{1-\nu}\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}-\boldsymbol{\upsilon}^*)\|.$$

With assumption (B_1) and

$$\mathrm{R}_0(\mathtt{x})=\frac{6\nu_{\mathtt{r}}}{\mathtt{b}(1-\nu)}\sqrt{\mathtt{x}+\mathbb{Q}+\frac{\mathtt{b}}{9\nu_{\mathtt{r}}^2}\mathrm{K}_0},$$

this implies the desired result as $\mathcal{L}(\boldsymbol{\upsilon}_{k,k(+1)},\boldsymbol{\upsilon}^*)\geq\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0,\boldsymbol{\upsilon}^*)$ such that with Theorem 28

$$\begin{aligned}\mathbb{P}\left(\bigcap_{k=0}^{K}(C_{k,k}\cap C_{k,k+1})\right)&\geq\mathbb{P}\left(\bigcap_{k=0}^{K}(C_{k,k}\cap C_{k,k+1})\cap\{\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0,\boldsymbol{\upsilon}^*)\geq-\mathrm{K}_0\}\right)\\&\quad-\mathbb{P}(\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0,\boldsymbol{\upsilon}^*)\leq-\mathrm{K}_0)\\&\geq\mathbb{P}\left\{\Upsilon(\mathrm{K}_0)\subset\Upsilon_\circ\Big((1-\nu)\mathrm{R}_0(\mathtt{x})\Big)\right\}-\beta_{(\boldsymbol{A})}\\&\geq 1-\mathrm{e}^{-\mathtt{x}}-\beta_{(\boldsymbol{A})}.\end{aligned}$$

Remark 29 *This also shows that the sets of maximizers* $(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)})$ *are nonempty and well defined since the maximization always takes place on compact sets of the form* $\{\boldsymbol{\theta}\in\mathbb{R}^p,(\boldsymbol{\theta},\boldsymbol{\eta})\in\Upsilon_\circ(R_0)\}$ *or* $\{\boldsymbol{\eta}\in\mathbb{R}^m,(\boldsymbol{\theta},\boldsymbol{\eta})\in\Upsilon_\circ(R_0)\}$.

Remark 30 *To address the claim of Remark 5 we present the following lemma:*

Lemma 31 *On the set* $C(\nabla)\cap\{\widetilde{\boldsymbol{\upsilon}}_0\in\Upsilon_\circ(R_K)\}$ *it holds*

$$\mathcal{L}(\boldsymbol{\upsilon}_0,\boldsymbol{\upsilon}^*)\geq-(1/2+\epsilon(1+12\nu_0))(R+\mathfrak{z}(\mathtt{x}))^2.$$

Proof *With similar arguments as in the proof of Lemma 32 we have on* $C(\nabla)\subset\Omega$ *that*

$$\begin{aligned}&\mathcal{L}(\boldsymbol{\upsilon}_0)-\mathcal{L}(\boldsymbol{\upsilon}^*)\\&\quad\geq\mathbb{E}[\mathcal{L}(\boldsymbol{\upsilon}_0)-\mathcal{L}(\boldsymbol{\upsilon}^*)]-\|\mathcal{D}^{-1}\nabla\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\|R-|\{\nabla\boldsymbol{\zeta}(\widehat{\boldsymbol{\upsilon}})-\nabla\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\}(\boldsymbol{\upsilon}_0-\boldsymbol{\upsilon}^*)|\\&\quad\geq-\|\mathcal{D}(\boldsymbol{\upsilon}_0-\boldsymbol{\upsilon}^*)\|^2/2-\|\mathcal{D}^{-1}\nabla\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\|R\\&\qquad-\|\mathcal{D}^{-1}\big\{\nabla\mathcal{L}(\widehat{\boldsymbol{\upsilon}})-\nabla\mathcal{L}(\boldsymbol{\upsilon}^*)\big\}\|R-\epsilon R_K^2\\&\quad\geq-R^2/2-\mathfrak{z}(\mathtt{x})R-\epsilon(1+12\nu_0)\left(R^2+\mathfrak{z}(\mathtt{x})^2\right).\end{aligned}$$

■

4.4 Proof Convergence

We derive the a priori bound $\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}\in\Upsilon_\circ(\mathtt{r}_k)$ with an adequately decreasing sequence $(\mathtt{r}_k)\subset\mathbb{R}_+$ using the argument of Section 4.1, where $\limsup\mathtt{r}_k\approx\mathfrak{z}(\mathtt{x})$. For this purpose we define the *parametric uniform spread*

$$\diamondsuit_Q(\mathtt{r},\mathtt{x})\stackrel{\text{def}}{=}\epsilon\left\{2\mathtt{r}^2+\mathfrak{z}_Q(\mathtt{x},4p^*)^2\right\}(1+6\nu_1^2).\tag{31}$$

Lemma 32 *Assume that for some sequence* $(\mathtt{r}_k^{(l)})_{k\in\mathbb{N}}$

$$\Omega(\mathtt{x})\subseteq\bigcap_{k\in\mathbb{N}}\left\{\boldsymbol{\upsilon}_{k,k(+1)}\in\Upsilon_\circ\left(\mathtt{r}_k^{(l)}\right)\right\}.$$

Then under the assumptions of Theorem 7 we get on $\Omega(\mathtt{x})$ *for all* $k\in\mathbb{N}_0$

$$\begin{aligned}\left\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)}-\boldsymbol{\upsilon}^*)\right\|&\leq 2\sqrt{2}(1-\sqrt{\nu})^{-1}\left(\mathfrak{z}(\mathtt{x})+(1+\sqrt{\nu})\nu^k R_0(\mathtt{x})\right)\\&\quad+2\sqrt{2}(1+\sqrt{\nu})\sum_{r=0}^{k-1}\nu^r\diamondsuit_Q\left(\mathtt{r}_r^{(l)}\right)\\&=:\mathtt{r}_k^{(l+1)}.\end{aligned}$$

Proof

1. We first show that on $\Omega(\mathtt{x})$

$$\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k-\boldsymbol{\theta}^*)=\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*)-\mathrm{D}^{-1}\mathcal{A}(\widetilde{\boldsymbol{\eta}}_k-\boldsymbol{\eta}^*)+\boldsymbol{\tau}\big(\mathtt{r}_k^{(l)}\big),\tag{32}$$
$$\mathrm{H}(\widetilde{\boldsymbol{\eta}}_k-\boldsymbol{\eta}^*)=\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\mathcal{L}(\boldsymbol{\upsilon}^*)-\mathrm{H}^{-1}\mathcal{A}^\top(\widetilde{\boldsymbol{\theta}}_{k-1}-\boldsymbol{\theta}^*)+\boldsymbol{\tau}\big(\mathtt{r}_k^{(l)}\big),$$

where - with $\diamondsuit_Q(\mathtt{r},\mathtt{x})$ defined in (31) -

$$\|\boldsymbol{\tau}(\mathtt{r})\| \leq \diamondsuit_Q(\mathtt{r},\mathtt{x}).$$

The proof is the same in each step for both statements such that we only prove the first one. The arguments presented here are similar to those of Theorem D.1 in (Andresen and Spokoiny, 2014). By assumption on $\Omega(\mathtt{x})$ we have $\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} \in \Upsilon_\circ\big(\mathtt{r}_k^{(l)}\big)$. Define with $\zeta = \mathcal{L} - \mathbb{E}\mathcal{L}$

$$\alpha(\boldsymbol{\upsilon},\boldsymbol{\upsilon}^*) := \mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\boldsymbol{\upsilon}^*) - \big(\nabla\zeta(\boldsymbol{\upsilon}^*)(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*) - \|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\|^2/2\big).$$

Note that

$$\begin{aligned}\mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\boldsymbol{\upsilon}^*) &= \nabla\zeta(\boldsymbol{\upsilon}^*)(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*) - \|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\|^2/2 + \alpha(\boldsymbol{\upsilon},\boldsymbol{\upsilon}^*)\\ &= \nabla_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*)(\boldsymbol{\theta}-\boldsymbol{\theta}^*) - \|\mathrm{D}(\boldsymbol{\theta}-\boldsymbol{\theta}^*)\|^2/2 + (\boldsymbol{\theta}-\boldsymbol{\theta}^*)^\top\mathcal{A}(\boldsymbol{\eta}-\boldsymbol{\eta}^*)\\ &\quad +\nabla_{\boldsymbol{\eta}}\zeta(\boldsymbol{\upsilon}^*)(\boldsymbol{\eta}-\boldsymbol{\eta}^*) - \|\mathrm{H}(\boldsymbol{\eta}-\boldsymbol{\eta}^*)\|^2/2 + \alpha(\boldsymbol{\upsilon},\boldsymbol{\upsilon}^*).\end{aligned}$$

Setting $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\widetilde{\boldsymbol{\theta}}_k,\widetilde{\boldsymbol{\eta}}_k) = 0$ we find

$$\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) - \mathcal{D}^{-1}\big(\nabla_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*) - \mathcal{A}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*)\big) = \mathcal{D}^{-1}\nabla_{\boldsymbol{\theta}}\alpha(\widetilde{\boldsymbol{\upsilon}}_{k,k},\boldsymbol{\upsilon}^*).$$

As we assume that $\widetilde{\boldsymbol{\upsilon}}_{k,k} \in \Upsilon_\circ(\mathrm{R}_0)$ it suffices to show that with dominating probability

$$\sup_{(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\in\Upsilon_\circ(\mathrm{R}_0)} \|\mathcal{U}_{\boldsymbol{\theta}}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\| \leq \diamondsuit(\mathtt{r}_k^{(l)}),$$

where

$$\mathcal{U}_{\boldsymbol{\theta}}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k) \stackrel{\mathrm{def}}{=} \mathrm{D}^{-1}\big\{\nabla_{\boldsymbol{\theta}}\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_{k,k}) - \nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*) - \mathrm{D}^2(\boldsymbol{\theta}-\boldsymbol{\theta}^*) - \mathcal{A}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*)\big\}.$$

To see this note first that with Lemma 27 $\|\mathrm{D}^{-1}\Pi_{\boldsymbol{\theta}}\mathcal{D}\boldsymbol{\upsilon}\| \leq \|\mathcal{D}^{-1}\mathcal{D}\boldsymbol{\upsilon}\|$. This gives by condition $(\boldsymbol{\mathcal{L}_0})$, Lemma 27 and Taylor expansion

$$\begin{aligned}\sup_{(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\in\Upsilon_\circ(\mathtt{r})} \|\mathbb{E}\mathcal{U}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\| &\leq \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \|\mathrm{D}^{-1}\Pi_{\boldsymbol{\theta}}\Big(\nabla\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}) - \nabla\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}^*) - \mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\Big)\|\\ &\leq \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \|\mathrm{D}^{-1}\Pi_{\boldsymbol{\theta}}\mathcal{D}\|\|\mathcal{D}^{-1}\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})^2\mathcal{D}^{-1} - I_{p^*}\|^{1/2}\mathtt{r}\\ &\leq \epsilon\mathtt{r}^2.\end{aligned}$$

For the remainder note that again with Lemma 27

$$\|\mathrm{D}^{-1}\Big(\nabla_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}) - \nabla_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*)\Big)\| \leq \|\mathcal{D}^{-1}\Big(\nabla\zeta(\boldsymbol{\upsilon}) - \nabla\zeta(\boldsymbol{\upsilon}^*)\Big)\|.$$

This yields that on $\Omega(\mathtt{x})$

$$\begin{aligned}\sup_{(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\in\Upsilon_\circ(\mathtt{r})} \Big\|\mathcal{U}_{\boldsymbol{\theta}}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k) - \mathbb{E}\mathcal{U}_{\boldsymbol{\theta}}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\Big\| &\leq \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \Big\|\mathrm{D}^{-1}\Big(\nabla_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}) - \nabla_{\boldsymbol{\theta}}\zeta(\boldsymbol{\upsilon}^*)\Big)\Big\|\\ &\leq \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \Big\{\frac{1}{6\nu_1\epsilon}\|\mathcal{Y}(\boldsymbol{\upsilon})\|\Big\} 6\nu_1\epsilon \leq 6\nu_1\epsilon\big\{\mathfrak{z}_Q(\mathtt{x},4p^*) + 2\mathtt{r}^2\big\}.\end{aligned}$$

Using the same argument for $\widetilde{\boldsymbol{\eta}}_k$ gives the claim.

2. We prove the apriori bound for the distance of the k. estimator to the oracle

$$\big\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \boldsymbol{\upsilon}^*)\big\| \leq \mathtt{r}_k^{(l+1)}.$$

To see this we first use the inequality

$$\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \boldsymbol{\upsilon}^*)\| \leq \sqrt{2}\|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*)\| + \sqrt{2}\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_{k(+1)} - \boldsymbol{\eta}^*)\|.$$

Now we find with (32)

$$\begin{aligned}\|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*)\| &\leq \|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*)\| + \|\mathrm{D}^{-1}\mathcal{A}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*)\| + \|\boldsymbol{\tau}\big(\mathtt{r}_k^{(l)}\big)\|\\ &\leq \|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*)\| + \|\mathrm{D}^{-1}\mathcal{A}\mathrm{H}^{-1}\|\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*)\| + \|\boldsymbol{\tau}\big(\mathtt{r}_k^{(l)}\big)\|.\end{aligned}$$

Next we use that on $\Omega(\mathtt{x})$

$$\|\mathrm{D}^{-1}\mathcal{A}\mathrm{H}^{-1}\| \leq \sqrt{\nu},\ \|\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}^*)\| \leq \mathfrak{z}(\mathtt{x}),\ \|\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\mathcal{L}(\boldsymbol{\upsilon}^*)\| \leq \mathfrak{z}(\mathtt{x}),$$

and

$$\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*)\| \leq \|\mathrm{H}^{-1}\nabla_{\boldsymbol{\eta}}\mathcal{L}(\boldsymbol{\upsilon}^*)\| + \|\mathrm{H}^{-1}\mathcal{A}^\top(\widetilde{\boldsymbol{\theta}}_{k-1} - \boldsymbol{\theta}^*)\| + \|\boldsymbol{\tau}\big(\mathtt{r}_k^{(l)}\big)\|,$$

to derive the recursive formula

$$\|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*)\| \leq (1+\sqrt{\nu})\Big(\mathfrak{z}(\mathtt{x}) + \|\boldsymbol{\tau}\big(\mathtt{r}_k^{(l)}\big)\|\Big) + \nu\|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_{k-1} - \boldsymbol{\theta}^*)\|.$$

Deriving the analogous formula for $\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*)\|$ and solving the recursion gives the claim. ■

Lemma 33 *Assume the same as in Theorem 7. Further assume that* (17) *is met with* $\mathtt{C}(\nu)$ *defined in* (18)*. Then*

$$\Omega(\mathtt{x}) \subseteq \bigcap_{k\in\mathbb{N}} \big\{\boldsymbol{\upsilon}_{k,k(+1)} \in \Upsilon_\circ(\mathtt{r}_k^*)\big\},$$

where - with $\mathtt{C}(\nu) \geq 0$ *defined in* (18) -

$$\begin{aligned}\mathtt{r}_k^* \leq & \left(\mathtt{C}(\nu) + \frac{4\epsilon\mathtt{C}(\nu)^4\mathfrak{z}(\mathtt{x})}{1-\epsilon\mathtt{C}(\nu)\mathfrak{z}(\mathtt{x})}\right)2\mathfrak{z}(\mathtt{x}) \\ & +\nu^k\left(\mathtt{C}(\nu) + \frac{4\epsilon\mathtt{C}(\nu)^4 R_0}{1-\epsilon\mathtt{C}(\nu)R_0}\right)R_0.\end{aligned} \tag{33}$$

Proof We proof this claim via induction. On $\Omega(\mathtt{x})$ we have

$$\boldsymbol{v}_{k,k(+1)} \in \Upsilon_\circ(\mathrm{R}_0), \quad \text{set } \mathtt{r}_k^{(0)} \stackrel{\text{def}}{=} \mathrm{R}_0.$$

Now with Lemma 32 we find that if

$$\Omega(\mathtt{x}) \subseteq \bigcap_{k\in\mathbb{N}}\left\{\boldsymbol{v}_{k,k(+1)} \in \Upsilon_\circ(\mathtt{r}_k^{(l)})\right\},$$

that then

$$\Omega(\mathtt{x}) \subseteq \bigcap_{k\in\mathbb{N}}\left\{\boldsymbol{v}_{k,k(+1)} \in \Upsilon_\circ(\mathtt{r}_k^{(l+1)})\right\},$$

where

$$\begin{aligned}\mathtt{r}_k^{(l)} \leq & \, 2\sqrt{2}(1-\sqrt{\nu})^{-1}\left(\mathfrak{z}(\mathtt{x}) + (1+\sqrt{\nu})\nu^k\mathrm{R}_0(\mathtt{x})\right) \\ & +2\sqrt{2}(1+\sqrt{\nu})\sum_{r=0}^{k-1}\nu^r \diamondsuit_Q\left(\mathtt{r}_{k-r}^{(l-1)}, \mathtt{x}\right).\end{aligned}$$

Setting $l = 1$ this gives

$$\mathtt{r}_k^{(1)} \leq 2\sqrt{2}(1-\sqrt{\nu})^{-1}\left\{(\mathfrak{z}(\mathtt{x}) + \diamondsuit_Q(\mathrm{R}_0, \mathtt{x})) + (1+\sqrt{\nu})\nu^k\mathrm{R}_0(\mathtt{x})\right\}.$$

We show that

$$\Omega(\mathtt{x}) \subseteq \bigcap_{k\in\mathbb{N}}\left\{\boldsymbol{v}_{k,k(+1)} \in \Upsilon_\circ\left(\limsup_{l\to\infty}\mathtt{r}_k^{(l)}\right)\right\} \subseteq \bigcap_{k\in\mathbb{N}}\left\{\boldsymbol{v}_{k,k(+1)} \in \Upsilon_\circ(\mathtt{r}_k^*)\right\}.$$

So we have to show that $\limsup_{l\to\infty}\mathtt{r}_k^{(l)} \leq \mathtt{r}_k^*$ in (33). For this we estimate further

$$\begin{aligned}\mathtt{r}_k^{(l)} \leq & \, 2\sqrt{2}(1-\sqrt{\nu})^{-1}\left(\mathfrak{z}(\mathtt{x}) + (1+\sqrt{\nu})\nu^k\mathrm{R}_0(\mathtt{x})\right) \\ & +2\sqrt{2}(1+\sqrt{\nu})\epsilon\sum_{r=0}^{k-1}\nu^r\left(\left(\mathtt{r}_{k-r}^{(l-1)}\right)^2 + \mathfrak{z}(\mathtt{x})^2\right) \\ \leq & \, 2\sqrt{2}(1-\sqrt{\nu})^{-1}\left(\mathfrak{z}(\mathtt{x}) + \epsilon\mathfrak{z}(\mathtt{x})^2 + (1+\sqrt{\nu})\nu^k\mathrm{R}_0(\mathtt{x})\right) \\ & +2\sqrt{2}(1+\sqrt{\nu})\epsilon\sum_{r=0}^{k-1}\nu^r\left(\mathtt{r}_{k-r}^{(l-1)}\right)^2 \\ \leq & \, \mathtt{C}(\nu)\left\{\left(\mathfrak{z}(\mathtt{x}) + \epsilon\mathfrak{z}(\mathtt{x})^2\right) + \nu^k\mathrm{R}_0 + \epsilon\sum_{r=0}^{k-1}\nu^r\left(\mathtt{r}_{k-r}^{(l-1)}\right)^2\right\},\end{aligned}$$

where $\mathtt{C}(\nu) > 0$ is defined in (18). We set

$$A_{s,k}^{(l)} \stackrel{\text{def}}{=} \sum_{r_1=0}^{k-1}\nu^{r_1}\left(\sum_{r_2=0}^{k-r_1-1}\nu^{r_2}\left(\dots\sum_{r_s=0}^{k-r_1-\dots-r_{s-1}-1}\nu^{r_s}\left(\mathtt{r}_{k-r_1-\dots-r_s}^{(l-1)}\right)^2\dots\right)^2\right)^2.$$

Claim

$$\begin{aligned}A_{s,k}^{(l)} \leq & \, 4^{\sum_{t=0}^{s-1}2^t}\mathtt{C}(\nu)^{2^s}\left\{\left(\frac{1}{1-\nu}\right)^{\sum_{t=0}^{s-1}2^t}\left(\mathfrak{z}(\mathtt{x}) + \epsilon\mathfrak{z}(\mathtt{x})^2\right)^{2^s}\right. \\ & \left.+\nu^k\left(\frac{1}{\nu^{-1}-1}\right)^{\sum_{t=0}^{s-1}2^t}\mathrm{R}_0^{2^s}\right\} \\ & +4^{\sum_{t=0}^{s-1}2^t}(\mathtt{C}(\nu)\epsilon)^{2^s}A_{s+1,k}^{(l-1)}.\end{aligned} \tag{34}$$

We proof this claim via induction. Clearly

$$\begin{aligned}A_{1,k}^{(l)} = & \sum_{r_1=0}^{k-1}\nu^{r_1}\left(\mathtt{r}_{k-r_1}^{(l-1)}\right)^2 \leq 4\mathtt{C}(\nu)^2\sum_{r_1=0}^{k-1}\nu^{r_1}\left\{\left(\mathfrak{z}(\mathtt{x}) + \epsilon\mathfrak{z}(\mathtt{x})^2\right)^2 + \nu^{2(k-r_1)}\mathrm{R}_0^2\right\} \\ & +4\mathtt{C}(\nu)^2\epsilon^2\sum_{r_1=0}^{k-1}\nu^{r_1}\left(\sum_{r_2=0}^{k-r_1-r_2-1}\nu^{r_2}\left(\mathtt{r}_{k-r_1-r_2}^{(l-2)}\right)^2\right)^2 \\ \leq & \, 4\mathtt{C}(\nu)^2\left\{\frac{1}{1-\nu}\left(\mathfrak{z}(\mathtt{x}) + \epsilon\mathfrak{z}(\mathtt{x})^2\right)^2 + \frac{\nu^k}{\nu^{-1}-1}\mathrm{R}_0^2\right\} \\ & +4\mathtt{C}(\nu)^2\epsilon^2A_{2,k}^{(l-1)}.\end{aligned}$$

Furthermore

$$A_{s,k}^{(l)} \stackrel{\text{def}}{=} \sum_{r_1=0}^{k-1} \nu^{r_1} \left(\sum_{r_2=0}^{k-r_1-1} \nu^{r_2} \left(\dots \sum_{r_s=0}^{k-r_1-\dots-r_{s-1}-1} \nu^{r_s} \left(\mathtt{r}_{k-r_1-\dots-r_s}^{(l-1)} \right)^2 \dots \right)^2 \right)^2$$
$$= \sum_{r_1=0}^{k-1} \nu^{r_1} \left(A_{s-1,k-r_1}^{(l)} \right)^2. \tag{35}$$

Plugging in (34) we get for $s \geq 2$

$$A_{s,k}^{(l)} \leq \sum_{r_1=0}^{k-1} \nu^{r_1} \left(4^{\sum_{t=0}^{s-2} 2^t} \mathtt{C}(\nu)^{2^{s-1}} \left\{ \left(\frac{1}{1-\nu} \right)^{\sum_{t=0}^{s-2} 2^t} \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right)^{2^{s-1}} \right.\right.$$
$$\left.\left. + \nu^k \left(\frac{1}{\nu^{-1}-1} \right)^{\sum_{t=0}^{s-2} 2^t} \mathrm{R}_0^{2^{s-1}} \right\} \right.$$
$$\left. + 4^{\sum_{t=0}^{s-2} 2^t} (\mathtt{C}(\nu)\epsilon)^{2^{s-1}} A_{s,k-r_1}^{(l-1)} \right)^2.$$

Shifting the index this gives

$$A_{s,k}^{(l)} \leq 4 \sum_{r_1=0}^{k-1} \nu^{r_1} \left(4^{\sum_{t=1}^{s-1} 2^t} \mathtt{C}(\nu)^{2^s} \left\{ \left(\frac{1}{1-\nu} \right)^{\sum_{t=1}^{s-1} 2^{t-1}} \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right)^{2^s} \right.\right.$$
$$\left.\left. + \nu^k \left(\frac{1}{\nu^{-1}-1} \right)^{\sum_{t=1}^{s-1} 2^t} \mathrm{R}_0^{2^s} \right\} \right.$$
$$\left. + 4^{\sum_{t=1}^{s-1} 2^t} (\mathtt{C}(\nu)\epsilon)^{2^s} (A_{s,k-r_1}^{(l-1)})^2 \right).$$

Direct calculation then leads to

$$A_{s,k}^{(l)} \leq 4^{\sum_{t=0}^{s-1} 2^t} \mathtt{C}(\nu)^{2^s} \left\{ \left(\frac{1}{1-\nu} \right)^{\sum_{t=0}^{s-1} 2^t} \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right)^{2^s} \right.$$
$$\left. + \nu^k \left(\frac{1}{\nu^{-1}-1} \right)^{\sum_{t=0}^{s-1} 2^t} \mathrm{R}_0^{2^s} \right\}$$
$$+ 4^{\sum_{t=0}^{s-1} 2^t} (\mathtt{C}(\nu)\epsilon)^{2^s} \sum_{r_1=0}^{k-1} \nu^{r_1} (A_{s,k-r_1}^{(l-1)})^2,$$

which gives (34) with (35). Similarly we can prove

$$A_{s,k}^{(1)} = \left(\frac{1}{1-\nu} \right)^{2^s-1} \mathrm{R}_0^{2^s}.$$

Abbreviate

$$\lambda_s \stackrel{\text{def}}{=} 4^{2^s-1} \mathtt{C}(\nu)^{2^s}, \quad \beta_s \stackrel{\text{def}}{=} 4^{2^s-1} (\mathtt{C}(\nu)\epsilon)^{2^s},$$
$$\mathfrak{z}_s(\mathtt{x}) \stackrel{\text{def}}{=} \left(\frac{1}{1-\nu} \right)^{2^s-1} \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right)^{2^s}, \quad \mathrm{R}_s \stackrel{\text{def}}{=} \left(\frac{1}{\nu^{-1}-1} \right)^{2^s-1} \mathrm{R}_0^{2^s}.$$

Then

$$\mathtt{r}_k^{(l)} \leq \mathtt{C}(\nu) \left\{ \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right) + \nu^k \mathrm{R}_0 + \epsilon A_{1,k}^{(l)} \right\}$$
$$\leq \sum_{s=0}^{l-1} \lambda_s \prod_{r=0}^{s-1} \beta_r \mathfrak{z}_s(\mathtt{x}) + \nu^k \sum_{s=0}^{l-1} \lambda_s \prod_{r=0}^{s-1} \beta_r \mathrm{R}_s + \prod_{r=0}^{l-1} \beta_r \mathrm{R}_l. \tag{36}$$

We estimate further

$$\sum_{s=0}^{l-1} \lambda_s \prod_{r=0}^{s-1} \beta_r \mathfrak{z}_s(\mathtt{x}) - \mathtt{C}(\nu) \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right) = \sum_{s=1}^{l-1} \lambda_s \prod_{r=0}^{s-1} \beta_r \mathfrak{z}_s(\mathtt{x})$$
$$\leq \sum_{s=1}^{l-1} 4^{2^s} \mathtt{C}(\nu)^{2^{s+1}} \epsilon^{2^s-1} \left(\frac{1}{1-\nu} \right)^{2^s-1} \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right)^{2^s}$$
$$= \epsilon 4^2 \mathtt{C}(\nu)^4 \left(\frac{1}{1-\nu} \right) \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right)^2$$
$$\sum_{s=1}^{l-1} \left(\epsilon 4 \mathtt{C}(\nu) \frac{1}{1-\nu} \left(\mathfrak{z}(\mathtt{x}) + \epsilon \mathfrak{z}(\mathtt{x})^2 \right) \right)^{2^s-1}.$$

Assuming (17) and the definition of $\mathrm{R}_0 > \mathfrak{z}(\mathtt{x})$ this gives

$$\sum_{s=0}^{l-1} \lambda_s \prod_{r=0}^{s-1} \beta_r \mathfrak{z}_s(\mathtt{x}) \leq \left(\mathtt{C}(\nu) + \frac{4\epsilon \mathtt{C}(\nu)^4 \mathfrak{z}(\mathtt{x})}{1 - \epsilon \mathtt{C}(\nu) \mathfrak{z}(\mathtt{x})} \right) 2\mathfrak{z}(\mathtt{x}).$$

With the same argument we find under (17) that

$$\nu^k \sum_{s=0}^{l-1} \lambda_s \prod_{r=0}^{s-1} \beta_r \mathrm{R}_s \leq \nu^k \left(\mathtt{C}(\nu) + \frac{4\epsilon \mathtt{C}(\nu)^4 \mathrm{R}_0}{1 - \epsilon \mathtt{C}(\nu) \mathrm{R}_0} \right) \mathrm{R}_0.$$

Additionally (17) implies

$$\prod_{r=0}^{l-1} \beta_r \mathrm{R}_r \leq \left(\epsilon 4 \mathtt{C}(\nu) \frac{1}{\nu^{-1}-1} \right)^{2^{l-1}} \mathrm{R}_0^{2^l} \to 0.$$

Plugging these bounds into (36) and letting $l \to \infty$ gives the claim. ■

4.5 Result after Convergence

In the previous section we showed that

$$\Omega(\mathtt{x}) \subset \bigcap_{\mathtt{r}\leq 4\mathrm{R}_0(\mathtt{x})} \left\{ \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \left\{ \frac{1}{6\breve{\epsilon}\breve{\nu}_1}\|\breve{\mathcal{Y}}(\boldsymbol{\upsilon})\| - 2\mathtt{r}^2 \right\} \leq \mathfrak{z}_Q(\mathtt{x}, 2p^* + 2p)^2 \right\}$$
$$\cap \bigcap_{k\in\mathbb{N}} \{\boldsymbol{\upsilon}_{k,k} \in \Upsilon_\circ(\mathtt{r}_k^*),\, \boldsymbol{\upsilon}_{k,k+1} \in \Upsilon_\circ(\mathtt{r}_k^*)\} \cap \{\widetilde{\boldsymbol{\upsilon}}, \widetilde{\boldsymbol{\upsilon}}_{\boldsymbol{\theta}^*} \in \Upsilon_\circ(\mathtt{r}_0)\},$$

where $\mathtt{r}_k^*$ is defined in (33). The claim of Theorem 7 follows with the following lemma:

Lemma 34 *Assume* $(\breve{\mathcal{E}}\mathcal{D}_1)$, $(\breve{\mathcal{L}}_0)$, *and* $(\mathcal{I})$. *Then it holds on* $\Omega(\mathtt{x}) \subseteq \epsilon$ *that for all* $k \in \mathbb{N}$

$$\left\|\breve{D}\big(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*\big) - \breve{\boldsymbol{\xi}}\right\| \leq \breve{\Diamond}_Q(\mathtt{r}_k, \mathtt{x}), \tag{37}$$
$$\left|2\breve{L}(\widetilde{\boldsymbol{\theta}}_k, \boldsymbol{\theta}^*) - \|\breve{\boldsymbol{\xi}}\|^2\right| \leq 5\left(\|\breve{D}^{-1}\breve{\nabla}\| + \breve{\Diamond}_Q(\mathtt{r}_k, \mathtt{x})\right)\breve{\Diamond}_Q(\mathtt{r}_k, \mathtt{x}), \tag{38}$$

where the spread $\breve{\Diamond}(\mathtt{r}, \mathtt{x})$ *is defined in* (21) *and where*

$$\mathtt{r}_k \stackrel{\mathrm{def}}{=} \mathtt{r}_k^* \vee \mathtt{r}_0.$$

Proof The proof is nearly the same as that of Theorem 2.2 of (Andresen and Spokoiny, 2014) which is inspired by the proof of Theorem 1 of (Murphy and van der Vaart, 2000). So we only sketch it and refer the reader to (Andresen and Spokoiny, 2014) for the skipped arguments. We define

$$l : \mathbb{R}^p \times \Upsilon \to \mathbb{R}, \;\; (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\eta}) \mapsto \mathcal{L}(\boldsymbol{\theta}_1, \boldsymbol{\eta} + \mathrm{H}^{-2}\mathcal{A}^\top(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)).$$

Note that

$$\nabla_{\boldsymbol{\theta}_1} l(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\eta}) = \breve{\nabla}_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_1, \boldsymbol{\eta} + \mathrm{H}^{-2}\mathcal{A}^\top(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)), \;\; \widetilde{\boldsymbol{\theta}}_k = \operatorname*{argmax}_{\boldsymbol{\theta}} l(\boldsymbol{\theta}, \widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k),$$

such that $\breve{\nabla}_{\boldsymbol{\theta}}\mathcal{L}(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k) = 0$. This gives

$$\left\|\breve{\mathrm{D}}\big(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*\big) - \breve{\boldsymbol{\xi}}\right\| = \left\|\breve{\mathrm{D}}^{-1}\breve{\nabla}\mathcal{L}(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k) - \breve{\mathrm{D}}^{-1}\breve{\nabla}\mathcal{L}(\boldsymbol{\upsilon}^*) + \breve{\mathrm{D}}\big(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*\big)\right\|.$$

Now the right hand side can be bounded just as in the proof of Theorem 2.2 of (Andresen and Spokoiny, 2014). This gives (37).

For (38) we can represent:

$$\breve{L}(\widetilde{\boldsymbol{\theta}}_k) - \breve{L}(\boldsymbol{\theta}^*) = l(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_{k+1}) - l(\boldsymbol{\theta}^*, \boldsymbol{\theta}^*, \widetilde{\boldsymbol{\eta}}_{\boldsymbol{\theta}^*}),$$

where

$$\widetilde{\boldsymbol{\eta}}_{\boldsymbol{\theta}^*} \stackrel{\mathrm{def}}{=} \Pi_{\boldsymbol{\eta}} \operatorname*{argmax}_{\substack{\boldsymbol{\upsilon}\in\Upsilon,\\ \Pi_{\boldsymbol{\theta}}\boldsymbol{\upsilon}=\boldsymbol{\theta}^*}} \mathcal{L}(\boldsymbol{\upsilon}).$$

Due to the definition of $\widetilde{\boldsymbol{\theta}}_k$ and $\widetilde{\boldsymbol{\eta}}_{k+1}$

$$l(\widetilde{\boldsymbol{\theta}}_k, \boldsymbol{\theta}^*, \widetilde{\boldsymbol{\eta}}_{\boldsymbol{\theta}^*}) - l(\boldsymbol{\theta}^*, \boldsymbol{\theta}^*, \widetilde{\boldsymbol{\eta}}_{\boldsymbol{\theta}^*}) \leq \breve{L}(\widetilde{\boldsymbol{\theta}}_k) - \breve{L}(\boldsymbol{\theta}^*) \leq l(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_{k+1}) - l(\boldsymbol{\theta}^*, \widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_{k+1}).$$

Again the remaining steps are exactly the same as in the proof of Theorem 2.2 of (Andresen and Spokoiny, 2014).

■

5. Proof of Corollary 13

Proof Note that with the argument of Section 4.3 $\mathbb{P}(\epsilon'(\mathtt{x})) \geq 1 - 8\mathrm{e}^{-\mathtt{x}} - \beta$ where with $\Omega(\mathtt{x})$ from (29)

$$\epsilon'(\mathtt{x}) = \Omega(\mathtt{x}) \cap \{\widetilde{\boldsymbol{\upsilon}} \in \Upsilon_\circ(\mathtt{r}_0)\}.$$

On $\epsilon'(\mathtt{x})$ it holds due to Theorem 7 and due to Theorem 2.1 of (Andresen and Spokoiny, 2014)

$$\|\breve{\mathrm{D}}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) - \breve{\boldsymbol{\xi}}\| \leq \breve{\Diamond}_Q(\mathtt{r}_k, \mathtt{x}), \;\; \|\breve{\mathrm{D}}(\widetilde{\boldsymbol{\theta}} - \boldsymbol{\theta}^*) - \breve{\boldsymbol{\xi}}\| \leq \breve{\Diamond}(\mathtt{r}_0, \mathtt{x}).$$

Now the claim follows with the triangular inequality and noting that $\breve{\Diamond}(\mathtt{r}_0, \mathtt{x}) \leq \breve{\Diamond}_Q(\mathtt{r}_0, \mathtt{x})$.
■

6. Proof of Theorem 14

We prove this Theorem in a similar manner to the convergence result in Lemma 32. Redefine the set $\Omega(\mathtt{x})$

$$\Omega(\mathtt{x}) \stackrel{\mathrm{def}}{=} \bigcap_{k=0}^{K} (C_{k,k} \cap C_{k,k+1}) \cap C(\nabla) \cap \{\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}_0) - \mathcal{L}(\boldsymbol{\upsilon}^*) \geq -\mathrm{K}_0\}, \text{ where} \tag{39}$$
$$C_{k,k(+1)} = \Big\{\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \boldsymbol{\upsilon}^*)\| \leq \mathrm{R}_0(\mathtt{x}),\, \|\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*)\| \leq \mathrm{R}_0(\mathtt{x}),$$
$$\|\mathrm{H}(\widetilde{\boldsymbol{\eta}}_{k(+1)} - \boldsymbol{\eta}^*)\| \leq \mathrm{R}_0(\mathtt{x})\Big\},$$
$$C(\nabla) = \left\{ \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathrm{R}_0(\mathtt{x}))} \|\mathcal{Y}(\nabla^2)(\boldsymbol{\upsilon})\| \leq 9\nu_2\epsilon_2\mathfrak{z}_1(\mathtt{x}, 6p^*)\mathrm{R}_0(\mathtt{x}) \right\}$$
$$\cap\{\|\mathcal{D}^{-1}\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\| \leq \mathfrak{z}(\mathtt{x}, \nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*))\}.$$

where

$$\mathcal{Y}(\nabla^2)(\boldsymbol{\upsilon}) \stackrel{\text{def}}{=} \mathcal{D}^{-1}\left(\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\right) \in \mathbb{R}^{p^{*2}}.$$

We see that on $\Omega(\mathtt{x})$

$$\boldsymbol{\upsilon}_{k,k(+1)} \in \widetilde{\Upsilon}_\circ(\mathrm{R}_0) \stackrel{\text{def}}{=} \{\|\mathcal{D}(\boldsymbol{\upsilon}-\widetilde{\boldsymbol{\upsilon}})\| \le \mathrm{R}_0 + \mathtt{r}_0\} \cap \Upsilon_\circ(\mathrm{R}_0).$$

Lemma 35 *Under the conditions of Theorem 14*

$$\mathbb{P}(\Omega(\mathtt{x})) \ge 1 - 3\mathrm{e}^{-\mathtt{x}} - \beta.$$

Proof The proof is very similar to the one presented in Section 4.3, so we only give a sketch. By assumption

$$\mathbb{P}\left(\|\mathcal{D}^{-1}\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\| \le \mathfrak{z}(\mathtt{x}, \nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*))\right) \ge 1 - \mathrm{e}^{-\mathtt{x}},$$

and due to $(\mathcal{ED}_2)$ with Theorem 47

$$\mathbb{P}\left(\sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathrm{R}_0(\mathtt{x}))} \|\mathcal{Y}(\nabla^2)(\boldsymbol{\upsilon})\| \le 9\nu_2\epsilon_2\mathfrak{z}_1(\mathtt{x}, 6p^*)\mathrm{R}_0(\mathtt{x})\right) \ge 1 - \mathrm{e}^{-\mathtt{x}}.$$

■

Lemma 36 *Assume for some sequence* $(\mathtt{r}_k^{(l)})$ *that*

$$\bigcap_{k\in\mathbb{N}} \left\{\left\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \widetilde{\boldsymbol{\upsilon}})\right\| \le \mathtt{r}_k^{(l)}\right\} \subseteq \Omega(\mathtt{x}).$$

Then we get on $\Omega(\mathtt{x})$

$$\begin{aligned}\left\|\mathcal{D}(\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} - \widetilde{\boldsymbol{\upsilon}})\right\| &\le 2\sqrt{2}(1+\sqrt{\nu})\sum_{r=0}^{k-1}\nu^r\|\boldsymbol{\tau}(\mathtt{r}_{k-r}^{(l)})\| + 2\sqrt{2}\nu^k(R_0 + \mathtt{r}_0),\\ &=: \mathtt{r}_k^{(l+1)}. \end{aligned} \tag{40}$$

where

$$\|\boldsymbol{\tau}(\mathtt{r})\| \le \left[\epsilon R_0 + 9\nu_2\epsilon_2\|\mathcal{D}^{-1}\|\mathfrak{z}_1(\mathtt{x}, 6p^*)R_0 + \|\mathcal{D}^{-1}\|\mathfrak{z}(\mathtt{x}, \nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*))\right]\mathtt{r}.$$

Proof

1. We first show that on $\Omega(\mathtt{x})$

$$\begin{aligned}\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \widetilde{\boldsymbol{\theta}}) &= -\mathrm{D}^{-1}\mathcal{A}(\widetilde{\boldsymbol{\eta}}_k - \widetilde{\boldsymbol{\eta}}) + \boldsymbol{\tau}(\mathtt{r}_k^{(l)}),\\ \mathrm{H}(\widetilde{\boldsymbol{\eta}}_k - \boldsymbol{\eta}^*) &= -\mathrm{H}^{-1}\mathcal{A}^\top(\widetilde{\boldsymbol{\theta}}_{k-1} - \widetilde{\boldsymbol{\theta}}) + \boldsymbol{\tau}(\mathtt{r}_k^{(l)}).\end{aligned}$$

The proof is very similar to that of Lemma 32. Define

$$\alpha(\boldsymbol{\upsilon}, \widetilde{\boldsymbol{\upsilon}}) := \mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\widetilde{\boldsymbol{\upsilon}}) + \|\mathcal{D}(\boldsymbol{\upsilon}-\widetilde{\boldsymbol{\upsilon}})\|^2/2.$$

Note that

$$\begin{aligned}\mathcal{L}(\boldsymbol{\upsilon}) - \mathcal{L}(\widetilde{\boldsymbol{\upsilon}}) &= \nabla\mathcal{L}(\boldsymbol{\upsilon}) - \|\mathcal{D}(\boldsymbol{\upsilon}-\widetilde{\boldsymbol{\upsilon}})\|^2/2 + \alpha(\boldsymbol{\upsilon}, \boldsymbol{\upsilon}^*)\\ &= -\|\mathrm{D}(\boldsymbol{\theta}-\widetilde{\boldsymbol{\theta}})\|^2/2 + (\boldsymbol{\theta}-\boldsymbol{\theta}^*)^\top\mathcal{A}(\boldsymbol{\eta}-\widetilde{\boldsymbol{\eta}})\\ &\quad -\|\mathrm{H}(\boldsymbol{\eta}-\widetilde{\boldsymbol{\eta}})\|^2/2 + \alpha(\boldsymbol{\upsilon}, \widetilde{\boldsymbol{\upsilon}}).\end{aligned}$$

Setting $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\widetilde{\boldsymbol{\theta}}_k, \widetilde{\boldsymbol{\eta}}_k) = 0$ we find

$$\mathrm{D}(\widetilde{\boldsymbol{\theta}}_k - \widetilde{\boldsymbol{\theta}}) = \mathrm{D}^{-1}\mathcal{A}(\widetilde{\boldsymbol{\eta}}_k - \widetilde{\boldsymbol{\eta}}) + \mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\alpha(\widetilde{\boldsymbol{\upsilon}}_{k,k}, \widetilde{\boldsymbol{\upsilon}}).$$

We want to show

$$\sup_{(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\in\widetilde{\Upsilon}_\circ\left(\mathtt{r}_k^{(l)}\right)\cap\Upsilon_\circ(\mathrm{R}_0)} \mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\alpha((\boldsymbol{\theta}, \widetilde{\boldsymbol{\eta}}_k), \widetilde{\boldsymbol{\upsilon}}) \le \|\boldsymbol{\tau}\left(\mathtt{r}_k^{(l)}\right)\|,$$

where

$$\mathrm{D}^{-1}\nabla_{\boldsymbol{\theta}}\alpha(\boldsymbol{\upsilon}, \widetilde{\boldsymbol{\upsilon}}) \stackrel{\text{def}}{=} \mathrm{D}^{-1}\left\{\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\upsilon}) - \mathrm{D}^2\left(\boldsymbol{\theta}-\widetilde{\boldsymbol{\theta}}\right) - \mathcal{A}(\widetilde{\boldsymbol{\eta}}_k - \widetilde{\boldsymbol{\eta}})\right\}.$$

To see this note that by assumption we have $\Omega(\mathtt{x}) \subseteq \{\widetilde{\boldsymbol{\upsilon}} \in \Upsilon_\circ(\mathtt{r}_0)\} \subseteq \{\widetilde{\boldsymbol{\upsilon}} \in \Upsilon_\circ(\mathrm{R}_0)\}$. By condition $(\boldsymbol{\mathcal{L}_0})$, Lemma 27 and Taylor expansion we have

$$\begin{aligned}&\sup_{(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\in\widetilde{\Upsilon}_\circ(\mathtt{r}_k^{(l)})\cap\Upsilon_\circ(\mathrm{R}_0)} \|\mathbb{E}\mathcal{U}(\boldsymbol{\theta}, \widetilde{\boldsymbol{\eta}}_k)\|\\ &\le \sup_{\boldsymbol{\upsilon}\in\widetilde{\Upsilon}_\circ(\mathtt{r}_k^{(l)})\cap\Upsilon_\circ(\mathrm{R}_0)} \|\mathrm{D}^{-1}\Pi_{\boldsymbol{\theta}}\Big(\nabla\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon}) - \nabla\mathbb{E}\mathcal{L}(\widetilde{\boldsymbol{\upsilon}}) - \mathcal{D}\left(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*\right)\Big)\|\\ &\le \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathrm{R}_0)} \|\mathrm{D}^{-1}\Pi_{\boldsymbol{\theta}}\mathcal{D}\|\|\mathcal{D}^{-1}\nabla^2\mathbb{E}\mathcal{L}(\boldsymbol{\upsilon})\mathcal{D}^{-1} - I_{p^*}\|\mathtt{r}_k^{(l)}\\ &\le \epsilon\mathtt{r}_k^{(l)2}.\end{aligned}$$

For the remainder note that with $\zeta = \mathcal{L} - \mathbb{E}\mathcal{L}$ on $\Omega(\mathtt{x})$ using Lemma 27 we can bound

$$\begin{aligned}
&\sup_{(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\in\widetilde{\Upsilon}_\circ(\mathtt{r}_k^{(l)})\cap\Upsilon_\circ(\mathrm{R}_0)} \left\|\mathcal{U}_{\boldsymbol{\theta}}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k) - \mathbb{E}\mathcal{U}_{\boldsymbol{\theta}}(\boldsymbol{\theta},\widetilde{\boldsymbol{\eta}}_k)\right\| \\
&\quad\le \sup_{\boldsymbol{\upsilon}\in\widetilde{\Upsilon}_\circ(\mathtt{r}_k^{(l)})\cap\Upsilon_\circ(\mathrm{R}_0)} \left\|\mathrm{D}^{-1}\Big(\nabla_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \nabla_{\boldsymbol{\theta}}\boldsymbol{\zeta}(\widetilde{\boldsymbol{\upsilon}})\Big)\right\| \\
&\quad\le \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})} \left\|\mathcal{D}^{-1}\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon})\mathcal{D}^{-1}\right\| \mathtt{r}_k^{(l)} \\
&\quad\le \sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathrm{R}_0)} \left\{\frac{1}{9\nu_2\epsilon_2}\|\mathcal{D}^{-1}\left(\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}) - \nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\right)\mathcal{D}^{-1}\|\right\} 6\nu_1\epsilon\mathtt{r}_k^{(l)} \\
&\qquad + \left\{\|\mathcal{D}^{-1}\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*)\mathcal{D}^{-1}\|\right\}\mathtt{r}_k^{(l)} \\
&\quad\le \left[9\nu_2\epsilon_2\|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x},6p^*)\mathrm{R}_0 + \|\mathcal{D}^{-1}\|_{\mathfrak{z}}(\mathtt{x},\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*))\right]\mathtt{r}_k^{(l)}.
\end{aligned}$$

Using the same argument for $\widetilde{\boldsymbol{\eta}}_k$ gives the claim.

Now the claim follows as in the proof of Lemma 32. ∎

Lemma 37 *Assume that* $\kappa(\mathtt{x},R_0) < 1-\nu$ *where*

$$\kappa(\mathtt{x},R_0) \stackrel{\mathrm{def}}{=} \frac{2\sqrt{2}(1+\sqrt{\nu})}{\sqrt{1-\nu}}\Big(\epsilon R_0 + 9\epsilon_2\nu_2\|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x},6p^*)R_0 + \|\mathcal{D}^{-1}\|_{\mathfrak{z}2}\left(\mathtt{x},\nabla^2\mathcal{L}(\boldsymbol{\upsilon}^*)\right)\Big).$$

Then

$$\Omega(\mathtt{x}) \subseteq \bigcap_{k\in\mathbb{N}}\left\{\boldsymbol{\upsilon}_{k,k(+1)} \in \widetilde{\Upsilon}_\circ(\mathtt{r}_k)\right\},$$

where $(\mathtt{r}_k)_{k\in\mathbb{N}}$ *satisfy the bound* (24).

Proof Define for all $k\in\mathbb{N}_0$ the sequence $\mathtt{r}_k^{(0)} = \mathrm{R}_0$. We estimate

$$\|\boldsymbol{\tau}(\mathtt{r}_k^{(l)})\| \le \frac{1}{\sqrt{1-\nu}}\left(\epsilon\mathrm{R}_0 + 6\nu_1\epsilon_2\|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x},6p^*)\mathrm{R}_0 + \|\mathcal{D}^{-1}\|_{\mathfrak{z}}(\mathtt{x},\mathit{I\!B}(\nabla^2))\right)\mathtt{r}_k^{(l)},$$

such that by definition

$$2\sqrt{2}(1+\sqrt{\nu})\sum_{r=0}^{k-1}\nu^r\|\boldsymbol{\tau}(\mathtt{r}_{k-r}^{(l)})\| \le \kappa(\mathtt{x},\mathrm{R}_0)\sum_{r=0}^{k-1}\nu^r\mathtt{r}_{k-r}^{(l)}.$$

Plugging in the recursive formula for $\mathtt{r}_k^{(l)}$ from (40) and denoting $\widetilde{\mathrm{R}}_0 \stackrel{\mathrm{def}}{=} \mathrm{R}_0 + \mathtt{r}_0$ we find

$$\begin{aligned}
\mathtt{r}_k^{(l)} &\le \kappa(\mathtt{x},\mathrm{R}_0)\sum_{r=0}^{k-1}\nu^r\mathtt{r}_{k-r}^{(l-1)} + 2\sqrt{2}\nu^k\widetilde{\mathrm{R}}_0 \\
&\le \kappa(\mathtt{x},\mathrm{R}_0)\sum_{r=0}^{k-1}\nu^r\left(\kappa(\mathtt{x},\mathrm{R}_0)\sum_{s=0}^{k-r-1}\nu^s\mathtt{r}_{k-r-s}^{(l-2)} + 2\nu^{k-r}\widetilde{\mathrm{R}}_0\right) + 2\sqrt{2}\widetilde{\mathrm{R}}_0\nu^k \\
&\le \kappa(\mathtt{x},\mathrm{R}_0)^2\sum_{r=0}^{k-1}\nu^r\sum_{s=0}^{k-r-1}\nu^s\mathtt{r}_{k-r-s}^{(l-2)} + 2\sqrt{2}\nu^k\widetilde{\mathrm{R}}_0\left(\kappa(\mathtt{x},\mathrm{R}_0)k+1\right) \\
&\le \kappa(\mathtt{x},\mathrm{R}_0)^2\sum_{r=0}^{k-1}\nu^r\sum_{s=0}^{k-r-1}\nu^s\left(\kappa(\mathtt{x},\mathrm{R}_0)\sum_{t=0}^{k-r-s-1}\nu^t\mathtt{r}_{k-r-s-t}^{(l-3)} + 2\nu^{k-r-s}\widetilde{\mathrm{R}}_0\right) \\
&\quad + 2\sqrt{2}\nu^k\widetilde{\mathrm{R}}_0\left(\kappa(\mathtt{x},\mathrm{R}_0)k+1\right) \\
&\le \kappa(\mathtt{x},\mathrm{R}_0)^3\sum_{r=0}^{k-1}\nu^r\sum_{s=0}^{k-r-1}\nu^s\mathtt{r}_{k-r-s}^{(l-3)} + 2\sqrt{2}\nu^k\widetilde{\mathrm{R}}_0\left(\kappa(\mathtt{x},\mathrm{R}_0)^2k^2 + \kappa(\mathtt{x},\mathrm{R}_0)k+1\right).
\end{aligned}$$

By induction this gives for $l\in\mathbb{N}$

$$\begin{aligned}
\mathtt{r}_k^{(l)} &\le \kappa(\mathtt{x},\mathrm{R}_0)^l\sum_{r_1=0}^{k-1}\nu^{r_1}\sum_{r_2=0}^{k-r_1-1}\nu^{r_2}\ldots\sum_{r_l=0}^{k-\sum_{s=1}^{l-1}r_s-1}\nu^{r_l}\widetilde{\mathrm{R}}_0 \\
&\quad + 2\sqrt{2}\nu^k\widetilde{\mathrm{R}}_0\sum_{s=0}^{l-1}\kappa(\mathtt{x},\mathrm{R}_0)^s k^s \\
&\le \left(\left(\frac{\kappa(\mathtt{x},\mathrm{R}_0)}{1-\nu}\right)^l + 2\sqrt{2}\nu^k\sum_{s=0}^{l-1}\left(\kappa(\mathtt{x},\mathrm{R}_0)k\right)^s\right)\widetilde{\mathrm{R}}_0 \\
&\le \begin{cases}\left(\left(\frac{\kappa(\mathtt{x},\mathrm{R}_0)}{1-\nu}\right)^l + 2\sqrt{2}\nu^k\frac{1}{1-\kappa(\mathtt{x},\mathrm{R}_0)k}\right)\widetilde{\mathrm{R}}_0, & \kappa(\mathtt{x},\mathrm{R}_0)k\le 1,\\ \kappa(\mathtt{x},\mathrm{R}_0)^l\left(\left(\frac{1}{1-\nu}\right)^l + 2\sqrt{2}\nu^k\frac{k^l}{\kappa(\mathtt{x},\mathrm{R}_0)k-1}\right)\widetilde{\mathrm{R}}_0, & \text{otherwise.}\end{cases}
\end{aligned}$$

By Lemma 36

$$\Omega(\mathtt{x}) \subset \bigcap_{k\in\mathbb{N}_0}\bigcap_{l\in\mathbb{N}}\left\{\widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} \in \widetilde{\Upsilon}_\circ\big(\mathtt{r}_k^{(l)}\big)\right\}.$$

Set if $\kappa(\mathtt{x},\mathrm{R}_0)/(1-\nu) < 1$

$$l(k) \stackrel{\mathrm{def}}{=} \begin{cases}\infty, & \kappa(\mathtt{x},\mathrm{R}_0)k\le 1,\\ \frac{k\log(\nu)+\log(2\sqrt{2})-\log(\kappa(\mathtt{x},\mathrm{R}_0)k-1)}{-\log(1-\nu)-\log(k)}, & \text{otherwise.}\end{cases}$$

Then with $\mathtt{r}_k^* \stackrel{\text{def}}{=} \mathtt{r}_k^{(\lfloor l(k) \rfloor)}$ we get

$$\Omega(\mathtt{x}) \subset \bigcap_{k\in\mathbb{N}_0} \left\{ \widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} \in \widetilde{\Upsilon}_\circ\big(\mathtt{r}_k^*\big) \right\},$$

$$\mathtt{r}_k^* \leq \begin{cases} \frac{\nu^k 2\sqrt{2}}{1-\kappa(\mathtt{x},\mathrm{R}_0)k} \widetilde{\mathrm{R}}_0, & \kappa(\mathtt{x},\mathrm{R}_0)k \leq 1, \\ 2\left(\frac{\kappa(\mathtt{x},\mathrm{R}_0)}{1-\nu}\right)^{\frac{k}{\log(k)}L(k)-1} \widetilde{\mathrm{R}}_0, & \text{otherwise.} \end{cases}$$

The sequence $L(k) > 0$ is defined as

$$L(k) \stackrel{\text{def}}{=} \left\lfloor \frac{\log(1/\nu) - \frac{1}{k}\left(\log(2\sqrt{2}) - \log(\kappa(\mathtt{x},\mathrm{R}_0)k-1)\right)}{1 + \frac{1}{\log(k)}\log(1-\nu)} \right\rfloor \in \mathbb{N},$$

where $\lfloor x \rfloor \in \mathbb{N}_0$ denotes the largest natural number smaller than $x > 0$. To ensure that $L(k) > 0$ we assume that $k\log(1/\nu) - \log(2\sqrt{2}) > k$. Further as $\kappa(\mathtt{x},\mathrm{R}_0) < (1-\nu)$ and $L(k)$ is only relevant once $\kappa(\mathtt{x},\mathrm{R}_0)k > 1$ it follows that

$$0 < 1 + \frac{1}{\log(k)}\log(1-\nu) < 1.$$

Then

$$L(k) \geq \log(1/\nu) - \frac{1}{k}\Big(\log(2\sqrt{2}) - \log(\kappa(\mathtt{x},\mathrm{R}_0)k-1)\Big) > 1.$$

Consequently

$$\begin{aligned} \left(\frac{\kappa(\mathtt{x},\mathrm{R}_0)}{1-\nu}\right)^{\frac{k}{\log(k)}L(k)} &\leq \nu^{\frac{k}{\log(k)}\log\left(\frac{1-\nu}{\kappa(\mathtt{x},\mathrm{R}_0)}\right)} \left(\frac{\kappa(\mathtt{x},\mathrm{R}_0)}{1-\nu}\right)^{-\frac{1}{\log(k)}\left(\log(2\sqrt{2})-\log(\kappa(\mathtt{x},\mathrm{R}_0)k-1)\right)} \\ &\stackrel{\text{def}}{=} \nu^{\frac{k}{\log(k)}\log\left(\frac{1-\nu}{\kappa(\mathtt{x},\mathrm{R}_0)}\right)} c_k, \end{aligned}$$

where $c_k \to \frac{\kappa(\mathtt{x},\mathrm{R}_0)}{1-\nu}$. Finally note that $\widetilde{\mathrm{R}}_0 \leq 2\mathrm{R}_0$ and the proof is complete. ■

Remark 38 *As pointed out in Remark 18 the above result can be improved. Redefine* $\Omega(\mathtt{x})$ *as the intersection of the two sets in* (29) *and* (39). *Then* $\mathbb{P}(\Omega(\mathtt{x})) \geq 1 - 10\mathrm{e}^{-\mathtt{x}}$. *Also redefine*

$$\kappa(\mathtt{x},\mathtt{r}) \stackrel{\text{def}}{=} \frac{2\sqrt{2}(1+\sqrt{\nu})}{\sqrt{1-\nu}} \Big(\epsilon\mathtt{r} + 3\epsilon_2 \|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x}, 6p^*)\mathtt{r} + \|\mathcal{D}^{-1}\|_{\mathfrak{z}2}\left(\mathtt{x}, \nabla^2\mathcal{L}(\boldsymbol{\upsilon}^*)\right) \Big).$$

By the arguments of the proof of Theorem 7 we find with $\mathtt{r}_k^*$ *defined in* (33)

$$\Omega(\mathtt{x}) \subset \bigcap_{k\in\mathbb{N}} \{\boldsymbol{\upsilon}_{k,k(+1)} \in \Upsilon_\circ(\mathtt{r}_k^*)\}.$$

Using this in Lemma 36 instead of $\cap_{k\in\mathbb{N}}\{\boldsymbol{\upsilon}_{k,k(+1)} \in \Upsilon_\circ(R_0)\}$ *we can bound*

$$\begin{aligned} \|\boldsymbol{\tau}(\mathtt{r}_k^{(l)})\| &\leq \frac{1}{\sqrt{1-\nu}} \Big(\epsilon\mathtt{r}_k^* + 6\nu_1\epsilon_2 \|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x}, 6p^*)\mathtt{r}_k^* \\ &\quad + \|\mathcal{D}^{-1}\|_{\mathfrak{z}}(\mathtt{x}, I\!B(\nabla^2)) \Big) \mathtt{r}_k^{(l)}. \end{aligned}$$

Consequently, representing $\mathtt{r}_k^* = \mathtt{C}\left(\mathfrak{z}(\mathtt{x}) + \nu^k R_0\right)$ *we find*

$$\begin{aligned} \mathtt{r}_k^{(l)} &\leq \kappa(\mathtt{x}, \mathtt{C}_\mathfrak{z}(\mathtt{x})) \sum_{r=0}^{k-1} \nu^r \mathtt{r}_{k-r}^{(l-1)} + 2\sqrt{2}\nu^k(R_0 + \mathtt{r}_0) \\ &\quad + \mathtt{C}\epsilon(1 + \|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x}, 6p^*)) \sum_{r=0}^{k-1} \nu^k R_0 \mathtt{r}_{k-r}^{(l-1)} \\ &\leq \kappa(\mathtt{x}, \mathtt{C}_\mathfrak{z}(\mathtt{x})) \sum_{r=0}^{k-1} \nu^r \mathtt{r}_{k-r}^{(l-1)} + \mathtt{C}_1\epsilon R_0 k\nu^k(R_0 + \mathtt{r}_0), \end{aligned}$$

where $\mathtt{C}_1 \leq 2\sqrt{2} + \mathtt{C}(1 + \|\mathcal{D}^{-1}\|_{\mathfrak{z}1}(\mathtt{x}, 6p^*))$. *With the same arguments as in the proof of Lemma 37 we infer*

$$\begin{aligned} &\mathtt{r}_k^{(l)}/(R_0 + \mathtt{r}_0) \\ &\leq \begin{cases} \left(\left(\frac{\kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))}{1-\nu}\right)^l + k\nu^k \frac{\mathtt{C}_1\epsilon R_0}{1-\kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))k} \right), & \kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))k \leq 1, \\ \kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))^l \left(\left(\frac{1}{1-\nu}\right)^l + \nu^k \frac{k^{l+1}\mathtt{C}_1\epsilon R_0}{\kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))k-1} \right), & \textit{otherwise.} \end{cases} \end{aligned}$$

Set

$$l(k) \stackrel{\text{def}}{=} \begin{cases} \infty, & \kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))k \leq 1, \\ \frac{k\log(\nu) + \log(\mathtt{C}_1\epsilon R_0) - \log(\kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))k-1) - \log(k)}{-\log(1-\nu) - \log(k)}, & \textit{otherwise.} \end{cases}$$

Then with $\mathtt{r}_k^* \stackrel{\text{def}}{=} \mathtt{r}_k^{(\lfloor l(k) \rfloor)}$ *we get with a slight adaptation of* $L(k)$

$$\Omega(\mathtt{x}) \subset \bigcap_{k\in\mathbb{N}_0} \left\{ \widetilde{\boldsymbol{\upsilon}}_{k,k(+1)} \in \widetilde{\Upsilon}_\circ\big(\mathtt{r}_k^*\big) \right\},$$

$$\mathtt{r}_k^* \leq \begin{cases} \frac{\nu^k 2\sqrt{2}}{1-\kappa(\mathtt{x}\mathtt{C}_\mathfrak{z}(\mathtt{x}))k}(R_0 + \mathtt{r}_0), & \kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))k \leq 1, \\ 2\left(\frac{\kappa(\mathtt{x},\mathtt{C}_\mathfrak{z}(\mathtt{x}))}{1-\nu}\right)^{\frac{k}{\log(k)}L(k)-1}(R_0 + \mathtt{r}_0), & \textit{otherwise.} \end{cases}$$

This gives the claim.

Acknowledgments

The first author was supported by Research Units 1735 "Structural Inference in Statistics: Adaptation and Efficiency". The research was partly supported by the Russian Science Foundation grant (project 14-50-00150). Financial support by the German Research Foundation (DFG) through the Collaborative Research Center 649 "Economic Risk" is gratefully acknowledged.

Appendix A. Deviation Bounds for Quadratic Forms

This section is the same as Section A of Andresen and Spokoiny (2014). The following general result from Spokoiny (2012) helps to control the deviation for quadratic forms of type $\|I\!B\boldsymbol{\xi}\|^2$ for a given positive matrix $I\!B$ and a random vector $\boldsymbol{\xi}$. It will be used several times in our proofs. Suppose that

$$\log \mathbb{E} \exp\bigl(\boldsymbol{\gamma}^\top \boldsymbol{\xi}\bigr) \le \|\boldsymbol{\gamma}\|^2/2, \qquad \boldsymbol{\gamma} \in \mathbb{R}^p, \|\boldsymbol{\gamma}\| \le \mathtt{g}.$$

For a symmetric matrix $I\!B$, define

$$\mathtt{p} = \operatorname{tr}(I\!B^2), \qquad \mathtt{v}^2 = 2\operatorname{tr}(I\!B^4), \qquad \lambda^* \stackrel{\text{def}}{=} \|I\!B^2\|_\infty \stackrel{\text{def}}{=} \lambda_{\max}(I\!B^2).$$

For ease of presentation, suppose that $\mathtt{g}^2 \ge 2\mathtt{p}_{I\!B}$. The other case only changes the constants in the inequalities. Note that $\|\boldsymbol{\xi}\|^2 = \boldsymbol{\eta}^\top I\!B\,\boldsymbol{\eta}$. Define $\mu_c = 2/3$ and

$$\begin{aligned} \mathtt{g}_c &\stackrel{\text{def}}{=} \sqrt{\mathtt{g}^2 - \mu_c \mathtt{p}_{I\!B}}, \\ 2(\mathtt{x}_c + 2) &\stackrel{\text{def}}{=} (\mathtt{g}^2/\mu_c - \mathtt{p}_{I\!B})/\lambda^* + \log\det\bigl(I_p - \mu_c I\!B/\lambda^*\bigr). \end{aligned}$$

Proposition 39 *Let (ED_0) hold with $\nu_0 = 1$ and $\mathtt{g}^2 \ge 2\mathtt{p}_{I\!B}$. Then for each $\mathtt{x} > 0$*

$$\mathbb{P}\bigl(\|I\!B\boldsymbol{\xi}\| \ge \mathfrak{z}(\mathtt{x}, I\!B)\bigr) \le 2\mathrm{e}^{-\mathtt{x}},$$

where $\mathfrak{z}(\mathtt{x}, I\!B)$ is defined by

$$\mathfrak{z}^2(I\!B, \mathtt{x}) \stackrel{\text{def}}{=} \begin{cases} \mathtt{p}_{I\!B} + 2\mathtt{v}_{I\!B}(\mathtt{x}+1)^{1/2}, & \mathtt{x}+1 \le \mathtt{v}_{I\!B}/(18\lambda^*), \\ \mathtt{p}_{I\!B} + 6\lambda^*(\mathtt{x}+1), & \mathtt{v}_{I\!B}/(18\lambda^*) < \mathtt{x}+1 \le \mathtt{x}_c + 2, \\ \bigl|\mathtt{y}_c + 2\lambda^*(\mathtt{x} - \mathtt{x}_c + 1)/\mathtt{g}_c\bigr|^2, & \mathtt{x} > \mathtt{x}_c + 1, \end{cases}$$

with $\mathtt{y}_c^2 \le \mathtt{p}_{I\!B} + 6\lambda^*(\mathtt{x}_c + 2)$.

Appendix B. A Uniform Bound for the Norm of a Random Process

We want to derive for a random process $\breve{\mathtt{y}}(\boldsymbol{\upsilon}) \in \mathbb{R}^p$ a bound of the kind

$$\mathbb{P}\left(\sup_{\mathtt{r} \le \mathtt{r}^*} \sup_{\boldsymbol{\upsilon} \in \Upsilon_\circ(\mathtt{r})} \left\{\frac{1}{\epsilon}\|\breve{\mathtt{y}}(\boldsymbol{\upsilon})\| - 2\mathtt{r}^2\right\} \ge \mathtt{C}\mathfrak{z}_Q(\mathtt{x}, p^*)\right) \le \mathrm{e}^{-\mathtt{x}}.$$

This is a slightly stronger result than the one derived in Section D of (Andresen and Spokoiny, 2014) but the ideas employed here are very similar.

We want to apply Corollary 2.5 of the supplement of Spokoiny (2012) which we cite here as a Theorem. Note that we slightly generalized the formulation of the theorem, to make it applicable in out setting. The proof remains the same.

Theorem 40 *Let $(U(\mathtt{r}))_{0 \le \mathtt{r} \le \mathtt{r}^*} \subset \mathbb{R}^p$ be a sequence of balls around $\boldsymbol{\upsilon}^*$ induced by the metric $d(\cdot,\cdot)$. Let a random real valued process $\mathcal{U}(\mathtt{r}, \boldsymbol{\upsilon})$ fulfill for any $0 \le \mathtt{r} \le \mathtt{r}^*$ that $\mathcal{U}(\mathtt{r}, \boldsymbol{\upsilon}^*) = 0$ and*

($\mathcal{E}d$) *For any* $\upsilon, \upsilon^\circ \in U(\mathtt{r})$

$$\log \mathbb{E} \exp\Big\{ \lambda \frac{\mathcal{U}(\mathtt{r},\upsilon) - \mathcal{U}(\mathtt{r},\upsilon^\circ)}{d(\upsilon,\upsilon^\circ)} \Big\} \le \frac{\nu_0^2 \lambda^2}{2}, \qquad |\lambda| \le \mathtt{g}. \tag{41}$$

Finally assume that $\sup_{\upsilon \in U(\mathtt{r})} (\mathcal{U}(\mathtt{r},\upsilon))$ *increases in* $\mathtt{r}$. *Then with probability greater* $1 - \mathrm{e}^{-\mathtt{x}}$

$$\sup_{\upsilon \in U(\mathtt{r})} \Big\{ \frac{1}{3\nu_1} \mathcal{U}(\mathtt{r},\upsilon) - d(\upsilon,\upsilon^*)^2 \Big\} \le \mathfrak{z}_Q(\mathtt{x}, p^*)^2,$$

where $\mathfrak{z}_Q(\mathtt{x}, p^*) \stackrel{\mathrm{def}}{=} \mathbb{Q}(U(\mathtt{r}^*))$ *denotes the entropy of the set* $U(\mathtt{r}^*) \subset \mathbb{R}^p$ *and where with* $\mathtt{g}_0 = \nu_0 \mathtt{g}$ *and for some* $\mathbb{Q} > 0$

$$\mathfrak{z}_Q(\mathtt{x}, \mathbb{Q})^2 \stackrel{\mathrm{def}}{=} \begin{cases} (1 + \sqrt{\mathtt{x} + \mathbb{Q}})^2 & \text{if } 1 + \sqrt{\mathtt{x} + \mathbb{Q}} \le \mathtt{g}_0, \\ 1 + \{2\mathtt{g}_0^{-1}(\mathtt{x} + \mathbb{Q}) + \mathtt{g}_0\}^2 & \text{otherwise.} \end{cases} \tag{42}$$

To use this result let $\breve{\mathtt{y}}(\upsilon)$ be a smooth centered random vector process with values in $\mathbb{R}^p$ and let $\mathcal{D} : \mathbb{R}^{p^*} \to \mathbb{R}^{p^*}$ be some linear operator. We aim at bounding the maximum of the norm $\|\breve{\mathtt{y}}(\upsilon)\|$ over a vicinity $\Upsilon_\circ(\mathtt{r}) \stackrel{\mathrm{def}}{=} \{\|\mathcal{D}(\upsilon - \upsilon^*)\| \le \mathtt{r}\}$ of υ^*. Suppose that $\breve{\mathtt{y}}(\upsilon)$ satisfies for each $0 < \mathtt{r} < \mathtt{r}^*$ and for all pairs $\upsilon, \upsilon^\circ \in \Upsilon_\circ(\mathtt{r}) = \big\{\upsilon \in \Upsilon \colon \|\mathcal{D}(\upsilon - \upsilon^*)\| \le \mathtt{r}\big\} \subset \mathbb{R}^{p^*}$

$$\sup_{\|\boldsymbol{u}\| \le 1} \log \mathbb{E} \exp\Big\{ \lambda \frac{\boldsymbol{u}^\top \big(\breve{\mathtt{y}}(\upsilon) - \breve{\mathtt{y}}(\upsilon^\circ)\big)}{\epsilon \|\mathcal{D}(\upsilon - \upsilon^\circ)\|} \Big\} \le \frac{\nu_0^2 \lambda^2}{2}. \tag{43}$$

Remark 41 *In the setting of Theorem 7 we have*

$$\breve{\mathtt{y}}(\upsilon) = \breve{D}^{-1} \Big(\breve{\nabla} \boldsymbol{\zeta}(\upsilon) - \breve{\nabla} \boldsymbol{\zeta}(\upsilon^*) \Big),$$

and condition (43) *becomes* $(\mathcal{E}\mathcal{D}_1)$ *from 2.1.*

Theorem 42 *Let a random* p*-vector process* $\breve{\mathtt{y}}(\upsilon)$ *fulfill* $\breve{\mathtt{y}}(\upsilon^*) = 0$ *and let condition* (43) *be satisfied. Then for each* $0 \le \mathtt{r} \le \mathtt{r}^*$, *on a set of probability greater* $1 - \mathrm{e}^{-\mathtt{x}}$

$$\sup_{\mathtt{r} \le \mathtt{r}^*} \sup_{\upsilon \in \Upsilon_\circ(\mathtt{r})} \Big\{ \frac{1}{6\epsilon\nu_1} \|\breve{\mathtt{y}}(\upsilon)\| - 2\mathtt{r}^2 \Big\} \le \mathfrak{z}_Q(\mathtt{x}, 2p^* + 2p)^2,$$

with $\mathtt{g}_0 = \nu_0 \mathtt{g}$.

Remark 43 *Note that the entropy of the original set* $\Upsilon_\circ(\mathtt{r}) \subset \mathbb{R}^{p^*}$ *is equal to* $2p^*$. *So in order to control the norm* $\|\breve{\mathtt{y}}(\upsilon)\|$ *one only pays with the additional sumand* $2p$.

Proof In what follows, we use the representation

$$\|\breve{\mathtt{y}}(\upsilon)\| = \epsilon \sup_{\|\boldsymbol{u}\| \le \|\mathcal{D}(\upsilon - \upsilon^*)\|} \frac{1}{\epsilon \|\mathcal{D}(\upsilon - \upsilon^*)\|} \boldsymbol{u}^\top \breve{\mathtt{y}}(\upsilon).$$

This implies

$$\sup_{\upsilon \in \Upsilon_\circ(\mathtt{r})} \|\breve{\mathtt{y}}(\upsilon)\| = \epsilon \sup_{\upsilon \in \Upsilon_\circ(\mathtt{r})} \sup_{\|\boldsymbol{u}\| \le \|\mathcal{D}(\upsilon - \upsilon^*)\|} \frac{1}{\epsilon \|\mathcal{D}(\upsilon - \upsilon^*)\|} \boldsymbol{u}^\top \breve{\mathtt{y}}(\upsilon).$$

Due to Lemma 44 the process $\mathcal{U}(\mathtt{r},\upsilon,\boldsymbol{u}) \stackrel{\mathrm{def}}{=} \frac{1}{\epsilon \|\mathcal{D}(\upsilon - \upsilon^*)\|} \boldsymbol{u}^\top \breve{\mathtt{y}}(\upsilon)$ satisfies condition $(\mathcal{E}d)$ (see (41)) as process on $U(\mathtt{r}^*)$ where

$$U(\mathtt{r}) \stackrel{\mathrm{def}}{=} \Upsilon_\circ(\mathtt{r}) \times B_{\mathtt{r}}(0). \tag{44}$$

Further $\sup_{(\upsilon,\boldsymbol{u}) \in U(\mathtt{r})} \mathcal{U}(\mathtt{r},\upsilon,\boldsymbol{u})$ is increasing in $\mathtt{r}$. This allows to apply Theorem 42 to obtain the desired result. Set $d((\upsilon,\boldsymbol{u}),(\upsilon^\circ,\boldsymbol{u}^\circ))^2 = \|\mathcal{D}(\upsilon - \upsilon^*)\|^2 + \|\boldsymbol{u} - \boldsymbol{u}^\circ\|^2$. We get on a set of probability greater $1 - \mathrm{e}^{-\mathtt{x}}$

$$\sup_{(\upsilon,\boldsymbol{u}) \in U(\mathtt{r}^*)} \Big\{ \frac{1}{6\epsilon\nu_1 \|\mathcal{D}(\upsilon - \upsilon^*)\|} \boldsymbol{u}^\top \breve{\mathtt{y}}(\upsilon) - \|\mathcal{D}(\upsilon - \upsilon^*)\|^2 - \|\boldsymbol{u}\|^2 \Big\} \le \mathfrak{z}_Q\Big(\mathtt{x}, \mathbb{Q}\big(U(\mathtt{r}^*)\big)\Big).$$

The constant $\mathbb{Q}\big(U(\mathtt{r}^*)\big) > 0$ quantifies the complexity of the set $U(\mathtt{r}^*) \subset \mathbb{R}^{p^*} \times \mathbb{R}^p$. We point out that for compact $M \subset \mathbb{R}^{p^*}$ we have $\mathbb{Q}(M) = 2p^*$ (see Supplement of Spokoiny (2012), Lemma 2.10). This gives $\mathbb{Q}\big(U\big) = 2p^* + 2p$. Finally observe that

$$\begin{aligned} &\sup_{\mathtt{r} \le \mathtt{r}^*} \sup_{\upsilon \in \Upsilon_\circ(\mathtt{r})} \Big\{ \frac{1}{6\epsilon\nu_1} \|\breve{\mathtt{y}}(\upsilon)\| - 2\mathtt{r}^2 \Big\} \\ &\quad \le \sup_{\mathtt{r} \le \mathtt{r}^*} \sup_{(\upsilon,\boldsymbol{u}) \in U(\mathtt{r})} \Big\{ \frac{1}{6\epsilon\nu_1 \|\mathcal{D}(\upsilon - \upsilon^*)\|} \boldsymbol{u}^\top \breve{\mathtt{y}}(\upsilon) - \|\mathcal{D}(\upsilon - \upsilon^*)\|^2 - \|\boldsymbol{u}\|^2 \Big\} \\ &\quad = \sup_{(\upsilon,\boldsymbol{u}) \in U(\mathtt{r}^*)} \Big\{ \frac{1}{6\epsilon\nu_1 \|\mathcal{D}(\upsilon - \upsilon^*)\|} \boldsymbol{u}^\top \breve{\mathtt{y}}(\upsilon) - \|\mathcal{D}(\upsilon - \upsilon^*)\|^2 - \|\boldsymbol{u}\|^2 \Big\}. \end{aligned}$$

■

Lemma 44 *Suppose that* $\breve{\mathtt{y}}(\upsilon)$ *satisfies for each* $\|\boldsymbol{u}\| \le 1$ *and* $|\lambda| \le \mathtt{g}$ *the inequality* (43). *Then the process* $\mathcal{U}(\upsilon,\boldsymbol{u}) = \frac{1}{2\epsilon \|\mathcal{D}(\upsilon - \upsilon^*)\|} \breve{\mathtt{y}}(\upsilon)^\top \boldsymbol{u}_1$ *satisfies* **($\mathcal{E}d$)** *from* (41) *with* $|\lambda| \le \mathtt{g}/2$, $d((\upsilon,\boldsymbol{u}),(\upsilon^\circ,\boldsymbol{u}^\circ))^2 = \|\mathcal{D}(\upsilon - \upsilon^*)\|^2 + \|\boldsymbol{u} - \boldsymbol{u}^\circ\|^2$, $\nu = 2\nu_0$ *and* $U \subset \mathbb{R}^{p^* + p}$ *defined in* (44), *i.e. for any* $(\upsilon,\boldsymbol{u}_1),(\upsilon^\circ,\boldsymbol{u}_2) \in U$

$$\log \mathbb{E} \exp\Big\{ \lambda \frac{\mathcal{U}(\upsilon,\boldsymbol{u}_1) - \mathcal{U}(\upsilon^\circ,\boldsymbol{u}_2)}{d((\upsilon,\boldsymbol{u}_1),(\upsilon^\circ,\boldsymbol{u}_2))} \Big\} \le \frac{\nu_0^2 \lambda^2}{2}, \qquad |\lambda| \le \mathtt{g}/2.$$

Proof Let $(\boldsymbol{\upsilon},\boldsymbol{u}_1),(\boldsymbol{\upsilon}^\circ,\boldsymbol{u}_2)\in U$ and w.l.o.g. $\boldsymbol{u}_1\le\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\|\le\|\mathcal{D}(\boldsymbol{\upsilon}^\circ-\boldsymbol{\upsilon}^*)\|$. By the Hölder inequality and (43), we find

$$\begin{aligned}
&\log\mathbb{E}\exp\biggl\{\lambda\frac{\mathcal{U}(\boldsymbol{\upsilon},\boldsymbol{u}_1)-\mathcal{U}(\boldsymbol{\upsilon},\boldsymbol{u}_2)}{d((\boldsymbol{\upsilon},\boldsymbol{u}_1),(\boldsymbol{\upsilon}^\circ,\boldsymbol{u}_2))}\biggr\}\\
&\quad=\log\mathbb{E}\exp\biggl\{\lambda\frac{\mathcal{U}(\boldsymbol{\upsilon},\boldsymbol{u}_1)-\mathcal{U}(\boldsymbol{\upsilon}^\circ,\boldsymbol{u}_1)+\mathcal{U}(\boldsymbol{\upsilon}^\circ,\boldsymbol{u}_1)-\mathcal{U}(\boldsymbol{\upsilon}^\circ,\boldsymbol{u}_2)}{d((\boldsymbol{\upsilon},\boldsymbol{u}_1),(\boldsymbol{\upsilon}^\circ,\boldsymbol{u}_2))}\biggr\}\\
&\quad\le\frac{1}{2}\log\mathbb{E}\exp\biggl\{2\lambda\frac{\boldsymbol{u}_1^\top\bigl(\frac{1}{\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\|}\breve{\mathcal{Y}}(\boldsymbol{\upsilon})-\frac{1}{\|\mathcal{D}(\boldsymbol{\upsilon}^\circ-\boldsymbol{\upsilon}^*)\|}\breve{\mathcal{Y}}(\boldsymbol{\upsilon}^\circ)\bigr)}{\epsilon\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^\circ)\|}\biggr\}\\
&\qquad+\frac{1}{2}\log\mathbb{E}\exp\biggl\{2\lambda\frac{(\boldsymbol{u}_1^\top-\boldsymbol{u}_2^\top)\breve{\mathcal{Y}}(\boldsymbol{\upsilon}^\circ)}{\epsilon\|\boldsymbol{u}_1-\boldsymbol{u}_2\|\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\|}\biggr\}\\
&\quad\le\sup_{\|\boldsymbol{u}\|\le1}\frac{1}{2}\log\mathbb{E}\exp\biggl\{2\lambda\frac{\boldsymbol{u}^\top\bigl(\breve{\mathcal{Y}}(\boldsymbol{\upsilon})-\breve{\mathcal{Y}}(\boldsymbol{\upsilon}^\circ)\bigr)}{\epsilon\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^\circ)\|}\biggr\}\\
&\qquad+\sup_{\|\boldsymbol{u}\|\le1}\frac{1}{2}\log\mathbb{E}\exp\biggl\{2\lambda\frac{\boldsymbol{u}^\top\bigl(\breve{\mathcal{Y}}(\boldsymbol{\upsilon}^\circ)-\breve{\mathcal{Y}}(\boldsymbol{\upsilon}^*)\bigr)}{\epsilon\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*)\|}\biggr\}\\
&\quad\le\frac{4\nu_0^2\lambda^2}{2},\qquad\lambda\le\mathtt{g}/2.
\end{aligned}$$

■

Appendix C. A Bound for the Spectral Norm of a Random Matrix Process

We want to derive for a random process $\breve{\mathcal{Y}}(\boldsymbol{\upsilon})\in\mathbb{R}^{p^*\times p^*}$ a bound of the kind

$$\mathbb{P}\biggl(\sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})}\Bigl\{\|\breve{\mathcal{Y}}(\boldsymbol{\upsilon})\|\Bigr\}\ge\mathtt{C}\epsilon_2\mathfrak{z}_1(\mathtt{x},p^*)\mathtt{r}\biggr)\le\mathrm{e}^{-\mathtt{x}}.$$

We derive such a bound in a very similar manner to Theorem E.1 of Andresen and Spokoiny (2014).

We want to apply Corollary 2.2 of the supplement of Spokoiny (2012). Again we slightly generalized the formulation but the proof remains the same.

Corollary 45 *Let $(U(\mathtt{r}))_{0\le\mathtt{r}\le\mathtt{r}^*}\subset\mathbb{R}^p$ be a sequence of balls around $\boldsymbol{\upsilon}^*$ induced by the metric $d(\cdot,\cdot)$. Let a random real valued process $\mathcal{U}(\boldsymbol{\upsilon})$ fulfill that $\mathcal{U}(\boldsymbol{\upsilon}^*)=0$ and*

($\mathcal{E}d$) *For any $\boldsymbol{\upsilon},\boldsymbol{\upsilon}^\circ\in U(\mathtt{r})$*

$$\log\mathbb{E}\exp\biggl\{\lambda\frac{\mathcal{U}(\boldsymbol{\upsilon})-\mathcal{U}(\boldsymbol{\upsilon}^\circ)}{d(\boldsymbol{\upsilon},\boldsymbol{\upsilon}^\circ)}\biggr\}\le\frac{\nu_0^2\lambda^2}{2},\qquad|\lambda|\le\mathtt{g}.\tag{45}$$

Then for each $0\le\mathtt{r}\le\mathtt{r}^*$, on a set of probability greater $1-\mathrm{e}^{-\mathtt{x}}$

$$\sup_{\boldsymbol{\upsilon}\in U(\mathtt{r})}\mathcal{U}(\boldsymbol{\upsilon})\le3\nu_1\mathfrak{z}_1(\mathtt{x},p^*)^2d(\boldsymbol{\upsilon},\boldsymbol{\upsilon}^*),$$

where $\mathfrak{z}_1(\mathtt{x},p^*)\stackrel{\mathrm{def}}{=}\mathbb{Q}(U(\mathtt{r}^*))$ denotes the entropy of the set $U(\mathtt{r}^*)\subset\mathbb{R}^p$ and where with $\mathtt{g}_0=\nu_0\mathtt{g}$ and for some $\mathbb{Q}>0$

$$\mathfrak{z}_1(\mathtt{x},\mathbb{Q})\stackrel{\mathrm{def}}{=}\begin{cases}\sqrt{2(\mathtt{x}+\mathbb{Q})} & \text{if }\sqrt{2(\mathtt{x}+\mathbb{Q})}\le\mathtt{g}_0,\\ \mathtt{g}_0^{-1}(\mathtt{x}+\mathbb{Q})+\mathtt{g}_0/2 & \text{otherwise.}\end{cases}\tag{46}$$

To use this result let $\mathcal{Y}(\boldsymbol{\upsilon})$ be a smooth centered random process with values in $\mathbb{R}^{p^*\times p^*}$ and let $\mathcal{D}:\mathbb{R}^{p^*}\to\mathbb{R}^{p^*}$ be some linear operator. We aim at bounding the maximum of the spectral norm $\|\mathcal{Y}(\boldsymbol{\upsilon})\|$ over a vicinity $\Upsilon_\circ(\mathtt{r})\stackrel{\mathrm{def}}{=}\{\|\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*\|_{\mathcal{Y}}\le\mathtt{r}\}$ of $\boldsymbol{\upsilon}^*$. Suppose that $\mathcal{Y}(\boldsymbol{\upsilon})$ satisfies $\mathcal{Y}(\boldsymbol{\upsilon}^*)=0$ and for each $0<\mathtt{r}<\mathtt{r}^*$ and for all pairs $\boldsymbol{\upsilon},\boldsymbol{\upsilon}^\circ\in\Upsilon_\circ(\mathtt{r})=\bigl\{\boldsymbol{\upsilon}\in\Upsilon\colon\|\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^*\|_{\mathcal{Y}}\le\mathtt{r}\bigr\}\subset\mathbb{R}^{p^*}$

$$\sup_{\|\boldsymbol{u}_1\|\le1}\sup_{\|\boldsymbol{u}_2\|\le1}\log\mathbb{E}\exp\biggl\{\lambda\frac{\boldsymbol{u}_1^\top\bigl(\mathcal{Y}(\boldsymbol{\upsilon})-\mathcal{Y}(\boldsymbol{\upsilon}^\circ)\bigr)\boldsymbol{u}_2}{\epsilon_2\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^\circ)\|}\biggr\}\le\frac{\nu_2^2\lambda^2}{2}.\tag{47}$$

Remark 46 *In the setting of Theorem 14 we have $\|\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^\circ\|_{\mathcal{Y}}=\|\mathcal{D}(\boldsymbol{\upsilon}-\boldsymbol{\upsilon}^\circ)\|$ and*

$$\mathcal{Y}(\boldsymbol{\upsilon})=\mathcal{D}^{-1}\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon})-\mathcal{D}^{-1}\nabla^2\boldsymbol{\zeta}(\boldsymbol{\upsilon}^*),$$

and condition (47) *becomes* ($\mathcal{E}\mathcal{D}_2$) *from 2.1.*

Theorem 47 *Let a random process $\mathcal{Y}(\boldsymbol{\upsilon})\in\mathbb{R}^{p^*\times p^*}$ fulfill $\mathcal{Y}(\boldsymbol{\upsilon}^*)=0$ and let condition* (47) *be satisfied. Then for each $0\le\mathtt{r}\le\mathtt{r}^*$, on a set of probability greater than $1-\mathrm{e}^{-\mathtt{x}}$*

$$\sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})}\|\mathcal{Y}(\boldsymbol{\upsilon})\|\le9\epsilon_2\nu_2\mathfrak{z}_1(\mathtt{x},6p^*)\mathtt{r},$$

with $\mathtt{g}_0=\nu_0\mathtt{g}$.

Remark 48 *Note that the entropy of the original set $\Upsilon_\circ(\mathtt{r})\subset\mathbb{R}^{p^*}$ is multiplied by 3. So in order to control the spectral norm $\|\mathcal{Y}(\boldsymbol{\upsilon})\|$ one only pays with this factor.*

Proof In what follows, we use the representation

$$\|\mathcal{Y}(\boldsymbol{\upsilon})\|=\epsilon_2\sup_{\|\boldsymbol{u}_1\|\le\mathtt{r}}\sup_{\|\boldsymbol{u}_2\|\le\mathtt{r}}\frac{1}{\epsilon_2\mathtt{r}^2}\boldsymbol{u}_1^\top\breve{\mathcal{Y}}(\boldsymbol{\upsilon})\boldsymbol{u}_2.$$

This implies

$$\sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})}\|\mathcal{Y}(\boldsymbol{\upsilon})\|=\epsilon\sup_{\boldsymbol{\upsilon}\in\Upsilon_\circ(\mathtt{r})}\sup_{\|\boldsymbol{u}_2\|\le\mathtt{r}}\sup_{\|\boldsymbol{u}_2\|\le\mathtt{r}}\frac{1}{\epsilon\mathtt{r}^2}\boldsymbol{u}_1^\top\breve{\mathcal{Y}}(\boldsymbol{\upsilon})\boldsymbol{u}_2.$$

Due to Lemma 49 the process $\mathcal{U}(\upsilon) \stackrel{\mathrm{def}}{=} \frac{1}{\epsilon \mathtt{r}^2} \boldsymbol{u}_1^\top \mathcal{Y}(\upsilon) \boldsymbol{u}_2$ satisfies condition $(\mathcal{E}d)$ (see (45)) as process on

$$U(\mathtt{r}) \stackrel{\mathrm{def}}{=} \Upsilon_\circ(\mathtt{r}) \times B_{\mathtt{r}}(0) \times B_{\mathtt{r}}(0) \subset \mathbb{R}^{3p^*}. \tag{48}$$

This allows to apply Corollary 45 to obtain the desired result. We get on a set of probability greater $1 - \mathrm{e}^{-\mathtt{x}}$

$$\sup_{\upsilon \in \Upsilon_\circ(\mathtt{r})} \|\mathcal{Y}(\upsilon)\| \le \sup_{(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2) \in U(\mathtt{r})} \left\{ \frac{1}{\mathtt{r}^2} \boldsymbol{u}_1^\top \mathcal{Y}(\upsilon) \boldsymbol{u}_2 \right\} \le 9 \epsilon_2 \nu_2 \mathfrak{z}_1 \Bigl(\mathtt{x}, \mathbb{Q}\bigl(U(\mathtt{r}^*)\bigr) \Bigr) \mathtt{r}.$$

The constant $\mathbb{Q}\bigl(U(\mathtt{r})\bigr) > 0$ quantifies the complexity of the set $U(\mathtt{r}) \subset \mathbb{R}^{3p^*}$. We point out that for compact $M \subset \mathbb{R}^{3p^*}$ we have $\mathbb{Q}(M) = 6p^*$ (see Supplement of Spokoiny (2012), Lemma 2.10). This gives the claim. ∎

Lemma 49 *Suppose that* $\mathcal{Y}(\upsilon) \in \mathbb{R}^{p^* \times p^*}$ *satisfies* $\mathcal{Y}(\upsilon^*) = 0$ *and for each* $\|\boldsymbol{u}_1\| \le 1$, $\|\boldsymbol{u}_2\| \le 1$ *and* $|\lambda| \le \mathtt{g}$ *the inequality* (47). *Then the process*

$$\mathcal{U}(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2) = \frac{1}{2\epsilon_2 \mathtt{r}^2} \boldsymbol{u}_1^\top \mathcal{Y}(\upsilon)^\top \boldsymbol{u}_2$$

satisfies $(\boldsymbol{\mathcal{E}d})$ *from* (45) *with* $U \subset \mathbb{R}^{3p^*}$ *defined in* (48), *with* $|\lambda| \le \mathtt{g}/3$ *and with*

$$d((\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ))^2 = \|\mathcal{D}(\upsilon - \upsilon^*)\|^2 + \|\boldsymbol{u}_1 - \boldsymbol{u}_1^\circ\|^2 + \|\boldsymbol{u}_2 - \boldsymbol{u}_2^\circ\|^2,$$

i.e. for any $(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ) \in U$

$$\log \mathbb{E} \exp\left\{ \lambda \frac{\mathcal{U}(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2) - \mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ)}{d((\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ))} \right\} \le \frac{9\nu_2^2 \lambda^2}{2}, \qquad |\lambda| \le \mathtt{g}/3.$$

Proof Let $(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ) \in U$. By the Hölder inequality and (47), we find

$$\begin{aligned}
&\log \mathbb{E} \exp\left\{ \lambda \frac{\mathcal{U}(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2) - \mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ)}{d((\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ))} \right\} \\
&\quad = \log \mathbb{E} \exp\left\{ \lambda \left(\frac{\mathcal{U}(\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2) - \mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1, \boldsymbol{u}_2)}{d((\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ))} + \frac{\mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1, \boldsymbol{u}_2) - \mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2)}{d((\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ))} \right. \right. \\
&\qquad \left. \left. + \frac{\mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2) - \mathcal{U}(\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ)}{d((\upsilon, \boldsymbol{u}_1, \boldsymbol{u}_2), (\upsilon^\circ, \boldsymbol{u}_1^\circ, \boldsymbol{u}_2^\circ))} \right) \right\} \\
&\quad \le \frac{1}{3} \log \mathbb{E} \exp\left\{ 3\lambda \frac{\boldsymbol{u}_1^\top \bigl(\frac{1}{\mathtt{r}^2} \breve{\mathcal{Y}}(\upsilon) - \frac{1}{\mathtt{r}^2} \breve{\mathcal{Y}}(\upsilon^\circ) \bigr) \boldsymbol{u}_2}{\epsilon_2 \|\mathcal{D}(\upsilon - \upsilon^\circ)\|} \right\} \\
&\qquad + \frac{1}{3} \log \mathbb{E} \exp\left\{ 3\lambda \frac{(\boldsymbol{u}_1 - \boldsymbol{u}_1^\circ)^\top) \mathcal{Y}(\upsilon^\circ) \boldsymbol{u}_2}{\epsilon_2 \|\boldsymbol{u}_1 - \boldsymbol{u}_2\| \mathtt{r}^2} \right\} \\
&\qquad + \frac{1}{3} \log \mathbb{E} \exp\left\{ 3\lambda \frac{(\boldsymbol{u}_1^\circ)^\top) \mathcal{Y}(\upsilon^\circ) (\boldsymbol{u}_2 - \boldsymbol{u}_2^\circ)}{\epsilon_2 \|\boldsymbol{u}_1 - \boldsymbol{u}_2\| \mathtt{r}^2} \right\} \\
&\quad \le \frac{1}{3} \sup_{\|\boldsymbol{u}_1\| \le 1} \sup_{\|\boldsymbol{u}_2\| \le 1} \log \mathbb{E} \exp\left\{ 3\lambda \frac{\boldsymbol{u}_1^\top \bigl(\mathcal{Y}(\upsilon) - \mathcal{Y}(\upsilon^\circ) \bigr) \boldsymbol{u}_2}{\epsilon_2 \|\mathcal{D}(\upsilon - \upsilon^\circ)\|} \right\} \\
&\qquad + \frac{2}{3} \sup_{\|\boldsymbol{u}_1\| \le 1} \sup_{\|\boldsymbol{u}_2\| \le 1} \log \mathbb{E} \exp\left\{ 3\lambda \frac{\boldsymbol{u}_1^\top \bigl(\mathcal{Y}(\upsilon^\circ) - \mathcal{Y}(\upsilon^*) \bigr) \boldsymbol{u}_2}{\epsilon_2 \|\mathcal{D}(\upsilon - \upsilon^*)\|} \right\} \\
&\quad \le \frac{9\nu_2^2 \lambda^2}{2}, \qquad \lambda \le \mathtt{g}/3.
\end{aligned}$$

∎

References

A. Andresen. Finite sample analysis of profile m-estimation in the single index model. *Electronic Journal of Statistics*, 9(2):2528–2641, 2015.

A. Andresen and V. Spokoiny. Critical dimension in profile semiparametric estimation. *Electron. J. Statist.*, 8(2):3077–3125, 2014. ISSN 1935-7524. doi: 10.1214/14-EJS982.

S. Balakrishnan, M. J. Wainwright, and B. Yu. Satistical guarantees for the em algorithm: From population to sample-based analysis. *arXiv: 1408.2156*, 2014.

Xiaohong Chen. Large sample sieve estimation of semi-nonparametric models. *Handbook of Econometrics*, 6:55495632, 2007.

G. Cheng. How many iterations are sufficient for efficient semiparametric estimation? *Scandinavian Journal of Statistics,*, 40(3):592–618, 2013.

M. Delecroix., W. Haerdle, and M. Hristache. Efficient estimation in single-index regression. Technical report, SFB 373, Humboldt Univ. Berlin, 1997.

A.P Dempster, N.M. Laird, and D.B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society,*, Series B, 39:1–38, 1977.

U. Grenander. *Abstract Inference*. Whiley, New York, 1981.

W. Haerdle, P. Hall, and H. Ichimura. Optimal smoothing in single-index models. *Ann. Statist.*, 21:157–178, 1993.

I.A. Ibragimov and R.Z. Khas'minskij. *"Statistical estimation. Asymptotic theory. Transl. from the Russian by Samuel Kotz."*. "New York - Heidelberg -Berlin: Springer-Verlag", 1981.

P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. *STOC*, pages 665–674, 2013.

R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from few entries. *IEEE Transactions onInformation Theory*, 56(6):2980–2998, 2010.

M.R. Kosorok. *Introduction to Empirical Processes and Semiparametric Inference*. Springer in Statistics, 2005.

G.J. McLachlan and T Krishnan. *The EM Algorithm and Extensions*. Wiley, New York, 1997.

S. A. Murphy and A. W. van der Vaart. On profile likelihood. *Journal of the American Statistical Association*, 95(450):449–465, 2000.

P. Netrapalli, P. Jain, and S. Sanghavi. Phase retrieval using alternating minimization. *NIPS*, pages 2796–2804, 2013.

Vladimir Spokoiny. Parametric estimation. Finite sample theory. *Ann. Statist.*, 40(6): 2877–2909, 2012.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12:389–434, 2012.

C.F.J Wu. On the convergence properties of the em algorithm. *Annals of Statistics*, 11: 95–103, 1983.

X. Yi, C. Caramanis, and S. Sanghavi. Alternating minimization for mixed linear regression. *arXiv: 1310.3745*, 2013.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 10/15; Revised 3/16; Published 5/16

StructED : Risk Minimization in Structured Prediction

Yossi Adi ADIYOSS@CS.BIU.AC.IL
Joseph Keshet JOSEPH.KESHET@BIU.AC.IL
Department of Computer Science
Bar-Ilan University
Ramat Gan, 52900, Israel

Editor: Kevin Murphy

Abstract

Structured tasks are distinctive: each task has its own measure of performance, such as the word error rate in speech recognition, the BLEU score in machine translation, the NDCG score in information retrieval, or the intersection-over-union score in visual object segmentation. This paper presents StructED, a software package for learning structured prediction models with training methods that aimed at optimizing the task measure of performance. The package was written in Java and released under the MIT license. It can be downloaded from `http://adiyoss.github.io/StructED/`.

Keywords: structured prediction, structural SVM, CRF, direct loss minimization

1. Introduction

Ultimately the objective of discriminative training is to learn the model parameters so as to optimize a desired measure of performance. In binary classification, the objective is to find a prediction function that assigns a binary label to a single object, and minimizes the error rate (0-1 loss) on unseen data. In structured prediction, however, the goal is to predict a structured label (e.g., a graph, a sequence of words, a human pose, etc.), which often cannot be evaluated using the binary error rate, but rather with a task-specific measure of performance, generally called here *task loss*. There is a voluminous amount of work on training procedures for maximizing the BLEU score in machine translation, minimizing word/phone error rate in speech recognition or maximizing NDCG in information retrieval.

The most common approaches to learning parameters for structured prediction, namely structured perceptron, structural support vector machine (SSVM) and conditional random fields (CRF) do not directly minimize the task loss. The structured perceptron (Collins, 2002) solves a feasibility problem, which is independent of the task loss. The surrogate loss of SSVM (Joachims et al., 2005) is the structured hinge loss, a convex upper bound to the task loss, which is based on a generalization of the binary SVM hinge loss. However, while the binary SVM is strongly consistent, namely, converges to the error rate of the optimal linear predictor in the limit of infinite training data, the structured hinge loss is not consistent and does not converge to the expected task loss, i.e., the risk, of the optimal linear predictor even when the task loss is the 0-1 loss (McAllester, 2006). The objective of CRF is the log loss function, which is independent of the task loss (Lafferty et al., 2001).

Several structured prediction libraries already exist, such as CRF++ (Kudo, 2005), CRFsuite (Okazaki, 2007), Dlib (King, 2009), PyStruct (Müller and Behnke, 2014), and SVM-Struct (Joachims et al., 2009). All provide a general framework for training a model with either structured perceptron, CRF, or SSVM using several optimization techniques. Nevertheless, CRF++ and CRFsuite were not designed to support discriminative training with a task-specific evaluation metric, but rather use the binary error rate as their task loss function; whereas SVM-Struct, PyStruct, and Dlib support the training of structural SVM with a user-defined task loss function, but do not include, in their current phase, other training methods that are directly aimed at minimizing the task loss.

In this work we present StructED, a software package that is focused on training methods for structured prediction models that are aimed at minimizing a given task loss. The package provides several interfaces to define and implement a structured task, and allows the user to estimate the model parameters with several different training methods. The goal is not to provide several optimization alternatives to the structured hinge loss or the log loss as was already done in the previous packages, but rather to propose an implementation of a variety of surrogate loss functions that were designed to minimize the task loss and were theoretically motivated (McAllester and Keshet, 2011).

Note that most training methods require the task loss function to be decomposable in the size of the output label. Decomposable task loss functions are required in order to solve the loss-augmented inference that is used within the training procedure (Ranjbar et al., 2013), and evaluation metrics like intersection-over-union or word error rate which are not decomposable need to be approximated when utilized in SSVM, for example. Our package provides an implementation of methods (structured probit and orbit loss) that do not expect the task loss to be decomposable and can handle it directly without being approximated.

2. Structured Prediction and Risk Minimization

Consider a supervised learning setting with input instances $\boldsymbol{x} \in \mathcal{X}$ and target labels $\boldsymbol{y} \in \mathcal{Y}$, which refers to a set of objects with an internal structure. We assume a fixed mapping $\boldsymbol{\phi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ from the set of input objects and target labels to a real vector of length d, where we call the elements of this mapping *feature functions*. Also, consider a linear decoder with parameters $\boldsymbol{w} \in \mathbb{R}^d$, such that $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ is a good approximation to the true label of $\boldsymbol{x}$, as follows:

$$\hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \ \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) \tag{1}$$

Ideally, the learning algorithm finds $\boldsymbol{w}$ such that the prediction rule optimizes the expected desired *measure of preference* or *evaluation metric* on unseen data. We define the task loss function, $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}})$, to be a non-negative measure of error when predicting $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ instead of $\boldsymbol{y}$ as the label of $\boldsymbol{x}$. Our goal is to find the parameter vector $\boldsymbol{w}$ that minimizes this function. When the desired evaluation metric is a utility function that needs to be maximized (like BLEU or NDCG), we define the task loss to be 1 minus the evaluation metric.

Our goal is to set $\boldsymbol{w}$ so as to minimize the expected task loss (i.e., risk) for predicting $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$,

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}} \ \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim \rho}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))], \tag{2}$$

where the expectation is taken over pairs $(\boldsymbol{x}, \boldsymbol{y})$ drawn from an unknown probability distribution ρ. This objective function is hard to minimize directly. A common practice is to replace the task loss with a surrogate loss function, denoted $\bar{\ell}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y})$, which is easier to minimize; and to replace the expectation with a regularized average with respect to a set of training examples $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^m$, where each pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is drawn i.i.d from ρ:

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{m} \sum_{i=1}^{m} \bar{\ell}(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{y}_i) + \frac{\lambda}{2} \|\boldsymbol{w}\|^2, \tag{3}$$

where λ is a trade-off parameter between the loss term and the regularization. Different training algorithms are defined by different surrogate loss functions, e.g., the surrogate loss of max-margin Markov model (Taskar et al., 2003) is the structured hinge loss with the Hamming distance, whereas the one for CRF is the log loss function. These loss functions are convex in the model parameters, $\boldsymbol{w}$.

Recently, several works proposed different surrogate loss functions which are closer to the task loss in some sense: structured ramp-loss (McAllester and Keshet, 2011), structured probit loss (Keshet et al., 2011), and orbit loss (Karmon and Keshet, 2015). Direct loss minimization (McAllester et al., 2010) is a perceptron-like method of performing direct gradient descent on the risk (2) in the case where $\mathcal{Y}$ is discrete but $\mathcal{X}$ is continuous. All of the training objectives that were proposed in these works are non-convex functions in the model parameters, are strongly consistent and perform well in general.

3. Implementation

Implementation of a structured prediction task involves defining a set of feature functions. Given a trained model, the prediction is performed by finding the output label that maximizes the weighted sum of those feature functions according to (1). Such maximization involves the enumeration over all possible output labels, which is often intractable and handled by dynamic programming or by approximated inference techniques. As a result, and in contrast to packages for binary classification, here the user has to write code that implements the feature functions, the decoder and the evaluation metric.

Currently the implemented training methods are structured perceptron (SP), SSVM, CRF, structured passive-aggressive (SPA; Crammer et al., 2006), direct loss minimization (DLM), structured ramp loss (SRL), structured probit loss (SPL), and orbit loss (Karmon and Keshet, 2015).

The package exposes interfaces for task loss functions, feature extractors, decoding algorithms and training algorithms. In order to train a new model the user has to consider four interfaces, which correspond to: (i) a set of feature functions, $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})$; (ii) a task loss function[1], $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$; (iii) a decoder to perform the inference in (1), as well as the loss-augmented inference (needed by SPA, SRL, DLM, SSVM); and (iv) a training algorithm. The user can either implement those interfaces or use any of the already implemented interfaces.

The objective in (3) is optimized using stochastic gradient descent (SGD) for both the convex (SSVM and CRF) and non-convex (SRL, SPL, orbit) surrogate losses. DLM cannot be expressed as a surrogate loss function in objective (3), and is optimized using SGD as

1. BLUE and NDCG are currently not implemented.

well. SP and SPA are online algorithms and are implemented as batch algorithms with averaging of the weight vectors as an online-to-batch conversion (Cesa-Bianchi et al., 2004).

4. Experiments

While the library focuses on different training methods for minimizing a given task loss, it is important to verify that the implemented algorithms run in an acceptable amount of time. We compared our implementation of SSVM optimized by SGD to the corresponding implementation in PyStruct on the MNIST data set of handwritten digits. We got similar accuracy and training times (StructED : 92.6%, 149 sec; PyStruct: 90.2%, 145 sec).

We conclude the paper by demonstrating the advantage of the package with a simple structured prediction task of automatic vowel duration measurement. In this task the input is a speech segment of arbitrary duration that contains a vowel between two consonants (e.g., *bat*, *taught*, etc.), and the goal is to accurately predict the duration of the vowel. This task is a required measurement in linguistic studies and is often done manually. The training data includes speech segments that are labeled with the vowel onset and offset times, denoted y_b and y_e, respectively. The task loss is defined as $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \max\{0, |y_b - \hat{y}_b| - \tau_b\} + \max\{0, |y_e - \hat{y}_e| - \tau_e\}$, where $\hat{y}_b$ and $\hat{y}_e$ are predicted vowel onset and offset times, respectively, and τ_b and τ_e are parameters (τ_b=10 msec and τ_e=15 msec). We had a training set of 90 examples, a validation set of 20 examples and a test set of 20 examples. We extracted 21 unique acoustic features every 5 msec, including the confidence of a frame-based phoneme classifier, the first and the second formants and other acoustic features. We trained all algorithms on this task and present their performance in terms of task loss (the lower the better) and training times in Table 1 (training parameters for each of the training methods can be found in the package's Examples folder).

Algorithm	Task loss [msec]	Time [sec]
SP	72.5	13
SSVM	72.0	13
CRF	70.5	31
SPA	69.0	12
SRL	64.5	25
SPL	64.5	617
DLM	63.5	24
Orbit loss	58.5	13

Table 1: Error rate and training times on vowel duration measurement task.

Results suggest that training methods, which are designed to minimize the task loss, lead to improved performance compared to SP, SSVM, CRF or SPA. It is also evident that these methods, except orbit loss, take at least twice as much time to train: DLM and SRL need two inference operations per training iteration and SPL needs hundreds to thousands of inferences per training iteration.

The implementation of the above example is straightforward and is given in the Examples section of the package website `http://adiyoss.github.io/StructED/` along with several other usage examples and further details.

References

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Trans. on Information Theory*, 50(9):2050–2057, 2004.

M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceeding of EMNLP*, 2002.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

I. Joachims, T. Tsochantaridis, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

T. Joachims, T. Finley, and Chun-Nam Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

D. Karmon and J. Keshet. Risk minimization in structured prediction using orbit loss. 2015. (under submission).

J. Keshet, D. McAllester, and T. Hazan. PAC-Bayesian approach for minimization of phoneme error rate. In *Proceeding of ICASSP*, 2011.

D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

T. Kudo. CRF++: Yet another CRF toolkit, 2005.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, 2001.

D. McAllester. Generalization bounds and consistency for structured labeling. In B. Schölkopf, A. Smola, B. Taskar, and S.V.N. Vishwanathan, editors, *Predicting Structured Data*, pages 247–262. MIT Press, 2006.

D. McAllester and J. Keshet. Generalization bounds and consistency for latent structural probit and ramp loss. In *Proceedings of NIPS (25)*, 2011.

D. McAllester, T. Hazan, and J. Keshet. Direct loss minimization for structured prediction. In *Proceeding of NIPS (24)*, 2010.

A. C. Müller and S. Behnke. PyStruct - learning structured prediction in python. *Journal of Machine Learning Research*, 15:2055–2060, 2014.

N. Okazaki. CRFsuite: a fast implementation of conditional random fields (CRFs), 2007.

M. Ranjbar, T. Lan, Y. Wang, S.N. Robinovitch, Z.-N. Li, and G. Mori. Optimizing nondecomposable loss functions in structured prediction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 35(4):911–924, 2013.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Proceedings of NIPS (17)*, 2003.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 10/15; Revised 4/16; Published 4/16

Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches

Jure Žbontar[*] JURE.ZBONTAR@FRI.UNI-LJ.SI
Faculty of Computer and Information Science
University of Ljubljana
Večna pot 113, SI-1001 Ljubljana, Slovenia

Yann LeCun[†] YANN@CS.NYU.EDU
Courant Institute of Mathematical Sciences
New York University
715 Broadway, New York, NY 10003, USA

Editor: Zhuowen Tu

Abstract

We present a method for extracting depth information from a rectified image pair. Our approach focuses on the first stage of many stereo algorithms: the matching cost computation. We approach the problem by learning a similarity measure on small image patches using a convolutional neural network. Training is carried out in a supervised manner by constructing a binary classification data set with examples of similar and dissimilar pairs of patches. We examine two network architectures for this task: one tuned for speed, the other for accuracy. The output of the convolutional neural network is used to initialize the stereo matching cost. A series of post-processing steps follow: cross-based cost aggregation, semiglobal matching, a left-right consistency check, subpixel enhancement, a median filter, and a bilateral filter. We evaluate our method on the KITTI 2012, KITTI 2015, and Middlebury stereo data sets and show that it outperforms other approaches on all three data sets.

Keywords: stereo, matching cost, similarity learning, supervised learning, convolutional neural networks

1. Introduction

Consider the following problem: given two images taken by cameras at different horizontal positions, we wish to compute the disparity d for each pixel in the left image. Disparity refers to the difference in horizontal location of an object in the left and right image—an object at position (x, y) in the left image appears at position $(x-d, y)$ in the right image. If we know the disparity of an object we can compute its depth z using the following relation:

$$z = \frac{fB}{d},$$

∗. Jure Žbontar is also with the *Courant Institute of Mathematical Sciences, New York University, 715 Broadway, New York, NY 10003, USA.*
†. Yann LeCun is also with *Facebook AI Research, 770 Broadway, New York, NY 10003, USA.*

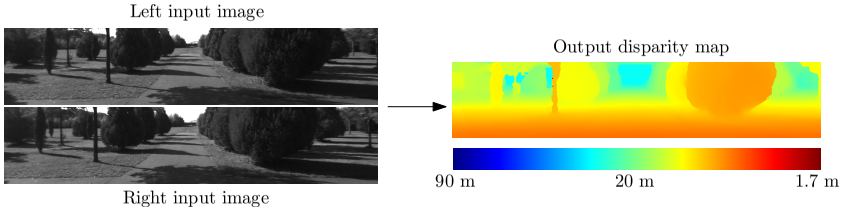


Figure 1: The input is a pair of images from the left and right camera. The two input images differ mostly in horizontal locations of objects (other differences are caused by reflections, occlusions, and perspective distortions). Note that objects closer to the camera have larger disparities than objects farther away. The output is a dense disparity map shown on the right, with warmer colors representing larger values of disparity (and smaller values of depth).

where f is the focal length of the camera and B is the distance between the camera centers. Figure 1 depicts the input to and the output from our method.

The described problem of stereo matching is important in many fields such as autonomous driving, robotics, intermediate view generation, and 3D scene reconstruction. According to the taxonomy of Scharstein and Szeliski (2002), a typical stereo algorithm consists of four steps: matching cost computation, cost aggregation, optimization, and disparity refinement. Following Hirschmüller and Scharstein (2009) we refer to the first two steps as computing the matching cost and the last two steps as the stereo method. The focus of this work is on computing a good matching cost.

We propose training a convolutional neural network (LeCun et al., 1998) on pairs of small image patches where the true disparity is known (for example, obtained by LIDAR or structured light). The output of the network is used to initialize the matching cost. We proceed with a number of post-processing steps that are not novel, but are necessary to achieve good results. Matching costs are combined between neighboring pixels with similar image intensities using cross-based cost aggregation. Smoothness constraints are enforced by semiglobal matching and a left-right consistency check is used to detect and eliminate errors in occluded regions. We perform subpixel enhancement and apply a median filter and a bilateral filter to obtain the final disparity map.

The contributions of this paper are

- a description of two architectures based on convolutional neural networks for computing the stereo matching cost;
- a method, accompanied by its source code, with the lowest error rate on the KITTI 2012, KITTI 2015, and Middlebury stereo data sets; and
- experiments analyzing the importance of data set size, the error rate compared with other methods, and the trade-off between accuracy and runtime for different settings of the hyperparameters.

This paper extends our previous work (Žbontar and LeCun, 2015) by including a description of a new architecture, results on two new data sets, lower error rates, and more thorough experiments.

2. Related Work

Before the introduction of large stereo data sets like KITTI and Middlebury, relatively few stereo algorithms used ground truth information to learn parameters of their models; in this section, we review the ones that did. For a general overview of stereo algorithms see Scharstein and Szeliski (2002).

Kong and Tao (2004) used the sum of squared distances to compute an initial matching cost. They then trained a model to predict the probability distribution over three classes: the initial disparity is correct, the initial disparity is incorrect due to fattening of a foreground object, and the initial disparity is incorrect due to other reasons. The predicted probabilities were used to adjust the initial matching cost. Kong and Tao (2006) later extend their work by combining predictions obtained by computing normalized cross-correlation over different window sizes and centers. Peris et al. (2012) initialized the matching cost with AD-Census (Mei et al., 2011), and used multiclass linear discriminant analysis to learn a mapping from the computed matching cost to the final disparity.

Ground-truth data was also used to learn parameters of probabilistic graphical models. Zhang and Seitz (2007) used an alternative optimization algorithm to estimate optimal values of Markov random field hyperparameters. Scharstein and Pal (2007) constructed a new data set of 30 stereo pairs and used it to learn parameters of a conditional random field. Li and Huttenlocher (2008) presented a conditional random field model with a non-parametric cost function and used a structured support vector machine to learn the model parameters.

Recent work (Haeusler et al., 2013; Spyropoulos et al., 2014) focused on estimating the confidence of the computed matching cost. Haeusler et al. (2013) used a random forest classifier to combine several confidence measures. Similarly, Spyropoulos et al. (2014) trained a random forest classifier to predict the confidence of the matching cost and used the predictions as soft constraints in a Markov random field to decrease the error of the stereo method.

A related problem to computing the matching cost is learning local image descriptors (Brown et al., 2011; Trzcinski et al., 2012; Simonyan et al., 2014; Revaud et al., 2015; Paulin et al., 2015; Han et al., 2015; Zagoruyko and Komodakis, 2015). The two problems share a common subtask: to measure the similarity between image patches. Brown et al. (2011) introduced a general framework for learning image descriptors and used Powell's method to select good hyperparameters. Several methods have been suggested for solving the problem of learning local image descriptors, such as boosting (Trzcinski et al., 2012), convex optimization (Simonyan et al., 2014), hierarchical moving-quadrant similarity (Revaud et al., 2015), convolutional kernel networks (Paulin et al., 2015), and convolutional neural networks (Zagoruyko and Komodakis, 2015; Han et al., 2015). Works of Zagoruyko and Komodakis (2015) and Han et al. (2015), in particular, are very similar to our own, differing mostly in the architecture of the network; concretely, the inclusion of pooling and subsampling to account for larger patch sizes and larger variation in viewpoint.

3. Matching Cost

A typical stereo algorithm begins by computing a matching cost at each position $\mathbf{p}$ for all disparities d under consideration. A simple method for computing the matching cost is the sum of absolute differences:

$$C_{\text{SAD}}(\mathbf{p}, d) = \sum_{\mathbf{q} \in \mathcal{N}_{\mathbf{p}}} |I^L(\mathbf{q}) - I^R(\mathbf{q} - \mathbf{d})|, \tag{1}$$

where $I^L(\mathbf{p})$ and $I^R(\mathbf{p})$ are image intensities at position $\mathbf{p}$ in the left and right image and $\mathcal{N}_{\mathbf{p}}$ is the set of locations within a fixed rectangular window centered at $\mathbf{p}$.

We use bold lowercase letters $\mathbf{p}$ and $\mathbf{q}$ to denote image locations. A bold lowercase $\mathbf{d}$ denotes the disparity d cast to a vector, that is, $\mathbf{d} = (d, 0)$. We use `typewriter` font for the names of hyperparameters. For example, we would use `patch_size` to denote the size of the neighbourhood area $\mathcal{N}_{\mathbf{p}}$.

Equation (1) can be interpreted as measuring the cost associated with matching a patch from the left image, centered at position $\mathbf{p}$, with a patch from the right image, centered at position $\mathbf{p} - \mathbf{d}$. We want the cost to be low when the two patches are centered around the image of the same 3D point, and high when they are not.

Since examples of good and bad matches can be constructed from publicly available data sets (for example, the KITTI and Middlebury stereo data sets), we can attempt to solve the matching problem by a supervised learning approach. Inspired by the successful application of convolutional neural networks to vision problems, we used them to assess how well two small image patches match.

3.1 Constructing the Data Set

We use ground truth disparity maps from either the KITTI or Middlebury stereo data sets to construct a binary classification data set. At each image position where the true disparity is known we extract one negative and one positive training example. This ensures that the data set contains an equal number of positive and negative examples. A positive example is a pair of patches, one from the left and one from the right image, whose center pixels are the images of the same 3D point, while a negative example is a pair of patches where this is not the case. The following section describes the data set construction step in detail.

Let $< \mathcal{P}^L_{n\times n}(\mathbf{p}), \mathcal{P}^R_{n\times n}(\mathbf{q}) >$ denote a pair of patches, where $\mathcal{P}^L_{n\times n}(\mathbf{p})$ is an $n \times n$ patch from the left image centered at position $\mathbf{p} = (x, y)$, $\mathcal{P}^R_{n\times n}(\mathbf{q})$ is an $n \times n$ patch from the right image centered at position $\mathbf{q}$, and d denotes the correct disparity at position $\mathbf{p}$. A negative example is obtained by setting the center of the right patch to

$$\mathbf{q} = (x - d + o_{\text{neg}}, y),$$

where o_{neg} is chosen from either the interval $[\texttt{dataset\_neg\_low}, \texttt{dataset\_neg\_high}]$ or, its origin reflected counterpart, $[-\texttt{dataset\_neg\_high}, -\texttt{dataset\_neg\_low}]$. The random offset o_{neg} ensures that the resulting image patches are not centered around the same 3D point.

A positive example is derived by setting

$$\mathbf{q} = (x - d + o_{\text{pos}}, y),$$

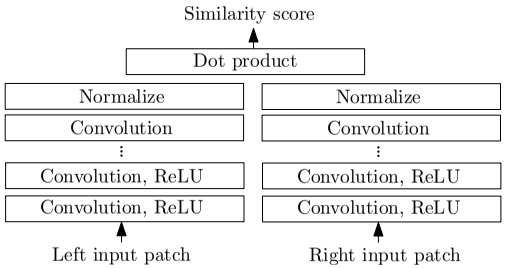


Figure 2: The fast architecture is a siamese network. The two sub-networks consist of a number of convolutional layers followed by rectified linear units (abbreviated "ReLU"). The similarity score is obtained by extracting a vector from each of the two input patches and computing the cosine similarity between them. In this diagram, as well as in our implementation, the cosine similarity computation is split in two steps: normalization and dot product. This reduces the running time because the normalization needs to be performed only once per position (see Section 3.3).

where o_{pos} is chosen randomly from the interval $[-\texttt{dataset\_pos}, \texttt{dataset\_pos}]$. The reason for including o_{pos}, instead of setting it to zero, has to do with the stereo method used later on. In particular, we found that cross-based cost aggregation performs better when the network assigns low matching costs to good matches as well as near matches. In our experiments, the hyperparameter `dataset_pos` was never larger than one pixel.

3.2 Network Architectures

We describe two network architectures for learning a similarity measure on image patches. The first architecture is faster than the second, but produces disparity maps that are slightly less accurate. In both cases, the input to the network is a pair of small image patches and the output is a measure of similarity between them. Both architectures contain a trainable feature extractor that represents each image patch with a feature vector. The similarity between patches is measured on the feature vectors instead of the raw image intensity values. The fast architecture uses a fixed similarity measure to compare the two feature vectors, while the accurate architecture attempts to learn a good similarity measure on feature vectors.

3.2.1 Fast Architecture

The first architecture is a siamese network, that is, two shared-weight sub-networks joined at the head (Bromley et al., 1993). The sub-networks are composed of a number of convolutional layers with rectified linear units following all but the last layer. Both sub-networks output a vector capturing the properties of the input patch. The resulting two vectors are

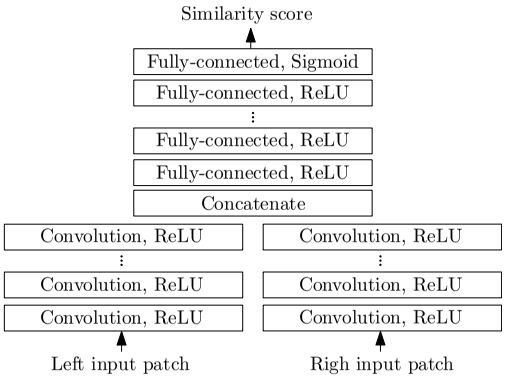


Figure 3: The accurate architecture begins with two convolutional feature extractors. The extracted feature vectors are concatenated and compared by a number of fully-connected layers. The inputs are two image patches and the output is a single real number between 0 and 1, which we interpret as a measure of similarity between the input images.

compared using the cosine similarity measure to produce the final output of the network. Figure 2 provides an overview of the architecture.

The network is trained by minimizing a hinge loss. The loss is computed by considering pairs of examples centered around the same image position where one example belongs to the positive and one to the negative class. Let s_+ be the output of the network for the positive example, s_- be the output of the network for the negative example, and let m, the margin, be a positive real number. The hinge loss for that pair of examples is defined as $\max(0, m + s_- - s_+)$. The loss is zero when the similarity of the positive example is greater than the similarity of the negative example by at least the margin m. We set the margin to 0.2 in our experiments.

The hyperparameters of this architecture are the number of convolutional layers in each sub-network (`num_conv_layers`), the size of the convolution kernels (`conv_kernel_size`), the number of feature maps in each layer (`num_conv_feature_maps`), and the size of the input patch (`input_patch_size`).

3.2.2 Accurate Architecture

The second architecture is derived from the first by replacing the cosine similarity measure with a number of fully-connected layers (see Figure 3). This architectural change increased the running time, but decreased the error rate. The two sub-networks comprise a number of convolutional layers, with a rectified linear unit following each layer. The resulting two vectors are concatenated and forward-propagated through a number of fully-connected

layers followed by rectified linear units. The last fully-connected layer produces a single number which, after being transformed with the sigmoid nonlinearity, is interpreted as the similarity score between the input patches.

We use the binary cross-entropy loss for training. Let s denote the output of the network for one training example and t denote the class of that training example; $t = 1$ if the example belongs to the positive class and $t = 0$ if the example belongs to the negative class. The binary cross-entropy loss for that example is defined as $t \log(s) + (1 - t) \log(1 - s)$.

The decision to use two different loss functions, one for each architecture, was based on empirical evidence. While we would have preferred to use the same loss function for both architectures, experiments showed that the binary cross-entropy loss performed better than the hinge loss on the accurate architecture. On the other hand, since the last step of the fast architecture is the cosine similarity computation, a cross-entropy loss was not directly applicable.

The hyperparameters of the accurate architecture are the number of convolutional layers in each sub-network (`num_conv_layers`), the number of feature maps in each layer (`num_conv_feature_maps`), the size of the convolution kernels (`conv_kernel_size`), the size of the input patch (`input_patch_size`), the number of units in each fully-connected layer (`num_fc_units`), and the number of fully-connected layers (`num_fc_layers`).

3.3 Computing the Matching Cost

The output of the network is used to initialize the matching cost:

$$C_{\text{CNN}}(\mathbf{p}, d) = -s(< \mathcal{P}^L(\mathbf{p}), \mathcal{P}^R(\mathbf{p} - \mathbf{d}) >),$$

where $s(< \mathcal{P}^L(\mathbf{p}), \mathcal{P}^R(\mathbf{p} - \mathbf{d}) >)$ is the output of the network when run on input patches $\mathcal{P}^L(\mathbf{p})$ and $\mathcal{P}^R(\mathbf{p} - \mathbf{d})$. The minus sign converts the similarity score to a matching cost.

To compute the entire matching cost tensor $C_{\text{CNN}}(\mathbf{p}, d)$ we would, naively, have to perform the forward pass for each image location and each disparity under consideration. The following three implementation details kept the running time manageable:

- The outputs of the two sub-networks need to be computed only once per location, and do not need to be recomputed for every disparity under consideration.

- The output of the two sub-networks can be computed for all pixels in a single forward pass by propagating full-resolution images, instead of small image patches. Performing a single forward pass on the entire $w \times h$ image is faster than performing $w \cdot h$ forward passes on small patches because many intermediate results can be reused.

- The output of the fully-connected layers in the accurate architecture can also be computed in a single forward pass. This is done by replacing each fully-connected layer with a convolutional layer with 1×1 kernels. We still need to perform the forward pass for each disparity under consideration; the maximum disparity d is 228 for the KITTI data set and 400 for the Middlebury data set. As a result, the fully-connected part of the network needs to be run d times, and is a bottleneck of the accurate architecture.

To compute the matching cost of a pair of images, we run the sub-networks once on each image and run the fully-connected layers d times, where d is the maximum disparity under consideration. This insight was important in designing the architecture of the network. We could have chosen an architecture where the two images are concatenated before being presented to the network, but that would imply a large cost at runtime because the whole network would need to be run d times. This insight also led to the development of the fast architecture, where the only layer that is run d times is the dot product of the feature vectors.

4. Stereo Method

The raw outputs of the convolutional neural network are not enough to produce accurate disparity maps, with errors particularly apparent in low-texture regions and occluded areas. The quality of the disparity maps can be improved by applying a series of post-processing steps referred to as the stereo method. The stereo method we used was influenced by Mei et al. (2011) and comprises cross-based cost aggregation, semiglobal matching, a left-right consistency check, subpixel enhancement, a median, and a bilateral filter.

4.1 Cross-based Cost Aggregation

Information from neighboring pixels can be combined by averaging the matching cost over a fixed window. This approach fails near depth discontinuities, where the assumption of constant depth within a window is violated. We might prefer a method that adaptively selects the neighborhood for each pixel, so that support is collected only from pixels of the same physical object. In cross-based cost aggregation (Zhang et al., 2009) we build a local neighborhood around each location comprising pixels with similar image intensity values with the hope that these pixels belong to the same object.

The method begins by constructing an upright cross at each position; this cross is used to define the local support region. The left arm $\mathbf{p}_l$ at position $\mathbf{p}$ extends left as long as the following two conditions hold:

- $|I(\mathbf{p}) - I(\mathbf{p}_l)| <$ `cbca_intensity`; the image intensities at positions $\mathbf{p}$ and $\mathbf{p}_l$ should be similar, their difference should be less than `cbca_intensity`.

- $\|\mathbf{p} - \mathbf{p}_l\| <$ `cbca_distance`; the horizontal distance (or vertical distance in case of top and bottom arms) between positions $\mathbf{p}$ and $\mathbf{p}_l$ is less than `cbca_distance` pixels.

The right, bottom, and top arms are constructed analogously. Once the four arms are known, we can compute the support region $U(\mathbf{p})$ as the union of horizontal arms of all positions $\mathbf{q}$ laying on $\mathbf{p}$'s vertical arm (see Figure 4).

Zhang et al. (2009) suggest that aggregation should consider the support regions of both images in a stereo pair. Let U^L and U^R denote the support regions in the left and right image. We define the combined support region U_d as

$$U_d(\mathbf{p}) = \{\mathbf{q} | \mathbf{q} \in U^L(\mathbf{p}), \mathbf{q} - \mathbf{d} \in U^R(\mathbf{p} - \mathbf{d})\}.$$

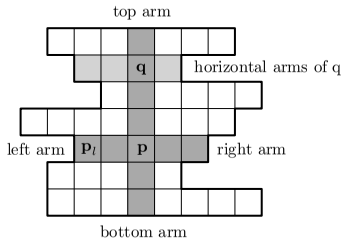


Figure 4: The support region for position $\mathbf{p}$ is the union of horizontal arms of all positions $\mathbf{q}$ on $\mathbf{p}$'s vertical arm.

The matching cost is averaged over the combined support region:

$$C^0_{\text{CBCA}}(\mathbf{p}, d) = C_{\text{CNN}}(\mathbf{p}, d),$$
$$C^i_{\text{CBCA}}(\mathbf{p}, d) = \frac{1}{|U_d(\mathbf{p})|} \sum_{\mathbf{q} \in U_d(\mathbf{p})} C^{i-1}_{\text{CBCA}}(\mathbf{q}, d),$$

where i is the iteration number. We repeat the averaging a number of times. Since the support regions are overlapping, the results can change at each iteration. We skip cross-based cost aggregation in the fast architecture because it is not crucial for achieving a low error rate and because it is relatively expensive to compute.

4.2 Semiglobal Matching

We refine the matching cost by enforcing smoothness constraints on the disparity image. Following Hirschmüller (2008), we define an energy function $E(D)$ that depends on the disparity image D:

$$E(D) = \sum_{\mathbf{p}} \Bigg(C^4_{\text{CBCA}}(\mathbf{p}, D(\mathbf{p})) + \sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} P_1 \cdot 1\{|D(\mathbf{p}) - D(\mathbf{q})| = 1\} + \sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} P_2 \cdot 1\{|D(\mathbf{p}) - D(\mathbf{q})| > 1\} \Bigg),$$

where $1\{\cdot\}$ denotes the indicator function. The first term penalizes disparities with high matching costs. The second term adds a penalty P_1 when the disparity of neighboring pixels differ by one. The third term adds a larger penalty P_2 when the neighboring disparities differ by more than one.

Rather than minimizing $E(D)$ in all directions simultaneously, we could perform the minimization in a single direction with dynamic programming. This solution would introduce unwanted streaking effects, since there would be no incentive to make the disparity image smooth in the directions we are not optimizing over. In semiglobal matching we

minimize the energy in a single direction, repeat for several directions, and average to obtain the final result. Although Hirschmüller (2008) suggested choosing sixteen direction, we only optimized along the two horizontal and the two vertical directions; adding the diagonal directions did not improve the accuracy of our system. To minimize $E(D)$ in direction $\mathbf{r}$, we define a matching cost $C_\mathbf{r}(\mathbf{p}, d)$ with the following recurrence relation:

$$C_\mathbf{r}(\mathbf{p}, d) = C^4_{\text{CBCA}}(\mathbf{p}, d) - \min_k C_r(\mathbf{p} - \mathbf{r}, k) + \min \Big\{ C_r(\mathbf{p} - \mathbf{r}, d), C_r(\mathbf{p} - \mathbf{r}, d-1) + P_1, C_r(\mathbf{p} - \mathbf{r}, d+1) + P_1, \min_k C_\mathbf{r}(\mathbf{p} - \mathbf{r}, k) + P_2 \Big\}.$$

The second term is subtracted to prevent values of $C_\mathbf{r}(\mathbf{p}, d)$ from growing too large and does not affect the optimal disparity map.

The penalty parameters P_1 and P_2 are set according to the image gradient so that jumps in disparity coincide with edges in the image. Let $D_1 = |I^L(\mathbf{p}) - I^L(\mathbf{p} - \mathbf{r})|$ and $D_2 = |I^R(\mathbf{p} - \mathbf{d}) - I^R(\mathbf{p} - \mathbf{d} - \mathbf{r})|$ be the difference in image intensity between two neighboring positions in the direction we are optimizing over. We set P_1 and P_2 according to the following rules:

$P_1 = $ `sgm_P1`,	$P_2 = $ `sgm_P2`	if $D_1 <$ `sgm_D`, $D_2 <$ `sgm_D`;
$P_1 = $ `sgm_P1/sgm_Q2`,	$P_2 = $ `sgm_P2/sgm_Q2`	if $D_1 \geq$ `sgm_D`, $D_2 \geq$ `sgm_D`;
$P_1 = $ `sgm_P1/sgm_Q1`,	$P_2 = $ `sgm_P2/sgm_Q1`	otherwise.

The hyperparameters `sgm_P1` and `sgm_P2` set a base penalty for discontinuities in the disparity map. The base penalty is reduced by a factor of `sgm_Q1` if one of D_1 or D_2 indicate a strong image gradient or by a larger factor of `sgm_Q2` if both D_1 and D_2 indicate a strong image gradient. The value of P_1 is further reduced by a factor of `sgm_V` when considering the two vertical directions; in the ground truth, small changes in disparity are much more frequent in the vertical directions than in the horizontal directions and should be penalised less.

The final cost $C_{\text{SGM}}(\mathbf{p}, d)$ is computed by taking the average across all four directions:

$$C_{\text{SGM}}(\mathbf{p}, d) = \frac{1}{4} \sum_\mathbf{r} C_\mathbf{r}(\mathbf{p}, d).$$

After semiglobal matching we repeat cross-based cost aggregation, as described in the previous section. Hyperparameters `cbca_num_iterations_1` and `cbca_num_iterations_2` determine the number of cross-based cost aggregation iterations before and after semiglobal matching.

4.3 Computing the Disparity Image

The disparity image $D(\mathbf{p})$ is computed by the winner-takes-all strategy, that is, by finding the disparity d that minimizes $C(\mathbf{p}, d)$,

$$D(\mathbf{p}) = \operatorname*{argmin}_d C(\mathbf{p}, d).$$

4.3.1 Interpolation

The interpolation steps attempt to resolve conflicts between the disparity map predicted for the left image and the disparity map predicted for the right image. Let D^L denote the disparity map obtained by treating the left image as the reference image—this was the case so far, that is, $D^L(\mathbf{p}) = D(\mathbf{p})$—and let D^R denote the disparity map obtained by treating the right image as the reference image. D^L and D^R sometimes disagree on what the correct disparity at a particular position should be. We detect these conflicts by performing a left-right consistency check. We label each position $\mathbf{p}$ by applying the following rules in turn:

$$\begin{array}{ll} \textit{correct} & \text{if } |d - D^R(\mathbf{p} - \mathbf{d})| \leq 1 \text{ for } d = D^L(\mathbf{p}), \\ \textit{mismatch} & \text{if } |d - D^R(\mathbf{p} - \mathbf{d})| \leq 1 \text{ for any other } d, \\ \textit{occlusion} & \text{otherwise.} \end{array}$$

For positions marked as *occlusion*, we want the new disparity value to come from the background. We interpolate by moving left until we find a position labeled *correct* and use its value. For positions marked as *mismatch*, we find the nearest *correct* pixels in 16 different directions and use the median of their disparities for interpolation. We refer to the interpolated disparity map as D_{INT}.

4.3.2 Subpixel Enhancement

Subpixel enhancement provides an easy way to increase the resolution of a stereo algorithm. We fit a quadratic curve through the neighboring costs to obtain a new disparity image:

$$D_{\text{SE}}(\mathbf{p}) = d - \frac{C_+ - C_-}{2(C_+ - 2C + C_-)},$$

where $d = D_{\text{INT}}(\mathbf{p})$, $C_- = C_{\text{SGM}}(\mathbf{p}, d-1)$, $C = C_{\text{SGM}}(\mathbf{p}, d)$, and $C_+ = C_{\text{SGM}}(\mathbf{p}, d+1)$.

4.3.3 Refinement

The final steps of the stereo method consist of a 5×5 median filter and the following bilateral filter:

$$D_{\text{BF}}(\mathbf{p}) = \frac{1}{W(\mathbf{p})} \sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} D_{\text{SE}}(\mathbf{q}) \cdot g(\|\mathbf{p} - \mathbf{q}\|) \cdot 1\{|I^L(\mathbf{p}) - I^L(\mathbf{q})| < \texttt{blur\_threshold}\},$$

where $g(x)$ is the probability density function of a zero mean normal distribution with standard deviation `blur_sigma` and $W(\mathbf{p})$ is the normalizing constant,

$$W(\mathbf{p}) = \sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} g(\|\mathbf{p} - \mathbf{q}\|) \cdot 1\{|I^L(\mathbf{p}) - I^L(\mathbf{q})| < \texttt{blur\_threshold}\}.$$

The role of the bilateral filter is to smooth the disparity map without blurring the edges. D_{BF} is the final output of our stereo method.

5. Experiments

We used three stereo data sets in our experiments: KITTI 2012, KITTI 2015, and Middlebury. The test set error rates reported in Tables 1, 2, and 4 were obtained by submitting

Rank	Method		Setting	Error	Runtime
1	**MC-CNN-acrt**	**Accurate architecture**		2.43	67
2	Displets	Güney and Geiger (2015)		2.47	265
3	MC-CNN	Žbontar and LeCun (2015)		2.61	100
4	PRSM	Vogel et al. (2015)	F, MV	2.78	300
	MC-CNN-fst	**Fast architecture**		2.82	0.8
5	SPS-StFl	Yamaguchi et al. (2014)	F, MS	2.83	35
6	VC-SF	Vogel et al. (2014)	F, MV	3.05	300
7	Deep Embed	Chen et al. (2015)		3.10	3
8	JSOSM	Unpublished work		3.15	105
9	OSF	Menze and Geiger (2015)	F	3.28	3000
10	CoR	Chakrabarti et al. (2015)		3.30	6

Table 1: The highest ranking methods on the KITTI 2012 data set as of October 2015. The "Setting" column provides insight into how the disparity map is computed: "F" indicates the use of optical flow, "MV" indicates more than two temporally adjacent images, and "MS" indicates the use of epipolar geometry for computing the optical flow. The "Error" column reports the percentage of misclassified pixels and the "Runtime" column measures the time, in seconds, required to process one pair of images.

the generated disparity maps to the online evaluation servers. All other error rates were computed by splitting the data set in two, using one part for training and the other for validation.

5.1 KITTI Stereo Data Set

The KITTI stereo data set (Geiger et al., 2013; Menze and Geiger, 2015) is a collection of rectified image pairs taken from two video cameras mounted on the roof of a car, roughly 54 centimeters apart. The images were recorded while driving in and around the city of Karlsruhe, in sunny and cloudy weather, at daytime. The images were taken at a resolution of 1240×376. A rotating laser scanner mounted behind the left camera recorded ground truth depth, labeling around 30 % of the image pixels.

The ground truth disparities for the test set are withheld and an online leaderboard is provided where researchers can evaluate their method on the test set. Submissions are allowed once every three days. Error is measured as the percentage of pixels where the true disparity and the predicted disparity differ by more than three pixels. Translated into distance, this means that, for example, the error tolerance is 3 centimeters for objects 2 meters from the camera and 80 centimeters for objects 10 meters from the camera.

Rank	Method		Setting	Error	Runtime
1	**MC-CNN-acrt**	**Accurate architecture**		3.89	67
	MC-CNN-fst	**Fast architecture**		4.62	0.8
2	SPS-St	Yamaguchi et al. (2014)		5.31	2
3	OSF	Menze and Geiger (2015)	F	5.79	3000
4	PR-Sceneflow	Vogel et al. (2013)	F	6.24	150
5	SGM+C+NL	Hirschmüller (2008); Sun et al. (2014)	F	6.84	270
6	SGM+LDOF	Hirschmüller (2008); Brox and Malik (2011)	F	6.84	86
7	SGM+SF	Hirschmüller (2008); Hornacek et al. (2014)	F	6.84	2700
8	ELAS	Geiger et al. (2011)		9.72	0.3
9	OCV-SGBM	Hirschmüller (2008)		10.86	1.1
10	SDM	Kostková and Sára (2003)		11.96	60

Table 2: The leading submission on the KITTI 2015 leaderboard as of October 2015. The "Setting", "Error", and "Runtime" columns have the same meaning as in Table 1.

Two KITTI stereo data sets exist: KITTI 2012[1] and, the newer, KITTI 2015[2]. For the task of computing stereo they are nearly identical, with the newer data set improving some aspects of the optical flow task. The 2012 data set contains 194 training and 195 testing images, while the 2015 data set contains 200 training and 200 testing images. There is a subtle but important difference introduced in the newer data set: vehicles in motion are densely labeled and car glass is included in the evaluation. This emphasizes the method's performance on reflective surfaces.

The best performing methods on the KITTI 2012 data set are listed in Table 1. Our accurate architecture ranks first with an error rate of 2.43 %. Third place on the leaderboard is held by our previous work (Žbontar and LeCun, 2015) with an error rate of 2.61 %. The two changes that reduced the error from 2.61 % to 2.43 % were augmenting the data set (see Section 5.4) and doubling the number of convolution layers while reducing the kernel size from 5×5 to 3×3. The method in second place (Güney and Geiger, 2015) uses the matching cost computed by our previous work (Žbontar and LeCun, 2015). The test error rate of the fast architecture is 2.82 %, which would be enough for fifth place had the method been allowed to appear in the public leaderboard. The running time for processing a single image pair is 67 seconds for the accurate architecture and 0.8 seconds for the fast architecture. Figure 5 contains a pair of examples from the KITTI 2012 data set, together with the predictions of our method.

Table 2 presents the frontrunners on the KITTI 2015 data sets. The error rates of our methods are 3.89 % for the accurate architecture and 4.46 % for the fast architecture, occupying first and second place on the leaderboard. Since one submission per paper is

1. The KITTI 2012 scoreboard: `http://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php?benchmark=stereo`
2. The KITTI 2015 scoreboard: `http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo`

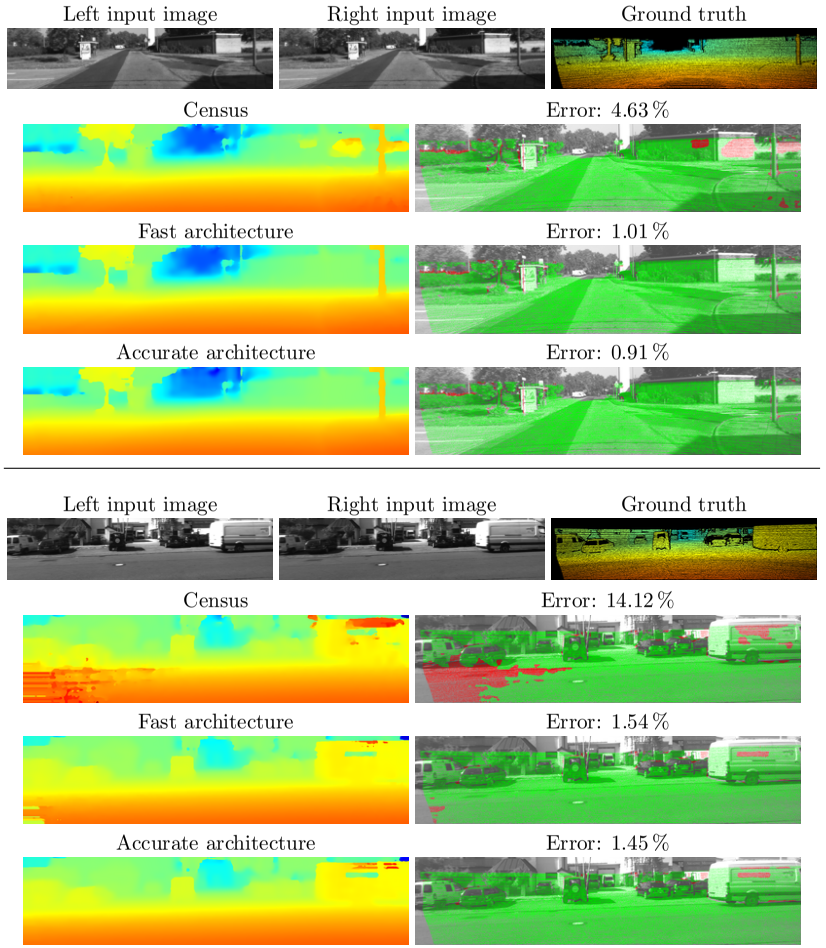


Figure 5: Examples of predicted disparity maps on the KITTI 2012 data set. Note how some regions of the image (the white wall in the top example, and the asphalt in the bottom example) cause problems for the census transform. The fast and the accurate architecture perform better, with the accurate architecture making fewer mistakes on average.

Year	Number of Image Pairs	Resolution	Maximum Disparity
2001	8	380×430	30
2003	2	1800×1500	220
2005	6	1400×1100	230
2006	21	1400×1100	230
2014	23	3000×2000	800

Table 3: A summary of the five Middlebury stereo data sets. The column "Number of Image Pairs" counts only the image pairs for which ground truth is available. The 2005 and 2014 data sets additionally contain a number of image pairs with ground truth disparities withheld; these image pairs constitute the test set.

allowed, only the result of the accurate architecture appears on the public leaderboard. See Figure 6 for the disparity maps produced by our method on the KITTI 2015 data set.

5.2 Middlebury Stereo Data Set

The image pairs of the Middlebury stereo data set are indoor scenes taken under controlled lighting conditions. Structured light was used to measure the true disparities with higher density and precision than in the KITTI data set. The data sets were published in five separate works in the years 2001, 2003, 2005, 2006, and 2014 (Scharstein and Szeliski, 2002, 2003; Scharstein and Pal, 2007; Hirschmüller and Scharstein, 2007; Scharstein et al., 2014). In this paper, we refer to the Middlebury data set as the concatenation of all five data sets; a summary of each is presented in Table 3.

Each scene in the 2005, 2006, and 2014 data sets was taken under a number of lighting conditions and shutter exposures, with a typical image pair taken under four lighting conditions and seven exposure settings for a total of 28 images of the same scene.

An online leaderboard[3], similar to the one provided by KITTI, displays a ranked list of all submitted methods. Participants have only one opportunity to submit their results on the test set to the public leaderboard. This rule is stricter than the one on the KITTI data set, where submissions are allowed every three days. The test set contains 15 images borrowed from the 2005 and 2014 data sets.

The data set is provided in full, half, and quarter resolution. The error is computed at full resolution; if the method outputs half or quarter resolution disparity maps, they are upsampled before the error is computed. We chose to run our method on half resolution images because of the limited size of the graphic card's memory available.

Rectifying a pair of images using standard calibration procedures, like the ones present in the OpenCV library, results in vertical disparity errors of up to nine pixels on the Middlebury data set (Scharstein et al., 2014). Each stereo pair in the 2014 data set is rectified twice: once using a standard, imperfect approach, and once using precise 2D correspondences for perfect rectification (Scharstein et al., 2014). We train the network on imperfectly rectified

3. The Middlebury scoreboard: `http://vision.middlebury.edu/stereo/eval3/`

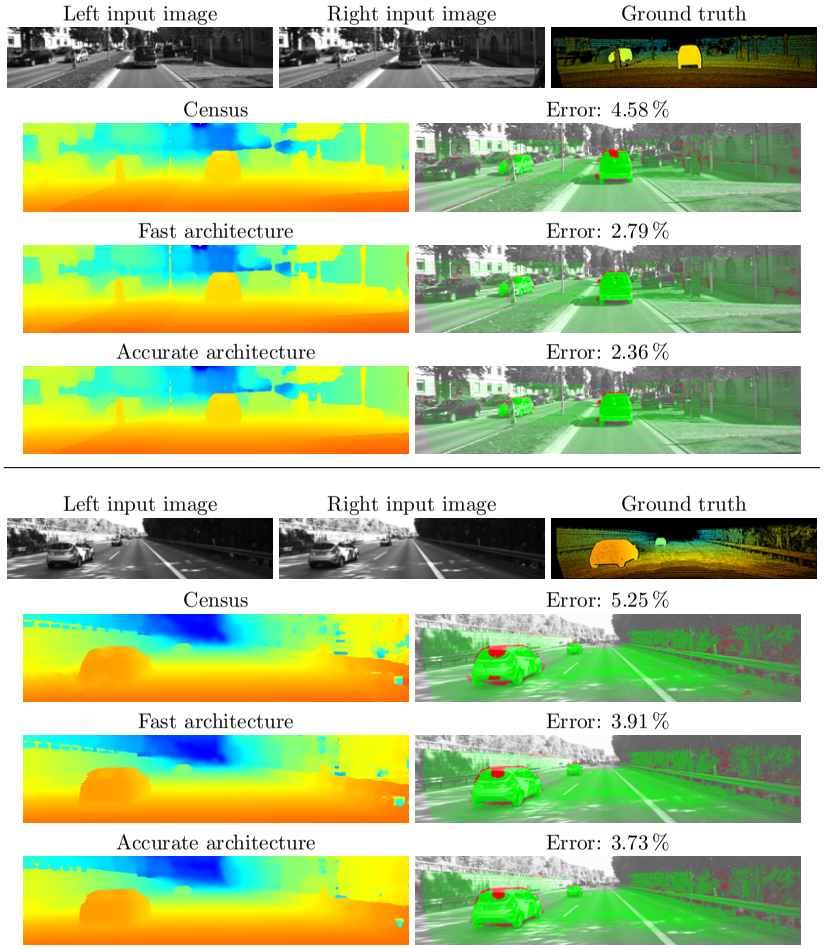


Figure 6: Examples of predictions on the KITTI 2015 data set. Observe that vehicles in motion are labeled densely in the KITTI 2015 data set.

Rank	Method		Resolution	Error	Runtime
1	**MC-CNN-acrt**	**Accurate architecture**	Half	8.29	150
2	MeshStereo	Zhang et al. (2015)	Half	13.4	65.3
3	LCU	Unpublished work	Quarter	17.0	6567
4	TMAP	Psota et al. (2015)	Half	17.1	2435
5	IDR	Kowalczuk et al. (2013)	Half	18.4	0.49
6	SGM	Hirschmüller (2008)	Half	18.7	9.90
7	LPS	Sinha et al. (2014)	Half	19.4	9.52
8	LPS	Sinha et al. (2014)	Full	20.3	25.8
9	SGM	Hirschmüller (2008)	Quarter	21.2	1.48
10	SNCC	Einecke and Eggert (2010)	Half	22.2	1.38

Table 4: The top ten methods on the Middlebury stereo data set as of October 2015. The "Error" column is the weighted average error after upsampling to full resolution and "Runtime" is the time, in seconds, required to process one pair of images.

image pairs, since only two of the fifteen test images (*Australia* and *Crusade*) are rectified perfectly.

The error is measured as the percentage of pixels where the true disparity and the predicted disparity differ by more than two pixels; this corresponds to an error tolerance of one pixel at half resolution. The error on the evaluation server is, by default, computed only on non-occluded pixels. The final error reported online is the weighted average over the fifteen test images, with weights set by the authors of the data set.

Table 4 contains a snapshot of the third, and newest, version of the Middlebury leaderboard. Our method ranks first with an error rate of 8.29 % and a substantial lead over the second placed MeshStereo method, whose error rate is 13.4 %. See Figure 7 for disparity maps produced by our method on one image pair from the Middlebury data set.

5.3 Details of Learning

We construct a binary classification data set from all available image pairs in the training set. The data set contains 25 million examples on the KITTI 2012, 17 million examples on the KITTI 2015, and 38 million examples on the Middlebury data set.

At training time, the input to the network was a batch of 128 pairs of image patches. At test time, the input was the entire left and right image. We could have used entire images during training as well, as it would allow us to implement the speed optimizations described in Section 3.3. There were several reasons why we preferred to train on image patches: it was easier to control the batch size, the examples could be shuffled so that one batch contained patches from several different images, and it was easier to maintain the same number of positive and negative examples within a batch.

We minimized the loss using mini-batch gradient descent with the momentum term set to 0.9. We trained for 14 epochs with the learning rate initially set to 0.003 for the accurate architecture and 0.002 for the fast architecture. The learning rate was decreased

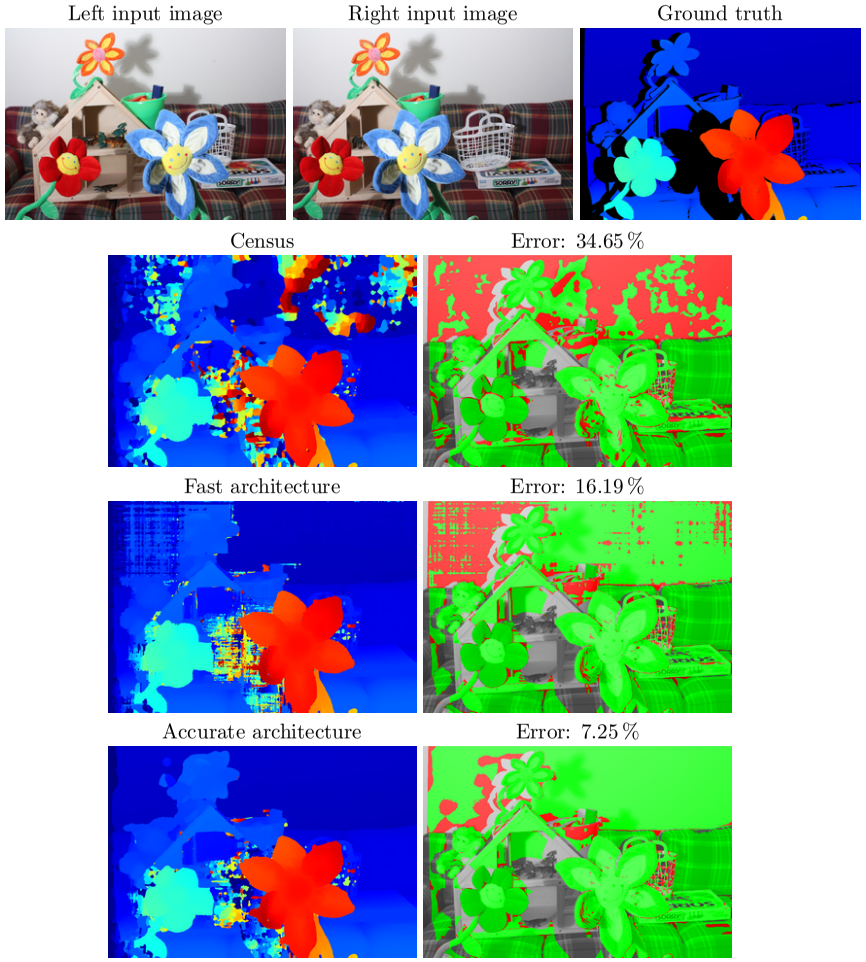


Figure 7: An example of a particularly difficult image pair from the Middlebury data set; the white wall in the background is practically textureless. The accurate architecture is able to classify most of it correctly. The fast architecture doesn't do as well but still performs better than census.

	KITTI 2012		KITTI 2015		Middlebury	
Hyperparameter	fst	acrt	fst	acrt	fst	acrt
`input_patch_size`	9×9	9×9	9×9	9×9	11×11	11×11
`num_conv_layers`	4	4	4	4	5	5
`num_conv_feature_maps`	64	112	64	112	64	112
`conv_kernel_size`	3	3	3	3	3	3
`num_fc_layers`		4		4		3
`num_fc_units`		384		384		384
`dataset_neg_low`	4	4	4	4	1.5	1.5
`dataset_neg_high`	10	10	10	10	6	18
`dataset_pos`	1	1	1	1	0.5	0.5
`cbca_intensity`		0.13		0.03		0.02
`cbca_distance`		5		5		14
`cbca_num_iterations_1`		2		2		2
`cbca_num_iterations_2`		0		4		16
`sgm_P1`	4	1.32	2.3	2.3	2.3	1.3
`sgm_P2`	223	32	42.3	55.8	55.9	18.1
`sgm_Q1`	3	3	3	3	4	4.5
`sgm_Q2`	7.5	6	6	6	8	9
`sgm_V`	1.5	2	1.25	1.75	1.5	2.75
`sgm_D`	0.02	0.08	0.08	0.08	0.08	0.13
`blur_sigma`	7.74	6	4.64	6	6	1.7
`blur_threshold`	5	6	5	5	2	2

Table 5: The hyperparameter values we used for the fast and accurate architectures (abbreviated "fst" and "acrt"). Note that hyperparameters concerning image intensity values (`cbca_intensity` and `sgm_D`) apply to the preprocessed images and not to raw images with intensity values in the range from 0 to 255.

by a factor of 10 on the 11^{th} epoch. The number of epochs, the initial learning rate, and the learning rate decrease schedule where treated as hyperparameters and were optimized with cross-validation. Each image was preprocessed by subtracting the mean and dividing by the standard deviation of its pixel intensity values. The left and right image of a stereo pair were preprocessed separately. Our initial experiments suggested that using color information does not improve the quality of the disparity maps; therefore, we converted all color images to grayscale.

The post-processing steps of the stereo method were implemented in CUDA (Nickolls et al., 2008), the network training was done with the Torch environment (Collobert et al., 2011) using the convolution routines from the cuDNN library (Chetlur et al., 2014). The OpenCV library (Bradski, 2000) was used for the affine transformation in the data augmentation step.

The hyperparameters where optimized with manual search and simple scripts that helped automate the process. The hyperparameters we selected are shown in Table 5.

5.4 Data Set Augmentation

Augmenting the data set by repeatedly transforming the training examples is a commonly employed technique to reduce the network's generalization error. The transformations are applied at training time and do not affect the runtime performance. We randomly rotate, scale and shear the training patches; we also change their brightness and contrast. Since the transformations are applied to patches after they have been extracted from the images, the data augmentation step does not alter the ground truth disparity map or ruin the rectification.

The parameters of the transformation are chosen randomly for each pair of patches, and after one epoch of training, when the same example is being presented to the network for the second time, new random parameters are selected. We choose slightly different transformation parameters for the left and right image; for example, we would rotate the left patch by 10 degrees and the right by 14. Different data sets benefited from different types of transformations and, in some cases, using the wrong transformations increased the error.

On the Middlebury data set we took advantage of the fact that the images were taken under different lighting conditions and different shutter exposures by training on all available images. The same data set augmentation parameters were used for the KITTI 2012 and KITTI 2015 data sets.

The Middlebury test data sets contains two images worth mentioning: *Classroom*, where the right image is underexposed and, therefore, darker than the left; and *Djembe*, where the left and right images were taken under different light conditions. To handle these two cases we train, 20 % of the time, on images where either the shutter exposure or the arrangements of lights are different for the left and right image.

We combat imperfect rectification on the Middlebury data set by including a small vertical disparity between the left and right image patches.

Before describing the steps of data augmentation, let us introduce some notation: in the following, a word in `typewriter` is used to denote the name of a hyperparameter defining a set, while the same word in *italic* is used to denote a number drawn randomly from that set. For example, `rotate` is a hyperparameter defining the set of possible rotations and *rotate* is a number drawn randomly from that set. The steps of data augmentation are presented in the following list:

- Rotate the left patch by *rotate* degrees and the right patch by *rotate* + *rotate_diff* degrees.
- Scale the left patch by *scale* and the right patch by *scale* · *scale_diff*.
- Scale the left patch in the horizontal direction by *horizontal_scale* and the right patch by *horizontal_scale* · *horizontal_scale_diff*.
- Shear the left patch in the horizontal direction by *horizontal_shear* and the right patch by *horizontal_shear* + *horizontal_shear_diff*.

	KITTI 2012		Middlebury	
Hyperparameter	Range	Error	Range	Error
`rotate`	$[-7, 7]$	2.65	$[-28, 28]$	7.99
`scale`			$[0.8, 1]$	8.17
`horizontal_scale`	$[0.9, 1]$	2.62	$[0.8, 1]$	8.08
`horizontal_shear`	$[0, 0.1]$	2.61	$[0, 0.1]$	7.91
`brightness`	$[0, 0.7]$	2.61	$[0, 1.3]$	8.16
`contrast`	$[1, 1.3]$	2.63	$[1, 1.1]$	7.95
`vertical_disparity`			$[0, 1]$	8.05
`rotate_diff`			$[-3, 3]$	8.00
`horizontal_scale_diff`			$[0.9, 1]$	7.97
`horizontal_shear_diff`			$[0, 0.3]$	8.05
`brightness_diff`	$[0, 0.3]$	2.63	$[0, 0.7]$	7.92
`contrast_diff`			$[1, 1.1]$	8.01
No data set augmentation		2.73		8.75
Full data set augmentation		2.61		7.91

Table 6: The hyperparameters governing data augmentation and how they affect the validation error. The "Error" column reports the validation error when a particular data augmentation step is not used. The last two rows report validation errors with and without data augmentation. For example, the validation error on the KITTI 2012 is 2.73 % if no data augmentation is used, 2.65 % if all steps except rotation are used, and 2.61 % if all data augmentation steps are used.

- Translate the right patch in the vertical direction by *vertical_disparity*.
- Adjust the brightness and contrast by setting the left and right image patches to:

$$\mathcal{P}^L \leftarrow \mathcal{P}^L \cdot \mathit{contrast} + \mathit{brightness} \text{ and}$$
$$\mathcal{P}^R \leftarrow \mathcal{P}^R \cdot (\mathit{contrast} \cdot \mathit{contrast\_diff}) + (\mathit{brightness} + \mathit{brightness\_diff}),$$

  with addition and multiplication carried out element-wise where appropriate.

Table 6 contains the hyperparameters used and measures how each data augmentation step affected the validation error.

Data augmentation reduced the validation error from 2.73 % to 2.61 % on the KITTI 2012 data set and from 8.75 % to 7.91 % on the Middlebury data set.

5.5 Runtime

We measure the runtime of our implementation on a computer with a NVIDIA Titan X graphics processor unit. Table 7 contains the runtime measurements across a range of hyperparameter settings for three data sets: KITTI, Middlebury half resolution, and a

new, fictitious data set, called Tiny, which we use to demonstrate the performance of our method on the kind of images typically used for autonomous driving or robotics. The sizes of images we measured the runtime on were: 1242 × 350 with 228 disparity levels for the KITTI data set, 1500 × 1000 with 200 disparity levels for the Middlebury data set, and 320 × 240 with 32 disparity levels for the Tiny data set.

Table 7 reveals that the fast architecture is up to 90 times faster than the accurate architecture. Furthermore, the running times of the fast architecture are 0.78 seconds on KITTI, 2.03 seconds on Middlebury, and 0.06 seconds on the Tiny data set. We can also see that the fully-connected layers are responsible for most of the runtime in the accurate architecture, as the hyperparameters controlling the number of convolutional layer and the number of feature maps have only a small effect on the runtime.

Training times depended on the size of the data set and the architecture, but never exceeded two days.

5.6 Matching Cost

We argue that the low error rate of our method is due to the convolutional neural network and not a superior stereo method. We verify this claim by replacing the convolutional neural network with three standard approaches for computing the matching cost:

- *The sum of absolute differences* computes the matching cost according to Equation (1), that is, the matching cost between two image patches is computed by summing the absolute differences in image intensities between corresponding locations. We used 9 × 9 patches.
- *The census transform* (Zabih and Woodfill, 1994) represents each image position as a bit vector. The size of this vector is a hyperparameter whose value, after examining several, we set to 81. The vector is computed by cropping a 9×9 image patch centered around the position of interest and comparing the intensity values of each pixel in the patch to the intensity value of the pixel in the center. When the center pixel is brighter the corresponding bit is set. The matching cost is computed as the hamming distance between two census transformed vectors.
- *Normalized cross-correlation* is a window-based method defined with the following equation:

$$C_{NCC}(\mathbf{p}, d) = \frac{\sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} I^L(\mathbf{q}) I^R(\mathbf{q} - \mathbf{d})}{\sqrt{\sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} I^L(\mathbf{q})^2 \sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} I^R(\mathbf{q} - \mathbf{d})^2}}.$$

  The normalized cross-correlation matching cost computes the cosine similarity between the left and right image patch, when the left and right image patches are viewed as vectors instead of matrices. This is the same function that is computed in the last two layers of the fast architecture (normalization and dot product). The neighbourhood $\mathcal{N}_\mathbf{p}$ was set to a square 11×11 window around $\mathbf{p}$.

The "sad", "cens", and "ncc" columns of Table 8 contain the results of the sum of absolute differences, the census transform, and normalized cross-correlation on the KITTI 2012, KITTI 2015, and Middlebury data sets. The validation errors in the last rows of Table 8

Hyperparameter		KITTI fst	KITTI acrt	Middlebury fst	Middlebury acrt	Tiny fst	Tiny acrt
num_conv_layers	1	0.23	66.1	0.70	74.9	0.01	1.7
	2	0.26	66.2	0.82	75.2	0.01	1.8
	3	0.30	66.3	0.97	75.4	0.02	1.8
	4	0.34	66.4	1.11	75.6	0.03	1.8
	5	0.38	66.5	1.24	75.7	0.03	1.9
	6	0.42	66.7	1.37	76.0	0.04	1.9
num_conv_feature_maps	16	0.09	59.4	0.27	64.8	0.01	1.6
	32	0.15	60.4	0.51	66.2	0.01	1.6
	48	0.25	61.5	0.94	68.2	0.02	1.7
	64	0.34	62.7	1.24	70.0	0.03	1.7
	80	0.44	64.0	1.63	72.0	0.04	1.8
	96	0.53	65.3	1.93	73.9	0.04	1.8
	112	0.61	66.4	2.28	75.7	0.05	1.8
	128	0.71	67.7	2.61	77.8	0.06	1.9
num_fc_layers	1		16.3		25.3		0.5
	2		32.9		50.7		0.9
	3		49.6		75.7		1.4
	4		66.4		101.2		1.8
	5		82.9		126.4		2.3
num_fc_units	128		17.4		21.4		0.6
	256		38.5		44.9		1.1
	384		66.4		75.7		1.8
	512		101.0		113.3		2.7
No stereo method		0.34	66.4	1.24	75.7	0.03	1.8
Full stereo method		0.78	67.1	2.03	84.8	0.06	1.9

Table 7: The time, in seconds, required to compute the matching cost, that is, the time spent in the convolutional neural network without any post-processing steps. The time does include computing the matching cost twice: once when the left image is taken to be the reference image and once when the right image is taken to be the reference image. We measure the runtime as a function of four hyperparameters controlling the network architecture; for example, the first six rows contain the runtime as the number of convolutional layers in the network increases from one to six. The last row of the table contains the running time for the entire method, including the post-processing steps. As before, we abbreviate the fast and accurate architectures as "fst" and "acrt".

should be used to compare the five methods. On all three data sets the accurate architecture performs best, followed by the fast architecture, which in turn is followed by the census transform. These are the three best performing methods on all three data sets. Their error rates are 2.61 %, 3.02 %, and 4.90 % on KITTI 2012; 3.25 %, 3.99 %, and 5.03 % on KITTI 2015; and 7.91 %, 9.87 %, and 16.72 % on Middlebury. The sum of absolute differences and the normalized cross-correlation matching costs produce disparity maps with larger errors. For a visual comparison of our method and the census transform see Figures 5, 6, and 7.

5.7 Stereo Method

The stereo method includes a number of post-processing steps: cross-based cost aggregation, semiglobal matching, interpolation, subpixel enhancement, a median, and a bilateral filter. We ran a set of experiments in which we excluded each of the aforementioned steps and recorded the validation error (see Table 8).

The last two rows of Table 8 allude to the importance of the post-processing steps of the stereo method. We see that, if all post-processing steps are removed, the validation error of the accurate architecture increases from 2.61 % to 13.49 % on KITTI 2012, from 3.25 % to 13.38 % on KITTI 2015, and from 7.91 % to 28.33 % on Middlebury.

Out of all post-processing steps of the stereo method, semiglobal matching affects the validation error the strongest. If we remove it, the validation error increases from 2.61 % to 4.26 % on KITTI 2012, from 3.25 % to 4.51 % on KITTI 2015, and from 7.91 % to 11.99 % on Middlebury.

We did not use the left-right consistency check to eliminate errors in occluded regions on the Middlebury data set. The error rate increased from 7.91 % to 8.22 % using the left-right consistency check on the accurate architecture, which is why we decided to remove it.

5.8 Data Set Size

We used a supervised learning approach to measure the similarity between image patches. It is, therefore, natural to ask how does the size of the data set affect the quality of the disparity maps. To answer this question, we retrain our networks on smaller training sets obtained by selecting a random set of examples (see Table 9).

We observe that the validation error decreases as we increase the number of training examples. These experiments suggest a simple strategy for improving the results of our stereo method: collect a larger data set.

5.9 Transfer Learning

Up to this point the training and validation sets were created from the same stereo data set, either KITTI 2012, KITTI 2015, or Middlebury. To evaluate the performance of our method in the transfer learning setting, we run experiments where the validation error is computed on a different data set than the one used for training. For example, we would use the Middlebury data set to train the matching cost neural network and evaluate its performance on the KITTI 2012 data set. These experiments give us some idea of the expected performance in a real-world application, where it isn't possible to train a specialized

	KITTI 2012				
	fst	acrt	sad	cens	ncc
Cross-based cost aggregation	3.02	2.73	8.22	5.21	8.93
Semiglobal matching	8.78	4.26	19.58	8.84	10.72
Interpolation	3.48	2.96	9.21	5.96	11.16
Subpixel Enhancement	3.03	2.65	8.16	4.95	8.93
Median filter	3.03	2.63	8.16	4.92	9.00
Bilateral filter	3.26	2.79	8.75	5.70	9.76
No stereo method	15.70	13.49	32.30	53.55	22.21
Full stereo method	3.02	2.61	8.16	4.90	8.93
	KITTI 2015				
	fst	acrt	sad	cens	ncc
Cross-based cost aggregation	3.99	3.39	9.94	5.20	8.89
Semiglobal matching	8.40	4.51	19.80	7.25	9.36
Interpolation	4.47	3.33	10.39	5.83	10.98
Subpixel Enhancement	4.02	3.28	9.44	5.03	8.91
Median filter	4.05	3.25	9.44	5.05	8.96
Bilateral filter	4.20	3.43	9.95	5.84	9.77
No stereo method	15.66	13.38	30.67	50.35	18.95
Full stereo method	3.99	3.25	9.44	5.03	8.89
	Middlebury				
	fst	acrt	sad	cens	ncc
Cross-based cost aggregation	9.87	10.63	43.09	29.28	33.89
Semiglobal matching	25.50	11.99	51.25	19.51	35.36
Interpolation	9.87	7.91	41.86	16.72	33.89
Subpixel Enhancement	10.29	8.44	42.71	17.18	34.12
Median filter	10.16	7.91	41.90	16.73	34.17
Bilateral filter	10.39	7.96	41.97	16.96	34.43
No stereo method	30.84	28.33	59.57	64.53	39.23
Full stereo method	9.87	7.91	41.86	16.72	33.89

Table 8: The numbers measure validation error when a particular post-processing step is excluded from the stereo method. The last two rows of the tables should be interpreted differently: they contain the validation error of the raw convolutional neural network and the validation error after the complete stereo method. For example, if we exclude semiglobal matching, the fast architecture achieves an error rate of 8.78 % on the KITTI 2012 data set and an error rate of 3.02 % after applying the full stereo method. We abbreviate the method names as "fst" for the fast architecture, "acrt" for the accurate architecture, "sad" for the sum of absolute differences, "cens" for the census transform, and "ncc" for the normalized cross-correlation matching cost.

	KITTI 2012		KITTI 2015		Middlebury	
Data Set Size (%)	fst	acrt	fst	acrt	fst	acrt
20	3.17	2.84	4.13	3.53	11.14	9.73
40	3.11	2.75	4.10	3.40	10.35	8.71
60	3.09	2.67	4.05	3.34	10.14	8.36
80	3.05	2.65	4.02	3.29	10.09	8.21
100	3.02	2.61	3.99	3.25	9.87	7.91

Table 9: The validation error as a function of training set size.

		Test Set					
		KITTI 2012		KITTI 2015		Middlebury	
		fst	acrt	fst	acrt	fst	acrt
Training Set	KITTI 2012	3.02	2.61	4.12	3.99	12.78	11.09
	KITTI 2015	3.60	4.28	3.99	3.25	13.70	14.19
	Middlebury	3.16	3.07	4.48	4.49	9.87	7.91

Table 10: The validation error when the training and test sets differ. For example, the validation error is 3.16 % when the Middlebury data set is used for training the fast architecture and the trained network is tested on the KITTI 2012 data set.

network because no ground truth is available. The results of these experiments are shown in Table 10.

Some results in Table 10 were unexpected. For example, the validation error on KITTI 2012 is lower when using the Middlebury training set compared to the KITTI 2015 training set, even though the KITTI 2012 data set is obviously more similar to KITTI 2015 than Middlebury. Furthermore, the validation error on KITTI 2012 is lower when using the fast architecture instead of the accurate architecture when training on KITTI 2015.

The matching cost neural network trained on the Middlebury data set transfers well to the KITTI data sets. Its validation error is similar to the validation errors obtained by networks trained on the KITTI data sets.

5.10 Hyperparameters

Searching for a good set of hyperparameters is a daunting task—with the search space growing exponentially with the number of hyperparameters and no gradient to guide us. To better understand the effect of each hyperparameter on the validation error, we conduct a series of experiments where we vary the value of one hyperparameter while keeping the others fixed to their default values. The results are shown in Table 11 and can be summarized by observing that increasing the size of the network improves the generalization

Hyperparameter		KITTI 2012 fst	KITTI 2012 acrt	KITTI 2015 fst	KITTI 2015 acrt	Middlebury fst	Middlebury acrt
num_conv_layers	1	5.96	3.97	5.61	4.06	20.74	12.37
	2	3.52	2.98	4.19	3.45	12.11	9.20
	3	3.10	2.72	4.04	3.27	10.81	8.56
	4	3.02	2.61	3.99	3.25	10.26	8.21
	5	3.03	2.64	3.99	3.30	9.87	7.91
	6	3.05	2.70	4.01	3.38	9.71	8.11
num_conv_feature_maps	16	3.33	2.84	4.32	3.51	11.79	10.06
	32	3.15	2.68	4.12	3.35	10.48	8.67
	48	3.07	2.66	4.06	3.32	10.20	8.47
	64	3.02	2.64	3.99	3.30	9.87	8.12
	80	3.02	2.64	3.99	3.29	9.81	7.95
	96	2.99	2.68	3.97	3.27	9.62	8.03
	112	2.98	2.61	3.96	3.25	9.59	7.91
	128	2.97	2.63	3.95	3.23	9.45	7.92
num_fc_layers	1		2.83		3.50		8.52
	2		2.70		3.31		8.33
	3		2.62		3.30		8.06
	4		2.61		3.25		8.00
	5		2.62		3.29		7.91
num_fc_units	128		2.72		3.36		8.44
	256		2.65		3.28		8.03
	384		2.61		3.25		7.91
	512		2.60		3.23		7.90
dataset_neg_low	1.0	3.00	2.76	3.97	3.35	9.84	8.00
	1.5	3.00	2.71	3.97	3.33	9.87	7.91
	2.0	2.99	2.63	3.98	3.31	9.98	8.08
	4.0	3.02	2.61	3.99	3.25	10.20	8.66
	6.0	3.06	2.63	4.05	3.28	10.13	8.86
dataset_neg_high	6	3.00	2.72	3.98	3.30	9.87	8.59
	10	3.02	2.61	3.99	3.25	9.97	8.23
	14	3.04	2.61	4.02	3.25	10.00	8.05
	18	3.07	2.60	4.06	3.23	9.98	8.11
	22	3.07	2.61	4.05	3.24	10.16	7.91
dataset_pos	0.0	3.04	2.67	4.00	3.26	9.92	7.97
	0.5	3.02	2.65	3.99	3.28	9.87	7.91
	1.0	3.02	2.61	3.99	3.25	9.86	8.04
	1.5	3.04	2.62	4.04	3.27	10.00	8.34
	2.0	3.04	2.66	4.04	3.29	10.16	8.51

Table 11: Validation errors computed across a range of hyperparameter settings.

performance, but only up to a point, when presumably, because of the size of the data set, the generalization performance starts do decrease.

Note that the `num_conv_layers` hyperparameter implicitly controls the size of the image patches. For example, a network with one convolutional layer with 3×3 kernels compares image patches of size 3×3, while a network with five convolutional layers compares patches of size 11×11.

6. Conclusion

We presented two convolutional neural network architectures for learning a similarity measure on image patches and applied them to the problem of stereo matching.

The source code of our implementation is available at `https://github.com/jzbontar/mc-cnn`. The online repository contains procedures for computing the disparity map, training the network, as well as the post-processing steps of the stereo method.

The accurate architecture produces disparity maps with lower error rates than any previously published method on the KITTI 2012, KITTI 2015, and Middlebury data sets. The fast architecture computes the disparity maps up to 90 times faster than the accurate architecture with only a small increase in error. These results suggest that convolutional neural networks are well suited for computing the stereo matching cost even for applications that require real-time performance.

The fact that a relatively simple convolutional neural network outperformed all previous methods on the well-studied problem of stereo is a rather important demonstration of the power of modern machine learning approaches.

References

G. Bradski. The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 2000.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993.

Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1): 43–57, 2011.

Thomas Brox and Jitendra Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

Ayan Chakrabarti, Ying Xiong, Steven J. Gortler, and Todd Zickler. Low-level vision by consensus in a spatial hierarchy of regions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Zhuoyuan Chen, Xun Sun, Yinan Yu, Liang Wang, and Chang Huang. A deep visual correspondence embedding model for stereo matching costs. *IEEE International Conference on Computer Vision (ICCV)*, 2015.

Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. cuDNN: Efficient primitives for deep learning. *CoRR*, abs/1410.0759, 2014. URL `http://arxiv.org/abs/1410.0759`.

Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.

Nils Einecke and Julian Eggert. A two-stage correlation method for stereoscopic depth estimation. In *Digital Image Computing: International Conference on Techniques and Applications (DICTA)*, pages 227–234, 2010.

Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In *Proceedings of the 10th Asian Conference on Computer Vision - Volume Part I*, ACCV'10, pages 25–38. Springer-Verlag, Berlin, Heidelberg, 2011.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: the KITTI dataset. *International Journal of Robotics Research (IJRR)*, 2013.

Fatma Güney and Andreas Geiger. Displets: Resolving stereo ambiguities using object knowledge. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Ralf Haeusler, Rahul Nair, and Daniel Kondermann. Ensemble learning for confidence measures in stereo vision. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. MatchNet: Unifying feature and metric learning for patch-based matching. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008.

Heiko Hirschmüller and Daniel Scharstein. Evaluation of cost functions for stereo matching. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

Heiko Hirschmüller and Daniel Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1582–1599, 2009.

Michael Hornacek, Andrew Fitzgibbon, and Carsten Rother. SphereFlow: 6 DoF scene flow from RGB-D pairs. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Dan Kong and Hai Tao. A method for learning matching errors for stereo computation. *British Machine Vision Conference (BMVC)*, 2004.

Dan Kong and Hai Tao. Stereo matching via learning multiple experts behaviors. *British Machine Vision Conference (BMVC)*, 2006.

Jana Kostková and Radim Sára. Stratified dense matching for stereopsis in complex scenes. *British Machine Vision Conference (BMVC)*, 2003.

Jedrzej Kowalczuk, Eric T Psota, and Lance C Perez. Real-time stereo matching on CUDA using an iterative refinement method for adaptive support-weight correspondences. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(1):94–104, 2013.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yunpeng Li and Daniel P Huttenlocher. Learning for stereo vision using the structured support vector machine. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

Xing Mei, Xun Sun, Mingcai Zhou, Haitao Wang, Xiaopeng Zhang, et al. On building an accurate stereo matching system on graphics hardware. *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 467–474, 2011.

Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with CUDA. *Queue*, 6(2):40–53, 2008.

Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid. Local convolutional features with unsupervised training for image retrieval. In *IEEE International Conference on Computer Vision (ICCV)*, pages 91–99, 2015.

Martin Peris, Atsuto Maki, Sara Martull, Yasuhiro Ohkawa, and Kazuhiro Fukui. Towards a simulation driven stereo vision system. In *21st International Conference on Pattern Recognition (ICPR)*, pages 1038–1042, 2012.

Eric T Psota, Jedrzej Kowalczuk, Mateusz Mittek, and Lance C Perez. Map disparity estimation using hidden markov trees. *IEEE International Conference on Computer Vision (ICCV)*, 2015.

Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Deepmatching: Hierarchical deformable dense matching. *ArXiv e-prints*, 1(7):8, 2015.

Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2007.

Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3): 7–42, 2002.

Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2003.

Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. *German Conference on Pattern Recognition (GCPR)*, September 2014.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1573–1585, 2014.

Sudipta N Sinha, Daniel Scharstein, and Richard Szeliski. Efficient high-resolution stereo matching using local plane sweeps. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Aristotle Spyropoulos, Nikos Komodakis, and Philippos Mordohai. Learning to detect ground control points for improving the accuracy of stereo matching. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision*, 106(2):115–137, 2014.

Tomasz Trzcinski, Mario Christoudias, Vincent Lepetit, and Pascal Fua. Learning image descriptors with the boosting-trick. In *Advances in neural information processing systems*, pages 269–277, 2012.

Christoph Vogel, Konrad Schindler, and Stefan Roth. Piecewise rigid scene flow. *IEEE International Conference on Computer Vision (ICCV)*, 2013.

Christoph Vogel, Stefan Roth, and Konrad Schindler. View-consistent 3D scene flow estimation over multiple frames. *European Conference on Computer Vision (ECCV)*, September 2014.

Christoph Vogel, Konrad Schindler, and Stefan Roth. 3D scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision*, pages 1–28, 2015.

Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. *European Conference on Computer Vision (ECCV)*, September 2014.

Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. *European Conference on Computer Vision (ECCV)*, 1994.

Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Jure Žbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Chi Zhang, Zhiwei Li, Yanhua Cheng, Rui Cai, Hongyang Chao, and Yong Rui. Meshstereo: A global stereo model with mesh alignment regularization for view interpolation. *IEEE International Conference on Computer Vision (ICCV)*, 2015.

Ke Zhang, Jiangbo Lu, and Gauthier Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7):1073–1079, 2009.

Li Zhang and Steven M Seitz. Estimating optimal parameters for MRF stereo from a single image pair. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2): 331–342, 2007.

Journal of Machine Learning Research 17 (2016) 1-53 Submitted 9/10; Revised 4/16; Published 8/16

Bayesian Policy Gradient and Actor-Critic Algorithms

Mohammad Ghavamzadeh* MOHAMMAD.GHAVAMZADEH@INRIA.FR
Adobe Research & INRIA
Yaakov Engel YAKIENGEL@GMAIL.COM
Rafael Advanced Defence System, Israel
Michal Valko MICHAL.VALKO@INRIA.FR
INRIA Lille — SequeL team, France

Editor: Jan Peters

Abstract

Policy gradient methods are reinforcement learning algorithms that adapt a parameterized policy by following a performance gradient estimate. Many conventional policy gradient methods use Monte-Carlo techniques to estimate this gradient. The policy is improved by adjusting the parameters in the direction of the gradient estimate. Since Monte-Carlo methods tend to have high variance, a large number of samples is required to attain accurate estimates, resulting in slow convergence. In this paper, we first propose a Bayesian framework for policy gradient, based on modeling the policy gradient as a Gaussian process. This reduces the number of samples needed to obtain accurate gradient estimates. Moreover, estimates of the natural gradient as well as a measure of the uncertainty in the gradient estimates, namely, the gradient covariance, are provided at little extra cost. Since the proposed Bayesian framework considers system trajectories as its basic observable unit, it does not require the dynamics within trajectories to be of any particular form, and thus, can be easily extended to partially observable problems. On the downside, it cannot take advantage of the Markov property when the system is Markovian.

To address this issue, we proceed to supplement our Bayesian policy gradient framework with a new actor-critic learning model in which a Bayesian class of non-parametric critics, based on Gaussian process temporal difference learning, is used. Such critics model the action-value function as a Gaussian process, allowing Bayes' rule to be used in computing the posterior distribution over action-value functions, conditioned on the observed data. Appropriate choices of the policy parameterization and of the prior covariance (kernel) between action-values allow us to obtain closed-form expressions for the posterior distribution of the gradient of the expected return with respect to the policy parameters. We perform detailed experimental comparisons of the proposed Bayesian policy gradient and actor-critic algorithms with classic Monte-Carlo based policy gradient methods, as well as with each other, on a number of reinforcement learning problems.

Keywords: reinforcement learning, policy gradient methods, actor-critic algorithms, Bayesian inference, Gaussian processes

∗. Mohammad Ghavamzadeh is at Adobe Research, on leave of absence from INRIA Lille - Team SequeL.

1. Introduction

Policy gradient (PG) methods[1] are reinforcement learning (RL) algorithms that maintain a parameterized action-selection policy and update the policy parameters by moving them in the direction of an estimate of the gradient of a performance measure. Early examples of PG algorithms are the class of REINFORCE algorithms (Williams, 1992),[2] which are suitable for solving problems in which the goal is to optimize the average reward. Subsequent work (e.g., Kimura et al., 1995, Marbach, 1998, Baxter and Bartlett, 2001) extended these algorithms to the cases of infinite-horizon Markov decision processes (MDPs) and partially observable MDPs (POMDPs), while also providing much needed theoretical analysis.[3] However, both the theoretical results and empirical evaluations have highlighted a major shortcoming of these algorithms, namely, the high variance of the gradient estimates. This problem may be traced to the fact that in most interesting cases, the time-average of the observed rewards is a high-variance (although unbiased) estimator of the true average reward, resulting in the sample-inefficiency of these algorithms.

One solution proposed for this problem was to use an artificial *discount factor* in these algorithms (Marbach, 1998, Baxter and Bartlett, 2001), however, this creates another problem by introducing bias into the gradient estimates. Another solution, which does not involve biasing the gradient estimate, is to subtract a *reinforcement baseline* from the average reward estimate in the updates of PG algorithms (e.g., Williams, 1992, Marbach, 1998, Sutton et al., 2000). In Williams (1992) an average reward baseline was used, and in Sutton et al. (2000) it was conjectured that an approximate value function would be a good choice for a state-dependent baseline. However, it was shown in Weaver and Tao (2001) and Greensmith et al. (2004), perhaps surprisingly, that the mean reward is in general *not* the optimal constant baseline, and that the true value function is generally *not* the optimal state-dependent baseline.

A different approach for speeding-up PG algorithms was proposed by Kakade (2002) and refined and extended by Bagnell and Schneider (2003) and Peters et al. (2003). The idea is to replace the policy gradient estimate with an estimate of the so-called *natural* policy gradient. This is motivated by the requirement that the policy updates should be invariant to bijective transformations of the parametrization. Put more simply, a change in the way the policy is parametrized should not influence the result of the policy update. In terms of the policy update rule, the move to natural-gradient amounts to linearly transforming the gradient using the inverse Fisher information matrix of the policy. In empirical evaluations, natural PG has been shown to significantly outperform conventional PG (e.g., Kakade 2002, Bagnell and Schneider 2003, Peters et al. 2003, Peters and Schaal 2008).

1. The term has been coined in Sutton et al. (2000), but here we use it more liberally to refer to a whole class of reinforcement learning algorithms.
2. Note that policy gradient methods have been studied in the control community (see e.g., Dyer and McReynolds 1970, Jacobson and Mayne 1970, Hasdorff 1976) before REINFORCE. However, unlike REINFORCE that is model-free, they were all based on the exact model of the system (model-based).
3. It is important to note that the pioneering work of Gullapali and colleagues in the early 1990s (Gullapalli, 1990, 1992, Gullapalli et al., 1994) in applying policy gradient methods to robot learning problems had an important role in popularizing this class of algorithms. In fact policy gradient methods have been continuously proven to be one of the most effective class of algorithms in learning in robots.

Another approach for reducing the variance of policy gradient estimates, and as a result making the search in the policy-space more efficient and reliable, is to use an explicit representation for the value function of the policy. This class of PG algorithms are called actor-critic algorithms. Actor-critic (AC) algorithms comprise a family of RL methods that maintain two distinct algorithmic components: An *actor*, whose role is to maintain and update an action-selection policy; and a *critic*, whose role is to estimate the value function associated with the actor's policy. Thus, the critic addresses the problem of *prediction*, whereas the actor is concerned with *control*. Actor-critic methods were among the earliest to be investigated in RL (Barto et al., 1983, Sutton, 1984). They were largely supplanted in the 1990's by methods, such as SARSA (Rummery and Niranjan, 1994), that estimate action-value functions and use them directly to select actions without maintaining an explicit representation of the policy. This approach was appealing because of its simplicity, but when combined with function approximation was found to be unreliable, often failing to converge. These problems led to renewed interest in PG methods.

Actor-critic algorithms (e.g., Sutton et al. 2000, Konda and Tsitsiklis 2000, Peters et al. 2005, Bhatnagar et al. 2007) borrow elements from these two families of RL algorithms. Like value-function based methods, a critic maintains a value function estimate, while an actor maintains a separately parameterized stochastic action-selection policy, as in policy based methods. While the role of the actor is to select actions, the role of the critic is to evaluate the performance of the actor. This evaluation is used to provide the actor with a feedback signal that allows it to improve its performance. The actor typically updates its policy along an estimate of the gradient (or natural gradient) of some measure of performance with respect to the policy parameters. When the representations used for the actor and the critic are *compatible*, in the sense explained in Sutton et al. (2000) and Konda and Tsitsiklis (2000), the resulting AC algorithm is simple, elegant, and provably convergent (under appropriate conditions) to a local maximum of the performance measure used by the critic, plus a measure of the temporal difference (TD) error inherent in the function approximation scheme (Konda and Tsitsiklis, 2000, Bhatnagar et al., 2009).

Existing AC algorithms are based on parametric critics that are updated to optimize frequentist fitness criteria. By "frequentist" we mean algorithms that return a point estimate of the value function, rather than a complete posterior distribution computed using Bayes' rule. A Bayesian class of critics based on Gaussian processes (GPs) has been proposed by Engel et al. (2003, 2005), called Gaussian process temporal difference (GPTD). By their Bayesian nature, these algorithms return a full posterior distribution over value functions. Moreover, while these algorithms may be used to learn a parametric representation for the posterior, they are generally capable of searching for value functions in an infinite-dimensional Hilbert space of functions, resulting in a non-parametric posterior.

Both conventional and natural policy gradient and actor-critic methods rely on Monte-Carlo (MC) techniques in estimating the gradient of the performance measure. MC estimation is a frequentist procedure, and as such violates the *likelihood principle* (Berger and Wolpert, 1984).[4] Moreover, although MC estimates are unbiased, they tend to suffer from high variance, or alternatively, require excessive sample sizes (see O'Hagan, 1987 for a discussion). In the case of policy gradient estimation this is exacerbated by the fact that consistent policy improvement requires multiple gradient estimation steps.

4. The likelihood principle states that in a parametric statistical model, all the information about a data sample that is required for inferring the model parameters is contained in the likelihood function of that sample.

In O'Hagan (1991) a Bayesian alternative to MC estimation is proposed.[5] The idea is to model integrals of the form $\int f(x)g(x)dx$ as GPs. This is done by treating the first term f in the integrand as a *random function*, the randomness of which reflects our subjective uncertainty concerning its true identity. This allows us to incorporate our prior knowledge of f into its prior distribution. Observing (possibly noisy) samples of f at a set of points $\{x_1, \ldots, x_M\}$ allows us to employ Bayes' rule to compute a posterior distribution of f conditioned on these samples. This, in turn, induces a posterior distribution over the value of the integral.

In this paper, we first propose a Bayesian framework for policy gradient estimation by modeling the gradient as a GP. Our Bayesian policy gradient (BPG) algorithms use GPs to define a prior distribution over the gradient of the expected return, and compute its posterior conditioned on the observed data. This reduces the number of samples needed to obtain accurate gradient estimates. Moreover, estimates of the natural gradient as well as a measure of the uncertainty in the gradient estimates, namely the gradient covariance, are provided at little extra cost. Additional gains may be attained by learning a transition model of the environment, allowing knowledge transfer between policies. Since our BPG models and algorithms consider complete system trajectories as their basic observable unit, they do not require the dynamics within each trajectory to be of any special form. In particular, it is not necessary for the dynamics to have the Markov property, allowing the resulting algorithms to handle partially observable MDPs, Markov games, and other non-Markovian systems. On the downside, BPG algorithms cannot take advantage of the Markov property when this property is satisfied. To address this issue, we supplement our BPG framework with actor-critic methods and propose an AC algorithm that incorporates GPTD as its critic. However, rather than merely plugging-in our critic into an existing AC algorithm, we show how the posterior moments returned by the GPTD critic allow us to obtain closed-form expressions for the posterior moments of the policy gradient. This is made possible by utilizing the Fisher kernel (Shawe-Taylor and Cristianini, 2004) as our prior covariance kernel for the GPTD state-action *advantage* values. Unlike the BPG methods, the Bayesian actor-critic (BAC) algorithm takes advantage of the Markov property of the system trajectories and uses individual state-action-reward transitions as its basic observable unit. This helps reduce variance in the gradient estimates, resulting in steeper learning curves when compared to BPG and the classic MC approach.

It is important to note that a short version of the two main parts of this paper, *Bayesian policy gradient* and *Bayesian actor-critic*, appeared in Ghavamzadeh and Engel (2006) and Ghavamzadeh and Engel (2007), respectively. This paper extends these conference papers in the following ways:

- We have included a discussion on using Bayesian Quadrature (BQ) for estimating vector-valued integrals to the paper. This is totally relevant to this work because the gradient of a policy (the quantity that we are interested in estimating using BQ) is a vector-valued integral when the size of the policy parameter vector is more than 1, which is usually the case. This also helps to better see the difference between

5. O'Hagan (1991) mentions that this approach may be traced even as far back as Poincaré (1896).

the two models we propose for BPG. In Model 1, we place a vector-valued Gaussian process (GP) over a component of the gradient integrant, while in Model 2, we put a scalar-valued GP over a different component of the gradient integrant.

- We describe the BPG and BAC algorithms in more details and show the details of using online sparsification in these algorithms. Moreover, we show how BPG can be extended to partially observable Markov decision processes (POMDPs) along the same lines that the standard PG algorithms can be used in such problems.

- In comparison to Ghavamzadeh and Engel (2006), we report more details of the experiments and more experimental results, especially in using the posterior variance (covariance) of the gradient to select the step size for updating the policy parameters.

- We include all the proofs in this paper (almost none was reported in the two conference papers), in particular, the proofs of Propositions 3, 4, 5, and 6. These proofs are important and the proof techniques are novel and definitely useful for the community. The importance of these proofs come from the fact that they show how with the right choice of GP prior (the one that uses the family of Fisher information kernels), we are able to use BQ and have a Bayesian estimate of the gradient integral, while initially everything indicates that BQ cannot be used for the estimation of this integral.

- We apply the BAC algorithm to two new domains: "Mountain Car", a 2-dimensional continuous state and 1-dimensional discrete action problem, and "Ship Steering", a 4-dimensional continuous state and 1-dimensional continuous action problem.

2. Reinforcement Learning, Policy Gradient, and Actor-Critic Methods

Reinforcement learning (RL) (Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998) is term describing a class of learning problems in which an agent (or controller) interacts with a dynamic, stochastic, and incompletely known environment (or plant), with the goal of finding an action-selection strategy, or *policy*, to optimize some measure of its long-term performance. This interaction is conventionally modeled as a Markov decision process (MDP) (Puterman, 1994), or if the environmental state is not completely observable, as a partially observable MDP (POMDP) (Astrom, 1965, Smallwood and Sondik, 1973, Kaelbling et al., 1998). In this work we restrict our attention to the discrete-time MDP setting.

Let $\mathcal{P}(\mathcal{X})$, $\mathcal{P}(\mathcal{A})$, and $\mathcal{P}(\mathbb{R})$ denote the set of probability distributions on (Borel) subsets of the sets $\mathcal{X}$, $\mathcal{A}$, and $\mathbb{R}$, respectively. A MDP is a tuple $(\mathcal{X}, \mathcal{A}, q, P, P_0)$ where $\mathcal{X}$ and $\mathcal{A}$ are the state and action spaces; $q(\cdot|x,a) \in \mathcal{P}(\mathbb{R})$ and $P(\cdot|x,a) \in \mathcal{P}(\mathcal{X})$ are the probability distributions over rewards and next states when action a is taken at state x (we assume that q and P are stationary); and $P_0(\cdot) \in \mathcal{P}(\mathcal{X})$ is the probability distribution according to which the initial state is selected. We denote the random variable distributed according to $q(\cdot|x,a)$ as $r(x,a)$ and its mean as $\bar{r}(x,a)$.

In addition, we need to specify the rule according to which the agent selects an action at each possible state. We assume that this rule is *stationary*, i.e., does not depend explicitly on time. A stationary policy $\mu(\cdot|x) \in \mathcal{P}(\mathcal{A})$ is a probability distribution over actions, conditioned on the current state. A MDP controlled by a policy μ induces a

Markov chain over state-action pairs $\boldsymbol{z}_t = (x_t, a_t) \in \mathcal{Z} = \mathcal{X} \times \mathcal{A}$, with a transition probability density $P^\mu(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) = P(x_t|x_{t-1}, a_{t-1})\mu(a_t|x_t)$, and an initial state density $P_0^\mu(\boldsymbol{z}_0) = P_0(x_0)\mu(a_0|x_0)$. We generically denote by $\xi = (\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_T) \in \Xi$, $T \in \{0, 1, \ldots, \infty\}$ a path generated by this Markov chain. Note that Ξ is the set of all possible trajectories that can be generated by the Markov chain induced by the current policy μ. The probability (density) of such a path is given by

$$\Pr(\xi;\mu) = P_0^\mu(\boldsymbol{z}_0)\prod_{t=1}^{T} P^\mu(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) = P_0(x_0)\prod_{t=0}^{T-1}\mu(a_t|x_t)P(x_{t+1}|x_t,a_t). \tag{1}$$

We denote by $R(\xi) = \sum_{t=0}^{T-1}\gamma^t r(x_t,a_t)$ the (possibly discounted, $\gamma \in [0,1]$) *cumulative return* of the path ξ. $R(\xi)$ is a random variable both because the path ξ itself is a random variable, and because, even for a given path, each of the rewards sampled in it may be stochastic. The expected value of $R(\xi)$ for a given path ξ is denoted by $\bar{R}(\xi)$. Finally, we define the *expected return* of policy μ as

$$\eta(\mu) = \mathbf{E}\big[R(\xi)\big] = \int_\Xi \bar{R}(\xi)\Pr(\xi;\mu)d\xi. \tag{2}$$

The t-step state-action occupancy density of policy μ is given by

$$P_t^\mu(\boldsymbol{z}_t) = \int_{\mathcal{Z}^t} d\boldsymbol{z}_0 \ldots d\boldsymbol{z}_{t-1} P_0^\mu(\boldsymbol{z}_0)\prod_{i=1}^{t} P^\mu(\boldsymbol{z}_i|\boldsymbol{z}_{i-1}).$$

It can be shown that under certain regularity conditions (Sutton et al., 2000), the expected return of policy μ may be written in terms of state-action pairs (rather than in terms of trajectories as in Equation 2) as

$$\eta(\mu) = \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\bar{r}(\boldsymbol{z}), \tag{3}$$

where $\pi^\mu(\boldsymbol{z}) = \sum_{t=0}^{\infty}\gamma^t P_t^\mu(\boldsymbol{z})$ is a discounted weighting of state-action pairs encountered while following policy μ. Integrating a out of $\pi^\mu(\boldsymbol{z}) = \pi^\mu(x,a)$ results in the corresponding discounted weighting of states encountered by following policy μ: $\nu^\mu(x) = \int_{\mathcal{A}} da\pi^\mu(x,a)$. Unlike ν^μ and π^μ, $(1-\gamma)\nu^\mu$ and $(1-\gamma)\pi^\mu$ are distributions. They are analogous to the stationary distributions over states and state-action pairs of policy μ in the undiscounted setting, respectively, since as $\gamma \to 1$, they tend to these distributions, if they exist.

Our aim is to find a policy μ^* that maximizes the expected return, i.e., $\mu^* = \arg\max_\mu \eta(\mu)$. A policy μ is assessed according to the expected cumulative rewards associated with states x or state-action pairs $\boldsymbol{z}$. For all states $x \in \mathcal{X}$ and actions $a \in \mathcal{A}$, the action-value function and the value function of policy μ are defined as

$$Q^\mu(\boldsymbol{z}) = \mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t r(\boldsymbol{z}_t)|\boldsymbol{z}_0 = \boldsymbol{z}\right] \qquad \text{and} \qquad V^\mu(x) = \int_{\mathcal{A}} da\mu(a|x)Q^\mu(x,a).$$

In policy gradient (PG) methods, we define a class of smoothly parameterized stochastic policies $\{\mu(\cdot|x;\boldsymbol{\theta}), x \in \mathcal{X}, \boldsymbol{\theta} \in \Theta\}$. We estimate the gradient of the expected return, defined

by Equation 2 (or Equation 3), with respect to the policy parameters $\boldsymbol{\theta}$, from the observed system trajectories. We then improve the policy by adjusting the parameters in the direction of the gradient (e.g., Williams 1992, Marbach 1998, Baxter and Bartlett 2001). Since in this setting a policy μ is represented by its parameters $\boldsymbol{\theta}$, policy dependent functions such as $\eta(\mu)$, $\Pr(\xi;\mu)$, $\pi^\mu(\boldsymbol{z})$, $\nu^\mu(x)$, $V^\mu(x)$, and $Q^\mu(\boldsymbol{z})$ may be written as $\eta(\boldsymbol{\theta})$, $\Pr(\xi;\boldsymbol{\theta})$, $\pi(\boldsymbol{z};\boldsymbol{\theta})$, $\nu(x;\boldsymbol{\theta})$, $V(x;\boldsymbol{\theta})$, and $Q(\boldsymbol{z};\boldsymbol{\theta})$, respectively. We assume

Assumption 1 (Differentiability) *For any state-action pair (x,a) and any policy parameter $\boldsymbol{\theta}\in\Theta$, the policy $\mu(a|x;\boldsymbol{\theta})$ is continuously differentiable in the parameters $\boldsymbol{\theta}$.*

The *score function* or *likelihood ratio* method has become the most prominent technique for gradient estimation from simulation. It has been first proposed in the 1960's (Aleksandrov et al., 1968, Rubinstein, 1969) for computing performance gradients in i.i.d. (independently and identically distributed) processes, and was then extended to *regenerative* processes including MDPs by Glynn (1986, 1990), Reiman and Weiss (1986, 1989), Glynn and L'Ecuyer (1995), and to episodic MDPs by Williams (1992). This method estimates the gradient of the expected return with respect to the policy parameters $\boldsymbol{\theta}$, defined by Equation 2, using the following equation:[6]

$$\nabla\eta(\boldsymbol{\theta}) = \int \bar{R}(\xi)\frac{\nabla\Pr(\xi;\boldsymbol{\theta})}{\Pr(\xi;\boldsymbol{\theta})}\Pr(\xi;\boldsymbol{\theta})d\xi. \tag{4}$$

In Equation 4, the quantity $\frac{\nabla\Pr(\xi;\boldsymbol{\theta})}{\Pr(\xi;\boldsymbol{\theta})} = \nabla\log\Pr(\xi;\boldsymbol{\theta})$ is called the (Fisher) score function or likelihood ratio. Since the initial-state distribution P_0 and the state-transition distribution P are independent of the policy parameters $\boldsymbol{\theta}$, we may write the score function for a path ξ using Equation 1 as[7]

$$\boldsymbol{u}(\xi;\boldsymbol{\theta}) = \nabla\log\Pr(\xi;\boldsymbol{\theta}) = \frac{\nabla\Pr(\xi;\boldsymbol{\theta})}{\Pr(\xi;\boldsymbol{\theta})} = \sum_{t=0}^{T-1}\frac{\nabla\mu(a_t|x_t;\boldsymbol{\theta})}{\mu(a_t|x_t;\boldsymbol{\theta})} = \sum_{t=0}^{T-1}\nabla\log\mu(a_t|x_t;\boldsymbol{\theta}). \tag{5}$$

Previous work on policy gradient used classical MC to estimate the gradient in Equation 4. These methods generate i.i.d. sample paths $\xi_1,\ldots,\xi_M$ according to $\Pr(\xi;\boldsymbol{\theta})$, and estimate the gradient $\nabla\eta(\boldsymbol{\theta})$ using the MC estimator

$$\widehat{\nabla\eta}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}R(\xi_i)\nabla\log\Pr(\xi_i;\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M}R(\xi_i)\sum_{t=0}^{T_i-1}\nabla\log\mu(a_{t,i}|x_{t,i};\boldsymbol{\theta}). \tag{6}$$

This is an unbiased estimate, and therefore, by the law of large numbers, $\widehat{\nabla\eta}(\boldsymbol{\theta})\to\nabla\eta(\boldsymbol{\theta})$ as M goes to infinity, with probability one.

The policy gradient theorem (Marbach, 1998, Proposition 1; Sutton et al., 2000, Theorem 1; Konda and Tsitsiklis, 2000, Theorem 1) states that the gradient of the expected return, defined by Equation 3, for parameterized policies satisfying Assumption 1 is given by

$$\nabla\eta(\boldsymbol{\theta}) = \int dxda\;\nu(x;\boldsymbol{\theta})\nabla\mu(a|x;\boldsymbol{\theta})Q(x,a;\boldsymbol{\theta}). \tag{7}$$

6. Throughout the paper, we use the notation ∇ to denote $\nabla_{\boldsymbol{\theta}}$ – the gradient w.r.t. the policy parameters.
7. To simplify notation, we omit $\boldsymbol{u}$'s dependence on the policy parameters $\boldsymbol{\theta}$, and denote $\boldsymbol{u}(\xi;\boldsymbol{\theta})$ as $\boldsymbol{u}(\xi)$ in the sequel.

Observe that if $b:\mathcal{X}\to\mathbb{R}$ is an arbitrary function of x (also called a *baseline*), then

$$\int_{\mathcal{Z}} dxda\;\nu(x;\boldsymbol{\theta})\nabla\mu(a|x;\boldsymbol{\theta})b(x) = \int_{\mathcal{X}} dx\;\nu(x;\boldsymbol{\theta})b(x)\nabla\Big(\int_{\mathcal{A}} da\mu(a|x;\boldsymbol{\theta})\Big) = \int_{\mathcal{X}} dx\;\nu(x;\boldsymbol{\theta})b(x)\nabla(1) = 0,$$

and thus, for any baseline $b(x)$, the gradient of the expected return can be written as

$$\nabla\eta(\boldsymbol{\theta}) = \int dxda\;\nu(x;\boldsymbol{\theta})\nabla\mu(a|x;\boldsymbol{\theta})\big(Q(x,a;\boldsymbol{\theta})+b(x)\big) = \int_{\mathcal{Z}} d\boldsymbol{z}\pi(\boldsymbol{z};\boldsymbol{\theta})\nabla\log\mu(a|x;\boldsymbol{\theta})\big(Q(\boldsymbol{z};\boldsymbol{\theta})+b(x)\big). \tag{8}$$

The baseline may be chosen in such a way so as to minimize the variance of the gradient estimates (Greensmith et al., 2004).

Now consider the actor-critic (AC) framework in which the action-value function for a fixed policy μ, Q^μ, is approximated by a learned function approximator. If the approximation is sufficiently good, we may hope to use it in place of Q^μ in Equations 7 and 8, and still point roughly in the direction of the true gradient. Sutton et al. (2000) and Konda and Tsitsiklis (2000) showed that if the approximation $\hat{Q}^\mu(\cdot;\boldsymbol{w})$ with parameter $\boldsymbol{w}$ is *compatible*, i.e., $\nabla_{\boldsymbol{w}}\hat{Q}^\mu(x,a;\boldsymbol{w}) = \nabla\log\mu(a|x;\boldsymbol{\theta})$, and if it minimizes the mean squared error

$$\mathcal{E}^\mu(\boldsymbol{w}) = \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\big[Q^\mu(\boldsymbol{z}) - \hat{Q}^\mu(\boldsymbol{z};\boldsymbol{w})\big]^2 \tag{9}$$

for parameter value $\boldsymbol{w}^*$, then we may replace Q^μ with $\hat{Q}^\mu(\cdot;\boldsymbol{w}^*)$ in Equations 7 and 8. The second condition means that $\hat{Q}^\mu(\cdot;\boldsymbol{w}^*)$ is the projection of Q^μ onto the space $\{\hat{Q}^\mu(\cdot;\boldsymbol{w})|\boldsymbol{w}\in\mathbb{R}^n\}$, with respect to a ℓ_2-norm weighted by π^μ.

An approximation for the action-value function, in terms of a linear combination of basis functions, may be written as $\hat{Q}^\mu(\boldsymbol{z};\boldsymbol{w}) = \boldsymbol{w}^\top\boldsymbol{\psi}(\boldsymbol{z})$. This approximation is compatible if the $\boldsymbol{\psi}$'s are compatible with the policy, i.e., $\boldsymbol{\psi}(\boldsymbol{z};\boldsymbol{\theta}) = \nabla\log\mu(a|x;\boldsymbol{\theta})$. Note that compatibility is well defined under Assumption 1. Let $\mathcal{E}^\mu(\boldsymbol{w})$ denote the mean squared error

$$\mathcal{E}^\mu(\boldsymbol{w}) = \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\big[Q^\mu(\boldsymbol{z}) - \boldsymbol{w}^\top\boldsymbol{\psi}(\boldsymbol{z}) - b(x)\big]^2 \tag{10}$$

of our compatible linear approximation $\boldsymbol{w}^\top\boldsymbol{\psi}(\boldsymbol{z})$ and an arbitrary baseline $b(x)$. Let $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}}\mathcal{E}^\mu(\boldsymbol{w})$ denote the optimal parameter. It can be shown that the value of $\boldsymbol{w}^*$ does not depend on the baseline $b(x)$. As a result, the mean squared-error problems of Equations 9 and 10 have the same solutions (see e.g., Bhatnagar et al. 2007, 2009). It can also be shown that if the parameter $\boldsymbol{w}$ is set to be equal to $\boldsymbol{w}^*$, then the resulting mean squared error $\mathcal{E}^\mu(\boldsymbol{w}^*)$, now treated as a function of the baseline $b(x)$, is further minimized by setting $b(x) = V^\mu(x)$ (Bhatnagar et al., 2007, 2009). In other words, the variance in the action-value function estimator is minimized if the baseline is chosen to be the value function itself.

A convenient and rather flexible choice for a space of policies that ensures compatibility between the policy and the action-value representation is a parametric exponential family

$$\mu(a|x;\boldsymbol{\theta}) = \frac{1}{Z_{\boldsymbol{\theta}}(x)}\exp\Big(\boldsymbol{\theta}^\top\boldsymbol{\phi}(x,a)\Big),$$

where $Z_{\boldsymbol{\theta}}(x) = \int_{\mathcal{A}} da \exp\big(\boldsymbol{\theta}^\top \boldsymbol{\phi}(x,a)\big)$ is a normalizing factor, referred to as the *partition function*. It is easy to show that $\boldsymbol{\psi}(\boldsymbol{z}) = \boldsymbol{\phi}(\boldsymbol{z}) - \mathbf{E}_{a|x}[\boldsymbol{\phi}(\boldsymbol{z})]$, where $\mathbf{E}_{a|x}[\cdot] = \int_{\mathcal{A}} da \mu(a|x;\boldsymbol{\theta})[\cdot]$, and as a result, $\hat{Q}^\mu(\boldsymbol{z};\boldsymbol{w}^*) = \boldsymbol{w}^{*\top}\big(\boldsymbol{\phi}(\boldsymbol{z}) - \mathbf{E}_{a|x}[\boldsymbol{\phi}(\boldsymbol{z})]\big) + b(x)$ is a compatible action-value function for this family of policies. Note that $\mathbf{E}_{a|x}[\hat{Q}(\boldsymbol{z};\boldsymbol{w}^*)] = b(x)$, since $\mathbf{E}_{a|x}\big[\boldsymbol{\phi}(\boldsymbol{z}) - \mathbf{E}_{a|x}[\boldsymbol{\phi}(\boldsymbol{z})]\big] = 0$. This means that if $\hat{Q}^\mu(\boldsymbol{z};\boldsymbol{w}^*)$ approximates $Q^\mu(\boldsymbol{z})$, then $b(x)$ must approximate the value function $V^\mu(x)$. The term $\hat{A}^\mu(\boldsymbol{z};\boldsymbol{w}^*) = \hat{Q}^\mu(\boldsymbol{z};\boldsymbol{w}^*) - b(x) = \boldsymbol{w}^{*\top}\big(\boldsymbol{\phi}(\boldsymbol{z}) - \mathbf{E}_{a|x}[\boldsymbol{\phi}(\boldsymbol{z})]\big)$ approximates the *advantage function* $A^\mu(\boldsymbol{z}) = Q^\mu(\boldsymbol{z}) - V^\mu(x)$ (Baird, 1993).

3. Bayesian Quadrature

Bayesian quadrature (BQ) (O'Hagan, 1991) is, as its name suggests, a Bayesian method for evaluating an integral using samples of its integrand. We consider the problem of evaluating the integral

$$\rho = \int f(x) g(x) dx. \tag{11}$$

If $g(x)$ is a probability density function, i.e., $g(x) = p(x)$, this becomes the problem of evaluating the expected value of $f(x)$. A well known frequentist approach to evaluating such expectations is the Monte-Carlo (MC) method. For MC estimation of such expectations, it is typically required that samples $x_1, x_2, \ldots, x_M$ are drawn from $p(x)$.[8] The integral in Equation 11 is then estimated as

$$\hat{\rho}_{MC} = \frac{1}{M} \sum_{i=1}^{M} f(x_i). \tag{12}$$

It is easy to show that $\hat{\rho}_{MC}$ is an unbiased estimate of ρ, with variance that diminishes to zero as $M \to \infty$. However, as O'Hagan (1987) points out, MC estimation is fundamentally unsound, as it violates the likelihood principle, and moreover, does not make full use of the data at hand. The alternative proposed in O'Hagan (1991) is based on the following reasoning: In the Bayesian approach, $f(\cdot)$ is random simply because it is unknown. We are therefore uncertain about the value of $f(x)$ until we actually evaluate it. In fact, even then, our uncertainty is not always completely removed, since measured samples of $f(x)$ may be corrupted by noise. Modeling f as a Gaussian process (GP) means that our uncertainty is completely accounted for by specifying a Normal prior distribution over functions. This prior distribution is specified by its mean and covariance, and is denoted by $f(\cdot) \sim \mathcal{N}\big(\bar{f}(\cdot), k(\cdot,\cdot)\big)$. This is shorthand for the statement that f is a GP with prior mean and covariance

$$\mathbf{E}\big[f(x)\big] = \bar{f}(x) \quad \text{and} \quad \mathbf{Cov}\big[f(x), f(x')\big] = k(x,x'), \quad \forall x, x' \in \mathcal{X}, \tag{13}$$

respectively. The choice of kernel function k allows us to incorporate prior knowledge on the smoothness properties of the integrand into the estimation procedure. When we are provided with a set of samples $\mathcal{D}_M = \big\{\big(x_i, y(x_i)\big)\big\}_{i=1}^M$, where $y(x_i)$ is a (possibly noisy) sample of $f(x_i)$, we apply Bayes' rule to condition the prior on these sampled values. If the measurement noise is normally distributed, the result is a Normal posterior distribution of

8. If samples are drawn from some other distribution, importance sampling variants of MC may be used.

$f|\mathcal{D}_M$. The expressions for the posterior mean and covariance are standard:

$$\begin{aligned} \mathbf{E}\big[f(x)|\mathcal{D}_M\big] &= \bar{f}(x) + \boldsymbol{k}(x)^\top \boldsymbol{C}(\boldsymbol{y} - \bar{\boldsymbol{f}}), \\ \mathbf{Cov}\big[f(x), f(x')|\mathcal{D}_M\big] &= k(x,x') - \boldsymbol{k}(x)^\top \boldsymbol{C} \boldsymbol{k}(x'). \end{aligned} \tag{14}$$

Here and in the sequel, we make use of the definitions:

$$\begin{aligned} &\bar{\boldsymbol{f}} = \big(\bar{f}(x_1), \ldots, \bar{f}(x_M)\big)^\top, && && \boldsymbol{k}(x) = \big(k(x_1,x), \ldots, k(x_M,x)\big)^\top, \\ &\boldsymbol{y} = \big(y(x_1), \ldots, y(x_M)\big)^\top, && [\boldsymbol{K}]_{i,j} = k(x_i,x_j), && \boldsymbol{C} = (\boldsymbol{K} + \boldsymbol{\Sigma})^{-1}, \end{aligned}$$

where $\boldsymbol{K}$ is the kernel (or Gram) matrix, and $[\boldsymbol{\Sigma}]_{i,j}$ is the measurement noise covariance between the ith and jth samples. It is typically assumed that the measurement noise is i.i.d., in which case $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}$, where σ^2 is the noise variance and $\boldsymbol{I}$ is the (appropriately sized - here $M \times M$) identity matrix.

Since integration is a linear operation, the posterior distribution of the integral in Equation 11 is also Gaussian, and the posterior moments are given by (O'Hagan, 1991)

$$\begin{aligned} \mathbf{E}[\rho|\mathcal{D}_M] &= \int \mathbf{E}\big[f(x)|\mathcal{D}_M\big] g(x) dx, \\ \mathbf{Var}[\rho|\mathcal{D}_M] &= \iint \mathbf{Cov}\big[f(x), f(x')|\mathcal{D}_M\big] g(x) g(x') dx dx'. \end{aligned} \tag{15}$$

Substituting Equation 14 into Equation 15, we obtain

$$\mathbf{E}[\rho|\mathcal{D}_M] = \rho_0 + \boldsymbol{b}^\top \boldsymbol{C}(\boldsymbol{y} - \bar{\boldsymbol{f}}) \quad \text{and} \quad \mathbf{Var}[\rho|\mathcal{D}_M] = b_0 - \boldsymbol{b}^\top \boldsymbol{C} \boldsymbol{b},$$

where we made use of the definitions:

$$\rho_0 = \int \bar{f}(x) g(x) dx\,, \qquad \boldsymbol{b} = \int \boldsymbol{k}(x) g(x) dx\,, \qquad b_0 = \iint k(x,x') g(x) g(x') dx dx'. \tag{16}$$

Note that ρ_0 and b_0 are the prior mean and variance of ρ, respectively.

Rasmussen and Ghahramani (2003) experimentally demonstrated how this approach, when applied to the evaluation of an expectation, can outperform MC estimation by orders of magnitude in terms of the mean-squared error.

In order to prevent the problem from "degenerating into infinite regress", as phrased by O'Hagan (1991),[9] we should choose the functions g, k, and $\bar{f}$ so as to allow us to solve the integrals in Equation 16 analytically. For example, O'Hagan provides the analysis required for the case where the integrands in Equation 16 are products of multivariate Gaussians and polynomials, referred to as Bayes-Hermite quadrature. One of the contributions of our work is in providing analogous analysis for kernel functions that are based on the *Fisher kernel* (Jaakkola and Haussler, 1999, Shawe-Taylor and Cristianini, 2004).

It is important to note that in MC estimation, samples must be drawn from the distribution $p(x) = g(x)$, whereas in the Bayesian approach, samples may be drawn from arbitrary distributions. This affords us with flexibility in the choice of sample points, allowing us, for instance, to actively design the samples $x_1, \ldots, x_M$ so as to maximize information gain.

9. What O'Hagan means by "degenerating into infinite regress" is simply that if we cannot compute the posterior integrals of Equation 16 analytically, then we have started with estimating one integral (Equation 11) and ended up with three (Equation 16), and if we repeat this process, this can go forever and leave us with infinite integrals to evaluate. Therefore, for Bayesian MC to work, it is crucial to be able to analytically calculate the posterior integrals, and this can be achieved through the way we divide the integrant into two parts and the way we select the kernel function.

3.1 Vector-Valued Integrals

O'Hagan (1991) treated the case where the integral to be estimated is a scalar-valued integral. However, in the context of our PG method, it is useful to consider vector-valued integrals, such as the gradient of the expected return with respect to the policy parameters, which we shall study in Section 4. In the BQ framework, an integral of the form in Equation 11 may be vector-valued for one of two possible reasons: either f is a vector-valued GP and g is a scalar-valued function, or f is a scalar-valued GP and g is a vector-valued function. These two possibilities correspond to two very different data-generation models. In the first of these, an n-valued function $f(\cdot) = \big(f_1(\cdot), \ldots, f_n(\cdot)\big)^\top$ is sampled from the GP distribution of f. This distribution may include correlations between different components of f. Hence, in general, to specify the GP prior distribution, one needs to specify not only the covariance kernel of each component j of f, $k_{j,j}(x,x') = \mathbf{Cov}\big[f_j(x), f_j(x')\big]$, but also cross-covariance kernels for pairs of different components, $k_{j,\ell}(x,x') = \mathbf{Cov}\big[f_j(x), f_\ell(x')\big]$. Thus, instead of a single kernel function, we now need to specify a matrix of kernel functions.[10] Similarly, we also need to specify a vector of prior means, consisting of a function for each component: $\bar{f}_j(x) = \mathbf{E}\big[f_j(x)\big]$. The distribution of the measurement noise added to $f(x)$ to produce $y(x)$ may also include correlations, requiring us to specify an array of noise covariance matrices $\mathbf{\Sigma}_{j,\ell}$. As we show below, the GP posterior distribution is also specified in similar terms.

In the second model, a scalar-valued function is sampled from the GP prior distribution, which is specified by a single prior mean function and a single prior covariance-kernel function. Gaussian noise may be added, and the result is then multiplied by each of the components of the n-valued function g to produce the integrand. This model is significantly simpler, both conceptually and in terms of the number of parameters required to specify it. To see how a model of the first kind may arise, consider the following example.

Example 1 *Let $\rho(\boldsymbol{\theta}) = \int f(x;\boldsymbol{\theta})g(x)dx$, where f is a scalar GP, parameterized by a vector of parameters $\boldsymbol{\theta}$. Its prior mean and covariance functions must therefore depend on $\boldsymbol{\theta}$. We denote these dependencies by writing:*

$$\mathbf{E}\big[f(x;\boldsymbol{\theta})\big] = \bar{f}(x;\boldsymbol{\theta}), \quad \mathbf{Cov}\big[f(x;\boldsymbol{\theta}), f(x';\boldsymbol{\theta})\big] = k(x,x';\boldsymbol{\theta}), \quad \forall x, x' \in \mathcal{X}.$$

We choose $\bar{f}(x;\boldsymbol{\theta})$ and $k(x,x';\boldsymbol{\theta})$ so as to be once and twice differentiable in $\boldsymbol{\theta}$, respectively. Suppose now that we are not interested in estimating $\rho(\boldsymbol{\theta})$, but rather in its gradient with respect to the parameters $\boldsymbol{\theta}$: $\nabla_{\boldsymbol{\theta}}\rho(\boldsymbol{\theta}) = \int \nabla_{\boldsymbol{\theta}} f(x;\boldsymbol{\theta})g(x)dx$. It may be easily verified that the mean functions and covariance kernels of the vector-valued GP $\nabla_{\boldsymbol{\theta}} f(x;\boldsymbol{\theta})$ are given by

$$\mathbf{E}\big[\nabla_{\boldsymbol{\theta}} f(x;\boldsymbol{\theta})\big] = \nabla_{\boldsymbol{\theta}}\bar{f}(x;\boldsymbol{\theta}) \quad \text{and} \quad \mathbf{Cov}\big[\partial_{\theta_j} f(x;\boldsymbol{\theta}), \partial_{\theta_\ell} f(x';\boldsymbol{\theta})\big] = \partial_{\theta_j}\partial_{\theta_\ell} k(x,x';\boldsymbol{\theta}),$$

where ∂_{θ_j} denotes the jth component of $\nabla_{\boldsymbol{\theta}}$. □

Propositions 1 and 2 specify the form taken by the mean and covariance functions of the integral GP under the two models discussed above.

10. Note that to satisfy the symmetry property of the covariance operator, we require that $k_{j,\ell}(x,x') = k_{\ell,j}(x',x) = k_{\ell,j}(x,x')$, for all $x, x' \in \mathcal{X}$ and $j, \ell \in \{1,\ldots,n\}$.

Proposition 1 (Vector-valued GP) *Let f be an n-valued GP with mean functions $\bar{f}_j(x) = \mathbf{E}\big[f_j(x)\big]$ and covariance functions $k_{j,\ell}(x,x') = \mathbf{Cov}\big[f_j(x), f_\ell(x')\big]$, $\forall j,\ell \in \{1,\ldots,n\}$, and let g be a scalar-valued function. Then, the mean and covariance of ρ defined by Equation 11 are of the following form:*

$$\mathbf{E}\big[\rho_j\big] = \int \bar{f}_j(x)g(x)dx, \quad \mathbf{Cov}\big[\rho_j, \rho_\ell\big] = \iint k_{j,\ell}(x,x')g(x)g(x')dxdx', \quad \forall j,\ell \in \{1,\ldots,n\}.$$

Proposition 2 (Scalar-valued GP) *Let f be a scalar-valued GP with mean function $\bar{f}(x) = \mathbf{E}\big[f(x)\big]$ and covariance function $k(x,x') = \mathbf{Cov}\big[f(x), f(x')\big]$, and let g be an n-valued function. Then, the mean and covariance of ρ defined by Equation 11 are of the following form:*

$$\mathbf{E}\big[\rho_j\big] = \int \bar{f}(x)g_j(x)dx, \quad \mathbf{Cov}\big[\rho_j, \rho_\ell\big] = \iint k(x,x')g_j(x)g_\ell(x')dxdx', \quad \forall j,\ell \in \{1,\ldots,n\}.$$

The proofs of these two propositions follow straightforwardly from the definition of the covariance operator in terms of expectations, and the order-exchangeability of GP expectations and integration with respect to x.

To wrap things up, we need to describe the form taken by the posterior moments of f in the vector-valued GP case. Using the standard Gaussian conditioning formulas, it is straightforward to show that

$$\begin{aligned}
\mathbf{E}\big[f_j(x)|\mathcal{D}_M\big] &= \bar{f}_j(x) + \sum_{m,m'} \boldsymbol{k}_{j,m}(x)^\top \boldsymbol{C}_{m,m'}(\boldsymbol{y}_{m'} - \bar{\boldsymbol{f}}_{m'}),\\
\mathbf{Cov}\big[f_j(x), f_\ell(x')|\mathcal{D}_M\big] &= k_{j,\ell}(x,x') - \sum_{m,m'} \boldsymbol{k}_{j,m}(x)^\top \boldsymbol{C}_{m,m'}\boldsymbol{k}_{m',\ell}(x'),
\end{aligned} \tag{17}$$

where

$$\begin{aligned}
&\bar{\boldsymbol{f}}_j = \big(\bar{f}_j(x_1), \ldots, \bar{f}_j(x_M)\big)^\top, && \boldsymbol{k}_{j,\ell}(x) = \big(k_{j,\ell}(x_1,x), \ldots, k_{j,\ell}(x_M,x)\big)^\top,\\
&\boldsymbol{y}_\ell = \big(y_\ell(x_1), \ldots, y_\ell(x_M)\big)^\top, && [\boldsymbol{K}_{j,\ell}]_{i,i'} = k_{j,\ell}(x_i, x_{i'}),
\end{aligned}$$

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{1,1} & \ldots & \boldsymbol{K}_{1,n} \\ \vdots & & \vdots \\ \boldsymbol{K}_{n,1} & \ldots & \boldsymbol{K}_{n,n} \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{1,1} & \ldots & \boldsymbol{\Sigma}_{1,n} \\ \vdots & & \vdots \\ \boldsymbol{\Sigma}_{n,1} & \ldots & \boldsymbol{\Sigma}_{n,n} \end{bmatrix}, \quad \boldsymbol{C} = (\boldsymbol{K} + \boldsymbol{\Sigma})^{-1},$$

and $\boldsymbol{C}_{j,\ell}$ is the (j,ℓ)th $M \times M$ block of $\boldsymbol{C}$. The posterior moments of f, given in Equation 17, may now be substituted into the expressions for the moments of the integral ρ, given in Proposition 1, to produce the posterior moments of ρ in the vector-valued GP case.

Clearly, models of the first type (vector-valued GP) are potentially richer and more complex than models of the second type (scalar-valued GP), as the latter only requires us to define a single prior mean function, a single prior covariance kernel function, and a single noise covariance matrix; while the former requires us to define n prior mean functions, and $n(n+1)/2$ prior covariance kernel functions and noise covariance matrices. One way to simplify the first type of models is to define a single prior mean function $\bar{f}$, a single prior covariance kernel function k, and to postulate that $\bar{f}_j(x) = \bar{f}(x)$, $k_{j,\ell}(x,x') = \delta_{j,\ell}\, k(x,x')$, and

$\boldsymbol{\Sigma}_{j,\ell} = \delta_{j,\ell}\boldsymbol{\Sigma}$, $\forall j, \ell \in \{1,\dots,n\}$, where δ denotes the Kronecker delta function. Applying these simplifying assumptions to the expressions for the posterior moments (Equation 17) results in a complete decoupling between the posterior moments for the different components of f, and consequently a decoupling between the components of the integral ρ as well, since Equation 17 becomes

$$\begin{aligned}\mathbf{E}\big[f_j(x)|\mathcal{D}_M\big] &= \bar{f}_j(x) + \boldsymbol{k}_{j,j}(x)^\top \boldsymbol{C}_{j,j}(\boldsymbol{y}_j - \bar{\boldsymbol{f}}_j),\\ \mathbf{Cov}\big[f_j(x), f_\ell(x')|\mathcal{D}_M\big] &= \delta_{j,\ell}\left(k_{j,j}(x,x') - \boldsymbol{k}_{j,j}(x)^\top \boldsymbol{C}_{j,\ell}\boldsymbol{k}_{\ell,\ell}(x')\right),\end{aligned} \tag{18}$$

where $\boldsymbol{C}_{j,\ell} = \delta_{j,\ell}\left(\boldsymbol{K}_{j,\ell} + \boldsymbol{\Sigma}_{j,\ell}\right)^{-1}$. Note that all the terms in Equation 18, except $\boldsymbol{y}_j$, do not depend on the indices j and ℓ. In other words, these simplifying assumptions amount to assuming that ρ is a vector of n independent integrals, each of which may be estimated individually as

$$\begin{aligned}\mathbf{E}\big[f_j(x)|\mathcal{D}_M\big] &= \bar{f}(x) + \boldsymbol{k}(x)^\top \boldsymbol{C}(\boldsymbol{y}_j - \bar{\boldsymbol{f}}),\\ \mathbf{Cov}\big[f_j(x), f_\ell(x')|\mathcal{D}_M\big] &= \delta_{j,\ell}\left(k(x,x') - \boldsymbol{k}(x)^\top \boldsymbol{C}\boldsymbol{k}(x')\right),\end{aligned}$$

where $\boldsymbol{C} = (\boldsymbol{K} + \boldsymbol{\Sigma})^{-1}$. It should, however, be kept in mind that ignoring correlations between the components of f, when such correlations exist, may result in suboptimal use of the available data (see Rasmussen and Williams, 2006, Chapter 9 for references on GP regression with multiple outputs).

4. Bayesian Policy Gradient

In this section, we use vector-valued Bayesian quadrature to estimate the gradient of the expected return with respect to the policy parameters, allowing us to propose new *Bayesian policy gradient* (BPG) algorithms. In the frequentist approach to policy gradient, the performance measure used is $\eta(\boldsymbol{\theta}) = \int \bar{R}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi$ (Equation 2). In order to serve as a useful performance measure, it has to be a deterministic function of the policy parameters $\boldsymbol{\theta}$. This is achieved by averaging the cumulative return $R(\xi)$ over all possible paths ξ and all possible returns accumulated in each path. In the Bayesian approach we have an additional source of randomness, namely, our subjective Bayesian uncertainty concerning the process generating the cumulative return. Let us denote

$$\eta_B(\boldsymbol{\theta}) = \int \bar{R}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi, \tag{19}$$

where $\eta_B(\boldsymbol{\theta})$ is a random variable because of the Bayesian uncertainty. Under the quadratic loss, the optimal Bayesian performance measure is the posterior expected value of $\eta_B(\boldsymbol{\theta})$, $\mathbf{E}\big[\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big]$. However, since we are interested in optimizing the performance rather than evaluating it,[11] we would rather evaluate the posterior distribution of the *gradient* of $\eta_B(\boldsymbol{\theta})$ with respect to the policy parameters $\boldsymbol{\theta}$. The posterior mean of the gradient is [12]

$$\nabla\mathbf{E}\big[\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \mathbf{E}\left[\int R(\xi)\frac{\nabla\Pr(\xi;\boldsymbol{\theta})}{\Pr(\xi;\boldsymbol{\theta})}\Pr(\xi;\boldsymbol{\theta})d\xi\Big|\mathcal{D}_M\right]. \tag{20}$$

11. Although evaluating the posterior distribution of performance is an interesting question in its own right.
12. We may interchange the order of the gradient and the expectation operators for the mean, $\nabla\mathbf{E}\big[\eta_B(\boldsymbol{\theta})\big] = \mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})\big]$, but the same is not true for the variance, namely, $\nabla\mathbf{Var}\big[\eta_B(\boldsymbol{\theta})\big] \neq \mathbf{Cov}\big[\nabla\eta_B(\boldsymbol{\theta})\big]$.

Consequently, in BPG we cast the problem of estimating the gradient of the expected return (Equation 20) in the form of Equation 11. As described in Section 3, we need to partition the integrand into two parts, $f(\xi;\boldsymbol{\theta})$ and $g(\xi;\boldsymbol{\theta})$. We will model f as a GP and assume that g is a function known to us. We will then proceed by calculating the posterior moments of the gradient $\nabla\eta_B(\boldsymbol{\theta})$ conditioned on the observed data. Because in general, $R(\xi)$ cannot be known exactly, even for a given ξ (due to the stochasticity of the rewards), $R(\xi)$ should always belong to the GP part of the model, i.e., $f(\xi;\boldsymbol{\theta})$. Interestingly, in certain cases it is sufficient to know the Fisher information matrix corresponding to $\Pr(\xi;\boldsymbol{\theta})$, rather than having exact knowledge of $\Pr(\xi;\boldsymbol{\theta})$ itself. We make use of this fact in the sequel. In the next two sections, we investigate two different ways of partitioning the integrand in Equation 20, resulting in two distinct Bayesian policy gradient models.

4.1 Model 1 – Vector-Valued GP

In our first model, we define g and f as follows:

$$g(\xi;\boldsymbol{\theta}) = \Pr(\xi;\boldsymbol{\theta}) \quad , \quad f(\xi;\boldsymbol{\theta}) = \bar{R}(\xi)\frac{\nabla\Pr(\xi;\boldsymbol{\theta})}{\Pr(\xi;\boldsymbol{\theta})} = \bar{R}(\xi)\nabla\log\Pr(\xi;\boldsymbol{\theta}).$$

We place a vector-valued GP prior over $f(\xi;\boldsymbol{\theta})$ which induces a GP prior over the corresponding noisy measurement $y(\xi;\boldsymbol{\theta}) = R(\xi)\nabla\log\Pr(\xi;\boldsymbol{\theta})$. We adopt the simplifying assumptions discussed at the end of Section 3.1: We assume that each component of $f(\xi;\boldsymbol{\theta})$ may be evaluated independently of all other components, and use the same kernel function and noise covariance for all components of $f(\xi;\boldsymbol{\theta})$. We therefore omit the component index j from $\boldsymbol{K}_{j,j}$, $\boldsymbol{\Sigma}_{j,j}$ and $\boldsymbol{C}_{j,j}$, denoting them simply as $\boldsymbol{K}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{C}$, respectively. Hence, for the jth component of f and y we have, a priori

$$\begin{aligned}\boldsymbol{f}_j &= \big(f_j(\xi_1;\boldsymbol{\theta}),\dots,f_j(\xi_M;\boldsymbol{\theta})\big)^\top \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{K}),\\ \boldsymbol{y}_j &= \big(y_j(\xi_1;\boldsymbol{\theta}),\dots,y_j(\xi_M;\boldsymbol{\theta})\big)^\top \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{K}+\boldsymbol{\Sigma}).\end{aligned}$$

In this vector-valued GP model, the posterior mean and covariance of $\nabla\eta_B(\boldsymbol{\theta})$ are

$$\mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \boldsymbol{Y}\boldsymbol{C}\boldsymbol{b} \qquad \text{and} \qquad \mathbf{Cov}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \Big(b_0 - \boldsymbol{b}^\top\boldsymbol{C}\boldsymbol{b}\Big)\boldsymbol{I}, \tag{21}$$

respectively, where

$$\boldsymbol{Y} = \begin{bmatrix}\boldsymbol{y}_1^\top\\ \vdots\\ \boldsymbol{y}_n^\top\end{bmatrix}, \quad \boldsymbol{b} = \int \boldsymbol{k}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi\,, \quad \text{and} \quad b_0 = \iint k(\xi,\xi')\Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi'. \tag{22}$$

Our choice of kernel, which allows us to derive closed-form expressions for $\boldsymbol{b}$ and b_0, and as a result for the posterior moments of the gradient, is the quadratic Fisher kernel (Jaakkola and Haussler, 1999, Shawe-Taylor and Cristianini, 2004)

$$k(\xi_i,\xi_j) = \Big(1 + \boldsymbol{u}(\xi_i)^\top\boldsymbol{G}(\boldsymbol{\theta})^{-1}\boldsymbol{u}(\xi_j)\Big)^2, \tag{23}$$

where $\boldsymbol{u}(\xi) = \nabla \log \Pr(\xi;\boldsymbol{\theta})$ is the Fisher score function of the path ξ defined by Equation 5, and $\boldsymbol{G}(\boldsymbol{\theta})$ is the corresponding Fisher information matrix defined as[13]

$$\boldsymbol{G}(\boldsymbol{\theta}) = \mathbf{E}\big[\boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top\big] = \int \boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top \Pr(\xi;\boldsymbol{\theta}) d\xi. \tag{24}$$

Proposition 3 *Using the quadratic Fisher kernel from Equation 23, the integrals $\boldsymbol{b}$ and b_0 in Equation 22 have the following closed form expressions*

$$(\boldsymbol{b})_i = 1 + \boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i) \qquad \textit{and} \qquad b_0 = 1 + n.$$

Proof See Appendix A. ■

4.2 Model 2 – Scalar-Valued GP

In our second model, we define g and f as follows:

$$g(\xi;\boldsymbol{\theta}) = \nabla \Pr(\xi;\boldsymbol{\theta}) \quad , \quad f(\xi) = \bar{R}(\xi).$$

Now g is a vector-valued function, while f is a scalar valued GP representing the expected return of the path given as its argument. The noisy measurement corresponding to $f(\xi_i)$ is $y(\xi_i) = R(\xi_i)$, namely, the *actual* return accrued while following the path ξ_i. In this model, the posterior mean and covariance of the gradient $\nabla\eta_B(\boldsymbol{\theta})$ are

$$\mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \boldsymbol{BCy} \qquad \text{and} \qquad \mathbf{Cov}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \boldsymbol{B}_0 - \boldsymbol{BCB}^\top, \tag{25}$$

respectively, where

$$\boldsymbol{y} = \big(R(\xi_1),\ldots,R(\xi_M)\big)^\top, \qquad \boldsymbol{B} = \int \nabla \Pr(\xi;\boldsymbol{\theta})\boldsymbol{k}(\xi)^\top d\xi\,,$$
$$\boldsymbol{B}_0 = \iint k(\xi,\xi')\nabla \Pr(\xi;\boldsymbol{\theta})\nabla \Pr(\xi';\boldsymbol{\theta})^\top d\xi d\xi'. \tag{26}$$

Here, our choice of kernel function, which again allows us to derive closed-form expressions for $\boldsymbol{B}$ and $\boldsymbol{B}_0$, is the Fisher kernel (Jaakkola and Haussler, 1999, Shawe-Taylor and Cristianini, 2004)

$$k(\xi,\xi') = \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi'). \tag{27}$$

Proposition 4 *Using the Fisher kernel from Equation 27, the integrals $\boldsymbol{B}$ and $\boldsymbol{B}_0$ in Equation 26 have the following closed-form expressions*

$$\boldsymbol{B} = \boldsymbol{U} \qquad \textit{and} \qquad \boldsymbol{B}_0 = \boldsymbol{G},$$

where $\boldsymbol{U} = \big[\boldsymbol{u}(\xi_1),\ldots,\boldsymbol{u}(\xi_M)\big]$.

13. To simplify notation, we omit $\boldsymbol{G}$'s dependence on the policy parameters $\boldsymbol{\theta}$, and denote $\boldsymbol{G}(\boldsymbol{\theta})$ as $\boldsymbol{G}$ in the sequel.

	Model 1	**Model 2**
Deter. factor (g)	$g(\xi;\boldsymbol{\theta}) = \Pr(\xi;\boldsymbol{\theta})$	$g(\xi;\boldsymbol{\theta}) = \nabla \Pr(\xi;\boldsymbol{\theta})$
GP factor (f)	$f(\xi;\boldsymbol{\theta}) = \bar{R}(\xi)\nabla \log \Pr(\xi;\boldsymbol{\theta})$	$f(\xi) = \bar{R}(\xi)$
Measurement (y)	$y(\xi;\boldsymbol{\theta}) = R(\xi)\nabla \log \Pr(\xi;\boldsymbol{\theta})$	$y(\xi) = R(\xi)$
Prior mean of f	$\mathbf{E}\big[f_j(\xi;\boldsymbol{\theta})\big] = 0$	$\mathbf{E}\big[f(\xi)\big] = 0$
Prior Cov. of f	$\mathbf{Cov}\big[f_j(\xi;\boldsymbol{\theta}), f_\ell(\xi';\boldsymbol{\theta})\big] = \delta_{j,\ell}\, k(\xi,\xi')$	$\mathbf{Cov}\big[f(\xi), f(\xi')\big] = k(\xi,\xi')$
Kernel function	$k(\xi,\xi') = \big(1 + \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi')\big)^2$	$k(\xi,\xi') = \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi')$
$\mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})\|\mathcal{D}_M\big] =$	$\boldsymbol{YCb}$	$\boldsymbol{BCy}$
$\mathbf{Cov}\big[\nabla\eta_B(\boldsymbol{\theta})\|\mathcal{D}_M\big] =$	$(b_0 - \boldsymbol{b}^\top\boldsymbol{Cb})\boldsymbol{I}$	$\boldsymbol{B}_0 - \boldsymbol{BCB}^\top$
$\boldsymbol{b}$ or $\boldsymbol{B}$	$(\boldsymbol{b})_i = 1 + \boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)$	$\boldsymbol{B} = \boldsymbol{U}$
b_0 or $\boldsymbol{B}_0$	$b_0 = 1 + n$	$\boldsymbol{B}_0 = \boldsymbol{G}$

Table 1: Summary of the Bayesian policy gradient Models 1 and 2.

Proof See Appendix B. ■

Table 1 summarizes the two BPG models presented in Sections 4.1 and 4.2. Our choice of Fisher-type kernels was motivated by the notion that a good representation should depend on the process generating the data (see Jaakkola and Haussler, 1999, Shawe-Taylor and Cristianini, 2004, for a thorough discussion). Our particular selection of linear and quadratic Fisher kernels were guided by the desideratum that the posterior moments of the gradient be analytically tractable as discussed in Section 3.

As described above, in either model, we are restricted in the choice of kernel (quadratic Fisher kernel in Model 1 and Fisher kernel in Model 2) in order to be able to derive closed-form expressions for the posterior mean and covariance of the gradient integral. The loss due to this restriction depends on the problem at hand and is hard to quantify. This loss is exactly the loss of selecting an inappropriate prior in any Bayesian algorithm or, more generally, of choosing a wrong representation (function space) in a machine learning algorithm (referred to as *approximation error* in approximation theory). However, the experimental results of Section 6 indicate that this restriction did not cause a significant error (especially for Model 1) in our gradient estimates, as those estimated by BPG were more accurate than the ones estimated by the MC-based method, given the same number of samples.

4.3 A Bayesian Policy Gradient Evaluation Algorithm

We can now use our two BPG models to define corresponding algorithms for evaluating the gradient of the expected return with respect to the policy parameters. Pseudo-code for these algorithms is shown in Algorithm 1. The generic algorithm (for either model) takes a set of policy parameters $\boldsymbol{\theta}$ and a sample size M as input, and returns an estimate of the posterior moments of the gradient of the expected return with respect to the policy parameters. This algorithm generates M sample paths to evaluate the gradient. For each path ξ_i, the algorithm first computes its score function $\boldsymbol{u}(\xi_i)$ (Line 6). The score function is needed for computing the kernel function k, the measurement $\boldsymbol{y}$ in Model 1, and $\boldsymbol{b}$ or

$\boldsymbol{B}$. The algorithm then computes the return R and the measurement $y(\xi_i)$ for the observed path ξ_i (Lines 7 and 9), and updates the kernel matrix $\boldsymbol{K}$ (Line 8) using

$$\boldsymbol{K} := \begin{bmatrix} \boldsymbol{K} & \boldsymbol{k}(\xi_i) \\ \boldsymbol{k}^\top(\xi_i) & k(\xi_i,\xi_i) \end{bmatrix}. \tag{28}$$

Finally, the algorithm adds the measurement error $\boldsymbol{\Sigma}$ to the covariance matrix $\boldsymbol{K}$ (Line 12) and computes the posterior moments of the policy gradient (Line 14). $\boldsymbol{B}(:,i)$ on Line 10 denotes the ith column of the matrix $\boldsymbol{B}$.

Algorithm 1 A Bayesian Policy Gradient Evaluation Algorithm

1: **BPG_Eval**$(\boldsymbol{\theta}, M)$
   - sample size $M > 0$
   - a vector of policy parameters $\boldsymbol{\theta} \in \mathbb{R}^n$

2: Set $\boldsymbol{G} = \boldsymbol{G}(\boldsymbol{\theta})$, $\mathcal{D} = \emptyset$
3: **for** $i = 1$ to M **do**
4: Sample a path ξ_i using the policy $\mu(\boldsymbol{\theta})$
5: $\mathcal{D} := \mathcal{D} \bigcup \{\xi_i\}$
6: $\boldsymbol{u}(\xi_i) = \sum_{t=0}^{T_i-1} \nabla \log \mu(a_{t,i}|x_{t,i};\boldsymbol{\theta})$
7: $R(\xi_i) = \sum_{t=0}^{T_i-1} r(x_{t,i}, a_{t,i})$
8: Update $\boldsymbol{K}$ using Equation 28
9: $y(\xi_i) = R(\xi_i)\boldsymbol{u}(\xi_i)$ (Model 1) or $y(\xi_i) = R(\xi_i)$ (Model 2)
10: $(\boldsymbol{b})_i = 1 + \boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)$ (Model 1) or $\boldsymbol{B}(:,i) = \boldsymbol{u}(\xi_i)$ (Model 2)
11: **end for**
12: $\boldsymbol{C} = (\boldsymbol{K} + \boldsymbol{\Sigma})^{-1}$
13: $b_0 = 1 + n$ (Model 1) or $\boldsymbol{B}_0 = \boldsymbol{G}$ (Model 2)
14: Compute the posterior mean and covariance of the policy gradient
$\mathbf{E}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}\big) = \boldsymbol{YCb}$, $\mathbf{Cov}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}\big) = (b_0 - \boldsymbol{b}^\top\boldsymbol{Cb})\boldsymbol{I}$ (Model 1)
or
$\mathbf{E}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}\big) = \boldsymbol{BCy}$, $\mathbf{Cov}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}\big) = \boldsymbol{B}_0 - \boldsymbol{BCB}^\top$ (Model 2)
15: **return** $\mathbf{E}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}\big)$ and $\mathbf{Cov}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}\big)$

The kernel functions used in Models 1 and 2 (Equations 23 and 27) are both based on the Fisher kernel. Computing the Fisher kernel requires calculating the Fisher information matrix $\boldsymbol{G}(\boldsymbol{\theta})$ (Equation 24). Consequently, every time we update the policy parameters, we need to recompute $\boldsymbol{G}$. In Algorithm 1 we assume that the Fisher information matrix is known. However, in most practical situations this will not be the case, and consequently the Fisher information matrix must be estimated. Let us briefly outline two possible approaches for estimating the Fisher information matrix in an online manner.

1) Monte-Carlo Estimation: The BPG algorithm generates a number of sample paths using the current policy parameterized by $\boldsymbol{\theta}$ in order to estimate the gradient $\nabla\eta_B(\boldsymbol{\theta})$. We can use these generated sample paths to estimate the Fisher information matrix $\boldsymbol{G}(\boldsymbol{\theta})$ in an online manner, by replacing the expectation in $\boldsymbol{G}$ with empirical averaging as $\hat{\boldsymbol{G}}_M(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{u}(\xi_i)\boldsymbol{u}(\xi_i)^\top$. It can be shown that $\hat{\boldsymbol{G}}_M$ is an unbiased estimator of $\boldsymbol{G}$. One may obtain this estimate recursively $\hat{\boldsymbol{G}}_{i+1} = (1-\frac{1}{i})\hat{\boldsymbol{G}}_i + \frac{1}{i}\boldsymbol{u}(\xi_i)\boldsymbol{u}(\xi_i)^\top$, or more generally $\hat{\boldsymbol{G}}_{i+1} = (1-\zeta_i)\hat{\boldsymbol{G}}_i + \zeta_i\boldsymbol{u}(\xi_i)\boldsymbol{u}(\xi_i)^\top$, where ζ_i is a step-size with $\sum_i \zeta_i = \infty$ and $\sum_i \zeta_i^2 < \infty$. Using the Sherman-Morrison matrix inversion lemma, it is possible to directly estimate the inverse of the Fisher information matrix as

$$\hat{\boldsymbol{G}}_{i+1}^{-1} = \frac{1}{1-\zeta_i}\left[\hat{\boldsymbol{G}}_i^{-1} - \zeta_i \frac{\hat{\boldsymbol{G}}_i^{-1}\boldsymbol{u}(\xi_i)\big(\hat{\boldsymbol{G}}_i^{-1}\boldsymbol{u}(\xi_i)\big)^\top}{1-\zeta_i + \zeta_i\boldsymbol{u}(\xi_i)^\top\hat{\boldsymbol{G}}_i^{-1}\boldsymbol{u}(\xi_i)}\right].$$

2) Maximum Likelihood Estimation: The Fisher information matrix defined by Equation 24 depends on the probability distribution over paths. This distribution is a product of two factors, one corresponding to the current policy and the other corresponding to the MDP's state-transition probability P (see Equation 1). Thus if P is known, the Fisher information matrix may be evaluated offline. We can model P using a parameterized model and then estimate the maximum likelihood (ML) model parameters. This approach may lead to a model-based treatment of policy gradients, which could allow us to transfer information between different policies. Current policy gradient algorithms, including the algorithms described in this paper, are extremely wasteful of training data, since they do not have any *disciplined* way to use data collected for previous policy updates in computing the update of the current policy. Model-based policy gradient may help solve this problem.

4.4 BPG Online Sparsification

Algorithm 1 can be made more efficient, both in time and memory, by sparsifying the solution. Such sparsification may be performed incrementally and helps to numerically stabilize the algorithm when the kernel matrix is singular, or nearly so. Sparsification may, in some cases, reduce the accuracy of the solution (the posterior moments of the policy gradient), but it often makes the algorithms significantly faster, especially for large sample sizes. Here we use an online sparsification method proposed by Engel et al. (2002) (see also Csató and Opper, 2002) to selectively add a new observed path to a set of *dictionary* paths $\tilde{\mathcal{D}}$ used as a basis for representing or approximating the full solution. We only add a new path ξ_i to $\tilde{\mathcal{D}}$, if $k(\xi_i,\xi_i) - \tilde{\boldsymbol{k}}(\xi_i)^\top\tilde{\boldsymbol{K}}^{-1}\tilde{\boldsymbol{k}}(\xi_i) > \tau$, where $\tilde{\boldsymbol{k}}$ and $\tilde{\boldsymbol{K}}$ are the dictionary kernel vector and kernel matrix before observing ξ_i, respectively, and τ is a positive threshold parameter that determines the level of accuracy in the approximation as well as the level of sparsity attained. If the new path is added to $\tilde{\mathcal{D}}$ the dictionary kernel matrix $\tilde{\boldsymbol{K}}$ is expanded as shown in Equation 28.

Proposition 5 *Let $\tilde{\boldsymbol{K}}$ be the $m \times m$ sparse kernel matrix, where $m \leq M$ is the cardinality of $\tilde{\mathcal{D}}_M$. Let $\boldsymbol{A}$ be the $M \times m$ matrix, whose ith row is $[\boldsymbol{A}]_{i,|\tilde{\mathcal{D}}_i|} = 1$ and $[\boldsymbol{A}]_{i,j} = 0$; $\forall j \neq |\tilde{\mathcal{D}}_i|$, if we add the sample path ξ_i to the set of sample paths, and be $\tilde{\boldsymbol{k}}(\xi_i)^\top\tilde{\boldsymbol{K}}^{-1}$ followed by zeros, otherwise. Finally, let $(\tilde{\boldsymbol{b}})_i = 1 + \boldsymbol{u}(\xi_i)^\top\boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)$ and $\tilde{\boldsymbol{B}} = [\boldsymbol{u}(\xi_1), \ldots, \boldsymbol{u}(\xi_m)]$ with $\xi_i \in \tilde{\mathcal{D}}$. Then, using the sparsification method described above, the posterior moments of the gradient*

are given by

$$\mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \boldsymbol{Y}\boldsymbol{\Sigma}^{-1}\boldsymbol{A}\big(\tilde{\boldsymbol{K}}\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{A}+\boldsymbol{I}\big)^{-1}\tilde{\boldsymbol{b}}$$

for Model 1

$$\mathbf{Cov}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \Big[(1+n)-\tilde{\boldsymbol{b}}^\top\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{A}\big(\tilde{\boldsymbol{K}}\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{A}+\boldsymbol{I}\big)^{-1}\tilde{\boldsymbol{b}}\Big]\boldsymbol{I}$$

and

$$\mathbf{E}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \tilde{\boldsymbol{B}}\big(\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{A}\tilde{\boldsymbol{K}}+\boldsymbol{I}\big)^{-1}\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{y}$$

for Model 2

$$\mathbf{Cov}\big[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big] = \boldsymbol{G}-\tilde{\boldsymbol{B}}\big(\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{A}\tilde{\boldsymbol{K}}+\boldsymbol{I}\big)^{-1}\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{A}\tilde{\boldsymbol{B}}^\top\,.$$

Proof See Appendix C. ■

4.5 A Bayesian Policy Gradient Algorithm

So far we were concerned with estimating the gradient of the expected return with respect to the policy parameters. In this section, we present a Bayesian policy gradient (BPG) algorithm that employs the Bayesian gradient estimation methods proposed in Section 4.3 to update the policy parameters. The pseudo-code of this algorithm is shown in Algorithm 2. The algorithm starts with an initial vector of policy parameters $\boldsymbol{\theta}_0$, and updates the parameters in the direction of the posterior mean of the gradient of the expected return estimated by Algorithm 1. This is repeated N times, or alternatively, until the gradient estimate is sufficiently close to zero.

Algorithm 2 A Bayesian Policy Gradient Algorithm

BPG$(\boldsymbol{\theta}_0, \boldsymbol{\beta}, N, M)$
- initial policy parameters $\boldsymbol{\theta}_0 \in \mathbb{R}^n$
- learning rates β_j , $j = 0, \ldots, N-1$
- number of policy updates $N > 0$
- sample size $M > 0$ for the gradient evaluation algorithm (**BPG_Eval**)

for $j = 0$ to $N-1$ **do**
  $\Delta\boldsymbol{\theta}_j = \mathbf{E}\big(\nabla\eta_B(\boldsymbol{\theta}_j)|\mathcal{D}_M\big)$ from **BPG_Eval**$(\boldsymbol{\theta}_j, M)$
  $\boldsymbol{\theta}_{j+1} = \boldsymbol{\theta}_j + \beta_j\Delta\boldsymbol{\theta}_j$ (Conventional Gradient)
  or
  $\boldsymbol{\theta}_{j+1} = \boldsymbol{\theta}_j + \beta_j\boldsymbol{G}(\boldsymbol{\theta}_j)^{-1}\Delta\boldsymbol{\theta}_j$ (Natural Gradient)
end for
return $\boldsymbol{\theta}_N$

5. Extension to Partially Observable Markov Decision Processes

The Bayesian policy gradient models and algorithms of Section 4 can be extended to partially observable Markov decision processes (POMDPs) along the same lines as in Section 6

of Baxter and Bartlett (2001). In the partially observable case, the stochastic parameterized policy $\mu(\cdot|\cdot;\boldsymbol{\theta})$ controls a POMDP, i.e., the policy has access to an observation process that depends on the state, but it may not observe the state itself directly.

Specifically, for each state $x \in \mathcal{X}$, an observation $o \in \mathcal{O}$ is generated independently according to a probability distribution P_o over observations in $\mathcal{O}$. We denote the probability of observation o at state x by $P_o(o|x)$. A stationary stochastic parameterized policy $\mu(\cdot|\cdot;\boldsymbol{\theta})$ is a function mapping observations $o \in \mathcal{O}$ into probability distributions over the actions $\mu(\cdot|o;\boldsymbol{\theta}) \in \mathcal{P}(\mathcal{A})$. In this case, the probability of a path $\xi = (x_0, a_0, x_1, a_1, \ldots, x_{T-1}, a_{T-1}, x_T)$, $T \in \{0, 1, \ldots, \infty\}$ generated by the Markov chain induced by policy $\mu(\cdot|\cdot;\boldsymbol{\theta})$ is given by

$$\Pr(\xi;\mu) = \Pr(\xi;\boldsymbol{\theta}) = P_0(x_0)\int\prod_{t=0}^{T-1}P_o(o_t|x_t)\mu(a_t|o_t;\boldsymbol{\theta})P(x_{t+1}|x_t,a_t)do_0da_0\ldots do_{T-1}da_{T-1}.$$

The Fisher score of this path may be written as

$$\boldsymbol{u}(\xi;\boldsymbol{\theta}) = \nabla\log\Pr(\xi;\boldsymbol{\theta}) = \frac{\nabla\Pr(\xi;\boldsymbol{\theta})}{\Pr(\xi;\boldsymbol{\theta})} = \int\left(\sum_{t=0}^{T-1}\nabla\log\mu(a_t|o_t;\boldsymbol{\theta})\right)do_0da_0\ldots do_{T-1}da_{T-1},$$

which is the same as in the observable case (Equation 5), except here the policy is defined over observations instead of states. As a result, the models and algorithms of Section 4 may be used in the partially observable case with no change, substituting observations for states.

Moreover, similarly to the gradient estimated by the GPOMDP algorithm in Baxter and Bartlett (2001), the gradient estimated by Algorithm 1, $\nabla\eta_B(\boldsymbol{\theta})$, may be employed with the conjugate-gradients and line-search methods of Baxter et al. (2001) for making better use of gradient information. This allows us to exploit the information contained in the gradient estimate more aggressively than by simply adjusting the parameters by a small amount in the direction of $\nabla\eta_B(\boldsymbol{\theta})$. Conjugate-gradients and line-search are two widely used techniques in non-stochastic optimization that allow us to find better gradient directions than the pure gradient direction, and to obtain better step sizes, respectively.

Note that in this section, we followed Baxter and Bartlett (2001) (the GPOMDP algorithm) and considered stochastic policies that map observations to actions. However, as mentioned by Baxter and Bartlett (2001), it is immediate that the same algorithm works for any finite history of observations. Moreover, along the same way that Aberdeen and Baxter (2001) showed that GPOMDP can be extended to apply to policies with internal state, our BPG POMDP algorithm can also be extended to handle such policies.

6. BPG Experimental Results

In this section, we compare the Bayesian quadrature (BQ) and the plain MC gradient estimates on a simple bandit problem as well as on a continuous state and action linear quadratic regulator (LQR). We also evaluate the performance of the Bayesian policy gradient (BPG) algorithm described in Algorithm 2 on the LQR, and compare it with a Monte-Carlo based policy gradient (MCPG) algorithm.

	Exact	MC (10)	BQ (10)	MC (100)	BQ (100)
$r(a) = a$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0.995 \pm 0.438\\-0.001 \pm 0.977\end{pmatrix}$	$\begin{pmatrix}0.986 \pm 0.050\\0.001 \pm 0.060\end{pmatrix}$	$\begin{pmatrix}1.000 \pm 0.140\\0.004 \pm 0.317\end{pmatrix}$	$\begin{pmatrix}1.000 \pm 0.000001\\0.000 \pm 0.000004\end{pmatrix}$
$r(a) = a^2$	$\begin{pmatrix}0\\2\end{pmatrix}$	$\begin{pmatrix}0.014 \pm 1.246\\2.034 \pm 2.831\end{pmatrix}$	$\begin{pmatrix}0.001 \pm 0.082\\1.925 \pm 0.226\end{pmatrix}$	$\begin{pmatrix}0.005 \pm 0.390\\1.987 \pm 0.857\end{pmatrix}$	$\begin{pmatrix}0.000 \pm 0.000003\\2.000 \pm 0.000011\end{pmatrix}$

Table 2: The true gradient of the expected return and its MC and BQ estimates for two instances of the bandit problem corresponding to two different reward functions.

System:
Initial State: $x_0 \sim \mathcal{N}(0.3, 0.001)$
Reward: $r_t = x_t^2 + 0.1a_t^2$
Transition: $x_{t+1} = x_t + a_t + n_x \; ; \; n_x \sim \mathcal{N}(0, 0.01)$

Policy:
Actions: $a_t \sim \mu(\cdot|x_t; \boldsymbol{\theta}) = \mathcal{N}(\lambda x_t, \sigma^2)$
Parameters: $\boldsymbol{\theta} = (\lambda \; , \; \sigma)^\top$

6.1 A Simple Bandit Problem

The goal of this example is to compare the BQ and MC estimates of the gradient (for some fixed set of policy parameters) using the same sample. Our bandit problem has a single state and a continuous action space $\mathcal{A} = \mathbb{R}$, thus, each path ξ_i consists of a single action a_i. The policy, and therefore also the distribution over the paths is given by $a \sim \mathcal{N}(\theta_1 = 0, \theta_2^2 = 1)$. The parameters θ_1 and θ_2 are the mean and the standard deviation of this distribution. The score function of the path $\xi = a$ and the Fisher information matrix for the policy are $\boldsymbol{u}(\xi) = [a, a^2 - 1]^\top$ and $\boldsymbol{G} = \text{diag}(1, 2)$, respectively.

Table 2 shows the exact gradient of the expected return and its MC and BQ estimates using 10 and 100 samples for two instances of the bandit problem corresponding to two different deterministic reward functions $r(a) = a$ and $r(a) = a^2$. The average over 10^4 runs of the MC and BQ estimates and their standard deviations are reported in Table 2. The true gradient is analytically tractable and is reported as "Exact" in Table 2 for reference.

As shown in Table 2, the variance of the BQ estimates are lower than the variance of the MC estimates by an order of magnitude for the small sample size ($M = 10$), and by 6 orders of magnitude for the large sample size ($M = 100$). The BQ estimate is also more accurate than the MC estimate for the large sample size, and is roughly the same for the small sample size.

6.2 Linear Quadratic Regulator

In this section, we consider the following linear system in which the goal is to minimize the expected return over 20 steps.[14] Thus, it is an episodic problem with paths of length 20.

We run two sets of experiments on this system. We first fix the set of policy parameters and compare the BQ and MC estimates of the gradient of the expected return using the same sample. We then proceed to solving the complete policy gradient problem and compare the performance of the BPG algorithm (with both conventional and natural gradients) with a Monte-Carlo based policy gradient (MCPG) algorithm.

14. What we mean by reward and return in this section is in fact cost and loss, and this is why we are dealing with a minimization, and not a maximization, problem here. The reason for this is to maintain consistency in notations and definitions throughout the paper.

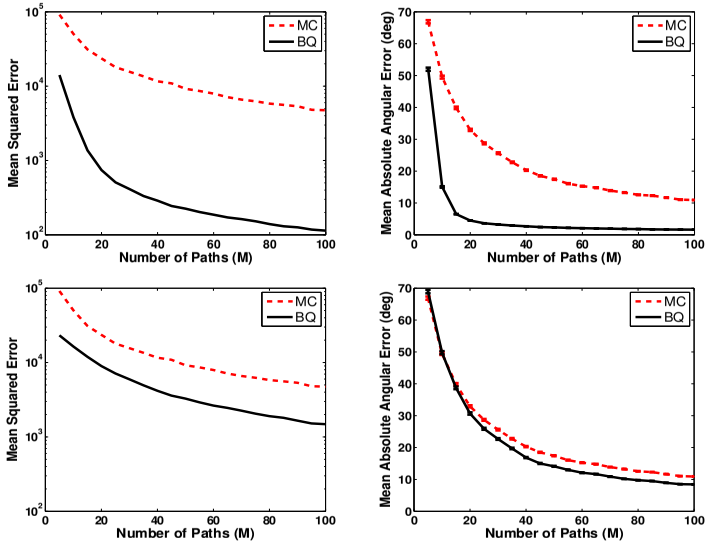


Figure 1: Results for the LQR problem using Model 1 (top row) and Model 2 (bottom row) with sparsification. Shown are the MSE (left column) and the mean absolute angular error (right column) of the MC and BQ estimates as a function of the number of sample paths M. All results are averaged over 10^4 runs.

6.2.1 Gradient Estimation

In this section, we compare the BQ and MC estimates of the gradient of the expected return for the policy induced by parameters $\lambda = -0.2$ and $\sigma = 1$. We use several different sample sizes (number of paths used for gradient estimation) $M = 5j \; , \; j = 1, \ldots, 20$ for the BQ and MC estimates. For each sample size, we compute the MC and BQ estimators using the same sample, repeat this process 10^4 times, and then compute the average. The true gradient is analytically tractable and is used for comparison purposes.

Figure 1 shows the mean squared error (MSE) (left column) and the mean absolute angular error (right column) of the MC and BQ estimates of the gradient for several different sample sizes. The absolute angular error is the absolute value of the angle between the true and estimated gradients. In this figure, the BQ gradient estimates were calculated using Model 1 (top row) and Model 2 (bottom row) with sparsification. The error bars in the figures on the right column are the standard errors of the mean absolute angular errors.

We ran another set of experiments in which we added i.i.d. Gaussian noise to the rewards: $r_t = x_t^2 + 0.1a_t^2 + n_r \; ; \; n_r \sim \mathcal{N}(0, \sigma_r^2)$. Note that in Models 1 and 2, $y(\xi)$, the noisy sample

of $f(\xi)$, is of the form $R(\xi)\nabla \log \Pr(\xi;\boldsymbol{\theta})$ and $R(\xi)$, respectively (see Sections 4.1 and 4.2). Moreover, since each reward r_t is a Gaussian random variable with variance σ_r^2, the return $R(\xi) = \sum_{t=0}^{T-1} r_t$ is also a Gaussian random variable with variance $T\sigma_r^2$. Therefore in this case, the measurement noise covariance matrices for Models 1 and 2 may be written as $\boldsymbol{\Sigma} = T\sigma_r^2 \operatorname{diag}\left(\left(\frac{\partial}{\partial\theta_i}\log p(\xi_1;\boldsymbol{\theta})\right)^2,\ldots,\left(\frac{\partial}{\partial\theta_i}\log p(\xi_M;\boldsymbol{\theta})\right)^2\right)$ and $\boldsymbol{\Sigma} = T\sigma_r^2\boldsymbol{I}$, respectively, where $T = 20$ is the path length.[15] We tried two different Gaussian reward noise standard deviations: $\sigma_r = 0.1$ and 1 in our experiments. Adding noise to the rewards slightly increased the error of the BQ and MC estimates of the gradient. However, the graphs comparing these estimates remained quite similar to those shown in Figure 1. Hence in Figure 2, we compare the MSE (left column) and the mean absolute angular error (right column) of the BQ estimates with and without noise in the rewards as a function of the number of sample paths M. In this figure, the noise in the rewards has variance $\sigma_r^2 = 1$, and the BQ gradient estimates were calculated using Model 1 (top row) and Model 2 (bottom row) with sparsification.

6.2.2 Policy Optimization

In this section, we use Bayesian policy gradient (BPG) to optimize the policy parameters in the LQR problem. Figure 3 shows the performance of the BPG algorithm with the conventional (BPG) and natural (BPNG) gradient estimates, versus a MC-based policy gradient (MCPG) algorithm, for sample sizes (number of sample paths used to estimate the gradient of each policy) $M = 5$, 10, 20, and 40. We use Algorithm 2 with the number of updates set to $N = 100$, and Model 1 with sparsification for the BPG and BPNG methods. Since Algorithm 2 computes the Fisher information matrix for each set of policy parameters, the estimate of the natural gradient is provided at little extra cost at each step. The returns obtained by these methods are averaged over 10^4 runs. The policy parameters are initialized randomly at each run. In order to ensure that the learned parameters do not exceed an acceptable range, the policy parameters are defined as $\lambda = -1.999 + 1.998/(1+e^{\kappa_1})$ and $\sigma = 0.001 + 1/(1+e^{\kappa_2})$. The optimal solution is $\lambda^* \approx -0.92$, $\sigma^* = 0.001$, $\eta_B(\lambda^*,\sigma^*) = 0.3067$, corresponding to $\kappa_1^* \approx -0.16$ and $\kappa_2^* \to \infty$.

Figure 3 shows that the MCPG algorithm performs better than BPG and BPNG only for the smallest sample size ($M = 5$), whereas for larger samples BPG and BPNG dominate MCPG. The better performance of MCPG for very small sample size is due to the fact that in this case, the Bayesian estimators, BPG and BPNG, like any other Bayesian estimator or posterior in such case, rely more on the prior, and thus, are not accurate if the prior is not very informative. A similar phenomenon was also reported by Rasmussen and Ghahramani (2003). We used two different learning rates for the two components of the gradient. For a fixed sample size, BPG and MCPG methods start with an initial learning rate and decrease it according to the schedule $\beta_j = \beta_0\big(20/(20+j)\big)$. The BPNG algorithm uses a fixed learning rate multiplied by the determinant of the Fisher information matrix. We tried many values for the initial learning rates used by these algorithms and those in Table 3 yielded the best performance of those we tried.

So far we have assumed that the Fisher information matrix is known. In the next experiment, we estimate it using both MC and a model-based maximum likelihood (ML)

15. In Model 1, $\boldsymbol{\Sigma}$ is the measurement noise covariance matrix for the ith component of the gradient $\frac{\partial}{\partial\theta_i}\eta_B(\boldsymbol{\theta})$. Note that $\frac{\partial}{\partial\theta_i}\log p(\xi_j;\boldsymbol{\theta})$ depends only on the policy and can be calculated using Equation 5.

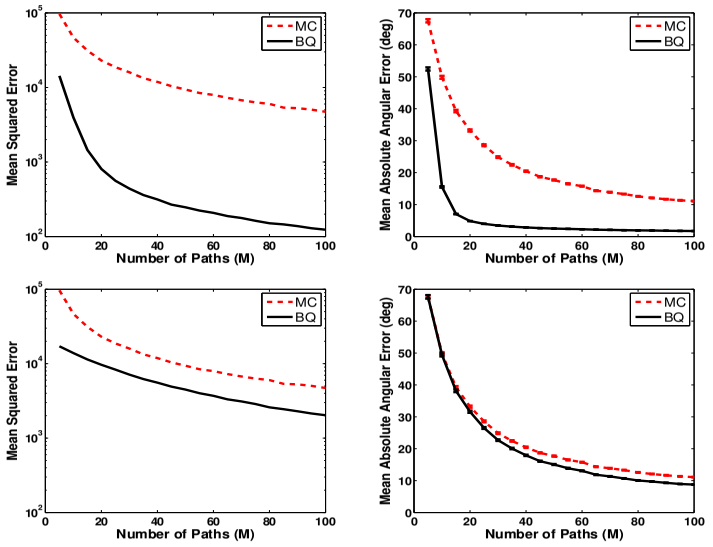


Figure 2: Results for the LQR problem in which the rewards are corrupted by i.i.d. Gaussian noise with $\sigma_r^2 = 1$. Shown are the MSE (left column) and the mean absolute angular error (right column) of the BQ estimates with and without noise in the rewards as a function of the number of sample paths M. The BQ gradient estimates were calculated using Model 1 (top row) and Model 2 (bottom row) with sparsification. All results are averaged over 10^4 runs.

β_0	$M=5$	$M=10$	$M=20$	$M=40$
MCPG	0.01, 0.05	0.05, 0.05	0.05, 0.10	0.05, 0.10
BPG	0.01, 0.05	0.07, 0.10	0.15, 0.15	0.10, 0.30
BPNG	0.010, 0.005	0.010, 0.005	0.015, 0.005	0.015, 0.005
BPG-var	0.05, 0.05	0.10, 0.10	0.10, 0.15	0.15, 0.30

Table 3: Initial learning rates β_0 used by the policy gradient algorithms for the two components of the gradient.

method, as discussed in Section 4.3. In the ML approach, we model the transition probability function as $P(x_{t+1}|x_t,a_t) = \mathcal{N}(c_1x_t + c_2a_t + c_3, c_4^2)$, and then estimate its parameters (c_1,c_2,c_3,c_4) using observing state transitions. Figure 4 shows that the BPG algorithm, when the Fisher information matrix is estimated using ML and MC, still performs bet-

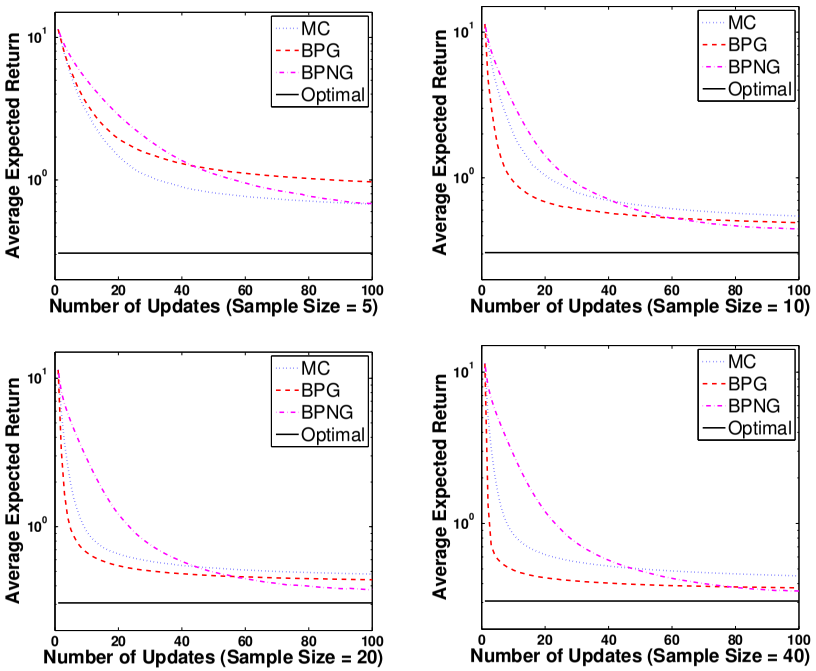


Figure 3: A comparison of the average expected returns of the Bayesian policy gradient algorithm using conventional (BPG) and natural (BPNG) gradient estimates, with the average expected return of a MC-based policy gradient algorithm (MCPG) for sample sizes $M = 5$, 10, 20, and 40. All results are averaged over 10^4 runs.

ter than MCPG. Top and bottom rows contain the results for the BPG algorithm with conventional (BPG-ML and BPG-MC) and natural (BPNG-ML and BPNG-MC) gradient estimates, respectively. Although the BPG-ML (BPNG-ML) outperforms BPG-MC (BPNG-MC) for small sample sizes, the difference in their performance disappears as we increase the sample size. One reason for the good performance of BPG-ML is that the form of the state transition function $P(x_{t+1}|x_t, a_t)$ has been selected correctly. Here we used the same initial learning rates and learning rate schedules as in the experiments of Figure 3 (see Table 3).

Although the proposed Bayesian policy gradient algorithm (Algorithm 2) uses only the posterior mean of the gradient in its updates, it can be extended to make judicious use of the

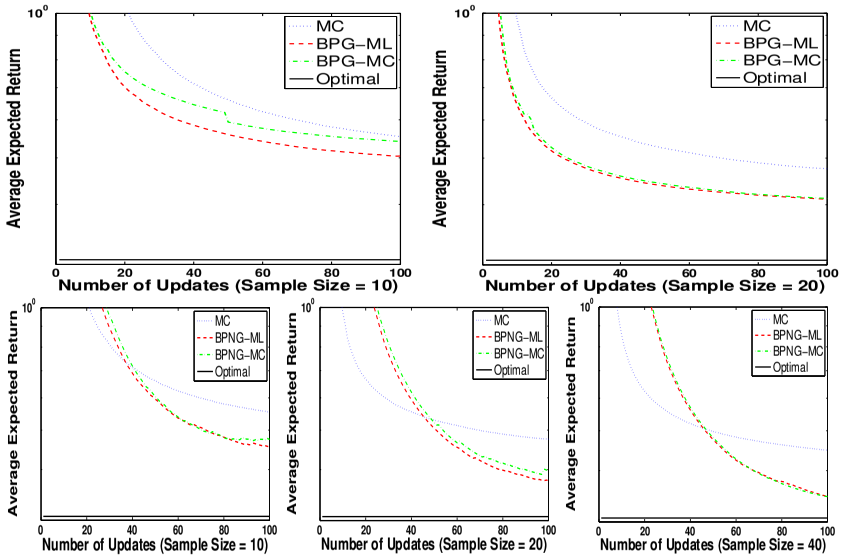


Figure 4: A comparison of the average expected returns of the BPG algorithm, when the Fisher information matrix is estimated using ML and MC, with the average expected return of a MC-based policy gradient algorithm (MCPG). The top and bottom rows contain the results for the BPG algorithm with conventional (BPG-ML and BPG-MC) and natural (BPNG-ML and BPNG-MC) gradient estimates, respectively. All results are averaged over 10^4 runs.

second moment information provided by the Bayesian policy gradient estimation algorithm (Algorithm 1). In the last experiment of this section, we use the posterior covariance of the gradient, provided by Algorithm 1, to select the learning rate and the direction of the updates in Algorithm 2. The idea is to use a small learning rate when the variance of the gradient estimate is large, and to have a large update when it is small. We refer to the resulting algorithm by the name BPG-var. This algorithm uses a fixed learning rate parameter (see Table 3) multiplied by $\Big[(1+n)\boldsymbol{I} - \mathbf{Cov}\big(\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\big)\Big]/(1+n)$ in its updates. Note that $n+1$ is b_0 in the calculation of the posterior covariance of the gradient in Model 1 (see Proposition 3), and is used here as an upper bound for the posterior covariance of the gradient. Figure 5 compares the average expected return of BPG-var with BPG and MCPG for sample sizes $M = 5$, 10, 20, and 40. The figure shows that BPG-var performs better than BPG and MCPG for all the sample sizes. It even has a better performance than MCPG for the smallest sample size ($M = 5$). Comparing Figures 3 and 5 shows that

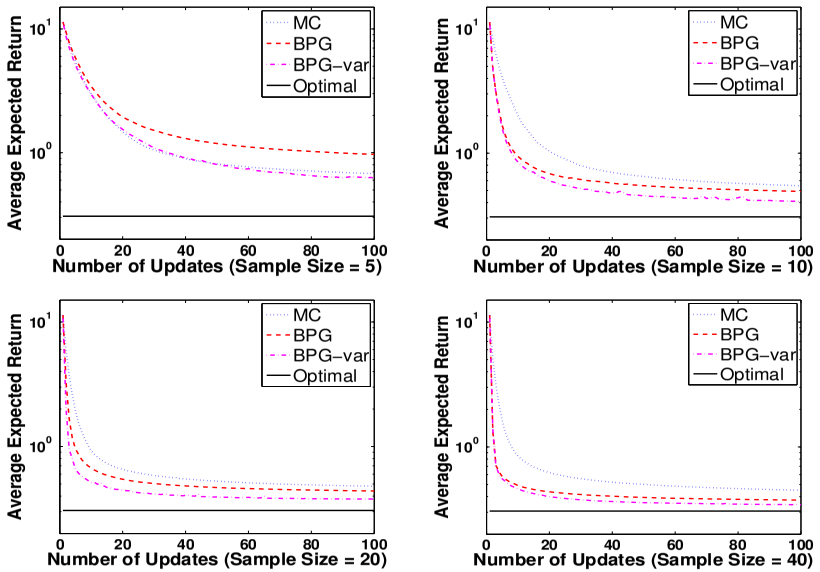


Figure 5: A comparison of the average expected returns of the BPG algorithm that uses the posterior covariance in its updates (BPG-var), with the average expected return of the BPG and a MC-based policy gradient algorithm (MCPG) for sample sizes $M = 5$, 10, 20, and 40. All results are averaged over 10^4 runs.

BPG-var converges faster than BPNG and has similar final performance. As we expected, BPG-var and BPG perform more and more alike as we increase the sample size. This is because by increasing the sample size the estimated gradient (the posterior mean of the gradient), and as a result, the update direction used by BPG becomes more reliable.

In an approach similar to the one used in the experiments of Figure 5, Vien et al. (2011) used BQ to estimate the Hessian matrix distribution, and then used its mean as learning rate schedule to improve the performance of BPG. They empirically showed that their method performs better than BPG and BPNG in terms of convergence speed.

7. Bayesian Actor-Critic

The models and algorithms of Section 4 consider complete trajectories as the basic observable unit, and thus, do not require the dynamics within each trajectory to be of any special form. In particular, it is not necessary for the dynamics to have the Markov property,

allowing the resulting algorithms to handle partially observable MDPs, Markov games, and other non-Markovian systems. On the down side, these algorithms cannot take advantage of the Markov property when operating in Markovian systems. Moreover, since the unit of observation of these algorithms is the entire trajectory, their gradient estimates have larger variance than the algorithms that will be discussed in this section, whose unit of observation is *(current state, action, next state)*, since they take advantage of the Markov property, especially when the size of the trajectories is large.

In this section, we apply the Bayesian quadrature idea to the policy gradient expression given by Equation 7, i.e.,

$$\nabla\eta(\boldsymbol{\theta}) = \int dxda\ \nu(x;\boldsymbol{\theta})\nabla\mu(a|x;\boldsymbol{\theta})Q(x,a;\boldsymbol{\theta}),$$

and derive a family of Bayesian actor-critic (BAC) algorithms. In this approach, we place a Gaussian process (GP) prior over action-value functions using a prior covariance kernel defined on state-action pairs: $k(\boldsymbol{z},\boldsymbol{z}') = \mathbf{Cov}\big[Q(\boldsymbol{z}),Q(\boldsymbol{z}')\big]$. We then compute the GP posterior conditioned on the sequence of individual observed transitions. In the same vein as Section 4, by an appropriate choice of a prior on action-value functions, we are able to derive closed-form expressions for the posterior moments of $\nabla\eta(\boldsymbol{\theta})$. The main questions here are: **1)** how to compute the GP posterior of the action-value function given a sequence of observed transitions? and **2)** how to choose a prior for the action-value function that allows us to derive closed-form expressions for the posterior moments of $\nabla\eta(\boldsymbol{\theta})$? Fortunately, well developed machinery for computing the posterior moments of $Q(\boldsymbol{z})$ is provided in a series of papers by Engel et al. (2003, 2005) (for a thorough treatment see Engel, 2005). In the next two sections, we will first briefly review some of the main results pertaining to the Gaussian process temporal difference (GPTD) model proposed in Engel et al. (2005), and then will show how they may be combined with the Bayesian quadrature idea in developing a family of Bayesian actor-critic algorithms.

7.1 Gaussian Process Temporal Difference Learning

The Gaussian process temporal difference (GPTD) learning (Engel et al., 2003, 2005) model is based on a statistical generative model relating the observed reward signal r to the unobserved action-value function Q

$$r(\boldsymbol{z}_i) = Q(\boldsymbol{z}_i) - \gamma Q(\boldsymbol{z}_{i+1}) + N(\boldsymbol{z}_i,\boldsymbol{z}_{i+1}), \tag{29}$$

where $N(\boldsymbol{z}_i,\boldsymbol{z}_{i+1})$ is a zero-mean noise signal that accounts for the discrepancy between $r(\boldsymbol{z}_i)$ and $Q(\boldsymbol{z}_i) - \gamma Q(\boldsymbol{z}_{i+1})$. Let us define the finite-dimensional processes $\boldsymbol{r}_t$, Q_t, N_t, and the $t \times (t+1)$ matrix $\boldsymbol{H}_t$ as follows:

$$\begin{aligned} &\boldsymbol{r}_t = \big(r(\boldsymbol{z}_0),\ldots,r(\boldsymbol{z}_t)\big)^\top, && Q_t = \big(Q(\boldsymbol{z}_0),\ldots,Q(\boldsymbol{z}_t)\big)^\top, \\ &N_t = \big(N(\boldsymbol{z}_0,\boldsymbol{z}_1),\ldots,N(\boldsymbol{z}_{t-1},\boldsymbol{z}_t)\big)^\top, \end{aligned} \tag{30}$$

$$\boldsymbol{H}_t = \begin{bmatrix} 1 & -\gamma & 0 & \ldots & 0 \\ 0 & 1 & -\gamma & \ldots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \ldots & 1 & -\gamma \end{bmatrix}. \tag{31}$$

The set of Equations 29 for $i = 0, \ldots, t-1$ may be written as $\boldsymbol{r}_{t-1} = \boldsymbol{H}_t Q_t + N_t$. Under certain assumptions on the distribution of the discounted return random process (Engel et al., 2005), the covariance of the noise vector N_t is given by

$$\boldsymbol{\Sigma}_t = \sigma^2 \boldsymbol{H}_t \boldsymbol{H}_t^\top = \sigma^2 \begin{bmatrix} 1+\gamma^2 & -\gamma & 0 & \ldots & 0 \\ -\gamma & 1+\gamma^2 & -\gamma & \ldots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & \ldots & -\gamma & 1+\gamma^2 \end{bmatrix}. \tag{32}$$

In episodic tasks, if $\boldsymbol{z}_{t-1}$ is the last state-action pair in the episode (i.e., $\boldsymbol{x}_t$ is a zero-reward absorbing terminal state), $\boldsymbol{H}_t$ becomes a square $t \times t$ invertible matrix of the form shown in Equation 31 with its last column removed. The effect on the noise covariance matrix $\boldsymbol{\Sigma}_t$ is that the bottom-right element becomes 1 instead of $1+\gamma^2$.

Placing a GP prior on Q and assuming that N_t is also normally distributed, we may use Bayes' rule to obtain the posterior moments of Q:

$$\begin{aligned} \hat{Q}_t(\boldsymbol{z}) &= \mathbf{E}\left[Q(\boldsymbol{z})|\mathcal{D}_t\right] = \boldsymbol{k}_t(\boldsymbol{z})^\top \boldsymbol{\alpha}_t, \\ \hat{S}_t(\boldsymbol{z}, \boldsymbol{z}') &= \mathbf{Cov}\left[Q(\boldsymbol{z}), Q(\boldsymbol{z}')|\mathcal{D}_t\right] = k(\boldsymbol{z}, \boldsymbol{z}') - \boldsymbol{k}_t(\boldsymbol{z})^\top \boldsymbol{C}_t \boldsymbol{k}_t(\boldsymbol{z}'), \end{aligned} \tag{33}$$

where $\mathcal{D}_t$ denotes the observed data up to and including time step t. We used here the following definitions:

$$\begin{aligned} \boldsymbol{k}_t(\boldsymbol{z}) &= \left(k(\boldsymbol{z}_0, \boldsymbol{z}), \ldots, k(\boldsymbol{z}_t, \boldsymbol{z})\right)^\top, & \boldsymbol{K}_t &= \left[\boldsymbol{k}_t(\boldsymbol{z}_0), \boldsymbol{k}_t(\boldsymbol{z}_1), \ldots, \boldsymbol{k}_t(\boldsymbol{z}_t)\right], \\ \boldsymbol{\alpha}_t &= \boldsymbol{H}_t^\top \left(\boldsymbol{H}_t \boldsymbol{K}_t \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_t\right)^{-1} \boldsymbol{r}_{t-1}\,, & \boldsymbol{C}_t &= \boldsymbol{H}_t^\top \left(\boldsymbol{H}_t \boldsymbol{K}_t \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_t\right)^{-1} \boldsymbol{H}_t\,. \end{aligned} \tag{34}$$

Note that $\hat{Q}_t(\boldsymbol{z})$ and $\hat{S}_t(\boldsymbol{z}, \boldsymbol{z}')$ are the posterior mean and covariance functions of the posterior GP, respectively. As more samples are observed, the posterior covariance decreases, reflecting a growing confidence in the Q-function estimate $\hat{Q}_t$.

7.2 A Family of Bayesian Actor-Critic Algorithms

We are now in a position to describe the main idea behind our BAC approach. Making use of the linearity of Equation 7 in Q and denoting $\boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) = \pi^\mu(\boldsymbol{z}) \nabla \log \mu(\boldsymbol{a}|\boldsymbol{x}; \boldsymbol{\theta})$, we obtain the following expressions for the posterior moments of the policy gradient (O'Hagan, 1991):

$$\begin{aligned} \mathbf{E}[\nabla \eta(\boldsymbol{\theta})|\mathcal{D}_t] &= \int_{\mathcal{Z}} d\boldsymbol{z} \boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) \hat{Q}_t(\boldsymbol{z}; \boldsymbol{\theta}), \\ \mathbf{Cov}\left[\nabla \eta(\boldsymbol{\theta})|\mathcal{D}_t\right] &= \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) \hat{S}_t(\boldsymbol{z}, \boldsymbol{z}') \boldsymbol{g}(\boldsymbol{z}'; \boldsymbol{\theta})^\top. \end{aligned} \tag{35}$$

Substituting the expressions for the posterior moments of Q from Equation 33 into Equation 35, we obtain

$$\begin{aligned} \mathbf{E}[\nabla \eta(\boldsymbol{\theta})|\mathcal{D}_t] &= \int_{\mathcal{Z}} d\boldsymbol{z} \boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) \boldsymbol{k}_t(\boldsymbol{z})^\top \boldsymbol{\alpha}_t, \\ \mathbf{Cov}\left[\nabla \eta(\boldsymbol{\theta})|\mathcal{D}_t\right] &= \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) \left(k(\boldsymbol{z}, \boldsymbol{z}') - \boldsymbol{k}_t(\boldsymbol{z})^\top \boldsymbol{C}_t \boldsymbol{k}_t(\boldsymbol{z}')\right) \boldsymbol{g}(\boldsymbol{z}'; \boldsymbol{\theta})^\top. \end{aligned}$$

These equations provide us with the general form of the posterior policy gradient moments. We are now left with a computational issue, namely, how to compute the integrals appearing in these expressions? We need to be able to evaluate the following integrals:

$$\boldsymbol{B}_t = \int_{\mathcal{Z}} d\boldsymbol{z} \boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) \boldsymbol{k}_t(\boldsymbol{z})^\top, \qquad \boldsymbol{B}_0 = \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \boldsymbol{g}(\boldsymbol{z}; \boldsymbol{\theta}) k(\boldsymbol{z}, \boldsymbol{z}') \boldsymbol{g}(\boldsymbol{z}'; \boldsymbol{\theta})^\top. \tag{36}$$

Using these definitions, we may write the gradient posterior moments compactly as

$$\mathbf{E}[\nabla \eta(\boldsymbol{\theta})|\mathcal{D}_t] = \boldsymbol{B}_t \boldsymbol{\alpha}_t\,, \qquad \mathbf{Cov}\left[\nabla \eta(\boldsymbol{\theta})|\mathcal{D}_t\right] = \boldsymbol{B}_0 - \boldsymbol{B}_t \boldsymbol{C}_t \boldsymbol{B}_t^\top. \tag{37}$$

In order to render these integrals analytically tractable, we choose our prior covariance kernel to be the sum of an arbitrary state-kernel k_x and the (invariant) Fisher kernel k_F between state-action pairs (see e.g., Shawe-Taylor and Cristianini, 2004, Chapter 12). The (policy dependent) Fisher information kernel and our overall state-action kernel are then given by

$$k_F(\boldsymbol{z}, \boldsymbol{z}') = \boldsymbol{u}(\boldsymbol{z}; \boldsymbol{\theta})^\top \boldsymbol{G}(\boldsymbol{\theta})^{-1} \boldsymbol{u}(\boldsymbol{z}')\,, \qquad k(\boldsymbol{z}, \boldsymbol{z}') = k_x(\boldsymbol{x}, \boldsymbol{x}') + k_F(\boldsymbol{z}, \boldsymbol{z}')\,, \tag{38}$$

where $\boldsymbol{u}(\boldsymbol{z}; \boldsymbol{\theta})$ and $\boldsymbol{G}(\boldsymbol{\theta})$ are the score function and Fisher information matrix defined as[16]

$$\boldsymbol{u}(\boldsymbol{z}; \boldsymbol{\theta}) = \nabla \log \mu(a|x; \boldsymbol{\theta}), \tag{39}$$

$$\boldsymbol{G}(\boldsymbol{\theta}) = \mathbf{E}_{x \sim \nu^\mu, a \sim \mu}\left[\nabla \log \mu(a|x; \boldsymbol{\theta}) \nabla \log \mu(a|x; \boldsymbol{\theta})^\top\right] = \mathbf{E}_{\boldsymbol{z} \sim \pi^\mu}\left[\boldsymbol{u}(\boldsymbol{z}; \boldsymbol{\theta}) \boldsymbol{u}(\boldsymbol{z}; \boldsymbol{\theta})^\top\right]. \tag{40}$$

Although here we have total flexibility in selecting the state kernel, we are restricted to the Fisher kernel for state-action pairs. This restriction may cause an error in approximating some action-value functions Q. This error depends on the problem at hand and is hard to quantify. This is exactly the same as selecting an inaccurate prior in any Bayesian algorithm or choosing a wrong representation (function space) in any machine learning algorithm (referred to as *approximation error* in the approximation theory). However, this restriction did not cause a significant error in our experiments (see Section 8), as in almost all of them, the gradients estimated by BAC were more accurate than those estimated by the MC-based method, given the same number of samples.

Note that in Sections 4 to 6 we used a formulation in which the observable unit is a system trajectory, and thus, the expected return and its gradient are defined by Equations 2 and 4. In this formulation, the score function and Fisher information matrix are defined by Equations 5 and 24. However, in the formulation used in this section and in the rest of the paper, where the observable unit is an individual state-action-reward transition, the expected return and its gradient are defined by Equations 3 and 7. In this formulation, the score function and Fisher information matrix are defined by Equations 39 and 40, respectively.

A nice property of the Fisher kernel is that while $k_F(\boldsymbol{z}, \boldsymbol{z}')$ depends on the policy, it is invariant to policy reparameterization. In other words, it only depends on the actual probability mass assigned to each action and not on its explicit dependence on the policy parameters. As mentioned above, another attractive property of this particular choice of kernel is that it renders the integrals in Equation 36 analytically tractable, as made explicit in the following proposition

16. Similar to $\boldsymbol{u}(\xi)$ and $\boldsymbol{G}$ defined by Equations 5 and 24, to simplify the notation, we omit the dependence of $\boldsymbol{u}$ and $\boldsymbol{G}$ to the policy parameters $\boldsymbol{\theta}$, and replace $\boldsymbol{u}(\boldsymbol{z}; \boldsymbol{\theta})$ and $\boldsymbol{G}(\boldsymbol{\theta})$ with $\boldsymbol{u}(\boldsymbol{z})$ and $\boldsymbol{G}$ in the sequel.

Proposition 6 *Let $k(\boldsymbol{z}, \boldsymbol{z}') = k_x(x, x') + k_F(\boldsymbol{z}, \boldsymbol{z}')$ for all $(\boldsymbol{z}, \boldsymbol{z}') \in \mathcal{Z}^2$, where $k_x : \mathcal{X}^2 \to \mathbb{R}$ is an arbitrary positive definite state-kernel and $k_F : \mathcal{Z}^2 \to \mathbb{R}$ is the Fisher information kernel. Then $\boldsymbol{B}_t$ and $\boldsymbol{B}_0$ from Equation 36 satisfy*

$$\boldsymbol{B}_t = \boldsymbol{U}_t \,, \qquad\qquad \boldsymbol{B}_0 = \boldsymbol{G}, \tag{41}$$

where $\boldsymbol{U}_t = \big[\boldsymbol{u}(\boldsymbol{z}_0), \boldsymbol{u}(\boldsymbol{z}_1), \ldots, \boldsymbol{u}(\boldsymbol{z}_t)\big]$.

Proof See Appendix D. ■

An immediate consequence of Proposition 6 is that, in order to compute the posterior moments of the policy gradient, we only need to be able to evaluate (or estimate) the score vectors $\boldsymbol{u}(\boldsymbol{z}_i)$, $i = 0, \ldots, t$ and the Fisher information matrix $\boldsymbol{G}$ of our policy. Evaluating the Fisher information matrix $\boldsymbol{G}$ is somewhat more challenging, since on top of taking the expectation with respect to the policy $\mu(a|x;\boldsymbol{\theta})$, computing $\boldsymbol{G}$ involves an additional expectation over the state-occupancy density $\nu^\mu(x)$, which is not generally known. In most practical situations we therefore have to resort to estimating $\boldsymbol{G}$ from data. When ν^μ in the definition of the Fisher information matrix (Equation 40) is the stationary distribution over states under policy μ, one straightforward method to estimate $\boldsymbol{G}$ from a trajectory $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_t$ is to use the (unbiased) estimator (see Proposition 6 for the definition of $\boldsymbol{U}_t$):

$$\hat{\boldsymbol{G}}_t = \frac{1}{t+1} \sum_{i=0}^{t} \boldsymbol{u}(\boldsymbol{z}_i)\boldsymbol{u}(\boldsymbol{z}_i)^\top = \frac{1}{t+1} \boldsymbol{U}_t \boldsymbol{U}_t^\top. \tag{42}$$

In case ν^μ in Equation 40 is a discounted weighting of states encountered by following policy μ (as it is considered in this paper), a method for estimating $\boldsymbol{G}$ from a number of trajectories is shown in Algorithm 3. Note that $(1-\gamma)\nu^\mu$ corresponds to the distribution of a Markov chain that starts from a state sampled according to P_0 and at each step either follows the policy μ with probability γ or restarts from a new initial state drawn from P_0 with probability $1-\gamma$. It is easy to show that the average number of steps between two successive restarts of this distribution is $1/(1-\gamma)$.

Algorithm 4 is a pseudocode sketch of the Bayesian actor-critic algorithm, using either the conventional gradient or the natural gradient in the policy update, and with $\boldsymbol{G}$ estimated using either $\hat{\boldsymbol{G}}_t$ in Equation 42 or $\hat{\boldsymbol{G}}(\boldsymbol{\theta})$ in Algorithm 3.

7.3 BAC Online Sparsification

As was done for the BPG algorithms in Section 4.4, Algorithm 4 may be made more efficient, both in time and memory, by sparsifying the solution. Engel et al. (2005) presented a sparse approximation of the GPTD algorithm by using an online sparsification method from Engel et al. (2002). This sparsification method incrementally constructs a dictionary $\tilde{\mathcal{D}}$ of representative state-action pairs. Upon observing a new state-action pair $\boldsymbol{z}_i$, the distance between the feature-space image of $\boldsymbol{z}_i$ and the span of the images of current dictionary members is computed. If the squared distance $\delta_i = k(\boldsymbol{z}_i, \boldsymbol{z}_i) - \tilde{\boldsymbol{k}}_{i-1}^\top(\boldsymbol{z}_i)\tilde{\boldsymbol{K}}_{i-1}^{-1}\tilde{\boldsymbol{k}}_{i-1}(\boldsymbol{z}_i)$ exceeds some positive threshold τ, $\boldsymbol{z}_i$ is added to the dictionary, otherwise, it is left out. In

Algorithm 3 Fisher Information Matrix Estimation Algorithm

G-EST$(\boldsymbol{\theta}, M)$
- $\boldsymbol{\theta}$ policy parameters
- $M > 0$ number of episodes used to estimate the Fisher information matrix

$\hat{\boldsymbol{G}}(\boldsymbol{\theta}) = \boldsymbol{0}$
for $i = 1$ to M **do**
  done = false, term = false, $t = -1$
  Draw $x_0^i \sim P_0(\cdot)$
  while not done **do**
    $t = t + 1$
    Draw $a_t^i \sim \mu(\cdot|x_t^i;\boldsymbol{\theta})$ and $x_{t+1}^i \sim P(\cdot|x_t^i, a_t^i)$
    if $x_{t+1}^i = x_{\text{term}}$ **then** done = true
    if (done = false $\wedge$ term = false) **then**
      $\hat{\boldsymbol{G}}(\boldsymbol{\theta}) := \hat{\boldsymbol{G}}(\boldsymbol{\theta}) + \boldsymbol{u}(\boldsymbol{z}_t^i;\boldsymbol{\theta})\boldsymbol{u}(\boldsymbol{z}_t^i;\boldsymbol{\theta})^\top$ and w.p. $1-\gamma$ term = true
    end if
  end while
  if term = false **then** $\hat{\boldsymbol{G}}(\boldsymbol{\theta}) = \hat{\boldsymbol{G}}(\boldsymbol{\theta}) + \big(\boldsymbol{u}(\boldsymbol{z}_t^i;\boldsymbol{\theta})\boldsymbol{u}(\boldsymbol{z}_t^i;\boldsymbol{\theta})^\top\big)/(1-\gamma)$
end for
return $\hat{\boldsymbol{G}}(\boldsymbol{\theta}) := \hat{\boldsymbol{G}}(\boldsymbol{\theta})/M$

Algorithm 4 A Bayesian Actor-Critic Algorithm

BAC$(\boldsymbol{\theta}, M, \epsilon)$
- $\boldsymbol{\theta}$ initial policy parameters
- $M > 0$ episodes for gradient evaluation
- $\epsilon > 0$ termination threshold

done = false
while not done **do**
  Run **GPTD** for M episodes. **GPTD** returns $\boldsymbol{\alpha}_t$ and $\boldsymbol{C}_t$ (Equation 34)
  Compute an estimate of the Fisher information matrix $\hat{\boldsymbol{G}}_t$ (Equation 42) or $\hat{\boldsymbol{G}}(\boldsymbol{\theta})$ (Algorithm 3)
  Compute $\boldsymbol{U}_t$ (Proposition 6)
  $\Delta\boldsymbol{\theta} = \boldsymbol{U}_t\boldsymbol{\alpha}_t$ **(conventional gradient)** or
  $\Delta\boldsymbol{\theta} = \hat{\boldsymbol{G}}_t^{-1}\boldsymbol{U}_t\boldsymbol{\alpha}_t$ or $\Delta\boldsymbol{\theta} = \hat{\boldsymbol{G}}(\boldsymbol{\theta})^{-1}\boldsymbol{U}_t\boldsymbol{\alpha}_t$ **(natural gradient)**
  $\boldsymbol{\theta} := \boldsymbol{\theta} + \beta\Delta\boldsymbol{\theta}$
  if $|\Delta\boldsymbol{\theta}| < \epsilon$ **then** done = true
end while
return $\boldsymbol{\theta}$

calculating δ_i, $\tilde{\boldsymbol{k}}_{i-1}$ and $\tilde{\boldsymbol{K}}_{i-1}$ are the dictionary kernel vector and kernel matrix before observing $\boldsymbol{z}_i$, respectively. Engel et al. (2005) showed that using this sparsification procedure, the posteriors moments of Q may be compactly approximated as $\hat{Q}_t(\boldsymbol{z}) = \tilde{\boldsymbol{k}}_t^\top(\boldsymbol{z})\tilde{\boldsymbol{\alpha}}_t$ and $\hat{S}_t(\boldsymbol{z},\boldsymbol{z}') = k(\boldsymbol{z},\boldsymbol{z}') - \tilde{\boldsymbol{k}}_t^\top(\boldsymbol{z})\tilde{\boldsymbol{C}}_t\tilde{\boldsymbol{k}}_t(\boldsymbol{z}')$, where

$$\tilde{\boldsymbol{\alpha}}_t = \tilde{\boldsymbol{H}}_t^\top\left(\tilde{\boldsymbol{H}}_t\tilde{\boldsymbol{K}}_t\tilde{\boldsymbol{H}}_t^\top + \boldsymbol{\Sigma}_t\right)^{-1}\boldsymbol{r}_{t-1}\,, \qquad \tilde{\boldsymbol{C}}_t = \tilde{\boldsymbol{H}}_t^\top\left(\tilde{\boldsymbol{H}}_t\tilde{\boldsymbol{K}}_t\tilde{\boldsymbol{H}}_t^\top + \boldsymbol{\Sigma}_t\right)^{-1}\tilde{\boldsymbol{H}}_t\,. \quad (43)$$

In Equation 43, $\tilde{\boldsymbol{H}}_t = \boldsymbol{H}_t\boldsymbol{A}_t$, where $\boldsymbol{A}_t$ is a $|\mathcal{D}_t| \times |\tilde{\mathcal{D}}_t|$ matrix whose i'th row is $[\boldsymbol{A}]_{i,|\tilde{\mathcal{D}}_i|} = 1$ and $[\boldsymbol{A}]_{i,j} = 0\ ;\ \forall j \neq |\tilde{\mathcal{D}}_i|$, if we add the state-action pair $\boldsymbol{z}_i$ to the dictionary, and is $\tilde{\boldsymbol{k}}_{i-1}^\top(\boldsymbol{z}_i)\tilde{\boldsymbol{K}}_{i-1}^{-1}$ followed by zeros otherwise.

Proposition 7 *Using the sparsification method described above, the posterior moments of the gradient are approximated as*

$$\mathbf{E}\big[\nabla\eta(\boldsymbol{\theta})|\mathcal{D}_t\big] = \tilde{\boldsymbol{U}}_t\tilde{\boldsymbol{\alpha}}_t\,, \qquad \mathbf{Cov}\big[\nabla\eta(\boldsymbol{\theta})|\mathcal{D}_t\big] = \boldsymbol{G} - \tilde{\boldsymbol{U}}_t\tilde{\boldsymbol{C}}_t\tilde{\boldsymbol{U}}_t^\top\,,$$

where $\tilde{\boldsymbol{\alpha}}_t$ and $\tilde{\boldsymbol{C}}_t$ are given by Equation 43 and $\tilde{\boldsymbol{U}}_t = \big[\boldsymbol{u}(\boldsymbol{z}_1),\ldots,\boldsymbol{u}(\boldsymbol{z}_{|\tilde{\mathcal{D}}_t|})\big]$ with $\boldsymbol{z}_i \in \tilde{\mathcal{D}}_t$.

Proof The proof is straightforward by plugging the sparsified posterior mean and covariance of Q with $\tilde{\boldsymbol{\alpha}}_t$ and $\tilde{\boldsymbol{C}}_t$ from Equation 43 in Equation 35 and following the steps until the end of Proposition 6. ■

8. BAC Experimental Results

In this section, we empirically[17] evaluate the performance of the Bayesian actor-critic method presented in this paper in a 10-state random walk problem as well as in the widely used continuous-state-space mountain car problem (Sutton and Barto, 1998) and ship steering problem (Miller et al., 1990). In Section 8.1, we first compare BAC, Bayesian quadrature (BQ), and Monte Carlo (MC) gradient estimates in the 10-state random walk problem. We then evaluate the performance of the BAC algorithm on the same problem, and compare it with a Bayesian policy gradient (BPG) algorithm and a MC-based policy gradient (MCPG) algorithm. In Section 8.2, we compare the performance of the BAC algorithm with a MCPG algorithm on the mountain car problem. The BPG, BAC, and MCPG algorithms used in our experiments are Algorithms 2 and 4 presented in this paper, and Algorithm 1 in Baxter and Bartlett (2001), respectively. In Section 8.3, we compare the performance of the BAC algorithm with a MCPG algorithm on a problem in the ship steering domain. Similar to Section 8.2, the BAC, and MCPG algorithms used in our experiments are Algorithm 4 presented in this paper and Algorithm 1 in Baxter and Bartlett (2001), respectively.

17. The code for all the experiments of this section is available at `https://sequel.lille.inria.fr/Software/BAC`.

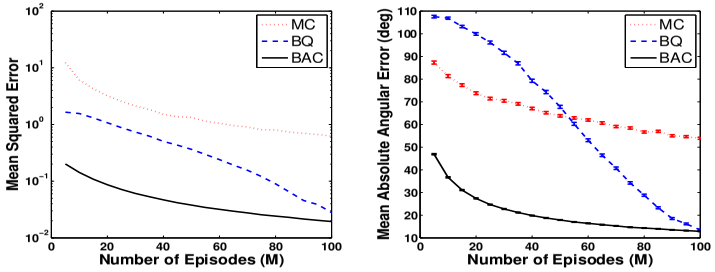


Figure 6: The mean squared error and the mean absolute angular error of MC, BQ, and BAC gradient estimations as a function of the number of sample episodes M. All results are averaged over 10^3 runs.

8.1 A Random Walk Problem

In this section, we consider a 10-state random walk problem, $\mathcal{X} = \{1,2,\ldots,10\}$, with states arranged linearly from state 1 on the left to state 10 on the right. The agent has two actions to choose from: $\mathcal{A} = \{left, right\}$. The left wall is a retaining barrier, meaning that if the $left$ action is taken at state 1, in the next time-step the state will be 1 again. State 10 is a zero reward absorbing state. The only stochasticity in the transitions is induced by the policy, which is defined as $\mu(right|x) = 1/1+\exp(-\theta_x)$ and $\mu(left|x) = 1-\mu(right|x)$, for all $x \in \mathcal{X}$. Note that each state x has an independent parameter θ_x. Each episode begins at state 1 and ends when the agent reaches state 10. The mean reward is 1 for states 1–9 and is 0 for state 10. The observed rewards for states 1–9 are obtained by corrupting the mean rewards with a 0.1 standard deviation i.i.d. Gaussian noise. The discount factor is $\gamma = 0.99$. In the BAC experiments, we use the Gaussian state kernel $k_x(x,x') = \exp(-||x-x'||^2/(2\sigma_k^2))$ with $\sigma_k = 3$ and the state-action kernel $0.01k_F(\boldsymbol{z},\boldsymbol{z}')$.

We first compare the MC, BQ, and BAC estimates of $\nabla\eta(\boldsymbol{\theta})$ for the policy induced by the parameters $\theta_x = \log(41/9)$ for all $x \in \mathcal{X}$, which is equivalent to $\mu(right|x) = 0.82$. We use several different sample sizes: $M = 5j,\ j = 1,\ldots,20$. Here, by sample size we mean the number of episodes used to estimate the gradient. For each value of M, we compute the gradient estimates 10^3 times. The true gradient is calculated analytically for reference. Figure 6 shows the mean squared error and the mean absolute angular error of MC, BQ, and BAC estimates of the gradient for different sample sizes M. The error bars in the right figure are the standard errors of the mean absolute angular errors. The results depicted in Figure 6 indicate that the BAC gradient estimates are more accurate and have lower variance than their MC and BQ counterparts.

Next, we use BAC to optimize the policy parameters and compare its performance with a BPG algorithm and a MCPG algorithm for $M = 1$, 25, 50, and 75. The BPG algorithm uses Model 1 of Section 4.1. We use Algorithm 4 with the number of policy updates set to

β	$M=1$	$M=25$	$M=50$	$M=75$
MCPG	0.005	0.075	0.1	0.75
BPG	0.0035	0.015	0.09	0.5
BAC	5	5	5	5

Table 4: Learning rates used by the algorithms in the experiments of Figure 7.

500 and the same kernels as in the previous experiment. The Fisher information matrix is estimated using Algorithm 3. The returns obtained by these methods are averaged over 10^3 runs. For a fixed sample size M, we tried many values of the learning rate, β, for MCPG, BPG, and BAC, and those in Table 4 yielded the best performance. Note that the learning rate used for each algorithm in each experiment is fixed and does not converge to zero. BAC showed a very robust performance when we changed the learning rate. By robust we mean that it never generated a policy for which an episode does not end after 10^6 steps. This seems to be due to the fact that BAC gradient estimates are more accurate and have less variance than their MC and BPG counterparts. The performance of BPG improves as we increase the sample size M. It performs worse than MCPG for $M=1$ and 25, but achieves a performance similar to BAC for $M=100$.

Figure 7 depicts the results of these experiments. From left to right and top to bottom the sub-figures correspond to the experiment in which all the algorithms used $M=1,\ 25,\ 50$, and 75 trajectories per policy update, respectively. Each curve depicts the difference between the exact average discounted return for the 500 policies that follow each policy update and η^* – the optimal average discounted return. All curves are averaged over 10^3 repetitions of the experiment. The BAC algorithm clearly learns significantly faster than the other algorithms (note that the vertical scale is logarithmic).

Remark: Since BQ (and as a result BPG) is based on defining a kernel over system trajectories (quadratic Fisher kernel in Model 1 and Fisher kernel in Model 2), its performance degrades when the system generates trajectories of different size. This effect can be observed by most kernels that have been used in the literature for the trajectories generated by dynamical systems. This can be also observed in our experiments: BQ performs much better than MC in the "Linear Quadratic Regulator" problem (Section 6.2), in which all the system trajectories are of size 20, while its superiority over MC is less apparent in the "Random Walk" problem (Section 8.1). This is why we are not going to use BQ and BPG in the "Mountain Car" (Section 8.2) and "Ship Steering" (Section 8.3) problems, in which the system trajectories have different lengths.

8.2 Mountain Car

In this section, we consider the mountain car problem as formulated in Sutton and Barto (1998), and report the results of applying the BAC and MCPG algorithms to optimize the policy parameters in this task. The state $\boldsymbol{x}$ consists of the position x and the velocity $\dot{x}$ of the car: $\boldsymbol{x}=(x,\dot{x})$. The reward is -1 on all time steps until the car reaches its goal at the top of the hill, which ends the episode. There are three possible actions: *forward*, *reverse*, and *zero*. The car moves according to the following simplified dynamics:

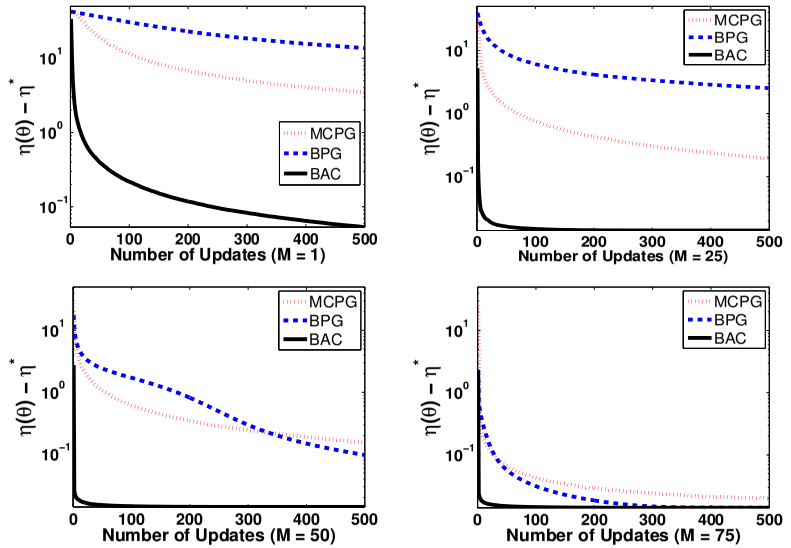


Figure 7: Results for the policy learning experiment. The graphs depict the performance of the policies learned by each algorithm during 500 policy updates. From left to right and top to bottom the number of episodes used to estimate the gradient is $M=1,\ 25,\ 50$ and 75. All results are averaged over 10^3 independent runs.

$$-1.2 \le x_{t+1} \le 0.5 \quad , \quad -0.07 \le \dot{x}_{t+1} \le 0.07 \ ,$$
$$x_{t+1} = \text{bound}[x_t + \dot{x}_{t+1}] \quad , \quad \dot{x}_{t+1} = \text{bound}\left[\dot{x}_t + 0.001 a_t - 0.0025\cos(3x_t)\right] .$$

When x_{t+1} reaches the left boundary, $\dot{x}_{t+1}$ is set to zero and when it reaches the right boundary, the goal is reached and the episode ends. Each episode starts from a random position and velocity uniformly sampled from their domains. We use the discount factor $\gamma=0.99$.

In order to define the policy, we first map the states $\boldsymbol{x}=(x,\dot{x})$ to the unit square $[0,1]\times[0,1]$. The policy used in our experiments has the following form:

$$\mu(a_i|\boldsymbol{x}) = \frac{\exp\left(\phi(\boldsymbol{x},a_i)^\top \boldsymbol{\theta}\right)}{\sum_{j=1}^3 \exp\left(\phi(\boldsymbol{x},a_j)^\top \boldsymbol{\theta}\right)} \ , \qquad i=1,2,3.$$

The policy feature vector is defined as $\phi(\boldsymbol{x},a_i) = \left(\phi(\boldsymbol{x})^\top \delta_{a_1 a_i}, \phi(\boldsymbol{x})^\top \delta_{a_2 a_i}, \phi(\boldsymbol{x})^\top \delta_{a_3 a_i}\right)^\top$, where $\delta_{a_j a_i}$ is 1 if $a_j = a_i$, and is 0 otherwise. The state feature vector $\phi(\boldsymbol{x})$ is composed of

β	$M = 5$	$M = 10$	$M = 20$	$M = 40$
MCPG	0.025 , ∞	0.1 , 100	0.2 , 100	0.25 , ∞
BAC	0.025 , ∞	0.05 , ∞	0.1 , ∞	0.1 , 250

Table 5: Learning rates used by the algorithms in the experiments of Figure 8.

16 Gaussian functions arranged in a 4×4 grid over the unit square as follows:

$$\phi(\boldsymbol{x}) = \Big(\exp\big(-||\boldsymbol{x} - \bar{\boldsymbol{x}}_1||^2/(2\kappa^2)\big), \ldots, \exp\big(-||\boldsymbol{x} - \bar{\boldsymbol{x}}_{16}||^2/(2\kappa^2)\big) \Big)^\top,$$

where the $\bar{\boldsymbol{x}}_i$'s are the 16 points of the grid $\{0, 0.25, 0.5, 1\} \times \{0, 0.25, 0.5, 1\}$ and $\kappa = 1.3 \times 0.25$.

In Figure 8, we compare the performance of BAC with a MCPG algorithm for $M = 5,\ 10,\ 20$, and 40 episodes used to estimate each gradient. For BAC, we use Algorithm 4 with the number of policy updates set to 500, a Gaussian state kernel $k_x(\boldsymbol{x}, \boldsymbol{x}') = \exp\big(-||\boldsymbol{x} - \boldsymbol{x}'||^2/(2\sigma_k^2)\big)$, with $\sigma_k = 1.3 \times 0.25$, and the state-action kernel $k_F(\boldsymbol{z}, \boldsymbol{z}')$. The Fisher information matrix is estimated using Algorithm 3. After every 50 policy updates the learned policy is evaluated for 10^3 episodes to estimate accurately the average number of steps to goal. Each evaluation episode starts from a random position and velocity uniformly chosen from their ranges, and continues until the car either reaches the goal or a limit of 200 time-steps is exceeded. The experiment is repeated 100 times for the entire horizontal axis to obtain average results and confidence intervals. The error bars in this figure are the standard errors of the performance of the algorithms.

For a fixed sample size M, each method starts with an initial learning rate and decreases it according to the schedule $\beta_t = \beta_0 \beta_c/(\beta_c + t)$. We tried many values of the learning rate parameters (β_0, β_c) for MCPG and BAC, and those in Table 5 yielded the best performance. Note that $\beta_c = \infty$ means that we used a fixed learning rate β_0 for that experiment. The graphs indicate that BAC performs better and has lower variance than MCPG. It is able to find a good policy with only $M = 5$ sample size and its performance does not change much as the sample size is increased. On the other hand, the performance of MCPG improves and its variance is reduced as we increase the sample size. Note that for $M = 40$, MCPG finally achieves a similar performance (still with slower rate) as BAC.

8.3 Ship Steering

In this section, we perform comparative experiments between BAC and MCPG on a more challenging problem in the continuous state continuous action *ship steering* domain (Miller et al., 1990).

Domain Description In this domain, a ship is located in a 150×150 meter square water surface. At any point in time t, the state of the ship is described by four continuous variables that are defined below along with their range of values

$$\mathbf{x}_t = (x_t, y_t, \theta_t, \dot{\theta}_t) \in [0\text{m}, 150\text{m}] \times [0\text{m}, 150\text{m}] \times [-180^\circ, 180^\circ] \times [-15^\circ/\text{s}, 15^\circ/\text{s}],$$

where x_t and y_t represent the position of the ship, θ_t the angle between the vertical axis and the ship orientation, and $\dot{\theta}_t$ the actual turning rate (see the upper-left panel in Figure 9).

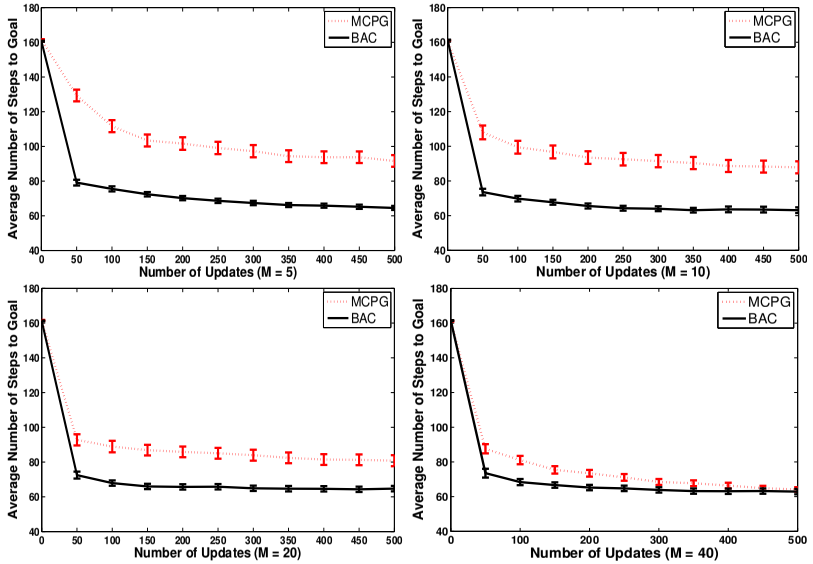


Figure 8: The graphs depict the performance of the policies learned by BAC and a MCPG algorithm during 500 policy updates in the mountain car problem. From left to right and top to bottom the number of episodes used to estimate the gradient is $M = 5,\ 10,\ 20$, and 40. All results are averaged over 100 independent runs.

At the beginning of each episode, the ship starts at $(x_1, y_1) = (40\text{m}, 40\text{m})$, with θ_1 and $\dot{\theta}_1$ sampled uniformly at random from their ranges. The only available action variable is $a_t \in [-15^\circ, 15^\circ]$, which is the *desired* turning rate. To model the ship inertia and water resistance, there is a $T = 5$ time steps lag for the desired turning rate to become the actual turning rate. Moreover, the ship moves with the constant speed of $V = 3\text{m/s}$ and $\Delta = 0.2\text{s}$ is the sampling interval. The following set of equations summarizes the ship's dynamics:

$$x_{t+1} = x_t + \Delta\, V \sin\theta_t \qquad y_{t+1} = y_t + \Delta\, V \cos\theta_t$$
$$\theta_{t+1} = \theta_t + \Delta\, \dot{\theta}_t \qquad \dot{\theta}_{t+1} = \dot{\theta}_t + \frac{\Delta}{T}(a_t - \dot{\theta}_t)$$

The goal of the ship is to navigate to $(x_*, y_*) = (100\text{m}, 100\text{m})$ within 500 times steps. If this does not happen or the ship moves out of the boundary, the episode terminates as a failure. The goal of the policy is to maximize the probability of the ship successfully reaching (x_*, y_*). Thus, we set the discount factor to $\gamma = 1$ in this problem.

β	$M = 5$	$M = 10$	$M = 20$
MCPG	0.01	0.01	0.01
BAC	0.5	0.4	0.5

Table 6: Learning rates used by the algorithms in the experiments of Figure 9.

Learning For both MCPG and BAC, we used a CMAC function approximator with 9 four dimensional tilings, each of them discretizing the state space into $5 \times 5 \times 36 \times 5 = 4500$ tiles. Therefore, each policy parameter $\boldsymbol{w}$ is of size $N = 9 \times 4500 = 40500$. Each state $\mathbf{x}$ is represented by a binary vector $\phi(\mathbf{x})$, where $\phi_i(\mathbf{x}) = 1$ if and only if the state $\mathbf{x}$ falls in the ith tile, and thus, $\sum_{i=1}^{N} \phi_i(\mathbf{x}) \leq 9$. To define a precise mapping from states to actions, $\boldsymbol{w}_t : \mathbf{x_t} = (x_t, y_t, \theta_t, \dot{\theta}_t) \rightarrow a_t$, we first sample $\tilde{a}_t$ from the Gaussian

$$\tilde{a}_t \sim \mathcal{N}\left(\frac{\sum_{i=1}^{N} \boldsymbol{w}_t^{(i)} \phi_i(\mathbf{x}_t)}{\sum_{i=1}^{N} \phi_i(\mathbf{x}_t)}, 1\right),$$

and then map it to the allowed range $[-15°, 15°]$ using the sigmoid transformation

$$a_t = 15° \cdot \frac{2}{\pi} \cdot \arctan\left(\frac{\pi}{2} \cdot \tilde{a}_t\right).$$

For the BAC experiments, we used the Gaussian state kernel $k_x(\mathbf{x}, \mathbf{x}') = \exp(-||\mathbf{x} - \mathbf{x}'||^2/(2\sigma_k^2))$, with $\sigma_k = 1$ and the state-action kernel $k_F(\boldsymbol{z}, \boldsymbol{z}')$, i.e., the Fisher kernel.

Setup In order to improve the computational efficiency, we use several numerical approximations. First, to calculate the score function for a trajectory (Equation 5) in both MCPG and BAC, we approximate the gradient of the action distribution in a_t with the one in $\tilde{a}_t$, i.e.,

$$\nabla \log \mu(a_t|\mathbf{x}_t; \boldsymbol{w}_t) \approx \nabla \log \mu(\tilde{a}_t|\mathbf{x}_t; \boldsymbol{w}_t).$$

Second, we calculate the gradient using the online sparsification procedure described in Section 4.4. Finaly, we never explicitly calculate the inverse of the Fisher information matrix $\hat{\boldsymbol{G}}$ and instead calculate the product of $\hat{\boldsymbol{G}}^{-1}$ with the score. For the numerical stability we also add 10^{-6} to the diagonal of $\hat{\boldsymbol{G}}$.

Similar to the other experiments in the paper, we varied the number of trajectories used to estimate the gradient of a policy as $M = 5$, 10, and 20. Table 6 shows the best values of the learning rate β for both MCPG and BAC for different values of M. To evaluate each method, we ran 100 independent learning trials. At each trial, we evaluate the performance of the policy every 100 iterations by executing it 100 (independent) times with θ_1 and $\dot{\theta}_1$ randomly sampled. For each of these execution, we observe if the ship reached (x_*, y_*) within 500 steps and estimate the policy success ratio. We set the total number of gradient updates to $T = 3000$ for $M = 5$ and 10 and to $T = 1000$ for $M = 20$.

Results The results for all the experiments are presented in Figure 9 along with their standard deviations. Naturally, using more trajectories for the gradient update improves both methods. However, this improvement is bigger for the BAC method. In the case of $M = 5$, MCPG produces slightly better policies at the beginning of learning, but is

soon outperformed by BAC. For $M = 10$ and 20, BAC produces better policies from the beginning, especially for $M = 20$. This is consistent with the results of the other experimental domains reported in the paper. For all values of M, BAC converges to a policy with a better success ratio than MCPG. Finally, as expected, BAC has usually less variance in its performance than MCPG.

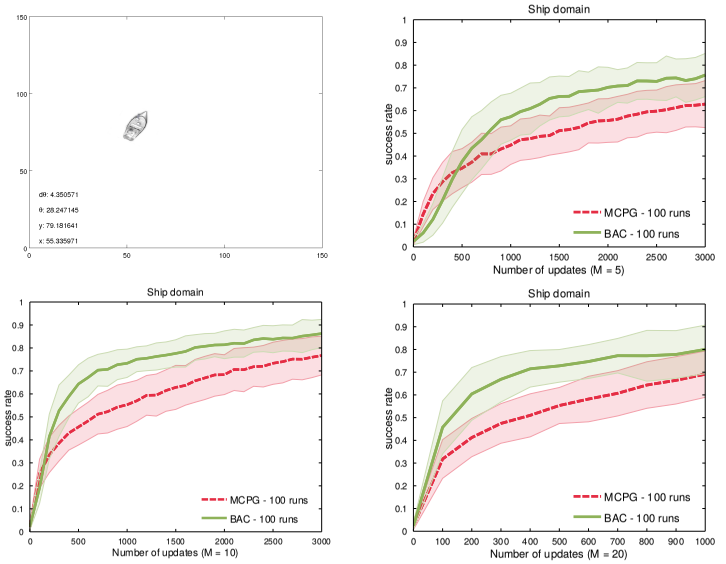


Figure 9: Success rate of the policies learned by BAC and MCPG in the ship steering problem.

9. Other Advancements in Bayesian Reinforcement Learning

The algorithms presented in this paper belong to the class of *Bayesian model-free RL*, as they do not assume that the system's dynamic is known and do not explicitly construct a model of the system. In recent years, Bayesian methodology has been used to develop algorithms in several other areas of RL. In this section, we provide a brief overview of these results (for more details, see the survey by Ghavamzadeh et al. 2015).

Another widely-used class of RL algorithms are those that build an explicit model of the system and use it to find a good (or optimal) policy, thus, are known as *model-based RL* algorithms. Recent years have witnessed many applications of the Bayesian methodology to this class of RL algorithms. The main idea of model-based Bayesian RL is to explicitly

maintain a posterior over the model parameters and to use it to select actions in order to appropriately balance exploration and exploitation. The class of model-based Bayesian RL algorithms include those that work with MDPs and those that work with POMDPs (e.g., Ross et al. 2008, Doshi et al. 2008). The MDP-based algorithms can be further divided to those that are offline (e.g., Duff 2001, Poupart et al. 2006), those that are online (e.g., Dearden et al. 1999, Strens 2000, Wang et al. 2005, Ross et al. 2008), and those that have probably approximately correct (PAC)-guarantees (e.g., Kolter and Ng 2009, Asmuth et al. 2009, Sorg et al. 2010).

The use of Bayesian methodology has also been explored to solve the *inverse RL* (IRL) problem, i.e., learning the underlying model of the decision-making agent (expert) from its observed behavior and the dynamics of the system (Russell, 1998). The main idea of Bayesian IRL (BIRL) is to use a prior to encode the reward preference and to formulate the compatibility with the expert's policy as a likelihood in order to derive a probability distribution over the space of reward functions, from which the expert's reward function is somehow extracted. The most notable works in the area of BIRL include those by Ramachandran and Amir (2007), Choi and Kim (2011, 2012), Michini and How (2012a,b).

Bayesian techniques have also been used to derive algorithms for the *collaborative multi-agent RL* problem. When dealing with multi-agent systems, the complexity of the decision problem is increased in the following way: while single-agent BRL requires maintaining a posterior over the MDP parameters (in the case of model-based methods) or over the value/policy (in the case of model-free methods), in multi-agent BRL, it is also necessary to keep a posterior over the policies of the other agents. Chalkiadakis and Boutilier (2013) showed that this belief can be maintained in a tractable manner subject to certain structural assumptions on the domain, for example that the strategies of the agents are independent of each other.

Multi-task RL (MTRL) is another area that has witnessed the application of Bayesian methodology. All approaches to MTRL assume that the tasks share similarity in some components of the problem such as dynamics, reward structure, or value function. The Bayesian MTRL methods assume that the shared components are drawn from a common generative model (Wilson et al., 2007, Mehta et al., 2008, Lazaric and Ghavamzadeh, 2010). In Mehta et al. (2008), tasks share the same dynamics and reward features, and only differ in the weights of the reward function. The proposed method initializes the value function for a new task using the previously learned value functions as a prior. Wilson et al. (2007) and Lazaric and Ghavamzadeh (2010) both assume that the distribution over some components of the tasks is drawn from a hierarchical Bayesian model.

Bayesian learning methods have also been used for regret minimization in multi-armed bandits. This area that goes back to the seminal work of Gittins (1979), has become very active with the Bayesian version of the upper confidence bound (UCB) algorithm (Kaufmann et al., 2012a) and the recent advancements in the analysis of Thompson Sampling (Agrawal and Goyal, 2012, Kaufmann et al., 2012b, Agrawal and Goyal, 2013a,b, Russo and Van Roy, 2014, Gopalan et al., 2014, Guha and Munagala, 2014, Liu and Li, 2015) and its state-of-the-art empirical performance (Scott, 2010, Chapelle and Li, 2011), which has also led to its use in several industrial applications (Graepel et al., 2010, Tang et al., 2013).

10. Discussion

In this paper, we first proposed an alternative approach to the conventional frequentist (Monte-Carlo based) policy gradient estimation procedure. Our approach is based on *Bayesian quadrature* (O'Hagan, 1991), a Bayesian method for integral evaluation. The idea is to model the gradient of the expected return with respect to the policy parameters, which is of the form of an integral, as Gaussian processes (GPs). This is done by dividing the integrand into two parts, treating one as a random function (or random field), whose random nature reflects our subjective uncertainty concerning its true identity. This allows us to incorporate our prior knowledge of this term (part) into its prior distribution. Observing (possibly noisy) samples of this term allows us to employ Bayes' rule to compute a posterior distribution of it conditioned on these samples. This in turn induces a posterior distribution over the value of the integral, which is the gradient of the expected return. By properly partitioning the integrand and by appropriately selecting a prior distribution, a closed-form expression for the posterior moments of the gradient of the expected return is obtained. We proposed two different ways of partitioning the integrand resulting in two distinct Bayesian models. For each model, we showed how the posterior moments of the gradient conditioned on the observed data are calculated. In line with previous work on Bayesian quadrature, our Bayesian approach tends to significantly reduce the number of samples needed to obtain accurate gradient estimates. Moreover, estimates of the natural gradient and the gradient covariance are provided at little extra cost. We performed detailed experimental comparisons of the Bayesian policy gradient (BPG) algorithms presented in the paper with classic Monte-Carlo based algorithms on a bandit problem as well as on a linear quadratic regulator problem. The experimental results are encouraging, but we conjecture that even better gains may be attained using this approach. This calls for additional theoretical and empirical work. It is important to note that the gradient estimated by Algorithm 1 may be employed in conjunction with conjugate-gradients and line-search methods for making better use of the gradient information. We also showed that the models and algorithms presented in this paper can be extended to partially observable problems without any change along the same lines as Baxter and Bartlett (2001). This is due to the fact that our BPG framework considers complete system trajectories as its basic observable unit, and thus, does not require the dynamic within each trajectory to be of any special form. This generality has the downside that our proposed framework cannot take advantage of the Markov property when the system is Markovian.

To address this issue, we then extended our BPG framework to actor-critic algorithms and presented a new Bayesian take on the actor-critic architecture. By using GPs and choosing their prior distributions to make them compatible with a parametric family of policies, we were able to derive closed-form expressions for the posterior distribution of the policy gradient updates. The posterior mean is used to update the policy and the posterior covariance to gauge the reliability of this update. Our Bayesian actor-critic (BAC) framework uses individual state-action-reward transitions as its basic observable unit, and thus, is able to take advantage of the Markov property of the system trajectories (when the system is indeed Markovian). This improvement seems to be borne out in our experiments, where BAC provides more accurate estimates of the policy gradient than either of the two BPG models for the same amount of data. Similar to BPG, another feature of BAC

is that its natural-gradient variant is obtained at little extra cost. For both BPG and BAC, we derived the sparse form of the algorithms, which would make them significantly more time and memory efficient. Finally, we performed an experimental evaluation of the BAC algorithm, comparing it with classic Monte-Carlo based policy gradient algorithms, as well as our BPG algorithms, on a random walk problem, the widely used mountain car problem (Sutton and Barto, 1998), and the continuous state and continuous action ship steering domain (Miller et al., 1990).

Additional experimental work is required to investigate the behavior of BPG and BAC algorithms in larger and more realistic domains, involving continuous and high-dimensional state and action spaces. The BPG and BAC algorithms proposed in the paper use only the posterior mean of the gradient in their updates. We conjecture that the second-order statistics obtained from BPG and BAC (both in the actor and critic) may be used to devise more efficient algorithms. In one of the experiments in Section 6, we employed the covariance information provided by Algorithm 1 for risk-aware selection of the step size in the gradient updates, which showed promising performance. Other interesting directions for future work include **1)** investigating other possible partitions of the integrand in the expression for $\nabla\eta_B(\boldsymbol{\theta})$ into a GP term and a deterministic term, **2)** using other types of kernel functions such as sequence kernels, **3)** combining our approach with MDP model estimation to allow transfer of learning between different policies (model-based Bayesian policy gradient), and **4)** investigating more efficient methods for estimating the Fisher information matrix. Another direction is to derive a fully non-parametric actor-critic algorithm. In BAC, the critic is based on Gaussian process temporal difference learning, which is a non-parametric method, while the actor uses a family of parameterized policies. The idea here would be to replace the actor in the BAC algorithm with a non-parametric actor that performs gradient search in a function space (e.g., a reproducing kernel Hilbert space) of policies.

Acknowledgments

Part of the computational experiments was conducted using the Grid'5000 experimental testbed (https://www.grid5000.fr). Yaakov Engel was supported by an Alberta Ingenuity fellowship.

Appendix A. Proof of Proposition 3

We start the proof with the $M \times 1$ vector $\boldsymbol{b}$, whose ith element can be written as

$$
\begin{aligned}
(\boldsymbol{b})_i &= \int k(\xi,\xi_i)\Pr(\xi;\boldsymbol{\theta})d\xi \\
&\overset{(a)}{=} \int \left(1+\boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)\right)^2 \Pr(\xi;\boldsymbol{\theta})d\xi \\
&\overset{(b)}{=} \int \Pr(\xi;\boldsymbol{\theta})d\xi + 2\left(\int \boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi\right)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i) + \int \boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)\Pr(\xi;\boldsymbol{\theta})d\xi \\
&\overset{(c)}{=} 1 + \left(\boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\right)\left(\int \boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top \Pr(\xi;\boldsymbol{\theta})d\xi\right)\left(\boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)\right) \\
&\overset{(d)}{=} 1 + \left(\boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\right)\boldsymbol{G}\left(\boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)\right) \overset{(e)}{=} 1 + \boldsymbol{u}(\xi_i)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)
\end{aligned}
$$

(a) substitutes $k(\xi,\xi_i)$ with the quadratic Fisher kernel from Equation 23, **(b)** is algebra, **(c)** follows from **(i)** $\int \Pr(\xi;\boldsymbol{\theta})d\xi = 1$, and **(ii)** $\int \boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi = \int \nabla \log \Pr(\xi;\boldsymbol{\theta})\Pr(\xi;\boldsymbol{\theta})d\xi$ $= \int \nabla \Pr(\xi;\boldsymbol{\theta})d\xi = \nabla \int \Pr(\xi;\boldsymbol{\theta})d\xi = \nabla(1) = 0$, **(d)** is the result of replacing the integral with the Fisher information matrix $\boldsymbol{G}$, **(e)** is algebra, and thus, the claim follows.

Now the proof for the scalar b_0

$$
\begin{aligned}
b_0 &= \iint k(\xi,\xi')\Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi' \\
&\overset{(a)}{=} \iint \left(1+\boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi')\right)^2 \Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi' \\
&\overset{(b)}{=} \iint \Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi' + 2\iint \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi')\Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi' \\
&+ \iint \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi')\boldsymbol{u}(\xi')^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi' \\
&\overset{(c)}{=} 1 + 2\left(\int \boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi\right)^\top \boldsymbol{G}^{-1}\left(\int \boldsymbol{u}(\xi')\Pr(\xi';\boldsymbol{\theta})d\xi'\right) \\
&+ \int \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\left(\int \boldsymbol{u}(\xi')\boldsymbol{u}(\xi')^\top \Pr(\xi';\boldsymbol{\theta})d\xi'\right)\boldsymbol{G}^{-1}\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi \\
&\overset{(d)}{=} 1 + \int \boldsymbol{u}(\xi)^\top \boldsymbol{G}^{-1}\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi
\end{aligned} \tag{44}
$$

(a) substitutes $k(\xi,\xi')$ with the quadratic Fisher kernel from Equation 23, **(b)** is algebra, **(c)** follows from **(i)** $\iint \Pr(\xi;\boldsymbol{\theta})\Pr(\xi';\boldsymbol{\theta})d\xi d\xi' = 1$, and **(ii)** $\int \boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi = 0$, and finally **(d)** is the result of replacing the integral within the parentheses with the Fisher information matrix $\boldsymbol{G}$.

The Fisher information matrix $\boldsymbol{G}$ is positive definite and symmetric. Thus, it can be written as $\boldsymbol{G} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top$, where $\boldsymbol{V} = [\boldsymbol{v}_1,\ldots,\boldsymbol{v}_n]$ and $\boldsymbol{\Lambda} = \mathrm{diag}[\lambda_1,\ldots,\lambda_n]$ are the matrix of orthonormal eigenvectors and the diagonal matrix of eigenvalues of matrix $\boldsymbol{G}$, respectively.

By replacing $\boldsymbol{G}^{-1}$ with $\boldsymbol{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{V}^\top$ in Equation 44 we obtain

$$
\begin{aligned}
b_0 &= 1 + \int \boldsymbol{u}^\top(\xi)\boldsymbol{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{V}^\top\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi \\
&\overset{\text{(a)}}{=} 1 + \int \left(\boldsymbol{V}^\top\boldsymbol{u}(\xi)\right)^\top \boldsymbol{\Lambda}^{-1}\left(\boldsymbol{V}^\top\boldsymbol{u}(\xi)\right)\Pr(\xi;\boldsymbol{\theta})d\xi \overset{\text{(b)}}{=} 1 + \int \left(\sum_{i=1}^n \lambda_i^{-1}\left(\boldsymbol{V}^\top\boldsymbol{u}(\xi)\right)_i^2\right)\Pr(\xi;\boldsymbol{\theta})d\xi \\
&\overset{\text{(c)}}{=} 1 + \sum_{i=1}^n \lambda_i^{-1}\left(\int \left(\boldsymbol{v}_i^\top\boldsymbol{u}(\xi)\right)^2\Pr(\xi;\boldsymbol{\theta})d\xi\right) \overset{\text{(d)}}{=} 1 + \sum_{i=1}^n \lambda_i^{-1}\left(\int \boldsymbol{v}_i^\top\boldsymbol{u}(\xi)\boldsymbol{v}_i^\top\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi\right) \\
&\overset{\text{(e)}}{=} 1 + \sum_{i=1}^n \lambda_i^{-1}\left(\int \boldsymbol{v}_i^\top\boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top\boldsymbol{v}_i\Pr(\xi;\boldsymbol{\theta})d\xi\right) \overset{\text{(f)}}{=} 1 + \sum_{i=1}^n \lambda_i^{-1}\boldsymbol{v}_i^\top\left(\int \boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top\Pr(\xi;\boldsymbol{\theta})d\xi\right)\boldsymbol{v}_i \\
&\overset{\text{(g)}}{=} 1 + \sum_{i=1}^n \lambda_i^{-1}\boldsymbol{v}_i^\top\boldsymbol{G}\boldsymbol{v}_i \overset{\text{(h)}}{=} 1 + \sum_{i=1}^n \lambda_i^{-1}\boldsymbol{v}_i^\top\lambda_i\boldsymbol{v}_i = 1 + \sum_{i=1}^n \boldsymbol{v}_i^\top\boldsymbol{v}_i = 1 + \sum_{i=1}^n ||\boldsymbol{v}_i||^2 \overset{\text{(i)}}{=} 1 + n
\end{aligned}
$$

(a) and **(b)** are algebra, **(c)** is the result of switching the sum and the integral, **(d)** is algebra, **(e)** follows from the fact that $\boldsymbol{v}_i^\top\boldsymbol{u}(\xi)$ is a scalar, and thus, can be replaced by its transpose, **(f)** is algebra, **(g)** substitutes the integral within the parentheses with the Fisher information matrix $\boldsymbol{G}$, **(h)** replaces $\boldsymbol{G}\boldsymbol{v}_i$ with $\lambda_i\boldsymbol{v}_i$, **(i)** follows from the orthonormality of $\boldsymbol{v}_i$'s, and thus, the claim follows.

Appendix B. Proof of Proposition 4

We start with the proof of $\boldsymbol{B}$. This $n \times M$ matrix may be written as

$$
\begin{aligned}
\boldsymbol{B} &= \int \nabla\Pr(\xi;\boldsymbol{\theta})\boldsymbol{k}(\xi)^\top d\xi = \int \nabla\Pr(\xi;\boldsymbol{\theta})\left[k(\xi,\xi_1),\ldots,k(\xi,\xi_M)\right]d\xi \\
&\overset{\text{(a)}}{=} \int \nabla\Pr(\xi;\boldsymbol{\theta})\left[\boldsymbol{u}(\xi)^\top\boldsymbol{G}^{-1}\boldsymbol{u}(\xi_1),\ldots,\boldsymbol{u}(\xi)^\top\boldsymbol{G}^{-1}\boldsymbol{u}(\xi_M)\right]d\xi \\
&\overset{\text{(b)}}{=} \left(\int \nabla\Pr(\xi;\boldsymbol{\theta})\boldsymbol{u}(\xi)^\top d\xi\right)\boldsymbol{G}^{-1}\left[\boldsymbol{u}(\xi_1),\ldots,\boldsymbol{u}(\xi_M)\right] \\
&\overset{\text{(c)}}{=} \left(\int \boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top\Pr(\xi;\boldsymbol{\theta})d\xi\right)\boldsymbol{G}^{-1}\left[\boldsymbol{u}(\xi_1),\ldots,\boldsymbol{u}(\xi_M)\right] \\
&\overset{\text{(d)}}{=} \boldsymbol{G}\boldsymbol{G}^{-1}\left[\boldsymbol{u}(\xi_1),\ldots,\boldsymbol{u}(\xi_M)\right] \overset{\text{(e)}}{=} \left[\boldsymbol{u}(\xi_1),\ldots,\boldsymbol{u}(\xi_M)\right] = \boldsymbol{U}
\end{aligned}
$$

(a) substitutes $k(\xi,\xi_i)$ with the Fisher kernel from Equation 27, **(b)** is algebra, **(c)** follows from $\nabla\Pr(\xi;\boldsymbol{\theta}) = \boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})$, **(d)** substitutes the integral within the parentheses with the Fisher information matrix $\boldsymbol{G}$, **(e)** is algebra, and thus, the claim follows.

Now the proof for the $n \times n$ matrix $\boldsymbol{B}_0$

$$
\begin{aligned}
\boldsymbol{B}_0 &= \iint k(\xi,\xi')\nabla\Pr(\xi;\boldsymbol{\theta})\nabla\Pr(\xi';\boldsymbol{\theta})^\top d\xi d\xi' \\
&\overset{\text{(a)}}{=} \iint \nabla\Pr(\xi;\boldsymbol{\theta})k(\xi,\xi')\nabla\Pr(\xi';\boldsymbol{\theta})^\top d\xi d\xi' \\
&\overset{\text{(b)}}{=} \iint \left(\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})\right)\boldsymbol{u}(\xi)^\top\boldsymbol{G}^{-1}\boldsymbol{u}(\xi')\left(\boldsymbol{u}(\xi')\Pr(\xi';\boldsymbol{\theta})\right)^\top d\xi d\xi' \\
&\overset{\text{(c)}}{=} \left(\int \boldsymbol{u}(\xi)\boldsymbol{u}(\xi)^\top\Pr(\xi;\boldsymbol{\theta})d\xi\right)\boldsymbol{G}^{-1}\left(\int \boldsymbol{u}(\xi')\boldsymbol{u}(\xi')^\top\Pr(\xi';\boldsymbol{\theta})d\xi'\right) \overset{\text{(d)}}{=} \boldsymbol{G}\boldsymbol{G}^{-1}\boldsymbol{G} = \boldsymbol{G}
\end{aligned}
$$

(a) follows from the fact that $k(\xi,\xi')$ is scalar, **(b)** substitutes $k(\xi,\xi')$ with the Fisher information kernel from Equation 27 and $\nabla\Pr(\xi;\boldsymbol{\theta})$ with $\boldsymbol{u}(\xi)\Pr(\xi;\boldsymbol{\theta})$, **(c)** is algebra, **(d)** is the result of substituting the integrals within the parentheses with the Fisher information matrix $\boldsymbol{G}$, and thus, the claim follows.

Appendix C. Proof of Proposition 5

Here we only show the proof for Model 1, the proof for Model 2 is straightforward following the same arguments. The sparse approximations of the kernel matrix $\boldsymbol{K}$ and kernel vector $\boldsymbol{k}(\cdot)$ may be written as $\boldsymbol{K} \approx \boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top$ and $\boldsymbol{k}(\cdot) \approx \boldsymbol{A}\tilde{\boldsymbol{k}}(\cdot)$, respectively (Equations. 2.2.8 and 2.2.9 in Engel, 2005). If we replace $\boldsymbol{K}$ and $\boldsymbol{k}(\cdot)$ in Equation 21 with their sparse approximations, we obtain

$$
\begin{aligned}
\mathbf{E}\left[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\right] &= \boldsymbol{Y}\left(\boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top + \boldsymbol{\Sigma}\right)^{-1}\boldsymbol{b}, \\
\mathbf{Cov}\left[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\right] &= \left[b_0 - \boldsymbol{b}^\top\left(\boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top + \boldsymbol{\Sigma}\right)^{-1}\boldsymbol{b}\right]\boldsymbol{I}.
\end{aligned} \tag{45}
$$

Sparsification does not change b_0 and it remains equal to $n+1$ (see Proposition 3), however it modifies $\boldsymbol{b}$ to

$$
\boldsymbol{b} = \int \boldsymbol{k}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi = \int \boldsymbol{A}\tilde{\boldsymbol{k}}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi = \boldsymbol{A}\int \tilde{\boldsymbol{k}}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi = \boldsymbol{A}\tilde{\boldsymbol{b}},
$$

where $\tilde{\boldsymbol{b}} = \int \tilde{\boldsymbol{k}}(\xi)\Pr(\xi;\boldsymbol{\theta})d\xi$ is exactly $\boldsymbol{b}$, only the kernel vector $\boldsymbol{k}(\cdot)$ has been replaced by the sparse kernel vector $\tilde{\boldsymbol{k}}(\cdot)$. Thus using Proposition 3, we have $(\tilde{\boldsymbol{b}})_i = 1 + \boldsymbol{u}(\xi_i)^\top\boldsymbol{G}^{-1}\boldsymbol{u}(\xi_i)$, with $\xi_i \in \tilde{\mathcal{D}}$. By replacing $\boldsymbol{b}$ with $\boldsymbol{A}\tilde{\boldsymbol{b}}$ in Equation 45, we obtain

$$
\begin{aligned}
\mathbf{E}\left[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\right] &= \boldsymbol{Y}\left(\boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top + \boldsymbol{\Sigma}\right)^{-1}\boldsymbol{A}\tilde{\boldsymbol{b}} = \boldsymbol{Y}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1} + \boldsymbol{I}\right)^{-1}\boldsymbol{A}\tilde{\boldsymbol{b}}, \\
\mathbf{Cov}\left[\nabla\eta_B(\boldsymbol{\theta})|\mathcal{D}_M\right] &= \left(b_0 - \tilde{\boldsymbol{b}}^\top\boldsymbol{A}^\top\left(\boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top + \boldsymbol{\Sigma}\right)^{-1}\boldsymbol{A}\tilde{\boldsymbol{b}}\right)\boldsymbol{I} = \left(b_0 - \tilde{\boldsymbol{b}}^\top\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{A}\tilde{\boldsymbol{K}}\boldsymbol{A}^\top\boldsymbol{\Sigma}^{-1} + \boldsymbol{I}\right)^{-1}\boldsymbol{A}\tilde{\boldsymbol{b}}\right)\boldsymbol{I}.
\end{aligned}
$$

The claim follows using Lemma 1.3.2 in Engel (2005).

Appendix D. Proof of Proposition 6

We start the proof with the $n \times (t+1)$ matrix $\boldsymbol{B}_t$, whose ith column may be written as

$$
\begin{aligned}
(\boldsymbol{B}_t)_i &= \int_{\mathcal{Z}} d\boldsymbol{z}\boldsymbol{g}(\boldsymbol{z};\boldsymbol{\theta})k(\boldsymbol{z},\boldsymbol{z}_i) = \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\nabla\log\mu(a|x;\boldsymbol{\theta})\Big(k_x(x,x_i) + k_F(\boldsymbol{z},\boldsymbol{z}_i)\Big) \\
&= \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\nabla\log\mu(a|x;\boldsymbol{\theta})k_x(x,x_i) + \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\nabla\log\mu(a|x;\boldsymbol{\theta})k_F(\boldsymbol{z},\boldsymbol{z}_i) \\
&= \int_{\mathcal{X}} dx\nu^\mu(x)k_x(x,x_i)\int_{\mathcal{A}} da\mu(a|x;\boldsymbol{\theta})\nabla\log\mu(a|x;\boldsymbol{\theta}) + \int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\boldsymbol{u}(\boldsymbol{z})\boldsymbol{u}(\boldsymbol{z})^\top\boldsymbol{G}^{-1}\boldsymbol{u}(\boldsymbol{z}_i) \\
&= \int_{\mathcal{X}} dx\nu^\mu(x)k_x(x,x_i)\int_{\mathcal{A}} da\nabla\mu(a|x;\boldsymbol{\theta}) + \left(\int_{\mathcal{Z}} d\boldsymbol{z}\pi^\mu(\boldsymbol{z})\boldsymbol{u}(\boldsymbol{z})\boldsymbol{u}(\boldsymbol{z})^\top\right)\boldsymbol{G}^{-1}\boldsymbol{u}(\boldsymbol{z}_i) \\
&= \int_{\mathcal{X}} dx\nu^\mu(x)k_x(x,x_i)\nabla\left(\int_{\mathcal{A}} da\mu(a|x;\boldsymbol{\theta})\right) + \boldsymbol{G}\boldsymbol{G}^{-1}\boldsymbol{u}(\boldsymbol{z}_i) \\
&= \int_{\mathcal{X}} dx\nu^\mu(x)k_x(x,x_i)\nabla(1) + \boldsymbol{u}(\boldsymbol{z}_i) = \boldsymbol{u}(\boldsymbol{z}_i)
\end{aligned}
$$

The 1st line follows from the definition of matrix $\boldsymbol{B}_t$, function $\boldsymbol{g}$, and kernel k, the 2nd line is algebra, the 3rd line follows from the definition of π^μ and the Fisher kernel k_F, the 4th line is algebra, the 5th line is the result of replacing the integral in the parentheses with the Fisher information matrix $\boldsymbol{G}$, finally the 6th line is algebra, and the claim follows.

Now the proof for the $n \times n$ matrix $\boldsymbol{B}_0$

$$
\begin{aligned}
\boldsymbol{B}_0 &= \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \boldsymbol{g}(\boldsymbol{z};\boldsymbol{\theta}) k(\boldsymbol{z},\boldsymbol{z}') \boldsymbol{g}(\boldsymbol{z}';\boldsymbol{\theta})^\top \\
&\overset{\text{(a)}}{=} \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \pi^\mu(\boldsymbol{z}) \nabla \log \mu(a|x;\boldsymbol{\theta}) \Big(k_x(x,x') + k_F(\boldsymbol{z},\boldsymbol{z}') \Big) \nabla \log \mu(a'|x';\boldsymbol{\theta})^\top \pi^\mu(\boldsymbol{z}') \\
&\overset{\text{(b)}}{=} \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \pi^\mu(\boldsymbol{z}) \nabla \log \mu(a|x;\boldsymbol{\theta}) k_x(x,x') \nabla \log \mu(a'|x';\boldsymbol{\theta})^\top \pi^\mu(\boldsymbol{z}') \\
&+ \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \pi^\mu(\boldsymbol{z}) \nabla \log \mu(a|x;\boldsymbol{\theta}) k_F(\boldsymbol{z},\boldsymbol{z}') \nabla \log \mu(a'|x';\boldsymbol{\theta})^\top \pi^\mu(\boldsymbol{z}') \\
&\overset{\text{(c)}}{=} \int_{\mathcal{X}^2} dx dx' \nu^\mu(x) \nu^\mu(x') k_x(x,x') \int_{\mathcal{A}^2} da da' \mu(a|x;\boldsymbol{\theta}) \nabla \log \mu(a|x;\boldsymbol{\theta}) \nabla \log \mu(a'|x';\boldsymbol{\theta})^\top \mu(a'|x';\boldsymbol{\theta}) \\
&+ \int_{\mathcal{Z}^2} d\boldsymbol{z} d\boldsymbol{z}' \pi^\mu(\boldsymbol{z}) \boldsymbol{u}(\boldsymbol{z}) \boldsymbol{u}(\boldsymbol{z})^\top \boldsymbol{G}^{-1} \boldsymbol{u}(\boldsymbol{z}') \boldsymbol{u}(\boldsymbol{z}')^\top \pi^\mu(\boldsymbol{z}') \\
&\overset{\text{(d)}}{=} \int_{\mathcal{X}^2} dx dx' \nu^\mu(x) \nu^\mu(x') k_x(x,x') \int_{\mathcal{A}} da \nabla \mu(a|x;\boldsymbol{\theta}) \int_{\mathcal{A}} da' \nabla \mu(a'|x';\boldsymbol{\theta})^\top \\
&+ \left(\int_{\mathcal{Z}} d\boldsymbol{z} \pi^\mu(\boldsymbol{z}) \boldsymbol{u}(\boldsymbol{z}) \boldsymbol{u}(\boldsymbol{z})^\top \right) \boldsymbol{G}^{-1} \left(\int_{\mathcal{Z}} d\boldsymbol{z}' \pi^\mu(\boldsymbol{z}') \boldsymbol{u}(\boldsymbol{z}') \boldsymbol{u}(\boldsymbol{z}')^\top \right) = \boldsymbol{G} \boldsymbol{G}^{-1} \boldsymbol{G} = \boldsymbol{G}
\end{aligned}
$$

(a) follows from the definition of function $\boldsymbol{g}$ and kernel k, **(b)** is algebra, **(c)** follows from the definition of π^μ and the Fisher kernel k_F, **(c)** is algebra, finally **(d)** follows from $\int_{\mathcal{A}} da \nabla \mu(a|x;\boldsymbol{\theta}) = 0$ and $\boldsymbol{G} = \int_{\mathcal{Z}} d\boldsymbol{z} \pi^\mu(\boldsymbol{z}) \boldsymbol{u}(\boldsymbol{z}) \boldsymbol{u}(\boldsymbol{z})^\top$, and the claim follows.

References

D. Aberdeen and J. Baxter. Policy-gradient learning of controllers with internal state. Technical report, Australian National University, 2001.

S. Agrawal and N. Goyal. Analysis of Thompson sampling for the multi-armed bandit problem. In *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23, pages 39.1 – 39.26, 2012.

S. Agrawal and N. Goyal. Further optimal regret bounds for Thompson sampling. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, pages 99–107, 2013a.

S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th International Conference on Machine Learning*, pages 127–135, 2013b.

V. Aleksandrov, V. Sysoyev, and V. Shemeneva. Stochastic optimization. *Engineering Cybernetics*, 5:11–16, 1968.

J. Asmuth, L. Li, M. Littman, A. Nouri, and D. Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2009.

K. Astrom. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

J. Bagnell and J. Schneider. Covariant policy search. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 2003.

L. Baird. Advantage updating. Technical Report WL-TR-93-1146, Wright Laboratory, 1993.

A. Barto, R. Sutton, and C. Anderson. Neuron-like elements that can solve difficult learning control problems. *IEEE Transaction on Systems, Man and Cybernetics*, 13:835–846, 1983.

J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

J. Baxter, P. Bartlett, and L. Weaver. Experiments with infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:351–381, 2001.

J. Berger and R. Wolpert. *The Likelihood Principle*. Institute of Mathematical Statistics, Hayward, CA, 1984.

D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee. Incremental natural actor-Critic algorithms. In *Proceedings of Advances in Neural Information Processing Systems 20*, pages 105–112, 2007.

S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

G. Chalkiadakis and C. Boutilier. Coordination in multiagent reinforcement learning: A Bayesian approach. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, 2013.

O. Chapelle and L. Li. An empirical evaluation of Thompson sampling. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 2249–2257, 2011.

J. Choi and K. Kim. Map inference for Bayesian inverse reinforcement learning. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1989–1997, 2011.

J. Choi and K. Kim. Nonparametric Bayesian inverse reinforcement learning for multiple reward functions. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 305–313, 2012.

L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14:641–668, 2002.

R. Dearden, N. Friedman, and D. Andre. Model based Bayesian exploration. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 1999.

F. Doshi, J. Pineau, and N. Roy. Reinforcement learning with limited reinforcement: using Bayes risk for active learning in POMDPs. In *International Conference on Machine Learning*, 2008.

M. Duff. Monte-Carlo algorithms for the improvement of finite-state stochastic controllers: Application to Bayes-adaptive Markov decision processes. In *International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2001.

P. Dyer and S. McReynolds. *The Computation and Theory of Optimal Control*. Academic Press, 1970.

Y. Engel. *Algorithms and Representations for Reinforcement Learning*. PhD thesis, The Hebrew University of Jerusalem, Israel, 2005.

Y. Engel, S. Mannor, and R. Meir. Sparse online greedy support vector regression. In *Proceedings of the Thirteenth European Conference on Machine Learning*, pages 84–96, 2002.

Y. Engel, S. Mannor, and R. Meir. Bayes meets Bellman: The Gaussian process approach to temporal difference learning. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 154–161, 2003.

Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with Gaussian processes. In *Proceedings of the Twenty Second International Conference on Machine Learning*, pages 201–208, 2005.

M. Ghavamzadeh and Y. Engel. Bayesian policy gradient algorithms. In *Proceedings of Advances in Neural Information Processing Systems 19*, pages 457–464, 2006.

M. Ghavamzadeh and Y. Engel. Bayesian Actor-Critic algorithms. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, pages 297–304, 2007.

M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar. Bayesian reinforcement learning: A survey. *Foundations and Trends in Machine Learning*, 8(5-6):359–483, 2015.

J. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177, 1979.

P. Glynn. Stochastic approximation for Monte Carlo optimization. In *Proceedings of the Winter Simulation Conference*, pages 356–365, 1986.

P. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33:75–84, 1990.

P. Glynn and P. L'Ecuyer. Likelihood ratio gradient estimation for regenerative stochastic recursions. *Advances in Applied Probability*, 27(4):1019–1053, 1995.

A. Gopalan, S. Mannor, and Y. Mansour. Thompson sampling for complex online problems. In *Proceedings of the 31st International Conference on Machine Learning*, pages 100–108, 2014.

T. Graepel, J.Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proceedings of the 27th International Conference on Machine Learning*, pages 13–20, 2010.

E. Greensmith, P. Bartlett, and J. Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

S. Guha and K. Munagala. Stochastic regret minimization via Thompson sampling. In *Proceedings of The 27th Conference on Learning Theory*, pages 317–338, 2014.

V. Gullapalli. A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural Networks*, 3(6):671–692, 1990.

V. Gullapalli. Learning control under extreme uncertainty. In *Proceedings of Advances in Neural Information Processing Systems 4*, pages 327–334, 1992.

V. Gullapalli, J. Franklin, and H. Benbrahim. Acquiring robot skills via reinforcement learning. *IEEE Control Systems*, 4(1):13–24, 1994.

L. Hasdorff. *Gradient Optimization and Nonlinear Control*. John Wiley & Sons, 1976.

T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Proceedings of Advances in Neural Information Processing Systems 11*, 1999.

D. Jacobson and D. Mayne. *Differential Dynamic Programming*. American Elsevier Publishing Company, 1970.

L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

S. Kakade. A natural policy gradient. In *Proceedings of Advances in Neural Information Processing Systems 14*, 2002.

E. Kaufmann, O. Cappé, and A. Garivier. On Bayesian upper confidence bounds for bandit problems. In *International Conference on Artificial Intelligence and Statistics*, pages 592–600, 2012a.

E. Kaufmann, N. Korda, and R. Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *Algorithmic Learning Theory*, volume 7568 of *Lecture Notes in Computer Science*, pages 199–213, 2012b.

H. Kimura, M. Yamamura, and S. Kobayashi. Reinforcement learning by stochastic hill-climbing on discounted reward. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 295–303, 1995.

J. Kolter and A. Ng. Near-Bayesian exploration in polynomial time. In *International Conference on Machine Learning*, 2009.

V. Konda and J. Tsitsiklis. Actor-Critic algorithms. In *Proceedings of Advances in Neural Information Processing Systems 12*, pages 1008–1014, 2000.

A. Lazaric and M. Ghavamzadeh. Bayesian multi-task reinforcement learning. In *Proceedings of the 27th International Conference on Machine Learning*, pages 599–606, 2010.

C. Liu and L. Li. On the prior sensitivity of Thompson sampling. *CoRR*, abs/1506.03378, 2015.

P. Marbach. *Simulated-Based Methods for Markov Decision Processes*. PhD thesis, Massachusetts Institute of Technology, 1998.

N. Mehta, S. Natarajan, P. Tadepalli, and A. Fern. Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning*, 73(3):289–312, 2008.

B. Michini and J. How. Bayesian nonparametric inverse reinforcement learning. In *Proceedings of the European Conference on Machine Learning*, 2012a.

B. Michini and J. How. Improving the efficiency of Bayesian inverse reinforcement learning. In *IEEE International Conference on Robotics and Automation*, pages 3651–3656, 2012b.

W. Miller, R. Sutton, and P. Werbos. *Neural Networks for Control*. MIT Press, 1990.

A. O'Hagan. Monte-Carlo is fundamentally unsound. *The Statistician*, 36:247–249, 1987.

A. O'Hagan. Bayes-Hermite quadrature. *Journal of Statistical Planning and Inference*, 29: 245–260, 1991.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

J. Peters, S. Vijayakumar, and S. Schaal. Reinforcement learning for humanoid robotics. In *Proceedings of the Third IEEE-RAS International Conference on Humanoid Robots*, 2003.

J. Peters, S. Vijayakumar, and S. Schaal. Natural actor-critic. In *Proceedings of the Sixteenth European Conference on Machine Learning*, pages 280–291, 2005.

H. Poincaré. *Calcul des Probabilités*. Georges Carré, Paris, 1896.

P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement learning. In *International Conference on Machine learning*, pages 697–704, 2006.

M. Puterman. *Markov Decision Processes*. Wiley Interscience, 1994.

D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2586–2591, 2007.

C. Rasmussen and Z. Ghahramani. Bayesian Monte Carlo. In *Proceedings of Advances in Neural Information Processing Systems 15*, pages 489–496, 2003.

C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

M. Reiman and A. Weiss. Sensitivity analysis via likelihood ratios. In *Proceedings of the Winter Simulation Conference*, 1986.

M. Reiman and A. Weiss. Sensitivity analysis for simulations via likelihood ratios. *Operations Research*, 37, 1989.

S. Ross, B. Chaib-draa, and J. Pineau. Bayes-adaptive POMDPs. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 20, pages 1225–1232, 2008.

R. Rubinstein. *Some Problems in Monte Carlo Optimization*. PhD thesis, Polytechnic Institute, Riga, Latvia, 1969.

G. Rummery and M. Niranjan. *On-line Q-learning using Connectionist Systems*. Technical Report CUED/F-INFENG/TR 166, Engineering Department, Cambridge University, 1994.

S. Russell. Learning agents for uncertain environments (extended abstract). In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 101–103, 1998.

D. Russo and B. Van Roy. An information-theoretic analysis of Thompson sampling. *CoRR*, abs/1403.5341, 2014.

S. Scott. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

R. Smallwood and E. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21(5):1071–1088, 1973.

V. Sorg, S. Sing, and R. Lewis. Variance-based rewards for approximate Bayesian reinforcement learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2010.

M. Strens. A Bayesian framework for reinforcement learning. In *International Conference on Machine Learning*, 2000.

R. Sutton. *Temporal credit assignment in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 1984.

R. Sutton and A. Barto. *An Introduction to Reinforcement Learning*. MIT Press, 1998.

R. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of Advances in Neural Information Processing Systems 12*, pages 1057–1063, 2000.

L. Tang, R. Rosales, A. Singh, and D. Agarwal. Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1587–1594. ACM, 2013.

N. Vien, H. Yu, and T. Chung. Hessian matrix distribution for Bayesian policy gradient reinforcement learning. *Information Sciences*, 181(9):1671–1685, 2011.

T. Wang, D. Lizotte, M. Bowling, and D. Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *International Conference on Machine learning*, pages 956–963, 2005.

L. Weaver and N. Tao. The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth International Conference on Uncertainty in Artificial Intelligence*, pages 538–545, 2001.

R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

A. Wilson, A. Fern, S. Ray, and P. Tadepalli. Multi-task reinforcement learning: A hierarchical bayesian approach. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, pages 1015–1022, 2007.

Journal of Machine Learning Research 17 (2016) 1-70 Submitted 3/13; Revised 12/14; Published 4/16

Practical Kernel-Based Reinforcement Learning

André M. S. Barreto AMSB@LNCC.BR
Laboratório Nacional de Computação Científica
Petrópolis, Brazil

Doina Precup DPRECUP@CS.MCGILL.CA
Joelle Pineau JPINEAU@CS.MCGILL.CA
School of Computer Science
McGill University
Montreal, Canada

Editor: George Konidaris

Abstract

Kernel-based reinforcement learning (KBRL) stands out among approximate reinforcement learning algorithms for its strong theoretical guarantees. By casting the learning problem as a local kernel approximation, KBRL provides a way of computing a decision policy which converges to a unique solution and is statistically consistent. Unfortunately, the model constructed by KBRL grows with the number of sample transitions, resulting in a computational cost that precludes its application to large-scale or on-line domains. In this paper we introduce an algorithm that turns KBRL into a practical reinforcement learning tool. *Kernel-based stochastic factorization* (KBSF) builds on a simple idea: when a transition probability matrix is represented as the product of two stochastic matrices, one can swap the factors of the multiplication to obtain another transition matrix, potentially much smaller than the original, which retains some fundamental properties of its precursor. KBSF exploits such an insight to compress the information contained in KBRL's model into an approximator of fixed size. This makes it possible to build an approximation considering both the difficulty of the problem and the associated computational cost. KBSF's computational complexity is linear in the number of sample transitions, which is the best one can do without discarding data. Moreover, the algorithm's simple mechanics allow for a fully incremental implementation that makes the amount of memory used independent of the number of sample transitions. The result is a kernel-based reinforcement learning algorithm that can be applied to large-scale problems in both off-line and on-line regimes. We derive upper bounds for the distance between the value functions computed by KBRL and KBSF using the same data. We also prove that it is possible to control the magnitude of the variables appearing in our bounds, which means that, given enough computational resources, we can make KBSF's value function as close as desired to the value function that would be computed by KBRL using the same set of sample transitions. The potential of our algorithm is demonstrated in an extensive empirical study in which KBSF is applied to difficult tasks based on real-world data. Not only does KBSF solve problems that had never been solved before, but it also significantly outperforms other state-of-the-art reinforcement learning algorithms on the tasks studied.

Keywords: reinforcement learning, dynamic programming, Markov decision processes, kernel-based approximation, stochastic factorization

1. Introduction

Reinforcement learning provides a conceptual framework with the potential to materialize a long-sought goal in artificial intelligence: the construction of situated agents that learn how to behave from direct interaction with the environment (Sutton and Barto, 1998). But such an endeavor does not come without its challenges; among them, extrapolating the field's basic machinery to large-scale domains has been a particularly persistent obstacle.

It has long been recognized that virtually any real-world application of reinforcement learning must involve some form of approximation. Given the mature stage of the supervised-learning theory, and considering the multitude of approximation techniques available today, this realization may not come across as a particularly worrisome issue at first glance. However, it is well known that the sequential nature of the reinforcement learning problem renders the incorporation of function approximators non-trivial (Bertsekas and Tsitsiklis, 1996).

Despite the difficulties, in the last two decades the collective effort of the reinforcement learning community has given rise to many reliable approximate algorithms (Szepesvári, 2010). Among them, Ormoneit and Sen's (2002) kernel-based reinforcement learning (KBRL) stands out for two reasons. First, unlike other approximation schemes, KBRL always converges to a unique solution. Second, KBRL is consistent in the statistical sense, meaning that adding more data improves the quality of the resulting policy and eventually leads to optimal performance.

Unfortunately, the good theoretical properties of KBRL come at a price: since the model constructed by this algorithm grows with the number of sample transitions, the cost of computing a decision policy quickly becomes prohibitive as more data become available. Such a computational burden severely limits the applicability of KBRL. This may help explain why, in spite of its nice theoretical guarantees, kernel-based learning has not been widely adopted as a practical reinforcement learning tool.

This paper presents an algorithm that can potentially change this situation. *Kernel-based stochastic factorization* (KBSF) builds on a simple idea: when a transition probability matrix is represented as the product of two stochastic matrices, one can swap the factors of the multiplication to obtain another transition matrix, potentially much smaller than the original, which retains some fundamental properties of its precursor (Barreto and Fragoso, 2011). KBSF exploits this insight to compress the information contained in KBRL's model into an approximator of fixed size. Specifically, KBSF builds a model, whose size is independent of the number of sample transitions, which serves as an approximation of the model that would be constructed by KBRL. Since the size of the model becomes a parameter of the algorithm, KBSF essentially detaches the structure of KBRL's approximator from its configuration. This extra flexibility makes it possible to build an approximation that takes into account *both* the difficulty of the problem *and* the computational cost of finding a policy using the constructed model.

KBSF's computational complexity is linear in the number of sample transitions, which is the best one can do without throwing data away. Moreover, we show in the paper that the amount of memory used by our algorithm is independent of the number of sample transitions. Put together, these two properties make it possible to apply KBSF to large-scale problems in both off-line and on-line regimes. To illustrate this possibility in practice,

we present an extensive empirical study in which KBSF is applied to difficult control tasks based on real-world data, some of which had never been solved before. KBSF outperforms least-squares policy iteration and fitted Q-iteration on several off-line problems and SARSA on a difficult on-line task.

We also show that KBSF is a sound algorithm from a theoretical point of view. Specifically, we derive results bounding the distance between the value function computed by our algorithm and the one computed by KBRL using the same data. We also prove that it is possible to control the magnitude of the variables appearing in our bounds, which means that we can make the difference between KBSF's and KBRL's solutions arbitrarily small.

We start the paper presenting some background material in Section 2. Then, in Section 3, we introduce the stochastic-factorization trick, the insight underlying the development of our algorithm. KBSF itself is presented in Section 4. This section is divided in two parts, one theoretical and one practical. In Section 4.1 we present theoretical results showing that not only is the difference between KBSF's and KBRL's value functions bounded, but it can also be controlled. Section 4.2 brings experiments with KBSF on four reinforcement-learning problems: single and double pole-balancing, HIV drug schedule domain, and epilepsy suppression task. In Section 5 we introduce the incremental version of our algorithm, which can be applied to on-line problems. This section follows the same structure of Section 4, with theoretical results followed by experiments. Specifically, in Section 5.1 we extend the results of Section 4.1 to the on-line scenario, and in Section 5.2 we present experiments on the triple pole-balancing and helicopter tasks. In Section 6 we take a closer look at the approximation computed by KBSF and present a practical guide on how to configure our algorithm to solve a reinforcement learning problem. In Section 7 we summarize related works and situate KBSF in the context of kernel-based learning. Finally, in Section 8 we present the main conclusions regarding the current research and discuss some possibilities of future work.

The paper has three appendices. Appendix A has the proofs of our theoretical results. The details of the experiments that were omitted in the main body of the text are described in Appendix B. In Appendix C we provide a table with the main symbols used in the paper that can be used as a reference to facilitate reading.

Parts of the material presented in this article have appeared before in two papers published in the *Neural Information Processing Systems* conference (NIPS, Barreto et al., 2011, 2012). The current manuscript is a substantial extension of the aforementioned works.

2. Background

We consider the standard framework of reinforcement learning, in which an *agent* interacts with an *environment* and tries to maximize the amount of *reward* collected in the long run (Sutton and Barto, 1998). The interaction between agent and environment happens at discrete time steps: at each instant t the agent occupies a state $s_{(t)} \in S$ and must choose an action a from a finite set A. The sets S and A are called the *state* and *action spaces*, respectively. The execution of action a in state $s_{(t)}$ moves the agent to a new state $s_{(t+1)}$, where a new action must be selected, and so on. Each transition has a certain probability of occurrence and is associated with a reward $r \in \mathbb{R}$. The goal of the agent is to find a *policy* $\pi : S \mapsto A$, that is, a mapping from states to actions, that maximizes the expected

return. Here we define the *return* from time t as:

$$R_{(t)} = r_{(t+1)} + \gamma r_{(t+2)} + \gamma^2 r_{(t+3)} + ... = \sum_{i=1}^{T} \gamma^{i-1} r_{(t+i)}, \quad (1)$$

where $r_{(t+1)}$ is the reward received at the transition from state $s_{(t)}$ to state $s_{(t+1)}$. The interaction of the agent with the environment may last forever ($T = \infty$) or until the agent reaches a terminal state ($T < \infty$); each sequence of interactions is usually referred to as an *episode*. The parameter $\gamma \in [0, 1)$ is the *discount factor*, which determines the relative importance of individual rewards depending on how far in the future they are received.

2.1 Markov Decision Processes

As usual, we assume that the interaction between agent and environment can be modeled as a *Markov decision process* (MDP, Puterman, 1994). An MDP is a tuple $M \equiv (S, A, P^a, R^a, \gamma)$, where P^a and R^a describe the dynamics of the task at hand. For each action $a \in A$, $P^a(\cdot|s)$ defines the next-state distribution upon taking action a in state s. The reward received at transition $s \xrightarrow{a} s'$ is given by $R^a(s, s')$, with $|R^a(s, s')| \leq R_{\max} < \infty$. Usually, one is interested in the expected reward resulting from the execution of action a in state s, that is, $r^a(s) = E_{s' \sim P^a(\cdot|s)}\{R^a(s, s')\}$.

Once the interaction between agent and environment has been modeled as an MDP, a natural way of searching for an optimal policy is to resort to *dynamic programming* (Bellman, 1957). Central to the theory of dynamic-programming is the concept of a *value function*. The value of state s under a policy π, denoted by $V^\pi(s)$, is the expected return the agent will receive from s when following π, that is, $V^\pi(s) = E_\pi\{R_{(t)}|s_{(t)} = s\}$ (here the expectation is over all possible sequences of rewards in (1) when the agent follows π). Similarly, the value of the state-action pair (s, a) under policy π is defined as $Q^\pi(s, a) = E_{s' \sim P^a(\cdot|s)}\{R^a(s, s') + \gamma V^\pi(s')\} = r^a(s) + \gamma E_{s' \sim P^a(\cdot|s)}\{V^\pi(s')\}$.

The notion of value function makes it possible to impose a partial ordering over decision policies. In particular, a policy π' is considered to be at least as good as another policy π if $V^{\pi'}(s) \geq V^\pi(s)$ for all $s \in S$. The goal of dynamic programming is to find an optimal policy π^* that performs no worse than any other. It is well known that there always exists at least one such policy for a given MDP (Puterman, 1994). When there is more than one optimal policy, they all share the same value function V^*.

When both the state and action spaces are finite, an MDP can be represented in matrix form: each function P^a becomes a matrix $\mathbf{P}^a \in \mathbb{R}^{|S| \times |S|}$, with $p^a_{ij} = P^a(s_j|s_i)$, and each function r^a becomes a vector $\mathbf{r}^a \in \mathbb{R}^{|S|}$, where $r^a_i = r^a(s_i)$. Similarly, V^π can be represented as a vector $\mathbf{v}^\pi \in \mathbb{R}^{|S|}$ and Q^π can be seen as a matrix $\mathbf{Q}^\pi \in \mathbb{R}^{|S| \times |A|}$. [1]

When the MDP is finite, dynamic programming can be used to find an optimal decision-policy $\pi^* \in A^{|S|}$ in time polynomial in the number of states $|S|$ and actions $|A|$ (Ye, 2011).

1. Throughout the paper we will use the conventional and matrix notations interchangeably, depending on the context. When using the latter, vectors will be denoted by small boldface letters and matrices will be denoted by capital boldface letters. We will also use the same notation for all MDPs and associated components, distinguishing between them through the use of math accents. So, for example, if $\bar{M}$ is an MDP, its transition functions and matrices will be referred to as $\bar{P}^a$ and $\bar{\mathbf{P}}^a$, its expect-reward functions and vectors will be denoted by $\bar{r}^a$ and $\bar{\mathbf{r}}^a$, its optimal decision policy will be $\bar{\pi}^*$, and so on (see Table 2).

Let $\mathbf{v} \in \mathbb{R}^{|S|}$ and let $\mathbf{Q} \in \mathbb{R}^{|S|\times|A|}$. Define the operator $\Gamma : \mathbb{R}^{|S|\times|A|} \mapsto \mathbb{R}^{|S|}$ such that $\Gamma\mathbf{Q} = \mathbf{v}$ if and only if $v_i = \max_j q_{ij}$ for all i. Also, given an MDP M, define $\Delta : \mathbb{R}^{|S|} \mapsto \mathbb{R}^{|S|\times|A|}$ such that $\Delta\mathbf{v} = \mathbf{Q}$ if and only if $q_{ia} = r_i^a + \gamma \sum_{j=1}^{|S|} p_{ij}^a v_j$ for all i and all a. The *Bellman operator* of the MDP M is given by $T \equiv \Gamma\Delta$. A fundamental result in dynamic programming states that, starting from $\mathbf{v}^{(0)} = \mathbf{0}$, the expression $\mathbf{v}^{(t)} = T\mathbf{v}^{(t-1)} = \Gamma\mathbf{Q}^{(t)}$ gives the optimal t-step value function, and as $t \to \infty$ the vector $\mathbf{v}^{(t)}$ approaches $\mathbf{v}^*$. At any point, the optimal t-step policy can be obtained by selecting $\pi_i^{(t)} \in \text{argmax}_j\, q_{ij}^{(t)}$ (Puterman, 1994).

In contrast with dynamic programming, in reinforcement learning it is assumed that the MDP is unknown, and the agent must learn a policy based on transitions sampled from the environment. If the process of learning a decision policy is based on a fixed set of sample transitions, we call it *batch* reinforcement learning. On the other hand, in *on-line* reinforcement learning the computation of a decision policy takes place concomitantly with the collection of data (Sutton and Barto, 1998).

2.2 Kernel-Based Reinforcement Learning

Kernel-based reinforcement learning (KBRL) is a batch algorithm that uses a finite model approximation to solve a continuous MDP $M \equiv (\mathbb{S}, A, P^a, R^a, \gamma)$, where $\mathbb{S} \subseteq [0,1]^{d_\mathbb{S}}$ and $d_\mathbb{S} \in \mathbb{N}_*^+$ is the dimension of the state space (Ormoneit and Sen, 2002). Let $S^a \equiv \{(s_k^a, r_k^a, \hat{s}_k^a) | k = 1, 2, ..., n_a\}$ be sample transitions associated with action $a \in A$, where $s_k^a, \hat{s}_k^a \in \mathbb{S}$ and $r_k^a \in \mathbb{R}$. Let $\phi : \mathbb{R}^+ \mapsto \mathbb{R}^+$ be a Lipschitz continuous function satisfying $\int_0^1 \phi(x)dx = 1$. Let $\mathrm{k}_\tau(s, s')$ be a kernel function defined as

$$\mathrm{k}_\tau(s, s') = \phi\left(\frac{\|s - s'\|}{\tau}\right), \tag{2}$$

where $\tau \in \mathbb{R}$ and $\|\cdot\|$ is a norm in $\mathbb{R}^{d_\mathbb{S}}$ (for concreteness, the reader may think of $\mathrm{k}_\tau(s, s')$ as the Gaussian kernel, although the definition also encompasses other functions). Finally, define the normalized kernel function associated with action a as

$$\kappa_\tau^a(s, s_i^a) = \frac{\mathrm{k}_\tau(s, s_i^a)}{\sum_{j=1}^{n_a} \mathrm{k}_\tau(s, s_j^a)}. \tag{3}$$

KBRL uses (3) to build a finite MDP whose state space $\hat{S}$ is composed solely of the $n = \sum_a n_a$ states $\hat{s}_i^a$. We assume without loss of generality that the action space A is ordered and the sampled states are ordered lexicographically as $\hat{s}_i^a < \hat{s}_j^b \iff a < b$ or $(a = b$ and $i < j)$; if a given state $s \in \mathbb{S}$ occurs more than once in the set of sample transitions, each occurrence will be treated as a distinct state in the finite MDP. The transition functions of KBRL's model, $\hat{P}^a : \hat{S} \times \hat{S} \mapsto [0, 1]$, are given by:

$$\hat{P}^a\left(\hat{s}_i^b | s\right) = \begin{cases} \kappa_\tau^a(s, s_i^b), & \text{if } a = b, \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where $a, b \in A$. Similarly, the reward functions of the MDP constructed by KBRL, $\hat{R}^a : \hat{S} \times \hat{S} \mapsto \mathbb{R}$, are

$$\hat{R}^a(s, \hat{s}_i^b) = \begin{cases} r_i^a, & \text{if } a = b, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Based on (4) and (5) we can define the transition matrices and expected-reward vectors of KBRL's MDP. The matrices $\hat{\mathbf{P}}^a$ are derived directly from the definition of $\hat{P}^a(\hat{s}_i^b|s)$, replacing s with the sampled states $\hat{s}_i^a$ (see Figure 2a). The vectors $\hat{\mathbf{r}}^a$ are computed as follows. Let $\mathbf{r} \equiv [(\mathbf{r}^1)^\intercal, (\mathbf{r}^2)^\intercal, ..., (\mathbf{r}^{|A|})^\intercal]^\intercal \in \mathbb{R}^n$, where $\mathbf{r}^a \in \mathbb{R}^{n_a}$ are the vectors composed of the sample rewards, that is, the i^{th} element of $\mathbf{r}^a$ is $r_i^a \in S^a$. Since $R^a(s, \hat{s}_i^b)$ does not depend on the start state s, we can write

$$\hat{\mathbf{r}}^a = \hat{\mathbf{P}}^a\mathbf{r}. \tag{6}$$

KBRL's MDP is thus given by $\hat{M} \equiv (\hat{S}, A, \hat{\mathbf{P}}^a, \hat{\mathbf{r}}^a, \gamma)$.

Once $\hat{M}$ has been defined, one can use dynamic programming to compute its optimal value function $\hat{V}^*$. Then, the value of any state-action pair of the continuous MDP can be determined as:

$$\hat{Q}(s, a) = \sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a)\left[r_i^a + \gamma\hat{V}^*(\hat{s}_i^a)\right], \tag{7}$$

where $s \in \mathbb{S}$ and $a \in A$. Ormoneit and Sen (2002) have shown that, if $n_a \to \infty$ for all $a \in A$ and the kernel's width τ shrink at an "admissible" rate, the probability of choosing a suboptimal action based on $\hat{Q}(s, a)$ converges to zero (see their Theorem 4).

As discussed, using dynamic programming one can compute the optimal value function of $\hat{M}$ in time polynomial in the number of sample transitions n (which is also the number of states in $\hat{M}$). However, since each application of the Bellman operator $\hat{T}$ is $O(n^2|A|)$, the computational cost of such a procedure can easily become prohibitive in practice. Thus, the use of KBRL leads to a dilemma: on the one hand one wants as much data as possible to describe the dynamics of the task, but on the other hand the number of transitions should be small enough to allow for the numerical solution of the resulting model. In the following sections we describe a practical approach to weigh the relative importance of these two conflicting objectives.

3. Stochastic Factorization

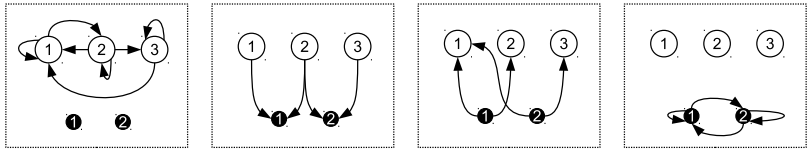
A *stochastic matrix* has only nonnegative elements and each of its rows sums to 1. That said, we can introduce the concept that will serve as a cornerstone for the rest of the paper:

Definition 1 *Given a stochastic matrix $\mathbf{P} \in \mathbb{R}^{n\times p}$, the relation $\mathbf{P} = \mathbf{DK}$ is called a* stochastic factorization *of $\mathbf{P}$ if $\mathbf{D} \in \mathbb{R}^{n\times m}$ and $\mathbf{K} \in \mathbb{R}^{m\times p}$ are also stochastic matrices. The integer $m > 0$ is the* order *of the factorization.*

This mathematical construct has been explored before. For example, Cohen and Rothblum (1991) briefly discuss it as a special case of nonnegative matrix factorization, while Cutler and Breiman (1994) study slightly modified versions of stochastic factorization for statistical data analysis. However, in this paper we will focus on a useful property of this type of factorization that has only recently been noted (Barreto and Fragoso, 2011).

3.1 Stochastic-Factorization Trick

Let $\mathbf{P} \in \mathbb{R}^{n\times n}$ be a transition matrix, that is, a square stochastic matrix, and let $\mathbf{P} = \mathbf{DK}$ be an order m stochastic factorization. In this case, one can see the elements of



$$\mathbf{P} = \begin{bmatrix} \times & \times & 0 \\ \times & \times & \times \\ \times & 0 & \times \end{bmatrix} \quad \mathbf{D} = \begin{bmatrix} \times & 0 \\ \times & \times \\ 0 & \times \end{bmatrix} \quad \mathbf{K} = \begin{bmatrix} \times & \times & 0 \\ \times & 0 & \times \end{bmatrix} \quad \bar{\mathbf{P}} = \begin{bmatrix} \times & \times \\ \times & \times \end{bmatrix}$$

Figure 1: Reducing the dimension of a transition model from $n = 3$ states to $m = 2$ artificial states. Original states s_i are represented as big white circles; small black circles depict artificial states $\bar{s}_h$. The symbol '$\times$' is used to represent nonzero elements. These figures have appeared before in the article by Barreto and Fragoso (2011).

$\mathbf{D}$ and $\mathbf{K}$ as probabilities of transitions between the states s_i and a set of m artificial states $\bar{s}_h$. Specifically, the elements in each row of $\mathbf{D}$ can be interpreted as probabilities of transitions from the original states to the artificial states, while the rows of $\mathbf{K}$ can be seen as probabilities of transitions in the opposite direction. Under this interpretation, each element $p_{ij} = \sum_{h=1}^{m} d_{ih}k_{hj}$ is the sum of the probabilities associated with m two-step transitions: from state s_i to each artificial state $\bar{s}_h$ and from these back to state s_j. In other words, p_{ij} is the accumulated probability of all possible paths from s_i to s_j with a stopover in one of the artificial states $\bar{s}_h$. Following similar reasoning, it is not difficult to see that by *swapping* the factors of a stochastic factorization, that is, by switching from $\mathbf{DK}$ to $\mathbf{KD}$, one obtains the transition probabilities between the artificial states $\bar{s}_h$, $\bar{\mathbf{P}} = \mathbf{KD}$. If $m < n$, $\bar{\mathbf{P}} \in \mathbb{R}^{m \times m}$ will be a compact version of $\mathbf{P}$. Figure 1 illustrates this idea for the case in which $n = 3$ and $m = 2$.

The stochasticity of $\bar{\mathbf{P}}$ follows immediately from the same property of $\mathbf{D}$ and $\mathbf{K}$. What is perhaps more surprising is the fact that this matrix shares some fundamental characteristics with the original matrix $\mathbf{P}$. Specifically, it is possible to show that: (*i*) for each recurrent class in $\mathbf{P}$ there is a corresponding class in $\bar{\mathbf{P}}$ with the same period and, given some simple assumptions about the factorization, (*ii*) $\mathbf{P}$ is irreducible if and only if $\bar{\mathbf{P}}$ is irreducible and (*iii*) $\mathbf{P}$ is regular if and only if $\bar{\mathbf{P}}$ is regular (for details, see the article by Barreto and Fragoso, 2011). We will refer to this insight as the "*stochastic-factorization trick*":

Given a stochastic factorization of a transition matrix, $\mathbf{P} = \mathbf{DK}$, swapping *the factors of the factorization yields another transition matrix* $\bar{\mathbf{P}} = \mathbf{KD}$, *potentially much smaller than the original, which retains the basic topology and properties of* $\mathbf{P}$.

Given the strong connection between $\mathbf{P} \in \mathbb{R}^{n \times n}$ and $\bar{\mathbf{P}} \in \mathbb{R}^{m \times m}$, the idea of replacing the former by the latter comes almost inevitably. The motivation for this would be, of course, to save computational resources when $m < n$. For example, Barreto and Fragoso

(2011) have shown that it is possible to recover the stationary distribution of $\mathbf{P}$ through a linear transformation of the corresponding distribution of $\bar{\mathbf{P}}$. In this paper we will use the stochastic-factorization trick to reduce the computational cost of KBRL. The strategy will be to summarize the information contained in KBRL's MDP in a model of fixed size.

3.2 Reducing a Markov Decision Process

The idea of using stochastic factorization to reduce dynamic programming's computational requirements is straightforward: given factorizations of the transition matrices $\mathbf{P}^a$, we can apply our trick to obtain a reduced MDP that will be solved in place of the original one. In the most general scenario, we would have one independent factorization $\mathbf{P}^a = \mathbf{D}^a\mathbf{K}^a$ for each action $a \in A$ and then use $\bar{\mathbf{P}}^a = \mathbf{K}^a\mathbf{D}^a$ instead of $\mathbf{P}^a$. However, in the current work we will focus on the particular case in which there is a single matrix $\mathbf{D}$, which will prove to be convenient both mathematically and computationally.

Obviously, in order to apply the stochastic-factorization trick to an MDP, we have to first *compute* the matrices involved in the factorization. Unfortunately, such a procedure can be computationally demanding, exceeding the number of operations necessary to calculate $\mathbf{v}^*$ (Vavasis, 2009; Barreto et al., 2014). Thus, in practice we may have to replace the exact factorizations $\mathbf{P}^a = \mathbf{DK}^a$ with approximations $\mathbf{P}^a \approx \mathbf{DK}^a$. The following proposition bounds the error in the value-function approximation resulting from the application of our trick to approximate stochastic factorizations:

Proposition 2 *Let* $M \equiv (S, A, \mathbf{P}^a, \mathbf{r}^a, \gamma)$ *be a finite MDP with* $|S| = n$ *and* $0 \leq \gamma < 1$. *Let* $\mathbf{D} \in \mathbb{R}^{n \times m}$ *be a stochastic matrix and, for each* $a \in A$, *let* $\mathbf{K}^a \in \mathbb{R}^{m \times n}$ *be stochastic and let* $\bar{\mathbf{r}}^a$ *be a vector in* $\mathbb{R}^m$. *Define the MDP* $\bar{M} \equiv (\bar{S}, A, \bar{\mathbf{P}}^a, \bar{\mathbf{r}}^a, \gamma)$, *with* $|\bar{S}| = m$ *and* $\bar{\mathbf{P}}^a = \mathbf{K}^a\mathbf{D}$. *Then,*

$$\left\|\mathbf{v}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\right\|_\infty \leq \xi_v \equiv \frac{1}{1-\gamma}\max_a \|\mathbf{r}^a - \mathbf{D}\bar{\mathbf{r}}^a\|_\infty + \frac{\bar{R}_{\text{dif}}}{(1-\gamma)^2}\left(\frac{\gamma}{2}\max_a \|\mathbf{P}^a - \mathbf{DK}^a\|_\infty + \sigma(\mathbf{D})\right), \quad (8)$$

where

$$\sigma(\mathbf{D}) = \max_i (1 - \max_j d_{ij}), \quad (9)$$

$\|\cdot\|_\infty$ *is the maximum norm, and* $\bar{R}_{\text{dif}} = \max_{a,i} \bar{r}_i^a - \min_{a,i} \bar{r}_i^a$.[2]

The proofs of most of our theoretical results are in Appendix A.1. We note that Proposition 2 is only valid for the maximum norm; in Appendix A.2 we derive another bound for the distance between $\mathbf{v}^*$ and $\Gamma\mathbf{D}\bar{\mathbf{Q}}^*$ which is valid for any norm.

Our bound depends on two factors: the quality of the MDP's factorization, given by $\max_a \|\mathbf{P}^a - \mathbf{DK}^a\|_\infty$ and $\max_a \|\mathbf{r}^a - \mathbf{D}\bar{\mathbf{r}}^a\|_\infty$, and the "level of stochasticity" of $\mathbf{D}$, measured by $\sigma(\mathbf{D})$. When the MDP factorization is exact, we recover a computable version of Sorg and Singh's (2009) bound for soft homomorphisms (see (32)). On the other hand, when $\mathbf{D}$ is deterministic—that is, when all its nonzero elements are 1—expression (8) reduces to Whitt's (1978) classical result regarding state aggregation in dynamic programming. Finally, if we have exact deterministic factorizations, the right-hand side of (8) reduces to

2. We recall that $\|\cdot\|_\infty$ induces the following norm over the space of matrices: $\|\mathbf{A}\|_\infty = \max_i \sum_j |a_{ij}|$.

zero. This also makes sense, since in this case the stochastic-factorization trick gives rise to an exact homomorphism (Ravindran, 2004).

Proposition 2 elucidates the basic mechanism through which one can use the stochastic-factorization trick to reduce the number of states in an MDP (and hence the computational cost of finding a policy using dynamic programming). One possible way to exploit this result is to see the computation of $\mathbf{D}$, $\mathbf{K}^a$, and $\bar{\mathbf{r}}^a$ as an optimization problem in which the objective is to minimize some function of $\max_a \|\mathbf{P}^a - \mathbf{D}\mathbf{K}^a\|_\infty$, $\max_a \|\mathbf{r}^a - \mathbf{D}\bar{\mathbf{r}}^a\|_\infty$, and possibly also $\sigma(\mathbf{D})$ (Barreto et al., 2014). Note though that addressing the factorization problem as an optimization may be computationally infeasible when the dimension of the matrices $\mathbf{P}^a$ is large—which is exactly the case we are interested in here. To illustrate this point, we will draw a connection between the stochastic factorization and a popular problem known in the literature as *nonnegative matrix factorization* (Paatero and Tapper, 1994; Lee and Seung, 1999).

In a nonnegative matrix factorization the elements of $\mathbf{D}$ and $\mathbf{K}^a$ are greater or equal to zero, but in general no stochasticity constraint is imposed. Cohen and Rothblum (1991) have shown that it is always possible to derive a stochastic factorization from a nonnegative factorization of a stochastic matrix, which formally characterizes the former as a particular case of the latter. Unfortunately, nonnegative factorization is hard: Vavasis (2009) has shown that a particular version of the problem is in fact NP-hard.

Instead of solving the problem exactly, one can resort instead to an approximate nonnegative matrix factorization. However, since the number of states n in an MDP determines both the number of rows and the number of columns of the matrices $\mathbf{P}^a$, even the fast "linear" approximate methods run in $O(n^2)$ time, which is infeasible for large n (Barreto et al., 2014). One can circumvent this computational obstacle by exploiting structure in the optimization problem or by resorting to heuristics. In another article on the subject we explore both these alternatives at length (Barreto et al., 2014). However, in this paper we adopt a different approach. Since the model $\hat{M}$ built by KBRL is itself an approximation, instead of insisting in finding a near-optimal factorization for $\hat{M}$ we apply our trick to *avoid* the construction of $\mathbf{P}^a$ and $\mathbf{r}^a$. As will be seen, this is done by applying KBRL's own approximation scheme to $\hat{M}$.

4. Kernel-Based Stochastic Factorization

In Section 2 we presented KBRL, an approximation framework for reinforcement learning whose main drawback is its high computational complexity. In Section 3 we discussed how the stochastic-factorization trick can in principle be useful to reduce an MDP, as long as one circumvents the computational burden imposed by the calculation of the matrices involved in the process. We now show that by combining these two approaches we get an algorithm that overcomes the computational limitations of its components. We call it *kernel-based stochastic factorization*, or KBSF for short.

KBSF emerges from the application of the stochastic-factorization trick to KBRL's MDP $\hat{M}$ (Barreto et al., 2011). Similarly to Ormoneit and Sen (2002), we start by defining a "mother kernel" $\bar{\phi}(x) : \mathbb{R}^+ \mapsto \mathbb{R}^+$. In Section 4.1 we list our assumptions regarding $\bar{\phi}$. Here, it suffices to note that, since our assumptions and Ormoneit and Sen's (2002) are not mutually exclusive, we can have $\phi \equiv \bar{\phi}$ (by using the Gaussian function in both

cases, for example). Let $\bar{S} \equiv \{\bar{s}_1, \bar{s}_2, ..., \bar{s}_m\}$ be a set of *representative states*. Analogously to (2) and (3), we define the kernel $\bar{\mathrm{k}}_{\bar{\tau}}(s, s') = \bar{\phi}(\|s - s'\|/\bar{\tau})$ and its normalized version $\bar{\kappa}_{\bar{\tau}}(s, \bar{s}_i) = \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_i)/\sum_{j=1}^{m} \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_j)$. We will use κ_τ^a to build matrices $\mathbf{K}^a$ and $\bar{\kappa}_{\bar{\tau}}$ to build matrix $\mathbf{D}$.

As shown in Figure 2a, KBRL's matrices $\hat{\mathbf{P}}^a$ have a very specific structure, since only transitions ending in states $\hat{s}_i^a \in S^a$ have a nonzero probability of occurrence. Suppose that we want to apply the stochastic-factorization trick to KBRL's MDP. Assuming that the matrices $\mathbf{K}^a$ have the same structure as $\hat{\mathbf{P}}^a$, when computing $\bar{\mathbf{P}}^a = \mathbf{K}^a\mathbf{D}$ we only have to look at the sub-matrices of $\mathbf{K}^a$ and $\mathbf{D}$ corresponding to the n_a nonzero columns of $\mathbf{K}^a$. We call these matrices $\dot{\mathbf{K}}^a \in \mathbb{R}^{m \times n_a}$ and $\dot{\mathbf{D}}^a \in \mathbb{R}^{n_a \times m}$. The strategy of KBSF is to fill out matrices $\dot{\mathbf{K}}^a$ and $\dot{\mathbf{D}}^a$ with elements

$$\dot{k}_{ij}^a = \kappa_\tau^a(\bar{s}_i, s_j^a) \quad \text{and} \quad \dot{d}_{ij}^a = \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_j). \tag{10}$$

Note that, based on $\dot{\mathbf{D}}^a$, one can easily recover $\mathbf{D}$ as $\mathbf{D}^\intercal \equiv [(\dot{\mathbf{D}}^1)^\intercal, (\dot{\mathbf{D}}^2)^\intercal, ...(\dot{\mathbf{D}}^{|A|})^\intercal] \in \mathbb{R}^{n \times m}$. Similarly, if we let $\mathbf{K} \equiv [\dot{\mathbf{K}}^1, \dot{\mathbf{K}}^2, ...\dot{\mathbf{K}}^{|A|}] \in \mathbb{R}^{m \times n}$, then $\mathbf{K}^a \in \mathbb{R}^{m \times n}$ is matrix $\mathbf{K}$ with all elements replaced by zeros except for those corresponding to matrix $\dot{\mathbf{K}}^a$(see Figures 2b and 2c for an illustration). It should be thus obvious that $\bar{\mathbf{P}}^a = \mathbf{K}^a\mathbf{D} = \dot{\mathbf{K}}^a\dot{\mathbf{D}}^a$.

In order to conclude the construction of KBSF's MDP, we have to define the vectors of expected rewards $\bar{\mathbf{r}}^a$. As shown in expression (5), the reward functions of KBRL's MDP, $\hat{R}^a(s, s')$, only depend on the ending state s'. Recalling the interpretation of the rows of $\mathbf{K}^a$ as transition probabilities from the representative states to the original ones, illustrated in Figure 1, it is clear that

$$\bar{\mathbf{r}}^a = \dot{\mathbf{K}}^a\mathbf{r}^a = \mathbf{K}^a\mathbf{r}. \tag{11}$$

Therefore, the formal specification of KBSF's MDP is given by $\bar{M} \equiv (\bar{S}, A, \dot{\mathbf{K}}^a\dot{\mathbf{D}}^a, \dot{\mathbf{K}}^a\mathbf{r}^a, \gamma) = (\bar{S}, A, \mathbf{K}^a\mathbf{D}^a, \mathbf{K}^a\mathbf{r}, \gamma) = (\bar{S}, A, \bar{\mathbf{P}}^a, \bar{\mathbf{r}}^a, \gamma)$.

As discussed in Section 2.2, KBRL's approximation scheme can be interpreted as the derivation of a finite MDP. In this case, the sample transitions define both the finite state space $\hat{S}$ and the model's transition and reward functions. This means that the state space and the dynamics of KBRL's model are inexorably linked: except maybe for degenerate cases, changing one also changes the other. By defining a set of representative states, KBSF decouples the MDP's structure from its particular instantiation. To see why this is so, note that, if we fix the representative states, different sets of sample transitions will give rise to different models. Conversely, the same set of transitions can generate different MDPs, depending on how the representative states are defined.

A step by step description of KBSF is given in Algorithm 1. As one can see, KBSF is very simple to understand and to implement. It works as follows: first, the MDP $\bar{M}$ is built as described above. Then, its action-value function $\bar{\mathbf{Q}}^*$ is determined through any dynamic programming algorithm. Finally, KBSF returns an approximation of $\hat{\mathbf{v}}^*$—the optimal value function of KBRL's MDP—computed as $\tilde{\mathbf{v}} = \Gamma\mathbf{D}\bar{\mathbf{Q}}^*$. Based on $\tilde{\mathbf{v}}$, one can compute an approximation of KBRL's action-value function $\hat{Q}(s, a)$ by simply replacing $\tilde{V}$ for $\hat{V}^*$ in (7), that is,

$$\tilde{Q}(s, a) = \sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a) \left[r_i^a + \gamma \tilde{V}(\hat{s}_i^a) \right], \tag{12}$$

$$\hat{\mathbf{P}}^a = \begin{array}{c} \\ \hat{s}_1^a \\ \hat{s}_2^a \\ \hat{s}_3^a \\ \hat{s}_1^b \\ \hat{s}_2^b \end{array} \begin{array}{c} \begin{array}{cccc} \hat{s}_1^a & \hat{s}_2^a & \hat{s}_3^a & \hat{s}_1^b \quad \hat{s}_2^b \end{array} \\ \begin{bmatrix} \kappa_\tau^a(\hat{s}_1^a, s_1^a) & \kappa_\tau^a(\hat{s}_1^a, s_2^a) & \kappa_\tau^a(\hat{s}_1^a, s_3^a) & 0 \quad 0 \\ \kappa_\tau^a(\hat{s}_2^a, s_1^a) & \kappa_\tau^a(\hat{s}_2^a, s_2^a) & \kappa_\tau^a(\hat{s}_2^a, s_3^a) & 0 \quad 0 \\ \kappa_\tau^a(\hat{s}_3^a, s_1^a) & \kappa_\tau^a(\hat{s}_3^a, s_2^a) & \kappa_\tau^a(\hat{s}_3^a, s_3^a) & 0 \quad 0 \\ \kappa_\tau^a(\hat{s}_1^b, s_1^a) & \kappa_\tau^a(\hat{s}_1^b, s_2^a) & \kappa_\tau^a(\hat{s}_1^b, s_3^a) & 0 \quad 0 \\ \kappa_\tau^a(\hat{s}_2^b, s_1^a) & \kappa_\tau^a(\hat{s}_2^b, s_2^a) & \kappa_\tau^a(\hat{s}_2^b, s_3^a) & 0 \quad 0 \end{bmatrix} \end{array},$$

$$\hat{\mathbf{P}}^b = \begin{array}{c} \\ \hat{s}_1^a \\ \hat{s}_2^a \\ \hat{s}_3^a \\ \hat{s}_1^b \\ \hat{s}_2^b \end{array} \begin{array}{c} \begin{array}{ccccc} \hat{s}_1^a & \hat{s}_2^a & \hat{s}_3^a & \hat{s}_1^b & \hat{s}_2^b \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & \kappa_\tau^a(\hat{s}_1^a, s_1^b) & \kappa_\tau^a(\hat{s}_1^a, s_2^b) \\ 0 & 0 & 0 & \kappa_\tau^a(\hat{s}_2^a, s_1^b) & \kappa_\tau^a(\hat{s}_2^a, s_2^b) \\ 0 & 0 & 0 & \kappa_\tau^a(\hat{s}_3^a, s_1^b) & \kappa_\tau^a(\hat{s}_3^a, s_2^b) \\ 0 & 0 & 0 & \kappa_\tau^a(\hat{s}_1^b, s_1^b) & \kappa_\tau^a(\hat{s}_1^b, s_2^b) \\ 0 & 0 & 0 & \kappa_\tau^a(\hat{s}_2^b, s_1^b) & \kappa_\tau^a(\hat{s}_2^b, s_2^b) \end{bmatrix} \end{array}$$

(a) KBRL's matrices

$$\mathbf{D} = \begin{array}{c} \\ \hat{s}_1^a \\ \hat{s}_2^a \\ \hat{s}_3^a \\ \hat{s}_1^b \\ \hat{s}_2^b \end{array} \begin{array}{c} \begin{array}{cc} \bar{s}_1 & \bar{s}_2 \end{array} \\ \begin{bmatrix} \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^a, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^a, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^a, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^a, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_3^a, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_3^a, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^b, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^b, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^b, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^b, \bar{s}_2) \end{bmatrix} \end{array},$$

$$\mathbf{K}^a = \begin{array}{c} \\ \bar{s}_1 \\ \bar{s}_2 \end{array} \begin{array}{c} \begin{array}{ccccc} \hat{s}_1^a & \hat{s}_2^a & \hat{s}_3^a & \hat{s}_1^b & \hat{s}_2^b \end{array} \\ \begin{bmatrix} \kappa_\tau^a(\bar{s}_1, s_1^a) & \kappa_\tau^a(\bar{s}_1, s_2^a) & \kappa_\tau^a(\bar{s}_1, s_3^a) & 0 & 0 \\ \kappa_\tau^a(\bar{s}_2, s_1^a) & \kappa_\tau^a(\bar{s}_2, s_2^a) & \kappa_\tau^a(\bar{s}_2, s_3^a) & 0 & 0 \end{bmatrix} \end{array},$$

$$\mathbf{K}^b = \begin{array}{c} \\ \bar{s}_1 \\ \bar{s}_2 \end{array} \begin{array}{c} \begin{array}{ccccc} \hat{s}_1^a & \hat{s}_2^a & \hat{s}_3^a & \hat{s}_1^b & \hat{s}_2^b \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & \kappa_\tau^a(\bar{s}_1, s_1^b) & \kappa_\tau^a(\bar{s}_1, s_2^b) \\ 0 & 0 & 0 & \kappa_\tau^a(\bar{s}_2, s_1^b) & \kappa_\tau^a(\bar{s}_2, s_2^b) \end{bmatrix} \end{array}.$$

(b) KBSF's sparse matrices

$$\dot{\mathbf{D}}^a = \begin{array}{c} \\ \hat{s}_1^a \\ \hat{s}_2^a \\ \hat{s}_3^a \end{array} \begin{array}{c} \begin{array}{cc} \bar{s}_1 & \bar{s}_2 \end{array} \\ \begin{bmatrix} \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^a, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^a, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^a, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^a, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_3^a, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_3^a, \bar{s}_2) \end{bmatrix} \end{array},$$

$$\dot{\mathbf{K}}^a = \begin{array}{c} \\ \bar{s}_1 \\ \bar{s}_2 \end{array} \begin{array}{c} \begin{array}{ccc} \hat{s}_1^a & \hat{s}_2^a & \hat{s}_3^a \end{array} \\ \begin{bmatrix} \kappa_\tau^a(\bar{s}_1, s_1^a) & \kappa_\tau^a(\bar{s}_1, s_2^a) & \kappa_\tau^a(\bar{s}_1, s_3^a) \\ \kappa_\tau^a(\bar{s}_2, s_1^a) & \kappa_\tau^a(\bar{s}_2, s_2^a) & \kappa_\tau^a(\bar{s}_2, s_3^a) \end{bmatrix} \end{array},$$

$$\dot{\mathbf{D}}^b = \begin{array}{c} \\ \hat{s}_1^b \\ \hat{s}_2^b \end{array} \begin{array}{c} \begin{array}{cc} \bar{s}_1 & \bar{s}_2 \end{array} \\ \begin{bmatrix} \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^b, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_1^b, \bar{s}_2) \\ \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^b, \bar{s}_1) & \bar{\kappa}_{\bar{\tau}}(\hat{s}_2^b, \bar{s}_2) \end{bmatrix} \end{array},$$

$$\dot{\mathbf{K}}^b = \begin{array}{c} \\ \bar{s}_1 \\ \bar{s}_2 \end{array} \begin{array}{c} \begin{array}{cc} \hat{s}_1^b & \hat{s}_2^b \end{array} \\ \begin{bmatrix} \kappa_\tau^a(\bar{s}_1, s_1^b) & \kappa_\tau^a(\bar{s}_1, s_2^b) \\ \kappa_\tau^a(\bar{s}_2, s_1^b) & \kappa_\tau^a(\bar{s}_2, s_2^b) \end{bmatrix} \end{array}.$$

(c) KBSF's dense matrices

Figure 2: Matrices built by KBRL and KBSF for the case in which the original MDP has two actions, a and b, and $n_a = 3$, $n_b = 2$, and $m = 2$.

where $s \in \mathbb{S}$ and $a \in A$. Note that $\tilde{V}(\hat{s}_i^a)$ corresponds to one specific entry of vector $\tilde{\mathbf{v}}$, whose index is given by $\sum_{b=0}^{a-1} n_b + i$, where we assume that $n_0 = 0$.

Algorithm 1 Batch KBSF

Input: $S^a = \{(s_k^a, r_k^a, \hat{s}_k^a) | k = 1, 2, ..., n_a\}$ for all $a \in A$ ▷ Sample transitions
$\bar{S} = \{\bar{s}_1, \bar{s}_2, ..., \bar{s}_m\}$ ▷ Set of representative states
Output: $\tilde{\mathbf{v}} \approx \hat{\mathbf{v}}^*$
for each $a \in A$ **do**
  Compute matrix $\dot{\mathbf{D}}^a$: $\dot{d}_{ij}^a = \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_j)$
  Compute matrix $\dot{\mathbf{K}}^a$: $\dot{k}_{ij}^a = \kappa_\tau^a(\bar{s}_i, s_j^a)$
  Compute vector $\bar{\mathbf{r}}^a$: $\bar{r}_i^a = \sum_j \dot{k}_{ij}^a r_j^a$
  Compute matrix $\bar{\mathbf{P}}^a = \dot{\mathbf{K}}^a \dot{\mathbf{D}}^a$
Solve $\bar{M} \equiv (\bar{S}, A, \bar{\mathbf{P}}^a, \bar{\mathbf{r}}^a, \gamma)$ ▷ *i.e.*, compute $\bar{\mathbf{Q}}^*$
Return $\tilde{\mathbf{v}} = \Gamma \mathbf{D} \bar{\mathbf{Q}}^*$, where $\mathbf{D}^\intercal = \left[(\dot{\mathbf{D}}^1)^\intercal, (\dot{\mathbf{D}}^2)^\intercal, ...(\dot{\mathbf{D}}^{|A|})^\intercal\right]$

As shown in Algorithm 1, the key point of KBSF's mechanics is the fact that the matrices $\check{\mathbf{P}}^a = \mathbf{D}\mathbf{K}^a$ are never actually computed, but instead we directly solve the MDP $\bar{M}$ containing m states only. This results in an efficient algorithm that requires only $O(nm|A|d_{\mathbb{S}} + \hat{n}m^2|A|)$ operations and $O(\hat{n}m)$ bits to build a reduced version of KBRL's MDP, where $\hat{n} = \max_a n_a$. After the reduced model $\bar{M}$ has been constructed, KBSF's computational cost becomes a function of m only. In particular, the cost of solving $\bar{M}$ through dynamic programming becomes polynomial in m instead of n: while one application of $\hat{T}$, the Bellman operator of $\hat{M}$, is $O(n\hat{n}|A|)$, the computation of $\bar{T}$ is $O(m^2|A|)$. Therefore, KBSF's time and memory complexities are only linear in n.

We note that, in practice, KBSF's computational requirements can be reduced even further if one enforces the kernels κ_τ^a and $\bar{\kappa}_{\bar{\tau}}$ to be sparse. In particular, given a fixed $\bar{s}_i$, instead of computing $\bar{\mathrm{k}}_{\bar{\tau}}(\bar{s}_i, s_j^a)$ for $j = 1, 2, ..., n_a$, one can evaluate the kernel on a prespecified neighborhood of $\bar{s}_i$ only. Assuming that $\bar{\mathrm{k}}_{\bar{\tau}}(\bar{s}_i, s_j^a)$ is zero for all s_j^a outside this region, one can avoid not only computing the kernel but also storing the resulting values (the same reasoning applies to the computation of $\mathrm{k}_\tau(\hat{s}_i^a, \bar{s}_j)$ for a fixed $\hat{s}_i^a$).

4.1 Theoretical results

Since KBSF comes down to the solution of a finite MDP, it always converges to the same approximation $\tilde{\mathbf{v}}$, whose distance to KBRL's optimal value function $\hat{\mathbf{v}}^*$ is bounded by Proposition 2. Once $\tilde{\mathbf{v}}$ is available, the value of any state-action pair can be determined through (12). The following result generalizes Proposition 2 to the entire continuous state space $\mathbb{S}$:

Proposition 3 *Let $\hat{Q}$ be the value function computed by KBRL through (7) and let $\tilde{Q}$ be the value function computed by KBSF through (12). Then, for any $s \in \mathbb{S}$ and any $a \in A$, $|\hat{Q}(s,a) - \tilde{Q}(s,a)| \leq \gamma \xi_v$, with ξ_v defined in (8).*

Proof

$$|\hat{Q}(s,a) - \tilde{Q}(s,a)| = \left| \sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a) \left[r_i^a + \gamma \hat{V}^*(\hat{s}_i^a) \right] - \sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a) \left[r_i^a + \gamma \tilde{V}(\hat{s}_i^a) \right] \right|$$
$$\leq \gamma \sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a) \left| \hat{V}^*(\hat{s}_i^a) - \tilde{V}(\hat{s}_i^a) \right| \leq \gamma \sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a) \xi_v \leq \gamma \xi_v,$$

where the second inequality results from the application of Proposition 2 and the third inequality is a consequence of the fact that $\sum_{i=1}^{n_a} \kappa_\tau^a(s, s_i^a)$ defines a convex combination. ■

Proposition 3 makes it clear that the approximation computed by KBSF depends crucially on ξ_v. In the remainder of this section we will show that, if the distances between sampled states and the respective nearest representative states are small enough, then we can make ξ_v as small as desired by setting $\bar{\tau}$ to a sufficiently small value.

4.1.1 General results

We assume that KBSF's kernel $\bar{\phi}(x) : \mathbb{R}^+ \mapsto \mathbb{R}^+$ has the following properties:

(i) $\bar{\phi}(x) \geq \bar{\phi}(y)$ if $x < y$,

(ii) $\exists\, A_{\bar{\phi}} > 0, \lambda_{\bar{\phi}} \geq 1, B_{\bar{\phi}} \geq 0$ such that $A_{\bar{\phi}} \exp(-x) \leq \bar{\phi}(x) \leq \lambda_{\bar{\phi}} A_{\bar{\phi}} \exp(-x)$ if $x \geq B_{\bar{\phi}}$.

Assumption (ii) implies that the function $\bar{\phi}$ is positive and will eventually decay exponentially. Note that we assume that $\bar{\phi}$ is greater than zero everywhere in order to guarantee that $\bar{\kappa}_{\bar{\tau}}$ is well defined for any value of $\bar{\tau}$. It should be straightforward to generalize our results for the case in which $\bar{\phi}$ has finite support by ensuring that, given sets of sample transitions S^a and a set of representative states $\bar{S}$, $\bar{\tau}$ is such that, for any $\hat{s}_i^a \in S^a$, with $a \in A$, there is a $\bar{s}_j \in \bar{S}$ for which $\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_j) > 0$ (note that this assumption is naturally satisfied by the "sparse kernels" used in some of the experiments—see Appendix B).

Let $rs : \mathbb{S} \times \{1, 2, ..., m\} \mapsto \bar{S}$ be a function that orders the representative states according to their distance to a given state s, that is,

$$rs(s,i) = \bar{s}_k \iff \bar{s}_k \text{ is the } i^{\text{th}} \text{ nearest representative state to } s. \tag{13}$$

Define

$$dist : \mathbb{S} \times \{1, 2, ..., m\} \mapsto \mathbb{R}$$
$$dist(s,i) = \|s - rs(s,i)\|. \tag{14}$$

We will show that, for any $\epsilon > 0$ and any $w \in \{1, 2, ..., m\}$, there is a $\delta > 0$ such that, if $\max_{a,i} dist(\hat{s}_i^a, w) < \delta$, then we can set $\bar{\tau}$ in order to guarantee that $\xi_v < \epsilon$. To show that, we will need the following two lemmas:

Lemma 4 *For any $s_i^a \in S^a$ and any $\epsilon > 0$, there is a $\delta > 0$ such that $|\kappa_\tau^a(s, s_i^a) - \kappa_\tau^a(s', s_i^a)| < \epsilon$ if $\|s - s'\| < \delta$.*

Lemma 5 *Let $s \in \mathbb{S}$, let $m > 1$, and assume there is a $w \in \{1, 2, ..., m-1\}$ such that $dist(s,w) < dist(s,w+1)$. Define*

$$W^w(s) \equiv \{k \mid \|s - \bar{s}_k\| \leq dist(s,w)\} \text{ and } \bar{W}^w(s) \equiv \{1, 2, ..., m\} - W^w(s). \tag{15}$$

Then, for any $\alpha > 0$, $\sum_{k \in W^w(s)} \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_k) < \alpha \sum_{k \in \bar{W}^w(s)} \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_k)$ for $\bar{\tau}$ sufficiently small.

Lemma 4 is basically a continuity argument: it shows that, for any fixed s_i^a, $|\kappa_\tau^a(s, s_i^a) - \kappa_\tau^a(s', s_i^a)| \to 0$ as $\|s - s'\| \to 0$. Lemma 5 states that, if we order the representative states according to their distance to a fixed state s, and then partition them in two subsets, we can control the relative magnitude of the corresponding kernels's sums by adjusting the parameter $\bar{\tau}$ (we redirect the reader to Appendix A.1 for details on how to set $\bar{\tau}$). Based on these two lemmas, we present the main result of this section:

Proposition 6 *Let $w \in \{1, 2, ..., m\}$. For any $\epsilon > 0$, there is a $\delta > 0$ such that, if $\max_{a,i} dist(\hat{s}_i^a, w) < \delta$, then we can guarantee that $\xi_v < \epsilon$ by making $\bar{\tau}$ sufficiently small.*

Proposition 6 tells us that, regardless of the specific reinforcement learning problem at hand, if the distances between sampled states $\hat{s}_i^a$ and the respective w nearest representative states are small enough, then we can make KBSF's approximation of KBRL's value function as accurate as desired by setting $\bar{\tau}$ to a sufficiently small value (one can see how exactly to set $\bar{\tau}$ in the proof of the proposition). How small the maximum distance $\max_{a,i} dist(\hat{s}_i^a, w)$ should be depends on the particular choice of kernel k_τ and on the sets of sample transitions S^a.

Note that a fixed number of representative states m imposes a minimum possible value for $\max_{a,i} dist(\hat{s}_i^a, w)$, and if this value is not small enough decreasing $\bar{\tau}$ may actually hurt the approximation (this is easier to see if we consider that $w = 1$). The optimal value for $\bar{\tau}$ in this case is again context-dependent. As a positive flip side of this statement, we note that, even if $\max_{a,i} dist(\hat{s}_i^a, w) > \delta$, it might be possible to make $\xi_v < \epsilon$ by setting $\bar{\tau}$ appropriately. Therefore, rather than as a practical guide on how to configure KBSF, Proposition 6 should be seen as a theoretical argument showing that KBSF is a sound algorithm, in the sense that in the limit it recovers KBRL's solution.

4.1.2 Error rate

In the previous section we deliberately refrained from making assumptions on the kernel $\bar{\kappa}_{\bar{\tau}}$ used by KBSF in order to make Proposition 6 as general as possible. In what follows we show that, by restricting the class of kernels used by our algorithm, we can derive stronger results regarding its behavior. In particular, we derive an upper bound for ξ_v that shows how this approximation error depends on the variables of a given reinforcement learning problem. Our strategy will be to define an "admissible kernel" whose width is determined based on data.

We will need the following assumption:

(iii) $|\phi(x) - \phi(y)| \leq C_\phi |x - y|$, with $C_\phi \geq 0$.

Assumption (iii) simply states that the function ϕ used to construct the kernel κ_τ^a is Lipschitz continuous with constant C_ϕ. This is actually part of Ormoneit and Sen's (2002) Assumption 4, and is explicitly listed here for convenience only.

We will now define an auxiliary function which will be used in the definition of our admissible kernel. To simplify the notation, let

$$\varsigma_k^{a,i,j} \equiv |\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)|. \tag{16}$$

Based on (15) and (16), we define the following function:

$$\mathcal{F}(\bar{\tau}, w | S^a, \bar{S}, \kappa_\tau^a, \bar{\mathrm{k}}_{\bar{\tau}}) = \frac{R_{\max}}{(1-\gamma)^2}\left[\max_{a,i} \sum_{k \in \bar{W}^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} + \frac{1}{2}\max_{a,i}(1 - \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)))\right], \tag{17}$$

where $\bar{\tau} > 0$, $w \in \{1, 2, ..., m\}$, $R_{\max} = \max_{a,i} |r_i^a|$, and rs and $\bar{W}^w$ are defined in (13) and (15), respectively. Note that the definition of $\mathcal{F}$ makes it clear its dependency on the representative states, sets of sample transitions, and kernels adopted.

Lemma 5 implies that, as $\bar{\tau} \to 0$, $\bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) \to 1$ and $\sum_{k \in \bar{W}^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \to 0$ for all a, i, and w (see equation (35) in Appendix A.1 for a clear picture). Thus, for any w, $\mathcal{F}(\bar{\tau}, w) \to 0$ as $\bar{\tau} \to 0$. This leads to the following definition:

Definition 7 *Given $\epsilon > 0$ and $w \in \{1, 2, ..., m\}$, an* admissible kernel $\bar{\kappa}_{\bar{\tau}}^{\epsilon,w}$ *is any kernel whose parameter $\bar{\tau}$ is such that $\mathcal{F}(\bar{\tau}, w) < \epsilon$.*

The definition above allows us to enunciate the following proposition:

Proposition 8 *Let $\mathrm{k}_{\min} = \phi\left(\sqrt{d_{\mathbb{S}}}/\tau\right)$, where τ is the parameter used by KBRL's kernel κ_τ^a. Given $\epsilon > 0$ and $w \in \{1, 2, ..., m\}$, suppose that KBSF is adopted with an admissible kernel $\bar{\kappa}_{\bar{\tau}}^{\epsilon,w}$. Then,*

$$\xi_v \leq \frac{2wC_\phi R_{\max}}{\tau \mathrm{k}_{\min}(1-\gamma)^2} \max_{a,i} dist(\hat{s}_i^a, w) + \epsilon, \tag{18}$$

where C_ϕ is the Lipschitz constant of function ϕ appearing in Assumption (iii), $R_{\max} = \max_{a,i} |r_i^a|$, and $\gamma \in [0, 1)$ is the discount factor of the underlying MDP.

Proposition 8 shows that ξ_v is bounded by the maximum distance between a sampled state and the w^{th} closest representative state scaled by constants characterizing the MDP and the kernels used by KBSF.

Assuming that ϵ and w are fixed, among the quantities appearing in (18) we only have control over τ and $\max_{a,i} dist(\hat{s}_i^a, w)$—the latter through the definition of the representative states. Regarding $\max_{a,i} dist(\hat{s}_i^a, w)$, the same observation made above applies here: given sample transitions S^a, a fixed value for $m < n$ imposes a lower bound on this term, and thus on the right-hand side of (18). As $m \to n$, this lower bound approaches some value ϵ', with $0 \leq \epsilon' < \epsilon$. The fact that (18) is generally greater than zero even when $m = n$ reflects the fact that ξ_v depends on $\sigma(\mathbf{D})$, the level of stochasticity of $\mathbf{D}$ (see (8) and (9)). Since Assumption (ii) implies that $\bar{\mathrm{k}}_{\bar{\tau}}(s, s') > 0$ for all $s, s' \in \mathbb{S}$, matrix $\mathbf{D}$ will never become deterministic, no matter how small $\bar{\tau}$ is (see Figure 2b). If we replace $\bar{\mathrm{k}}_{\bar{\tau}}$ by a kernel with finite support, then we can set $\epsilon = 0$ in Definition 7, and therefore in the right-hand side of (18). Needless to say, this does not mean that using a kernel with finite support will lead to better performance in practice.

As for the definition of τ, we see that the right-hand side of (18) decreases as $\tau \to \infty$. This means that we can make the value function computed by KBSF arbitrarily close to the one computed by KBRL. Note though that increasing τ also changes the model constructed by KBRL, and an excessively large value for this parameter makes the resulting approximation meaningless (see (4) and (5)). Similarly, we might be tempted to always set w to 1 in order to minimize the right-hand side of (18). Note however that a kernel $\bar{\kappa}_{\bar{\tau}}^{\epsilon,w}$ that is admissible for $w > 1$ may not be so for $w = 1$, and in this case the bound would no longer be valid.

4.2 Empirical results

We now present a series of computational experiments designed to illustrate the behavior of KBSF in a variety of challenging domains. We start with a simple problem, the "puddle world", to show that KBSF is indeed capable of compressing the information contained in KBRL's model. We then move to more difficult tasks, and compare KBSF with other state-of-the-art reinforcement-learning algorithms. We start with two classical control tasks, single and double pole-balancing. Next we study two medically-related problems based on real data: HIV drug schedule and epilepsy-suppression domains.

All problems considered in this paper have a continuous state space and a finite number of actions, and were modeled as discounted tasks. The algorithms's results correspond to the performance of the greedy decision policy derived from the final value function computed. In all cases, the decision policies were evaluated on challenging test states from which the tasks cannot be easily solved. The details of the experiments are given in Appendix B.

4.2.1 Puddle world (proof of concept)

In order to show that KBSF is indeed capable of summarizing the information contained in KBRL's model, we use the puddle world task (Sutton, 1996). The puddle world is a simple two-dimensional problem in which the objective is to reach a goal region avoiding two "puddles" along the way. We implemented the task exactly as described by Sutton (1996), except that we used a discount factor of $\gamma = 0.99$ and evaluated the decision policies on a set of pre-defined test states surrounding the puddles (see Appendix B).

The experiment was carried out as follows: first, we collected a set of n sample transitions $(s_k^a, r_k^a, \hat{s}_k^a)$ using a random exploration policy (that is, a policy that selects actions uniformly at random). In the case of KBRL, this set of sample transitions defined the model used to approximate the value function. In order to define KBSF's model, the states $\hat{s}_k^a$ were grouped by the k-means algorithm into m clusters and a representative state $\bar{s}_j$ was placed at the center of each resulting cluster (Kaufman and Rousseeuw, 1990). As for the kernels's widths, we varied both τ and $\bar{\tau}$ in the set $\{0.01, 0.1, 1\}$ (see Table 1). The results reported represent the best performance of the algorithms over 50 runs; that is, for each n and each m we picked the combination of parameters that generated the maximum average return. We use the following convention to refer to specific instances of each method: the first number enclosed in parentheses after an algorithm's name is n, the number of sample transitions used in the approximation, and the second one is m, the size of the model used to approximate the value function. Note that for KBRL n and m coincide.

In Figure 3a and 3b we observe the effect of fixing the number of transitions n and varying the number of representative states m. As expected, KBSF's results improve as $m \to n$. More surprising is the fact that KBSF has essentially the same performance as KBRL using models one order of magnitude smaller. This indicates that KBSF is summarizing well the information contained in the data. Depending on the values of n and m, such a compression may represent a significant reduction on the consumption of computational resources. For example, by replacing KBRL(8000) with KBSF(8000, 100), we obtain a decrease of approximately 99.58% on the number of operations performed to find a policy, as shown in Figure 3b (the cost of constructing KBSF's MDP is included in all reported run times).

In Figures 3c and 3d we fix m and vary n. Observe in Figure 3c how KBRL and KBSF have similar performances, and both improve as n increases. However, since KBSF is using a model of fixed size, its computational cost depends only linearly on n, whereas KBRL's cost grows with $n^2\hat{n}$, roughly. This explains the huge difference in the algorithms's run times shown in Figure 3d.

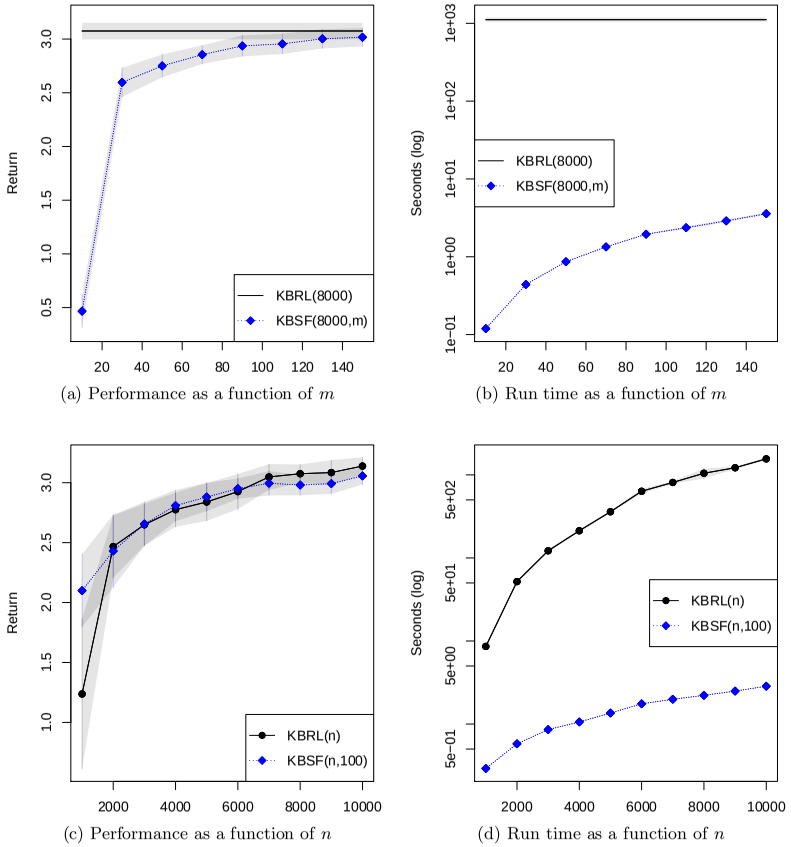
4.2.2 Single and double pole-balancing (comparison with LSPI)

We now evaluate how KBSF compares to other modern reinforcement learning algorithms on more difficult tasks. We first contrast our method with Lagoudakis and Parr's (2003) least-squares policy iteration algorithm (LSPI). Besides its popularity, LSPI is a natural candidate for such a comparison for three reasons: it also builds an approximator of fixed size out of a batch of sample transitions, it has good theoretical guarantees, and it has been successfully applied to several reinforcement learning tasks.

We compare the performance of LSPI and KBSF on the pole balancing task. Pole balancing has a long history as a benchmark problem because it represents a rich class of unstable systems (Michie and Chambers, 1968; Anderson, 1986; Barto et al., 1983). The objective in this problem is to apply forces to a wheeled cart moving along a limited track in order to keep one or more poles hinged to the cart from falling over. There are several variations of the task with different levels of difficulty; among them, balancing two poles side by side is particularly hard (Wieland, 1991). In this paper we compare LSPI and KBSF on both the single- and two-poles versions of the problem. We implemented the tasks using a realistic simulator described by Gomez (2003). We refer the reader to Appendix B for details on the problems's configuration.

The experiments were carried out as described in the previous section, with sample transitions collected by a random policy and then clustered by the k-means algorithm. In both versions of the pole-balancing task LSPI used the same data and approximation architectures as KBSF. To make the comparison with LSPI as fair as possible, we fixed the width of KBSF's kernel κ_τ^a at $\tau = 1$ and varied $\bar{\tau}$ in $\{0.01, 0.1, 1\}$ for both algorithms. Also, policy iteration was used to find a decision policy for the MDPs constructed by KBSF, and this algorithm was run for a maximum of 30 iterations, the same limit used for LSPI.

Figure 4 shows the results of LSPI and KBSF on the single and double pole-balancing tasks. We call attention to the fact that the version of the problems used here is significantly harder than the more commonly-used variants in which the decision policies are evaluated on a single state close to the origin. This is probably the reason why LSPI achieves a



(a) Performance as a function of m

(b) Run time as a function of m

(c) Performance as a function of n

(d) Run time as a function of n

Figure 3: Results on the puddle-world task averaged over 50 runs. The algorithms were evaluated on a set of test states distributed over a region of the state space surrounding the "puddles" (details in Appendix B). The shadowed regions represent 99% confidence intervals.

success rate of no more than 60% on the single pole-balancing task, as shown in Figure 4a. In contrast, KBSF's decision policies are able to balance the pole in 90% of the attempts, on average, using as few as $m = 30$ representative states.

The results of KBSF on the double pole-balancing task are still more impressive. As Wieland (1991) rightly points out, this version of the problem is considerably more difficult than its single pole variant, and previous attempts to apply reinforcement-learning techniques to this domain resulted in disappointing performance (Gomez et al., 2006). As shown in Figure 4c, KBSF(10^6, 200) is able to achieve a success rate of more than 80%. To put this number in perspective, recall that some of the test states are quite challenging, with the two poles inclined and falling in opposite directions.

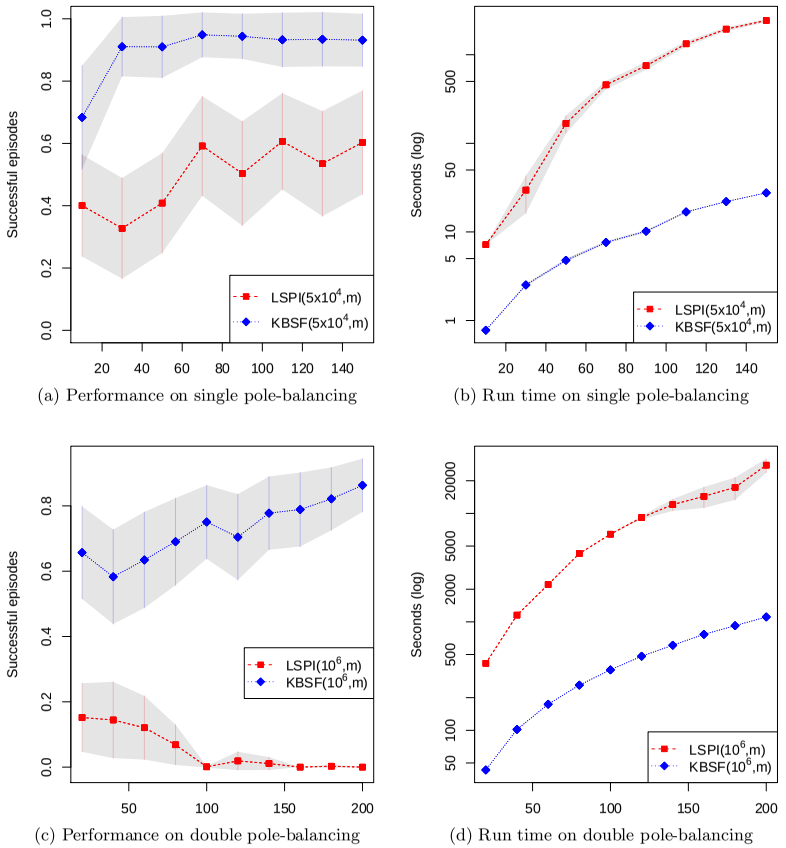
The good performance of KBSF comes at a relatively low computational cost. A conservative estimate reveals that, were KBRL(10^6) run on the same computer used for these experiments, we would have to wait for more than 6 *months* to see the results. KBSF(10^6, 200) delivers a decision policy in less than 7 minutes. KBSF's computational cost also compares well with that of LSPI, as shown in Figures 4b and 4d. LSPI's policy evaluation step involves the update and solution of a linear system of equations, which take $O(nm^2)$ and $O(m^3|A|^3)$, respectively. In addition, the policy-update stage requires the definition of $\pi(\hat{s}_k^a)$ for all n states in the set of sample transitions. In contrast, at each iteration KBSF only performs $O(m^3)$ operations to evaluate a decision policy and $O(m^2|A|)$ operations to update it.

4.2.3 HIV drug schedule domain (comparison with fitted Q-iteration)

We now compare KBSF with the fitted Q-iteration algorithm (Ernst et al., 2005; Antos et al., 2007; Munos and Szepesvári, 2008). Fitted Q-iteration is a conceptually simple method that also builds its approximation based solely on sample transitions. Here we adopt this algorithm with an ensemble of trees generated by Geurts et al.'s (2006) extra-trees algorithm. This combination, which we call FQIT, generated the best results on the extensive empirical evaluation performed by Ernst et al. (2005).

We chose FQIT for our comparisons because it has shown excellent performance on both benchmark and real-world reinforcement-learning tasks (Ernst et al., 2005, 2006). In all experiments reported in this paper we used FQIT with ensembles of 30 trees. As detailed in Appendix B, besides the number of trees, FQIT has three main parameters. Among them, the minimum number of elements required to split a node in the construction of the trees, denoted here by $\eta_{\min}$, has a particularly strong effect on both the algorithm's performance and computational cost. Thus, in our experiments we fixed FQIT's parameters at reasonable values—selected based on preliminary experiments—and only varied $\eta_{\min}$. The respective instances of the tree-based approach are referred to as FQIT($\eta_{\min}$).

We compare FQIT and KBSF on an important medical problem which we will refer to as the HIV drug schedule domain (Adams et al., 2004; Ernst et al., 2006). Typical HIV treatments use drug cocktails containing two types of medication: reverse transcriptase inhibitors (RTI) and protease inhibitors (PI). Despite the success of drug cocktails in maintaining low viral loads, there are several complications associated with their long-term use. This has attracted the interest of the scientific community to the problem of optimizing drug-scheduling strategies. One strategy that has been receiving a lot of attention recently



(a) Performance on single pole-balancing
(b) Run time on single pole-balancing
(c) Performance on double pole-balancing
(d) Run time on double pole-balancing

Figure 4: Results on the pole-balancing tasks, as a function of the number of representative states m, averaged over 50 runs. The values correspond to the fraction of episodes initiated from the test states in which the pole(s) could be balanced for 3000 steps (one minute of simulated time). The test sets were regular grids defined over the hypercube centered at the origin and covering 50% of the state-space axes in each dimension (see Appendix B). Shadowed regions represent 99% confidence intervals.

is structured treatment interruption (STI), in which patients undergo alternate cycles with and without the drugs. Although many successful STI treatments have been reported in the literature, as of now there is no consensus regarding the exact protocol that should be followed (Bajaria et al., 2004).

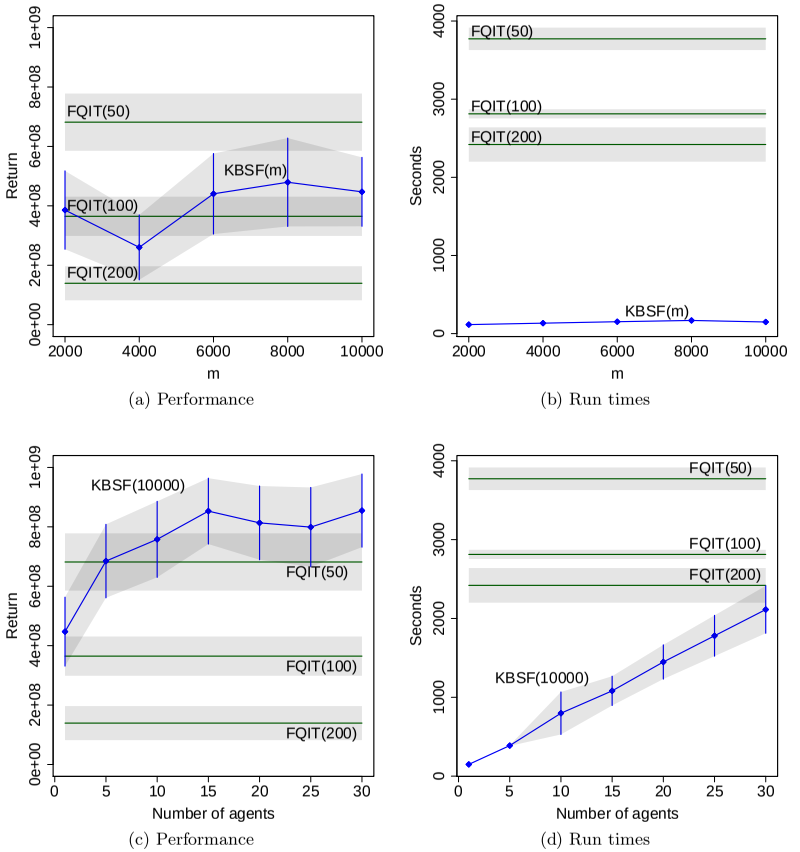
The scheduling of STI treatments can be seen as a sequential decision problem in which the actions correspond to the types of cocktail that should be administered to a patient (Ernst et al., 2006). To simplify the problem's formulation, it is assumed that RTI and PI drugs are administered at fixed amounts, reducing the actions to the four possible combinations of drugs: none, RTI only, PI only, or both. The goal is to minimize the viral load using as little drugs as possible. Following Ernst et al. (2006), we performed our experiments using a model that describes the interaction of the immune system with HIV. This model was developed by Adams et al. (2004) and has been identified and validated based on real clinical data. The resulting reinforcement learning task has a 6-dimensional continuous state space whose variables describe the overall patient's condition.

We formulated the problem exactly as proposed by Ernst et al. (2006, see Appendix B for details). The strategy used to generate the data also followed the protocol proposed by these authors, which we now briefly explain. Starting from a batch of 6000 sample transitions generated by a random policy, each algorithm first computed an initial approximation of the problem's optimal value function. Based on this approximation, a 0.15-greedy policy was used to collect a second batch of 6000 transitions, which was merged with the first.[3] This process was repeated for 10 rounds, resulting in a total of 60000 sample transitions.

We varied FQIT's parameter $\eta_{\min}$ in the set $\{50, 100, 200\}$. For the experiments with KBSF, we fixed $\tau = \bar{\tau} = 1$ and varied m in $\{2000, 4000, ..., 10000\}$ (in the rounds in which $m \geq n$ we simply used all states $\hat{s}_i^a$ as representative states). As discussed in the beginning of this section, it is possible to reduce KBSF's computational cost with the use of sparse kernels. In our experiments with the HIV drug schedule task, we only computed the $\mu = 2$ largest values of $\mathrm{k}_\tau(\bar{s}_i, \cdot)$ and the $\bar{\mu} = 3$ largest values of $\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \cdot)$ (see Appendix B.2). The representative states $\bar{s}_i$ were selected at random from the set of sampled states $\hat{s}_i^a$ (the reason for this will become clear shortly). Since in the current experiments the number of sample transitions n was fixed, we will refer to the particular instances of our algorithm simply as KBSF(m).

Before discussing the results, we point out that FQIT's performance on the HIV drug schedule task is very good, comparable to that of Adams et al.'s (2004) approach, which uses a model of the task. FQIT's results, along with KBSF's, are shown in Figure 5. As shown in Figure 5a, the performance of FQTI improves when $\eta_{\min}$ is decreased, as expected. In contrast, increasing the number of representative states m does not have a strong impact on the quality of KBSF's solutions (in fact, in some cases the average return obtained by the resulting policies decreases slightly when m grows). Overall, the performance of KBSF on the HIV drug schedule task is not nearly as impressive as on the previous problems. For example, even when using $m = 10000$ representative states, which corresponds to one sixth of the sampled states, KBSF is unable to reproduce the performance of FQIT with $\eta_{\min} = 50$.

3. As explained by Sutton and Barto (1998), an ϵ-greedy policy selects the action with maximum value with probability $1 - \epsilon$, and with probability ϵ it picks an action uniformly at random.



(a) Performance (b) Run times

(c) Performance (d) Run times

Figure 5: Results on the HIV drug schedule task averaged over 50 runs. The STI policies were evaluated for 5000 days starting from a state representing a patient's unhealthy state (see Appendix B). The shadowed regions represent 99% confidence intervals.

On the other hand, when we look at Figure 5b, it is clear that the difference on the algorithms's performance is counterbalanced by a substantial difference on the associated computational costs. As an illustration, note that KBSF(10000) is 15 times faster than FQTI(100) and 20 times faster than FQTI(50). This difference on the algorithms's run times is expected, since each iteration of FQIT involves the construction (or update) of an ensemble of trees, each one requiring at least $O(n \log(n/\eta_{\min}))$ operations, and the improvement of the current decision policy, which is $O(n|A|)$ (Geurts et al., 2006). As discussed before, KBSF's efficiency comes from the fact that its computational cost per iteration is independent of the number of sample transitions n.

Note that the fact that FQIT uses an ensemble of trees is both a blessing and a curse. If on the one hand this reduces the variance of the approximation, on the other hand it also increases the algorithm's computational cost (Geurts et al., 2006). Given the big gap between FQIT's and KBSF's time complexities, one may wonder if the latter can also benefit from averaging over several models. In order to verify this hypothesis, we implemented a very simple model-averaging strategy with KBSF: we trained several agents independently, using Algorithm 1 on the same set of sample transitions, and then put them together on a single "committee". In order to increase the variability within the committee of agents, instead of using k-means to determine the representative states $\bar{s}_j$ we simply selected them uniformly at random from the set of sampled states $\hat{s}_i^a$ (note that this has the extra benefit of reducing the method's overall computational cost). The actions selected by the committee of agents were determined through "voting"—that is, we simply picked the action chosen by the majority of agents, with ties broken randomly.

We do not claim that the approach described above is the best model-averaging strategy to be used with KBSF. However, it seems to be sufficient to boost the algorithm's performance considerably, as shown in Figure 5c. Note how KBSF already performs comparably to FQTI(50) when using only 5 agents in the committee. When this number is increased to 15, the expected return of KBSF's agents is considerably larger than that of the best FQIT's agent, with only a small overlap between the 99% confidence intervals associated with the algorithms's results. The good performance of KBSF is still more impressive when we look at Figure 5d, which shows that even when using a committee of 30 agents this algorithm is faster than FQIT(200).

4.2.4 Epilepsy-suppression domain (comparison with LSPI and fitted Q-iteration)

We conclude our empirical evaluation of KBSF by using it to learn a neuro-stimulation policy for the treatment of epilepsy. It has been shown that the electrical stimulation of specific structures in the neural system at fixed frequencies can effectively suppress the occurrence of seizures (Durand and Bikson, 2001). Unfortunately, *in vitro* neuro-stimulation experiments suggest that fixed-frequency pulses are not equally effective across epileptic systems. Moreover, the long term use of this treatment may potentially damage the patients's neural tissues. Therefore, it is desirable to develop neuro-stimulation policies that replace the fixed-stimulation regime with an adaptive scheme.

The search for efficient neuro-stimulation strategies can be seen as a reinforcement learning problem. Here we study this problem using a generative model developed by Bush et al.

(2009) based on real data collected from epileptic rat hippocampus slices. Bush et al.'s model was shown to reproduce the seizure pattern of the original dynamical system and was later validated through the deployment of a learned treatment policy on a real brain slice (Bush and Pineau, 2009). The associated decision problem has a five-dimensional continuous state space and highly non-linear dynamics. At each time step the agent must choose whether or not to apply an electrical pulse. The goal is to suppress seizures as much as possible while minimizing the total amount of stimulation needed to do so.

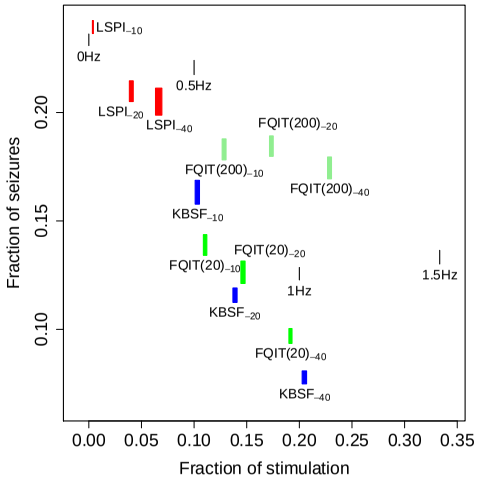
The experiments were performed as described in Section 4.2.1, with a single batch of sample transitions collected by a policy that selects actions uniformly at random. Specifically, the random policy was used to collect 50 trajectories of length 10000, resulting in a total of 500000 sample transitions. We use as a baseline for our comparisons the already mentioned fixed-frequency stimulation policies usually adopted in *in vitro* clinical studies (Bush and Pineau, 2009). In particular, we considered policies that apply electrical pulses at frequencies of 0 Hz, 0.5 Hz, 1 Hz, and 1.5 Hz.

We compare KBSF with LSPI and FQIT. For this task we ran both LSPI and KBSF with sparse kernels, that is, we only computed the kernels at the 6-nearest neighbors of a given state ($\mu = \bar{\mu} = 6$; see Appendix B.2 for details). This modification made it possible to use $m = 50000$ representative states with KBSF. Since for LSPI the reduction on the computational cost was not very significant, we fixed $m = 50$ to keep its run time within reasonable bounds. Again, KBSF and LSPI used the same approximation architectures, with representative states defined by the k-means algorithm. We fixed $\tau = 1$ and varied $\bar{\tau}$ in $\{0.01, 0.1, 1\}$. FQIT was configured as described in the previous section, with the parameter $\eta_{\min}$ varying in $\{20, 30, ..., 200\}$. In general, we observed that the performance of the tree-based method improved with smaller values for $\eta_{\min}$, with an expected increase in the computational cost. Thus, in order to give an overall characterization of FQIT's performance, we only report the results obtained with the extreme values of $\eta_{\min}$.

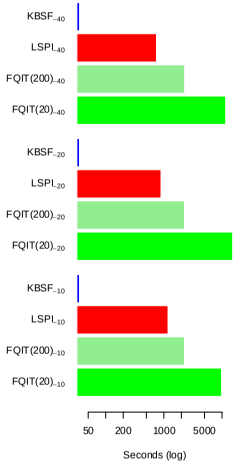
Figure 6 shows the results on the epilepsy-suppression task. In order to obtain different compromises between the problem's two conflicting objectives, we varied the relative magnitude of the penalties associated with the occurrence of seizures and with the application of an electrical pulse (Bush et al., 2009; Bush and Pineau, 2009). Specifically, we fixed the latter at -1 and varied the former with values in $\{-10, -20, -40\}$. This appears in the plots as subscripts next to the algorithms's names. As shown in Figure 6a, LSPI's policies seem to prioritize reduction of stimulation at the expense of higher seizure occurrence, which is clearly sub-optimal from a clinical point of view. FQIT(200) also performs poorly, with solutions representing no advance over the fixed-frequency stimulation strategies. In contrast, FQTI(20) and KBSF are both able to generate decision policies that are superior to the 1 Hz policy, which is the most efficient stimulation regime known to date in the clinical literature (Jerger and Schiff, 1995). However, as shown in Figure 6b, KBSF is able to do it at least 100 times faster than the tree-based method.

5. Incremental KBSF

As clear in the previous section, one characteristic of KBSF that sets it apart from other methods is its low demand in terms of computational resources. Specifically, both time and memory complexities of our algorithm are linear in the number of sample transitions n. In



(a) Performance. The length of the rectangles's edges represent 99% confidence intervals.



(b) Run times (confidence intervals do not show up in logarithmic scale)

Figure 6: Results on the epilepsy-suppression problem averaged over 50 runs. The decision policies were evaluated on episodes of 10^5 transitions starting from a fixed set of 10 test states drawn uniformly at random.

terms of the number of operations performed by the algorithm, this is the best one can do without discarding transitions. However, in terms of memory usage, it is possible to do even better. In this section we show how to build KBSF's approximation incrementally, without ever having access to the entire set of sample transitions at once. Besides reducing the memory complexity of the algorithm, this modification has the additional advantage of making KBSF suitable for on-line reinforcement learning.

In the batch version of KBSF, described in Section 4, the matrices $\bar{\mathbf{P}}^a$ and vectors $\bar{\mathbf{r}}^a$ are determined using all the transitions in the corresponding sets S^a. This has two undesirable consequences. First, the construction of the MDP $\bar{M}$ requires an amount of memory of $O(\hat{n}m)$, where $\hat{n} = \max_a n_a$. Although this is a significant improvement over KBRL's memory usage, which is lower bounded by $(\min_a n_a)^2|A|$, in more challenging domains even a linear dependence on $\hat{n}$ may be impractical. Second, in the batch version of KBSF the only way to incorporate new data into the model $\bar{M}$ is to recompute the multiplication $\bar{\mathbf{P}}^a = \dot{\mathbf{K}}^a\dot{\mathbf{D}}^a$ for all actions a for which there are new sample transitions available. Even if we ignore the issue with memory usage, this is clearly inefficient in terms of computation. In what follows we present an incremental version of KBSF that circumvents these important limitations (Barreto et al., 2012).

We assume the same scenario considered in Section 4: there is a set of sample transitions $S^a = \{(s^a_k, r^a_k, \hat{s}^a_k)|k = 1, 2, ..., n_a\}$ associated with each action $a \in A$, where $s^a_k, \hat{s}^a_k \in \mathbb{S}$ and $r^a_k \in \mathbb{R}$, and a set of representative states $\bar{S} = \{\bar{s}_1, \bar{s}_2, ..., \bar{s}_m\}$, with $\bar{s}_i \in \mathbb{S}$. Suppose now that we split the set of sample transitions S^a in two subsets S_1 and S_2 such that $S_1 \cap S_2 = \emptyset$ and $S_1 \cup S_2 = S^a$ (we drop the "a" superscript in the sets S_1 and S_2 to improve clarity). Without loss of generality, suppose that the sample transitions are indexed so that $S_1 \equiv \{(s^a_k, r^a_k, \hat{s}^a_k)|k = 1, 2, ..., n_1\}$ and $S_2 \equiv \{(s^a_k, r^a_k, \hat{s}^a_k)|k = n_1 + 1, n_1 + 2, ..., n_1 + n_2 = n_a\}$. Let $\bar{\mathbf{P}}^{S_1}$ and $\bar{\mathbf{r}}^{S_1}$ be matrix $\bar{\mathbf{P}}^a$ and vector $\bar{\mathbf{r}}^a$ computed by KBSF using only the n_1 transitions in S_1 (if $n_1 = 0$, we define $\bar{\mathbf{P}}^{S_1} = \mathbf{0} \in \mathbb{R}^{m\times m}$ and $\bar{\mathbf{r}}^{S_1} = \mathbf{0} \in \mathbb{R}^m$ for all $a \in A$). We want to compute $\bar{\mathbf{P}}^{S_1\cup S_2}$ and $\bar{\mathbf{r}}^{S_1\cup S_2}$ from $\bar{\mathbf{P}}^{S_1}$, $\bar{\mathbf{r}}^{S_1}$, and S_2, without using the set of sample transitions S_1.

We start with the transition matrices $\bar{\mathbf{P}}^a$. We know that

$$\begin{aligned}\bar{p}^{S_1}_{ij} &= \textstyle\sum_{t=1}^{n_1} \dot{k}^a_{it}\dot{d}^a_{tj} = \sum_{t=1}^{n_1} \frac{\mathrm{k}_\tau(\bar{s}_i, s^a_t)}{\sum_{l=1}^{n_1}\mathrm{k}_\tau(\bar{s}_i, s^a_l)} \frac{\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}^a_t, \bar{s}_j)}{\sum_{l=1}^{m}\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}^a_t, \bar{s}_l)} \\ &= \frac{1}{\sum_{l=1}^{n_1}\mathrm{k}_\tau(\bar{s}_i, s^a_l)} \sum_{t=1}^{n_1} \frac{\mathrm{k}_\tau(\bar{s}_i, s^a_t)\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}^a_t, \bar{s}_j)}{\sum_{l=1}^{m}\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}^a_t, \bar{s}_l)}.\end{aligned}$$

To simplify the notation, define

$$z^{S_1}_i = \sum_{l=1}^{n_1}\mathrm{k}_\tau(\bar{s}_i, s^a_l),\ z^{S_2}_i = \sum_{l=n_1+1}^{n_1+n_2}\mathrm{k}_\tau(\bar{s}_i, s^a_l),\ \text{and}\ b^t_{ij} = \frac{\mathrm{k}_\tau(\bar{s}_i, s^a_t)\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}^a_t, \bar{s}_j)}{\sum_{l=1}^{m}\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}^a_t, \bar{s}_l)},$$

with $t \in \{1, 2, ..., n_1 + n_2\}$. Then, we can write

$$\bar{p}^{S_1\cup S_2}_{ij} = \frac{1}{z^{S_1}_i + z^{S_2}_i}\left(\textstyle\sum_{t=1}^{n_1} b^t_{ij} + \sum_{t=n_1+1}^{n_1+n_2} b^t_{ij}\right) = \frac{1}{z^{S_1}_i + z^{S_2}_i}\left(\bar{p}^{S_1}_{ij} z^{S_1}_i + \textstyle\sum_{t=n_1+1}^{n_1+n_2} b^t_{ij}\right).$$

Now, defining $b_{ij}^{S_2} = \sum_{t=n_1+1}^{n_1+n_2} b_{ij}^t$, we have the simple update rule:

$$\boxed{\bar{p}_{ij}^{S_1 \cup S_2} = \frac{1}{z_i^{S_1} + z_i^{S_2}} \left(b_{ij}^{S_2} + \bar{p}_{ij}^{S_1} z_i^{S_1} \right)}. \tag{19}$$

We can apply similar reasoning to derive an update rule for the rewards $\bar{r}_i^a$. We know that

$$\bar{r}_i^{S_1} = \frac{1}{\sum_{l=1}^{n_1} \mathrm{k}_\tau(\bar{s}_i, s_l^a)} \sum_{t=1}^{n_1} \mathrm{k}_\tau(\bar{s}_i, s_t^a) r_t^a = \frac{1}{z_i^{S_1}} \sum_{t=1}^{n_1} \mathrm{k}_\tau(\bar{s}_i, s_t^a) r_t^a.$$

Let $e_i^t = \mathrm{k}_\tau(\bar{s}_i, s_t^a) r_t^a$, with $t \in \{1, 2, ..., n_1 + n_2\}$. Then,

$$\bar{r}_i^{S_1 \cup S_2} = \frac{1}{z_i^{S_1} + z_i^{S_2}} \left(\sum_{t=1}^{n_1} e_i^t + \sum_{t=n_1+1}^{n_1+n_2} e_i^t \right) = \frac{1}{z_i^{S_1} + z_i^{S_2}} \left(z_i^{S_1} \bar{r}_i^{S_1} + \sum_{t=n_1+1}^{n_1+n_2} e_i^t \right).$$

Defining $e_i^{S_2} = \sum_{t=n_1+1}^{n_1+n_2} e_i^t$, we have the following update rule:

$$\boxed{\bar{r}_i^{S_1 \cup S_2} = \frac{1}{z_i^{S_1} + z_i^{S_2}} \left(e_i^{S_2} + \bar{r}_i^{S_1} z_i^{S_1} \right)}. \tag{20}$$

Since $b_{ij}^{S_2}$, $e_i^{S_2}$, and $z_i^{S_2}$ can be computed based on S_2 only, we can discard the sample transitions in S_1 after computing $\bar{\mathbf{P}}^{S_1}$ and $\bar{\mathbf{r}}^{S_1}$. In order to do that, we only have to keep the variables $z_i^{S_1}$. These variables can be stored in $|A|$ vectors $\mathbf{z}^a \in \mathbb{R}^m$, resulting in a modest memory overhead. Note that we can apply the ideas above recursively, further splitting the sets S_1 and S_2 in subsets of smaller size. Thus, we have a fully incremental way of computing KBSF's MDP which requires almost no extra memory.

Algorithm 2 shows a step-by-step description of how to update $\bar{M}$ based on a set of sample transitions. Using this method to update its model, KBSF's space complexity drops from $O(\hat{n}m)$ to $O(m^2)$. Since the amount of memory used by KBSF is now independent of n, it can process an arbitrary number of sample transitions (or, more precisely, the limit on the amount of data it can process is dictated by time only, not space).

Instead of assuming that S_1 and S_2 are a partition of a fixed data set S^a, we can consider that S_2 was generated based on the policy learned by KBSF using the transitions in S_1. Thus, Algorithm 2 provides a flexible framework for integrating learning and planning within KBSF. Specifically, our algorithm can cycle between learning a model of the problem based on sample transitions, using such a model to derive a policy, and resorting to this policy to collect more data. Algorithm 3 shows a possible implementation of this framework. In order to distinguish it from its batch counterpart, we will call the incremental version of our algorithm *i*KBSF. *i*KBSF updates the model $\bar{M}$ and the value function $\bar{\mathbf{Q}}$ at fixed intervals t_m and t_v, respectively. When $t_m = t_v = n$, we recover the batch version of KBSF; when $t_m = t_v = 1$, we have a fully on-line method which stores no sample transitions.

Algorithm 3 allows for the inclusion of new representative states to the model $\bar{M}$. Using Algorithm 2 this is easy to do: given a new representative state $\bar{s}_{m+1}$, it suffices to set $z_{m+1}^a = 0$, $\bar{r}_{m+1}^a = 0$, and $\bar{p}_{m+1,j} = \bar{p}_{j,m+1} = 0$ for $j = 1, 2, ..., m+1$ and all $a \in A$. Then, in the following applications of update rules (19) and (20), the dynamics of $\bar{M}$ will naturally reflect the existence of state $\bar{s}_{m+1}$. Note that the inclusion of new representative states

Algorithm 2 Update KBSF's MDP

Input: $\bar{\mathbf{P}}^a$, $\bar{\mathbf{r}}^a$, $\mathbf{z}^a$ for all $a \in A$ ▷ Current model
$S^a = \{(s_k^a, r_k^a, \hat{s}_k^a) | k = 1, 2, ..., n_a\}$ for all $a \in A$ ▷ Sample transitions
Output: Updated $\bar{M}$ and $\mathbf{z}^a$

for $a \in A$ **do**
  for $t = 1, ..., n_a$ **do** $\bar{z}_t \leftarrow \sum_{l=1}^m \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_t^a, \bar{s}_l)$
  $n_a \leftarrow |S^a|$
  for $i = 1, 2, ..., m$ **do**
    $z' \leftarrow \sum_{t=1}^{n_a} \mathrm{k}_\tau(\bar{s}_i, s_t^a)$
    for $j = 1, 2, ..., m$ **do**
      $b \leftarrow \sum_{t=1}^{n_a} \mathrm{k}_\tau(\bar{s}_i, s_t^a) \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_t^a, \bar{s}_j) / \bar{z}_t$
      $\bar{p}_{ij} \leftarrow \frac{1}{z_i^a + z'} (b + \bar{p}_{ij} z_i^a)$ ▷ Update transition probabilities
    $e \leftarrow \sum_{t=1}^{n_a} \mathrm{k}_\tau(\bar{s}_i, s_t^a) r_t^a$
    $\bar{r}_i \leftarrow \frac{1}{z_i^a + z'} (e + \bar{r}_i z_i^a)$ ▷ Update rewards
    $z_i^a \leftarrow z_i^a + z'$ ▷ Update normalization factor

Algorithm 3 Incremental KBSF (*i*KBSF)

$\bar{S} = \{\bar{s}_1, \bar{s}_2, ..., \bar{s}_m\}$ ▷ Set of representative states
Input: t_m ▷ Interval to update model
t_v ▷ Interval to update value function
Output: Approximate value function $\tilde{Q}(s, a)$

$\bar{\mathbf{P}}^a \leftarrow \mathbf{0} \in \mathbb{R}^{m \times m}$, $\bar{\mathbf{r}}^a \leftarrow \mathbf{0} \in \mathbb{R}^m$, $\mathbf{z}^a \leftarrow \mathbf{0} \in \mathbb{R}^m$, for all $a \in A$
$\bar{\mathbf{Q}} \leftarrow$ arbitrary matrix in $\mathbb{R}^{m \times |A|}$
$s \leftarrow$ initial state
$a \leftarrow$ random action
for $t \leftarrow 1, 2, ...$ **do**
  Execute a in s and observe r and $\hat{s}$
  $S^a \leftarrow S^a \bigcup \{(s, r, \hat{s})\}$
  if $(t \mod t_m = 0)$ **then** ▷ Update model
    Add new representative states to $\bar{M}$ using S^a ▷ This step is optional
    Update $\bar{M}$ and $\mathbf{z}^a$ using Algorithm 2 and S^a
    $S^a \leftarrow \emptyset$ for all $a \in A$ ▷ Discard transitions
  if $(t \mod t_v = 0)$ update $\bar{\mathbf{Q}}$ ▷ Update value function
  $s \leftarrow \hat{s}$
  Select a based on $\tilde{Q}(s, a) = \sum_{i=1}^m \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_i) \bar{q}_{ia}$

does not destroy the information already in the model. This allows *i*KBSF to refine its approximation on the fly, as needed. One can think of several ways of detecting the need for new representative states. A simple strategy, based on Proposition 6, is to impose a maximum distance allowed between a sampled state $\hat{s}_i^a$ and the nearest representative state, $dist(\hat{s}_i^a, 1)$. So, anytime the agent encounters a new state $\hat{s}_i^a$ for which $dist(\hat{s}_i^a, 1)$ is above a given threshold, $\hat{s}_i^a$ is added to the model as $\bar{s}_{m+1}$. In Section 5.2 we report experiments with *i*KBSF using this approach. Before that, though, we discuss the theoretical properties of the incremental version of our algorithm.

5.1 Theoretical results

As discussed, *i*KBSF does not need to store sample transitions to build its approximation. However, the computation of $\tilde{Q}(s,a)$ through (12) requires all the tuples $(s_i^a, r_i^a, \hat{s}_i^a)$ to be available. In some situations, it may be feasible to keep the transitions in order to compute $\tilde{Q}(s,a)$. However, if we want to use *i*KBSF to its full extend, we need a way of computing $\tilde{Q}(s,a)$ without using the sample transitions. This is why upon reaching state s at time step t *i*KBSF selects the action to be performed based on

$$\tilde{Q}_t(s,a) = \sum_{i=1}^{m} \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_i) \bar{Q}_t(\bar{s}_i, a), \tag{21}$$

where $\bar{Q}_t(\bar{s}_i, a)$ is the action-value function available to *i*KBSF at the t^{th} iteration (see Algorithm 3). Note that we do not assume that *i*KBSF has computed the optimal value function of its current model $\bar{M}_t$—that is, it may be the case that $\bar{Q}_t(\bar{s}_i, a) \neq \bar{Q}_t^*(\bar{s}_i, a)$.

Unfortunately, when we replace (12) with (21) Proposition 3 no longer applies. In this section we address this issue by deriving an upper bound for the difference between $\tilde{Q}_t(s,a)$ and $\hat{Q}_t(s,a)$, the action-value function that would be computed by KBRL using all the transitions processed by *i*KBSF up to time step t. In order to derive our bound, we assume that *i*KBSF uses a fixed set $\bar{S}$—meaning that no representative states are added to the model $\bar{M}$—and that it never stops refining its model, doing so at every iteration t (*i.e.*, $t_m = 1$ in Algorithm 3). We start by showing the following lemma:

Lemma 9 *Let $M \equiv (S, A, \mathbf{P}^a, \mathbf{r}^a, \gamma)$ and $\tilde{M} \equiv (S, A, \tilde{\mathbf{P}}^a, \tilde{\mathbf{r}}^a, \gamma)$ be two finite MDPs. Then, for any $s \in S$ and any $a \in A$,*

$$|Q^*(s,a) - \tilde{Q}^*(s,a)| \leq \frac{1}{1-\gamma} \max_a \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \frac{\gamma}{2(1-\gamma)^2} R_{\text{dif}} \max_a \|\mathbf{P}^a - \tilde{\mathbf{P}}^a\|_\infty,$$

where $R_{\text{dif}} = \max_{a,i} r_i^a - \min_{a,i} r_i^a$.

Lemma 9 provides an upper bound for the difference in the action-value functions of any two MDPs having the same state space S, action space A, and discount factor γ.[4] Our strategy will be to use this result to bound the error introduced by the application of the stochastic-factorization trick in the context of *i*KBSF.

4. Strehl and Littman's (2008) Lemma 1 is similar to our result. Their bound is more general than ours, as it applies to any Q^π, but it is also slightly looser.

When $t_m = 1$, at any time step t *i*KBSF has a model $\bar{M}_t$ built based on the t transitions observed thus far. As shown in the beginning of this section, $\bar{M}_t$ exactly matches the model that would be computed by batch KBSF using the same data and the same set of representative states. Thus, we can think of matrices $\bar{\mathbf{P}}_t^a$ and vectors $\bar{\mathbf{r}}_t^a$ available at the t^{th} iteration of *i*KBSF as the result of the stochastic-factorization trick applied with matrices $\mathbf{D}_t$ and $\mathbf{K}_t^a$. Although *i*KBSF does not explicitly compute such matrices, they serve as a theoretical foundation to build our result on.

Proposition 10 *Suppose iKBSF is executed with a fixed set of representative states $\bar{S}$ using $t_m = 1$. Let $\mathbf{D}_t$, $\mathbf{K}_t^a$ and $\bar{\mathbf{r}}_t^a$ be the matrices and the vector (implicitly) computed by this algorithm at iteration t. Then, if s is the state encountered by iKBSF at time step t,*

$$|\hat{Q}_t(s,a) - \tilde{Q}_t(s,a)| \leq \frac{1}{1-\gamma} \max_a \|\hat{\mathbf{r}}_t^a - \mathbf{D}_t \bar{\mathbf{r}}_t^a\|_\infty + \frac{\bar{R}_{\text{dif},t}}{(1-\gamma)^2} \left(\frac{\gamma}{2} \max_a \|\hat{\mathbf{P}}_t^a - \mathbf{D}_t \mathbf{K}_t^a\|_\infty + \sigma(\mathbf{D}_t) \right) + \epsilon_{\bar{Q}_t}, \tag{22}$$

for any $a \in A$, where $\tilde{Q}_t$ is the value function computed by iKBSF at time step t through (21), $\hat{Q}_t$ is the value function computed by KBRL through (7) based on the same data, $\bar{R}_{\text{dif},t} = \max_{a,i} \bar{r}_{i,t}^a - \min_{a,i} \bar{r}_{i,t}^a$, $\sigma(\mathbf{D}_t) = \max_i (1 - \max_j d_{ij,t})$, and $\epsilon_{\bar{Q}_t} = \max_{i,a} |\bar{Q}_t^*(\bar{s}_i, a) - \bar{Q}_t(\bar{s}_i, a)|$.

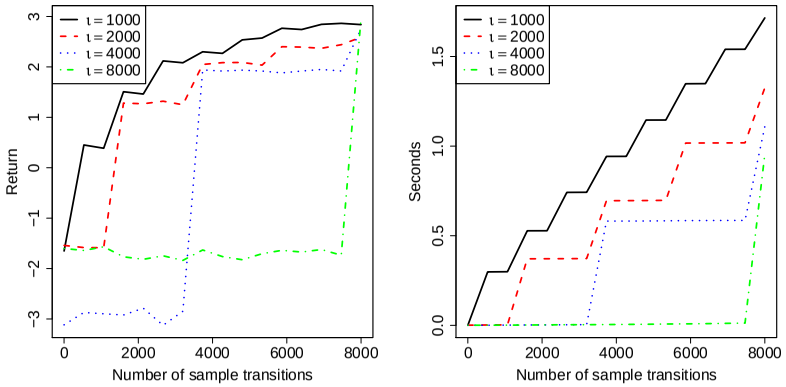
Proposition 10 shows that, at any time step t, the error in the action-value function computed by *i*KBSF is bounded above by the quality and the level of stochasticity of the stochastic factorization implicitly computed by the algorithm. The term $\epsilon_{\bar{Q}_t}$ accounts for the possibility that *i*KBSF has not computed the optimal value function of its model at step t, either because $t_m \neq t_v$ or because the update of $\bar{\mathbf{Q}}$ in Algorithm 3 is not done to completion (for example, one can apply the Bellman operator $\bar{T}$ a fixed number of times, stopping short of convergence). The restriction $t_m = 1$ is not strictly necessary if we are willing to compare $\tilde{Q}_t(s,a)$ with $\hat{Q}_{t'}(s,a)$, where $t' = \lfloor (t + t_m)/t \rfloor$ (the next time step scheduled for a model update). However, such a result would be somewhat circular, since the sample transitions used to build $\hat{Q}_{t'}(s,a)$ may depend on $\tilde{Q}_t(s,a)$. Finally, we note that, given the similarity between the right-hand sides of (8) and (22), it should be trivial to derive results analogous to Propositions 6 and 8 that also apply to *i*KBSF.

5.2 Empirical results

We now look at the empirical performance of the incremental version of KBSF. Following the structure of Section 4.2, we start with the puddle world task to show that *i*KBSF is indeed able to match the performance of batch KBSF without storing all sample transitions. Next we exploit the scalability of *i*KBSF to solve two difficult control tasks, triple pole-balancing and helicopter hovering. We also compare *i*KBSF's performance with that of other reinforcement learning algorithms.

5.2.1 Puddle world (proof of concept)

We use the puddle world problem as a proof of concept (Sutton, 1996). In this first experiment we show that *i*KBSF is able to recover the model that would be computed by its batch counterpart. In order to do so, we applied Algorithm 3 to the puddle-world task using a random policy to select actions.



(a) Performance (b) Run times

Figure 7: Results on the puddle-world task averaged over 50 runs. KBSF used 100 representative states evenly distributed over the state space and $t_m = t_v = \iota$ (see legends). Sample transitions were collected by a random policy. The agents were tested on two sets of states surrounding the "puddles" (see Appendix B).

Figure 7 shows the result of the experiment when we vary the parameters t_m and t_v. Note that the case in which $t_m = t_v = 8000$ corresponds to the batch version of KBSF, whose results on the puddle world are shown in Figure 3. As expected, the performance of KBSF policies improves gradually as the algorithm goes through more sample transitions, and in general the intensity of the improvement is proportional to the amount of data processed. More important, the performance of the decision policies after all sample transitions have been processed is essentially the same for all values of t_m and t_v, which confirms that *i*KBSF can be used as an instrument to circumvent KBSF's memory demand. Thus, if one has a batch of sample transitions that does not fit in the available memory, it is possible to split the data in chunks of smaller sizes and still get the same value-function approximation that would be computed if the entire data set were processed at once. As shown in Figure 7b, there is only a small computational overhead associated with such a strategy (this results from unnormalizing and normalizing the elements of $\bar{\mathbf{P}}^a$ and $\bar{\mathbf{r}}^a$ multiple times through update rules (19) and (20)).

5.2.2 Triple pole-balancing (comparison with fitted Q-iteration)

As discussed in Section 4.2.2, the pole balancing task has been addressed in several different versions, and among them simultaneously balancing two poles is particularly challenging (Wieland, 1991). Figures 4c and 4d show that the batch version of KBSF was able to satisfactorily solve the double pole-balancing task. In order to show the scalability of the incremental version of our algorithm, in this section we raise the bar, adding a third pole

to the problem. We perform our simulations using the parameters usually adopted with the two-pole problem, with the extra pole having the same length and mass as the longer pole (Gomez, 2003, see Appendix B). This results in a difficult control problem with an 8-dimensional state space $\mathbb{S}$.

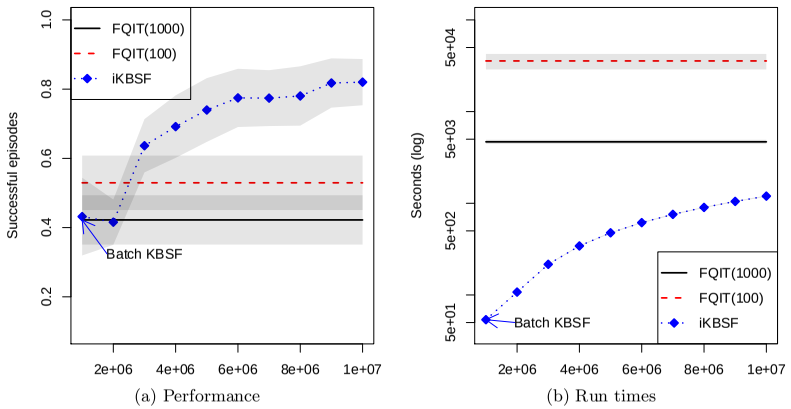
In our experiments with KBSF on the two-pole task we used 200 representative states and 10^6 sample transitions collected by a random policy. Here we start our experiment with triple pole-balancing using exactly the same configuration, and then we let *i*KBSF refine its model $\bar{M}$ by incorporating more sample transitions through update rules (19) and (20). We also let *i*KBSF grow its model if necessary. Specifically, a new representative state is added to $\bar{M}$ on-line every time the agent encounters a sample state $\hat{s}_i^a$ for which $\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_j) < 0.01$ for all $j \in 1, 2, ..., m$. This corresponds to setting a maximum allowed distance from a sampled state to the closest representative state, $\max_{a,i} dist(\hat{s}_i^a, 1)$.

Given the poor performance of LSPI on the double pole-balancing task, shown in Figures 4c and 4d, on the three-pole version of the problem we only compare KBSF with FQIT. We used FQIT with the same configuration adopted in Sections 4.2.3 and 4.2.4, with the parameter $\eta_{\min}$ varying in the set $\{10000, 1000, 100\}$. As for KBSF, the widths of the kernels were fixed at $\tau = 100$ and $\bar{\tau} = 1$ and sparse kernels were used ($\mu = 50$ and $\bar{\mu} = 10$).

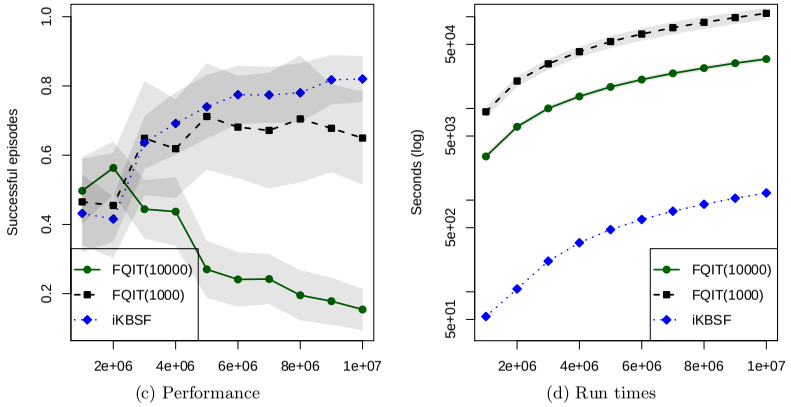
In order to show the benefits provided by the incremental version of our algorithm, we assumed that both KBSF and FQIT could store at most 10^6 sample transitions in memory. In the case of *i*KBSF, this is not a problem, since we can always split the data in subsets of smaller size and process them incrementally. Here, we used Algorithm 3 with a 0.3-greedy policy, $t_m = t_v = 10^6$, and $n = 10^7$. In the case of FQIT, we have two options to circumvent the limited amount of memory available. The first one is to use a single batch of 10^6 sample transitions. The other option is to use the initial batch of transitions to compute an approximation of the problem's value function, then use an 0.3-greedy policy induced by this approximation to collect a second batch, and so on. Here we show the performance of FQIT using both strategies.

We first compare the performance of *i*KBSF with that of FQIT using a single batch of sample transitions. This is shown in Figure 8a and 8b. For reference, we also show the results of batch KBSF—that is, we show the performance of the policy that would be computed by our algorithm if we did not have a way of computing its approximation incrementally. As shown in Figure 8a, both FQIT and batch KBSF perform poorly in the triple pole-balancing task, with average success rates below 55%. These results suggest that the amount of data used by these algorithms is insufficient to describe the dynamics of the control task. Of course, we could give more sample transitions to FQIT and batch KBSF. Note however that, since they are batch learning methods, there is an inherent limit on the amount of data that these algorithms can use to construct their approximation. In contrast, the amount of memory required by *i*KBSF is independent of the number of sample transitions n. This fact together with the fact that KBSF's computational complexity is only linear in n allow our algorithm to process a large amount of data in reasonable time. This can be clearly observed in Figure 8b, which shows that *i*KBSF can build an approximation using 10^7 sample transitions in under 20 minutes. As a reference for comparison, FQIT(1000) took an average of 1 hour and 18 minutes to process 10 times less data.

As shown in Figure 8a, *i*KBSF's ability to process a large number of sample transitions allows our algorithm to achieve a success rate of approximately 80%. This is similar to the



(a) Performance (b) Run times



(c) Performance (d) Run times

Figure 8: Results on the triple pole-balancing task, as a function of the number of sample transitions n, averaged over 50 runs. Figures 8a and 8b show results of FQIT when using a single batch of 10^6 transitions; in Figures 8c and 8d ten sets of 10^6 transitions were used in sequence by the algorithm (see text for details). The values correspond to the fraction of episodes initiated from the test states in which the 3 poles could be balanced for 3000 steps (one minute of simulated time). The test sets were regular grids of 256 cells defined over the hypercube centered at the origin and covering 50% of the state-space axes in each dimension (see Appendix B for details). Shadowed regions represent 99% confidence intervals.

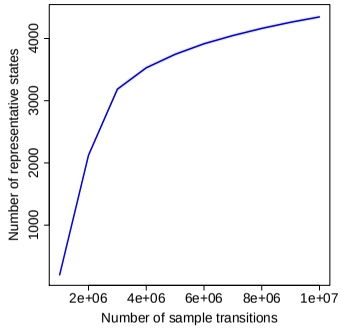


Figure 9: Number of representative states used by iKBSF on the triple pole-balancing task. Results were averaged over 50 runs (99% confidence intervals are almost imperceptible in the figure).

performance of batch KBSF on the two-pole version of the problem (*cf.* Figure 4). The good performance of iKBSF on the triple pole-balancing task is especially impressive when we recall that the decision policies were evaluated on a set of test states representing all possible directions of inclination of the three poles. In order to achieve the same level of performance with KBSF, approximately 2 Gb of memory would be necessary, even using sparse kernels, whereas iKBSF used less than 0.03 Gb of memory.

One may argue that the above comparison between FQIT and KBSF is not fair, since the latter used ten times the amount of data used by the former. Thus, in Figures 8c and 8d we show the results of FQIT using 10 batches of 10^6 transitions—exactly the same number of transitions processed by iKBSF. Here we cannot compare iKBSF with FQIT(100) because the computational cost of the tree-based approach is prohibitively large (it would take over 4 days only to train a single agent, not counting the test phase). When we look at the other instances of the algorithm, we see two opposite trends. Surprisingly, the extra sample transitions actually made the performance of FQIT(10000) *worse*. On the other hand, FQIT(1000) performs significantly better using more data, though still not as well as iKBSF (both in terms of performance and computing time).

To conclude, observe in Figure 9 how the number of representative states m grows as a function of the number of sample transitions processed by KBSF. As expected, in the beginning of the learning process m grows fast, reflecting the fact that some relevant regions of the state space have not been visited yet. As more and more data come in, the number of representative states starts to stabilize.

5.2.3 Helicopter hovering task (comparison with SARSA)

In the previous two sections we showed how iKBSF can be used to circumvent the inherent memory limitations of batch learning. We now show how our algorithm performs in a fully

on-line regime. For that, we focus on a challenging reinforcement learning task in which the goal is to control an autonomous helicopter.

Helicopters have unique control capabilities, such as low speed flight and in-place hovering, that make them indispensable instruments in many contexts. Such flexibility comes at a price, though: it is widely recognized that a helicopter is significantly harder to control than a fixed-wing aircraft (Ng et al., 2003; Abbeel et al., 2007). Part of this difficulty is due to the complex dynamics of the helicopter, which is not only non-linear, noisy, and asymmetric, but also counterintuitive in some aspects (Ng et al., 2003).

An additional complication of controlling an autonomous helicopter is the fact that a wrong action can easily lead to a crash, which is both dangerous and expensive. Thus, the usual practice is to first develop a model of the helicopter's dynamics and then use the model to design a controller (Ng et al., 2003). Here we use the model constructed by Abbeel et al. (2005) based on data collected on actual flights of an XCell Tempest helicopter (see Appendix B). The resulting reinforcement learning problem has a 12-dimensional state space whose variables represent the aircraft's position, orientation, and the corresponding velocities and angular velocities along each axis.

In the version of the task considered here the goal is to keep the helicopter hovering as close as possible to a fixed position. All episodes start at the target location, and at each time step the agent receives a negative reward proportional to the distance from the current state to the desired position. Because the tail rotor's thrust exerts a sideways force on the helicopter, the aircraft cannot be held stationary in the zero-cost state even in the absence of wind. The episode ends when the helicopter leaves the hover regime, that is, when any of the state's variables exceeds pre-specified thresholds.

The helicopter is controlled via a 4-dimensional continuous vector whose variables represent the longitudinal cyclic pitch, the latitudinal cyclic pitch, the tail rotor collective pitch, and the main rotor collective pitch. By adjusting the value of these variables the pilot can rotate the helicopter around its axes and control the thrust generated by the main rotor. Since KBSF was designed to deal with a finite number of actions, we discretized the set A using 4 values per dimension, resulting in 256 possible actions. The details of the discretization process are given below.

Here we compare iKBSF with the SARSA(λ) algorithm using tile coding for value function approximation (Rummery and Niranjan, 1994, Sutton, 1996—see Appendix B). We applied SARSA with $\lambda = 0.05$, a learning rate of 0.001, and 24 tilings containing 4^{12} tiles each. Except for λ, all the parameters were adjusted in a set of preliminary experiments in order to improve the performance of the SARSA agent. We also defined the action-space discretization based on SARSA's performance. In particular, instead of partitioning each dimension in equally-sized intervals, we spread the break points unevenly along each axis in order to maximize the return obtained by the SARSA agent. The result of this process is described in Appendix B. The interaction of the SARSA agent with the helicopter hovering task was dictated by an ϵ-greedy policy. Initially we set $\epsilon = 1$, and at every 50000 transitions the value of ϵ was decreased in 30%.

The iKBSF agent collected sample transitions using the same exploration regime. Based on the first batch of 50000 transitions, $m = 500$ representative states were determined by the k-means algorithm. No representative states were added to iKBSF's model after that.

Both the value function and the model were updated at fixed intervals of $t_v = t_m = 50000$ transitions. We fixed $\tau = \bar{\tau} = 1$ and $\mu = \bar{\mu} = 4$.

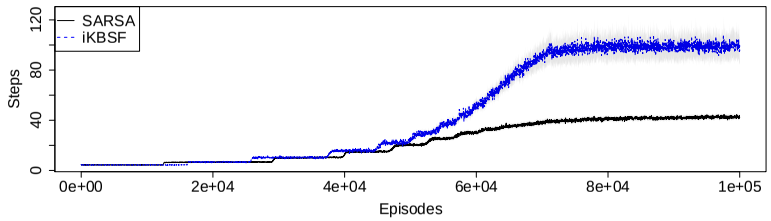
Figure 10 shows the results obtained by SARSA and KBSF on the helicopter hovering task. Note in Figure 10a how the average episode length increases abruptly at the points in which the value of ϵ is decreased. This is true for both SARSA and KBSF. Also, since the number of steps executed per episode increases over time, the interval in between such abrupt changes decreases in length, as expected. Finally, observe how the performance of both agents stabilizes after around 70000 episodes, probably because at this point there is almost no exploration taking place anymore.

When we compare KBSF and SARSA, it is clear that the former significantly outperforms the latter. Specifically, after the cut-point of 70000 episodes, the KBSF agent executes approximately 2.25 times the number of steps performed by the SARSA agent before crashing. Looking at Figures 10a and 10b, one may argue at first that there is nothing surprising here: being a model-based algorithm, KBSF is more sample efficient than SARSA, but it is also considerably slower (Atkeson and Santamaria, 1997). Notice though that the difference between the run times of SARSA and KBSF shown in Figure 10b is in part a consequence of the good performance of the latter: since KBSF is able to control the helicopter for a larger number of steps, the corresponding episodes will obviously take longer. A better measure of the algorithms's computational cost can be seen in Figure 10c, which shows the average time taken by each method to perform one transition. Observe how KBSF's computing time peaks at the points in which the model and the value function are updated. In the beginning KBSF's MDP changes considerably, and as a result the value function updates take longer. As more data come in, the model starts to stabilize, accelerating the computation of $\bar{\mathbf{Q}}^*$ (we "warm start" policy iteration with the value function computed in the previous round). At this point, KBSF's computational cost per step is only slightly higher than SARSA's, even though the former computes a model of the environment while the latter directly updates the value function approximation.

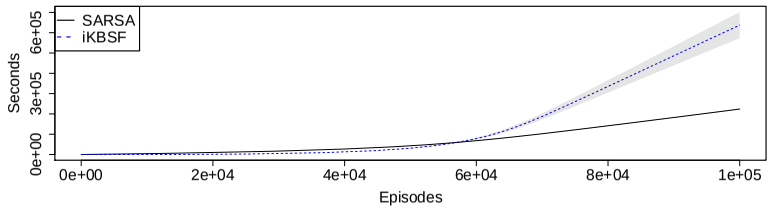
To conclude, we note that our objective in this section was exclusively to show that KBSF can outperform a well-known on-line algorithm with compatible computational cost. Therefore, we focused on the comparison of the algorithms rather than on obtaining the best possible performance on the task. Also, it is important to mention that more difficult versions of the helicopter task have been addressed in the literature, usually using domain knowledge in the configuration of the algorithms or to guide the collection of data (Ng et al., 2003; Abbeel et al., 2007). Since our focus here was on evaluating the on-line performance of KBSF, we addressed the problem in its purest form, without using any prior information to help the algorithms solve the task (see Asbah et al.'s paper for experiments on the helicopter hovering task with a more specialized version of KBSF, 2013).

6. Discussion

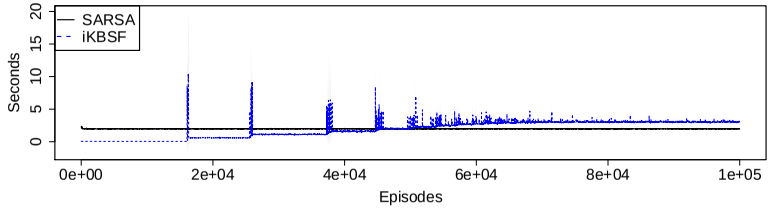
In this section we deepen our discussion about KBSF. We start with an analysis of the approximation computed by our algorithm, in which we draw interesting connections with KBRL, and then we proceed to discuss some issues that come up when one uses KBSF in practice.



(a) Performance



(b) Run time



(c) Average time per step (time of an episode divided by the number of steps)

Figure 10: Results on the helicopter hovering task averaged over 50 runs. The learned controllers were tested from a fixed state (see text for details). The shadowed regions represent 99% confidence intervals.

6.1 A closer look at KBSF's approximation

As outlined in Section 2, KBRL defines the probability of a transition from state $\hat{s}_i^b$ to state $\hat{s}_k^a$ as being $\kappa_\tau^a(\hat{s}_i^b, s_k^a)$, where $a, b \in A$ (see Figure 2a). Note that the kernel κ_τ^a is computed with the initial state s_k^a, and not $\hat{s}_k^a$ itself. The intuition behind this is simple: since we know the transition $s_k^a \xrightarrow{a} \hat{s}_k^a$ has occurred before, the more "similar" $\hat{s}_i^b$ is to s_k^a, the more likely the transition $\hat{s}_i^b \xrightarrow{a} \hat{s}_k^a$ becomes (Ormoneit and Sen, 2002).

From (10), it is clear that the computation of matrices $\mathbf{K}^a$ performed by KBSF follows the same reasoning underlying the computation of KBRL's matrices $\hat{\mathbf{P}}^a$; in particular, $\kappa_\tau^a(\bar{s}_j, s_k^a)$ gives the probability of a transition from $\bar{s}_j$ to $\hat{s}_k^a$. However, when we look at matrix $\mathbf{D}$ things are slightly different: here, the probability of a "transition" from $\hat{s}_i^b$ to representative state $\bar{s}_j$ is given by $\bar{\kappa}_{\bar{\tau}}(\hat{s}_i^b, \bar{s}_j)$—a computation that involves $\bar{s}_j$ itself. If we were to strictly adhere to KBRL's logic when computing the transition probabilities to the representative states $\bar{s}_j$, the probability of transitioning from $\hat{s}_i^b$ to $\bar{s}_j$ upon executing action a should be a function of $\hat{s}_i^b$ and a state s' from which we knew a transition $s' \xrightarrow{a} \bar{s}_j$ had occurred. In this case we would end up with one matrix $\mathbf{D}^a$ for each action $a \in A$. Note though that this formulation of the method is not practical, because the computation of the matrices $\mathbf{D}^a$ would require a transition $(\cdot) \xrightarrow{a} \bar{s}_j$ for each $a \in A$ and each $\bar{s}_j \in \bar{S}$. Clearly, such a requirement is hard to fulfill even if we have a generative model available to generate sample transitions.

In this section we take a closer look at KBSF's approximation, and provide two interpretations that support the way the matrices involved are built. We argue that KBSF can be seen as a kernel-based approximation of both the model and the value function computed by KBRL. Based on these interpretations we later discuss how KBSF can potentially be beneficial from a statistical point of view.

6.1.1 Approximating the model

We start by looking at how KBRL constructs its model. As shown in Figure 2a, for each action $a \in A$ the state $\hat{s}_i^b$ has an associated stochastic vector $\hat{\mathbf{p}}_j^a \in \mathbb{R}^{1\times n}$ whose nonzero entries correspond to the kernel $\kappa_\tau^a(\hat{s}_i^b, \cdot)$ evaluated at $s_k^a, k = 1, 2, \ldots, n_a$. Since we are dealing with a continuous state space, it is possible to compute an analogous vector for any $s \in \mathbb{S}$ and any $a \in A$. Focusing on the nonzero entries of $\hat{\mathbf{p}}_j^a$, we define the function

$$\begin{aligned} \hat{\mathcal{P}}_{S^a}: \quad & \mathbb{S} \mapsto \mathbb{R}^{1\times n_a} \\ & \hat{\mathcal{P}}_{S^a}(s) = \hat{\mathbf{p}}^a \iff \hat{p}_i^a = \kappa_\tau^a(s, s_i^a) \text{ for } i = 1, 2, ..., n_a. \end{aligned} \tag{23}$$

Clearly, full knowledge of the function $\hat{\mathcal{P}}_{S^a}$ allows for an exact computation of KBRL's transition matrix $\hat{\mathbf{P}}^a$. Now suppose we do not know $\hat{\mathcal{P}}_{S^a}$ and we want to compute an approximation of this function in the points $\hat{s}_i^a \in S^a$, for all $a \in A$. Suppose further that we are only given a "training set" composed of m pairs $(\bar{s}_j, \hat{\mathcal{P}}_{S^a}(\bar{s}_j))$. One possible way of approaching this problem is to resort to kernel smoothing techniques. In this case, a particularly common choice is the so-called Nadaraya-Watson kernel-weighted estimator (Hastie et al., 2002, Chapter 6):

$$\bar{\mathcal{P}}_{S^a}(s) = \frac{\sum_{j=1}^m \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_j)\hat{\mathcal{P}}_{S^a}(\bar{s}_j)}{\sum_{j=1}^m \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_j)} = \sum_{j=1}^m \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_j)\hat{\mathcal{P}}_{S^a}(\bar{s}_j). \tag{24}$$

Contrasting the expression above with (10), we see that this is exactly how KBSF computes its approximation $\mathbf{D}\mathbf{K}^a \approx \hat{\mathbf{P}}^a$, with $\bar{\mathcal{P}}_{S^a}$ evaluated at the points $\hat{s}_i^b \in S^b$, $b = 1, 2, ..., |A|$. In this case, $\bar{\kappa}_{\bar{\tau}}(\hat{s}_i^b, \bar{s}_j)$ are the elements of matrix $\mathbf{D}$, and $\hat{\mathcal{P}}_{S^a}(\bar{s}_j)$ is the j^{th} row of matrix $\dot{\mathbf{K}}^a$. Thus, in some sense, KBSF uses KBRL's own kernel approximation principle to compute a stochastic factorization of $\hat{M}$. One can easily extend the reasoning above to also include rewards by defining a function $\hat{\mathcal{M}}_{S^a} : \mathbb{S} \mapsto \mathbb{R}^{1 \times n_a + 1}$ such that $\hat{\mathcal{M}}_{S^a}(s) = [\hat{\mathcal{P}}_{S^a}(s), \hat{\mathcal{P}}_{S^a}(s)^\top \mathbf{r}^a]$.

The exposition above also makes it clear that it is possible to build the matrices $\dot{\mathbf{D}}^a$ using different sets of representative states $\bar{S}^a$ (this corresponds to having $|A|$ training sets composed of pairs $(\bar{s}_j^a, \hat{\mathcal{P}}_{S^a}(\bar{s}_j^a))$). As long as all the sets $\bar{S}^a$ have the same cardinality m, the stochastic-factorization trick can still be applied. However, in order for the approximation computed by KBSF to make sense, we would have to have one matrix $\mathbf{D}^a$ for each action $a \in A$ (see in Figure 2b how matrix $\mathbf{D}$ is computed). Unfortunately, when different matrices $\mathbf{D}^a$ are used, our "core" theoretical result, Proposition 2, no longer applies. Although alternative error bounds are possible, as shown by Barreto (2014), they are in general significantly looser than (8). In any case, the extension of KBSF to use different sets of representative states $\bar{S}^a$ may be an interesting topic for future research.

6.1.2 Approximating the value function

We now turn our attention to the way KBRL computes its value function, and show an intuitive way to derive KBSF's update equations. Based on (4) and (5), we see that value iteration's update rule for KBRL's MDP, $\hat{\Delta}\Gamma$, is

$$\hat{Q}_{t+1}(\hat{s}_i^c, a) = \sum_{j=1}^{n_a} \kappa_\tau^a(\hat{s}_i^c, s_j^a) \left[r_j^a + \gamma \max_b \hat{Q}_t(\hat{s}_j^a, b) \right], \tag{25}$$

where $a, b, c \in A$. Note that at any time step t the equation above can be used to compute an approximation of the t-step value function over the entire state space $\mathbb{S}$; after convergence to $\hat{Q}^*$ equation (25) gives rise to (7).

From a computational point of view, the potential problem with rule (25) is the fact that updating $\hat{Q}$ takes $O(n\hat{n}|A|)$ operations (recall that $n = \sum_a n_a$ and $\hat{n} = \max_a n_a$). KBRL makes it possible to compute an approximate solution for an MDP with continuous state space $\mathbb{S}$ by only updating the value of a finite subset $\hat{S} \subset \mathbb{S}$. It is reasonable to ask whether similar strategy can come to the rescue when $\hat{S}$ is itself too large. Specifically, if the values of the states $\hat{s}_i^a \in \hat{S}$ can be approximated based on the values of a small set of states $\bar{S} \subset \mathbb{S}$, with $|\bar{S}| = m$, then only the latter must be updated during dynamic programming's iterative process. Following this reasoning, we can replace $\hat{Q}_t(\hat{s}_j^a, b)$ with $\sum_{l=1}^m \bar{\kappa}_{\bar{\tau}}(\hat{s}_j^a, \bar{s}_l)\hat{Q}_t(\bar{s}_l, b)$ and rewrite (25) as

$$\begin{aligned}
\hat{Q}'_{t+1}(\bar{s}_i, a) &= \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) \left[r_j^a + \gamma \max_b \sum_{l=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_j^a, \bar{s}_l)\hat{Q}'_t(\bar{s}_l, b) \right] \\
&= \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) r_j^a + \gamma \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) \max_b \sum_{l=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_j^a, \bar{s}_l)\hat{Q}'_t(\bar{s}_l, b).
\end{aligned} \tag{26}$$

Updating $\hat{Q}'$ through (26) is $O(m\hat{n}|A|)$. This is already an improvement over the computational complexity of (25), but the dependence on $\hat{n}$ still precludes the use of (26) in

scenarios involving many sample transitions, such as on-line problems, for example. Now, if we *swap* the positions of the 'max' operator and the kernel $\bar{\kappa}_{\bar{\tau}}$ in (26), we can get rid of the dependence on the number of transitions, in the following way:

$$\begin{aligned}
\bar{Q}_{t+1}(\bar{s}_i, a) &= \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) r_j^a + \gamma \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) \sum_{l=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_j^a, \bar{s}_l) \max_b \bar{Q}_t(\bar{s}_l, b) \\
&= \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) r_j^a + \gamma \sum_{l=1}^{m} \sum_{j=1}^{n_a} \kappa_\tau^a(\bar{s}_i, s_j^a) \bar{\kappa}_{\bar{\tau}}(\hat{s}_j^a, \bar{s}_l) \max_b \bar{Q}_t(\bar{s}_l, b) \\
&= \bar{r}_i^a + \gamma \sum_{l=1}^{m} \bar{p}_{il}^a \max_b \bar{Q}_t(\bar{s}_l, b).
\end{aligned} \tag{27}$$

Updating $\bar{Q}$ through (27) takes only $O(m^2|A|)$ operations. It is not difficult to see that (27) is value iteration's update rule for the MDP $\bar{M}$ defined by KBSF, $\bar{\Delta}\Gamma$ (see Algorithm 1).

Summarizing, if we use a local kernel approximation in the update rule of KBRL's MDP, potentially introducing some error, we get (26), which is faster to compute than (25). If in addition we change the order in which the operations are applied, we get KBSF's update rule (27), which is in some sense a greater deviation from (25) than (26), but is even faster to compute. Therefore, KBSF can be interpreted as a computationally efficient way of applying kernel approximation, the basic principle behind KBRL, to KBRL itself.

6.1.3 Curse of dimensionality

In this paper we emphasized the role of KBSF as a technique to reduce KBRL's computational cost. However, it is equally important to ask whether our algorithm provides benefits from a statistical point of view. In particular, instead of trying to approximate KBRL's solution, it may be possible to compute *better* solutions using the same amount of data.

Ormoneit and Sen (2002) showed that, in general, the number of sample transitions needed by KBRL to achieve a certain approximation accuracy grows exponentially with the dimension of the state space—a phenomenon usually referred to as the "curse of dimensionality" (Bellman, 1961). As with other methods, the only way to avoid such an exponential dependency is to exploit some sort of regularity in the problem's structure—paraphrasing Ormoneit and Sen (2002), one can only "break" the curse of dimensionality by incorporating prior knowledge into the approximation. In this section we argue that KBSF can be seen as a strategy to do so. In particular, the definition of the representative states can be seen as a practical mechanism to incorporate additional assumptions about the continuous MDP besides the sample transitions.

The fact that KBRL's sample complexity is exponential in $d_{\mathbb{S}}$ is not surprising. As noted by Ernst et al. (2005), KBRL can be seen as a particular case of the fitted Q-iteration algorithm, which solves a reinforcement learning problem by breaking it into a succession of supervised learning problems. If the MDP defined by KBRL is solved through value iteration, for example, each iteration of this algorithm can be seen as a non-parametric kernel-based regression (Barreto, 2014). It is well known that non-parametric kernel methods suffer from the curse of dimensionality (Stone, 1982; Györfi et al., 2002).

As mentioned above, one can circumvent the curse of dimensionality associated with non-parametric methods by exploiting regularities of the learning problem (Györfi et al.,

2002). For example, Kveton and Theocharous (2013) showed that, when the dynamics of the underlying MDP are factored, incorporating this knowledge into KBRL leads to a dramatic reduction on the number of sample transitions needed for learning a good decision policy. Another type of regularity arises when the "intrinsic dimension" of the state space is much smaller than $d_{\mathbb{S}}$—that is, when the data lies close to a low-dimensional manifold. Farahmand et al. (2007) have shown that simple k-nearest neighborhood regression is manifold-adaptive, meaning that it naturally exploits this type of regularity in the approximation problem. Since KBRL's approximation scheme is very similar to k-nearest neighborhood regression, it is likely that this algorithm also has such a property.

Another way to avoid the curse of dimensionality is to resort to parametric methods. Parametric methods circumvent the exponential growth of the number of samples by restricting the space of functions spanned by the approximator—which corresponds to making assumptions about the target function (Györfi et al., 2002). Since in KBRL the structure of the approximator is defined by the sampled states $\hat{s}_i^a$, one has little flexibility in controlling the induced function space (see (25)). Based on the discussions in Sections 6.1.1 and 6.1.2, one can see that KBSF can be cast as a kernel-based approximation in which the structure of the approximator is defined by the representative states—that is, the representative states play the role of basis functions or "features"(this architecture is similar to that of radial basis function networks, see Chapter 7 of Györfi et al.'s book, 2002). It should be clear, then, that when the representative states are fixed KBSF can be seen as a parametric model—and thus as a potential tool for helping KBRL circumvent the curse of dimensionality.

Parametric regression methods have a serious drawback, though: if the target function cannot be well approximated by any function in the space induced by the parametric model, the approximation error will be large regardless of the data. A popular strategy to overcome this issue without incurring in unacceptable sample complexity is to resort to adaptive methods that use the available data to adjust the complexity of the approximator to the problem at hand (Farahmand and Szepesvári, 2011). In the case of KBRL, the complexity of the induced function space can be controlled via the kernel's width τ (Györfi et al., 2002). KBSF provides an alternative way of controlling the complexity of the approximator through the parameter m (this is akin to limiting the depth of a regression tree, for example—see Barreto's paper for a detailed discussion, 2014). If we think in terms of the classical bias-variance analysis of statistical estimators, intuitively we are decreasing bias and increasing variance as $m \to n$—assuming that the representative states are systematically defined by a reasonable method; see Section 6.2.2 (Hastie et al., 2002). Note that, unlike τ, m has a clear effect on the method's computational cost.

Therefore, KBSF provides both a way of determining the structure of the approximator and extra flexibility in controlling the complexity of the resulting model. Of course, this does not automatically translate into reduced sample complexity. Even if the structural assumptions induced by KBSF arise in problems of interest, we must develop methods to configure the parameters of the algorithm appropriately (see Section 6.2.2). New theoretical results regarding the approximation properties of KBSF would probably be needed as well, since the current results were derived under the interpretation of our algorithm as a computational device to accelerate KBRL (and thus use its solution as a reference point). These might be interesting directions for future investigations.

6.2 Practical issues

During the execution of our experiments we observed several interesting facts about KBSF which are not immediate from its conceptual definition. In this section we share some of the lessons learned with the reader. We start by discussing the impact of deviating from the theoretical assumptions over the performance of our algorithm. We then present general guidelines on how to configure KBSF to solve reinforcement learning problems.

6.2.1 KBSF's applicability

The theoretical guarantees regarding KBRL's solution assume that the initial states s_i^a in the transitions $(s_i^a, r_i^a, \hat{s}_i^a)$ are uniformly sampled from $\mathbb{S}$ (Ormoneit and Sen, 2002, see Assumption 3). This is somewhat restrictive because it precludes the collection of data through direct interaction with the environment. Ormoneit and Sen conjectured that sampling the states s_i^a from an uniform distribution is not strictly necessary, and indeed later Ormoneit and Glynn (2002) relaxed this assumption for the case in which KBRL is applied to an average-reward MDP. In this case, it is only required that the exploration policy used to collect data chooses all actions with positive probability. As described in Sections 4.2 and 5.2, in our computational experiments we collected data through an ϵ-greedy policy (in many cases with $\epsilon = 1$). The good performance of KBSF corroborates Ormoneit and Sen's conjecture and suggests that Ormoneit and Glynn's results can be generalized to the discounted reward case, but more theoretical analysis is needed.

Ormoneit and Sen (2002) also make some assumptions regarding the smoothness of the reward function and the transition kernel of the continuous MDP (Assumptions 1 and 2). Unfortunately, such assumptions are usually not verifiable in practice. Empirically, we observed that KBSF indeed performs better in problems with "smooth dynamics"—loosely speaking, problems in which a small perturbation in s_i^a results in a small perturbation in $\hat{s}_i^a$, such as the pole balancing task. In problems with "rougher" dynamics, like the epilepsy-suppression task, it is still possible to get good results with KBSF, but in this case it is necessary to use more representative states and narrower kernels (that is, smaller values for $\bar{\tau}$). As a result, in problems of this type KBSF is less effective in reducing KBRL's computational cost.

6.2.2 KBSF's configuration

KBSF depends on two basic definitions: the set of representative states and the widths of the kernels. As discussed in Section 6.1, the former can be seen as the definition of the "structure" of the approximator, while the latter are parameters of the approximation. In what follows we discuss the impact of each of them on KBSF's performance.

Definition of Representative States In order to define the representative states we must determine their number, m, and their "position" in the state space. Both theory and practice indicate that KBSF's performance gets closer to KBRL's as m increases. Thus, a "rule of thumb" to define the number of representative states is to simply set m to the largest value allowed by the available computational resources (but see Section 6.1.3 for an alternative view).

As for the position of the representative states, looking at expression (24) we see that ideally $\bar{s}_j$ would be such that the rows of the matrices $\mathbf{K}^a$ would form a convex hull containing the rows of the corresponding $\hat{\mathbf{P}}^a$. However, it is easy to see that when $m < n$ such a set of states may not exist. Besides, even when it does exist, finding this set is not a trivial problem.

Instead of insisting on finding representative states that allow for an exact representation of the matrices $\hat{\mathbf{P}}^a$, it sounds more realistic to content oneself with an approximate solution for this problem. Assuming the dynamics of the underlying MDP are reasonably smooth, one strategy is to guarantee that every sampled state $\hat{s}_i^a$ is close to at least one representative state $\bar{s}_j$. Since covering the entire state space is generally impractical, we want $\bar{s}_j$ to reflect the distribution of the data, that is, we want the representative states to be in the regions of the state space where the data lies. This is why in our experiments we clustered the states $\hat{s}_i^a$ and used the clusters's centers as our representative states (this method is also used in the configuration of radial basis function networks, as discussed by Györfi et al., 2002). Despite its simplicity, this strategy usually results in good performance, as shown in Sections 4.2 and 5.2.

In our experiments we clustered the data using the popular k-means algorithm, which minimizes the average square distance from a given point to the center of the corresponding cluster. However, Proposition 6 states that KBSF's approximation error can be controlled by $\max_{a,i} dist(\hat{s}_i^a, w)$, the maximum distance from a sampled state $\hat{s}_i^a$ to the w^{th} nearest representative state. It is reasonable to ask whether KBSF's performance would improve if we used a clustering method that directly minimizes this quantity.

In the k-center clustering problem one seeks a set of k clusters that minimize the maximum distance from a point to the corresponding cluster center (Gonzales, 1985). Thus, in this case we would be minimizing $\max_{a,i} dist(\hat{s}_i^a, 1)$. Although finding an exact solution for this problem is NP hard, there exist fast algorithms that compute good approximations in time linear in the number of points (Feder and Greene, 1988). In order to provide some intuition on how k-center clustering compares to k-means, we implemented an algorithm by Gonzales (1985) which is extremely simple: at each iteration one selects a new cluster center by simply picking the point whose distance to the closest cluster center is maximal. Although simple, this method provides a solution for the k-center problem whose cost is within a factor of two from the optimal solution, which is the best one can hope for in polynomial time (Feder and Greene, 1988).

Figure 11 shows the performance of KBSF on the puddle-world task using different strategies to define the representative states. The experiment was carried out exactly as the one described in Section 4.2.1. In addition to the results provided by k-means and k-centers, we also show KBSF's performance when using representative states sampled uniformly at random and evenly distributed over the state space. As expected, the worst results correspond to the case in which representative states are picked at random. On the opposite extreme, representative states evenly distributed over $\mathbb{S}$ result in the best performance. Unfortunately, in more realistic scenarios it is impractical to cover the entire state space with representative states, and this is precisely why one may have to resort to strategies like k-means and k-centers. The results shown in Figure 11 suggest that k-centers clustering may indeed be a better choice than k-means, although the difference in the resulting performance is not very significant. One advantage of k-center clustering is that

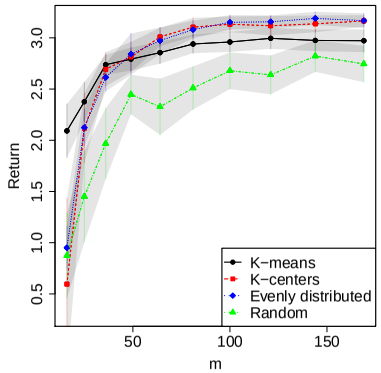
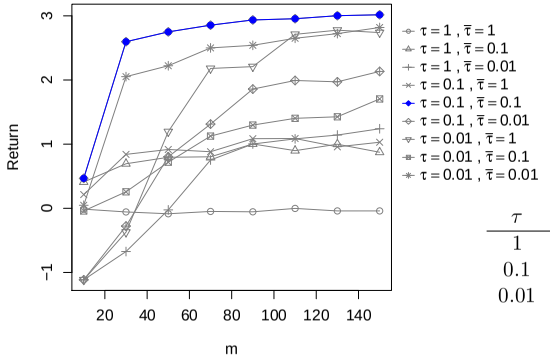


Figure 11: Comparison of different strategies to define the representative states. Results on the puddle-world task averaged over 50 runs. The errors around the mean correspond to the 99% confidence intervals. See Figure 3 for details.

it is very simple to implement and is also very fast. Besides, there are algorithms available that compute an approximation on-line, which allows their use with *i*KBSF (Charikar et al., 1997; Beygelzimer et al., 2006).

Clustering methods are a natural choice for defining the representative states because they summarize the distribution of the data, but of course they are not the only way to solve the problem. Ideally, we want a set of representative states that summarize the *dynamics* of the underlying dynamical system, and this may not reflect the spatial distribution of the data, regardless of the strategy used to collect transitions. As discussed in Section 6.1, the problem of defining representative states is in some sense akin to the problem of defining features in supervised learning (Guyon and Elisseeff, 2003). Such a connection emphasizes the difficulty of the problem and suggests that the best solution may be domain-dependent. On the other hand, as with the definition of features, the definition of representative states can be seen as an opportunity to incorporate prior knowledge about the domain of interest into the approximation model. For example, if one knows that some regions of the state space are more important than others, this information can be used to allocate more representative states to those regions. Similar reasoning applies to tasks in which the level of accuracy required from the decision policy varies across the state space.

Definition of Kernels's Widths Given a well-defined strategy to select representative states, the use of KBSF requires the definition of two parameters, τ and $\bar{\tau}$. The kernels's widths τ and $\bar{\tau}$ may have a strong effect on KBSF's performance. To illustrate this point, we show in Figure 12 the results of this algorithm on the puddle world task when τ and $\bar{\tau}$ are varied in the set $\{0.01, 0.1, 1\}$ (these were the results used to generate Figure 3).



(a) Performance of KBSF(8000, ·)

τ	Average return
1	1.47 ± 0.42
0.1	3.01 ± 0.08
0.01	3.00 ± 0.08

(b) Performance of KBRL(8000). Errors around the mean correspond to 99% confidence intervals.

Figure 12: The impact of the kernels's widths on the performance of KBSF and KBRL. Results on the puddle-world task averaged over 50 runs. See Figure 3 for details.

Of course, the best combination of values for τ and $\bar{\tau}$ depends on the specific problem at hand and on the particular choice of kernels. Here we give some general advice as to how to set these parameters, based on both theory in practice. Since τ is the same parameter used by KBRL, it should decrease with the number of sample transitions n at an "admissible rate" (see Ormoneit and Sen's Lemma 2, 2002). Analogously, Proposition 6 suggests that $\bar{\tau}$ should get smaller as $m \to n$ (details in the proof of the proposition in Appendix A). Empirically, we found out that a simple strategy that usually facilitates the configuration of KBSF is to rescale the data so that all the variables have approximately the same magnitude—which corresponds to using a weighted norm in the computation of the kernels. Using this strategy we were able to obtain good results with KBSF on all problems by performing a coarse search in the space of parameters in which we only varied the order of magnitude of τ and $\bar{\tau}$ (see Table 1).

Alternatively, one can fix τ and $\bar{\tau}$ and define the neighborhood used to compute $\mathrm{k}_\tau(\bar{s}_j, \cdot)$ and $\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \cdot)$. As explained in Appendix B.2, in some of our experiments we only computed $\mathrm{k}_\tau(\bar{s}_j, \cdot)$ for the μ closest sampled states s_i^a from $\bar{s}_j$, and only computed $\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \cdot)$ for the $\bar{\mu}$ closest representative states from $\hat{s}_i^a$. When using this approach, a possible way of configuring KBSF is to set τ and $\bar{\tau}$ to sufficiently large values (so as to guarantee a minimum level of overlap between the kernels) and then adjust μ and $\bar{\mu}$. The advantage is that adjusting μ and $\bar{\mu}$ may be more intuitive than directly configuring τ and $\bar{\tau}$ (*cf.* Table 1).

7. Previous work

In our experiments we compared KBSF with KBRL, LSPI, fitted Q-iteration, and SARSA, both in terms of computational cost and in terms of the quality of the resulting decision policies. In this section we situate our algorithm in the broader context of approximate reinforcement learning. Approximation in reinforcement learning is an important topic that has generated a huge body of literature. For a broad overview of the subject, we refer the reader to the books by Sutton and Barto (1998), Bertsekas and Tsitsiklis (1996), and Szepesvári (2010). Here we will narrow our attention to *kernel-based* approximation techniques.

We start by noting that the label "kernel based" is used with two different meanings in the literature. On one side we have kernel smoothing techniques like KBRL and KBSF, which use local kernels essentially as a device to implement smooth instance-based approximation (Hastie et al., 2002). On the other side we have methods that use reproducing kernels to implicitly represent an inner product in a high-dimensional state space (Schölkopf and Smola, 2002). Although these two frameworks can give rise to approximators with similar structures, they rest on different theoretical foundations. Since reproducing-kernels methods are less directly related to KBSF, we will only describe them briefly. We will then discuss the kernel smoothing approaches in more detail.

The basic idea of reproducing-kernel methods is to apply the "kernel trick" in the context of reinforcement learning (Schölkopf and Smola, 2002). Roughly speaking, the approximation problem is rewritten in terms of inner products only, which are then replaced by a properly-defined kernel. This modification corresponds to mapping the problem to a high-dimensional feature space, resulting in more expressiveness of the function approximator. Perhaps the most natural way of applying the kernel trick in the context of reinforcement learning is to "kernelize" some formulation of the value-function approximation problem (Xu et al., 2005; Engel et al., 2005; Farahmand, 2011). Another alternative is to approximate the *dynamics* of an MDP using a kernel-based regression method (Rasmussen and Kuss, 2004; Taylor and Parr, 2009). Following a slightly different line of work, Bhat et al. (2012) propose to kernelize the linear programming formulation of dynamic programming. However, this method is not directly applicable to reinforcement learning, since it is based on the assumption that one has full knowledge of the MDP. A weaker assumption is to suppose that only the reward function is known and focus on the approximation of the transition function. This is the approach taken by Grunewalder et al. (2012), who propose to embed the conditional distributions defining the transitions of an MDP into a Hilbert space induced by a reproducing kernel.

We now turn our attention to kernel-smoothing techniques, which are more closely related to KBRL and KBSF. Kroemer and Peters (2011) propose to apply kernel density estimation to the problem of policy evaluation. They call their method *non-parametric dynamic programming* (NPDP). If we use KBRL to compute the value function of a fixed policy, we see many similarities with NPDP, but also some important differences. Like KBRL, NPDP is statistically consistent. Unlike KBRL, which assumes a finite action space A and directly approximates the conditional density functions $P^a(s'|s)$, NPDP assumes that A is continuous and models the joint density $P(s, a, s')$. Kroemer and Peters (2011) showed that the value function of NPDP has a Nadaraya-Watson kernel regression form. Not

surprisingly, this is also the form of KBRL's solution if we fix the policy being evaluated (*cf.* equation (7)). In both cases, the coefficients of the kernel-based approximation are derived from the value function of the approximate MDP. The key difference is the way the transition matrices are computed in each algorithm. As shown in (4), the transition probabilities of KBRL's model are given by the kernel values themselves. In contrast, the computation of each element of NDPD's transition matrix requires an integration over the continuous state space $\mathbb{S}$. In practice, this is done by numerical integration techniques that may be very computationally demanding (see for example the experiments performed by Grunewalder et al., 2012).

We directly compared NPDP with KBRL because both algorithms build a model whose number of states is dictated by the number of sample transitions n, and neither method explicitly attempts to keep n small. Since in this case each application of the Bellman operator is $O(n^2)$, these methods are not suitable for problems in which a large number of transitions is required, nor are they applicable to on-line reinforcement learning.[5] There are however kernel-smoothing methods that try to avoid this computational issue by either keeping n small or by executing a number of operations that grows only linearly with n. These algorithms are directly comparable with KBSF.

One of the first attempts to adapt KBRL to the on-line scenario was that of Jong and Stone (2006). Instead of collecting a batch of sample transitions before the learning process starts, the authors propose to grow such a set incrementally, based on an exploration policy derived from KBRL's current model. To avoid running a dynamic-programming algorithm to completion in between two transitions, which may not be computationally feasible, Jong and Stone (2006) resort to Moore and Atkeson's (1993) "prioritized sweeping" method to propagate the changes in the value function every time the model is modified. The idea of exploiting the interpretation of KBRL as the derivation of a finite MDP in order to use tabular exploration methods is insightful. However, it is not clear whether smart exploration is sufficient to overcome the computational difficulties arising from the fact that the size of the underlying model is inexorably linked to the number of sample transitions. For example, even using sparse kernels in their experiments, Jong and Stone (2006) had to fix an upper limit for the size of KBRL's model. In this case, once the number of sample transitions has reached the upper limit, all subsequent data must be ignored.

Following the same line of work, Jong and Stone (2009) later proposed to guide KBRL's exploration of the state space using Brafman and Tennenholtz's (2003) R-MAX algorithm. In this new paper the authors address the issue with KBRL's scalability more aggressively. First, they show how to combine their approach with Dietterich's (2000) MAX-Q algorithm, allowing the decomposition of KBRL's MDP into a hierarchy of simpler models. While this can potentially reduce the computational burden of finding a policy, such a strategy transfer to the user the responsibility of identifying a useful decomposition of the task. A more practical approach is to combine KBRL with some stable form of value-function approximation. For that, Jong and Stone (2009) suggest the use of Gordon's (1995) averagers. As shown in Appendix A.2, this setting corresponds to a particular case of KBSF in which representative states are selected among the set of sampled states $\hat{s}_i^a$. It should be noted that, even when using temporal abstraction and function approximation, Jong and Stone's (2009) approach

5. We note that, incidentally, all the reproducing-kernel methods discussed in this section also have a computational complexity super-linear in n.

requires recomputing KBRL's transition probabilities at each new sample, which can be infeasible in reasonably large problems.

Kveton and Theocharous (2012) propose a more practical algorithm to reduce KBRL's computational cost. Their method closely resembles the batch version of KBSF. As with our algorithm, Kveton and Theocharous's method defines a set of representative states $\bar{s}_i$ that give rise to a reduced MDP. The main difference in the construction of the models is that, instead of computing a similarity measure between each sampled state $\hat{s}_i^a$ and all representative states $\bar{s}_j$, their algorithm associates each $\hat{s}_i^a$ with a single $\bar{s}_j$—which comes down to computing a hard aggregation of the state space $\hat{S}$. Such an aggregation corresponds to having a matrix $\mathbf{D}$ with a single nonzero element per row. In fact, it is possible to rewrite Kveton and Theocharous's (2012) algorithm using KBSF's formalism. In this case, the elements of $\dot{\mathbf{D}}^a$ and $\dot{\mathbf{K}}^a$ would be defined as:

$$\dot{k}_{ij}^a = \kappa_\tau^a(\bar{s}_i, rs(s_j^a, 1)), \quad \text{and} \quad \dot{d}_{ij}^a = \bar{\kappa}_0(rs(\hat{s}_i^a, 1), \bar{s}_j) \tag{28}$$

where $\bar{\kappa}_0$ is the normalized kernel induced by an infinitely "narrow kernel $\bar{\mathrm{k}}_0(s, s')$ whose value is greater than zero if and only if $s = s'$ (recall from Section 4.1 that $rs(s, 1)$ gives the closest representative state from s). It is easy to see that we can make matrix $\mathbf{D}$ computed by KBSF as close as desired to a hard aggregation by setting $\bar{\tau}$ to a sufficiently small value (see Lemma 5). More practically, we can simply plug (28) in place of (10) in Algorithm 1 to exactly recover Kveton and Theocharous's method. Note though that, by replacing $\kappa_\tau^a(\bar{s}_i, s_j^a)$ with $\kappa_\tau^a(\bar{s}_i, rs(s_j^a, 1))$ in the computation of $\dot{\mathbf{K}}^a$, we would be in some sense deviating from KBRL's framework. To see why this is so, observe that if the representative states $\bar{s}_i$ are sampled from the set of states $\hat{s}_i^a$, the rows of matrix $\mathbf{K}^a$ computed by KBSF would coincide with a subset of the rows of the corresponding KBRL's matrix $\hat{\mathbf{P}}^a$ (*cf.* (23)). However, this property is lost if one uses (28) instead of (10).[6]

8. Conclusion

This paper presented KBSF, a reinforcement learning algorithm that results from the application of the stochastic-factorization trick to KBRL. KBSF summarizes the information contained in KBRL's MDP in a model of fixed size. By doing so, our algorithm decouples the structure of the model from its configuration. This makes it possible to build an approximation which accounts for both the difficulty of the problem *and* the computational resources available.

One of the main strengths of KBSF is its simplicity. As shown in the paper, its uncomplicated mechanics can be unfolded into two update rules that allow for a fully incremental version of the algorithm. This makes the amount of memory used by KBSF independent of the number of sample transitions. Therefore, with a few lines of code one has a reinforcement-learning algorithm that can be applied to large-scale problems, in both off-line and on-line regimes.

KBSF is also a sound method from a theoretical point of view. As discussed, the distance between the value function computed by this algorithm and the one computed by KBRL

6. Note that this observation only means that KBSF is closer to KBRL in this strict sense; in particular, it does not imply that Kveton and Theocharous's algorithm is not a principled method, nor does it mean that it will perform worse than KBSF.

is bounded by two factors: the quality and the level of stochasticity of the underlying stochastic factorization. We showed that both factors can be made arbitrarily small, which implies that, in theory, we can make KBSF's solution as close to KBRL's solution as desired.

Theoretical guarantees do not always translate into good performance in practice, though, either because they are built upon unrealistic assumptions or because they do not account for procedural difficulties that arise in practical situations. To ensure that this is not the case with our algorithm, we presented an extensive empirical study in which KBSF was successfully applied to different problems, some of them quite challenging. We also presented general guidelines on how to configure KBSF to solve a reinforcement learning problem.

For all the reasons listed above, we believe that KBSF has the potential of becoming a valuable resource for the solution of reinforcement learning problems. This is not to say that the subject has been exhausted. There are several possibilities for future research, some of which we now briefly discuss.

From an algorithmic point of view, perhaps the most pressing demand is for more principled methods to define the representative states. Incidentally, this also opens up the possibility of an automated procedure to set the kernel's widths $\bar{\tau}$ based solely on data. Taking the idea one step further, one can think of having one distinct $\bar{\tau}_i$ associated with each kernel $\bar{\kappa}_{\bar{\tau}}(\cdot, \bar{s}_i)$ (which would make the similarity between KBSF's approximation and radial basis function networks even stronger). Another important advance would be to endow *i*KBSF with more elaborate exploration strategies, maybe following the line of research initiated by Jong and Stone (2006, 2009).

From a theoretical perspective, it may be desirable to "detach" KBSF from KBRL and analyze the former on its own. In this paper we emphasized the view of our algorithm as a tool to apply KBRL in practice. This view is clearly reflected in our theoretical results, which show how KBSF's solution deviates from KBRL's and how we can control such a deviation. But KBRL is itself an approximation, so perhaps it makes sense to analyze KBSF independently from its precursor. This possibility is particularly appealing when we look at KBSF as an intermediate approach between fully nonparametric and parametric models, as discussed in Section 6.1.3.

Looking at KBSF against a broader context, a subject that deserves further investigation is the possibility of building an approximation based on multiple models. Model averaging is not inherently linked to KBSF, and in principle it can be used with virtually any reinforcement learning algorithm. However, KBSF's low computational cost makes it particularly amenable to this technique. Since our algorithm is significantly faster than any method whose complexity per iteration is a function of the number of sample transitions, we can afford to compute several approximations and still have a solution in comparable time (see Section 4.2.3 and Barreto's paper, 2014). Understanding how we can randomize the construction of the individual models and to what extend this can improve the quality of the resulting decision policy is a matter of interest.

We conclude by noting that KBSF represents one particular way in which the stochastic-factorization trick can be exploited in the context of reinforcement learning. In principle, any algorithm that builds a model based on sample transitions can resort to the same trick to leverage the use of the data. The basic idea remains the same: instead of estimating the transition probabilities between every pair of states, one focuses on a small set of representative states whose values are propagated throughout the state space based on some

notion of similarity. We believe that this general framework can potentially be materialized into a multitude of useful reinforcement learning algorithms.

Acknowledgments

The authors would like to thank Amir-massoud Farahmand and Yuri Grinberg for interesting discussions about approximation in reinforcement learning and related subjects. We also thank Branislav Kveton for insightful comments regarding the theory underlying kernel-based reinforcement learning, which were essential for the derivation of our Proposition 8. We are grateful to Keith Bush for making the epilepsy simulator available, and to Alicia Bendz and Ryan Primeau for helping in some of the computational experiments. Finally, we thank the anonymous reviewers for their valid comments and helpful suggestions to improve the paper. Funding for this research was provided by the NSERC Discovery Grant program.

Appendix A. Theoretical Results

A.1 Proofs

Proposition 2

Proof Let $\check{M} \equiv (S, A, \check{\mathbf{P}}^a, \check{\mathbf{r}}^a, \gamma)$, with $\check{\mathbf{P}}^a = \mathbf{D}\mathbf{K}^a$ and $\check{\mathbf{r}}^a = \mathbf{D}\bar{\mathbf{r}}^a$. From the triangle inequality, we know that

$$\left\|\mathbf{v}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\right\|_\infty \leq \left\|\mathbf{v}^* - \check{\mathbf{v}}^*\right\|_\infty + \left\|\check{\mathbf{v}}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\right\|_\infty, \tag{29}$$

where $\check{\mathbf{v}}^*$ is the optimal value function of $\check{M}$. Our strategy will be to bound $\|\mathbf{v}^* - \check{\mathbf{v}}^*\|_\infty$ and $\left\|\check{\mathbf{v}}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\right\|_\infty$. In order to find an upper bound for $\|\mathbf{v}^* - \check{\mathbf{v}}^*\|_\infty$, we apply Whitt's (1978) Theorem 3.1 and Corollary (b) of his Theorem 6.1, with all mappings between M and $\check{M}$ taken to be identities, to obtain

$$\|\mathbf{v}^* - \check{\mathbf{v}}^*\|_\infty \leq \frac{1}{1-\gamma}\left(\max_a \|\mathbf{r}^a - \mathbf{D}\bar{\mathbf{r}}^a\|_\infty + \frac{\gamma\bar{R}_{\text{dif}}}{2(1-\gamma)}\max_a \|\mathbf{P}^a - \mathbf{D}\mathbf{K}^a\|_\infty\right), \tag{30}$$

where we used the fact that $\max_{a,i} \check{r}_i^a - \min_{a,i} \check{r}_i^a \leq \bar{R}_{\text{dif}}$. It remains to bound $\left\|\check{\mathbf{v}}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\right\|_\infty$. Since $\check{\mathbf{r}}^a = \mathbf{D}\bar{\mathbf{r}}^a$ and $\mathbf{D}\bar{\mathbf{P}}^a = \mathbf{D}\mathbf{K}^a\mathbf{D} = \check{\mathbf{P}}^a\mathbf{D}$ for all $a \in A$, the stochastic matrix $\mathbf{D}$ satisfies Sorg and Singh's (2009) definition of a *soft homomorphism* between $\check{M}$ and $\bar{M}$ (see equations (25)–(28) in their paper). Applying Theorem 1 by the same authors, we know that

$$\left\|\Gamma(\check{\mathbf{Q}}^* - \mathbf{D}\bar{\mathbf{Q}}^*)\right\|_\infty \leq \frac{1}{1-\gamma}\sup_{i,t}(1 - \max_j d_{ij})\ \bar{\delta}_i^{(t)}, \tag{31}$$

where $\bar{\delta}_i^{(t)} = \max_{j:d_{ij}>0,k} \bar{q}_{jk}^{(t)} - \min_{j:d_{ij}>0,k} \bar{q}_{jk}^{(t)}$ and $\bar{q}_{jk}^{(t)}$ are elements of $\bar{\mathbf{Q}}^{(t)}$, the optimal t-step action-value function of $\bar{M}$. Since $\|\Gamma\check{\mathbf{Q}}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\|_\infty \leq \|\Gamma(\check{\mathbf{Q}}^* - \mathbf{D}\bar{\mathbf{Q}}^*)\|_\infty$ and, for all

$t > 0$, $\bar{\delta}_i^{(t)} \leq (1-\gamma)^{-1}(\max_{a,k} \bar{r}_k^a - \min_{a,k} \bar{r}_k^a)$, we can write

$$\left\|\check{\mathbf{v}}^* - \Gamma \mathbf{D}\bar{\mathbf{Q}}^*\right\|_\infty \leq \frac{\bar{R}_{\text{dif}}}{(1-\gamma)^2}\max_i (1 - \max_j d_{ij}) = \frac{\bar{R}_{\text{dif}}}{(1-\gamma)^2}\sigma(\mathbf{D}). \tag{32}$$

Substituting (30) and (32) back into (29), we obtain (8). ■

Lemma 4
Proof Define the function

$$\psi_{\tau,s}^{a,i}(s') = \left| \frac{\mathrm{k}_\tau(s, s_i^a)}{\sum_{j=1}^{n_a} \mathrm{k}_\tau(s, s_j^a)} - \frac{\mathrm{k}_\tau(s', s_i^a)}{\sum_{j=1}^{n_a} \mathrm{k}_\tau(s', s_j^a)} \right| = \left| \frac{\phi\left(\|s - s_i^a\|/\tau\right)}{\sum_{j=1}^{n_a} \phi\left(\|s - s_j^a\|/\tau\right)} - \frac{\phi\left(\|s' - s_i^a\|/\tau\right)}{\sum_{j=1}^{n_a} \phi\left(\|s' - s_j^a\|/\tau\right)} \right|.$$

Since ϕ is continuous, it is obvious that $\psi_{\tau,s}^{a,i}(s')$ is also continuous in s'. The property follows from the fact that $\lim_{s' \to s} \psi_{\tau,s}^{a,i}(s') = 0$. ■

Lemma 5† *Let $s \in \mathbb{S}$, let $m > 1$, and assume there is a $w \in \{1, 2, ..., m-1\}$ such that $dist(s,w) < dist(s, w+1)$. Define $W^w(s) \equiv \{k \mid \|s - \bar{s}_k\| \leq dist(s,w)\}$ and $\bar{W}^w(s) \equiv \{1, 2, ..., m\} - W^w(s)$. Then, for any $\alpha > 0$, we can guarantee that*

$$\sum_{k \in \bar{W}^w(s)} \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_k) < \alpha \sum_{k \in W^w(s)} \bar{\kappa}_{\bar{\tau}}(s, \bar{s}_k) \tag{33}$$

by making $\bar{\tau} < \varphi(s, w, m, \alpha)$, where

$$\varphi(s, w, m, \alpha) = \min(\varphi_1(s,w), \varphi_2(s, w, m, \alpha)) \tag{34}$$

and

$$\varphi_1(s,w) = \begin{cases} \frac{dist(s,w)}{B_{\bar{\phi}}}, \text{ if } B_{\bar{\phi}} > 0, \\ \infty, \text{ otherwise,} \end{cases} \quad \varphi_2(s,w,m,\alpha) = \begin{cases} \frac{dist(s,w) - dist(s,w+1)}{\ln(\alpha w/(m-w)\lambda_{\bar{\phi}})}, \text{ if } \frac{\alpha w}{(m-w)\lambda_{\bar{\phi}}} < 1, \\ \infty, \text{ otherwise.} \end{cases}$$

Proof Expression (33) can be rewritten as

$$\frac{\sum_{k \in \bar{W}^w(s)} \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_k)}{\sum_{i=1}^m \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_i)} < \alpha \frac{\sum_{k \in W^w(s)} \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_k)}{\sum_{i=1}^m \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_i)} \iff \sum_{k \in \bar{W}^w(s)} \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_k) < \alpha \sum_{k \in W^w(s)} \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_k), \tag{35}$$

which is equivalent to

$$\sum_{k \in \bar{W}^w(s)} \bar{\phi}\left(\frac{\|s - \bar{s}_k\|}{\bar{\tau}}\right) < \alpha \sum_{k \in W^w(s)} \bar{\phi}\left(\frac{\|s - \bar{s}_k\|}{\bar{\tau}}\right). \tag{36}$$

†. We restate the lemma here showing explicitly how to define $\bar{\tau}$. This detail was omitted in the main body of the text to improve clarity.

Based on Assumption (i), we know that a sufficient condition for (36) to hold is

$$\bar{\phi}\left(\frac{dist(s, w+1)}{\bar{\tau}}\right) < \frac{\alpha w}{m-w}\bar{\phi}\left(\frac{dist(s,w)}{\bar{\tau}}\right). \tag{37}$$

Let $\beta = \alpha w/(m-w)$. If $\beta > 1$, then (37) is always true, regardless of the value of $\bar{\tau}$. We now show that, when $\beta \leq 1$, it is always possible to set $\bar{\tau}$ in order to guarantee that (37) holds. Let $z = dist(s,w)$ and let $\delta = dist(s, w+1) - z$. From Assumption (ii), we know that, if $B_{\bar{\phi}} = 0$ or $\bar{\tau} < z/B_{\bar{\phi}}$,

$$\frac{\bar{\phi}((z+\delta)/\bar{\tau})}{\bar{\phi}(z/\bar{\tau})} \leq \frac{\lambda_{\bar{\phi}} A_{\bar{\phi}} \exp(-(z+\delta)/\bar{\tau})}{A_{\bar{\phi}} \exp(-z/\bar{\tau})} = \frac{\lambda_{\bar{\phi}} \exp(-(z+\delta)/\bar{\tau})}{\exp(-z/\bar{\tau})}.$$

Thus, in order for the result to follow, it suffices to show that

$$\frac{\exp(-(z+\delta)/\bar{\tau})}{\exp(-z/\bar{\tau})} < \frac{\beta}{\lambda_{\bar{\phi}}}. \tag{38}$$

We know that, since $\delta > 0$, if $\beta/\lambda_{\bar{\phi}} = 1$ inequality (38) is true. Otherwise,

$$\frac{\exp(-(z+\delta)/\bar{\tau})}{\exp(-z/\bar{\tau})} < \frac{\beta}{\lambda_{\bar{\phi}}} \iff \ln\left(\frac{\exp(-(z+\delta)/\bar{\tau})}{\exp(-z/\bar{\tau})}\right) < \ln\left(\frac{\beta}{\lambda_{\bar{\phi}}}\right)$$
$$\iff -\frac{\delta}{\bar{\tau}} < \ln\left(\frac{\beta}{\lambda_{\bar{\phi}}}\right) \iff \bar{\tau} < -\frac{\delta}{\ln(\beta/\lambda_{\bar{\phi}})}.$$

Thus, by taking $\bar{\tau} < -\delta/\ln(\beta/\lambda_{\bar{\phi}})$ if $B_{\bar{\phi}} > 0$, or $\bar{\tau} < \min(-\delta/\ln(\beta/\lambda_{\bar{\phi}}), z/B_{\bar{\phi}})$ otherwise, the result follows. ■

Proposition 6
Proof From (6) and (11), we know that

$$\|\hat{\mathbf{r}}^a - \mathbf{D}\bar{\mathbf{r}}^a\|_\infty = \|\hat{\mathbf{P}}^a\mathbf{r} - \mathbf{D}\mathbf{K}^a\mathbf{r}\|_\infty = \|(\hat{\mathbf{P}}^a - \mathbf{D}\mathbf{K}^a)\mathbf{r}\|_\infty \leq \|\hat{\mathbf{P}}^a - \mathbf{D}\mathbf{K}^a\|_\infty \|\mathbf{r}\|_\infty. \tag{39}$$

Thus, plugging (39) back into (8), it is clear that there is a $\nu > 0$ such that $\xi_v < \epsilon$ if

$$\max_a \|\hat{\mathbf{P}}^a - \mathbf{D}\mathbf{K}^a\|_\infty < \nu \tag{40}$$

and

$$\max_i (1 - \max_j d_{ij}) < \nu. \tag{41}$$

We start by showing that there is a $\delta > 0$ and a $\theta > 0$ such that expression (40) is true if $\max_{a,i} dist(\hat{s}_i^a, w) < \delta$ and $\bar{\tau} < \theta$. Let $\check{\mathbf{P}}^a = \mathbf{D}\mathbf{K}^a$ and let $\hat{\mathbf{p}}_i^a \in \mathbb{R}^{1\times n}$ and $\check{\mathbf{p}}_i^a \in \mathbb{R}^{1\times n}$ be the i^{th} rows of $\hat{\mathbf{P}}^a$ and $\check{\mathbf{P}}^a$, respectively. Then,

$$\begin{aligned} \|\hat{\mathbf{p}}_i^a - \check{\mathbf{p}}_i^a\|_\infty &= \textstyle\sum_{j=1}^{n_a} |\hat{p}_{ij}^a - \sum_{k=1}^m \dot{d}_{ik}^a \dot{\kappa}_{kj}^a| \\ &= \textstyle\sum_{j=1}^{n_a} |\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \sum_{k=1}^m \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\kappa_\tau^a(\bar{s}_k, s_j^a)| \\ &= \textstyle\sum_{j=1}^{n_a} |\sum_{k=1}^m \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \sum_{k=1}^m \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\kappa_\tau^a(\bar{s}_k, s_j^a)| \\ &= \textstyle\sum_{j=1}^{n_a} |\sum_{k=1}^m \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)[\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)]| \\ &\leq \textstyle\sum_{j=1}^{n_a} \sum_{k=1}^m \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \left|\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)\right|. \end{aligned} \tag{42}$$

Our strategy will be to show that, for any a, i, and j, there is a $\delta^{a,i,j} > 0$ and a $\theta^{a,i,j} > 0$ such that

$$\sum_{k=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)|\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)| = \sum_{k=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^{a,i,j} < \frac{\nu}{n_a} \tag{43}$$

if $dist(\hat{s}_i^a, w) < \delta^{a,i,j}$ and $\bar{\tau} < \theta^{a,i,j}$ (recall that $\varsigma_k^{a,i,j}$ had already been defined in (16)). To simplify the notation, we will use the superscript 'z' meaning 'a, i, j'. From Lemma 4 we know that there is a $\delta^z > 0$ such that $\varsigma_k^z < \nu/n_a$ if $\|\hat{s}_i^a - \bar{s}_k\| < \delta^z$. Let $W^z \equiv \{k \mid \|\hat{s}_i^a - \bar{s}_k\| < \delta^z\}$ and $\bar{W}^z \equiv \{1, 2, ..., m\} - W^z$. Since we are assuming that $dist(\hat{s}_i^a, w) < \delta^z$, we know that $W^z \neq \emptyset$. In this case, we can write:

$$\sum_{k=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z = \sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z + \sum_{k \in \bar{W}^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z.$$

Let

$$\varsigma_{\min}^z = \begin{cases} \min\limits_{k \in W^z} \{\varsigma_k^z | \varsigma_k^z > 0\} \text{ if } \max\limits_{k \in W^z} \varsigma_k^z > 0, \\ 0 \text{ otherwise} \end{cases} \quad \text{and} \quad \bar{\varsigma}_{\max}^z = \begin{cases} \max\limits_{k \in \bar{W}^z} \varsigma_k^z \text{ if } |W^z| < m, \\ 0 \text{ otherwise.} \end{cases}$$

If $\bar{\varsigma}_{\max}^z = 0$, inequality (43) is necessarily true, since $\sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z \leq \max\limits_{k \in W^z} \varsigma_k^z < \nu/n_a$. We now turn to the case in which $\bar{\varsigma}_{\max}^z > 0$. Suppose first that $\varsigma_{\min}^z = 0$. In this case, we have to show that there is a $\bar{\tau}$ that yields

$$\textstyle\sum_{k \in \bar{W}^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z < \frac{\nu}{n_a}. \tag{44}$$

A sufficient condition for (44) to be true is

$$\sum_{k \in \bar{W}^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) < \frac{\nu}{n_a \bar{\varsigma}_{\max}^z} \iff \frac{1}{\sum_{j=1}^m \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_j)} \sum_{k \in \bar{W}^z} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) < \frac{\nu}{n_a \bar{\varsigma}_{\max}^z}. \tag{45}$$

Obviously, if $\bar{\varsigma}_{\max}^z \leq \nu/n_a$ inequality (45) is always true, regardless of the value of $\bar{\tau}$. Otherwise, we can rewrite (45) as

$$\sum_{k \in \bar{W}^z} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) < \frac{\nu}{n_a \bar{\varsigma}_{\max}^z} \left(\sum_{j \in W^z} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_j) + \sum_{k \in \bar{W}^z} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \right),$$

and, after a few algebraic manipulations, we obtain

$$\sum_{k \in \bar{W}^z} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) < \frac{\nu}{n_a \bar{\varsigma}_{\max}^z - \nu} \sum_{k \in W^z} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \iff \sum_{k \in \bar{W}^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) < \frac{\nu}{n_a \bar{\varsigma}_{\max}^z - \nu} \sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k). \tag{46}$$

We can guarantee that (46) is true by applying Lemma 5. Before doing so, though, lets analyze the case in which $\varsigma_{\min}^z > 0$. Define

$$\beta^z = \frac{\nu}{n_a \sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z} - 1 \tag{47}$$

(note that $\beta^z > 0$ because $\sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z < v/n_a$). In order for (43) to hold, we must show that there is a $\bar{\tau}$ that guarantees that

$$\textstyle\sum_{k \in \bar{W}^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z - \beta^z \sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k)\varsigma_k^z < 0. \tag{48}$$

A sufficient condition for (48) to hold is

$$\textstyle\sum_{k \in \bar{W}^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) < \frac{\beta^z \varsigma_{\min}^z}{\bar{\varsigma}_{\max}^z} \sum_{k \in W^z} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k). \tag{49}$$

Observe that expressions (46) and (49) only differ in the coefficient multiplying the right-hand side of the inequalities. Let

$$\alpha_1^z = \begin{cases} \nu/(\bar{\varsigma}_{\max}^z n_a - \nu), \text{ if } \bar{\varsigma}_{\max}^z > \nu/n_a \\ \infty, \text{otherwise,} \end{cases}$$

and

$$\alpha_2^z = \begin{cases} \beta^z \varsigma_{\min}^z / \bar{\varsigma}_{\max}^z, \text{ if } \bar{\varsigma}_{\max}^z > 0 \text{ and } \varsigma_{\min}^z > 0, \\ \infty, \text{ otherwise.} \end{cases}$$

Let $\alpha^z < \min(\alpha_1^z, \alpha_2^z)$. Then, if we make $\theta^z = \varphi(\hat{s}_i^a, |W|, m, \alpha^z)$, with φ defined in (34), we can apply Lemma 5 to guarantee that (43) holds. Finally, if we let $\delta = \min_z \delta^z = \min_{a,i,j} \delta^{a,i,j}$ and $\theta = \min_z \theta^z = \min_{a,i,j} \theta^{a,i,j}$, we can guarantee that (43) is true for all a, i, and j, which implies that (40) is also true (see (42)).

It remains to show that there is a $\omega > 0$ such that (41) is true if $\bar{\tau} < \omega$. Recalling that, for any i and any a,

$$\max_j \dot{d}_{ij}^a = \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)),$$

we want to show that

$$\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) > (1 - \nu) \left[\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) + \sum_{k=2}^{m} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, k)) \right],$$

which is equivalent to

$$(1 - \nu) \sum_{k=2}^{m} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, k)) < \nu \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)). \tag{50}$$

If $\nu \geq 1$, inequality (50) is true regardless of the particular choice of $\bar{\tau}$. Otherwise, we can rewrite (50) as

$$\sum_{k=2}^{m} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, k)) < \frac{\nu}{1 - \nu} \bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) \iff \sum_{k=2}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, k)) < \frac{\nu}{1 - \nu} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)). \tag{51}$$

Let

$$\alpha = \begin{cases} \nu/(1 - \nu), \text{ if } \nu < 1, \\ \infty, \text{ otherwise.} \end{cases}$$

Then, if we make $\omega^{a,i} = \varphi(\hat{s}_i^a, 1, m, \alpha)$, with φ defined in (34), we can resort to Lemma 5 to

guarantee that (51) holds. As before, if we let $\omega = \min_{a,i} \omega^{a,i}$, we can guarantee that (41) is true. Finally, by making $\bar{\tau} = \min(\theta, \omega)$, the result follows. ■

Proposition 8

Proof

We start by plugging (39) into the definition of ξ_v, given in (8), to get:

$$\begin{aligned}
\xi_v &\leq \frac{1}{1-\gamma} \max_a \|\hat{\mathbf{P}}^a - \mathbf{D}\mathbf{K}^a\|_\infty \|\mathbf{r}\|_\infty + \frac{\bar{R}_{\text{dif}}}{(1-\gamma)^2} \left(\frac{\gamma}{2} \max_a \|\mathbf{P}^a - \mathbf{D}\mathbf{K}^a\|_\infty + \sigma(\mathbf{D}) \right) \\
&= \max_a \|\mathbf{P}^a - \mathbf{D}\mathbf{K}^a\|_\infty \left(\frac{\|\mathbf{r}\|_\infty}{1-\gamma} + \frac{\gamma \bar{R}_{\text{dif}}}{2(1-\gamma)^2} \right) + \sigma(\mathbf{D}) \frac{\bar{R}_{\text{dif}}}{(1-\gamma)^2}. \\
&\leq \max_a \|\mathbf{P}^a - \mathbf{D}\mathbf{K}^a\|_\infty \left(\frac{R_{\max}}{1-\gamma} + \frac{\gamma R_{\max}}{(1-\gamma)^2} \right) + \sigma(\mathbf{D}) \frac{R_{\max}}{2(1-\gamma)^2} \\
&= \max_a \|\mathbf{P}^a - \mathbf{D}\mathbf{K}^a\|_\infty \frac{R_{\max}}{(1-\gamma)^2} + \sigma(\mathbf{D}) \frac{R_{\max}}{2(1-\gamma)^2}, \qquad (52)
\end{aligned}$$

where we used the fact that $\bar{R}_{\text{dif}} \leq R_{\max}/2$, which follows trivially from the observation that $\min_{a,i} \bar{r}_i^a \geq \min_{a,i} r_i^a$ and $\max_{a,i} \bar{r}_i^a \leq \max_{a,i} r_i^a$. To simplify the notation, let $\mathcal{R} = R_{\max}/(1-\gamma)^2$. Then, from (42), the definition of $\varsigma_k^{a,i,j}$ in (16), and the definition of $\sigma(\mathbf{D})$ in (9), we can rewrite (52) as

$$\begin{aligned}
\xi_v &\leq \max_{a,i} \left(\sum_{j=1}^{n_a} \sum_{k=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \varsigma_k^{a,i,j} \right) \mathcal{R} + \max_{a,i} \left[1 - \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) \right] \frac{\mathcal{R}}{2} \\
&= \max_{a,i} \left(\sum_{k=1}^{m} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} \right) \mathcal{R} + \max_{a,i} \left[1 - \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) \right] \frac{\mathcal{R}}{2}
\end{aligned}$$

Let $W^w(\hat{s}_i^a)$ and $\bar{W}^w(\hat{s}_i^a)$ be defined as in (15). Then,

$$\begin{aligned}
\xi_v &\leq \max_{a,i} \left(\sum_{k \in W^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} + \sum_{k \in \bar{W}^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} \right) \mathcal{R} + \max_{a,i} \left(1 - \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) \right] \frac{\mathcal{R}}{2} \\
&\leq \max_{a,i} \left(\sum_{k \in W^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} \right) \mathcal{R} \\
&\quad + \underbrace{\max_{a,i} \left(\sum_{k \in \bar{W}^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} \right) \mathcal{R} + \max_{a,i} \left(1 - \bar{\kappa}_{\bar{\tau}}(\hat{s}_i^a, rs(\hat{s}_i^a, 1)) \right] \frac{\mathcal{R}}{2}}_{\mathcal{F}(\bar{\tau}, w)}. \qquad (53)
\end{aligned}$$

We see that the two last terms of (53) coincide with the definition of $\mathcal{F}(\bar{\tau}, w)$, given in (17). Thus, if we make sure that $\bar{\kappa}_{\bar{\tau}}$ is an admissible kernel $\bar{\kappa}_{\bar{\tau}}^{\epsilon,w}$ (see Definition 7), we can

rewrite (53) as

$$\begin{aligned}
\xi_v &\leq \max_{a,i} \left(\sum_{k \in W^w(\hat{s}_i^a)} \bar{\kappa}_{\bar{\tau}}^{\epsilon,w}(\hat{s}_i^a, \bar{s}_k) \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} \right) \mathcal{R} + \epsilon \\
&\leq \max_{a,i} \left(w \max_{k \in W^w(\hat{s}_i^a)} \sum_{j=1}^{n_a} \varsigma_k^{a,i,j} \right) \mathcal{R} + \epsilon \\
&= w \max_{a,i} \left(\max_{k \in W^w(\hat{s}_i^a)} \sum_{j=1}^{n_a} |\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)| \right) \mathcal{R} + \epsilon. \qquad (54)
\end{aligned}$$

Let $z_i^a = \sum_l \mathrm{k}_\tau(\hat{s}_i^a, s_l^a)$ and $\bar{z}_k^a = \sum_l \mathrm{k}_\tau(\bar{s}_k, s_l^a)$. Then, for fixed a, i, j, and k:

$$\begin{aligned}
|\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)| &= \left| \frac{\mathrm{k}_\tau(\hat{s}_i^a, s_j^a)}{\sum_l \mathrm{k}_\tau(\hat{s}_i^a, s_l^a)} - \frac{\mathrm{k}_\tau(\bar{s}_k, s_j^a)}{\sum_l \mathrm{k}_\tau(\bar{s}_k, s_l^a)} \right| \\
&= \left| \frac{\mathrm{k}_\tau(\hat{s}_i^a, s_j^a)}{z_i^a} - \frac{\mathrm{k}_\tau(\bar{s}_k, s_j^a)}{\bar{z}_k^a} \right| \\
&= \left| \frac{\bar{z}_k^a \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - z_i^a \mathrm{k}_\tau(\bar{s}_k, s_j^a)}{z_i^a \bar{z}_k^a} \right| \\
&= \left| \frac{\bar{z}_k^a \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - z_i^a \mathrm{k}_\tau(\bar{s}_k, s_j^a) + z_i^a \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - z_i^a \mathrm{k}_\tau(\hat{s}_i^a, s_j^a)}{z_i^a \bar{z}_k^a} \right| \\
&= \left| \frac{\mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a)}{\bar{z}_k^a} + \frac{\mathrm{k}_\tau(\hat{s}_i^a, s_j^a)(\bar{z}_k^a - z_i^a)}{z_i^a \bar{z}_k^a} \right| \\
&\leq \frac{\left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right|}{\bar{z}_k^a} + \frac{\mathrm{k}_\tau(\hat{s}_i^a, s_j^a) |\bar{z}_k^a - z_i^a|}{z_i^a \bar{z}_k^a}.
\end{aligned}$$

Thus,

$$\begin{aligned}
\sum_{j=1}^{n_a} |\kappa_\tau^a(\hat{s}_i^a, s_j^a) - \kappa_\tau^a(\bar{s}_k, s_j^a)| &\leq \sum_{j=1}^{n_a} \frac{\left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right|}{\bar{z}_k^a} + \sum_{j=1}^{n_a} \frac{\mathrm{k}_\tau(\hat{s}_i^a, s_j^a) |\bar{z}_k^a - z_i^a|}{z_i^a \bar{z}_k^a} \\
&= \sum_{j=1}^{n_a} \frac{\left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right|}{\bar{z}_k^a} + \sum_{j=1}^{n_a} \kappa_\tau^a(\hat{s}_i^a, s_j^a) \frac{|\bar{z}_k^a - z_i^a|}{\bar{z}_k^a} \\
&= \sum_{j=1}^{n_a} \frac{\left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right|}{\bar{z}_k^a} + \frac{|\bar{z}_k^a - z_i^a|}{\bar{z}_k^a} \\
&= \frac{1}{\sum_{l=1}^{n_a} \mathrm{k}_\tau(\bar{s}_k, s_l^a)} \left(\sum_{j=1}^{n_a} \left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right| + \left| \sum_{j=1}^{n_a} \mathrm{k}_\tau(\bar{s}_k, s_j^a) - \sum_{j=1}^{n_a} \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) \right| \right) \\
&\leq \frac{1}{\sum_{l=1}^{n_a} \mathrm{k}_\tau(\bar{s}_k, s_l^a)} \left(\sum_{j=1}^{n_a} \left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right| + \sum_{j=1}^{n_a} \left| \mathrm{k}_\tau(\bar{s}_k, s_j^a) - \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) \right| \right) \\
&= \frac{2}{\sum_{l=1}^{n_a} \mathrm{k}_\tau(\bar{s}_k, s_l^a)} \sum_{j=1}^{n_a} \left| \mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a) \right|. \qquad (55)
\end{aligned}$$

In order to derive our result, it suffices to replace (55) in (54). Before doing so, though, note that Assumption (iii) implies that

$$\begin{aligned}\left|\mathrm{k}_\tau(s,s') - \mathrm{k}_\tau(s,s'')\right| &= \left|\phi\left(\frac{\|s-s'\|}{\tau}\right) - \phi\left(\frac{\|s-s''\|}{\tau}\right)\right| \\ &\leq \frac{C_\phi}{\tau}\left|\|s-s'\| - \|s-s''\|\right| \\ &\leq \frac{C_\phi}{\tau}\left|\|s-s''\| + \|s''-s'\| - \|s-s''\|\right| \\ &= \frac{C_\phi}{\tau}\|s''-s'\|. \end{aligned} \tag{56}$$

Proceeding with the substitution mentioned above, we have

$$\begin{aligned}\xi_v &\leq 2w \max_{a,i}\left(\max_{k\in W^w(\hat{s}_i^a)} \frac{1}{\sum_{l=1}^{n_a}\mathrm{k}_\tau(\bar{s}_k, s_l^a)} \sum_{j=1}^{n_a}\left|\mathrm{k}_\tau(\hat{s}_i^a, s_j^a) - \mathrm{k}_\tau(\bar{s}_k, s_j^a)\right|\right)\mathcal{R} + \epsilon \\ &\leq 2w \max_{a,i}\left(\max_{k\in W^w(\hat{s}_i^a)} \frac{1}{\sum_{l=1}^{n_a}\mathrm{k}_\tau(\bar{s}_k, s_l^a)} \frac{C_\phi}{\tau} n_a \|\hat{s}_i^a - \bar{s}_k\|\right)\mathcal{R} + \epsilon \\ &= \frac{2wC_\phi}{\tau}\max_{a,i}\left(\max_{k\in W^w(\hat{s}_i^a)} \frac{n_a\|\hat{s}_i^a - \bar{s}_k\|}{\sum_{l=1}^{n_a}\mathrm{k}_\tau(\bar{s}_k, s_l^a)}\right)\mathcal{R} + \epsilon \\ &\leq \frac{2wC_\phi}{\tau}\max_{a,i}\left(\max_{k\in W^w(\hat{s}_i^a)} \frac{n_a\|\hat{s}_i^a - \bar{s}_k\|}{n_a \mathrm{k}_{\min}}\right)\mathcal{R} + \epsilon \\ &= \frac{2wC_\phi}{\tau\mathrm{k}_{\min}}\max_{a,i}\max_{k\in W^w(\hat{s}_i^a)}\|\hat{s}_i^a - \bar{s}_k\|\mathcal{R} + \epsilon. \end{aligned}$$

■

Lemma 9

Proof Let $\mathbf{q}_*^a, \tilde{\mathbf{q}}_*^a \in \mathbb{R}^{|S|}$ be the a^{th} columns of $\mathbf{Q}^*$ and $\tilde{\mathbf{Q}}^*$, respectively. Then,

$$\begin{aligned}\|\mathbf{q}_*^a - \tilde{\mathbf{q}}_*^a\|_\infty &= \left\|\mathbf{r}^a + \gamma\mathbf{P}^a\mathbf{v}^* - \tilde{\mathbf{r}}^a - \gamma\tilde{\mathbf{P}}^a\tilde{\mathbf{v}}^*\right\|_\infty \\ &\leq \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \gamma\left\|\mathbf{P}^a\mathbf{v}^* - \tilde{\mathbf{P}}^a\tilde{\mathbf{v}}^*\right\|_\infty \\ &= \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \gamma\left\|\mathbf{P}^a\mathbf{v}^* - \tilde{\mathbf{P}}^a\mathbf{v}^* + \tilde{\mathbf{P}}^a\mathbf{v}^* - \tilde{\mathbf{P}}^a\tilde{\mathbf{v}}^*\right\|_\infty \\ &\leq \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \gamma\left\|(\mathbf{P}^a - \tilde{\mathbf{P}}^a)\mathbf{v}^*\right\|_\infty + \gamma\left\|\tilde{\mathbf{P}}^a(\mathbf{v}^* - \tilde{\mathbf{v}}^*)\right\|_\infty \\ &\leq \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \gamma\left\|(\mathbf{P}^a - \tilde{\mathbf{P}}^a)\mathbf{v}^*\right\|_\infty + \gamma\|\mathbf{v}^* - \tilde{\mathbf{v}}^*\|_\infty, \end{aligned} \tag{57}$$

where in the last step we used the fact that $\tilde{\mathbf{P}}^a$ is stochastic, and thus $\|\tilde{\mathbf{P}}^a\mathbf{v}\|_\infty \leq \|\mathbf{v}\|_\infty$ for any $\mathbf{v}$. We now provide a bound for $\|(\mathbf{P}^a - \tilde{\mathbf{P}}^a)\mathbf{v}^*\|_\infty$. Let $\mathbf{A} = \mathbf{P}^a - \tilde{\mathbf{P}}^a$. Then, for any i, $\sum_j a_{ij} = \sum_j (p_{ij}^a - \tilde{p}_{ij}^a) = \sum_j p_{ij}^a - \sum_j \tilde{p}_{ij}^a = 0$, that is, the elements in each row of $\mathbf{A}$ sum to zero. Let a_i^+ be the sum of positive elements in the i^{th} row of $\mathbf{A}$ and let $a_{\max}^+ = \max_i a_i^+$.

It should be clear that $\|\mathbf{A}\|_\infty = 2a_{\max}^+$. Then, for any i,

$$\begin{aligned}\left|\sum_j a_{ij}v_j^*\right| &\leq \sum_{(j:a_{ij}>0)} a_{ij}v_{\max}^* + \sum_{(j:a_{ij}<0)} a_{ij}v_{\min}^* = a_i^+ v_{\max}^* - a_i^+ v_{\min}^* \leq a_{\max}^+(v_{\max}^* - v_{\min}^*) \\ &\leq \frac{a_{\max}^+}{1-\gamma}(r_{\max}^a - r_{\min}^a) \leq \frac{a_{\max}^+ R_{\text{dif}}}{1-\gamma} = \frac{R_{\text{dif}}}{2(1-\gamma)}\|\mathbf{P}^a - \tilde{\mathbf{P}}^a\|_\infty, \end{aligned} \tag{58}$$

where we used the convention $v_{\max} = \max_i v_i$ (analogously for $v_{\min}$). As done in (30), we can resort to Whitt's (1978) Theorem 3.1 and Corollary (b) of his Theorem 6.1 to obtain a bound for $\|\mathbf{v}^* - \tilde{\mathbf{v}}^*\|_\infty$. Substituting such a bound and expression (58) in (57), we obtain

$$\begin{aligned}\|\mathbf{q}_*^a - \tilde{\mathbf{q}}_*^a\|_\infty &\leq \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \frac{\gamma R_{\text{dif}}}{2(1-\gamma)}\|\mathbf{P}^a - \tilde{\mathbf{P}}^a\|_\infty + \frac{\gamma}{1-\gamma}\left(\max_a \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \frac{\gamma R_{\text{dif}}}{2(1-\gamma)}\max_a \|\mathbf{P}^a - \tilde{\mathbf{P}}^a\|_\infty\right) \\ &\leq \max_a \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \frac{\gamma R_{\text{dif}}}{2(1-\gamma)}\max_a \|\mathbf{P}^a - \tilde{\mathbf{P}}^a\|_\infty + \frac{\gamma}{1-\gamma}\left(\max_a \|\mathbf{r}^a - \tilde{\mathbf{r}}^a\|_\infty + \frac{\gamma R_{\text{dif}}}{2(1-\gamma)}\max_a \|\mathbf{P}^a - \tilde{\mathbf{P}}^a\|_\infty\right). \end{aligned}$$

■

Proposition 10

Proof Let $\check{M}_t \equiv (\hat{S}_t, A, \check{\mathbf{P}}_t^a, \check{\mathbf{r}}_t^a, \gamma)$, with $\check{\mathbf{P}}_t^a = \mathbf{D}_t\mathbf{K}_t^a$ and $\check{\mathbf{r}}_t^a = \mathbf{D}_t\bar{\mathbf{r}}_t^a$. From the triangle inequality, we know that

$$|\hat{Q}_t(s,a) - \tilde{Q}_t(s,a)| \leq |\hat{Q}_t(s,a) - \check{Q}_t^*(s,a)| + |\check{Q}_t^*(s,a) - \tilde{Q}_t^*(s,a)| + |\tilde{Q}_t^*(s,a) - \tilde{Q}_t(s,a)|, \tag{59}$$

where $\hat{Q}_t$ and $\tilde{Q}_t$ are defined in the proposition's statement, $\check{Q}_t^*$ is the optimal action-value function of $\check{M}_t$, and $\tilde{Q}_t^*(s,a) = \sum_{i=1}^m \bar{\kappa}_{\bar{\tau}}(s,\bar{s}_i)\bar{Q}_t^*(\bar{s}_i,a)$ (the reader will forgive a slight abuse of notation here, since in general $\bar{Q}_t^*$ is not the optimal value function of any MDP). Our strategy will be to bound each term on the right-hand side of (59). Since $\hat{M}_t$ is the model constructed by KBRL using all the data seen by iKBSF up to time step t, state s will correspond to one of the states $\hat{s}_i^b$ in this MDP. Thus, from (7), we see that $\hat{Q}_t(s,a) = \hat{Q}_t^*(\hat{s}_i^b, a)$ for some i and some b. Therefore, applying Lemma 9 to $\hat{M}_t$ and $\check{M}_t$, we can write

$$|\hat{Q}_t(s,a) - \check{Q}_t^*(s,a)| \leq \frac{1}{1-\gamma}\max_a \|\hat{\mathbf{r}}_t^a - \mathbf{D}_t\bar{\mathbf{r}}_t^a\|_\infty + \frac{\gamma}{2(1-\gamma)^2}\bar{R}_{\text{dif},t}\max_a \|\hat{\mathbf{P}}_t^a - \mathbf{D}_t\mathbf{K}_t^a\|_\infty. \tag{60}$$

In order to bound $|\check{Q}_t^*(s,a) - \tilde{Q}_t^*(s,a)|$, we note that, since the information contained in the transition to state s has been incorporated to iKBSF's model $\bar{M}$ at time t, $\tilde{Q}_t^*(s,a) = \sum_{i=1}^m d_{ti,t}\bar{Q}_t^*(\bar{s}_i,a)$, for any $a \in A$, where $d_{ti,t}$ is the element in the t^{th} row and i^{th} column of $\mathbf{D}_t$ (see Figure 2b). In matrix form, we have $\tilde{\mathbf{Q}}_t^* = \mathbf{D}_t\bar{\mathbf{Q}}_t^*$. As $\mathbf{D}_t$ is a soft homomorphism between $\bar{M}_t$ and $\bar{M}_t$, we can resort to Sorg and Singh's (2009) Theorem 1, as done in Proposition 2, to write:

$$|\check{Q}_t^*(s,a) - \tilde{Q}_t^*(s,a)| \leq \frac{\bar{R}_{\text{dif},t}}{(1-\gamma)^2}\sigma(\mathbf{D}_t) \tag{61}$$

(see (31) and (32)). Finally,

$$|\tilde{Q}_t^*(s,a) - \tilde{Q}_t(s,a)| = \left|\sum_{i=1}^{m} \bar{\kappa}_{\bar{\tau}}(s,\bar{s}_i)\bar{Q}_t^*(\bar{s}_i,a) - \sum_{i=1}^{m} \bar{\kappa}_{\bar{\tau}}(s,\bar{s}_i)\bar{Q}_t(\bar{s}_i,a)\right|$$
$$\leq \sum_{i=1}^{m} \bar{\kappa}_{\bar{\tau}}(s,\bar{s}_i)\left|\bar{Q}_t^*(\bar{s}_i,a) - \bar{Q}_t(\bar{s}_i,a)\right| \leq \epsilon_{\bar{Q}_t}, \tag{62}$$

where the last step follows from the fact that $\sum_{i=1}^{m} \bar{\kappa}_{\bar{\tau}}(s,\bar{s}_i)$ is a convex combination. Substituting (60), (61), and (62) in (59), we obtain the desired bound. ■

A.2 Alternative error bound

In Section 3 we derived an upper bound for the approximation error introduced by the application of the stochastic-factorization trick. In this section we introduce another bound that has different properties. First, the bound is less applicable, because it depends on quantities that are usually unavailable in a practical situation (the fixed points of two contraction mappings). On the bright side, unlike the bound presented in Proposition 2, the new bound is valid for any norm. Also, it draws an interesting connection with an important class of approximators known as *averagers* (Gordon, 1995).

We start by deriving a theoretical result that only applies to stochastic factorizations of order n. We then generalize this result to the case in which the factorizations are of order $m < n$.

Lemma 11 *Let $M \equiv (S, A, \mathbf{P}^a, \mathbf{r}^a, \gamma)$ be a finite MDP with $|S| = n$ and $0 \leq \gamma < 1$. Let $\mathbf{E}\mathbf{L}^a = \mathbf{P}^a$ be $|A|$ stochastic factorizations of order n and let $\bar{\bar{\mathbf{r}}}^a$ be vectors in $\mathbb{R}^n$ such that $\mathbf{E}\bar{\bar{\mathbf{r}}}^a = \mathbf{r}^a$ for all $a \in A$. Define the MDPs $\check{M} \equiv (S, A, \mathbf{L}^a, \bar{\bar{\mathbf{r}}}^a, \gamma)$ and $\bar{\bar{M}} \equiv (S, A, \bar{\bar{\mathbf{P}}}^a, \bar{\bar{\mathbf{r}}}^a, \gamma)$, with $\bar{\bar{\mathbf{P}}}^a = \mathbf{L}^a\mathbf{E}$. Then,*

$$\|\mathbf{v}^* - T\mathbf{E}\check{\mathbf{v}}^*\| \leq \xi_v' \equiv \frac{2\gamma}{1-\gamma}\|\mathbf{v}^* - \mathbf{u}\| + \frac{\gamma(1+\gamma)}{1-\gamma}\|\mathbf{v}^* - \check{\mathbf{v}}^*\|, \tag{63}$$

where $\|\cdot\|$ is a norm in R^n and $\mathbf{u}$ is a vector in $\mathbb{R}^n$ such that $\mathbf{E}\mathbf{u} = \mathbf{u}$.

Proof The Bellman operators of M, $\check{M}$, and $\bar{\bar{M}}$ are given by $T = \Gamma\Delta$, $\check{T} = \Gamma\check{\Delta}$, and $\bar{\bar{T}} = \Gamma\bar{\bar{\Delta}}$. Note that $\mathbf{q}^a = \mathbf{r}^a + \gamma\mathbf{P}^a\mathbf{v} = \mathbf{E}\bar{\bar{\mathbf{r}}}^a + \gamma\mathbf{E}\mathbf{L}^a\mathbf{v} = \mathbf{E}(\bar{\bar{\mathbf{r}}}^a + \gamma\mathbf{L}^a\mathbf{v})$, where $\mathbf{q}^a$ is the a^{th} column of $\mathbf{Q}$. Thus, $\Delta = \mathbf{E}\check{\Delta}$. Since $\mathbf{E}$ is stochastic, we can think of it as one of Gordon's (1995) averagers given by $A(\mathbf{v}) = \mathbf{E}\mathbf{v}$, and then resort to Theorem 4.1 by the same author to conclude that $\bar{\bar{T}} = \mathbf{E}\check{T}$. Therefore,[7]

$$T\mathbf{v} = \Gamma\mathbf{E}\check{\Delta}\mathbf{v} \quad \text{and} \quad \bar{\bar{T}}\mathbf{v} = \mathbf{E}\Gamma\check{\Delta}\mathbf{v}. \tag{64}$$

7. Interestingly, the effect of swapping matrices $\mathbf{E}$ and $\mathbf{L}^a$ is to also swap the operators Γ and $\mathbf{E}$.

Using (64), it is easy to obtain the desired upper bound by resorting to the triangle inequality, the definition of a contraction map, and Denardo's (1967) Theorem 1:

$$\begin{aligned}
\|\mathbf{v}^* - T\mathbf{E}\bar{\bar{\mathbf{v}}}^*\| &\leq \gamma\|\mathbf{v}^* - \mathbf{E}\bar{\bar{\mathbf{v}}}^*\| \leq \gamma(\|\mathbf{v}^* - \mathbf{u}\| + \|\mathbf{u} - \mathbf{E}\bar{\bar{\mathbf{v}}}^*\|) \leq \gamma(\|\mathbf{v}^* - \mathbf{u}\| + \|\mathbf{u} - \check{\mathbf{v}}^*\|) \\
&\leq \gamma\left(\|\mathbf{v}^* - \mathbf{u}\| + \frac{1}{1-\gamma}\|\mathbf{u} - \mathbf{E}\Gamma\check{\Delta}\mathbf{u}\|\right) \leq \gamma\left(\|\mathbf{v}^* - \mathbf{u}\| + \frac{1}{1-\gamma}\|\mathbf{u} - \Gamma\check{\Delta}\mathbf{u}\|\right) \\
&\leq \gamma\left[\|\mathbf{v}^* - \mathbf{u}\| + \frac{1}{1-\gamma}\left(\|\mathbf{u} - \check{\mathbf{v}}^*\| + \|\check{\mathbf{v}}^* - \Gamma\check{\Delta}\mathbf{u}\|\right)\right] \\
&\leq \gamma\left[\|\mathbf{v}^* - \mathbf{u}\| + \frac{1}{1-\gamma}\left(\|\mathbf{u} - \check{\mathbf{v}}^*\| + \gamma\|\check{\mathbf{v}}^* - \mathbf{u}\|\right)\right] = \gamma\left[\|\mathbf{v}^* - \mathbf{u}\| + \frac{1+\gamma}{1-\gamma}\|\mathbf{u} - \check{\mathbf{v}}^*\|\right] \\
&\leq \gamma\left[\|\mathbf{v}^* - \mathbf{u}\| + \frac{1+\gamma}{1-\gamma}\left(\|\mathbf{u} - \mathbf{v}^*\| + \|\mathbf{v}^* - \check{\mathbf{v}}^*\|\right)\right] \\
&= \gamma\|\mathbf{v}^* - \mathbf{u}\| + \frac{\gamma(1+\gamma)}{1-\gamma}\|\mathbf{v}^* - \mathbf{u}\| + \frac{\gamma(1+\gamma)}{1-\gamma}\|\mathbf{v}^* - \check{\mathbf{v}}^*\| \\
&= \frac{\gamma - \gamma^2 + \gamma + \gamma^2}{1-\gamma}\|\mathbf{v}^* - \mathbf{u}\| + \frac{\gamma(1+\gamma)}{1-\gamma}\|\mathbf{v}^* - \check{\mathbf{v}}^*\|.
\end{aligned}$$

■

The derived upper bound depends on two fixed points: $\mathbf{u}$, a fixed point of $\mathbf{E}$, and $\check{\mathbf{v}}^*$, the unique fixed point of $\check{T} = \Gamma\check{\Delta}$. Since the latter is defined by $\bar{\bar{\mathbf{r}}}^a$ and $\mathbf{L}^a$, the bound is essentially a function of the factorization terms, as expected. Notice that the bound is valid for any norm and any fixed point of $\mathbf{E}$ (we may think of $\mathbf{u}$ as the closest vector to $\mathbf{v}^*$ in $\mathbb{R}^n$ which satisfies this property). Notice also that the first term on the right-hand side of (63) is exactly the error bound derived in Gordon's (1995) Theorem 6.2. When $\mathbf{L}^a = \mathbf{P}^a$ and $\mathbf{r}^a = \bar{\bar{\mathbf{r}}}^a$ for all $a \in A$, the operators T and $\check{T}$ coincide, and hence the second term of (63) vanishes. This makes sense, since in this case $\bar{\bar{T}} = \mathbf{E}T$, that is, the stochastic-factorization trick reduces to the averager $A(\mathbf{v}) = \mathbf{E}\mathbf{v}$.

As mentioned above, one of the assumptions of Lemma 11 is that the factorizations $\mathbf{E}\mathbf{L}^a = \mathbf{P}^a$ are of order n. This is unfortunate, since the whole motivation behind the stochastic-factorization trick is to create an MDP with $m < n$ states. One way to obtain such a reduction is to suppose that matrix $\mathbf{E}$ has $n - m$ columns with zeros only. Define $\mathcal{E} \subset \{1, 2, ..., n\}$ as the set of columns of $\mathbf{E}$ with at least one nonzero element and let $\mathbf{H}$ be a matrix in $\mathbb{R}^{m\times n}$ such that $h_{ij} = 1$ if j is the i^{th} smallest element in $\mathcal{E}$ and $h_{ij} = 0$ otherwise. The following proposition generalizes the previous result to a stochastic factorization of order m:

Proposition 12 *Suppose the assumptions of Lemma 11 hold. Let $\mathbf{D} = \mathbf{E}\mathbf{H}^\intercal$, $\mathbf{K}^a = \mathbf{H}\mathbf{L}^a$, and $\bar{\mathbf{r}}^a = \mathbf{H}\bar{\bar{\mathbf{r}}}^a$, with $\mathbf{H}$ defined as described above. Define the MDP $\bar{M} \equiv (\bar{S}, A, \bar{\mathbf{P}}^a, \bar{\mathbf{r}}^a, \gamma)$, with $|\bar{S}| = m$ and $\bar{\mathbf{P}}^a = \mathbf{K}^a\mathbf{D}$. Then, $\|\mathbf{v}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\| \leq \xi_v'$, with ξ_v' defined in (63).*

Proof Let $\bar{\mathbf{q}}_*^a \in \mathbb{R}^m$ be the a^{th} column of $\bar{\mathbf{Q}}^*$. Then,

$$\begin{aligned}
\mathbf{D}\bar{\mathbf{q}}_*^a &= \mathbf{D}\left(\bar{\mathbf{r}}^a + \gamma\bar{\bar{\mathbf{P}}}^a\bar{\mathbf{v}}^*\right) = \mathbf{D}\bar{\mathbf{r}}^a + \gamma\mathbf{D}\mathbf{K}^a\mathbf{D}\bar{\mathbf{v}}^* = \mathbf{E}\mathbf{H}^\intercal\mathbf{H}\bar{\bar{\mathbf{r}}}^a + \gamma\mathbf{E}\mathbf{H}^\intercal\mathbf{H}\mathbf{L}^a\mathbf{E}\mathbf{H}^\intercal\bar{\mathbf{v}}^* \\
&= \mathbf{E}\bar{\bar{\mathbf{r}}}^a + \gamma\mathbf{E}\bar{\bar{\mathbf{P}}}^a\mathbf{H}^\intercal\bar{\mathbf{v}}^* = \mathbf{E}\bar{\bar{\mathbf{r}}}^a + \gamma\mathbf{E}\bar{\bar{\mathbf{P}}}^a\bar{\bar{\mathbf{v}}}^* = \mathbf{E}\left(\bar{\bar{\mathbf{r}}}^a + \gamma\bar{\bar{\mathbf{P}}}^a\bar{\bar{\mathbf{v}}}^*\right) = \mathbf{E}\bar{\bar{\mathbf{q}}}_*^a,
\end{aligned} \tag{65}$$

where the equality $\mathbf{E}\mathbf{H}^\intercal\mathbf{H} = \mathbf{E}$ follows from the definition of $\mathbf{H}$. The identity $\bar{\bar{\mathbf{P}}}^a\mathbf{H}^\intercal\bar{\mathbf{v}}^* = \bar{\bar{\mathbf{P}}}^a\bar{\bar{\mathbf{v}}}^*$ is a consequence of the fact that the i^{th} column of $\bar{\bar{\mathbf{P}}}^a$ only contains zeros if $i \notin \mathcal{E}$. Since the i^{th} state of $\bar{\bar{M}}$ is transient, the values of the recurrent states of $\bar{\bar{M}}$, which effectively define the multiplication $\bar{\bar{\mathbf{P}}}^a\bar{\bar{\mathbf{v}}}^*$, will coincide with the values of the states of $\bar{M}$. Expression (65) then leads to $\mathbf{D}\bar{\mathbf{Q}}^* = \mathbf{E}\bar{\bar{\mathbf{Q}}}^*$. Also, since $\mathbf{E}\bar{\bar{\mathbf{q}}}^a_* = \mathbf{E}\bar{\bar{\mathbf{r}}}^a + \gamma\mathbf{E}\mathbf{L}^a\mathbf{E}\bar{\bar{\mathbf{v}}}^* = \mathbf{r}^a + \gamma\mathbf{P}^a\mathbf{E}\bar{\bar{\mathbf{v}}}^*$, we know that $\mathbf{E}\bar{\bar{\mathbf{Q}}}^* = \Delta\mathbf{E}\bar{\bar{\mathbf{v}}}^*$. Putting these results together, we obtain $\|\mathbf{v}^* - \Gamma\mathbf{D}\bar{\mathbf{Q}}^*\| = \|\mathbf{v}^* - \Gamma\Delta\mathbf{E}\bar{\bar{\mathbf{v}}}^*\| = \|\mathbf{v}^* - T\mathbf{E}\bar{\bar{\mathbf{v}}}^*\|$, and Lemma 11 applies. ■

The derived bound can be generalized to the case of approximate stochastic factorizations through the triangle inequality, as done in (29). However, if one resorts to Whitt's (1978) results to bound the distance between $\mathbf{v}^*$ and $\check{\mathbf{v}}^*$—where $\check{\mathbf{v}}^*$ is the optimal value function of $\check{M} \equiv (S, A, \mathbf{D}\mathbf{K}^a, \mathbf{D}\bar{\mathbf{r}}^a, \gamma)$—the compounded bound will no longer be valid for all norms, since (30) only holds for the infinity norm.

Appendix B. Details of the experiments

This appendix describes the details of the experiments omitted in the paper.

B.1 Tasks

Puddle World: The puddle-world task was implemented as described by Sutton (1996), but here the task was modeled as a discounted problem with $\gamma = 0.99$. All the transitions were associated with a zero reward, except those leading to the goal, which resulted in a reward of $+5$, and those ending inside one of the puddles, which lead to a penalty of -10 times the distance to the puddle's nearest edge. If the agent did not reach the goal after 300 steps the episode was interrupted and considered as a failure. The algorithms were evaluated on two sets of states distributed over disjoint regions of the state space surrounding the puddles. The first set was a 3×3 grid defined over $[0.1, 0.3] \times [0.3, 0.5]$ and the second one was composed of four states: $\{0.1, 0.3\} \times \{0.9, 1.0\}$.

Pole Balancing: We implemented the simulator of the three versions of the pole-balancing task using the equations of motion and parameters given in the appendix of Gomez's (2003) PhD thesis. For the integration we used the 4^{th} order Runge-Kutta method with a time step of 0.01 seconds and actions chosen every 2 time steps. The problem was modeled as a discounted task with $\gamma = 0.99$. We considered the version of the task in which the angle between the pole and the vertical plane must be kept within $[-36^o, 36^o]$. In this formulation, an episode is interrupted and the agent gets a reward of -1 if the pole falls past a 36-degree angle or the cart reaches the boundaries of the track, located at 2.4m from its center. At all other steps the agent receives a reward of 0. In all versions of the problem an episode was considered a success if the pole(s) could be balanced for 3000 steps (one minute of simulated time). The test set was comprised of 81 states equally spaced in the region defined by $\pm[1.2\text{m}, 1.2/5\text{m}, 18^o, 75^o/s]$, for the single pole case, and by $\pm[1.2\text{m}, 1.2/5\text{m}, 18^o, 75^o/s, 18^o, 150^o/s]$ for the double-pole version of the problem. These values correspond to a hypercube centered at the origin and covering 50% of the state-space axes in each dimension (since the velocity of the cart and the angular velocity of the poles are theoretically not bounded, we defined the limits of these variables based on samples generated in simple preliminary experiments). For the triple pole-balancing task we performed our simulations using the parameters usually adopted with the two pole version of the problem, but we added a third pole with the same length and mass as the longer pole (Gomez, 2003). In this case the decision policies were evaluated on a test set containing 256 states equally distributed in the region $\pm[1.2\text{m}, 1.2/5\text{m}, 18^o, 75^o/s, 18^o, 150^o/s, 18^o, 75^o/s]$.

HIV drug schedule: The HIV drug schedule task was implemented using the system of ordinary differential equations (ODEs) given by Adams et al. (2004). Integration was carried out by the Euler method using a step size of 0.001 with actions selected at each 5000 steps (corresponding to 5 days of simulated time). As suggested by Ernst et al. (2006), the problem was modeled as a discounted task with $\gamma = 0.98$. All other parameters of the task, as well as the protocol used for the numerical simulations, also followed the suggestions of the same authors. In particular, we assumed the existence of 30 patients who were monitored for 1000 days. During the monitoring period, the content of the drug cocktail administered to each patient could be changed at fixed intervals of 5 days. Thus, in a sample transition $(s_i^a, r_i^a, \hat{s}_i^a)$: s_i^a is the initial patient condition, a is one of the four types of cocktails to be administered for the next 5 days, $\hat{s}_i^a$ is the patient condition 5 days later, and r_i^a is a reward computed based on the amount of drug in the selected cocktail a and on the difference between the patient's condition from s_i^a to $\hat{s}_i^a$ (Ernst et al., 2006). The results reported in Section 4.2.3 correspond to the performance of the greedy policy induced by the value function computed by the algorithms using all available sample transitions. The decision policies (in this case STI treatments) were evaluated for 5000 days starting from an "unhealthy" state corresponding to a basin of attraction of the ODEs describing the problem's dynamics (see the papers by Adams et al. and Ernst et al.).

Epilepsy suppression: We used a generative model developed by Bush et al. (2009) to perform our experiments with the epilepsy suppression task. The model was generated based on labeled field potential recordings of five rat brain slices electrically stimulated at frequencies of 0.0 Hz, 0.5 Hz, 1.0 Hz, and 2.0 Hz. The data was used to construct a manifold embedding which in turn gave rise to the problem's state space. The objective is to minimize the occurrence of seizures using as little stimulation as possible, therefore there is a negative reward associated with both events (see Section 4.2.4). Bush et al.'s generative model is public available as an environment for the RL-Glue package (Tanner and White, 2009). In our experiments the problem was modeled as a discounted task with $\gamma = 0.99$. The decision policies were evaluated on episodes of 10^5 transitions starting from a fixed set of 10 test states drawn uniformly at random from the problem's state space.

Helicopter hovering: In the experiments with the helicopter hovering task we used the simulator developed by Abbeel et al. (2005), which is available as an environment for the RL-Glue package (Tanner and White, 2009). The simulator was built based on data collected from two separate flights of an XCell Tempest helicopter. The data was used to adjust the parameters of an "acceleration prediction model", which is more accurate than the linear model normally adopted by industry. The objective in the problem is to keep the helicopter hovering as close as possible to a specific location. Therefore, at each time step the agent gets a negative reward proportional to the distance from the target position. Since the problem's original action space is $A \equiv [-1, 1]^4$, we discretized each dimension using 4 break points distributed unevenly over $[-1, 1]$. We tried several possible discretizations and picked the one which resulted in the best performance of the SARSA

agent (see Section 5.2.3). After this process, the problem's action space was redefined as $A \equiv \{-0.25, -0.05, +0.05, +0.25\}^4$. The problem was modeled as a discounted task with $\gamma = 0.99$. The decision policies were evaluated in episodes starting from the target position and ending when the helicopter crashed.

B.2 Algorithms

In all experiments, we used

$$\phi(z) \equiv \bar{\phi}(z) \equiv \exp(-z) \tag{66}$$

to define the kernels used by KBRL, LSPI, and KBSF. In the experiments involving a large number of sample transitions we used sparse kernels, that is, we only computed the μ largest values of $\mathrm{k}_\tau(\bar{s}_i, \cdot)$ and the $\bar{\mu}$ largest values of $\bar{\mathrm{k}}_{\bar{\tau}}(\hat{s}_i^a, \cdot)$. In order to implement this feature, we used a KD-tree to find the μ ($\bar{\mu}$) nearest neighbors of $\bar{s}_i$ ($\hat{s}_i^a$) and only computed k_τ ($\bar{\mathrm{k}}_{\bar{\tau}}$) in these states (Bentley, 1975). The value of k_τ and $\bar{\mathrm{k}}_{\bar{\tau}}$ outside this neighborhood was truncated to zero (we used specialized data structures to avoid storing those).

We now list a few details regarding the algorithms's implementations which were not described in the paper:

- **KBRL** and **KBSF**: We used modified policy iteration to compute $\hat{\mathbf{Q}}^*$ (Puterman and Shin, 1978). The value function of a fixed policy π was approximated through value iteration using the stop criterion described by Puterman (1994, Proposition 6.6.5) with $\varepsilon = 10^{-6}$. Table 1 shows the parameters's values used by KBSF across the experiments.

- **LSPI**: As explained above, LSPI used the kernel derived from (66) as its basis function. Following Lagoudakis and Parr (2003), we adopted one block of basis functions for each action $a \in A$. Singular value decomposition was used to avoid eventual numerical instabilities in the system of linear equations constructed at each iteration of LSPI (Golub and Loan, 1993).

- **Fitted Q-iteration and extra trees**: FQIT has four main parameters: the number of iterations, the number of trees composing the ensemble, the number of candidate cut-points evaluated during the generation of the trees, and the minimum number of elements required to split a node, denoted here $\eta_{\min}$. In general, increasing the first three improves performance, while $\eta_{\min}$ has an inverse relation with the quality of the final value function approximation. Our experiments indicate that the following configuration of FQIT usually results in good performance on the tasks considered in this paper: 50 iterations (with the structure of the trees fixed after the 10^{th} one), an ensemble of 30 trees, and $d_{\mathbb{S}}$ candidate cut points, where $d_{\mathbb{S}}$ is the dimension of the state space $\mathbb{S}$. The parameter $\eta_{\min}$ has a particularly strong effect on FQIT's performance and computational cost, and its correct value seems to be more problem-dependent. Therefore, in all of our experiments we fixed the parameters of FQIT as described above and only varied $\eta_{\min}$.

- **SARSA**: We adopted the implementation of SARSA(λ) available in the RL-Glue package (Tanner and White, 2009). The algorithm uses gradient descent temporal-difference learning to configure a tile coding function approximator.

Problem	Section	$\bar{s}_i$	m	τ	$\bar{\tau}$	μ	$\bar{\mu}$
Puddle	4.2.1	k-means	$\{10, 30, ..., 150\}$	$\{0.01, 0.1, 0.1\}$	$\{0.01, 0.1, 0.1\}$	∞	∞
Puddle	5.2.1	evenly	100	$\{0.01, 0.1, 0.1\}$	$\{0.01, 0.1, 0.1\}$	∞	∞
Single Pole	4.2.2	k-means	$\{10, 30, ..., 150\}$	1	$\{0.01, 0.1, 0.1\}$	∞	∞
Double Pole	4.2.2	k-means	$\{20, 40, ..., 200\}$	1	$\{0.01, 0.1, 0.1\}$	∞	∞
Triple Pole	5.2.2	on-line	on-line	100^*	1^*	50^*	10^*
HIV	4.2.3	random	$\{2000, 4000, ..., 10000\}$	1	1	2^*	3^*
Epilepsy	4.2.4	k-means	50000^*	1	$\{0.01, 0.1, 0.1\}$	6^*	6^*
Helicopter	5.2.3	k-means	500^*	1	1	4^*	4^*

Table 1: Parameters used by KBSF on the computational experiments. The values marked with an asterisk (*) were determined by trial and error on preliminary tests. The remaining parameters were kept fixed from the start or were defined based on a very coarse search.

Appendix C. Table of Symbols

Table 2 shows the main symbols used in the paper. Auxiliary symbols and functions whose use is restricted to a specific part of the text are not listed in the table.

References

Pieter Abbeel, Varun Ganapathi, and Andrew Y. Ng. Learning vehicular dynamics, with application to modeling helicopters. In *Advances in Neural Information Processing Systems (NIPS)*, 2005.

Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

Brian M. Adams, Harvey T. Banks, Hee-Dae Kwon, and Hien T. Tran. Dynamic multidrug therapies for HIV: optimal and STI control approaches. *Mathematical Biosciences and Engineering*, 1(2):223–41, 2004.

Charles W. Anderson. *Learning and Problem Solving with Multilayer Connectionist Systems*. PhD thesis, Computer and Information Science, University of Massachusetts, 1986.

András Antos, Rémi Munos, and Csaba Szepesvári. Fitted Q-iteration in continuous action-space MDPs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 9–16, 2007.

Anwarissa Asbah, André M. S. Barreto, Clement Gehring, Joelle Pineau, and Doina Precup. Reinforcement learning competition 2013: Controllability and kernel-based stochastic factorization. In *Proceedings of the ICML Workshop on the Reinforcement Learning Competition*, 2013.

Christopher G. Atkeson and Juan Carlos Santamaria. A comparison of direct and model-based reinforcement learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3557–3564, 1997.

Symbol	Meaning	Defined in Sec.
γ	Discount factor in $[0, 1)$	2
A, a	Action space, generic action	2
S	Generic state space	2
$\mathbb{S}$, $d_{\mathbb{S}}$	Continuous state space, dimension of $\mathbb{S}$	2.2
S^a	Sample transitions associated with action a, $S^a \equiv \{(s_i^a, r_i^a, \hat{s}_i^a) \vert i = 1, 2, ..., n_a\}$	2.2
s, s_i, $s_{(t)}$, s_i^a, $\hat{s}_i^a$	Generic state, i^{th} state of an MDP, state occupied at time t, i^{th} tuple in S^a	2
r, $r_{(t)}$, r_i^a	Generic reward, reward received at time t, i^{th} sampled reward in S^a	2
n_a, n, $\hat{n}$	Number of sample transitions in S^a, $n = \sum_a n_a$, and $\hat{n} = \max_a n_a$	2.2
P^a, P^π, $\mathbf{P}^a$, $\mathbf{P}^\pi$	Transition function associated with a or π and their matrix counterparts	2.1
R^a, R^π, $\mathbf{r}^a$, $\mathbf{r}^\pi$	Reward function associated with a or π and corresponding reward vectors	2.1
M	MDP, $M \equiv (S, A, P^a, R^a, \gamma)$	2.1
π, π^*	Generic decision policy, optimal decision policy in M	2
V^π, V^*, $\mathbf{v}^\pi$, $\mathbf{v}^*$	Value function of π, optimal value function of M, vector counterparts	2.1
Q^π, Q^*, $\mathbf{Q}^\pi$, $\mathbf{Q}^*$	Action-value functions of π and M and matrix counterparts	2.1
Γ	Operator: $\Gamma\mathbf{Q} = \mathbf{v} \iff v_i = \max_j q_{ij},\ \forall i$	2.1
Δ	Operator associated with M: $\Delta\mathbf{v} = \mathbf{Q} \iff q_{ia} = r_i^a + \gamma \sum_{j=1}^{\vert S\vert} p_{ij}^a v_j,\ \forall i, a$	2.1
T	Bellman operator of M, given by $T \equiv \Gamma\Delta$	2.1
ϕ, $\bar{\phi}$	Non-increasing functions in $\mathbb{R}^+ \mapsto \mathbb{R}^+$ used to construct the kernels	2.2, 4
k_τ, $\bar{\mathrm{k}}_{\bar{\tau}}$	Kernel functions: $\mathrm{k}_\tau(s, s') = \phi(\Vert s - s'\Vert/\tau)$ and $\bar{\mathrm{k}}_{\bar{\tau}}(s, s') = \bar{\phi}(\Vert s - s'\Vert/\bar{\tau})$	2.2, 4
τ, $\bar{\tau}$	Scalars defining the "widths" of k_τ and $\bar{\mathrm{k}}_{\bar{\tau}}$	2.2, 4
κ_τ^a, $\bar{\kappa}_{\bar{\tau}}$	Normalized kernels: $\kappa_\tau^a(s, s_i^a) = \frac{\mathrm{k}_\tau(s, s_i^a)}{\sum_{j=1}^{n_a} \mathrm{k}_\tau(s, s_j^a)}$ and $\bar{\kappa}_{\bar{\tau}}(s, \bar{s}_i) = \frac{\bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_i)}{\sum_{j=1}^{m} \bar{\mathrm{k}}_{\bar{\tau}}(s, \bar{s}_j)}$	2.2, 4
$\bar{S}$	Set of representative states, $\bar{S} \equiv \{\bar{s}_1, \bar{s}_2, ..., \bar{s}_m\}$	4
$\bar{s}_i$	i^{th} representative state	4
m	Number of representative states	4
$\mathbf{D}$, $\dot{\mathbf{D}}^a$	Stochastic matrix in $\mathbb{R}^{n \times m}$ and its $n_a \times m$ block associated with a	4
$\mathbf{K}^a$, $\dot{\mathbf{K}}^a$	Stochastic matrix in $\mathbb{R}^{m \times n}$ and its dense version in $\mathbb{R}^{m \times n_a}$	4
rs	Function: $rs(s, i) = \bar{s}_k \iff \bar{s}_k$ is the i^{th} closest representative state to s	4.1
$dist$	Function: $dist(s, i) = \Vert s - rs(s, i)\Vert$	4.1

Table 2: List of main symbols used throughout the paper. We use the same notation to refer to all MDPs and the associated elements, and resort to math accents to distinguish between them. So, for example, if $\bar{M}$ is an MDP, the associated elements are referred to as $\bar{\mathbf{P}}^a$, $\bar{\mathbf{r}}^a$, $\bar{\mathbf{v}}^*$, $\bar{\mathbf{Q}}^*$, $\bar{\pi}^*$, $\bar{\Delta}$, and $\bar{T}$.

Seema H. Bajaria, Glenn Webb, and Denise E. Kirschner. Predicting differential responses to structured treatment interruptions during HAART. *Bulletin of Mathematical Biology*, 66(5):1093 – 1118, 2004.

André M. S. Barreto. Tree-based on-line reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2417–2423, 2014.

André M. S. Barreto and Marcelo D. Fragoso. Computing the stationary distribution of a finite Markov chain through stochastic factorization. *SIAM Journal on Matrix Analysis and Applications*, 32:1513–1523, 2011.

André M. S. Barreto, Doina Precup, and Joelle Pineau. Reinforcement learning using kernel-based stochastic factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 720–728, 2011.

André M. S. Barreto, Doina Precup, and Joelle Pineau. On-line reinforcement learning using incremental kernel-based stochastic factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1484–1492, 2012.

André M. S. Barreto, Joelle Pineau, and Doina Precup. Policy iteration based on stochastic factorization. *Journal of Artificial Intelligence Research*, 50:763–803, 2014.

Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:834–846, 1983.

Richard E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

Richard E. Bellman. *Adaptive Control Processes*. Princeton University Press, 1961.

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

Alina Beygelzimer, Sham Kakade, and John Langford. Cover trees for nearest neighbor. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 97–104, 2006.

Nikhil Bhat, Ciamac Moallemi, and Vivek Farias. Non-parametric approximate dynamic programming via the kernel method. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Ronen I. Brafman and Moshe Tennenholtz. R-MAX - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2003.

Keith Bush, Joelle Pineau, and Massivo Avoli. Manifold embeddings for model-based reinforcement learning of neurostimulation policies. In *Proceedings of the ICML/UAI/COLT Workshop on Abstraction in Reinforcement Learning*, 2009.

Keith Bush and Joelle Pineau. Manifold embeddings for model-based reinforcement learning under partial observability. In *Advances in Neural Information Processing Systems (NIPS)*, pages 189–197, 2009.

Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 626–635, 1997.

Joel E. Cohen and Uriel G. Rothblum. Nonnegative ranks, decompositions and factorizations of nonnegative matrices. *Linear Algebra and its Applications*, 190:149–168, 1991.

Adele Cutler and Leo Breiman. Archetypal analysis. *Technometrics*, 36(4):338–347, 1994.

Eric V. Denardo. Contraction mappings in the theory underlying dynamic programming. *SIAM Review*, 9(2):165–177, 1967.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

Dominique M. Durand and Marom Bikson. Suppression and control of epileptiform activity by electrical stimulation: a review. *Proceedings of the IEEE*, 89(7):1065 –1082, 2001.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with Gaussian processes. In *Proceedings of the International Conference on Machine learning (ICML)*, pages 201–208, 2005.

Damien Ernst, Guy-Bart Stan, Jorge Gonçalves, and Louis Wehenkel. Clinical data based optimal STI strategies for HIV: a reinforcement learning approach. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 2006.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

Amir-massoud Farahmand. *Regularization in reinforcement learning*. PhD thesis, University of Alberta, 2011.

Amir-massoud Farahmand and Csaba Szepesvári. Model selection in reinforcement learning. *Machine Learning Journal*, 85(3):299–332, 2011.

Amir-massoud Farahmand, Csaba Szepesvári, and Jean-Yves Audibert. Toward manifold-adaptive learning. In *NIPS Workshop on Topology learning*, 2007.

Tomás Feder and Daniel Greene. Optimal algorithms for approximate clustering. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 434–444, 1988.

Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 36(1):3–42, 2006.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 2nd edition, 1993.

Faustino J. Gomez, Juergen Schmidhuber, and Risto Miikkulainen. Efficient non-linear control through neuroevolution. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 654–662, 2006.

Faustino J. Gomez. *Robust non-linear control through neuroevolution*. PhD thesis, The University of Texas at Austin, 2003. Technical Report AI-TR-03-303.

Teofilo Gonzales. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

Geoffrey Gordon. Stable function approximation in dynamic programming. Technical Report CMU-CS-95-103, Computer Science Department, Carnegie Mellon University, 1995.

Steffen Grunewalder, Guy Lever, Luca Baldassarre, Massi Pontil, and Arthur Gretton. Modelling transition dynamics in MDPs with RKHS embeddings. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 535–542, 2012.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer Verlag, 2002.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2002.

Kristin Jerger and Steven Schiff. Periodic pacing an in vitro epileptic focus. *Journal of Neurophysiology*, (2):876–879, 1995.

Nicholas Jong and Peter Stone. Kernel-based models for reinforcement learning in continuous state spaces. In *Proceedings of the International Conference on Machine Learning (ICML)—Workshop on Kernel Machines and Reinforcement Learning*, 2006.

Nicholas Jong and Peter Stone. Compositional models for reinforcement learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 644–659, 2009.

Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

Oliver B. Kroemer and Jan R. Peters. A non-parametric approach to dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1719–1727, 2011.

Branislav Kveton and Georgios Theocharous. Kernel-based reinforcement learning on representative states. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 124–131, 2012.

Branislav Kveton and Georgios Theocharous. Structured kernel-based reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2013.

Michail G. Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

Donald Michie and Roger Chambers. BOXES: An experiment on adaptive control. *Machine Intelligence 2*, pages 125–133, 1968.

Andrew W. Moore and Christopher G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13:103–130, 1993.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9:815–857, 2008.

Andrew Y. Ng, H. Jin Kim, Michael I. Jordan, and Shankar Sastry. Autonomous helicopter flight via reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

Dirk Ormoneit and Peter Glynn. Kernel-based reinforcement learning in average-cost problems. *IEEE Transactions on Automatic Control*, 47(10):1624–1636, 2002.

Dirk Ormoneit and Śaunak Sen. Kernel-based reinforcement learning. *Machine Learning*, 49 (2–3):161–178, 2002.

Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

Martin L. Puterman. *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

Martin L. Puterman and Moon C. Shin. Modified policy iteration algorithms for discounted Markov decision problems. *Management Science*, 24(11):1127–1137, 1978.

Carl Edward Rasmussen and Malte Kuss. Gaussian processes in reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 751–759, 2004.

Balaraman Ravindran. *An Algebraic Approach to Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts, 2004.

Gavin Rummery and Mahesan Niranjan. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Engineering Department, Cambridge University, 1994.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

Jonathan Sorg and Satinder Singh. Transfer via soft homomorphisms. In *Autonomous Agents & Multiagent Systems/Agent Theories, Architectures, and Languages*, pages 741–748, 2009.

Charles J. Stone. Optimal global rates of convergence for nonparametric regression. *The Annals of Statistics*, 10:1040–1053, 1982.

Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Richard S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1038–1044, 1996.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

Brian Tanner and Adam M. White. RL-Glue: Language-independent software for reinforcement-learning experiments. *Journal of Machine Learning Research*, 10:2133–2136, 2009.

Gavin Taylor and Ronald Parr. Kernelized value function approximation for reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1017–1024, 2009. .

Stephen A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20:1364–1377, 2009.

Ward Whitt. Approximations of dynamic programs, I. *Mathematics of Operations Research*, 3(3):231–243, 1978.

Alexis P. Wieland. Evolving neural network controllers for unstable systems. In *Proceedings of the International Joint Conference on Neural Networks*, pages 667–673, 1991.

Xin Xu, Tao Xie, Dewen Hu, and Xicheng Lu. Kernel Least-Squares Temporal Difference Learning. *Information Technology*, pages 54–63, 2005.

Yinyu Ye. The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate. *Mathematics of Operations Research*, 36(4): 593–603, 2011.

Journal of Machine Learning Research 17 (2016) 1-30 Submitted 2/14; Revised 8/15; Published 4/16

An Information-Theoretic Analysis of Thompson Sampling

Daniel Russo DJRUSSO@STANFORD.EDU
Department of Management Science and Engineering
Stanford University
Stanford, California 94305

Benjamin Van Roy BVR@STANFORD.EDU
Departments of Management Science and Engineering and Electrical Engineering
Stanford University
Stanford, California 94305

Editor: Kevin Murphy

Abstract

We provide an information-theoretic analysis of Thompson sampling that applies across a broad range of online optimization problems in which a decision-maker must learn from partial feedback. This analysis inherits the simplicity and elegance of information theory and leads to regret bounds that scale with the entropy of the optimal-action distribution. This strengthens preexisting results and yields new insight into how information improves performance.

Keywords: Thompson sampling, online optimization, mutli–armed bandit, information theory, regret bounds

1. Introduction

This paper considers the problem of repeated decision making in the presence of model uncertainty. A decision-maker repeatedly chooses among a set of possible actions, observes an outcome, and receives a reward representing the utility derived from this outcome. The decision-maker is uncertain about the underlying system and is therefore initially unsure of which action is best. However, as outcomes are observed, she is able to learn over time to make increasingly effective decisions. Her objective is to choose actions sequentially so as to maximize the expected cumulative reward.

We focus on settings with *partial feedback*, under which the decision-maker does not generally observe what the reward would have been had she selected a different action. This feedback structure leads to an inherent tradeoff between *exploration and exploitation*: by experimenting with poorly understood actions one can learn to make more effective decisions in the future, but focusing on better understood actions may lead to higher rewards in the short term. The classical multi–armed bandit problem is an important special case of this formulation. In such problems, the decision–maker only observes rewards she receives, and rewards from one action provide no information about the reward that can be attained by selecting other actions. We are interested here primarily in models and algorithms that accommodate cases where the number of actions is very large and there is a richer

information structure relating actions and observations. This category of problems is often referred to as *online optimization with partial feedback*.

A large and growing literature treats the design and analysis of algorithms for such problems. An online optimization algorithm typically starts with two forms of prior knowledge. The first – *hard knowledge* – posits that the mapping from action to outcome distribution lies within a particular family of mappings. The second – *soft knowledge* – concerns which of these mappings are more or less likely to match reality. Soft knowledge evolves with observations and is typically represented in terms of a probability distribution or a confidence set.

Much recent work concerning online optimization algorithms focuses on establishing performance guarantees in the form of *regret bounds*. Surprisingly, virtually all of these regret bounds depend on hard knowledge but not soft knowledge, a notable exception being the bounds of Srinivas et al. (2012) which we discuss further in Section 1.2. Regret bounds that depend on hard knowledge yield insight into how an algorithm's performance scales with the complexity of the family of mappings and are useful for delineating algorithms on that basis, but if a regret bound does not depend on soft knowledge, it does not have much to say about how future performance should improve as data is collected. The latter sort of insight should be valuable for designing better ways of trading off between exploration and exploitation, since it is soft knowledge that is refined as a decision-maker learns. Another important benefit to understanding how performance depends on soft knowledge arises in practical applications: when designing an online learning algorithm, one may have access to historical data and want to understand how this prior information benefits future performance.

In this paper, we establish regret bounds that depend on both hard and soft knowledge for a simple online learning algorithm alternately known as *Thompson sampling*, *posterior sampling*, or *probability matching*. The bounds strengthen results from prior work not only with respect to Thompson sampling but relative to regret bounds for *any* online optimization algorithm. Further, the bounds offer new insight into how regret depends on soft knowledge. Indeed, forthcoming work of ours leverages this to produce an algorithm that outperforms Thompson sampling.

We found information theory to provide tools ideally suited for deriving our new regret bounds, and our analysis inherits the simplicity and elegance enjoyed by work in that field. Our formulation encompasses a broad family of information structures, including as special cases multi–armed bandit problems with independent arms, online optimization problems with full information, linear bandit problems, and problems with combinatorial action sets and "semi–bandit" feedback. We leverage information theory to provide a unified analysis that applies to each of those special cases.

A novel feature of our bounds is their dependence on the entropy of the optimal-action distribution. To our knowledge, these are the first bounds on the expected regret of *any* algorithm that depend on the magnitude of the agent's uncertainty about which action is optimal. The fact that our bounds only depend on uncertainties relevant to optimizing performance highlights the manner in which Thompson sampling naturally exploits complex information structures. Further, in practical contexts, a decision-maker may begin with an understanding that some actions are more likely to be optimal than others. For example, when dealing with a shortest path problem, one might expect paths that traverse fewer

edges to generally incur less cost. Our bounds are the first to formalize the performance benefits afforded by such an understanding.

1.1 Preview of Results

Our analysis is based on a general probabilistic, or Bayesian, formulation in which uncertain quantities are modeled as random variables. In principle, the optimal strategy for such a problem could be calculated via dynamic programing, but for problem classes of practical interest this would be computationally intractable. *Thompson sampling* serves as a simple and elegant heuristic strategy. In this section, we provide a somewhat informal problem statement, and a preview of our main results about Thompson sampling. In the next subsection we discuss how these results relate to the existing literature.

A decision maker repeatedly chooses among a finite set of possible actions $\mathcal{A}$ and upon taking action $a \in \mathcal{A}$ at time $t \in \mathbb{N}$ she observes a random outcome $Y_{t,a} \in \mathcal{Y}$. She associates a reward with each outcome as specified by a reward function $R : \mathcal{Y} \to \mathbb{R}$. The outcomes $(Y_{t,a})_{t\in\mathbb{N}}$ are drawn independently over time from a fixed probability distribution p_a^* over $\mathcal{Y}$. The decision maker is uncertain about the distribution of outcomes $p^* = (p_a^*)_{a\in\mathcal{A}}$, which is itself distributed according to a prior distribution over a family $\mathcal{P}$ of such distributions. However, she is able to learn about p^* as the outcomes of past decisions are observed, and this allows her to learn over time to attain improved performance.

We first present a special case of our main result that applies to online linear optimization problems under bandit feedback. This result applies when each action is associated with a d–dimensional feature vector and the mean reward of each action is the inner product between an unknown parameter vector and the action's known feature vector. More precisely, suppose $\mathcal{A} \subset \mathbb{R}^d$ and that for every $p \in \mathcal{P}$ there is a vector $\theta_p \in \mathbb{R}^d$ such that

$$\mathop{\mathbb{E}}_{y\sim p_a}[R(y)] = a^T\theta_p \tag{1}$$

for all $a \in \mathcal{A}$. When an action is sampled, a random reward in $[0, 1]$ is received, where the mean reward is given by (1). Then, our analysis establishes that the expected cumulative regret of Thompson sampling up to time T is bounded by

$$\sqrt{\frac{\text{Entropy}(A^*)dT}{2}}, \tag{2}$$

where $A^* \in \arg\max_{a\in\mathcal{A}} \mathbb{E}[R(Y_{t,a})|p^*]$ denotes the optimal action. This bound depends on the time horizon, the entropy of the of the prior distribution of the optimal action A^*, and the dimension d of the linear model.

Because the entropy of A^* is always less than $\log|\mathcal{A}|$, (2) yields a bound of order $\sqrt{\log(|\mathcal{A}|)dT}$, and this scaling cannot be improved in general (see Section 6.5). The bound (2) is stronger than this worst–case bound, since the entropy of A^* can be much smaller than $\log(|\mathcal{A}|)$.

Thompson sampling incorporates prior knowledge in a flexible and coherent way, and the benefits of this are reflected in two distinct ways by the above bound. First, as in past work (see e.g. Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010), the bound depends on the dimension of the linear model instead of the number of actions. This reflects that the

algorithm is able to learn more rapidly by exploiting the known model, since observations from selecting one action provide information about rewards that would have been generated by other actions. Second, the bound depends on the entropy of the prior distribution of A^* instead of a worst case measure like the logarithm of the number of actions. This highlights the benefit of prior knowledge that some actions are more likely to be optimal than others. In particular, this bound exhibits the natural property that as the entropy of the prior distribution of A^* goes to zero, expected regret does as well.

Our main result extends beyond the class of linear bandit problems. Instead of depending on the linear dimension of the model, it depends on a more general measure of the problem's information complexity: what we call the problem's *information ratio*. By bounding the information ratio in specific settings, we recover the bound (2) as a special case, along with bounds for problems with full feedback and problems with combinatorial action sets and "semi–bandit" feedback.

1.2 Related Work

Though Thompson sampling was first proposed in 1933 (Thompson, 1933), until recently it was largely ignored in the academic literature. Interest in the algorithm grew after empirical studies (Scott, 2010; Chapelle and Li, 2011) demonstrated performance exceeding state of the art. Over the past several years, it has also been adopted in industry.[1] This has prompted a surge of interest in providing theoretical guarantees for Thompson sampling.

One of the first theoretical guarantees for Thompson sampling was provided by May et al. (2012), but they showed only that the algorithm converges asymptotically to optimality. Agrawal and Goyal (2012); Kauffmann et al. (2012); Agrawal and Goyal (2013a) and Korda et al. (2013) studied on the classical multi-armed bandit problem, where sampling one action provides no information about other actions. They provided frequentist regret bounds for Thompson sampling that are asymptotically optimal in the sense defined by Lai and Robbins (1985). To attain these bounds, the authors fixed a specific uninformative prior distribution, and studied the algorithm's performance assuming this prior is used.

Our interest in Thompson sampling is motivated by its ability to incorporate rich forms of prior knowledge about the actions and the relationship among them. Accordingly, we study the algorithm in a very general framework, allowing for an *arbitrary* prior distribution over the true outcome distributions $p^* = (p_a^*)_{a\in\mathcal{A}}$. To accommodate this level of generality while still focusing on finite–time performance, we study the algorithm's *expected regret* under the prior distribution. This measure is sometimes called *Bayes risk* or *Bayesian regret*.

Our recent work (Russo and Van Roy, 2014) provided the first results in this setting. That work leverages a close connection between Thompson sampling and upper confidence bound (UCB) algorithms, as well as existing analyses of several UCB algorithms. This confidence bound analysis was then extended to a more general setting, leading to a general regret bound stated in terms of a new notion of model complexity – what we call the eluder dimension. While the connection with UCB algorithms may be of independent interest,

1. Microsoft (Graepel et al., 2010), Google analytics (Scott, 2014) and Linkedin (Tang et al., 2013) have used Thompson sampling.

it's desirable to have a simple, self–contained, analysis that does not rely on the–often complicated–construction of confidence bounds.

Agrawal and Goyal (2013a) provided the first "distribution independent" bound for Thompson sampling. They showed that when Thompson sampling is executed with an independent uniform prior and rewards are binary the algorithm satisfies a frequentist regret bound[2] of order $\sqrt{|\mathcal{A}|T\log(T)}$. Russo and Van Roy (2014) showed that, for an arbitrary prior over bounded reward distributions, the expected regret of Thompson sampling under this prior is bounded by a term of order $\sqrt{|\mathcal{A}|T\log(T)}$. Bubeck and Liu (2013) showed that this second bound can be improved to one of order $\sqrt{|\mathcal{A}|T}$ using more sophisticated confidence bound analysis, and also studied a problem setting where the regret of Thompson sampling is bounded uniformly over time. In this paper, we are interested mostly in results that replace the explicit $\sqrt{|\mathcal{A}|}$ dependence on the number of actions with a more general measure of the problem's information complexity. For example, as discussed in the last section, for the problem of linear optimization under bandit feedback one can provide bounds that depend on the dimension of the linear model instead of the number of actions.

To our knowledge, Agrawal and Goyal (2013b) are the only other authors to have studied the application of Thompson sampling to linear bandit problems. They showed that, when the algorithm is applied with an uncorrelated Gaussian prior over θ_{p^*}, it satisfies frequentist regret bounds of order $\frac{d^2}{\epsilon}\sqrt{T^{1+\epsilon}}(\log(Td)$. Here ϵ is a parameter used by the algorithm to control how quickly the posterior concentrates. Russo and Van Roy (2014) allowed for an arbitrary prior distribution, and provided a bound on expected regret (under this prior) of order $d\log(T)\sqrt{T}$. Unlike the bound (2), these results hold whenever $\mathcal{A}$ is a compact subset of $\mathbb{R}^d$, but we show in Appendix D.1 that through discretization argument the bound (2) also yields a similar bound whenever $\mathcal{A}$ is compact. In the worst case, that bound is of order $d\sqrt{T\log(T)}$.

Other very recent work (Gopalan et al., 2014) provided a general asymptotic guarantee for Thompson sampling. They studied the growth rate of the cumulative number of times a suboptimal action is chosen as the time horizon T tends to infinity. One of their asymptotic results applies to the problem of online linear optimization under "semi–bandit" feedback, which we study in Section 6.6.

An important aspect of our regret bound is its dependence on soft knowledge through the entropy of the optimal-action distribution. One of the only other regret bounds that depends on soft–knowledge was provided very recently by Li (2013). Inspired by a connection between Thompson sampling and exponential weighting schemes, that paper introduced a family of Thompson sampling like algorithms and studied their application to contextual bandit problems. While our analysis does not currently treat contextual bandit problems, we improve upon their regret bound in several other respects. First, their bound depends on the entropy of the prior distribution of mean rewards, which is never smaller, and can be much larger, than the entropy of the distribution of the optimal action. In addition, their bound has an order $T^{2/3}$ dependence on the problem's time horizon, and, in order to guarantee each action is explored sufficiently often, requires that actions are frequently selected uniformly at random. In contrast, our focus is on settings where the number of actions is large and the goal is to learning without sampling each one.

2. They bound regret conditional on the true reward distribution: $\mathbb{E}\left[\text{Regret}(T,\pi^{\text{TS}})|p^*\right]$.

Another regret bound that to some extent captures dependence on soft knowledge is that of Srinivas et al. (2012). This excellent work focuses on extending algorithms and expanding theory to address multi-armed bandit problems with arbitrary reward functions and possibly an infinite number of actions. In a sense, there is no hard knowledge while soft knowledge is represented in terms of a Gaussian process over a possibly infinite dimensional family of functions. An upper-confidence-bound algorithm is proposed and analyzed. Our earlier work (Russo and Van Roy, 2014) showed similar bounds also apply to Thompson sampling. Though our results here do not treat infinite action spaces, it should be possible to extend the analysis in that direction. One substantive difference is that our results apply to a much broader class of models: distributions are not restricted to Gaussian and more complex information structures are allowed. Further, the results of Srinivas et al. (2012) do not capture the benefits of soft knowledge to the extent that ours do. For example, their regret bounds do not depend on the mean of the reward function distribution, even though mean rewards heavily influence the chances that each action is optimal. Our regret bounds, in contrast, establish that regret vanishes as the probability that a particular action is optimal grows.

Our review has discussed the recent literature on Thompson sampling as well as two papers that have established regret bounds that depend on soft knowledge. There is of course an immense body of work on alternative approaches to efficient exploration, including work on the Gittins index approach (Gittins et al., 2011), Knowledge Gradient strategies (Ryzhov et al., 2012), and upper-confidence bound strategies (Lai and Robbins, 1985; Auer et al., 2002). In an adversarial framework, authors often study exponential-weighting shemes or, more generally, strategies based on online stochastic mirror descent. Bubeck and Cesa-Bianchi (2012) provided a general review of upper–confidence strategies and algorithms for the adversarial multi-armed bandit problem.

2. Problem Formulation

The decision–maker sequentially chooses actions $(A_t)_{t\in\mathbb{N}}$ from the action set $\mathcal{A}$ and observes the corresponding outcomes $(Y_{t,A_t})_{t\in\mathbb{N}}$. To avoid measure-theoretic subtleties, we assume the space of possible outcomes $\mathcal{Y}$ is a subset of a finite dimensional Euclidean space. There is a random outcome $Y_{t,a}\in\mathcal{Y}$ associated with each $a\in\mathcal{A}$ and time $t\in\mathbb{N}$. Let $Y_t\equiv(Y_{t,a})_{a\in\mathcal{A}}$ be the vector of outcomes at time $t\in\mathbb{N}$. The "true outcome distribution" p^* is a distribution over $\mathcal{Y}^{|\mathcal{A}|}$ that is itself randomly drawn from the family of distributions $\mathcal{P}$. We assume that, conditioned on p^*, $(Y_t)_{t\in\mathbb{N}}$ is an iid sequence with each element Y_t distributed according to p^*. Let p^*_a be the marginal distribution corresponding to $Y_{t,a}$.

The agent associates a reward $R(y)$ with each outcome $y\in\mathcal{Y}$, where the reward function $R:\mathcal{Y}\to\mathbb{R}$ is fixed and known. Uncertainty about p^* induces uncertainty about the true optimal action, which we denote by $A^*\in\arg\max_{a\in\mathcal{A}}\mathbb{E}\left[R(Y_{t,a})|p^*\right]$. The T period *regret* of the sequence of actions $A_1,..,A_T$ is the random variable,

$$\text{Regret}(T):=\sum_{t=1}^{T}\left[R(Y_{t,A^*})-R(Y_{t,A_t})\right], \tag{3}$$

which measures the cumulative difference between the reward earned by an algorithm that always chooses the optimal action, and actual accumulated reward up to time T. In this paper we study expected regret

$$\mathbb{E}\left[\text{Regret}(T)\right] = \mathbb{E}\left[\sum_{t=1}^{T}\left[R(Y_{t,A^*}) - R(Y_{t,A_t})\right]\right], \tag{4}$$

where the expectation is taken over the randomness in the actions A_t and the outcomes Y_t, and over the prior distribution over p^*. This measure of performance is commonly called *Bayesian regret* or *Bayes risk*.

Filtrations and randomized policies: We define all random variables with respect to a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Actions are chosen based on the history of past observations and possibly some external source of randomness. To represent this external source of randomness more formally, we introduce a sequence of random variables $(U_t)_{t\in\mathbb{N}}$, where for each $i \in \mathbb{N}$, U_i is jointly independent of $\{U_t\}_{t\neq i}$, the outcomes $\{Y_{t,a}\}_{t\in\mathbb{N},a\in\mathcal{A}}$, and p^*. We fix the filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$ where $\mathcal{F}_t \subset \mathcal{F}$ is the sigma–algebra generated by $\left(A_1, Y_{1,A_1}, ..., A_{t-1}, Y_{t-1,A_{t-1}}\right)$. The action A_t is measurable with respect to the sigma–algebra generated by $(\mathcal{F}_t, U_t)$. That is, given the history of past observations, A_t is random only through its dependence on U_t.

The objective is to choose actions in a manner that minimizes expected regret. For this purpose, it's useful to think of the actions as being chosen by a *randomized policy* π, which is an $\mathcal{F}_t$–adapted sequence $(\pi_t)_{t\in\mathbb{N}}$. An action is chosen at time t by randomizing according to $\pi_t(\cdot) = \mathbb{P}(A_t \in \cdot|\mathcal{F}_t)$, which specifies a probability distribution over $\mathcal{A}$. We explicitly display the dependence of regret on the policy π, letting $\mathbb{E}\left[\text{Regret}(T, \pi)\right]$ denote the expected value given by (4) when the actions $(A_1, .., A_T)$ are chosen according to π.

Further Assumptions: To simplify the exposition, our main results will be stated under two further assumptions. The first requires that rewards are uniformly bounded, effectively controlling the worst-case variance of the reward distribution. In particular, this assumption is used only in proving Fact 9. In the technical appendix, we show that Fact 9 can be extended to the case where reward distributions are sub-Gaussian, which yields results in that more general setting.

Assumption 1 $\sup_{\overline{y}\in\mathcal{Y}} R(\overline{y}) - \inf_{\underline{y}\in\mathcal{Y}} R(\underline{y}) \leq 1.$

Our next assumption requires that the action set is finite. In the technical appendix we show that some cases where $\mathcal{A}$ is infinite can be addressed through discretization arguments.

Assumption 2 *$\mathcal{A}$ is finite.*

Because the Thompson sampling algorithm only chooses actions from the support of A^*, all of our results hold when the finite set $\mathcal{A}$ denotes only the actions in the support of A^*. This difference can be meaningful. For example, when $\mathcal{A}$ is a polytope and the objective function is linear, the support of A^* contains only extreme points of the polytope: a finite set.

3. Basic Measures and Relations in Information Theory

Before proceeding, we will define several common information measures – entropy, mutual information, and Kullback-Leibler divergence — and state several facts about these measures that will be referenced in our analysis. When all random variables are discrete, each of these facts is shown in chapter 2 of Cover and Thomas (2012). A treatment that applies to general random variables is provided in chapter 5 of Gray (2011).

Before proceeding, we will define some simple shorthand notation. Let $P(X) = \mathbb{P}(X \in \cdot)$ denote the distribution function of random variable X. Similarly, define $P(X|Y) = \mathbb{P}(X \in \cdot|Y)$ and $P(X|Y = y) = \mathbb{P}(X \in \cdot|Y = y)$.

Throughout this section, we will fix random variables X, Y, and Z that are defined on a joint probability space. We will allow Y and Z to be general random variables, but will restrict X to be supported on a finite set $\mathcal{X}$. This is sufficient for our purposes, as we will typically apply these relations when X is A^*, and is useful, in part, because the entropy of a general random variable can be infinite.

The *Shannon entropy* of X is defined as

$$H(X) = -\sum_{x\in\mathcal{X}} \mathbb{P}(X = x)\log\mathbb{P}(X = x).$$

The first fact establishes uniform bounds on the entropy of a probability distribution.

Fact 1 $0 \leq H(X) \leq \log(|\mathcal{X}|)$.

If Y is a discrete random variable, the entropy of X conditional on $Y = y$ is

$$H(X|Y = y) = \sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x|Y = y\right)\log\mathbb{P}(X = x|Y = y)$$

and the conditional entropy is of X given Y is

$$H(X|Y) = \sum_{y} H(X|Y = y)\mathbb{P}(Y = y).$$

For a general random variable Y, the conditional entropy of X given Y is,

$$H(X|Y) = \mathbb{E}_Y\left[-\sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x|Y\right)\log\mathbb{P}(X = x|Y)\right],$$

where this expectation is taken over the marginal distribution of Y. For two probability measures P and Q, if P is absolutely continuous with respect to Q, the *Kullback–Leibler divergence* between them is

$$D(P||Q) = \int \log\left(\frac{dP}{dQ}\right) dP \tag{5}$$

where $\frac{dP}{dQ}$ is the Radon–Nikodym derivative of P with respect to Q. This is the expected value under P of the log-likelihood ratio between P and Q, and is a measure of how different P and Q are. The next fact establishes the non-negativity of Kullback–Leibler divergence.

Fact 2 *(Gibbs' inequality) For any probability distributions P and Q such that P is absolutely continuous with respect to Q, $D(P||Q) \geq 0$ with equality if and only if $P = Q$ P–almost everywhere.*

The *mutual information* between X and Y

$$I(X;Y) = D\left(P(X,Y) \,||\, P(X)\, P(Y)\right) \tag{6}$$

is the Kullback–Leibler divergence between the joint distribution of X and Y and the product of the marginal distributions. From the definition, it's clear that $I(X;Y) = I(Y;X)$, and Gibbs' inequality implies that $I(X;Y) \geq 0$ and $I(X;Y) = 0$ when X and Y are independent.

The next fact, which is Lemma 5.5.6 of Gray (2011), states that the mutual information between X and Y is the expected reduction in the entropy of the posterior distribution of X due to observing Y.

Fact 3 *(Entropy reduction form of mutual information)*

$$I(X;Y) = H(X) - H(X|Y)$$

The mutual information between X and Y, conditional on a third random variable Z is

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z),$$

the expected additional reduction in entropy due to observing Y given that Z is also observed. This definition is also a natural generalization of the one given in (6), since

$$I(X;Y|Z) = \mathbb{E}_Z\left[D\left(P\left((X,Y)|Z\right) \,||\, P(X|Z)\, P(Y|Z)\right)\right].$$

The next fact shows that conditioning on a random variable Z that is independent of X and Y does not affect mutual information.

Fact 4 *If Z is jointly independent of X and Y, then $I(X;Y|Z) = I(X;Y)$.*

The mutual information between a random variable X and a collection of random variables $(Z_1, ..., Z_T)$ can be expressed elegantly using the following "chain rule."

Fact 5 *(Chain Rule of Mutual Information)*

$$I(X;(Z_1, ...Z_T)) = I(X;Z_1) + I(X;Z_2|\, Z_1) + ... + I(X;Z_T|\, Z_1, ..., Z_{T-1}).$$

We provide some details related to the derivation of Fact 6 in the appendix.

Fact 6 *(KL divergence form of mutual information)*

$$\begin{aligned} I(X;Y) &= \mathbb{E}_X\left[D\left(P(Y|X) \,||\, P(Y)\right)\right] \\ &= \sum_{x \in \mathcal{X}} \mathbb{P}(X=x) D\left(P(Y|X=x) \,||\, P(Y)\right) \end{aligned}$$

While Facts 1 and 6, are standard properties of mutual information, it's worth highlighting their surprising power. It's useful to think of X as being A^*, the optimal action, and Y as being $Y_{t,a}$, the observation when selecting some action a. Then, combining these properties, we see that the next observation Y is expected to greatly reduce uncertainty about the optimal action A^* if and only if the distribution of Y varies greatly depending on the realization of A^*, in the sense that $D\left(P(Y|A^*=a^*) \,||\, P(Y)\right)$ is large on average. This fact is crucial to our analysis.

One implication of the next fact is that the expected reduction in entropy from observing the outcome $Y_{t,a}$ is always at least as large as that from observing the reward $R(Y_{t,a})$.

Fact 7 *(Weak Version of the Data Processing Inequality) If $Z = f(Y)$ for a deterministic function f, then $I(X;Y) \geq I(X;Z)$. If f is invertible, so Y is also a deterministic function of Z, then $I(X;Y) = I(X;Z)$.*

We close this section by stating a fact that guarantees $D\left(P(Y|X=x) \,||\, P(Y)\right)$ is well defined. It follows from a general property of conditional probability: for any random variable Z and event $E \subset \Omega$, if $\mathbb{P}(E) = 0$ then $\mathbb{P}(E|Z) = 0$ almost surely.

Fact 8 *For any $x \in \mathcal{X}$ with $\mathbb{P}(X=x) > 0$, $P(Y|X=x)$ is absolutely continuous with respect to $P(Y)$.*

3.1 Notation Under Posterior Distributions

As shorthand, we let

$$\mathbb{P}_t(\cdot) = \mathbb{P}(\cdot|\mathcal{F}_t) = \mathbb{P}(\cdot|\, A_1, Y_{1,A_1}, ..., A_{t-1}, Y_{t-1,A_{t-1}})$$

and $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\mathcal{F}_t]$. As before, we will define some simple shorthand notation for the distribution function of a random variable under $\mathbb{P}_t$. Let $P_t(X) = \mathbb{P}_t(X \in \cdot)$, $P_t(X|Y) = \mathbb{P}_t(X \in \cdot|Y)$ and $P_t(X|Y=y) = \mathbb{P}_t(X \in \cdot|Y=y)$.

The definitions of entropy and mutual information implicitly depend on some base measure over the sample space Ω. We will define special notation to denote entropy and mutual information under the posterior measure $\mathbb{P}_t(\cdot)$. Define

$$\begin{aligned} H_t(X) &= -\sum_{x \in \mathcal{X}} \mathbb{P}_t(X=x) \log \mathbb{P}_t(X=x) \\ H_t(X|Y) &= \mathbb{E}_t\left[-\sum_{x \in \mathcal{X}} \mathbb{P}_t(X=x|Y) \log \mathbb{P}_t(X=x|Y)\right] \\ I_t(X;Y) &= H_t(X) - H_t(X|Y). \end{aligned}$$

Because these quantities depend on the realizations of $A_1, Y_{1,A_1}, ..., A_{t-1}, Y_{t-1,A_{t-1}}$, they are random variables. By taking their expectation, we recover the standard definition of conditional entropy and conditional mutual information:

$$\begin{aligned} \mathbb{E}[H_t(X)] &= H(X|A_1, Y_{1,A_1}, ..., A_{t-1}, Y_{t-1,A_{t-1}}) \\ \mathbb{E}[I_t(X;Y)] &= I\left(X;Y|A_1, Y_{1,A_1}, ..., A_{t-1}, Y_{t-1,A_{t-1}}\right). \end{aligned}$$

4. Thompson Sampling

The Thompson sampling algorithm simply samples actions according to the posterior probability they are optimal. In particular, actions are chosen randomly at time t according to the sampling distribution $\pi_t^{\text{TS}} = \mathbb{P}(A^* = \cdot|\mathcal{F}_t)$. By definition, this means that for each $a \in \mathcal{A}$, $\mathbb{P}(A_t = a|\mathcal{F}_t) = \mathbb{P}(A^* = a|\mathcal{F}_t)$. This algorithm is sometimes called *probability matching* because the action selection distribution is *matched* to the posterior distribution of the optimal action.

This conceptually elegant probability matching scheme often admits a surprisingly simple and efficient implementation. Consider the case where $\mathcal{P} = \{p_\theta\}_{\theta\in\Theta}$ is some parametric family of distributions. The true outcome distribution p^* corresponds to a particular random index $\theta^* \in \Theta$ in the sense that $p^* = p_{\theta^*}$ almost surely. Practical implementations of Thompson sampling typically use two simple steps at each time t to randomly generate an action from the distribution α_t. First, an index $\hat{\theta}_t \sim \mathbb{P}(\theta^* \in \cdot|\mathcal{F}_t)$ is sampled from the posterior distribution of the true index θ^*. Then, the algorithm selects the action $A_t \in \arg\max_{a\in\mathcal{A}} \mathbb{E}\left[R(Y_{t,a})|\theta^* = \hat{\theta}_t\right]$ that would be optimal if the sampled parameter were actually the true parameter. We next provide an example of a Thompson sampling algorithm designed to address the problem of online linear optimization under bandit feedback.

4.1 Example of Thompson Sampling

Suppose each action $a \in \mathcal{A} \subset \mathbb{R}^d$ is defined by a d-dimensional feature vector, and almost surely there exists $\theta^* \in \mathbb{R}^d$ such that for each $a \in \mathcal{A}$, $\mathbb{E}\left[R(Y_{t,a})|p^*\right] = a^T\theta^*$. Assume θ^* is drawn from a normal distribution $N(\mu_0, \Sigma_0)$. When a is selected, only the realized reward $Y_{t,a} = R(Y_{t,a}) \in \mathbb{R}$ is observed. For each action a, reward noise $R(Y_{t,a}) - \mathbb{E}\left[R(Y_{t,a})|p^*\right]$ follows a Gaussian distribution with known variance. One can show that, conditioned on the history of observed data $\mathcal{F}_t$, θ^* remains normally distributed. Algorithm 1 provides an implementation of Thompson sampling for this problem. The expectations in step 3 can be computed efficiently via Kalman filtering.

Algorithm 1 Linear–Gaussian Thompson Sampling

1: **Sample Model**:
   $\hat{\theta}_t \sim N(\mu_{t-1}, \Sigma_{t-1})$
2: **Select Action**:
   $A_t \in \arg\max_{a\in\mathcal{A}}\langle a, \hat{\theta}_t\rangle$
3: **Update Statistics**:
   $\mu_t \leftarrow \mathbb{E}[\theta^*|\mathcal{F}_t]$
   $\Sigma_t \leftarrow \mathbb{E}[(\theta^* - \mu_t)(\theta^* - \mu_t)^\top|\mathcal{F}_t]$
4: **Increment t and Goto Step 1**

Algorithm 1 is efficient as long as the linear objective $\langle a, \hat{\theta}_t\rangle$ can be maximized efficiently over the action set $\mathcal{A}$. For this reason, the algorithm is implementable in important cases where other popular approaches, like the ConfidenceBall$_2$ algorithm of Dani et al. (2008),

are computationally intractable. Because the posterior distribution of θ^* has a closed form, Algorithm 1 is particularly efficient. Even when the posterior distribution is complex, however, one can often generate samples from this distribution using Markov chain Monte Carlo algorithms, enabling efficient implementations of Thompson sampling. A more detailed discussion of the strengths and potential advantages of Thompson sampling can be found in earlier work (Scott, 2010; Chapelle and Li, 2011; Russo and Van Roy, 2014; Gopalan et al., 2014).

5. The Information Ratio and a General Regret Bound

Our analysis will relate the expected regret of Thompson sampling to its expected information gain: the expected reduction in the entropy of the posterior distribution of A^*. The relationship between these quantities is characterized by what we call the *information ratio*,

$$\Gamma_t := \frac{\mathbb{E}_t\left[R(Y_{t,A^*}) - R(Y_{t,A_t})\right]^2}{I_t\left(A^*; (A_t, Y_{t,A_t})\right)}$$

which is the ratio between the square of expected regret and information gain in period t. Recall that, as described in Subsection 3.1, the subscript t on $\mathbb{E}_t$ and I_t indicates that these quantities are evaluated under the posterior measure $\mathbb{P}_t(\cdot) = \mathbb{P}(\cdot|\mathcal{F}_t)$.

Notice that if the information ratio is small, Thompson sampling can only incur large regret when it is expected to gain a lot of information about which action is optimal. This suggests its expected regret is bounded in terms of the maximum amount of information any algorithm could expect to acquire, which is at most the entropy of the prior distribution of the optimal action. Our next result shows this formally. We provide a general upper bound on the expected regret of Thompson sampling that depends on the time horizon T, the entropy of the prior distribution of A^*, $H(A^*)$, and any worst–case upper bound on the information ratio Γ_t. In the next section, we will provide bounds on Γ_t for some of the most widely studied classes of online optimization problems.

Proposition 1 *For any $T \in \mathbb{N}$, if $\Gamma_t \leq \overline{\Gamma}$ almost surely for each $t \in \{1, .., T\}$,*

$$\mathbb{E}\left[\text{Regret}(T, \pi^{\text{TS}})\right] \leq \sqrt{\overline{\Gamma} H(A^*) T}.$$

Proof Recall that $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\mathcal{F}_t]$ and we use I_t to denote mutual information evaluated under the base measure $\mathbb{P}_t(\cdot) = \mathbb{P}(\cdot|\mathcal{F}_t)$. Then,

$$\begin{aligned}
\mathbb{E}\left[\text{Regret}(T, \pi^{\text{TS}})\right] \overset{(a)}{=} \mathbb{E}\sum_{t=1}^{T}\mathbb{E}_t\left[R(Y_{t,A^*}) - R(Y_{t,A_t})\right] &= \mathbb{E}\sum_{t=1}^{T}\sqrt{\Gamma_t I_t\left(A^*; (A_t, Y_{t,A_t})\right)} \\
&\leq \sqrt{\overline{\Gamma}}\left(\mathbb{E}\sum_{t=1}^{T}\sqrt{I_t\left(A^*; (A_t, Y_{t,A_t})\right)}\right) \\
&\overset{(b)}{\leq} \sqrt{\overline{\Gamma} T \mathbb{E}\sum_{t=1}^{T} I_t\left(A^*; (A_t, Y_{t,A_t})\right)},
\end{aligned}$$

where (a) follows from the tower property of conditional expectation, and (b) follows from the Cauchy-Schwartz inequality. We complete the proof by showing that expected information gain cannot exceed the entropy of the prior distribution. For the remainder of this proof, let $Z_t = (A_t, Y_{t,A_t})$. Then, as discussed in Subsection 3.1,

$$\mathbb{E}\left[I_t\left(A^*; Z_t\right)\right] = I\left(A^*; Z_t | Z_1, ..., Z_{t-1}\right),$$

and therefore

$$\begin{aligned}\mathbb{E}\sum_{t=1}^{T} I_t\left(A^*; Z_t\right) = \sum_{t=1}^{T} I\left(A^*; Z_t | Z_1, ..., Z_{t-1}\right) &\overset{(c)}{=} I\left(A^*;\, (Z_1, ...Z_T)\right) \\ &= H(A^*) - H(A^* | Z_1, ...Z_T) \\ &\overset{(d)}{\leq} H(A^*),\end{aligned}$$

where (c) follows from the chain rule for mutual information (Fact 5), and (d) follows from the non-negativity of entropy (Fact 1). ■

6. Bounding the Information Ratio

This section establishes upper bounds on the information ratio in several important settings. This yields explicit regret bounds when combined with Proposition 1, and also helps to clarify the role the information ratio plays in our results: it roughly captures the extent to which sampling some actions allows the decision maker to make inferences about *different* actions. In the worst case, the information ratio depends on the number of actions, reflecting the fact that actions could provide no information about others. For problems with full information, the information ratio is bounded by a numerical constant, reflecting that sampling one action perfectly reveals the rewards that would have been earned by selecting any other action. The problems of online linear optimization under "bandit feedback" and under "semi–bandit feedback" lie between these two extremes, and the information ratio provides a natural measure of each problem's information structure. In each case, our bounds reflect that Thompson sampling is able to automatically exploit this structure.

We will compare our upper bounds on expected regret with known lower bounds. Some of these lower bounds were developed and stated in an adversarial framework, but were proved using the *probabilistic method*; authors fixed a family of distributions $\mathcal{P}$ and an initial distribution over p^* and lower bounded the expected regret under this environment of any algorithm. This provides lower bounds on $\inf_\pi \mathbb{E}\left[\text{Regret}(T, \pi)\right]$ in our framework at least for particular problem instances. Unfortunately, we are not aware of any general prior-dependent lower bounds, and this remains an important direction for the field.

Our bounds on the information ratio Γ_t will hold at any time t and under any posterior measure $\mathbb{P}_t$. To simplify notation, in our proofs we will omit the subscript t from $\mathbb{E}_t, \mathbb{P}_t, P_t, A_t, Y_t, H_t$, and I_t.

6.1 An Alternative Representation of the Information Ratio

Recall that the information ratio is defined to be

$$\frac{\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right]^2}{I\left(A^*; (A, Y_A)\right)}.$$

The following proposition expresses the information ratio of Thompson sampling in a form that facilitates further analysis. The proof uses that Thompson sampling matches the action selection distribution to the posterior distribution of the optimal action, in the sense that $\mathbb{P}\left(A^* = a\right) = \mathbb{P}\left(A = a\right)$ for all $a \in \mathcal{A}$.

Proposition 2

$$\begin{aligned}I\left(A^*; (A, Y_A)\right) &= \sum_{a \in \mathcal{A}} \mathbb{P}(A = a) I(A^*; Y_a) \\ &= \sum_{a, a^* \in \mathcal{A}} \mathbb{P}(A^* = a)\mathbb{P}(A^* = a^*)\left[D\left(P(Y_a | A^* = a^*) \,||\, P(Y_a)\right)\right].\end{aligned}$$

and

$$\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right] = \sum_{a \in \mathcal{A}} \mathbb{P}(A^* = a)\left(\mathbb{E}\left[R(Y_a) | A^* = a\right] - \mathbb{E}[R(Y_a)]\right.$$

The numerator of the information ratio measures the average difference between rewards generated from $P(Y_a)$, the posterior predictive distribution at a, and $P(Y_a | A^* = a)$, the posterior predictive distribution at a conditioned on a being the optimal action. It roughly captures how much knowing that the *selected action is optimal* influences the expected reward observed. The denominator measures how much, on average, knowing *which action is optimal* changes the observations at the selected action. Intuitively, the information ratio tends to be small when knowing which action is optimal significantly influences the anticipated observations at many other actions.

It's worth pointing out that this form of the information ratio bears a superficial resemblance to fundamental complexity terms in the multi-armed bandit literature. The results of Lai and Robbins (1985) and Agrawal et al. (1989) show the optimal asymptotic growth rate of regret is characterized by a ratio where the numerator depends on the difference between means of the reward distributions and the denominator depends on Kullback–Leibler divergences.

Proof Both proofs will use that the action A is selected based on past observations and independent random noise. Therefore, conditioned on the history, A is jointly independent

of A^* and the outcome vector $Y \equiv (Y_a)_{a\in\mathcal{A}}$.

$$\begin{aligned}
I(A^*;(A,Y_A)) &\overset{(a)}{=} I(A^*;A) + I(A^*;Y_A|A) \\
&\overset{(b)}{=} I(A^*;Y_A|A) \\
&= \sum_{a\in\mathcal{A}} \mathbb{P}(A=a) I(A^*;Y_A|A=a) \\
&\overset{(c)}{=} \sum_{a\in\mathcal{A}} \mathbb{P}(A=a) I(A^*;Y_a) \\
&\overset{(d)}{=} \sum_{a\in\mathcal{A}} \mathbb{P}(A=a) \left(\sum_{a^*\in\mathcal{A}} \mathbb{P}(A^*=a^*) D\left(P(Y_a|A^*=a^*) \,||\, P(Y_a)\right)\right) \\
&= \sum_{a,a^*\in\mathcal{A}} \mathbb{P}(A^*=a)\mathbb{P}(A^*=a^*) \left[D\left(P(Y_a|A^*=a^*) \,||\, P(Y_a)\right)\right],
\end{aligned}$$

where (a) follows from the chain rule for mutual information (Fact 5), (b) uses that A and A^* are independent and the mutual information between independent random variables is zero (Fact 4), (c) uses Fact 4 and that A is jointly independent of Y and A^*, and equality (d) uses Fact 6. Now, the numerator can be rewritten as,

$$\begin{aligned}
\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right] &= \sum_{a\in\mathcal{A}} \mathbb{P}(A^*=a)\mathbb{E}\left[R(Y_a)|A^*=a\right] - \sum_{a\in\mathcal{A}} \mathbb{P}(A=a)\mathbb{E}[R(Y_a)|A=a] \\
&= \sum_{a\in\mathcal{A}} \mathbb{P}(A^*=a)\left(\mathbb{E}\left[R(Y_a)|A^*=a\right] - \mathbb{E}[R(Y_a)]\right),
\end{aligned}$$

where the second equality uses that $\mathbb{P}(A=a) = \mathbb{P}(A^*=a)$ by the definition of Thompson sampling, and that Y is independent of the chosen action A. ■

6.2 Preliminaries

Here we state two basic facts that are used in bounding the information ratio. Proofs of both results are provided in the appendix for completeness.

The first fact lower bounds the Kullback–Leibler divergence between two bounded random variables in terms of the difference between their means. It follows trivially from an application of Pinsker's inequality.

Fact 9 *For any distributions P and Q such that that P is absolutely continuous with respect to Q, any random variable $X : \Omega \to \mathcal{X}$ and any $g : \mathcal{X} \to \mathbb{R}$ such that* $\sup g - \inf g \leq 1$*,*

$$\mathbb{E}_P\left[g(X)\right] - \mathbb{E}_Q\left[g(X)\right] \leq \sqrt{\frac{1}{2} D\left(P||Q\right)},$$

where $\mathbb{E}_P$ and $\mathbb{E}_Q$ denote the expectation operators under P and Q.

Because of Assumption 1, this fact shows

$$\mathbb{E}\left[R(Y_a)|A^*=a^*\right] - \mathbb{E}\left[R(Y_a)\right] \leq \sqrt{\frac{1}{2} D\left(P(Y_a|A^*=a^*) \,||\, P(Y_a)\right)}.$$

By the Cauchy–Schwartz inequality, for any vector $x \in \mathbb{R}^n$, $\sum_i x_i \leq \sqrt{n}\|x\|_2$. The next fact provides an analogous result for matrices. For any rank r matrix $M \in \mathbb{R}^{n\times n}$ with singular values $\sigma_1, ..., \sigma_r$, let

$$\|M\|_* := \sum_{i=1}^{r} \sigma_i, \qquad \|M\|_F := \sqrt{\sum_{k=1}^{m}\sum_{j=1}^{n} M_{i,j}^2} = \sqrt{\sum_{i=1}^{r}\sigma_i^2}, \qquad \text{Trace}(M) := \sum_{i=1}^{n} M_{ii},$$

denote respectively the Nuclear norm, Frobenius norm and trace of M.

Fact 10 *For any matrix $M \in \mathbb{R}^{k\times k}$,*

$$\text{Trace}\left(M\right) \leq \sqrt{\text{Rank}(M)}\|M\|_{\text{F}}.$$

6.3 Worst Case Bound

The next proposition provides a bound on the information ratio that holds whenever rewards are bounded, but that has an explicit dependence on the number of actions. This scaling cannot be improved in general, but we go on to show tighter bounds are possible for problems with different *information structures*.

Proposition 3 *For any $t \in \mathbb{N}$, $\Gamma_t \leq |\mathcal{A}|/2$ almost surely.*

Proof We bound the numerator of the information ratio by $|\mathcal{A}|/2$ times its denominator:

$$\begin{aligned}
\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right]^2 &\overset{(a)}{=} \left(\sum_{a\in\mathcal{A}} \mathbb{P}(A^*=a)\left(\mathbb{E}\left[R(Y_a)|A^*=a\right] - \mathbb{E}[R(Y_a)]\right)\right)^2 \\
&\overset{(b)}{\leq} |\mathcal{A}| \sum_{a\in\mathcal{A}} \mathbb{P}(A^*=a)^2 \left(\mathbb{E}\left[R(Y_a)|A^*=a\right] - \mathbb{E}[R(Y_a)]\right)^2 \\
&\leq |\mathcal{A}| \sum_{a,a^*\in\mathcal{A}} \mathbb{P}(A^*=a)\mathbb{P}(A^*=a^*) \left(\mathbb{E}\left[R(Y_a)|A^*=a^*\right] - \mathbb{E}[R(Y_a)]\right)^2 \\
&\overset{(c)}{\leq} \frac{|\mathcal{A}|}{2} \sum_{a,a^*\in\mathcal{A}} \mathbb{P}(A^*=a)\mathbb{P}(A^*=a^*) D\left(P(Y_a|A^*=a^*) \,||\, P(Y_a)\right) \\
&\overset{(d)}{=} \frac{|\mathcal{A}| I(A^*;(A,Y))}{2}
\end{aligned}$$

where (b) follows from the Cauchy–Schwarz inequality, (c) follows from Fact 9, and (a) and (d) follow from Proposition 2. ■

Combining Proposition 3 with Proposition 1 shows that $\mathbb{E}\left[\text{Regret}(T,\pi^{\text{TS}})\right] \leq \sqrt{\frac{1}{2}|\mathcal{A}|H(A^*)T}$. Bubeck and Liu (2013) show $\mathbb{E}\left[\text{Regret}(T,\pi^{\text{TS}})\right] \leq 14\sqrt{|\mathcal{A}|T}$ and that this bound is order optimal, in the sense that for any time horizon T and number of actions $|\mathcal{A}|$ there exists a prior distribution over p^* such that $\inf_\pi \mathbb{E}\left[\text{Regret}(T,\pi)\right] \geq \frac{1}{20}\sqrt{|\mathcal{A}|T}$.

6.4 Full Information

Our focus in this paper is on problems with *partial feedback*. For such problems, what the decision maker observes depends on the actions selected, which leads to a tension between exploration and exploitation. Problems with full information arise as an extreme point of our formulation where the outcome $Y_{t,a}$ is perfectly revealed by observing $Y_{t,\tilde{a}}$ for any $\tilde{a} \neq a$; what is learned does not depend on the selected action. The next proposition shows that under full information, the information ratio is bounded by 1/2.

Proposition 4 *Suppose for each $t \in \mathbb{N}$ there is a random variable $Z_t : \Omega \to \mathcal{Z}$ such that for each $a \in \mathcal{A}$, $Y_{t,a} = (a, Z_t)$. Then for all $t \in \mathbb{N}$, $\Gamma_t \leq 1/2$ almost surely.*

Proof As before we bound the numerator of Γ_t by $\sqrt{1/2}$ times the denominator:

$$\begin{aligned}
\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right] &\overset{(a)}{=} \sum_{a\in\mathcal{A}} \mathbb{P}(A^* = a)\left(\mathbb{E}\left[R(Y_a)|A^* = a\right] - \mathbb{E}[R(Y_a)]\right) \\
&\overset{(b)}{\leq} \sum_{a\in\mathcal{A}} \mathbb{P}(A^* = a)\sqrt{\frac{1}{2} D\left(P(Y_a|A^* = a) \,||\, P(Y_a)\right)} \\
&\overset{(c)}{\leq} \sqrt{\frac{1}{2}\sum_{a\in\mathcal{A}} \mathbb{P}(A^* = a) D\left(P(Y_a|A^* = a) \,||\, P(Y_a)\right)} \\
&\overset{(d)}{=} \sqrt{\frac{1}{2}\sum_{a,a^*\in\mathcal{A}} \mathbb{P}(A^* = a)\mathbb{P}(A^* = a^*) D\left(P(Y_a|A^* = a^*) \,||\, P(Y_a)\right)} \\
&\overset{(e)}{=} \sqrt{\frac{I(A^*;(A,Y))}{2}},
\end{aligned}$$

where again (a) and (e) follow from Proposition 2, (b) follows from Fact 9 and (c) follows from Jensen's inequality. Equality (d) follows because $D\left(P(Y_a|A^* = a^*) \,||\, P(Y_a)\right) = D\left(P(Z|A^* = a^*) \,||\, P(Z)\right)$ does not depend on the sampled action a under full information. ■

Combining this result with Proposition 1 shows $\mathbb{E}\left[\text{Regret}(T, \pi^{\text{TS}})\right] \leq \sqrt{\frac{1}{2}H(A^*)T}$. Further, a worst–case bound on the entropy of A^* shows that $\mathbb{E}\left[\text{Regret}(T, \pi^{\text{TS}})\right] \leq \sqrt{\frac{1}{2}\log(|\mathcal{A}|)T}$. The bound here improves upon this worst case bound since $H(A^*)$ can be much smaller than $\log(|\mathcal{A}|)$ when the prior distribution is informative.

6.5 Linear Optimization Under Bandit Feedback

The stochastic linear bandit problem has been widely studied (e.g Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010; Abbasi-Yadkori et al., 2011) and is one of the most important examples of a multi-armed bandit problem with "correlated arms." In this setting, each action is associated with a finite dimensional feature vector, and the mean reward generated by an action is the inner product between its known feature vector and some unknown parameter vector. Because of this structure, observations from taking one action allow the decision–maker to make inferences about other actions. The next proposition

bounds the information ratio for such problems. Its proof is essentially a generalization of the proof of Proposition 3.

Proposition 5 *If $\mathcal{A} \subset \mathbb{R}^d$ and for each $p \in \mathcal{P}$ there exists $\theta_p \in \mathbb{R}^d$ such that for all $a \in \mathcal{A}$*

$$\mathop{\mathbb{E}}_{y\sim p_a}[R(y)] = a^T\theta_p,$$

then for all $t \in \mathbb{N}$, $\Gamma_t \leq d/2$ almost surely.

Proof Write $\mathcal{A} = \{a_1, ..., a_K\}$ and, to reduce notation, for the remainder of this proof let $\alpha_i = \mathbb{P}(A^* = a_i)$. Define $M \in \mathbb{R}^{K\times K}$ by

$$M_{i,j} = \sqrt{\alpha_i\alpha_j}\left(\mathbb{E}[R(Y_{a_i})|A^* = a_j] - \mathbb{E}[R(Y_{a_i})]\right),$$

for all $i, j \in \{1, .., K\}$. Then, by Proposition 2,

$$\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right] = \sum_{i=1}^{K} \alpha_i\left(\mathbb{E}[R(Y_{a_i})|A^* = a_i] - \mathbb{E}[R(Y_{a_i})]\right) = \text{Trace(M)}.$$

Similarly, by Proposition 2,

$$\begin{aligned}
I(A^*;(A,Y_A)) &= \sum_{i,j} \alpha_i\alpha_j D\left(P(Y_{a_i}|A^* = a_j) \,||\, P(Y_{a_i})\right) \\
&\overset{(a)}{\geq} 2\sum_{i,j} \alpha_i\alpha_j\left(\mathbb{E}[R(Y_{a_i})|A^* = a_j] - \mathbb{E}[R(Y_{a_i})]\right)^2 \\
&= 2\|M\|_{\text{F}}^2,
\end{aligned}$$

where inequality (a) uses Fact 9. This shows, by Fact 10, that

$$\frac{\mathbb{E}\left[R(Y_{A^*}) - R(Y_A)\right]^2}{I(A^*;(A,Y_A))} \leq \frac{\text{Trace(M)}^2}{2\|M\|_{\text{F}}^2} \leq \frac{\text{Rank}(M)}{2}.$$

We now show $\text{Rank}(M) \leq d$. Define

$$\mu = \mathbb{E}\left[\theta_{p^*}\right] \qquad \mu^j = \mathbb{E}\left[\theta_{p^*}|A^* = a_j\right].$$

Then, by the linearity of the expectation operator,

$$M_{i,j} = \sqrt{\alpha_i\alpha_j}((\mu^j - \mu)^T a_i)$$

and therefore

$$M = \begin{bmatrix} \sqrt{\alpha_1}\left(\mu^1 - \mu\right)^T \\ \vdots \\ \vdots \\ \sqrt{\alpha_K}\left(\mu^k - \mu\right)^T \end{bmatrix} \begin{bmatrix} \sqrt{\alpha_1}a_1 & \cdots & \cdots & \sqrt{\alpha_K}a_K \end{bmatrix}.$$

This shows M is the product of a K by d matrix and a d by K matrix, and hence has rank at most d. ■

This result shows that $\mathbb{E}\left[\text{Regret}(T, \pi^{\text{TS}})\right] \leq \sqrt{\frac{1}{2} H(A^*) dT} \leq \sqrt{\frac{1}{2} \log(|\mathcal{A}|) dT}$ for linear bandit problems. Dani et al. (2007) show this bound is order optimal, in the sense that for any time horizon T and dimension d if the actions set is $\mathcal{A} = \{0,1\}^d$, there exists a prior distribution over p^* such that $\inf_\pi \mathbb{E}\left[\text{Regret}(T, \pi)\right] \geq c_0 \sqrt{\log(|\mathcal{A}|) dT}$ where c_0 is a constant the is independent of d and T. The bound here improves upon this worst case bound since $H(A^*)$ can be much smaller than $\log(|\mathcal{A}|)$ when the prior distribution is informative.

6.6 Combinatorial Action Sets and "Semi–Bandit" Feedback

To motivate the information structure studied here, consider a simple resource allocation problem. There are d possible projects, but the decision–maker can allocate resources to at most $m \leq d$ of them at a time. At time t, project $i \in \{1, .., d\}$ yields a random reward $\theta_{t,i}$, and the reward from selecting a subset of projects $a \in \mathcal{A} \subset \{a' \subset \{0, 1, ..., d\} : |a'| \leq m\}$ is $m^{-1} \sum_{i \in \mathcal{A}} \theta_{t,i}$. In the linear bandit formulation of this problem, upon choosing a subset of projects a the agent would only observe the overall reward $m^{-1} \sum_{i \in a} \theta_{t,i}$. It may be natural instead to assume that the outcome of each selected project $(\theta_{t,i} : i \in a)$ is observed. This type of observation structure is sometimes called "semi–bandit" feedback (Audibert et al., 2013).

A naive application of Proposition 5 to address this problem would show $\Gamma_t \leq d/2$. The next proposition shows that since the entire parameter vector $(\theta_{t,i} : i \in a)$ is observed upon selecting action a, we can provide an improved bound on the information ratio. The proof of the proposition is provided in the appendix.

Proposition 6 *Suppose $\mathcal{A} \subset \{a \subset \{0, 1, ..., d\} : |a| \leq m\}$, and that there are random variables $(\theta_{t,i} : t \in \mathbb{N}, i \in \{1, ..., d\})$ such that*

$$Y_{t,a} = (\theta_{t,i} : i \in a) \quad \textit{and} \quad R\left(Y_{t,a}\right) = \frac{1}{m} \sum_{i \in a} \theta_{t,i}.$$

Assume that the random variables $\{\theta_{t,i} : i \in \{1, ..., d\}\}$ are independent conditioned on $\mathcal{F}_t$ and $\theta_{t,i} \in [\frac{-1}{2}, \frac{1}{2}]$ almost surely for each (t, i). Then for all $t \in \mathbb{N}$, $\Gamma_t \leq \frac{d}{2m^2}$ almost surely.

In this problem, there are as many as $\binom{d}{m}$ actions, but because Thompson sampling exploits the structure relating actions to one another, its regret is only polynomial in m and d. In particular, combining Proposition 6 with Proposition 1 shows $\mathbb{E}\left[\text{Regret}(T, \pi^{\text{TS}})\right] \leq \frac{1}{m} \sqrt{dH(A^*)T}$. Since $H(A^*) \leq \log |\mathcal{A}| = O(m \log(\frac{d}{m}))$ this also yields a bound of order $\sqrt{\frac{d}{m} \log\left(\frac{d}{m}\right) T}$. As shown by Audibert et al. (2013), the lower bound for this problem is of order $\sqrt{\frac{d}{m} T}$, so our bound is order optimal up to a $\sqrt{\log(\frac{d}{m})}$ factor.[3] It's also worth

3. In their formulation, the reward from selecting action a is $\sum_{i \in a} \theta_{t,i}$, which is m times larger than in our formulation. The lower bound stated in their paper is therefore of order $\sqrt{mdT}$. They don't provide a complete proof of their result, but note that it follows from standard lower bounds in the bandit literature. In the proof of Theorem 5 in that paper, they construct an example in which the decision

pointing that, although there may be as many as $\binom{d}{m}$ actions, the action selection step of Thompson sampling is computationally efficient whenever the offline decision problem $\max_{a \in \mathcal{A}} \theta^T a$ can be solved efficiently.

7. Conclusion

This paper has provided a new analysis of Thompson sampling based on tools from information theory. As such, our analysis inherits the simplicity and elegance enjoyed by work in that field. Further, our results apply to a much broader range of information structures than those studied in prior work on Thompson sampling. Our analysis leads to regret bounds that highlight the benefits of soft knowledge, quantified in terms of the entropy of the optimal-action distribution. Such regret bounds yield insight into how future performance depends on past observations. This is key to assessing the benefits of exploration, and as such, to the design of more effective schemes that trade off between exploration and exploitation. In forthcoming work, we leverage this insight to produce an algorithm that outperforms Thompson sampling.

While our focus has been on providing theoretical guarantees for Thompson sampling, we believe the techniques and quantities used in the analysis may be of broader interest. Our formulation and notation may be complex, but the proofs themselves essentially follow from combining known relations in information theory with the tower property of conditional expectation, Jensen's inequality, and the Cauchy–Schwartz inequality. In addition, the information theoretic view taken in this paper may provide a fresh perspective on this class of problems.

Acknowledgments

Daniel Russo is supported by a Burt and Deedee McMurty Stanford Graduate Fellowship. This work was supported, in part, by the National Science Foundation under Grant CMMI-0968707. The authors would like to thank the anonymous reviewers for their helpful comments.

Appendix A. Proof of Basic Facts

A.1 Proof of Fact 9

This result is a consequence of *Pinsker's inequality*, (see Lemma 5.2.8 of Gray (2011)) which states that

$$\sqrt{\frac{1}{2} D(P||Q)} \geq \|P - Q\|_{\text{TV}} \tag{7}$$

maker plays m bandit games in parallel, each with d/m actions. Using that example, and the standard bandit lower bound (see Theorem 3.5 of Bubeck and Cesa-Bianchi (2012)), the agent's regret from each component must be at least $\sqrt{\frac{d}{m} T}$, and hence her overall expected regret is lower bounded by a term of order $m\sqrt{\frac{d}{m} T} = \sqrt{mdT}$.

where $\|P-Q\|_{\text{TV}}$ is the total variation distance between P and Q. When Ω is countable, $\|P-Q\|_{\text{TV}} = \frac{1}{2}\sum_{\omega}|P(\omega)-Q(\omega)|$. More generally, if P and Q are both absolutely continuous with respect to some base measure μ, then $\|P-Q\|_{\text{TV}} = \frac{1}{2}\int_{\Omega}|\frac{dP}{d\mu} - \frac{dQ}{d\mu}|d\mu$, where $\frac{dP}{d\mu}$ and $\frac{dQ}{d\mu}$ are the Radon–Nikodym derivatives of P and Q with respect to μ. We now prove Fact 9.

Proof Choose a base measure μ so that P and Q are absolutely continuous with respect to μ. This is always possible: since P is absolutely continuous with respect to Q by hypothesis, one could always choose this base measure to be Q. Let $f(\omega) = g(X(\omega)) - \inf_{\omega'} g(X(\omega')) - 1/2$ so that $f : \Omega \to [-1/2, 1/2]$ and f and $g(X)$ differ only by a constant. Then,

$$\begin{aligned}
\sqrt{\frac{1}{2}D(P||Q)} \geq \frac{1}{2}\int\left|\frac{dP}{d\mu} - \frac{dQ}{d\mu}\right|d\mu &\geq \frac{1}{2}\int\left|2\left(\frac{dP}{d\mu} - \frac{dQ}{d\mu}\right)f\right|d\mu \\
&\geq \int\left(\frac{dP}{d\mu} - \frac{dQ}{d\mu}\right)f d\mu \\
&= \int f dP - \int f dQ \\
&= \mathbb{E}_P[g(X)] - \mathbb{E}_Q[g(X)],
\end{aligned}$$

where the first inequality is Pinsker's inequality. ■

A.2 Proof of Fact 10

Proof Fix a rank r matrix $M \in \mathbb{R}^{k\times k}$ with singular values $\sigma_1 \geq ... \geq \sigma_r$. By the Cauchy Shwartz inequality,

$$\|M\|_* \overset{\text{Def}}{=} \sum_{i=1}^{r}\sigma_i \leq \sqrt{r}\sqrt{\sum_{i=1}^{r}\sigma_i^2} \overset{\text{Def}}{=} \sqrt{r}\|M\|_F. \tag{8}$$

Now, we show

$$\begin{aligned}
\text{Trace}(M) = \text{Trace}\left(\frac{1}{2}M + \frac{1}{2}M^T\right) \overset{(a)}{\leq} \left\|\frac{1}{2}M + \frac{1}{2}M^T\right\|_* &\overset{(b)}{\leq} \frac{1}{2}\|M\|_* + \frac{1}{2}\left\|M^T\right\|_* \\
&\overset{(c)}{=} \|M\|_* \overset{(d)}{\leq} \sqrt{r}\|M\|_F.
\end{aligned}$$

Here (b) follows from the triangle inequality and the fact that norms are homogeneous of degree one. Inequality (c) uses that the singular values of M and M^T are the same, and inequality (d) follows from equation (8).

To justify inequality (a), we show that for any symmetric matrix W, $\text{Trace}(W) \leq \|W\|_*$. To see this, let W be a rank r matrix and let $\tilde{\sigma}_1 \geq ... \geq \tilde{\sigma}_r$ denote its singular values. Since W is symmetric, it has r nonzero real valued eigenvalues $\lambda_1, .., \lambda_r$. If these are sorted so that $|\lambda_1| \geq ... \geq |\lambda_r|$ then $\tilde{\sigma}_i = |\lambda_i|$ for each $i \in \{1, .., r\}$.[4] This shows $\text{Trace}(M) = \sum_{i=1}^{r}\lambda_i \leq \sum_{i=1}^{r}\tilde{\sigma}_i = \|M\|_*$. ■

4. This fact is stated, for example, in Appendix A.5 of Boyd and Vandenberghe (2004).

Appendix B. Proof of Proposition 6

The proof relies on the following lemma, which lower bounds the information gain due to selecting an action a. The proof is provided below, and relies on the chain–rule of Kullback–Leibler divergence.

Lemma 1 *Under the conditions of Proposition 6, for any $a \in \mathcal{A}$,*

$$I(A^*; Y_a) \geq 2\sum_{i\in a}\mathbb{P}(i \in A^*)\left(\mathbb{E}\left[\theta_i | i \in A^*\right] - \mathbb{E}\left[\theta_i\right]\right)^2$$

Proof In the proof of this lemma, for any $i \in a$ we let $\theta_{<i} = (\theta_j : j < i, j \in a)$. Since $a \in \mathcal{A}$ is fixed throughout this proof, we do not display the dependence of $\theta_{<i}$ on a. Recall that by Fact 6,

$$I(A^*; Y_a) = \sum_{a^*\in\mathcal{A}}\mathbb{P}(A^* = a^*)D\left(P(Y_a|A^* = a^*)\,||\,P(Y_a)\right).$$

where here,

$$\begin{aligned}
D\left(P(Y_a|A^* = a^*)\,||\,P(Y_a)\right) &= D\left(P\left((\theta_i : i \in a)|A^* = a^*\right)\;||\;P\left((\theta_i : i \in a)\right)\right) \\
&\overset{(a)}{=} \sum_{i\in a}\mathbb{E}\left[D\left(P\left(\theta_i|A^* = a^*, \theta_{<i}\right)\;||\;P\left(\theta_i|\theta_{<i}\right)\right)\middle|A^* = a^*\right] \\
&\overset{(b)}{\geq} 2\sum_{i\in a}\mathbb{E}\left[\left(\mathbb{E}\left[\theta_i|A^* = a^*, \theta_{<i}\right] - \mathbb{E}\left[\theta_i|\theta_{<i}\right]\right)^2\middle|A^* = a^*\right] \\
&\overset{(c)}{=} 2\sum_{i\in a}\mathbb{E}\left[\left(\mathbb{E}\left[\theta_i|A^* = a^*, \theta_{<i}\right] - \mathbb{E}\left[\theta_i\right]\right)^2\middle|A^* = a^*\right] \\
&\overset{(d)}{\geq} 2\sum_{i\in a}\left(\mathbb{E}\left[\theta_i|A^* = a^*\right] - \mathbb{E}\left[\theta_i\right]\right)^2.
\end{aligned}$$

Equality (a) follows from the chain rule of Kullback–Leibler divergence, (b) follows from Fact 9, (c) follows from the assumption that $(\theta_i : 1 \leq i \leq d)$ are independent conditioned on any history of observations, and (d) follows from Jensen's inequality and the tower property

of conditional expectation. Now, we can show

$$\begin{aligned}
I(A^*;Y_a) &\geq 2\sum_{i\in a}\sum_{a^*\in\mathcal{A}}\mathbb{P}(A^*=a^*)\left(\mathbb{E}\left[\theta_i|A^*=a^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2\\
&= 2\sum_{i\in a}\mathbb{P}(i\in A^*)\sum_{a^*\in\mathcal{A}}\frac{\mathbb{P}(A^*=a^*)}{\mathbb{P}(i\in A^*)}\left(\mathbb{E}\left[\theta_i|A^*=a^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2\\
&\geq 2\sum_{i\in a}\mathbb{P}(i\in A^*)\sum_{a^*:i\in a^*}\frac{\mathbb{P}(A^*=a^*)}{\mathbb{P}(i\in A^*)}\left(\mathbb{E}\left[\theta_i|A^*=a^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2\\
&= 2\sum_{i\in a}\mathbb{P}(i\in A^*)\sum_{a^*:i\in a^*}\mathbb{P}(A^*=a^*|i\in A^*)\left(\mathbb{E}\left[\theta_i|A^*=a^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2\\
&\overset{(e)}{\geq} 2\sum_{i\in a}\mathbb{P}(i\in A^*)\left(\sum_{a^*:i\in a^*}\mathbb{P}(A^*=a^*|i\in A^*)\left(\mathbb{E}\left[\theta_i|A^*=a^*\right]-\mathbb{E}\left[\theta_i\right]\right)\right)^2\\
&\overset{(f)}{=} 2\sum_{i\in a}\mathbb{P}(i\in A^*)\left(\mathbb{E}\left[\theta_i|i\in A^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2,
\end{aligned}$$

where (e) and (f) follow from Jensen's inequality and the tower property of conditional expectation, respectively. ■

Lemma 2 *Under the conditions of Proposition 6,*

$$I(A^*;(A,Y_A))\geq 2\sum_{i=1}^{d}\mathbb{P}(i\in A^*)^2\left(\mathbb{E}\left[\theta_i|i\in A^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2.$$

Proof By Lemma 1 and the tower property of conditional expectation,

$$\begin{aligned}
I(A^*;(A;Y_A)) &= \mathbb{P}(A^*=a)I(A^*;Y_a)\\
&\geq 2\sum_{a\in\mathcal{A}}\mathbb{P}(A^*=a)\left[\sum_{i\in a}\mathbb{P}(i\in A^*)\left(\mathbb{E}\left[\theta_i|i\in A^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2\right]\\
&= 2\sum_{i=1}^{d}\sum_{a:i\in a}\mathbb{P}(A^*=a)\left(\mathbb{P}(i\in A^*)\left(\mathbb{E}\left[\theta_i|i\in A^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2\right)\\
&= 2\sum_{i=1}^{d}\mathbb{P}(i\in A^*)^2\left(\mathbb{E}\left[\theta_i|i\in A^*\right]-\mathbb{E}\left[\theta_i\right]\right)^2.
\end{aligned}$$

■

Now, we complete the proof of Proposition 6.

Proof The proof establishes that the numerator of the information ratio is less than $d/2m^2$ times its denominator:

$$\begin{aligned}
m\mathbb{E}[R(Y_{A^*})-R(Y_A)] &= \mathbb{E}\sum_{i\in A^*}\theta_i-\mathbb{E}\sum_{i\in A}\theta_i\\
&= \sum_{i=1}^{d}\mathbb{P}(i\in A^*)\mathbb{E}[\theta_i|i\in A^*]-\sum_{i=1}^{d}\mathbb{P}(i\in A)\mathbb{E}[\theta_i|i\in A]\\
&= \sum_{i=1}^{d}\mathbb{P}(i\in A^*)\left(\mathbb{E}[\theta_i|i\in A^*]-\mathbb{E}[\theta_i]\right)\\
&\leq \sqrt{d}\sqrt{\sum_{i=1}^{d}\mathbb{P}(i\in A^*)^2\left(\mathbb{E}[\theta_i|i\in A^*]-\mathbb{E}[\theta_i]\right)^2}\\
&\leq \sqrt{\frac{dI(A^*;(A,Y_A))}{2}}.
\end{aligned}$$

■

Appendix C. Proof of Fact 6

We could not find a reference that proves Fact 6 in a general setting, and will therefore provide a proof here.

Consider random variables $X:\Omega\to\mathcal{X}$ and $Y:\Omega\to\mathcal{Y}$ where $\mathcal{X}$ is assumed to be a finite set but Y is a general random variable. We show

$$I(X;Y)=\sum_{x\in\mathcal{X}}\mathbb{P}\left(X=x\right)D\left(\mathbb{P}\left(Y\in\cdot|X=x\right)||\,\mathbb{P}\left(Y\in\cdot\right)\right).$$

When Y has finite support this result follows easily by using that $\mathbb{P}(X=x,Y=y)=\mathbb{P}(X=x)\mathbb{P}(Y=y|X=x)$. For an extension to general random variables Y we follow chapter 5 of Gray (2011).

The extension follows by considering quantized versions of the random variable Y. For a finite partition $\mathcal{Q}=\{Q_i\subset\mathcal{Y}\}_{i\in I}$, we denote by $Y_{\mathcal{Q}}$ the quantization of Y defined so that $Y_{\mathcal{Q}}(\omega)=i$ if and only if $Y(\omega)\in Q_i$. As shown in Gray (2011), for two probability measures P_1 and P_2 on $(\Omega,\mathcal{F})$,

$$D\left(P_1(Y\in\cdot)||P_2(Y\in\cdot)\right)=\sup_{\mathcal{Q}}D\left(P_1(Y_{\mathcal{Q}}\in\cdot)||P_2(Y_{\mathcal{Q}}\in\cdot)\right),\tag{9}$$

where the supremum is taken over all finite partitions of $\mathcal{Y}$, and whenever $\overline{\mathcal{Q}}$ is a refinement of $\mathcal{Q}$,

$$D\left(P_1(Y_{\overline{\mathcal{Q}}}\in\cdot)||P_2(Y_{\overline{\mathcal{Q}}}\in\cdot)\right)\geq D\left(P_1(Y_{\mathcal{Q}}\in\cdot)||P_2(Y_{\mathcal{Q}}\in\cdot)\right).\tag{10}$$

Since $I(X;Y)=D\left(\mathbb{P}(X\in\cdot,Y\in\cdot)\,||\,\mathbb{P}(X\in\cdot)\mathbb{P}(Y\in\cdot)\right)$ these properties also apply to $I(X;Y_{\mathcal{Q}})$. Now, for each $x\in\mathcal{X}$ we can introduce a sequence of successively refined partitions $(\mathcal{Q}_n^x)_{n\in\mathbb{N}}$

so that

$$\begin{aligned} D\left(\mathbb{P}\left(Y \in \cdot | X = x\right) || \mathbb{P}\left(Y \in \cdot\right)\right) &= \lim_{n\to\infty} D\left(\mathbb{P}\left(Y_{\mathcal{Q}_n^x} \in \cdot | X = x\right) || \mathbb{P}\left(Y_{\mathcal{Q}_n^x} \in \cdot\right)\right) \\ &= \sup_{\mathcal{Q}} D\left(\mathbb{P}\left(Y_{\mathcal{Q}} \in \cdot | X = x\right) || \mathbb{P}\left(Y_{\mathcal{Q}} \in \cdot\right)\right). \end{aligned}$$

Let the finite partition $\mathcal{Q}_n$ denote the refinement of the partitions $\{\mathcal{Q}_n^x\}_{x\in\mathcal{X}}$. That is, for each x, every set in Q_n^x can be written as the union of sets in $\mathcal{Q}_n$. Then,

$$\begin{aligned} I(X;Y) &\geq \lim_{n\to\infty} I(X; Y_{\mathcal{Q}_n}) \\ &= \lim_{n\to\infty} \sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x\right) D\left(\mathbb{P}\left(Y_{\mathcal{Q}_n} \in \cdot | X = x\right) || \mathbb{P}\left(Y_{\mathcal{Q}_n} \in \cdot\right)\right) \\ &\geq \lim_{n\to\infty} \sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x\right) D\left(\mathbb{P}\left(Y_{\mathcal{Q}_n^x} \in \cdot | X = x\right) || \mathbb{P}\left(Y_{\mathcal{Q}_n^x} \in \cdot\right)\right) \\ &= \sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x\right) D\left(\mathbb{P}\left(Y \in \cdot | X = x\right) || \mathbb{P}\left(Y \in \cdot\right)\right) \\ &= \sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x\right) \sup_{\mathcal{Q}^x} D\left(\mathbb{P}\left(Y_{\mathcal{Q}^x} \in \cdot | X = x\right) || \mathbb{P}\left(Y_{\mathcal{Q}^x} \in \cdot\right)\right) \\ &\geq \sup_{\mathcal{Q}} \sum_{x\in\mathcal{X}} \mathbb{P}\left(X = x\right) D\left(\mathbb{P}\left(Y_{\mathcal{Q}} \in \cdot | X = x\right) || \mathbb{P}\left(Y_{\mathcal{Q}} \in \cdot\right)\right) \\ &= \sup_{\mathcal{Q}} I(X; Y_{\mathcal{Q}}) = I(X;Y), \end{aligned}$$

hence the inequalities above are equalities and our claim is established.

Appendix D. Technical Extensions

D.1 Infinite Action Spaces

In this section, we discuss how to extend our results to problems where the action space is infinite. For concreteness, we will focus on linear optimization problems. Specifically, throughout this section our focus is on problems where $\mathcal{A} \subset \mathbb{R}^d$ and for each $p \in \mathcal{P}$ there is some $\theta_p \in \Theta \subset \mathbb{R}^d$ such that for all $a \in \mathcal{A}$

$$\mathop{\mathbb{E}}_{y\sim p_a}\left[R(y)\right] = a^T\theta_p.$$

We assume further that there is a fixed constant $c \in \mathbb{R}$ such that $\sup_{a\in\mathcal{A}} \|a\|_2 \leq c$ and $\sup_{\theta\in\Theta} \|\theta\|_2 \leq c$. Because $\mathcal{A}$ is compact, it can be extremely well approximated by a finite set. In particular, we can choose a finite cover $\mathcal{A}_\epsilon \subset \mathcal{A}$ with $\log|A_\epsilon| = O(d\log(dc/\epsilon))$ such that for every $a \in \mathcal{A}$, $\min_{\tilde{a}\in\mathcal{A}_\epsilon} \|a - \tilde{a}\|_2 \leq \epsilon$. By the Cauchy-Schwartz inequality, this implies that $|\left(a - \tilde{a}\right)^T \theta_P| \leq c\epsilon$ for every θ_P.

We introduce a quantization A_Q^* of the random variable A^*. A_Q^* is supported only on the set $\mathcal{A}_\epsilon$, but satisfies $\|A^*(\omega) - A_Q^*(\omega)\|_2 \leq \epsilon$ for each $\omega \in \Omega$.

Consider a variant of Thompson sampling that randomizes according to the distribution of A_Q^*. That is, for each $a \in \mathcal{A}_\epsilon$, $\mathbb{P}\left(A_t = a|\mathcal{F}_t\right) = \mathbb{P}\left(A_Q^* = a|\mathcal{F}_t\right)$. Then, our analysis shows,

$$\begin{aligned} \mathbb{E}\sum_{t=1}^T \left[R(Y_{t,A^*}) - R(Y_{t,A_t})\right] &= \mathbb{E}\sum_{t=1}^T \left[R(Y_{t,A^*}) - R(Y_{t,A_Q^*})\right] + \mathbb{E}\sum_{t=1}^T \left[R(Y_{t,A_Q^*}) - R(Y_{t,A_t})\right] \\ &\leq \epsilon T + \sqrt{dTH\left(A_Q^*\right)}. \end{aligned}$$

This new bound depends on the time horizon T, the dimension d of the linear model, the entropy of the prior distribution of the quantized random variable A_Q^* and the discretization error ϵT. Since $H(A_Q^*) \leq \log\left(|\mathcal{A}_\epsilon|\right)$, by choosing $\epsilon = (cT)^{-1}$ and choosing a finite cover with $\log\left(|\mathcal{A}_\epsilon|\right) = O(d\log(dcT))$ we attain a regret bound of order $d\sqrt{T\log(dcT)}$. Note that for $d \leq T$, this bound is of order $d\sqrt{T\log(cT)}$, but for $d > T \geq 1$, we have a trivial regret bound of $T < \sqrt{dT}$.

Here, for simplicity, we have considered a variant of Thompson sampling that randomizes according to the posterior distribution of the quantized random variable A_Q^*. However, with a somewhat more careful analysis, one can provide a similar result for an algorithm that randomizes according to the posterior distribution of A^*.

D.2 Unbounded Noise

In this section we relax Assumption 1, which requires that reward distributions are uniformly bounded. This assumption is required for Fact 9 to hold, but otherwise was never used in our analysis. Here, we show that an analogue to Fact 9 holds in the more general setting where reward distributions are *sub–Gaussian*. This yields more general regret bounds.

A random variable is sub–Gaussian if its moment generating function is dominated by that of a Gaussian random variable. Gaussian random variables are sub–Gaussian, as are real-valued random varaibles with bounded support.

Definition 1 *Fix a deterministic constant $\sigma \in \mathbb{R}$. A real–valued random variable X is σ sub–Gaussian if for all $\lambda \in \mathbb{R}$*

$$\mathbb{E}\left[\exp\{\lambda X\}\right] \leq \exp\{\lambda^2\sigma^2/2\}.$$

More generally X is σ sub–Gaussian conditional on a sigma-algebra $\mathcal{G} \subset \mathcal{F}$ if for all $\lambda \in \mathbb{R}$,

$$\mathbb{E}\left[\exp\{\lambda X\}|\mathcal{G}\right] \leq \exp\{\lambda^2\sigma^2/2\}$$

almost surely.

The next lemma extends Fact 9 to sub-Gaussian random variables.

Lemma 3 *Suppose there is a $\mathcal{F}_t$ measurable random variable σ such that for each $a \in \mathcal{A}$, $R(Y_{t,a}))$ is σ sub-Gaussian conditional on $\mathcal{F}_t$. Then for every $a, a^* \in \mathcal{A}$*

$$\mathbb{E}_t[R(Y_{t,a})|A^* = a^*] - \mathbb{E}_t[R(Y_{t,a})] \leq \sigma\sqrt{2D\left(P_t(Y_{t,a}|A* = a^*) \,||\, P_t(Y_{t,a})\right)},$$

almost surely.

Using this Lemma in place of Fact 9 in our analysis leads to the following corollary.

Corollary 1 *Fix a deterministic constant $\sigma \in \mathbb{R}$. Suppose that conditioned on $\mathcal{F}_t$, $R(Y_{t,a}) - \mathbb{E}\left[R(Y_{t,a})|\mathcal{F}_t\right]$ is σ sub–Gaussian. Then*

$$\Gamma_t \leq 2|\mathcal{A}|\sigma^2$$

almost surely for each $t \in \mathbb{N}$. Furthermore, $\Gamma_t \leq 2\sigma^2$ under the conditions of Proposition 4, $\Gamma_t \leq 2d\sigma^2$ under the conditions of Proposition 5, and $\Gamma_t \leq \frac{2d\sigma^2}{m^2}$ under the conditions of Proposition 6.

It's worth highlighting two cases under which the conditions of Corollary 1 are satisfied. The first is a setting with a Gaussian prior and Guassian reward noise. The second case is when under any model $p \in \mathcal{P}$ expected rewards lie in a bounded interval and reward noise is sub–Gaussian.

1. Suppose that for each $a \in \mathcal{A}$, $R(Y_{t,a})$ follows a Guassian distribution with variance less than σ^2, $Y_{t,a} = R(Y_{t,a})$ and, reward noise $R(Y_{t,a}) - \mathbb{E}[R(Y_{t,a})|p^*]$ is Gaussian with known variance. Then, the posterior predictive distribution of $R(Y_{t,a})$, $\mathbb{P}\left(R(Y_{t,a}) \in \cdot|\mathcal{F}_t\right)$ is Gaussian with variance less than σ^2 for each $a \in \mathcal{A}$ and $t \in \mathbb{N}$.
2. Fix constants $C \in \mathbb{R}$ and $\sigma \in \mathbb{R}$. Suppose that almost surely $\mathbb{E}\left[R(Y_{t,a})|p^*\right] \in \left[\frac{-C}{2}, \frac{C}{2}\right]$ and reward noise is $R(Y_{t,a}) - \mathbb{E}\left[R(Y_{t,a})|p^*\right]$ is σ sub–Gaussian. Then, conditioned on the history, $R(Y_{t,a}) - \mathbb{E}\left[R(Y_{t,a})|\mathcal{F}_t\right]$ is $\sqrt{C^2 + \sigma^2}$ sub-Gaussian almost surely.

The first case relies on the fact that $\mathbb{E}[R(Y_{t,a})|p^*]$ is C–sub-Gaussain, as well as the following fact about sub-Gaussian random variables.

Fact 11 *Consider random variables $(X_1, ..., X_K)$. If for each X_i there is a deterministic σ_i such that X_i is σ_i sub–Gaussian conditional on $X_1, .., X_{i-1}$, then $\sum_{i=1}^K X_i$ is $\sqrt{\sum_{i=1}^K \sigma_i^2}$ sub-Gaussian.*

Proof For any $\lambda \in \mathbb{R}$,

$$\begin{aligned}\mathbb{E}\left[\exp\{\lambda \sum_{i=1}^K X_i\}\right] = \mathbb{E}\left[\prod_{i=1}^K \exp\{\lambda X_i\}\right] &= \mathbb{E}\left[\prod_{i=1}^K \mathbb{E}\left[\exp\{\lambda X_i\}\middle| X_1, ..X_{i-1}\right]\right] \\ &\leq \prod_{i=1}^K \exp\left\{\frac{\lambda^2\sigma_i^2}{2}\right\} = \exp\left\{\frac{\lambda^2(\sum_i \sigma_i^2)}{2}\right\}.\end{aligned}$$

■

The proof of Lemma 3 relies on the following property of Kullback–Leibler divergence.

Fact 12 *(Variational form of KL–Divergence given in Theorem 5.2.1 of Gray (2011)) Fix two probability distributions P and Q such that P is absolutely continuous with respect to Q. Then,*

$$D(P||Q) = \sup_X \left\{\mathbb{E}_P\left[X\right] - \log \mathbb{E}_Q\left[\exp\{X\}\right]\right\},$$

where $\mathbb{E}_P$ and $\mathbb{E}_Q$ denote the expectation operator under P and Q respectively, and the supremum is taken over all real valued random variables X such that $\mathbb{E}_P\left[X\right]$ is well defined and $\mathbb{E}_Q\left[\exp\{X\}\right] < \infty$.

We now show that Lemma 3 follows from the variational form of KL–Divergence.

Proof Define the random variable $X = R\left(Y_{t,a}\right) - \mathbb{E}_t\left[R\left(Y_{t,a}\right)\right]$. Then, for abitrary $\lambda \in \mathbb{R}$, applying Fact 12 to λX yields

$$\begin{aligned}D\left(P_t\left(Y_{t,a}|A^* = a^*\right) \,||\, P_t(Y_{t,a})\right) &\geq \lambda\mathbb{E}_t\left[X|A^* = a^*\right] - \log \mathbb{E}_t\left[\exp\{\lambda X\}\right] \\ &= \lambda\left(\mathbb{E}_t[R\left(Y_{t,a}\right)|A^* = a^*] - \mathbb{E}_t\left[R\left(Y_{t,a}\right)\right]\right) - \log \mathbb{E}\left[\exp\{\lambda X\}|\mathcal{F}_t\right] \\ &\geq \lambda\left(\mathbb{E}_t[R\left(Y_{t,a}\right)|A^* = a^*] - \mathbb{E}_t\left[R\left(Y_{t,a}\right)\right]\right) - (\lambda^2\sigma^2/2).\end{aligned}$$

Maximizing over λ yields the result. ■

References

Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in Neural Information Processing Systems*, 24, 2011.

R. Agrawal, D. Teneketzis, and V. Anantharam. Asymptotically efficient adaptive allocation schemes for controlled iid processes: Finite parameter space. *IEEE Transactions on Automatic Control*, 34(3), 1989.

S. Agrawal and N. Goyal. Analysis of Thompson sampling for the multi-armed bandit problem. In *Proceedings of the 21st Annual Conference on Learning Theory*, 2012.

S. Agrawal and N. Goyal. Further optimal regret bounds for thompson sampling. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 99–107, 2013a.

S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th International Conference on Machine Learning*, pages 127–135, 2013b.

J.-Y. Audibert, S. Bubeck, and G. Lugosi. Regret in online combinatorial optimization. *Mathematics of Operations Research*, 39(1):31–45, 2013.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

S.P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and trends in machine learning*, 5(1):1–122, 2012.

S. Bubeck and C.-Y. Liu. Prior-free and prior-dependent regret bounds for Thompson sampling. In *Advances in Neural Information Processing Systems*, 2013.

O. Chapelle and L. Li. An empirical evaluation of Thompson sampling. In *Neural Information Processing Systems (NIPS)*, 2011.

T.M. Cover and J.A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

V. Dani, S.M. Kakade, and T.P. Hayes. The price of bandit information for online optimization. In *Advances in Neural Information Processing Systems*, pages 345–352, 2007.

V. Dani, T.P. Hayes, and S.M. Kakade. Stochastic linear optimization under bandit feedback. In *Proceedings of the 21st Annual Conference on Learning Theory*, pages 355–366, 2008.

J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed bandit allocation indices*. John Wiley & Sons, Ltd, 2011. ISBN 9780470980033.

A. Gopalan, S. Mannor, and Y. Mansour. Thompson sampling for complex online problems. In *Proceedings of the 31st International Conference on Machine Learning*, pages 100–108, 2014.

T. Graepel, J.Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proceedings of the 27th International Conference on Machine Learning*, pages 13–20, 2010.

R.M. Gray. *Entropy and information theory*. Springer, 2011.

E. Kauffmann, N. Korda, and R. Munos. Thompson sampling: an asymptotically optimal finite time analysis. In *International Conference on Algorithmic Learning Theory*, 2012.

N. Korda, E. Kaufmann, and R. Munos. Thompson sampling for one-dimensional exponential family bandits. In *Advances in Neural Information Processing Systems*, 2013.

T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

L. Li. Generalized Thompson sampling for contextual bandits. *arXiv preprint arXiv:1310.7163*, 2013.

B.C. May, N. Korda, A. Lee, and D.S. Leslie. Optimistic Bayesian sampling in contextual-bandit problems. *The Journal of Machine Learning Research*, 98888:2069–2106, 2012.

P. Rusmevichientong and J.N. Tsitsiklis. Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411, 2010.

D. Russo and B. Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 2014. URL `http://dx.doi.org/10.1287/moor.2014.0650`.

I.O. Ryzhov, W.B. Powell, and P.I. Frazier. The knowledge gradient algorithm for a general class of online learning problems. *Operations Research*, 60(1):180–195, 2012.

S.L. Scott. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010.

S.L. Scott. Overview of content experiments: Multi-armed bandit experiments, 2014. URL `https://support.google.com/analytics/answer/2844870?hl=en`. [Online; accessed 9-February-2014].

N. Srinivas, A. Krause, S.M. Kakade, and M. Seeger. Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250 –3265, 2012.

L. Tang, R. Rosales, A. Singh, and D. Agarwal. Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1587–1594. ACM, 2013.

W.R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

Journal of Machine Learning Research 17 (2016) 1-26 Submitted 6/14; Revised 12/15; Published 5/16

Compressed Gaussian Process for Manifold Regression

Rajarshi Guhaniyogi rguhaniy@ucsc.edu
Department of Applied Mathematics & Statistics
University of California
Santa Cruz, CA 95064, USA

David B. Dunson dunson@duke.edu
Department of Statistical Science
Duke University
Durham, NC 27708-0251, USA

Editor: Francois Caron

Abstract

Nonparametric regression for large numbers of features (p) is an increasingly important problem. If the sample size n is massive, a common strategy is to partition the feature space, and then separately apply simple models to each partition set. This is not ideal when n is modest relative to p, and we propose an alternative approach relying on random compression of the feature vector combined with Gaussian process regression. The proposed approach is particularly motivated by the setting in which the response is conditionally independent of the features given the projection to a low dimensional manifold. Conditionally on the random compression matrix and a smoothness parameter, the posterior distribution for the regression surface and posterior predictive distributions are available analytically. Running the analysis in parallel for many random compression matrices and smoothness parameters, model averaging is used to combine the results. The algorithm can be implemented rapidly even in very large p and moderately large n nonparametric regression, has strong theoretical justification, and is found to yield state of the art predictive performance.

Keywords: Compressed regression; Gaussian process; Gaussian random projection; Large p; Manifold regression.

1. Introduction

With recent technological progress, it is now routine in many disciplines to collect data containing large numbers of features, ranging from thousands to millions. To account for complex nonlinear relationships between the features and the response, nonparametric regression models are employed. For example,

$$y = \mu_0(\boldsymbol{x}) + \epsilon, \ \epsilon \sim N(0, \sigma^2),$$

where $\boldsymbol{x} \in \mathcal{R}^p$, $\mu_0(\cdot)$ is the unknown regression function and ϵ is a residual. When p is large, estimating μ_0 can lead to a statistical and computational curse of dimensionality. One strategy for combatting this curse is dimensionality reduction via variable selection or (more broadly) subspace learning, with the high-dimensional features replaced with their projection to a d-dimensional subspace or manifold with $d \ll p$. In many applications, the relevant information about the high-dimensional features can be encoded in such low dimensional coordinates.

There is a vast frequentist literature on subspace learning for regression, typically employing a two stage approach. In the first stage, a dimensionality reduction technique is used to obtain lower dimensional features that can "faithfully" represent the higher dimensional features. Examples include principal components analysis and more elaborate methods that accommodate non-linear subspaces, such as isomap (Tenenbaum et al., 2000) and Laplacian eigenmaps (Belkin and Niyogi, 2003; Guerrero et al., 2011). Once lower dimensional features are obtained, the second stage uses these features in standard regression and classification procedures as if they were observed initially. Such two stage approaches rely on learning the manifold structure embedded in the high dimensional features, which adds unnecessary computational burden when inferential interest lies mainly in prediction.

Another thread of research focuses on prediction using divide-and-conquer techniques. As the number of features increases, the problem of finding the best splitting attribute becomes intractable, so that CART (Breiman et al., 1984), MARS and multiple tree models, such as Random Forest (Breiman, 2001), cannot be efficiently applied. A much simpler approach is to apply high dimensional clustering techniques, such as metis, cover trees and spectral clustering. Once the observations are clustered into a few groups, simple models (glm, Lasso etc) are fitted in each cluster (Zhang et al., 2013). Such methods are sensitive to clustering, do not characterize predictive uncertainty, and may lack efficiency, an important consideration outside the $n \gg p$ setting. There is also a recent literature on scaling up sparse optimization methods, such as Lasso, to large p and n settings relying on algorithms that can exploit multiple processors in a distributed manner e.g., (Boyd et al., 2011). However, such methods are yet to be developed for non-linear manifold regression, which is the central focus of this article.

This naturally motivates Bayesian models that simultaneously learn the mapping to the lower-dimensional subspace along with the regression function in the coordinates on this subspace, providing a characterization of predictive uncertainties. Tokdar et al. (2010) proposes a logistic Gaussian process approach, while Reich et al. (2011) use finite mixture models for sufficient dimension reduction. Page et al. (2013) propose a Bayesian nonparametric model for learning of an affine subspace in classification problems. These approaches have the disadvantages of being limited to linear subspaces, lacking scalability beyond a few dozen features and having potential sensitivity to features corrupted with noise. There is also a literature on Bayesian methods that accommodate non-linear subspaces, ranging from Gaussian process latent variable models (GP-LVMs) (Lawrence, 2005) for probabilistic nonlinear PCA to mixture factor models Chen et al. (2010). However, such methods similarly face barriers in scaling up to large p and/or n. There is a heavy computational price for learning the number of latent variables, the distribution of the latent variables, and the mapping functions while maintaining identifiability restrictions.

Recently, Yang and Dunson (2013) show that this computational burden can be largely bypassed by using usual Gaussian process (GP) regression without attempting to learn the mapping to the lower-dimensional subspace. They showed that when the features lie on a d-dimensional manifold embedded in the p-dimensional feature space with $d \ll p$ and the regression function is not highly smooth, the optimal rate can be obtained using GP regression with a squared exponential covariance in the original high-dimensional feature

space. This is an exciting theoretical result, which provides motivation for the approach in this article, which is focused on scalable Bayesian nonparametric regression in large p settings. For broader applicability than Yang and Dunson (2013), we accommodate features that are contaminated by noise and hence do not lie exactly on a low-dimensional manifold. In addition, we facilitate computational efficiency by bypassing MCMC and reducing matrix inversion bottlenecks via random projections. Sensitivity to the random projection and to tuning parameters is reduced through the use of Bayesian model averaging. The proposed approach that accommodates all these features is coined as the *compressed Gaussian process* (CGP).

Snelson and Ghahramani (2012) also considered manifold regression for big data, comprising feature vectors via pre-multiplying with a short and fat projection matrix. Their approach involves estimating a total of $(M+p)m$ parameters in a feature compression matrix and input points, with M the number of input points, leading to intractability as p increases. We demonstrate substantial advantages of our random compression approach in Section 5 in terms of computational scalability and predictive performance. In addition, SG lacks theory guarantees, while we show that CGP has a minimax optimal adaptive convergence rate dependent only on the true manifold dimension (assumed small). Calandra et al. (2014) instead use a neural network-like mapping of the input space, requiring non-convex optimization in high-dimensions. Scaling to moderate n, such as $n \sim 5,000 - 10,000$, is problematic. Other manifold regression methods (see Bickel and Li, 2007; Aswani et al., 2011) either lack scalability even for moderate p and n, or fail to characterize predictive uncertainties.

Section 2 proposes the model and computational approach in large p settings. Section 3 describes extensions to moderately large n, and Section 4 develops theoretical justification. Section 5 contains simulation examples relative to state-of-the-art competitors. Section 6 presents an image data application, and Section 7 concludes the paper with a discussion.

2. Compressed Gaussian process regression

This section details out compressed Gaussian process model with the associated prior and posterior distributions of the parameters.

2.1 Model

For subjects $i = 1, \ldots, n$, let $y_i \in \mathcal{Y}$ denote a response with associated features $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})' = (z_{i1}, \ldots, z_{ip})' + (\delta_{i1}, \ldots, \delta_{ip})' = \boldsymbol{z}_i + \boldsymbol{\delta}_i$, $\boldsymbol{z}_i \in \mathcal{M}$, $\boldsymbol{\delta}_i \in \mathcal{R}^p$, where $\mathcal{M}$ is a d-dimensional manifold embedded in the ambient space $\mathcal{R}^p$. We assume that the response $y \in \mathcal{Y}$ is continuous. The measured features do not fall exactly on the manifold $\mathcal{M}$ but are corrupted by noise. We assume a compressed nonparametric regression model

$$y_i = \mu\big(\boldsymbol{\Psi}\boldsymbol{x}_i\big) + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2), \tag{1}$$

with the residuals modeled as Gaussian with variance σ^2, though other distributions including heavy-tailed ones can be accommodated. $\boldsymbol{\Psi}$ is an $m \times p$ matrix that compresses p-dimensional features to dimension m. Following a Bayesian approach, we choose a prior distribution for the regression function μ and residual variance σ^2, while randomly generating $\boldsymbol{\Psi}$ following precedence in the literature on feature compression (Maillard and Munos, 2009; Fard et al., 2012; Guhaniyogi and Dunson, 2013). These earlier approaches differ from ours in focusing on parametric regression. We independently draw elements $\{\Psi_{ij}\}$ of $\boldsymbol{\Psi}$ from $N(0,1)$, and then normalize the rows using Gram-Schmidt orthogonalization.

We assume that $\mu \in \mathcal{H}_s$ is a continuous function belonging to $\mathcal{H}_s$, a Holder class with smoothness s. To allow μ to be unknown, we use a Gaussian process (GP) prior, $\mu \sim \text{GP}(0, \sigma^2\kappa)$ with the covariance function chosen to be squared exponential

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j; \lambda) = \exp\left(-\lambda||\boldsymbol{x}_i - \boldsymbol{x}_j||^2\right), \tag{2}$$

with λ a smoothness parameter and $||\cdot||^2$ the Euclidean norm. To additionally allow the residual variance σ^2 and smoothness λ to be unknown, we let

$$\sigma^2 \sim IG(a, b), \quad \lambda^d \sim Ga(a_0, b_0),$$

with $IG()$ and $Ga()$ denoting the inverse-gamma and gamma densities, respectively. The powered gamma prior for λ is motivated by the result of van der Vaart and van Zanten (2009) showing minimax adaptive rates of $n^{-s/(2s+p)}$ for a GP prior with squared exponential covariance and powered gamma prior. This is the optimal rate for nonparametric regression in the original p-dimensional ambient space. The rate can be improved to $n^{-s/(2s+d)}$ when $\boldsymbol{x}_i \in \mathcal{M}$, with $\mathcal{M}$ a d-dimensional manifold. Yang and Dunson (2013) shows that a GP prior with powered gamma prior on the smoothness can achieve this rate. In practice, replacing the powered gamma prior for λ with a gamma prior has essentially no impact on the results in examples we have considered.

In many applications, features may not lie exactly on $\mathcal{M}$ due to noise and corruption in the data. We apply random compression in (1) to de-noise the features, obtaining $\boldsymbol{\Psi}\boldsymbol{x}_i$ much more concentrated near a lower-dimensional subspace than the original $\boldsymbol{x}_i$. With this enhanced concentration, the theory in Yang and Dunson (2013) suggests excellent performance for an appropriate GP prior. In addition to de-noising, this approach has the major advantage of bypassing estimation of a geodesic distance along the unknown manifold $\mathcal{M}$ between any two data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$.

2.2 Posterior form

Let $\boldsymbol{\mu} = (\mu(\boldsymbol{\Psi}\boldsymbol{x}_1), ..., \mu(\boldsymbol{\Psi}\boldsymbol{x}_n))'$ and $\boldsymbol{K}_1 = (\kappa(\boldsymbol{\Psi}\boldsymbol{x}_i, \boldsymbol{\Psi}\boldsymbol{x}_j; \lambda))_{i,j=1}^n$. The prior distribution on μ, σ^2 induces a normal-inverse gamma (NIG) prior on $(\boldsymbol{\mu}, \sigma^2)$,

$$(\boldsymbol{\mu} \,|\, \sigma^2) \sim N(\boldsymbol{0}, \sigma^2\boldsymbol{K}_1),\ \sigma^2 \sim IG(a, b),$$

leading to a NIG posterior distribution for $(\boldsymbol{\mu}, \sigma^2)$ given $y, \boldsymbol{\Psi}\boldsymbol{x}, \lambda$. In the special case in which $a, b \to 0$, we obtain Jeffrey's prior and the posterior distribution is

$$\boldsymbol{\mu} \,|\, \boldsymbol{y} \sim t_n(\boldsymbol{m}, \boldsymbol{\Sigma}) \tag{3}$$

$$\sigma^2 \,|\, \boldsymbol{y} \sim IG(a_1, b_1), \tag{4}$$

where $a_1 = n/2$, $b_1 = \boldsymbol{y}'\left(\boldsymbol{K}_1 + \boldsymbol{I}\right)^{-1}\boldsymbol{y}/2$, $\boldsymbol{m} = \left[\boldsymbol{I} + \boldsymbol{K}_1^{-1}\right]^{-1}\boldsymbol{y}$, $\boldsymbol{\Sigma} = (2b_1/n)\left[\boldsymbol{I} + \boldsymbol{K}_1^{-1}\right]^{-1}$, and $t_\nu(\boldsymbol{m}, \boldsymbol{\Sigma})$ denotes a multivariate-t distribution with ν degrees of freedom, mean $\boldsymbol{m}$ and covariance $\boldsymbol{\Sigma}$.

Hence, the exact posterior distribution of $(\boldsymbol{\mu}, \sigma^2)$ conditionally on $(\boldsymbol{\Psi}, \lambda)$ is available analytically. The predictive of $\boldsymbol{y}^* = (y_1^*, ..., y_{n_{pred}}^*)'$ given $\boldsymbol{X}^* = \left(\boldsymbol{x}_1^{*'}, ..., \boldsymbol{x}_{n_{pred}}^{*'}\right)'$ and $\boldsymbol{\Psi}, \lambda$ for new n_{pred} subjects marginalizing out $(\boldsymbol{\mu}, \sigma^2)$ over their posterior distribution is available analytically as

$$\boldsymbol{y}^* | \boldsymbol{x}_1^*, ..., \boldsymbol{x}_{n_{pred}}^*, \boldsymbol{y} \sim t_n\left(\mu_{pred}, \sigma^2_{pred}\right), \tag{5}$$

where $\boldsymbol{K}_{pred} = \{\kappa(\boldsymbol{x}_i^*, \boldsymbol{x}_j^*; \lambda)\}_{i,j=1}^{n_{pred}}$, $\boldsymbol{K}_{pred,1} = \{\kappa(\boldsymbol{x}_i^*, \boldsymbol{x}_j; \lambda)\}_{i=1,j=1}^{i=n_{pred},j=n}$, $\boldsymbol{K}_{1,pred} = \boldsymbol{K}'_{pred,1}$, $\mu_{pred} = \boldsymbol{K}_{pred,1}\left(\boldsymbol{I} + \boldsymbol{K}_1\right)^{-1}\boldsymbol{y}$, $\sigma^2_{pred} = (2b_1/n)\left[\boldsymbol{I} + \boldsymbol{K}_{pred} - \boldsymbol{K}_{pred,1}\left\{\boldsymbol{I} + \boldsymbol{K}_1\right\}^{-1}\boldsymbol{K}_{1,pred}\right]$.

2.3 Model averaging

The approach described in the previous section can be used to obtain a posterior distribution for $\boldsymbol{\mu}$ and a predictive distribution for $\boldsymbol{y}^* = (y_1^*, ..., y_{n_{pred}}^*)$ given $\boldsymbol{X}^*$ for a new set of n_{pred} subjects *conditionally* on the $m \times p$ random projection matrix $\boldsymbol{\Psi}$ and the scaling parameter λ. To accomplish robustness with respect to the choice of $(\boldsymbol{\Psi}, \lambda)$ and the subspace dimension m, following Guhaniyogi and Dunson (2013), we propose to generate s random matrices having different m, s and λ from the marginal posterior distribution, $(\Psi^{(l)}, \lambda^{(l)})$, $l = 1, ..., s$, and then use model averaging to combine the results. To make matters more clear, let $\mathcal{M}_l$, $l = 1, \ldots, s$, represent (1) with m_l number of rows. Corresponding to the model $\mathcal{M}_l$, we denote $\boldsymbol{\Psi}$, λ, $\boldsymbol{\mu}$ and σ^2 by $\boldsymbol{\Psi}^{(l)}$, $\lambda^{(l)}$, $\boldsymbol{\mu}^{(l)}$ and $\sigma^{2(l)}$ respectively. Given $\boldsymbol{\Psi}^{(l)}$, we draw a few $\lambda_1, ..., \lambda_k$ randomly from $U(3/dmax, 3/dmin)$ where $dmax = \max_{i,j} ||\boldsymbol{x}_i - \boldsymbol{x}_j||^2$ and $dmin = \min_{i,j} ||\boldsymbol{x}_i - \boldsymbol{x}_j||^2$. Next we use the fact that the marginal posterior distribution of $\lambda | \boldsymbol{\Psi}^{(l)}, \boldsymbol{y}$ is given by

$$f(\lambda | \boldsymbol{y}, \boldsymbol{\Psi}^{(l)}) \propto \frac{1}{|\boldsymbol{K}_1 + \boldsymbol{I}|^{\frac{1}{2}}} \frac{2^{\frac{n}{2}}\Gamma(\frac{n}{2})}{\left[\boldsymbol{y}'\left(\boldsymbol{K}_1 + \boldsymbol{I}\right)^{-1}\boldsymbol{y}\right]^{\frac{n}{2}}(\sqrt{2\pi})^n} \times \pi(\lambda),$$

where $\pi(\lambda)$ is the prior distribution of λ. Clearly, a discrete approximation of $\lambda | \boldsymbol{\Psi}^{(l)}, \boldsymbol{y}$ is given by $\sum_{i=1}^k w_i \delta_{\lambda_i}$, where $w_i = \frac{f(\lambda_i | \boldsymbol{y}, \boldsymbol{\Psi}^{(l)})}{\sum_{j=1}^k f(\lambda_j | \boldsymbol{y}, \boldsymbol{\Psi}^{(l)})}$ and δ_{λ_i} is the Dirac Delta function at λ_i. Finally, $\lambda^{(l)}$ is drawn from $\sum_{i=1}^k w_i \delta_{\lambda_i}$. Although Section 4 shows minimax optimality of CGP with $\lambda^d \sim Gamma(a, b)$, we use $d = 1$ in practical implementations with no practical loss in cases we have considered.

Let $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_s\}$ denote the set of models corresponding to different random projections, $\mathcal{D} = \{(y_i, \boldsymbol{x}_i), i = 1, \ldots, n\}$ denote the observed data, and $\boldsymbol{y}^*$ denote the data for future subjects with features $\boldsymbol{X}^*$. Then, the predictive density of $\boldsymbol{y}^*$ given $\boldsymbol{X}^*$ is

$$f(\boldsymbol{y}^* | \boldsymbol{X}^*, \mathcal{D}) = \sum_{l=1}^s f(\boldsymbol{y}^* | \boldsymbol{X}^*, \mathcal{M}_l, \mathcal{D}) P(\mathcal{M}_l \,|\, \mathcal{D}), \tag{6}$$

where the predictive density of $\boldsymbol{y}^*$ given $\boldsymbol{X}^*$ under projection $\mathcal{M}_l$ is given in (9) and the posterior probability weight on projection $\mathcal{M}_l$ is

$$P(\mathcal{M}_l \,|\, \mathcal{D}) = \frac{P(\mathcal{D} \,|\, \mathcal{M}_l) P(\mathcal{M}_l)}{\sum_{h=1}^s P(\mathcal{D} \,|\, \mathcal{M}_h) P(\mathcal{M}_h)}.$$

Assuming equal prior weights for each random projection, $P(\mathcal{M}_l) = 1/s$. In addition, the marginal likelihood under $\mathcal{M}_l$ is

$$P(\mathcal{D} \,|\, \mathcal{M}_l) = \int P(\mathcal{D} \,|\, \mathcal{M}_l, \boldsymbol{\mu}^{(l)}, \sigma^{2(l)}) \pi(\boldsymbol{\mu}^{(l)}, \sigma^{2(l)}). \tag{7}$$

After a little algebra, one observes that for (1) with $(\boldsymbol{\mu} \,|\, \sigma^2) \sim N(\boldsymbol{0}, \sigma^2 \boldsymbol{K}_1)$, $\pi(\sigma^2) \propto \frac{1}{\sigma^2}$,

$$P(\mathcal{D} \,|\, \mathcal{M}_l) = \frac{1}{|\boldsymbol{K}_1 + \boldsymbol{I}|^{\frac{1}{2}}} \frac{2^{\frac{n}{2}}\Gamma(\frac{n}{2})}{\left[\boldsymbol{y}'\left(\boldsymbol{K}_1 + \boldsymbol{I}\right)^{-1}\boldsymbol{y}\right]^{\frac{n}{2}}(\sqrt{2\pi})^n}.$$

Plugging in the above expressions in (6), one obtains the posterior predictive distribution as a weighted average of t densities. Given that the computation over different sets of $\boldsymbol{\Psi}, \lambda$ are not dependent on each other, the calculations are embarrassingly parallel with a trivial expense for combining. The main computational expense comes from the inversion of an $n \times n$ matrix under the lth random projection. There is a vast literature on obtaining rapid approximations to such inversions under low rank assumptions. In the next section, we describe one such approach for enabling scaling to moderate n. Other recent methods can be easily substituted to scale to very large or massive n.

3. Scaling to moderately large n

Fitting (1) using model averaging requires computing inverses and determinants of covariance matrices of the order $n \times n$. In problems with even moderate n, this adds a heavy computational burden of the order of $O(n^3)$. Additionally, as dimension increases, matrix inversion becomes more unstable with the propagation of errors due to finite machine precision. This problem is further exacerbated if the covariance matrix is nearly rank deficient.

To address such issues, existing solutions rely on approximating $\mu(\cdot)$ by another process $\tilde{\mu}(\cdot)$, which is more tractable computationally. One popular approach constructs $\tilde{\mu}(\cdot)$ as a finite basis approximation via kernel convolution (Higdon, 2002) or kalman filtering (Wikle and Cressie, 1999). Alternatively, one can let $\tilde{\mu}(\cdot) = \mu(\cdot)\eta(\cdot)$, where $\eta(\cdot)$ is a Gaussian process having compactly supported correlation function that essentially makes the covariance matrix of $(\tilde{\mu}(\boldsymbol{x}_1), ..., \tilde{\mu}(\boldsymbol{x}_n))$ sparse (Kaufman et al., 2008), facilitating inversion through efficient sparse solvers.

Banerjee et al. (2008) proposes a low rank approach that imputes $\mu(\cdot)$ conditionally on a few knot-points, closely related to subset of regressor methods in machine learning (Smola and Schölkopf, 2000). Subsequently, Finley et al. (2009) in statistics and Snelson and Ghahramani (2006) in machine learning report bias in both variance and length-scale parameter estimation which affects predictive estimates for the proposed approaches (Banerjee et al., 2008; Smola and Schölkopf, 2000). They also suggest possible remedies for bias adjustments. To avoid sensitivity to knot selection in the low rank approaches, Banerjee et al. (2013) approximates $\mu(\cdot)$ using $\tilde{\mu}(\cdot) = E[\mu(\cdot) \,|\, \boldsymbol{\Phi}\mu(\boldsymbol{X})] + \epsilon_{\boldsymbol{\Phi}}(\cdot)$, with $\boldsymbol{\Phi}$ an $m \times n$, $m \ll n$ random matrix with $\Phi_{ij} \sim N(0, 1)$. $\epsilon_{\boldsymbol{\Phi}}(\boldsymbol{x})$ are independent feature specific noises with $\epsilon_{\boldsymbol{\Phi}}(\boldsymbol{x}) \sim N(\boldsymbol{0}, \mathrm{var}(\mu(\boldsymbol{x})) - \mathrm{var}(\tilde{\mu}(\boldsymbol{x})))$, which are introduced for bias correction similar to Finley et al. (2009). There is a parallel literature on nearest neighbor Gaussian processes which is built upon approximating a multivariate high dimensional Gaussian distribution

by a product of lower dimensional conditional distributions. Such an idea was first pursued by Vecchia (1988) and Stein et al. (2004), and has recently gained traction in the computer experiments literature (Gramacy and Apley, 2015) and in spatial geo-statistics (Emery, 2009; Stroud et al., 2014; Datta et al., 2014). Some of the recent versions of the this idea are found to be amenable to parallel computations as well.

We adapt Banerjee et al. (2013) from usual GP regression to our compressed manifold regression setting. In particular, let

$$y = \tilde{\mu}_{\boldsymbol{\Phi}}(\boldsymbol{\Psi x}) + \epsilon_{\boldsymbol{\Phi}}(\boldsymbol{\Psi x}) + \epsilon,\ \epsilon \sim N(0, \sigma^2), \tag{8}$$

where $\tilde{\mu}_{\boldsymbol{\Phi}}(\boldsymbol{\Psi x}) = E[\mu(\boldsymbol{\Psi x}) \,|\, \boldsymbol{\Phi}\mu(\boldsymbol{X\Psi}')]$, $\epsilon_{\boldsymbol{\Phi}}(\boldsymbol{\Psi x}) \,|\, \sigma^2 \sim N(0, \sigma^2_\epsilon(\boldsymbol{x}))$, $\sigma^2_\epsilon(\boldsymbol{x}) = \sigma^2\left[\kappa(\boldsymbol{\Psi x}, \boldsymbol{\Psi x}; \lambda) - (\boldsymbol{\Phi k_x})'\left\{\boldsymbol{\Phi K}_1\boldsymbol{\Phi}'\right\}^{-1}(\boldsymbol{\Phi k_x})\right]$ and $\boldsymbol{k_x} = (\kappa(\boldsymbol{\Psi x}, \boldsymbol{\Psi x}_1; \lambda), ..., \kappa(\boldsymbol{\Psi x}, \boldsymbol{\Psi x}_n; \lambda))'$. Denoting $\boldsymbol{H}_1 = diag(\boldsymbol{K}_1 - \boldsymbol{K}_1\boldsymbol{\Phi}'(\boldsymbol{\Phi K}_1\boldsymbol{\Phi}')^{-1}\boldsymbol{\Phi K}_1) + \boldsymbol{I}$ and $\boldsymbol{H}_2 = \boldsymbol{K}_1\boldsymbol{\Phi}'(\boldsymbol{\Phi K}_1\boldsymbol{\Phi}')^{-1}\boldsymbol{\Phi}$, marginal posterior distributions of μ and σ^2 are available in analytical forms

$$\boldsymbol{\mu} \,|\, \boldsymbol{y} \sim t_n(\boldsymbol{m}_{RGP}, \boldsymbol{\Sigma}_{RGP}),\quad \sigma^2 \,|\, \boldsymbol{y} \sim IG(a_2, b_2),$$

where $a_2 = n/2$, $b_2 = \boldsymbol{y}'\left(\boldsymbol{H}_1 + \boldsymbol{H}_2\boldsymbol{K}_1\right)^{-1}\boldsymbol{y}/2$, $\boldsymbol{m}_{RGP} = \left[\boldsymbol{H}_2'\boldsymbol{H}_1^{-1}\boldsymbol{H}_2 + \boldsymbol{K}_1^{-1}\right]^{-1}\boldsymbol{H}_2'\boldsymbol{H}_1^{-1}\boldsymbol{y}$, $\boldsymbol{\Sigma}_{RGP} = (2b_2/n)\left[\boldsymbol{H}_2'\boldsymbol{H}_1^{-1}\boldsymbol{H}_2 + \boldsymbol{K}_1^{-1}\right]^{-1}$. Owing to the special structure of $\boldsymbol{\Sigma}_{RGP}$ and $\boldsymbol{m}_{RGP}$, $n \times n$ matrix inversion can be efficiently achieved by Sherman-Woodbury-Morrison matrix inversion technique.

Attention now turns to prediction from (8). The predictive of $\boldsymbol{y}^* = (y_1^*, ..., y_{n_{pred}}^*)'$ given $\boldsymbol{X}^* = \left(\boldsymbol{x}_1^{*'}, ..., \boldsymbol{x}_{n_{pred}}^{*'}\right)'$ and $\boldsymbol{\Psi}, \lambda$ for new n_{pred} subjects marginalizing out $(\boldsymbol{\mu}, \sigma^2)$ over their posterior distribution is available analytically as

$$\boldsymbol{y}^* | \boldsymbol{x}_1^*, ..., \boldsymbol{x}_{n_{pred}}^*, \boldsymbol{y} \sim t_n\left(\mu_{pred}, \sigma^2_{pred}\right), \tag{9}$$

where $\boldsymbol{K}_{pred} = \{\kappa(\boldsymbol{x}_i^*, \boldsymbol{x}_j^*; \lambda)\}_{i,j=1}^{n_{pred}}$, $\boldsymbol{K}_{pred,1} = \{\kappa(\boldsymbol{x}_i^*, \boldsymbol{x}_j; \lambda)\}_{i=1,j=1}^{i=n_{pred},j=n}$, $\boldsymbol{K}_{1,pred} = \boldsymbol{K}_{pred,1}'$, $H_3 = \boldsymbol{I} + diag(K_{pred} - K_{pred,1}\boldsymbol{\Phi}'(\boldsymbol{\Phi K}_1\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}K_{1,pred})$, $\mu_{pred} = \boldsymbol{K}_{pred,1}\boldsymbol{H}_2'\left(\boldsymbol{H}_1 + \boldsymbol{H}_2\boldsymbol{K}_1\right)^{-1}\boldsymbol{y}$, $\sigma^2_{pred} = (2b_1/n)\left[\boldsymbol{H}_3 + \boldsymbol{K}_{pred,1}\boldsymbol{\Phi}'(\boldsymbol{\Phi K}_1\boldsymbol{\Phi}')^{-1}\boldsymbol{\Phi K}_{1,pred} - \boldsymbol{K}_{pred,1}\boldsymbol{H}_2'(\boldsymbol{H}_1 + \boldsymbol{H}_2\boldsymbol{K}_1)^{-1}\boldsymbol{H}_2\boldsymbol{K}_{1,pred}\right]$. Evaluating the above expression requires inverting matrices of order $m_{\boldsymbol{\Phi}} \times m_{\boldsymbol{\Phi}}$. Model averaging is again employed to limit sensitivity over the choices of $\boldsymbol{\Psi}, \lambda$. Following similar calculations as in Section 2.3, model averaging weights are found to be

$$P(\mathcal{D} \,|\, \mathcal{M}_l) = \frac{1}{|\boldsymbol{H}_2\boldsymbol{K}_1 + \boldsymbol{H}_1|^{\frac{1}{2}}} \frac{2^{\frac{n}{2}}\Gamma(\frac{n}{2})}{\left[\boldsymbol{y}'\left(\boldsymbol{H}_2\boldsymbol{K}_1 + \boldsymbol{H}_1\right)^{-1}\boldsymbol{y}\right]^{\frac{n}{2}}(\sqrt{2\pi})^n}.$$

Model averaging is performed on a wide interval of possible m values determined by the "compressed sample size" $m_{\boldsymbol{\Phi}}$ and p, analogous to Section 2.3.

Although we focus in this article on using the Banerjee et al. (2013) approach within CGP for scaling to moderately large n, alternative low rank or scalable approximations to Gaussian processes can be substituted essentially without complication. For example, there has been a recent emphasis on methods that break the data into exhaustive and mutually exclusive subsets (Parikh and Boyd, 2011), run computation separately for each subset and then combine the results; such methods have been applied to GPs (Deisenroth and Ng, 2015) and have complexity that scales as $O\left(\left(\frac{n}{K}\right)^3\right)$, where K is the number of subsets. Choosing K large enough, with this approach, one can compute CGP with moderate sized subset in each processor followed by combining inferences from different subsets. This can be further reduced by using low rank approximations to the GPs within each subset.

An important question that remains is how much information is lost in compressing the high-dimensional feature vector to a much lower dimension? In particular, one would expect to pay a price for the huge computational gains in terms of predictive performance or other metrics. We address this question in two ways. First we argue satisfactory theoretical performance in prediction in a large p asymptotic paradigm in Section 4. Then, we will consider practical performance in finite samples using simulated and real data sets.

4. Convergence analysis

This section provides theory supporting the excellent practical performance of the proposed method. In our context the feature vector $\boldsymbol{x}$ is assumed to be $\boldsymbol{x} = \boldsymbol{z} + \boldsymbol{\delta}$, $\boldsymbol{z} \in \mathcal{M}$, $\boldsymbol{\delta} \in \mathcal{R}^p$. Compressing the feature vector results in compressing $\boldsymbol{z}$ and the noise followed by their addition, $\boldsymbol{\Psi x} = \boldsymbol{\Psi z} + \boldsymbol{\Psi\delta}$. The following two directions are used to argue that compression results in near optimal inference.

(A) When features lie on a manifold a two stage estimation procedure (compression followed by a Gaussian process regression) leads to optimal convergence properties. This is used to show that using $\{\boldsymbol{\Psi z}_i\}_{i=1}^n$ as features in the Gaussian process regression yields the optimal rate of convergence.

(B) Noise compression through $\boldsymbol{\Psi}$ mitigates the deleterious effect of noise in $\boldsymbol{x}$ on the resulting performance.

Let $\mu_0(\cdot)$ and $\mu(\cdot)$ be the true and the fitted regression functions respectively. Define $\rho(\mu, \mu_0)^2 = \frac{1}{n}\sum_{i=1}^n(\mu(\boldsymbol{x}_i) - \mu_0(\boldsymbol{x}_i))^2$ as the distance between μ, μ_0 under a fixed design. When the design is random, let $\rho(\mu, \mu_0)^2 = \int_{\mathcal{M}}(\mu(\boldsymbol{x}) - \mu_0(\boldsymbol{x}))^2 F(d\boldsymbol{x})$, where F is the marginal distribution of the features. Denote $\Pi(\cdot|y_1, ..., y_n)$ to be the posterior distribution given $y_1, ..., y_n$. Then the interest lies in the rate at which the posterior contracts around μ_0 under the metric $\rho(\cdot, \cdot)$. This calls for finding a sequence $\{\zeta_n\}_{n\geq 1}$ of lower bounds such that

$$\Pi(\rho(\mu, \mu_0) > \zeta_n \,|\, y_1, ..., y_n) \to 0,\ \text{as } n \to \infty. \tag{10}$$

Definition: Given two manifolds $\mathcal{M}$ and $\mathcal{N}$, a differentiable map $f : \mathcal{M} \to \mathcal{N}$ is called a diffeomorphism if it is a bijection and its inverse $f^{-1} : \mathcal{N} \to \mathcal{M}$ is differentiable. If these functions are r times continuously differentiable, f is called a C^r-diffeomorphism.

Our analysis builds on the following result (Theorem 2.3 in Yang and Dunson (2013)).

Theorem 1 *Assume $\mathcal{M}$ is a d dimensional $\mathcal{C}^{r_1}$ compact sub-manifold of $\mathcal{R}^p$. Let $G : \mathcal{M} \to \mathcal{R}^p$ be the embedding map so that $G(\mathcal{M}) \simeq \mathcal{M}$. Further assume $T : \mathcal{R}^p \to \mathcal{R}^m$ is a dimensionality reducing map s.t. the restriction $T_{\mathcal{M}}$ of T on $G(\mathcal{M})$ is a $\mathcal{C}^{r_2}$-diffeomorphism*

onto its image. Then for any $\mu_0 \in \mathcal{C}^s$ *with* $s \leq \min\{2, r_1 - 1, r_2 - 1\}$*, a Gaussian process prior on* μ *with features* $\{T(\boldsymbol{z}_i)\}_{i=1}^n$*,* $\boldsymbol{z}_i \in \mathcal{M}$*, leads to a posterior contraction rate at least* $\zeta_n = n^{-s/(2s+d)} \log(n)^{d+1}$*.*

This is a huge improvement upon the minimax optimal adaptive rate of $n^{-s/(2s+p)}$ without the manifold structure in the features. We use the above result in our context. Define the linear transformation $T(\boldsymbol{z}) = \boldsymbol{\Psi}\boldsymbol{z}$. Using properties of random projection matrix, we have that, given $\kappa \in (0,1)$, if the projected dimension $m > O(\frac{m}{\kappa^2}\log(Cp\kappa^{-1})\log(\phi_n^{-1}))$ then with probability greater than $1 - \phi_n$, the following relationship holds for every point $\boldsymbol{z}_i, \boldsymbol{z}_j \in \mathcal{M}$,

$$(1-\kappa)\sqrt{\frac{m}{p}}||\boldsymbol{z}_i - \boldsymbol{z}_j|| < ||T(\boldsymbol{z}_i) - T(\boldsymbol{z}_j)|| < (1+\kappa)\sqrt{\frac{m}{p}}||\boldsymbol{z}_i - \boldsymbol{z}_j||, \tag{11}$$

implying that T is a diffeomorphism onto its image with probability greater than $(1 - \phi_n)$. Define $\mathcal{A}_n = \{\text{Equation 11 holds}\}$ so that $P(\mathcal{A}_n) > 1 - \phi_n$.

$$\begin{aligned}
\Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n) &= \Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n, \mathcal{A}_n) P(\mathcal{A}_n) \\
&+ \Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n, \mathcal{A}_n') P(\mathcal{A}_n') \\
&< \Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n, \mathcal{A}_n) + P(\mathcal{A}_n') \\
&< \Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n, \mathcal{A}_n) + \phi_n.
\end{aligned}$$

On $\mathcal{A}_n$, T is a diffeomorphism. Therefore, Theorem 1 implies that with features $\{T(\boldsymbol{z}_i)\}_{i=1}^n$ $\Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n, \mathcal{A}_n) \to 0$. Finally, assuming $\phi_n \to 0$ yields $\Pi(d(\mu,\mu_0) > \zeta_n | y_1, ..., y_n) \to 0$ with features $\{T(\boldsymbol{z}_i)\}_{i=1}^n$. This proves (A).

Let $\boldsymbol{\Psi}^{(l)}$ be the l-th row of $\boldsymbol{\Psi}$, $l = 1, ..., m$. Denote $\boldsymbol{\Delta} = [\boldsymbol{\delta}_1 : \cdots : \boldsymbol{\delta}_n] \in \mathcal{R}^{p \times n}$ and assume $\boldsymbol{z}_i$ is the i-th row of $\boldsymbol{\Delta}$. Using Lemma 2.9.5 in Van der Vaart and Wellner (1996), we obtain

$$\sqrt{p}\sum_{j=1}^{p} \Psi_{lj}\boldsymbol{z}_j \to N(\boldsymbol{0}, \text{Cov}(\boldsymbol{z}_1)).$$

Therefore, $\sum_{j=1}^p \Psi_{lj}\boldsymbol{z}_j = O_p(p^{-1/2})$, reducing the magnitude of noise in the original features. Hence (B) is proved. Thus, even if noise exists, asymptotic performance of $\{T(\boldsymbol{x}_i)\}_{i=1}^n$ will be similar to $\{T(\boldsymbol{z}_i)\}_{i=1}^n$ in the GP regression (which by (A) has "optimal" asymptotic performance).

5. Simulation Examples

We assess the performance of compressed Gaussian process (CGP) regression in a number of simulation examples. We consider various numbers of features (p) and level of noise in the features (τ) to study their impact on the performance. In all the simulations out of sample predictive performance of the proposed CGP regression was compared to that of uncompressed Gaussian process (GP), BART (Bayesian Additive Regression Trees) Chipman et al. (2010), RF (Random Forests) Breiman (2001) and TGP (Treed Gaussian process) Gramacy and Lee (2008). Unfortunately, with massive number of features, traditional BART, RF and TGP are computationally prohibitive. Therefore, we consider compressed versions in

which we generate a single projection matrix to obtain a single set of compressed features, running the analysis with compressed features instead of original features. This idea leads to compressed versions of random forest (CRF), Bayesian additive regression tree (CBART) and Treed Gaussian process (CTGP). These methods entail faster implementation when the number of features is massive.

As a default in these analyses, we use $m = 60$, which seems to be a reasonable choice of upper bound for the dimension of the linear subspace to compress to. In addition, we implement two stage GP (2GP) where the p-dimensional features are projected into smaller dimension by using Laplacian eigenmap (Belkin and Niyogi, 2003; Guerrero et al., 2011) in the first stage and then a GP with projected features is fitted in the second stage. We also compared Lasso and partial least square regression (PLSR) to indicate advantages of our proposed method over linear regularizing methods. However, in presence of strong nonlinear relationship between the response and the features, Lasso and PLSR perform poorly and hence results for them are omitted.

When n is moderately large ~ 5000, to bypass heavy computational price associated with CGP for inverting an $n \times n$ matrix, we employ a low rank approximation of the compressed Gaussian process as described in Section 3. As an uncompressed competitor of CGP in settings with moderately large n, efficient Gaussian random projection technique Banerjee et al. (2013) is implemented. This is also referred to as the GP to avoid needless confusion. Along with GP, CBART and CRF are included as competitors. CTGP with moderately large n poses heavy computational burden and is, therefore, omitted.

As a more scalable competitor, we employ the popular two stage technique of clustering the massive sample into a number of clusters followed by fitting simple model such as Lasso in each of these clusters. To facilitate clustering of high dimensional features in the first stage, we use the spectral clustering algorithm (Ng et al., 2001) described in Algorithm 1. Once observations are clustered, separate Lasso is fitted in each of these clusters. Hence-

Algorithm 1 Spectral Clustering Algorithm

Input: features $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ and the number of clusters required $n.clust$.

- Form the affinity matrix $\boldsymbol{A} \in \mathcal{R}^{n \times n}$ defined by $A_{ij} = \exp\left(-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2/2\sigma^2\right)$ if $i \neq j$, $A_{ii} = 0$, for some judicious choice of σ^2.
- Define $\boldsymbol{D}$ to be the diagonal matrix whose (i,i)-th entry is the sum of the elements in the i-th row of $\boldsymbol{A}$. Construct $\boldsymbol{L} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$.
- Find $\boldsymbol{s}_1, ..., \boldsymbol{s}_{n.clust}$ be the eigenvectors corresponding to the $n.clust$ largest eigenvalues of $\boldsymbol{L}$. Form the matrix $\boldsymbol{S} = [\boldsymbol{s}_1 : \cdots : \boldsymbol{s}_{n.clust}] \in \mathcal{R}^{n \times n.clust}$ by stacking the eigenvectors in column.
- Normalize so that each row of $\boldsymbol{S}$ has unit norm.
- Now treating each row of $\boldsymbol{S}$ as a point in $\mathcal{R}^{n.clust}$ cluster them into $n.clust$ clusters via $K - means$ clustering.
- Finally assign $\boldsymbol{x}_i$ in cluster j if the i th row of $\boldsymbol{S}$ goes to cluster j.

forth, we refer to this procedure as distributed supervised learning (DSL). Along with the above methods, for large moderately n, we also implement the Bayesian analogue of sparse

Gaussian process with dimension reduction (Snelson and Ghahramani, 2012), referred to as the SG method.

The model averaging step in CGP requires choosing a window over the possible values of m. When n is small, we adopt the choice suggested in Guhaniyogi and Dunson (2013) to have a window of $[\lceil 2log(p) \rceil, min(n,p)]$, which implies that the number of possible models to be averaged across is $s = min(n,p) - \lceil 2log(p) \rceil + 1$. When n is moderately large, we choose the window of $[\lceil 2log(p) \rceil, min(m_{\mathbf{\Phi}}, p)]$. The number of rows of $\mathbf{\Phi}$ is fixed at $m_{\mathbf{\Phi}} = 100$ for the simulation study with moderately large n. However, changing $m_{\mathbf{\Phi}}$ moderately does not alter the performance of CGP.

5.1 Manifold Regression on Swiss Roll

To provide some intuition for our model, we start with a concrete example where the distribution of the response is a nonlinear function of the coordinates along a swissroll, which is embedded in a high dimensional ambient space. To be more specific, we sample manifold coordinates, $t \sim U(\frac{3\pi}{2}, \frac{9\pi}{2})$, $h \sim U(0,3)$. A high dimensional feature $\boldsymbol{x} = (x_1, ..., x_p)$ is then sampled following

$$x_1 = t\cos(t) + \delta_1,\ x_2 = h + \delta_2,\ x_3 = t\sin(t) + \delta_3,\ x_i = \delta_i,\ i \geq 4, \delta_1, .., \delta_p \sim N(0, \tau^2).$$

Finally responses are simulated to have nonlinear and non-monotonic relationship with the features

$$y_i = \sin(5\pi t) + h^2 + \epsilon_i,\ \epsilon_i \sim N(0, 0.02^2). \tag{12}$$

Clearly, $\boldsymbol{x}$ and y are conditionally independent given θ, h, which is the low-dimensional signal manifold. In particular, $\boldsymbol{x}$ lives on a (*noise corrupted*) swissroll embedded in a p-dimensional ambient space (see Figure 1(a)), but y is only a function of coordinates along the swissroll $\mathcal{M}$ (see Figure 1(b)).

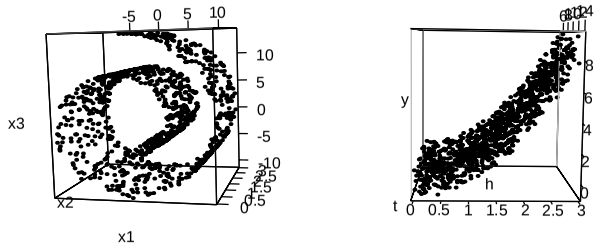
The geodesic distance between two points in a swiss roll can be substantially different from their Euclidean distance in the ambient space $\mathcal{R}^p$. For example, in Figure 1(c) two points joined by the line segment have much smaller Euclidean distance than geodesic distance. Theorem 1 in Section 4 guarantees optimal performance when the compact submanifold $\mathcal{M}$ is sufficiently smooth, so that the locally Euclidean distance serves as a good approximation of the geodesic distance. The Swiss roll presents a challenging set up for CGP, since points on $\mathcal{M}$ that are close in a Euclidean sense can be quite far in a geodesic sense.

To assess the impact of the number of features (p) and noise levels of the features (τ) on the performance of CGP, a number of simulation scenarios are considered in Table 1. For each of these simulation scenarios, we generate multiple datasets and present predictive inference such as mean squared prediction error (MSPE), coverage and lengths of 95% predictive intervals (PI) averaged over all replicates.

In our experiments, $\boldsymbol{y}$ and $\boldsymbol{X}$ are centered. To implement LASSO, we use `glmnet` (Friedman et al., 2009) package in R with the optimal tuning parameter selected through 10 fold cross validation. CRF, CBART and CTGP in R using `randomForest` (Liaw and Wiener, 2002), `BayesTree` (Chipman et al., 2009) and `tgp` (Gramacy, 2007) packages, respectively.

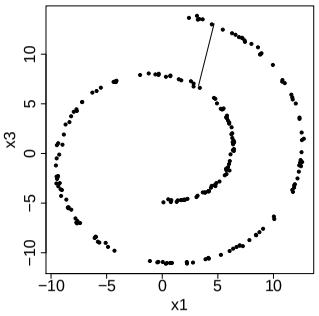
Simulation	sample size (n)	no. of features (p)	noise in the features (τ)
1	100	10,000	0.02
2	100	20,000	0.02
3	100	10,000	0.05
4	100	20,000	0.05
5	100	10,000	0.10
6	100	20,000	0.10

Table 1: Different Simulation settings for CGP.



(a) noise corrupted swiss roll

(b) response vs. x_1, x_2



(c) swiss roll shown in 2d

Figure 1: Simulated features and response on a *noisy* Swiss Roll, $\tau = 0.05$

5.1.1 MSPE Results

Predictive MSE for each of the simulation settings averaged over 50 simulated datasets is shown in Table 2. Subscripted values represent bootstrap standard errors for the averaged

MSPEs, calculated by generating 50 bootstrap datasets resampled from the MSPE values, finding the average MSPE of each, and then computing their standard error.

Table 2 shows that feeding randomly compressed features into any of the nonparametric methods leads to good predictive performance, while Lasso fails to improve much upon the null model (not shown here). For both $p = 10,000$ and $20,000$, when the swiss roll is corrupted with low noise, CGP performs significantly better than GP, while CBART and CRF provide competitive performance with GP. Increasing noise in the features results in deteriorating performances for all the competitors. CGP is an effective tool to reduce the effect of noise in the features, but at a *tipping point* (depending on n) noise distorts the manifold too much, and CGP starts performing similarly to GP. CRF and CTGP perform much worse than CGP in high noise scenarios, while CBART produces competitive performance. Two stage GP (2GP) performs much worse than all the other competitors; perhaps the two stage procedure is considerably more sensitive to noise. Increasing number of features does not alter MSPE for CGP significantly in presence of low noise, consistent with asymptotic results showing posterior convergence rates depend on the intrinsic dimension of $\mathcal{M}$ instead of p when features are concentrated close to $\mathcal{M}$. In the next section, we will study these aspects with increasing sample size and noise in the features.

	Noise in the feature			
		.02	.05	.10
$p = 10000$	CGP	$4.09_{0.08}$	$5.49_{0.08}$	$7.03_{0.11}$
	GP	$4.71_{0.10}$	$5.63_{0.10}$	$7.09_{0.12}$
	CRF	$4.13_{0.11}$	$6.23_{0.09}$	$7.44_{0.11}$
	CBART	$3.73_{0.13}$	$6.14_{0.10}$	$7.34_{0.12}$
	CTGP	$4.24_{0.14}$	$7.13_{0.11}$	$7.72_{0.14}$
	2GP	$5.72_{0.15}$	$6.55_{0.13}$	$7.85_{0.16}$
$p = 20000$	CGP	$4.43_{0.07}$	$6.21_{0.10}$	$7.28_{0.13}$
	GP	$4.86_{0.07}$	$6.25_{0.12}$	$7.18_{0.12}$
	CRF	$5.06_{0.11}$	$6.81_{0.11}$	$7.47_{0.13}$
	CBART	$4.84_{0.15}$	$6.77_{0.11}$	$7.33_{0.11}$
	CTGP	$5.59_{0.11}$	$7.40_{0.11}$	$7.51_{0.15}$
	2GP	$6.05_{0.10}$	$6.69_{0.13}$	$7.09_{0.19}$

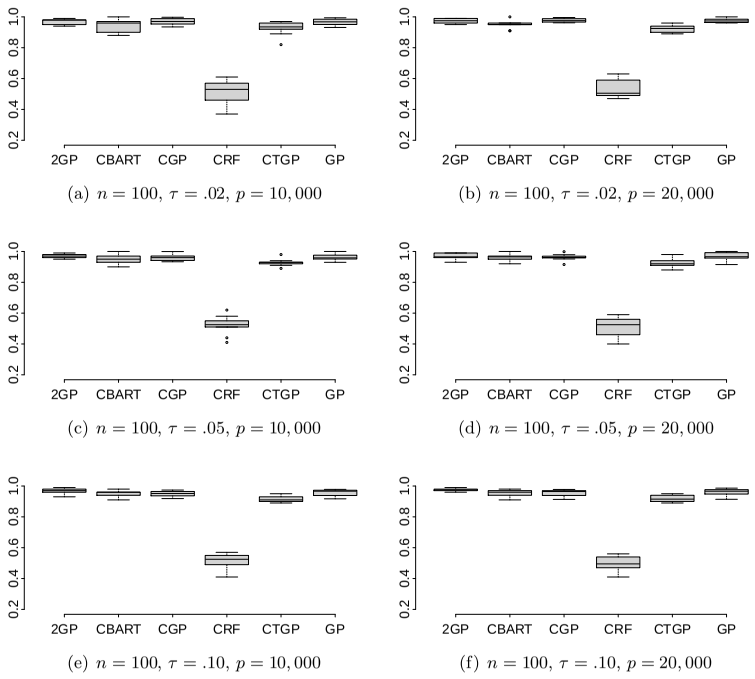
Table 2: Performance comparisons for competitors in terms of mean squared prediction errors (MSPE)

5.1.2 Coverage and Length of PIs

To assess if CGP is well calibrated in terms of uncertainty quantification, we compute coverage and length of 95% predictive intervals (PI) of CGP along with all the competitors. Although most frequentist methods such as CRF are unable to provide such coverage probabilities in producing point estimates, we present a measure of predictive uncertainty for those methods following the popular two stage plug-in approach, (i) estimate the regression function in the first stage; (ii) construct 95% PI based on the normal distribution centered

on the predictive mean from the regression model with variance equal to the estimated variance in the residuals. Boxplots for coverage probabilities in all the simulation cases are presented in Figure 2. Figure 3 presents median lengths of the 95% predictive intervals.

Both these Figures demonstrate that in all the simulation scenarios CGP, uncompressed GP, 2GP and CBART result in predictive coverage of around 95%, while CRF suffers from severe under-coverage. The gross under-coverage of CRF is attributed to the overly narrow predictive intervals. Additionally, CTGP shows some under-coverage, with shorter predictive intervals than CGP, GP, 2GP or CBART. CGP turns out to be an excellent choice among all the competitors in fairly broad simulation scenarios. We consider larger sample sizes and high noise scenarios in the next subsection.



(a) $n = 100$, $\tau = .02$, $p = 10,000$ (b) $n = 100$, $\tau = .02$, $p = 20,000$

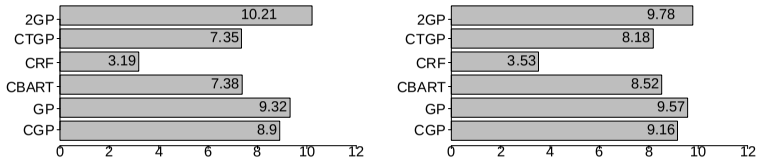
(c) $n = 100$, $\tau = .05$, $p = 10,000$ (d) $n = 100$, $\tau = .05$, $p = 20,000$

(e) $n = 100$, $\tau = .10$, $p = 10,000$ (f) $n = 100$, $\tau = .10$, $p = 20,000$

Figure 2: coverage of 95% PI's for CGP, GP, CBART, CTGP, CRF, 2GP

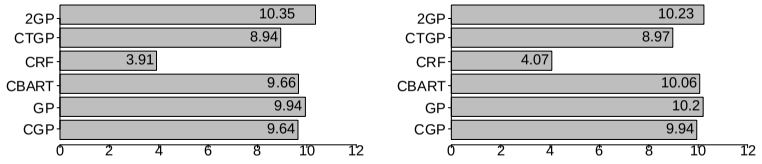
5.2 Manifold Regression on Swiss roll for Larger Samples

To assess how the relative performance of CGP changes for larger sample size, we implement manifold regression on swiss roll using methodologies developed in Section 3. For this simulation example, a data generation scheme similar to Section 5.1 is used. Ideally, larger



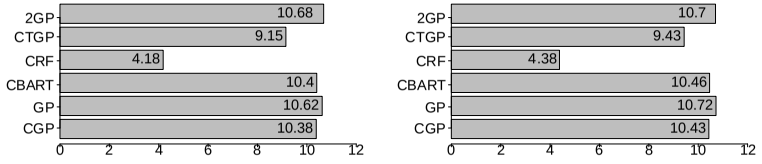
(a) $n = 100$, $\tau = .02$, $p = 10,000$

(b) $n = 100$, $\tau = .02$, $p = 20,000$



(c) $n = 100$, $\tau = .05$, $p = 10,000$

(d) $n = 100$, $\tau = .05$, $p = 20,000$



(e) $n = 100$, $\tau = .10$, $p = 10,000$

(f) $n = 100$, $\tau = .10$, $p = 20,000$

Figure 3: lengths of 95% PI's for CGP, GP, CBART, CRF, CTGP, 2GP

sample size should lead to better predictive performance. Therefore, one would expect more accurate prediction even with higher degree of noise in the features for larger sample size, as long as there is sufficient signal in the data. To accommodate higher signal than in Section 5.1, we simulate manifold coordinates as $t \sim U(\frac{3\pi}{2}, \frac{9\pi}{2})$, $h \sim U(0, 5)$ and sample responses as per (12). We also increase noise variability in the features for all the simulation settings. Simulation scenarios are described in Table 3.

MSPE of all the competing methods are calculated along with their bootstrap standard errors and presented in Table 4. Results in Table 4 provide more evidence supporting our conclusion in Section 5.1. With smaller noise variance, CGP along with other compressed methods outperform uncompressed GP and 2GP. However, when τ exceeds a certain limit, the manifold structure is more and more distorted, with performance of all the competitors worsening. In particular with increasing noise, performance of CGP and GP start becoming more comparable. On the other hand, SG method faces computational issues for $p \sim 10000 - 20000$ features. Therefore, we select only 500 features without disrupting the noisy manifold structure. Even with many fewer features, SG performs worse than CGP with

Simulation	sample size (n)	no. of features (p)	noise in the features (τ)
1	5,000	10,000	.03
2	5,000	20,000	.03
3	5,000	10,000	.06
4	5,000	20,000	.06
5	5,000	10,000	.10
6	5,000	20,000	.10

Table 3: Different Simulation settings for CGP for large n.

MSPE 46.3, 52.2 for $\tau = 0.03, 0.1$ respectively. Our investigation shows that the performance of SG is quite competitive when p is less than a few dozen. However, as p increases over a few hundreds, SG starts performing poorly. This is perhaps due to the fact that SG estimates a large number of poorly identifiable parameters resulting in inaccurate estimation. CGP with random compression of high dimensional features remarkably reduces the number of parameters to be estimated. Comparing results from the last section it is quite evident that with large samples, CGP is able to perform well even with very large number of features and moderate variance of noise in the features. This shows the effectiveness of CGP for large p and moderately large n when features are close to lying on a low-dimensional manifold.

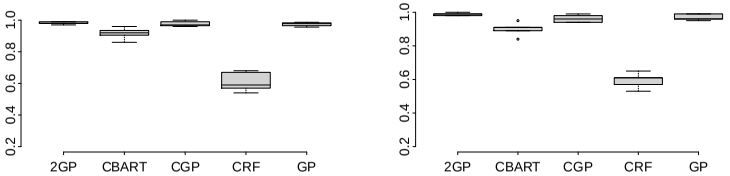
In all the simulation scenarios, DSL is the best performer in terms of MSPE, consistent with the routine use of DSL in large scale settings. However, the performance is extremely sensitive to the choice of clusters. In real data applications often inaccurate clustering leads to suboptimal performance, as will be seen in the data analysis. Additionally, we are not just interested in obtaining a point prediction approach, but want to obtain methods that provide an accurate characterization of predictive uncertainty. With this in mind, we additionally examine coverage probabilities and lengths of 95% predictive intervals (PIs). Boxplots for coverage probabilities of 95% PI's are presented in Figure 4. Figure 5 presents

		Noise in the feature		
		.03	.06	.10
$p = 10,000$	CGP	$0.56_{0.06}$	$1.06_{0.03}$	$2.18_{0.08}$
	GP	$2.05_{0.32}$	$2.37_{0.35}$	$3.35_{0.42}$
	CRF	$1.05_{0.10}$	$2.16_{0.11}$	$3.52_{0.09}$
	CBART	$0.69_{0.07}$	$1.72_{0.11}$	$2.79_{0.13}$
	DSL	$0.50_{0.07}$	$0.52_{0.03}$	$0.50_{0.03}$
	2GP	$3.78_{0.31}$	$3.95_{0.41}$	$4.05_{0.38}$
$p = 20,000$	CGP	$1.17_{0.048}$	$2.11_{0.107}$	$2.57_{0.222}$
	GP	$1.98_{0.418}$	$2.33_{0.321}$	$2.78_{0.330}$
	CRF	$1.46_{0.070}$	$2.76_{0.224}$	$3.88_{0.224}$
	CBART	$1.22_{0.092}$	$2.53_{0.151}$	$3.84_{0.192}$
	DSL	$0.48_{0.015}$	$0.45_{0.014}$	$0.57_{0.078}$
	2GP	$3.84_{0.581}$	$4.10_{0.370}$	$4.53_{0.481}$

Table 4: $MSPE \times 0.1$ along with the bootstrap $sd \times 0.1$ for all the competitors

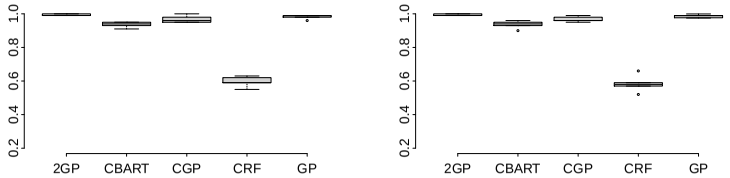
lengths of 95% prediction intervals for all the competitors. As expected, CGP, GP, 2GP

and CBART demonstrate better performance in terms of coverage. However, in low noise cases CGP and CBART achieve similar coverage with a two fold reduction in the length of PIs compared to GP or 2GP. CRF, like in the previous section, shows under-coverage with narrow predictive intervals. The predictive interval for CGP is found to be marginally wider than CBART with comparable coverage. With high noise, it becomes intractable to recover the manifold structure and hence performance is affected for all the competitors. It is observed that with high noise all approaches tend to have wider predictive intervals. DSL presents overly narrow predictive intervals (not shown here) yielding severe under-coverage.



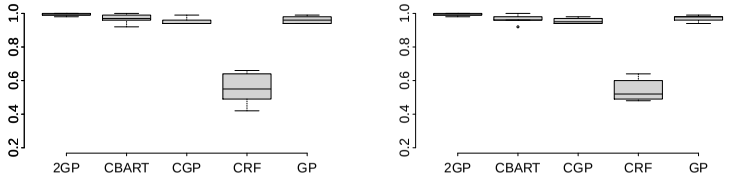
(a) $n = 5,000$, $p = 10,000$, $\tau = 0.03$

(b) $n = 5,000$, $p = 20,000$, $\tau = 0.03$



(c) $n = 5,000$, $p = 10,000$, $\tau = 0.06$

(d) $n = 5,000$, $p = 20,000$, $\tau = 0.06$



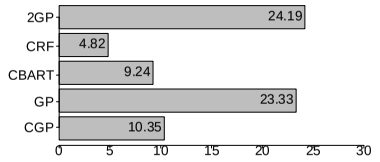
(e) $n = 5,000$, $p = 10,000$, $\tau = 0.10$

(f) $n = 5,000$, $p = 20,000$, $\tau = 0.10$

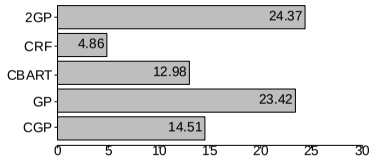
Figure 4: coverage of 95% PI's for CGP, GP, CRF, CBART, 2GP

5.3 Computation Time

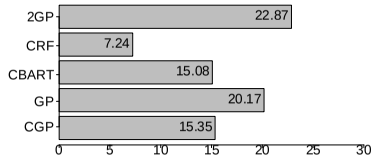
One of the major motivations in developing CGP was to improve computational scalability to large p settings. Clearly, the computational time for nonparametric estimation methods such as BART, TGP or RF applied to the original data will become notoriously prohibitive for large p, and hence we focus on comparisons with more scalable methods. The approach of applying BART, RF and TGP to the compressed features, which is employed in CBART,



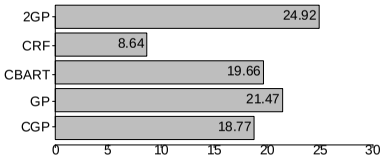
(a) $n = 5,000$, $\tau = .03$, $p = 10,000$



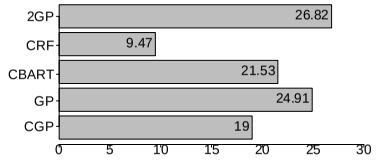
(b) $n = 5,000$, $\tau = .03$, $p = 20,000$



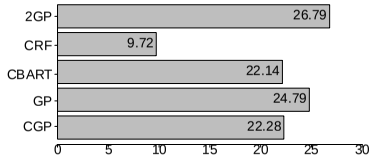
(c) $n = 5,000$, $\tau = .06$, $p = 10,000$



(d) $n = 5,000$, $\tau = .06$, $p = 20,000$



(e) $n = 5,000$, $\tau = .10$, $p = 10,000$



(f) $n = 5,000$, $\tau = .10$, $p = 20,000$

Figure 5: lengths of 95% PI's for CGP, GP, CBART, CRF, 2GP

CRF and CTGP respectively, is faster to implement. Using R code in a standard server, the computing time for 2,000 iterations of CBART for $n = 100$ and $p = 10,000, 20,000$ are only $7.21, 8.36$ seconds, while CGP has run time of $7.48, 8.05$ seconds, respectively. Increasing n moderately, we find CBART and CGP have similar run time. CRF is a bit faster than both of them, while CTGP has run time $37.64, 38.33$ seconds for $p = 10,000, 20,000$ respectively. For moderate n, 2GP is found to have similar run time as CBART.

With large n, CTGP is impractically slow and hence omitted in the comparison. GP needs to calculate and store a distance matrix of p features. Apart from the storage bottleneck, the computational complexity is $O(n^2p)$. CGP instead proposes calculating and storing a distance matrix of m compressed features, with a computational complexity of $O(n^2m)$. Computation time for CGP additionally depends on a number of factors, (i) Gram Schmidt orthogonalization of m rows of $m \times p$ matrices, (ii) inverting an $m_{\mathbf{\Phi}} \times m_{\mathbf{\Phi}}$ matrix, (iii) multiplying $n \times p$ and $p \times m$ matrices. Along with these three steps, one requires multiplying $m_{\mathbf{\Phi}} \times n$ matrix $\mathbf{\Phi}$ with $n \times n$ matrix K_1 at each MCMC iteration that incurs a computation complexity of order $n^2 m_{\mathbf{\Phi}}$. Typically the computation complex-

ity is dominated by n^2 and hence scaling with sample size is computationally feasible for about $n \sim 10000$ observations. For much larger n, one can resort to distributed GP based approaches as mentioned in Section 3. On the other hand, SPGP with dimensionality reduction (SG method) introduces exorbitantly large number of parameters even for moderate p.

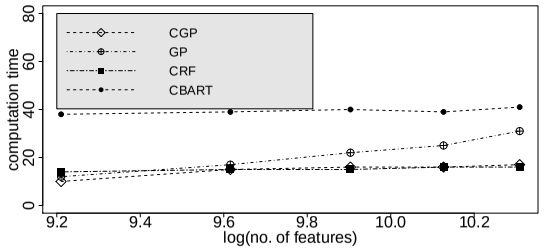
Figure 6 shows the computational speed comparison between CGP, GP, CBART and CRF for various n and p. Computational speed is recorded assuming existence of a number of processors on which parallelization can be executed. As n increases, CGP enjoys substantial computational advantage over competitors. The computational advantage is especially notable over CBART and GP. Run times of DSL are also recorded for $n = 5,000$ and $p = 10,000, 15,000, 20,000, 25,000, 30,000$ and they are $449, 599, 737, 945, 1158$ seconds, respectively. Alternatively, 2GP involves creating adjacency matrices followed by an eigen-decomposition of an $n \times n$ matrix. Both these steps are computationally demanding. We find 2GP takes $602, 723, 856, 983, 1108$ seconds to run for $n = 5000$ and $p = 10000, 15000, 20000, 25000, 30000$, respectively. Therefore, CGP can outperform even a simple two stage estimation procedure such as DSL in terms of computational speed.

6. Application to Face Images

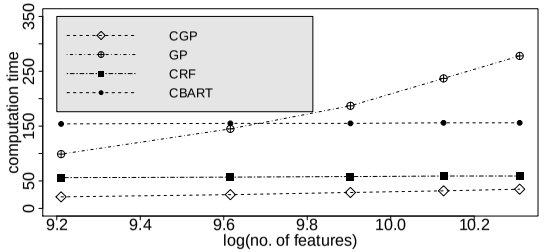
In our simulation examples, the underlying manifold is three dimensional and can be directly visualized. In this section we present an application in which both the dimension and the structure of the underlying manifold is unknown. The dataset consists of 698 images of an artificial face and is referred to as the *Isomap face data* (Tenenbaum et al., 2000). A few such representative images are presented in Figure 7. Each image is labeled with three different variables: illumination-, horizontal- and vertical-orientation. Two dimensional projections of the images are presented in the form of 64×64 pixel matrices. Intuitively, a limited number of additional features are needed for different views of the face. This is confirmed by the recent work of Levina and Bickel (2004); Aswani et al. (2011) where the intrinsic dimensionality is estimated to be small from these images. More details about the dataset can be found in `http://isomap.stanford.edu/datasets.html`.

We apply CGP and all the competitors to the dataset to assess relative performances. To set up the regression problem, we consider horizontal pose angles (vary in $[-75^0, 75^0]$) of the images, after standardization, as the responses. The features are taken $64 \times 64 = 4096$ dimensional vectorized images for each sample. To simulate more realistic situations, $N(0, \tau^2)$ noise is added to each pixel of the images, with varying τ, to make predictive inference more challenging from the noisy images. We carry out random splitting of the data into $n = 648$ training cases and $n_{pred} = 50$ test cases and run all the competitors to obtain predictive inference in terms of MSPE, length and coverage of 95% predictive intervals. To avoid spurious inference due to small validation set, this experiment is repeated 50 times.

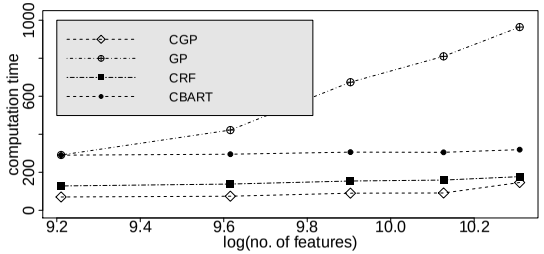
Table 5 presents MSPE for all the competing methods averaged over 50 experiments along with their standard errors computed using 100 bootstrap samples. Note that, because of the standardization, the null model yields MSPE 1. It is clear from Table 5 that CGP along with its compressed competitors explain a lot of variation in the response. DSL and 2GP are the worst performers in terms of MSPE. This is consistent with our experience that, in the presence of a complex and unknown manifold structure along with noise, DSL



(a) $n = 1,000$



(b) $n = 3,000$



(c) $n = 5,000$

Figure 6: Computational time in seconds for CGP, GP, CBART, CRF against log of the number of features.

can be unreliable relative to CGP which tends to be more robust to the type of manifold and noise level. GP also performs much worse than CGP and other compressed competitors especially in presence of small amount of noise in the features. As the noise in the features increases, performance of CGP and GP are found to be comparable. On the other hand

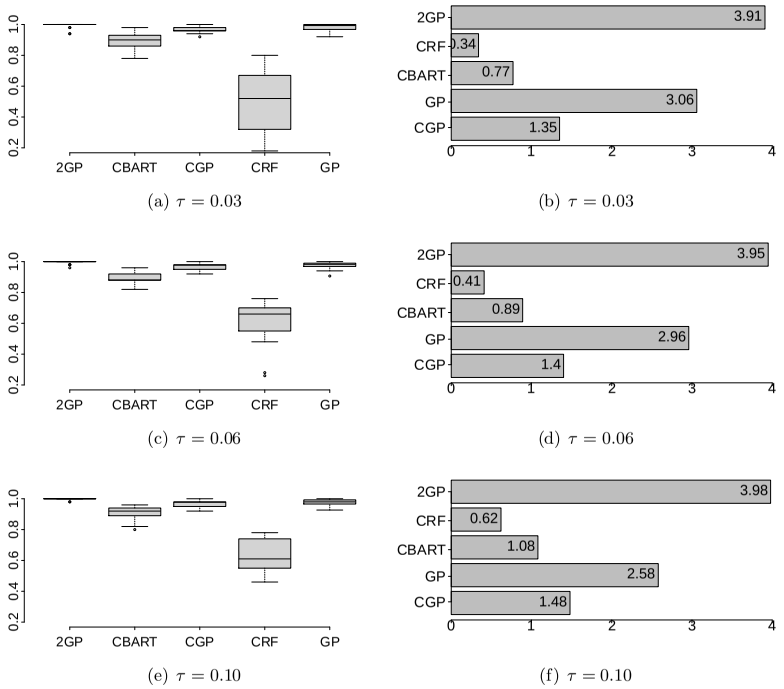
Figure 7: Representative images from the Isomap face data.

τ	CGP	GP	CBART	CRF	DSL	2GP
0.03	$0.07_{0.004}$	$0.85_{0.054}$	$0.07_{0.005}$	$0.06_{0.009}$	$0.70_{0.010}$	$0.98_{0.001}$
0.06	$0.08_{0.008}$	$0.75_{0.043}$	$0.09_{0.008}$	$0.10_{0.012}$	$0.78_{0.015}$	$0.94_{0.022}$
0.10	$0.09_{0.003}$	$0.68_{0.041}$	$0.11_{0.006}$	$0.11_{0.004}$	$0.83_{0.024}$	$0.98_{0.001}$

Table 5: MSPE and standard error (computed using 100 bootstrap samples) for all the competitors over 50 replications

SG implemented with only a subset of 500 features yields much worse performance (MSPE 0.97, 0.98 for $\tau = 0.1, 0.03$) respectively.

To see how well calibrated these methods are, Figure 8 provides coverage probabilities along with the lengths of predictive intervals for all the competitors. It is evident from the Figure that CGP, CBART, GP and 2GP yield excellent coverage. However, for CGP and CBART this coverage is achieved with much narrower predictive intervals compared to GP and 2GP. On the other hand, both CRF and DSL produce extremely narrow predictive intervals resulting in severe under-coverage. In fact for $\tau = 0.03, 0.06, 0.10$, length of 95% predictive intervals for DSL are $0.13, 0.19, 0.21$ respectively. Therefore, both in terms of MSPE and predictive coverage, CGP does a good job. More importantly, these results serve as a testimony of the robust performance demonstrated by compressed Bayesian nonparametric methods (CGP being one of them) even in the presence of unknown and complex manifold structure in the features.



(a) $\tau = 0.03$ (b) $\tau = 0.03$

(c) $\tau = 0.06$ (d) $\tau = 0.06$

(e) $\tau = 0.10$ (f) $\tau = 0.10$

Figure 8: Left panel: Boxplot for coverage of 95% predictive intervals over 50 replications; Right panel: Boxplot for length of 95% predictive intervals over 50 replications for CGP, GP, 2GP, CBART, CRF. In the left and right panels y-axis corresponds to the coverage and length respectively.

7. Discussion

The overarching goal of this article is to develop nonparametric regression methods that scale to large/very large p and/or moderately large $n \sim 5000$ when features lie on a *noise corrupted* manifold. The statistical and machine learning literature is somewhat limited in robust and flexible methods that can accurately provide predictive inference for large p with moderately large sample size, while taking into account the geometric structure. We develop a method based on nonparametric *low-rank* Gaussian process methods combined with random feature compression to accurately characterize predictive uncertainties quickly, bypassing the need to estimate the underlying manifold. The computational template exploits model averaging to limit sensitivity of the inference to the specific choices of the random

projection matrix $\mathbf{\Psi}$. The proposed method is also guaranteed to yield minimax optimal convergence rates.

There are many future directions motivated by our work. For example, the present work uses Banerjee et al. (2013) that is less suitable for massive n. It is quite straightforward to extend CGP to massive n by directly applying recently developed approaches for distributed computation in GP models (Deisenroth and Ng, 2015). Also the present work is not able to estimate the true dimensionality of the noise corrupted manifold. Arguably, a nonparametric method that can simultaneously estimate the intrinsic dimensionality of the manifold in the ambient space would improve performance both theoretically and practically. One possibility is to simultaneously learn the marginal distribution of the features, accounting for the low-dimensional structure. Other possible directions include adapting to massive streaming data where inference is to be made online. Although random compression both in n and p provides substantial benefit in terms of computation and inference, it might be worthwhile to learn the matrices $\mathbf{\Psi}$, $\mathbf{\Phi}$ while attempting to limit the associated computational burden.

References

Anil Aswani, Peter Bickel, and Claire Tomlin. Regression on manifolds: Estimation of the exterior derivative. *The Annals of Statistics*, 39(1):48–81, 2011.

Anjishnu Banerjee, David B Dunson, and Surya T Tokdar. Efficient Gaussian process regression for large datasets. *Biometrika*, 100(1):75–89, 2013.

Sudipto Banerjee, Alan E Gelfand, Andrew O Finley, and Huiyan Sang. Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):825–848, 2008.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Peter J Bickel and Bo Li. Local polynomial regression on unknown manifolds. *Lecture Notes-Monograph Series*, 54:177–186, 2007.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

Roberto Calandra, Jan Peters, Carl Edward Rasmussen, and Marc Peter Deisenroth. Manifold Gaussian processes for regression. *arXiv preprint arXiv:1402.5876*, 2014.

Minhua Chen, Jorge Silva, John Paisley, Chunping Wang, David Dunson, and Lawrence Carin. Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds. *Signal Processing, IEEE Transactions on*, 58(12):6140–6155, 2010.

Hugh Chipman, Robert McCulloch, and Maintainer Robert McCulloch. Package bayestree. 2009.

Hugh A Chipman, Edward I George, and Robert E McCulloch. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266–298, 2010.

Abhirup Datta, Sudipto Banerjee, Andrew O Finley, and Alan E Gelfand. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *arXiv preprint arXiv:1406.7343*, 2014.

Marc Peter Deisenroth and Jun Wei Ng. Distributed Gaussian processes. *arXiv preprint arXiv:1502.02843*, 2015.

Xavier Emery. The kriging update equations and their application to the selection of neighboring data. *Computational Geosciences*, 13(3):269–280, 2009.

Mahdi Milani Fard, Yuri Grinberg, Joelle Pineau, and Doina Precup. Compressed least-squares regression on sparse spaces. In *AAAI*, 2012.

Andrew O Finley, Sudipto Banerjee, Patrik Waldmann, and Tore Ericsson. Hierarchical spatial modeling of additive and dominance genetic variance for large spatial trial datasets. *Biometrics*, 65(2):441–451, 2009.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. glmnet: Lasso and elastic-net regularized generalized linear models. *R package version*, 1, 2009.

Robert B Gramacy. tgp: an r package for Bayesian nonstationary, semiparametric nonlinear regression and design by treed Gaussian process models. *Journal of Statistical Software*, 19(9):6, 2007.

Robert B Gramacy and Daniel W Apley. Local Gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics*, 24(2):561–578, 2015.

Robert B Gramacy and Herbert KH Lee. Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103 (483):1119–1130, 2008.

Ricardo Guerrero, Robin Wolz, and Daniel Rueckert. Laplacian eigenmaps manifold learning for landmark localization in brain mr images. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2011*, pages 566–573. Springer, 2011.

Rajarshi Guhaniyogi and David B Dunson. Bayesian compressed regression. *arXiv preprint arXiv:1303.0642*, 2013.

Dave Higdon. Space and space-time modeling using process convolutions. In *Quantitative methods for current environmental issues*, pages 37–56. Springer, 2002.

Cari G Kaufman, Mark J Schervish, and Douglas W Nychka. Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103(484):1545–1555, 2008.

Neil Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *The Journal of Machine Learning Research*, 6:1783–1816, 2005.

Elizaveta Levina and Peter J Bickel. Maximum likelihood estimation of intrinsic dimension. *Ann Arbor MI*, 48109:1092, 2004.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

Odalric-Ambrym Maillard and Rémi Munos. Compressed least-squares regression. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1213–1221, 2009.

Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Proceedings of Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press*, 14:849–856, 2001.

Garritt Page, Abhishek Bhattacharya, and David Dunson. Classification via Bayesian nonparametric learning of affine subspaces. *Journal of the American Statistical Association*, 108(501):187–201, 2013.

Neal Parikh and Stephen Boyd. Block splitting for large-scale distributed learning. In *Neural Information Processing Systems (NIPS), Workshop on Big Learning*. Citeseer, 2011.

Brian J Reich, Howard D Bondell, and Lexin Li. Sufficient dimension reduction via Bayesian mixture modeling. *Biometrics*, 67(3):886–895, 2011.

Alex J Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. pages 911–918, 2000.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18:1257, 2006.

Edward Snelson and Zoubin Ghahramani. Variable noise and dimensionality reduction for sparse gaussian processes. *arXiv preprint arXiv:1206.6873*, 2012.

Michael L Stein, Zhiyi Chi, and Leah J Welty. Approximating likelihoods for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66 (2):275–296, 2004.

Jonathan R Stroud, Michael L Stein, and Shaun Lysen. Bayesian and maximum likelihood estimation for Gaussian processes on an incomplete lattice. *arXiv preprint arXiv:1402.4281*, 2014.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Surya T Tokdar, Yu M Zhu, and Jayanta K Ghosh. Bayesian density regression with logistic gaussian process and subspace projection. *Bayesian analysis*, 5(2):319–344, 2010.

Aad W van der Vaart and J Harry van Zanten. Adaptive Bayesian estimation using a gaussian random field with inverse gamma bandwidth. *The Annals of Statistics*, 37(5B): 2655–2675, 2009.

AW Van der Vaart and JA Wellner. *Weak Convergence and Empirical Processes*. Springer, New York, 1996.

Aldo V Vecchia. Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 297–312, 1988.

Christopher K Wikle and Noel Cressie. A dimension-reduced approach to space-time kalman filtering. *Biometrika*, 86(4):815–829, 1999.

Yun Yang and David B Dunson. Bayesian manifold regression. *arXiv preprint arXiv:1305.0617*, 2013.

Yuchen Zhang, John C Duchi, and Martin J Wainwright. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *arXiv preprint arXiv:1305.5029*, 2013.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 8/14; Revised 10/15; Published 4/16

On the Characterization of a Class of Fisher-Consistent Loss Functions and its Application to Boosting

Matey Neykov MNEYKOV@PRINCETON.EDU
Department of Operations Research and Financial Engineering
Princeton University
Princeton, NJ 08540, USA

Jun S. Liu JLIU@STAT.HARVARD.EDU
Department of Statistics
Harvard University
Cambridge, MA 02138-2901, USA

Tianxi Cai TCAI@HSPH.HARVARD.EDU
Department of Biostatistics
Harvard University
Boston, MA 02115, USA

Editor: Alexander Rakhlin

Abstract

Accurate classification of categorical outcomes is essential in a wide range of applications. Due to computational issues with minimizing the empirical 0/1 loss, Fisher consistent losses have been proposed as viable proxies. However, even with smooth losses, direct minimization remains a daunting task. To approximate such a minimizer, various boosting algorithms have been suggested. For example, with exponential loss, the AdaBoost algorithm (Freund and Schapire, 1995) is widely used for two-class problems and has been extended to the multi-class setting (Zhu et al., 2009). Alternative loss functions, such as the logistic and the hinge losses, and their corresponding boosting algorithms have also been proposed (Zou et al., 2008; Wang, 2012). In this paper we demonstrate that a broad class of losses, including non-convex functions, achieve Fisher consistency, and in addition can be used for explicit estimation of the conditional class probabilities. Furthermore, we provide a generic boosting algorithm that is not loss-specific. Extensive simulation results suggest that the proposed boosting algorithms could outperform existing methods with properly chosen losses and bags of weak learners.

Keywords: Boosting, Fisher-Consistency, Multiclass Classification, SAMME

1. Introduction

Accurate classification of multi-class outcomes is essential in a wide range of applications. To construct an accurate classifier for the outcome $C \in \{1, ..., n\}$ based on a predictor vector $\mathbf{X}$, the target is often to minimize a misclassification rate, which corresponds to a 0/1 loss. We assume that the data $(C, \mathbf{X}^T)^T$ is generated from a fixed but unknown distribution $\mathscr{P}$. Specifically, one would aim to identify $\mathbf{f} = \{f_1(\cdot), ..., f_n(\cdot)\}$ that maximizes the misclassification rate

$$\mathbb{L}(\mathbf{f}) = \mathbb{E}\left[\mathbb{1}\{C \neq c_{\mathbf{f}}(\mathbf{X})\}\right] = \mathbb{P}\left\{C \neq c_{\mathbf{f}}(\mathbf{X})\right\}^{1}, \tag{1}$$

under the constraint $\sum_j f_j(\mathbf{X}) = 0$, where $\mathbb{1}(\cdot)$ is the indicator function and for any $\mathbf{f}$, $c_{\mathbf{f}}(\mathbf{X}) = \text{argmax}_j f_j(\mathbf{X})$. Obviously, $\mathbf{f}_{\text{Bayes}} = \{f_{\text{Bayes},j}(\cdot) = \mathbb{P}(C = j \mid \cdot) - n^{-1}, j = 1, ..., n\}$ minimizes (1). In practice, one may approximate the Bayes classifier $c_{\mathbf{f}_{\text{Bayes}}}(\cdot)$ by modeling $\mathbb{P}(C = j \mid \cdot)$ parametrically or non-parametrically. However, due to the curse of dimensionality and potential model mis-specification, such direct modeling may not work well when the underlying conditional risk functions are complex. On the other hand, due to both the discontinuity and the discrete nature of the empirical 0/1 loss, direct minimization is often computationally undesirable.

To overcome these challenges, many novel statistical procedures have been developed by replacing the 0/1 loss with a *Fisher consistent* loss ϕ such that its corresponding minimizer can be used to obtain the Bayes classifier. Lin (2004) showed that a class of smooth convex functions can achieve Fisher consistency (FC) for binary classification problems. Zou et al. (2008) further extended these results to the multi-class setting. Support vector machine (SVM) methods have been shown to yield Fisher consistent results for both binary and multi-class settings (Lin, 2002; Liu, 2007). Relying on these FC results, boosting algorithms for approximating the minimizers of the loss functions have also been proposed for specific choices of losses. Boosting algorithms search for the optimal solution by greedily aggregating a set of "weak-learners" $\mathcal{G}$ via minimization of an empirical risk, based on a loss function ϕ. The classical AdaBoost algorithm (Freund and Schapire, 1995) for example is based on the minimization of the exponential loss, $\phi(x) = e^{-x}$, using the forward stagewise additive modeling (FSAM) approach. Hastie et al. (2009) showed that the population minimizer of the AdaBoost algorithm corresponds to the Bayes rule $c_{\mathbf{f}_{\text{Bayes}}}(\cdot)$ for the two-class setting. Zhu et al. (2009) extended this algorithm and developed the Stagewise Additive Modeling using a Multi-class Exponential (SAMME) algorithm for the multi-class case.

Most existing work on Fisher consistent losses focuses on convex functions such as $\phi(x) = e^{-x}$ and $\phi(x) = |1 - x|_+$. However, there are important papers advocating the usage of non-convex loss functions, which we will briefly discuss here. Inspired by Shen et al. (2003), Collobert et al. (2006) explores SVM type of algorithms with the non-convex "ramp" loss instead of the typical "hinge" loss in order to speed up computations. Bartlett et al. (2006) consider the concept of "classification calibration" in the two-class case. Classification calibration of a loss can be understood as uniform FC, along all possible conditional probabilities on the simplex. They demonstrate that non-convex losses such as $1 - \tan(kx), k > 0$ can be classification calibrated in the two class case. More generally, Tewari and Bartlett (2007) extend classification calibration to the multiclass case, and provide elegant characterization theorems. We will draw a link between our work and the work of Tewari and Bartlett (2007) in Section 2.

Asymptotically, procedures such as the AdaBoost based on FC losses would lead to the optimal Bayes classifier, provided sufficiently large space of weak learner set $\mathcal{G}$. However, in finite samples, the estimated classifiers are often far from optimal, and the choice of the loss ϕ could greatly impact the accuracy of the resulting classifier. In this paper, we consider

1. Here the expectation and probability are both with respect to the unknown true distribution $\mathscr{P}$.

a broad class of loss functions that are potentially non-convex and demonstrate that the minimizer of these losses can lead to the Bayes rules for multi-category classification, and in fact can be used to explicitly restore the conditional probabilities. Moreover, we propose a generic algorithm leading to local minimizers of these potentially non-convex losses, which as we argue, can also recover the Bayes rule. The last observation has important consequences in practice, as global minimization of non-convex losses remains a challenging problem. On the other hand, non-convex losses, although not commonly used in the existing literature, could be more robust to outliers (Masnadi-Shirazi and Vasconcelos, 2008). The rest of the paper is organized as follows. In section 2 we detail the conditions for the losses and their corresponding FC results. In settings where the cost of misclassification may not be exchangeable between classes, we generalize our FC results to a weighted loss that accounts for differential costs. In section 3, we propose a generic boosting algorithm for approximating the minimizers and study some of its numerical convergence aspects. In section 4 we present simulation results and real data analysis comparing the performance of our proposed procedures to that of some existing methods including the SAMME. In addition in Section 4 we apply our proposed algorithms to identify subtypes of diabetic neuropathy with EMR data from the Partners. These numerical studies suggest that our proposed methods, with properly chosen losses, could potentially provide more accurate classification than existing procedures. Additional discussions are given section 5. Proofs of the theorems are provided in Appendix A.

2. Fisher Consistency for a general class of loss functions

In this section we characterize a broad class of loss functions which we deem relaxed Fisher consistent. This class encompasses previous classes of loss functions, provided in Zou et al. (2008), but also admits non-convex loss functions.

2.1 Fisher Consistency for 0/1 Loss

Suppose the training data available consists of N realizations of $(C, \mathbf{X}^T)^T$ drawn from $\mathscr{P}$, and let $\mathscr{D} = \{(C_i, \mathbf{X}_i^T)^T, i = 1, \ldots, N\}$. Moreover, we assume throughout that:

$$\min_{j \in \{1,\ldots,n\}} \mathbb{P}(C = j|\mathbf{X}) > 0 : \text{almost surely in } \mathbf{X}. \tag{2}$$

Assumption (2) states that any class C has a chance to be drawn for all $\mathbf{X}$, except on a set of measure 0, where determinism in the class assignment is allowed. For a given C, define a corresponding $n \times 1$ vector $\mathbf{Y}_C = (\mathbb{1}(C = 1), \ldots, \mathbb{1}(C = n))^T$. Under this notation, clearly $\mathbf{Y}_C^T \mathbf{f}(\mathbf{X}) = f_C(\mathbf{X})$. For identifiability the following constraint is commonly used in the existing literature (Lee et al., 2004; Zou et al., 2008; Zhu et al., 2009, e.g.) :

$$\sum_{j=1}^{n} f_j(\cdot) = 0. \tag{3}$$

To identify optimal $\mathbf{f}(\cdot)$ for classifying C based on $\mathbf{f}(\mathbf{X})$, we consider continuous loss functions ϕ as alternatives to the 0/1 loss and aim to minimize

$$\mathbb{L}_\phi(\mathbf{f}) = \mathbb{E}[\phi\{\mathbf{Y}_C^T \mathbf{f}(\mathbf{X})\}] = \mathbb{E}[\phi\{f_C(\mathbf{X})\}] = \sum_{j=1}^{n} \mathbb{E}[\phi\{f_j(\mathbf{X})\}\mathbb{P}(C = j \mid \mathbf{X})], \tag{4}$$

under the constraint (3). The loss function ϕ is deemed Fisher consistent Fisher consistent if the global minimizer (assuming it exists) $\mathbf{f}_\phi = \text{argmin}_{\mathbf{f}:\sum_j f_j = 0} \mathbb{L}_\phi(\mathbf{f})$ satisfies

$$c_{\mathbf{f}_\phi}(\mathbf{X}) = {}^2 c_{\mathbf{f}_{\text{Bayes}}}(\mathbf{X}). \tag{5}$$

Hence, with a Fisher consistent loss ϕ, the resulting $\text{argmax}_j \mathbf{f}_\phi(x)$ has the nice property of recovering the optimal Bayes classifier for the 0/1 loss. Clearly, the global minimizer $\mathbf{f}_\phi(x)$ also minimizes $\mathbb{E}[\phi\{f_C(\mathbf{X})\} \mid \mathbf{X} = x]$ for almost all x [3]. With a given data $\mathscr{D}$, we may approximate $\mathbf{f}_\phi$ by minimizing the empirical loss function

$$\widehat{L}_\phi(\mathbf{f}) = \frac{1}{N}\sum_{i=1}^{N} \phi\{\mathbf{Y}_{C_i}^T \mathbf{f}(\mathbf{X}_i)\} = \frac{1}{N}\sum_{i=1}^{N} \phi(f_{C_i}(\mathbf{X}_i)) = \frac{1}{N}\sum_{j=1}^{n}\sum_{i=1}^{N} \phi(f_j(\mathbf{X}_i))\mathbb{I}(C_i = j),$$

to obtain $\widehat{\mathbf{f}} = \text{argmin}_{\mathbf{f}:\sum_j f_j = 0} \widehat{L}_\phi(\mathbf{f})$.

Existing literature on the choice of ϕ focuses almost entirely on convex losses other than a few important exceptions (Bartlett et al., 2006; Tewari and Bartlett, 2007, e.g.). Here, we propose a general class of ϕ to include non-convex losses and generalize the concept of FC as we defined in (5). Specifically, we consider all continuous ϕ satisfying the following properties:

$$\phi(x) - \phi(x') \geq (g(x) - g(x'))k(x') \quad \text{for all } x \in \mathbb{R}, x' \in S = \{z \in \mathbb{R} : k(z) \leq 0\}, \tag{6}$$

where g and k are both strictly increasing continuous functions, with $g(0) = 1, \inf_{x\in\mathbb{R}} g(x) = 0, \sup_{x\in\mathbb{R}} g(x) = +\infty$, $k(0) < 0$ and $\sup_{z\in\mathbb{R}} k(z) \geq 0$. This suggests[4] that $\phi\{g^{-1}(\cdot)\}$ is continuously differentiable and convex on the set $g(S) = \{g(z) : z \in S\}$. However, ϕ itself is not required to be convex or differentiable. Extensively studied convex losses such as $\phi(x) = e^{-x}$ and $\phi(x) = \log(1 + e^{-x})$ both satisfy these conditions. For $\phi(x) = e^{-x}$, (6) would hold if we let $g(x) = e^x$ and $k(x) = -e^{-2x}$. For the logistic loss $\phi(x) = \log(1 + e^{-x})$, we may let $g(x) = e^{cx}$ and $k(x) = -\{ce^{cx}(1 + e^x)\}^{-1}$ for any positive constant $c > 0$. Alternatively, $g(x) = e^x(1 + e^x)/2$ and $k(x) = -2\{e^x(1 + e^x)(1 + 2e^x)\}^{-1}$ would also satisfy (6) for the logistic loss. Our class of losses also allows non-convex functions. For example, $\phi(x) = \log(\log(e^{-x} + e))$ is a non-convex loss and (6) holds if $g(x) = e^x$ and $k(x) = -\{e^x(e^{x+1} + 1)\log(e^{-x} + e)\}^{-1}$. On an important note, we would like to mention that all three examples above can be seen to fall into the general class of classification calibrated loss functions in the two class case, as defined by Bartlett et al. (2006) and hence are FC in the two-class case. We will see a more general statement relating condition (6) to the notion of classification calibration in the two class case (see Remark 2.2 below).

Next, we extend the FC property (5), to allow for more generic classification rules. For a loss function ϕ, if there exists a functional $\mathcal{H}$ such that the minimizer of (4) has the property:

$$\underset{j\in\{1,\ldots,n\}}{\text{argmax}}\, \mathcal{H}\{f_{\phi,j}(\mathbf{X})\} = c_{\mathbf{f}_{\text{Bayes}}}(\mathbf{X}), \tag{7}$$

2. Formally the "=" in (5) should be understood as "⊆". For the sake of simplicity, we keep this slight abuse of notation consistent throughout the paper.
3. Provided that this minimizer of $\mathbb{E}[\phi\{f_C(\mathbf{X})\} \mid \mathbf{X}]$ exists on a set of probability 1, and $\mathbb{E}[|\phi\{f_C(\mathbf{X})\}|] < \infty$ so that Fubini's theorem holds.
4. We provide a formal proof of this fact in Appendix A under Lemma A.1.

then we call it *relaxed* Fisher consistent (RFC). Obviously, the RFC property would still recover the Bayes classifier. Moreover FC losses are special cases of the RFC losses with an increasing $\mathcal{H}$.

We will now point out a connection between RFC and multiclass classification calibration as defined by Tewari and Bartlett (2007). Re-casting the definition of multiclass classification calibration to our framework, it requires that for any vector $\mathbf{w}$ on the simplex, the minimizer (assuming that it exists):

$$\widehat{\mathbf{F}}(\mathbf{w}) = \underset{\mathbf{F}:\sum_j F_j=0}{\operatorname{argmin}} \sum_{i=1}^{n} \phi(F_j) w_j \text{ satisfies } \underset{j\in\{1,\ldots,n\}}{\operatorname{argmax}} \mathcal{H}(\phi(\widehat{F}_j)) = \underset{j\in\{1,\ldots,n\}}{\operatorname{argmax}} w_j, \tag{8}$$

for some functional $\mathcal{H}$. In words, classification calibration ensures that regardless of the conditional distribution of $C|\mathbf{X}$, one can recover the Bayes rule. In contrast, RFC requires this to happen for the distribution at hand $C|\mathbf{X}$, for ($\mathscr{P}$ almost) all $\mathbf{X}$. This subtle but important distinction makes a difference. Example 4 in Tewari and Bartlett (2007) shows that if ϕ is positive and convex the conditions in (8) cannot be met for all vectors $\mathbf{w}$ on the simplex, when we have at least 3 classes. On the contrary, in the present paper we argue that in fact condition (8) remains plausible for both convex and non-convex losses, provided that we require that the points $\mathbf{w}$ are not allowed to be vertexes of the simplex (i.e. $w_j > 0$ for all j), which relates back to assumption (2).

The next result justifies that the proposed losses satisfying (6) are RFC with $\mathcal{H}(x) = H_\phi(x) \equiv g(x)k(x)$. We first present in Theorem 2.1 the property of a general constrained minimization problem, which is key to establishing the RFC.

Theorem 2.1 *For a loss ϕ satisfying (6), consider the optimization problem with some given $w_j > 0$:*

$$\min_{\mathbf{F}=(F_1,\ldots,F_n)^T} \sum_{j=1}^{n} \phi(F_j) w_j \quad \textit{under the constraint} \quad \prod_{j=1}^{n} g(F_j) = 1. \tag{9}$$

Assume that there exists a minimum denoted by $\widehat{\mathbf{F}} = (\widehat{F}_1, \ldots, \widehat{F}_n)^T$. Then the minimizer $\widehat{\mathbf{F}}$ must satisfy

$$H_\phi(\widehat{F}_j) w_j = \mathcal{C} \quad \textit{for some } \mathcal{C} < 0. \tag{10}$$

Moreover, if the function $H_\phi(\cdot)$ is strictly monotone there is a unique point with the property described above.

This result indicates that $H_\phi(\widehat{F}_j)$ is inversely proportional to the weight w_j. Now, consider $g(x) = \exp(x)$, $w_j = \mathbb{P}(C = j|\mathbf{X} = x)$, and $F_j = f_j(x)$, where we hold x . Then we can recover $c_{\mathbf{f}_{\text{Bayes}}}(x)$ by classifying C according to $\operatorname{argmax}_j \{-H_\phi(\widehat{F}_j)\}^{-1} = \operatorname{argmax}_j H_\phi(\widehat{F}_j)$ (the negative sign comes in because $\mathcal{C} < 0$), which implies that ϕ is RFC. Note that when $H_\phi(\cdot)$ is not increasing, Theorem 2.1 does not immediately imply that ϕ is a Fisher consistent loss according to definition (5), because the Bayes classifier need not be recovered by $\operatorname{argmax}_j \widehat{F}_j$. Nevertheless, we have the following:

Proposition 2.2 *Assume the same conditions as in Theorem 2.1. Then in addition to (10) we have:*

$$\underset{j\in\{1,\ldots,n\}}{\operatorname{argmax}} \widehat{F}_j = \underset{j\in\{1,\ldots,n\}}{\operatorname{argmax}} w_j,$$

and hence ϕ is also FC in the sense of (5).

The validity of Proposition 2.2 can be deduced from Theorem 2.1 and an application of Lemma 4 of Tewari and Bartlett (2007), but for completeness we include a simple standalone proof in Appendix A. While Proposition 2.2 states that ϕ is FC, Theorem 2.1 suggests that in addition to classification, one can recover conditional probabilities by calculating:

$$w_j = \frac{\{H_\phi(\widehat{F}_j)\}^{-1}}{\sum_{j=1}^{n} \{H_\phi(\widehat{F}_j)\}^{-1}}. \tag{11}$$

It is also worth noting here that the constraint in (9), generalizes the typical identifiability constraint (3), and the two coincide when $g(\cdot) = \exp(\cdot)$. We proceed by formulating a sufficient condition for the optimization problem in Theorem 2.1 to have a minimum without requiring the convexity or differentiability of ϕ.

Theorem 2.3 *The optimization problem in Theorem 2.1 has a minimum if either of the following conditions holds:*

i. ϕ is decreasing on the whole $\mathbb{R}$ and for all $c > 0$:

$$c\phi(g^{-1}(x)) + \phi(g^{-1}(x^{1-n})) \uparrow +\infty, \ \textit{as } x \downarrow 0, \tag{12}$$

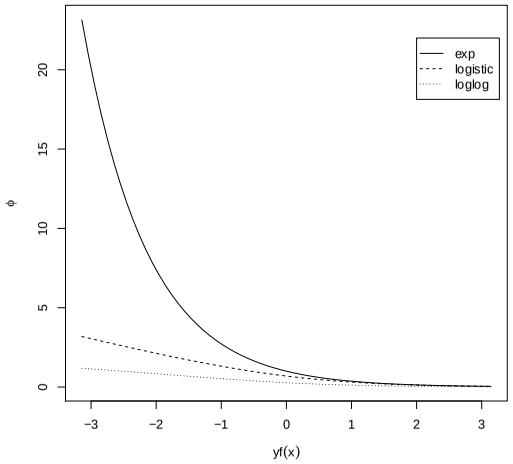
ii. ϕ is not decreasing on the whole $\mathbb{R}$.

Remark 2.1 *It follows that in any case, problem (9) has a minimum when the loss function is bounded from below and unbounded from above.*

Remark 2.2 *Take $g = \exp$ to match the constraint in (9) with the constraint considered by Bartlett et al. (2006). It turns out that a loss function obeying (6) and either i. or ii in Theorem 2.3. is classification calibrated in the two class case. See and Lemma A.2 in Appendix A for a formal proof of this fact.*

Clearly, by Remark 2.1, problem (9) would have a minimum for all three losses suggested earlier — the exponential, logistic (for both $g(x) = e^{cx}$, and $g(x) = e^x \frac{e^x+1}{2}$), and log-log loss.

Finally we conclude this subsection, by noting that the assumptions in both Theorem 1 and 2 in Zou et al. (2008) can be seen to imply that the assumptions in Theorems 2.1 and 2.3 hold, thus rendering these theorems as consequences of the main result shown above. For completeness we briefly recall what these conditions are. In Theorem 1, Zou et al. (2008) require a twice differentiable loss function ϕ such that $\phi'(0) < 0$ and $\phi'' > 0$. In Theorem 2 these conditions are slightly relaxed by allowing for part linear and part constant convex losses.



2.2 Fisher Consistency for Weighted 0/1 Loss

Although the expected 0/1 loss or equivalently the overall misclassification is an important summary for the overall performance of a classification, alternative measures may be preferred when the cost of misclassification is not exchangeable across outcome categories. For such settings, it would be desirable to incorporate the differential cost when evaluating the classification performance and consider a weighted misclassification rate. Consider a cost matrix $\mathcal{W} = [W(j,j)]_{n\times n}$ with $W(j,j)$ representing the cost in classifying the j^{th} class to the j^{th} class. Then, the optimal Bayes classifier is

$$c^{\mathcal{W}}_{\mathbf{f}_{\text{Bayes}}}(\mathbf{X}) = \underset{j}{\operatorname{argmin}} \sum_{j=1}^{n} W(j,j)\mathbb{P}(C=j|\mathbf{X}). \tag{13}$$

Setting $\mathcal{W} = 1-\mathcal{I}$ corresponds to the 0/1 loss and $c^{\mathcal{W}}_{\mathbf{f}_{\text{Bayes}}} = c_{\mathbf{f}_{\text{Bayes}}}$, where $\mathcal{I}$ is the identity matrix. Without loss of generality, we assume that $W(j,j) \geq 0$. For ϕ satisfying (6) and the condition in Theorem 2.3, we next establish the FC results for the weighted 0/1 loss parallel to those given in Theorems 2.1 and 2.3.

Proposition 2.4 *Define the weighted loss $\ell(F_j) = \sum_{j=1}^{n}\phi(F_j)W(j,j)$. Then the optimization problem:*

$$\min_{\mathbf{F}=(F_1,\ldots,F_n)^T:\prod_{j=1}^{n} g(F_j)=1} \sum_{j=1}^{n} \ell(F_j)\mathbb{P}(C=j|\mathbf{X}), \tag{14}$$

has a minimizer $\widehat{\mathbf{F}}^{\mathcal{W}} = (\widehat{F}_1^{\mathcal{W}},\ldots,\widehat{F}_n^{\mathcal{W}})^T$ which satisfies the property that:

$$H_\phi(\widehat{F}_j^{\mathcal{W}})w_j^{\mathcal{W}} = \widetilde{\mathcal{C}} \quad \text{for some } \widetilde{\mathcal{C}} < 0, \tag{15}$$

where $w_j^{\mathcal{W}} = \sum_{j=1}^{n} W(j,j)\mathbb{P}(C=j|\mathbf{X})$ assuming that $w_j^{\mathcal{W}} > 0$.

This proposition is a direct consequence of Theorem 2.1, after exchanging summations:

$$\sum_{j=1}^{n}\sum_{j=1}^{n}\phi(F_j)W(j,j)\mathbb{P}(C=j|\mathbf{X}) = \sum_{j=1}^{n}\phi(F_j)w_j^{\mathcal{W}}.$$

Remark 2.3 *Note that the above result hints on how one can relax assumption (2) by using the loss ℓ constructed with $\mathcal{W} = 1-\mathcal{I}$. Using this particular ℓ, Proposition 2.4 simply requires $w_j^{\mathcal{W}} > 0$, which would be satisfied if we required:*

$$\max_{j\in\{1,\ldots,n\}} \mathbb{P}(C=j|\mathbf{X}) < 1: \text{ almost surely in } \mathbf{X}.$$

This is indeed weaker than (2). If we wanted to recover the conditional probabilities simply note that $\mathbb{P}(C=j|\mathbf{X}) = 1 - w_j^{\mathcal{W}}$, and hence:

$$\mathbb{P}(C=j|\mathbf{X}) = 1 - \frac{\{H_\phi(\widehat{F}_j^{\mathcal{W}})\}^{-1}}{\sum_{j=1}^{n}\{H_\phi(\widehat{F}_j^{\mathcal{W}})\}^{-1}}.$$

The result also suggests that using the modified loss ℓ, we can attain the optimal weighted Bayes classifier $c^{\mathcal{W}}_{\mathbf{f}_{\text{Bayes}}}(\mathbf{X})$ based on $\operatorname{argmin}_j H_\phi(\widehat{F}_j^{\mathcal{W}})$.

3. Generic Algorithm for Constructing the Classifier

In this section we provide a generic boosting algorithm, based on the explicit structure (6) that the RFC loss functions possess, and analyze certain numerical convergence aspects of the algorithm in the special case when $g = \exp$.

3.1 A Generic Boosting Algorithm

The properties of ϕ and the results in Theorem 2.1 and 2.3 also lead to a natural iterative generic boosting algorithm to attain the minimizer.

3.1.1 A Conditional Iteration

In this subsection, we provide an iterative procedure, conditional on $\mathbf{X} = x$, which eventually leeds to a generic boosting algorithm. The usefulness of this conditional iteration is based on the following result.

Theorem 3.1 *Assume that ϕ satisfies (6) and the condition in Theorem 2.3. Starting from $\mathbf{F}^{(0)} \equiv 0$, i.e. $F_j^{(0)} = 0$ for all j, define the following iterative procedure:*

$$\mathbf{F}^{(m+1)} = \underset{\mathbf{F}:\prod g(F_j)=1}{\operatorname{argmax}} \sum_{j=1}^{n}\{g(F_j^{(m)}) - g(F_j)\}k(F_j)w_j. \tag{16}$$

This iteration is guaranteed to converge to a point $\mathbf{F}^*$ *with the following property:*

$$g(F_j^*)k(F_j^*)w_j = H_\phi(F_j^*)w_j = \mathcal{C} < 0.$$

In the theorem above, the iterations are defined conditionally on $\mathbf{X} = x$, and F_j can be understood as $f_j(x)$. If $H_\phi(\cdot)$ turns out to be monotone, the procedure above will converge to the global minimum, as we can conclude straight from Theorem 2.1. Even if the procedure does not converge to a global minimum, because of the property of the point that it converges to, $\mathbf{F}^*$ can be used to recover the Bayes classifier. This observation is particularly useful for minimizing a non-convex loss functions as in such cases it is often hard to arrive at the global minimum. Moreover, as before the point $\mathbf{F}^*$ can be used not only for classification purposes, but also to recover the exact probabilities w_j.

In practice, the procedure described in (16) can be used to derive algorithms for boosting. However, an unconditional version of (16) is needed since w_j are unknown in general. Noting that the expectation of $\mathbb{1}(C = j)$ given $\mathbf{X}$ is w_j, we have

$$\begin{aligned}
&\mathbb{E}\left(\sum_{j=1}^{n}\Big[g\{F_j^{(m)}(\mathbf{X})\} - g\{F_j(\mathbf{X})\}\Big]k\{F_j(\mathbf{X})\}w_j\right)\\
=&\mathbb{E}\left(\sum_{j=1}^{n}\Big[g\{F_j^{(m)}(\mathbf{X})\} - g\{F_j(\mathbf{X})\}\Big]k\{F_j(\mathbf{X})\}\mathbb{1}(C = j)\right)\\
=&\mathbb{E}\left(\Big[g\{\mathbf{Y}_C^T\mathbf{F}^{(m)}(\mathbf{X})\} - g\{\mathbf{Y}_C^T\mathbf{F}(\mathbf{X})\}\Big]k\{\mathbf{Y}_C^T\mathbf{F}(\mathbf{X})\}\right), \qquad (17)
\end{aligned}$$

where recall that $\mathbf{Y}_C = (\mathbb{1}(C = 1), ..., \mathbb{1}(C = n))^T$. We next derive a boosting algorithm iterating based on an empirical version of (17).

3.1.2 The Boosting Algorithm

To derive the boosting algorithm, we let $\mathcal{G} = \{G_b(\cdot), b = 1, ..., B\}$ denote the bag of weak learners with $G(\mathbf{X}) \in \{1, ..., n\}$ denoting the predicted class based on learner G. For the bth classifier in $\mathcal{G}$, define a corresponding vectorized version of G_b, $\mathbf{F}_b = (F_{b1}, ..., F_{bn})$, with

$$F_{bj}(\mathbf{X}) = \mathcal{C}_- + \mathbb{1}\{G_b(\mathbf{X}) = j\}(\mathcal{C}_+ - \mathcal{C}_-),$$

where $\mathcal{C}_- < 0$ and $\mathcal{C}_+ > 0$ are chosen such that $\prod_{j=1}^n g(F_{bj}) = 1$. Obviously, $\mathbf{Y}_C^T\mathbf{F}_b(\cdot) = \mathcal{C}_- + \mathbb{1}(G_b(\cdot) = C)(\mathcal{C}_+ - \mathcal{C}_-)$. Let $\mathcal{G}^* = \{\mathbf{F}_b(\cdot), b = 1, ..., B\}$ denote the bag of vectorized classification functions corresponding to the classifiers in $\mathcal{G}$.

We next propose a generic iterative boosting algorithm that greedily searches for an optimal weight and for which weak learner to aggregate at each iteration. The loss function ϕ is not directly used, and instead we rely on the $g(\cdot)$ and $k(\cdot)$ functions as specified in (6). Specifically, we initialize $\mathbf{F}^{(0)} := 0$. Then for $m = 1, ..., M$ with M being the total number of desired iterations, we obtain the maximizer of

$$\sum_{i=1}^{N} g\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i)\}\left[1 - g\{\mathbf{Y}_{Ci}^T\mathbf{F}(\mathbf{X}_i)\}^\beta\right]k\left(g^{-1}\left[g\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i)\}g\{\mathbf{Y}_{Ci}^T\mathbf{F}(\mathbf{X}_i)\}^\beta\right]\right),$$

with respect to $\mathbf{F} \in \mathcal{G}^*$ and $\beta \geq 0$, denoted by $\widehat{\mathbf{F}}$ and $\widehat{\beta}$. Then we update the classifier coordinate-wise as $F_j^{(m)} = g^{-1}\{g(F_j^{(m-1)})g(\widehat{F}_j)^{\widehat{\beta}}\}$ so that we have the following

$g\{\mathbf{Y}_C^T\mathbf{F}^{(m)}\} = g\{\mathbf{Y}_C^T\mathbf{F}^{(m-1)}\}g\{\mathbf{Y}_C^T\widehat{\mathbf{F}}\}^{\widehat{\beta}}$ holding for all C. This will ensure that the property $\prod_{j=1}^n g(F_j^{(m)}) = 1$ continues to hold throughout the iterations. Thus at each iteration, we would be greedily maximizing:

$$\sum_{i=1}^{N}\Big[g\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i)\} - g\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m)}(\mathbf{X}_i)\}\Big]k\left\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m)}(\mathbf{X}_i)\right\},$$

which is exactly the empirical version of (17).

For illustration, consider $g(x) = e^x$ with ϕ being differentiable and hence we may let $k(x) = \dot{\phi}(x)e^{-x}$. In this special case the update of $\mathbf{F}^{(m)}$ simplifies to $\mathbf{F}^{(m)} = \mathbf{F}^{(m-1)} + \widehat{\beta}\widehat{\mathbf{F}}$. Therefore following the iteration described above we have:

$$\underset{\mathbf{F}\in\mathcal{G}^*,\ \beta\geq 0}{\operatorname{argmin}} \sum_{i=1}^{N} -\{e^{-\beta\mathbf{Y}_{Ci}^T\mathbf{F}(\mathbf{X}_i)} - 1\}\dot{\phi}\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i) + \beta\mathbf{Y}_{Ci}^T\mathbf{F}(\mathbf{X}_i)\}. \qquad (18)$$

Note here, the apparent similarity between a coordinate descent (or gradient descent in a functional space) as proposed in Mason et al. (1999) and Friedman (2001), and the above iteration. Finally, we summarize the algorithm as follows.

Algorithm 1: Generic Boosting Algorithm

1. Set $\mathbf{F}^{(0)} = 0$
2. For $m = 1, \ldots, M$
   (a) Maximize $\sum_{i=1}^N \Big\{g\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i)\}\left[1 - g\{\mathbf{Y}_{Ci}^T\mathbf{F}(\mathbf{X}_i)\}^\beta\right]\times k\left(g^{-1}\left[g\{\mathbf{Y}_{Ci}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i)\}g\{\mathbf{Y}_{Ci}^T\mathbf{F}(\mathbf{X}_i)\}^\beta\right]\right)\Big\}$ with respect to $\mathbf{F} \in \mathcal{G}^*$, $\beta \geq 0$ to obtain $\widehat{\mathbf{F}}$ and $\widehat{\beta}$
   (b) Update $\mathbf{F}^{(m)}$ coordinate-wise as $F_j^{(m)} = g^{-1}\big\{g(F_j^{(m-1)})g(\widehat{F}_j)^{\widehat{\beta}}\big\}$.
3. Output $\mathbf{F}^{(M)}$ and classify via $\operatorname{argmax}_j H_\phi(F_j^{(M)})$.

3.2 Numerical Convergence of the Algorithm when $g(\cdot) = \exp(\cdot)$

In this subsection we illustrate how algorithm 1 performs in finite samples, if we let it run until convergence (using potentially infinitely many iterations). More specifically, we study the properties of the iteration above in the case when $g(x) = \exp(x)$, or in other words we are concerned with the iteration given by (18). In addition, we also want to explore the relationship between iteration (18) and the following optimization problem:

$$\inf_{\mathbf{F}\in\operatorname{span}\mathcal{G}^*} \sum_{i=1}^{N} \phi(\mathbf{Y}_{C_i}^T\mathbf{F}(\mathbf{X}_i)). \qquad (19)$$

To this end we formulate the following:

Definition 3.1 (Looping closure) *Let* π *be a permutation of the numbers* $\{1, \ldots, n\}$ *into* $\{\pi_1, \ldots, \pi_n\}$. *Consider the following "loop" functions, such that for all* $i = 1, \ldots, n$: $l^{(0)}(\pi_i) =$

$\pi_i, l^{(1)}(\pi_i) = \pi_{i+1}$, $l^{(k)}(\cdot) = l^{(1)}(l^{(k-1)}(\cdot))$, *where the indexing is* $\bmod n$, *and* $k = 1, \ldots, n-1$[5]. *We say that a classifier bag* $\mathcal{G}$ *is closed under "looping" if there exists a permutation* π *such that for all* $G \in \mathcal{G}$ *it follows that* $l^{(k)} \circ G \in \mathcal{G}$ *for all* $k = 0, \ldots, n-1$.

In practice, closure under looping can easily be achieved if it is not already present, by adding the missing classifiers to the bag. Similar bag closures have been considered in the two class case in Mason et al. (1999). We start our discussion with the following proposition, providing a property of the algorithm at its limiting points.

Proposition 3.2 *Suppose that the loss function* ϕ *is decreasing, continuously differentiable, bounded from below and satisfies (6) with* $g = \exp$. *Furthermore assume that, the classifier bag is closed under looping (see Definition 3.1). Then iterating (18), using possibly infinite amount of iterations until a limiting point* $\mathbf{F}^{(\infty)}$ *is reached, guarantees that the following condition holds:*

$$\sum_{i=1}^{N} \mathbf{Y}_{Ci}^T \mathbf{F}(\mathbf{X}_i) \dot{\phi}(\mathbf{Y}_{Ci}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) = 0, \tag{20}$$

for all $\mathbf{F} \in \mathcal{G}^*$.

Clearly, all examples of loss functions we considered satisfy the assumptions of Proposition 3.2. In the case when ϕ is convex, (20) shows that by iterating (18) we would arrive at an infimum of problem (19). This can be easily seen along the following lines. Assume that the $\widetilde{\mathbf{F}}$ is a point of infimum of (19) over the span of $\mathcal{G}^*$. By convexity of ϕ we have:

$$\sum_{i=1}^{N} \phi(\mathbf{Y}_{Ci}^T \widetilde{\mathbf{F}}(\mathbf{X}_i)) - \sum_{i=1}^{N} \phi(\mathbf{Y}_{Ci}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \geq \sum_{i=1}^{N} \mathbf{Y}_{Ci}^T [\widetilde{\mathbf{F}}(\mathbf{X}_i) - \mathbf{F}^{(\infty)}(\mathbf{X}_i)] \dot{\phi}(\mathbf{Y}_{Ci}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) = 0.$$

The last equality follows from (20) and the fact that $\widetilde{\mathbf{F}} - \mathbf{F}^{(\infty)}$ is in the span of classifiers.

In the case when ϕ is not convex, condition (20) remains meaningful, though it doesn't guarantee convergence to the infimum. In order for us to relate condition (20) to equation (11) in the general (non-convex loss) case, and make it more intuitive, we consider a simple and illustrative example. We restrict our attention to the two class case ($n = 2$), but the example can be easily generalized.

Example 3.1 *Consider a (disjoint) partition of the predictor support:* $\mathcal{X} = \mathcal{X}_1, \ldots, \mathcal{X}_B$. *Construct classifiers based on that partition in the following manner:*

$$G_b(x) = \begin{cases} 1, & \text{if } x \in \mathcal{X}_b \\ 2 & \text{otherwise} \end{cases},$$

and close them under looping. It is easily seen that, under this framework the vector $\mathbf{F}^{(\infty)}(x)$ *is constant for* $x \in \mathcal{X}_b$ *for a fixed* b. *Denote the value of this constant with* $\mathbf{F}_b^{(\infty)}$. *Plugging in the* b^{th} *classifier in equation (20) we obtain:* $N_b \dot{\phi}(\mathbf{Y}_1^T \mathbf{F}_b^{(\infty)}) - (N - N_b) \dot{\phi}(\mathbf{Y}_2^T \mathbf{F}_b^{(\infty)}) = 0$,

5. Note that the loop functions depend on the permutation π, but we suppress this dependence for clarity of exposition.

where N_b *is the number of observations correctly classified by the* b^{th} *classifier, or in other words:*

$$\frac{\{\dot{\phi}(\mathbf{Y}_1^T \mathbf{F}_b^{(\infty)})\}^{-1}}{\{\dot{\phi}(\mathbf{Y}_1^T \mathbf{F}_b^{(\infty)})\}^{-1} + \{\dot{\phi}(\mathbf{Y}_2^T \mathbf{F}_b^{(\infty)})\}^{-1}} = \frac{N_b}{N}.^6 \tag{21}$$

The LHS of (21) is an estimate of the probability $\mathbb{P}(C = 1 | \mathbf{X} \in \mathcal{X}_b)$, *which in turn is a proxy to the Bayes classifier. For the RHS of (21), note that our probability recovering rule (11) with* $g = \exp$ *becomes:*

$$\frac{\{\dot{\phi}(\mathbf{Y}_j^T \mathbf{F}^{(\infty)}(x))\}^{-1}}{\{\dot{\phi}(\mathbf{Y}_1^T \mathbf{F}^{(\infty)}(x))\}^{-1} + \{\dot{\phi}(\mathbf{Y}_2^T \mathbf{F}^{(\infty)}(x))\}^{-1}},$$

in the two class case. This expression is in complete agreement with the RHS of (21).

3.2.1 Convergence Analysis

In the convex loss function case, property (20) will be matched by a gradient descent methods in the function space such as AnyBoost (Mason et al., 1999). This motivates us to consider the question of the convergence rate of the newly suggested algorithm — is it slower, faster or the same as a gradient descent in the convex loss function case? At first glance the rate might appear to be slower as we are not using the "fastest" decrease at each iteration using simply the exp function. In the end of this subsection we establish a geometric rate of convergence under certain assumptions, which matches the convergence rate for gradient descent under similar assumptions.

As we argued in the previous subsection, in the case of a convex loss ϕ, (20) guarantees that iteration (18) converges to the infimum of problem (19). Let $\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)$ be the limiting (allowed to be $\pm\infty$) values achieving the infimum above. Before we formalize the convergence rate result, we will characterize the behavior of $\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)$. This question is of interest in its own right, as this characterization remains valid regardless of what boosting algorithm one decides to use to obtain the minimum/infimum.

For what follows we consider a loss function ϕ which satisfies a mildly strengthened condition (12). Namely, let ϕ be decreasing and for any $\alpha, c > 0$ it satisfies the following condition:

$$\phi(x) + c\phi(-\alpha x) \uparrow +\infty \text{ as } x \uparrow +\infty \tag{22}$$

It is worth noting that if condition (12) is satisfied for all n (recall that $g = \exp$ here) this would imply (22). Denote with B the total number of weak learners in the bag. Let $\mathbf{D} := \{\mathbf{Y}_{Ci}^T \mathbf{F}_j(\mathbf{X}_i)\}_{j,i}$ be a $B \times N$ matrix each entry of which is either $\mathcal{C}_+$ or $\mathcal{C}_-$. Again, we assume that the bag is closed under looping. Let $\mathbf{v} \in \mathbb{R}^N$ be a vector. Consider the equation $\mathbf{D}^T \boldsymbol{\alpha} = \mathbf{v}$ for some vector $\boldsymbol{\alpha} \in \mathbb{R}_{0,+}^B$, where $\mathbb{R}_{0,+} = [0, \infty)$. Note that because of the looping closure[7] the linear equation above has solution iff the equation $\mathbf{D}^T \boldsymbol{\alpha} = \mathbf{v}$ has a solution with $\boldsymbol{\alpha} \in \mathbb{R}^B$, since without loss of generality we can add a large positive constant

6. Here we assume that $\dot{\phi}(\mathbf{Y}_j^T \mathbf{F}_b^{(\infty)}) \neq 0, j \in \{1, 2\}$, which can be ensured if ϕ is unbounded from above
7. Looping closure (Definition 3.1) implies that the column sums of $\mathbf{D}$ are 0.

to the coordinates of $\boldsymbol{\alpha}$. It follows that the equation $\mathbf{D}^T\boldsymbol{\alpha} = \mathbf{v}$ has a solution $\boldsymbol{\alpha} \in \mathbb{R}^B_{0,+}$ iff $\mathbf{v} \in \text{row}(\mathbf{D})$.

To see the connection between the linear equation above and optimization problem (19) consider the following simple example:

Example 3.2 *The function* $\sum_{i=1}^N \phi(\sum_{j=1}^B \alpha_j \mathbf{Y}_{Ci}^T \mathbf{F}_j(\mathbf{X}_i))$ *cannot have a minimum, if there exists a vector* $\mathbf{v} \in \mathbb{R}^N_+$*, such that the equation* $\mathbf{D}^T\boldsymbol{\alpha} = \mathbf{v}$ *has a solution —* $\widehat{\boldsymbol{\alpha}} \in \mathbb{R}^B_{0,+}$*, where* $\mathbb{R}_+ = (0, \infty)$ *. To see this, suppose the contrary, take an arbitrary constant* $R > 0$ *and note that:*

$$\sum_{i=1}^N \phi(\sum_{j=1}^B R\widehat{\alpha}_j \mathbf{Y}_{Ci}^T \mathbf{F}_j(\mathbf{X}_i)) = \sum_{i=1}^N \phi(Rv_i).$$

Take the limit $R \to \infty$*, and it is clear that the infimum* $N\phi(+\infty)$ *is achieved.*

Example 3.2 illustrates that if we want to have a solution smaller than $N\phi(+\infty)$, $\mathbf{D}$ cannot have rank N. Denote the rank of $\mathbf{D}$ with r.

More generally, our next result provides a characterization of how many (and which) of the values $\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)$ are set to $+\infty$ at the infimum of (19). Consider the perp space of the row space of the matrix $\mathbf{D}$, $\mathbf{E} := \text{row}(\mathbf{D})^\perp$. Out of all possible bases of $\mathbf{E}$ including the 0 vector, select the one $\mathbf{e}_1, \ldots, \mathbf{e}_s$ $(s = \min(N, B) - r + 1)$ for which the vector $\mathbf{e}_1$ has the most strictly positive entries at I coordinates and zeros at the rest[8]. We have the following:

Proposition 3.3 *Let* ϕ *be a decreasing loss function satisfying (22). Set* $M := N - I$*, where* $I \in \{0, \ldots, N\}$*. We have that:*

$$(N - M - 1)\phi(0) + (M + 1)\phi(+\infty) < \inf_{\mathbf{F} \in \text{span}\, \mathcal{G}^*} \sum_{i=1}^N \phi(\mathbf{Y}_{Ci}^T \mathbf{F}(\mathbf{X}_i)) \leq (N - M)\phi(0) + M\phi(+\infty).$$

Moreover, exactly M *of the values* $\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)$ *(i will be corresponding to the* 0 *coordinates of* $\mathbf{e}_1$*) will be set to* $+\infty$ *at the infimum.*

Proposition 3.3 characterizes the cases when one should expect problem (19) to have a minimum. In fact, in the cases where $I > 0$, we can simply delete the observations corresponding to the rows of $\mathbf{e}_1$ that are 0, and solve the optimization only on the set of observations left, as it can be seen from the proof.

Next we formulate the speed of the convergence of the algorithm we suggested, in the case when the function ϕ is convex. For simplicity we assume that the matrix of classifier entries, $\mathbf{D}$, is such that there is a strictly positive vector in the perp of the row space of $\mathbf{D}$. If that is not the case as argued we can delete observations that will be set to $+\infty$ at the maximum, and work with the rest. Denote with $\mathcal{S} = \{\mathbf{v} : \mathbf{D}^T\boldsymbol{\alpha} = \mathbf{v} \text{ with } \boldsymbol{\alpha} \geq 0, \sum_{i=1}^N \phi(v_i) \leq N\phi(0)\}$. Proposition 3.3 then implies that, the set $\mathcal{S}$ is bounded coordinate-wise. Next we formulate the result:

Theorem 3.4 *Let the convex, decreasing loss function* ϕ *be strongly convex with Lipchitz and bounded derivative on any compact subset of* $\mathbb{R}$*, and satisfies (22) and (6) with* $g = \exp$*. Furthermore, assume that there is a stricly positive vector in* $\text{row}(\mathbf{D})^\perp$*, and define the set* $\mathcal{S}$ *as above. Let* $\mathbf{F}^* \in \text{span}\, \mathcal{G}^*$ *achieves the minimum in problem (19). Denote with* $\varepsilon_m = \sum_{i=1}^N \phi(\mathbf{Y}_{Ci}^T \mathbf{F}^{(m)}(\mathbf{X}_i)) - \sum_{i=1}^N \phi(\mathbf{Y}_{Ci}^T \mathbf{F}^*(\mathbf{X}_i))$*, where* $\mathbf{F}^{(m)}$ *is produced iteratively using (18). Then there exists a constant* $K < 1$ *depending on the matrix* $\mathbf{D}$*, the sample size* N *and the set* $\mathcal{S}$*, such that:*

$$\varepsilon_{m+1} \leq \varepsilon_m K.$$

As we can see from Theorem 3.4 if we use this algorithm in the convex loss function case, we wouldn't lose convergence speed to gradient descent (see Nesterov (2004) Theorem 2.1.14), but in the non-convex function case which still obeys (6) this algorithm will be converging to a local minimum. In the latter case we will still be capable of recovering the Bayes classifier, as indicated by equation (20).

8. We allow $I = 0$, in which case $\mathbf{e}_1$ would simply represent the 0 vector.

4. Numerical Studies and Data Examples

In this section we validate empirically the performance of the generic boosting algorithm developed in the previous section with various choices of ϕ, comparing it to popular classification algorithms such as SVM and SAMME on simulated datasets, real benchmark datasets from the UCI machine learning, as well as an EMR study on diabetic neuropathy conducted at the Partners Healthcare.

For each dataset, we evaluated our proposed boosting algorithm based on (i) $\phi(x) = \log(1+e^{-x})$ (Logistic) with $g(x) = e^x$ and $k(x) = -\{e^x(1+e^x)\}^{-1}$; (ii) $\phi(x) = \log(\log(e^{-x}+e))$ (LogLog) with $g(x) = e^x$ and $k(x) = \{e^x(e^{x+1}+1)\log(e^{-x}+e)\}^{-1}$. We also experimented with $g(x) = e^{cx}$ for different values of c, and our results (not reported) were robust with respect to the choice of c. We also compare each of these algorithms to the commonly used LASSO (Friedman et al., 2010)/Multinomial Logistic Regression and SVM procedures. The SVM was trained with RBF kernel where the tuning parameter for the kernel function was chosen via the `sigest` function of `ksvm` library. The `sigest` procedure outputs three quantiles — 0.1, 0.5, 0.9 of the distribution of $\|X - X'\|_2^{-2}$ where X and X' are two predictors sampled from the matrix $\mathbf{X}$, and we take the mean of these quantiles as the tunning parameter in the RBF kernel for robustness. The fitting was performed with the `kernlab` `R` package implementation – `ksvm` which uses the "one-against-one"-approach to deal with multi-class problems see Hsu and Lin (2002) for example. The LASSO procedure with $\mathbf{X}$ being the predictors was based on an ℓ_1 penalized logistic regression through the `glmnet` implementation, and the tuning parameter was selected based on 5-fold CV. Throughout, the bag of classifiers for all boosting algorithms consisted of decision trees and multinomial logistic regressions weak learners, the predictors in each of which represented a subsample of the whole predictor set.

4.1 Simulation Studies

To distinguish the performance of our algorithm to the classical SAMME algorithm we chose a simulation setting with a complicated decision boundary. We borrow the idea of this simulation from the celebrated paper by Friedman et al. (2000).

The predictor matrix is generated as follows $\mathbf{X} \sim [N(0, \mathbb{I}_{10\times 10})]^{10}$. Next for each observation $\mathbf{X}_i$ it's ℓ_1 or ℓ_2 norms were calculated. The class assignment was then performed

based on:

$$C_i = \sum_{j=1}^{n} j \mathbb{1}(r_j \leq \|\mathbf{X}_i\|_\ell < r_{j+1}),$$

where $0 = r_1 \leq r_2 \leq \ldots \leq r_{n+1} = \infty$, $\ell \in \{1, 2\}$ and for $\mathbf{X} \in \mathbb{R}^p$ we have $\|\mathbf{X}\|_\ell := \left(\sum_{i=1}^{p} |X_i|^\ell\right)^{1/\ell}$. The thresholds r_j were selected in such a way so that each class got an approximately equal number of observations. Geometrically the classes $C = \{1, \ldots, k\}$ for $k < n$ were observations belonging to a 10 dimensional ℓ_1 or ℓ_2 sphere. Moreover, the more cutoffs r_j, the more classes the problem has, and hence the more complicated classification becomes.

We compared 5 algorithms – the SAMME, our algorithm with the logistic and loglog losses, the SVM and the LASSO logistic regression algorithm. Each simulation, 3200 observations were generated, 200 of which were used for training and 3000 were left out for testing purposes. We performed in total of 500 simulation for each scenario. The results are summarized in the tables 1 and 2 below:

Table 1: Percent misclassification for ℓ_1 Spheres

	$n = 3$	$n = 4$	$n = 5$	$n = 6$
SVM	17.9% (0.46)	29.4% (0.45)	37.6% (0.81)	46.6% (1.21)
LASSO	7.8% (11.6)	27.6% (5.00)	30.4% (20.0)	43.9% (15.9)
SAMME	6.9% (0.31)	10.7% (0.20)	16.4%(0.33)	28.4% (0.34)
Logistic	6.5% (22.6)	10.0% (16.4)	15.1% (22.3)	25.1% (23.6)
LogLog	6.6% (32.7)	9.8% (24.0)	15.0% (33.5)	25.2% (34.5)

Table 2: Percent misclassification for ℓ_2 Spheres

	$n = 3$	$n = 4$	$n = 5$	$n = 6$
SVM	17.0% (0.44)	28.0% (0.78)	36.6% (0.83)	44.1% (1.15)
LASSO	8.7% (11.8)	26.7% (5.70)	31.9% (19.7)	45.0% (14.6)
SAMME	8.2% (0.33)	12.9% (0.46)	19.7% (0.30)	28.9% (0.29)
Logistic	7.5% (23.4)	11.7% (25.3)	17.3% (24.6)	25.4% (23.5)
LogLog	7.7% (33.7)	11.5% (35.6)	17.9% (35.7)	25.8% (35.7)

As we can see our boosting based procedures dominated in all scenarios with the difference becoming more pronounced the more number of classes we have. We observed that our algorithms seem to combine classifiers from the bag more efficiently as compared to the SAMME algorithm in this scenario, especially so when the number of classes was large.

4.2 Experiments with UCI Benchmark Datasets

To present a more comprehensive assessment of our procedure, we analyzed 5 datasets from the UCI machine learning repository. The summaries of the characteristics of the datasets we used can be found in Table 3.

We compare results from SAMME, LogLog and Logistic from our proposed methods, SVM and a multinomial logistic regression (referred to as MLogisticReg). The MLogisticReg

9. Originally 10 classes; Extreme classes were grouped together for balanced class counts.

Table 3: Summary of datasets

	Dataset	Train Set / Test Set Size	# Features	# Classes
IS	Image Segmentation	210/2100	19	7
LED	Led Display	200/5000	7	10
RWQ[9]	Red Wine Quality	700/899	11	4
SE	Seeds	100/110	7	3
EC	Ecoli	100/236	7	8

was used instead of the LASSO procedure, as these datasets have relatively few features, and the LASSO procedure generally performs worse than the standard. As before all boosting algorithms were ran for 50 iterations.

For the analysis of all 5 datasets, the bag of weak learners we used consisted of decision trees and multinomial logistic regressions with subsampled prefixed number of predictors.

Table 4: Percent misclassifications (run time in seconds)

	IS	RWQ	SE	LED	EC
SVM	16.19% (0.44)	29.37% (0.28)	8.18% (0.08)	28.64% (1.26)	37.28% (0.27)
MLogisticReg	10.57% (0.08)	28.59% (0.09)	5.45% (0.04)	28.34% (0.06)	41.52% (0.04)
SAMME	5.09% (0.17)	30.14% (0.74)	4.55% (0.02)	29.36% (0.04)	22.45% (0.02)
Logistic	5.00% (13.6)	27.36% (35.3)	3.64% (1.61)	27.02% (1.53)	19.49% (1.22)
LogLog	4.95% (16.9)	28.36% (48.8)	3.64% (1.30)	27.00% (2.01)	19.49% (1.45)

We observe that in nearly all cases the newly suggested procedure performed better than the SAMME procedure, which typically converged too fast failing to include enough classifiers. The misclassifications rates of both the Logistic and LogLog based boosting algorithms were also better compared to the SVM and logistic regression in all 4 datasets.

4.3 Summary of Simulation Results and Benchmark Data Analyses

The results provided above demonstrate that the suggested algorithms can be used in successfully practice, and can outperform existing algorithms in many cases. The proposed boosting algorithm with LogLog loss appears to perform well across all settings considered with respect to classification accuracy. The robustness and gained accuracy of our proposed procedures come at the price of being more computationally intensive than the competitors. The slow speed is mostly due to the optimization with respect to β (see Algorithm 1) required at each iteration for each classifier in the bag. The SAMME algorithm avoids this search as an explicit solution to its corresponding optimization problem exists.

Further computational complexity in boosting procedures comes from two sources: one of them is the complexity of the classifiers in the bag, and the other is the number of classifiers in the bag. For massive datasets (with both high numbers of observations and of predictors) the optimization involved in our algorithm might be prohibitive if one decides to include a lot of weak learners. The methods we suggest could work in a reasonable time,

if one opts for sampling not too many classifiers. A relaxation of the procedure might be warranted in cases where a lot of classifiers will be used, and such relaxed algorithms are left for future work.

4.4 Application to an EMR Study of Diabetic Neuropathy

To illustrate our proposed generic boosting algorithm and demonstrate the advantage of having multiple losses, we apply our procedures to an electronic medical record (EMR) study, conducted at the Partners Healthcare, aiming to identify patients with different subtypes of diabetic neuropathy. Diabetic neuropathy (DN), a serious complication of diabetes, is the most common neuropathy in industrialized countries with an estimate of about 20-30 million people affected by symptomatic DN worldwide (Said, 2007). Increasing rates of obesity and type 2 diabetes could potentially double the number of affected individuals by the year 2030. The prevalence of DN also increases with time and poor glycemic control (Martin et al., 2006). Although many types of neuropathy can be associated with diabetes, the most common type is diabetic polyneuropathy and pain can develop as a symptom of diabetic polyneuropathy (Thomas and Eliasson, 1984; Galer et al., 2000). Pain in the feet and legs was reported to occur in 11.6% of insulin dependent diabetics and 32.1% of noninsulin dependent diabetics (Ziegler et al., 1992). Unfortunately, risk factors for developing painful diabetic neuropathy (pDN) are generally poorly understood. pDN has been reported as more prevalent in patients with type 2 diabetes and women (Abbott et al., 2011). Prior studies have also reported an association between family history and pDN, suggesting a potential genetic predisposition to pDN (Galer et al., 2000). To enable a genetic study of pDN and non-painful DN (npDN), an EMR study was performed to identify patients with these two subtypes of DN by investigators from the informatics for integrating biology to the bedside (i2b2), a National Center for Biomedical Computing based at Partners HealthCare (Murphy et al., 2006, 2010).

To identify such patients, we created a datamart compromising 20,000 patients in the Partners Healthcare with relevant ICD9 (International Classification of Diseases, version 9) codes. Two sources of information were utilized to classify patients' DN status and subtypes: (i) structured clinical data searchable in the EMR such as ICD9 codes; and (ii) variable identified using natural language processing (NLP) to identify medical concepts in narrative clinical notes. Algorithm development and validation was performed in a training set of 611 patients sampled from the datamart. To obtain the gold standard disease status for these patients, several neurologists performed chart reviews and classified them into no DN, pDN and npDN. The distribution of the classes was $64\%, 14\%, 22\%$ respectively. To train the classification algorithms, we included a total of 85 predictors most of which are NLP variables, counting mentions of medical concepts such as *"pain"*, *"hypersensitivity"*, and *"diabetic neuropathy"*.

We trained boosting classification algorithms to classify these 3 disease classes. We used decision stumps as weak learners. They only have two nodes with the first node deciding between class C_1 vs C_2 and C_3 and the other node deciding between C_2 vs C_3, where $\{C_1, C_2, C_3\}$ is a permutation of {no DN,pDN,npDN}. In order to illustrate the algorithms we used 311 observations as a training set and the rest 300 patients we set off as a test set.

We report the percentage mis-classifications in Table 5. The boosting results show some improvement, as compared to standard methods. We can also see that the generic boosting algorithm performs slightly better than SAMME in this situation with both the logistic and the LogLog losses. It warrants further research whether picking richer tree structures as opposed to using stumps, would yield an even better performance on this dataset.

Table 5: Percent misclassifications (run time in seconds)

	% incorrect
SVM	43.67% (0.26)
LASSO	37.00% (15.8)
SAMME	33.33% (0.21)
Logistic	31.67% (9.55)
LogLog	31.67% (12.5)

5. Discussion

For multi-category classification problems, we described in this paper a class of loss functions that attain FC properties and provided theoretical justifications for how such loss functions can ultimately lead to optimal Bayes classifier. We extended the results to accommodate differential costs in misclassifying different classes. To approximate the minimizer of the empirical losses, we demonstrated that a natural iterative procedure can be used to derive generic boosting algorithms for any of the proposed losses. Numerical results suggest that non-convex losses such as LogLog could potentially lead to algorithms with better classification performance. Although the LogLog loss appears to perform well across different settings considered in the numerical studies, choosing an optimal loss for a given dataset warrants further research. Our preliminary studies (results not shown) suggest that procedures such as the cross-validation can potentially be used for loss selections.

Our proposed algorithm not only depends on the choice of ϕ but also the associated $g(\cdot)$ and $k(\cdot)$ functions as indicated in (6). We can think of g as a positive deformation of the real line and even with the same ϕ, changing g could also change the classifiers. Most existing boosting algorithms correspond to $g(x) = e^x$, in which case the constraint $\prod_{j=1}^n g(F_j) = 1$ simplifies to the commonly seen condition $\sum_j F_j = 0$. Moreover if ϕ is smooth and convex, one may let $k(x) = \dot{\phi}(x)/e^x$. Thus, under convexity, $H_\phi(x) = \dot{\phi}(x) = d\phi(x)/dx$ is an increasing function and ϕ is Fisher consistent in the traditional sense. We also saw, that even when ϕ is not convex, our suggested losses are Fisher consistent in the standard sense. Moreover, we argued that loss functions satisfying (6), can be used to recover the exact conditional probabilities. It would be interesting to develop adaptive boosting procedure where we use different g functions in the process of boosting adaptively. For example, in the suggested logistic loss boosting with $g(x) = e^{cx}$, we can adaptively select the parameter c, for better convergence results of the algorithm which will potentially result in a better classification results. We were provided a property of the limiting point of the algorithm in the case where $g = \exp$. Furthermore, we characterized when the problem has a minimum in the finite sample case under certain assumptions on ϕ. The resemblance of the proposed

generic boosting algorithm with coordinate descent, helped us to establish geometric rate of convergence in the convex loss function case. The consistency of the algorithm under conditions such as finite VC dimension of the classifier bag, warrants future research.

Acknowledgments

The authors would like to thank Alexander Rakhlin and three referees for their valuable suggestions and feedback, which led to improvements in the present manuscript. This research was partially supported by Research Grants NSF DMS1208771, NIH R01GM113242-01, NIH U54HG007963 and NIH RO1HL089778.

Appendix A. Proofs

Lemma A.1 *Assumption (6) implies that the function $\phi(g^{-1}(z))$ is continuously differentiable and convex for all $z \in g(S)$.*

Proof [Proof of Lemma A.1] Set $z := g(x), z' := g(x')$ in (6). When $x, x' \in S$ we have $z, z' \in g(S)$ and vice versa. Now (6) can be rewritten as:

$$\phi(g^{-1}(z)) - \phi(g^{-1}(z')) \geq (z - z')k(g^{-1}(z')). \tag{23}$$

Changing the roles of z and z' and using the fact that both $z, z' \in g(S)$ we obtain:

$$\phi(g^{-1}(z')) - \phi(g^{-1}(z)) \geq (z' - z)k(g^{-1}(z)).$$

The above two inequalities, combined with the fact that k and g^{-1} are increasing, give that for any $z \neq z', z, z' \in g(S)$ we have:

$$\min\{k(g^{-1}(z)), k(g^{-1}(z'))\} \leq \frac{\phi(g^{-1}(z')) - \phi(g^{-1}(z))}{z' - z} \leq \max\{k(g^{-1}(z)), k(g^{-1}(z'))\}. \tag{24}$$

By the continuity of k and g we have that the composition $k(g^{-1}(\cdot))$ is also continuous. Taking the limit $z' \to z$ in (24) shows that the function $\phi(g^{-1}(z))$ is differentiable on $g(S)$ with a continuous derivative equal to $k(g^{-1}(z))$. Now the convexity of $\phi(g^{-1}(z))$ follows from (23). ■

Proof [Proof of Theorem 2.1] To show that $H_\phi(\widehat{F}_j)w_j = \mathcal{C}$ for some $\mathcal{C} < 0$, define $\Omega = \{\mathbf{F} = (F_1, \ldots, F_n) : F_j \in S, j = 1, ..., n\}$, where recall that $S = \{z \in \mathbb{R} : k(z) \leq 0\}$. From (6),

$$\sum_{j=1}^n \phi(\widehat{F}_j)w_j \geq \sum_{j=1}^n \phi(F_j)w_j + \sum_{j=1}^n \{g(\widehat{F}_j) - g(F_j)\}k(F_j)w_j \quad \text{for any } \mathbf{F} \in \Omega. \tag{25}$$

Since $\widehat{\mathbf{F}}$ minimizes $\sum_{j=1}^n \phi(F_j)w_j$, (25) implies that

$$\sum_{j=1}^n g(F_j)k(F_j)w_j \geq \sum_{j=1}^n g(\widehat{F}_j)k(F_j)w_j \quad \text{for any } \mathbf{F} \in \Omega. \tag{26}$$

For any given constant $\mathcal{C} < 0$, let $\widetilde{F}_{\mathcal{C}j}$ be the solution to $g(\widehat{F}_j)k(\widetilde{F}_{\mathcal{C}j})w_j = \mathcal{C}$ or equivalently $\widetilde{F}_{\mathcal{C}j} = k^{-1}[\mathcal{C}/\{g(\widehat{F}_j)w_j\}]$. Obviously $\widetilde{\mathbf{F}}_{\mathcal{C}} \in \Omega$ for all $\mathcal{C} < 0$. We next show that there exists $\mathcal{C}_0 < 0$ such that $\prod_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j}) = \prod_{j=1}^n g(k^{-1}[\mathcal{C}_0/\{g(\widehat{F}_j)w_j\}]) = 1$. Since g and k are continuous and strictly increasing functions, it suffices to show that $\prod_{j=1}^n g(\widetilde{F}_{0j}) > 1$ and $\prod_{j=1}^n g(\widetilde{F}_{\mathcal{C}j}) \leq 1$ for some $\mathcal{C}$. Obviously $\prod_{j=1}^n g(\widetilde{F}_{0j}) > 1$ since $g\{k^{-1}(0)\} > g(0) = 1$. Now let $\mathcal{C}_1 = k(0)\max_j\{g(\widehat{F}_j)w_j\} < 0$. Then for all j, $\mathcal{C}_1/\{g(\widehat{F}_j)w_j\} \leq k(0)$ and thus $g(k^{-1}[\mathcal{C}_1/\{g(\widehat{F}_j)w_j\}]) \leq g(0) = 1$. Then by continuity of g and k, there exists $\mathcal{C}_0 \in [\mathcal{C}_1, 0)$ such that $\prod_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j}) = 1$. Thus, the constructed $\widetilde{\mathbf{F}}_{\mathcal{C}_0}$ possesses several properties: (i) $g(\widehat{F}_j)k(\widetilde{F}_{\mathcal{C}_0 j})w_j = \mathcal{C}_0$; (ii) $\prod_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j}) = 1$; and (iii) $k(\widetilde{F}_{\mathcal{C}_0 j}) < 0$ and hence $\widetilde{\mathbf{F}}_{\mathcal{C}_0} \in \Omega$. It then follows from the AM-GM inequality that

$$\sum_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j})\{-k(\widetilde{F}_{\mathcal{C}_0 j})\}w_j \geq n\left[\prod_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j})\{-k(\widetilde{F}_{\mathcal{C}_0 j})\}w_j\right]^{n^{-1}} = n\left[\prod_{j=1}^n g(\widehat{F}_j)\{-k(\widetilde{F}_{\mathcal{C}_0 j})\}w_j\right]^{n^{-1}} = -n\mathcal{C}_0,$$

where we used the fact that $\prod_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j}) = \prod_{j=1}^n g(\widehat{F}_j) = 1$. This, together with (26), implies that

$$n\mathcal{C}_0 \geq \sum_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j})k(\widetilde{F}_{\mathcal{C}_0 j})w_j \geq \sum_{j=1}^n g(\widehat{F}_j)k(\widetilde{F}_{\mathcal{C}_0 j})w_j = n\mathcal{C}_0$$

and hence $n\mathcal{C}_0 = \sum_{j=1}^n g(\widetilde{F}_{\mathcal{C}_0 j})k(\widetilde{F}_{\mathcal{C}_0 j})w_j$. Thus, the equality holds in the AM-GM inequality above, which also implies that $g(\widetilde{F}_{\mathcal{C}_0 j})k(\widetilde{F}_{\mathcal{C}_0 j})w_j = \mathcal{C}_0$. Since $g(\widehat{F}_j)k(\widetilde{F}_{\mathcal{C}_0 j})w_j = \mathcal{C}_0$, $k(\widetilde{F}_{\mathcal{C}_0 j}) \neq 0$ and g is strictly increasing, we have $g(\widehat{F}_j) = g(\widetilde{F}_{\mathcal{C}_0 j})$ and hence $\widehat{F}_j = \widetilde{F}_{\mathcal{C}_0 j}$. Therefore,

$$g(\widehat{F}_j)k(\widehat{F}_j)w_j = H_\phi(\widehat{F}_j)w_j = \mathcal{C}_0. \tag{27}$$

Obviously if $H_\phi(\cdot)$ is strictly monotone then $\widehat{F}_j = H_\phi^{-1}(\mathcal{C}_0/w_j)$ which is unique. ■

Proof [Proof of Proposition 2.2] The function ϕ is strictly decreasing on the set $S' := \{z : k(z) < 0\}$, as from (6) for any $x < x', x, x' \in S'$ we have:

$$\phi(x) - \phi(x') \geq (g(x) - g(x'))k(x') > 0.$$

Furthermore, it follows from Theorem 2.1, that $\widehat{F}_j \in S'$ since by (27) $k(\widehat{F}_j) < 0$ for all j. Next we show that if $w_\jmath > w_j$ we must have $\phi(\widehat{F}_\jmath) \leq \phi(\widehat{F}_j)$. This observation follows since:

$$\phi(\widehat{F}_\jmath)w_\jmath + \phi(\widehat{F}_j)w_j \leq \phi(\widehat{F}_j)w_\jmath + \phi(\widehat{F}_\jmath)w_\jmath,$$

or else $\widehat{\mathbf{F}}$ cannot be a minimum of (9), as we can swap $\widehat{F}_\jmath$ and $\widehat{F}_j$ to obtain a strictly smaller value while still satisfying the constraint. Furthermore by Theorem 2.1, $w_\jmath \neq w_j$ implies that $\widehat{F}_\jmath \neq \widehat{F}_j$ because otherwise $H_\phi(\widehat{F}_\jmath) = H_\phi(\widehat{F}_j)$ and hence $w_\jmath = w_j$ by (10). Since ϕ is strictly decreasing on S' it also implies $\phi(\widehat{F}_\jmath) \neq \phi(\widehat{F}_j)$. Hence $w_\jmath > w_j$ implies $\phi(\widehat{F}_\jmath) < \phi(\widehat{F}_j)$. Finally, the last observation gives:

$$\operatorname*{argmin}_{j\in\{1,\ldots,n\}} \phi(\widehat{F}_j) = {}^{10} \operatorname*{argmax}_{j\in\{1,\ldots,n\}} w_j.$$

The fact that ϕ is decreasing on S' completes the proof. ■

Proof [Proof of Theorem 2.3] To show that a finite minimizer $\widehat{\mathbf{F}}$ exists, it suffices to show that $g(\widehat{F}_j)$ is finite and bounded away from 0, for $j = 1, ..., n$. To this end, we note that the condition (12) is equivalent to,

$$\lim_{x\downarrow 0} c_1\phi(g^{-1}(x)) + c_2\phi(g^{-1}(x^{-(n-1)})) = +\infty \quad \text{for all } c_1, c_2 > 0. \tag{28}$$

We next show that at the minimizer $\widehat{\mathbf{F}}$, $\widehat{m} = \min_j g(\widehat{F}_j) = g(\widehat{F}_{j^*})$ is bounded away from 0, where $j^* = \text{argmin}_j\, g(\widehat{F}_j)$. Since $1 = \prod_{j=1}^n g(\widehat{F}_j) \geq g(\widehat{F}_j)\widehat{m}^{n-1}$, we have $\widehat{F}_j \leq g^{-1}(\widehat{m}^{-(n-1)})$ for $j = 1, ..., n$. If ϕ is *decreasing* over $\mathbb{R}$, then

$$\begin{aligned}\phi(0)\sum_{j=1}^n w_j &\geq \sum_{j=1}^n \phi(\widehat{F}_j)w_j = w_{j^*}\phi\{g^{-1}(\widehat{m})\} + \sum_{j\neq j^*} w_j\phi(\widehat{F}_j)\\ &\geq w_{j^*}\phi\{g^{-1}(\widehat{m})\} + \sum_{j\neq j^*} w_j\phi\{g^{-1}(\widehat{m}^{-(n-1)})\}.\end{aligned}$$

From (28) with $c_1 = w_{j^*}$ and $c_2 = \sum_{j\neq j^*} w_j$, we conclude that $\widehat{m}$ must be bounded away from 0 since $\sum_{j=1}^n \phi(\widehat{F}_j)w_j \to \infty$ if $\widehat{m} \to 0$. Thus, there exists $m_0 > 0$ such that $\widehat{m} \geq m_0$ and consequently

$$0 < m_0 \leq g(\widehat{F}_j) \leq m_0^{-(n-1)} < \infty, \quad j = 1, ..., n.$$

Now, if ϕ is not decreasing on the whole $\mathbb{R}$, then there must exist $F^* < \infty$ such that $k(F^*) = 0$ since ϕ is strictly decreasing on $S' = \{z : k(z) < 0\}$.

Now we show that $\widehat{\mathbf{F}} \in \Omega \equiv \{\mathbf{F} = (F_1, \ldots, F_n) : F_j \in S, j = 1, \ldots, n\}$ as defined in Theorem 2.1. To this end, we note that ϕ is strictly decreasing on S' and $(-\infty, 0] \subset S'$. We next argue by contradiction that $\widehat{F}_j \in S$ or equivalently $\widehat{F}_j \leq F^*$ for all j. For any $F > F^*$, $\phi(F) - \phi(F^*) \geq \{g(F) - g(F^*)\}k(F^*) = 0$ by (6). Let $\mathcal{A} = \{j : \widehat{F}_j > F^*\}$ and $\widehat{F}_j^* = \mathbb{1}(j \in \mathcal{A})F^* + \mathbb{1}(j \notin \mathcal{A})\widehat{F}_j$. If $\mathcal{A}$ is not an empty set, then $\sum_{j=1}^n \phi(\widehat{F}_j^*)w_j \leq \sum_{j=1}^n \phi(\widehat{F}_j)w_j$ and $\prod_{j=1}^n g(\widehat{F}_j^*) < 1$. Since $g(F^*) > 1$, there must exist some $\widehat{F}_j^{**}$ with $F^* \geq \widehat{F}_j^{**} \geq \widehat{F}_j$ for $j \notin \mathcal{A}$ and $\widehat{F}_j^{**} = F^*$ for $j \in \mathcal{A}$ such that $\prod_{j=1}^n g(\widehat{F}_j^{**}) = 1$ and $F^* \geq \widehat{F}_j^{**} > \widehat{F}_j$ for some $j \notin \mathcal{A}$. Since ϕ is strictly decreasing on S, $\sum_{j=1}^n \phi(\widehat{F}_j^{**})w_j < \sum_{j=1}^n \phi(\widehat{F}_j^*)w_j \leq \sum_{j=1}^n \phi(\widehat{F}_j)w_j$, which contradicts that $\widehat{\mathbf{F}}$ is the minimum. Therefore, $\widehat{\mathbf{F}} \in \Omega$.

Hence $\widehat{F}_j \leq F^*$ and $g(\widehat{F}_j) \leq g(F^*) = m_1 \in (0, \infty)$. On the other hand, since $\prod_{j=1}^n g(\widehat{F}_j) = 1$, we have $g(\widehat{F}_j) \geq m_1^{-(n-1)}$ and thus $g(\widehat{F}_j)$ is also bounded away from 0 and finite.

Remark A.1 *As a useful remark we mention that the same argument shows that given any finite vector $\widehat{\mathbf{F}}$, the vectors $\mathbf{F}$ with $\sum_j \phi(F_j)w_j \leq \sum_j \phi(\widehat{F}_j)w_j$ are located on a compact set (provided that $\widehat{F}_j < F^*$ for all j in the second case).*

■

10. Recall that "=" should be interpreted as "⊆".

Lemma A.2 *Any loss function ϕ satisfying (6) with $g = \exp$, and either i. or ii. from Theorem 2.3 is classification calibrated in the two class case.*

Remark A.2 *Recall that a loss function ϕ is classification calibrated in the two class case if, for any point $w_1 + w_2 = 1$ with $w_1 \neq \frac{1}{2}$ and $w_1, w_2 > 0$, we have:*

$$\inf_{x\in\mathbb{R}}(w_1\phi(x) + w_2\phi(-x)) > \inf_{x:x(2w_1-1)\leq 0}(w_1\phi(x) + w_2\phi(-x)).$$

Proof [Proof of Lemma A.2] Denote the two (distinct) class probabilities with $w_1 + w_2 = 1$. Without loss of generality we distinguish two cases: $w_1 > w_2 > 0$ and $w_1 = 1, w_2 = 0$. First, consider the case when $w_1 > w_2 > 0$. Since the conditions of Theorem 2.3 hold, we know that the optimization problem (9) has a minimum, and hence by Proposition 2.2 we have that $\text{argmax}_{j\in\{1,2\}} \widehat{F}_j \subseteq \{1\}$. Hence it follows that $\widehat{F}_1 > 0, \widehat{F}_2 < 0$ at the minimum. This implies that inequality in Remark A.2 is strict.

Next assume that $w_1 = 1, w_2 = 0$. This case is not covered by our results as we assume that the probabilities are bounded away from 0. As we argued earlier ϕ is strictly decreasing on the set S' and by assumption $(-\infty, 0] \subsetneq S'$. Thus by:

$$\widehat{\mathbf{F}} = \text{argmin}_{\mathbf{F}:F_1+F_2=0}\, w_1\phi(F_1) + w_2\phi(F_2) = \text{argmin}_{\mathbf{F}:F_1+F_2=0}\, \phi(F_1),$$

we must have $\widehat{F}_1 > 0$ and hence $\phi(0) > \phi(\widehat{F}_1)$. This finishes the proof. ■

Lemma A.3 *Let $\mathbf{F}^{(m)}$ be defined as in iteration (16) starting from $\mathbf{F}^{(0)} = 0$. Then we must have $\mathbf{F}^{(m)} \in \Omega$ for all m, where $\Omega = \{\mathbf{F} = (F_1, \ldots, F_n) : F_j \in S, j = 1, ..., n\}$.*

Proof [Proof of Lemma A.3] If $S = \mathbb{R}$ there is nothing to prove. Assume that there exists $F^* \in \mathbb{R}$ such that $k(F^*) = 0$. We show the statement by induction. By definition $\mathbf{F}^{(0)} \in \Omega$. Assume that $\mathbf{F}^{(m-1)} \in \Omega$ for some $m \geq 1$. We now show that $\mathbf{F}^{(m)} \in \Omega$. To arrive at a contradiction, assume the contrary. Let $\mathcal{A} = \{j : F_j^{(m)} > F^*\} \neq \varnothing$. Define $F_j^{*(m)} = \mathbb{1}(j \in \mathcal{A})F_j^{(m-1)} + \mathbb{1}(j \notin \mathcal{A})F_j^{(m)}$. Since $\mathbf{F}^{(m-1)} \in \Omega$, it follows that $\mathbf{F}^{*(m)} \in \Omega$ and $\prod_{j=1}^n g(\mathbf{F}_j^{*(m)}) < 1$. More importantly, observe that for all $j \in \mathcal{A}$ we have:

$$0 = (g(F_j^{(m-1)}) - g(F_j^{*(m)}))k(F_j^{*(m)})w_j > (g(F_j^{(m-1)}) - g(F_j^{(m)}))k(F_j^{(m)})w_j,$$

as $g(F_j^{(m-1)}) \leq g(F^*) < g(F_j^{(m)})$ and $k(F_j^{(m)}) > k(F^*) = 0$, and hence:

$$\sum_{j=1}^n (g(F_j^{(m-1)}) - g(F_j^{*(m)}))k(F_j^{*(m)})w_j > \sum_{j=1}^n (g(F_j^{(m-1)}) - g(F_j^{(m)}))k(F_j^{(m)})w_j.$$

Define the index set $\mathcal{B} = \{j : F_j^{*(m)} < F_j^{(m-1)}\}$. Since $\prod_{j=1}^n g(\mathbf{F}_j^{*(m)}) < 1$ and $\mathbf{F}^{(m-1)} \in \Omega$ it follows that $\mathcal{B}$ is not empty and $\mathcal{A} \cap \mathcal{B} = \varnothing$. Next for $\lambda \in [0, 1]$ define for all j:

$$F_j^{*(m),\lambda} := [\mathbb{1}(j \in \mathcal{A}) + \mathbb{1}(j \notin \mathcal{A})\mathbb{1}(j \notin \mathcal{B})]F_j^{*(m)} + \mathbb{1}(j \in \mathcal{B})((1-\lambda)F_j^{*(m)} + \lambda F_j^{(m-1)}).$$

Note that when $\lambda = 0$, we have $F_j^{*(m),0} \equiv F_j^{*(m)}$. Now we show that for any $\lambda \in [0,1]$ the following inequality holds:

$$\sum_{j=1}^{n}(g(F_j^{(m-1)}) - g(F_j^{*(m),\lambda}))k(F_j^{*(m),\lambda})w_j \geq \sum_{j=1}^{n}(g(F_j^{(m-1)}) - g(F_j^{*(m)}))k(F_j^{*(m)})w_j. \quad (29)$$

For any $\lambda \in (0,1]$: $F_j^{*(m),\lambda} \neq F_j^{*(m),\lambda}$ iff $j \in \mathcal{B}$. Next note that for any j the function $(g(F_j^{(m-1)}) - g(x))k(x)w_j$ is an increasing function for $x \leq F_j^{(m-1)}$. The last two observations imply (29). Finally since $\prod_{j=1}^n g(F_j^{*(m),0}) = \prod_{j=1}^n g(F_j^{*(m)}) < 1$ and $\prod_{j=1}^n g(F_j^{*(m),1}) \geq \prod_{j=1}^n g(F_j^{(m-1)}) = 1$, by the continuity of g there exists a $\lambda \in (0,1]$ such that $\prod_{j=1}^n g(F_j^{*(m),\lambda}) = 1$. These facts and inequality (29) imply that $\mathbf{F}^{(m)}$ would not be a maximum in the iteration which is a contradiction. ■

Proof [Proof of Theorem 3.1] By construction we have that on the m^{th} iteration the value $\mathbf{F}^{(m)}$ satisfies $\prod_{j=1}^m g(F_j^{(m)}) = 1$, and Lemma A.3 guarantees that $\mathbf{F}^{(m)} \in \Omega$ for all m. Hence, since $F_j^{(m)}$ are viable values for $F_j^{(m+1)}$, the iteration also guarantees that:

$$\sum_{j=1}^{n}\{\phi(F_j^{(m)}) - \phi(F_j^{(m+1)})\}w_j \geq \sum_{j=1}^{n}\{g(F_j^{(m)}) - g(F_j^{(m+1)})\}k(F_j^{(m+1)})w_j \geq 0.$$

Now, from Remark A.1, $F_j^{(m+1)}$ lie on a compact set for all j, since for our starting point we have $\mathbf{F}_j^{(0)} \equiv 0 \in \Omega$. Therefore there must exist a subsequence $\{m_\ell, \ell = 1, ...\}$ such that $\mathbf{F}^{(m_\ell)}$ converges coordinate-wise on this subsequence, and denote with $\mathbf{F}^*$ its limit.

The function ϕ is continuous and hence we have that $\sum_{j=1}^n \phi(F_j^{(m_\ell)})w_j - \sum_{j=1}^n \phi(F_j^{(m_{\ell+1})})w_j \to 0$. However by the construction of our iteration, the sequences $\sum_{j=1}^n \phi(F_j^{(m_{\ell+1})})w_j$ are decreasing for all ℓ. Therefore we have that: $\sum_{j=1}^n \phi(F_j^{(m)})w_j - \sum_{j=1}^n \phi(F_j^{(m+1)})w_j \to 0$ holds for all m, not only on the subsequence. But this implies that $\sum_{j=1}^n (g(F_j^{(m)}) - g(F_j^{(m+1)}))k(F_j^{(m+1)})w_j \to 0$, which again by the construction is non-negative for all m. Take m_ℓ in place of m in the limit above, and let L be the set of all limit points $\lim_{\ell\to\infty} \mathbf{F}^{(m_\ell+1)}$. By our construction we have the following inequality holding for any point $\mathbf{F}^l \in L$:

$$0 = \sum_{j=1}^{n}\{g(F_j^*) - g(F_j^l)\}k(F_j^l)w_j \geq \sum_{j=1}^{n}\{g(F_j^*) - g(F_j)\}k(F_j)w_j, \quad (30)$$

for any $\mathbf{F} \in \Omega$ with $\prod_{j=1}^n g(F_j) = 1$. Just as in the proof of Theorem 2.1 select $\widetilde{\mathbf{F}}$ so that $g(F_j^*)k(\widetilde{F}_j)w_j = \mathcal{C}$ for all j for some $\mathcal{C} < 0$. By the AM-GM inequality we get:

$$\sum_{j=1}^{n} g(\widetilde{F}_j)\{-k(\widetilde{F}_j)\}w_j \geq n\left[\prod_{j=1}^{n} g(\widetilde{F}_j)\{-k(\widetilde{F}_j)\}w_j\right]^{n^{-1}} = n\left[\prod_{j=1}^{n} g(F_j^*)\{-k(\widetilde{F}_j)\}w_j\right]^{n^{-1}}$$
$$= \sum_{j=1}^{n} g(F_j^*)\{-k(\widetilde{F}_j)\}w_j = -n\mathcal{C}.$$

Now by (30) it follows that equality must be achieved in the preceding display, which implies that $g(F_j^*)k(\widetilde{F}_j)w_j = \mathcal{C} = g(\widetilde{F}_j)k(\widetilde{F}_j)w_j$ and yields $\widetilde{F}_j = F_j^*$ for all j. Hence $g(F_j^*)k(F_j^*)w_j = \mathcal{C}$ for all j.

Thus we showed that on subsequences the iteration converges to points satisfying the equality described above. We are left to show, that all these subsequences converge to the same point. Next, take equation (30). By what we showed it follows that for any $\mathbf{F}^l \in L$, we have that $g(F_j^l)k(F_j^l)w_j = \mathcal{C}^l$ for some $\mathcal{C}^l < 0$. Then we have:

$$\sum_{j=1}^{n} g(F_j^*)\{-k(F_j^l)\}w_j \geq n\left[\prod_{j=1}^{n} g(F_j^*)\{-k(F_j^l)\}w_j\right]^{n^{-1}} = n\left[\prod_{j=1}^{n} g(F_j^l)\{-k(F_j^l)\}w_j\right]^{n^{-1}}$$
$$= \sum_{j=1}^{n} g(F_j^l)\{-k(F_j^l)\}w_j = -n\mathcal{C}^l.$$

Equation (30) implies that the above inequality is in fact equality which shows that:

$$g(F_j^*)k(F_j^l) = g(F_j^l)k(F_j^l) \quad \text{for all } j.$$

Thus since $k(F_j^l) \neq 0$ (recall that all values on the iteration $\mathbf{F}^{(m)} \in \Omega$) we conclude that $g(F_j^*) = g(F_j^l)$, and hence $\mathbf{F}^* = \mathbf{F}^l$. This shows that for any converging subsequence m_ℓ the limiting value coincides with that of the sequence $m_\ell + 1$, which finishes our proof. ■

Proof [Proof of Proposition 3.2.] It is sufficient to show that for all $\mathbf{F} \in \mathcal{G}^*$ we have:

$$\sum_{i=1}^{N} \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \geq 0.$$

The condition above is sufficient, because of the looping closure of $\mathcal{G}$. Writing the inequality for all "looped" versions of $\mathbf{F}$ and noting that they sum up to 0, gives us that the inequality is in fact an equality.

Note that with each iteration (18), we decrease the value of the target function. This can be seen by the following inequality:

$$\sum_{i=1}^{N}\phi(\mathbf{Y}_{C_i}^T\mathbf{F}^{(m-1)}(\mathbf{X}_i)) - \sum_{i=1}^{N}\phi(\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i)) \geq \sum_{i=1}^{N}[\exp(-\beta\mathbf{Y}_{C_i}^T\mathbf{F}(\mathbf{X}_i)) - 1]\dot{\phi}(\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i)) \geq 0,$$

where $\mathbf{F}^{(m)} = \mathbf{F}^{(m-1)} + \beta\mathbf{F}$. As a remark, the inequality in the preceding display holds, since ϕ is decreasing and thus by (6) we have $S = \mathbb{R}$.

Take a limiting point[11] $\mathbf{F}^{(\infty)}$ of iteration (18), where it is possible having coordinates of $\mathbf{F}^{(\infty)}(\mathbf{X}_i)$ equal to $\pm\infty$ for some i. Since ϕ is bounded from below, by our previous observation we have that for any $\beta \geq 0$ and $\mathbf{F} \in \mathcal{G}^*$:

$$\sum_{i=1}^{N}\phi(\mathbf{Y}_{C_i}^T\mathbf{F}^{(\infty)}(\mathbf{X}_i)) - \sum_{i=1}^{N}\phi(\mathbf{Y}_{C_i}^T\mathbf{F}^{(\infty)}(\mathbf{X}_i) + \beta\mathbf{Y}_{C_i}^T\mathbf{F}(\mathbf{X}_i)) \leq 0.$$

11. The existence of a limiting point is guaranteed as any sequence contains a monotone subsequence.

Let $\mathcal{A} = \{i : |\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)| \neq \infty\}$. Then the latter inequality also implies that:

$$\sum_{i \in \mathcal{A}} \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) - \sum_{i \in \mathcal{A}} \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)) \leq 0.^{12}$$

Applying inequality (6) the above implies:

$$\sum_{i \in \mathcal{A}} [\exp(-\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)) - 1] \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)) \leq 0,$$

and after a Taylor expansion of the exponent, and division by $\beta \geq 0$ we get:

$$\begin{aligned}
&\sum_{i \in \mathcal{A}} -\mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \\
&+ \sum_{i \in \mathcal{A}} -\mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) [\dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)) - \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i))] \\
&+ O(\beta) \sum_{i \in \mathcal{A}} \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)) \leq 0.
\end{aligned}$$

Letting $\beta \to 0$, by the continuity of $\dot{\phi}$ we get:

$$\sum_{i \in \mathcal{A}} \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \geq 0. \tag{31}$$

Next we argue that $\dot{\phi}(+\infty) = \lim_{x \to +\infty} \dot{\phi}(x) = 0$. As stated in the main text $\dot{\phi}(x) = k(x)e^x$. Let $K = \inf_{x \in \mathbb{R}} \phi(x)$. For any $\varepsilon > 0$, take a point x' such that $\phi(x') - K \leq \varepsilon$. Then for any $x \in \mathbb{R}$, by (6):

$$\varepsilon \geq \phi(x') - \phi(x) \geq (e^{x'} - e^x) k(x),$$

and thus $\varepsilon - e^{x'} k(x) \geq -\dot{\phi}(x) \geq 0$. Taking the limit $x \to +\infty$ and letting $\varepsilon \to 0$ shows that $\dot{\phi}(+\infty) = 0$.

Now consider two cases for ϕ. Suppose that ϕ is unbounded from above. We argue that $\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i) \neq -\infty$ for all i. Since we start form the point 0, and as we argued we are decreasing the target function we have that:

$$N\phi(0) \geq \sum_{i=1}^{N} \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \geq \max_i \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) + (N-1)K,$$

and hence $\max_i \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \leq N\phi(0) - (N-1)K$. Since ϕ is decreasing and unbounded from above it follows that $\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i) \neq -\infty$ for all i. In the second case suppose that ϕ is bounded from above, and let $M = \sup_{x \in \mathbb{R}} \phi(x)$. We show that $\dot{\phi}(-\infty) = 0$. For any $\varepsilon > 0$ take x so that $\varepsilon \geq M - \phi(x)$. Applying (6) for any $x' \in \mathbb{R}$ yields:

$$\varepsilon \geq \phi(x') - \phi(x) \geq (e^{x'} - e^x) k(x).$$

This gives $\varepsilon - e^{x'} k(x) \geq -\dot{\phi}(x) \geq 0$. Taking $x' \to -\infty$ gives that $\varepsilon \geq -\dot{\phi}(x) \geq 0$ for any x such that $\varepsilon \geq M - \phi(x)$. Since ϕ is decreasing we are allowed to take the limit $x \to -\infty$, and

12. Observe that since ϕ is bounded from below the values of ϕ at infinite points of the iteration i.e. $\phi(\pm\infty)$ have to be bounded.

taking $\varepsilon \to 0$ shows that $\dot{\phi}(-\infty) = 0$. In any case, all of the above arguments imply that we can substitute $\mathcal{A}$ in (31) to the whole index set $\{1, \ldots, N\}$, to finally conclude:

$$\sum_{i=1}^{N} \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(\infty)}(\mathbf{X}_i)) \geq 0,$$

for all $\mathbf{F} \in \mathcal{G}^*$. As argued in the beginning the looping closure gives us that in fact the "$\geq$" can be replaced with "=". This concludes the proof.

■

Proof [Proof of Proposition 3.3] First consider the case when $I = N$. Denote with $e_1^1, e_1^2, \ldots, e_1^N$ the positive coordinates of $\mathbf{e}_1$. For any vector $\mathbf{v}$ which is a solution to $\mathbf{D}^T \boldsymbol{\alpha} = \mathbf{v}$ we must have $\sum_{i=1}^N e_1^i v_i = 0$. Clearly then, if $\mathbf{v}$ is non-zero some of the v_i need to be negative. Let $l = \min v_i$. We know that $l < 0$. This immediately implies an upper bound on the maximal positive v_i — $\max_j v_i \leq |l|[\sum_{i=1}^N e_1^i / \min_j e_1^j - 1]$. We now show that $|l|$ is bounded, for all vectors $\mathbf{v}$ such that $\sum_{i=1}^N \phi(v_i) \leq N\phi(0)$. Note that $\sum_{i=1}^N \phi(v_i) \geq \phi(l) + (N-1)\phi(|l|[\sum_{i=1}^N e_1^i / \min_j e_1^j - 1])$, and thus (22) gives that $|l|$ is bounded. This in turn shows that all v_i are bounded as well.

We next consider the case where $I < N$. Without loss of generality, upon rearrangement, we can assume that the positive coordinates of $\mathbf{e}_1$ are located at the first I places. Let $\mathbf{E}_{N \times s} = (\mathbf{e}_1, \ldots, \mathbf{e}_s)$. Let $\widetilde{\mathbf{E}}_{(N-I) \times (s-1)}$ denote the sub-matrix corresponding to the 0 entries of $\mathbf{e}_1$ (excluding $\mathbf{e}_1$) (see (32) for a visualization). Note that the matrix $\widetilde{\mathbf{E}}$ cannot be of full column rank, because otherwise we would have that a vector with positive coordinates is inside the column space which is a contradiction (we can always scale it by a small number and add to $\mathbf{e}_1$). Thus we can eliminate all extra columns that do not contribute to the rank of $\widetilde{\mathbf{E}}$, by doing a linear manipulation on the columns of the whole matrix $\mathbf{E}$ (see (32)). In doing so, we can eliminate extra columns of the matrix $\widetilde{\mathbf{E}}$ so that we end up with a $\widetilde{\widetilde{\mathbf{E}}}$ matrix where the number of non-zero columns matches the rank, and some columns of $\mathbf{E}$ have 0 coefficients on the lower part. Here, observe that the columns of $\mathbf{E}$ with 0 sub-columns in $\widetilde{\mathbf{E}}$, are part of the space $\text{row}(\mathbf{D}_-)^\perp$, where $\mathbf{D}_-$ corresponds to the matrix $\mathbf{D}$ with observations corresponding to 0's of $\mathbf{e}_1$ removed.

We next note that if we discard the observations corresponding to 0 coordinates in e_1, and optimize the problem on the rest of the observations we will obtain some optimal solution $\mathbf{v} = (\hat{v}_1, \ldots, \hat{v}_I)^T$, the entries of which are bounded as argued in the first case. We next show that we can populate the vector $\mathbf{v}$ with positive numbers $p_1, \ldots, p_{N-I}$ to $\boldsymbol{\nu} = (\hat{v}_1, \ldots, \hat{v}_I, p_1, \ldots, p_{N-I})^T$, so that $\boldsymbol{\nu}$ is "perpendicular" to the matrix $\mathbf{E}$ (i.e. $\mathbf{E}^T \boldsymbol{\nu} = 0$), and thus can be written in the form $\mathbf{D}^T \boldsymbol{\alpha}$. Moreover, we will show that $p_1, \ldots, p_{N-I}$, can become arbitrarily large, which will complete the proof.

$$\mathbf{E} = \begin{array}{c} \\ I \\ \\ N-I \end{array} \begin{array}{|c|c|} \hline \mathbf{e}_1 & \mathbf{e}_2 \ \ldots \ \mathbf{e}_s \\ \hline e_1^1 \\ \vdots \\ e_1^I & \\ \hline 0 \\ \vdots \\ 0 & \widetilde{\mathbf{E}} \\ \hline \end{array} \quad \rightarrow \quad \begin{array}{|c|c|c|} \hline \mathbf{e}_1 & \ldots \ \widetilde{\mathbf{e}}_{l+1} & \widetilde{\mathbf{e}}_{l+2} \ \ldots \ \widetilde{\mathbf{e}}_s \\ \hline e_1^1 \\ \vdots \\ e_1^I & \mathbf{G} & \\ \hline 0 & & 0 \ \ldots \ 0 \\ \vdots & \widetilde{\widetilde{\mathbf{E}}} & \vdots \ \ldots \ \vdots \\ 0 & & 0 \ \ldots \ 0 \\ \hline \end{array} \tag{32}$$

Note that the only part of the matrix $\mathbf{E}$ that would be potentially non-zero upon multiplication by $\boldsymbol{\nu}$ would be the part corresponding to the non-zero parts of $\widetilde{\mathbf{E}}$, because as we argued earlier the columns of $\mathbf{E}$ with 0 sub-columns in $\widetilde{\mathbf{E}}$ belong to $\text{row}(\mathbf{D}_-)^\perp$ and on the other hand $\mathbf{v} \in \text{row}(\mathbf{D}_-)$. Denote with $\widetilde{\widetilde{\mathbf{E}}}_{(N-I)\times l}$ the full-rank sub-matrix of $\widetilde{\mathbf{E}}$, where l is the rank of $\widetilde{\mathbf{E}}$, and let $\mathbf{G}_{I\times l}$ be the sub-matrix of $\mathbf{E}$ above $\widetilde{\widetilde{\mathbf{E}}}$ (see (32)). Clearly $l < N - I$ as otherwise there is a positive vector in the column space, and we argued previously that would be a contradiction with the maximality property of $\mathbf{e}_1$. We need to find a positive vector $\mathbf{p}$ such that $(\widetilde{\widetilde{\mathbf{E}}}_{(N-I)\times l})^T \mathbf{p}_{(N-I)\times 1} = -(\mathbf{G}_{I\times l})^T \mathbf{v}_{I\times 1} = \mathbf{K}_{l\times 1}$. Therefore the proof will be completed, if we can find arbitrary large positive vectors $\mathbf{p}$ solving the system $\widetilde{\widetilde{\mathbf{E}}}^T \mathbf{p} = \mathbf{K}$, where $l < N - I$ and $\widetilde{\widetilde{\mathbf{E}}}^T$ has the property that any non-zero linear combination of its rows results into a vector with at least one positive and one negative entry.

Since $\widetilde{\widetilde{\mathbf{E}}}$ is full-rank and $l < N - I$, the linear system $\widetilde{\widetilde{\mathbf{E}}}^T \mathbf{p} = \mathbf{K}$ has a solution. Consider the homogeneous system $\widetilde{\widetilde{\mathbf{E}}}^T \mathbf{p} = 0$. We will show that the homogeneous equation admits arbitrary large positive solutions, which would complete the proof. Fix the value of the i^{th} parameter to be 1. The system then becomes $\widetilde{\widetilde{\mathbf{E}}}^T_{-i} \mathbf{p}_{-i} = -\widetilde{\widetilde{\mathbf{e}}}_i$, where by indexing with $-i$ we mean removing the i^{th} column or element and $\widetilde{\widetilde{\mathbf{e}}}_i$ is the i^{th} column of $\widetilde{\widetilde{\mathbf{E}}}^T$. Next we apply Farkas's lemma to show that the last equation has a non-negative solution. Assume that there is a vector $\mathbf{y}_{l\times 1}$ such that $\widetilde{\widetilde{\mathbf{E}}}_{-i}\mathbf{y} \geq 0$ (coordinate-wise) and $-\widetilde{\widetilde{\mathbf{e}}}_i^T \mathbf{y} < 0$. This is clearly a violation with the property that $\widetilde{\widetilde{\mathbf{E}}}$ satisfies. Therefore by Farkas's lemma the equation $\widetilde{\widetilde{\mathbf{E}}}^T_{-i} \mathbf{p}_{-i} = -\widetilde{\widetilde{\mathbf{e}}}_i$ has a non-negative solution. Since we can achieve this for any index i, averaging these solutions yields a positive solution to the homogeneous system $\widetilde{\widetilde{\mathbf{E}}}^T \mathbf{p} = 0$, and thus this system admits arbitrarily large positive solutions.

■

Proof [Proof of Theorem 3.4] Without loss of generality for the purposes of the proof we will consider $\mathcal{C}_+ = 1$ and $\mathcal{C}_- = -1/(n-1)$ (it's equivalent to rescaling the β in the iteration).

By the iteration's construction we know:

$$\varepsilon_m - \varepsilon_{m+1} \geq \max_{\beta\geq 0, \mathbf{F}\in\mathcal{G}^*} \sum_{i=1}^N \{e^{-\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)} - 1\} \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\}.$$

Note that we have the following simple inequality holding for $\exp(-x) \leq 1 - x + x^2$ for values of $-1/2 \leq x \leq 1/2$. Since $|\mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)| \leq \mathcal{C}_+ = 1$ and ϕ is decreasing, for values of $0 \leq \beta \leq 1/2$ we have that:

$$\begin{aligned}
&\sum_{i=1}^N \{e^{-\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)} - 1\} \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\} \\
&\geq -\sum_{i=1}^N (\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\}.
\end{aligned}$$

Let L denote the Lipschitz constant of $\dot{\phi}$ on the set $\mathcal{S}$. Consequently we have:

$$\begin{aligned}
-\sum_{i=1}^N (\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\} &= \sum_{i=1}^N -(\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\} \\
&- \sum_{i=1}^N (\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2)[\dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i) + \beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\} - \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\}] \geq \\
&\sum_{i=1}^N -(\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\} - L\left|\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2\right| \left|\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\right| \geq \\
&\sum_{i=1}^N -(\beta \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) - \beta^2) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\} - \frac{3}{2} LN\beta^2.
\end{aligned}$$

Thus we have established:

$$\varepsilon_m - \varepsilon_{m+1} \geq \beta \left(\sum_{i=1}^N (-\mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) + \beta) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\} - \frac{3}{2} LN\beta \right),$$

for any $0 \leq \beta \leq 1/2$. We select β so that we maximize the RHS in the expression above. It turns out that this happens for:

$$\beta_0 = \frac{\frac{1}{2}\sum_{i=1}^N \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\}}{-\frac{3}{2}LN + \sum_{i=1}^N \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\}}.$$

Since $\dot{\phi}$ is always negative and as we mentioned $\left|\mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i)\right| \leq 1$, provided that the numerator is ≤ 0, we have that $0 \leq \beta_0 \leq 1/2$. Then we would have:

$$\varepsilon_m - \varepsilon_{m+1} \geq -\frac{1}{2}\beta_0 \sum_{i=1}^N \mathbf{Y}_{C_i}^T \mathbf{F}(\mathbf{X}_i) \dot{\phi}\{\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)\}. \tag{33}$$

Next we show that there exists a classifier, such that the above expression is strictly positive, which will also ensure that $0 \leq \beta_0 \leq 1/2$ is in the correct range. Denote with B the total number of classifiers in the bag. Consider the representation $\mathbf{F}^*(\cdot) - \mathbf{F}^{(m)}(\cdot) = \sum_{j=1}^B \alpha_j \mathbf{F}_j(\cdot)$. Here the $\boldsymbol{\alpha}$ vector is any vector that yields a correct representation (note that we will have many possible $\boldsymbol{\alpha}$ vectors, in the case when $B > N$).

By convexity of ϕ we have:

$$\begin{aligned}
-\varepsilon_m = \sum_{i=1}^N \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^*(\mathbf{X}_i)) - \phi(\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)) &\geq \sum_{i=1}^N [\mathbf{Y}_{C_i}^T \mathbf{F}^*(\mathbf{X}_i) - \mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)] \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)) \\
&= \sum_{j=1}^B \sum_{i=1}^N \alpha_j \mathbf{Y}_{C_i}^T \mathbf{F}_j(\mathbf{X}_i) \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)).
\end{aligned}$$

By the pigeonhole principle it is clear that there exists an index $j \in \{1, \ldots, B\}$ such that:

$$\frac{\varepsilon_m}{B \max_j |\alpha_j|} \leq \frac{\varepsilon_m}{B|\alpha_j|} \leq -\text{sign}(\alpha_j) \sum_{i=1}^N \mathbf{Y}_{C_i}^T \mathbf{F}_j(\mathbf{X}_i) \dot{\phi}(\mathbf{Y}_{C_i}^T \mathbf{F}^{(m)}(\mathbf{X}_i)).$$

Now if $\text{sign}(\alpha_j) = 1$ we already have a "decent" lower bound. Otherwise if $\text{sign}(\alpha_j) = -1$, using the fact that the loop closed classifiers wrt to $\mathbf{F}_j$ sum up to 0, we can claim that for one of the looped classifiers $\mathbf{F}_j^l$ we would have a bound:

$$\frac{\varepsilon_m}{B(n-1)\max_j|\alpha_j|} \leq -\sum_{i=1}^{N}\mathbf{Y}_{C_i}^T\mathbf{F}_j^l(\mathbf{X}_i)\dot{\phi}(\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i)).$$

So that in both cases we established the existence of a classifier such that $\mathbf{F} \in \mathcal{G}^*$ and:

$$\frac{\varepsilon_m}{B(n-1)\max_j|\alpha_j|} \leq -\sum_{i=1}^{N}\mathbf{Y}_{C_i}^T\mathbf{F}(\mathbf{X}_i)\dot{\phi}(\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i))$$

We then know from (33) that:

$$\begin{aligned}\varepsilon_m - \varepsilon_{m+1} &\geq -\frac{1}{2}\beta_0\sum_{i=1}^{N}\mathbf{Y}_{C_i}^T\mathbf{F}(\mathbf{X}_i)\dot{\phi}\{\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i)\}\\ &\geq \frac{1}{4}\frac{\varepsilon_m^2}{B^2(n-1)^2\max_j\alpha_j^2(\frac{3}{2}LN - \sum_{i=1}^{N}\dot{\phi}\{\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i)\})}.\end{aligned}$$

Notice that the derivative is bounded on the set $\mathcal{S}$ and therefore collapsing all constants above into one constant say T we get the following:

$$\varepsilon_m - \varepsilon_{m+1} \geq \frac{\varepsilon_m^2}{T\max_j\alpha_j^2}.$$

Here T depends on the number of classifiers, number of classes, and the bound on the first derivative $\dot{\phi}$ on the set $\mathcal{S}$. We will proceed to bound the $\max_j\alpha_j^2$ for some of the representations from above.

Because on the set $\mathcal{S}$, ϕ is also strongly convex (with a constant say l), we have the following:

$$\begin{aligned}\varepsilon_m &= \sum_{i=1}^{N}\phi(\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i)) - \sum_{i=1}^{N}\phi(\mathbf{Y}_{C_i}^T\mathbf{F}^*(\mathbf{X}_i))\\ &\geq \sum_{i=1}^{N}[\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i) - \mathbf{Y}_{C_i}^T\mathbf{F}^*(\mathbf{X}_i)]\dot{\phi}(\mathbf{Y}_{C_i}^T\mathbf{F}^*(\mathbf{X}_i))\\ &+ l\sum_{i=1}^{N}\left(\sum_{j=1}^{B}\alpha_j\mathbf{Y}_{C_i}^T\mathbf{F}_j(\mathbf{X}_i)\right)^2.\end{aligned}$$

The expression $\sum_{i=1}^{N}[\mathbf{Y}_{C_i}^T\mathbf{F}^{(m)}(\mathbf{X}_i) - \mathbf{Y}_{C_i}^T\mathbf{F}^*(\mathbf{X}_i)]\dot{\phi}(\mathbf{Y}_{C_i}^T\mathbf{F}^*(X_i))$ is 0, as $\mathbf{F}^*$ is the minimum, ϕ is convex and the classifier bag is closed under looping. Let $\mathbf{D} = \{\mathbf{Y}_{C_i}^T\mathbf{F}_j(\mathbf{X}_i)\}_{j,i}$ is the $B\times N$ matrix, each entry of which is either $\mathcal{C}_+$ or $\mathcal{C}_-$. Let the rank of $\mathbf{D}$ is $r \leq \min(N, B)$. We then have $\sum_{i=1}^{N}\left(\sum_{j=1}^{B}\alpha_j\mathbf{Y}_{C_i}^T\mathbf{F}_j(\mathbf{X}_i)\right)^2 = \boldsymbol{\alpha}^T\mathbf{D}\mathbf{D}^T\boldsymbol{\alpha}$. Since, all the bounds above are true for any of the $\boldsymbol{\alpha}$ representations, we could have picked the representation corresponding to the $r\times N$ sub matrix of $\mathbf{D}$, $\mathbf{D}_r$ with rank r for which the smallest eigenvalue of $\mathbf{D}_r\mathbf{D}_r^T$ is the largest. Let this eigenvalue be $\lambda_r > 0$. For this eigenvalue and this choice of $\boldsymbol{\alpha}$ we clearly have $\boldsymbol{\alpha}^T\mathbf{D}\mathbf{D}^T\boldsymbol{\alpha} = \boldsymbol{\alpha}^T\mathbf{D}_r\mathbf{D}_r^T\boldsymbol{\alpha} \geq \lambda_r||\boldsymbol{\alpha}||_2^2 \geq \lambda_r\max_j\alpha_j^2$. (in the second equality we abuse notation deleting zeros from the $\boldsymbol{\alpha}$). Consequently we get:

$$\varepsilon_m \geq l\lambda_r\max_j\alpha_j^2.$$

Thus:

$$\begin{aligned}\varepsilon_{m+1} &\leq \varepsilon_m - \frac{\varepsilon_m^2}{T\max_j\alpha_j^2}\\ &\leq \varepsilon_m\left(1 - \frac{l\lambda_r}{T}\right).\end{aligned}$$

Since both $\varepsilon_{m+1}, \varepsilon_m \geq 0$ we must have $1 - \frac{l\lambda_r}{T} \geq 0$. Furthermore, by construction we have $\frac{l\lambda_r}{T} > 0$, which concludes the proof. ■

Remark A.3 *Since $\mathcal{S}$ is bounded we did not need to require that the first derivative — $\dot{\phi}$ is bounded since we already assumed its Lipschitz continuity.*

References

Caroline A Abbott, Rayaz A Malik, Ernest RE van Ross, Jai Kulkarni, and Andrew JM Boulton. Prevalence and characteristics of painful diabetic neuropathy in a large community-based diabetic population in the uk. *Diabetes care*, 34(10):2220–2224, 2011.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Ronan Collobert, Fabian Sinz, Jason Weston, and Léon Bottou. Trading convexity for scalability. In *Proceedings of the 23rd international conference on Machine learning*, pages 201–208. ACM, 2006.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics*, 28(2):337–407, 2000.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, pages 1189–1232, 2001.

Bradley S Galer, Ann Gianas, and Mark P Jensen. Painful diabetic polyneuropathy: epidemiology, pain description, and quality of life. *Diabetes research and clinical practice*, 47(2):123–128, 2000.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2009.

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

Yi Lin. Support vector machines and the bayes rule in classification. *Data Mining and Knowledge Discovery*, 6:259–275, 2002.

Yi Lin. A note on margin-based loss functions in classification. *Statistics & Probability Letters*, 68(1):73 – 82, 2004. ISSN 0167-7152.

Yufeng Liu. Fisher consistency of multicategory support vector machines. In *International Conference on Artificial Intelligence and Statistics*, pages 291–298, 2007.

Catherine L Martin, James Albers, William H Herman, Patricia Cleary, Barbara Waberski, Douglas A Greene, Martin J Stevens, and Eva L Feldman. Neuropathy among the diabetes control and complications trial cohort 8 years after trial completion. *Diabetes care*, 29 (2):340–344, 2006.

Hamed Masnadi-Shirazi and Nuno Vasconcelos. On the Design of Loss Functions for Classification: theory, robustness to outliers, and SavageBoost. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *NIPS*, pages 1049–1056. Curran Associates, Inc., 2008.

Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. Boosting algorithms as gradient descent in function space. *NIPS*, 1999.

Shawn N Murphy, Michael E Mendis, David A Berkowitz, Isaac Kohane, and Henry C Chueh. Integration of clinical and genetic data in the i2b2 architecture. In *AMIA Annual Symposium Proceedings*, volume 2006, page 1040. American Medical Informatics Association, 2006.

Shawn N Murphy, Griffin Weber, Michael Mendis, Vivian Gainer, Henry C Chueh, Susanne Churchill, and Isaac Kohane. Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). *Journal of the American Medical Informatics Association*, 17(2):124–130, 2010.

Yurii Nesterov. *Introductory lectures on convex optimization*, volume 87. Springer Science & Business Media, 2004.

Gerard Said. Diabetic neuropathya review. *Nature Clinical Practice Neurology*, 3(6):331–340, 2007.

Xiaotong Shen, George C Tseng, Xuegong Zhang, and Wing Hung Wong. On ψ-learning. *Journal of the American Statistical Association*, 98(463):724–734, 2003.

Ambuj Tewari and Peter L Bartlett. On the consistency of multiclass classification methods. *The Journal of Machine Learning Research*, 8:1007–1025, 2007.

PK Thomas and SG Eliasson. Diabetic neuropathy. *Peripheral neuropathy*, 2:1773–1810, 1984.

Zhu Wang. Multi-class hingeboost. method and application to the classification of cancer types using gene expression data. *Methods of information in medicine*, 51(2):162–167, 2012.

Ji Zhu, Saharon Rosset, Hui Zou, and Trevor Hastie. Multi-class adaboost. *Statistics and Its Interface*, 2:349–360, 2009.

D Ziegler, FA Gries, M Spüler, and F Lessmann. The epidemiology of diabetic neuropathy. *Journal of diabetes and its complications*, 6(1):49–57, 1992.

Hui Zou, Ji Zhu, and Trevor Hastie. New multicategory boosting algorithms based on multicategory fisher-consistent losses. *The Annals of Applied Statistics*, pages 1290–1306, 2008.

Journal of Machine Learning Research 17 (2016) 1-19 Submitted 10/14; Published 4/16

Exact Inference on Gaussian Graphical Models of Arbitrary Topology using Path-Sums

P.-L. Giscard PIERRE-LOUIS.GISCARD@YORK.AC.UK
Department of Computer Science
University of York
Deramore Lane, York, YO10 5GH, United Kingdom

Z. Choo ZHENG.CHOO@STATS.OX.AC.UK
Department of Statistics
University of Oxford
1 South Parks Road, Oxford OX1 3TG, UK

S. J. Thwaite SIMON.THWAITE@PHYSIK.UNI-MUENCHEN.DE
Department of Physics and Arnold Sommerfeld Center for Theoretical Physics,
Ludwig-Maximilians-Universität München
Theresienstraße 37, 80333 Munich, Germany

D. Jaksch D.JAKSCH1@PHYSICS.OX.AC.UK
Department of Physics
University of Oxford
Clarendon Laboratory, Oxford OX1 3PU, United Kingdom.
Centre for Quantum Technologies
National University of Singapore
3 Science Drive 2, Singapore 117543

Editor: Alexander Ihler

Abstract

We present the path-sum formulation for exact statistical inference of marginals on Gaussian graphical models of arbitrary topology. The path-sum formulation gives the covariance between each pair of variables as a branched continued fraction of finite depth and breadth. Our method originates from the closed-form resummation of infinite families of terms of the walk-sum representation of the covariance matrix. We prove that the path-sum formulation always exists for models whose covariance matrix is positive definite: i.e. it is valid for both walk-summable and non-walk-summable graphical models of arbitrary topology. We show that for graphical models on trees the path-sum formulation is equivalent to Gaussian belief propagation. We also recover, as a corollary, an existing result that uses determinants to calculate the covariance matrix. We show that the path-sum formulation formulation is valid for arbitrary partitions of the inverse covariance matrix. We give detailed examples demonstrating our results.

Keywords: Gaussian graphical models, belief propagation, path-sum, walk-sum, graphs of arbitrary topology, block matrices

1. Introduction

A Gaussian Markov random field (GMRF) is a random vector that follows a multivariate normal (or Gaussian) distribution and satisfies *conditional independence* assumptions, hence the *Markov* property. If X_1, X_2, X_3 are random variables with a joint probability density function (or joint probability mass function in a discrete case), we say that X_1 is *conditionally independent* of X_2 given X_3, denoted $X_1 \perp\!\!\!\perp X_2|X_3$, if (Lauritzen, 1996)

$$f(x_1, x_2|x_3) = f(x_1|x_3)f(x_2|x_3).$$

Here we use f as a generic symbol for the probability density function of the random variables corresponding to its arguments. GMRFs have a simple interpretation and find their applications, for example, in image analysis, spatial statistics, structural time series analysis and analysis of longitudinal and survival data (Rue and Held, 2005).

Consider a random vector $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ following a multivariate normal distribution with mean μ and covariance matrix Σ, denoted $\mathbf{X} \sim \mathcal{N}(\mu, \Sigma)$. The probability density function of $\mathbf{X}$ is given as

$$f(x) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma)}} \exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right].$$

Here Σ is a symmetric and positive definite matrix. We write $A \succ 0$ to denote that the matrix A is positive definite. Alternatively the probability density function of $\mathbf{X}$ can be expressed in a *canonical form*

$$\begin{aligned} f(x) &= g(x)\exp\left[-\frac{1}{2}x^TJx + h^Tx - k(\mu,\Sigma)\right] \\ &\propto \exp\left[-\frac{1}{2}x^TJx + h^Tx\right], \end{aligned} \tag{1}$$

where $J = \Sigma^{-1}$, $h = J\mu$, $g(x) = (2\pi)^{-\frac{n}{2}}$ and $k(\mu,\Sigma) = \frac{1}{2}\mu^TJ\mu - \frac{1}{2}\ln(\det(J))$. We call J the *information matrix* (or precision matrix) and h the *potential vector*.

One advantage of using the form of parametrization in (1) is that J admits a graphical model in the following sense. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with the vertex set $\mathcal{V}$ and the set of edges $\mathcal{E}$. Let $\mathbf{X}_{\setminus ij}$ denote the set of variables with X_i and X_j removed from $\mathbf{X}$. If $J = (J_{ij})_{i,j\in\mathcal{V}}$ is positive definite, then for $i, j \in \mathcal{V}$, where $i \neq j$, we have (Proposition 5.2, Lauritzen, 1996)

$$X_i \perp\!\!\!\perp X_j|\mathbf{X}_{\setminus ij} \Leftrightarrow J_{ij} = 0.$$

Then we define a GMRF as follows.

Definition 1 (Definition 2.1, Rue and Held, 2005) *A random vector* $\mathbf{X}$ *is called a GMRF* with respect to a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *with information matrix* J *and potential vector* h *if and only if its density has the form of (1) and* $J_{ij} \neq 0 \Leftrightarrow (i,j) \in \mathcal{E}$*, for all* i *and* j*.*

It is known that X satisfies the Markov property on $\mathcal{G}$ (see Theorem 2.4 by Rue and Held, 2005)[1]. Note that by Definition 1, there is a one-to-one correspondence between the structure of J and the structure of $\mathcal{G}$. Most information matrices J for GMRFs are sparse, i.e. there are $O(n)$ non-zero entries in J. The sparsity structure of J facilitates the simulation of GMRFs through J (Rue and Held, 2005).

Another advantage of using the canonical form concerns the estimation of $\mathbf{X}$ given noisy observations $\mathbf{Y}$ (Johnson, 2002). Indeed, assume that $\mathbf{Y} = C\mathbf{X} + \varepsilon$, where C is an $n \times n$ real matrix and $\varepsilon \sim \mathcal{N}(0, M)$. Then the conditional distribution of $\mathbf{X}|\mathbf{Y}$ is given by

$$\begin{aligned} f(x|y) &= \frac{f(y|x)f(x)}{f(y)} \\ &\propto f(y|x)f(x) \\ &\propto \exp\left[-\frac{1}{2}x^T\tilde{J}x + \tilde{h}^T x\right], \end{aligned}$$

where $\tilde{J} = J + C^T M^{-1} C$ and $\tilde{h} = h + C^T M^{-1} y$. Thus given noisy observations, one only needs to update the information matrix and the potential vector to construct a graphical model for $f(x|y)$. For simplicity, we use J and h to denote the parameters after absorption of observations.

Given the canonical form, one needs both the covariance matrix $\Sigma = J^{-1}$ and the mean vector $\mu = J^{-1}h$ to obtain the marginal distributions of $\mathbf{X}$ or $\mathbf{X}|\mathbf{Y}$. Since knowing J^{-1} is sufficient to recover μ, we focus our efforts on calculating J^{-1}. Direct inversion of J has complexity $O(n^3)$ and does not exploit the sparsity of J.[2] In a simple situation where the graph $\mathcal{G}$ of a GMRF is a tree, belief propagation (BP) efficiently calculates the correct marginals (Malioutov et al., 2006; Pearl, 1988). For graphs with cycles (also called loopy graphs), the method of loopy belief propagation (LBP) can be used to efficiently *approximate* the marginals. However, it was shown in Weiss and Freeman (2001) that while LBP gives correct means for the marginals, the estimates of the covariance matrices it provides are generally incorrect.

In this article, we present a novel approach to the calculation of the marginals of $\mathbf{X}$ or $\mathbf{X}|\mathbf{Y}$, which we term *method of path-sums*. This approach is a generalization and completion of the walk-sum formulation developed in Malioutov et al. (2006). The method of path-sums is based on results established by Giscard et al. (2012) concerning the algebraic structure of walk sets, which permit the systematic resummation of infinite families of walks in any walk-sum. These resummations transform a walk-sum into a branched continued fraction comprising only a *finite* number of terms. Furthermore, these terms have an elementary interpretation as simple paths and simple cycles on $\mathcal{G}$. A simple path is a walk (i.e. a trajectory on $\mathcal{G}$) whose vertices are all distinct. A simple cycle is a walk whose endpoints

1. For a GMRF, the pairwise Markov property, the local Markov property and the global Markov property are equivalent. This is proven by using Proposition 3.8 of Lauritzen (1996), in conjunction with the Hammersley-Clifford Theorem (Theorem 3.9, Lauritzen, 1996).
2. Algorithms that compute *some* entries (either diagonal entries or certain off-diagonal entries) of a symmetric sparse matrix with complexity less than $O(n^3)$ do exist (see Eastwood and Wan, 2013; Li et al., 2008; Lin et al., 2011; Tang and Saad, 2012). The path-sum representation achieves this as well, computing the covariance of a pair of variables with complexity $O(n)$ whenever $\mathcal{G}$ is a tree: see §5 and Giscard et al. (2013).

are identical and intermediate vertices (from the second to the penultimate) are all distinct and different from the endpoints. An important consequence of these observations is that if J is positive definite[3], the path-sum formulation of J^{-1} is convergent.

Consequently, one does not need the walk-summability of a model, which was devised by Malioutov et al. (2006) to guarantee the convergence of the infinite walk-sum representation of the covariance matrix $\Sigma = J^{-1}$. Let $R := I - J$ be the partial correlations matrix and $|R|$ be the entrywise absolute value of this matrix, i.e. $(|R|)_{ij} = |R_{ij}|$. The authors showed that a GMRF is walk-summable if and only if the spectral radius $\rho(|R|)$ is strictly less than 1, which implies $J \succ 0$. However, the converse does not hold: that is, there are positive definite information matrices which are not walk-summable. In contrast to the walk-summability criterion, our formulation only requires positive definiteness of J. Most importantly, the path-sum formulation gives the *correct* marginals when the graph $\mathcal{G}$ is loopy, and is equivalent to BP when $\mathcal{G}$ is a tree.

The rest of this article is organized as follows. In §2, we introduce the context and arguments underlying the path-sum formulation. We present the path-sum result in §2.2 and show its validity for all positive definite matrices, irrespective of the walk-summability criterion. In the following section, §3, we relate the path-sum representation of a covariance matrix to existing approaches. In §4 we prove that the path-sum representation can be applied to arbitrary partitions of the information matrix J. We give an example demonstrating this claim. Finally in §5 we briefly discuss the computational cost of our approach as well as future prospects. The proof of the path-sum result is deferred to Appendix A.

2. Path-Sum Representation

2.1 Context

The *"most basic result of algebraic graph theory"* (as described in Flajolet and Sedgewick, 2009) states that the powers of the adjacency matrix $A_{\mathcal{G}}$ of a graph $\mathcal{G}$ generate all the walks on this graph (Biggs, 1993). This result extends to weighted graphs if $A_{\mathcal{G}}$ is replaced by a weighted adjacency matrix R, with R_{ij} the weight of the edge from vertex j to vertex i on $\mathcal{G}$. Then $(R^\ell)_{ij}$ is the sum of the weights of all the walks of length ℓ from j to i (Flajolet and Sedgewick, 2009). The weight of a walk is simply the ordered product of the weight of the edges it traverses. (Note that the indices of a matrix are written right-to-left, but correspond to an edge written left-to-right. This is due to unfortunate conventions). We index the entries of R by the labels of the vertices in $\mathcal{G}$. We write these labels with roman letters, Greek letters, or numbers, as convenient. Now – assuming that the Taylor series converges – we have $(I - R)^{-1} = \sum_\ell R^\ell$. It follows that $(I - R)^{-1}_{ij}$ can be interpreted as the sum of the weights of all the walks from j to i. This directly implies the walk-sum interpretation advocated by Malioutov et al. (2006) for $J^{-1} = (I - R)^{-1}$, with $R = I - J$ and $\mathcal{G}$ the graphical model of the GMRF constructed by using Definition 1. Further, it follows that the calculation of J^{-1} is susceptible to a particular resummation technique from graph theory based on the structure of sets of walks, called the method of path-sums.

In its most general form, the method of path-sums stems from a fundamental algebraic property of the set of all walks on any weighted graph: namely, that any walk factorizes

3. $J \succ 0$ is equivalent to J being non-singular for a GMRF.

uniquely into products of prime walks, which are the simple paths and simple cycles of $\mathcal{G}$ (see §1 for the definitions). The path-sum representation of the series of all walks on the graph $\mathcal{G}$ is thus the representation of this series that only involves the prime walks.[4] Since a finite graph sustains only finitely many primes, the walk series (which is typically infinite) thus has an exact representation involving only finitely many terms. An important consequence of this observation is that the path-sum expression of J^{-1} is *convergent* as long as $J \succ 0$.

For a full exposition of the algebraic structure of walk sets at the origin of path-sums and its applications in linear algebra, the interested reader can refer to Giscard et al. (2012) and Giscard et al. (2013). In §2.2 we give the *explicit* and universal path-sum formulation for J^{-1}. This expression takes the form of a branched continued fraction of finite depth and breadth.

2.2 Path-Sum Formulation of the Covariance Matrix

Let $\mathcal{G} \setminus \{\alpha, \beta \dots\}$ denote the subgraph of $\mathcal{G}$ obtained by deleting from $\mathcal{G}$ the vertices $\{\alpha, \beta \dots\} \subset \mathcal{V}$ and the edges incident to them. For simplicity, we write $J^{-1}_{\alpha\beta}$ for $(J^{-1})_{\alpha\beta}$. The path-sum expression for $\Sigma = J^{-1}$ is presented in Theorem 2 below. We defer its proof to Appendix A.

Theorem 2 *Let $J \succ 0$ be an information matrix. Let $\Pi_{\mathcal{G};\,\alpha\omega}$ and $\Gamma_{\mathcal{G};\,\alpha}$ be the sets of simple paths from α to ω on $\mathcal{G}$ and the set of simple cycles from α to itself on $\mathcal{G}$, respectively. If $\mathcal{G}$ has finitely many vertices and edges, these two sets are finite.*

Then each entry of the covariance matrix $\Sigma = J^{-1}$ admits an expression involving only weighted prime walks, called a path-sum representation. It is explicitly given by

$$J^{-1}_{\omega\alpha} = \sum_{p \in \Pi_{\mathcal{G};\,\alpha\omega}} (-1)^{\ell(p)} \prod_{j=1}^{\ell(p)+1} \left\{ \left(J_{\mathcal{G}\setminus\{\alpha,\nu_2,\dots,\nu_{j-1}\}}\right)^{-1}_{\nu_j\nu_j} J_{\nu_{j+1}\nu_j} \right\} J^{-1}_{\alpha\alpha}\,, \tag{2}$$

$$J^{-1}_{\alpha\alpha} = \left(\sum_{\gamma \in \Gamma_{\mathcal{G};\,\alpha\alpha}} (-1)^{\ell(\gamma)+1} J_{\mu_1\mu_{\ell(\gamma)}} \prod_{j=2}^{\ell(\gamma)} \left\{ \left(J_{\mathcal{G}\setminus\{\alpha,\mu_2,\dots,\mu_{j-1}\}}\right)^{-1}_{\mu_j\mu_j} J_{\mu_j\mu_{j-1}} \right\} \right)^{-1}, \tag{3}$$

where the products are right-to-left (i.e. $\prod_{i=1}^{m} a_i = a_m \cdots a_1$), $p = (\nu_1, \nu_2, \dots, \nu_{\ell(p)+1})$ is a simple path of length $\ell(p)$ with $\alpha \equiv \nu_1$ and $\omega \equiv \nu_{\ell(p)+1}$ for convenience; and $\gamma = (\mu_1, \mu_2, \dots, \mu_{\ell_\gamma}, \mu_1)$ is a simple cycle of length $\ell(\gamma)$ from $\alpha \equiv \mu_1$ to itself.

Note that $J^{-1}_{\alpha\alpha}$ is obtained recursively through Eq. (3). Indeed it is expressed in terms of entries of inverses of submatrices of J, such as $(J_{\mathcal{G}\setminus\{\alpha,\mu_2,\dots,\mu_{j-1}\}})^{-1}_{\mu_j\mu_j}$, which is in turn obtained through Eq. (3) but on the subgraph $\mathcal{G}\setminus\{\alpha, \dots, \mu_{j-1}\}$ of $\mathcal{G}$. The recursion stops when vertex μ_j has no neighbor on this subgraph, in which case $(J_{\mathcal{G}\setminus\{\alpha,\mu_2,\dots,\mu_{j-1}\}})^{-1}_{\mu_j\mu_j} = 1/J_{\mu_j\mu_j}$ (note that $J_{\mu_j\mu_j} \neq 0$ since $J \succ 0$). The entry $J^{-1}_{\alpha\alpha}$ is therefore expressed as a branched continued fraction which terminates at a finite depth, and $J^{-1}_{\alpha\omega}$ is a finite sum of such continued fractions.

4. Since path-sums are the prime representations of walk series, they are the graph-theoretic analog of Euler product formulas for the Riemann zeta function and other totally multiplicative functions in number theory.

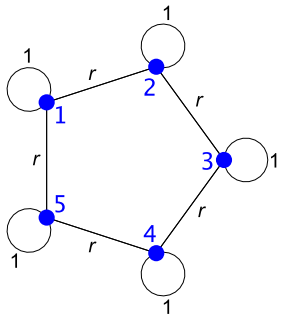


Figure 1: The circle graph on 5 vertices $\mathcal{C}_5$ *associated with* J in Example 1. The edge-weights are indicated next to the edges and the vertices are labeled 1 to 5.

Remark 3 Theorem 2 is valid even when $\mathcal{G}$ is loopy and/or J is not walk-summable. In particular there is no restriction on the spectrum of J as long as $J \succ 0$. An example showing this is given below.

Example 1 (An illustrative example) To illustrate Theorem 2, we consider an example taken from Malioutov et al. (2006), where J has the structure of a circle graph on 5 vertices, denoted $\mathcal{C}_5$, see Figure 1. The information matrix is

$$J = \begin{pmatrix} 1 & r & 0 & 0 & r \\ r & 1 & r & 0 & 0 \\ 0 & r & 1 & r & 0 \\ 0 & 0 & r & 1 & r \\ r & 0 & 0 & r & 1 \end{pmatrix}.$$

Then J is positive definite for $\frac{2}{1-\sqrt{5}} \leq r \leq \frac{2}{1+\sqrt{5}}$ (approximately $-1.618 \leq r \leq 0.618$), and walk-summable if and only if $-1/2 \leq r \leq 1/2$. This example thus provides a test case for both the walk-summable and non-walk-summable situations, depending on the value of r. Here we obtain J^{-1} correctly for all values of r such that J is not singular.

By the symmetry of $\mathcal{C}_5$, all the diagonal entries of J^{-1} are identical, and we choose to calculate J^{-1}_{11} without loss of generality. There are five simple cycles from vertex 1 to itself on $\mathcal{C}_5$: i) the self-loop $1 \to 1$; ii) the two backtracks $1 \to 2 \to 1$ and $1 \to 5 \to 1$; and iii) the two pentagons $1 \to 2 \to 3 \to 4 \to 5 \to 1$ and $1 \to 5 \to 4 \to 3 \to 2 \to 1$. By symmetry, the

two backtracks and the two pentagons have the same weights. Consequently Eq. (3) gives

$$(J^{-1})_{11} = \Bigg(\underbrace{1}_{\text{Self-loop}} - \underbrace{2\,r^2\left(J_{\mathcal{C}_5\setminus\{1\}}\right)^{-1}_{22}}_{\text{Backtracks}} + \underbrace{2\,r^5\left(J_{\mathcal{C}_5\setminus\{1,2,3,4\}}\right)^{-1}_{55}\left(J_{\mathcal{C}_5\setminus\{1,2,3\}}\right)^{-1}_{44}\left(J_{\mathcal{C}_5\setminus\{1,2\}}\right)^{-1}_{33}\left(J_{\mathcal{C}_5\setminus\{1\}}\right)^{-1}_{22}}_{\text{Pentagons}} \Bigg)^{-1}.$$

The required entries $(J_{\mathcal{C}_5\setminus\{1\}})^{-1}_{22}, \cdots, (J_{\mathcal{C}_5\setminus\{1,2,3,4\}})^{-1}_{55}$ remain to be calculated. To this end, we use again Eq. (3) of Theorem 2. For example, consider calculating $(J_{\mathcal{C}_5\setminus\{1\}})^{-1}_{22}$. Since the graph associated with $J_{\mathcal{C}_5\setminus\{1\}}$ is $\mathcal{C}_5$ with vertex 1 removed, the only simple cycles from vertex 2 to itself on $\mathcal{C}_5\setminus\{1\}$ are the self-loop $2\to 2$ and the backtrack $2\to 3\to 2$. We thus find

$$\left(J_{\mathcal{C}_5\setminus\{1\}}\right)^{-1}_{22} = \Bigg(\underbrace{1}_{\text{Self-loop}} - \underbrace{r^2\left(J_{\mathcal{C}_5\setminus\{1,2\}}\right)^{-1}_{33}}_{\text{Backtrack}} \Bigg)^{-1},$$

Similarly we obtain $(J_{\mathcal{C}_5\setminus\{1,2\}})^{-1}_{33} = \left(1-r^2(J_{\mathcal{C}_5\setminus\{1,2,3\}})^{-1}_{44}\right)^{-1}$, $(J_{\mathcal{C}_5\setminus\{1,2,3\}})^{-1}_{44} = \left(1-r^2(J_{\mathcal{C}_5\setminus\{1,2,3,4\}})^{-1}_{55}\right)^{-1}$ and finally $(J_{\mathcal{C}_5\setminus\{1,2,3,4\}})^{-1}_{55} = 1$. Combining these equations gives

$$\begin{aligned}
(J_{\mathcal{C}_5\setminus\{1,2,3,4\}})^{-1}_{55} = 1 &\Rightarrow \left(J_{\mathcal{C}_5\setminus\{1,2,3\}}\right)^{-1}_{44} = \frac{1}{1-r^2} \\
&\Rightarrow \left(J_{\mathcal{C}_5\setminus\{1,2\}}\right)^{-1}_{33} = \frac{1-r^2}{1-2r^2} \\
&\Rightarrow \left(J_{\mathcal{C}_5\setminus\{1\}}\right)^{-1}_{22} = \frac{1-2r^2}{1-3r^2+r^4}, \\
&\Rightarrow (J^{-1})_{11} = \frac{1-3r^2+r^4}{1-5r^2+5r^4+2r^5}.
\end{aligned}$$

As noted, the last equation gives the value of every diagonal entry of J^{-1}. We now consider the off-diagonal entries. We note again that, by the symmetry of $\mathcal{C}_5$, there are only two different entries, and we choose to calculate $(J^{-1})_{21}$ and $(J^{-1})_{31}$ without loss of generality. Following Theorem 2, these two entries are given by a sum over simple paths from vertex 1 to vertex 2, and vertex 1 to vertex 3, respectively. In each case there are only two simple paths: for example from 1 to 2, we have $1\to 2$ and $1\to 5\to 4\to 3\to 2$. Then Eq. (2)

yields

$$\left(J^{-1}\right)_{21} = \underbrace{-\,r\left(J_{\mathcal{C}_5\setminus\{1\}}\right)^{-1}_{22}\left(J^{-1}\right)_{11}}_{\text{Simple path}} + \underbrace{r^4\left(J_{\mathcal{C}_5\setminus\{1,5,4,3\}}\right)^{-1}_{22}\left(J_{\mathcal{C}_5\setminus\{1,5,4\}}\right)^{-1}_{33}\left(J_{\mathcal{C}_5\setminus\{1,5\}}\right)^{-1}_{44}\left(J_{\mathcal{C}_5\setminus\{1\}}\right)^{-1}_{55}\left(J^{-1}\right)_{11}}_{\text{Simple path}}.$$

By the symmetry of $\mathcal{C}_5$, we have $(J_{\mathcal{C}_5\setminus\{1,5,4,3\}})^{-1}_{22} = (J_{\mathcal{C}_5\setminus\{1,2,3,4\}})^{-1}_{55}$, $(J_{\mathcal{C}_5\setminus\{1,5,4\}})^{-1}_{33} = (J_{\mathcal{C}_5\setminus\{1,2,3\}})^{-1}_{44}$, $(J_{\mathcal{C}_5\setminus\{1,5\}})^{-1}_{44} = (J_{\mathcal{C}_5\setminus\{1,2\}})^{-1}_{33}$ and $(J_{\mathcal{C}_5\setminus\{1\}})^{-1}_{55} = (J_{\mathcal{C}_5\setminus\{1\}})^{-1}_{22}$. The previously obtained results then immediately give

$$(J^{-1})_{21} = \frac{r^4+2r^3-r}{1-5r^2+5r^4+2r^5},$$

and similarly

$$(J^{-1})_{31} = \frac{r^2-r^3-r^4}{1-5r^2+5r^4+2r^5}.$$

Piecing these results together, we finally obtain

$$J^{-1} = \frac{r^2}{2r^3+3r^2-r-1}\begin{pmatrix} \frac{(r-1)r-1}{r^2} & 1+r^{-1} & -1 & -1 & 1+r^{-1} \\ 1+r^{-1} & \frac{(r-1)r-1}{r^2} & 1+r^{-1} & -1 & -1 \\ -1 & 1+r^{-1} & \frac{(r-1)r-1}{r^2} & 1+r^{-1} & -1 \\ -1 & -1 & 1+r^{-1} & \frac{(r-1)r-1}{r^2} & 1+r^{-1} \\ 1+r^{-1} & -1 & -1 & 1+r^{-1} & \frac{(r-1)r-1}{r^2} \end{pmatrix}.$$

We now verify that this expression is correct in both the walk-summable and non-walk-summable situations:

(a) Walk-summable and $J \succ 0$: set $r = 0.3$. Then to 5 decimal places we have $(J^{-1})_{11} = 1.23975$, $(J^{-1})_{21} = -0.39959$ and $(J^{-1})_{31} = 0.09221$, so that

$$J^{-1} = \begin{pmatrix} 1.23975 & -0.39959 & 0.09221 & 0.09221 & -0.39959 \\ -0.39959 & 1.23975 & -0.39959 & 0.09221 & 0.09221 \\ 0.09221 & -0.39959 & 1.23975 & -0.39959 & 0.09221 \\ 0.09221 & 0.09221 & -0.39959 & 1.23975 & -0.39959 \\ -0.39959 & 0.09221 & 0.09221 & -0.39959 & 1.23975 \end{pmatrix}.$$

One can verify that this result obtained by using the path-sum formulation coincides with the one obtained by direct inversion of J.

(b) Non walk-summable and $J \succ 0$: set $r = 0.6$. Then to 5 decimal places, we have $(J^{-1})_{11} = 14.09091$, $(J^{-1})_{21} = -10.90909$ and $(J^{-1})_{31} = 4.09091$, and so

$$J^{-1} = \begin{pmatrix} 14.09091 & -10.90909 & 4.09091 & 4.09091 & -10.90909 \\ -10.90909 & 14.09091 & -10.90909 & 4.09091 & 4.09091 \\ 4.09091 & -10.90909 & 14.09091 & -10.90909 & 4.09091 \\ 4.09091 & 4.09091 & -10.90909 & 14.09091 & -10.90909 \\ -0.39959 & 4.09091 & 4.09091 & -0.39959 & 14.09091 \end{pmatrix}.$$

This result is again easily verified through direct inversion of J.

Remark 4 First, one could have calculated everything with numerical values from the start, as opposed to evaluating the analytic expression of J^{-1} for a specific value of r as we did here. This gives the same results, as expected. Second, the analytical formula for J^{-1} remains valid even when J is not positive definite and only fails for those values of r such that J is singular. The main *mathematical* role of the condition $J \succ 0$ in the path-sum representation is to guarantee that J is not singular.[5]

In the following section, we discuss the relations between Theorem 2 and two existing approaches.

3. Relation to Existing Approaches

Malioutov et al. (2006) provided a walk-sum derivation for Gaussian belief propagation on trees. Jones and West (2005) presented an expression for the entries of J^{-1} as a sum of simple paths. In this section, we show that these results are corollaries of Theorem 2 arising as special cases.

3.1 Path-Sums on Trees

The recursive structure of the path-sum representation of J^{-1} is especially simple on trees, for which we recover the Gaussian belief propagation results of Malioutov et al. (2006).

Let $J \succ 0$ be an information matrix associated with a tree model $\mathcal{T}$. For any vertex α of $\mathcal{T}$, let $\mathcal{N}(\alpha)$ be the set of neighbors of α on $\mathcal{T}$. Observe that since $\mathcal{T}$ is a tree, the only simple cycles from α to itself are the self-loop $\alpha \to \alpha$ with weight $J_{\alpha\alpha}$, and the backtracks to the neighbors of α on $\mathcal{T}$, e.g. $\alpha \to \beta \to \alpha$, $\beta \in \mathcal{N}(\alpha)$. Then Eq. (3) of Theorem 2 gives

$$\Sigma_{\alpha\alpha} = J^{-1}_{\alpha\alpha} = \left(J_{\alpha\alpha} + \sum_{\beta \in \mathcal{N}(\alpha)} -J_{\alpha\beta} (J_{\mathcal{T}\setminus\{\alpha\}})^{-1}_{\beta\beta} J_{\beta\alpha} \right)^{-1}. \tag{4a}$$

The quantity $(J_{\mathcal{T}\setminus\{\alpha\}})^{-1}_{\beta\beta}$ satisfies a similar relation on the subtree $\mathcal{T}\setminus\{\alpha\}$,

$$(J_{\mathcal{T}\setminus\{\alpha\}})^{-1}_{\beta\beta} = \left(J_{\beta\beta} + \sum_{\delta \in \mathcal{N}(\beta)\setminus\alpha} -J_{\beta\delta} (J_{\mathcal{T}\setminus\{\alpha,\beta\}})^{-1}_{\delta\delta} J_{\delta\beta} \right)^{-1}.$$

5. A path-sum result for singular matrices also exists, but it necessitates additional mathematical machinery. In this case the path-sum formulation yields a pseudo-inverse for the singular matrix, see Giscard et al. (2013).

Let $\mathcal{C}_{\mathcal{T}\setminus\{\beta\};\,\delta}$ be the connected component of $\mathcal{T}\setminus\{\beta\}$ that contains the vertex $\delta \in \mathcal{N}(\alpha)$. Similarly define $\mathcal{C}_{\mathcal{T}\setminus\{\alpha,\beta\};\,\delta}$ to be the connected component of $\mathcal{T}\setminus\{\alpha,\beta\}$ that contains the vertex δ. Since $\mathcal{T}$ is a tree, these components are identical: $\mathcal{C}_{\mathcal{T}\setminus\{\beta\};\,\delta} = \mathcal{C}_{\mathcal{T}\setminus\{\alpha,\beta\};\delta}$, and therefore $(J_{\mathcal{T}\setminus\{\beta\}})^{-1}_{\delta\delta} = (J_{\mathcal{T}\setminus\{\alpha,\beta\}})^{-1}_{\delta\delta}$. Thus, we have

$$(J_{\mathcal{T}\setminus\{\alpha\}})^{-1}_{\beta\beta} = \left(J_{\beta\beta} + \sum_{\delta \in \mathcal{N}(\beta)\setminus\alpha} -J_{\beta\delta} (J_{\mathcal{T}\setminus\{\beta\}})^{-1}_{\delta\delta} J_{\delta\beta} \right)^{-1}. \tag{4b}$$

In order to show that Eqs. (4a, 4b) are the Gaussian belief propagation results, we introduce some notation from Malioutov et al. (2006). Let $\hat{J}_\alpha := 1/\Sigma_{\alpha\alpha}$ and $\hat{J}^{-1}_{\beta\setminus\alpha} := (J_{\mathcal{T}\setminus\{\alpha\}})^{-1}_{\beta\beta}$. With these notations, Eqs. (4a, 4b) become

$$\hat{J}_\alpha = J_{\alpha\alpha} + \sum_{\beta \in \mathcal{N}(\alpha)} \Delta J_{\beta\to\alpha}, \quad \text{and} \quad \hat{J}_{\beta\setminus\alpha} = J_{\beta\beta} + \sum_{\delta \in \mathcal{N}(\beta)\setminus\alpha} \Delta J_{\delta\to\beta},$$

with

$$\Delta J_{\delta\to\beta} = -J_{\beta\delta} \hat{J}^{-1}_{\delta\setminus\beta} J_{\delta\beta}.$$

These are the Gaussian belief propagation equations (Eqs. 7, 8 and 9 in Malioutov et al., 2006), which immediately imply Propositions 16 and 17 in Malioutov et al. (2006) with the further definitions $r_{\beta\alpha} := R_{\beta\alpha} = -J_{\beta\alpha}$, $\gamma_{\beta\setminus\alpha} := (J_{\mathcal{T}\setminus\{\alpha\}})^{-1}_{\beta\beta}$. Note that $r_{\alpha\beta} = r_{\beta\alpha}$ since J is symmetric.

3.2 An Approach using Determinants

A determinant-based approach to the calculation of the covariance matrix $\Sigma = J^{-1}$ was demonstrated by Jones and West (2005). Here we show that this result follows from Eq. (2) by using the adjugate formula for the matrix inverse. We then point out the fundamental limitation of determinant-based approaches. We overcome this limitation using the path-sum formulation in §4.

We recall the adjugate formula for matrix inverses:

Proposition 5 (Adjugate formula, Strang, 2005) *Let $M \in \mathbb{C}^{n\times n}$ be a $n \times n$ non-singular complex matrix. Then*

$$\left(M^{-1}\right)_{ij} = (-1)^{i+j} \frac{\det M^i_j}{\det M}, \text{ and in particular } \left(M^{-1}\right)_{ii} = \frac{\det M^i}{\det M},$$

where M^i_j is the matrix M with its ith column and jth row removed and $M^i = M^i_i$.

To obtain this result we start with Eq. (2), which gives here

$$J^{-1}_{\omega\alpha} = \sum_{p \in \Pi_{\mathcal{G};\,\alpha\omega}} (-1)^{\ell(p)} (J_{\mathcal{G}\setminus\{\alpha,\,\nu_2,\ldots,\nu_{\ell(p)}\}})^{-1}_{\omega\omega} J_{\omega\nu_{\ell(p)}} \cdots J_{\nu_3\nu_2} (J_{\mathcal{G}\setminus\{\alpha\}})^{-1}_{\nu_2\nu_2} J_{\nu_2\alpha} (J^{-1})_{\alpha\alpha},$$

with $p = (\alpha, \nu_2, \ldots \nu_{\ell(p)}\omega)$ is a simple path of length $\ell(p)$. Since all the entries of J commute, we can reorganize the weights in the above expression to get

$$J^{-1}_{\omega\alpha} = \sum_{p \in \Pi_{\mathcal{G};\,\alpha\omega}} (-1)^{\ell(p)} \phi[p] \times (J_{\mathcal{G}\setminus\{\alpha,\nu_2,\ldots,\nu_{\ell(p)}\}})^{-1}_{\omega\omega} \times \ldots \times (J_{\mathcal{G}\setminus\{\alpha\}})^{-1}_{\nu_2\nu_2} \times (J^{-1})_{\alpha\alpha},$$

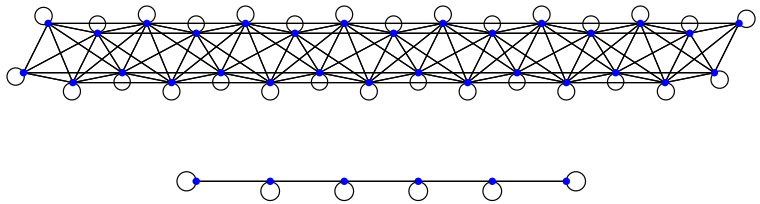


Figure 2: top: the graphical model associated with a 5-banded 30×30 positive definite information matrix. The weights of the edges are the entries of J. Bottom: the graphical model associated with a partition of J into 5×5 blocks. The edge weights are the blocks constituting J.

where $\phi[p] = J_{\omega\nu_{\ell(p)}} \dots J_{\nu_3\nu_2} J_{\nu_2\alpha}$ is the weight of the simple path p. Using the adjugate formula in Proposition 5, we have

$$J^{-1}_{\omega\alpha} = \sum_{p \in \Pi_{\mathcal{G};\,\alpha\omega}} (-1)^{\ell(p)} \phi[p] \times \frac{\det J^{\alpha,\nu_2,\dots,\nu_{\ell(p)},\omega}}{\det J^{\alpha,\nu_2,\dots,\nu_{\ell(p)}}} \times \dots \frac{\det J^{\alpha,\nu_2}}{\det J^{\alpha}} \times \frac{\det J^{\alpha}}{\det J},$$

since $J_{\mathcal{G}\setminus\{\alpha,\nu_2,\dots,\nu_j\}} = J^{\alpha,\nu_2,\dots,\nu_j}$. This simplifies to

$$J^{-1}_{\omega\alpha} = \sum_{p \in \Pi_{\mathcal{G};\,\alpha\omega}} (-1)^{\ell(p)} \phi[p] \frac{\det J^{\alpha,\nu_2,\dots,\omega}}{\det J}. \tag{5}$$

The last equation is given as Theorem 1 in Jones and West (2005). Note that we only used Eq. (2) in Theorem 2 to obtain this result. Jones and West (2005) did not give an expression equivalent to Eq. (3) in Theorem 2, which provides an explicit formula for the necessary determinants in terms of simple cycles on the graph associated with J.

The requirement that the entries of J be commutative to obtain Eq. (5) from Eq. (2) may seem trivial, but in fact it excludes the very important case of *block matrices*. In the next section, we demonstrate the use of Theorem 2 in the case where J is a block matrix.

4. Path-Sums for Arbitrary Partitions of J

An information matrix J may have a sparsity structure that is best exploited by partitioning J into blocks. For example, consider an $n \times n$ information matrix J which is banded: i.e. $J_{ij} = 0$ if and only if $|i - j| > b$ for some $b < n$. We say that J is b-banded. Then J is simply block-tridiagonal when partitioned $b \times b$ blocks. Consequently, the graph $\mathcal{G}$ associated with this partition of J is simpler than the graph associated with the full matrix: see for example Figure 2. It is therefore desirable to develop a method capable of exploiting these simplifications.

A fundamental impediment to determinant-based approaches to J^{-1} is that the notion of determinant itself does not extend to matrices with non-commutative blocks (Silvester, 2000). In contrast, the path-sum formulation for J^{-1} does not require the commutativity of the entries of J. For this reason, it continues to hold even when these entries do not commute, see (Giscard et al., 2012) and (Giscard et al., 2013).

Let J be an $n \times n$ information matrix and $I_1, \dots, I_B$ for some $1 \leq B \leq n$ be B disjoint subsets of $\{1, \dots, n\}$. Then we write $J_{I_i I_j}$ for the minor (i.e. block) of J that corresponds to the rows indexed by I_i and the columns indexed by I_j. We form a new graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ associated with this partition of J, such that $J_{I_i I_j} \neq \mathbf{0} \iff (j,i) \in \mathcal{E}'$, where $\mathbf{0}$ is a zero matrix. The block $J_{I_i I_j}$ is now the weight associated with the edge (j,i). With these conventions, Theorem 2 extends to block matrices without modification.

Remark 6 For a GMRF $\mathbf{X}$, the partition of J given above is equivalent to a partition of the set of random variables $\mathbf{X}$ into $B > 1$ disjoint subsets $\mathbf{X}_1, \dots, \mathbf{X}_B$. Let $\mathbf{X}' = (\mathbf{X}_1, \dots, \mathbf{X}_B)$ be a GMRF with respect to a new graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ with information matrix J'. Note that each $\mathbf{X}_i$ is now a random vector instead of a random variable. Following Definition 1:

$$\mathbf{X}_i \perp\!\!\!\perp \mathbf{X}_j | \mathbf{X}'_{\setminus ij} \iff J'_{ij} = \mathbf{0} \iff (j,i) \notin \mathcal{E}', \tag{6}$$

with $\mathbf{X}'_{\setminus ij}$ the set of variables with $\mathbf{X}_i$ and $\mathbf{X}_j$ removed from $\mathbf{X}'$. Note that $\mathbf{X}_i \perp\!\!\!\perp \mathbf{X}_j | \mathbf{X}'_{\setminus ij}$ implies the global Markov property (GMP), which, for GMRFs, is equivalent to the pairwise Markov property (PMP) (Lauritzen, 1996). However for a general distribution, the PMP does not imply the GMP (Lauritzen, 1996). Hence in this case, the partition of $\mathbf{X}$ as per Eq. (6) may not correspond to the partition of J.

Below we give a detailed synthetic example demonstrating the path-sum formulation of the covariance matrix $\Sigma = J^{-1}$ using a partition of J.

Example 2 (Non walk-summable block matrix on a loopy graph) Consider the following positive definite information matrix J of a thin membrane model

$$J = \tag{7}$$
$$\begin{pmatrix}
a+5b & -b & 0 & -b & -b & 0 & -b & -b & 0 \\
-b & a+5b & -b & -b & 0 & -b & -b & 0 & -b \\
0 & -b & a+5b & 0 & -b & -b & 0 & -b & -b \\
-b & -b & 0 & a+5b & -b & 0 & -b & -b & 0 \\
-b & 0 & -b & -b & a+5b & -b & -b & 0 & -b \\
0 & -b & -b & 0 & -b & a+5b & 0 & -b & -b \\
-b & -b & 0 & -b & -b & 0 & a+5b & -b & 0 \\
-b & 0 & -b & -b & 0 & -b & -b & a+5b & -b \\
0 & -b & -b & 0 & -b & -b & 0 & -b & a+5b
\end{pmatrix},$$

where $a, b > 0$. Recall that walk-summability is equivalent to the condition $\rho(|R|) < 1$, where $|R|$ is the entry-wise absolute value of $R := I - J$ (Malioutov et al., 2006). Here we have $\rho(|R|) = |a + 5b - 1| + \sqrt{19}\, b + b$, and thus—depending on the values of a and b—walk-summability does not always hold. For example, when $a = b = 1$ we have $\rho(|R|) \simeq 10.36$, $\rho(J) \simeq 9.36$ and $\rho(R) \simeq 8.36$ and J is not walk-summable.

The graph $\mathcal{G}$ associated to J is shown on the left of Figure 3. Instead of working on this complicated graph, we may partition the matrix into 3×3 blocks as follows

$$J = \begin{pmatrix} L & E & E \\ E & L & E \\ E & E & L \end{pmatrix}, \tag{8}$$

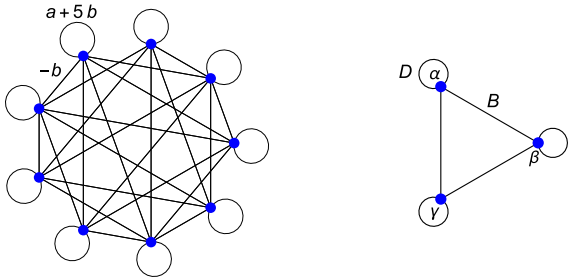


Figure 3: left: graphical model associated with the thin membrane model matrix J of Eq. (7). Every edge has weight $-b$ and every self-loop has weight $a+5b$. Right: triangular graphical model $\mathcal{K}_3$ associated with the partition of J given in Eq. (8). Edges and self-loops have matrix weights E and L, respectively. The vertices are labeled α, β and γ for convenience.

with

$$E := \begin{pmatrix} -b & -b & 0 \\ -b & 0 & -b \\ 0 & -b & -b \end{pmatrix}, \quad L := \begin{pmatrix} a+5b & -b & 0 \\ -b & a+5b & -b \\ 0 & -b & a+5b \end{pmatrix}.$$

The graph $\mathcal{G}'$ associated with this partition of J is a triangle, denoted $\mathcal{K}_3$ and shown on the right of Figure 3. Accordingly, the edge-weights are now the matrices E and L for edges and self-loops, respectively. Note that these weights do not commute: $[E, L] = EL - LE \neq 0$.

The only simple cycles from any vertex (say, α) to itself on this graph are the self-loop $\alpha \to \alpha$, the two backtracks $\alpha \to \beta \to \alpha$ and $\alpha \to \gamma \to \alpha$ and the two triangles $\alpha \to \beta \to \gamma \to \alpha$ and $\alpha \to \gamma \to \beta \to \alpha$. Theorem 2 therefore yields

$$J^{-1}_{\alpha\alpha} = \Big(\underbrace{L}_{\text{Self loop}} - \underbrace{E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\beta\beta} \,.\, E}_{\text{Backtrack}} - \underbrace{E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\gamma\gamma} \,.\, E}_{\text{Backtrack}} \tag{9a}$$

$$+ \underbrace{E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha,\beta\}})^{-1}_{\gamma\gamma} \,.\, E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\beta\beta} \,.\, E}_{\text{Triangle}} + \underbrace{E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha,\gamma\}})^{-1}_{\beta\beta} \,.\, E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\gamma\gamma} \,.\, E}_{\text{Triangle}} \Big)^{-1},$$

where, by Eq. (3),

$$(J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\beta\beta} = \Big(\underbrace{L}_{\text{Self loop}} - \underbrace{E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha,\beta\}})^{-1}_{\gamma\gamma} \,.\, E}_{\text{Backtrack}} \Big)^{-1} \text{ and } (J_{\mathcal{K}_3\setminus\{\alpha,\beta\}})^{-1}_{\gamma\gamma} = \underbrace{L^{-1}}_{\text{Self loop}}. \tag{9b}$$

Since $(J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\gamma\gamma} = (J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\beta\beta}$ and $(J_{\mathcal{K}_3\setminus\{\alpha,\gamma\}})^{-1}_{\beta\beta} = (J_{\mathcal{K}_3\setminus\{\alpha,\beta\}})^{-1}_{\gamma\gamma}$, Eqs. (9a, 9b) completely determine $J^{-1}_{\alpha\alpha}$. Thus, we have

$$J^{-1}_{\alpha\alpha} = \frac{b}{a^2+8ab-3b^2} \times \begin{pmatrix} \frac{a}{b} + \frac{a}{a+3b} + \frac{5a}{3(a+6b)} + \frac{1}{3} & 1 & \frac{3b}{a+3b} \\ 1 & \frac{a}{b} + \frac{12a}{5(a+5b)} + \frac{3}{5} & 1 \\ \frac{3b}{a+3b} & 1 & \frac{a}{b} + \frac{a}{a+3b} + \frac{5a}{3(a+6b)} + \frac{1}{3} \end{pmatrix}.$$

By symmetry of J, this gives also $J^{-1}_{\beta\beta}$ and $J^{-1}_{\gamma\gamma}$. The off-diagonal blocks, which are all identical because of the symmetry of the matrix, now follow easily. For example, we have

$$(J^{-1})_{\beta\alpha} = \underbrace{(J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\beta\beta} \,.\, E \,.\, (J^{-1})_{\alpha\alpha}}_{\text{Path}} - \underbrace{E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha,\gamma\}})^{-1}_{\beta\beta} \,.\, E \,.\, (J_{\mathcal{K}_3\setminus\{\alpha\}})^{-1}_{\gamma\gamma} \,.\, E \,.\, (J^{-1})_{\alpha\alpha}}_{\text{Path}},$$

$$= \frac{b}{a^2+8ab-3b^2} \begin{pmatrix} \frac{a}{a+3b} - \frac{5a}{6(a+6b)} + \frac{5}{6} & 1 & \frac{3b}{a+3b} \\ 1 & \frac{6b}{a+5b} & 1 \\ \frac{3b}{a+3b} & 1 & \frac{a}{a+3b} - \frac{5a}{6(a+6b)} + \frac{5}{6} \end{pmatrix}.$$

As long as $a, b > 0$ (which implies $J \succ 0$), these results hold regardless of the values of a and b. For example setting $a = b = 1$, we obtain the correct covariance matrix even though J is not walk-summable.

5. Conclusion

We have presented an approach for exact inference of marginals in Gaussian graphical models, called the path-sum formulation. This approach represents the covariance matrix Σ of a Gaussian Markov Random Field (GMRF) as a continued fraction of finite depth and breadth, and which only involves the simple paths and simple cycles on the graphical model of the GMRF. For GMRFs, we have shown that the path-sum formulation only requires $J \succ 0$. It also extends to arbitrary partitions of J, thereby providing the flexibility necessary to exploit the sparsity structure of the information matrix $J = \Sigma^{-1}$.

From an algorithmic point of view, the computational cost associated with the calculation of any entry of a $n \times n$ covariance matrix using a path-sum is known to be $O(n)$ on trees,

see Giscard et al. (2013). The $O(n)$ complexity is achieved for the most general situation; if J or $\mathcal{G}$ has symmetries, the complexity would be lower. Evaluating the complexity of the path-sum approach on a graph with an arbitrary topology remains an open problem. In practice, finding the simple paths and simple cycles required for the computation of a path-sum can be achieved using standard depth-first or breadth-first algorithms (Johnson, 1975). In addition, we have consistently observed in numerical experiments that the contribution of a simple cycle/path to any path-sum decays exponentially with its length. The method we propose is therefore likely to find practical applications on sufficiently sparse yet large Gaussian graphical models where the increase in the number of simple cycles/paths with the length ℓ is offset by the decay of their contributions with ℓ.

Alternatively, path-sums can be used to enhance existing walk-sum calculations. To this end, we first factorise the walks involved in the walk-sum using an algorithm provided by Giscard et al. (2012). The typical computational cost of this operation grows linearly, and at worst quadratically, with the length of the walks. This yields a set of simple cycles/paths, from which a path-sum can be constructed. This path-sum is guaranteed to contain all the walks present in the original walk-sum as well as (infinitely) many more walks, all of which are valid walks on the graphical model. Since the method of path-sums arises from exact resummations on walk-sums, every problem that has a walk-sum interpretation is susceptible to these resummations and therefore admits a path-sum expression. Consequently, every algorithm that has a walk-sum interpretation (see e.g. Chandrasekaran et al., 2008) necessarily has a path-sum formulation.

The interpretation of the entries of the covariance matrix as walk-sums and the existence of the path-sum formulation opens the door to many more walk-based methods, which are intermediary between these two approaches. The central idea is to exploit recent results regarding the algebraic structure of the set of walks on graphs with arbitrary topology (Thwaite, 2014) to identify certain infinite geometric series of terms appearing in a walk-sum. These geometric series can then be exactly resummed, thereby reducing the sum of all walk weights to a sum over the weights of a certain (yet infinite) subset of 'irreducible' walks. Each term in this sum is 'dressed' so as to exactly include the contributions of the infinite families of resummed terms. The exact form of both the dressing and the irreducible terms remaining in the sum depend on the structure of the resummed terms: choosing a different family of terms produces a different series.

Viewed in this context, the walk-sum and path-sum formulations of a given problem can be seen to be the extrema of a hierarchy of possible resummations: a walk-sum corresponds to the case where *no* terms are exactly resummed, such that an explicit summation over all walks remains to be carried out, while the path-sum expression corresponds to the case where *all possible* geometric series have been resummed, leaving behind only a (finite) sum over simple paths. In between these extremes lie many intermediate formulations, whose mathematical properties such as complexity and convergence are expected to interpolate between those of walk-sums and path-sums.

Acknowledgments

P.-L. Giscard and D. Jaksch acknowledge funding from EPSRC Grant EP/K038311/1. P.-L. Giscard also receives support from the Royal Commission for the Exhibition of 1851. D. Jaksch acknowledges funding from the ERC under the European Unions Seventh Framework Programme (FP7/2007-2013)/ERC Grant Agreement no. 319286 Q-MAC. Z. Choo was funded by a departmental studentship of the Department of Statistics, University of Oxford. S. J. Thwaite acknowledges funding from the Alexander von Humboldt Foundation.

Appendix A.

In this appendix we prove Theorem 2.

Proof The proof of the theorem is organized in two steps. First, assuming walk-summability, we obtain the path-sum formulation for J^{-1}. Second, we show that path-sum expression thus obtained is the unique analytic continuation of the sum of all walks, and continues to exist in the absence of walk-summability. Consequently, the path-sum remains a valid representation of J^{-1} when walk-summability fails.

Step 1: the path-sum expression for J^{-1} is a special case of the more general result concerning the path-sum formulation of the matrix inverse function presented and proved in Giscard et al. (2013). Below we briefly recount how we obtain result for the specific case of J^{-1}. Let $R := I - J$ and $|R|$ be the entry-wise absolute value of R, i.e. $|R|_{ij} = |R_{ij}|$. Let $\rho(|R|)$ be the spectral radius of $|R|$, i.e. the largest eigenvalue of $|R|$. As established by Malioutov et al. (2006), $\rho(|R|) < 1$ is equivalent to to walk-summability, and is enough to guarantee absolute convergence of the series $\sum_{n\geq 0} R^n = (I-R)^{-1} = J^{-1}$. This power series can be seen as a sum of walk weights on the graph $\mathcal{G}$ associated with R (Flajolet and Sedgewick, 2009), that is

$$\Big(\sum_{n\geq 0} R^n\Big)_{\omega\alpha} = \sum_{w\in\mathcal{W}_{\mathcal{G};\,\alpha\omega}} \phi[w], \tag{10}$$

with $\phi[w]$ the weight of the walk w, which is defined to be the product of the weights of the edges traversed by w, where the weight of an edge from α to β is given by

$$\phi[\alpha\beta] := R_{\beta\alpha} = \begin{cases} -J_{\beta\alpha} & \text{if } \alpha \neq \beta, \\ 1 - J_{\alpha\alpha} & \text{if } \alpha = \beta. \end{cases} \tag{11}$$

Note that since we assumed walk-summability, the right-hand side of Eq. (10) exists. We then use the result by Giscard et al. (2012), which reduces a series of weighted walks, such as the one of Eq. (10), to a sum of weighted simple paths and simple cycles. This result is reproduced here for the sake of completeness:

Theorem 7 (Path-sum, Giscard et al., 2012) *Let $\mathcal{G}$ be a graph and $\phi[.]$ be the weight function of Eq. (11). Suppose that the walk-sum $\sum_{w\in\mathcal{W}_{\mathcal{G};\alpha\omega}} \phi[w]$ exists. Then this sum is given by the weighted path-sum*

$$\sum_{w\in\mathcal{W}_{\mathcal{G};\alpha\omega}} \phi[w] = \sum_{p\in\Pi_{\mathcal{G};\,\alpha\omega}} \prod_{j=1}^{\ell(p)+1} \Big\{ \phi_{\mathcal{G}\setminus\{\alpha,\,\nu_2,\ldots,\nu_{j-1}\};\,\nu_j}\, \phi[\nu_{j+1}\nu_j] \Big\}\, \phi_{\mathcal{G};\,\alpha},$$

where $\phi_{\mathcal{G};\alpha} := \sum_{w \in \mathcal{W}_{\mathcal{G};\alpha\alpha}} \phi[w]$ *is the weighted sum of all walks from* α *to itself on* $\mathcal{G}$ *and is explicitly given by*

$$\phi_{\mathcal{G};\alpha} = \left(1 - \sum_{\gamma \in \Gamma_{\mathcal{G};\alpha\alpha}} \phi[\mu_1 \mu_{\ell(\gamma)}] \prod_{j=2}^{\ell(\gamma)} \left\{ \phi_{\mathcal{G}\setminus\{\alpha,\mu_2,\ldots,\mu_{j-1}\};\mu_j} \, \phi[\mu_j \mu_{j-1}] \right\}\right)^{-1},$$

and similarly for all $\phi_{\mathcal{G}\setminus\{\alpha,\nu_2,\ldots,\nu_{j-1}\};\nu_j}$ *and* $\phi_{\mathcal{G}\setminus\{\alpha,\mu_2,\ldots,\mu_{j-1}\};\mu_j}$*. In these expressions, the products are right-to-left,* $p = (\nu_1, \nu_2, \ldots, \nu_{\ell(p)+1})$ *is a simple path of length* $\ell(p)$ *with* $\alpha \equiv \nu_1$ *and* $\omega \equiv \nu_{\ell(p)+1}$ *for convenience; and* $\gamma = (\mu_1, \mu_2, \ldots, \mu_{\ell_\gamma}, \mu_1)$ *is a simple cycle of length* $\ell(\gamma)$ *from* $\alpha \equiv \mu_1$ *to itself.*

By using the edge weights of Eq. (11), we obtain $\phi_{\mathcal{G};\alpha} = J^{-1}_{\alpha\alpha}$ and similarly $\phi_{\mathcal{G}\setminus\{\alpha,\nu_2,\ldots,\nu_{j-1}\};\nu_j} = (J_{\mathcal{G}\setminus\{\alpha,\nu_2,\ldots,\nu_{j-1}\}})^{-1}_{\nu_j\nu_j}$ and Theorem 7 yields Eqs. (2, 3). Next we prove that Eqs. (2, 3) yield J^{-1} for any matrix $J \succ 0$, even when walk-summability does not hold and the walk-sum of Eq. (10) does not converge. Central to our proof is the (well-established) theory of analytic continuation (Priestley, 2003).

Step 2: we consider the three following functions of a complex variable z into $\mathbb{C}^{n\times n}$, $g_1(z) := (I - zR)^{-1}$, $g_2(z) := \sum_{n\geq 0} z^n R^n$ and, since $g_2(z)$ is a walk-sum, it has a path-sum expression which we denote $g_3(z)$. (The path-sum expression $g_3(z)$ is obtained from Theorem 2 on using the edge-weights $zR_{\beta\alpha}$ for an edge from α to β).

The first function, $g_1(z)$, is analytic on $z \in \mathbb{C}\backslash \mathrm{Sp}^{-1}(R)$ with $\mathrm{Sp}^{-1}(R)$ the inverse of the spectrum of R. The second function, $g_2(z)$, is analytic on the disk D_R of the complex plane where $z < 1/\rho(|R|)$. The third function, $g_3(z)$, comprises only a finite number of terms (since there are finitely many simple paths and simple cycles on a finite graph). Consequently $g_3(z)$ exists for $z \in \mathbb{C}\backslash \mathrm{Sp}^{-1}(R)$. Evidently g_1 and g_2 agree on D_R and by construction g_2 and g_3 agree on D_R as well. It follows that g_1 and g_3 constitute two *direct analytic continuations* of g_2 outside of D_R (Priestley, 2003) (also called *extensions* of g_2, Kreyszig, 1989). By the Uniqueness Theorem (Theorem 15.9, Priestley, 2003) only one such analytic continuation exists and $g_1(z) = g_3(z)$ on the domain $\mathbb{C}\backslash \mathrm{Sp}^{-1}(R)$.

We conclude the proof by showing that the point $z = 1$, for which $g_1(z)$ and $g_3(z)$ yield J^{-1}, is also in this domain. Indeed, the covariance matrix Σ of a Gaussian distribution must be positive definite, implying that it is non-singular (Corollary 7.17 in Horn and Johnson, 2013). Consequently, J is not singular: so J^{-1} exists exists and 1 is not an eigenvalue of $R = I - J$, i.e. $1 \in \mathbb{C}\backslash \mathrm{Sp}^{-1}(R)$. Then g_1 and g_3 exist at $z = 1$, in particular $g_3(1)$ is a valid representation of J^{-1}, even though $z = 1$ may not be in D_R (i.e. J is not walk-summable). This completes the proof of Theorem 2. ■

References

N. Biggs. *Algebraic Graph Theory*. Cambridge University Press, Cambridge, 2nd edition, 1993.

V. Chandrasekaran, J. K. Johnson, and A. S. Willsky. Estimation in Gaussian graphical models using tractable subgraphs: a walk-sum analysis. *IEEE Transactions on Signal Processing*, 56:1916–1930, 2008.

S. Eastwood and J. W. L. Wan. Finding off-diagonal entries of the inverse of a large symmetric sparse matrix. *Numerical Linear Algebra with Applications*, 20:74–92, 2013.

P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, Cambridge, 1st edition, 2009.

P.-L. Giscard, S. J. Thwaite, and D. Jaksch. Continued fractions and unique factorization on digraphs. *arXiv:1202.5523*, 2012.

P.-L. Giscard, S. J. Thwaite, and D. Jaksch. Evaluating matrix functions by resummations on graphs: the method of path-sums. *SIAM Journal on Matrix Analysis & Applications*, 34:445–469, 2013.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2nd edition, 2013.

D. B. Johnson. Finding all the elementary circuits of a directed graph. *SIAM J. Comput.*, 4:77–84, 1975.

J. Johnson. Walk-summable Gauss-Markov random fields, 2002. unpublished manuscript, available at `http://ssg.mit.edu/~jasonj/johnson-walksum-tr02.pdf`.

B. Jones and M. West. Covariance decomposition in undirected Gaussian graphical models. *Biometrika*, 92:779–786, 2005.

E. Kreyszig. *Introductory Functional Analysis with Applications*. John Wiley & Sons, New York; London, revised edition, 1989.

S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

S. Li, S. Ahmed, G. Klimeck, and E. Darve. Computing entries of the inverse of a sparse matrix using the FIND algorithm. *Journal of Computational Physics*, 227:9408–9427, 2008.

L. Lin, C. Yang, J. C. Meza, L. Ying J. Lu, and W. E. SelInv - An algorithm for selected inversion of a sparse matrix. *ACM Transactions on Mathematical Software*, 37:1–19, 2011.

D. M. Malioutov, J. K. Johnson, and A. S. Willsky. Walk-sums and belief propagation in Gaussian graphical models. *Journal of Machine Learning Research*, 7:2031–2064, 2006.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Mateo, California, 1988.

H. A. Priestley. *Introduction to Complex Analysis*. Oxford University Press, 2nd edition, 2003.

H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall/CRC, 2005.

J. R. Silvester. Determinants of block matrices. *The Mathematical Gazette*, 84:460–467, 2000.

G. Strang. *Linear Algebra and Its Applications*. Brooks Cole, 4th edition, 2005.

J. M. Tang and Y. Saad. A probing method for computing the diagonal of a matrix inverse. *Numerical Linear Algebra with Applications*, 19:485–501, 2012.

S. J. Thwaite. A family of partitions of the set of walks on a directed graph. *arXiv:1409.3555*, 2014.

Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, 13:2173–2200, 2001.

Journal of Machine Learning Research 17 (2016) 1-54 Submitted 11/14; Revised 1/16; Published 4/16

Challenges in multimodal gesture recognition

Sergio Escalera SERGIO@MAIA.UB.ES
Computer Vision Center UAB and University of Barcelona
Vassilis Athitsos ATHITSOS@UTA.EDU
University of Texas
Isabelle Guyon GUYON@CHALEARN.ORG
ChaLearn, Berkeley, California

Editors: Zhuowen Tu

Abstract

This paper surveys the state of the art on multimodal gesture recognition and introduces the JMLR special topic on gesture recognition 2011-2015. We began right at the start of the Kinect^{TM} revolution when inexpensive infrared cameras providing image depth recordings became available. We published papers using this technology and other more conventional methods, including regular video cameras, to record data, thus providing a good overview of uses of machine learning and computer vision using multimodal data in this area of application. Notably, we organized a series of challenges and made available several datasets we recorded for that purpose, including tens of thousands of videos, which are available to conduct further research. We also overview recent state of the art works on gesture recognition based on a proposed taxonomy for gesture recognition, discussing challenges and future lines of research.

Keywords: Gesture Recognition, Time Series Analysis, Multimodal Data Analysis, Computer Vision, Pattern Recognition, Wearable sensors, Infrared Cameras, Kinect^{TM}.

1. Introduction

Gestures are naturally performed by humans. Gestures are produced as part of deliberate actions, signs or signals, or subconsciously revealing intentions or attitude. They may involve the motion of all parts of the body, but the arms and hands, which are essential for action and communication, are often the focus of studies. Facial expressions are also considered gestures and provide important cues in communication.

Gestures are present in most daily human actions or activities, and participate to human communication by either complementing speech or substituting themselves to spoken language in environments requiring silent communication (under water, noisy environments, secret communication, etc.) or for people with hearing disabilities. The importance of gestures in communication is rooted in primal behaviors: the gesture-first theory, supported by the analysis of mirror neurons in primates (Hewes, 1973), indicated that the first steps of language phylogenetically were not speech, nor speech with gesture, but were gestures alone (McNeil, 2012; Hewes, 1973). See examples of primate communication by means of gestures in Figure 1.

Given the indubitable importance of gestures in human activities, there has been huge interest by the Computer Vision and Machine Learning communities to analyze human gestures from visual data in order to offer new non-intrusive technological solutions. For completeness, in this paper we also review some gesture recognition systems with data acquired from wearable sensors, although the comprehensive review of papers focus on the analysis of different visual modalities.

Figure 1: Example of possible bonobo iconic gestures. (a) Start of swing gesture (or shove); (b) End of swing gesture (or shove); (c) Start of iconic swing, other bonobo starts to move; (d) End of iconic swing, other moving. Image from (McNeil, 2012).

Applications are countless, like Human Computer Interaction (HCI), Human Robot Interaction (HRI) (also named human machine interaction HMI), communication, entertainment, security, art, semiotics, commerce and sports, while having an important social impact in assistive technologies for the handicapped and the elderly. Some examples of applications are illustrated in Fig. 2.

In addition to the recent advances in human and gesture recognition from classical RGB visual data, the automatic analysis of human body from sensor data keeps making rapid progress with the constant improvement of (i) new published methods that constantly push the state-of-the-art and (ii) the recent availability of inexpensive 3D video sensors such as Kinect^{TM}, providing a complementary source of information, and thus allowing the computation of new discriminative feature vectors and improved recognition by means of fusion strategies. In section 2 we review the state of the art in gesture recognition.

In order to push research and analyze the gain of multimodal methods for gesture recognition, in 2011 and 2012, ChaLearn organized a challenge on single user one-shot-learning gesture recognition with data recorded with Kinect^{TM} in which 85 teams competed. Starting from baseline methods making over 50% error (measured in Leveinshtein distance, a metric counting the number of substitutions, insertions and deletions, analogous to an error rate), the winners brought the error rate below 10%. While there was still some margin of improvement on such tasks to reach human performance (which is below 2% error), we were encouraged to make the task harder to push the state of the art in computer vision. In our second ChaLearn challenge on Multimodal Gesture Recognition in 2013, we proposed a user-independent task with data recorded with Kinect^{TM}, with a larger vocabulary and continuously performed gestures. Of 60 participating teams, the winner attained an error rate of 10% on this data set, in terms of Leveinshtein distance. In 2014, we used the same Multimodal Gesture Recognition dataset with the objective of performing gesture spotting. The winner of the competition, with a deep learning architecture, obtained an overlapping near 0.9. Lastly, in 2014 and 2015 we ran an action spotting challenge with a new dataset consisting of RGB sequences of actors performing different isolated and collaborative actions in outdoor environments. Future challenges we are planning include the analysis of gestures taking into account face and contextual information, involving many modalities in the recognition process. In this paper we also review other existing international challenges related to gesture recognition.

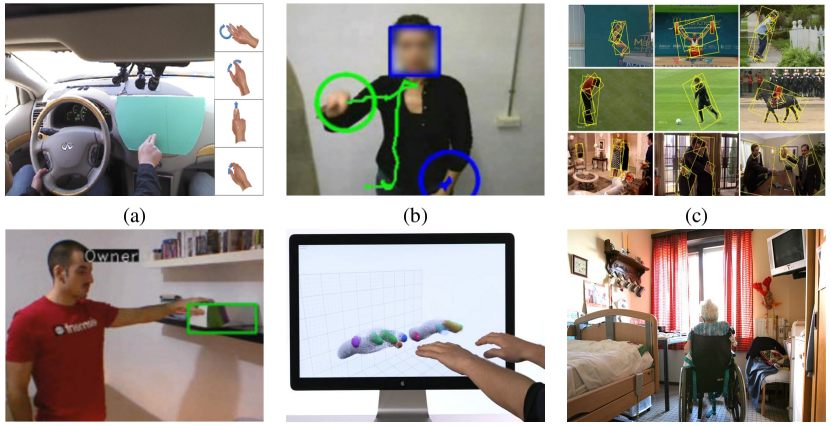


Figure 2: Some applications of gesture recognition. (a) Gesture recognition for driver assistance, from (Ohn-Bar and Trivedi, 2014), (b) Sign Language Recognition, (c) action/gesture recognition for content retrieval and categorization, from (Ma et al., 2013), (d) surveillance, (e) Human Computer/Robot/Machine Interaction, and (f) Assistive technology for people with reduced autonomy.

Our first workshop at CVPR from our 2011 challenge emphasized mostly 2D video data meanwhile our second and third workshops at CVPR, ICPR, ICMI, ECCV conferences from our 2012, 2013, and 2014 challenges were focused on affordable 3D sensors for gesture recognition research, also including audio information. In ECCV 2014 and CVPR 2015 workshops we also promoted different aspects of looking at people, including pose recovery, activity recognition, and scene understanding where humans are present. In addition to best challenge results, many research papers devoted to gesture recognition were published and presented in our challenge workshops. We also invited keynote speakers in diverse areas of pose and gesture research, including sign language recognition, body posture analysis, action and activity recognition, and facial expression or emotion recognition.

In this special topic on gesture recognition, extension of best challenge and workshop papers from previous events have been published. In addition, new description and learning strategies papers related to gesture recognition have been published. All of them will be shortly reviewed in the following sections.

The rest of the paper is organized as follows: Section 2 reviews the state of the art on gesture recognition, defining a taxonomy to describe existing works as well as available databases for gesture and action recognition. Section 3 describes the series of gesture and action recognition challenges organized by ChaLearn, describing the data, objectives, schedule, and achieved results by the participants. For completeness we also review other existing gesture challenge organizations. In Section 4 we review the published papers in this gesture recognition topic which are

related to ChaLearn competitions. Section 5 describes special topic published papers related to gesture recognition which are not based on ChaLearn competitions. Finally, Section 6 discusses main observations about the published papers.

2. Related Work in Gesture Recognition

In this section we present a taxonomy for action/gesture recognition, we review most influential works in the field, and finally we review existing datasets for action/gesture recognition together with the performance obtained by state of the art methods.

2.1 Taxonomy for gesture recognition

Fig. 3 is an attempt to create a taxonomy of the various components involved in conducting research in action/gesture recognition. We include various aspects relating to the problem setting, the data acquisition, the tools, the solutions, and the applications.

First, regarding the problem setting, the interpretation of gestures critically depends on a number of factors, including the environment in which gestures are performed, their span in time and space, and the intentional meaning in terms of symbolic description and/or the subconscious meaning revealing affective/emotional states. The problem setting also involves different actors who may participate in the execution of gestures and actions: human(s) and/or machine(s) (robot, computer, etc.), performing with or without tools or interacting or not with objects. Additionally, independently of the considered modality, for some gestures/actions different parts of the body are involved. While many gesture recognition systems only focus on arms and hands, full body motion/configuration and facial expressions can also play a very important role. Another aspect of the problem setting involves whether recognized gestures are static or dynamic. For the first case, just considering features from an input frame or any other acquisition device describing spatial configuration of body limbs, a gesture can be recognized. In the second case, the trajectory and pose of body limbs provide the highest discriminative information for gesture recognition. In some settings, gestures are defined based not only on the pose and motion of the human, but also on the surrounding context, and more specifically on the objects that the human interacts with. For such settings, one approach for achieving context awareness is scene analysis, where information is extracted from the scene around the subject (e.g., Pieropan et al. (2014); Shapovalova et al. (2011)). Another approach is to have the subject interact with intelligent objects. Such objects use embedded hardware and software to facilitate object recognition/localization, and in some cases to also monitor interactions between such objects and their environment (e.g., Czabke et al. (2010))

Second, the data are, of course, of very central importance, as in every machine learning application. The data sources may vary: when recognizing gestures, input data can come from different modalities, visual (RGB, 3D, or thermal, among others), audio, or wearable sensors (magnetic field trackers, instrumented (data) gloves, or body suits, among others). In the case of gloves, they can be active or passive. Active ones make use of a variety of sensors on a glove to measure the flexing of joints or the acceleration and communicates data to the host device using wired or wireless technology. Passive ones consist only of markers or colored gloves for finger detection by an external device such as a camera. Although most gestures are recognized by means of ambient intelligent systems, looking at the person from outside, some gesture recognition approaches are based on egocentric computing, using wearable sensors or wearable cameras that analyze, for instance, hand behaviors. Additionally, it is well-known that context provides rich information that can be useful

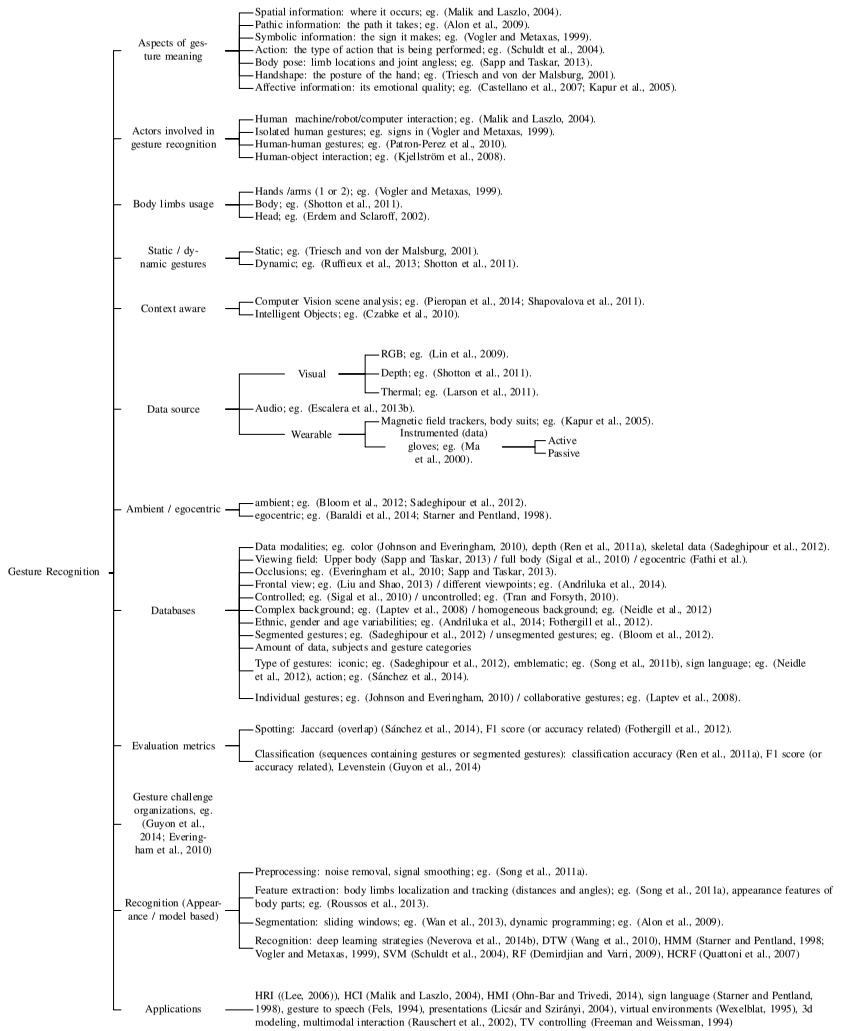


Figure 3: Taxonomy for gesture recognition.

to better infer the meaning of some gestures. Context information can be obtained by means of computer vision scene analysis, interaction with objects, but also via intelligent objects in the scene

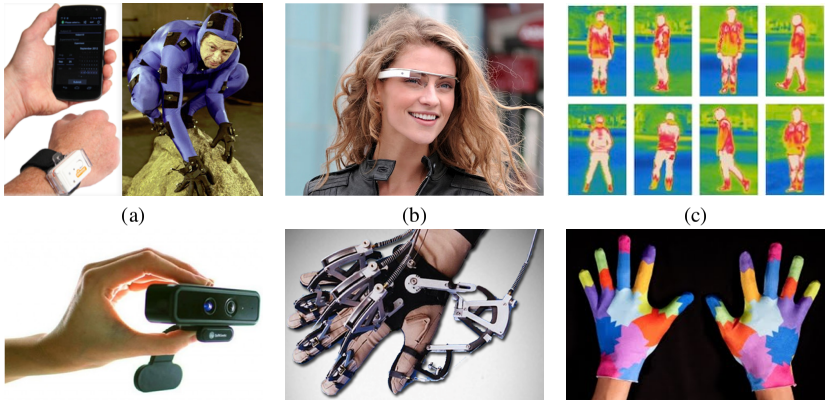


Figure 4: Some examples of acquisition devices for gesture recognition. (a) left: mobile with GPS and accelerometer, right: inertial sensor with accelerometer and gyroscope, (b) Google Glasses for egocentric computing, (c) thermal imagery for action recognition, (d) audio-RGB-depth device, (e) active glove, and (f) passive glove.

(objects with sensors that emit signals related to proximity and interaction). Some examples of acquisition devices are shown in Figure 4.

Third, the field of gesture recognition has shaped up thanks to the adoption of standard methodology. In order to advance in the design of robust action/recognition approaches, several datasets with different complexity have been published, and several world challenges helped to push the research in the area. This required the definition of standard evaluation metrics to render methods comparable. Notably, when one wants to recognize actions/gestures from data, common steps involve pre-processing of the acquired data, feature extraction, segmentation of begin-end of gesture and its final gesture/action label classification. Many datasets include preprocessed and/or thoroughly annotated data.

Fourth, gesture recognition has offered many opportunities to algorithm developers to innovate. The approaches, which essentially can be categorized into appearance-based and model-based methods, are going to be reviewed in the next section. We will mention only the most influential works for action/gesture recognition illustrating various aspects of the problem setting, data acquisition, and methodology defined in our taxonomy. Note that although we defined a general taxonomy for gesture recognition, in this paper, we put special emphasis on computer vision and machine learning methods for action/gesture recognition.

Finally, our taxonomy would not be complete without the wide array of applications of gesture/action recognition, already mentioned in the introduction.

2.2 Overview of gesture recognition methods

Different surveys have been published so far reviewing gesture recognition systems (LaViola Jr., 1999; Mitra and Acharya, 2007; Chaudhary et al., 2011; Ibraheem and Khan, 2012; Avci et al., 2010; Khan and Ibraheem; Kausar and Javed, 2011). In this section, we present an up-to-date review of most influential works in the field.

2.2.1 RECOGNIZING STATIC GESTURES AND HAND POSE

In the case of static gestures, frequently hand shape is the important differentiating feature (Cui and Weng, 2000; Freeman and Roth, 1996; Kelly et al., 2010; Ren et al., 2011b; Triesch and von der Malsburg, 2002), although the pose of the rest of the body can also be important, e.g., (Yang et al., 2010; Van den Bergh et al., 2009). For static hand pose classification, some approaches rely on visual markers, such as a color glove with a specific color for each finger, e.g., (Wang and Popović, 2009). Other approaches can recognize the hand pose on unadorned hands. Appearance-based methods, like (Moghaddam and Pentland, 1995; Triesch and von der Malsburg, 2002; Freeman and Roth, 1996; Wu and Huang, 2000), can be used for recognizing static hand postures observed from specific viewpoints.

Model-based methods for hand pose estimation (Oikonomidis et al., 2011; de La Gorce et al., 2011; Oikonomidis et al., 2010; Rehg and Kanade, 1995) typically match visual observations to instances of a predefined hand model. Single frame pose estimation methods try to solve the hand pose estimation problem without relying on temporal information (Athitsos and Sclaroff, 2003). Most recently, due to the advent of commercially available depth sensors, there is an increased interest in methods relying on depth data (Keskin et al., 2012; Mo and Neumann, 2006; Oikonomidis et al., 2011; Pugeault and Bowden, 2011; Lopes et al., 2014).

2.2.2 FROM BODY PART DETECTION TO HOLISTIC PATTERN DETECTION

Dynamic gestures are characterized by both the pose and the motion of the relevant body parts. Much effort has traditionally be put into detecting first **body parts** and then tracking their motion. In color videos, detecting hands can be quite challenging, although better performance can be achieved by placing additional constraints on the scene and the relative position of the subject and the hands with respect to the camera (Cui and Weng, 2000; Isard and Blake, 1998; Kolsch and Turk, 2004; Ong and Bowden, 2004; Stefanov et al., 2005; Stenger et al., 2003; Sudderth et al., 2004). Commonly-used visual cues for hand detection such as skin color, edges, motion, and background subtraction (Chen et al., 2003; Martin et al., 1998) may also fail to unambiguously locate the hands when the face, or other "hand-like" objects are moving in the background.

In (Li and Kitani, 2013) the authors propose a hand segmentation approach from egocentric RGB data by the combination of color and texture features. In(Baraldi et al., 2014), dense features are extracted around regions selected by a new hand segmentation technique that integrates super-pixel classification, temporal and spatial coherence. Bag of visual words and linear SVM are used for final representation and classification.

Depth cameras have become widely available in recent years, and hand detection (in tandem with complete body pose estimation) using such cameras (and also in combination with other visual modalities) can be performed sufficiently reliably for many applications (Shotton et al., 2011; Hernandez-Vela et al., 2012). The authors of (Ren et al., 2013) propose a part-based hand gesture recognition system using Kinect^{TM}sensor. Finger-EarthMover's Distance (FEMD) metric is proposed to measure the dissimilarity between hand shapes. It matches the finger parts while not the whole hand based on hand segmentation and contour analysis. The method is tested on their own 10-gesture dataset.

Instead of estimating hand position and/or body pose before recognizing the gesture, an alternative is to customize the recognition module so that it does not require the exact knowledge of hand positions, but rather accepts as input a list of several candidate hand locations (Alon et al., 2009; Sato and Kobayashi, 2002; Hernandez-Vela et al., 2013b).

Another approach is to use **global image/video features**. Such global features include motion energy images (Bobick and Davis, 2001), thresholded intensity images and difference images (Dreuw et al., 2006), 3D shapes extracted by identifying areas of motion in each video frame (Gorelick et al., 2007) and histograms of pairwise distances of edge pixels (Nayak et al., 2005). Gestures can also be modelled as rigid 3D patterns (Ke et al., 2005), from which features can be extracted using 3D extensions of rectangle filters (Viola and Jones, 2001). The work of (Kong et al., 2015) uses pixel-level attributes in a hierarchical architecture of 3D kernel descriptors, and efficient match kernel is used to recognize gestures from depth data.

Along similar lines, (Ali and Shah, 2010) propose a set of kinematic features that are derived from the optical flow for human action recognition in videos: divergence, vorticity, symmetric and antisymmetric flow fields, second and third principal invariants of flow gradient and rate of strain tensor, and third principal invariant of rate of rotation tensor, which define spatiotemporal patterns. These kinematic features are computed by Principal Component Analysis (PCA). Then multiple instance learning (MIL) is applied for recognition in which each action video is represented by a bag of kinematic modes. The proposal is evaluated on the RGB Weizmann and KTH action data sets, showing comparable result to state of the art performances.

Much effort has also been put into **spatiotemporal invariant features**. In (Yuan et al., 2011) the authors propose a RGB action recognition system based on a pattern matching approach, named naive Bayes mutual information maximization (NBMIM). Each action is characterized by a collection of spatiotemporal invariant features which are matched with an action class by measuring the mutual information between them. Based on this matching criterion, action detection is to localize a subvolume in the volumetric video space that has the maximum mutual information toward a specific action class. A novel spatiotemporal branch-and-bound (STBB) search algorithm is designed to efficiently find the optimal solution. Results show high recognition results on KTH, CMU, and MSR data sets, showing speed up inference in comparison with standard 3D branch-and-bound.

Another example is the paper of (Derpanis et al., 2013) in which a compact local descriptor of video dynamics is proposed for action recognition in RGB data sequences. The descriptor is based on visual spacetime oriented energy measurements. An associated similarity measure is introduced that admits efficient exhaustive search for an action template, derived from a single exemplar video, across candidate video sequences. The method is speeded up by means of a GPU implementation. Method is evaluated on UCF and KTH data sets, showing comparable results to state of the art methods.

The work of (Yang and Tian, 2014b) presents a coding scheme to aggregate low-level descriptors into the super descriptor vector (SDV). In order to incorporate the spatio-temporal information, the super location vector (SLV) models the space-time locations of local interest points in a compact way. SDV and SLV are combined as the super sparse coding vector (SSCV) which jointly models the motion, appearance, and location cues. The approach is tested on HMDB51 and Youtube with higher performance in comparison to state of the art approaches.

2.2.3 SEGMENTATION OF GESTURES AND GESTURE SPOTTING

Dynamic gesture recognition methods can be further categorized based on whether they make the assumption that gestures have already been segmented, so that the start frame and end frame of each gesture is known. Gesture spotting is the task of recognizing gestures in unsegmented video streams, that may contain an unknown number of gestures, as well as intervals were no gesture is being performed. Gesture spotting methods can be broadly classified into two general approaches: the direct approach, where temporal segmentation precedes recognition of the gesture class, and the indirect approach, where temporal segmentation is intertwined with recognition:

- **Direct methods** (also called heuristic segmentation) first compute low-level motion parameters such as velocity, acceleration, and trajectory curvature (Kang et al., 2004) or mid-level motion parameters such as human body activity (Kahol et al., 2004), and then look for abrupt changes (e.g., zero-crossings) in those parameters to identify candidate gesture boundaries.

- **Indirect methods** (also called recognition-based segmentation) detect gesture boundaries by finding, in the input sequence, intervals that give good recognition scores when matched with one of the gesture classes. Most indirect methods (Alon et al., 2009; Lee and Kim, 1999; Oka, 1998) are based on extensions of Dynamic Programming (DP) e.g., Dynamic Time Warping (DTW) (Darrell et al., 1996; Kruskal and Liberman, 1983), Continuous Dynamic Programming (CDP) (Oka, 1998), various forms of Hidden Markov Models (HMMs) (Brand et al., 1997; Chen et al., 2003; Stefanov et al., 2005; Lee and Kim, 1999; Starner and Pentland, 1998; Vogler and Metaxas, 1999; Wilson and Bobick, 1999), and most recently, Conditional Random Fields (Lafferty et al., 2001; Quattoni et al., 2007). Also hybrid probabilistic and dynamic programming approaches have been recently published (Hernandez-Vela et al., 2013a). In those methods, the gesture endpoint is detected by comparing the recognition likelihood score to a threshold. The threshold can be fixed or adaptively computed by a non-gesture garbage model (Lee and Kim, 1999; Yang et al., 2009), equivalent to silence models in speech.

When attempting to recognize unsegmented gestures, a frequently encountered problem is the *subgesture problem*: false detection of gestures that are similar to parts of other longer gestures. (Lee and Kim, 1999) address this issue using heuristics to infer the user's completion intentions, such as moving the hand out of camera range or freezing the hand for a while. An alternative is proposed in (Alon et al., 2009), where a learning algorithm explicitly identifies subgesture/supergesture relationships among gesture classes, from training data.

Another common approach for gesture spotting is to first extract features from each frame of the observed video, and then to provide a sliding window of those features to a recognition module, which performs the classification of the gesture (Corradini, 2001; Cutler and Turk, 1998; Darrell et al., 1996; Oka et al., 2002; Starner and Pentland, 1998; Yang et al., 2002)). Oftentimes, the extracted features describe the position and appearance of the gesturing hand or hands (Cutler and Turk, 1998; Darrell et al., 1996; Starner and Pentland, 1998; Yang et al., 2002)). This approach can be integrated with recognition-based segmentation methods.

2.2.4 ACTION AND ACTIVITY RECOGNITION

The work of (Li et al., 2010) presents an action graph to model explicitly the dynamics of 3D actions and a bag of 3D points to characterize a set of salient postures that correspond to the nodes

in the action graph. The authors propose a projection based sampling scheme to sample the bag of 3D points from the depth maps. In (Sminchisescu et al., 2006) it is proposed the first conditional/discriminative chain model for action recognition.

The work of (Zanfir et al., 2013) propose the non-parametric Moving Pose (MP) framework for low-latency human action and activity recognition. The moving pose descriptor considers both pose information as well as differential quantities (speed and acceleration) of the human body joints within a short time window around the current frame. The descriptor is used with a modified kNN classifier that considers both the temporal location of a particular frame within the action sequence as well as the discrimination power of its moving pose descriptor compared to other frames in the training set. The method shows comparable results to state of the art methods on MSR-Action3D and MSR-DailyActivities3D data sets.

In (Oreifej and Liu, 2013), it is proposed a new descriptor for activity recognition from videos acquired by a depth sensor. The depth sequence is described using a histogram capturing the distribution of the surface normal orientation in the 4D space of time, depth, and spatial coordinates. To build the histogram, 4D projectors are created, which quantize the 4D space and represent the possible directions for the 4D normal. Projectors are initialized using the vertices of a regular polychoron. Projectors are refined using a discriminative density measure, such that additional projectors are induced in the directions where the 4D normals are more dense and discriminative. The proposed descriptor is tested on MSR Actions 3D, MSR Gesture 3D, and MSR Daily Activity 3D, slightly improving state of the art results.

In (Wang et al., 2014), the authors propose to characterize the human actions with an "actionlet" ensemble model, which represents the interaction of a subset of human joints. Authors train an ensemble of SVM classifiers related to actionlet patterns, which includes 3D joint features, Local Occupancy Patterns, and Fourier Temporal Pyramid. Results on CMU MoCap, MSR-Action3D, MSR-DailyActivity3D, Cornell Activity, and Multiview 3D data sets show comparable and better performance than state of the art approaches.

The work of (Yang and Tian, 2014a) presents an approach for activity recognition in depth video sequences. Authors cluster hypersurface normals in a depth sequence to form the polynormal which is used to jointly characterize the local motion and shape information. In order to globally capture the spatial and temporal orders, an adaptive spatio-temporal pyramid is introduced to subdivide a depth video into a set of space-time grids. It is then proposed a scheme of aggregating the low-level polynormals into the super normal vector (SNV) which can be seen as a simplified version of the Fisher kernel representation. Authors validate the proposed approach on MSRAction3D, MSRDailyActivity3D, MSRGesture3D, and MSRActionPairs3D data sets slightly improving in all cases state of the art performances.

In (Yu et al., 2014) the authors propose the orderlets to capture discriminative information for gesture recognition from depth maps. Orderlet features are discovered looking for frequent sets of skeleton joints that provide discriminative information. Adaboost is used for orderlets selection. Results on the ORGBD data set shows a recognition rate of 71.4% mean class average accuracy, improving by near 5% state of the art results on this data set, and near 20% improvement regarding frame level classification. However the results showed on the MSR-DailyActivity3D data set are inferior to the ones reported in (Luo et al., 2014).

The work of (Liang et al., 2014) presents a depth-based method for hand detection and pose recognition by segmentation of different hand parts. Authors based on RF for initial multipart hand

segmentation. Then, a Superpixel-Markov Random Field (SMRF) parsing scheme is used to enforce the spatial smoothness and the label co-occurrence prior to remove the misclassified regions.

2.2.5 APPROACHES USING NON-VIDEO MODALITIES AND MULTIMODAL APPROACHES

In terms of multimodal approaches for gesture recognition, (Luo et al., 2014) propose a sparse coding-based temporal pyramid matching approach (ScTPM) for feature representation using depth maps. The authors also propose the Center-Symmetric Motion Local Ternary Pattern (CS-Mltp) descriptor to capture spatial-temporal features from RGB videos. By fusing both RGB and Depth descriptors, the authors improve state of the art results on MSR-Action3D and MSR-DailyActivity3D data sets, with a 6% and 7% of improvement, respectively.

In (Ionescu et al., 2014), it is presented the Human3.6M data set, consisting of 3.6Million accurate 3D Human poses, acquired by recording the performance of 5 female and 6 male subjects, under 4 different viewpoints, for training realistic human sensing systems and for evaluating the next generation of human pose estimation models and algorithms. Authors also provide a set of large scale statistical models and evaluation baselines for the dataset illustrating its diversity.

In (Xiao et al., 2014) a wearable Immersion CyberGlove II is used to capture the hand posture and the vision-based Microsoft Kinect^{TM}takes charge of capturing the head and arm posture. An effective and real-time human gesture recognition algorithm is also proposed.

In (Liang et al., 2013) it is proposed to detect and segment different body parts using RGB and Depth data sequences. The method uses both temporal constraints and spatial features, and performs hand parsing and 3D fingertip localization for hand pose estimation. The hand parsing algorithm incorporates a spatial-temporal feature into a Bayesian inference framework to assign the correct label to each image pixel. The 3D fingertip localization algorithm adapts is based on geodesic extrema extraction to fingertip detection. The detected 3D fingertip locations are finally used for hand pose estimation with an inverse kinematics solver. The work of (Joshi et al., 2015) use random forest for both segmenting and classifying gesture categories from data coming from different sensors.

Although many works base only on inertial data (Benbasat and Paradiso, 2001; Berlemont et al., 2015), multimodal approaches are often considered in order to combine trajectory information will pose analysis based on visual data. The works of (Liu et al., 2014; Pardo et al., 2013) present approaches for gesture recognition based on the combination of depth and inertial data. In (Liu et al., 2014) skeleton obtained from depth data and data from inertial sensors are train within HMM in order to perform hand gesture recognition. A similar approach is presented in (Pardo et al., 2013), but also recognizing objects present in the scene and using DTW for recognition with the objective of performing ambient intelligent analysis to support people with reduced autonomy. In (Gowing et al., 2014), it is presented a comparison of WIMU aWireless/Wearable Inertial Measurement Unit and Kinect^{TM}. However, comparison is performed independently, without considering a fusion strategy.

The work of (Appenrodt et al., 2009) is one of the few that compare the performance of different segmentation approaches for gesture recognition comparing RGB, depth, and thermal modalities. They propose a simple segmentation approach of faces and one hand for recognizing letters and numbers for HCI. They obtained higher performance by the use of depth maps. Unfortunately no mutimodal fusion approaches are tested in order to analyze when each modality can complement the information provided by the rest of modalities.

The work of (Escalera et al., 2013b) summarizes a 2013 challenge on multimodal gesture recognition, where in addition to RGB and depth data, audio can be used to identify the performed gestures.

Few works considered context information in order to improve gesture/action recognition systems. In (Wilhelm) it is proposed to adapt gesture recognition based on a dialogue manager as a partially observable Markov decision process (POMDP). In (Caon et al., 2011) two Kinect^{TM}devices and smart objects are used to estimate proximity and adapt the recognition prior of some gestures.

The recent emergence of deep learning systems in computer vision have also been applied to action/gesture recognition systems. In (Neverova et al., 2014a), it is presented a deep learning based approach for hand pose estimation, targeting gesture recognition. The method integrates local and global structural information into the training objective. In (Nagi et al., 2011), deep neural network (NN) combining convolution and max-pooling (MPCNN) is proposed for supervised feature learning and classification of RGB hand gestures given by humans to mobile robots using colored gloves. The hand contour is retrieved by color segmentation, then smoothed by morphological image processing which eliminates noisy edges. The system classifies 6 gesture classes with 96% accuracy, improving performance of several state of the art methods. The work of (Duffner et al., 2014) presents an approach that classifies 3D gestures using jointly accelerometer and gyroscope signals from a mobile device using convolutional neural network with a specific structure involving a combination of 1D convolution. In (Molchanov et al., 2015) convolutional deep neural networks are used to fuse data from multiple sensors (short-range radar, a color camera, and a depth camera) and to classify the gestures in a driver assistance scenario.

2.3 Sign language recognition

An important application of gesture recognition is sign language recognition. American Sign Language (ASL) is used by 500,000 to two million people in the U.S. (Lane et al., 1996; Schein, 1989). Overall, national and local sign languages are used all over the world as the natural means of communication in deaf communities.

Several methods exist for recognizing isolated signs, as well as continuous signing. Some researchers have reported results on continuous signing with vocabularies of thousands of signs, using input from digital gloves, e.g., (Yao et al., 2006). However, glove-based interfaces are typically expensive for adoption by the general public, as well as intrusive, since the user has to wear one or two gloves connected with wires to a computer.

Computer vision methods for sign language recognition offer hope for cheaper, non-intrusive interfaces compared to methods using digital gloves. Several such methods have been proposed (Bauer et al., 2000; Cui and Weng, 2000; Dreuw et al., 2006; Kadir et al., 2004; Starner and Pentland, 1998; Vogler and Metaxas, 1999; Wang et al., 2010; Zieren and Kraiss, 2005). However, computer vision methods typically report lower accuracies compared to methods using digital gloves, due to the difficulty of extracting accurate information about the articulated pose and motion of the signer.

An important constraint limiting the accuracy of computer vision methods is the availability of training data. Using more examples per sign typically improves accuracy (see, e.g., (Kadir et al., 2004; Zieren and Kraiss, 2005)). However, existing datasets covering large vocabularies have only a limited number of examples per sign. As an example, the ASLLVD dataset (Athitsos et al., 2008) includes about 3,000 signs, but only two examples are available for most of the signs. Some interest-

ing research has aimed at enabling automated construction of large datasets. For example, (Cooper and Bowden, 2009) aim at automatically generating large corpora by automatically segmenting signs from close-captioned sign language videos. As another example, (Farhadi et al., 2007) propose a method where sign models are learned using avatar-produced data, and then transfer learning is used to create models adapted for specific human signers.

The recent availability of depth cameras such as Kinect^{TM}has changed the methodology and improved performance. Depth cameras provide valuable 3D information about the position and trajectory of hands in signing. Furthermore, detection and tracking of articulated human motion is significantly more accurate in depth video than in color video. Several approaches have been published in recent years that use depth cameras to improve accuracy in sign language recognition Conly et al. (2015); Wang et al. (2015a); Zafrulla et al. (2011).

2.4 Data sets for gesture and action recognition

Tens of gesture recognition datasets have been made available to the research community over the last several years. A summary of available datasets is provided in Table 1. In that table, for each data set we mark some important attributes of the dataset, such as the type of gestures it contains, the data modalities it provides, the viewing field, background, amount of data, and so on. Regarding the "occlusions" attribute in that table, we should clarify that it only refers to occlusions of the subject by other objects (or subjects), and not to self occlusions. Self occlusions are quite common in gestures, and are observed in most datasets. We should also note that, regarding the complexity of the background, dynamic and/or cluttered backgrounds can make gesture recognition challenging in color images and video. At the same time, a complex background can be quite easy to segment if depth or skeletal information is available, as is the case in several datasets on Table 1.

In order to be able to fit Table 1 in a single page, we had to use abbreviations quite heavily. Table 2 defines the different acronyms and abbreviations used in Table 1.

The datasets we have created for our challenges have certain unique characteristics, that differentiate them from other existing datasets. The CDG2011 dataset (Guyon et al., 2014) has a quite diverse collection of gesture types, including static, pantomime, dance, signs, and activities. This is in contrast to other datasets, that typically focus on only one or maybe two gesture types. Furthermore, the CDG2011 dataset uses an evaluation protocol that emphasizes one-shot learning, whereas existing data sets typically have several training examples for each class. The CDG2013 dataset introduces audio data to the mix of color and depth that is available in some other data sets, such as (Sadeghipour et al., 2012; Bloom et al., 2012).

Dataset	Actors/ Objects	Body parts	Static/ Dynamic	Modalities	Viewing field	Occl.	View. Var.	Controlled Background	Variab.	Segm.	Amount	Subjects	Classes	Type	Indiv./ collab.	Eval.
HUMANEVA (Sigal et al., 2010)	H	F	SF	MC,ST	F	No	V	C	E,G	SF	40000 F	5	CBP	BP	I	MPJPE
Human3.6M (Ionescu et al., 2014)	H,O	F	SF	MC,D,ST	F	No	V	C	SF,C	G,AU,EU	SF	3.6M	11	CBP	BP	I	Multiple
LEEDS SPORTS (Johnson and Everingham, 2010)	H,O	F	SF	C	F	No	V	U	C,SF	AY,E,G	SF	2000 F	U1	CBP	BP	I	Fer.
Pascal VOC people (Everingham et al., 2010)	H,HH	F	SF	C	F	Some	V	U	C,SF	A,E,G	SF	632 F	U1	CBP	BP	B	Pascal
UIUC People (Tran and Forsyth, 2010)	H,O	F	SF	C	F	No	V	U	C,SF	AY,E,G	SF	593 F	U1	CBP	BP	I	Fer.
Buffy (Ferrari et al., 2008)	H,HH,O	U	SF	C	U	Some	V	U	C,SF	AY,G	SF	748 F	UM	CBP	BP	B	Fer.
Parse (Ramanan, 2006)	H,O	F	SF	C	F	No	V	U	C,SF	AY,E,G	SF	305 F	U1	CBP	BP	MI	Ram.
MPII Pose (Andriluka et al., 2014)	H,O	F	SF	C	F	Yes	V	U	C,SF	A,E,G	SF	25000 F	U1	CBP	BP	MI	Fer.
FLIC Pose (Sapp and Taskar, 2013)	H,HH,O	U	SF	C	U	Some	V	U	C,SF	AY,E,G	SF	5003 F	UM	CBP	BP	MI	Sap.
H3D (Bourdev and Malik, 2009)	H,HH,O	U	SF	C	U	Some	V	U	C,SF	A,E,G	SF	520 F	US	CBP	BP	MI	Sap.
CDG2011 (Guyon et al., 2014)	H	Mixed	Mixed	C,D	MU	Few	Fixed	C	ST,C	E,G	No	50000 G	20	CDG11	Mixed	I	L
CDG2013 (Escalera et al., 2013b)	H	H	Dynamic	A,C,D,ST	U	No	F	C	ST,C	G	No	13858 G	27	20	E	I	L
3DIG (Sadeghipour et al., 2012)	H	H	Dynamic	C,D,ST	U	No	F	C	ST,U	AY,E,G	Yes	1739 G	29	20	I	I	F
HuPBA8K+ (Sánchez et al., 2014)	H,HH	F	Dynamic	C	F	Yes	Fixed	C	ST,U	G, AY	No	8000 F	14	11	A	B	L
HOHA (Laptev et al., 2008)	H,HH,HO	F	Dynamic	C	F	Some	V	U	SD,C	AY,E,G	Yes	475 V	UM	8	A	B	CA
KTH (Schuldt et al., 2004)	H	F	Dynamic	G	F	No	SV	C	ST,U	G	Yes	2391 A	25	6	A	I	CA
MSRC-12 (Fothergill et al., 2012)	H	F	Dynamic	ST	F	No	F	C	N/A	A,E,G	No	719359 F	30	12	E,I	I	F
G3D (Bloom et al., 2012)	H	F	Dynamic	C,D,ST	F	No	Fixed	C	ST,C	U	No	80000 F	10	20	A	I	F
ASLLVD (Neidle et al., 2012)	H	H	Dynamic	MC	U	No	F	C	ST,U	G	Yes	9794 G	6	3314	S	I	RRC
UTA ASL (Conly et al., 2013)	H	H	Dynamic	C,D	U	No	F	C	ST,U	E,A	Yes	1313 G	2	1113	S	I	RCC
ChAirGest (Ruffieux et al., 2013)	H	H	Dynamic	ChAir	U	No	F	C	ST,C	U	No	1200 G	10	10	I,E	I	F,ATSR
SKIG (Liu and Shao, 2013)	H	H	Dynamic	C,D	A	No	F	C	ST,CU	U	Yes	1080 G	10	6	I,E	I	CA
6DMG (Chen et al., 2012)	H	H	Dynamic	6DMG	H	No	F	C	N/A	G,AU,EU	Yes	5600 G	28	20	I	I	CA
MSRGesture3D (Kurakin et al., 2012)	H	H	Dynamic	B	H	No	Fixed	C	N/A	U	Yes	336 G	10	12	S	I	CA
NATOPS (Song et al., 2011b)	H	U	Dynamic	S,D,B	U	No	Fixed	C	ST,U	U	Yes	9600 G	20	24	E	I	CA
NTU Dataset (Ren et al., 2011a)	H	H	Static	C,D	U	No	F	C	SF,C	G	SF	1000 G	10	10	HS	I	CA
Keck Gesture (Lin et al., 2009)	H	H	Dynamic	C	F	No	F	C	ST,U	No	Yes	294 G	3	14	E	I	CA
Cambridge Gesture (Kim et al., 2007)	H	H	Dynamic	C	H	No	F	C	ST,U	U	Yes	900 G	2	9	E	I	CA
(Triesch and von der Malsburg, 2001)	H	H	Static	G	H	No	Fixed	C	SF,CU	U	SF	717 G	24	10	HS	I	CA

Table 1: Datasets. "Occl." stands for "occlusions". "View. variab." stands for "viewpoint variabilities". "Variab." stands for "variabilities in gender, age, ethnicity". "Seg." stands for "segmented". "Indiv/Collab." stands for "individual or collaborative gestures". "Type" stands for "gesture type". "Eval." stands for "evaluation".

Taxonomy Attribute	Acronym/Abbreviation	Meaning
Actors/Objects	HH	human-human interactions
Actors/Objects	IH	isolated human
Actors/Objects	O	humans with objects
Body parts	F	full body
Body parts	H	hands
Static/Dynamic	SF	Subjects are in motion, but each frame is individually classified
Modalities	6DMG	WorldViz PPT-X4 (position + 3D orientation) + Wii Remote Plus (acceleration and angular speeds)
Modalities	A	audio
Modalities	B	binary segmentation mask
Modalities	C	RGB (color)
Modalities	ChAir	RGB, depth, skeletal, four inertial motion units
Modalities	D	depth
Modalities	G	grayscale
Modalities	MC	multiple cameras
Modalities	S	stereo images
Modalities	ST	skeletal tracking
Viewing field	A	arm and hand
Viewing field	E	egocentric
Viewing field	F	full body
Viewing field	MU	upper body in most cases
Viewing field	U	upper body
Occlusions		
Viewpoints	F	Frontal
Viewpoints	Fixed	Fixed viewpoint for each class
Viewpoints	SV	Fixed viewpoint for some classes, variable for other classes
Viewpoints	V	Variable viewpoint
Controlled/Uncontrolled	C	Controlled
Controlled/Uncontrolled	U	Uncontrolled
Background	CU	some cluttered, some uncluttered
Background	C	cluttered
Background	D	dynamic
Background	SF	each frame is individually classified, so background is seen only from a single frame
Background	SD	static in some cases, dynamic in some cases
Background	ST	static
Background	U	uncluttered
Variabilities in gender/age/ethnicity	A	variabilities in age
Variabilities in gender/age/ethnicity	AU	unspecified whether there are variabilities in age
Variabilities in gender/age/ethnicity	AY	mostly non-senior adults
Variabilities in gender/age/ethnicity	E	variabilities in ethnicity
Variabilities in gender/age/ethnicity	EU	unspecified whether there are variabilities in ethnicity
Variabilities in gender/age/ethnicity	G	variabilities in gender
Variabilities in gender/age/ethnicity	U	unspecified
Segmented/Unsegmented	SF	each frame is individually classified
Amount of data	A	action samples
Amount of data	F	frames
Amount of data	G	gesture samples
Amount of data	V	video clips
Number of subjects	U1	Unspecified, but most subjects appear in only one sample
Number of subjects	UM	Unspecified, but most subjects appear in several samples
Number of subjects	US	Unspecified, but some subjects appear in more than one sample
Classes	CBP	continuous space of body
Classes	CDG11	about 300, but broken into subsets of 8-12 classes
Gesture type	A	action
Gesture type	BP	body pose
Gesture type	D	deictic
Gesture type	E	emblematic
Gesture type	HS	handshape
Gesture type	I	iconic
Gesture type	S	sign
Individual or collaborative	B	both individual and collaborative
Individual or collaborative	C	collaborative
Individual or collaborative	I	individual
Individual or collaborative	MI	mostly individual
Evaluation criteria	ATSR	Defined in (Ruffieux et al., 2013), based on difference between detected and ground truth endpoints, normalized by duration of the gesture
Evaluation criteria	CA	Classification accuracy
Evaluation criteria	F	F-score
Evaluation criteria	Fer.	Defined in (Ferrari et al., 2008), checks if detected endpoints are within distance of half length (of the body part in question) from the ground truth position.
Evaluation criteria	L	Levenshtein distance
Evaluation criteria	MPJPE	Mean per-joint position error (measured as Euclidean distance).
Evaluation criteria	Pascal	At least 50% overlap of bounding boxes on all body parts.
Evaluation criteria	Ram.	Defined in (Ramanan, 2006), average negative log likelihood of correct pose.
Evaluation criteria	RCC	Defined in (Wang et al., 2010), based on rank of the correct class for each test sign. For any R, report percentage of test signs for which the correct class was in the top R classes.
Evaluation criteria	Sap.	Defined in (Sapp and Taskar, 2013), Accuracy is based on (variable) threshold pixel distance between joint location and ground truth, scaled so that the torso length in the ground truth is 100 pixels.

Table 2: Acronyms and abbreviations used in the table of datasets.

3. Gesture recognition challenges

In this section we review the series of gesture and action recognition challenges organized by ChaLearn from 2011 to 2015, as well as other international challenges related to gesture recognition.

3.1 First ChaLearn Gesture Recognition Challenge (2011-2012): One shot learning

ChaLearn launched in 2012 a challenge with prizes donated by Microsoft using datasets described in (Guyon et al., 2014). We organized two rounds in conjunction with the CVPR conference (Providence, Rhode Island, USA, June 2012) and the ICPR conference (Tsukuba, Japan, November 2012). Details on the challenge setting and results are found in (Guyon et al., 2013). We briefly summarize the setting and results.

3.1.1 2011-2012 CHALLENGE PROTOCOL AND EVALUATION

The task of the challenge was to built a learning system capable of learning a gesture classification problem from a **single training example** per class, from dynamic video data complemented by a depth map obtained with Kinect^{TM}. The rationale behind this setting is that, in many computer interface applications to gesture recognition, users want to customize the interface to use their own gestures. Therefore they should be able to retrain the interface using a small vocabulary of their own gestures. We have also experimented with other use cases in gaming and teaching gesture vocabularies. Additionally, the problem of one-shot-learning is of intrinsic interest in machine learning and the solutions deviced could carry over to other applications. It is in a certain way an extreme case of transfer learning.

To implement this setting in the challenge, we collected a large dataset consisting of batches, each batch corresponding to the video recording of short sequences of gestures performed by the same person. The gestures in one batch pertained to a small vocabulary of gestures taken from a variety of application domains (sign language for the deaf, traffic signals, pantomimes, dance postures, etc.). During the development phase, the participants had access to hundreds of batches of diverse gesture vocabularies. This played the role of "source domain data" in the transfer learning task. The goal of the participants was to get ready to receive new batches from different gesture performers and different gesture vocabularies, playing the role of "transfer domain data". Their system would then need to learn from a single example of gesture performed by the particular performer, before being capable of recognizing the rest of the gestures in that batch. The full dataset is available from `http://gesture.chalearn.org/data`.

More specifically, each batch was split into a training set (of one example for each gesture) and a test set of short sequences of one to 5 gestures. Each batch contained gestures from a different small vocabulary of 8 to 12 gestures, for instance diving signals, signs of American Sign Language representing small animals, Italian gestures, etc. The test data labels were provided for the development data only (source domain data), so the participants could self-evaluate their systems and pre-train parts of it as is expected from transfer learning methods. The data also included 20 validation batches and 20 final evaluation batches as transfer domain data used by the organizers to evaluate the participants. In those batches, only the labels of the training gestures (one example each) was provided, the rest of the gesture sequences were unlabelled and the goal of the participants

was to predict those labels. We used the Kaggle platform to manage submissions[1] The participants received immediate feed-back on validation data on a on-line leaderboard. The final evaluation was carried out on the final evaluation data, and those results were only revealed after the challenge was over. The participants had a few days to train their systems and upload their predictions. Prior to the end of the development phase, the participants were invited to submit executable software for their best learning system to a software vault. This allowed the competition organizers to check their results and ensure the fairness of the competition.

To compare prediction labels for gesture sequences to the truth values, we used the generalized Levenshtein distances (each gesture counting as one token). The final evaluation score was computed as the sum of such distances for all test sequences, divided by the total number of gestures in the test batch. This score is analogous to an error rate. However, it can exceed one. Specifically, for each video, the participants provided an ordered list of labels R corresponding to the recognized gestures. We compared this list to the corresponding list of labels T in the prescribed list of gestures that the user had to play. These are the "true" gesture labels (provided that the users did not make mistakes). We computed the generalized Levenshtein distance $L(R,T)$, that is the minimum number of edit operations (substitution, insertion, or deletion) that one has to perform to go from R to T (or vice versa). The Levenhstein distance is also known as "edit distance". For example: $L([124],[32]) = 2; L([1],[2]) = 1; L([222],[2]) = 2$.

We provided code to browse though the data, a library of computer vision and machine learning techniques written in Matlab featuring examples drawn from the challenge datasets, and an end-to-end baseline system capable of processing challenge data and producing a sample submission. The competition pushed the state of the art considerably. The participants narrowed down the gap in performance between the baseline recognition system initially provided ($\simeq$ 60% error) and human performance ($\simeq$ 2% error) by reaching $\simeq$ 7% error in the second round of the challenge. There remains still much room for improvement, particularly to recognize static postures and subtle finger positions.

3.1.2 2011-2012 CHALLENGE DATA

The datasets are described in details in a companion paper (Guyon et al., 2014). Briefly, the data are organized in batches: development batches devel01-480, validation batches valid01-20, and final evaluation batches final01-20 (for round 1) and final21-40 (for round 2). For the development batches, we provided all the labels. To evaluate the performances on "one-shot-learning" tasks, the valid and final batches were provided with labels only for **one example of each gesture class** in each batch (training examples). The goal was to automatically predict the gesture labels for the remaining unlabelled gesture sequences (test examples).

Each batch includes 100 recorded gestures grouped in sequences of 1 to 5 gestures performed by the same user. The gestures are drawn from a small vocabulary of 8 to 12 unique gestures, which we call a "lexicon". For instance a gesture vocabulary may consist of the signs to referee volleyball games or the signs to represent small animals in the sign language for the deaf. We selected lexicons from nine categories corresponding to various settings or application domains (Figure 5):

1. **Body language** gestures (like scratching your head, crossing your arms).

1. For round 1: `http://www.kaggle.com/c/GestureChallenge`. For round 2: `http://www.kaggle.com/c/GestureChallenge2`.

Team	Public score on validation set	Private score on final set #1	For comparison score on final set #2
Alfnie	0.1426	0.0996	0.0915
Pennect	0.1797	0.1652	0.1231
OneMillionMonkeys	0.2697	0.1685	0.1819
Immortals	0.2543	0.1846	0.1853
Zonga	0.2714	0.2303	0.2190
Balazs Godeny	0.2637	0.2314	0.2679
SkyNet	0.2825	0.2330	0.1841
XiaoZhuwWudi	0.2930	0.2564	0.2607
Baseline method 1	0.5976	0.6251	0.5646

Table 3: Results of round 1. In round 1 the baseline method was a simple template matching method (see text). For comparison, we show the results on the final set number 2 not available in round 1.

2. **Gesticulations** performed to accompany speech.
3. **Illustrators** (like Italian gestures).
4. **Emblems** (like Indian Mudras).
5. **Signs** (from sign languages for the deaf).
6. **Signals** (like referee signals, diving signals, or Marshalling signals to guide machinery or vehicle).
7. **Actions** (like drinking or writing).
8. **Pantomimes** (gestures made to mimic actions).
9. **Dance postures.**

During the challenge, we did not disclose the identity of the lexicons and of the users.

3.1.3 2011-2012 CHALLENGE RESULTS

The results of the top ranking participants were checked by the organizers who reproduced their results using the code provided by the participants before they had access to the final evaluation data. All of them passed successfully the verification process. These results are shown in Tables 3 and 4.

3.1.4 2011-2012 CHALLENGE, SUMMARY OF THE WINNER METHODS

The results of the challenge are analyzed in details, based on papers published in this special topic and on descriptions provided by the top ranking participants in their fact sheets (Guyon et al., 2013). We briefly summarize notable methods below.

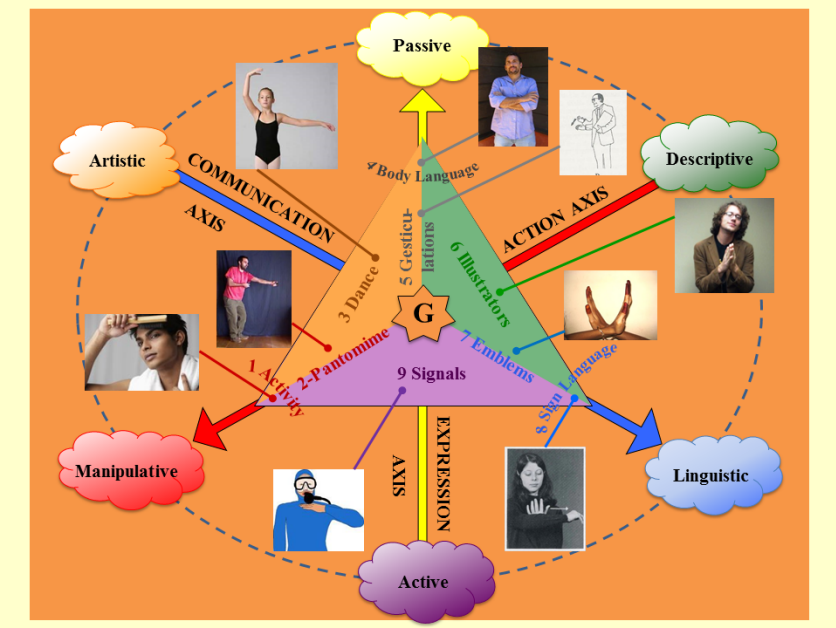


Figure 5: **Types of gestures.** We created a classification of gesture types according to purpose defined by three complementary axes: communication, expression and action. We selected 85 gesture vocabularies, including Italian gestures, Indian Mudras, Sign language for the deaf, diving signals, pantomimes, and body language.

The winner of both rounds (Alfonso Nieto Castañon of Spain, a.k.a. alfnie) used a novel technique called "Motion Signature analyses", inspired by the neural mechanisms underlying information processing in the visual system. This is an unpublished method using a sliding window to perform simultaneously recognition and temporal segmentation, based solely on depth images. The method, described by the authors as a "Bayesian network", is similar to a Hidden Markov Model (HMM). It performs simultaneous recognition and segmentation using the Viterbi algorithm. The preprocessing steps include Wavelet filtering replacement of missing values and outlier detection. Notably, this method is one of the fastest despite the fact that he implemented it in Matlab (close to real time on a regular laptop). The author claims that it has linear complexity in image size, number of frames, and number of training examples.

The second best ranked participants (team Pennect of Universit of Pennsylvania, USA, in round 1 and team Turtle Tamers of Slovakia, in round 2) used very similar methods and performed similarly. The second team published their results in this special topic (Konecny and Hagara, 2014).

Team	Public score on validation set	For comparison score on final set #1	Private score on final set #2
Alfnie	0.0951	0.0734	0.0710
Turtle Tamers	0.2001	0.1702	0.1098
Joewan	0.1669	0.1680	0.1448
Wayne Zhang	0.2814	0.2303	0.1846
Manavender	0.2310	0.2163	0.1608
HIT CS	0.1763	0.2825	0.2008
Vigilant	0.3090	0.2809	0.2235
Baseline method 2	0.3814	0.2997	0.3172

Table 4: Results of round 2. In round 2, the baseline method was the "Principal Motion" method (see text).

Both methods are based on an HMM-style model using HOG/HOF features to represent movie frames. They differ in that Pennect used RGB images only while Turtle Tamers used both RGB and depth. Another difference is that Pennect used HOG/HOF features at 3 different scales while Turtle Tamers created a bag of features using K-means clustering from only 40x40 resolution and 16 orientation bins. Pennect trained a one-vs-all linear classifier for each frame in every model and used the discriminant value as a local state score for the HMM while Turtle Tamers used a quadratic-chi kernel metric for comparing pairs of frames in the training and test movie. As preprocessing, Pennect uses mean subtraction and compensates for body translations while Turtle Tamers replaces the missing values by the median of neighboring values. Both teams claim a linear complexity in number of frames, number of training examples, and image size. They both provided Matlab software that processes all the batches of the final test set on a regular laptop in a few hours.

The next best ranked participants (who won **third place in round 2**), the Joewan team, who published in this special topic (Wan et al., 2013), used a slightly different approach. They relied on the motion segmentation method provided by the organizers to pre-segment videos. They then represented each video as a bag of 3D MOSIFT features (integrating RGB and depth data), and then used a nearest neighbor classifier. Their algorithm is super-quadratic in image size, linear in number of frames per video, and linear in number of training examples. The method is rather slow and takes over a day to process all the batches of the final test set on a regular laptop.

The third best ranked team in round 1 (OneMillionMonkeys) also used an HMM in which a state is created for each frame of the gesture exemplars. This data representation is based on edge detection in each frame. Edges are associated with several attributes including the X/Y coordinates, their orientation, their sharpness, their depth and location in an area of change. To provide a local state score to the HMM for test frames, OneMillionMonkeys calculated the joint probability of all the nearest neighbors in training frames using a Gaussian model. The system works exclusively from the depth images. The system is one of the slowest proposed. Its processing speed is linear in number of training examples but quadratic in image size and number of frames per video. The method is rather slow and takes over a day to process all the batches of the final test set on a regular laptop.

Methods robust against translation include those of Joewan (Wan et al., 2013) and Immortals/Manavender (this is the same author under two different pseudonyms for round 1 and round 2). The team Immortals/Manavender published their method in this special topic (Malgireddy et al., 2013). Their representations are based on a bag of visual words, inspired by techniques used in action recognition (Laptev, 2005). Such representations are inherently shift invariant. The slight performance loss in translated data may be due to partial occlusions.

Although the team Zonga did not end up ranking among top ranking participants, the authors, who published their method in this special topic, proposed a very original method and ended up winning the best paper award. Notably, their outperformed all baseline methods early on in the challenge by applying their method without tuning it to the tasks of the challenge and remained at the top of the leaderboard for several weeks. The used a novel technique based on tensor geometry, which provides a data representation exhibiting desirable invariances and yields a very discriminating structure for action recognition.

ChaLearn also organized demonstration competitions of gesture recognition systems using Kinect^{TM}, in conjunction with those events. Novel data representations were proposed to tackle with success, in real time, the problem of hand and finger posture recognition. The demonstration competition winners showed systems capable of accurately tracking in real time hand postures in application for touch free exploration of 3D medical images for surgeons in the operating room, finger spelling (sign language for the deaf), virtual shopping, and game controlling. Combining the methods proposed in the demonstration competition tackling the problem of hand postures and those of the quantitative evaluation focusing on the dynamics of hand and arm movements is a promising direction of future research. For a long lasting impact, the challenge platform, the data and software repositories have been made available for further research[2].

3.2 ChaLearn Multimodal Gesture Recognition Challenge 2013

The focus of this second challenge was on *multiple instance, user independent learning of gestures from multimodal data*, which means learning to recognize gestures from several instances for each category performed by different users, drawn from a vocabulary of 20 gesture categories(Escalera et al., 2013b,a). A gesture vocabulary is a set of unique gestures, generally related to a particular task. In this challenge we focus on the recognition of a vocabulary of 20 Italian cultural/anthropological signs (Escalera et al., 2013b), see Figure 6 for one example of each Italian gesture.

3.2.1 2013 CHALLENGE DATA

In all the sequences, a single user is recorded in front of a Kinect^{TM}, performing natural communicative gestures and speaking in fluent Italian. The main characteristics of the dataset of gestures are:

- 13.858 gesture samples recorded with the Kinect^{TM} camera, including audio, skeletal model, user mask, RGB, and depth images.
- RGB video stream, 8-bit VGA resolution (640×480) with a Bayer color filter, and depth sensing video stream in VGA resolution (640×480) with 11-bit. Both are acquired in 20 fps on average.
- Audio data is captured using Kinect^{TM} 20 multiarray microphone.
- A total number of 27 users appear in the data set.

2. http://gesture.chalearn.org/

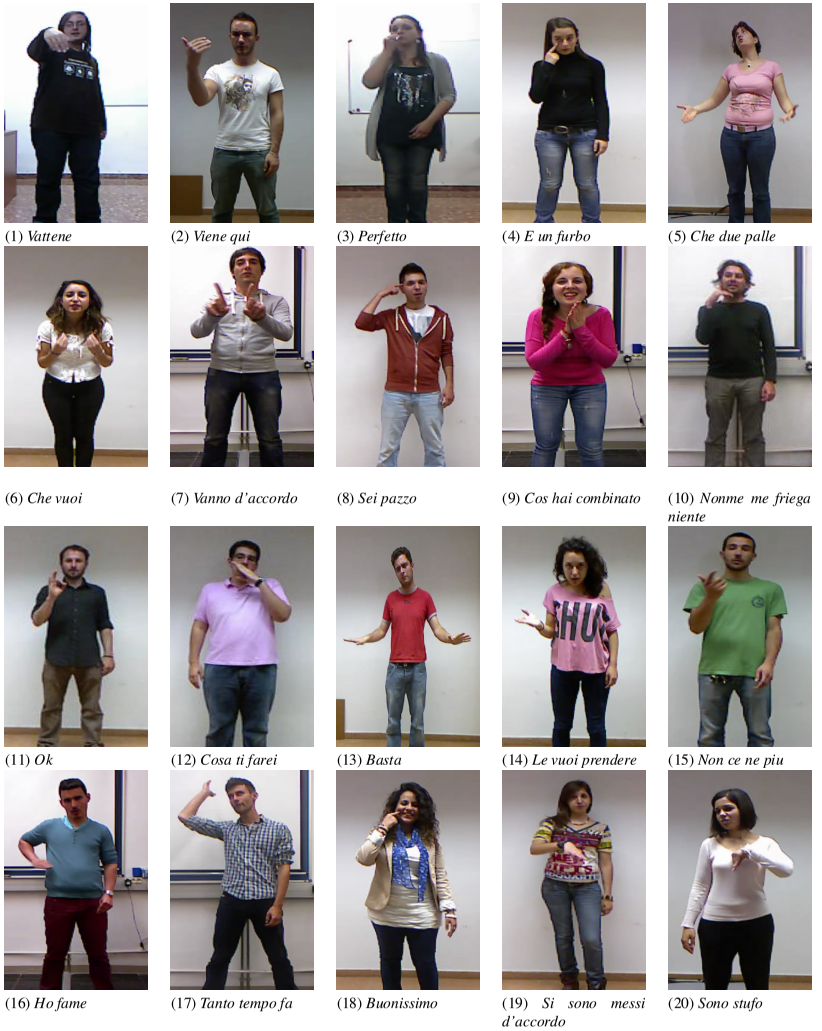


Figure 6: Data set gesture categories.

- The data set contains the following number of sequences, development: 393 (7.754 gestures), validation: 287 (3.362 gestures), and test: 276 (2.742 gestures), each sequence lasts between 1 and 2 minutes and contains between 8 and 20 gesture samples, around 1.800 frames. The total number

Table 5: Easy and challenging aspects of the data.

Easy
Fixed camera
Near frontal view acquisition
Within a sequence the same user
Gestures performed mostly by arms and hands
Camera framing upper body
Several available modalities: audio, skeletal model, user mask, depth, and RGB
Several instances of each gesture for training
Single person present in the visual field
Challenging
Within each sequence:
Continuous gestures without a resting pose
Many gesture instances are present
Distracter gestures out of the vocabulary may be present in terms of both gesture and audio
Between sequences:
High inter and intra-class variabilities of gestures in terms of both gesture and audio
Variations in background, clothing, skin color, lighting, temperature, resolution
Some parts of the body may be occluded
Different Italian dialects

of frames of the data set is 1.720.800.

• All the gesture samples belonging to 20 main gesture categories from an Italian gesture dictionary are annotated at frame level indicating the gesture label.

• 81% of the participants were Italian native speakers, while the remaining 19% of the users were not Italian, but Italian-speakers.

• All the audio that appears in the data is from the Italian dictionary. In addition, sequences may contain distractor words and gestures, which are not annotated since they do not belong to the main dictionary of 20 gestures.

This dataset, available at `http://sunai.uoc.edu/chalearn`, presents various features of interest as listed in Table 5. Examples of the provided visual modalities are shown in Figure 7.

RGB Depth User mask Skeletal model

Figure 7: Different data modalities of the provided data set.

3.2.2 2013 CHALLENGE PROTOCOL AND EVALUATION

As in our previous 2011-2012 challenge, it consisted of two main components: a development phase (April 30th to Aug 1st) and a final evaluation phase (Aug 2nd to Aug 15th). The submission and

evaluation of the challenge entries was via the *Kaggle* platform [3]. The official participation rules were provided on the website of the challenge. In addition, publicity and news on the ChaLearn Multimodal Gesture Recognition Challenge were published in well-known online platforms, such as LinkedIn, Facebook, Google Groups and the ChaLearn website.

During the development phase, the participants were asked to build a system capable of learning from several gesture samples a vocabulary of 20 Italian sign gesture categories. To that end, the teams received the development data to train and self-evaluate their systems. In order to monitor their progress they could use the validation data for which the labels were not provided. The prediction results on validation data could be submitted online to get immediate feed-back. A real-time leaderboard showed to the participants their current standing based on their validation set predictions.

During the final phase, labels for validation data were published and the participants performed similar tasks as those performed in previous phase, using the validation data and training data sets in order to train their system with more gesture instances. The participants had only few days to train their systems and upload them. The organizers used the final evaluation data in order to generate the predictions and obtain the final score and rank for each team. At the end, the final evaluation data was revealed, and authors submitted their own predictions and fact sheets to the platform.

As an evaluation metric we also used the Levenshtein distance described in previous section. A public score appeared on the leaderboard during the development period and was based on the validation data. Subsequently, a private score for each team was computed on the final evaluation data released at the end of the development period, which was not revealed until the challenge was over. The private score was used to rank the participants and determine the prizes.

3.2.3 2013 CHALLENGE RESULTS

The challenge attracted high level of participation, with a total of 54 teams and near 300 total number of entries. This is a good level of participation for a computer vision challenge requiring very specialized skills. Finally, 17 teams successfully submitted their prediction in final test set, while providing also their code for verification and summarizing their method by means of a fact sheet questionnaire.

After verifying the codes and results of the participants, the final scores of the top rank participants on both validation and test sets were made public: these results are shown in Table 6, where winner results on the final test set are printed in bold. In the end, the final error rate on the test data set was around 12%.

3.2.4 2013 CHALLENGE SUMMARY OF THE WINNER METHODS

Table 7 shows the summary of the strategies considered by each of the top ranked participants on the test set. Interestingly, the three top ranked participants agree in the modalities and segmentation strategy considered, although they differ in the final applied classifier. Next, we briefly describe in more detail the approach designed by the three winners of the challenge.

The first ranked team *IVAMM* on the test set used a feature vector based on audio and skeletal information, and applied late fusion to obtain final recognition results. A simple time-domain end-point detection algorithm based on joint coordinates is applied to segment continuous data sequences into candidate gesture intervals. A Gaussian Hidden Markov Model is trained with 39-

3. https://www.kaggle.com/c/multimodal-gesture-recognition

Table 6: Top rank results on validation and test sets.

TEAM	Validation score	Test score
IVA MM	0.20137	**0.12756**
WWEIGHT	0.46163	**0.15387**
ET	0.33611	**0.16813**
MmM	0.25996	0.17215
PPTK	0.15199	0.17325
LRS	0.18114	0.17727
MMDL	0.43992	0.24452
TELEPOINTS	0.48543	0.25841
CSI MM	0.32124	0.28911
SUMO	0.49137	0.31652
GURU	0.51844	0.37281
AURINKO	0.31529	0.63304
STEVENWUDI	1.43427	0.74415
JACKSPARROW	0.86050	0.79313
JOEWAN	0.13653	0.83772
MILAN KOVAC	0.87835	0.87463
IAMKHADER	0.93397	0.92069

Table 7: Team methods and results. Early and late refer to early and late fusion of features/classifier outputs. HMM: Hidden Markov Models. KNN: Nearest Neighbor. RF: Random Forest. Tree: Decision Trees. ADA: Adaboost variants. SVM: Support Vector Machines. Fisher: Fisher Linear Discriminant Analysis. GMM: Gaussian Mixture Models. NN: Neural Networks. DGM: Deep Boltzmann Machines. LR: Logistic Regression. DP: Dynamic Programming. ELM: Extreme Learning Machines.

TEAM	Test score	Rank	Modalities	Segmentation	Fusion	Classifier
IVA MM	0.12756	1	Audio,Skeleton	Audio	None	HMM,DP,KNN
WWEIGHT	0.15387	2	Audio,Skeleton	Audio	Late	RF,KNN
ET	0.16813	3	Audio,Skeleton	Audio	Late	Tree,RF,ADA
MmM	0.17215	4	Audio,RGB+Depth	Audio	Late	SVM,Fisher,GMM,KNN
PPTK	0.17325	5	Skeleton,RGB,Depth	Sliding windows	Late	GMM,HMM
LRS	0.17727	6	Audio,Skeleton,Depth	Sliding windows	Early	NN
MMDL	0.24452	7	Audio,Skeleton	Sliding windows	Late	DGM+LR
TELEPOINTS	0.25841	8	Audio,Skeleton,RGB	Audio,Skeleton	Late	HMM,SVM
CSI MM	0.28911	9	Audio,Skeleton	Audio	Early	HMM
SUMO	0.31652	10	Skeleton	Sliding windows	None	RF
GURU	0.37281	11	Audio,Skeleton,Depth	DP	Late	DP,RF,HMM
AURINKO	0.63304	12	Skeleton,RGB	Skeleton	Late	ELM
STEVENWUDI	0.74415	13	Audio,Skeleton	Sliding windows	Early	DNN,HMM
JACKSPARROW	0.79313	14	Skeleton	Sliding windows	None	NN
JOEWAN	0.83772	15	Skeleton	Sliding windows	None	KNN
MILAN KOVAC	0.87463	16	Skeleton	Sliding windows	None	NN
IAMKHADER	0.92069	17	Depth	Sliding windows	None	RF

dimension MFCC features and generates confidence scores for each gesture category. A Dynamic Time Warping based skeletal feature classifier is applied to provide complementary information. The confidence scores generated by the two classifiers are firstly normalized and then combined to produce a weighted sum. A single threshold approach is employed to classify meaningful gesture intervals from meaningless intervals caused by false detection of speech intervals.

The second ranked team *WWEIGHT* combined audio and skeletal information, using both joint spatial distribution and joint orientation. The method first searches for regions of time with high audio-energy to define 1.8-second-long windows of time that potentially contained a gesture. This had the effect that the development, validation, and test data were treated uniformly. Feature

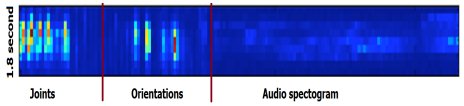


Figure 8: ExtraTreesClassifier Feature Importance.

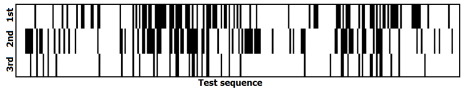


Figure 9: Recognition of test sequence by the three challenge winners. Black bin means that the complete list of ordered gestures has been successfully recognized.

vectors are then defined using a log-spaced audio spectrogram and the joint positions and orientations above the hips. At each time sample the method subtracts the average 3D position of the left and right shoulders from each 3D joint position. Data is down-sampled onto a 5 Hz grid considering 1.8 seconds. There were 1593 features total (9 time samples $\times$ 177 features per time sample). Since some of the detected windows can contain distractor gestures, an extra 21st label is introduced, defining the 'not in the dictionary' gesture category. Python's scikit-learn was used to train two models: an ensemble of randomized decision trees (ExtraTreesClassifier, 100 trees, 40% of features) and a K-Nearest Neighbor model (7 neighbors, L1 distance). The posteriors from these models are averaged with equal weight. Finally, a heuristic is used (12 gestures maximum, no repeats) to convert posteriors to a prediction for the sequence of gestures.

Figure 8 shows the mean feature importance for the windows size of 1.8 seconds for the three sets of features: joint coordinates, joint orientations, and audio spectogram. One can note that features from the three sets are selected as discriminative by the classifier, although skeletal features becomes more useful for the ExtraTreesClassifier. Additionally, the most discriminative features are those in the middle of the windows size, since begin-end features are shared among different gestures (transitions) and thus are less discriminative for the classifier.

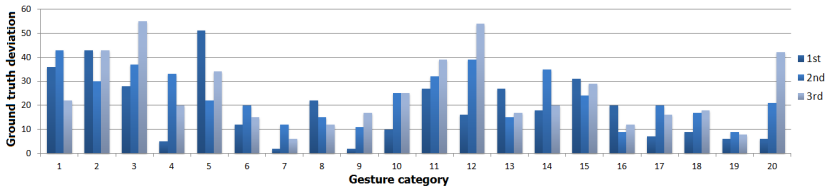


Figure 10: Deviation of the number of gesture samples for each category by the three winners in relation to the GT data.

The third ranked team *ET* combined the output decisions of two designed approaches. In the first approach, they look for gesture intervals (unsupervised) using the audio files and extract these features from intervals (MFCC). Using these features, authors train a random forest and gradient boosting classifier. The second approch uses simple statistics (median, var, min, max) on the first 40

frames for each gesture to build the training samples. The prediction phase uses a sliding window. The authors create a weighted average of the output of these two models. The features considered were skeleton information and audio signal.

Finally, we extracted some statistics from the results of the three challenge winners in order to analyze common points and difficult aspects of the challenge. Figure 9 shows the recognition of the 276 test sequences by the winners. Black bin means that the complete list of ordered gestures was successfully recognized for those sequences. Once can see that there exists some kind of correlation among methods. Taking into account that consecutive sequences belong to the same user performing gestures, it means that some some gestures are easier to recognize than others. Since different users appear in the training and test sequences, it is sometimes difficult for the models to generalize to the style of new users, based on the gesture instances used for training.

We also investigated the difficulty of the problem by gesture category, within each of the 20 Italian gesture categories. Figure 10 shows for each winner method the deviation between the number of gesture instances recognized and the total number of gestures, for each category. This was computed for each sequence independently, and adding the deviation for all the sequences. In that case, a zero value means that the participant method recognized the same number of gesture instances for a category that was recorded in the ground truth data. Although we cannot guarantee with this measure that the order of recognized gesture matches with the ground truth, it gives us an idea of how difficult the gesture sequences were to segment into individual gestures. Additionally, the sum of total deviation for all the gestures for all the teams was 378, 469, and 504, which correlates with the final rank of the winners. The figure suggests a correlation between the performance of the three winners. In particular, categories 6, 7, 8, 9, 16, 17, 18, and 19 were the ones that achieved most accuracy for all the participants, meanwhile 1, 2, 3, 5, and 12 were the ones that introduced the highest recognition error. Note that the public data set provides accurate label annotations of end-begin of gestures, and thus, a more detailed recognition analysis could be performed applying a different recognition measurement to Leveinstein, such as Jaccard overlapping or sensitivity score estimation, which will also allow for confusion matrix estimation based on both inter and intra user and gesture category variability. This is left to future work.

3.3 ChaLearn Multimodal Gesture Spotting Challenge 2014

In ChaLearn LAP 2014 (Escalera et al., 2014) we focused on the user-independent automatic spotting of a vocabulary of 20 Italian cultural/anthropological signs in image sequences, see Figure 6.

3.3.1 2014 GESTURE CHALLENGE DATA

This challenge was based on an Italian gesture data set, called *Montalbano gesture dataset*, an enhanced version of the ChaLearn 2013 multimodal gesture recognition challenge (Escalera et al., 2013b,a) with more ground-truth annotations. In all the sequences, a single user is recorded in front of a Kinect^{TM}, performing natural communicative gestures and speaking in fluent Italian. Examples of the different visual modalities are shown in Figure 7.

The main characteristics of the data set are:

• Largest data set in the literature, with a large duration of each individual performance showing no resting poses and self-occlusions.

• There is no information about the number of gestures to spot within each sequence, and several distracter gestures (out of the vocabulary) are present.

Training seq.	Validation seq.	Test seq.	Sequence duration	FPS
393 (7,754 gestures)	287 (3,362 gestures)	276 (2,742 gestures)	1-2 min	20
Modalities	Num. of users	Gesture categories	Labeled sequences	Labeled frames
RGB, Depth, User mask, Skeleton	27	20	13,858	1,720,800

Table 8: Main characteristics of the *Montalbano* gesture dataset.

• High intra-class variability of gesture samples and low inter-class variability for some gesture categories.

A list of data attributes for data set used in track 3 is described in Table 8.

3.3.2 2014 GESTURE CHALLENGE PROTOCOL AND EVALUATION

The challenge was managed using the Microsoft Codalab platform[4]. We followed a development (February 9 to May 20 2014) and tests phases (May 20th - June 1st 2014)) as in our previous challenges.

To evaluate the accuracy of action/interaction recognition, we use the Jaccard Index, For the n action, interaction, and gesture categories labelled for a RGB/RGBD sequence s, the Jaccard Index is defined as:

$$J_{s,n} = \frac{A_{s,n} \bigcap B_{s,n}}{A_{s,n} \bigcup B_{s,n}}, \quad (1)$$

where $A_{s,n}$ is the ground truth of action/interaction/gesture n at sequence s, and $B_{s,n}$ is the prediction for such an action at sequence s. $A_{s,n}$ and $B_{s,n}$ are binary vectors where 1-values correspond to frames in which the $n-$th action is being performed. The participants were evaluated based on the mean Jaccard Index among all categories for all sequences, where motion categories are independent but not mutually exclusive (in a certain frame more than one action, interaction, gesture class can be active).

In the case of false positives (e.g. inferring an action, interaction or gesture not labelled in the ground truth), the Jaccard Index is 0 for that particular prediction, and it will not count in the mean Jaccard Index computation. In other words n is equal to the intersection of action/interaction/gesture categories appearing in the ground truth and in the predictions.

An example of the calculation for two actions is shown in Figure 11. Note that in the case of recognition, the ground truth annotations of different categories can overlap (appear at the same time within the sequence). Also, although different actors appear within the sequence at the same time, actions/interactions/gestures are labelled in the corresponding periods of time (that may overlap), there is no need to identify the actors in the scene. The example in Figure 11 shows the mean Jaccard Index calculation for different instances of actions categories in a sequence (single red lines denote ground truth annotations and double red lines denote predictions). In the top part of the image one can see the ground truth annotations for actions walk and fight at sequence s. In the center part of the image a prediction is evaluated obtaining a Jaccard Index of 0.72. In the bottom part of the image the same procedure is performed with the action fight and the obtained Jaccard Index is 0.46. Finally, the mean Jaccard Index is computed obtaining a value of 0.59.

4. https://www.codalab.org/competitions/

Figure 11: Example of mean Jacquard Index calculation for gesture and action/interaction spotting.

3.3.3 2014 GESTURE CHALLENGE RESULTS

Table 10 summarizes the methods of the 17 participants that contributed to the test set of track 3. Although DTW and HMM (and variants) were in the last edition of the ChaLearn Multimodal Gesture competition (Escalera et al., 2013b,a), random forest has been widely applied in this 2014 edition. Also, three participants used deep learning architectures.

Team name	Accuracy	Rank	Features	Dimension reduction	Clustering	Classifier	Temporal coherence	Gesture representation
CUHK-SWJTU	**0.850173**	1	Improved trajectories (Wang and Schmid, 2013)	PCA	-	SVM	Sliding windows	Fisher Vector
ADSC	**0.850164**	2	Improved trajectories (Wang and Schmid, 2013)	-	-	SVM	Sliding windows	-
SBUVIS	**0.841405**	3	Improved trajectories (Wang and Schmid, 2013)	-	-	SVM	Sliding windows	-
DonkeyBurger	0.342192	4	MHI, STIP	-	Kmeans	Sparse code	Sliding windows	-
UC-T2	0.121565	5	Improved trajectories (Wang and Schmid, 2013)	PCA	-	Kmeans	Sliding windows	Fisher Vector
MindLAB	0.008383	6	MBF	-	Kmeans	RF	Sliding windows	BoW

MMLAB	1	IDT	PCA	-	SVM	-	Fisher Vector
FIKIE	**0.5239**	2	IDT	PCA	-	HMM	Appearance+Kalman filter	-

Table 9: Top rows: Action/interaction 2014 recognition results. MHI: Motion History Image; STIP: Spatio-Temporal interest points; MBF: Multiscale Blob Features; BoW: Bag of Visual Words; RF: Random Forest. Bottom two rows: Action/interaction 2015 recognition results. IDT: Improved Dense Trajectories (Wang and Schmid, 2013).

Team	Accuracy	Rank	Modalities	Features	Fusion	Temp. segmentation	Dimension reduction	Gesture representation	Classifier
LIRIS	**0.849987**	1	SK, Depth, RGB	RAW, SK joints	Early	Joints motion	-	-	DNN
CraSPN	**0.833904**	2	SK, Depth, RGB	HOG, SK	Early	Sliding windows	-	BoW	Adaboost
LPigou	0.788804	5	Depth, RGB	RAW, SK joints	Early	Sliding windows	Max-pooling CNN	-	CNN
stevenwudi	0.787310	6	SK, depth	RAW	Late	Sliding windows	PCA	Fisher Vector, VLAD	SVM
lsmsr	0.746632	7	SK	SK	-	Sliding windows	PCA	-	MRF; KNN
Quads	0.745449	8	SK	SK quads	-	Sliding windows	-	-	RF
Telepoints	0.688778	9	SK, Depth, RGB	STIPS, SK	Late	Joints motion	-	-	SVM
TUM-fortiss	0.648979	10	SK, Depth, RGB	STIPS	Late	Joints motion	-	-	RF; SVM
CSU-SCM	0.397177	11	Skeleton, Depth, mask	HOG, Skeleton	Late	Sliding windows	-	2DMTM	SVM, HMM
iva.mm	0.355251	12	Skeleton, RGB, depth	Skeleton, HOG	Late	Sliding windows	-	BoW	SVM, HMM
Terrier	0.339025	13	Skeleton	Skeleton	-	Sliding windows	-	-	RF
Team Netherlands	0.430709	14	Skeleton, Depth, RGB	MHI	Early	DTW	Preserving projections	-	SVM, RT
VecsRel	0.408012	15	Skeleton, Depth, RGB	RAW, skeleton joints	Late	DTW	-	-	DNN
Samgest	0.391613	16	Skeleton, Depth, RGB, mask	Skeleton, blobs, moments	Late	Sliding windows	-	-	HMM
YNL	0.270600	17	Skeleton	Skeleton	-	Sliding windows	-	Fisher Vector	HMM, SVM

Table 10: Multimodal gesture recognition results. SK: Skeleton; DNN: Deep Neural Network; RF: Ranfom Forest; 2DMTM: 2D motion trail model; RT: Regression Tree.

3.3.4 2014 GESTURE CHALLENGE SUMMARY OF THE WINNER METHODS

Next, we describe the main characteristics of the three winning methods.

First place: The proposed method was based on a deep learning architecture that iteratively learned and integrated discriminative data representations from individual channels, modelling cross-modality correlations and short- and long-term temporal dependencies. This framework combined three data modalities: depth information, grayscale video and skeleton stream ("articulated pose"). Articulated pose served as an efficient representation of large-scale body motion of the upper body and arms, while depth and video streams contained complementary information about more subtle hand articulation. The articulated pose was formulated as a set of joint angles and normalized distances between upper-body joints, augmented with additional information reflecting speed and acceleration of each joint. For the depth and video streams, the authors did not rely on hand-crafted descriptors, but on discriminatively learning joint depth-intensity data representations with a set of convolutional neural layers. Iterative fusion of data channels was performed at output layers of the neural architecture. The idea of learning at multiple scales was also applied to the temporal dimension, such that a gesture was considered as an ordered set of characteristic motion impulses, or dynamic poses. Additional skeleton-based binary classifier was applied for accurate gesture localization. Fusing multiple modalities at several spatial and temporal scales led to a significant increase in recognition rates, allowing the model to compensate for errors of the individual classifiers as well as noise in the separate channels.

Second place: The approach combined a sliding-window gesture detector with multimodal features drawn from skeleton data, color imagery, and depth data produced by a first-generation Kinect^{TM} sensor. The gesture detector consisted of a set of boosted classifiers, each tuned to a specific gesture or gesture mode. Each classifier was trained independently on labeled training data, employing bootstrapping to collect hard examples. At run-time, the gesture classifiers were evaluated in a one-vs-all manner across a sliding window. Features were extracted at multiple temporal scales to enable recognition of variable-length gestures. Extracted features included descriptive statistics of normalized skeleton joint positions, rotations, and velocities, as well as HOG descriptors of the hands. The full set of gesture detectors was trained in under two hours on a single machine, and was extremely efficient at runtime, operating at 1700 fps using skeletal data.

Third place: The proposed method was based on four features: skeletal joint position feature, skeletal joint distance feature, and histogram of oriented gradients (HOG) features corresponding to left and right hands. Under the naïve Bayes assumption, likelihood functions were independently defined for every feature. Such likelihood functions were non-parametrically constructed from the training data by using kernel density estimation (KDE). For computational efficiency, k-nearest neighbor (kNN) approximation to the exact density estimator was proposed. Constructed likelihood functions were combined to the multimodal likelihood and this serves as a unary term for our pairwise Markov random field (MRF) model. For enhancing temporal coherence, a pairwise term was additionally incorporated to the MRF model. Final gesture labels were obtained via 1D MRF inference efficiently achieved by dynamic programming.

3.4 ChaLearn Action and Interaction Spotting Challenge 2014

The goal of this challenge was to perform automatic action and interaction spotting of people appearing in RGB data sequences.

Training actions	Validation actions	Test actions	Sequence duration	FPS
150	90	95	9× 1-2 min	15
Modalities	**Num. of users**	**Action categories**	**interaction categories**	**Labelled sequences**
RGB	14	7	4	235

Table 11: Action and interaction data characteristics.

3.4.1 2014 ACTION CHALLENGE DATA

We presented a novel fully limb labelled dataset, the Human Pose Recovery and Behavior Analysis *HuPBA* 8k+ dataset (Sánchez et al., 2014). This dataset is formed by more than 8000 frames where 14 limbs are labelled at pixel precision, thus providing 124,761 annotated human limbs. The characteristics of the data set are:

• The images are obtained from 9 videos (RGB sequences) and a total of 14 different actors appear in the sequences. The image sequences have been recorded using a stationary camera with the same static background.

• Each video (RGB sequence) was recorded at 15 fps rate, and each RGB image was stored with resolution 480×360 in BMP file format.

• For each actor present in an image 14 limbs (if not occluded) were manually tagged: Head, Torso, R-L Upper-arm, R-L Lower-arm, R-L Hand, R-L Upper-leg, R-L Lower-leg, and R-L Foot.

• Limbs are manually labelled using binary masks and the minimum bounding box containing each subject is defined.

• The actors appear in a wide range of different poses and performing different actions/gestures which vary the visual appearance of human limbs. So there is a large variability of human poses, self-occlusions and many variations in clothing and skin color.

• Several actions and interactions categories are labelled at frame level.

A key frame example for each gesture/action category is shown in Figure 12. The challenges the participants had to deal with for this new competition are:

• 235 action/interaction samples performed by 14 actors.

• Large difference in length about the performed actions and interactions. Several distracter actions out of the 11 categories are also present.

• 11 action categories, containing isolated and collaborative actions: Wave, Point, Clap, Crouch, Jump, Walk, Run, Shake Hands, Hug, Kiss, Fight. There is a high intra-class variability among action samples.

Table 11 summarizes the data set attributes for the case of action/interaction spotting.

3.4.2 2014 ACTION CHALLENGE PROTOCOL AND EVALUATION

To evaluate the accuracy of action/interaction recognition, we use the Jaccard Index as defined in Section 3.3.2.

3.4.3 2014 ACTION CHALLENGE RESULTS

In this section we summarize the methods proposed by the participants and the winning methods. Six teams submitted their code and predictions for the test sets. Top rows of Table 9 summarizes the approaches of the participants who uploaded their models. One can see that most methods are based on similar approaches. In particular, alternative representations to classical BoW were considered, as Fisher Vector and VLAD (Jegou et al., 2012). Most methods perform sliding windows and SVM

(a) Wave (b) Point (c) Clap

(d) Crouch (e) Jump (f) Walk

(g) Run (h) Shake hands (i) Hug

(j) Kiss (k) Fight (l) Idle

Figure 12: Key frames of the *HuPBA* $8K+$ dataset used in the tracks 1 and 2, showing actions ((a) to (g)), interactions ((h) to (k)) and the idle pose (l).

classification. In addition, to refine the tracking of interest points, 4 participants used improved trajectories (Wang and Schmid, 2013).

3.4.4 2014 ACTION CHALLENGE SUMMARY OF THE WINNER METHODS

Next, we describe the main characteristics of the three winning methods.

First place: The method was composed of two parts: video representation and temporal segmentation. For the representation of video clip, the authors first extracted improved dense trajectories with HOG, HOF, MBHx, and MBHy descriptors. Then, for each kind of descriptor, the participants trained a GMM and used Fisher vector to transform these descriptors into a high dimensional super vector space. Finally, sum pooling was used to aggregate these codes in the whole video clip and normalize them with power L2 norm. For the temporal recognition, the authors resorted to a temporal sliding method along the time dimension. To speed up the processing of detection, the authors designed a temporal integration histogram of Fisher Vector, with which the pooled Fisher Vector was efficiently evaluated at any temporal window. For each sliding window, the authors used the pooled Fisher Vector as representation and fed it into the SVM classifier for action recognition.

Second place: a human action detection framework called "mixture of heterogeneous attribute analyzer" was proposed. This framework integrated heterogeneous attributes learned from various types of video features including static and dynamic, local and global features, to boost the action detection accuracy. The authors first detected a human from the input video by SVM-HOG detector and performed forward-backward tracking. Multiple local human tracks are linked into long trajectories by spatial-temporal graph based matching. Human key poses and local dense motion trajectories were then extracted within the tracked human bounding box sequences. Second, the authors proposed a mining method that learned discriminative attributes from three feature modalities: human trajectory, key pose and local motion trajectory features. The mining framework was based on the exemplar-SVM discriminative middle level feature detection approach. The learned discriminative attributes from the three types of visual features were then mixed in a max-margin learning algorithm which also explores the combined discriminative capability of heterogeneous feature modalities. The learned mixed analyzer was then applied to the input video sequence for action detection.

Third place: The framework for detecting actions in video is based on improved dense trajectories applied on a sliding windows fashion. Authors independently trained 11 one-versus-all kernel SVMs on the labelled training set for 11 different actions. The feature and feature descriptions used are improved dense trajectories, HOG, HOF, MBHx and MBHy. During training, for each action, a temporal sliding window is applied without overlapping. For every action, a segment was labelled 0 (negative) for a certain action only if there is no frame in this segment labelled 1. The feature coding method was bag-of-features. For a certain action, the features associated with those frames which are labelled 0 (negative) are not counted when we code the features of the action for the positive segments with bag-of-features. On the basis of the labelled segments and their features, a kernel SVM was trained for each action. During testing, non-overlap sliding window was applied for feature coding of the video. Every frame in a segment was consistently labelled as the output of SVM for each action. The kernel type, sliding window size and penalty of SVMs were selected during validation. When building the bag-of-features, the clustering method was K-means and the vocabulary size is 4000. For one trajectory feature in one frame, all the descriptors were connected to form one description vector. The bag-of-features were built upon this vector.

3.5 ChaLearn Action and Interaction Spotting Challenge 2015

The goal of this challenge was to perform automatic action and interaction spotting of people appearing in RGB data sequences. This corresponds to the second round of 2014 Action/Interaction challenge (Baró et al., 2015). Data, protocol, and evaluation were defined as explained in Section 3.4.

3.5.1 2015 ACTION CHALLENGE RESULTS

Results of the two top ranked participants are shown in bottom rows of Table 9. One can see that the methods of the participants are similar to the ones applied in the 2014 challenge for the same dataset (Top rows of Table 9). Results of this second competition round improved by 2% the results obtained in the first round of the challenge.

3.5.2 2015 ACTION CHALLENGE SUMMARY OF THE WINNER METHODS

First winner: This method is an improvement of the system proposed in (Peng et al., 2015), which is composed of two parts: video representation and temporal segmentation. For the representation of video clip, the authors first extracted improved dense trajectories with HOG, HOF, MBHx, and MBHy descriptors. Then, for each kind of descriptor, the participants trained a GMM and used Fisher vector to transform these descriptors into a high dimensional super vector space. Finally, sum pooling was used to aggregate these codes in the whole video clip and normalize them with power L2 norm. For the temporal recognition, the authors resorted to a temporal sliding method along the time dimension. To speed up the processing of detection, the authors designed a temporal integration histogram of Fisher Vector, with which the pooled Fisher Vector was efficiently evaluated at any temporal window. For each sliding window, the authors used the pooled Fisher Vector as representation and fed it into the SVM classifier for action recognition. A summary of this method is shown in Figure 13.

Second winner: The method implements an end-to-end generative approach from feature modelling to activity recognition. The system combines dense trajectories and Fisher Vectors with a temporally structured model for action recognition based on a simple grammar over action units. The authors modify the original dense trajectory implementation of (Wang and Schmid, 2013) to avoid the omission of neighborhood interest points once a trajectory is used (the improvement is shown in Figure 14). They use an open source speech recognition engine for the parsing and segmentation of video sequences. Because a large data corpus is typically needed for training such systems, images were mirrored to artificially generate more training data. The final result is achieved by voting over the output of various parameter and grammar configurations.

3.6 Other international competitions for gesture and action recognition

In addition to the series of ChaLearn Looking at People challenges, different international challenges have also been performed in the field of action/gesture recognition. Some of them are reviewed below.

The ChAirGest challenge (Ruffieux et al., 2013) is a research oriented competition designed to compare multimodal gesture recognizers. The provided data came from one Kinect camera and 4 Inertial Motion Units (IMU) attached to the right arm and neck of the subject. The dataset contains 10 different gestures, started from 3 different resting postures and recorded in two different lighting

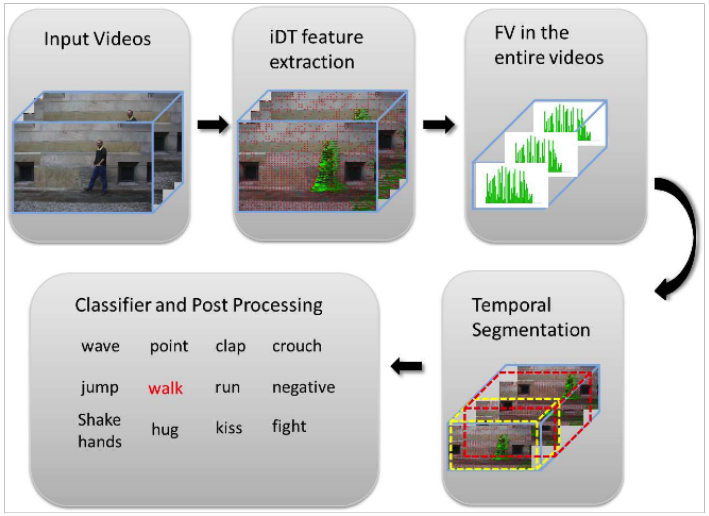


Figure 13: Method summary for MMLAB team (Wang et al., 2015b).

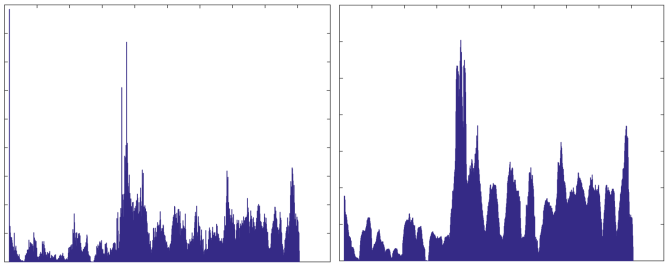


Figure 14: Example of DT feature distribution for the first 200 frames of Seq01 for FKIE team, (a) shows the distribution of the original implementation, (b) shows the distribution of the modified version.

conditions by 10 different subjects. Thus, the total dataset contains 1200 annotated gestures split in continuous video sequences containing a variable number of gestures. The goal of the challenge was to promote research on methods using multimodal data to spot and recognize gestures in the context of close human-computer interaction.

In 2015, OpenCV ran at CVPR different challenges[5], one of them focusing on gesture recognition, using the ChaLearn gesture recognition data. The winner of the competition was also the work that won the ChaLearn challenge at ECCV 2014 (Neverova et al., 2014b).

Another recent action recognition competition is the THUMOS challenge (Gorban et al., 2015). Last round of the competition was ran at CVPR 2015. The last round of the challenge contained a forward-looking data set with over 430 hours of video data and 45 million frames (70% larger than THUMOS'14). All videos were collected from YouTube. Two tracks were performed: 1) Action Classification, for whole-clip action classification on 101 action classes, and 2) Temporal Action Localization, for action recognition and temporal localization on a subset of 20 action classes.

4. Summary of Special Topic Papers Not Related to the Challenges

In this special topic, in addition to the papers that described systems that participated in the ChaLearn gesture challenges, there are also several papers relating to broader aspects of gesture recognition, including topics such as sign language recognition, facial expression analysis, and facilitating development of real-world systems.

Three of these papers propose new methods for gesture and action recognition (Malgireddy et al., 2013; Fanello et al., 2013; Lui, 2012), that were also evaluated on parts of the CDG2011 dataset (Guyon et al., 2014), used in the ChaLearn challenges held in 2011 and 2012. More specifically, (Malgireddy et al., 2013) present an approach for detecting and recognizing activities and gestures. Hierarchical models are built to describe each activity as a combination of other, more simple activities. Each video is recursively divided into segments consisting of a fixed number of frames. The relationship between observed features and latent variables is modelled using a generative model that combines aspects of dynamic Bayesian networks and hierarchical Bayesian models. (Fanello et al., 2013) describe a system for real-time action recognition using depth video. The paper proposes specific features that are well adapted to real-time constraints, based on histograms of oriented gradients and histograms of optical flow. Support vector machines trained on top of such features perform action segmentation and recognition. (Lui, 2012) propose a method for representing gestures as tensors. These tensors are treated as points on a product manifold. In particular, the product manifold is factorized into three manifolds, one capturing horizontal motion, one capturing vertical motion, and one capturing 2D appearance. The difference between gestures is measured using a geodesic distance on the product manifold.

One paper (Wang et al., 2012) studied an action recognition problem outside the scope of the ChaLearn contests, namely recognizing poses and actions from a single image. Hierarchical models are used for modelling body pose. Each model in this hierarchy covers a part of the human body that can range from the entire body to a specific rigid part. Different levels in this hierarchy correspond to different degrees of coarseness vs. detail in the models at each level.

Three papers proposed novel methods within the area of sign language recognition (Cooper et al., 2012; Nayak et al., 2012; Roussos et al., 2013). (Cooper et al., 2012) describe a method for sign language recognition using linguistic subunits that are learned automatically by the system. Different types of such subunits are considered, including subunits based on appearance and local motion of the hand, subunits based on combining tracked 2D hand trajectories and hand shape, and subunits based on tracked 3D hand trajectories. (Nayak et al., 2012) address the problem of learning a model for a sign that occurs multiple times in a set of sentences. One benefit from such an approach

5. http://code.opencv.org/projects/opencv/wiki/VisionChallenge

is that it does not require the start and end frame of each sign as training data. Another benefit is that the method identifies the aspects of a sign that are least affected by movement epenthesis, i.e., by signs immediately preceding or following the sign in question. (Roussos et al., 2013) present a method for classifying handshapes for the purpose of sign language recognition. Cropped hand images are converted to a normalized representation called "shape-appearance images", based on a PCA decomposition of skin pixel colors. Then, active appearance models are used to model the variation in shape and appearance of the hand.

One paper focused on the topic of facial expression analysis (Martinez and Du, 2012). The paper proposes a model for describing how humans perceive facial expressions of emotion. The proposed model consists of multiple distinct continuous spaces. Emotions can be represented using linear combinations of these separate spaces. The paper also discusses how the proposed model can be used to design algorithms for facial expression recognition.

Another set of papers contributed methods that address different practical problems, that are important for building real-world gesture interfaces (Nguyen-Dinh et al., 2014; Gillian and Paradiso, 2014; Kohlsdorf and Starner, 2013). One such problem is obtaining manual annotations and ground truth for large amounts of training data. Obtaining such manual annotations can be time consuming, and can be an important bottleneck in building a real system. Crowdsourcing is a potential solution, but crowdsourced annotations often suffer from noise, in the form of discrepancies in how different humans annotate the same data. In (Nguyen-Dinh et al., 2014), two template-matching methods are proposed, called SegmentedLCSS and WarpingLCSS, that explicitly deal with the noise present in crowdsourced annotations of gestures. These methods are designed for spotting gestures using wearable motion sensors. The methods quantize signals into strings of characters and then apply variations of the longest common subsequence algorithm (LCSS) to spot gestures.

In designing a real-world system, another important problem is rapid development. (Gillian and Paradiso, 2014) present a gesture recognition toolkit, a cross-platform open-source C++ library. The toolkit features a broad range of classification and regression algorithms and has extensive support for building real-time systems. This includes algorithms for signal processing, feature extraction and automatic gesture spotting.

Finally, choosing the gesture vocabulary can be an important implementation parameter. (Kohlsdorf and Starner, 2013) propose a method for choosing a vocabulary of gestures for a human-computer interface, so that gestures in that vocabulary have a low probability of being confused with each other. Candidate gestures for the interface can be suggested both by humans and by the system itself. The system compares examples of each gesture with a large repository of unlabelled sensor/motion data, to check how often such examples resemble typical session/motion patterns encountered in that specific application domain.

5. Summary of Special Topic Papers Related to 2011-2012 Challenges

Next we briefly review the contributions of the accepted papers for the special topic whose methods are applied on the data provided by 2011-2012 ChaLearn gesture recognition challenges. Interestingly, none of the methods proposed in these papers uses skeletal tracking. Instead, these methods use different features based on appearance and/or motion. Where the papers differ is in their choice of specific features, and also in the choice of gesture models that are built on top of the selected features.

(Konecny and Hagara, 2014) combine appearance (Histograms of Oriented Gradients) and motion descriptors (Histogram of Optical Flow) from RGB and depth images for parallel temporal segmentation and recognition. The Quadratic-Chi distance family is used to measure differences between histograms to capture cross-bin relationships. Authors also propose trimming videos by removing unimportant frames based on motion information. Finally, proposed descriptors with different Dynamic Time Warping variants are applied for final recognition.

In contrast to (Konecny and Hagara, 2014), which employes commonly used features, (Wan et al., 2013) proposes a new multimodal descriptor, as well as a new sparse coding method. The multimodal descriptor is called 3D EMoSIFT, is invariant to scale and rotation, and has more compact and richer visual representations than other state-of-the-art descriptors. The proposed sparse coding method is named simulation orthogonal matching pursuit (SOMP), and is a variant of BoW. Using SOMP, each feature can be represented by some linear combination of a small number of codewords.

A different approach is taken in (Jiang et al., 2015), which combines three different methods for classifying gestures. The first method uses an improved principal motion representation. In the second method, a particle-based descriptor and a weighted dynamic time warping are proposed for the location component classification. In the third method, the shape of the human subject is used, extracted from the frame in the gesture that exhibits the least motion. The explicit use of shape in this paper is in contrast to (Konecny and Hagara, 2014; Wan et al., 2013), where shape information is implicitly coded in the extracted features.

In (Goussies et al., 2014), the focus is on transfer learning. The proposed method did not do as well as the previous methods (Konecny and Hagara, 2014; Wan et al., 2013; Jiang et al., 2015) on the CDG 2011 dataset, but nonetheless it contributes novel ideas for transfer learning, that can be useful when the number of training examples per class is limited. This is in contrast to the other papers related to the 2011-2012 challenges, that did not address transfer learning. The paper introduces two mechanisms into the decision forest framework, in order to transfer knowledge from the source tasks to a given target task. The first one is mixed information gain, which is a data-based regularizer. The second one is label propagation, which infers the manifold structure of the feature space. The proposed approach show improvements over traditional decision forests in the ChaLearn Gesture Challenge and on the MNIST dataset.

6. Summary of Special Issue Papers Related to 2013 Challenge

Next we briefly review the contributions of the accepted papers for the special issue whose methods are applied on the data provided by 2013 ChaLearn multimodal gesture recognition challenge. An important difference between this challenge and the previous ChaLearn challenges is the multimodal nature of the data. Thus, a key focus area for methods applied on this data is the problem of fusing information from multiple modalities.

(Wu and Cheng, 2014) propose a Bayesian Co-Boosting framework for multimodal gesture recognition. Inspired by boosting learning and co-training method, the system combines multiple collaboratively trained weak classifiers, Hidden Markov Models in this case, to construct the final strong classifier. During each iteration round, randomly a number of feature subsets are samples and weak classifiers parameters for each subset are estimated. The optimal weak classifier and its corresponding feature subset are retained for strong classifier construction. Authors also define an upper bound of training error and derive the update rule of instance's weight, which guarantees the

error upper bound to be minimized through iterations. This methodology won the ChaLearn 2013 ICMI competition.

(Pitsikalis et al., 2014) present a framework for multimodal gesture recognition that is based on a multiple hypotheses rescoring fusion scheme. Authors employ multiple modalities, i.e., visual cues, such as skeleton data, color and depth images, as well as audio, and extract feature descriptors of the hands movement, handshape, and audio spectral properties. Using a common hidden Markov model framework authors build single-stream gesture models based on which they can generate multiple single stream-based hypotheses for an unknown gesture sequence. By multimodally rescoring these hypotheses via constrained decoding and a weighted combination scheme, authors end up with a multimodally-selected best hypothesis. This is further refined by means of parallel fusion of the monomodal gesture models applied at a segmental level. The proposed methodology is tested on the ChaLearn 2013 ICMI competition data.

7. Discussion

We reviewed the gesture recognition topic, defining a taxonomy to characterize state of the art works on gesture recognition. We also reviewed the gesture and action recognition challenges organized by ChaLearn from 2011 to 2015, as well as other international competitions related to gesture recognition. Finally, we reviewed the papers submitted to the Special Topic on Gesture Recognition 2011-2014 we organized at Journal of Machine Learning Research.

Regarding the ChaLearn gesture recognition challenges, we began right at the start of the Kinect^{TM} revolution when inexpensive infrared cameras providing image depth recordings became available. We published papers using this technology and other more conventional methods, including regular video cameras, to record data, thus providing a good overview of uses of machine learning and computer vision using multimodal data in this area of application. Notably, we organized a series of challenges and made available several datasets we recorded for that purpose, including tens of thousands of videos, which are available to conduct further research [6].

Regarding the papers published in the gesture recognition special topic related to 2011-2012 challenges with the objective of performing one-shot learning, most of the authors proposed new multimodal descriptors taking benefit from both RGB and Depth cues in order to describe human body features, both static and dynamic ones. As the recognition strategies, common techniques used were variants of classical well-known Dynamic Time Warping and Hidden Markov Models. In particular, the most efficient techniques so far have used sequences of features processed by graphical models of the HMM/CRF family, similar to techniques used in speech recognition. Authors also considered a gesture candidate sliding window and motion-based video-cutting approaches. Last approach was frequently used since sign language videos included a resting pose. Also interesting novel classification strategies were proposed, such as multilayered decomposition, where different length gesture units are recognize at different levels ((Jiang et al., 2015)).

Regarding the papers published in the gesture recognition special topic related to 2013 and 2014 challenges with the objective of performing user independent multiple gesture recognition from large volumes of multimodal data (RGB, Depth and audio), different classifiers for gesture recognition were used by the participants. In 2013, the preferred one was Hidden Markov Models (used by the first ranked team of the challenge), followed by Random Forest (used by the second and third winners). Although several state of the art learning and testing gesture techniques were

6. http://gesture.chalearn.org/

applied at the last stage of the methods of the participants, still the feature vector descriptions are mainly based on MFCC audio features and skeleton joint information. The published paper of the winner to the special topic presents a novel coboosting strategy, where a set of HMM classifiers and collaboratively included in a boosting strategy considering random sets of features ((Wu and Cheng, 2014)). In 2014, similar descriptors and classifiers were used, and in particular, three deep learning architectures were considered, including the method of the winner team (Neverova et al., 2014b).

In the case of the ChaLearn action/interaction challenges organized in 2014 and 2015 most methods were based on similar approaches. In particular, alternative representations to classical BoW were considered, as Fisher Vector and VLAD (Jegou et al., 2012). Most methods performed sliding windows and SVM classification. In addition, to refine the tracking of interest points, several participants used improved trajectories (Wang and Schmid, 2013).

From the review of the gesture recognition topic, the achieved results in the performed challenges and the rest of papers published in the gesture recognition Special Topic, one can observe that still it is possible that progress will also be made in feature extraction by making better use of the multimodal development data for better transfer learning. For instance, we think that structural hand information around hand joint could be useful to discriminate among gesture categories that may share similar trajectories of hand/arms. Also recent approaches have shown that Random Forest and Deep Learning, such as considering Convolutional Neural Networks, are powerful alternatives to classical gesture recognition approaches, which still open the door for future the design of new gesture recognition classifiers.

In the case of action detection or spotting, most of the methods are still based on sliding-windows approaches, which makes recognition a time-consuming task. Thus, the research on methods that can generate gesture/action candidates from data in a different fashion are still an open issue.

Acknowledgments

This work has been partially supported by ChaLearn Challenges in Machine Learning `http://chalearn.org`, the Human Pose Recovery and Behavior Analysis Group[7], the Pascal2 network of excellence, NSF grants 1128296, 1059235, 1055062, 1338118, 1035913, 0923494, and Spanish project TIN2013-43478-P. Our sponsors include Microsoft and Texas Instrument who donated prizes and provided technical support. The challenges were hosted by Kaggle.com and Coralab.org who are gratefully acknowledged. We thank our co-organizers of ChaLearn gesture and action recognition challenges: Miguel Reyes, Jordi Gonzalez, Xavier Baro, Jamie Shotton, Victor Ponce, Miguel Angel Bautista, and Hugo Jair Escalante.

References

S. Ali and M. Shah. Human action recognition in videos using kinematic features and multiple instance learning. *IEEE TPAMI*, 32, 2010.

J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(9):1685–1699, 2009.

7. HuPBA research group: http://www.maia.ub.es/∼sergio/

M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. Human pose estimation: New benchmark and state of the art analysis. In *CCVPR*. IEEE, 2014.

J. Appenrodt, A. Al-Hamadi, M. Elmezain, and B. Michaelis. Data gathering for gesture recognition systems based on mono color-, stereo color- and thermal cameras. In *Proceedings of the 1st International Conference on Future Generation Information Technology*, FGIT '09, pages 78–86, 2009. ISBN 978-3-642-10508-1.

V. Athitsos and S. Sclaroff. Estimating hand pose from a cluttered image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 432–439, 2003.

V. Athitsos, C. Neidle, S. Sclaroff, J. Nash, A. Stefan, Q. Yuan, and A. Thangali. The American Sign Language lexicon video dataset. In *IEEE Workshop on Computer Vision and Pattern Recognition for Human Communicative Behavior Analysis (CVPR4HB)*, 2008.

A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. J. M. Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In Michael Beigl and Francisco J. Cazorla-Almeida, editors, *ARCS Workshops*, pages 167–176, 2010. ISBN 978-3-8007-3222-7.

L. Baraldi, F. Paci, G. Serra, L. Benini, and R. Cucchiara. Gesture recognition in ego-centric videos using dense trajectories and hand segmentation. In *Proc. of 10th IEEE Embedded Vision Workshop (EVW)*, Columbus, Ohio, June 2014.

X. Baró, J. Gonzàlez, J. Fabian, M. A. Bautista, M. Oliu, H. J. Escalante, I. Guyon, and S. Escalera. ChaLearn looking at people 2015 challenges: action spotting and cultural event recognition. *ChaLearn Looking at People, Computer Vision and Pattern Recognition*, 2015.

B. Bauer, H. Hienz, and K.-F. Kraiss. Video-based continuous sign language recognition using statistical methods. In *International Conference on Pattern Recognition*, pages 2463–2466, 2000.

A. Y. Benbasat and J. A. Paradiso. Compact, configurable inertial gesture recognition. In *CHI '01: CHI '01 extended abstracts on Human factors in computing systems*, pages 183–184. ACM Press, 2001. ISBN 1581133405.

S. Berlemont, G. Lefebvre, S. Duffner, and C. Garcia. Siamese neural network based similarity metric for inertial gesture classification and rejection. *Automatic Face and Gesture Recognition*, 2015.

V. Bloom, D. Makris, and V. Argyriou. G3D: A gaming action dataset and real time action recognition evaluation framework. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 7–12, 2012.

A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(3):257–267, 2001.

L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, pages 1365–1372. IEEE, 2009.

M. Brand, N. Oliver, and A.P. Pentland. Coupled Hidden Markov Models for complex action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 994–999, 1997.

M. Caon, Y. Yong, J. Tscherrig, E. Mugellini, and O. Abou Khaled. Context-aware 3D gesture interaction based on multiple Kinects. In *The First International Conference on Ambient Computing, Applications, Services and Technologies*, page 712, 2011. ISBN 978-1-61208-170-0.

G. Castellano, S. D. Villalba, and A. Camurri. Recognising human emotions from body movement and gesture dynamics. In *Affective computing and intelligent interaction*, pages 71–82. Springer, 2007.

A. Chaudhary, J. L. Raheja, K. Das, and S. Raheja. A survey on hand gesture recognition in context of soft computing. 133:46–55, 2011.

F.S. Chen, C.M. Fu, and C.L. Huang. Hand gesture recognition using a real-time tracking method and Hidden Markov Models. *Image and Video Computing*, 21(8):745–758, August 2003.

M. Chen, G. AlRegib, and B.-H. Juang. 6DMG: A new 6D motion gesture database. In *Multimedia Systems Conference*, pages 83–88, 2012.

C. Conly, P. Doliotis, P. Jangyodsuk, R. Alonzo, and V. Athitsos. Toward a 3D body part detection video dataset and hand tracking benchmark. In *Pervasive Technologies Related to Assistive Environments (PETRA)*, 2013.

C. Conly, Z. Zhang, and V. Athitsos. An integrated RGB-D system for looking up the meaning of signs. In *Pervasive Technologies Related to Assistive Environments (PETRA)*, 2015.

H. Cooper and R. Bowden. Learning signs from subtitles: A weakly supervised approach to sign language recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2568–2574, 2009.

H. Cooper, E.-J. Ong, N. Pugeault, and R. Bowden. Sign language recognition using sub-units. *Journal of Machine Learning Research (JMLR)*, 13(7):2205–2231, 2012.

A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems (RATFG-RTS)*, pages 82–89, 2001.

Y. Cui and J. Weng. Appearance-based hand sign recognition from intensity image sequences. *Computer Vision and Image Understanding*, 78(2):157–176, 2000.

R. Cutler and M. Turk. View-based interpretation of real-time optical flow for gesture recognition. In *Automatic Face and Gesture Recognition*, pages 416–421, 1998.

A. Czabke, J. Neuhauser, and T. C. Lueth. Recognition of interactions with objects based on radio modules. In *International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, 2010.

T.J. Darrell, I.A. Essa, and A.P. Pentland. Task-specific gesture analysis in real-time using interpolated views. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(12): 1236–1242, 1996.

M. de La Gorce, D. J. Fleet, and N. Paragios. Model-based 3D hand pose estimation from monocular video. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(9):1793–1805, 2011.

D. Demirdjian and C. Varri. Recognizing events with temporal random forests. In *Proceedings of the 2009 International Conference on Multimodal Interfaces*, pages 293–296, 2009.

KG Derpanis, M Sizintsev, KJ Cannons, and RP Wildes. Action spotting and recognition based on a spatiotemporal orientation analysis. *IEEE TPAMI*, 35(3):527 – 540, 2013.

P. Dreuw, T. Deselaers, D. Keysers, and H. Ney. Modeling image variability in appearance-based gesture recognition. In *ECCV Workshop on Statistical Methods in Multi-Image and Video Processing*, pages 7–18, 2006.

S. Duffner, S. Berlemont, G. Lefebvre, and C. Garcia. 3D gesture classification with convolutional neural networks. In *The 39th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

U. M. Erdem and S. Sclaroff. Automatic detection of relevant head gestures in american sign language communication. In *International Conference on Pattern Recognition*, pages 460–463, 2002.

S. Escalera, J. Gonzàlez, X. Baró, M. Reyes, I. Guyon, V. Athitsos, H. J. Escalante, L. Sigal, A. Argyros, C. Sminchisescu, R. Bowden, and S. Sclaroff. Chalearn multi-modal gesture recognition 2013: grand challenge and workshop summary. *15th ACM International Conference on Multimodal Interaction*, pages 365–368, 2013a.

S. Escalera, J. Gonzàlez, X. Baró, M. Reyes, O. Lopés, I. Guyon, V. Athitsos, and H. J. Escalante. Multi-modal gesture recognition challenge 2013: Dataset and results. In *ChaLearn Multi-Modal Gesture Recognition Grand Challenge and Workshop, 15th ACM International Conference on Multimodal Interaction*, 2013b.

S. Escalera, X. Baro, J. Gonzalez, M. Bautista, M. Madadi, M. Reyes, V. Ponce, H. J. Escalante, J. Shotton, and I. Guyon. ChaLearn looking at people challenge 2014: Dataset and results. *ChaLearn Looking at People, European Conference on Computer Vision*, 2014.

M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010.

S. R. Fanello, I. Gori, G. Metta, and F. Odone. Keep it simple and sparse: Real-time action recognition. *Journal of Machine Learning Research (JMLR)*, 14(9):2617–2640, 2013.

A. Farhadi, D. A. Forsyth, and R. White. Transfer learning in sign language. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

A. Fathi, X. Ren, and J.M. Rehg. Learning to recognize objects in egocentric activities. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3281–3288.

S. S. Fels. *Glove-TalkII: Mapping hand gestures to speech using neural networks*. PhD thesis, University of Toronto, 1994.

V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *CVPR*, 2008.

S. Fothergill, H. Mentis, P. Kohli, and S. Nowozin. Instructing people for training gestural interactive systems. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 1737–1746, 2012.

W. T. Freeman and M. Roth. Computer vision for computer games. In *Automatic Face and Gesture Recognition*, pages 100–105, 1996.

W. T. Freeman and C. D. Weissman. Television control by hand gestures, 1994.

N. Gillian and J.A. Paradiso. The gesture recognition toolkit. *Journal of Machine Learning Research (JMLR)*, 14, 2014.

A. Gorban, H. Idrees, Y.-G. Jiang, A. Roshan Zamir, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://www.thumos.info/`, 2015.

L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007.

N. Goussies, S. Ubalde, and M. Mejail. Transfer learning decision forests for gesture recognition. *Journal of Machine Learning Research (JMLR)*, 2014.

M. Gowing, A. Ahmadi, F. Destelle, D. S. Monaghan, N. E. O'Connor, and K. Moran. Kinect vs. low-cost inertial sensing for gesture recognition. *Lecture Notes in Computer Science, Springer*, 8325:484–495, 2014.

I. Guyon, V. Athitsos, P. Jangyodsuk, H.J. Escalante, and B. Hamner. Results and analysis of the ChaLearn gesture challenge 2012. In Xiaoyi Jiang, Olga Regina Pereira Bellon, Dmitry Goldgof, and Takeshi Oishi, editors, *Advances in Depth Image Analysis and Applications*, volume 7854 of *Lecture Notes in Computer Science*, pages 186–204. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40302-6. doi: 10.1007/978-3-642-40303-3_19. URL `http://dx.doi.org/10.1007/978-3-642-40303-3_19`.

I. Guyon, V. Athitsos, P. Jangyodsuk, and H. J. Escalante. The ChaLearn gesture dataset (CGD 2011). *Machine Vision and Applications*, 25:1929–1951, 2014.

A. Hernandez-Vela, N. Zlateva, A. Marinov, M. Reyes, P. Radeva, D. Dimov, and S. Escalera. Graph cuts optimization for multi-limb human segmentation in depth maps. *IEEE Computer Vision and Pattern Recognition conference*, 2012.

A. Hernandez-Vela, M. A. Bautista, X. Perez-Sala, V. Ponce, S. Escalera, X. Baro, O. Pujol, and C. Angulo. Probability-based dynamic time warping and bag-of-visual-and-depth-words for human gesture recognition in RGB-D. *Pattern Recognition Letters*, http://dx.doi.org/10.1016/j.patrec.2013.09.009, 2013a.

A. Hernandez-Vela, M. Reyes, V. Ponce, and S. Escalera. Grabcut-based human segmentation in video sequences. *Sensors*, 12(1):15376–15393, 2013b.

G. Hewes. Primate communication and the gestural origins of language. *Current Antropology*, 14: 5–24, 1973.

N. A. Ibraheem and R. Z. Khan. Survey on various gesture recognition technologies and techniques. *International Journal of Computer Applications*, 50(7):38–44, 2012.

C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014.

M. Isard and A. Blake. CONDENSATION - conditional density propagation for visual tracking. *International Journal of Computer Vision (IJCV)*, 29(1):5–28, 1998.

H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE TPAMI*, 34(9):1704–1716, 2012.

F. Jiang, S. Zhang, S. Wu, Y. Gao, and D. Zhao. Multi-layered gesture recognition with Kinect. *Journal of Machine Learning Research (JMLR)*, 2015.

S. Johnson and M. Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *BMVC*, 2010. doi:10.5244/C.24.12.

A. Joshi, S. Sclaroff, M. Betke, and C. Monnier. A random forest approach to segmenting and classifying gestures. *Automatic Face and Gesture Recognition*, 2015.

T. Kadir, R. Bowden, E. Ong, and A. Zisserman. Minimal training, large lexicon, unconstrained sign language recognition. In *British Machine Vision Conference (BMVC)*, volume 2, pages 939–948, 2004.

K. Kahol, P. Tripathi, and S. Panchanathan. Automated gesture segmentation from dance sequences. In *Automatic Face and Gesture Recognition*, pages 883–888, 2004.

H. Kang, C.W. Lee, and K. Jung. Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, 25(15):1701–1714, November 2004.

A. Kapur, A. Kapur, N. Virji-Babul, G. Tzanetakis, and P. F. Driessen. Gesture-based affective computing on motion capture data. In Jianhua Tao, Tieniu Tan, and Rosalind W. Picard, editors, *Affective Computing and Intelligent Interaction*, Lecture Notes in Computer Science, pages 1–7. Springer Berlin Heidelberg, 2005.

S. Kausar and M.Y. Javed. A survey on sign language recognition. *Frontiers of Information Technology*, pages 95–98, 2011.

Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 166–173, 2005.

D. Kelly, J. McDonald, and C. Markham. A person independent system for recognition of hand postures used in sign language. *Pattern Recognition Letters*, 31(11):1359–1368, 2010.

C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *European Conference on Computer Vision (ECCV)*, pages 852–863, 2012.

R. Z. Khan and N. A. Ibraheem. Survey on gesture recognition for hand image postures. *Computer and Information Science*.

T.-K. Kim, S.-F. Wong, and R. Cipolla. Tensor canonical correlation analysis for action classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

H. Kjellström, J. Romero, D. Martínez, and D. Kragić. Simultaneous visual recognition of manipulation actions and manipulated objects. In *European Conference on Computer Vision*, volume 2, pages 336–349, 2008.

D. K. H. Kohlsdorf and T. E. Starner. MAGIC summoning: Towards automatic suggesting and testing of gestures with low probability of false positives during use. *Journal of Machine Learning Research (JMLR)*, 14(1):209–242, 2013.

M. Kolsch and M. Turk. Fast 2D hand tracking with flocks of features and multi-cue integration. In *IEEE Workshop on Real-Time Vision for Human-Computer Interaction*, pages 158–165, 2004.

J. Konecny and M. Hagara. One-shot-learning gesture recognition using hog-hof features. *Journal of Machine Learning Research*, 15:2513–2532, 2014. URL `http://jmlr.org/papers/v15/konecny14a.html`.

Y. Kong, B. Satarboroujeni, and Y. Fu. Hierarchical 3D kernel descriptors for action recognition using depth sequences. *Automatic Face and Gesture Recognition*, 2015.

J. B. Kruskal and M. Liberman. The symmetric time warping algorithm: From continuous to discrete. In *Time Warps*. Addison-Wesley, 1983.

A. Kurakin, Z. Zhang, and Z. Liu. A real time system for dynamic hand gesture recognition with a depth sensor. In *European Signal Processing Conference, EUSIPCO*, pages 1975–1979, 2012.

J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

H. Lane, R. J. Hoffmeister, and B. Bahan. *A Journey into the Deaf-World*. DawnSign Press, San Diego, CA, 1996.

I. Laptev. On space-time interest points. *Int. J. Comput. Vision*, 64(2-3):107–123, September 2005. ISSN 0920-5691. doi: 10.1007/s11263-005-1838-7. URL `http://dx.doi.org/10.1007/s11263-005-1838-7`.

I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, pages 1–8, 2008.

E. Larson, G. Cohn, S. Gupta, X. Ren, B. Harrison, D. Fox, and S. Patel. HeatWave: Thermal imaging for surface user interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2565–2574, 2011.

J. J. LaViola Jr. A survey of hand posture and gesture recognition techniques and technology. Technical report, Providence, RI, USA, 1999.

H.K. Lee and J.H. Kim. An HMM-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973, October 1999.

S.-W. Lee. Automatic gesture recognition for intelligent human-robot interaction. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 645–650, 2006.

C. Li and K. M. Kitani. Pixel-level hand detection for ego-centric videos. *CVPR*, 2013.

W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3D points. *CVPR workshops*, pages 9 – 14, 2010.

H. Liang, J. Yuan, D. Thalmann, and Z. Zhang. Model-based hand pose estimation via spatial-temporal hand parsing and 3D fingertip localization. *The Visual Computer*, 29(6-8):837–848, 2013.

H. Liang, J. Yuan, and D. Thalmann. Parsing the hand in depth images. *IEEE Transactions on Multimedia*, 16(5):1241–1253, 2014.

A. Licsár and T. Szirányi. Hand gesture recognition in camera-projector system*. In *Computer Vision in Human-Computer Interaction*, pages 83–93. Springer, 2004.

Z. Lin, Z. Jiang, and L. S. Davis. Recognizing actions by shape-motion prototype trees. In *IEEE International Conference on Computer Vision, ICCV*, pages 444–451, 2009.

K. Liu, C. Chen, R. Jafari, and N. Kehtarnavaz. Fusion of inertial and depth sensor data for robust hand gesture recognition. *IEEE Sensors Journal*, 14(6):1898–1903, 2014.

L. Liu and L. Shao. Learning discriminative representations from RGB-D video data. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1493–1500, 2013.

O. Lopes, M. Reyes, S. Escalera, and J. Gonzàlez. Spherical blurred shape model for 3-D object and pose recognition: Quantitative analysis and HCI applications in smart environments. *IEEE T. Cybernetics*, 44(12):2379–2390, 2014.

Y. M. Lui. Human gesture recognition on product manifolds. *Journal of Machine Learning Research (JMLR)*, 13(11):3297–3321, 2012.

J. Luo, W. Wang, and H. Qi. Spatio-temporal feature extraction and representation for RGB-D human action recognition. *PRL*, 2014.

J. Ma, W. Gao, J. Wu, and C. Wang. A continuous Chinese Sign Language recognition system. In *Automatic Face and Gesture Recognition*, pages 428–433, 2000.

S. Ma, J. Zhang, N. Ikizler-Cinbis, and S. Sclaroff. Action recognition and localization by hierarchical space-time segments. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

M. R. Malgireddy, I. Nwogu, and V. Govindaraju. Language-motivated approaches to action recognition. *Journal of Machine Learning Research*, 14:2189–2212, 2013. URL `http://jmlr.org/papers/v14/malgireddy13a.html`.

S. Malik and J. Laszlo. Visual touchpad: A two-handed gestural input device. In *International Conference on Multimodal Interfaces*, pages 289–296, 2004.

J. Martin, V. Devin, and J. L. Crowley. Active hand tracking. In *Automatic Face and Gesture Recognition*, pages 573–578, 1998.

A. Martinez and S. Du. A model of the perception of facial expressions of emotion by humans: Research overview and perspectives. *Journal of Machine Learning Research (JMLR)*, 13(5): 1589–1608, 2012.

D. McNeil. How language began, gesture and speech in human evolution. *Cambridge editorial*, 2012.

S. Mitra and T. Acharya. Gesture recognition: A survey. *Trans. Sys. Man Cyber Part C*, 37(3): 311–324, May 2007. ISSN 1094-6977.

Z. Mo and U. Neumann. Real-time hand pose recognition using low-resolution depth images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1499–1505, 2006.

B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. Technical Report 326, MIT, June 1995.

P. Molchanov, S. Gupta, K. Kim, and K. Pulli. Multi-sensor system for drivers hand-gesture recognition. *Automatic Face and Gesture Recognition*, 2015.

J. Nagi, F. Ducatelle, G. A. Di Caro, D. C. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *ICSIPA*, pages 342–347. IEEE, 2011. ISBN 978-1-4577-0243-3.

S. Nayak, S. Sarkar, and B. Loeding. Unsupervised modeling of signs embedded in continuous sentences. In *IEEE Workshop on Vision for Human-Computer Interaction*, 2005.

S. Nayak, K. Duncan, S. Sarkar, and B. Loeding. Finding recurrent patterns from continuous sign language sentences for automated extraction of signs. *Journal of Machine Learning Research (JMLR)*, 13(9):2589–2615, 2012.

C. Neidle, A. Thangali, and S. Sclaroff. Challenges in development of the American Sign Language lexicon video dataset (ASLLVD) corpus. In *Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon*, 2012.

N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout. Hand segmentation with structured convolutional learning. *ACCV*, 2014a.

N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout. Multi-scale deep learning for gesture detection and localization. *ChaLearn Looking at People, European Conference on Computer Vision*, 2014b.

L. Nguyen-Dinh, A. Calatroni, and G. Troster. Robust online gesture recognition with crowdsourced annotations. *Journal of Machine Learning Research (JMLR)*, 15, 2014.

E. Ohn-Bar and M. M. Trivedi. Hand gesture recognition in real-time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE Transactions on Intelligent Transportation Systems*, 2014.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Markerless and efficient 26-DOF hand pose recovery. In *Asian Conference on Computer Vision (ACCV)*, 2010.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2088–2095, 2011.

K. Oka, Y. Sato, and H. Koike. Real-time fingertip tracking and gesture recognition. *IEEE Computer Graphics and Applications*, 22(6):64–71, 2002.

R. Oka. Spotting method for classification of real world data. *The Computer Journal*, 41(8):559–565, July 1998.

E. J. Ong and R. Bowden. A boosted classifier tree for hand shape detection. In *Face and Gesture Recognition*, pages 889–894, 2004.

O. Oreifej and Z. Liu. HON4D: Histogram of oriented 4D normals for activity recognition from depth sequences. *CVPR*, pages 716 – 723, 2013.

A. Pardo, A. Clapes, S. Escalera, and O. Pujol. Actions in context: System for people with dementia. *2nd International Workshop on Citizen Sensor Networks (Citisen2013) at the European Conference on Complex Systems (ECCS'13)*, 2013.

A. Patron-Perez, M. Marszalek, A. Zisserman, and I. D. Reid. High five: Recognising human interactions in TV shows. In *British Machine Vision Conference BMVC*, 2010.

X. Peng, L. Wang, Z. Cai, and Y. Qiao. Action and gesture temporal spotting with super vector representation. In Lourdes Agapito, Michael M. Bronstein, and Carsten Rother, editors, *Computer Vision - ECCV 2014 Workshops*, volume 8925 of *Lecture Notes in Computer Science*, pages 518–527. Springer International Publishing, 2015. ISBN 978-3-319-16177-8. doi: 10.1007/978-3-319-16178-5_36. URL `http://dx.doi.org/10.1007/978-3-319-16178-5_36`.

A. Pieropan, G. Salvi, K.Pauwels, and H. Kjellstrm. Audio-visual classification and detection of human manipulation actions. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

V. Pitsikalis, A. Katsamanis, S. Theodorakis, and P. Maragos. Multimodal gesture recognition via multiple hypotheses rescoring. *Journal of Machine Learning Research (JMLR)*, 2014.

N. Pugeault and R. Bowden. Spelling it out: Real-time ASL fingerspelling recognition. In *ICCV Workshops*, pages 1114–1119, 2011.

A. Quattoni, S. B. Wang, L.-P. Morency, M. Collins, and T. Darrell. Hidden conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(10):1848–1852, 2007.

D. Ramanan. Learning to parse images of articulated bodies. In *NIPS*, pages 1129–1136, 2006.

I. Rauschert, P. Agrawal, R. Sharma, S. Fuhrmann, I. Brewer, and A. MacEachren. Designing a human-centered, multimodal GIS interface to support emergency management. In *ACM International Symposium on Advances in Geographic Information Systems*, pages 119–124, 2002.

J.M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *IEEE International Conference onComputer Vision (ICCV)*, volume 0, pages 612–617, 1995.

Z. Ren, J. Meng, J. Yuan, and Z. Zhang. Robust hand gesture recognition with Kinect sensor. In *ACM International Conference on Multimedia*, pages 759–760, 2011a.

Z. Ren, J. Yuan, and Z. Zhang. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *ACM International Conference on Multimedia*, pages 1093–1096, 2011b.

Z. Ren, J. Yuan, J. Meng, and Z. Zhang. Robust part-based hand gesture recognition using Kinect sensor. *IEEE Transactions on Multimedia*, 15(5):1110–1120, 2013.

A. Roussos, S. Theodorakis, V. Pitsikalis, and P. Maragos. Dynamic affine-invariant shape-appearance handshape features and classification in sign language videos. *Journal of Machine Learning Research (JMLR)*, 14(6):1627–1663, 2013.

S. Ruffieux, D. Lalanne, and E. Mugellini. ChAirGest: A challenge for multimodal mid-air gesture recognition for close HCI. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, pages 483–488, 2013.

A. Sadeghipour, L.-P. Morency, and S. Kopp. Gesture-based object recognition using histograms of guiding strokes. In *British Machine Vision Conference*, pages 44.1–44.11, 2012.

D. Sánchez, M. A. Bautista, and S. Escalera. HuPBA 8k+: Dataset and ECOC-graphcut based segmentation of human limbs. *Neurocomputing*, 2014.

B. Sapp and B. Taskar. Modec: Multimodal decomposable models for human pose estimation. In *CVPR*. IEEE, 2013.

Y. Sato and T. Kobayashi. Extension of Hidden Markov Models to deal with multiple candidates of observations and its application to mobile-robot-oriented gesture recognition. In *International Conference on Pattern Recognition (ICPR)*, pages II: 515–519, 2002.

J.D Schein. *At home among strangers*. Gallaudet U. Press, Washington, DC, 1989.

C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, volume 3, pages 32–36, 2004.

N. Shapovalova, W. Gong., M. Pedersoli, F. X. Roca, and J. Gonzalez. On importance of interactions and context in human action recognition. In *Pattern Recognition and Image Analysis*, pages 58–66, 2011.

J. Shotton, A. W. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1297–1304, 2011.

L. Sigal, A. O. Balan, and M. J. Black. HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision (IJCV)*, 87(1-2):4–27, 2010.

C. Sminchisescu, A. Kanaujia, and D. Metaxas. Conditional models for contextual human motion recognition. *Computer Vision and Image Understanding*, 104:210–220, 2006.

Y. Song, D. Demirdjian, and R. Davis. Multi-signal gesture recognition using temporal smoothing hidden conditional random fields. In *Automatic Face and Gesture Recognition*, pages 388–393, 2011a.

Y. Song, D. Demirdjian, and R. Davis. Tracking body and hands for gesture recognition: NATOPS aircraft handling signals database. In *Automatic Face and Gesture Recognition*, pages 500–506, 2011b.

T. Starner and A. Pentland. Real-time American Sign Language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20 (12):1371–1375, 1998.

N. Stefanov, A. Galata, and R. Hubbold. Real-time hand tracking with variable-length Markov Models of behaviour. In *Real Time Vision for Human-Computer Interaction*, 2005.

B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Filtering using a tree-based estimator. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1063–1070, 2003.

E. Sudderth, M. Mandel, W. Freeman, and A. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In *Neural Information Processing Systems (NIPS)*, 2004.

D. Tran and D. Forsyth. Improved human parsing with a full relational model. In *ECCV*, pages 227–240. IEEE, 2010.

J. Triesch and C. von der Malsburg. A system for person-independent hand posture recognition against complex backgrounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(12):1449–1453, 2001.

J. Triesch and C. von der Malsburg. Classification of hand postures against complex backgrounds using elastic graph matching. *Image and Vision Computing*, 20(13-14):937–943, 2002.

M. Van den Bergh, E. Koller-Meier, and L. Van Gool. Real-time body pose recognition using 2D or 3D haarlets. *International Journal of Computer Vision (IJCV)*, 83(1):72–84, 2009.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 511–518, 2001.

C. Vogler and D Metaxas. Parallel Hidden Markov Models for American Sign Language recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 116–122, 1999.

J. Wan, Q. Ruan, W. Li, and S. Deng. One-shot learning gesture recognition from RGB-D data using bag of features. *Journal of Machine Learning Research*, 14:2549–2582, 2013. URL http://jmlr.org/papers/v14/wan13a.html.

H. Wang and C. Schmid. Action recognition with improved trajectories. In *IEEE International Conference on Computer Vision*, 2013.

H. Wang, A. Stefan, S. Moradi, V. Athitsos, C. Neidle, and F. Kamangar. A system for large vocabulary sign search. In *Workshop on Sign, Gesture and Activity (SGA)*, 2010.

H. Wang, X. Chai, Y. Zhou, and X. Chen. Fast sign language recognition benefited from low rank approximation. In *Automatic Face and Gesture Recognition*, 2015a.

J. Wang, Z. Liu, Y. Wu, and J. Yuan. Learning actionlet ensemble for 3D human action recognition. *IEEE TPAMI*, 36(5):914 – 927, 2014.

R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Transactions on Graphics*, 28(3):63:1–63:8, July 2009.

Y. Wang, D. Tran, Z. Liao, and D. Forsyth. Discriminative hierarchical part-based models for human parsing and action recognition. *Journal of Machine Learning Research (JMLR)*, 13(10): 3075–3102, 2012.

Z. Wang, L. Wang, W. Du, and Q. Yu. Action spotting system using Fisher vector. In *In CVPR ChaLearn Looking at People Workshop 2015*, 2015b.

A. Wexelblat. An approach to natural gesture in virtual environments. *ACM Transactions on Computer-Human Interactions*, 2(3):179–200, 1995.

M. Wilhelm. A generic context aware gesture recognition framework for smart environments. *PerCom Workshops*.

A. D. Wilson and A. F. Bobick. Parametric Hidden Markov Models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 21(9), 1999.

J. Wu and J. Cheng. Bayesian co-boosting for multi-modal gesture recognition. *Journal of Machine Learning Research (JMLR)*, 2014.

Y. Wu and T. S. Huang. View-independent recognition of hand postures. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 88–94, 2000.

Y. Xiao, Z. Zhang, A. Beck, J. Yuan, and D. Thalmann. Human-robot interaction by understanding upper body gestures. *Presence*, 23(2):133 – 154, 2014.

H. D. Yang, S. Sclaroff, and S. W. Lee. Sign language spotting with a threshold model based on conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(7):1264–1277, July 2009.

M. H. Yang, N. Ahuja, and M. Tabb. Extraction of 2D motion trajectories and its application to hand gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(8):1061–1074, 2002.

W. Yang, Y. Wang, and G. Mori. Recognizing human actions from still images with latent poses. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2030–2037, 2010.

X. Yang and Y. Tian. Super normal vector for activity recognition using depth sequences. *CVPR*, 2014a.

X. Yang and Y. Tian. Action recognition using super sparse coding vector with spatio-temporal awareness. *ECCV*, 2014b.

G. Yao, H. Yao, X. Liu, and F. Jiang. Real time large vocabulary continuous sign language recognition based on OP/Viterbi algorithm. In *International Conference on Pattern Recognition*, volume 3, pages 312–315, 2006.

G. Yu, Z. Liu, and J. Yuan. Discriminative orderlet mining for real-time recognition of human-object interaction. *ACCV*, 2014.

J. Yuan, Z. Liu, and Y. Wu. Discriminative video pattern search for efficient action detection. *IEEE TPAMI*, 33(9):1728–1743, 2011.

Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti. American Sign Language recognition with the Kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, pages 279–286, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0641-6. doi: 10.1145/2070481.2070532. URL http://doi.acm.org/10.1145/2070481.2070532.

M. Zanfir, M. Leordeanu, and C. Sminchisescu. The moving pose: An efficient 3D kinematics descriptor for low-latency action recognition and detection. *ICCV*, 2013.

J. Zieren and K.-F. Kraiss. Robust person-independent visual sign language recognition. In *Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*, volume 1, pages 520–528, 2005.

Journal of Machine Learning Research 17 (2016) 1-29 Submitted 11/14; Revised 11/15; Published 5/16

An Emphatic Approach to the Problem of Off-policy Temporal-Difference Learning

Richard S. Sutton sutton@cs.ualberta.ca
A. Rupam Mahmood ashique@cs.ualberta.ca
Martha White whitem@cs.ualberta.ca
Reinforcement Learning and Artificial Intelligence Laboratory
Department of Computing Science, University of Alberta
Edmonton, Alberta, Canada T6G 2E8

Editor: Shie Mannor

Abstract

In this paper we introduce the idea of improving the performance of parametric temporal-difference (TD) learning algorithms by selectively emphasizing or de-emphasizing their updates on different time steps. In particular, we show that varying the emphasis of linear TD(λ)'s updates in a particular way causes its expected update to become stable under off-policy training. The only prior model-free TD methods to achieve this with per-step computation linear in the number of function approximation parameters are the gradient-TD family of methods including TDC, GTD(λ), and GQ(λ). Compared to these methods, our *emphatic TD(λ)* is simpler and easier to use; it has only one learned parameter vector and one step-size parameter. Our treatment includes general state-dependent discounting and bootstrapping functions, and a way of specifying varying degrees of interest in accurately valuing different states.

Keywords: Temporal-difference learning, Off-policy learning, Function approximation, Stability, Convergence.

1. Parametric Temporal-Difference Learning

Temporal-difference (TD) learning is perhaps the most important idea to come out of the field of reinforcement learning. The problem it solves is that of efficiently learning to make a sequence of long-term predictions about how a dynamical system will evolve over time. The key idea is to use the change (temporal difference) from one prediction to the next as an error in the earlier prediction. For example, if you are predicting on each day what the stock-market index will be at the end of the year, and events lead you one day to make a much lower prediction, then a TD method would infer that the predictions made prior to the drop were probably too high; it would adjust the parameters of its prediction function so as to make lower predictions for similar situations in the future. This approach contrasts with conventional approaches to prediction, which wait until the end of the year when the final stock-market index is known before adjusting any parameters, or else make only short-term (e.g., one-day) predictions and then iterate them to produce a year-end prediction. The TD approach is more convenient computationally because it requires less memory and because its computations are spread out uniformly over the year (rather than being bunched together all at the end of the year). A less obvious advantage of the TD approach is that it often produces statistically more accurate answers than conventional approaches (Sutton 1988).

Parametric temporal-difference learning was first studied as the key "learning by generalization" algorithm in Samuel's (1959) checker player. Sutton (1988) introduced the TD(λ) algorithm and proved convergence in the mean of episodic linear TD(0), the simplest parametric TD method. The potential power of parametric TD learning was convincingly demonstrated by Tesauro (1992, 1995) when he applied TD(λ) combined with neural networks and self play to obtain ultimately the world's best backgammon player. Dayan (1992) proved convergence in expected value of episodic linear TD(λ) for all $\lambda \in [0, 1]$, and Tsitsiklis and Van Roy (1997) proved convergence with probability one of discounted continuing linear TD(λ). Watkins (1989) extended TD learning to control in the form of Q-learning and proved its convergence in the tabular case (without function approximation, Watkins & Dayan 1992), while Rummery (1995) extended TD learning to control in an on-policy form as the Sarsa(λ) algorithm. Bradtke and Barto (1996), Boyan (1999), and Nedic and Bertsekas (2003) extended linear TD learning to a least-squares form called LSTD(λ). Parametric TD methods have also been developed as models of animal learning (e.g., Sutton & Barto 1990, Klopf 1988, Ludvig, Sutton & Kehoe 2012) and as models of the brain's reward systems (Schultz, Dayan & Montague 1997), where they have been particularly influential (e.g., Niv & Schoenbaum 2008, O'Doherty 2012). Sutton (2009, 2012) has suggested that parametric TD methods could be key not just to learning about reward, but to the learning of world knowledge generally, and to perceptual learning. Extensive analysis of parametric TD learning as stochastic approximation is provided by Bertsekas (2012, Chapter 6) and Bertsekas and Tsitsiklis (1996).

Within reinforcement learning, TD learning is typically used to learn approximations to the value function of a Markov decision process (MDP). Here the value of a state s, denoted $v_\pi(s)$, is defined as the sum of the expected long-term discounted rewards that will be received if the process starts in s and subsequently takes actions as specified by the decision-making policy π, called the *target policy*. If there are a small number of states, then it may be practical to approximate the function v_π by a table, but more generally a parametric form is used, such as a polynomial, multi-layer neural network, or linear mapping. Also key is the source of the data, in particular, the policy used to interact with the MDP. If the data is obtained while following the target policy π, then good convergence results are available for linear function approximation. This case is called *on-policy* learning because learning occurs while "on" the policy being learned about. In the alternative, *off-policy* case, one seeks to learn about v_π while behaving (selecting actions) according to a different policy called the *behavior policy*, which we denote by μ. Baird (1995) showed definitively that parametric TD learning was much less robust in the off-policy case (for $\lambda < 1$) by exhibiting counterexamples for which both linear TD(0) and linear Q-learning had unstable expected updates and, as a result, the parameters of their linear function approximation diverged to infinity. This is a serious limitation, as the off-policy aspect is key to Q-learning (perhaps the single most popular reinforcement learning algorithm), to learning from historical data and from demonstrations, and to the idea of using TD learning for perception and world knowledge.

Over the years, several different approaches have been taken to solving the problem of off-policy learning with TD learning ($\lambda < 1$). Baird (1995) proposed an approach based on gradient descent in the Bellman error for general parametric function approximation that has the desired computational properties, but which requires access to the MDP for double sampling and which in practice often learns slowly. Gordon (1995, 1996) proposed restricting attention to function approximators that are averagers, but this does not seem to be possible without storing many of the training examples, which would defeat the primary strength that we seek to obtain from parametric function approximation. The LSTD(λ) method was always relatively robust to off-policy training (e.g., Lagoudakis & Parr 2003, Yu 2010, Mahmood, van Hasselt & Sutton 2014), but its per-step computational complexity is quadratic in the number of parameters of the function approximator, as opposed to the linear complexity of TD(λ) and the other methods. Perhaps the most successful approach to date is the gradient-TD approach (e.g., Maei 2011, Sutton et al. 2009, Maei et al. 2010), including hybrid methods such as HTD (Hackman 2012). Gradient-TD methods are of linear complexity and guaranteed to converge for appropriately chosen step-size parameters but are more complex than TD(λ) because they require a second auxiliary set of parameters with a second step size that must be set in a problem-dependent way for good performance. The studies by White (2015), Geist and Scherrer (2014), and Dann, Neumann, and Peters (2014) are the most extensive empirical explorations of gradient-TD and related methods to date.

In this paper we explore a new approach to solving the problem of off-policy TD learning with function approximation. The approach has novel elements but is similar to that developed by Precup, Sutton, and Dasgupta in 2001. They proposed to use importance sampling to reweight the updates of linear TD(λ), emphasizing or de-emphasizing states as they were encountered, and thereby create a weighting equivalent to the stationary distribution under the target policy, from which the results of Tsitsiklis and Van Roy (1997) would apply and guarantee convergence. As we discuss later, this approach has very high variance and was eventually abandoned in favor of the gradient-TD approach. The new approach we explore in this paper is similar in that it also varies emphasis so as to reweight the distribution of linear TD(λ) updates, but to a different goal. The new goal is to create a weighting equivalent to the *followon distribution* for the target policy started in the stationary distribution of the behavior policy. The followon distribution weights states according to how often they would occur prior to termination by discounting *if the target policy was followed*.

Our main result is to prove that varying emphasis according to the followon distribution produces a new version of linear TD(λ), *called emphatic TD(λ)*, that is stable under general off-policy training. By "stable" we mean that the expected update over the ergodic distribution (Tsitsiklis & Van Roy 1997) is a contraction, involving a positive definite matrix. We concentrate on stability in this paper because it is a prerequisite for full convergence of the stochastic algorithm. Demonstrations that the linear TD(λ) is not stable under off-policy training have been the focus of previous counterexamples (Baird 1995, Tsitsiklis & Van Roy 1996, 1997, see Sutton & Barto 1998). Substantial additional theoretical machinery would be required for a full convergence proof. Recent work by Yu (2015a,b) builds on our stability result to prove that the emphatic TD(λ) converges with probability one.

In this paper we first treat the simplest algorithm for which the difficulties of off-policy temporal-difference (TD) learning arise—the TD(0) algorithm with linear function approximation. We examine the conditions under which the expected update of on-policy TD(0) is stable, then why those conditions do not apply under off-policy training, and finally how they can be recovered for off-policy training using established importance-sampling methods together with the emphasis idea. After introducing the basic idea of emphatic algorithms using the special case of TD(0), we then develop the general case. In particular, we consider a case with general state-dependent discounting and bootstrapping functions, and with a user-specified allocation of function approximation resources. Our new theoretical results and the emphatic TD(λ) algorithm are presented fully for this general case. Empirical examples elucidating the main theoretical results are presented in the last section prior to the conclusion.

2. On-policy Stability of TD(0)

To begin, let us review the conditions for stability of conventional TD(λ) under on-policy training with data from a continuing finite Markov decision process. Consider the simplest function approximation case, that of linear TD(λ) with $\lambda = 0$ and constant discount-rate parameter $\gamma \in [0, 1)$. Conventional linear TD(0) is defined by the following update to the parameter vector $\boldsymbol{\theta}_t \in \mathbb{R}^n$, made at each of a sequence of time steps $t = 0, 1, 2, \ldots$, on transition from state $S_t \in \mathcal{S}$ to state $S_{t+1} \in \mathcal{S}$, taking action $A_t \in \mathcal{A}$ and receiving reward $R_{t+1} \in \mathbb{R}$:

$$\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t + \alpha \left(R_{t+1} + \gamma \boldsymbol{\theta}_t^\top \boldsymbol{\phi}(S_{t+1}) - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}(S_t) \right) \boldsymbol{\phi}(S_t), \tag{1}$$

where $\alpha > 0$ is a step-size parameter, and $\boldsymbol{\phi}(s) \in \mathbb{R}^n$ is the feature vector corresponding to state s. The notation "$\doteq$" indicates an equality by definition rather than one that follows from previous definitions. In on-policy training, the actions are chosen according to a target policy $\pi : \mathcal{A} \times \mathcal{S} \to [0, 1]$, where $\pi(a|s) \doteq \mathbb{P}\{A_t = a | S_t = s\}$. The state and action sets $\mathcal{S}$ and $\mathcal{A}$ are assumed to be finite, but the number of states is assumed much larger than the number of learned parameters, $|\mathcal{S}| \doteq N \gg n$, so that function approximation is necessary. We use linear function approximation, in which the inner product of the parameter vector and the feature vector for a state is meant to be an approximation to the value of that state:

$$\boldsymbol{\theta}_t^\top \boldsymbol{\phi}(s) \approx v_\pi(s) \doteq \mathbb{E}_\pi[G_t | S_t = s], \tag{2}$$

where $\mathbb{E}_\pi[\cdot]$ denotes an expectation conditional on all actions being selected according to π, and G_t, the *return* at time t, is defined by

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots . \tag{3}$$

The TD(0) update (1) can be rewritten to make the stability issues more transparent:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha \Big(\underbrace{R_{t+1} \boldsymbol{\phi}(S_t)}_{\mathbf{b}_t \in \mathbb{R}^n} - \underbrace{\boldsymbol{\phi}(S_t) \left(\boldsymbol{\phi}(S_t) - \gamma \boldsymbol{\phi}(S_{t+1}) \right)^\top}_{\mathbf{A}_t \in \mathbb{R}^{n \times n}} \boldsymbol{\theta}_t \Big) \\ &= \boldsymbol{\theta}_t + \alpha(\mathbf{b}_t - \mathbf{A}_t \boldsymbol{\theta}_t) \\ &= (\mathbf{I} - \alpha \mathbf{A}_t) \boldsymbol{\theta}_t + \alpha \mathbf{b}_t. \end{aligned} \tag{4}$$

The matrix $\mathbf{A}_t$ multiplies the parameter $\boldsymbol{\theta}_t$ and is thereby critical to the stability of the iteration. To develop intuition, consider the special case in which $\mathbf{A}_t$ is a diagonal matrix. If any of the diagonal elements are negative, then the corresponding diagonal element of $\mathbf{I}-\alpha\mathbf{A}_t$ will be greater than one, and the corresponding component of $\boldsymbol{\theta}_t$ will be amplified, which will lead to divergence if continued. (The second term $(\alpha\mathbf{b}_t)$ does not affect the stability of the iteration.) On the other hand, if the diagonal elements of $\mathbf{A}_t$ are all positive, then α can be chosen smaller than one over the largest of them, such that $\mathbf{I}-\alpha\mathbf{A}_t$ is diagonal with all diagonal elements between 0 and 1. In this case the first term of the update tends to shrink $\boldsymbol{\theta}_t$, and stability is assured. In general, $\boldsymbol{\theta}_t$ will be reduced toward zero whenever $\mathbf{A}_t$ is positive definite.[1]

In actuality, however, $\mathbf{A}_t$ and $\mathbf{b}_t$ are random variables that vary from step to step, in which case stability is determined by the steady-state expectation, $\lim_{t\to\infty}\mathbb{E}[\mathbf{A}_t]$. In our setting, after an initial transient, states will be visited according to the steady-state distribution under π (which we assume exists). We represent this distribution by a vector $\mathbf{d}_\pi$, each component of which gives the limiting probability of being in a particular state[2] $[\mathbf{d}_\pi]_s \doteq d_\pi(s) \doteq \lim_{t\to\infty}\mathbb{P}\{S_t{=}s\}$, which we assume exists and is positive at all states (any states not visited with nonzero probability can be removed from the problem). The special property of the steady-state distribution is that once the process is in it, it remains in it. Let $\mathbf{P}_\pi$ denote the $N\times N$ matrix of transition probabilities $[\mathbf{P}_\pi]_{ij} \doteq \sum_a \pi(a|i)p(j|i,a)$ where $p(j|i,a) \doteq \mathbb{P}\{S_{t+1}{=}j|S_t{=}i, A_t{=}a\}$. Then the special property of $\mathbf{d}_\pi$ is that

$$\mathbf{P}_\pi^\top \mathbf{d}_\pi = \mathbf{d}_\pi. \tag{5}$$

Consider any stochastic algorithm of the form (4), and let $\mathbf{A} \doteq \lim_{t\to\infty}\mathbb{E}[\mathbf{A}_t]$ and $\mathbf{b} \doteq \lim_{t\to\infty}\mathbb{E}[\mathbf{b}_t]$. We define the stochastic algorithm to be *stable* if and only if the corresponding deterministic algorithm,

$$\bar{\boldsymbol{\theta}}_{t+1} \doteq \bar{\boldsymbol{\theta}}_t + \alpha(\mathbf{b}-\mathbf{A}\bar{\boldsymbol{\theta}}_t), \tag{6}$$

is convergent to a unique fixed point independent of the initial $\bar{\boldsymbol{\theta}}_0$. This will occur iff the $\mathbf{A}$ matrix has a full set of eigenvalues all of whose real parts are positive. If a stochastic algorithm is stable and α is reduced according to an appropriate schedule, then its parameter vector may converge with probability one. However, in this paper we focus only on stability as a prerequisite for convergence (of the original stochastic algorithm), leaving convergence itself to future work. If the stochastic algorithm converges, it is to a fixed point $\bar{\boldsymbol{\theta}}$ of the deterministic algorithm, at which $\mathbf{A}\bar{\boldsymbol{\theta}}=\mathbf{b}$, or $\bar{\boldsymbol{\theta}}=\mathbf{A}^{-1}\mathbf{b}$. (Stability assures existence of the inverse.) In this paper we focus on establishing stability by proving that $\mathbf{A}$ is positive definite. From definiteness it immediately follows that $\mathbf{A}$ has a full set of eigenvectors (because $\mathbf{y}^\top\mathbf{A}\mathbf{y}>0, \forall\mathbf{y}\neq\mathbf{0}$) and that the corresponding eigenvalues all have real parts.[3]

1. A real matrix $\mathbf{A}$ is defined to be *positive definite* in this paper iff $\mathbf{y}^\top\mathbf{A}\mathbf{y}>0$ for any real vector $\mathbf{y}\neq\mathbf{0}$.
2. Here and throughout the paper we use brackets with subscripts to denote the individual elements of vectors and matrices.
3. To see the latter, let $\mathrm{Re}(x)$ denote the real part of a complex number x, and let $\mathbf{y}^*$ denotes the conjugate transpose of a complex vector $\mathbf{y}$. Then, for any eigenvalue–eigenvector pair $\lambda,\mathbf{y}$: $0<\mathrm{Re}(\mathbf{y}^*\mathbf{A}\mathbf{y}) = \mathrm{Re}(\mathbf{y}^*\lambda\mathbf{y}) = \mathrm{Re}(\lambda)\mathbf{y}^*\mathbf{y} \implies 0<\mathrm{Re}(\lambda)$.

Now let us return to analyzing on-policy TD(0). Its $\mathbf{A}$ matrix is

$$\begin{aligned}
\mathbf{A} = \lim_{t\to\infty}\mathbb{E}[\mathbf{A}_t] &= \lim_{t\to\infty}\mathbb{E}_\pi\Big[\boldsymbol{\phi}(S_t)\left(\boldsymbol{\phi}(S_t)-\gamma\boldsymbol{\phi}(S_{t+1})\right)^\top\Big] \\
&= \sum_s d_\pi(s)\,\boldsymbol{\phi}(s)\left(\boldsymbol{\phi}(s)-\gamma\sum_{s'}[\mathbf{P}_\pi]_{ss'}\boldsymbol{\phi}(s')\right)^\top \\
&= \boldsymbol{\Phi}^\top\mathbf{D}_\pi(\mathbf{I}-\gamma\mathbf{P}_\pi)\boldsymbol{\Phi},
\end{aligned}$$

where $\boldsymbol{\Phi}$ is the $N\times n$ matrix with the $\boldsymbol{\phi}(s)$ as its rows, and $\mathbf{D}_\pi$ is the $N\times N$ diagonal matrix with $\mathbf{d}_\pi$ on its diagonal. This $\mathbf{A}$ matrix is typical of those we consider in this paper in that it consists of $\boldsymbol{\Phi}^\top$ and $\boldsymbol{\Phi}$ wrapped around a distinctive $N\times N$ matrix that varies with the algorithm and the setting, and which we call the *key matrix*. An $\mathbf{A}$ matrix of this form will be positive definite whenever the corresponding key matrix is positive definite.[4] In this case the key matrix is $\mathbf{D}_\pi(\mathbf{I}-\gamma\mathbf{P}_\pi)$.

For a key matrix of this type, positive definiteness is assured if all of its columns sum to nonnegative numbers. This was shown by Sutton (1988, p. 27) based on two previously established theorems. One theorem says that any matrix $\mathbf{M}$ is positive definite if and only if the symmetric matrix $\mathbf{S}=\mathbf{M}+\mathbf{M}^\top$ is positive definite (Sutton 1988, appendix). The second theorem says that any symmetric real matrix $\mathbf{S}$ is positive definite if the absolute values of its diagonal entries are greater than the sum of the absolute values of the corresponding off-diagonal entries (Varga 1962, p. 23). For our key matrix, $\mathbf{D}_\pi(\mathbf{I}-\gamma\mathbf{P}_\pi)$, the diagonal entries are positive and the off-diagonal entries are negative, so all we have to show is that each row sum plus the corresponding column sum is positive. The row sums are all positive because $\mathbf{P}_\pi$ is a stochastic matrix and $\gamma<1$. Thus it only remains to show that the column sums are nonnegative.

Note that the row vector of the column sums of any matrix $\mathbf{M}$ can be written as $\mathbf{1}^\top\mathbf{M}$, where $\mathbf{1}$ is the column vector with all components equal to 1. The column sums of our key matrix, then, are:

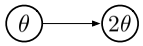
$$\begin{aligned}
\mathbf{1}^\top\mathbf{D}_\pi(\mathbf{I}-\gamma\mathbf{P}_\pi) &= \mathbf{d}_\pi^\top(\mathbf{I}-\gamma\mathbf{P}_\pi) \\
&= \mathbf{d}_\pi^\top - \gamma\mathbf{d}_\pi^\top\mathbf{P}_\pi \\
&= \mathbf{d}_\pi^\top - \gamma\mathbf{d}_\pi^\top \qquad \text{(by (5))} \\
&= (1-\gamma)\mathbf{d}_\pi,
\end{aligned}$$

all components of which are positive. Thus, the key matrix and its $\mathbf{A}$ matrix are positive definite, and on-policy TD(0) is stable. Additional conditions and a schedule for reducing α over time (as in Tsitsiklis and Van Roy 1997) are needed to prove convergence with probability one, $\boldsymbol{\theta}_\infty=\mathbf{A}^{-1}\mathbf{b}$, but the analysis above includes the most important steps that vary from algorithm to algorithm.

4. Strictly speaking, positive definiteness of the key matrix assures only that $\mathbf{A}$ is positive *semi*-definite, because it is possible that $\boldsymbol{\Phi}\mathbf{y}=\mathbf{0}$ for some $\mathbf{y}\neq\mathbf{0}$, in which case $\mathbf{y}^\top\mathbf{A}\mathbf{y}$ will be zero as well. To rule this out, we assume, as is commonly done, that the columns of $\boldsymbol{\Phi}$ are linearly independent (i.e., that the features are not redundant), and thus that $\boldsymbol{\Phi}\mathbf{y}=\mathbf{0}$ only if $\mathbf{y}=\mathbf{0}$. If this were not true, then convergence (if it occurs) may not be to a unique $\boldsymbol{\theta}_\infty$, but rather to a subspace of parameter vectors all of which produce the same approximate value function.

3. Instability of Off-policy TD(0)

Before developing the off-policy setting in detail, it is useful to understand informally why TD(0) is susceptible to instability. TD learning involves learning an estimate from an estimate, which can be problematic if there is generalization between the two estimates. For example, suppose there is a transition between two states with the same feature representation except that the second is twice as big:



where here θ and 2θ are the estimated values of the two states—that is, their feature representations are a single feature that is 1 for the first state and 2 for the second (cf. Tsitsiklis & Van Roy 1996). Now suppose that θ is 10 and the reward on the transition is 0. The transition is then from a state valued at 10 to a state valued at 20. If γ is near 1 and α is 0.1, then θ will be increased to approximately 11. But then the next time the transition occurs there will be an even bigger increase in value, from 11 to 22, and a bigger increase in θ, to approximately 12.1. If this transition is experienced repeatedly on its own, then the system is unstable and the parameter increases without bound—it diverges. We call this the $\theta \to 2\theta$ problem.

In on-policy learning, repeatedly experiencing just this single problematic transition cannot happen, because, after the highly-valued 2θ state has been entered, it must then be exited. The transition from it will either be to a lesser or equally-valued state, in which case θ will be significantly decreased, or to an even higher-valued state which in turn must be followed by an even larger decrease in its estimated value or a still higher-valued state. Eventually, the promise of high value must be made good in the form of a high reward, or else estimates will be decreased, and this ultimately constrains θ and forces stability and convergence. In the off-policy case, however, if there is a deviation from the target policy then the promise is excused and need never be fulfilled. Later in this section we present a complete example of how the $\theta \to 2\theta$ problem can cause instability and divergence under off-policy training.

With these intuitions, we now detail our off-policy setting. As in the on-policy case, the data is a single, infinite-length trajectory of actions, rewards, and feature vectors generated by a continuing finite Markov decision process. The difference is that the actions are selected not according to the target policy π, but according to a different *behavior policy* $\mu : \mathcal{A} \times \mathcal{S} \to [0, 1]$, yet still we seek to estimate state values under π (as in (2)). Of course, it would be impossible to estimate the values under π if the actions that π would take were never taken by μ and their consequences were never observed. Thus we assume that $\mu(a|s) > 0$ for every state and action for which $\pi(a|s) > 0$. This is called the assumption of *coverage*. It is trivially satisfied by any ϵ-greedy or soft behavior policy. As before we assume that there is a stationary distribution $d_\mu(s) \doteq \lim_{t\to\infty} \mathbb{P}\{S_t{=}s\} > 0, \forall s \in \mathcal{S}$, with corresponding N-vector $\mathbf{d}_\mu$.

Even if there is coverage, the behavior policy will choose actions with proportions different from the target policy. For example, some actions taken by μ might never be chosen by π. To address this, we use importance sampling to correct for the relative probability of taking the action actually taken, A_t, in the state actually encountered, S_t, under the target

and behavior policies:

$$\rho_t \doteq \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}.$$

This quantity is called the *importance sampling ratio* at time t. Note that its expected value is one:

$$\mathbb{E}_\mu[\rho_t|S_t{=}s] = \sum_a \mu(a|s)\frac{\pi(a|s)}{\mu(a|s)} = \sum_a \pi(a|s) = 1.$$

The ratio will be exactly one only on time steps on which the action probabilities for the two policies are exactly the same; these time steps can be treated the same as in the on-policy case. On other time steps the ratio will be greater or less than one depending on whether the action taken was more or less likely under the target policy than under the behavior policy, and some kind of correction is needed.

In general, for any random variable Z_{t+1} dependent on S_t, A_t and S_{t+1}, we can recover its expectation under the target policy by multiplying by the importance sampling ratio:

$$\begin{aligned}\mathbb{E}_\mu[\rho_t Z_{t+1}|S_t{=}s] &= \sum_a \mu(a|s)\frac{\pi(a|s)}{\mu(a|s)} Z_{t+1} \\ &= \sum_a \pi(a|s) Z_{t+1} \\ &= \mathbb{E}_\pi[Z_{t+1}|S_t{=}s], \qquad \forall s \in \mathcal{S}. \end{aligned} \tag{7}$$

We can use this fact to begin to adapt TD(0) for off-policy learning (Precup, Sutton & Singh 2000). We simply multiply the whole TD(0) update (1) by ρ_t:

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &\doteq \boldsymbol{\theta}_t + \rho_t\,\alpha\left(R_{t+1} + \gamma\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t \\ &= \boldsymbol{\theta}_t + \alpha\Big(\underbrace{\rho_t R_{t+1}\boldsymbol{\phi}_t}_{\mathbf{b}_t} - \underbrace{\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top}_{\mathbf{A}_t}\boldsymbol{\theta}_t\Big),\end{aligned} \tag{8}$$

where here we have used the shorthand $\boldsymbol{\phi}_t \doteq \boldsymbol{\phi}(S_t)$. Note that if the action taken at time t is never taken under the target policy in that state, then $\rho_t = 0$ and there is no update on that step, as desired. We call this algorithm *off-policy TD(0)*.

Off-policy TD(0)'s $\mathbf{A}$ matrix is

$$\begin{aligned}\mathbf{A} = \lim_{t\to\infty}\mathbb{E}[\mathbf{A}_t] &= \lim_{t\to\infty}\mathbb{E}_\mu\Big[\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top\Big] \\ &= \sum_s d_\mu(s)\mathbb{E}_\mu\Big[\rho_k\boldsymbol{\phi}_k\left(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1}\right)^\top\Big|S_k = s\Big] \\ &= \sum_s d_\mu(s)\mathbb{E}_\pi\Big[\boldsymbol{\phi}_k\left(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1}\right)^\top\Big|S_k = s\Big] \qquad \text{(by (7))} \\ &= \sum_s d_\mu(s)\,\boldsymbol{\phi}(s)\left(\boldsymbol{\phi}(s) - \gamma\sum_{s'}[\mathbf{P}_\pi]_{ss'}\boldsymbol{\phi}(s')\right)^\top \\ &= \boldsymbol{\Phi}^\top\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi},\end{aligned}$$

where $\mathbf{D}_\mu$ is the $N \times N$ diagonal matrix with the stationary distribution $\mathbf{d}_\mu$ on its diagonal. Thus, the key matrix that must be positive definite is $\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi)$ and, unlike in the on-policy case, the distribution and the transition probabilities do not match. We do not have an analog of (5), $\mathbf{P}_\pi^\top \mathbf{d}_\mu \neq \mathbf{d}_\mu$, and in fact the column sums may be negative and the matrix not positive definite, in which case divergence of the parameter is likely.

A simple $\theta \to 2\theta$ example of divergence that fits the setting in this section is shown in Figure 1. From each state there are two actions, left and right, which take the process to the left or right states. All the rewards are zero. As before, there is a single parameter θ and the single feature is 1 and 2 in the two states such that the approximate values are θ and 2θ as shown. The behavior policy is to go left and right with equal probability from both states, such that equal time is spent on average in both states, $\mathbf{d}_\mu = (0.5, 0.5)^\top$. The target policy is to go right in both states. We seek to learn the value from each state given that the right action is continually taken. The transition probability matrix for this example is:

$$\mathbf{P}_\pi = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

The key matrix is

$$\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \times \begin{bmatrix} 1 & -0.9 \\ 0 & 0.1 \end{bmatrix} = \begin{bmatrix} 0.5 & -0.45 \\ 0 & 0.05 \end{bmatrix}. \tag{9}$$

We can see an immediate indication that the key matrix may not be positive definite in that the second column sums to a negative number. More definitively, one can show that it is not positive definite by multiplying it on both sides by $\mathbf{y} = \mathbf{\Phi} = (1, 2)^\top$:

$$\mathbf{\Phi}^\top \mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{\Phi} = \begin{bmatrix} 1 & 2 \end{bmatrix} \times \begin{bmatrix} 0.5 & -0.45 \\ 0 & 0.05 \end{bmatrix} \times \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \end{bmatrix} \times \begin{bmatrix} -0.4 \\ 0.1 \end{bmatrix} = -0.2.$$

That this is negative means that the key matrix is not positive definite. We have also calculated here the $\mathbf{A}$ matrix; it is this negative scalar, $\mathbf{A} = -0.2$. Clearly, this expected update and algorithm are not stable.

It is also easy to see the instability of this example more directly, without matrices. We know that only transitions under the right action cause updates, as ρ_t will be zero for the others. Assume for concreteness that initially $\theta_t = 10$ and that $\alpha = 0.1$. On a right transition from the first state the update will be

$$\begin{aligned} \theta_{t+1} &= \theta_t + \rho_t\alpha\left(R_{t+1} + \gamma\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t \\ &= 10 + 2 \cdot 0.1\,(0 + 0.9 \cdot 10 \cdot 2 - 10 \cdot 1)\,1 \\ &= 10 + 1.6, \end{aligned}$$

Figure 1: $\theta \to 2\theta$ example without a terminal state.

whereas, on a right transition from the second state the update will be

$$\begin{aligned} \theta_{t+1} &= \theta_t + \rho_t\alpha\left(R_{t+1} + \gamma\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t \\ &= 10 + 2 \cdot 0.1\,(0 + 0.9 \cdot 10 \cdot 2 - 10 \cdot 2)\,2 \\ &= 10 - 0.8. \end{aligned}$$

These two transitions occur equally often, so the net change will be positive. That is, θ will increase, moving farther from its correct value, zero. Everything is linear in θ, so the next time around, with a larger starting θ, the increase in θ will be larger still, and divergence occurs. A smaller value of α would not prevent divergence, only reduce its rate.

4. Off-policy Stability of Emphatic TD(0)

The deep reason for the difficulty of off-policy learning is that the behavior policy may take the process to a distribution of states different from that which would be encountered under the target policy, yet the states might appear to be the same or similar because of function approximation. Earlier work by Precup, Sutton and Dasgupta (2001) attempted to completely correct for the different state distribution using importance sampling ratios to reweight the states encountered. It is theoretically possible to convert the state weighting from d_μ to d_π using the product of all importance sampling ratios from time 0, but in practice this approach has extremely high variance and is infeasible for the continuing (non-episodic) case. It works in theory because after converting the weighting the key matrix is $\mathbf{D}_\pi(\mathbf{I} - \gamma\mathbf{P}_\pi)$ again, which we know to be positive definite.

Most subsequent works abandoned the idea of completely correcting for the state distribution. For example, the work on gradient-TD methods (e.g., Sutton et al. 2009, Maei 2011) seeks to minimize the mean-squared projected Bellman error weighted by d_μ. We call this an *excursion* setting because we can think of the contemplated switch to the target policy as an excursion from the steady-state distribution of the behavior policy, d_μ. The excursions would start from d_μ and then follow π until termination, followed by a resumption of μ and thus a gradual return to d_μ. Of course these excursions never actually occur during off-policy learning, they are just contemplated, and thus the state distribution in fact never leaves d_μ. It is the excursion view that we take in this paper, but still we use techniques similar to those introduced by Precup et al. (2001) to determine an emphasis weighting that corrects for the state distribution, only toward a different goal (see also Kolter 2011).

The excursion notion suggests a different weighting of TD(0) updates. We consider that at every time step we are beginning a new contemplated excursion from the current state. The excursion thus would begin in a state sampled from d_μ. If an excursion started it would pass through a sequence of subsequent states and actions prior to termination. Some of the actions that are actually taken (under μ) are relatively likely to occur under the target policy as compared to the behavior policy, while others are relatively unlikely; the corresponding states can be appropriately reweighted based on importance sampling ratios. Thus, there will still be a product of importance sampling ratios, but only since the beginning of the excursion, and the variance will also be tamped down by the discounting; the variance will be much less than in the earlier approach. In the simplest case of an off-policy emphatic algorithm, the update at time t is emphasized or de-emphasized proportional to a new scalar

variable F_t, defined by $F_0 = 1$ and

$$F_t \doteq \gamma\rho_{t-1}F_{t-1} + 1, \qquad \forall t > 0, \tag{10}$$

which we call the *followon trace*. Specifically, we define *emphatic TD(0)* by the following update:

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &\doteq \boldsymbol{\theta}_t + \alpha F_t \rho_t \left(R_{t+1} + \gamma \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right) \boldsymbol{\phi}_t \\ &= \boldsymbol{\theta}_t + \alpha\Big(\underbrace{F_t\rho_t R_{t+1}\boldsymbol{\phi}_t}_{\mathbf{b}_t} - \underbrace{F_t\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top}_{\mathbf{A}_t}\boldsymbol{\theta}_t\Big).\end{aligned} \tag{11}$$

Emphatic TD(0)'s $\mathbf{A}$ matrix is

$$\begin{aligned}\mathbf{A} = \lim_{t\to\infty}\mathbb{E}[\mathbf{A}_t] &= \lim_{t\to\infty}\mathbb{E}_\mu\Big[F_t\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top\Big] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu\Big[F_t\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top\Big|S_t = s\Big] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t = s]\,\mathbb{E}_\mu\Big[\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top\Big|S_t = s\Big]\end{aligned}$$

(because, given S_t, F_t is independent of $\rho_t\boldsymbol{\phi}_t\left(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1}\right)^\top$)

$$\begin{aligned}&= \sum_s \underbrace{d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t = s]}_{f(s)}\,\mathbb{E}_\mu\Big[\rho_k\boldsymbol{\phi}_k\left(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1}\right)^\top\Big|S_k = s\Big] \\ &= \sum_s f(s)\mathbb{E}_\pi\Big[\boldsymbol{\phi}_k\left(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1}\right)^\top\Big|S_k = s\Big] \qquad \text{(by (7))} \\ &= \sum_s f(s)\,\boldsymbol{\phi}(s)\left(\boldsymbol{\phi}(s) - \gamma\sum_{s'}[\mathbf{P}_\pi]_{ss'}\boldsymbol{\phi}(s')\right)^\top \\ &= \boldsymbol{\Phi}^\top\mathbf{F}(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi},\end{aligned}$$

where $\mathbf{F}$ is a diagonal matrix with diagonal elements $f(s) \doteq d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t{=}s]$, which we assume exists. As we show later, the vector $\mathbf{f} \in \mathbb{R}^N$ with components $[\mathbf{f}]_s \doteq f(s)$ can be written as

$$\begin{aligned}\mathbf{f} &= \mathbf{d}_\mu + \gamma\mathbf{P}_\pi^\top\mathbf{d}_\mu + \left(\gamma\mathbf{P}_\pi^\top\right)^2\mathbf{d}_\mu + \cdots \qquad (12)\\ &= \left(\mathbf{I} - \gamma\mathbf{P}_\pi^\top\right)^{-1}\mathbf{d}_\mu. \qquad (13)\end{aligned}$$

The key matrix is $\mathbf{F}\left(\mathbf{I} - \gamma\mathbf{P}_\pi\right)$, and the vector of its column sums is

$$\begin{aligned}\mathbf{1}^\top\mathbf{F}(\mathbf{I} - \gamma\mathbf{P}_\pi) &= \mathbf{f}^\top(\mathbf{I} - \gamma\mathbf{P}_\pi) \\ &= \mathbf{d}_\mu^\top(\mathbf{I} - \gamma\mathbf{P}_\pi)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi) \qquad \text{(using (13))} \\ &= \mathbf{d}_\mu^\top,\end{aligned}$$

all components of which are positive. Thus, the key matrix and the $\mathbf{A}$ matrix are positive definite and the algorithm is stable. Emphatic TD(0) is the simplest TD algorithm with linear function approximation proven to be stable under off-policy training.

The $\theta \to 2\theta$ example presented earlier (Figure 1) provides some insight into how replacing $\mathbf{D}_\mu$ by $\mathbf{F}$ changes the key matrix to make it positive definite. In general, $\mathbf{f}$ is the expected number of time steps that would be spent in each state during an excursion starting from the behavior distribution $\mathbf{d}_\mu$. From (12), it is $\mathbf{d}_\mu$ plus where you would get to in one step from $\mathbf{d}_\mu$, plus where you would get to in two steps, etc., with appropriate discounting. In the example, excursions under the target policy take you to the second state (2θ) and leave you there. Thus you are only in the first state (θ) if you start there, and only for one step, so $f(1) = d_\mu(1) = 0.5$. For the second state, you can either start there, with probability 0.5, or you can get there on the second step (certain except for discounting), with probability 0.9, or on the third step, with probability 0.9^2, etc, so $f(2) = 0.5 + 0.9 + 0.9^2 + 0.9^3 + \cdots = 0.5 + 0.9 \cdot 10 = 9.5$. Thus, the key matrix is now

$$\mathbf{F}(\mathbf{I} - \gamma\mathbf{P}_\pi) = \begin{bmatrix}0.5 & 0\\ 0 & 9.5\end{bmatrix} \times \begin{bmatrix}1 & -0.9\\ 0 & 0.1\end{bmatrix} = \begin{bmatrix}0.5 & -0.45\\ 0 & 0.95\end{bmatrix}.$$

Note that because $\mathbf{F}$ is a diagonal matrix, its only effect is to scale the rows. Here it emphasizes the lower row by more than a factor of 10 compared to the upper row, thereby causing the key matrix to have positive column sums and be positive definite (cf. (9)). The $\mathbf{F}$ matrix emphasizes the second state, which would occur much more often under the target policy than it does under the behavior policy.

5. The General Case

We turn now to a very general case of off-policy learning with linear function approximation. The objective is still to evaluate a policy π from a single trajectory under a different policy μ, but now the value of a state is defined not with respect to a constant discount rate $\gamma \in [0, 1]$, but with respect to a discount rate that varies from state to state according to a *discount function* $\gamma : \mathcal{S} \to [0, 1]$ such that $\prod_{k=1}^{\infty}\gamma(S_{t+k}) = 0$, w.p.1, $\forall t$. That is, our approximation is still defined by (2), but now (3) is replaced by

$$G_t \doteq R_{t+1} + \gamma(S_{t+1})R_{t+2} + \gamma(S_{t+1})\gamma(S_{t+2})R_{t+3} + \cdots. \tag{14}$$

State-dependent discounting specifies a temporal envelope within which received rewards are accumulated. If $\gamma(S_k) = 0$, then the time of accumulation is fully terminated at step $k > t$, and if $\gamma(S_k) < 1$, then it is partially terminated. We call both of these *soft termination* because they are like the termination of an episode, but the actual trajectory is not affected. Soft termination ends the accumulation of rewards into a return, but the state transitions continue oblivious to the termination. Soft termination with state-dependent termination is essential for learning models of options (Sutton et al. 1999) and other applications.

Soft termination is particularly natural in the excursion setting, where it makes it easy to define excursions of finite and definite duration. For example, consider the deterministic MDP shown in Figure 2. There are five states, three of which do not discount at all, $\gamma(s) = 1$, and are shown as circles, and two of which cause complete soft termination, $\gamma(s) = 0$, and are shown as squares. The terminating states do not end anything other

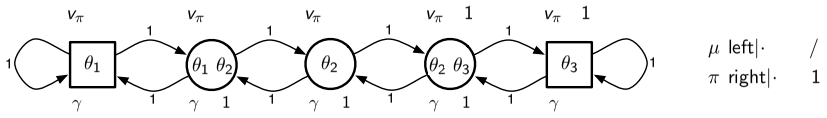


Figure 2: A 5-state chain MDP with soft-termination states at each end.

than the return; actions are still selected in them and, dependent on the action selected, they transition to next states indefinitely without end. In this MDP there are two actions, left and right, which deterministically cause transitions to the left or right except at the edges, where there may be a self transition. The reward on all transitions is +1. The behavior policy is to select left 2/3rds of the time in all states, which causes more time to be spent in states on the left than on the right. The stationary distribution can be shown to be $\mathbf{d}_\mu \approx (0.52, 0.26, 0.13, 0.06, 0.03)^\top$; more than half of the time steps are spent in the leftmost terminating state.

Consider the target policy π that selects the right action from all states. The correct value $v_\pi(s)$ of each state s is written above it in the figure. For both of the two rightmost states, the right action results in a reward of 1 and an immediate termination, so their values are both 1. For the middle state, following π (selecting right repeatedly) yields two rewards of 1 prior to termination. There is no discounting ($\gamma=1$) prior to termination, so the middle state's value is 2, and similarly the values go up by 1 for each state to its left, as shown. These are the correct values. The approximate values depend on the parameter vector $\boldsymbol{\theta}_t$ as suggested by the expressions shown inside each state in the figure. These expressions use the notation θ_i to denote the ith *component* of the current parameter vector $\boldsymbol{\theta}_t$. In this example, there are five states and only three parameters, so it is unlikely, and indeed impossible, to represent v_π exactly. We will return to this example later in the paper.

In addition to enabling definitive termination, as in this example, state-dependent discounting enables a much wider range of predictive questions to be expressed in the form of a value function (Sutton et al. 2011, Modayil, White & Sutton 2014, Sutton, Rafols & Koop 2006), including option models (Sutton, Precup & Singh 1999, Sutton 1995). For example, with state-dependent discounting one can formulate questions both about what will happen during a way of behaving and what will be true at its end. A general representation for predictions is a key step toward the goal of representing world knowledge in verifiable predictive terms (Sutton 2009, 2012). The general form is also useful just because it enables us to treat uniformly many of the most important episodic and continuing special cases of interest.

A second generalization, developed for the first time in this paper, is to explicitly specify the states at which we are most interested is obtaining accurate estimates of value. Recall that in parametric function approximation there are typically many more states than parameters ($N \gg n$), and thus it is usually not possible for the value estimates at all states to be exactly correct. Valuing some states more accurately usually means valuing others less accurately, at least asymptotically. In the tabular case where much of the theory of reinforcement learning originated, this tradeoff is not an issue because the estimates of each state are independent of each other, but with function approximation it is necessary to specify relative interest in order to make the problem well defined. Nevertheless, in the function approximation case little attention has been paid in the literature to specifing the relative importance of different states (an exception is Thomas 2014), though there are intimations of this in the initiation set of options (Sutton et al. 1999). In the past it was typically assumed that we were interested in valuing states in direct proportion to how often they occur, but this is not always the case. For example, in episodic problems we often care primarily about the value of the first state, or of earlier states generally (Thomas 2014). Here we allow the user to specify the relative interest in each state with a nonnegative *interest function* $i : \mathcal{S} \to [0, \infty)$. Formally, our objective is to minimize the Mean Square Value Error (MSVE) with states weighted both by how often they occur and by our interest in them:

$$\text{MSVE}(\boldsymbol{\theta}) \doteq \sum_{s\in\mathcal{S}} d_\mu(s) i(s) \Big(v_\pi(s) - \boldsymbol{\theta}^\top \boldsymbol{\phi}(s) \Big)^2. \tag{15}$$

For example, in the 5-state example in Figure 2, we could choose $i(s) = 1, \forall s \in \mathcal{S}$, in which case we would be primarily interested in attaining low error in the states on the left side, which are visited much more often under the behavior policy. If we want to counter this, we might chose $i(s)$ larger for states toward the right. Of course, with parametric function approximation we presumably do not have access to the states as individuals, but certainly we could set $i(s)$ as a function of the features in s. In this example, choosing $i(s) = 1 + \phi_2(s) + 2\phi_3(s)$ (where $\phi_i(s)$ denotes the ith component of $\boldsymbol{\phi}(s)$) would shift the focus on accuracy to the states on the right, making it substantially more balanced.

The third and final generalization that we introduce in this section is general bootstrapping. Conventional TD(λ) uses a bootstrapping parameter $\lambda \in [0, 1]$; we generalize this to a *bootstrapping function* $\lambda : \mathcal{S} \to [0, 1]$ specifying a potentially different degree of bootstrapping, $1 - \lambda(s)$, for each state s. General bootstrapping of this form has been partially developed in several previous works (Sutton 1995, Sutton & Barto 1998, Maei & Sutton 2010, Sutton et al. 2014, cf. Yu 2012). As a notational shorthand, let us use $\lambda_t \doteq \lambda(S_t)$ and $\gamma_t \doteq \gamma(S_t)$. Then we can define a general notion of bootstrapped return, the λ-return with state-dependent bootstrapping and discounting:

$$G_t^\lambda \doteq R_{t+1} + \gamma_{t+1} \Big[(1 - \lambda_{t+1}) \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} + \lambda_{t+1} G_{t+1}^\lambda \Big]. \tag{16}$$

The λ-return plays a key role in the theoretical understanding of TD methods, in particular, in their forward views (Sutton & Barto 1998, Sutton, Mahmood, Precup & van Hasselt 2014). In the forward view, G_t^λ is thought of as the target for the update at time t, even though it is not available until many steps later (when complete termination $\gamma(S_k) = 0$ has occurred for the first time for some $k > t$).

Given these generalizations, we can now specify our final new algorithm, *emphatic TD(λ)*, by the following four equations, for $t \geq 0$:

$$\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t + \alpha \Big(R_{t+1} + \gamma_{t+1} \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t \Big) \mathbf{e}_t \tag{17}$$

$$\mathbf{e}_t \doteq \rho_t \left(\gamma_t \lambda_t \mathbf{e}_{t-1} + M_t \boldsymbol{\phi}_t \right), \qquad \text{with } \mathbf{e}_{-1} \doteq \mathbf{0} \tag{18}$$

$$M_t \doteq \lambda_t \, i(S_t) + (1 - \lambda_t) F_t \tag{19}$$

$$F_t \doteq \rho_{t-1} \gamma_t F_{t-1} + i(S_t), \qquad \text{with } F_0 \doteq i(S_0), \tag{20}$$

where $F_t \geq 0$ is a scalar memory called the *followon trace*. The quantity $M_t \geq 0$ is termed the *emphasis* on step t. Note that, if desired, M_t can be removed from the algorithm by substituting its definition (19) into (18).

6. Off-policy Stability of Emphatic TD(λ)

As usual, to analyze the stability of the new algorithm we examine its $\mathbf{A}$ matrix. The stochastic update can be written:

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &\doteq \boldsymbol{\theta}_t + \alpha\left(R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right)\mathbf{e}_t \\ &= \boldsymbol{\theta}_t + \alpha\Big(\underbrace{\mathbf{e}_t R_{t+1}}_{\mathbf{b}_t} - \underbrace{\mathbf{e}_t\left(\boldsymbol{\phi}_t - \gamma_{t+1}\boldsymbol{\phi}_{t+1}\right)^\top}_{\mathbf{A}_t}\boldsymbol{\theta}_t\Big).\end{aligned}$$

Thus,

$$\begin{aligned}\mathbf{A} = \lim_{t\to\infty}\mathbb{E}[\mathbf{A}_t] &= \lim_{t\to\infty}\mathbb{E}_\mu\left[\mathbf{e}_t\left(\boldsymbol{\phi}_t - \gamma_{t+1}\boldsymbol{\phi}_{t+1}\right)^\top\right] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu\left[\mathbf{e}_t\left(\boldsymbol{\phi}_t - \gamma_{t+1}\boldsymbol{\phi}_{t+1}\right)^\top\Big|S_t{=}s\right] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu\left[\rho_t\left(\gamma_t\lambda_t\mathbf{e}_{t-1} + M_t\boldsymbol{\phi}_t\right)\left(\boldsymbol{\phi}_t - \gamma_{t+1}\boldsymbol{\phi}_{t+1}\right)^\top\Big|S_t{=}s\right] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[(\gamma_t\lambda_t\mathbf{e}_{t-1} + M_t\boldsymbol{\phi}_t)|S_t{=}s]\,\mathbb{E}_\mu\left[\rho_t(\boldsymbol{\phi}_t - \gamma_{t+1}\boldsymbol{\phi}_{t+1})^\top\Big|S_t{=}s\right]\end{aligned}$$

(because, given S_t, $\mathbf{e}_{t-1}$ and M_t are independent of $\rho_t(\boldsymbol{\phi}_t - \gamma_{t+1}\boldsymbol{\phi}_{t+1})^\top$)

$$\begin{aligned}&= \sum_s \underbrace{d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[(\gamma_t\lambda_t\mathbf{e}_{t-1} + M_t\boldsymbol{\phi}_t)|S_t{=}s]}_{\mathbf{e}(s)\in\mathbb{R}^n}\mathbb{E}_\mu\left[\rho_k(\boldsymbol{\phi}_k - \gamma_{k+1}\boldsymbol{\phi}_{k+1})^\top\Big|S_k{=}s\right] \\ &= \sum_s \mathbf{e}(s)\mathbb{E}_\pi[\boldsymbol{\phi}_k - \gamma_{k+1}\boldsymbol{\phi}_{k+1}|S_k{=}s]^\top \qquad \text{(by (7))} \\ &= \sum_s \mathbf{e}(s)\left(\boldsymbol{\phi}(s) - \sum_{s'}[\mathbf{P}_\pi]_{ss'}\gamma(s')\boldsymbol{\phi}(s')\right)^\top \\ &= \mathbf{E}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma})\boldsymbol{\Phi}, \qquad (21)\end{aligned}$$

where $\mathbf{E}$ is an $N \times n$ matrix $\mathbf{E}^\top \doteq [\mathbf{e}(1), \cdots, \mathbf{e}(N)]$, and $\mathbf{e}(s) \in \mathbb{R}^n$ is defined by[5]:

$$\begin{aligned}\mathbf{e}(s) &\doteq d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[\gamma_t\lambda_t\mathbf{e}_{t-1} + M_t\boldsymbol{\phi}_t|S_t{=}s] \qquad \text{(assuming this exists)} \\ &= \underbrace{d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[M_t|S_t{=}s]}_{m(s)}\boldsymbol{\phi}(s) + \gamma(s)\lambda(s)d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{e}_{t-1}|S_t{=}s] \\ &= m(s)\boldsymbol{\phi}(s) + \gamma(s)\lambda(s)d_\mu(s)\lim_{t\to\infty}\sum_{\bar{s},\bar{a}}\mathbb{P}\{S_{t-1}{=}\bar{s}, A_{t-1}{=}\bar{a}|S_t{=}s\}\,\mathbb{E}_\mu[\mathbf{e}_{t-1}|S_{t-1}{=}\bar{s}, A_{t-1}{=}\bar{a}]\end{aligned}$$

5. Note that this is a slight abuse of notation; $\mathbf{e}_t$ is a vector random variable, one per time step, and $\mathbf{e}(s)$ is a vector expectation, one per state.

$$\begin{aligned}&= m(s)\boldsymbol{\phi}(s) + \gamma(s)\lambda(s)d_\mu(s)\sum_{\bar{s},\bar{a}}\frac{d_\mu(\bar{s})\mu(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})}{d_\mu(s)}\lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{e}_{t-1}|S_{t-1}{=}\bar{s}, A_{t-1}{=}\bar{a}]\end{aligned}$$

(using the definition of a conditional probability, a.k.a. Bayes rule)

$$\begin{aligned}&= m(s)\boldsymbol{\phi}(s) + \gamma(s)\lambda(s)\sum_{\bar{s},\bar{a}}d_\mu(\bar{s})\mu(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\lim_{t\to\infty}\mathbb{E}_\mu[\gamma_{t-1}\lambda_{t-1}\mathbf{e}_{t-2} + M_{t-1}\boldsymbol{\phi}_{t-1}|S_{t-1}{=}\bar{s}] \\ &= m(s)\boldsymbol{\phi}(s) + \gamma(s)\lambda(s)\sum_{\bar{s}}\left(\sum_{\bar{a}}\pi(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})\right)\mathbf{e}(\bar{s}) \\ &= m(s)\boldsymbol{\phi}(s) + \gamma(s)\lambda(s)\sum_{\bar{s}}[\mathbf{P}_\pi]_{\bar{s}s}\mathbf{e}(\bar{s}).\end{aligned}$$

We now introduce three $N \times N$ diagonal matrices: $\mathbf{M}$, which has the $m(s) \doteq d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[M_t|S_t{=}s]$ on its diagonal; $\boldsymbol{\Gamma}$, which has the $\gamma(s)$ on its diagonal; and $\boldsymbol{\Lambda}$, which has the $\lambda(s)$ on its diagonal. With these we can write the equation above entirely in matrix form, as

$$\begin{aligned}\mathbf{E}^\top &= \boldsymbol{\Phi}^\top\mathbf{M} + \mathbf{E}^\top\mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda} \\ &= \boldsymbol{\Phi}^\top\mathbf{M} + \boldsymbol{\Phi}^\top\mathbf{M}\mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda} + \boldsymbol{\Phi}^\top\mathbf{M}(\mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^2 + \cdots \\ &= \boldsymbol{\Phi}^\top\mathbf{M}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^{-1}.\end{aligned}$$

Finally, combining this equation with (21) we obtain

$$\mathbf{A} = \boldsymbol{\Phi}^\top\mathbf{M}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^{-1}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma})\boldsymbol{\Phi}, \qquad (22)$$

and through similar steps one can also obtain emphatic TD(λ)'s $\mathbf{b}$ vector,

$$\mathbf{b} = \mathbf{E}\mathbf{r}_\pi = \boldsymbol{\Phi}^\top\mathbf{M}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^{-1}\mathbf{r}_\pi, \qquad (23)$$

where $\mathbf{r}_\pi$ is the N-vector of expected immediate rewards from each state under π.

Emphatic TD(λ)'s key matrix, then, is $\mathbf{M}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^{-1}(\mathbf{I} - \mathbf{P}_\pi\boldsymbol{\Gamma})$. To prove that it is positive definite we will follow the same strategy as we did for emphatic TD(0). The first step will be to write the last part of the key matrix in the form of the identity matrix minus a probability matrix. To see how this can be done, consider a slightly different setting in which actions are taken according to π, and in which $1 - \gamma(s)$ and $1 - \lambda(s)$ are considered probabilities of ending by terminating and by bootstrapping, respectively. That is, for any starting state, a trajectory involves a state transition according to $\mathbf{P}_\pi$, possibly terminating according to $\mathbf{I} - \boldsymbol{\Gamma}$, then possibly ending with a bootstrapping event according to $\mathbf{I} - \boldsymbol{\Lambda}$, and then, if neither of these occur, continuing with another state transition and more chances to end, and so on until an ending of one of the two kinds occurs. For any start state $i \in \mathcal{S}$, consider the probability that the trajectory ends in state $j \in \mathcal{S}$ with an ending event of the bootstrapping kind (according to $\mathbf{I} - \boldsymbol{\Lambda}$). Let $\mathbf{P}_\pi^\lambda$ be the matrix with this probability as its

ijth component. This matrix can be written

$$\begin{aligned}
\mathbf{P}_\pi^\lambda &= \mathbf{P}_\pi\mathbf{\Gamma}(\mathbf{I}-\mathbf{\Lambda}) + \mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda}\mathbf{P}_\pi\mathbf{\Gamma}(\mathbf{I}-\mathbf{\Lambda}) + \mathbf{P}_\pi\mathbf{\Gamma}(\mathbf{\Lambda}\mathbf{P}_\pi\mathbf{\Gamma})^2(\mathbf{I}-\mathbf{\Lambda}) + \cdots \\
&= \left(\sum_{k=0}^{\infty}(\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^k\right)\mathbf{P}_\pi\mathbf{\Gamma}(\mathbf{I}-\mathbf{\Lambda}) \\
&= (\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^{-1}\mathbf{P}_\pi\mathbf{\Gamma}(\mathbf{I}-\mathbf{\Lambda}). \\
&= (\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^{-1}(\mathbf{P}_\pi\mathbf{\Gamma}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda}) \\
&= (\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^{-1}(\mathbf{P}_\pi\mathbf{\Gamma}-\mathbf{I}+\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda}) \\
&= \mathbf{I}-(\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^{-1}(\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}),
\end{aligned}$$

or

$$\mathbf{I}-\mathbf{P}_\pi^\lambda = (\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^{-1}(\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}). \tag{24}$$

It follows then that $\mathbf{M}(\mathbf{I}-\mathbf{P}_\pi^\lambda) = \mathbf{M}(\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma}\mathbf{\Lambda})^{-1}(\mathbf{I}-\mathbf{P}_\pi\mathbf{\Gamma})$ is another way of writing emphatic TD(λ)'s key matrix (cf. (22)). This gets us considerably closer to our goal of proving that the key matrix is positive definite. It is now immediate that its diagonal entries are nonnegative and that its off diagonal entries are nonpositive. It is also immediate that its row sums are nonnegative.

There remains what is typically the hardest condition to satisfy: that the column sums of the key matrix are positive. To show this we have to analyze $\mathbf{M}$, and to do that we first analyze the N-vector $\mathbf{f}$ with components $f(s) \doteq d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t{=}s]$ (we assume that this limit and expectation exist). Analyzing $\mathbf{f}$ will also pay the debt we incurred in Section 4 when we claimed without proof that $\mathbf{f}$ (in the special case treated in that section) was as given by (13). In the general case:

$$\begin{aligned}
f(s) &= d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t{=}s] \\
&= d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[i(S_t)+\rho_{t-1}\gamma_t F_{t-1}|S_t{=}s] && \text{(by (20))} \\
&= d_\mu(s)i(s) + d_\mu(s)\gamma(s)\lim_{t\to\infty}\sum_{\bar s,\bar a}\mathbb{P}\{S_{t-1}{=}\bar s, A_{t-1}{=}\bar a|S_t{=}s\}\frac{\pi(\bar a|\bar s)}{\mu(\bar a|\bar s)}\mathbb{E}_\mu[F_{t-1}|S_{t-1}{=}\bar s] \\
&= d_\mu(s)i(s) + d_\mu(s)\gamma(s)\sum_{\bar s,\bar a}\frac{d_\mu(\bar s)\mu(\bar a|\bar s)p(s|\bar s,\bar a)}{d_\mu(s)}\frac{\pi(\bar a|\bar s)}{\mu(\bar a|\bar s)}\lim_{t\to\infty}\mathbb{E}_\mu[F_{t-1}|S_{t-1}{=}\bar s] \\
&\quad \text{(using the definition of a conditional probability, a.k.a. Bayes rule)} \\
&= d_\mu(s)i(s) + \gamma(s)\sum_{\bar s,\bar a}\pi(\bar a|\bar s)p(s|\bar s,\bar a)d_\mu(\bar s)\lim_{t\to\infty}\mathbb{E}_\mu[F_{t-1}|S_{t-1}{=}\bar s] \\
&= d_\mu(s)i(s) + \gamma(s)\sum_{\bar s}[\mathbf{P}_\pi]_{\bar s s}f(\bar s).
\end{aligned}$$

This equation can be written in matrix-vector form, letting $\mathbf{i}$ be the N-vector with components $[\mathbf{i}]_s \doteq d_\mu(s)i(s)$:

$$\begin{aligned}
\mathbf{f} &= \mathbf{i} + \mathbf{\Gamma}\mathbf{P}_\pi^\top\mathbf{f} \\
&= \mathbf{i} + \mathbf{\Gamma}\mathbf{P}_\pi^\top\mathbf{i} + (\mathbf{\Gamma}\mathbf{P}_\pi^\top)^2\mathbf{i} + \cdots \\
&= \left(\mathbf{I}-\mathbf{\Gamma}\mathbf{P}_\pi^\top\right)^{-1}\mathbf{i}.
\end{aligned} \tag{25}$$

This proves (13), because there $i(s) \doteq 1, \forall s$ (thus $\mathbf{i} = \mathbf{d}_\mu$), and $\gamma(s) \doteq \gamma, \forall s$.

We are now ready to analyze $\mathbf{M}$, the diagonal matrix with the $m(s)$ on its diagonal:

$$\begin{aligned}
m(s) &= d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[M_t|S_t{=}s] \\
&= d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[\lambda_t\, i(S_t) + (1-\lambda_t)F_t|S_t{=}s] && \text{(by (19))} \\
&= d_\mu(s)\lambda(s)i(s) + (1-\lambda(s))\, f(s),
\end{aligned}$$

or, in matrix-vector form, letting $\mathbf{m}$ be the N-vector with components $m(s)$,

$$\begin{aligned}
\mathbf{m} &= \mathbf{\Lambda}\mathbf{i} + (\mathbf{I}-\mathbf{\Lambda})\mathbf{f} \\
&= \mathbf{\Lambda}\mathbf{i} + (\mathbf{I}-\mathbf{\Lambda})\left(\mathbf{I}-\mathbf{\Gamma}\mathbf{P}_\pi^\top\right)^{-1}\mathbf{i} && \text{(using (25))} \\
&= \left[\mathbf{\Lambda}(\mathbf{I}-\mathbf{\Gamma}\mathbf{P}_\pi^\top) + (\mathbf{I}-\mathbf{\Lambda})\right](\mathbf{I}-\mathbf{\Gamma}\mathbf{P}_\pi^\top)^{-1}\mathbf{i} \\
&= \left(\mathbf{I}-\mathbf{\Lambda}\mathbf{\Gamma}\mathbf{P}_\pi^\top\right)\left(\mathbf{I}-\mathbf{\Gamma}\mathbf{P}_\pi^\top\right)^{-1}\mathbf{i} && (26) \\
&= \left(\mathbf{I}-\mathbf{P}_\pi^{\lambda\top}\right)^{-1}\mathbf{i}. && \text{(using (24))}
\end{aligned}$$

Now we are ready for the final step of the proof, showing that all the columns of the key matrix $\mathbf{M}(\mathbf{I}-\mathbf{P}_\pi^\lambda)$ sum to a positive number. Using the result above, the vector of column sums is

$$\begin{aligned}
\mathbf{1}^\top\mathbf{M}(\mathbf{I}-\mathbf{P}_\pi^\lambda) &= \mathbf{m}^\top(\mathbf{I}-\mathbf{P}_\pi^\lambda) \\
&= \mathbf{i}^\top(\mathbf{I}-\mathbf{P}_\pi^\lambda)^{-1}(\mathbf{I}-\mathbf{P}_\pi^\lambda) \\
&= \mathbf{i}^\top.
\end{aligned}$$

If we further assume that $i(s) > 0, \forall s \in \mathcal{S}$, then the column sums are all positive, the key matrix is positive definite, and emphatic TD(λ) and its expected update are stable. This result can be summarized in the following theorem, the main result of this paper, which we have just proved:

Theorem 1 (Stability of Emphatic TD(λ)) *For any*

- *Markov decision process* $\{S_t, A_t, R_{t+1}\}_{t=0}^\infty$ *with finite state and actions sets* $\mathcal{S}$ *and* $\mathcal{A}$,
- *behavior policy* μ *with a stationary invariant distribution* $d_\mu(s) > 0, \forall s \in \mathcal{S}$,
- *target policy* π *with coverage, i.e., s.t., if* $\pi(a|s) > 0$, *then* $\mu(a|s) > 0$,
- *discount function* $\gamma : \mathcal{S} \to [0,1]$ *s.t.* $\prod_{k=1}^\infty \gamma(S_{t+k}) = 0, w.p.1, \forall t > 0$,
- *bootstrapping function* $\lambda : \mathcal{S} \to [0,1]$,
- *interest function* $i : \mathcal{S} \to (0,\infty)$,
- *feature function* $\boldsymbol{\phi} : \mathcal{S} \to \mathbb{R}^n$ *s.t. the matrix* $\mathbf{\Phi} \in \mathbb{R}^{|\mathcal{S}|\times n}$ *with the* $\boldsymbol{\phi}(s)$ *as its rows has linearly independent columns,*

the $\mathbf{A}$ *matrix of linear emphatic TD(*λ*) (as given by (17–20), and assuming the existence of* $\lim_{t\to\infty}\mathbb{E}[F_t|S_t=s]$ *and* $\lim_{t\to\infty}\mathbb{E}[\mathbf{e}_t|S_t=s]$*,* $\forall s\in\mathcal{S}$*),*

$$\mathbf{A}=\lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{A}_t]=\lim_{t\to\infty}\mathbb{E}_\mu\Big[\mathbf{e}_t\left(\boldsymbol{\phi}_t-\gamma_{t+1}\boldsymbol{\phi}_{t+1}\right)^\top\Big]=\boldsymbol{\Phi}^\top\mathbf{M}(\mathbf{I}-\mathbf{P}_\pi^\lambda)\boldsymbol{\Phi}, \tag{27}$$

is positive definite. Thus the algorithm and its expected update are stable.

As mentioned at the outset, stability is necessary but not always sufficient to guarantee convergence of the parameter vector $\boldsymbol{\theta}_t$. Yu (2015a,b) has recently built on our stability result to show that in fact emphatic TD(λ) converges with probability one under a variety of step-size conditions. Convergence as anticipated is to the unique fixed point $\bar{\boldsymbol{\theta}}$ of the deterministic algorithm (6), in other words, to

$$\mathbf{A}\bar{\boldsymbol{\theta}}=\mathbf{b}\quad\text{ or }\quad\bar{\boldsymbol{\theta}}=\mathbf{A}^{-1}\mathbf{b}. \tag{28}$$

This solution can be characterized as a minimum (in fact, a zero) of the Projected Bellman Error (PBE, Sutton et al. 2009) using the λ-dependent Bellman operator $T^{(\lambda)}:\mathbb{R}^N\to\mathbb{R}^N$ (Tsitiklis & Van Roy 1997) and the weighting of states according to their emphasis. For our general case, we need a version of the $T^{(\lambda)}$ operator extended to state-dependent discounting and bootstrapping. This operator looks ahead to future states to the extent that they are bootstrapped from, that is, according to $\mathbf{P}_\pi^\lambda$, taking into account the reward received along the way. The appropriate operator, in vector form, is

$$T^{(\lambda)}\mathbf{v}\doteq(\mathbf{I}-\mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^{-1}\mathbf{r}_\pi+\mathbf{P}_\pi^\lambda\mathbf{v}. \tag{29}$$

This operator is a contraction with fixed point $\mathbf{v}=v_\pi$. Recall that our approximate value function is $\boldsymbol{\Phi}\boldsymbol{\theta}$, and thus the difference between $\boldsymbol{\Phi}\boldsymbol{\theta}$ and $T^{(\lambda)}(\boldsymbol{\Phi}\boldsymbol{\theta})$ is a Bellman-error vector. The projection of this with respect to the feature matrix and the emphasis weighting is the emphasis-weighted PBE:

$$\begin{aligned}
\text{PBE}(\boldsymbol{\theta})&\doteq\Pi\left(T^{(\lambda)}(\boldsymbol{\Phi}\boldsymbol{\theta})-\boldsymbol{\Phi}\boldsymbol{\theta}\right)\\
&\doteq\boldsymbol{\Phi}(\boldsymbol{\Phi}^\top\mathbf{M}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\top\mathbf{M}\left(T^{(\lambda)}(\boldsymbol{\Phi}\boldsymbol{\theta})-\boldsymbol{\Phi}\boldsymbol{\theta}\right) && \text{(see Sutton et al. 2009)}\\
&=\boldsymbol{\Phi}(\boldsymbol{\Phi}^\top\mathbf{M}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\top\mathbf{M}\left((\mathbf{I}-\mathbf{P}_\pi\boldsymbol{\Gamma}\boldsymbol{\Lambda})^{-1}\mathbf{r}_\pi+\mathbf{P}_\pi^\lambda\boldsymbol{\Phi}\boldsymbol{\theta}-\boldsymbol{\Phi}\boldsymbol{\theta}\right) && \text{(by (29))}\\
&=\boldsymbol{\Phi}(\boldsymbol{\Phi}^\top\mathbf{M}\boldsymbol{\Phi})^{-1}\left(\mathbf{b}+\boldsymbol{\Phi}^\top\mathbf{M}(\mathbf{P}_\pi^\lambda-\mathbf{I})\boldsymbol{\Phi}\boldsymbol{\theta}\right) && \text{(by (23))}\\
&=\boldsymbol{\Phi}(\boldsymbol{\Phi}^\top\mathbf{M}\boldsymbol{\Phi})^{-1}\left(\mathbf{b}-\mathbf{A}\boldsymbol{\theta}\right). && \text{(by (27))}
\end{aligned}$$

From (28), it is immediate that this is vector zero at the fixed point $\bar{\boldsymbol{\theta}}$, thus $\text{PBE}(\bar{\boldsymbol{\theta}})=\mathbf{0}$.

7. Derivation of the Emphasis Algorithm

Emphatic algorithms are based on the idea that if we are updating a state by a TD method, then we should also update each state that it bootstraps from, in direct proportion. For example, suppose we decide to update the estimate at time t with unit emphasis, perhaps because $i(S_t)=1$, and then at time $t+1$ we have $\gamma(S_{t+1})=1$ and $\lambda(S_{t+1})=0$. Because of

the latter, we are fully bootstrapping from the value estimate at $t+1$ and thus we should also make an update of it with emphasis equal to t's emphasis. If instead $\lambda(S_{t+1})=0.5$, then the update of the estimate at $t+1$ would gain a half unit of emphasis, and the remaining half would still be available to allocate to the updates of the estimate at $t+2$ or later times depending on their λs. And of course there may be some emphasis allocated directly updating the estimate at $t+1$ if $i(S_{t+1})>0$. Discounting and importance sampling also have effects. At each step t, if $\gamma(S_t)<1$, then there is some degree of termination and to that extent there is no longer any chance of bootstrapping from later time steps. Another way bootstrapping may be cut off is if $\rho_t=0$ (a complete deviation from the target policy). More generally, if $\rho\neq1$, then the opportunity for bootstrapping is scaled up or down proportionally.

It may seem difficult to work out precisely how each time step's estimates bootstrap from which later states' estimates for all cases. Fortunately, it has already been done. Equation 6 of the paper by Sutton, Mahmood, Precup, and van Hasselt (2014) specifies this in their "forward view" of off-policy TD(λ) with general state-dependent discounting and bootstrapping. From this equation (and their (5)) it is easy to determine the degree to which the update of the value estimate at time k bootstraps from (multiplicatively depends on) the value estimates of each subsequent time t. It is

$$\rho_k\left(\prod_{i=k+1}^{t-1}\gamma_i\lambda_i\rho_i\right)\gamma_t(1-\lambda_t).$$

It follows then that the total emphasis on time t, M_t, should be the sum of this quantity for all times $k<t$, each times the emphasis M_k for those earlier times, plus any intrinsic interest $i(S_t)$ in time t:

$$\begin{aligned}
M_t&\doteq i(S_t)+\sum_{k=0}^{t-1}M_k\rho_k\left(\prod_{i=k+1}^{t-1}\gamma_i\lambda_i\rho_i\right)\gamma_t(1-\lambda_t)\\
&=\lambda_t i(S_t)+(1-\lambda_t)i(S_t)+(1-\lambda_t)\gamma_t\sum_{k=0}^{t-1}\rho_k M_k\prod_{i=k+1}^{t-1}\gamma_i\lambda_i\rho_i\\
&=\lambda_t i(S_t)+(1-\lambda_t)F_t,
\end{aligned}$$

which is (19), where

$$\begin{aligned}
F_t&\doteq i(S_t)+\gamma_t\sum_{k=0}^{t-1}\rho_k M_k\prod_{i=k+1}^{t-1}\gamma_i\lambda_i\rho_i\\
&=i(S_t)+\gamma_t\left(\rho_{t-1}M_{t-1}+\sum_{k=0}^{t-2}\rho_k M_k\prod_{i=k+1}^{t-1}\gamma_i\lambda_i\rho_i\right)\\
&=i(S_t)+\gamma_t\left(\rho_{t-1}M_{t-1}+\rho_{t-1}\lambda_{t-1}\gamma_{t-1}\sum_{k=0}^{t-2}\rho_k M_k\prod_{i=k+1}^{t-2}\gamma_i\lambda_i\rho_i\right)\\
&=i(S_t)+\gamma_t\rho_{t-1}\Bigg(\underbrace{\lambda_{t-1}i(S_{t-1})+(1-\lambda_{t-1})F_{t-1}}_{M_{t-1}}+\lambda_{t-1}\gamma_{t-1}\sum_{k=0}^{t-2}\rho_k M_k\prod_{i=k+1}^{t-2}\gamma_i\lambda_i\rho_i\Bigg)
\end{aligned}$$

$$= i(S_t) + \gamma_t \rho_{t-1} \Bigg(F_{t-1} + \lambda_{t-1} \Bigg(-F_{t-1} + \underbrace{i(S_{t-1}) + \gamma_{t-1} \sum_{k=0}^{t-2} \rho_k M_k \prod_{i=k+1}^{t-2} \gamma_i \lambda_i \rho_i}_{F_{t-1}} \Bigg) \Bigg)$$

$$= i(S_t) + \gamma_t \rho_{t-1} F_{t-1},$$

which is (20), completing the derivation of the emphasis algorithm.

8. Empirical Examples

In this section we present empirical results with example problems that verify and elucidate the formal results already presented. A thorough empirical comparison of emphatic TD(λ) with other methods is beyond the scope of the present article.

The main focus in this paper, as in much previous theory of TD algorithms with function approximation, has been on the stability of the expected update. If an algorithm is unstable, as Q-learning and off-policy TD(λ) are on Baird's (1995) counterexample, then there is no chance of its behaving in a satisfactory manner. On the other hand, even if the update is stable it may be of very high variance. Off-policy algorithms involve products of potentially an infinite number of importance-sampling ratios, which can lead to fluxuations of infinite variance.

As an example of what can happen, let's look again at the $\theta \to 2\theta$ problem shown in Figure 1 (and shown again in the upper left of Figure 3). Consider what happens to F_t in this problem if we have interest only in the first state, and the right action happens to be taken on every step (i.e., $i(S_0) = 1$ then $i(S_t) = 0, \forall t > 0$, and $A_t = \text{right}, \forall t \geq 0$). In this case, from (20),

$$F_t = \rho_{t-1}\gamma_t F_{t-1} + i(S_t) = \prod_{j=0}^{t-1} \rho_j \gamma = (2 \cdot 0.9)^t,$$

which of course goes to infinity as $t \to \infty$. On the other hand, the probability of this specific infinite action sequence is zero, and in fact F_t will rarely take on very high values. In particular, the expected value of F_t remains finite at

$$\begin{aligned} \mathbb{E}_\mu[F_t] &= 0.5 \cdot 2 \cdot 0.9 \cdot \mathbb{E}_\mu[F_{t-1}] + 0.5 \cdot 0 \cdot 0.9 \cdot \mathbb{E}_\mu[F_{t-1}] \\ &= 0.9 \cdot \mathbb{E}_\mu[F_{t-1}] \\ &= 0.9^t, \end{aligned}$$

which tends to zero as $t \to \infty$. Nevertheless, this problem is indeed a difficult case, as the *variance* of F_t *is* infinite:

$$\begin{aligned} \text{Var}[F_t] &= \mathbb{E}\big[F_t^2\big] - (\mathbb{E}[F_t])^2 \\ &= 0.5^t (2^t 0.9^t)^2 - (0.9^t)^2 \\ &= (0.9^2 \cdot 2)^t - (0.9^2)^t \\ &= 1.62^t - 0.81^t, \end{aligned}$$

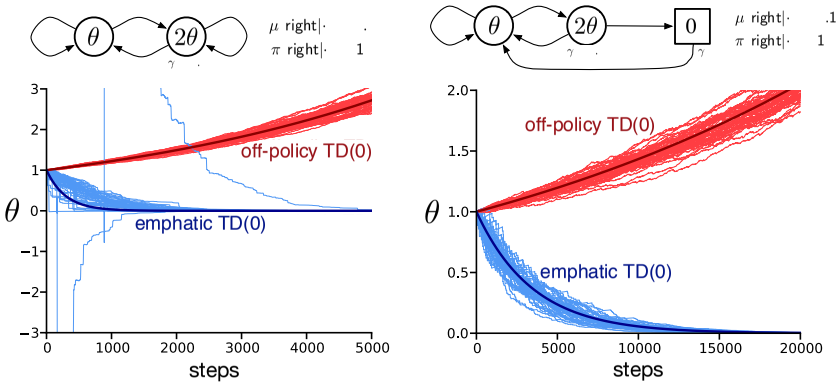


Figure 3: Emphatic TD approaches the correct value of zero, whereas conventional off-policy TD diverges, on fifty trajectories on the $\theta \to 2\theta$ problems shown above each graph. Also shown as a thick line is the trajectory of the deterministic expected-update algorithm (6). On the continuing problem (left) emphatic TD had occasional high variance deviations from zero.

which tends to ∞ as $t \to \infty$.

What does actually happen on this problem? The thin blue lines in Figure 3 (left) show the trajectories of the single parameter θ over time in 50 runs with this problem with $\lambda = 0$ and $\alpha = 0.001$, starting at $\theta = 1.0$. We see that most trajectories of emphatic TD(0) rapidly approached the correct value of $\theta = 0$, but a few made very large steps away from zero and then returned. Because the variance of F_t (and thus of M_t and $\mathbf{e}_t$) grows to infinity as t tends to infinity, there is always a small chance of an extremely large fluxuation taking θ far away from zero. Off-policy TD(0), on the other hand, diverged to infinity in all individual runs.

For comparison, Figure 3 (right) shows trajectories for a $\theta \to 2\theta$ problem in which F_t and all the other variables and their variances are bounded. In this problem, the target policy of selecting right on all steps leads to a soft terminal state ($\gamma(s) = 0$) with fixed value zero, which then transitions back to start again in the leftmost state, as shown in the upper right of the figure. (This is an example of how one can reproduce the conventional notions of terminal state and episode in a soft termination setting.) Here we have chosen the behavior policy to take the action left with probability 0.9, so that its stationary distribution distinctly favors the left state, whereas the target policy would spend equal time in each of the two states. This change increases the variance of the updates, so we used a smaller step size, $\alpha = 0.0001$; other settings were unchanged. Conventional off-policy TD(0) still diverged in this case, but emphatic TD(0) converged reliably to zero.

Finally, Figure 4 shows trajectories for the 5-state example shown earlier (and again in the upper part of the figure). In this case, everything is bounded under the target policy, and both algorithms converged. The emphatic algorithm achieved a lower MSVE in

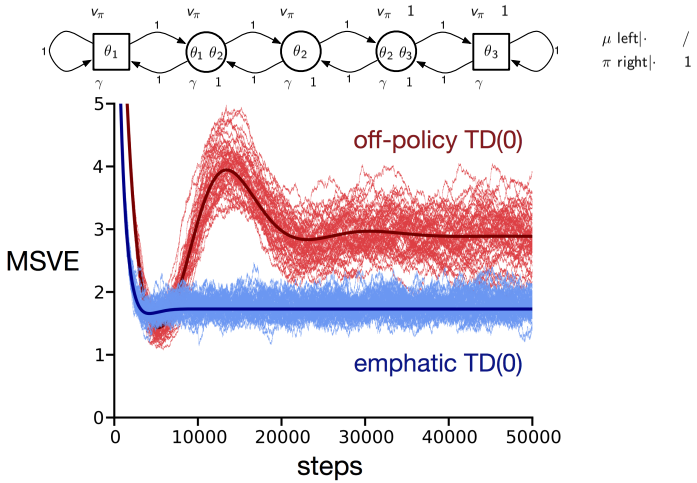


Figure 4: Twenty learning curves and their analytic expectation on the 5-state problem from Section 5, in which excursions terminate promptly and both algorithms converge reliably. Here $\lambda = 0$, $\boldsymbol{\theta}_0 = \mathbf{0}$, $\alpha = 0.001$, and $i(s) = 1, \forall s$. The MSVE performance measure is defined in (15).

this example (nevertheless, we do not mean to claim any general empirical advantage for emphatic TD(λ) at this time).

Also shown in these figures as a thick dark line is the trajectory of the deterministic algorithm: $\bar{\boldsymbol{\theta}}_{t+1} = \bar{\boldsymbol{\theta}}_t + \alpha(\mathbf{b} - \mathbf{A}\bar{\boldsymbol{\theta}}_t)$ (6). Tsitsiklis and Van Roy (1997) argued that, for small step-size parameters and in the steady-state distribution, on-policy TD(λ) follows its expected-update algorithm in an "average" sense, and we see much the same here for emphatic TD(λ).

These examples show that although emphatic TD(λ) is stable for any MDP and all functions λ, γ and (positive) i, for some problems and functions the parameter vector continues to fluxuate with a chance of arbitrarily large deviations (for constant $\alpha > 0$). It is not clear how great of a problem this is. Certainly it is much less of a problem than the positive instability (Baird 1995) that can occur with off-policy TD(λ) (stability of the expected update precludes this). The possibility of large fluxuations may be inherent in any algorithm for off-policy learning using importance sampling with long eligibility traces. For example, the updates of GTD(λ) and GQ(λ) (Maei 2011) with $\lambda = 1$ will tend to infinite variance as $t \to \infty$ on Baird's counterexample and on the example in Figures 1 and 3(left). And, as mentioned earlier, convergence with probability one can still be guaranteed if α is reduced appropriately (Yu 2015a,b).

In practice, however, even when asymptotic convergence can be guaranteed, high variance can be problematic as it may require very small step sizes and slow learning. High

variance frequently arises in off-policy algorithms when they are Monte Carlo algorithms (no TD learning) or they have eligibility traces with high λ (at $\lambda = 1$, TD algorithms become Monte Carlo algorithms). In both cases the root problem is the same: importance sampling ratios that become very large when multiplied together. For example, in the $\theta \to 2\theta$ problem discussed at the beginning of this section, the ratio was only two, but the products of successive twos rapidly produced a very large F_t. Thus, the first way in which variance can be controlled is to ensure that large products cannot occur. We are actually concerned with products of both ρ_ts and γ_ts. Occasional termination ($\gamma_t = 0$), as in the 5-state problem, is thus one reliable way of preventing high variance. Another is through choice of the target and behavior policies that together determine the importance sampling ratios. For example, one could define the target policy to be equal to the behavior policy whenever the followon or eligibility traces exceed some threshold. These tricks can also be done prospectively. White (personal communication) has proposed that the learner compute at each step the variance of what GTD(λ)'s traces would be on the following step. If the variance is in danger of becoming too large, then λ_t is reduced for that step to prevent it. For emphatic TD(λ), the same conditions could be used to adjust γ_t or one of the policies to prevent the variance from growing too large. Another idea for reducing variance is to use *weighted* importance sampling (as suggested by Precup et al. 2001) together with the ideas of Mahmood et al. (2014, 2015a) for extending weighted importance sampling to linear function approximation. Finally, a good solution may even be found by something as simple as bounding the values of F_t or $\mathbf{e}_t$. This would limit variance at the cost of bias, which might be a good tradeoff if done properly (e.g., see Hallak, Tamar, Munos & Mannor 2015).

9. Conclusions and Future Work

We have introduced a way of varying the emphasis or strength of the updates of TD learning algorithms from step to step, based on importance sampling, that should result in much lower variance than previous methods (Precup et al. 2001). In particular, we have introduced the emphatic TD(λ) algorithm and shown that it solves the problem of instability that plagues conventional TD(λ) when applied in off-policy training situations in conjunction with linear function approximation. Compared to gradient-TD methods, emphatic TD(λ) is simpler in that it has a single parameter vector and a single step size rather than two of each. The per-time-step complexities of gradient-TD and emphatic-TD methods are both linear in the number of parameters; both are much simpler than quadratic complexity methods such LSTD(λ) and its off-policy variants. We have also presented a few empirical examples of emphatic TD(0) compared to conventional TD(0) adapted to off-policy training. These examples illustrate some of emphatic TD(λ)'s basic strengths and weaknesses, but a proper empirical comparison with other methods remains for future work. Extensions of the emphasis idea to action-value and control methods such as Sarsa(λ) and Q(λ), to true-online forms (van Seijen & Sutton 2014, Sutton 2015), and to weighted importance sampling (Mahmood et al. 2014, 2015a) are also natural and remain for future work.

Yu (2015a) has recently extended the emphatic idea to a least-squares algorithm and proved that it and our emphatic TD(λ) are convergent with probability one. She has also obtained a stronger result that does not require that the interest be positive in all states,

using an argument similar to that given here about column sums and the Varga (1962) theorem for irreducibly diagonally dominant matrices (Yu 2015a, see also Mahmood et al. 2015b). Yu (2015b) proves convergence for weaker conditions on the step size. Asymptotic bounds on the error of emphatic TD(λ) have been obtained by Hallak et al. (2015).

Two additional ideas for future work deserve special mention.

First, note that the present work has focused on ways of ensuring that the key matrix is positive definite, which implies positive definiteness of the $\mathbf{A}$ matrix and thus that the update is stable. An alternative strategy would be to work directly with the $\mathbf{A}$ matrix. Recall that the $\mathbf{A}$ matrix is vastly smaller than the key matrix; it has a row and column for each *feature*, whereas the key matrix has a row and column for each *state*. It might be feasible then to keep statistics for each row and column of $\mathbf{A}$, whereas of course it would not be feasible for the large key matrix. For example, one might try to use such statistics to directly test for diagonal dominance (and thus positive definiteness) of $\mathbf{A}$. If it were possible to adjust some of the free parameters (e.g., the λ or i functions) to ensure positive definiteness while reducing the variance of F_t, then a substantially improved algorithm might be found.

The second idea for future work is that the emphasis algorithm, by tracing the dependencies among the estimates at various states, is doing something clever that ought to show up as improved bounds on the asymptotic approximation error. The bound given by Tsitsiklis and Van Roy (1997) probably cannot be significantly improved if λ, γ, i, and ρ are all constant, because in this case emphasis asymptotes to a constant that can be absorbed into the step size. But if any of these vary from step to step, then emphatic TD(λ) is genuinely different and may improve over conventional TD(λ). In particular, consider an episodic on-policy case where $i(s) \doteq 1$ and $\lambda(s) \doteq 0$, for all $s \in \mathcal{S}$, and $\gamma(s) \doteq 1$ for all states except for a terminal state where it is zero (and from which a new episode starts). In this case emphasis would increase linearly within an episode to a maximum on the final state, whereas conventional TD(λ) would give equal weight to all steps within the episode. If the feature representation were insufficient to represent the value function exactly, then the emphatic algorithm might improve over the conventional algorithm in terms of asymptotic MSVE (15). Similarly, improvements in asymptotic MSVE over conventional algorithms might be possible whenever i varies from state to state, such as in the common episodic case in which we are interested only in accurately valuing the start state of the episode, and yet we choose $\lambda < 1$ to reduce variance. There may be a wide range of interesting theoretical and empirical work to be done along these lines.

Acknowledgements

The authors thank Hado van Hasselt, Doina Precup, Huizhen Yu, Susan Murphy, and Brendan Bennett for insights and discussions contributing to the results presented in this paper, and the entire Reinforcement Learning and Artificial Intelligence research group for providing the environment to nurture and support this research. We gratefully acknowledge funding from Alberta Innovates – Technology Futures and from the Natural Sciences and Engineering Research Council of Canada.

References

Baird, L. C. (1995). Residual algorithms: Reinforcement learning with function approximation. *International Conference on Machine Learning 12*, pp. 30–37. Morgan Kaufmann. Important modifications and errata added to the online version on November 22, 1995.

Bertsekas, D. P. (2012). *Dynamic Programming and Optimal Control: Approximate Dynamic Programming*, Fourth Edition. Athena Scientific, Belmont, MA.

Bertsekas, D. P., Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA.

Boyan, J. A., (1999). Least-squares temporal difference learning. *International Conference on Machine Learning 16*, pp. 49–56.

Bradtke, S., Barto, A. G. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning 22*:33–57.

Dayan, P. (1992). The convergence of TD(λ) for general λ. *Machine Learning 8*:341–362.

Dann, C., Neumann, G., Peters, J. (2014). Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research 15*:809–883.

Geist, M., Scherrer, B. (2014). Off-policy learning with eligibility traces: A survey. *Journal of Machine Learning Research 15*:289–333.

Gordon, G. J. (1995). Stable function approximation in dynamic programming. *International Conference on Machine Learning 12*, pp. 261–268. An expanded version was published as Technical Report CMU-CS-95-103, Carnegie Mellon University, Pittsburgh, PA, 1995.

Gordon, G. J. (1996). Stable fitted reinforcement learning. *Advances in Neural Information Processing Systems 8*, pp. 1052–1058.

Hackman, L. (2012). *Faster Gradient-TD Algorithms.* MSc thesis, University of Alberta.

Hallak, A., Tamar, A., Munos, R., Mannor, S. (2015). Generalized emphatic temporal difference learning: Bias-variance analysis. ArXiv:1509.05172.

Klopf, A. H. (1988). A neuronal model of classical conditioning. *Psychobiology 16*(2):85–125.

Kolter, J. Z. (2011). The fixed points of off-policy TD. *Advances in Neural Information Processing Systems 24*, pp. 2169–2177.

Lagoudakis, M., Parr, R. (2003). Least squares policy iteration. *Journal of Machine Learning Research 4*:1107–1149.

Ludvig, E. A., Sutton, R. S., Kehoe, E. J. (2012). Evaluating the TD model of classical conditioning. *Learning & behavior 40*(3):305–319.

Maei, H. R. (2011). *Gradient Temporal-Difference Learning Algorithms.* PhD thesis, University of Alberta.

Maei, H. R., Sutton, R. S. (2010). GQ(λ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. *Artificial General Intelligence 3*, pp. 91–96.

Maei, H. R., Szepesvári, Cs., Bhatnagar, S., Sutton, R. S. (2010). Toward off-policy learning control with function approximation. *International Conference on Machine Learning 27*, pp. 719–726.

Mahmood, A. R., van Hasselt, H., Sutton, R. S. (2014). Weighted importance sampling for off-policy learning with linear function approximation. *Advances in Neural Information Processing Systems 27*.

Mahmood, A. R., Sutton, R. S. (2015a). Off-policy learning based on weighted importance sampling with linear computational complexity. *Uncertainty in Artificial Intelligence 31*.

Mahmood, A. R., Yu, H., White, M., Sutton, R. S. (2015b). Emphatic temporal-difference learning. *European Workshop on Reinforcement Learning 12*, ArXiv:1507.01569.

Modayil, J., White, A., Sutton, R. S. (2014). Multi-timescale nexting in a reinforcement learning robot. *Adaptive Behavior 22*(2):146–160.

Nedić, A., Bertsekas, D. P. (2003). Least squares policy evaluation algorithms with linear function approximation. *Discrete Event Dynamic Systems 13*(1-2):79–110.

Niv, Y., Schoenbaum, G. (2008). Dialogues on prediction errors. *Trends in cognitive sciences 12*(7):265–272.

O'Doherty, J. P. (2012). Beyond simple reinforcement learning: The computational neurobiology of reward learning and valuation. *European Journal of Neuroscience 35*(7):987–990.

Precup, D., Sutton, R. S., Dasgupta, S. (2001). Off-policy temporal-difference learning with function approximation. *International Conference on Machine Learning 18*, pp. 417–424.

Precup, D., Sutton, R. S., Singh, S. (2000). Eligibility traces for off-policy policy evaluation. *International Conference on Machine Learning 17*, pp. 759–766.

Rummery, G. A. (1995). *Problem Solving with Reinforcement Learning*. PhD thesis, University of Cambridge.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal on Research and Development 3*:210–229. Reprinted in E. A. Feigenbaum & J. Feldman (Eds.), *Computers and thought*, McGraw-Hill, New York.

Schultz, W., Dayan, P., Montague, P. R. (1997). A neural substrate of prediction and reward. *Science 275*(5306):1593–1599.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning 3*:9–44, erratum p. 377.

Sutton, R. S. (1995). TD models: Modeling the world at a mixture of time scales. *International Conference on Machine Learning 12*, pp. 531–539.

Sutton, R. S. (2009). The grand challenge of predictive empirical abstract knowledge. *Working Notes of the IJCAI-09 Workshop on Grand Challenges for Reasoning from Experiences*.

Sutton, R. S. (2012). Beyond reward: The problem of knowledge and data. *International Conference on Inductive Logic Programming 21*, S. H. Muggleton, A. Tamaddoni-Nezhad, F. A. Lisi (Eds.): ILP 2011, LNAI 7207, pp. 2–6. Springer, Heidelberg.

Sutton, R. S. (2015). True online emphatic TD(λ): Quick reference and implementation guide. ArXiv 1507.07147.

Sutton, R. S., Barto, A. G. (1990). Time-derivative models of Pavlovian reinforcement. In M. Gabriel and J. Moore (Eds.), *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, pp. 497–537. MIT Press, Cambridge, MA.

Sutton, R. S., Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Sutton, R. S., Mahmood, A. R., Precup, D., van Hasselt, H. (2014). A new Q(λ) with interim forward view and Monte Carlo equivalence. *International Conference on Machine Learning 31. JMLR W&CP 32*(2).

Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, Cs., Wiewiora, E. (2009). Fast gradient-descent methods for temporal-difference learning with linear function approximation. *International Conference on Machine Learning 26*, pp. 993–1000.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. *International Conference on Autonomous Agents and Multiagent Systems 10*, pp. 761–768.

Sutton, R. S., Precup D., Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence 112*:181–211.

Sutton, R. S., Rafols, E. J., Koop, A. (2006). Temporal abstraction in temporal-difference networks. *Advances in Neural Information Processing Systems 18*.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning 8*:257–277.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM 38*(3):58–68.

Thomas, P. (2014). Bias in natural actor–critic algorithms. *International Conference on Machine Learning 31. JMLR W&CP 32*(1):441–448.

Tsitsiklis, J. N., Van Roy, B. (1996). Feature-based methods for large scale dynamic programming. *Machine Learning 22*:59–94.

Tsitsiklis, J. N., Van Roy, B. (1997). An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control 42*:674–690.

van Seijen, H., Sutton, R. S. (2014). True online TD(λ). *International Conference on Machine Learning 31. JMLR W&CP 32*(1):692–700.

Varga, R. S. (1962). *Matrix Iterative Analysis*. Englewood Cliffs, NJ: Prentice-Hall.

Wang, M., Bertsekas, D. P. (2013). Stabilization of stochastic iterative methods for singular and nearly singular linear systems. *Mathematics of Operations Research 39*(1):1-30.

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards*. PhD thesis, University of Cambridge.

Watkins, C. J. C. H., Dayan, P. (1992). Q-learning. *Machine Learning 8*:279–292.

White, A. (2015). *Developing a Predictive Approach to Knowledge*. Phd thesis, University of Alberta.

Yu, H. (2010). Convergence of least squares temporal difference methods under general conditions. *International Conference on Machine Learning 27*, pp. 1207–1214.

Yu, H. (2012). Least squares temporal difference methods: An analysis under general conditions. *SIAM Journal on Control and Optimization 50*(6):3310–3343.

Yu, H. (2015a). On convergence of emphatic temporal-difference learning. ArXiv:1506.02582. A shorter version appeared in *Conference on Learning Theory 18*, *JMLR W&CP 40*.

Yu, H. (2015b). Weak convergence properties of constrained emphatic temporal-difference learning with constant and slowly diminishing stepsize. ArXiv:1511.07471.

Journal of Machine Learning Research 17 (2016) 1-25 Submitted 12/14; Revised 8/15; Published 4/16

Learning Algorithms for Second-Price Auctions with Reserve

Mehryar Mohri and **Andrés Muñoz Medina**
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, NY, 10012

Editor: Kevin Murphy

Abstract

Second-price auctions with reserve play a critical role in the revenue of modern search engine and popular online sites since the revenue of these companies often directly depends on the outcome of such auctions. The choice of the reserve price is the main mechanism through which the auction revenue can be influenced in these electronic markets. We cast the problem of selecting the reserve price to optimize revenue as a learning problem and present a full theoretical analysis dealing with the complex properties of the corresponding loss function. We further give novel algorithms for solving this problem and report the results of several experiments in both synthetic and real-world data demonstrating their effectiveness.

Keywords: Learning Theory, Auctions, Revenue Optimization

1. Introduction

Over the past few years, advertisement has gradually moved away from the traditional printed promotion to the more tailored and directed online publicity. The advantages of online advertisement are clear: since most modern search engine and popular online site companies such as Microsoft, Facebook, Google, eBay, or Amazon may collect information about the users' behavior, advertisers can better target the population sector for which their brand is intended.

More recently, a new method for selling advertisements has gained momentum. Unlike the standard contracts between publishers and advertisers where some amount of impressions are required to be fulfilled by the publisher, an Ad Exchange works in a way similar to a financial exchange where advertisers bid and compete between each other for an ad slot. The winner then pays the publisher and his ad is displayed.

The design of such auctions and their properties are crucial since they generate a large fraction of the revenue of popular online sites. These questions have motivated extensive research on the topic of auctioning in the last decade or so, particularly in the theoretical computer science and economic theory communities. Much of this work has focused on the analysis of mechanism design, either to prove some useful property of an existing auctioning mechanism, to analyze its computational efficiency, or to search for an optimal revenue maximization truthful mechanism (see Muthukrishnan (2009) for a good discussion of key research problems related to Ad Exchange and references to a fast growing literature therein).

One particularly important problem is that of determining an auction mechanism that achieves optimal revenue (Muthukrishnan, 2009). In the ideal scenario where the valuation of the bidders is drawn i.i.d. from a given distribution, this is known to be achievable (see for example (Myerson, 1981)). But, even good approximations of such distributions are not known in practice. Game theoretical approaches to the design of auctions have given a series of interesting results including (Riley and Samuelson, 1981; Milgrom and Weber, 1982; Myerson, 1981; Nisan et al., 2007), all of them based on some assumptions about the distribution of the bidders, e.g., the monotone hazard rate assumption.

The results of these publications have set the basis for most Ad Exchanges in practice: the mechanism widely adopted for selling ad slots is that of a *Vickrey auction* (Vickrey, 1961) or *second-price auction with reserve price* (Easley and Kleinberg, 2010). In such auctions, the winning bidder (if any) pays the maximum of the second-place bid and the reserve price. The reserve price can be set by the publisher or automatically by the exchange. The popularity of these auctions relies on the fact that they are incentive-compatible, i.e., bidders bid exactly what they are willing to pay. It is clear that the revenue of the publisher depends greatly on how the reserve price is set: if set too low, the winner of the auction might end up paying only a small amount, even if his bid was high; on the other hand, if it is set too high, then bidders may not bid higher than the reserve price and the ad slot will not be sold.

We propose a learning approach to the problem of determining the reserve price to optimize revenue in such auctions. The general idea is to leverage the information gained from past auctions to predict a beneficial reserve price. Since every transaction on an Exchange is logged, it is natural to seek to exploit that data. This could be used to estimate the probability distribution of the bidders, which can then be used indirectly to come up with the optimal reserve price (Myerson, 1981; Ostrovsky and Schwarz, 2011). Instead, we will seek a discriminative method making use of the loss function related to the problem and taking advantage of existing user features.

Learning methods have been used in the past for the related problems of designing incentive-compatible auction mechanisms (Balcan et al., 2008; Blum et al., 2004), for algorithmic bidding (Langford et al., 2010; Amin et al., 2012), and even for predicting bid landscapes (Cui et al., 2011). Another closely related problem for which machine learning solutions have been proposed is that of revenue optimization for sponsored search ads and click-through rate predictions (Zhu et al., 2009; He et al., 2013; Devanur and Kakade, 2009). But, to our knowledge, no prior work has used historical data in combination with user features for the sole purpose of revenue optimization in this context. In fact, the only publications we are aware of that are directly related to our objective are (Ostrovsky and Schwarz, 2011) and (Cesa-Bianchi et al., 2013), which considers a more general case than (Ostrovsky and Schwarz, 2011).

The scenario studied by Cesa-Bianchi et al. (2013) is that of censored information, which motivates their use of a regret minimization algorithm to optimize the revenue of the seller. Our analysis assumes instead access to full information. We argue that this is a more realistic scenario since most companies do have access to the full historical data. The learning scenario we consider is also more general since it includes the use of features, as is standard in supervised learning. Since user information is communicated to advertisers and bids are made based on that information, it is only natural to include user features in the formulation of the learning problem. A special case of our analysis coincides with the no-feature scenario considered by Cesa-Bianchi et al. (2013), assuming full information. But, our results further extend those of this paper even in that scenario. In particular, we present an $O(m \log m)$ algorithm for solving a key optimization problem used as a subroutine by these authors, for which they do not seem to give an algorithm. We also do not assume that buyers' bids are sampled i.i.d. from a common distribution. Instead, we only assume that the full outcome of each auction is independent and identically distributed. This subtle distinction makes our scenario closer to reality as it is unlikely for all bidders to follow the same underlying value distribution. Moreover, even though our scenario does not take into account a possible strategic behavior of bidders between rounds, it allows for bidders to be correlated, which is common in practice.

This paper is organized as follows: in Section 2, we describe the setup and give a formal description of the learning problem. We discuss the relations between the scenario we consider and previous work on learning in auctions in Section 3. In particular, we show that, unlike previous work, our problem can be cast as that of minimizing the expected value of a loss function, which is a standard learning problem. Unlike most work in this field, however, the loss function naturally associated to this problem does not admit favorable properties such as convexity or Lipschitz continuity. In fact the loss function is discontinuous. Therefore, the theoretical and algorithmic analysis of this problem raises several non-trivial technical issues. Nevertheless, we use a decomposition of the loss to derive generalization bounds for this problem (see Section 4). These bounds suggest the use of structural risk minimization to determine a learning solution benefiting from strong guarantees. This, however, poses a new challenge: solving a highly non-convex optimization problem. Similar algorithmic problems have been of course previously encountered in the learning literature, most notably when seeking to minimize a regularized empirical 0-1 loss in binary classification. A standard method in machine learning for dealing with such issues consists of resorting to a convex surrogate loss (such as the hinge loss commonly used in linear classification). However, we show in Section 4.2 that no convex loss function is calibrated for the natural loss function for this problem. That is, minimizing a convex surrogate could be detrimental to learning. This fact is further empirically verified in Section 6.

The impossibility results of Section 4.2 prompt us to search for surrogate loss functions with weaker regularity properties such as Lipschitz continuity. We describe a loss function with precisely that property which we further show to be consistent with the original loss. We also provide finite sample learning guarantees for that loss function, which suggest minimizing its empirical value while controlling the complexity

of the hypothesis set. This leads to an optimization problem which, albeit non-convex, admits a favorable decomposition as a difference of two convex functions (DC-programming). Thus, we suggest using the DC-programming algorithm (DCA) introduced by Tao and An (1998) to solve our optimization problem. This algorithm admits favorable convergence guarantees to a *local minimum*. To further improve upon DCA, we propose a combinatorial algorithm to cycle through different local minima with the guarantee of reducing the objective function at every iteration. Finally, in Section 6, we show that our algorithm outperforms several different baselines in various synthetic and real-world revenue optimization tasks.

2. Setup

We start with the description of the problem and our formulation of the learning setup. We study second-price auctions with reserve, the type of auctions adopted in many Ad Exchanges. In such auctions, the bidders submit their bids simultaneously and the winner, if any, pays the maximum of the value of the second-place bid and a reserve price r set by the seller. This type of auctions benefits from the same truthfulness property as second-price auctions (or Vickrey auctions) Vickrey (1961): truthful bidding can be shown to be a dominant strategy in such auctions. The choice of the reserve price r is the only mechanism through which the seller can influence the auction revenue. Its choice is thus critical: if set too low, the amount paid by the winner could be too small; if set too high, the ad slot could be lost. How can we select the reserve price to optimize revenue?

We consider the problem of learning to set the reserve prices to optimize revenue in second-price auctions with reserve. The outcome of an auction can be encoded by the highest and second-highest bids which we denote by a vector $\mathbf{b} = (b^{(1)}, b^{(2)}) \in \mathcal{B} \subset \mathbb{R}^2_+$. We will assume that there exists an upper bound $M \in (0, +\infty)$ for the bids: $\sup_{b \in \mathcal{B}} b^{(1)} = M$. For a given reserve price r and bid pair $\mathbf{b}$, by definition, the revenue of an auction is given by

$$\text{Revenue}(r, \mathbf{b}) = b^{(2)} \mathbb{1}_{r < b^{(2)}} + r \mathbb{1}_{b^{(2)} \leq r \leq b^{(1)}}.$$

We consider the general scenario where a feature vector $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^N$ is associated with each auction. In the auction theory literature, this feature vector is commonly referred to as *public information*. In the context of online advertisement, this could be for example information about the user's location, gender or age. The learning problem can thus be formulated as that of selecting out of a hypothesis set H of functions mapping $\mathcal{X}$ to $\mathbb{R}$ a hypothesis h with high expected revenue

$$\mathop{\mathbb{E}}_{(\mathbf{x}, \mathbf{b}) \sim D} [\text{Revenue}(h(\mathbf{x}), \mathbf{b})], \tag{1}$$

where D is an unknown distribution according to which pairs $(\mathbf{x}, \mathbf{b})$ are drawn. Instead of the revenue, we will consider a loss function L defined for all $(r, \mathbf{b})$ by $L(r, \mathbf{b}) = -\text{Revenue}(r, \mathbf{b})$, and will equivalently seek a hypothesis h with small expected loss $\mathcal{L}(h) := \mathbb{E}_{(\mathbf{x}, \mathbf{b}) \sim D}[L(h(\mathbf{x}), \mathbf{b})]$. As in standard supervised learning scenarios, we assume access to a training sample $S = ((\mathbf{x}_1, \mathbf{b}_1), \ldots, (\mathbf{x}_m, \mathbf{b}_m))$ of size $m \geq 1$ drawn i.i.d. according to D. We will denote by $\widehat{\mathcal{L}}_S(h)$ the empirical loss $\widehat{\mathcal{L}}_S(h) = \frac{1}{m} \sum_{i=1}^m L(h(\mathbf{x}_i, \mathbf{b}_i))$. Notice that we only assume that the auction outcomes are i.i.d. and not that bidders are independent of each other with the same underlying bid distribution, as in some previous work (Cesa-Bianchi et al., 2013; Ostrovsky and Schwarz, 2011). In the next sections, we will present a detailed study of this learning problem, starting with a review of the related literature.

3. Previous work

Here, we briefly discuss some previous work related to the study of auctions from a learning standpoint. One of the earliest contributions in this literature is that of Blum et al. (2004) where the authors studied a posted-price auction mechanism where a seller offers some good at a certain price and where the buyer decides to either accept that price or reject it. It is not hard to see that this type of auctions is equivalent to second-price auctions with reserve with a single buyer. The authors consider a scenario of repeated interactions with different buyers where the goal is to design an incentive-compatible method of setting prices that is competitive with the best fixed-priced strategy in hindsight. A fixed-price strategy is one that simply offers the same price to all buyers. Using a variant of the EXP3 algorithm of Auer et al. (2002), the authors designed a pricing algorithm achieving a $(1+\epsilon)$-approximation to the best fixed-price strategy. This same scenario was also studied by Kleinberg and Leighton (2003) who gave an online algorithm whose regret after T rounds is in $O(T^{2/3})$.

A step further in the design of optimal pricing strategies was proposed by Balcan et al. (2008). One of the problems considered by the authors was that of setting prices for n buyers in a posted-price auction as a function of their public information. Unlike the on-line scenario of Blum et al. (2004), Balcan et al. (2008) considered a batch scenario where all buyers are known in advance. However, the comparison class considered was no longer that of simple fixed-price strategies but functions mapping public information to prices. This makes the problem more challenging and closer to the scenario we consider. The authors showed that finding a $(1+\epsilon)$-optimal truthful mechanism is equivalent to finding an algorithm to optimize the empirical risk associated to the loss function we consider (in the case $b^{(2)} \equiv 0$). There are multiple connections between this work and our results. In particular, the authors pointed out that the discontinuity and asymmetry of the loss function presented several challenges to their analysis. We will see that, in fact, the same problems appear in the derivation of our learning guarantees. But, we will present an algorithm for minimizing the empirical risk which was a crucial element missing in their results.

A different line of work by Cui et al. (2011) focused on predicting the highest bid of a second-price auction. To estimate the distribution of the highest bid, the authors partitioned the space of advertisers based on their campaign objectives and estimated the distribution for each partition. Within each partition, the distribution of the highest bid was modeled as a mixture of log-normal distributions where the means and standard deviations of the mixtures were estimated as a function of the data features. While it may seem natural to seek to predict the highest bid, we show that this is not necessary and that accurate predictions of the highest bid do not necessarily translate into algorithms achieving large revenue (see Section 6).

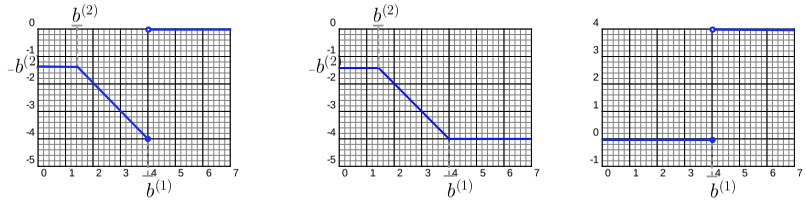
As already mentioned, the closest previous work to ours is that of Cesa-Bianchi et al. (2013), who studied the problem of directly optimizing the revenue under a partial information setting where the learner can only observe the value of the second-highest bid, if it is higher than the reserve price. In particular, the highest bid remains unknown to the learner. This is a natural scenario for auctions such as those of eBay where only the price at which an object is sold is reported. To do so, the authors expressed the expected revenue in terms of the quantity $q(t) = \mathbb{P}[b^{(2)} > t]$. This can be done as follows:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{b}}[\text{Revenue}(r, \mathbf{b})] &= \mathop{\mathbb{E}}_{b^{(2)}}[b^{(2)} \mathbb{1}_{r < b^{(2)}}] + r\, \mathbb{P}[b^{(2)} \leq r \leq b^{(1)}] \\
&= \int_0^{+\infty} \mathbb{P}[b^{(2)} \mathbb{1}_{r < b^{(2)}} > t]\, dt + r\, \mathbb{P}[b^{(2)} \leq r \leq b^{(1)}] \\
&= \int_0^{r} \mathbb{P}[r < b^{(2)}]\, dt + \int_r^{\infty} \mathbb{P}[b^{(2)} > t] dt + r\, \mathbb{P}[b^{(2)} \leq r \leq b^{(1)}] \\
&= \int_r^{\infty} \mathbb{P}[b^{(2)} > t]\, dt + r(\mathbb{P}[b^{(2)} > r] + 1 - \mathbb{P}[b^{(2)} > r] - \mathbb{P}[b^{(1)} < r]) \\
&= \int_r^{\infty} \mathbb{P}[b^{(2)} > t]\, dt + r\, \mathbb{P}[b^{(1)} \geq r].
\end{aligned} \tag{2}$$

The main observation of Cesa-Bianchi et al. (2013) was that the quantity $q(t)$ can be estimated from the observed outcomes of previous auctions. Furthermore, if the buyers' bids are i.i.d., then, one can express $\mathbb{P}[b^{(1)} \geq r]$ as a function of the estimated value of $q(r)$. This implies that the right-hand side of (2) can be accurately estimated and therefore an optimal reserve price can be selected. Their algorithm makes calls to a procedure that maximizes the empirical revenue. The authors, however, did not describe an algorithm for that maximization. A by-product of our work is an efficient algorithm for that procedure. The guarantees of Cesa-Bianchi et al. (2013) are similar to those presented in the next section in the special case of learning without features. However, our derivation is different since we consider a batch scenario while Cesa-Bianchi et al. (2013) treated an online setup for which they presented regret guarantees.

4. Learning Guarantees

The problem we consider is an instance of the well known family of supervised learning problems. However, the loss function L does not admit any of the properties such as convexity or Lipschitz continuity often assumed in the analysis of the generalization error, as shown by Figure 1(a). Furthermore, L is discontinuous and, unlike the 0-1 loss function whose discontinuity point is independent of the label, its discontinuity depends on the outcome $\mathbf{b}$ of the auction. Thus, the problem of learning with the loss function L requires a new analysis.



(a) (b)

Figure 1: (a) Plot of the loss function $r \mapsto L(r, \mathbf{b})$ for fixed values of $b^{(1)}$ and $b^{(2)}$; (b) Functions l_1 on the left and l_2 on the right.

4.1 Generalization bound

To analyze the complexity of the family of functions L_H mapping $\mathcal{X} \times \mathcal{B}$ to $\mathbb{R}$ defined by

$$L_H = \{(\mathbf{x}, \mathbf{b}) \mapsto L(h(\mathbf{x}), \mathbf{b}) \colon h \in H\},$$

we decompose L as a sum of two loss functions l_1 and l_2 with more favorable properties than L. We have $L = l_1 + l_2$ with l_1 and l_2 defined for all $(r, \mathbf{b}) \in \mathbb{R} \times \mathcal{B}$ by

$$l_1(r, \mathbf{b}) = -b^{(2)} \mathbb{1}_{r < b^{(2)}} - r \mathbb{1}_{b^{(2)} \leq r \leq b^{(1)}} - b^{(1)} \mathbb{1}_{r > b^{(1)}}$$
$$l_2(r, \mathbf{b}) = b^{(1)} \mathbb{1}_{r > b^{(1)}}.$$

These functions are shown in Figure 1(b). Note that, for a fixed $\mathbf{b}$, the function $r \mapsto l_1(r, \mathbf{b})$ is 1-Lipschitz since the slope of the lines defining the function is at most 1. We will consider the corresponding families of loss functions: $l_{1H} = \{(\mathbf{x}, \mathbf{b}) \mapsto l_1(h(\mathbf{x}), \mathbf{b}) \colon h \in H\}$ and $l_{2H} = \{(\mathbf{x}, \mathbf{b}) \mapsto l_2(h(\mathbf{x}), \mathbf{b}) \colon h \in H\}$ and use the notion of pseudo-dimension as well as those of empirical and average Rademacher complexity to measure their complexities. The pseudo-dimension is a standard complexity measure (Pollard, 1984) extending the notion of VC-dimension to real-valued functions (see also Mohri et al. (2012)). For a family of functions G and finite sample $S = (z_1, \ldots, z_m)$ of size m, the empirical Rademacher complexity is defined by $\widehat{\mathfrak{R}}_S(G) = \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{g \in G} \frac{1}{m} \sum_{i=1}^m \sigma_i g(z_i)\right]$, where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_m)^\top$, with σ_is independent uniform random variables taking values in $\{-1, +1\}$. The Rademacher complexity of G is defined as $\mathfrak{R}_m(G) = \mathbb{E}_{S \sim D^m}[\widehat{\mathfrak{R}}_S(G)]$.

To bound the complexity of L_H, we will first bound the complexity of the family of loss functions l_{1H} and l_{2H}. Since l_1 is 1-Lipschitz, the complexity of the class l_{1H} can be readily bounded by that of H, as shown by the following proposition.

Proposition 1 *For any sample $S = ((\mathbf{x}_1, \mathbf{b}_1), \ldots, (\mathbf{x}_m, \mathbf{b}_m))$, the empirical Rademacher complexity of l_{1H} can be bounded as follows:*

$$\widehat{\mathfrak{R}}_S(l_{1H}) \leq \widehat{\mathfrak{R}}_S(H).$$

Proof By definition of the empirical Rademacher complexity, we can write

$$\widehat{\mathfrak{R}}_S(l_{1H}) = \frac{1}{m} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h \in H} \sum_{i=1}^m \sigma_i l_1(h(\mathbf{x}_i), \mathbf{b}_i)\right] = \frac{1}{m} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h \in H} \sum_{i=1}^m \sigma_i (\psi_i \circ h)(\mathbf{x}_i)\right],$$

where, for all $i \in [1, m]$, ψ_i is the function defined by $\psi_i \colon r \mapsto l_1(r, \mathbf{b}_i)$. For any $i \in [1, m]$, ψ_i is 1-Lipschitz, thus, by the contraction lemma of Appendix A (Lemma 14), the following inequality holds:

$$\widehat{\mathfrak{R}}_S(l_{1H}) \leq \frac{1}{m} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h \in H} \sum_{i=1}^m \sigma_i h(\mathbf{x}_i)\right] = \widehat{\mathfrak{R}}_S(H),$$

which completes the proof. ∎

As shown by the following proposition, the complexity of l_{2H} can be bounded in terms of the pseudo-dimension of H.

Proposition 2 *Let $d = Pdim(H)$ denote the pseudo-dimension of H, then, for any sample $S = ((\mathbf{x}_1, \mathbf{b}_1), \ldots, (\mathbf{x}_m, \mathbf{b}_m))$, the empirical Rademacher complexity of l_{2H} can be bounded as follows:*

$$\widehat{\mathfrak{R}}_S(l_{2H}) \leq M \sqrt{\frac{2d \log \frac{em}{d}}{m}}.$$

Proof By definition of the empirical Rademacher complexity, we can write

$$\widehat{\mathfrak{R}}_S(l_{2H}) = \frac{1}{m} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h \in H} \sum_{i=1}^m \sigma_i b_i^{(1)} \mathbb{1}_{h(\mathbf{x}_i) > b_i^{(1)}}\right] = \frac{1}{m} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h \in H} \sum_{i=1}^m \sigma_i \Psi_i(\mathbb{1}_{h(\mathbf{x}_i) > b_i^{(1)}})\right],$$

where, for all $i \in [1, m]$, Ψ_i is the M-Lipschitz function $x \mapsto b_i^{(1)} x$. Thus, by Lemma 14 combined with Massart's lemma (see for example Mohri et al. (2012)), we can write

$$\widehat{\mathfrak{R}}_S(l_{2H}) \leq \frac{M}{m} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h \in H} \sum_{i=1}^m \sigma_i \mathbb{1}_{h(\mathbf{x}_i) > b_i^{(1)}}\right] \leq M \sqrt{\frac{2d' \log \frac{em}{d'}}{m}},$$

where $d' = \text{VCdim}(\{(\mathbf{x}, \mathbf{b}) \mapsto \mathbb{1}_{h(\mathbf{x}) - b^{(1)} > 0} \colon (\mathbf{x}, \mathbf{b}) \in \mathcal{X} \times \mathcal{B}\})$. Since the second bid component $b^{(2)}$ plays no role in this definition, d' coincides with $\text{VCdim}(\{(\mathbf{x}, b^{(1)}) \mapsto \mathbb{1}_{h(\mathbf{x}) - b^{(1)} > 0} \colon (\mathbf{x}, b^{(1)}) \in \mathcal{X} \times \mathcal{B}_1\})$, where $\mathcal{B}_1$ is the projection of $\mathcal{B} \subseteq \mathbb{R}^2$ onto its first component, and is upper-bounded by $\text{VCdim}(\{(\mathbf{x}, t) \mapsto \mathbb{1}_{h(\mathbf{x}) - t > 0} \colon (\mathbf{x}, t) \in \mathcal{X} \times \mathbb{R}\})$, that is, the pseudo-dimension of H. ∎

Propositions 1 and 2 can be used to derive the following generalization bound for the learning problem we consider.

Theorem 3 *For any $\delta > 0$, with probability at least $1 - \delta$ over the draw of an i.i.d. sample S of size m, the following inequality holds for all $h \in H$:*

$$\mathcal{L}(h) \leq \widehat{\mathcal{L}}_S(h) + 2\mathfrak{R}_m(H) + 2M \sqrt{\frac{2d \log \frac{em}{d}}{m}} + M \sqrt{\frac{\log \frac{1}{\delta}}{2m}}.$$

Proof By a standard property of the Rademacher complexity, since $L = l_1 + l_2$, the following inequality holds: $\mathfrak{R}_m(L_H) \leq \mathfrak{R}_m(l_{1H}) + \mathfrak{R}_m(l_{2H})$. Thus, in view of Propositions 1 and 2, the Rademacher complexity of L_H can be bounded via

$$\mathfrak{R}_m(L_H) \leq \mathfrak{R}_m(H) + M \sqrt{\frac{2d \log \frac{em}{d}}{m}}.$$

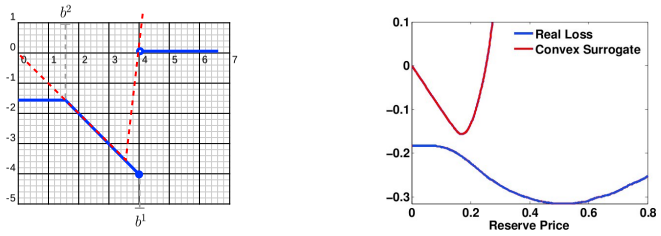
The result then follows by the application of a standard Rademacher complexity bound (Koltchinskii and Panchenko, 2002). ∎

This learning bound invites us to consider an algorithm seeking $h \in H$ to minimize the empirical loss $\widehat{\mathcal{L}}_S(h)$, while controlling the complexity (Rademacher complexity and pseudo-dimension) of the hypothesis set H. In the following section, we discuss the computational problem of minimizing the empirical loss and suggest the use of a surrogate loss leading to a more tractable problem.

4.2 Surrogate Loss

As previously mentioned, the loss function L does not admit most properties of traditional loss functions used in machine learning: for any fixed $\mathbf{b}$, $L(\cdot, \mathbf{b})$ is not differentiable (at two points), it is not convex nor Lipschitz, and in fact it is discontinuous. For any fixed $\mathbf{b}$, $L(\cdot, \mathbf{b})$ is quasi-convex,[1] a property that is often desirable since there exist several solutions for quasi-convex optimization problems. However, in general, a sum of quasi-convex functions, such as the sum $\sum_{i=1}^m L(\cdot, \mathbf{b}_i)$ appearing in the definition of the empirical loss, is not quasi-convex and a fortiori not convex.[2] In general, such a sum may admit exponentially many local minima. This leads us to seek a surrogate loss function with more favorable optimization properties.

1. A function $f \colon \mathbb{R} \to \mathbb{R}$ is said to be *quasi-convex* if for any $\alpha \in \mathbb{R}$ the sub-level set $\{x \colon f(x) \leq \alpha\}$ is convex.
2. It is known that, under some separability condition, if a finite sum of quasi-convex functions on an open convex set is quasi-convex, then all but perhaps one of them is convex (Debreu and Koopmans, 1982).



(a) (b)

Figure 2: (a) Piecewise linear convex surrogate loss L_p. (b) Comparison of the sum of real losses $\sum_{i=1}^m L(\cdot, \mathbf{b}_i)$ for $m = 500$ with the sum of convex surrogate losses. Note that the minimizers are significantly different.

A standard method in machine learning consists of replacing the loss function L with a convex upper bound (Bartlett et al., 2006). A natural candidate in our case is the piecewise linear function L_p shown in Figure 2(a). While this is a convex loss function, and thus convenient for optimization, it is not calibrated. That is, it is possible for $r_p \in \operatorname{argmin} \mathbb{E}_{\mathbf{b}}[L_p(r, \mathbf{b})]$ to have a large expected true loss. Therefore, it does not provide us with a useful surrogate. The calibration problem is illustrated by Figure 2(b) in dimension one, where the true objective function to be minimized $\sum_{i=1}^m L(r, \mathbf{b}_i)$ is compared with the sum of the surrogate losses. The next theorem shows that this problem affects *any* non-constant convex surrogate. It is expressed in terms of the loss $\widetilde{L}\colon \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}$ defined by $\widetilde{L}(r, b) = -r\mathbb{1}_{r \le b}$, which coincides with L when the second bid is 0.

Definition 4 *We say that a function $L_c\colon [0, M] \times [0, M] \to \mathbb{R}$ is* consistent with $\widetilde{L}$ *if, for any distribution D, there exists a minimizer $r^* \in \operatorname{argmin}_r \mathbb{E}_{b\sim D}[L_c(r, b)]$ such that $r^* \in \operatorname{argmin}_r \mathbb{E}_{b\sim D}[\widetilde{L}(r, b)]$.*

Definition 5 *We say that a sequence of functions $(L_n)_{n\in\mathbb{N}}$ mapping $[0, M] \times [0, M]$ to $\mathbb{R}$ is* weakly consistent with $\widetilde{L}$ *if there exists a sequence $(r_n)_{n\in\mathbb{N}}$ in $\mathbb{R}$ with $r_n \in \operatorname{argmin}_r \mathbb{E}_{b\sim D}[L_n(r, b)]$ for all $n \in \mathbb{N}$ such that $\lim_{n\to+\infty} r_n = r^*$ with $r^* \in \operatorname{argmin} \mathbb{E}_{b\sim D}[\widetilde{L}(r, b)]$.*

Proposition 6 (Convex surrogates) *Let $L_c\colon [0, M] \times [0, M] \to \mathbb{R}$ be a bounded function, convex with respect to its first argument. If L_c is consistent with $\widetilde{L}$, then $L_c(\cdot, b)$ is constant for any $b \in [0, M]$.*

Proof The idea behind the proof is the following: for any two bids $b_1 < b_2$, there exists a distribution D with support $\{b_1, b_2\}$ such that $\mathbb{E}_{b\sim D}[\widetilde{L}(r, b)]$ is minimized at both $r = b_1$ and $r = b_2$. We show this implies that $\mathbb{E}_{b\sim D}[L_c(r, b)]$ must attain a minimum at both points too. By convexity of L_c, it follows that $\mathbb{E}_{b\sim D}[L_c(r, b)]$ must be constant on the interval $[b_1, b_2]$. The main part of the proof will be showing that this implies that the function $L_c(\cdot, b_1)$ must also be constant on the interval $[b_1, b_2]$. Finally, since the value of b_2 was chosen arbitrarily, it will follow that $L_c(\cdot, b_1)$ is constant.

Let $0 < b_1 < b_2 < M$ and, for any $\mu \in [0, 1]$, let D_μ denote the probability distribution with support included in $\{b_1, b_2\}$ defined by $D_\mu(b_1) = \mu$ and let $\mathbb{E}_\mu$ denote the expectation with respect to this distribution. A straightforward calculation shows that the unique minimizer of $\mathbb{E}_\mu[\widetilde{L}(r, b)]$ is given by b_2 if $\mu > \frac{b_2 - b_1}{b_2}$ and by b_1 if $\mu < \frac{b_2 - b_1}{b_2}$. Therefore, if $F_\mu(r) = \mathbb{E}_\mu[L_c(r, b)]$, it must be the case that b_2 is a minimizer of F_μ for $\mu > \frac{b_2 - b_1}{b_2}$ and b_1 is a minimizer of F_μ for $\mu < \frac{b_2 - b_1}{b_2}$.

For a convex function $f\colon \mathbb{R} \to \mathbb{R}$, we denote by f^- its left-derivative and by f^+ its right-derivative, which are guaranteed to exist. We will also denote here, for any $b \in \mathbb{R}$, by $g^-(\cdot, b)$ and $g^+(\cdot, b)$ the left- and right-derivatives of the function $g(\cdot, b)$ and by $g'(\cdot, b)$ its derivative, when it exists. Recall that for a convex function f, if x_0 is a minimizer, then $f^-(x_0) \le 0 \le f^+(x_0)$. In view of that and the minimizing properties

of b_1 and b_2, the following inequalities hold:

$$0 \ge F_\mu^-(b_2) = \mu L_c^-(b_2, b_1) + (1 - \mu) L_c^-(b_2, b_2) \qquad \text{for } \mu > \frac{b_2 - b_1}{b_2}, \tag{3}$$

$$0 \le F_\mu^+(b_1) \le F_\mu^-(b_2) \qquad \text{for } \mu < \frac{b_2 - b_1}{b_2}, \tag{4}$$

where the second inequality in (4) holds by convexity of F_μ and the fact that $b_1 < b_2$. By setting $\mu = \frac{b_2 - b_1}{b_2}$, it follows from inequalities (3) and (4) that $F_\mu^-(b_2) = 0$ and $F_\mu^+(b_1) = 0$. By convexity of F_μ, it follows that F_μ is constant on the interval (b_1, b_2). We now show this may only happen if $L_c(\cdot, b_1)$ is also constant. By rearranging terms in (3) and plugging in the expression of μ, we obtain the equivalent condition

$$(b_2 - b_1) L_c^-(b_2, b_1) = -b_1 L_c^-(b_2, b_2).$$

Since L_c is a bounded function, it follows that $L_c^-(b_2, b_1)$ is bounded for any $b_1, b_2 \in (0, M)$, therefore as $b_1 \to b_2$ we must have $b_2 L_c^-(b_2, b_2) = 0$, which implies $L_c^-(b_2, b_2) = 0$ for all $b_2 > 0$. In view of this, inequality (3) may only be satisfied if $L_c^-(b_2, b_1) \le 0$. However, the convexity of L_c implies $L_c^-(b_2, b_1) \ge L_c^-(b_1, b_1) = 0$. Therefore, $L_c^-(b_2, b_1) = 0$ must hold for all $b_2 > b_1 > 0$. Similarly, by definition of F_μ, the first inequality in (4) implies

$$\mu L_c^+(b_1, b_1) + (1 - \mu) L_c^+(b_1, b_2) \ge 0. \tag{5}$$

Nevertheless, for any $b_2 > b_1$ we have $0 = L_c^-(b_1, b_1) \le L_c^+(b_1, b_1) \le L_c^-(b_2, b_1) = 0$. Consequently, $L_c^+(b_1, b_1) = 0$ for all $b_1 > 0$. Furthermore, $L_c^+(b_1, b_2) \le L_c^+(b_2, b_2) = 0$. Therefore, for inequality (5) to be satisfied, we must have $L_c^+(b_1, b_2) = 0$ for all $b_1 < b_2$.

Thus far, we have shown that for any $b > 0$, if $r \ge b$, then $L_c^-(r, b) = 0$, while $L_c^+(r, b) = 0$ for $r \le b$. A simple convexity argument shows that $L_c(\cdot, b)$ is then differentiable and $L_c'(r, b) = 0$ for all $r \in (0, M)$, which in turn implies that $L_c(\cdot, b)$ is a constant function. ∎

The result of the previous proposition can be considerably strengthened, as shown by the following theorem. As in the proof of the previous proposition, to simplify the notation, for any $b \in \mathbb{R}$, we will denote by $g'(\cdot, b)$ the derivative of a differentiable function $g(\cdot, b)$.

Theorem 7 *Let $(L_n)_{n\in\mathbb{N}}$ denote a sequence of functions mapping $[0, M] \times [0, M]$ to $\mathbb{R}$ that are convex and differentiable with respect to their first argument and satisfy the following conditions:*

- $\sup_{b\in[0,M], n\in\mathbb{N}} \max(|L_n'(0, b)|, |L_n'(M, b)| = K < \infty$*;*
- $(L_n)_n$ *is weakly consistent with $\widetilde{L}$;*
- $L_n(0, b) = 0$ *for all $n \in \mathbb{N}$ and for all b.*

If the sequence $(L_n)_n$ converges pointwise to a function L_c, then $L_n(\cdot, b)$ converges uniformly to $L_c(\cdot, b) \equiv 0$.

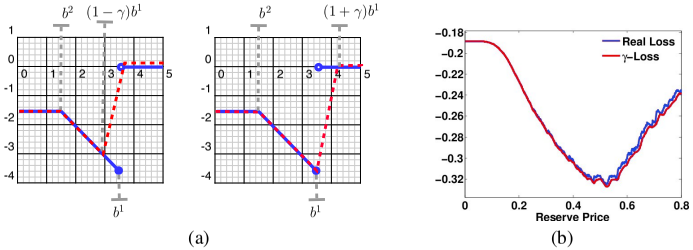
We defer the proof of this theorem to Appendix B and present here only a sketch of the proof. We first show that the convexity of the functions L_n implies that the convergence to L_c must be uniform and that L_c is convex with respect to its first argument. This fact and the weak consistency of the sequence L_n will then imply that L_c is consistent with $\widetilde{L}$ and therefore must be constant by Proposition 6.

The theorem just presented shows that even a weakly consistent sequence of convex losses is uniformly close to a constant function and therefore not helpful to tackle the learning task we consider. This suggests searching for surrogate losses that admit weaker regularity assumptions such as Lipschitz continuity.

Perhaps, the most natural surrogate loss function is then $\overline{L}_\gamma$, an upper bound on L defined for all $\gamma > 0$ by:

$$\overline{L}_\gamma(r, \mathbf{b}) = -b^{(2)}\mathbb{1}_{r \le b^{(2)}} - r\mathbb{1}_{b^{(2)} < r \le \left((1-\gamma)b^{(1)}\right) \vee b^{(2)}} + \left(\frac{1-\gamma}{\gamma} \vee \frac{b^{(2)}}{b^{(1)} - b^{(2)}}\right)(r - b^{(1)})\mathbb{1}_{\left((1-\gamma)b^{(1)}\right) \vee b^{(2)} < r \le b^{(1)}},$$

where $c \vee d = \max(c, d)$. The plot of this function is shown in Figure 3(a). The max terms ensure that the function is well defined if $(1 - \gamma)b^{(1)} < b^{(2)}$. However, this turns out to be also a poor choice as $\overline{L}_\gamma$ is a loose



(a) (b)

Figure 3: (a) Comparison of the true loss L with surrogate loss $\overline{L}_\gamma$ on the left and surrogate loss L_γ on the right, for $\gamma = 0.1$. (b) Comparison of $\sum_{i=1}^{500} L(r, \mathbf{b}_i)$ and $\sum_{i=1}^{500} L_\gamma(r, \mathbf{b}_i)$

upper bound on L in the most critical region, that is around the minimum of the loss L. Thus, instead, we will consider, for any $\gamma > 0$, the loss function L_γ defined as follows:

$$L_\gamma(r, \mathbf{b}) = -b^{(2)} \mathbb{1}_{r \leq b^{(2)}} - r \mathbb{1}_{b^{(2)} < r \leq b^{(1)}} + \frac{1}{\gamma}(r - (1+\gamma)b^{(1)}) \mathbb{1}_{b^{(1)} < r \leq (1+\gamma)b^{(1)}}, \tag{6}$$

whose plot is shown in Figure 3(a).[3] A comparison between the sum of L-losses and the sum of L_γ-losses is shown in Figure 3(b). Observe that the fit is considerably better than that of the piecewise linear convex surrogate loss shown in Figure 2(b). A possible concern associated with the loss function L_γ is that it is a lower bound for L. One might think then that minimizing it would not lead to an informative solution. However, we argue that this problem arises significantly with upper bounding losses such as the convex surrogate, which we showed not to lead to a useful minimizer, or $\overline{L}_\gamma$, which is a poor approximation of L near its minimum. By matching the original loss L in the region of interest, around the minimal value, the loss function L_γ leads to more informative solutions for this problem. In fact, we show that that the expected loss $\mathcal{L}_\gamma(h) \colon = \mathbb{E}_{\mathbf{x},\mathbf{b}}[L_\gamma(h)]$ admits a minimizer close to the minimizer of $\mathcal{L}(h)$. Since $L_\gamma \to L$ as $\gamma \to 0$, this result may seem trivial. However, this convergence is not uniform and therefore calibration is not guaranteed.

Theorem 8 *Let H be a closed, convex subset of a linear space of functions containing 0. Then, the following inequality holds for all $\gamma \geq 0$:*

$$\mathcal{L}(h^*_\gamma) - \mathcal{L}_\gamma(h^*_\gamma) \leq \gamma M.$$

Notice that, since $L \geq L_\gamma$ for all $\gamma \geq 0$, the theorem implies that $\lim_{\gamma \to 0} \mathcal{L}(h^*_\gamma) = \mathcal{L}(h^*)$. Indeed, let h^* denote the best-in-class hypothesis for the loss function L. Then, the following straightforward inequalities hold:

$$\begin{aligned}\mathcal{L}(h^*) &\leq \mathcal{L}(h^*_\gamma) \\ &\leq \mathcal{L}_\gamma(h^*_\gamma) + \gamma M \\ &\leq \mathcal{L}_\gamma(h^*) + \gamma M \leq \mathcal{L}(h^*) + \gamma M.\end{aligned}$$

By letting $\gamma \to 0$, we see that $\mathcal{L}(h^*_\gamma) \to \mathcal{L}(h^*)$. This is a remarkable result as it not only provides a convergence guarantee, but it also gives us an explicit rate of convergence. We will later exploit this fact to derive an optimal choice for γ.

The proof of Theorem 8 is based on the following partitioning of $\mathcal{X} \times \mathcal{B}$ in four regions where L_γ is defined as an affine function:

$$\begin{aligned} I_1 &= \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) \leq b^{(2)}\} & I_2 &= \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) \in (b^{(2)}, b^{(1)}]\} \\ I_3 &= \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) \in (b^{(1)}, (1+\gamma)b^{(1)}]\} & I_4 &= \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) > (1+\gamma)b^{(1)}\}, \end{aligned}$$

3. Technically, the theoretical and algorithmic results we present for L_γ could be developed in a somewhat similar way for $\overline{L}_\gamma$.

Notice that L_γ and L differ only on I_3. Therefore, we only need to bound the measure of this set which can be done as in Lemma 15 (see Appendix C).

Proof [Theorem 8]. We can express the difference as

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})\right] &= \sum_{k=1}^{4} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[(L(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})) \mathbb{1}_{I_k}(\mathbf{x}, \mathbf{b})\right] \\ &= \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[(L(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})) \mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\right] \\ &= \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[\frac{1}{\gamma}((1+\gamma)b^{(1)} - h^*_\gamma(\mathbf{x})) \mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b}))\right]. \end{aligned} \tag{7}$$

Furthermore, for $(\mathbf{x}, \mathbf{b}) \in I_3$, we know that $b^{(1)} < h^*_\gamma(\mathbf{x})$. Thus, we can bound (7) by $\mathbb{E}_{\mathbf{x},\mathbf{b}}[h^*_\gamma(\mathbf{x}) \mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})]$, which, by Lemma 15 in Appendix C, is upper bounded by $\gamma \mathbb{E}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x}) \mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right]$. Thus, the following inequalities hold:

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L(h^*_\gamma(\mathbf{x}), \mathbf{b})\right] - \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})\right] \leq \gamma \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x}) \mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right] \leq \gamma \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[b^{(1)} \mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right] \leq \gamma M,$$

using the fact that $h^*_\gamma(\mathbf{x}) \leq b^{(1)}$ for $(\mathbf{x}, \mathbf{b}) \in I_2$. ■

The $1/\gamma$-Lipschitzness of L_γ can be used to prove the following generalization bound.

Theorem 9 *Fix $\gamma \in (0, 1]$ and let S denote a sample of size m. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the choice of the sample S, for all $h \in H$, the following holds:*

$$\mathcal{L}_\gamma(h) \leq \widehat{\mathcal{L}}_\gamma(h) + \frac{2}{\gamma}\mathfrak{R}_m(H) + M\sqrt{\frac{\log \frac{1}{\delta}}{2m}}. \tag{8}$$

Proof Let $L_{\gamma,H}$ denote the family of functions $\{(\mathbf{x}, \mathbf{b}) \to L_\gamma(h(\mathbf{x}), b) \colon h \in H\}$. The loss function L_γ is $\frac{1}{\gamma}$-Lipschitz since the slope of the lines defining it is at most $\frac{1}{\gamma}$. Thus, using the contraction lemma (Lemma 14) as in the proof of Proposition 1, gives $\mathfrak{R}_m(L_{\gamma,H}) \leq \frac{1}{\gamma}\mathfrak{R}_m(H)$. The application of a standard Rademacher complexity bound to the family of functions $L_{\gamma,H}$ then shows that for any $\delta > 0$, with probability at least $1 - \delta$, for any $h \in H$, the following holds:

$$\mathcal{L}_\gamma(h) \leq \widehat{\mathcal{L}}_\gamma(h) + \frac{2}{\gamma}\mathfrak{R}_m(H) + M\sqrt{\frac{\log \frac{1}{\delta}}{2m}}.$$

■

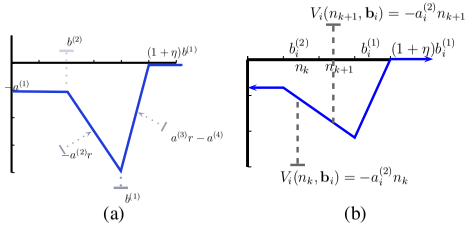
We conclude this section by showing that L_γ admits a stronger form of consistency. More precisely, we prove that the generalization error of the best-in-class hypothesis $\mathcal{L}^* := \mathcal{L}(h^*)$ can be lower bounded in terms of that of the empirical minimizer of L_γ, $\widehat{h}_\gamma \colon = \operatorname{argmin}_{h \in H} \widehat{\mathcal{L}}_\gamma(h)$.

Theorem 10 *Let $M = \sup_{b \in \mathcal{B}} b^{(1)}$ and let H be a hypothesis set with pseudo-dimension $d = Pdim(H)$. Then, for any $\delta > 0$ and a fixed value of $\gamma > 0$, with probability at least $1 - \delta$ over the choice of a sample S of size m, the following inequality holds:*

$$\mathcal{L}^* \leq \mathcal{L}(\widehat{h}_\gamma) \leq \mathcal{L}^* + \frac{2\gamma + 2}{\gamma}\mathfrak{R}_m(H) + \gamma M + 2M\sqrt{\frac{2d \log \frac{em}{d}}{m}} + 2M\sqrt{\frac{\log \frac{2}{\delta}}{2m}}.$$

Proof By Theorem 3, with probability at least $1 - \delta/2$, the following holds:

$$\mathcal{L}(\widehat{h}_\gamma) \leq \widehat{\mathcal{L}}_S(\widehat{h}_\gamma) + 2\mathfrak{R}_m(H) + 2M\sqrt{\frac{2d \log \frac{em}{d}}{m}} + M\sqrt{\frac{\log \frac{2}{\delta}}{2m}}. \tag{9}$$



(a) (b)

Figure 4: (a) Prototypical v-function. (b) Illustration of the fact that the definition of $V_i(r, \mathbf{b}_i)$ does not change on an interval $[n_k, n_{k+1}]$.

Applying Lemma 15 with the empirical distribution induced by the sample, we can bound $\widehat{\mathcal{L}}_S(\widehat{h}_\gamma)$ by $\widehat{\mathcal{L}}_\gamma(\widehat{h}_\gamma) + \gamma M$. The first term of the previous expression is less than $\widehat{\mathcal{L}}_\gamma(h^*_\gamma)$ by definition of $\widehat{h}_\gamma$. Moreover, the same analysis used in the proof of Theorem 9 shows that with probability $1 - \delta/2$,

$$\widehat{\mathcal{L}}_\gamma(h^*_\gamma) \leq \mathcal{L}_\gamma(h^*_\gamma) + \frac{2}{\gamma}\mathfrak{R}_m(H) + M\sqrt{\frac{\log \frac{2}{\delta}}{2m}}.$$

Finally, by definition of h^*_γ and using the fact that L is an upper bound on L_γ, we can write $\mathcal{L}_\gamma(h^*_\gamma) \leq \mathcal{L}_\gamma(h^*) \leq \mathcal{L}(h^*)$. Thus,

$$\widehat{\mathcal{L}}_S(\widehat{h}_\gamma) \leq \mathcal{L}(h^*) + \frac{2}{\gamma}\mathfrak{R}_m(H) + M\sqrt{\frac{\log \frac{2}{\delta}}{2m}} + \gamma M.$$

Replacing this inequality in (9) and applying the union bound yields the result. ■

This bound can be extended to hold uniformly over all γ at the price of a term in $O\Big(\frac{\sqrt{\log \log_2 \frac{1}{\gamma}}}{\sqrt{m}}\Big)$. Thus, for appropriate choices of γ as a function of m (for instance $\gamma = 1/m^{1/4}$) we can guarantee the convergence of $\mathcal{L}(\widehat{h}_\gamma)$ to $\mathcal{L}^*$, a stronger form of consistency (See Appendix C).

These results are reminiscent of the standard margin bounds with γ playing the role of a margin. The situation here is however somewhat different. Our learning bounds suggest, for a fixed $\gamma \in (0, 1]$, to seek a hypothesis h minimizing the empirical loss $\widehat{\mathcal{L}}_\gamma(h)$ while controlling a complexity term upper bounding $\mathfrak{R}_m(H)$, which in the case of a family of linear hypotheses could be $\|h\|^2_K$ for some PSD kernel K. Since the bound can hold uniformly for all γ, we can use it to select γ out of a finite set of possible grid search values. Alternatively, γ can be set via cross-validation. In the next section, we present algorithms for solving this regularized empirical risk minimization problem.

5. Algorithms

In this section, we show how to minimize the empirical risk under two regimes: first we analyze the no-feature scenario considered in Cesa-Bianchi et al. (2013) and then we present an algorithm to solve the more general feature-based revenue optimization problem.

5.1 No-Feature Case

We now present a general algorithm to optimize sums of functions similar to L_γ or L in the one-dimensional case.

Definition 11 *We will say that function* $V \colon \mathbb{R} \times \mathcal{B} \to \mathbb{R}$ *is a* v-function *if it admits the following form:*

$$V(r, \mathbf{b}) = -a^{(1)}\mathbb{1}_{r \leq b^{(2)}} - a^{(2)} r \mathbb{1}_{b^{(2)} < r \leq b^{(1)}} + (a^{(3)} r - a^{(4)})\mathbb{1}_{b^{(1)} < r < (1+\eta)b^{(1)}},$$

with $a^{(1)} > 0$ *and* $\eta > 0$ *constants and* $a^{(2)}, a^{(3)}, a^{(4)}$ *defined by* $a^{(1)} = \eta a^{(3)} b^{(2)}$, $a^{(2)} = \eta a^{(3)}$, *and* $a^{(4)} = a^{(3)}(1+\eta)b^{(1)}$.

Figure 4(a) illustrates this family of loss functions. A v-function is a generalization of L_γ and L. Indeed, any v-function V satisfies $V(r, \mathbf{b}) \leq 0$ and attains its minimum at $b^{(1)}$. Finally, as can be seen straightforwardly from Figure 3, L_γ is a v-function for any $\gamma > 0$. We consider the following general problem of minimizing a sum of v-functions:

$$\min_{r \geq 0} F(r) := \sum_{i=1}^{m} V_i(r, \mathbf{b}_i). \tag{10}$$

Observe that this is not a trivial problem since, for any fixed $\mathbf{b}_i$, $V_i(\cdot, \mathbf{b}_i)$ is non-convex and that, in general, a sum of m such functions may admit many local minima. Of course, we can seek a solution that is ϵ-close to the optimal reserve via a grid search over points $r_i = i\epsilon$. However, the guarantees for that algorithm would depend on the continuity of the function. In particular, this algorithm might fail for the loss L. Instead, we exploit the particular structure of a v-function to exactly minimize F. The following proposition, which is proven in Appendix D, shows that the minimum is attained at one of the highest bids, which matches the intuition. Notice that for the loss function L this is immediate since if r is not a highest bid, one can raise the reserve price without increasing any of the component losses.

Proposition 12 *Problem* (10) *admits a solution* r^* *that satisfies* $r^* = b_i^{(1)}$ *for some* $i \in [1, m]$.

Problem (10) can thus be reduced to examining the value of the function for the m arguments $b_i^{(1)}$, $i \in [1, m]$. This yields a straightforward method for solving the optimization which consists of computing $F(b_i^{(1)})$ for all i and taking the minimum. But, since the computation of each $F(b_i^{(1)})$ takes $O(m)$, the overall computational cost is in $O(m^2)$, which can be prohibitive for even moderately large values of m.

Instead, we present a combinatorial algorithm to solve the optimization problem (10) in $O(m \log m)$. Let $\mathcal{N} = \bigcup_i \{b_i^{(1)}, b_i^{(2)}, (1+\eta)b_i^{(1)}\}$ denote the set of all *boundary points* associated with the functions $V(\cdot, \mathbf{b}_i)$. The algorithm proceeds as follows: first, sort the set $\mathcal{N}$ to obtain the ordered sequence $(n_1, \ldots, n_{3m})$, which can be achieved in $O(m \log m)$ using a comparison-based sorting algorithm. Next, evaluate $F(n_1)$ and compute $F(n_{k+1})$ from $F(n_k)$ for all k.

The main idea of the algorithm is the following: since the definition of $V_i(\cdot, b_i)$ can only change at boundary points (see also Figure 4(b)), computing $F(n_{k+1})$ from $F(n_k)$ can be achieved in constant time. Indeed, since between n_k and n_{k+1} there are only two boundary points, we can compute $V(n_{k+1}, \mathbf{b}_i)$ from $V(n_k, \mathbf{b}_i)$ by calculating V for only two values of $\mathbf{b}_i$, which can be done in constant time. We now give a more detailed description and proof of correctness of our algorithm.

Proposition 13 *There exists an algorithm to solve the optimization problem* (10) *in* $O(m \log m)$.

Proof The pseudocode of the algorithm is given in Algorithm 1, where $a_i^{(1)}, \ldots, a_i^{(4)}$ denote the parameters defining the functions $V_i(r, \mathbf{b}_i)$. We will prove that, after running Algorithm 1, we can compute $F(n_j)$ in constant time using:

$$F(n_j) = c_j^{(1)} + c_j^{(2)} n_j + c_j^{(3)} n_j + c_j^{(4)}. \tag{11}$$

This holds trivially for n_1 since by definition $n_1 \leq b_i^{(2)}$ for all i and therefore $F(n_1) = -\sum_{i=1}^m a_i^{(1)}$. Now, assume that (11) holds for j, we prove that it must also hold for $j+1$. Suppose $n_j = b_i^{(2)}$ for some i (the cases $n_j = b_i^{(1)}$ and $n_j = (1+\eta)b_i^{(1)}$ can be handled in the same way). Then $V_i(n_j, \mathbf{b}_i) = -a_i^{(1)}$ and we can write

$$\sum_{k \neq i} V_k(n_j, \mathbf{b}_k) = F(n_j) - V(n_j, \mathbf{b}_i) = (c_j^{(1)} + c_j^{(2)} n_j + c_j^{(3)} n_j + c_j^{(4)}) + a_i^{(1)}.$$

Thus, by construction we would have:

$$\begin{aligned} c_{j+1}^{(1)} + c_{j+1}^{(2)} n_{j+1} + c_{j+1}^{(3)} n_{j+1} + c_{j+1}^{(4)} &= c_j^{(1)} + a_i^{(1)} + (c_j^{(2)} - a_i^{(2)}) n_{j+1} + c_j^{(3)} n_{j+1} + c_j^{(4)} \\ &= (c_j^{(1)} + c_j^{(2)} n_{j+1} + c_j^{(3)} n_{j+1} + c_j^{(4)}) + a_i^{(1)} - a_i^{(2)} n_{j+1} \\ &= \sum_{k \neq i} V_k(n_{j+1}, \mathbf{b}_k) - a_i^{(2)} n_{j+1}, \end{aligned}$$

Algorithm 1 Sorting

$\mathcal{N} := \bigcup_{i=1}^m \{b_i^{(1)}, b_i^{(2)}, (1+\eta)b_i^{(1)}\}$;
$n_1, ..., n_{3m}) = \mathbf{Sort}(\mathcal{N})$;
Set $\mathbf{c}_i := (c_i^{(1)}, c_i^{(2)}, c_i^{(3)}, c_i^{(4)}) = 0$ for $i = 1, ..., 3m$;
Set $c_1^{(1)} = -\sum_{i=1}^m a_i^{(1)}$;
for $j = 2, ..., 3m$ **do**
  Set $\mathbf{c}_j = \mathbf{c}_{j-1}$;
  if $n_{j-1} = b_i^{(2)}$ for some i **then**
    $c_j^{(1)} = c_j^{(1)} + a_i^{(1)}$;
    $c_j^{(2)} = c_j^{(2)} - a_i^{(2)}$;
  else if $n_{j-1} = b_i^{(1)}$ for some i **then**
    $c_j^{(2)} = c_j^{(2)} + a_i^{(2)}$;
    $c_j^{(3)} = c_j^{(3)} + a_i^{(3)}$;
    $c_j^{(4)} = c_j^{(4)} - a_i^{(4)}$;
  else
    $c_j^{(3)} = c_j^{(3)} - a_i^{(3)}$;
    $c_j^{(4)} = c_j^{(4)} + a_i^{(4)}$;
  end if
end for

where the last equality holds since the definition of $V_k(r, \mathbf{b}_k)$ does not change for $r \in [n_j, n_{j+1}]$ and $k \neq i$. Finally, since n_j was a boundary point, the definition of $V_i(r, \mathbf{b}_i)$ must change from $-a_i^{(1)}$ to $-a_i^{(2)} r$, thus the last equation is indeed equal to $F(n_{j+1})$. A similar argument can be given if $n_j = b_i^{(1)}$ or $n_j = (1+\eta)b_i^{(1)}$.

We proceed to analyze the complexity of the algorithm: sorting the set $\mathcal{N}$ can be performed in $O(m \log m)$ and each iteration takes only constant time. Thus, the evaluation of all points can be achieved in linear time and, clearly, the minimum can then also be obtained in linear time. Therefore, the overall time complexity of the algorithm is in $O(m \log m)$. ■

The algorithm just proposed can be straightforwardly extended to solve the minimization of F over a set of r-values bounded by Λ, that is $\{r : 0 \leq r \leq \Lambda\}$. Indeed, we need only compute $F(b_i^{(1)})$ for $i \in [1, m]$ such that $b_i^{(1)} < \Lambda$ and of course also $F(\Lambda)$, thus the computational complexity in this regularized case remains in $O(m \log m)$.

5.2 General Case

We now present our main algorithm for revenue optimization in the presence of features. This problem presents new challenges characteristic of non-convex optimization problems in higher dimensions. Therefore, our proposed algorithm can only guarantee convergence to a local minimum. Nevertheless, we provide a simple method for cycling through these local minima with the guarantee of reducing the objective function at each time.

We consider the case of a hypothesis set $H \subset \mathbb{R}^N$ of linear functions $\mathbf{x} \mapsto \mathbf{w} \cdot \mathbf{x}$ with bounded norm, $\|\mathbf{w}\| \leq \Lambda$, for some $\Lambda \geq 0$. This can be immediately generalized to non-linear hypotheses by using a positive definite kernel.

The results of Theorem 9 suggest seeking, for a fixed $\gamma \geq 0$, the vector $\mathbf{w}$ solution to the following optimization problem: $\min_{\|\mathbf{w}\| \leq \Lambda} \sum_{i=1}^m L_\gamma(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$. Replacing the original loss L with L_γ helped us remove the discontinuity of the loss. But, we still face an optimization problem based on a sum of non-convex functions. This problem can be formulated as a DC-programming (difference of convex functions programming) problem which is a well studied problem in non-convex optimization. Indeed, L_γ can be decomposed as follows for all $(r, \mathbf{b}) \in \mathbb{R} \times \mathcal{B}$: $L_\gamma(r, \mathbf{b}) = u(r, \mathbf{b}) - v(r, \mathbf{b})$, with the convex functions u

and v defined by

$$u(r, \mathbf{b}) = -r\mathbb{1}_{r<b^{(1)}} + \frac{r-(1+\gamma)b^{(1)}}{\gamma}\mathbb{1}_{r \geq b^{(1)}}$$
$$v(r, \mathbf{b}) = (-r + b^{(2)})\mathbb{1}_{r<b^{(2)}} + \frac{r-(1+\gamma)b^{(1)}}{\gamma}\mathbb{1}_{r>(1+\gamma)b^{(1)}}.$$

Using the decomposition $L_\gamma = u - v$, our optimization problem can be formulated as follows:

$$\min_{\mathbf{w} \in \mathbb{R}^N} U(\mathbf{w}) - V(\mathbf{w}) \qquad \text{subject to } \|\mathbf{w}\| \leq \Lambda, \tag{12}$$

where $U(\mathbf{w}) = \sum_{i=1}^m u(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$ and $V(\mathbf{w}) = \sum_{i=1}^m v(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$, which shows that it can be formulated as a DC-programming problem. The global minimum of the optimization problem (12) can be found using a cutting plane method (Horst and Thoai, 1999), but that method only converges in the limit and does not admit known algorithmic convergence guarantees.[4] There exists also a branch-and-bound algorithm with exponential convergence for DC-programming (Horst and Thoai, 1999) for finding the global minimum. Nevertheless, in (Tao and An, 1997), it is pointed out that such combinatorial algorithms fail to solve real-world DC-programs in high dimensions. In fact, our implementation of this algorithm shows that the convergence of the algorithm in practice is extremely slow for even moderately high-dimensional problems. Another attractive solution for finding the global solution of a DC-programming problem over a polyhedral convex set is the combinatorial solution of Tuy (1964). However, this method requires explicitly specifying the slope and offsets for the piecewise linear function corresponding to a sum of L_γ losses and incurs an exponential cost in time and space.

An alternative consists of using the DC algorithm (DCA), a primal-dual sub-differential method of Dinh Tao and Hoai An Tao and An (1998), (see also Tao and An (1997) for a good survey). This algorithm is applicable when u and v are proper lower semi-continuous convex functions as in our case. When v is differentiable, the DC algorithm coincides with the CCCP algorithm of Yuille and Rangarajan (2003), which has been used in several contexts in machine learning and analyzed by Sriperumbudur and Lanckriet (2012).

The general proof of convergence of the DC algorithm was given by Tao and An (1998). In some special cases, the DC algorithm can be used to find the global minimum of the problem as in the trust region problem (Tao and An, 1998), but, in general, the DC algorithm or its special case CCCP are only guaranteed to converge to a critical point (Tao and An, 1998; Sriperumbudur and Lanckriet, 2012). Nevertheless, the number of iterations of the DC algorithm is relatively small. Its convergence has been shown to be in fact linear for DC-programming problems such as ours (Yen et al., 2012). The algorithm we are proposing goes one step further than that of Tao and An (1998): we use DCA to find a local minimum but then restart our algorithm with a new seed that is guaranteed to reduce the objective function. Unfortunately, we are not in the same regime as in the trust region problem of Tao and An (1998) where the number of local minima is linear in the size of the input. Indeed, here, the number of local minima can be exponential in the number of dimensions of the feature space and it is not clear to us how the combinatorial structure of the problem could help us rule out some local minima faster and make the optimization more tractable.

In the following, we describe more in detail the solution we propose for solving the DC-programming problem (12). The functions v and V are not differentiable in our context but they admit a sub-gradient at all points. We will denote by $\delta V(\mathbf{w})$ an arbitrary element of the sub-gradient $\partial V(\mathbf{w})$, which coincides with $\nabla V(\mathbf{w})$ at points $\mathbf{w}$ where V is differentiable. The DC algorithm then coincides with CCCP, modulo the replacement of the gradient of V by $\delta V(\mathbf{w})$. It consists of starting with a weight vector $\mathbf{w}_0 \leq \Lambda$ and of iteratively solving a sequence of convex optimization problems obtained by replacing V with its linear approximation giving $\mathbf{w}_t$ as a function of $\mathbf{w}_{t-1}$, for $t = 1, \ldots, T$: $\mathbf{w}_t \in \operatorname{argmin}_{\|\mathbf{w}\| \leq \Lambda} U(\mathbf{w}) - \delta V(\mathbf{w}_{t-1}) \cdot \mathbf{w}$. This problem can be rewritten in our context as the following:

$$\min_{\|\mathbf{w}\|^2 \leq \Lambda^2, \mathbf{s}} \sum_{i=1}^m s_i - \delta V(\mathbf{w}_{t-1}) \cdot \mathbf{w} \tag{13}$$
$$\text{subject to } (s_i \geq -\mathbf{w} \cdot \mathbf{x}_i) \wedge \left[s_i \geq \frac{1}{\gamma}\left(\mathbf{w} \cdot \mathbf{x}_i - (1+\gamma)b_i^{(1)}\right)\right].$$

4. Some claims of Horst and Thoai (1999), e.g., Proposition 4.4 used in support of the cutting plane algorithm, are incorrect (Tuy, 2002).

DC Algorithm

$\mathbf{w} \leftarrow \mathbf{w}_0$ ▷ initialization
while $\mathbf{v} \neq \mathbf{w}$ **do**
  $\mathbf{v} \leftarrow \mathrm{DCA}(\mathbf{w})$ ▷ DC algorithm
  $\mathbf{u} \leftarrow \frac{\mathbf{v}}{\|\mathbf{v}\|}$
  $\eta^* \leftarrow \min_{0 \leq \eta \leq \Lambda} \sum_{\mathbf{u} \cdot \mathbf{x}_i > 0} L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i)$
  $\mathbf{w} \leftarrow \eta^* \mathbf{v}$
end while

Figure 5: Pseudocode of our DC-programming algorithm.

The problem is equivalent to a QP (quadratic-programming). Indeed, by convex duality, there exists a $\lambda > 0$ such that the above problem is equivalent to

$$\min_{w \in R^N} \lambda \|w\|^2 + \sum_{i=1}^{m} s_i - \delta V(\mathbf{w}_{t-1}) \cdot \mathbf{w}$$
$$\text{subject to } (s_i \geq -\mathbf{w} \cdot \mathbf{x}_i) \wedge \left[s_i \geq \frac{1}{\gamma} \left(\mathbf{w} \cdot \mathbf{x}_i - (1+\gamma) b_i^{(1)} \right) \right]$$

which is a simple QP that can be tackled by one of many off-the-shelf QP solvers. Of course, the value of λ as a function of Λ does not admit a simple expression. Instead, we select λ through validation which is then equivalent to choosing the optimal value of Λ through validation.

We now address the problem of the DC algorithm converging to a local minimum. A common practice is to restart the DC algorithm at a new random point. Instead, we propose an algorithm that iterates along different local minima, with the guarantee of reducing the function at every change of local minimum. The algorithm is simple and is based on the observation that the function L_γ is positive homogeneous. Indeed, for any $\eta > 0$ and $(r, \mathbf{b})$,

$$\begin{aligned} L_\gamma(\eta r, \eta \mathbf{b}) &= -\eta b^{(2)} \mathbb{1}_{\eta r < \eta b^{(2)}} - \eta r \mathbb{1}_{\eta b^{(2)} \leq \eta r \leq \eta b^{(1)}} + \frac{\eta r - (1+\gamma)\eta b^{(1)}}{\gamma} \mathbb{1}_{\eta b^{(1)} < \eta r < \eta(1+\gamma) b^{(1)}} \\ &= \eta L_\gamma(r, \mathbf{b}). \end{aligned}$$

Minimizing the objective function of (12) in a fixed direction $\mathbf{u}$, $\|\mathbf{u}\| = 1$, can be reformulated as follows: $\min_{0 \leq \eta \leq \Lambda} \sum_{i=1}^{m} L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i)$. Since for $\mathbf{u} \cdot \mathbf{x}_i \leq 0$ the function $\eta \mapsto L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i)$ is constant and equal to $-b_i^{(2)}$, the problem is equivalent to solving

$$\min_{0 \leq \eta \leq \Lambda} \sum_{\mathbf{u} \cdot \mathbf{x}_i > 0} L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i).$$

Furthermore, since L_γ is positive homogeneous, for all $i \in [1, m]$ with $\mathbf{u} \cdot \mathbf{x}_i > 0$, $L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i) = (\mathbf{u} \cdot \mathbf{x}_i) L_\gamma(\eta, \mathbf{b}_i / (\mathbf{u} \cdot \mathbf{x}_i))$. But $\eta \mapsto (\mathbf{u} \cdot \mathbf{x}_i) L_\gamma(\eta, \mathbf{b}_i / (\mathbf{u} \cdot \mathbf{x}_i))$ is a v-function and thus the problem can efficiently be optimized using the combinatorial algorithm for the no-feature case (Section 5.1). This leads to the optimization algorithm described in Figure 5. The last step of each iteration of our algorithm can be viewed as a *line search* and this is in fact the step that reduces the objective function the most in practice. This is because we are then precisely minimizing the objective function even though this is for some fixed direction. Since in general this line search does not find a local minimum (we are likely to decrease the objective value in other directions that are not the one in which the line search was performed) running DCA helps us find a better direction for the next iteration of the line search.

6. Experiments

In this section, we report the results of several experiments with synthetic and real-world data demonstrating the benefits of our algorithm. Since the use of features for reserve price optimization has not been previously studied in the literature, we are not aware of any baseline for comparison with our algorithm. Therefore, its performance is measured against three natural strategies that we now describe.

As mentioned before, a standard solution for solving this problem would be the use of a convex surrogate loss. In view of that, we compare against the solution of the regularized empirical risk minimization of the convex surrogate loss L_α shown in Figure 2(a) parametrized by $\alpha \in [0, 1]$ and defined by

$$L_\alpha(r, \mathbf{b}) = \begin{cases} -r & \text{if } r < b^{(1)} + \alpha(b^{(2)} - b^{(1)}) \\ \left(\frac{(1-\alpha) b^{(1)} + \alpha b^{(2)}}{\alpha (b^{(1)} - b^{(2)})} \right) (r - b^{(1)}) & \text{otherwise.} \end{cases}$$

A second alternative consists of using ridge regression to estimate the first bid and of using its prediction as the reserve price. A third algorithm consists of minimizing the loss while ignoring the feature vectors $\mathbf{x}_i$, i.e., solving the problem $\min_{r \leq \Lambda} \sum_{i=1}^{n} L(r, \mathbf{b}_i)$. It is worth mentioning that this third approach is very similar to what advertisement exchanges currently use to suggest reserve prices to publishers. By using the empirical version of equation (2), we see that this algorithm is equivalent to finding the empirical distribution of bids and optimizing the expected revenue with respect to this empirical distribution as in (Ostrovsky and Schwarz, 2011) and (Cesa-Bianchi et al., 2013).

6.1 Artificial Data Sets

We generated 4 different synthetic data sets with different correlation levels between features and bids. For all our experiments, the feature vectors $\mathbf{x} \in \mathbb{R}^{21}$ were generated in as follows: $\tilde{\mathbf{x}} \in \mathbb{R}^{20}$ was sampled from a standard Gaussian distribution and $\mathbf{x} = (\tilde{\mathbf{x}}, 1)$ was created by adding an offset feature. We now describe the bid generating process for each of the experiments as a function of the feature vector $\mathbf{x}$. For our first three experiments, shown in Figure 6(a)-(c), the highest bid and second highest bid were set to $\max\left(\left| \sum_{i=1}^{21} x_i \right| + \epsilon_1, \left| \sum_{i=1}^{21} \frac{x_i}{2} \right| + \epsilon_2 \right)_+$ and $\min\left(\left| \sum_{i=1}^{21} x_i \right| + \epsilon_1, \left| \sum_{i=1}^{21} \frac{x_i}{2} \right| + \epsilon_2 \right)_+$ respectively, where ϵ_i is a Gaussian random variable with mean 0. The standard deviation of the Gaussian noise was varied over the set $\{0, 0.25, 0.5\}$.

For our last artificial experiment, we used a generative model motivated by previous empirical observations (Ostrovsky and Schwarz, 2011; Lahaie and Pennock, 2007): bids were generated by sampling two values from a log-normal distribution with means $\mathbf{x} \cdot \mathbf{w}$ and $\frac{\mathbf{x} \cdot \mathbf{w}}{2}$ and standard deviation 0.5, with $\mathbf{w}$ a random vector sampled from a standard Gaussian distribution.

For all our experiments, the parameters λ, γ and α were selected respectively from the sets $\{2^i | i \in [-5, 5]\}$, $\{0.1, 0.01, 0.001\}$, and $\{0.1, 0.2, \ldots, 0.9\}$ via validation over a set consisting of the same number of examples as the training set. Our algorithm was initialized using the best solution of the convex surrogate optimization problem. The test set consisted of 5,000 examples drawn from the same distribution as the training set. Each experiment was repeated 10 times and the mean revenue of each algorithm is shown in Figure 6. The plots are normalized in such a way that the revenue obtained by setting no reserve price is equal to 0 and the maximum possible revenue (which can be obtained by setting the reserve price equal to the highest bid) is equal to 1. The performance of the ridge regression algorithm is not included in Figure 6(d) as it was too inferior to be comparable with the performance of the other algorithms.

By inspecting the results in Figure 6(a), we see that, even in the simplest noiseless scenario, our algorithm outperforms all other techniques. The reader could argue that these results are not surprising since the bids were generated by a locally linear function of the feature vectors, thereby ensuring the success of our algorithm. Nevertheless, one would expect this to be the case too for algorithms that leverage the use of features such as the convex surrogate and ridge regression. But one can see that this is in fact not true even for low levels of noise. It is also worth noticing that the use of ridge regression is actually worse than setting the reserve price to 0. This fact can be easily understood by noticing that the square loss used in regression is symmetric. Therefore, we can expect several reserve prices to be above the highest bid, making the revenue of these auctions equal to zero. Another notable feature is that as the noise level increases, the performance of feature-based algorithms decreases. This is true for any learning algorithm: if the features are not relevant to the prediction task, the performance of the algorithm will suffer. However, for the convex surrogate algorithm, a more critical issue occurs: the performance of this algorithm actually decreases as the sample size increases, which shows that in general learning with a convex surrogate is not possible. This is an empirical verification of the inconsistency result provided in Section 4.2. This lack of calibration can also be seen in Figure 6(d), where in fact the performance of this algorithm approaches the use of no reserve price.

In order to better understand the reason behind the performance discrepancy between feature-based algorithms, we analyze the reserve prices offered by each algorithm. In Figure 7 we see that the convex surrogate

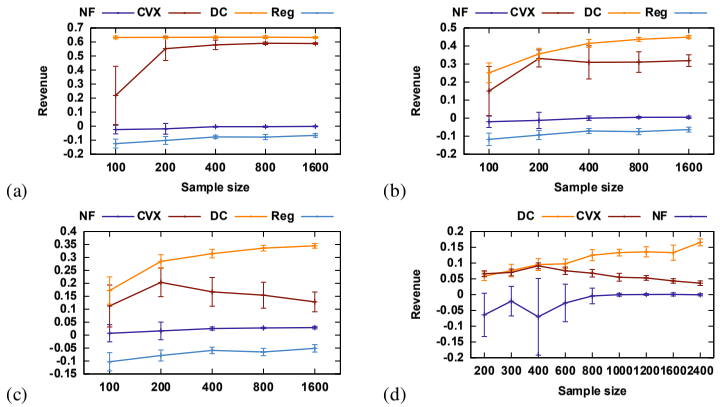


Figure 6: Plots of expected revenue against sample size for different algorithms: DC algorithm (DC), convex surrogate (CVX), ridge regression (Reg) and the algorithm that uses no feature to set reserve prices (NF). For (a)-(c) bids are generated with different noise standard deviation (a) 0, (b) 0.25, (c) 0.5. The bids in (d) were generated using a generative model.

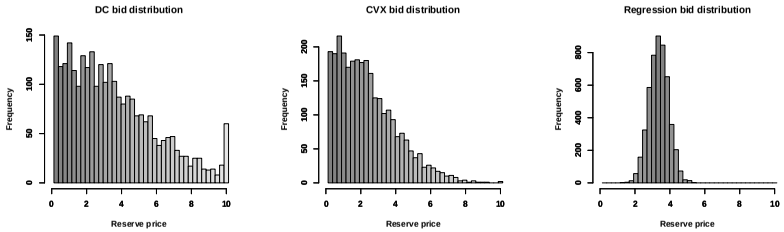


Figure 7: Distribution of reserve prices for each algorithm. The algorithms were trained on 800 samples using noisy bids with standard deviation 0.5.

algorithm tends to offer lower reserve prices. This should be intuitively clear as high reserve prices are over-penalized by the chosen convex surrogate as shown in Figure 2(b). On the other hand, reserve prices suggested by the regression algorithm seem to be concentrated and symmetric around their mean. Therefore we can infer that about 50% of the reserve prices offered will be higher than the highest bid thereby yielding zero revenue. Finally, our algorithm seems to generally offer higher prices. This suggests that the increase in revenue comes from auctions where the highest bid is large but the second bid is small. This bidding phenomenon is in fact commonly observed in practice (Amin et al., 2013).

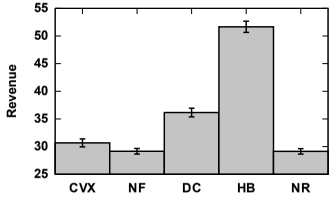


Figure 8: Results of the eBay data set. Comparison of our algorithm (DC) against a convex surrogate (CVX), using no features (NF), setting no reserve (NR) and setting reserve price to highest bid (HB).

6.2 Real-World Data Sets

Due to proprietary data and confidentiality reasons, we cannot present empirical results for AdExchange data. However, we were able to procure an eBay data set consisting of approximately 70,000 second-price auctions of collector sport cards. The full data set can be accessed using the following URL: `http://cims.nyu.edu/~munoz/data`. Some other sources of auction data are accessible (e.g., `http://modelingonlineauctions.com/datasets`), but features are not available for those data sets. To the best of our knowledge, with the exception of the one used here, there is no publicly available data set for online auctions including features that could be readily used with our algorithm. The features used here include information about the seller such as positive feedback percent, seller rating and seller country; as well as information about the card such as whether the player is in the sport's Hall of Fame. The final dimension of the feature vectors is 78. The values of these features are both continuous and categorical. For our experiments we also included an extra offset feature.

Since the highest bid is not reported by eBay, our algorithm cannot be straightforwardly used on this data set. In order to generate highest bids, we calculated the mean price of each object (each card was generally sold more than once) and set the highest bid to be the maximum between this average and the second highest bid.

Figure 8 shows the revenue gained using different algorithms including our DC algorithm, using a convex surrogate, or the algorithm that ignores features. It also shows the results obtained by using no reserve price (NR) and the highest possible revenue obtained by setting the reserve price to the highest bid (HB). We randomly sampled 2,000 examples for training, 2,000 examples for validation and 2,000 examples for testing. This experiment was repeated 10 times. Figure 8(b) shows the mean revenue for each algorithm and their standard deviations. The results of this experiment show that the use of features is crucial for revenue optimization. Indeed, setting an optimal reserve price for all objects seems to achieve the same revenue as no reserve price. Instead, our algorithm achieves a 22% increase on the revenue obtained by not setting a reserve price whereas the non-calibrated convex surrogate algorithm only obtains a 3% revenue improvement. Furthermore, our algorithm is able to obtain as much as 70% of the achievable revenue with knowledge of the highest bid.

7. Conclusion

We presented a comprehensive theoretical and algorithmic analysis of the learning problem of revenue optimization in second-price auctions with reserve. The specific properties of the loss function for this problem required a new analysis and new learning guarantees. The algorithmic solutions we presented are practically applicable to revenue optimization problems for this type of auctions in most realistic settings. Our experimental results further demonstrate their effectiveness. Much of the analysis and algorithms presented, in particular our study of calibration questions, can also be of interest in other learning problems. In particular, they are relevant to the study of learning problems arising in the study of generalized second-price auctions (Mohri and Medina, 2015).

Acknowledgments

We thank Afshin Rostamizadeh and Umar Syed for several discussions about the topic of this work and ICML and JMLR reviewers for useful comments. We also warmly thank Jay Grossman for providing us access to the eBay data set used in this paper. Finally, we thank Le Thi Hoai An for extensive discussions on DC programming. This work was partly funded by the NSF award IIS-1117591.

Appendix A. Contraction Lemma

The following is a version of Talagrand's contraction lemma Ledoux and Talagrand (2011). Since our definition of Rademacher complexity does not use absolute values, we give an explicit proof below.

Lemma 14 *Let H be a hypothesis set of functions mapping $\mathcal{X}$ to $\mathbb{R}$ and $\Psi_1, \ldots, \Psi_m$, μ-Lipschitz functions for some $\mu > 0$. Then, for any sample S of m points $x_1, \ldots, x_m \in \mathcal{X}$, the following inequality holds*

$$\frac{1}{m}\mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h\in H}\sum_{i=1}^{m}\sigma_i(\Psi_i\circ h)(x_i)\right] \leq \frac{\mu}{m}\mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h\in H}\sum_{i=1}^{m}\sigma_i h(x_i)\right] = \mu\,\widehat{\mathfrak{R}}_S(H).$$

Proof The proof is similar to the case where the functions Ψ_i are all equal. Fix a sample $S = (x_1, \ldots, x_m)$. Then, we can rewrite the empirical Rademacher complexity as follows:

$$\frac{1}{m}\mathop{\mathbb{E}}_{\boldsymbol{\sigma}}\left[\sup_{h\in H}\sum_{i=1}^{m}\sigma_i(\Psi_i\circ h)(x_i)\right] = \frac{1}{m}\mathop{\mathbb{E}}_{\sigma_1,\ldots,\sigma_{m-1}}\left[\mathop{\mathbb{E}}_{\sigma_m}\left[\sup_{h\in H} u_{m-1}(h) + \sigma_m(\Psi_m\circ h)(x_m)\right]\right],$$

where $u_{m-1}(h) = \sum_{i=1}^{m-1}\sigma_i(\Psi_i\circ h)(x_i)$. Assume that the suprema can be attained and let $h_1, h_2 \in H$ be the hypotheses satisfying

$$u_{m-1}(h_1) + \Psi_m(h_1(x_m)) = \sup_{h\in H} u_{m-1}(h) + \Psi_m(h(x_m))$$
$$u_{m-1}(h_2) - \Psi_m(h_2(x_m)) = \sup_{h\in H} u_{m-1}(h) - \Psi_m(h(x_m)).$$

When the suprema are not reached, a similar argument to what follows can be given by considering instead hypotheses that are ϵ-close to the suprema for any $\epsilon > 0$.

By definition of expectation, since σ_m uniform distributed over $\{-1, +1\}$, we can write

$$\begin{aligned}\mathop{\mathbb{E}}_{\sigma_m}\left[\sup_{h\in H} u_{m-1}(h) + \sigma_m(\Psi_m\circ h)(x_m)\right] &= \frac{1}{2}\sup_{h\in H} u_{m-1}(h) + (\Psi_m\circ h)(x_m) + \frac{1}{2}\sup_{h\in H} u_{m-1}(h) - (\Psi_m\circ h)(x_m)\\ &= \frac{1}{2}[u_{m-1}(h_1) + (\Psi_m\circ h_1)(x_m)] + \frac{1}{2}[u_{m-1}(h_2) - (\Psi_m\circ h_2)(x_m)].\end{aligned}$$

Let $s = \operatorname{sgn}(h_1(x_m) - h_2(x_m))$. Then, the previous equality implies

$$\begin{aligned}\mathop{\mathbb{E}}_{\sigma_m}\left[\sup_{h\in H} u_{m-1}(h) + \sigma_m(\Psi_m\circ h)(x_m)\right] &\leq \frac{1}{2}[u_{m-1}(h_1) + u_{m-1}(h_2) + s\mu(h_1(x_m) - h_2(x_m))]\\ &= \frac{1}{2}[u_{m-1}(h_1) + s\mu h_1(x_m)] + \frac{1}{2}[u_{m-1}(h_2) - s\mu h_2(x_m)]\\ &\leq \frac{1}{2}\sup_{h\in H}[u_{m-1}(h) + s\mu h(x_m)] + \frac{1}{2}\sup_{h\in H}[u_{m-1}(h) - s\mu h(x_m)]\\ &= \mathop{\mathbb{E}}_{\sigma_m}\left[\sup_{h\in H} u_{m-1}(h) + \sigma_m\mu h(x_m)\right],\end{aligned}$$

where we used the $\mu-$Lipschitzness of Ψ_m in the first equality and the definition of expectation over σ_m for the last equality. Proceeding in the same way for all other σ_i's ($i \neq m$) proves the lemma. ■

Appendix B. Proof of Theorem 7

Proof We first show that the functions L_n are uniformly bounded for any b:

$$\begin{aligned}|L_n(r,b)| = \left|\int_0^r L_n'(r,b)dr\right| &\leq \int_0^M \max\left(\left|L_n'(0,b)\right|, \left|L_n'(M,b)\right|\right)dr\\ &\leq \int_0^M K dr = MK,\end{aligned}$$

where the first inequality holds since, by convexity, the derivative of L_n with respect to r is an increasing function.

Next, we show that the sequence $(L_n)_{n\in\mathbb{N}}$ is also equicontinuous. It will follow then by the theorem of Arzela-Ascoli that the sequence $L_n(\cdot, b)$ converges uniformly to $L_c(\cdot, b)$. Let $r_1, r_2 \in [0, M]$, for any $b \in [0, M]$ we have

$$\begin{aligned}|L_n(r_1, b) - L_n(r_2, b)| &\leq \sup_{r\in[0,M]} \left|L_n'(r, b)\right| |r_1 - r_2| \\ &= \max\left(\left|L_n'(0, b)\right|, \left|L_n'(M, b))\right|\right) |r_1 - r_2| \\ &\leq K|r_1 - r_2|,\end{aligned}$$

where, again, the convexity of L_n was used for the first equality. Let $F_n(r) = \mathbb{E}_{b\sim D}[L_n(r, b)]$ and $F(r) = \mathbb{E}_{b\sim D}[L_c(r, b)]$. F_n is a convex function as the expectation of a convex function. By the theorem of Arzela-Ascoli, the sequence $(F_n)_n$ admits a uniformly convergent subsequence. Furthermore, by the dominated convergence theorem, we have $(F_n(r))_n$ converges pointwise to $F(r)$. Therefore, the uniform limit of F_n must be F. This implies that

$$\min_{r\in[0,M]} F(r) = \lim_{n\to+\infty} \min_{r\in[0,M]} F_n(r) = \lim_{n\to+\infty} F_n(r_n) = F(r^*),$$

where the first and third equalities follow from the uniform convergence of F_n to F. The last equation implies that L_c is consistent with L. Furthermore, the function $L_c(\cdot, b)$ is convex since it is the uniform limit of convex functions. It then follows by Proposition 6 that $L_c(\cdot, b) \equiv L_c(0, b) = 0$. ■

Appendix C. Consistency of L_γ

Lemma 15 *Let H be a closed, convex subset of a linear space of functions containing 0 and let $h^*_\gamma = \operatorname{argmin}_{h\in H} \mathcal{L}_\gamma(h)$. Then, the following inequality holds:*

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right] \geq \frac{1}{\gamma} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\right].$$

Proof Let $0 < \lambda < 1$. Since H is a convex set, it follows that $\lambda h^*_\gamma \in H$. Furthermore, by the definition of h^*_γ, we must have:

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})\right] \leq \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b})\right]. \tag{14}$$

If $h^*_\gamma(\mathbf{x}) < 0$, then $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) = L_\gamma(\lambda h^*_\gamma(\mathbf{x})) = -b^{(2)}$ by definition of L_γ. If on the other hand $h^*_\gamma(\mathbf{x}) > 0$, since $\lambda h^*_\gamma(\mathbf{x}) < h^*_\gamma(\mathbf{x})$, we must have that for $(\mathbf{x}, \mathbf{b}) \in I_1$ $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) = L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) = -b^{(2)}$ too. Moreover, from the fact that $L_\gamma \leq 0$ and $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) = 0$ for $(\mathbf{x}, \mathbf{b}) \in I_4$ it follows that $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) \geq L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b})$ for $(\mathbf{x}, \mathbf{b}) \in I_4$, and therefore the following inequality trivially holds:

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})(\mathbb{1}_{I_1}(\mathbf{x}, \mathbf{b}) + \mathbb{1}_{I_4}(\mathbf{x}, \mathbf{b}))\right] \geq \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b})(\mathbb{1}_{I_1}(\mathbf{x}, \mathbf{b}) + \mathbb{1}_{I_4}(\mathbf{x}, \mathbf{b}))\right]. \tag{15}$$

Subtracting (15) from (14) we obtain

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})(\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b}) + \mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b}))\right] \leq \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b})(\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b}) + \mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b}))\right].$$

Rearranging terms shows that this inequality is equivalent to

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[(L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}))\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right] \geq \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[(L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}))\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\right] \tag{16}$$

Notice that if $(\mathbf{x}, \mathbf{b}) \in I_2$, then $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) = -h^*_\gamma(\mathbf{x})$. If $\lambda h^*_\gamma(\mathbf{x}) > b^{(2)}$ too then $L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) = -\lambda h^*_\gamma(\mathbf{x})$. On the other hand if $\lambda h^*_\gamma(\mathbf{x}) \leq b^{(2)}$ then $L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) = -b^{(2)} \leq -\lambda h^*_\gamma(\mathbf{x})$. Thus

$$\mathbb{E}(L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}))\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})) \leq (1 - \lambda)\,\mathbb{E}(h^*_\gamma(\mathbf{x})\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})) \tag{17}$$

This gives an upper bound for the left-hand side of inequality (16). We now seek to derive a lower bound on the right-hand side. To do so, we analyze two different cases:

1. $\lambda h^*_\gamma(\mathbf{x}) \leq b^{(1)}$;
2. $\lambda h^*_\gamma(\mathbf{x}) > b^{(1)}$.

In the first case, we know that $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) = \frac{1}{\gamma}(h^*_\gamma(\mathbf{x}) - (1 + \gamma)b^{(1)}) > -b^{(1)}$ (since $h^*_\gamma(\mathbf{x}) > b^{(1)}$ for $(\mathbf{x}, \mathbf{b}) \in I_3$). Furthermore, if $\lambda h^*_\gamma(\mathbf{x}) \leq b^{(1)}$, then, by definition $L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) = \min(-b^{(2)}, -\lambda h^*_\gamma(\mathbf{x})) \leq -\lambda h^*_\gamma(\mathbf{x})$. Thus, we must have:

$$L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) > \lambda h^*_\gamma(\mathbf{x}) - b^{(1)} > (\lambda - 1)b^{(1)} \geq (\lambda - 1)M, \tag{18}$$

where we used the fact that $h^*_\gamma(\mathbf{x}) > b^{(1)}$ for the second inequality and the last inequality holds since $\lambda - 1 < 0$.

We analyze the second case now. If $\lambda h^*_\gamma(\mathbf{x}) > b^{(1)}$, then for $(\mathbf{x}, \mathbf{b}) \in I_3$ we have $L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b}) = \frac{1}{\gamma}(1 - \lambda)h^*_\gamma(\mathbf{x})$. Thus, letting $\Delta(\mathbf{x}, \mathbf{b}) = L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(\lambda h^*_\gamma(\mathbf{x}), \mathbf{b})$, we can lower bound the right-hand side of (16) as:

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[\Delta(\mathbf{x}, \mathbf{b})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\right] &= \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[\Delta(\mathbf{x}, \mathbf{b})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\mathbb{1}_{\{\lambda h^*_\gamma(\mathbf{x}) > b^{(1)}\}}\right] + \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[\Delta(\mathbf{x}, \mathbf{b})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\mathbb{1}_{\{\lambda h^*_\gamma(\mathbf{x}) \leq b^{(1)}\}}\right] \\ &\geq \frac{1 - \lambda}{\gamma} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\mathbb{1}_{\{\lambda h^*_\gamma(\mathbf{x}) > b^{(1)}\}}\right] + (\lambda - 1)M\,\mathbb{P}\left[h^*_\gamma(\mathbf{x}) > b^{(1)} \geq \lambda h^*_\gamma(\mathbf{x})\right],\end{aligned} \tag{19}$$

where we have used (18) to bound the second summand. Combining inequalities (16), (17) and (19) and dividing by $(1 - \lambda)$ we obtain the bound

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right] \geq \frac{1}{\gamma} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\mathbb{1}_{\{\lambda h^*_\gamma(\mathbf{x}) > b^{(1)}\}}\right] - M\,\mathbb{P}\left[h^*_\gamma(\mathbf{x}) > b^{(1)} \geq \lambda h^*_\gamma(\mathbf{x})\right].$$

Finally, taking the limit $\lambda \to 1$, we obtain

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_2}(\mathbf{x}, \mathbf{b})\right] \geq \frac{1}{\gamma} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_3}(\mathbf{x}, \mathbf{b})\right].$$

Taking the limit inside the expectation is justified by the bounded convergence theorem and $\mathbb{P}[h^*_\gamma(\mathbf{x}) > b^{(1)} \geq \lambda h^*_\gamma(\mathbf{x})] \to 0$ holds by the continuity of probability measures. ■

Proposition 16 *For any $\delta > 0$, with probability at least $1 - \delta$ over the choice of a sample S of size m, the following holds for all $\gamma \in (0, 1]$ and $h \in H$:*

$$\mathcal{L}_\gamma(h) \leq \widehat{\mathcal{L}}_\gamma(h) + \frac{2}{\gamma}\mathfrak{R}_m(H) + M\left[\sqrt{\frac{\log\log_2 \frac{1}{\gamma}}{m}} + \sqrt{\frac{\log \frac{1}{\delta}}{2m}}\right].$$

Proof Consider two sequences $(\gamma_k)_{k\geq 1}$ and $(\epsilon_k)_{k\geq 1}$, with $\epsilon_k \in (0, 1)$. By Theorem 9, for any fixed $k \geq 1$,

$$\mathbb{P}\left[\mathcal{L}_{\gamma_k}(h) - \widehat{\mathcal{L}}_{\gamma_k}(h) > \frac{2}{\gamma_k}\mathfrak{R}_m(H) + M\epsilon_k\right] \leq \exp(-2m\epsilon_k^2).$$

Choose $\epsilon_k = \epsilon + \sqrt{\frac{\log k}{m}}$, then, by the union bound,

$$\begin{aligned}\mathbb{P}\left[\exists k\colon \mathcal{L}_{\gamma_k}(h) - \widehat{\mathcal{L}}_{\gamma_k}(h) > \frac{1}{\gamma_k}\mathfrak{R}_m(H) + M\epsilon_k\right] &\leq \sum_{k\geq 1} \exp\left[-2m(\epsilon + \sqrt{(\log k)/m})^2\right] \\ &\leq \left(\sum_{k\geq 1} 1/k^2\right) \exp(-2m\epsilon^2) \\ &= \frac{\pi^2}{6}\exp(-2m\epsilon^2) \leq 2\exp(-2m\epsilon^2).\end{aligned}$$

For any $\gamma \in (0,1]$, there exists $k \geq 1$ such that $\gamma \in (\gamma_k, \gamma_{k-1})$ with $\gamma_k = 1/2^k$. For such a k, $\frac{1}{\gamma_{k-1}} \leq \frac{1}{\gamma}$, $\gamma_{k-1} \leq \frac{\gamma}{2}$, and $\sqrt{\log(k-1)} = \sqrt{\log\log_2(1/\gamma_{k-1})} \leq \sqrt{\log\log_2(1/\gamma)}$. Since for any $h \in H$, $\mathcal{L}_{\gamma_{k-1}}(h) \leq \mathcal{L}_\gamma(h)$, we can write

$$\mathbb{P}\left[\mathcal{L}(h) - \widehat{\mathcal{L}}_\gamma(h) > \frac{2}{\gamma}\mathfrak{R}_m(H) + M\Big(K(\gamma) + \epsilon\Big)\right] \leq \exp(-2m\epsilon^2),$$

where $K(\gamma) = \sqrt{\frac{\log\log_2 \frac{1}{\gamma}}{m}}$. This concludes the proof. ■

Corollary 17 *Let H be a hypothesis set with pseudo-dimension $d = Pdim(H)$. Then, for any $\delta > 0$ and any $\gamma > 0$, with probability at least $1-\delta$ over the choice of a sample S of size m, the following inequality holds:*

$$\mathcal{L}(\widehat{h}_\gamma) \leq \mathcal{L}^* + \frac{2\gamma+2}{\gamma}\mathfrak{R}_m(H) + \gamma M + M\left[2\sqrt{\frac{2d\log\frac{em}{d}}{m}} + 2\sqrt{\frac{\log\frac{2}{\delta}}{2m}} + \sqrt{\frac{\log\log_2\frac{1}{\gamma}}{m}}\right].$$

The proof follows the same steps as Theorem 10 and uses the results of Proposition 16. Notice that, by setting $\gamma = \frac{1}{m^{1/4}}$, we can guarantee the convergence of $\mathcal{L}(\widehat{h}_\gamma)$ to $\mathcal{L}^*$. Indeed, with this choice, the bound can be expressed as follows:

$$\mathcal{L}(\widehat{h}_\gamma) \leq \mathcal{L}^* + (2+m^{1/4})\mathfrak{R}_m(H) + \frac{1}{m^{1/4}}M + M\left[2\sqrt{\frac{2d\log\frac{em}{d}}{m}} + 2\sqrt{\frac{\log\frac{2}{\delta}}{2m}} + \sqrt{\frac{\log\log_2 m^{1/4}}{m}}\right].$$

Furthermore, when H has finite pseudo-dimension, it is known that $\mathfrak{R}_m(H)$ is in $O\big(\frac{1}{m^{1/2}}\big)$. Thus, this shows that $\mathcal{L}(\widehat{h}_\gamma) = \mathcal{L}^* + O\big(\frac{1}{m^{1/4}}\big)$.

Appendix D. Proof of Proposition 12

Proof From the definition of v-function, it is immediate that V_i is differentiable everywhere except at the three points $n_i^{(1)} = b_i^{(2)}, n_i^{(2)} = b_i^{(1)}$ and $n_i^{(3)} = (1+\eta)b_i^{(1)}$. Let r^* be a minimizer of F. If $r^* \neq n_i^{(j)}$ for every $j \in \{1,2,3\}$ and $i \in \{1,\ldots,m\}$, then F must be differentiable at r^* and $F'(r^*) = 0$. Now, let $n^* = \max\{n_i^{(j)} | n_i^{(j)} < r^*\}$. Since F is a linear function over the interval $(n^*, r^*]$, we must have $F'(r) = F'(r^*) = 0$ for every $r \in (n^*, r^*]$. Thus, F reduces to a constant over this interval and continuity of F implies that $F(n^*) = F(r^*)$.

We conclude the proof by showing that n^* is equal to $b_i^{(1)}$ for some i. Suppose this is not the case and let U be an open interval around n^* satisfying $b_i^{(1)} \notin U$ for all i. It is not hard to verify that V_i is a concave function over every interval not containing $b_i^{(1)}$. In particular V_i is concave over U for any i and, as a sum of concave functions, F is concave too over the interval U. Moreover, by definition, n^* minimizes F restricted to U. This implies that F is constant over U as a non-constant concave function cannot reach its minimum over an open set. Finally, let $b^* = \text{argmin}_i |n^* - b_i^{(1)}|$. Since U was an arbitrary open interval, it follows that there exists r arbitrarily close to b^* such that $F(r) = F(n^*)$. By the continuity of F, we must then have $F(b^*) = F(n^*)$. ■

References

Kareem Amin, Michael Kearns, Peter Key, and Anton Schwaighofer. Budget optimization for sponsored search: Censored learning in MDPs. In *Proceedings of UAI 2012*, pages 54–63, 2012.

Kareem Amin, Afshin Rostamizadeh, and Umar Syed. Learning prices for repeated auctions with strategic buyers. In *Proceedings of NIPS 2012*, pages 1169–1177, 2013.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002.

Maria-Florina Balcan, Avrim Blum, Jason D. Hartline, and Yishay Mansour. Reducing mechanism design to algorithm design via machine learning. *J. Comput. Syst. Sci.*, 74(8):1245–1270, 2008.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. Online learning in online auctions. *Theor. Comput. Sci.*, 324(2-3): 137–146, 2004.

Nicolò Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. Regret minimization for reserve prices in second-price auctions. In *Proceedings of SODA 2013*, pages 1190–1204, 2013.

Ying Cui, Ruofei Zhang, Wei Li, and Jianchang Mao. Bid landscape forecasting in online ad exchange marketplace. In *Proceedings of KDD 2011*, pages 265–273, 2011.

Gerard Debreu and Tjalling C. Koopmans. Additively decomposed quasiconvex functions. *Mathematical Programming*, 24, 1982.

Nikhil R. Devanur and Sham M. Kakade. The price of truthfulness for pay-per-click auctions. In *Proceedings of EC 2009*, pages 99–106, 2009.

David A. Easley and Jon M. Kleinberg. *Networks, Crowds, and Markets - Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

Di He, Wei Chen, Liwei Wang, and Tie-Yan Liu. Online learning for auction mechanism in bandit setting. *Decision Support Systems*, 56:379–386, 2013.

R Horst and Nguyen V Thoai. DC programming: overview. *Journal of Optimization Theory and Applications*, 103(1): 1–43, 1999.

Robert D. Kleinberg and Frank Thomson Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *Proceedings of FOCS 2003*, pages 594–605, 2003.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 2002.

Sébastien Lahaie and David M. Pennock. Revenue analysis of a family of ranking rules for keyword auctions. In *Proceedings of EC 2007*, pages 50–56, 2007.

John Langford, Lihong Li, Yevgeniy Vorobeychik, and Jennifer Wortman. Maintaining equilibria during exploration in sponsored search auctions. *Algorithmica*, 58(4):990–1021, 2010.

Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces*. Classics in Mathematics. Springer-Verlag, Berlin, 2011. Isoperimetry and processes, Reprint of the 1991 edition.

P.R. Milgrom and R.J. Weber. A theory of auctions and competitive bidding. *Econometrica: Journal of the Econometric Society*, pages 1089–1122, 1982.

Mehryar Mohri and Andrés Muñoz Medina. Non-parametric revenue optimization for generalized second-price auctions. In *Proceedings of UAI 2015*, Amsterdam, The Netherlands, 2015.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT Press, Cambridge, MA, 2012.

S Muthukrishnan. Ad exchanges: Research issues. *Internet and Network Economics*, pages 1–12, 2009.

R.B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, Cambridge, 2007.

Michael Ostrovsky and Michael Schwarz. Reserve prices in internet advertising auctions: a field experiment. In *Proceedings of EC 2011*, pages 59–60, 2011.

David Pollard. *Convergence of Stochastic Processes*. Springer Series in Statistics. Springer-Verlag, New York, 1984.

J.G. Riley and W.F. Samuelson. Optimal auctions. *The American Economic Review*, pages 381–392, 1981.

Bharath K. Sriperumbudur and Gert R. G. Lanckriet. A proof of convergence of the concave-convex procedure using Zangwill's theory. *Neural Computation*, 24(6):1391–1407, 2012.

Pham Dinh Tao and Le Thi Hoai An. Convex analysis approach to DC programming: theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

Pham Dinh Tao and Le Thi Hoai An. A DC optimization algorithm for solving the trust-region subproblem. *SIAM Journal on Optimization*, 8(2):476–505, 1998.

Hoang Tuy. Concave programming under linear constraints. *Translated Soviet Mathematics*, 5:1437–1440, 1964.

Hoang Tuy. Counter-examples to some results on DC optimization. Technical report, Institute of Mathematics, Hanoi, Vietnam, 2002.

William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961.

Ian E.H. Yen, Nanyun Peng, Po-Wei Wang, and Shou-De Lin. On convergence rate of concave-convex procedure. In *Proceedings of the NIPS 2012 Optimization Workshop*, 2012.

Alan L. Yuille and Anand Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

Yunzhang Zhu, Gang Wang, Junli Yang, Dakan Wang, Jun Yan, Jian Hu, and Zheng Chen. Optimizing search engine revenue in sponsored search. In *Proceedings of ACM-SIGIR 2009*, pages 588–595, 2009.

Journal of Machine Learning Research 17 (2016) 1-25 Submitted 1/15; Published 2/16

Distributed Coordinate Descent Method for Learning with Big Data

Peter Richtárik PETER.RICHTARIK@ED.AC.UK
School of Mathematics
University of Edinburgh
6317 James Clerk Maxwell Building
Peter Guthrie Tait Road
Edinburgh, EH9 3FD

Martin Takáč TAKAC.MT@GMAIL.COM
Industrial and Systems Engineering Department
Lehigh University
H.S. Mohler Laboratory
200 West Packer Avenue
Bethlehem, PA 18015, USA

Editor: Koby Crammer

Abstract

In this paper we develop and analyze Hydra: HYbriD cooRdinAte descent method for solving loss minimization problems with big data. We initially partition the coordinates (features) and assign each partition to a different node of a cluster. At every iteration, each node picks a random subset of the coordinates from those it owns, independently from the other computers, and in parallel computes and applies updates to the selected coordinates based on a simple closed-form formula. We give bounds on the number of iterations sufficient to approximately solve the problem with high probability, and show how it depends on the data and on the partitioning. We perform numerical experiments with a LASSO instance described by a 3TB matrix.

Keywords: stochastic methods, parallel coordinate descent, distributed algorithms, boosting

1. Introduction

Randomized coordinate descent methods (CDMs) are increasingly popular in many learning tasks, including boosting, large scale regression and training linear support vector machines. CDMs update a single randomly chosen coordinate at a time by moving in the direction of the negative partial derivative (for smooth losses). Methods of this type, in various settings, were studied by several authors, including Hsieh et al. (2008); Shalev-Shwartz and Tewari (2009); Nesterov (2012); Richtárik and Takáč (2014); Necoara et al. (2012); Tappenden et al. (2013); Shalev-Shwartz and Zhang (2013b); Lu and Xiao (2015).

It is clear that in order to utilize modern shared-memory parallel computers, more coordinates should be updated at each iteration. One way to approach this is via partitioning the coordinates into blocks, and operating on a single randomly chosen block at a time, utilizing parallel linear algebra libraries. This approach was pioneered by Nesterov (2012) for smooth losses, and was extended to regularized problems in (Richtárik and Takáč, 2014). Another popular approach involves working with a random subset of coordinates (Bradley et al., 2011). These approaches can be combined, and

theory was developed for methods that update a random subset of blocks of coordinates at a time Richtárik and Takáč (2015); Fercoq and Richtárik (2013). Further recent works on parallel coordinate descent include (Richtárik and Takáč, 2012; Mukherjee et al., 2013; Fercoq, 2013; Tappenden et al., 2015; Shalev-Shwartz and Zhang, 2013a).

However, none of these methods are directly scalable to problems of sizes so large that a single computer is unable to store the data describing the instance, or is unable to do so efficiently (e.g., in memory). In a big data scenario of this type, it is imperative to split the data across several nodes (computers) of a cluster, and design efficient methods for this memory-distributed setting.

Hydra. In this work we design and analyze the first distributed coordinate descent method: *Hydra: HYbriD cooRdinAte descent*. The method is "hybrid" in the sense that it uses parallelism at two levels: i) across a number of nodes in a cluster and ii) utilizing the parallel processing power of individual nodes[1].

Assume we have c nodes (computers) available, each with parallel processing power. In Hydra, we initially partition the coordinates $\{1, 2, \dots, d\}$ into c sets, $\mathcal{P}_1, \dots, \mathcal{P}_c$, and assign each set to a single computer. For simplicity, we assume that the partition is balanced: $|\mathcal{P}_k| = |\mathcal{P}_l|$ for all k, l. Each computer *owns* the coordinates belonging to its partition for the duration of the iterative process. Also, these coordinates are stored locally. The data matrix describing the problem is partitioned in such a way that all data describing features belonging to $\mathcal{P}_l$ is stored at computer l. Now, at each iteration, each computer, independently from the others, chooses a random subset of τ coordinates from those they own, and computes and applies updates to these coordinates. Hence, once all computers are done, $c\tau$ coordinates will have been updated. The resulting vector, stored as c vectors of size $s = d/c$ each, in a distributed way, is the new iterate. This process is repeated until convergence. It is important that the computations are done locally on each node, with minimum communication overhead. We comment on this and further details in the text.

The main insight. We show that the parallelization potential of Hydra, that is, its ability to accelerate as τ is increased, depends on two data-dependent quantities: i) the *spectral norm of the data* (σ) and ii) a *partition-induced norm of the data* (σ'). The first quantity completely describes the behavior of the method in the $c = 1$ case. If σ is small, then utilization of more processors (i.e., increasing τ) leads to nearly linear speedup. If σ is large, speedup may be negligible, or there may be no speedup whatsoever. Hence, the size of σ suggests whether it is worth to use more processors or not. The second quantity, σ', characterizes the effect of the initial partition on the algorithm, and as such is relevant in the $c > 1$ case. Partitions with small σ' are preferable. We show that, surprisingly, that as long as $\tau \geq 2$, the effect of a bad partitioning is that it most doubles the number of iterations of Hydra. Hence, data partitioning can be used to optimize for different aspects of the method, such as reducing communication complexity, if needed.

For all of these quantities we derive easily computable and interpretable estimates (ω for σ and ω' for σ'), which may be used by practitioners to gauge, a-priori, whether their problem of interest is likely to be a good fit for Hydra or not. We show that for strongly convex losses, Hydra outputs an ϵ-accurate solution with probability at least $1 - \rho$ after

$$\frac{d\beta}{c\tau\mu} \log\left(\frac{1}{\epsilon\rho}\right)$$

iterations (we ignore some small details here), where a single iteration corresponds to changing of τ coordinates by each of the c nodes; β is a stepsize parameter and μ is a strong convexity constant.

1. We like to think of each node of the cluster as one of the many heads of the mythological Hydra.

square loss (SL)	$\frac{1}{2}(y^j - \mathbf{A}_{j:}x)^2$
logistic loss (LL)	$\log(1 + \exp(-y^j \mathbf{A}_{j:}x))$
square hinge loss (HL)	$\frac{1}{2}\max\{0, 1 - y^j \mathbf{A}_{j:}x\}^2$

Table 1: Examples of loss functions ℓ covered by our analysis.

Outline. In Section 2 we describe the structure of the optimization problem we consider in this paper and state assumptions. We then proceed to Section 3, in which we describe the method. In Section 4 we prove bounds on the number of iterations sufficient for Hydra to find an approximate solution with arbitrarily high probability. A discussion of various aspects of our results, as well as a comparison with existing work, can be found in Section 5. Implementation details of our distributed communication protocol are laid out in Section 6. Finally, we comment on our computational experiments with a big data (3TB matrix) L1 regularized least-squares instance in Section 7.

2. The Problem

We study the problem of minimizing regularized loss,

$$\min_{x \in \mathbb{R}^d} L(x) := f(x) + R(x), \tag{1}$$

where f is a smooth convex loss, and R is a convex (and possibly nonsmooth) regularizer.

2.1 Loss Function f

We assume that there exists a positive definite matrix $\mathbf{M} \in \mathbb{R}^{d \times d}$ such that for all $x, h \in \mathbb{R}^d$,

$$f(x + h) \leq f(x) + (f'(x))^T h + \tfrac{1}{2} h^T \mathbf{M} h, \tag{2}$$

and write $\mathbf{M} = \mathbf{A}^T \mathbf{A}$, where $\mathbf{A}$ is some n-by-d matrix.

Example. These assumptions are natural satisfied in many popular problems. A typical loss function has the form

$$f(x) = \sum_{j=1}^{n} \ell(x, \mathbf{A}_{j:}, y^j), \tag{3}$$

where $\mathbf{A} \in \mathbb{R}^{n \times d}$ is a matrix encoding n examples with d features, $\mathbf{A}_{j:}$ denotes j-th row of $\mathbf{A}$, ℓ is some loss function acting on a single example and $y \in \mathbb{R}^n$ is a vector of labels. For instance, in the case of the three losses ℓ in Table 1, assumption (2) holds with $\mathbf{M} = \mathbf{A}^T \mathbf{A}$ for SL and HL, and $\mathbf{M} = \frac{1}{4}\mathbf{A}^T \mathbf{A}$ for LL (Bradley et al., 2011).

2.2 Regularizer R

We assume that R is separable, i.e., that it can be decomposed as $R(x) = \sum_{i=1}^{d} R_i(x^i)$, where x^i is the i-th coordinate of x, and the functions $R_i : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ are convex and closed.

Example. The choice $R_i(t) = 0$ for $t \in [0, 1]$ and $R_i(t) = +\infty$, otherwise, effectively models bound constraints, which are relevant for SVM dual. Other popular choices are $R(x) = \lambda \|x\|_1$ (L1-regularizer) and $R(x) = \frac{\lambda}{2}\|x\|_2^2$ (L2-regularizer).

3. Distributed Coordinate Descent

We consider a setup with c computers (nods) and first partition the d coordinates (features) into c sets $\mathcal{P}_1, \ldots, \mathcal{P}_c$ of equal cardinality, $s := d/c$, and assign set $\mathcal{P}_l$ to node l. Hydra is described in Algorithm 1. Hydra's convergence rate depends on the partition; we comment on this later in Sections 4 and 5. Here we simply assume that we work with a fixed partition. We now comment on the steps.

Algorithm 1: Hydra: HYbriD cooRdinAte descent

Parameters: $x_0 \in \mathbb{R}^d$; $\{\mathcal{P}_1, \ldots, \mathcal{P}_c\}$; $\beta > 0$, τ; $k \leftarrow 0$;
repeat
  $x_{k+1} \leftarrow x_k$;
  for each *computer* $l \in \{1, \ldots, c\}$ **in parallel do**
    Pick a random set of coordinates $\hat{S}_l \subseteq \mathcal{P}_l$, $|\hat{S}_l| = \tau$
    for each $i \in \hat{S}_l$ **in parallel do**
      $h_k^i \leftarrow \arg\min_t f_i'(x_k)t + \frac{M_{ii}\beta}{2}t^2 + R_i(x_k^i + t)$;
      Apply the update: $x_{k+1}^i \leftarrow x_{k+1}^i + h_k^i$;
until *happy*;

Step 3. Here we are just establishing a way of labeling the iterates. Starting with x_k, all c computers modify $c\tau$ entries of x_k in total, in a distributed way, and the result is called x_{k+1}. No computer is allowed to proceed before all computers are done with computing their updates. The resulting vector, x_{k+1}, is the new iterate. Note that, due to this, our method is inherently *synchronous*. In practice, a carefully designed asynchronous implementation will be faster, and our experiments in Section 7 are done with such an implementation.

Steps 4–5. At every iteration, each of the c computers picks a random subset of τ features from those that it owns, uniformly at random, independently of the choice of the other computers. Let $\hat{S}_l$ denote the set picked by node l . More formally, we require that i) $\hat{S}_l \subseteq \mathcal{P}_l$, ii) $\mathbf{Prob}(|\hat{S}_l| = \tau) = 1$, where $1 \leq \tau \leq s$, and that iii) all subsets of $\mathcal{P}_l$ of cardinality τ are chosen equally likely. In summary, at every iteration of the method, features belonging to the random set $\hat{S} := \cup_{l=1}^{c} \hat{S}_l$ are updated. Note that $\hat{S}$ has size $c\tau$, but that, as a sampling from the set $\{1, 2, \ldots, d\}$, it does not choose all cardinality $c\tau$ subsets of $\{1, 2, \ldots, d\}$ with equal probability. Hence, the analysis of parallel coordinate descent methods of Richtárik and Takáč (2015) does not apply. We will say that $\hat{S}$ is a *τ-distributed sampling* with respect to the partition $\{\mathcal{P}_1, \ldots, \mathcal{P}_c\}$.

Step 6. Once computer l has chosen its set of τ coordinates to work on in Step 5, it will *in parallel* compute (Step 7) and apply (Step 8) updates to them.

Step 7. This is a critical step where updates to coordinates $i \in \hat{S}_l$ are computed. By $f_i'(x)$ we denote the i-th partial derivative of f at x. Notice that the formula is very simple as it involves one dimensional optimization.

Closed-form formulas. Often, h_k^i can be computed in closed form. For $R_i(t) = \lambda_i|t|$ (weighted L1 regularizer), h_k^i is the point in the interval

$$\left[\frac{-\lambda_i - f_i'(x_k)}{\mathbf{M}_{ii}\beta}, \frac{\lambda_i - f_i'(x_k)}{\mathbf{M}_{ii}\beta}\right]$$

which is closest to $-x_k^i$. If $R_i(t) = \frac{\lambda_i}{2}t^2$ (weighted L2 regularizer), then $h_k^i = -\frac{f_i'(x_k)+\lambda_i x_k^i}{\lambda_i + \mathbf{M}_{ii}\beta}$.

Choice of β. The choice of the step-size parameter β is of paramount significance for the performance of the algorithm, as argued for different but related algorithms by Richtárik and Takáč (2015); Takáč et al. (2013); Fercoq and Richtárik (2013). We will discuss this issue at length in Sections 4 and 5.

Implementation issues: Note that computer l needs to know the partial derivatives of f at x_k for coordinates $i \in \hat{S}_l \subseteq \mathcal{P}_l$. However, x_k, as well as the data describing f, is distributed among the c computers. One thus needs to devise a fast and communication efficient way of computing these derivatives. This issue will be dealt with in Section 6.

Step 8. Here all the τ updates computed in Step 7 are applied to the iterate. Note that the updates are *local*: computer l only updates coordinates it owns, which are stored locally. Hence, this step is communication-free.

4. Convergence Rate Analysis

Notation: For any $\mathbf{G} \in \mathbb{R}^{d\times d}$, let $D^{\mathbf{G}} = \mathrm{Diag}(\mathbf{G})$. That is, $D_{ii}^{\mathbf{G}} = \mathbf{G}_{ii}$ for all i and $D_{ij}^{\mathbf{G}} = 0$ for $i \neq j$. Further, let $B^{\mathbf{G}} \in \mathbb{R}^{d\times d}$ be the block diagonal of $\mathbf{G}$ associated with the partition $\{\mathcal{P}_1, \ldots, \mathcal{P}_c\}$. That is, $B_{ij}^{\mathbf{G}} = \mathbf{G}_{ij}$ whenever $i, j \in \mathcal{P}_l$ for some l, and $B_{ij}^{\mathbf{G}} = 0$ otherwise.

4.1 Four Important Quantities: $\sigma', \omega', \sigma, \omega$

Here we define two quantities, σ' and σ, which, as we shall see, play an important role in the computation of the stepsize parameter β of Algorithm 1, and through it, in understanding its rate of convergence and potential for speedup by parallelization and distribution. As we shall see, these quantities might not be easily computable. We therefore also provide each with an easily computable and interpretable upper bound, ω' for σ' and ω for σ.

Let

$$\mathbf{Q} := (D^{\mathbf{M}})^{-1/2}\mathbf{M}(D^{\mathbf{M}})^{-1/2}, \tag{4}$$

and notice that, by construction, $\mathbf{Q}$ has ones on the diagonal. Since M is positive definite, $\mathbf{Q}$ is as well. For each $l \in \{1, \ldots, c\}$, let $\mathbf{A}_l \in \mathbb{R}^{n\times s}$ be the column submatrix of $\mathbf{A}$ corresponding to coordinates $i \in \mathcal{P}_l$. The diagonal blocks of $B^{\mathbf{Q}}$ are the matrices $\mathbf{Q}^{ll}$, $l = 1, 2, \ldots, c$, where

$$\mathbf{Q}^{kl} := (D^{\mathbf{A}_k^T\mathbf{A}_k})^{-1/2}\mathbf{A}_k^T\mathbf{A}_l(D^{\mathbf{A}_l^T\mathbf{A}_l})^{-1/2} \in \mathbb{R}^{s\times s} \tag{5}$$

for each $k, l \in \{1, 2, \ldots, c\}$. We now define

$$\sigma' := \max\{x^T\mathbf{Q}x \ : \ x \in \mathbb{R}^d, \ x^T B^{\mathbf{Q}}x \leq 1\}, \tag{6}$$

$$\sigma := \max\{x^T\mathbf{Q}x \ : \ x \in \mathbb{R}^d, \ x^T x \leq 1\}. \tag{7}$$

A useful consequence of (6) is the inequality

$$x^T(\mathbf{Q} - B^{\mathbf{Q}})x \leq (\sigma' - 1)x^T B^{\mathbf{Q}}x. \tag{8}$$

4.1.1 SPARSITY

Let a_{rl} be the r-th row of $\mathbf{A}_l$, and define

$$\omega' := \max_{1\leq r\leq n}\left\{\omega'(r) := |\{l \ : \ l \in \{1, \ldots, c\}, \ a_{rl} \neq 0\}|\right\},$$

where $\omega'(r)$ is the number of matrices $\mathbf{A}_l$ with a nonzero in row r. Likewise, define

$$\omega := \max_{1\leq r\leq n}\left\{\omega(r) := |\{l \ : \ l \in \{1, \ldots, c\}, \ \mathbf{A}_{rl} \neq 0\}|\right\},$$

where $\omega(r)$ is the number of nonzeros in the r-th row of $\mathbf{A}$.

Lemma 1. *The following relations hold:*

$$\max\{1, \tfrac{\sigma}{s}\} \leq \sigma' \leq \omega' \leq c, \qquad 1 \leq \sigma \leq \omega \leq d. \tag{9}$$

The proof can be found in the appendix.

4.2 Choice of the Stepsize Parameter β

We analyze Hydra with stepsize parameter $\beta \geq \beta^*$, where

$$\begin{aligned} \beta^* &:= \beta_1^* + \beta_2^*, \\ \beta_1^* &:= 1 + \frac{(\tau-1)(\sigma-1)}{s_1}, \\ \beta_2^* &:= \left(\frac{\tau}{s} - \frac{\tau-1}{s_1}\right)\frac{\sigma'-1}{\sigma'}\sigma, \end{aligned} \tag{10}$$

and $s_1 = \max(1, s-1)$.

As we shall see in Theorem 5, fixing c and τ, the number of iterations needed by Hydra find a solution is proportional to β. Hence, we would wish to use β which is as small as possible, but not smaller than the safe choice $\beta = \beta^*$, for which convergence is guaranteed. In practice, β can often be chosen smaller than the save but conservative value of β^*, leading to larger steps and faster convergence.

If the quantities σ and σ' are hard to compute, then one can replace them by the easily computable upper bounds ω and ω', respectively. However, there are cases when σ can be efficiently approximated and is much smaller than ω. In some ML data sets with $\mathbf{A} \in \{0,1\}^{n\times d}$, σ is close to the average number of nonzeros in a row of $\mathbf{A}$, which can be significantly smaller than the maximum, ω. On the other hand, if σ is difficult to compute, ω may provide a good proxy. Similar remarks apply to σ'.

More importantly, if $\tau \geq 2$ (which covers all interesting uses of Hydra), we may ignore β_2^* altogether, as implied by the following result.

Lemma 2. *If $\tau \geq 2$, then $\beta^* \leq 2\beta_1^*$.*

Proof. It is enough to argue that $\beta_2^* \leq \beta_1^*$. Notice that β_2^* is increasing in σ'. On the other hand, from Lemma 1 we know that $\sigma' \leq c = \frac{d}{s}$. So, it suffices to show that

$$\left(\frac{\tau}{s} - \frac{\tau-1}{s-1}\right)\left(1 - \frac{s}{d}\right)\sigma \leq 1 + \frac{(\tau-1)(\sigma-1)}{s-1}.$$

After straightforward simplification we observe that this inequality is equivalent to $(s-\tau) + (\tau - 2)\sigma + \frac{\sigma}{d}(s+\tau) \geq 0$, which clearly holds. □

Clearly, $\beta^* \geq \beta_1^*$. Hence, if in Hydra we instead of $\beta = \beta^*$ (best/smallest value prescribed by our theory) use $\beta = 2\beta_1^*$ —eliminating the need to compute σ'—the number of iterations will *at most double*. Since σ', present in β_2^*, captures the effect of the initial partition on the iteration complexity of the algorithm, we conclude that this effect is under control.

4.3 Separable Approximation

We first establish a useful identity for the expected value of a random quadratic form obtained by sampling the rows and columns of the underlying matrix via the distributed sampling $\hat{S}$. Note that the result is a direct generalization of Lemma 1 in (Takáč et al., 2013) to the $c > 1$ case.

For $x \in \mathbb{R}^d$ and $\emptyset \neq S \subseteq [d] := \{1, 2, \ldots, d\}$, we write $x^S := \sum_{i \in S} x^i e_i$, where e_i is the i-th unit coordinate vector. That is, x^S is the vector in $\mathbb{R}^d$ whose coordinates $i \in S$ are identical to those of x, but are zero elsewhere.

Lemma 3. *Fix arbitrary $\mathbf{G} \in \mathbb{R}^{d \times d}$ and $x \in \mathbb{R}^d$ and let $s_1 = \max(1, s-1)$. Then*

$$\mathbf{E}[(x^{\hat{S}})^T \mathbf{G} x^{\hat{S}}] = \frac{\tau}{s}\left[\alpha_1 x^T D^{\mathbf{G}} x + \alpha_2 x^T \mathbf{G} x + \alpha_3 x^T (\mathbf{G} - B^{\mathbf{G}}) x\right], \tag{11}$$

where $\alpha_1 = 1 - \frac{\tau-1}{s_1}$, $\alpha_2 = \frac{\tau-1}{s_1}$, $\alpha_3 = \frac{\tau}{s} - \frac{\tau-1}{s_1}$.

Proof. In the $s = 1$ case the statement is trivially true. Indeed, we must have $\tau = 1$ and thus $\mathbf{Prob}(\hat{S} = \{1, 2, \ldots, d\}) = 1$, $h^{\hat{S}} = h$, and hence

$$\mathbf{E}\left[(h^{\hat{S}})^T \mathbf{Q} h^{\hat{S}}\right] = h^T \mathbf{Q} h.$$

This finishes the proof since $\frac{\tau-1}{s_1} = 0$.

Consider now the $s > 1$ case. From Lemma 3 in Richtárik and Takáč (2015) we get

$$\mathbf{E}\left[(h^{\hat{S}})^T \mathbf{Q} h^{\hat{S}}\right] = \mathbf{E}\left[\sum_{i \in \hat{S}} \sum_{j \in \hat{S}} \mathbf{Q}_{ij} h^i h^j\right] = \sum_{i=1}^{d} \sum_{j=1}^{d} p_{ij} \mathbf{Q}_{ij} h^i h^j, \tag{12}$$

where $p_{ij} = \mathbf{Prob}(i \in \hat{S} \ \& \ j \in \hat{S})$. One can easily verify that

$$p_{ij} = \begin{cases} \frac{\tau}{s}, & \text{if } i = j, \\ \frac{\tau(\tau-1)}{s(s-1)}, & \text{if } i \neq j \text{ and } i \in \mathcal{P}_l,\ j \in \mathcal{P}_l \text{ for some } l, \\ \frac{\tau^2}{s^2}, & \text{if } i \neq j \text{ and } i \in \mathcal{P}_k,\ j \in \mathcal{P}_l \text{ for } k \neq l. \end{cases}$$

In particular, the first case follows from Eq (32) and the second from Eq (37) in Richtárik and Takáč (2015). It only remains to substitute p_{ij} into (12) and transform the result into the desired form. □

We now use the above lemma to compute a separable quadratic upper bound on $\mathbf{E}[(h^{\hat{S}})^T \mathbf{M} h^{\hat{S}}]$.

Lemma 4. *For all $h \in \mathbb{R}^d$,*

$$\mathbf{E}\left[(h^{\hat{S}})^T \mathbf{M} h^{\hat{S}}\right] \leq \frac{\tau}{s} \beta^* \left(h^T D^{\mathbf{M}} h\right). \tag{13}$$

Proof. For $x := (D^{\mathbf{M}})^{1/2} h$, we have $(h^{\hat{S}})^T \mathbf{M} h^{\hat{S}} = (x^{\hat{S}})^T \mathbf{Q} x^{\hat{S}}$. Taking expectations on both sides, and applying Lemma 3, we see that $\mathbf{E}[(h^{\hat{S}})^T \mathbf{M} h^{\hat{S}}]$ is equal to (11) for $\mathbf{G} = \mathbf{Q}$. It remains to bound the three quadratics in (11). Since $D^{\mathbf{Q}}$ is the identity matrix, $x^T D^{\mathbf{Q}} x = h^T D^{\mathbf{M}} h$. In view of (7), the 2nd term is bounded as $x^T \mathbf{Q} x \leq \sigma x^T x = \sigma h^T D^{\mathbf{M}} h$. Finally,

$$\begin{aligned} x^T(\mathbf{Q} - B^{\mathbf{Q}})x &= \frac{\sigma'-1}{\sigma'} x^T(\mathbf{Q} - B^{\mathbf{Q}})x + \frac{1}{\sigma'} x^T(\mathbf{Q} - B^{\mathbf{Q}})x \\ &\overset{(8)}{\leq} \frac{\sigma'-1}{\sigma'} x^T(\mathbf{Q} - B^{\mathbf{Q}})x + \frac{\sigma'-1}{\sigma'} x^T B^{\mathbf{Q}} x \\ &= \frac{\sigma'-1}{\sigma'} x^T \mathbf{Q} x \\ &\overset{(7)}{\leq} \frac{\sigma'-1}{\sigma'} \sigma x^T x \\ &= \frac{\sigma'-1}{\sigma'} \sigma h^T D^{\mathbf{M}} h. \end{aligned}$$

It only remains to plug in these three bounds into (11). □

Inequalities of type (13) were first proposed and studied by Richtárik and Takáč (2015)—therein called Expected Separable Overapproximation (ESO)—and were shown to be important for the convergence of parallel coordinate descent methods. However, they studied a different class of loss functions f (convex smooth and partially separable) and different types of random samplings $\hat{S}$, which did not allow them to propose an efficient distributed sampling protocol leading to a distributed algorithm. An ESO inequality was recently used by Takáč et al. (2013) to design a mini-batch stochastic dual coordinate ascent method (parallelizing the original SDCA methods of Hsieh et al. (2008)) and mini-batch stochastic subgradient descent method (Pegasos of Shalev-Shwartz et al. (2011)), and give bounds on how mini-batching leads to acceleration. While it was long observed that mini-batching often accelerates Pegasos in practice, it was only shown with the help of an ESO inequality that this is so also in theory. Recently, Fercoq and Richtárik (2013) have derived ESO inequalities for smooth approximations of nonsmooth loss functions and hence showed that parallel coordinate descent methods can accelerate on their serial counterparts on a class of structured nonsmooth convex losses. As a special case, they obtain a parallel randomized coordinate descent method for minimizing the logarithm of the exponential loss. Again, the class of losses considered in that paper, and the samplings $\hat{S}$, are different from ours. None of the above methods are distributed.

4.4 Fast Rates for Distributed Learning with Hydra

Let x_0 be the starting point of Algorithm 1, x_* be an optimal solution of problem (1) and let $L^* = L(x_*)$. Further, define $\|x\|_{\mathbf{M}}^2 := \sum_{i=1}^d \mathbf{M}_{ii}(x^i)^2$ (a weighted Euclidean norm on $\mathbb{R}^d$) and assume

that f and R are strongly convex with respect to this norm with convexity parameters μ_f and μ_R, respectively. A function ϕ is strongly convex with parameter $\mu_\phi > 0$ if for all $x, h \in \mathbb{R}^d$,

$$\phi(x+h) \geq \phi(x) + (\phi'(x))^T h + \frac{\mu_\phi}{2}\|h\|^2_{\mathbf{M}},$$

where $\phi'(x)$ is a subgradient (or gradient) for ϕ at x.

We now show that Hydra decreases strongly convex L with an exponential rate in ϵ.

Theorem 5. *Assume L is strongly convex with respect to the norm $\|\cdot\|_{\mathbf{M}}$, with $\mu_f + \mu_R > 0$. Choose $x_0 \in \mathbb{R}^d$, $0 < \rho < 1$, $0 < \epsilon < L(x_0) - L^*$ and*

$$T \quad \geq \quad \frac{d}{c\tau} \times \frac{\beta + \mu_R}{\mu_f + \mu_R} \times \log\left(\frac{L(x_0) - L^*}{\epsilon\rho}\right), \tag{14}$$

where $\beta \geq \beta^*$ and β^* is given by (10)*. If $\{x_k\}$ are the random points generated by Hydra (Algorithm 1), then*

$$\mathbf{Prob}(L(x_T) - L^* \leq \epsilon) \geq 1 - \rho.$$

Proof. We first claim that for all $x, h \in \mathbb{R}^d$,

$$\mathbf{E}\left[f(x + h^{\hat{S}})\right] \leq f(x) + \frac{\mathbf{E}[|\hat{S}|]}{d}\left((f'(x))^T h + \frac{\beta}{2} h^T D^{\mathbf{M}} h\right). \tag{15}$$

To see this, substitute $h \leftarrow h^{\hat{S}}$ into (2), take expectations on both sides and then use Lemma 4 together with the fact that for any vector a, $\mathbf{E}[a^T h^{\hat{S}}] = \frac{\mathbf{E}[|\hat{S}|]}{d} = \frac{\tau c}{sc} = \frac{\tau}{s}$. The rest follows by following the steps in the proof in (Richtárik and Takáč, 2015, Theorem 20). □

A similar result, albeit with the weaker rate $O(\frac{s\beta}{\tau\epsilon})$, can be established in the case when neither f nor R are strongly convex. In big data setting, where parallelism and distribution is unavoidable, it is much more relevant to study the dependence of the rate on parameters such as τ and c. We shall do so in the next section.

5. Discussion

In this section we comment on several aspects of the rate captured in (14) and compare Hydra to selected methods.

5.1 Insights Into the Convergence Rate

Here we comment in detail on the influence of the various design parameters (c = # computers, s = # coordinates owned by each computer, and τ = # coordinates updated by each computer in each iteration), instance-dependent parameters ($\sigma, \omega, \mu_R, \mu_f$), and parameters depending both on the instance and design (σ', ω'), on the stepsize parameter β, and through it, on the convergence rate described in Theorem 5.

special case	β^*	$\beta^*/(c\tau)$
any c $\tau = 1$	$1 + \frac{\sigma}{s}\left(\frac{\sigma' - 1}{\sigma'}\right)$	$s + \sigma\left(\frac{\sigma' - 1}{\sigma'}\right)$
$c = 1$ any τ	$1 + \frac{(\tau - 1)(\sigma - 1)}{d - 1}$	$\frac{d}{\tau}\left(1 + \frac{(\tau - 1)(\sigma - 1)}{d - 1}\right)$
$\tau c = d$	σ	σ

Table 2: Stepsize parameter $\beta = \beta^*$ and the leading factor in the rate (14) (assuming $\mu_R = 0$) for several special cases of Hydra.

5.2 Strong Convexity

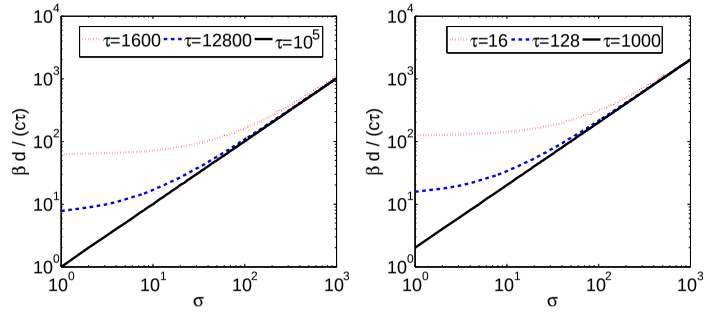
Notice that the size of $\mu_R > 0$ mitigates the effect of a possibly large β on the bound (14). Indeed, for large μ_R, the factor $(\beta + \mu_R)/(\mu_f + \mu_R)$ approaches 1, and the bound (14) is dominated by the term $\frac{d}{c\tau}$, which means that Hydra enjoys linear speedup in c and τ. In the following comments we will assume that $\mu_R = 0$, and focus on studying the dependence of the leading term $d\frac{\beta}{c\tau}$ on various quantities, including τ, c, σ and σ'.

5.3 Search for Small but Safe β

As shown by Takáč et al. (2013, Section 4.1), mini-batch SDCA might *diverge* in the setting with $\mu_f = 0$ and $R(x) \equiv 0$, even for a simple quadratic function with $d = 2$, provided that $\beta = 1$. Hence, small values of β need to be avoided. However, in view of Theorem 5, it is good if β is as small as possible. So, there is a need for a "safe" formula for a small β. Our formula (10), $\beta = \beta^*$, is serving that purpose. For a detailed introduction into the issues related to selecting a good β for parallel coordinate descent methods, we refer the reader to the first 5 pages of (Fercoq and Richtárik, 2013).

5.4 The Effect of σ'

If $c = 1$, then by Lemma 9, $\sigma' = c = 1$, and hence $\beta^*_2 = 0$. However, for $c > 1$ we *may* have $\beta^*_2 > 0$, which can hence be seen as a price we need to pay for using more nodes. The price depends on the way the data is partitioned to the nodes, as captured by σ'. In favorable circumstances, $\sigma' \approx 1$ even if $c > 1$, leading to $\beta^*_2 \approx 0$. However, in general we have the bound $\sigma' \geq \frac{c\sigma}{d}$, which gets worse as c increases and, in fact, σ' can be as large as c. Note also that ξ is decreasing in τ, and that $\xi(s, s) = 0$. This means that by choosing $\tau = s$ (which effectively removes randomization from Hydra), the effect of β^*_2 is eliminated. This may not be always possible as often one needs to solve problems with s vastly larger than the number of updates that can be performed on any given node in parallel. If $\tau \ll s$, the effect of β^*_2 can be controlled, to a certain extent, by choosing a partition



(a) $(c, s) = (1, 10^5)$ (b) $(c, s) = (10^2, 10^3)$

Figure 1: In terms of the number of iterations, very little is lost by using $c > 1$ as opposed to $c = 1$.

with small σ'. Due to the way σ' is defined, this may not be an easy task. However, it may be easier to find partitions that minimize ω', which is often a good proxy for σ'. Alternatively, we may ignore estimating σ' altogether by setting $\beta = 2\beta_1^*$, as mentioned before, at the price of at most doubling the number of iterations.

5.5 Speedup by Increasing τ

Let us fix c and compare the quantities $\gamma_\tau := \frac{\beta^*}{c\tau}$ for $\tau = 1$ and $\tau = s$. We now show that $\gamma_1 \geq \gamma_s$, which means that if all coordinates are updated at every node, as opposed to one only, then Hydra run with $\beta = \beta^*$ will take fewer iterations. Comparing the 1st and 3rd row of Table 2, we see that $\gamma_1 = s + \sigma\frac{\sigma'-1}{\sigma'}$ and $\gamma_s = \sigma$. By Lemma 1, $\gamma_1 - \gamma_s = s - \frac{\sigma}{\sigma'} \geq 0$.

5.6 Price of Distribution

For illustration purposes, consider a problem with $d = 10^5$ coordinates. In Figure 1(a) we depict the size of $\frac{d\beta_1^*}{c\tau}$ for $c = 1$ and several choices of τ, as a function of σ. We see that Hydra works better for small values of σ and that with increasing σ, the benefit of using updating more coordinates diminishes. In Figure 1(a) we consider the same scenario, but with $c = 100$ and $s = 1000$, and we plot $\frac{d2\beta_1^*}{c\tau}$ on the y axis. Note that the red dotted line in both plots corresponds to a parallel update of 1600 coordinates. In (a) all are updated on a single node, whereas in (b) we have 100 nodes, each updating 16 coordinates at a time. Likewise, the dashed blue dashed and solid black lines are also comparable in both plots. Note that the setup with $c = 10$ has a slightly weaker performance, the lines are a bit lower. This is the price we pay for using c nodes as opposed to a single node (obviously, we are ignoring communication cost here). However, in big data situations one simply has no other choice but to utilize more nodes.

5.7 Comparison with Other Methods

While we are not aware of any other *distributed* coordinate descent method, Hydra in the $c = 1$ case is closely related to several existing parallel coordinate descent methods.

5.7.1 HYDRA VS SHOTGUN

The Shotgun algorithm (parallel coordinate descent) of Bradley et al. (2011) is similar to Hydra for $c = 1$. Some of the differences: Bradley et al. (2011) only consider R equal to the $L1$ norm and their method works in dimension $2d$ instead of the native dimension d. Shotgun was not analyzed for strongly convex f, and convergence in expectation was established. Moreover, Bradley et al. (2011) analyze the step-size choice $\beta = 1$, fixed independently of the number of parallel updates τ, and give results that hold only in a "small τ" regime. In contrast, our analysis works for any choice of τ.

5.7.2 HYDRA VS PCDM

For $c = 1$, Hydra reduces to the parallel coordinate descent method (PCDM) of Richtárik and Takáč (2015), but with a *better* stepsize parameter β. We were able to achieve smaller β (and hence better rates) because we analyze a different and more specialized class of loss functions (those satisfying (2)). In comparison, Richtárik and Takáč (2015) look at a general class of partially separable losses. Indeed, in the $c = 1$ case, our distributed sampling $\hat{S}$ reduces to the sampling considered in (Richtárik and Takáč, 2015) (τ-nice sampling). Moreover, our formula for β (see Table 2) is essentially identical to the formula for β provided in (Richtárik and Takáč, 2015, Theorem 14), with the exception that we have σ where they have ω. By (9), we have $\sigma \leq \omega$, and hence our β is smaller.

5.7.3 HYDRA VS SPCDM

SPCDM of Fercoq and Richtárik (2013) is PCDM applied to a smooth approximation of a nonsmooth convex loss; with a special choice of β, similar to β_1. As such, it extends the reach of PCDM to a large class of nonsmooth losses, obtaining $O(\frac{1}{\epsilon^2})$ rates. It is possible to develop accelerated Hydra with $O(\frac{1}{\epsilon})$ rates by combining ideas from this paper, Fercoq and Richtárik (2013) with the APPROX method of Fercoq and Richtárik (2015).

5.7.4 HYDRA VS MINI-BATCH SDCA

Takáč et al. (2013) studied the performance of a mini-batch stochastic dual coordinate ascent for SVM dual ("mini-batch SDCA"). This is a special case of our setup with $c = 1$, convex quadratic f and $R_i(t) = 0$ for $t \in [0, 1]$ and $R_i(t) = +\infty$ otherwise. Our results can thus be seen as a generalization of the results in that paper to a larger class of loss functions f, more general regularizers R, and most importantly, to the distributed setting ($c > 1$). Also, we give $O(\log \frac{1}{\epsilon})$ bounds under strong convexity, whereas (Takáč et al., 2013) give $O(\frac{1}{\epsilon})$ results without assuming strong convexity. However, Takáč et al. (2013) perform a primal-dual analysis, whereas we do not.

6. Distributed Computation of the Gradient

In this section we describe some important elements of our distributed implementation.

loss function ℓ	$f_i'(x)$	$\mathbf{M}_{ii}$
SL	$\sum_{j=1}^{m} -\mathbf{A}_{ji}(y^j - \mathbf{A}_{j:}x)$	$\|\mathbf{A}_{:i}\|_2^2$
LL	$\sum_{j=1}^{m} -y^j\mathbf{A}_{ji}\frac{\exp(-y^j\mathbf{A}_{j:}x)}{1+\exp(-y^j\mathbf{A}_{j:}x)}$	$\frac{1}{4}\|\mathbf{A}_{:i}\|_2^2$
HL	$\sum_{j\,:\,y^j\mathbf{A}_{j:}x<1} \left(-y^j\mathbf{A}_{ji}(1-y^j\mathbf{A}_{j:}x)\right)$	$\|\mathbf{A}_{:i}\|_2^2$

Table 3: Information needed in Step 5 of Hydra for f given by (3) in the case of the three losses ℓ from Table 1.

Note that in Hydra, x_k is stored in a distributed way. That is, the values x_k^i for $i \in \mathcal{P}_l$ are stored on computer l. Moreover, Hydra partitions $\mathbf{A}$ columnwise as $\mathbf{A} = [\mathbf{A}_1, \ldots, \mathbf{A}_c]$, where $\mathbf{A}_l$ consists of columns $i \in \mathcal{P}_l$ of $\mathbf{A}$, and stores $\mathbf{A}_l$ on computer l. So, $\mathbf{A}$ is chopped into smaller pieces with stored in a distributed way in fast memory (if possible) across the c nodes. Note that this allows the method to work with large matrices.

At Step 5 of Hydra, node l at iteration $k+1$ needs to know the partial derivatives $f_i'(x_{k+1})$ for $i \in \hat{S}_l \subseteq \mathcal{P}_l$. We now describe several efficient distributed protocols for the computation of $f_i'(x_{k+1})$ for functions f of the form (3), in the case of the three losses ℓ given in Table 1 (SL, LL, HL). The formulas for $f_i'(x)$ are summarized in Table 3 ($\mathbf{A}_{j:}$ refers to the j-th row of $\mathbf{A}$). Let $D^y := \mathrm{Diag}(y)$.

6.1 Basic Protocol

If we write $h_k^i = 0$ if i is not updated in iteration k, then

$$x_{k+1} = x_k + \sum_{l=1}^{c}\sum_{i\in\hat{S}_l} h_k^i e_i. \tag{16}$$

Now, if we let

$$g_k := \begin{cases} \mathbf{A}x_k - y, & \text{for SL,} \\ -D^y\mathbf{A}x_k, & \text{for LL and HL,} \end{cases} \tag{17}$$

then by combining (16) and (17), we get

$$g_{k+1} = g_k + \sum_{l=1}^{c} \delta g_{k,l}, \qquad \text{where}$$

$$\delta g_{k,l} = \begin{cases} \sum_{i\in\hat{S}_l} h_k^i \mathbf{A}_{:i}, & \text{for SL,} \\ \sum_{i\in\hat{S}_l} -h_k^i D^y \mathbf{A}_{:i}, & \text{for LL and HL.} \end{cases}$$

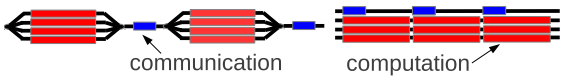


Figure 2: Parallel-serial (PS; left) vs Fully Parallel (FP; right) approach.

Note that the value $\delta g_{k,l}$ can be computed on node l as all the required data is stored locally. Hence, we let each node compute $\delta g_{k,l}$, and then use a *reduce all* operation to add up the updates to obtain g_{k+1}, and pass the sum to all nodes. Knowing g_{k+1}, node l is then able to compute $f_i'(x_{k+1})$ for any $i \in \mathcal{P}_l$ as follows:

$$f_i'(x_{k+1}) = \begin{cases} \mathbf{A}_{:i}^T g_{k+1} = \sum_{j=1}^n \mathbf{A}_{ji} g_{k+1}^j, & \text{for SL,} \\ \sum_{j=1}^n y^j \mathbf{A}_{ji} \frac{\exp(g_{k+1}^j)}{1+\exp(g_{k+1}^j)}, & \text{for LL,} \\ \sum_{j\,:\,g_{k+1}^j>-1} y^j \mathbf{A}_{ji}(1+g_{k+1}^j), & \text{for HL.} \end{cases}$$

6.2 Advanced Protocols

The basic protocol discussed above has obvious drawbacks. Here we identify them and propose modifications leading to better performance.

- *alternating Parallel and Serial regions (PS):* The basic protocol alternates between two procedures: i) a computationally heavy one (done in parallel) with no MPI communication, and ii) MPI communication (serial). An easy fix would be to dedicate 1 thread to deal with communication and the remaining threads within the same computer for computation. We call this protocol *Fully Parallel (FP)*. Figure 2 compares the basic (left) and FP (right) approaches.

- *Reduce All (RA):* In general, reduce all operations may significantly degrade the performance of distributed algorithms. Communication taking place only between nodes close to each other in the network, e.g., nodes directly connected by a cable, is more efficient. Here we propose the *Asynchronous StreamLined (ASL)* communication protocol in which each node, in a given iteration, sends only 1 message (asynchronously) to a nearby computer, and also receives only one message (asynchronously) from another nearby computer. Communication hence takes place in an *Asynchronous Ring*. This communication protocol requires significant changes in the algorithm. Figure 3 illustrates the flow of messages at the end of the k-th iteration for $c = 4$.

  We order the nodes into a ring, denoting l_- and l_+ the two nodes neighboring node l. Node l only receives data from l_-, and sends data to l_+. Let us denote by $\delta G_{k,l}$ the data sent by node l to l_+ at the end of iteration k. When l starts iteration k, it already knows $\delta G_{k-1,l_-}$.[2] Hence, data which will be sent at the end of the k-th iteration by node l is given by

$$\delta G_{k,l} = \delta G_{k-1,l_-} - \delta g_{k-c,l} + \delta g_{k,l}. \tag{18}$$

  This leads to the update rule

$$g_{k+1,l} = g_{k,l} + \delta g_{k,l} + \delta G_{k,l_-} - \delta g_{k-c+1,l}.$$

2. Initially, we let $\delta g_{k,l} = \delta G_{k,l} = 0$ for all $k \le 0$.

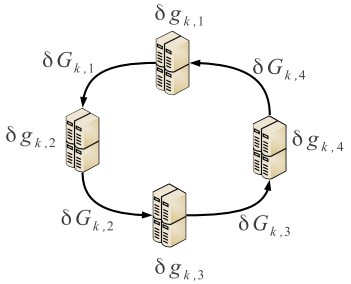


Figure 3: ASL protocol with $c = 4$ nodes. In iteration k, node l computes $\delta g_{k,l}$, and sends $\delta G_{k,l}$ to l_+.

ASL needs less communication per iteration. On the other hand, information is propagated more slowly to the nodes through the ring, which may adversely affect the number of iterations till convergence (note that we do not analyze Hydra with this communication protocol). Indeed, it takes $c-1$ iterations to propagate information to all nodes. Also, storage requirements have increased: at iteration k we need to store the vectors $\delta g_{t,l}$ for $k-c \leq t \leq k$ on computer l.

7. Experiments

In this section we present numerical evidence that Hydra is capable to efficiently solve big data problems. We have a C++ implementation, using Boost::MPI and OpenMP. Experiments were executed on a Cray XE6 cluster with 128 nodes; with each node equipped with two AMD Opteron Interlagos 16-core processors and 32 GB of RAM.

7.1 Performance of Communication Protocols

In this experiment we consider a LASSO problem, i.e., f given by (3) with ℓ being the square loss (SL) and $R(x) = \|x\|_1$. In order to to test Hydra under controlled conditions, we adapted the LASSO generator proposed by Nesterov (2013, Section 6); modifications were necessary as the generator does not work well in the big data setting.

As discussed in Section 6, the advantage of the RA protocol is the fact that Theorem 5 was proved in this setting, and hence can be used as a safe benchmark for comparison with the advanced protocols.

Table 4 compares the average time per iteration for the 3 approaches and 3 choices of τ. We used 128 nodes, each running 4 MPI processes (hence $c = 512$). Each MPI process runs 8 OpenMP threads, giving 4,096 cores in total. The data matrix $\mathbf{A}$ has $n = 10^9$ rows and $d = 5 \times 10^8$ columns, and has 3 TB, double precision. One can observe that in all cases, ASL-FP yields largest gains compared to the benchmark RA-PS protocol. Note that ASL has some overhead in each iteration, and hence in cases when computation per node is small ($\tau = 10$), the speedup is only 1.62. When $\tau = 10^2$ (in this case the durations of computation and communication were comparable), ASL-FP

τ	comm. protocol	organization	avg. time	speedup
10	RA	PS	0.040	—
10	RA	FP	0.035	1.15
10	ASL	FP	0.025	1.62
10^2	RA	PS	0.100	—
10^2	RA	FP	0.077	1.30
10^2	ASL	FP	0.032	3.11
10^3	RA	PS	0.321	—
10^3	RA	FP	0.263	1.22
10^3	ASL	FP	0.249	1.29

Table 4: Duration of a single Hydra iteration for 3 communication protocols. The basic RA-PS protocol is always the slowest, but follows the theoretical analysis. ASL-FP can be $3\times$ faster.

is 3.11 times faster than RA-PS. But the gain becomes again only moderate for $\tau = 10^3$; this is because computation now takes much longer than communication, and hence the choice of strategy for updating the auxiliary vector g_k is less significant. Let us remark that the use of larger τ requires larger $beta$, and hence possibly more iterations (in the worst case).

We now move on to solving an artificial big data LASSO problem with matrix $\mathbf{A}$ in block angular form, depicted in (19).

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1^{loc} & 0 & \cdots & 0 \\ 0 & \mathbf{A}_2^{loc} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_1^{glob} & \mathbf{A}_2^{glob} & \cdots & \mathbf{A}_c^{glob} \end{pmatrix}. \tag{19}$$

Such matrices often arise in stochastic optimization. Each Hydra head (=node) l owns two matrices: $\mathbf{A}_l^{loc} \in \mathbb{R}^{1,952,148 \times 976,562}$ and $\mathbf{A}_l^{glob} \in \mathbb{R}^{500,224 \times 976,562}$. The average number of nonzero elements per row in the local part of $\mathbf{A}_l$ is 175, and $1,000$ for the global part. Optimal solution x_* has exactly $160,000$ nonzero elements. Figure 4 compares the evolution of $L(x_k) - L^*$ for ASL-FP and RA-FP.

Remark: When communicating g_{kl}, only entries corresponding to the global part of $\mathbf{A}_l$ need to be communicated, and hence in RA, a *reduce all* operation is applied to vectors $\delta g_{glob,l} \in \mathbb{R}^{500,224}$. In ASL, vectors with the same length are sent.

7.2 Updating All Coordinates on Each Node in Each Iteration Might not be Optimal

In this section we give an experimental demonstration that it may not be optimal for each node of Hydra to update all coordinates it owns (in each iteration). That is, we will show that the seemingly ideal choice $\tau = s$ is not necessarily optimal.

In the experiment, we use the *astro-ph* data set consisting of abstracts of physics papers (see Shalev-Shwartz et al. (2011)). This data set consists of $d = 29,882$ samples with $n = 32,487$ fea-

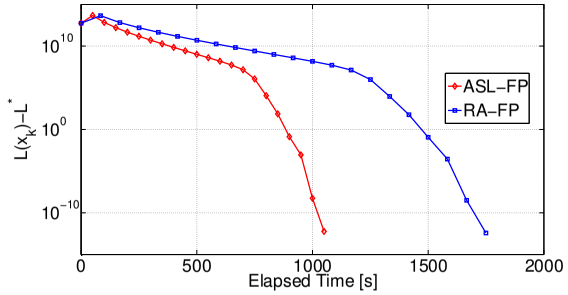


Figure 4: Evolution of $L(x_k) - L^*$ in time. ASL-FP significantly outperforms RA-FP. The loss L is pushed down by 25 degrees of magnitude in less than 30 minutes (3TB problem).

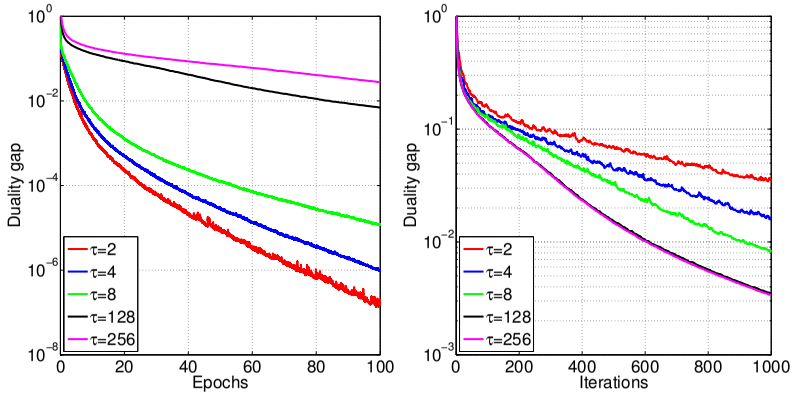


Figure 5: Evolution of the duality gap for several choices of τ.

tures; and hence is a relatively small data set. We have chosen a random partition of the coordinates (=samples) to $c = 32$ nodes, with each partition consisting of $s = 933$ samples (the last partition consists of 959 samples).

In Figure 5 we depict the evolution of the duality gap as a function of effective passes over data (epochs) (left plot) and as a function of iterations (right right). In each plot, we depict the performance of Hydra run with various choices of τ.

With larger τ, i.e., when each node updates more coordinates in a single iteration, more epochs are needed to obtain a solution of any given accuracy. However, if increasing τ by a fair amount only leads to a small increase in computation time within a node (say, because, each node utilizes a multicore processor and τ is proportional to the number of processors), then the increased number of passes over data will not be matched with a similar increase in compute time. The reverse side

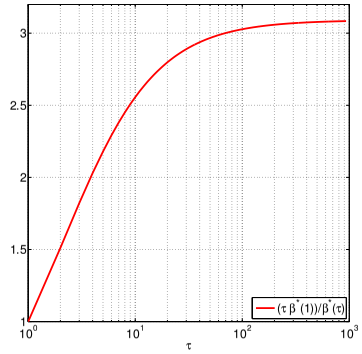


Figure 6: Evolution of the duality gap for several choices of τ.

of this situation is depicted in the right plot. That is, as τ increases, one needs fewer iterations to obtain any given accuracy.

In Figure 6 we depict the theoretical speed-up guarantee, i.e., the fraction $\frac{\tau}{\beta^*}$. Looking at the right plot of Figure 5, we see that the choice $\tau = 256$ does not lead to further speedup when compared to the $\tau = 128$ choice. This phenomenon is also captured in Figure 6, where the line becomes quite flat above $\tau \approx 10^2$.

.

Remark: We have used 10 iterations of the power method in order to estimate parameters σ and σ' which were used to obtain β^* using (10) (note that, in view of Lemma 2, we could have also used $\beta = 2\beta_1^*$). Their values are: $\sigma = 440.61$ and $\sigma' = 29.65$.

7.3 A Simple Model for Communication-Computation Trade-Off: Search for Optimal τ

Assume all c nodes of the cluster are identical. Also assume $s \geq 2$. Further, assume there is a constant $1 \leq K \leq s$ such that while each node is able to compute K coordinate updates in time T_1, the computation of jK updates, for $j = 2, 3, \ldots$ takes jT_1 units of time. This will be approximately true in reality for K sufficiently large, typically a small multiple of the number of cores. It may be useful to think that K is, in fact, equal to the number of cores of each node. Further, assume that the time of updating g_k to g_{k+1} is T_2 (note that this involves communication).

Bases on this simple model, a single iteration of Hydra run with $\tau = jK$ coordinate updates on each node (for $1 \leq j \leq s/K$) takes

$$jT_1 + T_2 \tag{20}$$

units of time.

In what follows, we will assume, for simplicity, that $\mu_R = 0$ (i.e., the regularizer is not strongly convex). The computations simplify with this assumption, but similar computations can be performed in the $\mu_R > 0$ case.

Focusing on the terms which depend on τ, Theorem 5 says that to obtain an ϵ-solution, Hydra needs $\mathcal{O}(\frac{\beta}{\tau})$ iterations. Combining this with the time it takes to perform a single iteration, the overall

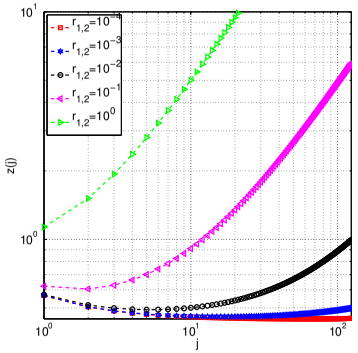


Figure 7: The dependence of the total "time complexity" $T(j)$ on j for various values of the computation-communication ratio $r_{1,2}$.

time, as a function of j, is proportional to

$$\begin{aligned} T(j) &:= \frac{\beta}{jK}(jT_1 + T_2) \\ &\overset{(10)}{=} \frac{1 + \frac{(jK-1)(\sigma-1)}{s_1} + \left(\frac{jK}{s} - \frac{jK-1}{s_1}\right)\frac{\sigma'-1}{\sigma'}\sigma}{jK}(\mathcal{T}_2 + \mathcal{T}_1 j) \\ &= \frac{1 + \frac{(jK-1)(\sigma-1)}{s_1} + \left(\frac{jK}{s} - \frac{jK-1}{s_1}\right)\frac{\sigma'-1}{\sigma'}\sigma}{jK}(1 + jr_{1,2}), \end{aligned}$$

where

$$r_{1,2} := \frac{T_1}{T_2}.$$

Note that $r_{1,2}$ is high if computation is expensive relative to communication, and vice versa. This ratio can be estimated in any particular distributed system.

In Figure 7 we plot $T(j)$ as a function of j for $r_{1,2} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$, $K = 8$ and the rest of the parameters appearing in the definition of $T(j)$ identical to those used in Section 7.2. For large enough $r_{1,2}$, i.e., if communication is relatively cheap, then T is an increasing function of j. This means that it is optimal to choose $j = 1$, which means that it is optimal to only update $\tau = jK$ coordinates on each node in each iteration. This observation is informally summarized as follows:

> *If communication is inexpensive relative to computation, it is optimal for each node to perform a small number of coordinate updates in each iteration. That is, if communication is cheap, do less computation in each iteration.*

For small enough $r_{1,2}$, however, the optimal point j is strictly larger than 1 and smaller than s/K. As $r_{1,2}$ decreases, optimal j increases. This observation is informally summarized as:

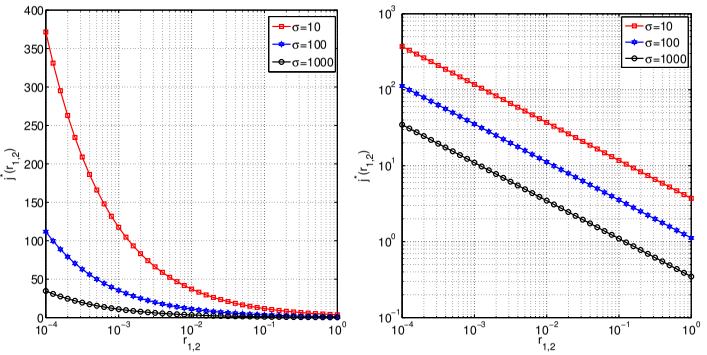


Figure 8: Comparison of $j^*(r_{1,2})$ for various values of parameter σ (left: semilog plot; right: loglog plot).

> *As communication becomes more expensive relative to computation, it is optimal for each node to perform a larger number of coordinate updates in each iteration. That is, if communication is more expensive, do more computation in each iteration.*

Since T is a simple function of j, it is possible to find j^* which minimizes $T(j)$ in closed form (if we relax the requirement of keeping j integral, which is not very important):

$$j^* := \arg\min_j T(j) = \sqrt{\frac{s(s\sigma' - \sigma)}{r_{1,2}K(s\sigma'(\sigma-1) + \sigma - \sigma'\sigma)}}.$$

In Figure 8 we plot j^* as a function of $r_{1,2}$ for $\sigma = 10, 100, 1000$. Recall that smaller σ means less correlated data. This means that less synchronization is needed, and eventually allows us to do more coordinate updates in a single iteration (i.e., j^* is larger). If communication is relatively expensive ($r_{1,2}$ is small), then j^* is big. However, as communication gets cheaper, j^* gets smaller.

7.4 Experiments with a Real Data Set in a Real Distributed Environment

In this section we report on experiments with the WebSpam data set Libsvm. This data set encodes a binary classification problem with $d = 350,000$ samples and $n = 16,609,143$ features. The total size of the data set describing the problem exceeds 23 GB. We split this data into $c = 16$ balanced groups, each containing $s = 21,875$ samples and applied Hydra to the problem (the dual of SVM with hinge loss) for various choices of τ. The results are depicted in Figure 9. As each node in our experiments had an 8-core processor, we have chosen τ in multiples of 8: $\tau \in \{8, 80, 160, 320, 640, 1280, 2560, 5120\}$.

In the plot on the left we have run the RA variant with $\beta := 2\beta_1^*$ and in the plot on the right we have run the RA with $\beta := \frac{2\beta_1^*}{100}$. By comparing the plots, we can observe that for this particular data set, the theoretically safe choice of β ($\beta := 2\beta_1^*$) is too conservative. Indeed, massive speedups

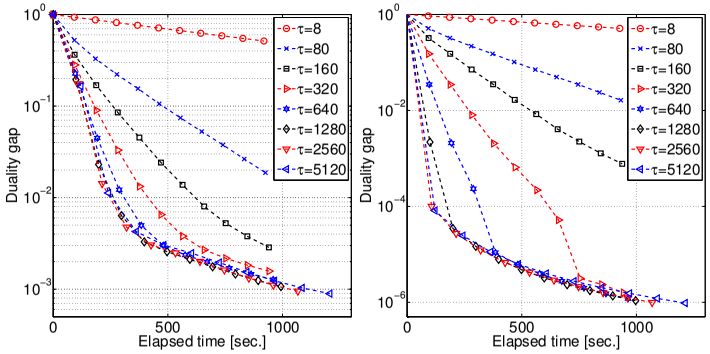


Figure 9: The duality gap evolving in time for Hydra run with various levels of parallelism (as given by τ) within each node.

can be obtained by using a more aggressive stepsize strategy, i.e., by choosing β hundred times smaller than the one recommended by theory. While it is to be expected that particular data sets will tolerate such aggressive stepsize strategies, there are data sets where such stepsizes might lead to a diverging algorithm. However, it is clear that Hydra would benefit from a line-search procedure for the selection of β.

One can also observe from Figure 9 that with beyond some point, increasing τ does not bring much benefit, and only increases the runtime.

8. Extensions

Our results can be extended to the setting where coordinates are replaced by blocks of coordinates, as in (Nesterov, 2012), and to partially separable losses, as in (Richtárik and Takáč, 2015). We expect the block setup to potentially lead to further significant speedups in a practical implementation due to the fact that this will allow us to design data-dependent block norms, which will in turn enable Hydra to use more curvature information the information contained in the diagonal of M. In such a setup, in Step 7 we will instead have a problem of a larger dimension, and the quadratic term can involve a submatrix of M.

Acknowledgements

The first author would like to acknowledge support from the EPSRC Grant EP/K02325X/1, "Accelerated Coordinate Descent Methods for Big Data Optimization" and EPSRC Grant EP/I017127/1, "Mathematics for Vast Digital Resources". The second author was supported by the Centre for Numerical Algorithms and Intelligent Software (funded by EPSRC grant EP/G036136/1 and the Scottish Funding Council).

Appendix A. Proof Lemma 1

1. The inequality $\omega' \leq c$ is obviously true. By considering x with zeroes in all coordinates except those that belong to $\mathcal{P}_k$ (where k is an arbitrary but fixed index), we see that $x^T\mathbf{Q}x = x^T B^{\mathbf{Q}} x$, and hence $\sigma' \geq 1$.

2. We now establish that $\sigma' \leq \omega'$. Let $\phi(x) = \frac{1}{2}x^T\mathbf{Q}x$, $x \in \mathbb{R}^d$; its gradient is

$$\phi'(x) = \mathbf{Q}x. \tag{21}$$

For each $k = 1, 2, \ldots, c$, define a pair of conjugate norms on $\mathbb{R}^s$ as follows:

$$\|v\|_{(k)}^2 := \langle \mathbf{Q}^{kk} v, v\rangle, \qquad (\|v\|_{(k)}^*)^2 := \max_{\|v'\|_{(k)} \leq 1} \langle v', v\rangle = \langle (\mathbf{Q}^{kk})^{-1} v, v\rangle. \tag{22}$$

Let $\mathbf{U}_k$ be a column submatrix of the d-by-d identity matrix corresponding to columns $i \in \mathcal{P}_k$. Clearly, $\mathbf{A}_k = \mathbf{A}\mathbf{U}_k$ and $\mathbf{U}_k^T\mathbf{Q}\mathbf{U}_k$ is the k-th diagonal block of $\mathbf{Q}$, i.e.,

$$\mathbf{U}_k^T\mathbf{Q}\mathbf{U}_k \overset{(4)}{=} \mathbf{Q}^{kk}. \tag{23}$$

Moreover, for $x \in \mathbb{R}^d$ and $k \in \{1, 2, \ldots, c\}$, let $x^{(k)} = \mathbf{U}_k^T x$ and, fixing positive scalars $w_1, \ldots, w_c$, define a norm on $\mathbb{R}^d$ as follows:

$$\|x\|_w := \left(\sum_{k=1}^c w_k \|x^{(k)}\|_{(k)}^2\right)^{1/2}. \tag{24}$$

Now, we claim that for each k,

$$\|\mathbf{U}_k^T\phi'(x + \mathbf{U}_k h^{(k)}) - \mathbf{U}_k^T\phi'(x)\|_{(k)}^* \leq \|h^{(k)}\|_{(i)}.$$

This means that ϕ' is block Lipschitz (with blocks corresponding to variables in $\mathcal{P}_k$), with respect to the norm $\|\cdot\|_{(k)}$, with Lipschitz constant 1. Indeed, this is, in fact, satisfied with equality:

$$\begin{aligned}
\|\mathbf{U}_k^T\phi'(x + \mathbf{U}_k h^{(k)}) - \mathbf{U}_k^T\phi'(x)\|_{(k)}^* &\overset{(21)}{=} \|\mathbf{U}_k^T\mathbf{Q}(x + \mathbf{U}_k h^{(k)}) - \mathbf{U}_k\mathbf{Q}x\|_{(k)}^* \\
&= \|\mathbf{U}_k^T\mathbf{Q}\mathbf{U}_k h^{(k)}\|_{(k)}^* \\
&\overset{(23)}{=} \|\mathbf{Q}^{kk} h^{(k)}\|_{(k)}^* \\
&\overset{(22)}{=} \langle (\mathbf{Q}^{kk})^{-1}\mathbf{Q}^{kk} h^{(k)}, \mathbf{Q}^{kk} h^{(k)}\rangle \\
&\overset{(22)}{=} \|h^{(k)}\|_{(k)}.
\end{aligned}$$

This is relevant because then, by Richtárik and Takáč (2015, Theorem 7; see comment 2 following the theorem), it follows that ϕ' is Lipschitz with respect to $\|\cdot\|_w$, where $w_k = 1$ for all $k = 1, \ldots, c$, with Lipschitz constant ω' (ω' is the degree of partial block separability of ϕ with respect to the blocks $\mathcal{P}_k$). Hence,

$$\frac{1}{2}x^T\mathbf{Q}x = \phi(x) \leq \phi(0) + (\phi'(0))^T x + \frac{\omega'}{2}\|x\|_w^2 \overset{(22)+(24)}{=} \frac{\omega'}{2}\sum_{k=1}^c \langle \mathbf{Q}^{kk} x^{(k)}, x^{(k)}\rangle = \frac{\omega'}{2}(x^T B^{\mathbf{Q}} x),$$

which establishes the inequality $\sigma' \leq \omega'$.

3. We now show that $\frac{\sigma}{s} \leq \sigma'$. If we let $\theta := \max\{x^T B^{\mathbf{Q}} x : x^T x \leq 1\}$, then $x^T B^{\mathbf{Q}} x \leq \theta x^T x$ and hence $\{x \ : \ x^T x \leq 1\} \subseteq \{x \ : \ x^T B^{\mathbf{Q}} x \leq \theta\}$. This implies that

$$\sigma = \max_x \{x^T \mathbf{Q} x \ : \ x^T x \leq 1\} \leq \max_x \{x^T \mathbf{Q} x \ : \ x^T B^{\mathbf{Q}} x \leq \theta\} = \theta \sigma'.$$

It now only remains to argue that $\theta \leq s$. For $x \in \mathbb{R}^d$, let $x^{(k)}$ denote its subvector in $\mathbb{R}^s$ corresponding to coordinates $i \in \mathcal{P}_k$ and $\Delta = \{p \in \mathbb{R}^c : p \geq 0, \ \sum_{k=1}^c p_k = 1\}$. We can now write

$$\begin{aligned}
\theta &= \max_x \left\{ \sum_{k=1}^c (x^{(k)})^T \mathbf{Q}^{kk} x^{(k)} \ : \ \sum_{k=1}^c (x^{(k)})^T x^{(k)} \leq 1 \right\} \\
&= \max_{p \in \Delta} \sum_{k=1}^c \left\{ \max (x^{(k)})^T \mathbf{Q}^{kk} x^{(k)} \ : \ (x^{(k)})^T x^{(k)} = p_k \right\} \\
&= \max_{p \in \Delta} \sum_{k=1}^c p_k \max \left\{ (x^{(k)})^T \mathbf{Q}^{kk} x^{(k)} \ : \ (x^{(k)})^T x^{(k)} = 1 \right\} \\
&= \max_{1 \leq k \leq c} \max \left\{ (x^{(k)})^T \mathbf{Q}^{kk} x^{(k)} \ : \ (x^{(k)})^T x^{(k)} = 1 \right\} \\
&\leq s.
\end{aligned}$$

In the last step we have used the fact that $\sigma(\mathbf{Q}) = \sigma \leq c = \dim(\mathbf{Q})$, proved in steps 1 and 2, applied to the setting $\mathbf{Q} \leftarrow \mathbf{Q}^{kk}$.

4. The chain of inequalities $1 \leq \sigma \leq \omega \leq c$ is obtained as a special case of the chain $1 \leq \sigma' \leq \omega' \leq d$ (proved above) when $c = d$ (and hence $\mathcal{P}_l = \{l\}$ for $l = 1, \ldots, d$). Indeed, in this case $B^{\mathbf{Q}} = D^{\mathbf{Q}}$, and so $x^T B^{\mathbf{Q}} x = x^T D^{\mathbf{Q}} x = x^T x$, which means that $\sigma' = \sigma$ and $\omega' = \omega$.

References

J. Bradley, A. Kyrola, D. Bickson, and C. Guestrin. Parallel coordinate descent for l1-regularized loss minimization. In *ICML*, 2011.

O. Fercoq. Parallel coordinate descent for the AdaBoost problem. In *ICMLA*, 2013.

O. Fercoq and P. Richtárik. Smooth minimization of nonsmooth functions with parallel coordinate descent methods. *arXiv:1309.5885*, 2013.

O. Fercoq and P. Richtárik. Accelerated, parallel and proximal coordinate descent. *SIAM Journal on Optimization*, 25(4):1997–2023, 2015.

C-J. Hsieh, K-W. Chang, C-J. Lin, S.S. Keerthi, , and S. Sundarajan. A dual coordinate descent method for large-scale linear SVM. In *ICML*, 2008.

Libsvm. *Datasets*. http://www.csie.ntu.edu.tw/~cjlin/ libsvmtools/datasets/binary.html.

Z. Lu and L. Xiao. On the complexity analysis of randomized block-coordinate descent methods. *Mathematical Programming*, 152(1):615–642, 2015.

I. Mukherjee, Y. Singer, R. Frongillo, and K. Canini. Parallel boosting with momentum. In *ECML*, 2013.

I. Necoara, Yu. Nesterov, and F. Glineur. Efficiency of randomized coordinate descent methods on optimization problems with linearly coupled constraints. Technical report, 2012.

Yu. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Yu. Nesterov. Gradient methods for minimizing composite objective function. *Mathematical Programming*, 140(1):125–161, 2013.

P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, pages 1–52, 2015.

P. Richtárik and M. Takáč. Efficient serial and parallel coordinate descent methods for huge-scale truss topology design. In *Operations Research Proceedings*, pages 27–32. Springer, 2012.

P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(2):1–38, 2014.

S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1 regularized loss minimization. In *ICML*, 2009.

S. Shalev-Shwartz and T. Zhang. Accelerated mini-batch stochastic dual coordinate ascent. In *NIPS*, pages 378–385, 2013a.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14:567–599, 2013b.

S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30, 2011.

M. Takáč, A. Bijral, P. Richtárik, and N. Srebro. Mini-batch primal and dual methods for SVMs. In *ICML*, 2013.

R. Tappenden, P. Richtárik, and J. Gondzio. Inexact coordinate descent: complexity and preconditioning. *arXiv:1304.5530*, 2013.

R. Tappenden, P. Richtárik, and B. Büke. Separable approximations and decomposition methods for the augmented Lagrangian. *Optimization Methods and Software*, 30(3):643–668, 2015.

Journal of Machine Learning Research 17 (2016) 1-36 Submitted 1/15; Revised 4/16; Published 6/16

Scaling-up Empirical Risk Minimization: Optimization of Incomplete U-statistics

Stephan Clémençon STEPHAN.CLEMENCON@TELECOM-PARISTECH.FR
Igor Colin IGOR.COLIN@TELECOM-PARISTECH.FR
LTCI, CNRS, Télécom ParisTech
Université Paris-Saclay, 75013, Paris, France

Aurélien Bellet AURELIEN.BELLET@INRIA.FR
Magnet Team, INRIA Lille – Nord Europe
59650 Villeneuve d'Ascq, France

Editor: Xiaotong Shen

Abstract

In a wide range of statistical learning problems such as ranking, clustering or metric learning among others, the risk is accurately estimated by U-statistics of degree $d \geq 1$, *i.e.* functionals of the training data with low variance that take the form of averages over k-tuples. From a computational perspective, the calculation of such statistics is highly expensive even for a moderate sample size n, as it requires averaging $O(n^d)$ terms. This makes learning procedures relying on the optimization of such data functionals hardly feasible in practice. It is the major goal of this paper to show that, strikingly, such empirical risks can be replaced by drastically computationally simpler Monte-Carlo estimates based on $O(n)$ terms only, usually referred to as *incomplete U-statistics*, without damaging the $O_{\mathbb{P}}(1/\sqrt{n})$ learning rate of *Empirical Risk Minimization* (ERM) procedures. For this purpose, we establish uniform deviation results describing the error made when approximating a U-process by its incomplete version under appropriate complexity assumptions. Extensions to model selection, fast rate situations and various sampling techniques are also considered, as well as an application to stochastic gradient descent for ERM. Finally, numerical examples are displayed in order to provide strong empirical evidence that the approach we promote largely surpasses more naive subsampling techniques.

Keywords: big data, empirical risk minimization, U-processes, rate bound analysis, sampling design, stochastic gradient descent

1. Introduction

In classification/regression, empirical risk estimates are sample mean statistics and the theory of *Empirical Risk Minimization* (ERM) has been originally developed in this context, see Devroye et al. (1996). The ERM theory essentially relies on the study of maximal deviations between these empirical averages and their expectations, under adequate complexity assumptions on the set of prediction rule candidates. The relevant tools are mainly concentration inequalities for empirical processes, see Ledoux and Talagrand (1991) for instance.

In a wide variety of problems that received a good deal of attention in the machine learning literature and ranging from clustering to image recognition through ranking or learning on graphs, natural estimates of the risk are not basic sample means but take the form of averages of d-tuples, usually referred to as U-statistics in Probability and Statistics, see Lee (1990). In Clémençon et al. (2005) for instance, ranking is viewed as pairwise classification and the empirical ranking error of any given prediction rule is a U-statistic of order 2, just like the *within cluster point scatter* in cluster analysis (see Clémençon, 2014) or empirical performance measures in metric learning, refer to Cao et al. (2012) for instance. Because empirical functionals are computed by averaging over tuples of sampling observations, they exhibit a complex dependence structure, which appears as the price to be paid for low variance estimates. *Linearization techniques* (see Hoeffding, 1948) are the main ingredient in studying the behavior of empirical risk minimizers in this setting, allowing to establish probabilistic upper bounds for the maximal deviation of collection of centered U-statistics under appropriate conditions by reducing the analysis to that of standard empirical processes. However, while the ERM theory based on minimization of U-statistics is now consolidated (see Clémençon et al., 2008), putting this approach in practice generally leads to significant computational difficulties that are not sufficiently well documented in the machine learning literature. In many concrete cases, the mere computation of the risk involves a summation over an extremely high number of tuples and runs out of time or memory on most machines.

Whereas the availability of massive information in the Big Data era, which machine learning procedures could theoretically now rely on, has motivated the recent development of *parallelized / distributed* approaches in order to scale-up certain statistical learning algorithms, see Bekkerman et al. (2011) or Bianchi et al. (2013) and the references therein, the present paper proposes to use *sampling techniques* as a remedy to the apparent intractability of learning from data sets of explosive size, in order to break the current computational barriers. More precisely, it is the major goal of this article to study how a simplistic sampling technique (*i.e.* drawing with replacement) applied to risk estimation, as originally proposed by Blom (1976) in the context of asymptotic pointwise estimation, may efficiently remedy this issue without damaging too much the "reduced variance" property of the estimates, while preserving the learning rates (including certain "fast-rate" situations). For this purpose, we investigate to which extent a U-process, that is a collection of U-statistics, can be accurately approximated by a Monte-Carlo version (which shall be referred to as an *incomplete U-process* throughout the paper) involving much less terms, provided it is indexed by a class of kernels of controlled complexity (in a sense that will be explained later). A maximal deviation inequality connecting the accuracy of the approximation to the number of terms involved in the approximant is thus established. This result is the key to the analysis of the statistical performance of minimizers of risk estimates when they are in the form of an incomplete U-statistic. In particular, this allows us to show the advantage of using this specific sampling technique, compared to more naive approaches with exactly the same computational cost, consisting for instance in first drawing a subsample and then computing a risk estimate of the form of a (complete) U-statistic based on it. We also show how to incorporate this sampling strategy into iterative statistical learning techniques based on stochastic gradient descent (SGD), see Bottou (1998). The variant of the SGD method we propose involves the computation of an incomplete U-statistic to estimate the gradient at each step. For the estimator thus produced, rate bounds describing its statistical performance are established under mild assumptions. Beyond theoretical results, we

present illustrative numerical experiments on metric learning and clustering with synthetic and real-world data that support the relevance of our approach.

The rest of the article is organized as follows. In Section 2, we recall basic definitions and concepts pertaining to the theory of U-statistics/processes and present important examples in machine learning where natural estimates of the performance/risk measure are U-statistics. We then review the existing results for the empirical minimization of complete U-statistics. In Section 3, we recall the notion of incomplete U-statistic and we derive maximal deviation inequalities describing the error made when approximating a U-statistic by its incomplete counterpart uniformly over a class of kernels that fulfills appropriate complexity assumptions. This result is next applied to derive (possibly fast) learning rates for minimizers of the incomplete version of the empirical risk and to model selection. Extensions to incomplete U-statistics built by means of other sampling schemes than sampling with replacement are also investigated. In Section 4, estimation by means of incomplete U-statistics is applied to stochastic gradient descent for iterative ERM. Section 5 presents some numerical experiments. Finally, Section 6 collects some concluding remarks. Technical details are deferred to the Appendix.

2. Background and Preliminaries

As a first go, we briefly recall some key notions of the theory of U-statistics (Section 2.1) and provide several examples of statistical learning problems for which natural estimates of the performance/risk measure are in the form of U-statistics (Section 2.2). Finally, we review and extend the existing rate bound analysis for the empirical minimization of (complete) generalized U-statistics (Section 2.3). Here and throughout, $\mathbb{N}^*$ denotes the set of all strictly positive integers, $\mathbb{R}_+$ the set of nonnegative real numbers.

2.1 U-Statistics/Processes: Definitions and Properties

For clarity, we recall the definition of generalized U-statistics. An excellent account of properties and asymptotic theory of U-statistics can be found in Lee (1990).

Definition 1 (Generalized U-statistic) *Let $K \geq 1$ and $(d_1, \ldots, d_K) \in \mathbb{N}^{*K}$. Let $\mathbf{X}_{\{1,\ldots,n_k\}} = (X_1^{(k)}, \ldots, X_{n_k}^{(k)})$, $1 \leq k \leq K$, be K independent samples of sizes $n_k \geq d_k$ and composed of i.i.d. random variables taking their values in some measurable space $\mathcal{X}_k$ with distribution $F_k(dx)$ respectively. Let $H : \mathcal{X}_1^{d_1} \times \cdots \times \mathcal{X}_K^{d_K} \to \mathbb{R}$ be a measurable function, square integrable with respect to the probability distribution $\mu = F_1^{\otimes d_1} \otimes \cdots \otimes F_K^{\otimes d_K}$. Assume in addition (without loss of generality) that $H(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(K)})$ is symmetric within each block of arguments $\mathbf{x}^{(k)}$ (valued in $\mathcal{X}_k^{d_k}$), $1 \leq k \leq K$. The generalized (or K-sample) U-statistic of degrees $(d_1, \ldots, d_K)$ with kernel H, is then defined as*

$$U_{\mathbf{n}}(H) = \frac{1}{\prod_{k=1}^{K} \binom{n_k}{d_k}} \sum_{I_1} \cdots \sum_{I_K} H(\mathbf{X}_{I_1}^{(1)}, \mathbf{X}_{I_2}^{(2)}, \ldots, \mathbf{X}_{I_K}^{(K)}), \tag{1}$$

where the symbol $\sum_{I_k}$ refers to summation over all $\binom{n_k}{d_k}$ subsets $\mathbf{X}_{I_k}^{(k)} = (X_{i_1}^{(k)}, \ldots, X_{i_{d_k}}^{(k)})$ related to a set I_k of d_k indexes $1 \leq i_1 < \ldots < i_{d_k} \leq n_k$ and $\mathbf{n} = (n_1, \ldots, n_K)$.

The above definition generalizes standard sample mean statistics, which correspond to the case $K = 1 = d_1$. More generally when $K = 1$, $U_{\mathbf{n}}(H)$ is an average over all d_1-tuples of observations, while $K \geq 2$ corresponds to the multi-sample situation with a d_k-tuple for each sample $k \in \{1, \ldots, K\}$. A U-process is defined as a collection of U-statistics indexed by a set $\mathcal{H}$ of kernels. This concept generalizes the notion of empirical process.

Many statistics used for pointwise estimation or hypothesis testing are actually generalized U-statistics (*e.g.* the sample variance, the Gini mean difference, the Wilcoxon Mann-Whitney statistic, Kendall tau). Their popularity mainly arises from their "reduced variance" property: the statistic $U_{\mathbf{n}}(H)$ has minimum variance among all unbiased estimators of the parameter

$$\begin{aligned} \mu(H) &= \mathbb{E}\left[H(X_1^{(1)}, \ldots, X_{d_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{d_K}^{(K)})\right] \\ &= \int_{\mathbf{x}^{(1)} \in \mathcal{X}_1^{d_1}} \cdots \int_{\mathbf{x}^{(K)} \in \mathcal{X}_K^{d_K}} H(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(K)}) dF_1^{\otimes d_1}(\mathbf{x}^{(1)}) \cdots dF_K^{\otimes d_K}(\mathbf{x}^{(K)}) = \mathbb{E}\left[U_{\mathbf{n}}(H)\right]. \end{aligned} \tag{2}$$

Classically, the limit properties of these statistics (law of large numbers, central limit theorem, *etc.*) are investigated in an asymptotic framework stipulating that, as the size of the full pooled sample

$$n \stackrel{\text{def}}{=} n_1 + \ldots + n_K \tag{3}$$

tends to infinity, we have:

$$n_k/n \to \lambda_k > 0 \text{ for } k = 1, \ldots, K. \tag{4}$$

Asymptotic results and deviation/moment inequalities for K-sample U-statistics can be classically established by means of specific representations of this class of functionals, see (15) and (27) introduced in later sections. Significant progress in the analysis of U-statistics and U-processes has then recently been achieved by means of decoupling theory, see de la Peña and Giné (1999). For completeness, we point out that the asymptotic behavior of (multisample) U-statistics has been investigated under weaker integrability assumptions than that stipulated in Definition 1, see Lee (1990).

2.2 Motivating Examples

In this section, we review important supervised and unsupervised statistical learning problems where the empirical performance/risk measure is of the form of a generalized U-statistics. They shall serve as running examples throughout the paper.

2.2.1 Clustering

Clustering refers to the unsupervised learning task that consists in partitioning a set of data points $X_1, \ldots, X_n$ in a feature space $\mathcal{X}$ into a finite collection of subgroups depending on their similarity (in a sense that must be specified): roughly, data points in the same subgroup should be more similar to each other than to those lying in other subgroups. One may refer to Chapter 14 in Friedman et al. (2009) for an account of state-of-the-art clustering techniques. Formally, let $M \geq 2$ be the number of desired clusters and consider a symmetric function $D : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ such that $D(x, x) = 0$ for any $x \in \mathcal{X}$. D measures the

dissimilarity between pairs of observations $(x, x') \in \mathcal{X}^2$: the larger $D(x, x')$, the less similar x and x'. For instance, if $\mathcal{X} \subset \mathbb{R}^d$, D could take the form $D(x, x') = \Psi(\|x - x'\|_q)$, where $q \geq 1$, $\|a\|_q = (\sum_{i=1}^d |a_i|^q)^{1/q}$ for all $a \in \mathbb{R}^d$ and $\Psi : \mathbb{R}_+ \to \mathbb{R}_+$ is any borelian nondecreasing function such that $\Psi(0) = 0$. In this context, the goal of clustering methods is to find a partition $\mathcal{P}$ of the feature space $\mathcal{X}$ in a class Π of partition candidates that minimizes the following *empirical clustering risk*:

$$\widehat{W}_n(\mathcal{P}) = \frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} D(X_i, X_j) \cdot \Phi_{\mathcal{P}}(X_i, X_j), \tag{5}$$

where $\Phi_{\mathcal{P}}(x, x') = \sum_{C \in \mathcal{P}} \mathbb{I}\{(x, x') \in C^2\}$. Assuming that the data $X_1, \ldots, X_n$ are i.i.d. realizations of a generic random variable X drawn from an unknown probability distribution $F(dx)$ on $\mathcal{X}$, the quantity $\widehat{W}_n(\mathcal{P})$, also known as the *intra-cluster similarity* or *within cluster point scatter*, is a one sample U-statistic of degree two ($K = 1$ and $d_1 = 2$) with kernel given by:

$$\forall (x, x') \in \mathcal{X}^2, \quad H_{\mathcal{P}}(x, x') = D(x, x') \cdot \Phi_{\mathcal{P}}(x, x'), \tag{6}$$

according to Definition 1 provided that $\int\int_{(x,x') \in \mathcal{X}^2} D^2(x, x') \cdot \Phi_{\mathcal{P}}(x, x') F(dx) F(dx') < +\infty$. The expectation of the empirical clustering risk $\widehat{W}_n(\mathcal{P})$ is given by

$$W(\mathcal{P}) = \mathbb{E}\left[D(X, X') \cdot \Phi_{\mathcal{P}}(X, X')\right], \tag{7}$$

where X' is an independent copy of the r.v. X, and is named the *clustering risk* of the partition $\mathcal{P}$. The statistical analysis of the clustering performance of minimizers $\widehat{\mathcal{P}}_n$ of the empirical risk (5) over a class Π of appropriate complexity can be found in Clémençon (2014). Based on the theory of U-processes, it is shown in particular how to establish rate bounds for the excess of clustering risk of any empirical minimizer, $W(\widehat{\mathcal{P}}_n) - \inf_{\mathcal{P} \in \Pi} W(\mathcal{P})$ namely, under appropriate complexity assumptions on the cells forming the partition candidates.

2.2.2 Metric Learning

Many problems in machine learning, data mining and pattern recognition (such as the clustering problem described above) rely on a metric to measure the distance between data points. Choosing an appropriate metric for the problem at hand is crucial to the performance of these methods. Motivated by a variety of applications ranging from computer vision to information retrieval through bioinformatics, metric learning aims at adapting the metric to the data and has attracted a lot of interest in recent years (see for instance Bellet et al., 2013, for an account of metric learning and its applications). As an illustration, we consider the metric learning problem for supervised classification. In this setting, we observe independent copies $(X_1, Y_1), \ldots, (X_n, Y_n)$ of a random couple (X, Y), where the r.v. X takes values in some feature space $\mathcal{X}$ and Y in a finite set of labels, $\mathcal{Y} = \{1, \ldots, C\}$ with $C \geq 2$ say. Consider a set $\mathcal{D}$ of distance measures $D : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$. Roughly speaking, the goal of metric learning in this context is to find a metric under which pairs of points with the same label are close to each other and those with different labels are far away. The risk of a metric D can be expressed as:

$$R(D) = \mathbb{E}\left[\phi\left((1 - D(X, X') \cdot (2\mathbb{I}\{Y = Y'\} - 1)\right)\right], \tag{8}$$

where $\phi(u)$ is a convex loss function upper bounding the indicator function $\mathbb{I}\{u \geq 0\}$, such as the hinge loss $\phi(u) = \max(0, 1 - u)$. The natural empirical estimator of this risk is

$$R_n(D) = \frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} \phi\left((D(X_i, X_j) - 1) \cdot (2\mathbb{I}\{Y_i = Y_j\} - 1)\right), \tag{9}$$

which is a one sample U-statistic of degree two with kernel given by:

$$H_D\left((x, y), (x', y)\right) = \phi\left((D(x, x') - 1) \cdot (2\mathbb{I}\{y = y'\} - 1)\right). \tag{10}$$

The convergence to (8) of a minimizer of (9) has been studied in the frameworks of algorithmic stability (Jin et al., 2009), algorithmic robustness (Bellet and Habrard, 2015) and based on the theory of U-processes under appropriate regularization (Cao et al., 2012).

2.2.3 Multipartite Ranking

Given objects described by a random vector of attributes/features $X \in \mathcal{X}$ and the (temporarily hidden) ordinal labels $Y \in \{1, \ldots, K\}$ assigned to it, the goal of *multipartite ranking* is to rank them in the same order as that induced by the labels, on the basis of a training set of labeled examples. This statistical learning problem finds many applications in a wide range of fields (*e.g.* medicine, finance, search engines, e-commerce). Rankings are generally defined by means of a scoring function $s : \mathcal{X} \to \mathbb{R}$, transporting the natural order on the real line onto the feature space and the gold standard for evaluating the ranking performance of $s(x)$ is the ROC manifold, or its usual summary the VUS criterion (VUS standing for *Volume Under the* ROC *Surface*), see Clémençon and Robbiano (2014) and the references therein. In Clémençon et al. (2013), optimal scoring functions have been characterized as those that are optimal for all bipartite subproblems. In other words, they are increasing transforms of the likelihood ratio dF_{k+1}/dF_k, where F_k denotes the class-conditional distribution for the k-th class. When the set of optimal scoring functions is non-empty, the authors also showed that it corresponds to the functions which maximize the volume under the ROC surface

$$VUS(s) = \mathbb{P}\{s(X_1) < \ldots < s(X_K) | Y_1 = 1, \ldots, Y_K = K\}.$$

Given K independent samples $(X_1^{(k)}, \ldots, X_{n_k}^{(k)}) \overset{\text{i.i.d.}}{\sim} F_k(dx)$ for $k = 1, \ldots, K$, the empirical counterpart of the VUS can be written in the following way:

$$\widehat{VUS}(s) = \frac{1}{\prod_{k=1}^K n_k} \sum_{i_1=1}^{n_1} \ldots \sum_{i_K=1}^{n_K} \mathbb{I}\{s(X_{i_1}^{(1)}) < \ldots < s(X_{i_K}^{(K)})\}. \tag{11}$$

The empirical VUS (11) is a K-sample U-statistic of degree $(1, \ldots, 1)$ with kernel given by:

$$H_s(x_1, \ldots, x_K) = \mathbb{I}\{s(x_1) < \ldots < s(x_K)\}. \tag{12}$$

2.3 Empirical Minimization of U-Statistics

As illustrated by the examples above, many learning problems can be formulated as finding a certain rule g in a class $\mathcal{G}$ in order to minimize a risk of the same form as (2), $\mu(H_g)$, with

kernel $H = H_g$. Based on $K \geq 1$ independent i.i.d. samples

$$\mathbf{X}^{(k)}_{\{1,\,\ldots,\,n_k\}} = (X^{(k)}_1,\,\ldots,\,X^{(k)}_{n_k}) \text{ with } 1 \leq k \leq K,$$

the ERM paradigm in statistical learning suggests to replace the risk by the U-statistic estimation $U_{\mathbf{n}}(H_g)$ in the minimization problem. The study of the performance of minimizers $\widehat{g}_{\mathbf{n}}$ of the empirical estimate $U_{\mathbf{n}}(H_g)$ over the class $\mathcal{G}$ of rule candidates naturally leads to analyze the fluctuations of the U-process

$$\{U_{\mathbf{n}}(H_g) - \mu(H_g) :\ g \in \mathcal{G}\}. \tag{13}$$

Given the bound

$$\mu(H_{\widehat{g}_{\mathbf{n}}}) - \inf_{g\in\mathcal{G}} \mu(H_g) \leq 2 \sup_{g\in\mathcal{G}} |U_{\mathbf{n}}(H_g) - \mu(H_g)|, \tag{14}$$

a probabilistic control of the maximal deviation $\sup_{g\in\mathcal{G}} |U_{\mathbf{n}}(H_g) - \mu(H_g)|$ naturally provides statistical guarantees for the generalization ability of the empirical minimizer $\widehat{g}_{\mathbf{n}}$. As shown at length in the case $K = 1$ and $d_1 = 2$ in Clémençon et al. (2008) and in Clémençon (2014) for specific problems, this can be achieved under adequate complexity assumptions of the class $\mathcal{H}_{\mathcal{G}} = \{H_g :\ g \in \mathcal{G}\}$. These results rely on the *Hoeffding's representation* of U-statistics, which we recall now for clarity in the general multisample U-statistics setting. Denote by $\mathfrak{S}_m$ the symmetric group of order m for any $m \geq 1$ and by $\sigma(i)$ the i-th coordinate of any permutation $\sigma \in \mathfrak{S}_m$ for $1 \leq i \leq m$. Let $\lfloor z \rfloor$ be the integer part of any real number z and set

$$N = \min\{\lfloor n_1/d_1 \rfloor,\,\ldots,\,\lfloor n_K/d_K \rfloor\}.$$

Observe that the K-sample U-statistic (1) can be expressed as

$$U_{\mathbf{n}}(H) = \frac{1}{\prod_{k=1}^K n_k!} \sum_{\sigma_1\in\mathfrak{S}_{n_1}} \cdots \sum_{\sigma_K\in\mathfrak{S}_{n_K}} V_H\left(X^{(1)}_{\sigma_1(1)},\,\ldots,\,X^{(K)}_{\sigma_K(n_K)}\right), \tag{15}$$

where

$$\begin{aligned} V_H\left(X^{(1)}_1,\,\ldots,\,X^{(1)}_{n_1},\,\ldots,\,X^{(K)}_1,\,\ldots,\,X^{(K)}_{n_K}\right) = \frac{1}{N}\Big[&H\left(X^{(1)}_1,\,\ldots,\,X^{(1)}_{d_1},\,\ldots,\,X^{(K)}_1,\,\ldots,\,X^{(K)}_{d_K}\right) \\ &+ H\left(X^{(1)}_{d_1+1},\,\ldots,\,X^{(1)}_{2d_1},\,\ldots,\,X^{(K)}_{d_K+1},\,\ldots,\,X^{(K)}_{2d_K}\right) + \ldots \\ &+ H\left(X^{(1)}_{(N-1)d_1+1},\,\ldots,\,X^{(1)}_{Nd_1},\,\ldots,\,X^{(K)}_{(N-1)d_K+1},\,\ldots,\,X^{(K)}_{Nd_K}\right)\Big]. \end{aligned}$$

This representation, sometimes referred to as the *first Hoeffding's decomposition* (see Hoeffding, 1948), allows to reduce a first order analysis to the case of sums of i.i.d. random variables. The following result extends Corollary 3 in Clémençon et al. (2008) to the multisample situation.

Proposition 2 *Let $\mathcal{H}$ be a collection of bounded symmetric kernels on $\prod_{k=1}^K \mathcal{X}_k^{d_k}$ such that*

$$M_{\mathcal{H}} \stackrel{\text{def}}{=} \sup_{(H,x)\in\mathcal{H}\times\mathcal{X}} |H(x)| < +\infty. \tag{16}$$

Suppose also that $\mathcal{H}$ is a VC major class of functions with finite Vapnik-Chervonenkis dimension $V < +\infty$. For all $\delta \in (0,1)$, we have with probability at least $1-\delta$,

$$\sup_{H\in\mathcal{H}} |U_{\mathbf{n}}(H) - \mu(H)| \leq M_{\mathcal{H}} \left\{ 2\sqrt{\frac{2V\log(1+N)}{N}} + \sqrt{\frac{\log(1/\delta)}{N}} \right\}, \tag{17}$$

where $N = \min\{\lfloor n_1/d_1 \rfloor,\,\ldots,\,\lfloor n_K/d_K \rfloor\}$.

Observe that, in the usual asymptotic framework (4), the bound (17) shows that the learning rate is, as expected, of order $O_{\mathbb{P}}(\sqrt{\log n/n})$, where n denotes the size of the pooled sample.

Remark 3 (Uniform boundedness) *We point out that condition (16) is clearly satisfied for the class of kernels considered in the multipartite ranking situation, whatever the class of scoring functions considered. In the case of the clustering example, it is fulfilled as soon as the essential supremum of $D(X,X') \cdot \Phi_{\mathcal{P}}(X,X')$ is uniformly bounded over $\mathcal{P} \in \Pi$, whereas in the metric learning example, it is satisfied when the essential supremum of the r.v. $\phi((D(X,X')-1)\cdot(2\mathbb{I}\{Y=Y'\}-1))$ is uniformly bounded over $D \in \mathcal{D}$. We underline that this simplifying condition can be easily relaxed and replaced by appropriate tail assumptions for the variables $H(X^{(1)}_1,\,\ldots,\,X^{(K)}_{d_K})$, $H \in \mathcal{H}$, combining the arguments of the subsequent analysis with the classical "truncation trick" originally introduced in Fuk and Nagaev (1971).*

Remark 4 (Complexity assumptions) *Following in the footsteps of Clémençon et al. (2008) which considered 1-sample U-statistics of degree 2, define the Rademacher average*

$$\mathcal{R}_N = \sup_{H\in\mathcal{H}} \frac{1}{N} \left| \sum_{l=1}^N \epsilon_l H\left(X^{(1)}_{(l-1)d_1+1},\,\ldots,\,X^{(1)}_{ld_1},\,\ldots,\,X^{(K)}_{(l-1)d_K+1},\,\ldots,\,X^{(K)}_{ld_K}\right) \right|, \tag{18}$$

where $\epsilon_1,\,\ldots,\,\epsilon_N$ are independent Rademacher random variables (random symmetric sign variables), independent from the $X^{(k)}_i$'s. As can be seen by simply examining the proof of Proposition 2 (Appendix A), a control of the maximal deviations similar to (17) relying on this particular complexity measure can be obtained: the first term on the right hand side is then replaced by the expectation of the Rademacher average $\mathbb{E}[\mathcal{R}_N]$, up to a constant multiplicative factor. This expected value can be bounded by standard metric entropy techniques and in the case where $\mathcal{H}$ is a VC major class of functions of dimension V, we have:

$$\mathbb{E}[\mathcal{R}_N] \leq M_{\mathcal{H}} \sqrt{\frac{2V\log(N+1)}{N}}.$$

See Appendix A for further details.

3. Empirical Minimization of Incomplete U-Statistics

We have seen in the last section that the empirical minimization of U-statistics leads to a learning rate of $O_{\mathbb{P}}(\sqrt{\log n/n})$. However, the computational cost required to find the empirical minimizer in practice is generally prohibitive, as the number of terms to be summed up to compute the U-statistic (1) is equal to:

$$\binom{n_1}{d_1} \times \cdots \times \binom{n_K}{d_K}.$$

In the usual asymptotic framework (4), it is of order $O(n^{d_1+\ldots+d_K})$ as $n \to +\infty$. It is the major purpose of this section to show that, in the minimization problem, the U-statistic $U_{\mathbf{n}}(H_g)$ can be replaced by a Monte-Carlo estimation, referred to as an *incomplete* U-*statistic*, whose computation requires to average much less terms, without damaging the learning rate (Section 3.1). We further extend these results to model selection (Section 3.2), fast rates situations (Section 3.3) and alternative sampling strategies (Section 3.4).

3.1 Uniform Approximation of Generalized U-Statistics

As a remedy to the computational issue mentioned above, the concept of *incomplete generalized* U-*statistic* has been introduced in the seminal contribution of Blom (1976). The calculation of such a functional involves a summation over low cardinality subsets of the $\binom{n_k}{d_k}$ d_k-tuples of indices, $1 \leq k \leq K$, solely. In the simplest formulation, the subsets of indices are obtained by *sampling independently with replacement*, leading to the following definition.

Definition 5 (Incomplete Generalized U-statistic) *Let $B \geq 1$. The incomplete version of the U-statistic (1) based on B terms is defined by:*

$$\widetilde{U}_B(H) = \frac{1}{B} \sum_{I=(I_1,\ldots,I_K)\in\mathcal{D}_B} H(\mathbf{X}_{I_1}^{(1)}, \ldots, \mathbf{X}_{I_K}^{(K)}) = \frac{1}{B} \sum_{I\in\mathcal{D}_B} H(\mathbf{X}_I), \tag{19}$$

where $\mathcal{D}_B$ is a set of cardinality B built by sampling with replacement in the set

$$\Lambda = \{((i_1^{(1)}, \ldots, i_{d_1}^{(1)}), \ldots, (i_1^{(K)}, \ldots, i_{d_K}^{(K)})) : 1 \leq i_1^{(k)} < \ldots < i_{d_k}^{(k)} \leq n_k,\ 1 \leq k \leq K\}, \tag{20}$$

and $\mathbf{X}_I = (\mathbf{X}_{I_1}^{(1)}, \ldots, \mathbf{X}_{I_K}^{(K)})$ for all $I = (I_1, \ldots, I_K) \in \Lambda$.

We stress that the distribution of a complete U-statistic built from subsamples of reduced sizes n_k' drawn uniformly at random is quite different from that of an incomplete U-statistic based on $B = \prod_{k=1}^K \binom{n_k'}{d_k}$ terms sampled with replacement in Λ, although they involve the summation of the same number of terms, as depicted by Fig. 1.

In practice, B should be chosen much smaller than the cardinality of Λ, namely $\#\Lambda = \prod_{k=1}^K \binom{n_k}{d_k}$, in order to overcome the computational issue previously mentioned. We emphasize the fact that the cost related to the computation of the value taken by the kernel H at a given point $(x_{I_1}^{(1)}, \ldots, x_{I_K}^{(K)})$ depending on the form of H is not considered here: the focus is on the number of terms involved in the summation solely. As an estimator of $\alpha(H)$, the statistic (19) is still unbiased, *i.e.* $\mathbb{E}[\widetilde{U}_B(H)] = \alpha(H)$, but its variance is naturally larger than that of the complete U-statistic $U_n(H)$. Precisely, writing the variance of the r.v. $\widetilde{U}_B(H)$ as the expectation of its conditional variance given $(\mathbf{X}_I)_{I\in\Lambda}$ plus the variance of its conditional expectation given $(\mathbf{X}_I)_{I\in\Lambda}$, we obtain

$$\mathrm{Var}(\widetilde{U}_B(H)) = \left(1 - \frac{1}{B}\right)\mathrm{Var}(U_{\mathbf{n}}(H)) + \frac{1}{B}\mathrm{Var}(H(X_1^{(1)}, \ldots, X_{d_K}^{(K)})). \tag{21}$$

One may easily check that $\mathrm{Var}(\widetilde{U}_B(H)) \geq \mathrm{Var}(U_{\mathbf{n}}(H))$, and the difference vanishes as B increases. Refer to Lee (1990) for further details (see p. 193 therein). Incidentally,

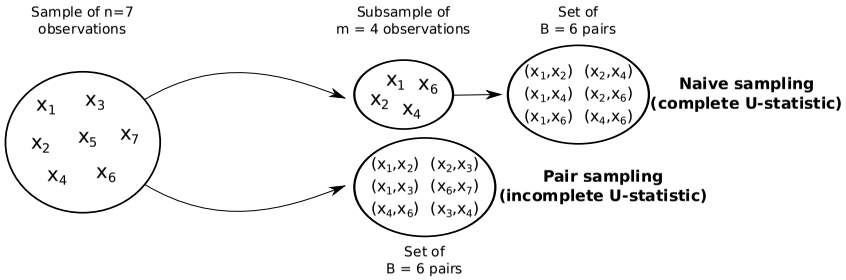


Figure 1: Illustration of the difference between an incomplete U-statistic and a complete U-statistic based on a subsample. For simplicity, we focus on the case $K = 1$ and $d_1 = 2$. In this simplistic example, a sample of $n = 7$ observations is considered. To construct a complete U-statistic of reduced complexity, we first sample a set of $m = 4$ observations and then form all possible pairs from this subsample, *i.e.* $B = m(m-1)/2 = 6$ pairs in total. In contrast, an incomplete U-statistic with the same number of terms is obtained by sampling B pairs directly from the set Λ of all possible pairs based on the original statistical population.

we underline that the empirical variance of (19) is not easy to compute either since it involves summing approximately $\#\Lambda$ terms and bootstrap techniques should be used for this purpose, as proposed in Bertail and Tressou (2006). The asymptotic properties of incomplete U-statistics have been investigated in several articles, see Janson (1984); Brown and Kildea (1978); Enqvist (1978). The angle embraced in the present paper is of very different nature: the key idea we promote here is to use incomplete versions of collections of U-statistics in learning problems such as that described in Section 2.2. The result stated below shows that this approach solves the numerical problem, while not damaging the learning rates under appropriate complexity assumptions on the collection $\mathcal{H}$ of (symmetric) kernels H considered, the complexity being described here in terms of VC dimension for simplicity. In particular, it reveals that concentration results established for U-processes (*i.e.* collections of U-statistics) such as Proposition 2 may extend to their incomplete versions, as shown by the following theorem.

Theorem 6 (Maximal deviation) *Let $\mathcal{H}$ be a collection of bounded symmetric kernels on $\prod_{k=1}^K \mathcal{X}_k^{d_k}$ that fulfills the assumptions of Proposition 2. Then, the following assertions hold true.*

(i) For all $\delta \in (0,1)$, with probability at least $1-\delta$, we have: $\forall \mathbf{n} = (n_1, \ldots, n_K) \in \mathbb{N}^{*K}$, $\forall B \geq 1$,

$$\sup_{H\in\mathcal{H}} \left|\widetilde{U}_B(H) - U_{\mathbf{n}}(H)\right| \leq \mathcal{M}_{\mathcal{H}} \times \sqrt{2\frac{V\log(1+\#\Lambda) + \log(2/\delta)}{B}}$$

(ii) For all $\delta \in (0,1)$, with probability at least $1-\delta$, we have: $\forall \mathbf{n} \in \mathbb{N}^{*K}$, $\forall B \geq 1$,

$$\frac{1}{\mathcal{M}_{\mathcal{H}}} \sup_{H \in \mathcal{H}} \left| \widetilde{U}_B(H) - U(H) \right| \leq 2\sqrt{\frac{2V\log(1+N)}{N}} + \sqrt{\frac{\log(2/\delta)}{N}} + \sqrt{2\frac{V\log(1+\#\Lambda)+\log(4/\delta)}{B}},$$

where $N = \min\{\lfloor n_1/d_1 \rfloor, \ldots, \lfloor n_K/d_K \rfloor\}$.

Remark 7 (Complexity assumptions continued) *We point out that a bound of the same order as that stated above can be obtained under standard metric entropy conditions by means of classical chaining arguments, or under the assumption that the Rademacher average defined by*

$$\widetilde{\mathcal{R}}_B = \sup_{H \in \mathcal{H}} \frac{1}{B} \left| \sum_{b=1}^{B} \epsilon_b \left\{ \sum_{I \in \Lambda} \zeta_b(I) H(\mathbf{X}_I) \right\} \right| \tag{22}$$

has an expectation of the order $O(1/\sqrt{B})$. The quantity $\zeta_b(I)$ indicates whether the subset of indexes I has been picked at the b-th draw ($\zeta_b(I) = +1$) or not ($\zeta_b(I) = 0$), see the calculation at the end of Appendix C. Equipped with this notation, notice that the ζ_b's are i.i.d. multinomial random variables such that $\sum_{I \in \Lambda} \zeta_b(I) = +1$. This assumption can be easily shown to be fulfilled in the case where $\mathcal{H}$ is a VC major class of finite VC dimension (see the proof of Theorem 6 in Appendix B). Notice however that although the variables $\sum_{I \in \Lambda} \zeta_b(I) H(\mathbf{X}_I)$, $1 \leq b \leq B$, are conditionally i.i.d. given $(\mathbf{X}_I)_{I \in \Lambda}$, they are not independent and the quantity (22) cannot be related to complexity measures of the type (18) mentioned in Remark 4.

Remark 8 *We underline that, whereas $\sup_{H \in \mathcal{H}} |U_{\mathbf{n}}(H) - U(H)|$ can be proved to be of order $O_{\mathbb{P}}(1/n)$ under adequate complexity assumptions in the specific situation where $\{U_{\mathbf{n}}(H) : H \in \mathcal{H}\}$ is a collection of degenerate U-statistics (see Section 3.3), the bound (i) in Theorem 6 cannot be improved in the degenerate case. Observe indeed that, conditioned upon the observations $X_i^{(k)}$, the deviations of the approximation (19) from its mean are of order $O_{\mathbb{P}}(1/\sqrt{B})$, since it is a basic average of B i.i.d. terms.*

From the theorem stated above, one may straightforwardly deduce a bound on the excess risk of kernels $\widehat{H}_B$ minimizing the incomplete version of the empirical risk based on B terms, *i.e.* such that

$$\widetilde{U}_B\left(\widehat{H}_B\right) = \min_{H \in \mathcal{H}} \widetilde{U}_B(H). \tag{23}$$

Corollary 9 *Let $\mathcal{H}$ be a collection of symmetric kernels on $\prod_{k=1}^{K} \mathcal{X}_k^{d_k}$ that satisfies the conditions stipulated in Proposition 2. Let $\delta > 0$. For any minimizer $\widehat{H}_B$ of the statistical estimate of the risk (19), the following assertions hold true*

(i) We have with probability at least $1-\delta$: $\forall \mathbf{n} \in \mathbb{N}^{*K}$, $\forall B \geq 1$,

$$U(\widehat{H}_B) - \inf_{H \in \mathcal{H}} U(H) \leq 2\mathcal{M}_{\mathcal{H}} \times \left\{ 2\sqrt{\frac{2V\log(1+N)}{N}} + \sqrt{\frac{\log(2/\delta)}{N}} + \sqrt{2\frac{V\log(1+\#\Lambda)+\log(4/\delta)}{B}} \right\}.$$

Empirical risk criterion	Nb of terms	Rate bound
Complete U-statistic	$O(n^{d_1+\ldots+d_K})$	$O_{\mathbb{P}}(\sqrt{\log(n)/n})$
Complete U-statistic based on subsamples	$O(n)$	$O_{\mathbb{P}}\left(\sqrt{\log(n)/n^{\frac{1}{d_1+\ldots+d_K}}}\right)$
Incomplete U-statistic (our result)	$\mathbf{O(n)}$	$\mathbf{O_{\mathbb{P}}(\sqrt{\log(n)/n})}$

Table 1: Rate bound for the empirical minimizer of several empirical risk criteria *versus* the number of terms involved in the computation of the criterion. For a computational budget of $O(n)$ terms, the rate bound for the incomplete U-statistic criterion is of the same order as that of the complete U-statistic, which is a huge improvement over a complete U-statistic based on a subsample.

(ii) We have: $\forall \mathbf{n} \in \mathbb{N}^{*K}$, $\forall B \geq 1$,

$$\mathbb{E}\left[\sup_{H \in \mathcal{H}} \left| \widetilde{U}_B(H) - U(H) \right| \right] \leq \mathcal{M}_{\mathcal{H}} \left\{ 2\sqrt{\frac{2V\log(1+N)}{N}} + \sqrt{\frac{2(\log 2 + V\log(1+\#\Lambda))}{B}} \right\}.$$

The first assertion of Theorem 6 provides a control of the deviations between the U-statistic (1) and its incomplete counterpart (19) uniformly over the class $\mathcal{H}$. As the number of terms B increases, this deviation decreases at a rate of $O(1/\sqrt{B})$. The second assertion of Theorem 6 gives a maximal deviation result with respect to $U(H)$. Observe in particular that, with the asymptotic settings previously specified, $N = O(n)$ and $\log(\#\Lambda) = O(\log n)$ as $n \to +\infty$. The bounds stated above thus show that, for a number $B = B_n$ of terms tending to infinity at a rate $O(n)$ as $n \to +\infty$, the maximal deviation $\sup_{H \in \mathcal{H}} |\widetilde{U}_B(H) - U(H)|$ is asymptotically of the order $O_{\mathbb{P}}((\log(n)/n)^{1/2})$, just like $\sup_{H \in \mathcal{H}} |U_{\mathbf{n}}(H) - U(H)|$, see bound (17) in Proposition 2. In short, when considering an incomplete U-statistic (19) with $B = O(n)$ terms only, the learning rate for the corresponding minimizer is of the same order as that of the minimizer of the complete risk (1), whose computation requires to average $\#\Lambda = O(n^{d_1+\ldots+d_K})$ terms. Minimizing such incomplete U-statistics thus yields a significant gain in terms of computational cost while fully preserving the learning rate. In contrast, as implied by Proposition 2, the minimization of a complete U-statistic involving $O(n)$ terms, obtained by drawing subsamples of sizes $n_k' = O(n^{1/(d_1+\ldots+d_K)})$ uniformly at random, leads to a rate of convergence of $O(\sqrt{\log(n)/n^{1/(d_1+\ldots+d_K)}})$, which is much slower except in the trivial case where $K = 1$ and $d_1 = 1$. These striking results are summarized in Table 1.

The important practical consequence of the above is that when n is too large for the complete risk (1) to be used, one should instead use the incomplete risk (19) (setting the number of terms B as large as the computational budget allows).

3.2 Model Selection Based on Incomplete U-Statistics

Automatic selection of the model complexity is a crucial issue in machine learning: it includes the number of clusters in cluster analysis (see Clémençon, 2014) or the choice of the number of possible values taken by a piecewise constant scoring function in multipartite ranking for instance (*cf.* Clémençon and Vayatis, 2009). In the present situation, this boils down to choosing the adequate level of complexity of the class of kernels $\mathcal{H}$, measured through its (supposedly finite) VC dimension for simplicity, in order to minimize the (theoretical) risk of the empirical minimizer. It is the purpose of this subsection to show that the incomplete U-statistic (19) can be used to define a penalization method to select a prediction rule with nearly minimal risk, avoiding procedures based on data splitting/resampling and extending the celebrated *structural risk minimization* principle, see Vapnik (1999). Let $\mathcal{H}$ be the collection of all symmetric kernels on $\prod_{k=1}^{K} \mathcal{X}_k^{d_k}$ and set $\mu^* = \inf_{H\in\mathcal{H}} \mu(H)$. Let $\mathcal{H}_1, \mathcal{H}_2, \ldots$ be a sequence of uniformly bounded major subclasses of $\mathcal{H}$, of increasing complexity (VC dimension). For any $m \geq 1$, let V_m denote the VC dimension of the class $\mathcal{H}_m$ and set $\mathcal{M}_{\mathcal{H}_m} = \sup_{(H,x)\in\mathcal{H}_m\times\mathcal{X}} |H(x)| < +\infty$. We suppose that there exists $\mathcal{M} < +\infty$ such that $\sup_{m\geq 1} \mathcal{M}_{\mathcal{H}_m} \leq \mathcal{M}$. Given $1 \leq B \leq \#\Lambda$ and $m \geq 1$, the complexity penalized empirical risk of a solution $\widetilde{U}_{B,m}$ of the ERM problem (23) with $\mathcal{H} = \mathcal{H}_m$ is

$$\widetilde{U}_B(\widehat{H}_{B,m}) + \mathrm{pen}(B, m), \tag{24}$$

where the quantity $\mathrm{pen}(B, m)$ is a *distribution free* penalty given by:

$$\begin{aligned} \mathrm{pen}(B, m) &= 2\mathcal{M}_{\mathcal{H}_m} \left\{ \sqrt{\frac{2V_m \log(1+N)}{N}} + \sqrt{\frac{2(\log 2 + V_m \log(1+\#\Lambda))}{B}} \right\} \\ &+ 2\mathcal{M}\sqrt{\frac{(B+n)\log m}{B^2}}. \end{aligned} \tag{25}$$

As shown in Assertion (ii) of Corollary 9, the quantity above is an upper bound for the expected maximal deviation $\mathbb{E}[\sup_{H\in\mathcal{H}_m} |\widetilde{U}_B(H) - \mu(H)|]$ and is thus a natural penalty candidate to compensate the overfitting within class $\mathcal{H}_m$. We thus propose to select

$$\widehat{m}_B = \arg\min_{m\geq 1} \left\{ \widetilde{U}_B(\widehat{H}_{B,m}) + \mathrm{pen}(B, m) \right\}. \tag{26}$$

As revealed by the theorem below, choosing $B = O(n)$, the prediction rule $\widehat{H}_{\widehat{m}_B}$ based on a penalized criterion involving the summation of $O(n)$ terms solely, achieves a nearly optimal trade-off between the bias and the distribution free upper bound (25) on the variance term.

Theorem 10 (Oracle inequality) *Suppose that Theorem 6's assumptions are fulfilled for all* $m \geq 1$ *and that* $\sup_{m\geq 1} \mathcal{M}_{\mathcal{H}_m} \leq \mathcal{M} < +\infty$. *Then, we have:* $\forall \mathbf{n} \in \mathbb{N}^{*K}$, $\forall B \in \{1, \ldots, \#\Lambda\}$,

$$\mu(\widehat{H}_{B,\widehat{m}}) - \mu^* \leq \inf_{k\geq 1} \left\{ \inf_{H\in\mathcal{H}_m} \mu(H) - \mu^* + \mathrm{pen}(B, m) \right\} + \mathcal{M}\frac{\sqrt{2\pi(B+n)}}{B}.$$

We point out that the argument used to obtain the above result can be straightforwardly extended to other (possibly data-dependent) complexity penalties (*cf.* Massart, 2006), see the proof in Appendix D.

3.3 Fast Rates for ERM of Incomplete U-Statistics

In Clémençon et al. (2008), it has been proved that, under certain "low-noise" conditions, the minimum variance property of the U-statistics used to estimate the ranking risk (corresponding to the situation $K = 1$ and $d_1 = 2$) leads to learning rates faster than $O_{\mathbb{P}}(1/\sqrt{n})$. These results rely on the *Hajek projection*, a linearization technique originally introduced in Hoeffding (1948) for the case of one sample U-statistics and next extended to the analysis of a much larger class of functionals in Hájek (1968). It consists in writing $U_{\mathbf{n}}(H)$ as the sum of the orthogonal projection

$$\widehat{U}_{\mathbf{n}}(H) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} \mathbb{E}\left[U_{\mathbf{n}}(H) \mid X_i^{(k)} \right] - (n-1)\mu(H), \tag{27}$$

which is itself a sum of K independent basic sample means based on i.i.d. r.v.'s (of the order $O_{\mathbb{P}}(1/\sqrt{n})$ each, after recentering), plus a possible negligible term. This representation was used for instance by Grams and Serfling (1973) to refine the CLT in the multisample U-statistics framework. Although useful as a theoretical tool, it should be noticed that the quantity $\widehat{U}_{\mathbf{n}}(H)$ is not of practical interest, since the conditional expectations involved in the summation are generally unknown.

Although incomplete U-statistics do not share the minimum variance property (see Section 3.1), we will show that the same fast rate bounds for the excess risk as those reached by ERM of U-statistics (corresponding to the summation of $O(n^2)$ pairs of observations) can be attained by empirical ranking risk minimizers, when estimating the ranking risk by incomplete U-statistics involving the summation of $o(n^2)$ terms solely.

For clarity (and comparison purpose), we first recall the statistical learning framework considered in Clémençon et al. (2008). Let (X, Y) be a pair of random variables defined on the same probability space, where Y is a real-valued label and X models some input information taking its values in a measurable space $\mathcal{X}$ hopefully useful to predict Y. Denoting by (X', Y') an independent copy of the pair (X, Y). The goal pursued here is to learn how to rank the input observations X and X', by means of an antisymmetric *ranking rule* $r : \mathcal{X}^2 \to \{-1, +1\}$ (*i.e.* $r(x, x') = -r(x'x)$ for any $(x, x') \in \mathcal{X}^2$), so as to minimize the *ranking risk*

$$L(r) = \mathbb{P}\{(Y - Y') \cdot r(X, X') < 0\}. \tag{28}$$

The minimizer of the ranking risk is the ranking rule $r^*(X, X') = 2\mathbb{I}\{\mathbb{P}\{Y > Y' \mid (X, X')\} \geq \mathbb{P}\{Y < Y' \mid (X, X')\}\} - 1$ (see Proposition 1 in Clémençon et al., 2008). The natural empirical counterpart of (28) based on a sample of independent copies $(X_1, Y_1), \ldots, (X_n, Y_n)$ of the pair (X, Y) is the 1-sample U-statistic $U_n(H_r)$ of degree two with kernel $H_r((x, y), (x', y')) = \mathbb{I}\{(y - y') \cdot r(x, x') < 0\}$ for all (x, y) and (x', y')) in $\mathcal{X} \times \mathbb{R}$ given by:

$$L_n(r) = U_n(H_r) = \frac{2}{n(n-1)} \sum_{i<j} \mathbb{I}\{(Y_i - Y_j) \cdot r(X_i, X_j) < 0\}. \tag{29}$$

Equipped with these notations, a statistical version of the excess risk $\Lambda(r) = L(r) - L(r^*)$ is a U-statistic $\lambda_n(r)$ with kernel $q_r = H_r - H_{r^*}$. The key "noise-condition", which allows to exploit the Hoeffding/Hajek decomposition of $\Lambda_n(r)$, is stated below.

Assumption 1 *There exist constants* $c > 0$ *and* $\alpha \in [0,1]$ *such that:*

$$\forall r \in \mathcal{R}, \quad \mathrm{Var}\left(h_r(X,Y)\right) \leq c\Lambda(r)^{\alpha},$$

where we set $h_r(x,y) = \mathbb{E}[q_r((x,y),(X',Y'))]$.

Recall incidentally that very general sufficient conditions guaranteeing that this assumption holds true have been exhibited, see Section 5 in Clémençon et al. (2008) (notice that the condition is void for $\alpha = 0$). Since our goal is to explain the main ideas rather than achieving a high level of generality, we consider a very simple setting, stipulating that the cardinality of the class of ranking rule candidates $\mathcal{R}$ under study is finite, $\#\mathcal{R} = M < +\infty$, and that the optimal rule r^* belongs to $\mathcal{R}$. The following proposition is a simplified version of the fast rate result proved in Clémençon et al. (2008) for the empirical minimizer $\widehat{r}_n = \arg\min_{r\in\mathcal{R}} L_n(r)$.

Proposition 11 *(Clémençon et al. (2008),* Corollary 6*) Suppose that Assumption 1 is fulfilled. Then, there exists a universal constant* $C > 0$ *such that for all* $\delta \in (0,1)$*, we have:* $\forall n \geq 2$*,*

$$L(\widehat{r}_n) - L(r^*) \leq C\left(\frac{\log(M/\delta)}{n}\right)^{\frac{1}{2-\alpha}}. \tag{30}$$

Consider now the minimizer $\widetilde{r}_B$ of the incomplete U-statistic risk estimate

$$\widetilde{U}_B(h_r) = \frac{1}{B}\sum_{k=1}^{B} \sum_{(i,j):1\leq i<j\leq n} \epsilon_k((i,j))\mathbb{I}\{(Y_i - Y_j)\cdot r(X_i,X_j) < 0\} \tag{31}$$

over $\mathcal{R}$, where $\epsilon_k((i,j))$ indicates whether the pair (i,j) has been picked at the k-th draw ($\epsilon_k((i,j)) = 1$ in this case, which occurs with probability $1/\binom{n}{2}$) or not (then, we set $\epsilon_k((i,j)) = 0$). Observe that $\widetilde{r}_B$ also minimizes the empirical estimate of the excess risk $\widetilde{\Lambda}_B(r) = \widetilde{U}_B(q_r)$ over $\mathcal{R}$.

Theorem 12 *Let* $\alpha \in [0,1]$ *and suppose that Assumption 1 is fulfilled. If we set* $B = O(n^{2/(2-\alpha)})$*, there exists some constant* $C < +\infty$ *such that, for all* $\delta \in (0,1)$*, we have with probability at least* $1-\delta$*:* $\forall n \geq 2$*,*

$$L(\widetilde{r}_B) - L(r^*) \leq C\left(\frac{\log(M/\delta)}{n}\right)^{\frac{1}{2-\alpha}}.$$

As soon as $\alpha < 1$, this result shows that the same fast rate of convergence as that reached by $\widehat{r}_n$ can be attained by the ranking rule $\widetilde{r}_B$, which minimizes an empirical version of the ranking risk involving the summation of $O(n^{2/(2-\alpha)})$ terms solely. For comparison purpose, minimization of the criterion (28) computed with a number of terms of the same order leads to a rate bound of order $O_{\mathbb{P}}(n^{1/(2-\alpha)^2})$.

Finally, we point out that fast rates for the clustering problem have been also investigated in Clémençon (2014), see Section 5.2 therein. The present analysis can be extended to the clustering framework by means of the same arguments.

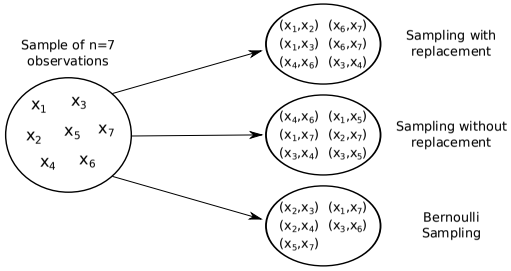


Figure 2: Illustration of different sampling schemes for approximating a U-statistic. For simplicity, consider again the case $K = 1$ and $d_1 = 2$. Here $n = 7$ and the expected number of terms is $B = 6$. Sampling with or without replacement results in exactly B terms, with possible repetitions when sampling with replacement, *e.g.* (x_6, x_7) in this example. In contrast, Bernoulli sampling with $\pi_I = B/\#\Lambda$ results in B terms only in expectation, with individual realizations that may exhibit more or fewer terms.

3.4 Alternative Sampling Schemes

Sampling with replacement is not the sole way of approximating generalized U-statistics with a controlled computational cost. As proposed in Janson (1984), other sampling schemes can be considered, Bernoulli sampling or sampling without replacement in particular (see Figure 2 for an illustration). We now explain how the results of this paper can be extended to these situations. The population of interest is the set Λ and a *survey sample* of (possibly random) size $b \leq n$ is any subset s of cardinality $b = b(s)$ less than $\#\Lambda$ in the power set $\mathcal{P}(\Lambda)$. Here, a general *survey scheme without replacement* is any conditional probability distribution R on the set of all possible samples $s \in \mathcal{P}(\Lambda)$ given $(\mathbf{X}_I)_{I\in\Lambda}$. For any $I \in \Lambda$, the first order *inclusion probability* $\pi_I(R) = \mathbb{P}_R\{I \in S\}$, is the probability that the unit I belongs to a random sample S drawn from distribution R. We set $\boldsymbol{\pi}(R) = (\pi_I(R))_{I\in\Lambda}$. The second order inclusion probabilities are denoted by $\pi_{I,J}(R) = \mathbb{P}_R\{(I,J) \in S^2\}$ for any $I \neq J$ in Λ. When no confusion is possible, we omit to mention the dependence in R when writing the first/second order probabilities of inclusion. The information related to the observed sample $S \subset \Lambda$ is fully enclosed in the random vector $\boldsymbol{\Delta} = (\Delta(I))_{I\in\Lambda}$, where $\Delta(I) = \mathbb{I}\{I \in S\}$ for all $I \in \Lambda$. The 1-d marginal distributions of the sampling scheme $\boldsymbol{\Delta}_n$ are the Bernoulli distributions with parameters π_I, $I \in \Lambda$, and the covariance matrix of the r.v. $\boldsymbol{\Delta}_n$ is given by $\Gamma = \{\pi_{I,J} - \pi_I\pi_J\}_{I,J}$ with the convention $\pi_{I,I} = \pi_I$ for all $I \in \Lambda$. Observe that, equipped with the notations above, $\sum_{I\in\Lambda}\Delta(I) = b(S)$.

One of the simplest survey plans is the Poisson scheme (without replacement), for which the $\Delta(I)$'s are independent Bernoulli random variables with parameters π_I, $I \in \Lambda$, in $(0,1)$. The first order inclusion probabilities fully characterize such a plan. Observe in addition that the size $b(\mathcal{S})$ of a sample generated this way is random with expectation $B = \mathbb{E}[b(S) \mid (\mathbf{X}_I)_{I\in\Lambda}] = \sum_{I\in\Lambda}\pi_I$. The situation where the π_I's are all equal corresponds to the Bernoulli

sampling scheme: $\forall I \in \Lambda$, $\pi_I = B/\#\Lambda$. The Poisson survey scheme plays a crucial role in sampling theory, inso far as a wide range of survey schemes can be viewed as conditional Poisson schemes, see Hájek (1964). For instance, one may refer to Cochran (1977) or Deville (1987) for accounts of survey sampling techniques.

Following in the footsteps of the seminal contribution of Horvitz and Thompson (1951), an estimate of (1) based on a sample drawn from a survey scheme R with first order inclusion probabilities $(\pi_I)_{I\in\Lambda}$ is given by:

$$\bar{U}_{HT}(H) = \frac{1}{\#\Lambda}\sum_{I\in\Lambda}\frac{\Delta(I)}{\pi_I}H(\mathbf{X}_I), \tag{32}$$

with the convention that $0/0 = 0$. Notice that it is an unbiased estimate of (1):

$$\mathbb{E}[\bar{U}_{HT}(H) \mid (\mathbf{X}_I)_{I\in\Lambda}] = U_{\mathbf{n}}(H).$$

In the case where the sample size is deterministic, its conditional variance is given by:

$$\mathrm{Var}(\bar{U}_{HT}(H) \mid (\mathbf{X}_I)_{I\in\Lambda}) = \frac{1}{2}\sum_{I\neq J}\left(\frac{H(\mathbf{X}_I)}{\pi_I} - \frac{H(\mathbf{X}_J)}{\pi_J}\right)^2(\pi_{I,J} - \pi_I\pi_J).$$

We point out that the computation of (32) involves summing over a possibly random number of terms, equal to $B = \mathbb{E}[b(S)] = \sum_{I\in\Lambda}\pi_I$ in average and whose variance is equal to $\mathrm{Var}(b(S)) = \sum_{I\in\Lambda}\pi_I(1-\pi_I) + \sum_{I\neq J}\{\pi_{I,J} - \pi_I\pi_J\}$.

Here, we are interested in the situation where the $\Delta(I)$'s are independent from $(X_I)_{I\in\Lambda}$, and either a sample of size $B \leq \#\Lambda$ fixed in advance is chosen uniformly at random among the $\binom{\#\Lambda}{B}$ possible choices (this survey scheme is sometimes referred to as *rejective sampling* with equal first order inclusion probabilities), or else it is picked by means of a Bernoulli sampling with parameter $B/\#\Lambda$. Observe that, in both cases, we have $\pi_I = B/\#\Lambda$ for all $I \in \Lambda$. The following theorem shows that in both cases, similar results as those obtained for *sampling with replacement* can be derived for minimizers of the Horvitz-Thompson risk estimate (32).

Theorem 13 *Let $\mathcal{H}$ be a collection of bounded symmetric kernels on $\prod_{k=1}^K \mathcal{X}_k^{d_k}$ that fulfills the assumptions involved in Proposition 2. Let $B \in \{1, \ldots, \#\Lambda\}$. Suppose that, for any $H \in \mathcal{H}$, $\bar{U}_{HT}(H)$ is the incomplete U-statistic based on either a Bernoulli sampling scheme with parameter $B/\#\Lambda$ or else a sampling without replacement scheme of size B. For all $\delta \in (0,1)$, we have with probability at least $1-\delta$: $\forall \mathbf{n} \in \mathbb{N}^{*K}$, $\forall B \in \{1, \ldots, \#\Lambda\}$,*

$$\sup_{H\in\mathcal{H}}\left|\bar{U}_{HT}(H) - U_{\mathbf{n}}(H)\right| \leq 2\mathcal{M}_{\mathcal{H}}\sqrt{\frac{\log(2(1+\#\Lambda)^V/\delta)}{B}} + \frac{2\log(2(1+\#\Lambda)^V/\delta)\mathcal{M}_{\mathcal{H}}}{3B},$$

in the case of the Bernoulli sampling design, and

$$\sup_{H\in\mathcal{H}}\left|\bar{U}_{HT}(H) - U_{\mathbf{n}}(H)\right| \leq \sqrt{2}\mathcal{M}_{\mathcal{H}}\sqrt{\frac{\log(2(1+\#\Lambda)^V/\delta)}{B}},$$

in the case of the sampling without replacement design.

We highlight the fact that, from a computational perspective, sampling with replacement is undoubtedly much more advantageous than Bernoulli sampling or sampling without replacement. Indeed, although its expected value is equal to B, the size of a Bernoulli sample is stochastic and the related sampling algorithm requires a loop through the elements I of Λ and the practical implementation of sampling without replacement is generally based on multiple iterations of sampling with replacement, see Tillé (2006).

4. Application to Stochastic Gradient Descent for ERM

The theoretical analysis carried out in the preceding sections focused on the properties of empirical risk minimizers but ignored the issue of finding such a minimizer. In this section, we show that the sampling technique introduced in Section 3 also provides practical means of scaling up iterative statistical learning techniques. Indeed, large-scale training of many machine learning models, such as SVM, Deep Neural Networks or soft K-means among others, is based on stochastic gradient descent (SGD in abbreviated form), see Bottou (1998). When the risk is of the form (2), we now investigate the benefit of using, at each iterative step, a gradient estimate of the form of an incomplete U-statistic, instead of an estimate of the form of a complete U-statistic with exactly the same number of terms based on subsamples drawn uniformly at random.

Let $\Theta \subset \mathbb{R}^q$ with $q \geq 1$ be some parameter space and $H : \prod_{k=1}^K \mathcal{X}_k^{d_k} \times \Theta \to \mathbb{R}$ be a loss function which is convex and differentiable in its last argument. Let $(X_1^{(k)}, \ldots, X_{d_k}^{(k)})$, $1 \leq k \leq K$, be K independent random vectors with distribution $F_k^{\otimes d_k}(dx)$ on $\mathcal{X}_k^{d_k}$ respectively such that the random vector $H(X_1^{(1)}, \ldots, X_{d_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{d_K}^{(K)}; \theta)$ is square integrable for any $\theta \in \Theta$. For all $\theta \in \Theta$, set

$$L(\theta) = \mathbb{E}[H(X_1^{(1)}, \ldots, X_{d_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{d_K}^{(K)}; \theta)] = \mu(H(\cdot; \theta))$$

and consider the *risk minimization* problem $\min_{\theta\in\Theta} L(\theta)$. Based on K independent i.i.d. samples $X_1^{(k)}, \ldots, X_{n_k}^{(k)}$ with $1 \leq k \leq K$, the empirical version of the risk function is $\theta \in \Theta \mapsto \widehat{L}_{\mathbf{n}}(\theta) \overset{\text{def}}{=} U_{\mathbf{n}}(H(\cdot; \theta))$. Here and throughout, we denote by ∇_θ the gradient operator w.r.t. θ.

Gradient descent Many practical machine learning algorithms use variants of the standard gradient descent method, following the iterations:

$$\theta_{t+1} = \theta_t - \eta_t\nabla_\theta\widehat{L}_{\mathbf{n}}(\theta_t), \tag{33}$$

with an arbitrary initial value $\theta_0 \in \Theta$ and a learning rate (step size) $\eta_t \geq 0$ such that $\sum_{t=1}^{+\infty}\eta_t = +\infty$ and $\sum_{t=1}^{+\infty}\eta_t^2 < +\infty$.

Here we place ourselves in a large-scale setting, where the sample sizes $n_1, \ldots, n_K$ of the training data sets are so large that computing the gradient of $\widehat{L}_{\mathbf{n}}$

$$\widehat{g}_{\mathbf{n}}(\theta) = \frac{1}{\prod_{k=1}^K\binom{n_k}{d_k}}\sum_{I_1}\cdots\sum_{I_K}\nabla_\theta H(\mathbf{X}_{I_1}^{(1)}; \mathbf{X}_{I_2}^{(2)}; \ldots; \mathbf{X}_{I_K}^{(K)}; \theta) \tag{34}$$

at each iteration (33) is computationally too expensive. Instead, Stochastic Gradient Descent uses an unbiased estimate $\widetilde{g}(\theta)$ of the gradient (34) that is cheap to compute. A

natural approach consists in replacing (34) by a complete U-statistic constructed from subsamples of reduced sizes $n'_k << n_k$ drawn uniformly at random, leading to the following gradient estimate:

$$\widetilde{g}_{\mathbf{n}'}(\theta) = \frac{1}{\prod_{k=1}^{K} \binom{n'_k}{d_k}} \sum_{I_1} \cdots \sum_{I_K} \nabla_\theta H(\mathbf{X}^{(1)}_{I_1}; \mathbf{X}^{(2)}_{I_2}; \ldots; \mathbf{X}^{(K)}_{I_K}; \theta), \tag{35}$$

where the symbol $\sum_{I_k}$ refers to summation over all $\binom{n'_k}{d_k}$ subsets $\mathbf{X}^{(k)}_{I_k} = (X^{(k)}_{i_1}, \ldots, X^{(k)}_{i_{d_k}})$ related to a set I_k of d_k indexes $1 \leq i_1 < \ldots < i_{d_k} \leq n'_k$ and $\mathbf{n}' = (n'_1, \ldots, n'_K)$.

We propose an alternative strategy based on the sampling scheme described in Section 3, *i.e.* a gradient estimate in the form of an *incomplete* U-statistic:

$$\widetilde{g}_B(\theta) = \frac{1}{B} \sum_{(I_1, \ldots, I_K) \in \mathcal{D}_B} \nabla_\theta H(\mathbf{X}^{(1)}_{I_1}, \ldots, \mathbf{X}^{(K)}_{I_K}; \theta), \tag{36}$$

where $\mathcal{D}_B$ is built by sampling with replacement in the set Λ.

It is well-known that the variance of the gradient estimate negatively impacts on the convergence of SGD. Consider for instance the case where the loss function H is $(1/\gamma)$-smooth in its last argument, *i.e.* $\forall \theta_1, \theta_2 \in \Theta$:

$$\|\nabla_\theta H(\cdot; \theta_1) - \nabla_\theta H(\cdot; \theta_2)\| \leq \frac{1}{\gamma}\|\theta_1 - \theta_2\|.$$

Then one can show that if $\widetilde{g}$ is the gradient estimate:

$$\begin{aligned}
\mathbb{E}[\widehat{L}_{\mathbf{n}}(\theta_{t+1})] &= \mathbb{E}[\widehat{L}_{\mathbf{n}}(\theta_t - \eta_t \widetilde{g}(\theta_t))] \\
&\leq \mathbb{E}[\widehat{L}_{\mathbf{n}}(\theta_t)] - \eta_t \|\mathbb{E}[\widehat{g}_{\mathbf{n}}(\theta_t)]\|^2 + \frac{\eta_t^2}{2\gamma}\mathbb{E}[\|\widetilde{g}(\theta_t)\|^2] \\
&\leq \mathbb{E}[\widehat{L}_{\mathbf{n}}(\theta_t)] - \eta_t \left(1 - \frac{\eta_t}{2\gamma}\right) \mathbb{E}[\|\widehat{g}_{\mathbf{n}}(\theta_t)\|^2] + \frac{\eta_t^2}{2\gamma}\mathrm{Var}[\widetilde{g}(\theta_t)].
\end{aligned}$$

In other words, the smaller the variance of the gradient estimate, the larger the expected reduction in objective value. Some recent work has focused on variance-reduction strategies for SGD when the risk estimates are basic sample means (see for instance Le Roux et al., 2012; Johnson and Zhang, 2013).

In our setting where the risk estimates are of the form of a U-statistic, we are interested in comparing the variance of $\widetilde{g}_{\mathbf{n}'}(\theta)$ and $\widetilde{g}_B(\theta)$ when $B = \prod_{k=1}^{K} \binom{n'_k}{d_k}$ so that their computation requires to average over the same number of terms and thus have similar computational cost.[1] Our result is summarized in the following proposition.

Proposition 14 *Let* $B = \prod_{k=1}^{K} \binom{n'_k}{d_k}$ *for* $n'_k \ll n_k$, $k = 1, \ldots, K$. *In the asymptotic framework* (4)*, we have:*

$$\mathrm{Var}[\widetilde{g}_{\mathbf{n}'}(\theta)] = O\left(\frac{1}{\sum_{k=1}^{K} n'_k}\right), \quad \mathrm{Var}[\widetilde{g}_B(\theta)] = O\left(\frac{1}{\prod_{k=1}^{K} \binom{n'_k}{d_k}}\right),$$

as $n' = n'_1 + \ldots + n'_K \to +\infty$.

1. Note that sampling B sets from Λ to obtain (36) is potentially more efficient than sampling n'_k points from $\mathbf{X}_{\{1,\ldots,n_k\}}$ for each $k = 1, \ldots, K$ and then forming all combinations to obtain (35).

Proposition 14 shows that the convergence rate of $\mathrm{Var}[\widetilde{g}_B(\theta)]$ is faster than that of $\mathrm{Var}[\widetilde{g}_{\mathbf{n}'}(\theta)]$ except when $K = 1$ and $d_1 = 1$. Thus the expected improvement in objective function at each SGD step is larger when using a gradient estimate in the form of (36) instead of (35), although both strategies require to average over the same number of terms. This is also supported by the experimental results reported in the next section.

5. Numerical Experiments

We show the benefits of the sampling approach promoted in this paper on two applications: metric learning for classification, and model selection in clustering.

5.1 Metric Learning

In this section, we focus on the metric learning problem (see Section 2.2.2). As done in much of the metric learning literature, we restrict our attention to the family of pseudo-distance functions $D_M : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ defined as

$$D_M(x, x') = (x - x')M(x - x')^T,$$

where $M \in \mathbb{S}^d_+$, and $\mathbb{S}^d_+$ is the cone of $d \times d$ symmetric positive-semidefinite (PSD) matrices.

Given a training sample $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$ and $y_i \in \{1, \ldots, C\}$, let $y_{ij} = 1$ if $y_i = y_j$ and 0 otherwise for any pair of samples. Given a threshold $b \geq 0$, we define the empirical risk as follows:

$$R_n(D_M) = \frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} [y_{ij}(b - D_M(x_i, x_j))]_+, \tag{37}$$

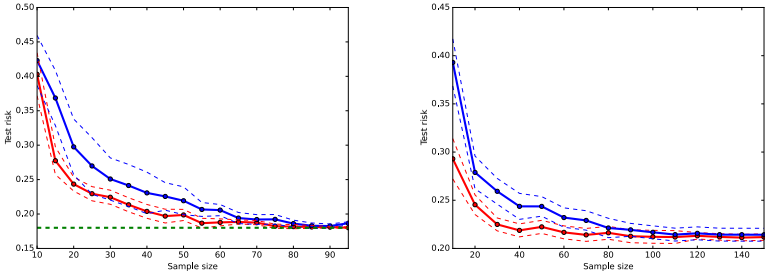
where $[u]_+ = \max(0, 1 - u)$ is the hinge loss. This risk estimate is convex and was used for instance by Jin et al. (2009) and Cao et al. (2012). Our goal is the find the empirical risk minimizer among our family of distance functions, i.e.:

$$\widehat{M} = \arg\min_{M \in \mathbb{S}^d_+} R_n(D_M). \tag{38}$$

In our experiments, we use the following two data sets:

- **Synthetic data set**: some synthetic data that we generated for illustration. X is a mixture of 10 gaussians in $\mathbb{R}^{40}$ – each one corresponding to a class – such that all gaussian means are contained in an subspace of dimension 15 and their shared covariance matrix is proportional to identity with a variance factor such that some overlap is observed. That is, the solution to the metric learning problem should be proportional to the linear projection over the subspace containing the gaussians means. Training and testing sets contain respectively 50,000 and 10,000 observations.
- **MNIST data set**: a handwritten digit classification data set which has 10 classes and consists of 60,000 training images and 10,000 test images.[2] This data set has been used extensively to benchmark metric learning (Weinberger and Saul, 2009). As done by previous authors, we reduce the dimension from 784 to 164 using PCA so as to retain 95% of the variance, and normalize each sample to unit norm.

2. See `http://yann.lecun.com/exdb/mnist/`.



(a) Synthetic data set (b) MNIST data set

Figure 3: Test risk with respect to the sample size p of the ERM when the risk is approximated using complete (blue) or incomplete (red) U-statistics. Solid lines represent means and dashed ones represent standard deviation. For the synthetic data set, the green dotted line represent the performance of the true risk minimizer.

Note that for both data sets, merely computing the empirical risk (37) for a given M involves averaging over more than 10^9 pairs.

We conduct two types of experiment. In Section 5.1.1, we subsample the data before learning and evaluate the performance of the ERM on the subsample. In Section 5.1.2, we use Stochastic Gradient Descent to find the ERM on the original sample, using subsamples at each iteration to estimate the gradient.

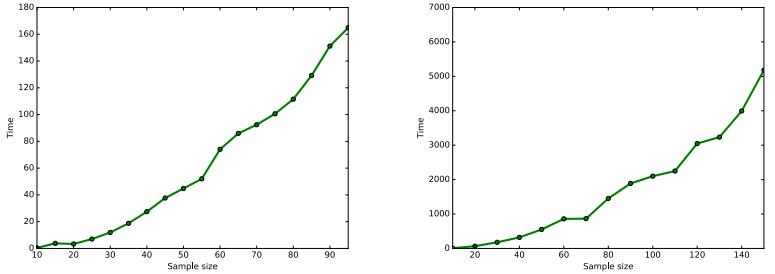
5.1.1 One-Time Sampling

We compare two sampling schemes to approximate the empirical risk:

- Complete U-statistic: p indices are uniformly picked at random in $\{1,\dots,n\}$. The empirical risk is approximated using any possible pair formed by the p indices, that is $\frac{p(p-1)}{2}$ pairs.
- Incomplete U-statistic: the empirical risk is approximated using $\frac{p(p-1)}{2}$ pairs picked uniformly at random in $\{1,\dots,n\}^2$.

For each strategy, we use a projected gradient descent method in order to solve (38), using several values of p and averaging the results over 50 random trials. As the testing sets are large, we evaluate the test risk on 100,000 randomly picked pairs.

Figure 3(a) shows the test risk of the ERM with respect to the sample size p for both sampling strategies on the synthetic data set. As predicted by our theoretical analysis, the incomplete U-statistic strategy achieves a significantly smaller risk on average. For instance, it gets within 5% error of the true risk minimizer for $p = 50$, while the complete U-statistic needs $p > 80$ to reach the same performance. This represents twice more computational time, as shown in Figure 4(a) (as expected, the runtime increases roughly quadratically with



(a) Synthetic data set (b) MNIST data set

Figure 4: Average training time (in seconds) with respect to the sample size p.

p). The incomplete U-statistic strategy also has the advantage of having a much smaller variance between the runs, which makes it more reliable. The same conclusions hold for the MNIST data set, as can be seen in Figure 3(b) and Figure 4(b).

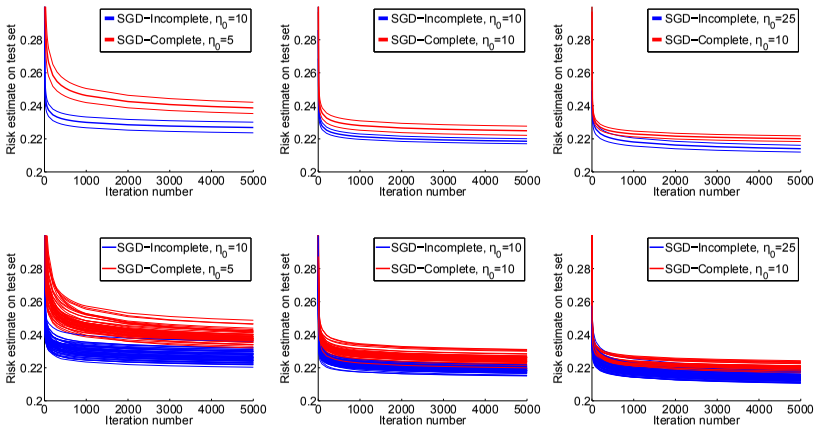
5.1.2 Stochastic Gradient Descent

In this section, we focus on solving the ERM problem (38) using Stochastic Gradient Descent and compare two approaches (analyzed in Section 4) to construct a mini-batch at each iteration. The first strategy, SGD-Complete, is to randomly draw (with replacement) a subsample and use the complete U-statistic associated with the subsample as the gradient estimate. The second strategy, SGD-Incomplete (the one we promote in this paper), consists in sampling an incomplete U-statistic with the same number of terms as in SGD-Complete.

For this experiment, we use the MNIST data set. We set the threshold in (37) to $b = 2$ and the learning rate of SGD at iteration t to $\eta_t = 1/(\eta_0 t)$ where $\eta_0 \in \{1, 2.5, 5, 10, 25, 50\}$. To reduce computational cost, we only project our solution onto the PSD cone at the end of the algorithm, following the "one projection" principle used by Chechik et al. (2010). We try several values m for the mini-batch size, namely $m \in \{10, 28, 55, 105, 253\}$.[3] For each mini-batch size, we run SGD for 10,000 iterations and select the learning rate parameter η_0 that achieves the minimum risk on 100,000 pairs randomly sampled from the training set. We then estimate the generalization risk using 100,000 pairs randomly sampled from the test set.

For all mini-batch sizes, SGD-Incomplete achieves significantly better test risk than SGD-Complete. Detailed results are shown in Figure 5 for three mini-batch sizes, where we plot the evolution of the test risk with respect to the iteration number.[4] We make several comments. First, notice that the best learning rate is often larger for SGD-Incomplete than for SGD-Complete ($m = 10$ and $m = 253$). This confirms that gradient estimates from the

3. For each m, we can construct a complete U-statistic from n' samples with $n'(n'-1)/2 = m$ terms.
4. We point out that the figures look the same if we plot the runtime instead of the iteration number. Indeed, the time spent on computing the gradients (which is the same for both variants) largely dominates the time spent on the random draws.



(a) $m = 10$ (b) $m = 55$ (c) $m = 253$

Figure 5: SGD results on the MNIST data set for various mini-batch size m. The top row shows the means and standard deviations over 50 runs, while the bottom row shows each run separately.

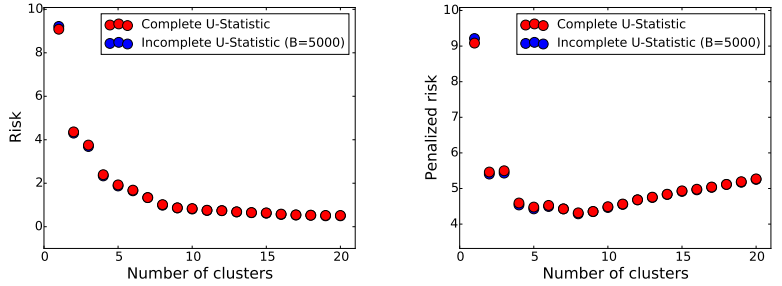
former strategy are generally more reliable. This is further supported by the fact that even though larger learning rates increase the variance of SGD, in these two cases SGD-Complete and SGD-Incomplete have similar variance. On the other hand, for $m = 55$ the learning rate is the same for both strategies. SGD-Incomplete again performs significantly better on average and also has smaller variance. Lastly, as one should expect, the gap between SGD-Complete and SGD-Incomplete reduces as the size of the mini-batch increases. Note however that in practical implementations, the relatively small mini-batch sizes (in the order of a few tens or hundreds) are generally those which achieve the best error/time trade-off.

5.2 Model Selection in Clustering

In this section, we are interested in the clustering problem described in Section 2.2.1. Specifically, let $X_1, \ldots, X_n \in \mathbb{R}^d$ be the set of points to be clustered. Let the clustering risk associated with a partition $\mathcal{P}$ into M groups $\mathcal{C}_1, \ldots, \mathcal{C}_M$ be:

$$\widehat{W}_n(\mathcal{P}) = \frac{2}{n(n-1)} \sum_{m=1}^{M} \sum_{1 \leq i < j \leq n} D(X_i, X_j) \cdot \mathbb{I}\{(X_i, X_j) \in \mathcal{C}_m^2\}. \tag{39}$$

In this experiment, given a set of candidate partitions, we want to perform model selection by picking the partition which minimizes the risk (39) plus some term penalizing the complexity of the partition. When the number of points n is large, the complete risk is very



(a) Risk (b) Penalized risk

Figure 6: Clustering model selection results on the forest cover type data set. Figure 6(a) shows the risk (complete and incomplete with $B = 5,000$ terms) for the first 20 partitions, while Figure 6(b) shows the penalized risk for $c = 1.1$.

expensive to compute. Our strategy is to replace it with an incomplete approximation with much fewer terms. Like in the approach theoretically investigated in Section 3.2, the goal here is to show that using the incomplete approximation instead of the complete version as the goodness-of-fit measure in a complexity penalized criterion does not damage the selection, while reducing the computational cost. For simplicity, the complexity penalty we use below is not of the same type as the structural VC dimension-based penalty considered in Theorem 10, but we will see that the incomplete approximation is very accurate and can thus effectively replace the complete version regardless of the penalty used.

The experimental setup is as follows. We used the forest cover type data set,[5] which is popular to benchmark clustering algorithms (see for instance Kanungo et al., 2004). To be able to evaluate the complete risk, we work with $n = 5,000$ points subsampled at random from the entire data set of 581,012 points in dimension 54. We then generated a hierarchical clustering of these points using agglomerative clustering with Ward's criterion (Ward, 1963) as implemented in the `scikit-learn` Python library (Pedregosa et al., 2011). This defines n partitions $\mathcal{P}_1, \ldots, \mathcal{P}_n$ where $\mathcal{P}_m$ consists of m clusters ($\mathcal{P}_1$ corresponds to a single cluster containing all points, while in $\mathcal{P}_n$ each point has its own cluster).

For each partition size, we first compare the value of the complete risk (39) with $n(n-1) = 24,995,000$ terms with that of an incomplete version with only $B = n = 5,000$ pairs drawn at random. As shown in Figure 6(a), the incomplete U-statistic is a very accurate approximation of the complete one, despite consisting of 5000 times less terms. It will thus lead to similar results in model selection. To illustrate, we use a simple penalty term of the form $\text{pen}(\mathcal{P}_m) = c \cdot \log(m)$ where c is a scaling constant. Figure 6(b) shows that both selection criteria choose the same model $\mathcal{P}_8$. Performing this model selection over $\mathcal{P}_1, \ldots, \mathcal{P}_{20}$

5. See `https://archive.ics.uci.edu/ml/datasets/Covertype`.

took about 66 seconds for the complete U-statistic, compared to only 0.1 seconds for the incomplete version.[6]

Finally, we generated 100 incomplete U-statistics with different random seeds ; all of them correctly identified $\mathcal{P}_8$ as the best model. Using $B = 5,000$ pairs is thus sufficient to obtain reliable results with an incomplete U-statistic for this data set. In contrast, the complete U-statistics based on a subsample (leading to the same number of pairs) selected the correct model in only 57% of cases.

6. Conclusion

In a wide variety of statistical learning problems, U-statistics are natural estimates of the risk measure one seeks to optimize. As the sizes of the samples increase, the computation of such functionals involves summing a rapidly exploding number of terms and becomes numerically unfeasible. In this paper, we argue that for such problems, *Empirical Risk Minimization* can be implemented using statistical counterparts of the risk based on much less terms (picked randomly by means of sampling with replacement), referred to as *incomplete U-statistics*. Using a novel deviation inequality, we have shown that this approximation scheme does not deteriorate the learning rates, even preserving fast rates in certain situations where they are proved to occur. Furthermore, we have extended these results to U-statistics based on different sampling schemes (Bernoulli sampling, sampling without replacement) and shown how such functionals can be used for the purpose of model selection and for implementing ERM iterative procedures based on stochastic gradient descent. Beyond theoretical rate bounds, the efficiency of the approach we promote is illustrated by several numerical experiments.

Acknowledgments

This work is supported by the Chair "Machine Learning for Big Data" of Télécom ParisTech, and was conducted while A. Bellet was affiliated with Télécom ParisTech. The authors are grateful to the reviewers for their careful reading of the paper, which permitted to improve significantly the presentation of the results.

Appendix A. Proof of Proposition 2

Set $N = \min\{\lfloor n_1/d_1 \rfloor, \ldots, \lfloor n_K/d_K \rfloor\}$ and let

$$V_H\left(X_1^{(1)}, \ldots, X_{n_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{n_K}^{(K)}\right) = \frac{1}{N}\Big[H\left(X_1^{(1)}, \ldots, X_{d_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{d_K}^{(K)}\right) + H\left(X_{d_1+1}^{(1)}, \ldots, X_{2d_1}^{(1)}, \ldots, X_{d_K+1}^{(K)}, \ldots, X_{2d_K}^{(K)}\right) + \ldots + H\left(X_{Nd_1-d_1+1}^{(1)}, \ldots, X_{Nd_1}^{(1)}, \ldots, X_{Nd_K-d_K+1}^{(K)}, \ldots, X_{Nd_K}^{(K)}\right)\Big],$$

6. The $n \times n$ distance matrix was precomputed before running the agglomerative clustering algorithm. The associated runtime is thus not taken into account in these timing results.

for any $H \in \mathcal{H}$ Recall that the K-sample U-statistic $U_{\mathbf{n}}(H)$ can be expressed as

$$U_{\mathbf{n}}(H) = \frac{1}{n_1! \cdots n_K!} \sum_{\sigma_1 \in \mathfrak{S}_{n_1}, \ldots, \sigma_K \in \mathfrak{S}_{n_K}} V_H\left(X_{\sigma_1(1)}^{(1)}, \ldots, X_{\sigma_1(n_1)}^{(1)}, \ldots, X_{\sigma_K(1)}^{(K)}, \ldots, X_{\sigma_K(n_K)}^{(K)}\right), \tag{40}$$

where $\mathfrak{S}_m$ denotes the symmetric group of order m for any $m \geq 1$. This representation as an average of sums of N independent terms is known as the (first) Hoeffding's decomposition, see Hoeffding (1948). Then, using Jensen's inequality in particular, one may easily show that, for any nondecreasing convex function $\psi : \mathbb{R}_+ \to \mathbb{R}$, we have:

$$\mathbb{E}\left[\psi\left(\sup_{H \in \mathcal{H}} \left|U_{\mathbf{n}}(\bar{H})\right|\right)\right] \leq \mathbb{E}\left[\psi\left(\sup_{H \in \mathcal{H}} \left|V_{\bar{H}}(X_1^{(1)}, \ldots, X_{n_1}^{(1)}, \ldots, X_1^{(K)}, \ldots, X_{n_K}^{(K)})\right|\right)\right], \tag{41}$$

where we set $\bar{H} = H - \mu(H)$ for all $H \in \mathcal{H}$. Now, using standard symmetrization and randomization arguments (see Giné and Zinn (1984) for instance) and (41), we obtain that

$$\mathbb{E}\left[\psi\left(\sup_{H \in \mathcal{H}} \left|U_{\mathbf{n}}(\bar{H})\right|\right)\right] \leq \mathbb{E}\left[\psi\left(2\mathcal{R}_N\right)\right], \tag{42}$$

where

$$\mathcal{R}_N = \sup_{H \in \mathcal{H}} \frac{1}{N} \sum_{l=1}^{N} \epsilon_l H\left(X_{(l-1)d_1+1}^{(1)}, \ldots, X_{ld_1}^{(1)}, \ldots, X_{(l-1)d_K+1}^{(K)}, \ldots, X_{ld_K}^{(K)}\right),$$

is a Rademacher average based on the Rademacher chaos $\epsilon_1, \ldots, \epsilon_N$ (independent random symmetric sign variables), independent from the $X_i^{(k)}$'s. We now apply the bounded difference inequality (see McDiarmid (1989)) to the functional $\mathcal{R}_N$, seen as a function of the i.i.d. random variables $(\epsilon_l, X_{(l-1)d_1+1}^{(1)}, \ldots, X_{ld_1}^{(1)}, \ldots, X_{(l-1)d_K+1}^{(K)}, \ldots, X_{ld_K}^{(K)})$, $1 \leq l \leq N$: changing any of these random variables change the value of $\mathcal{R}_N$ by at most $\mathcal{M}_{\mathcal{H}}/N$. One thus obtains from (42) with $\psi(x) = \exp(\lambda x)$, where $\lambda > 0$ is a parameter which shall be chosen later, that:

$$\mathbb{E}\left[\exp\left(\lambda \sup_{H \in \mathcal{H}} \left|U_{\mathbf{n}}(\bar{H})\right|\right)\right] \leq \exp\left(2\lambda \mathbb{E}[\mathcal{R}_N] + \frac{\mathcal{M}_{\mathcal{H}}^2 \lambda^2}{4N}\right). \tag{43}$$

Applying Chernoff's method, one then gets:

$$\mathbb{P}\left\{\sup_{H \in \mathcal{H}} \left|U_{\mathbf{n}}(\bar{H})\right| > \eta\right\} \leq \exp\left(-\lambda\eta + 2\lambda \mathbb{E}[\mathcal{R}_N] + \frac{\mathcal{M}_{\mathcal{H}}^2 \lambda^2}{4N}\right). \tag{44}$$

Using the bound (see Eq. (6) in Boucheron et al. (2005) for instance)

$$\mathbb{E}[\mathcal{R}_N] \leq \mathcal{M}_{\mathcal{H}} \sqrt{\frac{2V \log(1+N)}{N}}$$

and taking $\lambda = 2N(\eta - 2\mathbb{E}[\mathcal{R}_N])/\mathcal{M}_{\mathcal{H}}^2$ in (44), one finally establishes the desired result.

Appendix B. Proof of Theorem 6

For convenience, we introduce the random sequence $\zeta = ((\zeta_k(I))_{I\in\Lambda})_{1\leq k\leq B}$, where $\zeta_k(I)$ is equal to 1 if the tuple $I = (I_1, \ldots, I_K)$ has been selected at the k-th draw and to 0 otherwise: the ζ_k's are i.i.d. random vectors and, for all $(k, I) \in \{1, \ldots, B\} \times \Lambda$, the r.v. $\zeta_k(I)$ has a Bernoulli distribution with parameter $1/\#\Lambda$. We also set $\mathbf{X}_I = (\mathbf{X}_{I_1}^{(1)}, \ldots, \mathbf{X}_{I_K}^{(K)})$ for any I in Λ. Equipped with these notations, observe first that one may write: $\forall B \geq 1$, $\forall \mathbf{n} \in \mathbb{N}^{*K}$,

$$\widetilde{U}_B(H) - U_{\mathbf{n}}(H) = \frac{1}{B}\sum_{k=1}^{B} \mathcal{Z}_k(H),$$

where $\mathcal{Z}_k(H) = \sum_{I\in\Lambda}(\zeta_k(I) - 1/\#\Lambda)H(\mathbf{X}_I)$ for any $(k, I) \in \{1, \ldots, B\} \times \Lambda$. It follows from the independence between the $\mathbf{X}_I$'s and the $\zeta(I)$'s that, for all $H \in \mathcal{H}$, conditioned upon the $\mathbf{X}_I$'s, the variables $\mathcal{Z}_1(H), \ldots, \mathcal{Z}_B(H)$ are independent, centered and almost-surely bounded by $2\mathcal{M}_{\mathcal{H}}$ (notice that $\sum_{I\in\Lambda}\zeta_k(I) = 1$ for all $k \geq 1$). By virtue of Sauer's lemma, since $\mathcal{H}$ is a VC major class with finite VC dimension V, we have, for fixed $\mathbf{X}_I$'s:

$$\#\{(H(\mathbf{X}_I))_{I\in\Lambda} : H \in \mathcal{H}\} \leq (1 + \#\Lambda)^V.$$

Hence, conditioned upon the $\mathbf{X}_I$'s, using the union bound and next Hoeffding's inequality applied to the independent sequence $\mathcal{Z}_1(H), \ldots, \mathcal{Z}_B(H)$, for all $\eta > 0$, we obtain that:

$$\begin{aligned}\mathbb{P}\left\{\sup_{H\in\mathcal{H}}\left|\widetilde{U}_B(H) - U_{\mathbf{n}}(H)\right| > \eta \mid (\mathbf{X}_I)_{I\in\Lambda}\right\} &\leq \mathbb{P}\left\{\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\mathcal{Z}_k(H)\right| > \eta \mid (\mathbf{X}_I)_{I\in\Lambda}\right\} \\ &\leq 2(1 + \#\Lambda)^V e^{-B\eta^2/(2\mathcal{M}_{\mathcal{H}}^2)}.\end{aligned}$$

Taking the expectation, this proves the first assertion of the theorem. Notice that this can be formulated: for any $\delta \in (0, 1)$, we have with probability at least $1 - \delta$:

$$\sup_{H\in\mathcal{H}}\left|\widetilde{U}_B(H) - U_{\mathbf{n}}(H)\right| \leq \mathcal{M}_{\mathcal{H}} \times \sqrt{2\frac{V\log(1 + \#\Lambda) + \log(2/\delta)}{B}}.$$

Turning to the second part of the theorem, it straightforwardly results from the first part combined with Proposition 2.

Appendix C. Proof of Corollary 9

Assertion (i) is a direct application of Assertion (ii) in Theorem 6 combined with the bound $\mu(\widehat{H}_B) - \inf_{H\in\mathcal{H}}\mu(H) \leq 2\sup_{H\in\mathcal{H}}|\widetilde{U}_B(H) - \mu(H)|$.

Turning next to Assertion (ii), observe that by triangle inequality we have:

$$\mathbb{E}\left[\sup_{H\in\mathcal{H}_m}|\widetilde{U}_B(H) - \mu(H)|\right] \leq \mathbb{E}\left[\sup_{H\in\mathcal{H}_m}|\widetilde{U}_B(H) - U_{\mathbf{n}}(H)|\right] + \mathbb{E}\left[\sup_{H\in\mathcal{H}_m}|U_{\mathbf{n}}(H) - \mu(H)|\right]. \quad (45)$$

The same argument as that used in Theorem 6 (with $\psi(u) = u$ for any $u \geq 0$) yields a bound for the second term on the right hand side of Eq. (45):

$$\mathbb{E}\left[\sup_{H\in\mathcal{H}}|U_{\mathbf{n}}(H) - \mu(H)|\right] \leq 2\mathcal{M}_{\mathcal{H}}\sqrt{\frac{2V\log(1 + N)}{N}}. \quad (46)$$

The first term can be controlled by means of the following lemma, whose proof can be found for instance in Lugosi (2002, Lemmas 1.2 and 1.3).

Lemma 15 *The following assertions hold true.*

(i) Hoeffding's lemma. Let Z be an integrable r.v. with mean zero such that $a \leq Z \leq b$ almost-surely. Then, we have: $\forall s > 0$

$$\mathbb{E}[\exp(sZ)] \leq \exp\left(s^2(b - a)^2/8\right).$$

(ii) Let $M \geq 1$ and $Z_1, \ldots, Z_M$ be real valued random variables. Suppose that there exists $\sigma > 0$ such that $\forall s \in \mathbb{R}$: $\mathbb{E}[\exp(sZ_i)] \leq e^{s^2\sigma^2/2}$ for all $i \in \{1, \ldots, M\}$. Then, we have:

$$\mathbb{E}\left[\max_{1\leq i\leq M}|Z_i|\right] \leq \sigma\sqrt{2\log(2M)}. \quad (47)$$

Assertion (i) shows that, since $-\mathcal{M}_{\mathcal{H}} \leq \mathcal{Z}_k(H) \leq \mathcal{M}_{\mathcal{H}}$ almost surely,

$$\mathbb{E}\left[\exp(s\sum_{k=1}^{B}\mathcal{Z}_k(H)) \mid (\mathbf{X}_I)_{I\in\Lambda}\right] \leq e^{\frac{1}{2}Bs^2\mathcal{M}_{\mathcal{H}}^2}.$$

With $\sigma = \mathcal{M}_{\mathcal{H}}\sqrt{B}$ and $M = \#\{H(\mathbf{X}_I) : H \in \mathcal{H}\} \leq (1 + \#\Lambda)^V$, conditioning upon $(\mathbf{X}_I)_{I\in\Lambda}$, this result yields:

$$\mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\mathcal{Z}_k(H)\right| \mid (\mathbf{X}_I)_{I\in\Lambda}\right] \leq \mathcal{M}_{\mathcal{H}}\sqrt{\frac{2(\log 2 + V\log(1 + \#\Lambda))}{B}}. \quad (48)$$

Integrating next over $(\mathbf{X}_I)_{I\in\Lambda}$ and combining the resulting bound with (45) and (46) leads to the inequality stated in (ii).

A bound for the expected value. For completeness, we point out that the expected value of $\sup_{H\in\mathcal{H}}|(1/B)\sum_{k=1}^{B}\mathcal{Z}_k(H)|$ can also be bounded by means of classical symmetrization and randomization devices. Considering a "ghost" i.i.d. sample $\zeta_1', \ldots, \zeta_B'$ independent from $((\mathbf{X}_I)_{I\in\Lambda}, \zeta)$, distributed as ζ, Jensen's inequality yields:

$$\begin{aligned}\mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\mathcal{Z}_k(H)\right|\right] &= \mathbb{E}\left[\sup_{H\in\mathcal{H}}\left\{\mathbb{E}\left[\left|\frac{1}{B}\sum_{k=1}^{B}\sum_{I\in\Lambda}H(\mathbf{X}_I)\left(\zeta_k(I) - \zeta_k'(I)\right)\right| \mid (\mathbf{X}_I)_{I\in\Lambda}\right]\right\}\right] \\ &\leq \mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\sum_{I\in\Lambda}H(\mathbf{X}_I)\left(\zeta_k(I) - \zeta_k'(I)\right)\right|\right].\end{aligned}$$

Introducing next independent Rademacher variables $\epsilon_1, \ldots, \epsilon_B$, independent from $((\mathbf{X}_I)_{I\in\Lambda}, \zeta, \zeta')$, we have:

$$\mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\sum_{I\in\Lambda} H(\mathbf{X}_I)\left(\zeta_k(I)-\zeta'_k(I)\right)\right| \mid (\mathbf{X}_I)_{I\in\Lambda}\right] = \mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\epsilon_k\sum_{I\in\Lambda} H(\mathbf{X}_I)\left(\zeta_k(I)-\zeta'_k(I)\right)\right| \mid (\mathbf{X}_I)_{I\in\Lambda}\right] \leq 2\mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\epsilon_k\sum_{I\in\Lambda} H(\mathbf{X}_I)\zeta_k(I)\right| \mid (\mathbf{X}_I)_{I\in\Lambda}\right].$$

We thus obtained:

$$\mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\mathcal{Z}_k(H)\right|\right] \leq 2\mathbb{E}\left[\sup_{H\in\mathcal{H}}\left|\frac{1}{B}\sum_{k=1}^{B}\epsilon_k\sum_{I\in\Lambda} H(\mathbf{X}_I)\zeta_k(I)\right|\right].$$

Appendix D. Proof of Theorem 10

We start with proving the intermediary result, stated below.

Lemma 16 *Under the assumptions stipulated in Theorem 10, we have:* $\forall m \geq 1$, $\forall \epsilon > 0$,

$$\mathbb{P}\left\{\sup_{H\in\mathcal{H}_m}|\mu(H)-\widetilde{U}_B(H)| > 2\mathcal{M}_{\mathcal{H}_m}\left\{\sqrt{\frac{2V_m\log(1+N)}{N}}+\sqrt{\frac{2(\log 2+V_m\log(1+\#\Lambda))}{B}}\right\}+\epsilon\right\} \leq \exp\left(-B^2\epsilon^2/\left(2(B+n)\mathcal{M}^2_{\mathcal{H}_m}\right)\right).$$

Proof This is a direct application of the bounded difference inequality (see McDiarmid (1989)) applied to the quantity $\sup_{H\in\mathcal{H}_m}|\mu(H)-\widetilde{U}_B(H)|$, viewed as a function of the $(B+n)$ independent random variables $(X_1^{(1)}, X_{n_K}^{(K)}, \epsilon_1, \ldots, \epsilon_B)$ (jumps being bounded by $2\mathcal{M}_H/B$), combined with Assertion (ii) of Corollary 9. ∎

Let $m \geq 1$ and decompose the expected excess of risk of the rule picked by means of the complexity regularized incomplete U-statistic criterion as follows:

$$\mathbb{E}\left[\mu(\widehat{H}_{B,\widehat{m}})-\mu^*_m\right] = \mathbb{E}\left[\mu(\widehat{H}_{B,\widehat{m}})-\widetilde{U}_B(\widehat{H}_{B,\widehat{m}})-\mathrm{pen}(B,\widehat{m})\right] + \mathbb{E}\left[\inf_{j\geq 1}\left\{\widetilde{U}_B(\widehat{H}_{B,j})+\mathrm{pen}(B,j)\right\}-\mu^*_m\right],$$

where we set $\mu^*_m = \inf_{H\in\mathcal{H}_m}\mu(H)$. In order to bound the first term on the right hand side of the equation above, observe that we have: $\forall \epsilon > 0$,

$$\begin{aligned}\mathbb{P}\left\{\mu(\widehat{H}_{B,\widehat{m}})-\widetilde{U}_B(\widehat{H}_{B,\widehat{m}})-\mathrm{pen}(B,\widehat{m})>\epsilon\right\} &\leq \mathbb{P}\left\{\sup_{j\geq 1}\left\{\mu(\widehat{H}_{B,j})-\widetilde{U}_B(\widehat{H}_{B,j})-\mathrm{pen}(B,j)\right\}>\epsilon\right\}\\ &\leq \sum_{j\geq 1}\mathbb{P}\left\{\mu(\widehat{H}_{B,j})-\widetilde{U}_B(\widehat{H}_{B,j})-\mathrm{pen}(B,j)>\epsilon\right\}\\ &\leq \sum_{j\geq 1}\mathbb{P}\left\{\sup_{H\in\mathcal{H}_j}|\mu(\widehat{H})-\widetilde{U}_B(H)|-\mathrm{pen}(B,j)>\epsilon\right\}\\ &\leq \sum_{j\geq 1}\exp\left(-\frac{-B^2}{2(B+n)\mathcal{M}^2}\left(\epsilon+2\mathcal{M}\sqrt{\frac{(B+n)\log j}{B^2}}\right)^2\right)\\ &\leq \exp\left(-\frac{B^2\epsilon^2}{2(B+n)\mathcal{M}^2}\right)\sum_{j\geq 1}1/j^2 \leq 2\exp\left(-\frac{B^2\epsilon^2}{2(B+n)\mathcal{M}^2}\right),\end{aligned}$$

using successively the union bound and Lemma 16. Integrating over $[0,+\infty)$, we obtain that:

$$\mathbb{E}\left[\mu(\widehat{H}_{B,\widehat{m}})-\widetilde{U}_B(\widehat{H}_{B,\widehat{m}})-\mathrm{pen}(B,\widehat{m})\right] \leq \mathcal{M}\frac{\sqrt{2\pi(B+n)}}{B}. \tag{49}$$

Considering now the second term, notice that

$$\mathbb{E}\left[\inf_{j\geq 1}\left\{\widetilde{U}_B(\widehat{H}_{B,j})+\mathrm{pen}(B,j)\right\}-\mu^*_m\right] \leq \mathbb{E}\left[\widetilde{U}_B(\widehat{H}_{B,m})+\mathrm{pen}(B,m)-\mu^*_m\right] \leq \mathrm{pen}(B,m).$$

Combining the bounds, we obtain that: $\forall m \geq 1$,

$$\mathbb{E}\left[\mu(\widehat{H}_{B,\widehat{m}})\right] \leq \mu^*_m + \mathrm{pen}(B,m) + \mathcal{M}\frac{\sqrt{2\pi(B+n)}}{B}.$$

The oracle inequality is thus proved.

Appendix E. Proof of Theorem 12

We start with proving the following intermediary result, based on the U-statistic version of the Bernstein exponential inequality.

Lemma 17 *Suppose that the assumptions of Theorem 12 are fulfilled. Then, for all* $\delta \in (0,1)$*, we have with probability larger than* $1-\delta$*:* $\forall r \in \mathcal{R}$, $\forall n \geq 2$,

$$0 \leq \Lambda_n(r) - \Lambda(r) + \sqrt{\frac{2c\Lambda(r)^\alpha\log(\#\mathcal{R}/\delta)}{n}} + \frac{4\log(\#\mathcal{R}/\delta)}{3n}.$$

Proof The proof is a straightforward application of Theorem A on p. 201 in Serfling (1980), combined with the union bound and Assumption 1. ∎

The same argument as that used to prove Assertion (i) in Theorem 6 (namely, freezing the

$\mathbf{X}_I$'s, applying Hoeffding inequality and the union bound) shows that, for all $\delta \in (0,1)$, we have with probability at least $1-\delta$: $\forall r \in \mathcal{R}$,

$$0 \leq \widetilde{U}_B(q_r) - U_{\mathbf{n}}(q_r) + \sqrt{\frac{M + \log(M/\delta)}{B}}$$

for all $n \geq 2$ and $B \geq 1$ (observe that $\mathcal{M}_{\mathcal{H}} \leq 1$ in this case). Now, combining this bound with the previous one and using the union bound, one gets that, for all $\delta \in (0,1)$, we have with probability larger than $1-\delta$: $\forall r \in \mathcal{R}$, $\forall n \geq 2$, $\forall B \geq 1$,

$$0 \leq \widetilde{U}_B(q_r) - \Lambda(r) + \sqrt{\frac{2c\Lambda(r)^\alpha \log(2M/\delta)}{n}} + \frac{4\log(2M/\delta)}{3n} + \sqrt{\frac{M + \log(2M/\delta)}{B}}.$$

Observing that, $\widetilde{U}_B(q_{\widetilde{r}_B}) \leq 0$ by definition, we thus have with probability at least $1-\delta$:

$$\Lambda(\widetilde{r}_B) \leq \sqrt{\frac{2c\Lambda(\widetilde{r}_B)^\alpha \log(2M/\delta)}{n}} + \frac{4\log(2M/\delta)}{3n} + \sqrt{\frac{M + \log(2M/\delta)}{B}}.$$

Choosing finally $B = O(n^{2/(2-\alpha)})$, the desired result is obtained by solving the inequality above for $\Lambda(\widetilde{r}_B)$.

Appendix F. Proof of Theorem 13

As shown by the following lemma, which is a slight modification of Lemma 1 in Janson (1984), the deviation between the incomplete U-statistic and its complete version is of order $O_{\mathbb{P}}(1/\sqrt{B})$ for both sampling schemes.

Lemma 18 *Suppose that the assumptions of 13 are fulfilled. Then, we have:* $\forall H \in \mathcal{H}$,

$$\mathbb{E}\left[\left(\bar{U}_{HT}(H) - U_{\mathbf{n}}(H)\right)^2 \mid (\mathbf{X}_I)_{I\in\Lambda}\right] \leq 2\mathcal{M}_{\mathcal{H}}^2/B.$$

Proof Observe first that, in both cases (sampling without replacement and Bernoulli sampling), we have: $\forall I \neq J$ in Λ,

$$\mathbb{E}\left[\left(\Delta(I) - \frac{B}{\#\Lambda}\right)^2\right] \leq \frac{B}{\#\Lambda} \text{ and } \mathbb{E}\left[\left(\Delta(I) - \frac{B}{\#\Lambda}\right)\left(\Delta(J) - \frac{B}{\#\Lambda}\right)\right] \leq \frac{1}{\#\Lambda}\cdot\frac{B}{\#\Lambda}.$$

Hence, as $(\Delta(I))_{I\in\Lambda}$ and $(\mathbf{X}_I)_{I\in\Lambda}$ are independent by assumption, we have:

$$B^2\mathbb{E}\left[\left(\bar{U}_{HT}(H) - U_{\mathbf{n}(H)}\right)^2 \mid (\mathbf{X}_I)_{I\in\Lambda}\right] = \mathbb{E}\left[\left(\sum_{I\in\Lambda}\left(\Delta(I) - \frac{B}{\#\Lambda}\right)H(\mathbf{X}_I)\right)^2 \mid (\mathbf{X}_I)_{I\in\Lambda}\right]$$
$$\leq \mathcal{M}_{\mathcal{H}}^2 \sum_{I\in\Lambda}\mathbb{E}\left[\left(\Delta(I) - \frac{B}{\#\Lambda}\right)^2\right] + \mathcal{M}_{\mathcal{H}}^2 \sum_{I\neq J}\mathbb{E}\left[\left(\Delta(I) - \frac{B}{\#\Lambda}\right)\left(\Delta(J) - \frac{B}{\#\Lambda}\right)\right] \leq 2B\mathcal{M}_{\mathcal{H}}^2.$$

■

Consider first the case of Bernoulli sampling. By virtue of Bernstein inequality applied to the independent variables $(\Delta(I) - B/\#\Lambda)H(\mathbf{X}_I)$ conditioned upon $(\mathbf{X}_I)_{I\in\Lambda}$, we have: $\forall H \in \mathcal{H}$, $\forall t > 0$,

$$\mathbb{P}\left\{\left|\sum_{I\in\Lambda}(\Delta(I) - B/\#\Lambda)H(\mathbf{X}_I)\right| > t \mid (\mathbf{X}_I)_{I\in\Lambda}\right\} \leq 2\exp\left(-\frac{t^2}{4B\mathcal{M}_{\mathcal{H}}^2 + 2\mathcal{M}_{\mathcal{H}}t/3}\right).$$

Hence, combining this bound and the union bound, we obtain that: $\forall t > 0$,

$$\mathbb{P}\left\{\sup_{H\in\mathcal{H}}\left|\bar{U}_{HT}(H) - U_{\mathbf{n}(H)}\right| > t \mid (\mathbf{X}_I)_{I\in\Lambda}\right\} \leq 2(1+\#\Lambda)^V \exp\left(-\frac{Bt^2}{4\mathcal{M}_{\mathcal{H}}^2 + 2\mathcal{M}_{\mathcal{H}}t/3}\right).$$

Solving

$$\delta = 2(1+\#\Lambda)^V \exp\left(-\frac{Bt^2}{4\mathcal{M}_{\mathcal{H}}^2 + 2\mathcal{M}_{\mathcal{H}}t/3}\right)$$

yields the desired bound.

Consider next the case of the sampling without replacement scheme. Using the exponential inequality tailored to this situation proved in Serfling (1974) (see Corollary 1.1 therein), we obtain: $\forall H \in \mathcal{H}$, $\forall t > 0$,

$$\mathbb{P}\left\{\frac{1}{B}\left|\sum_{I\in\Lambda}(\Delta(I) - B/\#\Lambda)H(\mathbf{X}_I)\right| > t \mid (\mathbf{X}_I)_{I\in\Lambda}\right\} \leq 2\exp\left(-\frac{Bt^2}{2\mathcal{M}_{\mathcal{H}}^2}\right).$$

The proof can be then ended using the union bound, just like above.

Appendix G. Proof of Proposition 14

For simplicity, we focus on one sample U-statistics of degree two ($K = 1$, $d_1 = 2$) since the argument easily extends to the general case. Let $U_n(H)$ be a non-degenerate U-statistic of degree two:

$$U_n(H) = \frac{2}{n(n-1)}\sum_{i<j} H(x_i, x_j).$$

In order to express the variance of $U_n(H)$ based on its second Hoeffding decomposition (see Section 2.1), we first introduce more notations: $\forall (x, x') \in \mathcal{X}_1^2$,

$$H_1(x) \stackrel{\text{def}}{=} \mathbb{E}[H(x,X)] - \mu(H) \text{ and } H_2(x,x') \stackrel{\text{def}}{=} H(x,x') - \mu(H) - H_1(x) - H_1(x').$$

Equipped with these notations, the (orthogonal) Hoeffding/Hajek decomposition of $U_n(H)$ can be written as

$$U_n(H) = \mu(H) + 2T_n(H) + W_n(H),$$

involving centered and decorrelated random variables given by

$$T_n(H) = \frac{1}{n}\sum_{i=1}^n H_1(x_i),$$
$$W_n(H) = \frac{2}{n(n-1)}\sum_{i<j} H_2(x_i, x_j).$$

Recall that the U-statistic $W_n(H)$ is said to be degenerate, since $\mathbb{E}[H_2(x,X)] = 0$ for all $x \in \mathcal{X}_1$. Based on this representation and setting $\sigma_1^2 = \mathrm{Var}[H_1(X)]$ and $\sigma_2^2 = \mathrm{Var}[H_2(X,X')]$, the variance of $U_n(H)$ is given by

$$\mathrm{Var}[U_n(H)] = \frac{4\sigma_1^2}{n} + \frac{2\sigma_2^2}{n(n-1)}. \tag{50}$$

As already pointed out in Section 3.1, the variance of the incomplete U-statistic built by sampling with replacement is

$$\begin{aligned} \mathrm{Var}[\widetilde{U}_B(H)] &= \mathrm{Var}[U_n(H)] + \frac{1}{B}\left(1 - \frac{2}{n(n-1)}\right)\mathrm{Var}[H(X,X')] \\ &= \mathrm{Var}[U_n(H)] + \frac{1}{B}\left(1 - \frac{2}{n(n-1)}\right)(2\sigma_1^2 + \sigma_2^2). \end{aligned} \tag{51}$$

Take $B = n'(n'-1)$ for $n' \ll n$. It follows from (50) and (51) that in the asymptotic framework (4), the quantities $\mathrm{Var}[U_{n'}(H)]$ and $\mathrm{Var}[\widetilde{U}_B(H)]$ are of the order $O(1/n')$ and $O(1/n'^2)$ respectively as $n' \to +\infty$. Hence these convergence rates hold for $\widetilde{g}_{n'}(\theta)$ and $\widetilde{g}_B(\theta)$ respectively.

References

R. Bekkerman, M. Bilenko, and J. Langford. *Scaling Up Machine Learning*. Cambridge, 2011.

A. Bellet and A. Habrard. Robustness and generalization for metric learning. *Neurocomputing*, 151(1):259–267, 2015.

A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *ArXiv e-prints*, June 2013.

P. Bertail and J. Tressou. Incomplete generalized U-statistics for food risk assessment. *Biometrics*, 62(1):66–74, 2006.

P. Bianchi, S. Clémençon, J. Jakubowicz, and G. Moral-Adell. On-line learning gossip algorithm in multi-agent systems with local decision rules. In *Proceedings of the IEEE International Conference on Big Data*, 2013.

G. Blom. Some properties of incomplete U-statistics. *Biometrika*, 63(3):573–580, 1976.

L. Bottou. *Online Algorithms and Stochastic Approximations: Online Learning and Neural Networks*. Cambridge University Press, 1998.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

B. M. Brown and D. G. Kildea. Reduced U-statistics and the Hodges-Lehmann estimator. *The Annals of Statistics*, 6:828–835, 1978.

Q. Cao, Z.-C. Guo, and Y. Ying. Generalization bounds for metric and similarity learning. Technical report, University of Exeter, July 2012. arXiv:1207.5437.

G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11:1109–1135, 2010.

S. Clémençon. A statistical view of clustering performance through the theory of U-processes. *Journal of Multivariate Analysis*, 124:42–56, 2014.

S. Clémençon and S. Robbiano. Building confidence regions for the ROC surface. *To appear in Pattern Recognition Letters*, 2014.

S. Clémençon and N. Vayatis. Tree-based ranking methods. *IEEE Transactions on Information Theory*, 55(9):4316–4336, 2009.

S. Clémençon, G. Lugosi, and N. Vayatis. Ranking and scoring using empirical risk minimization. In *Proceedings of COLT*, 2005.

S. Clémençon, G. Lugosi, and N. Vayatis. Ranking and empirical risk minimization of U-statistics. *The Annals of Statistics*, 36(2):844–874, 2008.

S. Clémençon, S. Robbiano, and N. Vayatis. Ranking data with ordinal labels: optimality and pairwise aggregation. *Machine Learning*, 91(1):67–104, 2013.

W. G. Cochran. *Sampling techniques*. Wiley, NY, 1977.

V. de la Peña and E. Giné. *Decoupling: from Dependence to Independence*. Springer, 1999.

J. C. Deville. *Réplications d'échantillons, demi-échantillons, Jackknife, bootstrap dans les sondages*. Economica, Ed. Droesbeke, Tassi, Fichet, 1987.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

E. Enqvist. *On sampling from sets of random variables with application to incomplete U-statistics*. PhD thesis, 1978.

J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning*. Springer, 2009.

D. K. Fuk and S. V. Nagaev. Probability inequalities for sums of independent random variables. *Prob. Th. Appl.*, 16(4):643660, 1971.

E. Giné and J. Zinn. Some limit theorems for empirical processes. *The Annals of Probability*, 12(4):929–989, 1984.

W. Grams and R. Serfling. Convergence rates for U-statistics and related statistics. *Ann. Stat.*, 1(1):153–160, 1973.

J. Hájek. Asymptotic theory of rejective sampling with varying probabilities from a finite population. *The Annals of Mathematical Statistics*, 35(4):1491–1523, 1964.

J. Hájek. Asymptotic normality of simple linear rank statistics under alternatives. *Ann. Math. Stat.*, 39:325–346, 1968.

W. Hoeffding. A class of statistics with asymptotically normal distribution. *Ann. Math. Stat.*, 19:293–325, 1948.

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *JASA*, 47:663–685, 1951.

S. Janson. The asymptotic distributions of incomplete U-statistics. *Z. Wahrsch. verw. Gebiete*, 66:495–505, 1984.

R. Jin, S. Wang, and Y. Zhou. Regularized distance metric learning: theory and algorithm. In *Advances in Neural Information Processing Systems 22*, pages 862–870, 2009.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, pages 315–323, 2013.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Computational Geometry*, 28(2–3):89–112, 2004.

N. Le Roux, M. W. Schmidt, and F. Bach. A Stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems 25*, pages 2672–2680, 2012.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces*. Springer, New York, 1991.

A. J. Lee. *U-statistics: Theory and Practice*. Marcel Dekker, Inc., New York, 1990.

G. Lugosi. Pattern classification and learning theory. In L. Györfi, editor, *Principles of Nonparametric Learning*, pages 1–56. Springer, NY, 2002.

P. Massart. *Concentration Inequalities and Model Selection*. Lecture Notes in Mathematics. Springer, 2006.

C. McDiarmid. *On the method of bounded differences*, pages 148–188. Cambridge Univ. Press, 1989.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, and David Cournapeau. Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

R. J. Serfling. Probability inequalities for the sum in sampling without replacement. *The Annals of Statistics*, 2(1):39–48, 1974.

R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, 1980.

Y. Tillé. *Sampling Algorithms*. Springer Series in Statistics, 2006.

V. N. Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999, 1999.

Joe H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.

K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

Journal of Machine Learning Research 17 (2016) 1-38 Submitted 3/15; Revised 3/16; Published 5/16

Iterative Regularization for Learning with Convex Loss Functions

Junhong Lin JHLIN5@HOTMAIL.COM
Department of Mathematics
City University of Hong Kong
Kowloon, Hong Kong, China

Lorenzo Rosasco LROSASCO@MIT.EDU
DIBRIS, Università di Genova
Via Dodecaneso, 35 — 16146 Genova, Italy
Laboratory for Computational and Statistical Learning
Istituto Italiano di Tecnologia and Massachusetts Institute of Technology
Bldg. 46-5155, 77 Massachusetts Avenue, Cambridge, MA 02139, USA

Ding-Xuan Zhou MAZHOU@CITYU.EDU.HK
Department of Mathematics
City University of Hong Kong
Kowloon, Hong Kong, China

Editor: Leon Bottou

Abstract

We consider the problem of supervised learning with convex loss functions and propose a new form of iterative regularization based on the subgradient method. Unlike other regularization approaches, in iterative regularization no constraint or penalization is considered, and generalization is achieved by (early) stopping an empirical iteration. We consider a nonparametric setting, in the framework of reproducing kernel Hilbert spaces, and prove consistency and finite sample bounds on the excess risk under general regularity conditions. Our study provides a new class of efficient regularized learning algorithms and gives insights on the interplay between statistics and optimization in machine learning.

1. Introduction

Availability of large high-dimensional data sets has motivated an interest in the interplay between statistics and optimization, towards developing new and more efficient learning solutions (Bousquet and Bottou, 2008). Indeed, while much theoretical work has been classically devoted to study statistical properties of estimators defined by variational schemes (for example Empirical Risk Minimization (Vapnik, 1998) or Tikhonov regularization (Tikhonov and Arsenin, 1977)), and to the computational properties of optimization procedures to solve the corresponding minimization problems (see for example Sra et al., 2011), much less work has considered the integration of statistical and optimization aspects, see for example Chandrasekaran and Jordan (2013); Wainwright (2014); Orabona (2014).

With the latter objective in mind, in this paper, we focus on iterative regularization. This class of methods, originated in a series of work in the mid-eighties (Nemirovskii, 1986; Polyak, 1987), is based on the observation that early termination of an iterative optimization scheme applied to empirical data has a regularization effect. A critical implication of this fact is that the number of iterations serves as a regularization parameter, hence linking modeling and computational aspects: computational resources are directly linked to the generalization properties in the data, rather than their raw amount. Further, iterative regularization algorithms have a built-in "warm restart" property which allows to compute automatically a whole sequence of solutions corresponding to different levels of regularization (the regularization path). This latter property is especially relevant to efficiently determine the appropriate regularization level via model selection.

Iterative regularization techniques are well known in solving inverse problems, where several variants have been studied, see Engl et al. (1996); Kaltenbacher et al. (2008) and references therein. In machine learning, iterative regularization is often simply referred to as early stopping and is a well known "trick", for example in training neural networks (LeCun et al., 1998). Theoretical studies of iterative regularization in machine learning have mostly focused on the least square loss function (Buhlmann and Yu, 2003; Yao et al., 2007; Bauer et al., 2007; Blanchard and Nicole, 2010; Raskutti et al., 2014). Indeed, it is in this latter case that the connection to inverse problems can be made precise (De Vito et al., 2005). Interestingly, early stopping with the square loss has been shown to be related to boosting (Buhlmann and Yu, 2003) and also to be a special case of a large class of regularization approaches based on spectral filtering (Gerfo et al., 2008; Bauer et al., 2007). The regularizing effect of early stopping for loss functions other than the square loss has hardly been studied. Indeed, to the best of our knowledge the only papers considering related ideas are Bartlett and Traskin (2007); Bickel et al. (2006); Jiang (2004); Zhang and Yu (2005), where early stopping is studied in the context of boosting algorithms.

This paper is a different step towards understanding how early stopping can be employed with general convex loss functions. Within a statistical learning setting, we consider convex loss functions and propose a new form of iterative regularization based on the subgradient method, or the gradient descent if the loss is smooth. The resulting algorithms provide iterative regularization alternatives to Support Vector Machines or regularized logistic regression, and have built in the property of computing the whole regularization path. Our primary contribution in this paper is theoretical. By integrating optimization and statistical results, we establish consistency and non-asymptotic bounds quantifying the generalization properties of the proposed method under standard regularity assumptions. Interestingly, our study shows that considering the last iterate leads to essentially the same results as considering averaging, or selecting of the "best" iterate, as typically done in subgradient methods (Boyd and Vandenberghe, 2004). From a technical point of view, considering a general convex loss requires different error decompositions than the square loss. Moreover, operator theoretic techniques and matrix concentration inequalities need to be replaced by convex analysis and empirical process results. The error decomposition we consider, accounts for the contribution of both optimization and statistics to the error, and could be useful also for other methods.

The rest of the paper is organized as follows. We begin in Section 2 by briefly recalling the supervised learning problem, and then introduce our learning algorithm, discussing its numerical realization. In Section 3, after discussing the assumptions that underlie our analysis, we present and discuss our main theorems and illustrate the general error decom-

position which is composed of three error terms: computational, sample and approximation error. In Section 4, we will estimate the computational error, while in Section 5, we develop sample error bounds, and finally prove our main results.

2. Learning Algorithm

After briefly recalling the supervised learning problem, we introduce the algorithm we propose and give some comments on its numerical realization.

2.1 Problem Statement

In this paper we consider the problem of supervised learning. Let X be a separable metric space, $Y \subseteq \mathbb{R}$ and let ρ be a Borel probability measure on $Z = X \times Y$. Moreover, let $V : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ be a so-called loss function, measuring the *local error* $V(y, f(x))$ for $(x, y) \in Z$ and $f : X \to \mathbb{R}$. The *generalization error (or expected risk)* $\mathcal{E} = \mathcal{E}^V$ associated with V is given by

$$\mathcal{E}(f) = \int_Z V(y, f(x)) d\rho,$$

and is well defined for any measurable loss function V and measurable function f. We assume throughout that there exists a function f_ρ^V that minimizes the expected error $\mathcal{E}(f)$ among all measurable functions $f : X \to Y$. Roughly speaking, the goal of learning is to find an approximation of f_ρ^V when the measure ρ is known only through a sample $\mathbf{z} = \{z_i = (x_i, y_i)\}_{i=1}^m$ of size $m \in \mathbb{N}$ independently and identically drawn according to ρ. More precisely, given $\mathbf{z}$ the goal is to design a computational procedure to efficiently estimate a function $f_\mathbf{z}$, an estimator, for which it is possible to derive an explicitly probabilistic bound on the excess expected risk

$$\mathcal{E}(f_\mathbf{z}) - \mathcal{E}(f_\rho^V).$$

We end this section with a remark and an example.

Remark 1 *For several loss functions, it is possible to show that f_ρ^V exists, see the example below. However, as will be seen in the following, the search for an estimator in practice is often restricted to some hypothesis space $\mathcal{H}$ of measurable functions. In this case one should replace $\mathcal{E}(f_\rho^V)$ by $\inf_{f\in\mathcal{H}} \mathcal{E}(f)$. Interestingly, examples of hypothesis spaces are known for which $\mathcal{E}(f_\rho^V) = \inf_{f\in\mathcal{H}} \mathcal{E}(f)$, namely universal hypothesis spaces (Steinwart and Christmann, 2008). In the following, we consider $\mathcal{E}(f_\rho^V)$, with the understanding that it should be replaced by the infimum over $\mathcal{H}$, if the latter is not universal.*

Example 1 *The most classical example of loss function is probably the square loss $V(y, a) = (y - a)^2$, $y, a \in \mathbb{R}$. In this case, f_ρ^V is the regression function, defined at every point as the expectation of the conditional distribution of y given x (Cucker and Zhou, 2007; Steinwart and Christmann, 2008). Further examples include the absolute value loss $V(y, a) = |y - a|$ for which f_ρ^V is the median of the conditional distribution and more generally p-loss functions $V(y, a) = |y - a|^p$, $p \in \mathbb{N}$ [1]. Vapnik's ϵ-insensitive loss $V(y, a) = \max\{|y - a| - \epsilon, 0\}$, $\epsilon > 0$ and its generalizations $V(y, a) = \max\{|y - a|^p - \epsilon, 0\}$, $\epsilon > 0, p > 1$ provide yet other examples. For classification, i.e., $Y = \{\pm 1\}$, other examples of loss functions used include the hinge loss $V(y, a) = \max\{1 - ya, 0\}$, the logistic loss $V(y, a) = \log(1 + e^{-ya})$ and the exponential loss $V(y, a) = e^{-ya}$. For all these examples f_ρ^V can be characterized, see for example Steinwart and Christmann (2008), and its measurability is easy to check.*

1. We denote the set of positive integers by $\mathbb{N}$.

2.2 Learning via Subgradient Methods with Early Stopping

To present the proposed learning algorithm we need a few preliminary definitions. Consider a reproducing kernel $K : X \times X \to \mathbb{R}$, that is a symmetric function, such that the matrix $(K(u_i, u_j))_{i,j=1}^\ell$ is positive semidefinite for any finite set of points $\{u_i\}_{i=1}^\ell$ in X. Recall that a reproducing kernel K defines a reproducing kernel Hilbert space (RKHS) $(\mathcal{H}_K, \|\cdot\|_K)$ as the completion of the linear span of the set $\{K_x(\cdot) := K(x, \cdot) : x \in X\}$ with respect to the inner product $\langle K_x, K_u\rangle_K := K(x, u), x, u \in X$ (Aronszajn, 1950). Moreover, assume the loss function V to be measurable and convex in its second argument, so that the corresponding left derivative V'_- exists and is non-decreasing at every point.

For a stepsize sequence $\{\eta_t > 0\}$, a stopping iteration $T > 2$ and an initial value $f_1 = 0$, we consider the iteration

$$f_{t+1} = f_t - \eta_t \frac{1}{m} \sum_{j=1}^m V'_-(y_j, f_t(x_j)) K_{x_j}, \qquad t = 1, \ldots, T. \tag{1}$$

The above iteration corresponds to the subgradient method (Bertsekas, 1999; Boyd et al., 2003) for minimizing the *empirical error* $\mathcal{E}_\mathbf{z} = \mathcal{E}_\mathbf{z}^V$ with respect to the loss V, which is given by

$$\mathcal{E}_\mathbf{z}(f) = \frac{1}{m} \sum_{j=1}^m V(y_j, f(x_j)).$$

Indeed, it is easy to see that $\frac{1}{m}\sum_{j=1}^m V'_-(y_j, f(x_j))K_{x_j} \in \partial\mathcal{E}_\mathbf{z}(f)$, the subgradient of the empirical risk for $f \in \mathcal{H}_K$. In the special case where the loss function is smooth, then (1) reduces to the gradient descent algorithm. Since the subgradient method is not a descent algorithm, rather then the last iterate, the so-called Cesáro mean is often considered, corresponding, for $T \in \mathbb{N}$, to the following weighted average

$$a_T = \sum_{t=1}^T \omega_t f_t, \qquad \omega_t = \frac{\eta_t}{\sum_{t=1}^T \eta_t}, \quad t = 1, \ldots, T. \tag{2}$$

Alternatively, the *best iterate* is also often considered, which is defined for $T \in \mathbb{N}$ by

$$b_T = \operatorname*{arg\,min}_{f_t : t=1,\cdots,T} \mathcal{E}_\mathbf{z}(f_t). \tag{3}$$

In what follows, we will consider the learning algorithms obtained with these different choices.

We note that, classical results (Bertsekas, 1999; Boyd et al., 2003; Boyd and Vandenberghe, 2004) on the subgradient method focus on how the iteration (1) can be used to minimize $\mathcal{E}_\mathbf{z}$. Different to these studies, in the following we are interested in showing how iteration (1) can be used to define a *statistical estimator*, hence a learning algorithm to minimize the expected risk $\mathcal{E}$, rather than the empirical risk $\mathcal{E}_\mathbf{z}$. We add two remarks.

Remark 2 (Early Stopping SVM and Kernel Perceptron) *If we consider the hinge loss function in* (1)*, the corresponding algorithm is closely related to a batch (kernel) version of the perceptron (Rosenblatt, 1962; Aizerman et al., 1964), where an entire pass over the data is done before updating the solution. Such an algorithm can also be seen as an early stopping version of Support Vector Machines (Cortes and Vapnik, 1995). Interestingly, in this case the whole regularization path is computed incrementally albeit sparsity could be lost.*

Remark 3 (Multiple Passes SGD) *In practice stochastic/incremental approaches are often used. The latter correspond to considering the iteration*

$$f_{t+1} = f_t - \eta_t V'_-(y_{j_t}, f_t(x_{j_t}))K_{x_{j_t}}, \qquad t = 1, \dots, T.$$

given some initialization. Compared to (1) *in the above expression the "batch" gradient is replaced by a point-wise gradient. The sequence $(j_t)_t$ defines the order in which points are visited and can be stochastic. The obtained iteration is a form of stochastic gradient/subgradient method. When $T > n$ the algorithm visits the point multiple times. Each full pass over the data is called an epoch, or a cycle, and the obtained iteration corresponds to a form of incremental gradient/subgradient. The analysis in the paper can be modified to account for these iterations. However, to keep the paper self-contained we defer such an analysis to a future paper.*

2.3 Numerical Realization

The simplest case to derive a numerical procedure from Algorithm 1 is when $X = \mathbb{R}^d$ for some $d \in \mathbb{N}$ and K is the associated inner product. In this case it is straightforward to see that $f_{t+1}(x) = w_{t+1}^\top x$ for all $x \in X$, with $w_1 = 0_{d\times 1} \in \mathbb{R}^d$ and

$$w_{t+1} = w_t - \eta_t \frac{1}{m} \sum_{j=1}^m V'_-(y_j, w_t^\top x_j) x_j, \qquad t = 1, \dots, T.$$

Here, $w_t \in \mathbb{R}^d$ for all t. Beyond the linear kernel, it can be easily seen that given a finite *dictionary*

$$\{\phi_i : X \to \mathbb{R}, i = 1, \dots, p\}, \qquad p \in \mathbb{N},$$

one can consider the kernel $K(x, x') = \sum_{i=1}^p \phi_i(x')\phi_i(x)$. In this case, it holds $f_{t+1}(x) = \sum_{i=1}^p w_{t+1}^i \phi_i(x) = w_{t+1}^\top \Phi(x)$, $\Phi(x) = (\phi_1(x), \dots, \phi_p(x))^\top$ for all $x \in X$, with $w_1 = 0_{p\times 1} \in \mathbb{R}^p$ and

$$w_{t+1} = w_t - \eta_t \frac{1}{m} \sum_{j=1}^m V'_-(y_j, w_t^\top \Phi(x_j))\Phi(x_j), \qquad t = 1, \dots, T.$$

Finally, for a general kernel it is easy to prove by induction that $f_{t+1}(x) = \sum_{j=1}^m c_{t+1}^j K(x, x_j)$ for all $x \in X$, with

$$c_{t+1} = c_t - \eta_t \frac{1}{m} g_t, \qquad t = 1, \dots, T,$$

for $c_1 = 0_{m\times 1} \in \mathbb{R}^m$ and $g_t \in \mathbb{R}^m$ with its i-th component $g_t^i = V'_-(y_i, \sum_{j=1}^m c_t^j K(x_i, x_j)), \forall i = 1, \cdots, m$. Here, $c_t = (c_t^1, \cdots, c_t^m)^\top$ for $t \in \mathbb{N}$. Indeed, the base case is straightforward to

check and moreover by the inductive hypothesis

$$f_{t+1} = \sum_{j=1}^m c_t^j K_{x_j} - \eta_t \frac{1}{m} \sum_{j=1}^m V'_-(y_j, f_t(x_j))K_{x_j} = \sum_{j=1}^m K_{x_j} \left(c_t^j - \eta_t \frac{1}{m} V'_-(y_j, f_t(x_j)) \right).$$

3. Main Results with Discussions

After presenting our main assumptions, in this section we state and discuss our main results.

3.1 Assumptions

Our results will be stated under several conditions on the triplet (ρ, V, K), that we describe and comment next. We begin with a basic assumption.

Assumption 1 *We assume the kernel to be bounded, that is $\kappa = \sup_{x\in X} \sqrt{K(x,x)} < \infty$. Moreover $\|f_\rho^V\|_\infty < \infty$ and $|V|_0 := \sup_{y\in Y} V(y, 0) < \infty$. Furthermore, we consider the following growth condition for the left derivative $V'_-(y, \cdot)$. For some $q \geq 0$ and constant $c_q > 0$, it holds,*

$$\left|V'_-(y, a)\right| \leq c_q(1 + |a|^q), \qquad \forall a \in \mathbb{R}, y \in Y. \tag{4}$$

The boundedness conditions on K, f_ρ^V and V are fairly common (Cucker and Zhou, 2007; Steinwart and Christmann, 2008). They could probably be weakened by considering a more involved analysis which is outside the scope of this paper. Interestingly, the *growth* condition on the left derivative of V is weaker than assuming the loss, or its gradient, to be Lipschitz in its second entry which is standard both in learning theory (Cucker and Zhou, 2007; Steinwart and Christmann, 2008) and in optimization (Boyd and Vandenberghe, 2004). We note that the growth condition (4) is implied by the requirement for the loss function to be Nemitski when Y is bounded, as introduced in De Vito et al. (2004) (see also Steinwart and Christmann, 2008). This latter condition, which is satisfied by most loss functions, is natural to provide variational characterizations of the learning problem.

The second assumption refines the above boundedness condition by considering a variance-expectation bound which quantifies the noise (level) in the measure ρ with respect to *balls* in the RKHS $B_R = \{f \in \mathcal{H}_K : \|f\|_K \leq R\}$, $R > 0$ (Cucker and Zhou, 2007; Steinwart and Christmann, 2008).

Assumption 2 *We assume that there exists an exponent $\tau \in [0, 1]$ and a positive constant c_τ such that for any $R \geq 1$ and $f \in B_R$, we have*

$$\int_Z \left\{ \left(V(y, f(x)) - V(y, f_\rho^V(x)\right)^2 \right\} d\rho \leq c_\tau R^{2+q-\tau} \left\{ \mathcal{E}(f) - \mathcal{E}(f_\rho^V) \right\}^\tau. \tag{5}$$

Assumption 2 always holds true for the square loss with $q = \tau = 1$, the hinge loss with $q = \tau = 0$, and more generally for Lipschitz loss functions with $\tau = 0$ and c_τ depending on $\|f_\rho^V\|_\infty$. In classification, the above condition can be related to the so-called Tsybakov margin condition. The latter quantifies the intuition that a classification problem is hard if the conditional probability of y given x is close to $1/2$ for many input points. More precisely if we denote by $\rho(y|x)$ the conditional probability for all $(x, y) \in Z$ and by ρ_X the marginal

probability on X, then we say that ρ satisfies the Tsybakov margin condition with exponent s if there exists a constant $C > 0$ such that for all $\delta > 0$

$$\rho_X(\{x \in X \ : \ |\rho(1|x) - \frac{1}{2}| \leq \delta\}) \leq (C\delta)^s.$$

Interestingly, under the Tsybakov margin condition, Assumption 2 holds for the hinge loss with $\tau = \frac{s}{s+1}$ and c_τ depending only on C.

The third condition is about the decay of the approximation error (Smale and Zhou, 2003).

Assumption 3 *Let $\lambda > 0$ and f_λ be a minimizer of:*

$$f_\lambda := \arg\min_{f \in \mathcal{H}_K} \mathcal{E}(f) + \lambda \|f\|_K^2. \tag{6}$$

The approximation error associated with the triplet (ρ, V, K) is defined by

$$\mathcal{D}(\lambda) = \mathcal{E}(f_\lambda) - \mathcal{E}(f_\rho^V) + \lambda \|f_\lambda\|_K^2. \tag{7}$$

We assume that for some $\beta \in (0, 1]$ and $c_\beta > 0$, the approximation error satisfies

$$\mathcal{D}(\lambda) \leq c_\beta \lambda^\beta, \qquad \forall \ \lambda > 0. \tag{8}$$

The above assumption is standard when analyzing regularized empirical risk minimization schemes and is related to the definition of interpolation spaces by means of K- functional (Cucker and Zhou, 2007). Interestingly, we will see in the following that it is also important when analyzing the approximation properties of the subgradient algorithm (1).

Finally, the last condition characterizes the *capacity* of a RKHS $\mathcal{H}_K$ in terms of empirical covering numbers, and plays an essential role in sample error estimates. Recall that for a subset $\mathcal{G}$ of a metric space (H, d), the covering number $\mathcal{N}(\mathcal{G}, \epsilon, d)$ is defined by

$$\mathcal{N}(\mathcal{G}, \epsilon, d) = \inf \left\{ l \in \mathbb{N} : \exists f_1, f_2, \cdots, f_l \subset H \text{ such that } \mathcal{G} \subset \bigcup_{i=1}^{\ell} \{f \in \mathcal{G} : d(f, f_i) \leq \epsilon\} \right\}.$$

Assumption 4 *Let $\mathcal{G}$ be a set of functions on X. The metric $d_{2,\mathbf{z}}$ is defined on $\mathcal{G}$ by*

$$d_{2,\mathbf{z}}(f, g) = \left\{ \frac{1}{m} \sum_{i=1}^{m} (f(z_i) - g(z_i))^2 \right\}^{1/2}, \quad f, g \in \mathcal{G}.$$

We assume that for some $\zeta \in (0, 2)$, $c_\zeta > 0$, the covering number of the unit ball B_1 in $\mathcal{H}_K$ with respect to $d_{2,\mathbf{z}}$ satisfies

$$\mathbb{E}_{\mathbf{z}} \left[\log \mathcal{N}\left(B_1, \epsilon, d_{2,\mathbf{z}}\right) \right] \leq c_\zeta \left(\frac{1}{\epsilon} \right)^\zeta, \qquad \forall \ \epsilon > 0. \tag{9}$$

The smaller ζ, the more stringent is the capacity assumption. As ζ approaches 2 we are essentially considering a capacity independent scenario, that is an arbitrary RKHS. In what follows, we will briefly comment on the connection between the above assumption and

other related assumptions. Recall that capacity of the RKHS may be measured by various concepts: covering numbers of balls B_R in $\mathcal{H}_K$, (dyadic) entropy numbers and decay of the eigenvalues of the integral operator $L_K : L^2_\rho \to L^2_\rho$ given by $L_K(f) = \int_X f(x) K_x d\rho_X(x)$, where $L^2_\rho = \{f : X \to \mathbb{R} \ : \ \int |f(x)|^2 d\rho_X(x) < \infty\}$. For a subset $\mathcal{G}$ of a metric space (H, d), the n-th entropy number is defined by

$$e_n(\mathcal{G}, d) = \inf \left\{ \varepsilon > 0 : \exists f_1, f_2, \cdots, f_{2^{n-1}} \text{ such that } \mathcal{G} \subset \bigcup_{i=1}^{2^{n-1}} \{f \in \mathcal{G} : d(f, f_i) \leq \varepsilon\} \right\}.$$

First, note that the covering and entropy numbers are equivalent (see for example Steinwart and Christmann, 2008, Lemma 6.21). Indeed, for $\zeta > 0$, the covering number $\mathcal{N}(\mathcal{G}, \epsilon, d)$ satisfies

$$\log \mathcal{N}(\mathcal{G}, \epsilon, d) \leq a_\zeta \left(\frac{1}{\epsilon} \right)^\zeta, \qquad \forall \epsilon > 0,$$

for some $a_\zeta > 0$, if and only if the entropy number $e_n(\mathcal{G}, d)$ satisfies

$$e_n(\mathcal{G}, d) \leq a'_\zeta \left(\frac{1}{n} \right)^{\frac{1}{\zeta}},$$

for some $a'_\zeta > 0$. Second, it is shown in Steinwart (2009) that if the eigenvalues of the integral operator L_K satisfy

$$\lambda_n \leq \tilde{a}_\zeta \left(\frac{1}{n} \right)^{\frac{2}{\zeta}}, \qquad n \geq 1,$$

for some constants $\tilde{a}_\zeta \geq 1$ and $\zeta \in (0, 2)$, then the expectation of the random entropy number $\mathbb{E}_{\mathbf{z}}[e_n\left(B_1, d_{2,\mathbf{z}}\right)]$ satisfies

$$\mathbb{E}_{\mathbf{z}}[e_n\left(B_1, d_{2,\mathbf{z}}\right)] \leq a_\zeta \left(\frac{1}{n} \right)^{\frac{1}{\zeta}}, \qquad n \geq 1,$$

for some constant a_ζ. Hence, using the equivalence of covering and entropy numbers, $\mathbb{E}_{\mathbf{z}}\left[\log \mathcal{N}\left(B_1, \epsilon, d_{2,\mathbf{z}}\right)\right]$ can be estimated from the eigenvalue decay of the integral operator L_K. Finally, since $d_{2,\mathbf{z}}(f, g) \leq \|f - g\|_\infty$, one has that for any $\epsilon > 0$, $\mathcal{N}\left(B_1, \epsilon, d_{2,\mathbf{z}}\right)$ is bounded by $\mathcal{N}\left(B_1, \epsilon, \|\cdot\|_\infty\right)$, the uniform covering number of B_1 under the metric $\|\cdot\|_\infty$. Thus, the covering number $\mathcal{N}\left(B_1, \epsilon, d_{2,\mathbf{z}}\right)$ can be also estimated given the uniform smoothness of the kernel (Zhou, 2003).

3.2 Finite Sample Bounds for General Convex Loss Functions

Our main results, in Theorems 4, 7 and 8, provide general stopping rules and corresponding upper bounds involving all the parameters defining the problem. These results are then illustrated and discussed in a series of corollaries considering special cases that allow for simpler statements, see in particular Corollary 6 in this subsection, Corollaries 9 and 10 in Subsection 3.3, and Theorems 11 and 12 in Subsection 3.4.

The following is our main result providing a general finite sample bound for the iterative regularization induced by the subgradient method for convex loss functions considering the last iterate. Here, $\lceil x \rceil$ denotes the smallest integer greater than or equal to $x \in \mathbb{R}$.

Theorem 4 *Assume (4) with $q \geq 0$, (5) with $\tau \in [0,1]$, (8) with $\beta \in (0,1]$ and (9) with $\zeta \in (0,2)$. Let $\eta_t = \eta_1 t^{-\theta}$ with $\frac{q}{q+1} < \theta < 1$ and η_1 satisfying*

$$0 < \eta_1 \leq \min\left\{ \frac{\sqrt{1-\theta}}{\sqrt{2}c_q(\kappa+1)^{q+1}}, \frac{1-\theta}{4|V|_0} \right\}. \tag{10}$$

If $T = \lceil m^\gamma \rceil$, then for any $0 < \delta < 1$, with confidence $1-\delta$, we have

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \leq \begin{cases} \widetilde{C} m^{-\alpha} \log \frac{2}{\delta}, & \text{when } \theta > \frac{q+1}{q+2}, \\ \widetilde{C} m^{-\alpha} \log m \log \frac{2}{\delta}, & \text{when } \theta \leq \frac{q+1}{q+2}, \end{cases}$$

where the power indices γ and α are defined as

$$\gamma = \begin{cases} \frac{2}{1-\theta} \frac{1}{(1+2\beta)(2-\tau+\zeta\tau/2)+q(1+\zeta/2)}, & \text{when } \theta \geq \frac{q+1}{q+2}, \\ \frac{2}{1-\theta}\left(1+\frac{2\beta(\theta(1+q)-q)}{1-\theta}\right) \frac{1}{(2-\tau+\zeta\tau/2)+q(1+\zeta/2)}, & \text{when } \theta < \frac{q+1}{q+2}, \end{cases} \tag{11}$$

$$\alpha = \begin{cases} \frac{\beta}{\beta(2-\tau+\zeta\tau/2)+\left\{\frac{2-\tau+\zeta\tau/2}{2}+\frac{q(1+\zeta/2)}{2}\right\}}, & \text{when } \theta \geq \frac{q+1}{q+2}, \\ \frac{\beta}{\beta(2-\tau+\zeta\tau/2)+\frac{1-\theta}{\theta(1+q)-q}\left\{\frac{2-\tau+\zeta\tau/2}{2}+\frac{q(1+\zeta/2)}{2}\right\}}, & \text{when } \theta < \frac{q+1}{q+2}, \end{cases} \tag{12}$$

and $\widetilde{C}$ is a positive constant independent of m or δ (given explicitly in the proof).

The proof is deferred to Section 5 and is based on a novel error decomposition, discussed in Section 3.6, integrating statistical and optimization aspects. We begin illustrating the above result for Lipschitz loss functions, that is considering $q = 0$, as follows.

Corollary 5 *Assume (4) with $q = 0$, (9) with $\zeta \in (0,2)$ and (8) with $\beta \in (0,1]$. Let $\eta_t = \eta_1 t^{-\theta}$ with $0 < \theta < 1$ and η_1 satisfying $0 < \eta_1 \leq \min\left\{ \frac{\sqrt{1-\theta}}{\sqrt{2}c_q(\kappa+1)}, \frac{1-\theta}{4|V|_0} \right\}$. If $T = \lceil m^\gamma \rceil$, then for any $0 < \delta < 1$, with confidence $1-\delta$, we have*

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \leq \begin{cases} \widetilde{C} m^{-\alpha} \log \frac{2}{\delta}, & \text{when } \theta > \frac{1}{2}, \\ \widetilde{C} m^{-\alpha} \log m \log \frac{2}{\delta}, & \text{when } \theta \leq \frac{1}{2}, \end{cases}$$

where the power indices γ and α are defined as

$$\gamma = \begin{cases} \frac{2}{(1-\theta)(2\beta+1)(2-\tau+\zeta\tau/2)}, & \text{when } \theta \geq \frac{1}{2}, \\ \frac{2}{(1-\theta+2\beta\theta)(2-\tau+\zeta\tau/2)}, & \text{when } \theta < \frac{1}{2}, \end{cases}$$

$$\alpha = \begin{cases} \frac{2\beta}{(2\beta+1)(2-\tau+\zeta\tau/2)}, & \text{when } \theta \geq \frac{1}{2}, \\ \frac{2\theta\beta}{(1-\theta+2\beta\theta)(2-\tau+\zeta\tau/2)}, & \text{when } \theta < \frac{1}{2}, \end{cases}$$

and $\widetilde{C}$ is a positive constant independent of m or δ.

For Lipschitz loss functions, Assumption 2 always holds true for $\tau = 0$. Also, if $f_\rho^V \in \mathcal{H}_K$, then Assumption 3 holds for $\beta = 1$ and $c_\beta \leq \|f_\rho^V\|_K^2$. In this case, γ and α from the above theorem are given by

$$\gamma = \max\left\{ \frac{1}{3(1-\theta)}, \frac{1}{1+\theta} \right\} \quad \text{and} \quad \alpha = \min\left\{ \frac{1}{3}, \frac{\theta}{1+\theta} \right\}.$$

Setting $\theta = 1/2$, we get the following result.

Corollary 6 *Assume (4) with $q = 0$, (9) with $\zeta \in (0,2)$ and $f_\rho^V \in \mathcal{H}_K$. Let $\eta_t = \eta_1 t^{-1/2}$ with η_1 satisfying $0 < \eta_1 \leq \min\left\{ \frac{1}{2c_q(\kappa+1)}, \frac{1}{8|V|_0} \right\}$. If $T = \lceil m^{2/3} \rceil$, then for any $0 < \delta < 1$, with confidence $1-\delta$, we have*

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \leq \widetilde{C} m^{-1/3} \log m \log \frac{2}{\delta}.$$

The above results give finite sample bounds on the excess risk, provided that a suitable stopping rule is considered. While the stopping rule in the above theorems is distribution dependent, a data-driven stopping rule can be given by hold-out cross validation and adaptively achieves the same bounds. The proof of this latter result is straightforward using the techniques in Caponnetto and Yao (2010) and is omitted. The above bounds directly yield strong consistency (almost sure convergence) using standard arguments. Interestingly, our analysis suggests that a decaying stepsize needs to be chosen to achieve meaningful error bounds. The stepsize choice can influence both the early stopping rule and the error bounds. More precisely, if the stepsize decreases fast enough, i.e., $\theta \geq \frac{q+1}{q+2}$, the stopping rule depends on the decay speed but the error bound does not. In this case, the best possible choice for the early stopping rule is $\theta = \frac{q+1}{q+2}$, that is $\eta_t \sim 1/\sqrt{t}$ in the case of Lipschitz loss functions. With this choice, if for example we take the limit $\beta \to 1$, $\tau \to 0$, we have that the stopping rule scales as $O(m^{2/3})$ whereas the corresponding finite sample bounds are of order $O(m^{-1/3})$. A slower stepsize decay given by $\theta < \frac{q+1}{q+2}$ affects both the stopping rule and the error bounds, but these results are worse. A more detailed discussion of the obtained bounds in comparison to other learning algorithms is postponed to Section 3.5.

To see how the number of passes and the decaying rate θ of stepsize affects the performances of our algorithms, we carry out simple numerical simulations that complement the above result. In Fig. 1 we consider simulated data, i.e. simple binary classification problem where the input space is two dimensional. The training and test error as a function of the number of iterations are reported for different stepsize values. In Fig. 2 we consider a real benchmark data-set and again report the training and test error for different stepsize values. The same qualitative behavior can be observed in simulated and real data. The empirical error decreases as a function of the number of iterations while the expected (test) error as a minimum. The effect is more evident when the stepsize choice is more aggressive, that is for θ close to zero.

Next we discuss the behavior of different variants of the proposed algorithm. As mentioned before, in the subgradient method, when the goal is empirical risk minimization, the average or best iterates are often considered (see Equations (2), (3)). It is natural to ask what are the properties of the estimator obtained with these latter choices, that is when they are used as approximate minimizers of the expected, rather than the empirical risk. The following theorem provides an answer.

Theorem 7 *Under the assumptions of Theorem 4, if $T = \lceil m^\gamma \rceil$ and $g_T = a_T$ (or b_T) then for any $0 < \delta < 1$, with confidence $1-\delta$, we have*

$$\mathcal{E}(g_T) - \mathcal{E}(f_\rho^V) \leq \begin{cases} \overline{C} m^{-\alpha} \log \frac{2}{\delta}, & \text{when } \theta \neq \frac{q+1}{q+2}, \\ \overline{C} m^{-\alpha} \log m \log \frac{2}{\delta}, & \text{when } \theta = \frac{q+1}{q+2}, \end{cases}$$

where the power indices γ and α are defined as in Theorem 4 and $\overline{C}$ is a positive constant independent of m or δ (can be given explicitly).

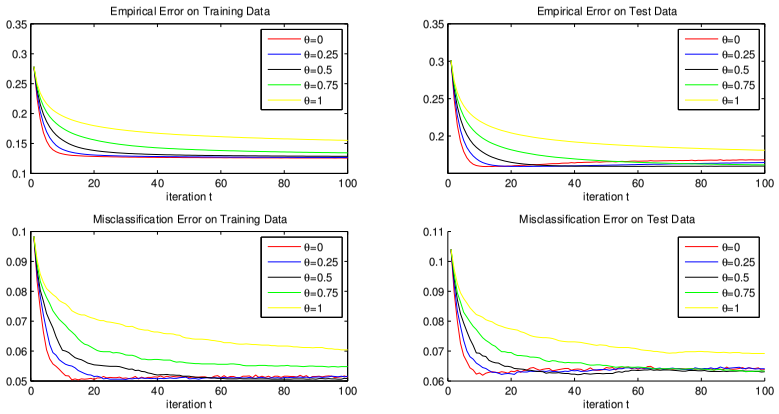


Figure 1: Performance of Algorithm (1) for the last iterates applied to synthetic data in binary classification with different θ, setting $\eta_1 = 1$, $V(y,f) = \max\{1-yf, 0\}$ and $\mathcal{H}_K = \mathbb{R}^2$. Samples for two classes are drawn from bivariate Gaussian distributions. The parameters for the Gaussian distributions are $\mu_1 = [2,0]^\top$, $\Sigma_1 = [5,3;3,5]/2$ and $\mu_{-1} = -\mu_1$, $\Sigma_{-1} = \Sigma_1$. For each given θ, we run Algorithm (1) 100 times for 100 independent training data, and calculate the corresponding test errors for 100 independent test data. In each trial, both of the training data and the test data are of 100. The errors averaged over these 100 trials are depicted as the above.

The above result shows, perhaps surprisingly, that the behavior of the average or best iterates is essentially the same as the last iterate. Indeed, there is only a subtle difference between the upper bounds in Theorem 7 and those in Theorem 4, since the latter have an extra $\log m$ factor when $\theta < \frac{q+1}{q+2}$.

In the next section, we consider the case where loss is not only convex but also smooth.

3.3 Finite Sample Bounds for Smooth Loss Functions

In this section, we additionally assume that $V(y,\cdot)$ is differentiable and $V'(y,\cdot)$ is Lipschitz continuous with constant $L > 0$, i.e., for any $y \in Y$ and $a, b \in \mathbb{R}$,

$$|V'(y,b) - V'(y,a)| \le L|b-a|.$$

For the logistic loss in binary classification (see Example 1), it is easy to prove that both $V(y,\cdot)$ and $V'(y,\cdot)$ are Lipschitz continuous with $L = 1$, for all $y \in Y$. With the above smoothness assumption, we prove the following convergence result.

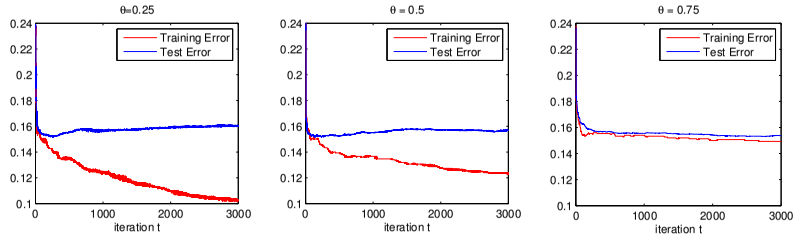


Figure 2: Misclassification errors of Algorithm (1) for the last iterates applied to Adult dataset with different values of θ, setting $V(y,f) = \max\{1-yf, 0\}$, $K(x,x') = \exp\{-\frac{\|x-x'\|_2^2}{2\sigma^2}\}$ and $m = 1500$. Here, σ is chosen as the median of the vector that consists of all Euclidean distances between training input vectors with different labels (Jaakkola et al., 1999). For each θ, η_1 is tuned using a holdout method.

Theorem 8 *Assume (4) with $q \ge 0$, (5) with $\tau \in [0,1]$, (8) with $\beta \in (0,1]$ and (9) with $\zeta \in (0,2)$. Assume that $V(y,\cdot)$ is differentiable and $V'(y,\cdot)$ is Lipschitz continuous with constant $L > 0$. Let $\eta_t = \eta_1 t^{-\theta}$ with $0 \le \theta < 1$ and $0 < \eta_1 \le \min(\frac{1-\theta}{2|V|_0}, (L\kappa^2)^{-1})$. If $T = \lceil m^\gamma \rceil$, then for any $0 < \delta < 1$, with confidence $1-\delta$, we have*

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \le \widetilde{C} m^{-\alpha} \log \frac{2}{\delta},$$

where the power indices γ and α are defined as

$$\gamma = \frac{2}{1-\theta} \frac{1}{(1+2\beta)(2-\tau+\zeta\tau/2) + q(1+\zeta/2)},$$
$$\alpha = \frac{\beta}{\beta(2-\tau+\zeta\tau/2) + \left\{\frac{2-\tau+\zeta\tau/2}{2} + \frac{q(1+\zeta/2)}{2}\right\}},$$

and $\widetilde{C}$ is a positive constant independent of m or δ.

The proof of this result will be given in Section 5. We can simplify the result by considering Lipschitz loss function ($q = 0$) and setting $\tau = 0$.

Corollary 9 *Under the assumptions of Theorem 8, let $q = 0$. If $T = \lceil m^{\frac{1}{(1-\theta)(2\beta+1)}} \rceil$, then for any $0 < \delta < 1$, with confidence $1-\delta$, we have*

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \le \widetilde{C} m^{-\frac{\beta}{2\beta+1}} \log \frac{2}{\delta},$$

where $\widetilde{C}$ is a positive constant independent of m or δ.

The finite sample bound obtained above is essentially the same as the best possible bound obtained for general convex loss functions. However, the important difference is that for

smooth loss functions, a constant stepsize can be chosen and allows to considerably improve the stopping rule. Indeed, if for example we can consider the limit $\beta \to 1$, $\tau \to 0$, we have that the stopping is $O(m^{1/3})$, rather than $O(m^{2/3})$, whereas the corresponding finite sample bound is again $O(m^{-1/3})$.

A similar simplification can be done for the square loss. Here, as mentioned in Example 1, f_ρ^V is the regression function f_ρ, and there holds $\mathcal{E}(f) - \mathcal{E}(f_\rho) = \|f - f_\rho\|^2_{L^2_{\rho_X}}$. In this case, Assumption 2 holds true for $\tau = 1$ and $c_\tau = 1$, and condition (8) can be characterized by requiring that $f_\rho \in L_K^{\beta/2}(L^2_{\rho_X})$ (Smale and Zhou, 2003; Caponnetto and De Vito, 2007), where $L_K^{\beta/2}$ is the $\frac{\beta}{2}$-th power of the positive operator L_K.

Corollary 10 *Let $V(y,a) = (y-a)^2$ and $|y| \leq |V|_0 < \infty$ almost surely. Assume $f_\rho \in L_K^{\beta/2}(L^2_{\rho_X})$ with $\beta \in (0,1]$ and (9) with $\zeta \in (0,2)$. Let $\eta_t = \eta_1 t^{-\theta}$ with $0 \leq \theta < 1$ and $0 < \eta_1 \leq \min(\frac{1-\theta}{2|V|_0}, \kappa^{-2})$. If $T = \lceil m^{\frac{2}{(1-\theta)(\beta+1)(\zeta+2)}} \rceil$, then for any $0 < \delta < 1$, with confidence $1-\delta$, we have*

$$\|f_T - f_\rho\|^2_{L^2_{\rho_X}} \leq \widetilde{C} m^{-\frac{2\beta}{(\beta+1)(\zeta+2)}} \log \frac{2}{\delta}.$$

In particular, if $f_\rho \in \mathcal{H}_K$,

$$\|f_T - f_\rho\|^2_{L^2_{\rho_X}} \leq \widetilde{C} m^{-\frac{1}{\zeta+2}} \log \frac{2}{\delta}.$$

Before comparing our bounds with obtained with other algorithms we last specialize our results to a binary classification setting.

3.4 Iterative Regularization for Classification: Surrogate Loss Functions and Hinge Loss

We briefly discuss how the above results allow to derive error bounds in binary classification problems. In this latter case $Y = \{1, -1\}$ and a natural choice for the loss function is the misclassification loss given by

$$V(y, b(x)) = \Theta\left(-y b(x)\right) \tag{13}$$

for $b : X \to Y$ and $\Theta(a) = 1$, if $a \geq 0$, and $\Theta(a) = 0$ otherwise. The corresponding generalization error, denoted by $\mathcal{R}$, is called misclassification risk, since it can be shown to be the probability of the event $\{(x,y) \in Z : y \neq b(x)\}$. The minimizer of the misclassification error is the Bayes rule $b_\rho : X \to Y$ given by

$$b_\rho(x) = \begin{cases} 1, & \text{if the conditional probability } \rho(1|x) \geq 1/2, \\ -1, & \text{otherwise.} \end{cases}$$

The misclassification loss (13) is neither convex nor smooth and thus leads to computationally intractable problems. In practice, a convex (so-called *surrogate*) loss function is typically considered and a classifier is obtained by estimating a real function f and then taking its sign defined as

$$\operatorname{sign}(f)(x) = \begin{cases} 1, & \text{if } f(x) \geq 0, \\ -1, & \text{otherwise.} \end{cases}$$

The question arises of if, and how, error bounds on the excess risk $\mathcal{E}(f) - \mathcal{E}(f_\rho^V)$ yields results on $\mathcal{R}(\operatorname{sign} f) - \mathcal{R}(b_\rho)$. Indeed, the so-called *comparison* results are known relating these different error measures, see for example Cucker and Zhou (2007); Steinwart and Christmann (2008) and references therein. We discuss in particular the case of the hinge loss function (see Example 1). In this case for all measurable functions f it holds that

$$\mathcal{R}(\operatorname{sign} f) - \mathcal{R}(b_\rho) \leq \mathcal{E}(f) - \mathcal{E}(f_\rho^V).$$

Indeed, the hinge loss satisfies Assumption (4) with $q = 0$ and, under Tsybakov noise condition, Assumption (5). Misclassification error bound, for iterative regularization with the hinge loss, can then be obtained as a corollary of Theorem 4.

Theorem 11 *Let $Y = \{1, -1\}$ and V be the hinge loss. Let $0 < \epsilon < \frac{1}{3}$ and (8) be satisfied with $\beta \in (0,1]$. Let $\eta_t = \eta_1 t^{-\theta}$ with $\theta > 1/2$ and $0 < \eta_1 \leq \min\left\{\frac{\sqrt{(1-\theta)}}{\sqrt{2}(\kappa+1)}, \frac{1-\theta}{4}\right\}$. If (9) is valid with $\zeta \in (0,2)$ and $T = \lceil m^{\frac{1}{(1-\theta)(2\beta+1)}} \rceil$, then with confidence $1-\delta$, we have*

$$\mathcal{R}\left(sign(f_T)\right) - \mathcal{R}(f_c) \leq \widetilde{C} m^{-\frac{\beta}{2\beta+1}} \log \frac{2}{\delta}. \tag{14}$$

In particular, if $\beta > \frac{1-3\epsilon}{1+6\epsilon}$ with $\epsilon \in (0, 1/3)$, then with confidence $1-\delta$,

$$\mathcal{R}\left(sign(f_T)\right) - \mathcal{R}(f_c) \leq \widetilde{C} m^{\epsilon - \frac{1}{3}} \log \frac{2}{\delta}.$$

The proof of the above result is given in Section 5, and comments on the obtained rates are given in the next section.

We end noting that, as illustrated by the next result where the stopping rule is kept fixed while the stepsize is chosen in a distribution dependent way. This observation is made precise by the following result.

Theorem 12 *Let $Y = \{1, -1\}$ and V be the hinge loss. Let $0 < \epsilon < \frac{1}{3}$ and (8) is satisfied with $1 > \beta > \frac{4-3\epsilon}{4+6\epsilon}$. Let $\eta_t = \eta_1 t^{-\theta}$ with $\theta = \frac{4\beta - 1 + 3\epsilon(2\beta+1)}{(2\beta+1)(2+3\epsilon)}$ and $0 < \eta_1 \leq \min\left\{\frac{\sqrt{2(1-\theta)}}{\kappa+1}, \frac{1-\theta}{4}\right\}$. If (9) is valid with $\zeta \in (0,2)$ and $T = \lceil m^{\frac{2}{3}+\epsilon} \rceil$, then with confidence $1-\delta$, we have*

$$\mathcal{R}\left(sign(f_T)\right) - \mathcal{R}(f_c) \leq \widetilde{C} m^{\frac{\epsilon}{4} - \frac{1}{3}} \log \frac{2}{\delta}.$$

3.5 Comparison with Other Learning Algorithms

As mentioned before iterative regularization has nice computational properties. The algorithm reduces to a simple first order method with low iteration cost and allows to easily compute the estimators corresponding to different regularization level (the regularization path), a crucial fact since model selection needs to be performed. In this view, the proposed procedure can be compared with standard approaches for example based on considering Support Vector Machines (SVM) or online variants such as Pegasos. In the former case, in principle a quadratic programming problem need to be solved for each regularization

parameter values. Our approach can be compared to more sophisticated approaches to compute the full SVM regularization path. In the latter case, the main difference is that in iterative regularization the early stopping rule is explicitly linked to the regularization level and in practice can be chosen by cross validation.

It is natural to compare the obtained statistical bounds with those for other learning algorithms. For general convex loss functions, the methods for which sharp bounds are available, are penalized empirical risk minimization (Tikhonov regularization), i.e.

$$f_{\mathbf{z},\lambda} = \arg\min_{f\in\mathcal{H}_K} \left\{ \mathcal{E}_{\mathbf{z}}(f) + \lambda \|f\|_K^2 \right\}, \quad \lambda > 0,$$

see for example Cucker and Zhou (2007); Steinwart and Christmann (2008) and references therein. The best error bounds for Tikhonov regularization with Lipschitz loss functions, see for example Steinwart and Christmann (2008, Chapter 7), are of order $O(m^{-\alpha'})$ with

$$\alpha' = \min\left\{ \frac{2\beta}{\beta+1}, \frac{\beta}{(2-\zeta/2-\tau+\tau\zeta/2)\beta+\zeta/2} \right\},$$

which reduces to

$$\alpha' = \frac{\beta}{\beta+1}$$

in the capacity independent limit ($\zeta \to 2$). From Corollary 5 for Lipschitz loss functions, we see that the bounds we obtain are of order $O(m^{-\alpha})$ with the exponent

$$\alpha = \frac{2\beta}{(2\beta+1)(2-\tau+\zeta\tau/2)},$$

reducing to

$$\alpha = \frac{\beta}{2\beta+1}$$

in the capacity independent limit. Hence, the obtained bounds are worse than the best ones available for Tikhonov regularization. However, the analysis of the latter does not take into account the optimization error and it is still an open question whether the best rate is preserved when such an error is incorporated. At this point we believe this gap to be a byproduct of our analysis rather than a fundamental fact, and addressing this point should be a subject of further work. Moreover, we note that our analysis allows to derive error bound for all Nemitski loss functions rather than only Lipschiz loss functions.

Beyond Tikhnov regularization, we can compare with the online regularization scheme for the hinge loss. The online learning algorithms with a regularization sequence $\{\lambda_t > 0\}_t$ defined by

$$f_{t+1} = \begin{cases} (1-\eta_t\lambda_t) f_t, & \text{if } y_t f_t(x_t) > 1, \\ (1-\eta_t\lambda_t) f_t + \eta_t y_t K_{x_t}, & \text{if } y_t f_t(x_t) \le 1. \end{cases} \tag{15}$$

were studied in Ying and Zhou (2006); Ye and Zhou (2007). Our results improve the results in Ying and Zhou (2006); Ye and Zhou (2007) in two aspects. The bound obtained in Ying and Zhou (2006) is of the form $O(T^{\epsilon-\frac{1}{4}})$ while the bound in Theorem 12 is of type $O(T^{\frac{9}{8}\epsilon-\frac{1}{2}})$ by substituting the expression $m^{\frac{2}{3}+\epsilon}$ for T. Moreover, our results are with high probability and promptly yield almost sure convergence whereas the results in Ying and

Zhou (2006) are only in expectation. We note that, interestingly, sharp bounds for Lipschitz loss functions are derived in Orabona (2014), although the obtained results do not take into account the capacity and variance assumptions that could lead to large improvements.

We next compare with the previous results on iterative regularization. The main results available thus far have been obtained for the square loss, for which bounds have been first derived for gradient descent in Buhlmann and Yu (2003), but only for a fixed design regression setting, and in Yao et al. (2007) for a general statistical learning setting. While the bounds in Yao et al. (2007) are suboptimal, they have later been improved in Bauer et al. (2007); Caponnetto and Yao (2010); Raskutti et al. (2014). Interestingly, sharp error bounds have also been proved for iterative regularization induced by other, potentially faster, iterative techniques, including incremental gradient (Rosasco and Villa, 2014), conjugate gradient (Blanchard and Nicole, 2010) and the so-called ν-method (Bauer et al., 2007; Caponnetto and Yao, 2010), an accelerated gradient descent technique related to Chebyshev method (Engl et al., 1996). The best obtained bounds are of order $O(m^{-\frac{2\beta}{2\beta+\zeta}})$ and can be shown to be optimal since they match the corresponding minimax lower bound (Caponnetto and De Vito, 2007). The bound obtained in Corollary 10 holds for all smooth Nemitski loss functions but is of order $O(m^{-\frac{2\beta}{(2+\zeta)(\beta+1)}})$, which is worse. In the capacity independent limit, the best available bound we obtain is of order $O(m^{-\frac{\beta}{2(\beta+1)}})$, whereas the optimal bound is of order $O(m^{-\frac{\beta}{\beta+1}})$. Also, in this case, the reason for the gap appears to be of technical reason and should be further studied.

Finally, before giving the detailed proofs, in the next subsection, we discuss the general error decomposition underlying our approach, which highlights the interplay between statistics and optimization and could be also useful in other contexts.

3.6 Error Decomposition

Theorems 4 and 8 rely on a natural error decomposition that we derive next. The goal is to estimate the excess risk $\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)$, and the starting point is to split the error by introducing a *reference* function $f_* \in \mathcal{H}_K$,

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) = \mathcal{E}(f_T) - \mathcal{E}(f_*) + \mathcal{E}(f_*) - \mathcal{E}(f_\rho^V). \tag{16}$$

The above equation can be further developed by considering

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) = (\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_*)) + (\mathcal{E}(f_T) - \mathcal{E}_{\mathbf{z}}(f_T) + \mathcal{E}_{\mathbf{z}}(f_*) - \mathcal{E}(f_*)) + \left(\mathcal{E}(f_*) - \mathcal{E}(f_\rho^V)\right). \tag{17}$$

Inspection of the above expression provides several insights. The first term is a computational error related to optimization. It quantifies the discrepancy between the empirical errors of the iterate defined by the subgradient method and that of the reference function. The second term is a sample error and can be studied using empirical process theory, provided that a bound on the norm of the iterates (and of the reference function) is available. Indeed, to get a sharper concentration estimate *recentering* can be considered (Cucker and Zhou, 2007; Steinwart and Christmann, 2008)

$$\left\{ \left(\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)\right) - \left(\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)\right) \right\} + \left\{ \left(\mathcal{E}_{\mathbf{z}}(f_*) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)\right) - \left(\mathcal{E}(f_*) - \mathcal{E}(f_\rho^V)\right) \right\}.$$

Note that the second addend can be negative so that we effectively only need to control

$$\mathcal{F}_{\mathbf{z}}(f_*) = \max\left\{(\mathcal{E}_{\mathbf{z}}(f_*) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)) - (\mathcal{E}(f_*) - \mathcal{E}(f_\rho^V)),\ 0\right\}. \tag{18}$$

Finally the last term suggests that a natural choice for the reference function is an *almost minimizer* of the expected risk, having bounded norm, and for which the approximation level can be quantified. While there is a certain degree of freedom in the latter choice, in the following we will consider $f_* = f_\lambda$, the minimizer of (6). With this latter choice we can control

$$\mathcal{A}(f_*) = \mathcal{E}(f_*) - \mathcal{E}(f_\rho^V)$$

by $\mathcal{D}(\lambda)$ given in Assumption 3.

Collecting some of the above observations, we have the following lemma.

Lemma 13 *For $f_* \in \mathcal{H}_K$, we have*

$$\begin{aligned}&\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)\\ &\le \left\{\left(\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)\right) - \left(\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)\right) + \mathcal{F}_{\mathbf{z}}(f_*)\right\} + (\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_*)) + \mathcal{A}(f_*).\end{aligned} \tag{19}$$

In the next sections, we proceed estimating the various error terms in the above error decomposition. We will first deal with the computational error, the analysis of which is the main technical contribution of the paper and then proceed to consider the sample and approximation error terms. The best stopping criterion and corresponding rates are derived by suitably balancing these different error terms.

4. Computational Error

In this section, we will bound the iterates and estimate the computational error from Lemma 13.

4.1 Bounds on Iterates

We introduce the following key lemma, which will be used several times in our analysis.

Lemma 14 *For any fixed $f \in \mathcal{H}_K$ and $t = 1, \dots, T$,*

$$\|f_{t+1} - f\|_K^2 \le \|f_t - f\|_K^2 + \eta_t^2 G_t^2 + 2\eta_t[\mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_t)], \tag{20}$$

where

$$G_t^2 = \left\| \frac{1}{m}\sum_{j=1}^m V_-'(y_j, f_t(x_j))K_{x_j} \right\|_K^2 \le c_q^2(\kappa+1)^{2q+2}\max\left\{1, \|f_t\|_K^{2q}\right\}. \tag{21}$$

Proof Computing inner product $\langle f_{t+1} - f, f_{t+1} - f\rangle_K$ with f_{t+1} given by (1) yields

$$\|f_{t+1} - f\|_K^2 = \|f_t - f\|_K^2 + \eta_t^2 G_t^2 + \frac{2\eta_t}{m}\sum_{j=1}^m V_-'(y_j, f_t(x_j))\left\langle K_{x_j}, f - f_t\right\rangle_K.$$

Using the reproducing property

$$f(x) = \langle f, K_x\rangle_K, \qquad \forall f \in \mathcal{H}_K, x \in X, \tag{22}$$

and Assumption 1, we get

$$\|f\|_\infty \le \kappa\|f\|_K, \qquad \forall f \in \mathcal{H}_K, \tag{23}$$

and

$$\|f_{t+1} - f\|_K^2 = \|f_t - f\|_K^2 + \eta_t^2 G_t^2 + \frac{2\eta_t}{m}\sum_{j=1}^m V_-'(y_j, f_t(x_j))(f(x_j) - f_t(x_j)). \tag{24}$$

Since $V(y_j, \cdot)$ is a convex function, we have

$$V_-'(y_j, a)(b-a) \le V(y_j, b) - V(y_j, a), \qquad \forall a, b \in \mathbb{R}.$$

Using this expression to (24) gives

$$\|f_{t+1} - f\|_K^2 \le \|f_t - f\|_K^2 + \eta_t^2 G_t^2 + \frac{2\eta_t}{m}\sum_{j=1}^m \left[V(y_j, f(x_j)) - V(y_j, f_t(x_j))\right],$$

where the last term is exactly $2\eta_t[\mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_t)]$.

By (4), (23), and the observation $\|K_{x_j}\|_K = \sqrt{K(x_j, x_j)} \le \kappa$, we find

$$\begin{aligned} G_t &= \left\| \frac{1}{m}\sum_{j=1}^m V_-'(y_j, f_t(x_j))K_{x_j} \right\|_K \le \frac{\kappa}{m}\sum_{j=1}^m \left|V_-'(y_j, f_t(x_j))\right| \\ &\le \frac{\kappa}{m}\sum_{j=1}^m c_q(1 + |f_t(x_j)|^q) \le \kappa c_q(1 + \kappa^q\|f_t\|_K^q), \end{aligned}$$

and the desired bound follows. ∎

Using the above lemma, we can bound the iterated sequence as follows.

Lemma 15 *Let $0 \le \theta < 1$ satisfying $\theta \ge \frac{q}{q+1}$ and $\eta_t = \eta_1 t^{-\theta}$ with η_1 satisfying (10). Then for $t = 1, \dots, T$,*

$$\|f_{t+1}\|_K \le t^{\frac{1-\theta}{2}}. \tag{25}$$

Proof We prove our statement by induction. Taking $f = 0$ in Lemma 14, we know that

$$\|f_{t+1}\|_K^2 \le \|f_t\|_K^2 + \eta_t^2 G_t^2 + 2\eta_t[\mathcal{E}_{\mathbf{z}}(0) - \mathcal{E}_{\mathbf{z}}(f_t)] \le \|f_t\|_K^2 + \eta_t^2 G_t^2 + 2\eta_t|V|_0.$$

This verifies (25) for the case $t = 1$ since $f_1 = 0$ and $\eta_1^2 c_q^2(\kappa+1)^{2q+2} + 2\eta_1|V|_0 \le 1$. Assume $\|f_t\|_K \le (t-1)^{\frac{1-\theta}{2}}$ with $t \ge 2$. Then

$$G_t^2 \le c_q^2(\kappa+1)^{2q+2}(t-1)^{(1-\theta)q}.$$

Hence,

$$\begin{aligned}\|f_{t+1}\|_K^2 &\leq (t-1)^{1-\theta} + \eta_1^2 t^{-2\theta} c_q^2 (\kappa+1)^{2q+2} t^{(1-\theta)q} + 2\eta_1 t^{-\theta}|V|_0 \\ &\leq t^{1-\theta}\left\{\left(1-\frac{1}{t}\right)^{1-\theta} + \frac{\eta_1^2 c_q^2(\kappa+1)^{2q+2}}{t^{(q+1)\theta+1-q}} + \frac{2\eta_1|V|_0}{t}\right\}.\end{aligned}$$

Since $\left(1-\frac{1}{t}\right)^{1-\theta} \leq 1-\frac{1-\theta}{t}$ and the condition $\theta \geq \frac{q}{q+1}$ implies $(q+1)\theta+1-q \geq 1$, we have

$$\|f_{t+1}\|_K^2 \leq t^{1-\theta}\left\{1-\frac{1-\theta}{t} + \frac{\eta_1^2 c_q^2(\kappa+1)^{2q+2}}{t} + \frac{2\eta_1|V|_0}{t}\right\}.$$

Finally we use the restriction (10) for η_1 and find $\|f_{t+1}\|_K^2 \leq t^{1-\theta}$. This completes the induction procedure and proves our conclusion. ■

By taking $f = f_t$ in (20), we see the following estimate for $\|f_{t+1} - f_t\|_K$ from Lemmas 14 and 15.

Corollary 16 *Under the assumptions of Lemma 15, we have for $t = 1, \ldots, T$,*

$$\|f_{t+1} - f_t\|_K \leq \eta_1 c_q (\kappa+1)^{q+1} t^{\frac{(1-\theta)q}{2}-\theta}. \tag{26}$$

Observe from the restriction $\theta \geq \frac{q}{q+1}$ in Lemma 15 that the power index in (26) satisfies $\frac{(1-\theta)q}{2} - \theta \leq -\frac{q}{2(q+1)} \leq 0$.

4.2 Computational Error for the Last Iterate

In this subsection, we estimate the computational error $\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_*)$ for an arbitrary $f_* \in \mathcal{H}_K$. Some ideas for estimating the average error in our proof are taken from Boyd et al. (2003); Shamir and Zhang (2013).

Lemma 17 *Assume (4) with $q \geq 0$. Let $f_* \in \mathcal{H}_K$. If $\eta_t = \eta_1 t^{-\theta}$ with $0 < \theta < 1$ satisfying $\theta > \frac{q}{q+1}$ and η_1 satisfying (10), then we have*

$$\begin{aligned}\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_*) \leq &\left(\frac{\|f_*\|_K^2}{2\eta_1} + \widetilde{C}_1\right)\Lambda_{T,\theta} \\ &+ \frac{T^\theta}{2\eta_1}\sum_{k=1}^{T-1}\frac{1}{k+1}\left[\frac{1}{k}\sum_{t=T-k+1}^{T} 2\eta_t - 2\eta_{T-k}\right]\{\mathcal{E}_{\mathbf{z}}(f_{T-k}) - \mathcal{E}_{\mathbf{z}}(f_*)\},\end{aligned} \tag{27}$$

where $\Lambda_{T,\theta}$ is defined by

$$\Lambda_{T,\theta} = \begin{cases} \frac{1}{(q+2)\theta-(q+1)}T^{-(1-\theta)}, & \text{when } \theta > \frac{q+1}{q+2}, \\ (\log T)T^{-(1-\theta)}, & \text{when } \theta = \frac{q+1}{q+2}, \\ \frac{1}{(q+1)-(q+2)\theta}(\log T)T^{-(\theta(1+q)-q)}, & \text{when } \theta < \frac{q+1}{q+2}, \end{cases} \tag{28}$$

and $\widetilde{C}_1$ is a positive constant depending on q, κ, θ (independent of T, m or f_* and given explicitly in the proof).

Proof Lemma 14 plays a key role in our proof. In particular, we shall apply the following equivalent form of inequality (20) from Lemma 14 several times with various choices of $f \in \mathcal{H}_K$:

$$2\eta_t\left[\mathcal{E}_{\mathbf{z}}(f_t) - \mathcal{E}_{\mathbf{z}}(f)\right] \leq \left\{\|f_t - f\|_K^2 - \|f_{t+1} - f\|_K^2\right\} + \eta_t^2 G_t^2. \tag{29}$$

Step 1: Error decomposition. Decompose the weighted empirical error $2\eta_T\mathcal{E}_{\mathbf{z}}(f_T)$ as

$$\begin{aligned}2\eta_T\mathcal{E}_{\mathbf{z}}(f_T) &= \frac{1}{2}\{2\eta_T\mathcal{E}_{\mathbf{z}}(f_T) + 2\eta_{T-1}\mathcal{E}_{\mathbf{z}}(f_{T-1})\} \\ &\quad + \frac{1}{2}2\eta_T\{\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_{T-1})\} + \frac{1}{2}\{2\eta_T - 2\eta_{T-1}\}\mathcal{E}_{\mathbf{z}}(f_{T-1}) \\ &= \frac{1}{3}\{2\eta_T\mathcal{E}_{\mathbf{z}}(f_T) + 2\eta_{T-1}\mathcal{E}_{\mathbf{z}}(f_{T-1}) + 2\eta_{T-2}\mathcal{E}_{\mathbf{z}}(f_{T-2})\} \\ &\quad + \frac{1}{2\times 3}\{2\eta_T[\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_{T-2})] + 2\eta_{T-1}[\mathcal{E}_{\mathbf{z}}(f_{T-1}) - \mathcal{E}_{\mathbf{z}}(f_{T-2})]\} \\ &\quad + \frac{1}{2}2\eta_T\{\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_{T-1})\} + \frac{1}{2}\{2\eta_T - 2\eta_{T-1}\}\mathcal{E}_{\mathbf{z}}(f_{T-1}) \\ &\quad + \frac{1}{2\times 3}\{[2\eta_T - 2\eta_{T-2}] + [2\eta_{T-1} - 2\eta_{T-2}]\}\mathcal{E}_{\mathbf{z}}(f_{T-2}).\end{aligned}$$

Repeating the above process by means of the decomposition

$$\begin{aligned}&\frac{1}{k}\sum_{j=0}^{k-1} 2\eta_{T-j}\mathcal{E}_{\mathbf{z}}(f_{T-j}) = \frac{1}{k+1}\sum_{j=0}^{k} 2\eta_{T-j}\mathcal{E}_{\mathbf{z}}(f_{T-j}) \\ &+\frac{1}{k(k+1)}\sum_{j=0}^{k-1} 2\eta_{T-j}\{\mathcal{E}_{\mathbf{z}}(f_{T-j}) - \mathcal{E}_{\mathbf{z}}(f_{T-k})\} + \frac{1}{k(k+1)}\sum_{j=0}^{k-1}\{2\eta_{T-j} - 2\eta_{T-k}\}\mathcal{E}_{\mathbf{z}}(f_{T-k})\end{aligned}$$

with $k = 3, \ldots, T-1$, we know that

$$\begin{aligned}2\eta_T\mathcal{E}_{\mathbf{z}}(f_T) &= \frac{1}{T}\sum_{j=0}^{T-1} 2\eta_{T-j}\mathcal{E}_{\mathbf{z}}(f_{T-j}) + \sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{j=0}^{k-1} 2\eta_{T-j}\{\mathcal{E}_{\mathbf{z}}(f_{T-j}) - \mathcal{E}_{\mathbf{z}}(f_{T-k})\} \\ &\quad + \sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{j=0}^{k-1}\{2\eta_{T-j} - 2\eta_{T-k}\}\mathcal{E}_{\mathbf{z}}(f_{T-k}).\end{aligned}$$

Applying the same process to the sequence $\{2\eta_t\mathcal{E}_{\mathbf{z}}(f_*)\}_{t=1}^T$ yields

$$2\eta_T\mathcal{E}_{\mathbf{z}}(f_*) = \frac{1}{T}\sum_{j=0}^{T-1} 2\eta_{T-j}\mathcal{E}_{\mathbf{z}}(f_*) + \sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{j=0}^{k-1}\{2\eta_{T-j} - 2\eta_{T-k}\}\mathcal{E}_{\mathbf{z}}(f_*).$$

Hence the following error decomposition holds true:

$$\begin{aligned}2\eta_T\left\{\mathcal{E}_{\mathbf{z}}(f_T)-\mathcal{E}_{\mathbf{z}}(f_*)\right\} &= \frac{1}{T}\sum_{t=1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_*)\right\}\\ &+\sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\\ &+\sum_{k=1}^{T-1}\frac{1}{k+1}\left[\frac{1}{k}\sum_{t=T-k+1}^{T}2\eta_t-2\eta_{T-k}\right]\left\{\mathcal{E}_{\mathbf{z}}(f_{T-k})-\mathcal{E}_{\mathbf{z}}(f_*)\right\}.\end{aligned}\tag{30}$$

Step 2: Average error in the first term of (30). Choosing $f=f_*$ in (29) and taking summation over $t=1,\ldots,T$ together with (21) and Lemma 15 yield

$$\begin{aligned}\sum_{t=1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_*)\right\} &\le \|f_1-f_*\|_K^2-\|f_{T+1}-f_*\|_K^2+\sum_{t=1}^{T}\eta_t^2G_t^2\\ &\le \|f_*\|_K^2+\sum_{t=1}^{T}\eta_1^2c_q^2(\kappa+1)^{2q+2}t^{q(1-\theta)-2\theta}.\end{aligned}$$

Since $1>\theta>\frac{q}{q+1}$, we find $-2<q(1-\theta)-2\theta<0$. Moreover, $q(1-\theta)-2\theta<-1$ if and only if $\theta>\frac{q+1}{q+2}$. The following bound for the first term of (30) then follows

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_*)\right\}\\ &\le\begin{cases}\left(\|f_*\|_K^2+C_{q,\kappa}\frac{(2+q)\theta-q}{(2+q)\theta-q-1}\right)T^{-1}, & \text{when } \theta>\frac{q+1}{q+2},\\ \left(\|f_*\|_K^2+2C_{q,\kappa}\right)(\log T)T^{-1}, & \text{when } \theta=\frac{q+1}{q+2},\\ \left(\|f_*\|_K^2+C_{q,\kappa}\frac{2}{q+1-(2+q)\theta}\right)T^{q-(2+q)\theta}, & \text{when } \theta<\frac{q+1}{q+2},\end{cases}\end{aligned}$$

where $C_{q,\kappa}$ is the constant given by

$$C_{q,\kappa}=\eta_1^2c_q^2(\kappa+1)^{2q+2}.$$

Step 3: Moving average error in the second term of (30). Let $k\in\{1,\ldots,T-1\}$. Choosing $f=f_{T-k}$ in (29) and taking summation over $t=T-k+1,\ldots,T$ yield

$$\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\le\|f_{T-k+1}-f_{T-k}\|_K^2+\sum_{t=T-k+1}^{T}\eta_t^2G_t^2$$

By Corollary 16,

$$\|f_{T-k+1}-f_{T-k}\|_K^2\le\eta_1^2c_q^2(\kappa+1)^{2(q+1)}(T-k)^{(1-\theta)q-2\theta}.$$

This bound is the term with $t=T-k+1$ of the following estimate which is a consequence of Lemma 15

$$\sum_{t=T-k+1}^{T}\eta_t^2G_t^2\le\sum_{t=T-k+1}^{T}\eta_1^2c_q^2(\kappa+1)^{2q+2}t^{q(1-\theta)-2\theta}.$$

Hence

$$\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\le C_{q,\kappa}\sum_{t=T-k}^{T}t^{q(1-\theta)-2\theta}=C_{q,\kappa}\sum_{t=T-k}^{T}t^{-q^*},$$

where we denote $q^*=2\theta-q(1-\theta)$. We know that $0<q^*<2$ and $q^*=1$ when $\theta=\frac{q+1}{q+2}$. So

$$\sum_{t=T-k+1}^{T}t^{-q^*}\le\int_{T-k}^{T}x^{-q^*}dx\le\begin{cases}\frac{T^{1-q^*}-(T-k)^{1-q^*}}{1-q^*}, & \text{when } \theta\ne\frac{q+1}{q+2},\\ \log\frac{T}{T-k}, & \text{when } \theta=\frac{q+1}{q+2}.\end{cases}$$

When $\theta<\frac{q+1}{q+2}$, we have $q^*<1$ and for $k\le\frac{T}{2}$, we see from the mean value theorem that

$$\frac{T^{1-q^*}-(T-k)^{1-q^*}}{1-q^*}=T^{1-q^*}\frac{1-(1-\frac{k}{T})^{1-q^*}}{1-q^*}\le T^{1-q^*}\frac{(1-q^*)(1-\frac{k}{T})^{-q^*}\frac{k}{T}}{1-q^*}$$

which is exactly $(T-k)^{-q^*}k$. Thus,

$$\sum_{t=T-k}^{T}t^{-q^*}\le(T-k)^{-q^*}k+(T-k)^{-q^*}\le2k(T-k)^{-q^*}\le2\cdot2^{q^*}T^{-q^*}k.$$

For $k\ge\frac{T}{2}$,

$$\sum_{t=T-k}^{T}t^{-q^*}\le\frac{T^{1-q^*}-(T-k)^{1-q^*}}{1-q^*}+(T-k)^{-q^*}\le\frac{T^{1-q^*}}{1-q^*}.$$

It follows that

$$\begin{aligned}&\sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\\ &\le C_{q,\kappa}\sum_{k\le T/2}\frac{1}{k(k+1)}\sum_{t=T-k}^{T}t^{-q^*}+C_{q,\kappa}\sum_{T-1\ge k>T/2}\frac{1}{k(k+1)}\sum_{t=T-k}^{T}t^{-q^*}\\ &\le 2C_{q,\kappa}\sum_{k\le T/2}\frac{1}{k(k+1)}2^{q^*}T^{-q^*}k+2C_{q,\kappa}\sum_{T-1\ge k>T/2}\frac{1}{k(k+1)}\frac{T^{1-q^*}}{1-q^*}\\ &\le 2C_{q,\kappa}\left(2^{q^*}+\frac{2}{1-q^*}\right)(\log T)T^{q(1-\theta)-2\theta}.\end{aligned}$$

When $\theta=\frac{q+1}{q+2}$, we see from the mean value theorem that

$$\log\frac{T}{T-k}=-\log\left(1-\frac{k}{T}\right)\le\frac{k}{T}\frac{1}{1-\frac{k}{T}}=\frac{k}{T-k}.$$

It follows that

$$\begin{aligned}&\sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\\&\le C_{q,\kappa}\sum_{k=1}^{T-1}\frac{1}{(T-k)k}=C_{q,\kappa}\frac{1}{T}\sum_{k=1}^{T-1}\left\{\frac{1}{k}+\frac{1}{T-k}\right\}\\&\le 4C_{q,\kappa}\frac{\log T}{T}.\end{aligned}$$

When $\theta>\frac{q+1}{q+2}$, we have $q^*>1$ and for $k\le\frac{T}{2}$,

$$\frac{T^{1-q^*}-(T-k)^{1-q^*}}{1-q^*}=T^{1-q^*}\frac{(1-\frac{k}{T})^{1-q^*}-1}{q^*-1}\le 2^{q^*}T^{-q^*}k.$$

For $k>\frac{T}{2}$, $\sum_{t=T-k}^{T}t^{-q^*}\le(k+1)2^{q^*}T^{-q^*}$. Then,

$$\begin{aligned}&\sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\\&\le 2^{q^*+1}C_{q,\kappa}T^{-q^*}\sum_{k=1}^{T-1}\frac{1}{k+1}\le 2^{q^*+1}C_{q,\kappa}T^{-q^*}\log T\\&\le 2^{q^*+1}C_{q,\kappa}\frac{1}{\mathrm{e}(q^*-1)}T^{-1}.\end{aligned}$$

Thus the second term of (30) can also be bounded as

$$\begin{aligned}&\sum_{k=1}^{T-1}\frac{1}{k(k+1)}\sum_{t=T-k+1}^{T}2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}\\&\le\begin{cases}\frac{2^{q^*+1}C_{q,\kappa}}{\mathrm{e}(q^*-1)}T^{-1}, & \text{when } \theta>\frac{q+1}{q+2},\\ 4C_{q,\kappa}(\log T)T^{-1}, & \text{when } \theta=\frac{q+1}{q+2},\\ 2C_{q,\kappa}\left(2^{q^*}+\frac{2}{1-q^*}\right)(\log T)T^{q-(2+q)\theta}, & \text{when } \theta<\frac{q+1}{q+2}.\end{cases}\end{aligned}$$

Putting all the above estimates for the first two terms into (30), we see that the desired bound (27) holds true with the constant $\widetilde{C}_1$ given by

$$\widetilde{C}_1=\begin{cases}\eta_1c_q^2(\kappa+1)^{2q+2}\left((2+q)\theta-q+2^{(2+q)\theta-q}\right), & \text{when } \theta>\frac{q+1}{q+2},\\ 6\eta_1c_q^2(\kappa+1)^{2q+2}, & \text{when } \theta=\frac{q+1}{q+2},\\ \eta_1c_q^2(\kappa+1)^{2q+2}\left(2^{(2+q)\theta-q}+3\right), & \text{when } \theta<\frac{q+1}{q+2}.\end{cases}$$

The proof of Lemma 17 is complete. ∎

Lemma 17 is useful and can be used in a stochastic convex optimization problem, other than learning. In what follows, we shall see that how it can be used in our specified learning problems. For notational simplicity, with $\widetilde{R}>0$ we denote

$$\mathcal{M}_{\mathbf{z}}(\widetilde{R})=\sup_{f\in B_{\widetilde{R}}}\max\left\{\mathcal{E}_{\mathbf{z}}(f_\rho^V)-\mathcal{E}_{\mathbf{z}}(f),\ 0\right\}. \tag{31}$$

Proposition 18 *Under the assumptions of Lemma 17, we have*

$$\mathcal{E}_{\mathbf{z}}(f_T)-\mathcal{E}_{\mathbf{z}}(f_*)\le\frac{3}{1-\theta}\left\{\mathcal{M}_{\mathbf{z}}\left(T^{\frac{1-\theta}{2}}\right)+\mathcal{F}_{\mathbf{z}}(f_*)+\mathcal{A}(f_*)\right\}+\frac{\|f_*\|_K^2}{2\eta_1}\Lambda_{T,\theta}+\widetilde{C}_1\Lambda_{T,\theta}, \tag{32}$$

where $\Lambda_{T,\theta}$ and $\widetilde{C}_1$ are defined in Lemma 17.

Proof Note that by Lemma 17, we have (27). We only need to estimate the second term of (27) denoted as

$$J_{T,\mathbf{z}}:=\frac{T^\theta}{2\eta_1}\sum_{k=1}^{T-1}\frac{1}{k+1}\left[2\eta_{T-k}-\frac{1}{k}\sum_{t=T-k+1}^{T}2\eta_t\right]\left\{\mathcal{E}_{\mathbf{z}}(f_*)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\right\}.$$

Denote $\widetilde{R}=T^{\frac{1-\theta}{2}}$. Lemma 15 tells us that $f_k\in B_{\widetilde{R}}$ for each $k=1,\cdots,T$. It follows that for $k=1,\cdots,T-1$,

$$\begin{aligned}\mathcal{E}_{\mathbf{z}}(f_*)-\mathcal{E}_{\mathbf{z}}(f_{T-k}) &= \left\{\left(\mathcal{E}_{\mathbf{z}}(f_*)-\mathcal{E}_{\mathbf{z}}(f_\rho^V)\right)-\left(\mathcal{E}(f_*)-\mathcal{E}(f_\rho^V)\right)\right\}\\&\quad+\left(\mathcal{E}(f_*)-\mathcal{E}(f_\rho^V)\right)+\mathcal{E}_{\mathbf{z}}(f_\rho^V)-\mathcal{E}_{\mathbf{z}}(f_{T-k})\\&\le \mathcal{F}_{\mathbf{z}}(f_*)+\mathcal{A}(f_*)+\mathcal{M}_{\mathbf{z}}(\widetilde{R}).\end{aligned}$$

By the choice of the stepsizes, $2\eta_{T-k}-\frac{1}{k}\sum_{t=T-k+1}^{T}2\eta_t\ge 0$ for any $k\in\{1,\dots,T-1\}$. Therefore, $J_{T,\mathbf{z}}$ can be bounded by

$$J_{T,\mathbf{z}}\le\frac{T^\theta}{2\eta_1}\sum_{k=1}^{T-1}\frac{1}{k+1}\left[2\eta_{T-k}-\frac{1}{k}\sum_{t=T-k+1}^{T}2\eta_t\right]\left\{\mathcal{F}_{\mathbf{z}}(f_*)+\mathcal{A}(f_*)+\mathcal{M}_{\mathbf{z}}(\widetilde{R})\right\}.$$

Now we need to bound the above summation. Note that, for each k,

$$2\eta_{T-k}-\frac{1}{k}\sum_{t=T-k+1}^{T}2\eta_t=\frac{2\eta_1}{k}\sum_{t=T-k+1}^{T}\left((T-k)^{-\theta}-t^{-\theta}\right).$$

Applying the mean value theorem to the function $g(x)=-x^{-\theta}$ on $[T-k,t]$ with $t\in\{T-k+1,\dots,T\}$, we find that for some $c\in[T-k,t]$,

$$(T-k)^{-\theta}-t^{-\theta}=g(t)-g(T-k)=(t-(T-k))g'(c)\le(t-(T-k))\theta(T-k)^{-\theta-1}.$$

Hence

$$\begin{aligned}&\sum_{k=1}^{T-1}\frac{1}{k+1}\left[2\eta_{T-k}-\frac{1}{k}\sum_{t=T-k+1}^{T}2\eta_t\right]\\&\le 2\eta_1\theta\sum_{k<T/2}\frac{(T-k)^{-\theta-1}}{k(k+1)}\sum_{t=T-k+1}^{T}(t-T+k)+\sum_{k\ge T/2}\frac{1}{k+1}2\eta_{T-k}\\&\le 2\eta_1\theta\sum_{k<T/2}\frac{(T-k)^{-\theta-1}}{k(k+1)}\frac{k(k+1)}{2}+\sum_{k\ge T/2}\frac{2}{T}2\eta_{T-k}\\&\le\eta_1\theta\sum_{k<T/2}(T-k)^{-\theta-1}+\frac{4\eta_1}{T}\sum_{k\ge T/2}(T-k)^{-\theta}\le\frac{6\eta_1}{1-\theta}T^{-\theta}.\end{aligned}$$

Thus

$$J_{T,\mathbf{z}} \leq \frac{3}{1-\theta}\left\{\mathcal{F}_{\mathbf{z}}(f_*) + \mathcal{A}(f_*) + \mathcal{M}_{\mathbf{z}}(\widetilde{R})\right\}.$$

Then the desired bound follows from Lemma 17. ■

4.3 Computational Errors for Weighted Average and Best Iterate

Lemma 19 *Under the assumptions of Lemma 17, let $g_T = a_T$ (or $g_T = b_T$). Then*

$$\mathcal{E}_{\mathbf{z}}(g_T) - \mathcal{E}_{\mathbf{z}}(f_*) \leq \left(\frac{2\|f_*\|_K^2}{\eta_1} + \overline{C}_1\right)\overline{\Lambda}_{T,\theta},$$

where $\overline{\Lambda}_{T,\theta}$ is given by

$$\overline{\Lambda}_{T,\theta} = \begin{cases} \frac{1}{(2+q)\theta-(q+1)}T^{-(1-\theta)}, & \text{when } \theta > \frac{q+1}{q+2}, \\ (\log T)T^{-(1-\theta)}, & \text{when } \theta = \frac{q+1}{q+2}, \\ \frac{1}{(q+1)-(2+q)\theta}T^{-(\theta(1+q)-q)}, & \text{when } \theta < \frac{q+1}{q+2}, \end{cases} \tag{33}$$

and $\overline{C}_1$ is a positive constant depending on q, κ and θ (independent of T, m or f_* and given explicitly in the proof.)

Note that there is a subtle difference between $\overline{\Lambda}_{T,\theta}$ and $\Lambda_{T,\theta}$ defined by (39), where the latter term has an extra $\log T$ for $\theta < \frac{q+1}{q+2}$.

Proof For any $u \in \mathbb{R}$, we have

$$\sum_{t=1}^T \eta_t(\mathcal{E}_{\mathbf{z}}(f_t) - u) \geq \left(\sum_{t=1}^T \eta_t\right)\min_{t=1,\ldots,T}\mathcal{E}_{\mathbf{z}}(f_t) - \left(\sum_{t=1}^T \eta_t\right)u,$$

and by convexity of $\mathcal{E}_{\mathbf{z}}$,

$$\mathcal{E}_{\mathbf{z}}(a_T) = \mathcal{E}_{\mathbf{z}}\left(\sum_{t=1}^T \omega_t f_t\right) \leq \sum_{t=1}^T \omega_t \mathcal{E}_{\mathbf{z}}(f_T) = \frac{1}{\sum_{t=1}^T \eta_t}\sum_{t=1}^T \eta_t \mathcal{E}_{\mathbf{z}}(f_t).$$

Therefor, we have

$$\mathcal{E}_{\mathbf{z}}(b_T) - u \leq \frac{1}{\sum_{t=1}^T \eta_t}\sum_{t=1}^T \eta_t(\mathcal{E}_{\mathbf{z}}(f_t) - u)$$

and

$$\mathcal{E}_{\mathbf{z}}(a_T) - u \leq \frac{1}{\sum_{t=1}^T \eta_t}\sum_{t=1}^T \eta_t(\mathcal{E}_{\mathbf{z}}(f_t) - u).$$

We thus get

$$\mathcal{E}_{\mathbf{z}}(g_T) - \mathcal{E}_{\mathbf{z}}(f^*) \leq \frac{1}{\sum_{t=1}^T \eta_t}\sum_{t=1}^T \eta_t(\mathcal{E}_{\mathbf{z}}(f_t) - \mathcal{E}_{\mathbf{z}}(f^*)). \tag{34}$$

Following Step 2 of the proof of Lemma 17, we have

$$\sum_{t=1}^T 2\eta_t\left\{\mathcal{E}_{\mathbf{z}}(f_t) - \mathcal{E}_{\mathbf{z}}(f_*)\right\}$$
$$\leq \begin{cases} \left(\|f_*\|_K^2 + C_{q,\kappa}\frac{(2+q)\theta-q}{(2+q)\theta-q-1}\right), & \text{when } \theta > \frac{q+1}{q+2}, \\ \left(\|f_*\|_K^2 + 2C_{q,\kappa}\right)(\log T), & \text{when } \theta = \frac{q+1}{q+2}, \\ \left(\|f_*\|_K^2 + C_{q,\kappa}\frac{2}{q+1-(2+q)\theta}\right)T^{(1+q)-(2+q)\theta}, & \text{when } \theta < \frac{q+1}{q+2}. \end{cases}$$

Introducing the above inequality into (34), and using $\sum_{t=1}^T \eta_t \geq \eta_1 \int_{t=1}^{T+1} u^{-\theta}du \geq \eta_1 T^{1-\theta}/\mathrm{e}$, we get our desired result with $\overline{C}_1$ given by

$$\overline{C}_1 = \begin{cases} 3\eta_1 c_q^2(\kappa+1)^{2q+2}\left((2+q)\theta - q\right), & \text{when } \theta > \frac{q+1}{q+2}, \\ 3\eta_1 c_q^2(\kappa+1)^{2q+2}, & \text{when } \theta = \frac{q+1}{q+2}, \\ 3\eta_1 c_q^2(\kappa+1)^{2q+2}, & \text{when } \theta < \frac{q+1}{q+2}. \end{cases}$$

■

While the above proof is shorter and easier than the proof of Lemma 17, it is surprising that the computational error bounds for the last iterate and the average (or the best one) are roughly of the same order.

4.4 Iterate Bound and Computational Error for Smooth Loss Functions

The following result can be proved by using the fact that $V'(y,\cdot)$ is Lipschitz. Its proof is a simple modification to RKHS of that in Nesterov (2004, Theorem 2.1.14), where the cases of Euclidean spaces are studied. For completeness, we provide the proof in Appendix A.

Lemma 20 *Assume that $V(y,\cdot)$ is differentiable and $V'(y,\cdot)$ is Lipschitz continuous with constant $L > 0$. Let $0 < \eta_t \leq (L\kappa^2)^{-1}$ for all $t \in \mathbb{N}$. Then we have*

$$\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_*) \leq \frac{\|f_*\|_K^2}{\sum_{k=1}^T 2\eta_k}.$$

In particular, if $\eta_t = \eta_1 t^{-\theta}$ with $\theta \in [0,1)$ satisfying $\eta_1 \leq (L\kappa^2)^{-1}$, then

$$\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_*) \leq \frac{2\|f_*\|_K^2}{\eta_1}T^{\theta-1}.$$

Using the above lemma, we can bound the iterates as follows.

Lemma 21 *Under the assumptions of Lemma 20, we have for $t = 1, \cdots, T$,*

$$\|f_{t+1}\|_K \leq \sqrt{2|V|_0\sum_{k=1}^t \eta_k}.$$

In particular, if $\eta_t = \eta_1 t^{-\theta}$ with $\theta \in [0,1)$ satisfying $\eta_1 \leq \frac{1-\theta}{2|V|_0}$, then

$$\|f_{t+1}\|_K \leq t^{\frac{1-\theta}{2}}.$$

Proof Choosing $f_* = 0$ in (46) (see Appendix A), we get for $k = 1, \cdots, t$,

$$\|f_{k+1}\|_K^2 \le \|f_k\|_K^2 + 2\eta_k(\mathcal{E}_{\mathbf{z}}(0) - \mathcal{E}_{\mathbf{z}}(f_{k+1})) \le \|f_k\|_K^2 + 2\eta_k|V|_0.$$

Applying this relationship iteratively for $k = t, \cdots, 1$, with $f_1 = 0$, we get

$$\|f_{t+1}\|_K^2 \le 2|V|_0 \sum_{k=1}^{t} \eta_k,$$

which leads to the first conclusion. The second inequality can be proved by noting that

$$\sum_{k=1}^{t} \eta_k = \eta_1 \sum_{k=1}^{t} k^{-\theta} \le \eta_1 \left(1 + \frac{t^{1-\theta} - 1}{1-\theta}\right) \le \eta_1 \frac{t^{1-\theta}}{1-\theta}.$$

■

5. Sample Error and Finite Sample Bounds

In this subsection, we will estimate sample errors and then prove our main results.

5.1 Sample Error

We first bound the sample error term $\mathcal{F}_{\mathbf{z}}(f_*)$ for some fixed $f_* \in \mathcal{H}_K$ as follows. This is done by applying Bernstein inequality, see Appendix B for the proof.

Lemma 22 *Assume conditions (4) and (5) hold. For any $f_* \in \mathcal{H}_K$ with $\|f_*\|_K \le R$, where $R \ge 1$, with confidence at least $1 - \frac{\delta}{2}$,*

$$\mathcal{F}_{\mathbf{z}}(f_*) \le (C_1' + 2\sqrt{c_\tau}) \log\frac{2}{\delta} \max\left\{\frac{R^{q+1}}{m}, \left(\frac{R^{2+q-\tau}}{m}\right)^{\frac{1}{2-\tau}}, \mathcal{A}(f_*)\right\}, \tag{35}$$

where C_1' is a positive constant independent of T, m, δ, given explicitly in the proof.

We next bound the empirical process over the ball $B_{\widetilde{R}}$ for some $\widetilde{R} > 0$ in the following lemma. It is essentially contained in Wu et al. (2007). We provide a short proof in Appendix B for the sake of completeness.

Lemma 23 *Assume (4) with $q \ge 0$, (5) with $\tau \in [0,1]$, (8) with $\beta \in (0,1]$ and (9) with $\zeta \in (0,2)$. Let $\widetilde{R} > 1$. Then with confidence at least $1 - \frac{\delta}{2}$, there holds for every $g \in B_{\widetilde{R}}$,*

$$\begin{aligned}
&\left(\mathcal{E}(g) - \mathcal{E}(f_\rho^V)\right) - \left(\mathcal{E}_{\mathbf{z}}(g) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)\right) \\
\le &\frac{1}{2}\left(\mathcal{E}(g) - \mathcal{E}(f_\rho^V)\right) + C_3' \log\frac{2}{\delta} \max\left\{\left(\frac{\widetilde{R}^{\frac{q(2+\zeta)+(4-2\tau+\zeta\tau)}{2}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}}, \frac{\widetilde{R}^{q+1}}{m^{\frac{2}{2+\zeta}}}, \left(\frac{\widetilde{R}^{2+q-\tau}}{m}\right)^{\frac{1}{2-\tau}}\right\},
\end{aligned} \tag{36}$$

and

$$\mathcal{M}_{\mathbf{z}}(\widetilde{R}) \le C_3' \log\frac{2}{\delta} \max\left\{\left(\frac{\widetilde{R}^{\frac{q(2+\zeta)+(4-2\tau+\zeta\tau)}{2}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}}, \frac{\widetilde{R}^{q+1}}{m^{\frac{2}{2+\zeta}}}, \left(\frac{\widetilde{R}^{2+q-\tau}}{m}\right)^{\frac{1}{2-\tau}}\right\}. \tag{37}$$

Here C_3' is a positive constant independent of T, m, δ, given explicitly in the proof.

5.2 Deriving the Finite Sample Bounds

We have the following result, which will be used for the proof of Theorem 4.

Proposition 24 *Assume (4) with $q \ge 0$, (5) with $\tau \in [0,1]$, (8) with $\beta \in (0,1]$ and (9) with $\zeta \in (0,2)$. Let $\eta_t = \eta_1 t^{-\theta}$ with $0 < \theta < 1$ satisfying $\theta > \frac{q}{q+1}$ and η_1 satisfying (10). Let $f_* \in \mathcal{H}_K$ be such that $\|f_*\|_K \le R$, where $R \ge 1$. If $1 \le R \le T^{\frac{1-\theta}{2}}$ and $T^{\frac{q(1-\theta)}{2}} m^{-\frac{2}{2+\zeta}} \le 1$, then with confidence $1 - \delta$, we have*

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \le \widetilde{C}_3 \log\frac{2}{\delta} \max\left\{\left(\frac{T^{\frac{(1-\theta)(q(2+\zeta)+(4-2\tau+\zeta\tau))}{4}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}}, R^2\Lambda_T, \mathcal{A}(f_*)\right\}, \tag{38}$$

where Λ_T is defined by

$$\Lambda_T = \begin{cases} T^{-(1-\theta)}, & \text{when } \theta > \frac{q+1}{q+2}, \\ (\log T) T^{-(1-\theta)}, & \text{when } \theta = \frac{q+1}{q+2}, \\ (\log T) T^{-(\theta(1+q)-q)}, & \text{when } \theta < \frac{q+1}{q+2}, \end{cases} \tag{39}$$

and $\widetilde{C}_3$ is a positive constant independent of T, m, δ, given explicitly in the proof.

Proof Recall Lemma 13. Let $\tilde{R} = T^{\frac{1-\theta}{2}}$. Introducing with (32), we have

$$\begin{aligned}
\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \le &\left\{\left(\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)\right) - \left(\mathcal{E}_{\mathbf{z}}(f_T) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)\right)\right\} + \frac{3}{1-\theta}\mathcal{M}_{\mathbf{z}}\left(\tilde{R}\right) \\
&+ \frac{4-\theta}{1-\theta}\left(\mathcal{F}_{\mathbf{z}}(f_*) + \mathcal{A}(f_*)\right) + \frac{R^2}{2\eta_1}\Lambda_{T,\theta} + \widetilde{C}_1\Lambda_{T,\theta}.
\end{aligned}$$

Applying Lemmas 22 and 23 with $g = f_T$, with $R \in [1, \widetilde{R}]$ and the notation Λ_T defined by (39), we know that with confidence at least $1 - \delta$,

$$\begin{aligned}
\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \le\ & C_4' \log\frac{2}{\delta} \max\left\{\left(\frac{\widetilde{R}^{\frac{q(2+\zeta)+(4-2\tau+\zeta\tau)}{2}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}}, \frac{\widetilde{R}^{q+1}}{m^{\frac{2}{2+\zeta}}}, \right. \\
&\left. \frac{\widetilde{R}^{\frac{2+q-\tau}{2-\tau}}}{m^{\frac{1}{2-\tau}}}, R^2\Lambda_T, \mathcal{A}(f_*)\right\} + \frac{1}{2}\left(\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)\right),
\end{aligned} \tag{40}$$

where C_4' is the constant given by

$$C_4' = \frac{4-\theta}{1-\theta}C_3' + \frac{4-\theta}{1-\theta}\left(1 + C_1' + 2\sqrt{c_\tau}\right) + \left(\frac{1}{2\eta_1} + \widetilde{C}_1\right) c_{\theta,q}.$$

Here $c_{\theta,q}$ is given by

$$c_{\theta,q} = \begin{cases} \frac{1}{|(q+2)\theta-(q+1)|}, & \text{when } \theta \neq \frac{q+1}{q+2}, \\ 1, & \text{when } \theta = \frac{q+1}{q+2}. \end{cases}$$

Since $\widetilde{R}^q m^{-2/(2+\zeta)} \leq 1$, $\tau \in [0,1]$ and $\zeta \in (0,2)$, one finds

$$\left(\frac{\widetilde{R}^{\frac{q(2+\zeta)+(4-2\tau+\zeta\tau)}{2}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}} \cdot \frac{m^{\frac{2}{2+\zeta}}}{\widetilde{R}^{q+1}} = \left\{\frac{\widetilde{R}^q}{m^{\frac{2}{2+\zeta}}}\right\}^{\frac{-(1-\tau)(2-\zeta)}{4-2\tau+\zeta\tau}} \geq 1,$$

and

$$\left(\frac{\widetilde{R}^{\frac{q(2+\zeta)+(4-2\tau+\zeta\tau)}{2}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}} \cdot \frac{m^{\frac{1}{2-\tau}}}{\widetilde{R}^{\frac{2+q-\tau}{2-\tau}}} = \left(\widetilde{R}^{2q(1-\tau)} m^{\tau}\right)^{\frac{\zeta}{(2-\tau)(4-2\tau+\zeta\tau)}} \geq 1.$$

Subtracting $\frac{1}{2}\left(\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V)\right)$ from both sides of (40), and setting $\widetilde{C}_3 = 2C_4'$, we get the desired results. ■

Now we are in a position to prove the probabilistic upper bounds stated in Theorem 4.

Proof of Theorem 4 We use Proposition 24 with $f_* = f_\lambda$ to prove our result. Define a power index $\tilde{\theta}$ as

$$\tilde{\theta} = \begin{cases} 1-\theta, & \text{when } \theta \geq \frac{q+1}{q+2}, \\ \theta(1+q)-q, & \text{when } \theta < \frac{q+1}{q+2}. \end{cases} \tag{41}$$

Comparing this with the definition (39) for Λ_T, we see that

$$\Lambda_T = \begin{cases} T^{-\tilde{\theta}}, & \text{when } \theta > \frac{q+1}{q+2}, \\ T^{-\tilde{\theta}} \log T, & \text{when } \theta \leq \frac{q+1}{q+2}. \end{cases}$$

From the definition of $\mathcal{D}(\lambda)$, we have

$$\mathcal{A}(f_\lambda) \leq \mathcal{D}(\lambda) \qquad \text{and} \qquad \lambda \|f_\lambda\|_K^2 \leq \mathcal{D}(\lambda), \tag{42}$$

which imply $\|f_\lambda\|_K \leq \sqrt{\mathcal{D}(\lambda)/\lambda} = R$. Balancing the orders of the last two terms of (38) by setting

$$\lambda = \Lambda_T, \tag{43}$$

we find that the last two terms of (38) can be bounded as

$$\max\left\{R^2\Lambda_T,\ \mathcal{A}(f_\lambda)\right\} \leq \mathcal{D}(\lambda) \leq c_\beta \lambda^\beta \leq \begin{cases} c_\beta T^{-\beta\tilde{\theta}}, & \text{when } \theta > \frac{q+1}{q+2}, \\ c_\beta T^{-\beta\tilde{\theta}} \log T, & \text{when } \theta \leq \frac{q+1}{q+2}. \end{cases}$$

Then we balance the above main part with the first term of (38) by setting

$$\left(\frac{T^{\frac{(1-\theta)(q(2+\zeta)+(4-2\tau+\zeta\tau))}{4}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}} = T^{-\beta\tilde{\theta}}.$$

This leads us to choose T to be the integer part of

$$\lceil m^\gamma \rceil, \text{ where } \gamma := \frac{2}{\left(\frac{1-\theta}{2} + \beta\tilde{\theta}\right)(4-2\tau+\zeta\tau) + \frac{q(2+\zeta)(1-\theta)}{2}}. \tag{44}$$

With this choice, the main part of (38) can be bounded as

$$\max\left\{\left(\frac{T^{\frac{(1-\theta)(q(2+\zeta)+(4-2\tau+\zeta\tau))}{4}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}},\ R^2\Lambda_T,\ \mathcal{A}(f_*)\right\} \leq \begin{cases} 2c_\beta m^{-\beta\tilde{\theta}\gamma}, & \text{when } \theta > \frac{q+1}{q+2}, \\ 2\gamma c_\beta m^{-\beta\tilde{\theta}\gamma} \log m, & \text{when } \theta \leq \frac{q+1}{q+2}. \end{cases}$$

Noticing from the definition of $\tilde{\theta}$, one can easily prove that $\tilde{\theta} \leq 1-\theta$. Then $R/\sqrt{c_\beta} \leq \sqrt{\lambda^{\beta-1}} \leq \Lambda_T^{\frac{\beta-1}{2}} \leq T^{\frac{1-\theta}{2}}$ and the restriction for R in Theorem 24 is satisfied up to constants. The restriction $T^{\frac{q(1-\theta)}{2}} m^{-\frac{2}{2+\zeta}} \leq 1$ is also satisfied (up to a constant), because

$$T^{\frac{q(1-\theta)}{2}} \lesssim m^{\frac{q(1-\theta)\gamma}{2}} \lesssim m^{\frac{2}{2+\zeta}}.$$

Observe that $\gamma \leq \frac{2}{1-\theta}$. So by Proposition 24, with confidence $1-\delta$, we have

$$\mathcal{E}(f_T) - \mathcal{E}(f_\rho^V) \leq \begin{cases} 2c_\beta \widetilde{C}_3 m^{-\beta\tilde{\theta}\gamma} \log\frac{2}{\delta}, & \text{when } \theta > \frac{q+1}{q+2}, \\ \frac{4c_\beta\widetilde{C}_3}{1-\theta} m^{-\beta\tilde{\theta}\gamma} \log m \log\frac{2}{\delta}, & \text{when } \theta \leq \frac{q+1}{q+2}. \end{cases}$$

Observe that the power index $\beta\tilde{\theta}\gamma$ is

$$\beta\tilde{\theta}\gamma = \begin{cases} \frac{\beta}{\beta(2-\tau+\zeta\tau/2)+\left\{\frac{2-\tau+\zeta\tau/2}{2} + \frac{q(1+\zeta/2)}{2}\right\}}, & \text{when } \theta \geq \frac{q+1}{q+2}, \\ \frac{\beta}{\beta(2-\tau+\zeta\tau/2)+\frac{1-\theta}{\theta(1+q)-q}\left\{\frac{2-\tau+\zeta\tau/2}{2} + \frac{q(1+\zeta/2)}{2}\right\}}, & \text{when } \theta < \frac{q+1}{q+2}, \end{cases}$$

while the index γ can be expressed by (11). Then our desired learning rates are verified by setting the constant $\widetilde{C} = 2c_\beta\widetilde{C}_3$ when $\theta > \frac{q+1}{q+2}$ while $\widetilde{C} = \frac{4c_\beta\widetilde{C}_3}{1-\theta}$ when $\theta \leq \frac{q+1}{q+2}$. The proof of Theorem 4 is complete. ■

Proof of Theorem 7 We only sketch the proof for the case $g_T = a_T$. By applying Lemma 15, it is easy to prove that

$$\|a_T\|_K \leq T^{\frac{1-\theta}{2}}.$$

With the upper bound on a_T and Lemma 19, a similar argument as that for Theorem 4, one can prove the results. We omit the details. ■

Proof of Theorem 8 With Lemmas 20, 21, and a similar approach as that for Theorem 4, we can prove the convergence results for smooth loss functions. We omit the details. ■

Proof of Theorem 11 We use Theorem 4 to prove the results. The hinge loss satisfies (4) with $q = 0$ and $c_q = \frac{1}{2}$, $|V|_0 = 1$ and $\|f_\rho^V\|_\infty = 1$ where f_ρ^V is the Bayes rule f_c. Condition (5) is valid with $\tau = 0$ and $c_\tau = 1$. Since $\theta > 1/2$, by simple calculations, one finds that $\gamma = \frac{1}{(1-\theta)(2\beta+1)}$ and $\alpha = \frac{\beta}{2\beta+1}$.

Using the comparison theorem from Zhang (2004), we have

$$\mathcal{R}\left(\text{sign}(f_T)\right) - \mathcal{R}(f_c) \leq \mathcal{E}(f_T) - \mathcal{E}(f_\rho^V).$$

So the desired probabilistic upper bound (14) for the hinge loss follows from the above inequality and Theorem 4.

It remains to prove the second part of the theorem. Since $0 < \epsilon < \frac{1}{3}$, the restriction $\beta > \frac{1-3\epsilon}{1+6\epsilon}$ for the approximation order tells us that

$$\alpha = \frac{\beta}{2\beta+1} = \frac{1}{2+1/\beta} \geq \frac{1}{3} - \epsilon.$$

The proof of Theorem 11 is complete. ■

Proof of Theorem 12 Since $0 < \epsilon < \frac{1}{3}$, the restriction $\beta > \frac{4-3\epsilon}{4+6\epsilon}$ for the approximation order tells us that the parameter θ satisfies $\frac{1}{2} < \theta < 1$ and the index

$$\gamma = \frac{1}{(1-\theta)(2\beta+1)} = \frac{2}{3} + \epsilon.$$

Finally we find that the index

$$\alpha = \frac{\beta}{2\beta+1} = \frac{1}{2+1/\beta} \geq \frac{1}{3} - \frac{\epsilon}{4}.$$

So the desired probabilistic upper bound follows from the first conclusion of Theorem 11. The proof of Theorem 12 is complete. ■

6. Conclusions

This paper proposes and studies iterative regularization approaches for learning with convex loss functions. More precisely, we study how regularization can be achieved by early stopping an empirical iteration induced by the subgradient method, or gradient descent in the case that the loss is also smooth. Finite sample bounds are established providing indications on how to suitably choose the stepsize and the stopping rule. Different to classical results on the subgradient method, we analyze the behavior of the last iterate showing that it has essentially the same properties of the average, to the best, iterate. These results provide a theoretical foundation for early stopping with convex losses.

Beyond the analysis in the paper our error decomposition provides an approach to incorporating statistics and optimization aspects in the analysis of learning algorithms. While a natural development will be to sharpen the bounds and perform extensive empirical tests, we hope the study in the paper can help deriving novel and faster algorithms, for example analyzing accelerations (Nesterov, 2004), or distributed approaches, within the framework we propose.

Acknowledgments

The work described in this paper is supported partially by the Research Grants Council of Hong Kong [Project No. CityU 104012] and by National Natural Science Foundation of China under Grant 11461161006. LR is supported by the FIRB project RBFR12M3AC and the Center for Minds, Brains and Machines (CBMM), funded by NSF STC award CCF-1231216. JL is now within LCSL, MIT & Istituto Italiano di Tecnologia. The authors would like to thank the referees and Dr. Yunlong Feng for their valuable comments.

References

A. Aizerman, E. M. Braverman and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical society*, 68:337–404, 1950.

P. Bartlett and M. Traskin. Adaboost is consistent. *Journal of Machine Learning Research*, 8:2347–2368, 2007.

F. Bauer, S. Pereverzev and L. Rosasco. On regularization algorithms in learning theory. *Journal of Complexity*, 1:52–72, 2007.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

P. J. Bickel, Y. Ritov and A. Zakai. Some theory for generalized boosting algorithms. *Journal of Machine Learning Research*, 7:705–732, 2006.

G. Blanchard and N. Krämer. Optimal learning rates for kernel conjugate gradient regression. *Advances in Neural Information Processing Systems*, 226–234, 2010.

O. Bousquet and L. Bottou. The tradeoffs of large scale learning. *Advances in Neural Information Processing Systems*, 161–168, 2008.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd, L. Xiao and A. Mutapcic. *Subgradient Methods*. Lecture notes of EE392o, Stanford University, Autumn Quarter, 2004 (2003).

P. Buhlmann and B. Yu. Boosting with the L_2 loss: regression and classification. *Journal of the American Statistical Association*, 462:324–339, 2003.

A. Caponnetto and E. De Vito. Optimal rates for regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7:331–368, 2007.

A. Caponnetto and Y. Yao. Cross-validation based adaptation for regularization operators in learning theory. *Analysis and Applications*, 8:161–183, 2010.

V. Chandrasekaran and M. I. Jordan. Computational and statistical tradeoffs via convex relaxation. *Proceedings of the National Academy of Sciences*, 110:E1181–E1190, 2013.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

F. Cucker and D.-X. Zhou. *Learning Theory: An Approximation Theory Viewpoint*. Cambridge University Press, 2007.

E. De Vito, L. Rosasco, A. Caponnetto, M. Piana and A. Verri. Some properties of regularized kernel methods. *Journal of Machine Learning Research*, 5:1363–1390, 2004.

E. De Vito, L. Rosasco, A. Caponnetto, U. Giovannini and F. Odone. Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6:1532–4435, 2005.

H. W. Engl, M. Hanke and A. Neubauer. *Regularization of Inverse Problems*. Kluwer, 1996.

L. Gerfo, L. Rosasco, F. Odone, E. De Vito and A. Verri. Spectral algorithms for supervised learning. *Neural Computation*, 20:1873–1897, 2008.

T. Hu, J. Fan, Q. Wu and D.-X. Zhou. Regularization schemes for minimum error entropy principle. *Analysis and Applications*, 13:437–455, 2015.

T. Jaakkola, M. Diekhaus and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, 99:149–158, 1999.

W. Jiang. Process consistency for adaboost. *Annals of Statistics*, 32:13–29, 2004.

B. Kaltenbacher, A. Neubauer and O. Scherzer. *Iterative Regularization Methods for Nonlinear Ill-posed Problems*. Radon Series on Computational and Applied Mathematics, de Gruyter, Berlin, 2008.

Y. LeCun, L. Bottou, G. Orr and K. Muller. *Efficient Backprop*. Neural Networks: Tricks of the Trade, Springer, 1998.

A. Nemirovskii. The regularization properties of adjoint gradient method in ill-posed problems.*USSR Computational Mathematics and Mathematical Physics*, 26:7–16, 1986.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

F. Orabona. Simultaneous model selection and optimization through parameter-free stochastic learning. *Advances in Neural Information Processing Systems*, 1116–1124, 2014.

B. Polyak. *Introduction to Optimization*. Optimization Software, 1987.

G. Raskutti, M. J. Wainwright and B. Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *Journal of Machine Learning Research*, 15:335–366, 2014.

L. Rosasco and S. Villa. Learning with incremental iterative regularization. *Advances in Neural Information Processing Systems*, 1621–1629, 2015.

F. Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington DC: Spartan Books, 1962.

O. Shamir and T. Zhang. Stochastic gradient descent for non-smooth optimization: convergence results and optimal averaging schemes. *International Conference on Machine Learning*, 71–79, 2013.

S. Smale and D.-X. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1:17–41, 2003.

S. Sra, S. Nowozin and S. J. Wright. *Optimization for Machine Learning, Neural Information Processing Series*. MIT Press, 2011.

I. Steinwart. Oracle inequailities for support vector machines that are based on random entropy numbers. *Journal of Complexity*, 25:437–454, 2009.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, 2008.

A. N. Tikhonov and V. Y. Arsenin. *Solution of Ill-posed Problems*. Winston & Sons, 1977.

V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

M. J. Wainwright. Structured regularizers for high-dimensional problems: statistical and computational issues. *Annual Review of Statistics and Its Application*, 1:233–253, 2014.

Q. Wu, Y. Ying and D.-X. Zhou. Multi-kernel regularized classifiers. *Journal of Complexity*, 23:108–134, 2007.

Y. Yao, L. Rosasco and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26:289–315, 2007.

G. B. Ye and D.-X. Zhou. Fully online classification by regularization. *Applied and Computational Harmonic Analysis*, 23:198–214, 2007.

Y. Ying and D.-X. Zhou. Online regularized classification algorithms. *IEEE Transaction on Information Theory*, 52:4775–4788, 2006.

T. Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics*, 32:56-85, 2004.

T. Zhang and B. Yu. Boosting with early stopping: convergence and consistency. *Annals of Statistics*, 33:1538–1579, 2005.

D.-X. Zhou. Capacity of reproducing kernel spaces in learning theory. *IEEE Transaction on Information Theory*, 49:1743–1752, 2003.

Appendix A. Proof of Lemma 20

Proof Since $V'(y,\cdot)$ is Lipschitz with constant L for any $y \in Y$, we have for any $a, b \in \mathbb{R}$,

$$V(y,b) \le V(y,a) + V'(y,a)(b-a) + \frac{L}{2}(b-a)^2.$$

Choosing $y = y_j$, $b = f_{t+1}(x_j)$ and $a = f_t(x_j)$, according to the properties (22) and (23), we get for $j = 1, \cdots, m$ and $t \in \mathbb{N}$,

$$V(y_j, f_{t+1}(x_j)) \le V(y_j, f_t(x_j)) + V'(y_j, f_t(x_j))\langle f_{t+1} - f_t, K_{x_j}\rangle_K + \frac{L\kappa^2}{2}\|f_{t+1} - f_t\|_K^2.$$

Summing up over $j = 1, \cdots, m$, with $g_t = \frac{1}{m}\sum_{j=1}^m V'(y_j, f_t(x_j))K_{x_j}$, we get

$$\mathcal{E}_{\mathbf{z}}(f_{t+1}) \le \mathcal{E}_{\mathbf{z}}(f_t) + \langle f_{t+1} - f_t, g_t\rangle_K + \frac{L\kappa^2}{2}\|f_{t+1} - f_t\|_K^2.$$

Introducing with (1) and noting that $\eta_t \le (L\kappa^2)^{-1}$, we get

$$\mathcal{E}_{\mathbf{z}}(f_{t+1}) \le \mathcal{E}_{\mathbf{z}}(f_t) - \frac{\eta_t}{2}\|g_t\|_K^2. \tag{45}$$

By the convexity of $V(y,\cdot)$, it is easy to prove that

$$\mathcal{E}_{\mathbf{z}}(f_t) \le \mathcal{E}_{\mathbf{z}}(f_*) + \langle f_t - f_*, g_t\rangle_K.$$

Introducing this inequality into (45), we get

$$\begin{aligned}
\mathcal{E}_{\mathbf{z}}(f_{t+1}) &\le \mathcal{E}_{\mathbf{z}}(f_*) + \frac{1}{2\eta_t}\left(2\eta_t\langle f_t - f_*, g_t\rangle_K - \eta_t^2\|g_t\|_K^2\right) \\
&= \mathcal{E}_{\mathbf{z}}(f_*) + \frac{1}{2\eta_t}\left(\|f_t - f_*\|_K^2 - \|f_t - f_* - \eta_t g_t\|_K^2\right) \\
&= \mathcal{E}_{\mathbf{z}}(f_*) + \frac{1}{2\eta_t}\left(\|f_t - f_*\|_K^2 - \|f_{t+1} - f_*\|_K^2\right),
\end{aligned}$$

so that,

$$2\eta_t(\mathcal{E}_{\mathbf{z}}(f_{t+1}) - \mathcal{E}_{\mathbf{z}}(f_*)) \le \|f_t - f_*\|_K^2 - \|f_{t+1} - f_*\|_K^2. \tag{46}$$

Summing up over $t = 1 \cdots, T$, with $f_1 = 0$, we have

$$\sum_{t=1}^T 2\eta_t(\mathcal{E}_{\mathbf{z}}(f_{t+1}) - \mathcal{E}_{\mathbf{z}}(f_*)) \le \|f_1 - f_*\|_K^2 - \|f_{T+1} - f_*\|_K^2 \le \|f_*\|_K^2.$$

By (45), we have $\mathcal{E}_{\mathbf{z}}(f_{T+1}) \le \mathcal{E}_{\mathbf{z}}(f_{t+1})$ for all $t \le T$. It thus follows that

$$\sum_{t=1}^T 2\eta_t(\mathcal{E}_{\mathbf{z}}(f_{T+1}) - \mathcal{E}_{\mathbf{z}}(f_*)) \le \sum_{t=1}^T 2\eta_t(\mathcal{E}_{\mathbf{z}}(f_{t+1}) - \mathcal{E}_{\mathbf{z}}(f_*)) \le \|f_*\|_K^2,$$

which leads to the first argument of the lemma. The rest of the proof can be finished by noting that

$$\sum_{t=1}^T \eta_t \ge \eta_1 \int_1^{T+1} u^{-\theta} du \ge \eta_1 \frac{T^{1-\theta}}{\mathrm{e}}.$$

■

Appendix B. Proofs for the Sample Error

Proof of Lemma 22 We apply Bernstein inequality which asserts that, for a random variable ξ bounded by $\widetilde{M} > 0$ and for any $\epsilon > 0$,

$$\mathrm{Prob}\left\{\frac{1}{m}\sum_{i=1}^m \xi(z_i) - \mathbb{E}(\xi) > \epsilon\right\} \le \exp\left\{-\frac{m\epsilon^2}{2\left(\sigma^2(\xi) + \frac{1}{3}\widetilde{M}\epsilon\right)}\right\}. \tag{47}$$

Here the random variable ξ on Z is given by $\xi(x,y) = V(y, f_*(x)) - V(y, f_\rho^V(x))$. The increment condition (4) implies that ξ is bounded by $M := C_1' R^{q+1}$, where C_1' is the constant given by

$$C_1' := c_q\left(\kappa + \kappa^{q+1} + \|f_\rho^V\|_\infty + \|f_\rho^V\|_\infty^{q+1}\right). \tag{48}$$

By condition (5), its variance $\sigma^2(\xi)$ is bounded by

$$c_\tau R^{2+q-\tau}\left\{\mathcal{E}(f_*) - \mathcal{E}(f_\rho^V)\right\}^\tau \le c_\tau R^{2+q-\tau}(\mathcal{A}(f_*))^\tau.$$

Solving the quadratic equation from Bernstein inequality (47), we see that with confidence at least $1 - \frac{\delta}{2}$, there holds

$$\begin{aligned}
\mathcal{F}_{\mathbf{z}}(f_*) &\le \frac{2M\log\frac{2}{\delta}}{3m} + \sqrt{\frac{2\log\frac{2}{\delta}}{m}\sigma^2(\xi)} \\
&\le (C_1' + 2\sqrt{c_\tau})\log\frac{2}{\delta}\max\left\{\frac{R^{q+1}}{m},\ \frac{R^{1+\frac{q-\tau}{2}}(\mathcal{A}(f_*))^{\frac{\tau}{2}}}{\sqrt{m}}\right\}.
\end{aligned}$$

Applying the elementary inequality

$$x^\tau y^{1-\tau} \le \tau x + (1-\tau)y, \qquad \forall\, \tau \in [0,1],\ x, y \ge 0, \tag{49}$$

yields

$$\begin{aligned}
\frac{R^{1+\frac{q-\tau}{2}}(\mathcal{A}(f_*))^{\frac{\tau}{2}}}{\sqrt{m}} &= \left[\left(\frac{R^{2+q-\tau}}{m}\right)^{\frac{1}{2-\tau}}\right]^{1-\frac{\tau}{2}}(\mathcal{A}(f_*))^{\frac{\tau}{2}} \\
&\le \left(1 - \frac{\tau}{2}\right)\left(\frac{R^{2+q-\tau}}{m}\right)^{\frac{1}{2-\tau}} + \frac{\tau}{2}\mathcal{A}(f_*).
\end{aligned}$$

Then the desired result follows. ■

To bound the empirical process over the ball $B_{\widetilde{R}}$ for some $\widetilde{R} > 0$, we need the following concentration inequality. Its proof is similar to that of Proposition 6 in Wu et al. (2007), as well as applying Theorem 3.5 from Steinwart (2009) and Exercise 6.8 in Steinwart and Christmann (2008). We omit the proof.

Lemma 25 *Let $\mathcal{G}$ be a set of measurable functions on $\mathcal{Z}$, and $B, c > 0, \tau \in [0,1]$ be constants such that each function $f \in \mathcal{G}$ satisfies $\|f\|_\infty \le B$ and $\mathbb{E}(f^2) \le c(\mathbb{E}f)^\tau$. If for some $a \ge B^\zeta$ and $\zeta \in (0,2)$,*

$$\mathbb{E}_{\mathbf{z}}[\log\mathcal{N}(\mathcal{G}, \epsilon, d_{2,\mathbf{z}})] \le a\epsilon^{-\zeta}, \qquad \forall \epsilon > 0,$$

then there exists a positive constant c'_ζ depending only on ζ such that for any $b>0$, with probability at least $1-\mathrm{e}^{-b}$, there holds

$$\mathbb{E}f - \frac{1}{m}\sum_{i=1}^m f(z_i) \le \frac{1}{2}\eta^{1-\tau}(\mathbb{E}f)^\tau + c'_\zeta\eta + 2\Big(\frac{cb}{m}\Big)^{1/(2-\tau)} + \frac{18Bb}{m}, \qquad \forall f\in\mathcal{G},$$

where

$$\eta := \max\left\{c^{\frac{2-\zeta}{4-2\tau+\zeta\tau}}\Big(\frac{a}{m}\Big)^{\frac{2}{4-2\tau+\zeta\tau}},\ B^{\frac{2-\zeta}{2+\zeta}}\Big(\frac{a}{m}\Big)^{\frac{2}{2+\zeta}}\right\}.$$

Now, we are ready to prove Lemma 23.

Proof of Lemma 23 We first apply Lemma 25 to the function set

$$\mathcal{G} = \left\{f(x,y) = V(y,g(x)) - V(y,f_\rho^V(x)) :\ g\in B_{\widetilde{R}}\right\}.$$

Condition (5) tells us that with $c=c_\tau\widetilde{R}^{2+q-\tau}$, each function $f\in\mathcal{G}$ satisfies $\mathbb{E}(f^2)\le c(\mathbb{E}f)^\tau$. Also, condition (4) implies that $\|f\|_\infty$ is upper bounded by $\widetilde{M} := C'_1\widetilde{R}^{q+1}$, with C'_1 given by (48). Noticing from (4) that for $f,f'\in\mathcal{G}$,

$$|f(x,y)-f'(x,y)| = |V(y,g(x))-V(y,g'(x))| \le c_q(1+\kappa^q)\widetilde{R}^q|g(x)-g'(x)|,$$

there holds

$$\mathcal{N}(\mathcal{G},\epsilon,d_{2,\mathbf{z}}) \le \mathcal{N}\left(B_{\widetilde{R}}, \frac{\epsilon}{c_q(1+\kappa^q)\widetilde{R}^q}, d_{2,\mathbf{z}}\right) \le \mathcal{N}\left(B_1, \frac{\epsilon}{c_q(1+\kappa^q)\widetilde{R}^{q+1}}, d_{2,\mathbf{z}}\right).$$

Hence, condition (9) yields the covering number condition in Lemma 25 with $a=C''\widetilde{R}^{(q+1)\zeta}$, where

$$C'' = \max\{c_\zeta c_q^\zeta(1+\kappa^q)^\zeta, (C'_1)^\zeta\}$$

So we apply Lemma 25 and find that with confidence at least $1-\frac{\delta}{2}$, there holds for every $f\in\mathcal{G}$,

$$\mathbb{E}(f) - \frac{1}{m}\sum_{i=1}^m f(z_i) \le \frac{1}{2}\eta^{1-\tau}(\mathbb{E}f)^\tau + c'_\zeta\eta + 2\Big(\frac{c_\tau\widetilde{R}^{2+q-\tau}\log\frac{2}{\delta}}{m}\Big)^{\frac{1}{2-\tau}} + \frac{18\widetilde{M}\log\frac{2}{\delta}}{m},$$

where

$$\begin{aligned}\eta &= \max\left\{\left(c_\tau\widetilde{R}^{2+q-\tau}\right)^{\frac{2-\zeta}{4-2\tau+\zeta\tau}}\left(\frac{C''\widetilde{R}^{(q+1)\zeta}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}},\right.\\ &\qquad \left.\widetilde{M}^{\frac{2-\zeta}{2+\zeta}}\left(\frac{C''\widetilde{R}^{(q+1)\zeta}}{m}\right)^{\frac{2}{2+\zeta}}\right\}\\ &\le C'_2\max\left\{\left(\frac{\widetilde{R}^{\frac{q(2+\zeta)+(4-2\tau+\zeta\tau)}{2}}}{m}\right)^{\frac{2}{4-2\tau+\zeta\tau}}, \frac{\widetilde{R}^{q+1}}{m^{\frac{2}{2+\zeta}}}\right\},\end{aligned}$$

where C'_2 is the constant given by

$$\left(c_\tau^{\frac{2-\zeta}{2}}C''\right)^{\frac{2}{4-2\tau+\zeta\tau}} + C_1'^{\frac{2-\zeta}{2+\zeta}}\left(C''\right)^{\frac{2}{2+\zeta}}.$$

Apply the elementary inequality (49) which yields $\eta^{1-\tau}(\mathbb{E}f)^\tau \le \eta + \mathbb{E}f$, and notice that $\mathbb{E}(f) = \mathcal{E}(g) - \mathcal{E}(f_\rho^V)$ while $\frac{1}{m}\sum_{i=1}^m f(z_i) = \mathcal{E}_{\mathbf{z}}(g) - \mathcal{E}_{\mathbf{z}}(f_\rho^V)$. We get that with confidence at least $1-\frac{\delta}{2}$, there holds for every $g\in B_{\widetilde{R}}$,

$$\begin{aligned}&\left(\mathcal{E}(g)-\mathcal{E}(f_\rho^V)\right) - \left(\mathcal{E}_{\mathbf{z}}(g)-\mathcal{E}_{\mathbf{z}}(f_\rho^V)\right)\\ &\le \left(\frac{1}{2}+c'_\zeta\right)\eta + \frac{1}{2}\left(\mathcal{E}(g)-\mathcal{E}(f_\rho^V)\right) + 2\Big(\frac{c_\tau\widetilde{R}^{2+q-\tau}}{m}\Big)^{\frac{1}{2-\tau}}\log\frac{2}{\delta} + \frac{18\widetilde{M}\log\frac{2}{\delta}}{m},\end{aligned}$$

which leads to (36) with

$$C'_3 = \left(\frac{1}{2}+c'_\zeta\right)C'_2 + 2c_\tau^{\frac{1}{2-\tau}} + 18\widetilde{M}.$$

Now, introducing (36) into the equality

$$\mathcal{E}_{\mathbf{z}}(f_\rho^V) - \mathcal{E}_{\mathbf{z}}(g) = \left\{\left(\mathcal{E}(g)-\mathcal{E}(f_\rho^V)\right) - \left(\mathcal{E}_{\mathbf{z}}(g)-\mathcal{E}_{\mathbf{z}}(f_\rho^V)\right)\right\} - \left(\mathcal{E}(g)-\mathcal{E}(f_\rho^V)\right),$$

with $\mathcal{E}(g)-\mathcal{E}(f_\rho^V)\ge 0$ and by recalling that $\mathcal{M}_{\mathbf{z}}(\widetilde{R})$ is given by (31), we can derive (37). The proof is completed. ■

Journal of Machine Learning Research 17 (2016) 1-41 Submitted 4/15; Published 4/16

Latent Space Inference of Internet-Scale Networks

Qirong Ho* HOQIRONG@GMAIL.COM
Institute for Infocomm Research
*A*STAR*
Singapore 138632

Junming Yin* JUNMINGY@EMAIL.ARIZONA.EDU
Department of Management Information Systems
Eller College of Management
University of Arizona
Tucson, AZ 85721

Eric P. Xing EPXING@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Editor: Jure Leskovec

Abstract

The rise of Internet-scale networks, such as web graphs and social media with hundreds of millions to billions of nodes, presents new scientific opportunities, such as overlapping community detection to discover the structure of the Internet, or to analyze trends in online social behavior. However, many existing probabilistic network models are difficult or impossible to deploy at these massive scales. We propose a scalable approach for modeling and inferring latent spaces in Internet-scale networks, with an eye towards overlapping community detection as a key application. By applying a succinct representation of networks as a bag of triangular motifs, developing a parsimonious statistical model, deriving an efficient stochastic variational inference algorithm, and implementing it as a distributed cluster program via the Petuum parameter server system, we demonstrate overlapping community detection on real networks with up to 100 million nodes and 1000 communities on 5 machines in under 40 hours. Compared to other state-of-the-art probabilistic network approaches, our method is several orders of magnitude faster, with competitive or improved accuracy at overlapping community detection.

Keywords: probabilistic network models, triangular modeling, stochastic variational inference, distributed computation, big data

1. Introduction

The rapid growth of the Internet, particularly the explosion of social media, has led to unprecedented increases in the volume of network data worldwide. Already, the Yahoo web graph collected in 2002 contains in excess of one billion URLs (Yahoo, 2013), the Facebook social network recently exceeded one billion users (Facebook, 2013), and numerous other social networks or online communities easily claim memberships in the millions of users (Leskovec and Krevl, 2014). To understand the structural and functional properties of massive networks, one important task in network analysis is unsupervised detection of community structure—a task that frequently occurs in social media and Internet studies (Palla et al., 2005; Newman, 2006; Prakash et al., 2010; Psorakis et al., 2011; Xie and Szymanski, 2012; Gopalan and Blei, 2013). Although there is no clear consensus in the literature on the definition of a network community, it is generally accepted that actors or nodes from the same community tend to interact more frequently with each other and share more common characteristics. In a social network, relevant characteristics might be "(people) going to the same school, social club, or workplace", while on the Internet, a characteristic might be "(websites) about a specific topic, such as politics, sports, or technology". It is well-understood that actors in a network can have multiple characteristics, resulting in multiple community memberships that usually overlap (Palla et al., 2005; Yang and Leskovec, 2012b); hence, it is widely accepted that overlapping communities provide a more accurate picture of network structure than disjoint communities (Xie et al., 2013).

To date, however, only a few overlapping community detection algorithms have been successfully applied to large graphs in excess of hundreds of millions of nodes (Prakash et al., 2010; Kang et al., 2011), and, to the best of our knowledge, none of them are based on a probabilistic formulation. Probabilistic methods are often more flexible in that they can serve as "building blocks" to more sophisticated models over additional non-network data sources—particularly text and feature data (Rosen-Zvi et al., 2004; Dietz et al., 2007; Nallapati et al., 2008; Chang and Blei, 2009; Balasubramanyan and Cohen, 2011; Ho et al., 2012a). They may also be extended to settings with multiple networks, e.g., the time-varying setting in which we are given snapshots of the same network at multiple time points (Fu et al., 2009; Ho et al., 2011), or to the nonparametric setting, which provides a principled way to automatically select the number of communities (Kemp et al., 2006; Ho et al., 2012a,b).

Despite some recent work showing that probabilistic methods can already be scaled up to networks with several million nodes (Gopalan and Blei, 2013), there are currently no probabilistic network models that have been reported to scale to hundreds of millions of nodes or more. In this paper, we develop a new probabilistic network model and accompanying inference algorithm for these massive scales. We take a latent space approach, in which each node is associated with an unobserved mixed-membership vector in the latent space that represents a mixture distribution over the multiple possible characteristics (roles or properties) it can have. The mixed-membership assumption can naturally capture the multi-faceted and heterogeneous nature of nodes in real networks—for example, actors in social networks tend to belong to multiple distinct social groups, depending on whom they are interacting with. This is in contrast to the single-membership assumption, where each node can only play a single latent role in its relationship with other nodes. We focus on probabilistic inference of mixed-membership vectors of individual nodes, taking as input the observed structural relation of Internet-scale networks with millions to hundreds of millions of nodes. The inferred mixed-membership position vectors can then be used for overlapping community detection. We demonstrate that our method can infer 1000 communities from a 101-million-node web graph in less than 40 hours using a small cluster of 5 machines, and that, on real-world networks with ground truth, our community recovery accuracy is competitive with or outperforms other scalable probabilistic models.

∗. Qirong Ho and Junming Yin contributed equally.

1.1 Challenges in Probabilistic Modeling of Large Networks

For a network with N nodes, its dense adjacency matrix representation contains $\Theta\left(N^2\right)$ elements. Although the network data itself can be stored in a sparse format (e.g., edge list or adjacency list), several popular probabilistic network models—such as mixed-membership stochastic blockmodels (Airoldi et al., 2008) and latent feature models (Miller et al., 2009)—associate latent variables to each of the N^2 elements in the adjacency matrix, essentially treating the network as non-sparse. Consequently, the inference algorithm naively requires $\Omega(N^2)$ computational complexity. This may still be acceptable for small networks, but is untenable for networks with hundreds of millions of nodes, as considered in this work.

A second difficulty is that larger networks have been observed to have more communities (Leskovec et al., 2005), so the number of hidden variables or parameters in the model often has to grow super-linearly in the number of communities K (in addition to N). Having super-linear growth in the state space can quickly render the inference algorithm computationally intractable, even at modest network scales. Such challenges have already been observed in the mixed-membership stochastic blockmodel (MMSB), which is infeasible for networks with $N \geq 100$ million nodes or $K \geq 100$ communities because of its $O(N^2K^2)$ per-iteration runtime. Recent work by Gopalan and Blei (2013) has brought the per-iteration inference complexity of MMSB down to $O(MK)$, where M is the number of edges. The resulting algorithm allows scalability of up to $N = 4$ million nodes and $K = 1000$ communities in about 1 week on a single machine. However, even that result remains about 1.4 orders of magnitude below our desired target of $N \geq 100$ million nodes.

1.2 Proposed Approach

From a scalability perspective, an ideal large-scale network inference algorithm would require only linear-time work in the number of nodes and communities. Towards this end, we take a different approach to data representation, model construction, and algorithm design that avoids the common edge-based representation of a network (e.g., adjacency matrices or adjacency lists), in favor of viewing the network in terms of *triangular motifs* (also known as triads, or subgraphs of size 3).

Our triangular motif representation captures certain edge patterns observed in node triples (i, j, k), such as the 2-edge triangle $i-j-k$ and the 3-edge triangle $i-j-k-i$. This is a hypergraph representation of the network, where every hyperedge is labeled with the edge pattern over (i, j, k)[1]. Hypergraph representations of networks have been studied in both the social networks (Ghoshal et al., 2009) and statistics (Stasi et al., 2014) literature, and there is an empirical evidence showing that triangles and higher-order subgraphs can substantially reveal the structure and communities within a network (Milo et al., 2002; Yang and Leskovec, 2012b). Moreover, this triangular representation has desirable properties for large-scale network analysis; in particular, it is possible to subsample triangular motifs in a manner that approximately preserves important network attributes such as the *clustering coefficient*, a popular measure of cluster strength.

In the following sections, we shall discuss how to build a scalable, linear-time network inference algorithm based on triangular motifs. We begin with the triangular data representation (Section 2), followed by statistical model design (Section 3), and then parallelizable

1. This is technically a 3-uniform hypergraph, where all hyperedges are associated with 3 nodes.

inference algorithm construction (Section 4). Finally, we implement the algorithm in the distributed, multi-machine cluster setting (Section 5), using a large-scale machine learning platform called Petuum (`www.petuum.org`). The resulting latent space inference algorithm can be used for overlapping community detection on networks with more than 100 million nodes and thousands of communities.

1.3 Related Work

Understanding network structure is highly important to a number of scientific domains, and each domain has developed its own unique tools to analyze the global and local properties of networks. Here, we provide a general overview of network analysis methods from the computer science and statistics literature, with a focus on how well they scale to very large networks with hundreds of millions of nodes or more.

In the general computer science literature, there has been an emphasis on achieving fast, linear-time overlapping community detection on medium-sized networks (usually millions of nodes). Examples include the GCE (Lee et al., 2010), Link (Ahn et al., 2010) and SLPA (Xie and Szymanski, 2012) algorithms. Although these algorithms are fast (less than one hour running time on 1 million nodes), their clustering strategies are not always based on a clearly defined optimization function or statistical model, and might be considered ad-hoc in a statistical or machine learning setting. For example, SLPA propagates community labels through the graph, and does not have an underlying statistical model or objective function. There is neither consensus on which approach works best nor on generalization guarantees to networks with different properties. Moreover, compared to a model-based approach, it is unclear how to incorporate additional network attributes in a principled manner, such as text or other meta-data associated with each node.

In statistics and machine learning, network algorithms are typically created by defining a formal model of the network, and deriving an inference or optimization algorithm to learn the model's parameters. Predominant statistical network models include exponential random graph models (ERGMs) (Morris et al., 2008; Hunter et al., 2008; Guo et al., 2007), stochastic blockmodels (Bickel et al., 2013; Anandkumar et al., 2014; Airoldi et al., 2008; Ho et al., 2011; Gopalan et al., 2012) and latent factor models (Miller et al., 2009; Handcock et al., 2007; Hoff et al., 2002). All of these models are popular for their ease of interpretability, but their inference algorithms generally require $\mathrm{O}(N^2)$ time due to how they model the adjacency matrix, making them patently unscalable to massive networks with $N \geq 100$ million. Exceptions to this trend include assortative MMSB (Gopalan et al., 2012; Gopalan and Blei, 2013), which applied stochastic gradient techniques to achieve linear runtime on the MMSB (Airoldi et al., 2008), and sparse block model (Parkkinen et al., 2009) with $\mathrm{O}(M)$ latent variables (M is the number of edges). Although these methods are scalable in their own right, they still treat *network edges* as the basis for the task of community detection. In this paper, we will argue that *triangular motifs* are a good, if not better, network representation for this task (Section 3), and design a network model based on these triangular motifs.

The natural language processing and information retrieval communities have extended the LDA topic model (Blei et al., 2003) to a variety of text-network models by exploring the links between documents. Examples include the relational topic model (Chang and

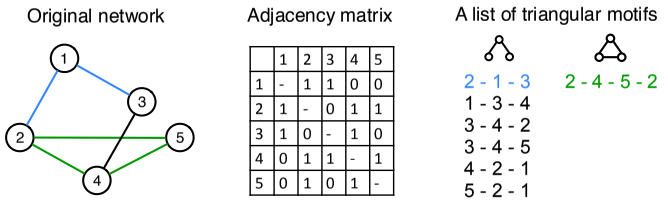
Blei, 2009), Link-PLSA-LDA (Nallapati et al., 2008), Block-LDA (Balasubramanyan and Cohen, 2011), the author-topic model (Rosen-Zvi et al., 2004), and the citation influence model (Dietz et al., 2007). While these models are scalable in that they require O(M) work on the network data, most of them (except for the Block-LDA model) cannot perform network inference in the absence of text data because they do not use a stand-alone network model.

In the data mining and machine learning literature, matrix factorization methods provide a principled framework for decomposing relational data (of which networks are one type) into simpler "basis" components. Although the basis components are sometimes less interpretable than the probability distributions that arise from statistical models (e.g., the basis components may have negative values), the optimization algorithms to perform matrix factorization are usually faster than the inference algorithms for statistical models. In particular, the HEigen algorithm (Kang et al., 2011) computes the rank-k singular value decomposition (SVD) on networks with $N \geq 1$ billion nodes, given a cluster with hundreds of Hadoop machines. Furthermore, matrix factorization can be reinterpreted in a probabilistic setting, as seen in the work of Singh and Gordon (2010), who have developed a probabilistic matrix factorization framework for an arbitrary number of relational matrices. To incorporate network context into such a coupled matrix factorization framework, one would require at least two matrices: the adjacency matrix and one or more matrices of nodes against their features. Whether the resulting algorithm is scalable strongly depends on how the objective function is constructed. If the objective function is dense in the adjacency matrix (i.e., work is performed even on the zeros) such as in Wang et al. (2011), then the algorithm requires at least $\Omega(N^2)$ work. It will not finish in a realistic amount of time on networks with 100 million nodes, even if a large computer cluster is used. On the other hand, if the objective function is sparse (i.e., no work performed on the zeros), then one may take advantage of existing software for distributed, large-scale coupled matrix factorization, such as FlexiFaCT (Beutel et al., 2014). Another example of coupled matrix factorization is the linear-time PICS network analysis algorithm (Akoglu et al., 2012), which can analyze networks with $N \approx 75$k nodes in about 1-2 hours.

2. Data Representation: Triangular Motif Representation of Networks

In many real-world problems where computational cost is a bottleneck, transforming the original raw data into a task-dependent representation might well suffice for subsequent analyses and save significantly on computational cost. Classical examples include: (1) the *bag-of-words* representation of a document, in which the ordering information of words is discarded—although much grammatical information is lost, this representation has proven effective in natural language processing tasks such as topic modeling (Blei et al., 2003); (2) the use of *superpixels* to represent images, in which adjacent pixels are grouped into larger superpixels—the resulting image representation is compact, and leads to faster and better-performing segmentation algorithms in computer vision (Cao and Fei-Fei, 2007; Fulkerson et al., 2009).

Similarly, in probabilistic modeling of networks, the traditional dense adjacency matrix representation (Figure 1) is not suitable for the massive scales ($N \geq 100$ million) considered in this work. In particular, many probabilistic network algorithms touch every entry of



	1	2	3	4	5
1	-	1	1	0	0
2	1	-	0	1	1
3	1	0	-	1	0
4	0	1	1	-	1
5	0	1	0	1	-

Figure 1: Three different representations of the same network: as a graph, as an adjacency matrix, and as a list of 2-edge and 3-edge triangular motifs.

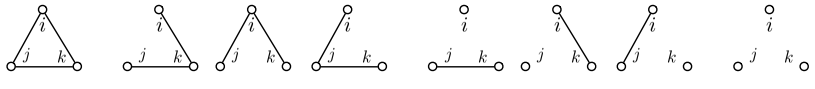


Figure 2: Four types of triangular motifs: (a) 3-edge triangle; (b) 2-edge triangle; (c) 1-edge triangle; (d) empty-triangle. For latent space inference of Internet-scale networks, we only focus on 3-edge and 2-edge triangles.

the adjacency matrix, leading to $\Theta(N^2)$ run-time complexity per iteration. To solve the representational and computational challenges of large-scale probabilistic network inference, we advocate the use of triangular motifs (Figures 1 and 2) as a foundation to achieve succinct yet informative representation of networks. Triangular motifs have historically played an important role in biological network analysis (Milo et al., 2002) as well as in social network analysis (Simmel and Wolff, 1950; Granovetter, 1973; Krackhardt and Handcock, 2007). After describing triangular motifs, we will show in the next few sections how they can be used to construct a mixed-membership network model with $O(CNK)$ per-iteration inference cost for a small constant C.

Given an *undirected* network[2] with N vertices, we shall use the term "1-edge" to refer to edges that exist between two vertices, and the term "0-edge" to refer to missing edges. A triangular motif E_{ijk} over 3 vertices $i < j < k$ is simply the type of subgraph exhibited by those 3 vertices. There are four basic classes of triangular motifs (Figure 2), distinguished by their number of 1-edges: 3-edge triangle Δ_3 (three 1-edges), 2-edge triangle Δ_2 (two 1-edges), 1-edge triangle Δ_1 (one 1-edge), and empty-triangle Δ_0 (no 1-edges). The total number of triangular motifs, over all four classes, is $\Theta(N^3)$. However, our goal is not to account for all four classes in a network representation; instead, we will focus on 3-edge and 2-edge triangles (Δ_3 and Δ_2) while ignoring 1-edge triangles and empty-triangles (Δ_1 and Δ_0). There are two key motivations for this approach:

2. Our approach can also be generalized to directed networks, though the analysis is more involved since directed networks can have more triangular motifs than undirected networks.

1. In the network literature, many important network measures and characteristics are quantified and captured by the three-node *connected* subgraphs (namely Δ_3 and Δ_2). To name a few: (1) the network clustering coefficient (Watts and Strogatz, 1998; Newman and Park, 2003), a common measure of triadic closure in social network theory (Simmel and Wolff, 1950; Granovetter, 1973; Krackhardt and Handcock, 2007), is defined as the relative number of 3-edge triangles compared to the total number of connected vertex triplets (i.e., 3-edge and 2-edge triangles); (2) the most significant and recurrent structural patterns in many complex networks, so-called "network motifs", are the three-node connected subgraphs (Milo et al., 2002). Therefore, one naturally expects the classes of 3-edge and 2-edge triangles to capture most of the informative structure in a large network.

2. The four classes of triangular motifs certainly contain redundant information, because in principle one needs only $\Theta(N^2)$ bits to fully describe a network. Though node triplets with only one or zero 1-edge (namely Δ_1 and Δ_0) are not uncommon in real-world networks (Leskovec et al., 2009), those information might already be contained in the list of Δ_3 and Δ_2 triangles. For instance, one can easily conclude from the 2/3-edge triangular motif representation of the network in Figure 1 that a 1-edge triangle Δ_1 is formed among the node triplet $\{1, 3, 5\}$. In fact, almost all network information found in an adjacency matrix representation is preserved by Δ_3 and Δ_2 triangles: every 1-edge that appears in some Δ_3 or Δ_2 can be identified in the corresponding triangular motif. The only exception is a set of isolated 1-edges. However, these small strongly connected components of size 2 can be easily detected and are less interesting from a large-scale community detection perspective.

One main advantage of characterizing a network in terms of triangular motifs Δ_3 and Δ_2 is that such representation is typically more compact than the adjacency matrix representation[3], via the following lemma:

Lemma 1 *The total number of Δ_3's and Δ_2's is $\Theta(\sum_i D_i^2)$, where D_i is vertex i's degree.*

Proof Let $\mathcal{N}_i$ be the neighbor set of vertex i. For each vertex i, form the set $\mathcal{T}_i$ of tuples (i, j, k) where $j < k$ and $j, k \in \mathcal{N}_i$, which represents the set of all pairs of neighbors of i. Because j and k are neighbors of i, the triangular motif formed among every tuple $(i, j, k) \in \mathcal{T}_i$ is either a 3-edge triangle Δ_3 or a 2-edge triangle Δ_2. It is easy to see that each Δ_2 is accounted for by exactly one $\mathcal{T}_i$, where i is the center vertex of the Δ_2, and that each Δ_3 is accounted for by three sets $\mathcal{T}_i, \mathcal{T}_j$ and $\mathcal{T}_k$, one for each vertex in the 3-edge triangle. Thus, $\sum_i |\mathcal{T}_i| = \sum_i \frac{1}{2}(D_i)(D_i - 1)$ is an upper bound to the total number of Δ_3's and Δ_2's. By modifying the preceding argument slightly, we can also show that the total number of Δ_3's and Δ_2's is bounded below by $\frac{1}{3}\sum_i |\mathcal{T}_i|$. ■

For networks with low maximum degree $D_{\max}$, $\Theta(\sum_i D_i^2) = O(ND_{\max}^2)$ is typically much smaller than $\Theta(N^2)$. Even in real-world networks with power-law behavior, the number of Δ_3's and Δ_2's tends to be still much smaller than the size of adjacency matrix N^2

3. Recall the size of data (including both 1-edges and 0-edges) in the adjacency matrix representation of a network is $\Theta(N^2)$, even if the data can be stored in a sparse manner.

Network	# nodes N	# edges M	adj matrix size N^2	Upper bound of $\#(\Delta_3, \Delta_2)$
DBLP	317,080	1,049,866	1.01×10^{11}	2.2×10^7
Amazon	334,863	925,872	1.12×10^{11}	9.8×10^6
Youtube	1,134,890	2,987,624	1.29×10^{12}	1.5×10^9
Livejournal	3,997,962	34,681,189	1.60×10^{13}	4.3×10^9
WDC Subdomain/Host	101 million	2 billion	1.02×10^{16}	1.0×10^{13}

Table 1: Networks used in this paper: number of 2-edge and 3-edge triangles versus adjacency matrix size. Many (though not all) probabilistic network models use the entire non-sparse adjacency matrix as input. The 2/3-edge triangular representation provides a more compact alternative—in all example networks, the upper bound for the number of Δ_3's and Δ_2's is at least 1000 times smaller than the full adjacency matrix.

(Table 1), and this allows us to construct a more efficient inference algorithm scalable to larger networks. However, the large number of triangles still poses storage and computational challenges. Our solution is to keep the triangular motifs in an *implicit* representation, and subsample a mini-batch of them only when they are needed (and discard them afterwards) in the stochastic variational inference algorithm. We will describe such technique in Section 5.1, and our results will surprisingly show that the inference algorithm only needs to touch a small fraction of triangles before converging to an accurate result.

3. Model: Scalable Network Model (STM) Based on Triangular Motifs

Given a network, now represented by a bag of triangular motifs Δ_3 and Δ_2 (Figure 1), the goal is to perform probabilistic inference of mixed-membership vectors of individual nodes, which can then be used for overlapping community detection. This is as opposed to traditional single-membership community detection, which assigns each vertex to exactly one community. By taking a mixed-membership approach, one gains many benefits over single-membership models, such as outlier detection, improved visualization, and better interpretability (Blei et al., 2003; Airoldi et al., 2008).

To construct the mixed-membership network model based on the triangular motif representation, which we call STM for scalable triangle model, we first establish some notation used throughout the paper. Recall that we are concerned only with 3-edge and 2-edge triangles (Δ_3 and Δ_2) in the triangular motif representation of networks. For each triplet of vertices $i, j, k \in \{1, \ldots, N\}, i < j < k$, if the subgraph on i, j, k is a 2-edge triangle with i, j, or k at the center, then let $E_{ijk} = 1$, 2, or 3 respectively; if a 3-edge triangle is formed among i, j, k, then let $E_{ijk} = 4$. In other words, E_{ijk} denotes the observed type of triangular motif on vertices i, j, k. Whenever the subgraph on i, j, k is a 1-edge triangle or an empty-triangle, we simply discard it (i.e., E_{ijk} is not part of the model) for the reasons that have been described in Section 2 .

Next, we assume K latent communities, and that each vertex takes a mixture distribution over them. The observed 2-edge and 3-edge triangles $\{E_{ijk}\}$ are generated according to a latent space model that defines (1) the mixture distribution over community memberships at each vertex, and (2) a tensor-like data structure of triangle-generating probabilities. More formally, each vertex i is associated with a community mixed-membership vector $\theta_i \in \Delta^{K-1}$ restricted to the $(K-1)$-simplex Δ^{K-1}. This mixed-membership vector θ_i is used to gen-

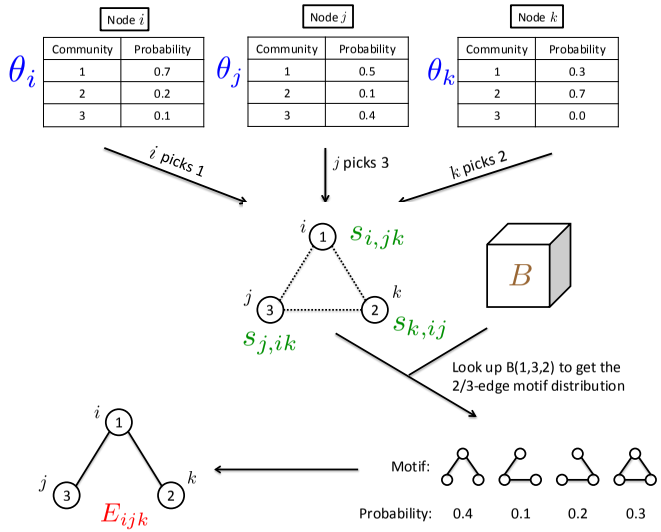


Figure 3: High-level generative process of how the mixed-membership vectors $\theta_i, \theta_j, \theta_k$ and a "tensor" of triangle-generating probabilities $\mathbf{B}$ are used to generate the triangular motif E_{ijk}. In this example, $(s_{i,jk} = 1, s_{j,ik} = 3, s_{k,ij} = 2)$ represents a context-dependent instantiation of communities for three vertices i, j, k when they are interacting, and the entry B_{132} is examined to obtain the multinomial parameter of generating E_{ijk}. All these probabilities are unobserved and need to be inferred from the observed triangular motifs.

erate community indicators $s_{i,jk} \in \{1, \ldots, K\}$, each of which represents the community chosen by vertex i when it is forming a triangle with vertices j and k. The probability of observing a triangular motif E_{ijk} depends on the community triplet $(s_{i,jk}, s_{j,ik}, s_{k,ij})$ and a tensor of triangle-generating parameters $\mathbf{B}$. Each element B_{xyz} of this tensor contains the multinomial probabilities of generating the four possible 2-edge and 3-edge triangles, when three vertices with respective communities x, y, z interact. A schematic of this generative process is illustrated in Figure 3.

3.1 Parsimonious Parameter Structure

The triangle-generating probability tensor $\mathbf{B}$ described above contains $O(K^3)$ distributions, one for each possible community combination (x, y, z). This presents challenging issues in both statistical estimation and computational complexity: (1) the large number of $O(K^3)$ parameters to be estimated leads to a loss of statistical efficiency (particularly when $K \geq 1000$); (2) inference requires $O(K^3)$ time per iteration, which is computationally intractable. An elegant solution to this problem is to reduce the number of triangle-generating

parameters by partitioning the $O(K^3)$ community combination space intro several groups, and then sharing parameters within the same group (Yin et al., 2013). Our parameter-sharing strategy is based on the number of distinct states in the configuration of the community triplet (x, y, z):

1. If all three communities are the same x, the triangle-generating probability is determined by B_{xxx} (this is identical to the diagonal of the $O(K^3)$ parameterization of $\mathbf{B}$ tensor). There are K such parameters $B_{111}, \ldots, B_{KKK}$.

2. If only two communities indices exhibit the same state x (called the majority community), the probability of triangles is governed by B_{xx}, no matter what the third, minority community y is. The intuition is that the identity of the majority community is the dominant factor in determining the triangle-generating probabilities, regardless of the minority community. As with any statistical assumption, this is never perfectly true for real data, but nevertheless turns out to be a good assumption, as our results will show. There are K such parameters $B_{11}, \ldots, B_{KK}$.

3. If the three community indices are all distinct, the probability of triangular motifs depends on B_0, a single parameter independent of the community identities. This is similar in spirit to the combined off-diagonal blockmatrix parameters used in hierarchical blockmodels (Ho et al., 2012a) and the assortative MMSB (Gopalan et al., 2012; Gopalan and Blei, 2013).

This sharing strategy yields just $O(K)$ parameters $B_0, B_{xx}, B_{xxx}, x \in \{1, \ldots, K\}$ (since there are at most four distinct 2-edge and 3-edge triangles), allowing STM to scale to far more communities than a naive $O(K^3)$ parameterization of $\mathbf{B}$ tensor.

3.2 Equivalence Classes of Triangles

For certain configurations of community triplet $(s_{i,jk}, s_{j,ik}, s_{k,ij})$, some of the triangular motifs can become indistinguishable. To illustrate, consider the example illustrated in Figure 4: suppose that nodes i and j take membership in community x (the majority community), while the node k takes community y—that is, we have $x = s_{i,jk} = s_{j,ik} \neq s_{k,ij} = y$. Under these assignments, one cannot distinguish the 2-edge triangle with i in the center (right panel, $E_{ijk} = 1$) from the triangle with j in the center (left panel, $E_{ijk} = 2$). This is because both are 2-edge triangles centered at a vertex with majority community x, and are thus *equivalent* with respect to the underlying community configuration $x - x - y$. Formally, this $x - x - y$ community configuration induces a set of triangle equivalence classes $\{\{1, 2\}, \{3\}, \{4\}\}$ of all possible triangular motifs $E_{ijk} \in \{1, 2, 3, 4\}$. We treat the triangular motifs within the same equivalence class as *stochastically equivalent*; that is, the conditional probabilities of events $E_{ijk} = 1$ and $E_{ijk} = 2$ are assumed to be the same if $x = s_{i,jk} = s_{j,ik} \neq s_{k,ij}$. All possible cases are enumerated as follows (see also Table 2):

1. If all three vertices have the same community x, all three 2-edge triangles are equivalent and the induced set of equivalence classes is $\{\{1, 2, 3\}, \{4\}\}$. The probability of E_{ijk} is determined by $B_{xxx} \in \Delta^1$, where $B_{xxx,1}$ represents the *total* probability of sampling an 2-edge triangle from $\{1, 2, 3\}$ and $B_{xxx,2}$ represents the 3-edge triangle probability. Thus, the probability of a particular 2-edge triangle is $B_{xxx,1}/3$.

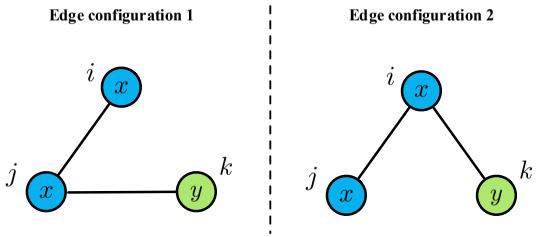


Figure 4: An example of two indistinguishable, equivalent motifs. Although the edge configuration is different in two motifs ($i-j-k$ on the left versus $j-i-k$ on the right), the motifs are indistinguishable under the given community combination, because they are both cases in which the central node (j and i respectively) has the majority community x.

2. If only two vertices have the same community x (majority community), the probability of E_{ijk} is governed by $B_{xx} \in \Delta^2$. Here, $B_{xx,1}$ and $B_{xx,2}$ represent the 2-edge triangle probabilities (for 2-edge triangles centered at a vertex in majority and minority community respectively), and $B_{xx,3}$ represents the 3-edge triangle probability. There are two possible 2-edge triangles with a vertex in majority community at the center, and hence each has probability $B_{xx,1}/2$.

3. If all three vertices have distinct community, the probability of E_{ijk} depends on $B_0 \in \Delta^1$, where $B_{0,1}$ represents the *total* probability of sampling an 2-edge triangle from $\{1,2,3\}$ (regardless of the center vertex's community) and $B_{0,2}$ represents the 3-edge triangle probability.

3.3 Generative Process for Scalable Triangular Model (STM)

We summarize the generative process for a bag of triangular motifs under the STM:

1. Draw $B_0 \in \Delta^1$, $B_{xx} \in \Delta^2$ and $B_{xxx} \in \Delta^1$ for each community $x \in \{1,\ldots,K\}$, according to symmetric Dirichlet distributions $\text{Dirichlet}(\lambda)$.

2. For each vertex $i \in \{1,\ldots,N\}$, draw a mixed-membership vector $\theta_i \sim \text{Dirichlet}(\alpha)$.

3. For each triplet of vertices $(i,j,k), i<j<k$,

   (a) Draw community configuration $s_{i,jk} \sim \text{Discrete}(\theta_i)$, $s_{j,ik} \sim \text{Discrete}(\theta_j)$, $s_{k,ij} \sim \text{Discrete}(\theta_k)$.

   (b) Draw an *equivalence class* of triangular motifs $\{E_{ijk}\}$ based on B_0, B_{xx}, B_{xxx} and the configuration of $(s_{i,jk}, s_{j,ik}, s_{k,ij})$. See Table 2 for the conditional probabilities of each equivalence class.

   (c) Draw a triangular motif E_{ijk} uniformly at random from the chosen equivalence class (this is what we meant by *stochastic equivalence* earlier).

$(s_{i,jk}, s_{j,ik}, s_{k,ij})$	Equivalence classes	Conditional probability of $E_{ijk} \in \{1,2,3,4\}$
$x = s_{i,jk} = s_{j,ik} = s_{k,ij}$	$\{1,2,3\},\{4\}$	$\text{Discrete}\big(\big[\frac{B_{xxx,1}}{3}, \frac{B_{xxx,1}}{3}, \frac{B_{xxx,1}}{3}, B_{xxx,2}\big]\big)$
$x = s_{i,jk} = s_{j,ik} \neq s_{k,ij}$	$\{1,2\},\{3\},\{4\}$	$\text{Discrete}\big(\big[\frac{B_{xx,1}}{2}, \frac{B_{xx,1}}{2}, B_{xx,2}, B_{xx,3}\big]\big)$
$x = s_{i,jk} = s_{k,ij} \neq s_{j,ik}$	$\{1,3\},\{2\},\{4\}$	$\text{Discrete}\big(\big[\frac{B_{xx,1}}{2}, B_{xx,2}, \frac{B_{xx,1}}{2}, B_{xx,3}\big]\big)$
$x = s_{j,ik} = s_{k,ij} \neq s_{i,jk}$	$\{2,3\},\{1\},\{4\}$	$\text{Discrete}\big(\big[B_{xx,2}, \frac{B_{xx,1}}{2}, \frac{B_{xx,1}}{2}, B_{xx,3}\big]\big)$
$s_{k,ij} \neq s_{i,jk} \neq s_{j,ik}$	$\{1,2,3\},\{4\}$	$\text{Discrete}\big(\big[\frac{B_{0,1}}{3}, \frac{B_{0,1}}{3}, \frac{B_{0,1}}{3}, B_{0,2}\big]\big)$

Table 2: Equivalence classes and conditional probabilities of E_{ijk} given $s_{i,jk}, s_{j,ik}, s_{k,ij}$ (see text for details).

It is worth noting that STM is not a generative model of networks since (a) empty-triangles and 1-edge triangles are not modeled, and (b) one can possibly generate a set of triangles that does not correspond to any network. This is a consequence of using a bag-of-triangles model in which the generative process does not force overlapping triangles to have consistent edge values. For example, generating a 2-edge triangle centered at i for (i,j,k) followed by a 3-edge triangle for (j,k,ℓ) can produce a mismatch on the edge (j,k). However, given a bag of triangular motifs $\mathbf{E}$ extracted from a network, the above procedure defines a valid probabilistic model $p(\mathbf{E} \mid \alpha, \lambda)$, and we can use it for performing posterior inference $p(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B} \mid \mathbf{E}, \alpha, \lambda)$. We stress that our goal is fast and scalable latent space inference for overlapping community detection, not network simulation[4].

4. Inference: Efficient Stochastic Variational Inference (SVI) for STM

In this section, we present a stochastic variational inference (SVI) algorithm (Hoffman et al., 2013) with $O(CNK)$ per-iteration cost for performing approximate inference under our STM (where C is a small constant). The inferred posterior probability distribution of the latent variables, in particular the mixed-membership vectors θ_i, can then be used for overlapping community discovery. The high-level ideas are to (1) develop a structured mean-field approximation—a more accurate approximation to the true posterior than the naive mean-field solution explored in earlier mixed-membership network models (Airoldi et al., 2008), but requires $O(K^3)$ run-time per triangular motif; (2) refine the structured mean-field approximation to lower the run-time complexity to $O(K)$ while maintaining its high accuracy, inspired by a careful understanding of the typical structure of the posterior distribution; (3) apply stochastic variational inference (SVI) to yield a faster approximate inference algorithm; (4) propose a new stochastic subsampling strategy in the SVI algorithm that empirically achieves high-quality results without having to touch every 2/3-edge triangle in the network, thus ensuring scalability to very large networks which may have more than trillions (10^{12}) of triangles.

4. For applications in which simulation is vital, it is possible to design a variant of STM that sequentially generates motifs so as to have consistent edge values across overlapping triangles. In such model, all possible triples (i,j,k) are sequentially considered, and a new motif among them is simulated conditioning on the patterns of other motifs that have already been generated. This non-exchangeable model is, however, out of the scope of this work.

4.1 Structured Mean-field Approximation

As with other mixed-membership models, computing the true posterior of the latent variables $p(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B} \mid \mathbf{E}, \alpha, \lambda)$ under the STM is intractable, and approximate inference must be employed. There are three common strategies to perform approximate inference in mixed-membership models: Markov chain Monte Carlo (MCMC) (Griffiths and Steyvers, 2004), variational inference (Blei et al., 2003; Airoldi et al., 2008), and spectral methods (Anandkumar et al., 2014). MCMC methods are typically viewed as the "gold standard" as they are guaranteed to eventually (but could be slow in practice) converge to the true posterior. Variational inference provides a fast approximation that can be accurate enough for the task at hand, provided care has been taken to design an appropriate factorized variational distribution. Spectral methods have shown much promise in terms of accuracy and scalability, but the $O(K^3)$ run-time complexity limits its application to the setting of $K \geq 1000$ we are considering.

Based on these considerations, we choose a structured mean-field approximation for the STM model. The corresponding inference algorithm proceeds via coordinate ascent update on an objective function known as "variational lower bound" that depends on a set of "variational parameters". To construct the variational lower bound, we begin by approximating the true (but intractable) posterior $p(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B} \mid \mathbf{E}, \alpha, \lambda)$ by a *partially* factorized distribution $q(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B})$,

$$\begin{aligned} q(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B}) &= q(\mathbf{s} \mid \boldsymbol{\phi}) q(\boldsymbol{\theta} \mid \boldsymbol{\gamma}) q(\mathbf{B} \mid \boldsymbol{\eta}) \\ &= \left[\prod_{(i,j,k) \in I} q(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \phi_{ijk}) \right] \left[\prod_{i=1}^{N} q(\theta_i \mid \gamma_i) \right] \left[\prod_{x=1}^{K} q(B_{xxx} \mid \eta_{xxx}) \right] \left[\prod_{x=1}^{K} q(B_{xx} \mid \eta_{xx}) \right] [q(B_0 \mid \eta_0)], \end{aligned} \tag{1}$$

where I is the set of all 2/3-edge triangles in the network, i.e., $I = \{(i,j,k) : i < j < k, E_{ijk} \in \{1,2,3,4\}\}$. The factorized distribution $q(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B})$ contains 3 types of terms:

1. Terms for the community triplet distribution for each triangle, $q(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \phi_{ijk})$;
2. Terms for the mixed-membership community distribution for each node, $q(\theta_i \mid \gamma_i)$;
3. Terms for the triangle-generating distributions, $q(B_{xxx} \mid \eta_{xxx})$, $q(B_{xx} \mid \eta_{xx})$, and $q(B_0 \mid \eta_0)$.

In particular, we have defined a *joint* (as opposed to fully factorized) distribution $q(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \phi_{ijk})$ over the community triplet $(s_{i,jk}, s_{j,ik}, s_{k,ij})$:

$$q(s_{i,jk} = x, s_{j,ik} = y, s_{k,ij} = z) \doteq q_{ijk}(x,y,z) = \phi_{ijk}^{xyz}, \quad x,y,z = 1, \ldots, K. \tag{2}$$

The posterior $q(\theta_i)$ is a Dirichlet(γ_i), and the posteriors of B_{xxx}, B_{xx}, B_0 are parameterized as: $q(B_{xxx}) = \text{Dirichlet}(\eta_{xxx}), q(B_{xx}) = \text{Dirichlet}(\eta_{xx})$, and $q(B_0) = \text{Dirichlet}(\eta_0)$.

Rationale for the Structured Approximation. Conditioning on the observed type of triangular motif E_{ijk} on vertex triplet (i,j,k)—either a 2-edge triangle Δ_2 or a 3-edge triangle Δ_3, the community indicators $(s_{i,jk},\ s_{j,ik},\ s_{k,ij})$ are often highly correlated. For example, suppose that nodes i,j,k form a 3-edge triangle, then it is empirically and sociologically far more likely that they all belong to the same community, as opposed to being a mix of different communities. As a result, the true posterior distribution $p(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \boldsymbol{\theta}, \mathbf{B}, \mathbf{E}, \alpha, \lambda)$—a discrete distribution over K^3 events—tend to concentrate around (some of) the K "all-the-same-community" events $\{s_{i,jk} = s_{j,ik} = s_{k,ij} = 1, 2, \ldots, K\}$. If we think of the posterior as a third-order tensor of dimension $K \times K \times K$ in which each element (x,y,z) represents the posterior probability $p(s_{i,jk} = x, s_{j,ik} = y, s_{k,ij} = z \mid \boldsymbol{\theta}, \mathbf{B}, \mathbf{E}, \alpha, \lambda)$, we see that it is very likely to be a tensor with rank greater than one due to its non-zero diagonal entries. Therefore, it cannot be accurately approximated by a fully factorized distribution $q(s_{i,jk} \mid \phi_{i,jk}) q(s_{j,ik} \mid \phi_{j,ik}) q(s_{k,ij} \mid \phi_{k,ij})$ as used in the naive mean-field approximation, because the outer product yields a rank-1 tensor. A similar argument holds for 2-edge triangles, whose nodes are likely to be in one or two communities, with a similar correlation structure as to the 3-edge case. For this reason, we have chosen a joint form $q(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \phi_{ijk})$ for the variational posterior, in order to capture the correlation structure of the true posterior $p(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \boldsymbol{\theta}, \mathbf{B}, \mathbf{E}, \alpha, \lambda)$.

However, having a structured approximation is computationally more expensive: the variational parameter ϕ_{ijk} used in the variational posterior $q(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \phi_{ijk})$ is a tensor containing K^3 entries, which would cause an unacceptable $O(K^3)$ run-time per triangular motif if handled naively. In the next section, we will discuss a more refined approximation strategy that maintains the high accuracy of the structured mean-field approximation but only requires $O(K)$ run-time per triangular motif, thus ensuring scalability to thousands of communities K.

4.2 Stochastic Variational Inference (SVI) Algorithm

The structured mean-field approximation (Wainwright and Jordan, 2008) aims to minimize the KL divergence $\text{KL}(q \parallel p)$ between the approximating distribution q and the true posterior p; it is equivalent to maximizing a lower bound $\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ of the log marginal likelihood of the triangular motifs (based on Jensen's inequality) with respect to the variational parameters $\{\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma}\}$ defined in Equation 1:

$$\log p(\mathbf{E} \mid \alpha, \lambda) \geq \mathbb{E}_q[\log p(\mathbf{E}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{B} \mid \alpha, \lambda)] - \mathbb{E}_q[\log q(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B})] \doteq \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma}). \tag{3}$$

To simplify the notation, we decompose the variational objective $\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ into a "global" term and a summation of "local" terms, where each local term corresponds to one 2/3-edge triangle in the network (see Appendix A.1 for their exact forms).

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma}) = g(\boldsymbol{\eta}, \boldsymbol{\gamma}) + \sum_{(i,j,k) \in I} \ell(\phi_{ijk}, \boldsymbol{\eta}, \boldsymbol{\gamma}). \tag{4}$$

The global term $g(\boldsymbol{\eta}, \boldsymbol{\gamma})$ depends only on the global variational parameters $\boldsymbol{\eta}$, which govern the posterior of the triangle-generating probabilities $\mathbf{B}$, as well as the per-node mixed-membership parameters $\boldsymbol{\gamma}$. Each local term $\ell(\phi_{ijk}, \boldsymbol{\eta}, \boldsymbol{\gamma})$ depends on per-triangle variational parameter ϕ_{ijk} as well as the global parameters. Our distributed and parallel implementation strategy will involve splitting the inferential work along disjoint sets of local variational parameters, as we will show in Algorithm 1.

To understand how an SVI algorithm accelerates approximate inference, let us consider Equation 4 with the local parameters $\boldsymbol{\phi}$ being maximized out. Define $\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma}) \doteq$

$\max_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma})$, i.e., the variational objective achieved by fixing the global parameters $\boldsymbol{\eta}, \boldsymbol{\gamma}$ and optimizing the local parameters $\boldsymbol{\phi}$:

$$\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma}) = g(\boldsymbol{\eta}, \boldsymbol{\gamma}) + \sum_{(i,j,k)\in I} \max_{\phi_{ijk}} \ell(\phi_{ijk}, \boldsymbol{\eta}, \boldsymbol{\gamma}). \tag{5}$$

The idea behind SVI (Hoffman et al., 2013) is as follows: instead of computing the gradient $\nabla\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$ to perform the variational inference of the global parameters $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$, which involves a costly summation over all triangular motifs as in Equation 5, an unbiased noisy approximation of the gradient can be obtained much more cheaply by summing over a small subsample of triangles. One can then use this approximate gradient in the stochastic gradient ascent algorithm (Bottou, 2004) to maximize the variational objective $\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$. With this unbiased estimate of the gradient and a suitable adaptive step size, the algorithm is guaranteed to converge (in probability) to a stationary point of the variational objective $\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$ (Robbins and Monro, 1951). It should be noted that the subsampling strategy comes at the cost of introducing a variance in gradient computation, and hence more iterations might be required for convergence. However, the time taken by each iteration can be greatly reduced to a small fraction of the original time; for example, if we subsample 10% of the triangles, then only 10% computation time is required to obtain the approximate gradient. In fact, there is significant empirical evidence that SVI algorithm can lead to faster convergence on various problems (Hoffman et al., 2013; Yin et al., 2013).

Subsampling Strategy. In our setting, the most natural way to obtain an unbiased gradient of $\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$ is to sample a "mini-batch" of triangular motifs at each iteration, and then average the gradient of local terms in Equation 5 *only* for these sampled triangles. Formally, let $m = |I|$ be the total number of 2/3-edge triangles[5] and define

$$\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = g(\boldsymbol{\eta}, \boldsymbol{\gamma}) + \frac{m}{|S|} \sum_{(i,j,k)\in S} \max_{\phi_{ijk}} \ell(\phi_{ijk}, \boldsymbol{\eta}, \boldsymbol{\gamma}), \tag{6}$$

where S is a mini-batch of triangles sampled uniformly at random. It is easy to verify that $\mathbb{E}_S[\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})] = \mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$, hence $\nabla\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ is unbiased: $\mathbb{E}_S[\nabla\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})] = \nabla\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$.

In our implementation, we subsample C pairs of neighbors for each node $i \in \{1, \ldots, N\}$ in parallel, where C is a small constant[6], resulting in a mini-batch set S containing CN triangles in each SVI iteration. Such fast parallel subsampling procedure, however, may induce a bias in the estimate of the gradient because the CN triangles in S are not necessarily sampled from the space of all triangles uniformly at random. We defer the discussion of this subtlety to Section 5.1, after the detailed description of our implementation. The performance on overlapping community detection in Section 6 shows that our SVI algorithm converges to a high-quality result without having to touch all 2/3-edge triangles in the network, i.e., $CNt_{\max} \ll m$, where $t_{\max}$ is the number of iterations until convergence.

5. The total number of 2/3-edge triangles in large networks may be computationally infeasible to count—in fact, triangle counting is a hard problem in its own right and an area of active research (Low et al., 2012). In Lemma 1, we have shown that $\frac{1}{3}T \le m \le T$ where $T = \sum_i \frac{1}{2}(D_i)(D_i - 1)$. In our implementation, we simply let $m \approx T$, thus approximating m to within a factor of 3 of its true value.
6. All our experiments are conducted with the minimum value $C = 1$, which is shown to be sufficiently accurate for large-scale networks (Section 6.4).

Exact Local Update. To obtain the gradient $\nabla\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$, one needs to compute the optimal local variational parameters ϕ_{ijk} (keeping $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ fixed) for each sampled triangle (i, j, k) in the mini-batch S; these optimal ϕ_{ijk}'s are then used in Equation 6 to compute $\nabla\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$. Taking partial derivatives of Equation 4 with respect to each local term ϕ_{ijk}^{xyz} and setting them to zero, we get for distinct $x, y, z \in \{1, \ldots, K\}$,

$$\phi_{ijk}^{xyz} \propto \exp\Big\{\mathbb{E}_q[\log B_{0,2}]\mathbb{I}[E_{ijk} = 4] + \mathbb{E}_q[\log(B_{0,1}/3)]\mathbb{I}[E_{ijk} \neq 4] + \mathbb{E}_q[\log\theta_{i,x} + \log\theta_{j,y} + \log\theta_{k,z}]\Big\}.$$

See Appendix A.2 for the update equations of ϕ_{ijk}^{xxx} and ϕ_{ijk}^{xxy} $(x \neq y)$.

O(K) Approximation to Local Update. As explained earlier, because there are K^3 variational parameters ϕ_{ijk}^{xyz} for each sampled triangle (i, j, k), the exact local update requires $O(K^3)$ work to solve for all ϕ_{ijk}^{xyz}, making it unscalable. We now describe an $O(K)$ approximation that is almost as accurate as the exact $O(K^3)$ update (and is certainly far more accurate than a fully factorized approximation as in the naive mean-field approach). To enable a faster local update, we replace $q_{ijk}(x, y, z \mid \phi_{ijk})$ in Equation 2 with a simpler, but still structured (as opposed to fully factorized), "mixture-of-deltas" variational distribution,

$$q_{ijk}(x, y, z \mid \delta_{ijk}) = \sum_{(a,b,c)\in\mathcal{A}} \delta_{ijk}^{abc}\, \mathbb{I}[x = a, y = b, z = c],$$

where $\mathcal{A}$ is a set containing O(K) strategically chosen triples (a, b, c), and $\sum_{(a,b,c)\in\mathcal{A}} \delta_{ijk}^{abc} = 1$. In other words, this "mixture-of-deltas" variational distribution is still a discrete probability distribution over K^3 events, but its probability mass is assumed to fall entirely on the O(K) randomly chosen entries and community combinations not in $\mathcal{A}$ are assumed to have zero probability. Conveniently, the update equations for the O(K) free parameters δ_{ijk}^{abc} are practically identical to the full $O(K^3)$ update equations for ϕ_{ijk} (Equations 9, 10 and 11 in Appendix A.2). The only difference is normalization, which is now performed over the set of O(K) δ's instead of all K^3 events. Since we subsample CN triangles in each SVI iteration, the total time complexity of local update is $O(CNK)$ per iteration.

Careful selection of the O(K) entries $(a, b, c) \in \mathcal{A}$ is critical to the performance of the O(K) "mixture-of-deltas" approximation. In our implementation, we choose these entries in $\mathcal{A}$ based on the observation that most nodes in real networks belong to just a few communities (Yang and Leskovec, 2012b), and hence the true mixed-membership vectors are also concentrated around just a few entries. In fact, more than 90% of nodes in most real-world networks studied in this paper have no more than 10 ground-truth communities. This suggests that for any triangle (i, j, k), the number of plausible assignments to its community triplet $(s_{i,jk}, s_{j,ik}, s_{k,ij})$ is usually very small compared to K^3, and we should focus our choice on triplets (a, b, c) such that the variational posteriors $q(\theta_{i,a}), q(\theta_{j,b}), q(\theta_{k,c})$ are all large. Furthermore, it is reasonable to add $O(K)$ random community combinations into consideration so as to enable the SVI algorithm to explore other communities to avoid becoming stuck in a local optimum. We employ the following procedure to construct $\mathcal{A}$:

`STM-ChooseA:`

1. Find the U_0 largest communities in each of $q(\theta_i)$, $q(\theta_j)$ and $q(\theta_k)$, and insert them into a set R_{core} ($|R_{\text{core}}| \le 3U_0$). In our experiments, we use $U_0 = 10$ because empirically most nodes have no more than 10 communities.

2. Pick U_1 communities uniformly at random from $\{1, \ldots, K\}$, and put them into a set R_{random}. In our experiments, we use $U_1 = K/10$ so that $U_1 > U_0$ for large K, thus encouraging exploration.

3. Combine both sets: $R = R_{\text{core}} \cup R_{\text{random}}$. Note that $|R| = O(K)$.

4. Generate $|R|$ diagonal community combinations $\mathcal{A}_{\text{diag}} = \{(a, a, a) \text{ for all } a \in R\}$.

5. Generate $3|R|$ off-diagonal community combinations $\mathcal{A}_{\text{off}}$. Each combination is generated as follows: first draw $a \in R$ uniformly at random, then draw $b \in R$ where $b \neq a$. Finally, draw o uniformly at random from $\{1, 2, 3\}$. If $o = 1$, add (a, a, b); if $o = 2$, add (a, b, a); and if $o = 3$, add (b, a, a).

6. Generate $3|R|$ off-off-diagonal community combinations $\mathcal{A}_{\text{offoff}}$. Each combination is generated as follows: first draw $a \in R$ uniformly at random, then draw $b \in R$ where $b \neq a$, and finally draw $c \in R$ where $c \neq a, c \neq b$. Add (a, b, c) to $\mathcal{A}_{\text{offoff}}$.

7. Combine all three sets to obtain $\mathcal{A} = \mathcal{A}_{\text{diag}} \cup \mathcal{A}_{\text{off}} \cup \mathcal{A}_{\text{offoff}}$.

Note that we do not pick all possible community combinations for $\mathcal{A}_{\text{off}}$ and $\mathcal{A}_{\text{offoff}}$, which are of size $\mathrm{O}(K^2)$ and $\mathrm{O}(K^3)$, respectively. This is an intentional trade-off; we limit the size of $\mathcal{A}$ to $\mathrm{O}(K)$ to keep the inference feasible[7]. Because we re-select $\mathcal{A}$ every time we perform the local variational update for some triangle (i, j, k), the SVI algorithm still manages to explore the most likely community combinations with reasonably high probability, thus avoiding any bias due to a single choice of $\mathcal{A}$. In practice, we find that this $\mathrm{O}(K)$ "mixture-of-deltas" approximation works nearly as well as the full parameterization in Equation 2, while requiring only $O(K)$ work per sampled triangle. Finally, we stress that the "mixture-of-deltas" variational approximation is theoretically well-justified and not a heuristic. Standard variational inference theory states that any choice of $\mathcal{A}$ yields a valid lower bound to the log marginal likelihood at the current iteration, and therefore we are always updating variational parameters to maximize some variational lower bound.

Global Update. We appeal to stochastic natural gradient ascent (Amari, 1998; Sato, 2001; Hoffman et al., 2013) to optimize the global parameters $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$, as it greatly simplifies the update rules while maintaining the same asymptotic convergence properties as classical stochastic gradient. The natural gradient $\tilde{\nabla}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ is obtained by a premultiplication of the ordinary gradient $\nabla\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ with the inverse of the Fisher information of the variational posterior q. See Appendix A.3 for the exact forms of the natural gradients with respect to $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$. To update the parameters $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$, we apply the stochastic natural gradient ascent rule

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t + \rho_t \tilde{\nabla}_{\boldsymbol{\eta}} \mathcal{L}_S(\boldsymbol{\eta}_t, \boldsymbol{\gamma}_t), \quad \boldsymbol{\gamma}_{t+1} = \boldsymbol{\gamma}_t + \rho_t \tilde{\nabla}_{\boldsymbol{\gamma}} \mathcal{L}_S(\boldsymbol{\eta}_t, \boldsymbol{\gamma}_t), \tag{7}$$

7. A similar variational approximation was employed in our earlier development (Yin et al., 2013). The most salient difference is that the `STM-ChooseA` uses significantly fewer variational parameters (and is therefore faster), while still maintaining high experimental accuracy. In particular, for $K \geq 1000$, the choices of $U_0 = 10$ and $U_1 = K/10$ make the SVI algorithm almost 10 times faster than the method of Yin et al. (2013).

where the step size is given by $\rho_t = \tau_0(\tau_1 + t)^{-\kappa}$. To ensure convergence, the τ_0, τ_1, κ are set such that $\sum_t \rho_t^2 < \infty$ and $\sum_t \rho_t = \infty$. The specific values used in our experiments are $\tau_0 = 50$, $\tau_1 = 10000$, and $\kappa = 0.5$. The global update only costs $O(NK)$ time per iteration due to the parsimonious $O(K)$ parameterization of the triangle-generating probability tensor $\mathbf{B}$ (and hence its variational parameter $\boldsymbol{\eta}$) in our STM (Section 3.1).

Initialization. The SVI algorithm described above searches for a local maximum in a highly non-concave variational objective function (Equation 5). In principle, this makes it sensitive to the choice of initialization or seeding, a property shared by other overlapping community detection algorithms (Xie et al., 2013). If the initialization is not too far from the true community structure, then the algorithm will usually find the correct communities. However, a completely random initialization will often merge many neighboring communities into one giant community, which is undesirable.

To address this issue, we conduct the initialization of the SVI algorithm systematically, which is highly effective on all the ground-truth networks we test on (Section 6.1). It consists of two steps. First, we renumber all node indices via the following procedure:

`STM-CANONIZE`:

1. Initialize `MAP` to an empty dictionary and `COUNT` $= 1$. When the algorithm terminates, `MAP`$[i] = a$. In other words, we have renumbered the old node index i to the new node index a.

2. For each edge (i, j) in the edge list:

   (a) If i is not in `MAP`, then assign `MAP`$[i]$ = `COUNT` and `COUNT` = `COUNT` $+ 1$.

   (b) If j is not in `MAP`, then assign `MAP`$[j]$ = `COUNT` and `COUNT` = `COUNT` $+ 1$.

3. For each edge (i, j) in the edge list:

   (a) Output the re-indexed edge (`MAP`$[i]$, `MAP`$[j]$).

This procedure requires only linear-time work in the number of edges M, and executes in less than 10 minutes for all ground-truth networks in our experiments. In addition to renumbering all nodes in the range $[1, N]$, `STM-CANONIZE` often creates many sequences of contiguous node indices $a, a+1, a+2, \ldots, a+b$ such that a is connected to $a+1, a+2, \ldots, a+b$. The rationale is that nodes with numerically close indices are more likely to be close in terms of network distance, and thus are likely to be in the same community. This happens because the input edge list is not randomly ordered in practice, but rather usually groups adjacent edges together—as an example of how this may happen, when an adjacency matrix is converted to an edge list by scanning the rows one at a time, edges connecting to same node are adjacent in the edge list. When `STM-CANONIZE` encounters adjacent edges whose nodes have not been seen before, it gives those nodes contiguous indices.

The second step takes advantage of these contiguous sequences, by initializing contiguous blocks of node indices to different communities: the first N/K nodes are seeded to community 1, the second N/K nodes are seeded to community 2, and so on. Recall that γ_i's are the variational parameters for the mixed-membership vectors θ_i's; to seed node i to

community k, we initialize $\gamma_{i,k} = 10K$ and the remaining non-seeded elements of γ_i to be small, randomly-generated numbers close to 1.

For the variational parameters $\boldsymbol{\eta}$ corresponding to the triangle-generating probabilities $\mathbf{B}$, we initialize them as follows: $[\eta_{xxx,1}, \eta_{xxx,2}] = [1, 3]$, and $[\eta_{xx,1}, \eta_{xx,2}, \eta_{xx,3}] = [2, 1, 1]$ and $[\eta_{0,1}, \eta_{0,2}] = [3, 1]$. This reflects the intuition that three nodes with the same community are likely to form a 3-edge triangle, whereas for other cases (two nodes have the same community or three nodes have distinct community), it is likely to form a 2-edge triangle. Note that there is no need to initialize the local variational parameters $q_{ijk}(x, y, z)$, as they are solved through fixed-point iteration given the current values of $\boldsymbol{\gamma}, \boldsymbol{\eta}$ (Equations 9, 10 and 11). Finally, we fix the hyperparameters of $\boldsymbol{\theta}, \mathbf{B}$ to $\alpha = \lambda = 0.1$.

5. Distributed Implementation for Internet-Scale Networks

In order to apply our SVI algorithm to massive networks, we turn to its distributed implementation. The specific challenges we are facing include:

1. Big data: given that contemporary server machines only contain between 16GB to 256GB of memory, a massive network and its triangular representation cannot fit into the memory of a single machine. For example, the 101-million-node, 2-billion-edge web graph (Section 7) requires over 30GB to simply store it as an adjacency list, and the full triangular representation would be many orders of magnitude larger (in the TB range).

2. Big model: similarly, for a massive network, the STM parameters cannot fit into the memory of a single machine. With $N = 100$ million nodes and $K = 1000$ communities, we would need 400GB just to store the mixed-membership vectors θ_i's.

3. Slow inter-machine communication: typical network speeds range from 1Gbps to 10Gbps, hence inter-machine communication is several orders of magnitude slower than CPU-to-RAM communication. This means that we cannot synchronize parameters as frequently as in the single machine setting.

Unfortunately, the existing Hadoop MapReduce framework (Hadoop, 2012) is not well-suited to implement iterative convergent algorithms such as our SVI, because every map-reduce iteration incurs significant overheads and consequently takes orders of magnitude longer than other systems (Zaharia et al., 2010; Low et al., 2010). Another concern is that the map-reduce programming model does not provide a natural way to store model parameters in a persistent and distributed fashion, a challenge which has recently been addressed by high-performance parameter servers (Li et al., 2013; Ho et al., 2013). Based on these considerations, we develop our distributed-parallel SVI algorithm for STM (Algorithm 1) on top of the Petuum parameter server (Ho et al., 2013; Lee et al., 2014; Dai et al., 2015), a recent framework for general-purpose machine learning on big data and models. The high-level ideas to tackle challenges outlined above are to (1) keep the triangles in an implicit representation and use disk-based access to save machine memory; (2) partition the model over worker machines and exploit parameter sparsity to further reduce memory usage; (3) use a bounded-asynchronous communication technique to strike a balance between parameter accuracy and speed. We now discuss detailed approaches to each idea.

Algorithm 1 Distributed-parallel Stochastic Variational Inference for STM

1: $t = 0$. Initialize the global parameters $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$, and store them in the Petuum parameter server (`www.petuum.org`) for global access by any worker thread in the cluster.
2: Repeat the following steps in parallel until convergence. Parallelization is conducted in a data-parallel fashion: each worker thread in the cluster is responsible for a disjoint set of nodes, and all N nodes are collectively covered by all workers.
   (1) Sample CN triangles to form a mini-batch S. We do this by sampling C pairs of neighbors for each node $i \in \{1, \ldots, N\}$ in parallel (see Section 5.1 for details of implementation).
   (2) Optimize the local parameters $q_{ijk}(x, y, z)$ for all sampled triangles in parallel by Equations 9, 10 and 11.
   (3) Accumulate sufficient statistics for the natural gradients of $\boldsymbol{\eta}, \boldsymbol{\gamma}$, and then discard local parameters $q_{ijk}(x, y, z)$ and the mini-batch S. Since the sufficient statistics are additive, we use the Petuum parameter server to accumulate them in parallel at each worker thread.
   (4) Optimize the global parameters $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ by the stochastic natural gradient ascent rule (Equations 7), and then distribute the new global parameters to the worker threads.
   (5) Update the step size for the next iteration: $\rho_t \leftarrow \tau_0(\tau_1 + t)^{-\kappa}$, $t \leftarrow t + 1$.

5.1 Handling Big Network Data and Trillions of Triangles

As shown in Lemma 1, the number of 2/3-edge triangles is bounded below by $\frac{1}{3}\sum_i \frac{1}{2}(D_i)(D_i - 1)$. This means even a single node with 1 million neighbors, which can occur in very large networks ($N \geq 100$ million nodes and $M \geq 1$ billion edges) with a power-law degree distribution, will contribute half a trillion triangles to the triangular representation and hence require TBs of storage.

Implicit Triangle Representation and Disk-based Storage. Our solution is to keep the triangles in an *implicit* representation, where we leverage the stochastic nature of the SVI algorithm and only sample triangles only when they are needed in step (1) of Algorithm 1 (and discard them afterwards). Specifically, we represent the original network using the following three data structures:

1. An adjacency list `Adj` represented as a dictionary (key-value) data structure. If the a-th neighbor of i is j, then the dictionary contains $\texttt{Adj}[(i, a)] = j$. While this key-value schema seems unorthodox compared to storing the entire neighbor set $\mathcal{N}_i$ as the value, they are easier to store at large scale and possibly can help improve CPU cache efficiency because values j are fixed-byte-size scalars whereas sets of neighbors $\mathcal{N}_i$ can have arbitrary byte length.

2. An edge list `Edg` represented as a set of edge tuples (i, j).

3. A degree list `Deg` represented as a vector, where $\texttt{Deg}[i]$ contains the degree of node i.

In order to conserve machine memory, we store these data structures in a disk-based key-value store—our implementation uses Google leveldb (`www.leveldb.org`), but any other disk database should work as well. Although storing the data on hard disks incurs significant read latencies in principle (around 10 milliseconds per read), we have observed that leveldb's memory cache amortizes latencies across multiple reads very well, and therefore disk-based access is not a bottleneck for our implementation of the SVI algorithm. In our large-scale experiments, leveldb could sustain 100s of thousands of reads per second per machine

(using multiple worker threads) on a standard hard drive. This throughput is more than sufficient for the SVI algorithm for STM. The use of disk-based storage frees up roughly 40GB of memory on each machine, when we run our implementation of the SVI algorithm on the 101-million-node web graph. Note that converting an edge list of a network with M edges to these disk-based data structures only costs $\mathrm{O}(M)$ (for hash-based dictionaries) or $\mathrm{O}(M \log M)$ (for tree-based dictionaries) work, and in practice this step takes only a small fraction of the time required by the SVI algorithm.

Subsampling CN Triangles from the Implicit Representation. We employ the following procedure to subsample a mini-batch S of CN triangles (step (1) in Algorithm 1), based on the aforementioned disk-based data structures.

`STM-SUBSAMPLE`:

- **For each node $i = 1$ to N in parallel:**
  - **For $c = 1$ to C:**
    1. Let $D_i := \texttt{Deg}[i]$.
    2. Draw two *distinct* indices $a, b \in \{1, \ldots, D_i\}$ uniformly at random.
    3. Let $j := \texttt{Adj}[(i, a)]$ and $k := \texttt{Adj}[(i, b)]$.
    4. Check if $(i, j) \in \texttt{Edg}$. If yes, output (i, j, k) as a 3-edge triangle. If no, output (i, j, k) as a 2-edge triangle centered at node i.

Because there is no need to keep all local variational parameters after accumulating all natural gradient sufficient statistics for the global parameters $\boldsymbol{\eta}, \boldsymbol{\gamma}$, we discard the mini-batch S after step (3) of Algorithm 1 in order to save memory. By combining implicit triangle representation and subsampling strategy in this manner, our SVI algorithm for STM avoids having to store trillions of triangles in an explicit form, thus saving TBs (or more) of disk storage and memory space.

Discussion of Subsampling Strategy. Our fast parallel procedure to subsample a mini-batch S of CN triangles may not provide an unbiased estimate of the natural gradient of the global parameters $\tilde{\nabla}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ (Equation 6). The reason is that the CN triangles in S are not necessarily sampled from the space of all triangles uniformly at random—one can see that triangles adjacent to low-degree nodes tend to be more likely to be sampled than triangles attached to high-degree nodes. However, it is worth emphasizing that since low-degree neighbors of a high-degree node are very likely to sample triangles involving that high-degree node, the high-degree nodes are still well represented in the set of subsampled triangles. While this bias due to nonuniform sampling can be corrected by appropriately reweighting the subsampled triangles[8], we have counterintuitively observed that community detection performance is actually *more accurate* without the correction factor. Our hypothesis is that, by not reweighting the triangles, low-degree nodes are given more attention because our subsampling procedure is more likely to select their adjacent triangles, compared to uniform triangle sampling. This improves community detection accuracy on low-degree

8. One reweighting scheme is to divide each triangle's natural gradient contribution by its actual probability of being sampled. Similar techniques were employed to reweight the subsampled node pairs in the assortative MMSB inference algorithm (Gopalan et al., 2012; Gopalan and Blei, 2013).

nodes, which in turn improves overall accuracy because the majority of nodes in real-world scale-free networks are low-degree. For this reason, we choose to not reweight the triangles in our experiments.

Earlier development on triangular modeling (Ho et al., 2012c; Yin et al., 2013) advocated a triangle pre-selection technique called δ-subsampling, where a proper subset of all triangles is subsampled *prior to* inference. Our distributed-parallel SVI algorithm for STM differs in that we *never* pre-select triangles, allowing the inference algorithm to (in principle) access all triangles—thus eliminating one source of approximation. It also avoids the need to tune the parameter δ, the number of triangles to pre-select. From a practical standpoint, our new subsampling strategy is far more memory-efficient, as triangles are immediately discarded after use.

5.2 Handling Big STM Models with Over 100 Billion Parameters

At first estimation, the variational parameters γ_i's for all the mixed-membership vectors θ_i's require NK floating point values to be stored. If $N = 100$ million and $K = 1000$, we would need 100 billion 4-byte floating point values (400GB)[9]. Although the Petuum parameter server discussed in the next subsection can evenly distribute the memory load over all participating machines, it requires additional memory to cache for high performance because the parameter server is a caching system. Thus, the actual memory requirement of SVI algorithm for STM is much more than $NK \times 4$ bytes—we have observed that 400GB of γ_i's would require multiple TBs of memory across all machines.

To reduce memory usage to practical levels, we exploit the observation that most nodes in real networks belong to just a few communities, as seen in ground-truth data (Yang and Leskovec, 2012b). In other words, the mixed-membership vectors θ_i's and the corresponding variational parameters γ_i's are extremely sparse. Therefore, we use dictionary data structures to store γ_i's, and perform two approximations at every iteration of the SVI algorithm in order to reduce the memory requirement:

1. We delete elements of γ_i's that are already close to zero—defined as being less than $2\alpha = 0.2$ in the ground-truth experiments (Section 6). See Section 7.1 for an alternative threshold value used for a 101-million-node web graph.

2. During the global update for each γ_i (Equation 7), we only commit the U_0 largest vector elements in the natural gradient of γ_i and set the rest of elements to be zeros. We use $U_0 = 10$ in our experiments, because empirically most nodes exhibit no more than 10 communities and thus it suffices to retain the 10 most significant communities per node.

When the SVI algorithm terminates, we also output γ_i's in a sparse format in order to conserve hard disk space. Experimentally, we have observed that these approximations incur little impact on community detection accuracy, and they allow us to analyze a 101-million-node network with 1000 communities, using only 500GB of memory spread across 5 machines (i.e., 100GB per machine, which is readily available from cloud compute providers such as Amazon EC2).

9. Note that there are only $O(K)$ variational parameters $\boldsymbol{\eta}$ for the triangle-generating probability tensor $\mathbf{B}$, which is extremely small compared to the size of γ_i's.

5.3 Overcoming Slow Network Interfaces

Our distributed-parallel SVI algorithm for STM (Algorithm 1) adopts a data-parallel scheme: each worker thread in the cluster is responsible for a disjoint set of nodes and all N nodes are collectively covered by all workers, in order to update a single set of shared global parameters γ and η. This requires workers to frequently synchronize all parameter updates to each other. However, due to the higher latency and lower bandwidth of computer network interfaces (e.g., 1Gb or 10Gb Ethernet) compared to internal CPU-to-RAM communication, the workers are often forced to wait for communication to complete, thus wasting as much as $\geq 80\%$ of CPU power (Ho et al., 2013).

In order to alleviate this communication bottleneck, various bounded-asynchronous parameter synchronization systems have been developed (Ho et al., 2013; Li et al., 2013; Dai et al., 2015), which essentially reduce the frequency of communication so as to allow for much higher CPU utilization. The trade-off is that reduced communication leads to *staleness* (i.e., out-of-date values) in the workers' view of the model parameters, which could incur errors in algorithm execution. By using the right "consistency models", most iterative optimization and sampling-based algorithms can be made to theoretically and empirically converge despite staleness. Furthermore, while having stale values of the model parameters could decrease convergence progress per iteration, the increase in CPU utilization more than makes up by enabling more iterations per minute, thus yielding much faster distributed machine learning algorithm execution.

These considerations apply to our SVI algorithm for STM as well, because the global variational parameters γ and η are subject to frequent additive updates from all workers (Equation 7) due to the relatively small mini-batch sizes being used (CN triangles per iteration with $C = 1$ in our experiments). To overcome these challenges, we develop our C++ implementation of the SVI algorithm on top of the Petuum system for scalable distributed machine learning (Ho et al., 2013; Lee et al., 2014; Dai et al., 2015), using its bounded-asynchronous parameter server for data-parallel machine learning programming. Petuum features an machine-learning-specific "consistency model", Stale Synchronous Parallel (SSP), that exploits the concept of *bounded staleness* to speed up distributed computation while still providing theoretical guarantees on the convergence of various machine learning algorithms (e.g., stochastic gradient descent). More formally, a worker reading parameters at iteration c will see the effects of all updates from iteration 0 to $c - s - 1$, where $s \geq 0$ is a user-controlled staleness threshold. In our experiments, we found that configuring the SSP staleness setting to either $s = 0$ or $s = 1$ yielded very good, near-linear scaling from 1 through 5 machines (note that 5 machines are sufficient for performing community detection on a 101-million-node network).

6. Empirical Study on Networks with Ground-truth Communities

In this section, we evaluate the overlapping community detection performance of the SVI algorithm for STM, comparing it to both probabilistic and non-probabilistic baseline algorithms. It is also possible to use our algorithm to perform link prediction, although we do not pursue it in this paper. See Yin et al. (2013) for an appropriate procedure.

Network	# nodes N	# edges M	# communities	Edge density M/N^2	Fraction of closed triangles
DBLP	317,080	1,049,866	13,477	1.044×10^{-5}	0.1283
Amazon	334,863	925,872	75,149	8.257×10^{-6}	0.07925
Youtube	1,134,890	2,987,624	8,385	2.320×10^{-6}	0.002081
Livejournal	3,997,962	34,681,189	287,512	2.170×10^{-6}	0.04559

Table 3: Basic statistics of the networks with ground-truth overlapping communities used in our evaluation. These networks are available at `http://snap.stanford.edu/data/`.

6.1 Ground-truth Data

Because there has been debate over the appropriateness of simulated networks in the community detection task (Leskovec et al., 2009; Yang and Leskovec, 2012b), we performed all our experiments on real-world networks with ground-truth communities provided by Yang and Leskovec (2012b), rather than using synthetic benchmarks such as Lancichinetti-Fortunato-Radicchi (LFR) (Lancichinetti and Fortunato, 2009). The ground-truth communities were constructed based on the publicly available meta-data associated with each network (e.g., self-declared interests, hobbies, and affiliations in social networks such as LiveJournal). The size of these networks ranges from $N \approx 300000$ to 65 million, and the number of created communities ranges from $K \approx 8000$ in a 1.1-million-node network to $K \approx 6.3$ million in a 3-million-node network. However, not all of the communities were well-defined. Thus, Yang and Leskovec (2012b) also provided the top 5000 communities with highest quality in each network, where the quality of a community is measured by averaging over several well-known community scoring functions such as conductance, modularity, and triangle-participation-ratio—the idea being that communities that score well on all functions are very likely to be of high quality. We do not claim these ground-truth communities are the only valid way to decompose the network, and acknowledge that there may exist alternative ways to extract plausible communities from large networks.

Table 3 provides basic statistics that are taken from the original directed form of each network in this evaluation. When our algorithm loaded a network, it converted the network to its undirected form via symmetrization because our STM is a probabilistic network model for undirected triangular motifs. When running those baselines that are able to to exploit directed form, we always provided the original directed network as input.

6.2 Evaluation by Normalized Mutual Information (NMI)

We used the normalized mutual information (NMI) (Lancichinetti and Fortunato, 2009), one of the most widely used measures, for evaluating the quality of discovered overlapping communities. The NMI is on a scale of 0 to 1, with 1 corresponding to a perfect matching with the ground-truth communities. In these ground-truth networks, the top 5000 communities were provided by Yang and Leskovec (2012b) as *hard assignments* of nodes: for a particular community, each node is either in that community or it is not (i.e., no partial membership). Two issues must be addressed before the outputs of different algorithms can be evaluated against the ground truth.

First, because the SVI algorithm for STM outputs the variational parameters γ_i's that are then normalized to produce continuous-valued mixed-membership vectors to represent soft community assignments, we must threshold to obtain hard community assignments

that can be compared to the ground truth. Based on the observation that more than 90% of nodes have no more than 10 ground-truth communities, we chose the threshold value to be 0.1, i.e., community k contains node i if $\hat{\theta}_{i,k} = \gamma_{i,k} / \sum_{k'} \gamma_{i,k'} \geq 0.1$. This allows us to detect up to 10 communities per node. We also applied this thresholding procedure to any baselines that output soft community assignments; baselines that output hard assignments were left as it is.

The second issue is that the top 5000 ground-truth communities only cover a small fraction of the network—in other words, the majority of nodes have missing ground-truth community assignments. However, all the algorithms being evaluated assign every node to at least one community, and the NMI cannot handle missing community assignments. Thus, for every node i without a ground-truth assignment in the top 5000 communities, we removed node i from its corresponding communities discovered by each algorithm. This ensures that the NMI is computed only on nodes found within the top 5000 ground-truth communities. We emphasize that such post-processing is fair because it is performed for all the algorithms being evaluated.

6.3 Experimental Settings

Termination Criterion. We monitored the convergence of the SVI algorithm by computing the variational mini-batch lower bound $\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ (Equation 6) at each iteration, which serves as an unbiased approximation to the true variational lower bound $\mathcal{L}(\boldsymbol{\eta}, \boldsymbol{\gamma})$ (Equation 5). We never compute the true lower bound as it involves all triangles in the network, which would be computationally prohibitive. In our ground-truth experiments, we terminated the algorithm when $\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ decreases for the first time. This signifies that the SVI algorithm is beginning to oscillate, and will not make much further progress. All our trials on ground-truth networks terminated with high-quality results within 200 iterations under this criterion (using $C = 1$ triangle subsampled for each node, per iteration); in other words, the SVI algorithm only had to process $< 200N$ triangles in total. The intuition is that node community memberships are empirically sparse—each node is unlikely to participate in more than a few communities, and thus a small number of subsampled adjacent triangles suffices to determine its community membership.

Baselines. For comparison with the SVI algorithm for STM, we selected baselines that (1) are able to perform overlapping community detection and (2) are scalable enough to analyze 1-million-node networks in at most a few days. These two criteria greatly narrow the list of candidate baselines, and eventually we considered the following four algorithms:

1. `a-MMSB`: the assortative MMSB (Gopalan et al., 2012; Gopalan and Blei, 2013), which requires the number of communities K to be input as a parameter. We used the single-machine C code available at `https://github.com/premgopalan/svinet`, and applied the "link-sampling" subsampling scheme, as recommended by the user manual.

2. `PM`: the Poisson model (Ball et al., 2011), which requires the number of communities K to be input as a parameter. We used the single-machine C code provided by the authors.

3. `SLPA`: the speaker-listener label propagation algorithm (SLPA) (Xie and Szymanski, 2012), which can automatically detect the number of communities in the network, given a threshold $r \in [0, 1]$. We found that the NMI score did not vary significantly with different choices of r, so we fixed $r = 0.25$ in all experiments. The single-machine Java code is available at `https://sites.google.com/site/communitydetectionslpa/ganxis` under the name GANXiS.

4. `SVD+M`: a baseline that first applies a rank-K SVD to the adjacency matrix and then extracts communities from the singular vectors using modularity as a stopping criterion, as proposed in Prakash et al. (2010) and Kang et al. (2011). As code was unavailable from the authors, we wrote our own single-machine MATLAB implementation, based on the multi-threaded `svds()` sparse low-rank SVD function.

Both a-MMSB and PM are mixed-membership models based on adjacency matrix representation of networks; SLPA is a message-passing algorithm, in which community labels are propagated along edges until convergence; SVD+M is a matrix factorization algorithm. We ran all algorithms with their default settings, unless otherwise stated. For algorithms that require the number of communities K as input (including our SVI algorithm for STM), we repeated the experiments for different values of K: 5000, 10000, 15000 and 20000.

Machine Configuration. We used server machines equipped with 128GB RAM and 2 Intel Xeon E5-2450 8-core processors, for a total of 16 CPU cores per machine running at 2.10GHz. We ran the distributed-parallel SVI algorithm using 4 such machines, for a total of 64 cores/worker threads and 512GB distributed RAM. The baselines a-MMSB, PM, and SLPA are all single-machine and single-threaded, while SVD+M is single-machine but multithreaded; we ran all algorithms using one machine with 128GB RAM.

6.4 Experimental Results

We first investigate how the mini-batch size CN influences the performance of the SVI algorithm for STM. Then we compare it to various baseline algorithms in terms of NMI score, runtime, and size of detected communities.

Why Choose $C = 1$? Table 4 shows NMI scores and time to convergence for running the SVI algorithm with $C = 1$ v.s. $C = 10$ on three networks. Note that using $C = 1$ is 5-10 times faster than $C = 10$, while maintaining comparable NMI scores. Accordingly, we set the mini-batch size to $C = 1$ triangle per node, resulting in a total of N triangles per iteration.

NMI Score and Runtime. Table 5 shows NMI scores and runtimes for all the algorithms being evaluated. Performing overlapping community detection at these large scales is extremely computationally intensive, and our distributed-parallel SVI algorithm for STM finished execution in far less time than the baselines. None of the baseline algorithms were able to finish the 4-million-node, 35-million-edge Livejournal network within our experimental limit of 5 days, and the SVD+M algorithm was not able to finish any experiment within this limit[10]. Given a larger compute cluster, we expect our approach to finish more rapidly,

10. The main reason is that high-rank SVD is expensive to compute in MATLAB. The call to `svds()` alone took 4 days to complete on the smallest DBLP network ($N = 317$K) for just $K = 5000$, despite being a multithreaded implementation. Furthermore, `svds()` simply ran out of memory on larger experiments, despite equipped with 128GB RAM.

	NMI		Time to convergence	
Network	$C = 1$	$C = 10$	$C = 1$	$C = 10$
DBLP ($N = 317$K)				
$K = 5000$	0.439	0.451	15min	2.8h
$K = 10000$	0.506	0.505	18min	1.5h
$K = 15000$	0.542	0.535	26min	2.1h
$K = 20000$	0.559	0.553	30min	3.1h
Amazon ($N = 334$K)				
$K = 5000$	0.790	0.791	38min	3.4h
$K = 10000$	0.758	0.771	18min	1.4h
$K = 15000$	0.743	0.752	24min	2.2h
$K = 20000$	0.733	0.739	31min	3.2h
Youtube ($N = 1.1$M)				
$K = 20000$	0.433	0.434	7.6h	93h

Table 4: NMI scores and time to convergence for running the SVI algorithm for STM with $C = 1$ v.s. $C = 10$ on three networks. Using $C = 1$ is nearly as accurate as $C = 10$, while requiring much less computational time.

allowing even larger networks to be analyzed in a matter of hours. The STM SVI algorithm is also memory-efficient, due to sparse data structures in the Petuum parameter server used for sharing the global variational parameters γ_i's. In particular, the largest experiment on the Livejournal network with $K = 20000$ only required 24GB RAM on each of the 4 machines (for a total of 96GB). The a-MMSB inference algorithm implementation does not use sparse storage, which caused it to run out of memory on relatively small experiments, even on a machine with 128GB RAM. We believe this underscores the importance of sparse memory management for large-scale problems (Section 5.2).

In terms of accuracy of recovering ground-truth overlapping communities, the STM SVI algorithm outperformed other mixed-membership models a-MMSB and PM, but had lower NMI scores than SLPA (a message-passing algorithm that is not based on a statistical model). This suggests that mixed-membership network models have some room for improvement. However, there are still two advantages of the STM SVI algorithm compared to SLPA: (1) the STM SVI algorithm finishes execution in far less time on a distributed compute cluster, allowing it to scale to much larger networks; (2) SLPA only outputs hard, binary community assignments, so it may not be suitable for analyses that require soft, probabilistic assignments, which the STM SVI algorithm can provide.

Though faster execution of our STM SVI algorithm hinges on the distributed implementation, we would like to point out that even on smaller networks (N ranges from 10000 to 200000) in the absence of distributed computing, our earlier development (Ho et al., 2012c; Yin et al., 2013) has demonstrated the benefits of triangular modeling over adjacency-matrix-based modeling in terms of scalability and accuracy. The single-machine SVI algorithm on these smaller networks usually converges after several passes on all triangles in the network (4-5 passes at most), and achieves competitive or improved accuracy for latent space recovery and link prediction compared to MMSB (Yin et al., 2013). This is compatible with our observation that, in the distributed setting, at most $200N$ triangles needed to be subsampled and processed to achieve the indicated NMI scores in Table 5.

	NMI					Time to completion				
Network	STM SVI	a-MMSB	PM	SLPA	SVD+M	STM SVI	a-MMSB	PM	SLPA	SVD+M
DBLP ($N = 317$K)										
$K = 5000$	0.439	0.379	0.251	0.581	DNF	15min	17.4h	8.9h	2.6h	> 5 days
$K = 10000$	0.506	0.437	0.294	(16874 coms)	DNF	18min	48h	18h		> 5 days
$K = 15000$	0.542	OOM	0.322		OOM	26min	OOM	51h		OOM
$K = 20000$	0.559	OOM	0.341		OOM	30min	OOM	96h		OOM
Amazon ($N = 335$K)										
$K = 5000$	0.790	0.750	0.483	0.867	DNF	38min	31h	4.4h	3.2h	> 5 days
$K = 10000$	0.758	OOM	0.548	(29021 coms)	DNF	18min	OOM	11h		> 5 days
$K = 15000$	0.743	OOM	0.571		OOM	24min	OOM	22h		OOM
$K = 20000$	0.733	OOM	0.576		OOM	31min	OOM	31h		OOM
Youtube ($N = 1.1$M)										
$K = 5000$	0.374	OOM	0.129	0.424	OOM	2.2h	OOM	76h	21h	OOM
$K = 10000$	0.422	OOM	DNF	(5972 coms)	OOM	3.2h	OOM	> 5 days		OOM
$K = 15000$	0.456	OOM	DNF		OOM	3.9h	OOM	> 5 days		OOM
$K = 20000$	0.433	OOM	DNF		OOM	7.6h	OOM	> 5 days		OOM
Livejournal ($N = 4.0$M)										
$K = 20000$	0.743	OOM	DNF	DNF	OOM	63h	OOM	> 5 days	> 5 days	OOM

Table 5: NMI scores and runtimes for all the algorithms being evaluated. "OOM" means the algorithm ran out of memory and failed, while "DNF" means the algorithm did not finish within a reasonable amount of time (5 days of continuous computation). Note that the SLPA algorithm can automatically select the number of communities K, thus we report only one NMI score and runtime per network (with the number of detected communities in parentheses).

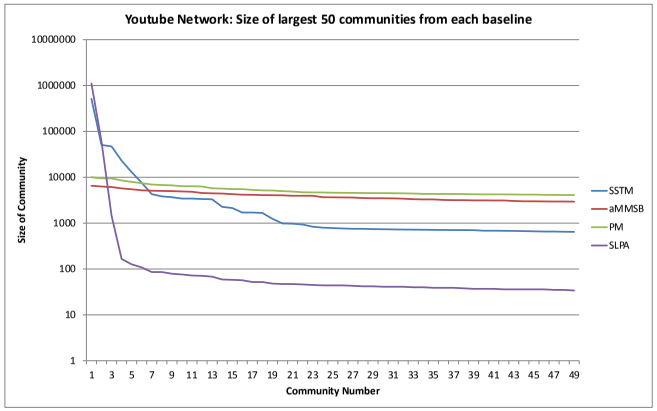


Figure 5: Youtube network: size of the largest 50 detected communities by STM SVI, a-MMSB, PM, and SLPA, plotted on a logarithmic scale.

Size of Detected Communities. Figure 5 plots the size of the largest 50 detected communities in the Youtube network by STM SVI, a-MMSB, PM, and SLPA, on a logarithmic scale. For algorithms that require the number of communities K as input, we set $K = 1000$. It was necessary for us to use the Youtube network with $K = 1000$ communities, in order to obtain a plot of community size for a-MMSB; otherwise, it ran out of memory (see Table 5). We observed that STM SVI and SLPA were able to detect several very large communities containing over 50K nodes, whereas a-MMSB and PM could not recover communities with more than 10K nodes. Furthermore, the vast majority of communities detected by STM SVI and SLPA are much smaller than the ones discovered by a-MMSB and PM. In other words, the distributions of STM SVI and SLPA community size are far more skewed (power-law-like) than a-MMSB's and PM's. We hypothesize that the better NMI performance of STM SVI and SLPA is partly due to their ability to detect very large and very small communities, which are present in real-world networks with power-law behavior (Leskovec et al., 2009). We also conjecture that because larger communities in real-world networks are likely to be edge-sparse, with intra-community edge probability p not much larger than the inter-community edge probability q (Leskovec et al., 2009), it would be theoretically difficult for models such as a-MMSB to detect these large communities (Anandkumar et al., 2014).

7. Empirical Study on a 101-million-node Web Graph

Our SVI algorithm for STM is intended for latent space inference in very large, Internet-scale networks. In that vein, we conclude with a brief analysis of the 101-million-node, 2-billion-edge Subdomain/Host web graph, from the Web Data Commons (available at `http://webdatacommons.org/hyperlinkgraph/`). This web graph was constructed from the Common Crawl 2012 web corpus, which originally contained 3.5 billion web pages and 128

billion hyperlinks. The 3.5 billion web pages were aggregated by their subdomain[11] or host, resulting in a new graph with 101 million nodes/subdomains. An edge was created between two subdomains if at least one hyperlink was found between their aggregated pages, resulting in 2 billion directed edges. When we ran our algorithm, it was treated as an undirected, unweighted network via symmetrization.

Our primary aim is to demonstrate that the our distributed-parallel SVI algorithm for STM finished execution on this massive web graph in an acceptable amount of time on a small cluster—37.3 hours using $K = 1000$ overlapping communities. This scale is 25 times larger than a recent experiment by Gopalan and Blei (2013), who ran the a-MMSB inference algorithm on an $N \approx 4$M patent network using $K = 1000$. We also performed light qualitative analysis to show that the discovered mixed-membership vectors $\hat{\theta}_i$'s reveal sensible insights about the structure of the web graph.

7.1 Experimental Settings

We used the experimental settings that were mostly similar to the experiments on ground-truth networks, with the following exceptions:

Number of Machines. We used 5 machines with 16 CPU cores and 128GB RAM per machine, configured identically to the 4 machines used in the ground-truth experiments. The extra machine was required because the algorithm ran out of memory on 4 machines.

Termination Criterion. On such a large network, the SVI algorithm may take a long time before the variational mini-batch lower bound starts oscillating, which is the termination criterion used in the ground-truth experiments. For this web graph, we observed that by iteration $t = 50$, the lower bound was only changing at the 5th significant figure (relative to the first few iterations). We thus concluded that the algorithm was no longer making significant progress, and stopped the algorithm at $t = 50$ iterations.

Threshold of γ_i's to Maintain Sparsity. As explained in Section 5.2, this massive web graph requires 100 billion floating point values for the variational parameters γ_i's, which is estimated at TBs of memory if stored densely (after accounting for algorithmic overheads). In order to keep the γ_i's sparse enough to fit on 5 machines with 640GB total RAM, we zeroed out any element $\gamma_{i,k} < 10$ at the end of every iteration. Though this is more aggressive than the threshold $2\alpha = 0.2$ used in the ground-truth experiments, it has little impact on the detected communities because of the way we obtain the final hard community assignments based on the inferred $\hat{\theta}_i$'s (Section 6.2). When the SVI algorithm terminated, the vast majority of nodes i had an unnormalized "total variational mass" $\sum_k \gamma_{i,k} > 100$. Thus, any zeroed-out element $\gamma_{i,k}$ would have been normalized to $\hat{\theta}_{i,k} = \gamma_{i,k}/(\sum_{k'} \gamma_{i,k'}) < 0.1$, which is below the threshold of being detected as part of community k.

7.2 Qualitative Analysis

To decide the size of detected communities, we treated the mixed-membership vectors $\hat{\theta}_{i,k}$ as an $N \times K$ continuous-valued matrix. The column sum of the kth column accounts for

11. A subdomain is an Internet host name with 3 or more levels. For example, `www.cmu.edu` and `www.ml.cmu.edu` are considered to be two distinct subdomains.

all partial memberships in community k, and thus is regarded as the effective number of nodes in that community.

To identify significant nodes associated with each community k, we computed a score $s_k(i) = \hat{\theta}_{i,k} \times D_i$ for each node i—its partial membership in community k multiplied by its node degree—and then sorted $s_k(1), \ldots, s_k(N)$ in descending order. Table 6 shows the top-scoring 10 nodes from each of the largest 5 communities, as well as the estimated fraction of 3-edge triangles in each community (i.e., the parameter $B_{kkk,2}$ in Table 2). The largest community is a giant core dominated by well-known websites including youtube.com, google.com, and twitter.com, with a much lower fraction of 3-edge triangles (0.062) than the other communities. The 2nd to 5th largest communities are:

- Community 2: mostly "online stores" that focus on specific products (kitchenware, phones, or cars). However, these are not real online stores as the entries all link to eBay auction pages. The community memberships $\hat{\theta}_{i,k}$ of the top 10 nodes are all perfect (1.0), which suggests that these sites (1) form a compact community with few outside links and (2) are probably copies of each other. It is likely that these sites are meant to increase the visibility of certain eBay auctions—essentially, a form of search engine optimization (SEO). It is unclear whether the auctions themselves are fraudulent or not.

- Community 3: dominated by webpages from Lycos/Tripod, two related companies in the search and website creation business. It also contains w3schools.com, a site that provides instructions for website creation. The top websites in this community do not appear suspicious.

- Community 4: Spanish-language websites, particularly Hispavista, an Internet firm based in Spain. The top websites in this community do not appear suspicious.

- Community 5: dominated by blackmagic.org/com, a website that maintains a large list of websites crawled from the Internet in 2006. It is likely that blackmagic.org/com acts as a hub for the other nodes in the community, which all have much smaller scores $s_k(i)$. The extremely high fraction of 3-edge triangles (0.985) suggests that this community is close to a full clique, or perhaps several cliques connected by one or two bridge nodes. Previous works have shown that such clique-like patterns in web graphs are highly indicative of web duplication or fraud (Kang et al., 2009, 2011); a cursory look at the websites in this community (such as the skimium.* nodes) reveals that many of them are near-exact copies of each other.

We also explored significant nodes that are in 2 or more communities in this web graph (recall that a node i is assigned to community k if $\hat{\theta}_{i,k} \geq 0.1$). Of these 700K websites, the one with the highest node degree, wordpress.org, has a partial membership 0.89 in the giant core and 0.11 in the 25th largest community. By examining the top websites in the 25th largest community, we found several domains of hostgator.com, a website hosting business. Further inspection revealed that the wordpress forums frequently recommend hostgator.com to host wordpress-powered blogs, and similarly, the hostgator support pages explain how to set up a wordpress blog on hostgator itself.

k-th largest community	Fraction of 3-edge triangles	Website	$\hat{\theta}_{i,k}$	$s_k(i)$
1	0.062	youtube.com	0.96	2906030.5
(mass 71038.3K)		wordpress.org	0.89	2102190.9
		en.wikipedia.org	0.93	1899359.3
		gmpg.org	0.92	1638740.7
		tumblr.com	0.94	1096803.3
		twitter.com	0.95	1057107.2
		flickr.com	1.00	931931.7
		serebella.com	1.00	757930.4
		google.com	0.92	738368.1
		top20directory.com	1.00	691007.1
2	0.604	kitchensnstuff.com	1.00	163675.0
(mass 165.7K)		shopping.mia.net	1.00	105814.0
		thenichestorebuilder.com	1.00	57502.3
		generator.mia.net	1.00	56772.0
		phone.mia.net	1.00	53360.0
		seekwonder.com	1.00	44767.9
		hostinglizard.com	1.00	44496.5
		corvette-auction.com	1.00	43633.9
		gotomeeting.mia.net	1.00	43611.0
		cashadvance.mia.net	1.00	43587.9
3	0.481	tripod.lycos.com	0.69	697557.6
(mass 88.7K)		w3schools.com	0.99	499886.4
		domains.lycos.com	0.99	476899.0
		club.tripod.com	0.13	63429.6
		wired.com	0.37	46150.2
		search.lycos.com	0.94	29350.7
		news.lycos.com	0.94	20329.7
		moreover.com	0.15	613.1
		metallica.com	0.09	421.4
		google-pagerank.net	0.56	407.8
4	0.695	hispavista.com	0.80	156246.8
(mass 69.7K)		dominios.hispavista.com	0.94	138596.5
		inmobiliaria.hispavista.com	0.93	138045.6
		globedia.com	0.87	134000.4
		galeon.com	0.84	133910.6
		neopolis.com	0.94	133033.7
		trabajos.com	0.89	132235.4
		horoscopo.hispavista.com	0.95	131548.8
		paginasamarillas.hispavista.com	0.95	131476.2
		software.hispavista.com	0.94	131310.0
5	0.985	blackmagic.org	0.60	103660.8
(mass 65.2K)		blackmagic.com	0.60	102844.9
		opera.com	0.04	2265.0
		skimium.fr	0.88	1289.3
		skimium.co.uk	0.97	1181.0
		skimium.es	0.97	1167.6
		skimium.nl	0.96	1140.9
		skimium.it	0.94	1137.8
		skimium.be	0.94	1111.3
		inetgiant.com	0.24	665.6

Table 6: Subdomain/Host web graph: the 10 top-scoring nodes in each of the largest 5 communities by mass (the effective number of nodes in the community). The score $s_k(i)$ is computed as $\hat{\theta}_{i,k} \times D_i$, i.e., the partial membership of node i in community k multiplied by its node degree. For each community, we also report the estimated fraction of 3-edge triangles. A quick look at the top websites in communities 2 and 5 reveals suspicious behavior (see main text for details).

8. Conclusion and Discussion

Massive Internet-scale networks are technically challenging to explore, manipulate, and visualize. One approach to understanding the structural and functional properties of massive networks is to perform latent space inference for overlapping community detection. At the time of writing, there are few inference algorithms that can scale to hundreds of millions of nodes in order to detect thousands of distinct communities, and almost none of them are based on a probabilistic model.

In this work, starting with the mixed-membership class of statistical models as a foundation, we systematically tackle the statistical and computational challenges associated with Internet-scale network inference. Our major contributions include: (1) characterization of the network as a more compact—and arguably more salient for community detection—triangular representation; (2) design of a parsimonious generative model with only $O(K)$ instead of K^2 or K^3 parameters; (3) construction of an efficient structured stochastic variational inference algorithm; (4) a careful consideration of the distributed implementation in order to handle big network data, big network model, and slow inter-machine communication. The resulting distributed-parallel STM SVI algorithm is able to detect 1000 communities from a 100-million-node network in 1.5 days on just 5 cluster machines, and we believe that networks with $N > 1$ billion nodes can be analyzed with a sufficiently large cluster, thus opening the door to Facebook-scale social networks and beyond.

We would like to end with some directions and open issues for future research. One limitation of STM is that it only applies to undirected and unweighted networks (and triangular motifs), although we have demonstrated good community detection NMI scores relative to baselines. We believe that accounting for edge direction and weights could further improve accuracy, which are more common in non-social-network domains such as biology and finance. Sensitivity to initialization is another issue that we would like to address: while the `STM-CANONIZE` procedure worked well in our experiments, we believe that there are more systematic ways to address the issue, such as augmenting the variational inference algorithm with split-merge-like search moves that can change many node community assignments at once. Finally, we believe that triangular modeling could also be extended to other probabilistic network model for overlapping community detection, such as the Community-Affiliation Graph Model (AGM) by Yang and Leskovec (2012a).

Acknowledgments

The authors would like to thank two anonymous reviewers for their helpful comments that improved the manuscript. This work was supported by AFOSR FA9550010247, NIH 1R01GM093156, and DARPA FA87501220324 to Eric P. Xing. Qirong Ho is supported by an A-STAR, Singapore fellowship. Junming Yin is partly supported by a Ray and Stephanie Lane Research Fellowship from CMU and a research grant from the Center for Management Innovations in Health Care at the Eller College of Management.

Appendix A. Details of Stochastic Variational Inference

In this section, we provide details of our SVI algorithm, including the exact form the variational lower bound, the exact and approximate local update equations, and the natural gradients with respect to the global parameters.

A.1 Exact Form of the Variational Lower Bound

The variational lower bound (Equation 3) of the log marginal likelihood of the triangular motifs based on the variational distribution (Equation 1) is

$$\log p(\mathbf{E} \mid \alpha, \lambda) \geq \mathbb{E}_q[\log p(\mathbf{E}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{B} \mid \alpha, \lambda)] - \mathbb{E}_q[\log q(\mathbf{s}, \boldsymbol{\theta}, \mathbf{B})] \doteq \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\gamma}) \tag{8}$$

$$\begin{aligned}
&= \mathbb{E}_q[\log p(B_0 \mid \lambda)] - E_q[\log q(B_0 \mid \eta_0)] + \sum_{x=1}^{K} \Big\{ \mathbb{E}_q[\log p(B_{xx} \mid \lambda)] - \mathbb{E}_q[\log q(B_{xx} \mid \eta_{xx})] \Big\} \\
&+ \sum_{x=1}^{K} \Big\{ \mathbb{E}_q[\log p(B_{xxx} \mid \lambda)] - \mathbb{E}_q[\log q(B_{xxx} \mid \eta_{xxx})] \Big\} + \sum_{i=1}^{N} \Big\{ \mathbb{E}_q[\log p(\theta_i \mid \alpha)] - \mathbb{E}_q[\log q(\theta_i \mid \gamma_i)] \Big\} \\
&+ \sum_{(i,j,k) \in I} \Big\{ \mathbb{E}_q[\log p(s_{i,jk} \mid \theta_i) + \log p(s_{j,ik} \mid \theta_j) + \log p(s_{k,ij} \mid \theta_k)] \Big\} \\
&+ \sum_{(i,j,k) \in I} \Big\{ \mathbb{E}_q[\log p(E_{ijk} \mid s_{i,jk}, s_{j,ik}, s_{k,ij}, \mathbf{B})] - \mathbb{E}_q[\log q(s_{i,jk}, s_{j,ik}, s_{k,ij} \mid \phi_{ijk})] \Big\}.
\end{aligned}$$

The first two lines of Equation 8 represent the global terms $g(\boldsymbol{\gamma}, \boldsymbol{\eta})$ that depend only the global variational parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\eta}$, whereas the last two lines are a summation of the local terms $\ell(\phi_{ijk}, \boldsymbol{\gamma}, \boldsymbol{\eta})$, one for each triangle.

A.2 Local Update (Exact and Approximate)

For each sampled triangle (i, j, k) in a mini-batch, the exact local update algorithm updates all the K^3 entries of ϕ_{ijk}, and then renormalizes them to sum to one. The K^3 entries of ϕ_{ijk} can be segregated into three broad categories: ϕ_{ijk}^{xxx}, ϕ_{ijk}^{xxy}, ϕ_{ijk}^{xyz}, which correspond to the following cases:

1. ϕ_{ijk}^{xxx} corresponds to the case that all nodes choose the same community: $s_{i,jk} = s_{j,ik} = s_{k,ij} = x$;
2. ϕ_{ijk}^{xxy} (or ϕ_{ijk}^{xyx}, or ϕ_{ijk}^{yxx}) corresponds to the case that two nodes choose the same community: $s_{i,jk} = s_{j,ik} = x$ and $s_{k,ij} = y$ (with similar variations for ϕ_{ijk}^{xyx} and ϕ_{ijk}^{yxx});
3. ϕ_{ijk}^{xyz} corresponds to the case that all nodes choose different communities: $s_{i,jk} = x$, $s_{j,ik} = y$, $s_{k,ij} = z$.

The update equations for each category are:

1. For $x \in \{1, \ldots, K\}$,

$$\phi_{ijk}^{xxx} \propto \exp\Big\{ \mathbb{E}_q[\log B_{xxx,2}] \mathbb{I}[E_{ijk} = 4] + \mathbb{E}_q[\log(B_{xxx,1}/3)] \mathbb{I}[E_{ijk} \neq 4] + \mathbb{E}_q[\log \theta_{i,x} + \log \theta_{j,x} + \log \theta_{k,x}] \Big\}. \tag{9}$$

2. For $x, y \in \{1, \ldots, K\}$ and $x \neq y$,

$$\phi_{ijk}^{xxy} \propto \exp\Big\{\mathbb{E}_q[\log B_{xx,3}]\mathbb{I}[E_{ijk}=4] + \mathbb{E}_q[\log B_{xx,2}]\mathbb{I}[E_{ijk}=3] + \mathbb{E}_q[\log(B_{xx,1}/2)]\mathbb{I}[E_{ijk}=1 \text{ or } 2] \\ + \mathbb{E}_q[\log\theta_{i,x} + \log\theta_{j,x} + \log\theta_{k,y}]\Big\}. \tag{10}$$

   The update equations for ϕ_{ijk}^{xyx} and ϕ_{ijk}^{yxx} are similar to ϕ_{ijk}^{xxy} (up to a trivial rearrangement of variables), thus we omit their details.
3. For distinct $x, y, z \in \{1, \ldots, K\}$,

$$\phi_{ijk}^{xyz} \propto \exp\Big\{\mathbb{E}_q[\log B_{0,2}]\mathbb{I}[E_{ijk}=4] + \mathbb{E}_q[\log(B_{0,1}/3)]\mathbb{I}[E_{ijk}\neq 4] + \mathbb{E}_q[\log\theta_{i,x} + \log\theta_{j,y} + \log\theta_{k,z}]\Big\}. \tag{11}$$

The above are the *exact* update equations for ϕ_{ijk}, and they require $\mathrm{O}(K^3)$ run-time per triangle (i, j, k). The update equations for the $\mathrm{O}(K)$ "mixture-of-deltas" approximation (described in Section 4.2) are almost exactly equivalent to the exact $\mathrm{O}(K^3)$ update procedure, with two minor modifications: (1) we simply zero out entries (a, b, c) that are not in the chosen set $\mathcal{A}$, and (2) we renormalize the chosen entries $(a, b, c) \in \mathcal{A}$ amongst themselves so that they sum to 1. This follows because the "mixture-of-deltas" variational distribution is essentially a categorical or multinomial distribution with some elements constrained to be zero.

A.3 Global update

The global update (Equation 7) in our SVI algorithm requires computing the natural gradient $\tilde{\nabla}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$. For clarity, we decompose $\tilde{\nabla}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ over each part of $\boldsymbol{\eta}, \boldsymbol{\gamma}$. The natural gradient with respect to $\boldsymbol{\eta}$ is:

- For $x \in \{1, \ldots, K\}$,

$$\tilde{\nabla}_{\eta_{xxx,1}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\left[\sum_{(i,j,k)\in S} q_{ijk}(x,x,x)\mathbb{I}[E_{ijk}\neq 4]\right] - \eta_{xxx,1}, \tag{12}$$

$$\tilde{\nabla}_{\eta_{xxx,2}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\left[\sum_{(i,j,k)\in S} q_{ijk}(x,x,x)\mathbb{I}[E_{ijk}=4]\right] - \eta_{xxx,2}. \tag{13}$$

- For $x \in \{1, \ldots, K\}$,

$$\tilde{\nabla}_{\eta_{xx,1}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\Bigg[\sum_{(i,j,k)\in S}\sum_{y:y\neq x}\Big(q_{ijk}(x,x,y)\mathbb{I}[E_{ijk}=1,2] + q_{ijk}(x,y,x)\mathbb{I}[E_{ijk}=1,3] \\ + q_{ijk}(y,x,x)\mathbb{I}[E_{ijk}=2,3]\Big)\Bigg] - \eta_{xx,1}, \tag{14}$$

$$\tilde{\nabla}_{\eta_{xx,2}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\Bigg[\sum_{(i,j,k)\in S}\sum_{y:y\neq x}\Big(q_{ijk}(x,x,y)\mathbb{I}[E_{ijk}=3] + q_{ijk}(x,y,x)\mathbb{I}[E_{ijk}=2] \\ + q_{ijk}(y,x,x)\mathbb{I}[E_{ijk}=1]\Big)\Bigg] - \eta_{xx,2}, \tag{15}$$

$$\tilde{\nabla}_{\eta_{xx,3}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\left[\sum_{(i,j,k)\in S}\sum_{y:y\neq x}\Big(q_{ijk}(x,x,y) + q_{ijk}(x,y,x) + q_{ijk}(y,x,x)\Big)\mathbb{I}[E_{ijk}=4]\right] - \eta_{xx,3}. \tag{16}$$

- For the sole η_0 parameter:

$$\tilde{\nabla}_{\eta_{0,1}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\left[\sum_{(i,j,k)\in S}\sum_{(x,y,z):x\neq y\neq z} q_{ijk}(x,y,z)\mathbb{I}[E_{ijk}\neq 4]\right] - \eta_{0,1}, \tag{17}$$

$$\tilde{\nabla}_{\eta_{0,2}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \lambda + \frac{m}{|S|}\left[\sum_{(i,j,k)\in S}\sum_{(x,y,z):x\neq y\neq z} q_{ijk}(x,y,z)\mathbb{I}[E_{ijk}=4]\right] - \eta_{0,2}. \tag{18}$$

The natural gradient $\tilde{\nabla}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma})$ with respect to $\boldsymbol{\gamma}$ is, for each $i = 1, \ldots, N$ and $x = 1, \ldots, K$,

$$\tilde{\nabla}_{\gamma_{i,x}}\mathcal{L}_S(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \alpha + \frac{m}{|S|}\Bigg[\sum_{(j,k):(i,j,k)\in S}\sum_{y,z} q_{ijk}(x,y,z) + \sum_{(j,k):(j,i,k)\in S}\sum_{y,z} q_{jik}(y,x,z) \\ + \sum_{(j,k):(j,k,i)\in S}\sum_{y,z} q_{jki}(y,z,x)\Bigg] - \gamma_{i,x}. \tag{19}$$

References

Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307):761–764, 2010.

E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

L. Akoglu, H. Tong, B. Meeder, and C. Faloutsos. PICS: Parameter-free identification of cohesive subgroups in large attributed graphs. In *SIAM International Conference on Data Mining*, pages 439–450, 2012.

S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor approach to learning mixed membership community models. *Journal of Machine Learning Research*, 15:2239–2312, 2014.

R. Balasubramanyan and W. W. Cohen. Block-LDA: Jointly modeling entity-annotated text and entity-entity links. In *SIAM International Conference on Data Mining*, pages 450–461, 2011.

B. Ball, B. Karrer, and M. E. J. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3):036103, 2011.

A. Beutel, A. Kumar, E. E. Papalexakis, P. P. Talukdar, C. Faloutsos, and E. P. Xing. FlexiFaCT: Scalable flexible factorization of coupled tensors on hadoop. In *SIAM International Conference on Data Mining*, pages 109–117, 2014.

P. Bickel, D. Choi, X. Chang, and H. Zhang. Asymptotic normality of maximum likelihood and its variational approximation for stochastic blockmodels. *Annals of Statistics*, 41(4): 1922–1943, 2013.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

L. Bottou. Stochastic learning. *Advanced Lectures on Machine Learning*, pages 146–168, 2004.

L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes. In *International Conference on Computer Vision*, pages 1–8, 2007.

J. Chang and D. M. Blei. Relational topic models for document networks. In *International Conference on Artificial Intelligence and Statistics*, pages 81–88, 2009.

W. Dai, A. Kumar, J. Wei, Q. Ho, G. Gibson, and E. P. Xing. High-performance distributed ML at scale through parameter server consistency models. In *AAAI Conference on Artificial Intelligence*, pages 79–87, 2015.

L. Dietz, S. Bickel, and T. Scheffer. Unsupervised prediction of citation influences. In *International Conference on Machine Learning*, pages 233–240, 2007.

Facebook. Anatomy of facebook, January 2013. URL `www.facebook.com/note.php?note_id=10150388519243859`.

W. Fu, L. Song, and E. P. Xing. Dynamic mixed membership blockmodel for evolving networks. In *International Conference on Machine Learning*, pages 329–336, 2009.

B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *International Conference on Computer Vision*, pages 670–677, 2009.

G. Ghoshal, V. Zlatić, G. Caldarelli, and M. E. J. Newman. Random hypergraphs and their applications. *Physical Review E*, 79:066118, 2009.

P. Gopalan and D. M. Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.

P. Gopalan, D. Mimno, S. Gerrish, M. Freedman, and D. M. Blei. Scalable inference of overlapping communities. In *Neural Information Processing Systems*, pages 2249–2257. 2012.

M. Granovetter. The strength of weak ties. *American Journal of Sociology*, 78(6):1360–1380, 1973.

T. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(Suppl 1):5228–5235, 2004.

F. Guo, S. Hanneke, W. Fu, and E. P. Xing. Recovering temporally rewiring networks: A model-based approach. In *International Conference on Machine Learning*, pages 321–328, 2007.

Hadoop. Apache Hadoop. `http://hadoop.apache.org/`, 2012.

M. S. Handcock, A. E. Raftery, and J. M. Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A*, 170(2):301–354, 2007.

Q. Ho, L. Song, and E. P. Xing. Evolving cluster mixed-membership blockmodel for time-varying networks. In *International Conference on Artificial Intelligence and Statistics*, pages 342–350, 2011.

Q. Ho, J. Eisenstein, and E. P. Xing. Document hierarchies from text and links. In *International Conference on World Wide Web*, pages 739–748, 2012a.

Q. Ho, A. Parikh, and E. Xing. A multiscale community blockmodel for network exploration. *Journal of the American Statistical Association*, 107(499), 2012b.

Q. Ho, J. Yin, and E. P. Xing. On triangular versus edge representations — towards scalable modeling of networks. In *Neural Information Processing Systems*, pages 2132–2140, 2012c.

Q. Ho, J. Cipar, H. Cui, J.-K. Kim, S. Lee, P. B. Gibbons, G. Gibson, G. R. Ganger, and E. P. Xing. More effective distributed ML via a stale synchronous parallel parameter server. In *Neural Information Processing Systems*, pages 1223–1231, 2013.

P. Hoff, A. Raftery, and M. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

M. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

D. Hunter, S. Goodreau, and M. Handcock. Goodness of fit of social network models. *Journal of the American Statistical Association*, 103(481):248–258, 2008.

U. Kang, C. E. Tsourakakis, and C. Faloutsos. PEGASUS: A peta-scale graph mining system implementation and observations. In *International Conference on Data Mining*, pages 229–238, 2009.

U. Kang, B. Meeder, and C. Faloutsos. Spectral analysis for billion-scale graphs: Discoveries and implementation. In *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 13–25, 2011.

C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *AAAI Conference on Artificial Intelligence*, pages 381–388, 2006.

D. Krackhardt and M. Handcock. Heider vs Simmel: Emergent features in dynamic structures. In *Conference on Statistical Network Analysis*, pages 14–27, 2007.

A. Lancichinetti and S. Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80 (1):016118, 2009.

C. Lee, F. Reid, A. McDaid, and N. Hurley. Detecting highly-overlapping community structure by greedy clique expansion. In *Workshop on Social Network Mining and Analysis held in Conjunction with the International Conference on Knowledge Discovery and Data Mining*, pages 33–42, 2010.

S. Lee, J. K. Kim, X. Zheng, Q. Ho, G. A Gibson, and E. P. Xing. Primitives for dynamic big model parallelism. *CoRR*, abs/1406.4580, 2014.

J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 177–187, 2005.

J. Leskovec, K. J Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

M. Li, L. Zhou, Z. Yang, A. Li, F. Xia, D. G. Andersen, and A. Smola. Parameter server for distributed machine learning. In *NIPS Workshop on Parallel and Large-scale Machine Learning*, 2013.

Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. GraphLab: A new parallel framework for machine learning. In *Uncertainty in Artificial Intelligence*, 2010.

Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. Distributed GraphLab: A framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment*, 5(8):716–727, 2012.

K. Miller, T. Griffiths, and M. I. Jordan. Nonparametric latent feature models for link prediction. In *Neural Information Processing Systems*, pages 1276–1284. 2009.

R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

M. Morris, M. Handcock, and D. Hunter. Specification of exponential-family random graph models: terms and computational aspects. *Journal of Statistical Software*, 24(4):1548, 2008.

R. Nallapati, A. Ahmed, E. P. Xing, and W. W. Cohen. Joint latent topic models for text and citations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 542–550, 2008.

M. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

M. Newman and J. Park. Why social networks are different from other types of networks. *Physical Review E*, 68(3):036122+, 2003.

G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

J. Parkkinen, J. Sinkkonen, A. Gyenge, and S. Kaski. A block model suitable for sparse graphs. In *International Workshop on Mining and Learning with Graphs*, 2009.

B. A. Prakash, A. Sridharan, M. Seshadri, S. Machiraju, and C. Faloutsos. EigenSpokes: Surprising patterns and scalable community chipping in large graphs. In *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 435–448, 2010.

I. Psorakis, S. Roberts, M. Ebden, and B. Sheldon. Overlapping community detection using Bayesian non-negative matrix factorization. *Physical Review E*, 83(6):066114, 2011.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *Uncertainty in Artificial Intelligence*, pages 487–494, 2004.

M. Sato. Online model selection based on the variational Bayes. *Neural Computation*, 13 (7):1649–1681, 2001.

G. Simmel and K. Wolff. *The Sociology of Georg Simmel*. Free Press, 1950.

A. Singh and G. Gordon. A bayesian matrix factorization model for relational data. In *Uncertainty in Artificial Intelligence*, pages 556–563, 2010.

D. Stasi, K. Sadeghi, A. Rinaldo, S. Petrovic, and S. Fienberg. β models for random hypergraphs with a given degree sequence. In *International Conference on Computational Statistics*, 2014.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

F. Wang, T. Li, X. Wang, S. Zhu, and C. Ding. Community discovery using nonnegative matrix factorization. *Data Mining and Knowledge Discovery*, 22(3):493–521, 2011.

D. Watts and S. Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393: 440–442, 1998.

J. Xie and B. Szymanski. Towards linear time overlapping community detection in social networks. In *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 25–36, 2012.

J. Xie, S. Kelley, and B. Szymanski. Overlapping community detection in networks: the state of the art and comparative study. *ACM Computing Surveys*, 45(4), 2013.

Yahoo. Webscope from yahoo! labs, January 2013. URL `http://webscope.sandbox.yahoo.com/catalog.php?datatype=g`.

J. Yang and J. Leskovec. Community-affiliation graph model for overlapping network community detection. In *International Conference on Data Mining*, pages 1170–1175, 2012a.

J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. In *ACM SIGKDD Workshop on Mining Data Semantics*, pages 3:1–3:8, 2012b.

J. Yin, Q. Ho, and E. P. Xing. A scalable approach to probabilistic latent space inference of large-scale networks. In *Neural Information Processing Systems*, pages 422–430. 2013.

M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: cluster computing with working sets. In *USENIX Conference on Hot Topics in Cloud Computing*, 2010.

Journal of Machine Learning Research 17 (2016) 1-23 Submitted 5/15; Revised 3/16; Published 4/16

Patient Risk Stratification with Time-Varying Parameters: A Multitask Learning Approach

Jenna Wiens WIENSJ@UMICH.EDU
Computer Science & Engineering
University of Michigan
Ann Arbor, MI

John Guttag GUTTAG@CSAIL.MIT.EDU
Department of EECS
Massachusetts Institute of Technology
Cambridge, MA

Eric Horvitz HORVITZ@MICROSOFT.COM
Microsoft Research
Redmond, WA

Editor: Benjamin M. Marlin, C. David Page, and Suchi Saria

Abstract

The proliferation of electronic health records (EHRs) frames opportunities for using machine learning to build models that help healthcare providers improve patient outcomes. However, building useful risk stratification models presents many technical challenges including the large number of factors (both intrinsic and extrinsic) influencing a patient's risk of an adverse outcome and the inherent evolution of that risk over time. We address these challenges in the context of learning a risk stratification model for predicting which patients are at risk of acquiring a *Clostridium difficile* infection (CDI). We take a novel data-centric approach, leveraging the contents of EHRs from nearly 50,000 hospital admissions. We show how, by adapting techniques from multitask learning, we can learn models for patient risk stratification with unprecedented classification performance. Our model, based on thousands of variables, both time-varying and time-invariant, changes over the course of a patient admission. Applied to a held out set of approximately 25,000 patient admissions, we achieve an area under the receiver operating characteristic curve of 0.81 (95% CI 0.78-0.84). The model has been integrated into the health record system at a large hospital in the US, and can be used to produce daily risk estimates for each inpatient. While more complex than traditional risk stratification methods, the widespread development and use of such data-driven models could ultimately enable cost-effective, targeted prevention strategies that lead to better patient outcomes.

Keywords: risk stratification, time-varying coefficients, multitask learning, *Clostridium difficile*, healthcare-associated infections

1. Introduction

Over recent years, there has been enormous growth in 1) our capacity to gather clinically relevant data and 2) the availability of such data sets. The collection of these data, in particular electronic health records (EHRs), holds out the promise of using machine learning to build models that can be harnessed to improve patient outcomes. Transforming patient data into actionable knowledge presents a barrage of pragmatic and technical challenges. But if we are successful in addressing these challenges, the knowledge embedded in these data has the potential to revolutionize clinical medicine.

One way in which these data can be leveraged is in the development of accurate data-driven models for predicting potentially avoidable adverse outcomes and using such predictions to guide interventions aimed at reducing the probability of these outcomes. The hypothesis is that we can extract from the data generalizable information that can help accurately identify a patient's *future* pathological states. If pathologies are predicted far enough in advance, then it may be possible for healthcare workers to intervene. Such targeted interventions could, in turn, lead to better patient outcomes.

In recent years, there has been a significant amount of research effort devoted to using clinical data to predict patient outcomes (Shoeb and Guttag, 2010; Syed and Rubinfeld, 2010; Syed et al., 2011; Chia et al., 2012; Saria et al., 2010; Saeed et al., 2011; Kleinberg and Hripcsak, 2011; Aboukhalil et al., 2008; Kansagara et al., 2011). We focus on the specific task of predicting which patients in a hospital will acquire an infection with *Clostridium difficile* (*C. difficile*), a largely preventable adverse outcome (Yokoe et al., 2008). *C. difficile* is a type of bacteria that takes over a patient's gut when normal flora get wiped out (often from receipt of antimicrobials). *C. difficile* infection (CDI) can lead to severe diarrhea and intestinal diseases (e.g., colitis), or even death. The infection is often treated with specific antimicrobials: oral vancomycin and metronidazole (and less frequently, fidaxomicin). However, it is estimated that approximately 20% of cases relapse within 60 days (Pépin et al., 2005). The incidence of CDI in the US is estimated at 200,000 cases per year (Dubberke et al., 2009); this is on par with the number of new cases of invasive breast cancer discovered each year in the US (DeSantis et al., 2014).

Infection with *C. difficile* is a type of healthcare-associated infection (HAI). HAIs are a serious problem in healthcare facilities across the world. It is estimated that, on any given day, HAIs affect approximately 1 in every 25 inpatients in US acute care hospitals (Magill et al., 2014). In addition to *C. difficile*, other common HAIs include ventilator-associated pneumonia, surgical site infection, and infections with methicillin-resistant *Staphylococcus aureus* (MRSA) and vancomycin-resistant *Enterococcus* (VRE). Though many risk factors are well-known (e.g., healthcare-associated exposure, age, underlying disease, etc.), HAIs continue to be a significant problem throughout the world (Klevens et al., 2007). In recent years there have been numerous articles citing our inability to prevent HAIs (Miller et al., 2011; Umscheid et al., 2011; Sievert et al., 2013). We hypothesize that one of the reasons HAIs remain so stubbornly prevalent is because we lack an effective clinical tool for accurately measuring patient risk. In this work, we chose to focus on infections with *C. difficile*, one of the most prevalent HAIs (Miller et al., 2011).

We take a data-centric approach to the problem of developing a model to predict a patient's daily risk of acquiring an infection with *C. difficile*. We leverage the contents of EHRs from over 50,000 patient admissions from a single hospital. These clinical data contain information regarding medications, procedures, in-hospital locations, healthcare staff, lab results, measurements of vitals, patient demographics, patient history and admission details.

We seek a mapping from this information describing a patient to an estimate of the patient's probability of acquiring an infection.

Automated patient risk stratification, based on the contents of the patient's EHR, can serve several purposes. Firstly, risk-stratification models can help clinicians match high-risk patients with the appropriate interventions, monitoring policies, or therapies. In the absence of effective risk stratification, widespread implementation of known interventions (e.g., isolating patients or performing specialized analyses of antibiotic regimens) is prohibitively expensive. Secondly, data-driven models can help generate hypotheses regarding potential risk factors, in turn improving our understanding of the disease. For example, the model could identify "hot-spots" within a hospital that could benefit from additional environmental cleanings. The construction of a predictive model can also help to frame new scientific hypotheses through the identification of discriminatory observations. Such insights can lead to the pursuit and confirmation of causal relationships. Thirdly, such models could aid in designing more efficient clinical trials by identifying a study population at higher risk for disease, increasing the fraction of patients expected to test positive in the trial. This could significantly reduce the cost of a clinical trial without compromising the statistical power of the study.

Learning accurate risk-stratification models from EHR data presents a number of technical challenges. Two main issues we focus on in our work include the high dimensionality of the problem and the complex temporal dependencies among the variables. There can be thousands of variables representing each day of a patient admission and it is likely that many of these variables affect a patient's risk of CDI. Moreover, many of these variables change over time. These time-varying data suggest that as a patient spends time in the hospital, his/her actual risk of CDI will vary. Furthermore, how these variables affect risk is likely to change over time. Recent efforts on building models for identifying patients at high risk of acquiring a CDI have ignored these issues. Prior work on risk-stratification models for CDI has centered on the consideration of a small number of risk factors selected by clinical experts and time-invariant parameters.

Our hospital-specific approach to patient risk stratification for CDI, produces *daily* estimates of patient risk. The novel aspects of our work and our main contributions are outlined as follows:

- We move beyond known risk factors to leverage the entire structured contents of the EHR. Our model is based on thousands of extracted binary variables, many of which are hospital-specific (e.g., the locations of patient rooms within the hospital).
- We include both time-varying and time-invariant variables and we explicitly consider the evolution of patient risk during admission, when estimating current risk.
- We develop a novel multitask learning approach to modeling the time-varying effects of risk factors, based on the domain adaptation techniques presented in Daumé III (2007).
- We propose an evaluation scheme that is representative of how the model will be applied in practice. In contrast to previous work, we do not evaluate how our model performs at a single point in time, but rather how the model performs when applied to each day of a patient's admission.

When tested on a holdout set consisting of 24,399 patient admissions from a single year, our proposed model achieved an area under the receiver operating characteristic curve (AUROC) of 0.81 (95%CI 0.78-0.84) and consistently outperformed a baseline model with time-invariant coefficients, for patients with longer risk periods. We have shown that our algorithm can be integrated into the health record system of a hospital and can be used to automatically calculate daily probabilities of risk of CDI for every adult inpatient. These estimates could in turn be used for the selective targeting of high-risk patients with specific interventions that could lead to changes in clinical practice and ultimately a reduction in the incidence of CDI and HAIs. Beyond HAIs, we believe the multitask approach for learning the time-dependent structure of risk factors is a promising methodology for building predicting models for other adverse outcomes.

2. Background and Related Work

In previous work, risk-stratification models for CDI considered no more than a dozen risk factors identified by experts and many of these risk factors pertained to time-invariant features (i.e., observations that do not change over the course of the hospitalization) and researchers ignored changes in patient risk over time (Tanner et al., 2009; Dubberke et al., 2011; Garey et al., 2008; Krapohl, 2011). In our work, we have shown how leveraging the entire structured contents of the EHR leads to significantly better predictions compared to a model based solely on a set of known risk factors easily extracted from the EHR (Wiens et al., 2014). Moreover, we have incorporated both time-varying (e.g., current medications) and time-invariant (e.g., gender) risk factors into the model. Dubberke et al. (2011) also consider a risk prediction model based on both variables collected at the time of admission and throughout the admission. However, they ignore any trend in patient risk. In contrast, we have investigated different methods for incorporating the evolution of patient risk into the current risk estimate, transforming the problem into a time-series classification task (Wiens et al., 2012a). These extensions lead to significant improvements in patient risk stratification.

While our previous work has touched on time-varying variables, to date, risk stratification models for CDI have only considered time-invariant model parameters. That is, although the patient changes over time, and so does the estimate of risk, the models used to compute patient risk do not. This approach does not allow the relative importance of risk factors to change over time as the patient spends more time the hospital. Models to date have not explicitly considered the time-dependence of such factors as a patient's susceptibility and exposure over time. We argue that, in addition to changes in patient state and hospital conditions, the relative importance of risk factors may change during an admission. For example, the important of evidence drawn from a patient's history may diminish as the patient spends more time in the hospital. We propose a methodology that can capture and represent such rich temporal dynamics of the relevance of risk factors in real-world healthcare settings.

Models with time-varying parameters have been studied in other contexts like survival analysis (Fan and Zhang, 2008). Over the years, standard approaches to survival analysis, like Cox proportional hazards, have been extended to include time-dependent parameters (Hastie and Tibshirani, 1993). Extensions typically involve the addition of interaction terms

between features and time-varying functions (Gray, 1992; Murphy and Sen, 1991; Zucker and Karr, 1990; Tian et al., 2005; Sun et al., 2009). In many cases, the user must specify these functions. Researchers have developed non-parametric extensions, but these methods can be computationally inefficient for large, high-dimensional data sets. In practice, researchers often end up partitioning time into intervals and analyze each time period with a simple model. Our proposed approach is similar in the sense that we break the problem up into multiple tasks. However, instead of learning the models independently, we propose learning the models jointly using a multitask learning (MTL) framework.

MTL is a popular branch of machine learning that leverages the intrinsic relatedness among different tasks (Caruana, 1997). It has been studied extensively in many different applications, including healthcare (Caruana, 1996). As an example, Zhou et al. (2014) employ multitask learning in a patient risk stratification context for handling missing features values. In other work, (Zhou et al., 2011) employed an MTL framework in their work on Alzheimer's disease progression, centering on the prediction of the cognitive functioning of patients at different times in the future. They consider each time point as a different regression task and learn the tasks jointly. The authors employ a temporal group LASSO regularization framework, which ensures that only a small subset of the variables are chosen while penalizing large deviations of predictions at neighboring time points. Similarly, we treat each time point of prediction as a different task. However, instead of predicting multiple points into the future based only on the covariates at baseline, we predict risk each day based on time-varying covariates. In our application, a patient's risk of CDI is affected by several external risk factors that can change over the course of the hospitalization e.g., exposure to disease. We incorporate these changes at each time point, since ignoring them is likely to lead to inaccurate predictions.

3. Study Population

We considered all adult inpatient admissions to a large private hospital in the US over a two year period. The statistical analysis of retrospective medical records was approved by the Institutional Review Board of the Office of Research Integrity of the hospital network's research institute. We considered patients admitted on or after 2011-04-12 and discharged on or before 2013-04-12 (n=73,454). We exclude admissions in which the patient was discharged or tested positive for *C. difficile* before the end of the third day (n=24,389) and admissions for which the patient had a positive test result for *C. difficile* within the 14 days preceding the current admission (n=59). This resulted in 49,006 unique admissions. These criteria exclude many predictable low-risk patients with shorter stays, and focus on those patients who we believe acquire the infection during the current hospital admission (as opposed to those who are already infected at the time of admission). Our final study population is described in Table 1.

	Study Population (n=49,006)
Age, median (IQR)	60 (46-73)
Female gender, %	55.25
Hospital Service (%)	
medicine	47.85
cardiology	11.98
surgery	9.60
obstetrics	8.12
psychiatry	5.4
LOS (days), median (IQR)	5.4 (3.8-4.4)
CDI (%)	
current visit	1.02
1-year history	1.11
any history	1.54

Table 1: Demographics of our study population.

CDI cases are typically defined as healthcare-associated if they occur within 48 hours of the time of admission. In our work, we exclude patients who test positive before the end of the third calendar day of admission. By defining the cutoff as the end of the third calendar day, we ensure that the minimum cutoff of 48 hours is achieved, while allowing for simultaneous predictions for every patient (at the end of the day). A uniform time of prediction makes sense from a clinical perspective, since it streamlines the risk-stratification process. Most recently, the Centers for Disease Control and Prevention (CDC) updated their definition of HAIs to include positive test results that occur on the third calendar day of admission (CDC, 2015). While we do not consider these cases here, this work could easily be extended to do so.

4. Methods

In this section, we begin with the problem setup and define notation that will be used throughout the paper. We then describe the feature extraction and learning algorithms used to train the risk prediction model.

4.1 Problem Setup & Notation

The task at hand is to learn a model to accurately predict an inpatient's risk of acquiring CDI during the current hospitalization. Predictions are made daily; we consider time at the granularity of a day, and each day t of an admission is represented by a d dimensional binary feature vector: $\mathbf{x}_t \in \{0,1\}^d$. While we focus on a binary representation of the data, we incorporate both continuous and discrete variables as discussed in Section 4.2. Since each admission in our study population consists of multiple days, the i^{th} patient admission is represented by a series of feature vectors: $(\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, ..., \mathbf{x}_{m_i}^{(i)})$, where m_i varies across patient admissions since the length of a visit varies (note: $m_i \geq 3$ in our study population). We use boldface notation to denote vectors.

In addition to a series of feature vectors, each patient admission is also associated with a binary label $y \in \{+1, -1\}$. Each day of a visit in which the patient eventually tests positive is labeled $+1$ and -1 otherwise. Thus each patient admission $p^{(i)}$ consists of m_i (feature vector, label) pairs:

$$p^{(i)} = \{(\mathbf{x}_t^{(i)}, y_t^{(i)})\}_{t=1}^{m_i}$$

For patients who do not test positive, m_i is equal to one less than the length of the visit in calendar days. We do not consider the day of discharge in our analysis since by the

time a patient is discharged the prediction is meaningless. For patients who eventually test positive, we consider a patient admission up to and including the day before the day of the positive test result. Finally, our data set is defined as:

$$\mathcal{D} = \{p^{(i)}\}_{i=1}^{n}$$

where n represents the number of unique patient admissions in the data set. Note that a patient may be represented multiple times in our data set if he/she has multiple admissions during the time period we consider.

Data	Description
Admission details	Admission details (e.g., date and time of admission, date and time of discharge, and type of visit) and other information pertaining to the admission such as the financial class code, the source of the admission, the hospital service, and the attending doctor are extracted for each patient admission.
Patient demographics	Information pertaining to patient demographics such as age at the time of admission, gender, race, marital status, and city of residence are extracted. Aside from age, all data in this table are categorical.
Laboratory results	Results pertaining to ordered laboratory tests are extracted. Each entry in the database table is associated with a patient admission, an observation identifier, an observation value, an observation time, a reference range (e.g., 120-200 for cholesterol) and an abnormal flag (e.g., H=high, L=low, C=critical, or empty=normal). We represent time-stamped laboratory results based on the observation identifier and the associated flag.
Diagnoses	Patient diagnoses are encoded using ICD-9 codes (NCHS, 2008). Patient visits can be associated with multiple ICD-9 codes; in our data the average visit (including outpatient visits) is associated with two distinct ICD-9 codes. ICD-9 codes, widely used for billing purposes, can get coded well after a patient is discharged (Iezzoni, 1990). For this reason, we do not use the codes associated with a patient's current visit in our model. Instead, we consider only the codes from a patient's most recent hospital admission.
Medications	Orders for medications are associated with an admission identifier, an 8-digit medication identifier, and a start/stop time. Each medication identifier is associated with a medication, a dosage and a form (e.g., in solution). Since the dosage and form are encoded in the 8-digit medication identifier, we represent patient medications using only this identifier.
Locations	For each hospital admission we have time-stamped location data. Location data refer to the patient's location within the hospital. Locations are collected at both the unit and the room level. Using these time-stamped data we can trace a patient's path through the hospital.
Vitals	Each entry in the vitals table corresponds to a visit, an observation identifier (e.g., "BPSYSTOLIC" for systolic blood pressure), an observation value, a reference range, an abnormal flag, and an observation timestamp. When extracting information about vitals for a patient we encode the observations the same way we encode laboratory results, i.e., as a concatenation of the observation identifier and an abnormal flag (e.g., "BPSYSTOLIC_H" for high systolic blood pressure).
Procedures	In the EHR, procedures are encoded using both Current Procedural Terminology (CPT) codes and ICD-9 procedure codes. Each row in the procedures table records a procedure, an admission identifier, and a procedure timestamp. Since both coding systems are used to describe procedures, in our analysis we consider both CPT and ICD-9 codes.

Table 2: Relevant information extracted from the EHR.

4.2 Feature and Label Extraction

As Paxton et al. (2013) state, there are many challenges that come with working with EHR data in research. Addressing these challenges requires careful consideration of the data and the intended application. Moreover, electronic health information systems will continue to change and therefore it is important that researchers take this into consideration when

developing models based on EHR data. In our work these challenges motivated a simple, flexible, data-driven approach to extracting and representing EHR data.

4.2.1 Data Extraction

We represent each day of a patient's admission with a single feature vector, $\mathbf{x}_t$ for $t = 1...m_i$, composed of both *time-invariant* features collected at the time of admission and *time-varying* features collected over the course of each day. The time-invariant features aim to capture the baseline state of each patient while the time-varying features capture changes in patient state during the hospital admission. In the EHR, data are stored across different tables in several databases. We describe the relevant variables and how they are stored and extracted from the EHR in Table 2.

Each patient admission (i.e., encounter) is represented by a unique identifier, in addition each patient is associated with a unique identifier. These unique identifiers allow us to retrieve information across hospital databases for each admission, and across time for multiple admissions pertaining to the same patient. For each patient admission in our study population, we extract knowledge pertaining to the EHR data described above. We augment the data pertaining to the current admission with data extracted from previous admissions including diagnoses and medications.

4.2.2 Feature Engineering

The majority of the extracted data pertain to categorical features, e.g., medications or in-hospital locations. Vitals and laboratory results are also represented using categorical variables as described in Table 2. This eliminates the need to define our own cutoffs for discretization, since the cutoffs are encoded directly in the database using "reference ranges." Data pertaining to diagnoses were coded as ICD-9 codes, a hierarchical classification system with 13,000 unique codes. For our application, we do not expect that this level of granularity is informative. Diagnostic codes are used largely for billing purposes and are not timestamped, therefore their utility is limited. Given these limitations we focus on only diagnostic codes associated with the previous visit, and consider only the highest level of the codes.

We also consider a small number of continuous and discrete variables (e.g., age and statistics related to previous hospitalizations). We map all of these data to binary variables resulting in a high-dimensional feature space. Doing so allows us to later capture some of the nonlinear relationships that may be present in the data without using a nonlinear classifier. We discretize all continuous variables (except for age) using cutoffs based on quintiles from the training data.

From the laboratory data described in Table 2 and the in-hospital location data we were able to extract information regarding patient exposure to the disease throughout the hospitalization. *Colonization pressure* aims to measure the number of patients in a specific unit of the hospital colonized or infected with a particular disease. We define colonization pressure as in Wiens et al. (2014) and measure patient exposure over time based on the patient's location during the hospital admission. We measure exposure at both the hospital-wide and unit-wide level.

We focus on capturing events at the temporal resolution of a day. Thus, we do not consider the order of events within a single day. For example, we know which medications were ordered each day but not the temporal ordering of when the medications were taken. We handle cases where the same variable, e.g., blood pressure, is observed multiple times during a day by simply including all relevant values that were observed when building the daily feature vector. For our particular task this resolution suffices, though it may not be optimal.

4.2.3 Ground Truth

We label each example in our data set as either positive or negative depending on the laboratory data pertaining to positive stool tests for toxigenic *C. difficile*, as obtained during the hospital admission. Patient admissions with a positive test result are labeled positive in our data and negative otherwise. These laboratory results are time-stamped and thus we also noted the calendar day in which the patient tested positive. In the data set, patients are only tested if they exhibit symptoms. Thus, we expect the laboratory results to be highly sensitive and specific. However, since not all patients are tested every day for the disease there is a small possibility that some patients may have acquired an infection and yet were never tested.

4.3 Learning to Predict Daily Risk

We produce daily estimates of patient risk using a two-stage process as in Wiens et al. (2012a). However, in contrast to our previous work, our model in the first stage is not static but varies over the course of the hospitalization, incorporating time-varying model parameters. The first stage produces initial estimates of daily risk and the second stage incorporates risk estimates on previous days in order to capture the variation in risk over time. We explain both stages in detail below.

4.3.1 Baseline Approach

Given the set of labeled feature vectors $\mathcal{D} = \{(\mathbf{x}_t^{(i)}, y_t^{(i)})_{t=1}^{m_i}\}_{i=1}^{n}$ representing each day of a patient admission we can learn a classifier $\theta \in \mathbb{R}^d$, linear in $\mathbf{x}$, using $L2$-regularized logistic regression:

$$\min_{\theta} \frac{1}{2}\theta^T\theta + C\sum_{i=1}^{n}\sum_{t=1}^{m_i} log(1 + e^{-y_t^{(i)}\theta^T\mathbf{x}_t^{(i)}}) \quad (1)$$

Once we have learned θ we can produce an initial estimate of the i^{th} patient's risk on day t, $\hat{y}_t^{(i)} \in [0,1]$:

$$\hat{y}_t^{(i)} = \frac{1}{1 + e^{(-\theta^T\mathbf{x}_t^{(i)})}} \quad (2)$$

This formulation simply pools all training examples together and learns a single model, θ ignoring the index t. As a patient spends additional time in the hospital, we expect the factors contributing to patient risk to change. Therefore, we extend the learning framework to produce a model that changes as the patient spends more time in the hospital.

4.3.2 Proposed Approach

We divide the problem into T learning tasks, splitting $\mathcal{D}$ into separate tasks: $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, ..., \mathcal{D}_T$. Where the task $\mathcal{D}_j$ is the task associated with the τ_j time period for $j = 1, 2, 3, ..., T$. The data are split into different tasks according to t, the index of the day of the visit, such that each task contains roughly an equal number of examples.

We could learn a separate model for each task independently of the others, but in doing so we would be limiting ourselves to using only $\frac{1}{T}$ of the original training data available. Moreover, the tasks themselves are related in time and thus are not independent. While the parameters may vary from one day to another, we do not expect large fluctuations in time. Therefore, we use a multitask learning framework to leverage the inherent relatedness among the different tasks and take advantage of the entire corpus of the training data.

We extend the $L2$-regularized logistic regression framework presented above to incorporate multiple tasks. We specifically chose $L2$ regularization over other regularization frameworks (e.g., L1) since many factors contributing to the risk of CDI are not well understood and effective interventions and preventative measures for reducing patient risk are still being studied. Thus we are more interested in capturing/identifying perhaps novel risk factors, than selecting a sparse subset. Moreover, given the collinearity present in the data, there's a risk that $L1$ regularization could "select-out" such known risk factors. We want to be careful that we do not exclude known risk factors in the final model. Even if other factors that are highly correlated with known risk factors remain in the model, clinicians are unlikely to use or trust a model that does not consider known risk factors. We employ domain adaptation techniques from Daumé III (2007), remapping each feature vector $\mathbf{x}_t^{(i)}$ to a feature vector $\in \{0,1\}^{d(T+1)}$ using the mapping function $\Phi(\mathbf{x}_t^{(i)})$:

$$\Phi(\mathbf{x}_t^{(i)}) = \ [\mathbf{x}_t^{(i)}, \langle\mathbf{0}\rangle^{j-1}, \mathbf{x}_t^{(i)}, \langle\mathbf{0}\rangle^{T-j}] \quad \forall\ t \in \tau_j \quad \text{for } j = 1, 2, 3, ..., T$$

The new feature vectors consist of two copies of the original feature vector, padded with zeros $\mathbf{0} = [0_1, 0_2, 0_3, ..., 0_d]$. Here the notation $\langle\mathbf{0}\rangle^k$ represents k concatenated copies of the zero vector $\mathbf{0}$.

We then learn the regression parameters $\theta \in \mathbb{R}^{d(T+1)}$ using Equation (1) simply replacing $\mathbf{x}_t^{(i)}$ with $\Phi(\mathbf{x}_t^{(i)})$ in the objective function. One can decompose θ into $T+1$ vectors each in $\mathbb{R}^d$, the original dimensionality of the problem, $\theta = [\theta_0, \theta_1, \theta_2, ..., \theta_T]$. Here, θ_0 corresponds to a vector of shared feature weights since it is based on data from all days, where as θ_1 is based on only data from $t \in \tau_1$ and so on.

By substituting $\Phi(\mathbf{x}_t^{(i)})$ for $\mathbf{x}_t^{(i)}$ in Equation (2), the risk of patient i on day $t \in \tau_j$ becomes proportional to $(\theta_0 + \theta_j)^T\mathbf{x}_t^{(i)}$. Writing the function this way shows how learning the models jointly results in T different models all with a shared component θ_0.

$$\theta'_j = \ \theta_0 + \theta_j \quad \text{for } j = 1, 2, 3, ..., T$$

In general, we can estimate the risk of a new patient day $\mathbf{x}_t^{(i)}$ using Equation 2, but replacing θ with θ'_j, without having to remap the feature vector.

4.3.3 Temporal Smoothing

Applying the model described above to a patient's data results in a single estimate of risk for each day $\hat{y}_t^{(i)}$ of a patient's visit. This estimate is based on both the baseline state of the patient (as captured by the time-invariant features in $\mathbf{x}_t^{(i)}$) and the measured time-varying variables. In previous work, we showed that this *snapshot* approach to measuring patient risk, ignores important information contained in the evolution of patient risk. Thus, we incorporate risk estimates from previous days using a cumulative moving average. Given the initial risk estimates, the predicted risk for patient i on day t is calculated as $risk_t^{(i)} = \frac{\hat{y}_1+...+\hat{y}_t}{t}$. This biases new estimates toward the estimates from previous days; while large fluctuations in patient risk in close temporal proximity are possible, they are unlikely. In earlier work, we considered a formulation based on a weighted average in which days closer to the current day receive more weight. However, we found these methods did not yield better estimates than a simple cumulative average (Wiens et al., 2012b). The approach considered here is equivalent to the *RP+Average* approach described in Wiens et al. (2012a), the simplest of the investigated approaches that significantly outperformed the snapshot approach.

4.4 Risk Stratification Model Evaluation - Daily Predictions

In the clinical literature, risk stratification models are often evaluated at a single point in time e.g., two days before an index event or at the time of admission (Tanner et al., 2009; Dubberke et al., 2011). Evaluating a model's ability to risk stratify patients at the time of admission is fine. However, a patient's risk is likely to change over the course of the hospitalization. Evaluating a model at a specific point in time, e.g., n days before an index event, brings to the fore two additional issues. First, such an evaluation requires you to define an index event for negative patients. For risk stratification for CDI the index event is often defined as the first positive test result for patients who test positive and the day of discharge otherwise. However, the task of distinguishing between patients about to test positive for CDI and patients about to be discharged from the hospital is relatively easy since the patients tend to look quite different in the feature space. More importantly, the ability to distinguish such patients is of little clinical utility. Second, evaluating a model at a specific point in time does not yield an accurate representation of how the model will perform in a clinical setting. While such an evaluation scheme is useful for comparing classifier performance, in practice, clinicians have no way of knowing when a patient is n days from an index event.

In a clinical setting, we expect to apply the risk stratification model to each day of a patient's visit. This results in multiple predictions for each patient admission, one corresponding to each day of the admission. We consider a model evaluation scheme that takes all of these predictions into consideration, yet still yields a single measure of performance.

One could imagine a validation scheme in which the performance of a classifier is evaluated for each day independently. While complete, this evaluation still lacks relevance from a clinical perspective since it is not clear how to interpret the utility of a classifier that correctly classifies a patient m days out of a total of n days. Here, we consider an evaluation scheme that is driven by the use case of the model. We assume that once a patient is identified as high-risk s/he will receive some form of an intervention (e.g., relocated to

a private room or special stewardship of the patient's antibiotic protocol) and that this intervention will last for a certain period of time determined by the physicians treating the patient (e.g., 10 days or for the remainder of the visit). Thus, while the predicted risk remains low, we continue making daily predictions, each day deciding whether or not to intervene. However, once the predicted risk exceeds some threshold, we classify the patient as high-risk and discontinue making predictions, since the question of whether or not to intervene becomes irrelevant once physicians have intervened.

Thus, while the model's daily predictions are allowed to fluctuate (from low to high and high to low), when evaluating the model we choose a single decision threshold. We apply this decision threshold to each day of a patient's visit, up to the day before a positive test result is observed or the day before the day of discharge. If the patient's daily estimated risk ever exceeds the decision threshold they are classified as high risk, otherwise they are classified as low risk. This mimics the primary way we expect the model to be used in practice. By taking the maximum prediction for each patient and sweeping over different values of the decision threshold, we can evaluate the model in terms of the area under the receiver operating characteristic curve (AUROC). The AUROC alone is not enough to quantify the performance of the model; since there is high class-imbalance, we also consider the area under the precision recall curve (AUPR). We estimate 95% confidence intervals for these performance measures by applying a bootstrap method. Finally, in addition to these performance metrics, we also evaluate how *far in advance* we correctly identify positive cases. This is an important measure of performance that is often overlooked, but has crucial implication regarding the utility of the model in real-world clinical practice. The earlier we can identify a patient as high-risk, the earlier we can intervene, with the goal of reducing the likelihood of an adverse outcome for a patient and also reducing the spread of the pathogen.

5. Experiments and Results

Employing the methods described in the previous section, we learned and validated a risk stratification model for identifying inpatients at high-risk of acquiring an infection with *C. difficile* throughout their hospital admissions using data from our study population.

5.1 Learning the Risk Model

Our feature extraction yielded close to 10,000 binary variables for each patient day. To reduce the dimensionality, we filter out features that do not occur in at least 1% of the training set. This resulted in a lower dimensional feature vector, where each day is represented by a vector of 905 binary variables. The remaining variables are presented in Table 3. As shown in Table 3 the majority of the features pertain to laboratory results and medications, both time-varying variables.

We split the data into a training set and a holdout set based on time, training on data from the first year, and validating our model on data from the second year. The training data consisted of patient admissions from 2011-04-12 to 2012-04-11, totaling 190,675 visit days pertaining to 24,607 unique visits. Within the training data, 258 admissions had a positive test for *C. difficile* resulting in 2,608 training days with a positive label. To mitigate the influence of patients already showing symptoms, we removed patient days corresponding

Category	Feature Name	#Binary Features	Original Format
Patient History	Previous Medications	160	Categorical
	Previous Diagnoses	16	Categorical
	Number of Hospital Visits (90 days)	3	Continuous
	Avg. LOS of Hospital Visits (90 days)	6	Continuous
	Total LOS of Hospital Visits (90 days)	6	Continuous
	History of C. diff	1	Categorical
	1yr History of C. diff	1	Categorical
Patient Demographics	Age	5	Continuous
	Marital Status	5	Categorical
	Race	4	Categorical
	Gender	1	Categorical
	Financial Class	8	Categorical
	City of Residence	13	Categorical
Admission Details	Admission Month	12	Categorical
	Admission Year	3	Categorical
	Admission Type	4	Categorical
	Hospital Service	12	Categorical
	Admission Source	7	Categorical
	Attending Doctor	14	Categorical
	Expected Surgery	1	Categorical
Daily Admission Details	Laboratory Results	267	Categorical
	Medications	274	Categorical
	Vitals	24	Categorical
	Locations Units/Rooms	38	Categorical
	Procedures	4	Categorical
	Unit Exposure	6	Continuous
	Hospital Exposure	5	Continuous
	Day of Admission	5	Continuous

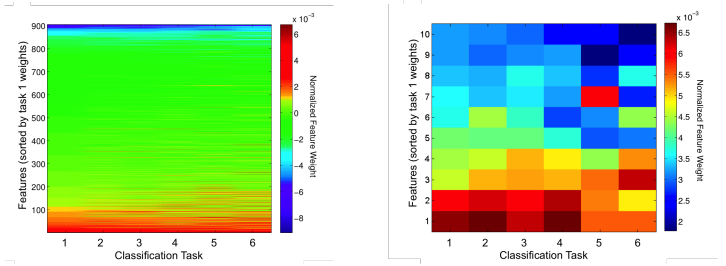
Table 3: High-level description of final features included in the model. The final feature vector consisted of 905 binary features belonging to four different categories.

to the day of and the day before the positive test result. In addition, when training the classifier, we randomly subsampled the data such that no patient contributed more than three days of the data to the training set. This is similar to reweighting samples such that each patient contributes equally to the overall classifier. If we had not done this, some patients would have been represented up to 10 times more often than other patients. Given the small number of positive examples, patients with longer visits could have significantly biased the classifier. In turn, this could have resulted in overfitting to patients with longer visits.

We selected the number of tasks T, and the corresponding temporal intervals τ_j for $j = 1, ..., T$ based on the number of training examples available for each interval. For our data, this resulted in six distinct tasks, corresponding to six distinct time periods: $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4, \mathcal{D}_5, \mathcal{D}_6$. We learned a model for each time period using $L2$-regularized logistic regression and the multitask learning framework described in Section 4.3.2. To select the hyperparameter C in (1), we performed repeated five-fold cross validation on the training data, choosing a setting that maximized the AUROC. The model parameters, i.e., θ, were solved for using LIBLINEAR (Fan et al., 2008). This resulted in six different models, $\theta'_t \in \mathcal{R}^{905}$:

$$\theta'_t = \begin{cases} \theta_0 + \theta_1 & : t \in [1] \\ \theta_0 + \theta_2 & : t \in [2] \\ \theta_0 + \theta_3 & : t \in [3] \\ \theta_0 + \theta_4 & : t \in [4, 5] \\ \theta_0 + \theta_5 & : t \in [6, 9] \\ \theta_0 + \theta_6 & : t \in [10, \infty) \end{cases}$$

θ'_t is allowed to deviate from the shared model θ_0 based on the data collected during each time period. Thus, the relative importance of risk factors is allowed to vary over time. Figure 1 (a) shows the extent to which the weights vary across time. The columns correspond to the different time periods (i.e., tasks) and the rows correspond to the different features. The features are sorted in descending order according to their weight on the first day. The color of each cell is related to the weight of the corresponding feature for the specified task, normalized by the sum of the absolute value of all weights for that task. All cells corresponding to features that have high positive weight are red and those with high negative weight are dark blue. If the relative importance of the weights remains constant over time, each column would appear identical to the first column (i.e., solid horizontal bands of color). However, as Figure 1 (a) shows, the relative importance of features changes. Still, as we expected, we see great deal of continuity in the feature ranking across time periods as all of the models share a common component.



(a) All features (b) The top ten features

Figure 1: The changing relative importance of features over time. For each time period τ_j for $j = 1...6$ (i.e., task), the features are ranked according to the feature weight for the task associated with τ_1. The color represents the normalized feature weight for each task.

Figure 1 (b) shows that, even among the top ten features (from θ'_1), there are changes in the relative importance of features over time. In Figure 1 (b) the first two features become less important over time, while the third feature becomes more important. Table 4 lists the five features with the greatest positive weight in θ'_1 through θ'_6. Note that initially the most important feature is the patient's one year history of CDI. As a patient spends more time in the hospital, this feature loses importance relative to the location of the patient in the hospital.

Ranking	1	2	3	4	5
θ'_1	1 Yr. hist. of CDI	Hist. of CDI	Daily Units:XX	Temp.:High	Prev. Meds:Sevelamer
θ'_2	1 Yr. hist. of CDI	Hist. of CDI	Daily Units:XX	Temp.:High	Service:MED
θ'_3	1 Yr. hist. of CDI	Hist. of CDI	Daily Units:XX	Temp.:High	Prev. Meds:Sevelamer
θ'_4	1 Yr. hist. of CDI	Hist. of CDI	Daily Units:XX	Temp.:High	Mean Platelet Vol.:normal
θ'_5	Meds: Pantoprazole	1 Yr. hist. of CDI	Daily Units:XX	Hist. of CDI	Mean Platelet Vol.:normal
θ'_6	Daily Units:XX	1 Yr. Hist. of CDI	Temp.:High	Hist. of CDI	Service:MED

Table 4: We show features with greatest weight for tasks 1 through 6, using color to highlight certain trends.

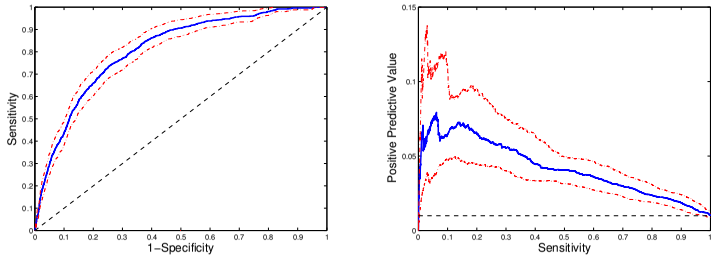
In Table 5, we note the 25 features with the greatest weight according to the shared model i.e., θ_0. We are not surprised that patient history of CDI appears at the top of this list. The medications that appear in Table 5 include drugs administered to patients receiving kidney dialysis, drugs for the treatment of high blood pressure and heart disease, and proton pump inhibitors. When interpreting these weights, it is important to keep in mind that many of the features in our model are highly correlated. These features may be directly or indirectly linked with an increased risk of CDI. Further analysis could generate hypotheses about causal relationships that could be tested in a randomized controlled trial.

Rank	Feature Index	Feature Name	Feature Description	Shared Weight
1	905	OneYear_History	positive test for toxigenic C. diff in past year	0.2472
2	904	All_History	positive test for toxigenic C. diff ever in the past	0.2314
3	124	daily_units:XX	medicine patient care unit	0.2084
4	427	daily_vitals:temporal_h	temperature oral high	0.1885
5	867	daily_meds:63604250	pantoprazole 40mg Inj	0.1508
6	44	v_hospital_service:MED	medicine hospital service	0.1466
7	605	prev_meds:63713135	sevelamer 800 mg Tab	0.1418
8	674	daily_meds:63715254	vitamin B comp w/C, FA Tab	0.1321
9	234	daily_labs:wbc_h	white blood cell count high	0.1320
10	475	prev_meds:63616924	vancomycin 1 gm/250 mL NaCl 0.9%	0.1177
11	269	daily_labs:mpv_	mean platelet volume measured	0.1175
12	433	daily_vitals:bgnas_	oxygen flow rate (nasal cannula)	0.1158
13	418	daily_labs:bun_h	blood urea nitrogen high	0.1094
14	74	attendingdoctornumber:xxxx	attending (anonymized)	0.1088
15	615	prev_meds:63708390	lisinopril 10 mg Tab	0.1081
16	590	prev_meds:63715676	zolpidem 5 mg Tab	0.1077
17	434	daily_vitals:tempax_	temperature axillary	0.1075
18	292	daily_labs:k_l	Potassium Lvl low	0.1059
19	587	prev_meds:63715254	vitamin B comp w/C, FA Tab	0.1038
20	591	prev_meds:63744403	pharmacy comment	0.1037
21	441	daily_vitals:bgpet_l	end tidal CO2 low	0.1036
22	506	prev_meds:63608947	metroNIDAZOLE 500 mg/100 mL 0.9% NaCl	0.1024
23	719	daily_meds:63713259	simvastatin 40 mg Tab	0.1007
24	267	daily_labs:phos_l	Phosphorus Lvl low	0.1003
25	688	daily_meds:63707897	K phos-Na phos Oral Pwdr	0.0990

Table 5: Features with greatest "shared" weight. Hospital unit and staff identifiers have been removed.

5.2 Evaluating the Model

In previous work, risk stratification models have been evaluated at a single point in time. Here, we employ a more realistic evaluation methodology (as described above). We believe this evaluation provides a more accurate representation of how the model will perform in a clinical setting.

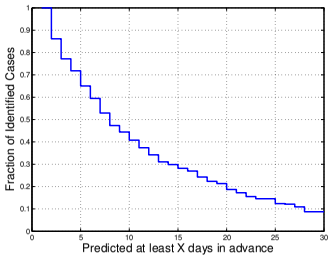


(a) Receiver Operating Characteristic Curve (b) Precision Recall Curve

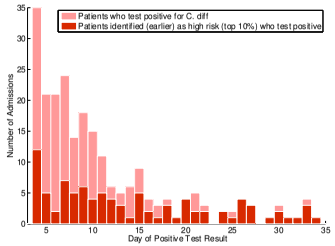
Figure 2: We plot two performance curves generated by applying the risk stratification method described in the previous section to the set of held-out patient admissions. We achieve an AUROC of 0.81 (95%CI 0.78-0.84) and a AUPR of 0.04 (95%CI 0.03-0.05). The 95% CI are represented by the red dashed lines. The performance of a random classifier is given by the black dashed lines. (A classifier no better than random achieves an AUROC=0.5 and AUPR=0.01)

We applied the model to the set of patient admissions held out for validation, which consisted of patient admissions from 2012-04-12 to 2013-04-12 and was composed of 24,399 admissions of which 242 had a positive test result for *C. difficile*. When validating the model, we did not subsample the test data as we did with the training data. Each patient admission in the held out set has at least three daily predictions of risk, since we considered only patients who were still present in the hospital at the end of the third day. However, we do not start applying the decision threshold until the end of the third day (given our exclusion criteria). Also, we do not evaluate the model on the final day of the admission (the day in which the patient was discharged) as such a prediction would be meaningless for guiding in-hospital interventions. Recall that the task is to predict a patient's probability of acquiring CDI during the current hospitalization and that predictions are made at the end of the day.

Figure 2 shows the ROC curve, and the precision recall curve for the held out patient admissions. Applied to the validation data, our model results in an AUROC of 0.81 (95%CI 0.78-0.84) and an area under the precision recall curve (AUPR) of 0.04 (95%CI 0.03-0.05). Although the AUPR appears low, the performance is significantly better than a baseline classifier (Precision=0.01), since the incidence of infection is approximately 1% in the study population.



(a) Fraction of correctly identified cases we can predict at least x days in advance of the patient testing positive for CDI.



(b) Distribution of time to positive test from admission (in days)

Figure 3: Classification performance resulting from a classifier based on the 90^{th} percentile.

To evaluate the ability of our model to distinguish high-risk patients from low-risk patients in the held out set set of patient admissions, we selected a decision threshold based on the 90^{th} percentile. We chose this cutoff to limit the number false positives, since CDI cases are relatively rare. Given this decision threshold, we correctly identify 103 patients out of 242 as high risk, and achieve a sensitivity of 0.43, a positive predictive value of 0.04, an F-score 0.08, and an odds ratio of 6.67 (confusion matrix TP=103 TN=21,820 FN=139 FP=2337). Lowering the decision threshold will increase the sensitivity and increase the number of false positives. Ultimately, the choice of decision threshold depends on the expected costs and benefits of the intervention that one intends to apply to high-risk patients.

Figure 3(a) illustrates how far in advance we can predict positive test results. We note that in approximately half of the cases correctly identified, we identify cases at least 7 days in advance of their testing positive for CDI. Figure 3(b) illustrates *when* patients are testing positive. While we identify more patients who test positive earlier, the *fraction* of patients we correctly identify increases as the length of stay increases.

Finally, in Figure 4, we illustrate the performance of our model (the *Multi-Task Joint* approach) compared to that of a model learned by simply pooling all the data (the *Single-Task* approach), in terms of the AUROC. In addition, we consider a third approach, *Multi-Task Independent*, that also learns multiple models, one for each day, but where the optimization is performed independently for each model. Applied to all patients in the held out set of patient admissions (risk period>3 days), the proposed approach, *Multi-Task Joint*, and the *Single-Task* classifier perform almost identically, with both achieving an AUROC of 0.81. In contrast, the *Multi-Task Independent* performs worse, achieving an AUROC of 0.80. We believe this reduction is due to the inability of the method to leverage all of the data as in the *Multi-Task Joint* approach. When we divide the patients into subsets based on their risk period the difference between the *Single-Task* and *Multi-Task* approaches be-

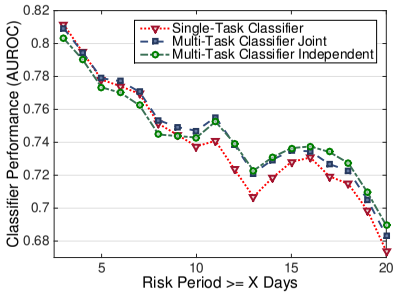


Figure 4: We compare a single-task classifier, where the model is time-invariant, to a multitask classifier where the model varies over time. The difference between the two classifiers is apparent for those patients who test positive later during the admission. These are the patients we are more interested in.

comes apparent. For patients with a longer risk period, the *Multi-Task* approach results in an improvement in performance over the *Single-Task* approach. While the difference is small, it is consistent. This difference is relevant, since the potential to intervene in a timely manner is greater for patients who test positive later in a visit. Therefore, the ability to identify such cases accurately is of considerable practical importance.

6. Discussion and Conclusion

In summary, we presented a novel data-driven patient risk stratification model for CDI that utilizes the entire structured contents of the EHR. The main contributions of our work are a novel approach to learning time-varying parameters and an evaluation scheme that is representative of how the model will be applied in practice.

The model provides estimates of patient risk daily based on time-varying parameters. We propose a multitask learning framework to efficiently learn the time-varying model. By learning the models jointly, we leverage the inherent relatedness among the different tasks. We observed changes in the ranking of features over time, suggesting that the proposed method could be used to further investigate the temporal effects of risk factors over the course of a hospital admission. These findings may shed new light on the relationship between risk factors and time, and in turn improve our understanding of the disease.

Furthermore, we demonstrated a consistent improvement in the classification performance using our multitask approach over a single-task approach, particularly among patients with longer risk periods. While consistent, the difference was not significant; this may be due to the fact that the number of cases with longer risk periods is small. We have approximately 5,000 visits in the heldout set with a risk period greater than 10 days, and only a small fraction of those patients end up testing positive. Additionally, in our formulation, we consider the same decision threshold every day. However, use of a variable

decision threshold could lead to better results. Furthermore, our results showed a clear overall improvement when learning the models jointly versus independently. However, on patients with very long risk periods e.g., greater than 15 days, the independent classifiers appear to perform slightly better. This may be an indication that the averaging or smoothing effect of the joint optimization approach is perhaps too strong. This may be addressed by only penalizing differences between successive models. In future work, such models could be improved by considering additional temporal smoothness constraints.

We proposed a new evaluation scheme motivated by a clinical use case in which the model is used daily to evaluate patients in the hospital and a decision is made about whether or not to intervene. Prior to this work, such models have only been evaluated at a single point in time during a hospital visit (e.g., at the time of admission or n days before the index event). We argue that evaluating the model over the entire course of the admission is a more accurate approximation of how the model would be used and would perform in practice. We also evaluated our final model in terms of how many days in advance we could predict high-risk cases. In a clinical setting, how far in advance one can predict an infection is as relevant as traditional performance metrics like sensitivity and specificity. This approach to evaluation is independent of both the disease and our method of building a classifier. With the increasing interest in applying machine learning to clinical problems, ensuring that evaluation criteria are clinically relevant will be of great importance.

In addition, we were careful to split our data temporally, learning on data from one year and evaluating the model on data from the next year. Over time, hospital populations, physical layouts, clinical protocols, and staff can change. Furthermore, EHR systems can change both in terms of what is collected and the precise meanings of variables. Thus, when evaluating predictive models in medicine, it is important to ensure that all examples in the training set precede all examples in the set held out for validation. Failure to do this can produce misleading results since future changes may be captured by the training set. To this end, we expect the predictive performance of our model will deteriorate over time, if such changes are not readily incorporated. An important future direction of study is how such models transfer across time, and best practices for building models that incorporate not only changes but also take into consideration possible interventions.

The models learned in this work are based on hospital-specific data, and thus may not generalize to hospitals with very different patient populations or clinical protocols. However, we expect the *methods* to generalize beyond this specific hospital. Our data-driven approach to feature engineering can be applied in straightforward manner to the structured contents of any hospital database. Such a data-driven approach can readily incorporate hospital-specific features resulting in more accurate predictive models. Here, we considered only the structured contents of the electronic health record. Future models, however, could incorporate features extracted from clinical notes or even genomic data pertaining to the microbiome of patients.

We focused on developing predictive models for CDI. However, our contributions extend to building models for other types of healthcare-associated infections and other patient populations. Once incorporated into the hospital workflow, risk stratification models, like the one presented here have the potential to reduce the incidence of adverse patient outcomes. Widespread development and use of such data-driven models promise to enable cost-effective, targeted prevention strategies that will ultimately improve patient care.

References

A Aboukhalil, L Nielsen, M Saeed, R Mark, and G Clifford. Reducing false alarm rates for critical arrhythmias using the arterial blood pressure waveform. *Journal of Biomedical Informatics*, 41(3):442–451, 2008.

R Caruana. Algorithms and applications for multitask learning. In *International Conference on Machine Learning (ICML)*, pages 87–95, 1996.

R Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

CDC. Identifying healthcare-associated infections (HAI) for NHSN surveillance, January 2015.

CC Chia, I Rubinfeld, B Scirica, S McMillan, H Gurm, and Z Syed. Looking beyond historical patient outcomes to improve clinical models. *Science Translational Medicine*, 4(131):131ra49–131ra49, 2012.

H Daumé III. Frustratingly easy domain adaptation. In *Association for Computational Linguistics (ACL)*, volume 1785, page 1787, 2007.

C DeSantis, J Ma, and A Bryan, Land Jemal. Breast cancer statistics, 2013. *CA: a Cancer Journal for Clinicians*, 64(1):52–62, 2014.

E Dubberke, A Wertheimer, et al. Review of current literature on the economic burden of *Clostridium difficile* infection. *Infection Control and Hospital Epidemiology*, 30(1):57–66, 2009.

E Dubberke, Y Yan, K Reske, A Butler, J Doherty, V Pham, and V Fraser. Development and validation of a *Clostridium difficile* infection risk prediction model. *Infection Control and Hospital Epidemiology*, 32(4):360–366, 2011.

J Fan and W Zhang. Statistical methods with varying coefficient models. *Statistics and Its Interface*, 1(1):179, 2008.

RE Fan, KW Chang, CJ Hsieh, XR Wang, and CJ Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

K Garey, T Dao-Tran, Z Jiang, M Price, L Gentry, and H DuPont. A clinical risk index for *Clostridium difficile* infection in hospitalized patients receiving broad-spectrum antibiotics. *Journal of Hospital Infection*, 70(2):142–147, 2008.

RJ Gray. Flexible methods for analyzing survival data using splines, with applications to breast cancer prognosis. *Journal of the American Statistical Association*, 87(420): 942–951, 1992.

T Hastie and R Tibshirani. Varying-coefficient models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 757–796, 1993.

LI Iezzoni. Using administrative diagnostic data to assess the quality of hospital care: pitfalls and potential of ICD-9-CM. *International Journal of Technology Assessment in Health Care*, 6(02):272–281, 1990.

D Kansagara, H Englander, A Salanitro, D Kagen, C Theobald, M Freeman, and S Kripalani. Risk prediction models for hospital readmission: a systematic review. *Journal of the American Medical Association*, 306(15):1688–1698, 2011.

S Kleinberg and G Hripcsak. A review of causal inference for biomedical informatics. *Journal of Biomedical Informatics*, 44(6):1102–1112, 2011.

RM Klevens, J Edwards, C Richards, T Horan, R Gaynes, D Pollock, and DM Cardo. Estimating health care-associated infections and deaths in us hospitals, 2002. *Public Health Reports*, 122(2):160, 2007.

G Krapohl. Preventing health care-associated infection: Development of a clinical prediction rule for *Clostridium difficile* infection. PhD Thesis, 2011.

SS Magill, JR Edwards, W Bamberg, ZG Beldavs, G Dumyati, MA Kainer, R Lynfield, M Maloney, L McAllister-Hollod, J Nadle, et al. Multistate point-prevalence survey of health care–associated infections. *New England Journal of Medicine*, 370(13):1198–1208, 2014.

B Miller, L Chen, Dl Sexton, and D Anderson. Comparison of the burdens of hospital-onset, healthcare facility–associated *Clostridium difficile* infection and of healthcare-associated infection due to methicillin-resistant staphylococcus aureus in community hospitals. *Infection Control and Hospital Epidemiology*, 32(4):387–390, 2011.

SA Murphy and PK Sen. Time-dependent coefficients in a Cox-type regression model. *Stochastic Processes and their Applications*, 39(1):153–180, 1991.

NCHS. International classification of diseases, 9th revision, clinical modification. National Center for Health Statistics. http://icd9cm.chrisendres.com, 2008.

C Paxton, A Niculescu-Mizil, and S Saria. Developing predictive models using electronic medical records: Challenges and pitfalls. In *Americal Medical Informatics Association (AMIA) Symposium*, 2013.

J Pépin, ME Alary, L Valiquette, E Raiche, J Ruel, K Fulop, D Godin, and C Bourassa. Increasing risk of relapse after treatment of *Clostridium difficile* colitis in Quebec, Canada. *Clinical Infectious Diseases*, 40(11):1591–1597, 2005.

M Saeed, M Villarroel, A Reisner, G Clifford, LW Lehman, G Moody, Ts Heldt, T Kyaw, B Moody, and RG Mark. Multiparameter intelligent monitoring in intensive care ii (MIMIC-II): a public-access intensive care unit database. *Critical Care Medicine*, 39(5): 952, 2011.

S Saria, A Rajani, J Gould, D Koller, and A Penn. Integration of early physiological responses predicts later illness severity in preterm infants. *Science Translational Medicine*, 2(48):48ra65–48ra65, 2010.

A Shoeb and JV Guttag. Application of machine learning to epileptic seizure detection. In *International Conference on Machine Learning (ICML)*, pages 975–982, 2010.

D Sievert, P Ricks, J Edwards, A Schneider, J Patel, A Srinivasan, A Kallen, B Limbago, and S Fridkin. Antimicrobial-resistant pathogens associated with healthcare-associated infections: summary of data reported to the national healthcare safety network at the centers for disease control and prevention, 2009–2010. *Infection Control and Hospital Epidemiology*, 34(1):1–14, 2013.

Y Sun, R Sundaram, and Y Zhao. Empirical likelihood inference for the Cox model with time-dependent coefficients via local partial likelihood. *Scandinavian Journal of Statistics*, 36(3):444–462, 2009.

Z Syed and I Rubinfeld. Unsupervised risk stratification in clinical datasets: Identifying patients at risk of rare outcomes. In *International Conference on Machine Learning (ICML)*, pages 1023–1030, 2010.

Z Syed, C Stultz, B Scirica, and J Guttag. Computationally generated cardiac biomarkers for risk stratification after acute coronary syndrome. *Science Translational Medicine*, 3 (102):102ra95–102ra95, 2011.

J Tanner, D Khan, D Anthony, and J Paton. Waterlow score to predict patients at risk of developing *Clostridium difficile*-associated disease. *Journal of Hospital Infection*, 71(3): 239–244, 2009.

L Tian, D Zucker, and LJ Wei. On the Cox model with time-varying regression coefficients. *Journal of the American Statistical Association*, 100(469):172–183, 2005.

C Umscheid, Al Rajender, W Kendal, P Brennan, et al. Estimating the proportion of healthcare-associated infections that are reasonably preventable and the related mortality and costs. *Infection Control and Hospital Epidemiology*, 32(2):101–114, 2011.

J Wiens, J Guttag, and E Horvitz. Patient risk stratification for hospital-associated c. diff as a time-series classification task. In *Neural Information Processing Systems (NIPS)*, 2012a.

J Wiens, E Horvitz, and J Guttag. Learning evolving patient risk processes for *C. diff* colonization. In *ICML Workshop on Machine Learning from Clinical Data*, 2012b.

J Wiens, WN Campbell, ES Franklin, JV Guttag, and E Horvitz. Learning data-driven patient risk stratification models for clostridium difficile. In *Open Forum Infectious Diseases*, volume 1, page ofu045. Oxford University Press, 2014.

DS Yokoe, LA Mermel, DJ Anderson, KM Arias, H Burstin, DP Calfee, SE Coffin, ER Dubberke, V Fraser, DN Gerding, et al. Executive summary: a compendium of strategies to prevent healthcare-associated infections in acute care hospitals. *Infection Control*, 29 (S1):S12–S21, 2008.

J Zhou, L Yuan, J Liu, and J Ye. A multi-task learning formulation for predicting disease progression. In *Conference on Knowledge Discovery and Data Mining (SigKDD)*, pages 814–822. ACM, 2011.

Jiayu Zhou, Fei Wang, Jianying Hu, and Jieping Ye. From micro to macro: data driven phenotyping by densification of longitudinal electronic medical records. In *Conference on Knowledge Discovery and Data Mining (KDD)*, pages 135–144. ACM, 2014.

D Zucker and AF Karr. Nonparametric survival analysis with time-dependent covariate effects: a penalized partial likelihood approach. *Annals of Statistics*, pages 329–353, 1990.

Journal of Machine Learning Research 17 (2016) 1-33 Submitted 5/15; Revised 11/15; Published 4/16

Multiplicative Multitask Feature Learning

Xin Wang WANGXIN@ENGR.UCONN.EDU
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06279, USA

Jinbo Bi JINBO@ENGR.UCONN.EDU
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06279, USA

Shipeng Yu SHIPENG.YU@SIEMENS.COM
Health Services Innovation Center
Siemens Healthcare
Malvern, PA 19355, USA

Jiangwen Sun JAVON@ENGR.UCONN.EDU
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06279, USA

Minghu Song MINGHU.SONG@PFIZER.COM
Worldwide Research and Development
Pfizer Inc.
Groton, CT 06340, USA

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

We investigate a general framework of multiplicative multitask feature learning which decomposes individual task's model parameters into a multiplication of two components. One of the components is used across all tasks and the other component is task-specific. Several previous methods can be proved to be special cases of our framework. We study the theoretical properties of this framework when different regularization conditions are applied to the two decomposed components. We prove that this framework is mathematically equivalent to the widely used multitask feature learning methods that are based on a joint regularization of all model parameters, but with a more general form of regularizers. Further, an analytical formula is derived for the across-task component as related to the task-specific component for all these regularizers, leading to a better understanding of the shrinkage effects of different regularizers. Study of this framework motivates new multitask learning algorithms. We propose two new learning formulations by varying the parameters in the proposed framework. An efficient blockwise coordinate descent algorithm is developed suitable for solving the entire family of formulations with rigorous convergence analysis. Simulation studies have identified the statistical properties of data that would be in favor of the new formulations. Extensive empirical studies on various classification and regression benchmark data sets have revealed the relative advantages of the two new formulations by comparing with the state of the art, which provides instructive insights into the feature learning problem with multiple tasks.

Keywords: Multitask learning, regularization, sparse modeling, blockwise coordinate descent

1. Introduction

Multitask learning (MTL) improves the generalization of the estimated models for multiple related learning tasks by capturing and exploiting the task relationships. It has been theoretically and empirically shown to be more effective than learning tasks individually. Especially when single task learning suffers from limited sample size, multitask learning reinforces a single learning process with the transferable knowledge learned from the related tasks. Multi-task learning has been widely applied in many scientific fields, such as robotics (Zhang and Yeung, 2010), natural language processing (Ando and Zhang, 2005), computer aided diagnosis (Bi et al., 2008), and computer vision (Kang et al., 2011).

Research efforts have been devoted to various MultiTask Feature Learning (MTFL) algorithms. One direction of these works directly learns the dependencies among tasks, either by modeling the correlated regression or classification noise (Greene, 2002), or assuming that the model parameters share a common prior (Yu et al., 2005; Lee et al., 2007; Xue et al., 2007; Yu et al., 2007; Jacob et al., 2008), or by examining the tasks' covariance matrix (Bonilla et al., 2007; Zhang and Yeung, 2010; Guo et al., 2011; Yang et al., 2013). Another research direction relies on a basic assumption that the different tasks may share a substructure in the feature space. In order to exploit this shared substructure, Rai and Daume (2010) project the task parameters to explore the latent common substructure. Kang et al. (2011) form a shared low-dimensional representation of data by feature learning. More recent methods explore the latent basis that can be used to characterize the entire set of tasks and examine the potential clusters of tasks. For instance, Passos et al. (2012) assume that subsets of features may be shared only between tasks in the same cluster. Kumar and Daume III (2012) allow overlapping of tasks in different groups by having several bases in common. Maurer et al. (2013) exploit a dictionary allowing sparse representations of the tasks. Gong et al. (2012) detect if certain tasks are outliers from the majority of the tasks.

Regularization methods are widely used in MTFL to learn a shared subset of features. A common strategy is to impose a blockwise joint regularization (Meier et al., 2008; Zhao et al., 2009) on all task model parameters to shrink the effects of features across the tasks and simultaneously minimize the regression or classification loss. These methods employ the so-called $\ell_{1,p}$ matrix norm (Lee et al., 2010; Liu et al., 2009a; Obozinski and Taskar, 2006; Zhang et al., 2010; Zhou et al., 2010) that is the sum of the ℓ_p norms of the rows in a matrix. As a result, this regularizer encourages sparsity among the rows. If a row of the parameter matrix corresponds to a feature and a column represents an individual task, the $\ell_{1,p}$ regularizer intends to rule out the unrelated features across tasks by shrinking the entire rows of the matrix to zero. Typical choices for p are 2 (Obozinski and Taskar, 2006; Evgeniou and Pontil, 2004) and ∞ (Turlach et al., 2005). Effective algorithms have since then been developed for the $\ell_{1,2}$ (Liu et al., 2009a) and $\ell_{1,\infty}$ (Quattoni et al., 2009) regularization. Later, the $\ell_{1,p}$ norm is generalized to include $1 < p \leq \infty$ with a probabilistic interpretation that the resultant MTFL method solves a relaxed optimization problem with a generalized

normal prior for all tasks (Zhang et al., 2010). Although the matrix-norm based regularizers lead to convex learning formulations for MTFL, recent studies show that a convex regularizer may be too relaxed to approximate the ℓ_0-type regularizer for the shrinkage effects in the feature space and thus results in suboptimal performance (Gong et al., 2013). To address this problem, non-convex regularizers, such as the capped-ℓ_1, ℓ_1 regularizer (Gong et al., 2013), have been proposed for the multitask joint regularization. However, using non-convex regularizers may bring up computational challenges. For instance, non-convex formulations are usually difficult to solve, and require more complicated optimization algorithms to guarantee satisfactory performance.

For the existing MTFL methods based on joint regularization, a major limitation is that it either selects a feature as relevant to all tasks or excludes it from all models, which is very restrictive in practice where tasks may share some features but may also have their own specific features that are not relevant to other tasks. To overcome this limitation, one of the most effective strategies is to decompose the model parameters into either summation (Jalali et al., 2010; Chen et al., 2011; Gong et al., 2012) or multiplication (Xiong et al., 2006; Bi et al., 2008; Lozano and Swirszcz, 2012) of two components with separate regularizers applied to the two components. One regularizer is imposed on the component taking care of the task-specific model parameters and the other one is imposed on the component for mining the cross-feature sparsity. Specifically, for the methods that decompose the parameter matrix into summation of two matrices, the dirty model in (Jalali et al., 2010) employs $\ell_{1,1}$ and $\ell_{1,\infty}$ regularizers to the two components. A robust MTFL method in (Chen et al., 2011) uses the trace norm on one component for mining a low-rank structure shared by tasks and a column-wise $\ell_{1,2}$-norm on the other component for identifying task outliers. A more recent method applies the $\ell_{1,2}$-norm both row-wisely to one component and column-wisely to the other (Gong et al., 2012). For these additive decomposition methods, it requires the corresponding entries in both components to be zero in order to exclude a feature from a task.

For the methods that work with multiplicative decompositions, the parameter vector of each task is decomposed into an element-wise product of two vectors where one is used across tasks and the other is task-specific. To exclude a feature from a task, the multiplicative decomposition only requires one of the components to be zero. Existing methods of this line apply the same regularization to both of the component vectors, by either the ℓ_2-norm penalty (Bi et al., 2008), or the sparse ℓ_1-norm (i.e., multi-level LASSO (Lozano and Swirszcz, 2012)). The multi-level LASSO method has been analytically compared to the dirty model (Lozano and Swirszcz, 2012), showing that the multiplicative decomposition creates better shrinkage on the global and task-specific parameters. The across-task component can screen out the features irrelevant to all tasks. An individual task's component can further select features from those selected by the cross-task component for use in its corresponding model. Although there are different ways to regularize the two components in the product, no systematic work has been done to analyze the algorithmic and statistical properties of the different regularizers. It is insightful to answer the questions such as how these learning formulations differ from the early methods based on blockwise joint regularization, how the optimal solutions of the two components intervene, and how the resultant solutions are compared with those of other methods that also learn both shared and task-specific features. We highlight the contributions of this paper as follows:

- We propose and examine a general framework of the multiplicative decomposition that enables a variety of regularizers to be applied. The general form corresponds to a family of MTFL methods, including all early methods that decompose model parameters as a product of two components (Bi et al., 2008; Lozano and Swirszcz, 2012).

- Our theoretical analysis has revealed that this family of methods is actually equivalent to the joint regularization based approach but with a more general form of regularizers, including matrix-norm based and non-matrix-norm based regularizers. The non-matrix-norm based joint regularizers derived from the proposed framework have never been considered previously. If they are considered in the joint regularization form, the resultant optimization problems will be difficult to solve. However, our equivalent multiplicative MTFL framework in this case uses convex regularizers on the two components, which can be solved efficiently.

- Further analysis reveals that the optimal solution of the across-task component can be analytically computed by a formula of the optimal task-specific parameters. This analytical result facilitates a better understanding of the shrinkage effects of the different regularizers applied to the two components.

- Statistical justification is also derived for this family of formulations. It proves that the proposed framework is equivalent to maximizing a lower bound of the *maximum a posterior* solution under a probabilistic model that assumes generalized normal priors on model parameters.

- Two new MTFL formulations are derived from the proposed general framework. Unlike the existing methods (Bi et al., 2008; Lozano and Swirszcz, 2012) where the same kind of vector norm is applied to both components, the shrinkage of the global and task-specific parameters differs in the new formulations. We empirically illustrate the scenarios where the two new formulations are more suitable for solving the MTFL problems.

- An efficient blockwise coordinate descent algorithm is derived suitable for solving the entire family of the methods. Given some of the methods (including the two new formulations we study) correspond to non-matrix-norm based joint regularizers, our algorithm provides a powerful alternative to solving the related difficult optimization problems, allowing us to explore the behaviors and properties of these regularizers in an effective way. Convergence analysis is thoroughly discussed.

To depict the differences between our approach and previous methods, Table 1 summaries various regularizers used in the joint regularization based and model decomposition based MTFL methods. There are some fundamental connections among these methods. As studied in the present work, multiplicative MTFL models can be connected with the early blockwise joint regularization models. We also attempt to empirically compare the model behaviors between the multiplicative and additive MTFL methods, and particularly compare the applicability of the four different choices of regularization listed in Table 1 for multiplicative MTFL .

Table 1: The regularization terms used in various MTFL methods.

	Model	Methods	Norms
Joint regularization models	$\mathbf{A}$	Evgeniou and Pontil (2004)	$\ell_{1,2}$
		Turlach et al. (2005)	$\ell_{1,\infty}$
		Lee et al. (2010)	both $\ell_{1,1}$ and $\ell_{1,2}$
		Zhang et al. (2010)	$\ell_{1,p}, 1 < p \leq \infty$
		Gong et al. (2012)	capped $\ell_{1,1}$
Decomposed models	$\mathbf{A} = \mathbf{P} + \mathbf{Q}$	Jalali et al. (2010)	$\ell_{1,1}$ on $\mathbf{P}$, $\ell_{1,\infty}$ on $\mathbf{Q}$
		Gong et al. (2012)	$\ell_{1,2}$ on $\mathbf{P}$, $\ell_{1,2}$ on $\mathbf{Q}^T$
	$\mathbf{A} = \text{diag}(\mathbf{c}) \cdot \mathbf{B}$	**The proposed framework**	$(\ell_k)^k$ on $\mathbf{c}$, $(\ell_p)^p$ on $\mathbf{B}$
		Bi et al. (2008)	k=2, p=2
		Lozano and Swirszcz (2012)	k=1, p=1
		The new formulation 1	k=1, p=2
		The new formulation 2	k=2, p=1

The rest of the paper is organized as follows. Section 2 defines the mathematical notation and introduces the proposed models in detail. Section 3 discusses several important theoretical properties of the proposed models including equivalence analysis. Section 4 provides the statistical justification of the multiplicative MTFL models. In Section 5, we develop an efficient algorithm to solve the optimization problems in the proposed models with a convergence analysis. Section 6 shows the empirical results, in which simulations have been designed to examine the various feature sharing patterns for which a specific choice of regularizer may be preferred. Extensive experiments with a variety of classification and regression benchmark data sets are also described in Section 6. In Section 7, we conclude this work.

2. The Proposed Multiplicative MTFL

Given T tasks in total, for each task t, $t \in \{1, \cdots, T\}$, we have sample set ($\mathbf{X}_t \in \mathbb{R}^{\ell_t \times d}, \mathbf{y}_t \in \mathbb{R}^{\ell_t}$). The data set of $\mathbf{X}_t$ has ℓ_t examples, where the i-th row corresponds to the i-th example $\mathbf{x}_i^t$ of task t, $i \in \{1, \cdots, \ell_t\}$, and each column represents a feature and there are totally d features. The vector $\mathbf{y}_t$ contains y_i^t, the label of the i-th example of task t. We consider functions of the linear form $y_t = \boldsymbol{\alpha}_t^\top \mathbf{x}_i^t$ where $\boldsymbol{\alpha}_t \in \mathbb{R}^d$, which corresponds to computing $\mathbf{X}_t\boldsymbol{\alpha}_t$ on the training data as the estimate of $\mathbf{y}_t$. We define the parameter matrix or weight matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T]$ and denote the rows of $\mathbf{A}$ by $\boldsymbol{\alpha}^j$, $j \in \{1, \cdots, d\}$.

The joint regularization based MTFL method with the $\ell_{1,p}$ regularizer minimizes

$$\sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} ||\boldsymbol{\alpha}^j||_p, \tag{1}$$

for the best $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, where $L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the loss function of task t which computes the discrepancy between the observed $\mathbf{y}_t$ and the model output $\mathbf{X}_t\boldsymbol{\alpha}_t$, and λ is a tuning parameter to balance between the loss and the regularizer. Although any suitable loss function can be used in the formulation (1), convex loss functions are the common choices such as the least squares loss $\sum_{i=1}^{\ell_t}(y_i^t - \boldsymbol{\alpha}_t^\top \mathbf{x}_i^t)^2$ for regression problems or the logistic loss

$\sum_{i=1}^{\ell_t} \log(1 + e^{-y_i^t(\boldsymbol{\alpha}_t^\top \mathbf{x}_i^t)})$ for classification problems. These loss functions are strictly convex with respect to the model parameters $\boldsymbol{\alpha}_t$. The ℓ_p norm is computed for each row of the matrix $\mathbf{A}$ corresponding to a feature (rather than a task) so to enforce sparsity on the features.

A family of multiplicative MTFL methods can be derived by rewriting $\boldsymbol{\alpha}_t = \text{diag}(\mathbf{c})\boldsymbol{\beta}_t$ where $\text{diag}(\mathbf{c})$ is a diagonal matrix with its diagonal elements composing a vector $\mathbf{c}$. The $\mathbf{c}$ vector is used across all tasks, indicating if a feature is useful for any of the tasks, and the vector $\boldsymbol{\beta}_t$ is only for task t. Let j index the entries in these vectors. We have $\alpha_j^t = c_j\beta_j^t$. Typically $\mathbf{c}$ comprises binary entries that are equal to 0 or 1, but the integer constraint is often relaxed to require just non-negativity ($\mathbf{c} \geq 0$). We minimize a regularized loss function as follows for the best $\mathbf{c}$ and $\boldsymbol{\beta}_{t:t=1,\cdots,T}$:

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_p^p + \gamma_2 ||\mathbf{c}||_k^k, \tag{2}$$

where $L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the same loss function used in Eq.(1) but with $\boldsymbol{\alpha}_t$ replaced by the new vector of $(c_j\beta_j^t)_{j=1,\cdots,d}$, $||\boldsymbol{\beta}_t||_p^p = \sum_{j=1}^{d} |\beta_j^t|^p$ and $||\mathbf{c}||_k^k = \sum_{j=1}^{d}(c_j)^k$, which are the ℓ_p-norm of $\boldsymbol{\beta}_t$ to the power of p and the ℓ_k-norm of $\mathbf{c}$ to the power of k if p and k are positive integers. The tuning parameters γ_1, γ_2 are used to balance the empirical loss and regularizers. At optimality, if $c_j = 0$, the j-th variable is removed for all tasks, and the corresponding row vector $\boldsymbol{\alpha}^j = \mathbf{0}$; otherwise the j-th variable is selected for use in at least one of the $\boldsymbol{\alpha}$'s. Then, a specific $\boldsymbol{\beta}_t$ can rule out the j-th variable from task t if $\beta_j^t = 0$.

If both $p = k = 2$, Problem (2) becomes the formulation used in Bi et al. (2008). Since the ℓ_2-norm regularization is applied on both $\boldsymbol{\beta}_t$ and $\mathbf{c}$, this model does not impose strong sparsity on the model parameters. According to our empirical study, this model may be suitable for the scenarios where only a few features can be excluded from all of the tasks, and the different models (tasks) share a lot features between each other. There could exist features that, although irrelevant to most of the tasks, cannot be completely excluded only due to few tasks.

If $p = k = 1$, Problem (2) becomes the formulation used in Lozano and Swirszcz (2012), where the ℓ_1-norm regularization is applied on $\boldsymbol{\beta}_t$ and $\mathbf{c}$ and thus it induces very strong sparsity both on task-specific parameters and the across-task component to select the features. Compared to the model in Bi et al. (2008), this model is more suitable for learning from the tasks with persistently sparse models. For example, many features are irrelevant to any of the tasks, and only a few of the features selected by $\mathbf{c}$ are used by an individual task, indicating the sparse pattern in sharing the selected features among tasks.

Besides the above two existing models, any other choices of p and k will derive into new formulations for MTFL. Note that the two existing methods discussed in Bi et al. (2008); Lozano and Swirszcz (2012) use $p = k$ in their formulations, which renders β_j^t and c_j the same amount of shrinkage. To explore other feature sharing patterns among tasks, we propose two new formulations where $p \neq k$.

Formulation 1:
If $p = 2$ but $k = 1$ in Problem (2), then we obtain the following optimization problem

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_2^2 + \gamma_2 ||\mathbf{c}||_1. \tag{3}$$

When there exists a large subset of features irrelevant to any of the tasks, it requires a sparsity-inducing norm on $\mathbf{c}$. However, within the relevant features selected by $\mathbf{c}$, the majority of these features are shared between tasks. In other words, the features used in each task are not sparse relative to the features selected by $\mathbf{c}$, which requires a non-sparsity-inducing norm on $\boldsymbol{\beta}$. Hence, we use ℓ_1 norm on $\mathbf{c}$ and ℓ_2 norm on each $\boldsymbol{\beta}$ in our formulation (2).

Formulation 2:
If $p = 1$ but $k = 2$ in Problem (2), we obtain the following optimization problem

$$\min_{\boldsymbol{\beta}_t, \mathbf{c} \geq 0} \sum_{t=1}^{T} L(\mathbf{c}, \boldsymbol{\beta}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{t=1}^{T} ||\boldsymbol{\beta}_t||_1 + \gamma_2 ||\mathbf{c}||_2^2. \tag{4}$$

When the union of the features relevant to any given tasks includes many or even all features, the ℓ_2 norm penalty on $\mathbf{c}$ may be preferred. However, only a limited number of features are shared between tasks, i.e., the features used by individual tasks are sparse with respect to the features selected as useful across tasks by $\mathbf{c}$. We can impose the ℓ_1 norm penalty on $\boldsymbol{\beta}$.

Clearly, many other choices of p and k values can be used, such as those corresponding to higher order polynomials (e.g., $\sum_{j=1}^{d} c_j^3$). Our theoretical results in the next few sections apply to all positive value choices of p and k unless otherwise specified. In our empirical studies, however, we have implemented algorithms for the two existing models and the two new models for comparison. Some other choices of regularizers may require significant re-programming of our algorithms and we will leave them for more thorough individual examinations in the future.

3. Theoretical Analysis

We first extend the formulation (1) to allow more choices of regularizers. We introduce a new notation that is an operator applied to a vector, such as $\boldsymbol{\alpha}^j$. The operator $||\boldsymbol{\alpha}^j||_p^{p/q} = \sqrt[q]{\sum_{t=1}^{T} |\alpha_j^t|^p}$, $p, q \geq 0$, which corresponds to the ℓ_p norm if $p = q$ and both are positive integers. A joint regularized MTFL approach can solve the following optimization problem with pre-specified values of p, q and λ, for the best parameters $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$:

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}. \tag{5}$$

Our main results of this paper include (i) a theorem that establishes the equivalence between the models derived from solving Problem (2) and Problem (5) for properly chosen values of λ, q, k, γ_1 and γ_2; (ii) a theorem that delineates the conditions for (2) to impose a convex (or concave) regularizer on the model parameter matrix $\mathbf{A}$; and (iii) an analytical solution of Problem (2) for $\mathbf{c}$ which shows how the sparsity of the across-task component is relative to the sparsity of task-specific components.

Theorem 1 (Main Result 1) *Let $\hat{\boldsymbol{\alpha}}_t$ be the optimal solution to Problem (5) and ($\hat{\boldsymbol{\beta}}_t$, $\hat{\mathbf{c}}$) be the optimal solution to Problem (2). Then $\hat{\boldsymbol{\alpha}}_t = diag(\hat{\mathbf{c}})\hat{\boldsymbol{\beta}}_t$ when $\lambda = 2\sqrt{\gamma_1^{2-\frac{p}{kq}} \gamma_2^{\frac{p}{kq}}}$ and $q = \frac{k+p}{2k}$ (or equivalently, $k = \frac{p}{2q-1}$).*

Proof Theorem 1 can be proved by establishing the following two lemmas and two theorems. The two lemmas provide the basis for the proofs of the two theorems and then from the first theorem, we conclude that the solution $\hat{\boldsymbol{\alpha}}_t$ of Problem (5) also minimizes the following optimization problem:

$$\min_{\boldsymbol{\alpha}_t, \boldsymbol{\sigma} \geq 0} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \mu_1 \sum_{j=1}^{d} \sigma_j^{-1} ||\boldsymbol{\alpha}^j||_p^{p/q} + \mu_2 \sum_{j=1}^{d} \sigma_j, \tag{6}$$

and the optimal solution of Problem (6) also minimizes Problem (5) when proper values of λ, μ_1 and μ_2 are chosen. The second theorem connects Problem (6) to the proposed formulation (2). We show that the optimal $\hat{\sigma}_j$ is equal to $(\hat{c}_j)^k$, and then the optimal $\hat{\boldsymbol{\alpha}}$ can be computed as $\text{diag}(\hat{\mathbf{c}})\hat{\boldsymbol{\beta}}_t$ from the optimal $\hat{\boldsymbol{\beta}}_t$. ■

Note that when $p = 2$ and $q = 1$, the intermediate problem (6) uses a similar regularizer to that in Micchelli et al. (2013) where $\frac{|\alpha^j|^2}{\sigma_j} + \sigma_j$ is used to approximate $|\alpha^j|$ in the ℓ_1-norm regularizer. Problem (6) extends the discussion in Micchelli et al. (2013) to include more general regularizers according to p and q.

Lemma 2 *For any given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, Problem (6) can be optimized with respect to $\boldsymbol{\sigma}$ by the following analytical solution*

$$\sigma_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}. \tag{7}$$

Proof By the Cauchy-Schwarz inequality, we can derive a lower bound for the sum of the two regularizers in Problem (6) as follows:

$$\mu_1 \sum_{j=1}^{d} \sigma_j^{-1} ||\boldsymbol{\alpha}^j||_p^{p/q} + \mu_2 \sum_{j=1}^{d} \sigma_j \geq 2\sqrt{\mu_1 \mu_2} \sum_{j=1}^{d} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}, \tag{8}$$

where the equality holds if and only if $\sigma_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}$.

Using the method of proof by contradiction, suppose that $\boldsymbol{\sigma}^*$ optimizes Problem (6) with a given set of $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, but $\sigma_j^* \neq \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}$. Thus, $\boldsymbol{\sigma}^*$ does not make the regularization term reach its lower bound. Then, we can choose another $\tilde{\boldsymbol{\sigma}}$ where $\tilde{\sigma}_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}}$, so $\tilde{\boldsymbol{\sigma}}$ delivers the lower bound in Eq.(8). Because the lower bound (the right hand side of Eq.(8)) only depends on $\boldsymbol{\alpha}$, it is a constant for fixed $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$. Hence, since the loss function is also fixed for given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, $\tilde{\boldsymbol{\sigma}}$ reaches a lower objective value of Problem (6) than that of $\boldsymbol{\sigma}^*$, which contradicts to the optimality of $\boldsymbol{\sigma}^*$. Therefore, the optimal $\boldsymbol{\sigma}$ always takes the form of Eq.(7). ■

Remark 3 *Based on the proof of Lemma 2, we also know that the objective function of (5) is the lower bound of the objective function of (6) for any given $\boldsymbol{\alpha}_t$ (including the optimal $\hat{\boldsymbol{\alpha}}_t$) when $\lambda = 2\sqrt{\mu_1 \mu_2}$, and the lower bound can be attained if and only if $\boldsymbol{\sigma}$ is set according to the formula (7). Hence, we can also conclude that if $(\hat{\boldsymbol{\alpha}}_t, \hat{\boldsymbol{\sigma}})$ is the optimal solution of Problem (6), then $\hat{\sigma}_j = \mu_1^{\frac{1}{2}} \mu_2^{-\frac{1}{2}} \sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$.*

Lemma 4 *Let $\alpha_j^t = c_j\beta_j^t$ for all t and j. Replacing $\boldsymbol{\beta}_t$ by $\boldsymbol{\alpha}_t$ in Problem (2) yields the following optimization problem*

$$\min_{\boldsymbol{\alpha}_t, \mathbf{c}\geq 0} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{j=1}^{d} c_j^{-p}||\boldsymbol{\alpha}^j||_p^p + \gamma_2 \sum_{j=1}^{d}(c_j)^k. \tag{9}$$

For any given $\boldsymbol{\alpha}_{t:t=1,\cdots,T}$, Problem (9) can be optimized with respect to $\mathbf{c}$ by the following analytical solution

$$c_j = (\gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\alpha_j^t)^p)^{\frac{1}{p+k}}. \tag{10}$$

Proof This lemma can be proved following a similar argument in the proof of Lemma 2. By the Cauchy-Schwarz inequality, the sum of the two regularizers in (9) satisfies the following inequality

$$\gamma_1 \sum_{j=1}^{d} c_j^{-p}||\boldsymbol{\alpha}^j||_p^p + \gamma_2 \sum_{j=1}^{d} c_j^k \geq 2\gamma_1^{\frac{k}{p+k}}\gamma_2^{\frac{p}{p+k}} \sum_{j=1}^{d}(||\boldsymbol{\alpha}^j||_p^p)^{\frac{k}{p+k}}, \tag{11}$$

and the equality holds if and only if $\gamma_1 c_j^{-p}||\boldsymbol{\alpha}^j||_p^p = \gamma_2 c_j^k$ (and note that all parameters γ_1, γ_2, $||\hat{\boldsymbol{\alpha}}^j||_p^p$ and $\mathbf{c}$ are non-negative), which yields the following formula

$$c_j = (\gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\alpha_j^t)^p)^{\frac{1}{p+k}}.$$

Through proof by contradiction, we know that the optimal $\mathbf{c}$ has to take the above formula. ■

Based on Lemma 2, we will prove that Problem (5) is equivalent to Problem (6) in the sense that an optimal solution of Problem (5) is also an optimal solution of Problem (6) and vice versa when λ, μ_1, and μ_2 satisfy certain conditions.

Theorem 5 *The solution sets of Problem (5) and Problem (6) are identical when $\lambda = 2\sqrt{\mu_1\mu_2}$.*

Proof First, if $\hat{\mathbf{A}} = [\hat{\boldsymbol{\alpha}}_{t:t=1,\cdots,T}]$ minimizes Problem (5), we show that the pair $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ minimizes Problem (6) where $\hat{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$.

By Remark 3, the objective function of (5) is the lower bound of the objective function of (6) for any given $\mathbf{A}$ (including the optimal solution $\hat{\mathbf{A}}$). When $\hat{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$, Problem (6) reaches the lower bound. In this case, Problem (5) at $\hat{\mathbf{A}}$ and Problem (6) at $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ have the same objective value. Now, suppose that the pair $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ does not minimize Problem (6), there exists another pair $(\tilde{\mathbf{A}}, \tilde{\boldsymbol{\sigma}}) \neq (\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ that achieves a lower objective value than that of $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$. By Lemma 2, we have that $\tilde{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\tilde{\boldsymbol{\alpha}}^j||_p^{p/q}}$, and then Problem (6), at $\tilde{\mathbf{A}}$, reaches the lower bound which is formulated as the objective of Problem (5)

when $\lambda = 2\sqrt{\mu_1\mu_2}$. In other words, the objective values of (6) and (5) are identical at $\tilde{\mathbf{A}}$. Hence, $\tilde{\mathbf{A}}$ achieves a lower objective value than that of $\hat{\mathbf{A}}$ for Problem (5), contradicting to the optimality of $\hat{\mathbf{A}}$.

Second, if $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ minimizes Problem (6), we show that $\hat{\mathbf{A}}$ minimizes Problem (5).

Suppose that $\hat{\mathbf{A}}$ does not minimize Problem (5), which means that there exists $\tilde{\boldsymbol{\alpha}}^j$ ($\neq \hat{\boldsymbol{\alpha}}^j$ for some j) that achieves a lower objective value than that of $\hat{\boldsymbol{\alpha}}^j$. We set $\tilde{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\tilde{\boldsymbol{\alpha}}^j||_p^{p/q}}$. Then $(\hat{\mathbf{A}}, \tilde{\boldsymbol{\sigma}})$ is an optimal solution of Problem (6) as proved in the first paragraph, and will bring the objective function of Problem (6) to a lower value than that of $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$, contradicting to the optimality of $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$.

Therefore, Problems (5) and (6) have identical solution sets when $\lambda = 2\sqrt{\mu_1\mu_2}$. ■

In order to link Problem (6) to our formulation (2), we let $\sigma_j = (c_j)^k$, $k \in \mathbb{R}$, $k \neq 0$ and $\alpha_j^t = c_j\beta_j^t$ for all t and j, and derive an equivalent objective function of Problem (6) based on Lemmas 2 and 4.

Theorem 6 *The optimal solution $(\hat{\mathbf{A}}, \hat{\boldsymbol{\sigma}})$ of Problem (6) is equivalent to the optimal solution $(\hat{\mathbf{B}}, \hat{\mathbf{c}})$ of Problem (2) where $\hat{\alpha}_j^t = \hat{c}_j\hat{\beta}_j^t$ and $\hat{\sigma}_j = (\hat{c}_j)^k$ when $\gamma_1 = \mu_1^{\frac{kq}{2kq-p}}\mu_2^{\frac{kq-p}{2kq-p}}$, $\gamma_2 = \mu_2$, and $k = \frac{p}{2q-1}$.*

Proof First, if $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ optimize Problem (6), we show that $\hat{c}_j = \sqrt[k]{\hat{\sigma}_j}$ and $\hat{\beta}_j^t = \hat{\alpha}_j^t/\hat{c}_j$ optimize Problem (2).

By a change of variables (replacing $\hat{\boldsymbol{\alpha}}_t$ and $\hat{\boldsymbol{\sigma}}$ by $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ in Problem (6)), we know that $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ minimize the following objective function

$$J(\hat{\beta}_j^t, \hat{c}_j) = \sum_{t=1}^{T} L(\hat{\mathbf{c}}, \hat{\boldsymbol{\beta}}_t, \mathbf{X}_t, \mathbf{y}_t) + \mu_1 \sum_{j=1}^{d} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q}\hat{c}_j^{(p-kq)/q} + \mu_2 \sum_{j=1}^{d}(\hat{c}_j)^k. \tag{12}$$

By Lemma 2 and Remark 3, the optimal $\hat{\sigma}_j = \mu_1^{\frac{1}{2}}\mu_2^{-\frac{1}{2}}\sqrt{||\hat{\boldsymbol{\alpha}}^j||_p^{p/q}}$. Because $\hat{c}_j = \sqrt[k]{\hat{\sigma}_j}$, we derive that

$$\hat{c}_j = \left(\mu_1\mu_2^{-1}||\hat{\boldsymbol{\beta}}^j||_p^{p/q}\right)^{\frac{q}{2kq-p}}.$$

Substituting the formula of $\hat{c}_j$ into Eq.(12) yields the same objective function of Problem (2) after replacing μ_1 and μ_2 by γ_1 and γ_2 with the equations $\gamma_1 = \mu_1^{\frac{kq}{2kq-p}}\mu_2^{\frac{kq-p}{2kq-p}}$, $\gamma_2 = \mu_2$. Therefore, $\hat{\boldsymbol{\beta}}_t$ and $\hat{\mathbf{c}}$ optimize Problem (2) because otherwise, if any other solution $(\boldsymbol{\beta}, \mathbf{c})$ can further reduce the objective value of Problem (2), then the corresponding $\boldsymbol{\alpha}$ and $\boldsymbol{\sigma}$ will bring the objective function of Problem (6) to a lower value than $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\sigma}}$.

Next, if $\hat{\beta}_j^t$ and $\hat{c}_j$ optimize Problem (2), we show that $\hat{\alpha}_j^t = \hat{c}_j\hat{\beta}_j^t$ and $\hat{\sigma}_j = (\hat{c}_j)^k$ optimize Problem (6).

Substituting $\hat{\alpha}_j^t$, $\hat{\sigma}_j$ for $\hat{\beta}_j^t$, $\hat{c}_j$ in Problem (2) yields an objective function

$$J(\hat{\alpha}_j^t, \hat{\sigma}_j) = \sum_{t=1}^{T} L(\hat{\boldsymbol{\alpha}}_t, \mathbf{X}_t, \mathbf{y}_t) + \gamma_1 \sum_{j=1}^{d} ||\hat{\boldsymbol{\alpha}}^j||_p^p\hat{\sigma}_j^{-(p/k)} + \gamma_2 \sum_{j=1}^{d}\hat{\sigma}_j. \tag{13}$$

We hence know that $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ minimize Eq.(13). Similar to the proof of Lemma 4, we can show that the optimal $\hat{\boldsymbol{\sigma}}$ takes the form of

$$\hat{\sigma}_j = (\gamma_1\gamma_2^{-1})^{\frac{k}{k+p}}(||\hat{\boldsymbol{\alpha}}^j||_p^p)^{\frac{k}{k+p}}.$$

Substituting the formula into Eq.(13) and setting $\gamma_1 = \mu_1^{\frac{kq}{2kq-p}}\mu_2^{\frac{kq-p}{2kq-p}}$ and $\gamma_2 = \mu_2$ transfer Eq.(13) to the objective function of Problem (6). Thus, $\hat{\alpha}_j^t$ and $\hat{\sigma}_j$ optimize Problem (6). ■

Now, based on the above two lemmas and two theorems, we can derive that when $\lambda = 2\sqrt{\gamma_1^{2-\frac{p}{kq}}\gamma_2^{\frac{p}{kq}}}$ and $q = \frac{k+p}{2k}$, the optimal solutions to Problems (2) and (5) are equivalent. Solving Problem (2) will yield an optimal solution $\hat{\boldsymbol{\alpha}}$ to Problem (5) and vice versa.

By the equivalence analysis, the proposed framework corresponds to a family of joint regularization methods as defined by Eq.(5). Assuming a convex loss function is used, this family includes some convex formulations when convex regularizers are applied to $\mathbf{A}$ in (5) and some other non-convex formulations when non-matrix-norm based regularizers are applied to $\mathbf{A}$. Particularly, when $q = p/2$, the regularization term on $\boldsymbol{\alpha}^j$ in (5) becomes the standard ℓ_p-norm. Correspondingly, when $k = p/(p-1)$ which is commonly not an integer except $p = 2$, our formulation (2) amounts to imposing a $\ell_{1,p}$-norm on $\mathbf{A}$. When both k and p take positive integers (except $p = k = 2$), Problem (2) is equivalent to using a non-matrix-norm regularizer in (5). Combinations of different p and k values will render the models different algorithmic behaviors.

Theorem 7 (Main Result 2) *For any positive k and p, if $kp \geq k+p$, then the formulation (2) imposes a convex regularizer on the model parameter matrix $\mathbf{A}$; or otherwise, it imposes a concave regularizer on $\mathbf{A}$.*

Proof According to Theorem 1, the formulation (2) is equivalent to the formulation (5) that imposes the following regularizer on $\mathbf{A}$:

$$\lambda\sum_{j=1}^{d}\sqrt{||\boldsymbol{\alpha}^j||_p^{p/q}} = \lambda\sum_{j=1}^{d}(||\boldsymbol{\alpha}^j||_p)^{\frac{p}{2q}} = \lambda\sum_{j=1}^{d}(||\boldsymbol{\alpha}^j||_p)^{\frac{kp}{k+q}}. \tag{14}$$

where $q = \frac{k+p}{2k}$ and $\lambda = 2\sqrt{\gamma_1^{2-\frac{p}{kq}}\gamma_2^{\frac{p}{kq}}}$.

A function x^a is a convex function in terms of $x > 0$ if $a \geq 1$; or otherwise, it is concave. Hence, if $kp \geq k+p$, Eq.(14) is a composite function of two functions: a convex function $x^{\frac{kp}{k+p}}$ and another convex function which is the ℓ_p vector norm of $\boldsymbol{\alpha}^j$. Thus, the overall regularizer is convex. Otherwise, if $kp < k+p$, the regularizer becomes a concave function in terms of $\boldsymbol{\alpha}$'s. ■

Remark 8 *For a particular choice of $p = 2$ and $k = 2$, Problem (2) is formulated as*

$$\min_{\boldsymbol{\beta}_t,\mathbf{c}\geq 0}\ \sum_{t=1}^{T}L(\mathbf{c},\boldsymbol{\beta}_t,\mathbf{X}_t,\mathbf{y}_t) + \gamma_1\sum_{t=1}^{T}||\boldsymbol{\beta}_t||_2^2 + \gamma_2||\mathbf{c}||_2^2, \tag{15}$$

which is used in Bi et al. (2008). This problem is equivalent to the following joint regularization method as used in Obozinski and Taskar (2006); Argyriou et al. (2007).

$$\min_{\boldsymbol{\alpha}_t}\sum_{t=1}^{T}L(\boldsymbol{\alpha}_t,\mathbf{X}_t,\mathbf{y}_t) + \lambda\sum_{j=1}^{d}\sqrt{\sum_{t=1}^{T}(\alpha_j^t)^2}, \tag{16}$$

when $\lambda = 2\sqrt{\gamma_1\gamma_2}$. Problem (16) uses the so called $\ell_{1,2}$-norm to regularize the matrix $\mathbf{A}$.

Remark 9 *For a particular choice of $p = 1$ and $k = 1$, Problem (2) is formulated as*

$$\min_{\boldsymbol{\beta}_t,\mathbf{c}\geq 0}\ \sum_{t=1}^{T}L(\mathbf{c},\boldsymbol{\beta}_t,\mathbf{X}_t,\mathbf{y}_t) + \gamma_1\sum_{t=1}^{T}||\boldsymbol{\beta}_t||_1 + \gamma_2||\mathbf{c}||_1, \tag{17}$$

which is used in Lozano and Swirszcz (2012). This problem is equivalent to the following joint regularization method

$$\min_{\boldsymbol{\alpha}_t}\sum_{t=1}^{T}L(\boldsymbol{\alpha}_t,\mathbf{X}_t,\mathbf{y}_t) + \lambda\sum_{j=1}^{d}\sqrt{\sum_{t=1}^{T}|\alpha_j^t|}, \tag{18}$$

when $\lambda = 2\sqrt{\gamma_1\gamma_2}$. Problem (17) imposes stronger sparsity on $\boldsymbol{\beta}$ and $\mathbf{c}$ (and correspondingly on $\boldsymbol{\alpha}$) than (15), which intends to shrink more model parameters to zero. Problem (18) uses a concave non-matrix-norm regularizer.

Remark 10 *For a particular choice of $p = 2$ and $k = 1$, which corresponds to the new formulation (3). This problem is equivalent to the following joint regularization method*

$$\min_{\boldsymbol{\alpha}_t}\sum_{t=1}^{T}L(\boldsymbol{\alpha}_t,\mathbf{X}_t,\mathbf{y}_t) + \lambda\sum_{j=1}^{d}\sqrt[3]{\sum_{t=1}^{T}(\alpha_j^t)^2}, \tag{19}$$

when $\lambda = 2\gamma_1^{\frac{1}{3}}\gamma_2^{\frac{2}{3}}$. Problem (19) uses a concave non-matrix-norm regularizer as well but the concavity is weaker than Problem (18) in the sense that the polynomial order is $\frac{2}{3}$ rather than $\frac{1}{2}$.

The proposed formulation (3) imposes stronger sparsity induction on the across-task component than on the task-specific component. Thus, it has stronger shrinkage effects to exclude many features for all the tasks. If the jth feature is considered as unrelated to most of the tasks, the model of $p = k = 2$ may shrink c_j to a small value but not zero, the new model (3) might shrink c_j to zero instead. Therefore this model would be more favorable to jointly learning from tasks where a large portion of noisy features exist that may be irrelevant or redundant to all of the tasks. Compared to the method in Lozano and Swirszcz (2012) where $p = k = 1$, the new formulation (3) may allow more selected features to be shared across different tasks.

Remark 11 *For a particular choice of $p = 1$ and $k = 2$, which corresponds to the new formulation (4). This problem is equivalent to the following joint regularization method*

$$\min_{\boldsymbol{\alpha}_t} \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \lambda \sum_{j=1}^{d} \sqrt[3]{\left(\sum_{t=1}^{T} |\alpha_j^t|\right)^2}, \tag{20}$$

when $\lambda = 2\gamma_1^{\frac{2}{3}}\gamma_2^{\frac{1}{3}}$. Problem (20) uses another concave non-matrix-norm regularizer that has the similar polynomial order of $\frac{2}{3}$ to Problem (19). In comparison with (19), the cross-task quadratic terms (e.g., $|\alpha_j^1||\alpha_j^2|$, $|\alpha_j^2||\alpha_j^3|$) are allowed inside the cube root in this formulation.

The new formulation (4) imposes stronger sparsity induction on task-specific component than on the across-task vector. This model can be favorable in the cases where few tasks share a limited number of selected features. As shown in our empirical results, for instance, when every two or three tasks share a limited subset of selected features but no common features are used by more than three tasks, this model performs the best among the four multiplicative MTFL formulations. In comparison to the model in Lozano and Swirszcz (2012), this model allows more of the features that are only relevant to a few tasks to be selected. In comparison with the model in Bi et al. (2008), this model can help to remove a lot of irrelevant or redundant features for individual tasks from the selected features.

We further derive another main result that characterizes the optimal across-task vector as a formula of the optimal task-specific vectors. This connection can be easily developed from the above equivalence analysis, and can help us understand the relationship and interaction between the two components.

Theorem 12 (Main Result 3) *Let $\hat{\boldsymbol{\beta}}_t$, $t = 1, \cdots, T$, be the optimal solutions of Problem (2), Let $\hat{\mathbf{B}} = [\hat{\boldsymbol{\beta}}_1, \cdots, \hat{\boldsymbol{\beta}}_T]$ and $\hat{\boldsymbol{\beta}}^j$ denote the j-th row of the matrix $\hat{\mathbf{B}}$. Then,*

$$\hat{c}_j = (\gamma_1/\gamma_2)^{\frac{1}{k}} \sqrt[k]{\sum_{t=1}^{T} (\hat{\beta}_j^t)^p}, \tag{21}$$

for all $j = 1, \cdots, d$, is optimal to Problem (2).

Proof This analytical formula can be directly derived from Theorem 5 and Theorem 6. When we set $\hat{\sigma}_j = (\hat{c}_j)^k$ and $\hat{\alpha}_j^t = \hat{c}_j \hat{\beta}_j^t$ in the proof of Theorem 6, we obtain $\hat{c}_j = \left(\mu_1 \mu_2^{-1} ||\hat{\boldsymbol{\beta}}^j||_p^{p/q}\right)^{\frac{q}{2kq-p}}$. In the same proof, we also show that $\mu_1 = \gamma_1^{\frac{2kq-p}{kq}} \gamma_2^{\frac{p-kq}{kq}}$ and $\mu_2 = \gamma_2$. Substituting these formulas into the formula of $\mathbf{c}$ yields $\hat{c}_j = (\gamma_1/\gamma_2)^{\frac{1}{k}} ||\hat{\boldsymbol{\beta}}^j||^{\frac{p}{2kq-p}}$. Moreover, to establish the equivalence between (2) and (5), we require $q = \frac{k+p}{2k}$, which leads to $k = 2kq - p$. Hence, $||\hat{\boldsymbol{\beta}}^j||^{\frac{p}{2kq-p}} = \sqrt[k]{\sum_{t=1}^{T} (\hat{\beta}_j^t)^p}$. We then obtain the formula (21). ■

Remark 13 *For particular choices of p and k, the relation between the optimal $\mathbf{c}$ and $\boldsymbol{\beta}$ can be computed according to Theorem 12. Table 2 summarizes the relation formula for the common choices when $p \in \{1, 2\}$ and $k \in \{1, 2\}$.*

Table 2: The shrinkage effect of $\mathbf{c}$ with respect to $\boldsymbol{\beta}$ for four common choices of p and k.

p	k	$\mathbf{c}$
2	2	$\hat{c}_j = \sqrt{\gamma_1\gamma_2^{-1}}\sqrt{\sum_{t=1}^{T}(\hat{\beta}_j^t)^2}$
1	1	$\hat{c}_j = \gamma_1\gamma_2^{-1}\sum_{t=1}^{T}\lvert\hat{\beta}_j^t\rvert$
2	1	$\hat{c}_j = \gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\hat{\beta}_j^t)^2$
1	2	$\hat{c}_j = \sqrt{\gamma_1\gamma_2^{-1}}\sqrt{\sum_{t=1}^{T}\lvert\hat{\beta}_j^t\rvert}$

4. Probabilistic Interpretation

In this section we show that the proposed multiplicative formalism is related to the *maximum a posteriori* (MAP) solution of a probabilistic model. Let $p(\mathbf{A}|\boldsymbol{\Delta})$ be the prior distribution of the weight matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_T] = [\boldsymbol{\alpha}^{1\top}, \ldots, \boldsymbol{\alpha}^{d\top}]^\top \in \mathbb{R}^{d\times T}$, where $\boldsymbol{\Delta}$ denote the parameter of the prior. Then the *a posteriori* distribution of $\mathbf{A}$ can be calculated via Bayes rule as

$$p(\mathbf{A}|\mathbf{X}, \mathbf{y}, \boldsymbol{\Delta}) \propto p(\mathbf{A}|\boldsymbol{\Delta}) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{X}_t, \boldsymbol{\alpha}_t). \tag{22}$$

Denote $z \sim \mathcal{GN}(\mu, \rho, p)$ the *univariate generalized normal distribution*, with the density function

$$p(z) = \frac{1}{2\rho\Gamma(1 + 1/p)} \exp\left(-\frac{|z-\mu|^p}{\rho^p}\right), \tag{23}$$

in which $\rho > 0$, $p > 0$, and $\Gamma(\cdot)$ is the Gamma function (Goodman and Kotz, 1973). Now let each element of $\mathbf{A}$, α_j^t, follow a generalized normal prior, $\alpha_j^t \sim \mathcal{GN}(0, \delta_j, p)$. Then with the *independent and identically distributed* assumption, the prior takes the form (also refer to Zhang et al. (2010) for a similar treatment)

$$p(\mathbf{A}|\boldsymbol{\Delta}) \propto \prod_{j=1}^{d}\prod_{t=1}^{T} \frac{1}{\delta_j} \exp\Big(-\frac{|\alpha_j^t|^p}{\delta_j^p}\Big) = \prod_{j=1}^{d} \frac{1}{\delta_j^T} \exp\Big(-\frac{\|\boldsymbol{\alpha}^j\|_p^p}{\delta_j^p}\Big), \tag{24}$$

where $\|\cdot\|_p$ denotes the vector p-norm. With an appropriately chosen likelihood function $p(\mathbf{y}_t|\mathbf{X}_t, \boldsymbol{\alpha}_t) \propto \exp(-L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t))$, finding the MAP solution is equivalent to solving the following problem:

$$\min_{\mathbf{A},\boldsymbol{\Delta}} \quad J = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \sum_{j=1}^{d} \Big(\frac{\|\boldsymbol{\alpha}^j\|_p^p}{\delta_j^p} + T\ln\delta_j\Big). \tag{25}$$

Setting the derivative of J with respect to δ_j to zero, we obtain:

$$\delta_j = \Big(\frac{p}{T}\Big)^{1/p} \|\boldsymbol{\alpha}^j\|_p. \tag{26}$$

Bringing this back to (25), we have the following equivalent problem:

$$\min_{\mathbf{A}} \quad J = \sum\nolimits_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + T \sum\nolimits_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_p. \tag{27}$$

Now let us look at the multiplicative nature of α_j^t with different $p \in [1, \infty)$.

When $p = 1$, we have:

$$\begin{aligned}
\sum_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_1 &= \sum_{j=1}^{d} \ln \left(\sum_{t=1}^{T} |\alpha_j^t| \right) \\
&= \sum_{j=1}^{d} \ln \left(\sum_{t=1}^{T} |c_j \beta_j^t| \right) \\
&= \sum_{j=1}^{d} \left(\ln |c_j| + \ln \sum_{t=1}^{T} |\beta_j^t| \right) \\
&\leq \sum_{j=1}^{d} |c_j| + \sum_{j=1}^{d} \sum_{t=1}^{T} |\beta_j^t| - 2d,
\end{aligned}$$

where the inequality is because of the fact that $\ln z \leq z - 1$ for any $z > 0$. Therefore we can optimize an upper bound of J in (27),

$$\min_{\mathbf{A}} \; J_1 = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + T \sum_{j=1}^{d} |c_j| + T \sum_{j=1}^{d} \sum_{t=1}^{T} |\beta_j^t| - 2dT, \tag{28}$$

which is equivalent to the multiplicative formulation (2) where $\{p, k\} = \{1, 1\}$. This proves the following theorem:

Theorem 14 *When $\{p, k\} = \{1, 1\}$, optimizing the multiplicative formulation* (2) *is equivalent to maximizing a lower bound of the MAP solution under probabilistic model* (22) *with $p = 1$ in the prior definition.*

In the general case, we have:

$$\begin{aligned}
\sum_{j=1}^{d} \ln \|\boldsymbol{\alpha}^j\|_p &= \frac{1}{p} \sum_{j=1}^{d} \ln \left(\sum_{t=1}^{T} |c_j \beta_j^t|^p \right) \\
&= \frac{1}{p} \sum_{j=1}^{d} \ln \left((c_j^k)^{\frac{p}{k}} \cdot \sum_{t=1}^{T} |\beta_j^t|^p \right) \\
&= \frac{1}{k} \sum_{j=1}^{d} \ln |c_j|^k + \frac{1}{p} \sum_{j=1}^{d} \ln \sum_{t=1}^{T} |\beta_j^t|^p \\
&\leq \frac{1}{k} \sum_{j=1}^{d} |c_j|^k + \frac{1}{p} \sum_{t=1}^{T} \|\boldsymbol{\beta}_t\|_p^p - d(\frac{1}{k} + \frac{1}{p}).
\end{aligned}$$

These inequalities lead to an upper bound of J in (27). By minimizing the upper bound, the problem is formulated as:

$$\min_{\mathbf{A}} \quad J_{p,k} = \sum_{t=1}^{T} L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t) + \frac{T}{k} \|\mathbf{c}\|_k^k + \frac{T}{p} \sum_{t=1}^{T} \|\boldsymbol{\beta}_t\|_p^p, \tag{29}$$

which is equivalent to the general multiplicative formulation in (2). Therefore we have proved the following theorem:

Theorem 15 *Optimizing the multiplicative formulation* (2)*, in the form of* (29)*, is equivalent to maximizing a lower bound of the MAP solution under probabilistic model* (22) *with $p \in (1, \infty)$ in the prior definition.*

5. Optimization Algorithm

Alternating optimization algorithms have been used in both of the early methods (Bi et al., 2008; Lozano and Swirszcz, 2012) to solve Problem (2) which alternate between solving two subproblems: solve for $\boldsymbol{\beta}_t$ with fixed $\mathbf{c}$; solve for $\mathbf{c}$ with fixed $\boldsymbol{\beta}_t$. The convergence property of such an alternating algorithm has been analyzed in Bi et al. (2008); Lozano and Swirszcz (2012) that it converges to a local minimizer. In these early methods, both of the two subproblems have to be solved using iterative algorithms such as gradient descent, linear or quadratic program solvers.

Besides the algorithm that solves for $\mathbf{c}$ and $\boldsymbol{\beta}_t$ alternatively and can be applied to our formulations, we design an alternating optimization algorithm that utilizes the closed-form solution for $\mathbf{c}$ we have derived in Lemma 4 and the property that both Problems (2) and (5) are equivalent to the intermediate Problem (6) (or Problem (9)). In the algorithm to solve Problem (2), we start from an initial choice of $\mathbf{c}$. At iteration s, we start from $\mathbf{c}^s$, and solve for $\boldsymbol{\beta}_t^s$ with the fixed $\mathbf{c}^s$. We then compute the value of $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$, which is used to update $\mathbf{c}^s$ to $\mathbf{c}^{s+1}$ according to Eq.(10) in Lemma 4. The overall procedure is summarized in **Algorithm 1**. As a central idea in designing this algorithm, at each iteration we update $\boldsymbol{\beta}$ to reduce the loss function, and update $\mathbf{c}$ to reduce the regularizer while maintaining the same loss function value. Note that if the value of $\boldsymbol{\alpha}$ is fixed, the loss will remain the same.

To analyze the convergence property of Algorithm 1, we utilize the fact that Problem (2) and Problem (9) are equivalent. For notational convenience, we denote the objective function of Problem (2) by $g(\mathbf{B}, \mathbf{c})$ that takes inputs $\boldsymbol{\beta}_t$ and $\mathbf{c}$. We denote the objective function of Problem (9) by $f(\mathbf{A}, \mathbf{c})$. Both objective functions comprise the sum of three parts. For instance, f can be written as follows:

$$f(\mathbf{A}, \mathbf{c}) = f_0(\mathbf{A}, \mathbf{c}) + f_{\mathbf{A}}(\mathbf{A}) + f_{\mathbf{c}}(\mathbf{c}), \tag{31}$$

where $f_0(\mathbf{A}, \mathbf{c}) = \gamma_1 \sum_{j=1}^{d} (c_j^{-p} \sum_{t=1}^{T} (\alpha_j^t)^p)$ is the part that involves both $\boldsymbol{\alpha}$ and $\mathbf{c}$, $f_{\mathbf{A}}(\mathbf{A}) = \sum_t L(\boldsymbol{\alpha}_t, \mathbf{X}_t, \mathbf{y}_t)$ is the part relying only on $\boldsymbol{\alpha}$, and $f_{\mathbf{c}}(\mathbf{c}) = \mu_2 \sum_{j=1}^{d} (c_j)^k$ is the part for $\mathbf{c}$ only. Let $\mathbf{z}$ be the vector consisting of all variables in Problem (9). Similar to what has been defined in Tseng (2001), we define that the point $\mathbf{z}$ is a stationary point of f if $\mathbf{z} \in dom\ f$ where $dom\ f$ is the feasible region of f, and

$$\lim_{\lambda \to 0} [f(\mathbf{z} + \lambda \mathbf{b}) - f(\mathbf{z})]/\lambda \geq 0, \quad \forall \mathbf{b} \quad \text{such that} \quad (\mathbf{z} + \lambda \mathbf{b}) \in dom\ f. \tag{32}$$

Algorithm 1 The blockwise coordinate descent algorithm for multiplicative MTFL

Input: $\mathbf{X}_t$, $\mathbf{y}_t$, $t = 1, \cdots, T$, as well as γ_1, γ_2, p and k
Initialize: $c_j = 1$, $\forall j = 1, \cdots, d$, and $s = 1$
repeat
  Compute $\tilde{\mathbf{X}}_t = \mathbf{X}_t \text{diag}(\mathbf{c}^s)$, $\forall\, t = 1, \cdots, T$
  for $t = 1, \cdots, T$ **do**
    Solve the following problem for $\boldsymbol{\beta}_t^s$

$$\min_{\boldsymbol{\beta}_t} L(\boldsymbol{\beta}_t, \tilde{\mathbf{X}}_t, \mathbf{y}_t) + \gamma_1 ||\boldsymbol{\beta}_t||_p^p \tag{30}$$

  end for
  Compute $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$
  Set $s = s + 1$
  Compute $\mathbf{c}^{s+1}$ using $\boldsymbol{\alpha}_t^s$ according to Eq.(10)
until $\max_{t,j}(|(\alpha_j^t)^s - (\alpha_j^t)^{s-1}|) < \epsilon$ (or other proper termination rules)
Output: $\boldsymbol{\alpha}_t$, $\mathbf{c}$ and $\boldsymbol{\beta}_t$, $t = 1, \cdots, T$

where $\mathbf{b}$ denotes any feasible direction, (which corresponds to $\bigtriangledown f(\mathbf{z}) = 0$ if f is differentiable, or $\mathbf{0} \in \partial f(\mathbf{z})$ if f is non-differentiable where $\partial f(\mathbf{z})$ is the subgradient of f at $\mathbf{z}$). In our case, f is not differentiable at $\boldsymbol{\alpha} = 0$ or $\mathbf{c} = 0$ when p is set to an odd number. We also define that a point $\mathbf{z}$ is a coordinate-wise minimum point of f if $\mathbf{z} \in dom\ f$, and $\forall \mathbf{b}_k \in \mathbb{R}^{d_k}$ that makes $(0, \cdots, \mathbf{b}_k, \cdots, 0)$ a feasible direction, there exists a small $\epsilon > 0$, such that for all positive $\lambda \leq \epsilon$,

$$f(\mathbf{z} + \lambda(0, \cdots, \mathbf{b}_k, \cdots, 0)) \geq f(\mathbf{z}), \tag{33}$$

where k indexes the blocks of variables in our algorithm, which include $\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T$ and $\mathbf{c}$, so $k = \{1, \cdots, T+1\}$, and the $(T+1)$-th block is for $\mathbf{c}$, d_k is the number of variables in the kth coordinate block and in our case $d_k = d$. The vector $(0, \cdots, \mathbf{b}_k, \cdots, 0)$ is a vector in $\mathbb{R}^{d \times (T+1)}$ and used to only vary $\mathbf{z}$ in the k-th block.

We first prove that for the sequence of points generated by Algorithm 1, the objective function f is monotonically non-increasing. Then we prove the sequence of points is bounded, because of which, the sequence will have accumulation points. We prove that each accumulation point is a coordinate-wise minimum point. Then according to Tseng (2001), if f is regular at an accumulation point $\mathbf{z}^*$, this $\mathbf{z}^*$ is a stationary point.

Lemma 16 *Let the sequence of iterates generated by Algorithm 1 be* $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ *and the function* f *is defined by (31), then* $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$.

Proof First, note that for each $\{\mathbf{A}^s, \mathbf{c}^s\}$, we accordingly have $\{\mathbf{B}^s, \mathbf{c}^s\}$ where $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$, $\forall t$, and we have $f(\mathbf{A}^s, \mathbf{c}^s) = g(\mathbf{B}^s, \mathbf{c}^s)$. Because we start with $\mathbf{c}$ so at each iteration $\mathbf{c}$ gets updated first (the order dose not matter actually). At iteration $s+1$, we compute $\mathbf{c}^{s+1}$ based on $\mathbf{A}^s$ according to Eq.(10). According to Lemma 4, $\mathbf{c}^{s+1}$ will reach the lower bound of f when $\mathbf{A}$ is fixed to $\mathbf{A}^s$. Hence $f(\mathbf{A}^s, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$. Moreover, when $\mathbf{c}$ is updated, there is an implicit new value of $\tilde{\mathbf{B}}^s$ that is just computed as $(\tilde{\beta}_j^t)^s = (\alpha_j^t)^s/(c_j)^{s+1}$,

$\forall j$ and t. Then, $g(\tilde{\mathbf{B}}^s, \mathbf{c}^{s+1}) = f(\mathbf{A}^s, \mathbf{c}^{s+1})$. Next in Algorithm 1, we obtain $\mathbf{B}^{s+1}$ by solving Problem (30), i.e., by optimizing g with respect $\mathbf{B}$ when $\mathbf{c}$ is fixed to $\mathbf{c}^{s+1}$. Hence, $g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\tilde{\mathbf{B}}^s, \mathbf{c}^{s+1})$. Then $\mathbf{A}$ will be updated by $\mathbf{A}^{s+1} = \text{diag}(\mathbf{c}^{s+1})\mathbf{B}^{s+1}$, which leads to $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) = g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1})$. Overall, we have

$$f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) = g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\tilde{\mathbf{B}}^s, \mathbf{c}^{s+1}) = f(\mathbf{A}^s, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s).$$

This proves that $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$. ■

Based on the proof of Lemma 16, we can also show that $g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\mathbf{B}^s, \mathbf{c}^s)$ for the sequence of $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ that is also created by Algorithm 1.

Lemma 17 *The sequence of iterates generated by Algorithm 1,* $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$, *(or equivalently,* $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$*) is bounded.*

Proof According to Lemma 16, we know that $f(\mathbf{A}^{s+1}, \mathbf{c}^{s+1}) \leq f(\mathbf{A}^s, \mathbf{c}^s)$, and $g(\mathbf{B}^{s+1}, \mathbf{c}^{s+1}) \leq g(\mathbf{B}^s, \mathbf{c}^s)$, $\forall s = 1, 2, \cdots$. Hence, $g(\mathbf{B}^s, \mathbf{c}^s) \leq g(\mathbf{B}^1, \mathbf{c}^1)$, $\forall s = 1, 2, \cdots$. Let $g(\mathbf{B}^1, \mathbf{c}^1) = C$. Then, $g(\mathbf{B}^s, \mathbf{c}^s)$ is upper bounded by C.

In our algorithm, we assume that the loss function in g can be either the least squares loss or the logistic regression loss of all the tasks. Hence, the loss terms are non-negative (actually most other loss functions, such as the hinge loss, are also non-negative). The two regularizers, one on $\boldsymbol{\beta}_t$ and the other on $\mathbf{c}$, are both non-negative. Thus, we have that $||\boldsymbol{\beta}_t^s||_p^p \leq C/\gamma_1$ and $||\mathbf{c}^s||_k^k \leq C/\gamma_2$, $\forall s = 1, 2, \cdots$. This shows that the sequence of $\{\mathbf{B}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ is bounded. Because $\boldsymbol{\alpha}_t^s = \text{diag}(\mathbf{c}^s)\boldsymbol{\beta}_t^s$, $||\boldsymbol{\alpha}_t^s||_p^p = \sum_{j=1}^d |(\alpha_j^t)^s|^p = \sum_{j=1}^d |(c_j^s(\beta_j^t)^s|^p \leq ||\mathbf{c}^s||_{2p}^{2p} + ||\boldsymbol{\beta}_t^s||_{2p}^{2p} \leq \tilde{C}$ where $\tilde{C}$ is a constant computed from C, γ_1 and γ_2. Thus, the sequence of $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ is also bounded. ■

Theorem 18 *The sequence* $\mathbf{z}^s = \{\mathbf{A}^s, \mathbf{c}^s\}_{s=1,2,\cdots}$ *generated by Algorithm 1 has at least one accumulation point. For any accumulation point* $\mathbf{z}^* = \{\mathbf{A}^*, \mathbf{c}^*\}$, $\mathbf{z}^*$ *is a coordinate-wise minimum point of* f.

Proof According to Lemma 17, the sequence $\mathbf{z}^s$ is bounded, so there exists at least one accumulation point $\mathbf{z}^*$ and a subsequence of $\{\mathbf{z}^s\}_{s=1,2,\cdots}$ that converges to $\mathbf{z}^*$. Without loss of generality and for notational convenience, let us just assume that $\{\mathbf{z}^s\}_{s=1,2,\cdots}$ converges to $\mathbf{z}^*$. Because $\mathbf{z}^*$ is an accumulation point, if it is the iterate at the current iteration s, then in the next iteration $s+1$, the same iterate $\mathbf{z}^*$ will be obtained. Hence, $\boldsymbol{\beta}_t^*$ is the optimal solution of Problem (30) when $\mathbf{c}$ is set to $\mathbf{c}^*$ (and all other $\boldsymbol{\beta}_{k \neq t} = \boldsymbol{\beta}_k^*$). Correspondingly, $\mathbf{c}^*$ is the optimal solution of Problem (2) when $\mathbf{B}$ is set to $\mathbf{B}^*$. Hence, for any feasible direction $(0, \cdots, \mathbf{b}_t, \cdots, 0)$, $\forall t = 1, 2, \cdots, T+1$, we have

$$f(\mathbf{z}^* + \lambda(0, \cdots, \mathbf{b}_t, \cdots, 0)) \geq f(\mathbf{z}^*), \tag{34}$$

for small λ values. Hence, $\mathbf{z}^*$ is a coordinate-wise minimum point of f. ■

Theorem 19 *If both p and k are positive, and $k \geq 1$, the accumulation point $\mathbf{z}^*$ is a stationary point of f.*

Proof Due to Lemma 4, we know that the optimal solution of Problem (9) can only occur when $\mathbf{c}$ and $\boldsymbol{\alpha}_t$ satisfy Eq.(10), so any other points can be excluded from discussion. Hence, we define a new level set $Z^0 = \{\mathbf{z}|f(\mathbf{z}) \leq C\} \cap \{\mathbf{z}|c_j = (\gamma_1\gamma_2^{-1}\sum_{t=1}^{T}(\alpha_j^t)^p)^{\frac{1}{p+k}}, \forall \alpha_j^t\}$. We now prove that f is continuous on this set Z^0 and the gradient of f exists when p is even (differentiable case), and examine $\mathbf{0} \in \partial f$ at $\mathbf{z}^*$ for non-differentiable cases.

Given the definition of f, f is continuous with respect to α_j^t, $\forall j$ and t. Because f_0 is a division term that has a divisor based on c_j, we have the *division-by*-0 issue, so f is in general not continuous with respect to c_j at 0 (but continuous and differentiable at other values). However, using L'Hospital's rule (i.e., $\lim_{\phi\to 0}\frac{f_1(\phi)}{f_2(\phi}) = \lim_{\phi\to 0}\frac{f_1'(\phi)}{f_2'(\phi}$), we show that f is continuous with respect to c_j at $c_j = 0$ in the set Z^0. When $c_j = 0$, $||\boldsymbol{\alpha}^j||_p^p = \sum_{t=1}^{T}(\alpha_j^t)^p = 0$ due to Eq.(10). Let $\phi = ||\boldsymbol{\alpha}^j||_p^p$. Since the function f_0 is a ratio of two items both approaching 0 as functions of ϕ, we can apply L'Hospital's rule as follows

$$\lim_{\phi\to 0}\frac{||\boldsymbol{\alpha}^j||_p^p}{(c_j)^p} = \lim_{\phi\to 0}\frac{\phi}{(\gamma_1\gamma_2^{-1}\phi)^{\frac{p}{p+k}}} = \lim_{\phi\to 0}\frac{1}{\frac{p}{p+k}(\gamma_1\gamma_2^{-1})^{\frac{p}{p+k}}\phi^{-\frac{k}{p+k}}} = \lim_{\phi\to 0}\frac{(p+k)\phi^{\frac{k}{p+k}}}{p(\gamma_1\gamma_2^{-1})^{\frac{p}{p+k}}} = 0.$$

We then compute the partial derivative of f_0 with respect to c_j, which is $\frac{\partial f_0}{\partial c_j} = \frac{-p||\boldsymbol{\alpha}^j||_p^p}{(c_j)^{p+1}}$ for $c_j \neq 0$. Now when c_j approaches 0, we can prove continuity using L'Hospital's rule:

$$\frac{\partial f_0}{\partial c_j}|_{c_j\to 0} = \lim_{\phi\to 0}\frac{-p||\boldsymbol{\alpha}^j||_p^p}{(c_j)^{p+1}} = \lim_{\phi\to 0}\frac{-p\phi}{(\gamma_1\gamma_2^{-1}\phi)^{\frac{p+1}{p+k}}} = \lim_{\phi\to 0}\frac{-p(p+k)\phi^{\frac{k-1}{p+k}}}{(p+1)(\gamma_1\gamma_2^{-1})^{\frac{(p+1)}{p+k}}} = 0,$$

when $k > 1$. When $k = 1$, the limit is a finite number $-p(p+k)/((p+1)(\gamma_1\gamma_2^{-1})^{\frac{(p+1)}{p+k}})$. Note that when an odd p is taken, $||\boldsymbol{\alpha}_t||_p^p = \sum|\alpha_j^t|^p$ is not differentiable at 0. However, the above limit exists no matter f is differentiable or not because we take ϕ as the varying parameter. With the above conditions, we use the results in Tseng (2001) that when each subproblem has a unique minimum, which is the case in our algorithm because Subproblem (30) is strictly convex (for our chosen loss functions) and we have already proved the unique analytical solution of $\mathbf{c}$, $\mathbf{z}^*$ is a stationary point of f. ■

We briefly discuss the computation cost of Algorithm 1. Subproblem (30) is essentially for single task learning, which can be solved by many existing efficient algorithms, such as gradient-based optimization methods. The second subproblem has a closed-form solution for $\mathbf{c}$, which requires only a minimal level of computation. The computation cost of Algorithm 1 only linearly increases with the number of tasks. Due to the nature of Algorithm 1, it can be easily parallelizable and distributed to multiple processors when optimizing individual $\boldsymbol{\beta}_t$. More efficient algorithms may be designed for specific choices of p in the future.

6. Experiments

We empirically evaluated the performance of the multiplicative MTFL algorithms on both synthetic data sets and a variety of real-world data sets, where we solved either classification (using the logistic regression loss) or regression (using the least squares loss) problems. In the experiments, we implemented and compared Algorithm 1 for four parameter settings: $(p, k) = (2, 2)$, $(1, 1)$, $(2, 1)$, and $(1, 2)$, corresponding to four multiplicative MTFL (MMTFL) methods as listed in Table 2. Although when the values of (p, k) were (2,2) and (1,1), the two models corresponded respectively to the same methods in Bi et al. (2008) and Lozano and Swirszcz (2012), they were solved differently from prior methods using our Algorithm 1 with higher computational efficiency. When $(p, k) = (2, 1)$ and $(1, 2)$, the resultant models corresponded to the two new formulations.

In our experiments, the multiplicative MTFL methods were also compared with the additive MTFL methods that decompose the model parameters into an addition of two components, such as the Dirty model (DMTL) (Jalali et al., 2010) and the robust MTFL (rMTFL) (Gong et al., 2012). Single task learning (STL) approaches were also implemented as baselines and compared with all of the MTFL algorithms in the experiments. We list the various methods used for comparison in our experiments as follows:

- STL_lasso : Learning each task independently with $||\boldsymbol{\alpha}_t||_1$ as the regularizer.
- STL_ridge : Learning each task independently with $||\boldsymbol{\alpha}_t||_2^2$ as the regularizer.
- DMTL (Jalali et al., 2010) : The dirty model with regularizers $||\mathbf{P}||_{1,1}$ and $||\mathbf{Q}||_{1,\infty}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.
- rMTFL (Gong et al., 2012) : Robust multitask feature learning with the regularizers $||\mathbf{P}||_{1,2}$ and $||\mathbf{Q}^T||_{1,2}$, where $\mathbf{A} = \mathbf{P} + \mathbf{Q}$.
- MMTFL$\{2, 2\}$ (Bi et al., 2008) : Multiplicative multitask feature learning with regularizers $||\mathbf{B}||_F^2$ and $||\mathbf{c}||_2^2$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.
- MMTFL$\{1, 1\}$ (Lozano and Swirszcz, 2012) : Multiplicative multitask feature learning with regularizers $||\mathbf{B}||_{1,1}$ and $||\mathbf{c}||_1$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.
- MMTFL$\{2, 1\}$(New formulation 1) : Multiplicative multitask feature learning with regularizers $||\mathbf{B}||_F^2$ and $||\mathbf{c}||_1$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.
- MMTFL$\{1, 2\}$(New formulation 2) : Multiplicative multitask feature learning with regularizers $||\mathbf{B}||_{1,1}$ and $||\mathbf{c}||_2^2$, where $\mathbf{A} = \text{diag}(\mathbf{c})\mathbf{B}$.

In all experiments, unless otherwise noted, the original data set was partitioned to have 25%, 33% or 50% of the data in a training set and the rest used for testing. For each specified partition ratio (corresponding to a trial), we randomly partitioned the data 15 times and reported the average performance. The same tuning process was used to tune the hyperparameters (e.g.,γ_1 and γ_2) of every method in the comparison. In every trial, an internal three-fold cross validation (CV) was performed within the training data of the first partition to select a proper hyperparameter value for each of the methods from the choices of

2^k with $k = -10, -9, \cdots, 7$. In the subsequent partitions of each trial, the hyperparameters were fixed to the values that yielded the best performance in the CV.

The regression performance was measured by the coefficient of determination, denoted by R^2, which measures the explained variance of the data by the fitted model. In particular, we used the following formula to report performance $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - f_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$ where $\bar{y}$ is the mean of the observed values of y and f_i is the prediction of the observed y_i. The reported values in our experiments were the averaged R^2 over all tasks. The R^2 value ranges from 0 to 1, and a higher value indicates better regression performance. The classification performance was measured by the F1 score, which is the harmonic mean of precision and recall. The numbers we reported in each trial was the F1 score averaged over all tasks. Similarly, the F1 score also ranges from 0 to 1 and higher values represent better classification performance.

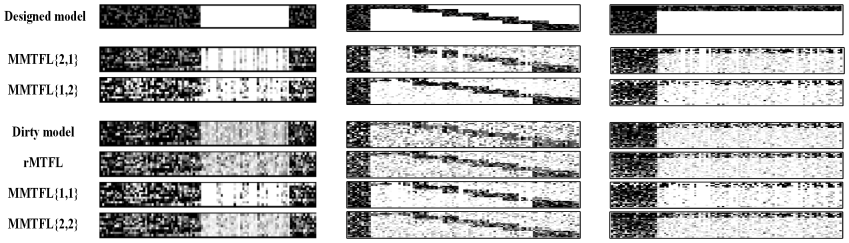
6.1 Simulation Studies

We created three categories of data sets, all of which were designed for regression experiments to evaluate the behaviors of the different methods. The data sets in each category were created with a pre-specified feature sharing structure. The first two categories were designed to validate the scenarios that we hypothesized for our two new formulations to work. We performed sensitivity analyses using these two categories of data sets, i.e., studying how performance was altered when the number of tasks or the number of features varied. Because we also empirically compared with a few additive decomposition based methods, it would be interesting to see how multiplicative MTFL behaved in a scenario that was actually in favor of additive MTFL. Hence, in the third category, the feature sharing structure was generated following the assumption that the robust MTFL method used (Gong et al., 2012).

In all experiments, we created input examples $\mathbf{X}_t$ for each task t using a number of features randomly drawn from the standard multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$, and pre-defined $\mathbf{A} = [\boldsymbol{\alpha}_1, \cdots, \boldsymbol{\alpha}_T]$ across all tasks. The responses for each task was computed by $\mathbf{y}_t = \mathbf{X}_t\boldsymbol{\alpha}_t + \boldsymbol{\epsilon}_t$ where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, 0.5)$ was the noise introduced to the model. If an entry of $\mathbf{A}$ was set to non-zero, its value was randomly sampled from a uniform distribution in the interval $[0.5, 1.5]$. As a side note, we transposed $\mathbf{A}$ in Figure 1 for better illustration.

6.1.1 Synthetic Datasets

Category 1 (C1). In the C1 experiments, 40% of the rows in the matrix $\mathbf{A}$ were set to 0. Since each row corresponded to a feature across the tasks, the zero rows made the features irrelevant to all tasks. All remaining features received non-zero values in $\mathbf{A}$ so that they were used in every task's model although with different combination weights. Hence, the individual models were sparse with respect to all synthesized features, but not sparse with respect to the selected features. The pre-defined $\mathbf{A}$ is demonstrated in Figure 1a(top), where we transpose $\mathbf{A}$ to have columns representing features and a darker spot indicates that the particular element of $\mathbf{A}$ had a larger absolute value. Note that this synthetic data follows the assumption that has motivated the early block-wise joint regularized methods that used matrix-norm-based regularizers. Those methods were developed with an assumption that a subset of features was shared by all tasks. However, we observed that the level of shrinkage needed would be different for $\mathbf{c}$ and $\boldsymbol{\beta}$, which corresponded to non-matrix-norm-



(a) Synthetic data C1 (b) Synthetic data C2 (c) Synthetic data C3

Figure 1: The true parameter matrix versus the parameter matrices constructed by the various methods on synthetic data. The figure shows the results when 200 examples and 100 features were created for 20 tasks each. Darker color indicates larger values in magnitude.

based regularizers. We hypothesized that the proposed new formulation (3) would produce better regression performance than existing models on this data category.

To examine how the number of tasks influenced the performance of different methods, we varied the number of tasks from 1, 5, 10, 50, and 100 to 1000. For each task, we created 200 examples, each represented by 100 features. We also tested the different methods when increasing the number of features. In this set of experiments, we used 20 tasks and each task contained 1000 examples. Each example was represented by a number of features ranging from 200 to 1000 with a step size of 200.

Category 2 (C2). In the C2 experiments, 10% of the rows in $\mathbf{A}$ were set to non-zero and these features were shared by all tasks. We then arranged the tasks to follow six different sparse structures (the staircases) as shown in Figure 1b(top), where we once again transpose $\mathbf{A}$. Each of the remaining features except the 10% common features was used by a comparatively small proportion of the tasks. Consecutive tasks were grouped such that the neighboring groups of tasks shared 7% of the features besides the 10% common features, whereas the non-neighboring groups of tasks did not share any features. Therefore, no feature could be excluded from all tasks, but a majority of individual features (90%) was only useful for few tasks (i.e., the useful features for one task were sparse). In this case, the non-sparsity-inducing norm was suitable for regularizing $\mathbf{c}$ and sparsity-inducing norm was more suitable for regularizing $\boldsymbol{\beta}$. We hypothesized that the new formulation (4) would produce better regression performance than the other models on this data set.

We created 20, 50, 100, 500 and 1000 tasks, respectively, to test the algorithms' sensitivity to the number of tasks. The numbers of tasks were chosen to make sure enough tasks in each of the six groups. The number of tasks in each group ranged from 3 to 170. We generated 200 examples and 100 features for each task. We also created another set of C2 data sets with the number of features changing from 200 to 1000 with a step size of 200 for 20 tasks and 1000 examples for each task.

Category 3 (D3). This category was synthesized following the model of the additive decomposition methods. It only contained one data set where 200 examples, each represented

Table 3: Comparison of the test R^2 values obtained by the different MTFL methods on synthetic data sets using different partition ratios for training data (where standard deviation 0 means that it is less than 0.01).

Dataset		STL_lasso	STL_ridge	DMTL	rMTFL	MMTFL(2,2)	MMTFL(1,1)	MMTFL(2,1)	MMTFL(1,2)
C1	25%	0.32±0.02	0.45±0.01	0.60±0.02	0.58±0.02	0.64±0.02	0.54±0.03	**0.73±0.02**	0.42±0.04
	33%	0.54±0.03	0.62±0.02	0.73±0.01	0.61±0.02	0.79±0.02	0.76±0.01	**0.86±0.01**	0.65±0.03
	50%	0.70±0.02	0.86±0.01	0.75±0.01	0.66±0.01	0.86±0.01	0.88±0.01	**0.90±0.01**	0.84±0.01
C2	25%	0.42±0.03	0.33±0.01	0.36±0.01	0.46±0.01	0.45±0.01	0.35±0.05	0.46±0.02	**0.49±0.02**
	33%	0.77±0.02	0.63±0.01	0.42±0.02	0.63±0.03	0.69±0.02	0.75±0.01	0.67±0.03	**0.83±0.02**
	50%	0.95±0.01	0.89±0.01	0.81±0.01	0.83±0.01	0.91±0	0.95±0	0.92±0.01	**0.97±0**
C3	25%	0.28±0.03	0.43±0.03	0.50±0.04	**0.55±0.03**	0.54±0.03	0.31±0.02	0.43±0.02	0.34±0.02
	33%	0.47±0.01	0.56±0.02	0.60±0.02	**0.65±0.02**	0.64±0.02	0.45±0.04	0.60±0.04	0.47±0.03
	50%	0.77±0.01	0.78±0.01	0.76±0.02	**0.83±0.01**	0.81±0.02	0.75±0.01	0.79±0.02	0.76±0.01

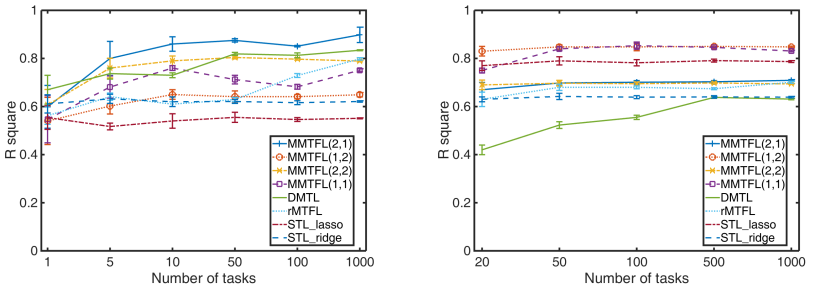
by 100 features, were generated for each of 20 tasks. The parameter matrix $\mathbf{A} = \mathbf{P} + \mathbf{Q}$ where 80 rows in $\mathbf{P}$ and 16 columns in $\mathbf{Q}$ were set to 0. The component $\mathbf{P}$ was used to indicate the subset of relevant features across all tasks, and the component $\mathbf{Q}$ was used to tell that there were outlier tasks that did not share features with other tasks. Given this simulation process, this data set would be in favor of the rMTFL model proposed in Gong et al. (2012). The designed model parameter matrix was shown in Figure 1c(top).

6.1.2 Performance on synthetic data sets

We first compared the regression performance of the different methods on the three categories of data sets. Table 3 shows the averaged R^2 values together with standard deviations for each method and each trial setting. The best results are shown in bold fonts. The results in Table 3 were obtained on synthetic data sets that had 20 tasks with 200 examples and 100 features for each task. We reported the test R^2 obtained on each data set when 25%, 33% and 50% of the data were used in training. From Table 3, we observe that the proposed formulation (3)(MMTFL(2,1)) consistently outperformed other models on *C1* data sets, whereas the proposed model (4)(MMTFL(1,2)) consistently outperformed on *C2* data sets. The results confirmed our hypotheses that the two proposed models could be more suitable for learning the type of sharing structures in *C1* and *C2*. As anticipated, rMTFL model constantly outperformed other models on the *C3* data set. Among the multiplicative MTFL methods, MMTFL(2,2) achieved similar performance to that of rMTFL (off only by $0.01 \sim 0.02$ for average R^2).

In order to elucidate the different shrinkage effects of the different decomposition strategies and regularizers, we compared the true parameter matrix with the constructed parameter matrices by the six MTFL methods used in our experiments in Figure 1. From the results on the *C1* and *C2* data sets, we observe that only MMTFL(2,1), MMTFL(1,2) and MMTFL(1,1) produced reasonably sparse structures. The two additive decomposition methods could not yield a sufficient level of sparsity in the models. Although the unused features did receive smaller weights in general, they were not completely excluded. To evaluate the accuracy of feature selection, we quantitatively measured the discrepancy between the estimated models and the true model by computing the mean squared error (MSE) $trace((\mathbf{A} - \mathbf{A}_{est})^\top(\mathbf{A} - \mathbf{A}_{est}))$ where $\mathbf{A}_{est}$ was the matrix estimated by a method. We compared MSE values of individual mothods.

On the *C1* data, MMTFL(2,1) learned better combination weights (darker areas) for the relevant features than MMTFL(1,1). MMTFL(1,1) appeared to be unnecessarily too



(a) On *C1* data (b) On *C2* data

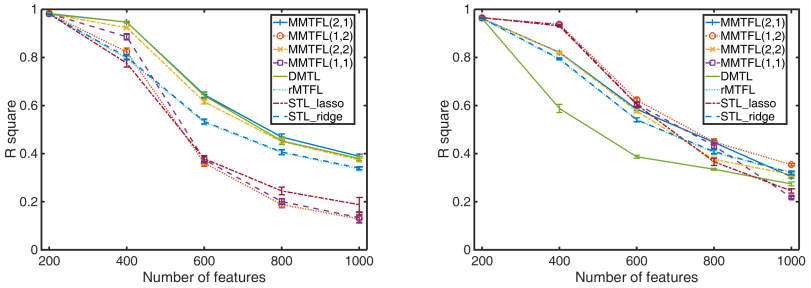
Figure 2: The regression performance of different models on synthetic data *C1* and *C2* when the number of tasks is varied.

sparse because the useful features received much smaller weights than needed (lighter than the true model). The smallest MSE was achieved by MMTFL(2,1) with a value of 0.1, and the second best model, MMTFL(1,1), had MSE = 0.2 whereas the rMTFL model had the largest MSE = 0.25.

On the *C2* data, MMTFL(1,2) learned a model that was most comparable to the true model. Both MMTFL(1,2) and MMTFL(1,1) eliminated well the irrelevant features. However, if we compared the two rows corresponding to these two models in Figure 1, we could see that MMTFL(1,1) broke the staircases in several places (e.g., towards the lower right and the up left corners) by excluding more features than necessary. Note that the feature sharing patterns (particularly in synthetic data *C2*) may not be revealed by the recent methods on clustered multitask learning that cluster tasks into groups (Kang et al., 2011; Jacob et al., 2008; Zhou et al., 2011) because no cluster structure is present in Figure 1b. Rather, the sharing pattern in Figure 1b is actually in between the consecutive groups of tasks. MMTFL(1,2) had the smallest MSE = 0.05, which was smaller than that of the second best model MMTFL(1,1) by 0.025. DMTL received the largest MSE = 0.19.

Figure 1c shows the results on the *C3* data set. MMTFL(2,1), MMTFL(1,2), and MMTFL(1,1) imposed excessive sparsity on the parameter matrix, which removed some useful features. The other three models, DMTL, rMTFL and MMTFL(2,2), produced similar parameter matrices, but rMTFL was originally designed to detect outlier tasks and thus was more favorable for this data set. The rMTFL model obtained the smallest MSE (0.03), and MMTFL(2,2) had a similar performance (MSE=0.04), which was the same as that of DMTL. On this data set, MMTFL(1,1) got the largest MSE (0.09). These results bring out an interesting observation that for the MTL scenarios that have outlier tasks but relevant tasks share the same set of features, MMTFL(2,2) (which corresponds to the very early joint regularized method using the $\ell_{1,2}$ matrix norm) is most suitable among the multiplicative MTFL methods.

Figure 2 compares the performance of different methods when we vary the number of tasks in the *C1* and *C2* categories. On each data set, we used 33% of the data in training



(a) On *C1* data (b) On *C2* data

Figure 3: The regression performance of different models on synthetic data *C1* and *C2* when the number of features is varied.

with 15 trials, and reported here the average R^2 values and standard deviation bars. From Figure 2, MMTFL(2,1) constantly performed the best among all methods on the *C1* data sets (but not in the single task learning) whereas MMTFL(1,2) outperformed the other models on the *C2* data sets. We also observed that on *C2* data, MMTFL(1,1) obtained very similar performance to that of MMTFL(1,2) after the number of tasks reached 50. On this data category, almost all methods reached a stable level of accuracy after the number of tasks reached 50 except DMTL. DMTL continued to gain knowledge from more relevant tasks until it reached 500 tasks but it produced the lowest R^2 values among all methods. Overall, the results indicate that with the fixed dimension and sample size, when the number of tasks reaches a certain level, the transferable knowledge learned from the tasks can be saturated for a specific feature sharing structure. On *C1* data, we observe that the performance was not always monotonically improved or non-degraded (for all methods) when more tasks were included, which may indicate that when an unnecessarily large number of tasks was used, it could add more uncertainty to the learning process.

Figure 3 compares the performance of different methods when we vary the number of features in the *C1* and *C2* categories. Obviously, when the problem dimension was higher, the learning problem became more difficult (especially when the number of tasks and sample size remained the same). All methods dropped their performance substantially with increasing numbers of features although MMTFL(2,1) and MMTFL(1,2) still outperformed other methods, respectively, on the *C1* and *C2* data sets. This figure also shows that MMTFL(1,1) performed well on the *C2* data sets but much worse on the *C1* data sets. DMTL produced good performance, close to that of MMTFL(2,1), on the *C1* data sets.

6.2 Experiments with Benchmark Data

Extensive empirical studies were conducted on benchmark data sets where we tested the proposed multiplicative MTFL algorithms on ten real-world data sets. Among these data sets, three were for regression experiments and all others were for classification experiments. Characteristics of these data sets are summarized in the next section.

6.2.1 Benchmark Datasets

Sarcos (Argyriou et al., 2007): Sarcos data were collected for a robotics problem of learning the inverse dynamics of a 7 degrees-of-freedom SARCOS anthropomorphic robot arm. Each observation has 21 features corresponding to 7 joint positions and their velocities and accelerations. We needed to map from the 21-dimensional input space to 7 joint torques, which corresponded to 7 tasks. For each task, we randomly selected 2000 cases for training and the remaining 5291 cases for test. Readers can consult with http://www.gaussianprocess.org/gpml/data/ for more details.

CollegeDrinking (Bi et al., 2013): The college drinking data were collected in order to identify alcohol use patterns of college students and the risk factors associated with the binge drinking. The data set contained daily responses from 100 college students to a survey questionnaire measuring various daily measures, such as drinking expectation, negative affects, and level of stress, every day in a 30 day period. The goal was to predict the amount of nighttime drinks based on 51 daily measures for each student, corresponding to 100 regression tasks. Because there were only 30 records for each person, we used 66%, 75% and 80% of the records to form the training set, and the rest for test.

QSAR (Ma et al., 2015): The quantitative structure-activity relationship (QSAR) methods are commonly used to predict biological activities of chemical compounds in the field of drug discovery. The data sets we used were collected from three different types of drug activities, including binding to cannabinoid receptor 1 (CB1), inhibition of dipeptidyl peptidase 4 (DPP4) and time dependent 3A4 inhibitions (TDI). For each activity, there were 200 molecule examples represented by 2618 features. Three regression models were constructed to simultaneously predict the targets $-log(\text{IC50})$) of the CB1, DPP4 and TDI effectiveness based on the molecular features.

C.M.S.C. (Lucas et al., 2013): The Climate Model Simulation Crashes (C.M.S.C.) data set contained records of simulated crashes encountered during climate model uncertainty quantification ensembles. The data set comprised 3 tasks. There were 180 examples for each task. Each example was represented by an 18-dimensional feature vector. Each task is formed by a binary classification problem, which was to predict simulation outcomes (either fail or succeed) from the input parameter values for a climate model.

Landmine (Xue et al., 2007): The original Landmine data contained 29 data sets where sets 1-15 corresponded to the geographical regions that were highly foliated and sets 16-29 corresponded to the regions with bare earth or desert. Each data set could be used to build a binary classifier. We used the data sets 1-10 and 16-25 to form 20 tasks where each example was represented by 9 features extracted from radar images. The number of examples varied between individual tasks ranging from 445 to 690.

Alphadigits (Maurer et al., 2013): This data set was composed of binary 20×16 images of the 10 digits and capital letters. We used all the images of digits to form 10 binary classification tasks. For each digit, there were 39 images in this data set. We labeled the images of a single digit as positive examples, and randomly selected other 39 images from other digits and labeled them as negative examples. All the pixels were concatenated to form a 320-dimensional feature vector for each image.

Underwatermine (Liu et al., 2009b): This data set was originally used in the underwater mine classification problem that aimed to detect mines from non-mines based on the

synthetic-aperture sonar images. The data set consisted of 8 tasks with sample sizes ranging from 756 to 3562 for each task, and each task was a binary classification problem. Each example was represented by 13 features.

Animal recognition (Kang et al., 2011): This data set consisted of images from 20 animal classes. Each image was originally represented by 2000 features extracted using the bag of word descriptors from the Scale-invariant Feature Transform (SIFT), and then the dimensionality was reduced to 202 by a principal component analysis, retaining 95% of the data variance. For each animal class, there were 100 images. We formed 20 binary classification tasks where for each task, 100 positive examples were from a specific animal class and 100 negative examples were randomly sampled from other classes.

HWMA_base and HWMA_peak (Qazi et al., 2007; Bi and Wang, 2015): The heart wall motion abnormality (HWMA) detection data set was used to analyze and predict if a heart had abnormal motion based on the image features extracted from stress test echocardiographs of 220 patients. The images were taken at the base dose and also peak dose of stress contrast. The wall of left ventricle was medically segmented into 16 segments, and 25 features were extracted from each segment. Every segment of every heart case was annotated by radiologists in terms of normal or abnormal motions. Thus, there were 16 binary classification tasks, each corresponding to one of the 16 heart segments, and each task comprised 220 examples.

6.2.2 Performance on real world data sets

Three real-world data sets, the Sarcos, CollegeDrinking and QSAR, were used in regression experiments. The performance of the different methods is summarized in Table 4 depicting the R^2 values averaged over the 15 re-partitions in each trial. MMTFL(2,1) achieved the best R^2 values on the Sarcos data set (in all of the 3 trials) and the CollegeDrinking data set (in 2 of the 3 trials). The Sarcos data appeared to be in favor of denser models given MMTFL(2,2) also performed reasonably well on this data set. MMTFL(1,2) models achieved the best performance on the QSAR data set consistently across all the 3 trial settings. On this data set, it was obvious that MMTFL(1,2) was more suitable, which indicated that most of the 2618 features were useful for some tasks, but the tasks shared few features between each other. The difference between the proposed models and the additively decomposed models ranged from 1 % to 10%, and most importantly, the trend was consistent for the proposed models to outperform on these data sets.

The other seven real-world data sets were used in classification experiments. Table 5 summarizes the results where the F1 scores were averaged across the 15 random splits in each trial together with standard derivations. MMTFL(2,1) models achieved consistently the best performance on the C.S.M.C. and Landmine data sets in comparison with other models. In particular, we observed both MMTFL(1,1) and the MMTFL(2,1) models produced the best F1 scores in the trial with 33% training split whereas MMTFL(2,1) outperformed all other models in the trial with other partition ratios. These two data sets may prefer across-task sparse models, indicating that many irrelevant features may exist in the data. For the remaining five data sets used in classification experiments, MMTFL(1,2) models showed generally better performance than all other models. The difference between the best model and other MMTFL models could reach 4% to 8%.

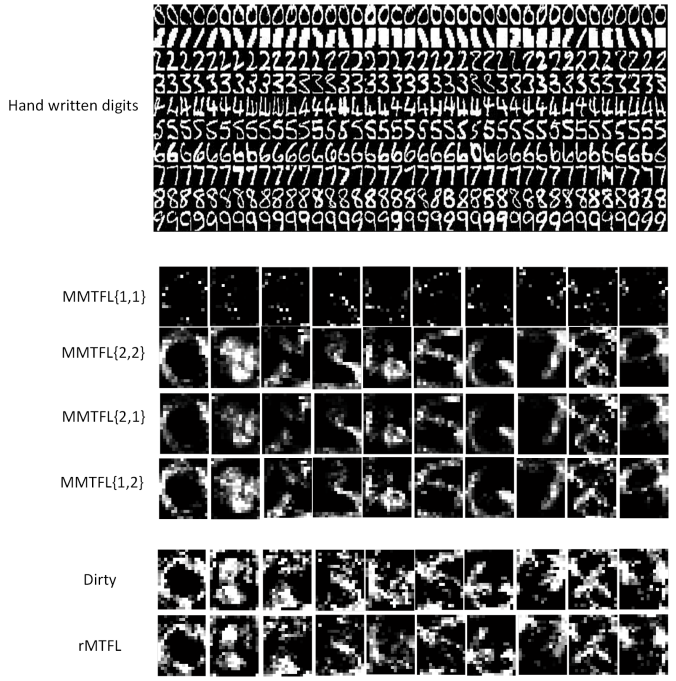


Figure 4: The models constructed by MTFL methods for individual digits using the pixels in hand written digit images. The models can be re-organized back into images. The pixel in the above images ranges from 0 to 1. The lighter, the closer to 1 for lucid illustration.

In particular, for the Alphadigits data set, we used the raw pixels of the hand written digits as the features to build models. Each task aimed to learn a linear model in the original pixel dimensions to distinguish a digit from other nine digits. Thus, each model, or equivalently the weight vector of the linear model, could be re-shaped back into an image. Figure 4 compares the constructed models for each digit by each of the six MTFL methods. In the top of Figure 4, we illustrate some sample images. In the middle, we show the results by the four MMTFL methods whereas at the bottom we include the constructed additively decomposed models. Clearly, the additively decomposed models were much noisier and selected many undesirable features. Among the multiplicative MTFL methods, MMTFL(1,1) models were too sparse to trace out the digits. Overall, MMTFL(1,2) models were the closest to the shapes of the different digits. If we compare MMTFL(2,1) and (1,2), MMTFL(2,1) excluded too many features from all digits. It indicated that most of the pixels were useful for predicting a digit but not many pixels were shared by multiple digits.

Table 4: Comparison of the R^2 values of the different MTFL methods on the benchmark regression data sets.

Dataset		STL_lasso	STL_ridge	DMTL	rMTFL	MMTFL(2,2)	MMTFL(1,1)	MMTFL(2,1)	MMTFL(1,2)
SARCOS	25%	0.75±0.02	0.78±0.02	0.86± 0	0.86±0	**0.89± 0**	0.88± 0	**0.89±0.01**	0.86±0.01
	33%	0.78±0.01	0.78±0.02	0.81±0.11	0.81±0.1	**0.90± 0**	0.89± 0	**0.90±0.01**	0.82±0.01
	50%	0.82±0.01	0.83±0.06	0.87± 0.1	0.87±0.1	0.89± 0	0.90± 0.01	**0.91±0.01**	0.86±0.01
CollegeDrinking	66%	0.15±0.05	0.09±0.02	0.11± 0.07	0.22±0.02	**0.23±0.04**	0.15± 0.02	0.21±0.01	0.19±0.04
	75%	0.18±0.05	0.15±0.02	0.18± 0.08	0.18±0.05	0.14±0.05	0.25± 0.08	**0.27±0.05**	0.21±0.08
	80%	0.11±0.06	0.09±0.04	0.19± 0.06	0.16±0.06	0.19±0.05	0.15± 0.04	**0.21±0.04**	0.15±0.04
QSAR	25%	0.39±0.03	0.36±0.01	0.39±0.01	0.39±0.01	0.34±0.02	0.38±0.03	0.38±0.03	**0.40±0.03**
	33%	0.39±0.04	0.39±0.04	0.40±0.04	0.39±0.04	0.37±0.04	0.41±0.04	0.40±0.05	**0.43±0.03**
	50%	0.44±0.06	0.41±0.03	0.44±0.03	0.44±0.04	0.40±0.06	0.44±0.04	0.45±0.05	**0.48±0.04**

Table 5: Comparison of the F1 scores of the different MTFL methods on the benchmark classification data sets.

Dataset		STL_lasso	STL_ridge	DMTL	rMTFL	MMTFL(2,2)	MMTFL(1,1)	MMTFL(2,1)	MMTFL(1,2)
C.S.M.C.	25%	0.30±0.04	0.31±0.03	0.36±0	0.26±0.01	0.40±0	0.39±0.01	**0.43±0**	0.39±0
	33%	0.37±0.02	0.37±0.06	0.37±0.01	0.39±0.01	0.40±0.01	**0.45±0.01**	**0.45±0.01**	0.39±0.01
	50%	0.40±0.02	0.46±0.01	0.37±0.01	0.44±0.01	0.48±0.01	0.46±0.01	**0.49±0.01**	0.46±0
Landmine	25%	0.14±0.01	0.06±0.01	0.04±0.01	0.06±0	0.1±0.01	**0.17±0.01**	**0.17±0.01**	**0.17±0.01**
	33%	0.13±0.01	0.15±0.01	0.16±0.01	0.16±0.01	0.1±0.01	**0.18±0.01**	**0.18±0.01**	0.12±0.01
	50%	0.15±0.01	0.15±0.01	0.18±0.01	0.21±0.01	0.11±0.01	0.19±0.01	**0.23±0.01**	0.15±0.01
Alphadigits	25%	0.86±0.01	0.77±0.01	0.86±0.01	0.86±0	0.85±0.01	0.84±0.01	0.78±0.01	**0.88±0.01**
	33%	0.90±0.01	0.87±0.01	0.90±0.01	0.90±0.01	0.87±0.01	0.85±0.01	0.81±0.01	**0.91±0.01**
	50%	0.90±0.01	0.87±0.01	0.91±0.01	0.92±0.01	0.91±0.01	0.90±0.01	0.85±0.01	**0.93±0.01**
Underwatermine	25%	0.24±0.01	0.27±0.02	0.21±0.01	0.27±0	**0.29±0.01**	0.24±0.01	0.25±0.01	0.27±0.01
	33%	0.25±0.01	0.31±0.02	0.31±0.01	0.29±0.01	0.3±0.01	0.3±0.01	0.27±0.01	**0.32±0.01**
	50%	0.27±0.01	0.28±0.01	0.32±0.01	0.34±0.01	0.32±0.01	0.34±0.01	0.35±0.01	**0.37±0.01**
Animal	25%	0.63±0.01	0.63±0.01	0.65±0.01	0.65±0	**0.66±0.01**	0.64±0.01	0.63±0.01	**0.66±0.01**
	33%	0.66±0.01	0.67±0.01	0.66±0.01	0.65±0.01	0.67±0.01	0.66±0.01	0.66±0.01	**0.68±0.01**
	50%	0.67±0	0.68±0.01	0.68±0.01	0.66±0.01	**0.69±0.01**	0.67±0.01	0.68±0.01	**0.69±0.01**
HWMA_base	25%	0.35±0.01	0.37±0.01	0.42±0.01	0.4±0	**0.46±0.01**	0.36±0.01	0.44±0.01	0.45±0.01
	33%	0.38±0.01	0.34±0	0.45±0.01	0.47±0.01	046±0.01	0.44±0.01	0.47±0.01	**0.49±0.01**
	50%	0.44±0.01	0.45±0.01	0.48±0.01	0.48±0.01	0.51±0.01	0.47±0.01	0.47±0.01	**0.55±0.01**
HWMA_peak	25%	0.41±0	0.48±0.01	0.47±0.01	0.47±0	0.53±0.01	0.53±0.01	0.51±0.01	**0.54±0.01**
	33%	0.42±0.01	0.47±0.01	0.47±0.01	0.5±0.01	**0.56±0.01**	0.55±0.01	0.52±0.01	0.53±0.01
	50%	0.44±0.02	0.50±0.02	0.49±0.01	0.51±0.01	0.56±0.01	0.56±0.01	0.55±0.01	**0.58±0.01**

7. Conclusion

In this paper, we have studied a general framework of multiplicative multitask feature learning. By decomposing the model parameter of each task into a product of two components: the across-task feature indicator and task-specific parameters, and applying different regularizers to the two components, we can select features for individual tasks and also search for the shared features among tasks. We have examined the theoretical properties of this framework when different regularizers are applied and found that this family of methods creates models equivalent to those of the joint regularized multitask learning methods but with a more general form of regularization. Further, we show that this family consists of some convex and some non-convex formulations and specify the conditions to obtain convexity. An analytical formula is derived for the across-task component as related to the task-specific component, which sheds light on the different shrinkage effects in the various regularizers. An efficient algorithm is derived to solve the entire family of methods and also tested in our experiments for some chosen parameter values. Empirical results on synthetic data clearly show that there may not be a particular choice of regularizers that is universally better than other choices. We empirically show a few feature sharing patterns that are in favor of the two newly-proposed choices of regularizers. The extensive experimental results on real-world benchmark data sets also confirm the observation and demonstrate the advantages of the proposed two formulations over several existing methods.

Acknowledgments

This work was supported by NSF grants IIS-1320586, DBI-1356655 and the NIH grant R01DA037349. Jinbo Bi was also supported by NSF grants CCF-1514357, IIS-1407205 and IIS-1447711. Correspondence should be addressed to Jinbo Bi (jinbo@engr.uconn.edu).

References

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, December 2005.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 41–48. 2007.

Jinbo Bi and Xin Wang. Learning classifiers from dual annotation ambiguity via a minmax framework. *Neurocomputing*, 151, Part 2(0):891 – 904, 2015.

Jinbo Bi, Tao Xiong, Shipeng Yu, Murat Dundar, and R. Bharat Rao. An improved multi-task learning approach with applications in medical diagnosis. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, pages 117–132, 2008.

Jinbo Bi, Jiangwen Sun, Yu Wu, Howard Tennen, and Stephen Armeli. A machine learning approach to college drinking prediction and risk factor identification. *ACM Transactions on Intelligent Systems Technology*, 4:72:1–72:24, 2013.

Edwin Bonilla, Kian Ming Chai, and Christopher Williams. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pages 153–160, 2007.

Jianhui Chen, Jiayu Zhou, and Jieping Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 42–50, 2011.

Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 109–117, 2004.

Pinghua Gong, Jieping Ye, and Changshui Zhang. Robust multi-task feature learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 895–903, 2012.

Pinghua Gong, Jieping Ye, and Changshui Zhang. Multi-stage multi-task feature learning. *The Journal of Machine Learning Research*, 14(1):2979–3010, 2013.

Irwin R Goodman and Samuel Kotz. Multivariate θ-generalized normal distributions. *Journal of Multivariate Analysis*, 3(2):204–219, 1973.

William H. Greene. *Econometric Analysis*. Prentice Hall, fifth edition, 2002.

Shengbo Guo, Onno Zoeter, and Cédric Archambeau. Sparse bayesian multi-task learning. In *Advances in Neural Information Processing Systems*, pages 1755–1763, 2011.

Laurent Jacob, Bach Francis, and Jean-Philippe Vert. Clustered multi-task learning: a convex formulation. In *Advances in Neural Information Processing Systems*, pages 745–752, 2008.

Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep K Ravikumar. A dirty model for multi-task learning. In *Advances in Neural Information Processing Systems*, pages 964–972, 2010.

Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of the International Conference on Machine Learning*, pages 521–528, 2011.

Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the International Conference on Machine Learning*, pages 1383–1390, 2012.

Seunghak Lee, Jun Zhu, and Eric Xing. Adaptive multi-task lasso: with application to eQTL detection. In *Advances in Neural Information Processing Systems*, pages 1306–1314. 2010.

Su-In Lee, Vassil Chatalbashev, David Vickrey, and Daphne Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th International Conference on Machine Learning*, pages 489–496, New York, NY, USA, 2007.

Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient $\ell_{1,2}$-norm minimization. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 339–348, 2009a.

Qiuhua Liu, Xuejun Liao, Hui Li, Jason R Stack, and Lawrence Carin. Semisupervised multitask learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(6):1074–1086, 2009b.

Aurelie Lozano and Grzegorz Swirszcz. Multi-level lasso for sparse multi-task regression. In *Proceedings of the International Conference on Machine Learning*, pages 361–368, 2012.

D. D. Lucas, R. Klein, J. Tannahill, D. Ivanova, S. Brandon, D. Domyancic, and Y. Zhang. Failure analysis of parameter-induced simulation crashes in climate models. *Geoscientific Model Development Discussions*, 6(1):585–623, 2013.

Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structureactivity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 343–351, 2013.

L. Meier, S. v. d. Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B*, 70 Part 1:53–71, 2008.

Charles A Micchelli, Jean M Morales, and Massimiliano Pontil. Regularizers for structured sparsity. *Advances in Computational Mathematics*, 38(3):455–489, 2013.

Guillaume Obozinski and Ben Taskar. Multi-task feature selection. In *Technical report, Statistics Department, UC Berkeley*, 2006.

Alexandre Passos, Piyush Rai, Jacques Wainer, and Hal Daume III. Flexible modeling of latent task structures in multitask learning. In *Proceedings of the International Conference on Machine Learning*, pages 1103–1110, 2012.

M. Qazi, G. Fung, S. Krishnan, J. Bi, B. Rao, and A. Katz. Automated heart abnormality detection using sparse linear classifiers. *IEEE Engineering in Medicine and Biology*, 26 (2):56–63, 2007.

Ariadna Quattoni, Xavier Carreras, Michael Collins, and Trevor Darrell. An efficient projection for l1,infinity regularization. In *Proceedings of the International Conference on Machine Learning*, pages 108–115, 2009.

Piyush Rai and Hal Daume. Infinite predictor subspace models for multitask learning. In *International Conference on Artificial Intelligence and Statistics*, pages 613–620, 2010.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

B. A. Turlach, W. N. Wenables, and S. J. Wright. Simultaneous variable selection. *Technometrics*, 47(3):349–363, 2005.

T. Xiong, J. Bi, B. Rao, and V. Cherkassky. Probabilistic joint feature selection for multi-task learning. In *Proceedings of SIAM International Conference on Data Mining*, pages 69–76, 2006.

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

Ming Yang, Yingming Li, and Zhongfei Mark Zhang. Multi-task learning with gaussian matrix generalized inverse gaussian model. In *Proceedings of the 30th International Conference on Machine Learning*, pages 423–431, 2013.

Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22Nd International Conference on Machine Learning*, pages 1012–1019, New York, NY, USA, 2005.

Shipeng Yu, Volker Tresp, and Kai Yu. Robust multi-task learning with t-processes. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1103–1110, 2007.

Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationship in multi-task learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 733–742, 2010.

Yu Zhang, Dit-Yan Yeung, and Qian Xu. Probabilistic multi-task feature selection. In *Advances in Neural Information Processing Systems*, pages 2559–2567, 2010.

Peng Zhao, Guilherme Rocha, and Bin Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, pages 3468–3497, 2009.

Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. In *Advances in Neural Information Processing Systems*, pages 702–710, 2011.

Y. Zhou, R. Jin, and S. C.H. Hoi. Exclusive lasso for multi-task feature selection. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 988–995, 2010.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 5/15; Revised 10/15; Published 4/16

The Benefit of Multitask Representation Learning

Andreas Maurer AM@ANDREAS-MAURER.EU
Adalbertstrasse 55, D-80799 München, Germany
Massimiliano Pontil MASSIMILIANO.PONTIL@IIT.IT
Istituto Italiano di Tecnologia, 16163, Genoa, Italy
Department of Computer Science, University College London, WC1E 6BT, UK
Bernardino Romera-Paredes BERNARD@ROBOTS.OX.AC.UK
Department of Engineering Science, University of Oxford, OX1 3PJ, UK

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

We discuss a general method to learn data representations from multiple tasks. We provide a justification for this method in both settings of multitask learning and learning-to-learn. The method is illustrated in detail in the special case of linear feature learning. Conditions on the theoretical advantage offered by multitask representation learning over independent task learning are established. In particular, focusing on the important example of half-space learning, we derive the regime in which multitask representation learning is beneficial over independent task learning, as a function of the sample size, the number of tasks and the intrinsic data dimensionality. Other potential applications of our results include multitask feature learning in reproducing kernel Hilbert spaces and multilayer, deep networks.

Keywords: learning-to-learn, multitask learning, representation learning, statistical learning theory, transfer learning

1. Introduction

Multitask learning (MTL) can be characterized as the problem of learning multiple tasks *jointly*, as opposed to learning each task in isolation. This problem is becoming increasingly important due to its relevance in many applications, ranging from modelling users' preferences for products, to multiple object classification in computer vision, to patient healthcare data analysis in health informatics, to mention but a few. Multitask learning algorithms which exploit structure and similarities across different learning problems have been studied by the machine learning community since the mid 90's, initially in connection to neural network models (see Baxter, 2000; Caruana, 1998; Thrun and Pratt, 1998, and reference therein). More recent approaches have been based on kernel methods (Evgeniou et al., 2005), structured sparsity and convex optimization (Argyriou et al., 2008), among others.

Closely related to multitask learning but more challenging is the problem of *learning-to-learn* (LTL), namely learning to perform a new task by exploiting knowledge acquired when solving previous tasks. Arguably, a solution to this problem would have major impact in Artificial Intelligence as we could build machines which learn from experience to perform new tasks, similar to what we observe in human behavior.

An influential line of research on multitask and transfer learning is based on the idea that the tasks are related by means of a common low dimensional representation, which is learned jointly with the tasks' parameters. This approach was first advocated in (Baxter, 2000; Caruana, 1998; Thrun and Pratt, 1998) and more recently reconsidered in (Argyriou et al., 2008) from the perspective of convex optimization and sparsity regularization. Representation learning is also a key problem in AI, and in the past years there has been much renewed interest in learning nonlinear hierarchical representations from multiple tasks using multilayer, deep networks. Researchers have shown improved results in a number of empirical domains; the case of computer vision is perhaps most remarkable, (see e.g. Girshick et al., 2014, and references therein). This success has increased interest in multitask representation learning (MTRL) as it is a core component of deep networks. Still, the understanding of why this methodology works remains largely unexplored.

In this paper we analyze a general method for MTRL and discuss its potential advantage in both the MTL setting, where the learned representation is applied to the same tasks used during training, and in the domain of LTL, where the representation is applied to new tasks. We derive upper bounds on the error of these methods and quantify their advantage over independent task learning. When the original data representation is high dimensional and the number of examples provided to solve a regression or classification problem is limited, any learning algorithm which does not use any sort of prior knowledge will perform poorly because there is not enough data to reliably estimate the model parameters. We make this statement precise by considering the example of half space learning.

1.1 Previous Work

Many papers have proposed multitask learning methods and studied their applications to specific problems (see Ando and Zhang, 2005; Argyriou et al., 2008; Baxter, 2000; Ben-David and Schuller, 2003; Caruana, 1998; Cavallanti et al., 2010; Kuzborskij and Orabona, 2013; Maurer et al., 2013; Pentina and Lampert, 2014; Widmer et al., 2013, and references therein). There is a vast literature on these subjects and the list of papers provided here is necessarily incomplete.

Despite the considerable success of multitask learning and in particular multitask representation learning there are only few theoretical investigations (Ando and Zhang, 2005; Baxter, 2000; Ben-David and Schuller, 2003). Other statistical learning bounds are restricted to linear multitask learning such as (Cavallanti et al., 2010; Lounici et al., 2011; Maurer, 2006a,a).

Learning-to-learn (also called inductive bias learning or transfer learning) has been proposed by Thrun and Pratt (1998) and theoretically studied by Baxter (2000) where an error analysis is provided, showing that a common representation which performs well on the training tasks will also generalize to new tasks obtained from the same "environment". More recent papers which present dimension independent bounds appear in Maurer (2006a,b); Maurer and Pontil (2013); Maurer et al. (2013); Pentina and Lampert (2014).

1.2 Our Contributions

There are two main contributions of this work. First we present bounds to both the MTL and LTL settings, which apply to a very general MTRL method. Our analysis goes well beyond linear representation learning considered in most previous works. It improves over the analysis by Baxter (2000) based on covering numbers. We use more recent techniques of empirical process theory to achieve bounds which are independent of the input dimension (hence also valid in reproducing kernel Hilbert spaces) and to avoid logarithmic factors. Furthermore our analysis can be made fully data dependent. When specialized to subspace learning (i.e. linear feature learning) we get best bounds valid for infinite dimensional input spaces.

As the second main contribution of this paper, we explain the advantage of MTRL in terms of specificity of feature maps and expose conditions when MTRL is beneficial or when it is not worth the effort. We further specialize our upper bounds to half-space learning (noiseless binary classification) and compare them to a general lower bound for learning isolated tasks. We observe that if the number of tasks grows then the performance of the method (both in the MTL and LTL setting) matches the performance of square norm regularization with best a priori known representation. This analysis highlights the advantage of multitask learning over learning the tasks independently. We also present numerical experiments for half-space learning, which indicate the good agreement between theory and experiments.

1.3 Organization

The paper is organized as follows. In Section 2, we introduce the problem and present our main results. In Section 3, we specialize these results to subspace learning and illustrate the role played by the data covariance matrices in our bounds. In Section 3.1 we further illustrate our results in the case of half-space learning, rigorously comparing our upper bounds to a general lower bound for orthogonal equivariant algorithms. In Section 4, we present the proof of our main results, developing in particular uniform bounds on the estimation error. Finally, in Section 5 we summarize our findings and suggest directions for future research.

2. Multitask Representation Learning

The set of possible observations is denoted by $\mathcal{Z} = (\mathcal{X}, \mathbb{R})$, where the members of $\mathcal{X}$ are interpreted as inputs and the members of $\mathbb{R}$ are interpreted as outputs, or labels. A learning task is modelled by a probability measure μ on $\mathcal{Z}$ where $\mu(x, y)$ is the probability to encounter the input-output pair $(x, y) \in \mathcal{Z}$ in the context of task μ. We want to learn how to predict outputs. If we predict y while the true output is y', we suffer a loss $\ell(y, y')$, where the loss function $\ell : \mathbb{R} \times \mathbb{R} \to [0, 1]$ is assumed to be 1-Lipschitz in the first argument for every value of the second argument. Different Lipschitz constants can be absorbed in the scaling of the predictors and different ranges than $[0, 1]$ can be handled by a simple scaling of our results.

If g is a real function defined on $\mathcal{X}$, then the values $g(x)$ can be interpreted as predictors and the expectation $\mathbb{E}_{(X,Y)\sim\mu}[\ell(g(X), Y)]$ is the risk associated with hypothesis g on the task μ.

Multitask learning simultaneously considers many tasks $\mu_1, \ldots, \mu_T$ and hopes to exploit some suspected common property of these tasks. For the purpose of this paper this property is the existence of a representation or common feature-map, which simultaneously simplifies the learning problem for most, or all of the tasks at hand. We consider predictors g which factorize

$$g = f \circ h,$$

where "$\circ$" stands for functional composition, that is, $(f \circ h)(x) = f(h(x))$, for every $x \in \mathcal{X}$. The function $h : \mathcal{X} \to \mathbb{R}^K$ is called the representation, or feature-map, and it is used across different tasks, while f is a function defined on $\mathbb{R}^K$, a predictor specialized to the task at hand. In the sequel K will always be the dimension of the representation space.

As usual in learning theory the functions $h : \mathcal{X} \to \mathbb{R}^K$ and $f : \mathbb{R}^K \to \mathbb{R}$ are chosen from respective hypothesis classes $\mathcal{H}$ and $\mathcal{F}$, which we refer to as the class of representations and the class of specialized predictors, respectively. These classes can be quite general, but we require that the functions in $\mathcal{F}$ have Lipschitz constant at most L, for some positive real number L.

The choice of representation and specialized predictors is based on the data observed for all the tasks. This data takes the form of a multi-sample $\bar{\mathbf{Z}} = (\mathbf{Z}_1, \ldots, \mathbf{Z}_T)$, with $\mathbf{Z}_t = (Z_{t1}, \ldots, Z_{tn}) \sim \mu_t^n$. Here and in the sequel an exponent on a measure indicates a product measure, so that μ_t^n is a measure on $\mathcal{Z}^n$ and $\mathbf{Z}_t$ is an iid sample of n random variables distributed as μ_t. We also write $Z_{ti} = (X_{ti}, Y_{ti})$, $\mathbf{Z}_t = (\mathbf{X}_t, \mathbf{Y}_t)$ and $\bar{\mathbf{Z}} = (\bar{\mathbf{X}}, \bar{\mathbf{Y}})$.

Multitask representation learning (MTRL) solves the optimization problem

$$\min\left\{\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\ell(f_t(h(X_{ti})), Y_{ti}) : h \in \mathcal{H},\ (f_1, \ldots, f_T) \in \mathcal{F}^T\right\}. \tag{1}$$

In this paper, we are not concerned with the algorithmics of this problem, but rather with the statistical properties of its solutions $\hat{h}$ and $\hat{f}_1, \ldots, \hat{f}_T$. Note that these are functional random variables in their dependence on $\bar{\mathbf{Z}}$.

We consider two possible applications of these solutions. One application, which we will refer to as multitask learning (MTL), retains both the representation $\hat{h}$ and the specializations $\hat{f}_1, \ldots, \hat{f}_T$ to be applied to the tasks at hand. The other, perhaps more important, application assumes that the tasks μ_t are related by a probabilistic law, called an *environment*, and keeps only the representation $\hat{h}$ to be used when specializing to new tasks obeying the same law. In this way the parametrization of a learning algorithm is learned, hence the name "learning-to-learn" (LTL).

We will give general statistical guarantees in both cases. Our bounds consist of three terms. The first term can be interpreted as the cost of estimating the representation h and decreases with the number T of tasks available for training. The second term corresponds to the cost of estimating task-specific predictors and decreases with the number n of training examples available for each task. The last term contains the confidence parameter and typically makes only a very small contribution.

It is not surprising that the complexity of the representation class $\mathcal{H}$ (first term in the bounds) plays a central role. We measure this complexity on the observed input data $\bar{\mathbf{X}} \in \mathcal{X}^{Tn}$. Define a random set $\mathcal{H}\left(\bar{\mathbf{X}}\right) \subseteq \mathbb{R}^{KTn}$ by

$$\mathcal{H}\left(\bar{\mathbf{X}}\right) = \left\{\left(h_k\left(X_{ti}\right)\right) : h \in \mathcal{H}\right\}.$$

The complexity measure relevant to estimation of the representation is the Gaussian average

$$G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right) = \mathbb{E}\left[\sup_{h\in\mathcal{H}} \sum_{kti} \gamma_{kti} h_k\left(X_{ti}\right) | X_{ti}\right], \tag{2}$$

where the γ_{kti} are independent standard normal variables. The Gaussian average is of order $\sqrt{nT}$ in T and n for many classes of interest. These include kernel machines with Lipschitz kernels (e.g. Gaussian RBF) and arbitrarily deep compositions thereof, see Maurer (2014) for a discussion. As we shall see, this increase of $O(\sqrt{nT})$ is compensated in our bounds and the cost of learning the representation vanishes in the multi-task limit $T \to \infty$.

The second term in the bounds is governed by the quantity

$$\sup_{h\in\mathcal{H}} \frac{1}{n\sqrt{T}} \left\|h\left(\bar{\mathbf{X}}\right)\right\| = \frac{1}{\sqrt{n}} \sup_{h\in\mathcal{H}} \sqrt{\frac{1}{nT} \sum_{kti} h_k\left(X_{ti}\right)^2} \tag{3}$$

or an equivalent distribution-dependent expression. If the feature-maps in $\mathcal{H}$ are very specific, in the sense that their components are appreciably different from zero only for very special data, the quantity in (3) can become much smaller than $1/\sqrt{n}$, a phenomenon which can give a considerable competitive edge to MTRL, in particular if the per-task sample size n is small. We will demonstrate this in Section 3, where we apply Theorems 1 and 2 to subspace-learning and show that the above quantity is related to the operator norm of the data covariance.

2.1 Bounding the Excess Task-averaged Risk (MTL)

If we make no further assumptions on the generation of the task-measures $\mu_1, \ldots, \mu_T$, a conceptually simple performance measure for a representation h and specialized predictors $f_1, \ldots, f_T$ is the task-averaged risk

$$\mathcal{E}_{\text{avg}}\left(h, f_1, \ldots, f_T\right) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{(X,Y)\sim\mu_t} \ell\left(f_t\left(h\left(X\right)\right), Y\right).$$

We want to compare this to the very best we can do using the classes $\mathcal{H}$ and $\mathcal{F}$, given complete knowledge of the distributions $\mu_1, \ldots, \mu_T$. The minimal risk is clearly

$$\mathcal{E}^*_{\text{avg}} = \min_{h\in\mathcal{H}, (f_1,\ldots,f_T)\in\mathcal{F}^T} \mathcal{E}_{\text{avg}}\left(h, f_1, \ldots, f_T\right).$$

It is a fundamental hope underlying our approach that the classes $\mathcal{H}$ and $\mathcal{F}$ are large enough for this quantity to be sufficiently small for practical purposes. We use the words "hope" and "belief" because an "assumption" would imply a statement to be used in analytical

reasoning. Instead our approach is agnostic, and our results are valid independent of the size of the minimal risk above.

Our first result bounds the excess average risk, which measures the difference between the task-averaged true risk of the solutions to (1) and the theoretical optimum above.

Theorem 1 *Let $\mu_1, \ldots, \mu_T$, $\mathcal{H}$ and $\mathcal{F}$ be as above, and assume $0 \in \mathcal{H}$ and $f\left(0\right) = 0$ for all $f \in \mathcal{F}$. Then for $\delta > 0$ with probability at least $1-\delta$ in the draw of $\bar{\mathbf{Z}} \sim \prod_{t=1}^{T} \mu_t^n$ we have that*

$$\mathcal{E}_{\text{avg}}(\hat{h}, \hat{f}_1, \ldots, \hat{f}_T) - \mathcal{E}^*_{\text{avg}} \leq \frac{c_1 L\ G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right)}{nT} + \frac{c_2 Q \sup_{h\in\mathcal{H}} \left\|h\left(\bar{\mathbf{X}}\right)\right\|}{n\sqrt{T}} + \sqrt{\frac{8\ln\left(4/\delta\right)}{nT}},$$

where c_1 and c_2 are universal constants, $G(\mathcal{H}(\bar{\mathbf{X}}))$ is the Gaussian average in Equation (2), and Q is the quantity

$$Q \equiv Q(\mathcal{F}) \sup_{y\neq y'\in\mathbb{R}^{Kn}} \frac{1}{\|y-y'\|} \mathbb{E} \sup_{f\in\mathcal{F}} \sum_{i=1}^{n} \gamma_i \left(f\left(y_i\right) - f\left(y_i'\right)\right). \tag{4}$$

Remarks:

1. The assumptions $0 \in \mathcal{H}$ and $f\left(0\right) = 0$ for all $f \in \mathcal{F}$ are made to give the result a simpler appearance. They are not essential, as the reader can verify from the proof.
2. If $G\left(\mathcal{H}\left(\bar{\mathbf{x}}\right)\right)$ is of order $\sqrt{nT}$ then the first term on the right hand side above is of order $1/\sqrt{Tn}$ and vanishes in the *multi-task limit* $T \to \infty$ even for small values of n.
3. For reasonable classes $\mathcal{F}$ one can find a bound on Q, which is independent of n, because the $\|y-y'\|$ in the denominator balances the Gaussian average depending on the class $\mathcal{F}$.
4. The quantity $\sup_h \left\|h\left(\bar{\mathbf{X}}\right)\right\|$ is of order $\sqrt{nT}$ whenever $\mathcal{H}$ is uniformly bounded, a crude bound being $\sqrt{nT} \sup_{h\in\mathcal{H}} \max_{ti} \|h\left(x_{ti}\right)\|$. The second term is thus typically of order $1/\sqrt{n}$. As explained in the discussion of Equation (3) above it can be very small if the representation components in $\mathcal{H}$ are very data-specific.

2.2 Bounding the Excess Risk for Learning-to-learn (LTL)

Now we consider the case where we only retain the representation $\hat{h}$ obtained from (1) and specialize it to future, hitherto unknown tasks. This is of course only possible, if there is some common law underlying the generation of tasks. Following Baxter (2000) we suppose that the tasks originate in a common environment η, which is by definition a probability measure on the set of probability measures on $\mathcal{Z}$. The draw of $\mu \sim \eta$ models the encounter of a learning task μ in the environment η.

The environment η induces a measure μ_η on $\mathcal{Z}$ by

$$\mu_\eta\left(A\right) = \mathbb{E}_{\mu\sim\eta}\left[\mu\left(A\right)\right] \text{ for } A \subseteq \mathcal{Z}.$$

This simple mixture plays an important role in the interpretation of our results.

The measure η also induces a measure ρ_η on $\mathcal{Z}^n$ which corresponds to the draw of an n-sample from a random task in the environment. To draw a sample $\mathbf{Z} \in \mathcal{Z}^n$ from ρ_η we first draw a task μ from η and then generate the sample $\mathbf{Z} = (Z_1, \ldots, Z_T)$ from n independent draws from μ. Formally

$$\rho_\eta(A) = \mathbb{E}_{\mu\sim\eta}\left[\mu^n(A)\right] \text{ for } A \subseteq \mathcal{Z}^n.$$

We assume that the tasks $\mu_1, \ldots, \mu_T$ are drawn independently from η and, consequently, that the multisample $\bar{\mathbf{Z}} = (\mathbf{Z}_1, \ldots, \mathbf{T}_T)$ is obtained in T independent draws from ρ_η, that is, $\bar{\mathbf{Z}} \sim \rho_\eta^T$.

The way we plan to use a representation $h \in \mathcal{H}$ on a new task $\mu \sim \eta$ is as follows: we draw a training sample $\mathbf{Z} = (Z_1, \ldots, Z_n)$ from μ^n and solve the optimization problem

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell\left(f\left(h\left(X_i\right)\right), Y_i\right).$$

Let $\hat{f}_{h,\mathbf{Z}}$ denote the minimizer and $m_{h,\mathbf{Z}}$ the corresponding minimum. We will then use the hypothesis $a(h)_{\mathbf{Z}} = \hat{f}_{h,\mathbf{Z}} \circ h = \hat{f}_{h,\mathbf{Z}}(h(\cdot))$ for the new task. In this way any representation $h \in \mathcal{H}$ parametrizes a learning algorithm, which is a function $a(h) : \mathcal{Z}^n \to \mathcal{F} \circ h$, defined, for every $\mathbf{Z} \in \mathcal{Z}^n$, as

$$a(h)_{\mathbf{Z}} = \hat{f}_{h,\mathbf{Z}} \circ h.$$

In this sense the problem of optimizing such a representation can properly be called "learning-to-learn". It can also be interpreted as "learning a hypothesis space" as in (Baxter, 2000), namely selecting a hypothesis space $\mathcal{F} \circ h$ from the collection of hypothesis spaces $\{\mathcal{F} \circ h : h \in \mathcal{H}\}$.

We can test the algorithm $a(h)$ on the environment η in the following way:

- we draw a task $\mu \sim \eta$,
- we draw a sample $\mathbf{Z} \in \mathcal{Z}^n$ from μ^n,
- we run the algorithm to obtain $a(h)_{\mathbf{Z}} = \hat{f}_{h,\mathbf{Z}} \circ h$,
- finally, we measure the loss of $a(h)_{\mathbf{Z}}$ on a random data-point $Z = (X, Y) \sim \mu$.

To define the risk $\mathcal{E}_\eta(h)$ associated with the algorithm $a(h)$ parametrized by h we just replace all random draws with corresponding expectations, so

$$\mathcal{E}_\eta(h) = \mathbb{E}_{\mu\sim\eta}\mathbb{E}_{\mathbf{Z}\sim\mu^n}\mathbb{E}_{(X,Y)\sim\mu}\left[\ell\left(a(h)_{\mathbf{Z}}(X), Y\right)\right].$$

The best value for any representation h in $a(h)$, given complete knowledge of the environment, is then

$$\min_{h\in\mathcal{H}} \mathcal{E}_\eta(h).$$

But, given complete knowledge of the environment, this is still not the best we can do using the classes $\mathcal{F}$ and $\mathcal{H}$, because for given μ and h we still use the expected performance

$\mathbb{E}_{\mathbf{Z}\sim\mu^n}\mathbb{E}_{Z\sim\mu}\,\ell\left(a(h)_{\mathbf{Z}}(X), Y\right)$ of the empirical risk minimization algorithm $a(h)$, instead of using knowledge of μ to replace it by $\min_{f\in\mathcal{F}} \mathbb{E}_{Z\sim\mu}\ell\left(f\left(h\left(X\right)\right), Y\right)$. The very best we can do is thus

$$\mathcal{E}_\eta^* = \min_{h\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\left[\min_{f\in\mathcal{F}} \mathbb{E}_{Z\sim\mu}\ell\left(f\left(h\left(X\right)\right), Y\right)\right].$$

The excess risk associated with any representation h is thus

$$\mathcal{E}_\eta(h) - \mathcal{E}_\eta^*.$$

We give the following bound for the excess risk associated with the representation $\hat{h}$ found as solution to the optimization problem (1).

Theorem 2 *Let η be an environment on $\mathcal{Z}$ and $\mathcal{H}$ and $\mathcal{F}$ as above. Then: (i) with probability at least $1-\delta$ in the draw of $\bar{\mathbf{Z}} \sim \rho_\eta^T$*

$$\mathcal{E}_\eta(\hat{h}) - \mathcal{E}_\eta^* \le \frac{\sqrt{2\pi}L\; G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right)}{T\sqrt{n}} + \sqrt{2\pi}Q' \sup_{h\in\mathcal{H}} \sqrt{\frac{\mathbb{E}_{(X,Y)\sim\mu_\eta}\left[\|h(X)\|^2\right]}{n}} + \sqrt{\frac{8\ln\left(4/\delta\right)}{T}},$$

and (ii) with the same probability

$$\mathcal{E}_\eta(\hat{h}) - \mathcal{E}_\eta^* \le \frac{\sqrt{2\pi}L\; G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right)}{T\sqrt{n}} + \frac{\sqrt{2\pi}Q'\left(1/T\right)\sum_t \sup_{h\in\mathcal{H}}\|h\left(\mathbf{X}_t\right)\|}{n} + 5\sqrt{\frac{\ln\left(8/\delta\right)}{T}},$$

where $\hat{h}$ is solution to the problem (1), $G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right)$ is the Gaussian average introduced in (2), and Q' is the quantity

$$Q' \equiv Q'(\mathcal{F}) = \sup_{y\in\mathbb{R}^{Kn}\setminus\{0\}} \frac{1}{\|y\|}\mathbb{E}\sup_{f\in\mathcal{F}}\sum_{i=1}^{n}\gamma_i f\left(y_i\right). \tag{5}$$

We make some remarks and comparison to the previous result.

1. The constants are now explicit and small. For Theorem 1, uniform estimation had to be controlled simultaneously in $\mathcal{H}$ and $\mathcal{F}$, while for LTL the problem can be more easily decoupled.
2. The first term is equivalent to the first term in Theorem 1 except for $\sqrt{n}$ replacing n in the denominator. It is therefore typically of order $1/\sqrt{T}$ instead of $1/\sqrt{nT}$. The different order is due to the estimation of a hitherto unknown task, for which the sample sizes are irrelevant. To understand this point assume the η has the property that every $\mu \sim \eta$ is deterministic, that is supported on a single point $z_\mu \in \mathcal{Z}$. Then clearly the sample size n is irrelevant, and the problem becomes equivalent to learning a single task with a sample of size T.

3. The quantity Q' is very much like the quantity Q in Equation (4), and it is uniformly bounded in n for the classes we consider. For linear classes $Q = Q'$.

4. The bound in part (i) is not fully data-dependent, but more convenient for our applications below. The quantity

$$\sup_{h\in\mathcal{H}} \sqrt{\mathbb{E}_{(X,Y)\sim\mu_\eta} \|h(X)\|^2} = \sup_{h\in\mathcal{H}} \sqrt{\sum_k \mathbb{E}_{(X,Y)\sim\mu_\eta} \left[h_k(X)^2\right]}$$

plays a similar role to (3), which is its empirical counterpart. Again, if the features h_k are very specific, as the dictionary atoms of the next section or the atoms in a radial basis function network, then the above quantity can become very small.

2.3 Comparison to Previous Bounds

The first and most important theoretical study of MTL and LTL was carried out by Baxter (2000), where sample complexity bounds are given for both settings. Instead of a feature map a hypothesis space is selected from a class of hypothesis spaces. Clearly every feature map with values in $\mathbb{R}^K$ defines a hypothesis space while the reverse is not true in general, so Baxter's setting is certainly more general than ours. On the other hand the practical applications discussed in (Baxter, 2000) can be cast in the language of feature learning.

To prove his sample complexity bounds Baxter uses covering numbers. This classical method requires to cover a (meta-)hypothesis space (or its evaluation on a sample) with a set of balls in an appropriately chosen metric. The uniform bound is then obtained as a union bound over the cover and bounds valid on the individual balls. The latter bounds follow from Lipschitz properties L of the loss function relative to the chosen metric. For a bound of order ϵ the radius of the balls has to be of order ϵ/L. This leads to covering numbers of order ϵ^{-d}, where d is some exponent (see the last inequalities in the proof of in (Baxter, 2000), and has the consequence that the dominant term in the bound has an additional factor of $\ln(1/\epsilon)$. This is manifest in Theorem 8, Theorem 12 and Corollary 13 in (Baxter, 2000) and constitutes an essential weakness of the method of covering numbers. For bounds on the excess risk it implies that the orders of $\sqrt{1/T}$ and $\sqrt{1/n}$ obtained from Rademacher or Gaussian complexities have to be replaced by $\sqrt{\ln(T)/T}$ and $\sqrt{\ln(n)/n}$.

Rademacher and Gaussian complexities make it easy to handle infinite dimensional input spaces (see our Theorems 4 and 5 below). They also lead to data dependent bounds, which allows us to explain the benefits of multi-task learning in terms of the spectrum of the data covariance operator and the effective input dimension. Bounding Gaussian complexities for linear classes is comparatively simple, see the proof of our Lemma 3. There is a wealth of recent literature on the Rademacher complexity of matrices with spectral regularizers (see e.g. Kakade et al., 2012; Maurer and Pontil, 2013, and references therein), while it is unclear to us how Baxter's method could be applied if the feature map is constrained by a bound on, say, the trace norm of the associated matrix. In the case of LTL, our approach also leads to explicit and small constant factors.

On the other hand it must be admitted, that it is relatively easy to obtain bounds (also provided by Baxter) of order $\ln(n)/n$ or $\ln(T)/T$ with covering numbers in the realizable case. Such bounds would be more difficult to obtain with our techniques.

The work of Ando and Zhang (2005) proposes the use of MTL as a method of semi-supervised learning through the creation of artificial tasks from unlabelled data, for example predicting concealed components of vectors. They analyze a specific algorithm where the class of feature maps can be seen as a linear mixture of a fixed feature map with subspace projections as discussed in our paper. The bounds given apply to the task-averaged risk and not to LTL. The analysis is based on Rademacher averages and is independent of the input dimension. The bound itself is expressed as an entropy integral as given by Dudley (see e.g. Van Der Vaart and Wellner, 1996) but it is not very explicit. In particular the role of the spectrum of the data covariance is not apparent.

3. Multi-task Subspace Learning

We illustrate the general results of the previous section with an important special case. We assume that the input space $\mathcal{X}$ is a bounded subset of a Hilbert space H, which could for example be a reproducing kernel Hilbert space. We denote by $\langle\cdot,\cdot\rangle$ the inner product in H and by $\|\cdot\|$ the induced norm. We hope that sufficiently good results can be obtained by predictors of the form g, where $g : H \to \mathbb{R}$ is linear with bounded norm. We also suspect that only few linear features in H suffice for most tasks, so that the vectors defining the hypotheses g can all be chosen from one and the same, albeit unknown, K-dimensional subspace M of H.

Consequently we will factorize predictors as $f \circ h$, where h is a partial isometry $h : H \to \mathbb{R}^K$ and f is a linear functional on $\mathbb{R}^K$ chosen from some ball of bounded radius. Specifically, we introduce the classes

$$\begin{aligned}\mathcal{H} &= \left\{H \ni x \mapsto (\langle d_1, x\rangle, \ldots, \langle d_K, x\rangle) \in \mathbb{R}^K \ : \ D = (d_1, \ldots, d_K) \in H^K \text{ orthonormal}\right\}\\ \mathcal{F} &= \left\{\mathbb{R}^K \ni y \mapsto \sum_k w_k y_k \in \mathbb{R} \ : \ \sum_k w_k^2 \le B^2\right\}.\end{aligned}$$

The D's appearing in the definition of $\mathcal{H}$ are also called dictionaries and the individual d_k are called atoms (see Maurer et al., 2013).

It does no harm to our analysis if we immediately generalize the class $\mathcal{H}$ so as to include certain two-layer neural networks by allowing a nonlinear activation function ϕ with Lipschitz constant L_ϕ and satisfying $\phi(0) = 0$, to be applied with each atom. We can also drop the condition of orthonormality and allow the atoms to trade some of their norms when needed. The enlarged class of representations is

$$\mathcal{H} = \left\{x \in H \mapsto (\phi(\langle d_1, x\rangle), \ldots, \phi(\langle d_K, x\rangle)) \in \mathbb{R}^K : d_1, \ldots, d_K \in H, \ \sum_k \|d_k\|^2 \le K\right\}.$$

The results can then be re-specialized to subspace learning by setting ϕ to the identity and L_ϕ to one.

When applied to subspace learning, our bounds are expressed in terms of covariances. If ν is a probability measure on H the corresponding covariance operator C_ν is defined by

$$\langle C_\nu v, w\rangle = \mathbb{E}_{X\sim\nu} \langle v, X\rangle \langle X, w\rangle \ \text{ for } v, w \in H.$$

For an environment η we denote the covariance operator corresponding to the data-marginal of the mixture measure μ_η simply by C.

If $\mathbf{x} = (x_1, \ldots, x_m) \in H^m$ we define the empirical covariance operator $\hat{C}(\mathbf{x})$ by

$$\left\langle \hat{C}(\mathbf{x})\, v, w \right\rangle = \frac{1}{m} \sum_i \langle v, x_i \rangle \langle x_i, w \rangle \text{ for } v, w \in H,$$

in particular

$$\left\langle \hat{C}\left(\bar{\mathbf{X}}\right) v, w \right\rangle = \frac{1}{nT} \sum_{ti} \langle v, X_{ti} \rangle \langle X_{ti}, w \rangle .$$

The following lemma establishes the necessary ingredients for the application of Theorems 1 and 2 to the case of subspace learning. Recall that if A is a selfadjoint positive linear operator on H, we denote by $\|A\|_\infty$ and $\|A\|_1$ its spectral and trace norms, respectively. They are defined as $\|A\|_\infty = \sup_{\|z\|\leq 1} \|Az\|$ and $\|A\|_1 = \sum_{i\in\mathbb{N}} \langle e_i, Ae_i \rangle$, where $\{e_i\}_{i\in\mathbb{N}}$ is an orthonormal basis in H. Recall also the definition of $Q(\mathcal{F})$ and $Q'(\mathcal{F})$ given in Equations (4) and (5), respectively.

Lemma 3 *Let $\bar{\mathbf{x}} = (x_{ti})$ be a $T \times n$ matrix with values in a Hilbert space and let ϕ, $\mathcal{H}$ and $\mathcal{F}$ be defined as above. Then*

(i) $G(\mathcal{H}(\bar{\mathbf{x}})) \leq L_\phi K \sqrt{nT \left\| \hat{C}(\bar{\mathbf{x}}) \right\|_1}$.

(ii) For every $h \in \mathcal{H}$, $\|h(\bar{\mathbf{x}})\| \leq L_\phi \sqrt{KnT \left\| \hat{C}(\bar{\mathbf{x}}) \right\|_\infty}$.

(iii) For an environment η *and every* $h \in \mathcal{H}$

$$\mathbb{E}_{(X,Y)\sim\mu_\eta} \|h(X)\|^2 \leq L_\phi^2 K \|C\|_\infty .$$

(iv) $L(\mathcal{F}) \leq B$.

(v) $Q(\mathcal{F}) \leq B$ *and* $Q'(\mathcal{F}) \leq B$.

Proof *(i)* Using the contraction lemma, Corollary 11, in the first inequality and Cauchy-Schwarz and Jensen's inequality in the second we get

$$\begin{aligned}
G(\mathcal{H}(\bar{\mathbf{x}})) &\leq L_\phi \mathbb{E} \sup_{d\in\mathcal{H}} \sum_{kti} \gamma_{kti} \langle d_k, x_{ti} \rangle \\
&= L_\phi \mathbb{E} \sup_{d\in\mathcal{H}} \sum_k \left\langle d_k, \sum_{ti} \gamma_{kti} x_{ti} \right\rangle \\
&\leq L_\phi \sqrt{K} \left(\sum_k \mathbb{E} \left\| \sum_{ti} \gamma_{ti} x_{ti} \right\|^2 \right)^{1/2} \\
&\leq L_\phi K \left(\sum_{ti} \|x_{ti}\|^2 \right)^{1/2} = L_\phi K \sqrt{nT \left\| \hat{C}(\mathbf{x}) \right\|_1}.
\end{aligned}$$

(ii) For any $D \in \mathcal{H}$

$$\begin{aligned}
\sum_{kti} \phi(\langle d_k, x_{ti} \rangle)^2 &\leq L_\phi^2 \sum_{kti} \langle d_k, x_{ti} \rangle^2 \\
&= L_\phi^2 \sum_k \|d_k\|^2 \sum_{ti} \left\langle \frac{d_k}{\|d_k\|}, x_{ti} \right\rangle^2 \\
&\leq L_\phi^2 K \sup_{v:\|v\|\leq 1} \sum_{ti} \langle v, x_{ti} \rangle^2 \\
&= L_\phi^2 K nT \left\| \hat{C}(\bar{\mathbf{x}}) \right\|_\infty ,
\end{aligned}$$

where we used $\phi(0) = 0$ in the first step.

(iii) Similarly, we have that

$$\begin{aligned}
\mathbb{E}_{(X,Y)\sim\mu_\eta} \sum_k \phi(\langle d_k, X \rangle)^2 &\leq L_\phi^2 \sum_k \|d_k\|^2 \mathbb{E}_{(X,Y)\sim\mu_\eta} \left\langle \frac{d_k}{\|d_k\|}, X \right\rangle^2 \\
&\leq L_\phi^2 K \sup_{\|v\|\leq 1} \mathbb{E}_{(X,Y)\sim\mu_\eta} \langle v, X \rangle^2 \\
&= L_\phi^2 K \|C\|_\infty .
\end{aligned}$$

(iv) Let $y, y' \in \mathbb{R}^K$. Then

$$\sup_{w\in\mathcal{F}} \left\{ \sum_k w_k y_k - \sum_k w_k y'_k \right\} \leq \left(\sum_k w_k^2 \right)^{1/2} \|y - y'\| \leq B \|y - y'\| ,$$

so $L \leq B$.

(v) Similarly, we have that

$$\begin{aligned}
\mathbb{E} \sup_{w\in\mathcal{F}} \sum_i \gamma_i \left(\sum_k w_k y_{ki} - \sum_k w_k y_{ki} \right) &= \mathbb{E} \sup_{w\in\mathcal{F}} \sum_k w_k \sum_i \gamma_i (y_{ki} - y_{ki}) \\
&\leq \sup_{w\in\mathcal{F}} \sqrt{\sum_k w_k^2 \sum_k \mathbb{E} \left(\sum_i \gamma_i (y_{ki} - y_{ki}) \right)^2} \\
&\leq B \sqrt{\sum_{ki} (y_{ki} - y_{ki})^2} = B \|y - y'\| ,
\end{aligned}$$

so $Q \leq B$. The same proof works for Q'. ■

Substitution in Theorem 1 immediately gives

Theorem 4 (subspace MTL) *With probability at least $1 - \delta$ in $\bar{\mathbf{X}}$ the excess risk is bounded by*

$$\mathcal{E}_{\text{avg}}(\hat{h}, \hat{f}_1, \ldots, \hat{f}_T) - \mathcal{E}^*_{\text{avg}} \leq c_1 L_\phi B K \sqrt{\frac{\left\| \hat{C}(\bar{\mathbf{X}}) \right\|_1}{nT}} + c_2 L_\phi B \sqrt{\frac{K \left\| \hat{C}(\bar{\mathbf{X}}) \right\|_\infty}{n}} + \sqrt{\frac{8 \ln(2/\delta)}{nT}}. \tag{6}$$

We remark that in the linear case the best competing bound for MTL, obtained by Maurer and Pontil (2013) from noncommutative Bernstein inequalities, is

$$2B\sqrt{\frac{K\left\|\hat{C}\left(\bar{\mathbf{X}}\right)\right\|_1 \ln\left(Tn\right)}{nT}} + B\sqrt{\frac{8K\left\|\hat{C}\left(\bar{\mathbf{X}}\right)\right\|_\infty}{n}} + \sqrt{\frac{8\ln\left(2/\delta\right)}{nT}}. \tag{7}$$

If we disregard the constants this is worse than the bound (6) whenever $K < \ln(Tn)$. Its approach to the multitask limit is slower ($\sqrt{\ln(T)/T}$ as opposed to $\sqrt{1/T}$), but of course it has the advantage of smaller constants. The methods used to obtain (7), however, break down for nonlinear dictionaries.

For the LTL setting, we use the distribution dependent bound, Theorem 2 (i), and obtain

Theorem 5 (subspace LTL) *With probability at least $1-\delta$ in $\bar{\mathbf{X}}$, the excess risk is bounded by*

$$\mathcal{E}_\eta(\hat{h}) - \mathcal{E}_\eta^* \leq \sqrt{2\pi}L_\phi B\left(\frac{K\sqrt{\left\|\hat{C}\left(\bar{\mathbf{X}}\right)\right\|_1}}{\sqrt{T}} + \sqrt{\frac{K\left\|C\right\|_\infty}{n}}\right) + \sqrt{\frac{8\ln\left(4/\delta\right)}{T}}.$$

The two most important common features of Theorems 4 and 5 are the decay to zero of the first term, as $T \to \infty$, and the occurrence of the operator norm of the empirical or true covariances in the second term. The first implies that for very large numbers of tasks the bounds are dominated by the second term.

To understand the second term we must first realize that the ratio of trace and operator norms of the true covariances can be interpreted as an effective dimension of the distribution. This is easily seen if the mixture of task-marginals is concentrated and uniform on a d-dimensional unit-sphere. In this case $\left\|C\right\|_1 = 1$ and by isotropy all eigenvalues are equal, so $\left\|C\right\|_\infty = 1/d$, whence $\left\|C\right\|_1 / \left\|C\right\|_\infty = d$. In such a case the second term in Theorem 5 above becomes

$$B\sqrt{\frac{K}{dn}}. \tag{8}$$

The appropriate standard bound for learning the tasks independently would be $B\sqrt{1/n}$ (see Bartlett and Mendelson, 2002). The ratio $\sqrt{K/d}$ of the two bounds in the multitask limit is the quotient of utilized information (the dimension of the representation space) to available information (the dimension of the data). This highlights the potential advantages of MTRL: if the data is already low-dimensional in the order of K then multi-task learning isn't worth the extra computational labour. If the data is high dimensional however, then multi-task learning may be superior.

The expression (8) above might suggest that there really is a benefit of high dimensions for learning-to-learn. This is of course not the case, because the regularizer B has to be chosen large, in fact proportional to $\sqrt{d}$ to allow a small empirical error. The correct interpretation of (8) is that the burden of high dimensions vanishes in the limit $T \to \infty$. In the next section we will explain this point in more detail.

3.1 Learning to Learn Half-spaces

In this section, we illustrate the benefit of MTRL over independent task learning (ITL) in the case of noiseless linear binary classification (or half-space learning). We compare our upper bounds for LTL to a general lower bound on the performance of ITL algorithms and quantify the parameter regimes where LTL is superior to ITL.

We assume that all the input marginals are given by the uniform distribution σ on the unit sphere $\mathcal{S}_d$ in $\mathbb{R}^d$, and the objective is for each task μ to classify membership in the half-space $\{x : \langle x, u_\mu\rangle > 0\}$ defined by a task-specific (unknown) unit vector u_μ. In the given environment all the vectors u_μ are assumed to lie in some (unknown) K-dimensional subspace M of $\mathbb{R}^d$. We are interested in the regime that

$$K \ll n \ll d$$

and T grows. This is the safe regime in which our upper bounds for MTL or LTL (cf. Theorems 4 and 5) are smaller than a uniform lower bound for independent task learning, which we discuss below. We need $n \ll d$ for the lower bound to be large and $K \ll n$ for the middle term in our upper bounds to be small. If T is large enough, the second term in our upper bounds dominates the first (task dependent) term. A safe choice is $T \gg K^2 d$, see Equation (9) below.

The 0-1-loss is unsuited for our bounds because it is not Lipschitz. Instead we will use the truncated hinge loss with unit margin given by $\ell(y', y) = \xi(y'y)$, where ξ is the real function

$$\xi(t) = \begin{cases} 1 & \text{if} \quad t \leq 0, \\ 1-t & \text{if} \quad 0 < t \leq 1, \\ 0 & \text{if} \quad 1 < t. \end{cases}$$

This loss is an upper bound of the 0-1-loss, so upper bounds for this loss function are also upper bounds for the classification error.

Let $\mathcal{H}$ and $\mathcal{F}$ be as given at the beginning of Section 3 in its linear variant, where $\mathcal{H}$ is defined by orthonormal dictionaries without activation functions. Thus, $\mathcal{H}$ can be viewed as the set of partial isometries $D : H \to \mathbb{R}^K$.

Recall the definition of the minimal risk for LTL

$$\begin{aligned} \mathcal{E}_\eta^* &= \min_{h\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\left[\min_{f\in\mathcal{F}} \mathbb{E}_{Z\sim\mu}\ell\Big(f\big(h\left(X\right)\big), Y\Big)\right] \\ &= \min_{D\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\left[\min_{\|w\|\leq B} \mathbb{E}_{Z\sim\mu}\xi\big(\left\langle w, DX\right\rangle \operatorname{sgn}\left(\left\langle u_\mu, X\right\rangle\right)\big)\right]. \end{aligned}$$

Let D_M be the partial isometry mapping M onto $\mathbb{R}^K$. Then $D_M \in \mathcal{H}$ and for every unit vector $u \in H$ we have $D_M(Bu) \in \mathcal{F}$. Thus

$$\begin{aligned} \mathcal{E}_\eta^* &\leq \mathbb{E}_{\mu\sim\eta}\left[\mathbb{E}_{Z\sim\mu}\xi\big(\left\langle D_M\left(Bu_\mu\right), D_M X\right\rangle \operatorname{sgn}\left(\left\langle u_\mu, X\right\rangle\right)\big)\right] \\ &= \mathbb{E}_{\mu\sim\eta}\left[\mathbb{E}_{X\sim\sigma}\xi\left(B\left|\left\langle u_\mu, X\right\rangle\right|\right)\right] \\ &\leq \sup_{\|u\|\leq 1} \mathbb{E}_{X\sim\sigma}\xi\left(B\left|\left\langle u, X\right\rangle\right|\right). \end{aligned}$$

For any unit vector $u \in H$ the density of the distribution of $|\langle u, X\rangle|$ under σ has maximum A_{d-1}/A_d, where A_d is the volume of $\mathcal{S}_d$ in the metric inherited from $\mathbb{R}^d$. This density can

therefore be bounded by $\sqrt{d}/2$. Thus

$$\mathcal{E}_\eta^* \leq \sqrt{d} \int_{-\infty}^{\infty} \xi\left(B\,|s|\right) ds = \frac{\sqrt{d}}{2B} = \epsilon,$$

if we set $B = \sqrt{d}/\left(2\epsilon\right)$. This choice is made to ensure that the Lipschitz loss upper bounds the 0-1-loss.

Now let $\bar{\mathbf{Z}}$ be a multi-sample generated from the environment η and assume that we have solved the optimization problem (1) to obtain the representation (or feature-map) $\hat{D} \in \mathcal{H}$. Using the excess risk bound, Theorem 5, and the fact that $\|C\|_\infty = 1/d$ and $\|C\|_1 = 1$, we get with probability at least $1-\delta$ in the draw of $\bar{\mathbf{Z}}$, that

$$\begin{aligned} \mathcal{E}_\eta(\hat{D}) &\leq \epsilon + \frac{\sqrt{2\pi}}{2\epsilon}\left(K\sqrt{\frac{d}{T}} + \sqrt{\frac{K}{n}}\right) + \sqrt{\frac{8\ln\left(4/\delta\right)}{T}} \\ &\leq \sqrt{\sqrt{2\pi}\left(K\sqrt{\frac{d}{T}} + \sqrt{\frac{K}{n}}\right)} + \sqrt{\frac{8\ln\left(4/\delta\right)}{T}}, \end{aligned} \tag{9}$$

if we optimize ϵ. This guarantees the expected performance of future uses of the representation $\hat{D}$. The high dimension still is a hindrance to the estimation of the representation, but, as announced, its effect vanishes in the limit $T \to \infty$. The individual samples must only well outnumber the dimension K, roughly the number of shared features.

We compare this upper bound to a lower bound for a large class of algorithms which learn the tasks independently.

Definition 6 *An algorithm $f : \mathcal{S}_d \times \{-1,1\}^n \to \mathcal{S}_d$ is called* orthogonally equivariant *if*

$$f\left(V\mathbf{x}, \mathbf{y}\right) = Vf\left(\mathbf{x}, \mathbf{y}\right), \text{ for every orthogonal matrix } V \in \mathbb{R}^{d\times d}. \tag{10}$$

For data transformed by an orthogonal transformation an orthogonally equivariant algorithm produces a correspondingly transformed hypothesis. Any algorithm which does not depend on a specific coordinate system is orthogonally equivariant. This class of algorithms includes all kernel methods, but it excludes the Lasso (L1-norm regularization). If the known properties of the problem posses a rotation symmetry only equivariant algorithms make sense.

Below we denote by $\mathrm{err}(u,v)$ the misclassification error between the half-spaces associated with unit vectors u and v, that is $\mathrm{err}(u,v) = \Pr_{x\sim\sigma}\{\langle u,x\rangle\langle v,x\rangle < 0\}$. The following lower error bound is given in (Maurer and Pontil, 2008).

Theorem 7 *Let $n < d$ and suppose that $f : \mathcal{S}_d^n \times \{-1,1\}^n \to \mathcal{S}_d$ is an orthogonally equivariant algorithm. Then for $\delta > 0$ with probability at least $1-\delta$ in the draw of $\mathbf{X} \sim \boldsymbol{\sigma}^n$ we have for every $u \in \mathcal{S}_d$ that*

$$\mathrm{err}\big(u, f(\mathbf{X}, u(\mathbf{X}))\big) \geq \frac{1}{\pi}\left(\sqrt{\frac{d-n}{d}} - \sqrt{\frac{\ln\left(1/\delta\right)}{d}}\right),$$

where $u\left(\mathbf{X}\right) = \left(sgn\left\langle u, X_1\right\rangle, \ldots, sgn\left\langle u, X_n\right\rangle\right)$.

If we use a union bound to subtract the upper bound (9) from this lower bound we obtain high probability guarantees for the advantage of representation learning over other algorithms.

In the following section we plot the phase diagram derived here, namely the difference between the uniform lower bound and our upper bound, and compare it with empirical results (see Figure 4).

3.2 Numerical Experiments

The purpose of the experiments is to compare MTL and LTL to independent task learning (ITL) in the simple setting of linear feature learning (or subspace learning)[1]. We wish to study the regime in which MTL/LTL learning is beneficial over ITL as a function of the number of tasks T and the sample size per task n.

We consider noiseless linear binary classification tasks, namely halfspace learning. We generated the data in the following way. The ground truth weight vectors $u_1, \ldots, u_T$ are obtained by the equation $u_t = Dc_t$, where $c_t \in \mathbb{R}^K$ is sampled from the uniform distribution on the unit sphere in $\mathbb{R}^K$, and the dictionary $D \in \mathbb{R}^{d\times K}$ is created by first sampling a d-dimension orthonormal matrix from the Haar measure, and then selecting the first K columns (atoms). We create all input marginals by sampling from the uniform distribution on the $\sqrt{d}$ radius sphere in $\mathbb{R}^d$. For each task we sample n instances to build the training set, and 1000 instances for the test set.

We train the methods with the hinge loss function $h(z) := \max\{0, 1 - z/c\}$, where c is the margin. We choose $c = 2/\epsilon$, so that the true error relative to the best hypothesis is of order ϵ. We fixed the value of ϵ to be $(K/n)^{1/2}$. For ITL we optimize that loss function constraining the ℓ_2-norm of the weights, for MTL and LTL we constrain D to have a Frobenius norm less or equal than 1, and each c_t is constrained to have an ℓ_2 norm less or equal than 1. During testing we use the 0-1 loss. For example the task-average error is evaluated as

$$\frac{1}{T}\sum_{t=1}^{T}\frac{1}{1000}\sum_{i=1}^{1000} 1\{\mathrm{sign}(\langle u_t, x_i\rangle) \neq \mathrm{sign}(\langle \hat{u}_t, x_i\rangle)\} \tag{11}$$

where $\hat{u}_t$ are the weight vectors learned by the assessed method.

3.3 MTL Experiment

We first discuss the MTL experiment. We let $d = 50$, and vary $T \in \{5, 10, \ldots, 150\}$, $n \in \{5, 10, \ldots, 150\}$ considering the cases $K = 2$ and $K = 5$. In Figure 1 we report the difference between the classification error of the two methods. These results are obtained by repeating the experiment 10 times, reporting the average difference. In each trial a different set of input points and underlying weight vectors are generated for each task. In the MTL case the training error was always below 0.1 and on average it was smaller than 0.04. This suggests that despite the problem being non-convex, the gradient optimization algorithm finds a good suboptimal solution.

1. The code used for the experiments presented in this section is available at `http://romera-paredes.com/multitask-representation`.

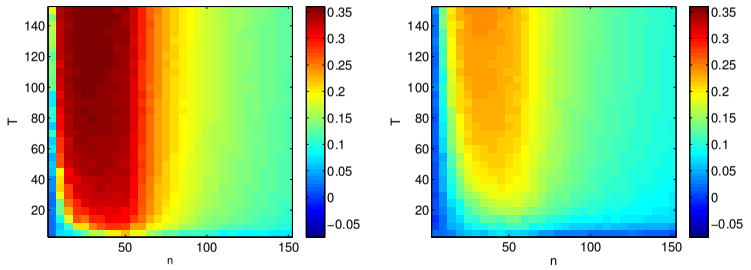


Figure 1: Difference of test classification error, computed according to eq. (11), between ITL and MTL. The vertical axis represents the number of training tasks, and the horizontal axis the number of training instances per task. In the left column $K = 2$, and in the right column $K = 5$.

We have made further experiments to assess the influence of other data settings on the difference between ITL and MTL. In the first of those experiments we have explored the cases in which the dictionary size is overestimated and underestimated. The results are shown in Figure 2. In the left plot the dictionary size is overestimated, in particular the ground truth number of atoms is 2, and the number of atoms used in the MTL method is 5. We can appreciate a similar pattern as the one we saw in Figure 1, although differences between ITL and MTL are not as high. The performance is slightly hampered, as expected due to an overestimation of the number of atoms. On the other hand in Figure 2 (right) we show the results when the number of atoms in the ground truth dictionary is 5, whereas the number of atoms used in the MTL approach is 2. In this case we see that the performance is severely affected by the underestimation of the size of the dictionary, yet we observe that MTL performs better than ITL in the same regime as in the previous experiments.

In the second of these experiments we study how the results are affected when the data are noisy. To do so, we have generated the data so that the ground truth label for instance x_i for task t is given by $\text{sign}(\langle u_t, x_i\rangle + \varepsilon_{ti})$, where $\varepsilon_{ti} \sim \mathcal{N}(0, 1)$. The dictionary size, for both the ground truth and the MTL approach, is $K = 2$. The results are shown in Figure 3, and we can see a similar behaviour as the one in Figure 1, with somewhat smaller differences between ITL and MTL.

3.4 LTL Experiment

In this experiment we test how the dictionary learned at the training stage helps learning new tasks, and we assess how similar the resultant figure is in comparison to the phase diagram derived in the previous section.

The data is generated according to the settings given in the MTL experiment. Furthermore, 50 new tasks are sampled following the same scheme previously described for the purpose of computing the LTL test error. We present the results in Figure 4 (Top). Similar

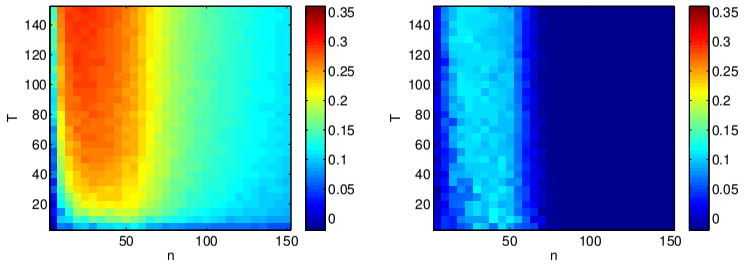


Figure 2: Difference of test classification error, computed according to eq. (11), between ITL and MTL, when the number of atoms of the ground truth dictionary does not match the number of atoms of the MTL model. The plot in the left shows the experiment in which the ground truth number of atoms is 2, whereas the number of atoms used in the MTL approach is 5. The plot in the right shows the opposite scenario: 5 atoms as ground truth, and 2 atoms in the MTL model. The vertical axis represents the number of training tasks, and the horizontal axis the number of training instances per task.

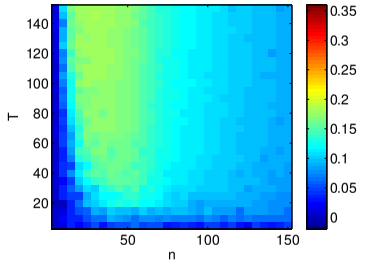


Figure 3: Difference of test classification error, computed according to eq. (11), between ITL and MTL, when adding Gaussian noise to the ground truth labels. The vertical axis represents the number of training tasks, and the horizontal axis the number of training instances per task.

to the previous experiment, we report the average difference between the test error of ITL and LTL after 10 trials.

In Figure 4 (Bottom) we present the theoretical phase diagram, which was generated using $1 \leq T \leq 10^{11}$, $1 \leq n \leq 10^5$, $d = 10^5$, $\delta = 0.0001$. We also plot as a dark line the points in which there is no difference in the performances between ITL and LTL.

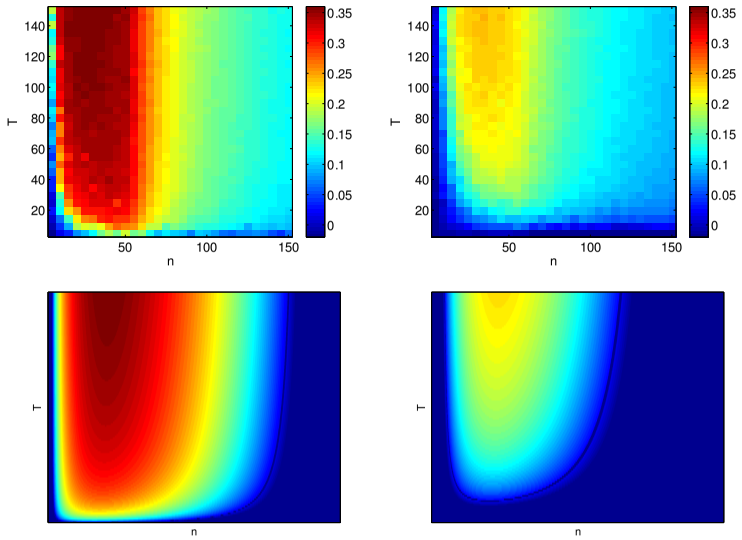


Figure 4: The vertical axis represents the number of training tasks, and the horizontal axis the number of training instances per task. Plots in the top row show the difference of test classification error, computed on 50 new tasks, between ITL and LTL. Plots in the bottom row show the region where the upper bound for LTL is smaller than the lower bound for any equivariant algorithm for ITL (see the discussion in Section 3.1, in particular Equation 9) using $1 \leq T \leq 10^{11}$, $1 \leq n \leq 10^5$, $d = 10^5$, and $\delta = 0.0001$. In the left column $K = 2$, and in the right column $K = 5$.

The reader may object about the much larger parameter values used to generate the plots of theoretical differences, in comparison to the experimental settings. These large parameters are partly a consequence of an accumulation of somewhat loose estimates in the derivation of both the upper and lower bounds. Another reason is that in applying it to a noiseless, finite-dimensional problem (for clarity) we have sacrificed two strong points of our results: independence of input dimension and its agnostic nature. Apart from the large parameter values the theoretical prediction shown in Figure 4 (Bottom) is in very good agreement with the experimental results in Figure 4 (Top).

We have also performed experiments in order to evaluate the influence of noise and under/overestimation of the dictionary size on the difference between ITL and LTL. We obtained similar results as the ones reported for MTL in Figures 2 and 3.

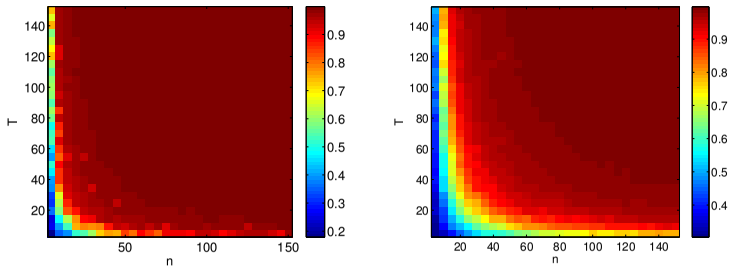


Figure 5: Similarity between the learned dictionary $\hat{D}$ and the ground truth dictionary D, according the similarity measure $s(\hat{D}, D)$ in Equation (12). The vertical axis represents the number of training tasks, and the horizontal axis the number of training instances per task. Left plot: $K = 2$. Right plot: $K = 5$.

Finally, we have compared the learned dictionary, $\hat{D}$, with the ground truth, D, in the same regime of parameters used for the previous experiments. Note that a dictionary could be correct up to permutations and changes of sign of its atoms. To overcome this issue we use the similarity measure

$$s(\hat{D}, D) = \frac{1}{K} \left\| D^\top \hat{D} \right\|_{\mathrm{tr}}, \tag{12}$$

where $\| \cdot \|_{\mathrm{tr}}$ is the sum of singular values of a matrix. Note that $s(\hat{D}, D) = 1$ if $\hat{D}$ and D are the same matrix up to permutation of columns and changes of sign, as requested. The results are found in Figure 5.

Figure 5 indicate that the learned dictionary is close to the true dictionary even for small sample sizes, provide T is large. This supports the results in Figure 1 and the top plots in Figure 4, where MTL or LTL are found to be superior to ITL in this regime, respectively.

4. Proofs of the Main Theorems

In this section we prove our principal results, Theorem 1 and Theorem 2. In preparation for the proofs we will first present some important auxiliary results.

4.1 Tools

We denote by γ a generic vector of independent standard normal variables, whose dimension will be clear from context. A central role in this paper is played by the Gaussian average $G(Y)$ of a set $Y \subseteq \mathbb{R}^n$, which is defined as

$$G(Y) = \mathbb{E} \sup_{y \in Y} \langle \gamma, y \rangle = \mathbb{E} \sup_{y \in Y} \sum_{i=1}^{n} \gamma_i y_i.$$

The reader who is concerned about the measurability of the random variable on the right hand side should replace Y by a countable dense subset of Y, with similar adjustments wherever the Gaussian averages occur.

Rademacher averages, where the γ_i are replaced by uniform $\{-1,1\}$-distributed variables, are somewhat more popular in the literature. We use Gaussian averages instead, because in most cases they are just as easy to bound and possess special properties (Theorem 10 and Theorem 12 below) which we need in our analysis.

The first result is a standard tool to prove uniform bounds on the estimation error in terms of Gaussian averages (Bartlett and Mendelson, 2002).

Theorem 8 *Let $\mathcal{F}$ be a real-valued function class on a space $\mathcal{X}$ and let $\mathbf{X} = (X_1, ..., X_n)$ be a vector of independent random variables and $\mathbf{X}'$ iid to $\mathbf{X}$. Then*

(i)

$$\mathbb{E}_{\mathbf{X}} \sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \left(\mathbb{E}_{\mathbf{X}'} \left[f\left(X_i'\right) \right] - f\left(X_i\right) \right) \leq \frac{\sqrt{2\pi} \mathbb{E}_{\mathbf{X}} G\left(\mathcal{F}\left(\mathbf{X}\right)\right)}{n}$$

(ii) if the members of $\mathcal{F}$ have values in $[0,1]$ then with probability greater than $1-\delta$ in $\mathbf{X}$ for all $f \in \mathcal{F}$

$$\frac{1}{n} \sum_{i=1}^{n} \left(\mathbb{E}_{\mathbf{X}'} \left[f\left(X_i'\right) \right] - f\left(X_i\right) \right) \leq \frac{\sqrt{2\pi} G\left(\mathcal{F}\left(\mathbf{X}\right)\right)}{n} + \sqrt{\frac{9 \ln\left(2/\delta\right)}{2n}}.$$

The following theorem is a vector-valued version of the above, is useful for bounds on the task-averaged estimation error (Ando and Zhang (2005), Maurer (2006b)).

Theorem 9 *Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to [0,1]^T$, and let $\mu_1, ..., \mu_T$ be probability measures on $\mathcal{X}$ with $\bar{\mathbf{X}} = (\mathbf{X}_1, ..., \mathbf{X}_T) \sim \prod_{t=1}^{T} \left(\mu_t\right)^n$ where $\mathbf{X}_t = (X_{t1}, ..., X_{tn})$. Then with probability greater than $1-\delta$ in $\bar{\mathbf{X}}$ for all $f \in \mathcal{F}$*

$$\frac{1}{T} \sum_t \left(\mathbb{E}_{X \sim \mu_t} \left[f_t\left(X\right) \right] - \frac{1}{n} \sum_{ti} f_t\left(X_{ti}\right) \right) \leq \frac{\sqrt{2\pi} G\left(Y\right)}{nT} + \sqrt{\frac{9 \ln\left(2/\delta\right)}{2nT}},$$

where $Y \subset \mathbb{R}^{Tn}$ is the random set defined by $Y = \left\{ \left(f_t\left(X_{ti}\right) \right) : f \in \mathcal{F} \right\}$.

The previous two theorems replace the problem of proving uniform bounds by the problem of bounding Gaussian averages. One key result in the latter direction is known as Slepian's Lemma (Slepian (1962), Ledoux and Talagrand (1991)).

Theorem 10 *Let Ω and Ξ be mean zero, separable Gaussian processes indexed by a common set $\mathcal{S}$, such that*

$$\mathbb{E}\left(\Omega_{s_1} - \Omega_{s_2}\right)^2 \leq \mathbb{E}\left(\Xi_{s_1} - \Xi_{s_2}\right)^2 \text{ for all } s_1, s_2 \in \mathcal{S}.$$

Then

$$\mathbb{E} \sup_{s \in \mathcal{S}} \Omega_s \leq \mathbb{E} \sup_{s \in \mathcal{S}} \Xi_s.$$

The following corollary is the key to our bound for LTL.

Corollary 11 *Let $Y \subseteq \mathbb{R}^n$ and let $\phi : Y \to \mathbb{R}^m$ be (Euclidean) Lipschitz with Lipschitz constant L. Then*

$$G\left(\phi\left(Y\right)\right) \leq L G\left(Y\right).$$

Proof Define two Gaussian processes indexed by Y as

$$\Omega_y = \sum_{k=1}^{m} \gamma_k \phi\left(y\right)_k \text{ and } \Xi_y = L \sum_{i=1}^{n} \gamma_i' y_i,$$

with independent γ_k and γ_i'. Then for any $y, y' \in Y$

$$\mathbb{E}\left(\Omega_y - \Omega_{y'}\right)^2 = \left\| \phi\left(y\right) - \phi\left(y'\right) \right\|^2 \leq L^2 \left\| y - y' \right\|^2 = \mathbb{E}\left(\Xi_{s_1} - \Xi_{s_2}\right)^2,$$

so that, by Slepian's Lemma,

$$G\left(\phi\left(Y\right)\right) = \mathbb{E} \sup_{y \in Y} \Omega_y \leq \mathbb{E} \sup_{y \in Y} \Xi_y = L G\left(Y\right).$$

■

In many applications this is applied when $n = m$ and ϕ is defined by $\phi\left(y_1, ..., y_n\right) = \left(\phi_1\left(y_1\right), ..., \phi_n\left(y_n\right)\right)$ where the real functions $\phi_1, ..., \phi_n$ have Lipschitz constant L.

At one point we will need a generalization of the above corollary, which allows to select ϕ from an entire class of Lipschitz functions. We will use the following result, which is taken from Maurer (2014). It will play an important role in the proof of Theorem 13 below.

Theorem 12 *Let $Y \subseteq \mathbb{R}^n$ have (Euclidean) diameter $D\left(Y\right)$ and let $\mathcal{F}$ be a class of functions $f : Y \to \mathbb{R}^m$, all of which have Lipschitz constant at most $L\left(\mathcal{F}\right)$. Then for any $y_0 \in Y$*

$$G\left(\mathcal{F}\left(Y\right)\right) \leq c_1 L\left(\mathcal{F}\right) G\left(Y\right) + c_2 D\left(Y\right) Q\left(\mathcal{F}\right) + G\left(\mathcal{F}\left(y_0\right)\right),$$

where c_1 and c_2 are universal constants and

$$Q\left(\mathcal{F}\right) = \sup_{\mathbf{y}, \mathbf{y}' \in Y,\ \mathbf{y} \neq \mathbf{y}'} \mathbb{E} \sup_{f \in \mathcal{F}} \frac{\left\langle \boldsymbol{\gamma}, f\left(\mathbf{y}\right) - f\left(\mathbf{y}'\right) \right\rangle}{\left\| \mathbf{y} - \mathbf{y}' \right\|}.$$

Note that the result allows us to minimize the right hand side in y_0. Analogs of Theorem 10 and Theorem 12 are not available for Rademacher averages. This is the reason why we use the slightly more exotic Gaussian averages.

4.2 Proof of the Excess Risk Bound for the Average Risk

We first establish the following uniform bound. It is of some interest in its own right, in particular since the problem (1) is often non-convex, so that the excess risk bound may not be meaningful in practice. Recall the definition of Q given in Equation (4).

Theorem 13 *Let $\mu_1, \ldots, \mu_T$ be probability measures on $\mathcal{Z}$ and let $Z_{t1}, \ldots, Z_{tn}$ be i.i.d. from μ_t, for $t = 1, \ldots, T$. Let $\delta \in (0,1)$. With probability at least $1-\delta$ in the draw of a multisample $\bar{\mathbf{Z}}$, it holds for every $h \in \mathcal{H}$ and every $f_1, \ldots, f_T \in \mathcal{F}$ that*

$$\mathcal{E}_{\text{avg}}\left(h, f_1, ..., f_T\right) - \frac{1}{Tn}\sum_{ti} \ell\left(f_t\left(h\left(X_{ti}\right)\right), Y_{ti}\right) \leq c_1 \frac{LG(\mathcal{H}(\bar{\mathbf{X}}))}{nT} + c_2 \frac{Q \sup_{h\in\mathcal{H}} \left\|h\left(\bar{\mathbf{X}}\right)\right\|}{n\sqrt{T}} + \sqrt{\frac{9\ln\left(2/\delta\right)}{2nT}},$$

where c_1 and c_2 are universal constants.

Proof By Theorem 9, with probability at least $1-\delta$ in $\bar{\mathbf{Z}}$, for all $h \in \mathcal{H}$ and all $f_1, ..., f_T \in \mathcal{F}$, we have that

$$\mathcal{E}_{\text{avg}}\left(h, f_1, ..., f_T\right) - \frac{1}{Tn}\sum_{ti} \ell\left(f_t\left(h\left(X_{ti}\right)\right), Y_{ti}\right) \leq \frac{\sqrt{2\pi}}{nT} G\left(S\right) + \sqrt{\frac{9\ln\left(2/\delta\right)}{2nT}}, \tag{13}$$

where $S = \left\{\left(\ell\left(f_t\left(h\left(X_{ti}\right)\right), Y_{ti}\right)\right) : f \in \mathcal{F}^T \text{ and } h \in \mathcal{H}\right\} \subseteq \mathbb{R}^{Tn}$. By the Lipschitz property of the loss function ℓ and the contraction lemma Corollary 11 (recall the remark which follows its proof) we have $G\left(S\right) \leq G\left(S'\right)$, where $S' = \left\{\left(f_t\left(h\left(X_{ti}\right)\right)\right) : f \in \mathcal{F}^T \text{ and } h \in \mathcal{H}\right\} \subseteq \mathbb{R}^{Tn}$.

Recall that $\mathcal{H}\left(\bar{\mathbf{X}}\right) \subseteq \mathbb{R}^{KTn}$ is defined by

$$\mathcal{H}\left(\bar{\mathbf{X}}\right) = \left\{\left(h_k\left(X_{ti}\right)\right) : h \in \mathcal{H}\right\},$$

and define a class of functions $\mathcal{F}' : \mathbb{R}^{KTn} \to \mathbb{R}^{Tn}$ by

$$\mathcal{F}' = \left\{y \in \mathbb{R}^{KTn} \mapsto \left(f_t\left(y_{ti}\right)\right) : \left(f_1, ..., f_T\right) \in \mathcal{F}^T\right\}.$$

Then $S' = \mathcal{F}'\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right)$, and by Theorem 12 for universal constants c_1' and c_2'

$$G\left(S'\right) \leq c_1' L\left(\mathcal{F}'\right) G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right) + c_2' D\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right) Q\left(\mathcal{F}'\right) + \min_{y\in Y} G\left(\mathcal{F}\left(y\right)\right). \tag{14}$$

We now proceed by bounding the individual terms in the right hand side above. Let $y, y' \in \mathbb{R}^{KTn}$, where $y = \left(y_{ti}\right)$ with $y_{ti} \in \mathbb{R}^K$ and $y' = \left(y'_{ti}\right)$ with $y'_{ti} \in \mathbb{R}^K$. Then for $f = \left(f_1, ..., f_T\right) \in \mathcal{F}^T$

$$\begin{aligned}\left\|f\left(y\right) - f\left(y'\right)\right\|^2 &= \sum_{ti}\left(f_t\left(y_{ti}\right) - f_t\left(y'_{ti}\right)\right)^2 \\ &\leq L^2 \sum_{ti}\left\|y_{ti} - y'_{ti}\right\|^2 = L^2\left\|y - y'\right\|^2,\end{aligned}$$

so that $L\left(\mathcal{F}'\right) \leq L$. Also

$$\begin{aligned}&\mathbb{E}\sup_{g\in\mathcal{F}'}\left\langle\gamma, g\left(y\right) - g\left(y'\right)\right\rangle \\ =\ &\mathbb{E}\sup_{(f_1,...,f_T)\in\mathcal{F}^T}\sum_{ti}\gamma_{ti}\left(f_t\left(y_{ti}\right) - f_t\left(y'_{ti}\right)\right) \\ =\ &\sum_t \mathbb{E}\sup_{f\in\mathcal{F}}\sum_i \gamma_i\left(f\left(y_{ti}\right) - f\left(y'_{ti}\right)\right) \\ \leq\ &\sqrt{T}\left(\sum_t\left(\mathbb{E}\sup_{f\in\mathcal{F}}\sum_i\gamma_i\left(f\left(y_{ti}\right) - f\left(y'_{ti}\right)\right)\right)^2\right)^{1/2} \\ \leq\ &\sqrt{T}\left(\sum_t Q^2\sum_i\left\|y_{ti} - y'_{ti}\right\|^2\right)^{1/2} \\ =\ &\sqrt{T}Q\left\|y - y'\right\|,\end{aligned}$$

whence $Q\left(\mathcal{F}'\right) = \sqrt{T}Q$. Finally we take $y_0 = 0$ and the last term in (14) vanishes since $f\left(0\right) = 0$ for all $f \in \mathcal{F}$. Substitution in (14) and using $G\left(S\right) \leq G\left(S'\right)$ we arrive at

$$G\left(S\right) \leq c_1' L G\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right) + c_2'\sqrt{T} D\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right) Q.$$

Bounding $D\left(\mathcal{H}\left(\bar{\mathbf{X}}\right)\right) \leq 2\sup_h\left\|h\left(\bar{\mathbf{X}}\right)\right\|$ and substitution in (13) gives the result. ■

Proof of Theorem 1 Let h^* and $f_1^*, ..., f_T^*$ be the minimizers in the definition of $\mathcal{E}_{\text{avg}}^*$. Then

$$\begin{aligned}&\mathcal{E}_{\text{avg}}(\hat{h}, \hat{f}_1, ..., \hat{f}_T) - \mathcal{E}_{\text{avg}}^* \\ =\ &\left(\mathcal{E}_{\text{avg}}(\hat{h}, \hat{f}_1, ..., \hat{f}_T) - \frac{1}{nT}\sum_{ti}\ell(\hat{f}_t(\hat{h}(X_{ti})), Y_{ti})\right) \\ +\ &\left(\frac{1}{nT}\sum_{ti}\ell(\hat{f}_t(\hat{h}(X_{ti})), Y_{ti}) - \frac{1}{nT}\sum_{ti}\ell\left(f_t^*\left(h^*\left(X_{ti}\right)\right), Y_{ti}\right)\right) \\ +\ &\left(\frac{1}{nT}\sum_{ti}\ell\left(f_t^*\left(h^*\left(X_{ti}\right)\right), Y_{ti}\right) - \frac{1}{T}\sum_t \mathbb{E}_{(X,Y)\sim\mu_t}\ell\left(f_t^*\left(h^*\left(X\right)\right), Y\right)\right).\end{aligned}$$

The last term involves only the nT random variables $\ell\left(f_t^*\left(h^*\left(X_{ti}\right)\right), Y_{ti}\right)$ with values in $[0,1]$. It can be bounded with probability $1-\delta/2$ by $\sqrt{\ln\left(2/\delta\right)/\left(2Tn\right)}$ using Hoeffding's inequality. The middle term is non-positive by definition of $\hat{h}, \hat{f}_1, ..., \hat{f}_T$ being the corresponding minimizers. There remains the first term which we bound by

$$\sup_{h\in\mathcal{H}, f_1, ..., f_T\in\mathcal{F}} \mathcal{E}_{\text{avg}}\left(h, f_1, ..., f_T\right) - \frac{1}{Tn}\sum_{ti}\ell\left(f_t\left(h\left(X_{ti}\right)\right), Y_{ti}\right).$$

and appeal to Theorem 13 to bound the supremum. A union bound then completes the proof. ■

4.3 Proof of the Excess Risk Bound for Learning-to-learn

Recall the definition of the algorithm parametrized by $h \in \mathcal{H}$

$$a(h)_{\mathbf{Z}} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_i \ell(f(h(X_i)), Y_i) \text{ for } \mathbf{Z} \in \mathcal{Z}^n$$

and the associated minimum $m(h)_{\mathbf{Z}}$. Also recall that

$$\mathcal{E}_\eta(h) = \mathbb{E}_{\mu\sim\eta}\mathbb{E}_{\mathbf{Z}\sim\mu^n}\mathbb{E}_{(X,Y)\sim\mu}\ell(a(h)_{\mathbf{Z}}(X), Y)$$

and the two measures μ_η and ρ_η induced by the environment η and defined by

$$\mu_\eta(A) = \mathbb{E}_{\mu\sim\eta}\mu(A) \text{ for } A \subseteq \mathcal{Z} \text{ and } \rho_\eta(A) = \mathbb{E}_{\mu\sim\eta}\mu^n(A) \text{ for } A \subseteq \mathcal{Z}^n.$$

Also recall the definition of Q' given in Equation (5). Again we begin with a uniform bound.

Theorem 14 *Let $\delta \in (0,1)$. (i) With probability at least $1-\delta$ in $\bar{\mathbf{Z}} \sim \rho_\eta^T$ it holds for every $h \in \mathcal{H}$ that*

$$\mathcal{E}_\eta(h) - \frac{1}{T}\sum_t m(h)_{\mathbf{Z}_t} \leq$$

$$\frac{\sqrt{2\pi}LG(\mathcal{H}(\bar{\mathbf{x}}))}{T\sqrt{n}} + \sqrt{2\pi}Q' \sup_{h\in\mathcal{H}} \sqrt{\frac{\mathbb{E}_{(X,Y)\sim\mu_\eta}\left[\|h(X)\|^2\right]}{n}} + \sqrt{\frac{9\ln(2/\delta)}{2T}}.$$

(ii) With probability at least $1-\delta$ in $\bar{\mathbf{Z}} \sim \rho_\eta^T$ it holds for every $h \in \mathcal{H}$ that

$$\mathcal{E}_\eta(h) - \frac{1}{T}\sum_t m(h)_{\mathbf{Z}_t} \leq$$

$$\frac{\sqrt{2\pi}LG(\mathcal{H}(\bar{\mathbf{x}}))}{T\sqrt{n}} + \frac{\sqrt{2\pi}Q'\sum_t \sup_{h\in\mathcal{H}}\|h(\mathbf{X}_t)\|}{nT} + \sqrt{\frac{16\ln(4/\delta)}{T}}.$$

Proof The key to the proof is the decomposition bound

$$\begin{aligned}\sup_{h\in\mathcal{H}} \mathcal{E}_\eta(h) - \frac{1}{T}\sum_t m(h)_{\mathbf{Z}_t} \leq & \sup_{h\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\mathbb{E}_{\mathbf{Z}\sim\mu^n}\left[\mathbb{E}_{(X,Y)\sim\mu}\ell(a(h)_{\mathbf{Z}}(X), Y) - m(h)_{\mathbf{Z}}\right] \\ & + \sup_{h\in\mathcal{H}}\left[\mathbb{E}_{\mathbf{Z}\sim\rho_\eta}[m(h)_{\mathbf{Z}}] - \frac{1}{T}\sum_{t=1}^T m(h)_{\mathbf{Z}_t}\right]. \end{aligned} \quad (15)$$

In turn we will bound both terms on the right hand side above. A bound on the second term means that we can predict the empirical risk on the data of a future task uniformly in h. A bound on the first term means that we can predict the true risk from the empirical risk on the future task.

We first bound the second term in the right hand side of (15), and use Theorem 8-(ii) on the class of functions

$$\{\mathbf{z} \in \mathcal{Z}^n \mapsto m(h)_{\mathbf{z}} : h \in \mathcal{H}\}$$

to get with probability at least $1-\delta$ in $\bar{\mathbf{Z}} \sim \rho_\eta^T$ that

$$\sup_{h\in\mathcal{H}}\left[\mathbb{E}_{\mathbf{Z}\sim\rho_\eta}[m(h)_{\mathbf{Z}}] - \frac{1}{T}\sum_{t=1}^T m(h)_{\mathbf{Z}_t}\right] \leq \frac{\sqrt{2\pi}}{T}G(S) + \sqrt{\frac{9\ln(2/\delta)}{2T}},$$

where S is the subset of $\mathbb{R}^T$ defined by

$$S = \left\{\left(m(h)_{\mathbf{Z}_1}, ..., m(h)_{\mathbf{Z}_T}\right) : h \in \mathcal{H}\right\}.$$

We will bound the Gaussian average of S using Slepian's inequality (Theorem 10). Define two Gaussian processes indexed by $\mathcal{H}$ as

$$\Omega_h = \sum_t \gamma_t m(h)_{\mathbf{z}_t} \text{ and } \Xi_h = \frac{L}{\sqrt{n}}\sum_{kti}\gamma_{kti}h_k(x_{ti}).$$

Now for any $\mathbf{z} \in \mathcal{Z}^n$ and representations $h, h' \in \mathcal{H}$

$$\begin{aligned}\left(m(h)_{\mathbf{z}} - m\left(h'\right)_{\mathbf{z}}\right)^2 &= \left(\min_{f\in\mathcal{F}}\frac{1}{n}\sum_i \ell(f(h(x_i)), y_i) - \min_{f\in\mathcal{F}}\frac{1}{n}\sum_i \ell\left(f\left(h'(x_i)\right), y_i\right)\right)^2 \\ &\leq \left(\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_i \ell(f(h(x_i)), y_i) - \ell\left(f\left(h'(x_i)\right), y_i\right)\right)^2 \\ &\leq \frac{1}{n}\sup_{f\in\mathcal{F}}\sum_i\left(\ell(f(h(x_i)), y_i) - \ell\left(f\left(h'(x_i)\right), y_i\right)\right)^2 \\ &\leq \frac{L^2}{n}\sum_{ki}\left(h_k(x_i) - h'_k(x_i)\right)^2,\end{aligned}$$

where in the last step we used the Lipschitz properties of the loss function ℓ and of the members in the class $\mathcal{F}$. It follows that

$$\begin{aligned}\mathbb{E}(\Omega_h - \Omega_{h'})^2 &= \sum_t\left(m(h)_{\mathbf{z}_t} - m\left(h'\right)_{\mathbf{z}_t}\right)^2 \\ &\leq \frac{L(\mathcal{F})^2}{n}\sum_{kti}\left(h_k(x_{ti}) - h'_k(x_{ti})\right)^2 = \mathbb{E}(\Xi_h - \Xi_{h'})^2,\end{aligned}$$

so by Theorem 10

$$G(S) = \mathbb{E}\sup_k \Omega_h \leq \mathbb{E}\sup_k \Xi_h = \frac{L}{\sqrt{n}}G(\mathcal{H}(\bar{\mathbf{x}})).$$

The second term in the right hand side of (15) is thus bounded with probability $1-\delta$ by

$$\frac{\sqrt{2\pi}LG(\mathcal{H}(\bar{\mathbf{x}}))}{T\sqrt{n}} + \sqrt{\frac{9\ln(2/\delta)}{2T}}. \quad (16)$$

We now bound the first term on the right hand side of (15) by

$$\begin{aligned}
&\sup_{h\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\mathbb{E}_{\mathbf{Z}\sim\mu^n}\left[\mathbb{E}_{(X,Y)\sim\mu}\ell\left(a\left(h\right)_{\mathbf{X}}\left(X\right),Y\right) - m\left(h\right)_{\mathbf{Z}}\right] \\
\leq\ & \sup_{h\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\mathbb{E}_{\mathbf{Z}\sim\mu^n} \sup_{f\in\mathcal{F}}\left[\mathbb{E}_{(X,Y)\sim\mu}\ell\left(f\left(h\left(X\right)\right),Y\right) - \frac{1}{n}\sum_i \ell\left(f\left(h\left(X_i\right)\right),Y_i\right)\right].
\end{aligned}$$

For $\mathbf{Z}=(\mathbf{X},\mathbf{Y})\in\mathcal{Z}^n$ and $h\in\mathcal{H}$ denote with $\ell\left(\mathcal{F}\circ h\left(\mathbf{X}\right),\mathbf{Y}\right)$ the subset of $\mathbb{R}^n$ defined by

$$\ell\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right),\mathbf{Y}\right) = \left\{\left(\ell\left(f\left(h\left(X_i\right)\right),Y_i\right)\right) : f\in\mathcal{F}\right\}.$$

Using Theorem 8-(i) and the contraction lemma, Corollary 11, we can bound the last expression above by

$$\begin{aligned}
&\sup_{h\in\mathcal{H}} \mathbb{E}_{\mu\sim\eta}\mathbb{E}_{\mathbf{Z}\sim\mu^n} \sup_{f\in\mathcal{F}}\left[\mathbb{E}_{(X,Y)\sim\mu}\ell\left(f\left(h\left(X\right)\right),Y\right) - \frac{1}{n}\sum_i \ell\left(f\left(h\left(X_i\right)\right),Y_i\right)\right] \\
\leq\ & \sqrt{2\pi}\sup_{h\in\mathcal{H}} \mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\frac{G\left(\ell\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right),\mathbf{Y}\right)\right)}{n} \\
\leq\ & \sqrt{2\pi}\sup_{h\in\mathcal{H}} \mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\frac{G\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right)\right)}{n} \\
=\ & \frac{\sqrt{2\pi}}{n}\sup_{h\in\mathcal{H}} \mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\frac{G\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right)\right)}{\left\|h\left(\mathbf{X}\right)\right\|}\left\|h\left(\mathbf{X}\right)\right\| \\
\leq\ & \frac{\sqrt{2\pi}}{n}Q'\sup_{h\in\mathcal{H}} \mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\left\|h\left(\mathbf{X}\right)\right\|,
\end{aligned}$$

using Hoelder's inequality and the definition of Q' in the last step. But, using Jensen's inequality,

$$\mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\left\|h\left(\mathbf{X}\right)\right\| \leq \sqrt{\mathbb{E}_{\mathbf{Z}\sim\rho_\eta}\sum_i\left\|h\left(X_i\right)\right\|^2} = \sqrt{n\,\mathbb{E}_{(X,Y)\sim\mu_\eta}\left\|h\left(X\right)\right\|^2},$$

since $\mathbf{Z}\sim\rho_\eta$ is iid. Inserting this in the previous chain of inequalities and combining with (16) gives the first part of the theorem.

To obtain the data dependent bound we use the fact that, with probability at least $1-\delta/4$,

$$\begin{aligned}
\sup_{h\in\mathcal{H}} \mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\frac{G\left(\ell\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right),\mathbf{Y}\right)\right)}{n} &\leq \mathbb{E}_{\mathbf{X}\sim\rho_\eta}\sup_{h\in\mathcal{H}}\frac{G\left(\ell\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right),\mathbf{Y}\right)\right)}{n} &(17)\\
&\leq \frac{1}{T}\sum_t \sup_{h\in\mathcal{H}}\frac{G\left(\ell\left(\mathcal{F}\left(h\left(\mathbf{X}_t\right)\right),\mathbf{Y}_t\right)\right)}{n} + \sqrt{\frac{\ln\left(4/\delta\right)}{2T}} &(18)
\end{aligned}$$

The last inequality follows from Hoeffding's inequality since for any $h\in\mathcal{H}$ and any sample $\mathbf{Z}\in\mathcal{Z}^n$

$$\begin{aligned}
0 \ \leq\ & \frac{G\left(\ell\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right),\mathbf{Y}\right)\right)}{n} = \frac{1}{n}\mathbb{E}_\gamma\sup_f\sum_i\gamma_i\ell\left(f\left(h\left(X_i\right)\right),Y_i\right) \\
\leq\ & \frac{1}{n}\mathbb{E}_\gamma\left(\sum_i\gamma_i^2\right)^{1/2}\sup_f\left(\sum_i\ell\left(f\left(h\left(X_i\right)\right),Y_i\right)^2\right)^{1/2} \leq 1,
\end{aligned}$$

where we have also used the fact that the loss function ℓ has range in $[0,1]$. Bounding

$$G\left(\ell\left(\mathcal{F}\circ h\left(\mathbf{X}_t\right),\mathbf{Y}_t\right)\right) \leq Q'\left\|h\left(\mathbf{X}_t\right)\right\|$$

as above and combining (18) and (16) in (15) with a union bound gives the second inequality of the theorem.

■

Remark: In the proof of the fully data-dependent part above the bound on

$$\sup_{h\in\mathcal{H}}\mathbb{E}_{\mathbf{Z}\sim\boldsymbol{\rho}_\eta}\frac{G\left(\ell\left(\mathcal{F}\left(h\left(\mathbf{X}\right)\right),\mathbf{Y}\right)\right)}{n}$$

is very crude. Instead we could have again invoked Theorem 8 to get a better bound with a more complicated expression involving nested Gaussian averages. We have chosen the simpler path for greater clarity.

Proof of Theorem 2 Recall that

$$\mathcal{E}_\eta^* = \min_{h\in\mathcal{H}}\mathbb{E}_{\mu\sim\eta}\left[\min_{f\in\mathcal{F}}\mathbb{E}_{(X,Y)\sim\mu}\ell\left(f\left(h\left(X\right)\right),Y\right)\right].$$

We denote with h^* the minimizer in $\mathcal{H}$ occurring in the definition of $\mathcal{E}_\eta^*$. We have the following decomposition

$$\begin{aligned}
\mathcal{E}_\eta(\hat{h}) - \mathcal{E}_\eta^* \ =\ & \left(\mathcal{E}_\eta(\hat{h}) - \frac{1}{T}\sum_t m(\hat{h})_{\mathbf{Z}_t}\right) &(19)\\
&+\left(\frac{1}{T}\sum_t m(\hat{h})_{\mathbf{Z}_t} - \frac{1}{T}\sum_t m(h^*)_{\mathbf{Z}_t}\right) &(20)\\
&+\left(\frac{1}{T}\sum_t m(h^*)_{\mathbf{Z}_t} - \mathbb{E}_{\mathbf{Z}\sim\rho_\eta}\left[m\left(h^*\right)_{\mathbf{Z}}\right]\right) &(21)\\
&+\mathbb{E}_{\mu\sim\eta}\left[\mathbb{E}_{\mathbf{Z}\sim\mu^n}\left[m\left(h^*\right)_{\mathbf{Z}}\right] - \min_{f\in\mathcal{F}}\mathbb{E}_{(X,Y)\sim\mu}\ell\left(f\left(h^*\left(X\right)\right),Y\right)\right]. &(22)
\end{aligned}$$

For a fixed distribution μ let f_μ^* be the minimizer in $\min_{f\in\mathcal{F}}\mathbb{E}_{(X,Y)\sim\mu}\ell\left(f\left(h^*\left(X\right)\right),Y\right)$. By definition of $m\left(h^*\right)_{\mathbf{Z}}$ we have for every $\mu\sim\eta$ that

$$\begin{aligned}
\mathbb{E}_{\mathbf{Z}\sim\mu^n}\left[m\left(h^*\right)_{\mathbf{Z}}\right] &= \mathbb{E}_{\mathbf{Z}\sim\mu^n}\min_{f\in\mathcal{F}}\frac{1}{m}\sum_{i=1}^m\ell\left(f\left(h^*\left(X_i\right)\right),Y_i\right) \\
&\leq \mathbb{E}_{\mathbf{Z}\sim\mu^n}\frac{1}{m}\sum_{i=1}^m\ell\left(f_\mu^*\left(h^*\left(X_i\right)\right),Y_i\right) \\
&= \mathbb{E}_{(X,Y)\sim\mu}\ell\left(f_\mu^*\left(h^*\left(X\right)\right),Y\right),
\end{aligned}$$

since $\mathbf{Z}$ is iid. The term in (22) is therefore non-positive.

The term in (21) involves the deviation of the empirical and true averages of the T iid $[0,1]$-valued random variables $m(h^*)_{\mathbf{Z}_t}$. With Hoeffding's inequality this can be bounded with probability at least $1-\delta/8$ by $\sqrt{\ln(8/\delta)/(2T)}$. The term (20) is non-positive by the definition of $\hat{h}$.

There remains the term (19), which we bound by Theorem 14. The result now follows by combining this bound with the bound on (21) in a union bound and some numerical simplifications. ■

5. Conclusion

Several works have advocated that sharing features among tasks as a means to learning representations which capture invariant properties to tasks can be highly beneficial. In this paper, we studied the statistical properties of a general MTRL method, presenting bounds on its learning performance in both settings of MTL and LTL. Our work provides a rigorous justification of the benefit offered by MTRL over learning the tasks independently. To give the paper a clear focus we have illustrated this advantage in the case of linear feature learning. Our results however apply to fairly general classes of representations $\mathcal{H}$ and specifications $\mathcal{F}$, and similar conclusions may be derived for other nonlinear MTRL methods. We conclude by sketching specific cases which deserve a separated study:

- *Deep networks.* As we noted our bounds directly apply to multilayer, deep architectures obtained by iteratively composing linear transformations with nonlinear activation functions, such as the rectifier linear unit or the sigmoid functions. The representations learned by such methods tend to be specific in that only a subset of components are "active" on each given input, which makes our bounds particularly attractive for further analysis.

- *Sparse coding.* Another interesting case of our framework is obtained when the specialized class $\mathcal{F}$ consists of sparse linear predictors. This case has been considered in Maurer et al. (2013); Ruvolo and Eaton (2014) when the representation class consists of linear functions. Different choices of sparse classes $\mathcal{F}$ could lead to interesting learning methods.

- *Representations in RKHS.* As we already noted the feature maps forming the class $\mathcal{H}$ could be vector-valued functions in a reproducing kernel Hilbert space. Although kernel methods are more difficult to apply to large datasets required for MTRL and need additional approximation steps, the representations learned using for example Gaussian kernels would be very specific and suitable for our bounds.

Acknowledgments

We thank the reviewers for their helpful comments.

References

R. K. Ando, T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817-1853, 2005.

M. Anthony, P. Bartlett. *Learning in Neural Networks: Theoretical Foundations*. Cambridge University Press, 1999.

A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. *Machine Learning*, 73(3):243-272, 2008

P.L. Bartlett and S. Mendelson. Rademacher and Gaussian Complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463-482, 2002.

J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149-198, 2000.

S. Ben-David, N. Eiron, H. U. Simon. Limitations of Learning Via Embeddings in Euclidean Half Spaces, Journal of Machine Learning Research, (3):441-461, 2002.

S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. *Proc. 16th Annual Conference on Computational Learning Theory (COLT)*, pages 567–580, 2003.

R. Bhatia. *Matrix Analysis*. Springer, 1997.

R. Caruana. Multi-task learning. *Machine Learning*, 28(1):41–75, 1997.

G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 11:2597–2630, 2010.

S. Dasgupta, A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures and Algorithms*, 22: 60-65, 2003.

A. Ehrenfeucht, D. Haussler, M. Kearns, L. G. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3): 247-251, 1989.

T. Evgeniou, C. Micchelli and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

R. Girshick, J. Donahue, T. Darrell, J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2014.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

S. M. Kakade, S. Shalev-Shwartz, A. Tewari. Regularization techniques for learning with matrices. *Journal of Machine Learning Research* 13:1865–1890, 2012.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 2002.

I. Kuzborskij, and F. Orabona. Stability and Hypothesis Transfer Learning. *Proc. International Conference on Machine Learning (ICML)*, 2013.

M. Ledoux, M. Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes.* Springer, Berlin, 1991.

P. M. Long. On the sample complexity of PAC learning halfspaces against the uniform distribution. *IEEE Transactions on Neural Networks*, 6(6):1556-1559, 1995.

K. Lounici, M. Pontil, A.B. Tsybakov and S. van de Geer. Oracle inequalities and optimal inference under group sparsity. *Annals of Statistics*, 39(4):2164–2204, 2011.

A. Maurer. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:117–139, 2006.

A. Maurer. The Rademacher complexity of linear transformation classes. *Proc. 19th Annual Conference on Learning Theory (COLT)*, pages 65–78, 2006.

A. Maurer. A chain rule for the expected suprema of Gaussian processes. *Proc. 25th International Conference on Algorithmic Learning Theory*, 2014.

A. Maurer and M. Pontil. A uniform lower error bound for half-space learning. *Proc. 19th International Conference on Algorithmic Learning Theory*, pages 70–78, 2008.

A. Maurer and M. Pontil. Excess risk bounds for multitask learning with trace norm regularization. *Proc. 26th Annual Conference on Computational Learning Theory (COLT)*, 2013.

A. Maurer, M. Pontil, B. Romera-Paredes. Sparse coding for multitask and transfer learning. *Proc. 30th International Conference on Machine Learning*, JMLR W&CP 28 (2):343-351, 2013

A. Maurer, M. Pontil, B. Romera-Paredes. An inequality with applications to structured sparsity and multitask dictionary learning. *Proc. 27th Annual Conference on Computational Learning Theory (COLT)*, 2014.

S. Mendelson and A. Pajor. On singular values of matrices with independent rows. *Bernoulli* 12(5):761–773, 2006.

A. Pentina and C.H. Lampert. A PAC-Bayesian bound for lifelong learning. *Proc. International Conference on Machine Learning (ICML)*, 2014.

P. Ruvolo and E. Eaton. Online multi-task learning via sparse dictionary optimization. In Proc. of the 28th AAAI Conference on Artificial Intelligence, 2014.

D. Slepian. The one-sided barrier problem for gaussian noise. *Bell System Tech. J.*, 41:463-501, 1962.

S. Thrun and L. Pratt. *Learning to Learn*. Springer, 1998.

C. Widmer, M. Kloft, X. Lou X and G.Rätsch. Regularization-based multitask learning: With applications to genome biology and biomedical imaging. *German Journal on Artificial Intelligence*, 2013.

A.W. Van Der Vaart and J.A. Wellner. *Weak Convergence of Empirical Processes*. Springer, 1996

Journal of Machine Learning Research 17 (2016) 1-24 Submitted 7/15; Revised 2/16; Published 5/16

Model-free Variable Selection in Reproducing Kernel Hilbert Space

Lei Yang LY888@NYU.EDU
Department of Population Health
New York Univeristy
New York, NY, 10016, USA

Shaogao Lv LVSG716@SWUFE.EDU.CN
Center of Statistics
Southwestern University of Finance and Economics
Chengdu, Sichuan, 610074, China

Junhui Wang J.H.WANG@CITYU.EDU.HK
Department of Mathematics
City University of Hong Kong
Kowloon Tong, 999077, HongKong

Editor: Jie Peng

Abstract

Variable selection is popular in high-dimensional data analysis to identify the truly informative variables. Many variable selection methods have been developed under various model assumptions. Whereas success has been widely reported in literature, their performances largely depend on validity of the assumed models, such as the linear or additive models. This article introduces a model-free variable selection method via learning the gradient functions. The idea is based on the equivalence between whether a variable is informative and whether its corresponding gradient function is substantially non-zero. The proposed variable selection method is then formulated in a framework of learning gradients in a flexible reproducing kernel Hilbert space. The key advantage of the proposed method is that it requires no explicit model assumption and allows for general variable effects. Its asymptotic estimation and selection consistencies are studied, which establish the convergence rate of the estimated sparse gradients and assure that the truly informative variables are correctly identified in probability. The effectiveness of the proposed method is also supported by a variety of simulated examples and two real-life examples.

Keywords: group Lasso, high-dimensional data, kernel regression, learning gradients, reproducing kernel Hilbert space (RKHS), variable selection

1. Introduction

The rapid advance of technology has led to an increasing demand for modern statistical techniques, such as high-dimensional data analysis that has attracted tremendous interests in the past two decades. When analyzing high-dimensional data, it is often believed that only a small number of variables are truly informative while others are noise. Therefore, identifying the truly informative variables is regarded as one of the primary goals in high-dimensional data analysis as well as many real applications such as health studies.

In literature, a wide spectrum of variable selection methods have been proposed based on various model assumptions. For example, under the linear model assumption, regularized regression models are popularly used for variable selection, including the nonnegative garrote (Breiman and Friedman, 1985), the least absolute shrinkage and selection operator (Tibshirani, 1996), the smoothly clipped absolute deviation (Fan and Li, 2001), the adaptive Lasso (Zou, 2006), the combined L_0 and L_1 penalty (Liu and Wu, 2007), the truncated L_1 penalty (Shen et al., 2012), and many others. The main strategy is to associate the least square loss function with a sparsity-inducing penalty, leading to sparse representation of the resultant regression function. With the linear regression model, the sparse representation leads to variable selection based on whether the corresponding regression coefficient is zero.

The aforementioned variable selection methods have demonstrated superior performance in many real applications. Yet their success largely relies on the validity of the linear model assumption. To relax the model assumption, attempts have been made to extend the variable selection methods to a nonparametric regression context. For example, under the additive regression model assumption, a number of variable selection methods have been developed (Shively et al., 1999; Huang and Yang, 2004; Xue, 2009; Huang et al., 2010). Furthermore, higher-order additive models can be considered, allowing each functional component contain more than one variables, such as the component selection and smoothing operator (Cosso) method (Lin and Zhang, 2006). While this method provides a more flexible and still interpretable model compared to the classical additive models, the number of functional components increases exponentially with the dimension. Another stream of research on variable selection is to conduct screening (Fan et al., 2011; Zhu et al., 2011; Li et al., 2012), which treats each individual variable separately and assures the sure screening properties. To overcome the issue of ignoring interaction effects, a higher-order interaction screening method is also developed (Hao and Zhang, 2014). Model-free variable selection has also been approached in the context of sufficient dimension reduction (Li et al., 2005; Bondell and Li, 2009). More recently, Stefanski et al. (2014) introduced a novel measurement-error-model-based variable selection method that can be adapted to a nonparametric kernel regression.

In this article, we propose a novel model-free variable selection method, which requires no explicit model assumptions and allows for general variable effects. The method is based on the idea that a variable is truly informative with respect to the regression function if the gradient of the regression function along the corresponding coordinate is substantially different from zero. Thus the proposed variable selection method is formulated in a gradient learning framework equipped with a flexible reproducing kernel Hilbert space (Wahba, 1999). Learning gradients can be traced back to Härdle and Gasser (1985). Some of its recent developments include Jarrow et al. (2004), Mukherjee and Zhou (2006), Ye and Xie (2012), and Brabanter et al. (2013), where the main focus is to estimate the gradient functions.

As opposed to estimating the gradient functions, this article focuses on variable selection whose primary interest is to identify the truly informative variables corresponding to the non-zero gradient functions. To attain the sparsity in the estimated gradients, we consider a learning algorithm generated by a coefficient-based regularization scheme (Scholköpf and Smola, 2002), and a group Lasso penalty (Yuan and Lin, 2006) is enforced on the coefficients so that the proposed method can conduct gradient learning and variable selection

simultaneously. Specifically, the proposed variable selection method via gradient learning is formulated in a regularization form that consists of a pairwise loss function for estimating the gradient functions and a group Lasso penalty.

One of the main features of the proposed variable selection method is that it does not require any explicit model assumption and detect informative variables with various effects on the regression function. This is a major advantage over most existing model-based variable selection methods which need to pre-specify a working model. If higher-order variable effects are considered, the model-based methods need to enumerate the possible components, whose number increases exponentially with the dimension p. In sharp contrast, our proposed method only needs to estimate p components, while allowing for general variable effects.

Another interesting feature of the proposed method is the use of coefficient-based representation in estimating the gradient function. It follows directly from the representor theorem (Wahba, 1999) in a RKHS, and turns out to greatly facilitate variable selection in the gradient learning framework. With the coefficient-based representation, the group Lasso penalty can be naturally enforced on all the coefficients associated with the same variable. This leads to a well-structured optimization task, and can be efficiently solved through a blockwise coordinate descent algorithm (Yang and Zou, 2015). This is contrast to the existing gradient learning methods such as Ye and Xie (2012), where standard RHKS is used and a squared RKHS-norm penalty is enforced to attain the sparsity structure in the estimated gradients, and a forward-backward splitting algorithm is required for computation.

Finally, the effectiveness of the proposed method is supported by a variety of simulated and real examples. More importantly, its asymptotic estimation and selection consistencies are established, showing that the proposed method shall recover the truly informative variables with probability tending to one, and estimate the true gradient function at a fast convergence rate. Note that the variable selection consistency is not established in Ye and Xie (2012), and the estimation consistency of our method is more challenging due to the additional hypothesis error arises in the coefficient-based formulation. Also, as in many nonparametric variable selection methods (Lin and Zhang, 2006; Xue, 2009; Huang et al., 2010), the results are obtained in the sneario of fixed dimension, which are particlarly interesting given the fact that the variable selection consistency is obtained without assuming any explicit model.

The rest of the article is organized as follows. Section 2 presents a general framework of the proposed model-free variable selection method as well as an efficient computing algorithm to tackle the resultant large-scale optimization task. Section 3 establishes the asymptotic results of the proposed method in terms of both estimation and variable selection. The numerical experiments on the simulated examples and real applications are contained in Section 4. A brief discussion is provided in Section 5, and the Appendix is devoted to the technical proofs.

2. Model-free variable selection

2.1 Preambles

Suppose that a training set consists of $(\mathbf{x}_i, y_i)$; $i = 1, \ldots, n$, where $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^T \in \mathcal{R}^p$ and $y_i \in \mathcal{R}$ are independently sampled from some unknown joint distribution. We consider

the following regression model,

$$y = f^*(\mathbf{x}) + \epsilon,$$

where $\mathrm{E}(\epsilon|\,\mathbf{x}) = 0$, $\mathrm{Var}(\epsilon|\,\mathbf{x}) = \sigma^2$, $\mathbf{x} = (x^{(1)}, \ldots, x^{(p)})^T$ is supported on a compact metric space $\mathcal{X}$, and f^* is the true regression function that is assumed to be twice differentiable everywhere.

When p is large, it is generally believed that only a small number of variables are truly informative. In literature, to define the truly informative variables, f^* is often assumed to be of certain form. For instance, if $f^*(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta}^*$ with $\boldsymbol{\beta}^* = (\beta_1^*, \ldots, \beta_p^*)^T$, then $x^{(j)}$ is regarded as truly informative if $\beta_j^* \neq 0$. However, this linear model assumption on f^* can be too restrictive in practice, and whether a variable is informative shall not depend on the assumed model. In this article, a model-free variable selection method is developed without assuming any explicit form for f^*.

Since no explicit form is assumed for f^*, we note that if $x^{(l)}$ is non-informative in f^*, the corresponding gradient function $\nabla f_l^*(\mathbf{x}) = \partial f^*(\mathbf{x})/\partial x^{(l)} \equiv 0$ for any $\mathbf{x}$. This fact motivates the proposed model-free variable selection method in a gradient learning framework. Denote $\mathbf{g}^*(\mathbf{x}) = \nabla f^*(\mathbf{x}) = (\nabla f_1^*(\mathbf{x}), \ldots, \nabla f_p^*(\mathbf{x}))^T$ the true gradient function, and the estimation error as

$$\begin{aligned}\mathcal{E}(\mathbf{g}) &= E_{(\mathbf{x},y),(\mathbf{u},v)} w(\mathbf{x},\mathbf{u})(y - v - \mathbf{g}(\mathbf{x})^T(\mathbf{x}-\mathbf{u}))^2 \\ &= 2\sigma_s^2 + E_{\mathbf{x},\mathbf{u}} w(\mathbf{x},\mathbf{u})(f^*(\mathbf{x}) - f^*(\mathbf{u}) - \mathbf{g}(\mathbf{x})^T(\mathbf{x}-\mathbf{u}))^2, \end{aligned} \tag{1}$$

where $\sigma_s^2 = E_{(\mathbf{x},y),(\mathbf{u},v)}[w(\mathbf{x},\mathbf{u})(y - f^*(\mathbf{x}))^2]$ is independent of $\mathbf{g}$, and $w(\mathbf{x},\mathbf{u})$ is a weight function that decreases as $\|\,\mathbf{x}-\mathbf{u}\,\|$ increases and ensures the local neighborhood of $\mathbf{x}$ contributes more to estimating $\mathbf{g}^*(\mathbf{x})$. Typically, $w(\mathbf{x},\mathbf{u}) = e^{-\|\,\mathbf{x}-\mathbf{u}\,\|^2/\tau_n^2}$ with a pre-specified positive parameter τ_n^2, which plays a key role in the asymptotic estimation consistency and is to be elaborated.

2.2 Coefficient-based formulation

Given the training set $(\mathbf{x}_i, y_i)$; $i = 1, \ldots, n$, $\mathcal{E}(\mathbf{g})$ is approximated by its empirical version, and then the proposed variable selection method is formulated as

$$\underset{\mathbf{g}\in\mathcal{H}_K^p}{\operatorname{argmin}}\ s(\mathbf{g}) = \frac{1}{n(n-1)}\sum_{i,j=1}^{n} w_{ij}\Big(y_i - y_j - \mathbf{g}(\mathbf{x}_i)^T(\mathbf{x}_i - \mathbf{x}_j)\Big)^2 + J(\mathbf{g}), \tag{2}$$

where $w_{ij} = w(\mathbf{x}_i, \mathbf{x}_j)$, $\mathcal{H}_K$ denotes a RKHS induced by a pre-specified kernel $K(\cdot,\cdot)$, $J(\mathbf{g}) = \lambda_n \sum_{l=1}^p \pi_l J(g_l)$ is a penalty function on the complexity of $\mathbf{g}$, and π_l's are the adaptive tuning parameters to be specified. The representor theorem assures that the minimizer of (2) must be of the following coefficient-based representation,

$$g_l(\mathbf{x}) = \sum_{t=1}^{n} \alpha_t^l K(\mathbf{x}, \mathbf{x}_t);\ l = 1, \ldots, p.$$

Thanks to the explicit form of $g_l(\mathbf{x})$, it is clear that $g_l(\mathbf{x}) \equiv 0$ is equivalent to $\alpha_t^l = 0$ for all t's, or more concisely, $\|\boldsymbol{\alpha}^{(l)}\|_2 = 0$ with $\boldsymbol{\alpha}^{(l)} = (\alpha_1^l, ..., \alpha_n^l)^T$. A similar formulation connecting

between ridge regression with a coefficient-based representation and support vector machine (Cortes and Vapnik, 1995) is also established in Scholköpf and Smola (2002).

Furthermore, to exploit the sparse structure in the regression model, we propose to consider the following sparsity-inducing penalty,

$$J(g_l) = \inf\left\{\|\boldsymbol{\alpha}^{(l)}\|_2 : g_l(\cdot) = \sum_{t=1}^{n} \alpha_t^l K(\cdot, \mathbf{x}_t)\right\}. \tag{3}$$

Here the group Lasso type of penalty $\|\boldsymbol{\alpha}^{(l)}\|_2$ attains the effect of pushing all or none of α_t^l's to be exactly 0 and thus achieves the purpose of variable selection. The infimum is necessary for defining the penalty as the kernel basis $\{K(\cdot, \mathbf{x}_t)\}_{t=1}^n$ may not be linearly independent and thus the representation of g_l in $\mathcal{H}_K$ may not be unique. This penalty term differs from that in Ye and Xie (2012) in that our coefficient-based penalty does not rely on K and usually leads to sparser solutions. On the contrary, the penalty $\|g_l\|_K$ in Ye and Xie (2012) can be sensitive to the choice of K as its minimum eigenvalue can be very small. In addition, the finite dimensional hypothesis space is more flexible than the standard RHKS, and particularly the positive definite K is no longer needed. This relaxation can be critical in scenarios when such kernels are inappropriate.

With the coefficient-based representation and the group Lasso penalty, the proposed variable selection formulation can be rewritten as

$$\underset{\boldsymbol{\alpha}^{(1)},\ldots,\boldsymbol{\alpha}^{(p)}}{\operatorname{argmin}} \frac{1}{n(n-1)} \sum_{i,j=1}^{n} w_{ij}\left(y_i - y_j - \sum_{l=1}^{p} \mathbf{K}_i^T \boldsymbol{\alpha}^{(l)}(x_{il} - x_{jl})\right)^2 + \lambda_n \sum_{l=1}^{p} \pi_l \|\boldsymbol{\alpha}^{(l)}\|_2, \tag{4}$$

where $\mathbf{K}_i = (K(\mathbf{x}_i, \mathbf{x}_1), \ldots, K(\mathbf{x}_i, \mathbf{x}_n))^T$ is the i-th column of $\mathbf{K} = (K(\mathbf{x}_i, \mathbf{x}_j))_{n\times n}$, and λ_n is a tuning parameter. The infimum operator in (3) is absorbed in the minimization in (4). Clearly, (4) simplifies the original formulation (2) from a functional space to a finite-dimensional vector space. However, the vector space is of dimension np and thus still requires an efficient large-scale optimization scheme, which will be developed in the next section.

2.3 Computing algorithm

To solve (4), we develop a block coordinate descent algorithm as in Yang and Zou (2015). First, after dropping the $\boldsymbol{\alpha}$-unrelated terms, the cost function in (4) can be simplified as

$$\underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \; -\boldsymbol{\alpha}^T \mathbf{U} + \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{M} \boldsymbol{\alpha} + \lambda_n \sum_{l=1}^{p} \pi_l \|\boldsymbol{\alpha}^{(l)}\|_2, \tag{5}$$

where $\boldsymbol{\alpha}^T = \left((\boldsymbol{\alpha}^{(1)})^T, \ldots, (\boldsymbol{\alpha}^{(p)})^T\right)$, $\mathbf{U} = \frac{2}{n(n-1)} \sum_{i,j=1}^n w_{ij}\mathbf{U}_{ij}$, $\mathbf{M} = \frac{2}{n(n-1)} \sum_{i,j=1}^n w_{ij}\mathbf{M}_{ij}$,

$$\begin{aligned} \mathbf{U}_{ij} &= (y_i - y_j)(\mathbf{x}_i - \mathbf{x}_j) \otimes \mathbf{K}_i, \\ \mathbf{M}_{ij} &= \left((\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T\right) \otimes \left(\mathbf{K}_i \mathbf{K}_i^T\right), \end{aligned}$$

$\mathbf{K}_i$ is the i-th column of $\mathbf{K} = (K(\mathbf{x}_i, \mathbf{x}_j))_{n\times n}$, $\mathbf{I}_n$ is a n-dimensional identity matrix, and $\otimes$ denotes the kronecker product.

Then we update one $\boldsymbol{\alpha}^{(l)}$ at a time pretending others fixed, and the l-th subproblem becomes

$$\underset{\boldsymbol{\alpha}^{(l)}}{\operatorname{argmin}} \; L(\boldsymbol{\alpha}) + \lambda_n \pi_l \|\boldsymbol{\alpha}^{(l)}\|_2 = -\boldsymbol{\alpha}^T \mathbf{U} + \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{M} \boldsymbol{\alpha} + \lambda_n \pi_l \|\boldsymbol{\alpha}^{(l)}\|_2,$$

To solve the subproblem, a similar approximation as in Yang and Zou (2015) can be employed, where the updated $\boldsymbol{\alpha}^{(l)}$ is obtained by solving

$$\underset{\boldsymbol{\alpha}^{(l)}}{\operatorname{argmin}} \nabla_l L(\widetilde{\boldsymbol{\alpha}})(\boldsymbol{\alpha}^{(l)} - \widetilde{\boldsymbol{\alpha}}^{(l)}) + \frac{\gamma^{(l)}}{2}(\boldsymbol{\alpha}^{(l)} - \widetilde{\boldsymbol{\alpha}}^{(l)})^T(\boldsymbol{\alpha}^{(l)} - \widetilde{\boldsymbol{\alpha}}^{(l)}) + \lambda_n \pi_l \|\boldsymbol{\alpha}^{(l)}\|_2. \tag{6}$$

Here $\widetilde{\boldsymbol{\alpha}}$ is the current estimate for $\boldsymbol{\alpha}$, $\widetilde{\boldsymbol{\alpha}}^{(l)}$ is the l-th column of $\widetilde{\boldsymbol{\alpha}}$,

$$\nabla L(\widetilde{\boldsymbol{\alpha}}) = \frac{2}{n(n-1)} \sum_{i,j=1}^{n} w_{ij}(\mathbf{M}_{ij}\widetilde{\boldsymbol{\alpha}} - \mathbf{U}_{ij}),$$

$\nabla_l L(\widetilde{\boldsymbol{\alpha}})$ denotes the l-th block vector of $\nabla L(\widetilde{\boldsymbol{\alpha}})$, and

$$\nabla_l L(\widetilde{\boldsymbol{\alpha}}) = \frac{2}{n(n-1)} \sum_{i,j=1}^{n} w_{ij}\left(\sum_{s=1}^{p}((\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T)_{ls}\mathbf{K}_i\mathbf{K}_i^T\widetilde{\boldsymbol{\alpha}}^{(s)} - (y_i - y_j)(x_{il} - x_{jl})\mathbf{K}_i\right),$$

where $((\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T)_{ls}$ is the (l,s)-th entry of $(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$. Furthermore, denote $\gamma^{(l)}$ the largest eigenvalue of

$$\mathbf{M}^{(l)} = \frac{2}{n(n-1)} \sum_{i,j=1}^{n} w_{ij}((\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T)_{ll}\mathbf{K}_i\mathbf{K}_i^T,$$

which is the l-th $n \times n$ block diagonal of $\mathbf{M}$.

It is straightforward to show that (6) has an analytical solution,

$$\boldsymbol{\alpha}^{(l)} = \left(\widetilde{\boldsymbol{\alpha}}^{(l)} - \frac{\nabla_l L(\widetilde{\boldsymbol{\alpha}})}{\gamma^{(l)}}\right)\left(1 - \frac{\lambda_n \pi_l}{\|\gamma^{(l)}\widetilde{\boldsymbol{\alpha}}^{(l)} - \nabla_l L(\widetilde{\boldsymbol{\alpha}})\|_2}\right)_+. \tag{7}$$

The proposed algorithm then iteratively updates $\boldsymbol{\alpha}^{(l)}$ for $l = 1, \ldots, p, 1, \ldots$ until convergence. The algorithm is guaranteed to converge to the global minimum, since the cost function in (5) is convex and its value is decreased in each updating step. Furthermore, the computational complexity of the block coordinate descent algorithm is $O(n^2p^2D)$ with D being the number of iterations until convergence, which can be substantially less than the complexity of solving (5) with standard optimization packages.

3. Asymptotic theory

This section presents the asymptotic estimation and variable selection consistencies of the proposed model-free variable selection method. The estimation consistency assures that the distance between $\hat{\mathbf{g}}$ and $\mathbf{g}^*$ converges to 0 at a fast rate, and the variable selection consistency assures that the truly informative variables can be exactly recovered with probability

tending to 1. Both consistency results are established for fixed p. For simplicity, we assume only the first p_0 variables $x^{(1)}, \ldots, x^{(p_0)}$ are truly informative. The following technical assumptions are made.

Assumption A1. The support $\mathcal{X}$ is a non-degenerate compact subset of $\mathcal{R}^p$, and there exists a cosntatnt c_1 such that $\sup_{\mathbf{x}} \| \mathbf{H}^*(\mathbf{x}) \|_2 \leq c_1$, where $\mathbf{H}^*(\mathbf{x}) = \nabla^2 f^*(\mathbf{x})$. Also, $\sup_{\mathbf{x}} |K(\mathbf{x}, \mathbf{x})| = 1$, and the largest eigenvalue of $\mathbf{K}$ is of order $O(n^{\psi})$ with $0 \leq \psi \leq 1$.

Assumption A2. For some constants c_2 and $\theta > 0$, the probability density $p(\mathbf{x})$ exists and satisfies

$$|p(\mathbf{x}) - p(\mathbf{u})| \leq c_2 d_X(\mathbf{x}, \mathbf{u})^{\theta}, \text{ for any } \mathbf{x}, \mathbf{u} \in \mathcal{X}, \tag{8}$$

where $d_X(\cdot, \cdot)$ is the Euclidean distance on $\mathcal{X}$.

Assumption A3. There exists some constant c_4 and c_5 such that $c_4 \leq \lim_{n\to\infty} \min_{1\leq l\leq p} \pi_l \leq \lim_{n\to\infty} \max_{1\leq l\leq p_0} \pi_l \leq c_5$ and $n^{-1/2}\lambda_n \min_{l>p_0} \pi_l \to \infty$.

Assumption A4. For any $j \leq p_0$, there exists a constant t such that $\int_{\mathcal{X}\setminus\mathcal{X}_t} \|g_j^*(\mathbf{x})\|_2 d\rho_X(\mathbf{x}) > 0$, and for any $j \geq p_0 + 1$, $g_j^*(\mathbf{x}) \equiv 0$ for any $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}_t = \{\mathbf{x} \in \mathcal{X} : d_X(\mathbf{x}, \partial\mathcal{X}) < t\}$.

In Assumption A1, the compact support is assumed for the technical simplicity, which has been often used in the literature of nonparametric models (Horowitz and Mammen, 2007; Ye and Xie, 2012). The non-degenerate $\mathcal{X}$ rules out the complete multicollinearity and thus assures the unique minimizer of (4) and the true gradient function $\mathbf{g}^*(\mathbf{x})$. And $\| \mathbf{H}^*(\mathbf{x}) \|_2$ is a matrix-2 norm of $\mathbf{H}^*(\mathbf{x})$ for any given $\mathbf{x}$, denoting its largest eigenvalue. The bounded assumption on $\| \mathbf{H}^*(\mathbf{x}) \|_2$ implies that $|f^*(\mathbf{u}) - f^*(\mathbf{x}) - (\mathbf{g}^*(\mathbf{x}))^T(\mathbf{u} - \mathbf{x})| \leq c_1 \| \mathbf{u} - \mathbf{x} \|_2^2$ for any $\mathbf{u}$ and $\mathbf{x}$, which is necessary to prevent the loss function from diverging at certain values. Furthermore, for the Gaussian Kernel, the assumption for its largest eigenvalue can be verified with $\psi = 1$. (Gregory et al., 2012). Assumption A2 introduces a Lipschitz condition on the density function, assuring the smoothness of the distribution of $\mathbf{x}$. Assumption A1 and A2 also imply that the probability density $p(\mathbf{x})$ is bounded. Assumption A3 provides some guideline on setting the adaptive weights, and is satisfied with $\pi_l = \| (\tilde{\boldsymbol{\alpha}}^{(l)})^T \mathbf{K} \tilde{\boldsymbol{\alpha}}^{(l)} \|_2^{-\gamma}$ and some positive γ. For example, the initializer $\tilde{\boldsymbol{\alpha}}^{(l)}$ can be obtained by solving (4) without the Lasso penalty and $\gamma = 3 - 2\psi$, which can be verified following Lemma 1 and Theorem 14 in Mukherjee and Zhou (2006). Other consistent estimators can also be employed to initialize the weights. Assumption A4 requires that the gradient function along a truly informative variable needs to be substantially different from 0, and that along a non-informative variable is 0 everywhere.

Lemma 1 *Let $\mathbf{g}^0$ be the minimizer of $\mathcal{E}(\mathbf{g})$ over all functionals. If Assumption A1-A2 are met, then as $\tau_n^2 \to 0$, $\mathbf{g}^0(\mathbf{x})$ converges to $\mathbf{g}^*(\mathbf{x})$ in probability, and $\mathcal{E}(\mathbf{g}^0) - 2\sigma_s^2 \to 0$.*

Lemma 1 is analogous to the Fisher consistency established for margin-based classification (Lin, 2004; Liu, 2007). It shows that the error measure $\mathcal{E}(\mathbf{g})$ in (1) is reasonable in the sense that its global minimizer well approximates the true gradient function $\mathbf{g}^*$ as τ_n^2 shrinks to 0. Note that it may not appropriate to set τ_n^2 to be exactly 0 in the gradient learning framework, but a sufficient small τ_n^2 is necessary in order to assure the estimation consistency.

Theorem 2 *Suppose Assumptions A1-A4 are met. Then there exists some constant M_0 and c_6 such that with probability at least $1 - \delta$,*

$$\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \leq c_6 \sqrt{\log(4/\delta)} \Big(n^{-1/4} + n^{\frac{2\psi-1}{2}} \lambda_n^{-2} + \tau_n^{p+4} + n^{-\frac{1}{2(p+2)}} + n^{-\left(1-\frac{1}{2(p+2)}\right)} \lambda_n^2 \tau_n^{-4} \Big).$$

Theorem 2 establishes the estimation consistency of $\hat{\mathbf{g}}$ in terms of its estimation error $\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2$, which is governed by the choice of λ_n and τ_n. Specifically, let $\lambda_n = n^{\frac{2\psi-1}{4} + \frac{1}{4(p+2)}}$ and $\tau_n = n^{-\frac{\theta}{4p(p+2)(p+4+3\theta)}}$, and we have $\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \to 0$ in probability.

Next, let $\mathcal{A}_T = \{1, \cdots, p_0\}$ consist of all the truly informative variables, and $\widehat{\mathcal{A}} = \{j : \|\widehat{\boldsymbol{\alpha}}^{(j)}\|_2 > 0\}$ consist of all the estimated informative variables, where $\widehat{\boldsymbol{\alpha}}^{(j)}$ is the solution of (4).

Theorem 3 *Suppose all the assumptions in Theorem 2 are met. Let $\lambda_n = n^{\frac{2\psi-1}{4} + \frac{1}{4(p+2)}}$ and $\tau_n = n^{-\frac{\theta}{4p(p+2)(p+4+3\theta)}}$, then $P(\widehat{\mathcal{A}} = \mathcal{A}^*) \to 1$ as n diverges.*

Theorem 3 assures that the selected variables by the proposed method can exactly recover the true active set with probability tending to 1. In fact, $P(\widehat{\mathcal{A}} = \mathcal{A}^*)$ can be upper bounded by $1 - O(n^{-1/4})$ with an appropriate choice of δ. This result is particularly interesting given the fact that it is established without assuming any explicit model assumptions.

4. Numerical experiments

This section examines the effectiveness of the proposed model-free variable selection method, and compares it against some popular model based methods in literature, including variable selection with the additive model (Xue, 2009), Cosso (Lin and Zhang, 2006), sparse gradient learning (Ye and Xie, 2012) and multivariate adaptive regression splines (Friedman, 1991), denoted as MF, Add, Cosso, SGL and Mars respectively. In all the experiments, the Gaussian kernel $K(\mathbf{x}, \mathbf{u}) = e^{-\| \mathbf{x} - \mathbf{u} \|_2^2 / 2\sigma_n^2}$ is used, where the scalar parameters σ_n^2 and τ_n^2 in $w(\mathbf{x}, \mathbf{u})$ are set as the median over the pairwise distances among all the sample points (Mukherjee and Zhou, 2006). Other tuning parameters in these competitors, such as the number of knots in Xue (2009), are set as the default values in the available R and Matlab packages.

The tuning parameters in each method are determined by the stability-based selection criterion in Sun et al. (2013). The idea is to conduct a cross-validation-like scheme, and measure the stability as the agreement between two estimated active sets. It randomly splits the training set into two subsets, applies the candidate variable selection method to each subset, and obtains two estimated active sets, denoted as $\widehat{\mathcal{A}}_{1b}$ and $\widehat{\mathcal{A}}_{2b}$. The variable selection stability can approximated by $s_\lambda = \frac{1}{B} \sum_{b=1}^{B} \kappa(\widehat{\mathcal{A}}_{1b}, \widehat{\mathcal{A}}_{2b})$, where B is the number of splitting in the cross validation scheme, and $\kappa(\cdot, \cdot)$ is the standard Cohen's kappa statistic measuring the agreement between two sets. The tuning parameter λ is then selected as the one maximizing s_λ. Finally, the performance of all methods is evaluated by a number of measures regarding the variable selection accuracy.

4.1 Simulated examples

Two simulated examples are considered. The first example was used in Xue (2009) and Huang et al. (2010), where the true regression model is an additive model. The second example modifies the generating scheme of the first one and includes interaction terms.

Example 1: First generate p-dimensional variables $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^T$ with $x_{ij} = \frac{W_{ij} + \eta U_i}{1 + \eta}$, where W_{ij} and U_i are independently from $U(-0.5, 0.5)$, for $i = 1, \ldots, n$ and

$j = 1, \ldots, p$. When $\eta = 0$ all variables are independent, whereas when $\eta = 1$ correlation presents among the variables. Next, set $f^*(\mathbf{x}_i) = 5f_1(x_{i1}) + 3f_2(x_{i2}) + 4f_3(x_{i3}) + 6f_4(x_{i4})$, with $f_1(u) = u$, $f_2(u) = (2u-1)^2$, $f_3(u) = \frac{\sin(\pi u)}{2-\sin(\pi u)}$, and $f_4(u) = 0.1\sin(\pi u) + 0.2\cos(\pi u) + 0.3\sin^2(\pi u) + 0.4\cos^3(\pi u) + 0.5\sin^3(\pi u)$. Finally, generate y_i by $y_i = f(x_i) + \epsilon_i$ with $\epsilon_i \sim N(0, 1.31^2)$. Clearly, the true underlying regression model is additive.

Example 2: The generating scheme is similar as Example 1, except that $f^*(\mathbf{x}_i) = (2x_{i1} - 1)(2x_{i2} - 1)$, W_{ij} and U_i are independently from $N(0, 1)$ and $\epsilon_i \sim N(0, 1)$. It is clear that the underlying regression model includes interaction terms, and thus the additive model assumption is no longer valid.

For each example, different scenarios are considered with $\eta = 0$ or 1, and $(n, p) = (100, 10)$, $(100, 20)$ or $(200, 50)$. Each scenario is replicated 50 times, and the averaged performance measures are summarized in Tables 1 and 2. Specifically, Size represents the averaged number of selected informative variables, TP represents the number of truly informative variables selected, FP represents the number of truly non-informative variables selected and C, U, O are the times of correct-fitting, under-fitting and over-fitting, respectively.

It is evident that the proposed MF method delivers superior selection performance against the other three competitors. In Table 1 where the true model is indeed additive, MF performs similarly as Add and SGL, whereas Cosso and Mars appear more likely to overfit. In Table 2 where the true model consists of interaction terms, the performance of MF becomes competitive, but Add tends to under-fit more frequently, and Cosso, Mars and SGL tend to overfit as the dimension increases. Furthermore, in both examples with $\eta = 1$, it is clear that the correlation among variables increases the difficulty of selecting the truly informative variables, yet the proposed MF method still outperforms its competitors. Furthermore, it is also noted that the estimation accuracy of MF outperforms SGL, but it is omitted here as only MF and SGL estimate the gradient function $\mathbf{g}$, whereas Add, Cosso and Mars estimate the regression function f.

4.2 Real examples

The proposed model-free variable selection method is applied to three real data examples, the Boston housing data, the Ozone concentration data, and the digit recognition data, all of which are publicly available. The Boston housing data concerns the median value of owner-occupied homes in each of the 506 census tracts in the Boston Standard Metropolitan Statistical Area in 1970. It consists of 13 variables, including per capita crime rate by town (CRIM), proportion of residential land zoned for lots over 25,000 square feet (ZN), proportion of non-retail business acres per town (INDUS), Charles River dummy variable (CHAS), nitric oxides concentration (NOX), average number of rooms per dwelling (RM), proportion of owner-occupied units built prior to 1940 (AGE), weighted distances to five Boston employment centers (DIS), index of accessibility to radial highways (RAD), full-value property-tax rate per $10000 (TAX), pupil-teacher ratio by town (PTRATIO), the proportion of blacks by town (B), lower status of the population (LSTAT), which may affect the housing price. The Ozone concentration data concerns the daily measurements of Ozone concentration in Los Angeles basin in 330 days. The Ozone concertration may be influenced by 11 meteorological quantities, such as month (M), day of month (DM), day of week

(n, p, η)	Method	Size	TP	FP	C	U	O
(100,10,0)	MF	4.000	**4.000**	**0.000**	50	0	0
	Add	4.080	**4.000**	0.080	46	0	4
	Cosso	4.200	3.960	0.240	41	1	8
	SGL	4.020	3.600	0.420	12	20	18
	Mars	5.200	**4.000**	1.200	12	0	38
(100,20,0)	MF	4.040	**4.000**	0.300	35	0	15
	Add	4.040	**4.000**	**0.040**	48	0	2
	Cosso	4.280	**4.000**	0.280	40	0	10
	SGL	4.220	3.620	0.600	16	18	16
	Mars	6.000	**4.000**	2.000	10	0	40
(200,50,0)	MF	4.500	**4.000**	0.500	39	0	11
	Add	5.200	**4.000**	1.200	30	0	20
	Cosso	5.600	**4.000**	1.600	31	0	19
	SGL	3.600	3.400	**0.200**	12	30	8
	Mars	12.400	**4.000**	8.400	0	0	50
(100,10,1)	MF	4.160	3.800	0.360	33	10	7
	Add	3.960	3.960	**0.000**	48	2	0
	Cosso	4.200	3.760	0.440	24	8	18
	SGL	4.200	3.600	0.600	24	12	14
	Mars	5.240	**4.000**	1.240	16	0	34
(100,20,1)	MF	4.080	3.800	0.280	30	10	10
	Add	3.960	3.840	**0.120**	36	8	6
	Cosso	3.960	3.800	0.160	37	5	8
	SGL	4.020	3.500	0.520	10	20	20
	Mars	6.240	**4.000**	2.240	8	0	42
(200,50,1)	MF	4.700	3.900	**0.800**	35	5	10
	Add	5.600	**4.000**	1.600	21	0	29
	Cosso	5.000	3.900	1.100	26	4	20
	SGL	3.720	3.500	0.220	20	20	10
	Mars	13.220	**4.000**	9.220	0	0	50

Table 1: The averaged performance measures of various variable selection methods in Example 1.

(DW), Vandenburg 500 millibar height (VDHT), wind speed (WDSP), humidity (HMDT), temperature at Sandburg (SBTH), inversion base height (IBHT), Daggett pressure gradient (DGPG), inversion base temperature (IBTP) and visibility (VSTY). These two datasets have been widely analyzed in literature, including Breiman and Friedman (1985), Xue (2009), and Lin and Zhang (2006). For the digit recognition data, each digit is described by a 8×8 gray-scale image with each entry ranging from 0 to 16. We focus on digits 3 and 5 due to their similarity, and the resultant dataset consists of 365 observations and 64 attributes.

(n,p,η)	Method	Size	TP	FP	C	U	O
(100,10,0)	MF	1.960	1.920	**0.080**	43	3	4
	Add	2.140	1.760	0.380	25	9	16
	Cosso	2.920	1.920	1.000	15	3	32
	SGL	2.320	1.920	0.400	30	4	16
	Mars	4.000	**2.000**	2.000	8	0	42
(100,20,0)	MF	2.100	**2.000**	**0.100**	45	0	5
	Add	2.200	1.800	0.400	30	8	12
	Cosso	4.320	1.920	2.400	10	3	37
	SGL	2.220	**2.000**	0.220	42	0	8
	Mars	4.240	1.920	2.320	14	2	34
(200,50,0)	MF	2.100	**2.000**	0.100	45	0	5
	Add	2.920	1.920	1.000	28	2	20
	Cosso	2.200	1.800	0.400	25	10	15
	SGL	1.800	1.800	**0.000**	42	8	0
	Mars	8.200	**2.000**	6.200	0	0	50
(100,10,1)	MF	2.160	**2.000**	**0.160**	42	0	8
	Add	2.360	1.560	0.800	16	12	22
	Cosso	3.600	**2.000**	1.600	10	0	40
	SGL	2.300	**2.000**	0.300	35	0	15
	Mars	4.240	**2.000**	2.240	10	0	40
(100,20,1)	MF	2.040	1.920	**0.120**	40	4	6
	Add	2.460	1.920	0.540	34	4	12
	Cosso	3.240	1.800	1.440	9	10	31
	SGL	2.120	1.800	0.320	28	10	12
	Mars	6.740	**2.000**	4.740	0	0	50
(200,50,1)	MF	2.160	**1.960**	**0.200**	40	2	8
	Add	16.200	1.800	14.400	17	9	24
	Cosso	2.340	1.800	0.540	28	10	22
	SGL	2.460	**1.960**	0.500	23	2	25
	Mars	8.160	**1.960**	6.240	0	2	48

Table 2: The averaged performance measures of various variable selection methods in Example 2.

In our analysis, all the variables and responses are standardized and the selected variables are summarized. The selected informative variables by MF, Add, Cosso and Mars are summarized in Tables 3 and 4. As the truly informative variables are unknown in real examples, averaged prediction errors with the selected variables are also reported to compare the performance. To compute the averaged prediction error, each dataset is randomly split into two parts: m observations for testing and the remaining for training. Specifically, $m = 30$ for the Boston housing data, $m = 50$ for the Ozone concentration data, and $m = 35$ for

digit recognition data. Each example is replicated 100 times, and the averaged prediction errors by MF, Add, Cosso and Mars are summarized in Tables 3-5.

Variables	MF	Add	Cosso	SGL	Mars
CRIM	-	√	√	-	√
ZN	-	-	-	-	-
INDUS	-	-	-	-	-
CHAS	-	-	-	-	-
NOX	-	√	-	-	√
RM	√	√	√	√	√
AGE	-	-	-	-	-
DIS	-	√	-	-	√
RAD	-	-	-	-	√
TAX	-	√	-	-	-
PTRATIO	-	√	-	-	√
B	-	-	-	-	√
LSTAT	√	√	√	√	√
Pred. Err.	**1.774(0.0931)**	1.780(0.0916)	1.797(0.0924)	**1.774(0.0931)**	1.956(0.0939)

Table 3: The selected variables as well as the corresponding prediction errors by various selection methods in the Boston housing dataset.

Variables	MF	Add	Cosso	SGL	Mars
M	√	√	√	√	√
DM	-	-	-	-	-
DW	-	-	-	-	-
VDHT	-	√	-	√	√
WDSP	-	-	-	-	√
HMDT	√	-	√	√	√
SBTH	√	√	√	√	√
IBHT	-	√	-	√	√
DGPG	-	√	-	-	√
IBTP	√	-	√	√	√
VSTY	-	√	-	√	√
Pred. Err.	**1.768(0.0416)**	1.769(0.0425)	**1.768(0.0416)**	1.776(0.0426)	1.784(0.0463)

Table 4: The selected variables as well as the corresponding prediction errors by various selection methods in the Ozone concentration dataset.

	MF	Add	Cosso	SGL	Mars
No. of variables	2	48	8	4	18
Prediction error	**1.857(0.0316)**	1.871(0.0310)	1.878(0.0314)	1.875(0.0324)	1.879(0.0310)

Table 5: The number of selected variables and the prediction errors by various selection methods in the digit recognition dataset.

For the Boston housing data, MF and SGL select two informative variables, RM and LSTAT, whereas Add, Cosso and Mars tend to select more variables. However, the corresponding prediction errors of Add, Cosso and Mars appear to be larger than that of MF and SGL, implying that the additional selected variables by Add, Cosso and Mars may hinder the prediction performance. For the Ozone concentration data, both MF and Cosso

select four variables but Add and Mars select more. One discrepancy is the variable IBTP, which is selected by MF, Cosso, SGL and Mars but not by Add. As claimed in Gregory et al. (2012), M, SBTH and IBTP are three most important meteorological variables related to Ozone concentration as all of them describe the temperature changes. Meanwhile, MF and Cosso show smaller prediction error than SGL and Mars, which implies that SGL and Mars may include some non-informative variables. Figure 1 displays scatter plots of the responses against the selected variables by MF in the Boston housing data and the Ozone concentration data. It is clear that all the selected variables show moderate to strong relationship with the responses. For digit recognition data, MF selects much less variables than the other competitors and provides smaller prediction error. Figure 2 shows some randomly selected digits of 3 and 5 and the two selected informative variables, where the left informative variable is always contained in digit 5 and the right one is always contained in digit 3.

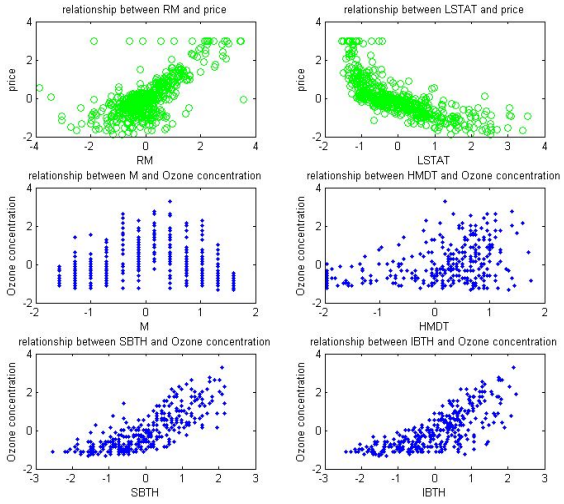


Figure 1: The scatter plots of the responses and the selected variables by MF in the Boston housing data (first row) and the Ozone concentration data (second and third rows).

Figure 2: Some randomly selected digit 3, digit 5 and the two selected informative variables.

5. Summary

This article proposes a model-free variable selection method, which is in sharp contrast to most existing methods relying on various model assumptions. The proposed method makes use of the natural connection between informative variables and sparse gradients, and formulates the variable selection task in a flexible framework of learning gradients. Additionally, we introduce a coefficient-based representation to facilitate variable selection in the learning framework. A block-wise coordinate decent algorithm is developed to make efficient computation for large-scale problems feasible. More importantly, we establish the estimation and variable selection consistencies of the proposed method without assuming any restrictive model assumption. The effectiveness of the proposed method is also supported by numerical experiments on simulated and real examples. It is worth pointing out that the computational cost of the proposed method can be expensive, as it allows for a more flexible modeling framework in RKHS. The extension of the proposed method to diverging dimension is also challenging as a model-free framework with diverging dimension can be too flexible to analyze. One possible remedy is to pre-screen the non-informative variables via some model-free screening methods (Li et al., 2012) to shrink the size of candidate variables.

Acknowledgments

SL's research is partially supported by NSFC-11301421, and JW's research is partially supported by HK GRF-11302615 and CityU SRG-7004244. The authors also thank the Action Editor and two referees for their constructive comments and suggestions, which have led to a significantly improved paper.

Appendix A. technical proofs

Proof of Lemma 1: First, note that under Assumption A1 and A2, the probability density $p(\mathbf{x})$ is bounded, and thus there exists some constant c_7 such that $\sup_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}) \le c_7$. Moreover, denote $\mathcal{X}_t = \{\mathbf{x} \in \mathcal{X} : d_X(\mathbf{x}, \partial\mathcal{X}) < t\}$, then we have $\rho_X(\mathcal{X}_t) \le c_8 t$ for any t given a constant c_8, where $\partial\mathcal{X}$ is the boundary of the compact support $\mathcal{X}$, ρ_X is the marginal distribution and $d_X(\mathbf{x}, \partial\mathcal{X}) = \inf_{\mathbf{u}\in\partial\mathcal{X}} d_X(\mathbf{x}, \mathbf{u})$.

Since $\mathbf{g}^0$ is the minimizer of $\mathcal{E}(\mathbf{g})$, the functional derivative of $\mathcal{E}(\mathbf{g})$ at $\mathbf{g}^0$ yields that for any arbitrary function vector $\delta(\mathbf{x})$,

$$\int_{\mathcal{X}}\int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})\big(f(\mathbf{x}) - f(\mathbf{u}) + \mathbf{g}^0(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\big)(\mathbf{u}-\mathbf{x})^T\delta(\mathbf{x})d\rho_X(\mathbf{u})d\rho_X(\mathbf{x}) = \mathbf{0}_p,$$

where $\mathbf{0}_p$ is a p-dimensional vector with all zeros. As the above equality is true for any $\delta(\mathbf{x})$, it implies that for any given $\mathbf{x}$,

$$\int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})\left(f(\mathbf{x}) - f(\mathbf{u}) + \mathbf{g}^0(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\right)(\mathbf{u}-\mathbf{x})d\rho_X(\mathbf{u}) = \mathbf{0}_p.$$

For simplicity, denote $\mathbf{M}(\mathbf{x}) = \int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})(\mathbf{u}-\mathbf{x})(\mathbf{u}-\mathbf{x})^T d\rho_X(\mathbf{u})$ a function matrix, and $\mathbf{d}(\mathbf{x}) = \int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})(\mathbf{u}-\mathbf{x})\left(f(\mathbf{u}) - f(\mathbf{x})\right)d\rho_X(\mathbf{u})$ a function vector. Then $\mathbf{M}(\mathbf{x})\,\mathbf{g}^0(\mathbf{x}) - \mathbf{d}(\mathbf{x}) = 0$ for any given $\mathbf{x}$. Let $\mathcal{X}_\tau = \{\mathbf{x} : d_X(\mathbf{x}, \partial\mathcal{X}) \ge \tau_n, p(\mathbf{x}) \ge c_2\tau_n^\theta + \tau_n^{1/2}\}$, then by Assumption A2,

$$P(\mathcal{X}_\tau^c) \le P\left(d_X(\mathbf{x}, \partial\mathcal{X}) < \tau_n\right) + P\left(p(\mathbf{x}) < c_2\tau_n^\theta + \tau_n^{1/2}\right) \le c_8\tau_n + (c_2\tau_n^\theta + \tau_n^{1/2})|\mathcal{X}|,$$

where $|\mathcal{X}|$ denotes the Lebesgue measure of $\mathcal{X}$. For any $\mathbf{x} \in \mathcal{X}_\tau$,

$$\begin{aligned}\mathbf{M}(\mathbf{x}) &= \int w(\mathbf{x},\mathbf{u})(\mathbf{u}-\mathbf{x})(\mathbf{u}-\mathbf{x})^T p(\mathbf{u})d\,\mathbf{u}\\ &\ge \tau_n^{1/2}\int_{d_X(\mathbf{x},\mathbf{u})<\tau_n} e^{-\frac{\|\,\mathbf{x}-\mathbf{u}\,\|_2^2}{2\tau_n^2}}(\mathbf{u}-\mathbf{x})(\mathbf{u}-\mathbf{x})^T d\,\mathbf{u} = \tau_n^{p+5/2}\int_{\|\,\mathbf{t}\,\|_2<1} e^{-\frac{\|\,\mathbf{t}\,\|_2^2}{2}}\,\mathbf{t}\,\mathbf{t}^T d\,\mathbf{t},\end{aligned}$$

where $\mathbf{t} = (\mathbf{u}-\mathbf{x})/\tau_n$. The inequality follows from Assumption A2 and the fact that $p(\mathbf{u}) \ge p(\mathbf{x}) - |p(\mathbf{u}) - p(\mathbf{x})| \ge p(\mathbf{x}) - c_2 d(\mathbf{x},\mathbf{u})^\theta \ge \tau_n^{1/2}$ on $\mathcal{X}_\tau$. As the support $\mathcal{X}$ is non-degenerate by assumption A1, $\int_{\|\,\mathbf{t}\,\|_2<1} e^{-\frac{\|\,\mathbf{t}\,\|_2^2}{2}}\,\mathbf{t}\,\mathbf{t}^T d\,\mathbf{t}$ is always positive definite. So its smallest eigenvalue, denoted as ϕ_{min}, is positive, and thus the smallest eigenvalue of $\mathbf{M}(\mathbf{x})$ must be larger than $\phi_{min}\tau_n^{p+5/2}$, which is also positive.

As $\mathbf{M}(\mathbf{x})$ is invertible for any $\mathbf{x} \in \mathcal{X}_\tau$, we have $\mathbf{g}^0(\mathbf{x}) = \mathbf{M}(\mathbf{x})^{-1}\mathbf{d}(\mathbf{x})$, and thus

$$\|\mathbf{g}^0(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})\|_2 \le \|(\mathbf{M}(\mathbf{x}))^{-1}\|_2\|\mathbf{d}(\mathbf{x}) - \mathbf{M}(\mathbf{x})\mathbf{g}^*(\mathbf{x})\|_2.$$

Furthermore,

$$\begin{aligned}\|\mathbf{d}(\mathbf{x}) - \mathbf{M}(\mathbf{x})\mathbf{g}^*(\mathbf{x})\|_2 &= \int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})(\mathbf{u}-\mathbf{x})\left(f(\mathbf{u}) - f(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\right)d\rho_X(\mathbf{u})\\ &\le \int\left|w(\mathbf{x},\mathbf{u})(\mathbf{u}-\mathbf{x})\left(f(\mathbf{u}) - f(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\right)\right|p(\mathbf{u})d\,\mathbf{u}\\ &\le c_1c_8\int w(\mathbf{x},\mathbf{u})\|\,\mathbf{u}-\mathbf{x}\,\|_2^3 d\,\mathbf{u} \le c_1c_8\tau_n^{p+3}\int e^{-\frac{\|\,\mathbf{t}\,\|_2^2}{2}}\|\,\mathbf{t}\,\|_2^3 d\,\mathbf{t}\,.\end{aligned}$$

Therefore, for any $\mathbf{x} \in \mathcal{X}_\tau$,

$$\|\mathbf{g}^0(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})\|_2 \le \|\mathbf{M}(\mathbf{x})^{-1}\|_2\|\mathbf{d}(\mathbf{x}) - \mathbf{M}(\mathbf{x})\mathbf{g}^*(\mathbf{x})\|_2 \le \frac{c_1c_8\tau_n^{1/2}}{\phi_{min}}\int e^{-\frac{\|\,\mathbf{t}\,\|_2^2}{2}}\|\,\mathbf{t}\,\|_2^3 d\,\mathbf{t},$$

which converges to 0 for any $\mathbf{x} \in \mathcal{X}_\tau$. Since $P(\mathbf{x} \in \mathcal{X}_\tau) \to 1$ as $\tau_n \to 0$, the desired result follows immediately.

Next, as $\mathcal{E}(\mathbf{g}) - 2\sigma_s^2 > 0$ for any $\mathbf{g}$, we have $0 \le \mathcal{E}(\mathbf{g}^0) - 2\sigma_s^2 \le \mathcal{E}(\mathbf{g}^*) - 2\sigma_s^2$. By Proposition 3 in Ye and Xie (2012), $\mathcal{E}(\mathbf{g}^*) - 2\sigma_s^2 \le O(\tau_n^{p+4}) \to 0$ as $\tau_n \to 0$. Therefore, $\mathcal{E}(\mathbf{g}^0) - 2\sigma_s^2 \to 0$ as $\tau_n \to 0$. ■

To proceed further, we note that the proof of Theorem 2 is substantially different from conventional error analysis as in Mukerjee and Zhou (2006) and Ye and Xie (2012). In our setting, we consider the coefficient-based space $\mathcal{H}_\mathbf{z} = \{\mathbf{g} : \mathbf{g}(x) = \sum_{i=1}^n a_i K(\mathbf{x}_i, \mathbf{x}),\ a_i \in \mathcal{R}\}$ as the candidate functional space, which depends on $\{\mathbf{x}_i\}_{i=1}^n$. One difficulty arises is that $\mathbf{g}^*$ may not be contained in $\mathcal{H}_\mathbf{z}^p$ and thus $J(\mathbf{g}^*)$ can not be defined. To circumvent this difficulty, we introduce an intermediate learning algorithm as a bridge for the error analysis, so that standard empirical process and approximation theories can be used.

Define a vector-valued functional space as $\mathcal{H}_K^p = \{\mathbf{g} = (g_1, ..., g_p)^T, g_j \in \mathcal{H}_K\}$, and $\mathcal{H}_\mathbf{z}^p = \{\mathbf{g} = (g_1, ..., g_p)^T, g_j \in \mathcal{H}_\mathbf{z}\}$. Furthermore, denote the empirical error used in our algorithm as

$$\mathcal{E}_\mathbf{z}(\mathbf{g}) = \frac{1}{n(n-1)}\sum_{i,j=1}^n \omega_{ij}\Big(y_j - y_i - \mathbf{g}(\mathbf{x}_i)^T(\mathbf{x}_j - \mathbf{x}_i)\Big)^2.$$

Clearly, $E(\mathcal{E}_\mathbf{z}(\mathbf{g})) = \mathcal{E}(\mathbf{g})$ for any $\mathbf{g} \in \mathcal{H}_K^p$.

In order to establish the consistency results, we introduce an intermediate learning algorithm,

$$\bar{\mathbf{g}} = \underset{\mathbf{g}\in\mathcal{H}_K^p}{\operatorname{argmin}}\ \frac{1}{n(n-1)}\sum_{i,j=1}^n w_{ij}\Big(y_i - y_j - \mathbf{g}(\mathbf{x}_i)(\mathbf{x}_i - \mathbf{x}_j)\Big)^2 + \rho_n\sum_{l=1}^p \pi_l\|g_l\|_K^2, \tag{9}$$

where $\rho_n = n^{-\eta}$ with $\eta = \frac{1}{4(p+2)}$. Note that (9) is a weighted version of the original gradient learning in Mukherjee and Zhou (2006). By the representor theorem, each element of $\bar{\mathbf{g}}$ in (9) has a closed solution with the form

$$\bar{g}_l = \sum_{t=1}^n \bar{\alpha}_t^l K(\mathbf{x}, \mathbf{x}_t), \quad \text{for} \quad l = 1, ..., p.$$

Denote $\bar{\boldsymbol{\alpha}}^l = (\bar{\alpha}_1^l, ..., \bar{\alpha}_n^l)^T$ satisfies the linear system

$$\rho_n\pi_l\,\mathbf{K}\,\bar{\boldsymbol{\alpha}}^l + \frac{1}{n(n-1)}\sum_{i,j=1}^n \omega_{ij}\Big(y_j - y_i - \bar{\mathbf{g}}(\mathbf{x}_i)^T(\mathbf{x}_j - \mathbf{x}_i)\Big)(\mathbf{K})_i^T[\mathbf{x}_j - \mathbf{x}_i]_l = 0, \tag{10}$$

where $(\mathbf{K})_i$ represents the i-th row of $\mathbf{K}$. Without loss of generality, we assume that $\mathbf{K}$ is invertible. In this case, we can solve for $\bar{\alpha}_t^l$ as follows:

$$\rho_n\pi_l\bar{\alpha}_t^l = -\frac{1}{n(n-1)}\sum_{j=1}^n \omega_{tj}\Big(y_j - y_t - \bar{\mathbf{g}}(\mathbf{x}_t)^T(\mathbf{x}_j - \mathbf{x}_t)\Big)[\mathbf{x}_j - \mathbf{x}_t]_l. \tag{11}$$

With these preparations, we are now in the position to decompose the excess error as follows.

Proposition 4 *Let $\varphi_0(\mathbf{z}) = \mathcal{E}_{\mathbf{z}}(\mathbf{g}^*) - \mathcal{E}(\mathbf{g}^*)$ and $\varphi_1(\mathbf{z}) = \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}})$. Then the following inequality holds for any $0 < \varepsilon \leq 1$,*

$$\mathcal{E}(\hat{\mathbf{g}}) + \lambda_n \sum_{l=1}^{p} \pi_l J(\hat{g}_l) \leq \varphi_1(\mathbf{z}) + 2\varphi_0(\mathbf{z}) + \Lambda_n(\varepsilon, \rho, K),$$

where

$$\Lambda_n(\varepsilon, \rho, K) = (1+\varepsilon)\mathcal{E}(\mathbf{g}^*) + \sum_{l=1}^{p_0} \rho_n \pi_l (\|g_l^*\|_K^2 - \|\bar{g}_l\|_K^2) + c_n^2/\varepsilon, \tag{12}$$

with $c_n = \frac{c_x p \lambda_n}{\rho_n \sqrt{n-1}}$ and $c_x \geq \sup_{\mathbf{x}} \|\mathbf{x}\|$. In the literature of statistical learning theory, $\varphi_0(\mathbf{z})$, $\varphi_1(\mathbf{z})$ are called the sample error and $\Lambda(\lambda_n)$ is the approximation error.

Proof of Proposition 4: First of all, by the Hölder inequality, it follows from 11 that:

$$J(\bar{g}_l) \leq \frac{c_x}{\rho_n \pi_l \sqrt{n-1}} \sqrt{\mathcal{E}_{\mathbf{z}}(\bar{\mathbf{g}})}, \quad l = 1, ..., p. \tag{13}$$

The above inequality in connection with the definition of $\hat{\mathbf{g}}$ yields that

$$\begin{aligned}
\mathcal{E}(\hat{\mathbf{g}}) + \lambda_n \sum_{l=1}^{p} \pi_l J(\hat{g}_l) &= \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + \lambda_n \sum_{l=1}^{p} \pi_l J(\hat{g}_l) \\
&\leq \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + \mathcal{E}_{\mathbf{z}}(\bar{\mathbf{g}}) + \lambda_n \sum_{l=1}^{p} \pi_l J(\bar{g}_l) \\
&\leq \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + \mathcal{E}_{\mathbf{z}}(\bar{\mathbf{g}}) + \Big(\frac{c_x \lambda_n}{\sqrt{n-1}} \sum_{l=1}^{p} \frac{1}{\rho_n}\Big) \sqrt{\mathcal{E}_{\mathbf{z}}(\bar{\mathbf{g}})} \\
&\leq \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + (1+\varepsilon)\mathcal{E}_{\mathbf{z}}(\bar{\mathbf{g}}) + c_n^2/\varepsilon \\
&\leq \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + (1+\varepsilon)\mathcal{E}_{\mathbf{z}}(\mathbf{g}^*) + 2\sum_{l=1}^{p} \rho_n \pi_l (\|g_l^*\|_K^2 - \|\bar{g}_l\|_K^2) + c_n^2/\varepsilon \\
&\leq \mathcal{E}(\hat{\mathbf{g}}) - \mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + (1+\varepsilon)\mathcal{E}_{\mathbf{z}}(\mathbf{g}^*) + 2\sum_{l=1}^{p_0} \rho_n \pi_l (\|g_l^*\|_K^2 - \|\bar{g}_l\|_K^2) + c_n^2/\varepsilon,
\end{aligned}$$

where the first inequality follows from the definition of $\hat{\mathbf{g}}$, the second inequality is derived based on 13, the third inequality follows from the fact $\sqrt{xy} \leq \frac{\varepsilon x + y/\varepsilon}{2}$ for any $\varepsilon > 0$, the fourth inequality follows from the definition of $\bar{\mathbf{g}}$, and the last inequality is due to the assumption that $g_l^* = 0$ for any $l > p_0$. ■

Next, For any given value $R > 0$, define the functional subspace with bounded $J(\mathbf{g})$ as

$$\mathcal{H}_R^p = \{\mathbf{g} \in \mathcal{H}_{\mathbf{z}}^p, \text{ with } J(\mathbf{g}) \leq R\},$$

and

$$\mathcal{S}(R, \lambda_n) = \sup_{\mathbf{g} \in \mathcal{H}_R^p} |\mathcal{E}(\mathbf{g}) - \mathcal{E}_{\mathbf{z}}(\mathbf{g})|.$$

Then the quantity $\mathcal{S}(R, \lambda_n)$ can be bounded using the McDiarmid's inequality (McDiarmid, 1989).

Lemma 5 *(McDiarmid's Inequality) Let $Z_1, ..., Z_n$ be independent random variables taking values in a set $\mathcal{Z}$, and assume that $\mathbf{f} : \mathcal{Z}^n \to \mathbb{R}$ satisfies*

$$\sup_{z_1, ..., z_n, z_i' \in \mathcal{Z}} |\mathbf{f}(z_1, ..., z_n) - \mathbf{f}(z_1, ..., z_i', ..., z_n)| \leq C_i,$$

for every $i \in \{1, 2, ..., n\}$. Then, for every $t > 0$,

$$\mathbb{P}\{|\mathbf{f}(z_1, ..., z_n) - \mathbb{E}\left(\mathbf{f}(z_1, ..., z_n)\right)| \geq t\} \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^n C_i^2}\right).$$

This result implies that, as soon as one has a function of n independent random variables, whose variation is bounded when only one variable is modified, the function will satisfy a Hoeffding-type inequality.

Lemma 6 *If $|y| \leq M_n$ and Assumptions A1-A3 hold, then for any constant $R > 0$ and $\varepsilon > 0$, there holds*

$$P(|\mathcal{S}(R, \lambda_n) - \mathbb{E}(\mathcal{S}(R, \lambda_n))| \geq \varepsilon) \leq 2\exp\left(-\frac{n\varepsilon^2}{8(M_n + \frac{c_x n^{\psi/2} R}{c_4 \lambda_n})^4}\right).$$

In addition, there exists a constant c_5, such that

$$P(|\mathcal{E}_{\mathbf{z}}(\mathbf{g}^*) - \mathcal{E}(\mathbf{g}^*)| \geq \varepsilon) \leq 2\exp\left(-\frac{n\varepsilon^2}{8(M_n + c_x \sum_{l=1}^p \|g_l^*\|_K)^4}\right). \tag{14}$$

Proof of Lemma 6: It suffices to verify the conditions required by the McDiarmid's inequality. For this purpose, we define $(\mathbf{x}', y')$ as a sample point drawn from the distribution $\rho(\mathbf{x}, y)$ and independent of $(\mathbf{x}_i, y_i)$. Denote by $\mathbf{z}'$ the modified training sample which is the same as $\mathbf{z}$ except that the i-th observation $(\mathbf{x}_i, y_i)$ is replaced with $(\mathbf{x}', y')$. Let $h(\mathbf{z}_i, \mathbf{z}_j) = \omega_{ij}(y_j - y_i - \mathbf{g}(\mathbf{x}_i)^T(\mathbf{x}_j - \mathbf{x}_i))^2$ with any fixed $\mathbf{g} \in \mathcal{H}_R^p$, then we decompose $\mathcal{E}_{\mathbf{z}}(\mathbf{g})$ as follows,

$$\mathcal{E}_{\mathbf{z}}(\mathbf{g}) = \frac{1}{n(n-1)} \sum_{k \neq i, j \neq i}^{n} h(\mathbf{z}_k, \mathbf{z}_j) + \frac{1}{n(n-1)} \sum_{j=1}^{n} h(\mathbf{z}_i, \mathbf{z}_j) + \frac{1}{n(n-1)} \sum_{k=1}^{n} h(\mathbf{z}_k, \mathbf{z}_i).$$

Note that if $\mathbf{z}$ is replaced by $\mathbf{z}'$, the difference between $\mathcal{E}_{\mathbf{z}}(\mathbf{g})$ and $\mathcal{E}_{\mathbf{z}'}(\mathbf{g})$ boils down to the differences between the second and third components of the above decomposition. By Assumption A3, we see that $\pi_l > c_4$ for any l. Then it follows that

$$\mathcal{E}_{\mathbf{z}}(\mathbf{g}) - \mathcal{E}_{\mathbf{z}'}(\mathbf{g}) \leq \frac{4(M_n + c_x \sum_{l=1}^p \|g_l\|_K)^2}{n} \leq \frac{4(M_n + c_x n^{\psi/2} \sum_{l=1}^p \|\boldsymbol{\alpha}^{(l)}\|_2)^2}{n} \leq \frac{4(M_n + \frac{c_x n^{\psi/2} R}{c_4 \lambda_n})^2}{n},$$

where the second inequality follows from the Hölder inequality and Assumption A1. Interchanging the roles of $\mathbf{z}$ and $\mathbf{z}'$ yields that

$$|\mathcal{E}_{\mathbf{z}}(\mathbf{g}) - \mathcal{E}_{\mathbf{z}'}(\mathbf{g})| \leq \frac{4(M_n + \frac{c_x n^{\psi/2} R}{c_4 \lambda_n})^2}{n}, \quad \forall\ \mathbf{g} \in \mathcal{H}_R^p.$$

Then applying the McDiarmid's inequality, we have

$$P(|\mathcal{S}(R,\lambda_n) - \mathbb{E}(\mathcal{S}(R,\lambda_n))| \geq \varepsilon) \leq 2\exp\left(-\frac{n\varepsilon^2}{8(M_n + \frac{c_x n^{\psi/2} R}{c_4\lambda_n})^4}\right).$$

In contrast with the first one, it is easier to obtain the second result in Lemma 6, since it only involves the fixed function $\mathbf{g}^*$. As a similar argument to the first one, we can set $C_i = \frac{4(M_n + c_x \sum_{l=1}^p \|g_l^*\|_K)^2}{n}$. Thus plugging C_i into the McDiarmid's inequality, our desired result follows immediately. ■

Proposition 7 *Assume the assumptions of Theorem 2 are met. If* $\mathcal{E}_{\mathbf{z}}(0) = \frac{1}{n(n-1)}\sum_{i,j=1}^n (y_i - y_j)^2$ *is upper bounded by* M_0*, then there exists a constant* c_9 *such that for any* $\delta \in (0,1)$*, with probability at least* $1-\delta$*,*

$$J(\hat{\mathbf{g}}) \leq c_9\sqrt{\log(4/\delta)}\left\{M_n^2 n^{-1/2} + n^{\frac{2\psi-1}{2}} M_0^2 \lambda_n^{-2} + \varepsilon\tau_n^p + \max_{l\leq p_0}\rho_n\pi_l\|g_l^* - \bar{g}_l\|_K + c_n^2/\varepsilon\right\}$$

where c_n *is defined as Proposition 4. In addition, there holds*

$$\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \leq c_9\sqrt{\log(4/\delta)}\left\{M_n^2 n^{-1/2} + n^{\frac{2\psi-1}{2}}\lambda_n^{-2} + \varepsilon\tau_n^p + \max_{l\leq p_0}\rho_n\pi_l\|g_l^* - \bar{g}_l\|_K + c_n^2/\varepsilon\right\}$$

Proof of Proposition 7: By Lemma 2 of Ye and Xie (2012), we have

$$\mathbb{E}(\mathcal{S}(R,\lambda_n)) \leq \frac{(M_n + \frac{c_x n^{\psi/2} R}{c_4\lambda_n})^2}{\sqrt{n}},$$

which, together with Lemma 6, implies that with probability at least $1-\delta$,

$$\varphi_1(\mathbf{z}) \leq |\mathcal{S}(R,\lambda_n)| \leq 3\sqrt{\frac{\log(2/\delta)}{n}}\Big(M_n + \frac{c_x n^{\psi/2} R}{c_4\lambda_n}\Big)^2. \tag{15}$$

By Proposition 4, we recall that

$$J(\hat{\mathbf{g}}) + \mathcal{E}(\hat{\mathbf{g}}) \leq \varphi_1(\mathbf{z}) + 2\varphi_0(\mathbf{z}) + \Lambda_n(\varepsilon,\rho,K),$$

where $\Lambda_n(\varepsilon,\rho,K) = (1+\varepsilon)\mathcal{E}(\mathbf{g}^*) + \sum_{l=1}^{p_0}\rho_n\pi_l(\|g_l^*\|_K^2 - \|\bar{g}_l\|_K^2) + c_n^2/\varepsilon$. In addition, note that the Hessian matrix $\mathbf{H}^*(\mathbf{x})$ of f^* is bounded uniformly on $\mathbf{x}$, it is easy to verify that

$$\mathcal{E}(\mathbf{g}^*) - 2\sigma_s^2 = O(\tau_n^{4+p}),$$

which implies that

$$\Lambda_n(\varepsilon,\rho,K) - 2\sigma_s^2 \leq O(\varepsilon\tau_n^p + \max_{l\leq p_0}\rho_n\pi_l\|g_l^* - \bar{g}_l\|_K + c_n^2/\varepsilon), \tag{16}$$

since $\sigma_s^2 = O(\tau_n^p)$ by definition.

Thus, combining 14 in Lemma 6, 15 with 16 , for some constant c_9, we have

$$\begin{aligned} & J(\hat{\mathbf{g}}) + \mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \\ \leq\ & c_9\sqrt{\log(4/\delta)}\left\{M_n^2 n^{-1/2} + n^{\frac{2\psi-1}{2}}R^2\lambda_n^{-2} + \varepsilon\tau_n^p + \max_{l\leq p_0}\rho_n\pi_l\|g_l^* - \bar{g}_l\|_K + c_n^2/\varepsilon\right\} \end{aligned}$$

with probability at least $1-\delta$. Finally, we give an explicit bound for R. Following the definition of $\hat{\mathbf{g}}$, we have

$$\mathcal{E}_{\mathbf{z}}(\hat{\mathbf{g}}) + J(\hat{\mathbf{g}}) \leq \mathcal{E}_{\mathbf{z}}(0) + J(0) \leq M_0,$$

which implies $\hat{\mathbf{g}} \in \mathcal{H}_{M_0}^p$, and thus $R = M_0$. As a consequence, the first and the second desired inequalities follow immediately after the fact that $\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \geq 0$ and $J(\hat{\mathbf{g}}) \geq 0$. ■

Proof of Theorem 2: For given constant $c_6 > 0$, $\mathcal{C}$ is denoted to be the following event,

$$\mathcal{C} = \Big\{\hat{\mathbf{g}} : \mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 > c_6\sqrt{\log(4/\delta)}\Big(n^{-1/4} + n^{\frac{2\psi-1}{2}}\lambda_n^{-2} + \varepsilon\tau_n^p + n^{-\frac{1}{2(p+2)}} + n^{-\left(1-\frac{1}{2(p+2)}\right)}\lambda_n^2/\varepsilon\Big).\Big\}.$$

Then, we split $\mathcal{C}$ into three different events as follows,

$$\begin{aligned} P(\mathcal{C}) &= P\Big(\mathcal{C}\cap\{|y| \leq n^{1/8}, U \leq M_0\}^c\Big) + P\Big(\mathcal{C}\cap\{|y| \leq n^{1/8}, U \leq M_0\}\Big) \\ &\leq P(|y| > n^{1/8}) + P(|y| \leq n^{1/8}, U > M_0) + P\Big(\mathcal{C}\cap\{|y| \leq n^{1/8}, U \leq M_0\}\Big), \end{aligned}$$

where $U = \frac{1}{n(n-1)}\sum_{i,j=1}^n (y_i - y_j)^2$ and $M_0 = 4B^2 + 2\sigma^2 + 1$ with B an upper bound of $f^*(\mathbf{x})$. The existence of B is due to the assumptions that $\mathbf{x}$ has a compact support and f^* is continuous Now we bound these three probabilities one by one.

First, by Chebyshev inequality, $P(|y| > n^{\frac{1}{8}}) = E(P(f^*(\mathbf{x}) + \epsilon > n^{\frac{1}{8}}|\,\mathbf{x})) \leq O(n^{-\frac{1}{4}})$, where the last inequality is due to bounded $f^*(\mathbf{x})$. For the second probability, note that U is a U-statistic with mean $E(U) = E(E(U|\,\mathbf{x}_i,\mathbf{x}_j)) = E(f^*(\mathbf{x}_i) - f^*(\mathbf{x}'))^2 + 2\sigma^2 \leq 4B^2 + 2\sigma^2$. By Bernstein's inequality for U-statistic (Janson, 2004),

$$P(|y| \leq n^{1/8}, U > M_0) \leq P(U > E(U) + 1 | |y| \leq n^{1/8}) \leq \exp\left\{-\frac{1}{16}n^{3/4}\right\},$$

where $(y_i - y_j)^2$ is upper bounded by $4n^{1/4}$, which completes the second term.

Now we turn to the third term. Within the set $\{|y| \leq n^{1/8}, U \leq M_0\}$, by Proposition 7,

$$\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \leq c_9\sqrt{\log(4/\delta)}\Big(n^{-1/4} + n^{\frac{2\psi-1}{2}}M_0^2\lambda_n^{-2} + \varepsilon\tau_n^p + \max_{l\leq p_0}\rho_n\pi_l\|g_l^* - \bar{g}_l\|_K + c_n^2/\varepsilon\Big).$$

With the choice of $\rho_l = n^{-\eta}$ with $\eta = \frac{1}{4(p+2)}$ for all l, we have $\|g_l^* - \bar{g}_l\|_K = O(n^{-\frac{1}{4(p+2)}})$ according to theorems 14, 17, 19 in Mukherjee and Zhou (2006). In addition, $c_n = \frac{c_x p\lambda_n}{\rho_n\sqrt{n-1}} = O(n^{-(\frac{1}{2}-\eta)}\lambda_n)$. Thus with probability at least $1-\delta$, for some constant c_6,

$$\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \leq c_6\sqrt{\log(4/\delta)}\Big(n^{-1/4} + n^{\frac{2\psi-1}{2}}\lambda_n^{-2} + \varepsilon\tau_n^p + n^{-\frac{1}{2(p+2)}} + n^{-\left(1-\frac{1}{2(p+2)}\right)}\lambda_n^2/\varepsilon\Big).$$

Specifically, with $\lambda_n = n^{\frac{2\psi-1}{4}+\frac{1}{4(p+2)}}$, $\tau_n = n^{-\frac{\theta}{4p(p+2)(p+4+3\theta)}}$ and $\varepsilon = \tau_n^4$, we have

$$\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2 \leq c_6\sqrt{\log(4/\delta)}n^{-\Theta},$$

where $\Theta = \min\Big\{\frac{\theta(p+4)}{4p(p+2)(p+4+3\theta)}, \frac{1}{2(p+2)}, \frac{p^2(p+4+3\theta)-2\theta}{2p(p+2)(p+4+3\theta)}\Big\}$. ■

Proof of Theorem 3: First, we show that $\|\hat{\boldsymbol{\alpha}}^{(l)}\|_2 = 0$ for any $l > p_0$ by contradiction. Suppose that $\|\hat{\boldsymbol{\alpha}}^{(l)}\|_2 > 0$ for some $l > p_0$. Taking the first derivative of (4) with respect to $\boldsymbol{\alpha}^{(l)}$ yields that

$$\frac{2}{n(n-1)}\sum_{i,j=1}^{n} w_{ij}(y_i - y_j - \hat{\mathbf{g}}(\mathbf{x}_i)^T(\mathbf{x}_i - \mathbf{x}_j))(x_{il} - x_{jl})\mathbf{K}_l = -\frac{\lambda_n\pi_l\hat{\boldsymbol{\alpha}}^{(l)}}{\|\hat{\boldsymbol{\alpha}}^{(l)}\|_2}. \tag{17}$$

Note that the norm of the right-hand side divided by $n^{1/2}$ is $n^{-1/2}\lambda_n\pi_l$, which diverges to ∞ by Assumption A3. Then the contradiction can be concluded by showing the norm of the left-hand side is smaller than $O(n^{1/2})$.

For simplicity, denote $\mathcal{B}_{\mathbf{z}}(\mathbf{g}) = \frac{2}{n(n-1)}\sum_{i,j=1}^{n} w_{ij}(y_i - y_j - \mathbf{g}(\mathbf{x}_i)^T(\mathbf{x}_i - \mathbf{x}_j))$. As the elements in both $\mathbf{x}$ and $\mathbf{K}_l$ are bounded by Assumption A1, it suffices to show $|\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}})|$ is bounded. Denote $\mathcal{C} = \Big\{\hat{\mathbf{g}} : |\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}})| > c_{10}\sqrt{\log(4/\delta)}(n^{-\Theta/2} + n^{\frac{\psi-1}{2}}\lambda_n^{-1} + n^{-3/8})\Big\}$. As in the proof of Theorem 2, we decompose $P(\mathcal{C})$ as

$$\begin{aligned} P(\mathcal{C}) &= P\Big(\mathcal{C}\cap\{|y| \leq n^{1/8}, U \leq M_0\}^c\Big) + P\Big(\mathcal{C}\cap\{|y| \leq n^{1/8}, U \leq M_0\}\Big) \\ &\leq P(|y| > n^{1/8}) + P(|y| \leq n^{1/8}, U > M_0) + P\Big(\mathcal{C}\cap\{|y| \leq n^{1/8}, U \leq M_0\}\Big). \end{aligned}$$

where U and M_0 are the same as in Theorem 2.

Following the proof of Theorem 2, the first two probabilities can be bounded as $P(|y| > n^{1/8}) \leq O(n^{-1/4})$ and $P(|y| \leq n^{1/8}, U > M_0) \leq \exp\{-\frac{1}{16}n^{3/4}\}$. To bound the third probability, a slight modification of the proof of Proposition 7 yields that when $|y| \leq n^{1/8}$ and $U \leq M_0$, we have $J(\hat{\mathbf{g}}) \leq M_0$ and with probability at least $1 - \delta/2$,

$$|\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}}) - E(\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}})| \leq 3\sqrt{\frac{\log(4/\delta)}{n}}\Big(n^{1/8} + \frac{c_x n^{\psi/2}M_0}{c_4\lambda_n}\Big).$$

We then bound $|E(\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}})|$ as follows,

$$\begin{aligned} |E(\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}}))| &= \Big|\int_{\mathcal{X}}\int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})\Big(f^*(\mathbf{x}) - f^*(\mathbf{u}) + \hat{\mathbf{g}}(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\Big)d\rho_X(\mathbf{x})d\rho_X(\mathbf{u})\Big| \\ &\leq \Big(\int_{\mathcal{X}}\int_{\mathcal{X}}\Big|w(\mathbf{x},\mathbf{u})\Big(f^*(\mathbf{x}) - f^*(\mathbf{u}) + \hat{\mathbf{g}}(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\Big)\Big|^2 d\rho_X(\mathbf{x})d\rho_X(\mathbf{u})\Big)^{1/2} \\ &\leq \Big(\int_{\mathcal{X}}\int_{\mathcal{X}} w(\mathbf{x},\mathbf{u})\Big(f^*(\mathbf{x}) - f^*(\mathbf{u}) + \hat{\mathbf{g}}(\mathbf{x})^T(\mathbf{u}-\mathbf{x})\Big)^2 d\rho_X(\mathbf{x})d\rho_X(\mathbf{u})\Big)^{1/2} \\ &= \Big(\mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2\Big)^{1/2}. \end{aligned}$$

The first inequality follows from Hölder inequality, and the second inequality follows from the fact that $w(\mathbf{x},\mathbf{u}) \leq 1$ for any $\mathbf{x}$ and $\mathbf{u}$. Therefore, within the set $\{|y| \leq n^{1/8}, U \leq M_0\}$,

we have with probability at least $1 - \delta/2$ that

$$|\mathcal{B}_{\mathbf{z}}(\hat{\mathbf{g}})| \leq c_{10}\sqrt{\log(4/\delta)}(n^{-\Theta/2} + n^{\frac{\psi-1}{2}}\lambda_n^{-1} + n^{-3/8}). \tag{18}$$

This implies that $P(\mathcal{C}) \leq \delta/2 + O(n^{-1/4})$ for any given δ. Then it is clear that the norm of the left-hand side of (17) divided by $n^{1/2}$ will converge to 0 in probability, which contradicts with the fact that the norm of the right-hand side divided by $n^{1/2}$ diverges to ∞. Therefore, we have $\|\hat{\boldsymbol{\alpha}}^{(l)}\|_2 = 0$ for any $l > p_0$.

Next, we show that $\|\hat{\boldsymbol{\alpha}}^{(l)}\|_2 > 0$ for any $l \leq p_0$. Let $\bar{\mathcal{X}}_\tau = \{\mathbf{x} \in \mathcal{X} : d(\mathbf{x}, \partial\mathcal{X}) > \tau_n, p(\mathbf{x}) > \tau_n + c_2\tau_n^\theta\}$. Same as the proof of Theorem 5 in Ye and Xie (2012), for some given constant c_{11}, we have

$$\int_{\bar{\mathcal{X}}_\tau} \|\hat{\mathbf{g}}(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})\|_2^2 d\rho_X(\mathbf{x}) \leq c_{11}\tau_n^{-(p+3)}\Big(\tau_n^{p+4} + \mathcal{E}(\hat{\mathbf{g}}) - 2\sigma_s^2\Big).$$

According to Theorem 2, it can be showed that

$$\int_{\bar{\mathcal{X}}_\tau} \|\hat{\mathbf{g}}(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})\|_2^2 d\rho_X(\mathbf{x}) \to 0.$$

Now suppose $\|\hat{\boldsymbol{\alpha}}^{(l)}\|_2 = 0$ for some $l \leq p_0$, which implies

$$\int_{\bar{\mathcal{X}}_\tau} \|\hat{\mathbf{g}}(\mathbf{x}) - \mathbf{g}^*(\mathbf{x})\|_2^2 d\rho_X(\mathbf{x}) = \int_{\bar{\mathcal{X}}_\tau} \|g_l^*(\mathbf{x})\|_2^2 d\rho_X(\mathbf{x}).$$

As $\tau_n \to 0$, $\int_{\bar{\mathcal{X}}_\tau} \|g_l^*(\mathbf{x})\|_2^2 d\rho_X(\mathbf{x}) \geq \int_{\mathcal{X}\setminus\mathcal{X}_l} \|g_l^*(\mathbf{x})\|_2^2 d\rho_X(\mathbf{x})$, which is a positive constant by Assumption A4, and then leads to the contradiction. Combining the above two statements implies the desired variable selection consistency. ■

References

H., Bondell and L., Li. Shrinkage inverse regression estimation for model free variable selection. *Journal of the Royal Statistical Society, Series B.*, 71: 287-299, 2009.

K. Brabanter, J. Brabanter and B. Moor. Derivative estimation with local polynomial fitting. *Journal of Machine Learning Research*, 14: 281-301, 2014.

L. Breiman. Better subset regresson using nonnegative garrote. *Technometrics*, 37: 373-384, 1995.

L. Breiman and J. Friedman. Estimating optimal transformations for multiple regression and correlation. *Journal of the American Statistical Association*, 80: 580-598, 1985.

C. Cortes and V. Vapnik. Support vector networks. *Journal of Machine Learning Research*, 20: 273-297, 1995.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96: 1348-1360, 2001.

J. Fan, F. Yang and R. Song. Nonparametric independence screening in sparse ultrahigh dimensional additive models. *Journal of the American Statistical Association*, 106: 544-557, 2011.

J. Friedman. Multivariate Adaptive Regression Splines. *The Annal of Statistics*, 19: 1-67, 1991.

E. Gregory, J. Fred and W. Henryk. On dimension-independent rates of convergence for function approximation with Gaussian Kernels. *SIAM Journal on Numerical Analysis*, 50: 247-271, 2012.

R. Genuer, J. Poggi and C. Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31: 2225-2236, 2010.

N. Hao and H. Zhang. Interaction screening for ultrahigh-dimensional data. *Journal of the American Statistical Association*, 109: 1285-1301, 2014.

W. Härdle and T. Gasser. On robust kernel estimation of derivatives of regression functions. *Scandinavian Journal of Statistics*, 12: 233-240, 1985.

J. L. Horowitz and E. Mammen. Rate-optimal estimation for a general class of nonparametric regression models with unknown link functions. *Annal of Statistics*, 35: 2589-2619, 2007.

J. Huang, J. Horowitz and F. Wei. Variable selection in nonparameteric additive models. *The Annals of Statistics*, 38: 2282-2313, 2010.

J. Huang, and L. Yang. Identification of non-linear additive autoregressive models. *Journal of the Royal Statistical Society: Series B*, 66: 463-477, 2004.

S. Janson. Large deviations for sums of partly dependent random variables. *Random Structures and Algorithms*, 24: 234-248, 2004.

R. Jarrow, D. Ruppert and Y. Yu. Estimating the term structure of corporate debt with a semiparametric penalized spline model. *Journal of the American Statistical Association*, 99: 57-66, 2004.

L. Li, D. Cook and C. Nachtsheim. Model-free variable selection. *Journal of the Royal Statistical Society, Series B.*, 67: 285-299, 2005.

R. Li, W. Zhong and L. Zhu. Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107: 1129-1139, 2012.

Y. Lin. A note on margin-based loss functions in classification. *Statistics and Probability Letters*, 68: 73-82, 2004.

Y. Lin and H. Zhang. Component selection and smoothing in multivariate non-parametric regression. *Annal of Statistics*, 34: 2272-2297, 2006.

Y. Liu. Fisher consistency of multicategory support vector machines. *Eleventh International Conference on Artificial Intelligence and Statistics*, 289-296, 2006.

Y. Liu and Y. Wu. Variable selection via a combination of the L0 and L1 penalties. *Journal of Computational and Graphical Statistics*, 16: 782-798, 2007.

C. McDiarmid,. On the method of bounded differences. *In Surveys in Combinatorics*, 141: 148-188.

S. Mukherjee and D. Zhou. Learning coordinate covariates via gradient. *Journal of Machine Learning Research*, 7: 519-549, 2006.

B. Scholköpf and A.J. Smola. Learning with kernels: Support vector machines, regularization, optimization, and beyond. MIT Press, 2002.

X. Shen, W. Pan and Y. Zhu. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107: 223-232, 2012.

T. Shively, R. Kohn and S. Wood. Variable selection and function estimation in additive non-parametric regression using a data-based prior. *Journal of the American Statistical Association*, 94: 777-794, 1999.

L. Stefanski, Y. Wu and K. White. Variable selection in nonparametric classification via measurement error model selection likelihoods. *Journal of the American Statistical Association*, 109: 574-589, 2014.

W. Sun, J. Wang and Y. Fang. Consistent selection of tuning parameters via variable selection stability. *Journal of Machine Learning Research*, 14: 3419-3440, 2013.

R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society: Series B*, 58: 267-288, 1996.

G. Wahba. Support vector machines, reproducing kernel hilbert spaces, and randomized GACV. *Advances in kernel methods*, 69-88, 1999.

L. Xue. Consistent variable selection in additive models. *Statistica Sinica*, 19: 1281-1296, 2009.

Y. Yang and H. Zou. A fast unified algorithm for solving group-lasso penalized learning problems. *Statistics and Computing*, 25: 1129-1141, 2015

G. Ye and X. Xie. Learning sparse gradients for variable selection and dimension reduction. *Journal of Machine Learning Research*, 87: 303-355, 2012.

M. Yuan and Y. Lin. Model selection and estimation in regression with group variables. *Journal of the Royal Statistical Society: Series B*, 68: 49-67, 2006

L. Zhu, L. Li, R. Li and L. Zhu. Model-free feature screening for ultra-high dimensional data. *Journal of the American Statistical Association*, 106: 1464-1475, 2011.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101: 1418-1429.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 8/15; Published 4/16

CVXPY: A Python-Embedded Modeling Language for Convex Optimization

Steven Diamond DIAMOND@CS.STANFORD.EDU
Stephen Boyd BOYD@STANFORD.EDU
Departments of Computer Science and Electrical Engineering
Stanford University
Stanford, CA 94305, USA

Editor: Antti Honkela

Abstract

CVXPY is a domain-specific language for convex optimization embedded in Python. It allows the user to express convex optimization problems in a natural syntax that follows the math, rather than in the restrictive standard form required by solvers. CVXPY makes it easy to combine convex optimization with high-level features of Python such as parallelism and object-oriented design. CVXPY is available at `http://www.cvxpy.org/` under the GPL license, along with documentation and examples.

Keywords: convex optimization, domain-specific languages, Python, conic programming, convexity verification

1. Introduction

Convex optimization has many applications to fields as diverse as machine learning, control, finance, and signal and image processing (Boyd and Vandenberghe, 2004). Using convex optimization in an application requires either developing a custom solver or converting the problem into a standard form. Both of these tasks require expertise, and are time-consuming and error prone. An alternative is to use a domain-specific language (DSL) for convex optimization, which allows the user to specify the problem in a natural way that follows the math; this specification is then automatically converted into the standard form required by generic solvers. CVX (Grant and Boyd, 2014), YALMIP (Lofberg, 2004), QCML (Chu et al., 2013), PICOS (Sagnol, 2015), and Convex.jl (Udell et al., 2014) are examples of such DSLs for convex optimization.

CVXPY is a new DSL for convex optimization. It is based on CVX (Grant and Boyd, 2014), but introduces new features such as signed disciplined convex programming analysis and parameters. CVXPY is an ordinary Python library, which makes it easy to combine convex optimization with high-level features of Python such as parallelism and object-oriented design.

CVXPY has been downloaded by thousands of users and used to teach multiple courses (Boyd, 2015). Many tools have been built on top of CVXPY, such as an extension for stochastic optimization (Ali et al., 2015).

2. CVXPY Syntax

CVXPY has a simple, readable syntax inspired by CVX (Grant and Boyd, 2014). The following code constructs and solves a least squares problem where the variable's entries are constrained to be between 0 and 1. The problem data $A \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^m$ could be encoded as NumPy ndarrays or one of several other common matrix representations in Python.

```
# Construct the problem.
x = Variable(n)
objective = Minimize(sum_squares(A*x - b))
constraints = [0 <= x, x <= 1]
prob = Problem(objective, constraints)

# The optimal objective is returned by prob.solve().
result = prob.solve()
# The optimal value for x is stored in x.value.
print x.value
```

The variable, objective, and constraints are each constructed separately and combined in the final problem. In CVX, by contrast, these objects are created within the scope of a particular problem. Allowing variables and other objects to be created in isolation makes it easier to write high-level code that constructs problems (see §6).

3. Solvers

CVXPY converts problems into a standard form known as conic form (Nesterov and Nemirovsky, 1992), a generalization of a linear program. The conversion is done using graph implementations of convex functions (Grant and Boyd, 2008). The resulting cone program is equivalent to the original problem, so by solving it we obtain a solution of the original problem.

Solvers that handle conic form are known as cone solvers; each one can handle combinations of several types of cones. CVXPY interfaces with the open-source cone solvers CVXOPT (Andersen et al., 2015), ECOS (Domahidi et al., 2013), and SCS (O'Donoghue et al., 2016), which are implemented in combinations of Python and C. These solvers have different characteristics, such as the types of cones they can handle and the type of algorithms employed. CVXOPT and ECOS are interior-point solvers, which reliably attain high accuracy for small and medium scale problems; SCS is a first-order solver, which uses OpenMP to target multiple cores and scales to large problems with modest accuracy.

4. Signed DCP

Like CVX, CVXPY uses disciplined convex programming (DCP) to verify problem convexity (Grant et al., 2006). In DCP, problems are constructed from a fixed library of functions with known curvature and monotonicity properties. Functions must be composed according to a simple set of rules such that the composition's curvature is known. For a visualization of the DCP rules, visit `dcp.stanford.edu`.

CVXPY extends the DCP rules used in CVX by keeping track of the signs of expressions. The monotonicity of many functions depends on the sign of their argument, so keeping track of signs allows more compositions to be verified as convex. For example, the composition `square(square(x))` would not be verified as convex under standard DCP because the `square` function is nonmonotonic. But the composition is verified as convex under signed DCP because `square` is increasing for nonnegative arguments and `square(x)` is nonnegative.

5. Parameters

Another improvement in CVXPY is the introduction of parameters. Parameters are constants whose symbolic properties (*e.g.*, dimensions and sign) are fixed but whose numeric value can change. A problem involving parameters can be solved repeatedly for different values of the parameters without redoing computations that do not depend on the parameter values. Parameters are an old idea in DSLs for optimization, appearing in AMPL (Fourer et al., 2002).

A common use case for parameters is computing a trade-off curve. The following code constructs a LASSO problem (Boyd and Vandenberghe, 2004) where the positive parameter γ trades off the sum of squares error and the regularization term. The problem data are $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$.

```
x = Variable(n)
gamma = Parameter(sign="positive") # Must be positive due to DCP rules.
error = sum_squares(A*x - b)
regularization = norm(x, 1)
prob = Problem(Minimize(error + gamma*regularization))
```

Computing a trade-off curve is trivially parallelizable, since each problem can be solved independently. CVXPY can be combined with Python multiprocessing (or any other parallelism library) to distribute the trade-off curve computation across many processes.

```
# Assign a value to gamma and find the optimal x.
def get_x(gamma_value):
    gamma.value = gamma_value
    result = prob.solve()
    return x.value

# Get a range of gamma values with NumPy.
gamma_vals = numpy.logspace(-4, 6)
# Do parallel computation with multiprocessing.
pool = multiprocessing.Pool(processes = N)
x_values = pool.map(get_x, gamma_vals)
```

6. Object-Oriented Convex Optimization

CVXPY enables an object-oriented approach to constructing optimization problems. As an example, consider an optimal flow problem on a directed graph $G = (V, E)$ with vertex

set V and (directed) edge set E. Each edge $e \in E$ carries a flow $f_e \in \mathbf{R}$, and each vertex $v \in V$ has an internal source that generates $s_v \in \mathbf{R}$ flow. (Negative values correspond to flow in the opposite direction, or a sink at a vertex.) The (single commodity) flow problem is (with variables f_e and s_v)

$$\begin{array}{ll} \text{minimize} & \sum_{e\in E} \phi_e(f_e) + \sum_{v\in V} \psi_v(s_v), \\ \text{subject to} & s_v + \sum_{e\in I(v)} f_e = \sum_{e\in O(v)} f_e, \quad \text{for all } v \in V, \end{array}$$

where the ϕ_e and ψ_v are convex cost functions and $I(v)$ and $O(v)$ give vertex v's incoming and outgoing edges, respectively.

To express the problem in CVXPY, we construct vertex and edge objects, which store local information such as optimization variables, constraints, and an associated objective term. These are exported as a CVXPY problem for each vertex and each edge.

```
class Vertex(object):
    def __init__(self, cost):
        self.source = Variable()
        self.cost = cost(self.source)
        self.edge_flows = []

    def prob(self):
        net_flow = sum(self.edge_flows) + self.source
        return Problem(Minimize(self.cost), [net_flow == 0])

class Edge(object):
    def __init__(self, cost):
        self.flow = Variable()
        self.cost = cost(self.flow)

    def connect(self, in_vertex, out_vertex):
        in_vertex.edge_flows.append(-self.flow)
        out_vertex.edge_flows.append(self.flow)

    def prob(self):
        return Problem(Minimize(self.cost))
```

The vertex and edge objects are composed into a graph using the edges' `connect` method. To construct the single commodity flow problem, we sum the vertices and edges' local problems. (Addition of problems is overloaded in CVXPY to add the objectives together and concatenate the constraints.)

```
prob = sum([object.prob() for object in vertices + edges])
prob.solve() # Solve the single commodity flow problem.
```

Acknowledgments

We thank the many contributors to CVXPY. This work was supported by DARPA XDATA.

References

A. Ali, Z. Kolter, S. Diamond, and S. Boyd. Disciplined convex stochastic programming: A new framework for stochastic optimization. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 62–71, 2015.

M. Andersen, J. Dahl, and L. Vandenberghe. CVXOPT: Python software for convex optimization, version 1.1. `http://cvxopt.org/`, May 2015.

S. Boyd. EE364a: Convex optimization I. `http://stanford.edu/class/ee364a/`, December 2015.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

E. Chu, N. Parikh, A. Domahidi, and S. Boyd. Code generation for embedded second-order cone programming. In *Proceedings of the European Control Conference*, pages 1547–1552, 2013.

A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *Proceedings of the European Control Conference*, pages 3071–3076, 2013.

R. Fourer, D. Gay, and B. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Cengage Learning, 2002.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer, 2008.

M. Grant and S. Boyd. CVX: MATLAB software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

M. Grant, S. Boyd, and Y. Ye. Disciplined convex programming. In L. Liberti and N. Maculan, editors, *Global Optimization: From Theory to Implementation*, Nonconvex Optimization and its Applications, pages 155–210. Springer, 2006.

J. Lofberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Proceedings of the IEEE International Symposium on Computed Aided Control Systems Design*, pages 294–289, 2004.

Y. Nesterov and A. Nemirovsky. Conic formulation of a convex programming problem and duality. *Optimization Methods and Software*, 1(2):95–115, 1992.

B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, pages 1–27, 2016.

G. Sagnol. PICOS: A Python interface for conic optimization solvers, version 1.1. `http://picos.zib.de/index.html`, April 2015.

M. Udell, K. Mohan, D. Zeng, J. Hong, S. Diamond, and S. Boyd. Convex optimization in Julia. In *Proceedings of the Workshop for High Performance Technical Computing in Dynamic Languages*, pages 18–28, 2014.

Journal of Machine Learning Research 17 (2016) 1-42 Submitted 8/15; Revised 4/16; Published 5/16

Lenient Learning in Independent-Learner Stochastic Cooperative Games

Ermo Wei EWEI@CS.GMU.EDU
Sean Luke SEAN@CS.GMU.EDU
Department of Computer Science
George Mason University
4400 University Drive MSN 4A5
Fairfax, VA 22030, USA

Editor: Kevin Murphy

Abstract

We introduce the *Lenient Multiagent Reinforcement Learning 2* (LMRL2) algorithm for independent-learner stochastic cooperative games. LMRL2 is designed to overcome a pathology called *relative overgeneralization*, and to do so while still performing well in games with stochastic transitions, stochastic rewards, and miscoordination. We discuss the existing literature, then compare LMRL2 against other algorithms drawn from the literature which can be used for games of this kind: traditional ("Distributed") Q-learning, Hysteretic Q-learning, WoLF-PHC, SOoN, and (for repeated games only) FMQ. The results show that LMRL2 is very effective in both of our measures (complete and correct policies), and is found in the top rank more often than any other technique. LMRL2 is also easy to tune: though it has many available parameters, almost all of them stay at default settings. Generally the algorithm is optimally tuned with a single parameter, if any. We then examine and discuss a number of side-issues and options for LMRL2.

Keywords: multiagent learning, reinforcement learning, game theory, lenient learning, independent learner

1. Introduction

In a cooperative game, as we use the term here, some $N \geq 2$ players simultaneously choose an action from among those actions each is permitted to make, and then each receives the same (potentially stochastic) reward arising from their joint action. It is this equally-shared reward that makes the game cooperative: agents succeed or fail together. Such cooperative games are very common in multiagent systems: for example, a group of robots might work together to map out a room; a swarm of game agents might collaborate to defeat a hard-coded Bad Guy; or a team of wireless beacons might work together to form an optimal ad-hoc routing network.

We are interested in the situation where the agents in question do not know the reward function over their joint actions and must *learn* to collaboratively explore this space and ultimately adapt to the highest-reward joint action. Furthermore, the agents are *independent learners* in the sense that they do not know what actions the other agents are performing (although they may assume that other agents *exist*). In a primary alternative, where the agents are told the others' action choices along with the rewards received, the agents are known as *joint-action learners*.

From a game theoretic perspective, there are two common types of games in which such agents may learn to optimize their actions. First, a game may be *repeated*, meaning that the agents play the same game over and over again, potentially choosing new actions each time. Second, there is the so-called *stochastic* game, where not only does a joint action affect the reward received, but it also changes the game played next time. In a stochastic game there is a set of possible sub-games (or *states*) for the agents to play, each with potentially different kinds of actions available to the agents, and each with their own reward function. Furthermore, each joint action in each state is associated with a transition function which maps joint actions to distributions over states: from the appropriate distribution the next state will be selected. There is a distinguished initial state for the game, and there may optionally be a special end state: the game simply terminates when it reaches this state. The agents know in which state they are playing, but not its reward function nor its transition functions. A repeated game is just a stochastic game with only one state.

There is significant literature in applying multiagent versions of reinforcement learning, policy search, or similar methods to repeated and stochastic games. Much of the literature has focused on general-sum or zero-sum learners, and the remainder has focused nearly entirely on the pathology of *miscoordination* in cooperative scenarios, as discussed later. But there is another critical pathology which arises specially in cooperative games, known as *relative overgeneralization*. Overcoming relative overgeneralization may require a more explorative approach than simple epsilon-greedy action selection: indeed as shown by Wiegand (2004), relative overgeneralization can cause players to not just hill-climb into local optima but get actively sucked into them despite action selection with a high degree of randomness. We discuss these pathologies in more detail in Section 3.

Perhaps because relative overgeneralization as a phenomenon necessitates at least three actions available per player, and ideally more, it has not shown up much in the small problems common to the multiagent reinforcement learning (MARL) literature, though it has been studied at some length in the related field of cooperative co-evolutionary algorithms (Panait, 2006; Panait et al., 2006a, 2008, 2004), which involve very large numbers of actions. We are interested in solving both relative overgeneralization and miscoordination. To this end we present and discuss a *lenient* learning algorithm meant for independent learning in stochastic cooperative games (and also repeated cooperative games) with any number $N \geq 2$ of agents. We call this algorithm *Lenient Multiagent Reinforcement Learning 2*, or LMRL2 for short. This is a heavy revision of the original LMRL algorithm, which was originally meant only for repeated games (Panait et al., 2006b).

As we will show, LMRL2 performs well across a spectrum of stochastic and repeated games when compared to other algorithms from the literature, and is able to deal robustly with overgeneralization, miscoordination, and high degrees of stochasticity in the reward and transition functions with relatively little parameter tuning. In this paper we will discuss issues in MARL which lenient learning is meant to address, then survey previous work and alternative approaches. We will then detail LMRL2, compare it to other methods, and discuss various side issues which arise.

2. Previous Work

Reinforcement Learning is a popular approach to solving multiagent learning problems (Busoniu et al., 2008), because many such problems may be readily cast into normal-form games, which themselves map straightforwardly into Markov Decision Processes (MDPs) for which reinforcement learning is well suited. Use of reinforcement learning in the multiagent context is, not suprisingly, known as *Multiagent Reinforcement Learning*, or MARL.

Many MARL methods may be organized according to three different characteristics. First, there is the *reward structure* involved: MARL algorithms may be designed for zero-sum (or constant-sum), cooperative, and general-sum games. Second, there is the *type of information* being provided to the algorithms. Here we distinguish between *joint action* and *independent learner* algorithms, following Claus and Boutilier (1998). Third, there is the *game type*: some MARL algorithms are meant only for repeated games, while others are meant for more general stochastic games. LMRL2 itself is an independent learner, cooperative, stochastic game algorithm.

Joint Action Learners In a *joint action learner* scenario, after playing a game, each agent receives a reward, is told which game is to be played next (the next *state*), and is also told what actions were chosen by the other agents. There is significant joint action learner literature.

Only a small portion of the literature focuses on pure zero-sum games. One of the first zero-sum joint-action-learning algorithms was Minimax-Q (Littman, 1994), which assumes that the alternative agent is attempting to minimize one's own reward. Thus, rather than select the action which with the maximum reward achievable, Minimax-Q uses a minimax approach: it selects the highest reward achievable assuming that the alternative agent will select its action so as to minimize that reward. This solution may be determined by solving a linear program.

There are many general-sum joint-action learners. However, learning in general-sum games is very complicated due to the wide variety in reward structure. Even the goal of learning in this kind of game can be hard to define in some cases (Shoham et al., 2004). To simplify the issue, much of the literature simply aims to converge to an equilibrium of some sort during self-play (that is, playing against other agents who use the same algorithm as yourself). The most well-known algorithm of this kind is called Nash Q-Learning (Hu and Wellman, 2003), which has been proven to converge to the Q-values in some Nash Equilibrium under very restricted conditions, such as if each state has a global joint optimum. Other learning algorithms which have been designed to converge to equilibria include Wellman and Hu (1998); Greenwald et al. (2003); Jafari et al. (2001). Rather than convergence, some work has instead proposed playing a best response against some restricted classes of opponents (Weinberg and Rosenschein, 2004; Tesauro, 2004). There has also been work in combining the two, where the learner converges to Nash Equilibrium in self-play, and plays a best response when not in self-play (Conitzer and Sandholm, 2007).

Like the zero-sum game literature, the pure cooperative game literature is limited. Littman (2001b) introduced Team Q-learning, but as it is a straightforward extension of Q-Learning to cooperative games, it fails to handle miscoordination. To address this Wang and Sandholm proposed OAL (2002), which is guaranteed to converge to the optimal Nash Equilibrium in cooperative games: but due to necessary constraints on the reward function, converging to the optimal policy is not particularly interesting for joint action learners in cooperative games. Other work has imported techniques from the reinforcement learning community to enhance online performance. Notably Chalkiadakis and Boutilier applied Bayesian Reinforcement Learning to the multiagent scenario (2003). Here, agents have some prior knowledge on distributions of the game model and also the possible strategies that can be used by other learners. Each iteration of the game, an agent chooses a best action with respect to the distributions it is maintaining, and the distributions are then updated based on the individual actions of the other agents and the results from the joint action. Typical of Bayesian Reinforcement Learning methods, a bad prior can make the algorithm converge to a suboptimal policy in some cases, but Bayesian methods in general may help agents accumulate more reward while learning. Bayesian methods have also been used with policy search in a MARL context (Wilson et al., 2010), where the distributions are instead over policy parameters and possible roles of other agents. Unusually, agents here make decisions sequentially rather than simultaneously, which is rare in the MARL literature.

Another way to incorporate prior knowledge into learning is to construct the policy as a hierarchy. Hierarchical reinforcement learning has been shown to accelerate learning in the single agent case by allowing state abstraction and by reusing learned policy (Dietterich, 2000). This has been extended to the multiagent setting (Makar et al., 2001; Ghavamzadeh et al., 2006) by manually specifying the cooperation task at a higher level in the task hierarchy. This allows non-cooperative subtasks to be learned separately, and efficiently as single-agent learning problems. The authors have further used this idea in a setting where knowing the actions of other agents incurs a communication cost (Ghavamzadeh and Mahadevan, 2004).

Independent Learners An *independent learner* scenario differs from a joint action scenario in that, after playing a game, each agent is *not* told what action was chosen by each of the other agents. He is only told the reward he received and the game (state) to be played next time. This is a much more difficult learning problem. Additionally, because they are designed for a more general scenario, independent learners may participate in joint action games, but not the other way around.

Perhaps the best-known example of a general-sum independent learner is WoLF-PHC (Bowling and Veloso, 2001b, 2002). This algorithm is designed to meet two criteria for learning in general sum games: first, a learner should learn to play optimally against stationary opponents, and second, a learner should converge to a Nash Equilibrium when in self-play. WoLF-PHC is an independent-learning policy search approach which uses one of two different learning rates depending on whether the player is in some sense "winning" or "losing". This "WoLF" (Win or Learn Fast) principle has since been applied to other algorithms (Bowling and Veloso, 2001a; Banerjee and Peng, 2003; Bowling, 2005).

Other approaches, based on gradient descent, also try to converge to Nash Equilibria in self play (Abdallah and Lesser, 2008; Zhang and Lesser, 2010). Additionally, Kaisers and Tuyls linked evolutionary game theory to reinforcement learning to study the dynamic of Q-learning in multiagent problems. Their proposed algorithm, Frequency Adjusted Q-learning (FAQ), resembles regular Q-learning except that it changes the update method to compensate for the fact that actions are updated at different frequencies (Kaisers and Tuyls, 2010).

Cooperative learning presents a variety of special problems for independent learners as opposed to joint-action learners. Fulda and Ventura (2007) identified three problematic factors — "poor individual behavior", "action shadowing" (related to relative overgeneralization, as discussed later), and "equilibrium selection" (miscoordination) — and suggested that optimal performance might be achieved by solving them. However in their survey, Matignon et al. (2012) identified at least two more problems, and compared several independent learners in several games along with different exploration strategies to overcome these problems.

Some work has been done in cooperative games. Lauer and Riedmiller introduced Distributed Q-learning (2000), which addresses two major problems mentioned in (Fulda and Ventura, 2007). It tackles relative overgeneralization problem by only updating using the maximum Q-value — an approach we will develop further with LMRL2 — and it tackles the miscoordination problem by only changing the policy when the Q-value is updated. However this learner is vulnerable to games with stochastic rewards or transitions. Building on Distributed Q-learning, Hysteretic Q-learning (Matignon et al., 2007) uses two different learning rates to address stochasticity. One study (Bloembergen et al., 2010) adds our concept of leniency into FAQ, using a slightly different approach from what we do here. Leniency seems to help FAQ solve initial miscoordination difficulties in cooperative games.

Instead of changing the update procedure, researchers have also investigated the possibility of using different exploration strategies. Frequency Maximum Q-Value heuristic (FMQ), meant only for repeated games, used a form version of Boltzmann action selection where the Q-value was substituted by an expectation (Kapetanakis and Kudenko, 2002). FMQ was modified by Matignon et al. to handle stochasticity in repeated games (2008) and stochastic games (2009). The latter version of this algorithm was called *Swing between Optimistic or Neutral*, or SOoN.

Learners Based on the Nature of the Other Agents Most of the literature discussed so far focuses on learning in a certain kind of game. However, instead of making assumptions about the nature of a game, some methods are based on the nature of the other agents. In so-called Friend-or-Foe Q-learning (Littman, 2001a), the learner is essentially a mixture of two different Q-learners: one which updates the Q-table like a regular Q-learner if the other agent is thought to be a friend, and one which updates the Q-table using Minimax-Q if the agent is thought to be an opponent. Asymmetric Q-learning (Könönen, 2004), distinguishes between "leaders" and "followers" in the game, where a "leader" knows what action a "follower" will choose can guide the action selection of the "follower" (see also the leader-follower reward shaping in Babes et al. 2008). CMLeS provides different performance guarantees depending on different set of agents in the game (Chakraborty and Stone, 2013a,b). Specifically, if the agents are doing self-play, the algorithm will achieve a Nash Equilibrium joint policy; if the alternative agents are opponents with limited memory, the algorithm can guarantee a near-optimal response in polynomial time steps; and in other cases, the algorithm will play similarly to Minimax-Q.

Another area of research has studied adaptation to arbitrary agents of unknown type. Sullivan et al. (2006) studied FMQ and Lenient Learners in non-self play scenarios, and showed that while these learners could easily converge to an optimal Nash Equilbrium in self-play for various games, it was difficult for them to do so when paired with learners from entirely different algorithms. Stone et al. (2010) consider ad-hoc teams of multiple agents

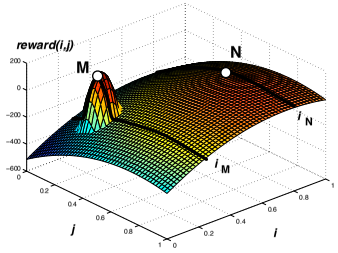


Figure 1: The relative overgeneralization pathology in multiagent learning. The axes i and j are the various actions that agents A_i and A_j may perform, and the axis $reward(i, j)$ is the joint reward received by the agents from a given joint action $\langle i, j \rangle$. Higher rewards are better. Joint action M has a higher reward than joint action N. However, the average (or sum) of all possible rewards for action i_M, of agent A_i is lower than the average of all possible rewards for action i_N. To illustrate this, bold lines show the set of outcomes from pairing i_M or i_N instead with other possible actions in j.

with different capabilities whose goals and utilities are aligned but which have had no prior coordination. They give a method to evaluate the performance of an agent in this scenario, and provide theoretical approaches to building a perfect ad hoc team. Additional follow-on methods have since been suggested (Stone and Kraus, 2010; Agmon et al., 2014; Genter et al., 2015; Barrett and Stone, 2015).

3. Lenient Learning

The notion of lenient learning, and the pathology it is meant to tackle, did not start in multiagent reinforcement learning research, but rather in an unexpected but surprisingly related topic: cooperative coevolutionary algorithms (CCEAs) (Potter and De Jong, 1994). CCEAs are multiagent stochastic optimization procedures and exhibit nearly identical pathologies to those found in repeated games.

Relative Overgeneralization Lenient learning was originally designed to combat a particular pathology in repeated games and CCEAs called *relative overgeneralization* (Wiegand, 2004). Relative overgeneralization occurs when a suboptimal Nash Equilibrium in the joint space of actions is preferred over an optimal Nash Equilibrium because each agents' action in the suboptimal equilibrium is a better choice when matched with *arbitrary* actions from the collaborating agents. Consider Figure 1, showing the reward over the joint action space for a two-agent game for agents A_i and A_j. Here, the joint candidate solution labeled M should clearly be preferred over another one, N. However if one assessed the quality of individual actions based on the sum or average reward received when paired with all possible other actions from the collaborating agent, then agent A_i's portion of the joint action N

(called i_N) would be preferred over its portion of the joint action M (called i_M). That is, quality$(i_M) = \sum_j \text{reward}(i_M, j) <$ quality$(i_N) = \sum_j \text{reward}(i_N, j)$. We will call such an approach an *average-based* algorithm.

This situation can easily arise in cooperative, repeated games for independent learners. Here is an example of relative overgeneralization in a repeated game:

		Agent 2		
		a	b	c
Agent 1	a	10	0	0
	b	0	5	6
	c	0	6	7

In this game, though $\langle a, a\rangle$ is clearly the best joint move for the two agents, if the agents sampled their actions randomly and based decisions on the average reward, action a would have the worst score, action c would have the best score, and b would be in the middle. Thus the agents would tend to converge to $\langle c, c\rangle$.

The obvious alternative this is to base the quality of a solution not on its *average* reward, but rather on its *best* reward when paired with various other actions. That is, quality$(i_M) = \max_j \text{reward}(i_M, j)$. In this case, a would be the best action, c the next highest, and b the worst. Thus the agents would tend to converge to the proper equilibrium, $\langle a, a\rangle$. We call this a *maximum-based* approach.

Relative overgeneralization is a specific and problematic subcase of the slightly more general notion of *action shadowing*, occasionally used by the MARL community, but has only been relatively recently studied (Panait et al., 2006b; Panait, 2006; Fulda and Ventura, 2007; Panait et al., 2008). One possible reason for this might be that common MARL test problems are relatively low in number of actions, and relative overgeneralization can only arise if each agent has at least three actions available, ideally many more. In contrast, in CCEAs the number of "actions", so to speak, is very high and often infinite, and so the pathology is a common occurrence there.

Stochastic Rewards and Lenient Learners Some reinforcement learning methods, so-called *optimistic methods* (Matignon et al., 2012), are generally maximum-based learners, or at least prefer superior results, for example, updating superior results with a higher learning rate. In a repeated game with deterministic rewards, these techniques are likely to perform very well. However, games with stochastic rewards will mislead such learners, since the highest rewards they see will often be due to noise. This problem isn't likely to occur with average-based learners.

Our approach instead begins with a maximum-reward learner and gradually shifts to an average-reward learner. The idea here is that early on none of the agents have a good understanding of their best joint actions, and so each agent must initially be *lenient* to the foolish and arbitrary actions being made by its collaborators. Later on, each agent focuses more on average reward, which helps escape the trap laid by stochastic rewards. We call this approach a *lenient multiagent reinforcement learning* algorithm.

Miscoordination Another common pathology which appears in repeated games is *miscoordination*, where two or more equivalent equilibria are offset in such a way that agents, each selecting what appears to be an optimal action, wind up with poor joint actions. For

example, in the game below, both actions a and b are reasonable for each agent. But if one agent chooses a while the other agent chooses b, they receive a low reward.

		Agent 2	
		a	b
Agent 1	a	10	0
	b	0	10

Stochastic Games and Deception Stochastic games add some new wrinkles to the above pathologies, because, at least for temporal difference learners, Q-values come partly from accumulated reward and partly from the backed-up rewards from follow-on states. This means that both miscoordination and relative overgeneralization, among other challenges, can arrive via backed-up rewards as well as immediate rewards.

The probabilistic transitions common in stochastic games present an additional hurdle for lenient learners, since such learners may consider only the highest backed-up rewards, even if they are very unlikely to occur. As was the case for probabilistic rewards, probabilistic transitions are largely overcome by the learner's gradual shift from being maximum-based to being average-based.

This hurdle leads to *deception*, another pathology which can arise in stochastic games. A deceptive game is one in which certain states have a high local reward but lead ultimately to states with poor future rewards: this is also known as the *delayed reward* problem. This creates a garden-path scenario where greedy agents are deceptively led away from the truly high-performing states.

4. The LMRL2 Algorithm

In (Panait et al., 2006b, 2013) we demonstrated a lenient learning algorithm, LMRL, for repeated games with independent learners, comparing it favorably to the FMQ algorithm (Kapetanakis and Kudenko, 2002) for variations of the Climb and Penalty games (Claus and Boutilier, 1998). We begin here with a slight modification of the algorithm, LMRL2, which is the degenerate case (for repeated games) of the full LMRL2 algorithm for stochastic games. We will then extend it to the stochastic game scenario.

LRML2 is a modified version of Q-learning which maintains per-action *temperatures* which are slowly decreased throughout the learning process. An action's temperature affects two things. First, it affects the degree of randomness of action selection: with high temperatures, action selection is largely random, and with low temperatures, action selection is greedily based on the actions with the highest Q-values. To do this, LRML2 applies a temperature-based Boltzmann Selection common in other algorithms in the literature. However, when the average temperature drops below a certain *minimum temperature*, and LRML2's action selection will suddenly become purely greedy. This minimum temperature was originally added to avoid floating point overflows common in Boltzmann Selection; but we have found that it also is beneficial for nearly all games.

Second, temperature affects the lenience of the algorithm: high temperatures cause LRML2 to be lenient, and so only mix rewards into Q-values if they are better than or equal to the current Q-value. With a low temperature LRML2 mixes all rewards into the

Q-values no matter what. Unlike action selection, lenience is not affected by the minimum temperature bound.

Repeated Games LRML2 for repeated games relies on the following parameters. Except for θ and ω, all of them will be fixed to the defaults shown, and will not be modified:

$\alpha \leftarrow 0.1$	learning rate
$\gamma \leftarrow 0.9$	discount for infinite horizon
$\delta \leftarrow 0.995$	temperature decay coefficient
$\mathit{MaxTemp} \leftarrow 50.0$	maximum temperature
$\mathit{MinTemp} \leftarrow 2.0$	minimum temperature
$\omega > 0$	action selection moderation factor (by default 1.0)
$\theta > 0$	lenience moderation factor (by default 1.0)

The α and γ parameters are standard parameters found in Q-learning, and their values here are typical settings from the literature. The parameters *MaxTemp,* δ, and *MinTemp* are also generally constants. The parameter ω determines the degree to which temperature affects the randomness of the Boltzmann action selection. In all cases but two (discussed later) we set this to 1.0. Finally and crucially, the θ parameter determines the degree to which temperature affects the dropoff in lenience. This parameter is the primary, and usually only, parameter which must be tuned, as different problems require different amounts of lenience.

LMRL2 maintains two tables: Q, a table of Q-values per action a, and T, a table of temperatures per-action. Initially $\forall a$:

$$Q(a) \leftarrow \text{initialize}(a) \qquad \triangleleft \textit{ See discussion below}$$
$$T(a) \leftarrow \mathit{MaxTemp}$$

In lenient learning, the choice of initial Q values is important to the operation of the algorithm. Consider if $\forall a : Q(a) = 0$ initially. What if a game consisted only of negative rewards? The lenient learner, at least during its high-temperature period, would refuse to merge *any* of them into the Q-values because they are too small. We have two strategies for initializing Q:

- **Initialize to Infinity** $\forall a : Q(a) = \infty$. This signals to LMRL2 to later reinitialize each Q value to the first reward received. Furthermore, as long as one or more action has an infinite Q-value, LRML2 will only select among such actions. This forces the algorithm to try every action at least once initially in order to initialize them. The disadvantage of this initialization approach is that if the game has stochastic rewards, and first reward received is high, LMRL2 will be locked to this high reward early on.
- **Initialize to Minimum** $\forall a : Q(a) =$ the minimum possible reward over any action the game. The disadvantage of this approach is that it requires that LMRL2 know the minimum reward beforehand.

We will by default initialize to the minimum possible reward in the game.

LMRL2 then iterates for some n times through the following four steps. First, it computes a mean temperature $\overline{T}$. Second, using this mean temperature it selects an action to perform. Third, it performs the action and receives a reward resulting from the joint actions of all agents (all agents perform this step simultaneously and synchronously). Fourth, it updates the Q and T tables. This iteration is:

1. Compute the mean temperature as: $\overline{T} \leftarrow \text{mean}_a\, T(a)$
2. Select a as follows. If $\overline{T} < \mathit{MinTemp}$, or if $\max_a Q(a) = \infty$, select $\text{argmax}_a\, Q(a)$, breaking ties randomly. Otherwise use Boltzmann Selection:
   (a) Compute the action selection weights as: $\forall a : W_a \leftarrow e^{\frac{Q(a)}{\omega \overline{T}}}$
   (b) Normalize to the action selection probabilities as: $\forall a : P_a \leftarrow \frac{W_a}{\sum_i W_i}$
   (c) Use the probability distribution P to select action a.
3. The agent does action a and receives reward r.
4. Update $Q(a)$ and $T(a)$ only for the performed action a as:

$$\mathit{Rand} \leftarrow \text{random real value between 0 and 1}$$
$$Q(a) \leftarrow \begin{cases} r & \text{if } Q(a) = \infty \qquad \text{(initialization was to infinity)} \\ (1-\alpha)Q(a) + \alpha r & \text{else if } Q(a) \leq r \text{ or } (\mathit{Rand} < 1 - e^{\frac{-1}{\theta T(a)}}) \\ Q(a) & \text{else} \end{cases}$$
$$T(a) \leftarrow \delta T(a)$$

5. Go to 1.

Note that each action has its own separate temperature which is decreased only when that action is selected. This allows LMRL2 to keep temperatures high for actions which have not been visited much and still require lenience and exploration, while permitting other actions to cool down.

Stochastic Games In stochastic games, the agents are at any particular time in some state s; and after performing their joint action, receive a reward r and transition to some new state s'. Accordingly, the extension of LMRL2 to stochastic games is largely the same as the repeated game version, except that the $Q(a)$ and $T(a)$ tables are modified to include the current state s: that is, they are now defined as $Q(s,a)$ and $T(s,a)$ respectively.

The iteration is largely the same, except in how $Q(s,a)$ and $T(s,a)$ are updated. First, $Q(s,a)$ is updated in standard Q-learning style to incorporate both the reward and expectation of future utility, as $r + \gamma \max_{a'} Q(s',a')$. However, if not all actions have been explored in s', then $\max_{a'} Q(s',a')$ will still be infinite, in which case it is ignored and $Q(s,a)$ just incorporates r.

Second and more interestingly, not only do we decrease $T(s,a)$, but we also fold into it some portion τ of the mean temperatures found for s'. τ is a new constant, fixed like α to 0.1. The idea is as follows: in many games (particularly episodic ones), early states are often explored much more than later states, and thus cool down faster. We want to keep

these early states hot long enough that propagation of future rewards from the later states will inform the Q-values of the early states before they are set in stone as the temperature falls. To do this, we take some of the temperature s' of the later state and back it up into s.

Because stochastic games can terminate, we wrap the entire process in an outer loop to repeatedly play the game:

Parameters:

$\alpha \leftarrow 0.1$	learning rate
$\gamma \leftarrow 0.9$	discount for infinite horizon
$\tau \leftarrow 0.1$	temperature diffusion coefficient
$\delta \leftarrow 0.995$	temperature decay coefficient
$MaxTemp \leftarrow 50.0$	maximum temperature
$MinTemp \leftarrow 2.0$	minimum temperature
$\omega > 0$	action selection moderation factor (by default 1.0)
$\theta > 0$	lenience moderation factor (by default 1.0)

Initially $\forall s, a$:

$Q(s,a) \leftarrow \text{initialize}(s,a)$ $\quad\quad$ $\triangleleft$ *See Previous Discussion*
$T(s,a) \leftarrow MaxTemp$

Repeat:

1. Current state $s \leftarrow$ initial state.
2. Repeat until the current state s is the end state (if any):
   (a) Compute the mean temperature for current state s as: $\overline{T}(s) \leftarrow \text{mean}_a\, T(s,a)$
   (b) Select a as follows. If $\overline{T}(s) < MinTemp$, or if $\max_a Q(s,a) = \infty$, select $\text{argmax}_a\, Q(s,a)$, breaking ties randomly. Otherwise use Boltzmann Selection:
      i. Compute the action selection weights in current state s as: $\forall a : W_a \leftarrow e^{\frac{Q(s,a)}{\omega T(s)}}$
      ii. Normalize to the action selection probabilities in current state s as: $\forall a : P_a \leftarrow \frac{W_a}{\sum_i W_i}$
      iii. Use the probability distribution P to select action a.
   (c) The agent, in current state s, does action a, receives reward r, and transitions to new state s'.

   (d) Update $Q(s,a)$ and $T(s,a)$ only for the performed action a as:

$$Rand \leftarrow \text{random real value between 0 and 1}$$

$$R \leftarrow \begin{cases} r & \text{if } \max_{a'} Q(s',a') = \infty \\ r + \gamma \max_{a'} Q(s',a') & \text{else} \end{cases}$$

$$Q(s,a) \leftarrow \begin{cases} R & \text{if } Q(s,a) = \infty \text{ (initialization was to infinity)} \\ (1-\alpha)Q(s,a) + \alpha R & \text{else if } Q(s,a) \leq R \text{ or } (Rand < 1 - e^{\frac{-1}{\theta T(s,a)}}) \\ Q(s,a) & \text{else} \end{cases}$$

$$T(s,a) \leftarrow \delta \times \begin{cases} (1-\tau)T(s,a) + \tau\overline{T}(s') & \text{if } s' \text{ is not the end state (if any)} \\ T(s,a) & \text{else} \end{cases}$$

   (e) $s \leftarrow s'$

If we defined a repeated game as consisting of an initial state s which always transitions to the end state as s', this algorithm degenerates to the repeated version of LMRL2 discussed earlier.

5. Comparison with Other Methods

We begin with a comparison of LMRL2 against six other independent-learner algorithms in self-play in several cooperative test problems. The algorithms are standard (classic) Q-Learning, Distributed Q-Learning, Hysteretic Q-Learning, WoLF-PHC, SOoN (Swing between Optimistic or Neutral), and FMQ. The different techniques use a variety of parameters and have several action selection methods as options. We note that WoLF-PHC is a general-sum algorithm rather than strictly a cooperative one, and that FMQ can only be applied to repeated games. In a MARL context, standard Q-Learning is sometimes known as *Decentralized Q-Learning*. We briefly discuss all except for standard Q-Learning below.

Distributed Q-Learning We believe that Distributed Q-Learning (Lauer and Riedmiller, 2000) was the earliest independent-learner algorithm specifically designed for cooperative multiagent learning scenarios. In Distributed Q-Learning, each learner has a Q-table and a policy table. Unlike the regular Q-Learning, where a value in the Q-table is updated by combining it with some portion of the reward and follow-on utility, in Distributed Q-Learning the Q-value is completely replaced by the new reward and follow-on utility (that is, the learning rate is effectively 1.0), but only if doing so would increase it. This makes Distributed Q-Learning a maximum-based learner; and it is intended to address relative overgeneralization. Because it is highly optimistic, the algorithm has problems with stochasticity: indeed Lauer and Riedmiller (2000) acknowledged that this was still an open question.

To deal with miscoordination, Distributed Q-Learning has another trick up its sleeve: its policy is only updated when the Q-value of the policy's chosen action is no longer the highest such value. At this point a new action for the policy is chosen at random among those with the highest Q-values. The idea here is to cause agents to lock onto the earliest-discovered good action (and Nash Equilibrium) even when other equivalent Equilibria exist. Through this implicit agreement among agents, miscoordination can be avoided.

Hysteretic Learning Matignon et al. (2007) proposed Hysteretic Learning to address Distributed Q-Learning's vulnerability to stochasticity. Hysteretic Learning is not a fully maximum-based learner: rather, if updating the Q-value would reduce it, it is reduced with a smaller learning rate than when it would be if it were increased. Hysteretic Learning does not attempt to solve miscoordination explicitly, but experiments suggest that the algorithm is very robust to miscoordination issues, in part due to the randomization in its exploration strategy (Matignon et al., 2012).

FMQ Rather than change the update strategy, as is done in the previous two methods, the Frequency Maximum Q-Value (FMQ) heuristic instead tries changing the action-selection (exploration) strategy (Kapetanakis and Kudenko, 2002). FMQ is only meant for repeated games. FMQ uses a modified version of Boltzmann exploration, similar to the one used in LMRL2 as discussed later: the value produced via Boltzmann is sometimes called the *expected reward* or *estimated value*. However, this alone is not sufficiently informative to deal with relative overgeneralization. Thus, FMQ adds an additional term to its Q-value when using it during action selection. This additional term is the product of the *highest reward* seen so far when selecting that action, the *frequency* of getting that reward, and a *weighting factor* c which controls how much this term affects action selection: that is, how much action selection is based on the "average" reward versus typical "high" rewards.

FMQ are designed to deal with stochasticity in relative overgeneralization problems, but its learned policy can still be wrong if the variance of the reward function is high.

SOoN SOoN may be thought as a heavily modified FMQ. First, the frequency term is broken into two terms. The *myopic frequency* is the the same frequency term in original FMQ algorithm, and the *farsighted frequency* is used to deal with deception in games. Both frequency values are updated using temporal-difference style methods rather than a simple average. Additionally, the estimated value is computed as a linear interpolation between an Optimistic Q-value (the Q-value in Distributed Q-Learning) and an Average Q-value (the classical Q-Learning value). The algorithm introduces two new parameters, α_f and α_g, to replace the weighting factor c in FMQ to control the impact of the frequency terms.

WoLF-PHC Given its notoriety, we chose WoLF-PHC as our exemplar general-sum algorithm to compare against. WoLF-PHC is a policy hill-climbing algorithm and so is somewhat different from the various Q-Learning methods (Bowling and Veloso, 2001b). WoLF-PHC compares the expected value of current policy with the expected value of the average policy. If the former is lower, then the agent is "losing", else it is "winning". WoLF-PHC then uses one of two different learning rates depending on whether it is winning or losing (the "winning" learning rate is higher). As it is meant for general-sum games, WoLF-PHC can be straightforwardly applied to cooperative games: but obviously since it is more general-purpose, a comparison against it is somewhat unfair.

5.1 Test Problems

We tested against twelve games, either from the literature or of our own devising. This collection was meant to test a diverse array of situations, including: stochastic and repeated games, recurrent and episodic games, deterministic and stochastic rewards, deterministic and stochastic state transition functions, deceptive problems, miscoordination, and relative

overgeneralization. The games are defined in Appendix **??**, but their various features are summarized here.

We tested with four repeated-game test problems from the literature. The widely used *Climb* and *Penalty* games (Claus and Boutilier, 1998) are designed to test some degree of relative overgeneralization and miscoordination. We also included versions of the Climb game with partially stochastic and fully stochastic rewards, here designated *Climb-PS* and *Climb-FS* respectively (Kapetanakis and Kudenko, 2002). In these versions, the reward for a joint action was potentially one of two different values with certain probabilities; though the expected reward was always the same as in the original Climb game. Stochasticity greatly complicates the problem: we noticed that no existing algorithm can completely solve Climb-FS.

We also tested against several stochastic games. The *Boutilier* game (Boutilier, 1999) was a repeated game with deterministic transitions which distributed a miscoordination situation among several stages. The *Common Interest* game (Vrancx et al., 2008) is also recurrent, but with stochastic transitions and some miscoordination. This game is notable in that its rewards are unusually close to zero compared to other games; this is the reason that LMRL2 required a modification of its ω parameter to perform well in this game.

The remaining games are of our own design. The *Gradient 1* game is a deterministic episodic game designed to be highly deceptive and to cause miscoordination. *Gradient 2* is similar, except that it incorporates fully stochastic rewards. The *Heaven and Hell* game also causes miscoordination and deception, but is recurrent. This game has a high-reward state ("heaven"), a low-reward state ("hell"), and two medium ("purgatory") states of different levels of reward. Choice of a high-reward action will deceptively transition to a lower-reward future state, and the converse is also true. The *Relative Overgeneralization 1* game (or RO 1) causes miscoordination and relative overgeneralization not in local rewards but in propagated utilities from later states. *Relative Overgeneralization 2* (or RO 2) causes relative overgeneralization in both local rewards *and* in propagated utilities. Finally *Relative Overgeneralization 3* (or RO 3) causes miscoordination and relative overgeneralization from propagated utilities, but does so entirely through a stochastic transition function. We summarize the properties of each game in Table 1.

5.2 Parameters

All the techniques had a variety of tunable parameters, and many of them could apply different action selection methods. We considered two such methods, *epsilon-greedy selection* and *Boltzmann selection*. Epsilon-greedy selection is characterized by an initial random action selection probability ϵ which is optionally reduced each timestep by multiplying it by a cut-down factor ν. The version of Boltzmann selection was the one we employed in the LMRL2 algorithm, and so was characterized by the *MaxTemp, MinTemp,* δ, and ω parameters. Boltzmann selection could be feasibly used by LMRL2, Q-Learning, and Hysteretic Q-Learning, though usually only LMRL2 benefitted from it (other learners generally worked better with Epsilon-greedy). FMQ used its own algorithmic-specific version of Boltzmann selection. Table 2 shows the default (baseline) parameter settings we set for each of the techniques.

Test Problem	Repeated Game	Episodic Game	Deception	Stochastic Transitions	Stochastic Rewards	Relative Overgeneralization	Miscoordination
Boutilier			×				×
Common Interest			×	×			×
Gradient 1		×	×				×
Gradient 2		×	×		×		×
Heaven and Hell			×				×
RO1		×	×			×	
RO2		×	×			×	
RO3		×	×	×		×	×
Climb	×	×				×	
Climb-PS	×	×			×	×	
Climb-FS	×	×			×	×	
Penalty	×	×					×

Table 1: Properties of each test problem

Test Problem	Default Parameters
LMRL2	$\alpha : 0.1,\ \gamma : 0.9,\ \tau : 0.1,\ \delta : 0.995,$ MaxTemp : 50, MinTemp : 2, $\omega : 1,\ \theta : 1,$ Boltzmann
Q	$\alpha : 0.1,\ \gamma : 0.9,\ \epsilon : 0.1,\ \nu : 1.0,$ Epsilon-greedy
Distributed Q	$\gamma : 0.9,\ \epsilon : 0.1,\ \nu : 1.0,$ Epsilon-greedy
Hysteretic Q	$\alpha : 0.1,\ \gamma : 0.9,\ \epsilon : 0.1,\ \nu : 1.0,\ \beta : 0.01,$ Epsilon-greedy
WoLF-PHC	$\alpha : 0.1,\ \gamma : 0.9,\ \epsilon : 0.1,\ \nu : 1.0,\ \delta_w : 0.03,\ \delta_l : 0.06,$ Epsilon-greedy
SOoN	$\alpha : 0.1,\ \gamma : 0.9,\ \epsilon : 0.1,\ \alpha_f : 0.05,\ \alpha_g : 0.3,$ Epsilon-greedy
FMQ	$\alpha : 0.1,\ \gamma : 0.9,\ c : 10,$ MaxTemp : 500, MaxMove : 2000, FMQ-specific Boltzmann

Table 2: Default Parameter Settings for each technique.

To make the comparison as fair as possible, we then optimized the above parameter settings and action selection method choices on a per-problem, per-technique basis, through a combination of manual tuning and automated hill-climbing. Table 3 shows the resulting optimized parameter changes.

5.3 Comparison

We compared all the aforementioned techniques using all appropriate test problems (FMQ was included only for repeated games). We considered two possible measures of successful convergence on a problem. First, a technique may converge to the *correct policy* for a given game, meaning that it is optimal if it follows this policy. Second, a technique may converge to the more general *complete policy*, meaning that it determines the correct joint action for *every state*, even ones which would never be visited if it followed a correct policy. For some games (all repeated games, and games with fully stochastic transitions) correct and complete policies are the same.

One might imagine that the superior technique would be the one which converged to the complete policy the most often. However what if in doing so it also converged to a great many entirely wrong policies? Consider the following exaggerated scenario. Technique A converged to 2 complete policies, 98 correct policies, and no incorrect policies; while technique B converged to 3 complete policies but 97 incorrect policies. Would we then

Test Problem	LMRL2	Q-Learning	Distributed Q	Hysteretic Q	WoLF-PHC	SOoN	FMQ
Boutilier	—	α:0.05 γ:0.8	—	—	ν:0.997	—	
Common Interest	ω:0.1	α:0.03	ϵ:0.2 ν:0.9	β:0.02	—	α:0.05 α_f:0.3 ϵ:0.05	
Gradient 1	θ:10^7	α:0.15 γ:1 ϵ:0.35	ϵ:0.4	ϵ:0.4	α:0.15 γ:1 δ_l:0.05 ϵ:0.5	α_g:0.05 α_f:0.1 ϵ:0.3	
Gradient 2	—	α:0.95 ϵ:0.3	ϵ:0.01	β:0.15 MinTemp:0.015 MaxTemp:500 δ:0.999 Boltzmann	α:0.15 γ:1 ϵ:0.4	α_g:0.1 α_f:0.3 ϵ:0.05	
Heaven and Hell	θ:10^7	α:0.15 γ:0.7	—	α:0.15 β:0.03 γ:0.7	α:0.2 γ:0.8 ϵ:0.2	α_g:0.05 α_f:0.03	
RO 1	θ:10^3	α:0.15 γ:1 ϵ:0.13	ϵ:0.15	β:0.0001 ϵ:0.2	δ_w:0.06 δ_l:0.12 ϵ:0.01	α_g:0.02	
RO 2	θ:10^3	ϵ:0.25	ϵ:0.4	α:0.3 β:0.0001 γ:1 ϵ:0.5	δ_w:0.06 δ_l:0.12 ϵ:0.01	α_g:0.01	
RO 3	ω:0.3	α:0.13 ϵ:0.13	ϵ:0	β:0.0001 ϵ:0.05	δ_w:0.06 δ_l:0.12	α:0.05 α_g:0.03 α_f:0.03	
Climb	θ:10^7	ϵ:0.0	ϵ:0.1	β:0.0001	δ_l:0.21 ϵ:0.01	—	—
Climb-PS	θ:10^3	ϵ:0.01	ϵ:0	β:0.01 MinTemp:2 MaxTemp:40 δ:0.99 Boltzmann	δ_l:0.21 ϵ:0.01	α_g:0.2 α_f:0.05 ϵ:0.05	—
Climb-FS	θ:10	α:0.05 γ:0.8	ϵ:0.01	β:0.001 ν:0.99	δ_l:0.18 ϵ:0.01	α_g:0.2 α_f:0.3 α:0.03	c:200
Penalty	—	α:0.05 ϵ:0.35	ϵ:0.1	β:0.01 ϵ:0.12	ϵ:0.28	—	—

Table 3: Tuned Parameter Settings. Shown are deviations from the default settings (in Table 2) for each method when tuned to perform best on a given problem.

Test Problem	Iterations	Complete	Correct
Boutilier	30000	$\overline{\text{L Q D}}$ $\overline{\text{H W}}$ S	$\overline{\text{L Q D H W S}}$
Common Interest	40000	$\overline{\text{H L W}}$ $\overline{\text{Q S}}$ D	
Gradient 1	40000	D H S $\overline{\text{Q L}}$ W	$\overline{\text{L D S}}$ H Q W
Gradient 2	40000	W S Q L H D	L S $\overline{\text{Q W H}}$ D
Heaven and Hell	40000	$\overline{\text{D L}}$ W H S Q	$\overline{\text{L D H W S Q}}$
RO 1	30000	$\overline{\text{D H}}$ S L Q W	$\overline{\text{L D H S}}$ Q W
RO 2	30000	L $\overline{\text{S H}}$ D $\overline{\text{Q W}}$	$\overline{\text{L H S}}$ D $\overline{\text{Q W}}$
RO 3	30000	L $\overline{\text{Q S W H}}$ D	L $\overline{\text{Q S W H}}$ D
Climb	15000	$\overline{\text{D H S L}}$ F Q W	
Climb-PS	15000	S L F H D Q W	
Climb-FS	15000	L S $\overline{\text{F D}}$ H Q W	
Penalty	15000	$\overline{\text{D H F S L Q}}$ W	

L=LMRL2 Q=Q-Learning D=Distributed Q H=Hysteretic Q W=WoLF-PHC S=SOoN F=FMQ

Table 4: Summary of Statistically Significant differences among methods for complete and correct solution counts on different test problems. Techniques are ordered left to right in decreasing performance. $\overline{\text{Overbars}}$ group together settings with no statistically significant differences among them. In the Common Interest, Penalty, and various Climb games, correct solutions are by definition complete. Also shown is the number of iterations run on each test problem.

really consider Technique B to be superior? For this reason, we must consider *both* of these approaches as comparison measures.

Our interest is not in convergence time so much as quality of the converged result. As such we ran each technique for until both the value function stabilized and the policy remained unchanged. We used the largest iteration number required among all learners for our formal experiments. This often benefitted LMRL2: being based on a temperature schedule, LMRL2 converges more slowly than other (greedier) techniques. However, it also benefitted other methods which occasionally needed large numbers of iterations and small settings of ϵ to produce good results. Not surprisingly, the number of iterations needed was closely correlated to the maximum path length of the game.

Because the results are from Bernoulli trials, we needed a large number of runs to do proper statistical comparison. We chose 10,000 iterations up front, and compared differences in complete or correct solution counts. Statistical significance was verified using the Marasquilo procedure for χ^2. We used $p = 0.05$ for the entire experiment, Bonferroni-corrected to $p = 0.0026315789$ per problem.

5.4 Results

Table 4 summarizes the rank order among the methods and statistically significant differences. Table 5 shows the actual results.

- LMRL2 fell in the top statistical significance tier for "complete" eight times, two times more than the next-best method (Distributed Q-Learning).

Test Problem	LMRL2	Q-Learning	Distributed Q	Hysteretic Q	WoLF-PHC	SOoN FMQ
Boutilier	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**	**9998**/ **9998**	**9996**/ **9996**
Common Interest	**9971**	9942	2080	**9990**	**9968**	9896
Gradient 1	*8407/**10000**	8609/ 8695	**9999**/ **9999**	9925/ 9926	2329/ 2335	8966/ **9986**
Gradient 2	548/ **9997**	2430/ 5995	0/ 1266	157/ 5609	**5147**/ 5897	3329/ 6770
Heaven and Hell	**9991**/**10000**	8289/ 9942	**10000**/**10000**	9848/**10000**	9954/**10000**	9530/ **9975**
RO 1	†9368/**10000**	3350/ 3350	**10000**/**10000**	**10000**/**10000**	1943/ 2280	9877/**10000**
RO 2	**9999**/**10000**	2/ 514	9258/ 9258	9897/**10000**	2/ 512	9924/**10000**
RO 3	**7332**/ **7332**	5337/ 5337	2272/ 2272	2737/ 2737	2769/ 2769	5171/ 5171
Climb	**9999**	1661	**10000**	**10000**	387	**10000** 9956
Climb-PS	9930	1820	2821	7454	391	**9995** 9857
Climb-FS	**9016**	1763	3874	2558	676	8723 3894
Penalty	**9999**	**9997**	**10000**	**10000**	9951	**10000** **10000**

* With $\alpha = 0.05$ this value rose to 9207. This changed the ranking of "complete" results such that LMRL2 moved from worse than Q-Learning to better than Q-Learning (both statistically significant differences).

† With $\alpha = 0.05$ this value rose to 9750, but did not change any ranking.

Table 5: Complete/Correct solution counts among methods on various test problems. Boldface values indicate methods which were not statistically significantly different with the highest-performing method for a given problem. Note that in the Common Interest, Penalty, and various Climb games, correct solutions are by definition complete.

- LMRL2 fell in the top statistical significance tier for "correct" eleven times, three times more than the next-best method (Swing between Optimistic or Neutral).
- LMRL2 was uniquely statistically significantly best in RO 1 and Climb-FS.
- SOoN was uniquely statistically significantly best in Climb-PS.
- The Gradient 2 game proved very challenging to all methods. Yet WoLF-PHC, which often failed in other games, was statistically significantly best at Gradient 2 in "complete". LMRL2 underperformed WoLF-PHC on Gradient 2 in "complete", but outperformed it (and all others) in "correct".
- Changing α improved LMRL2 twice, and once statistically significantly, but not by much.
- All methods easily solved the Boutilier game.

In short, LMRL2 performed very well. It often did not reach 100%, but it was usually easily close enough to fall in the top statistical significance tier for "complete", and almost always in the top tier for "correct". No other method was as consistent across games. Furthermore, LMRL2 did so with very few changes to its default parameters: in all cases at most one parameter needed to be tuned. We are particularly interested in the fact that while LMRL2 performed very well in RO 2 and RO 3 — games which exhibit the relative overgeneralization pathology for which LMRL2 was designed — it underperformed in RO 1!

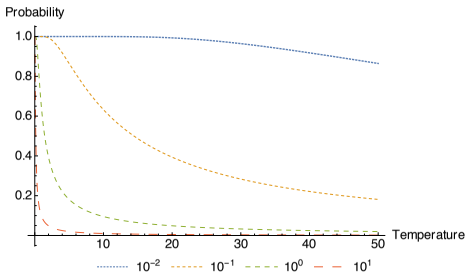


Figure 2: Probability that LMRL2 will accept poor results, by current temperature, for various θ values.

6. Analysis and Discussion

We have experimented with a number of variations of LMRL2 and have examined different issues which arise when using it. In the following section, we discuss five of these issues. First, we discuss the critical issue of initialization of Q-values and its impact on LMRL2's success. Second, we discuss how LMRL2 deals with issues in miscoordination. Third, we examine the disadvantages of Boltzmann selection and compare alternatives. Fourth, we consider the impact of *latching*, whereby temperature is propagated to earlier states only when it is hotter than those states. Finally, we examine what happens when LMRL2 is paired with other techniques in non-self-play, and so consider if, so to speak, two heads are better than one.

6.1 Lenience and Q-value Initialization

We first consider how θ affects lenience. LMRL2 (initially), Hysteretic Q-learning, and Distributed Q-learning are all *optimistic learners*, meaning that they unilaterally update the Q-values to more closely reflect new results if the new results are superior: but they are more conservative when the new results are inferior. That is, they incorporate some portion of maximum-based learning. More specifically: LMRL2 updates to inferior results only with a certain probability (which increases over time). Hysteretic Q-learning updates to inferior results at a fixed, slower rate. And Distributed Q-learning never updates to inferior results.

In LMRL2, optimism is controlled by θ. Figure 2 shows the probability curves of doing a Q-value update corresponding to different θ values and different temperatures. We can see that, with some given initial temperature (perhaps 50), a higher θ value has *two* effects: a higher initial probability of doing Q-value updates and a different probability curve shape. With very high θ, the curve approaches the maximum-learner rule employed by Distributed Q-learning. On the other hand, with a very low θ value, the curve approaches the averaging-learner rule employed by standard Q-learning. We note that a similar relationship can also be made with Hysteretic Q-learning. In Hysteretic Q-learning, improved results are updated

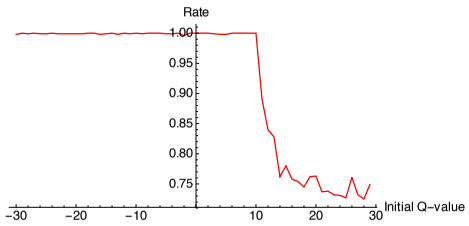


Figure 3: Rate of complete solutions for the LMRL2 in the Climb game, by various initial Q-values. Each point reflects the average of 1000 independent runs.

using α as usual, but worse results are updated using a smaller learning rate $\beta \leq \alpha$. If $\beta = 0$, Hysteretic Q-learning more or less degenerates to regular Q-learning.

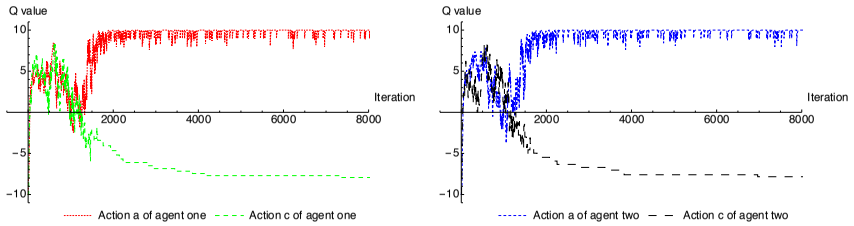
As a lenient learner, LMRL2's optimism is also tempered by the current *temperature*. As can be seen in Figure 2, the initial temperature has a significant effect on the progression of lenience in the system: and indeed if the temperature is infinity, LMRL2 becomes fairly similar to Distributed Q-learning.

In Section 4 we spent significant time discussing Q-value initialization options: this is because optimistic learners, and the like, are very sensitive to initialization. If the Q-values are initialized too high, then these learners have no easy way to adjust them to correct settings: indeed Distributed Q-learning may be completely unable to learn at all. Because in traditional average-based Q-learning the initial estimate of Q-value is less of a concern for proper convergence (Jaakkola et al., 1994), many algorithms simply ignore the discussion of initialization, setting the initial value to 0.

LMRL2 is initially highly optimistic, and so poor initialization can severely hinder it. If the Q-value is greater than the actual feedback, initially the lenient learner will largely refuse to update the Q-value, and so the Q-value won't change, and LMRL2 will thus just pick randomly from among actions during action selection. The temperature of all of the actions then will drop roughly the same rate. Only when the temperature drops enough that LMRL2 shifts to an average-based mode will it start to properly update the Q-values at all. Figure 3 shows how strongly the choice of initial Q values can effect performance: here LMRL2 quickly drops off in performance when the initial value rises to above 10.

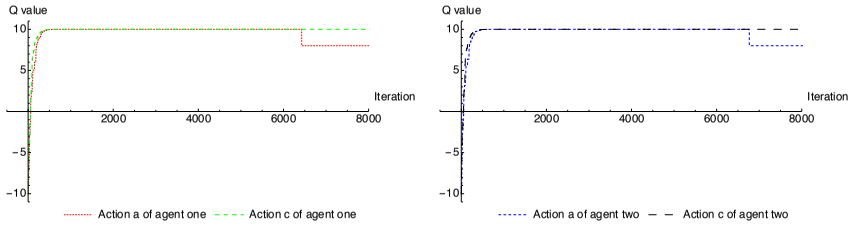
6.2 Miscoordination

To solve the miscoordination problem, an algorithm must break the symmetry among the various optimal equilibria. One way to do this is to always ϵ-greedily select the action with best Q-value, (crucially) breaking ties by selecting the action used last time. This is the approach used in Distributed Q-learning: but this method will only work in self-play. Another way to break symmetry is via different convergence times, which we will talk about in Section 6.5. LMRL2 at present simply relies on the fact that miscoordination is unstable, and that with enough random action, it will likely eventually work its way out.



(a) Agent One (b) Agent Two

Figure 4: Q-values of two agents in the Penalty game with $\theta = 0.1$



(a) Agent One (b) Agent Two

Figure 5: Q-values of two agents in the Penalty game with $\theta = 10^7$

Miscoordination is primarily a problem for a lenient learner early on, when miscoordinated actions both appear to have the same Q-values due to the use of maximum-based learning. However at some point one action will randomly have a slightly higher Q-value than the other, and as the temperature drops, action selection will increasingly choose that action. We have found that this tends to create a sudden collapse to one equilibrium or another. What often happens is that one agent starts using a given action more often, which quickly makes the miscoordinated action a bad deal for the other agent.

Figure 4 shows the Q-value of two LMRL2 agents playing the Penalty game, which has two miscoordinated equillibria, $\langle a, a\rangle$ and $\langle c, c\rangle$. During the first several hundred iterations, the Q-value of both actions increases due to lenience, though neither reaches the actual value of 10, due to a low lenience factor ($\theta = 0.1$). Thereafter, the Q-value of both actions for both agents start to drop, as lenience starts to wear off. At some point, action a is slightly preferred by at least one agent, which at this point is sufficient to quickly lock the agents into the equilibrium $\langle a, a\rangle$.

If learners are being lenient for a very long time, this lock-in could also take a long time: but it will eventually happen. In our experiments, we have found that low lenience is not necessary: even with a high value of θ, lock-in to a given equilibrium will occur at some point. We experimented with θ values of 1, 10, 100, 1000, 10,000, and 100,000 on the Penalty game: in every case, LMRL2 worked around miscoordination and found the correct policy in 1000 out of 1000 runs.

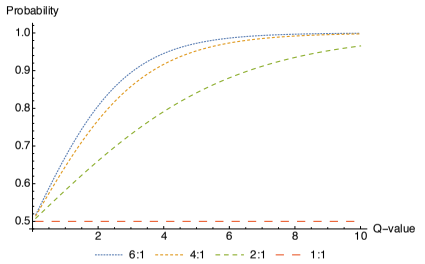


Figure 6: Given two actions A and B, with Q-values $Q_A \geq Q_B$, the probability of selecting action A over B using Boltzmann Selection, for various settings of Q_A. The different lines reflect the ratio of $Q_A : Q_B$, that is, how much larger Q_A is compared to Q_B.

Here is a typical example. Figure 5 shows the Q-values of the agent in the same game but with *extreme* lenience level $\theta = 10^7$. In this run, both miscoordinated equilibria have converged to near 10, and stay like this for a long time, but at about 6500 iterations, Agent 1 suddenly collapses to action a. This then forces Agent 2 to collapse to action a about 250 iterations later.

6.3 Exploration Strategy

A reinforcement learner may be roughly divided into two parts, the *learning policy* (or *action selection*) and the *update rule* (Singh et al., 2000). When applied to cooperative MARL, significant effort has been expended in the literature on the update rule, but the learning policy has received less attention; but coordination among learners involves both parts. Here we examine LMRL2's choices of learning policy as part of its exploration strategy.

LMRL2 begins using a modified version of Boltzmann Selection (Kaelbling et al., 1996), but when the temperature drops below some minimum value, it suddenly jumps to a fully greedy strategy (that is, $\epsilon = 0$). Our original reason to have LMRL2 do this was that with low temperatures, Boltzmann Selection is subject to floating-point overflow problems. However we have found that this scheme also happens to perform quite well compared to a traditional Boltzmann Selection algorithm. We begin here with an analysis of Boltzmann Selection as used in LMRL2, and particularly with the use of the ω action-selection tuning parameter. We then go on to some analysis of why jumping to a fully greedy strategy later on seems to be an effective mechanism.

Boltzmann Selection Strategy In two of the games, LMRL2 must adjust the parameter ω to achieve good performance. One of the difficulties with Boltzmann Selection is that when Q-values are close (and importantly, when they are all small), Boltzmann Selection does not distinguish among them well (Kaelbling et al., 1996). For example, Figure 6 shows a simple repeated game with two actions A and B, where the Q value of action A

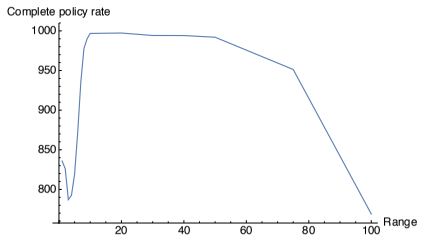


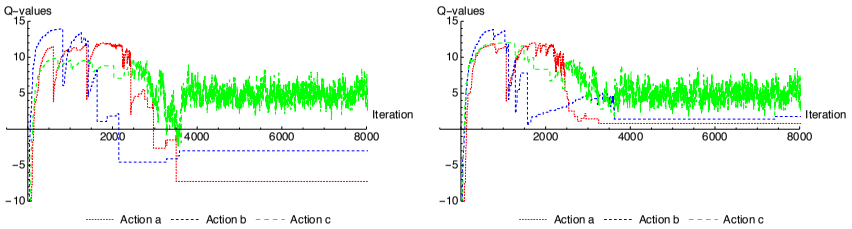
Figure 7: Rate of complete solutions for the LMRL2 in the Common Interest game, using Ranked Boltzmann Selection with values of n ranging from 1 to 100. Each point reflects the average of 1000 independent runs.

Test Problem	Original LMRL2 Results	LRML2 with Ranked Boltzmann Selection
Boutilier	**10000**/**10000**	**10000**/**10000**
Common Interest	**9971**	**9968**
Gradient 1	8407/**10000**	**9983**/**10000**
Gradient 2	**548**/ **9997**	**502**/ 9431
Heaven and Hell	**9991**/**10000**	**9996**/ **9997**
RO 1	**9368**/**10000**	8593/**10000**
RO 2	**9999**/**10000**	5267/ 6390
RO 3	**7332**/ **7332**	1739/ 1739
Climb	**9999**	9971
Climb-PS	9930	**9990**
Climb-FS	9016	**9217**
Penalty	**9999**	**9993**

Table 6: Comparison of original results from Table 5 with Ranked Boltzmann Selection using $w = 10$. Boldface values indicate methods which were not statistically significantly different with the highest-performing method for a given problem.

is some n times that of B (that is, the ratio between the two is $n : 1$). We note that even for large ratio differences between the two actions, as the Q-values become relatively small, Boltzmann quickly approaches uniform selection (0.5 probability for each). We believe this explains the necessity to change ω for LMRL2 in the Common Interest game: it has very small rewards, hence very small Q-values.

To experiment with this, we designed a new version of Boltzmann Selection called *Ranked Boltzmann Selection*. Here, when selecting an action, we first rank-ordered the possible actions by their current Q-values, breaking ties arbitrarily. We then stretched the rank orderings uniformly under some maximum value w. For example, if there were four actions, their values would be set to $1/4w$, $1/2w$, $3/4w$, and w. We then performed Boltzmann Selection using these resulting values. The point of Ranked Boltzmann Selection is that, given a value of n, it is insensitive to the exact Q-values in question, and thus to "close" Q-values among actions. w is essentially a rigorous, tunable replacement for ω in the ranked case.



(a) Agent One (b) Agent Two

Figure 8: Q-values of actions of two agents using Boltzmann Selection in the Fully Stochastic Climb game, with no minimum temperature. Note that optimal joint action $\langle a, a\rangle$ is abandoned at about step 2500 in favor of suboptimal joint action $\langle c, c\rangle$.

We tested this procedure on the Common Interest game for various values of w from 1 to 100, with 1000 trials each. Figure 7 shows the average rate of complete policy solutions. As can be seen, the algorithm performed much better for certain values than others: and for sufficiently low values of w it performed poorly.

We hoped that Ranked Boltzmann Selection might permit us to eliminate ω entirely as a parameter, since with a good selection of w it would be insensitive to variations among Q-values. To this end, we chose the highest-performing setting from the previous experiment ($w = 10$) and applied it to every test problem. Table 6 compares this against the original results from Table 5. Unfortunately, although some games saw an improvement, many others dropped significantly in performance. Indeed for RO3, which also required a tuned ω value, Ranked Boltzmann Selection performed quite poorly.

Uncoordinated Exploration and Greedy Strategies Uncoordinated exploration in independent learners occurs when one or more agents is selecting an explorative action while the remaining agents are selecting exploitative (greedy) actions. As a result, exploitative agents may receive misleadingly poor reward due to other agents' exploration. As the number of agents increases, this becomes much more likely. For example, with an ϵ-greedy strategy, with n agents in the game, the probability that all agents are exploit is $(1-\epsilon)^n$, and the probability that all the agents are explore is ϵ^n. As n increases, these probabilities become exponentially smaller.

Besides posing a scaling problem, uncoordinated exploration can also cause a difficulty we call *optimal policy destruction*. Here, the agents have found and converged to the optimal policy, but because certain agents then choose to do exploration, the learning process is destabilized and the agents as a whole wind up re-converging to some inferior solution. This may be particularly easy in relative overgeneralization scenarios, and in strongly stochastic games, where optimal solutions can be made to look bad by a single wayward agent or random reward.

Figure 8 shows a situation like this. Here we used a pure Boltzmann Selection strategy (that is, no minimum temperature) in the Fully Stochastic Climb game. We can see that at

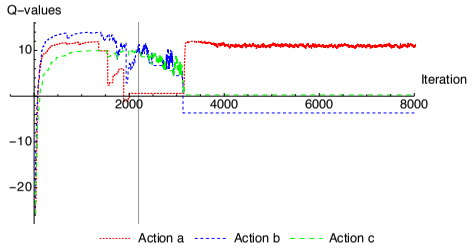


Figure 9: Q-values of actions of one LMRL2 agent in the Fully Stochastic Climb game. The vertical bar indicates where the agent commits to a fully greedy selection strategy, even though the optimal action (a) has not yet been discovered.

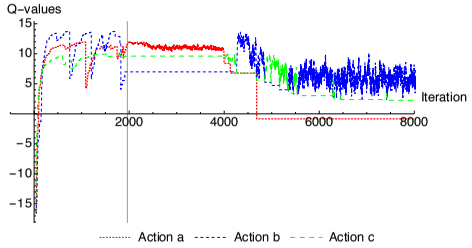


Figure 10: Q-values of actions of one LMRL2 agent in the Fully Stochastic Climb game. The vertical bar indicates where the agent commits to a fully greedy selection strategy.

around 2000 steps, Action a has the highest Q-value for both agents ($\langle a, a\rangle$ is the optimal joint solution). If we switched to fully greedy selection at this point, the agent would stick with this optimal policy. But in this experimental run, the Q-value then decreases enough that it can never recover, and the agents reconverge to $\langle c, c\rangle$. The only plausible source of this optimal policy destruction phenomenon is the uncoordinated exploration from the Boltzmann Selection strategy.

This lock-in would naturally occur when agents are no longer very lenient, but are still fairly explorative. We believe this is the reason that jumping to fully greedy selection strategy is effective in LMRL2: it occurs at the point where optimal policy destruction often shows up. It's possible to converge to the correct solution even when lock-in has occurred *before* discovery of the solution. If the agents lock into suboptimal actions, repeated play may still bring the Q-values low enough that greedy selection then selects a different joint action. This "optimism in the face of uncertainty" (Kaelbling et al., 1996) technique is widely used in many RL algorithms. Figure 9 shows an example run where has this occurred. Action

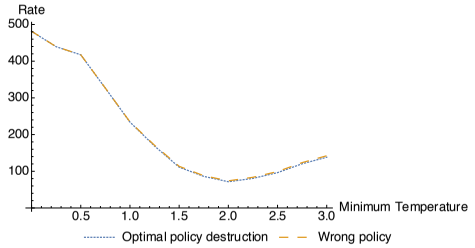


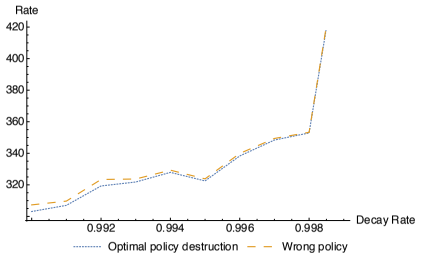
Figure 11: Optimal policy destruction and wrong-policy rates (out of 1000 runs) for different minimum temperature settings. Results shown are the average of ten 1000-run trials.

a is the optimal action. After the greedy selection began, only the action with the highest Q-value is updated (and this is presently not a). Eventually the other actions worsen to the point that a is then tried.

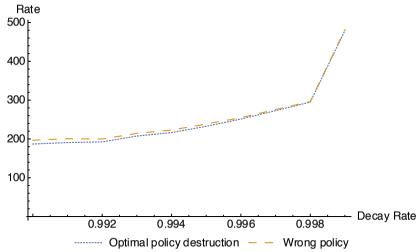
However, this greedy selection lock-in is not a silver bullet for two reasons. First, agents will not adopt the greedy strategy simultaneously: one will become greedy while the other is still exploring. Second, occasionally when the reward function of the optimal Nash Equilibrium has a very large variance, or punishment for uncoordinated behavior is very high, the Q-value of the optimal action can also accidentally be pulled down irrevocably. Figure 10 shows an example. Here we see that the Q-value of action a drops several times after the greedy selection begins, eventually causing other actions to be preferred.

However, generally the greedy strategy usually helps to reduce the rate of optimal policy destruction, particularly when the right minimum temperature is chosen (hence the right lock-in time). Figure 11 shows the optimal policy destruction and incorrect-policy rates, out of 1000 runs, for different minimum temperatures. It is clear that the two rates are strongly linked, and with a suitable minimum temperature (here 2.0), by reducing optimal policy destruction we think we can also eliminate most incorrect policies.

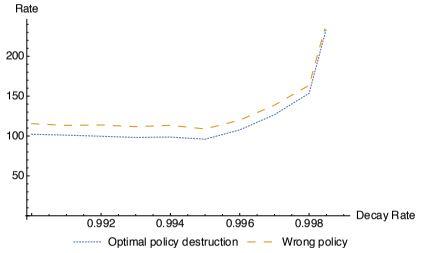
We also studied optimal policy destruction under different combinations of decay rate and minimum temperature. Figure 12 shows the resulting optimal policy destruction and wrong-policy rates. We note that many of the plots form partially bowl-shaped curves, which suggests that there is reasonable range of temperature decay for various minimum temperatures: often this is somewhere around $0.996 - 0.998$. This can be seen as the complement to Figure 11, where we had a range of good minimum temperatures for a fixed decay. However with very low minimum temperatures, the "lowest" points on curve simply slide towards the minimum temperature decay rate. This is expected, since a lower minimum temperature means later lock-in time, and thus a smaller decay rate is preferred. We also note that the difference between the optimal policy destruction and wrong-policy rates can become large, but will rapidly converge after reaching the "lowest" point on the curve. We think the reason for this is that the lower temperature decay rate drops the temperature too rapidly, thus the "optimal action" cannot reach its "optimum" given the short lenience period.



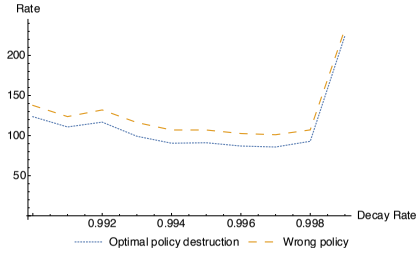
(a) minimum temperature = 0.5



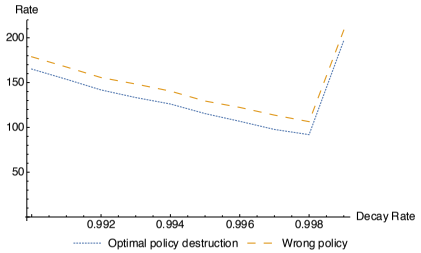
(b) minimum temperature = 1.0



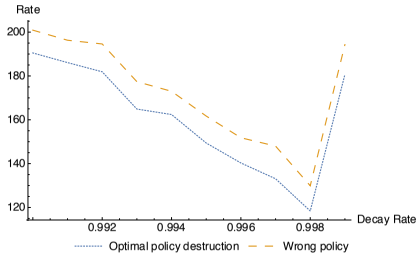
(a) minimum temperature = 1.5



(b) minimum temperature = 2.0



(a) minimum temperature = 2.5



(b) minimum temperature = 3.0

Figure 12: Optimal policy destruction and wrong-policy rates (out of 1000 runs) for different minimum temperature settings with different temperature decay rate. Results shown are the average of ten 1000-run trials.

Test Problem	Without Latching	With Latching
Boutilier	**10000**/**10000**	**10000**/**10000**
Common Interest	**9971**	**9955**
Gradient 1	**8407**/**10000**	7851/ 9834
Gradient 2	**548**/ **9997**	374/ 8775
Heaven and Hell	**9991**/**10000**	**10000**/**10000**
RO 1	9368/**10000**	**9507**/**10000**
RO 2	**9999**/**10000**	**9999**/**10000**
RO 3	**7332**/ **7332**	767/ 767

Table 7: Learning result of using latching in games with same parameter settings.

6.4 Latching

One of the critical features of LMRL2 is its propagation of temperatures from state to state, notionally in order to keep earlier states "hot" long enough to be lenient to late propagated utilities. Given a previous state s, action a, and next state s', this is done as:

$$T(s,a) \leftarrow \delta \times \begin{cases} (1-\tau)T(s,a) + \tau \operatorname{mean}_{a'} T(s',a') & \text{if } s' \text{ is not the end state (if any)} \\ T(s,a) & \text{else} \end{cases}$$

This can not only heat up states, but cool them down as well, which would seem to be an undesirable quality. We have considered an alternative (called *latching*), where states can only be heated up:

$$T(s,a) \leftarrow \delta \times \begin{cases} (1-\tau)T(s,a) + \tau \operatorname{mean}_{a'} T(s',a') & \text{if } s' \text{ is not the end state (if any)} \\ & \quad \text{and } T(s,a) < \operatorname{mean}_{a'} T(s',a') \\ T(s,a) & \text{else} \end{cases}$$

It turns out that this method does not work particularly well. Table 7 compares LMRL2 with and without latching on the various stochastic games from our testbed (latching is irrelevant to repeated games). This is a very counterintuitive result. Our experiment shows that latching only does better in two cases, and considerably worse in the Gradient games and in RO 3. (Note that in Table 7 we purposely maintained the same (difficult) level of Bonferroni correction to be consistent with other results in the paper). Finer-grained examination on a per-state basis (not reported here) did not reveal a conclusive reason as to why.

6.5 Non-self play

Last, we were interested in knowing how well LMRL2 performed when faced with a learning method of some other kind (that is, not in self-play). This is a follow-on to similar work we did using LMRL (Sullivan et al., 2006). To test the performance of LMRL2 when paired with some other learner, we re-ran all the test problems where LMRL2 was paired with some other algorithm. As we wanted to see what would happen to LMRL2 if simply faced with a different kind of learner, and so we did *not* re-tune the parameters of the other agents, but rather kept them at their default settings (from Table 2).

	Alternative Learner					
Test Problem	LMRL2	Q-Learning	Distributed Q	Hysteretic Q	WoLF-PHC	SOoN
Boutilier	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**
Common Interest	**9971**	9888	**9997**	9867	**9977**	9889
Gradient 1	**8407**/**10000**	574/ 1387	7458/ 8266	602/ 1383	63/ 85	1778/ 8601
Gradient 2	**548**/ **9997**	39/ 691	7/ 1119	46/ 706	30/ 376	94/ 2186
Heaven and Hell	**9991**/**10000**	9838/ **9999**	**9981**/**10000**	9830/ **9996**	9787/ **9999**	9855/**10000**
RO 1	9368/**10000**	8031/ 9625	**9998**/ **9998**	7970/ 9617	6554/ 6931	5307/ 7786
RO 2	**9999**/**10000**	0/ 1	6004/**10000**	0/ 0	0/ 0	0/ 14
RO 3	**7332**/ **7332**	899/ 899	2817/ 2817	884/ 884	530/ 530	542/ 542
Climb	**9999**	27	9933	7	0	31
Climb-PS	**9930**	3	56	6	1	26
Climb-FS	**9016**	7	2	24	22	25
Penalty	**9999**	8475	**10000**	8514	9417	**9999**

Table 8: LMRL2 performance when paired with an alternative learning algorithm. Results shown are Complete/Correct solution counts for various test problems. Note that in the Common Interest, Penalty, and various Climb games, correct solutions are by definition complete.

As shown in Table 8, for most of the games and most of the learners, LMRL2 performed best when in self-play. There was one chief exception: for two problems (RO1 and Common Interest), the combination of LMRL2 and Distributed Q-learning outperformed LMRL2 in self-play by a statistically significant margin. To understand why Distributed Q-learning would help LMRL2 in some cases, we plotted the Q-values for the two learners in State 1 of the Common Interest game. State 1 is the state that LMRL2 would normally fail. In State 1, there are two optimal Nash Equilibria, $\langle a, b\rangle$ and $\langle b, a\rangle$, and the primary difficulty in avoiding miscoordination, which which our lenient learner sometimes fails at due to the sensitivity of Boltzmann selection.

The Q-values are shown in Figure 13. It is easy to see that Distributed Q-learning converges much faster than LMRL2. After it has converged, we are effectively dealing with single-agent rather than multiagent reinforcement learning: LMRL2 is just trying to get the best result it can in an essentially fixed environment. This fast convergence doesn't restrict LMRL2's ability to find an optimal Nash Equilibrium, however, since it will converge to which ever Equilibrium Distributed Q-learning has chosen. Here, Distributed Q-learning has effectively forced LMRL2 into its decision.

This result reveals another way we might solve miscoordination. The way Distributed Q-learning normally approaches the miscoordination problem is to make an implicit deal that agents will stick with the first Nash Equilibrium they have chosen. But an alternative would be for one learner to converge to a Nash Equilibrium faster than the other learner, forcing the second learner to adopt its decision.

7. Conclusions

We have introduced LMRL2, a reinforcement learning algorithm for stochastic cooperative independent-learner games, and which can also be used for repeated games. LMRL2 is

Test Problem	LRML2 (Self Play)	Distributed Q (Self Play)	LRML2 + Distributed Q
Boutilier	**10000**/**10000**	**10000**/**10000**	**10000**/**10000**
Common Interest	9971	2080	**9997**
Gradient 1	8407/**10000**	**9999**/ **9999**	7458/ 8266
Gradient 2	**548**/ **9997**	0/ 1266	7/ 1119
Heaven and Hell	**9991**/**10000**	**10000**/**10000**	**9981**/**10000**
RO 1	9368/**10000**	**10000**/**10000**	**9998**/ **9998**
RO 2	**9999**/**10000**	9258/ 9258	6004/**10000**
RO 3	**7332**/ **7332**	2272/ 2272	2817/ 2817
Climb	**9999**	**10000**	9933
Climb-PS	**9930**	2821	56
Climb-FS	**9016**	3874	2
Penalty	**9999**	**10000**	**10000**

Table 9: Comparison of LMRL2 in self play, Distributed Q-Learning in Self-Play, and the scenario where LMRL2 controls one agent and Distributed Q-Learning controls the other. Results shown are Complete/Correct solution counts for various test problems. Note that in the Common Interest, Penalty, and various Climb games, correct solutions are by definition complete.

particularly meant to overcome the *relative overgeneralization* pathology which can appear in cooperative multiagent games of three or more actions per agent, and to do so successfully in the face of stochasticity in the reward or transition function. The algorithm is an extension of an earlier version of ours, LMRL, which was meant only for repeated games.

We compared LMRL2 against several other algorithms which may be applied to cooperative independent-learner games, both stochastic and repeated. This comparison was done over a collection of test problems drawn from the literature and our own devising, and meant to test a wide variety of scenarios and pathologies. Our results have shown that LRML2 is an effective and robust learner, generally outperforming the other algorithms on the test problems in several different ways. LMRL2 placed in the top statistical significance tier for "complete" policies for eight problems, two more times than other algorithms. Further, it placed in the top tier for "correct" policies for eleven problems, three more times than other algorithms. And LMRL2 was uniquely best for two problems. And unlike some other methods, it did so with only a small amount of tuning: though LMRL2 has many available parameters, it performs well even when almost all of them are fixed to default values.

Future Work Because it works on a temperature schedule, LMRL2 is not a particularly fast algorithm. It often takes rather more time to converge than other more greedy methods. Because we are more interested in the converged result than in the speed of learning, this was not a vital issue for us: but as future work, we wish to examine how to speed up LMRL2's convergence without hurting its performance. We also want to investigate other possible stable exploration methods which achieve coordinated exploration: while LMRL2's current exploration strategy is generally fixed, it is nonetheless overcomplicated.

We have also been surprised by our *latching* results. We would have expected to get better performance by restricting pushed-back temperature to only those situations where it would heat up the earlier states, but the results are mixed and disappointing. We intend to examine this more carefully as future work.

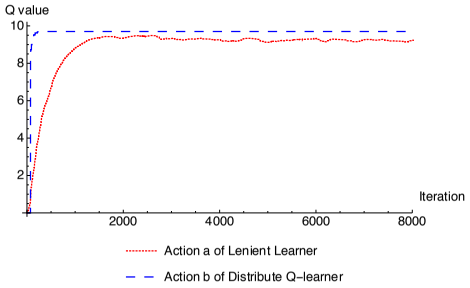


Figure 13: Q-values of two learners in State 1 of Common Interest game

Finally, we note that while the work in this paper reflects somewhat larger and more complex games than are found in much of the MARL literature, it does not scale beyond the literature in important ways. For example, while the algorithm works for any $N \geq 2$ number of agents, the test problems presented are for $N = 2$. In future work, we will extend the research to more complex and more practical domains, scaling both in terms of agent numerosity and heterogeneity, and involving continuous state and action spaces. To this end, we wish to adapt lenience with function approximation and policy gradient methods.

Acknowledgments

The authors thank the reviewers for their feedback. The work presented in this paper is supported by NSF NRI grant 1317813.

Appendix A. Game Defintions

Below we provide tables defining each game in this study, accompanied by illustrations. Each table shows a state S, two agent actions A_1 and A_2, a resulting joint *reward*, and a resulting transition to new state S'.

Actions Actions may be explicitly labeled, such as b or c, or they may have the word *any*, meaning "any action", or (for agent 2) they may have the word *same*, meaning "the same action that agent 1 selected", or they may have the word *other*, meaning "a different action than agent 1 selected". **Boldface** actions indicate members of a complete policy. **Boldface** actions shown in circled states (such as②) indicate members of a correct policy.

States, Rewards, and Transitions The *start state* is always state 1. The absorbing *end state* (if any) is indicated with the word "end". All other states are numbered. If indicated with a single number (or "end"), then rewards and transitions are deterministic. Otherwise, a distribution is shown. For example, 1 (10%), 2 (90%) means "10% of the time choose 1, 90% of the time choose 2".

Illustrations Tables are accompanied by illustrations, which show the games as finite-state automata. States are shown by their numbers: the start state is 1. The absorbing state, if any, is *End*. Each state has a two-player game reward matrix corresponding to the actions

$\langle a,a\rangle$	$\langle a,b\rangle$
$\langle b,a\rangle$	$\langle b,b\rangle$

or

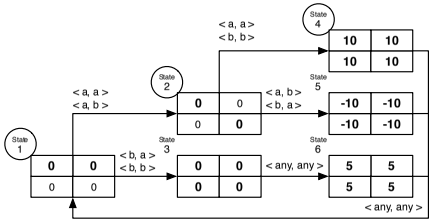
$\langle a,a\rangle$	$\langle a,b\rangle$	$\langle a,c\rangle$
$\langle b,a\rangle$	$\langle b,b\rangle$	$\langle b,c\rangle$
$\langle c,a\rangle$	$\langle c,b\rangle$	$\langle c,c\rangle$

actions for players $\langle A_1, A_2\rangle$ respectively. Values in the matrices indicate rewards for joint actions. If the values are of the form x/y, this indicates that 50% of the time value x is rewarded, and 50% of the time y is rewarded. **Boldface** values indicate joint actions which are part of correct policies. Some states have their state number circled: **boldface** values in these states are part of complete policies. Directed transition edges between states are labeled with the joint actions which trigger those transitions. In some cases a joint action will trigger a transition with a certain probability, else it will trigger some other transition. Transitions may also be labeled ⟨any, any⟩, indicating that all joint actions trigger that transition.

Boutilier

From (Boutilier, 1999).

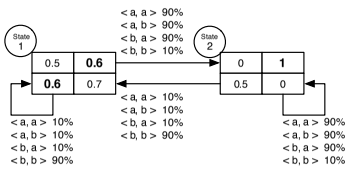
S	A_1	A_2	Reward	S'
①	**a**	**any**	0	2
	b	any	0	3
②	**any**	**same**	0	4
	any	other	0	5
3	**any**	**any**	0	6
④	**any**	**any**	10	1
5	**any**	**any**	-10	1
6	**any**	**any**	5	1



Common Interest

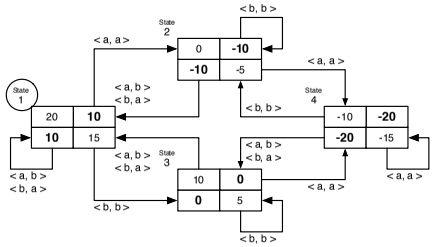
From (Vrancx et al., 2008).

S	A_1	A_2	Reward	S'
①	a	a	0.5	1 (10%), 2 (90%)
	any	**other**	0.6	1 (10%), 2 (90%)
	b	b	0.7	1 (90%), 1 (10%)
②	a	a	0	1 (10%), 2 (90%)
	a	**b**	1.0	1 (10%), 2 (90%)
	b	a	0.5	1 (10%), 2 (90%)
	b	b	0	1 (90%), 1 (10%)



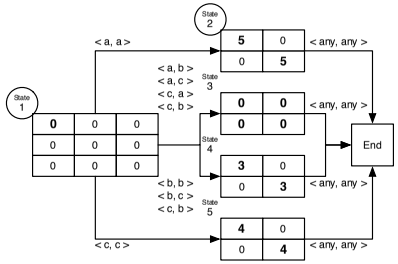
Heaven and Hell

S	A_1	A_2	Reward	S'
①	a	a	20	2
	any	**other**	10	1
	b	b	15	3
2	a	a	0	4
	any	**other**	-10	1
	b	b	-5	2
3	a	a	10	4
	any	**other**	0	1
	b	b	5	2
4	a	a	-10	4
	any	**other**	-20	3
	b	b	-15	2



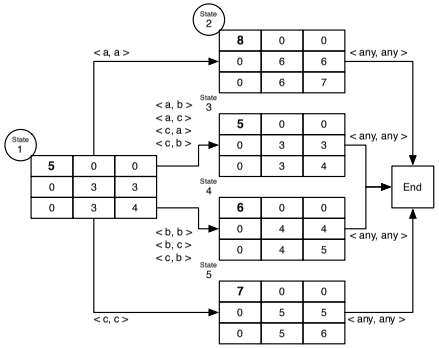
Relative Overgeneralization 1 (RO 1)

S	A_1	A_2	Reward	S'
①	**a**	**a**	0	2
	a	b	0	3
	a	c	0	3
	b	a	0	3
	b	b	0	4
	b	c	0	4
	c	a	0	3
	c	b	0	4
	c	c	0	5
②	**a**	**other**	5	end
	b	other	0	end
3	**any**	**any**	0	end
4	**a**	**other**	3	end
	b	other	0	end
5	**a**	**other**	4	end
	b	other	0	end



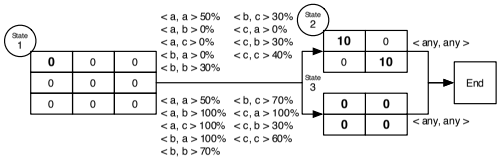
Relative Overgeneralization 2 (RO 2)

S	A_1	A_2	Reward	S'
①	**a**	**a**	5	2
	a	b	0	3
	a	c	0	3
	b	a	0	3
	b	b	3	4
	b	c	3	4
	c	a	0	3
	c	b	3	4
	c	c	4	5
②	a	a	8	end
	a	b	0	end
	a	c	0	end
	b	a	0	end
	b	b	6	end
	b	c	6	end
	c	a	0	end
	c	b	6	end
	c	**c**	7	end
3	a	a	5	end
	a	b	0	end
	a	c	0	end
	b	a	0	end
	b	b	3	end
	b	c	3	end
	c	a	0	end
	c	b	3	end
	c	**c**	4	end
4	a	a	6	end
	a	b	0	end
	a	c	0	end
	b	a	0	end
	b	b	4	end
	b	c	4	end
	c	a	0	end
	c	b	4	end
	c	**c**	5	end
5	a	a	7	end
	a	b	0	end
	a	c	0	end
	b	a	0	end
	b	b	5	end
	b	c	5	end
	c	a	0	end
	c	b	5	end
	c	**c**	6	end



Relative Overgeneralization 3 (RO3 3)

S	A_1	A_2	Reward	S'
	a	**a**	0	2 (50%), 3 (50%)
	a	b	0	3
	a	c	0	3
	b	a	0	3
①	b	b	0	2 (30%), 3 (70%)
	b	c	0	2 (30%), 3 (70%)
	c	a	0	3
	c	b	0	2 (30%), 3 (70%)
	c	c	0	2 (40%), 3 (60%)
②	**a**	**other**	10	end
	b	other	0	end
3	**any**	**any**	0	end

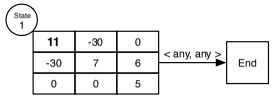


Climb, Climb Partially Stochastic, and Climb Fully Stochastic

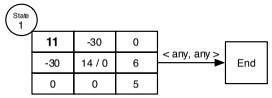
Climb is from (Claus and Boutilier, 1998). Climb Partially Stochastic and Climb Fully Stochastic are from (Kapetanakis and Kudenko, 2002).

			Reward			
				Climb		*Climb*
S	A_1	A_2	*Climb*	*Partially Stochastic*	*Fully Stochastic*	S'
	a	**a**	11	11	10 (50%), 12 (50%)	end
	a	b	-30	-30	5 (50%), -65 (50%)	end
	a	c	0	0	8 (50%), -8 (50%)	end
	b	a	-30	-30	5 (50%), -65 (50%)	end
①	b	b	7	14 (50%), 0 (50%)	14 (50%), 0 (50%)	end
	b	c	6	6	12 (50%), 0 (50%)	end
	c	a	0	0	5 (50%), -5 (50%)	end
	c	b	0	0	5 (50%), -5 (50%)	end
	c	c	5	5	10 (50%), 0 (50%)	end

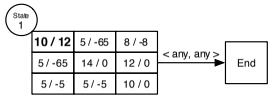
Climb



Climb Partially Stochastic



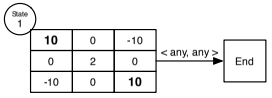
Climb Fully Stochastic



Penalty

From (Claus and Boutilier, 1998).

S	A_1	A_2	Reward	S'
	a	**a**	10	end
	a	b	0	end
	a	c	-10	end
	b	a	0	end
①	b	b	2	end
	b	c	0	end
	c	a	-10	end
	c	b	0	end
	c	**c**	10	end

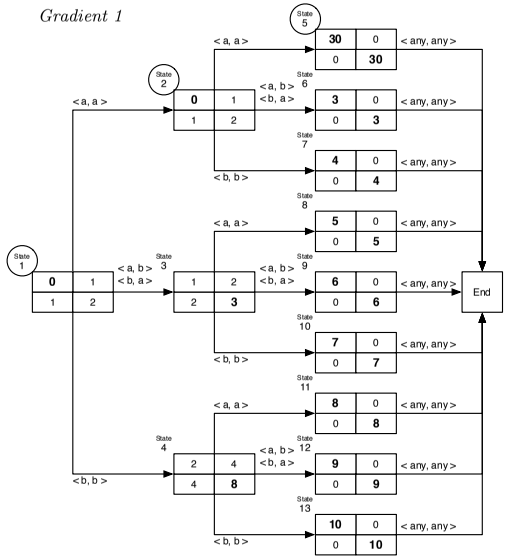


Gradient 1, Gradient 2

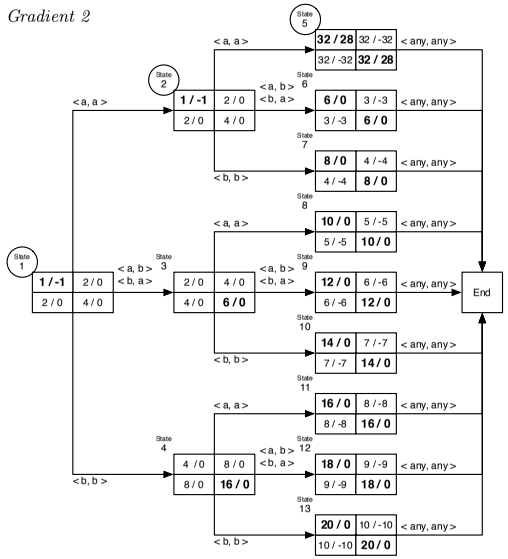
			Reward		
S	A_1	A_2	*Gradient 1*	*Gradient 2*	S'
	a	**a**	0	1 (50%), -1 (50%)	2
①	any	other	1	2 (50%), 0 (50%)	3
	b	b	2	4 (50%), 0 (50%)	4
	a	**a**	0	1 (50%), -1 (50%)	5
②	any	other	1	2 (50%), 0 (50%)	6
	b	b	2	4 (50%), 0 (50%)	7
	a	a	1	2 (50%), 0 (50%)	8
3	any	other	2	4 (50%), 0 (50%)	9
	b	**b**	3	6 (50%), 0 (50%)	10
	a	a	2	(50%), 0 (50%)	11
4	any	other	4	8 (50%), 0 (50%)	12
	b	**b**	8	16 (50%), 0 (50%)	13
⑤	**any**	**same**	30	32 (50%), 28 (50%)	end
	any	other	0	32 (50%), -32 (50%)	end
6	**any**	**same**	3	6 (50%), 0 (50%)	end
	any	other	0	3 (50%), -3 (50%)	end
7	**any**	**same**	4	8 (50%), 0 (50%)	end
	any	other	0	4 (50%), -4 (50%)	end
8	**any**	**same**	5	10 (50%), 0 (50%)	end
	any	other	0	5 (50%), -5 (50%)	end
9	**any**	**same**	6	12 (50%), 0 (50%)	end
	any	other	0	6 (50%), -6 (50%)	end
10	**any**	**same**	7	14 (50%), 0 (50%)	end
	any	other	0	7 (50%), -7 (50%)	end
11	**any**	**same**	8	16 (50%), 0 (50%)	end
	any	other	0	8 (50%), -8 (50%)	end
12	**any**	**same**	9	18 (50%), 0 (50%)	end
	any	other	0	9 (50%), -9 (50%)	end
13	**any**	**same**	10	20 (50%), 0 (50%)	end
	any	other	0	10 (50%), -10 (50%)	end

(Figures on following page)

Gradient 1



Gradient 2



References

Sherief Abdallah and Victor Lesser. A multiagent reinforcement learning algorithm with non-linear dynamics. *Journal of Artificial Intelligence Research*, 33:521–549, 2008.

Noa Agmon, Samuel Barrett, and Peter Stone. Modeling uncertainty in leading ad hoc teams. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 397–404, 2014.

Monica Babes, Munoz Enrique Cote De, and Michael L. Littman. Social reward shaping in the prisoner's dilemma. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 3, pages 1389–1392, 2008.

Bikramjit Banerjee and Jing Peng. Adaptive policy gradient in multiagent learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 686–692, 2003.

Samuel Barrett and Peter Stone. Cooperating with unknown teammates in complex domains: A robot soccer case study of ad hoc teamwork. In *AAAI Conference on Artificial Intelligence*, 2015.

Daan Bloembergen, Michael Kaisers, and Karl Tuyls. Lenient frequency adjusted Q-learning. In *Benelux Conference on Artificial Intelligence (BNAIC)*, 2010.

Craig Boutilier. Sequential optimality and coordination in multiagent systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 1, pages 478–485, 1999.

Michael Bowling. Convergence and no-regret in multiagent learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 17, pages 209–216, 2005.

Michael Bowling and Manuela Veloso. Convergence of gradient dynamics with a variable learning rate. In *International Conference on Machine Learning (ICML)*, pages 27–34, 2001a.

Michael Bowling and Manuela Veloso. Rational and convergent learning in stochastic games. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 2, pages 1021–1026, 2001b.

Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 38(2):156–172, 2008.

Doran Chakraborty and Peter Stone. Cooperating with a markovian ad hoc teammate. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1085–1092, 2013a.

Doran Chakraborty and Peter Stone. Multiagent learning in the presence of memory-bounded agents. *Autonomous Agents and Multiagent Systems*, 2013b.

Georgios Chalkiadakis and Craig Boutilier. Coordination in multiagent reinforcement learning: A Bayesian approach. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 709–716, 2003.

Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *National Conference on Artificial Intelligence*, pages 746–752, 1998.

Vincent Conitzer and Tuomas Sandholm. AWESOME: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1-2):23–43, 2007.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13(1):227–303, November 2000.

Nancy Fulda and Dan Ventura. Predicting and preventing coordination problems in cooperative Q-learning systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 780–785, 2007.

Katie Genter, Tim Laue, and Peter Stone. The robocup 2014 SPL drop-in player competition: Encouraging teamwork without pre-coordination. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1745–1746, 2015.

Mohammad Ghavamzadeh and Sridhar Mahadevan. Learning to communicate and act using hierarchical reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 3, pages 1114–1121, 2004.

Mohammad Ghavamzadeh, Sridhar Mahadevan, and Rajbala Makar. Hierarchical multi-agent reinforcement learning. *Autonomous Agents and Multiagent Systems*, 13(2):197–229, 2006.

Amy Greenwald, Keith Hall, and Roberto Serrano. Correlated Q-learning. In *AAAI Spring Symposium*, volume 3, pages 242–249, 2003.

Junling Hu and Michael P. Wellman. Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003.

Tommi Jaakkola, Michael I. Jordan, and Satinder P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6:1185–1201, 1994.

Amir Jafari, Amy Greenwald, David Gondek, and Gunes Ercal. On no-regret learning, fictitious play, and Nash equilibrium. In *International Conference on Machine Learning (ICML)*, volume 1, pages 226–233, 2001.

Leslie Pack Kaelbling, Michael L. Littman, and Andrew M. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4(1):237–285, 1996.

Michael Kaisers and Karl Tuyls. Frequency adjusted multi-agent Q-learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 1, pages 309–316, 2010.

Spiros Kapetanakis and Daniel Kudenko. Reinforcement learning of coordination in cooperative multi-agent systems. In *National Conference on Artificial Intelligence*, pages 326–331, 2002.

Ville Könönen. Asymmetric multiagent reinforcement learning. *Web Intelligence and Agent Systems*, 2:105–121, 2004.

Martin Lauer and Martin Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *International Conference on Machine Learning (ICML)*, pages 535–542, 2000.

Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*, volume 94, pages 157–163, 1994.

Michael L. Littman. Friend-or-foe Q-learning in general-sum games. In *International Conference on Machine Learning (ICML)*, volume 1, pages 322–328, 2001a.

Michael L. Littman. Value-function reinforcement learning in markov games. *Cognitive Systems Research*, 2(1):55–66, 2001b.

Rajbala Makar, Sridhar Mahadevan, and Mohammad Ghavamzadeh. Hierarchical multi-agent reinforcement learning. In *Proceedings of the 5th International Conference on Autonomous Agents (AGENT)*, pages 246–253, 2001.

Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Hysteretic Q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 64–69. IEEE, 2007.

Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. A study of FMQ heuristic in cooperative multi-agent games. In *Workshop on Multi-Agent Sequential Decision Making in Uncertain Multi-Agent Domains (at AAMAS)*, volume 1, pages 77–91, 2008.

Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Coordination of independent learners in cooperative markov games. Technical report, Institut FEMTO-ST, UniversitPé de Franche-Comté, https://hal.archives-ouvertes.fr/hal-00370889/document, 2009.

Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(01):1–31, 2012.

Liviu Panait. *The Analysis and Design of Concurrent Learning Algorithms for Cooperative Multiagent Systems*. PhD thesis, George Mason University, Fairfax, Virginia, 2006.

Liviu Panait, R. Paul Wiegand, and Sean Luke. A sensitivity analysis of a cooperative coevolutionary algorithm biased for optimization. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 587–584, 2004.

Liviu Panait, Sean Luke, and R. Paul Wiegand. Biasing coevolutionary search for optimal multiagent behaviors. *IEEE Transactions on Evolutionary Computation*, 10(6):629–645, 2006a.

Liviu Panait, Keith Sullivan, and Sean Luke. Lenient learners in cooperative multiagent systems. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2006b.

Liviu Panait, Karl Tuyls, and Sean Luke. Theoretical advantages of lenient learners: an evolutionary game theoretic perspective. *Journal of Machine Learning Research*, 9:423–457, March 2008.

Liviu Panait, Keith Sullivan, and Sean Luke. Lenience towards teammates helps in cooperative multiagent learning. Technical Report GMU-CS-TR-2013-2, Department of Computer Science, George Mason University, 4400 University Drive MSN 4A5, Fairfax, VA 22030-4444 USA, 2013.

Mitchell A. Potter and Kenneth A. De Jong. A cooperative coevolutionary approach to function optimization. In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Männer, editors, *Parallel Problem Solving from Nature III (PPSN)*, volume 866 of *Lecture Notes in Computer Science*, pages 249–257, 1994.

Yoav Shoham, Rob Powers, and Trond Grenager. On the agenda(s) of research on multi-agent learning. In *AAAI Fall Symposium on Artificial Multiagent Learning*, 2004.

Satinder P. Singh, Tommi Jaakkola, Michael L. Littman, and Csaba Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38 (3):287–308, 2000.

Peter Stone and Sarit Kraus. To teach or not to teach?: Decision making under uncertainty in ad hoc teams. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 1, pages 117–124, 2010.

Peter Stone, Gal A. Kaminka, Sarit Kraus, and Jeffrey S. Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *AAAI Conference on Artificial Intelligence*, 2010.

Keith Sullivan, Liviu Panait, Gabriel Balan, and Sean Luke. Can good learners always compensate for poor learners? In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 804–806, 2006.

Gerald Tesauro. Extending Q-learning to general adaptive multi-agent systems. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

Peter Vrancx, Karl Tuyls, and Ronald Westra. Switching dynamics of multi-agent learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 1, pages 307–313, 2008.

Xiaofeng Wang and Tuomas Sandholm. Reinforcement learning to play an optimal nash equilibrium in team Markov games. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1571–1578, 2002.

Michael Weinberg and Jeffrey S. Rosenschein. Best-response multiagent learning in non-stationary environments. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, volume 3, pages 506–513, 2004.

Michael P. Wellman and Junling Hu. Conjectural equilibrium in multiagent learning. *Machine Learning*, 33(2-3):179–200, 1998.

Rudolph Paul Wiegand. *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, Department of Computer Science, George Mason University, 2004.

Aaron Wilson, Alan Fern, and Prasad Tadepalli. Bayesian policy search for multi-agent role discovery. In *AAAI Conference on Artificial Intelligence*, 2010.

Chongjie Zhang and Victor Lesser. Multi-agent learning with policy prediction. In *AAAI Conference on Artificial Intelligence*, pages 927–934, 2010.

Journal of Machine Learning Research 17 (2016) 1-15 Submitted 8/15; Revised 3/16; Published 12/16

Structure-Leveraged Methods in Breast Cancer Risk Prediction

Jun Fan JUNFAN@STAT.WISC.EDU
Department of Statistics
University of Wisconsin-Madison
1300 University Avenue, Madison, WI 53706, United States
Yirong Wu YWU@UWHEALTH.ORG
Department of Radiology
University of Wisconsin-Madison
600 Highland Avenue, Madison, WI 53792, United States
Ming Yuan MYUAN@STAT.WISC.EDU
Department of Statistics
University of Wisconsin-Madison
1300 University Avenue, Madison, WI 53706, United States
David Page PAGE@BIOSTAT.WISC.EDU
Department of Biostatistics and Medical Informatics
University of Wisconsin-Madison
600 Highland Avenue, Madison, WI 53792, United States
Jie Liu LIU6@UW.EDU
Department of Genome Sciences
University of Washington-Seattle
3720 15th Avenue, Seattle, WA 98105, United States
Irene M. Ong ONG@CS.WISC.EDU
Department of Biostatistics and Medical Informatics
University of Wisconsin-Madison
600 Highland Avenue, Madison, WI 53792, United States
Peggy Peissig PEISSIG.PEGGY@MCRF.MFLDCLIN.EDU
Marshfield Clinic Research Foundation
1000 North Oak Avenue, Marshfield, WI 54449, United States
Elizabeth Burnside EBURNSIDE@UWHEALTH.ORG
Department of Radiology
University of Wisconsin-Madison
600 Highland Avenue, Madison, WI 53792, United States

Editor: Benjamin M. Marlin and Suchi Saria

Abstract

Predicting breast cancer risk has long been a goal of medical research in the pursuit of precision medicine. The goal of this study is to develop novel penalized methods to improve breast cancer risk prediction by leveraging structure information in electronic health records. We conducted a retrospective case-control study, garnering 49 mammography descriptors and 77 high-frequency/low-penetrance single-nucleotide polymorphisms (SNPs) from an existing personalized medicine data repository. Structured mammography reports and breast imaging features have long been part of a standard electronic health record (EHR), and genetic markers likely will be in the near future. Lasso and its variants are widely used approaches to integrated learning and feature selection, and our methodological contribution is to incorporate the dependence structure among the features into these approaches. More specifically, we propose a new methodology by combining group penalty and ℓ^p $(1 \leq p \leq 2)$ fusion penalty to improve breast cancer risk prediction, taking into account structure information in mammography descriptors and SNPs. We demonstrate that our method provides benefits that are both statistically significant and potentially significant to people's lives.

Keywords: structure information, breast cancer risk prediction, mammography descriptors, genetic variants, personalized medicine

1. Introduction

Breast cancer is the most common non-skin malignancy affecting women, with approximately 1.67 million cases diagnosed annually worldwide (Ferlay et al., 2013). If an individual's risk of breast cancer could be predicted, then screening, prevention, and treatment strategies could be targeted toward those women to maximize survival benefit and minimize harm. Risk prediction models are important tools to improve breast cancer care by leveraging multi-dimensional electronic health data. Traditional breast cancer risk prediction models use demographic risk factors to estimate breast cancer risk, but they demonstrate only limited discriminatory power. In clinical practice, mammography is the most common breast cancer screening test, and the only imaging modality supported by randomized trials demonstrating reduction in mortality rate. However, its effectiveness is not universally accepted (Freedman et al., 2004). Recent advances in genome-wide association studies (GWAS) have revitalized the quest for genetic variants (single-nucleotide polymorphisms—SNPs) in risk prediction. However, the optimism of these studies has been tempered by disappointment and caution (Gail, 2008, 2009; Wacholder et al., 2010).

Although many breast cancer risk prediction models have been developed, current applications of these models are inadequate in the following respects: (1) due to the rare occurrence of breast cancer, many seemingly 'large' studies have small effective sample size to adequately model a large number of variables; (2) even for large studies, investigators often fail to systematically model risk factor interactions to avoid overly complicated models which are hard to interpret; and (3) they do not take available structure information into consideration. For example, there are five descriptors for mass margins in mammogram: circumscribed, microlobulated, obscured, indistinct, and spiculated, with an order of increasing probability of malignancy. However, few models utilize this structure information (group structure and dependence structure) to improve predictive performance. The quest for novel breast cancer risk prediction models is motivated to address these shortcomings.

In this paper, we propose to develop novel penalized methods to improve breast cancer risk prediction by incorporating unique structure information embedded in electronic health record data. Regularization is a common technique used in regression and classification problems. The lasso (Tibshirani, 1996) is one of the most popular penalized method and

has achieved great success in various fields. However, lasso does not take into account the prior structure information among features. The group lasso (Yuan and Lin, 2006) is a natural extension of the lasso by taking advantage of the underlying group structure of features. It leads to the selection for groups of features and can improve the predictive performance in many real applications such as microarray data analysis (Ma et al., 2007) and GWAS (Liu et al., 2013). To incorporate the dependence structure of features, fused lasso (Tibshirani et al., 2005; Tibshirani and Wang, 2008) is introduced by penalizing the ℓ^1 norm of both the coefficients and their successive differences. To the best of our knowledge, no breast cancer prediction models utilize group penalty and within-group ℓ^p fusion penalty simultaneously to improve risk prediction by leveraging structure information.

The rest of the paper is organized as follows. Section 2 describes our data, proposed methods, and study design. Section 3 presents the results. The conclusions are described in Section 4.

2. Materials and Methods

The main purpose of this paper is to take into account both the group structure and the dependence structure within each group of features by imposing both group penalty and ℓ^p fusion penalty simultaneously.

2.1 Data

The Marshfield Clinic Institutional Review Board approved the use of Marshfield Clinic's Personalized Medicine Research Project (PMRP) (McCarty et al., 2005) cohort in our study.

2.1.1 Subjects

The population-based PMRP cohort, details of which have been previously published (McCarty et al., 2005), was used in this study. Though the details of this population have been described previously (Burnside et al., 2015), we will summarize here, in brief, for the convenience of the reader. Women with an available DNA sample, a mammogram, and a breast biopsy within 12 months after the mammogram were included in the study. For this case/control study, cases were defined as women having a confirmed diagnosis of breast cancer obtained from the institutional cancer registry. Controls were confirmed through the Marshfield Clinic electronic medical records as never having had a breast cancer diagnosis (and absence from cancer registry).

We identified 362 cases and 376 controls (738 in total) who have both genetics and mammogram data available. The majority of mammograms were performed between 1993 and 2005 (Burnside et al., 2015). The age range for the subjects in this study was 29 to 90 years of age, with mean 62 and standard deviation 12.8. Among the cases, there were 358 Caucasians, three non-Caucasians and one case whose race information was unknown. Among the controls, there were 372 Caucasians and four non-Caucasians. These race distributions are consistent with that of the general population in this area. For the family history of breast cancer, we observed a considerably larger proportion of people with family

history in the case group (45.30%) than in the control group (33.51%), which demonstrated the family aggregation of breast cancer (Table 1).

Family history	Cases	Controls	All
Yes	164(45.30%)	126(33.51%)	290(39.30%)
No	188(51.93%)	236(62.77%)	424(57.45%)
N/A	10(2.77%)	14(3.72%)	24(3.25%)

Table 1: Family aggregation of breast cancer.

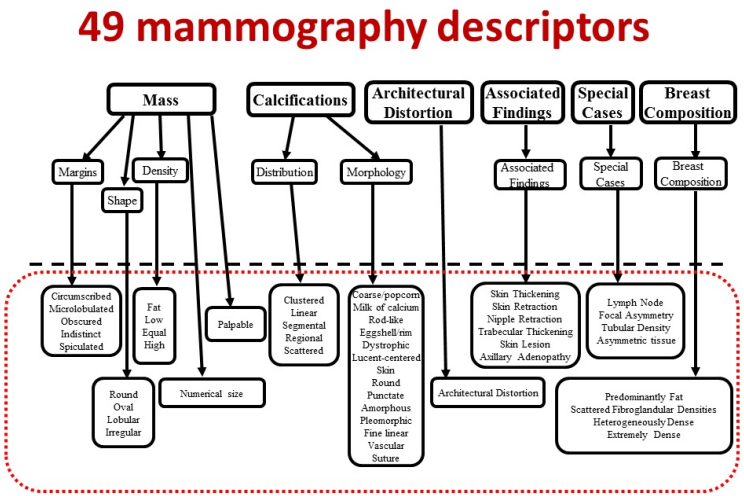


Figure 1: Mammography descriptors described in BI-RADS lexicon.

2.1.2 Mammography features

Mammography features are recorded in the Breast Imaging Reporting and Data System (BI-RADS) lexicon (BI-RADS, 2014) developed by the American College of Radiology. The BI-RADS lexicon consists of 49 descriptors, including the characteristics of masses and microcalcifications, breast composition and other associated findings (Figure 1). In this study, mammography data was recorded as free text reports in the electronic health record, from which we used a parser to extract these mammography features (Nassif et al., 2009).

SNP	Chr	SNP	Chr
rs616488	1	rs11814448	10
rs11249433	1	rs7072776	10
rs1550623	2	rs7904519	10
rs16857609	2	rs2981582	10
rs2016394	2	rs10995190	10
rs4849887	2	rs2380205	10
rs1045485	2	rs2981579	10
rs13387042	2	rs704010	10
rs17468277	2	rs11820646	11
rs4666451	2	rs3903072	11
rs12493607	3	rs3817198	11
rs6762644	3	rs2107425	11
rs4973768	3	rs614367	11
rs6828523	4	rs12422552	12
rs9790517	4	rs17356907	12
rs10472076	5	rs6220	12
rs1353747	5	rs10771399	12
rs1432679	5	rs1292011	12
rs10941679	5	rs11571833	13
rs889312	5	rs2236007	14
rs30099	5	rs2588809	14
rs981782	5	rs941764	14
rs10069690	5	rs999737	14
rs11242675	6	rs13329835	16
rs204247	6	rs17817449	16
rs2046210	6	rs3803662	16
rs2180341	6	rs12443621	16
rs17530068	6	rs8051542	16
rs3757318	6	rs6504950	17
rs720475	7	rs1436904	18
rs11780156	8	rs527616	18
rs2943559	8	rs3760982	19
rs6472903	8	rs4808801	19
rs9693444	8	rs8170	19
rs13281615	8	rs2284378	20
rs10759243	9	rs2823093	21
rs1011970	9	rs132390	22
rs865686	9	rs6001930	22
rs11199914	10		

Table 2: The 77 SNPs identified to be associated with breast cancer

2.1.3 Genetic variants

We decided to focus on high-frequency/low-penetrance SNPs that affect breast cancer risk as opposed to low frequency SNPs with high penetrance or intermediate penetrance. We consolidated a list of 77 common genetic variants (Table 2) which were identified by recent large-scale GWAS studies or used to generate published predictive models (Liu et al., 2014). The list included 41 SNPs identified by COGS through a meta-analysis of 9 GWAS studies (Michailidou et al., 2013). Recently, a similar set of 77 breast cancer-associated SNPs is also studied for risk prediction (Mavaddat et al., 2015).

2.2 Logistic Regression

Assume that we have independent and identical distributed subjects $\{(x_i, y_i)\}_{i=1}^n$, where the explanatory variable $X \in \mathcal{R}^d$ and the binary response variable $Y \in \{-1, 1\}$. Note that the conditional probability $\eta(x) = \mathbb{P}(Y = 1|X = x)$ plays an important role in the classification problem. Denote $x_i = (x_{i1}, ..., x_{id})^T$, and linear logistic regression model is defined by

$$\log \frac{\eta(x_i)}{1 - \eta(x_i)} = x_i^T \beta, \quad i = 1, ..., n,$$

where $\beta = (\beta_1, ..., \beta_d)^T$ is the slope parameter. And the logistic regression estimator $\hat{\beta}$ is given by the minimizer of the negative log-likelihood function

$$L(\beta) = \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i \cdot x_i^T \beta)). \tag{1}$$

With $\hat{\beta}$, we then estimate the conditional probability $\eta(x_i)$ by

$$\hat{\eta}(x_i) = \frac{\exp(x_i^T \hat{\beta})}{1 + \exp(x_i^T \hat{\beta})} = \frac{1}{1 + \exp(-x_i^T \hat{\beta})}.$$

Then we should predict $y_i = 1$ if $\hat{\eta}(x_i) \geq 0.5$ and $y_i = -1$ if $\hat{\eta}(x_i) < 0.5$.

2.3 Group Penalty and ℓ^p Fusion Penalty

Note that there exist natural group structure and dependence structure in mammography features (Figure 1), which allows us to include the structure information into our risk prediction models directly. For genetic variants, group structures also exist (Liu et al., 2012, 2013). In this paper, we apply hierarchical clustering to cluster the 77 SNPs based on their dissimilarity matrix obtained by computing Spearman's correlation or Hamming distance among them. More details are provided in Section 2.5.

Suppose that d features are divided into G groups with d_g the number of features in group g. Define $\beta_g \in \mathbb{R}^{d_g}$ to be the corresponding coefficient vector in group g. The group lasso logistic regression (Meier et al., 2008) is defined as the following optimization problem

$$\min_{\beta \in \mathbb{R}^d} \left\{ L(\beta) + \lambda_1 \sum_{g=1}^{G} \sqrt{d_g} \|\beta_g\|_2 \right\},$$

where $L(\beta)$ is defined by (1) and $\lambda_1 \geq 0$ is the tuning parameter. It includes lasso as a special case with $G = d$.

The fact that there exist dependence structure within each mammography feature group and each SNP group encourages us to propose the following novel method by combining group lasso logistic regression and ℓ^p fusion penalty.

$$\min_{\beta\in\mathbb{R}^d}\left\{L(\beta)+\sum_{g=1}^{G}\left(\lambda_1\sqrt{d_g}\|\beta_g\|_2+\lambda_2\|D_g\beta_g\|_p^p\right)\right\}, \tag{2}$$

where D_g is a $(d_g-1)\times d_g$ sparse matrix with only $D[i,i]=1$ and $D[i,i+1]=-1$, $\lambda_2\geq 0$ is the tuning parameter, and $1\leq p\leq 2$ is the shrinkage parameter.

Moreover, if the within-group dependence structures are different for groups $\{1,...,G_1\}$ and $\{G_1+1,...,G\}$, we can split the ℓ^p fusion penalty into two parts as

$$\min_{\beta\in\mathbb{R}^d}\left\{L(\beta)+\lambda_1\sum_{g=1}^{G}\sqrt{d_g}\|\beta_g\|_2+\lambda_2\left(\sum_{g=1}^{G_1}\|D_g\beta_g\|_{p_1}^{p_1}+\sum_{g=G_1+1}^{G}\|D_g\beta_g\|_{p_2}^{p_2}\right)\right\}, \tag{3}$$

where $1\leq p_1,p_2\leq 2$ are selected based on cross validation.

The novelty of our method compared to previous works is three-fold: First, it includes within-group fusion penalty in the model and makes the coefficients of features in the same group close to each other, which reflects the dependence structure of features and improves the risk prediction; Second, in breast cancer risk prediction, we find that the dependence structures are different for mammography features and SNPs, which are actually two different views of the same data. And the utilization of method (3) will improve the predictive performance further; At last, we find that genetic variants improve risk prediction on mammography features, which provides some insight regarding personalized breast cancer diagnosis.

2.4 Computational Algorithms

Many algorithms have been proposed in the literatures to solve the logistic regression with fused lasso regularization (Lin, 2015; Yu et al., 2015). In this subsection we adopt the fast iterative shrinkage thresholding algorithm (Beck and Teboulle, 2009) to solve (2) as

$$\beta^{k+1}=\arg\min_{\beta\in\mathbb{R}^d}L(\beta^k)+\langle\beta-\beta^k,\nabla L(\beta^k)\rangle+\frac{\tau}{2}\|\beta-\beta^k\|_2^2+\sum_{g=1}^{G}\left(\lambda_1\sqrt{d_g}\|\beta_g\|_2+\lambda_2\|D_g\beta_g\|_p^p\right)$$

with $\beta=(\beta_1,\cdots,\beta_d)^T$ and $\tau>0$ the Lipschitz constant of $L(\cdot)$.

And the iteration step is equivalent to solving

$$\min_{\beta\in\mathbb{R}^d}\left\{\frac{1}{2}\|\beta-(\beta^k-\frac{1}{\tau}\nabla L(\beta^k))\|_2^2+\sum_{g=1}^{G}\left(\frac{\lambda_1\sqrt{d_g}}{\tau}\|\beta_g\|_2+\frac{\lambda_2}{\tau}\|D_g\beta_g\|_p^p\right)\right\}. \tag{4}$$

Therefore, it suffices to solve the following optimization problem within each group

$$\min_{\beta_g\in\mathbb{R}^{d_g}}\left\{\frac{1}{2}\|\beta_g-z\|_2^2+\rho_1\|\beta_g\|_2+\rho_2\|D_g\beta_g\|_p^p\right\}, \tag{5}$$

where $z=\beta_g^k-\frac{1}{\tau}\nabla L(\beta_g^k)$, $\rho_1=\frac{\lambda_1\sqrt{d_g}}{\tau}$ and $\rho_2=\frac{\lambda_2}{\tau}$.

The proximity operator (Polson et al., 2015) of a function f is defined as

$$P_f(z)=\arg\min_t\left\{\frac{1}{2}\|t-z\|^2+\lambda f(t)\right\}.$$

- For $f(t)=|t|$ and $z\in\mathbb{R}$, $P_f(z):=S_1(z,\lambda)=sign(z)\max\{|z|-\lambda,0\}$, which is also called soft threshold operator.
- For $f(t)=|t|^p$ with $1<p\leq 2$ and $z\in\mathbb{R}$, $P_f(z):=S_p(z,\lambda)=sign(z)\xi$, where ξ is the unique nonnegative solution to $\xi+p\lambda\xi^{p-1}=|\xi|$. In particular, we have $S_2(z,\lambda)=\frac{z}{2\lambda+1}$, $S_{3/2}(z,\lambda)=z+9\lambda^2 sign(z)(1-\sqrt{1+16|z|/(9\lambda^2)})/8$ and $S_{4/3}(z,\lambda)=z+\frac{4\lambda}{32^{\frac{1}{3}}}((\chi-z)^{1/3}-(\chi+z)^{1/3})$ with $\chi=\sqrt{z^2+256\lambda^3/729}$.
- For $f(t)=\|t\|_2$ and $z\in\mathbb{R}^d$, $P_f(z):=S_{2,1}(z,\lambda)=\max\{1-\frac{\lambda}{\|z\|_2},0\}*z$.

With the help of these proximity operators and Bregman splitting algorithm (Ye and Xie, 2011), we can solve (5) by iteratively solving the following procedures:

$$\begin{cases}\beta^{k+1}=\arg\min_{\beta_g}\frac{1}{2}\|\beta_g-z\|_2^2+\langle u^k,\beta_g-a^k\rangle+\langle v^k,D_g\beta_g-b^k\rangle\\ \qquad\qquad +\frac{\mu}{2}\|\beta_g-a^k\|_2^2+\frac{\mu}{2}\|D_g\beta_g-b^k\|_2^2\\ a^{k+1}=\arg\min_a\rho_1\|a\|_2+\langle u^k,\beta^{k+1}-a\rangle+\frac{\mu}{2}\|\beta^{k+1}-a\|_2^2\\ b^{k+1}=\arg\min_b\rho_2\|b\|_p^p+\langle v^k,D_g\beta^{k+1}-b\rangle+\frac{\mu}{2}\|D_g\beta^{k+1}-b\|_2^2\\ u^{k+1}=u^k+\mu(\beta^{k+1}-a^{k+1})\\ v^{k+1}=v^k+\mu(D_g\beta^{k+1}-b^{k+1})\end{cases}$$

where μ acts like a step size in this algorithm.

Remark 1 *The minimization over β, a and b can all be solved in closed form.*

- $\beta^{k+1}=[(\mu+1)I+\mu D_g^TD_g]^{-1}[z+\mu(a^k-u^k/\mu)+\mu D_g^T(b^k-v^k/\mu)]$
- $a^{k+1}=S_{2,1}(\beta^{k+1}+u^k/\mu,\rho_1/\mu)$
- $b^{k+1}=S_p(D_g\beta^{k+1}+v^k/\mu,\rho_2/\mu)$

Note that $(\mu+1)I+\mu D_g^TD_g$ is a tridiagonal positive definite matrix.

Remark 2 *For $p=1$, we can solve (5) more efficiently by the algorithm proposed in Zhou et al. (2012) based on the fact*

$$P_{\|\cdot\|_2+\|D_g(\cdot)\|_1}=P_{\|\cdot\|_2}\circ P_{\|D_g(\cdot)\|_1}.$$

However, we cannot show this equation for $1<p\leq 2$.

Remark 3 *For $p = 2$, since $\|\cdot\|_2^2$ is Lipschitz continuous, we can rewrite (4) as*

$$\min_{\beta\in\mathbb{R}^d}\left\{\frac{1}{2}\left\|\beta-\left(\beta^k-\frac{1}{\tilde{\tau}}(\nabla L(\beta^k)+2\lambda_2\sum_{g=1}^{G}D_g^TD_g\beta_g^k)\right)\right\|_2^2+\sum_{g=1}^{G}\frac{\lambda_1\sqrt{d_g}}{\tilde{\tau}}\|\beta_g\|_2\right\},$$

where $\tilde{\tau}$ is the Lipschitz constant of $L(\beta)+\lambda_2\sum_{g=1}^{G}\|D_g\beta_g\|_2^2$. Then we can solve it efficiently via the proximity operator of $\|\cdot\|_2$.

2.5 Study Design and Statistical Analysis

We apply the ℓ^p fused group lasso logistic regression algorithm to the Marshfield breast cancer data set. There are 11 groups for 49 mammography features (Figure 1). For SNPs, we compute the Hamming distances (Wang et al., 2015) of 77 SNPs to get the dissimilarity matrix and then apply hierarchical clustering to obtain 10 groups.

We built three prediction models based on different sets of risk factors: the Mammo model developed by using mammography features only, the SNP77 model developed by using 77 SNPs only, and the Combined model developed by using both mammography features and 77 SNPs. We furthermore apply five methods for each model: logistic regression (LR), lasso in logistic regression (LR+Lasso), ℓ^p fused lasso logistic regression (LR+fusedLasso), group lasso logistic regression (LR+groupLasso), and ℓ^p fused group lasso logistic regression (LR+Structure).

The ℓ^p fused group lasso logistic regression method has several parameters. For the tuning parameters λ_1 and λ_2, we let them vary among a given set of values, and the shrinkage parameter p (or p_1 and p_2) among $\{1, 4/3, 3/2, 2\}$. Each combination of these parameters is evaluated using stratified 5-fold cross-validation, and AUC (the area under the receiver operating characteristic (ROC) curve) is used as the performance measure. All 738 samples are randomly partitioned into five equal sized folds with approximately equal proportions of cases and controls. In each iteration (totally five iterations), four folds are used as training set and the rest one as validation set to compute AUC. And the parameters with the best average AUC are selected. At last we repeat this process ten times and report the average AUC. We obtain p-value by performing two-tailed-two-sample t-test when we compare AUCs.

3. Experimental Results

In this section, we demonstrate the performance of the ℓ^p fused group lasso logistic regression method from three aspects: the significant improvement of AUCs by considering the structure information, the predictive performance under different p (or p_1 and p_2), and the detected important mammography features and SNPs.

3.1 Performance of Fused Group Lasso

The result is summarized in Table 3.

Models/Methods	LR	Lasso	fusedLasso	groupLasso	Structure	p-value
Mammo	0.700	0.710	0.710	0.716	0.723	< 0.001
SNP77	0.590	0.598	0.676	0.614	0.684	< 0.001
Combined	0.697	0.721	0.754	0.727	0.766	< 0.001

Table 3: Predictive performance of three prediction models by using five different methods. The p-values represent the differences between AUCs of LR and LR+Structure.

1) The fifth column describes the predictive performance of the three prediction models by considering structure information in the logistic regression method. We find that the predictive performance of the three prediction models has been improved respectively, compared to those described in the first column. For each prediction model, the difference of the predictive performance is significant between LR+Structure and LR (p-value < 0.001), which demonstrates that breast cancer prediction models utilizing structure information can improve risk prediction significantly. We also find that mammography descriptors demonstrate a significantly higher predictive performance than 77 SNPs in terms of AUC (0.723 vs. 0.684, p-value < 0.001). The Combined model demonstrates significant improvement of the prediction performance, compared to the Mammo model (0.766 vs. 0.723, p-value < 0.001).

2) The first column describes the predictive performance of the three prediction models by using the logistic regression method. Mammography descriptors demonstrate a significantly higher predictive performance than 77 SNPs in terms of AUC (0.700 vs. 0.590, p-value$<$ 0.001). We find that the difference of predictive performance between the Combined model and the Mammo model is negligible (0.697 vs. 0.700, p-value=0.277).

3) The second column describes the predictive performance of the three prediction models by using lasso in the logistic regression method. The predictive performance of the three prediction models has been improved, compared to those without lasso (using logistic regression method only). Mammography descriptors still demonstrate a significantly higher predictive performance than 77 SNPs in terms of AUC (0.710 vs. 0.598, p-value$<$ 0.001). However, the Combined model demonstrates modest improvement of prediction performance, compared to the Mammo model (0.721 vs. 0.710, p-value=0.0057).

4) The third and fourth columns describe the predictive performance of the three prediction models by considering group structure or dependence structure in the logistic regression method. For the SNP77 model, fused lasso demonstrates a significantly higher performance than group lasso in terms of AUC (0.676 vs. 0.614, p-value$<$ 0.001). For the Mammo model, group lasso plays a more important role than fused lasso (0.716 vs. 0.710, p-value=0.0073). Moreover, both fused lasso and group lasso demonstrate improved prediction performance compared to lasso.

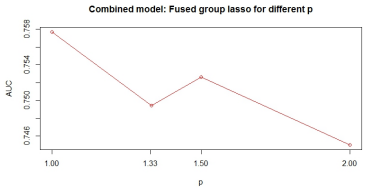
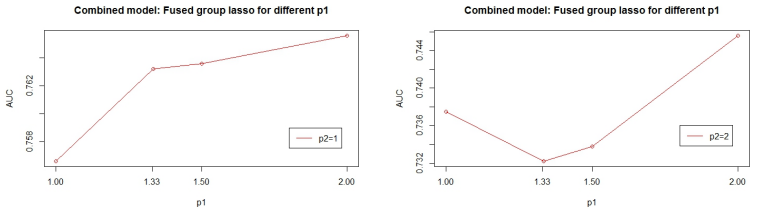


Figure 2: The AUCs under different values of p by using method (2).



(a) Fix $p_2 = 1$ (b) Fix $p_2 = 2$

(c) Fix $p_1 = 1$ (d) Fix $p_1 = 2$

Figure 3: The AUCs under different values of p_1 and p_2 by using method (3).

3.2 Performance under Different Values of p

Figure 2 and Figure 3 describe the the pattern of predictive performance for ℓ^p fused group lasso logistic regression over the shrinkage parameter p (or p_1 and p_2) in terms of AUC.

1)The Combined model demonstrates a higher predictive performance for $p = 1$ compared to $p = 2$ in terms of AUC (0.757 vs. 0.745, p-value$<$ 0.001), see Figure 2.

2) Figure 3 describes the prediction performance of method (3) under different values of p_1 for mammography descriptors and p_2 for 77 SNPs.

- Fix $p_2 = 1$ or $p_2 = 2$, the fused group lasso with $p_1 = 2$ demonstrates higher predictive performance compared to $p_1 = 1$, see Figure 3(a) and 3(b).
- Fix $p_1 = 1$ or $p_1 = 2$, the predictive performance of the fused group lasso logistic regression decreases as p_2 increases, see Figure 3(c) and 3(d).
- The fused group lasso logistic regression with $p_1 = 2$ and $p_2 = 1$ demonstrates higher predictive performance than $p_1 = p_2 = 1$ (0.766 vs. 0.757, p-value=0.0053) and $p_1 = p_2 = 2$ (0.766 vs. 0.745, p-value$<$ 0.001).

3.3 Important Features Detected by Fused Group Lasso

To take into account both group and dependence structure information in mammography features and SNPs, two penalty terms (group penalty and fusion penalty) are introduced into the logistic regression model. The idea of group penalty is to force the coefficients of features in the same group to be all zero or nonzero in order to achieve the goal of selecting features within a group simultaneously. The idea of fusion penalty is to shrink the successive difference of coefficients of features in the same group in order to take advantage of the dependence structure information. Applying fusion penalty with $p = 1$ tends to result in zero successive difference of coefficients, while $p = 2$ tends to small but nonzero successive difference of coefficients.

From a feature selection point of view, we can get the order of feature groups selected by fused group lasso via choosing the tuning parameters appropriately. We list below the feature groups selected from high to low in terms of predictive performance.

1) For mammography descriptors, the following features are predictive of malignancy (from most to least): "Mass Size", "Mass Margins", "Mass Shape", "Architectural Distortion" and "Mass Palpability", consistent with the literature (BI-RADS, 2014).

2) For 77 SNPs, three groups are selected in order, see Table 4.

Feature Group	SNPs
Group 1	rs2016394 rs1432679, rs13281615, rs4666451 rs981782, rs1292011, rs1436904, rs527616
Group 2	rs11249433 rs13387042, rs4973768, rs10069690 rs7904519, rs8051542, rs3760982
Group 3	rs2981579, rs2981582

Table 4: SNP groups selected by fused group lasso.

Remark 4 *It verifies that "Mass size", "Mass Margins" and "Mass Shape" are the most important mammography descriptors in breast cancer diagnosis. These results are consistent with previous studies about comparing the importance of mammography features and SNPs in breast cancer risk prediction(Wu et al., 2013, 2014).*

4. Discussion and Conclusions

This study demonstrates that models utilizing the novel combination of clinically relevant structure and ℓ^p fused group lasso logistic regression can improve breast cancer risk prediction significantly. Our study also shows that both mammography features and SNPs contribute to this improvement.

The structure information of the mammography features is derived from the BI-RADS lexicon, which is used widely in breast imaging practice. Thus, our model would likely be generalizable to other practices. On the other hand, we extracted the structure information of SNPs by computing Hamming distances (Wang et al., 2015). This method may not perform as well in small sample sizes, which may affect our results perhaps making our predictive performance results conservative.

Our methods for SNPs may not take advantage of biological knowledge that currently exists. For example, it may be possible to utilize the biological information available in HapMap (which encodes linkage disequilibrium) to more accurately emulate the patterns or dependence structure of SNPs, as in (Liu et al., 2012). Furthermore, we realize that taking into account more complicated structure information such as graph or tree structure (Sun and Wang, 2012) may further improve predictive performance of risk prediction models. We leave these promising directions for future work.

In conclusion, our results demonstrate that including structure information in the computational methods we test improves breast cancer risk prediction. Our models use diverse breast cancer risk factors including demographics, genetics, and imaging and leverage structure found in a standardized lexicon that is universally captured in electronic health records (EHRs) throughout the US. This information will increasingly be combined in complex ways. Merging imaging features, clinical notes and genetic data with models that accurately predict disease risk has the potential to provide powerful knowledge to practicing physicians.

Acknowledgments

The authors acknowledge the support of the Wisconsin Genomics Initiative, NCI grant R01CA127379-01 and its ARRA supplement R01CA127379-03S1, NIGMS grant R01GM097618-01, NLM grant R01LM011028-01, NIEHS grant 5R01ES017400-03, NIH grant 1U54AI117924-01, NIH grant K24CA194251 and NSF FRG grant DMS-1265202. We also acknowledge support from the Clinical and Translational Science Award (CTSA) program through the NIH National Center for Advancing Translational Sciences (NCATS) grant UL1TR000427 and the University of Wisconsin Carbone Cancer Center Cancer Center Support Grant P30CA014520. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH. We thank the anonymous reviewers for their valuable comments and suggestions.

References

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183-202, 2009.

Breast Imaging Reporting And Data System (BI-RADS). 5th ed. Reston VA: American College of Radiology, 2014.

E. S. Burnside, J. Liu, Y. Wu, A. A. Onitilo, C. A. McCarty, C. D. Page, P. Peissig, A. Trentham-Dietz, T. Kitchner, J. Fan, and M. Yuan. Comparing Mammography Abnormality Features and Genetic Variants in the Prediction of Breast Cancer in Women Recommended for Breast Biopsy. *Academic Radiology*, 23(1):62-9, 2016.

J. Ferlay, I. Soerjomataram, M. Ervik, et al. *GLOBOCAN 2012 cancer incidence and mortality worldwide: IARC cancerbase No. 11*. Lyon, France: International Agency for Research on Cancer, 2013.

D. Freedman, D. Petitti, and J. Robins. On the efficacy of screening for breast cancer. *Int J Epidemiol.*, 33(1):43-55, 2004.

M. Gail. Discriminatory accuracy from single-nucleotide polymorphisms in models to predict breast cancer risk. *J Natl Cancer Inst.*, 100(14):1037-41, 2008.

M. Gail. Value of adding single-nucleotide polymorphism genotypes to a breast cancer risk model. *J Natl Cancer Inst.*, 101(13):959-63, 2009.

T. Lin, S. Ma, and S. Zhang. An extragradient-based alternating direction method for convex minimization. *Found Comput Math*, DOI 10.1007/s10208-015-9282-8, 2015.

J. Liu, J. Huang, S. Ma, and K. Wang. Incorporating group correlations in genome-wide association studies using smoothed group lasso. *Biostatistics*, 14:205-219, 2013.

J. Liu, C. D. Page, P. L. Peissig, et al. New genetic variants improve personalized breast cancer diagnosis. *AMIA Summit on Translational Bioinformatics (AMIA-TBI)*, 2014.

J. Liu, C. Zhang, C. McCarty, P. L. Peissig, E. S. Burnside, and D. Page. Graphical-model based multiple testing under dependence, with applications to genome-wide association studies. In *Proceedings of the 28th conference on uncertainty in artificial intelligence*, 2012.

S. Ma, X. Song, and J. Huang. Supervised group Lasso with applications to microarray data analysis. *BMC bioinformatics*, 8:60, 2007.

N. Mavaddat, et al. Prediction of breast cancer risk based on profiling with common genetic variants. *J Natl Cancer Inst*, 107(5):djv036, 2015.

C. McCarty, R. Wilke, P. Giampietro, S. Wesbrook, and M. Caldwell. Marshfield Clinic Personalized Medicine Research Project (PMRP): design, methods and recruitment for a large population-based biobank. *Personalized Med.*, 2(1):49-79, 2005.

L. Meier, S. Van De Geer, and P. Bohlmann. The Group Lasso for logistic regression. *J. R. Statist. Soc. B*, 70:53-71, 2008.

K. Michailidou, P. Hall, A. Gonzalez-Neira, et al. Large-scale genotyping identifies 41 new loci associated with breast cancer risk. *Nat Genet.*, 45(4):353-61, 2013.

H. Nassif, R. Woods, E. S. Burnside, M. Ayvaci, J. Shavlik, C. D. Page. Information extraction for clinical data mining: a mammography case study. *IEEE International Conference on Data Mining Workshops*, 2009.

N. G. Polson, J. G. Scott, and B. T. Willard. Proximal Algorithms in Statistics and Machine Learning. *Statistical Science*, 30(4):559-581, 2015.

H. Sun and S. Wang. Penalized logistic regression for high-dimensional DNA methylation data with case-control studies. *Bioinformatics*, 28(10):1368-1375, 2012.

R. Tibshirani. Regression shrinkage and selection via the lasso. *J. R. Statist. Soc. B*, 58:267-288, 1996.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *J. R. Statist. Soc. B*, 67:91-108, 2005.

R. Tibshirani and P. Wang. Spatial smoothing and hot spot detection for CGH data using the fused lasso. *Biostatistics*, 9(1):18-29, 2008.

S. Wacholder, P. Hartge, R. Prentice, et al. Performance of common genetic variants in breast-cancer risk models. *N Engl J Med.*, 362(11):986-93, 2010.

C. Wang, W. H. Kao, and C. K. Hsiao. Using Hamming distance as information for SNP-sets clustering and testing in disease association studies. *PLoS One*, 10(8), 2015.

Y. Wu, O. Alagoz, M. Ayvaci, A. M. del Rio, D. J. Vanness, R. Woods, and E. S. Burnside. A comprehensive methodology for determining the most informative mammographic features. *J Digit Imaging*, 26(5):941-947, 2013.

Y. Wu, J. Liu, C. D. Page, P. L. Peissig, C. A. McCarty, A. A. Onitilo, and E. S. Burnside. Comparing the Value of Mammographic Features and Genetic Variants in Breast Cancer Risk Prediction. *AMIA Annu Symp Proc.*, 1228-1237, 2014.

G. Ye and X. Xie. Split Bregman method for large scale fused Lasso. *Computational Statistics and Data Analysis*, 55(4):1552-1569, 2011.

D. Yu, S. Lee, W. Lee, S. Kim, J. Lim, and S. Kwon. Classification of spectral data using fused lasso logistic regression. *Chemometrics and Intelligent Laboratory Systems*, 142:70-77, 2015.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *J. R. Statist. Soc. B*, 68:49-67, 2006.

J. Zhou, J. Liu, V. A. Narayan, and J. Ye. Modeling disease progression via fused sparse group lasso. In *KDD*, pages 1095-1103, 2012.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 9/15; Published 5/16

LIBMF: A Library for Parallel Matrix Factorization in Shared-memory Systems

Wei-Sheng Chin d01944006@csie.ntu.edu.tw
Bo-Wen Yuan r03944049@csie.ntu.edu.tw
Meng-Yuan Yang b01902037@csie.ntu.edu.tw
Yong Zhuang r01922139@csie.ntu.edu.tw
Yu-Chin Juan r01922136@csie.ntu.edu.tw
Chih-Jen Lin cjlin@csie.ntu.edu.tw
Department of Computer Science
National Taiwan University
Taipei, Taiwan

Editor: Mark Reid

Abstract

Matrix factorization (MF) plays a key role in many applications such as recommender systems and computer vision, but MF may take long running time for handling large matrices commonly seen in the big data era. Many parallel techniques have been proposed to reduce the running time, but few parallel MF packages are available. Therefore, we present an open source library, LIBMF, based on recent advances of parallel MF for shared-memory systems. LIBMF includes easy-to-use command-line tools, interfaces to C/C++ languages, and comprehensive documentation. Our experiments demonstrate that LIBMF outperforms state of the art packages. LIBMF is BSD-licensed, so users can freely use, modify, and redistribute the code.

Keywords: Matrix factorization, non-negative matrix factorization, binary matrix factorization, logistic matrix factorization, one-class matrix factorization, stochastic gradient method, adaptive learning rate, parallel computation

1. Introduction

Matrix factorization (MF) and its variants cover a wide range of applications including recommender systems, link prediction, image processing, and document clustering. Although parallel MF has been widely investigated in recent years, it is still not easy to find an MF package supporting both rich formulations and parallel computation. For example, the parallel MF algorithm in LIBPMF[1] is known to be state of the art, but LIBPMF can only solve real-valued matrix factorization with a squared loss and L2-norm regularization. Some packages (e.g., scikit-learn,[2] mlpack,[3] and nimfa[4]) support a variety of MF problems, but unfortunately they do not implement parallel MF algorithms published in recent five years. We thus develop LIBMF for solving a family of MF problems based on the techniques proposed in Chin et al. (2015a) and Chin et al. (2015b) targeting at shared-memory systems with multi-core CPUs (e.g., mordern PCs). Many computational issues were addressed to make LIBMF efficient. Our experiments show that LIBMF is faster than state of the art packages. Moreover, LIBMF allows users to use their disk as a buffer to factorize a huge matrix that may not fit into their memory. LIBMF is released under BSD license at

`http://www.csie.ntu.edu.tw/~cjlin/libmf/` .

This paper is organized as follows. Section 2 introduces LIBMF in detail. Problems solved by LIBMF are presented in Section 2.1 and we demonstrate the practical usage in Section 2.2. The documentation is summarized in Section 2.3. Comparisons of LIBMF and state of the art packages are shown in Section 3. Finally, we conclude in Section 4. We provide the implementation details in a supplementary document.[5]

2. The Software Package

LIBMF provides efficient parallel MF solvers with rich documentation for a family of MF problems. For practical usage, users can directly run LIBMF using command-line instructions, make a C/C++ function call, or use the third-party R interface. LIBMF is portable by following the C++11 standard and can be complied on Unix-like systems as well as Microsoft Windows.

2.1 Problems Solved by LIBMF

In general, MF is a process to find two factor matrices, $P \in \mathbb{R}^{k\times m}$ and $Q \in \mathbb{R}^{k\times n}$, to describe a given m-by-n training matrix R in which some entries may be missing. MF can be found in many applications, but we only use collaborative filtering in recommender systems as examples. We further assume that the entries of R are the historical users' preferences for merchandises, and the task on hand is to predict unobserved user behavior (i.e., missing entries in R) to make a suitable recommendation. Let u and v stand for row index and column index, respectively. For rating prediction (e.g., Koren et al., 2009), the entry value $r_{u,v} \in \mathbb{R}$ indicates that the vth item was rated $r_{u,v}$ by the uth user. Once P and Q are learned, a missing rating at the (u', v') entry can be predicted by the inner product of the u'th column of P (i.e., $\boldsymbol{p}_{u'}$) and the v'th column of Q (i.e., $\boldsymbol{q}_{v'}$). It means that we can generate the predicted scores on all items for a user, and then the one with highest score may be recommended. Sometimes, the users preferences recorded may be binary signals (e.g., "like" or "dislike"); for example, Das et al. (2007). The entry $r_{u,v}$ is positive if we know that the uth user likes the vth item, and vice versa. In this case, we can use the sign of $\boldsymbol{p}_{u'}^T \boldsymbol{q}_{v'}$ as the prediction value of an unobserved signal $r_{u',v'}$. As a result, the domain of the entries in R is allowed to be either real values $\mathbb{R}$ or a binary set $\{-1, 1\}$ to cover the two scenarios.

LIBMF supports two types of matrix factorization, real-valued matrix factorization (RVMF) and binary matrix factorization (BMF). RVMF aims to find P and Q to approximate the values of entries as accurate as possible while BMF focuses on learning the correct signs of entries. Both RVMF and BMF can be formulated as a non-convex optimization

1. `http://www.cs.utexas.edu/~rofuyu/libpmf`
2. `http://scikit-learn.org/stable/`
3. `http://www.mlpack.org`
4. `http://nimfa.biolab.si/`
5. `http://www.csie.ntu.edu.tw/~cjlin/papers/libmf/libmf_supp.pdf`

problem

$$\min_{P,Q} \sum_{(u,v)\in R} \left[f\left(\boldsymbol{p}_u, \boldsymbol{q}_v; r_{u,v}\right) + \mu_p \left\|\boldsymbol{p}_u\right\|_1 + \mu_q \left\|\boldsymbol{q}_v\right\|_1 + \frac{\lambda_p}{2}\left\|\boldsymbol{p}_u\right\|_2^2 + \frac{\lambda_q}{2}\left\|\boldsymbol{q}_v\right\|_2^2 \right], \quad (1)$$

where $r_{u,v}$ is the (u,v) entry of R, $f(\cdot)$ is a non-convex loss function of $\boldsymbol{p}_u$ and $\boldsymbol{q}_v$, and μ_p, μ_q, λ_p, and λ_q are regularization coefficients. For RVMF, the loss function can be a squared loss, an absolute loss, or generalized KL-divergence. If R is a binary matrix, users may select among logistic loss, hinge loss, and squared hinge loss to perform BMF. Note that, the non-negative constraints, $P \in \mathbb{R}_+^{k\times m}$ and $Q \in \mathbb{R}_+^{k\times n}$, can be activated for (1). With the general setting in (1), currently LIBMF can solve more problems than its previous versions. The newly covered problems include, for example, NMF with generalized KL-divergence (Lee and Seung, 2001), sparse NMF (Lee et al., 2007), logistic NMF (Tomé et al., 2015), and maximum-margin MF (Srebro et al., 2005).

A special case of BMF is one-class matrix factorization (OCMF) in which the training matrix is still binary but contains only positive entries. OCMF is worthy to be considered as in some online activities (e.g., Pan et al., 2008) only positive feedbacks (i.e., purchase or click) from users can be monitored. The OCMF model in LIBMF is Bayesian personalized ranking (BPR) proposed by Rendle et al. (2009).

Refer to the supplementary materials for details of RVMF, BMF, and OCMF.

2.2 Practical Usage

In a sub-directory `demo`, we prepare three data sets for helping users to understand the data format and demonstrating RVMF, BMF, and OCMF. Each data set has training and test files, of which each line is a 3-tuple of row index, column index, and the value. For RVMF, the value is a real number. For BMF, any value greater than zero would be treated as a positive label and a negative label otherwise. For OCMF, the training file should contain only positive entries. A shell script `demo.sh` contains the training and prediction commands for different types of MF problems. For example, we may use the following command to perform RVMF on `train_file` and save the result to a model file `model`.

```
$ ./mf-train train_file model
```

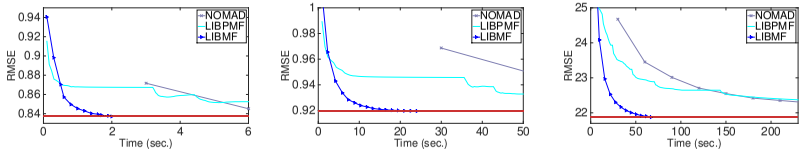
The model file contains two parts: the columns of $P \in \mathbb{R}^{k\times m}$ and then the columns of $Q \in \mathbb{R}^{k\times n}$. To evaluate the model on a test set, users may run

```
$ ./mf-predict test_file model out
```

The file `out` contains predictions of the entries specified in `test_file`, and the default evaluation measure, root mean square error (RMSE), is printed. Users can also choose absolute error, generalized KL-divergence, logarithmic loss, accuracy, mean percentile rank, or area under the curve.

2.3 Documentation

The script `demo.sh` mentioned in Section 2.2 is a quick start guide. The README file covers information including an installation guide, a C/C++ API guide, and examples of command line usages. Users can refer to Chin et al. (2015b) to see how we select parameters for some representative data sets. The two files `mf-train.cpp` and `mf-predict.cpp` for the



(a) `MovieLens` (b) `Netflix` (c) `Yahoo!Music`

Figure 1: Comparisons with state of the art packages. The figure is generated from the results in Chin et al. (2015b). The platform for the experiments is a Linux server with two Intel Xeon 2620 CPUs and 64GB memory. Twelve threads are used for all packages.

training and prediction commands, respectively, are good examples demonstrating the use of API functions. For the parallel stochastic gradient methods used in LIBMF, details are in Chin et al. (2015a), Chin et al. (2015b), and the supplementary material.

3. Comparisons

We compare LIBMF with state of the art packages, LIBPMF (Yu et al., 2012) and NOMAD (Yun et al., 2014), for parallel matrix factorization using a squared loss and L2-regularization. Results in Figure 1 indicate that LIBMF is efficient.

4. Conclusions

We develop LIBMF for fully using the computational power of modern multi-core machines to solve several MF problems. The newly added features listed below make LIBMF significantly improved in compared with its previous versions.

1. More loss functions are supported for RVMF.
2. The framework is extended to cover BMF, OCMF, and L1 regularization.
3. For different type of MF problems, we implement suitable evaluation criteria such as mean percentile rank and area under the curve.
4. LIBMF can use disk to cache the training matrix, so the capability to handle huge matrices is largely enhanced.

References

Wei-Sheng Chin, Yong Zhuang, Yu-Chin Juan, and Chih-Jen Lin. A fast parallel stochastic gradient method for matrix factorization in shared memory systems. *ACM Transactions on Intelligent Systems and Technology*, 6:2:1–2:24, 2015a. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/libmf/libmf_journal.pdf`.

Wei-Sheng Chin, Yong Zhuang, Yu-Chin Juan, and Chih-Jen Lin. A learning-rate schedule for stochastic gradient methods to matrix factorization. In *Proceedings of the Pacific-*

Asia Conference on Knowledge Discovery and Data Mining (PAKDD), 2015b. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/libmf/mf_adaptive_pakdd.pdf`.

Abhinandan Das, Mayur Datar, Shyam Rajaram, and Ashutosh Garg. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280, 2007.

Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, 2009.

Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001.

Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808. 2007.

Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *IEEE International Conference on Data Mining (ICDM)*, pages 502–511, 2008.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 452–461, 2009.

Nathan Srebro, Jason D. M. Rennie, and Tommi S. Jaakola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, pages 1329–1336, 2005.

Ana Maria Tomé, Reinhard Schachtner, Vincent Vigneron, Carlos Garcia Puntonet, and Elmar Wolfgang Lang. A logistic non-negative matrix factorization approach to binary data sets. *Multidimensional Systems and Signal Processing*, 26(1):125–143, 2015.

Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit S. Dhillon. Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In *Proceedings of the IEEE International Conference on Data Mining*, pages 765–774, 2012.

Hyokun Yun, Hsiang-Fu Yu, Cho-Jui Hsieh, S.V.N. Vishwanathan, and Inderjit S. Dhillon. Nomad: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion. In *International Conference on Very Large Data Bases (VLDB)*, 2014.

Journal of Machine Learning Research 17 (2016) 1-37 Submitted 1/16; Revised 5/16; Published 5/16

L_1-Regularized Least Squares for Support Recovery of High Dimensional Single Index Models with Gaussian Designs

Matey Neykov MNEYKOV@PRINCETON.EDU
Department of Operations Research and Financial Engineering
Princeton University
Princeton, NJ 08544, USA

Jun S. Liu JLIU@STAT.HARVARD.EDU
Department of Statistics
Harvard University
Cambridge, MA 02138, USA

Tianxi Cai TCAI@HSPH.HARVARD.EDU
Department of Biostatistics
Harvard University
Boston, MA 02115, USA

Editor: Xiaotong Shen

Abstract

It is known that for a certain class of single index models (SIMs) $Y = f(\boldsymbol{X}_{p\times 1}^{\intercal}\boldsymbol{\beta}_0, \varepsilon)$, support recovery is impossible when $\boldsymbol{X} \sim \mathcal{N}(0, \mathbb{I}_{p\times p})$ and a *model complexity adjusted sample size* is below a critical threshold. Recently, optimal algorithms based on Sliced Inverse Regression (SIR) were suggested. These algorithms work provably under the assumption that the design $\boldsymbol{X}$ comes from an i.i.d. Gaussian distribution. In the present paper we analyze algorithms based on covariance screening and least squares with L_1 penalization (i.e. LASSO) and demonstrate that they can also enjoy optimal (up to a scalar) rescaled sample size in terms of support recovery, albeit under slightly different assumptions on f and ε compared to the SIR based algorithms. Furthermore, we show more generally, that LASSO succeeds in recovering the signed support of $\boldsymbol{\beta}_0$ if $\boldsymbol{X} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, and the covariance $\boldsymbol{\Sigma}$ satisfies the irrepresentable condition. Our work extends existing results on the support recovery of LASSO for the linear model, to a more general class of SIMs.

Keywords: Single index models, Sparsity, Support recovery, High-dimensional statistics, LASSO

1. Introduction

Modern data applications often require scientists to deal with high-dimensional problems in which the sample size n could be much less than the dimensionality of the covariates p. To handle such challenging problems, structural assumptions on the data generating mechanism are often imposed. Such assumptions are motivated by the fact that classical procedures such as linear regression provably fail, unless the ratio p/n converges to 0. However, modern procedures based on regularization may work well under the high dimensional setting with additional sparsity assumptions. In the sparse high dimensional setting, it is often of interest to uncover the sparsity pattern, or in other words to select the relevant variables for the model. Under generalized linear models, support recovery can be achieved by fitting these models with penalized optimization procedures such as LASSO (Tibshirani, 1996) or Dantzig Selector (Candes and Tao, 2007), which are computationally inexpensive compared to exhaustive search approaches. The LASSO algorithm's variable selection/support recovery capabilities under generalized linear models, have been extensively studied (Meinshausen and Bühlmann, 2006; Zhao and Yu, 2006; Wainwright, 2009; Lee et al., 2013, e.g. among others). However, much less is known under potential mis-specification of these commonly used models or under more general models.

In this paper, we focus on recovering the support of the regression coefficients $\boldsymbol{\beta}_0$ under a *single index model* (SIM):

$$Y = f(\boldsymbol{X}^{\intercal}\boldsymbol{\beta}_0, \varepsilon), \quad \varepsilon \perp\!\!\!\perp \boldsymbol{X} \tag{1}$$

where both the link function f and the distribution of ε are left unspecified. Throughout, we assume that $\boldsymbol{\beta}_0^{\intercal}\boldsymbol{\Sigma}\boldsymbol{\beta}_0 = 1$ for identifiability and $E(\boldsymbol{X}) = 0$, where $\boldsymbol{\Sigma} = E(\boldsymbol{X}\boldsymbol{X}^{\intercal})$. We are specifically interested in the case where $\boldsymbol{X} \in \mathbb{R}^p \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ and $\boldsymbol{\beta}_0$ is s-sparse with $s < p$. Obviously, the SIM includes many commonly used parametric or semi-parametric regression models such as the linear regression model as special cases. Under (1) and the sparsity assumption, we aim to show that the standard least squares LASSO algorithm,

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \left\{ \frac{1}{2n}\sum_{i=1}^{n}(Y_i - \boldsymbol{X}_i^{\intercal}\boldsymbol{\beta})^2 + \lambda\|\boldsymbol{\beta}\|_1 \right\}, \tag{2}$$

can successfully recover the support of $\boldsymbol{\beta}_0$ provided standard regularity conditions and that the *model complexity adjusted effective sample size,*

$$n_{p,s} = n/\{s\log(p-s)\},$$

is sufficiently large. Obviously, for most choices of f, fitting (2) is essentially making inference under the mis-specified linear regression model.

The least squares LASSO algorithm has been used frequently in practice to perform variable selection for analyzing genomic data (Cantor et al., 2010; Zhao et al., 2011; Wang et al., 2013, e.g.). However, the linear model with Gaussian error is unlikely to be the true model in many such cases. Hence it is of practical importance to theoretically establish that the LASSO's support recovery capabilities are in fact robust to mis-specification. In addition to arguing that LASSO is robust, and perhaps even more surprisingly, we demonstrate that fitting the mis-specified linear model with LASSO penalty can optimally (up to a scalar) achieve support recovery with respect to the effective sample size $n_{p,s}$, for certain classes of $\boldsymbol{\Sigma}$ and SIMs in a minimax sense. In the special case when $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$, the LASSO algorithm can be slightly modified into a simple covariance screening procedure which possesses similar properties as the LASSO procedure.

1.1 Overview of Related Work

When the dimension p is small, inference under a SIM has been studied extensively in the literature (Xia and Li, 1999; Horowitz, 2009; Peng and Huang, 2011; McCullagh and Nelder,

1989, e.g.) among many others. In the highly relevant line of work on sufficient dimension reduction, many seminal insights can be found in Li and Duan (1989); Li (1991); Cook and Ni (2005). When $\boldsymbol{X} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, results given in Li and Duan (1989) can be used to show that $\operatorname{argmin}_{\boldsymbol{\beta}}\{\sum_{i=1}^n (Y_i - \boldsymbol{X}_i^\intercal \boldsymbol{\beta})^2\}$ consistently estimates $\boldsymbol{\beta}_0$ up to a scalar. When $\boldsymbol{\beta}_0$ is sparse, the sparse sliced inverse regression procedure given in Li and Nachtsheim (2006) can be used to effectively recover $\boldsymbol{\beta}_0$ under model (1) although their procedure requires a consistent estimator of $\boldsymbol{\Sigma}^{-1/2}$.

In the high dimensional setting with diverging p, Alquier and Biau (2013) were the first to consider the sparse SIM, and proposed an estimation framework using a PAC-Bayesian approach. Wang and Zhu (2015) and Wang et al. (2012) demonstrated that when support recovery can be achieved when $p = O(n^k)$ under a SIM via optimizations in the form of $\widehat{\boldsymbol{\beta}} = \operatorname{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2n} \sum_{i=1}^n (F_n(Y_i) - 1/2 - \boldsymbol{X}_i^\intercal \boldsymbol{\beta})^2 + \sum_{j=1}^p J_\lambda(\beta_j)$, where J_λ is a penalty function and $F_n(x) = \frac{1}{n}\sum_{i=1}^n \mathbb{1}(Y_i \leq x)$. Regularized procedures have also been proposed for specific choices of f and Y. For example, Yi et al. (2015) study consistent estimation under the model $\mathbb{P}(Y = 1|\boldsymbol{X}) = \{f(\boldsymbol{\beta}^\intercal \boldsymbol{X}) + 1\}/2$ with binary Y, where $f : \mathbb{R} \mapsto [-1, 1]$. Yang et al. (2015) consider the model $Y = f(\boldsymbol{X}^\intercal \boldsymbol{\beta}) + \varepsilon$ with known f, and develop estimation and inferential procedures based on the L_1 regularized least squares loss.

With p potentially growing with n exponentially and under a general SIM, Radchenko (2015) proposed a non-parametric least squares with an equality L_1 constraint to handle simultaneous estimation of $\boldsymbol{\beta}$ as well as f. The support recovery properties of this procedure are not investigated, and in addition the results do not exhibit the optimal scaling of the triple (n, p, s). Han and Wang (2015) suggest a penalized approach, in which they use a loss function related to Kendall's tau correlation coefficient. They also establish the L_2 consistency for the coefficient $\boldsymbol{\beta}$ but do not consider support recovery. Neykov et al. (2015) analyzed two algorithms based on Sliced Inverse Regression (Li, 1991) under the assumption that $\boldsymbol{X} \sim \mathcal{N}(0, \mathbb{I}_{p\times p})$, and demonstrated that they can uncover the support optimally in terms of the rescaled sample size. Plan and Vershynin (2015) and Thrampoulidis et al. (2015) demonstrated that a constrained version of LASSO can be used to obtain an L_2 consistent estimator of $\boldsymbol{\beta}_0$. None of these procedures provide results on the performance of the LASSO algorithm in support recovery, which relates to L_2 consistency but is a fundamentally different theoretical aspect. In addition, no existing work on the SIM estimation procedures demonstrates that the performance depends on (n, p, s) only through the effective sample size $n_{p,s}$.

1.2 Organization

The rest of the paper is organized as follows. Our main results are formulated in section 2. In particular, we show results on the covariance screening algorithm when $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$ in section 2.2 and our main result on the LASSO support recovery in section 2.3. Proof for the main results are given in section 3. In addition we demonstrate that for a class of SIMs, any algorithm provably fails to recover the support, unless the rescaled sample size $n_{p,s}$ is large enough. Numerical studies, confirming our main result are shown in section 4. We discuss potential future directions in section 5. Technical proofs are deferred to the appendixes.

2. Main Results

In this section we formulate our main results, which include the analysis of a simple covariance screening algorithm, and the LASSO algorithm for SIMs. Before we move on to the algorithms we summarize notation that we use throughout the paper and discuss several useful definitions and preliminary results.

2.1 Preliminary and Notation

For a (sparse) vector $\mathbf{v} = (v_1, \ldots, v_p)^\intercal$, we let $S(\mathbf{v}) := \operatorname{supp}(\mathbf{v}) = \{j : v_j \neq 0\}$ denote its support, $S_\pm(\mathbf{v}) := \{(\operatorname{sign}(v_j), j) : v_j \neq 0\}$ be its signed support, $\|\mathbf{v}\|_p$ denote the L_p norm, $\|\mathbf{v}\|_0 = |\operatorname{supp}(\mathbf{v})|$, and $\mathbf{v}^{\min} = \min_{i\in\operatorname{supp}(\mathbf{v})} |v_i|$. For a real random variable X, define

$$\|X\|_{\psi_2} = \sup_{p\geq 1} p^{-1/2}(\mathbb{E}|X|^p)^{1/p}, \quad \|X\|_{\psi_1} = \sup_{p\geq 1} p^{-1}(\mathbb{E}|X|^p)^{1/p}.$$ [1]

Recall that a random variable is called *sub-Gaussian* if $\|X\|_{\psi_2} < \infty$ and *sub-exponential* if $\|X\|_{\psi_1} < \infty$. For any integer $k \in \mathbb{N}$ we use the shorthand notation $[k] = \{1, \ldots, k\}$. For a matrix $\mathbf{M} \in \mathbb{R}^{d_1\times d_2}$, sets $S_1, S_2 \subseteq [d]$, we let $\mathbf{M}_{,S_2} = [M_{ij}]_{i\in[d_1]}^{j\in S_2}$ and $\mathbf{M}_{S_1,S_2} = [M_{ij}]_{i\in S_1}^{j\in S_2}$. For a vector $\boldsymbol{Z} = (Z_1, \ldots, Z_d)^\intercal$ and set $S \subseteq [d]$, $\boldsymbol{Z}_S$ denotes the subvector corresponding to the set S. Furthermore, let $\|\mathbf{M}\|_{p,q} = \sup_{\|\mathbf{v}\|_p=1} \|\mathbf{M}\mathbf{v}\|_q$. In particular, we have $\|\mathbf{M}\|_{2,2} = \max_{i\in[\max(d_1,d_2)]}\{s_i(\mathbf{M})\}$, where $s_i(\mathbf{M})$ is the i^{th} singular value of $\mathbf{M}$, and $\|\mathbf{M}\|_{\infty,\infty} = \max_{i\in[d_1]} \sum_{j=1}^{d_2} |M_{ij}|$. For a matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$, we put $D_{\max}(\mathbf{M}) = \max_{i\in[d]} |M_{ii}|$ for its maximal diagonal element, and $\operatorname{diag}(\mathbf{M}) = [M_{ii}]_{i\in[d]}$ for the collection of diagonal entries of $\mathbf{M}$. We also use standard asymptotic notations. Given two sequences $\{a_n\}, \{b_n\}$ we write $a_n = O(b_n)$ if there exists a constant $C < \infty$ such that $a_n \leq Cb_n$; $a_n = \Omega(b_n)$ if there exists a positive constant $c > 0$ such that $a_n \geq cb_n$, $a_n = o(b_n)$ if $a_n/b_n \to 0$, and $a_n \asymp b_n$ if there exists positive constants c and C such that $c < a_n/b_n < C$. Throughout, we also assume that there exists a constant $0 < \iota < 1$ such that $s < p - p^\iota$, which implies that $\frac{\log(p-s)}{\log(p)} \geq \iota$.

We assume that data for analysis consists of n independent and identically distributed (i.i.d.) random vectors $\mathcal{D} = \{(Y_i, \boldsymbol{X}_i^\intercal)^\intercal, i = 1, \ldots, n\}$ and we focus primarily on $\boldsymbol{X} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$. In matrix form, we let $\mathbb{X} = [\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n]_{n\times p}^\intercal = [X_{ij}]_{i\in[n]}^{j\in[p]}$, $\boldsymbol{Y} = (Y_1, \ldots, Y_n)^\intercal$, and $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_n)^\intercal$. The recovery of $\boldsymbol{\beta}_0$ under SIM often relies on the linearity of expectation assumption given in Li and Duan (1989) and Li (1991):

Definition 2.1.1 (Linearity of Expectation) *A p-dimensional random variable $\boldsymbol{X}$ is said to satisfy linearity of expectation in the direction $\boldsymbol{\beta}$ if for any direction $\boldsymbol{b} \in \mathbb{R}^p$:*

$$\mathbb{E}[\boldsymbol{X}^\intercal \boldsymbol{b} | \boldsymbol{X}^\intercal \boldsymbol{\beta}] = c_{\boldsymbol{b}} \boldsymbol{X}^\intercal \boldsymbol{\beta} + a_{\boldsymbol{b}},$$

where $a_{\boldsymbol{b}}, c_{\boldsymbol{b}} \in \mathbb{R}$ are some real constants which might depend on the direction $\boldsymbol{b}$.

Remark 2.1.2 *Note that if additionally $\mathbb{E}[\boldsymbol{X}] = 0$, then by taking expectation it is evident that $a_{\boldsymbol{b}} \equiv 0$. Clearly, linearity of expectation is direction specific by definition. Elliptical*

1. There are multiple equivalent (up to universal constants) definitions of the so-called Orlicz or ψ norms. See Vershynin (2010) Lemma 5.5 for a succinct formal treatment of this.

distributions (Fang et al., 1990) including multiviariate normal are known to satisfy the linearity in expectation uniformly in all directions (Cambanis et al., 1981, e.g.).

Next, we record a simple but very useful observation, which forms the basis of our work. From Theorem 2.1 of Li and Duan (1989), we note that under a SIM and normality of $\boldsymbol{X}$ with covariance $\boldsymbol{\Sigma} > 0$,

$$\operatorname*{argmin}_{\boldsymbol{b}} \mathbb{E}(Y - \boldsymbol{b}^\intercal \boldsymbol{X})^2 = c_0 \boldsymbol{\beta}_0,$$

for some $c_0 \in \mathbb{R}$. More generally, we have

Lemma 2.1.3 *Assume that the SIM (1) holds, $\boldsymbol{\Sigma} = \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^\intercal) > 0$, and $\boldsymbol{X}$ satisfies the linearity in expectation condition in the direction $\boldsymbol{\beta}_0$ such that $\mathbb{E}[(\boldsymbol{X}^\intercal \boldsymbol{\beta}_0)^2] > 0$. Then we have $\boldsymbol{\Sigma}^{-1}\mathbb{E}(Y\boldsymbol{X}) = c_0\boldsymbol{\beta}_0$, where $c_0 := \mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0)$. Obviously, $\mathbb{E}(Y\boldsymbol{X}) = c_0\boldsymbol{\beta}_0$ when $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$.*

In view of Lemma 2.1.3, under sparsity assumptions, an L_1 regularized least square estimator can recover $\boldsymbol{\beta}_0$ proportionally and hence the support of $\boldsymbol{\beta}_0$. Furthermore, when $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$, the covariance $\mathbb{E}(Y\boldsymbol{X})$ can be directly used to recover $\boldsymbol{\beta}_0$. It is noteworthy to remark that in the special case $\boldsymbol{X} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$, a simple application of Stein's Lemma (Stein, 1981) can help quantify the constant c_0 precisely:

$$c_0 = \mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) = \mathbb{E}(Z\underbrace{[\mathbb{E}f(Z,\varepsilon)|Z]}_{\varphi(Z)}) = \int D\varphi(z)\frac{\exp(-z^2/2)}{\sqrt{2\pi}}dz = \mathbb{E}D\varphi(Z),$$

where $D\varphi$ is the distributional derivative of φ, $Z \sim \mathcal{N}(0,1)$ and we abused the notation slightly in the last equality for simplicity.

The remaining of this section is structured as follows — in section 2.2 we study a simple covariance thresholding algorithm, which is a manifestation of program (2) under the assumption $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$. In section 2.3 we consider the full-fledged least squares LASSO algorithm (2) with a general covariance matrix $\boldsymbol{\Sigma}$. Throughout, we assume $E(\boldsymbol{X}) = 0$ and let $\sigma^2 = E(Y^2)$, $\eta = \mathrm{Var}(Y^2)$,

$c_0 = \mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0)$, $\gamma = \mathrm{Var}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0)$, $\xi^2 = \mathbb{E}\{(Y - c_0\boldsymbol{X}^\intercal\boldsymbol{\beta}_0)^2\}$, and $\theta^2 = \mathrm{Var}\{(Y - c_0\boldsymbol{X}^\intercal\boldsymbol{\beta}_0)^2\}$.

In addition, to simplify the presentation we will assume that the above constants are not scaling with (n, p, s), and belong to a compact set which is bounded away from 0.

2.2 Covariance Screening under $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$

In this section, we propose a simple covariance screening procedure for signed support recovery of $\boldsymbol{\beta}_0$ under a SIM with $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$, which relates to the sure independence screening procedures (Fan and Lv, 2008; Fan et al., 2010) under the linear model. Note that the LASSO procedure (2) can equivalently be expressed as:

$$\widehat{\boldsymbol{\beta}} = \operatorname*{argmin}_{\boldsymbol{\beta}\in\mathbb{R}^p}\left\{-\frac{1}{n}\sum_{i=1}^n Y_i\boldsymbol{X}_i^\intercal\boldsymbol{\beta} + \frac{1}{2n}\boldsymbol{\beta}^\intercal\sum_{i=1}^n \boldsymbol{X}_i\boldsymbol{X}_i^\intercal\boldsymbol{\beta} + \lambda\|\boldsymbol{\beta}\|_1\right\}.$$

Under the assumption $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$, we replace $\frac{1}{n}\sum_{i=1}^n \boldsymbol{X}_i\boldsymbol{X}_i^\intercal$ with $\mathbb{I}_{p\times p}$ and instead consider

$$\widehat{\boldsymbol{\beta}} = \operatorname*{argmin}_{\boldsymbol{\beta}\in\mathbb{R}^p}\left\{-\frac{1}{n}\sum_{i=1}^n Y_i\boldsymbol{X}_i^\intercal\boldsymbol{\beta} + \frac{1}{2}\boldsymbol{\beta}^\intercal\boldsymbol{\beta} + \lambda\|\boldsymbol{\beta}\|_1\right\}.$$

It is well known that the solution to the above program takes the form:

$$\widehat{\beta}_j = \mathrm{sign}\left(n^{-1}\sum_{i=1}^n Y_iX_{ij}\right)\left(\left|n^{-1}\sum_{i=1}^n Y_iX_{ij}\right| - \lambda\right)_+,$$

where $x_+ = x\mathbb{1}(x \geq 0)$. Hence, in this special case, the regularization parameter can be equivalently interpreted as a thresholding parameter, filtering all small $|n^{-1}\sum_{i=1}^n Y_iX_{ij}|$. Motivated by this, we consider the following simple covariance screening procedure, which acts as a filter taking covariances and their corresponding signs, only if they pass a critical threshold:

Algorithm 1: Covariance Screening Algorithm

input: $(Y_i, \boldsymbol{X}_i)_{i=1}^n$: data; tuning parameter $\nu > 0$

1. Calcluate $\boldsymbol{V} := \widehat{\mathrm{Cov}}(Y, \boldsymbol{X}) := n^{-1}\sum_{i=1}^n Y_i\boldsymbol{X}_i$,
2. Set $\widehat{S} := \left\{V_j : |V_j| > \nu\sqrt{\frac{\log p}{n}}\right\}$;
3. Output the set $\{(\mathrm{sign}(V_j), j) : |V_j| \in \widehat{S}\}$.

The following proposition shows that the algorithm recovers the support with probability approaching 1 provided that the effective sample size $n_{p,s}$ is sufficiently large under the normality assumption of $\boldsymbol{X} \sim \mathcal{N}(0, \mathbb{I}_{p\times p})$.

Proposition 2.2.1 *Assume that $\boldsymbol{X} \sim \mathcal{N}(0, \mathbb{I}_{p\times p})$, $c_0 \neq 0$, $E(Y^4) < \infty$, and $\beta_{0j} \in \{\pm\frac{1}{\sqrt{s}}, 0\}$ for all $j \in [p]$ and some $s \in \mathbb{N}$. Set $\nu = \frac{1}{\sqrt{s}} + 2\sqrt{2}\sigma$. Then as long as*

$$n_{p,s} \geq \Upsilon,$$

for a large enough $\Upsilon = O(1)$ (depending on c_0, σ^2) Algorithm 1 recovers the signed support of $c_0\boldsymbol{\beta}_0$ with asymptotic probability 1.

The proof of Proposition 2.2.1 can be found in Appendix D. We note that this result does not require Y to be sub-Gaussian. When sub-Gaussianity is assumed, the convergence rate of the probability approaching 1 can be improved. Furthermore, under sub-Gaussianity of Y, we can relax the normality assumption of $\boldsymbol{X}$. Specifically, one may instead consider the following assumptions

Assumption 2.2.2 (Spherical Distribution of $\boldsymbol{X}$ and sub-Gaussian of Y) *Let $\boldsymbol{X}$ be a spherically distributed p dimensional random variable with $\mathbb{E}[\boldsymbol{X}] = 0, \mathrm{Var}[\boldsymbol{X}] = \mathbb{I}_{p\times p}$, whose moment generating function exists, and takes the form $\mathbb{E}\{\exp(\boldsymbol{t}^\intercal\boldsymbol{X})\} \equiv \Psi(\boldsymbol{t}^\intercal\boldsymbol{t}), \boldsymbol{t} \in \mathbb{R}^p$, and in addition $\Psi : \mathbb{R} \mapsto \mathbb{R}$ is such that $\Psi(t) \leq \exp(Ct)$ for some $C > 0$ for all $t \in \mathbb{R}^+$. In addition, we assume that Y is sub-Gaussian, i.e. $\|Y\|_{\psi_2} \leq K_Y$.*

Note that (Vershynin, 2010, see Lemma 5.5. e.g.) assumption 2.2.2 implies $\max_{j\in[p]}\|X_j\|_{\psi_2} < \infty$. Let $K := \max_{j\in[p]}\|X_j\|_{\psi_2}$. In parallel to Proposition 2.2.1, we have

Proposition 2.2.3 *In addition to Assumption 2.2.2, we assume that s, $c_0 \neq 0$ and $\beta_{0j} \in \{\pm\frac{1}{\sqrt{s}}, 0\}$ for all $j \in [p]$ and some $s \in \mathbb{N}$. Set the tuning parameter $\nu = \omega K_Y K$ for some absolute constant $\omega > 0$. Then there exits an absolute constant $\Upsilon \in \mathbb{R}$ depending on c_0, C, K and K_Y such that if:*

$$n_{p,s} \geq \Upsilon,$$

Algorithm 1 recovers the signed support of $c_0\boldsymbol{\beta}_0$ with asymptotic probability 1.

2.3 LASSO Algorithm with General $\boldsymbol{\Sigma}$

In this section we consider the LASSO algorithm (2),

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2n}\|\boldsymbol{Y} - \mathbb{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{3}$$

under the assumption $\boldsymbol{X} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ with a generic and unknown covariance matrix $\boldsymbol{\Sigma}$. Under certain sufficient conditions our goal is to show that (2) recovers the support of $\boldsymbol{\beta}_0$ with asymptotic probability 1 in optimal (up to a scalar) effective sample size. In contrast to section 2.2, we also no longer require each of the signals of $\boldsymbol{\beta}$ to be of the same magnitude.

We first summarize a primal dual witness (PDW) construction which we borrow from Wainwright (2009). The PDW construction lays out steps allowing one to prove sign consistency for L_1 constrained quadratic programming (3). We will only provide the sufficient conditions to show sign-consistency, and the interested reader can check Wainwright (2009) for the necessary conditions. We note that the validity of the PDW construction is generic, in that it does not rely on the distribution of the residual $\mathbf{w} = \boldsymbol{Y} - c_0\mathbb{X}\boldsymbol{\beta}_0$, and hence extends to the current framework. Note that unlike the linear regression case, in our setting $\mathbf{w}$ does not necessarily have mean 0 although $\mathbb{E}[\mathbb{X}^\intercal\mathbf{w}] = 0$.

Recall that a vector $\mathbf{z}$ is a subgradient of the L_1 norm evaluated at a vector $\mathbf{v} \in \mathbb{R}^p$ (i.e. $\mathbf{z} \in \partial\|\mathbf{v}\|_1$) if we have $z_j = \operatorname{sign}(v_j), v_j \neq 0$ and $z^j \in [-1, 1]$ otherwise. It follows from Karush-Kuhn-Tucker's theorem that a vector $\widehat{\boldsymbol{\beta}} \in \mathbb{R}^p$ is optimal for the LASSO problem (3) iff there exists a subgradient $\widehat{\mathbf{z}} \in \partial\|\widehat{\boldsymbol{\beta}}\|_1$ such that:

$$\frac{1}{n}\mathbb{X}^\intercal\mathbb{X}(\widehat{\boldsymbol{\beta}} - c_0\boldsymbol{\beta}_0) - \frac{1}{n}\mathbb{X}^\intercal\mathbf{w} + \lambda\widehat{\mathbf{z}} = 0. \tag{4}$$

Put $S_0 := S(\boldsymbol{\beta}_0)$ for brevity. In what follows we will assume that the matrix $\mathbb{X}_{\cdot,S_0}^\intercal\mathbb{X}_{\cdot,S_0}$ is invertible (which it is with probability 1), even though this is not required by the PDW. The PDW method constructs a pair $(\check{\boldsymbol{\beta}}, \check{\mathbf{z}}) \in \mathbb{R}^p \times \mathbb{R}^p$ by following the steps:

- Solve:
$$\check{\boldsymbol{\beta}}_{S_0} = \underset{\boldsymbol{\beta}_{S_0}\in\mathbb{R}^s}{\operatorname{argmin}} \frac{1}{2n}\|\boldsymbol{Y} - \mathbb{X}_{\cdot,S_0}\boldsymbol{\beta}_{S_0}\|_2^2 + \lambda\|\boldsymbol{\beta}_{S_0}\|_1,$$
where $s = |S_0|$. This solution is unique under the invertibility of $\mathbb{X}_{\cdot,S_0}^\intercal\mathbb{X}_{\cdot,S_0}$ since in this case the function is strictly convex. Set $\check{\boldsymbol{\beta}}_{S_0^c} = 0$.

- Choose a $\check{\mathbf{z}}_{S_0} \in \partial\|\check{\boldsymbol{\beta}}_{S_0}\|_1$.
- For $j \in S_0^c$ set $Z_j := \mathbb{X}_{\cdot,j}^\intercal[\mathbb{X}_{\cdot,S_0}(\mathbb{X}_{\cdot,S_0}^\intercal\mathbb{X}_{\cdot,S_0})^{-1}\check{\mathbf{z}}_{S_0} + \mathbf{P}_{\mathbb{X}_{\cdot,S_0}^\perp}\left(\frac{\mathbf{w}}{\lambda n}\right)]$[2], where $\mathbf{P}_{\mathbb{X}_{\cdot,S_0}^\perp} = \mathbb{I} - \mathbb{X}_{\cdot,S_0}(\mathbb{X}_{\cdot,S_0}^\intercal\mathbb{X}_{\cdot,S_0})^{-1}\mathbb{X}_{\cdot,S_0}^\intercal$ is an orthogonal projection. Checking that $|Z_j| < 1$ for all $j \in S_0^c$ ensures that there is a unique solution $\check{\boldsymbol{\beta}} = (\check{\boldsymbol{\beta}}_{S_0}^\intercal, \check{\boldsymbol{\beta}}_{S_0^c}^\intercal)^\intercal$ satisfying $S(\check{\boldsymbol{\beta}}) \subseteq S(c_0\boldsymbol{\beta}_0)$.
- To check sign consistency we need $\check{\mathbf{z}}_{S_0} = \operatorname{sign}(c_0\boldsymbol{\beta}_{0S_0})$. For each $j \in S_0$, define:
$$\Delta_j := \mathbf{e}_j^\intercal(n^{-1}\mathbb{X}_{\cdot,S_0}^\intercal\mathbb{X}_{\cdot,S_0})^{-1}\left[n^{-1}\mathbb{X}_{\cdot,S_0}^\intercal\mathbf{w} - \lambda\operatorname{sign}(c_0\boldsymbol{\beta}_{0S_0})\right],$$ [3]
where $\mathbf{e}_j \in \mathbb{R}^s$ is a canonical unit vector with 1 at the j^{th} position. Checking $\check{\mathbf{z}}_{S_0} = \operatorname{sign}(c_0\boldsymbol{\beta}_{0S_0})$ is equivalent to checking:
$$\operatorname{sign}(c_0\beta_{0j} + \Delta_j) = \operatorname{sign}(c_0\beta_{0j}),\ \forall j \in S_0.$$

To this end we require several restrictions on $\boldsymbol{\Sigma}$ and moment conditions on Y. We partition the covariance matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{S_0,S_0} & \boldsymbol{\Sigma}_{S_0,S_0^c} \\ \boldsymbol{\Sigma}_{S_0^c,S_0} & \boldsymbol{\Sigma}_{S_0^c,S_0^c} \end{bmatrix},$$

where $\boldsymbol{\Sigma}_{S_0,S_0}$ corresponds to the covariance of $\boldsymbol{X}_{S_0}$. Furthermore, we let

$$\boldsymbol{\Sigma}_{S_0^c|S_0} := \boldsymbol{\Sigma}_{S_0^c,S_0^c} - \boldsymbol{\Sigma}_{S_0^c,S_0}\boldsymbol{\Sigma}_{S_0,S_0}^{-1}\boldsymbol{\Sigma}_{S_0,S_0^c}, \tag{5}$$

$$\rho_\infty(\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}) := \|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}\|\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}\|_{\infty,\infty}, \tag{6}$$

be the conditional covariance matrix of $\boldsymbol{X}_{S_0^c}|\boldsymbol{X}_{S_0}$, and the condition number of $\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}$ with respect to $\|\cdot\|_{\infty,\infty}$, respectively. We assume that

Assumption 2.3.1 (Irrepresentable Condition)

$$\|\boldsymbol{\Sigma}_{S_0^c,S_0}\boldsymbol{\Sigma}_{S_0,S_0}^{-1}\|_{\infty,\infty} \leq (1-\kappa), \quad \textit{for some } \kappa > 0.$$

Assumption 2.3.2 (Bounded Spectrum) *For some fixed $0 < \lambda_{\min} \leq \lambda_{\max} < \infty$,*

$$\lambda_{\min} \leq \boldsymbol{\Sigma}_{S_0,S_0} \leq \lambda_{\max}.$$

Assumption 2.3.3 (Bounded 4$^{\text{th}}$ Moment) $\mathbb{E}(Y^4) < \infty$, *and does not scale with* (n, p, s).

Recall that the irrepresentable condition is proved to be necessary for successful support recovery (Wainwright, 2009, Theorem 4) in the linear model, and hence one should not expect that Assumption 2.3.1 can be weakened. Assumption 2.3.3 guarantees that σ^2, η, c_0, γ, ξ^2 and θ^2 are well defined and finite. Finally, successful support recovery will depend on the strength of the minimal signal of $\boldsymbol{\beta}_0$. Recall that $\boldsymbol{\beta}_0^{\min} := \min_{j\in S_0}|\beta_{0j}|$, is the minimal non-zero signal in the vector $\boldsymbol{\beta}_0$. We are now ready to provide sufficient conditions for the LASSO signed support recovery, in the setting of SIMs

2. Z_j are derived by simply plugging in $\check{\boldsymbol{\beta}}$ and $\check{\mathbf{z}}_{S_0}$ and solving (4) for $\check{\mathbf{z}}_{S_0^c}$.
3. Δ_j can be seen to equal $\check{\beta}_j - c_0\beta_{0j}$ for $j \in S_0$, when $\check{\mathbf{z}}_{S_0} = \operatorname{sign}(c_0\boldsymbol{\beta}_0)$.

Theorem 2.3.4 *Let Assumptions 2.3.1 — 2.3.3 hold. Then for the LASSO estimator given in (3) under a SIM (1), we have the following sufficient conditions:*

i. If

$$n_{p,s} \geq \frac{4D_{\max}(\mathbf{\Sigma}_{S_0^c|S_0})\left(\frac{4}{\lambda_{\min}} + \frac{\xi^2+1}{\lambda^2 s}\right)}{\kappa^2},$$

then $S(\widehat{\boldsymbol{\beta}}) \subset S(c_0\boldsymbol{\beta}_0)$, with probability at least $1 - O\{p^{-1} + n^{-1} + e^{-\Omega(s)}\}$.

ii. Let further, for some positive constant $\alpha > 0$ we have $n_{p,s} \geq \alpha$. Then there exist some positive constants $\Upsilon_0, \Upsilon_1, \Upsilon_2 > 0$ which may depend[1] on c_0, σ such that if:

$$\boldsymbol{\beta}_0^{\min} \geq \|\mathbf{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}^2 \lambda \Upsilon_0 + \left[\Upsilon_1 \rho_\infty(\mathbf{\Sigma}_{S_0,S_0}^{1/2})\sqrt{\frac{s}{n\log(p-s)}}\|\boldsymbol{\beta}_0\|_\infty + \frac{\Upsilon_2 \|\mathbf{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}}{\sqrt{s}}\right] n_{p,s}^{-1/2} \tag{7}$$

we have $S_\pm(\widehat{\boldsymbol{\beta}}) = S_\pm(c_0\boldsymbol{\beta}_0)$ with probability at least $1 - O\{e^{-\Omega(s \wedge \log(p-s))} + (\log p)^{-1} + n^{-1}\}$.

Before we proceed with the proof of our main result, we would like to mention a few remarks on our sufficient conditions, in particular the ones suggested in ii.

Remark 2.3.5 *Firstly, the slow probability convergence rate $(\log p)^{-1}$ in part ii. is purely due to the fact we are not willing to assume that Y is coming from a sub-Gaussian distribution. If such an assumption is made the rate reduces to the usual p^{-1}. Secondly, observe that $\lambda_{\max}^{-1} \leq \|\boldsymbol{\beta}_0\|_2 \leq \lambda_{\min}^{-1}$. Hence the value of $\boldsymbol{\beta}_0^{\min}$ is of "largest" order when $\boldsymbol{\beta}_0^{\min} \asymp \|\boldsymbol{\beta}_0\|_\infty \asymp \frac{1}{\sqrt{s}}$. Setting*

$$\lambda := \lambda_T = \sqrt{(\xi^2+1)\frac{4C_T D_{\max}(\mathbf{\Sigma}_{S_0^c|S_0})}{\kappa^2}\frac{\log(p-s)}{n}},$$

for some $C_T > 1$ gives us that the condition from i. is equivalent to:

$$n_{p,s} \geq \frac{16 D_{\max}(\mathbf{\Sigma}_{S_0^c|S_0})}{(1 - C_T^{-1})\kappa^2 \lambda_{\min}}.$$

Note that due to positive definiteness: $D_{\max}(\mathbf{\Sigma}_{S_0^c|S_0}) \leq D_{\max}(\mathbf{\Sigma}_{S_0^c,S_0^c})$, and hence it is reasonable to assume $D_{\max}(\mathbf{\Sigma}_{S_0^c|S_0}) = O(1)$. Assume additionally that $\|\mathbf{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty} = O(1)$, $\rho_\infty(\mathbf{\Sigma}_{S_0,S_0}^{1/2}) = O(1)$, and let $\boldsymbol{\beta}_0^{\min} \asymp \frac{1}{\sqrt{s}}$. Notice that the space of matrices satisfying assumptions 2.3.1 — 2.3.3 and the latter assumptions is non-empty as one can easily show that Toeplitz matrices with entries $\Sigma_{kj} = \rho^{|k-j|}$ with $|\rho| < 1$ satisfy these conditions e.g. Using the same $\lambda = \lambda_T$ we can clearly achieve the sufficient condition in ii. provided that the model complexity adjusted sample size $n_{p,s}$ is large enough. On the other hand, this scaling can no longer be guaranteed if $\boldsymbol{\beta}_0^{\min} \asymp \frac{1}{\sqrt{s}}$ fails to hold. In any case, it is clear that when $\mathbf{\Sigma} = \mathbb{I}_{p\times p}$ all conditions required are met, and hence Theorem 2.3.4 shows that the LASSO algorithm will work optimally (up to a scalar) in terms of the rescaled sample size. Below we formulate a result which allows us to claim certain optimality for the LASSO algorithm in the case of a more general $\mathbf{\Sigma}$ matrix whenever we have approximately equally sized coefficients.

4. The dependency is inversely proportional to $|c_0|$ and proportional to σ; For more details refer to the proof.

Proposition 2.3.6 *Consider a special example of a SIM with $Y = G\{h(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) + \varepsilon\}$ for $\boldsymbol{X} \sim \mathcal{N}(0, \mathbf{\Sigma})$, and G, h are known strictly increasing continuous functions and in addition h is an L-Lipschitz function. Assume that there exists a set $S \subset [p], |S| = s$ such that $\|\mathbf{\Sigma}_{S,S}\|_{\infty,\infty} < R$, Assumptions 2.3.1 and 2.3.2 hold on S and $0 < d \leq \operatorname{diag}(\mathbf{\Sigma}_{S^c,S^c}) \leq D < \infty$. In addition, assume that ε is a continuously distributed random variable with density p_ε satisfying:*

$$p_\varepsilon(x) \propto \exp(-P(x^2)), \tag{8}$$

where P is any non-zero polynomial with non-negative coefficients such that $P(0) = 0$. We restrict the parameter space to $\left\{\boldsymbol{\beta} \in \mathbb{R}^p : \boldsymbol{\beta}^\intercal\mathbf{\Sigma}\boldsymbol{\beta} = 1, \|\boldsymbol{\beta}\|_0 = s, \frac{\boldsymbol{\beta}^{\min}}{\|\boldsymbol{\beta}\|_\infty} \geq c_{\mathbf{\Sigma}}\right\}$, where $c_{\mathbf{\Sigma}} > 0$ is a sufficiently small constant depending solely on $\mathbf{\Sigma}$ (see (20) for details). If

$$n_{p,s} < C,$$

for some constant $C > 0$ (depending on $P, G, h, \mathbf{\Sigma}$) and s is sufficiently large, any algorithm for support recovery makes errors with probability at least $\frac{1}{2}$ asymptotically.

Proposition 2.3.6 justifies that the LASSO is sample size optimal even when more generic covariance matrices than identity are considered, provided that we can show the class of SIMs defined satisfy the assumptions of this section. We do so in the following Remark.

Remark 2.3.7 *We will now argue that if we have a model as described in Proposition 2.3.6, the LASSO will recover the support provided that the the covariance matrix $\mathbf{\Sigma}$ satisfies Assumptions 2.3.1 and 2.3.2, $\mathbb{E}(Y^4) < \infty$ and the minimal signal strength is sufficiently strong as required by Remark 2.3.5. Notice that by Chebyshev's association inequality (Boucheron et al., 2013), we have:*

$$\mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) > \mathbb{E}(Y)\mathbb{E}(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) = 0,$$

where the inequality is strict since G and h are strictly increasing and $\boldsymbol{X}^\intercal\boldsymbol{\beta}_0 \sim \mathcal{N}(0,1)$. In fact, using exactly the same argument, one can show more generally that if $r(z) = \mathbb{E}[Y|\boldsymbol{X}^\intercal\boldsymbol{\beta}_0 = z]$ is a strictly monotone function it follows that $\mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) \neq 0$. We close this remark by pointing out that the logistic regression model $P(Y = 1 \mid \boldsymbol{X}) = g(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0)$ with $g(x) = e^x/(1+e^x)$ satisfies the condition since $r(z) = g(z)$ is strictly monotone, and hence using the LASSO algorithm one can recover the support correctly. This is an example that even discrete valued Y outcomes can be solved by the least squares LASSO algorithm.

2.3.1 Outcome Transformations

In this subsection we provide brief comments on possible strategies to transform the data in view of the results of Theorem 2.3.4. A crucial condition in order for the signed support recovery of $\boldsymbol{\beta}_0$ to hold is $c_0 \neq 0$, which should not be expected to hold in general, but nevertheless naturally occurs in many cases. If this condition does not hold, one can potentially transform the outcome $\widetilde{Y} = g(Y)$ by a function g in order to achieve $\mathbb{E}(\widetilde{Y}\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) \neq 0$ even if $\mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) = 0$. If we use $\widetilde{Y}_i = g(Y_i)$ instead of Y_i then clearly LASSO succeeds under the assumptions from Theorem 2.3.4, only with assumptions on Y_i being replaced by $\widetilde{Y}_i$. The following proposition characterizes when one should expect a correlation inducing transformation g to exist.

Proposition 2.3.8 *There exists a measurable function $g : \mathbb{R} \mapsto \mathbb{R}$ such that $\mathbb{E}\{g(Y)\boldsymbol{X}^\intercal\boldsymbol{\beta}_0\} \neq 0$ if and only if $\mathrm{Var}\{\mathbb{E}(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0|Y)\} > 0$.*

Another potential advantage of performing a transformation is to ensure that $\widetilde{Y} = g(Y)$ is sub-Gaussian. For example, if we let $g(y) = F(y) = P(Y \leq y)$, then the sub-Gaussianity of $\widetilde{Y}$ is guaranteed, which would improve the rate of support recovery. For many choices of g such as F, the transformation may be defined at the population level and is unknown a priori. Thus it would be desirable to employ data dependent estimate $\widehat{g}$ of g. In other words we consider fitting the following LASSO to recover the support:

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2n}\|\widehat{g}(\boldsymbol{Y}) - \mathbb{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{9}$$

where $\widehat{g}(\boldsymbol{Y})$ should be understood as element-wise application of $\widehat{g}$. The following Corollary extends Theorem 2.3.4 to allow for data dependent transformations.

Corollary 2.3.9 *Let the assumptions of Theorem 2.3.4 hold for $\widetilde{Y}_i = g(Y_i)$ in place of Y_i and assume additionally that:*

$$\|\widehat{g}(\boldsymbol{Y}) - g(\boldsymbol{Y})\|_2 \leq O(\sqrt{\log p}), \tag{10}$$

with probability at least $1 - O(p^{-1})$, then if (7) holds we have $S_\pm(\widehat{\boldsymbol{\beta}}) = S_\pm(c_0\boldsymbol{\beta}_0)$ with probability at least $1 - O\{e^{-\Omega(s\wedge\log(p-s))} + (\log p)^{-1} + n^{-1}\}$.

Remark 2.3.10 *Akin to (2), we do not require an intercept in the model after doing a transformation in (9). This is possible since $\boldsymbol{X}$ is assumed to have mean 0. In practice if this were not the case, one would have to either center $\boldsymbol{X}$ or include an intercept which is not penalized.*

3. Proof of Theorem 2.3.4

Our proof follows similar steps as Theorem 3 of Wainwright (2009) although many critical modifications are needed, since the error term $\mathbf{w} = \mathbf{Y} - c_0\mathbb{X}\boldsymbol{\beta}_0$ is no longer independent of $\mathbb{X}$ and is not mean 0.

3.1 Verifying Strict Dual Feasibility

For $j \in S_0^c$ decompose

$$\mathbb{X}_{\cdot j}^\intercal = \boldsymbol{\Sigma}_{\{j\},S_0}\boldsymbol{\Sigma}_{S_0,S_0}^{-1}\mathbb{X}_{\cdot S_0}^\intercal + \boldsymbol{E}_j^\intercal, \tag{11}$$

where the entries of the prediction error vector $\boldsymbol{E}_j = (E_{1j}, \ldots, E_{nj})^\intercal \in \mathbb{R}^n$ are i.i.d. with $E_{ij} \sim \mathcal{N}(0, [\boldsymbol{\Sigma}_{S_0^c|S_0}]_{jj}), i \in [n]$. In addition, observe that by this construction we have that $\boldsymbol{E}_j$ is independent of $\mathbb{X}_{\cdot S_0}$ which can be verified upon multiplication by $\mathbb{X}_{\cdot S_0}$ in (11) and taking expectation. Following the definition of Z_j gives us that $Z_j = A_j + B_j$, where:

$$A_j := \boldsymbol{E}_j^\intercal\left[\mathbb{X}_{\cdot S_0}(\mathbb{X}_{\cdot S_0}^\intercal\mathbb{X}_{\cdot S_0})^{-1}\check{\mathbf{z}}_{S_0} + \mathbf{P}_{\mathbb{X}_{\cdot S_0}^\perp}\left(\frac{\mathbf{w}}{\lambda n}\right)\right], \tag{12}$$

$$B_j := \boldsymbol{\Sigma}_{\{j\},S_0}(\boldsymbol{\Sigma}_{S_0,S_0})^{-1}\check{\mathbf{z}}_{S_0}. \tag{13}$$

Under the irrepresentable condition, we have that $\max_{j\in S^c}|B_j| \leq (1-\kappa)$. Conditional on $\mathbb{X}_{\cdot S_0}$ and $\boldsymbol{\varepsilon}$ (which determine $\mathbf{w} = \boldsymbol{Y} - c_0\mathbb{X}\boldsymbol{\beta}_0$) we have that the gradient $\check{\mathbf{z}}_{S_0}$ is independent of the vector $\boldsymbol{E}_j$ because the gradient is deterministic after conditioning on these quantities. We have that $\mathrm{Var}(E_{ij}) \leq D_{\max}(\boldsymbol{\Sigma}_{S_0^c|S_0})$, and thus conditionally on $\mathbb{X}_{\cdot S_0}$ and $\boldsymbol{\varepsilon}$ we get:

$$\begin{aligned}\mathrm{Var}(A_j) &\leq D_{\max}(\boldsymbol{\Sigma}_{S_0^c|S_0})\left\|\mathbb{X}_{\cdot S_0}(\mathbb{X}_{\cdot S_0}^\intercal\mathbb{X}_{\cdot S_0})^{-1}\check{\mathbf{z}}_{S_0} + \mathbf{P}_{\mathbb{X}_{\cdot S_0}^\perp}\left(\frac{\mathbf{w}}{\lambda n}\right)\right\|_2^2 \\ &= D_{\max}(\boldsymbol{\Sigma}_{S_0^c|S_0})\left[\check{\mathbf{z}}_{S_0}^\intercal(\mathbb{X}_{\cdot S_0}^\intercal\mathbb{X}_{\cdot S_0})^{-1}\check{\mathbf{z}}_{S_0} + \left\|\mathbf{P}_{\mathbb{X}_{\cdot S_0}^\perp}\left(\frac{\mathbf{w}}{\lambda n}\right)\right\|_2^2\right].\end{aligned}$$

Next we need a lemma, which is a slight modification of Lemma 4 in Wainwright (2009). The reason for this modification is that in our case $\mathbf{w}$ is no longer $\sim\mathcal{N}(0,\sigma^2\mathbb{I})$.

Lemma 3.1.1 *Assume that $\frac{s}{n} \leq \frac{1}{16}$. Then we have:*

$$\max_{j\in S_0^c}\mathrm{Var}(A_j) \leq \underbrace{D_{\max}(\boldsymbol{\Sigma}_{S_0^c|S_0})\left(\frac{4s}{\lambda_{\min}n} + \frac{\xi^2+1}{\lambda^2 n}\right)}_{M},$$

with probability at least $1 - 2e^{-s/2} - n^{-1}\theta^2$.

Now since conditionally on $\mathbb{X}_{\cdot S_0}$ and $\boldsymbol{\varepsilon}$ we have $A_j \sim \mathcal{N}(0, \mathrm{Var}(A_j))$, using a standard normal tail bound and the union bound we conclude:

$$\mathbb{P}(\max_{j\in S_0^c}|A_j| \geq \kappa) \leq 2(p-s)e^{-\frac{\kappa^2}{2M}} + 2e^{-\frac{s}{2}} + n^{-1}\theta^2.$$

We need to select M so that the exponential term is decaying in the above display. A sufficient condition for this is $\kappa^2/(2M) \geq 2\log(p-s)$. The last is equivalent to:

$$n_{p,s} \geq \frac{4D_{\max}(\boldsymbol{\Sigma}_{S_0^c|S_0})\left(\frac{4}{\lambda_{\min}} + \frac{\xi^2+1}{\lambda^2 s}\right)}{\kappa^2}.$$

3.2 Verifying Sign Consistency

The first part of the proof shows that the LASSO has a unique solution $\widehat{\boldsymbol{\beta}}$ which satisfies $S(\widehat{\boldsymbol{\beta}}) \subseteq S(c_0\boldsymbol{\beta}_0)$ with high probability. Now we need to verify the sign-consistency, in order to show that the supports coincide. We have the following:

$$\max_{j\in S_0}|\Delta_j| \leq \underbrace{\lambda\left\|(n^{-1}\mathbb{X}_{\cdot,S_0}^{\intercal}\mathbb{X}_{\cdot,S_0})^{-1}\operatorname{sign}(c_0\boldsymbol{\beta}_{0S_0})\right\|_{\infty}}_{\boldsymbol{I}_1} + \underbrace{\left\|(\mathbb{X}_{\cdot,S_0}^{\intercal}\mathbb{X}_{\cdot,S_0})^{-1}\mathbb{X}_{\cdot,S_0}^{\intercal}\mathbf{w}\right\|_{\infty}}_{\boldsymbol{I}_2}.$$

To deal with the first term we need the following:

Lemma 3.2.1 *There exist positive constants $K_1, C_2 > 0$, such that the following holds:*

$$\mathbb{P}(\boldsymbol{I}_1 \geq \lambda K_1\|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}^2) \leq 4\exp(-C_2(s\wedge\log(p-s))),$$

The proof of this lemma is part of the proof of Theorem 3 in Wainwright (2009) and we omit the details. Next we turn to bounding the term $\boldsymbol{I}_2$. Here our proof departs substantially from the proof in Wainwright (2009), as $\boldsymbol{I}_2$ no longer has a simple structure required in the original argument. In our case $\mathbf{w}$ depends on $\mathbb{X}_{\cdot,S_0}$, and it is not mean 0. We will make usage of the following result, whose proof is provided in the appendix

Lemma 3.2.2 *Let $\|\boldsymbol{\beta}_0\|_2 = 1$. We have n i.i.d. observations $Y = f(\boldsymbol{X}_{S_0}^{\intercal}\boldsymbol{\beta}_{0S_0}, \varepsilon)$ from a SIM, where $\boldsymbol{X}_{S_0} \sim \mathcal{N}(0, \mathbb{I}_{s\times s})$, with $s < n$ and $n_{p,s} \geq \alpha > 0$ for some positive constant α. Then there exist some positive constants $\Upsilon_1, \Upsilon_2 > 0$ (depending on σ and $|c_0|$), such that:*

$$\|[\mathbb{X}_{\cdot,S_0}^{\intercal}\mathbb{X}_{\cdot,S_0}]^{-1}\mathbb{X}_{\cdot,S_0}^{\intercal}\boldsymbol{Y} - c_0\boldsymbol{\beta}_{0S_0}\|_{\infty} \leq \Big(\Upsilon_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_{\infty} + \Upsilon_2\sqrt{\frac{\log(p-s)}{n}}\Big).$$

with probability at least $1 - O\{e^{-s/2} + (\log p)^{-1} + n^{-1} + p^{-1}\}$. Denote for brevity the RHS of the inequality as $\delta(\|\boldsymbol{\beta}_{0S_0}\|_{\infty}, n, s, p)$.

While Lemma 3.2.2 is stated in terms of standard multivariate normal distribution $\mathcal{N}(0, \mathbb{I}_{s\times s})$, we can easily adapt it to more general situations where we observe non-standard normal random variables $\mathcal{N}(0, \boldsymbol{\Sigma}_{S_0,S_0})$. Recall that the rows of $\mathbb{X}_{\cdot,S_0}$ are distributed as $\mathcal{N}(0, \boldsymbol{\Sigma}_{S_0,S_0})$, $Y_i = f(\boldsymbol{X}_i^{\intercal}\boldsymbol{\beta}_0, \varepsilon)$, and $\boldsymbol{\beta}_{0S_0}^{\intercal}\boldsymbol{\Sigma}_{S_0,S_0}\boldsymbol{\beta}_{0S_0} = 1$. Denote with $\mathbf{Z} = \mathbb{X}_{\cdot,S_0}\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}$. Then we have the following inequality, with high probability:

$$\begin{aligned}\boldsymbol{I}_2 = \|[\mathbb{X}_{\cdot,S_0}^{\intercal}\mathbb{X}_{\cdot,S_0}]^{-1}\mathbb{X}_{\cdot,S_0}^{\intercal}\boldsymbol{Y} - c_0\boldsymbol{\beta}_{0S_0}\|_{\infty} &= \|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}[\mathbf{Z}^{\intercal}\mathbf{Z}]^{-1}\mathbf{Z}^{\intercal}\boldsymbol{Y} - c_0\boldsymbol{\beta}_{0S_0}\|_{\infty}\\ &\leq \|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}\|[\mathbf{Z}^{\intercal}\mathbf{Z}]^{-1}\mathbf{Z}^{\intercal}\boldsymbol{Y} - c_0\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}\boldsymbol{\beta}_{0S_0}\|_{\infty}\\ &\leq \|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}\delta(\|\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}\boldsymbol{\beta}_{0S_0}\|_{\infty}, s, n, p).\end{aligned}$$

The last two inequalities imply that:

$$\max_{j\in S}|\Delta_j| \leq \lambda K_1\|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}^2 + \|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}\delta(\|\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}\boldsymbol{\beta}_{0S_0}\|_{\infty}, s, n, p).$$

Hence as long as for $\boldsymbol{\beta}_0^{\min} = \min\{|\beta_{0j}| : j \in S_0\}$ we have:

$$|c_0|\boldsymbol{\beta}_0^{\min} \geq \lambda K_1\|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}^2 + \|\boldsymbol{\Sigma}_{S_0,S_0}^{-1/2}\|_{\infty,\infty}\delta(\|\boldsymbol{\Sigma}_{S_0,S_0}^{1/2}\|_{\infty,\infty}\|\boldsymbol{\beta}_0\|_{\infty}, s, n, p),$$

the LASSO will recover the support with high-probability. This concludes the proof.

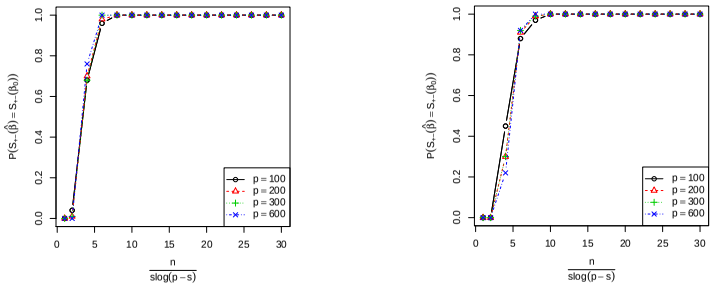
4. Numerical Studies

To support our theoretical claims, and in particular Theorem 2.3.4 we provide brief numeric analysis in this section. We consider the following models:

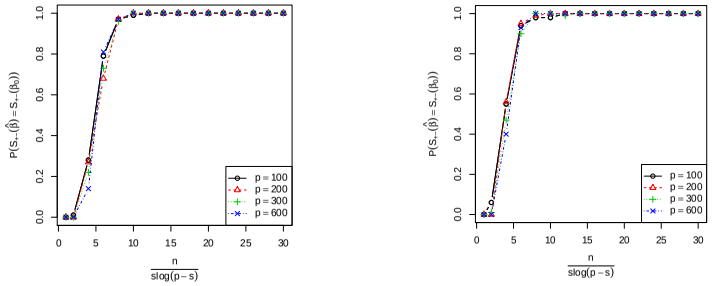
$$Y = \boldsymbol{X}^{\intercal}\boldsymbol{\beta}_0 + \sin(\boldsymbol{X}^{\intercal}\boldsymbol{\beta}_0) + \mathcal{N}(0,1), \tag{14}$$
$$Y = 2\operatorname{atan}(\boldsymbol{X}^{\intercal}\boldsymbol{\beta}_0) + \mathcal{N}(0,1), \tag{15}$$
$$Y = (\boldsymbol{X}^{\intercal}\boldsymbol{\beta}_0)^3 + \mathcal{N}(0,1), \tag{16}$$
$$Y = \sinh(\boldsymbol{X}^{\intercal}\boldsymbol{\beta}_0) + \mathcal{N}(0,1). \tag{17}$$

We use a Toeplitz covariance matrix for the simulations $\boldsymbol{\Sigma} = \mathbb{I}$ and $\Sigma_{kj} = 2^{-|k-j|}$. The vector $\boldsymbol{\beta}_0$ is selected so that $\boldsymbol{\beta}_0^{\intercal}\boldsymbol{\Sigma}\boldsymbol{\beta}_0 = 1$, and its entries have equal magnitude, with the first one having a negative sign, and the remaining being positive. We check whether the solution path of the LASSO contains an s-sparse vector $\boldsymbol{\beta}_0$ whose support coincides with the support of $\boldsymbol{\beta}_0$. This verifies the validity of one implication of our theory, as it shows that the solution path indeed contains the true signed support of $\boldsymbol{\beta}_0$.

Figure 1: LASSO, $s = \sqrt{p}$, $\boldsymbol{\Sigma} = \mathbb{I}_{p\times p}$



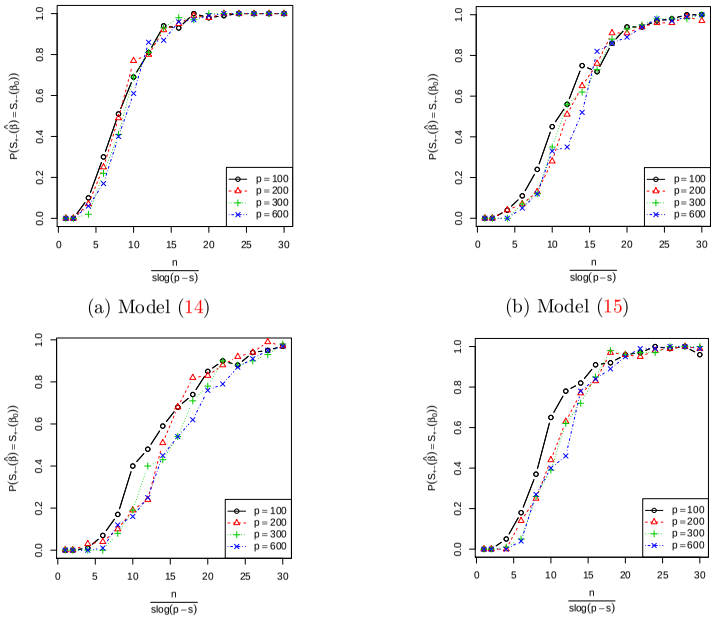
(a) Model (14) (b) Model (15)



(c) Model (16) (d) Model (17)

In Figure 1, we show results of signed support recovery for different values of p, in the regime $s = \sqrt{p}$ in the case of an identity covariance matrix $\mathbf{\Sigma} = \mathbb{I}_{p\times p}$. Similarly, Figure 2 shows results for the case of Toeplitz covaraince matrix $\Sigma_{kj} = 2^{-|k-j|}$. As expected, the support recovery is harder in the presence of correlation between the variables. These results illustrate different phase transitions occurring for the four different models. We observe empirically that values of $n_{p,s}$ achieving reasonable success probability can be large in some cases. It could be the case that using a transformed version of Y might lead to better results for the model complexity adjusted sample size, as suggested in Section 2.3.1. Figure 2 supports the result of Theorem 2.3.4 as all curves merge when the effective sample size $n_{p,s}$ becomes sufficiently large. In addition, these results suggest that the performance of the support recovery is largely determined by (n, p, s) through the magnitude of $n_{p,s}$.

Figure 2: LASSO, $s = \sqrt{p}$, $\Sigma_{kj} = 2^{-|k-j|}$



(a) Model (14) (b) Model (15)

(c) Model (16) (d) Model (17)

In addition to the verification of our theory, we also compare the vanilla least squares LASSO to a *version* of the sparse sliced inverse regression (SSIR) algorithm suggested by Li and Nachtsheim (2006). The SSIR algorithm is also based on a LASSO estimation. In its original form however, this algorithm is not applicable for high-dimensional settings such as ours, since it needs an estimate of the matrix $\mathbf{\Sigma}^{-1/2}$. To make use of the SSIR under the

high-dimensional setting, we estimate $\mathbf{\Sigma}^{-1/2}$ by the CLIME procedure (Cai et al., 2011) to $\widehat{\mathbf{\Sigma}}^{1/2}$ under sparsity assumptions. Due to space considerations we only show comparisons for models (14) and (17), and the plots are attached in Appendix F. In the majority of cases LASSO outperforms the SSIR algorithm substantially for small values of $n_{p,s}$, although it seems that both approaches reach perfect support recovery at similar rescaled sample sizes. We would like to emphasize that the SSIR algorithm requires solving an extra optimization problem, and that furthermore, there are no theoretical results ensuring that SSIR in general recovers the support. On an important note, the SSIR algorithm is designed to estimate the central space of the more general class of multi-index models, which we do not discuss in the present paper. For a brief discussion on how our work can be related to multi-index model please refer to section 5.

5. Discussion

In this paper, we demonstrate that under a high dimensional SIM, L_1-regularized least squares, including a simplified covariance screening procedure under orthogonal design, is robust in terms of model selection consistency, in that it correctly recovers the support of the true regression parameter $\boldsymbol{\beta}_0$ provided that $c_0 = \mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) \neq 0$, the minimal signal strength is sufficiently large and $\boldsymbol{X} \sim \mathcal{N}(0, \mathbf{\Sigma})$ under standard assumptions on $\mathbf{\Sigma}$ which are necessary even in the linear regression case. Thus, our results extend known results on the support recovery performance of LASSO under linear models to a much broader class of SIMs. We furthermore demonstrate that the support recovery is achieved in a sample size optimal $n_{p,s}$ manner within a certain class of SIMs.

As we indicated in section 2.3.1, the assumption $c_0 \neq 0$ does not always hold, and in addition it cannot be easily verified. A potential remedy for this approach will be to transform the Y variable. From theoretical point of view it is of interest to develop procedures which can adaptively estimate a "good" outcome transformation. Additionally, a downside of the L_1-regularization is the fact that the irrepresentable condition on the covariance matrix is unavoidable. This could potentially be remedied by using more general and non-convex penalties such as the SCAD penalty (Fan and Li, 2001). We focused on the setting with $\boldsymbol{X} \sim \mathcal{N}(0, \mathbf{\Sigma})$, however we suspect that the support recovery holds in more general cases where $\boldsymbol{X}$ comes from an elliptical distribution. It is less clear, however, whether sample size optimality continues to hold in such situations, as we crucially rely on the normality of $\boldsymbol{X}$, in particular when using Lemma A.0.1 which follows by Gordon's comparison theorem, and through numerous projection-independence properties which are characteristic of the Gaussian distribution.

The proposed method focuses on SIMs as the true underlying model. Extensions to incorporate general multi-index models are not straightforward. For the special case of multi-index models of the form

$$Y = \sum_{j=1}^{k} f_j(\boldsymbol{X}^\intercal\boldsymbol{\beta}_j, \varepsilon_j),$$

our method should also be able to recover the support, assuming that k is fixed and the vectors $\boldsymbol{\beta}_j$ have disjoint supports, and $\mathbb{E}(Y\boldsymbol{X}^\intercal\boldsymbol{\beta}_j) \neq 0$ for all $j \in [k]$. When applied to such a model, the LASSO estimate $\widehat{\boldsymbol{\beta}}$ will include the union of the supports of $\boldsymbol{\beta}_j$, provided that

sufficiently strong minimal signal (or sufficiently large sample size) and an irrepresantable condition are present. How to apply the LASSO algorithm for support recovery under the more general class of multi-index models warrants further research.

A note on the choice of the tuning parameter λ in practice is also in order. According to Remark 2.3.5 and optimal choice of λ may depend on the unknowable parameter ξ^2. A procedure which we found to work well in practice is based on simple cross validation. To find a good tuning parameter λ, from a grid of ℓ values of λ: $\{\lambda_1, \ldots, \lambda_\ell\}$ which are of $\sqrt{\frac{\log p}{n}}$ magnitude, we recommend using K-fold cross validation, and setting λ to the value minimizing the average least squares loss across (i.e. mean squared-error) the K folds. Notice that according to Lemma 2.1.3, this criteria is very sensible. An important question is whether one can arrive at a procedure with theoretical guarantees for λ selection, and we hope to address this problem in our future work.

Finally, under a SIM and proper distributional assumptions on $\boldsymbol{X}$, one may also recover $\boldsymbol{\beta}_0$ proportionally using other convex loss functions. For example, when Y is binary, the logistic log-likelihood loss may be more efficient than the L_2 loss. Thus, it is of interest to investigate the support recovery properties of the LASSO (or more general penalization procedures) with other convex losses — such as the logistic/hinge losses, which could be less susceptible to outliers.

Acknowledgments

The authors would like to thank the Associate Editor and anonymous reviewers for their insightful remarks which led to the improvement of this manuscript. We are also grateful to Professor Noureddine El Karoui, who among other observations, brought to our attention that outcome transformations may be performed to facilitate the usage of LASSO even when $c_0 = 0$ for the original outcome Y. This research was partially supported by Research Grants NSF DMS1208771, NIH R01 GM113242-01, NIH U54 HG007963 and NIH RO1 HL089778.

Appendix A. Auxiliary Lemmas

In this section, for convenience of the reader, we state couple of lemmas that we use often in our analysis.

Lemma A.0.1 (Corollary 5.35 Vershynin (2010)) *Let $\mathbf{A}_{n\times s}$ matrix whose entries are i.i.d. standard normal random variables. Then for every $t \geq 0$, with probability at least $1 - 2\exp(-t^2/2)$ one has:*

$$\sqrt{n} - \sqrt{s} - t \leq s_{\min}(\mathbf{A}) \leq s_{\max}(\mathbf{A}) \leq \sqrt{n} + \sqrt{s} + t,$$

where $s_{\min}(\mathbf{A})$ and $s_{\max}(\mathbf{A})$ are the smallest and largest singular values of $\mathbf{A}$ correspondingly.

Lemma A.0.2 *Consider a fixed nonzero vector $\mathbf{z} \in \mathbb{R}^s$ and a random matrix $\mathbf{A}_{n\times s}$, whose entries are i.i.d. standard normal random variables. If p, s, n are such that $s \geq 2$, $\frac{s}{n} \leq \frac{1}{64}$ and $\frac{\log p}{n-s+1} \leq \frac{1}{32}$, there are positive absolute constants C_1 and C_2 satisfying:*

$$\mathbb{P}\left(\|[(n^{-1}\mathbf{A}^\intercal\mathbf{A})^{-1} - \mathbb{I}_{s\times s}]\mathbf{z}\|_\infty \geq C_1\frac{s}{n}\|\mathbf{z}\|_\infty + C_2\|\mathbf{z}\|_2\sqrt{\frac{\log p}{n}} \right) \leq 4p^{-1}.$$

Proof [Proof of Lemma A.0.2] This Lemma is a generalization/modification of Lemma 5 of Wainwright (2009), allowing us to make usage of the L_2 norm $\|\mathbf{z}\|_2$ to obtain more precise bounds. For self-content we spell out the full details of the proof. Using the spectral theorem, decompose the matrix $(n^{-1}\mathbf{A}^\intercal\mathbf{A})^{-1} - \mathbb{I} = \mathbf{V}\mathbf{D}\mathbf{V}^\intercal$, where $\mathbf{D}$ is a diagonal matrix and $\mathbf{V}$ is an independent of $\mathbf{D}$ unitary matrix. Define the random variables:

$$U_i = \mathbf{e}_i^\intercal\mathbf{V}\mathbf{D}\mathbf{V}^\intercal\mathbf{z} = z_i\mathbf{v}_i^\intercal\mathbf{D}\mathbf{v}_i + \mathbf{v}_i^\intercal\mathbf{D}\sum_{j\neq i} z_j\mathbf{v}_j,$$

where $\mathbf{v}_i^\intercal$ is the i^{th} row vector of the matrix $\mathbf{V}$. To bound $\max_i |U_i|$ we deal with these two terms in turn. First notice that $\mathbf{v}_i^\intercal\mathbf{D}\mathbf{v}_i$ is the i^{th} diagonal entry of the matrix $(n^{-1}\mathbf{A}^\intercal\mathbf{A})^{-1} - \mathbb{I}$. By the assumption $(\mathbf{A}^\intercal\mathbf{A})^{-1} \sim \mathcal{W}^{-1}(\mathbb{I}_{s\times s}, n)$ where $\mathcal{W}^{-1}$ is an inverse Wishart distribution. By the properties of the inverse Wishart distribution we conclude that $\mathbf{v}_i^\intercal\mathbf{D}\mathbf{v}_i \sim n\chi^{-2}(n-s+1) - 1$, where χ^{-2} is the inverse χ^2 distribution. Hence using Lemma 1 of Laurent and Massart (2000) and the union bound we have:

$$\mathbb{P}(\max_i |1 - ((\mathbf{v}_i^\intercal\mathbf{D}\mathbf{v}_i + 1)(n-s+1)/n)^{-1}| \geq 2\sqrt{y} + 2y) \leq 2s\exp(-(n-s+1)y).$$

Selecting $y = \frac{2\log p}{n-s+1}$ bounds the above probability by $2s/p^2 \leq 2/p$. For values of $y < 1/32$ we can lump $2\sqrt{y} + 2y < \left(2 + \frac{\sqrt{2}}{4}\right)\sqrt{y}$. Thus inverting the inequality inside the probability we conclude that for each $i \in [s]$:

$$\frac{n}{(n-s+1)(1 + \left(2+\frac{\sqrt{2}}{4}\right)\sqrt{\frac{2\log p}{n-s+1}})} \leq \mathbf{v}_i^\intercal\mathbf{D}\mathbf{v}_i + 1 \leq \frac{n}{(n-s+1)(1 - \left(2+\frac{\sqrt{2}}{4}\right)\sqrt{\frac{2\log p}{n-s+1}})}$$

It is simple to see that when $\frac{\log(p)}{n-s+1} \leq \frac{1}{32}$ the above implies:

$$\max_i |z_i \mathbf{v}_i^\mathsf{T} \mathbf{D} \mathbf{v}_i| < \|\mathbf{z}\|_\infty \widetilde{c}_1 \frac{s}{n-s} + \|\mathbf{z}\|_2 \widetilde{c}_2 \sqrt{\frac{\log p}{n-s+1}}, \tag{18}$$

where $\widetilde{c}_1 = (1-(1/2+\sqrt{2}/16))^{-1}$ and $\widetilde{c}_2 = (2+\sqrt{2}/4)\widetilde{c}_1$. Next we show that the function $F(\mathbf{v}_i) = \mathbf{v}_i^\mathsf{T} \mathbf{D} \sum_{j \neq i} z_j \mathbf{v}_j$ is Lipschitz with a constant $8\sqrt{s/n}\|\mathbf{z}\|_2$. We have:

$$\|\nabla F\|_2 \leq \|\mathbf{D}\|_{2,2} \Big\| \sum_{j \neq i} z_j \mathbf{v}_j \Big\|_2 \leq 9\sqrt{\frac{s}{n}} \|\mathbf{z}\|_2,$$

with the last inequality holding with probability at least $1 - 2\exp(-s/2)$ when $\frac{s}{n} \leq \frac{1}{64}$. We used that $\mathbf{v}_j$ are orthonormal, and we bounded the maximum eigenvalue of $\mathbf{D}$, which follows just as in the proof of Lemma E.0.2 so we omit the details. Since the variables $\{\mathbf{v}_j\}_{j \neq i}$ are uniformly distributed on a $(s-1)$-dimensional sphere, the proof is completed by invoking the concentration of Lipschitz functions on the sphere to bound $\max_{i \in [s]} |F(\mathbf{v}_i)|$:

$$\mathbb{P}(\max_{i \in [s]} |F(\mathbf{v}_i)| \geq t) \leq 2s \exp\left(-\widetilde{c}(s-1) \frac{t^2}{81 \frac{s}{n} \|\mathbf{z}\|_2^2} \right),$$

for an absolute constant $\widetilde{c}$. Under the assumption $s \geq 2$, we can select $t = 18\|\mathbf{z}\|_2 \sqrt{\frac{\log p}{\widetilde{c} n}}$ and taking into account that $p > s$ completes the proof, after noticing that we can absorb the second term of (18) to the above expression. ■

Appendix B. Preliminary Results

Proof [Proof of Lemma 2.1.3] First let $\mathbf{\Sigma} = \mathbb{I}_{p \times p}$ (hence by assumption $\|\boldsymbol{\beta}_0\|_2 = 1$) and take any $\boldsymbol{b} \perp \boldsymbol{\beta}_0$. Note that by the linearity of expectation:

$$\mathbb{E}[\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0 \boldsymbol{X}^\mathsf{T}\boldsymbol{b} | \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0] = c_{\boldsymbol{b}} (\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0)^2.$$

Taking another expectation above we have $\mathbb{E}[\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0 \boldsymbol{X}^\mathsf{T}\boldsymbol{b}] = c_{\boldsymbol{b}} \mathbb{E}[(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0)^2]$. However

$$\mathbb{E}[\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0 \boldsymbol{X}^\mathsf{T}\boldsymbol{b}] = \boldsymbol{b}^\mathsf{T}\boldsymbol{\beta}_0 = 0,$$

and hence $c_{\boldsymbol{b}} = 0$. Thus if $\boldsymbol{b} \perp \boldsymbol{\beta}_0$, $\mathbb{E}[\boldsymbol{X}^\mathsf{T}\boldsymbol{b} | \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0] = 0$. Next, for any $\boldsymbol{b} \perp \boldsymbol{\beta}_0$ we have:

$$\mathbb{E}[Y \boldsymbol{X}^\mathsf{T}\boldsymbol{b}] = \mathbb{E}[\mathbb{E}[Y \boldsymbol{X}^\mathsf{T}\boldsymbol{b} | \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0]] = \mathbb{E}[\mathbb{E}[Y | \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0] \mathbb{E}[\boldsymbol{X}^\mathsf{T}\boldsymbol{b} | \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0]] = 0.$$

Hence $\mathbb{E}[Y\boldsymbol{X}] \propto \boldsymbol{\beta}_0$. Finally, a projection on $\boldsymbol{\beta}_0$ yields

$$c_0 \|\boldsymbol{\beta}_0\|_2^2 = \mathbb{E}[Y \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0].$$

This completes the proof in the case when $\mathbf{\Sigma} = \mathbb{I}_{p \times p}$. For the more general case observe that $Y = f(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0, \varepsilon) = f(\boldsymbol{X}^\mathsf{T}\mathbf{\Sigma}^{-1/2}\mathbf{\Sigma}^{1/2}\boldsymbol{\beta}_0, \varepsilon)$, and thus by what we just saw we have:

$$\mathbb{E}[Y \mathbf{\Sigma}^{-1/2} \boldsymbol{X}] = c_0 \mathbf{\Sigma}^{1/2} \boldsymbol{\beta}_0,$$

which becomes what we wanted to show after multiplying by $\mathbf{\Sigma}^{-1/2}$ on the left. ■

Appendix C. Lower Bound

For two probability measures P and Q, which are absolutely continuous with respect to a third probability measure μ (i.e. $P, Q \ll \mu$), define their KL divergence by $D_{\mathrm{KL}}(P\|Q) = \int p \log \frac{p}{q} d\mu$, where $p = \frac{dP}{d\mu}$, $q = \frac{dQ}{d\mu}$.

Lemma C.0.1 *Assume conditions required in Proposition 2.3.6. In addition, let for any fixed $u, v \in \mathbb{R}$ and some positive constant Ξ, $f(u,\varepsilon)$ and $f(v,\varepsilon)$ satisfy*

$$D_{KL}(p(f(u,\varepsilon)) \| p(f(v,\varepsilon))) \leq \exp(\Xi(u-v)^2) - 1, \tag{19}$$

Then if

$$n_{p,s} < \frac{1}{8\Xi}, \quad \textit{and} \quad s \geq 8\Xi,$$

any algorithm recovering the support of $\boldsymbol{\beta}_0$ under (1) will have errors with probability at least $\frac{1}{2}$ asymptotically.

Proof [Proof of Lemma C.0.1] We start by constructing a set of $p-s$ vectors $\boldsymbol{B} = \{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{p-s}\}$, belonging to the parameter space $\{\boldsymbol{\beta} \in \mathbb{R}^p : \boldsymbol{\beta}^\mathsf{T}\mathbf{\Sigma}\boldsymbol{\beta} = 1, \|\boldsymbol{\beta}\|_0 = s, \frac{\boldsymbol{\beta}^{\min}}{\|\boldsymbol{\beta}\|_\infty} \geq c_{\mathbf{\Sigma}}\}$, for a sufficiently small (to be chosen) $c_{\mathbf{\Sigma}} > 0$, such that $\mathrm{Var}(\boldsymbol{X}^\mathsf{T}(\boldsymbol{\beta}_k - \boldsymbol{\beta}_j)) \leq \frac{4}{s}$ for all $k, j \in [p-s]$. Once this set is constructed we will use standard Fano type of argument to finish the proof.

Without loss of generality let us assume that $S_0 = [s]$. To construct the set $\boldsymbol{B}$, first focus on the sub-matrix $\mathbf{\Sigma}_{S_0,S_0}$. We take the s dimensional vector $\boldsymbol{\gamma} = a(1/\sqrt{s}, \ldots, 1/\sqrt{s}, 0)^\mathsf{T}$ where $a > 0$ is selected so that $\boldsymbol{\gamma}^\mathsf{T}\mathbf{\Sigma}_{S_0,S_0}\boldsymbol{\gamma} = \frac{s-1}{s}$. Since $\mathbf{\Sigma}_{S_0,S_0}$ is assumed to have bounded eigenvalues we know that such a indeed exists, and can be chosen in the interval $a \in [\frac{1}{2}\frac{1}{\sqrt{\lambda_{\max}}}, \frac{1}{\sqrt{\lambda_{\min}}}]$ for $s \geq 2$. Next to construct $\boldsymbol{\beta}_k$ we use:

$$\beta_{kr} = \gamma_r \mathbb{1}(r \in S_0) + b_k \mathbb{1}(r = k+s)/\sqrt{s},$$

where b_k is chosen so that $\boldsymbol{\beta}_k^\mathsf{T}\mathbf{\Sigma}\boldsymbol{\beta}_k = 1$. Below we argue that such b_k indeed exists. By Hölder's inequality we have $\|\mathbf{\Sigma}_{S_0,S_0}\boldsymbol{\gamma}\|_\infty \leq a\|\mathbf{\Sigma}_{S_0,S_0}\|_{\infty,\infty}/\sqrt{s} \leq aR/\sqrt{s}$. Hence using Assumption 2.3.1 we have:

$$\|\mathbf{\Sigma}_{S_0^c,S_0}\mathbf{\Sigma}_{S_0,S_0}^{-1}\mathbf{\Sigma}_{S_0,S_0}\boldsymbol{\gamma}\|_\infty \leq \|\mathbf{\Sigma}_{S_0^c,S_0}\mathbf{\Sigma}_{S_0,S_0}^{-1}\|_{\infty,\infty} aR/\sqrt{s} \leq (1-\kappa)aR/\sqrt{s}.$$

Note that due to the last inequality, for any $k \in [p-s]$ we have:

$$\boldsymbol{\beta}_k^\mathsf{T}\mathbf{\Sigma}\boldsymbol{\beta}_k = \boldsymbol{\gamma}^\mathsf{T}\mathbf{\Sigma}_{S_0,S_0}\boldsymbol{\gamma} + 2\mathbf{\Sigma}_{\{k+s\},S_0}\boldsymbol{\gamma}\frac{b_k}{\sqrt{s}} + \frac{b_k^2\Sigma_{k+s,k+s}}{s} \leq \frac{s-1}{s} + 2\frac{(1-\kappa)a|b_k|R}{s} + \frac{b_k^2\Sigma_{k+s,k+s}}{s},$$

where we remind the reader that $k+s \in S_0^c$. We chose b_k such that $\mathrm{sign}(b_k) = \mathrm{sign}(\mathbf{\Sigma}_{\{k+s\},S_0}\boldsymbol{\gamma})$. Hence we also have:

$$\frac{s-1}{s} + \frac{b_k^2\Sigma_{k+s,k+s}}{s} \leq \boldsymbol{\beta}_k^\mathsf{T}\mathbf{\Sigma}\boldsymbol{\beta}_k.$$

Combining the last two inequalities we conclude that there exists:

$$|b_k| \in \Big[\frac{\sqrt{(1-\kappa)^2a^2R^2 + \Sigma_{k+s,k+s}} - (1-\kappa)aR}{\Sigma_{k+s,k+s}}, \Sigma_{k+s,k+s}^{-1/2} \Big],$$

with the desired properties. One can easily check that when:

$$c_{\boldsymbol{\Sigma}} \leq \min\left(\frac{1}{2}\sqrt{\frac{d}{\lambda_{\max}}}, \frac{\sqrt{(1-\kappa)^2R^2 + d\lambda_{\min}} - (1-\kappa)R}{D}\right), \tag{20}$$

we have that $\min_{j\in[p-s]} \frac{\boldsymbol{\beta}_j^{\min}}{\|\boldsymbol{\beta}_j\|_\infty} \geq c_{\boldsymbol{\Sigma}}$, and in addition as we promised we have:

$$\mathrm{Var}(\boldsymbol{X}^\mathsf{T}(\boldsymbol{\beta}_k - \boldsymbol{\beta}_j)) = \frac{\Sigma_{k+s,k+s}b_k^2 - 2\Sigma_{k+s,j+s}b_kb_j + \Sigma_{j+s,j+s}b_j^2}{s} \leq \frac{4}{s}.$$

Next, let $\mathcal{J}$ be a uniform distribution on $\boldsymbol{B}$. Under the $0-1$ loss the risk equals the probability of error:

$$\frac{1}{p-s}\sum_{j\in[p-s]} P_{\boldsymbol{\beta}_j}\{\widehat{S} \neq S(\boldsymbol{\beta}_j)\}, \tag{21}$$

where by $P_{\boldsymbol{\beta}_j}$ we are measuring the probability under a dataset generated with $\boldsymbol{\beta}_j$, and $\widehat{S}$ is an estimate of the true support produced by any (possibly randomized) algorithm. By Fano's inequality that:

$$\mathbb{P}(\text{error}) \geq 1 - \frac{\mathcal{I}(\mathcal{J};\mathcal{D}) + \log(2)}{\log|\boldsymbol{B}|}, \tag{22}$$

where $\mathcal{I}(\mathcal{J};\mathcal{D})$ is the mutual information between the sample $\mathcal{J}$ and the sample $\mathcal{D}$. Note now that for the mutual information we have

$$\begin{aligned}\mathcal{I}(\mathcal{J};\mathcal{D}) &= \mathcal{I}(\mathcal{J};[\{f(\boldsymbol{X}_j^\mathsf{T}\boldsymbol{\beta}_0,\varepsilon_j),\boldsymbol{X}_j\}, j=1,\ldots,n]) \\ &\leq n\mathcal{H}[\{f(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0,\varepsilon),\boldsymbol{X}\}] - n\mathcal{H}[\{f(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0,\varepsilon),\boldsymbol{X}\}|\mathcal{J}] \\ &\leq n\max_{k,j\in[p-s]} D_{\mathrm{KL}}\big(p[\{f(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_k,\varepsilon),\boldsymbol{X}\}]\|p[\{f(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_j,\varepsilon),\boldsymbol{X}\}]\big),\end{aligned}$$

where $\mathcal{H}(\cdot)$ denotes the marginal entropy, $\mathcal{H}(\cdot\mid\cdot)$ denotes the conditional entropy, D_{KL} denotes the KL divergence, the first inequality follows from the chain inequality of entropy and the second inequality follows from a standard bound. Since the KL divergence is invariant under change of variables, we let $U_k = \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_k$ and $\boldsymbol{W}_{kj} = \mathbf{P}_{\boldsymbol{\Sigma},\{\boldsymbol{\beta}_k,\boldsymbol{\beta}_j\}^\perp}\boldsymbol{X}$, where $\mathbf{P}_{\boldsymbol{\Sigma},\{\boldsymbol{\beta}_k,\boldsymbol{\beta}_j\}^\perp} \in \mathbb{R}^{(p-2)\times p}$ is chosen such that $\mathbf{P}_{\boldsymbol{\Sigma},\{\boldsymbol{\beta}_k,\boldsymbol{\beta}_j\}^\perp}\boldsymbol{\Sigma}\boldsymbol{\beta}_k = 0$ and $\mathbf{P}_{\boldsymbol{\Sigma},\{\boldsymbol{\beta}_k,\boldsymbol{\beta}_j\}^\perp}\boldsymbol{\Sigma}\boldsymbol{\beta}_j = 0$. Noting that $\boldsymbol{W}_{kj}$ is independent of $U_k, U_j, \varepsilon, U_k - U_j \sim \mathcal{N}(0,V)$ with $V \leq 4/s$, and applying assumption (19), we have

$$\begin{aligned}n^{-1}\mathcal{I}(\mathcal{J};\mathcal{D}) &\leq \max_{k,j\in[p-s]} D_{\mathrm{KL}}\big(p[\{f(U_k,\varepsilon),U_k,U_j\}]\|p[\{f(U_j,\varepsilon),U_k,U_j\}]\big) \\ &\leq \max_{k,j\in[p-s]} \mathbb{E}\exp(\Xi(U_k-U_j)^2) - 1 \leq \sqrt{\frac{s}{s-8\Xi}} - 1 \leq \frac{8\Xi}{s}.\end{aligned}$$

where the second inequality can be obtained by conditioning on U_k, U_j, and we assume that the value of s is large enough so that $s \geq 16\Xi$. We conclude that:

$$\mathcal{I}(\mathcal{J};\mathcal{D}) \leq \frac{8\Xi n}{s}.$$

Consequently by (22) if $n_{p,s} < 1/(8\Xi)$ we will have errors with probability at least $\frac{1}{2}$, asymptotically. This is what we wanted to show. ■

Proof [Proof of Proposition 2.3.6] Note that all moments of the random variable ε exist. Next we verify that condition (19) of Lemma C.0.1 holds in this setup. Since G is 1-1 and KL divergence is invariant under changes of variables WLOG we can assume our model is simply $Y = h(\boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}_0) + \varepsilon$ or in other words $f(u,\varepsilon) = h(u) + \varepsilon$. This is a location family for $u \in \mathbb{R}$ and thus the normalizing constant of the densities will stay the same regardless of the value of u. Direct calculation yields:

$$D_{\mathrm{KL}}[p\{f(u,\varepsilon)\}\|p\{f(v,\varepsilon)\}] = \mathbb{E}[P((\xi + h(u) - h(v))^2) - P(\xi^2)] = \widetilde{P}((h(u)-h(v))^2),$$

where ξ has a density $p_\xi(x) \propto \exp(-P(x^2))$, and $\widetilde{P}$ is another non-zero polynomial with nonnegative coefficients, with $\widetilde{P}(0) = 0$ of the same degree as P. The last observation follows from the fact that all odd moments of ξ are 0, since ξ is a symmetric about 0 distribution. Since h is L-Lipschitz we conclude that:

$$D_{\mathrm{KL}}[p\{f(u,\varepsilon)\}\|p\{f(v,\varepsilon)\}] \leq \widetilde{P}(L^2(u-v)^2).$$

The last can be clearly dominated by $\exp(C(u-v)^2) - 1$ for a large enough constant C. ■

Appendix D. Covariance Thresholding

Proof [Proof of Proposition 2.2.3] Using the fact that for any two random variables R, T we have $\|RT\|_{\psi_1} \leq 2\|R\|_{\psi_2}\|T\|_{\psi_2}$ we can conclude that the random vector $Y\boldsymbol{X}$ is coordinate-wise sub-exponentially distributed since $\sup_{j\in[p]}\|YX_j\|_{\psi_1} \leq \mathcal{K} := 2K_YK$. An application of Proposition 5.16 of Vershynin (2010) and the union bound then gives us that:

$$\mathbb{P}\left(\left\|\frac{1}{n}\sum_{i=1}^n Y_i\boldsymbol{X}_i - \mathbb{E}[Y\boldsymbol{X}]\right\|_\infty \geq t\right) \leq 2p\exp\left[-\widetilde{c}\min\left(\frac{nt^2}{\mathcal{K}^2}, \frac{nt}{\mathcal{K}}\right)\right],$$

where $\widetilde{c} > 0$ is some absolute constant. This inequality then implies:

$$\sup_{j\in[p]}\left|\frac{1}{n}\sum_{i=1}^n Y_iX_{ij} - \mathbb{E}(YX_j)\right| \leq \mathcal{K}\sqrt{\frac{2\log p}{\widetilde{c}n}},$$

with probability at least $1 - 2p^{-1}$ for values of n, p such that $\frac{\log p}{n} \leq \frac{\widetilde{c}}{2}$. Note that the inequality in the preceding display implies that if:

$$\frac{|c_0|}{\sqrt{s}} > R\mathcal{K}\sqrt{\frac{2\log p}{\widetilde{c}n}},$$

for any $R > 2$ there will be a gap in the absolute values of the coefficients of $U_j = |n^{-1}\sum_{i=1}^n Y_iX_{ij}|$ for $j \in S_0$ and $j \notin S_0$. The latter happens because:

$$\frac{|c_0|}{\sqrt{s}} - \mathcal{K}\sqrt{\frac{2\log p}{\widetilde{c}n}} \geq (R-1)\mathcal{K}\sqrt{\frac{2\log p}{\widetilde{c}n}} > \mathcal{K}\sqrt{\frac{2\log p}{\widetilde{c}n}}.$$

This also shows that the coefficients will achieve the correct sign. Thus, as long as $\frac{n}{s\log p} \geq \Upsilon$, for $\Upsilon = \frac{2R^2K^2}{c_0^2\tilde{c}^2}$ signed support recovery happens with asymptotic probability 1. Under our assumption the latter is implied by $n_{p,s} > \Upsilon/\iota$ which completes the proof. ■

Proof [Proof of Proposition 2.2.1] We follow the same steps as the proof of Proposition 2.2.3. We will use the following Lemma which we justify after the proof:

Lemma D.0.1 *Let us observe n data points from the model described in Proposition 2.2.1 with $\boldsymbol{\beta}$ being an arbitrary unit vector. Then with probability at least $1 - \frac{\eta}{n\sigma^4} - \frac{\gamma}{\log p} - \frac{2}{p}$ the following event holds:*

$$\left\| \frac{1}{n}\sum_{i=1}^{n} Y_i \boldsymbol{X}_i - \mathbb{E}(Y\boldsymbol{X}) \right\|_\infty \leq \left(\|\boldsymbol{\beta}_0\|_\infty + 2\sqrt{2\sigma^2} \right) \sqrt{\frac{\log p}{n}}.$$

Using the fact that in our case $\mathbb{E}(Y\boldsymbol{X}) = c_0\boldsymbol{\beta}_0$, and that $\|\boldsymbol{\beta}_0\|_\infty = \frac{1}{\sqrt{s}}$, we have that if:

$$\left(\|\boldsymbol{\beta}_0\|_\infty + 2\sqrt{2\sigma^2} \right) \sqrt{\frac{\log p}{n}} \leq (1 + 2\sqrt{2}\sigma)\frac{1}{\sqrt{s}} \sqrt{\frac{s\log p}{n}} < \frac{|c_0|}{2}\frac{1}{\sqrt{s}}$$

there will be a gap between the coefficients corresponding to $j \in S_0 := S(\boldsymbol{\beta}_0)$ and $j \notin S_0$. Note that the last inequality holds if $\frac{s\log p}{n} < \frac{c_0^2}{4(1+2\sqrt{2}\sigma)^2}$. ■

Remark D.0.2 *The slow convergence in probability rate $(\log p)^{-1}$ observed in Lemma D.0.1 is due to the fact that we are not requiring that Y is sub-Gaussian. If we do require it, the convergence rate of the probability can be seen to reduce to the usual p^{-1} level.*

Proof [Proof of Lemma D.0.1] Note that, sub-exponential concentration bounds do not apply in this case. However, observe that by the properties of the multivariate normal distribution the random variable $(\mathbb{I} - \boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal)\boldsymbol{X}$ is independent of $\boldsymbol{X}^\intercal\boldsymbol{\beta}_0$ and hence is independent of Y. Furthermore it is clear that the random variable $Y(\mathbb{I} - \boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal)\boldsymbol{X}$ has mean 0. Note that conditional on $Y_i, i \in [n]$ we have that $\frac{1}{n}\sum_{i=1}^n Y_i(\mathbb{I} - \boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal)\boldsymbol{X}_i \sim \mathcal{N}(0, n^{-2}\sum_{i=1}^n Y_i^2(\mathbb{I} - \boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal))$. Thus by a standard Gaussian tail bound:

$$\mathbb{P}\left(\left\| \frac{1}{n}\sum_{i=1}^{n} Y_i(\mathbb{I} - \boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal)\boldsymbol{X}_i \right\|_\infty \geq t \middle| \boldsymbol{Y} \right) \leq 2p\exp\left[-\frac{nt^2}{2\overline{Y^2}} \right],$$

where $\overline{Y^2} = n^{-1}\sum_{i=1}^n Y_i^2$, and we used that $\|\mathbb{I} - \boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal\|_{2,2} \leq 1$. By Chebyshev's inequality $\mathbb{P}(|\overline{Y^2} - \sigma^2| \geq r) \leq \frac{\eta}{nr^2}$. Hence selecting $r = \sigma^2$ will keep the above probability going to 1 at rate $\frac{1}{n}$ and moreover for large n we have $\overline{Y^2} \leq 2\sigma^2$. Using this bound in the tail bound above yields that for a choice of $t = 2\sqrt{2\sigma^2\frac{\log p}{n}}$ the tail bound will go to 0 at rate $2p^{-1}$ as claimed.

Next consider controlling:

$$\mathbb{P}\left(\left\| \frac{1}{n}\sum_{i=1}^{n} Y_i\boldsymbol{\beta}_0\boldsymbol{\beta}_0^\intercal\boldsymbol{X}_i - c_0\boldsymbol{\beta}_0 \right\|_\infty \geq t \right) = \mathbb{P}\left(\left| \frac{1}{n}\sum_{i=1}^{n} Y_i\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0 - c_0 \right| \geq t/\|\boldsymbol{\beta}_0\|_\infty \right),$$

where recall that $\mathbb{E}(Y\boldsymbol{X}) = c_0\boldsymbol{\beta}_0$, and c_0 is defined in the main text. Applying Chebyshev's inequality once again we get that $t = \|\boldsymbol{\beta}_0\|_\infty\sqrt{\frac{\log p}{n}}$ suffices to keep the above probability going to 0. By the triangle inequality we conclude that, with probability going to 1:

$$\left\| \frac{1}{n}\sum_{i=1}^{n} Y_i\boldsymbol{X}_i - \mathbb{E}(Y\boldsymbol{X}) \right\|_\infty \leq \|\boldsymbol{\beta}_0\|_\infty\sqrt{\frac{\log p}{n}} + 2\sqrt{2\sigma^2\frac{\log p}{n}}.$$

This is what we claimed. ■

Appendix E. LASSO Support Recovery

Proof [Proof of Lemma 3.1.1] Note that since $\mathbf{P}_{\mathbb{X}_{,S_0}^\perp}$ is an orthogonal projection matrix it contracts length and hence:

$$\left\| \mathbf{P}_{\mathbb{X}_{,S_0}^\perp}\left(\frac{\mathbf{w}}{\lambda n} \right) \right\|_2^2 \leq \frac{\|\mathbf{w}\|_2^2}{\lambda^2 n^2}.$$

Next observe that $\mathbf{w} = \boldsymbol{Y} - c_0\mathbf{X}\boldsymbol{\beta}_0$ is a vector with non-zero mean. However, by Chebyshev's inequality we have:

$$\mathbb{P}\left(\left| \frac{\|\mathbf{w}\|_2^2}{n} - \xi^2 \right| \geq t \right) \leq \frac{\theta^2}{nt^2}.$$

Then setting $t = 1$ brings the above probability to 0 at a rate $\frac{\theta^2}{n}$. Next:

$$n^{-1}\check{\mathbf{z}}_{S_0}^\intercal(n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0})^{-1}\check{\mathbf{z}}_{S_0} \leq \frac{1}{\lambda_{\min}(1 - 2\sqrt{\frac{s}{n}})^2}\frac{\|\check{\mathbf{z}}_{S_0}\|_2^2}{n} \leq \frac{1}{\lambda_{\min}(1 - 2\sqrt{\frac{s}{n}})^2}\frac{s}{n},$$

with probability at least $1 - 2\exp(-s/2)$, where we used Lemma A.0.1. This completes the proof. ■

Proof [Proof of Lemma 3.2.2] First, we note the following decomposition:

$$[\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\mathbb{X}_{,S_0}^\intercal\boldsymbol{Y} - c_0\boldsymbol{\beta}_{0S_0} = (n[\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1} - \mathbb{I})n^{-1}\mathbb{X}_{,S_0}^\intercal\boldsymbol{Y} + (n^{-1}\mathbb{X}_{,S_0}^\intercal\boldsymbol{Y} - c_0\boldsymbol{\beta}_{0S_0}).$$

Note that the second term is mean 0. Applying Lemma D.0.1 gives us a bound on the second term. We next move on to consider the first term.

Consider a "symmetrization" transformation of the predictor matrix $\widetilde{\mathbb{X}}_{,S_0}^\intercal = (\mathbb{I} - \boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal)\mathbb{X}_{,S_0}^\intercal + \boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal\mathbb{X}_{,S_0}^{*\intercal}$, where $[\mathbb{X}_{,S_0}^*]_{n\times s}$ is an i.i.d. copy of $\mathbb{X}_{,S_0}$, or in other words the columns of $\mathbb{X}_{,S_0}^*$: $\boldsymbol{X}_j^* \sim \mathcal{N}(0, \mathbb{I}_{n\times n}), j = 1, \ldots, s$ and are independent of $\mathbb{X}_{,S_0}$. Note that in doing this

construction, we guarantee that $\widetilde{\mathbb{X}}_{,S_0}$ is independent of $\mathbb{X}^\intercal_{,S_0}\boldsymbol{\beta}_{0S_0}$. Now we further decompose the first term as follows:

$$\begin{aligned}(n[\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-\mathbb{I})n^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y} &= \underbrace{(n[\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-\mathbb{I})\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0}n^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}}_{\boldsymbol{I}_1}\\ &+\underbrace{n([\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-[\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1})(\mathbb{I}-\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0})n^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}}_{\boldsymbol{I}_2}\\ &+\underbrace{(n[\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1}-\mathbb{I})n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\boldsymbol{Y}}_{\boldsymbol{I}_3}\\ &-\underbrace{(n[\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1}-\mathbb{I})\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0}n^{-1}\mathbb{X}^{*\intercal}_{,S_0}\boldsymbol{Y}}_{\boldsymbol{I}_4}.\end{aligned}$$

We next deal with each of these terms separately. For the first and last terms we can directly apply Lemma A.0.2. Under the same event as in Lemma E.0.1, taking into account that $\|\boldsymbol{\beta}_{0S_0}\|_2=1$ we have that $\|([n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-\mathbb{I})\boldsymbol{\beta}_{0S_0}\|_\infty \leq C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty + C_2\sqrt{\frac{\log p}{n}}$ and $\|([n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1}-\mathbb{I})\boldsymbol{\beta}_{0S_0}\|_\infty \leq C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty + C_2\sqrt{\frac{\log p}{n}}$. Furthermore, $\boldsymbol{\beta}^\intercal_{0S_0}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}/n$ is a mean c_0 random variable. Just as in the proof of Lemma D.0.1 by Chebyshev's inequality we have that with probability at least $1-\frac{\gamma}{\log p}$ we have $|\boldsymbol{\beta}^\intercal_{0S_0}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}/n| \leq |c_0|+\sqrt{\frac{\log p}{n}}$.

Furthermore, notice that $n^{-1}\boldsymbol{\beta}^\intercal_{0S_0}\mathbb{X}^{*\intercal}_{,S_0}\boldsymbol{Y}$ is a mean 0 random variable. Conditionally on $\boldsymbol{Y}$ it has a $\mathcal{N}(0, n^{-2}\sum Y_i^2)$ distribution. With exactly the same argument as in the proof of Lemma D.0.1 we conclude that with probability at least $1-\frac{\eta}{n\sigma^4}-\frac{2}{p}$:

$$|n^{-1}\boldsymbol{\beta}^\intercal_{0S_0}\mathbb{X}^{*\intercal}_{,S_0}\boldsymbol{Y}| \leq 2\sqrt{\sigma^2\frac{\log p}{n}}.$$

Hence, combining the above results we obtain:

$$\|\boldsymbol{I}_1\|_\infty+\|\boldsymbol{I}_4\|_\infty \leq \left(C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty + C_2\sqrt{\frac{\log p}{n}}\right)\left(|c_0|+\sqrt{\frac{\log p}{n}}+2\sqrt{\sigma^2\frac{\log p}{n}}\right). \tag{23}$$

To deal with the term $\boldsymbol{I}_2$ first note that by Hölder's inequality we have:

$$\|\boldsymbol{I}_2\|_\infty \leq \|n([\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-[\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1})\|_{\infty,\infty}\|(\mathbb{I}-\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0})n^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}\|_\infty. \tag{24}$$

To deal with the first term we make usage of the following result:

Lemma E.0.1 *Suppose that s,n satisfy $\frac{s}{n}\leq\frac{1}{16}$. The following bound holds:*

$$\|[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-[n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1}\|_{\infty,\infty} \leq 40\sqrt{s}\left(C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty+\widetilde{C}_2\sqrt{\frac{\log p}{n}}\right),$$

with probability at least $1-4\exp(-s/2)-\frac{12}{p}-\frac{4}{n}$, where $C_1>0$ and $\widetilde{C}_2=C_2+4$ are the same constants as in (23).

Note that the second term is a mean 0 random variable since $(\mathbb{I}-\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0})\mathbb{X}_{,S_0}$ is independent of $\boldsymbol{Y}$. Just as in Lemma D.0.1 we can show that $\|(\mathbb{I}-\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0})n^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}\|_\infty \leq 2\sqrt{2\sigma^2\frac{\log p}{n}}$ with probability at least $1-\frac{2s}{p^2}\geq 1-\frac{2}{p}$ (this event is in fact a sub-event of the bounds of the first term $n^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}-c_0\boldsymbol{\beta}_{0S_0}$). Lemma E.0.1 gives us a bound on $\|n([\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-[\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1})\|_{\infty,\infty}$ which in conjunction with the previous inequality suffices to control the term $\boldsymbol{I}_2$.

Finally, to deal with the term $\boldsymbol{I}_4$ we will make use of the following:

Lemma E.0.2 *Let $\frac{s}{n}\leq\frac{1}{64}$. Then there exists a constant $\Upsilon\asymp\sigma>0$, such that the term:*

$$\|(n[\widetilde{\mathbb{X}}^\intercal_{,S_0}\widetilde{\mathbb{X}}_{,S_0}]^{-1}-\mathbb{I})n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\boldsymbol{Y}\|_\infty \leq \Upsilon\sqrt{\frac{\log p}{n}},$$

with probability at least $1-\frac{2}{p}-\frac{\eta}{n\sigma^4}-2\exp(-s/2)$.

Applying Lemma E.0.2 we have in conjunction with our previous bounds (23) and (24) we get:

$$\begin{aligned}\|[\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}\mathbb{X}^\intercal_{,S_0}\boldsymbol{Y}-c_0\boldsymbol{\beta}_{0S_0}\|_\infty &\leq \left(C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty+C_2\sqrt{\frac{\log p}{n}}\right)\left(|c_0|+\sqrt{\frac{\log p}{n}}+2\sqrt{\sigma^2\frac{\log p}{n}}\right)\\ &+80\sqrt{s}\left(C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty+\widetilde{C}_2\sqrt{\frac{\log p}{n}}\right)\sqrt{2\sigma^2\frac{\log p}{n}}\\ &+\Upsilon\sqrt{\frac{\log p}{n}}+\|\boldsymbol{\beta}_{0S_0}\|_\infty\sqrt{\frac{\log p}{n}}+2\sqrt{\sigma^2\frac{\log p}{n}},\end{aligned}$$

with probability at least $1-4\exp(-s/2)-\frac{4}{n}-\frac{18}{p}-2\frac{\eta}{n\sigma^4}-2\frac{\gamma}{\log p}$[5], which finishes the proof, after grouping terms and recalling the fact that $\log(p-s)\asymp\log p$. ■

Proof [Proof of Lemma E.0.1] We first compare $[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}$ to $[n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0}]^{-1}$. The latter matrix might happen to be non-invertible but this is irrelevant for our proof as we argue below. Using Woodbury's matrix identity we have:

$$[n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0}]^{-1}-[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}=\frac{[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0}\mathbf{M}[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}}{1-\boldsymbol{\beta}^\intercal_{0S_0}\mathbf{M}[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}},$$

where $\mathbf{M}=n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}-\mathbb{I}-n^{-1}\mathbb{X}^{*\intercal}_{,S_0}\mathbb{X}_{,S_0}$. Note that whenever the right hand side of Woodbury's identity is well defined, the matrix $n^{-1}\widetilde{\mathbb{X}}^\intercal_{,S_0}\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}^\intercal_{0S_0}$ is indeed invertible, and the inverse satisfies the above identity. As we argue below the right hand side is well defined (i.e. the denominator is non-zero) with high probability hence the proof goes through. Next we handle the term $\boldsymbol{\beta}^\intercal_{0S_0}\mathbf{M}[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}$. By the triangle inequality have:

$$\|\boldsymbol{\beta}^\intercal_{0S_0}\mathbf{M}[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}\|_\infty \leq \|\boldsymbol{\beta}^\intercal_{0S_0}([n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}-\mathbb{I})\|_\infty+\|\boldsymbol{\beta}^\intercal_{0S_0}n^{-1}\mathbb{X}^{*\intercal}_{,S_0}\mathbb{X}_{,S_0}[n^{-1}\mathbb{X}^\intercal_{,S_0}\mathbb{X}_{,S_0}]^{-1}\|_\infty.$$

5. Here we are recognizing the fact that the events of some probability bounds we derived above, in fact coincide.

For the first term Lemma A.0.2 is directly applicable. Applying this lemma gives us the existence of constants C_1 and C_2 such that:

$$\|([n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}-\mathbb{I})\boldsymbol{\beta}_{0S_0}\|_\infty \leq C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty + C_2\sqrt{\frac{\log p}{n}},$$

with probability at least $1-4p^{-1}$. For the second term, we have that conditionally on $\mathbb{X}_{,S_0}$ it has a normal distribution: $\mathcal{N}(0, n^{-1}(n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0})^{-1})$. Since $\mathbb{X}_{,S_0}$ is standard normal, we can apply Lemma A.0.1 to claim that $\|n(\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0})^{-1}\|_{2,2} \leq \left(\frac{1}{1-\sqrt{\frac{s}{n}}-t}\right)^2$ with probability at least $1-2\exp(-nt^2/2)$. Taking $t=\sqrt{\frac{s}{n}}$ gives us that $\|n(\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0})^{-1}\|_{2,2} \leq \frac{1}{(1-2\sqrt{\frac{s}{n}})^2}$ with probability at least $1-2\exp(-s/2)$. Thus conditioning on this event, by a standard normal tail bound and a union bound we have:

$$\mathbb{P}(\|\boldsymbol{\beta}_{0S_0}^\intercal n^{-1}\mathbb{X}^{*\intercal}_{,S_0}\mathbb{X}_{,S_0}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\|_\infty \geq t) \leq 2s\exp\left(-t^2n\left(1-2\sqrt{\frac{s}{n}}\right)^2/2\right).$$

Selecting $t=4\sqrt{\frac{\log p}{n}}$, we get the probability above is bounded by $\frac{2s}{p^2}\leq\frac{2}{p}$ (where we used the assumption $\sqrt{\frac{s}{n}}\leq\frac{1}{4}$). So finally on the intersection event we have:

$$\|\boldsymbol{\beta}_{0S_0}^\intercal\mathbf{M}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\|_\infty \leq C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty + \widetilde{C}_2\sqrt{\frac{\log p}{n}},$$

with probability at least $1-6p^{-1}-2\exp(-s/2)$ where $\widetilde{C}_2=C_2+4$. Let us now consider the denominator:

$$\begin{aligned}
&1-\boldsymbol{\beta}_{0S_0}^\intercal\mathbf{M}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\\
=&1-\boldsymbol{\beta}_{0S_0}^\intercal(\mathbb{I}-[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1})\boldsymbol{\beta}_{0S_0}+n^{-1}\boldsymbol{\beta}_{0S_0}^\intercal\mathbb{X}^{*\intercal}_{,S_0}\mathbb{X}_{,S_0}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\\
=&\boldsymbol{\beta}_{0S_0}^\intercal[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}+n^{-1}\boldsymbol{\beta}_{0S_0}^\intercal\mathbb{X}^{*\intercal}_{,S_0}\mathbb{X}_{,S_0}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}.
\end{aligned}$$

Using Lemma A.0.1 we have $\lambda_{\min}([n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}) \geq \frac{1}{(1+2\sqrt{\frac{s}{n}})^2} > \frac{1}{4}$ with the last bound holding since $\frac{s}{n}<\frac{1}{4}$. Hence $\boldsymbol{\beta}_{0S_0}^\intercal[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\geq\frac{1}{4}$. For the second term just as before, conditionally on $\mathbb{X}_{,S_0}$ we have

$$n^{-1}\boldsymbol{\beta}_{0S_0}^\intercal\mathbb{X}^{*\intercal}_{,S_0}\mathbb{X}_{,S_0}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0} \sim \mathcal{N}(0, n^{-1}\boldsymbol{\beta}_{0S_0}^\intercal[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}).$$

Then (given that $\|n(\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0})^{-1}\|_{2,2}\leq\frac{1}{(1-2\sqrt{\frac{s}{n}})^2}$) by a standard tail bound we have that the second term is $\leq 4\sqrt{\frac{\log n}{n}}$ with probability at least $1-\frac{2}{n}$. Putting everything together we have:

$$1-\boldsymbol{\beta}_{0S_0}^\intercal\mathbf{M}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\geq\frac{1}{4}-4\sqrt{\frac{\log n}{n}}.$$

The last expression is clearly bigger than $\frac{1}{5}$ for large enough values of n. Hence we conclude that with high probability we have:

$$\begin{aligned}
&\|[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal]^{-1}-[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\|_{\infty,\infty}\\
&\leq 5\|[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\|_1\|\boldsymbol{\beta}_{0S_0}^\intercal\mathbf{M}[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\|_\infty
\end{aligned}$$

For the first term, by the definition of matrix $\|\cdot\|_{2,2}$ norm we further have:

$$\begin{aligned}
\|[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\|_1 &\leq \sqrt{s}\|[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\boldsymbol{\beta}_{0S_0}\|_2 \leq \sqrt{s}\|\boldsymbol{\beta}_{0S_0}\|_2\|[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\|_{2,2}\\
&\leq \frac{\sqrt{s}}{(1-2\sqrt{\frac{s}{n}})^2}.
\end{aligned}$$

Combining this inequality with our previous bound we get:

$$\|[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal]^{-1}-[n^{-1}\mathbb{X}_{,S_0}^\intercal\mathbb{X}_{,S_0}]^{-1}\|_{\infty,\infty} \leq \frac{5\sqrt{s}}{(1-2\sqrt{\frac{s}{n}})^2}\left(C_1\frac{s}{n}\|\boldsymbol{\beta}_{0S_0}\|_\infty+\widetilde{C}_2\sqrt{\frac{\log p}{n}}\right).$$

Next we show that $[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal]^{-1}$ is also close to $[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1}$. Another usage of Woodbury's matrix identity yields:

$$[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\mathbb{X}_{,S_0}+\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal]^{-1}-[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1} = \frac{[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1}\widetilde{\mathbf{M}}\boldsymbol{\beta}_{0S_0}\boldsymbol{\beta}_{0S_0}^\intercal[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1}}{1-\boldsymbol{\beta}_{0S_0}^\intercal[n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1}\widetilde{\mathbf{M}}\boldsymbol{\beta}_{0S_0}},$$

where $\widetilde{\mathbf{M}}=n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}-\mathbb{I}-n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\mathbb{X}_{,S_0}$. Note that since $\widetilde{\mathbb{X}}_{,S_0}^\intercal \perp\!\!\!\perp \mathbb{X}_{,S_0}\boldsymbol{\beta}_{0S_0}$, the same argument as before goes through. Combining the bounds with a triangle inequality completes the proof, using the fact that $\sqrt{\frac{s}{n}}\leq\frac{1}{4}$. ■

Proof [Proof of Lemma E.0.2] We first perform a singular value decomposition on the $\widetilde{\mathbb{X}}_{,S_0}=\mathbf{U}_{n\times s}\mathbf{D}_{s\times s}\mathbf{V}_{s\times s}^\intercal$ matrix. Note that since multiplying $\widetilde{\mathbb{X}}_{,S_0}$ by a unitary $s\times s$ matrix on the right, or by a unitary $n\times n$ matrix on the left doesn't change the distribution of $\widetilde{\mathbb{X}}_{,S_0}$ we conclude that the matrices $\mathbf{U},\mathbf{D}$ and $\mathbf{V}$ are independent. This representation gives us that $(n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0})^{-1}-\mathbb{I}=\mathbf{V}(n\mathbf{D}^{-2}-\mathbb{I})\mathbf{V}^\intercal$. With this notation we can rewrite:

$$(n[\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1}-\mathbb{I})n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\boldsymbol{Y}=\mathbf{V}\underbrace{(n\mathbf{D}^{-2}-\mathbb{I})n^{-1/2}\mathbf{D}}_{\mathbf{W}}n^{-1/2}\mathbf{U}^\intercal\boldsymbol{Y}.$$

We recall that by construction $\widetilde{\mathbb{X}}_{,S_0}$ is independent of $\boldsymbol{Y}$. The elements of the matrix $\mathbf{W}$ can be bounded in a simple manner. We have $\|\mathbf{W}\|_{2,2}\leq\|(n\mathbf{D}^{-2}-\mathbb{I})\|_{2,2}\|n^{-1/2}\mathbf{D}\|_{2,2}$, and by Lemma A.0.1, as before we have: $\|(n\mathbf{D}^{-2}-\mathbb{I})\|_{2,2}\leq\frac{1}{(1-2\sqrt{\frac{s}{n}})^2}-1\leq\frac{4\sqrt{\frac{s}{n}}}{(1-2\sqrt{\frac{s}{n}})^2}$ and $\|n^{-1/2}\mathbf{D}\|_{2,2}\leq 1+2\sqrt{\frac{s}{n}}$ with probability at least $1-2\exp(-s/2)$. We will condition on the event $\|\mathbf{W}\|_{2,2}\leq\frac{4\sqrt{\frac{s}{n}}}{(1-2\sqrt{\frac{s}{n}})^2}(1+2\sqrt{\frac{s}{n}})<9\sqrt{\frac{s}{n}}$, with the last inequality holding for $\sqrt{\frac{s}{n}}\leq\frac{1}{8}$. Since every random variable in the above display is independent from $\mathbf{W}$, the distributions of $\mathbf{V},\mathbf{U}$ and $\boldsymbol{Y}$ stay unchanged under this conditioning. Let $\mathbf{e}_i$ be a unit vector with 1 on the i^{th} position. Since we are interested in bounding the $\|\cdot\|_\infty$ we will start with the following:

$$\mathbf{e}_i^\intercal(n[\widetilde{\mathbb{X}}_{,S_0}^\intercal\widetilde{\mathbb{X}}_{,S_0}]^{-1}-\mathbb{I})n^{-1}\widetilde{\mathbb{X}}_{,S_0}^\intercal\boldsymbol{Y}=\mathbf{v}_i^\intercal\mathbf{W}[n^{-1/2}\mathbf{U}^\intercal\boldsymbol{Y}],$$

where $\mathbf{v}_i^\intercal$ is the i^{th} row of the matrix $\mathbf{V}$. Condition on the vector $n^{-1/2}\mathbf{U}^\intercal\boldsymbol{Y}$. Since $\mathbf{v}_i$ is independent of $n^{-1/2}\mathbf{U}^\intercal\boldsymbol{Y}$ it follows that the distribution of $\mathbf{v}_i$ is uniform on the unit

sphere in $\mathbb{R}^s$. We next show that the function $F(\mathbf{v}_i) = \mathbf{v}_i^\intercal \mathbf{W}[n^{-1/2}\mathbf{U}^\intercal \boldsymbol{Y}]$ is Lipschitz. We have:

$$\|\nabla F\|_2 \leq \|\mathbf{W}\|_{2,2}\|n^{-1/2}\mathbf{U}^\intercal \boldsymbol{Y}\|_2 \leq 9\sqrt{\frac{s}{n}}n^{-1/2}\sqrt{\sum_{i=1}^{s}(\mathbf{u}_i^\intercal \boldsymbol{Y})^2}$$
$$\leq 9\sqrt{\frac{s}{n}}n^{-1/2}\|\boldsymbol{Y}\|_2,$$

where the last inequality follows from the fact that the vectors $\mathbf{u}_i$ are orthonormal and hence $\sum_{i=1}^{s}(\mathbf{u}_i^\intercal \boldsymbol{Y})^2 \leq \|\boldsymbol{Y}\|_2^2$. Since Y_i are assumed to have finite second moment, by Chebyshev's inequality we have that:

$$\mathbb{P}(|n^{-1}\|\boldsymbol{Y}\|_2^2 - \sigma^2| \geq t) \leq \frac{\eta}{nt^2}.$$

Selecting $t = \sigma^2$ is sufficient to keep the above probability going to 0, and furthermore for n large enough guarantees that $n^{-1}\|\boldsymbol{Y}\|_2^2 \leq 2\sigma^2$ and hence $n^{-1/2}\|\boldsymbol{Y}\|_2 \leq \sqrt{2}\sigma$. Thus conditional on this event the function F is Lipschitz with a constant equal to $\sqrt{2}9\sigma\sqrt{\frac{s}{n}}$. Since the expectation of the function F is 0, by concentration of measure for Lipschitz functions on the sphere (Ledoux, 2005; Ledoux and Talagrand, 2013), for any $t > 0$ we have:

$$\mathbb{P}(|F(\mathbf{v}_i)| \geq t\sigma) \leq 2\exp\left(-\widetilde{c}s\frac{t^2}{162\frac{s}{n}}\right),$$

for some absolute constant $\widetilde{c} > 0$. Taking a union bound the above becomes:

$$\mathbb{P}(\max_{i\in[s]}|F(\mathbf{v}_i)| \geq t\sigma) \leq 2s\exp\left(-\widetilde{c}\frac{t^2 n}{162}\right).$$

Selecting $t = 18\sqrt{\frac{\log p}{\widetilde{c}n}}$, keeps the probability vanishing at a rate faster than $2s/p^2 \leq 2/p$ and completes the proof. ■

Proof [Proof of Corollary 2.3.9] Tracing the proof of Theorem 2.3.4 we realize that it suffices to show the following two quantities remain well controlled under the usage of $\widehat{g}$:

i. $|n^{-1}\sum_{i=1}^{n}\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0\widehat{g}(Y_i) - c_0| \leq O(\sqrt{\log p/n})$,

ii. $n^{-1}\sum_{i=1}^{n}\widehat{g}^2(Y_i) = O(1)$,

with probability at least $1-O(p^{-1})$ and $1-O(n^{-1})$ correspondingly. To deal with i. observe that:

$$\left|n^{-1}\sum_{i=1}^{n}\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0\widehat{g}(Y_i) - c_0\right| \leq \left|n^{-1}\sum_{i=1}^{n}\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0 g(Y_i) - c_0\right| + \left|n^{-1}\sum_{i=1}^{n}\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0(g(Y_i) - \widehat{g}(Y_i))\right|$$
$$\leq \underbrace{\left|n^{-1}\sum_{i=1}^{n}\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0 g(Y_i) - c_0\right|}_{I_1}$$
$$+ \underbrace{\left\{n^{-1}\sum_{i=1}^{n}(\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0)^2\right\}^{1/2}\left\{n^{-1}\sum_{i=1}^{n}(\widehat{g}(Y_i) - g(Y_i))^2\right\}^{1/2}}_{I_2}.$$

The term I_1 remains controlled by the proof of Theorem 2.3.4, while for the term I_2 we have:

$$I_2 \leq O(\sqrt{\log p/n}),$$

with probability at least $1 - O(p^{-1})$, where we used the assumption on $\widehat{g}$ and the fact that the random variables $(\boldsymbol{X}_i^\intercal\boldsymbol{\beta}_0)^2 \sim \chi_1^2$ and hence concentrate exponentially about their mean — 1, by a standard tail bound (Boucheron et al., 2013).

Next, for ii., by the triangle inequality we have:

$$\sqrt{n^{-1}\sum_{i=1}^{n}\widehat{g}^2(Y_i)} \leq \sqrt{n^{-1}\sum_{i=1}^{n}g^2(Y_i)} + \sqrt{n^{-1}\sum_{i=1}^{n}(\widehat{g}(Y_i) - g(Y_i))^2}.$$

The first term is well controlled as before and is $O(1)$ with probability at least $1 - O(n^{-1})$ and the second term is at most $O(\sqrt{\log p/n})$ with probability at least $1 - O(p^{-1})$ by assumption which concludes the proof. ■

Proof [Proof of Proposition 2.3.8] First let g be such that $\mathbb{E}\{g(Y)\boldsymbol{X}^\intercal\boldsymbol{\beta}_0\} \neq 0$. Recall that $\mathbb{E}(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0) = 0$. Hence by Cauchy-Schwartz we have:

$$0 < [\mathbb{E}\{g(Y)\boldsymbol{X}^\intercal\boldsymbol{\beta}_0\}]^2 = (\mathbb{E}[g(Y)\mathbb{E}\{\boldsymbol{X}^\intercal\boldsymbol{\beta}_0|Y\}])^2 \leq \mathrm{Var}\{g(Y)\}\,\mathrm{Var}\{\mathbb{E}(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0|Y)\},$$

and therefore $\mathrm{Var}\{\mathbb{E}(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0|Y)\} > 0$. In the reverse case put $g(Y) = \mathbb{E}\{\boldsymbol{X}^\intercal\boldsymbol{\beta}_0|Y\}$ and apply conditional expectation to obtain $\mathbb{E}\{g(Y)\boldsymbol{X}^\intercal\boldsymbol{\beta}_0\} = \mathrm{Var}\{\mathbb{E}(\boldsymbol{X}^\intercal\boldsymbol{\beta}_0|Y)\} > 0$. ■

Appendix F. Additional Simulation Results

F.1 $\mathbf{\Sigma} = \mathbb{I}$

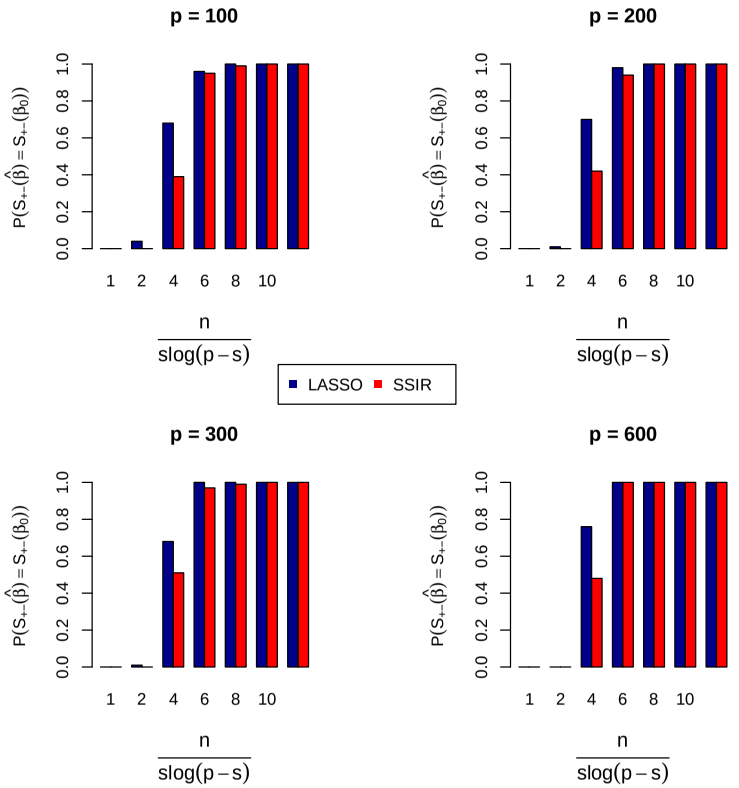


Figure 3: Model (14)

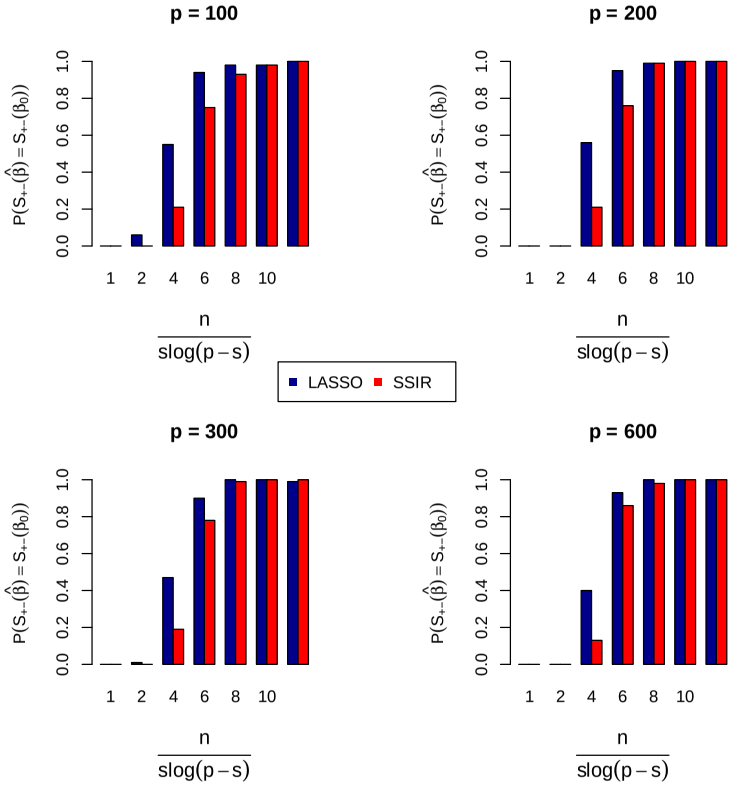


Figure 4: Model (17)

F.2 $\mathbf{\Sigma}$: $\Sigma_{kj} = 2^{-|k-j|}$

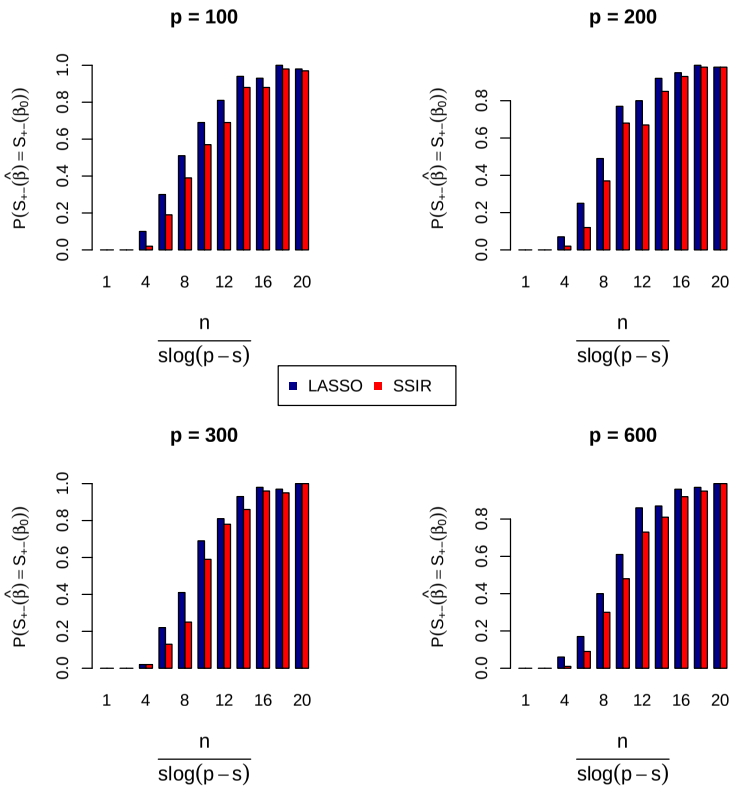


Figure 5: Model (14)

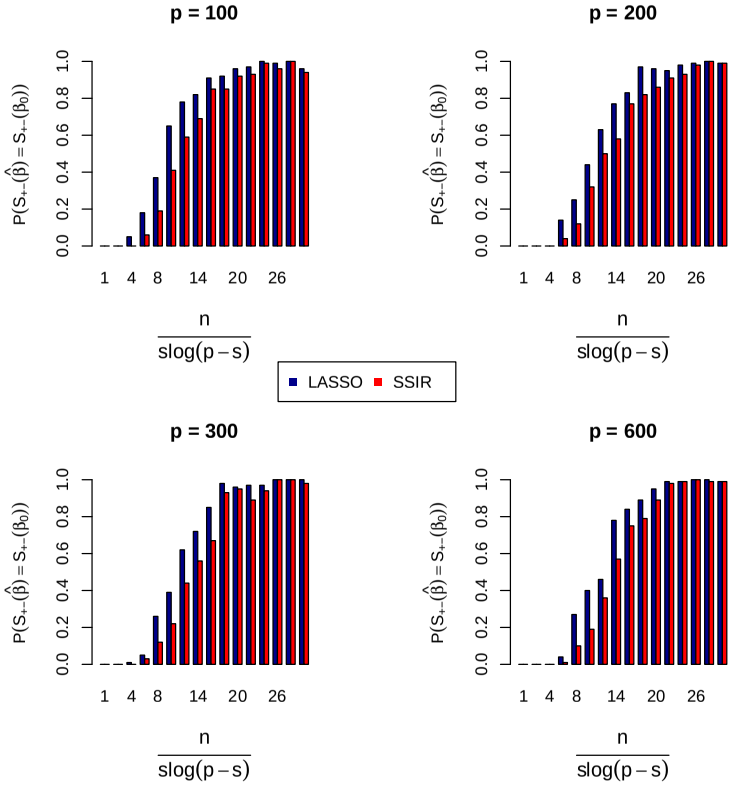


Figure 6: Model (17)

References

Pierre Alquier and Gérard Biau. Sparse single-index model. *The Journal of Machine Learning Research*, 14(1):243–280, 2013.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. OUP Oxford, 2013.

Tony Cai, Weidong Liu, and Xi Luo. A constrained l1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494): 594–607, 2011.

Stamatis Cambanis, Steel Huang, and Gordon Simons. On the theory of elliptically contoured distributions. *Journal of Multivariate Analysis*, 11(3):368–385, 1981.

Emmanuel Candes and Terence Tao. The dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

Rita M Cantor, Kenneth Lange, and Janet S Sinsheimer. Prioritizing gwas results: a review of statistical methods and recommendations for their application. *The American Journal of Human Genetics*, 86(1):6–22, 2010.

R Dennis Cook and Liqiang Ni. Sufficient dimension reduction via inverse regression. *Journal of the American Statistical Association*, 100(470), 2005.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5): 849–911, 2008.

Jianqing Fan, Rui Song, et al. Sure independence screening in generalized linear models with np-dimensionality. *The Annals of Statistics*, 38(6):3567–3604, 2010.

Kai-Tai Fang, Samuel Kotz, and Kai Wang Ng. *Symmetric multivariate and related distributions*. Chapman and Hall, 1990.

Fang Han and Honglang Wang. Provable smoothing approach in high dimensional generalized regression model. *arXiv preprint arXiv:1509.07158*, 2015.

Joel L Horowitz. *Semiparametric and nonparametric methods in econometrics*. Springer, 2009.

Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pages 1302–1338, 2000.

Michel Ledoux. *The concentration of measure phenomenon*. Number 89 in Mathematical Surveys and Monographs. American Mathematical Society, 2005.

Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: isoperimetry and processes*, volume 23. Springer Science & Business Media, 2013.

Jason D Lee, Yuekai Sun, and Jonathan E Taylor. On model selection consistency of m-estimators with geometrically decomposable penalties. *arXiv preprint arXiv:1305.7477*, 2013.

Ker-Chau Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.

Ker-Chau Li and Naihua Duan. Regression analysis under link violation. *The Annals of Statistics*, pages 1009–1052, 1989.

Lexin Li and Christopher J Nachtsheim. Sparse sliced inverse regression. *Technometrics*, 48(4), 2006.

Peter McCullagh and John A Nelder. *Generalized linear models*, volume 37. CRC press, 1989.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, pages 1436–1462, 2006.

Matey Neykov, Qian Lin, and Jun S Liu. Signed support recovery for single index models in high-dimensions. *arXiv preprint, arXiv:1511.02270*, 2015.

Heng Peng and Tao Huang. Penalized least squares for single index models. *Journal of Statistical Planning and Inference*, 141(4):1362–1379, 2011.

Yaniv Plan and Roman Vershynin. The generalized lasso with non-linear observations. *arXiv preprint, arXiv:1502.04071*, 2015.

Peter Radchenko. High dimensional single index models. *Journal of Multivariate Analysis*, 139:266–282, 2015.

Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pages 1135–1151, 1981.

Christos Thrampoulidis, Ehsan Abbasi, and Babak Hassibi. The lasso with non-linear measurements is the lasso with non-linear measurements is equivalent to one with linear measurements. *arXiv preprint, arXiv:1506.02181*, 2015.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (lasso). *IEEE Transactions on Information Theory*, 55(5):2183–2202, 2009.

J Wang, AT Bansal, M Martin, S Germer, R Benayed, L Essioux, JS Lee, A Begovich, A Hemmings, A Kenwright, et al. Genome-wide association analysis implicates the involvement of eight loci with response to tocilizumab for the treatment of rheumatoid arthritis. *The pharmacogenomics journal*, 13(3):235–241, 2013.

Tao Wang and LiXing Zhu. A distribution-based lasso for a general single-index model. *Science China Mathematics*, 58(1):109–130, 2015.

Tao Wang, Pei-Rong Xu, and Li-Xing Zhu. Non-convex penalized estimation in high-dimensional models with single-index structure. *Journal of Multivariate Analysis*, 109: 221–235, 2012.

Yingcun Xia and WK Li. On single-index coefficient regression models. *Journal of the American Statistical Association*, 94(448):1275–1285, 1999.

Zhuoran Yang, Zhaoran Wang, Han Liu, Yonina C. Eldar, and Tong Zhang. Sparse nonlinear regression: Parameter sparse nonlinear regression: Parameter estimation and asymptotic inference. *arXiv preprint arXiv; 1511:04514*, 2015.

Xinyang Yi, Zhaoran Wang, Constantine Caramanis, and Han Liu. Optimal linear estimation under unknown nonlinear transform. *arXiv preprint arXiv:1505.03257*, 2015.

Jingyuan Zhao, Simone Gupta, Mark Seielstad, Jianjun Liu, and Anbupalam Thalamuthu. Pathway-based analysis using reduced gene subsets in genome-wide association studies. *BMC bioinformatics*, 12(1):1, 2011.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

Journal of Machine Learning Research 17 (2016) 1-45 Submitted 1/16; Published 2/16

Spectral Ranking using Seriation

Fajwel Fogel FAJWEL.FOGEL@CMAP.POLYTECHNIQUE.FR
C.M.A.P.
École Polytechnique
Palaiseau, France.

Alexandre d'Aspremont ASPREMON@ENS.FR
CNRS & DI, UMR 8548
École Normale Supérieure
Paris, France.

Milan Vojnovic MILANV@MICROSOFT.COM
Microsoft Research
Cambridge, U.K.

Editor: Aapo Hyvarinen

Abstract

We describe a seriation algorithm for ranking a set of items given pairwise comparisons between these items. Intuitively, the algorithm assigns similar rankings to items that compare similarly with all others. It does so by constructing a similarity matrix from pairwise comparisons, using seriation methods to reorder this matrix and construct a ranking. We first show that this spectral seriation algorithm recovers the true ranking when all pairwise comparisons are observed and consistent with a total order. We then show that ranking reconstruction is still exact when some pairwise comparisons are corrupted or missing, and that seriation based spectral ranking is more robust to noise than classical scoring methods. Finally, we bound the ranking error when only a random subset of the comparions are observed. An additional benefit of the seriation formulation is that it allows us to solve semi-supervised ranking problems. Experiments on both synthetic and real datasets demonstrate that seriation based spectral ranking achieves competitive and in some cases superior performance compared to classical ranking methods.

Keywords: Ranking, seriation, spectral methods

1. Introduction

We study the problem of ranking a set of n items given pairwise comparisons between these items[1]. The problem of aggregating binary relations has been formulated more than two centuries ago, in the context of emerging social sciences and voting theories (de Borda, 1781; de Condorcet, 1785). The setting we study here goes back at least to (Zermelo, 1929; Kendall and Smith, 1940) and seeks to reconstruct a ranking of items from pairwise comparisons reflecting a total ordering. In this case, the directed graph of all pairwise comparisons, where every pair of vertices is connected by exactly one of two possible directed edges, is usually called a *tournament* graph in the theoretical computer science literature or a "round robin" in sports, where every player plays every other player once and each preference marks victory or defeat. The motivation for this formulation often stems from the fact that in many applications, e.g. music, images, and movies, preferences are easier to express in relative terms (e.g. a is better than b) rather than absolute ones (e.g. a should be ranked fourth, and b seventh). In practice, the information about pairwise comparisons is usually *incomplete*, especially in the case of a large set of items, and the data may also be *noisy*, that is some pairwise comparisons could be incorrectly measured and inconsistent with a total order.

Ranking is a classical problem but its formulations vary widely. In particular, assumptions about how the pairwise preference information is obtained vary a lot from one reference to another. A subset of preferences is measured adaptively in (Ailon, 2011; Jamieson and Nowak, 2011), while (Freund et al., 2003; Negahban et al., 2012) extract them at random. In other settings, the full preference matrix is observed, but is perturbed by noise: in e.g. (Bradley and Terry, 1952; Luce, 1959; Herbrich et al., 2006), a parametric model is assumed over the set of permutations, which reformulates ranking as a maximum likelihood problem.

Loss functions, performance metrics and algorithmic approaches vary as well. Kenyon-Mathieu and Schudy (2007), for example, derive a PTAS for the minimum feedback arc set problem on tournaments, i.e. the problem of finding a ranking that minimizes the number of upsets (a pair of players where the player ranked lower on the ranking beats the player ranked higher). In practice, the complexity of this method is relatively high, and other authors (see e.g. Keener, 1993; Negahban et al., 2012) have been using spectral methods to produce more efficient algorithms (each pairwise comparison is understood as a link pointing to the preferred item). In other cases, such as the classical Analytic Hierarchy Process (AHP) (Saaty, 1980; Barbeau, 1986) preference information is encoded in a "reciprocal" matrix whose Perron-Frobenius eigenvector provides the global ranking. Simple scoring methods such as the point difference rule (Huber, 1963; Wauthier et al., 2013) produce efficient estimates at very low computational cost. Website ranking methods such as PageRank (Page et al., 1998) and HITS (Kleinberg, 1999) seek to rank web pages based on the hyperlink structure of the web, where links do not necessarily express consistent preference relationships (e.g. a can link to b and b can link c, and c can link to a). (Negahban et al., 2012) adapt the PageRank argument to the ranking from pairwise comparisons and Vigna (2009) provides a review of ranking algorithms given pairwise comparisons, in particular those involving the estimation of the stationary distribution of a Markov chain. Ranking has also been approached as a prediction problem, i.e. learning to rank (Schapire et al., 1998; Rajkumar and Agarwal, 2014), with (Joachims, 2002) for example using support vector machines to learn a score function. Finally, in the Bradley-Terry-Luce framework, where multiple observations on pairwise preferences are observed and assumed to be generated by a generalized linear model, the maximum likelihood problem is usually solved using fixed point algorithms or EM-like majorization-minimization techniques (Hunter, 2004). Jiang et al. (2011) describes the HodgeRank algorithm, which formulates ranking given pairwise comparisons as a least-square problem. This formulation is based on Hodge theory and provides tools to measure the consistency of a set of pairwise comparisons with the existence of a global ranking. Duchi et al. (2010, 2013) analyze the consistency of various ranking algorithms given pairwise comparisons and a query. Preferences are aggregated through standard procedures, e.g., computing the mean of comparisons from different users, then

1. A subset of these results appeared at NIPS 2014.

ranking are derived using classical algorithms, e.g., Borda Count, Bradley-Terry-Model maximum likelihood estimation, least squares, odd-ratios (Saaty, 2003).

Here, we show that the ranking problem is directly related to another classical ordering problem, namely *seriation*. Given a similarity matrix between a set of n items and assuming that the items can be ordered along a chain (path) such that the similarity between items decreases with their distance within this chain (i.e. a total order exists), the seriation problem seeks to reconstruct the underlying linear ordering based on unsorted, possibly noisy, pairwise similarity information. Atkins et al. (1998) produced a spectral algorithm that exactly solves the seriation problem in the noiseless case, by showing that for similarity matrices computed from serial variables, the ordering of the eigenvector corresponding to the second smallest eigenvalue of the Laplacian matrix (a.k.a. the Fiedler vector) matches that of the variables. In practice, this means that performing spectral ordering on the similarity matrix exactly reconstructs the correct ordering provided items are organized in a chain.

We adapt these results to ranking to produce a very efficient *spectral ranking algorithm with provable recovery and robustness guarantees*. Furthermore, the seriation formulation allows us to handle semi-supervised ranking problems. Fogel et al. (2013) show that seriation is equivalent to the 2-SUM problem and study convex relaxations to seriation in a semi-supervised setting, where additional structural constraints are imposed on the solution. Several authors (Blum et al., 2000; Feige and Lee, 2007) have also focused on the directly related Minimum Linear Arrangement (MLA) problem, for which excellent approximation guarantees exist in the noisy case, albeit with very high polynomial complexity.

The main contributions of this paper can be summarized as follows. We link seriation and ranking by showing how to construct a consistent similarity matrix based on consistent pairwise comparisons. We then recover the true ranking by applying the spectral seriation algorithm in (Atkins et al., 1998) to this similarity matrix (we call this method SerialRank in what follows). In the noisy case, we then show that spectral seriation can perfectly recover the true ranking even when some of the pairwise comparisons are either corrupted or missing, provided that the pattern of errors is somewhat unstructured. We show in particular that, in a regime where a high proportion of comparisons are observed, some incorrectly, the spectral solution is more robust to noise than classical scoring based methods. On the other hand, when only few comparisons are observed, we show that for Erdös-Rényi graphs, i.e., when pairwise comparisons are observed independently with a given probability, $\Omega(n \log^4 n)$ comparisons suffice for ℓ_2 consistency of the Fiedler vector and hence ℓ_2 consistency of the retreived ranking w.h.p. On the other hand we need $\Omega(n^{3/2} \log^4 n)$ comparisons to retrieve a ranking whose local perturbations are bounded in ℓ_∞ norm. Since for Erdös-Rényi graphs the induced graph of comparisons is connected with high probability only when the total number of pairs sampled scales as $\Omega(n \log n)$ (aka the coupon collector effect), we need at least that many comparisons in order to retrieve a ranking, therefore the ℓ_2 consistency result can be seen as optimal up to a polylogarithmic factor. Finally, we use the seriation results in (Fogel et al., 2013) to produce semi-supervised ranking solutions.

The paper is organized as follows. In Section 2 we recall definitions related to seriation, and link ranking and seriation by showing how to construct well ordered similarity matrices from well ranked items. In Section 3 we apply the spectral algorithm of (Atkins et al., 1998) to reorder these similarity matrices and reconstruct the true ranking in the noiseless case.

In Section 4 we then show that this spectral solution remains exact in a noisy regime where a random subset of comparisons is corrupted. In Section 5 we analyze ranking perturbation results when only few comparisons are given following an Erdös-Rényi graph. Finally, in Section 6 we illustrate our results on both synthetic and real datasets, and compare ranking performance with classical MLE, spectral and scoring based approaches.

2. Seriation, Similarities & Ranking

In this section we first introduce the seriation problem, i.e. reordering items based on pairwise similarities. We then show how to write the problem of ranking given pairwise comparisons as a seriation problem.

2.1 The Seriation Problem

The seriation problem seeks to reorder n items given a similarity matrix between these items, such that the more similar two items are, the closer they should be. This is equivalent to supposing that items can be placed on a chain where the similarity between two items decreases with the distance between these items in the chain. We formalize this below, following (Atkins et al., 1998).

Definition 1 *We say that a matrix $A \in \mathbf{S}_n$ is an R-matrix (or Robinson matrix) if and only if it is symmetric and $A_{i,j} \leq A_{i,j+1}$ and $A_{i+1,j} \leq A_{i,j}$ in the lower triangle, where $1 \leq j < i \leq n$.*

Another way to formulate R-matrix conditions is to impose $A_{ij} \geq A_{kl}$ if $|i-j| \leq |k-l|$ off-diagonal, i.e. the coefficients of A decrease as we move away from the diagonal. We also introduce a definition for strict R-matrices A, whose rows and columns cannot be permuted without breaking the R-matrix monotonicity conditions. We call *reverse identity* permutation the permutation that puts rows and columns $1, 2, \ldots, n$ of a matrix A in reverse order $n, n-1, \ldots, 1$.

Definition 2 *An R-matrix $A \in \mathbf{S}_n$ is called strict-R if and only if the identity and reverse identity permutations of A are the only permutations reordering A as an R-matrix.*

Any R-matrix with only strict R-constraints is a strict R-matrix. Following (Atkins et al., 1998), we will say that A is *pre-R* if there is a permutation matrix Π such that $\Pi A \Pi^T$ is an R-matrix. Given a pre-R matrix A, the seriation problem consists in finding a permutation Π such that $\Pi A \Pi^T$ is an R-matrix. Note that there might be several solutions to this problem. In particular, if a permutation Π is a solution, then the reverse permutation is also a solution. When only two permutations of A produce R-matrices, A will be called *pre-strict-R*.

2.2 Constructing Similarity Matrices from Pairwise Comparisons

Given an ordered input pairwise comparison matrix, we now show how to construct a similarity matrix which is *strict-R* when all comparisons are given and consistent with the identity ranking (i.e., items are ranked in increasing order of indices). This means that the

similarity between two items decreases with the distance between their ranks. We will then be able to use the spectral seriation algorithm by (Atkins et al., 1998) described in Section 3 to reconstruct the true ranking from a disordered similarity matrix.

We first show how to compute a pairwise similarity from pairwise comparisons between items by counting the number of matching comparisons. Another formulation allows us to handle the generalized linear model. These two examples are only two particular instances of a broader class of ranking algorithms derived here. Any method which produces R-matrices from pairwise preferences yields a valid ranking algorithm.

2.2.1 Similarities from Pairwise Comparisons

Suppose we are given a matrix of pairwise comparisons $C \in \{-1,0,1\}^{n\times n}$ such that $C_{i,j} = -C_{j,i}$ for every $i \neq j$ and

$$C_{i,j} = \begin{cases} 1 & \text{if } i \text{ is ranked higher than } j \\ 0 & \text{if } i \text{ and } j \text{ are not compared or in a draw} \\ -1 & \text{if } j \text{ is ranked higher than } i \end{cases} \tag{1}$$

setting $C_{i,i} = 1$ for all $i \in \{1,\dots,n\}$. We define the pairwise similarity matrix S^{match} as

$$S^{\text{match}}_{i,j} = \sum_{k=1}^{n} \left(\frac{1 + C_{i,k}C_{j,k}}{2} \right). \tag{2}$$

Since $C_{i,k}C_{j,k} = 1$, if $C_{i,k}$ and $C_{j,k}$ have matching signs, and $C_{i,k}C_{j,k} = -1$ if they have opposite signs, $S^{\text{match}}_{i,j}$ counts the number of matching comparisons between i and j with other reference items k. If i or j is not compared with k, then $C_{i,k}C_{j,k} = 0$ and the term $(1 + C_{i,k}C_{j,k})/2$ has an neutral effect on the similarity of $1/2$. Note that we also have

$$S^{\text{match}} = \frac{1}{2}\left(n\mathbf{1}\mathbf{1}^T + CC^T \right). \tag{3}$$

The intuition behind the similarity S^{match} is easy to understand in a tournament setting: players that beat the same players and are beaten by the same players should have a similar ranking.

The next result shows that when all comparisons are given and consistent with the identity ranking, then the similarity matrix S^{match} is a strict R-matrix. Without loss of generality, we assume that items are ranked in increasing order of their indices. In the general case, we can simply replace the *strict-R* property by the *pre-strict-R* property.

Proposition 3 *Given all pairwise comparisons between items ranked according to the identity permutation (with no ties), the similarity matrix S^{match} constructed in (2) is a strict R-matrix and*

$$S^{\text{match}}_{i,j} = n - |i - j| \tag{4}$$

for all $i, j = 1,\dots,n$.

Proof Since items are ranked as $1, 2, \dots, n$ with no ties and all comparisons given, $C_{i,j} = -1$ if $i < j$ and $C_{i,j} = 1$ otherwise. Therefore we obtain from definition (2)

$$\begin{aligned} S^{\text{match}}_{i,j} &= \sum_{k=1}^{\min(i,j)-1} \left(\frac{1+1}{2}\right) + \sum_{k=\min(i,j)}^{\max(i,j)-1} \left(\frac{1-1}{2}\right) + \sum_{k=\max(i,j)}^{n} \left(\frac{1+1}{2}\right) \\ &= n - (\max(i,j) - \min(i,j)) \\ &= n - |i-j| \end{aligned}$$

This means in particular that S^{match} is strictly positive and its coefficients are strictly decreasing when moving away from the diagonal, hence S^{match} is a strict R-matrix. ■

2.2.2 Similarities in the Generalized Linear Model

Suppose that paired comparisons are generated according to a generalized linear model (GLM), i.e., we assume that the outcomes of paired comparisons are independent and for any pair of distinct items, item i is observed ranked higher than item j with probability

$$P_{i,j} = H(\nu_i - \nu_j) \tag{5}$$

where $\nu \in \mathbb{R}^n$ is a vector of skill parameters and $H : \mathbb{R} \to [0,1]$ is a function that is increasing on $\mathbb{R}$ and such that $H(-x) = 1 - H(x)$ for all $x \in \mathbb{R}$, and $\lim_{x\to-\infty} H(x) = 0$ and $\lim_{x\to\infty} H(x) = 1$. A well known special instance of the generalized linear model is the Bradley-Terry-Luce model for which $H(x) = 1/(1+e^{-x})$, for $x \in \mathbb{R}$.

Let $m_{i,j}$ be the number of times items i and j were compared, $C^s_{i,j} \in \{-1,1\}$ be the outcome of comparison s and Q be the matrix of corresponding sample probabilities, i.e. if $m_{i,j} > 0$ we have

$$Q_{i,j} = \frac{1}{m_{i,j}} \sum_{s=1}^{m_{i,j}} \frac{C^s_{i,j} + 1}{2}$$

and $Q_{i,j} = 1/2$ in case $m_{i,j} = 0$. We define the similarity matrix S^{glm} from the observations Q as

$$S^{\text{glm}}_{i,j} = \sum_{k=1}^{n} \mathbf{1}_{\{m_{i,k}m_{j,k}>0\}} \left(1 - |Q_{i,k} - Q_{j,k}|\right) + \frac{\mathbf{1}_{\{m_{i,k}m_{j,k}=0\}}}{2}. \tag{6}$$

Since the comparison observations are independent we have that $Q_{i,j}$ converges to $P_{i,j}$ as $m_{i,j}$ goes to infinity and the central limit theorem implies that $S^{\text{glm}}_{i,j}$ converges to a Gaussian variable with mean

$$\sum_{k=1}^{n} \left(1 - |P_{i,k} - P_{j,k}|\right).$$

The result below shows that this limit similarity matrix is a strict R-matrix when items are properly ordered.

Proposition 4 *If items are ordered according to the order in decreasing values of the skill parameters, the similarity matrix S^{glm} is a strict R matrix with high probability as the number of observations goes to infinity.*

Proof Without loss of generality, we suppose the true order is $1, 2, \ldots, n$, with $\nu(1) > \ldots > \nu(n)$. For any i, j, k such that $i > j$, using the GLM assumption (i) we get

$$P_{i,k} = H(\nu(i) - \nu(k)) < H(\nu(j) - \nu(k)) = P_{j,k}.$$

Since empirical probabilities $Q_{i,j}$ converge to $P_{i,j}$, when the number of observations is large enough, we also have $Q_{i,k} < Q_{j,k}$ for any i, j, k such that $i > j$ (we focus w.l.o.g. on the lower triangle), and we can therefore remove the absolute value in the expression of $S^{\text{glm}}_{i,j}$ for $i > j$. Hence for any $i > j$ we have

$$\begin{aligned} S^{\text{glm}}_{i+1,j} - S^{\text{glm}}_{i,j} &= -\sum_{k=1}^{n} |Q_{i+1,k} - Q_{j,k}| + \sum_{k=1}^{n} |Q_{i,k} - Q_{j,k}| \\ &= \sum_{k=1}^{n} (Q_{i+1,k} - Q_{j,k}) - (Q_{i,k} - Q_{j,k}) \\ &= \sum_{k=1}^{n} Q_{i+1,k} - Q_{i,k} < 0. \end{aligned}$$

Similarly for any $i > j$, $S^{\text{glm}}_{i,j-1} - S^{\text{glm}}_{i,j} < 0$, so S^{glm} is a strict R-matrix. ■

Notice that we recover the original definition of S^{match} in the case of binary comparisons, though it does not fit in the Generalized Linear Model. Note also that these definitions can be directly extended to the setting where multiple comparisons are available for each pair and aggregated in comparisons that take fractional values (e.g., a tournament setting where participants play several times against each other).

3. Spectral Algorithms

We first recall how spectral ordering can be used to recover the true ordering in seriation problems. We then apply this method to the ranking problem.

3.1 Spectral Seriation Algorithm

We use the spectral computation method originally introduced in (Atkins et al., 1998) to solve the seriation problem based on the similarity matrices defined in the previous section. We first recall the definition of the Fiedler vector (which is shown to be unique in our setting in Lemma 7).

Definition 5 *The Fiedler value of a symmetric, nonnegative and irreducible matrix A is the smallest non-zero eigenvalue of its Laplacian matrix $L_A = \mathbf{diag}(A\mathbf{1}) - A$. The corresponding eigenvector is called Fiedler vector and is the optimal solution to* $\min\{y^T L_A y : y \in \mathbb{R}^n, y^T \mathbf{1} = 0, \|y\|_2 = 1\}$.

The main result from (Atkins et al., 1998), detailed below, shows how to reorder pre-R matrices in a noise free case.

Proposition 6 *(Atkins et al., 1998, Th. 3.3) Let $A \in \mathbf{S}_n$ be an irreducible pre-R-matrix with a simple Fiedler value and a Fiedler vector v with no repeated values. Let $\Pi_1 \in \mathcal{P}$ (respectively, Π_2) be the permutation such that the permuted Fiedler vector $\Pi_1 v$ is strictly increasing (decreasing). Then $\Pi_1 A \Pi_1^T$ and $\Pi_2 A \Pi_2^T$ are R-matrices, and no other permutations of A produce R-matrices.*

The next technical lemmas extend the results in Atkins et al. (1998) to strict R-matrices and will be used to prove Theorem 10 in next section. The first one shows that without loss of generality, the Fiedler value is simple.

Lemma 7 *If A is an irreducible R-matrix, up to a uniform shift of its coefficients, A has a simple Fiedler value and a monotonic Fiedler vector.*

Proof We use (Atkins et al., 1998, Th. 4.6) which states that if A is an irreducible R-matrix with $A_{n,1} = 0$, then the Fiedler value of A is a simple eigenvalue. Since A is an R-matrix, $A_{n,1}$ is among its minimal elements. Subtracting it from A does not affect the nonnegativity of A and we can apply (Atkins et al., 1998, Th. 4.6). Monotonicity of the Fiedler vector then follows from (Atkins et al., 1998, Th. 3.2). ■

The next lemma shows that the Fiedler vector is strictly monotonic if A is a strict R-matrix.

Lemma 8 *Let $A \in \mathbf{S}_n$ be an irreducible R-matrix. Suppose there are no distinct indices $r < s$ such that for any $k \notin [r, s]$, $A_{r,k} = A_{r+1,k} = \ldots = A_{s,k}$, then, up to a uniform shift, the Fiedler value of A is simple and its Fiedler vector is strictly monotonic.*

Proof By Lemma 7, the Fiedler value of A is simple (up to a uniform shift of A). Let x be the corresponding Fiedler vector of A, x is monotonic by Lemma 7. Suppose $[r, s]$ is a nontrivial maximal interval such that $x_r = x_{r+1} = \ldots = x_s$, then by (Atkins et al., 1998, lemma 4.3), for any $k \notin [r, s]$, $A_{r,k} = A_{r+1,k} = \ldots = A_{s,k}$, which contradicts the initial assumption. Therefore x is strictly monotonic. ■

In fact, we only need a small portion of the R-constraints to be strict for the previous lemma to hold. We now show that the main assumption on A in Lemma 8 is equivalent to A being strict-R.

Lemma 9 *An irreducible R-matrix $A \in \mathbf{S}_n$ is strictly R if and only if there are no distinct indices $r < s$ such that for any $k \notin [r, s]$, $A_{r,k} = A_{r+1,k} = \ldots = A_{s,k}$.*

Proof Let $A \in \mathbf{S}_n$ an R-matrix. Let us first suppose there are no distinct indices $r < s$ such that for any $k \notin [r, s]$, $A_{r,k} = A_{r+1,k} = \ldots = A_{s,k}$. By Lemma 8 the Fiedler value of A is simple and its Fiedler vector is strictly monotonic. Hence by Proposition 6, only the identity and reverse identity permutations of A produce R-matrices. Now suppose there exist two distinct indices $r < s$ such that for any $k \notin [r, s]$, $A_{r,k} = A_{r+1,k} = \ldots = A_{s,k}$. In addition to the identity and reverse identity permutations, we can locally reverse the order of rows and columns from r to s, since the sub matrix $A_{r:s,r:s}$ is an R-matrix and for any

Algorithm 1 (SerialRank)

Input: A set of pairwise comparisons $C_{i,j} \in \{-1, 0, 1\}$ or $[-1, 1]$.

1: Compute a similarity matrix S as in §2.2

2: Compute the Laplacian matrix

$$L_S = \mathbf{diag}(S\mathbf{1}) - S \qquad \text{(SerialRank)}$$

3: Compute the Fiedler vector of S.

Output: A ranking induced by sorting the Fiedler vector of S (choose either increasing or decreasing order to minimize the number of upsets).

$k \notin [r, s], A_{r,k} = A_{r+1,k} = \ldots = A_{s,k}$. Therefore at least four different permutations of A produce R-matrices, which means that A is not strictly R. ■

3.2 SerialRank: a Spectral Ranking Algorithm

In Section 2, we showed that similarities S^{match} and S^{glm} are *pre-strict-R* when all comparisons are available and consistent with an underlying ranking of items. We now use the spectral seriation method in (Atkins et al., 1998) to reorder these matrices and produce a ranking. Spectral ordering requires computing an extremal eigenvector, at a cost of $O(n^2 \log n)$ flops (Kuczynski and Wozniakowski, 1992). We call this algorithm SerialRank and prove the following result.

Theorem 10 *Given all pairwise comparisons for a set of totally ordered items and assuming there are no ties between items, algorithm SerialRank, i.e., sorting the Fiedler vector of the matrix* S^{match} *defined in* (3)*, recovers the true ranking of items.*

Proof From Proposition 3, under assumptions of the proposition S^{match} is a pre-strict R-matrix. Now combining the definition of strict-R matrices in Lemma 9 with Lemma 8, we deduce that Fiedler value of S^{match} is simple and its Fiedler vector has no repeated values. Hence by Proposition 6, only the two permutations that sort the Fiedler vector in increasing and decreasing order produce strict R-matrices and are candidate rankings (by Proposition 3 S^{match} is a strict R-matrix when ordered according to the true ranking). Finally we can choose between the two candidate rankings (increasing and decreasing) by picking the one with the least upsets. ■

Similar results apply for S^{glm} given enough comparisons in the Generalized Linear Model. This last result guarantees recovery of the true ranking of items in the noiseless case. In the next section, we will study the impact of corrupted or missing comparisons on the inferred ranking of items.

4. Exact Recovery with Corrupted and Missing Comparisons

In this section we study the robustness of SerialRank using S^{match} with respect to noisy and missing pairwise comparisons. We will see that noisy comparisons cause ranking ambiguities for the point score method and that such ambiguities are to be lifted by the spectral ranking algorithm. We show in particular that the SerialRank algorithm recovers the exact ranking when the pattern of errors is random and errors are not too numerous. We first study the impact of one corrupted comparison on SerialRank, then extend the result to multiple corrupted comparisons. A similar analysis is provided for missing comparisons as Corollary 27. in the Appendix. Finally, Proposition 14 provides an estimate of the number of randomly corrupted entries that can be tolerated for perfect recovery of the true ranking. We begin by recalling the definition of the *point score* of an item.

Definition 11 *The* point score w_i *of an item* i*, also known as point-difference, or* row-sum *is defined as* $w_i = \sum_{k=1}^{n} C_{k,i}$*, which corresponds to the number of wins minus the number of losses in a tournament setting.*

In the following we will denote by w the point score vector.

Proposition 12 *Given all pairwise comparisons* $C_{s,t} \in \{-1, 1\}$ *between items ranked according to their indices, suppose the sign of one comparison* $C_{i,j}$ *(and its counterpart* $C_{j,i}$*) is switched, with* $i < j$*. If* $j - i > 2$ *then* S^{match} *defined in* (3) *remains* strict-R*, whereas the point score vector* w *has ties between items* i *and* $i+1$ *and items* j *and* $j-1$*.*

Proof We give some intuition for the result in Figure 1. We write the true score and comparison matrix w and C, while the observations are written $\hat{w}$ and $\hat{C}$ respectively. This means in particular that $\hat{C}_{i,j} = -C_{i,j} = 1$ and $\hat{C}_{j,i} = -C_{j,i} = -1$. To simplify notations we denote by S the similarity matrix S^{match} (respectively $\hat{S}$ when the similarity is computed from observations). We first study the impact of a corrupted comparison $C_{i,j}$ for $i < j$ on the point score vector $\hat{w}$. We have

$$\hat{w}_i = \sum_{k=1}^{n} \hat{C}_{k,i} = \sum_{k=1}^{n} C_{k,i} + \hat{C}_{j,i} - C_{j,i} = w_i - 2 = w_{i+1},$$

similarly $\hat{w}_j = w_{j-1}$, whereas $\hat{w}_k = w_k$ for $k \neq i, j$. Hence, the incorrect comparison induces two ties in the point score vector w. Now we show that the similarity matrix defined in (3) breaks these ties, by showing that it is a strict R-matrix. Writing $\hat{S}$ in terms of S, we get for any $t \neq i, j$

$$[\hat{C}\hat{C}^T]_{i,t} = \sum_{k \neq j} \left(\hat{C}_{i,k}\hat{C}_{t,k}\right) + \hat{C}_{i,j}\hat{C}_{t,j} = \sum_{k \neq j} (C_{i,k}C_{t,k}) + \hat{C}_{i,j}C_{t,j} = \begin{cases} [CC^T]_{i,t} - 2 & \text{if } t < j \\ [CC^T]_{i,t} + 2 & \text{if } t > j. \end{cases}$$

Thus we obtain

$$\hat{S}_{i,t} = \begin{cases} S_{i,t} - 1 & \text{if } t < j \\ S_{i,t} + 1 & \text{if } t > j, \end{cases}$$

(remember there is a factor $1/2$ in the definition of S). Similarly we get for any $t \neq i, j$

$$\hat{S}_{j,t} = \begin{cases} S_{j,t} + 1 & \text{if } t < i \\ S_{j,t} - 1 & \text{if } t > i. \end{cases}$$

Finally, for the single corrupted index pair (i,j), we get

$$\hat{S}_{i,j} = \frac{1}{2}\left(n + \sum_{k\neq i,j}\left(\hat{C}_{i,k}\hat{C}_{j,k}\right) + \hat{C}_{i,i}\hat{C}_{j,i} + \hat{C}_{i,j}\hat{C}_{j,j}\right) = S_{i,j} - 1 + 1 = S_{i,j}.$$

The diagonal of S is not impacted since $[\hat{C}\hat{C}^T]_{i,i} = \sum_{k=1}^{n}\left(\hat{C}_{i,k}\hat{C}_{i,k}\right) = n$. For all other coefficients (s,t) such that $s,t \neq i,j$, we also have $\hat{S}_{s,t} = S_{s,t}$, which means that all rows or columns outside of i,j are left unchanged. We first observe that these last equations, together with our assumption that $j-i>2$ and the fact that the elements of the exact S in (4) differ by at least one, imply that

$$\hat{S}_{s,t} \leq \hat{S}_{s+1,t} \quad \text{and} \quad \hat{S}_{s,t+1} \leq \hat{S}_{s,t}, \quad \text{for } s<t$$

so $\hat{S}$ remains an R-matrix. Note that this result remains true even when $j-i=2$, but we need some strict inequalities to show uniqueness of the retrieved order. Indeed, because $j-i>2$ all these R constraints are strict except between elements of rows i and $i+1$, and rows $j-1$ and j (and similarly for columns). These ties can be broken using the fact that

$$\hat{S}_{i,j-1} = S_{i,j-1} - 1 < S_{i+1,j-1} - 1 = \hat{S}_{i+1,j-1} - 1 < \hat{S}_{i+1,j-1}$$

which means that $\hat{S}$ is still a strict R-matrix (see Figure 1) since $j-1>i+1$ by assumption. ■

We now extend this result to multiple errors.

Proposition 13 *Given all pairwise comparisons $C_{s,t} \in \{-1,1\}$ between items ranked according to their indices, suppose the signs of m comparisons indexed $(i_1,j_1),\ldots,(i_m,j_m)$ are switched. If the following condition* (7) *holds true,*

$$|s-t|>2, \text{ for all } s,t \in \{i_1,\ldots,i_m,j_1,\ldots,j_m\} \text{ with } s\neq t, \tag{7}$$

then S^{match} defined in (3) *remains* strict-R*, whereas the point score vector w has $2m$ ties.*

Proof We write the true score and comparison matrix w and C, while the observations are written $\hat{w}$ and $\hat{C}$ respectively, and without loss of generality we suppose $i_l<j_l$. This implies that $\hat{C}_{i_l,j_l} = -C_{i_l,j_l} = 1$ and $\hat{C}_{j_l,i_l} = -C_{j_l,i_l} = -1$ for all l in $\{1,\ldots,m\}$. To simplify notations, we denote by S the similarity matrix S^{match} (respectively $\hat{S}$ when the similarity is computed from observations).

As in the proof of Proposition 12, corrupted comparisons indexed (i_l,j_l) induce shifts of ±1 on columns and rows i_l and j_l of the similarity matrix S^{match}, while $S^{\text{match}}_{i_l,j_l}$ values remain the same. Since there are several corrupted comparisons, we also need to check the values of $\hat{S}$ at the intersections of rows and columns with indices of corrupted comparisons. Formally, for any $(i,j) \in \{(i_1,j_1),\ldots(i_m,j_m)\}$ and $t \notin \{i_1,\ldots,i_m,j_1,\ldots,j_m\}$

$$\hat{S}_{i,t} = \begin{cases} S_{i,t}+1 & \text{if } t<j \\ S_{i,t}-1 & \text{if } t>j, \end{cases}$$

Similarly for $t \notin \{i_1,\ldots,i_m,j_1,\ldots,j_m\}$

$$\hat{S}_{j,t} = \begin{cases} S_{j,t}-1 & \text{if } t<i \\ S_{j,t}+1 & \text{if } t>i. \end{cases}$$

Let (s,s') and $(t,t') \in \{(i_1,j_1),\ldots(i_m,j_m)\}$, we have

$$\begin{aligned} \hat{S}_{s,t} &= \tfrac{1}{2}\left(n + \textstyle\sum_{k\neq s',t'}\left(\hat{C}_{s,k}\hat{C}_{t,k}\right) + \hat{C}_{s,s'}\hat{C}_{t,s'} + \hat{C}_{s,t'}\hat{C}_{t,t'}\right) \\ &= \tfrac{1}{2}\left(n + \textstyle\sum_{k\neq s',t'}\left(C_{s,k}C_{t,k}\right) - C_{s,s'}C_{t,s'} - C_{s,t'}C_{t,t'}\right) \end{aligned}$$

Without loss of generality we suppose $s<t$, and since $s<s'$ and $t<t'$, we obtain

$$\hat{S}_{s,t} = \begin{cases} S_{s,t} & \text{if } t>s' \\ S_{s,t}+2 & \text{if } t<s'. \end{cases}$$

Similar results apply for other intersections of rows and columns with indices of corrupted comparisons (i.e., shifts of 0, +2, or −2). For all other coefficients (s,t) such that $s,t \notin \{i_1,\ldots,i_m,j_1,\ldots,j_m\}$, we have $\hat{S}_{s,t} = S_{s,t}$. We first observe that these last equations, together with our assumption that $j_l - i_l > 2$, mean that

$$\hat{S}_{s,t} \leq \hat{S}_{s+1,t} \quad \text{and} \quad \hat{S}_{s,t+1} \leq \hat{S}_{s,t}, \quad \text{for any } s<t$$

so $\hat{S}$ remains an R-matrix. Moreover, since $j_l - i_l > 2$ all these R constraints are strict except between elements of rows i_l and i_l+1, and rows j_l-1 and j_l (similar for columns). These ties can be broken using the fact that for $k = j_l - 1$

$$\hat{S}_{i_l,k} = S_{i_l,k} - 1 < S_{i_l+1,k} - 1 = \hat{S}_{i_l+1,k} - 1 < \hat{S}_{i_l+1,k}$$

which means that $\hat{S}$ is still a strict R-matrix since $k = j_l - 1 > i_l + 1$. Moreover, using the same argument as in the proof of Proposition 12, corrupted comparisons induces $2m$ ties in the point score vector w. ■

For the case of one corrupted comparison, note that the separation condition on the pair of items (i,j) is necessary. When the comparison $C_{i,j}$ between two adjacent items is corrupted, no ranking method can break the resulting tie. For the case of arbitrary number of corrupted comparisons, condition (7) is a sufficient condition only. We study exact ranking recovery conditions with missing comparisons in the Appendix, using similar arguments. We now estimate the number of randomly corrupted entries that can be tolerated while maintaining exact recovery of the true ranking.

Proposition 14 *Given a comparison matrix for a set of n items with m corrupted comparisons selected uniformly at random from the set of all possible item pairs. Algorithm SerialRank guarantees that the probability of recovery $p(n,m)$ satisfies $p(n,m) \geq 1-\delta$, provided that $m = O(\sqrt{\delta n})$. In particular, this implies that $p(n,m) = 1 - o(1)$ provided that $m = o(\sqrt{n})$.*

Proof Let $\mathcal{P}$ be the set of all distinct pairs of items from the set $\{1, 2, \dots, n\}$. Let $\mathcal{X}$ be the set of all admissible sets of pairs of items, i.e. containing each $X \subseteq \mathcal{P}$ such that X satisfies condition (7). We consider the case of $m \geq 1$ distinct pairs of items sampled from the set $\mathcal{P}$ uniformly at random without replacement. Let X_i denote the set of sampled pairs given that i pairs are sampled. We seek to bound $p(n, m) = \mathbf{Prob}(X_m \in \mathcal{X})$. Given a set of pairs $X \in \mathcal{X}$, let $T(X)$ be the set of non-admissible pairs, i.e. containing $(i, j) \in \mathcal{P} \setminus X$ such that $X \cup (i, j) \notin \mathcal{X}$. We have

$$\mathbf{Prob}(X_m \in \mathcal{X}) = \sum_{x \in \mathcal{X} : |x| = m-1} \left(1 - \frac{|T(x)|}{|\mathcal{P}| - (m-1)}\right) \mathbf{Prob}(X_{m-1} = x). \tag{8}$$

Note that every selected pair from $\mathcal{P}$ contributes at most $15n$ non-admissible pairs. Indeed, given a selected pair (i, j), a non-admissible pair (s, t) should respect one of the following conditions $|s - i| \leq 2$, $|s - j| \leq 2$, $|t - i| \leq 2$, $|t - j| \leq 2$ or $|s - t| \leq 2$. Given any item s, there are 15 possible choice of t to output a non-admissible pair (s, t), resulting in at most $15n$ non-admissible pairs for the selected pair (i, j).

Hence, for every $x \in \mathcal{X}$ we have

$$|T(x)| \leq 15n|x|.$$

Combining this with (8) and the fact that $|\mathcal{P}| = \binom{n}{2}$, we have

$$\mathbf{Prob}(X_m \in \mathcal{X}) \geq \left(1 - \frac{15n}{\binom{n}{2} - (m-1)}(m-1)\right) \mathbf{Prob}(X_{m-1} \in \mathcal{X}).$$

From this it follows

$$\begin{aligned} p(n, m) &\geq \prod_{i=1}^{m-1} \left(1 - \frac{15n}{\binom{n}{2} - (i-1)} i\right) \\ &\geq \prod_{i=1}^{m-1} \left(1 - \frac{i}{a(n, m)}\right) \end{aligned}$$

where

$$a(n, m) = \frac{\binom{n}{2} - (m-1)}{15n}.$$

Notice that when $m = o(n)$ we have $\left(1 - \frac{i}{a(n,m)}\right) \sim \exp(-30i/n)$ and

$$\prod_{i=1}^{m-1} \left(1 - \frac{i}{a(n, m)}\right) \sim \prod_{i=1}^{m-1} \exp(-30i/n) \sim \exp\left(-\frac{15m^2}{n}\right) \text{ for large } n.$$

Hence, given $\delta > 0$, $p(n, m) \geq 1 - \delta$ provided that $m = O(\sqrt{n\delta})$. If $\delta = o(1)$, the condition is $m = o(\sqrt{n})$. ■

5. Spectral Perturbation Analysis

In this section we analyze how SerialRank performs when only a small fraction of pairwise comparisons are given. We show that for Erdös-Rényi graphs, i.e., when pairwise comparisons are observed independently with a given probability, $\Omega(n \log^4 n)$ comparisons suffice for ℓ_2 consistency of the Fiedler vector and hence ℓ_2 consistency of the retreived ranking w.h.p. On the other hand we need $\Omega(n^{3/2} \log^4 n)$ comparisons to retrieve a ranking whose local perturbations are bounded in ℓ_∞ norm. Since Erdös-Rényi graphs are connected with high probability only when the total number of pairs sampled scales as $\Omega(n \log n)$, we need at least that many comparisons in order to retrieve a ranking, therefore the ℓ_2 consistency result can be seen as optimal up to a polylogarithmic factor.

Our bounds are mostly related to the work of (Wauthier et al., 2013). In its simplified version (Theorem 4.2 Wauthier et al., 2013) shows that when ranking items according to their point score, for any precision parameter $\mu \in (0, 1)$, sampling independently with fixed probability $\Omega\left(\frac{n \log n}{\mu^2}\right)$ comparisons guarantees that the maximum displacement between the retrieved ranking and the true ranking, i.e., the ℓ_∞ distance to the true ranking, is bounded by μn with high probability for n large enough.

Sample complexity bounds have also been studied for the Rank Centrality algorithm (Dwork et al., 2001; Negahban et al., 2012). In their analysis, (Negahban et al., 2012) suppose that some pairs are sampled independently with fixed probability, and then k comparisons are generated for each sampled pair, under a Bradley-Terry-Luce model (BTL). When ranking items according to the stationary distribution of a transition matrix estimated from comparisons, sampling $\Omega(n \cdot \text{polylog}(n))$ pairs are enough to bound the relative ℓ_2 norm perturbation of the stationary distribution. However, as pointed out by (Wauthier et al., 2013), repeated measurements are not practical, e.g., if comparisons are derived from the outcomes of sports games or the purchasing behavior of a customer (a customer typically wants to purchase a product only once). Moreover, (Negahban et al., 2012) do not provide bounds on the relative ℓ_∞ norm perturbation of the ranking.

We also refer the reader to the recent work of Rajkumar and Agarwal (2014), who provide a survey of sample complexity bounds for Rank Centrality, maximum likelihood estimation, least-square ranking and an SVM based ranking, under a more flexible sampling model. However, those bounds only give the sampling complexity for exact recovery of ranking, which is usually prohibitive when n is large, and are more difficult to interpret.

Finally, we refer the interested reader to (Huang et al., 2008; Shamir and Tishby, 2011) for sampling complexity bounds in the context of spectral clustering.

Limitations. We emphasize that sampling models based on Erdös-Rényi graphs are not the most realistic, though they have been studied widely in the literature (see for instance Feige et al., 1994; Braverman and Mossel, 2008; Wauthier et al., 2013). Indeed, pairs are not likely to be sampled independently. For instance, when ranking movies, popular movies in the top ranks are more likely to be compared. Corrupted comparisons are also more likely between items that have close rankings. We hope to extend our perturbation analysis to more general models in future work.

A second limitation of our perturbation analysis comes from the setting of ordinal comparisons, i.e., binary comparisons, since in many applications, several comparisons are provided for each sampled pair. Nevertheless, the setting of ordinal comparisons is interesting

for the analysis of SerialRank, since numerical experiments suggest that it is the setting for which SerialRank provides the best results compared to other methods. Note that in practice, we can easily get rid of this limitation (see Section 2.2.2 and 6). We refer the reader to numerical experiments in Section 6, as well as a recent paper by Cucuringu (2015), which introduces another ranking algorithm called SyncRank, and provides extensive numerical experiments on state-of-the-art ranking algorithms, including SerialRank.

Choice of Laplacian: normalized vs. unnormalized. In the spectral clustering literature, several constructions for the Laplacian operators are suggested, namely the unnormalized Laplacian (used in SerialRank), the symmetric normalized Laplacian, and the non-symmetric normalized Laplacian. Von Luxburg et al. (2008) show stronger consistency results for spectral clustering by using the non-symmetric normalized Laplacian. Here, we show that the Fiedler vector of the normalized Laplacian is an affine function of the ranking, hence sorting the Fiedler vector still guarantees exact recovery of the ranking, when all comparisons are observed and consistent with a global ranking. In contrast, we only get an asymptotic expression for the unnormalized Laplacian (*cf.* section A). This motivated us to provide an analysis of SerialRank robustness based on the normalized Laplacian, though in practice the use of the unnormalized Laplacian is valid and seems to give better results (*cf.* Figures 2 and 5).

Notations. Throughout this section, we only focus on the similarity S^{match} in (3) and write it S to simplify notations. W.l.o.g. we assume in the following that the true ranking is the identity, hence S is an R-matrix. We write $\|\cdot\|_2$ the operator norm of a matrix, which corresponds to the maximal absolute eigenvalue for symmetric matrices. $\|\cdot\|_F$ denotes the Frobenius norm. We refer to the eigenvalues of the Laplacian as λ_i, with $\lambda_1 = 0 \leq \lambda_2 \leq \ldots \leq \lambda_n$. For any quantity x, we denote by $\tilde{x}$ its perturbed analogue. We define the residual matrix $R = \tilde{S} - S$ and write f the normalized Fiedler vector of the Laplacian matrix L_S. We define the degree matrix $D_S = \mathbf{diag}(D\mathbf{1})$ the diagonal matrix whose elements are the row-sums of matrix S. Whenever we use the abreviation w.h.p., this means that the inequality is true with probability greater than $1 - 2/n$. Finally, we will use $c > 0$ for absolute constants, whose values are allowed to vary from one equation to another.

We assume that our information on preferences is both incomplete and corrupted. Specifically, pairwise comparisons are independently sampled with probability q and these sampled comparisons are consistent with the underlying total ranking with probability p. Let us define $\tilde{C} = B \circ C$ the matrix of observed comparisons, where C is the true comparison matrix defined in (1), $\circ$ is the Hadamard product and B is a symmetric matrix with entries

$$B_{i,j} = \begin{cases} 0 & \text{with probability } 1-q \\ 1 & \text{with probability } qp \\ -1 & \text{with probability } q(1-p). \end{cases}$$

In order to obtain an unbiased estimator of the comparison matrix defined in (1), we normalize $\tilde{C}$ by its mean value $q(2p-1)$ and redefine $\tilde{S}$ as

$$\tilde{S} = \frac{1}{q^2(2p-1)^2}\tilde{C}\tilde{C}^T + n\mathbf{1}\mathbf{1}^T.$$

For ease of notations we have dropped the factor $1/2$ in (3) w.l.o.g. (positive multiplicative factors of the Laplacian do not affect its eigenvectors).

5.1 Results

We now state our main results. The first one bounds ℓ_2 perturbations of the Fiedler vector f with both missing and corrupted comparisons. Note that f and $\tilde{f}$ are normalized.

Theorem 21 *For every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, then*

$$\|\tilde{f} - f\|_2 \leq c\frac{\mu}{\sqrt{\log n}}$$

with probability at least $1 - 2/n$, where $c > 0$ is an absolute constant.

As n goes to infinity the perturbation of the Fiedler vector goes to zero, and we can retrieve the "true" ranking by reordering the Fiedler vector. Hence this bounds provides ℓ_2 consistency of the ranking, with an optimal sampling complexity (up to a polylogarithmic factor).

The second result bounds local perturbations of the ranking with π referring to the "true" ranking and $\tilde{\pi}$ to the ranking retrieved by SerialRank.

Theorem 24 *For every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4\sqrt{n}}$, then*

$$\|\tilde{\pi} - \pi\|_\infty \leq c\mu n$$

with probability at least $1 - 2/n$, where $c > 0$ is an absolute constant.

This bound quantifies the maximum displacement of any item's ranking. μ can be seen a "precision" parameter. For instance, if we set $\mu = 0.1$, Theorem 24 means that we can expect the maximum displacement of any item's ranking to be less than $0.1 \cdot n$ when observing $c^2 \cdot 100 \cdot n\sqrt{n} \cdot \log^4 n$ comparisons (with $p = 1$).

We conjecture Theorem 24 still holds true if the condition $q > \log^4 n/\mu^2(2p-1)^4\sqrt{n}$ is replaced by the weaker condition $q > \log^4 n/\mu^2(2p-1)^4 n$.

5.2 Sketch of the proof.

The proof of these results relies on classical perturbation arguments and is structured as follows.

- **Step 1:** Bound $\|\tilde{D}_S - D_S\|_2$, $\|\tilde{S} - S\|_2$ with high probability using concentration inequalities on quadratic forms of Bernoulli variables and results from (Achlioptas and McSherry, 2007).
- **Step 2.** Show that the normalized Laplacian $L = \mathbf{I} - D^{-1}S$ has a linear Fiedler vector and bound the eigengap between the Fiedler value and other eigenvalues.
- **Step 3.** Bound $\|\tilde{f} - f\|_2$ using Davis-Kahan theorem and bounds of steps 1 and 2.
- **Step 4.** Use the linearity of the Fiedler vector to translate this result into a bound on the maximum displacement of the retrieved ranking $\|\tilde{\pi} - \pi\|_\infty$.

We now turn to the proof itself.

5.3 Step 1: Bounding $\|\tilde{D}_S - D_S\|_2$ and $\|\tilde{S} - S\|_2$

Here, we seek to bound $\|\tilde{D}_S - D_S\|_2$ and $\|\tilde{S} - S\|_2$ with high probability using concentration inequalities.

5.3.1 Bounding the norm of the degree matrix

We first bound perturbations of the degree matrix with both missing and corrupted comparisons.

Lemma 15 *For every $\mu \in (0,1)$ and $n \geq 100$, if $q \geq \frac{\log^4 n}{\mu^2(2p-1)^4 n}$ then*

$$\|\tilde{D}_S - D_S\|_2 \leq \frac{3\mu n^2}{\sqrt{\log n}}$$

with probability at least $1 - 1/n$.

Proof Let $R = \tilde{S} - S$ and $\delta = \mathbf{diag}\, D_R = \mathbf{diag}((\tilde{S} - S)\mathbf{1})$. Since D_S and $\tilde{D}_S$ are diagonal matrices, $\|\tilde{D}_S - D_S\|_2 = \max|\delta_i|$. We first seek a concentration inequality for each δ_i and then derive a bound on $\|\tilde{D}_S - D_S\|_2$.

By definition of the similarity matrix S and its perturbed analogue $\tilde{S}$ we have

$$R_{ij} = \sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}B_{jk}}{q^2(2p-1)^2} - 1\right).$$

Hence

$$\delta_i = \sum_{j=1}^{n} R_{ij} = \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}B_{jk}}{q^2(2p-1)^2} - 1\right).$$

Notice that we can arbitrarily fix the diagonal values of R to zeros. Indeed, the similarity between an element and itself should be a constant by convention, which leads to $R_{ii} = \tilde{S}_{ii} - S_{ii} = 0$ for all items i. Hence we could take $j \neq i$ in the definition of δ_i, and we can consider B_{ik} independent of B_{jk} in the associated summation.

We first seek a concentration inequality for each δ_i. Notice that

$$\begin{aligned}\delta_i &= \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}B_{jk}}{q^2(2p-1)^2} - 1\right) \\ &= \underbrace{\sum_{k=1}^{n}\left(\frac{C_{ik}B_{ik}}{q(2p-1)}\sum_{j=1}^{n} C_{jk}\left(\frac{B_{jk}}{q(2p-1)} - 1\right)\right)}_{\text{Quad}} + \underbrace{\sum_{k=1}^{n}\sum_{j=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}}{q(2p-1)} - 1\right)}_{\text{Lin}}.\end{aligned}$$

The first term (denoted Quad in the following) is quadratic with respect to the B_{jk} while the second term (denoted Lin in the following) is linear. Both terms have mean zero since the B_{ik} are independent of the B_{jk}. We begin by bounding the quadratic term Quad. Let $X_{jk} = C_{jk}\left(\frac{1}{q(2p-1)}B_{jk} - 1\right)$. We have

$$\mathbf{E}(X_{jk}) = C_{jk}\left(\tfrac{qp - q(1-p)}{q(2p-1)} - 1\right) = 0,$$

$$\mathbf{var}(X_{jk}) = \frac{\mathbf{var}(B_{jk})}{q^2(2p-1)^2} = \frac{1}{q^2(2p-1)^2}(q - q^2(2p-1)^2) = \frac{1}{q(2p-1)^2} - 1 \leq \frac{1}{q(2p-1)^2},$$

and

$$|X_{jk}| = \left|\frac{B_{jk}}{q(2p-1)} - 1\right| \leq 1 + \frac{1}{q(2p-1)} \leq \frac{2}{q(2p-1)} \leq \frac{2}{q(2p-1)^2}.$$

By applying Bernstein's inequality for any $t > 0$

$$\mathbf{Prob}\left(\left|\sum_{j=1}^{n} X_{jk}\right| > t\right) \leq 2\exp\left(\frac{-q(2p-1)^2t^2}{2(n + 2t/3)}\right) \leq 2\exp\left(\frac{-q(2p-1)^2t^2}{2(n+t)}\right). \quad (9)$$

Now notice that

$$\begin{aligned}\mathbf{Prob}(|\text{Quad}| > t) &= \mathbf{Prob}\left(\left|\sum_{k=1}^{n}\left(C_{ik}\frac{B_{ik}}{q(2p-1)}\sum_{j=1}^{n} X_{jk}\right)\right| > t\right) \\ &\leq \mathbf{Prob}\left(\sum_{k=1}^{n}\left(\frac{|B_{ik}|}{q(2p-1)}\right)\max_l\left|\sum_{j=1}^{n} X_{jl}\right| > t\right).\end{aligned}$$

By applying a union bound to the first Bernstein inequality (9), for any $t > 0$

$$\mathbf{Prob}\left(\max_l\left|\sum_{j=1}^{n} X_{jl}\right| > \sqrt{t}\right) \leq 2n\exp\left(\frac{-tq(2p-1)^2}{2(n+\sqrt{t})}\right).$$

Moreover, since $\mathbf{E}\,|B_{ik}| = q$ we also get from Bernstein's inequality that for any $t > 0$

$$\mathbf{Prob}\left(\sum_{k=1}^{n}\frac{|B_{ik}|}{q(2p-1)} > \frac{n}{2p-1} + \sqrt{t}\right) \leq \exp\left(\frac{-tq(2p-1)^2}{2(n+\sqrt{t})}\right).$$

We deduce from these last three inequalities that for any $t > 0$

$$\mathbf{Prob}(|\text{Quad}| > t) \leq (2n+1)\exp\left(\frac{-tq(2p-1)^2}{2(n+\sqrt{t})}\right).$$

Taking $t = \mu^2(2p-1)^2n^2/\log n$ and $q \geq \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, with $\mu \leq 1$, we have $\sqrt{t} \leq n$ and we deduce that

$$\mathbf{Prob}\left(|\text{Quad}| > \frac{2\mu n^2}{\sqrt{\log n}}\right) \leq (2n+1)\exp\left(-\frac{\log^3 n}{4}\right). \quad (10)$$

We now bound the linear term Lin.

$$\begin{aligned}\mathbf{Prob}(|\text{Lin}| > t) &= \mathbf{Prob}\left(\left|\sum_{j=1}^{n}\sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}}{q(2p-1)} - 1\right)\right| > t\right) \\ &\leq \mathbf{Prob}\left(\sum_{k=1}^{n}|C_{ik}|\max_l\left|\sum_{j=1}^{n} X_{jl}\right| > t\right) \\ &\leq \mathbf{Prob}\left(\max_k\left|\sum_{j=1}^{n} X_{jk}\right| > t/n\right),\end{aligned}$$

hence

$$\mathbf{Prob}(|\mathrm{Lin}| > t) \leq 2n \exp\left(\frac{-t^2 q(2p-1)^2}{2n^2(n+t/n)}\right).$$

Taking $t = \mu n^2/(\log n)^{1/2}$ and $q \geq \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, with $\mu \leq 1$, we have $t \leq n^2$ and we deduce that

$$\mathbf{Prob}(|\mathrm{Lin}| > t) \leq 2n \exp\left(-\frac{\log^3 n}{4}\right). \tag{11}$$

Finally, combining equations (10) and (11), we obtain for $q \geq \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, with $\mu \leq 1$

$$\mathbf{Prob}\left(|\delta_i| > \frac{3\mu n^2}{\sqrt{\log n}}\right) \leq (4n+1) \exp\left(-\frac{\log^3 n}{4}\right).$$

Now, using a union bound, this shows that for $q \geq \frac{\log^4 n}{\mu^2(2p-1)^4 n}$,

$$\mathbf{Prob}\left(\max|\delta_i| > \frac{3\mu n^2}{\sqrt{\log n}}\right) \leq n(4n+1) \exp\left(-\frac{\log^3 n}{4}\right),$$

which is less than $1/n$ for $n \geq 100$. ■

5.3.2 Bounding perturbations of the comparison matrix C

Here, we adapt results in (Achlioptas and McSherry, 2007) to bound perturbations of the comparison matrix. We will then use bounds on the perturbations of C to bound $\|\tilde{S} - S\|_2$.

Lemma 16 *For $n \geq 104$ and $q \geq \frac{\log^3 n}{n}$,*

$$\|C - \tilde{C}\|_2 \leq \frac{c}{2p-1}\sqrt{\frac{n}{q}}, \tag{12}$$

with probability at least $1 - 2/n$, where c is an absolute constant.

Proof The main argument of the proof is to use the independence of the C_{ij} for $i < j$ in order to bound $\|\tilde{C} - C\|_2$ by a constant times $\sigma\sqrt{n}$, where σ is the standard deviation of C_{ij}. To isolate independent entries in the perturbation matrix, we first need to break the anti-symmetry of $\tilde{C} - C$ by decomposing $X = \tilde{C} - C$ into its upper triangular part and its lower triangular part, i.e., $\tilde{C} - C = X_{\mathrm{up}} + X_{\mathrm{low}}$, with $X_{\mathrm{up}} = -X_{\mathrm{low}}^T$ (diagonal entries of $\tilde{C} - C$ can be arbitrarily set to 0). Entries of X_{up} are all independent, with variance less than the variance of $\tilde{C}_{ij}$. Indeed, lower entries of X_{up} are equal to 0 and hence have variance 0. Notice that

$$\|\tilde{C} - C\|_2 = \|X_{\mathrm{up}} + X_{\mathrm{low}}\|_2 \leq \|X_{\mathrm{up}}\|_2 + \|X_{\mathrm{low}}\|_2 \leq 2\|X_{\mathrm{up}}\|_2,$$

so bounding $\|X_{\mathrm{up}}\|_2$ will give us a bound on $\|X\|_2$. In the rest of the proof we write X_{up} instead of X to simplify notations. We can now apply (Achlioptas and McSherry, 2007, Th. 3.1) to X. Since

$$X_{ij} = \tilde{C}_{ij} - C_{ij} = C_{ij}\left(\frac{B_{ij}}{q(2p-1)} - 1\right),$$

we have (*cf.* proof of Lemma 15) $\mathbf{E}(X_{ij}) = 0$, $\mathbf{var}(X_{ij}) \leq \frac{1}{q(2p-1)^2}$, and $|X_{ij}| \leq \frac{2}{q(2p-1)}$. Hence for a given $\epsilon > 0$ such that

$$\frac{4}{q(2p-1)} \leq \left(\frac{\log(1+\epsilon)}{2\log(2n)}\right)^2 \frac{\sqrt{2n}}{\sqrt{q}(2p-1)}, \tag{13}$$

for any $\theta > 0$ and $n \geq 76$,

$$\mathbf{Prob}\left(\|X\|_2 \geq 2(1+\epsilon+\theta)\frac{1}{\sqrt{q}(2p-1)}\sqrt{2n}\right) < 2\exp\left(-16\frac{\theta^2}{\epsilon^4}\log^3 n\right). \tag{14}$$

For $q \geq \frac{(\log 2n)^3}{n}$ and taking $\epsilon \geq \exp(\sqrt{(16/\sqrt{(2)})}) - 1$ (so $\log(1+\epsilon)^2 \geq 16/\sqrt{2}$) means inequality (13) holds. Taking (14) with $\epsilon = 30$ and $\theta = 30$ we get

$$\mathbf{Prob}\left(\|X\|_2 \geq \frac{2\sqrt{2}(1+30+30)}{2p-1}\sqrt{\frac{n}{q}}\right) < 2\exp\left(-10^{-2}\log^3 n\right). \tag{15}$$

Hence for $n \geq 104$, we have $\log^3 n > 100$ and

$$\mathbf{Prob}\left(\|X\|_2 \geq \frac{173}{2p-1}\sqrt{\frac{n}{q}}\right) < 2/n.$$

Noting that $\log 2n \leq 1.15 \log n$ for $n \geq 104$, we obtain the desired result by choosing $c = 2 \times 173 \times \sqrt{1.15} \leq 371$. ■

5.3.3 Bounding the perturbation of the similarity matrix $\|S\|$.

We now seek to bound $\|\tilde{S} - S\|$ with high probability.

Lemma 17 *For every $\mu \in (0,1)$, $n \geq 104$, if $q > \frac{\log^4 n}{\mu^2(2p-1)^2 n}$, then*

$$\|\tilde{S} - S\|_2 \leq c\frac{\mu n^2}{\sqrt{\log n}},$$

with probability at least $1 - 2/n$, where c is an absolute constant.

Proof Let $X = \tilde{C} - C$. We have

$$\tilde{C}\tilde{C}^T = (C+X)(C+X)^T = CC^T + XX^T + XC^T + CX^T,$$

hence

$$\tilde{S} - S = XX^T + XC^T + CX^T,$$

and

$$\|\tilde{S} - S\|_2 \leq \|XX^T\|_2 + \|XC^T\|_2 + \|CX^T\|_2 \leq \|X\|_2^2 + 2\|X\|_2\|C\|_2.$$

From Lemma 16 we deduce that for $n \geq 104$ and $q \geq \frac{\log^4 n}{n}$, with probability at least $1 - 2/n$

$$\|\tilde{S} - S\|_2 \leq \frac{c^2 n}{q(2p-1)^2} + \frac{2c}{2p-1}\sqrt{\frac{n}{q}}\|C\|_2. \tag{16}$$

Notice that $\|C\|_2^2 \leq \mathbf{Tr}(CC^T) = n^2$, hence $\|C\|_2 \leq n$ and

$$\|\tilde{S} - S\|_2 \leq \frac{c^2 n}{q(2p-1)^2} + \frac{2cn}{2p-1}\sqrt{\frac{n}{q}}. \tag{17}$$

By taking $q > \frac{\log^4 n}{\mu^2(2p-1)^2 n}$, we get for $n \geq 104$ with probability at least $1 - 2/n$

$$\|\tilde{S} - S\|_2 \leq \frac{c^2\mu^2 n^2}{\log^4 n} + \frac{2c\mu n^2}{\log^2 n}.$$

Hence setting a new constant c with $c = \max(c^2(\log 104)^{-7/2}, 2c(\log 104)^{-3/2}) \leq 270$,

$$\|\tilde{S} - S\|_2 \leq c\frac{\mu n^2}{\sqrt{\log n}}$$

with probability at least $1 - 2/n$, which is the desired result. ■

5.4 Step 2: Controlling the eigengap

In the following proposition we show that the normalized Laplacian of the similarity matrix S has a constant Fiedler value and a linear Fiedler vector. We then deduce bounds on the eigengap between the first, second and third smallest eigenvalues of the Laplacian.

Proposition 18 *Let $L^{\text{norm}} = \mathbf{I} - D^{-1}S$ be the non-symmetric normalized Laplacian of S. L^{norm} has a linear Fiedler vector, and its Fiedler value is equal to $2/3$.*

Proof Let $x_i = i - \frac{n+1}{2}$ (x is linear with mean zero). We want to show that $L^{\text{norm}}x = \lambda_2 x$ or equivalently $Sx = (1-\lambda_2)Dx$. We develop both sides of the last equation, and use the following facts

$$S_{i,j} = n - |j - i|, \quad \sum_{k=1}^{n} k = \frac{n(n+1)}{2}, \quad \sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}.$$

We first get an expression for the degree of S, defined by $d = S\mathbf{1} = \sum_{i=1}^n S_{i,k}$, with

$$\begin{aligned} d_i &= \sum_{k=1}^{i-1} S_{i,k} + \sum_{k=i}^{n} S_{i,k} \\ &= \sum_{k=1}^{i-1}(n - i + k) + \sum_{k=i}^{n}(n - k + i) \\ &= \frac{n(n-1)}{2} + i(n - i + 1). \end{aligned}$$

Similarly we have

$$\begin{aligned} \sum_{k=1}^{n} kS_{i,k} &= \sum_{k=1}^{i-1} k(n - i + k) + \sum_{k=i}^{n} k(n - k + i) \\ &= \frac{n^2(n+1)}{2} + \frac{i(i-1)(2i-1)}{3} - \frac{n(n+1)(2n+1)}{6} - i^2(i-1) + i\frac{n(n+1)}{2}. \end{aligned}$$

Finally, setting $\lambda_2 = 2/3$, notice that

$$\begin{aligned} [Sx]_i &= \sum_{k=1}^{n} S_{i,k}\left(k - \frac{n+1}{2}\right) \\ &= \sum_{k=1}^{n} kS_{i,k} - \frac{n+1}{2}d_i \\ &= \frac{1}{3}\left(\frac{n(n-1)}{2} + i(n-i+1)\right)\left(i - \frac{n+1}{2}\right) \\ &= (1-\lambda_2)d_i x_i, \end{aligned}$$

which shows that $Sx = (1-\lambda_2)Dx$. ■

The next corollary will be useful in following proofs.

Corollary 19 *The Fiedler vector f of the unperturbed Laplacian satisfies $\|f\|_\infty \leq 2/\sqrt{n}$.*

Proof We use the fact that f is collinear to the vector x defined by $x_i = i - \frac{n+1}{2}$ and verifies $\|f\|_2 = 1$. Let us consider the case of n odd. The Fiedler vector verifies $f_i = \frac{i-(n+1)/2}{a_n}$, with

$$a_n^2 = 2\sum_{k=0}^{(n-1)/2} k^2 = \frac{2}{6}\frac{n-1}{2}\left(\frac{n-1}{2} + 1\right)((n-1)+1) = \frac{n^3 - n}{12}.$$

Hence

$$\|f\|_\infty = f_n = \frac{n-1}{2a_n} \leq \sqrt{\frac{3}{n-1}} \leq \frac{2}{\sqrt{n}} \quad \text{for} \quad n \geq 5.$$

A similar reasoning applies for n even. ■

Lemma 20 *The minimum eigengap between the Fiedler value and other eigenvalues is bounded below by a constant for n sufficiently large.*

Proof The first eigenvalue of the Laplacian is always 0, so we have for any n, $\lambda_2 - \lambda_1 = \lambda_2 = 2/3$. Moreover, using results from (Von Luxburg et al., 2008), we know that eigenvalues of the normalized Laplacian that are different from one converge to an asymptotic spectrum, and that the limit eigenvalues are "isolated". Hence there exists $n_0 > 0$ and $c > 0$ such that for any $n \geq n_0$ we have $\lambda_3 - \lambda_2 > c$. ■

Numerical experiments show that λ_3 converges to $0.93\ldots$ very fast when n grows towards infinity.

5.5 Step 3: Bounding the perturbation of the Fiedler vector $\|\tilde{f} - f\|_2$

We can now compile results from previous sections to get a first perturbation bound and show ℓ_2 consistency of the Fiedler vector when comparisons are both missing and corrupted.

Theorem 21 *For every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, then*

$$\|\tilde{f} - f\|_2 \leq c\frac{\mu}{\sqrt{\log n}},$$

with probability at least $1 - 2/n$.

Proof In order to use Davis-Kahan theorem, we need to relate perturbations of the normalized Laplacian matrix to perturbations of the similarity and degree matrices. To simplify notations, we write $L = \mathbf{I} - D^{-1}S$ and $\tilde{L} = \mathbf{I} - \tilde{D}^{-1}\tilde{S}$.

Since the normalized Laplacian is not symmetric, we will actually apply Davis-Kahan theorem to the symmetric normalized Laplacian $L_{sym} = \mathbf{I} - D^{-1/2}SD^{-1/2}$. It is easy to see that L_{sym} and L have the same Fiedler value, and that the Fiedler vector f_{sym} of L_{sym} is equal to $D^{1/2}f$ (up to normalization). Indeed, if v is the eigenvector associated to the i^{th} eigenvalue of L (denoted by λ_i), then

$$L_{sym}D^{1/2}v = D^{-1/2}(D-S)D^{-1/2}D^{1/2}v = D^{-1/2}(D-S)v = D^{1/2}(\mathbf{I} - D^{-1}S)v = \lambda_i D^{1/2}v.$$

Hence perturbations of the Fiedler vector of L_{sym} are directly related to perturbations of the Fiedler vector of L.

The proof relies mainly on Lemma 15, which states that for $n \geq 100$, denoting by d the vector of diagonal elements of D_S,

$$\|D_R\|_2 = \max |\tilde{d}_i - d_i| \leq \frac{3\mu n^2}{\sqrt{\log n}}$$

with probability at least $1 - \frac{2}{n}$. Combined with the fact that $d_i = \frac{n(n-1)}{2} + i(n-i+1)$ (*cf.* proof of Proposition 18), this guarantees that d_i and $\tilde{d}_i$ are strictly positive. Hence $D^{-1/2}$ and $\tilde{D}^{-1/2}$ are well defined. We now decompose the perturbation of the Laplacian matrix. Let $\Delta = D^{-1/2}$, we have

$$\begin{aligned}
\|\tilde{L}_{sym} - L_{sym}\|_2 &= \|\tilde{\Delta}\tilde{S}\tilde{\Delta} - \Delta S \Delta\|_2 \\
&= \|\tilde{\Delta}\tilde{S}\tilde{\Delta} - \tilde{\Delta}S\tilde{\Delta} + \tilde{\Delta}S\tilde{\Delta} - \Delta S\Delta\|_2 \\
&= \|\tilde{\Delta}(\tilde{S}-S)\tilde{\Delta} + \tilde{\Delta}S\tilde{\Delta} - \Delta S\tilde{\Delta} + \Delta S\tilde{\Delta} - \Delta S\Delta\|_2 \\
&= \|\tilde{\Delta}(\tilde{S}-S)\tilde{\Delta} + (\tilde{\Delta}-\Delta)S\tilde{\Delta} + \Delta S(\tilde{\Delta}-\Delta)\|_2 \\
&\leq \|\tilde{\Delta}\|_2^2\|\tilde{S}-S\|_2 + \|S\|_2(\|\tilde{\Delta}\|_2 + \|\Delta\|_2)\|\tilde{\Delta}-\Delta\|_2.
\end{aligned}$$

We first bound $\|\tilde{\Delta} - \Delta\|_2$. Notice that

$$\|\tilde{\Delta} - \Delta\|_2 = \max_i |\tilde{d}_i^{-1/2} - d_i^{-1/2}|,$$

where d_i (respectively $\tilde{d}_i$) is the sum of elements of the i^{th} row of S (respectively $\tilde{S}$). Hence

$$\|\tilde{\Delta} - \Delta\|_2 = \max_i \frac{\left|\sqrt{\tilde{d}_i} - \sqrt{d_i}\right|}{\sqrt{\tilde{d}_i d_i}} = \max_i \frac{\left|\tilde{d}_i - d_i\right|}{\sqrt{\tilde{d}_i d_i}(\sqrt{\tilde{d}_i} + \sqrt{d_i})}.$$

Using Lemma 15 we obtain

$$\|\tilde{\Delta} - \Delta\|_2 \leq \max_i \frac{\frac{3\mu n^2}{\sqrt{\log n}}}{\sqrt{d_i}(d_i - \frac{3\mu n^2}{\sqrt{\log n}}) + d_i\sqrt{d_i - \frac{3\mu n^2}{\sqrt{\log n}}}}, \quad i = 1, \ldots, n, \text{ w.h.p.}$$

Since $d_i = \frac{n(n-1)}{2} + i(n-i+1)$ (*cf.* proof of Proposition 18), for $\mu < 1$ there exists a constant c such that $d_i > d_i - \frac{3\mu n^2}{\sqrt{\log n}} > cn^2$. We deduce that there exists an absolute constant c such that

$$\|\tilde{\Delta} - \Delta\|_2 \leq \frac{c\mu}{n\sqrt{\log n}} \text{ w.h.p.} \tag{18}$$

Similarly we obtain that

$$\|\Delta\|_2 \leq \frac{c}{n} \text{ w.h.p,} \tag{19}$$

and

$$\|\tilde{\Delta}\|_2 \leq \frac{c}{n} \text{ w.h.p.} \tag{20}$$

Moreover, we have

$$\|S\|_2 = \|CC^T + n\mathbf{1}\mathbf{1}^T\|_2 \leq \|C\|_2^2 + n\|\mathbf{1}\mathbf{1}^T\|_2 \leq 2n^2.$$

Hence,

$$\|S\|_2(\|\tilde{\Delta}\|_2 + \|\Delta\|_2)\|\tilde{\Delta} - \Delta\|_2 \leq \frac{c\mu}{\sqrt{\log n}} \text{ w.h.p,}$$

where $c := 4c^2$. Using Lemma 17, we can similarly bound $\|\tilde{\Delta}\|_2^2\|\tilde{S} - S\|_2$ and obtain

$$\|\tilde{L}_{sym} - L_{sym}\|_2 \leq \frac{c\mu}{\sqrt{\log n}} \text{ w.h.p,} \tag{21}$$

where c is an absolute constant. Finally, for small μ, Weyl's inequality, equation (21) together with Lemma 20 ensure that for n large enough with high probability $|\tilde{\lambda}_3 - \lambda_2| > |\lambda_3 - \lambda_2|/2$ and $|\tilde{\lambda}_1 - \lambda_2| > |\lambda_1 - \lambda_2|/2$. Hence we can apply Davis-Kahan theorem. Compiling all constants into c we obtain

$$\|\tilde{f}_{sym} - f_{sym}\|_2 \leq \frac{c\mu}{\sqrt{\log n}} \text{ w.h.p.} \tag{22}$$

Finally we relate the perturbations of f_{sym} to the perturbations of f. Since $f_{sym} = \frac{D^{1/2}f}{\|D^{1/2}f\|_2}$, letting $\alpha_n = \|D^{1/2}f\|$, we deduce that

$$\begin{aligned}
\|\tilde{f} - f\|_2 &= \|\tilde{\alpha}_n\tilde{\Delta}\tilde{f}_{sym} - \alpha_n\Delta f_{sym}\|_2 \\
&= \|\Delta(\tilde{\alpha}_n\tilde{f}_{sym} - \alpha_n f_{sym}) + \tilde{\alpha}_n(\tilde{\Delta} - \Delta)\tilde{f}_{sym}\|_2 \\
&\leq \|\Delta\|_2\|\tilde{\alpha}_n\tilde{f}_{sym} - \alpha_n f_{sym}\|_2 + \|\tilde{\alpha}_n\|_2\|\tilde{\Delta} - \Delta\|_2.
\end{aligned}$$

Similarly as for inequality (18), we can show that $\|\tilde{D}^{1/2}\|$ and $\|D^{1/2}\|$ are of the same order $O(n)$. Since $\|f\|_2 = \|\tilde{f}\|_2 = 1$, this is also true for $\|\alpha_n\|_2$ and $\|\tilde{\alpha}_n\|_2$. We conclude the proof using inequalities (18), (19) and (22). ■

5.6 Bounding ranking perturbations $\|\tilde{\pi} - \pi\|_\infty$

SerialRank's ranking is derived by sorting the Fiedler vector. While the consistency result in Theorem 21 shows the ℓ_2 estimation error going to zero as n goes to infinity, this is not sufficient to quantify the maximum displacement of the ranking. To quantify the maximum displacement of the ranking, as in (Wauthier et al., 2013), we need to bound $\|\tilde{\pi} - \pi\|_\infty$ instead.

We bound the maximum displacement of the ranking here with an extra factor $\sqrt{n}$ compared to the sampling rate in (Wauthier et al., 2013). We would only need a better component-wise bound on $\tilde{S} - S$ to get rid of this extra factor $\sqrt{n}$, and we hope to achieve it in future work.

The proof is in two parts: we first bound the ℓ_∞ norm of the perturbation of the Fiedler vector, then translate this perturbation of the Fiedler vector into a perturbation of the ranking.

5.6.1 Bounding the ℓ_∞ norm of the Fiedler vector perturbation

We start by a technical lemma bounding $\|(\tilde{S} - S)f\|_\infty$.

Lemma 22 *Let $r > 0$, for every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, then*

$$\|(\tilde{S} - S)f\|_\infty \leq \frac{3\mu n^{3/2}}{\sqrt{\log n}}$$

with probability at least $1 - 2/n$.

Proof The proof is very much similar to the proof of Lemma 15 and can be found the Appendix (section A.2). ■

We now prove the main result of this section, bounding $\|\tilde{f} - f\|_\infty$ with high probability when roughly $O(n^{3/2})$ comparisons are sampled.

Lemma 23 *For every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4\sqrt{n}}$, then*

$$\|\tilde{f} - f\|_\infty \leq c\frac{\mu}{\sqrt{n \log n}}$$

with probability at least $1 - 2/n$, where c is an absolute constant.

Proof Notice that by definition $\tilde{L}\tilde{f} = \tilde{\lambda}_2\tilde{f}$ and $Lf = \lambda_2 f$. Hence for $\tilde{\lambda}_2 > 0$

$$\begin{aligned}\tilde{f} - f &= \frac{\tilde{L}\tilde{f}}{\tilde{\lambda}_2} - f \\ &= \frac{\tilde{L}\tilde{f} - Lf}{\tilde{\lambda}_2} + \frac{(\lambda_2 - \tilde{\lambda}_2)f}{\tilde{\lambda}_2}.\end{aligned}$$

Moreover

$$\begin{aligned}\tilde{L}\tilde{f} - Lf &= (\mathbf{I} - \tilde{D}^{-1}\tilde{S})\tilde{f} - (\mathbf{I} - D^{-1}S)f \\ &= (\tilde{f} - f) + D^{-1}Sf - \tilde{D}^{-1}\tilde{S}\tilde{f} \\ &= (\tilde{f} - f) + D^{-1}Sf - \tilde{D}^{-1}\tilde{S}f + \tilde{D}^{-1}\tilde{S}f - \tilde{D}^{-1}\tilde{S}\tilde{f} \\ &= (\tilde{f} - f) + (D^{-1}S - \tilde{D}^{-1}\tilde{S})f + \tilde{D}^{-1}\tilde{S}(f - \tilde{f})\end{aligned}$$

Hence

$$(\mathbf{I}(\tilde{\lambda}_2 - 1) + \tilde{D}^{-1}\tilde{S})(\tilde{f} - f) = (D^{-1}S - \tilde{D}^{-1}\tilde{S} + (\lambda_2 - \tilde{\lambda}_2)\mathbf{I})f. \tag{23}$$

Writing S_i the i^{th} row of S and d_i the degree of row i, using the triangle inequality, we deduce that

$$|\tilde{f}_i - f_i| \leq \frac{1}{|\tilde{\lambda}_2 - 1|}\left(|(d_i^{-1}S_i - \tilde{d}_i^{-1}\tilde{S}_i)f| + |\lambda_2 - \tilde{\lambda}_2||f_i| + |\tilde{d}_i^{-1}\tilde{S}_i(\tilde{f} - f)|\right). \tag{24}$$

It remains to bound each term separately, using Weyl's inequality for the denominator and previous lemmas for numerator terms, which is detailed in the Appendix (section A.2). ■

5.6.2 Bounding the ℓ_∞ norm of the ranking perturbation

First note that the ℓ_∞-norm of the ranking perturbation is equal to the number of pairwise disagreements between the true ranking and the retrieved one, i.e., for any i

$$|\tilde{\pi}_i - \pi_i| = \sum_{j<i} \mathbf{1}_{\tilde{f}_j > \tilde{f}_i} + \sum_{j>i} \mathbf{1}_{\tilde{f}_j < \tilde{f}_i}.$$

Now we will argue that when i and j are far apart, with high probability

$$\tilde{f}_j - \tilde{f}_i = (\tilde{f}_j - f_j) + (f_j - f_i) + (f_i - \tilde{f}_i)$$

will have the same sign as $j - i$. Indeed $|\tilde{f}_j - f_j|$ and $|\tilde{f}_i - f_i|$ can be bounded with high probability by a quantity less than $|f_j - f_i|/2$ for i and j sufficiently "far apart". Hence, $|\tilde{\pi}_i - \pi_i|$ is bounded by the number of pairs that are not sufficiently "far apart". We quantify the term "far apart" in the following proposition.

Theorem 24 *For every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^2\sqrt{n}}$, then*

$$\|\tilde{\pi} - \pi\|_\infty \leq c\mu n,$$

with probability at least $1 - 2/n$, where c is an absolute constant.

Proof We assume w.l.o.g. in the following that the true ranking is the identity, hence the unperturbed Fiedler vector f is strictly increasing. We first notice that for any $j > i$

$$\tilde{f}_j - \tilde{f}_i = (\tilde{f}_j - f_j) + (f_j - f_i) + (f_i - \tilde{f}_i).$$

Hence for any $j > i$

$$\|\tilde{f} - f\|_\infty \leq \frac{|f_j - f_i|}{2} \implies \tilde{f}_j \geq \tilde{f}_i.$$

Consequently, fixing an index i_0,

$$\sum_{j>i_0} \mathbf{1}_{\tilde{f}_j < \tilde{f}_{i_0}} \leq \sum_{j>i_0} \mathbf{1}_{\|\tilde{f}-f\|_\infty > \frac{|f_j - f_{i_0}|}{2}}.$$

Now recall that by Lemma 23, for $q > \frac{\log^4 n}{\mu^2 (2p-1)^2 \sqrt{n}}$

$$\|\tilde{f} - f\|_\infty \leq c \frac{\mu}{\sqrt{n \log n}}$$

with probability at least $1 - 2/n$. Hence

$$\sum_{j>i_0} \mathbf{1}_{\tilde{f}_j < \tilde{f}_{i_0}} \leq \sum_{j>i_0} \mathbf{1}_{\|\tilde{f}-f\|_\infty > \frac{|f_j - f_{i_0}|}{2}} \leq \sum_{j>i_0} \mathbf{1}_{\frac{c\mu}{\sqrt{n \log n}} > \frac{|f_j - f_{i_0}|}{2}} \quad \text{w.h.p.}$$

We now consider the case of n odd (a similar reasoning applies for n even). We have $f_j = \frac{j-(n+1)/2}{a_n}$ for all j, with

$$a_n^2 = 2 \sum_{k=0}^{(n-1)/2} k^2 = \frac{2}{6} \frac{n-1}{2} \left(\frac{n-1}{2} + 1 \right) ((n-1)+1) = \frac{n^3 - n}{12}.$$

Therefore

$$\frac{c\mu}{\sqrt{n \log n}} > \frac{|f_j - f_{i_0}|}{2} \iff \frac{c\mu}{\sqrt{n \log n}} > \frac{|j - i_0| \sqrt{3}}{n^{3/2}} \iff \frac{c\mu n}{\sqrt{3 \log n}} > |j - i_0|.$$

Dividing c by $\sqrt{3}$, we deduce that

$$\sum_{j>i_0} \mathbf{1}_{\tilde{f}_j < \tilde{f}_{i_0}} \leq \sum_{j>i_0} \mathbf{1}_{\frac{c\mu n}{\sqrt{\log n}} > |j - i_0|} = \left\lfloor \frac{c\mu n}{\sqrt{\log n}} \right\rfloor \leq \frac{c\mu n}{\sqrt{\log n}} \quad \text{w.h.p.}$$

Similarly

$$\sum_{j<i_0} \mathbf{1}_{\tilde{f}_j > \tilde{f}_{i_0}} \leq \frac{c\mu n}{\sqrt{\log n}} \quad \text{w.h.p.}$$

Finally, we obtain

$$|\tilde{\pi}_{i_0} - \pi_{i_0}| = \sum_{j<i_0} \mathbf{1}_{\tilde{f}_j > \tilde{f}_{i_0}} + \sum_{j>i_0} \mathbf{1}_{\tilde{f}_j < \tilde{f}_{i_0}} \leq \frac{c\mu n}{\sqrt{\log n}} \quad \text{w.h.p.},$$

where c is an absolute constant. Since the last inequality relies on $\|\tilde{f} - f\|_\infty \leq \frac{c\mu}{\sqrt{n \log n}}$, it is true for all i_0 with probabilty $1 - 2/n$, which concludes the proof. ■

6. Numerical Experiments

We now describe numerical experiments using both synthetic and real datasets to compare the performance of SerialRank with several classical ranking methods.

6.1 Synthetic Datasets

The first synthetic dataset consists of a matrix of pairwise comparisons derived from a given ranking of n items with uniform, randomly distributed corrupted or missing entries. A second synthetic dataset consists of a full matrix of pairwise comparisons derived from a given ranking of n items, with added "local" noise on the similarity between nearby items. Specifically, given a positive integer m, we let $C_{i,j} = 1$ if $i < j - m$, $C_{i,j} \sim \text{Unif}[-1, 1]$ if $|i - j| \leq m$, and $C_{i,j} = -1$ if $i > j + m$. In Figure 2, we measure the Kendall τ correlation coefficient between the true ranking and the retrieved ranking, when varying either the percentage of corrupted comparisons or the percentage of missing comparisons. Kendall's τ counts the number of agreeing pairs minus the number of disagreeing pairs between two rankings, scaled by the total number of pairs, so that it takes values between -1 and 1. Experiments were performed with $n = 100$ and reported Kendall τ values were averaged over 50 experiments, with standard deviation less than 0.02 for points of interest (i.e., with Kendall $\tau > 0.8$).

Results suggest that SerialRank (SR, full red line) produces more accurate rankings than point score (PS, (Wauthier et al., 2013) dashed blue line), Rank Centrality (RC (Negahban et al., 2012) dashed green line), and maximum likelihood (BTL (Bradley and Terry, 1952), dashed magenta line) in regimes with limited amount of corrupted and missing comparisons. In particular SerialRank seems more robust to corrupted comparisons. On the other hand, the performance deteriorates more rapidly in regimes with very high number of corrupted/missing comparisons. For a more exhaustive comparison of SerialRank to state-of-the art ranking algorithms, we refer the interested reader to a recent paper by Cucuringu (2015), which introduces another ranking algorithm called SyncRank, and provides extensive numerical experiments.

6.2 Real Datasets

The first real dataset consists of pairwise comparisons derived from outcomes in the TopCoder algorithm competitions. We collected data from 103 competitions among 2742 coders over a period of about one year. Pairwise comparisons are extracted from the ranking of each competition and then averaged for each pair. TopCoder maintains ratings for each participant, updated in an online scheme after each competition, which were also included in the benchmarks. To measure performance in Figure 3, we compute the percentage of upsets (i.e. comparisons disagreeing with the computed ranking), which is closely related to the Kendall τ (by an affine transformation if comparisons were coming from a consistent ranking). We refine this metric by considering only the participants appearing in the top k, for various values of k, i.e. computing

$$l_k = \frac{1}{|\mathcal{C}_k|} \sum_{i,j \in \mathcal{C}_k} \mathbf{1}_{r(i) > r(j)} \mathbf{1}_{C_{i,j} < 0}, \tag{25}$$

where $\mathcal{C}$ are the pairs (i, j) that are compared and such that i, j are both ranked in the top k, and $r(i)$ is the rank of i. Up to scaling, this is the loss considered in (Kenyon-Mathieu and Schudy, 2007).

This experiment shows that SerialRank gives competitive results with other ranking algorithms. Notice that rankings could probably be refined by designing a similarity matrix taking into account the specific nature of the data.

Table 1: Ranking of teams in the England premier league season 2013-2014.

Official	**Row-sum**	**RC**	**BTL**	**SerialRank**	**Semi-Supervised**
Man City (86)	Man City	Liverpool	Man City	Man City	Man City
Liverpool (84)	Liverpool	Arsenal	Liverpool	Chelsea	Chelsea
Chelsea (82)	Chelsea	Man City	Chelsea	Liverpool	Liverpool
Arsenal (79)	Arsenal	Chelsea	Arsenal	Arsenal	Everton
Everton (72)	Everton	Everton	Everton	Everton	Arsenal
Tottenham (69)	Tottenham	Tottenham	Tottenham	Tottenham	Tottenham
Man United (64)	Man United	Man United	Man United	Southampton	Man United
Southampton (56)	Southampton	Southampton	Southampton	Man United	Southampton
Stoke (50)	Stoke	Stoke	Stoke	Stoke	Newcastle
Newcastle (49)	Newcastle	Newcastle	Newcastle	Swansea	Stoke
Crystal Palace (45)	Crystal Palace	Swansea	Crystal Palace	Newcastle	West Brom
Swansea (42)	Swansea	Crystal Palace	Swansea	West Brom	Swansea
West Ham (40)	West Brom	West Ham	West Brom	Hull	Crystal Palace
Aston Villa (38)	West Ham	Hull	West Ham	West Ham	Hull
Sunderland (38)	Aston Villa	Aston Villa	Aston Villa	Cardiff	West Ham
Hull (37)	Sunderland	West Brom	Sunderland	Crystal Palace	Fulham
West Brom (36)	Hull	Sunderland	Hull	Fulham	Norwich
Norwich (33)	Norwich	Fulham	Norwich	Norwich	Sunderland
Fulham (32)	Fulham	Norwich	Fulham	Sunderland	Aston Villa
Cardiff (30)	Cardiff	Cardiff	Cardiff	Aston Villa	Cardiff

6.3 Semi-Supervised Ranking

We illustrate here how, in a semi-supervised setting, one can interactively enforce some constraints on the retrieved ranking, using e.g. the semi-supervised seriation algorithm in (Fogel et al., 2013). We compute rankings of England Football Premier League teams for season 2013-2014 (*cf.* figure 4 for seasons 2011-2012 and 2012-2013). Comparisons are defined as the averaged outcome (win, loss, or tie) of home and away games for each pair of teams. As shown in Table 1, the top half of SerialRank ranking is very close to the official ranking calculated by sorting the sum of points for each team (3 points for a win, 1 point for a tie). However, there are significant variations in the bottom half, though the number of upsets is roughly the same as for the official ranking. To test semi-supervised ranking, suppose for example that we are not satisfied with the ranking of Aston Villa (last team when ranked by the spectral algorithm), we can explicitly enforce that Aston Villa appears before Cardiff, as in the official ranking. In the ranking based on the corresponding semi-supervised seriation problem, Aston Villa is not last anymore, though the number of disagreeing comparisons remains just as low (*cf.* Figure 3, *right*).

7. Conclusion

We have formulated the problem of ranking from pairwise comparisons as a seriation problem, i.e. the problem of ordering from similarity information. By constructing an adequate similarity matrix, we applied a spectral relaxation for seriation to a variety of synthetic and real ranking datasets, showing competitive and in some cases superior performance compared to classical methods, especially in low noise environments. We derived performance bounds for this algorithm in the presence of corrupted and missing (ordinal) comparisons showing that SerialRank produces state-of-the art results for ranking based on ordinal comparisons, e.g. showing exact reconstruction w.h.p. when only $O(\sqrt{n})$ comparisons are missing. On the other hand, performance deteriorates when only a small fraction of comparisons are observed, or in the presence of very high noise. In this scenario, we showed that local ordering errors can be bounded if the number of samples is of order $O(n^{1.5}\text{polylog}(n))$ which is significantly above the optimal bound of $O(n \log n)$.

A few questions thus remain open, which we pose as future research directions. First of all, from a theoretical perspective, is it possible to obtain an ℓ_∞ bound on local perturbations of the ranking using only $O(n\,\text{polylog(n)})$ sampled pairs? Or, on the contrary, can we find a lower bound for spectral algorithms (i.e. perturbation arguments) imposing more than $\Omega(n\,\text{polylog(n)})$ sampled pairs? Note that those questions hold for all current spectral ranking algorithms.

Another line of research concerns the generalization of spectral ordering methods to more flexible settings, e.g., enforcing structural or a priori constraints on the ranking. Hierarchical ranking, i.e. running the spectral algorithm on increasingly refined subsets of the original data should be explored too. Early experiments suggests this works quite well, but no bounds are available at this point.

Finally, it would be interesting to investigate how similarity measures could be tuned for specific applications in order to improve SerialRank predictive power, for instance to take into account more information than win/loss in sports tournaments. Additional experiments in this vein can be found in Cucuringu (2015).

Appendix A.

We now detail several complementary technical results.

A.1 Exact recovery results with missing entries

Here, as in Section 4, we study the impact of one missing comparison on SerialRank, then extend the result to multiple missing comparisons.

Proposition 25 *Given pairwise comparisons $C_{s,t} \in \{-1, 0, 1\}$ between items ranked according to their indices, suppose only one comparison $C_{i,j}$ is missing, with $j - i > 1$ (i.e., $C_{i,j} = 0$), then S^{match} defined in* (3) *remains* strict-R *and the point score vector remains strictly monotonic.*

Proof We use the same proof technique as in Proposition 12. We write the true score and comparison matrix w and C, while the observations are written $\hat{w}$ and $\hat{C}$ respectively. This means in particular that $\hat{C}_{i,j} = 0$. To simplify notations we denote by S the similarity

matrix S^{match} (respectively $\hat{S}$ when the similarity is computed from observations). We first study the impact of the missing comparison $C_{i,j}$ for $i < j$ on the point score vector $\hat{w}$. We have

$$\hat{w}_i = \sum_{k=1}^{n} \hat{C}_{k,i} = \sum_{k=1}^{n} C_{k,i} + \hat{C}_{j,i} - C_{j,i} = w_i + 1,$$

similarly $\hat{w}_j = w_j - 1$, whereas for $k \neq i, j$, $\hat{w}_k = w_k$. Hence, w is still strictly increasing if $j > i+1$. If $j = i+1$ there is a tie between w_i and w_{i+1}. Now we show that the similarity matrix defined in (3) is an R-matrix. Writing $\hat{S}$ in terms of S, we get

$$[\hat{C}\hat{C}^T]_{i,t} = \sum_{k \neq j} \left(\hat{C}_{i,k}\hat{C}_{t,k} \right) + \hat{C}_{i,j}\hat{C}_{t,j} = \sum_{k \neq j} (C_{i,k}C_{t,k}) = \begin{cases} [CC^T]_{i,t} - 1 & \text{if } t < j \\ [CC^T]_{i,t} + 1 & \text{if } t > j. \end{cases}$$

We thus get

$$\hat{S}_{i,t} = \begin{cases} S_{i,t} - \frac{1}{2} & \text{if } t < j \\ S_{i,t} + \frac{1}{2} & \text{if } t > j, \end{cases}$$

(remember there is a factor $1/2$ in the definition of S). Similarly we get for any $t \neq i$

$$\hat{S}_{j,t} = \begin{cases} S_{j,t} + \frac{1}{2} & \text{if } t < i \\ S_{j,t} - \frac{1}{2} & \text{if } t > i. \end{cases}$$

Finally, for the single corrupted index pair (i, j), we get

$$\hat{S}_{i,j} = \frac{1}{2} \left(n + \sum_{k \neq i,j} \left(\hat{C}_{i,k}\hat{C}_{j,k} \right) + \hat{C}_{i,i}\hat{C}_{j,i} + \hat{C}_{i,j}\hat{C}_{j,j} \right) = S_{i,j} - 0 + 0 = S_{i,j}.$$

For all other coefficients (s, t) such that $s, t \neq i, j$, we have $\hat{S}_{s,t} = S_{s,t}$. Meaning all rows or columns outside of i, j are left unchanged. We first observe that these last equations, together with our assumption that $j - i > 2$, mean that

$$\hat{S}_{s,t} \geq \hat{S}_{s+1,t} \quad \text{and} \quad \hat{S}_{s,t+1} \geq \hat{S}_{s,t}, \quad \text{for any } s < t$$

so $\hat{S}$ remains an R-matrix. To show uniqueness of the retrieved order, we need $j - i > 1$. Indeed, when $j - i > 1$ all these R constraints are strict, which means that $\hat{S}$ is still a strict R-matrix, hence the desired result. ■

We can extend this result to the case where multiple comparisons are missing.

Proposition 26 *Given pairwise comparisons $C_{s,t} \in \{-1, 0, 1\}$ between items ranked according to their indices, suppose m comparisons indexed $(i_1, j_1), \ldots, (i_m, j_m)$ are missing, i.e., $C_{i_l,j_j} = 0$ for $i = l, \ldots, m$. If the following condition* (26) *holds true,*

$$|s - t| > 1 \text{ for all } s \neq t \in \{i_1, \ldots, i_m, j_1, \ldots, j_m\} \tag{26}$$

then S^{match} defined in (3) *remains* strict-R *and the point score vector remains strictly monotonic.*

Proof Proceed similarly as in the proof of Proposition 13, except that shifts are divided by two. ■

We also get the following corollary.

Corollary 27 *Given pairwise comparisons $C_{s,t} \in \{-1, 0, 1\}$ between items ranked according to their indices, suppose m comparisons indexed $(i_1, j_1), \ldots, (i_m, j_m)$ are either corrupted or missing. If condition* (7) *holds true then S^{match} defined in* (3) *remains* strict-R.

Proof Proceed similarly as the proof of Proposition 13, except that shifts are divided by two for missing comparisons. ■

A.2 Standard theorems and technical lemmas used in spectral perturbation analysis (section 5)

We first recall Weyl's inequality and a simplified version of Davis-Kahan theorem which can be found in (Stewart and Sun, 1990; Stewart, 2001; Yu et al., 2015).

Theorem 28 (Weyl's inequality) *Consider a symmetric matrix A with eigenvalues $\lambda_1, \ldots, \lambda_n$ and $\tilde{A}$ a symmetric perturbation of A with eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$,*

$$\max_i |\tilde{\lambda}_i - \lambda_i| \leq \|\tilde{A} - A\|_2.$$

Theorem 29 (Variant of Davis-Kahan theorem (Corollary 3 Yu et al., 2015)) *Let $A, \tilde{A} \in \mathbb{R}^n$ be symmetric, with eigenvalues $\lambda_1 \leq \ldots \leq \lambda_n$ and $\tilde{\lambda}_1 \leq \ldots \leq \tilde{\lambda}_n$ respectively. Fix $j \in \{1, \ldots, n\}$, and assume that $\min(\lambda_j - \lambda_{j-1}, \lambda_{j+1} - \lambda_j) > 0$, where $\lambda_{n+1} := \infty$ and $\lambda_0 := -\infty$. If $v, \tilde{v} \in \mathbb{R}^n$ satisfy $Av = \lambda_j v$ and $\tilde{A}\tilde{v} = \tilde{\lambda}_j \tilde{v}$, then*

$$\sin \Theta(\tilde{v}, v) \leq \frac{2\|\tilde{A} - A\|_2}{\min(\lambda_j - \lambda_{j-1}, \lambda_{j+1} - \lambda_j)}.$$

Moreover, if $\tilde{v}^T v \geq 0$, then

$$\|\tilde{v} - v\|_2 \leq \frac{2\sqrt{2}\|\tilde{A} - A\|_2}{\min(\lambda_j - \lambda_{j-1}, \lambda_{j+1} - \lambda_j)}.$$

When analyzing the perturbation of the Fiedler vector f, we may always reverse the sign of $\tilde{f}$ such that $\tilde{f}^T f \geq 0$ and obtain

$$\|\tilde{f} - f\|_2 \leq \frac{2\sqrt{2}\|\tilde{L} - L\|_2}{\min(\lambda_2 - \lambda_1, \lambda_3 - \lambda_2)}.$$

Lemma 30 *Let $r > 0$, for every $\mu \in (0, 1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4 n}$, then*

$$\|(\tilde{S} - S)f\|_\infty \leq \frac{3\mu n^{3/2}}{\sqrt{\log n}}$$

with probability at least $1 - 2/n$.

Proof The proof is very much similar to the proof of Lemma 15. Let $R = \tilde{S} - S$. We have

$$R_{ij} = \sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}B_{jk}}{q^2(2p-1)^2} - 1\right).$$

Therefore, let $\delta = Rf$

$$\delta_i = \sum_{j=1}^{n} R_{ij}f_j = \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}B_{jk}}{q^2(2p-1)^2} - 1\right)f_j.$$

Notice that we can arbitrarily fix the diagonal values of R to zeros. Indeed, the similarity between an element and itself should be a constant by convention, which leads to $R_{ii} = \tilde{S}_{ii} - S_{ii} = 0$ for all items i. Hence we could take $j \neq i$ in the definition of d_i, and we can consider B_{ik} independent of B_{jk} in the associated summation.

We first obtain a concentration inequality for each δ_i. We will then use a union bound to bound $\|\delta\|_\infty = \max|\delta_i|$. Notice that

$$\begin{aligned}\delta_i &= \sum_{j=1}^{n}\sum_{k=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}B_{jk}}{q^2(2p-1)^2} - 1\right)f_j \\ &= \sum_{k=1}^{n}\left(\frac{C_{ik}B_{ik}}{q(2p-1)}\sum_{j=1}^{n} C_{jk}\left(\frac{B_{jk}}{q(2p-1)} - 1\right)f_j\right) + \sum_{k=1}^{n}\sum_{j=1}^{n} C_{ik}C_{jk}\left(\frac{B_{ik}}{q(2p-1)} - 1\right)f_j.\end{aligned}$$

The first term is quadratic while the second is linear, both terms have mean zero since the B_{ik} are independent of the B_{jk}. We begin by bounding the quadratic term. Let $X_{jk} = C_{jk}(\frac{1}{q(2p-1)}B_{jk} - 1)f_j$. We have

$$\mathbf{E}(X_{jk}) = f_j C_{jk}(\tfrac{qp-q(1-p)}{q(2p-1)} - 1) = 0,$$

$$\mathbf{var}(X_{jk}) = \frac{f_j^2\,\mathbf{var}(B_{jk})}{q^2(2p-1)^2} = \frac{f_j^2}{q^2(2p-1)^2}(q - q^2(2p-1)^2) \leq \frac{f_j^2}{q(2p-1)^2},$$

$$|X_{jk}| = |f_j||\frac{B_{jk}}{q(2p-1)} - 1| \leq \frac{2|f_j|}{q(2p-1)} \leq \frac{2\|f\|_\infty}{q(2p-1)^2}.$$

From corollary 19 $\|f\|_\infty \leq 2/\sqrt{n}$. Moreover $\sum_{j=0}^{n} f_j^2 = 1$ since f is an eigenvector. Hence, by applying Bernstein inequality we get for any $t > 0$

$$\mathbf{Prob}\left(|\sum_{j=1}^{n} X_{jk}| > t\right) \leq 2\exp\left(\frac{-q(2p-1)^2t^2}{2(1+2t/(3\sqrt{n}))}\right) \leq 2\exp\left(\frac{-q(2p-1)^2t^2n}{2(n+\sqrt{n}t)}\right). \quad (27)$$

The rest of the proof is identical to the proof of Lemma 15, replacing t by $\sqrt{n}t$. ■

Lemma 31 *For every $\mu \in (0,1)$ and n large enough, if $q > \frac{\log^4 n}{\mu^2(2p-1)^4\sqrt{n}}$, then*

$$\|\tilde{f} - f\|_\infty \leq c\frac{\mu}{\sqrt{n\log n}}$$

with probability at least $1 - 2/n$, where c is an absolute constant.

Proof Notice that by definition $\tilde{L}\tilde{f} = \tilde{\lambda}_2\tilde{f}$ and $Lf = \lambda_2 f$. Hence for $\tilde{\lambda}_2 > 0$

$$\begin{aligned}\tilde{f} - f &= \frac{\tilde{L}\tilde{f}}{\tilde{\lambda}_2} - f \\ &= \frac{\tilde{L}\tilde{f} - Lf}{\tilde{\lambda}_2} + \frac{(\lambda_2 - \tilde{\lambda}_2)f}{\tilde{\lambda}_2}.\end{aligned}$$

Moreover

$$\begin{aligned}\tilde{L}\tilde{f} - Lf &= (\mathbf{I} - \tilde{D}^{-1}\tilde{S})\tilde{f} - (\mathbf{I} - D^{-1}S)f \\ &= (\tilde{f} - f) + D^{-1}Sf - \tilde{D}^{-1}\tilde{S}\tilde{f} \\ &= (\tilde{f} - f) + D^{-1}Sf - \tilde{D}^{-1}\tilde{S}f + \tilde{D}^{-1}\tilde{S}f - \tilde{D}^{-1}\tilde{S}\tilde{f} \\ &= (\tilde{f} - f) + (D^{-1}S - \tilde{D}^{-1}\tilde{S})f + \tilde{D}^{-1}\tilde{S}(f - \tilde{f})\end{aligned}$$

Hence

$$(\mathbf{I}(\tilde{\lambda}_2 - 1) + \tilde{D}^{-1}\tilde{S})(\tilde{f} - f) = (D^{-1}S - \tilde{D}^{-1}\tilde{S} + (\lambda_2 - \tilde{\lambda}_2)\mathbf{I})f. \quad (28)$$

Writing S_i the i^{th} row of S and d_i the degree of row i, using the triangle inequality, we deduce that

$$|\tilde{f}_i - f_i| \leq \frac{1}{|\tilde{\lambda}_2 - 1|}\left(|(d_i^{-1}S_i - \tilde{d}_i^{-1}\tilde{S}_i)f| + |\lambda_2 - \tilde{\lambda}_2||f_i| + |\tilde{d}_i^{-1}\tilde{S}_i(\tilde{f} - f)|\right). \quad (29)$$

We will now bound each term separately. Define

$$\begin{aligned}&\text{Denom} = |\tilde{\lambda}_2 - 1|, \\ &\text{Num1} = |(d_i^{-1}S_i - \tilde{d}_i^{-1}\tilde{S}_i)f|, \\ &\text{Num2} = |\lambda_2 - \tilde{\lambda}_2||f_i|, \\ &\text{Num3} = |\tilde{d}_i^{-1}\tilde{S}_i(\tilde{f} - f)|.\end{aligned}$$

Bounding Denom First notice that using Weyl's inequality and equation (21) (*cf.* proof of Theorem 21), we have with probability at least $1 - 2/n$ $|\tilde{\lambda}_2 - \lambda_2| \leq \|L_R\|_2 \leq \frac{c\mu}{\sqrt{\log n}}$. Therefore there exists an absolute constant c such that with probability at least $1 - 2/n$

$$|\tilde{\lambda}_2 - 1| > c.$$

We now proceed with the numerator terms.

Bounding Num2 Using Weyl's inequality, corollary 19 and equation (21) (*cf.* proof of Theorem 21), we deduce that w.h.p.

$$|\lambda_2 - \tilde{\lambda}_2||f_i|| \leq \frac{c\mu}{\sqrt{n\log n}},$$

where c is an absolute constant.

Bounding Num1 We now bound $|d_i^{-1}S_i - \tilde{d}_i^{-1}\tilde{S}_i|$. We have

$$\begin{aligned}|(\tilde{d}_i^{-1}\tilde{S}_i - d_i^{-1}S_i)f| &= |(\tilde{d}_i^{-1}\tilde{S}_i - \tilde{d}_i^{-1}S_i + \tilde{d}_i^{-1}S_i - d_i^{-1}S_i)f| \\ &\leq |\tilde{d}_i^{-1}||(\tilde{S}_i - S_i)f| + |(\tilde{d}_i^{-1} - d_i^{-1})S_i f|.\end{aligned}$$

Using equation (18) from the proof of Theorem 21, we have w.h.p.

$$|\tilde{d}_i^{-1} - d_i^{-1}| \leq \frac{c\mu}{n^2\sqrt{\log n}}.$$

Moreover

$$|\tilde{d}_i^{-1}| \leq |\tilde{d}_i^{-1} - d_i^{-1}| + |d_i^{-1}| \leq \frac{c_1\mu}{n^2\sqrt{\log n}} + \frac{c_2}{n^2} \leq \frac{c}{n^2}$$

w.h.p., where c is an absolute constant. Therefore

$$|(\tilde{d}_i^{-1}\tilde{S}_i - d_i^{-1}S_i)f| \leq \frac{c\mu}{n^2\sqrt{\log n}}|S_i f| + \frac{c}{n^2}|(\tilde{S}_i - S_i)f| \text{ w.h.p.} \tag{30}$$

Using the definition of S and corollary 19, we get

$$|S_i f| \leq \sum_{j=1}^{n} S_{ij} \max_i |f_i| \leq c\frac{n^2}{\sqrt{n}} \leq cn^{3/2}, \tag{31}$$

where c is an absolute constant. Using Lemma 22, we get

$$|(\tilde{S}_i - S_i)f| \leq \frac{3\mu n^{3/2}}{\sqrt{\log n}} \text{ w.h.p.} \tag{32}$$

Combining (30), (31) and (32) we deduce that there exists a constant c such that

$$|(\tilde{d}_i^{-1}\tilde{S}_i - d_i^{-1}S_i)f| \leq \frac{c\mu}{\sqrt{n\log n}} \text{ w.h.p.}$$

Bounding Num3 Finally we bound the remaining term $|\tilde{d}_i^{-1}\tilde{S}_i(\tilde{f} - f)|$. By Cauchy-Schwartz inequality we have,

$$|\tilde{d}_i^{-1}\tilde{S}_i(\tilde{f} - f)| \leq |\tilde{d}_i^{-1}|\|\tilde{S}_i\|_2\|\tilde{f} - f\|_2.$$

Notice that

$$\|\tilde{S}_i\|_2 \leq \|S_i\|_2 + \|\tilde{S}_i - S_i\|_2 \leq \|S_i\|_2 + \|\tilde{S} - S\|_2.$$

Since $\|S_i\|_2^2 \leq \|S_1\|_2^2 \leq \frac{n(n+1)(2n+1)}{6}$ and $q > \frac{\log^4 n}{\mu^2(2p-1)^2\sqrt{n}}$ we deduce from Lemma 17 that w.h.p. $\|\tilde{S}_i\|_2 \leq \frac{c\mu n^{7/4}}{\sqrt{\log n}}$, where c is an absolute constant, for n large enough. Moreover, as shown above, $|\tilde{d}_i^{-1}| \leq \frac{c}{n^2}$ and we also get from Theorem 21 that $\|\tilde{f} - f\|_2 \leq \frac{c\mu}{n^{1/4}\sqrt{\log n}}$ w.h.p. Hence we have

$$|\tilde{d}_i^{-1}\tilde{S}_i(\tilde{f} - f)| \leq \frac{c\mu^2 n^{7/4}}{n^2 n^{1/4}(\log n)} \leq \frac{c\mu}{\sqrt{n\log n}} \text{ w.h.p.},$$

where c is an absolute constant. Combining bounds on the denominator and numerator terms yields the desired result. ■

A.3 Numerical experiments with normalized Laplacian

As shown in figure 5, results are very similar to those of SerialRank with unnormalized Laplacian. We lose a bit of performance in terms of robustness to corrupted comparisons.

A.4 Spectrum of the unnormalized Laplacian matrix

A.4.1 Asymptotic Fiedler value and Fiedler vector

We use results on the convergence of Laplacian operators to provide a description of the spectrum of the unnormalized Laplacian in SerialRank. Following the same analysis as in (Von Luxburg et al., 2008) we can prove that asymptotically, once normalized by n^2, apart from the first and second eigenvalue, the spectrum of the Laplacian matrix is contained in the interval $[0.5, 0.75]$. Moreover, we can characterize the eigenfunctions of the limit Laplacian operator by a differential equation, enabling to have an asymptotic approximation for the Fiedler vector.

Taking the same notations as in (Von Luxburg et al., 2008) we have here $k(x,y) = 1 - |x-y|$. The degree function is

$$d(x) = \int_0^1 k(x,y)d\,\mathbf{Prob}(y) = \int_0^1 k(x,y)d(y)$$

(samples are uniformly ranked). Simple calculations give

$$d(x) = -x^2 + x + 1/2.$$

We deduce that the range of d is $[0.5, 0.75]$. Interesting eigenvectors (i.e., here the second eigenvector) are not in this range. We can also characterize eigenfunctions f and corresponding eigenvalues λ by

$$\begin{aligned}& Uf(x) = \lambda f(x) \quad \forall x \in [0,1] \\ \Leftrightarrow\ & Mdf(x) - Sf(x) = \lambda f(x) \\ \Leftrightarrow\ & d(x)f(x) - \int_0^1 k(x,y)f(y)d(y) = \lambda f(x) \\ \Leftrightarrow\ & f(x)(-x^2 + x + 1/2) - \int_0^1 (1 - |x-y|)f(y)d(y) = \lambda f(x)\end{aligned}$$

Differentiating twice we get

$$f''(x)(1/2 - \lambda + x - x^2) + 2f'(x)(1 - 2x) = 0. \tag{33}$$

The asymptotic expression for the Fiedler vector is then a solution to this differential equation, with $\lambda < 0.5$. Let γ_1 and γ_2 be the roots of $(1/2 - \lambda + x - x^2)$ (with $\gamma_1 < \gamma_2$). We can suppose that $x \in (\gamma_1, \gamma_2)$ since the degree function is nonnegative. Simple calculations show that

$$f'(x) = \frac{A}{(x-\gamma_1)^2(x-\gamma_2)^2}$$

is solution to (33), where A is a constant. Now we note that

$$\begin{aligned}\frac{1}{(x-\gamma_1)^2(x-\gamma_2)^2} &= \frac{1}{(\gamma_1-\gamma_2)^2(\gamma_2-x)^2}+\frac{1}{(\gamma_1-\gamma_2)^2(\gamma_1-x)^2}\\ &\quad -\frac{2}{(\gamma_1-\gamma_2)^3(\gamma_2-x)}+\frac{2}{(\gamma_1-\gamma_2)^3(\gamma_1-x)}.\end{aligned}$$

We deduce that the solution f to (33) satisfies

$$f(x) = B + \frac{A}{(\gamma_1-\gamma_2)^2}\left(\frac{1}{\gamma_1-x}+\frac{1}{\gamma_2-x}\right) - \frac{2A}{(\gamma_1-\gamma_2)^3}\left(\log(x-\gamma_1)-\log(\gamma_2-x)\right),$$

where A and B are two constants. Since f is orthogonal to the unitary function for $x \in (0,1)$, we must have $f(1/2)=0$, hence B=0 (we use the fact that $\gamma_1 = \frac{1-\sqrt{1+4\alpha}}{2}$ and $\gamma_2 = \frac{1+\sqrt{1+4\alpha}}{2}$, where $\alpha = 1/2-\lambda$).

As shown in figure 6 , the asymptotic expression for the Fiedler vector is very accurate numerically, even for small values of n. The asymptotic Fiedler value is also very accurate (2 digits precision for $n=10$, once normalized by n^2).

A.4.2 Bounding the eigengap

We now give two simple propositions on the Fiedler value and the third eigenvalue of the Laplacian matrix, which enable us to bound the eigengap between the second and the third eigenvalues.

Proposition 32 *Given all comparisons indexed by their true ranking, let λ_2 be the Fiedler value of S^{match}, we have*

$$\lambda_2 \leq \frac{2}{5}(n^2+1).$$

Proof Consider the vector x whose elements are uniformly spaced and such that $x^T\mathbf{1}=0$ and $\|x\|_2 = 1$. x is a feasible solution to the Fiedler eigenvalue minimization problem. Therefore,

$$\lambda_2 \leq x^T L x.$$

Simple calculations give $x^TLx = \frac{2}{5}(n^2+1)$. ■

Numerically the bound is very close to the true Fiedler value: $\lambda_2/n^2 \approx 0.39$ and $2/5 = 0.4$.

Proposition 33 *Given all comparisons indexed by their true ranking, the vector $v = [\alpha, -\beta, \ldots, -\beta, \alpha]^T$ where α and β are such that $v^T\mathbf{1}=0$ and $\|v\|_2=1$ is an eigenvector of the Laplacian matrix L of S^{match} The corresponding eigenvalue is $\lambda = n(n+1)/2$.*

Proof Check that $Lv = \lambda v$. ■

A.5 Other choices of similarities

The results in this paper shows that forming a similarity matrix (R-matrix) from pairwise preferences will produce a valid ranking algorithm. In what follows, we detail a few options extending the results of Section 2.2.

A.5.1 Cardinal comparisons

When input comparisons take continuous values between -1 and 1, several choice of similarities can be made. First possibility is to use S^{glm}. An other option is to directly provide $1-\text{abs}(C)$ as a similarity to SerialRank. This option has a much better computational cost.

A.5.2 Adjusting contrast in S^{match}

Instead of providing S^{match} to SerialRank, we can change the "contrast" of the similarity, i.e., take the similarity whose elements are powers of the elements of S^{match}.

$$S^{\text{contrast}}_{i,j} = (S^{\text{match}}_{i,j})^{\alpha}.$$

This construction gives slightly better results in terms of robustness to noise on synthetic datasets.

A.6 Hierarchical Ranking

In a large dataset, the goal may be to rank only a subset of top items. In this case, we can first perform spectral ranking, then refine the ranking of the top set of items using either the SerialRank algorithm on the top comparison submatrix, or another seriation algorithm such as the convex relaxation in (Fogel et al., 2013). This last method also allows us to solve semi-supervised ranking problems, given additional information on the structure of the solution.

Acknowledgments

AA is at CNRS, at the Département d'Informatique at École Normale Supérieure in Paris, INRIA - Sierra team, PSL Research University. The authors would like to acknowledge support from a starting grant from the European Research Council (ERC project SIPA), the MSR-Inria Joint Centre, as well as support from the chaire *Économie des nouvelles données*, the *data science* joint research initiative with the *fonds AXA pour la recherche* and a gift from Société Générale Cross Asset Quantitative Research.

References

D. Achlioptas and F. McSherry. Fast computation of low-rank matrix approximations. *Journal of the ACM*, 54(2), 2007.

Nir Ailon. Active learning ranking from pairwise preferences with almost optimal query complexity. In *NIPS*, pages 810–818, 2011.

J.E. Atkins, E.G. Boman, B. Hendrickson, et al. A spectral algorithm for seriation and the consecutive ones problem. *SIAM J. Comput.*, 28(1):297–310, 1998.

Ed Barbeau. Perron's result and a decision on admissions tests. *Mathematics Magazine*, pages 12–22, 1986.

Avrim Blum, Goran Konjevod, R Ravi, and Santosh Vempala. Semidefinite relaxations for minimum bandwidth and other vertex ordering problems. *Theoretical Computer Science*, 235(1):25–42, 2000.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, pages 324–345, 1952.

Mark Braverman and Elchanan Mossel. Noisy sorting without resampling. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 268–276. Society for Industrial and Applied Mathematics, 2008.

Mihai Cucuringu. Sync-rank: Robust ranking, constrained ranking and rank aggregation via eigenvector and semidefinite programming synchronization. *arXiv preprint arXiv:1504.01070*, 2015.

Jean-Charles de Borda. Mémoire sur les élections au scrutin. 1781.

Nicolas de Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Imprimerie Royale, 1785.

John C. Duchi, Lester W. Mackey, and Michael I. Jordan. On the consistency of ranking algorithms. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 327–334, 2010.

John C. Duchi, Lester Mackey, Michael I. Jordan, et al. The asymptotics of ranking algorithms. *The Annals of Statistics*, 41(5):2292–2323, 2013.

C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. *Proceedings of the Tenth International World Wide Web Conference*, 2001.

Uriel Feige and James R. Lee. An improved approximation ratio for the minimum linear arrangement problem. *Information Processing Letters*, 101(1):26–29, 2007.

Uriel Feige, Prabhakar Raghavan, David Peleg, and Eli Upfal. Computing with noisy information. *SIAM Journal on Computing*, 23(5):1001–1018, 1994.

F. Fogel, R. Jenatton, F. Bach, and A. d'Aspremont. Convex relaxations for permutation problems. *NIPS 2013, arXiv:1306.4805*, 2013.

Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *The Journal of machine learning research*, 4:933–969, 2003.

Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill$^{\text{TM}}$: A bayesian skill rating system. In *Advances in Neural Information Processing Systems*, pages 569–576, 2006.

L. Huang, D. Yan, M.I. Jordan, and N. Taft. Spectral Clustering with Perturbed Data. *Advances in Neural Information Processing Systems (NIPS)*, 2008.

Peter J Huber. Pairwise comparison and ranking: optimum properties of the row sum procedure. *The annals of mathematical statistics*, pages 511–520, 1963.

David R Hunter. MM algorithms for generalized bradley-terry models. *Annals of Statistics*, pages 384–406, 2004.

Kevin G Jamieson and Robert D Nowak. Active ranking using pairwise comparisons. In *NIPS*, volume 24, pages 2240–2248, 2011.

Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial hodge theory. *Mathematical Programming*, 127(1):203–244, 2011.

Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

James P. Keener. The perron-frobenius theorem and the ranking of football teams. *SIAM review*, 35(1):80–93, 1993.

Maurice G. Kendall and B. Babington Smith. On the method of paired comparisons. *Biometrika*, 31(3-4):324–345, 1940.

Claire Kenyon-Mathieu and Warren Schudy. How to rank with few errors. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 95–103. ACM, 2007.

J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46: 604–632, 1999.

J. Kuczynski and H. Wozniakowski. Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. *SIAM J. Matrix Anal. Appl*, 13(4):1094–1122, 1992.

RD Luce. *Individual choice behavior*. Wiley, 1959.

Sahand Negahban, Sewoong Oh, and Devavrat Shah. Iterative ranking from pairwise comparisons. In *NIPS*, pages 2483–2491, 2012.

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. *Stanford CS Technical Report*, 1998.

Arun Rajkumar and Shivani Agarwal. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *Proceedings of the 31st International Conference on Machine Learning*, pages 118–126, 2014.

Thomas L. Saaty. The analytic hierarchy process: planning, priority setting, resources allocation. *New York: McGraw*, 1980.

Thomas L. Saaty. Decision-making with the ahp: Why is the principal eigenvector necessary. *European journal of operational research*, 145(1):85–91, 2003.

William W. Schapire, Robert E. Cohen, and Yoram Singer. Learning to order things. In *Advances in Neural Information Processing Systems 10: Proceedings of the 1997 Conference*, volume 10, page 451. MIT Press, 1998.

Ohad Shamir and Naftali Tishby. Spectral clustering on a budget. In *International Conference on Artificial Intelligence and Statistics*, pages 661–669, 2011.

G.W. Stewart. *Matrix Algorithms Vol. II: Eigensystems*. Society for Industrial Mathematics, 2001.

G.W. Stewart and J. Sun. *Matrix perturbation theory*. Academic Press, 1990.

Sebastiano Vigna. Spectral ranking. *arXiv preprint arXiv:0912.0238*, 2009.

Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

Fabian L. Wauthier, Michael I. Jordan, and Nebojsa Jojic. Efficient ranking from pairwise comparisons. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

Yi Yu, Tengyao Wang, and Richard J. Samworth. A useful variant of the davis–kahan theorem for statisticians. *Biometrika*, 102(2):315–323, 2015.

Ernst Zermelo. Die berechnung der turnier-ergebnisse als ein maximumproblem der wahrscheinlichkeitsrechnung. *Mathematische Zeitschrift*, 29(1):436–460, 1929.

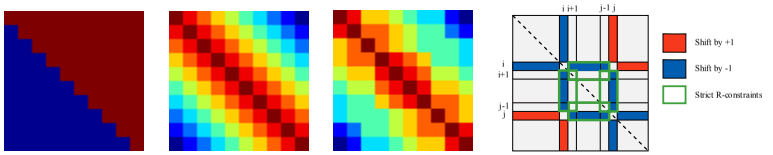


Figure 1: The matrix of pairwise comparisons C *(far left)* when the rows are ordered according to the true ranking. The corresponding similarity matrix S^{match} is a strict R-matrix *(center left)*. The same S^{match} similarity matrix with comparison (3,8) corrupted *(center right)*. With one corrupted comparison, S^{match} keeps enough strict R-constraints to recover the right permutation. In the noiseless case, the difference between all coefficients is at least one and after introducing an error, the coefficients inside the green rectangles still enforce strict R-constraints *(far right)*.

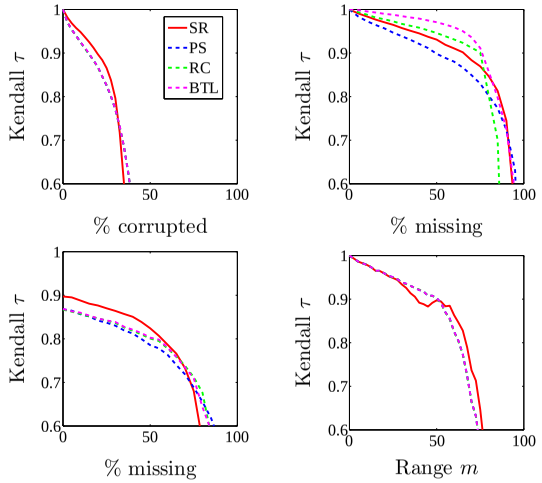


Figure 2: Kendall τ (higher is better) for SerialRank (SR, full red line), point score (PS, (Wauthier et al., 2013) dashed blue line), Rank Centrality (RC (Negahban et al., 2012) dashed green line), and maximum likelihood (BTL (Bradley and Terry, 1952), dashed magenta line). In the first synthetic dataset, we vary the proportion of corrupted comparisons *(top left)*, the proportion of observed comparisons *(top right)* and the proportion of observed comparisons, with 20% of comparisons being corrupted *(bottom left)*. We also vary the parameter m in the second synthetic dataset *(bottom right)*.

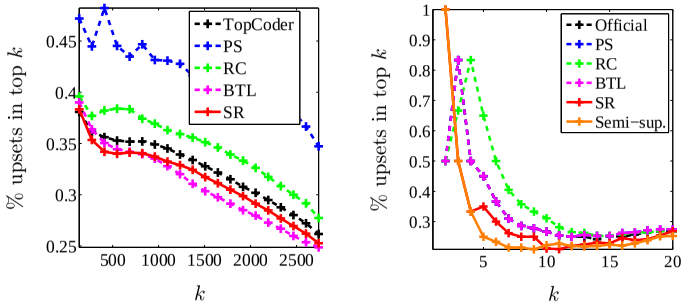


Figure 3: Percentage of upsets (i.e. disagreeing comparisons, lower is better) defined in (25), for various values of k and ranking methods, on TopCoder (*left*) and football data (*right*).

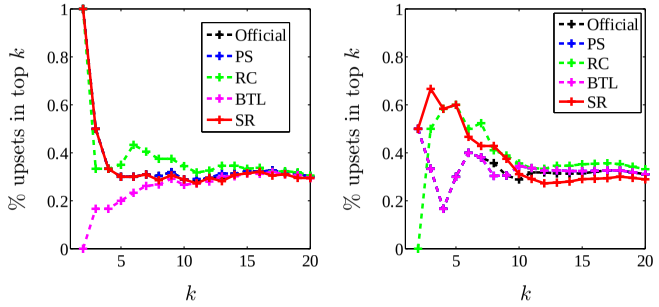


Figure 4: Percentage of upsets (i.e. disagreeing comparisons, lower is better) defined in (25), for various values of k and ranking methods, on England Premier League 2011-2012 season (*left*) and 2012-2013 season (*right*).

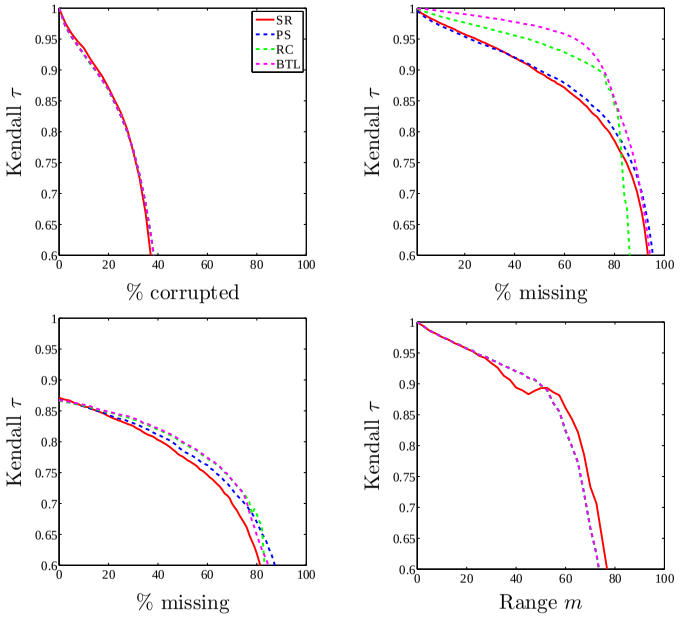


Figure 5: Kendall τ (higher is better) for SerialRank with normalized Laplacian (SR, full red line), row-sum (PS, (Wauthier et al., 2013) dashed blue line), rank centrality (RC (Negahban et al., 2012) dashed green line), and maximum likelihood (BTL (Bradley and Terry, 1952), dashed magenta line). In the first synthetic dataset, we vary the proportion of corrupted comparisons *(top left)*, the proportion of observed comparisons *(top right)* and the proportion of observed comparisons, with 20% of comparisons being corrupted *(bottom left)*. We also vary the parameter m in the second synthetic dataset *(bottom right)*.

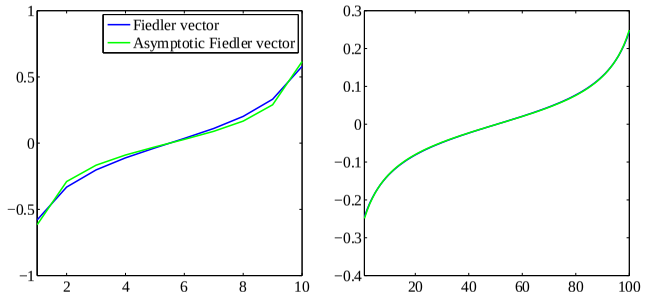


Figure 6: Comparison between the asymptotic analytical expression of the Fiedler vector and the numeric values obtained from eigenvalue decomposition, for $n = 10$ (*left*) and $n = 100$ (*right*).

Journal of Machine Learning Research 17 (2016) 1-34 Submitted 7/11; Revised 5/15; Published 6/16

Sparsity and Error Analysis of Empirical Feature-Based Regularization Schemes

Xin Guo X.GUO@POLYU.EDU.HK
Department of Applied Mathematics
The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong, China
Jun Fan JUNFAN@STAT.WISC.EDU
Department of Statistics
University of Wisconsin-Madison
1300 University Avenue, Madison, WI53706, USA
Ding-Xuan Zhou MAZHOU@CITYU.EDU.HK
Department of Mathematics
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong, China

Editor: Francis Bach

Abstract

We consider a learning algorithm generated by a regularization scheme with a concave regularizer for the purpose of achieving sparsity and good learning rates in a least squares regression setting. The regularization is induced for linear combinations of empirical features, constructed in the literatures of kernel principal component analysis and kernel projection machines, based on kernels and samples. In addition to the separability of the involved optimization problem caused by the empirical features, we carry out sparsity and error analysis, giving bounds in the norm of the reproducing kernel Hilbert space, based on a priori conditions which do not require assumptions on sparsity in terms of any basis or system. In particular, we show that as the concave exponent q of the concave regularizer increases to 1, the learning ability of the algorithm improves. Some numerical simulations for both artificial and real MHC-peptide binding data involving the ℓ^q regularizer and the SCAD penalty are presented to demonstrate the sparsity and error analysis.

Keywords: Sparsity, concave regularizer, reproducing kernel Hilbert space, regularization with empirical features, ℓ^q-penalty, SCAD penalty.

1. Introduction

Kernel methods provide efficient learning algorithms for analyzing nonlinear features, processing complex data, and studying data structures or relations. One may use a (unknown) probability measure ρ_X to model the distribution and structures of data on a compact metric space X (input space) and a Mercer kernel $K : X \times X \to \mathbb{R}$ to quantify by its value $K(x, u)$ similarities between two data points x and u. Then some ideas of kernel methods may be understood (Cristianini and Shawe-Taylor, 2000) in terms of eigenfunctions $\{\phi_i\}$ of the integral operator L_K defined by $L_K(f) = \int_X K(\cdot, x)f(x)d\rho_X(x)$ on the reproducing kernel Hilbert space (RKHS) $(\mathcal{H}_K, \|\cdot\|_K)$ of functions on X induced by the kernel K. These eigenfunctions can be used to represent a feature map and provide insightful, generally nonlinear, features regarding a particular learning problem. As the data distribution ρ_X is unknown, one needs to learn or approximate the features from a data set $\mathbf{x} = \{x_i\}_{i=1}^m \subset X$ and then carries out learning tasks based on the learned data dependent approximate features.

Here we are interested in a class of data dependent features $\{\phi_i^{\mathbf{x}}\}_{i=1}^\infty$ on X, called empirical features, constructed from the data set $\mathbf{x}$ and the kernel K. They have been used in kernel principal component analysis (Schölkopf et al., 1998), kernel ridge regression (Cristianini and Shawe-Taylor, 2000; Hastie et al., 2001), kernel projection machines (Blanchard et al., 2004), and spectral algorithms (Lo Gerfo et al., 2008; Caponnetto and Yao, 2010). They are defined by means of an *empirical integral operator* $L_K^{\mathbf{x}}$ on $\mathcal{H}_K$ expressed as

$$L_K^{\mathbf{x}} f = \frac{1}{m}\sum_{i=1}^m f(x_i)K_{x_i}, \qquad f \in \mathcal{H}_K, \tag{1}$$

where $K_x := K(\cdot, x)$ is a function in $\mathcal{H}_K$ for $x \in X$. It can be seen from the reproducing property $f(x_i) = \langle f, K_{x_i}\rangle_K$ that the operator $L_K^{\mathbf{x}}$ is symmetric, positive and of rank at most m. Denote $\{(\lambda_i^{\mathbf{x}}, \phi_i^{\mathbf{x}})\}_i$ the normalized eigenpairs of $L_K^{\mathbf{x}}$ with (possibly multiple) eigenvalues $\lambda_1^{\mathbf{x}} \ge \lambda_2^{\mathbf{x}} \ge \cdots \ge \lambda_m^{\mathbf{x}} \ge 0 = \lambda_{m+1}^{\mathbf{x}} = \cdots$, then the eigenfunctions $\{\phi_i^{\mathbf{x}}\}_i$ form an orthonormal basis of $\mathcal{H}_K$ and they are called *empirical features*.

In this paper we consider some empirical feature-based regularization schemes in a regression setting and study sparsity of these learning algorithms when the regularizer is a concave function. Here the output space is $Y = \mathbb{R}$. With a sample $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^m \in (X \times Y)^m$, the learning algorithm producing the output function

$$f^{\mathbf{z}} = \sum_{i=1}^\infty c_i^{\mathbf{z}} \phi_i^{\mathbf{x}} \tag{2}$$

is given in terms of its coefficient sequence $c^{\mathbf{z}} = (c_i^{\mathbf{z}})_{i=1}^\infty$ by the regularization scheme

$$c^{\mathbf{z}} = \arg\min_{c \in \ell^2} \left\{ \frac{1}{m}\sum_{i=1}^m \left(\sum_{j=1}^\infty c_j \phi_j^{\mathbf{x}}(x_i) - y_i \right)^2 + \gamma \sum_{j=1}^\infty \Omega(|c_j|) \right\}, \tag{3}$$

where $\gamma > 0$ is a regularization parameter and $\Omega : [0, \infty) \to [0, \infty)$ is a nonzero *concave* function satisfying $\Omega(0) = 0$. We shall show under some regularity assumptions that the above learning algorithm has *strong sparsity* in the sense that with confidence, the number of nonzero coefficients in the expression (2) is of order $O(m^{\theta_{sp}})$ with $0 < \theta_{sp} < 1$, much smaller than the sample size m.

The scheme (3) with special forms of regularizers can be found in the literature of kernel methods. When the regularization on the sequence $c = (c_j)_j$ is replaced by the restriction $c_j = 0$ for $j > N$, the scheme is the kernel principal component regression (Schölkopf et al., 1998) or spectral cut-off algorithm (Lo Gerfo et al., 2008; Caponnetto and Yao, 2010) where detailed error analysis can be found. The case $\Omega(|c|) = |c|^2$ corresponds to the kernel ridge regression (Cristianini and Shawe-Taylor, 2000; Hastie et al., 2001) with error analysis well conducted in a large literature (Caponnetto and De Vito, 2007; Bauer et al., 2007; Smale

and Zhou, 2007). The kernel projection machines can be expressed (Blanchard et al., 2004) by taking Ω to be the indicator function of the set $(0,\infty)$ and $\sum \Omega(|c_j|)$ to be the number of nonzero terms in the sequence c, hence correspond to the classical variable subset selection method. These algorithms were applied and analyzed for classification and regression in (Zwald, 2005; Zwald and Blanchard, 2006; Blanchard and Zwald, 2008).

A main choice of the regularizer in scheme (3) is $\Omega(|c|) = |c|^q$ with $0 < q < 2$. It can be viewed as a kernel version of the classical bridge regression (Frank and Friedman, 1993) which has advantages in some applications. To describe more details, we express the empirical features explicitly in terms of eigenpairs of the kernel (Gramian) matrix $\mathbb{K} := (K(x_i, x_j))_{i,j=1}^m$ (see e.g. Schölkopf et al. (1998); Guo and Zhou (2012)): if $\widehat{\lambda}_i^{\mathbf{x}} > 0$ is the i-th largest eigenvalue of $\mathbb{K}$ with a corresponding normalized eigenvector $\widehat{\mu}_i \in \mathbb{R}^m$, then $\lambda_i^{\mathbf{x}} = \widehat{\lambda}_i^{\mathbf{x}}/m$ and $\phi_i^{\mathbf{x}} = \sum_{j=1}^m (\widehat{\mu}_i)_j K_{x_j}/\sqrt{\widehat{\lambda}_i^{\mathbf{x}}}$. In particular, when $X \subset \mathbb{R}^n$ and K is the linear kernel $K(x,y) = x \cdot y$, we know that $\phi_i^{\mathbf{x}}$ is exactly the i-th principal component of the data matrix $A_{\mathbf{x}} = [x_1, \ldots, x_m]^T \in \mathbb{R}^{m\times n}$ and $\mathbb{K}$ is the kernel matrix $\mathbb{K} = A_{\mathbf{x}} A_{\mathbf{x}}^T$. So the scheme (3) may be viewed as regularized kernel principal component analysis (RKPCA). Moreover, a large statistical literature with the linear kernel on $\mathbb{R}^n$ reveals advantages of various methods (Frank and Friedman, 1993): principal component regression and ridge regression perform well in reducing variances when many variables together collectively effect the response with no small variable subset standing out. In particular, ridge regression (with $q = 2$ in $\Omega(|c|) = |c|^q$) has the best performance when a prior distribution of the regression vector in a Bayesian framework is Gaussian or rotationally invariant setting no preference for any particular directions. A Gaussian process interpretation can be used to understand some advantages of the kernel ridge regression. On the other hand, the variable subset selection method (with $q = 0$) has an optimal performance when the prior distribution puts the entire probability mass on the variable axes, only a few variables have influences on the response, but no information as to which ones is available. Bridge regression may have advantages when the prior distribution is concentrated along some favored directions. It also provides ways for automatic variable selection, for optimizing the power index $q \in (0,2)$ and expanding the model selection criterion by estimating jointly the optimal values of q and γ. As an extension to deal with nonlinear features in RKHSs, it is expected that the kernel bridge regression included in (3) has the same flexibility and some advantages, which will be simulated for real MHC-peptide binding data in subsection 5.2 and discussed in our sparsity and error analysis.

A crucial property of empirical features is their orthogonality with respect to the discrete measure $\frac{1}{m}\sum_{i=1}^m \delta_{x_i}$ stated as $\frac{1}{m}\sum_{i=1}^m \phi_j^{\mathbf{x}}(x_i)\phi_l^{\mathbf{x}}(x_i) = \delta_{j,l}\lambda_j^{\mathbf{x}}$. This is a classical fact and simplifies the empirical error term in (3) as $\frac{1}{m}\sum_{i=1}^m \left(\sum_{j=1}^\infty c_j \phi_j^{\mathbf{x}}(x_i) - y_i\right)^2 = \sum_{i=1}^m \lambda_i^{\mathbf{x}} c_i^2 - 2\sum_{i=1}^m \lambda_i^{\mathbf{x}} S_i^{\mathbf{z}} c_i + \frac{1}{m}\sum_{i=1}^m y_i^2$, where $S_i^{\mathbf{z}}$ is a number defined in terms of the sample $\mathbf{z}$ as

$$S_i^{\mathbf{z}} = \begin{cases} \frac{1}{m\lambda_i^{\mathbf{x}}}\sum_{j=1}^m y_j \phi_i^{\mathbf{x}}(x_j), & \text{if } \lambda_i^{\mathbf{x}} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

This simplification easily implies that the optimization problem (3) can be solved separately for each coefficient c_i, and $c_i^{\mathbf{z}} = 0$ for $i \geq m+1$. So we may replace the summations in (2) and (3) by those up to m (we keep them for the convenience of the proofs).

Theorem 1 *Let $\Omega : [0,\infty) \to [0,\infty)$, $\gamma > 0$ and $\mathbf{z} \in (X \times Y)^m$. Then a sequence $c^{\mathbf{z}} = (c_i^{\mathbf{z}})_{i=1}^\infty$ is a solution to (3) if and only if for each i, $c_i^{\mathbf{z}}$ is a minimizer of the univariate function defined by*

$$h_i(c) = h_{\lambda_i^{\mathbf{x}}, S_i^{\mathbf{z}}, \gamma, \Omega}(c) = \lambda_i^{\mathbf{x}}(c - S_i^{\mathbf{z}})^2 + \gamma\Omega(|c|), \qquad c \in \mathbb{R}. \tag{5}$$

2. Main Results on Sparsity and Error Analysis

The main purpose of this paper is to show that both strong sparsity and fast learning rate can be achieved by the learning algorithm (2) when the regularizing function Ω in (3) is concave. We describe the main ideas in this section and will provide detailed general analysis in Section 4 while some numerical simulations for both artificial and real data will be presented in Section 5.

2.1 Concave regularizing functions

The concavity of regularizing functions plays a central role in achieving sparsity in this paper. It has the following nice property.

Theorem 2 *If $\Omega : [0,\infty) \to [0,\infty)$ is a nonzero continuous concave function satisfying $\Omega(0) = 0$, then $\Omega(1) > 0$, and that*

$$\Omega(c) \geq \Omega(1)c, \qquad \forall c \in (0,1] \tag{6}$$

and

$$\Omega(c) \leq \Omega(1)c, \qquad \forall c \in [1,\infty). \tag{7}$$

Theorem 2 is part of Proposition 10 in Section 3 which will give more properties for concave regularizing functions.

Note that (6) is a lower bound for Ω on $(0,1]$. Our error bounds will be presented by means of the asymptotic behavior of the concave regularizing function Ω near the origin, which is characterized by a concave exponent $q \in [0,1]$.

Definition 3 *We say that a concave regularizing function Ω has a concave exponent $q \in [0,1]$ if there is a positive constant C_Ω^* such that*

$$\Omega(c) \leq C_\Omega^* c^q, \qquad \forall c \in (0,1]. \tag{8}$$

Theorem 2 tells us that the concave exponent q in (8) is at most 1. We also know from Proposition 10 in Section 3 that (8) is always true with $q = 0$ and $C_\Omega^* = \Omega(1)$. Sharper error bounds with better q are possible. The following are two such families of concave regularizing functions: ℓ^q-regularizer ($0 < q \leq 1$) which is well studied for bridge regression (Frank and Friedman, 1993; Fu and Knight, 2000; Liu et al., 2007; Xu et al., 2012), and SCAD penalties (Fan and Li, 2001).

Example 1 *Let $0 < q \leq 1$ and $\Omega : [0,\infty) \to [0,\infty)$ be the ℓ^q-regularizer given by $\Omega(c) = c^q$. Then (8) is satisfied with $C_\Omega^* = 1$.*

Example 2 *Let $b > 2$ and $\Omega : [0,\infty) \to [0,\infty)$ be a SCAD penalty given as a concave continuous function by $\Omega(0) = 0$ and*

$$\Omega'(c) = \begin{cases} 1, & \text{for } 0 < c < 1, \\ \frac{c-b}{1-b}, & \text{for } 1 < c < b, \\ 0, & \text{for } c > b. \end{cases}$$

Then $\Omega(c) = c$ for $c \in [0,1]$, $\Omega(c) = \frac{1+b}{2} - \frac{(c-b)^2}{2(b-1)}$ for $c \in (1,b]$ and $\Omega(c) = \frac{1+b}{2}$ for $c \in (b,\infty)$. Hence (8) is satisfied with $q = 1$ and $C^*_\Omega = 1$. Moreover, we have $\Omega(c) \le \frac{1+b}{2}$ for every $c \in [1,\infty)$.

In our results for sparsity and error analysis, we shall use a general power index $q \in [0,1]$ instead of the universal choice of $q = 0$.

2.2 Sparsity and learning rates

Throughout the paper, we assume that the sample set $\mathbf{z}$ is drawn independently according to a Borel probability measure ρ on $X \times Y$ and that for some constant $M > 0$, $|y| \le M$ almost surely. The regression function in our regression setting is defined as a function f_ρ on X given by

$$f_\rho(x) = \int_Y y d\rho(y|x), \qquad x \in X,$$

where $\rho(\cdot|x)$ is the conditional measure induced by ρ at $x \in X$. The regularity assumption we shall take for the regression function is

$$f_\rho = L_K^r(g_\rho) \quad \text{for some } r > 0 \text{ and } g_\rho \in \mathcal{H}_K. \tag{9}$$

Here L_K is a compact, self-adjoint and positive operator on $\mathcal{H}_K$ having eigenpairs $\{(\lambda_i, \phi_i)\}_i$ with the eigenvalues $\{\lambda_i\}$ forming a nonincreasing sequence tending to 0 and and eigenfunctions $\{\phi_i\}$ an orthonormal basis of $\mathcal{H}_K$. Its r-th power L_K^r is given by $L_K^r(\sum_i c_i\phi_i) = \sum_i c_i\lambda_i^r\phi_i$ and assumption (9) means $f_\rho = \sum_i d_i\lambda_i^r\phi_i$ for some sequence $\{d_i\} \in \ell^2$ representing $g_\rho = \sum_i d_i\phi_i$. The exponent r in (9) measures the decay of the coefficients $\{d_i\lambda_i^r\}$ of f_ρ with respect to the orthonormal basis $\{\phi_i\}$ of $\mathcal{H}_K$, and thereby the regularity of the regression function f_ρ.

Let us illustrate our general analysis for strong sparsity and learning rates by two special cases, derived from Corollary 16 (with $\alpha_1 = \alpha_2 = \alpha$) and Corollary 17 (with $\beta_1 = \beta_2 = \beta$) in Section 4, for which the eigenvalues of the integral operator L_K decay polynomially or exponentially.

Theorem 4 *Assume (9) with $r > \frac{1}{2}$, and that Ω has a concave exponent $q \in [0,1]$ with (8) valid. Suppose that for some positive constants D_1, D_2 and α, the eigenvalues $\{\lambda_i\}$ of L_K decay polynomially as*

$$D_1 i^{-\alpha} \le \lambda_i \le D_2 i^{-\alpha}, \qquad \forall i \in \mathbb{N} \tag{10}$$

with $2\alpha\max\{r,1\} > 1$. Let $0 < \delta < 1$. If we choose

$$\gamma = C_1(D_2/\lambda_1)^{r+1}\left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\frac{1+r}{1+2r}}, \tag{11}$$

then with confidence $1-\delta$ we have

$$c_i^{\mathbf{z}} = 0, \qquad \forall\ m^{\theta_{sp}} + 1 \le i \le m \quad \text{with } \theta_{sp} = \frac{1}{\alpha(1+2r)} < 1 \tag{12}$$

and

$$\|f^{\mathbf{z}} - f_\rho\|_K \le C_2\left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\theta_{rate}}, \qquad \theta_{rate} = \frac{\alpha\min\{4r, 4r(2-q)\} - 2(2-q)}{4(2r+1)(2-q)\alpha},$$

where C_1 and C_2 are constants independent of m or δ (to be specified in the proof).

The eigenvalue decay condition (10) is typical for Sobolev smooth kernels on domains in Euclidean spaces, with the power index α depending on the smoothness of the kernel (Reade, 1984).

The regularity assumptions (9) and (10) impose restrictions on the concave exponent $q \in [0,1]$. To see this, we express $g_\rho = \sum_i d_i\phi_i$ with $(d_i)_i \in \ell^2$ and $f_\rho = L_K^r(g_\rho) = \sum_i \lambda_i^r d_i\phi_i$. A natural requirement for f_ρ corresponding to the ℓ^q-regularizer is $(\lambda_i^r d_i)_i \in \ell^q$. Imposing this uniformly with respect to the coefficient sequence $(d_i)_i$ is the same as the boundedness from ℓ^2 to ℓ^q of the diagonal operator D_{λ^r} associated with the fixed non-indreasing sequence $(\lambda_i^r)_i$. This problem together with asymptotic behaviors of the entropy numbers of D_{λ^r} has been widely studied in the literature of function spaces and approximation theory (Edmunds and Triebel, 1996; Kühn, 2008) and the boundedness can be characterized by the condition

$$(\lambda_i^r)_i \in \ell^s \text{ with } \frac{1}{s} = \frac{1}{q} - \frac{1}{2}. \tag{13}$$

Under the eigenvalue decay assumption (10), the characterization condition (13) is equivalent to $\sum_{i=1}^\infty i^{-\alpha r s} = \sum_{i=1}^\infty i^{-\frac{2\alpha r q}{2-q}} < \infty$, which can be stated as

$$q > \frac{2}{2\alpha r + 1}. \tag{14}$$

Thus the concave exponent q is tailored to the regularity assumption and the eigenvalue decay, and a larger regularity index r leads to a wider range of the concave exponent q.

Combining the regularity assumption (9) and the eigenvalue decay condition (10) has been an approach for error analysis of learning algorithms. In particular, the minimax rates of convergence in the $L^2_{\rho_X}$ metric was derived in (Caponnetto and De Vito, 2007) under these conditions with the restrictions $\alpha > 1$ and $0 < r \le \frac{1}{2}$. Moreover, the well-known regularized least squares regression (RLS) scheme

$$f^{\mathbf{z}} = \arg\min_{f\in\mathcal{H}_K}\left\{\frac{1}{m}\sum_{i=1}^m (f(x_i) - y_i)^2 + \gamma\|f\|_K^2\right\} \tag{15}$$

achieves these rates in probability as $\|f^{\mathbf{z}} - f_\rho\|^2_{L^2_{\rho_X}} = O(m^{-\frac{\alpha(2r+1)}{\alpha(2r+1)+1}})$. Error estimates in the $\mathcal{H}_K$ metric provide error analysis for the distribution mismatch problem (where the distribution for predictions might be different from the sampling distribution ρ_X) and

for sampling processes with nonidentical distributions (Smale and Zhou, 2009; Zhou, 2003). Such estimates for the RLS algorithm (15) were conducted in (Smale and Zhou, 2007; Bauer et al., 2007) where the learning rates are $\|f^{\mathbf{z}}-f_\rho\|_K = O(m^{-\frac{r}{2r+2}})$ under the same restriction $0 < r \le \frac{1}{2}$ and the maximum exponent is $\frac{1}{6}$ when $r = \frac{1}{2}$. The maximum exponent for the RLS algorithm (15) cannot be improved further for $r > \frac{1}{2}$ and this is called a saturation effect in the theory of inverse problems (Bauer et al., 2007).

As pointed out in (Bauer et al., 2007; Lo Gerfo et al., 2008), spectral cut-off algorithms do not suffer from the saturation phenomenon. Theorem 4 confirms this advantage for the algorithm (2) in the range $r > \frac{1}{2}$ (the range $0 < r \le \frac{1}{2}$ is covered by Corollary 16 in Section 4). To be specific, let $\frac{1}{2} < q \le 1$ and $r \ge \frac{1}{4q-2}$. Then the power index θ_{rate} for the learning rate in Theorem 4 is

$$\theta_{rate} = \frac{2r\alpha - 2 + q}{2(2r+1)(2-q)\alpha} \tag{16}$$

which becomes larger as the regularity index r increases, and can be arbitrarily close to $\frac{1}{2}$ when r is large enough (f_ρ is smooth enough) and $q = 1$. This applied to the case when Ω is the SCAD penalty given in Example 2. Even in the range $0 < q \le \frac{1}{2}$, for a sufficiently large r, the power index θ_{rate} in Theorem 4 can be arbitrarily close to $\frac{1}{2(2-q)}$.

The estimate (12) for sparsity in Theorem 4 tells us that with confidence, the output function $f^{\mathbf{z}} = \sum c_i^{\mathbf{z}} \phi_i^{\mathbf{z}}$ has at most $m^{\theta_{sp}}$ nonzero coefficients with a sparsity exponent $\theta_{sp} < 1$, a small proportion of the m coefficients in the expression (2). Moreover, θ_{sp} decreases, leading to better sparsity, as r increases. Note that by our analysis, the restriction (14) is the only influence of the concave exponent q for the sparsity.

Theorem 5 *Assume (9) with $r > \frac{1}{2}$, and that Ω has a concave exponent $q \in [0,1]$ with (8) valid. Suppose that for some positive constants D_1, D_2 and β, the eigenvalues $\{\lambda_i\}$ of L_K decay exponentially as*

$$D_1\beta^{-i} \le \lambda_i \le D_2\beta^{-i}, \qquad \forall i \in \mathbb{N}. \tag{17}$$

Let $0 < \delta < 1$. If we choose γ as (11), then with confidence $1-\delta$ we have

$$c_i^{\mathbf{z}} = 0, \qquad \forall\ \frac{\log(m+1)}{(1+2r)\log\beta} + 1 \le i \le m \tag{18}$$

and

$$\|f^{\mathbf{z}} - f_\rho\|_K \le C_2 \left(\log\frac{4m}{\delta}\right)^{2r+1} m^{-\theta_{rate}}, \quad \theta_{rate} = \min\left\{\frac{r}{(2-q)(1+2r)}, \frac{(2-q)r}{(2-q)(1+2r)}\right\},$$

where C_2 is a constant independent of m or δ (to be specified in the proof).

Remark 6 *The eigenvalue decay condition (17) is typical for analytic kernels on domains in Euclidean spaces (Reade, 1984). When the regularity index r is large enough, the power index θ_{rate} for the learning rate is $\frac{1}{2(2-q)} - \epsilon$ with an arbitrarily small $\epsilon > 0$. So the learning rate depends on the concave exponent q, better as q increases. On the other hand, (18) tells us that with confidence, the output function $f^{\mathbf{z}} = \sum c_i^{\mathbf{z}} \phi_i^{\mathbf{z}}$ has at most $\frac{\log(m+1)}{(1+2r)\log\beta}$ nonzero coefficients, a logarithmic proportion of the m coefficients in the expression (2).*

2.3 Minimax lower bound

The learning rate stated in Theorem 4 is close to be optimal when r is large. One might use some existing methods for dealing with the $L^2_{\rho_X}$ error in the literature (Yang and Barron, 1999; Bauer et al., 2007; Caponnetto and De Vito, 2007; DeVore et al., 2004; Suzuki et al., 2012; Raskutti et al., 2012; Steinwart et al., 2009) to give lower bounds. Here we focus on the error in the $\mathcal{H}_K$-metric and present a minimax lower bound. Denote $\kappa = \max_{x\in X}\sqrt{K(x,x)}$.

Definition 7 *Let $\mathcal{P}(\alpha, r, M, R, D_1, D_2)$ be the set of all Borel probability measures ρ on $X \times Y$ such that the regularity assumption (9) is satisfied with $\|g_\rho\|_K \le R$, (10) holds true, and the conditional measure $\rho(\cdot|x)$ is supported on $[-M, M]$ for almost all $x \in X$.*

Theorem 8 *Let α, r, R, D_1, D_2 be positive constants and $M \ge 4\kappa^{r+\frac{1}{2}}R$. Let $f^{\mathbf{z}} \in \mathcal{H}_K$ be the output of an arbitrary learning algorithm based on the sample $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^m$. Then for every $0 < \delta < 1$, there exists a positive constant $C_{\delta,\alpha,r,M,R,D_1,D_2}$ such that*

$$\lim_{m\to\infty}\inf_{f^{\mathbf{z}}}\sup_{\rho\in\mathcal{P}(\alpha,r,M,R,D_1,D_2)} \mathbb{P}_{\mathbf{z}\sim\rho^m}\left\{\|f^{\mathbf{z}} - f_\rho\|_K \ge C_{\delta,\alpha,r,M,R,D_1,D_2} m^{-\frac{\alpha r}{\alpha(1+2r)+1}}\right\} \ge 1-\delta.$$

The proof of Theorem 8 follows from a more general result to be given in Appendix B. The power index $\frac{\alpha r}{\alpha(2r+1)+1}$ for the minimax lower bound stated in Theorem 8 corresponds to the upper bound index (16) in Theorem 4 for a smoother regularity class with $r' = r + \frac{2-q}{2\alpha}$. This shows the gap between our upper bound and the minimax lower bound. It would be interesting to derive minimax rates of convergence in the $\mathcal{H}_K$-metric which can be achieved by the learning algorithm (2) with $\Omega(c) = c^q$ for $0 < q \le 1$.

2.4 Connections to ridge regression and some other learning algorithms

The classical RLS algorithm (15) can be stated as the scheme (2) by taking the regularizer $\Omega(c) = c^2$ corresponding to the ridge regression. This follows from a representer theorem for (15), the identities $\mathrm{span}\{K_{x_j}\}_{j=1}^m = \mathrm{span}\{\phi_j^{\mathbf{x}}\}_{j=1}^m$ and $\left\|\sum_{j=1}^\infty c_j\phi_j^{\mathbf{x}}\right\|_K^2 = \sum_{j=1}^\infty |c_j|^2$.

The regularizer $\Omega(c) = c^q$ with $0 < q \le 2$ correspond to the bridge regression. When $1 < q \le 2$, this regularizer is convex instead of being concave. It has the special property that $\Omega'_+(0) = 0$ where $\Omega'_+(c)$ denotes the right-side derivative of Ω at $c \in [0,\infty)$. This leads to the observation that sparsity is hardly achieved for the learning algorithm (2) associated with such a convex regularizer.

Theorem 9 *Let $\Omega : [0,\infty) \to [0,\infty)$, $\gamma > 0$ and $\Omega(0) = 0$. If $\Omega'_+(0) = 0$, then for each i, $c_i^{\mathbf{z}}$ vanishes if and only if either $\lambda_i^{\mathbf{x}} = 0$ or $S_i^{\mathbf{z}} = 0$.*

An elastic net learning algorithm (Zou and Hastie, 2005) can be introduced by taking the regularizer in (3) as

$$\Omega^{en}(c) = c + \zeta c^2, \tag{19}$$

where $\zeta > 0$ is an elastic net parameter controlling the proportion of the ℓ^2-norm square in the regularizer Ω^{en}. Though the regularizer Ω^{en} is strictly convex, it does not satisfy the assumption $\Omega'_+(0) = 0$ in Theorem 9. When ζ is small, this regularizer is actually close to

the ℓ^1-penalty. Hence we would expect that the corresponding learning algorithm with a strictly convex regularizer has strong sparsity. This is beyond the discussion in this paper.

Let us mention that the learning scheme (2) is closely related to spectral algorithms (Lo Gerfo et al., 2008; Caponnetto and Yao, 2010) which can be stated in terms of the empirical features $\{\phi_j^{\mathbf{x}}\}_{j=1}^m$ and a filter function $g_\gamma : [0,1] \to \mathbb{R}$ as

$$f^{\mathbf{z}} = \sum_{j=1}^m \sqrt{\lambda_j^{\mathbf{x}}/m} \left(\sum_{i=1}^m (\widehat{\mu}_j)_i y_i \right) g_\gamma(\lambda_j^{\mathbf{x}}) \phi_j^{\mathbf{x}},$$

where $\{(m\lambda_j^{\mathbf{x}}, \widehat{\mu}_j)\}$ are the normalized eigenpairs of the kernel matrix $\mathbb{K}$.

Our analysis relies heavily on the special form of the least squares loss, as seen from Theorem 1. It would be interesting to establish similar analysis for schemes associated with other loss functions such as those in the minimum error entropy principle, at least when the scaling parameter is large (Hu et al., 2015).

3. Properties of Concave Regularizing Functions

In this section we give some properties of concave regularizing functions, and then estimate the solution $c^{\mathbf{z}}$ to (3) by means of the explicit expression stated in Theorem 1.

Proposition 10 *Let $\Omega : [0,\infty) \to [0,\infty)$ be a nonzero continuous concave function satisfying $\Omega(0) = 0$. Then it has the following properties.*

(a) The function Ω is nondecreasing on $[0,\infty)$, and $\Omega(c) > 0$ for $c \in (0,\infty)$. The right-hand derivative Ω'_+ is well defined, nonincreasing, finite, and nonnegative on $(0,\infty)$. At the origin, $\Omega'_+(0) \in (0,\infty]$.

(b) We have $\Omega(c) \ge \Omega(1)c$ for $c \in [0,1]$, and $\Omega(c) \le \Omega(1)c$ for $c \in [1,\infty)$.

(c) There holds $\Omega(a+b) \le \Omega(a) + \Omega(b)$ for any $a, b > 0$.

(d) The positive function $\frac{\Omega(c)}{c}$ defined on $(0,\infty)$ is nonincreasing and satisfies $\lim_{c\to 0^+} \frac{\Omega(c)}{c} = \Omega'_+(0)$.

(e) The positive function $\frac{\Omega(c)}{c^2}$ defined on $(0,\infty)$ is continuous and strictly decreasing from $\lim_{c\to 0^+} \frac{\Omega(c)}{c^2} = +\infty$ to $\lim_{c\to\infty} \frac{\Omega(c)}{c^2} = 0$.

Proposition 10 will be proved in Appendix C.

For our analysis, we need the following two auxiliary functions.

Definition 11 *Define an auxiliary function $\Omega^* : (0,\infty) \to (0,\infty)$ of a positive function Ω as*

$$\Omega^*(\lambda) = \inf_{c\in(0,\infty)} \left\{ \frac{\Omega(c)}{c} + \lambda c \right\}, \qquad \lambda \in (0,\infty).$$

Define another auxiliary function $\widetilde{\Omega} : (0,\infty) \to (0,\infty)$ as

$$\widetilde{\Omega}(\lambda) = \arg\sup \left\{ c \in (0,\infty) : \frac{\Omega(c)}{c^2} \ge \lambda \right\}, \qquad \lambda \in (0,\infty).$$

Remark 12 *The value $-\Omega^*(\lambda)$ is exactly equal to the value at the point $-\lambda$ of the conjugate function of $\frac{\Omega(c)}{c}$ defined in the literature of optimization.*

We can now estimate the solution $c^{\mathbf{z}}$ to (3) in terms of $S_i^{\mathbf{z}}, \lambda_i^{\mathbf{x}}$ and γ, by means of the explicit expression stated in Theorem 1.

Theorem 13 *Let $\gamma > 0$ and $\Omega : [0,\infty) \to [0,\infty)$ be a nonzero continuous concave function satisfying $\Omega(0) = 0$.*

(a) Both functions Ω^* and $\widetilde{\Omega}$ are well-defined and positive on $(0,\infty)$. The function Ω^* is nondecreasing while $\widetilde{\Omega}$ is non-increasing.

(b) Let $i \in \mathbb{N}$. If

$$|S_i^{\mathbf{z}}| < \frac{\Omega^*(\frac{\lambda_i^{\mathbf{x}}}{\gamma})}{2\frac{\lambda_i^{\mathbf{x}}}{\gamma}}, \tag{20}$$

then $c_i^{\mathbf{z}} = 0$. If $|S_i^{\mathbf{z}}| > \frac{\Omega^*(\frac{\lambda_i^{\mathbf{x}}}{\gamma})}{2\frac{\lambda_i^{\mathbf{x}}}{\gamma}}$, then $c_i^{\mathbf{z}}$ has the same sign as $S_i^{\mathbf{z}}$ and satisfies $|S_i^{\mathbf{z}}| - \widetilde{\Omega}\left(\frac{\lambda_i^{\mathbf{x}}}{\gamma}\right) \le |c_i^{\mathbf{z}}| \le |S_i^{\mathbf{z}}|$.

(c) Let $d^{\mathbf{x}} \le m$ be the rank of the Gramian matrix $\mathbb{K}$. Then $\lambda_i^{\mathbf{x}} = 0$ if and only if $i > d^{\mathbf{x}}$. Hence $c_i^{\mathbf{z}} = 0$ for $i > d^{\mathbf{x}}$.

Proof (a) The first statement follows easily from the definitions of the auxiliary functions and Proposition 10.

(b) Since $\gamma > 0$, when $\lambda_i^{\mathbf{x}} = 0$ or $S_i^{\mathbf{z}} = 0$, our statement follows from Theorem 1. So we consider the case that $\lambda_i^{\mathbf{x}} > 0$ and $S_i^{\mathbf{z}} \ne 0$. By symmetry we only need to prove our statement for the case $S_i^{\mathbf{z}} > 0$.

With $\lambda_i^{\mathbf{x}} > 0$ and $S_i^{\mathbf{z}} > 0$, we find that the left-side derivative of the function h_i is $(h_i)'_-(c) = 2\lambda_i^{\mathbf{x}}(c - S_i^{\mathbf{z}}) - \gamma\Omega'_+(|c|) < 0$ for $c \in (-\infty, 0]$, hence all its possible minimizers are achieved on $[0,\infty)$. Let us consider the difference function $h_i(c) - h_i(0)$ for $c > 0$ and factorize it as

$$h_i(c) - h_i(0) = c g_i(c), \quad \text{where} \quad g_i(c) := \gamma\frac{\Omega(c)}{c} + \lambda_i^{\mathbf{x}} c - 2\lambda_i^{\mathbf{x}} S_i^{\mathbf{z}}. \tag{21}$$

If $|S_i^{\mathbf{z}}| < \frac{\Omega^*(\frac{\lambda_i^{\mathbf{x}}}{\gamma})}{2\frac{\lambda_i^{\mathbf{x}}}{\gamma}}$, then $\inf_{c>0} g_i(c) > 0$ which implies $h_i(c) - h_i(0) = c g_i(c) > 0$ for every $c > 0$. Hence in this case h_i has the only minimizer at $0 = c_i^{\mathbf{z}}$.

If $|S_i^{\mathbf{z}}| > \frac{\Omega^*(\frac{\lambda_i^{\mathbf{x}}}{\gamma})}{2\frac{\lambda_i^{\mathbf{x}}}{\gamma}}$, then $\gamma \inf_{c>0} \left\{ \frac{\Omega(c)}{c} + \frac{\lambda_i^{\mathbf{x}}}{\gamma} c \right\} < 2\lambda_i^{\mathbf{x}} S_i^{\mathbf{z}}$ meaning that $\inf_{c>0} g_i(c) < 0$. It follows that a minimizer c_* of the function g_i on $(0,\infty)$ satisfies $g_i(c_*) < 0$. Hence $h_i(c_*) - h_i(0) = c_* g_i(c_*) < 0$. So 0 is not a minimizer of h_i.

Since Ω is nondecreasing on $[0,\infty)$, we know that h_i is strictly increasing on $(S_i^{\mathbf{z}}, \infty)$. Hence the minimizer $c_i^{\mathbf{z}}$ of h_i satisfies $0 < c_i^{\mathbf{z}} \le S_i^{\mathbf{z}}$. We also know from $h_i(c_i^{\mathbf{z}}) \le h_i(S_i^{\mathbf{z}})$ that

$$h_i(c_i^{\mathbf{z}}) = \lambda_i^{\mathbf{x}} (c_i^{\mathbf{z}} - S_i^{\mathbf{z}})^2 + \gamma\Omega(c_i^{\mathbf{z}}) \le h_i(S_i^{\mathbf{z}}) = \lambda_i^{\mathbf{x}} (S_i^{\mathbf{z}} - S_i^{\mathbf{z}})^2 + \gamma\Omega(S_i^{\mathbf{z}}) = \gamma\Omega(S_i^{\mathbf{z}}).$$

Express $S_i^{\mathbf{z}}$ as $S_i^{\mathbf{z}} - c_i^{\mathbf{z}} + c_i^{\mathbf{z}}$. Proposition 10 (c) yields $\Omega(S_i^{\mathbf{z}}) = \Omega(S_i^{\mathbf{z}} - c_i^{\mathbf{z}} + c_i^{\mathbf{z}}) \le \Omega(c_i^{\mathbf{z}}) + \Omega(S_i^{\mathbf{z}} - c_i^{\mathbf{z}})$. It follows that

$$\lambda_i^{\mathbf{x}} \left(c_i^{\mathbf{z}} - S_i^{\mathbf{z}}\right)^2 \le \gamma \Omega(S_i^{\mathbf{z}} - c_i^{\mathbf{z}}).$$

Therefore,

$$\frac{\Omega(S_i^{\mathbf{z}} - c_i^{\mathbf{z}})}{\left(S_i^{\mathbf{z}} - c_i^{\mathbf{z}}\right)^2} \ge \frac{\lambda_i^{\mathbf{x}}}{\gamma}.$$

By the definition of the function $\widetilde{\Omega}$, this implies that $S_i^{\mathbf{z}} - c_i^{\mathbf{z}} \le \widetilde{\Omega}\left(\frac{\lambda_i^{\mathbf{x}}}{\gamma}\right)$. This proves the range of $c_i^{\mathbf{z}}$ and verifies out second statement.

(c) It is well-known (e. g. Guo and Zhou (2012)) that the first $d^{\mathbf{x}}$ eigenvalues of the matrix $\mathbb{K}$ are given by $\{m\lambda_i^{\mathbf{x}}\}_{i=1}^{d^{\mathbf{x}}}$ while $\lambda_i^{\mathbf{x}} = 0$ for $i \ge d^{\mathbf{x}} + 1$. So $\lambda_i^{\mathbf{x}} = 0$ if and only if $i > d^{\mathbf{x}}$. In this case, condition (20) is satisfied and by the conclusion in part (b), $c_i^{\mathbf{z}} = 0$. The proof of Theorem 13 is thus complete. ■

4. General Analysis for Sparsity and Error Bounds

In this section we present a general result on sparsity and error bounds for the learning algorithm (2) generated by the regularization scheme (3) based on empirical features and concave regularizing functions. To this end, we need the following bounds for the auxiliary functions Ω^* and $\widetilde{\Omega}$.

Lemma 14 *If $\Omega : [0,\infty) \to [0,\infty)$ is a nonzero continuous concave function satisfying $\Omega(0) = 0$, then there exists a positive constant $C_{\Omega,1}$ such that*

$$\Omega^*(\lambda) \ge C_{\Omega,1} \min\{\sqrt{\lambda}, 1\}, \qquad \forall \lambda > 0. \tag{22}$$

If moreover, Ω has a concave exponent $q \in [0,1]$ with (8) valid, then there exists a positive constant $C_{\Omega,2}$ such that

$$\widetilde{\Omega}(\lambda) \le C_{\Omega,2} \max\left\{\left(\frac{1}{\lambda}\right)^{1/(2-q)}, \frac{1}{\lambda}\right\} \qquad \forall \lambda > 0. \tag{23}$$

Proof For $c \in (0,1]$, we apply Proposition 10 (d) and find

$$\frac{\Omega(c)}{c} + \lambda c \ge \Omega(1) + \lambda c \ge \Omega(1) \ge \Omega(1) \min\{\sqrt{\lambda}, 1\}.$$

For $c \in (1,\infty)$, we have $\Omega(c) \ge \Omega(1)$. Then $\frac{\Omega(c)}{c} + \lambda c \ge \frac{\Omega(1)}{c} + \lambda c \ge 2\sqrt{\Omega(1)\lambda} \ge 2\sqrt{\Omega(1)} \min\{\sqrt{\lambda}, 1\}$. Thus (22) holds with $C_{\Omega,1} = \max\{\Omega(1), 2\sqrt{\Omega(1)}\}$.

To prove (23), we let $\lambda \in (0,\infty)$. Denote $\widetilde{\Omega}(\lambda)$ as c^*. We know from the definition of $\widetilde{\Omega}(\lambda)$ that $\frac{\Omega(c^*)}{(c^*)^2} \ge \lambda$.

When $c^* \le 1$, we use condition (8) and find $\Omega(c^*) \le C_\Omega^* (c^*)^q$. But $\Omega(c^*) \ge \lambda (c^*)^2$. So $C_\Omega^* (c^*)^q \ge \lambda (c^*)^2$ and $c^* \le (C_\Omega^*)^{1/(2-q)} \left(\frac{1}{\lambda}\right)^{1/(2-q)}$.

When $c^* > 1$, we apply (7) in Theorem 2 to c^* and obtain $\lambda \le \frac{\Omega(c^*)}{(c^*)^2} \le \frac{\Omega(1)c^*}{(c^*)^2} \le \frac{\Omega(1)}{c^*}$ and thereby $c^* \le \frac{\Omega(1)}{\lambda}$. Combing the above two cases, we know that (23) is valid with $C_{\Omega,2} = \max\{(C_\Omega^*)^{1/(2-q)}, \Omega(1)\}$. This proves the lemma. ■

Theorem 15 *Assume (9) with $r > 0$, and that Ω has a concave exponent $q \in [0,1]$ with (8) valid. If $0 < \delta \le 1$ and for some $1 \le p \le m$, the regularization parameter γ satisfies*

$$\gamma \ge \begin{cases} C_1 \left(\log \frac{4m}{\delta}\right)^{1+2r} \left(\max\left\{\frac{\lambda_p}{\lambda_1}, \frac{1}{\sqrt{m}}\right\}\right)^{r+1}, & \text{if } 0 < r \le \frac{1}{2}, \\ C_1 \left(\log \frac{4m}{\delta}\right)^{1+2r} \max\left\{\left(\frac{\lambda_p}{\lambda_1}\right)^{r+\frac{1}{2}}, \frac{1}{\sqrt{m}}\right\} \left(\max\left\{\frac{\lambda_p}{\lambda_1}, \frac{1}{\sqrt{m}}\right\}\right)^{\frac{1}{2}}, & \text{if } r > \frac{1}{2}, \end{cases} \tag{24}$$

then with confidence $1 - \delta$ we have

$$c_i^{\mathbf{z}} = 0, \quad \forall i = p+1, \ldots, m$$

and

$$\begin{aligned} \|f^{\mathbf{z}} - f_\rho\|_K \;\le\; & C_{\Omega,2}\sqrt{p}\left\{\left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}} + \frac{2\gamma}{\lambda_p}\right\} + \|g_\rho\|_K \lambda_p^r + C_3 \frac{\sqrt{p}\log\frac{4m}{\delta}}{\sqrt{m}} \lambda_p^{\min\{-1/2, r-1\}} \\ & + C_4 \lambda_p^{\min\{r-1,0\}} \left(\sum_{i=p+1}^{\infty} \lambda_i^{2\max\{r,1\}}\right)^{1/2}, \end{aligned} \tag{25}$$

where $C_1 \ge 1$, C_3 and C_4 are constants independent of γ, p, δ, or m.

The detailed proof of Theorem 15 will be given in Appendix A where the constants C_1, C_3 and C_4 will be specified explicitly. Here we outline the ideas of the proof by referring to three lemmas, Lemmas 18, 19 and 20 to be given in Appendix A, for estimating three quantities $|\lambda_i^{\mathbf{x}} - \lambda_i|$, $\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K|$ and $\sqrt{\lambda_i^{\mathbf{x}}}|\langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K|$.

Step 1. To achieve the desired sparsity, we apply (22) in Lemma 14 and know that for verifying condition (20) in Theorem 13, it is sufficient to show that for $i \ge p+1$,

$$|S_i^{\mathbf{z}}| < \frac{C_{\Omega,1}}{2} \frac{\min\{\sqrt{\lambda_i^{\mathbf{x}}/\gamma}, 1\}}{\lambda_i^{\mathbf{x}}/\gamma}$$

or equivalently,

$$\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}| < \frac{C_{\Omega,1}}{2} \min\left\{\sqrt{\gamma}, \gamma/\sqrt{\lambda_i^{\mathbf{x}}}\right\}. \tag{26}$$

Step 2. Our desired bound (26) is verified by estimating

$$\lambda_i^{\mathbf{x}} \le |\lambda_i^{\mathbf{x}} - \lambda_i| + \lambda_i$$

by Lemma 18 and the decay of $\{\lambda_i\}$, and estimating

$$\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}| \le \sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K| + \sqrt{\lambda_i^{\mathbf{x}}}|\langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K|$$

by Lemma 19 and Lemma 20.

Step 3. To prove the error bound (25), we expand the error function $f^{\mathbf{z}} - f_\rho$ with respect to the orthonormal basis $\{\phi_i^{\mathbf{x}}\}$ of $\mathcal{H}_K$ and express the norm as

$$\|f^{\mathbf{z}} - f_\rho\|_K^2 = \sum_{i\in\mathbb{N}} \left(\langle f^{\mathbf{z}} - f_\rho, \phi_i^{\mathbf{x}}\rangle_K\right)^2 = \sum_{i\in\mathbb{N}} \left(c_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K\right)^2.$$

Split

$$c_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K = \{c_i^{\mathbf{z}} - S_i^{\mathbf{z}}\} + \left\{S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K\right\}.$$

While the term $|c_i^{\mathbf{z}} - S_i^{\mathbf{z}}|$ can be bounded by $\widetilde{\Omega}\left(\frac{\lambda_i^{\mathbf{x}}}{\gamma}\right)$ according to Theorem 13, the other term will be expressed as

$$|S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K| = \frac{\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K|}{\sqrt{\lambda_i^{\mathbf{x}}}}.$$

Step 4. We can control the denominator of the above expression by introducing a set with large $\lambda_i^{\mathbf{x}}$ as $\mathcal{S} := \{i \in \{1,\dots,p\}, \lambda_i^{\mathbf{x}} > \lambda_p/2\}$, and then bound the expression by Lemma 19. This together with our previous estimate Guo and Zhou (2012) for the terms involving $i \in \mathbb{N}\setminus\mathcal{S}$ finally yields the desired error bound.

Let us demonstrate how to apply our general analysis in Theorem 15 by two special cases where the eigenvalues of the integral operator L_K decay polynomially and exponentially.

Corollary 16 *Assume (9) with $r > 0$, and that Ω has a concave exponent $q \in [0,1]$ with (8) valid. Suppose that for some positive constants D_1, D_2, $\alpha_1 \geq \alpha_2$, the eigenvalues $\{\lambda_i\}$ of L_K decay polynomially as*

$$D_1 i^{-\alpha_1} \leq \lambda_i \leq D_2 i^{-\alpha_2}, \qquad \forall i \in \mathbb{N} \tag{27}$$

with $2\alpha_2 \max\{r,1\} > 1$. Let $0 < \delta < 1$. If we choose

$$\gamma = C_1 (D_2/\lambda_1)^{r+1} \left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\min\{\frac{1+r}{2}, \frac{1+r}{1+2r}\}}, \tag{28}$$

then with confidence $1-\delta$ we have

$$c_i^{\mathbf{z}} = 0 \qquad \forall\ m^{\frac{1}{\alpha_2 \max\{2,1+2r\}}} + 1 \leq i \leq m \tag{29}$$

and

$$\|f^{\mathbf{z}} - f_\rho\|_K \leq C_2 \left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\theta_{rate}},$$

where $\theta_{rate} = \min\{\theta_1, \theta_2\}$ with

$$\begin{aligned}
\theta_1 &= \begin{cases} \frac{2(r+1)\alpha_2 - 2\alpha_1 - 2 + q}{4(2-q)\alpha_2}, & \text{if } 0 < r \leq 1/2, \\ \frac{2(2r+2)\alpha_2 - 4\alpha_1 - 2(2-q)}{4(2r+1)(2-q)\alpha_2}, & \text{if } r > 1/2, \end{cases} \\
\theta_2 &= \frac{2\alpha_2 r - 1 - 2(\alpha_1 - \alpha_2)\max\left\{1-r, \frac{1}{2}\right\}}{2\alpha_2 \max\{2, 1+2r\}},
\end{aligned}$$

and C_1 and C_2 are constants independent of m or δ (given explicitly in the proof).

Proof Denote $\mu = \max\{2, 1+2r\}$. Take $p = \lceil m^{\frac{1}{\alpha_2\mu}}\rceil$, the smallest integer greater than or equal to $m^{\frac{1}{\alpha_2\mu}}$. Then we have

$$m^{\frac{1}{\alpha_2\mu}} \leq p \leq 2m^{\frac{1}{\alpha_2\mu}}$$

and by (27),

$$\lambda_p \leq D_2 p^{-\alpha_2} \leq D_2 m^{-\frac{\alpha_2}{\alpha_2\mu}} = D_2 m^{-\frac{1}{\mu}}.$$

It follows that $\frac{\lambda_p}{\lambda_1} \leq \frac{D_2}{\lambda_1} m^{-\frac{1}{\mu}} \leq \frac{D_2}{\lambda_1}\frac{1}{\sqrt{m}}$ for $0 < r \leq \frac{1}{2}$ and for $r > \frac{1}{2}$, there holds $(\lambda_p/\lambda_1)^{r+\frac{1}{2}} \leq (D_2/\lambda_1)^{r+\frac{1}{2}}\frac{1}{\sqrt{m}}$. Note that (27) implies $\lambda_1 \leq D_2$. Then (24) is satisfied if we choose γ by (28). Hence the conclusion of Theorem 15 holds true. In particular, the statement (29) about the sparsity follows from the choice of p. What is left is to bound the right-hand side of (25) by estimating the four terms separately.

The first term of (25) can be estimated by bounding $\frac{2\gamma}{\lambda_p}$ from the choice of γ and the lower bound of λ_p as

$$\frac{2\gamma}{\lambda_p} \leq \frac{2C_1}{D_1}(D_2/\lambda_1)^{r+1}\left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\min\{\frac{1+r}{2}, \frac{1+r}{1+2r}\}} \left(2m^{\frac{1}{\alpha_2\mu}}\right)^{\alpha_1}.$$

Observe that

$$\min\left\{\frac{1+r}{2}, \frac{1+r}{1+2r}\right\} - \frac{\alpha_1}{\alpha_2\mu} = \begin{cases} \frac{1+r}{2} - \frac{\alpha_1}{2\alpha_2} = \frac{(r+1)\alpha_2 - \alpha_1}{2\alpha_2}, & \text{if } 0 < r \leq 1/2, \\ \frac{1+r}{1+2r} - \frac{\alpha_1}{(2r+1)\alpha_2} = \frac{2(2r+2)\alpha_2 - 4\alpha_1}{4(2r+1)\alpha_2}, & \text{if } r > 1/2. \end{cases}$$

Therefore,

$$C_{\Omega,2}\sqrt{p}\left\{\left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}} + \frac{2\gamma}{\lambda_p}\right\} \leq C_{\Omega,2}\sqrt{2}2^{\alpha_1+2}\frac{C_1}{D_1}(D_2/\lambda_1)^{r+1}\left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\theta_1},$$

where

$$\theta_1 = \begin{cases} \frac{(r+1)\alpha_2 - \alpha_1}{2(2-q)\alpha_2} - \frac{1}{4\alpha_2} = \frac{2(r+1)\alpha_2 - 2\alpha_1 - 2 + q}{4(2-q)\alpha_2}, & \text{if } 0 < r \leq 1/2, \\ \frac{2(2r+2)\alpha_2 - 4\alpha_1 - 2(2-q)}{4(2r+1)(2-q)\alpha_2}, & \text{if } r > 1/2. \end{cases}$$

The second term of (25) can be estimated by the choice of γ and the upper bound of λ_p as

$$\|g_\rho\|_K \lambda_p^r \leq \|g_\rho\|_K D_2^r p^{-\alpha_2 r} \leq \|g_\rho\|_K D_2^r m^{-\frac{2\alpha_2 r}{2\alpha_2\mu}} \leq \|g_\rho\|_K D_2^r m^{-\theta_2},$$

where θ_2 is the power index defined in the statement of the theorem.

The third term of (25) can be estimated by the choice of p and the lower bound of λ_p as

$$\begin{aligned}
&C_3\frac{\sqrt{p}\log\frac{4m}{\delta}}{\sqrt{m}}\lambda_p^{\min\{-1/2, r-1\}} \\
&\leq C_3 D_1^{\min\{-1/2, r-1\}} p^{\frac{1}{2} - \alpha_1\min\{-1/2, r-1\}}\left(\log\frac{4m}{\delta}\right) m^{-1/2} \\
&\leq C_3\sqrt{2}\left(2^{-\alpha_1}D_1\right)^{\min\{-1/2, r-1\}}\left(\log\frac{4m}{\delta}\right) m^{-\frac{1}{2} + \frac{1}{\alpha_2\mu}\left(\frac{1}{2} - \alpha_1\min\left\{-\frac{1}{2}, r-1\right\}\right)}.
\end{aligned}$$

From the identity $\mu - 2r = \max\{2, 2r+1\} - 2r = 2\max\{1-r, 1/2\}$, we see that the power index of m equals

$$\begin{aligned}
&-\frac{1}{2} + \frac{1}{\alpha_2 \mu}\left(\frac{1}{2} + \alpha_1 \max\left\{\frac{1}{2}, 1-r\right\}\right) \\
&= -\frac{1}{2\alpha_2\mu}\left(2\alpha_2 r - 1 + \alpha_2(\mu - 2r) - 2\alpha_1 \max\{1/2, 1-r\}\right) \\
&= -\frac{1}{2\alpha_2\mu}\left(2r\alpha_2 - 1 - 2(\alpha_1 - \alpha_2)\max\{1/2, 1-r\}\right)
\end{aligned}$$

which is exactly $-\theta_2$.

Turn to the last term of (25). By the restriction $2\alpha_2 \max\{r,1\} > 1$, we can bound the series as

$$\begin{aligned}
\sum_{i=p+1}^{\infty} \lambda_i^{2\max\{r,1\}} &\le \sum_{i=p+1}^{\infty} D_2^{2\max\{r,1\}} i^{-2\alpha_2 \max\{r,1\}} \\
&\le D_2^{2\max\{r,1\}} \int_p^{\infty} x^{-2\alpha_2\max\{r,1\}}\, \mathrm{d}x = \frac{D_2^{2\max\{r,1\}} p^{1-2\alpha_2\max\{r,1\}}}{2\alpha_2 \max\{r,1\} - 1}.
\end{aligned}$$

This combining with the choice of p and the lower bound of λ_p yields

$$\begin{aligned}
&C_4 \lambda_p^{\min\{r-1,0\}} \left(\sum_{i=p+1}^{\infty} \lambda_i^{2\max\{r,1\}}\right)^{1/2} \\
&\le \frac{C_4 D_1^{\min\{r-1,0\}} D_2^{\max\{r,1\}}}{\sqrt{2\alpha_2\max\{r,1\} - 1}} m^{\frac{1}{2\alpha_2\mu}(1 - 2\alpha_2\max\{r,1\} - 2\alpha_1 \min\{r-1,0\})}.
\end{aligned}$$

But

$$\begin{aligned}
&\frac{1}{2\alpha_2\mu}\left(1 - 2\alpha_2\max\{r,1\} - 2\alpha_1 \min\{r-1, 0\}\right) \\
&= -\frac{1}{2\alpha_2\mu}\left(2r\alpha_2 - 1 - 2(\alpha_1 - \alpha_2)\max\{0, 1-r\}\right) \le -\theta_2.
\end{aligned}$$

So we can combine this bound with the above estimates for the first three terms of (25) and verify the learning rate stated in the theorem by taking

$$\begin{aligned}
C_2 = \quad & C_{\Omega,2}\sqrt{2} 2^{\alpha_1+2} \frac{C_1}{D_1}(D_2/\lambda_1)^{r+1} + \|g_\rho\|_K D_2^r \\
& + C_3\sqrt{2}\left(2^{-\alpha_1} D_1\right)^{\min\{-1/2, r-1\}} + \frac{C_4 D_1^{\min\{r-1,0\}} D_2^{\max\{r,1\}}}{\sqrt{2\alpha_2\max\{r,1\} - 1}}.
\end{aligned}$$

The proof of Corollary 16 is complete. ∎

Corollary 17 *Assume (9) with $r > 0$, and that Ω has a concave exponent $q \in [0,1]$ with (8) valid. Suppose that for some positive constants D_1, D_2, $\beta_1 \ge \beta_2$, the eigenvalues $\{\lambda_i\}$ of L_K decay exponentially as*

$$D_1 \beta_1^{-i} \le \lambda_i \le D_2 \beta_2^{-i}, \qquad \forall i \in \mathbb{N}. \tag{30}$$

Let $0 < \delta < 1$. If we choose γ as (28), then with confidence $1-\delta$ we have

$$c_i^{\mathbf{z}} = 0, \qquad \forall\ \frac{\log(m+1)}{\max\{2, 1+2r\}\log\beta_2} + 1 \le i \le m \tag{31}$$

and

$$\|f^{\mathbf{z}} - f_\rho\|_K \le C_2 \left(\log\frac{4m}{\delta}\right)^{2r+1} m^{-\theta_{rate}},$$

where

$$\begin{aligned}
\theta_{rate} = \quad & \frac{1}{(2-q)\max\{2, 1+2r\}} \min\left\{1 + r - \frac{\log\beta_1}{\log\beta_2}, \right. \\
& \left. (2-q)r - \frac{(2-q)\log(\beta_1/\beta_2)}{\log\beta_2} \max\left\{1-r, \frac{1}{2}\right\}\right\}
\end{aligned}$$

and C_2 is a constant independent of m or δ (specified in the proof).

Proof Take $\mu = \max\{2, 1+2r\}$ and $p = \lceil \frac{\log(m+1)}{\mu\log\beta_2} \rceil$. Then

$$\frac{\log(m+1)}{\mu\log\beta_2} \le p < 1 + \frac{\log(m+1)}{\mu\log\beta_2},$$

which implies $m^{1/\mu} \le \beta_2^p \le \beta_1^p \le \beta_1 (2m)^{\frac{\log\beta_1}{\mu\log\beta_2}}$. Hence $\frac{\lambda_p}{\lambda_1} \le \frac{D_2}{\lambda_1}\beta_2^{-p} \le \frac{D_2}{\lambda_1}\frac{1}{\sqrt{m}}$ for $0 < r \le \frac{1}{2}$ and for $r > \frac{1}{2}$, there holds $(\lambda_p/\lambda_1)^{r+\frac{1}{2}} \le (D_2/\lambda_1)^{r+\frac{1}{2}} \frac{1}{\sqrt{m}}$. Then (24) is satisfied and the conclusion of Theorem 15 holds true. The statement (31) about the sparsity follows from the choice of p, and the next step is to estimate the four summing terms of the error bound (25).

For the first term, we notice

$$\frac{2\gamma}{\lambda_p} \le \frac{2C_1}{D_1}(D_2/\lambda_1)^{r+1} \left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\min\{\frac{1+r}{2}, \frac{1+r}{1+2r}\}} \beta_1 (2m)^{\frac{\log\beta_1}{\mu\log\beta_2}}.$$

So the first term can be bounded as

$$C_{\Omega,2}\sqrt{p}\left\{\left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}} + \frac{2\gamma}{\lambda_p}\right\} \le C_{\Omega,2} C_5 \sqrt{\log(2m)} \left(\log\frac{4m}{\delta}\right)^{1+2r} m^{-\frac{1}{2-q}\min\{\frac{1+r}{2}, \frac{1+r}{1+2r}\} + \frac{\log\beta_1}{(2-q)\mu\log\beta_2}},$$

where C_5 is the constant given by

$$C_5 = 2\sqrt{\frac{1}{\log 2} + \frac{1}{\mu\log\beta_2}} \max\left\{\frac{2C_1}{D_1}(D_2/\lambda_1)^{r+1}\beta_1 2^{\frac{\log\beta_1}{\mu\log\beta_2}}, 1\right\}.$$

The second term of (25) is easy to handle:

$$\|g_\rho\|_K \lambda_p^r \le \|g_\rho\|_K D_2^r \beta_2^{-pr} \le \|g_\rho\|_K D_2^r m^{-r/\mu}.$$

The third term of (25) can be estimated by the choice of p and the lower bound of λ_p as

$$C_3 \frac{\sqrt{p}\log\frac{4m}{\delta}}{\sqrt{m}} \lambda_p^{\min\{-1/2, r-1\}} \le C_6 \sqrt{\log(2m)} \left(\log\frac{4m}{\delta}\right) m^{-\frac{1}{2}+\frac{\log\beta_1}{\mu\log\beta_2}\max\{1/2,1-r\}},$$

where

$$C_6 = C_3 \sqrt{\frac{1}{\log 2} + \frac{1}{\mu\log\beta_2}} D_1^{\min\{-1/2,r-1\}} \left(\beta_1 2^{\frac{\log\beta_1}{\mu\log\beta_2}}\right)^{\max\{1/2,1-r\}}.$$

Observe that $\max\{1/2, 1-r\} + r = \mu/2$. The power index of m equals

$$\begin{aligned} &-\frac{1}{2} + \frac{\log\beta_1}{\mu\log\beta_2}\max\{1/2, 1-r\} = -\frac{r}{\mu} + \frac{r\log\beta_2 - \frac{\mu}{2}\log\beta_2 + \frac{\mu}{2}\log\beta_1 - r\log\beta_1}{\mu\log\beta_2} \\ &= -\frac{r}{\mu} + \frac{\log\frac{\beta_1}{\beta_2}}{\mu\log\beta_2}\max\{1/2, 1-r\}. \end{aligned}$$

Finally, we bound the series in the last term of (25) and find

$$\begin{aligned} &C_4 \lambda_p^{\min\{r-1,0\}} \left(\sum_{i=p+1}^\infty \lambda_i^{2\max\{r,1\}}\right)^{1/2} \\ &\le C_4 (\beta_1/D_1)^{\max\{1-r,0\}} (2m)^{\frac{\log\beta_1}{\mu\log\beta_2}\max\{1-r,0\}} D_2^{\max\{r,1\}} \left(\sum_{i=p+1}^\infty \beta_2^{-2i\max\{r,1\}}\right)^{1/2} \\ &\le C_4 \left(\frac{\beta_1}{D_1}\right)^{\max\{1-r,0\}} (2m)^{\frac{\log\beta_1}{\mu\log\beta_2}\max\{1-r,0\}} D_2^{\max\{r,1\}} \frac{\beta_2^{-p\max\{r,1\}}}{\sqrt{\beta_2^{2\max\{r,1\}} - 1}} \\ &\le C_4 \left(\frac{\beta_1}{D_1}\right)^{\max\{1-r,0\}} \frac{D_2^{\max\{r,1\}}}{\sqrt{\beta_2^{2\max\{r,1\}} - 1}} (2m)^{\frac{\max\{1-r,0\}(\log\beta_1)}{\mu\log\beta_2}} m^{-\frac{\max\{r,1\}}{\mu}}. \end{aligned}$$

Note that the power index of m is

$$\frac{\max\{1-r,0\}\log\beta_1}{\mu\log\beta_2} - \frac{\max\{r,1\}}{\mu} = \frac{\max\{1-r,0\}\log(\beta_1/\beta_2) - r\log\beta_2}{\mu\log\beta_2}.$$

Then the desired learning rate is verified by observing $\min\{\frac{1+r}{2}, \frac{1+r}{1+2r}\}\mu = 1 + r$ and taking

$$C_2 = C_{\Omega,2}C_5 + \|g_\rho\|_K D_2^r + C_6 + C_4 \left(\frac{\beta_1}{D_1} 2^{\frac{\log\beta_1}{\mu\log\beta_2}}\right)^{\max\{1-r,0\}} \frac{D_2^{\max\{r,1\}}}{\sqrt{\beta_2^{2\max\{r,1\}} - 1}}.$$

The proof of Corollary 17 is complete. ∎

5. Simulations

In this section we give some simulations for both artificial and real data. We demonstrate that with either the ℓ^q-regularizer or the SCAD penalty, RKPCA is comparable with the regularized least squares in learning error, and achieves satisfactory sparsity.

5.1 Simulation on artificial data

We start with a simulation on artificial data. For simplicity we take $X = [0, 1]$. Let ρ be a Borel probability measure on $X \times Y$ to be specified later and $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^m$ be a sample of size m divisible by 5. We divide $\mathbf{z}$ evenly into five disjoint subsets $\mathbf{z} = \cup_{j=1}^5 \mathbf{z}_j$, and do 5-fold cross-validation to select the parameter γ^* from a geometric sequence $\{10^{-10}, \cdots, 10^{-2}\}$ of length 60, to minimize the root-mean-square error (RMSE). Here, with a fixed γ the RMSE score is defined by

$$\mathcal{E}_{\mathsf{RMSE},\mathbf{z}}(\gamma) = \left(\sum_{j=1}^5 \sum_{(x,y)\in\mathbf{z}\setminus\mathbf{z}_j} \left(f_\gamma^{\mathbf{z}_j}(x) - y\right)^2\right)^{1/2}. \tag{32}$$

Then RKPCA is trained with γ^* on $\mathbf{z}$ and outputs $f^{\mathbf{z}}$. Sparsity is evaluated by the percentage of the non-zero coefficients in (2). The prediction performance is evaluated with the oracle RMSE defined by

$$\mathcal{E}_{\mathsf{RMSE},f_\rho}(f^{\mathbf{z}}) = \left(\int_0^1 (f^{\mathbf{z}}(x) - f_\rho(x))^2 \,\mathrm{d}x\right)^{1/2}, \tag{33}$$

where the integral is computed with 1000 equispaced points.

First, we simulate with the Gaussian kernel

$$K_{\mathsf{G}} = \exp\left(-\frac{(x-y)^2}{0.6^2}\right).$$

We use the regression model $f_\rho(x) = \mathrm{e}^{-(x-1/3)^2/0.7^2}$. Let ρ_X be the uniform distribution on $[0, 1]$, and $\rho(\cdot|x)$ be the uniform distribution on $[f_\rho(x) - 0.1, f_\rho(x) + 0.1]$. The simulation is summarized in Table 1. We find that the behavior of the SCAD penalty is comparable on this data set with the penalty $\Omega(|c|) = |c|$, and despite of very strong sparsity, RKPCA achieves comparable precision with that of RLS.

Next, we simulate with the Sobolev kernel

$$K_{\mathsf{S}}(x, y) = \mathrm{e}^{-|x-y|}.$$

The regression function is set as $f_\rho(x) = |2x - 1|^\tau$. The marginal and conditional probabilities ρ_X and $\rho(\cdot|x)$ are defined as above. We use $\tau = 1$, 1.5, 2.5, and 4.5. Note that in addition to $\mathcal{E}_{\mathsf{RMSE},f_\rho}$, the RKHS norm is now also easy to compute as another measurement. In fact, one has for $f, g \in \mathcal{H}_{K_{\mathsf{S}}}$,

$$2\langle f, g\rangle_{K_{\mathsf{S}}} = f(0)g(0) + f(1)g(1) + \int_0^1 f(t)g(t)\,\mathrm{d}t + \int_0^1 f'(t)g'(t)\,\mathrm{d}t.$$

The simulation is summarized in Table 2, from which we have the following observations:

sample size	RKPCA				RLS
	$q=1$	$q=2/3$	$q=1/3$	SCAD	
100	3.5(2.0)%	3.2(1.7)%	3.2(1.9)%	3.3(1.5)%	100(0)%
	0.013(0.006)	0.012(0.005)	0.012(0.007)	0.011(0.005)	0.012(0.005)
300	1.4(1.2)%	1.2(1.4)%	1.1(0.7)%	1.1(0.5)%	100(0)%
	0.007(0.004)	0.007(0.006)	0.007(0.003)	0.006(0.002)	0.007(0.003)
1000	0.4(0.4)%	0.4(0.3)%	0.3(0.2)%	0.4(0.2)%	100(0)%
	0.004(0.002)	0.004(0.002)	0.004(0.002)	0.004(0.001)	0.004(0.002)

Table 1: A simulation with Gaussian kernel. Here SCAD and $q = 1$, $2/3$, and $1/3$ stand for RKPCA with penalty SCAD (as defined in Example 2, and we set $b = 2.5$) and $\Omega(|c|) = |c|^q$ respectively. The scores of RLS are also listed for comparison. In each cell, the top percentage gives the proportion of the non-zero coefficients, and the bottom score is $\mathcal{E}_{\mathsf{RMSE},f_\rho}$ as defined in (33). Each simulation is repeated 100 times. We present the mean scores in the table, and give the sample standard deviation in parentheses.

(a) The sparsity and learning error of RKPCA with the SCAD penalty is again comparable on this data set to that with the penalty $\Omega(|c|) = |c|$. This shows that the expression of the SCAD penalty near the origin (the same as that for $\Omega(|c|) = |c|$) and the concave exponent q play a crucial role in its performance.

(b) Compared with RLS, RKPCA achieves very strong sparsity while its approximation ability with $\Omega(|c|) = |c|$ in terms of the RKHS metric is consistently better. This might be caused by the orthogonality of the empirical features in the RKHS. The learning ability in terms of the root-mean-square error defined by (33) is comparable.

5.2 Simulation on MHC-peptide binding data

We apply RKPCA to the quantitative Immune Epitope Database (IEDB) benchmark data of human leukocyte antigen (HLA)–peptide binding affinities, introduced in (Nielsen et al., 2008). Nielsen and Lund (2009) developed an artificial neural network-based algorithm called NN-align, which gave on this data set the state-of-the-art prediction in 2009. Later, Shen et al. (2012) designed a string kernel denoted in their paper by $\hat{K}^3$, and applied it with the regularized least squares (RLS), which produced better prediction than NN-align on the same data set. We use this $\hat{K}^3$ in RKPCA, and show that RKPCA achieves some sparsity in addition to the precision comparable with that in (Shen et al., 2012).

Here are more details of our simulation. The quantitative IEDB benchmark data set in (Nielsen et al., 2008) as mentioned above, consists of 14 groups, each containing the affinities of a set of peptides to a specific HLA allele. We use the 14 groups separately. Now fix an allele a and denote $X = \mathscr{P}_a$ the set of peptides given in the data set. For $p \in \mathscr{P}_a$, the affinity $y_p \in [0,1] \subset Y = \mathbb{R}$ is a real number (see Nielsen and Lund (2009); Shen et al. (2012)). We divide $\mathscr{P}_a$ into 5 disjoint subsets $\mathscr{P}_a = \cup_{j=1}^5 \mathscr{P}_a^j$, following exactly the division in (Nielsen and Lund, 2009) and (Shen et al., 2012), for a 5-fold cross-validation. In the jth cross-validation round ($j = 1, \cdots, 5$), we take $\mathscr{P}_a^j$ as testing data and $\mathscr{P}_a \backslash \mathscr{P}_a^j$ as training data. Within the training data, another 5-fold cross-validation is employed to select the

parameter γ_j^* in (3), from a geometric sequence $\{10^{-8}, \cdots, 10^{-2}\}$ of length 60 to minimize the RMSE score defined in (32). Then RKPCA is trained on $\mathscr{P}_a \backslash \mathscr{P}_a^j$ with γ_j^* to predict the affinities on $\mathscr{P}_a^j$. After all the five rounds, each peptide $p \in \mathscr{P}_a$ has a predicted affinity $\tilde{y}_p$ obtained during the jth round where $\mathscr{P}_a^j \ni p$. Note that $\tilde{y}_p$ may not always fall in $[0,1]$, and might be projected back onto $[0,1]$ to increase precision. However we do not adopt the projection, for being consistent and comparable with (Shen et al., 2012) where they did not either. Since there is no oracle information, we use

$$\mathcal{E}_{\mathsf{RMSE},a} = \left(\frac{1}{\#\mathscr{P}_a} \sum_{p \in \mathscr{P}_a} (\tilde{y}_p - y_p)^2 \right)^{1/2} \tag{34}$$

as the RMSE score. A lower RMSE score indicates a better performance.

The area under the receiver operating characteristic (ROC) curve (AUC), defined as

$$\mathcal{E}_{\mathsf{AUC},a} = \frac{\#\left\{(p,p') : p \in \mathscr{P}_{a,B}, p' \in \mathscr{P}_{a,N}, \tilde{y}_p > \tilde{y}_{p'}\right\}}{(\#\mathscr{P}_{a,B})\,(\#\mathscr{P}_{a,N})} \in [0,1], \tag{35}$$

is another performance index. Here $\mathscr{P}_{a,B} = \{p \in \mathscr{P}_a : y_p > 0.426\}$ and $\mathscr{P}_{a,N} = \mathscr{P}_a \backslash \mathscr{P}_{a,B}$ are the sets of binding peptides and non-binding ones respectively, with the threshold 0.426 used in (Nielsen and Lund, 2009). A higher AUC score indicates a better performance. The above scores (34) and (35) are used in (Shen et al., 2012). See also (Nielsen and Lund, 2009) for details.

We test the RKPCA with $\Omega(c) = |c|^q$, where q is set to be 1, 2/3, and 1/3 in three separated tests, and with the SCAD penalty. For defining $\hat{K}^3$, the Hadamard power index is fixed to be 0.11387 for simplicity, as suggested in (Shen et al., 2012).

The simulation is summarized in Table 3, from which we have the following observations:

(a) In terms of AUC on this real data set, RLS (Shen et al., 2012) has better performance than NN-align (Nielsen and Lund, 2009). The improvement is 0.55% on average, with better AUC scores for 9 out of 14 test groups while the score difference is always at the second significant figure. RKPCA with $\Omega(c) = |c|$ has even slightly better performance, giving an improvement of 0.11% on average, and better AUC scores for 8 out of 14 test groups with the score difference always at the third significant figure only. Improvements in (Shen et al., 2012) and in our simulation seem to be small, but we regard the results to be valuable because this data set has been well investigated in the immunological literature and any improvement is difficult. In particular, the dissimilarity metric BLOSUM62-2 among the 20 basic amino-acids, based on which the string kernel $\hat{K}^3$ is constructed in (Shen et al., 2012), was obtained in a very tight form after long-term effort and a vast biological literature (see, e.g., Henikoff and Henikoff (1992)).

(b) Sparsity and error bounds in terms of both AUC and root-mean-square error for the simulation with the SCAD penalty is almost the same on this real data set as that with $\Omega(|c|) = |c|$, verifying again the role of the concave exponent $q = 1$.

sample size	τ	RKPCA				RLS
		$q=1$	$q=2/3$	$q=1/3$	SCAD	
100	1.0	14.2(7.5)%	9.7(4.4)%	8.1(3.7)%	16.0(8.0)%	100(0)%
		0.026(0.007)	0.026(0.006)	0.029(0.006)	0.025(0.005)	0.025(0.005)
		0.685(0.318)	0.761(0.401)	1.033(0.772)	0.738(0.378)	0.801(0.185)
	1.5	16.9(8.7)%	11.2(6.9)%	9.2(6.4)%	17.7(8.4)%	100(0)%
		0.026(0.007)	0.027(0.005)	0.030(0.006)	0.026(0.006)	0.027(0.008)
		0.780(0.384)	0.885(0.653)	1.006(0.966)	0.805(0.378)	0.908(0.228)
	2.5	22.5(10.3)%	14.2(6.7)%	13.1(11.7)%	20.4(8.8)%	100(0)%
		0.028(0.007)	0.031(0.008)	0.033(0.007)	0.029(0.007)	0.029(0.006)
		1.086(0.545)	1.217(0.678)	1.601(1.653)	1.124(1.235)	1.195(0.380)
	4.5	26.6(10.6)%	17.8(8.1)%	17.8(11.6)%	26.2(9.8)%	100(0)%
		0.033(0.007)	0.036(0.010)	0.039(0.010)	0.036(0.011)	0.035(0.010)
		1.483(0.515)	1.758(0.814)	2.488(1.979)	1.623(0.882)	1.685(0.385)
300	1.0	5.4(1.6)%	3.8(1.3)%	3.0(1.6)%	6.1(2.8)%	100(0)%
		0.015(0.003)	0.016(0.003)	0.018(0.003)	0.016(0.002)	0.016(0.002)
		0.503(0.073)	0.604(0.264)	0.865(1.084)	0.568(0.207)	0.652(0.104)
	1.5	6.4(2.3)%	4.1(1.4)%	3.2(1.2)%	6.0(2.1)%	100(0)%
		0.016(0.003)	0.017(0.003)	0.019(0.004)	0.016(0.003)	0.016(0.003)
		0.589(0.164)	0.666(0.242)	0.824(0.687)	0.578(0.148)	0.708(0.117)
	2.5	7.7(2.5)%	5.2(1.9)%	4.0(1.2)%	7.3(2.2)%	100(0)%
		0.018(0.004)	0.019(0.003)	0.021(0.003)	0.018(0.003)	0.018(0.002)
		0.802(0.166)	0.946(0.463)	1.044(0.683)	0.759(0.139)	0.966(0.152)
	4.5	10.2(2.7)%	6.9(2.0)%	5.2(1.2)%	9.8(3.1)%	100(0)%
		0.020(0.004)	0.022(0.003)	0.024(0.003)	0.020(0.004)	0.021(0.003)
		1.142(0.218)	1.372(0.475)	1.495(0.768)	1.164(0.537)	1.382(0.223)
1000	1.0	2.0(0.6)%	1.4(0.5)%	1.0(0.3)%	2.0(0.5)%	100(0)%
		0.009(0.001)	0.010(0.001)	0.011(0.002)	0.009(0.001)	0.010(0.001)
		0.434(0.085)	0.484(0.180)	0.533(0.368)	0.421(0.039)	0.570(0.114)
	1.5	2.3(0.7)%	1.5(0.4)%	1.2(0.3)%	2.4(0.6)%	100(0)%
		0.010(0.001)	0.010(0.002)	0.011(0.002)	0.010(0.001)	0.010(0.001)
		0.467(0.068)	0.516(0.122)	0.583(0.325)	0.477(0.066)	0.612(0.103)
	2.5	2.8(0.6)%	1.8(0.4)%	1.4(0.3)%	3.0(0.7)%	100(0)%
		0.011(0.001)	0.012(0.001)	0.013(0.002)	0.011(0.001)	0.011(0.001)
		0.642(0.090)	0.711(0.207)	0.846(0.533)	0.647(0.085)	0.781(0.085)
	4.5	3.8(0.9)%	2.4(0.4)%	1.9(0.4)%	3.7(0.8)%	100(0)%
		0.012(0.002)	0.013(0.001)	0.015(0.002)	0.012(0.002)	0.013(0.001)
		0.950(0.155)	0.998(0.184)	1.254(0.912)	0.931(0.108)	1.163(0.119)

Table 2: A simulation with Sobolev kernel. Here SCAD and $q = 1$, $2/3$, and $1/3$ stand for RKPCA with penalty SCAD (as defined in Example 2, and we set $b = 2.5$) and $\Omega(|c|) = |c|^q$ respectively. The scores of RLS are also listed for comparison. In each cell, the top percentage gives the proportion of the non-zero coefficients, the middle score is $\mathcal{E}_{\mathsf{RMSE},f_\rho}$ as defined in (33), and the bottom score gives the RKHS distance of $f^{\mathbf{z}}$ to f_ρ. Each simulation is repeated 100 times. We present the mean scores in the table, and give the sample standard deviation in parentheses.

Allele a	$\#\mathscr{P}_a$	NN-align	RLS	RKPCA			
				$q=1$	$q=2/3$	$q=1/3$	SCAD
DRB1*0101	5166	–	–	74.65%	59.30%	60.81%	74.66%
		–	0.18660	0.18690	0.18746	0.18830	0.18694
		0.836	0.85707	0.85651	0.85512	0.85306	0.85637
DRB1*0301	1020	–	–	88.04%	71.84%	56.47%	86.00%
		–	0.18497	0.18476	0.18495	0.18551	0.18483
		0.816	0.82813	0.82995	0.82950	0.82714	0.83008
DRB1*0401	1024	–	–	72.39%	60.16%	61.40%	73.36%
		–	0.24055	0.24089	0.24202	0.24277	0.24152
		0.771	0.78431	0.78023	0.77697	0.77505	0.77839
DRB1*0404	663	–	–	70.55%	57.84%	57.88%	71.12%
		–	0.20702	0.20797	0.20918	0.20878	0.20796
		0.818	0.81425	0.81695	0.81134	0.80801	0.81701
DRB1*0405	630	–	–	81.47%	69.56%	63.06%	78.85%
		–	0.20069	0.20037	0.20017	0.20076	0.20048
		0.781	0.79296	0.79837	0.79929	0.79791	0.79799
DRB1*0701	853	–	–	98.65%	91.76%	86.96%	98.65%
		–	0.21944	0.21826	0.21840	0.21849	0.21826
		0.841	0.83440	0.83883	0.83918	0.83916	0.83883
DRB1*0802	420	–	–	96.85%	93.75%	87.98%	96.90%
		–	0.19666	0.19555	0.19557	0.19572	0.19557
		0.832	0.83538	0.83968	0.83938	0.83749	0.83968
DRB1*0901	530	–	–	73.11%	53.35%	50.94%	74.15%
		–	0.25398	0.25563	0.25653	0.25784	0.25593
		0.616	0.66591	0.66293	0.66273	0.66163	0.66177
DRB1*1101	950	–	–	94.61%	83.82%	80.21%	94.61%
		–	0.20776	0.20799	0.20802	0.20780	0.20799
		0.823	0.83703	0.83679	0.83680	0.83706	0.83678
DRB1*1302	498	–	–	84.99%	72.64%	62.25%	81.28%
		–	0.22569	0.22518	0.22540	0.22578	0.22496
		0.831	0.80410	0.80479	0.80439	0.80303	0.80533
DRB1*1501	934	–	–	75.80%	64.94%	74.79%	77.89%
		–	0.23268	0.23318	0.23401	0.23419	0.23313
		0.758	0.76436	0.76258	0.76086	0.76058	0.76219
DRB3*0101	549	–	–	92.94%	89.57%	87.52%	92.49%
		–	0.15945	0.15932	0.15916	0.15911	0.15934
		0.844	0.80228	0.80504	0.80546	0.80622	0.80509
DRB4*0101	446	–	–	96.75%	81.28%	76.18%	96.75%
		–	0.20809	0.20765	0.20838	0.20834	0.20765
		0.811	0.81057	0.81096	0.80791	0.80713	0.81098
DRB5*0101	924	–	–	100.00%	99.95%	98.76%	100.00%
		–	0.23038	0.23045	0.23045	0.23046	0.23045
		0.797	0.80568	0.80549	0.80550	0.80557	0.80549
Average		–	–	85.77%	74.98%	71.80%	85.48%
		–	0.21100	0.21101	0.21141	0.21170	0.21107
		0.7982	0.80260	0.80351	0.80246	0.80136	0.80328

Table 3: Comparison of sparsity and error. Each cell consists of the average of proportions of the non-zero coefficients in the five rounds of test (the top percentage), RMSE defined by (34) (the middle number), and AUC defined by (35) (the bottom number). We cite the scores of NN-align from (Nielsen and Lund, 2009) and that of RLS from (Shen et al., 2012).

Acknowledgments

We would like to sincerely thank the referees for their constructive suggestions and comments. The work described in this paper was supported by the Research Grants Council of Hong Kong [Project No. CityU 105011] and by the National Natural Science Foundation of China under Grants 11461161006 and 11471292. The corresponding author is Ding-Xuan Zhou.

Appendix A. Proof of Theorem 15

In this appendix, we prove our general result on sparsity and error bounds stated in Theorem 15.

The following three lemmas are needed for proving Theorem 15. The first one is cited from (Zwald and Blanchard, 2006). See also (Koltchinskii and Giné, 2000; Guo and Zhou, 2012).

Lemma 18 *(a) We have*

$$\sum_{i=1}^{\infty}(\lambda_i-\lambda_i^{\mathbf{x}})^2 \le \|L_K - L_K^{\mathbf{x}}\|_{HS}^2. \tag{36}$$

(b) For any $0<\delta<1$, with confidence $1-\delta$ we have

$$\|L_K - L_K^{\mathbf{x}}\|_{HS} \le \frac{4\kappa^2\log\frac{2}{\delta}}{\sqrt{m}}. \tag{37}$$

The second lemma needed for proving Theorem 15 improves our previous estimate $\|\{\lambda_i^{\mathbf{x}}|S_i^{\mathbf{z}}-\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K|\}\|_{\ell^2} \le \frac{8M\kappa\log\frac{2}{\delta}}{\sqrt{m}}$ given in (Guo and Zhou, 2012) for the case of ℓ^1-penalty. The significant improvement we make here is to reduce the power of $\lambda_i^{\mathbf{x}}$ from 1 to $\frac{1}{2}$. Hence a different method for the proof is needed.

Lemma 19 *Let $f_\rho\in\mathcal{H}_K$. For $0<\delta<1$, with confident $1-\delta$ we have*

$$\sqrt{\lambda_i^{\mathbf{x}}}\left|S_i^{\mathbf{z}}-\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K\right| \le \frac{2\sqrt{2}M}{\sqrt{m}}\sqrt{\log\frac{2m}{\delta}}, \qquad \forall\, i\in\mathbb{N}. \tag{38}$$

Proof When $\lambda_i^{\mathbf{x}}=0$, (38) is obvious. When $i\ge m+1$, $\lambda_i^{\mathbf{x}}=0$ since the rank of $L_K^{\mathbf{x}}$ is not greater than m. For any fixed $\lambda_i^{\mathbf{x}}>0$, denote

$$\eta_j=\frac{y_j-f_\rho(x_j)}{\sqrt{m}}, \qquad a_j=\frac{\phi_i^{\mathbf{x}}(x_j)}{\sqrt{m\lambda_i^{\mathbf{x}}}}.$$

Then by the definition of $S_i^{\mathbf{z}}$, $\sqrt{\lambda_i^{\mathbf{x}}}\left(S_i^{\mathbf{z}}-\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K\right)=\sum_{j=1}^m\eta_j a_j$. Also, $\sum_{j=1}^m a_j^2=1$. Since $|y|\le M$ almost surely, we have $|a_j\eta_j|\le 2M|a_j|/\sqrt{m}$ almost surely. By Hoeffding's inequality, we have for any $\varepsilon>0$,

$$\mathbb{P}\left\{\left|\sum_{j=1}^m a_j\eta_j\right|\ge\varepsilon\right\}\le 2\exp\left(-\frac{m\varepsilon^2}{8M^2}\right).$$

Taking the union of the above at most m events, we know that

$$\mathbb{P}\left\{\max_{i=1,\dots,m}\left|\sqrt{\lambda_i^{\mathbf{x}}}\left(S_i^{\mathbf{z}}-\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K\right)\right|\ge\varepsilon\right\}\le 2m\exp\left(-\frac{m\varepsilon^2}{8M^2}\right).$$

One completes the proof by taking $\varepsilon>0$ to be the positive solution to the equation $2m\exp(-\frac{m\varepsilon^2}{8M^2})=\delta$. ■

Lemma 20 *Let $I\subset\mathbb{N}$ be a finite index set. If $f_\rho=L_K^r g_\rho$ for some $g_\rho\in\mathcal{H}_K$, then when $0<r<1/2$,*

$$\left(\sum_{i\in I}(\sqrt{\lambda_i^{\mathbf{x}}}\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K)^2\right)^{1/2} \le 2^r\|g_\rho\|_K(\#I)^{(1-2r)/4}\|L_K-L_K^{\mathbf{x}}\|_{\mathrm{HS}}^{(1+2r)/2} + 2^r\|g_\rho\|_K\left(\sum_{i\in I}(\lambda_i^{\mathbf{x}})^{1+2r}\right)^{1/2}, \tag{39}$$

and when $r\ge 1/2$,

$$\left(\sum_{i\in I}(\sqrt{\lambda_i^{\mathbf{x}}}\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K)^2\right)^{1/2} \le \sqrt{2}\lambda_1^{r-\frac{1}{2}}\|g_\rho\|_K\|L_K-L_K^{\mathbf{x}}\|_{\mathrm{HS}} + 2^r\|g_\rho\|_K\left(\sum_{i\in I}(\lambda_i^{\mathbf{x}})^{1+2r}\right)^{1/2}. \tag{40}$$

Proof Let $g_\rho=\sum_{j=1}^\infty d_j\phi_j$ with $\{d_j\}\in\ell^2$. Then $\|\{d_j\}\|_{\ell^2}=\|g_\rho\|_K$ and $f_\rho=\sum_{j=1}^\infty\lambda_j^r d_j\phi_j$. For $i\in I$, since whenever $\lambda_i^{\mathbf{x}}=0$, $\sqrt{\lambda_i^{\mathbf{x}}}\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K=0$, without loss of generality we assume $\lambda_i^{\mathbf{x}}>0$. Then we expand $\sqrt{\lambda_i^{\mathbf{x}}}\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K$ as

$$\sqrt{\lambda_i^{\mathbf{x}}}\langle f_\rho,\phi_i^{\mathbf{x}}\rangle_K=\left(\sum_{j:\lambda_j\ge 2\lambda_i^{\mathbf{x}}}+\sum_{j:\lambda_j<2\lambda_i^{\mathbf{x}}}\right)\sqrt{\lambda_i^{\mathbf{x}}}\lambda_j^r d_j\langle\phi_j,\phi_i^{\mathbf{x}}\rangle_K. \tag{41}$$

The second sum in (41) is easy to handle:

$$\left|\sum_{j:\lambda_j<2\lambda_i^{\mathbf{x}}}\sqrt{\lambda_i^{\mathbf{x}}}\lambda_j^r d_j\langle\phi_j,\phi_i^{\mathbf{x}}\rangle_K\right| \le 2^r(\lambda_i^{\mathbf{x}})^{(1+2r)/2}\|\{d_j\}\|_{\ell^2}\left(\sum_{j=1}^\infty\langle\phi_j,\phi_i^{\mathbf{x}}\rangle_K^2\right)^{1/2} = 2^r\|g_\rho\|_K(\lambda_i^{\mathbf{x}})^{(1+2r)/2}. \tag{42}$$

When $r\ge 1/2$ and $\lambda_j\ge 2\lambda_i^{\mathbf{x}}$, since $\sqrt{\lambda_i^{\mathbf{x}}\lambda_j}\le\frac{\lambda_j}{\sqrt{2}}\le\sqrt{2}(\lambda_j-\lambda_i^{\mathbf{x}})$, the first sum in (41) can be bounded as

$$\left|\sum_{j:\lambda_j\ge 2\lambda_i^{\mathbf{x}}}\sqrt{\lambda_i^{\mathbf{x}}}\lambda_j^r d_j\langle\phi_j,\phi_i^{\mathbf{x}}\rangle_K\right| \le \sum_{j:\lambda_j\ge 2\lambda_i^{\mathbf{x}}}\lambda_j^{r-\frac{1}{2}}\sqrt{2}\left|(\lambda_j-\lambda_i^{\mathbf{x}})\langle\phi_j,\phi_i^{\mathbf{x}}\rangle_K d_j\right| \le \sqrt{2}\lambda_1^{r-\frac{1}{2}}\|g_\rho\|_K\left(\sum_{j=1}^\infty(\lambda_j-\lambda_i^{\mathbf{x}})^2\langle\phi_j,\phi_i^{\mathbf{x}}\rangle_K^2\right)^{1/2}. \tag{43}$$

When $0 < r < 1/2$ and $\lambda_j \geq 2\lambda_i^{\mathbf{x}}$, we observe that

$$\frac{\sqrt{\lambda_i^{\mathbf{x}}}\lambda_j^r}{|\lambda_j - \lambda_i^{\mathbf{x}}|^{r+\frac{1}{2}}} \leq \frac{\lambda_j^{r+\frac{1}{2}}/\sqrt{2}}{(\lambda_j/2)^{r+\frac{1}{2}}} = 2^r.$$

So in this case the first sum in (41) can also be bounded as

$$\begin{aligned}
&\left| \sum_{j:\lambda_j \geq 2\lambda_i^{\mathbf{x}}} \sqrt{\lambda_i^{\mathbf{x}}}\lambda_j^r d_j \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K \right| \leq \sum_{j:\lambda_j \geq 2\lambda_i^{\mathbf{x}}} 2^r |\lambda_j - \lambda_i^{\mathbf{x}}|^{r+\frac{1}{2}} \left| d_j \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K \right| \\
&\leq 2^r \left(\sum_j d_j^2 \right)^{1/2} \left(\sum_j \left| (\lambda_j - \lambda_i^{\mathbf{x}}) \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K \right|^{(r+\frac{1}{2})\frac{2}{r+\frac{1}{2}}} \right)^{\frac{r+\frac{1}{2}}{2}} \left(\sum_j \left| \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K \right|^{(\frac{1}{2}-r)\frac{2}{\frac{1}{2}-r}} \right)^{\frac{\frac{1}{2}-r}{2}} \\
&= 2^r \|g_\rho\|_K \left(\sum_{j=1}^\infty (\lambda_j - \lambda_i^{\mathbf{x}})^2 \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K^2 \right)^{(1+2r)/4}. \qquad (44)
\end{aligned}$$

By (41), (42), and the triangle inequality,

$$\begin{aligned}
\left(\sum_{i\in I} |\sqrt{\lambda_i^{\mathbf{x}}} \langle f_\rho, \phi_i^{\mathbf{x}} \rangle_K|^2 \right)^{1/2} &\leq \left(\sum_{i\in I} \left(\sum_{j:\lambda_j \geq 2\lambda_i^{\mathbf{x}}} \sqrt{\lambda_i^{\mathbf{x}}}\lambda_j^r d_j \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K \right)^2 \right)^{1/2} \\
&+ 2^r \|g_\rho\|_K \left(\sum_{i\in I} (\lambda_i^{\mathbf{x}})^{1+2r} \right)^{1/2}. \qquad (45)
\end{aligned}$$

We denote the first term of the right-hand side of (45) as Υ. The definition of the Hilbert-Schmidt norm tells us that

$$\|L_K - L_K^{\mathbf{x}}\|_{HS}^2 = \sum \|(L_K - L_K^{\mathbf{x}})\phi_i^{\mathbf{x}}\|_K^2 = \sum_{i,j=1}^\infty (\lambda_j - \lambda_i^{\mathbf{x}})^2 \langle \phi_i^{\mathbf{x}}, \phi_j \rangle_K^2. \qquad (46)$$

So when $r \geq 1/2$, (43) and (46) give

$$\begin{aligned}
\Upsilon &\leq \sqrt{2}\lambda_1^{r-\frac{1}{2}} \|g_\rho\|_K \left(\sum_{i\in I} \sum_{j=1}^\infty (\lambda_j - \lambda_i^{\mathbf{x}})^2 \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K^2 \right)^{1/2} \\
&\leq \sqrt{2}\lambda_1^{r-\frac{1}{2}} \|g_\rho\|_K \|L_K - L_K^{\mathbf{x}}\|_{\mathrm{HS}},
\end{aligned}$$

which proves (40). When $0 < r < 1/2$, by (44), (46) and Hölder's inequality, we have

$$\begin{aligned}
\Upsilon &\leq 2^r \|g_\rho\|_K \left(\sum_{i\in I} \left(\sum_{j=1}^\infty (\lambda_j - \lambda_i^{\mathbf{x}})^2 \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K^2 \right)^{(1+2r)/2} \right)^{1/2} \\
&\leq 2^r \|g_\rho\|_K \left(\sum_{i\in I} \sum_{j=1}^\infty (\lambda_j - \lambda_i^{\mathbf{x}})^2 \langle \phi_j, \phi_i^{\mathbf{x}} \rangle_K^2 \right)^{(1+2r)/4} \left(\sum_{i\in I} 1^{\frac{2}{1-2r}} \right)^{(1-2r)/4} \\
&\leq 2^r \|g_\rho\|_K \|L_K - L_K^{\mathbf{x}}\|_{\mathrm{HS}}^{(1+2r)/2} (\#I)^{(1-2r)/4},
\end{aligned}$$

which verifies (39). The proof of the lemma is complete. ∎

We are now in a position to prove Theorem 15.

Proof of Theorem 15. By Lemmas 18 and 19, we know that for any $0 < \delta < \frac{1}{2}$ there exists a subset Z_δ of Z^m of measure at least $1 - 2\delta$ such that both (37) and (38) hold for each $\mathbf{z} \in Z_\delta$.

Let $\mathbf{z} \in Z_\delta$.

To prove $c_i^{\mathbf{z}} = 0$ for $i = p+1, \ldots, m$, we show that condition (50) for γ, to be defined below which is the same as condition (24) in the statement of the theorem after scaling δ to $\delta/2$, implies (26) and thereby (20) in Theorem 13, according to (22) in Lemma 14. To this end, we estimate $\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}|$ and $\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}| \cdot \sqrt{\lambda_i^{\mathbf{x}}}$.

We first apply Lemma 20 to the set $I = \{i\} \subset \{p+1, \ldots, m\}$ and Lemma 18 and see that in either case of $0 < r < 1/2$ and $r \geq 1/2$, there holds

$$\begin{aligned}
\sqrt{\lambda_i^{\mathbf{x}}} |\langle f_\rho, \phi_i^{\mathbf{x}} \rangle_K| &\leq \left(2^r + \sqrt{2}\lambda_1^{\max\{r-\frac{1}{2},0\}} \right) \|g_\rho\|_K \|L_K - L_K^{\mathbf{x}}\|_{\mathrm{HS}}^{\min\{\frac{1+2r}{2},1\}} + 2^r \|g_\rho\|_K (\lambda_i^{\mathbf{x}})^{\frac{1+2r}{2}} \\
&\leq C_1' \left(\lambda_i^{(1+2r)/2} + \left(\log \frac{2}{\delta} \right)^{(1+2r)/2} m^{-\min\{1/2,(1+2r)/4\}} \right),
\end{aligned}$$

where

$$C_1' = \left(2^r + \sqrt{2}\lambda_1^{\max\{r-\frac{1}{2},0\}} \right) (2\kappa)^{\min\{1+2r,2\}} \|g_\rho\|_K + 2^{2r+\frac{1}{2}} \|g_\rho\|_K (2\kappa+1)^{1+2r}.$$

This together with (38) in Lemma 19 gives

$$\begin{aligned}
&\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}| \leq \sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}} \rangle_K| + \sqrt{\lambda_i^{\mathbf{x}}} |\langle f_\rho, \phi_i^{\mathbf{x}} \rangle_K| \\
&\leq C_1' \lambda_i^{r+\frac{1}{2}} + \left(2\sqrt{2}M + C_1' \right) \left(\log \frac{2m}{\delta} \right)^{(1+2r)/2} m^{-\min\{1/2,(1+2r)/4\}} \\
&\leq \left(2\sqrt{2}M + 2C_1' \right) \left(\log \frac{2m}{\delta} \right)^{(1+2r)/2} \max\left\{ \left(\max\left\{ \lambda_p, \frac{1}{\sqrt{m}} \right\} \right)^{r+\frac{1}{2}}, \frac{1}{\sqrt{m}} \right\}. \qquad (47)
\end{aligned}$$

It follows that the first inequality $\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}| < \frac{C_{\Omega,1}}{2}\sqrt{\gamma}$ of (26) is valid if γ satisfies

$$\gamma > \left(\frac{4\sqrt{2}M + 4C_1'}{C_{\Omega,1}} \right)^2 \left(\log \frac{2m}{\delta} \right)^{1+2r} \max\left\{ \left(\max\left\{ \lambda_p, \frac{1}{\sqrt{m}} \right\} \right)^{2r+1}, \frac{1}{m} \right\}. \qquad (48)$$

Then we estimate $\lambda_i^{\mathbf{x}}$ by Lemma 18 as

$$\sqrt{\lambda_i^{\mathbf{x}}} \leq \sqrt{\lambda_i} + \sqrt{|\lambda_i - \lambda_i^{\mathbf{x}}|} \leq (2\kappa+1) \sqrt{\log \frac{2}{\delta}} \left(\max\left\{ \lambda_p, \frac{1}{\sqrt{m}} \right\} \right)^{\frac{1}{2}}.$$

Combining this with (47), we know that the second inequality $\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}}| \cdot \sqrt{\lambda_i^{\mathbf{x}}} < \frac{C_{\Omega,1}}{2}\gamma$ of (26) is valid if γ satisfies

$$\gamma > \frac{4\sqrt{2}M + 4C_1'}{C_{\Omega,1}} (2\kappa+1) \left(\log\frac{2m}{\delta}\right)^{1+r} \max\left\{ \left(\max\left\{\lambda_p, \frac{1}{\sqrt{m}}\right\}\right)^{r+1}, \frac{1}{\sqrt{m}}\left(\max\left\{\lambda_p, \frac{1}{\sqrt{m}}\right\}\right)^{\frac{1}{2}} \right\}. \tag{49}$$

Now we can choose the constant C_1 from (48) and (49) by

$$C_1 = \max\left\{ \left(\frac{4\sqrt{2}M + 4C_1'}{C_{\Omega,1}}\right)^2 (1+\lambda_1)^{2r+1}, \frac{4\sqrt{2}M + 4C_1'}{C_{\Omega,1}} (2\kappa+1)(1+\lambda_1)^{2r+1}, 1 \right\}.$$

With this choice, we know that for γ satisfying

$$\gamma \geq \begin{cases} C_1 \left(\log\frac{2m}{\delta}\right)^{1+2r} \left(\max\left\{\frac{\lambda_p}{\lambda_1}, \frac{1}{\sqrt{m}}\right\}\right)^{r+1}, & \text{if } 0 < r \leq \frac{1}{2}, \\ C_1 \left(\log\frac{2m}{\delta}\right)^{1+2r} \max\left\{ \left(\frac{\lambda_p}{\lambda_1}\right)^{r+\frac{1}{2}}, \frac{1}{\sqrt{m}} \right\} \left(\max\left\{\frac{\lambda_p}{\lambda_1}, \frac{1}{\sqrt{m}}\right\}\right)^{\frac{1}{2}}, & \text{if } r > \frac{1}{2}, \end{cases} \tag{50}$$

both (48) and (49) are valid, which implies (26). Then by (22) in Lemma 14, we see that condition (20) in Theorem 13 is valid and hence $c_i^{\mathbf{z}} = 0$ for $i = p+1, \ldots, m$.

Now we turn to the desired error bound. Assume (50) for γ. Define an index set $\mathcal{S}$ by $\mathcal{S} = \{i \in \{1, \cdots, p\} : \lambda_i^{\mathbf{x}} > \lambda_p/2\}$.

When $i \in \{1, \cdots, p\}$ but $\lambda_i^{\mathbf{x}} \leq \lambda_p/2$, we check the process in proving (47) and see from the restriction $\lambda_i^{\mathbf{x}} \leq \lambda_p/2$ that condition (50) for γ ensures (26). Then by (22) in Lemma 14, we see that condition (20) is valid for i. Hence $c_i^{\mathbf{z}} = 0$ for $i \in \mathbb{N}\backslash\mathcal{S}$. So we can expand $\|f_\rho - f^{\mathbf{z}}\|_K$ with respect to the orthonormal basis $\{\phi_i^{\mathbf{x}}\}$ of $\mathcal{H}_K$ as

$$\|f_\rho - f^{\mathbf{z}}\|_K^2 = \sum_{i\in\mathbb{N}\backslash\mathcal{S}} \left(\langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K\right)^2 + \sum_{i\in\mathcal{S}} \left(c_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K\right)^2. \tag{51}$$

For any $i \in \mathcal{S}$, we have $\lambda_i^{\mathbf{x}} > \lambda_p/2 > 0$ and

$$|c_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K| \leq |c_i^{\mathbf{z}} - S_i^{\mathbf{z}}| + \frac{\sqrt{\lambda_i^{\mathbf{x}}}|S_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K|}{\sqrt{\lambda_p/2}}.$$

Applying Theorem 13 (b), Lemma 14 and Lemma 19 gives

$$|c_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K| \leq C_{\Omega,2} \max\left\{ \left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}}, \frac{2\gamma}{\lambda_p} \right\} + \frac{2\sqrt{2}M}{\sqrt{m\lambda_p/2}} \sqrt{\log\frac{2m}{\delta}}.$$

It follows that

$$\sqrt{\sum_{i\in\mathcal{S}} \left(c_i^{\mathbf{z}} - \langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K\right)^2} \leq C_{\Omega,2}\sqrt{p} \left\{ \left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}} + \frac{2\gamma}{\lambda_p} \right\} + \frac{4\sqrt{p}M}{\sqrt{m\lambda_p}} \sqrt{\log\frac{2m}{\delta}}.$$

To estimate the first sum in (51) we cite an estimate from Guo and Zhou (2012) for the quantity $\left(\sum_{i\in\mathbb{N}\backslash\mathcal{S}} (\langle f_\rho, \phi_i^{\mathbf{x}}\rangle_K)^2\right)^{1/2}$ which is independent of the regularizing function Ω and know that it can be bounded by

$$\|g_\rho\|_K \lambda_{p+1}^r + 2^{\max\{r,1\}} \|g_\rho\|_K \lambda_p^{\min\{r-1,0\}} \left((\sum_{i=p+1}^\infty \lambda_i^{\max\{2r,2\}})^{\frac{1}{2}} + c_{r,\lambda_1} \|L_K - L_K^{\mathbf{x}}\|_{HS} \right),$$

where c_{r,λ_1} is the constant given by

$$c_{r,\lambda_1} = \begin{cases} 2^{1+r}\lambda_1^{r-1}, & \text{if } r \geq 1, \\ 2, & \text{if } r < 1. \end{cases}$$

Therefore

$$\begin{aligned} \|f^{\mathbf{z}} - f_\rho\|_K &\leq C_{\Omega,2}\sqrt{p} \left\{ \left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}} + \frac{2\gamma}{\lambda_p} \right\} + \|g_\rho\|_K \lambda_{p+1}^r + \frac{2\sqrt{p}M}{\sqrt{m\lambda_p}} \sqrt{\log\frac{2m}{\delta}} \\ &\quad + 2^{\max\{r,1\}} \|g_\rho\|_K \lambda_p^{\min\{r-1,0\}} \left(\left(\sum_{i=p+1}^\infty \lambda_i^{\max\{2r,2\}} \right)^{1/2} + \frac{4c_{r,\lambda_1}\kappa^2 \log\frac{2}{\delta}}{\sqrt{m}} \right) \\ &\leq C_{\Omega,2}\sqrt{p} \left\{ \left(\frac{2\gamma}{\lambda_p}\right)^{\frac{1}{2-q}} + \frac{2\gamma}{\lambda_p} \right\} + \|g_\rho\|_K \lambda_{p+1}^r + C_3 \frac{\sqrt{p}\log\frac{2m}{\delta}}{\sqrt{m}} \lambda_p^{\min\{-\frac{1}{2}, r-1\}} \\ &\quad + C_4 \lambda_p^{\min\{r-1,0\}} \left(\sum_{i=p+1}^\infty \lambda_i^{\max\{2r,2\}} \right)^{1/2}, \end{aligned}$$

where

$$C_3 = 2M\lambda_1^{\max\{\frac{1}{2}-r, 0\}} + 2^{\max\{r,1\}+2} \|g_\rho\|_K c_{r,\lambda_1} \kappa^2 \lambda_1^{\max\{r-\frac{1}{2}, 0\}}$$

and $C_4 = 2^{\max\{r,1\}} \|g_\rho\|_K$. After scaling δ to $\delta/2$, the proof of Theorem 15 is completed. ■

Appendix B. Minimax Lower Bounds

In this appendix, we derive a general minimax lower bound which includes Theorem 8 as a special case. First we define two sets of Borel probability measures.

Definition 21 *Let $\mathcal{P}(\alpha_1, \alpha_2, r, M, R, D_1, D_2)$ be the set of all Borel probability measures ρ on $X \times Y$ satisfying the following three conditions:*

1. *$|y| \leq M$ almost surely,*
2. *$f_\rho = L_K^r(g_\rho)$ for some $g_\rho \in \mathcal{H}_K$ with $\|g_\rho\|_K \leq R$,*
3. *$D_1 i^{-\alpha_1} \leq \lambda_i \leq D_2 i^{-\alpha_2}$ for each i.*

Let $\mathcal{P}(\beta_1,\beta_2,r,M,R,D_1,D_2)$ be the same as $\mathcal{P}(\alpha_1,\alpha_2,r,M,R,D_1,D_2)$ except that the last condition is replaced by $D_1\beta_1^{-i}\le\lambda_i\le D_2\beta_2^{-i}$ for each i.

For simplicity, we abbreviate these two sets as $\mathcal{P}(\alpha_1,\alpha_2,r)$ and $\mathcal{P}(\beta_1,\beta_2,r)$, respectively. Now we state the general minimax lower bound for the error in the $\mathcal{H}_K$-metric following the idea of (Caponnetto and De Vito, 2007). Our proof is mainly based on Lemma 2.9, Theorem 2.5 and the approach from (Tsybakov, 2009).

Theorem 22 *Assume $R>0$ and $M\ge 4\kappa^{2r+1}R$. Let $f^{\mathbf{z}}\in\mathcal{H}_K$ be the output of an arbitrary learning algorithm based on the sample $\boldsymbol{z}=\{(x_i,y_i)\}_{i=1}^m$. Then for every $0<\delta<\frac{1}{8}$, there exist positive constants τ_1,τ_2, independent of δ or m, such that*

$$\lim_{m\to\infty}\inf_{f^{\mathbf{z}}}\sup_{\rho\in\mathcal{P}(\alpha_1,\alpha_2,r)}\mathbb{P}_{\mathbf{z}\sim\rho^m}\left\{\|f^{\mathbf{z}}-f_\rho\|_K\ge\tau_1\delta^{\frac{\alpha_1 r}{\alpha_2(2r+1)+1}}m^{-\frac{\alpha_1 r}{\alpha_2(2r+1)+1}}\right\}\ge 1-2\delta \tag{52}$$

and

$$\lim_{m\to\infty}\inf_{f^{\mathbf{z}}}\sup_{\rho\in\mathcal{P}(\beta_1,\beta_2,r)}\mathbb{P}_{\mathbf{z}\sim\rho^m}\left\{\|f^{\mathbf{z}}-f_\rho\|_K\ge\tau_2\delta^{\frac{1}{2}}m^{-\frac{1}{2}}\sqrt{\log m}\right\}\ge 1-2\delta. \tag{53}$$

Proof First, we associate a probability measure $\rho_f\in\mathcal{P}(\alpha_1,\alpha_2,r)$ to a pair (μ,f) where μ is a Borel measure on Y such that the eigenvalues of the associated integral operator L_K satisfy $D_1 i^{-\alpha_1}\le\lambda_i\le D_2 i^{-\alpha_2}$, and $f=L_K^r g$ for some $g\in\mathcal{H}_K$ with $\|g\|_K\le R$. Define a probability measure ρ_f by

$$d\rho_f(x,y)=\left[\frac{B+f(x)}{2B}d\delta_B(y)+\frac{B-f(x)}{2B}d\delta_{-B}(y)\right]d\mu(x),$$

where $B=4\kappa^{2r+1}R$ and $d\delta_\xi$ denotes the Dirac delta with unit mass at ξ. By the reproducing property, $\|f\|_\infty\le\kappa\|L_K^r g\|_K\le\kappa^{2r+1}R=\frac{B}{4}$. It follows that ρ_f is a probability measure on $X\times Y$ with μ being the marginal distribution and f the regression function. Moreover, $M\ge 4\kappa^{2r+1}R$ ensures $|y|\le M$ almost surely. Hence $\rho_f\in\mathcal{P}(\alpha_1,\alpha_2,r)$.

Then we construct a finite sequence $f_0,...,f_N$ in the set $\{L_K^r g: g\in\mathcal{H}_K,\|g\|_K\le R\}$ based on the Varshamov-Gilbert bound (Lemma 2.9 in (Tsybakov, 2009)) which asserts that for any integer $\gamma\ge 8$, there exists a set $\Theta=\{w_0,w_1,...,w_N\}\subset\{0,1\}^\gamma$ such that

1. $w_0=(0,...,0)$.
2. For any $i\ne j$, $H(w_i,w_j)>\gamma/8$, where $H(\cdot,\cdot)$ is the Hamming distance.
3. $N\ge 2^{\gamma/8}$.

For $0<\delta<\frac{1}{8}$, let γ be the smallest integer greater than or equal to $c_\delta m^{\frac{1}{\alpha_2(2r+1)+1}}$ with a constant $c_\delta>0$ to be specified later. For $w_i=(w_i^{\gamma+1},...,w_i^{2\gamma})\in\Theta$ with $i\in\{0,\dots,N\}$, define $f_i=L_K^r g_i$ with

$$g_i=\sum_{k=\gamma+1}^{2\gamma}w_i^k R\gamma^{-\frac{1}{2}}\phi_k.$$

Note that $g_i\in\mathcal{H}_K$ and

$$\|g_i\|_K^2=\left\|\sum_{k=\gamma+1}^{2\gamma}w_i^k R\gamma^{-\frac{1}{2}}\phi_k\right\|_K^2=\sum_{k=\gamma+1}^{2\gamma}(w_i^k)^2\gamma^{-1}R^2\|\phi_k\|_K^2\le R^2.$$

Hence $\{f_0,...,f_N\}\subset\{L_K^r g: g\in\mathcal{H}_K,\|g\|_K\le R\}$, which implies $\{\rho_{f_0},...,\rho_{f_N}\}\subset\mathcal{P}(\alpha_1,\alpha_2,r)$.

Now we adopt Theorem 2.5 in (Tsybakov, 2009) to establish our desired lower bound.

Observe that for $0\le i<j\le N$, the Kullback-Leibler distance $\mathcal{D}_{KL}(\rho_{f_i}\|\rho_{f_j})$ between ρ_{f_i} and ρ_{f_j} satisfies

$$\begin{aligned}
&\mathcal{D}_{KL}(\rho_{f_i}\|\rho_{f_j})\\
&=\int_X\left\{\frac{B+f_i(x)}{2B}\ln\left(1+\frac{f_i(x)-f_j(x)}{B+f_j(x)}\right)+\frac{B-f_i(x)}{2B}\ln\left(1-\frac{f_i(x)-f_j(x)}{B-f_j(x)}\right)\right\}d\mu(x)\\
&\le\frac{f_i(x)-f_j(x)}{2B}\left\{\frac{B+f_i(x)}{B+f_j(x)}-\frac{B-f_i(x)}{B-f_j(x)}\right\}\\
&\le\frac{16}{15B^2}\lambda_\gamma^{2r+1}R^2\gamma^{-1}\sum_{k=\gamma+1}^{2\gamma}(w_i^k-w_j^k)^2\le\frac{D_2^{2r+1}}{15\kappa^{4r+2}}\gamma^{-\alpha_2(2r+1)},
\end{aligned}$$

which implies

$$\begin{aligned}
\frac{1}{N}\sum_{i=1}^N\mathcal{D}_{KL}((\rho_{f_i})^m\|(\rho_{f_0})^m) &\le \frac{D_2^{2r+1}}{15\kappa^{4r+2}}m\gamma^{-\alpha_2(2r+1)}\le\frac{D_2^{2r+1}}{15\kappa^{4r+2}}c_\delta^{-\alpha_2(2r+1)}m^{\frac{1}{\alpha_2(2r+1)+1}}\\
&\le\frac{D_2^{2r+1}}{15\kappa^{4r+2}c_\delta^{1+\alpha_2(2r+1)}}\gamma=\delta\log 2^{\gamma/8}\le\delta\log N
\end{aligned}$$

by taking

$$c_\delta=\left(\frac{8D_2^{2r+1}}{15\kappa^{4r+2}\log 2}\right)^{1/(\alpha_2(2r+1)+1)}\delta^{-1/(\alpha_2(2r+1)+1)}.$$

On the other hand, for any $0\le i<j\le N$.

$$\begin{aligned}
\|f_i-f_j\|_K^2 &= \|L_K^r(g_i-g_j)\|_K^2\\
&= \sum_{k=\gamma+1}^{2\gamma}R^2\gamma^{-1}\lambda_k^{2r}(w_i^k-w_j^k)^2\\
&\ge R^2\gamma^{-1}\lambda_{2\gamma}^{2r}\sum_{k=\gamma+1}^{2\gamma}(w_i^k-w_j^k)^2\\
&= R^2\gamma^{-1}\lambda_{2\gamma}^{2r}H(w_i,w_j)\\
&\ge \frac{R^2\lambda_{2\gamma}^{2r}}{8}\\
&\ge 2^{-(2\alpha_1 r+3)}R^2D_1^{2r}\gamma^{-2\alpha_1 r}\\
&\ge 2\tau_1^2\delta^{\frac{2\alpha_1 r}{\alpha_2(2r+1)+1}}m^{-\frac{2\alpha_1 r}{\alpha_2(2r+1)+1}}
\end{aligned}$$

for some constant $\tau_1 > 0$.

Therefore, as shown in (Tsybakov, 2009) we have

$$\inf_{f^{\mathbf{z}}} \sup_{\rho \in \mathcal{P}(\alpha_1,\alpha_2,r)} \mathbb{P}_{\mathbf{z}\sim\rho^m} \left\{ \|f^{\mathbf{z}} - f_\rho\|_K \geq \tau_1 \delta^{\frac{\alpha_1 r}{\alpha_2(2r+1)+1}} m^{-\frac{\alpha_1 r}{\alpha_2(2r+1)+1}} \right\}$$
$$\geq \frac{\sqrt{N}}{\sqrt{N}+1} \left(1 - 2\delta - \sqrt{\frac{2\delta}{\log N}} \right).$$

This completes the proof for the statement about $\mathcal{P}(\alpha_1, \alpha_2, r)$. The proof for the statement about $P(\beta_1, \beta_2, r)$ is similar. The proof of the theorem is complete. ■

Appendix C. Proof of Proposition 10

(a). Let $0 \leq \xi_1 < \xi_2 < \xi_3 < \xi_4$. For $i = 2$ or 3, one has

$$\Omega(\xi_i) \geq \frac{(\xi_{i+1} - \xi_i)\Omega(\xi_{i-1})}{\xi_{i+1} - \xi_{i-1}} + \frac{(\xi_i - \xi_{i-1})\Omega(\xi_{i+1})}{\xi_{i+1} - \xi_{i-1}}. \tag{54}$$

Let $i = 2$ in (54) to give

$$\begin{aligned} (\xi_2 - \xi_1)\Omega(\xi_3) &\leq (\xi_3 - \xi_1)\Omega(\xi_2) - (\xi_3 - \xi_2)\Omega(\xi_1) \\ &= \xi_3[\Omega(\xi_2) - \Omega(\xi_1)] - \xi_1\Omega(\xi_2) + \xi_2\Omega(\xi_1). \end{aligned} \tag{55}$$

If $\Omega(\xi_2) < \Omega(\xi_1)$, let $\xi_3 \to \infty$ to give $\Omega(\xi_3) \to -\infty$, which contradicts $\Omega([0,\infty)) \subset [0,\infty)$. So Ω is nondecreasing. Similarly we have $(\xi_3 - \xi_1)\Omega(\xi_2) \geq (\xi_3 - \xi_1 + \xi_1 - \xi_2)\Omega(\xi_1) + (\xi_2 - \xi_1)\Omega(\xi_3)$, so

$$\frac{\Omega(\xi_2) - \Omega(\xi_1)}{\xi_2 - \xi_1} \geq \frac{\Omega(\xi_3) - \Omega(\xi_1)}{\xi_3 - \xi_1} \geq 0. \tag{56}$$

If $\Omega(\xi_2) = 0$, since Ω is nondecreasing, $\Omega(\xi_1) = 0$ for all $0 \leq \xi_1 < \xi_2$, and (56) gives $\Omega(\xi_3) = 0$ for all $\xi_3 > \xi_2$, so we have $\Omega = 0$, a contradiction. Therefore $\Omega(c) > 0$ for $c > 0$.

From (56), the function $[\Omega(c) - \Omega(\xi_1)]/(c - \xi_1)$ of c is nonincreasing when $c > \xi_1$, so the right-hand derivative Ω'_+ is well-defined on $[0,\infty)$, taking values in $[0,\infty]$. We get from (55) that

$$\infty > \frac{\Omega(\xi_2) - \Omega(\xi_1)}{\xi_2 - \xi_1} \geq \frac{\Omega(\xi_3) - \Omega(\xi_2)}{\xi_3 - \xi_2}. \tag{57}$$

Let $\xi_3 \to \xi_2^+$ to give $\Omega'_+(\xi_2) < \infty$. Therefore $\Omega'_+(c)$ is finite for $c \in (0,\infty)$. Let $i = 3$. We have the analogue of (57),

$$\frac{\Omega(\xi_3) - \Omega(\xi_2)}{\xi_3 - \xi_2} \geq \frac{\Omega(\xi_4) - \Omega(\xi_3)}{\xi_4 - \xi_3}, \tag{58}$$

which, together with (57), gives that as $\xi_2 \to \xi_1^+$ and $\xi_4 \to \xi_3^+$, $\Omega'_+(\xi_1) \geq \Omega'_+(\xi_3)$. This proves that Ω'_+ is nonincreasing on $[0,\infty)$. If $\Omega'_+(0) = 0$, since $0 \leq \frac{\Omega(c)-\Omega(0)}{c-0}$ is nonincreasing

for $c > 0$ as we proved before, we have $\Omega(c) = 0$ for all $c > 0$, a contradiction again. So $\Omega'_+(0) \in (0,\infty]$.

(b). Let $\xi_1 = 0$ and $\xi_3 = 1$, then (55) gives $\Omega(\xi_2) \geq \xi_2\Omega(1)$, so for all $c \in (0,1]$, $\Omega(c) \geq c\Omega(1)$. In (55) let $\xi_1 = 0$ and $\xi_2 = 1$ to give $\Omega(\xi_3) \leq \xi_3\Omega(1)$, so for any $c \in [1,\infty)$, $\Omega(c) \leq \Omega(1)c$.

The properties stated in (c) and (d) follow immediately from the concavity of the function Ω.

(e). Write the function $\frac{\Omega(c)}{c^2}$ as $\frac{\Omega(c)}{c} \cdot \frac{1}{c}$. We see from (d) that this function is strictly decreasing on $(0,\infty)$. By (a), we obtain the limit $\lim_{c\to 0^+} \frac{\Omega(c)}{c^2} \geq \underline{\lim}_{c\to 0^+} \frac{\Omega(1)c}{c^2} = +\infty$. By (b), $\lim_{c\to\infty} \frac{\Omega(c)}{c^2} \leq \overline{\lim}_{c\to\infty} \frac{\Omega(1)c}{c^2} = 0$. The proof of Proposition 10 is complete. ■

References

F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *Journal of Complexity*, 23:52-72, 2007.

G. Blanchard, P. Massart, R. Vert, and L. Zwald. Kernel projection machine: a new tool for pattern recognition. *Advances in Neural Information Processing Systems*, 1649-1656, 2004.

G. Blanchard and L. Zwald. Finite dimensional projection for classification and statistical learning. *IEEE Transactions on Information Theory*, 54: 4169- 4182, 2008.

A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7:331-368, 2007.

A. Caponnetto and Y. Yao. Cross-validation based adaptation for regularization operators in learning theory. *Analysis and Applications*, 8:161-183, 2010.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, 2005.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

R. DeVore, G. Kerkyacharian, D. Picard, and V. Temlyakov. Mathematical methods for supervised learning. *IMI Preprints*, 22:1-51, 2004.

D. E. Edmunds and H. Triebel. *Function Spaces, Entropy Numbers and Differential Operators*. Cambridge University Press, Cambridge, 1996.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348-1360, 2001.

I. E. Frank and J. H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35: 109–148, 1993.

W. Fu and K. Knight. Asymptotics for lasso-type estimators. *Annals of Statistics* 28:1356-1378, 2000.

X. Guo and D. X. Zhou. An empirical feature-based learning algorithm producing sparse approximations. *Applied and Computational Harmonic Analysis*, 32:389-400, 2012.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 89:10915-9.

T. Hu, J. Fan, Q. Wu, and D. X. Zhou. Regularization schemes for minimum error entropy principle. *Analysis and Applications*, 13:437–455, 2015.

V. Koltchinskii and E. Giné. Random matrix approximation of spectra of integral operators. *Bernoulli*, 6:113-167, 2000.

T. Kühn. Entropy numbers in sequence spaces with an application to weighted function spaces. *Journal of Approximation Theory*, 153:40–52, 2008.

Y. Liu, H. H. Zhang, C. Park, and J. Ahn. Support vector machines with adaptive Lq penalties. *Computational Statistics and Data Analysis*, 51:6380-6394, 2007.

L. Lo Gerfo, L. Rosasco, F. Odone, E. De Vito and A. Verri. Spectral algorithms for supervised learning. *Neural Computation*, 20:1873-1897, 2008.

M. Nielsen and O. Lund. NN-align. An artificial neural network-based alignment algorithm for MHC class II peptide binding prediction. *BMC Bioinformatics*, 10:296, 2009.

M. Nielsen, C. Lundegaard, T. Blicher, B. Peters, A. Sette, S. Justesen, S. Buus, and O. Lund. Quantitative predictions of peptide binding to any HLA-DR molecule of known sequence: NetMHCIIpan. *PLOS Computational Biology*, 4:e1000107, 2008.

G. Raskutti, M. J. Wainwright, and B. Yu. Minimax-optimal rates for sparse additive models over kernel classes via convex programming. *Journal of Machine Learning Research*, 13:389–427, 2012.

J.B. Reade. Eigenvalues of positive definite kernels II. *SIAM Journal on Mathematical Analysis*, 15:137-142, 1984.

J.B. Reade. Eigenvalues of analytic kernels. *SIAM Journal on Mathematical Analysis*, 15:133-136, 1984.

B. Schölkopf, A. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

W.J. Shen, H.S. Wong, Q.W. Xiao, X. Guo, and S. Smale. Introduction to the peptide binding problem of computational immunology: new results. *Foundations of Computational Mathematics*, 14:951–984, 2014.

S. Smale and D.X. Zhou. Learning theory estimates via integral operators and their approximations. *Constructive Approximation*, 26:153-172, 2007.

S. Smale and D. X. Zhou. Online learning with Markov sampling. *Analysis and Applications*, 7:87-113, 2009.

I. Steinwart, D. Hush, and C. Scovel. Optimal rates for regularized least squares regression. *In Proceedings of the Annual Conference on Learning Theory*, 79-93, 2009.

T. Suzuki, R. Tomioka, and M. Sugiyama. Fast learning rate of multiple kernel learning: trade-off between sparsity and smoothness. *JMLR Workshop and Conference Proceedings*, 22: 1152-1183, 2012.

A. Tsybakov. *Introduction to Nonparametric Estimation*. New York, Springer, 2009.

Z. B. Xu, X. Y. Chang, F. M. Xu. L-1/2 regularization: A thresholding representation theory and a fast solver. *IEEE Transactions on Neural Networks and Learning Systems*, 23:1013-1027, 2012.

Y. Yang and A. Barron. Information-theoretic determination of minimax rates of convergence. *The Annals of Statistics*, 27(5): 1564–1599, 1999.

D. X. Zhou. Capacity of reproducing kernel spaces in learning theory. *IEEE Transactions on Information Theory*, 49:1743-1752, 2003.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.

L. Zwald. Performances statistiques d'algorithmes d'apprentissage: Kernel Projection Machine et analyse en composantes principales à noyau. PhD thesis, Université Paris-Sud 11, 2005.

L. Zwald and G. Blanchard. On the convergence of eigenspaces in kernel principal component analysis. *In Advances in Neural Information Processing Systems 18 (Y. Weiss, B. Schölkopf, and J. Platt, eds.)*, 1649-1656. MIT Press, Cambridge, MA, 2006.

Journal of Machine Learning Research 17 (2016) 1-29 Submitted 10/14; Revised 4/15; Published 6/16

Estimating Diffusion Networks: Recovery Conditions, Sample Complexity & Soft-thresholding Algorithm

Manuel Gomez-Rodriguez[◇†] MANUELGR@MPI-SWS.ORG
MPI for Software Systems
Paul-Ehrlich-Strasse, 67663 Kaiserslautern
Germany

Le Song[◇] LSONG@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Hadi Daneshmand SEYED.DANESHMAND@INF.ETHZ.CH
Computer Science Department
Universitätstrasse 6, 8092 Zürich
Switzerland

Bernhard Schölkopf BS@TUE.MPG.DE
MPI for Intelligent Systems
Spemannstrasse 38, 72076 Tübingen
Germany

Editor: Edo Airoldi

Abstract

Information spreads across social and technological networks, but often the network structures are hidden from us and we only observe the traces left by the diffusion processes, called *cascades*. Can we recover the hidden network structures from these observed cascades? What kind of cascades and how many cascades do we need? Are there some network structures which are more difficult than others to recover? Can we design efficient inference algorithms with provable guarantees?

Despite the increasing availability of cascade data and methods for inferring networks from these data, a thorough theoretical understanding of the above questions remains largely unexplored in the literature. In this paper, we investigate the network structure inference problem for a general family of continuous-time diffusion models using an ℓ_1-regularized likelihood maximization framework. We show that, as long as the cascade sampling process satisfies a natural incoherence condition, our framework can recover the correct network structure with high probability if we observe $O(d^3 \log N)$ cascades, where d is the maximum number of parents of a node and N is the total number of nodes. Moreover, we develop a simple and efficient soft-thresholding network inference algorithm which demonstrate the match between our theoretical prediction and empirical results. In practice, this new algorithm also outperforms other alternatives in terms of the accuracy of recovering hidden diffusion networks.

. [◇] These authors contributed equally to this work.
. [†] This work was done while Manuel Gomez-Rodriguez was still affiliated with the MPI for Intelligent Systems, Spemannstr. 38, 72076 Tuebingen, Germany
. Preliminary version of this work appeared in proceedings of the 31st International Conference on Machine Learning (ICML '14), 2014.

1. Introduction

Diffusion of information, behaviors, diseases, or irrepresentabilitymore generally, *contagions* can be naturally modeled as a stochastic process that occur over the edges of an underlying network (Rogers, 1995). In this scenario, we often observe the temporal traces that the diffusion generates, called *cascades*, but the edges of the network that gave rise to the diffusion remain unobservable (Adar and Adamic, 2005). For example, blogs or media sites often publish a new piece of information without explicitly citing their sources. Marketers may note when a social media user decides to adopt a new behavior but cannot tell which neighbor in the social network influenced them to do so. Epidemiologist observe when a person gets sick but usually cannot tell who infected her. In all these cases, given a set of cascades and a diffusion model, the network inference problem consists of inferring the edges (and model parameters) of the unobserved underlying network (Gomez-Rodriguez, 2013).

The network inference problem has attracted significant attention in recent years (Saito et al., 2009; Gomez-Rodriguez et al., 2010, 2011, 2013b, 2014; Snowsill et al., 2011; Du et al., 2012a, 2013; Zhou et al., 2013), since it is essential to reconstruct and predict the paths over which information can spread, and to maximize sales of a product or stop infections. Most previous work has focused on developing network inference algorithms and evaluating their performance experimentally on different synthetic and real networks, and a rigorous theoretical analysis of the problem has been missing. However, such analysis is of outstanding interest since it would enable us to answer many fundamental open questions. For example, which conditions are sufficient to guarantee that we can recover a network given a large number of cascades? If these conditions are satisfied, how many cascades are sufficient to infer the network with high probability? Until recently, there has been only two pieces of work along this direction (Netrapalli and Sanghavi, 2012; Abrahao et al., 2013), which leverage less realistic diffusion models. Moreover, none of them is able to identify a recovery condition relating the interaction between the network structure and the cascade sampling process, which we make precise in our paper.

1.1 Overview of results

We consider the network inference problem under the continuous-time diffusion model recently introduced by Gomez-Rodriguez et al. (2011), which has been extensively validated in real diffusion data, and, due to its flexibility, has been extended to support textual information (Wang et al., 2012), nonparametric pairwise likelihoods (Du et al., 2012a), topic modeling (Du et al., 2012b), dynamic networks (Gomez-Rodriguez et al., 2013a). We identify a natural irrepresentability condition for such a model which depends on both the network structure, the diffusion parameters and the sampling process of the cascades. This condition captures the intuition that we can recover the network structure if the co-occurrence of a node and its non-parent nodes is small in the cascades. Furthermore, we show that, if this condition holds for the population case, we can recover the network structure using

an ℓ_1-regularized maximum likelihood estimator and $O(d^3 \log N)$ cascades, where N is the number of nodes in the network and d is the maximum number of parents of a node, with the probability of success approaching 1 in a rate exponential in the number of cascades. Importantly, if this condition also holds for the finite sample case, then the guarantee can be improved to $O(d^2 \log N)$ cascades. Beyond theoretical results, we also propose a new, efficient and simple proximal gradient algorithm to solve the ℓ_1-regularized maximum likelihood estimation. The algorithm is especially well-suited for our problem since it is highly scalable and naturally finds sparse estimators, as desired, by using soft-thresholding. Using this algorithm, we perform various experiments illustrating the consequences of our theoretical results and demonstrating that it typically outperforms other state-of-the-art algorithms.

1.2 Related work

Netrapalli and Sanghavi (2012) propose a maximum likelihood network inference method for a variation of the discrete-time independent cascade model (Kempe et al., 2003) and show that, for general networks satisfying a *correlation decay*, the estimator recovers the network structure given $O(d^2 \log N)$ cascades, and the probability of success is approaching 1 in a rate exponential in the number of cascades. The rate they obtained is on a par with our results. However, their discrete diffusion model is less realistic in practice, and the correlation decay condition implies that, on average, each node can only infect one single node per cascade. Instead, we use a general continuous-time diffusion model (Gomez-Rodriguez et al., 2011), which has been extensively validated in real diffusion data and extended in various ways by different authors (Wang et al., 2012; Du et al., 2012a,b).

Abrahao et al. (2013) propose a simple network inference method, First-Edge, for a slightly different continuous-time independent cascade model (Gomez-Rodriguez et al., 2010), and show that, for general networks, if the cascade sources are chosen uniformly at random, the algorithm needs $O(Nd \log N)$ cascades to recover the network structure and the probability of success is approaching 1 in a rate polynomial in the number of cascades. Additionally, they study trees and bounded-degree networks and show that, if the cascade sources are chosen uniformly at random, the error decreases polynomially as long as $O(\log N)$ and $\Omega(d^9 \log^2 d \log N)$ cascades are recorded respectively. In our work, we show that, for general networks satisfying a natural irrepresentability condition, our method outperforms the First-Edge algorithm and the algorithm for bounded-degree networks in terms of rate and sample complexity.

Gripon and Rabbat (2013) propose a network inference method for unordered cascades, in which nodes that are infected together in the same cascade are connected by a path containing exactly the nodes in the trace, and give necessary and sufficient conditions for network inference. However, they consider a restrictive scenario in which cascades are all three nodes long.

2. Continuous-Time Diffusion Model

In this section, we revisit the continuous-time generative model for cascade data introduced by Gomez-Rodriguez et al. (2011). The model associates each edge $j \to i$ with a transmission function, $f(t_i|t_j;\alpha_{ji}) = f(t_i - t_j;\alpha_{ji})$, a density over time parameterized by α_{ji}. This is

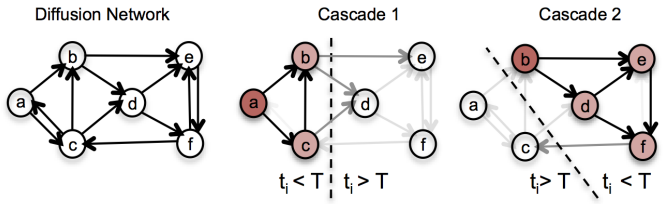


Figure 1: The diffusion network structure (left) is unknown and we only observe cascades, which are N-dimensional vectors recording the times when nodes get infected by contagions that spread (right). Cascade 1 is $(t_a, t_b, t_c, \infty, \infty, \infty)$, where $t_a < t_c < t_b$, and cascade 2 is $(\infty, t_b, \infty, t_d, t_e, t_f)$, where $t_b < t_d < t_e < t_f$. Each cascade contains a source node (dark red), drawn from a source distribution $\mathbb{P}(s)$, as well as infected (light red) and uninfected (white) nodes, and it provides information on black and dark gray edges but does not on light gray edges.

in contrast to previous discrete-time models which associate each edge with a fixed infection probability (Kempe et al., 2003). Moreover, it also differs from discrete-time models in the sense that events in a cascade are not generated iteratively in rounds, but event timings are sampled directly from the transmission functions in the continuous-time model.

2.1 Cascade generative process

Given a *directed* contact network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N nodes, the process begins with an infected source node, s, initially adopting certain *contagion* (idea, meme or product) at time zero, which we draw from a source distribution $\mathbb{P}(s)$. The contagion is transmitted from the source along her out-going edges to her direct neighbors. Each transmission through an edge entails a *random* transmission time, τ, drawn from an associated pairwise transmission likelihood $f(\tau;\alpha_{ji})$. We assume transmission times are independent, possibly distributed differently across edges, and, in some cases, can be arbitrarily large, $\tau \to \infty$. Then, the infected neighbors transmit the contagion to their respective neighbors, and the process continues. We assume that an infected node remains infected for the entire diffusion process. Thus, if a node i is infected by multiple neighbors, only the neighbor that first infects node i will be the *true parent*. As a result, although the contact network can be an arbitrary directed network, each contagion induces a Directed Acyclic Graph (DAG). Figure 1 illustrates the process and Table 1 gives several examples of well-known parametric transmission likelihoods (Gomez-Rodriguez et al., 2011, 2013a, 2014).

2.2 Cascade data

Observations from the model are recorded as a set C^n of cascades $\{\mathbf{t}^1, \ldots, \mathbf{t}^n\}$. Each cascade $\mathbf{t}^c$ is an N-dimensional vector $\mathbf{t}^c := (t_1^c, \ldots, t_N^c)$ recording when nodes are infected, $t_k^c \in [0, T^c] \cup \{\infty\}$. Symbol ∞ labels nodes that are not infected during observation window $[0, T^c]$ – it does not imply they are never infected. The 'clock' is reset to 0 at the start of each

Model	Transmission functions $f(\tau;\alpha_{ji})$	Log survival $y(t+\tau\|t;\alpha_{ji}) = \log S(t+\tau\|t;\alpha_{ji})$	Hazard $H(t+\tau\|t;\alpha_{ji})$
Exp	$\begin{cases} \alpha_{j,i}\cdot e^{-\alpha_{j,i}\tau} & \text{if } \tau \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$-\alpha_{j,i}\tau$	$\alpha_{j,i}$
Pow	$\begin{cases} \frac{\alpha_{j,i}}{\delta}\left(\frac{\tau}{\delta}\right)^{-1-\alpha_{j,i}} & \text{if } \tau \geq \delta \\ 0 & \text{otherwise} \end{cases}$	$-\alpha_{j,i}\log\left(\frac{\tau}{\delta}\right)$	$\frac{\alpha_{j,i}}{\tau}$
Ray	$\begin{cases} \alpha_{j,i}\tau e^{-\frac{1}{2}\alpha_{j,i}\tau^2} & \text{if } \tau \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$-\alpha_{j,i}\frac{\tau^2}{2}$	$\alpha_{j,i}\tau$

Table 1: Pairwise transmission models

cascade. We assume $T^c = T$ for all cascades; the results generalize trivially. Contagions often propagate simultaneously (Myers and Leskovec, 2012; Prakash et al., 2012) over the same network but we assume each contagion to propagate independently of each other. Finally, we also assume that all activated nodes except the first one are activated by network diffusion, *i.e.*, by previously activated nodes, ignoring external influences (Myers et al., 2012). Refer to Figure 1 for an example.

2.3 Likelihood of a cascade

Gomez-Rodriguez et al. (2011) showed that the likelihood of a cascade $\mathbf{t}$ under the continuous-time independent cascade model is

$$f(\mathbf{t};\mathbf{A}) = \prod_{t_i \leq T}\prod_{t_m > T} S(T|t_i;\alpha_{im}) \times \prod_{k:t_k<t_i} S(t_i|t_k;\alpha_{ki}) \sum_{j:t_j<t_i} H(t_i|t_j;\alpha_{ji}), \tag{1}$$

where $\mathbf{A} = \{\alpha_{ji}\}$ denotes the collection of parameters, $S(t_i|t_j;\alpha_{ji}) = 1 - \int_{t_j}^{t_i} f(t-t_j;\alpha_{ji})\,dt$ is the survival function and $H(t_i|t_j;\alpha_{ji}) = f(t_i - t_j;\alpha_{ji})/S(t_i|t_j;\alpha_{ji})$ is the hazard function. The survival terms in the first line account for the probability that uninfected nodes survive to all infected nodes in the cascade up to T and the survival and hazard terms in the second line account for the likelihood of the infected nodes. The survival and hazard functions are simple for several well-known parametric transmission likelihoods, as shown in Table 1. Then, assuming cascades are sampled independently, the likelihood of a set of cascades is the product of the likelihoods of individual cascades given by Eq. 1. For notational simplicity, we define $y(t_i|t_k;\alpha_{ki}) := \log S(t_i|t_k;\alpha_{ki})$, and $h(\boldsymbol{t};\boldsymbol{\alpha}_i) := \sum_{k:t_k\leq t_i} H(t_i|t_k;\alpha_{ki})$ if $t_i \leq T$ and 0 otherwise.

3. Network Inference Problem

Consider an instance of the continuous-time diffusion model defined above with a contact network $\mathcal{G}^* = (\mathcal{V}^*, \mathcal{E}^*)$ and associated parameters $\left\{\alpha^*_{ji}\right\}$. We denote the set of parents of node i as $\mathcal{N}^-(i) = \{j \in \mathcal{V}^* : \alpha^*_{ji} > 0\}$ with cardinality $d_i = |\mathcal{N}^-(i)|$ and the minimum positive transmission rate as $\alpha^*_{\min,i} = \min_{j:\alpha^*_{ji}>0} \alpha^*_{ji}$. Let C^n be a set of n cascades sampled from the model, where the source $s \in \mathcal{V}^*$ of each cascade is drawn from a source distribution

Function	Infected node ($t_i < T$)	Uninfected node ($t_i > T$)
$g_i(\mathbf{t};\boldsymbol{\alpha})$	$\log h(\mathbf{t};\boldsymbol{\alpha}) + \sum_{j:t_j<t_i} y(t_i\|t_j;\alpha_j)$	$\sum_{j:t_j<T} y(T\|t_j;\alpha_j)$
$[\nabla y_i(\mathbf{t};\boldsymbol{\alpha})]_k$	$-y'(t_i\|t_k;\alpha_k)$	$-y'(T\|t_k;\alpha_k)$
$[D(\mathbf{t};\boldsymbol{\alpha})]_{kk}$	$-y''(t_i\|t_k;\alpha_k) - h(\mathbf{t};\boldsymbol{\alpha})^{-1}H''(t_i\|t_k;\alpha_k)$	$-y''(T\|t_k;\alpha_k)$

Table 2: Functions. $g_i(\mathbf{t};\boldsymbol{\alpha})$ is node i's log-likelihood in a cascade $\mathbf{t}$, $y_i(\mathbf{t};\boldsymbol{\alpha})$ is the logarithm of node i's survivals in a cascade $\mathbf{t}$, $D(\mathbf{t};\boldsymbol{\alpha})$ is a diagonal matrix defined in Eq. 5, $H(t_i|t_k;\alpha_k)$ is the hazard function, and $h(\mathbf{t};\boldsymbol{\alpha})$ denotes the sum of node i's hazard functions in a cascade $\mathbf{t}$.

$\mathbb{P}(s)$. Then, the network inference problem consists of finding the directed edges and the associated parameters using only the temporal information from the set of cascades C^n.

This problem has been cast as a maximum likelihood estimation problem (Gomez-Rodriguez et al., 2011)

$$\begin{array}{ll} \text{minimize}_{\mathbf{A}} & -\frac{1}{n}\sum_{c\in C^n} \log f(\mathbf{t}^c;\mathbf{A}) \\ \text{subject to} & \alpha_{ji} \geq 0,\ i,j = 1,\ldots,N, i \neq j, \end{array} \tag{2}$$

where the inferred edges in the network correspond to those pairs of nodes with non-zero parameters, *i.e.* $\hat{\alpha}_{ji} > 0$.

In fact, the problem in Eq. 2 decouples into a set of independent smaller subproblems, one per node, where we infer the parents of each node and the parameters associated with these incoming edges. Without loss of generality, for a particular node i, we solve the problem

$$\begin{array}{ll} \text{minimize}_{\boldsymbol{\alpha}_i} & \ell^n(\boldsymbol{\alpha}_i) \\ \text{subject to} & \alpha_{ji} \geq 0,\ j = 1,\ldots,N, i \neq j, \end{array} \tag{3}$$

where the parameters $\boldsymbol{\alpha}_i := \{\alpha_{ji} \,|\, j = 1,\ldots,N, i \neq j\}$ are the relevant variables, and $\ell^n(\boldsymbol{\alpha}_i) = -\frac{1}{n}\sum_{c\in C^n} g_i(\mathbf{t}^c;\boldsymbol{\alpha}_i)$ corresponds to the terms in Eq. 2 involving $\boldsymbol{\alpha}_i$. The function $g(\,\cdot\,;\boldsymbol{\alpha}_i)$ is simple for several well-known parametric transmission likelihoods, including those described in Table 1. For example, for an exponential transmission likelihood,

$$g_i(\mathbf{t};\boldsymbol{\alpha}_i) = \log\left(\sum_{j:t_j<t_i}\alpha_{ji}\right) - \sum_{j:t_j<t_i}\alpha_{ji}(t_i - t_j)$$

for an infected node and $g_i(\mathbf{t};\boldsymbol{\alpha}_i) = -\sum_{j:t_j<T}\alpha_{ji}(T-t_j)$ for an uninfected node. Refer to Table 2 for a general definition of $g(\,\cdot\,;\boldsymbol{\alpha}_i)$. Moreover, in this subproblem, we only need to consider a super-neighborhood $\mathcal{V}_i = \mathcal{R}_i \cup \mathcal{U}_i$ of i, with cardinality $p_i = |\mathcal{V}_i| \leq N$, where $\mathcal{R}_i$ is the set of upstream nodes from which i is reachable, $\mathcal{U}_i$ is the set of nodes which are reachable from at least one node $j \in \mathcal{R}_i$. Here, we consider a node i to be reachable from a node j if and only if there is a directed path from j to i. We can skip all nodes in $\mathcal{V}\backslash\mathcal{V}_i$ from our analysis because they will never be infected in a cascade before i, and thus, the maximum likelihood estimation of the associated transmission rates will always be zero (and correct).

Below, we show that, as $n \to \infty$, the solution, $\hat{\boldsymbol{\alpha}}_i$, of the problem in Eq. 3 is a consistent estimator of the true parameter $\boldsymbol{\alpha}^*_i$. However, it is not clear whether it is possible to

recover the true network structure with this approach given a finite amount of cascades and, if so, how many cascades are needed. We will show that by adding an ℓ_1-regularizer to the objective function and solving instead the following optimization problem

$$\begin{array}{ll} \text{minimize}_{\boldsymbol{\alpha}_i} & \ell^n(\boldsymbol{\alpha}_i) + \lambda_n ||\boldsymbol{\alpha}_i||_1 \\ \text{subject to} & \alpha_{ji} \geq 0,\, j = 1, \ldots, N, i \neq j, \end{array} \tag{4}$$

we can provide finite sample guarantees for recovering the network structure (and parameters). Our analysis also shows that by selecting an appropriate value for the regularization parameter λ_n, the solution of Eq. 4 successfully recovers the network structure with probability approaching 1 exponentially fast in n.

In the remainder of the paper, we will focus on estimating the parent nodes of a particular node i. For simplicity, we will use $\boldsymbol{\alpha} = \boldsymbol{\alpha}_i$, $\alpha_j = \alpha_{ji}$, $\mathcal{N}^- = \mathcal{N}^-(i)$, $\mathcal{R} = \mathcal{R}_i$, $\mathcal{U} = \mathcal{U}_i$, $d = d_i$, $p_i = p$ and $\alpha^*_{\min} = \alpha^*_{\min,i}$.

4. Consistency

Can we recover the hidden network structures from the observed cascades? The answer is yes. We will show this by proving that the estimator provided by Eq. 3 is consistent, meaning that as the number of cascades goes to **infinity**, we can always recover the true network structure.

More specifically, Gomez-Rodriguez et al. (2011) showed that the network inference problem defined in Eq. 3 is convex in $\boldsymbol{\alpha}$ if the survival functions are log-concave and the hazard functions are concave in $\boldsymbol{\alpha}$. Under these conditions, the Hessian matrix, $\mathcal{Q}^n = \nabla^2 \ell^n(\boldsymbol{\alpha})$, can be expressed as the sum of a nonnegative diagonal matrix $\boldsymbol{D}^n$ and the outer product of a matrix $\boldsymbol{X}^n(\boldsymbol{\alpha})$ with itself, *i.e.*,

$$\mathcal{Q}^n = \boldsymbol{D}^n(\boldsymbol{\alpha}) + \tfrac{1}{n} \boldsymbol{X}^n(\boldsymbol{\alpha}) [\boldsymbol{X}^n(\boldsymbol{\alpha})]^\top. \tag{5}$$

Here the diagonal matrix $\boldsymbol{D}^n(\boldsymbol{\alpha}) = \frac{1}{n} \sum_c \boldsymbol{D}(\mathbf{t}^c; \boldsymbol{\alpha})$ is a sum over a set of diagonal matrices $\boldsymbol{D}(\mathbf{t}^c; \boldsymbol{\alpha})$, one for each cascade c (see Table 2 for the definition of its entries); and $\boldsymbol{X}^n(\boldsymbol{\alpha})$ is the Hazard matrix

$$\boldsymbol{X}^n(\boldsymbol{\alpha}) = \left[\boldsymbol{X}(\mathbf{t}^1; \boldsymbol{\alpha}) \,|\, \boldsymbol{X}(\mathbf{t}^2; \boldsymbol{\alpha}) \,|\, \ldots \,|\, \boldsymbol{X}(\mathbf{t}^n; \boldsymbol{\alpha}) \right], \tag{6}$$

with each column $\boldsymbol{X}(\mathbf{t}^c; \boldsymbol{\alpha}) := h(\boldsymbol{t}^c; \boldsymbol{\alpha})^{-1} \nabla_{\boldsymbol{\alpha}} h(\boldsymbol{t}^c; \boldsymbol{\alpha})$. Intuitively, the Hessian matrix captures the co-occurrence information of nodes in cascades. Both $\boldsymbol{D}(\mathbf{t}^c; \boldsymbol{\alpha})$ and $\boldsymbol{X}^n(\boldsymbol{\alpha})$ are simple for several well-known transmission likelihoods, including those described in Table 1. For example, for an exponential transmission likelihood, $[\boldsymbol{D}(\mathbf{t}^c; \boldsymbol{\alpha})]_{kk} = 0$ and $[\boldsymbol{X}^n(\boldsymbol{\alpha})]_j = \left(\sum_{k:t_k<t_i} \alpha_{ki} \right)^{-1}$ if $t_j < t_i$ and 0 otherwise. Then, we can prove the following consistency result:

Theorem 1 *If the source probability $\mathbb{P}(s)$ is strictly positive for all $s \in \mathcal{R}$, then, the maximum likelihood estimator $\hat{\boldsymbol{\alpha}}$ given by the solution of Eq. 3 is consistent.*

Proof We check the three criteria for consistency: continuity, compactness and identification of the objective function (Newey and McFadden, 1994). Continuity is obvious. For compactness, since $L \to -\infty$ for both $\alpha_{ij} \to 0$ and $\alpha_{ij} \to \infty$ for all i, j so we lose nothing imposing upper and lower bounds thus restricting to a compact subset. For the identification condition, $\boldsymbol{\alpha} \neq \boldsymbol{\alpha}^* \Rightarrow \ell^n(\boldsymbol{\alpha}) \neq \ell^n(\boldsymbol{\alpha}^*)$, we use Lemma 9 and 10 (refer to

Appendices 12.1 and 12.2), which establish that $\boldsymbol{X}^n(\boldsymbol{\alpha})$ has full row rank as $n \to \infty$, and hence $\mathcal{Q}^n$ is positive definite. ■

5. Recovery Conditions

In this section, we will find a set of sufficient conditions on the diffusion model and the cascade sampling process under which we can recover the network structure from **finite samples**. These results allow us to address two questions:

- *Are there some network structures which are more difficult than others to recover?*
- *What kind of cascades are needed for the network structure recovery?*

The answers to these questions are intertwined. The difficulty of finite-sample recovery depends crucially on an irrepresentability condition which is a function of both network structure, parameters of the diffusion model and the cascade sampling process. Intuitively, the sources of the cascades in a diffusion network have to be chosen in such a way that nodes without parent-child relation should co-occur less often compared to nodes with such relation. Many commonly used diffusion models and network structures can be naturally made to satisfy this condition.

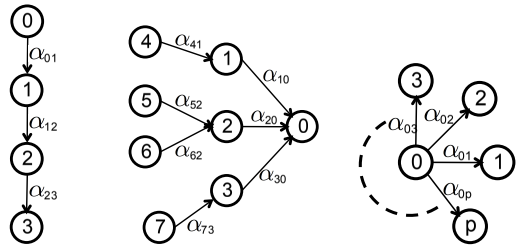
More specifically, we first place two conditions on the Hessian of the population log-likelihood, $\mathbb{E}_c \left[\ell^n(\boldsymbol{\alpha}) \right] = \mathbb{E}_c \left[\log g(\mathbf{t}^c; \boldsymbol{\alpha}) \right]$, where the expectation here is taken over the distribution $\mathbb{P}(s)$ of the source nodes, and the density $f(\mathbf{t}^c | s)$ of the cascades $\mathbf{t}^c$ given a source node s. In this case, we will further denote the Hessian of $\mathbb{E}_c \left[\log g(\mathbf{t}^c; \boldsymbol{\alpha}) \right]$ evaluated at the true model parameter $\boldsymbol{\alpha}^*$ as $\mathcal{Q}^*$. Then, we place two conditions on the Lipschitz continuity of $\boldsymbol{X}(\mathbf{t}^c; \boldsymbol{\alpha})$, and the boundedness of $\boldsymbol{X}(\mathbf{t}^c; \boldsymbol{\alpha}^*)$ and $\nabla g(\mathbf{t}^c; \boldsymbol{\alpha}^*)$ at the true model parameter $\boldsymbol{\alpha}^*$. For simplicity, we will denote the subset of indexes associated to node i's true parents as S, and its complement as S^c. Then, we use $\mathcal{Q}^*_{SS}$ to denote the sub-matrix of $\mathcal{Q}^*$ indexed by S and $\boldsymbol{\alpha}^*_S$ the set of parameters indexed by S. Note that $\boldsymbol{\alpha}^*_{S^c} = 0$.

Condition 1 (Dependency condition): There exists constants $C_{min} > 0$ and $C_{max} > 0$ such that $\Lambda_{min} \left(\mathcal{Q}^*_{SS} \right) \geq C_{min}$ and $\Lambda_{max} \left(\mathcal{Q}^*_{SS} \right) \leq C_{max}$ where $\Lambda_{min}(\cdot)$ and $\Lambda_{max}(\cdot)$ return the leading and the bottom eigenvalue of its argument respectively. This assumption ensures that two connected nodes co-occur reasonably frequently in the cascades but are not deterministically related.

Condition 2 (Irrepresentability condition): There exists a constant $\varepsilon \in (0, 1]$ such that $|||\mathcal{Q}^*_{S^cS} \left(\mathcal{Q}^*_{SS} \right)^{-1}|||_\infty \leq 1 - \varepsilon$, where $|||A|||_\infty = \max_j \sum_k |A_{jk}|$. This assumption captures the intuition that, node i and any of its neighbors should get infected together in a cascade more often than node i and any of its non-neighbors. A similar irrepresentability condition has been proposed on model selection consistency of Lasso (Zhao and Yu, 2006).

Condition 3 (Lipschitz Continuity): For any feasible cascade $\mathbf{t}^c$, the Hazard vector $\boldsymbol{X}(\mathbf{t}^c; \boldsymbol{\alpha})$ is Lipschitz continuous in the domain $\{\boldsymbol{\alpha} : \boldsymbol{\alpha}_S \geq \alpha^*_{\min}/2\}$,

$$\|\boldsymbol{X}(\mathbf{t}^c; \boldsymbol{\beta}) - \boldsymbol{X}(\mathbf{t}^c; \boldsymbol{\alpha})\|_2 \leq k_1 \|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2,$$



(a) Chain (b) Tree (c) Star

Figure 2: Example networks.

where k_1 is some positive constant. As a consequence, the spectral norm of the difference, $n^{-1/2}(\boldsymbol{X}^n(\boldsymbol{\beta}) - \boldsymbol{X}^n(\boldsymbol{\alpha}))$, is also bounded (refer to appendix 12.3), *i.e.*,

$$|||n^{-1/2}\big(\boldsymbol{X}^n(\boldsymbol{\beta}) - \boldsymbol{X}^n(\boldsymbol{\alpha})\big)|||_2 \leq k_1\|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2. \tag{7}$$

Furthermore, for any feasible cascade $\mathbf{t}^c$, $\boldsymbol{D}(\boldsymbol{\alpha})_{jj}$ is Lipschitz continuous for all $j \in \mathcal{V}$,

$$|\boldsymbol{D}(\mathbf{t}^c;\boldsymbol{\beta})_{jj} - \boldsymbol{D}(\mathbf{t}^c;\boldsymbol{\alpha})_{jj}| \leq k_2\|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2,$$

where k_2 is some positive constant.

Condition 4 (Boundedness): For any feasible cascade $\mathbf{t}^c$, the absolute value of each entry in the gradient of its log-likelihood and in the Hazard vector, as evaluated at the true model parameter $\boldsymbol{\alpha}^*$, is bounded,

$$\|\nabla g(\mathbf{t}^c;\boldsymbol{\alpha}^*)\|_\infty \leq k_3, \quad \|\boldsymbol{X}(\mathbf{t}^c;\boldsymbol{\alpha}^*)\|_\infty \leq k_4,$$

where k_3 and k_4 are positive constants. Then the absolute value of each entry in the Hessian matrix $\mathcal{Q}^*$, is also bounded $|||\mathcal{Q}^*|||_\infty \leq k_5$.

Remarks for condition 1 As stated in Theorem 1, as long as the source probability $\mathbb{P}(s)$ is strictly positive for all $s \in \mathcal{R}$, the maximum likelihood formulation is strictly convex and thus there exists $C_{min} > 0$ such that $\Lambda_{min}(\mathcal{Q}^*) \geq C_{min}$. Moreover, condition 4 implies that there exists $C_{max} > 0$ such that $\Lambda_{max}(\mathcal{Q}^*) \leq C_{max}$.

Remarks for condition 2 The irrepresentability condition depends, in a non-trivial way, on the network structure, diffusion parameters, observation window and source node distribution. Here, we give some intuition by studying three small canonical examples.

First, consider the chain graph in Fig. 2(a) and assume that we would like to find the incoming edges to node 3 when $T \to \infty$. Then, it is easy to show that the irrepresentability condition is satisfied if $(P_0+P_1)/(P_0+P_1+P_2) < 1-\varepsilon$ and $P_0/(P_0+P_1+P_2) < 1-\varepsilon$, where P_i denotes the probability of a node i to be the source of a cascade. Thus, for example, if the source of each cascade is chosen uniformly at random, the inequality is satisfied. Here, the irrepresentability condition depends on the source node distribution.

Second, consider the directed tree in Fig. 2(b) and assume that we would like to find the incoming edges to node 0 when $T \to \infty$. Then, it can be shown that the irrepresentability condition is satisfied as long as (1) $P_1 > 0$, (2) ($P_2 > 0$) or ($P_5 > 0$ and $P_6 > 0$), and (3) $P_3 > 0$. As in the chain, the condition depends on the source node distribution.

Finally, consider the star graph in Fig. 2(c), with exponential edge transmission functions, and assume that we would like to find the incoming edges to a leave node i when $T < \infty$. Then, as long as the root node has a nonzero probability $P_0 > 0$ of being the source of a cascade, it can be shown that the irrepresentability condition reduces to the inequalities $\left(1 - \frac{\alpha_{0j}}{\alpha_{0i}+\alpha_{0j}}\right) e^{-(\alpha_{0i}+\alpha_{0j})T} + \frac{\alpha_{0j}}{\alpha_{0i}+\alpha_{0j}} < 1 - \varepsilon(1 + e^{-\alpha_{0i}T})$, $j = 1, \ldots, p : j \neq i$, which always holds for some $\varepsilon > 0$. If $T \to \infty$, then the condition holds whenever $\varepsilon < \alpha_{0i}/(\alpha_{0i} + \max_{j:j\neq i}\alpha_{0j})$. Here, the larger the ratio $\max_{j:j\neq i}\alpha_{0j}/\alpha_{0i}$ is, the smaller the maximum value of ε for which the irrepresentability condition holds. To summarize, as long as $P_0 > 0$, there is always some $\varepsilon > 0$ for which the condition holds, and such ε value depends on the time window and the parameters α_{0j}.

Remarks for conditions 3 and 4 Well-known pairwise transmission likelihoods such as exponential, Rayleigh or Power-law, used in previous work Gomez-Rodriguez et al. (2011), satisfy conditions 3 and 4.

6. Sample Complexity

How many cascades do we need to recover the network structure? We will answer this question by providing a sample complexity analysis of the optimization in Eq. 4. Given the conditions spelled out in Section 5, we can show that the number of cascades needs to grow polynomially in the number of true parents of a node, and depends only logarithmically on the size of the network. This is a positive result, since the network size can be very large (millions or billions), but the number of parents of a node is usually small compared the network size. More specifically, for each individual node, we have the following result:

Theorem 2 *Consider an instance of the continuous-time diffusion model with parameters α^*_{ji} and associated edges $\mathcal{E}^*$ such that the model satisfies condition 1-4, and let C^n be a set of n cascades drawn from the model. Suppose that the regularization parameter λ_n is selected to satisfy*

$$\lambda_n \geq 8k_3\frac{2-\varepsilon}{\varepsilon}\sqrt{\frac{\log p}{n}}. \tag{8}$$

Then, there exist positive constants L and K, independent of (n, p, d), such that if

$$n > Ld^3 \log p, \tag{9}$$

then the following properties hold with probability at least $1 - 2\exp(-K\lambda_n^2 n)$:

1. *For each node $i \in \mathcal{V}$, the ℓ_1-regularized network inference problem defined in Eq. 4 has a unique solution, and so uniquely specifies a set of incoming edges of node i.*
2. *For each node $i \in \mathcal{V}$, the estimated set of incoming edges does not include any false edges and include all true edges.*

Furthermore, suppose that the finite sample Hessian matrix $\mathcal{Q}^n$ satisfies conditions 1 and 2. Then there exist positive constants L and K, independent of (n, p, d), such that the sample complexity can be improved to $n > Ld^2 \log p$ with other statements remain the same.

Remarks. The above sample complexity is proved for each node separately for recovering its parents. Using a union bound, we can provide the sample complexity for recovering the entire network structure by joining these parent-child relations together. The resulting sample complexity and the choice of regularization parameters will remain largely the same,

except that the dependency on d will change from d to d_{max} (the largest number of parents of a node), and the dependency on p will change from $\log p$ to $2\log N$ (N the number of nodes in the network).

6.1 Outline of Analysis

The proof of Theorem 2 uses a technique called primal-dual witness method, previously used in the proof of sparsistency of Lasso (Wainwright, 2009) and high-dimensional Ising model selection (Ravikumar et al., 2010). To the best of our knowledge, the present work is the first that uses this technique in the context of diffusion network inference. First, we show that the optimal solutions to Eq. 4 have shared sparsity pattern, and under a further condition, the solution is unique (proven in Appendix 12.4):

Lemma 3 *Suppose that there exists an optimal primal-dual solution $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ to Eq. 4 with an associated subgradient vector $\hat{\mathbf{z}}$ such that $||\hat{\mathbf{z}}_{S^c}||_\infty < 1$. Then, any optimal primal solution $\tilde{\boldsymbol{\alpha}}$ must have $\tilde{\boldsymbol{\alpha}}_{S^c} = 0$. Moreover, if the Hessian sub-matrix $\mathcal{Q}^n_{SS}$ is strictly positive definite, then $\hat{\boldsymbol{\alpha}}$ is the unique optimal solution.*

Next, we will construct a primal-dual vector $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ along with an associated subgradient vector $\hat{\mathbf{z}}$. Furthermore, we will show that, under the assumptions on (n, p, d) stated in Theorem 2, our constructed solution satisfies the KKT optimality conditions to Eq. 4, and the primal vector has the same sparsity pattern as the true parameter $\boldsymbol{\alpha}^*$, *i.e.*,

$$\hat{\alpha}_j > 0,\ \forall j : \alpha^*_j > 0, \tag{10}$$

$$\hat{\alpha}_j = 0,\ \forall j : \alpha^*_j = 0. \tag{11}$$

Then, based on Lemma 3, we can deduce that the optimal solution to Eq. 4 correctly recovers the sparsisty pattern of $\boldsymbol{\alpha}^*$, and thus the incoming edges to node i.

More specifically, we start by realizing that a primal-dual optimal solution $(\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\mu}})$ to Eq. 4 must satisfy the generalized Karush-Kuhn-Tucker (KKT) conditions Boyd and Vandenberghe (2004):

$$0 \in \nabla \ell^n(\tilde{\boldsymbol{\alpha}}) + \lambda_n \tilde{\mathbf{z}} - \tilde{\boldsymbol{\mu}}, \tag{12}$$

$$\tilde{\mu}_j \tilde{\alpha}_j = 0, \tag{13}$$

$$\tilde{\mu}_j \geq 0, \tag{14}$$

$$\tilde{z}_j = 1,\ \forall \tilde{\alpha}_j > 0, \tag{15}$$

$$|\tilde{z}_j| \leq 1,\ \forall \tilde{\alpha}_j = 0, \tag{16}$$

where $\ell^n(\tilde{\boldsymbol{\alpha}}) = -\frac{1}{n}\sum_{c\in C^n} \log g(\mathbf{t}^c; \tilde{\boldsymbol{\alpha}})$ and $\tilde{\mathbf{z}}$ denotes the subgradient of the ℓ_1-norm.

Suppose the true set of parent of node i is S. We construct the primal-dual vector $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ and the associated subgradient vector $\hat{\mathbf{z}}$ in the following way

1. We set $\hat{\boldsymbol{\alpha}}_S$ as the solution to the partial regularized maximum likelihood problem

$$\hat{\boldsymbol{\alpha}}_S = \underset{(\boldsymbol{\alpha}_S, 0), \boldsymbol{\alpha}_S \geq 0}{\operatorname{argmin}} \{\ell^n(\boldsymbol{\alpha}) + \lambda_n ||\boldsymbol{\alpha}_S||_1\}. \tag{17}$$

   Then, we set $\hat{\boldsymbol{\mu}}_S \geq 0$ as the dual solution associated to the primal solution $\hat{\boldsymbol{\alpha}}_S$.

2. We set $\hat{\boldsymbol{\alpha}}_{S^c} = 0$, so that condition (11) holds, and $\hat{\boldsymbol{\mu}}_{S^c} = \boldsymbol{\mu}^*_{S^c} \geq 0$, where $\boldsymbol{\mu}^*$ is the optimal dual solution to the following problem:

$$\begin{array}{ll} \text{minimize}_{\boldsymbol{\alpha}} & \mathbb{E}_c\left[\ell^n(\boldsymbol{\alpha})\right] \\ \text{subject to} & \alpha_j \geq 0,\ j = 1, \ldots, N, i \neq j. \end{array} \tag{18}$$

   Thus, our construction satisfies condition (14).

3. We obtain $\hat{\mathbf{z}}_{S^c}$ from (12) by substituting in the constructed $\hat{\boldsymbol{\alpha}}$, $\hat{\boldsymbol{\mu}}$ and $\hat{\mathbf{z}}_S$.

Then, we only need to prove that, under the stated scalings of (n, p, d), with high-probability, the remaining KKT conditions (10), (13), (15) and (16) hold.

For simplicity of exposition, we first assume that the dependency and irrepresentability conditions hold for the finite sample Hessian matrix $\mathcal{Q}^n$. Later we will lift this restriction and only place these conditions on the population Hessian matrix $\mathcal{Q}^*$. The following lemma (proven in Appendix 12.5) show that our constructed solution satisfies condition (10):

Lemma 4 *Under condition 3, if the regularization parameter is selected to satisfy*

$$\sqrt{d}\lambda_n \leq \frac{C^2_{\min}}{6(k_2 + 2k_1\sqrt{C_{\max}})},$$

and $\|\nabla_s \ell^n(\boldsymbol{\alpha}^*)\|_\infty \leq \frac{\lambda_n}{4}$, then,

$$\|\hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}^*_S\|_2 \leq \frac{3\sqrt{d}\lambda_n}{C_{\min}} \leq \frac{\alpha^*_{\min}}{2},$$

as long as $\alpha^*_{\min} \geq 6\sqrt{d}\lambda_n / C_{\min}$.

Based on this lemma, we can then further show that the KKT conditions (13) and (15) also hold for the constructed solution. This can be trivially deduced from condition (10) and (11), and our construction steps (a) and (b). Note that it also implies that $\hat{\boldsymbol{\mu}}_S = \boldsymbol{\mu}^*_S = 0$, and hence $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}^*$.

Proving condition (16) is more challenging. We first provide more details on how to construct $\hat{\mathbf{z}}_{S^c}$ mentioned in step (c). We start by using a Taylor expansion of Eq. 12,

$$\mathcal{Q}^n(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*) = -\nabla \ell^n(\boldsymbol{\alpha}^*) - \lambda_n \hat{\boldsymbol{z}} + \hat{\boldsymbol{\mu}} - \mathbf{R}^n, \tag{19}$$

where $\mathbf{R}^n$ is a remainder term with its j-th entry

$$R^n_j = \left[\nabla^2 \ell^n(\bar{\boldsymbol{\alpha}}_j) - \nabla^2 \ell^n(\boldsymbol{\alpha}^*)\right]^T_j (\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*),$$

and $\bar{\boldsymbol{\alpha}}_j = \theta_j \hat{\boldsymbol{\alpha}} + (1-\theta_j)\boldsymbol{\alpha}^*$ with $\theta_j \in [0, 1]$ according to the mean value theorem. Rewriting Eq. 19 using block matrices

$$\begin{pmatrix} \mathcal{Q}^n_{SS} & \mathcal{Q}^n_{SS^c} \\ \mathcal{Q}^n_{S^cS} & \mathcal{Q}^n_{S^cS^c} \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}^*_S \\ \hat{\boldsymbol{\alpha}}_{S_c} - \boldsymbol{\alpha}^*_{S^c} \end{pmatrix} = -\begin{pmatrix} \nabla_S \ell^n(\boldsymbol{\alpha}^*) \\ \nabla_{S^c} \ell^n(\boldsymbol{\alpha}^*) \end{pmatrix} - \lambda_n \begin{pmatrix} \hat{\boldsymbol{z}}_S \\ \hat{\boldsymbol{z}}_{S^c} \end{pmatrix} + \begin{pmatrix} \hat{\mu}_S \\ \hat{\mu}_{S^c} \end{pmatrix} - \begin{pmatrix} \mathbf{R}^n_S \\ \mathbf{R}^n_{S^c} \end{pmatrix} \tag{20}$$

and, after some algebraic manipulation, we have

$$\lambda \hat{\mathbf{z}}_{S^c} = -\nabla_{S^c} \ell^n(\boldsymbol{\alpha}^*) + \hat{\boldsymbol{\mu}}_{S^c} - \mathbf{R}^n_{S^c} - \mathcal{Q}^n_{S^cS}(\mathcal{Q}^n_{SS})^{-1}\big(-\nabla_s \ell^n(\boldsymbol{\alpha}^*) - \lambda \hat{\boldsymbol{z}}_S + \hat{\boldsymbol{\mu}}_S - \mathbf{R}^n_S\big). \tag{21}$$

Next, we upper bound $\|\hat{\boldsymbol{z}}_{S^c}\|_\infty$ using the triangle inequality

$$\begin{aligned} \|\hat{\mathbf{z}}_{S^c}\|_\infty \quad & \leq \lambda_n^{-1}\|\boldsymbol{\mu}^*_{S^c} - \nabla_{S^c}\ell^n(\boldsymbol{\alpha}^*)\|_\infty + \lambda_n^{-1}\|\mathbf{R}^n_{S^c}\|_\infty + \|\mathcal{Q}^n_{S^cS}(\mathcal{Q}^n_{SS})^{-1}\|_\infty \times \left[1 + \lambda_n^{-1}\|\mathbf{R}^n_S\|_\infty\right. \\ & \left. + \lambda_n^{-1}\|\mu^*_S - \nabla_S \ell^n(\boldsymbol{\alpha}^*)\|_\infty\right], \end{aligned}$$

and we want to prove that this upper bound is smaller than 1. This can be done with the help of the following two lemmas (proven in Appendices 12.6 and 12.7):

Lemma 5 *Given $\varepsilon \in (0,1]$ from the irrepresentability condition, we have,*

$$P\left(\frac{2-\varepsilon}{\lambda_n}\|\nabla \ell^n(\boldsymbol{\alpha}^*) - \boldsymbol{\mu}^*\|_\infty \geq 4^{-1}\varepsilon\right) \leq 2p\exp\left(-\frac{n\lambda_n^2\varepsilon^2}{32k_3^2\left(2-\varepsilon\right)^2}\right), \tag{22}$$

which converges to zero at rate $\exp(-c\lambda_n^2 n)$ as long as $\lambda_n \geq 8k_3\frac{2-\varepsilon}{\varepsilon}\sqrt{\frac{\log p}{n}}$.

Lemma 6 *Given $\varepsilon \in (0,1]$ from the irrepresentability condition, if conditions 3 and 4 holds, λ_n is selected to satisfy*

$$\lambda_n d \leq C_{\min}^2 \frac{\varepsilon}{36K(2-\varepsilon)},$$

where $K = k_1 + k_4k_1 + k_1^2 + k_1\sqrt{C_{\max}}$, and $\|\nabla_s \ell^n(\boldsymbol{\alpha}^*)\|_\infty \leq \frac{\lambda_n}{4}$, then, $\frac{\|\mathbf{R}^n\|_\infty}{\lambda_n} \leq \frac{\varepsilon}{4(2-\varepsilon)}$, as long as $\alpha^*_{\min} \geq 6\sqrt{d}\lambda_n/C_{\min}$.

Now, applying both lemmas and the irrepresentability condition on the finite sample Hessian matrix $\mathcal{Q}^n$, we have

$$\begin{aligned}\|\hat{\boldsymbol{z}}_{S^c}\|_\infty &\leq (1-\varepsilon) + \lambda_n^{-1}(2-\varepsilon)\|\mathbf{R}^n\|_\infty + \lambda_n^{-1}(2-\varepsilon)\|\boldsymbol{\mu}^* - \nabla\ell^n(\boldsymbol{\alpha}^*)\|_\infty \\ &\leq (1-\varepsilon) + 0.25\varepsilon + 0.25\varepsilon = 1 - 0.5\varepsilon,\end{aligned}$$

and thus condition (16) holds.

A possible choice of the regularization parameter λ_n and cascade set size n such that the conditions of the Lemmas 4-6 are satisfied is $\lambda_n = 8k_3(2-\varepsilon)\varepsilon^{-1}\sqrt{n^{-1}\log p}$ and $n > 288^2k_3^2(2-\varepsilon)^4C_{min}^{-4}\varepsilon^{-4}d^2\log p + \left(48k_3(2-\varepsilon)C_{min}^{-1}(\alpha^*_{min})^{-1}\varepsilon^{-1}\right)^2 d\log p$.

Last, we lift the dependency and irrepresentability conditions imposed on the finite sample Hessian matrix $\mathcal{Q}^n$. We show that if we only impose these conditions in the corresponding population matrix $\mathcal{Q}^*$, then they will also hold for $\mathcal{Q}^n$ with high probability (proven in Appendices 12.8 and 12.9).

Lemma 7 *If condition 1 holds for $\mathcal{Q}^*$, then, for any $\delta > 0$,*

$$P\left(\Lambda_{min}\left(\mathcal{Q}^n_{SS}\right) \leq C_{min} - \delta\right) \leq 2d^{B_1}\exp\left(-A_1\frac{\delta^2 n}{d^2}\right),$$

$$P\left(\Lambda_{max}\left(\mathcal{Q}^n_{SS}\right) \geq C_{max} + \delta\right) \leq 2d^{B_2}\exp\left(-A_2\frac{\delta^2 n}{d^2}\right),$$

where A_1, A_2, B_1 and B_2 are constants independent of (n,p,d).

Lemma 8 *If $|||\mathcal{Q}^*_{S^cS}\left(\mathcal{Q}^*_{SS}\right)^{-1}|||_\infty \leq 1-\varepsilon$, then,*

$$P\left(\|\mathcal{Q}^n_{S^cS}(\mathcal{Q}^n_{SS})^{-1}\|_\infty \geq 1 - \varepsilon/2\right) \leq p\exp\left(-K\frac{n}{d^3}\right),$$

where K is a constant independent of (n,p,d).

Note in this case the cascade set size need to increase to $n > Ld^3\log p$, where L is a sufficiently large positive constant independent of (n,p,d), for the error probabilities on these last two lemmas to converge to zero.

Algorithm 1 ℓ_1-regularized network inference

Require: C^n, λ_n, K, L
  for all $i \in \mathcal{V}$ **do**
    $k = 0$
    while $k < K$ **do**
      $\boldsymbol{\alpha}_i^{k+1} = \left(\boldsymbol{\alpha}_i^k - L\nabla_{\boldsymbol{\alpha}_i}\ell^n(\boldsymbol{\alpha}_i^k) - \lambda_n L\right)_+$
      $k = k + 1$
    end while
    $\hat{\boldsymbol{\alpha}}_i = \boldsymbol{\alpha}_i^{K-1}$
  end for
  return $\{\hat{\boldsymbol{\alpha}}_i\}_{i\in\mathcal{V}}$

7. Efficient soft-thresholding algorithm

Can we design efficient algorithms to solve Eq. (4) for network recovery? Here, we will design a proximal gradient algorithm which is well suited for solving non-smooth, constrained, large-scale or high-dimensional convex optimization problems Parikh and Boyd (2013). Moreover, they are easy to understand, derive, and implement. We first rewrite Eq. 4 as an unconstrained optimization problem:

$$\text{minimize}_{\boldsymbol{\alpha}} \quad \ell^n(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha}),$$

where the non-smooth convex function $g(\boldsymbol{\alpha}) = \lambda_n||\boldsymbol{\alpha}||_1$ if $\boldsymbol{\alpha} \geq 0$ and $+\infty$ otherwise. By rewriting both problems as a sum of a smooth convex function $\ell^n(\boldsymbol{\alpha})$ and a non-smooth convex function $g(\boldsymbol{\alpha})$, the general recipe from Parikh and Boyd (2013) for designing proximal gradient algorithm can be applied directly.

Algorithm 1 summarizes the resulting algorithm. In each iteration of the algorithm, we need to compute $\nabla\ell^n$ (Table 2) and the proximal operator $\mathbf{prox}_{L^k g}(\mathbf{v})$, where L^k is a step size that we can set to a constant value L or find using a simple line search Beck and Teboulle (2009). Using Moreau's decomposition, we have

$$\mathbf{prox}_{L^k g}(\mathbf{v}) = \mathbf{v} - L^k\,\mathbf{prox}_{g^*/L^k}(\mathbf{v}/L^k), \tag{23}$$

where

$$g^*(\mathbf{y}) = \sup_{\mathbf{x}}\left((\mathbf{y} - \lambda_n\mathbf{1})^T x - \mathbf{1}(\mathbf{x} \geq 0)\right) = \begin{cases}\infty & \text{if } \exists i : y_i > \lambda_n \\ 0 & \text{otherwise}\end{cases}$$

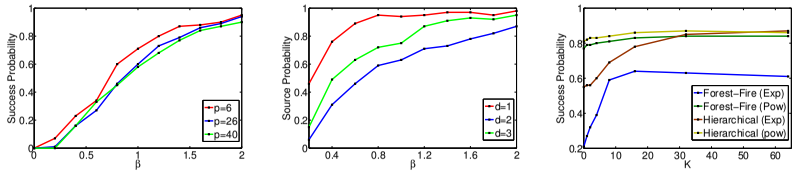
is the conjugate function of g. Then,

$$\mathbf{prox}_{g^*/L^k}(\mathbf{v}/L^k) = \operatorname*{argmin}_{\mathbf{y}}\{g^*(\mathbf{y}) + \frac{L^k}{2}\|y - \mathbf{v}/L^k\|_2^2\} = (\mathbf{v} - \lambda_n L^k)_+$$

In summary, the proximal operator for our particular function $g(\cdot)$ is a soft-thresholding operator, $(\mathbf{v} - \lambda_n L^k)_+$, which leads to a sparse optimal solution $\hat{\boldsymbol{\alpha}}$, as desired.

8. Experiments

In this section, we first illustrate some consequences of Th. 2 by applying our algorithm to several types of networks, parameters (n,p,d), and regularization parameter λ_n. Then, we compare our algorithm to two different state-of-the-art algorithms: NetRate Gomez-Rodriguez et al. (2011) and First-Edge Abrahao et al. (2013).



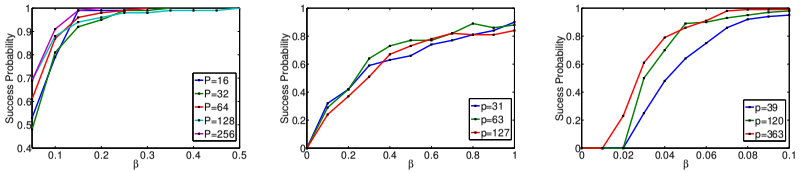
(a) Super-neighborhood p_i (b) In-degrees d_i (c) λ_n

Figure 3: Success probability vs. # of cascades.

Experimental Setup We focus on synthetic networks that mimic the structure of real-world diffusion networks – in particular, social networks. We consider two models of directed real-world social networks: the Forest Fire model (Barabási and Albert, 1999) and the Kronecker Graph model (Leskovec et al., 2010), and use simple pairwise transmission models such as exponential, power-law or Rayleigh. We use networks with 128 nodes and, for each edge, we draw its associated transmission rate from a uniform distribution $U(0.5, 1.5)$. In general, we proceed as follows: we generate a network $\mathcal{G}^*$ and transmission rates $\mathbf{A}^*$, simulate a set of cascades and, for each cascade, record the node infection times. Then, given the infection times, we infer a network $\hat{\mathcal{G}}$. Finally, when we illustrate the consequences of Th. 2, we evaluate the accuracy of the inferred neighborhood of a node $\hat{\mathcal{N}}^-(i)$ using probability of success $P(\hat{\mathcal{E}} = \mathcal{E}^*)$, estimated by running our method of 100 independent cascade sets. When we compare our algorithm to NetRate and First-Edge, we use the F_1 score, which is defined as $2PR/(P+R)$, where precision (P) is the fraction of edges in the inferred network $\hat{\mathcal{G}}$ present in the true network $\mathcal{G}^*$, and recall (R) is the fraction of edges of the true network $\mathcal{G}^*$ present in the inferred network $\hat{\mathcal{G}}$.

Parameters (n, p, d) According to Th. 2, the number of cascades that are necessary to successfully infer the incoming edges of a node will increase polynomially to the node's neighborhood size d_i and logarithmically to the super-neighborhood size p_i. Here, we first infer the incoming links of nodes on the same type of canonical networks as depicted in Fig. 2. We choose nodes the same in-degree but different super-neighboorhod set sizes p_i and experiment with different scalings β of the number of cascades $n = 10\beta d \log p$. We set the regularization parameter λ_n as a constant factor of $\sqrt{\log(p)/n}$ as suggested by Theorem 2 and, for each node, we used cascades which contained at least one node in the super-neighborhood of the node under study. We used an exponential transmission model and time window $T = 10$. As predicted by Theorem 2, very different p values lead to curves that line up with each other quite well.

Next, we infer the incoming links of nodes of a larger hierarchical Kronecker network. Again, we choose nodes with the same in-degree ($d_i = 3$) but different super-neighboorhod set sizes p_i under different scalings β of the number of cascades $n = 10\beta d \log p$. We used an exponential transmission model and $T = 5$. Fig. 3(a) summarizes the results, where, for each node, we used cascades which contained at least one node in the super-neighborhood of the node under study. Similarly as in the case of the canonical networks, very different p values lead to curves that line up with each other quite well.



(a) Chain ($d_i = 1$) (b) Stars ($d_i = 1$) (c) Tree ($d_i = 3$)

Figure 4: Success probability vs. # of cascades. Different super-neighborhood sizes p_i.

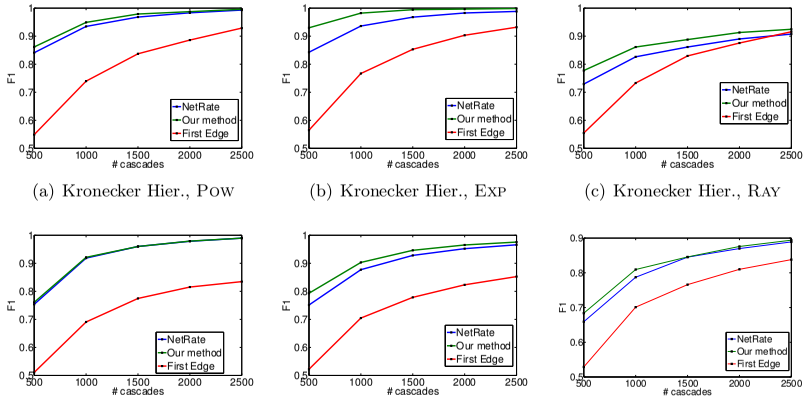
Finally, we infer the incoming links of nodes of a hierarchical Kronecker network with equal super neighborhood size ($p_i = 70$) but different in-degree (d_i) under different scalings β of the number of cascades $n = 10\beta d \log p$ and choose the regularization parameter λ_n as a constant factor of $\sqrt{\log(p)/n}$ as suggested by Theorem 2. We used an exponential transmission model and time window $T = 5$. Figure 3(b) summarizes the results, where we observe that, as predicted by Theorem 2, different d values lead to noticeably different curves.

Regularization parameter λ_n Our main result indicates that the regularization parameter λ_n should be a constant factor of $\sqrt{\log(p)/n}$. Fig. 3(c) shows the success probability of our algorithm against different scalings K of the regularization parameter $\lambda_n = K\sqrt{\log(p)/n}$ for different types of networks using 150 cascades and $T = 5$. We find that for sufficiently large λ_n, the success probability flattens, as expected from Th. 2. It flattens at values smaller than one because we used a fixed number of cascades n, which may not satisfy the conditions of Th. 2.

Comparison with NetRate and First-Edge Fig. 5 compares the accuracy of our algorithm, NetRate and First-Edge against number of cascades for three hierarchical Kronecker network and three Forest Fire networks, with power-law (Pow), exponential (Exp) and rayleigh (Ray) transmission models, and an observation window $T = 10$. Our method outperforms both competitive methods, finding especially striking the competitive advantage with respect to First-Edge, however, this may be explained by comparing the sample complexity results for both methods: First-Edge needs $O(Nd \log N)$ cascades to achieve a probability of success approaching 1 in a rate polynomial in the number of cascades while our method needs $O(d^3 \log N)$ to achieve a probability of success approaching 1 in a rate exponential in the number of cascades.

9. Discussion

Our results can be extended in multiple directions. First, our novel formulation of the diffusion network recovery problem as a ℓ_1-regularized convex optimization problem establishes a connection between the literature on information diffusion and a vast literature on high dimension sparse recovery problem from machine learning and statistics literature. This connection allows us to borrow analysis frameworks for graphical model structure estimation to analyze information diffusion. In terms of diffusion models, we can extend the current independent cascade model to deal with nonparametric transmission functions (Du



(a) Kronecker Hier., Pow (b) Kronecker Hier., Exp (c) Kronecker Hier., Ray

(d) Forest Fire, Pow (e) Forest Fire, Exp (f) Forest Fire, Ray

Figure 5: F_1-score vs. # of cascades.

et al., 2012a), transmission function conditioned on additional features (Du et al., 2013), diffusion models which allows for multiple events (Zhou et al., 2013). All three models will result in convex loss function; and for the former two models, we can employ grouped lasso regularization (Yuan and Lin, 2006), while for the latter model, we can employ a nuclear norm regularization (Recht et al., 2010). These models are more complicated than the independent cascade model we studied in the paper, but analysis for these models can be carried out using a general M-estimation analysis framework (Negahban et al., 2009), since these regularizers are decomposable and one needs to check the restricted strong convexity of the loss function. In terms of the estimation algorithms, we can employ proximal algorithms (Parikh and Boyd, 2013) or the conditional gradient algorithm (Jaggi, 2013) to deal with different type of diffusion models and regularizers. When the data is large, one can consider distributed (Boyd et al., 2011) and online estimation (Nemirovski et al., 2009) procedures.

Our results also bring out interesting further open problems on diffusion network estimations. For instance, the success of the network inference algorithm in Equation (2) relies on the fulfillment of the above mentioned irrepresentability condition on the Hessian, $\mathcal{Q}^*$, of the population log-likelihood $\mathbb{E}[\ell^n]$, where the expectation here is taken over the distribution $\mathbb{P}(s)$ of the source nodes and the random generative process of the diffusion model given a source node s. This condition captures the intuition that, node i and any of its neighbors should get infected together in a cascade more often than node i and any of its non-neighbors. Unfortunately, the irrepresentability condition depends, in a non-trivial way, on the network structure, diffusion parameters, and the source distribution $\mathbb{P}(s)$, which are all *unknown* during the network inference stage. Previous work has typically assumed the network structure, diffusion parameters, observation window and source distribution to be fixed, and source locations are sampled *passively* from the latter. However, in practice,

the source locations to sample from may be determined *actively* in a sequential manner, potentially based on the information gathered from previous source locations. Thus an interesting open question is:

> Suppose there exists an unknown $\mathbb{P}(s)$ where the irrepresentability conditions hold for the diffusion model. Under what conditions, can we design an "active" algorithm which samples the source location intelligently and achieves the sample complexity in Theorem 2, or even better sample complexity, *e.g.*, $o(d_i^3 \log N)$?

10. Conclusions

Our work contributes towards establishing a theoretical foundation of the network inference problem. Specifically, we proposed a ℓ_1-regularized maximum likelihood inference method for a well-known continuous-time diffusion model and an efficient proximal gradient implementation, and then show that, for general networks satisfying a natural irrepresentability condition, our method achieves an exponentially decreasing error with respect to the number of cascades as long as $O(d^3 \log N)$ cascades are recorded.

Our work also opens many interesting venues for future work. For example, given a fixed number of cascades, it would be useful to provide confidence intervals on the inferred edges. Further, a detailed theoretical analysis of the irrepresentability condition on large synthetic networks that mimic the structure of real-world diffusion networks, such as Kronecker or Forest-Fire networks, is still missing. Given a network with arbitrary pairwise likelihoods, it is an open question whether there always exists at least one source distribution and time window value such that the irrepresentability condition is satisfied, and, and if so, whether there is an efficient way of finding this distribution. Finally, our work assumes all activations occur due to network diffusion and are recorded. It would be interesting to allow for missing observations, as well as activations due to exogenous factors.

11. Acknowledgement

This research was supported in part by NSF/NIH BIGDATA 1R01GM108341-01, NSF IIS1116886, NSF CAREER IIS-1350983 and a Raytheon faculty fellowship to L. Song.

12. Appendix

12.1 Proof of Lemma 9

Lemma 9 *Given log-concave survival functions and concave hazard functions in the parameter(s) of the pairwise transmission likelihoods, then, a sufficient condition for the Hessian matrix $\mathcal{Q}^n$ to be positive definite is that the hazard matrix $X^n(\boldsymbol{\alpha})$ is non-singular.*

Proof Using Eq. 5, the Hessian matrix can be expressed as a sum of two matrices, $\mathbf{D}^n(\boldsymbol{\alpha})$ and $\mathbf{X}^n(\boldsymbol{\alpha})\mathbf{X}^n(\boldsymbol{\alpha})^\top$. The matrix $\mathbf{D}^n(\boldsymbol{\alpha})$ is trivially positive semidefinite by log-concavity of the survival functions and concavity of the hazard functions. The matrix $\mathbf{X}^n(\boldsymbol{\alpha})\mathbf{X}^n(\boldsymbol{\alpha})^\top$ is positive definite matrix since $\mathbf{X}^n(\boldsymbol{\alpha})$ is full rank by assumption. Then, the Hessian matrix is positive definite since it is a sum a positive semidefinite matrix and a positive definite matrix. ■

12.2 Proof of Lemma 10

Lemma 10 *If the source probability $\mathbb{P}(s)$ is strictly positive for all $s \in \mathcal{R}$, then, for an arbitrarily large number of cascades $n \to \infty$, there exists an ordering of the nodes and cascades within the cascade set such that the hazard matrix $\boldsymbol{X}^n(\boldsymbol{\alpha})$ is non-singular.*

Proof In this proof, we find a labeling of the nodes (row indices in $\mathbf{X}^n(\boldsymbol{\alpha})$) and ordering of the cascades (column indices in $\mathbf{X}^n(\boldsymbol{\alpha})$), such that, for an arbitrary large number of cascades, we can express the matrix $\mathbf{X}^n(\boldsymbol{\alpha})$ as $[T\ B]$, where $T \in \mathbb{R}^{p\times p}$ is an upper triangular with nonzero diagonal elements and $B \in \mathbb{R}^{p\times n-p}$. And, therefore, $\mathbf{X}^n(\boldsymbol{\alpha})$ has full rank (rank p). We proceed first by sorting nodes in $\mathcal{R}$ and then continue by sorting nodes in $\mathcal{U}$:

- **Nodes in $\mathcal{R}$**: For each node $u \in \mathcal{R}$, consider the set of cascades C_u in which u was a source and i got infected. Then, rank each node u according to the earliest position in which node i got infected across all cascades in C_u in decreasing order, breaking ties at random. For example, if a node u was, at least once, the source of a cascade in which node i got infected just after the source, but in contrast, node v was never the source of a cascade in which node i got infected the second, then node u will have a lower index than node v. Then, assign row k in the matrix $\mathbf{X}^n(\boldsymbol{\alpha})$ to node in position k and assign the first d columns to the corresponding cascades in which node i got infected earlier. In such ordering, $\mathbf{X}^n(\boldsymbol{\alpha})_{mk} = 0$ for all $m < k$ and $\mathbf{X}^n(\boldsymbol{\alpha})_{kk} \neq 0$.

- **Nodes in $\mathcal{U}$**: Similarly as in the first step, and assign them the rows $d+1$ to p. Moreover, we assign the columns $d+1$ to p to the corresponding cascades in which node i got infected earlier. Again, this ordering satisfies that $\mathbf{X}^n(\boldsymbol{\alpha})_{mk} = 0$ for all $m < k$ and $\mathbf{X}^n(\boldsymbol{\alpha})_{kk} \neq 0$. Finally, the remaining columns $n-p$ can be assigned to the remaining cascades at random.

This ordering leads to the desired structure $[T\ B]$, and thus it is non-singular. ■

12.3 Proof of Eq 7.

If the Hazard vector $\boldsymbol{X}(\mathbf{t}^c;\boldsymbol{\alpha})$ is Lipschitz continuous in the domain $\{\boldsymbol{\alpha} : \boldsymbol{\alpha}_S \geq \frac{\alpha^*_{\min}}{2}\}$,

$$\|\boldsymbol{X}(\mathbf{t}^c;\boldsymbol{\beta}) - \boldsymbol{X}(\mathbf{t}^c;\boldsymbol{\alpha})\|_2 \leq k_1\|\boldsymbol{\beta}-\boldsymbol{\alpha}\|_2,$$

where k_1 is some positive constant. Then, we can bound the spectral norm of the difference, $\frac{1}{\sqrt{n}}(\boldsymbol{X}^n(\boldsymbol{\beta}) - \boldsymbol{X}^n(\boldsymbol{\alpha}))$, in the domain $\{\boldsymbol{\alpha} : \boldsymbol{\alpha}_S \geq \frac{\alpha^*_{\min}}{2}\}$ as follows:

$$|||\frac{1}{\sqrt{n}}\big(\mathbf{X}^n(\boldsymbol{\beta}) - \mathbf{X}^n(\boldsymbol{\alpha})\big)|||_2 = \max_{\|\boldsymbol{u}\|_2=1}\frac{1}{\sqrt{n}}\|\boldsymbol{u}\big(\mathbf{X}^n(\boldsymbol{\beta}) - \mathbf{X}^n(\boldsymbol{\alpha})\big)\|_2$$

$$= \max_{\|\boldsymbol{u}\|_2=1}\frac{1}{\sqrt{n}}\sqrt{\sum_{c=1}^{n}\langle \boldsymbol{u}, \mathbf{X}(\mathbf{t}^c;\boldsymbol{\beta}) - \mathbf{X}(\mathbf{t}^c;\boldsymbol{\alpha})\rangle^2} \leq \frac{1}{\sqrt{n}}\sqrt{k_1^2 n\|\boldsymbol{u}\|_2^2\|\boldsymbol{\beta}-\boldsymbol{\alpha}\|_2^2} \leq k_1\|\boldsymbol{\beta}-\boldsymbol{\alpha}\|_2.$$

12.4 Proof of Lemma 3

By Lagrangian duality, the regularized network inference problem defined in Eq. 4 is equivalent to the following constrained optimization problem:

$$\begin{array}{ll}\text{minimize}_{\boldsymbol{\alpha}_i} & \ell^n(\boldsymbol{\alpha}_i) \\ \text{subject to} & \alpha_{ji} \geq 0,\ j = 1,\dots,N, i \neq j, \\ & ||\boldsymbol{\alpha}_i||_1 \leq C(\lambda_n)\end{array} \tag{24}$$

where $C(\lambda_n) < \infty$ is a positive constant. In this alternative formulation, λ_n is the Lagrange multiplier for the second constraint. Since λ_n is strictly positive, the constraint is active at any optimal solution, and thus $||\boldsymbol{\alpha}_i||_1$ is constant across all optimal solutions.

Using that $\ell^n(\boldsymbol{\alpha}_i)$ is a differentiable convex function by assumption and $\{\boldsymbol{\alpha} : \alpha_{ji} \geq 0, ||\boldsymbol{\alpha}_i||_1 \leq C(\lambda_n)\}$ is a convex set, we have that $\nabla\ell^n(\boldsymbol{\alpha}_i)$ is constant across optimal primal solutions Mangasarian (1988). Moreover, any optimal primal-dual solution in the original problem must satisfy the KKT conditions in the alternative formulation defined by Eq. 24, in particular,

$$\nabla\ell^n(\boldsymbol{\alpha}_i) = -\lambda_n\mathbf{z} + \boldsymbol{\mu},$$

where $\boldsymbol{\mu} \geq 0$ are the Lagrange multipliers associated to the non negativity constraints and $\mathbf{z}$ denotes the subgradient of the ℓ1-norm.

Consider the solution $\hat{\boldsymbol{\alpha}}$ such that $||\hat{\mathbf{z}}_{\mathbf{S}^c}||_\infty < 1$ and thus $\nabla_{\alpha_{S^c}}\ell^n(\hat{\boldsymbol{\alpha}}_i) = -\lambda_n\hat{\mathbf{z}}_{\mathbf{S}^c} + \hat{\boldsymbol{\mu}}_{S^c}$. Now, assume there is an optimal primal solution $\tilde{\boldsymbol{\alpha}}$ such that $\tilde{\alpha}_{ji} > 0$ for some $j \in S^c$, then, using that the gradient must be constant across optimal solutions, it should hold that $-\lambda_n\hat{z}_j + \hat{\mu}_j = -\lambda_n$, where $\tilde{\mu}_{ji} = 0$ by complementary slackness, which implies $\hat{\mu}_j = -\lambda_n(1-\hat{z}_j) < 0$. Since $\hat{\mu}_j \geq 0$ by assumption, this leads to a contradiction. Then, any primal solution $\tilde{\boldsymbol{\alpha}}$ must satisfy $\tilde{\boldsymbol{\alpha}}_{S^c} = 0$ for the gradient to be constant across optimal solutions.

Finally, since $\boldsymbol{\alpha}_{S^c} = 0$ for all optimal solutions, we can consider the restricted optimization problem defined in Eq. 17. If the Hessian sub-matrix $[\nabla^2 L(\hat{\boldsymbol{\alpha}})]_{SS}$ is strictly positive definite, then this restricted optimization problem is strictly convex and the optimal solution must be unique.

12.5 Proof of Lemma 4

To prove this lemma, we will first construct a function

$$G(\mathbf{u}_S) := \ell^n(\boldsymbol{\alpha}^*_S + \mathbf{u}_S) - \ell^n(\boldsymbol{\alpha}^*_S) + \lambda_n(\|\boldsymbol{\alpha}^*_S + \mathbf{u}_S\|_1 - \|\boldsymbol{\alpha}^*_S\|_1).$$

whose domain is restricted to the convex set $\mathcal{U} = \{\mathbf{u}_S : \boldsymbol{\alpha}^*_S + \mathbf{u}_S \geq \mathbf{0}\}$. By construction, $G(\mathbf{u}_S)$ has the following properties

1. It is convex with respect to $\mathbf{u}_S$.
2. Its minimum is obtained at $\hat{\boldsymbol{u}}_S := \hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}^*_S$. That is $G(\hat{\boldsymbol{u}}_S) \leq G(\mathbf{u}_S)$, $\forall \mathbf{u}_S \neq \hat{\boldsymbol{u}}_S$.
3. $G(\hat{\boldsymbol{u}}_S) \leq G(\mathbf{0}) = 0$.

Based on the properties 1 and 3 above, we deduce that any point in the segment, $\mathbb{L} := \{\tilde{\boldsymbol{u}}_S : \tilde{\boldsymbol{u}}_S = t\hat{\boldsymbol{u}}_S + (1-t)\mathbf{0}, t \in [0,1]\}$, connecting $\hat{\boldsymbol{u}}_S$ and $\mathbf{0}$ has $G(\tilde{\boldsymbol{u}}_S) \leq 0$. That is

$$G(\tilde{\boldsymbol{u}}_S) = G(t\hat{\boldsymbol{u}}_S + (1-t)\mathbf{0}) \leq tG(\hat{\boldsymbol{u}}_S) + (1-t)G(\mathbf{0}) \leq 0.$$

Next, we will find a sphere centered at $\mathbf{0}$ with strictly positive radius B, $\mathbb{S}(B) := \{\boldsymbol{u}_S : \|\boldsymbol{u}_S\|_2 = B\}$, such that function $G(\boldsymbol{u}_S) > 0$ (strictly positive) on $\mathbb{S}(B)$. We note that this sphere $\mathbb{S}(B)$ can not intersect with the segment $\mathbb{L}$ since the two sets have strictly different function values. Furthermore, the only possible configuration is that the segment is contained inside the sphere entirely, leading us to conclude that the end point $\hat{\boldsymbol{u}}_S := \hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}^*_S$ is also within the sphere. That is $\|\hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}^*_S\|_2 \leq B$.

In the following, we will provide details on finding such a suitable B which will be a function of the regularization parameter λ_n and the neighborhood size d. More specifically, we will start by applying a Taylor series expansion and the mean value theorem,

$$G(\mathbf{u}_S) = \nabla_S \ell^n(\boldsymbol{\alpha}^*_S)^\top \mathbf{u}_S + \mathbf{u}_S^\top \nabla^2_{SS} \ell^n(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S)\mathbf{u}_S + \lambda_n(\|\boldsymbol{\alpha}^*_S + \mathbf{u}_S\|_1 - \|\boldsymbol{\alpha}^*_S\|_1), \quad (25)$$

where $b \in [0,1]$. We will show that $G(\mathbf{u}_S) > 0$ by bounding below each term of above equation separately.

We bound the absolute value of the first term using the assumption on the gradient, $\nabla_S \ell(\cdot)$,

$$|\nabla_S \ell^n(\boldsymbol{\alpha}^*_S)^\top \mathbf{u}_S| \leq \|\nabla_S \ell\|_\infty \|\mathbf{u}_S\|_1 \leq \|\nabla_S \ell\|_\infty \sqrt{d}\|\mathbf{u}_S\|_2 \leq 4^{-1}\lambda_n B\sqrt{d}. \quad (26)$$

We bound the absolute value of the last term using the reverse triangle inequality.

$$\lambda_n |\|\boldsymbol{\alpha}^*_S + \mathbf{u}_S\|_1 - \|\boldsymbol{\alpha}^*_S\|_1| \leq \lambda_n \|\mathbf{u}_S\|_1 \leq \lambda_n \sqrt{d}\|\mathbf{u}_S\|_2. \quad (27)$$

Bounding the remaining middle term is more challenging. We start by rewriting the Hessian as a sum of two matrices, using Eq. 5,

$$\begin{aligned} q &= \min_{\mathbf{u}_S} \mathbf{u}_S^\top \mathbf{D}^n_{SS}(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S)\mathbf{u}_S + n^{-1}\mathbf{u}_S^\top \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S)\mathbf{X}^n_S(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S)^\top \mathbf{u}_S \\ &= \min_{\mathbf{u}_S} \mathbf{u}_S^\top \mathbf{D}^n_{SS}(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S)\mathbf{u}_S + \|\mathbf{u}_S^\top \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S)\|_2^2. \end{aligned}$$

Now, we introduce two additional quantities,

$$\Delta \mathbf{D}^n_{SS} = \mathbf{D}^n_{SS}(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S) - \mathbf{D}^n_{SS}(\boldsymbol{\alpha}^*_S) \quad \text{and} \quad \Delta \mathbf{X}^n_S = \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S) - \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S),$$

and rewrite q as

$$\begin{aligned} q = \min_{\mathbf{u}_S} \big[&\mathbf{u}_S^\top \mathbf{D}^n_{SS}(\boldsymbol{\alpha}^*_S)\mathbf{u}_S + n^{-1}\|\mathbf{u}_S^\top \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S)\|_2^2 + n^{-1}\|\mathbf{u}_S^\top \Delta \mathbf{X}^n_S\|_2^2 + \mathbf{u}_S^\top \Delta \mathbf{D}^n_{SS}\mathbf{u}_S \\ &+ 2n^{-1}\langle \mathbf{u}_S^\top \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S), \mathbf{u}_S^\top \Delta \mathbf{X}^n_S \rangle \big]. \end{aligned}$$

Next, we use dependency condition,

$$q \geq C_{\min} B^2 - \max_{\mathbf{u}_S} |\underbrace{\mathbf{u}_S^\top \Delta \mathbf{D}^n_{SS}\mathbf{u}_S}_{T_1}| - \max_{\mathbf{u}_S} 2|\underbrace{n^{-1}\langle \mathbf{u}_S^\top \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S), \mathbf{u}_S^\top \Delta \mathbf{X}^n_S \rangle}_{T_2}|,$$

and proceed to bound T_1 and T_2 separately. First, we bound T_1 using the Lipschitz condition,

$$|T_1| = |\sum_{k \in S} u_k^2 [\mathbf{D}^n_k(\boldsymbol{\alpha}^*_S + b\mathbf{u}_S) - \mathbf{D}^n_k(\boldsymbol{\alpha}^*_S)]| \leq \sum_{k \in S} u_k^2 k_2 \|b\mathbf{u}_S\|_2 \leq k_2 B^3.$$

Then, we use the dependency condition, the Lipschitz condition and the Cauchy-Schwartz inequality to bound T_2,

$$\begin{aligned} T_2 &\leq \frac{1}{\sqrt{n}}\|\mathbf{u}_S^\top \mathbf{X}^n_S(\boldsymbol{\alpha}^*_S)\|_2 \frac{1}{\sqrt{n}}\|\mathbf{u}_S^\top \Delta \mathbf{X}^n_S\|_2 \leq \sqrt{C_{\max}} B \frac{1}{\sqrt{n}}\|\mathbf{u}_S^\top \Delta \mathbf{X}^n_S\|_2 \\ &\leq \sqrt{C_{\max}} B \|\mathbf{u}_S\|_2 \frac{1}{\sqrt{n}}|\|\Delta \mathbf{X}^n_S\||_2 \leq \sqrt{C_{\max}} B^2 k_1 \|b\mathbf{u}_S\|_2 \\ &\leq k_1 \sqrt{C_{\max}} B^3, \end{aligned}$$

where we note that applying the Lipschitz condition implies assuming $B < \frac{\alpha_{min}}{2}$. Next, we incorporate the bounds of T_1 and T_2 to lower bound q,

$$q \geq C_{\min} B^2 - (k_2 + 2k_1\sqrt{C_{\max}})B^3. \quad (28)$$

Now, we set $B = K\lambda_n\sqrt{d}$, where K is a constant that we will set later in the proof, and select the regularization parameter λ_n to satisfy

$$\lambda_n \sqrt{d} \leq \frac{C_{\min}}{2K(k_2 + 2k_1\sqrt{C_{\max}})}. \quad (29)$$

Then,

$$\begin{aligned} G(\mathbf{u}_S) &\geq -4^{-1}\lambda_n\sqrt{d}B + 0.5C_{\min}B^2 - \lambda_n\sqrt{d}B \geq B(0.5C_{\min}B - 1.25\lambda_n\sqrt{d}) \\ &\geq B(0.5C_{\min}K\lambda_n\sqrt{d} - 1.25\lambda_n\sqrt{d}). \end{aligned}$$

In the last step, we set the constant $K = 3C^{-1}_{\min}$, and we have

$$G(\mathbf{u}_S) \geq 0.25\lambda_n\sqrt{d} > 0,$$

as long as

$$\begin{aligned} \sqrt{d}\lambda_n &\leq \frac{C^2_{\min}}{6(k_2 + 2k_1\sqrt{C_{\max}})} \\ \alpha^*_{\min} &\geq \frac{6\lambda_n\sqrt{d}}{C_{\min}}. \end{aligned}$$

Finally, convexity of $G(\mathbf{u}_S)$ yields

$$\|\hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}^*_S\|_2 \leq 3\lambda_n\sqrt{d}/C_{\min} \leq \frac{\alpha^*_{\min}}{2}.$$

12.6 Proof of Lemma 5

Define $z^c_j = [\nabla g(\mathbf{t}^c; \boldsymbol{\alpha}^*)]_j$ and $z_j = \frac{1}{n}\sum_c z^c_j$. Now, using the KKT conditions and condition 4 (Boundedness), we have that $\mu^*_j = \mathbb{E}_c\{z^c_j\}$ and $|z^c_j| \leq k_3$, respectively. Thus, Hoeffding's inequality yields

$$P\left(|z_j - \mu^*_j| > \frac{\lambda_n \varepsilon}{4(2-\varepsilon)}\right) \leq 2\exp\left(-\frac{n\lambda_n^2\varepsilon^2}{32k_3^2\,(2-\varepsilon)^2}\right),$$

and then,

$$P\left(\|\boldsymbol{z} - \boldsymbol{\mu}^*\|_\infty > \frac{\lambda_n \varepsilon}{4(2-\varepsilon)}\right) \leq 2\exp\left(-\frac{n\lambda_n^2\varepsilon^2}{32k_3^2\,(2-\varepsilon))^2} + \log p\right).$$

12.7 Proof of Lemma 6

We start by factorizing the Hessian matrix, using Eq. 5,

$$R_j^n = \left[\nabla^2 \ell^n(\bar{\boldsymbol{\alpha}}_j) - \nabla^2 \ell^n(\boldsymbol{\alpha}^*)\right]_j^\top (\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*) = \omega_j^n + \delta_j^n,$$

where,

$$\begin{aligned}
\omega_j^n &= \left[\mathbf{D}^n(\bar{\boldsymbol{\alpha}}_j) - \mathbf{D}^n(\boldsymbol{\alpha}^*)\right]_j^\top (\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*) \\
\delta_j^n &= \frac{1}{n} \boldsymbol{V}_j^n (\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*) \\
\boldsymbol{V}_j^n &= [\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)]_j \mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)^\top - [\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j \mathbf{X}^n(\boldsymbol{\alpha}^*)^\top.
\end{aligned}$$

Next, we proceed to bound each term separately. Since $[\bar{\boldsymbol{\alpha}}_j]_S = \theta_j \hat{\boldsymbol{\alpha}}_S + (1-\theta_j)\boldsymbol{\alpha}_S^*$ where $\theta_j \in [0,1]$, and $\|\hat{\boldsymbol{\alpha}}_S - \boldsymbol{\alpha}_S^*\|_\infty \leq \frac{\alpha_{\min}^*}{2}$ (Lemma 4), it holds that $[\bar{\boldsymbol{\alpha}}_j]_S \geq \frac{\alpha_{\min}^*}{2}$. Then, we can use condition 3 (Lipschitz Continuity) to bound ω_j^n.

$$|\omega_j^n| \leq k_1 \|\bar{\boldsymbol{\alpha}}_j - \boldsymbol{\alpha}^*\|_2 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2 \leq k_1 \theta_j \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2^2 \leq k_1 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2^2.$$

However, bounding term δ_j^n is more difficult. Let us start by rewriting δ_j^n as follows.

$$\delta_j^n = (\boldsymbol{\Lambda}_1 + \boldsymbol{\Lambda}_2 + \boldsymbol{\Lambda}_3)(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*),$$

where,

$$\begin{aligned}
\boldsymbol{\Lambda}_1 &= [\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j (\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)^\top - \mathbf{X}^n(\boldsymbol{\alpha}^*)^\top) \\
\boldsymbol{\Lambda}_2 &= \{[\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)]_j - [\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j\} (\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)^\top - \mathbf{X}^n(\boldsymbol{\alpha}^*)^\top) \\
\boldsymbol{\Lambda}_3 &= \left([\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)]_j - [\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j\right) \mathbf{X}^n(\boldsymbol{\alpha}^*)^\top.
\end{aligned}$$

Next, we bound each term separately. For the first term, we first apply Cauchy inequality,

$$|\boldsymbol{\Lambda}_1(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq \|[\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j\|_2 \times \|\|\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)^\top - \mathbf{X}^n(\boldsymbol{\alpha}^*)^\top\|\|_2 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2,$$

and then use condition 3 (Lipschtiz Continuity) and 4 (Boundedness),

$$|\boldsymbol{\Lambda}_1(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq n k_4 k_1 \|\bar{\boldsymbol{\alpha}}_j - \boldsymbol{\alpha}^*\|_2 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2 \leq n k_4 k_1 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2^2.$$

For the second term, we also start by applying Cauchy inequality,

$$|\boldsymbol{\Lambda}_2(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq \|[\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)]_j - [\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j\|_2 \times \|\|\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)^\top - \mathbf{X}^n(\boldsymbol{\alpha}^*)^\top\|\|_2 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2,$$

and then use condition 3 (Lipschtiz Continuity),

$$|\boldsymbol{\Lambda}_2(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq n k_1^2 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2^2.$$

Last, for third term, once more we start by applying Cauchy inequality,

$$|\boldsymbol{\Lambda}_3(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq \|[\mathbf{X}^n(\bar{\boldsymbol{\alpha}}_j)]_j - [\mathbf{X}^n(\boldsymbol{\alpha}^*)]_j\|_2 \times \|\|\mathbf{X}^n(\boldsymbol{\alpha}^*)^\top\|\|_2 \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2,$$

and then apply condition 1 (Dependency Condition) and condition 3 (Lipschitz Continuity),

$$|\boldsymbol{\Lambda}_3(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq n k_1 \sqrt{C_{\max}} \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2^2$$

Now, we combine the bounds,

$$\|\boldsymbol{R}^n\|_\infty \leq K \|\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_2^2,$$

where

$$K = k_1 + k_4 k_1 + k_1^2 + k_1 \sqrt{C_{\max}}.$$

Finally, using Lemma 4 and selecting the regularization parameter λ_n to satisfy $\lambda_n d \leq C_{\min}^2 \frac{\varepsilon}{36K(2-\varepsilon)}$ yields:

$$\frac{\|\boldsymbol{R}^n\|_\infty}{\lambda_n} \leq \frac{3K\lambda_n d}{C_{\min}^2} \leq \frac{\varepsilon}{4(2-\varepsilon)}$$

12.8 Proof of Lemma 7

We will first bound the difference in terms of nuclear norm between the population Fisher information matrix $\mathcal{Q}_{SS}$ and the sample mean cascade log-likelihood $\mathcal{Q}_{SS}^n$. Define $z_{jk}^c = [\nabla^2 g(\mathbf{t}^c; \boldsymbol{\alpha}^*) - \nabla^2 \ell^n(\boldsymbol{\alpha}^*)]_{jk}$ and $z_{jk} = \frac{1}{n}\sum_{c=1}^n z_{jk}^c$. Then, we can express the difference between the population Fisher information matrix $\mathcal{Q}_{SS}$ and the sample mean cascade log-likelihood $\mathcal{Q}_{SS}^n$ as:

$$|\|\mathcal{Q}_{SS}^n(\boldsymbol{\alpha}^*) - \mathcal{Q}_{SS}^*(\boldsymbol{\alpha}^*)\|\|_2 \leq |\|\mathcal{Q}_{SS}^n(\boldsymbol{\alpha}^*) - \mathcal{Q}_{SS}^*(\boldsymbol{\alpha}^*)\|\|_F = \sqrt{\sum_{j=1}^d \sum_{k=1}^d (z_{ik})^2}.$$

Since $|z_{jk}^{(c)}| \leq 2k_5$ by condition 4, we can apply Hoeffding's inequality to each z_{jk},

$$P(|z_{jk}| \geq \beta) \leq 2\exp\left(-\frac{\beta^2 n}{8k_5^2}\right), \tag{30}$$

and further,

$$P(|\|\mathcal{Q}_{SS}^n(\boldsymbol{\alpha}^*) - \mathcal{Q}_{SS}^*(\boldsymbol{\alpha}^*)\|\|_2 \geq \delta) \leq 2\exp\big(-K\frac{\delta^2 n}{d^2} + 2\log d\big) \tag{31}$$

where $\beta^2 = \delta^2/d^2$. Now, we bound the maximum eigenvalue of $\mathcal{Q}_{SS}^n$ as follows:

$$\begin{aligned}
\Lambda_{\max}(\mathcal{Q}_{SS}^n) &= \max_{\|x\|_2=1} x^\top \mathcal{Q}_{SS}^n x = \max_{\|x\|_2=1} \{x^\top \mathcal{Q}_{SS}^* x + x^\top (\mathcal{Q}_{SS}^n - \mathcal{Q}_{SS}^*) x\} \\
&\leq y^\top \mathcal{Q}_{SS}^* y + y^\top (\mathcal{Q}_{SS}^n - \mathcal{Q}_{SS}^*) y,
\end{aligned}$$

where y is unit-norm maximal eigenvector of $\mathcal{Q}_{SS}^*$. Therefore,

$$\Lambda_{\max}(\mathcal{Q}_{SS}^n) \leq \Lambda_{\max}(\mathcal{Q}_{SS}^*) + |\|\mathcal{Q}_{SS}^n - \mathcal{Q}_{SS}^*\|\|_2,$$

and thus,

$$P\big(\Lambda_{\max}(\mathcal{Q}_{SS}^n) \geq C_{\max} + \delta\big) \leq \exp\left(-K\frac{\delta^2 n}{d^2} + 2\log d\right).$$

Reasoning in a similar way, we bound the minimum eigenvalue of $\mathcal{Q}_{SS}^n$:

$$P\big(\Lambda_{\min}(\mathcal{Q}_{SS}^n) \leq C_{\min} - \delta\big) \leq \exp\left(-K\frac{\delta^2 n}{d^2} + 2\log d\right)$$

12.9 Proof of Lemma 8

We start by decomposing $\mathcal{Q}_{S^cS}^n(\boldsymbol{\alpha}^*)(\mathcal{Q}_{S^cS}^n(\boldsymbol{\alpha}^*))^{-1}$ as follows:

$$\mathcal{Q}_{S^cS}^n(\boldsymbol{\alpha}^*)(\mathcal{Q}_{S^cS}^n(\boldsymbol{\alpha}^*))^{-1} = A_1 + A_2 + A_3 + A_4,$$

where,

$$A_1 = \mathcal{Q}^*_{S^cS}[(\mathcal{Q}^n_{S^cS})^{-1} - (\mathcal{Q}^*_{S^cS})^{-1}],$$
$$A_2 = [\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}][(\mathcal{Q}^n_{S^cS})^{-1} - (\mathcal{Q}^*_{S^cS})^{-1}]$$
$$A_3 = [\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}](\mathcal{Q}^*_{SS})^{-1},$$
$$A_4 = \mathcal{Q}^*_{S^cS}(\mathcal{Q}^*_{SS})^{-1},$$

$\mathcal{Q}^* = \mathcal{Q}^*(\boldsymbol{\alpha}^*)$ and $\mathcal{Q}^n = \mathcal{Q}^n(\boldsymbol{\alpha}^*)$. Now, we bound each term separately. The fourth term, A_4, is the easiest to bound, using simply the incoherence condition:

$$|||A_4|||_\infty \leq 1 - \varepsilon.$$

To bound the other terms, we need the following lemma:

Lemma 11 *For any $\delta \geq 0$ and constants K and K', the following bounds hold:*

$$P[|||\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}|||_\infty \geq \delta] \leq 2\exp\left(-K\frac{n\delta^2}{d^2} + \log d + log(p-d)\right) \tag{32}$$

$$P[|||\mathcal{Q}^n_{SS} - \mathcal{Q}^*_{SS}|||_\infty \geq \delta] \leq 2\exp\left(-K\frac{n\delta^2}{d^2} + 2\log d\right) \tag{33}$$

$$P[|||(\mathcal{Q}^n_{SS})^{-1} - (\mathcal{Q}^*_{SS})^{-1}|||_\infty \geq \delta] \leq 4\exp\left(-K\frac{n\delta}{d^3} - K'\log d\right) \tag{34}$$

Proof We start by proving the first confidence interval. By definition of infinity norm of a matrix, we have:

$$P[|||\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}|||_\infty \geq \delta] = P\big[\max_{j\in S^c}\sum_{k\in S}|z_{jk}| \geq \delta\big] \leq (p-d)P\big[\sum_{k\in S}|z_{jk}| \geq \delta\big],$$

where $z_{jk} = [\mathcal{Q}^n - \mathcal{Q}^*]_{jk}$ and, for the last inequality, we used the union bound and the fact that $|S^c| \leq p - d$. Furthermore,

$$P\big[\sum_{k\in S}|z_{jk}| \geq \delta\big] \leq P[\exists k \in S||z_{jk}| \geq \delta/d] \leq dP[|z_{jk}| \geq \delta/d].$$

Thus,

$$P[|||\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}|||_\infty \geq \delta] \leq (p-d)dP[|z_{jk}| \geq \delta/d].$$

At this point, we can obtain the first confidence bound by using Eq. 30 with $\beta = \delta/d$ in the above equation. The proof of the second confidence bound is very similar and we omit it for brevity. To prove the last confidence bound, we proceed as follows:

$$\begin{aligned}
|||(\mathcal{Q}^n_{SS})^{-1} - (\mathcal{Q}^*_{SS})^{-1}|||_\infty &= |||(\mathcal{Q}^n_{SS})^{-1}[\mathcal{Q}^n_{SS} - \mathcal{Q}^*_{SS}](\mathcal{Q}^*_{SS})^{-1}|||_\infty \\
&\leq \sqrt{d}|||(\mathcal{Q}^n_{SS})^{-1}[\mathcal{Q}^n_{SS} - \mathcal{Q}^*_{SS}](\mathcal{Q}^*_{SS})^{-1}|||_2 \\
&\leq \sqrt{d}|||(\mathcal{Q}^n_{SS})^{-1}|||_2|||\mathcal{Q}^n_{SS} - \mathcal{Q}^*_{SS}|||_2|||(\mathcal{Q}^*_{SS})^{-1}|||_2 \\
&\leq \frac{\sqrt{d}}{C_{\min}}|||\mathcal{Q}^n_{SS} - \mathcal{Q}^*_{SS}|||_2|||(\mathcal{Q}^n_{SS})^{-1}|||_2.
\end{aligned}$$

Next, we bound each term of the final expression in the above equation separately. The first term can be bounded using Eq. 31:

$$P\left[|||\mathcal{Q}^n_{SS} - \mathcal{Q}^*_{SS}|||_2 \geq \frac{C^2_{min}\delta}{2\sqrt{d}}\right] \leq 2\exp\left(-K\frac{n\delta^2}{d^3} + 2\log d\right),$$

The second term can be bounded using Lemma 6:

$$P\left[|||(\mathcal{Q}^n_{SS})^{-1}|||_2 \geq \frac{2}{C_{\min}}\right] = P\left[\Lambda_{\min}(\mathcal{Q}^n_{SS}) \leq \frac{C_{\min}}{2}\right] \leq \exp\left(-K\frac{n}{d^2} + B\log d\right).$$

Then, the third confidence bound follows. ■

Control of A_1. We start by rewriting the term A_1 as

$$A_1 = \mathcal{Q}^*_{S^cS}(\mathcal{Q}^*_{SS})^{-1}[(\mathcal{Q}^*_{SS}) - (\mathcal{Q}^n_{SS})](\mathcal{Q}^n_{SS})^{-1},$$

and further,

$$|||A_1|||_\infty \leq |||\mathcal{Q}^*_{S^cS}(\mathcal{Q}^*_{SS})^{-1}|||_\infty \times |||(\mathcal{Q}^*_{SS}) - (\mathcal{Q}^n_{SS})|||_\infty|||(\mathcal{Q}^n_{SS})^{-1}|||_\infty.$$

Next, using the incoherence condition easily yields:

$$|||A_1|||_\infty \leq (1-\varepsilon)|||(\mathcal{Q}^*_{SS}) - (\mathcal{Q}^n_{SS})|||_\infty \times \sqrt{d}|||(\mathcal{Q}^n_{SS})^{-1}|||_2$$

Now, we apply Lemma 6 with $\delta = C_{\min}/2$ to have that $|||(\mathcal{Q}^n_{SS})^{-1}|||_2 \leq \frac{2}{C_{\min}}$ with probability greater than $1 - \exp(-Kn/d^2 + K'\log d)$, and then use Eq. 34 with $\delta = \frac{\varepsilon C_{\min}}{12\sqrt{d}}$ to conclude that

$$P\left[|||A_1|||_\infty \geq \frac{\varepsilon}{6}\right] \leq 2\exp\left(-K\frac{n}{d^3} + K'\log d\right).$$

Control of A_2. We rewrite the term A_2 as

$$|||A_2|||_\infty \leq |||\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}|||_\infty|||(\mathcal{Q}^n_{SS})^{-1} - (\mathcal{Q}^*_{SS})^{-1}|||_\infty,$$

and then use Eqs. 32 and 33 with $\delta = \sqrt{\varepsilon/6}$ to conclude that

$$P\left[|||A_2|||_\infty \geq \frac{\varepsilon}{6}\right] \leq 4\exp\left(-K\frac{n}{d^3} + \log(p-d) + K'\log p\right).$$

Control of A_3. We rewrite the term A_3 as

$$|||A_3|||_\infty = \sqrt{d}|||(\mathcal{Q}^*_{SS})^{-1}|||_2|||\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}|||_\infty \leq \frac{\sqrt{d}}{C_{\min}}|||\mathcal{Q}^n_{S^cS} - \mathcal{Q}^*_{S^cS}|||_\infty.$$

We then apply Eq. 32 with $\delta = \frac{\varepsilon C_{\min}}{6\sqrt{d}}$ to conclude that

$$P\left[|||A_3|||_\infty \geq \frac{\varepsilon}{6}\right] \leq \exp\left(-K\frac{n}{d^3} + \log(p-d)\right),$$

and thus,

$$P\left[|||\mathcal{Q}^n_{S^cS}(\mathcal{Q}^n_{SS})^{-1}|||_\infty \geq 1 - \frac{\varepsilon}{2}\right] = \mathcal{O}\left(\exp(-K\frac{n}{d^3} + \log p)\right).$$

References

B. Abrahao, F. Chierichetti, R. Kleinberg, and A. Panconesi. Trace complexity of network inference. In *KDD*, 2013.

E. Adar and L. A. Adamic. Tracking Information Epidemics in Blogspace. In *Web Intelligence*, pages 207–214, 2005.

A.-L. Barabási and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286: 509–512, 1999.

A. Beck and M. Teboulle. Gradient-based algorithms with applications to signal recovery. *Convex Optimization in Signal Processing and Communications*, 2009.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

N. Du, L. Song, A. Smola, and M. Yuan. Learning Networks of Heterogeneous Influence. In *NIPS*, 2012a.

N. Du, L. Song, H. Woo, and H. Zha. Uncover Topic-Sensitive Information Diffusion Networks. In *AISTATS*, 2012b.

Nan Du, Le Song, Hyenkyun Woo, and Hongyuan Zha. Uncover topic-sensitive information diffusion networks. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 229–237, 2013.

M. Gomez-Rodriguez. *Ph.D. Thesis*. Stanford University & MPI for Intelligent Systems, 2013.

M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring Networks of Diffusion and Influence. In *KDD*, 2010.

M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the Temporal Dynamics of Diffusion Networks. In *ICML*, 2011.

M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Structure and Dynamics of Information Pathways in On-line Media. In *WSDM*, 2013a.

M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Modeling Information Propagation with Survival Theory. In *ICML '13: Proceedings of the 30th International Conference on Machine Learning*, 2013b.

M. Gomez-Rodriguez, J. Leskovec, D. Balduzzi, and B. Schölkopf. Uncovering the Structure and Temporal Dynamics of Information Propagation. *Network Science*, 2014.

V. Gripon and M. Rabbat. Reconstructing a graph from path traces. *arXiv:1301.6916*, 2013.

Martin Jaggi. Revisiting {Frank-Wolfe}: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 427–435, 2013.

D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the Spread of Influence Through a Social Network. In *KDD*, 2003.

J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker Graphs: An Approach to Modeling Networks. *JMLR*, 2010.

O. L. Mangasarian. A simple characterization of solution sets of convex programs. *Operations Research Letters*, 7(1):21–26, 1988.

S. Myers and J. Leskovec. Clash of the contagions: Cooperation and competition in information diffusion. In *Proceedings of the IEEE International Conference on Data Mining*, 2012.

S. Myers, J. Leskovec, and C. Zhu. Information Diffusion and External Influence in Networks. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012.

Sahand Negahban, Bin Yu, Martin J Wainwright, and Pradeep K Ravikumar. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, pages 1348–1356, 2009.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust Stochastic Approximation Approach to Stochastic Programming. *SIAM Journal on Optimization*, 19(4):1574, 2009.

P. Netrapalli and S. Sanghavi. Finding the Graph of Epidemic Cascades. In *ACM SIGMETRICS*, 2012.

W. K. Newey and D. L. McFadden. Large Sample Estimation and Hypothesis Testing. In *Handbook of Econometrics*, volume 4, pages 2111–2245. 1994.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 2013.

B.A. Prakash, A. Beutel, R. Rosenfeld, and C. Faloutsos. Winner Takes All: Competing Viruses or Ideas on Fair-Play Networks. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1037–1046, 2012.

P. Ravikumar, M. J. Wainwright, and J. D. Lafferty. High-dimensional ising model selection using l1-regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, 2010.

Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

E. M. Rogers. *Diffusion of Innovations*. Free Press, New York, fourth edition, 1995.

K. Saito, M. Kimura, K. Ohara, and H. Motoda. Learning continuous-time information diffusion model for social behavioral data analysis. *Advances in Machine Learning*, pages 322–337, 2009.

T. Snowsill, N. Fyson, T. De Bie, and N. Cristianini. Refining Causality: Who Copied From Whom? In *KDD*, 2011.

M. J. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using l1-constrained quadratic programming (lasso). *IEEE Transactions on Information Theory*, 55(5):2183–2202, 2009.

L. Wang, S. Ermon, and J. Hopcroft. Feature-enhanced probabilistic models for diffusion network inference. In *ECML PKDD*, 2012.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

Ke Zhou, Hongyuan Zha, and Le Song. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 641–649, 2013.

Journal of Machine Learning Research 17 (2016) 1-42 Submitted 10/14; Revised 12/15; Published 4/16

Rounding-based Moves for Semi-Metric Labeling

M. Pawan Kumar PAWAN@ROBOTS.OX.AC.UK
Department of Engineering Science
University of Oxford
Parks Road OX1 3PJ
United Kingdom

Puneet K. Dokania PUNEET.KUMAR@ECP.FR
Center for Visual Computing
CentraleSupelec
92295 Châtenay-Malabry
France

Editor: Jeff Bilmes

Abstract

Semi-metric labeling is a special case of energy minimization for pairwise Markov random fields. The energy function consists of arbitrary unary potentials, and pairwise potentials that are proportional to a given semi-metric distance function over the label set. Popular methods for solving semi-metric labeling include (i) move-making algorithms, which iteratively solve a minimum st-cut problem; and (ii) the linear programming (LP) relaxation based approach. In order to convert the fractional solution of the LP relaxation to an integer solution, several randomized rounding procedures have been developed in the literature. We consider a large class of parallel rounding procedures, and design move-making algorithms that closely mimic them. We prove that the multiplicative bound of a move-making algorithm exactly matches the approximation factor of the corresponding rounding procedure for any arbitrary distance function. Our analysis includes all known results for move-making algorithms as special cases.

Keywords: semi-metric labeling, move-making algorithms, linear programming relaxation, multiplicative bounds

1. Introduction

A Markov random field (MRF) is a graph whose vertices are random variables, and whose edges specify a neighborhood over the random variables. Each random variable can be assigned a value from a set of labels, resulting in a labeling of the MRF. The putative labelings of an MRF are quantitatively distinguished from each other by an energy function, which is the sum of potential functions that depend on the cliques of the graph. An important optimization problem associate with the MRF framework is energy minimization, that is, finding a labeling with the minimum energy.

Semi-metric labeling is a special case of energy minimization, which models several useful low-level vision tasks (Boykov et al., 1998, 1999; Szeliski et al., 2008). It is characterized by a finite, discrete label set and a semi-metric distance function over the labels. The energy function in semi-metric labeling consists of arbitrary unary potentials and pairwise potentials that are proportional to the distance between the labels assigned to them. The problem is known to be NP-hard (Veksler, 1999). Two popular approaches for semi-metric labeling are: (i) move-making algorithms (Boykov et al., 1999; Gupta and Tardos, 2000; Kumar and Koller, 2009; Kumar and Torr, 2008; Veksler, 2007), which iteratively improve the labeling by solving a minimum st-cut problem; and (ii) linear programming (LP) relaxation (Chekuri et al., 2001; Koster et al., 1998; Schlesinger, 1976; Wainwright et al., 2005), which is obtained by dropping the integral constraints in the corresponding integer programming formulation. Move-making algorithms are very efficient due to the availability of fast minimum st-cut solvers (Boykov and Kolmogorov, 2004) and are very popular in the computer vision community. In contrast, the LP relaxation is significantly slower, despite the development of several specialized solvers (Globerson and Jaakkola, 2007; Hazan and Shashua, 2008; Kolmogorov, 2006; Komodakis et al., 2007; Ravikumar et al., 2008; Tarlow et al., 2011; Wainwright et al., 2005; Weiss et al., 2007; Werner, 2007, 2010). However, when used in conjunction with randomized rounding algorithms, the LP relaxation provides the best known polynomial-time theoretical guarantees for semi-metric labeling (Archer et al., 2004; Chekuri et al., 2001; Kleinberg and Tardos, 1999).

At first sight, the difference between move-making algorithms and the LP relaxation appears to be the standard accuracy vs. speed trade-off. However, we prove that, for any arbitrary semi-metric distance function, it is possible to design move-making algorithms that match the theoretical guarantees of a large class of randomized rounding procedures, which we call parallel rounding. Our proofs are constructive, which allows us to test the rounding-based move-making algorithms empirically. Our experimental results confirm that rounding-based moves provide similar accuracy to the LP relaxation while being significantly faster.

2. Related Work

As our work is concerned with only semi-metric labeling, we will focus on the related work for this special case of energy minimization. For a more general survey, we refer the reader to (Wang et al., 2013), and for a thorough empirical comparison of the various algorithms we refer the reader to (Kappes et al., 2015; Szeliski et al., 2008).

The earliest known theoretical guarantee of a move-making algorithm was provided by Veksler (1999), who presented an analysis of the popular expansion move-making algorithm. The analysis showed that the expansion algorithm and the parallel rounding procedure of Kleinberg and Tardos (1999) provide matching guarantees for the uniform distance function. Komodakis et al. (2008) also proposed a move-making algorithm based on the primal-dual scheme of optimizing the LP relaxation. The resulting algorithm is similar to the expansion algorithm in terms of its theoretical guarantees, but provides a principled formulation for improving its efficiency in dynamic settings. Despite the existing analysis of expansion-like algorithms, parallel rounding procedures (Chekuri et al., 2001; Kleinberg and Tardos, 1999) are known to perform better for several distance functions of interest such as the truncated linear distance, the truncated quadratic distance, and the hierarchically well-separated tree (HST) metric.

Kumar and Torr (2008) proposed the range expansion move-making algorithm that matches the guarantees of the rounding procedure of Chekuri et al. (2001) for the truncated linear and quadratic distance. Kumar and Koller (2009) proposed a hierarchical move-making algorithm that matches the guarantees of the rounding procedure of Kleinberg and Tardos (1999) for the HST metric. However, their analysis is restricted to special cases of semi-metric labeling, whereas our work does not make

any assumptions on the form of the semi-metric distance function. We note that, recently, a more general algorithm has also been proposed for non-convex priors (Ajanthan et al., 2014). However, this generalized algorithm does not provide any strong theoretical gurantees on the quality of the solution, which is the key focus of our work.

While our analysis focuses on parallel rounding, it is also worth noting that Archer et al. (2004) have proposed a serial rounding procedure for metric labeling (which is a special case of semi-metric labeling where the distance function also satisfies the triangular inequality property). The theoretical guarantee of serial rounding depends on the *tree decomposibility* of a distance function. Given an MRF with n random variables, the tree decomposibility for any metric distance function is guaranteed to be less than or equal to $O(\log n)$. This worst-case theoretical guarantee of $O(\log n)$ over all metric distance functions can be matched by a mixture-of-tree algorithm that can be derived form the work of Andrew et al. (2011). Specifically, Andrew et al. (2011) describe an approximate algorithm for capacitated metric labeling that builds on the cut-based decomposition method of Racke (2008). Ignoring the capacity constraints, which are not present in the standard metric labeling problem, results in an approximation of the original MRF using an $O(m \log n)$ mixture of tree-structured MRFs. Here, m is the number of edges in the MRF, and n is the number of random variables. Solving the metric labeling problem over each tree-structured MRF optimally using belief propagation (Pearl, 1998), and choosing the best labeling in terms of the original problem, provides the $O(\log n)$ guarantee. We refer the reader to (Andrew et al., 2011) for details.

Note that while the mixture-of-tree algorithm provides matching worst-case guarantees to serial rounding, it does not provide a matching guarantee for a given distance function. In other words, the guarantee of the mixture-of-tree algorithm does not depend on the tree decomposibility of the distance function. The existence or the impossibility of a matching combinatorial algorithm for serial rounding remains an open question.

3. Our Contributions

Our paper can be thought of as a generalization of (Kumar and Koller, 2009; Kumar and Torr, 2008; Veksler, 1999). Indeed, the move-making algorithms proposed in this paper are simple extensions of the expansion, the range expansion and the hierarchical move-making algorithms. The novelty of our work lies in our analysis, which improves on the state of the art in two significant aspects. First, we do not place any assumptions on the form of the distance function except that it is a semi-metric distance. This is in contrast to previous works that only focus on special cases such as the uniform metric, the truncated convex models or the HST metric. Second, we show that the matching guarantees provided by our move-making algorithms and the parallel rounding procedures are *tight*, that is, they cannot be improved through a more careful analysis.

A preliminary version of this article has appeared as (Kumar, 2014). There are two significant differences between the preliminary version and the current paper. First, we provide detailed proofs of all the theorems, which were omitted from the preliminary version due to a lack of space. Second, we provide an experimental comparison of the rounding-based move-making algorithms with the state of the art methods for semi-metric labeling, including those that are directly related to the LP relaxation such as TRW-S (Kolmogorov, 2006), and those that are indirectly related to the LP relaxation such as (Andrew et al., 2011).

Finally, we note that since the publication of the preliminary version, we have also extended rounding-based move-making algorithms to special cases of high-order potentials. This includes the

interval move-making algorithm for truncated max-of-convex potentials (Pansari and Kumar, 2015) and the hierarchical move-making algorithm for parsimonious labeling (Dokania and Kumar, 2015).

4. Preliminaries

4.1 Semi-metric Labeling

The problem of semi-metric labeling is defined over an undirected graph $\mathbf{G} = (\mathbf{X}, \mathbf{E})$. The vertices $\mathbf{X} = \{X_1, X_2, \cdots, X_n\}$ are random variables, and the edges $\mathbf{E}$ specify a neighborhood relationship over the random variables. Each random variable can be assigned a value from the label set $\mathbf{L} = \{l_1, l_2, \cdots, l_h\}$. We assume that we are also provided with a semi-metric distance function $d : \mathbf{L} \times \mathbf{L} \rightarrow \mathbb{R}^+$ over the labels. Recall that a semi-metric distance function satisfies the following properties: $d(l_i, l_j) \geq 0$ for all $l_i, l_j \in \mathbf{L}$, and $d(l_i, l_j) = 0$ if and only if $i = j$. Furthermore, a distance function is said to be metric if, in addition to the above condition, it also satisfies the triangular inequality, that is, $d(l_i, l_j) + d(l_j, l_k) \geq d(l_i, l_k)$ for all $l_i, l_j, l_k \in \mathbf{L}$.

We refer to an assignment of values to all the random variables as a labeling. In other words, a labeling is a vector $\mathbf{x} \in \mathbf{L}^n$, which specifies the label x_a assigned to each random variable X_a. The h^n different labelings are quantitatively distinguished from each other by an energy function $Q(\mathbf{x})$, which is defined as follows:

$$Q(\mathbf{x}) = \sum_{X_a \in \mathbf{X}} \theta_a(x_a) + \sum_{(X_a, X_b) \in \mathbf{E}} w_{ab} d(x_a, x_b).$$

Here, the unary potentials $\theta_a(\cdot)$ are arbitrary, and the edge weights w_{ab} are non-negative. Semi-metric labeling requires us to find a labeling with the minimum energy. It is known to be NP-hard.

4.2 Multiplicative Bound

As semi-metric labeling plays a central role in low-level vision, several approximate algorithms have been proposed in the literature. A common theoretical measure of accuracy for an approximate algorithm is the multiplicative bound. In this work, we are interested in the multiplicative bound of an algorithm with respect to a distance function. Formally, given a distance function d, the multiplicative bound of an algorithm is said to be B if the following condition is satisfied for all possible values of unary potentials $\theta_a(\cdot)$ and non-negative edge weights w_{ab}:

$$\sum_{X_a \in \mathbf{X}} \theta_a(\hat{x}_a) + \sum_{(X_a, X_b) \in \mathbf{E}} w_{ab} d(\hat{x}_a, \hat{x}_b) \leq \sum_{X_a \in \mathbf{X}} \theta_a(x^*_a) + B \sum_{(X_a, X_b) \in \mathbf{E}} w_{ab} d(x^*_a, x^*_b). \quad (1)$$

Here, $\hat{\mathbf{x}}$ is the labeling estimated by the algorithm for the given values of unary potentials and edge weights, and $\mathbf{x}^*$ is an optimal labeling. Multiplicative bounds are greater than or equal to 1, and are invariant to reparameterizations of the unary potentials. A multiplicative bound B is said to be tight if the above inequality holds as an equality for some value of unary potentials and edge weights.

4.3 Linear Programming Relaxation

An overcomplete representation of a labeling can be specified using the following variables: (i) unary variables $y_a(i) \in \{0, 1\}$ for all $X_a \in \mathbf{X}$ and $l_i \in \mathbf{L}$ such that $y_a(i) = 1$ if and only if X_a is assigned the label l_i; and (ii) pairwise variables $y_{ab}(i, j) \in \{0, 1\}$ for all $(X_a, X_b) \in \mathbf{E}$ and $l_i, l_j \in \mathbf{L}$ such

that $y_{ab}(i,j) = 1$ if and only if X_a and X_b are assigned labels l_i and l_j respectively. This allows us to formulate semi-metric labeling as follows:

$$\begin{aligned}
\min_{\mathbf{y}} \quad & \sum_{X_a \in \mathbf{X}} \sum_{l_i \in \mathbf{L}} \theta_a(l_i) y_a(i) + \sum_{(X_a,X_b) \in \mathbf{E}} \sum_{l_i,l_j \in \mathbf{L}} w_{ab} d(l_i,l_j) y_{ab}(i,j), \\
\text{s.t.} \quad & \sum_{l_i \in \mathbf{L}} y_a(i) = 1, \forall X_a \in \mathbf{X}, \\
& \sum_{l_j \in \mathbf{L}} y_{ab}(i,j) = y_a(i), \forall (X_a,X_b) \in \mathbf{E}, l_i \in \mathbf{L}, \\
& \sum_{l_i \in \mathbf{L}} y_{ab}(i,j) = y_b(j), \forall (X_a,X_b) \in \mathbf{E}, l_j \in \mathbf{L}, \\
& y_a(i) \in \{0,1\}, y_{ab}(i,j) \in \{0,1\}, \forall X_a \in \mathbf{X}, (X_a,X_b) \in \mathbf{E}, l_i, l_j \in \mathbf{L}.
\end{aligned}$$

The first set of constraints ensures that each random variables is assigned exactly one label. The second and third sets of constraints ensure that, for binary optimization variables, $y_{ab}(i,j) = y_a(i) y_b(j)$. By relaxing the final set of constraints such that the optimization variables can take any value between 0 and 1 inclusive, we obtain a linear program (LP). The computational complexity of solving the LP relaxation is polynomial in the size of the problem.

4.4 Rounding Procedure

In order to prove theoretical guarantees of the LP relaxation, it is common to use a rounding procedure that can covert a feasible fractional solution $\mathbf{y}$ of the LP relaxation to a feasible integer solution $\hat{\mathbf{y}}$ of the integer linear program. Several rounding procedures have been proposed in the literature. In this work, we focus on the randomized parallel rounding procedures proposed by Chekuri et al. (2001) and Kleinberg and Tardos (1999). These procedures have the property that, given a fractional solution $\mathbf{y}$, the probability of assigning a label $l_i \in \mathbf{L}$ to a random variable $X_a \in \mathbf{X}$ is equal to $y_a(i)$, that is,

$$\Pr(\hat{y}_a(i) = 1) = y_a(i). \tag{2}$$

We will describe the various rounding procedures in detail in Sections 5-7. For now, we would like to note that our reason for focusing on the parallel rounding of Chekuri et al. (2001) and Kleinberg and Tardos (1999) is that they provide the best known polynomial-time theoretical guarantees for semi-metric labeling. Specifically, we are interested in their approximation factor, which is defined next.

4.5 Approximation Factor

Given a distance function d, the approximation factor for a rounding procedure is said to be F if the following condition is satisfied for all feasible fractional solutions $\mathbf{y}$:

$$\mathbb{E}\left(\sum_{l_i,l_j \in \mathbf{L}} d(l_i,l_j) \hat{y}_a(i) \hat{y}_b(j) \right) \leq F \sum_{l_i,l_j \in \mathbf{L}} d(l_i,l_j) y_{ab}(i,j). \tag{3}$$

Here, $\hat{\mathbf{y}}$ refers to the integer solution, and the expectation is taken with respect to the randomized rounding procedure applied to the feasible solution $\mathbf{y}$.

Given a rounding procedure with an approximation factor of F, an optimal fractional solution $\mathbf{y}^*$ of the LP relaxation can be rounded to a labeling $\hat{\mathbf{y}}$ that satisfies the following condition:

$$\begin{aligned}
& \mathbb{E}\left(\sum_{X_a \in \mathbf{X}} \sum_{l_i \in \mathbf{L}} \theta_a(l_i) \hat{y}_a(i) + \sum_{(X_a,X_b) \in \mathbf{E}} \sum_{l_i,l_j \in \mathbf{L}} w_{ab} d(l_i,l_j) \hat{y}_a(i) \hat{y}_b(j) \right) \\
\leq & \sum_{X_a \in \mathbf{X}} \sum_{l_i \in \mathbf{L}} \theta_a(l_i) y^*_a(i) + F \sum_{(X_a,X_b) \in \mathbf{E}} \sum_{l_i,l_j \in \mathbf{L}} w_{ab} d(l_i,l_j) y^*_{ab}(i,j).
\end{aligned}$$

The above inequality follows directly from properties (2) and (3). Similar to multiplicative bounds, approximation factors are always greater than or equal to 1, and are invariant to reparameterizations of the unary potentials. An approximation factor F is said to be tight if the above inequality holds as an equality for some value of unary potentials and edge weights.

Approximation factors are closely linked to the integrality gap of the LP relaxation (roughly speaking, the ratio of the optimal value of the integer linear program to the optimal value of the relaxation), which in turn is related to the computational hardness of the semi-metric labeling problem (Manokaran et al., 2008). However, establishing the integrality gap of the LP relaxation for a given distance function is beyond the scope of this work. We are only interested in designing move-making algorithms whose multiplicative bounds match the approximation factors of the parallel rounding procedures.

4.6 Submodular Energy Function

We will use the following important fact throughout this paper. Given an energy function defined using arbitrary unary potentials, non-negative edge weights and a submodular distance function, an optimal labeling can be computed in polynomial time by solving an equivalent minimum st-cut problem (Flach and Schlesinger, 2006). Recall that a submodular distance function d' over a label set $\mathbf{L} = \{l_1, l_2, \cdots, l_h\}$ satisfies the following properties: (i) $d'(l_i,l_j) \geq 0$ for all $l_i, l_j \in \mathbf{L}$, and $d'(l_i,l_j) = 0$ if and only if $i = j$; and (ii) $d'(l_i,l_j) + d'(l_{i+1},l_{j+1}) \leq d'(l_i,l_{j+1}) + d'(l_{i+1},l_j)$ for all $l_i, l_j \in \mathbf{L} \backslash \{l_h\}$ (where $\backslash$ refers to set difference).

5. Complete Rounding and Complete Move

We start with a simple rounding scheme, which we call complete rounding. While complete rounding is not very accurate, it would help illustrate the flavor of our results. We will subsequently consider its generalizations, which have been useful in obtaining the best-known approximation factors for various special cases of semi-metric labeling.

The complete rounding procedure consists of a single stage where we use the set of all unary variables to obtain a labeling (as opposed to other rounding procedures discussed subsequently). Algorithm 1 describes its main steps. Intuitively, it treats the value of the unary variable $y_a(i)$ as the probability of assigning the label l_i to the random variable X_a. It obtains a labeling by sampling from all the distributions $\mathbf{y}_a = [y_a(i), \forall l_i \in \mathbf{L}]$ simultaneously using the same random number $r \in [0,1]$.

It can be shown that using a different random number to sample the distributions $\mathbf{y}_a$ and $\mathbf{y}_b$ of two neighboring random variables $(X_a, X_b) \in \mathbf{E}$ results in an infinite approximation factor. For example, let $\overline{y}_a(i) = \overline{y}_b(i) = 1/h$ for all $l_i \in \mathbf{L}$, where h is the number of labels. The pairwise variables $\overline{\mathbf{y}}_{ab}$ that minimize the energy function are $\overline{y}_{ab}(i,i) = 1/h$ and $\overline{y}_{ab}(i,j) = 0$ when $i \neq j$.

For the above feasible solution of the LP relaxation, the RHS of inequality (3) is 0 for any finite F, while the LHS of inequality (3) is strictly greater than 0 if $h > 1$. However, we will shortly show that using the same random number r for all random variables provides a finite approximation factor.

Algorithm 1 The complete rounding procedure.

input A feasible solution $\mathbf{y}$ of the LP relaxation.

1: Pick a real number r uniformly from $[0, 1]$.
2: **for all** $X_a \in \mathbf{X}$ **do**
3: Define $Y_a(0) = 0$ and $Y_a(i) = \sum_{j=1}^{i} y_a(j)$ for all $l_i \in \mathbf{L}$.
4: Assign the label $l_i \in \mathbf{L}$ to the random variable X_a if $Y_a(i-1) < r \leq Y_a(i)$.
5: **end for**

We now turn our attention to designing a move-making algorithm whose multiplicative bound matches the approximation factor of the complete rounding procedure. To this end, we modify the range expansion algorithm proposed by Kumar and Torr (2008) for truncated convex pairwise potentials to a general semi-metric distance function. Our method, which we refer to as the complete move-making algorithm, considers all putative labels of all random variables, and provides an approximate solution in a single iteration. Algorithm 2 describes its two main steps. First, it computes a submodular overestimation of the given semi-metric distance function by solving the following optimization problem:

$$\begin{aligned} \overline{d} = \quad & \operatorname*{argmin}_{d'} t \\ \text{s.t.} \quad & d'(l_i, l_j) \leq t d(l_i, l_j), \forall l_i, l_j \in \mathbf{L}, \\ & d'(l_i, l_j) \geq d(l_i, l_j), \forall l_i, l_j \in \mathbf{L}, \\ & d'(l_i, l_j) + d'(l_{i+1}, l_{j+1}) \leq d'(l_i, l_{j+1}) + d'(l_{i+1}, l_j), \forall l_i, l_j \in \mathbf{L} \backslash \{l_h\}. \end{aligned} \tag{4}$$

The above problem minimizes the maximum ratio of the estimated distance to the original distance over all pairs of labels, that is,

$$\max_{i \neq j} \frac{d'(l_i, l_j)}{d(l_i, l_j)}.$$

We will refer to the optimal value of problem (4) as the submodular distortion of the distance function d. Second, it replaces the original distance function by the submodular overestimation and computes an approximate solution to the original semi-metric labeling problem by solving a single minimum st-cut problem. Note that, unlike the range expansion algorithm (Kumar and Torr, 2008) that uses the readily available submodular overestimation of a truncated convex distance (namely, the corresponding convex distance function), our approach estimates the submodular overestimation via the LP (4). Since the LP (4) can be solved for any arbitrary distance function, it makes complete move-making more generally applicable.

The following theorem establishes the theoretical guarantees of the complete move-making algorithm and the complete rounding procedure.

Theorem 1 *The tight multiplicative bound of the complete move-making algorithm is equal to the submodular distortion of the distance function. Furthermore, the tight approximation factor of the complete rounding procedure is also equal to the submodular distortion of the distance function.*

Algorithm 2 The complete move-making algorithm.

input Unary potentials $\theta_a(\cdot)$, edge weights w_{ab}, distance function d.

1: Compute a submodular overestimation of d by solving problem (4).
2: Using the approach of Flach and Schlesinger (2006), solve the following problem via an equivalent minimum st-cut problem:

$$\hat{\mathbf{x}} = \operatorname*{argmin}_{\mathbf{x} \in \mathbf{L}^n} \sum_{X_a \in \mathbf{X}} \theta_a(x_a) + \sum_{(X_a, X_b) \in \mathbf{E}} w_{ab} \overline{d}(x_a, x_b).$$

The proof of Theorem 1 is given in Appendix A. The following corollary of the above theorem was previously stated by Chekuri et al. (2001) without a formal proof.

Corollary 2 *The complete rounding procedure is tight for submodular distance functions, that is, its approximation factor is equal to 1.*

In terms of computational complexities, complete move-making is significantly faster than solving the LP relaxation. Specifically, given an MRF with n random variables and m edges, and a label set with h labels, the LP relaxation requires at least $O(m^3 h^3 \log(m^2 h^3))$ time, since it consists of $O(mh^2)$ optimization variables and $O(mh)$ constraints. In contrast, complete move-making requires $O(nmh^3 \log(m))$ time, since the graph constructed using the method of Flach and Schlesinger (2006) consists of $O(nh)$ nodes and $O(mh^2)$ arcs. Note that complete move-making also requires us to solve the linear program (4). However, since problem (4) is independent of the unary potentials and the edge weights, it only needs to be solved once beforehand in order to compute the approximate solution for any semi-metric labeling problem defined using the distance function d.

6. Interval Rounding and Interval Moves

Theorem 1 implies that the approximation factor of the complete rounding procedure is very large for distance functions that are highly non-submodular. For example, consider the truncated linear distance function defined as follows over a label set $\mathbf{L} = \{l_1, l_2, \cdots, l_h\}$:

$$d(l_i, l_j) = \min\{|i - j|, M\}.$$

Here, M is a user specified parameter that determines the maximum distance. The tightest submodular overestimation of the above distance function is the linear distance function, that is, $d(l_i, l_j) = |i - j|$. This implies that the submodular distortion of the truncated linear metric is $(h-1)/M$, and therefore, the approximation factor for the complete rounding procedure is also $(h-1)/M$. In order to avoid this large approximation factor, Chekuri et al. (2001) proposed an interval rounding procedure, which captures the intuition that it is beneficial to assign similar labels to as many random variables as possible.

Algorithm 3 provides a description of interval rounding. The rounding procedure chooses an interval of at most q consecutive labels (step 2). It generates a random number r (step 3), and uses it to attempt to assign labels to previously unlabeled random variables from the selected interval (steps 4-7). It can be shown that the overall procedure converges in a polynomial number of iterations with a probability of 1 (Chekuri et al., 2001). Note that if we fix $q = h$ and $z = 1$, interval

rounding becomes equivalent to complete rounding. However, the analyses of Chekuri et al. (2001) and Kleinberg and Tardos (1999) shows that other values of q provide better approximation factors for various special cases.

Algorithm 3 The interval rounding procedure.

input A feasible solution $\mathbf{y}$ of the LP relaxation.

1: **repeat**
2: Pick an integer z uniformly from $[-q+2, h]$. Define an interval of labels $\mathbf{I} = \{l_s, \cdots, l_e\}$, where $s = \max\{z, 1\}$ is the start index and $e = \min\{z+q-1, h\}$ is the end index.
3: Pick a real number r uniformly from $[0, 1]$.
4: **for all** Unlabeled random variables X_a **do**
5: Define $Y_a(0) = 0$ and $Y_a(i) = \sum_{j=s}^{s+i-1} y_a(j)$ for all $i \in \{1, \cdots, e-s+1\}$.
6: Assign the label $l_{s+i-1} \in \mathbf{I}$ to the X_a if $Y_a(i-1) < r \leq Y_a(i)$.
7: **end for**
8: **until** All random variables have been assigned a label.

Our goal is to design a move-making algorithm whose multiplicative bound matches the approximation factor of interval rounding for any choice of q. To this end, we propose the interval move-making algorithm that generalizes the range expansion algorithm (Kumar and Torr, 2008), originally proposed for truncated convex distances, to arbitrary distance functions. Algorithm 4 provides its main steps. The central idea of the method is to improve a given labeling $\hat{\mathbf{x}}$ by allowing each random variable X_a to either retain its current label $\hat{x}_a$ or to choose a new label from an interval of consecutive labels. In more detail, let $\mathbf{I} = \{l_s, \cdots, l_e\} \subseteq \mathbf{L}$ be an interval of labels of length at most q (step 4). For the sake of simplicity, let us assume that $\hat{x}_a \notin \mathbf{I}$ for any random variable X_a. We define $\mathbf{I}_a = \mathbf{I} \bigcup \{\hat{x}_a\}$ (step 5). For each pair of neighboring random variables $(X_a, X_b) \in \mathbf{E}$, we compute a submodular distance function $\overline{d}_{\hat{x}_a,\hat{x}_b} : \mathbf{I}_a \times \mathbf{I}_b \rightarrow \mathbb{R}^+$ by solving the following linear program (step 6):

$$\begin{aligned}
\overline{d}_{\hat{x}_a,\hat{x}_b} = \quad & \operatorname*{argmin}_{d'} t \qquad (5)\\
\text{s.t.} \quad & d'(l_i, l_j) \leq t d(l_i, l_j), \forall l_i \in \mathbf{I}_a, l_j \in \mathbf{I}_b,\\
& d'(l_i, l_j) \geq d(l_i, l_j), \forall l_i \in \mathbf{I}_a, l_j \in \mathbf{I}_b,\\
& d'(l_i, l_j) + d'(l_{i+1}, l_{j+1}) \leq d'(l_i, l_{j+1}) + d'(l_{i+1}, l_j), \forall l_i, l_j \in \mathbf{I} \backslash \{l_e\},\\
& d'(l_i, l_e) + d'(l_{i+1}, \hat{x}_b) \leq d'(l_i, \hat{x}_b) + d'(l_{i+1}, l_e), \forall l_i \in \mathbf{I} \backslash \{l_e\},\\
& d'(l_e, l_j) + d'(\hat{x}_a, l_{j+1}) \leq d'(l_e, l_{j+1}) + d'(\hat{x}_a, l_j), \forall l_j \in \mathbf{I} \backslash \{l_e\},\\
& d'(l_e, l_e) + d(\hat{x}_a, \hat{x}_b) \leq d'(l_e, \hat{x}_b) + d'(\hat{x}_a, l_e).
\end{aligned}$$

Similar to problem (4), the above problem minimizes the maximum ratio of the estimated distance to the original distance. However, instead of introducing constraints for all pairs of labels, it only considers pairs of labels l_i and l_j where $l_i \in \mathbf{I}_a$ and $l_j \in \mathbf{I}_b$. Furthermore, it does not modify the distance between the current labels $\hat{x}_a$ and $\hat{x}_b$ (as can be seen in the last constraint of problem (5)).

Given the submodular distance functions $\overline{d}_{\hat{x}_a,\hat{x}_b}$, we can compute a new labeling $\overline{\mathbf{x}}$ by solving the following optimization problem via minimum st-cut using the method of Flach and Schlesinger

(2006) (step 7):

$$\begin{aligned}
\overline{\mathbf{x}} = \quad & \operatorname*{argmin}_{\mathbf{x}} \sum_{X_a \in \mathbf{X}} \theta_a(x_a) + \sum_{(X_a, X_b) \in \mathbf{E}} w_{ab} \overline{d}_{\hat{x}_a,\hat{x}_b}(x_a, x_b)\\
\text{s.t.} \quad & x_a \in \mathbf{I}_a, \forall X_a \in \mathbf{X}. \qquad (6)
\end{aligned}$$

If the energy of the new labeling $\overline{\mathbf{x}}$ is less than that of the current labeling $\hat{\mathbf{x}}$, then we update our labeling to $\overline{\mathbf{x}}$ (steps 8-10). Otherwise, we retain the current estimate of the labeling and consider another interval. The algorithm converges when the energy does not decrease for any interval of length at most q. Note that, once again, the main difference between interval move-making and the range expansion algorithm is the use of an appropriate optimization problem, namely the LP (5), to obtain a submodular overestimation of the given distance function. This allows us to use the interval move-making algorithm for the general semi-metric labeling problem, instead of focusing on only truncated convex models.

Algorithm 4 The interval move-making algorithm.

input Unary potentials $\theta_a(\cdot)$, edge weights w_{ab}, distance function d, initial labeling $\mathbf{x}^0$.

1: Set current labeling to initial labeling, that is, $\hat{\mathbf{x}} = \mathbf{x}^0$.
2: **repeat**
3: **for all** $z \in [-q+2, h]$ **do**
4: Define an interval of labels $\mathbf{I} = \{l_s, \cdots, l_e\}$, where $s = \max\{z, 1\}$ is the start index and $e = \min\{z+q-1, h\}$ is the end index.
5: Define $\mathbf{I}_a = \mathbf{I} \bigcup \{\hat{x}_a\}$ for all random variables $X_a \in \mathbf{X}$.
6: Obtain submodular overestimates $\overline{d}_{\hat{x}_a,\hat{x}_b}$ for each pair of neighboring random variables $(X_a, X_b) \in \mathbf{E}$ by solving problem (5).
7: Obtain a new labeling $\overline{\mathbf{x}}$ by solving problem (6).
8: **if** Energy of $\overline{\mathbf{x}}$ is less than energy of $\hat{\mathbf{x}}$ **then**
9: Update $\hat{\mathbf{x}} = \overline{\mathbf{x}}$.
10: **end if**
11: **end for**
12: **until** Energy cannot be decreased further.

The following theorem establishes the theoretical guarantees of the interval move-making algorithm and the interval rounding procedure.

Theorem 3 *The tight multiplicative bound of the interval move-making algorithm is equal to the tight approximation factor of the interval rounding procedure.*

The proof of Theorem 3 is given in Appendix B. While Algorithms 3 and 4 use intervals of consecutive labels, they can easily be modified to use subsets of (potentially non-consecutive) labels. Our analysis could be extended to show that the multiplicative bound of the resulting subset move-making algorithm matches the approximation factor of the subset rounding procedure. However, our reason for focusing on intervals of consecutive labels is that several special cases of Theorem 3 have previously been considered separately in the literature (Gupta and Tardos, 2000; Kumar and Koller, 2009; Kumar and Torr, 2008; Veksler, 1999). Specifically, the following known results are corollaries of the above theorem. Note that, while the following corollaries have been previously proved in the literature, our work is the first to establish the tightness of the theoretical guarantees.

Corollary 4 *When $q = 1$, the multiplicative bound of the interval move-making algorithm (which is equivalent to the expansion algorithm) for the uniform metric distance is 2.*

The above corollary follows from the approximation factor of the interval rounding procedure proved by Kleinberg and Tardos (1999), but it was independently proved by Veksler (1999).

Corollary 5 *When $q = M$, the multiplicative bound of the interval move-making algorithm for the truncated linear distance function is 4.*

The above corollary follows from the approximation factor of the interval rounding procedure proved by Chekuri et al. (2001), but it was independently proved by Gupta and Tardos (2000).

Corollary 6 *When $q = \sqrt{2}M$, the multiplicative bound of the interval move-making algorithm for the truncated linear distance function is $2 + \sqrt{2}$.*

The above corollary follows from the approximation factor of the interval rounding procedure proved by Chekuri et al. (2001), but it was independently proved by Kumar and Torr (2008). Finally, since our analysis does not use the triangular inequality of metric distance functions, we can also state the following corollary for the truncated quadratic distance.

Corollary 7 *When $q = \sqrt{M}$, the multiplicative bound of the interval move-making algorithm for the truncated quadratic distance function is $O(\sqrt{M})$.*

The above corollary follows from the approximation factor of the interval rounding procedure proved by Chekuri et al. (2001), but it was independently proved by Kumar and Torr (2008).

An interval move-making algorithm that uses an interval length of q runs for at most $O(h/q)$ iterations. This follows from a simple modification of the result by Gupta and Tardos (2000) (specifically, theorem 3.7). Hence, the total time complexity of interval move-making is $O(nmhq^2 \log(m))$, since each iteration solves a minimum st-cut problem of a graph with $O(nq)$ nodes and $O(mq^2)$ arcs. In other words, interval move-making is at most as computationally complex as complete move-making, which in turn is significantly less complex than solving the LP relaxation. Note that problem (5), which is required for interval move-making, is independent of the unary potentials and the edge weights. Hence, it only needs to be solved once beforehand for all pairs of labels $(\hat{x}_a, \hat{x}_b) \in \mathbf{L} \times \mathbf{L}$ in order to obtain a solution for any semi-metric labeling problem defined using the distance function d.

7. Hierarchical Rounding and Hierarchical Moves

We now consider the most general form of parallel rounding that has been proposed in the literature, namely the hierarchical rounding procedure (Kleinberg and Tardos, 1999). The rounding relies on a hierarchical clustering of the labels. Formally, we denote a hierarchical clustering of m levels for the label set $\mathbf{L}$ by $\mathbf{C} = \{\mathbf{C}(i), i = 1, \cdots, m\}$. At each level i, the clustering $\mathbf{C}(i) = \{\mathbf{C}(i,j) \subseteq \mathbf{L}, j = 1, \cdots, h^i\}$ is mutually exclusive and collectively exhaustive, that is,

$$\bigcup_j \mathbf{C}(i,j) = \mathbf{L}, \mathbf{C}(i,j) \cap \mathbf{C}(i,j') = \emptyset, \forall j \neq j'.$$

Furthermore, for each cluster $\mathbf{C}(i,j)$ at the level $i > 2$, there exists a unique cluster $\mathbf{C}(i-1,j')$ in the level $i-1$ such that $\mathbf{C}(i,j) \subseteq \mathbf{C}(i-1,j')$. We call the cluster $\mathbf{C}(i-1,j')$ the parent of the

Algorithm 5 The hierarchical rounding procedure.

input A feasible solution $\mathbf{y}$ of the LP relaxation.

1: Define $f_a^1 = 1$ for all $X_a \in \mathbf{X}$.
2: **for all** $i \in \{2, \cdots, m\}$ **do**
3: **for all** $X_a \in \mathbf{X}$ **do**
4: Define $z_a^i(j)$ for all $j \in \{1, \cdots, h^i\}$ as follows:

$$z_a^i(j) = \begin{cases} \sum_k \text{ s.t. } l_k \in \mathbf{C}(i,j)\, y_a(k) & \text{if } p(i,j) = f_a^{i-1}, \\ 0 & \text{otherwise.} \end{cases}$$

5: Define $y_a^i(j)$ for all $j \in \{1, \cdots, h^i\}$ as follows:

$$y_a^i(j) = \frac{z_a^i(j)}{\sum_{j'=1}^{h^i} z_a^i(j')}$$

6: **end for**
7: Using a rounding procedure (complete or interval) on $\mathbf{y}^i = [y_a^i(j), \forall X_a \in \mathbf{X}, j \in \{1, \cdots, h^i\}]$, obtain an integer solution $\hat{\mathbf{y}}^i$.
8: **for all** $X_a \in \mathbf{X}$ **do**
9: Let $k_a \in \{1, \cdots, h^i\}$ such that $\hat{y}^i(k_a) = 1$. Define $f_a^i = k_a$.
10: **end for**
11: **end for**
12: **for all** $X_a \in \mathbf{X}$ **do**
13: Let l_k be the unique label present in the cluster $\mathbf{C}(m, f_a^m)$. Assign l_k to X_a.
14: **end for**

cluster $\mathbf{C}(i,j)$ and define $p(i,j) = j'$. Similarly, we call $\mathbf{C}(i,j)$ a child of $\mathbf{C}(i-1,j')$. Without loss of generality, we assume that there exists a single cluster at level 1 that contains all the labels, and that each cluster at level m contains a single label.

Algorithm 5 describes the hierarchical rounding procedure. Given a clustering $\mathbf{C}$, it proceeds in a top-down fashion through the hierarchy while assigning each random variable to a cluster in the current level. Let f_a^i be the index of the cluster assigned to the random variable X_a in the level i. In the first step, the rounding procedure assigns all the random variables to the unique cluster $\mathbf{C}(1,1)$ (step 1). At each step i, it assigns each random variable to a unique cluster in the level i by computing a conditional probability distribution as follows. The conditional probability $y_a^i(j)$ of assigning the random variable X_a to the cluster $\mathbf{C}(i,j)$ is proportional to $\sum_{l_k \in \mathbf{C}(i,j)} y_a(k)$ if $p(i,j) = f_a^{i-1}$ (steps 3-6). The conditional probability $y_a^i(j) = 0$ if $p(i,j) \neq f_a^{i-1}$, that is, a random variable cannot be assigned to a cluster $\mathbf{C}(i,j)$ if it wasn't assigned to its parent in the previous step. Using a rounding procedure (complete or interval) for $\mathbf{y}^i$, we obtain an assignment of random variables to the clusters at level i (step 7). Once such an assignment is obtained, the values f_a^i are computed for all random variables X_a (steps 8-10). At the end of step m, hierarchical rounding would have assigned each random variable to a unique cluster in the level m. Since each cluster at level m consists of a single label, this provides us with a labeling of the MRF (steps 12-14).

Algorithm 6 The hierarchical move-making algorithm.

input Unary potentials $\theta_a(\cdot)$, edge weights w_{ab}, distance function d.

1: **for all** $j \in \{1, \cdots, h\}$ **do**
2: Let l_k be the unique label in the cluster $\mathbf{C}(m,j)$. Define $x_a^{m,j} = l_k$ for all $X_a \in \mathbf{X}$.
3: **end for**
4: **for all** $i \in \{2, \cdots, m\}$ **do**
5: **for all** $j \in \{1, \cdots, h^{m-i+1}\}$ **do**
6: Define $\mathbf{L}_a^{m-i+1,j} = \{x_a^{m-i+2,j'}, p(m-i+2,j') = j, j' \in \{1, \cdots, h^{m-i+2}\}\}$.
7: Using a move-making algorithm (complete or interval), compute the labeling $\mathbf{x}^{m-i+1,j}$ under the constraint $x_a^{m-i+1,j} \in \mathbf{L}_a^{m-i+1,j}$.
8: **end for**
9: **end for**
10: The final solution is $\mathbf{x}^{1,1}$.

Our goal is to design a move-making algorithm whose multiplicative bound matches the approximation factor of the hierarchical rounding procedure for any choice of hierarchical clustering $\mathbf{C}$. To this end, we propose the hierarchical move-making algorithm, which extends the hierarchical graph cuts approach for hierarchically well-separated tree (HST) metrics proposed by Kumar and Koller (2009). Algorithm 6 provides its main steps. In contrast to hierarchical rounding, the move-making algorithm traverses the hierarchy in a bottom-up fashion while computing a labeling for each cluster in the current level. Let $\mathbf{x}^{i,j}$ be the labeling corresponding to the cluster $\mathbf{C}(i,j)$. At the first step, when considering the level m of the clustering, all the random variables are assigned the same label. Specifically, $x_a^{m,j}$ is equal to the unique label contained in the cluster $\mathbf{C}(m,j)$ (steps 1-3). At step i, it computes the labeling $\mathbf{x}^{m-i+1,j}$ for each cluster $\mathbf{C}(m-i+1,j)$ by using the labelings computed in the previous step. Specifically, it restricts the label assigned to a random variable X_a in the labeling $\mathbf{x}^{m-i+1,j}$ to the subset of labels that were assigned to it by the labelings corresponding to the children of $\mathbf{C}(m-i+1,j)$ (step 6). Under this restriction, the labeling $\mathbf{x}^{m-i+1,j}$ is computed

by approximately minimizing the energy using a move-making algorithm (step 7). Implicit in our description is the assumption that that we will use a move-making algorithm (complete or interval) in step 7 of Algorithm 6 whose multiplicative bound matches the approximation factor of the rounding procedure (complete or interval) used in step 7 of Algorithm 5. Note that, unlike the hierarchical graph cuts approach (Kumar and Koller, 2009), the hierarchical move-making algorithm can be used for any arbitrary clustering and not just the one specified by an HST metric.

The following theorem establishes the theoretical guarantees of the hierarchical move-making algorithm and the hierarchical rounding procedure.

Theorem 8 *The tight multiplicative bound of the hierarchical move-making algorithm is equal to the tight approximation factor of the hierarchical rounding procedure.*

The proof of the above theorem is given in Appendix C. The following known result is its corollary.

Corollary 9 *The multiplicative bound of the hierarchical move-making algorithm is $O(1)$ for an HST metric distance.*

The above corollary follows from the approximation factor of the hierarchical rounding procedure proved by Kleinberg and Tardos (1999), but it was independently proved by Kumar and Koller (2009). It is worth noting that the above result was also used to obtain an approximation factor of $O(\log h)$ for the general metric labeling problem by Kleinberg and Tardos (1999) and a matching multiplicative bound of $O(\log h)$ by Kumar and Koller (2009).

The time complexity of the hierarchical move-making algorithm depends on two factors: (i) the hierarchy, which defines the subproblems; and (ii) whether we use a complete or an interval move-making algorithm to solve the subproblems. In what follows, we analyze its worst case time complexity over all possible hierarchies. Clearly, since complete move-making is more expensive than interval move-making, the worst case complexity of hierarchical move-making will be achieved when we use complete move-making to solve each subproblem. We begin by noting that each problem is defined using a smaller label set. For simplicitly, let us assume that the clusters of the hierarchy are balanced, and that each subproblem is solved over $h' \leq h$ labels. It follows that the total number of subproblems that we would be required to solve is $O((h/h')^{(\log(h)/\log(h'))-1})$. Next, observe that the complexity of the complete move-making algorithms is cubic in the number of labels. Thus, it follows that in order to maximize the total time complexity of hierarchical move-making, we need to set $h' = h$. In other words, hierarchical move-making algorithm is at most as computationally complex as the complete move-making algorithm. Recall that the complete move-making complexity is $O(nmh^3 \log(m))$. Hence, hierarchical move-making is significantly faster than solving the LP relaxation.

8. Experiments

We demonstrate the efficacy of rounding-based moves by comparing them to several state of the art methods using both synthetic and real data.

8.1 Synthetic Experiments

8.1.1 DATA

We generated two synthetic data sets to conduct our experiments. The first data set consists of random grid MRFs of size 100×100, where each random variable can take one of 10 labels. The unary

potentials were sampled from a uniform distribution over $[0, 10]$. The edge weights were sampled from a uniform distribution over $[0, 3]$. We considered four types of pairwise potentials: (i) truncated linear metric, where the truncation is sampled from a uniform distribution over $[1, 5]$; (ii) truncated quadratic semi-metric, where the truncation is sampled from a uniform distribution over $[1, 25]$; (iii) random metrics, generated by computing the shortest path on graphs whose vertices correspond to the labels and whose edge lengths are uniformly distributed over $[1, 10]$; (iv) random semi-metrics, where the distance between two labels is sampled from a uniform distribution over $[1, 10]$. For each type of pairwise potentials, we generated 500 different MRFs.

The second data set is similar to the first one, except that it is defined on a smaller grid of size 20×20. The smaller grid size allows us to test the mixture-of-tree algorithm (Andrew et al., 2011) that is closely related to the serial rounding procedure of Archer et al. (2004). We generated 5 different grids by sampling their edge weights from a uniform distribution over $[5, 100]$. For each set of edge weights, we use the cut-based decomposition method of Racke (2008) to approximate the original grid graph using a mixture of tree-structured graphs. We generated three types of unary potentials by uniformly sampling from the range $[0, 100]$, $[100, 1000]$ and $[1000, 10000]$ respectively. Similar to the first data set, we considered four types of pairwise potentials. For each pair of unary potential type and pairwise potential type, we generated 100 different MRFs. This provided us with 1200 MRFs for each set of edge weights, and a total of 6000 MRFs to perform our experiments.

8.1.2 METHODS

We report results obtained by the following state of the art methods using the first data set: (i) belief propagation (BP) (Pearl, 1998); (ii) sequential tree-reweighted message passing (TRW) (Kolmogorov, 2006), which optimizes the dual of the LP relaxation, and provides comparable results to other LP relaxation based approaches; (iii) expansion algorithm (EXP) (Boykov et al., 1999); and (iv) swap algorithm (SWAP) (Boykov et al., 1998). We compare the above methods to a hierarchy move-making algorithm (HIER), where a set of hierarchies is obtained by approximating a given semi-metric as a mixture of r-HST metrics using the method proposed by Fakcharoenphol et al. (2003). We refer the reader to (Fakcharoenphol et al., 2003; Kumar and Koller, 2009) for details. Each subproblem of the hierarchical move-making algorithm is solved by interval move-making with interval length $q = 1$ (which corresponds to the expansion algorithm). In addition, for the truncated linear and truncated quadratic cases, we present results of interval move-making (INT) using the optimal interval length reported by Kumar and Torr (2008).

For the second data set, we report the results for all the above methods, as well as the mixture-of-tree (MOT) algorithm. The smaller size of the grid in the second data set allows us to obtain a mixture of tree-structured MRFs using the method of Racke (2008), which was not possible for the larger 100×100 grid used in the first data set.

8.1.3 RESULTS

Figure 1 shows the results for the first data set. In terms of the energy, TRW is the most accurate. However, it is slow as it optimizes the dual of the LP relaxation. The labelings obtained by BP have high energy values. The standard move-making algorithms, EXP and SWAP, are fast due to the use of efficient minimum st-cut solvers. However, they are not as accurate as TRW. For the truncated linear and quadratic pairwise potentials, INT provides labelings with comparable energy to those of TRW, and is also computationally efficient. However, for general metrics and semi-metrics,

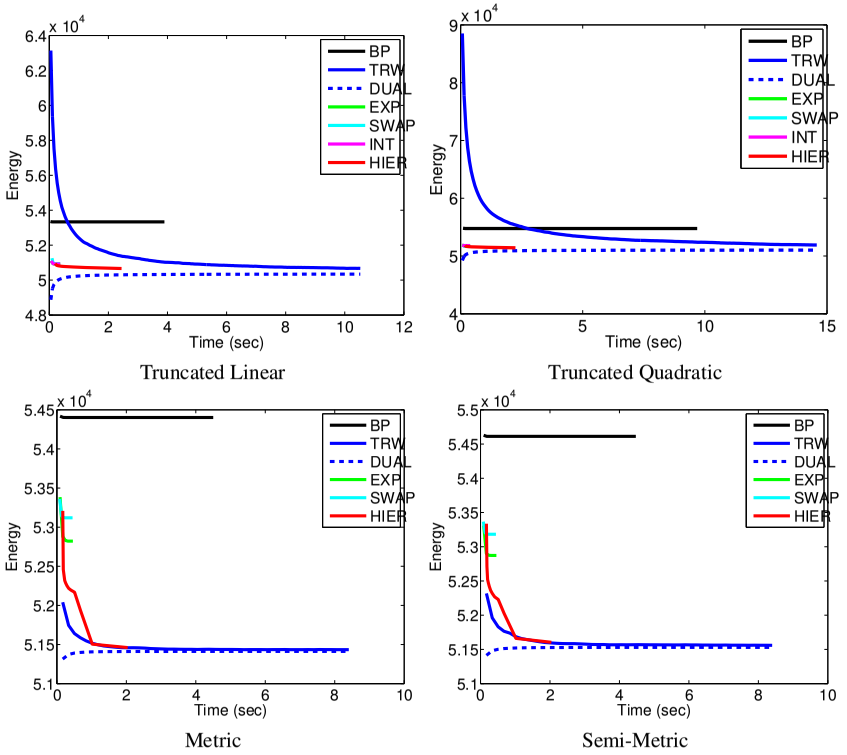


Figure 1: Results for the first synthetic data set. The x-axis shows the time in seconds, while the y-axis shows the energy value. The dashed line shows the value of the dual of the LP obtained by TRW. Best viewed in color.

it is not obvious how to obtain the optimal interval length. The HIER method is more generally applicable as there exist standard methods to approximate a semi-metric with a mixture of r-HST metrics (Fakcharoenphol et al., 2003). It provides very accurate labelings (comparable to TRW), and is efficient in practice as it relies on solving each subproblem using an iterative move-making algorithm.

Figure 2 shows the results for the second data set. The results of all the methods are similar to the first data set. However, the additional MOT algorithm performs very poorly compared to all other methods. This may be explained by the fact that, while MOT provides the same worst-case guarantees as serial rounding over all possible metric distance functions, it does not match the accuracy of serial rounding for a given distance function.

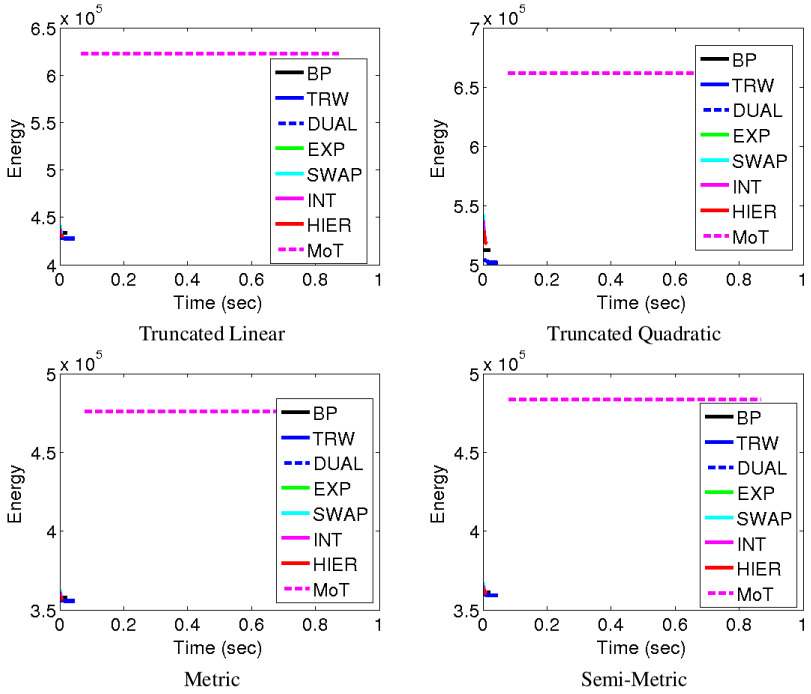


Figure 2: Results for the second synthetic data set. The x-axis shows the time in seconds, while the y-axis shows the energy value. The dashed line shows the value of the dual of the LP obtained by TRW. Best viewed in color.

8.2 Dense Stereo Correspondence

8.2.1 DATA

Given two epipolar rectified images of the same scene, the problem of dense stereo correspondence requires us to obtain a correspondence between the pixels of the images. This problem can be modeled as semi-metric labeling, where the random variables represent the pixels of one of the images, and the labels represent the disparity values. A disparity label l_i for a random variable X_a representing a pixel (u_a, v_a) of an image indicates that its corresponding pixel lies in location $(u_a + i, v_a)$. For the above problem, we use the unary potentials and edge weights that are specified by Szeliski et al. (2008). We use two types of pairwise potentials: (i) truncated linear with the truncation set at 4; and (ii) truncated quadratic with the truncation set at 16.

8.2.2 METHODS

We report results on all the baseline methods that were used in the synthetic experiments, namely, BP, TRW, EXP, and SWAP. Since the pairwise potentials are either truncated linear or truncated quadratic, we report results for the interval move-making algorithm INT, which uses the optimal value of the interval length. We also show the results obtained by the hierarchical move-making algorithm (HIER), where once again the hierarchies are obtained by approximating the semi-metric as a mixture of r-HST metrics.

8.2.3 RESULTS

Figure 3-Figure 8 shows the results for various standard pairs of images. Note that, similar to the synthetic experiments, TRW is the most accurate in terms of energy, but it is computationally inefficient. The results obtained by BP are not accurate. The standard move-making algorithms, EXP and SWAP, are fast but not as accurate as TRW. Among the rounding-based move-making algorithms INT is slower as it solves a minimum st-cut problem on a large graph at each iteration. In contrast, HIER uses an interval length of 1 for each subproblem and is therefore more efficient. The energy obtained by HIER is comparable to TRW.

9. Discussion

For any general distance function that can be used to specify the semi-metric labeling problem, we proved that the approximation factor of a large family of parallel rounding procedures is matched by the multiplicative bound of move-making algorithms. This generalizes previously known results on the guarantees of move-making algorithms in two ways: (i) in contrast to previous results (Kumar and Koller, 2009; Kumar and Torr, 2008; Veksler, 1999) that focused on special cases of distance functions, our results are applicable to arbitrary semi-metric distance functions; and (ii) the guarantees provided by our theorems are tight. Our experiments confirm that the rounding-based move-making algorithms provide similar accuracy to the LP relaxation, while being significantly faster due to the use of efficient minimum st-cut solvers.

Several natural questions arise. What is the exact characterization of the rounding procedures for which it is possible to design matching move-making algorithms? Can we design rounding-based move-making algorithms for other combinatorial optimization problems? Answering these questions

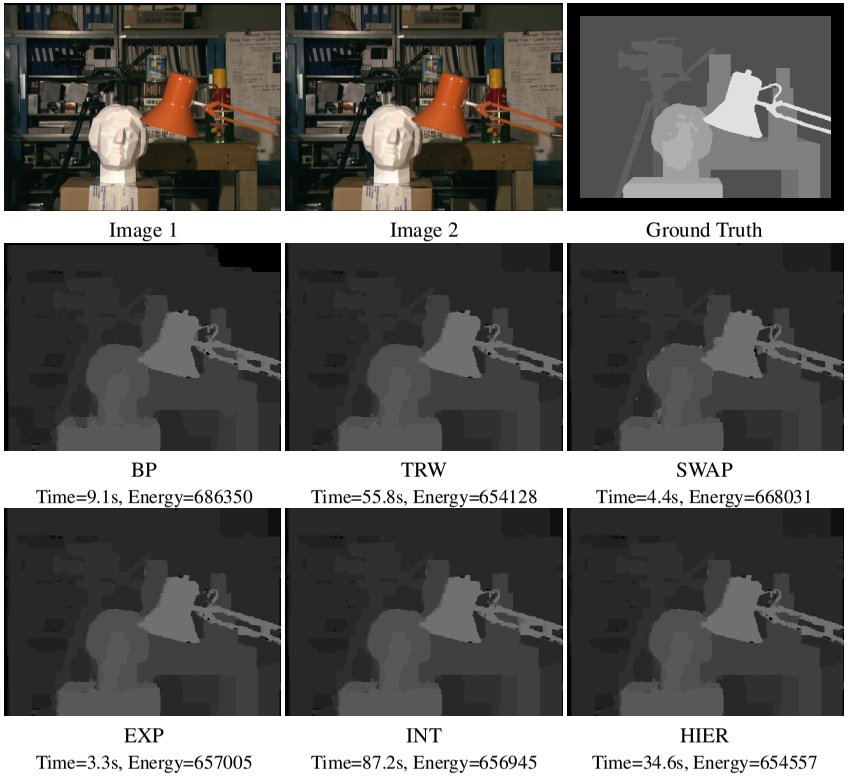


Figure 3: Results for the 'tsukuba' image pair with truncated linear pairwise potentials.

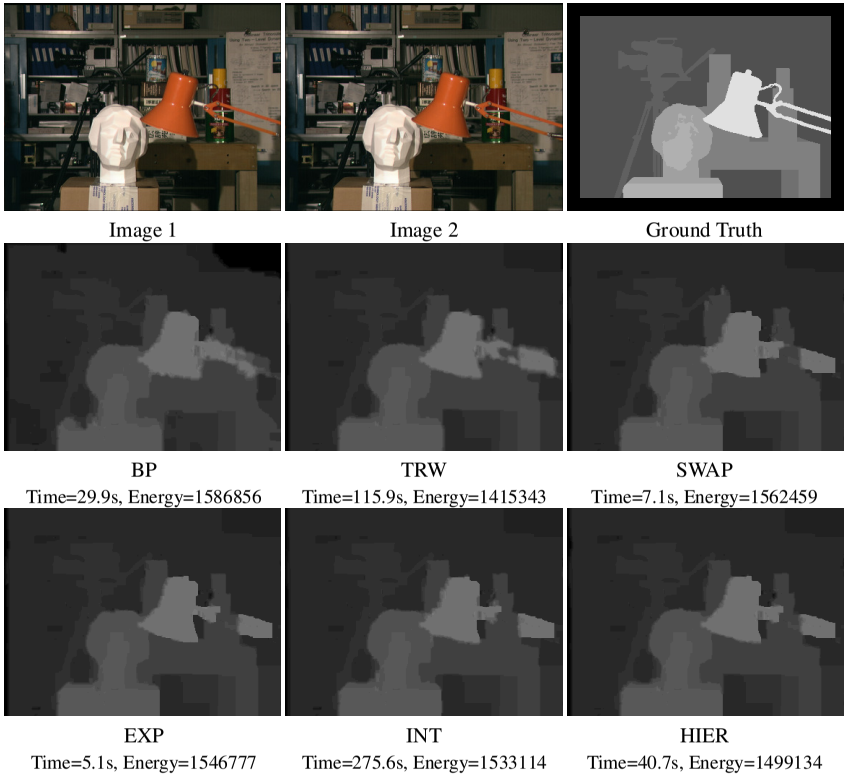


Figure 4: Results for the 'tsukuba' image pair with truncated quadratic pairwise potentials.

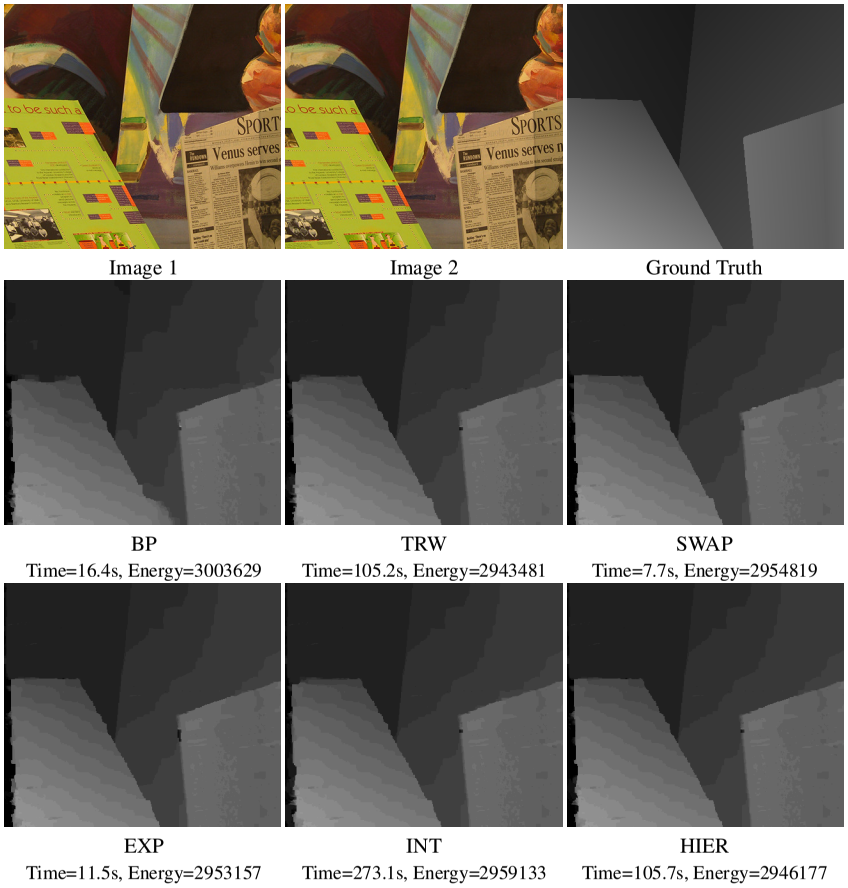


Figure 5: Results for the 'venus' image pair with truncated linear pairwise potentials.

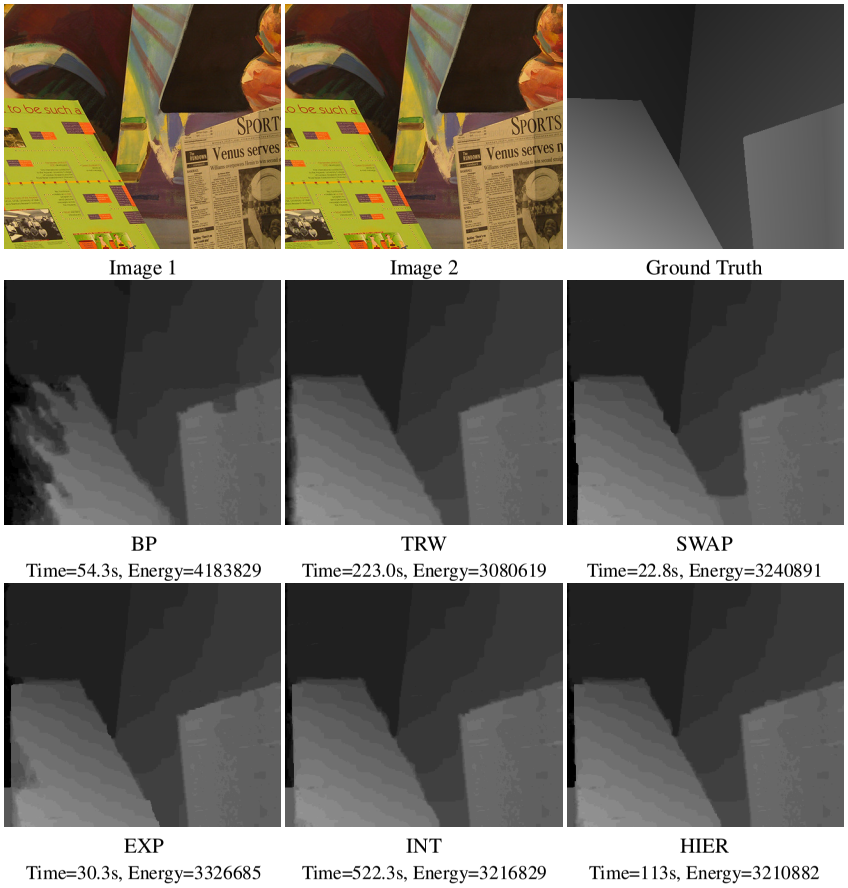


Figure 6: Results for the 'venus' image pair with truncated quadratic pairwise potentials.

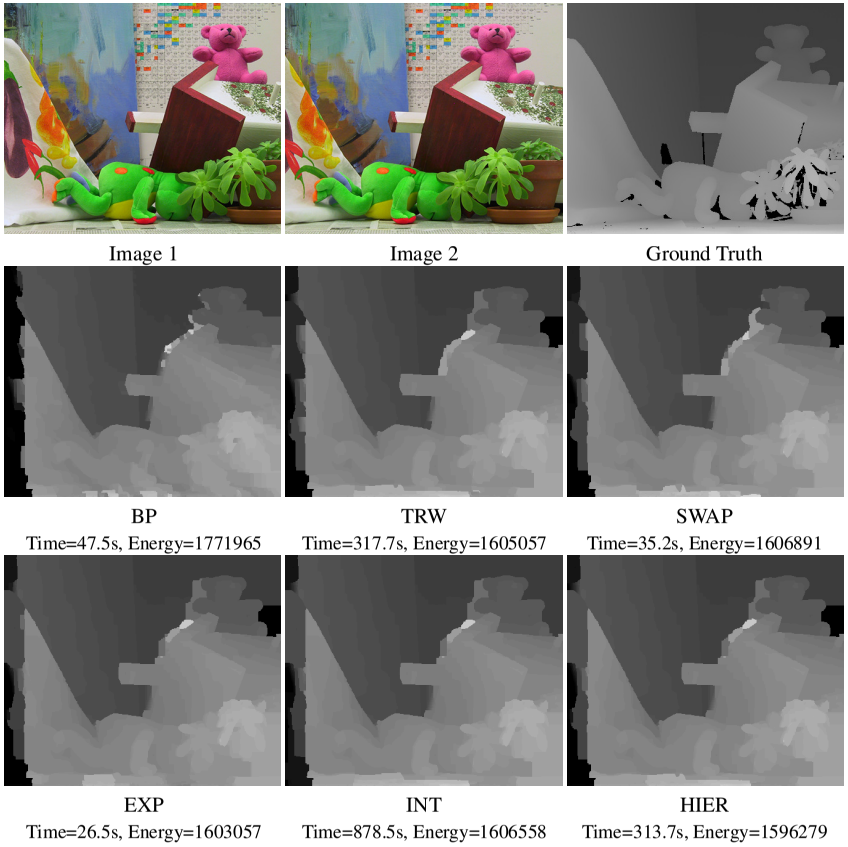


Figure 7: Results for the 'teddy' image pair with truncated linear pairwise potentials.

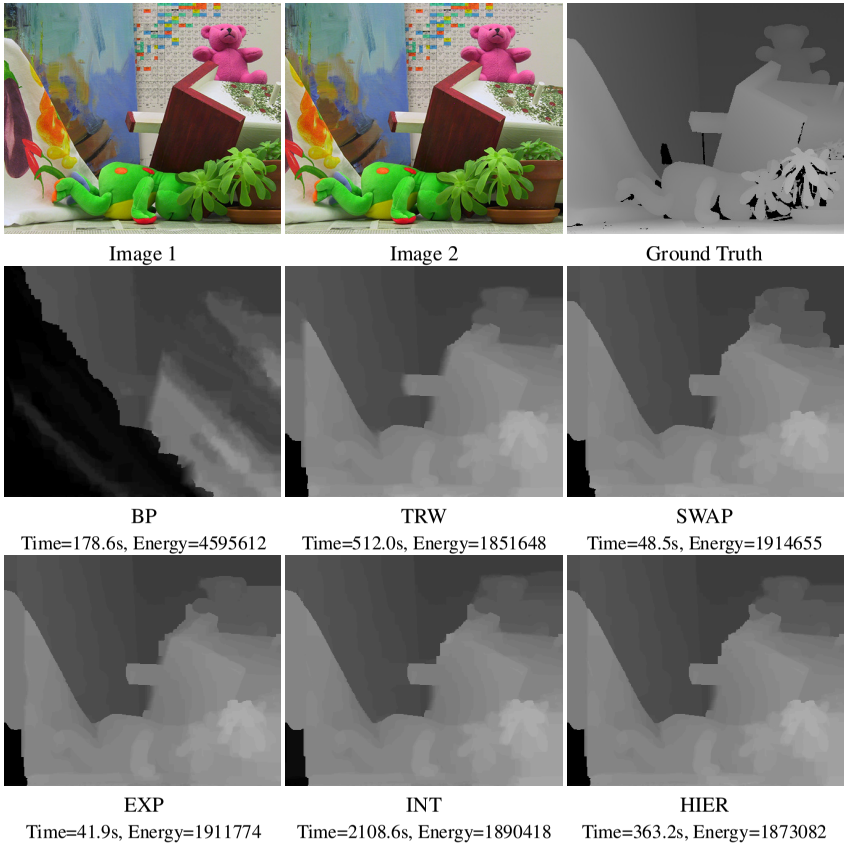


Figure 8: Results for the 'teddy' image pair with truncated quadratic pairwise potentials.

will not only expand our theoretical understanding, but also result in the development of efficient and accurate algorithms.

Acknowledgments

This work is partially funded by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013)/ERC Grant agreement number 259112.

Appendix A. Proof of Theorem 1

We first establish the theoretical property of the complete move-making algorithm using the following lemma.

Lemma 10 *The tight multiplicative bound of the complete move-making algorithm is equal to the submodular distortion of the distance function.*

Proof The submodular distortion of a distance function d is obtained by computing its tightest submodular overestimation as follows:

$$\begin{aligned} \overline{d} = \quad & \operatorname*{argmin}_{d'} t \\ \text{s.t.} \quad & d'(l_i,l_j) \leq t d(l_i,l_j), \forall l_i,l_j \in \mathbf{L}, \\ & d'(l_i,l_j) \geq d(l_i,l_j), \forall l_i,l_j \in \mathbf{L}, \\ & d'(l_i,l_j) + d'(l_{i+1},l_{j+1}) \leq d'(l_i,l_{j+1}) + d'(l_{i+1},l_j), \forall l_i,l_j \in \mathbf{L}\backslash\{l_h\}. \end{aligned} \tag{7}$$

In order to prove the theorem, it is important to note that the definition of submodular distance function implies the following:

$$\overline{d}(l_i,l_j) + \overline{d}(l_{i'},l_{j'}) \leq \overline{d}(l_i,l_{j'}) + \overline{d}(l_{i'},l_j), \forall i' > i, j' > j.$$

A simple proof for the above claim can be found in (Flach and Schlesinger, 2006).

We denote the submodular distortion of d by B. By definition, it follows that

$$d(l_i,l_j) \leq \overline{d}(l_i,l_j) \leq B d(l_i,l_j), \forall l_i,l_j \in \mathbf{L}. \tag{8}$$

We denote an optimal labeling of the original semi-metric labeling problem as $\mathbf{x}^*$, that is,

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x} \in \mathbf{L}^n} \sum_{X_a \in \mathbf{X}} \theta_a(x_a) + \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} d(x_a,x_b). \tag{9}$$

As the semi-metric labeling problem is NP-hard, an optimal labeling $\mathbf{x}^*$ cannot be computed efficiently using any known algorithm. In order to obtain an approximate solution $\hat{\mathbf{x}}$, the complete move-making algorithm replaces the original distance function d by its submodular overestimation $\overline{d}$, that is,

$$\hat{\mathbf{x}} = \operatorname*{argmin}_{\mathbf{x} \in \mathbf{L}^n} \sum_{X_a \in \mathbf{X}} \theta_a(x_a) + \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} \overline{d}(x_a,x_b). \tag{10}$$

Since the pairwise potentials in the above problem are submodular, the approximate solution $\hat{\mathbf{x}}$ can be obtained by solving a single minimum st-cut problem using the method of Flach and Schlesinger (2006). Using inequality (8), it follows that

$$\begin{aligned} & \sum_{X_a \in \mathbf{X}} \theta_a(\hat{x}_a) + \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} d(\hat{x}_a,\hat{x}_b) \\ \leq & \sum_{X_a \in \mathbf{X}} \theta_a(\hat{x}_a) + \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} \overline{d}(\hat{x}_a,\hat{x}_b) \\ \leq & \sum_{X_a \in \mathbf{X}} \theta_a(x^*_a) + \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} \overline{d}(x^*_a,x^*_b) \\ \leq & \sum_{X_a \in \mathbf{X}} \theta_a(x^*_a) + B \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} d(x^*_a,x^*_b). \end{aligned}$$

The above inequality proves that the multiplicative bound of the complete move-making algorithm is at most B. In order to prove that it is exactly equal to B, we need to construct an example for which the bound is tight. To this end, let l_k and $l_{k'}$ be two labels in the set $\mathbf{L}$ such that $k < k'$ and

$$\frac{\overline{d}(l_k,l_{k'})}{d(l_k,l_{k'})} = B.$$

Since B is the minimum possible value of the maximum ratio of the estimated distance $\overline{d}$ to the original distance d, such a pair of labels must exist (otherwise, the submodular distortion can be reduced further). Let us assume that there exists an $l_j \in \mathbf{L}$ such that $j < k$. Other cases (where $j > k'$ or $k < j < k'$) can be handled similarly. Note that since $\overline{d}$ is submodular, it follows that

$$\overline{d}(l_k,l_j) + \overline{d}(l_j,l_{k'}) \geq \overline{d}(l_k,l_{k'}). \tag{11}$$

We define a semi-metric labeling problem over two random variables X_a and X_b connected by an edge with weight $w_{ab} = 1$. The unary potentials are defined as follows:

$$\theta_a(i) = \begin{cases} 0, & \text{if } i = k, \\ \frac{\overline{d}(l_k,l_{k'}) + \overline{d}(l_k,l_j) - \overline{d}(l_j,l_{k'})}{2}, & \text{if } i = j, \\ \infty & \text{otherwise,} \end{cases}$$

$$\theta_b(i) = \begin{cases} 0, & \text{if } i = k', \\ \frac{\overline{d}(l_k,l_{k'}) - \overline{d}(l_k,l_j) + \overline{d}(l_j,l_{k'})}{2}, & \text{if } i = j, \\ \infty & \text{otherwise.} \end{cases}$$

For the above semi-metric labeling problem, it can be verified that an optimal solution $\mathbf{x}^*$ of problem (9) is the following: $x^*_a = l_k$ and $x^*_b = l_{k'}$. Furthermore, using inequality (11), it can be shown that the following is an optimal solution of problem (10): $\hat{x}_a = l_j$ and $\hat{x}_b = l_j$. In other words, $\hat{\mathbf{x}}$ is a valid approximate labeling provided by the complete move-making algorithm. The labelings $\mathbf{x}^*$ and $\hat{\mathbf{x}}$ satisfy the following equality:

$$\sum_{X_a \in \mathbf{X}} \theta_a(\hat{x}_a) + \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} d(\hat{x}_a,\hat{x}_b) = \sum_{X_a \in \mathbf{X}} \theta_a(x^*_a) + B \sum_{(X_a,X_b) \in \mathbf{E}} w_{ab} d(x^*_a,x^*_b).$$

Therefore, the tight multiplicative bound of the complete move-making algorithm is exactly equal to the submodular distortion of the distance function d. ■

We now turn our attention to the complete rounding procedure for the LP relaxation. Before we can establish its tight approximation factor, we need to compute the expected distance between the labels assigned to a pair of neighboring random variables. Recall that, in our notation, we denote a feasible solution of the LP relaxation by $\mathbf{y}$. For any feasible solution $\mathbf{y}$, we define $\mathbf{y}_a$ as the vector whose elements are the unary variables of $\mathbf{y}$ for the random variable $X_a \in \mathbf{X}$, that is,

$$\mathbf{y}_a = [y_a(i), \forall l_i \in \mathbf{L}]. \tag{12}$$

Similarly, we define $\mathbf{y}_{ab}$ as the vector whose elements are the pairwise variables of $\mathbf{y}$ for the neighboring random variables $(X_a, X_b) \in \mathbf{E}$, that is,

$$\mathbf{y}_{ab} = [y_{ab}(i,j), \forall l_i, l_j \in \mathbf{L}]. \tag{13}$$

Furthermore, using $\mathbf{y}_a$, we define $\mathbf{Y}_a$ as

$$Y_a(i) = \sum_{j=1}^{i} y_a(j).$$

In other words, if $\mathbf{y}_a$ is interpreted as the probability distribution over the labels of X_a, then $\mathbf{Y}_a$ is the corresponding cumulative distribution.

Given a feasible solution $\mathbf{y}$, we denote the integer solution obtained using the complete rounding procedure as $\hat{\mathbf{y}}$. The distance between the two labels encoded by vectors $\hat{\mathbf{y}}_a$ and $\hat{\mathbf{y}}_b$ will be denoted by $\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)$. In other words, if $\hat{f}_a$ and $\hat{f}_b$ are the indices of the labels assigned to X_a and X_b (that is, $\hat{y}_a(\hat{f}_a) = 1$ and $\hat{y}_b(\hat{f}_b) = 1$), then $\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b) = d(l_{\hat{f}_a}, l_{\hat{f}_b})$.

The following shorthand notation would be useful for our analysis.

$$D_1(i) = \frac{1}{2}\left(d(l_i, l_1) + d(l_i, l_h) - d(l_{i+1}, l_1) - d(l_{i+1}, l_h)\right), \forall i \in \{1, \cdots, h-1\}, \tag{14}$$
$$D_2(i,j) = \frac{1}{2}\left(d(l_i, l_{j+1}) + d(l_{i+1}, l_j) - d(l_i, l_j) - d(l_{i+1}, l_{j+1})\right), \forall i, j \in \{1, \cdots, h-1\}.$$

Using the above notation, we can state the following lemma on the expected distance of the rounded solution for two neighboring random variables.

Lemma 11 *Let $\mathbf{y}$ be a feasible solution of the LP relaxation, $\mathbf{Y}_a$ and $\mathbf{Y}_b$ be cumulative distributions of $\mathbf{y}_a$ and $\mathbf{y}_b$, and $\hat{\mathbf{y}}$ be the integer solution obtained by the complete rounding procedure for $\mathbf{y}$. Then, the following equation holds true:*

$$\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)) = \sum_{i=1}^{h-1} Y_a(i) D_1(i) + \sum_{j=1}^{h-1} Y_b(j) D_1(j) + \sum_{i=1}^{h-1}\sum_{j=1}^{h-1} |Y_a(i) - Y_b(j)| D_2(i,j).$$

Proof We define $\hat{f}_a$ and $\hat{f}_b$ to be the indices of the labels assigned to X_a and X_b by the rounded integer solution $\hat{\mathbf{y}}$. In other words, $\hat{y}_a(i) = 1$ if and only if $i = \hat{f}_a$ and $\hat{y}_b(j) = 1$ if and only if $j = \hat{f}_b$. We define binary variables $z_a(i)$ and $z_b(j)$ as follows:

$$z_a(i) = \begin{cases} 1 & \text{if } i \leq \hat{f}_a, \\ 0 & \text{otherwise,} \end{cases} \quad z_b(j) = \begin{cases} 1 & \text{if } j \leq \hat{f}_b, \\ 0 & \text{otherwise.} \end{cases}$$

For complete rounding, it can be verified that

$$\mathbb{E}(z_a(i)) = Y_a(i). \tag{15}$$

Furthermore, we also define binary variables $z_{ab}(i,j)$ that indicate whether i and j are contained within the interval defined by $\hat{f}_a$ and $\hat{f}_b$. Formally,

$$z_{ab}(i,j) = \begin{cases} 1 & \text{if } \min\{i,j\} \geq \min\{\hat{f}_a, \hat{f}_b\} \text{ and } \max\{i,j\} < \max\{\hat{f}_a, \hat{f}_b\}, \\ 0 & \text{otherwise.} \end{cases}$$

For complete rounding, it can be verified that

$$\mathbb{E}(z_{ab}(i,j)) = |Y_a(i) - Y_b(j)|. \tag{16}$$

Using the result of Flach and Schlesinger (2006), we know that

$$\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b) = \sum_{i=1}^{h-1} z_a(i) D_1(i) + \sum_{j=1}^{h-1} z_b(j) D_1(j) + \sum_{i=1}^{h-1}\sum_{j=1}^{h-1} z_{ab}(i,j) D_2(i,j).$$

The proof of the lemma follows by taking the expectation of the LHS and the RHS of the above equation and simplifying the RHS using the linearity of expectation and equations (15) and (16). ■

In order to state the next lemma, we require the definition of uncrossing pairwise variables. Given unary variables $\mathbf{y}_a$ and $\mathbf{y}_b$, the pairwise variable vector $\mathbf{y}'_{ab}$ is called uncrossing with respect to $\mathbf{y}_a$ and $\mathbf{y}_b$ if it satisfies the following properties:

$$\begin{aligned}
&\sum_{j=1}^{h} y'_{ab}(i,j) = y_a(i), \forall i \in \{1, 2, \cdots, h\}, \\
&\sum_{i=1}^{h} y'_{ab}(i,j) = y_b(i), \forall j \in \{1, 2, \cdots, h\}, \\
&y'_{ab}(i,j) \geq 0, \forall i, j \in \{1, 2, \cdots, h\}, \\
&\min\{y'_{ab}(i,j'), y'_{ab}(i',j)\} = 0, \forall i, j, i', j' \in \{1, 2, \cdots, h\}, i < i', j < j'.
\end{aligned} \tag{17}$$

The following lemma establishes a connection between the expected distance between the labels assigned by complete rounding and the pairwise cost specified by uncrossing pairwise variables.

Lemma 12 *Let $\mathbf{y}$ be a feasible solution of the LP relaxation, and $\hat{\mathbf{y}}$ be the integer solution obtained by the complete rounding procedure for $\mathbf{y}$. Furthermore, let $\mathbf{y}'_{ab}$ be uncrossing pairwise variables with respect to $\mathbf{y}_a$ and $\mathbf{y}_b$. Then, the following equation holds true:*

$$\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)) = \sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i, l_j) y'_{ab}(i,j).$$

Proof We define $\mathbf{Y}_a$ and $\mathbf{Y}_b$ to be the cumulative distributions corresponding to $\mathbf{y}_a$ and $\mathbf{y}_b$ respectively. We claim that the uncrossing property (17) implies the following condition:

$$\sum_{i=1}^{i'}\sum_{j=1}^{j'} y'_{ab}(i,j) = \min\{Y_a(i'), Y_b(j')\}, \forall i', j' \in \{1, \cdots, h\}. \tag{18}$$

To prove this claim, assume that $Y_a(i') < Y_b(j')$. The other cases can be handled similarly. Since $\mathbf{y}'_{ab}$ satisfies the constraints of the LP relaxation, it follows that:

$$\begin{aligned}\sum_{i=1}^{h}\sum_{j=1}^{j'} y'_{ab}(i,j) &= Y_b(j'),\\ \sum_{i=1}^{i'}\sum_{j=1}^{h} y'_{ab}(i,j) &= Y_a(i').\end{aligned} \tag{19}$$

Since the LHS of equality (18) is less than or equal to the LHS of both the above equations, it follows that

$$\sum_{i=1}^{i'}\sum_{j=1}^{j'} y'_{ab}(i,j) \leq \min\{Y_a(i'), Y_b(j')\}. \tag{20}$$

Therefore, there must exist a $k > i'$ and $k' \leq j'$ such that $y'_{ab}(k,k') \neq 0$. Otherwise, the LHS in the above inequality will be exactly equal to $Y_b(j')$, which would result in a contradiction. By the uncrossing property (17), we know that $\min\{y'_{ab}(i,j), y'_{ab}(k,k')\} = 0$ if $i \leq i'$ and $j > j'$. Therefore, $y'_{ab}(i,j) = 0$ for all $i \leq i'$ and $j > j'$, which proves the claim.

Combining equations (19) and (20), we get the following:

$$\sum_{i=1}^{i'}\sum_{j=j'+1}^{h} y'_{ab}(i,j) + \sum_{i=i'+1}^{h}\sum_{j=1}^{j'} y'_{ab}(i,j) = |Y_a(i) - Y_b(j)|, \forall i', j' \in \{1, \cdots, h\}.$$

By solving for $\mathbf{y}'_{ab}$ using the above equations, we get

$$\sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i, l_j) y'_{ab}(i,j) = \sum_{i=1}^{h-1} Y_a(i) D_1(i) + \sum_{j=1}^{h-1} Y_b(j) D_1(j) + \sum_{i=1}^{h-1}\sum_{j=1}^{h-1} |Y_a(i) - Y_b(j)| D_2(i,j).$$

Using the previous lemma, this proves that

$$\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)) = \sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i, l_j) y'_{ab}(i,j).$$

■

Our next lemma establishes that uncrossing pairwise variables are optimal for submodular distance functions.

Lemma 13 *Let $\mathbf{y}'_{ab}$ be the uncrossing pairwise variables with respect to the unary variables $\mathbf{y}_a$ and $\mathbf{y}_b$. Let $\overline{d} : \mathbf{L} \times \mathbf{L} \to \mathbb{R}^+$ be a submodular distance function. Then the following condition holds true:*

$$\begin{aligned}\mathbf{y}'_{ab} = \quad & \underset{\overline{\mathbf{y}}_{ab}}{\operatorname{argmin}} \sum_{i=1}^{h}\sum_{j=1}^{h} \overline{d}(i,j)\overline{y}_{ab}(i,j),\\ \textit{s.t.} \quad & \sum_{j=1}^{h} \overline{y}_{ab}(i,j) = y_a(i), \forall i \in \{1, \cdots, h\},\\ & \sum_{i=1}^{h} \overline{y}_{ab}(i,j) = y_b(j), \forall j \in \{1, \cdots, h\},\\ & \overline{y}_{ab}(i,j) \geq 0, \forall i, j \in \{1, \cdots, h\}.\end{aligned} \tag{21}$$

Proof We prove the lemma by contradiction. Suppose that the optimal solution to the above problem is $\mathbf{y}''_{ab}$, which is not uncrossing. Let

$$\min\{y''_{ab}(i,j'), y''_{ab}(i',j)\} = \lambda \neq 0,$$

where $i < i'$ and $j < j'$. Since $\overline{d}$ is submodular, it implies that

$$\overline{d}(l_i, l_j) + \overline{d}(l_{i'}, l_{j'}) \leq \overline{d}(l_i, l_{j'}) + \overline{d}(l_{i'}, l_j).$$

Therefore the objective function of problem (21) can be reduced further by the following modification:

$$\begin{aligned}y''_{ab}(i,j') &\leftarrow y''_{ab}(i,j') - \lambda, y''_{ab}(i',j) \leftarrow y''_{ab}(i',j) - \lambda,\\ y''_{ab}(i,j) &\leftarrow y''_{ab}(i,j) + \lambda, y''_{ab}(i',j') \leftarrow y''_{ab}(i',j') + \lambda.\end{aligned}$$

The resulting contradiction proves our claim that the uncrossing pairwise variables $\mathbf{y}'_{ab}$ are an optimal solution of problem (21). ■

Using the above lemmas, we will now obtain the tight approximation factor of the complete rounding procedure.

Lemma 14 *The tight approximation factor of the complete rounding procedure is equal to the submodular distortion of the distance function.*

Proof We denote a feasible fractional solution of the LP relaxation by $\mathbf{y}$ and the rounded solution by $\hat{\mathbf{y}}$. Consider a pair of neighboring random variables $(X_a, X_b) \in \mathbf{X}$. We define uncrossing pairwise variables $\mathbf{y}'_{ab}$ with respect to $\mathbf{y}_a$ and $\mathbf{y}_b$. Using Lemmas 12 and 13, the approximation factor of the

complete rounding procedure can be shown to be at most B as follows:

$$\begin{aligned}\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a,\hat{\mathbf{y}}_b)) = & \sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i,l_j)y'_{ab}(i,j)\\ \leq & \sum_{i=1}^{h}\sum_{j=1}^{h} \overline{d}(l_i,l_j)y'_{ab}(i,j)\\ \leq & \sum_{i=1}^{h}\sum_{j=1}^{h} \overline{d}(l_i,l_j)y_{ab}(i,j)\\ \leq & B\sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i,l_j)y_{ab}(i,j).\end{aligned}$$

In order to prove that the approximation factor of the complete rounding is exactly B, we need an example where the above inequality holds as an equality. The key to obtaining a tight example lies in the Lagrangian dual of problem (7). In order to specify its dual, we need three types of dual variables. The first type, denoted by $\alpha(i,j)$, corresponds to the constraint

$$d'(l_i,l_j) \leq t d(l_i,l_j).$$

The second type, denoted by $\beta(i,j)$, corresponds to the constraint

$$d'(l_i,l_j) \geq d(l_i,l_j).$$

The third type, denoted by $\gamma(i,j)$, corresponds to the constraint

$$d'(l_i,l_j) + d'(l_{i+1},l_{j+1}) \leq d'(l_i,l_{j+1}) + d'(l_{i+1},l_j).$$

Using the above variables, the dual of problem (7) is given by

$$\begin{aligned}\max \quad & \sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i,l_j)\beta(i,j) \qquad (22)\\ \text{s.t.} \quad & \sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i,l_j)\alpha(i,j) = 1,\\ & \beta(i,j) = \alpha(i,j) - \gamma(i,j-1) - \gamma(i-1,j) + \gamma(i-1,j-1) + \gamma(i,j),\\ & \forall i,j \in \{1,\cdots,h\},\\ & \alpha(i,j) \geq 0, \beta(i,j) \geq 0, \gamma(i,j) \geq 0, \forall i,j \in \{1,\cdots,h\}.\end{aligned}$$

We claim that the above dual problem has an optimal solution $(\boldsymbol{\alpha}^*,\boldsymbol{\beta}^*,\boldsymbol{\gamma}^*)$ that satisfies the following property:

$$\min\{\beta^*(i,j'),\beta^*(i',j)\} = 0, \forall i,i',j,j' \in \{1,\cdots,h\}, i<i', j<j'. \qquad (23)$$

We refer to the optimal dual solution $\boldsymbol{\beta}^*$ that satisfies the above property as uncrossing dual variables as it is analogous to uncrossing pairwise variables. The above claim, namely, the existence of an uncrossing optimal dual solution, can be proved by contradiction as follows. Suppose there exists no

optimal solution that satisfies the above property. Then consider the following problem, which is the dual of the problem of finding the tightest submodular overestimate of the submodular function $\overline{d}$:

$$\begin{aligned}\max \quad & \sum_{i=1}^{h}\sum_{j=1}^{h} \overline{d}(l_i,l_j)\beta(i,j) \qquad (24)\\ \text{s.t.} \quad & \sum_{i=1}^{h}\sum_{j=1}^{h} \overline{d}(l_i,l_j)\alpha(i,j) = 1,\\ & \beta(i,j) = \alpha(i,j) - \gamma(i,j-1) - \gamma(i-1,j) + \gamma(i-1,j-1) + \gamma(i,j),\\ & \forall i,j \in \{1,\cdots,h\},\\ & \alpha(i,j) \geq 0, \beta(i,j) \geq 0, \gamma(i,j) \geq 0, \forall i,j \in \{1,\cdots,h\}.\end{aligned}$$

By strong duality, problem (24) has an optimal value of 1. However, the optimal solution of problem (22), which is also a feasible solution for problem (24), provides a value strictly greater than 1. This results in a contradiction that proves our claim.

The optimal dual variables that satisfy property (23) allow us to construct an example that proves that the approximation factor B of the complete rounding procedure is tight. Specifically, we define

$$\begin{aligned}&\overline{y}_{ab}(i,j) = \frac{\alpha^*(i,j)}{\sum_{i'=1}^{h}\sum_{j'=1}^{h}\alpha^*(i',j')}, \forall i,j \in \{1,\cdots,h\},\\ &\overline{y}_a(i) = \sum_{j=1}^{h} \overline{y}_{ab}(i,j), \forall i \in \{1,\cdots,h\},\\ &\overline{y}_b(j) = \sum_{i=1}^{h} \overline{y}_{ab}(i,j), \forall j \in \{1,\cdots,h\}.\end{aligned}$$

Note that the pairwise variables $\overline{\mathbf{y}}_{ab}$ must minimize the pairwise potential corresponding to the unary variables $\mathbf{y}_a$ and $\mathbf{y}_b$, that is,

$$\begin{aligned}\overline{\mathbf{y}}_{ab} = & \operatorname*{argmin}_{\mathbf{y}_{ab}} \sum_{l_i,l_j \in \mathbf{L}} d(l_i,l_j)y_{ab}(i,j) \qquad (25)\\ \text{s.t.} \quad & \sum_{l_j \in \mathbf{L}} y_{ab}(i,j) = \overline{y}_a(i), \forall l_i \in \mathbf{L}\\ & \sum_{l_i \in \mathbf{L}} y_{ab}(i,j) = \overline{y}_b(j), \forall l_j \in \mathbf{L}\\ & y_{ab}(i,j) \geq 0, \forall l_i,l_j \in \mathbf{L}.\end{aligned}$$

If the above statement was not true, then the value of the dual problem (22) could be increased further.

We also define the following pairwise variables:

$$y'_{ab}(i,j) = \frac{\beta^*(i,j)}{\sum_{i'=1}^{h}\sum_{j'=1}^{h}\beta^*(i',j')}, \forall i,j \in \{1,\cdots,h\},$$
$$y'_a(i) = \sum_{j=1}^{h} y'_{ab}(i,j), \forall i \in \{1,\cdots,h\},$$
$$y'_b(j) = \sum_{i=1}^{h} y'_{ab}(i,j), \forall j \in \{1,\cdots,h\}.$$

It can be verified that

$$y'_a(i) = \overline{y}_a(i), y'_b(j) = \overline{y}_b(j), \forall i,j \in \{1,\cdots,h\}.$$

The above condition follows from the constraints of problem (22). Due to the uncrossing property of $\boldsymbol{\beta}^*$, the pairwise variables $\mathbf{y'_{ab}}$ are uncrossing with respect to $\overline{\mathbf{y}}_a$ and $\overline{\mathbf{y}}_b$. By Lemma 12, this implies that

$$\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)) = \sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i,l_j) y'_{ab}(i,j),$$

where $\hat{\mathbf{y}}_a$ and $\hat{\mathbf{y}}_b$ are integer solutions obtained by the complete rounding procedure. By strong duality, it follows that

$$\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)) = B\sum_{i=1}^{h}\sum_{j=1}^{h} d(l_i,l_j)\overline{y}_{ab}(i,j). \tag{26}$$

The existence of an example that satisfies properties (25) and (26) implies that the tight approximation factor of the complete rounding procedure is B. ■

Lemmas 10 and 14 together prove Theorem 1.

Appendix B. Proof of Theorem 3

We begin by establishing the theoretical properties of the interval-move making algorithm. Recall that, given an interval of labels $\mathbf{I} = \{l_s, \cdots, l_e\}$ of at most q consecutive labels and a labeling $\hat{\mathbf{x}}$, we define $\mathbf{I}_a = \mathbf{I} \cup \{\hat{x}_a\}$ for all random variables $X_a \in \mathbf{X}$. In order to use the interval move-making algorithm, we compute a submodular distance function $\overline{d}_{\hat{x}_a,\hat{x}_b} : \mathbf{I}_a \times \mathbf{I}_b \rightarrow \mathbb{R}^+$ for all pairs of neighboring random variables $(X_a, X_b) \in \mathbf{E}$ as follows:

$$\begin{aligned}
\overline{d}_{\hat{x}_a,\hat{x}_b} = \quad & \operatorname*{argmin}_{d'} t \qquad (27)\\
\text{s.t.} \quad & d'(l_i,l_j) \leq t d(l_i,l_j), \forall l_i \in \mathbf{I}_a, l_j \in \mathbf{I}_b,\\
& d'(l_i,l_j) \geq d(l_i,l_j), \forall l_i \in \mathbf{I}_a, l_j \in \mathbf{I}_b,\\
& d'(l_i,l_j) + d'(l_{i+1},l_{j+1}) \leq d'(l_i,l_{j+1}) + d'(l_{i+1},l_j), \forall l_i,l_j \in \mathbf{I}\backslash\{l_e\},\\
& d'(l_i,l_e) + d'(l_{i+1},\hat{x}_b) \leq d'(l_i,\hat{x}_b) + d'(l_{i+1},l_e), \forall l_i \in \mathbf{I}\backslash\{l_e\},\\
& d'(l_e,l_j) + d'(\hat{x}_a,l_{j+1}) \leq d'(l_e,l_{j+1}) + d'(\hat{x}_a,l_j), \forall l_j \in \mathbf{I}\backslash\{l_e\},\\
& d'(l_e,l_e) + d(\hat{x}_a,\hat{x}_b) \leq d'(l_e,\hat{x}_b) + d'(\hat{x}_a,l_e).
\end{aligned}$$

For any interval $\mathbf{I}$ and labeling $\mathbf{x}$, we define the following sets:

$$\begin{aligned}
&\mathbf{V}(\mathbf{x},\mathbf{I}) = \{X_a | X_a \in \mathbf{X}, x_a \in \mathbf{I}\},\\
&\mathbf{A}(\mathbf{x},\mathbf{I}) = \{(X_a,X_b) | (X_a,X_b) \in \mathbf{E}, x_a \in \mathbf{I}, x_b \in \mathbf{I}\},\\
&\mathbf{B}_1(\mathbf{x},\mathbf{I}) = \{(X_a,X_b) | (X_a,X_b) \in \mathbf{E}, x_a \in \mathbf{I}, x_b \notin \mathbf{I}\},\\
&\mathbf{B}_2(\mathbf{x},\mathbf{I}) = \{(X_a,X_b) | (X_a,X_b) \in \mathbf{E}, x_a \notin \mathbf{I}, x_b \in \mathbf{I}\},\\
&\mathbf{B}(\mathbf{x},\mathbf{I}) = \mathbf{B}_1(\mathbf{x}) \cup \mathbf{B}_2(\mathbf{x}).
\end{aligned}$$

In other words, $\mathbf{V}(\mathbf{x},\mathbf{I})$ is the set of all random variables whose label belongs to the interval $\mathbf{I}$. Similarly, $\mathbf{A}(\mathbf{x},\mathbf{I})$ is the set of all neighboring random variables such that the labels assigned to both the random variables belong to the interval $\mathbf{I}$. The set $\mathbf{B}(\mathbf{x},\mathbf{I})$ contains the set of all neighboring random variables such that only one of the two labels assigned to the two random variables belongs to the interval $\mathbf{I}$. Given the set of all intervals $\mathcal{I}$ and a labeling $\hat{\mathbf{x}}$, we define the following for all $x_a, x_b \in \mathbf{L}$:

$$\begin{aligned}
D(x_a,x_b;\hat{x}_a,\hat{x}_b) = \quad & \sum_{\mathbf{I}\in\mathcal{I},\mathbf{A}(\mathbf{x},\mathbf{I})\ni(X_a,X_b)} d_{\hat{x}_a,\hat{x}_b}(x_a,x_b)\\
& + \sum_{\mathbf{I}\in\mathcal{I},\mathbf{B}_1(\mathbf{x},\mathbf{I})\ni(X_a,X_b)} d_{\hat{x}_a,\hat{x}_b}(x_a,\hat{x}_b)\\
& + \sum_{\mathbf{I}\in\mathcal{I},\mathbf{B}_2(\mathbf{x},\mathbf{I})\ni(X_a,X_b)} d_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,x_b).
\end{aligned}$$

Using the above notation, we are ready to state the following lemma on the theoretical guarantee of the interval move-making algorithm.

Lemma 15 *The tight multiplicative bound of the interval move-making algorithm is equal to*

$$\frac{1}{q}\max_{x_a,x_b,\hat{x}_a,\hat{x}_b\in\mathbf{L},x_a\neq x_b}\frac{D(x_a,x_b;\hat{x}_a,\hat{x}_b)}{d(x_a,x_b)}.$$

Proof We denote an optimal labeling by $\mathbf{x}^*$ and the estimated labeling by $\hat{\mathbf{x}}$. Let $t \in [1,q]$ be a uniformly distributed random integer. Using t, we define the following set of non-overlapping intervals:

$$\mathcal{I}_t = \{[1,t],[t+1,t+q],\cdots,[.,h]\}.$$

For each interval $\mathbf{I} \in \mathcal{I}_t$, we define a labeling $\mathbf{x}^{\mathbf{I}}$ as follows:

$$x_a^{\mathbf{I}} = \begin{cases} x_a^* & \text{if } x_a^* \in \mathbf{I},\\ \hat{x}_a & \text{otherwise.}\end{cases}$$

Since $\hat{\mathbf{x}}$ is the labeling obtained after the interval move-making algorithm converges, it follows that

$$\sum_{X_a\in\mathbf{X}}\theta_a(\hat{x}_a) + \sum_{(X_a,X_b)\in\mathbf{E}} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,\hat{x}_b) \leq \sum_{X_a\in\mathbf{X}}\theta_a(x_a^{\mathbf{I}}) + \sum_{(X_a,X_b)\in\mathbf{E}} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(x_a^{\mathbf{I}},x_b^{\mathbf{I}}).$$

By canceling out the common terms, we can simplify the above inequality as

$$\begin{aligned}
& \sum_{X_a \in \mathbf{V}(\mathbf{x}^*,\mathbf{I})} \theta_a(\hat{x}_a) \\
& + \sum_{(X_a,X_b)\in\mathbf{A}(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,\hat{x}_b) \\
& + \sum_{(X_a,X_b)\in\mathbf{B}_1(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,\hat{x}_b) \\
& + \sum_{(X_a,X_b)\in\mathbf{B}_2(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,\hat{x}_b) \\
\leq & \sum_{X_a \in \mathbf{V}(\mathbf{x}^*,\mathbf{I})} \theta_a(x^*_a) \\
& + \sum_{(X_a,X_b)\in\mathbf{A}(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(x^*_a,x^*_b) \\
& + \sum_{(X_a,X_b)\in\mathbf{B}_1(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(x^*_a,\hat{x}_b) \\
& + \sum_{(X_a,X_b)\in\mathbf{B}_2(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,x^*_b).
\end{aligned}$$

We now sum the above inequality over all the intervals $\mathbf{I} \in \mathcal{I}_t$. Note that the resulting LHS is at least equal to the energy of the labeling $\hat{\mathbf{x}}$ when the distance function between the random variables (X_a, X_b) is $\overline{d}_{\hat{x}_a,\hat{x}_b}$. This implies that

$$\begin{aligned}
& \sum_{X_a\in\mathbf{X}} \theta_a(\hat{x}_a) + \sum_{(X_a,X_b)\in\mathbf{E}} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,\hat{x}_b) \\
\leq & \sum_{X_a\in\mathbf{X}} \theta_a(x^*_a) \\
& + \sum_{\mathbf{I}\in\mathcal{I}_t} \sum_{(X_a,X_b)\in\mathbf{A}(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(x^*_a,x^*_b) \\
& + \sum_{\mathbf{I}\in\mathcal{I}_t} \sum_{(X_a,X_b)\in\mathbf{B}_1(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(x^*_a,\hat{x}_b) \\
& + \sum_{\mathbf{I}\in\mathcal{I}_t} \sum_{(X_a,X_b)\in\mathbf{B}_2(\mathbf{x}^*,\mathbf{I})} w_{ab}\overline{d}_{\hat{x}_a,\hat{x}_b}(\hat{x}_a,x^*_b).
\end{aligned}$$

Taking the expectation on both sides of the above inequality with respect to the uniformly distributed random integer $t \in [1, q]$ proves that the multiplicative bound of the interval move-making algorithm is at most equal to

$$\frac{1}{q} \max_{x_a,x_b,\hat{x}_a,\hat{x}_b\in\mathbf{L},x_a\neq x_b} \frac{D(x_a,x_b;\hat{x}_a,\hat{x}_b)}{d(x_a,x_b)}.$$

A tight example with two random variables X_a and X_b with $w_{ab} = 1$ can be constructed similar to the one shown in Lemma 10. ■

We now turn our attention to the interval rounding procedure. Let $\mathbf{y}$ be a feasible solution of the LP relaxation, and $\hat{\mathbf{y}}$ be the integer solution obtained using interval rounding. Once again, we define the unary variable vector $\mathbf{y}_a$ and the pairwise variable vector $\mathbf{y}_{ab}$ as specified in equations (12) and (13) respectively. Similar to the previous appendix, we denote the expected distance between $\hat{\mathbf{y}}_a$ and $\hat{\mathbf{y}}_b$ as $\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)$.

Given an interval $\mathbf{I} = \{l_s, \cdots, l_e\}$ of at most q consecutive labels, we define a vector $\mathbf{Y}^{\mathbf{I}}_a$ for each random variable X_a as follows:

$$Y^{\mathbf{I}}_a(i) = \sum_{j=s}^{s+i-1} y_a(j), \forall i \in \{1, \cdots, e-s+1\}.$$

In other words, $\mathbf{Y}^{\mathbf{I}}_a$ is the cumulative distribution of $\mathbf{y}_a$ within the interval $\mathbf{I}$. Furthermore, for each pair of neighboring random variables $(X_a, X_b) \in \mathbf{E}$ we define

$$\begin{aligned}
Z^{\mathbf{I}}_{ab} &= \max\{Y^{\mathbf{I}}_a(e-s+1), Y^{\mathbf{I}}_b(e-s+1)\} - \min\{Y^{\mathbf{I}}_a(e-s+1), Y^{\mathbf{I}}_b(e-s+1)\}, \\
Z^{\mathbf{I}}_a(i) &= \min\{Y^{\mathbf{I}}_a(i), Y^{\mathbf{I}}_b(e-s+1)\}, \forall i \in \{1, \cdots, e-s+1\}, \\
Z^{\mathbf{I}}_b(j) &= \min\{Y^{\mathbf{I}}_b(j), Y^{\mathbf{I}}_a(e-s+1)\}, \forall j \in \{1, \cdots, e-s+1\}.
\end{aligned}$$

The following shorthand notation would be useful to concisely specify the exact form of $\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)$.

$$\begin{aligned}
D^{\mathbf{I}}_0 = & \max_{l_i,l_j,|\{l_i,l_j\}\cap\mathbf{I}|=1} d(l_i,l_j), \\
D^{\mathbf{I}}_1(i) = & \frac{1}{2}\left(d(l_{s+i-1},l_s) + d(l_{s+i-1},l_e) - d(l_{s+i},l_s) - d(l_{s+i},l_e)\right), \forall i \in \{1,\cdots,e-s\}, \\
D^{\mathbf{I}}_2(i,j) = & \frac{1}{2}\left(d(l_{s+i-1},l_{s+j}) + d(l_{s+i},l_{s+j-1}) - d(l_{s+i-1},l_{s+j-1}) - d(l_{s+i},l_{s+j})\right), \\
& \forall i,j \in \{1,\cdots,e-s\}.
\end{aligned}$$

In other words, $\mathbf{D}^{\mathbf{I}}_0$ is the maximum distance between two labels such that only one of the two labels lies in the interval $\mathbf{I}$. The terms $\mathbf{D}^{\mathbf{I}}_1$ and $\mathbf{D}^{\mathbf{I}}_2$ are analogous to the terms defined in equation (14). Using the above notation, we can state the following lemma on the expected distance of the rounded solution for two neighboring random variables.

Lemma 16 *Let $\mathbf{y}$ be a feasible solution of the LP relaxation, and $\hat{\mathbf{y}}$ be the integer solution obtained by the interval rounding procedure for $\mathbf{y}$ using the set of intervals $\mathcal{I}$. Then, the following inequality holds true:*

$$\begin{aligned}
\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a,\hat{\mathbf{y}}_b)) \leq & \frac{1}{q}\left(\sum_{\mathbf{I}=\{l_s,\cdots,l_e\}\in\mathcal{I}} Z^{\mathbf{I}}_{ab}D^{\mathbf{I}}_0 + \sum_{i=1}^{e-s} Z^{\mathbf{I}}_a(i)D^{\mathbf{I}}_1(i) + \sum_{j=1}^{e-s} Z^{\mathbf{I}}_b(j)D^{\mathbf{I}}_1(j) + \right. \\
& \left. \sum_{i=1}^{e-s}\sum_{j=1}^{e-s} |Z^{\mathbf{I}}_a(i) - Z^{\mathbf{I}}_b(j)| D^{\mathbf{I}}_2(i,j)\right).
\end{aligned}$$

Proof We begin the proof by establishing the probability of a random variable X_a being assigned a label in an iteration of the interval rounding procedure. The total number of intervals in the set

$\mathcal{I}$ is $h+q-1$. Out of all the intervals, each label l_i is present in q intervals. Thus, the probability of choosing an interval that contains the label l_i is $q/(h+q-1)$. Once an interval containing l_i is chosen, the probability of assigning the label l_i to X_a is $y_a(i)$. Thus, the probability of assigning a label l_i to X_a is $y_a(i)q/(h+q-1)$. Summing over all labels l_i, we observe that the probability of assigning a label to X_a in an iteration of the interval rounding procedure is $q/(h+q-1)$.

Now we consider two neighboring random variables $(X_a, X_b) \in \mathbf{E}$. In the current iteration of interval rounding, given an interval $\mathbf{I}$, the probability of exactly one of the two random variables getting assigned a label in $\mathbf{I}$ is exactly equal to $Z^{\mathbf{I}}_{ab}$. In this case, we have to assume that the expected distance between the two variables will be the maximum possible, that is, $D^{\mathbf{I}}_0$. If both the variables are assigned a label in $\mathbf{I}$, then a slight modification of Lemma 11 gives us the expected distance as

$$\sum_{i=1}^{e-s} Z^{\mathbf{I}}_a(i) D^{\mathbf{I}}_1(i) + \sum_{j=1}^{e-s} Z^{\mathbf{I}}_b(j) D^{\mathbf{I}}_1(j) + \sum_{i=1}^{e-s}\sum_{j=1}^{e-s} |Z^{\mathbf{I}}_a(i) - Z^{\mathbf{I}}_b(j)| D^{\mathbf{I}}_2(i,j).$$

Thus, the expected distance between the labels of X_a and X_b conditioned on at least one of the two random variables getting assigned in an iteration is equal to

$$\frac{1}{(h+q-1)} \left(\sum_{\mathbf{I}=\{l_s,\cdots,l_e\}\in\mathcal{I}} Z^{\mathbf{I}}_{ab} D^{\mathbf{I}}_0 + \sum_{i=1}^{e-s} Z^{\mathbf{I}}_a(i) D^{\mathbf{I}}_1(i) + \sum_{j=1}^{e-s} Z^{\mathbf{I}}_b(j) D^{\mathbf{I}}_1(j) + \sum_{i=1}^{e-s}\sum_{j=1}^{e-s} |Z^{\mathbf{I}}_a(i) - Z^{\mathbf{I}}_b(j)| D^{\mathbf{I}}_2(i,j) \right),$$

where the term $1/(h+q-1)$ corresponds to the probability of choosing an interval from the set $\mathcal{I}$. In order to compute $\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)$, we need to divide the above term with the probability of at least one of the two random variables getting assigned a label in the current iteration. Since the two cumulative distributions $\mathbf{Y}^{\mathbf{I}}_a$ and $\mathbf{Y}^{\mathbf{I}}_b$ can be arbitrarily close to each other without being exactly equal, we can only lower bound the probability of at least one of X_a and X_b being assigned a label at the current iteration by the probability of X_a being assigned a label in the current iteration. Since the probability of X_a being assigned a label is $q/(h+q-1)$, it follows that

$$\mathbb{E}(\hat{d}(\hat{\mathbf{y}}_a, \hat{\mathbf{y}}_b)) \leq \frac{1}{q} \left(\sum_{\mathbf{I}=\{l_s,\cdots,l_e\}\in\mathcal{I}} Z^{\mathbf{I}}_{ab} D^{\mathbf{I}}_0 + \sum_{i=1}^{e-s} Z^{\mathbf{I}}_a(i) D^{\mathbf{I}}_1(i) + \sum_{j=1}^{e-s} Z^{\mathbf{I}}_b(j) D^{\mathbf{I}}_1(j) + \sum_{i=1}^{e-s}\sum_{j=1}^{e-s} |Z^{\mathbf{I}}_a(i) - Z^{\mathbf{I}}_b(j)| D^{\mathbf{I}}_2(i,j) \right).$$

■

Using the above lemma, we can now establish the theoretical property of the interval rounding procedure.

Lemma 17 *The tight approximation factor of the interval rounding procedure is equal to*

$$\frac{1}{q} \max_{x_a, x_b, \hat{x}_a, \hat{x}_b \in \mathbf{L}, x_a \neq x_b} \frac{D(x_a, x_b; \hat{x}_a, \hat{x}_b)}{d(x_a, x_b)}.$$

Proof Similar to Lemma 14, the key to proving the above lemma lies in the dual of problem (27). Given the current labels $\hat{x}_a$ and $\hat{x}_b$ of the random variables X_a and X_b respectively, as well as an interval $\mathbf{I} = \{l_s, \cdots, l_e\}$, problem (27) computes the tightest submodular overestimation $\overline{d}_{\hat{x}_a,\hat{x}_b} : \mathbf{I}_a \times \mathbf{I}_b \rightarrow \mathbb{R}^+$ where $\mathbf{I}_a = \mathbf{I} \cup \{\hat{x}_a\}$ and $\mathbf{I}_b = \mathbf{I} \cup \{\hat{x}_b\}$. Similar to problem (22), the dual of problem (27) consists of three types of dual variables. The first type, denoted by $\alpha(i,j;\hat{x}_a,\hat{x}_b,\mathbf{I})$ corresponds to the constraint

$$d'(l_i, l_j) \leq t d(l_i, l_j).$$

The second type, denoted by $\beta(i,j;\hat{x}_b,\hat{x}_b,\mathbf{I})$, corresponds to the constraint

$$d'(l_i, l_j) \geq d(l_i, l_j).$$

The third type, denoted by $\gamma(i,j;\hat{x}_a,\hat{x}_b,\mathbf{I})$, corresponds to the constraint

$$d'(l_i, l_j) + d'(l_{i+1}, l_{j+1}) \leq d'(l_i, l_{j+1}) + d'(l_{i+1}, l_j).$$

Let $(\boldsymbol{\alpha}^*(\hat{x}_a,\hat{x}_b,\mathbf{I}), \boldsymbol{\beta}^*(\hat{x}_a,\hat{x}_b,\mathbf{I}), \boldsymbol{\gamma}^*(\hat{x}_a,\hat{x}_b,\mathbf{I}))$ denote an optimal solution of the dual. A modification of Lemma 14 can be used to show that there must exist an optimal solution such that $\boldsymbol{\beta}^*(\hat{x}_a,\hat{x}_b,\mathbf{I})$ is uncrossing.

We consider the values of $\hat{x}_a$ and $\hat{x}_b$ for which we obtain the tight multiplicative bound of the interval move-making algorithm. For these values of $\hat{x}_a$ and $\hat{x}_b$, we define

$$\overline{y}_{ab}(i,j;\hat{x}_a,\hat{x}_b) = \frac{\sum_{\mathbf{I}\in\mathcal{I}} \alpha^*(i,j;\hat{x}_a,\hat{x}_b,\mathbf{I})}{\sum_{i',j'}\sum_{\mathbf{I}\in\mathcal{I}} \alpha^*(i',j';\hat{x}_a,\hat{x}_b,\mathbf{I})}, \forall i,j \in \{1,\cdots,h\},$$

$$\overline{y}_a(i;\hat{x}_a,\hat{x}_b) = \sum_{j=1}^{h} \overline{y}_{ab}(i,j;\hat{x}_a,\hat{x}_b), \forall i \in \{1,\cdots,h\},$$

$$\overline{y}_b(j;\hat{x}_a,\hat{x}_b) = \sum_{i=1}^{h} \overline{y}_{ab}(i,j;\hat{x}_a,\hat{x}_b), \forall j \in \{1,\cdots,h\},$$

$$y'_{ab}(i,j;\hat{x}_a,\hat{x}_b) = \frac{\sum_{\mathbf{I}\in\mathcal{I}} \beta^*(i,j;\hat{x}_a,\hat{x}_b,\mathbf{I})}{\sum_{i',j'}\sum_{\mathbf{I}\in\mathcal{I}} \beta^*(i',j';\hat{x}_a,\hat{x}_b,\mathbf{I})}, \forall i,j \in \{1,\cdots,h\},$$

$$y'_a(i;\hat{x}_a,\hat{x}_b) = \sum_{j=1}^{h} y'_{ab}(i,j;\hat{x}_a,\hat{x}_b), \forall i \in \{1,\cdots,h\},$$

$$y'_b(j;\hat{x}_a,\hat{x}_b) = \sum_{i=1}^{h} y'_{ab}(i,j;\hat{x}_a,\hat{x}_b), \forall j \in \{1,\cdots,h\}.$$

The constraints of the dual of problem (27) ensure that

$$\overline{y}_a(i;\hat{x}_a,\hat{x}_b) = y'_a(i;\hat{x}_a,\hat{x}_b), \forall i \in \{1,\cdots,h\},$$
$$\overline{y}_b(j;\hat{x}_a,\hat{x}_b) = y'_b(j;\hat{x}_a,\hat{x}_b), \forall j \in \{1,\cdots,h\}.$$

Given the distributions $\overline{\mathbf{y}}_a(\hat{x}_a, \hat{x}_b)$ and $\overline{\mathbf{y}}_b(\hat{x}_a, \hat{x}_b)$, we claim that the pairwise variables $\overline{\mathbf{y}}_{ab}(\hat{x}_a, \hat{x}_b)$ must minimize the pairwise potentials corresponding to these distributions, that is,

$$\begin{aligned}
\overline{\mathbf{y}}_{ab}(\hat{x}_a, \hat{x}_b) = \quad & \operatorname*{argmin}_{\mathbf{y}_{ab}} \sum_{l_i, l_j \in \mathbf{L}} d(l_i, l_j) y_{ab}(i, j) \\
\text{s.t.} \quad & \sum_{l_j \in \mathbf{L}} y_{ab}(i, j) = \overline{y}_a(i; \hat{x}_a, \hat{x}_b), \forall l_i \in \mathbf{L} \\
& \sum_{l_i \in \mathbf{L}} y_{ab}(i, j) = \overline{y}_b(j; \hat{x}_a, \hat{x}_b), \forall l_j \in \mathbf{L} \\
& y_{ab}(i, j) \geq 0, \forall l_i, l_j \in \mathbf{L}.
\end{aligned}$$

If the above claim was not true, then we could further increase the value of at least one of the duals of problem (27) corresponding to some interval $\mathbf{I}$. Furthermore, since $\mathbf{y}'_{ab}(\hat{x}_a, \hat{x}_b)$ is constructed using $\boldsymbol{\beta}^*(\hat{x}_a, \hat{x}_b, \mathbf{I})$, which is uncrossing, a modification of Lemma 12 can be used to show that

$$\hat{d}(\hat{\mathbf{y}}_a(\hat{x}_a, \hat{x}_b), \hat{\mathbf{y}}_b(\hat{x}_a, \hat{x}_b)) = \sum_{l_i, l_j \in \mathbf{L}} d(l_i, l_j) y'_{ab}(i, j; \hat{x}_a, \hat{x}_b).$$

Here $\hat{\mathbf{y}}_a(\hat{x}_a, \hat{x}_b)$ and $\hat{\mathbf{y}}_b(\hat{x}_a, \hat{x}_b)$ are the rounded solutions obtained by the interval rounding procedure applied to $\overline{\mathbf{y}}_a(\hat{x}_a, \hat{x}_b)$ and $\overline{\mathbf{y}}_b(\hat{x}_a, \hat{x}_b)$.

By duality, it follows that

$$\frac{\sum_{l_i, l_j \in \mathbf{L}} d(l_i, l_j) y'_{ab}(i, j; \hat{x}_a, \hat{x}_b)}{\sum_{l_i, l_j \in \mathbf{L}} d(l_i, l_j) \overline{y}_{ab}(i, j; \hat{x}_a, \hat{x}_b)} \leq \frac{1}{q} \max_{x_a, x_b, \hat{x}_a, \hat{x}_b \in \mathbf{L}, x_a \neq x_b} \frac{D(x_a, x_b; \hat{x}_a, \hat{x}_b)}{d(x_a, x_b)}.$$

Strong duality implies that there must exist a set of variables for which the above inequality holds as an equality. This proves the lemma. ■

Lemmas 15 and 17 together prove Theorem 3.

Appendix C. Proof of Theorem 8

Proof The proof of the above theorem proceeds in a similar fashion to the previous two theorems, namely, by computing the dual of the problems that are used to obtain the tightest submodular overestimation of the given distance function. In what follows, we provide a brief sketch of the proof and omit the details.

We start with the hierarchical move-making algorithm. Recall that this algorithm computes a labeling $\mathbf{x}^{i,j}$ for each cluster $\mathbf{C}(i, j) \subseteq \mathbf{L}$, where $\mathbf{C}(i, j)$ denotes the j-th cluster at the i-th level. In order to obtain a labeling $\mathbf{x}^{i,j}$ it makes use of the labelings computed at the $(i + 1)$-th level. Specifically, let $\mathbf{L}_a^{i,j} = \{x_a^{i+1,j'}, p(i + 1, j') = j, j' \in \{1, \cdots, h^{i+1}\}\}$. In other words, $\mathbf{L}_a^{i,j}$ is the set of labels that were assigned to the random variable X_a by the labelings corresponding to the children of the current cluster $\mathbf{C}(i, j)$. In order to compute the labeling $\mathbf{x}^{i,j}$, the algorithm has to choose one label from the set $\mathbf{L}_a^{i,j}$. This is achieved either using the complete move-making algorithm or the interval move-making algorithm. The algorithm is initialized by define $x_a^{m,j} = k$ for all $X_a \in \mathbf{X}$ where k is the unique label in the cluster $\mathbf{C}(m, j)$ The algorithm terminates by computing $\mathbf{x}^{1,1}$, which is the final labeling.

Let the multiplicative bound corresponding to the computation of $\mathbf{x}^{i,j}$ be denoted by $B^{i,j}$. From the arguments of Theorems 1 and 3, it follows that there must exist dual variables $(\boldsymbol{\alpha}^*(i, j), \boldsymbol{\beta}^*(i, j), \boldsymbol{\gamma}^*(i, j))$ that provide an approximation factor that is exactly equal to $B^{i,j}$ and $\boldsymbol{\beta}^*(i, j)$ is uncrossing. For each cluster $\mathbf{C}(i, j)$, we define variables $\delta_a(k; i, j)$ for $k \in \mathbf{L}_a^{i,j}$ as follows:

$$\delta_a(k; i, j) = \frac{\sum_{k' \in \mathbf{L}_b^{i,j}} \alpha^*(k, k'; i, j)}{\sum_{l \in \mathbf{L}_a^{i,j}} \sum_{l' \in \mathbf{L}_b^{i,j}} \alpha^*(l, l'; i, j)}.$$

Using the above variables we define unary variables $y_a(k; i, j)$ for each $k \in \mathbf{L}^{i,j}$ as follows:

$$\begin{aligned}
& y_a(k; 1, 1) = \delta_a(k; 1, 1), \\
& y_a(k; i, j) = y_a(x_a^{i-1, p(i,j)}; i - 1, p(i, j)) \delta_a(k; i, j).
\end{aligned}$$

Similarly, we define variables $\delta_b(k'; i, j)$ for each $k' \in \mathbf{L}_b^{i,j}$ as follows:

$$\delta_b(k'; i, j) = \frac{\sum_{k \in \mathbf{L}_a^{i,j}} \alpha^*(k, k'; i, j)}{\sum_{l \in \mathbf{L}_a^{i,j}} \sum_{l' \in \mathbf{L}_b^{i,j}} \alpha^*(l, l'; i, j)}.$$

Using the above variables, we define unary variables $y_b(k'; i, j)$ for each $k' \in \mathbf{L}_b^{i,j}$ as follows:

$$\begin{aligned}
& y_b(k'; 1, 1) = \delta_b(k'; 1, 1), \\
& y_b(k'; i, j) = y_a(x_a^{i-1, p(i,j)}; i - 1, p(i, j)) \delta_b(k'; i, j).
\end{aligned}$$

Since each cluster in the m-th level contains a single unique label, it follows that

$$\begin{aligned}
& \mathbf{y}_a = [y_a(x_a^{m,j}; m, j), \forall j = 1, \cdots, h], \\
& \mathbf{y}_b = [y_b(x_b^{m,j}; m, j), \forall j = 1, \cdots, h],
\end{aligned}$$

are valid unary variables for the LP relaxation. It can be shown that applying the hierarchical rounding procedure on the variables $\mathbf{y}_a$ and $\mathbf{y}_b$ provides an approximation factor that is exactly equal to the multiplicative bound of the hierarchical move-making algorithm. ■

References

T. Ajanthan, R. Hartley, M. Salzmann, and H. Li. Iteratively reweighted graph cut for multi-label MRFs with non-convex priors. *CoRR*, 2014.

M. Andrew, M. Hajiaghayi, H. Karloff, and A. Moitra. Capacitated metric labeling. In *SODA*, 2011.

A. Archer, J. Fakcharoenphol, C. Harrelson, R. Krauthgamer, K. Talvar, and E. Tardos. Approximate classification via earthmover metrics. In *SODA*, 2004.

Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 2004.

Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *CVPR*, 1998.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV*, 1999.

C. Chekuri, S. Khanna, J. Naor, and L. Zosin. Approximation algorithms for the metric labeling problem via a new linear programming formulation. In *SODA*, 2001.

P. Dokania and M. P. Kumar. Parsimonious labeling. In *ICCV*, 2015.

J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *STOC*, 2003.

B. Flach and D. Schlesinger. Transforming an arbitrary minsum problem into a binary one. Technical report, TU Dresden, 2006.

A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *NIPS*, 2007.

A. Gupta and E. Tardos. A constant factor approximation algorithm for a class of classification problems. In *STOC*, 2000.

T. Hazan and A. Shashua. Convergent message-passing algorithms for inference over general graphs with convex free energy. In *UAI*, 2008.

J. Kappes, B. Andres, F. Hamprecht, C. Schnoerr, S. Nowozin, D. Batra, S. Kim, B. Kausler, T. Kroeger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother. A comparative study of modern inference techniques for structured discrete energy minimization problems. *IJCV*, 2015.

J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. In *STOC*, 1999.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 2006.

N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, 2007.

N. Komodakis, G. Tziritas, and N. Paragios. Performance vs computational efficiency for optimizing single and dynamic MRFs: Setting the state of the art with primal dual strategies. *CVIU*, 2008.

A. Koster, C. van Hoesel, and A. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 1998.

M. P. Kumar. Rounding-based moves for metric labeling. In *NIPS*, 2014.

M. P. Kumar and D. Koller. MAP estimation of semi-metric MRFs via hierarchical graph cuts. In *UAI*, 2009.

M. P. Kumar and P. Torr. Improved moves for truncated convex models. In *NIPS*, 2008.

R. Manokaran, J. Naor, P. Raghavendra, and R. Schwartz. SDP gaps and UGC hardness for multiway cut, 0-extension and metric labeling. In *STOC*, 2008.

P. Pansari and M. P. Kumar. Truncated max-of-convex models. *CoRR*, 2015.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kauffman, 1998.

H. Racke. Optimal hierarchical decompositions for congestion minimization in networks. In *STOC*, 2008.

P. Ravikumar, A. Agarwal, and M. Wainwright. Message-passing for graph-structured linear programs: Proximal projections, convergence and rounding schemes. In *ICML*, 2008.

M. Schlesinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Kibernetika*, 1976.

R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *PAMI*, 2008.

D. Tarlow, D. Batra, P. Kohli, and V. Kolmogorov. Dynamic tree block coordinate ascent. In *ICML*, 2011.

O. Veksler. *Efficient graph-based energy minimization methods in computer vision*. PhD thesis, Cornell University, 1999.

O. Veksler. Graph cut based optimization for MRFs with truncated convex priors. In *CVPR*, 2007.

M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on trees: Message passing and linear programming. *Transactions on Information Theory*, 2005.

C. Wang, N. Komodakis, and N. Paragios. Markov random field modeling, inference and learning in computer vision and image understanding: A survey. *CVIU*, 2013.

Y. Weiss, C. Yanover, and T. Meltzer. MAP estimation, linear programming and belief propagation with convex free energies. In *UAI*, 2007.

T. Werner. A linear programming approach to max-sum problem: A review. *PAMI*, 2007.

T. Werner. Revisting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *PAMI*, 2010.

Journal of Machine Learning Research 17 (2016) 1-27 Submitted 4/15; Revised 9/15; Published 6/16

Rate Optimal Denoising of Simultaneously Sparse and Low Rank Matrices

Dan Yang DYANG@STAT.RUTGERS.EDU
Department of Statistics and Biostatistics
Rutgers University
Piscataway, NJ 08854, USA

Zongming Ma ZONGMING@WHARTON.UPENN.EDU
Andreas Buja BUJA.AT.WHARTON@GMAIL.COM
Department of Statistics
University of Pennsylvania
Philadelphia, PA 19104, USA

Editor: Hui Zou

Abstract

We study minimax rates for denoising simultaneously sparse and low rank matrices in high dimensions. We show that an iterative thresholding algorithm achieves (near) optimal rates adaptively under mild conditions for a large class of loss functions. Numerical experiments on synthetic datasets also demonstrate the competitive performance of the proposed method.

Keywords: Denoising, High dimensionality, Low rank matrices, Minimax rates, Simultaneously structured matrices, Sparse SVD, Sparsity.

1. Introduction

In recent years, there has been a surge of interest in estimating and denoising structured large matrices. Leading examples include denoising low rank matrices (Donoho and Gavish, 2014), recovering low rank matrices from a small number of entries, i.e., matrix completion (Candès and Recht, 2009; Candès and Plan, 2010; Keshavan et al., 2010; Koltchinskii et al., 2011; Negahban and Wainwright, 2011), reduced rank regression (Bunea et al., 2011), group sparse regression (Yuan and Lin, 2006; Lounici et al., 2011), among others.

In the present paper, we study the problem of denoising an $m \times n$ data matrix

$$\mathbf{X} = \mathbf{M} + \mathbf{Z}. \tag{1}$$

The primary interest lies in the matrix $\mathbf{M}$ that is sparse in the sense that nonzero entries are assumed to be confined on a $k \times l$ block, which is not necessarily consecutive. In addition to being sparse, the rank of $\mathbf{M}$, denoted by r, is assumed to be low. Thus, $\mathbf{M}$ can be regarded as *simultaneously structured* as opposed to those simply structured cases where $\mathbf{M}$ is assumed to be either only sparse or only of low rank. To be concrete, we assume that $\mathbf{Z}$ consists of i.i.d. additive Gaussian white noise with variance σ^2. In the literature, the problem has also been referred to as the sparse SVD (singular value decomposition) problem. See, for instance, Yang et al. (2014) and the references therein.

The interest in this problem is motivated by a number of related problems:

1. *Biclustering.* It provides an ideal model for studying biclustering of microarray data. Let the rows of $\mathbf{X}$ correspond to cancer patients and the columns correspond to gene expression levels measured with microarrays. A subset of k patients can be clustered together as a subtype of the same cancer, which in turn is determined by a subset of l genes. Moreover, the gene expression levels on such a bicluster can be captured by a low rank matrix. See, e.g., Shabalin et al. (2009); Lee et al. (2010); Butucea and Ingster (2013); Sun and Nobel (2013); Chen et al. (2013); Gao et al. (2015).

2. *Recovery of simultaneously structured matrices with compressive measurements.* There has been emerging interest in the signal processing community in recovering such simultaneously structured matrices based on compressive measurements, partly motivated by problems such as sparse vector recovery from quadratic measurements and sparse phase retrieval. See, e.g., Shechtman et al. (2011); Li and Voroninski (2013); Lee et al. (2013); Oymak et al. (2013); Cai et al. (2015) and the references therein. The connection between the recovery problem and the denoising problem considered here is partially explored in Oymak and Hassibi (2013). An interesting phenomenon in the recovery setting is that convex relaxation approach no longer works well (Oymak et al., 2015) as it does in the simply structured cases.

3. *Sparse reduced rank regression.* The denoising problem is also closely connected to prediction in reduced rank regression where the coefficient matrix is also sparse. Indeed, let $n = l$, then problem (1) reduces to sparse reduced rank regression with orthogonal design. See Bunea et al. (2012) and Ma and Sun (2014) for more discussion.

The main contribution of the present paper includes the following: i) We provide information-theoretic lower bounds for the estimation error of $\mathbf{M}$ under squared Schatten-q norm losses for all $q \in [1, 2]$; ii) We propose a computationally efficient estimator that, under mild conditions, attains high probability upper bounds that match the minimax lower bounds within a multiplicative log factor (and sometimes even within a constant factor) simultaneously for all $q \in [1, 2]$. The theoretical results are further validated and supported by numerical experiments on synthetic data.

The rest of the paper is organized as follows. In Section 2, we precisely formulate the denoising problem and propose a denoising algorithm based on the idea of iterative thresholding. Section 3 establishes minimax risk lower bounds and high probability upper bounds that match the lower bounds within a multiplicative log factor for all squared Schatten-q norm losses with $q \in [1, 2]$. Section 4 presents several numerical experiments which demonstrate the competitive finite sample performance of the proposed denoising algorithm. Section 5 discusses several additional issues related to the present paper. The proofs of the main results are presented in Section 6, with some technical details relegated to Appendix A.

2. Problem Formulation and Denoising Method

Notation For any $a, b \in \mathbb{R}$, let $a \wedge b = \min(a, b)$ and $a \vee b = \max(a, b)$. For any two sequences of positive numbers $\{a_n\}$ and $\{b_n\}$, we write $a_n = O(b_n)$ if $a_n \leq C b_n$ for some

absolute positive constant C and all n. For any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, denote its successive singular values by $\sigma_1(\mathbf{A}) \geq \cdots \geq \sigma_{m\wedge n}(\mathbf{A}) \geq 0$. For any $q \in [1,\infty)$, the Schatten-q norm of $\mathbf{A}$ is defined as $\|\mathbf{A}\|_{\mathrm{S}_q} = (\sum_{i=1}^{m\wedge n} \sigma_i^q(\mathbf{A}))^{1/q}$. Thus, $\|\mathbf{A}\|_{\mathrm{S}_1}$ is the nuclear norm of $\mathbf{A}$ and $\|\mathbf{A}\|_{\mathrm{S}_2} = \|\mathbf{A}\|_{\mathrm{F}}$ is the Frobenius norm. In addition, the Schatten-∞ norm of $\mathbf{A}$ is $\|\mathbf{A}\|_{\mathrm{S}_\infty} = \sigma_1(\mathbf{A}) = \|\mathbf{A}\|_{\mathrm{op}}$, where $\|\cdot\|_{\mathrm{op}}$ stands for the operator norm. The rank of $\mathbf{A}$ is denoted by $\mathsf{rank}(\mathbf{A})$. For any vector $\mathbf{a}$, we denote its Euclidean norm by $\|\mathbf{a}\|$. For any integer m, $[m]$ stands for the set $\{1,\ldots,m\}$. For any subset $I \subset [m]$ and $J \subset [n]$, we use $\mathbf{A}_{IJ}$ to denote the submatrix of $\mathbf{A}$ with rows indexed by I and columns by J. When either I or J is the whole set, we replace it with $*$. For instance, $\mathbf{A}_{I*} = \mathbf{A}_{I[n]}$. Moreover, we use $\mathrm{supp}(\mathbf{A})$ to denote the set of nonzero rows of $\mathbf{A}$. For any set A, $|A|$ denotes its cardinality and A^c denotes its complement. A matrix $\mathbf{A}$ is called orthonormal, if the column vectors are of unit length and mutually orthogonal. For any event E, we use $\mathbf{1}_E$ to denote the indicator function on E, and E^c denotes its complement.

2.1 Problem Formulation

We now put the denoising problem in a decision-theoretic framework. Recall model (1). We are interested in estimating $\mathbf{M}$ based on the noisy observation $\mathbf{X}$, where $\mathbf{M}$ is simultaneously sparse and low rank. Let the singular value decomposition (SVD) of $\mathbf{M} = \mathbf{UDV}'$, where $\mathbf{U}$ is $m \times r$ orthonormal, $\mathbf{V}$ is $n \times r$ orthonormal and $\mathbf{D} = \mathrm{diag}(d_1,\ldots,d_r)$ is $r \times r$ diagonal with $d_1 \geq \cdots \geq d_r > 0$. In addition, since the nonzero entries on $\mathbf{M}$ concentrate on a $k \times l$ block, $\mathbf{U}$ has at most k nonzero rows and $\mathbf{V}$ at most l. Therefore, the parameter space of interest can be written as

$$\begin{aligned}\mathcal{F}(m,n,k,l,r,d,\kappa) = \{&\mathbf{M} = \mathbf{UDV}' \in \mathbb{R}^{m\times n} : \mathsf{rank}(\mathbf{M}) = r,\\ &|\mathrm{supp}(\mathbf{U})| \leq k, |\mathrm{supp}(\mathbf{V})| \leq l,\\ &d \leq d_r \leq \cdots \leq d_1 \leq \kappa d\}.\end{aligned} \tag{2}$$

We will focus on understanding the dependence of the minimax estimation error on the key model parameters (m,n,k,l,r,d), while $\kappa > 1$ is treated as an unknown universal constant. Without loss of generality, we assume $m \geq n$ here and after. Note that it is implicitly assumed in (2) that $m \geq k \geq r$ and $n \geq l \geq r$.

To measure the estimation accuracy, we use the following squared Schatten-q norm loss functions:

$$L_q(\mathbf{M},\widehat{\mathbf{M}}) = \|\widehat{\mathbf{M}} - \mathbf{M}\|_{\mathrm{s}_q}^2, \qquad q \in [1,2]. \tag{3}$$

The model (1), the parameter space (2) and the loss functions (3) give a precise formulation of the denoising problem.

2.2 Approach

From a matrix computation viewpoint, if one seeks a rank r approximation to a matrix $\mathbf{X}$, then one can first find its left and the right r leading singular vectors. If we organize these vectors as columns of the left and the right singular vector matrices $\mathbf{U}$ and $\mathbf{V}$, then the matrix $(\mathbf{UU}')\mathbf{X}(\mathbf{VV}')$ has the minimum Frobenius reconstruction error for $\mathbf{X}$ among all rank r matrices, since $\mathbf{U}'\mathbf{XV}$ will be a diagonal matrix consisting of the r leading singular values of $\mathbf{X}$. On the other hand, if one wants to enforce sparsity in the resulting matrix,

it is natural to utilize the idea of thresholding in the above calculation. Motivated by the above observation and also by an iterative thresholding idea previously used in solving sparse PCA problem (Ma, 2013; Yuan and Zhang, 2013), we propose the denoising scheme in Algorithm 1 via two-way iterative thresholding.

Algorithm 1: Matrix Denoising via Two-Way Iterative Thresholding

Input:
1. Observed data matrix $\mathbf{X}$.
2. Thresholding function η and thresholds γ_u and γ_v.
3. Rank r and noise standard deviation σ.
4. Initial orthonormal matrix $\mathbf{V}^{(0)} \in \mathbb{R}^{n\times r}$.

Output: Denoised matrix $\widehat{\mathbf{M}}$.

repeat

1 Right-to-Left Multiplication: $\mathbf{U}^{(t),\mathrm{mul}} = \mathbf{XV}^{(t-1)}$.

2 Left Thresholding: $\mathbf{U}^{(t),\mathrm{thr}} = (u_{ij}^{(t),\mathrm{thr}})$, with $\mathbf{U}_{i*}^{(t),\mathrm{thr}} = \frac{\mathbf{U}_{i*}^{(t),\mathrm{mul}}}{\|\mathbf{U}_{i*}^{(t),\mathrm{mul}}\|}\eta(\|\mathbf{U}_{i*}^{(t),\mathrm{mul}}\|,\gamma_u)$.

3 Left Orthonormalization with QR Decomposition: $\mathbf{U}^{(t)}\mathbf{R}_u^{(t)} = \mathbf{U}^{(t),\mathrm{thr}}$.

4 Left-to-Right Multiplication: $\mathbf{V}^{(t),\mathrm{mul}} = \mathbf{X}'\mathbf{U}^{(t)}$.

5 Right Thresholding: $\mathbf{V}^{(t),\mathrm{thr}} = (v_{ij}^{(t),\mathrm{thr}})$, with $\mathbf{V}_{i*}^{(t),\mathrm{thr}} = \frac{\mathbf{V}_{i*}^{(t),\mathrm{mul}}}{\|\mathbf{V}_{i*}^{(t),\mathrm{mul}}\|}\eta(\|\mathbf{V}_{i*}^{(t),\mathrm{mul}}\|,\gamma_v)$.

6 Right Orthonormalization with QR Decomposition: $\mathbf{V}^{(t)}\mathbf{R}_v^{(t)} = \mathbf{V}^{(t),\mathrm{thr}}$.

until *Convergence*;

7 Compute projection matrices $\widehat{\mathbf{P}}_u = \widehat{\mathbf{U}}\widehat{\mathbf{U}}'$ and $\widehat{\mathbf{P}}_v = \widehat{\mathbf{V}}\widehat{\mathbf{V}}'$, where $\widehat{\mathbf{U}}$ and $\widehat{\mathbf{V}}$ are $\mathbf{U}^{(t)}$ and $\mathbf{V}^{(t)}$ at convergence.

8 Compute denoised matrix $\widehat{\mathbf{M}} = \widehat{\mathbf{P}}_u\mathbf{X}\widehat{\mathbf{P}}_v$.

Without the two thresholding steps, the iterative part of the algorithm computes the leading singular vectors of any rectangular matrix, and can be viewed as a two-way generalization of the power iteration (Golub and Van Loan, 1996).

In the thresholding steps, we apply row-wise thresholding to the matrix $\mathbf{U}^{(t),\mathrm{mul}}$ (resp. $\mathbf{V}^{(t),\mathrm{mul}}$) obtained after the multiplication step. In the thresholding function $\eta(x,t)$, the second argument $t > 0$ is called the threshold level. In Algorithm 1, the first argument x is always non-negative. In order for the later theoretical results to work, we impose the following minimal assumption on the thresholding function η:

$$\begin{aligned}|\eta(x,t) - x| \leq t, &\quad \text{for any } x \geq 0,\ t > 0,\\ \eta(x,t) = 0, &\quad \text{for any } t > 0,\ x \in [0,t].\end{aligned} \tag{4}$$

Examples of such thresholding functions include the usual soft and hard thresholding, the SCAD (Fan and Li, 2001), the MCP (Zhang, 2010), etc. Thus, for instance, when thresholding $\mathbf{U}^{(t),\mathrm{mul}}$, if η is the hard thresholding function, then we are going to keep all the rows whose norms are greater than γ_u and kill all the rows whose norms are smaller than γ_u. For other thresholding function, we shrink the norms according to η while keeping the phases of the row vectors. Throughout the iterations, the threshold levels γ_u and γ_v are pre-specified

and remain unchanged. In order for the theorem to work, these levels can be chosen as in (10) below.

To determine the convergence of the iterative part, we could either run a pre-specified number of iterations or stop after the difference between successive iterates are sufficiently small, e.g.,

$$\|\mathbf{U}^{(t)}(\mathbf{U}^{(t)})' - \mathbf{U}^{(t-1)}(\mathbf{U}^{(t-1)})'\|_{\mathrm{F}}^2 \vee \|\mathbf{V}^{(t)}(\mathbf{V}^{(t)})' - \mathbf{V}^{(t-1)}(\mathbf{V}^{(t-1)})'\|_{\mathrm{F}}^2 \le \epsilon, \tag{5}$$

where ϵ is a pre-specified tolerance level.

Initialization To initialize Algorithm 1, we need to further specify the rank r, the noise standard deviation σ and a starting point $\mathbf{V}^{(0)}$ for the iteration. For the ease of exposition, we assume that r is known. Otherwise, it can be estimated by methods such as those described in Yang et al. (2014). When we have Gaussian noise and $kl < \frac{1}{2}mn$, the noise standard deviation can be estimated by

$$\widehat{\sigma} = 1.4826 \cdot \mathrm{MAD}(\{\mathbf{M}_{ij} : i \in [m], j \in [n]\}). \tag{6}$$

Finally, to obtain a reasonable initial orthonormal matrix $\mathbf{V}^{(0)}$, we propose to use Algorithm 2 for the case of Gaussian noise.

Algorithm 2: Initialization for Algorithm 1

Input:

1. Observed data matrix $\mathbf{X}$.
2. Tuning parameter α.
3. Rank r and noise standard deviation σ.

Output: Estimators $\widehat{\mathbf{U}} = \mathbf{U}^{(0)}$ and $\widehat{\mathbf{V}} = \mathbf{V}^{(0)}$.

1 Select the subset I_0 of rows and the subset J_0 of columns as

$$I^0 = \{i : \|\mathbf{X}_{i*}\|^2 \ge \sigma^2(n + \alpha\sqrt{n \log n})\}, \tag{7a}$$

$$J^0 = \{j : \|\mathbf{X}_{*j}\|^2 \ge \sigma^2(m + \alpha\sqrt{m \log m})\}. \tag{7b}$$

2 Compute $\mathbf{X}^{(0)} = (x_{ij}^{(0)})$, where $x_{ij}^{(0)} = x_{ij}\mathbf{1}_{i\in I^0}\mathbf{1}_{j\in J^0}$.

3 Compute $\mathbf{U}^{(0)} = [\mathbf{u}_1^{(0)}, \ldots, \mathbf{u}_r^{(0)}]$ and $\mathbf{V}^{(0)} = [\mathbf{v}_1^{(0)}, \ldots, \mathbf{v}_r^{(0)}]$, where $\mathbf{u}_\nu^{(0)}$ ($\mathbf{v}_\nu^{(0)}$) is the ν^{th} leading left (right) singular vector of $\mathbf{X}^{(0)}$.

Remark 1 *In practice, Algorithms 1 and 2 are not restricted to the denoising of matrices with Gaussian noise. With proper modification and robustification, they can be used together to deal with other noise distributions and/or outliers. See, e.g., Yang et al. (2014).*

3. Theoretical Results

In this section, we present a minimax theory underlying the denoising/estimation problem formulated in Section 2.1.

3.1 Minimax Lower Bounds

Theorem 2 *Let $\mathcal{F} = \mathcal{F}(m, n, k, l, r, d, \kappa)$ with $\kappa > 1$ and $k \wedge l \ge 2r$. There exists a positive constant c that depends only on κ, such that for any $q \in [1, 2]$, the minimax risk for estimating $\mathbf{M}$ under the squared Schatten-q error loss (3) satisfies*

$$\inf_{\widehat{\mathbf{M}}} \sup_{\mathcal{F}} \mathbb{E}L_q(\mathbf{M}, \widehat{\mathbf{M}}) \ge c\sigma^2 \left[\left(r^{\frac{2}{q}-1}\frac{d^2}{\sigma^2}\right) \wedge \Psi_q(m, n, k, l, r)\right] \tag{8}$$

where the rate function $\Psi_q(m, n, k, l, r) = r^{\frac{2}{q}}(k + l) + r^{\frac{2}{q}-1}\left(k \log \frac{em}{k} + l \log \frac{en}{l}\right)$.

A proof of the theorem is given in Section 6.1.

Remark 3 *Regardless of the value of q, the lower bounds reflect two different scenarios.*

The first scenario is the "low signal" case where

$$d^2 \le \sigma^2 \Psi_2(m, n, k, l, r). \tag{9}$$

In this case, the first term in the lower bound (8) dominates, and the rate is achieved by simply estimating $\mathbf{M}$ by $\mathbf{0} \in \mathbb{R}^{m\times n}$.

The second scenario is when (9) does not hold. In this case, the second term in (8) dominates. We note this term is expressed as the sum of two terms. As to be revealed by the proof, the first summand is an "oracle" error term which occurs even when the indices of the nonzero rows and columns of $\mathbf{M}$ are given by an oracle. In contrast, the second summand results from the combinatorial uncertainty about the locations of these nonzero rows and columns.

3.2 Minimax Upper Bounds

To state the upper bounds, we first specify the threshold levels used in Algorithm 1. In particular, for some sufficiently large constant $\beta > 0$, set

$$\gamma_u^2 = \gamma_v^2 = \gamma^2 = 1.01(r + 2\sqrt{r\beta \log m} + 2\beta \log m). \tag{10}$$

For Theorem 4 to hold, it suffices to choose any $\beta \ge 4$. In addition, we specify the stopping (convergence) rule for the loop in Algorithm 1. For $\mathbf{X}^{(0)}$ defined in Algorithm 2, let $d_r^{(0)}$ be its r^{th} largest singular value. Define

$$\widehat{T} = \frac{1.1}{2}\left[\frac{\log m}{\log 2} + \log \frac{(d_r^{(0)})^2}{\gamma^2}\right], \tag{11}$$

and

$$T = \frac{1.01}{2}\left[\frac{\log m}{\log 2} + \log \frac{d_r^2}{k\gamma_u^2 \vee l\gamma_v^2}\right]. \tag{12}$$

We propose to stop the iteration in Algorithm 1 after $\widehat{T}$ steps. Last but not least, we need the following technical condition.

Condition 1 *There exists a sufficiently small absolute constant c, such that $m \geq n$, $\log d \leq cm$, $c \leq \log m / \log n \leq 1/c$, $\log m \leq c[(m-k) \wedge (n-l)]$, $k \vee l \leq c(m \wedge n)$. In addition, there exists a sufficiently small constant c' that depends only on κ, such that $(\sigma/d)^2 r \left(k\sqrt{n \log m} + l\sqrt{m \log m}\right) \leq c'$.*

With the above definition, the following theorem establishes high probability upper bounds of the proposed estimator.

Theorem 4 *Let Condition 1 be satisfied. In Algorithm 1, let $\mathbf{V}^{(0)}$ be obtained by Algorithm 2 with $\alpha \geq 4$ in (7). Let γ_u and γ_v be defined as in (10) with $\beta \geq 4$. Moreover, we stop the iteration after $\widehat{T}$ steps with $\widehat{T}$ defined in (11), and use $\widehat{\mathbf{U}} = \mathbf{U}^{(\widehat{T})}$ and $\widehat{\mathbf{V}} = \mathbf{V}^{(\widehat{T})}$ in subsequent steps. For sufficiently large values of m and n, uniformly over $\mathcal{F}(m, n, k, l, r, d, \kappa)$, with probability at least $1 - O(m^{-2})$, $\widehat{T} \in [T, 3T]$ and*

$$\|\widehat{\mathbf{M}} - \mathbf{M}\|_{s_q}^2 \leq C\sigma^2 \left[r^{\frac{2}{q}} (k + l + \log m) + r^{\frac{2}{q}-1} (k+l) \log m \right]$$

where C is a positive constant that depends only on κ and β.

The proof of the theorem is given in Section 6.2.

Remark 5 *Under Condition 1, for sufficient large values of m and n, (9) cannot hold, and so the relevant lower bound is $c\sigma^2 \Psi_q(m, n, k, l, r)$. In comparison, when $k \wedge l \geq (1+\epsilon)r$ for any universal small constant $\epsilon > 0$, the upper bounds in Theorem 4 always matches the lower bounds for all $q \in [1, 2]$ up to a multiplicative log factor. If in addition, $\log m = O(k \vee l)$ and $k = O(m^a)$ and $l = O(n^a)$ for some constant $a \in (0, 1)$, then the rates in the lower and upper bounds match exactly for all $q \in [1, 2]$. We note that Condition 1 can essentially be interpreted as a minimum signal-to-noise ratio condition where we require $d^2/\sigma^2 \geq Cr(k\sqrt{n \log m} + l\sqrt{m \log m})$. The condition is needed in guaranteeing the success of the initialization method in Algorithm 2 and the rank selection method to be introduced below.*

Remark 6 *The proposed estimator is adaptive since it does not depend on the knowledge of k, l and q. Its dependence on r can also be removed, as we explain in the next subsection.*

3.3 Rank Selection

We now turn to data-based selection of the rank r. Recall the sets I^0 and J^0 defined in (7). We propose to use the following data-based choice of r:

$$\widehat{r} = \max\left\{ s : \sigma_s(\mathbf{X}_{I^0 J^0}) \geq \sigma\, \delta_{|I^0||J^0|} \right\}, \tag{13}$$

where for any $i \in [m]$ and $j \in [n]$, $\delta_{ij} = \sqrt{i} + \sqrt{j} + \sqrt{2i \log \frac{\mathrm{e}m}{i} + 2j \log \frac{\mathrm{e}n}{j} + 8 \log m}$. Here, the threshold $\sigma\delta_{|I^0||J^0|}$ used in rank selection (13) is motivated by the Davidson–Szarek bound (Davidson and Szarek, 2001) and the union bound. Basically, $\sigma\delta_{|I^0||J^0|}$ bounds the largest singular value of any submatrix of $\mathbf{Z}$ of size $|I^0|$ by $|J^0|$ and any singular value larger than the threshold has to come from the signal part. We note that it is straightforward to incorporate this rank selection step into Algorithm 2. Indeed, we can compute $\widehat{r}$ right after step 1 and replace all r in the subsequent steps by $\widehat{r}$. The following result justifies our proposal.

Proposition 7 *Under the condition of Theorem 4, $\widehat{r} = r$ holds with probability at least $1 - O(m^{-2})$.*

A proof of the proposition is given in Section 6.3. According to Proposition 7, we can use $\widehat{r}$ as the input for rank in Algorithm 1 and the conclusion of Theorem 4 continues to hold.

4. Simulation

In this section, we demonstrate the performance of the proposed denoising method on synthetic datasets.

In the first numerical experiment, we study the effect of the signal-to-noise ratio. To this end, we fix $m = 2000, n = 1000, k = l = 50$, $r = 10$, and set the singular values of $\mathbf{M}$ as $(d_1, \ldots, d_{10}) = (200, 190, \ldots, 120, 110)$. The signal-to-noise ratio is varied by varying noise standard deviation σ on ten equally spaced values between 0.2 and 2. The $\mathbf{U}$ matrix is obtained by orthonormalizing a $m \times r$ matrix the i^{th} row of which is filled i.i.d. $N(0, i^4)$ entries for any $i \in [k]$ and zeros otherwise. The $\mathbf{V}$ matrix is obtained in the same way with m and k replaced by n and l. Fig. 1 shows the boxplots of $L_q(\mathbf{M}, \widehat{\mathbf{M}})$ for $q = 1$ and 2 for the varying values of σ out of 100 repetitions for each of the ten values of σ. Throughout this section, we use (6) to estimate σ, Algorithm 2 with $\alpha = 4$ to compute $\mathbf{V}^{(0)}$ and (13) to select the rank. In Algorithm 1, we set $\beta = 3$ and we terminate the iteration once (5) holds with $\epsilon = 10^{-10}$. The thresholding function η is fixed to be hard thresholding $\eta(x, t) = x\mathbf{1}_{|x|>t}$. In all the repetitions, the proposed $\widehat{r}$ in (13) consistently yields the right rank $r = 10$. From Fig. 1, we conclude that the denoising error grows quadratically with the value of σ, which agrees well with the theoretical results in Theorem 4.

In the second experiment, we study the effect of the rank r on denoising error. To this end, we fix $m = 2000, n = 1000, k = l = 50$, $\sigma = 1$ and all the nonzero singular values of $\mathbf{M}$ as $d_1 = \cdots = d_r = 200$. We vary r between 1 and 10. For each value of r, the $\mathbf{U}$ and $\mathbf{V}$ matrices are generated in the same way as in the first experiment. Fig. 2 shows the boxplots of $L_q(\mathbf{M}, \widehat{\mathbf{M}})$ for $q = 1$ and 2 for the varying values of r out of 100 repetitions for each of the ten values of r. It is noticeable that for the L_1 loss, the denoising error grows quadratically as the rank r grows, while the trend is linear for the L_2 loss. Both cases agree with Theorem 4.

In the third experiment, we study the effect of the parameter q used in the loss function. To this end, we generated datasets in the same way as in the first experiment while fixing $\sigma = 1$. We take eleven different values of $q \in [1, 2]$ such that their reciprocals are equally spaced between 0.5 and 1. As Fig. 3 shows, the logarithm of the loss function scales linearly with $1/q$. Again, this is in accordance with the error bound in Theorem 4.

In the fourth experiment, we study the effect of sparsity levels. To this end, we fix $m = 2000, n = 1000, r = 10$ and the singular values of $\mathbf{M}$ are $(d_1, \ldots, d_{10}) = (200, 190, \ldots, 120, 110)$. On the other hand, we consider four different combinations of sparsity parameters: $(k, l) = (50, 50), (50, 200), (100, 200)$ and $(100, 50)$. For each (k, l) pair, the way we generate $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{X}$ is the same as that in the first experiment. Moreover, the tuning parameter values used in denoising are also the same as before. In all the repetitions, $\widehat{r}$ in (13) consistently select $r = 10$. In Table 1, we report the average values of $L_q(\mathbf{M}, \widehat{\mathbf{M}})$ for $q = 2$ and 1 and

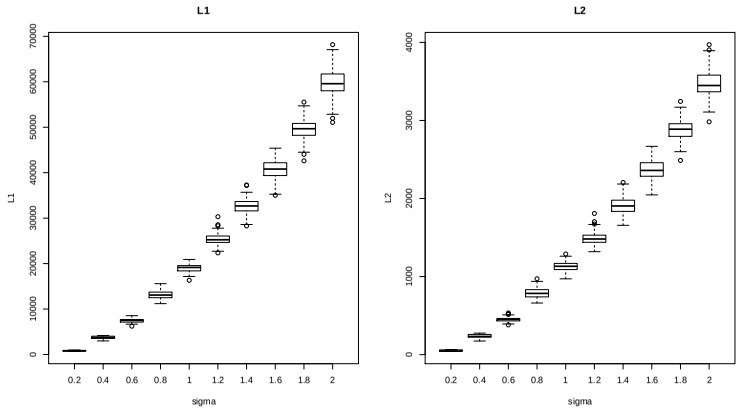


Figure 1: Loss vs. σ: boxplots of 100 repetitions. Left panel: L_1 loss. Right panel: L_2 loss.

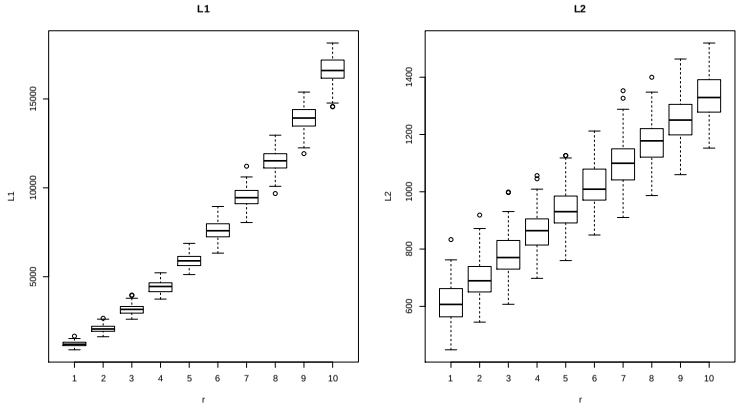


Figure 2: Loss vs. rank: boxplots of 100 repetitions. Left panel: L_1 loss. Right panel: L_2 loss.

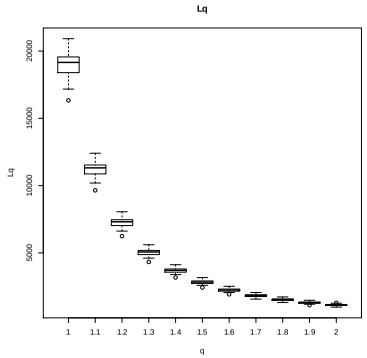


Figure 3: Loss vs. q: boxplots of 100 repetitions.

(k, l)	(50, 50)	(50, 200)	(100, 200)	(100, 50)
Average$(L_2(\mathbf{M}, \widehat{\mathbf{M}}))$	1133.03	2662.07	3598.69	1673.49
Standard error	(5.96)	(11.73)	(12.84)	(9.73)
Average$\left(\frac{L_2(\mathbf{M},\widehat{\mathbf{M}})}{(r+\log m)(k+l)}\right)$	0.64	0.60	0.68	0.63
Average$(L_1(\mathbf{M}, \widehat{\mathbf{M}}))$	19056.47	43035.95	65099.19	28347.12
Standard error	(88.42)	(172.39)	(231.98)	(146.07)
Average$\left(\frac{L_1(\mathbf{M},\widehat{\mathbf{M}})}{(r^2+r\log m)(k+l)}\right)$	1.08	0.98	1.23	1.07

Table 1: Average losses (with its standard error) and average rescaled losses of $\widehat{\mathbf{M}}$ out of 100 repetitions for different sparsity levels.

their standard errors over 100 repetitions. Moreover, we report the rescaled average loss where the rescaling constant is chosen to be $r^{\frac{2}{q}-1}(r+\log m)(k+l)$, the rate derived in Theorem 4. By the results reported in Table 1, we see that for either loss function, the rescaled average losses are stable with respect to different sparsity levels specified by different values of k and l. Again, this agrees well with the earlier theoretical results.

In the last experiment, we study the effect of "spikiness", i.e., how the concentration of energy of the nonzero entries in $\mathbf{M}$ affects denoising. To this end, we fix $m = 2000, n = 1000, k = l = 50$, $r = 1$, $d_1 = 200$ and $\sigma = 1$. In this case, $\mathbf{U}$ and $\mathbf{V}$ are both vectors. To vary the "spikiness", we set all nonzero entries of $\mathbf{V}$ to be equal and the nonzero entries of $\mathbf{U}$ as

$$\mathbf{U}_{i1} = C_s\, i^{-1/s}, \quad \text{for } i = 1, \ldots, k,$$

for a grid of twenty equally spaced s values in $\{0.1, 0.2, \ldots, 2\}$. Here, for any given s, C_s is the normalizing constant ensuring $\|\mathbf{U}\| = 1$. The smaller the s value, the faster the nonzero

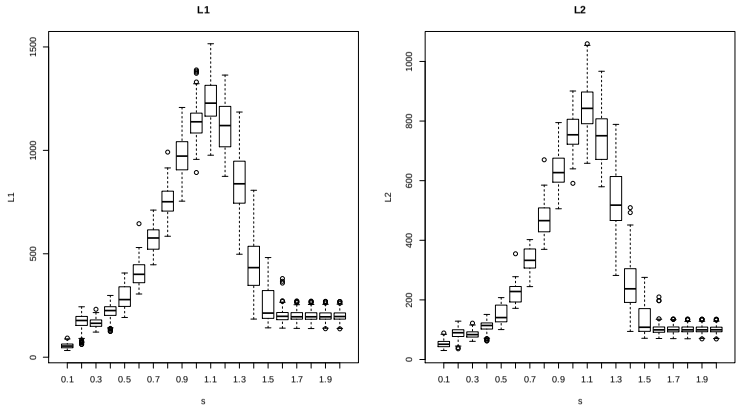


Figure 4: Loss vs. "spikiness": boxplots of 100 repetitions. Left panel: L_1 loss. Right panel: L_2 loss.

entries in $\mathbf{U}$ decay to zero and the more concentrated the energy of the nonzero entries in $\mathbf{U}$ and hence in $\mathbf{M}$. Fig. 4 shows the boxplots of $L_q(\mathbf{M}, \widehat{\mathbf{M}})$ for $q = 1$ and 2 for the varying values of s out of 100 repetitions for each of the twenty values of s. A somewhat surprising phenomenon is that the error is not monotone in "spikiness". For both L_1 and L_2 losses, the errors first increase as s increases until they reach the peak when $s = 1.1$ and then the errors decay as s further increases. Finally, they stabilize after s becomes 1.6 or larger. We feel that this intriguing phenomenon might be explained intuitively as the following. For small to medium values of s, both the bias and the variance in the estimator increases with s. However, due to the additional l_0 sparsity of $\mathbf{U}$, the bias starts to decrease after s grows larger than some critical value (1.1 in this simulation). Furthermore, when s grows larger than another critical value (1.6 in this simulation), the estimator essentially estimates the entire nonzero block in $\mathbf{M}$ and the bias term vanishes, which explains the stabilization near the end of the curves in Fig. 4.

5. Discussion

In this section, we discuss two additional issues related to the present work.

Sparse PCA The present paper is closely related to the sparse PCA problem where the interest is in estimating the sparse leading eigenvectors of the covariance matrix of the observed random vectors. Although the proposed algorithm in the present paper is partly motivated by the sparse PCA estimation schemes in Ma (2013) and Yuan and Zhang (2013), there are several important differences between the denoising model (1) and the sparse PCA model. First, we aim to recover the mean matrix in model (1) while sparse PCA is interested

in functionals (leading eigenvectors) of the covariance matrix. In addition, sparse PCA only deals with one set of sparse eigenvectors while the denoising problem studied here needs to deal with both left and right sparse singular vectors. In particular, even in the case of Schatten-2 (Frobenius) norm loss, the theoretical results of the present paper cannot be obtained directly from those for sparse PCA, such as those in Ma (2013), Cai et al. (2013a,b), etc.

Loss function In the present paper, we have considered the collection of squared Schatten-q norm for $q \in [1, 2]$ as possible loss functions. One of the referees raised the question whether it is possible to extend the results to the case of $q < 1$, where the loss function becomes a squared quasinorm[1]. This could serve as an interesting topic for future research. However, when $q < 1$, the quasinorm is no longer convex and so the technique for establishing estimation lower bounds which was first laid out in Ma and Wu (2015) no longer applies directly.

6. Proofs

6.1 Proof of Theorem 2

The proof of Theorem 2 relies on the following theorem, which is quoted without proof.

Theorem 8 (Theorem 2 in Ma and Sun (2014)) *Let the observed $\mathbf{X} \in \mathbb{R}^{n\times p}, \mathbf{Y} \in \mathbb{R}^{n\times m}$ be generated by the model $\mathbf{Y} = \mathbf{X}\mathbf{A}+\mathbf{Z}$ with $\mathbf{Z}$ having i.i.d. $N(0, \sigma^2)$ entries. Suppose for some absolute constant $\gamma > 1$, the coefficient matrix $\mathbf{A} \in \mathbb{R}^{p\times m}$ belongs to*

$$\Theta(s, r, d, \gamma) = \{\mathbf{A} : \mathrm{rank}(\mathbf{A}) = r, \gamma d \geq \sigma_1(\mathbf{A}) \geq \cdots \geq \sigma_r(\mathbf{A}) \geq d > 0, |\mathrm{supp}(\mathbf{A})| \leq s\}.$$

Moreover, suppose the $(2s)$-sparse Riesz constants of the design matrix $\mathbf{X}$ satisfy $K^{-1} \leq \kappa_-(2s) \leq \kappa_+(2s) \leq K$ for some absolute constant $K > 1$. Then there exists a positive constant c depending only on γ and $\kappa_+(2s)$ such that for all $q \in [1, 2]$,

$$\inf_{\widehat{\mathbf{A}}} \sup_{\Theta} \mathbb{E}L_q(\mathbf{A}, \widehat{\mathbf{A}}) \geq c\sigma^2 \left\{ \left(r^{2/q-1}\frac{d^2}{\sigma^2} \right) \wedge \left[r^{2/q}(s + m) + r^{2/q-1} s \log \frac{ep}{s} \right] \right\}.$$

Proof [Proof of Theorem 2] To establish the lower bound, first consider the subset $\mathcal{F}_1 \subset \mathcal{F}(m, n, k, l, r, d, \kappa)$ where we further require $\mathrm{supp}(\mathbf{V}) = [r]$. Thus, except for the first r columns, all columns of $\mathbf{M}$ are zeros. So, by a simple sufficiency argument, we may assume that $n = l = r$. In this case, the problem of estimating (the first r columns of) $\mathbf{M}$ under model (1) can be viewed as a special case of sparse reduced rank regression where the design matrix is the identity matrix $\mathbf{I}_m$. Note that the sparse Riesz constants for $\mathbf{I}_m$ are all equal to one. Therefore, Theorem 8 implies that

$$\inf_{\widehat{\mathbf{M}}} \sup_{\mathcal{F}} \mathbb{E}L_q(\mathbf{M}, \widehat{\mathbf{M}}) \geq \inf_{\widehat{\mathbf{M}}} \sup_{\mathcal{F}_1} \mathbb{E}L_q(\mathbf{M}, \widehat{\mathbf{M}}) \geq c \left[r^{\frac{2}{q}-1} d^2 \wedge \left(r^{\frac{2}{q}} k + r^{\frac{2}{q}-1} k \log \frac{em}{k} \right) \right].$$

By symmetry, we also have

$$\inf_{\widehat{\mathbf{M}}} \sup_{\mathcal{F}} \mathbb{E}L_q(\mathbf{M}, \widehat{\mathbf{M}}) \geq c \left[r^{\frac{2}{q}-1} d^2 \wedge \left(r^{\frac{2}{q}} l + r^{\frac{2}{q}-1} l \log \frac{en}{l} \right) \right].$$

1. A function satisfying all norm axioms except for the triangle inequality.

We complete the proof by noting that for any $a, b, c > 0$, $(a \wedge b) \vee (a \wedge c) = a \wedge (b \vee c) \asymp a \wedge (b+c)$. ■

6.2 Proof of Theorem 4

To prove Theorem 4, we follow the oracle sequence approach developed in Ma (2013). Throughout the proof, we assume that $\sigma = 1$ is known. The case of general $\sigma > 0$ comes from obvious scaling arguments. In what follows, we first define the oracle sequence and introduce some preliminaries. Then we give an overview of the proof, which is divided into three steps. After the overview, the three steps are carried out in order, which then leads to the final proof of the theorem. Due to the space limit, proofs of intermediate results are omitted.

Preliminaries We first introduce some notation. For any matrix $\mathbf{A}$, $\mathrm{span}(\mathbf{A})$ stands for the subspace spanned by the column vectors of $\mathbf{A}$. If we were given the oracle knowledge of $I = \mathrm{supp}(\mathbf{U})$ and $J = \mathrm{supp}(\mathbf{V})$, then we can define an oracle version of the observed matrix as

$$\widetilde{\mathbf{X}} = (x_{ij} \mathbf{1}_{i \in I} \mathbf{1}_{j \in J}) \in \mathbb{R}^{m \times n}. \tag{14}$$

With appropriate rearrangement of rows and columns, the $I \times J$ submatrix concentrates on the top-left corner. From now on, we assume that this is the case. We denote the singular value decomposition of $\mathbf{X}$ by

$$\widetilde{\mathbf{X}} = \begin{bmatrix} \widetilde{\mathbf{U}} & \widetilde{\mathbf{U}}_\perp \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{D}} & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{D}}_\perp \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{V}}' \\ (\widetilde{\mathbf{V}}_\perp)' \end{bmatrix}, \tag{15}$$

where $\widetilde{\mathbf{U}}, \widetilde{\mathbf{D}}, \widetilde{\mathbf{V}}$ consist of the first r singular triples of $\widetilde{\mathbf{X}}$, and $\widetilde{\mathbf{U}}_\perp, \widetilde{\mathbf{D}}_\perp, \widetilde{\mathbf{V}}_\perp$ contain the remaining $n - r$ triples (recall that we have assumed $m \geq n$). In particular, the successive singular values of $\widetilde{\mathbf{X}}$ are denoted by $\widetilde{d}_1 \geq \widetilde{d}_2 \geq \cdots \geq \widetilde{d}_n \geq 0$.

With the oracle knowledge of I and J, we can define oracle versions of Algorithm 2 and Algorithm 1. In the oracle version of Algorithm 2, we replace the subsets I^0 and J^0 by $\widetilde{I}^0 = I^0 \cap I$ and $\widetilde{J}^0 = J^0 \cap J$, and the output matrices are denoted by $\widetilde{\mathbf{U}}^{(0)}$ and $\widetilde{\mathbf{V}}^{(0)}$. In the oracle version of Algorithm 1, $\mathbf{X}$ is replaced by $\widetilde{\mathbf{X}}$ and $\mathbf{V}^{(0)}$ is replaced by $\widetilde{\mathbf{V}}^{(0)}$. The intermediate matrices obtained after each step within the loop are denoted by $\widetilde{\mathbf{U}}^{(t),\mathrm{mul}}$, $\widetilde{\mathbf{U}}^{(t),\mathrm{thr}}$, $\widetilde{\mathbf{U}}^{(t)}$ and $\widetilde{\mathbf{V}}^{(t),\mathrm{mul}}$, $\widetilde{\mathbf{V}}^{(t),\mathrm{thr}}$, $\widetilde{\mathbf{V}}^{(t)}$, respectively. We note that for any t, it is guaranteed that

$$\mathrm{supp}(\widetilde{\mathbf{U}}^{(t),\mathrm{thr}}) = \mathrm{supp}(\widetilde{\mathbf{U}}^{(t)}) \subset I,$$
$$\mathrm{supp}(\widetilde{\mathbf{V}}^{(t),\mathrm{thr}}) = \mathrm{supp}(\widetilde{\mathbf{V}}^{(t)}) \subset J.$$

To investigate the properties of the oracle sequence, we will trace the evolution of the columns subspaces of $\widetilde{\mathbf{U}}^{(t),\mathrm{mul}}$, $\widetilde{\mathbf{U}}^{(t)}$, $\widetilde{\mathbf{V}}^{(t),\mathrm{mul}}$ and $\widetilde{\mathbf{V}}^{(t)}$. To this end, denote the r canonical angles (Golub and Van Loan, 1996) between $\mathrm{span}(\widetilde{\mathbf{U}}^{(t),\mathrm{mul}})$ and $\mathrm{span}(\widetilde{\mathbf{U}})$ by $\pi/2 \geq \phi_{u,1}^{(t)} \geq \cdots \geq \phi_{u,r}^{(t)} \geq 0$, and define

$$\sin \Phi_u^{(t)} = \mathrm{diag}(\sin \phi_{u,1}^{(t)}, \ldots, \sin \phi_{u,r}^{(t)}). \tag{16}$$

Moreover, denote the canonical angles between $\mathrm{span}(\widetilde{\mathbf{U}}^{(t)})$ and $\mathrm{span}(\widetilde{\mathbf{U}})$ by $\pi/2 \geq \theta_{u,1}^{(t)} \geq \cdots \geq \theta_{u,r}^{(t)} \geq 0$, and let

$$\sin \Theta_u^{(t)} = \mathrm{diag}(\sin \theta_{u,1}^{(t)}, \ldots, \sin \theta_{u,r}^{(t)}). \tag{17}$$

The quantities $\phi_{v,i}^{(t)}$, $\sin \Phi_v^{(t)}$, $\theta_{v,i}^{(t)}$ and $\sin \Theta_v^{(t)}$ are defined analogously. For any pair of $m \times r$ orthonormal matrices $\mathbf{W}_1$ and $\mathbf{W}_2$, let the canonical angles between $\mathrm{span}(\mathbf{W}_1)$ and $\mathrm{span}(\mathbf{W}_2)$ be $\pi/2 \geq \theta_1 \geq \cdots \geq \theta_r \geq 0$ and $\sin \Theta = \mathrm{diag}(\sin \theta_1, \ldots, \sin \theta_r)$, then (Stewart and Sun, 1990)

$$\begin{aligned} \|\sin \Theta\|_{\mathrm{F}} &= \frac{1}{\sqrt{2}} \|\mathbf{W}_1 \mathbf{W}_1' - \mathbf{W}_2 \mathbf{W}_2'\|_{\mathrm{F}}, \\ \|\sin \Theta\|_{\mathrm{op}} &= \|\mathbf{W}_1 \mathbf{W}_1' - \mathbf{W}_2 \mathbf{W}_2'\|_{\mathrm{op}}. \end{aligned} \tag{18}$$

Overview Given the oracle sequence defined as above, we divide the proof into three steps. First, we show that the output of the oracle version of Algorithm 2 gives a good initial value for the oracle version of Algorithm 1. Next, we prove two recursive inequalities that characterize the evolution of the column subspaces of $\widetilde{\mathbf{U}}^{(t)}$ and $\widetilde{\mathbf{V}}^{(t)}$, and show that after T iterates, the output of the oracle version of Algorithm 1 estimates $\mathbf{M}$ well. Last but not least, we show that with high probability the oracle estimating sequence and the actual estimating sequence are identical up to $3T$ iterates and that $\widehat{T} \in [T, 3T]$. Therefore, the actual estimating sequence inherits all the nice properties that can be claimed for the oracle sequence.

In what follows, we carry out the three steps in order.

Initialization We first investigate the properties of $\widetilde{\mathbf{X}}$, $\widetilde{I}^0$, $\widetilde{J}^0$ and $\widetilde{\mathbf{V}}^{(0)}$.

Note that for any orthonormal matrix $\mathbf{W}$, $\mathbf{W}\mathbf{W}'$ gives the projection matrix onto $\mathrm{span}(\mathbf{W})$. The following lemma quantifies the difference between the leading singular structures of $\mathbf{X}$ and $\mathbf{M}$.

Lemma 9 *With probability at least* $1 - m^{-2}$,

$$\|\mathbf{U}\mathbf{U} - \widetilde{\mathbf{U}}\widetilde{\mathbf{U}}\|_{\mathrm{F}}, \|\mathbf{V}\mathbf{V} - \widetilde{\mathbf{V}}\widetilde{\mathbf{V}}\|_{\mathrm{F}} \leq \frac{\sqrt{2r}}{d_r} \left(\sqrt{k} + \sqrt{l} + 2\sqrt{\log m} \right), \tag{19}$$

and for any $i \in [n]$,

$$|\widetilde{d}_i - d_i| \leq \sqrt{k} + \sqrt{l} + 2\sqrt{\log m} = o(d_r), \tag{20}$$

where the last equality holds under Condition 1.

Proof By symmetry, we only need to spell out the arguments for $\mathbf{U}$ in (19). By definition, $\widetilde{\mathbf{X}} = \mathbf{U}\mathbf{D}\mathbf{V}' + \widetilde{\mathbf{Z}}$ where (after reordering of the rows and the columns) $\widetilde{\mathbf{Z}} = \begin{bmatrix} \mathbf{Z}_{IJ} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$. Thus, we have

$$\|\mathbf{U}\mathbf{U}' - \widetilde{\mathbf{U}}\widetilde{\mathbf{U}}'\|_{\mathrm{F}} \leq \sqrt{2r} \|\mathbf{U}\mathbf{U}' - \widetilde{\mathbf{U}}\widetilde{\mathbf{U}}'\|_{\mathrm{op}} \leq \frac{\sqrt{2r}}{d_r} \|\widetilde{\mathbf{Z}}\|_{\mathrm{op}}.$$

Here, the first inequality holds since $\mathsf{rank}(\mathbf{U}\mathbf{U}' - \widetilde{\mathbf{U}}\widetilde{\mathbf{U}}') \le 2r$ and the last inequality is due to Wedin's $\sin\theta$ theorem (Wedin, 1972). By the Davidson-Szarek bound (Davidson and Szarek, 2001), with probability at least $1 - m^{-2}$, $\|\widetilde{\mathbf{Z}}\|_{\mathrm{op}} = \|\mathbf{Z}_{IJ}\|_{\mathrm{op}} \le \sqrt{k} + \sqrt{l} + 2\sqrt{\log m}$. This completes the proof of (19).

On the other hand, Corollary 8.6.2 of Golub and Van Loan (1996) implies that $|\widetilde{d}_i - d_i| \le \|\mathbf{Z}_{IJ}\|_{\mathrm{op}}$. Together with the above discussion, we obtain the first inequality in (20). The second inequality is a direct consequence of Condition 1. This completes the proof. ■

Next, we investigate the properties of the sets selected in Algorithm 2. For some universal constants $0 < a_- < 1 < a_+$, define the following two deterministic sets

$$I^0_{\pm} = \left\{ i \in [m] : \|\mathbf{M}_{i*}\|^2 \ge a_{\mp}\alpha\sqrt{n\log m} \right\}, \quad J^0_{\pm} = \left\{ j \in [n] : \|\mathbf{M}_{*j}\|^2 \ge a_{\mp}\alpha\sqrt{m\log m} \right\}. \tag{21}$$

Lemma 10 *Let Condition 1 be satisfied, and let $\alpha \ge 4$, $a_- \le \frac{1}{20}$ and $a_+ \ge 2$ be fixed constants. For sufficiently large values of m and n, with probability at least $1 - O(m^{-2})$, we have $I_- \subseteq \widetilde{I}^0 \subseteq I_+$ and $J_- \subseteq \widetilde{J}^0 \subseteq J_+$, and so $I^0 = \widetilde{I}^0$ and $J^0 = \widetilde{J}^0$.*

Proof By symmetry, we only show the proof for $\widetilde{I}^0$ here. The arguments for $\widetilde{J}^0$ are similar. On the one hand, we have

$$\begin{aligned}
\mathbb{P}(I^0_- \nsubseteq \widetilde{I}^0) &\le \sum_{i \in I^0_-} \mathbb{P}\left(\|\mathbf{X}_{i*}\|^2 < n + \alpha\sqrt{n\log m} \right) \\
&\le m\,\mathbb{P}\left(\chi^2_n(a_+\alpha\sqrt{n\log m}) < n + \alpha\sqrt{n\log m} \right) \\
&\le m\,\exp\left(-\frac{(a_+ - 1)^2\alpha^2 n\log m}{4n + 8a_+\alpha\sqrt{n\log m}} \right) \\
&\le m\,\exp(-3\log m) = m^{-2}.
\end{aligned}$$

Here, the last inequality holds for fixed $a_+ \ge 2$, $\alpha \ge 4$ and all sufficiently large (m, n) such that $2a_+\alpha\sqrt{n\log m} \le n/3$, which is guaranteed by Condition 1.

On the other hand, for $x = \frac{(1-a_-)^2\alpha^2 n\log m}{(2.1)^2(n + 2a_-\sqrt{n\log m})}$, we have

$$\begin{aligned}
\mathbb{P}(\widetilde{I}^0 \nsubseteq I^0_+) &\le \sum_{i \in (I^0_+)^c} \mathbb{P}\left(\|\mathbf{X}_{i*}\|^2 > n + \alpha\sqrt{n\log m} \right) \\
&\le m\,\mathbb{P}\left(\chi^2_n(a_-\alpha\sqrt{n\log m}) > n + \alpha\sqrt{n\log m} \right) \\
&\le m\,\mathbb{P}\left(\chi^2_n(a_-\alpha\sqrt{n\log m}) > n + 2.1\sqrt{(n + 2\alpha_-\sqrt{n\log m})\,x} \right) \\
&\le m\,\mathbb{P}\left(\chi^2_n(a_-\alpha\sqrt{n\log m}) > n + 2\sqrt{(n + 2\alpha_-\sqrt{n\log m})x} + 2x \right) \\
&\le m\,\exp(-x) \\
&\le m\exp(-3\log m) = m^{-2}.
\end{aligned}$$

Here, the fourth inequality holds for fixed $\alpha \ge 4$, $a_- \le \frac{1}{20}$, and all sufficiently large (m, n) such that $n + 2a_-\sqrt{n\log m} \ge \frac{2}{0.21}(1 - a_-)^2\alpha\sqrt{n\log m}$. The last inequality holds when, in addition, $0.95^2 \cdot 16 \cdot n \ge 3 \cdot (2.1)^2 \cdot (n + 2a_-\sqrt{n\log m})$, which is again guaranteed by Condition 1.

Finally, when $I_- \subseteq \widetilde{I}^0 \subseteq I_+$, we have $I^0 = \widetilde{I}^0$ since $I_+ \subset I$. ■

The next lemma estimates the accuracy of the starting point $\widetilde{\mathbf{V}}^{(0)}$ for the oracle version of Algorithm 1.

Lemma 11 *Let Condition 1 be satisfied, and let $\alpha \ge 4$ and $a_+ \ge 2$ be fixed constants. For sufficiently large values of m and n, uniformly over $\mathcal{F}(m, n, k, l, r, d, \kappa)$, with probability at least $1 - O(m^{-2})$, for a positive constant C that depends only on κ, a_+ and α,*

$$\|\sin\widetilde{\Theta}^{(0)}_v\|_{\mathrm{F}} \le \frac{C}{d}\left[\left(r^2k^2 n\log m\right)^{1/4} + \left(r^2l^2 m\log m\right)^{1/4} \right] \le \frac{1}{6}.$$

Proof Let $\mathbf{X}^{(0)}$ be the matrix defined in Step 2 of Algorithm 2, but with I^0 and J^0 replaced by $\widetilde{I}^0$ and $\widetilde{J}^0$. Then we have

$$\|\sin\widetilde{\Theta}^{(0)}_v\|_{\mathrm{F}} = \frac{1}{\sqrt{2}}\|\widetilde{\mathbf{V}}^{(0)}\widetilde{\mathbf{V}}^{(0)} - \widetilde{\mathbf{V}}\widetilde{\mathbf{V}}'\|_{\mathrm{F}} \le \frac{\sqrt{2r}}{\sqrt{2}}\|\widetilde{\mathbf{V}}^{(0)}\widetilde{\mathbf{V}}^{(0)} - \widetilde{\mathbf{V}}\widetilde{\mathbf{V}}'\|_{\mathrm{op}} \le \frac{\sqrt{r}}{\widetilde{d}_r}\|\widetilde{\mathbf{X}} - \widetilde{\mathbf{X}}^{(0)}\|_{\mathrm{op}}.$$

Here, the first equality is from (18). The second inequality holds since $\mathsf{rank}(\widetilde{\mathbf{V}}^{(0)}\widetilde{\mathbf{V}}^{(0)} - \widetilde{\mathbf{V}}\widetilde{\mathbf{V}}') \le 2r$, and the last inequality is due to Wedin's $\sin\theta$ theorem (Wedin, 1972).

To further bound the rightmost side, we note that $\widetilde{\mathbf{X}}^{(0)}$ and $\widetilde{\mathbf{X}}$ are supported on $\widetilde{I}^0 \times \widetilde{J}^0$ and $I \times J$ respectively, with $\widetilde{I}^0 \times \widetilde{J}^0 \subset I \times J$. In addition, $(I \times J)\backslash(\widetilde{I}^0 \times \widetilde{J}^0)$ is the union of two disjoint subsets $(I\backslash\widetilde{I}^0) \times J$ and $\widetilde{I}^0 \times (J\backslash\widetilde{J}^0)$. Thus, the triangle inequality leads to

$$\begin{aligned}
\|\widetilde{\mathbf{X}} - \widetilde{\mathbf{X}}^{(0)}\|_{\mathrm{op}} &\le \|\widetilde{\mathbf{X}}_{I\backslash\widetilde{I}^0, J}\|_{\mathrm{op}} + \|\widetilde{\mathbf{X}}_{\widetilde{I}^0, J\backslash\widetilde{J}^0}\|_{\mathrm{op}} \\
&\le \|\mathbf{U}_{I\backslash\widetilde{I}^0,*}\mathbf{D}(\mathbf{V}_{J*})'\|_{\mathrm{op}} + \|\mathbf{U}_{\widetilde{I}^0*}\mathbf{D}(\mathbf{V}_{J\backslash\widetilde{J}^0*})'\|_{\mathrm{op}} + \|\mathbf{Z}_{I\backslash\widetilde{I}^0, J}\|_{\mathrm{op}} + \|\mathbf{Z}_{\widetilde{I}^0, J\backslash\widetilde{J}^0}\|_{\mathrm{op}}.
\end{aligned} \tag{22}$$

We now bound each of the four terms in (22) separately. For the first term, on the event such that the conclusion of Lemma 10 holds, we have

$$\|\mathbf{U}_{I\backslash\widetilde{I}^0,*}\mathbf{D}\mathbf{V}'_J\|_{\mathrm{op}} \le \|\mathbf{D}\|_{\mathrm{op}}\|\mathbf{V}_{J*}\|_{\mathrm{op}}\|\mathbf{U}_{I\backslash\widetilde{I}^0,*}\|_{\mathrm{op}} \le d_1\|\mathbf{U}_{I\backslash\widetilde{I}^0,*}\|_{\mathrm{F}} \le \frac{d_1}{d_r}(a_+\alpha)^{1/2}(k^2 n\log m)^{1/4}.$$

Here, the last inequality is due to $I^0_- \subset \widetilde{I}^0$, the definition of I^0_- in (21), and the facts that $\|\mathbf{M}_{i*}\| \ge d_r\|\mathbf{U}_{i*}\|$ for all $i \in [m]$ and that $|I\backslash\widetilde{I}^0| \le |I| \le k$. By similar argument, on the event such that the conclusion of Lemma 10 holds, we can bound the second term in (22) as

$$\begin{aligned}
\|\mathbf{U}_{\widetilde{I}^0*}\mathbf{D}(\mathbf{V}_{J\backslash\widetilde{J}^0*})'\|_{\mathrm{op}} &\le \|\mathbf{U}_{\widetilde{I}^0*}\|_{\mathrm{op}}\|\mathbf{D}\|_{\mathrm{op}}\|\mathbf{V}_{J\backslash\widetilde{J}^0*}\|_{\mathrm{op}} \le d_1\|\mathbf{U}\|_{\mathrm{op}}\|\mathbf{V}_{J\backslash\widetilde{J}^0*}\|_{\mathrm{F}} \\
&\le \frac{d_1}{d_r}(a_+\alpha)^{1/2}(l^2 m\log m)^{1/4}.
\end{aligned}$$

To bound the last two terms, we first note that on the event such that the conclusion of Lemma 10 holds, both terms are upper bounded by $\|\mathbf{Z}_{IJ}\|_{\mathrm{op}}$. Together with the Davidson–Szarek bound (Davidson and Szarek, 2001), this implies that with probability at least $1 - m^{-2}$,

$$\|\mathbf{Z}_{I\setminus\widetilde{I}^0,J}\|_{\mathrm{op}} + \|\mathbf{Z}_{\widetilde{I}^0,J\setminus\widetilde{J}^0}\|_{\mathrm{op}} \le 2\|\mathbf{Z}_{IJ}\|_{\mathrm{op}} \le 2\left(\sqrt{k} + \sqrt{l} + 2\sqrt{\log m}\right).$$

Assembling the last five displays and observe that $\widetilde{d}_r \ge 0.9 d_r$ for sufficiently large values of (m, n) on the event such that the conclusion of Lemma 9, we obtain the first inequality in the conclusion. The second inequality is a direct consequence of Condition 1. This completes the proof. ∎

Evolution We now study how the column subspaces of $\widetilde{\mathbf{U}}^{(t)}$ and $\widetilde{\mathbf{V}}^{(t)}$ evolve over iterations. To this end, let

$$\rho = \widetilde{d}_{r+1}/\widetilde{d}_r, \tag{23}$$

where $\widetilde{d}_i$ denotes the i^{th} singular value of $\widetilde{\mathbf{X}}$.

Proposition 12 *For any $t \ge 1$, let $x^t = \|\sin\Theta_u^{(t)}\|_{\mathrm{F}}$, $y^t = \|\sin\Theta_v^{(t)}\|_{\mathrm{F}}$. Moreover, define*

$$\omega_u = (2\widetilde{d}_r)^{-1}\sqrt{k\gamma_u^2}, \qquad \omega_v = (2\widetilde{d}_r)^{-1}\sqrt{l\gamma_v^2}, \qquad \omega = \omega_u \vee \omega_v. \tag{24}$$

Let Condition 1 be satisfied. Then for sufficiently large values of (m, n), on the event such that the conclusions of Lemmas 9–11 hold,

1) For any $t \ge 1$, if $y^{t-1} < 1$, then

$$x^t\sqrt{1-(y^{t-1})^2} \le \rho y^{t-1} + \omega_u, \qquad y^t\sqrt{1-(x^t)^2} \le \rho x^t + \omega_v. \tag{25}$$

2) For any $a \in (0, 1/2]$, if

$$y^{t-1} \le \frac{1.01\,\omega}{(1-a)(1-\rho)}, \tag{26}$$

then so is x^t. Otherwise,

$$x^t \le y^{t-1}[1 - a(1-\rho)]. \tag{27}$$

The same conclusions hold with the ordered pair (y^{t-1}, x^t) replaced by (x^t, y^t) in (26)–(27).

Proof 1) In what follows, we focus on showing the first inequality in (25). The second inequality follows from essentially the same argument.

Let $u^t = \|\sin\Phi_u^{(t)}\|_{\mathrm{F}}$. We first show that

$$u^t \le \frac{\rho y^{t-1}}{\sqrt{1-(y^{t-1})^2}}. \tag{28}$$

Recall the SVD of $\widetilde{\mathbf{X}}$ in (15). In addition, let the QR factorization of $\widetilde{\mathbf{U}}^{(t),\mathrm{mul}} = \widetilde{\mathbf{Q}}^{(t)}\widetilde{\mathbf{R}}^{(t),\mathrm{mul}}$. By definition, $\widetilde{\mathbf{U}}^{(t),\mathrm{mul}} = \widetilde{\mathbf{X}}\widetilde{\mathbf{V}}^{(t-1)}$. Premultiplying both sides by $\begin{bmatrix}\widetilde{\mathbf{U}} & \widetilde{\mathbf{U}}_\perp\end{bmatrix}'$, we obtain

$$\begin{bmatrix}\widetilde{\mathbf{D}} & \mathbf{0}\\ \mathbf{0} & \widetilde{\mathbf{D}}_\perp\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{V}}'\widetilde{\mathbf{V}}^{(t-1)}\\ (\widetilde{\mathbf{V}}_\perp)'\widetilde{\mathbf{V}}^{(t-1)}\end{bmatrix} = \begin{bmatrix}\widetilde{\mathbf{U}}'\widetilde{\mathbf{Q}}^{(t)}\\ (\widetilde{\mathbf{U}}_\perp)'\widetilde{\mathbf{Q}}^{(t)}\end{bmatrix}\widetilde{\mathbf{R}}^{(t),\mathrm{mul}}.$$

In addition, let

$$\begin{bmatrix}\widetilde{\mathbf{U}}'\widetilde{\mathbf{Q}}^{(t)}\\ (\widetilde{\mathbf{U}}_\perp)'\widetilde{\mathbf{Q}}^{(t)}\end{bmatrix} = \begin{bmatrix}\mathbf{O}^{(t)}\\ \mathbf{W}^{(t)}\end{bmatrix}.$$

By the last two displays, we have

$$\mathbf{W}^{(t)} = \widetilde{\mathbf{D}}_\perp(\widetilde{\mathbf{V}}_\perp)'\widetilde{\mathbf{V}}^{(t-1)}(\widetilde{\mathbf{R}}^{(t),\mathrm{mul}})^{-1} = \widetilde{\mathbf{D}}_\perp\left[(\widetilde{\mathbf{V}}_\perp)'\widetilde{\mathbf{V}}^{(t-1)}\right]\left[\widetilde{\mathbf{V}}'\widetilde{\mathbf{V}}^{(t-1)}\right]^{-1}\widetilde{\mathbf{D}}^{-1}\left[\widetilde{\mathbf{U}}'\widetilde{\mathbf{Q}}^{(t)}\right].$$

Thus,

$$\|\mathbf{W}^{(t)}\|_{\mathrm{F}} \le \|\widetilde{\mathbf{D}}_\perp\|_{\mathrm{op}}\|(\widetilde{\mathbf{V}}_\perp)'\widetilde{\mathbf{V}}^{(t-1)}\|_{\mathrm{F}}\|[\widetilde{\mathbf{V}}'\widetilde{\mathbf{V}}^{(t-1)}]^{-1}\|_{\mathrm{op}}\|\widetilde{\mathbf{D}}^{-1}\|_{\mathrm{op}}\|\widetilde{\mathbf{U}}\|_{\mathrm{op}}\|\widetilde{\mathbf{Q}}^{(t)}\|_{\mathrm{op}}.$$

By Corollary 5.5.4 of Stewart and Sun (1990), $\|\mathbf{W}^{(t)}\|_{\mathrm{F}} = u^t$, $\|(\widetilde{\mathbf{V}}_\perp)'\widetilde{\mathbf{V}}^{(t-1)}\|_{\mathrm{F}} = y^{t-1}$. Moreover, by Section 12.4.3 of Golub and Van Loan (1996), $\|[\widetilde{\mathbf{V}}'\widetilde{\mathbf{V}}^{(t-1)}]^{-1}\|_{\mathrm{op}} = 1/\cos\theta_{v,r}^{(t-1)} = 1/\sqrt{1-(\sin\theta_{v,r}^{(t-1)})^2} \le 1/\sqrt{1-(y^{t-1})^2}$. Here we have used the assumption that $y^{t-1} < 1$. Together with the facts that $\|\widetilde{\mathbf{D}}_\perp\|_{\mathrm{op}} = \widetilde{d}_{r+1}$, $\|\widetilde{\mathbf{D}}^{-1}\|_{\mathrm{op}} = \widetilde{d}_r^{-1}$, $\|\widetilde{\mathbf{U}}\|_{\mathrm{op}} = \|\widetilde{\mathbf{Q}}^{(t)}\|_{\mathrm{op}} = 1$, this leads to (28).

Next, we show that

$$x^t \le u^t + \frac{\omega_u}{\sqrt{1-(y^{t-1})^2}}. \tag{29}$$

To this end, let $w^t = \|\widetilde{\mathbf{Q}}^{(t)}(\widetilde{\mathbf{Q}}^{(t)})' - \widetilde{\mathbf{U}}^{(t)}(\widetilde{\mathbf{U}}^{(t)})'\|_{\mathrm{F}}$. Then, by (18) and the triangle inequality, we obtain

$$x^t \le u^t + \frac{1}{\sqrt{2}}w^t.$$

To bound w^t, note that Wedin's $\sin\theta$ theorem (Wedin, 1972) implies

$$w^t \le \frac{\|\widetilde{\mathbf{U}}^{(t),\mathrm{mul}} - \widetilde{\mathbf{U}}^{(t)}\|_{\mathrm{F}}}{\sigma_r(\widetilde{\mathbf{U}}^{(t),\mathrm{mul}})}.$$

In the oracle version, $\widetilde{\mathbf{U}}^{(t),\mathrm{mul}}$ has at most k nonzero rows, and so $\|\widetilde{\mathbf{U}}^{(t),\mathrm{mul}} - \widetilde{\mathbf{U}}^{(t)}\|_{\mathrm{F}} \le \sqrt{k\gamma_u^2}$. For any unit vector $\mathbf{y} \in \mathrm{span}(\widetilde{\mathbf{V}}^{(t-1)})$, decompose $\mathbf{y} = \mathbf{y}_0 + \mathbf{y}_1$ where $\mathbf{y}_0 \in \mathrm{span}(\widetilde{\mathbf{V}})$ and $\mathbf{y}_1 \in \mathrm{span}(\widetilde{\mathbf{V}}_\perp)$. Then by definition, $\|\mathbf{y}_0\| \ge \cos\theta_{v,1}^{(t-1)} \ge \sqrt{1-(y^{t-1})^2}$. Thus, for any unit vector $\mathbf{x}$, $\|\widetilde{\mathbf{U}}^{(t),\mathrm{mul}}\mathbf{x}\|^2 = \|\widetilde{\mathbf{X}}\mathbf{V}^{(t-1)}\mathbf{x}\|^2 = \|\widetilde{\mathbf{X}}\mathbf{y}\|^2 = \|\widetilde{\mathbf{X}}\mathbf{y}_0\|^2 + \|\widetilde{\mathbf{X}}\mathbf{y}_1\|^2 \ge \|\widetilde{\mathbf{X}}\mathbf{y}_0\|^2 = \|\widetilde{\mathbf{X}}\widetilde{\mathbf{V}}\widetilde{\mathbf{V}}'\mathbf{y}_0\|^2 \ge (\widetilde{d}_r)^2\|\mathbf{y}_0\|^2 \ge (\widetilde{d}_r)^2[1-(y^{t-1})^2]$. Hence,

$$\sigma_r(\widetilde{\mathbf{U}}^{(t),\mathrm{mul}}) \ge \inf_{\|\mathbf{x}\|=1}\|\widetilde{\mathbf{U}}^{(t),\mathrm{mul}}\mathbf{x}\| \ge \widetilde{d}_r\sqrt{1-(y^{t-1})^2}.$$

Assembling the last three display, we obtain (29). Finally, the first inequality in (25) comes from (28), (29) and the triangle inequality.

2) Given (25), we have

$$x^t \le \frac{\rho y^{t-1} + \omega}{\sqrt{1-(y^{t-1})^2}},$$

and that $y^0 \le \frac{1}{6} \le \frac{1}{5}(1-\rho)^2$ for sufficiently large values of (m,n) due to Condition 1 and Lemma 9. The proof of part (2) then follows from the same argument as in the proof of Proposition 6.1 in Ma (2013). ■

Convergence We say that the oracle sequence has *converged* if

$$x^t \vee y^t \le \frac{1.01\omega}{(1-m^{-1})(1-\rho)}. \tag{30}$$

This choice is motivated by the observation that $\frac{1.01\omega}{1-\rho}$ is the smallest possible value for x^t and y^t that Proposition 12 can lead to.

Proposition 13 *Let Condition 1 be satisfied and T be defined in (12). For sufficiently large values of (m,n), on the event such that the conclusions of Lemmas 9–11 hold, it takes at most T steps for the oracle sequence to converge in the sense of (30). For any t, let $\widetilde{\mathbf{P}}_u^{(t)} = \widetilde{\mathbf{U}}^{(t)}(\widetilde{\mathbf{U}}^{(t)})'$ and $\widetilde{\mathbf{P}}_v^{(t)} = \widetilde{\mathbf{V}}^{(t)}(\widetilde{\mathbf{V}}^{(t)})'$. Then there exists a constant C that depends only on κ, such that for all $t \ge T$,*

$$\|\widetilde{\mathbf{P}}_u^{(t)}\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{U}}\widetilde{\mathbf{D}}\widetilde{\mathbf{V}}\|_{\mathrm{F}}^2 \le C\left(k\gamma_u^2 + l\gamma_v^2\right).$$

Proof To prove the first claim, we rely on claim (2) of Proposition 12. Without loss of generality, assume that $m = 2^\nu$ for some integer $\nu \ge 1$. So $\nu = \log m/\log 2$. Let t_1 be the number of iterations needed to ensure that $x^t \vee y^t \le \frac{1.01\omega}{(1-\frac{1}{2})(1-\rho)}$. Note that when (26) does not hold, (27) ensures that

$$y^t \le y^{t-1}[1-a(1-\rho)]^2, \quad x^t \le x^{t-1}[1-a(1-\rho)]^2. \tag{31}$$

Thus, it suffices to have $\left[1-\frac{1}{2}(1-\rho)\right]^{2t_1} \ge \frac{1.01\omega}{(1-\frac{1}{2})(1-\rho)}$, i.e., $2t_1|\log(1-\frac{1}{2}(1-\rho))| \ge \log(1-\frac{1}{2})(1-\rho)/(1.01\omega)$. Since $|\log(1-x)| \ge x$ for all $x \in (0,1)$, it suffices to set

$$t_1 = \frac{1}{1-\rho}\log\frac{\frac{1}{2}(1-\rho)}{1.01\omega} = \frac{1+o(1)}{2}\log\left(\frac{d_r^2}{k\gamma_u^2 \vee l\gamma_v^2}\right).$$

Next, let $t_2 - t_1$ be the number of additional iterations needed to achieve $x^t \vee y^t \le 1.01\omega/[(1-\frac{1}{4})(1-\rho)]^2$. Before this is achieved, (31) is satisfied with $a = \frac{1}{4}$. So it suffices to have $[1-\frac{1}{4}(1-\rho)]^{2(t_2-t_1)} \le (1-\frac{1}{2})/(1-\frac{1}{4})$, which is guaranteed if $t_2 - t_1 \ge \frac{2}{1-\rho}[\log(1-\frac{1}{4}) - \log(1-\frac{1}{2})]$. Recursively, we define t_i for $i = 3,\dots,\nu$, such that $x^{t_i}, y^{t_i} \le 1.01\omega/[(1-2^{-i})(1-\rho)]$. Repeating the above argument shows that it suffices to have $t_i - t_{i-1} = \frac{2^{i-1}}{1-\rho}[\log(1-2^{-i}) - \log(1-2^{-(i-1)})]$ for $i = 3,\dots,\nu$. Therefore, if we let

$$t_\nu - t_1 = \frac{\nu+1/2}{2(1-\rho)} = \frac{(1+o(1))\log m}{2\log 2} \ge \sum_{i=1}^{\nu}\frac{2^{i-1}}{1-\rho}\left[\log(1-2^{-i}) - \log(1-2^{-(i-1)})\right],$$

then $x^t \vee y^t \le 1.01\omega/[(1-m^{-1})(1-\rho)]$ for all $t \ge t_\nu$. We complete the proof of the first claim by noting that $T \ge t_\nu$ for sufficiently large m, n under Condition 1.

To prove the second claim, let $\widetilde{\mathbf{P}}_u = \widetilde{\mathbf{U}}\widetilde{\mathbf{U}}'$ and $\widetilde{\mathbf{P}}_v = \widetilde{\mathbf{V}}\widetilde{\mathbf{V}}'$. Then we have

$$\begin{aligned}
\|\widetilde{\mathbf{P}}_u^{(t)}\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{U}}\widetilde{\mathbf{D}}\widetilde{\mathbf{V}}\|_{\mathrm{F}} &= \|\widetilde{\mathbf{P}}_u^{(t)}\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} && (32)\\
&\le \|(\widetilde{\mathbf{P}}_u^{(t)} - \widetilde{\mathbf{P}}_u)\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(t)}\|_{\mathrm{F}} + \|\widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}(\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{P}}_v)\|_{\mathrm{F}}\\
&\le \|\widetilde{\mathbf{P}}_u^{(t)} - \widetilde{\mathbf{P}}_u\|_{\mathrm{F}}\|\widetilde{\mathbf{X}}\|_{\mathrm{op}}\|\widetilde{\mathbf{P}}_v^{(t)}\|_{\mathrm{op}} + \|\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{P}}_v\|_{\mathrm{F}}\|\widetilde{\mathbf{X}}\|_{\mathrm{op}}\|\widetilde{\mathbf{P}}_u\|_{\mathrm{op}}\\
&= \widetilde{d}_1\left(\|\widetilde{\mathbf{P}}_u^{(t)} - \widetilde{\mathbf{P}}_u\|_{\mathrm{F}} + \|\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{P}}_v\|_{\mathrm{F}}\right) && (33)\\
&\le C\left(\sqrt{k\gamma_u^2} + \sqrt{l\gamma_v^2}\right). && (34)
\end{aligned}$$

Here, the equality (32) is due to the definitions of $\widetilde{\mathbf{P}}_u$, $\widetilde{\mathbf{P}}_v$ and the fact that $\widetilde{\mathbf{U}}$, $\widetilde{\mathbf{D}}$ and $\widetilde{\mathbf{V}}$ consist of the first r singular values and vectors of $\widetilde{\mathbf{X}}$. The equality (33) holds since $\|\widetilde{\mathbf{X}}\|_{\mathrm{op}} = \widetilde{d}_1$ and $\|\widetilde{\mathbf{P}}_u\|_{\mathrm{op}} = \|\widetilde{\mathbf{P}}_v^{(t)}\|_{\mathrm{op}} = 1$ as both are projection matrices. Finally, the inequality (34) holds since $\|\widetilde{\mathbf{P}}_u^{(t)} - \widetilde{\mathbf{P}}_u\|_{\mathrm{F}} = \sqrt{2}x^t$ and $\|\widetilde{\mathbf{P}}_v^{(t)} - \widetilde{\mathbf{P}}_v\|_{\mathrm{F}} = \sqrt{2}y^t$ due to (18), the definitions in (24) and (30), and the fact that on the event such that (20) holds, $\widetilde{d}_1/\widetilde{d}_r \le 2\kappa$ when m and n are sufficiently large. This completes the proof. ■

Remark 14 *It is worth noting that the conclusions of Proposition 12 and Proposition 13 hold for any $\gamma_u > 0$ and $\gamma_v > 0$, though they will be used later with the specific choice of γ_u and γ_v in (10).*

Proof of Upper Bounds We are now in the position to prove Theorem 4. To this end, we need to establish the equivalence between the oracle and the actual estimating sequences. The following lemma shows that with high probability, the oracle sequence and the actual sequence are identical up to $3T$ iterates.

Lemma 15 *Let γ_u and γ_v be defined as in (10) with some fixed constant $\beta \ge 4$ and let Condition 1 be satisfied. For sufficiently large m and n, with probability at least $1-O(m^{-2})$, for all $1 \le t \le 3T$, $\mathbf{U}_{I^c*}^{(t)} = \mathbf{0}$, $\mathbf{V}_{J^c*}^{(t)} = \mathbf{0}$, and so $\mathbf{U}^{(t)} = \widetilde{\mathbf{U}}^{(t)}$ and $\mathbf{V}^{(t)} = \widetilde{\mathbf{V}}^{(t)}$.*

Proof First of all, by Lemma 10, with probability at least $1-O(m^{-2})$, $J^0 = \widetilde{J}^0 \subset J_+ \subset J$, and so $\widetilde{\mathbf{V}}^{(0)} = \mathbf{V}^{(0)}$. Define event $E^{(0)} = \{\mathbf{V}^{(0)} = \widetilde{\mathbf{V}}^{(0)}\}$.

We now focus on the first iteration. Define event

$$E_u^{(1)} = \left\{\|\mathbf{Z}_{i*}\widetilde{\mathbf{V}}^{(0)}\| < \gamma_u,\ \forall i \in I^c\right\}.$$

On $E^{(0)} \cap E_u^{(1)}$, for any $i \in I^c$, $\mathbf{U}_{i*}^{(1),\mathrm{mul}} = \mathbf{X}_{i*}\mathbf{V}^{(0)} = \mathbf{Z}_{i*}\widetilde{\mathbf{V}}^{(0)}$. Thus, $\|\mathbf{U}_{i*}^{(1),\mathrm{mul}}\| < \gamma_u$ and so $\mathbf{U}_{i*}^{(1),\mathrm{thr}} = \mathbf{0}$ for all $i \in I^c$. This further implies $\mathbf{U}_{I^c*}^{(1)} = \mathbf{0}$ and $\mathbf{U}^{(1)} = \widetilde{\mathbf{U}}^{(1)}$. Further define event

$$E_v^{(1)} = \left\{\|(\mathbf{Z}_{*j})'\widetilde{\mathbf{U}}^{(1)}\| < \gamma_v,\ \forall j \in J^c\right\}.$$

Then by similar argument, on the event $E^0 \cap E_u^{(1)} \cap E_v^{(1)}$, we have $\mathbf{V}_{J^c *}^{(1)} = \mathbf{0}$ and $\mathbf{V}^{(1)} = \widetilde{\mathbf{V}}^{(1)}$.

We now bound the probability of $(E_u^{(1)})^c$. Without loss of generality, let $J \subset [l]$. Note that for any $j \in J$, $i \in I^c$, $\widetilde{\mathbf{V}}^{(0)}$ depends on Z_{ij} only through $\|\mathbf{Z}_{I^c j}\|^2$ in the selection of $\widetilde{J}^0$ in the oracle version of Algorithm 2. Therefore, $\widetilde{\mathbf{V}}^{(0)}$ is independent of $\frac{Z_{ij}}{\|\mathbf{Z}_{I^c j}\|}$. Let $k' = |I^c|$ and $Y_1, \dots Y_l$ be i.i.d. $\chi_{k'}$ random variables independent of $\mathbf{Z}$. For any $i \in I^c$ and $j \in [l]$, let

$$\check{Z}_{ij} = Y_j \frac{Z_{ij}}{\|\mathbf{Z}_{I^c j}\|},$$

and $\check{\mathbf{Z}}_{i[l]} = (\check{Z}_{i1}, \dots, \check{Z}_{il}) \in \mathbb{R}^{1 \times l}$. Since $\mathrm{supp}(\widetilde{\mathbf{V}}^{(0)}) \subset J \subset [l]$ on the event $E^{(0)}$, we obtain that for any $i \in I^c$, $\mathbf{Z}_{i*}\widetilde{\mathbf{V}}^{(0)} = \mathbf{Z}_{i[l]}\widetilde{\mathbf{V}}^{(0)} = \check{\mathbf{Z}}_{i[l]}\widetilde{\mathbf{V}}^{(0)} + (\mathbf{Z}_{i[l]} - \check{\mathbf{Z}}_{i[l]})\widetilde{\mathbf{V}}^{(0)}$. Thus,

$$\|\mathbf{Z}_{i*}\widetilde{\mathbf{V}}^{(0)}\| \le \|\check{\mathbf{Z}}_{i[l]}\widetilde{\mathbf{V}}^{(0)}_{[l]*}\| + \|(\mathbf{Z}_{i[l]} - \check{\mathbf{Z}}_{i[l]})\widetilde{\mathbf{V}}^{(0)}_{[l]*}\| \le \|\check{\mathbf{Z}}_{i[l]}\widetilde{\mathbf{V}}^{(0)}_{[l]*}\| + \|\mathbf{Z}_{i[l]} - \check{\mathbf{Z}}_{i[l]}\|\|\widetilde{\mathbf{V}}^{(0)}_{[l]*}\|_{\mathrm{op}}.$$

For the first term on the rightmost side, since $\check{\mathbf{Z}}_{i[l]}$ is independent of $\widetilde{\mathbf{V}}^{(0)}$, $\|\check{\mathbf{Z}}_{i[l]}\widetilde{\mathbf{V}}^{(0)}_{[l]*}\|^2 \sim \chi_r^2$, and so by Lemma 17, with probability at least $1 - O(m^{-\beta})$,

$$\|\check{\mathbf{Z}}_{i[l]}\widetilde{\mathbf{V}}^{(0)}_{[l]*}\|^2 \le r + 2\sqrt{\beta r \log m} + 2\beta \log m.$$

For the second term, we first note that $\|\widetilde{\mathbf{V}}^{(0)}_{[l]*}\|_{\mathrm{op}} = 1$ since it has orthonormal columns. Moreover, $\check{\mathbf{Z}}_{i[l]} - \mathbf{Z}_{i[l]} = \mathbf{Z}_{i[l]} \,\mathrm{diag}\left(\frac{Y_1}{\|\mathbf{Z}_{I^c 1}\|} - 1, \dots, \frac{Y_l}{\|\mathbf{Z}_{I^c l}\|} - 1\right)$. Thus,

$$\|\check{\mathbf{Z}}_{i[l]} - \mathbf{Z}_{i[l]}\| \le \|\mathbf{Z}_{i[l]}\| \max_{j \in [l]} \left| \frac{Y_j}{\|\mathbf{Z}_{I^c j}\|} - 1 \right|.$$

By Lemma 17, with probability at least $1 - O(m^{-\beta})$,

$$\|\mathbf{Z}_{i[l]}\|^2 \le l + 2\sqrt{\beta l \log m} + 2\beta \log m.$$

By Lemma 18, for any $j \in [l]$, with probability at least $1 - O(m^{-(\beta+1)})$,

$$\left| \frac{Y_j}{\|\mathbf{Z}_{I^c j}\|} - 1 \right| \le \left| \frac{Y_j^2}{\|\mathbf{Z}_{I^c j}\|^2} - 1 \right| \le 4 \cdot 1.01 \cdot \sqrt{\frac{(\beta+1)\log m}{k'}}.$$

Here, the last inequality holds for sufficient large values of m and n, since Condition 1 implies that $(\log m)/k' = o(1)$. By the union bound, with probability at least $1 - O(m^{-\beta})$, for sufficient large values of m and n,

$$\|\check{\mathbf{Z}}_{i[l]} - \mathbf{Z}_{i[l]}\| \le 0.01\sqrt{\log m},$$

since Condition 1 ensures that $l/k' = o(1)$. Assembling the last six displays, we obtain that for any $\beta > 1$, with probability at least $1 - O(m^{-\beta})$,

$$\|\mathbf{Z}_{i*}\widetilde{\mathbf{V}}^{(0)}\| \le \sqrt{r + 2\sqrt{\beta r \log m} + 2\beta \log m} + 0.01\sqrt{\log m} \le \gamma_v.$$

Applying the union bound again, we obtain that when $\beta \ge 4$ in (10),

$$\mathbb{P}\left\{(E_u^{(1)})^c\right\} \le O(m^{-3}). \tag{35}$$

Similarly, for any $j \in J^c$, $\widetilde{\mathbf{U}}^{(1)}$ depends on Z_{ij} only through $\|\mathbf{Z}_{iJ^c}\|$. Therefore, by analogous arguments, we also obtain (35) for $(E_v^1)^c$ with any fixed $\beta \ge 4$.

Turn to subsequent iterations, we further define events

$$E_u^{(t)} = \left\{\|\mathbf{Z}_{i*}\widetilde{\mathbf{V}}^{(t-1)}\| < \gamma_u,\, \forall i \in I^c\right\}, \quad E_v^{(t)} = \left\{\|(\mathbf{Z}_{*j})'\widetilde{\mathbf{U}}^{(t)}\| < \gamma_v,\, \forall j \in J^c\right\}, \quad t = 2, \dots, 3T.$$

Iterating the above arguments, we obtain that on the event $E^{(0)} \cap (\cap_{t=1}^{3T} E_u^{(t)}) \cap (\cap_{t=1}^{3T} E_v^{(t)})$, $\mathbf{U}_{I^c *}^{(t)} = \mathbf{0}$, $\mathbf{V}_{J^c *}^{(t)} = \mathbf{0}$, and so $\mathbf{U}^{(t)} = \widetilde{\mathbf{U}}^{(t)}$ and $\mathbf{V}^{(t)} = \widetilde{\mathbf{V}}^{(t)}$. Moreover, by similar argument to that for (35), we can bound each $\mathbb{P}\{(E_u^{(t)})^c\}$ and $\mathbb{P}\{(E_v^{(t)})^c\}$ by $O(m^{-3})$ for all $t = 2, \dots, 3T$ with any fixed $\beta \ge 4$ in (10). Finally, under Condition 1, $T = O(m)$, and so

$$\mathbb{P}\left\{E^{(0)} \cap (\cap_{t=1}^{3T} E_u^{(t)}) \cap (\cap_{t=1}^{3T} E_v^{(t)})\right\} = 1 - O(m^{-2}).$$

This completes the proof. ■

Lemma 16 *Let $\widehat{T}$ be defined in (11). With probability at least $1 - O(m^{-2})$, $T \le \widehat{T} \le 3T$.*

Proof By definition (10) and (12), we have

$$T \le \frac{1.01}{2}\left(\frac{\log m}{\log 2} + \log \frac{d_r^2}{\gamma^2}\right).$$

On the other hand, note that $1/\log 2 \ge 1.44$ and that $\log(k \vee l) \le \log m$ under the assumption that $m \ge n$, and hence

$$T \ge \frac{1.01}{2}\left(\frac{\log m}{\log 2} - \log m + \log \frac{d_r^2}{\gamma^2}\right) \ge \frac{1.01}{2}\left(0.44 \log m + \log \frac{d_r^2}{\gamma^2}\right).$$

On the other hand, on the event such that the conclusions of Lemmas 9–11 hold, we have

$$\begin{aligned} |d_r^{(0)} - d_r| &\le |d_r^{(0)} - \widetilde{d}_r| + |\widetilde{d}_r - d_r| \\ &= |\widetilde{d}_r^{(0)} - \widetilde{d}_r| + |\widetilde{d}_r - d_r| \\ &\le \|\widetilde{\mathbf{X}}^{(0)} - \widetilde{\mathbf{X}}\|_{\mathrm{op}} + o(d_r) \\ &= o(d_r). \end{aligned}$$

Hence for sufficiently large values of m and n, $\log \gamma^2 > 1$ and with probability at least $1 - O(m^{-2})$, $|\log(d_r^{(0)})^2 / \log d_r^2 - 1| \le 0.01$. When the above inequalities all hold, we obtain $\widehat{T} \in [T, 3T]$. ■

We are now in the position to prove Theorem 4.

Proof [Proof of Theorem 4] Note that on the events such that the conclusions of Lemmas 9–16 hold, we have

$$\begin{aligned}
&\|\widehat{\mathbf{M}} - \mathbf{M}\|_{\mathrm{F}} \\
&\quad = \|\widehat{\mathbf{P}}_u^{(\widehat{T})}\mathbf{X}\widehat{\mathbf{P}}_v^{(\widehat{T})} - \mathbf{U}\mathbf{D}\mathbf{V}'\|_{\mathrm{F}} \\
&\quad \le \|\widehat{\mathbf{P}}_u^{(\widehat{T})}\mathbf{X}\widehat{\mathbf{P}}_v^{(\widehat{T})} - \widetilde{\mathbf{U}}\widetilde{\mathbf{D}}\widetilde{\mathbf{V}}'\|_{\mathrm{F}} + \|\widetilde{\mathbf{U}}\widetilde{\mathbf{D}}\widetilde{\mathbf{V}}' - \mathbf{U}\mathbf{D}\mathbf{V}'\|_{\mathrm{F}} \\
&\quad = \|\widetilde{\mathbf{P}}_u^{(\widehat{T})}\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(\widehat{T})} - \widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} + \|\widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v - \mathbf{P}_u\mathbf{M}\mathbf{P}_v\|_{\mathrm{F}} \\
&\quad \le \|\widetilde{\mathbf{P}}_u^{(\widehat{T})}\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(\widehat{T})} - \widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} + \|\widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v - \widetilde{\mathbf{P}}_u\mathbf{M}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} \\
&\qquad + \|\widetilde{\mathbf{P}}_u\mathbf{M}\widetilde{\mathbf{P}}_v - \mathbf{P}_u\mathbf{M}\mathbf{P}_v\|_{\mathrm{F}}.
\end{aligned}$$

Here, the first and the second inequalities are both due to the triangle inequality. The second equality is due to Lemma 16 and the facts that $\mathrm{supp}(\widetilde{\mathbf{U}}^{(t)}) \subset I$, $\mathrm{supp}(\widetilde{\mathbf{V}}^{(t)}) \subset J$ and that $\widetilde{\mathbf{U}}$ and $\widetilde{\mathbf{V}}$ collect the first r left and right singular vectors of $\widetilde{\mathbf{X}}$.

We now bound each of the three terms on the rightmost side of the last display. First, on the event such that the conclusions of Proposition 13 and Lemma 16 hold, we have

$$\|\widetilde{\mathbf{P}}_u^{(\widehat{T})}\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v^{(\widehat{T})} - \widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} \le C\sqrt{k\gamma_u^2 + l\gamma_v^2}.$$

Next, by similar argument to that leading to the conclusion of Lemma 11, with probability at least $1 - O(m^{-2})$

$$\begin{aligned}
\|\widetilde{\mathbf{P}}_u\widetilde{\mathbf{X}}\widetilde{\mathbf{P}}_v - \widetilde{\mathbf{P}}_u\mathbf{M}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} &\le \|\widetilde{\mathbf{X}} - \mathbf{M}\|_{\mathrm{F}} = \|\mathbf{Z}_{IJ}\|_{\mathrm{F}} \\
&\le \sqrt{r}\|\mathbf{Z}_{IJ}\|_{\mathrm{op}} \\
&\le \sqrt{r}(\sqrt{k} + \sqrt{l} + 2\sqrt{\log m}).
\end{aligned}$$

Last but not least,

$$\begin{aligned}
&\|\widetilde{\mathbf{P}}_u\mathbf{M}\widetilde{\mathbf{P}}_v - \mathbf{P}_u\mathbf{M}\mathbf{P}_v\|_{\mathrm{F}} \\
&\quad \le \|(\widetilde{\mathbf{P}}_u - \mathbf{P}_u)\mathbf{M}\widetilde{\mathbf{P}}_v\|_{\mathrm{F}} + \|\mathbf{P}_u\mathbf{M}(\widetilde{\mathbf{P}}_v - \mathbf{P}_v)\|_{\mathrm{F}} \\
&\quad \le d_1(\|\widetilde{\mathbf{P}}_u - \mathbf{P}_u\|_{\mathrm{F}} + \|\widetilde{\mathbf{P}}_v - \mathbf{P}_v\|_{\mathrm{F}}) \\
&\quad \le \kappa\sqrt{r}(\sqrt{k} + \sqrt{l} + 2\sqrt{\log m}).
\end{aligned}$$

Assembling the last four displays, we complete the proof for the case of Frobenius norm, i.e., $q = 2$. To obtain the result for all $q \in [1, 2)$, simply note that for any matrix $\mathbf{A}$, $\|\mathbf{A}\|_{\mathbb{S}_q} \le (\mathsf{rank}(\mathbf{A}))^{\frac{1}{q}-\frac{1}{2}}\|\mathbf{A}\|_{\mathrm{F}}$ and that $\mathsf{rank}(\widetilde{\mathbf{M}} - \mathbf{M}) \le 2r$. This completes the proof. ■

6.3 Proof of Proposition 7

Proof [Proof of Proposition 7] Without loss of generality, assume that $\sigma = 1$. We first show that $\widehat{r} \le r$ with probability at least $1 - O(m^{-2})$. To this end, note that

$$\begin{aligned}
\mathbb{P}\{\widehat{r} > r\} &= \mathbb{P}\left\{\sigma_{r+1}(\mathbf{X}_{I^0J^0}) > \delta_{|I^0||J^0|}\right\} \\
&\le \mathbb{P}\left\{\max_{|A|=|I^0|,|B|=|J^0|} \sigma_{r+1}(\mathbf{X}_{AB}) > \delta_{|A||B|}\right\} \\
&\le \sum_{i=r+1}^{m}\sum_{j=r+1}^{n} \mathbb{P}\left\{\max_{|A|=i,|B|=j} \sigma_{r+1}(\mathbf{X}_{AB}) > \delta_{ij}\right\}.
\end{aligned}$$

By the interlacing property of singular values, we know that for $\mathbf{Z}$, a $m \times n$ standard Gaussian random matrix,

$$\max_{|A|=i,|B|=j} \sigma_{r+1}(\mathbf{X}_{AB}) \overset{st}{<} \max_{|A|=i-r,|B|=j-r} \sigma_1(\mathbf{Z}_{AB}) \overset{st}{<} \max_{|A|=i,|B|=j} \sigma_1(\mathbf{Z}_{AB}),$$

where $\overset{st}{<}$ means stochastically smaller. Together with the union bound, this implies

$$\begin{aligned}
\mathbb{P}\left\{\max_{|A|=i,|B|=j} \sigma_{r+1}(\mathbf{X}_{AB}) > \delta_{ij}\right\} &\le \binom{m}{i}\binom{n}{j}\mathbb{P}\{\sigma_1(\mathbf{Z}_{AB}) > \delta_{ij}\} \\
&\le \left(\frac{em}{i}\right)^i\left(\frac{en}{j}\right)^j \exp\left(-i\log\frac{em}{i} - j\log\frac{em}{j} - 4\log m\right) \\
&= m^{-4}.
\end{aligned}$$

Here, the second inequality is due to $\binom{p}{k} \le (ep/k)^k$ for any $k \in [p]$ and the Davidson-Szarek bound (Davidson and Szarek, 2001). As $n \le m$ under Condition 1, we obtain

$$\mathbb{P}\{\widehat{r} > r\} \le \sum_{i=r+1}^{m}\sum_{j=r+1}^{n} m^{-4} \le m^{-2}.$$

To show that $\widehat{r} \ge r$ with probability at least $1 - O(m^{-2})$, we note that on the event such that the conclusions of Lemmas 9–11 hold, $\sigma_r(\mathbf{X}_{I^0J^0}) = \sigma_r(\widetilde{\mathbf{X}}^0) = \widetilde{d}_r^{(0)}$. So by the triangle inequality, the conclusion of Lemma 9 and the proof of Lemma 11, we obtain that

$$\sigma_r(\mathbf{X}_{I^0J^0}) = \widetilde{d}_r^{(0)} \ge d_r - \|\mathbf{X} - \widetilde{\mathbf{X}}\|_{\mathrm{op}} - \|\widetilde{\mathbf{X}} - \widetilde{\mathbf{X}}^0\|_{\mathrm{op}} \ge d_r/4 > \delta_{kl},$$

where the second last and the last inequalities hold under Condition 1 for sufficiently large values of m and n. Note that on the event such that the conclusion of Lemma 10 holds, we have $|I^0| \le k$ and $|J^0| \le l$ and so $\delta_{|I^0||J^0|} \le \delta_{kl}$. This completes the proof. ■

Acknowledgments

Some preliminary results of this paper have been presented at the 2013 Allerton Conference as an invited paper. The research of Z.M. is supported in part by NSF Career Award DMS-1352060.

Appendix A. Appendix

Lemma 17 (Lemma 8.1 in Birgé (2001)) *Let X follow the non-central chi square distribution $\chi^2_\nu(\delta)$ with degrees of freedom ν and non-centrality parameter $\delta \geq 0$. Then for any $x > 0$,*

$$\mathbb{P}\left\{X \geq \nu + \delta + 2\sqrt{(\nu+2\delta)x} + 2x\right\} \leq \mathrm{e}^{-x},$$
$$\mathbb{P}\left\{X \leq \nu + \delta - 2\sqrt{(\nu+2\delta)x}\right\} \leq \mathrm{e}^{-x}.$$

Lemma 18 *Let X and Y be two independent χ^2_ν random variables. Then for any $x > 0$,*

$$\mathbb{P}\left\{\left|\frac{X}{Y} - 1\right| \leq \frac{4\sqrt{x/\nu}(1+\sqrt{x/\nu})}{1 - 2\sqrt{x/\nu}}\right\} \geq 1 - 4\mathrm{e}^{-x}.$$

Proof By the triangle inequality,

$$\left|\frac{X}{Y} - 1\right| \leq \frac{1}{|Y|}\left(|X-\nu| + |Y-\nu|\right).$$

By Lemma 17, for any $x > 0$, each of the following holds with probability at least $1 - 2\mathrm{e}^{-x}$:

$$|X-\nu| \leq 2\sqrt{\nu x} + 2x,$$
$$|Y-\nu| \leq 2\sqrt{\nu x} + 2x, \quad \text{and} \quad |Y| \geq \nu - 2\sqrt{\nu x}.$$

Assembling the last two displays, we complete the proof. ■

References

L. Birgé. *An alternative point of view on Lepski's method*, volume 36 of *Lecture Notes-Monograph Series*, pages 113–133. Institute of Mathematical Statistics, 2001.

F. Bunea, Y. She, and M.H. Wegkamp. Optimal selection of reduced rank estimators of high-dimensional matrices. *The Annals of Statistics*, 39(2):1282–1309, 2011.

F. Bunea, Y. She, and M.H. Wegkamp. Joint variable and rank selection for parsimonious estimation of high dimensional matrices. *The Annals of Statistics*, 40:2359–2388, 2012.

C. Butucea and Yu.I. Ingster. Detection of a sparse submatrix of a high-dimensional noisy matrix. *Bernoulli*, 19(5B):2652–2688, 2013.

T.T. Cai, Z. Ma, and Y. Wu. Sparse PCA: Optimal rates and adaptive estimation. *The Annals of Statistics*, 41(6):3074–3110, 2013a.

T.T. Cai, Z. Ma, and Y. Wu. Optimal estimation and rank detection for sparse spiked covariance matrices. *Probability Theory and Related Fields*, 161(3-4):781–815, 2013b.

T.T. Cai, X. Li, and Z. Ma. Optimal rates of convergence for noisy sparse phase retrieval via thresholded Wirtinger flow. *arXiv preprint arXiv:1506.03382*, 2015.

E.J. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6): 925–936, 2010.

E.J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

G. Chen, P.F. Sullivan, and M.R. Kosorok. Biclustering with heterogeneous variance. *Proceedings of the National Academy of Sciences*, 110(30):12253–12258, 2013.

K.R. Davidson and S. Szarek. *Handbook on the Geometry of Banach Spaces*, volume 1, chapter Local operator theory, random matrices and Banach spaces, pages 317–366. Elsevier Science, 2001.

D.L. Donoho and M. Gavish. Minimax risk of matrix denoising by singular value thresholding. *The Annals of Statistics*, 42(6):2413–2440, 2014.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, 2001.

C. Gao, Y. Lu, Z. Ma, and H. H. Zhou. Optimal estimation and completion of matrices with biclustering structures. *arXiv preprint arXiv:1512.00150*, 2015.

G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins University Press, 3rd edition, 1996.

R.H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *The Journal of Machine Learning Research*, 11:2057–2078, 2010.

V. Koltchinskii, K. Lounici, and A.B. Tsybakov. Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion. *The Annals of Statistics*, 39(5):2302–2329, 2011.

K. Lee, Y. Wu, and Y. Bresler. Near optimal compressed sensing of sparse rank-one matrices via sparse power factorization. *arXiv preprint arXiv:1312.0525*, 2013.

M. Lee, H. Shen, J.Z. Huang, and J.S. Marron. Biclustering via sparse singular value decomposition. *Biometrics*, 66:1087–1095, 2010.

X. Li and V. Voroninski. Sparse signal recovery from quadratic measurements via convex programming. *SIAM Journal on Mathematical Analysis*, 45(5):3019–3033, 2013.

K. Lounici, M. Pontil, S. Van De Geer, and A. B. Tsybakov. Oracle inequalities and optimal inference under group sparsity. *The Annals of Statistics*, 39(4):2164–2204, 2011.

Z. Ma. Sparse principal component analysis and iterative thresholding. *The Annals of Statistics*, 41(2):772–801, 2013.

Z. Ma and T. Sun. Adaptive sparse reduced-rank regression. *arXiv preprint arXiv:1403.1922*, 2014.

Z. Ma and Y. Wu. Volume ratio, sparsity, and minimaxity under unitarily invariant norms. *IEEE Transactions on Information Theory*, To appear, 2015.

S. Negahban and M.J. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *The Annals of Statistics*, 39(2):1069–1097, 2011.

S. Oymak and B. Hassibi. Asymptotically exact denoising in relation to compressed sensing. *arXiv preprint arXiv:1305.2714*, 2013.

S. Oymak, A. Jalali, M. Fazel, and B. Hassibi. Noisy estimation of simultaneously structured models: Limitations of convex relaxation. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 6019–6024. IEEE, 2013.

S. Oymak, A. Jalali, M. Fazel, Y.C. Eldar, and B. Hassibi. Simultaneously structured models with application to sparse and low-rank matrices. *IEEE Transactions on Information Theory*, 61(5):2886–2908, 2015.

A.A. Shabalin, V.J. Weigman, C.M. Perou, and A.B. Nobel. Finding large average submatrices in high dimensional data. *The Annals of Applied Statistics*, 3:985–1012, 2009.

Y. Shechtman, Y.C. Eldar, A. Szameit, and M. Segev. Sparsity based sub-wavelength imaging with partially incoherent light via quadratic compressed sensing. *Optics Express*, 19(16):14807–14822, 2011.

G.W. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Computer science and scientific computing. Academic Press, 1990.

X. Sun and A.B. Nobel. On the maximal size of large-average and anova-fit submatrices in a gaussian random matrix. *Bernoulli*, 19(1):275–294, 2013.

P.-A. Wedin. Perturbation bounds in connection with singular value decomposition. *BIT*, 12:99–111, 1972.

D. Yang, Z. Ma, and A. Buja. A sparse singular value decomposition method for high-dimensional data. *Journal of Computational and Graphical Statistics*, 23(4):923–942, 2014.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B*, 68(1):49–67, 2006.

X.-T. Yuan and T. Zhang. Truncated power method for sparse eigenvalue problems. *Journal of Machine Learning Research*, 14:899–925, 2013.

C.H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010.

Journal of Machine Learning Research 17 (2016) 1-50 Submitted 5/15; Revised 11/15; Published 6/16

Hierarchical Relative Entropy Policy Search

Christian Daniel[1] DANIEL@IAS.TU-DARMSTADT.DE
Gerhard Neumann[1] NEUMANN@IAS.TU-DARMSTADT.DE
Oliver Kroemer[1] KROEMER@IAS.TU-DARMSTADT.DE
Jan Peters[1,2] PETERS@IAS.TU-DARMSTADT.DE

[1]*Technische Universität Darmstadt*
Fachbereich Informatik , Fachgruppe Intelligente Autonome Systeme
Hochschulstraße 10
64289 Darmstadt
Germany

[2]*Max-Planck-Institut für Intelligente Systeme*
Spemannstraße 38
72076 Tübingen
Germany

Editor: Laurent Orseau

Abstract

Many reinforcement learning (RL) tasks, especially in robotics, consist of multiple sub-tasks that are strongly structured. Such task structures can be exploited by incorporating hierarchical policies that consist of gating networks and sub-policies. However, this concept has only been partially explored for real world settings and complete methods, derived from first principles, are needed. Real world settings are challenging due to large and continuous state-action spaces that are prohibitive for exhaustive sampling methods. We define the problem of learning sub-policies in continuous state action spaces as finding a hierarchical policy that is composed of a high-level gating policy to select the low-level sub-policies for execution by the agent. In order to efficiently share experience with all sub-policies, also called inter-policy learning, we treat these sub-policies as latent variables which allows for distribution of the update information between the sub-policies. We present three different variants of our algorithm, designed to be suitable for a wide variety of real world robot learning tasks and evaluate our algorithms in two real robot learning scenarios as well as several simulations and comparisons.

Keywords: Reinforcement Learning, Policy Search, Hierarchical Learning, Robot Learning, Motor Skill Learning, Robust Learning, Structured Learning, Temporal Correlation, HiREPS, REPS

1. Introduction

Employing robots in unpredictable environments, such as in hospitals, disaster sites or households, requires robotic agents to autonomously learn new tasks and adapt to new environments. Such robots will need to acquire new skills through trial and error, also known as reinforcement learning (Sutton and Barto, 1998), as well as being able to generalize their skills to solve a large variety of tasks. However, successful implementation of reinforcement learning (RL) methods on real robot tasks is challenging for multiple reasons. Most importantly, real robots have high dimensional and continuous state-action spaces which is difficult to deal with for many RL methods. Furthermore, evaluations on real robots are resource intensive and, hence, the methods need to be sample efficient. Methods should also not fully explore the state-action space due to the risk of damaging the robot or its environment. Finally, real robot RL does not allow resetting of the state to arbitrary initial conditions as is required for many RL methods.

While presenting additional challenges, robot tasks also offer some advantages over traditional RL settings as many real world motor tasks are heavily structured. Exploiting the environment's structure can drastically simplify the learning problem. First, the solutions of many tasks lie within a tube of the solution space (Peters and Schaal, 2006), such that learning can be given a head start by demonstrating a sub-optimal solution. Local RL methods can, subsequently, be used to improve upon this demonstration. Second, the motor commands for many motor tasks exhibit strong temporal correlations. Such correlations allow for a hierarchical decomposition of the task into a sequence of elemental movements, often also referred to movement primitives (Schaal et al., 2003; Ijspeert and Schaal, 2003), options (Sutton et al., 1999b) or motor templates (Neumann and Peters, 2009). For example, a tennis game can be decomposed into a sequence of single strokes, e.g., a backhand strokes, forehand strokes, lobs, volleys and a serve. Finally, many motor tasks can be solved in multiple, often incompatible, ways. Identifying and representing such solutions as separate sub-policies to be used by the agent increases the versatility as well as the robustness of the learned policy. Additionally, it simplifies the use of local learning methods. In this paper, we extend our work on a robot learning framework (Daniel et al., 2012a,b), (Daniel et al., 2013) that can take advantage of such structured environments by identifying multiple sub-policies for a given task and learning to adapt, sequence and combine these sub-policies.

We base our learning algorithm on the Relative Entropy Policy Search (REPS) algorithm (Peters et al., 2010). In REPS, the exploitation-exploration trade-off is balanced by bounding the loss of information between policy updates. Such a bound results in a smoother and more stable learning process that avoids wild exploration of the state action space, as required by the robotics domain. We extend the REPS algorithm such that we can use sub-policies as building blocks of a hierarchical policy to exploit the temporal correlations inherent to many tasks. We formulate the problem of inferring such a hierarchical policy as a latent variable estimation problem, where the index of the sub-policy that has generated a given action is modeled as a latent variable. The policy update is now performed in two steps. In the Expectation step, we compute the responsibilities of the latent variables, i.e., the probabilities that the individual sub-policies have generated the observed state action pairs. The sub-policies are kept fixed in the E-step. Subsequently, the responsibilities are used to update the hierarchical policy containing the sub-policies with the REPS algorithm in the Maximization step. We show that such an Expectation-Maximization (EM) algorithm (Dempster et al., 1977) improves a lower bound of the original REPS optimization problem and that the lower bound is tight after each expectation step.

Our algorithm also allows for using prior knowledge to constrain the structure of the estimated hierarchical policy. For example, we often want the sub-policies to represent individual, distinct solutions. In many cases, these solutions may be incompatible, i.e., the solution space may not be convex. Therefore, the set of sub-policies should be separable in the solution space. Adding a constraint that avoids such an overlap ensures that the policy search algorithm does not average over multiple modes of the solution space, which may result in a poor performance of the resulting policy (Neumann, 2011).

We will discuss different variants of the Hierarchical REPS (HiREPS) approach with increasing complexity. We start our discussion with the contextual, continuous-armed bandit case where the episode ends after executing all actions belonging to one sub-policy. sub-policies are selected according to a gating policy $\pi(o|\boldsymbol{s})$ and define a distribution over the action given the state for a predetermined time horizon. Subsequently, we show how to extend this framework to the finite horizon setting, where the agent sequences multiple sub-policies to achieve his goal. Finally, we present the infinite horizon formulation of HiREPS, where the agent keeps executing sub-policies until either the task is fulfilled or a random reset occurs.

1.1 Related Work

Policy search (PS) is a class of RL methods which have been shown to fulfill the requirements of robot RL and is widely used in real robot learning. In applications, PS methods are often preferred over other traditional RL methods such as value-based methods, since they do not require the explicit estimation of a value function. Estimating a value function often requires the agent to fill the state and action space with samples, which is infeasible in robot RL. Therefore, most PS methods are commonly local methods and improve only locally on an initial, sub-optimal solution.

Particular properties of PS methods have been highlighted by several papers. Bagnell and Schneider (2003) show their strong convergence guarantees, Sutton et al. (1999a) discuss their application in combination with function approximators and Stone (2001) improved the possibilities of incorporating domain knowledge. Backing up the theoretical strength of policy search methods, several authors have demonstrated impressive application results. Bagnell and Schneider (2001) use policy search methods to autonomously control helicopters, Rosenstein (2001) learns robot weight-lifting, Kohl and Stone (2003) demonstrates learning of a quadrupedal walking behavior and Kober et al. (2008) successfully learn the ball in a cup (also known as Kendama) game. Endo et al. (2008) show the application of PS methods on a biped locomotion and Kormushev et al. (2010) PS to learn how to flip a pan-cake with a robot arm.

Important advances in the area of policy search methods have included pair-wise policy comparisons (Strens, 2003), policy gradient methods (Baxter and Bartlett, 2001; Sutton et al., 1999a) and natural policy gradient methods (Bagnell and Schneider, 2003; Peters and Schaal, 2006). More recently, Kober et al. (2008) presented probabilistic policy search approaches based on expectation maximization. Using EM like methods for reinforcement learning was initially pioneered by Dayan and Hinton (1997). Toussaint et al. (2006) showed how an EM like method can be used not only for MDPs but also partially observed MDPs. Deisenroth (2010) developed a model-based policy search method based on probabilistic modeling. Theodorou et al. (2010) showed a policy improvement algorithm inspired by path integrals and Peters et al. (2010) proposed a PS method that limits the loss of information between policy updates. Recently, Levine and Abbeel (2014) and Schulman et al. (2015) have shown how policy search methods can be combined with neural networks to learn complicated tasks in high dimensional domains. As these results show, current methods are well suited to learn single tasks in isolation. Nevertheless, they frequently fail to scale up to more complex motor tasks as they cannot exploit the hierarchical structure inherent to many tasks.

A common way of representing a hierarchical task structure is to use the concept of options. Options are temporally extended actions and were first introduced by Sutton et al. (1999a) as a way of reducing task complexity. They consist of three components, an action-selection policy, an initiation set that defines in which states an option can be initiated and a termination condition

which defines the probability of terminating an option in a given state. Typically, options extend the action space of the agent, i.e., the agent can still take the primitive actions which are active for one time step, and, in addition, the agent can choose the options as temporally extended actions. The use of options can significantly improve the learning speed as the options can be used to connect parts of the state space that are otherwise only reachable with a long action sequence (Sutton et al., 1999a). Consequently, the state space becomes easier to explore and, hence, the learning problem gets simplified. Formally, learning with options can be formulated as learning a Semi-Markov Decision Process (SMDP). SMDPs are not fully Markovian, as the selected option does not only depend on the current state, but also on the active option in the previous time step.

Unlike in the simplest setup where the set of options is given and fixed during learning, it is often also desirable to learn useful options for a given task. In this case, the data efficiency for learning the options can be significantly improved by leveraging all information collected during the execution of one option, also denoted as intra-option learning (Sutton et al., 1998). The state transitions generated by a certain option can also be used to update other options in an off-policy learning setup. Levy and Shimkin (2012) proposed a different view of intra-option learning by augmenting the state space with the option-termination probability. In that augmented state space, the option selection policy and termination condition may be represented by orthogonal basis functions and is optimized individually by standard policy gradient methods.

Options are also used in many hierarchical RL approaches where they are often extended to subtasks. A sub-task also contains an individual reward function such that we can learn the policy of the subtask. Dietterich (2000) proposed the MAXQ framework that uses several layers of subtasks. In contrast to the traditional option framework, the policies of the subtasks can also choose to execute a new subtask instead of just a primitive action. The subtasks are given by an individual reward function per subtask, a termination condition and a set of actions or other subtasks that can be executed. The policy of the subtasks is learned by the MAXQ-Q learning algorithm that learns the optimal value function for each subtask. However, the structure of the subtasks, i.e., the individual reward functions, must be specified by the user. How to define such subtasks for complex robot motor tasks or even learn the structure from data remains an open question. Additionally, value function methods are less applicable in continuous state action domains and not well suited for robotic tasks. Barto et al. (2004) used an 'intrinsic motivation' mechanism to define the reward signal of each subtask. A naturally curious agent is exploring its environment and whenever it discovers an unexpected change in its environment, it creates a new subtask in its internal representation that tries to reproduce this unexpected event. Hence, the agent incrementally builds up a set of skills that help the agent to explore its environment more efficiently. After this exploration phase, new tasks can be learned more efficiently by using the learned set of skills.

The notion of subtasks has also been used in continuous environments. Morimoto and Doya (2001) proposed a hierarchical RL setup where the subtask is defined by reaching a specific joint configuration of the robot. In this work, the set of subtasks, i.e., desired joint configurations, is given and the robot learns to reach the individual joint configurations as well as to choose a reachable next desired joint configuration. Ghavamzadeh and Mahadevan (2003) used a policy gradient approach to learn the policies for the individual subtasks while a value based method is used to select the next subtask. Policy gradient approaches work well in continuous environments but often converge slowly. In both approaches, the subtasks have been specified by the user and could not be modified by the learning algorithm. Hence, both approaches are limited to simpler robots where appropriate subtasks can be hand-tuned. In Konidaris and Barto (2009), the structure of the subtasks is also

learned by discovering the initiation set, the termination set and the option policy for new options. The options are then be chained together to solve the overall task. However, the option discovery algorithm is to date still limited to rather simple agents, such as a ball navigating in a maze.

The approach which is probably most closely related to ours is Bayesian policy search with hierarchical policy priors (Wingate et al., 2011). In this approach, the probabilistic formulation of policy search (Kober et al., 2008) is used and a hierarchical Bayesian prior is used for the policy parameters. Similarly to our approach, the activations of options are modeled as latent variables in their model that are estimated by MCMC sampling methods. While the use of hierarchical Bayesian priors is a promising idea, the approach is restricted to directly learn on the trajectory data instead of state actions pairs, which might lead to inefficient policy updates. In addition, the approach cannot be extended straightforwardly to complex high dimensional robots as the structure of the options is limited.

Another approach to simplifying the hierarchical RL problem in continuous action spaces is to restrict the space of possible trajectories by using parametrized policies, often also called movement primitives (Schaal et al., 2003), motion templates (Neumann and Peters, 2009) or parametrized skills (Da Silva et al., 2012). Learning the option policy now reduces to learning an appropriate parameter vector for the option. In Neumann and Peters (2009), the agent learned to sequence such parametrized policies, i.e., it learned the correct order of the parametrized policy as well as the single parameter vectors of the policy. In total, the agent has to learn fewer decisions as opposed to directly learning with primitive actions. However, learning is often also more difficult as the effect of an inaccurate decision can be much more costly than for primitive actions. While Neumann and Peters (2009) used value function approximation to evaluate the quality of the chosen parameter vectors of the options, Stulp and Schaal (2012) directly used the reward to come as evaluation. This reward to come was subsequently used by the Policy Improvement by Path Integrals (PI^2) algorithm (Theodorou et al., 2010) to learn the option policy.

Parametrized options or skills were also used for skill transfer, i.e., generalizing the parameters of the options to new tasks (Kupcsik et al., 2014). Thomas and Barto (2012) showed how the structure of motor primitives for continuous state action tasks can be learned and later be used to accelerate learning of a similar task. Da Silva et al. (2012) as well as Kober et al. (2010b) generalized parametrized skills for reasonably similar tasks drawn from a task distribution. They do not only use the concept of an option for a single policy, but instead allow each option to adapt to a subset of the task space by smoothly changing the parameter vector of the option. Hence, individual options are responsible for areas of the task space that are locally smooth. Multiple options together span discontinuous areas of the task space. The proposed framework consists of a pipeline of machine learning tools to identify the individual smooth lower dimensional manifolds as well as to generalize and improve the policy parameters. Alternative approaches for task generalization of parametrized options include learning a mixture of experts (Mülling et al., 2013) and Gaussian processes (Ude et al., 2010). The method presented in this paper is applicable to both primitive actions as well as movement primitives.

In the remainder of the paper we will explain how to obtain at a hierarchical formulation of the relative entropy policy search (REPS) method. We treat the problem of learning a hierarchical policy as a latent variable estimation problem. For the policy update, we assume that we can only observe the resulting actions of the old policy. The underlying hierarchy is unknown and, therefore, unobserved. The resulting algorithm is closely related to expectation maximization (EM) for latent variable models. We prove that such EM mechanisms can be incorporated in the information theo-

retic regularization of the REPS algorithm in order to get a lower bound of the original optimization problem which can, subsequently, be optimized in closed form. Furthermore, we introduce an additional constraint into the optimization problem that bounds the uncertainty of identifying a sub-policy given an action. As a consequence, the sub-policies are separated in the action space, which results in finding more versatile solutions and also alleviates the problem of averaging over several modes in the solution space, which is present in many current policy search algorithms as shown in Neumann (2011). In the evaluation section, we compare our algorithm to other state-of-the-art methods on benchmark problems and evaluate our algorithm on real world robotic applications.

2. Background on Information Theoretic Policy Search

As our algorithm is based on information theoretic policy search (Peters et al., 2010; Daniel et al., 2012a), we will quickly review the most relevant concepts of information theoretic policy search in the non-hierarchical learning setup. In policy search, an agent tries to maximize the expected return by adapting a parametrized policy. Formally, in a Markov decision process setting, the agent is in a state $\boldsymbol{s} \in \mathbb{S}$ and chooses an action $\boldsymbol{a} \in \mathbb{A}$ to execute. Given state $\boldsymbol{s}$ and action $\boldsymbol{a}$, the agent transfers to a next state $\boldsymbol{s}'$ in accordance with a transition probability distribution $p(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a}) = \mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}$. For each transition, the agent also receives a reward $r \in \mathbb{R}$ that depends on $\boldsymbol{s}$ and $\boldsymbol{a}$, which we also write as $\mathcal{R}_{\boldsymbol{s}\boldsymbol{a}}$. The agent chooses actions $\boldsymbol{a}$ in the current state $\boldsymbol{s}$ according to a policy $\pi(\boldsymbol{a}|\boldsymbol{s})$. We will consider an average reward setting where the goal of the agent is to find an optimal policy that will maximize the average reward

$$J(\pi) = \mathbb{E}[\mathcal{R}_{\boldsymbol{s}\boldsymbol{a}}] = \iint \mu^{\pi}(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})\mathcal{R}_{\boldsymbol{s}\boldsymbol{a}}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \tag{1}$$

where $\mu^{\pi}(\boldsymbol{s})$ is the state visit distribution of policy π.

Information-theoretic policy search was introduced with the relative entropy policy search (REPS) algorithm (Peters et al., 2010). We will start by defining a constrained optimization problem for solving the discussed average reward reinforcement learning setting and, subsequently, add an information-theoretic constraint to make the optimization problem feasible.

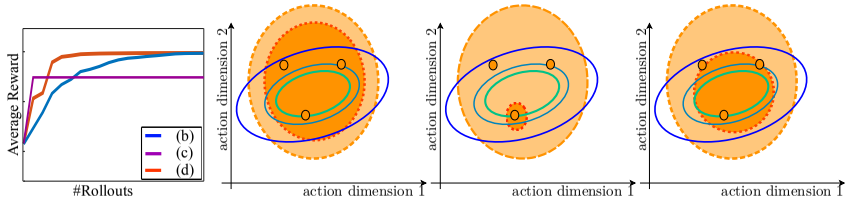
Constraints for the State Distribution. The objective of our optimization problem is given in Equation (1) which is supposed to be maximized with respect to $\mu^{\pi}(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})$. The agent can not freely choose the state distribution $\mu^{\pi}(\boldsymbol{s})$ of the policy, since $\mu^{\pi}(\boldsymbol{s})$ needs to comply with the policy $\pi(\boldsymbol{a}|\boldsymbol{s})$ and the system dynamics $\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}$, i.e.,

$$\forall \boldsymbol{s}' : \mu^{\pi}(\boldsymbol{s}') = \int_{\boldsymbol{s},\boldsymbol{a}} \mu^{\pi}(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}.$$

However, direct matching of the state probabilities is not feasible in continuous state spaces. Instead, we introduce state features $\boldsymbol{\phi}(\boldsymbol{s})$ and only match the feature averages

$$\int \boldsymbol{\phi}(\boldsymbol{s}')\mu^{\pi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}' = \iiint \mu^{\pi}(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}\,\mathrm{d}\boldsymbol{s}'. \tag{2}$$

For example, if we use all linear and squared terms of $\boldsymbol{s}$ in our feature vector $\boldsymbol{\phi}(\boldsymbol{s})$, we match the mean and the variance under both distributions. Additionally, we require the joint probability



(a) Learning Curves (b) Update not greedy enough (c) Update too greedy (d) Balanced Exploration

Figure 1: Schematic sketch of the behavior of policy updates. Solid lines show the contours of a quadratic reward function. The orange shaded area delimited by the dashed lines illustrates the area within two times the standard deviation of the sampling policy and the orange dots represent samples from the original policy. The darker shaded area delimited by the dotted line indicates the the possible policy update. The policy update needs to be balanced, such that it converges quickly to the local optimum, while not converging too fast to miss it. Such a balance is chosen by the user by specifying the relative entropy bound. Due to the relative entropy bound, the step size of the update is invariant to the task, the reward function or the parametrization of the policy. a) Learning curves corresponding to illustrations b-c. b) The update is not greedy enough, the policy will take too long to converge. c) The update is too greedy, the policy will not find the optimal solution. d) The update balances exploration and exploitation such that the policy quickly converges to a local optimum.

$p(\boldsymbol{s}, \boldsymbol{a}) = \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})$ to define a probability distribution, i.e., its integral has to evaluate to

$$1 = \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a}. \tag{3}$$

While the optimization criterion given in Equation (1) with constraints in Equations (2,3) describes the general average reward reinforcement learning problem, it does not consider that we have typically estimated $\mathcal{R}_{\boldsymbol{sa}}$ and $\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{ss}'}$ from a limited amount of data. Hence, we do not want the agent to 'jump' to the optimum of this problem but instead balance exploitation versus exploration. In REPS (Peters et al., 2010), this exploitation-exploration trade off is balanced by the insight that the loss of information in the policy updates should be limited. Independently, such regularization was also suggested and motivated from different perspectives by other authors. Still and Precup (2011) showed that the policy update with maximum information gain results in a similar solution and Azar et al. (2012) motivated a similar update as punishing the distance between the controlled system and the uncontrolled one. Finally, Rawlik et al. (2012) showed that even previous probabilistic policy search approaches are closely related. All these arguments have lead to similar solutions despite their different motivations and the resulting algorithms work well on benchmark problems.

Staying Close to the Data by Information Theoretic Constraints. As motivated by Peters et al. (2010), one key feature of effective PS methods is to limit the loss of information between policy updates, i.e., while we want to maximize the average reward of the new policy, we also want to stay close to the 'data', i.e., the state action distribution $q(\boldsymbol{s}, \boldsymbol{a})$ of the old policy. Staying close to the

data can be achieved by limiting the Kullback-Leibler (KL) divergence between the observed data $q(\boldsymbol{s}, \boldsymbol{a})$ and the next policy, i.e.,

$$\epsilon \geq D_{\mathrm{KL}}\left(\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s}) \,||\, q(\boldsymbol{s}, \boldsymbol{a})\right) = \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s}) \log \frac{\mu(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})}{q(\boldsymbol{s}, \boldsymbol{a})} \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a}. \tag{4}$$

Maximizing Equation (1) under the constraints of Equations (2, 3, 4) yields the optimization problem that defines the REPS method (Peters et al., 2010) which is the basis for our hierarchical policy search algorithm. The REPS optimization problem allows for a closed form solution for $\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})$ that can be derived by the method of Lagrangian multipliers. It is given by

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s}) \propto q(\boldsymbol{s}, \boldsymbol{a}) \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} + \mathbb{E}\left[\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}')|\boldsymbol{s}, \boldsymbol{a}\right] - \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s})}{\eta}\right), \tag{5}$$

where η and $\boldsymbol{\theta}$ are Lagrangian parameters that are obtained by optimizing the dual function

$$g(\eta, \boldsymbol{\theta}) = \eta \log \iint q(\boldsymbol{s}, \boldsymbol{a}) \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} + \mathbb{E}[V(\boldsymbol{s}')] - V(\boldsymbol{s})}{\eta}\right) \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a} + \eta\epsilon, \tag{6}$$

of the original optimization problem. If we interpret the term $V(\boldsymbol{s}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s})$ as value function linear in the parameters $\boldsymbol{\theta}$, the term $\mathcal{R}_{\boldsymbol{sa}} + \mathbb{E}\left[V(\boldsymbol{s}')\right] - V(\boldsymbol{s})$ can be viewed as advantage function. The Lagrangian parameter η is a scaling factor for the advantage. It becomes the temperature of the soft-max distribution defined in Equation (5) such that the KL-bound from Equation (4) is met.

Sample-Based REPS. As the reward function and the old distribution $q(\boldsymbol{s}, \boldsymbol{a})$ can have an arbitrary structure, the integral contained in the dual function can typically not be obtained in closed form. However, the reward function can typically be evaluated on samples from $q(\boldsymbol{s}, \boldsymbol{a})$ and these samples can be used to approximate the dual function g. As a result, we can only evaluate the distribution $\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})$ on a finite set of samples $\boldsymbol{s}_i$ and $\boldsymbol{a}_i$ with $i = 1 \ldots N$.

In order to sample new actions $\boldsymbol{a}$ in the next iteration of the algorithm, we need to find a parametric representation $\tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}; \boldsymbol{\beta})$ of the policy that approximates the distribution $\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)$, where $\boldsymbol{\beta}$ denotes the parameter vector of the policy. To do so, we aim to minimize the expected Kullback-Leibler divergence between $\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)$ and the next parametric policy $\tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}; \boldsymbol{\beta})$, i.e.,

$$\begin{aligned}
&\arg\min_{\boldsymbol{\beta}} \int p(\boldsymbol{s}) D_{\mathrm{KL}}\left(\pi(\boldsymbol{a}|\boldsymbol{s}) \,||\tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}; \boldsymbol{\beta})\right) \, \mathrm{d}\boldsymbol{s} \\
&= \arg\min_{\boldsymbol{\beta}} \int p(\boldsymbol{s}) \int \pi(\boldsymbol{a}|\boldsymbol{s}) \log \frac{\pi(\boldsymbol{a}|\boldsymbol{s})}{\tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}; \boldsymbol{\beta})} \, \mathrm{d}\boldsymbol{a} \, \mathrm{d}\boldsymbol{s} \\
&\approx \arg\max_{\boldsymbol{\beta}} \sum_i \frac{p(\boldsymbol{s}_i, \boldsymbol{a}_i)}{q(\boldsymbol{s}_i, \boldsymbol{a}_i)} \log \tilde{\pi}(\boldsymbol{a}_i|\boldsymbol{s}_i; \boldsymbol{\beta}) + \mathrm{const} \\
&= \arg\max_{\boldsymbol{\beta}} \sum_i w_i \log \tilde{\pi}(\boldsymbol{a}_i|\boldsymbol{s}_i; \boldsymbol{\beta}),
\end{aligned} \tag{7}$$

with

$$w_i = \exp\left(\frac{r(\boldsymbol{s}_i, \boldsymbol{a}_i) + \mathbb{E}\left[\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}')\middle|\, \boldsymbol{s}_i, \boldsymbol{a}_i\right] - \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}_i)}{\eta}\right). \tag{8}$$

Since the available samples are drawn from the distribution $q(\boldsymbol{s}, \boldsymbol{a})$, and not $p(\boldsymbol{s}, \boldsymbol{a})$, we need to perform importance sampling and divide by the sampling distribution $q(\boldsymbol{s}, \boldsymbol{a})$. Equation (7) is equivalent to computing the weighted maximum likelihood estimator for $\boldsymbol{\beta}$. In Section 4.3 we discuss how to compute the expectation over next states $\mathbb{E}\left[\boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}') \middle| \boldsymbol{s}_i, \boldsymbol{a}_i\right]$.

Illustration of Information Theoretic Policy Search. We illustrate the concept of information theoretic policy search on a two-dimensional toy problem with a quadratic reward function in Figure 1(a). Here, the algorithm typically starts with a broad explorative policy. Given a set of samples generated using this policy, we update the policy such that the exploitation-exploration trade-off is well balanced. This trade-off is chosen by the relative entropy bound of the REPS algorithm. Due to the relative entropy bound, the step-size of the policy update is largely independent of the task, the reward function, or the representation of the policy. Changing the policy only by a small step leads to further exploration as we stay close to the old exploration policy as illustrated in Figure 1(b). In contrast, a large KL divergence for the policy update greedily jumps to the best observed samples with very little further exploration, see Figure 1(c). This trade-off has to be chosen by the user by specifying the bound ϵ for the KL-divergence. Typically, we want to achieve a moderate exploration-exploitation trade-off as illustrated in Figure 1(d).

3. Learning Hierarchical Policies for Real Robot Reinforcement Learning

In this section, we will extend the REPS optimization problem to the hierarchical case, where the agent learns a hierarchical policy that is based on both a gating network and sub-policies. In order to obtain an efficient update rule for the hierarchical policy, we will rely on several insights detailed below. We will use these insights to derive three different learning algorithms that can be employed in different scenarios.

Hierarchical Policies. More complex tasks often require multiple sub-policies, such as a forehand stroke, backhand stroke, smash or lob in tennis. Different sub-policies are appropriate in different states of the environment. In order to model a policy consisting of several sub-policies, we use a hierarchical policy $\pi(\boldsymbol{a}|\boldsymbol{s})$ which consists of a set of sub-policies $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ and a gating network $\pi(o|\boldsymbol{s})$ that selects the currently active sub-policy. Thus, the hierarchical policy can be represented as

$$\pi(\boldsymbol{a}|\boldsymbol{s}) = \sum_{o \in \mathcal{O}} \pi(o|\boldsymbol{s}) \pi(\boldsymbol{a}|\boldsymbol{s}, o). \tag{9}$$

In order to determine the action $\boldsymbol{a}$ in state $\boldsymbol{s}$, we first sample a sub-policy from the gating network $\pi(o|\boldsymbol{s})$ and, subsequently, sample the action $\boldsymbol{a}$ from the specific sub-policy $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$. The benefit of representing multiple solutions has already been made evident in recent research results (Calinon et al., 2013; Daniel et al., 2012a).

Options as Latent Variables. Sharing of experience between sub-policies, known as inter-option learning, is an important property of a hierarchical learning algorithm for data-efficient learning. To realize inter-option learning, we treat the problem of learning sub-policies as a latent variable problem, i.e., we assume to observe only state-action samples $\{\boldsymbol{s}, \boldsymbol{a}\}$, but not the index of the generating sub-policy o. Instead, we infer the latent structure of the hierarchical policy that has generated the re-weighted samples. Expectation-maximization based methods can be used to iteratively estimate the responsibilities $p(o|\boldsymbol{s}, \boldsymbol{a})$ of the sub-policies for each sample $\{\boldsymbol{s}, \boldsymbol{a}\}$ and, subsequently, update

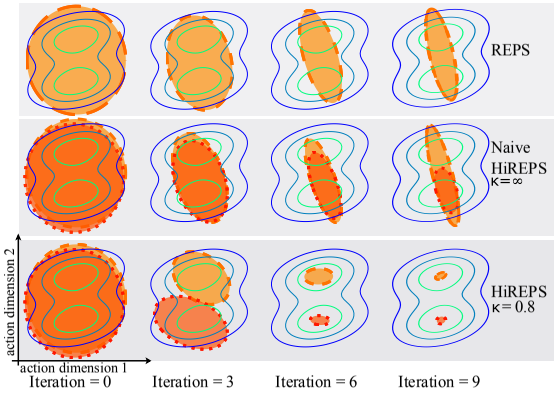


Figure 2: Schematic sketch of the behavior of REPS, and HiREPS with ($\tilde{\kappa} = 0.8$) and without ($\tilde{\kappa} = \infty$) bounding of the gating's entropy. A two-dimensional, bi-modal reward function is shown in the contour plot lines. Two sub-policies are shown in dashed and dotted outlines respectively (outlines show 95% confidence boundaries). REPS and naive HiREPS average over both modes. Constrained HiREPS separates the sub-policies and is therefore able to find both modes.

the sub-policies where the influence of each sample $\{\boldsymbol{s}, \boldsymbol{a}\}$ for the update of the sub-policy $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ is weighted by the responsibility $p(o_j|\boldsymbol{s}, \boldsymbol{a})$. Hence, the generated experience can be shared between the sub-policies. In Section 3.1, we integrate such an expectation-maximization mechanism into the REPS algorithm.

Learning Versatile Solutions. Being able to represent multiple solutions does not force the learning algorithm to find different solutions. Thus, without further constraints, we are likely to find multiple sub-policies that concentrate on the same mode of the solution space. Versatility of sub-policies can be achieved if the sub-policies are clearly separated in the state-action space. To enforce this separation of the sub-policies, we limit the expected change in the entropy H of the responsibilities of the sub-policies, i.e.,

$$\kappa \geq \frac{\mathbb{E}_{\boldsymbol{s},\boldsymbol{a}}[H\left(p(o|\boldsymbol{s}, \boldsymbol{a})\right)]}{\mathbb{E}_{q(\boldsymbol{s},\boldsymbol{a})}[H\left(q(o|\boldsymbol{s}, \boldsymbol{a})\right)]} = \frac{\iint \mu^{\pi}(\boldsymbol{s}) \pi(\boldsymbol{a}|\boldsymbol{s}) \sum_{o \in \mathcal{O}} p(o|\boldsymbol{s}, \boldsymbol{a}) \log p(o|\boldsymbol{s}, \boldsymbol{a}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a}}{\iint q(\boldsymbol{s}, \boldsymbol{a}) \sum_{o \in \mathcal{O}} q(o|\boldsymbol{s}, \boldsymbol{a}) \log q(o|\boldsymbol{s}, \boldsymbol{a}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a}}, \tag{10}$$

where $\mathbb{E}_{q(\boldsymbol{s},\boldsymbol{a})}[H\left(q(o|\boldsymbol{s}, \boldsymbol{a})\right)]$ is a constant and we write $\tilde{\kappa} = \mathbb{E}_{q(\boldsymbol{s},\boldsymbol{a})}[H\left(q(o|\boldsymbol{s}, \boldsymbol{a})\right)]\kappa$ to simplify the notation, such that the constraint reads

$$\tilde{\kappa} \geq \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}}[H\left(p(o|\boldsymbol{s}, \boldsymbol{a})\right)] = -\iint \mu^{\pi}(\boldsymbol{s}) \pi(\boldsymbol{a}|\boldsymbol{s}) \sum_{o \in \mathcal{O}} p(o|\boldsymbol{s}, \boldsymbol{a}) \log p(o|\boldsymbol{s}, \boldsymbol{a}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a}. \tag{11}$$

A high entropy of the responsibility $p(o|\boldsymbol{s}, \boldsymbol{a})$ indicates a high uncertainty in deciding which sub-policy has generated the state action pair, which implies that several sub-policies do overlap in the parameter space. By limiting the entropy, such overlapping is avoided and we ensure that different sub-policies concentrate on different and separate solutions.

We will discuss three settings for Hierarchical REPS (HiREPS). In the episodic setup, a single sub-policy is executed per episode, but the algorithm can choose between multiple sub-policies and adapt these to the current context. Subsequently, we will show how to sequence a fixed number of sub-policies with a finite horizon MDP formulation. Finally, we will use a infinite horizon MDP formulation to learn how to sequence a possibly infinite number of skills.

Illustration of Hierarchical Learning. We illustrate the advantage of learning with multiple sub-policies on a toy task with a two dimensional action space and a bi-modal reward function (see Figure 2). In this task, the reward function consists of two attractors $\boldsymbol{g}_1, \boldsymbol{g}_2$ and a reward scaling matrix Σ_r, such that the reward function is given by

$$r(\boldsymbol{a}) = -\min_i \left((\boldsymbol{a}-\boldsymbol{g}_i)\Sigma_r(\boldsymbol{a}-\boldsymbol{g}_i)^T\right),$$

and has two local optima at $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$, respectively. The illustration in Figure 2 demonstrates two key insights into PS methods. First, many standard policy search methods such as Policy Improvement with Path Integrals (Theodorou et al., 2010) or EM-based policy search methods (Kober et al., 2008) will be attracted to multiple optima in a multi-modal solution space and, therefore, converge slowly due to averaging over several modes (Neumann, 2011). Second, we would like to represent a versatile solution space by learning all modes of the reward function. Figure 2 shows a qualitative comparison of HiREPS to the standard REPS algorithm which can only use one sub-policy and to the naive implementation of HiREPS that does not bound the sub-policies' entropy (i.e., $\tilde{\kappa} = \infty$). The single sub-policy algorithm tries to average over both modes and, takes a long time to converge. The naive HiREPS exhibits similar behavior. Both sub-policies are attracted by both modes. In most cases, both sub-policies will find the same mode and convergence will be slower. Thus, only introducing hierarchical policies without additional constraints cannot take full advantage of the increased flexibility. When limiting the entropy, however, the sub-policies quickly separate and concentrate on the two individual modes, allowing for a fast improvement of the policy without getting stuck between two modes.

3.1 Episodic Selection of Sub-Policies

We start our discussion of HiREPS with the continuous multi-armed contextual bandit setting. In this setting, the agent is presented with an initial state according to an initial state distribution $p^0(\boldsymbol{s})$ and has to select a sub-policy o as well as an action $\boldsymbol{a}$ according to this state. In this setting, an episode consists of executing exactly one sub-policy, afterwards the episode is terminated and the environment is reset. Thus, no state transition is modeled, as the whole episode consists of only one step, i.e., executing one sub-policy until it terminates.

Contextual Optimization of the Policy. While we do not need to use the state distribution constraint from REPS, as we do not have to incorporate state transitions, the agent can still not freely choose its state action distribution $p(\boldsymbol{s},\boldsymbol{a}) = \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s})$ as the initial state distribution $p^0(\boldsymbol{s})$ is specified by the learning task. Hence, the estimated state distribution $\mu^\pi(\boldsymbol{s})$ has to satisfy $\mu^\pi(\boldsymbol{s}) = p^0(\boldsymbol{s})$ for all $\boldsymbol{s}$. As this requirement would result in an infinite number of constraints, we need to resort to matching feature averages, i.e.,

$$\hat{\boldsymbol{\phi}} = \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} \tag{12}$$

where $\hat{\boldsymbol{\phi}}$ denotes the average observed feature vector for the initial state

$$\hat{\boldsymbol{\phi}} = \iint q(\boldsymbol{s},\boldsymbol{a})\boldsymbol{\phi}(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}. \tag{13}$$

This constraint enforces that the learned state action distribution does not concentrate only on contexts where the task is easy to achieve, but considers all contexts sampled from the initial state distribution.

Resulting Optimization Problem. For the resulting optimization problem, we combine the insights from the previous section on hierarchical policies with the contextual episodic learning constraint in Eq. (12). The episodic learning problem of multiple sub-policies is then given as

$$\begin{aligned}
\max_{\pi,\mu} J(\pi) &= \max_{\pi,\mu} \sum \iint \mu^\pi(\boldsymbol{s})\pi(o|\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\mathcal{R}_{\boldsymbol{sa}}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a},\\
\text{s.t.}\quad \epsilon &\geq D_{\mathrm{KL}}\left(\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})||\; q(\boldsymbol{s},\boldsymbol{a})p(o|\boldsymbol{s},\boldsymbol{a})\right),\\
\tilde{\kappa} &\geq \mathbb{E}_{\boldsymbol{s},\boldsymbol{a}}\left[H\big(p(o|\boldsymbol{s},\boldsymbol{a})\big)\right],\\
\hat{\boldsymbol{\phi}} &= \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a},\\
1 &= \sum \iint \mu^\pi(\boldsymbol{s})\pi(o|\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)d\boldsymbol{s}d\boldsymbol{a}.
\end{aligned} \tag{14}$$

In this optimization problem, the responsibilities

$$p(o|\boldsymbol{s},\boldsymbol{a}) = \frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{\sum_{o\in\mathcal{O}}\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})},$$

occur inside the log-term of the KL-divergence. As the responsibilities contain a sum in the denominator, we also obtain the sum inside the log-term, preventing us from solving for $\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})$ in closed form. However, we can resort to an iterative, expectation maximization (EM) update strategy shown in Daniel et al. (2012a). In the expectation step, we first fix the sub-policies and compute the responsibilities $\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})$. In the maximization step, we replace $p(o|\boldsymbol{s},\boldsymbol{a})$ in our optimization problem with the pre-computed responsibilities $\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})$ and, therefore, neglect that the responsibilities will change once we change the sub-policies. In Appendix A we show that the EM-step maximizes a lower bound of the original optimization problem. We use this iterative update strategy in all further algorithms. The adapted optimization problem reads as

$$\begin{aligned}
\max_{\pi,\mu} J(\pi) &= \max_{\pi,\mu} \mathbb{E}_{\boldsymbol{s},\boldsymbol{a},o}\left[\mathcal{R}_{\boldsymbol{sa}}\right],\\
\text{s.t.}\quad \epsilon &\geq D_{\mathrm{KL}}\left(\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})||\; q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})\right),\\
\tilde{\kappa} &\geq -\sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\log\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a},\\
\hat{\boldsymbol{\phi}} &= \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a},\\
1 &= \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}.
\end{aligned} \tag{15}$$

The above optimization problem can be solved by the method of Lagrange multipliers. The Lagrangian also allows for a closed form of $\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})$ which is given as

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \propto q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\frac{\xi}{\eta}} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} - V(\boldsymbol{s})}{\eta}\right), \tag{16}$$

which depends on the Lagrangian parameters ξ, η and $\boldsymbol{\theta}$ with $V(\boldsymbol{s}) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s})$. The Lagrangian parameter ξ is associated to the overlapping constraint in Eq. (11). In this update equation, the parameter ξ controls the spreading of the sub-policies. A higher value of ξ will force the sub-policies to spread further apart and reduce overlapping of sub-policies as the influence of state-action pairs with a high entropy of $p(o|\boldsymbol{s}, \boldsymbol{a})$ is weakened. The derivation of the dual formulation is given in the appendix.

In Table 3, we give the algorithmic form of episodic HiREPS. The new parametric policy $\tilde{\pi}(\boldsymbol{a}|\boldsymbol{s}, o; \boldsymbol{\beta}_o)$ has to be computed from the weighted samples by performing a weighted maximum likelihood estimate for the single sub-policy parameters $\boldsymbol{\beta}_o$ as well as the parameter vector for the gating policy. In HiREPS, we obtain a weight

$$w_{i,o} = \tilde{p}(o|\boldsymbol{s}_i, \boldsymbol{a}_i)^{1+\frac{\xi}{\eta}} \exp\left(\frac{\mathcal{R}_{\boldsymbol{s}\boldsymbol{a}} - V(\boldsymbol{s}_i)}{\eta}\right),$$

for each sample and each sub-policy. These weights can be used to update the policy. HiREPS does not make any assumptions on the form of the policy and many different representations could be considered. In Section 4, we show the details of the policy model we chose to implement for the evaluations, i.e., a soft-max gating policy and linear Gaussian sub-policies.

Since HiREPS does not assume knowledge of which sub-policy generated a sample, it can use all samples to update all sub-policies, also known as inter-option learning. The weights $w_{i,o}$ are then used to update the sub-policies $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ and the gating policy $\pi(o|\boldsymbol{s})$.

3.2 Sequencing of Skills

Many real world tasks require not only one, but multiple sequential interactions. For example, in a game of tennis, we need to perform several tennis strokes in sequence in order to win the game. Just learning to return a ball is insufficient to win a game. Instead, the ball has to returned in such a way that future strokes can lead to situations where the opponent is unable to return the ball. Hence, tasks like tennis can be learned more efficiently if we learn a sequence of tennis strokes against an opponent. We will model the skill sequencing case with a limited number K of sequential decisions. We will denote the individual action selection problems as stages of the decision problem. Our goal is to maximize the sum of the expected reward over all stages, i.e.,

$$\begin{aligned} J &= \mathbb{E}_{\boldsymbol{a}_{1:K}, \boldsymbol{s}_{1:K+1}}\left[r_{K+1}(\boldsymbol{s}_{K+1}) + \sum\nolimits_{k=1}^{K} r_k(\boldsymbol{s}_k, \boldsymbol{a}_k)\right] \\ &= \int \mu^{\pi}_{K+1}(\boldsymbol{s}) r_{K+1}(\boldsymbol{s}) d\boldsymbol{s} + \iint \sum_{k=1}^{K} \mu^{\pi}_k(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}) r_k(\boldsymbol{s}, \boldsymbol{a}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a}, \end{aligned} \tag{17}$$

where $\mu^{\pi}_k(\boldsymbol{s})$ are the state distributions at for each decision step. The function $r_{K+1}(\boldsymbol{s}_{K+1})$ denotes the final reward for reaching state $\boldsymbol{s}_{K+1}$ and $r_k(\boldsymbol{s}_k, \boldsymbol{a}_k)$ denotes the reward for executing an action $\boldsymbol{a}_k$ in state $\boldsymbol{s}_k$. In the tennis example, the individual rewards could, for example, describe the energy efficiency of the movements, while the overall reward signal carries information about winning or losing the point. The sub-policy is given by

$$\pi_k(\boldsymbol{a}|\boldsymbol{s}) = \sum_{o \in \mathcal{O}} \pi_k(o|\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o). \tag{18}$$

Input: Information loss tolerance ϵ, Entropy tolerance $\tilde{\kappa}$, Number of sub-policies O, number of Iterations L, number of episodes per iteration M.
Initialize all $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ and $\pi(o|\boldsymbol{s})$.

for $l = 1$ to L ... # iterations

Collect samples

for $i = 1, \ldots, M$ (# episodes)

Sample initial state $\boldsymbol{s}_{1,i}$ from environment.

Sample action: $\boldsymbol{a}_i \sim q(\boldsymbol{a}|\boldsymbol{s}_i) = \sum_{o \in \mathcal{O}} \pi_{\text{old}}(o|\boldsymbol{s}_i) \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}_i, o)$.

Execute action $\boldsymbol{a}_i$ and observe reward $r(\boldsymbol{a}_i, \boldsymbol{s}_i)$.

Compute Responsibilities:

$\tilde{p}(o|\boldsymbol{s}_i, \boldsymbol{a}_i) = p_{\text{old}}(o|\boldsymbol{s}_i, \boldsymbol{a}_i) = \frac{\mu^{\pi}_{\text{old}}(\boldsymbol{s}_i) \pi_{\text{old}}(\boldsymbol{a}_i|\boldsymbol{s}_i o_i) \pi_{\text{old}}(o|\boldsymbol{s}_i)}{\sum_{o \in \mathcal{O}} \mu^{\pi}_{\text{old}}(\boldsymbol{s}_i) \pi_{\text{old}}(\boldsymbol{a}_i|\boldsymbol{s}_i o_i) \pi_{\text{old}}(o|\boldsymbol{s}_i)}$ for all i.

Minimize the dual function

$[\boldsymbol{\theta}^*, \eta^*, \xi^*] = \arg\min_{[\boldsymbol{\theta}, \eta, \xi]} g(\boldsymbol{\theta}, \eta, \xi)$.

Policy update:

Compute model distribution

$\mu^{\pi}(\boldsymbol{s}_i) \pi(\boldsymbol{a}_i|\boldsymbol{s}_i o_i) \pi(o|\boldsymbol{s}_i) \propto \tilde{p}(o|\boldsymbol{s}_i, \boldsymbol{a}_i)^{1+\xi^*/\eta^*} \exp\left(\frac{R_i - V^*(\boldsymbol{s}_i)}{\eta^*}\right)$.

Estimate policies

$\pi(o|\boldsymbol{s})$ and $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ for all $o = 1 \ldots O$ by weighted ML estimates.

Output: Policies $\pi(\boldsymbol{a}, o|\boldsymbol{s})$

Table 1: Episodic HiREPS. In each iteration the algorithm starts by sampling an sub-policy o from the gating policy $\pi(a|\boldsymbol{s})$ given the initial state $\boldsymbol{s}$ and an action $\boldsymbol{a}$ from $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ from the sub-policy. Subsequently, the action is executed to generate the reward $r(\boldsymbol{s}, \boldsymbol{a})$. The parameters η, ξ and $\boldsymbol{\theta}$ are determined by minimizing the dual-function g.

Alternatively, we can also share the sub-policies for all decision stages and only make the gating policy time dependent, i.e.,

$$\pi_k(\boldsymbol{a}|\boldsymbol{s}) = \sum_{o \in \mathcal{O}} \pi_k(o|\boldsymbol{s}) \pi(\boldsymbol{a}|\boldsymbol{s}, o).$$

This formulation allows for reusing experience between the stages. Whether this property is useful depends on the task at hand. For example in tennis, the strokes at the decision stages can taken from the same skill library[1]. In the case of skill sequencing, we have one latent variable $o_{i,k}$ per rollout i and per decision stage k.

Connecting the Decision Stages. The state distributions $\mu^{\pi}_k(\boldsymbol{s})$ are now connected through the transition dynamics $\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}$ yielding

$$\int \boldsymbol{\phi}(\boldsymbol{s}') \mu^{\pi}_{k+1}(\boldsymbol{s}') \, \mathrm{d}\boldsymbol{s}' = \sum_{o \in \mathcal{O}} \iiint \mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'} \mu^{\pi}_k(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o) \pi_k(o|\boldsymbol{s}) \boldsymbol{\phi}(\boldsymbol{s}') \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{s}' \, \mathrm{d}\boldsymbol{a}. \tag{19}$$

1. Except for the first stroke, as it is supposed to be the serve.

In addition, the agent cannot freely choose the initial state distribution $\mu_1^\pi(\boldsymbol{s})$, but needs to match the given initial state distribution $p_1(\boldsymbol{s})$, which is implemented in the same way as for the episodic case. Typically the dynamics $\mathcal{P}_{\boldsymbol{ss}'}^{\boldsymbol{a}}$ of a task are not known and need to be estimated from data.

Skill Sequencing Optimization Problem. After including the overlapping constraints from the previous section, the optimization problem reads

$$\max_{\pi_k, \mu_k^\pi} \mathbb{E}_{\boldsymbol{a}_{1:K}, \boldsymbol{s}_{1:K+1}} \left[r_{K+1}(\boldsymbol{s}_{K+1}) + \sum\nolimits_{k=1}^{K} r_k(\boldsymbol{s}_k, \boldsymbol{a}_k) \right],$$

$$\text{s.t.} \quad \epsilon \geq D_{\text{KL}} \left(\mu_{K+1}^\pi(\boldsymbol{s}) || \, q_{K+1}(\boldsymbol{s}) \right),$$

$$\hat{\boldsymbol{\phi}}_1 = \int \boldsymbol{\phi}(\boldsymbol{s}') \mu_1^\pi(\boldsymbol{s}') \, \mathrm{d}\boldsymbol{s}',$$

$$\forall k \leq K : \epsilon \geq D_{\text{KL}} \left(\mu_k^\pi(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o) \pi_k(o|\boldsymbol{s}) || \, q_k(\boldsymbol{s}, \boldsymbol{a}) \tilde{p}_k(o|\boldsymbol{s}, \boldsymbol{a}) \right),$$

$$\tilde{\kappa} \geq - \sum_{o \in \mathcal{O}} \iint \mu_k^\pi(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o) \pi_k(o|\boldsymbol{s}) \log \tilde{p}_k(o|\boldsymbol{s}, \boldsymbol{a}) \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{a},$$

$$\int \boldsymbol{\phi}(\boldsymbol{s}') \mu_{k+1}^\pi(\boldsymbol{s}') \, \mathrm{d}\boldsymbol{s}' = \sum_{o \in \mathcal{O}} \iiint \mathcal{P}_{\boldsymbol{ss}'}^{\boldsymbol{a}} \mu_k^\pi(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o) \pi_k(o|\boldsymbol{s}) \boldsymbol{\phi}(\boldsymbol{s}') \, \mathrm{d}\boldsymbol{s} \, \mathrm{d}\boldsymbol{s}' \, \mathrm{d}\boldsymbol{a},$$

$$1 = \sum\nolimits_{\boldsymbol{s}, \boldsymbol{a}, o} \mu_k^\pi(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o) \pi_k(o|\boldsymbol{s}). \tag{20}$$

The resulting policy update rules are given by

$$\mu_k^\pi(\boldsymbol{s}) \pi_k(\boldsymbol{a}|\boldsymbol{s}, o) \pi_k(o|\boldsymbol{s}) \propto$$
$$q_k(\boldsymbol{s}, \boldsymbol{a}) \tilde{p}_k(o|\boldsymbol{s}, \boldsymbol{a})^{1 + \frac{\xi_k}{\eta_k}} \exp\left(\frac{r_k(\boldsymbol{s}, \boldsymbol{a}) + \mathbb{E}[V_{k+1}(\boldsymbol{s}')|\boldsymbol{s}, \boldsymbol{a}] - V_k(\boldsymbol{s})}{\eta_k} \right), \tag{21}$$

where one set of Lagrangian parameters $\xi_k, \eta_t, \boldsymbol{\theta}_k$ is computed for each stage. As we observe, we now obtain an individual value function $V_k(\boldsymbol{s}_k)$ for each decision stage. As in the standard REPS algorithm, the value functions are connected by the advantage function term that occurs inside the exponent in the policy. The skill sequencing algorithm is given in Table 2. The derivations of the dual function and the update rules are given in Appendix A.2. In Section 4.3, we discuss how to compute the expectation over next states $\mathbb{E}[V_{k+1}(\boldsymbol{s}')|\boldsymbol{s}, \boldsymbol{a}]$.

3.3 Infinite Horizon

An infinite horizon setting is needed for all repetitive tasks where we sequence an unknown, possibly infinite amount of sub-policies to fulfill the task. For example, when bouncing a ball on a paddle, the repetitive paddling movements induce a clear structure in the motion that suggests the use of sub-policies. In addition, we want to bounce the ball on the paddle for an infinite amount of time. In this setting, the agent needs to consider the state of the environment before each stroke.

Input: Information loss tolerance ϵ, entropy tolerance $\tilde{\kappa}$, number of sub-policies n, number of time steps K, number of iterations L.
Initialize all $\pi_k(\boldsymbol{a}|\boldsymbol{s}, o)$ and $\pi_k(o|\boldsymbol{s})$.

for $l = 1$ to L ... # iterations
  Collect samples
    for $i = 1, \ldots, M$ (# episodes)
      Sample initial state $\boldsymbol{s}_{1,i}$ from environment.
      for $k = 1, \ldots, K$ (# motor primitives)
        Sample action: $\boldsymbol{a}_{k,i} \sim q_k(\boldsymbol{a}|\boldsymbol{s}_{k,i}) = \sum_{o \in \mathcal{O}} \pi_{k,\text{old}}(o|\boldsymbol{s}_{k,i}) \pi_{k,\text{old}}(\boldsymbol{a}|\boldsymbol{s}_{k,i}, o)$.
        Execute action $\boldsymbol{a}_{k,i}$, observe next state $\boldsymbol{s}_{k+1,i}$ and reward $r(\boldsymbol{a}_{k,i}, \boldsymbol{s}_{k,i})$.
      Observe Final Reward: $r(\boldsymbol{s}_{K+1,i})$.

  Compute Responsibilities:
    $\tilde{p}_k(o|\boldsymbol{s}_{k,i}, \boldsymbol{a}_{k,i}) = p_{k,\text{old}}(o|\boldsymbol{s}_{k,i}, \boldsymbol{a}_{k,i})$ for all k and i.

  Minimize the dual function
    $[\boldsymbol{\theta}_{1:K+1}{}^*, \eta_{1:K+1}{}^*, \xi_{1:K+1}{}^*] = \arg\min_{[\boldsymbol{\theta}_{1:K+1}, \eta_{1:K+1}, \xi_{1:K+1}]} g\left(\boldsymbol{\theta}_{1:K+1}, \eta_{1:K+1}, \xi_{1:K+1}\right)$.

  Policy update:
    for $k = 1, \ldots, K$
      Compute model distribution
        $\mu_k^\pi(\boldsymbol{s}_{k,i}) \pi_k(\boldsymbol{a}_{k,i}|\boldsymbol{s}_{k,i}, o) \pi_k(o|\boldsymbol{s}_{k,i}) \propto$
        $\tilde{p}_k(o|\boldsymbol{s}_{k,i}, \boldsymbol{a}_{k,i})^{1 + \xi_k{}^*/\eta_k{}^*} \exp\left(\frac{R_{k,i} + \mathbb{E}[V_{k+1}^*(\boldsymbol{s}')] - V_k^*(\boldsymbol{s}_{k,i})}{\eta_k{}^*} \right)$.
      Estimate policies
        $\pi_k(o|\boldsymbol{s})$ and $\pi_k(\boldsymbol{a}|\boldsymbol{s}, o)$ by weighted ML estimates.

Output: Policies $\pi_k(\boldsymbol{a}, o|\boldsymbol{s})$ for all $k = 1, \ldots, K$

Table 2: Time-indexed HiREPS. In each iteration the algorithm starts by sampling from the policy π_1 given the initial state $\boldsymbol{s}_1$ and executes the sampled action to generate the next state $\boldsymbol{s}_2$. From this state, the next action is sampled with policy π_2. This procedure is repeated until the final time-step is reached. The algorithm observes state transitions and rewards for each step k and the final reward signal $r(\boldsymbol{s})$. The parameters $\eta_{1:K+1}$, $\xi_{1:K+1}$ and $\boldsymbol{\theta}_{1:K+1}$ are determined by minimizing the dual-function g, where $\eta_{1:K+1}$ and $\xi_{1:K+1}$ are vectors containing the Lagrangian parameters η_k and ξ_k for each decision step.

The traditional objective in an infinite horizon MDP is to use the discounted accumulated future rewards, i.e.,

$$R = \sum_{t=0}^{\infty} \gamma^t r_t,$$

where $\gamma < 1$ is a discount factor. Such an objective can be easily transferred to the average reward setting by introducing a reset probability γ, where the agent jumps to a state sampled from the initial state distribution $\mu^0(\boldsymbol{s})$ with probability γ and transitions to the next state with probability $(1 - \gamma)$

(van Hoof et al., 2015). Thus, we obtain a transformed transition probability distribution given by

$$\tilde{p}(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a}) = \gamma p(\boldsymbol{s}'|\boldsymbol{s},\boldsymbol{a}) + (1-\gamma)\mu^0(\boldsymbol{s}), \tag{22}$$

where γ is the reset probability. Thus, the discount factor is considered as a termination probability. The constraint ensuring the state transition probabilities are satisfied reads as

$$\int \boldsymbol{\phi}(\boldsymbol{s}')\mu^\pi(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}' = \sum_{o\in\mathcal{O}} \iiint \tilde{\mathcal{P}}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{s}'\,\mathrm{d}\boldsymbol{a}. \tag{23}$$

The optimization problem for the infinite horizon case reads as

$$\begin{aligned}
\max_{\pi,\mu^\pi} J(\pi) &= \max_{\pi,\mu^\pi} \mathbb{E}_{\boldsymbol{s},\boldsymbol{a},o}\left[\mathcal{R}_{\boldsymbol{s}\boldsymbol{a}}\right], \\
\text{s. t.} \quad \epsilon &\geq D_{\mathrm{KL}}\left(\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})||\, q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})\right), \\
\tilde{\kappa} &\geq -\sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\log\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\
\int \boldsymbol{\phi}(\boldsymbol{s}')\mu^\pi(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}' &= \sum_{o\in\mathcal{O}} \iiint \gamma\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s}') + (1-\gamma)\mu^0(\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{s}'\,\mathrm{d}\boldsymbol{a}, \\
1 &= \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a},
\end{aligned} \tag{24}$$

with the resulting policy update

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \propto q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{s}\boldsymbol{a}} + \mathbb{E}[V(\boldsymbol{s}')|\boldsymbol{s},\boldsymbol{a}] - V(\boldsymbol{s})}{\eta}\right), \tag{25}$$

where

$$\mathbb{E}[V(\boldsymbol{s}')|\boldsymbol{s},\boldsymbol{a}] = \boldsymbol{\theta}^T\left[\int \gamma\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}' + \int(1-\gamma)\mu^0(\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}'\right].$$

In contrast to the finite horizon case, the value function, the policy and the state distributions are now stationary instead of time dependent.

4. Algorithmic Design Choices

Having shown the theoretical foundation of HiREPS, we use this Section to explain implementation details concerning the policy representation and parametrizations as well as feature representations and model learning. The choices detailed in this Section represent only some of the possible implementations and are design choices. The HiREPS framework as specified in the previous Section is independent of these design choices and can be used with arbitrary representations for the gating, the sub-policies as well as arbitrary feature representations. HiREPS only requires that the policy representation can be updated using weighted samples.

Input: Information loss tolerance ϵ, Entropy tolerance $\tilde{\kappa}$, Number of sub-policies O, number of Iterations L, number of rollouts per iteration M, reset probability γ.
Initialize all $\pi(\boldsymbol{a}|\boldsymbol{s},o)$ and $\pi(o|\boldsymbol{s})$.

for $l = 1$ to L ... # iterations

Collect samples

for $i = 1,\ldots,M$ (# rollouts)

t = 1; reset = 0;
Sample initial state $\boldsymbol{s}_{i,t}$ from environment.
while reset $< \gamma$
Sample action $\boldsymbol{a}_{i,t} \sim q(\boldsymbol{a}|\boldsymbol{s}_{i,t}) = \sum_{o\in\mathcal{O}} \pi_{\text{old}}(o|\boldsymbol{s}_{i,t})\pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}_{i,t},o)$.
Execute action $\boldsymbol{a}_{i,t}$ and observe reward $r_{i,t}(\boldsymbol{a}_{i,t},\boldsymbol{s}_{i,t})$.
Observe reward $r_{i,t}(\boldsymbol{a}_{i,t},\boldsymbol{s}_{i,t})$ and next state $\boldsymbol{s}_{i,t+1}$.
Sample reset value.

Compute Responsibilities:

$\tilde{p}(o|\boldsymbol{s}_{i,t},\boldsymbol{a}_{i,t}) = p_{\text{old}}(o|\boldsymbol{s}_{i,t},\boldsymbol{a}_{i,t}) = \frac{p_{\text{old}}(\boldsymbol{s}_{i,t},\boldsymbol{a}_{i,t},o)}{\sum_{o\in\mathcal{O}} p_{\text{old}}(\boldsymbol{s}_{i,t},\boldsymbol{a}_{i,t},o)}$ for all i,t.

Minimize the dual function

$[\boldsymbol{\theta}^*,\eta^*,\xi^*] = \arg\min_{[\boldsymbol{\theta},\eta,\xi]} g\left(\boldsymbol{\theta},\eta,\xi\right)$.

Policy update:

Compute model distribution

$p(\boldsymbol{s}_{i,t},\boldsymbol{a}_{i,t},o) \propto \tilde{p}(o|\boldsymbol{s}_{i,t},\boldsymbol{a}_{i,t})^{1+\xi^*/\eta^*}\exp\left(\frac{R_{i,t}+\mathbb{E}[V(\boldsymbol{s}')]-V^*(\boldsymbol{s}_{i,t})}{\eta^*}\right)$.

Estimate policies

$\pi(o|\boldsymbol{s})$ and $\pi(\boldsymbol{a}|\boldsymbol{s},o)$ for all $o = 1\ldots O$ by weighted ML estimates.

Output: Policies $\pi(\boldsymbol{a},o|\boldsymbol{s})$

Table 3: Infinite Horizon HiREPS. The algorithm follows the form of the episodic implementation, however one iteration produces state, action, reward triples for each time step in each rollout of an iteration. In the infinite horizon case, a model $\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}$ is required to compute $\mathbb{E}[V(\boldsymbol{s}')]$.

4.1 Policy Representation

We implemented the hierarchical policy using a soft-max gating network $\pi(o|\boldsymbol{s})$ and linear Gaussian sub-policies $\pi(\boldsymbol{a}|\boldsymbol{s},o)$. The gating network $\pi(o|\boldsymbol{s})$ chooses which option to activate according to the model

$$\pi(o=k|\boldsymbol{s}) = \frac{\exp\left(\boldsymbol{\phi}_G(\boldsymbol{s})^T\boldsymbol{\beta}_{G,k}\right)}{\sum_i^O \exp\left(\boldsymbol{\phi}_G(\boldsymbol{s})^T\boldsymbol{\beta}_{G,i}\right)},$$

where $\boldsymbol{\beta}_{G,1\ldots O}$ are the parameter vectors defining the influence of the sub-policies and $\boldsymbol{\phi}_G$ are the feature representations of the state used for the gating network. While this model itself is relatively simple, the complexity and expressiveness of the resulting policy depends largely on the feature transformations employed for the individual components of the hierarchical policy. In HiREPS, we are free to choose different feature transformations for the sub-policies, the gating and the value

function, where the value function features are the features in the primal optimization problem. In the presented experiments, the sub-policies themselves are represented by linear Gaussian policies, i.e.,

$$\pi(\boldsymbol{a}|\boldsymbol{s}, o; \boldsymbol{\beta}_o) = \mathcal{N}\left(\boldsymbol{a}|\boldsymbol{\mu}_o + \boldsymbol{F}_o\boldsymbol{s}, \boldsymbol{\Sigma}_o\right),$$

where $\boldsymbol{\beta}_o = [\boldsymbol{\mu}_o, \boldsymbol{F}_o, \boldsymbol{\Sigma}_o]$ is the parameter tuple describing sub-policy o. The sub-policies could also be defined on a feature transformation of the state. Equally well, non-linear sub-policies instead of the linear Gaussians could be used. However, since HiREPS performs a piecewise linear approximation in the state space using multiple sub-policies, linear sub-policies are usually sufficient. Both, the gating policy as well as the sub-policies can easily be updated using the sample weights computed by HiREPS. The update equations, as shown in the appendix, yield a weight matrix with one row per sample and cone column per sub-policy. The gating requires the full weight matrix as well as the responsibilities to be updated whereas each sub-policy is updated independently using the respective column of the weight matrix to weigh all state-action samples.

Generally, HiREPS can be initialized with many more sub-policies than solutions are expected to be available for the problem at hand. The gating network $\pi(o|\boldsymbol{s})$ will reduce the probability of selecting options that are not concentrated on good solutions to zero and effectively ignore these unnecessary sub-policies. Options can also be pruned throughout the learning process as in the first series of experiments described in Section 5.1.2. There, we report the number of actual solutions found. In this case, whenever the prior $p(o) = \int \pi(o|\boldsymbol{s})\mu(\boldsymbol{s})d\boldsymbol{s}$ of sub-policy activation gets too small (i.e., $p(o) < 10^{-4}$) we delete the sub-policy. However, in order to avoid sub-policies getting deleted too quickly, we assure that each sub-policy gets a minimum amount of samples in the sampling process. We also bound the minimum variance of our Gaussian sub-policies to small values in order to avoid singularities. After the sampling process, we evaluate the quality of the exploration-free policy (i.e., without variance) found so far.

4.2 Feature Based State Representation

We use feature transformations for representing the value function as well as achieving a higher flexibility in the gating policy. Different feature representations can be used for the gating and the value function. In the presented experiments, the gating network $\pi(o|\boldsymbol{s})$ is constructed on squared expansion of the state space. Thus, the squared features are constructed as the vector of all linear and squared combinations of the state dimensions. For example, for a two-dimensional state space the feature vector would be given by

$$\boldsymbol{\phi}([s_1, s_2]) = [1, s_1, s_2, s_1^2, s_2^2, s_1 s_2].$$

While the squared expansion of the state space can also be used as feature representation for the value function, more complex tasks often benefit from a more flexible representation, such as a kernel based feature transformation. In our experiments we used either kernel based features or features based on a Fourier transformation as shown by (Konidaris et al., 2011). The kernel based features used a squared exponential kernel, i.e.,

$$[\boldsymbol{\phi}_V]_i(\mathbf{s}) = \exp\left(-\frac{1}{2}(\mathbf{s} - \mathbf{s}_i)^T \boldsymbol{\Lambda}^{-1}(\mathbf{s} - \mathbf{s}_i)\right),$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix denoting the bandwidth of the kernel and the index i iterates over the reference set of observed states.

Choosing sufficiently expressive features for the representation of the value function is crucial to the success of the learning algorithm. As an example, we can consider a contextual bandit task, where the robot has to select the number of steps to take to walk to a goal position. The context in this task is given by the robot's initial distance to the goal. If the robot has access to insufficient features, e.g., only a constant feature, it cannot differentiate between the different starting positions, and, hence, cannot learn to adapt its policy to the initial position. This effect is compounded in the infinite horizon setting, were successful policies often depend on reaching intermediate states with potentially low rewards on the path to the goal states.

4.3 Model Learning

For the finite horizon, as well as the for the infinite horizon formulation, HiREPS requires a transition model $\mathcal{P}^{\boldsymbol{a}}_{\boldsymbol{s}\boldsymbol{s}'}$. In this paper, we use a sample based transition model, which reproduces previously observed state transitions and does not necessarily generalize. Using a sample based transition model effectively results in computing $V(\boldsymbol{s}')$ directly based on one single observed state transition instead of computing the expectation $\mathbb{E}[V(\boldsymbol{s}')|\boldsymbol{s}, \boldsymbol{a}]$ in the update, Equation (25), as well as in the corresponding dual function in Eq. (51) as given in Appendix A.3. Replacing the expectation in such a manner is only feasible if the controlled system has limited stochasticity. However, it is indeed still sufficiently robust to solve the real robot tasks as presented in the experimental Section. For systems with more stochasticity, van Hoof et al. (2015) show how to learn a better model that is able to generalize if sufficient data is available.

4.4 Sample Efficiency

One of the important trade-offs when using iterative policy update methods is the number of rollouts performed per iteration. More rollouts per iteration result in more stable policy updates but also increase the number of overall rollouts required to learn a task. When using sample based policy update techniques in high dimensional action spaces, the number of samples is crucial. Using fewer samples than action dimensions will lead to an underestimate of the variance or the variance might even collapse, as the new policy is just based on the available samples. In HiREPS, we are not restricted to using only samples from the previous iteration of rollouts when computing the sample weights. By considering samples from M multiple past iterations, we can stabilize the policy update while retaining a fast learning speed. As a general rule of thumb, keeping three times as many samples as used per rollout can stabilize the algorithm against aggressive choices in the number of rollouts per iteration (lower than number of parameters) or high ϵ (≥ 1.5). When keeping samples from previous iterations, we usually define our current state-action distribution $q(\boldsymbol{s}, \boldsymbol{a})$ to be given by the collection of these samples. Alternatively, importance weighting schemes can be employed to incorporate samples from previous iterations. However, in our experience, these importance weighting schemes are often prone to destabilizing the learning algorithm.

4.5 Parametrized Trajectories

For the results of this paper, we often rely on movement primitives that inherently encode temporal correlations, such that each sub-policy $\pi(\boldsymbol{a}|\boldsymbol{s}, o)$ describes the parametrization of one movement primitives. We are using the extended version of DMPs by Kober et al. (2010a), that parametrizes trajectories using a combination of weighted basis functions as well as the end point

g and final velocity $\dot{g}$ of the trajectory. DMPs are based on a simulated spring damper system $\ddot{x} = \alpha\left(\dot{g} - \dot{x}\right) + \beta\left(g - x\right)$ with damping factor α and spring stiffness β. The spring damper system will result in a trajectory that reaches the desired goal and goal velocity in a direct manner. However, the spring damper system alone is insufficient to encode arbitrary shapes along the trajectory. To represent arbitrary shapes, DMPs modulate the dynamical system with a forcing function $f(z, \boldsymbol{v}) = \boldsymbol{\Phi}(z)^T \boldsymbol{v}$ that depends on the phase z of the movement, the basis functions $\boldsymbol{\Phi}(z)$ and the basis functions weights $\boldsymbol{v}$. Using the forcing function to deviate from the direct path of the spring damper system allows DMPs to generate complicated trajectory shapes. To ensure that the desired final goal position and velocity constraints are met, the forcing function depends on the exponentially decaying phase variable z, such that the forcing function does not influence the system towards the end of the trajectory.

If a demonstration trajectory, for example from kinesthetic teach-in, exists, basis function weights $\boldsymbol{v}$ that reproduce the shape of that motion as well as the goal position and velocity can be computed to initialize the learning process. After initialization, the basis function weights as well as the desired goal position and velocity are the parameters that the reinforcement learner optimizes over. In the presented robot learning experiments we learn one DMP per degree of freedom. It is important to note that the output of the DMPs only represents the desired trajectories, which are then usually tracked by an internal controller of the robot itself. This distinction is important because even if the robot is unable to track trajectories, e.g., due to gain or torque limitations, the RL agent can adapt the DMP such that the resulting trajectory still solves the task. When using DMPs, each sub-policy lasts until the DMP has finished execution.

4.6 Computational Complexity

The presented algorithm depends on the three key computations: solving the dual function for HiREPS, fitting a gating policy and fitting the individual sub-policies. The optimization of the dual problem can be performed using existing optimizers such as, for example, the Broyden-Fletcher-Goldfarb-Shannon algorithm. The optimization can often be accelerated by providing the first and second derivative of the dual function. While the actual complexity depends on which algorithm is used, we can analyze the problem itself. The number of open parameters in this optimization problem stays constant with the number of options. However, the number of open parameters depends directly on the dimensionality of the features $\boldsymbol{\phi}_V(\boldsymbol{s})$ for the value function. Thus, the complexity of the optimization problem depends on the feature representation rather than on the number of options.

Fitting a gating policy based on the weights computed through the HiREPS optimization problem yields a multiclass classification problem. For example, multiclass logistic regression can be used. As before, a solution can be found through the use of third party optimization software. However, the number of open parameters scales multiplicative with the dimensionality of the gating feature representation and the number of options. Thus, for high dimensional gating features and a large number of options, the gating optimization problem can become computationally expensive.

Finally, the sub-policies have to be fitted. Given the gating, the sub-policies are independent and each sub-policy can be fitted individually using a weighted maximum-likelihood estimate. In the case of linear Gaussians, the complexity of this operation is governed by a linear regression operation which depends on the dimensionality of the sub-policy feature representation. However,

since these optimizations are performed individually per option, the computational effort for this last step is usually negligible.

4.7 Hyperparameter Tuning

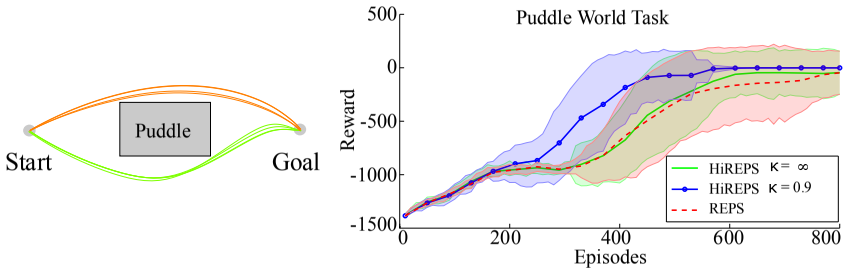
The presented method exposes three main hyperparameters to be set by the practitioner. These are the number of options O, the entropy bound κ as well as the relative entropy bound ϵ. The number of options can usually be chosen generously, i.e., around 20 seems to be reasonable for a wide range of problems. The algorithm will prioritize more promising options such that even if too many options are initialized, only options which yield high rewards will be sampled from after the first few iterations. The entropy bound κ is probably the most important parameter to consider since it does not have a clear equivalent in existing approaches. However, our experiments showed that a value of 0.9 seems to work well in almost all cases and no major tuning was necessary. The parameter ϵ is probably the parameter that is the most tuning-intensive in the proposed method, especially if the total number of episodes is crucial, e.g., in real robot experiments. In our experience values for ϵ between 0.5 and 1.5 are reasonable and, most often, we would start a new task with $\epsilon = 1$. Changing these parameters certainly influences the learning speed of the proposed method. However, while sub-optimal settings may lead to slower convergence, they usually do not prevent successful learning. Thus, in our experience, the algorithm is generally robust in that even sub-optimal settings will lead to convergence.

5. Experimental Validation

We validated the different presented formulations of HiREPS on both simulated and real robot tasks. We compare our algorithm to the non-hierarchical counterpart, REPS (Peters et al., 2010). Evaluating against REPS is interesting, as the two algorithms share the same basis. In effect, REPS is equivalent to our algorithm with just one sub-policy. This similarity allows us to directly investigate the influence of adding a hierarchical policy representation without additional confounders. The experimental section is structured analogously to the structure of Section 3, i.e., we first report results on the contextual episodic settings, subsequently the skill sequencing problem and finally the infinite horizon setting. For the contextual setting as well as for the skill sequencing, we report real robot results. In both cases, we start by presenting results on related simulated experiments to better investigate the properties of the presented algorithms as well as to compare to the REPS algorithm. For all presented results, we have optimized the parameters of the algorithms to deliver the best performance and both algorithms receive the same number of samples per iteration. For all experiments, if not noted otherwise, the experiments were repeated ten times to produce the errorbars reported in the results.

5.1 Episodic Skill Based Tasks

We start by presenting results for the episodic formulation of HiREPS. In the contextual setting, as well as for the skill sequencing setting in the next part, we start by presenting results of simpler experiments that allow a more in-depth analysis of the algorithms and then proceed to more complex tasks.



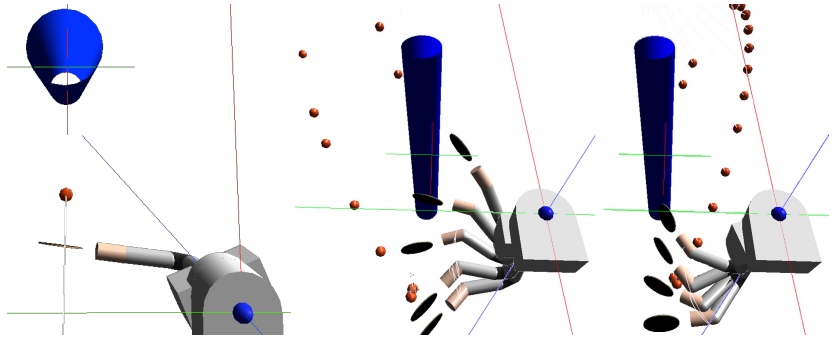
(a) Puddle World (b) Evaluation

Figure 3: (a) The puddle world task. Sub-policies are represented as two-dimensional DMPs with fixed end points. The DMPs have five basis functions per dimension and we learn the weights of the basis functions for the y dimension while leaving the weights for the x dimension fixed. The plot shows trajectories sampled from two sub-policies found after 80 iterations (using 10 samples per iteration) of HiREPS. Start and end points of the trajectory are pre-defined, the agent may choose the path in between those points. The puddle world has two solutions. The plot shows samples from both modes acquired by HiREPS after 30 iterations. (b) Performance of HiREPS and REPS on the puddle world task. As HiREPS can represent both solutions, it does not get stuck averaging over both modes. Note, that this effect is much more pronounced if we bound the entropy of the sub-policies.

5.1.1 PUDDLE WORLD EXPERIMENT

In a first toy task, we test HiREPS on a variation of the puddle world (Sutton, 1996). While this task is of limited difficulty it is interesting as it is a well known setting which exhibits the averaging problem of interest to us. Additionally, the simplicity of the problem allows us to thoroughly assess the quality of the solutions found by the RL agent, which is often difficult in real robot tasks. Our version differs from the standard version by having a continuous action space instead of a discrete one. While the agent proceeds with a constant velocity along the x-dimension of the environment, it has to learn the DMPs shape parameters such that the puddle is avoided. Thus, the actions $\mathbf{a}$ of our sub-policies are five-dimensional vectors. The reward of the task is given by the negative length of the line segments, which encourages shorter solutions. An additional punishment occurs for passing through the puddles. The arrangement of the puddles can be seen in Figure 3a. The presented puddle world has two solutions which are located close to each other. Still, the mean of both solutions leads through a puddle and, therefore, yields lower rewards.

In Figure 3b, we evaluate the performance of REPS and HiREPS with and without bounding the sub-policies' entropy. For HiREPS, just two sub-policies were used. REPS takes a longer time to reliably find good solutions, as the algorithm averages over both modes. HiREPS without bounded entropy performs slightly better than REPS. However, the advantage of HiREPS is much more pronounced when also bounding the expected entropy of the responsibilities. Furthermore, if we limit the entropy, the algorithm is able to reliably find both modes. The results also show that



(a) Top View (b) Left and right swings

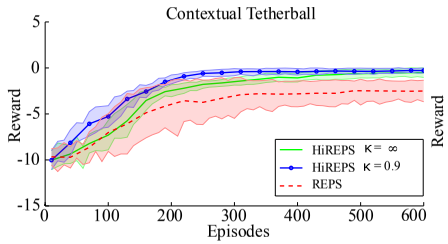
Figure 4: (a) Top view of the setup of robot tetherball. The ball is hung on a string from the ceiling in front of the pole. (b) Time series overlay of two swings in simulation, one left swing and one right swing. The two solutions use different sub-policies of the same hierarchical policy. HiREPS can also keep more sub-policies representing additional solutions to a task.

even after 80 iterations (using ten samples per iteration), the REPS learning curve still exhibits some variance, showing that the algorithm has not fully converged.

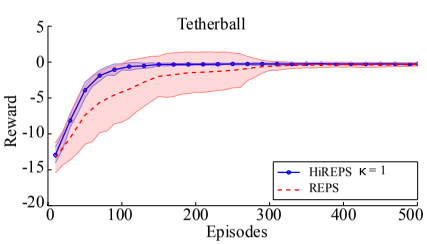
5.1.2 TETHERBALL

Our aim is to adapt the game 'Tetherball' for a robotic player, as shown in Figure 4b. Specifically, the task consists of a ball, a rope, an obstacle (i.e., a pole) and a target. The ball is hung in front of the pole, in a line with the robot, the pole and the target. This setup requires the robot to induce a circular trajectory to be able to wind the ball around the pole. In the planar simulation, the robot can induce this angular velocity through the elasticity of the rope. In the physically accurate simulation as well as on the real robot, a pre-strike is necessary to achieve similar results. This task presents a versatile solution space, as many different strategies successfully hit the target. Our goal is to model this versatile solution space with HiREPS. In order to thoroughly assess the proposed method, we split our investigation of the tetherball task into three parts of ascending difficulty. In a first evaluation, we implement a planar simulation of the task in which we abstract the robot into a force, i.e., we do not simulate a robotic arm but instead allow the agent to directly exert a force onto the ball. We use this simplified setup to compare HiREPS to its non-hierarchical counterpart as well as to investigate the effect of different settings. In a second evaluation, we use a detailed physical simulation of the task including the robot arm, in which we evaluate the benefits of a hierarchical policy in a more realistic setting. Finally, we present the results of the actual robotic task which show that HiREPS finds multiple solutions within one learning scenario.

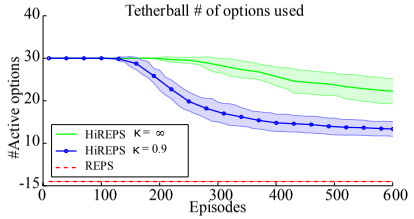
Planar Simulation. For the purpose of testing the HiREPS method, we use a strongly simplified setup where the tetherball task is implemented in a planar simulation. We initialize HiREPS with 30 randomly located sub-policies and use ten samples per iteration. The agent can accelerate the ball



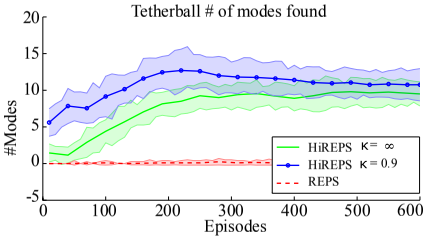
(a) Tetherball Average Reward



(b) Tetherball without States, Average Reward



(c) Tetherball Number of Sub-Policies Used



(d) Tetherball Number of Modes Found

Figure 5: (a) Average reward gathered by the REPS and the HiREPS with and without bounding the entropy of the sub-policies in the tetherball task including states. As REPS only uses a single sub-policy and thus a linear model as policy, it cannot represent the complicated structure of the solution. The HiREPS approach benefits from bounding the entropy of the sub-policies. (b) Average reward in the tetherball task without states. While REPS also finds one of the optimal solutions, HiREPS benefits from its structured policy representation and outperforms REPS in learning speed. At the same time, HiREPS finds both solutions. (c) Number of sub-policies used by the HiREPS approach with and without bounding the entropy. If the prior $p(o)$ of a sub-policy becomes too small it gets deleted. By bounding the overlap of the sub-policies, less sub-policies are used while the performance of the algorithm is increased. (d) Number of modes found by HiREPS with and without bounding the overlap of the sub-policies. We can see that, despite that HiREPS with bounding the overlap uses less sub-policies, it can find more modes. Without bounding the overlap of the sub-policies, many sub-policies concentrate on the same mode, which attenuates the advantages of the structured policy representation.

with a two dimensional impulse $\{F_x, F_y\}$. The reward is given by the negative minimum squared distance of the ball to the target throughout the ball's trajectory. The initial state of the agent is given by the initial position of the ball before hitting it. We only vary the x-position of the ball and learn different solutions to hit the target. As before, we compare HiREPS with and without bounding the overlap of the sub-policies to the standard REPS approach. In Figure 5a, we evaluate the average reward of all three approaches. The results show that HiREPS with the bound on the overlap outperforms the two other methods. The HiREPS approach already without the additional bound is better than the REPS approach, as REPS only uses one sub-policy to cover the whole

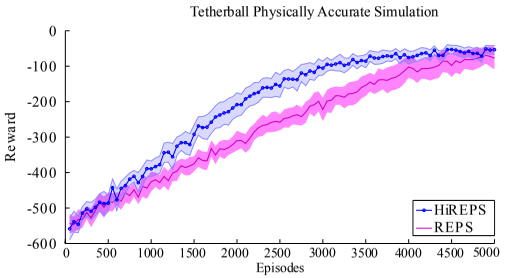


Figure 6: Results of the physically accurate tetherball simulation. Both REPS and HiREPS reach the same asymptotic reward. However, since HiREPS can differentiate between the multiple local optima and assign different options to these optima, it exhibits faster learning speed.

state space. Thus, REPS needs to approximate the optimal policy using a single linear model. Therefore, in order to have a fair comparison to REPS, we also compare HiREPS and REPS on the tetherball task without states, i.e., we always start from the same initial state in the middle. This comparison can be found in Figure 5b. We can observe that REPS is impaired by the multi-modality of the solution space as it tries to average over several modes, but finally learns a solution that is equally good as the solutions found by HiREPS. However, while REPS only learns one solutions, Figure 5d shows that HiREPS learns multiple solutions at the same time. In order to compute the number of modes, we divide each dimension of the state action space into 5 partitions and count the number of partitions which contain at least one sub-policy with an average reward larger than -1. The plot shows that, as the sub-policies distribute more uniformly in the state-action space due to the bounding, we can find more modes. Thus, the bound of the overlap also helps us to find more versatile solutions as it avoids situations where multiple sub-policies concentrate on the same solution. In Figure 5c, we show the number of sub-policies used for different bounds of the overlap $\tilde{\kappa}$. By bounding the overlap, the gating network can learn to select individual sub-policies more decisively, explaining the faster learning speed in the experiments when using the bound.

Physically Accurate Simulation. Having compared the properties of (Hi)REPS on the simpler simulation, we proceed to presenting the results of the physically accurate simulation as shown in Figure 4a, which also serves as a stepping stone to the real robot results. As described, the pole is placed on a line between the ball's resting position and the target location, such that it is impossible to hit the target with a single strike of the ball in direction of the target. In order to evoke a circular trajectory that arcs the ball around the ball, displacing the ball from its resting pose is necessary. Thus, we decompose our movement into a swing-in motion and a hitting motion. However, the parameters for both motions are combined into one parameter vector that is learned jointly. A more powerful approach would be to respect the natural separation of the successful trajectories and use the skill sequencing approach as presented in Section 3.2. The reward is determined by the speed of the ball when the ball winds around the pole. We define winding around the pole as the ball passing the pole on the opposite side of the pole. We run the algorithms with 50 samples per iteration and always keep the last 400 samples. We initialize our algorithm with 30 sub-policies and stop deleting sub-policies if only 5 sub-policies are left. The resulting learning curve in the simulation can is shown in Figure 6. After 100 iterations the robot has learned to wind the ball around the pole in 5/5 trials. In all trials, we were able to observe sub-policies for the left and for the right mode. The resulting movements are shown in Figure 4b and illustrate that the resulting movement of the two solutions are easily differentiated. The results show, that albeit HiREPS uses the same amount of

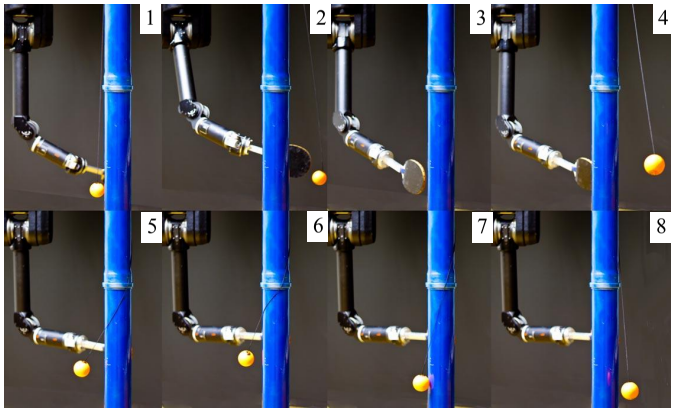


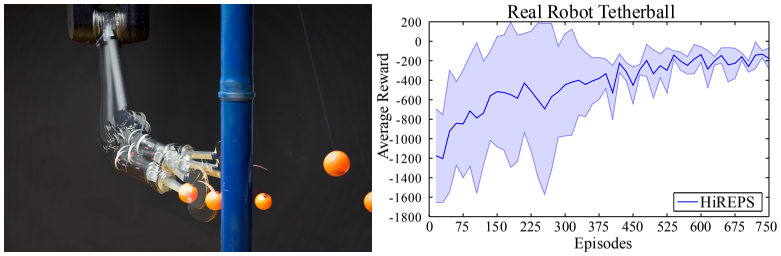
Figure 7: Time series of a successful swing of the robot. The robot first has to swing the ball to the pole and can, once the ball has swung backwards, arc the ball around the pole.

total samples per iteration as REPS, it can use those samples to learn multiple solutions while being faster than REPS can learn a single solution.

Real Robot Tetherball. After presenting the results on the two simulated tetherball settings, we proceed to show the results of the real robot experiment. For the robot experiment, we mounted a table-tennis paddle to the end-effector of the robot arm. The real-robot setup is depicted in Figure 8a and two successful hitting movements of the real robot are shown in Figure 7. In order to track the ball, a Kinect RGBD camera was setup to look at the robot from the opposite side of the pole. The vision data was used to compute the reward signal. As in the physically accurate simulation, the robot needed to perform a pre-swing as well as the actual swing. However, the real robot can easily be bootstrapped through imitation learning. Thus, we extract the shape parameters $\boldsymbol{v}$ by kinesthetic teach-in (Ijspeert and Schaal, 2003) for both motions. Subsequently, the robot learns the final positions and velocities of all seven joints through the presented approach. Additionally, we learn the waiting time between both movements. This task setup results in a $2 \times 2 \times 7 + 1 = 29$-dimensional action space. We initialized the algorithm with 15 sub-policies and sampled 15 trajectories per iteration. While this scheme amounts to considerably fewer samples than in simulation, it is sufficient to learn multiple solutions, given the initial demonstrations. We performed three trials to produce the errorbars reported in the results. The learning curve is shown in Figure 8b. The noisy reward signal is mostly due to the vision system and partly also due to real world effects such as friction which lead to a non-repeatability of rollouts. Two resulting movements of the robot are shown in Figures 7 and 4b.

5.2 Planar Reaching Task

To further evaluate the properties of the proposed algorithm, we present a series of experiments on a planar reaching task. In this simulated task, the agent controls the joint trajectories of a three-link



(a) Tetherball (b) Real robot results

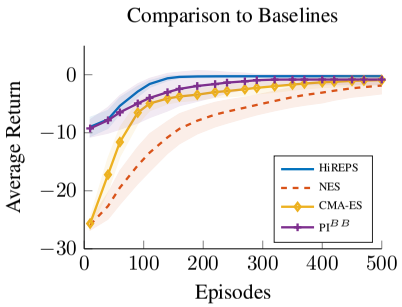
Figure 8: (a) The real robot tetherball setup. (b) Average rewards of HiREPS on the real robot Tetherball setup. Mean and standard deviation of three trials are shown. In all three trials, the robot has found solutions to wind the ball around the pole on either side after 50 iterations.

robotic arm using DMPs with two basis functions per joint. The robot starts from a fixed initial position and executes a trajectory for 100 time steps. At time steps 40 and 100 the robot is rewarded for passing through via points. While the robot is controlled in joint space, these via points are given in task space. Furthermore, for both time steps two possible via points exist.

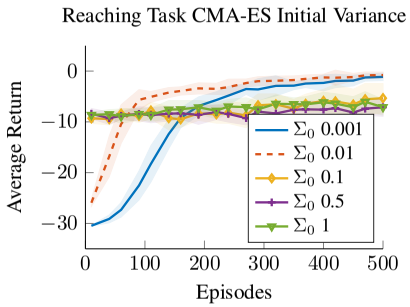
Comparison to Baseline Methods. Most of the presented experiments compare the performance of HiREPS to its natural competitor, REPS. To better analyze the performance levels of HiREPS, we also performed an experiment comparing HiREPS to other state of the art methods. Specifically, we compared to the PI^{BB} algorithm (Stulp and Sigaud, 2013), the CMA-ES algorithm (Hansen et al., 2003) as well as the NES algorithm (Wierstra et al., 2014). To compare HiREPS against these baselines we performed an empirical optimization of the open parameters these methods perform. For example, Fig. 9b shows the effects of the initial variance of CMA-ES. The results in Fig. 9a show that HiREPS converges faster than the alternative algorithms we compared to. Furthermore, the alternative algorithms are designed to represent a single solution. In this experiment, all algorithms used 10 samples per iteration. Both HiREPS and PI^{BB} used a higher initial variance than CMA-ES and NES in this experiment since the latter methods' performance decreases using higher initial variances as shown in Fig. 9b. Due to this higher initial variance, PI^{BB} and HiREPS start with a higher initial reward.

Initialization of the Algorithm. Since the presented algorithm is based on the availability of multiple sub-policies, these options have to be initialized to sensible values. In the presented experiments linear Gaussians were usually used as sub-policies, i.e., $\pi(\boldsymbol{a}|\boldsymbol{s}, o) = \mathcal{N}(\boldsymbol{a}|\boldsymbol{\mu}_o + \boldsymbol{F}_o\boldsymbol{s}, \Sigma_o)$. To initialize each linear Gaussian, we can keep $\boldsymbol{F}_o = \boldsymbol{0}$ and $\Sigma_o = c\boldsymbol{I}$, where the constant c is a task specific variable set by the experimenter. While F_o and Σ_o are initially identical for all sub-policies, the means $\boldsymbol{\mu}_o$ have to be different to allow for later separation of the sub-policies based on the responsibilities. To that effect, we usually sampled the individual sub-policy means $\boldsymbol{\mu}_o$ from a normal distribution, i.e., $\boldsymbol{\mu}_o \sim \mathcal{N}(\cdot|\boldsymbol{0}, \Sigma_\mu)$.

Figure 10a shows the results of varying Σ_μ relative to the admissible range of parameters. The results show that while a larger initial separation seems to help bootstrapping the learning process,



(a) Comparison To Baselines



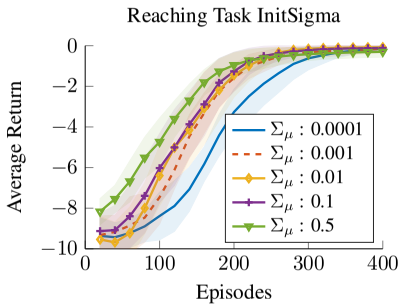
(b) Influence of Initial Variance in CMA-ES

Figure 9: (a) Comparison of HiREPS to popular baselines. HiREPS and PI^{BB} start at higher initial rewards because CMA-ES as well as NES have to be initiated with a much lower variance. While HiREPS and Pi^{BB} start with a high-variance initial policy (or sub-policies) and keep contracting it until convergence. CMA-ES and NES on the other hand work best with a smaller initial variance that is kept for longer. (b) Effects of different initial variances on CMA-ES. We chose the best initial variance for the comparative experiments.

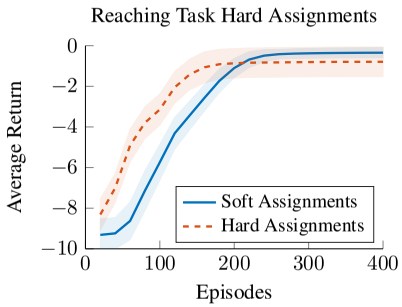
the overall effect of changing this parameter is negligible. In this experiment, a total of 20 options were available to the algorithm. While the comparison to the base-line methods was performed using ten samples per iteration, for the following experiments 20 samples per iteration were used. Using more samples per iteration allowed us to observe the effects under investigation more clearly, while the comparison to other baselines was optimized for the CMA-ES and NES algorithms.

The Influence of Probabilistic Option Assignments. The presented algorithm allows for inter-option learning, i.e., the sharing of information between options during learning which is expected to improve learning. To test this hypothesis, we performed an experiment comparing the proposed formulation of the algorithm to an alternative formulation based on hard assignments. In the alternative formulation, every option was effectively limited to using it's own samples to update its policy. This behavior was achieved by assigning a responsibility of 1 to the option that generated a sample while all other options had zero responsibility for the same sample. Using soft assignments allows the gating to shift responsibilities such that options can specialize on distinct sub-tasks. The results in Figure 10b show that using soft assignments did indeed improve asymptotic learning performance but was slower in the early stages. However, in our experiments we observed that the importance of this effect would diminish with an increasing number of options. As the number of options increases, HiREPS can implement a more effective divide and conquer strategy. Sharing of information can becomes less important after the division of the state-action space has been successfully performed.

Evaluation of the Entropy Bound. One of the main parameters to be considered in the proposed algorithm is the desired bound on the entropy of the responsibilities, $\tilde{\kappa}$. The lower the value for $\tilde{\kappa}$, the more aggressively the algorithm will separate the individual options. Figure 11a shows the



(a) Effect of Different Initializations



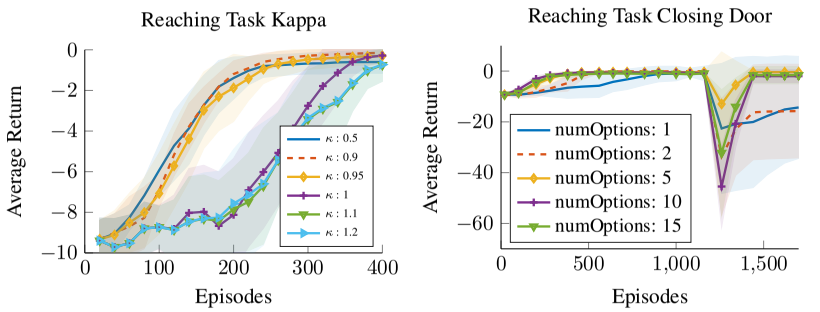
(b) Effect of Probabilistic Responsibilities

Figure 10: (a) Investigation of different initial distributions of sub-policies. Using a low initial variance of the distribution over option means, all options share similar responsibilities for all samples and the entropy bound forces a separation of the options. Using a higher initial variance for this distribution, the individual sub-policies will be responsible for different regions of the action space from the beginning. While the results show that such an initial separation can improve learning speed initially, it can also lead to sub-optimal asymptotic performance. (b) Using hard assignments, i.e., not sharing information between options, leads to high initial learning speed. However, using the hard assignments, the algorithm did not always find the optimal solution. In our experiments, the non-probabilistic (hard assignments) version of the algorithm would usually require more samples per iteration to consistently find equally good solutions as the proposed probabilistic approach. However, our experience shows that when using more samples per iteration, the learning speed of the hard assignment approach will actually decrease relative to the speed of the probabilistic approach. This effect can be explained by the fact that each option is drawn to all local optima more uniformly if more samples are used.

results of evaluating a wide range of possible parameter values for $\tilde{\kappa}$ in the reaching task. The results show that values for $\tilde{\kappa} \geq 1$ yield slower convergence speeds while the options still overlap and aim to explain multiple solutions. As in most other experiments presented in this paper, a value of $\tilde{\kappa} = 0.9$ seems to yield the best learning speeds. While even lower values of $\tilde{\kappa}$ do not noticeably decrease the learning speed in the presented experiment, our experience shows that lower values may prevent successfully learning multiple solutions. Since lower values of $\tilde{\kappa}$ force options apart, only few options may actually be fully developed while the probability of sampling from most other options will diminish.

The results reported for this experiment and the puddle world experiment regarding the entropy bound mirror our experience with the remaining experiments reported in this paper. In all experiments a bound of $\kappa = 0.9$ seems to give good results and much higher (> 0.95) or much lower (< 0.8) values usually deteriorate performance. Thus, for the remaining experiments, we chose $\kappa = 0.9$ and do not present further evaluations.

Robustness to Changing Environments. One main contribution of the proposed algorithm is to learn multiple solutions for the same task. Learning multiple solutions can be interesting if, for



(a) Influence of Entropy Bound

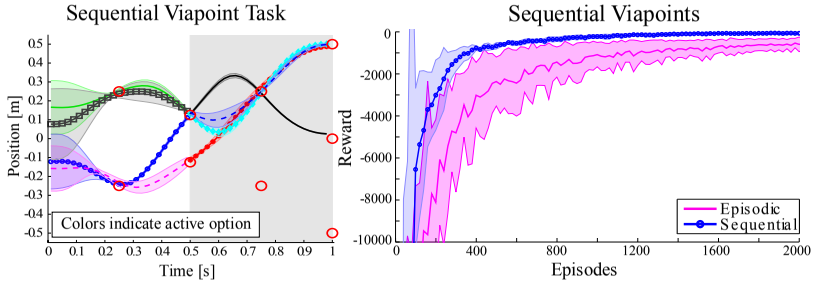
(b) Changing Environments

Figure 11: (a) Comparison of different relative values κ for the entropy bound. The results show that higher values for the entropy bound slow down learning, as the options are not forced to separate. Thus, multiple options will compete for the same local optimum for longer. Very low values for κ will also slow down learning and can potentially lead to sub-optimal asymptotic performance when the entropy bound 'overpowers' the reward maximization criterion. (b) Investigation of a task which requires multi-modal solutions. After learning a solution, some of the via points were randomly disabled and the agent had to rely on the availability of alternative solutions. The results show that increasing the number of options allowed for a robust policy which could recover from the change in the environment because the agent had previously learned multiple solutions.

example, the environment changes in a way that makes some solutions inaccessible. To evaluate the behavior of the proposed algorithm in such a scenario, we let the algorithm learn a solution for the reaching task as described above. After 30 iterations HiREPS typically converged to a good solution. At that point, the via points on either the lower or upper path were randomly disabled, which simulated blocking one of the paths. Figure 11b show the effects of learning multiple solutions. With only one option, the agent cannot recover in about half the trials. Using more options increases the likelihood that the agent has learned sufficiently versatile solutions to successfully perform the task even after cutting off of the paths.

5.3 Sequencing of Skills

As evident from the tetherball experiment, many real world tasks require multiple steps to be solved. To evaluate the skill sequencing implementation of HiREPS presented in Section 3.2, we present a second set of experiments. Before testing skill sequencing on a real robot task, we evaluated the time-indexed HiREPS algorithm on a via-point task in order to illustrate the properties of the approach. In this task, we modeled a second-order dynamical system. The state of the agent is given by its position x and velocity $\dot{x}$. The actions u control the accelerations $\ddot{x}$. The task of the agent is to reach specified via-points at four different points in time. For each of these time points, different via-points exist. The reward at the time points is given by the negative squared distance to the closest via-point. In addition to the deviation to the via-points at the four specified time



(a) Task Setup. Colors indicate active option.

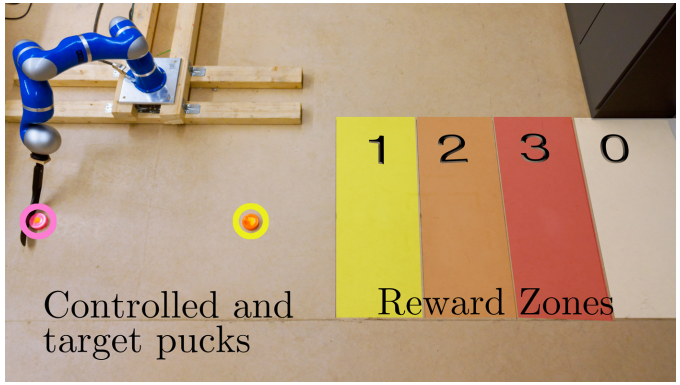
(b) Average Reward.

Figure 12: (a) We use this via-point task to illustrate our algorithm.The agent has to reach one of the via-points (denoted by red circles) at each of four specified times $[0.25, 0.5, 0.75, 1.0]$s. The reward is given by the negative squared distance to the closest via-point. The last via-point has to be reached with zero velocity. The initial positions and velocities are sampled from a Gaussian distribution with zero mean and a standard deviation of 0.25 for the position and 0.1 for the velocity. In this task, we learn to sequence two motor primitives, with the second primitive starting at $t = 0.5$s (shaded region in which the line colors change). This task is per construction multi-modal and illustrates how our algorithm learns distinct motor primitives. The mean and variance are indicated by shaded error bars. The agent learned several but not all possible solutions to solve the task. (b) The multi-modal via-point task learned with episodic and sequential motor primitive learning where the movement was decomposed into two primitives, see Figure 12a for a more detailed description. As we can see, our algorithm could exploit this decomposition resulting in increased learning speed and higher quality final policies.

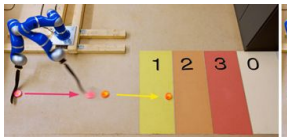
points, the reward function contains a squared punishment term for taking high accelerations. As we defined multiple via-points for each of the four time-points, this task has multiple solutions per construction. The agent used 20 samples per iteration for this task. The exact setting of the task including its via-points is depicted in Figure 12a.

In order to demonstrate sequencing of skills, we decomposed the task into two DMPs which were executed sequentially. We used five shape parameters for both DMPs. In addition, we also learn the goal-parameter of the DMP, resulting in 6 parameters per movement primitive. To compare our sequencing method to the commonly used episodic policy search setup, we also solve this task with the episodic version of HiREPS. In this case, we only used one DMP with ten shape parameters and the additional goal-parameter. For both scenarios, the agent could choose between four distinct sub-policies o_i at each decision time-point. As we can see from Figure 12a, the agent learned to select these primitives according to the state at the decision time-points as well as to adapt the primitives such the task can be solved. Our approach was able to learn multiple solutions for the task as can be seen from Figure 12a. However, only a subset of all possible solutions was found.

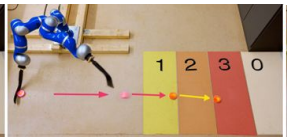
The comparison of the episodic and the sequential learning methods can be seen in Figure 12b and shows the advantage of the sequencing approach in learning speed as well as in the quality of the learned solution. Additionally, the episodic formulation will also require more options in total



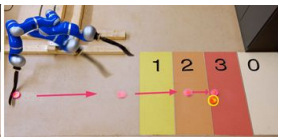
(a) Hockey Setup



(b) First strike



(c) Second strike



(d) Final strike

Figure 13: (a) The robot hockey task. The robot has two pucks, the pink control puck and the yellow target puck. The task is to shoot the yellow target puck into one of the colored reward zones. Since the best reward zone is too far away from the robot to be reached with only one shot, each episode consists of three strikes. After each strike the control puck is returned to the robot, but the target puck is only reset after one episode is concluded. Concluding an episode with the target puck in one of the reward zones yields rewards from one to three as indicated in the picture. However, if the robot shoots the target puck too far, the reward is zero. (b-d) One episode of the Hockey task, consisting of three strikes. Each picture shows the initial and final position of control and target puck. The movement of the pucks is indicated by arrows. The robot can shoot the pink control puck to move the target puck and tries to place the yellow target puck in one of the marked target zones while not overshooting. In the depicted episode the robot only needed two strikes to place the target puck into the highest reward zone. With the last strike the robot taps the target puck only slightly without actually moving it to avoid negative reward for missing it.

in order to find all possible solutions than the sequential formulation. However, in the presented experiment we only rewarded the agent for finding at least one solution such that this effect did not alter the resulting performance analysis.

5.3.1 EVALUATION ON THE ROBOT HOCKEY TASK

As before, after using a simpler task to investigate the properties of the proposed algorithm, we now present the results of a more sophisticated task. Similar to the structure of the tetherball experiments,

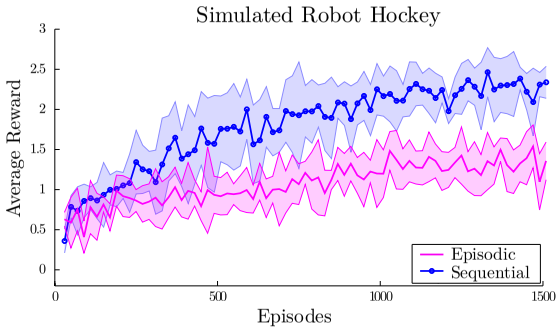


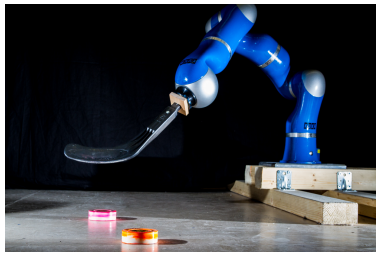
Figure 14: Comparison of sequential motor primitive learning to the episodic learning setup on the simulated robot hockey task. The sequential motor primitive learning framework was able to find a good strategy to place the puck in the third reward zone in most of the cases while the episodic learning scenario failed to learn such a strategy.

we first present a comparative analysis on a physically accurate simulation and then proceed to show the results of the real robot task. In the robot hockey task, the robot has to move a target puck into one of three target areas. This target puck can only be moved by shooting a control puck at it. The target areas are defined by a specified distance to the robot. The first zone is defined as distance from 1.4 to 1.8m, the second zone from 1.8 to 2.2m and the last zone from 2.2 to 2.6m. The robot gets rewards of 1, 2, and 3 for reaching zone 1, 2 or 3, respectively, with the target puck. If the robot overshoots the last target zone, the reward drops to zero. The reward is only given after each episode which consists of three shots of the control puck. After each shot, the control puck is returned to the robot. The target puck, however, is only reset after each episode. The setup of the robot hockey task is shown in Figure 13a.

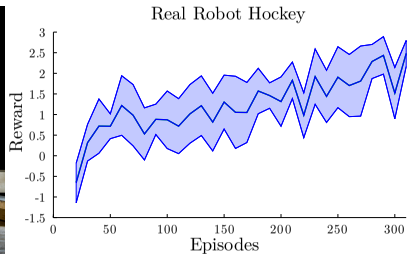
The 2-dimensional position of the target puck defines the state of the environment as perceived by the agent. After performing one shot, the agent observes the new position of the target puck to plan the subsequent shot. In order to give the agent an incentive to shoot at the target puck, we punished the agent with the negative minimum distance of the control puck to the target puck after each shot. While this reward was given after every step, the zone reward was only given at the end of the episode (every third step) as $r(\boldsymbol{s}_{K+1})$.

We used a DLR-Kuka lightweight arm with 7 degrees of freedom as depicted in Figure 15a. We used DMPs to represent single motor primitives where we only adapted the goal positions and velocities of the primitives. This setup resulted in 14 parameters per primitive per shot. Thus, the episodic version of HiREPS has to optimize one 42-dimensional parameter vector while the time indexed version of HiREPS that we use for skill sequencing has to optimize three policies with 14-dimensional parameter spaces each. Both, the episodic as well as the time-indexed version of HiREPS had access to five options. The shape parameters $\boldsymbol{v}$ of the single primitives were learned from imitation by collecting trajectory data via kinesthetic teach-in.

Simulation Results. We first implemented a realistic simulation of the robot hockey task. In simulation, we varied the initial position of the puck by sampling the position from a normal distribution with standard deviation of 10cm. The agent used 30 samples per iteration. We compared our sequential motor primitive learning method with its episodic variant. For the episodic variant, we encoded the policy-parameters of all three shots into one policy, resulting in 42 parameters. The episodic variant cannot use state-feedback except for the information of the initial position of the puck. To make the comparison as fair as possible we did not use any noise in our simulation



(a) Hockey Setup



(b) Real Robot Hockey Rewards

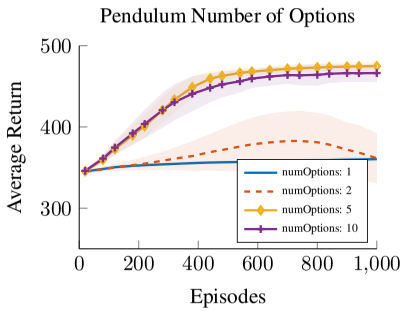
Figure 15: (a) The real robot Hockey setup. The robot first has to shoot the pink puck on the orange puck, to move the orange puck into a target zone. (b) One trial of of the real robot hockey tasks. The robot starts with a negative initial reward and learns to achieve an average reward of 2.5 after 300 episodes. The optimal theoretical reward of the presented task is 3.0. However, learning has not yet converged and had to be stopped prematurely due to time constraints.

and, hence, the initial position of the puck is sufficient to solve the task. The comparison of both methods can be seen in Figure 14. The episodic learning setup failed to learn a proper policy while our sequential motor primitive learning framework could steadily increase the average reward. Our method reached an average reward of 2.3 after learning for 1500 episodes. Note that an optimal strategy would have reached a reward value of 3, however, this is still a clear improvement in comparison to the episodic setup, which reached a final reward value of 1.4.

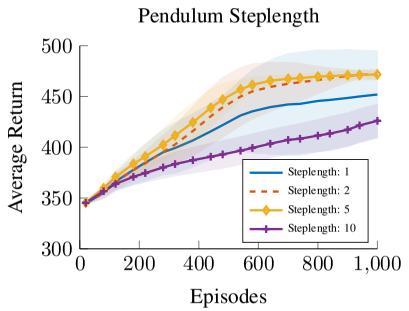
Real Robot Results We used a Kinect RGB-D camera to observe the state of target puck. For the real robot hockey task, the initial position was not varied. On the real robot, we could reproduce the simulation results. The robot learned a strategy which could move the target puck to the highest reward zone in most of the cases after 300 episodes, where the robot used ten samples per iteration. One episode of robot hockey is depicted in Figure 13. In the final trials, the robot tended to prefer using a soft hit in the first shot and to shoot the target puck to the last reward zone with the remaining two shots. This behavior yielded higher average reward, since it is easier to only tap the target puck without moving it too much while it is still closer to the robot. During learning the robot steadily adapted his strategy when it mastered the necessary motor skills to achieve higher rewards by placing the target puck in the highest reward zones.

5.4 Infinite Horizon Formulation for Sequencing Skills

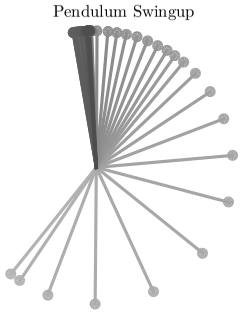
We evaluated the infinite horizon formulation of HiREPS on the pendulum swing-up task in simulation. In this task, the pendulum starts hanging down with a random perturbation. The goal of the robot is to find a solution that first swings up the pendulum and then stabilizes the pendulum at the top. Instead of directly choosing motor commands in each time step, the robot chooses a desired joint value which is tracked with a PD-controller that is active over multiple time steps d (in the standard setting $d = 5$). While using direct motor commands is generally feasible as well, using a PD-control scheme is often beneficial from a robotics point of view. Real systems are often



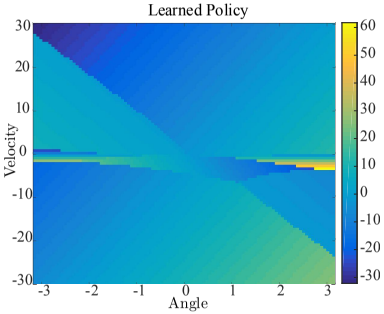
(a) Pendulum #Sub-Policies



(b) Pendulum #Steps per Decision



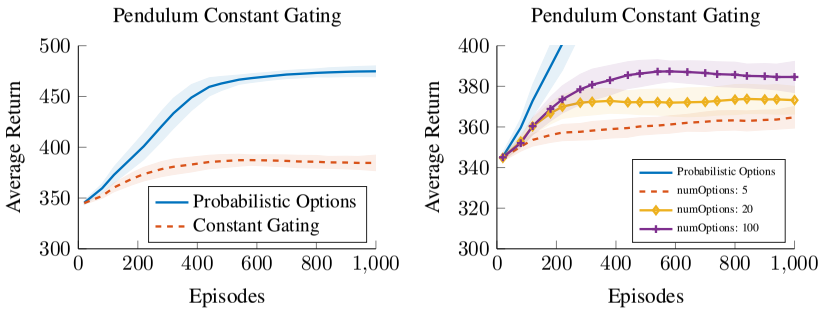
(c) Task Setup



(d) Policy

Figure 16: Evaluations of the pendulum swingup task. The task starts with the pendulum hanging down and the robot has to swing up and stabilize the pendulum. (a) The results show that one linear sub-policy is insufficient to learn this task. With five sub-policies the robot is able to learn the task, with ten sub-policies the asymptotic performance does not improve anymore. (b) The number of time steps d each sub-policy stays active. The results show that activating a sub-policy for multiple time steps increases the speed of learning. If a sub-policy stays active for more than five time steps, the performance decreases. (c) One rollout of the pendulum swingup task. The pendulum starts at the bottom and requires a pre-swing to be brought to the upright position. (d) Pendulum Swingup-Policy learned by HiREPS. Colors indicate the mean value of the policy.

controlled at very high frequencies internally, however, policy signals are usually only necessary at lower frequencies. Reducing the control frequency of the policy will increase the signal to noise ratio, especially in real systems. Furthermore, even in simulated systems, the temporal extension of the actions can benefit the learning process. Since this task is highly non-linear in nature, it cannot be solved with a single linear policy. However, since HiREPS is able to represent a piecewise linear policy, it can be used to solve the pendulum swing-up task with multiple linear sub-policies. In this



(a) Comparison to Constant Gating

(b) Number of Options in Constant Gating

Figure 17: (a) Evaluation of a pre-defined constant gating vs. a learned gating. In the constant gating approach, the gating was pre-determined by performing a K-Means clustering of the state space. The results show that the pre-defined constant gating does not allow the agent to successfully learn the task. (b) Effect of the number of options when using a constant gating. The results show that the constant gating depends strongly on the number of options used, which related directly to the resolution with which the state-action space is parceled. Even with 100 options the constant gating approach is outperformed by the learned approach which uses only five options.

task, the pendulum has a mass of 10kg, a length of 0.5m and a friction coefficient of 0.2. The robot can exert at most 30nm of torque and uses 20 samples per learning iteration. The internal robot control runs at 100Hz and the restart probability in the base setting is given as $(1-\gamma) = 0.02/d$, where d determines the control frequency of the learned policy. The primitive actions wich are chosen by the sub-policies are the desired joint angles of the pendulum. The robot can typically perform a swing-up within 1.5s. To represent the value function, we used the Fourier transform based feature transformation of the states using five Fourier bases, see Section 4. The gating uses a squared expansion of the states as feature representation. The reward function punishes deviation from the desired upright position with a factor of 500 and punishes velocities with a factor of 10. These punishments are subtracted from a base value of 500. The results of the experiments show that a single linear policy is insufficient for swinging up and stabilizing, however, HiREPS successfully combines multiple linear policies to solve the task, a resulting policy is shown in Fig 16.

Evaluation of a Constant Gating. In the pendulum swing-up task, a good gating policy is essential if the task is to be solved using simple sub-policies. To evaluate the effect of a learned gating versus a constant pre-determined gating, we compared the proposed approach to an alternative formulation where the individual action policies can be learned, but the gating stays constant. To find a such a constant gating, we divided the state space using K-Means clustering and kept the resulting gating constant afterwards. To perform K-Means clustering we sampled 1000 data points uniformly within $\pm[\pi, 50]$, where π in this case is the numerical value. The results show that the performance of such a constant gating strongly depended on the number of options available. However, even with 100 options the constant gating did not allow to learn this task successfully.

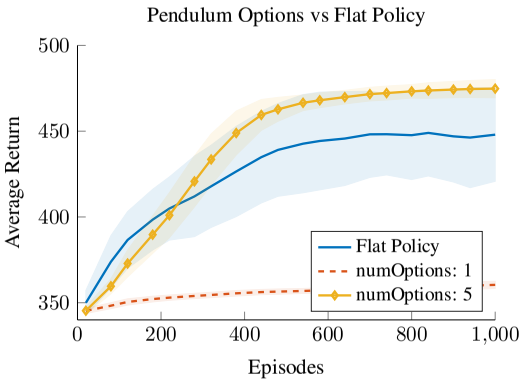
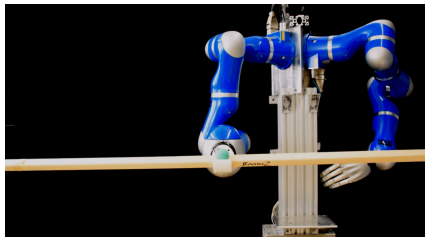


Figure 18: Comparison to flat but non-linear policy. The non-linear policy is given by a linear Gaussian based on a non-linear feature transformation of the states while the hierarchical policy uses only linear sub-policies. The results show that the non-linear flat policy is outperformed by the hierarchical policy. While the flat policy is able to learn the solution, the asymptotic performance varies over multiple runs and would require more samples per iteration to exhibit a robust learning performance.

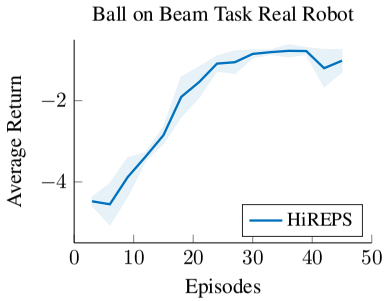
Comparison to a Non-linear Flat Policy. While the proposed approach can learn the pendulum swing-up task using a combination of linear sub-policies, the task can also be solved using a single non-linear policy. To evaluate the effects of the proposed approach, we compared to a non-linear flat policy which was based on the same features as the gating in HiREPS. The results in Figure 18 show that while the non-linear flat policy initially learned faster than the HiREPS, the average asymptotic performance of the HiREPS was higher. Initially, HiREPS requires some iterations to learn a good gating policy. However, once a good gating policy was available, having different options that specialize on sub-tasks such as high accelerations vs. stabilization yielded better policies.

5.5 Infinite Horizon Real Robot Experiment

To evaluate the practical applicability of the proposed infinite horizon approach, we performed a real robot evaluation on the 'ball on a beam' task as shown in Fig. 19a. In this task, the robot has to guide a ball to a prescribed position on a beam and balance it at this position. In our experiments, the robot had to choose between two possible goal positions, 10cm to the left or right of the beam center. To learn the task, the robot learned a policy which set a new desired end effector angle every 0.5s. Every policy was evaluated for 20s and in each iteration the agent collected three episodes before updating the policy. In this task, the state space was given by the ball's position and velocity as well as the current beam angle. The sub-policies were linear Gaussian policies. The gating operated on squared features of the state space and the value function was based on a Fourier feature transformation of the state space using five basis functions. The reward function in this task was given as the summed negative squared distance and velocity of the ball (to the closest goal position) in every control step. The ball's position and velocity were determined using a Microsoft Kinect which was mounted above the robot. The results in Fig. 19b show that the robot was able to learn this task within 30 episodes in all three trials performed.



(a) Ball On Beam Setup



(b) Ball On Beam Results

Figure 19: (a) The experimental setup for the 'ball on a beam' task. A beam of 1m length is attached as the robot's end effector. The robot has to balance a ball in one of two goal positions, $\pm$10cm from the center of the beam. The ball is initialized alternating on the right and left end of the beam. (b) Results of three trials on the real robot. In all trials the robot was able to learn the task within 30 episodes. In the second trial performance dropped slightly at the end of the trial before recovering again. This drop was most likely due to noise in the vision system.

6. Conclusion & Discussion

In this paper, we present a novel method, derived from first principles, to represent multiple solutions to a task. The representation of multiple versatile solutions is achieved though a hierarchical policy, which consists of a gating network and multiple sub-policies. We show that a naive implementation of a hierarchical policy is insufficient as it does not find distinct solutions. To address this problem, we introduce an entropy-based constraint which ensures that the agent finds distinct solutions with different sub-policies. Having a hierarchical policy based on multiple sub-policies additionally allows us to solve tasks that are non-linear using multiple linear sub-policies.

We evaluated the proposed method on two real robot tasks and several simulated tasks. The evaluations showed that our framework can be used to learn temporally extended sub-policies, also called options, and to sequence these sub-policies to learn complicated tasks on real systems.

The results show, that HiREPS is able to learn multiple solutions for complicated tasks and that HiREPS is able to learn piecewise linear solutions for non-linear tasks. The comparison to the non-hierarchical REPS method shows that HiREPS additionally often learns faster than REPS. This effect is especially intriguing as HiREPS does not only aim to learn one solution but multiple solutions at the same time and effectively uses fewer samples per sub-policy.

While the presented approach is able to learn a multi-modal policy using weights generated by solving one optimization problem, alternative solutions are possible. In the presented approach,

computing the lower bound and, subsequently, solving the optimization problem and fitting the sub-policies is equivalent to an EM approach with only one iteration. While EM approaches typically benefit from multiple iterations, we observed no further benefit from applying multiple passes in our experiments. The observation that one EM iteration is sufficient can be explained by the fact that we start with a distribution which is highly similar to the target distribution. Thus, using HiREPS with $\kappa \gg 1$ results in a version of the REPS algorithm which works on a mixture model. Furthermore, alternative mixture model fitting approaches could be used to incorporate the entropy bound which is integrated into the optimization problem in the proposed approach. For example, Graça et al. (2007, 2009) show how EM can be performed with additional constraints such as, for example, the entropy constraint presented in this paper.

In this paper we focus on learning multiple sub-policies and do not solve the temporal extension aspect of the options framework directly. Temporal extension is inherently achieved in many of the presented experiments through the use of movement primitives. Using movement primitives, the time horizon is usually pre-determined. However, this time horizon could be made adaptive by including the duration of the movement primitive into the set of parameters that is learned by the agent. Learning both when to terminate options as well as learning how to construct options from atomic state-action pairs in a unified framework is an important aspect for future work. Equally, extending the framework to not only allow pruning of sub-policies, but also generating new sub-policies during the learning process could be an important addition to the versatility of the presented framework. In the presented version, options might be separated in the state-action space in the beginning of learning when actually multi-modalities exist on a smaller scale than can initially be detected. In such cases, it would be helpful to split one sub-policy into two sub-policies that can adapt to the individual modes.

Acknowledgments

The authors acknowledge the support of the European Union project # FP7-ICT-270327 (ComPLACS). This project has also received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No. 645582 (RoMaNS) as well as funding from the DFG project LearnRobotS.

References

M. Gheshlaghi Azar, V. Gómez, and H. J. Kappen. Dynamic Policy Programming. *Journal of Machine Learning Research*, 13(Nov):3207–3245, 2012.

J. A. Bagnell and J. C. Schneider. Covariant Policy Search. In *Proceedings of the IEEE International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

J. A. Bagnell and J. G. Schneider. Autonomous Helicopter Control using Reinforcement Learning Policy Search Methods. In *Proceedings of the IEEE International Conference for Robotics and Automation (ICRA)*, pages 1615–1620, 2001.

A.G. Barto, S. Singh, and N. Chentanez. Intrinsically motivated learning of hierarchical collections of skills. *Proceedings of the International Conference on Developmental Learning (ICDL),*, 2004.

J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

S. Calinon, P. Kormushev, and D. Caldwell. Compliant Skills Acquisition and Multi-Optima Policy Search with EM-based Reinforcement Learning. *Robotics and Autonomous Systems (RAS)*, 61 (4):369 – 379, 2013.

B. Da Silva, G. Konidaris, and A.G. Barto. Learning parameterized skills. *In Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

C. Daniel, G. Neumann, and J. Peters. Hierarchical Relative Entropy Policy Search. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012a.

C. Daniel, G. Neumann, and J. Peters. Learning Concurrent Motor Skills in Versatile Solution Spaces. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012b.

C. Daniel, G. Neumann, O. Kroemer, and J. Peters. Learning Sequential Motor Tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

P. Dayan and G. E. Hinton. Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271–278, 1997. ISSN 0899-7667. doi: http://dx.doi.org/10.1162/neco.1997.9.2.271.

M. P. Deisenroth. *Efficient Reinforcement Learning using Gaussian Processes*. KIT Scientific Publishing, 2010. ISBN 978-3-86644-569-7.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

T. G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research (JAIR)*, 2000.

G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG-based Biped Locomotion with a Policy Gradient Method: Application to a Humanoid Robot. *International Journal of Robotics Research*, 2008.

M. Ghavamzadeh and S. Mahadevan. Hierarchical Policy Gradient Algorithms. In *International Conference for Machine Learning (ICML)*, pages 226–233. AAAI Press, 2003.

J. V. Graça, K. Ganchev, and B. Taskar. Expectation maximization and posterior constraints. *Advances in Neural Information Processing Systems (NIPS)*, 2007.

J. V. Graça, K. Ganchev, B. Taskar, and F. Pereira. Posterior vs parameter sparsity in latent variable models. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*. 2009.

N. Hansen, S.D. Muller, and P. Koumoutsakos. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003.

A. Ijspeert and S. Schaal. Learning Attractor Landscapes for Learning Motor Primitives. In *Advances in Neural Information Processing Systems 15*, (NIPS). MIT Press, Cambridge, MA, 2003.

J. Kober, B. J. Mohler, and J. Peters. Learning Perceptual Coupling for Motor Primitives. In *Intelligent Robots and Systems (IROS)*, pages 834–839, 2008.

J. Kober, K. Mülling, O. Kroemer, C. H. Lampert, B. Schölkopf, and J. Peters. Movement Templates for Learning of Hitting and Batting. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2010a.

J. Kober, E. Oztop, and J. Peters. Reinforcement Learning to adjust Robot Movements to New Situations. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*, 2010b.

N. Kohl and Peter Stone. Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion. In *Proceedings of the IEEE International Conference for Robotics and Automation (ICRA)*, 2003.

G. Konidaris and A. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1015–1023, 2009.

G. Konidaris, S. Osentoski, and P. Thomas. Value function approximation in reinforcement learning using the fourier basis. *AAAI Conference on Artificial Intelligence (AAAI*, 2011.

P. Kormushev, S. Calinon, and D. Caldwell. Robot Motor Skill Coordination with EM-based Reinforcement Learning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

A. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-Efficient Contextual Policy Search for Robot Movement Skills. *Journal of Artificial Intelligence*, 2014.

S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1071–1079, 2014.

K. Y. Levy and N. Shimkin. Unified inter and intra options learning using policy gradient methods. In *Recent Advances in Reinforcement Learning*, pages 153–164. Springer, New York City, 2012.

J. Morimoto and K. Doya. Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems*, 36(1):37–51, 2001.

K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. *International Journal of Robotics Research*, (3):263–279, 2013.

G. Neumann. Variational Inference for Policy Search in Changing Situations. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011.

G. Neumann and J. Peters. Fitted Q-Iteration by Advantage Weighted Regression. In *Advances in Neural Information Processing Systems (NIPS)*. MA: MIT Press, 2009.

J. Peters and S. Schaal. Policy Gradient methods for Robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robotics Systems (IROS)*, Beijing, China, 2006.

J. Peters, K. Mülling, and Y. Altun. Relative Entropy Policy Search. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI)*, 2010.

K. Rawlik, M. Toussaint, and S. Vijayakumar. On Stochastic Optimal Control and Reinforcement Learning by Approximate Inference. In *Proceedings of Robotics: Science and Systems*, 2012.

M. T. Rosenstein. Robot Weightlifting by Direct Policy Search. *Proceedings of the International Joint Conference on Artificial Intelligence (ICJAI)*, 2001.

S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert. Learning Movement Primitives. In *Proceedings of the International Symposium on Robotics Research*, (ISRR), 2003.

J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015.

S. Still and D. Precup. An Information-theoretic Approach to Curiosity-driven Reinforcement Learning. *Proceedings of the International Conference on Humanoid Robotics*, 2011.

P. Stone. Scaling reinforcement learning toward RoboCup soccer. *Proceedings of the Conference on Machine Learning (ICML)*, 2001.

M. J. A. Strens. Policy Search using Paired Comparisons. *Journal of Machine Learning Research (JMLR)*, 3:921–950, 2003.

F. Stulp and S. Schaal. Hierarchical Reinforcement Learning with Movement Primitives. In *IEEE International Conference on Humanoid Robots (HUMANOIDS)*, 2012.

F. Stulp and O. Sigaud. Policy improvement methods: Between black-box optimization and episodic reinforcement learning. *Journées Francophones Planification, Décision, et Apprentissage pour la conduite de systèmes*, 2013.

R. S. Sutton, D. Precup, and S. Singh. Intra-option learning about temporally abstract actions. In *Proccedings of the International Conference on Machine Learning (ICML)*, 1998.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems (NIPS)*, 1999a.

R. S. Sutton, D. Precup, and S. Singh. Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. *Artificial Intelligence*, 112:181–211, 1999b.

R.S. Sutton. Generalization in Reinforcement Learning: Successful Examples using Sparse Coarse Coding. In *Advances in Neural Information Processing Systems 8*, pages 1038–1044. MIT Press, 1996.

R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Boston, MA, 1998.

E.. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11:3137–3181, 2010.

P. S Thomas and A. G. Barto. Motor primitive discovery. In *IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, 2012.

M. Toussaint, S. Harmeling, and A. Storkey. Probabilistic inference for solving (po) mdps. *University of Edinburgh, School of Informatics Research Report EDI-INF-RR-0934*, 2006.

A. Ude, A. Gams, T. Asfour, and J. Morimoto. Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. *IEEE Transactions on Robotics*, 5, October 2010. ISSN 1552-3098.

H. van Hoof, J. Peters, and G. Neumann. Learning of non-parametric control policies with high-dimensional state features. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1):949–980, 2014.

D. Wingate, N. D. Goodman, D. M Roy, L. P. Kaelbling, and J. B. Tenenbaum. Bayesian policy search with policy priors. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

Appendix A. Derivation of the Lower Bound

Consider the optimization problem in Equation (14) with the real conditional $p(o|s,a)$ instead of the responsibilities $\tilde{p}(o|s,a)$. For simplicity, we do not introduce the steady-state distribution and the normalization constraint as, our results are not affected by these constraints. The Lagrangian of this problem is then given by

$$\begin{aligned} F(\pi,\eta,\xi) = &\sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})R_{\text{sa}}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} \\ &+\eta\left(\epsilon - \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \log \frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{q(\boldsymbol{s},\boldsymbol{a})p(o|\boldsymbol{s},\boldsymbol{a})}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}\right) \\ &+\xi\left(\tilde{\kappa} + \iint p(\boldsymbol{s},\boldsymbol{a}) \sum_{o\in\mathcal{O}} p(o|\boldsymbol{s},\boldsymbol{a}) \log p(o|\boldsymbol{s},\boldsymbol{a})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}\right). \end{aligned} \tag{26}$$

Simplifying the terms, we obtain

$$F(\pi,\eta,\xi) = \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\left(R_{\text{sa}} - \eta \log \frac{p(\boldsymbol{s},\boldsymbol{a},o)}{q(\boldsymbol{s},\boldsymbol{a})p(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}}\right)\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} + \eta\epsilon + \xi\tilde{\kappa}. \tag{27}$$

However, determining a closed form solution for $p(\boldsymbol{s},\boldsymbol{a},o)$ is infeasible as the conditional $p(o|\boldsymbol{s},\boldsymbol{a})$ is inside the $\log$. Yet, we can determine a lower bound $L(p,\eta,\xi,\tilde{p})$ by using a proposal distribution $\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})$ for $p(o|\boldsymbol{s},\boldsymbol{a})$ which we can iteratively maximize in an Expectation-Maximization like manner. We need to verify that L is a lower bound on F and that maximizing L w.r.t $\tilde{p}$ is equivalent to setting $\tilde{p}(o|\boldsymbol{s},\boldsymbol{a}) = p(o|\boldsymbol{s},\boldsymbol{a})$, both of which follows from the relation

$$L = F - (\eta+\xi) \iint p(\boldsymbol{s},\boldsymbol{a}) \underbrace{\sum_{o\in\mathcal{O}} p(o|\boldsymbol{s},\boldsymbol{a}) \log \frac{p(o|\boldsymbol{s},\boldsymbol{a})}{\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})}}_{D_{KL}\left(p(o|\boldsymbol{s},\boldsymbol{a})||\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})\right)\geq 0} \mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}.$$

After the Expectation step, the lower bound is tight, i.e., $\max_{\tilde{p}} L(p,\eta,\xi,\tilde{p}) = F(p,\eta,\xi)$. In the Maximization step, we fix $\tilde{p}$ and maximize L w.r.t p, η and ξ. This combination defines our constraint optimization problem.

A.1 Derivation of Contextual HiREPS

We first reiterate the complete optimization problem, i.e.,

$$\begin{aligned} \max_{\pi,\mu^\pi} J(\pi) &= \max_{\pi,\mu^\pi} \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\mathcal{R}_{\boldsymbol{sa}}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\ \text{s.t.} \quad \epsilon &\geq \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \log \frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\ \tilde{\kappa} &\geq -\sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)p(o|\boldsymbol{s},\boldsymbol{a}) \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\ \hat{\boldsymbol{\phi}} &= \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\ 1 &= \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \end{aligned} \tag{28}$$

which includes an additional constraint to represent prior knowledge about the desired behavior of sub-policies, i.e., that sub-policies should spread out and not overlap. In the entropy constraint, we replace only the responsibility term inside the log with the approximation $\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})$, as we can combine the expectation over the entropy into $\mathbb{E}_{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})} \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a})$.

The Lagrangian formulation reads

$$\begin{aligned} L = \sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \Big[\mathcal{R}_{\boldsymbol{sa}} - \eta \log \frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})} \\ + \xi \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a}) - \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}) - \lambda\Big]\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} + \eta\epsilon + \lambda + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \xi\tilde{\kappa}, \end{aligned} \tag{29}$$

where η , $\boldsymbol{\theta}$, ξ and λ are Lagrangian parameters. To arrive at a closed form update rule for $\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})$, we differentiate L

$$\begin{aligned} \frac{\mathrm{d}L}{\mathrm{d}\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})} &= \mathcal{R}_{\boldsymbol{sa}} - \eta \log \frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})} + \xi \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a}) - \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}) - \lambda - \eta, \\ &= \mathcal{R}_{\boldsymbol{sa}} - \eta \log \frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}} - \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}) - \lambda - \eta, \end{aligned} \tag{30}$$

with

$$\xi \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a}) = \eta\frac{\xi}{\eta} \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a}) = \eta \log \tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{\frac{\xi}{\eta}} = -\eta \log \frac{1}{\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{\xi/\eta}}.$$

We set the derivative to zero and write $V(\boldsymbol{s}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s})$ to solve for the update rule

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) = q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} - V(\boldsymbol{s})}{\eta}\right) \exp\left(-1 - \frac{\lambda}{\eta}\right). \tag{31}$$

Here, we use the fact that

$$\sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} = 1, \tag{32}$$

and, thus,

$$\exp\left(-1-\frac{\lambda}{\eta}\right)^{-1} = \sum_{o\in\mathcal{O}} \iint q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} - V(\boldsymbol{s})}{\eta}\right) \mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}. \tag{33}$$

Hence, the term $\exp(-1-\lambda/\eta)$ acts as a normalization constant and $\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})$ can be written as

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) = \frac{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}-V(\boldsymbol{s})}{\eta}\right)}{\sum_{o\in\mathcal{O}}\iint q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}-V(\boldsymbol{s})}{\eta}\right)\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}}.$$

However, for improved clarity, we resort to writing the update equation in the proportional formulation

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \propto q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}-V(\boldsymbol{s})}{\eta}\right), \tag{34}$$

which depends only on the Lagrangian parameters ξ, η and $\boldsymbol{\theta}$ with $V(\boldsymbol{s}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s})$, but not on the parameter λ. The values for these Lagrangian parameters can be calculated by optimizing the dual problem. The dual formulation can be obtained by inserting Eq. (31) into the original Lagrangian formulation, i.e.,

$$\begin{aligned} g(\eta,\boldsymbol{\theta},\xi,\lambda) = \sum_{o\in\mathcal{O}}\iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\Big[\mathcal{R}_{\boldsymbol{sa}} \\ -\eta\log\frac{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}-V(\boldsymbol{s})}{\eta}\right)\exp\left(-1-\frac{\lambda}{\eta}\right)}{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}} \\ -\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s})-\lambda\Big]\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} + \eta\epsilon + \lambda + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \xi\tilde{\kappa}, \end{aligned} \tag{35}$$

which can easily be simplified to

$$g(\eta,\boldsymbol{\theta},\xi,\lambda) = \eta\epsilon + \lambda + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \xi\tilde{\kappa} + \sum_{o\in\mathcal{O}}\iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})\eta\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \tag{36}$$

where we again use Eq. (32) to simplify further to

$$g(\eta,\boldsymbol{\theta},\xi,\lambda) = \eta\epsilon + \lambda + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \xi\tilde{\kappa} + \eta. \tag{37}$$

At this point we rewrite Eq. (33) as

$$\eta + \lambda = \eta\log\sum_{o\in\mathcal{O}}\iint q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}-V(\boldsymbol{s})}{\eta}\right)\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \tag{38}$$

where we used $\log(a^{-1}) = -\log(a)$ and multiplied by η. We insert Eq. (38) back into Eq. (36) and get

$$\begin{aligned} g(\eta,\boldsymbol{\theta},\xi) =& \eta\log\sum_{o\in\mathcal{O}}\iint q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta}\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}-V(\boldsymbol{s})}{\eta}\right) \\ &+ \eta\epsilon + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \xi\tilde{\kappa}. \end{aligned} \tag{39}$$

The dual function $g(\eta,\boldsymbol{\theta},\xi)$ is formulated in such a way that it does not depend on the Lagrangian parameter λ, which acts as a normalization factor.

If we work on samples $\{\boldsymbol{s},\boldsymbol{a}\}_i$ with $i = 1,2,\dots N$, we have to divide by the generating distribution $q(\boldsymbol{s},\boldsymbol{a})$ and obtain

$$\begin{aligned} g(\eta,\boldsymbol{\theta},\xi) =& \eta\log\frac{1}{N}\sum_{o\in\mathcal{O}}\sum_{i=1}^{N}\tilde{p}(o|\boldsymbol{s}_i,\boldsymbol{a}_i)^{1+\xi/\eta}\exp\left(\frac{R_i - V(\boldsymbol{s}_i)}{\eta}\right) \\ &+ \eta\epsilon + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \xi\tilde{\kappa}, \end{aligned} \tag{40}$$

which we can optimize using standard optimization libraries.

A.2 Derivation of Skill Sequencing

As above, we first reiterate the complete optimization problem, i.e.,

$$\begin{aligned} \max_{\pi_k,\mu_k^\pi} J(\pi_{1:K}) &= \max_{\pi,\mu_k^\pi}\sum_{k=1}^{K}\sum_{o\in\mathcal{O}}\iint \mathcal{R}_{\boldsymbol{sa}}^k\pi_k(\boldsymbol{a}|\boldsymbol{s})\pi_k(o|\boldsymbol{s})\mu_k^\pi(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} + \int\mu_{K+1}^\pi(\boldsymbol{s})\mathcal{R}_{\boldsymbol{s}}^{K+1}\,\mathrm{d}\boldsymbol{s}, \\ \text{s. t. } : \epsilon &\geq \int\mu_{K+1}^\pi(\boldsymbol{s})\log\frac{\mu_{K+1}^\pi(\boldsymbol{s})}{q_{K+1}(\boldsymbol{s})}\,\mathrm{d}\boldsymbol{s}, \\ \hat{\boldsymbol{\phi}}_0 &= \int\boldsymbol{\phi}(\boldsymbol{s}')\mu_1^\pi(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}', \\ \forall k\leq K : \epsilon &\geq \sum_{o\in\mathcal{O}}\iint\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})\log\frac{\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})}{q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})}\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\ \tilde{\kappa} &\geq -\sum_{o\in\mathcal{O}}\iint\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)p_k(\boldsymbol{s},\boldsymbol{a})\pi_k(o|\boldsymbol{s})\log\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}, \\ \int\boldsymbol{\phi}(\boldsymbol{s}')\mu_{k+1}^\pi(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}' &= \sum_{o\in\mathcal{O}}\iiint\mathcal{P}_{\boldsymbol{ss}'}^{\boldsymbol{a}}\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{s}'\,\mathrm{d}\boldsymbol{a}, \\ \forall k : 1 &= \sum\nolimits_{o\in\mathcal{O}}\iint\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a},o|\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}. \end{aligned} \tag{41}$$

The Lagrangian formulation reads

$$\begin{aligned} L =& \boldsymbol{\theta}_1^T\hat{\boldsymbol{\phi}}_0 + \sum_{k=1}^{K}\eta_k\epsilon + \lambda_k + \int\boldsymbol{\theta}_{k+1}^T\boldsymbol{\phi}(\boldsymbol{s})\mu_k^\pi(\boldsymbol{s})\,\mathrm{d}\boldsymbol{s} + \xi_k\tilde{\kappa} + \sum_{o\in\mathcal{O}}\iint\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s}) \\ &\left[\mathcal{R}_{\boldsymbol{sa}}^k - \eta_k\log\frac{\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})}{q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})} + \xi_k\log\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a}) - \int\mathcal{P}_{\boldsymbol{ss}'}^{\boldsymbol{a}}\boldsymbol{\theta}_{k+1}^T\boldsymbol{\phi}(\boldsymbol{s}')\,\mathrm{d}\boldsymbol{s}' - \lambda_k\right]\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} \\ &+ \int\mu_{K+1}^\pi(\boldsymbol{s})\Big[\mathcal{R}_{\boldsymbol{s}}^{K+1} - \eta_{K+1}\log\frac{\mu_{K+1}^\pi(\boldsymbol{s})}{q_{K+1}(\boldsymbol{s})} - \lambda_{K+1}\Big]\,\mathrm{d}\boldsymbol{s} + \eta_{K+1}\epsilon + \lambda_{K+1}, \end{aligned} \tag{42}$$

where η_k , $\boldsymbol{\theta}_k$, ξ_k and λ_k are Lagrangian parameters. To arrive at a closed form update rule for $\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})$, we differentiate L

$$\begin{aligned}\frac{\mathrm{d}L}{\mathrm{d}\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})} =&\mathcal{R}_{\boldsymbol{sa}}^k - \eta_k \log \frac{\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})}{q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})} + \xi_k \log \tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a}) \\ &+ \int \mathcal{P}_{\boldsymbol{ss'}}^{\boldsymbol{a}} \boldsymbol{\theta}_{k+1}^T \boldsymbol{\phi}(\boldsymbol{s'})\,\mathrm{d}\boldsymbol{s'} - \boldsymbol{\theta}_k^T\boldsymbol{\phi}(\boldsymbol{s}) - \lambda_k - \eta_k, \\ =&\mathcal{R}_{\boldsymbol{sa}}^k - \eta_k \log \frac{\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})}{q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k}} \\ &+ \int \mathcal{P}_{\boldsymbol{ss'}}^{\boldsymbol{a}} \boldsymbol{\theta}_{k+1}^T \boldsymbol{\phi}(\boldsymbol{s'})\,\mathrm{d}\boldsymbol{s'} - \boldsymbol{\theta}_k^T\boldsymbol{\phi}(\boldsymbol{s}) - \lambda_k - \eta_k, \end{aligned} \tag{43}$$

with

$$\xi_k \log \tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a}) = \eta_k \frac{\xi_k}{\eta_k} \log \tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a}) = \eta_k \log \tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{\xi_k/\eta_k} = -\eta_k \log \frac{1}{\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{\xi_k/\eta_k}}$$

and set the derivative to zero to solve for the update rule

$$\begin{aligned}\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s}) =&q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k} \\ &\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}^k + \int \mathcal{P}_{\boldsymbol{ss'}}^{\boldsymbol{a}} \boldsymbol{\theta}_{k+1}^T \boldsymbol{\phi}(\boldsymbol{s'})\,\mathrm{d}\boldsymbol{s'} - \boldsymbol{\theta}_k^T\boldsymbol{\phi}(\boldsymbol{s})}{\eta_k}\right) \exp\left(-1 - \frac{\lambda_k}{\eta_k}\right). \end{aligned} \tag{44}$$

For improved readability we write $\mathbb{E}[V_{k+1}(\boldsymbol{s'})] = \int \mathcal{P}_{\boldsymbol{ss'}}^{\boldsymbol{a}} \boldsymbol{\theta}_{k+1}^T \boldsymbol{\phi}(\boldsymbol{s'})\,\mathrm{d}\boldsymbol{s'}$ and $V_k(\boldsymbol{s}) = \boldsymbol{\theta}_k^T\boldsymbol{\phi}(\boldsymbol{s})$. As before, we use

$$\sum_{o\in\mathcal{O}} \iint \mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a},o|\boldsymbol{s})\pi_k(o|\boldsymbol{s})\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} = 1, \tag{45}$$

and, thus, obtain

$$\begin{aligned}\exp\left(-1 - \frac{\lambda_k}{\eta_k}\right)^{-1} =&\sum_{o\in\mathcal{O}} \iint q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k} \\ &\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}^k + \mathbb{E}[V_{k+1}(\boldsymbol{s'})] - V_k(\boldsymbol{s})}{\eta_k}\right) \mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}. \end{aligned} \tag{46}$$

Hence, the term $\exp\left(-1 - \frac{\lambda_k}{\eta_k}\right)$ acts as a normalization constant and $\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})$ can be written as

$$\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s}) = \frac{q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}^k + \mathbb{E}[V_{k+1}(\boldsymbol{s'})] - V_k(\boldsymbol{s})}{\eta_k}\right)}{\sum_{o\in\mathcal{O}} \iint q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}^k + \mathbb{E}[V_{k+1}(\boldsymbol{s'})] - V_k(\boldsymbol{s})}{\eta_k}\right) \mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}}.$$

For improved clarity, we resort to writing the update equation in the proportional formulation

$$\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s}) \propto q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}^k + \mathbb{E}[V_{k+1}(\boldsymbol{s'})] - V_k(\boldsymbol{s})}{\eta}\right), \tag{47}$$

which depends only on the Lagrangian parameters ξ, η and $\boldsymbol{\theta}$ with $V(\boldsymbol{s}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s})$, but not on the parameter λ. The values for these Lagrangian parameters can be calculated by optimizing the dual problem.

As before, we set the solution for $\mu_k^\pi(\boldsymbol{s})\pi_k(\boldsymbol{a}|\boldsymbol{s},o)\pi_k(o|\boldsymbol{s})$ back into the Lagrangian and simplify to get a dual function that does not depend on the Lagrangian parameters λ_k, i.e.,

$$\begin{aligned}g(\eta_{1:K+1},\boldsymbol{\theta}_{1:K+1},\xi_{1:K+1}) =&\boldsymbol{\theta}_1^T\hat{\boldsymbol{\phi}}_0 + \sum_{k=1}^{K+1} \eta_k\epsilon + \xi_k\tilde{\kappa} + \eta_k \log \sum_{o\in\mathcal{O}} \iint q_k(\boldsymbol{s},\boldsymbol{a})\tilde{p}_k(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi_k/\eta_k} \\ &\exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}}^k + \mathbb{E}[V_{k+1}(\boldsymbol{s'})] - V_k(\boldsymbol{s})}{\eta_k}\right) \mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a}. \end{aligned} \tag{48}$$

Now, the dual function $g(\eta_{1:K+1},\boldsymbol{\theta}_{1:K+1},\xi_{1:K+1})$ is formulated in such a way that it does not depend on the Lagrangian parameters λ_k, which acts as a normalization factor.

A.3 Derivation of Infinite Horizon HiREPS

The derivation of the infinite horizon case follows the derivations given above, where the Lagrangian is given by

$$\begin{aligned}L =&\sum_{o\in\mathcal{O}} \iint \mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \Big[\mathcal{R}_{\boldsymbol{sa}} - \eta\log\frac{\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s})}{q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})} \\ &+\xi\log\tilde{p}(o|\boldsymbol{s},\boldsymbol{a}) - \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s}) - \lambda\Big]\,\mathrm{d}\boldsymbol{s}\,\mathrm{d}\boldsymbol{a} + \eta\epsilon + \lambda + \int \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{s'})\,\mathrm{d}\boldsymbol{s'} + \xi\tilde{\kappa}, \end{aligned} \tag{49}$$

and the resulting update rule and dual function are given as

$$\mu^\pi(\boldsymbol{s})\pi(\boldsymbol{a}|\boldsymbol{s},o)\pi(o|\boldsymbol{s}) \propto q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} + \mathbb{E}[V(\boldsymbol{s'})] - V(\boldsymbol{s})}{\eta}\right), \tag{50}$$

and

$$\begin{aligned}g(\eta,\boldsymbol{\theta},\xi) =&\eta\log\sum_{o\in\mathcal{O}} \iint q(\boldsymbol{s},\boldsymbol{a})\tilde{p}(o|\boldsymbol{s},\boldsymbol{a})^{1+\xi/\eta} \exp\left(\frac{\mathcal{R}_{\boldsymbol{sa}} + \mathbb{E}[V(\boldsymbol{s'})] - V(\boldsymbol{s})}{\eta}\right) \\ &+\eta\epsilon + \xi\tilde{\kappa}. \end{aligned} \tag{51}$$

As for the other cases we can find the minimum of the dual function using standard optimization libraries.

Journal of Machine Learning Research 17 (2016) 1-31 Submitted 7/15; Revised 1/16; Published 6/16

Convex Regression with Interpretable Sharp Partitions

Ashley Petersen AJPETE@UW.EDU
Noah Simon NRSIMON@UW.EDU
Department of Biostatistics
University of Washington
Seattle, WA 98195
Daniela Witten DWITTEN@UW.EDU
Departments of Biostatistics and Statistics
University of Washington
Seattle, WA 98195

Editor: Maya Gupta

Abstract

We consider the problem of predicting an outcome variable on the basis of a small number of covariates, using an interpretable yet non-additive model. We propose *convex regression with interpretable sharp partitions* (CRISP) for this task. CRISP partitions the covariate space into blocks in a data-adaptive way, and fits a mean model within each block. Unlike other partitioning methods, CRISP is fit using a non-greedy approach by solving a convex optimization problem, resulting in low-variance fits. We explore the properties of CRISP, and evaluate its performance in a simulation study and on a housing price data set.

Keywords: convex optimization, interpretability, non-additivity, non-parametric regression, prediction

1. Introduction

Classification and regression trees (CART) are immensely popular for flexible and non-additive predictive modeling, despite the fact that they date back more than thirty years (Breiman et al., 1984). The trees are fit using a two-stage process in which the tree is first greedily "grown" to some maximum size, and then "pruned" to avoid overfitting. The final tree with K terminal nodes can be visually displayed as a decision tree with $K-1$ splits, or equivalently as K disjoint boxes that completely partition the covariate space. CART has stood the test of time, because its output is highly interpretable and it can easily incorporate complex non-additive relationships between features. However, it is a greedy procedure, and a small perturbation of the data can produce a very different tree. The high variability of the fitted values can compromise the scientific utility of the tree, as well as the tree's prediction accuracy on test data. While an ensemble approach, like random forests, can reduce CART's variability, this comes at the expense of interpretability (Breiman, 2001).

Two other well-known methods for flexible and non-additive predictive modeling are multivariate adaptive regression splines (MARS) (Friedman, 1991) and thin-plate splines (TPS) (Duchon, 1977). The MARS fit is a weighted sum of basis functions, which are greedily chosen and some of which involve pairs of features. TPS fits the observed data, regularized by smoothness penalties. In the case of two covariates x_1 and x_2 and a response y, the TPS fit is the solution to

$$\underset{f}{\text{minimize}} \quad \sum_{i=1}^{n}[y_i - f(x_{1i}, x_{2i})]^2 + \lambda \int\int_{\mathbb{R}^2} \|\nabla^2 f(x_1, x_2)\|_F^2 \, dx_1 \, dx_2.$$

The fits from MARS and TPS are incredibly flexible, but can be less interpretable than the fits from CART.

In recent years, the statistical community has been very interested in formulating predictive models as solutions to convex optimization problems. However, to the best of our knowledge, no proposals have been made for flexible, non-additive, and interpretable modeling via convex optimization. To close this gap, we propose a non-greedy procedure whose fits have a block structure reminiscent of CART. Our proposal, *convex regression with interpretable sharp partitions* (CRISP), is the solution to a convex optimization problem with predictions that are much less variable than those of CART. Also unlike CART, CRISP borrows information across the blocks, and is able to adequately model the data when the mean model is smooth. Thus our method provides a compromise between the interpretability of CART and the flexibility of MARS and TPS. In this paper, we consider the low-dimensional setting in which there are a small number of covariates of interest ($p \ll n$). We leave an extension to the $p > n$ setting to future work.

CRISP has a number of attractive properties:

- CRISP can accommodate interactions between pairs of covariates in a flexible way. This is useful when the impact of one covariate may depend on the value of another covariate, but there is not strong *a priori* knowledge about the form of the interaction.
- CRISP fits a piecewise constant model, which is easily interpreted by even those with limited statistical background.
- CRISP is formulated as a convex optimization problem. Thus we can solve for the global optimum, and can derive an expression for CRISP's degrees of freedom.

The remainder of this paper is organized as follows. In Section 2, we introduce our method and present an algorithm to implement it. We compare our method to existing approaches using simulated data in Section 3. In Section 4, we derive some properties of the method. In Section 5, we discuss connections between our method and other work. We illustrate our method on a housing price data set in Section 6. We consider a modification to our proposal in Section 7, and close with the discussion in Section 8. Proofs are in the Appendix.

2. Convex Regression with Interpretable Sharp Partitions

Throughout most of this paper, for ease of exposition, we focus on the case of $p = 2$ features. An extension to the case of $p > 2$ is given in Section 7.

We first present an overview of the CRISP approach. We wish to predict a random variable $y \in \mathbb{R}$ using $x_1, x_2 \in \mathbb{R}$. We assume that $y = f(x_1, x_2) + \epsilon$, where ϵ is a mean-zero

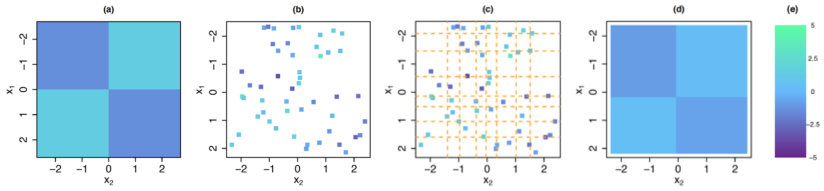


Figure 1: In *(a)*, the mean model $f(x_1, x_2)$ used to generate data. In *(b)*, each of the 50 squares represents an observation (x_1, x_2, y) with $y = f(x_1, x_2) + \epsilon$ with $\epsilon \sim N(0, 1)$. In *(c)*, there are $q^2 = 64$ bins of (x_1, x_2) values, whose boundaries coincide with the octiles (- - -) of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. In *(d)*, CRISP estimates $f(x_1, x_2)$ to be constant within each bin, and furthermore encourages adjacent bins to take on the same value. When applied to the data in *(b)* with $q = 8$, this leads to an estimated $f(x_1, x_2)$ with four *blocks*. In *(e)*, we show the heat scale legend.

error term, and f is an unknown function that we wish to estimate. An example of $f(x_1, x_2)$ is displayed in Figure 1(a). Figure 1(b) displays a training set of n i.i.d. observations of (x_1, x_2, y). We first partition the feature space into q^2 bins, as shown in Figure 1(c) with $q = 8$. The CRISP approach estimates $f(x_1, x_2)$ to be constant within each bin, and further encourages f to take on the same value at adjacent bins; this leads to constant-valued *blocks*. The CRISP output is shown in Figure 1(d); there are four estimated blocks. More details about this simulation set-up are provided in Section 3.

2.1 Notation and Goal of CRISP

We now introduce some new notation, and provide further intuition for CRISP, before presenting the optimization problem for CRISP in Section 2.2.

As is shown in Figure 1(c), we wish to estimate the mean model $f(x_1, x_2)$ for a $q \times q$ grid of bins, where $f(x_1, x_2)$ is estimated to be constant within each bin. Let $\boldsymbol{M} \in \mathbb{R}^{q \times q}$ denote a mean matrix whose element $M_{(i)(j)}$ contains the mean for pairs of covariate values within a *quantile range* of the observed predictors $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^n$. For example, $M_{(1)(2)}$ represents the mean of the observations with x_1 less than the $\frac{1}{q}$-quantile of $\boldsymbol{x}_1$, and x_2 between the $\frac{1}{q}$- and $\frac{2}{q}$-quantiles of $\boldsymbol{x}_2$. In Figure 1(c), the corner grid bins correspond to $M_{(1)(1)}$, $M_{(8)(1)}$, $M_{(8)(8)}$, and $M_{(1)(8)}$, starting at the top-left corner of the grid and moving counter-clockwise. In CRISP, our goal is to estimate the $q \times q$ matrix $\boldsymbol{M}$ on the basis of $\boldsymbol{y} \in \mathbb{R}^n$, which contains n noisy observations from various bins of $\boldsymbol{M}$.

In the example shown in Figure 1, we partition the feature space into an 8×8 grid (shown in Figure 1(c)), which translates to estimating an 8×8 matrix $\boldsymbol{M}$. Therefore, instead of estimating $f(x_1, x_2)$ over the entire joint range of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we need only estimate the 64 elements of $\boldsymbol{M}$. Furthermore, CRISP borrows information across bins of the grid by encouraging pairs of neighboring rows and columns of $\boldsymbol{M}^*$ to be equal, leading

to an estimated mean model with a *block structure*. For instance, in Figure 1(d), $\boldsymbol{M}$ is estimated to have four *blocks*, or regions of feature space over which $f(x_1, x_2)$ is constant. Consequently, the CRISP solution $\boldsymbol{M}^*$ shown in Figure 1(d) only has 4 unique elements, while $\boldsymbol{M}$ is an 8×8 matrix. If we examined the estimate $\boldsymbol{M}^*$, we would see that all pairs of neighboring rows and neighboring columns of $\boldsymbol{M}^*$ are identical, except for one pair of columns and one pair of rows.

While the true mean model in this example has a block structure (as seen in Figure 1(a)), we will show in Section 3 that CRISP can perform well even when the true mean model is smooth. The data in this example were uniformly distributed in the covariate space. CRISP is most suitable for data applications where observations are distributed throughout the covariate space. Highly correlated covariates will lead to an insufficient amount of data to estimate the mean model over the entire covariate space.

2.2 The Optimization Problem

The CRISP optimization problem balances the trade-off between fitting the data and encouraging a block structure. We estimate $\boldsymbol{M}$ by solving the convex optimization problem

$$\underset{\boldsymbol{M} \in \mathbb{R}^{q \times q}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{n} (y_i - \Omega(\boldsymbol{M}, x_{1i}, x_{2i}))^2 + \lambda P(\boldsymbol{M}). \tag{1}$$

In (1), the function Ω extracts the element of $\boldsymbol{M}$ corresponding to the bin to which the observation (x_{1i}, x_{2i}) belongs. For instance, in Figure 1(c), $\Omega(\boldsymbol{M}, 0, -1) = M_{(4)(2)}$. Note that Ω is explicitly defined in Appendix A. Furthermore, $\lambda \geq 0$ is a tuning parameter, and the penalty P is defined as

$$P(\boldsymbol{M}) = \sum_{i=1}^{q-1} \left[\left\| \boldsymbol{M}_{i\cdot} - \boldsymbol{M}_{(i+1)\cdot} \right\|_2 + \left\| \boldsymbol{M}_{\cdot i} - \boldsymbol{M}_{\cdot(i+1)} \right\|_2 \right], \tag{2}$$

where $\boldsymbol{M}_{i\cdot}$ and $\boldsymbol{M}_{\cdot i}$ denote the ith row and column of $\boldsymbol{M}$, respectively. The sum of squared errors in (1) encourages the estimate of $\boldsymbol{M}$ to fit the data, while the group lasso penalty (Yuan and Lin, 2006) in (2) encourages pairs of neighboring rows (or columns) to be exactly identical. This leads to the formation of constant-valued blocks, which are comprised of multiple bins of the $q \times q$ grid. Appendix B discusses other possible penalties that could be used in (1).

We now rewrite (1) in a way that will be useful later. We introduce a vectorized form of $\boldsymbol{M}$, which is denoted by $\boldsymbol{m} = \text{vec}(\boldsymbol{M}) = \left((\boldsymbol{M}_{\cdot 1})^T, (\boldsymbol{M}_{\cdot 2})^T, \cdots, (\boldsymbol{M}_{\cdot q})^T\right)^T$ where $\boldsymbol{M}_{\cdot i}$ is the ith column of $\boldsymbol{M}$. The correspondence between $\boldsymbol{M}$ and $\boldsymbol{m}$ is shown in Figure 10 of Appendix A. In what follows, we will switch between using the matrix $\boldsymbol{M}$ and the vectorized $\boldsymbol{m}$. Then (1) can be rewritten as

$$\underset{\boldsymbol{m} \in \mathbb{R}^{q^2}}{\text{minimize}} \quad \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{m}\|_2^2 + \lambda \sum_{i=1}^{q-1} \left[\|\boldsymbol{R}_i \boldsymbol{m}\|_2 + \|\boldsymbol{C}_i \boldsymbol{m}\|_2 \right], \tag{3}$$

where each row of $\boldsymbol{Q} \in \mathbb{R}^{n \times q^2}$ contains $q^2 - 1$ elements that equal 0, and a single 1, such that $\boldsymbol{Q}_{i\cdot}\boldsymbol{m} = \Omega(\boldsymbol{M}, x_{1i}, x_{2i})$, where $\boldsymbol{Q}_{i\cdot}$ indicates the ith row of $\boldsymbol{Q}$. (Though $\boldsymbol{Q}$ is a function

of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we suppress this to simplify the notation.) In (3), $\boldsymbol{R}_i, \boldsymbol{C}_i \in \mathbb{R}^{q \times q^2}$ extract differences between neighboring rows and columns of $\boldsymbol{M}$ (i.e., $\boldsymbol{R}_i\boldsymbol{m} = \boldsymbol{M}_{i\cdot} - \boldsymbol{M}_{(i+1)\cdot}$ and $\boldsymbol{C}_i\boldsymbol{m} = \boldsymbol{M}_{\cdot i} - \boldsymbol{M}_{\cdot(i+1)}$). An example of $\boldsymbol{Q}$ and explicit definitions of $\boldsymbol{Q}$, $\boldsymbol{R}_i$, and $\boldsymbol{C}_i$ are in Appendix A. We let $\boldsymbol{A} = \left(\boldsymbol{R}_1^T, \cdots, \boldsymbol{R}_{q-1}^T, \boldsymbol{C}_1^T, \cdots, \boldsymbol{C}_{q-1}^T\right)^T \in \mathbb{R}^{2q(q-1)\times q^2}$, and then rewrite (3) as

$$\underset{\boldsymbol{m}\in\mathbb{R}^{q^2}, \boldsymbol{z}\in\mathbb{R}^{2q(q-1)}}{\text{minimize}} \quad \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{m}\|_2^2 + \lambda \sum_{i=1}^{q-1} [\|\boldsymbol{z}_{1i}\|_2 + \|\boldsymbol{z}_{2i}\|_2] \quad \text{subject to } \boldsymbol{A}\boldsymbol{m} = \boldsymbol{z}, \tag{4}$$

where $\boldsymbol{z} = \left((\boldsymbol{z}_{11})^T, \cdots, (\boldsymbol{z}_{1(q-1)})^T, (\boldsymbol{z}_{21})^T, \cdots, (\boldsymbol{z}_{2(q-1)})^T\right)^T$ with $\boldsymbol{z}_{1i}, \boldsymbol{z}_{2i} \in \mathbb{R}^q$.

While (1), (3), and (4) have the same solution, it is most convenient to derive an algorithm to solve CRISP using the parameterization in (4). Throughout this paper, we will alternate between using the notation $\boldsymbol{M}^*$ and $\boldsymbol{m}^*$, where $\boldsymbol{m}^* = \text{vec}(\boldsymbol{M}^*)$, to represent the CRISP solution to (4). The training set predictions for CRISP are given by $\hat{\boldsymbol{y}} = \boldsymbol{Q}\boldsymbol{m}^*$.

2.3 An Algorithm for CRISP

We solve for the global optimum of the convex optimization problem (4) using the *alternating directions method of multipliers* (ADMM) algorithm (Boyd et al., 2011). This is summarized in Algorithm 1. Additional details are in Appendix C.

Algorithm 1 — Alternating Directions Method of Multipliers for Equation (4)

1. Let $\boldsymbol{u} = \left((\boldsymbol{u}_{11})^T, \ldots, (\boldsymbol{u}_{1(q-1)})^T, (\boldsymbol{u}_{21})^T, \ldots, (\boldsymbol{u}_{2(q-1)})^T\right)^T$ denote the scaled dual variables. Initialize $\boldsymbol{m}^{(0)} := \boldsymbol{0}, \boldsymbol{z}^{(0)} := \boldsymbol{0}$, and $\boldsymbol{u}^{(0)} := \boldsymbol{0}$.

2. For $k = 1, 2, \ldots$, until the primal and dual residuals satisfy a stopping criterion:

   (a) $\boldsymbol{m}^{(k)} := \left[\boldsymbol{Q}^T\boldsymbol{Q} + \rho\boldsymbol{A}^T\boldsymbol{A}\right]^{-1}\left[\boldsymbol{Q}^T\boldsymbol{y} + \rho\boldsymbol{A}^T(\boldsymbol{z}^{(k-1)} - \boldsymbol{u}^{(k-1)})\right]$

   (b) $\boldsymbol{z}_{1i}^{(k)} := (\boldsymbol{R}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{1i}^{(k-1)})(1 - \lambda/(\rho\|\boldsymbol{R}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{1i}^{(k-1)}\|_2))_+$,
   $\boldsymbol{z}_{2i}^{(k)} := (\boldsymbol{C}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{2i}^{(k-1)})(1 - \lambda/(\rho\|\boldsymbol{C}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{2i}^{(k-1)}\|_2))_+$ for $i = 1, \ldots, q-1$

   (c) $\boldsymbol{u}^{(k)} := \boldsymbol{u}^{(k-1)} + \boldsymbol{A}\boldsymbol{m}^{(k)} - \boldsymbol{z}^{(k)}$

In Algorithm 1, the computational bottleneck occurs in Step 2(a). Evaluating the q-banded matrix $\boldsymbol{Q}^T\boldsymbol{Q} + \rho\boldsymbol{A}^T\boldsymbol{A}$ has a one-time cost of $\mathcal{O}(n+q^4)$ operations, and computing its LU factorization requires an additional $\mathcal{O}(q^4)$ operations. Then Step 2(a) can be performed in $\mathcal{O}(q^3)$ operations (Boyd and Vandenberghe, 2004). Therefore, Algorithm 1 requires an initial step of $\mathcal{O}(n + q^4)$ operations, followed by a per-iteration complexity of $\mathcal{O}(q^3)$.

On a Macbook Pro with a 2.0 GHz Intel Sandy Bridge Core i7 processor, our `Python` implementation of CRISP with $n = q = 50$ takes 20.1 seconds for a sequence of 20 λ values. For $n = q = 100$ and $n = q = 200$, the run times are 84.7 and 383.6 seconds, respectively. Increasing n while holding q constant has little effect on the run times; this is consistent with the discussion in the previous paragraph. Thus even for very large n, the computational time is reasonable.

We chose to solve CRISP using an ADMM algorithm, as ADMM works well in related problems. For example, in the context of trend filtering, Ramdas and Tibshirani (forthcoming) found that their ADMM implementation converged more reliably across a variety of tuning parameter values and sample sizes than the primal-dual interior point method of Kim et al. (2009). In our setting, an interior point algorithm for CRISP involves solving a dense system of equations at each iteration, which has a computational complexity of $\mathcal{O}(q^6)$. Additionally, an interior point method would not recover the exact block structure (any strictly feasible solution would have no zero row or column differences). In contrast, we directly recover the block structure of our estimated mean model from the $\boldsymbol{z}$ variables of our ADMM algorithm. Furthermore, ADMM algorithms typically converge to moderate accuracy within only tens of iterations (Boyd et al., 2011), which is acceptable in our setting.

The value of λ can be chosen using K-fold cross-validation. Alternatively, λ can be selected using approaches based on Akaike's information criterion (AIC; Akaike, 1973) or Bayesian information criterion (BIC; Schwarz, 1978) using the degrees of freedom estimator proposed in Section 4.1. The roles of λ and q in controlling the granularity of the model are further characterized in Sections 4.2 and 4.3.

3. Simulations

In this section, we compare the performance of CRISP to CART, TPS, and competing methods. We consider a variety of mean models, as well as smaller ($n = 100$) and larger ($n = 10,000$) training set sample sizes.

3.1 Methods

We generate data with either $n = 100$ or $n = 10,000$, and $p = 2$. We independently sample each element of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ from a Unif$[-2.5, 2.5]$ distribution, and then take $\boldsymbol{y} = f(\boldsymbol{x}_1, \boldsymbol{x}_2) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \text{MVN}(\boldsymbol{0}, \sigma^2\boldsymbol{I}_n)$ with $\sigma = 1$ for $n = 100$ and $\sigma = 10$ for $n = 10,000$. Note that we use the notation MVN to indicate a multivariate normal distribution.

We consider four mean models for $f(x_1, x_2)$; these are displayed in the top panel of Figure 2, and defined in detail in Appendix D. In Scenario 1, the mean model is additive in $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Scenario 2 is similar to Scenario 1, but the mean model is non-additive. The mean model in Scenario 3 is piecewise constant, with the cut points for $\boldsymbol{x}_2$ depending on $\boldsymbol{x}_1$. Finally, Scenario 4 is a smooth mean model.

For each scenario, we generate 200 data sets and estimate $\boldsymbol{M}$ using CRISP (with $q = 100$) and several competitors: FLAM (implemented with the `R` package `flam` (Petersen, 2014)); CART (implemented with the `R` package `rpart` (Therneau et al., 2014)); TPS (implemented with the `R` package `fields` (Nychka et al., 2014)); a linear model with predictors $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and their interaction; and an "oracle" linear model based on knowing *a priori* which regions of the mean model take on a constant value.

For each of the four scenarios, we plot mean squared prediction error[1] versus degrees of freedom (a notion that will be discussed extensively in Section 4.1). CRISP and FLAM are fit over a sequence of exponentially decreasing λ values, with the degrees of freedom estimated using (6) and a result from Petersen et al. (forthcoming), respectively. TPS is fit over a sequence of degrees of freedom. For CART, we vary the number of terminal nodes in the tree, and average the estimator (7) over the replicates in order to estimate the degrees of freedom for each number of terminal nodes. Note that the number of degrees of freedom of CART is non-monotonic for small numbers of terminal nodes (as seen in Figure 3).

3.2 Results for $n = 100$

Results are shown in Figure 3. We see that both CRISP and TPS perform reasonably well in terms of prediction error in all scenarios, regardless of the true mean model. FLAM outperforms the other methods in Scenario 1, which is unsurprising as the mean model is truly additive, and FLAM boils down to CRISP with an additivity constraint (Section 5.2). However, FLAM performs poorly for mean models with substantial non-additivity (Scenarios 2 and 4). Outside of Scenario 1, CART performs worse than TPS and CRISP. CRISP, TPS, and CART all perform better than a linear model with an interaction in Scenarios 1–3. However, in Scenario 4, the mean model is well-approximated using a linear model. We also fit MARS for all scenarios; however, performance was poor and the results are omitted.

While CRISP and TPS have comparable prediction error, their fits are quite different. In Figure 2, we show the estimated mean models for CRISP, TPS, and CART for a single replicate of data in each scenario. CRISP provides fits that reflect the true mean model well, even when the true mean model is smooth. While TPS has low prediction error, the smooth fits from TPS are not easily interpreted and are far from the true mean model in some scenarios. While the fits from CART reflect the mean model reasonably well in Scenarios 1 and 2, the fits from CART in all scenarios are highly variable. CART fits from different replicates of Scenario 4 are shown in Figure 4. The average variance of an element of $\boldsymbol{M}^*$ across the 200 replicates for Scenario 4 was 0.843 for CART, compared to 0.0935 for CRISP and 0.0653 for TPS. The variance of CART's fitted values is similarly inflated for the other scenarios. Small perturbations of the data can produce very different qualitative conclusions when examining CART's fits.

3.3 Results for $n = 10,000$

We compare CRISP to TPS and CART. Results are in Figures 2 and 5. Again, CRISP performs well in all scenarios, and the CART fits are much more variable than those of CRISP and TPS. The average variance of an element of $\boldsymbol{M}^*$ across the 200 replicates for Scenario 1 was 0.111 for CART, compared to 0.051 for CRISP and 0.083 for TPS. For Scenario 2, the average variance was 1.42 for CART, compared to 0.056 for CRISP and 0.083 for TPS. For Scenario 3, the average variance was 0.692 for CART, compared to 0.077 for CRISP and 0.129 for TPS. And finally, for Scenario 4, the average variance was 1.89 for

1. Mean squared prediction error is defined as $\frac{1}{q^2}\|\boldsymbol{M}-\boldsymbol{M}^*\|_F^2$, where $\boldsymbol{M} \in \mathbb{R}^{q\times q}$ is the true mean matrix and $\boldsymbol{M}^* \in \mathbb{R}^{q\times q}$ is the estimate from a given method. For methods other than CRISP, $\boldsymbol{M}^*$ was constructed using the mean model estimate at the midpoint of each bin of the $q \times q$ grid.

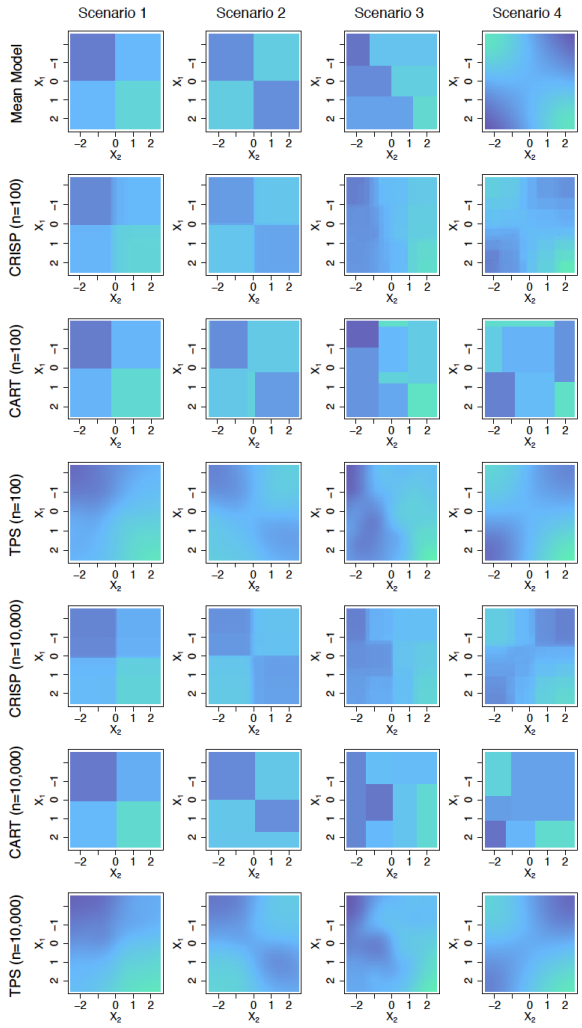


Figure 2: The mean models for Scenarios 1–4, as well as estimated mean models from CRISP, CART, and TPS for the simulations considered in Section 3. Each fit is from a single replicate of data, with the number of degrees of freedom indicated in Figures 3 and 5 for $n = 100$ and $n = 10,000$, respectively. The heat scale legend is in Figure 1(e).

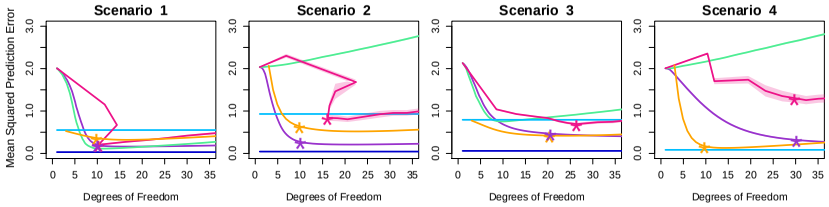


Figure 3: Mean squared prediction error, as a function of the degrees of freedom, for the four scenarios considered in the simulations of Section 3.2. The methods displayed are CRISP (—), FLAM (—), TPS (—), CART (—), linear model with an interaction (—), and the oracle linear model (—). The oracle linear model is only fit for Scenarios 1–3, for which the mean models have constant regions. Shaded bands (only visible for CART) indicate point-wise 95% confidence intervals over the 200 replicate data sets. The linear models have a fixed number of degrees of freedom, but are shown as horizontal lines. Asterisks indicate the degrees of freedom used for the fits shown in Figure 2.

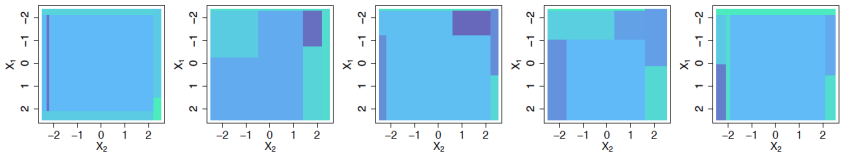


Figure 4: Fits for CART in Scenario 4 with $n = 100$ (as also shown in Figure 2) corresponding to five additional replicates of data. The heat scale legend is in Figure 1(e).

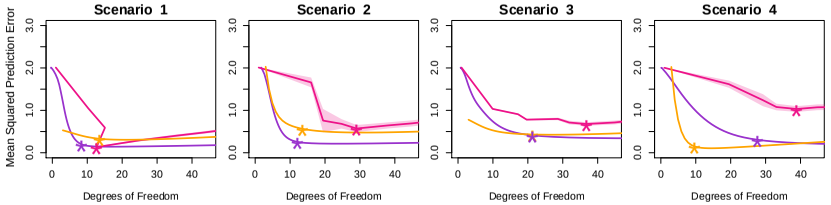


Figure 5: Results for $n = 10,000$ for CRISP (—), TPS (—), and CART (—) in the simulations of Section 3.3. Details are as given in Figure 3.

CART, compared to 0.096 for CRISP and 0.061 for TPS. Notably, a large sample size is not sufficient for producing stable CART fits, unless the signal-to-noise ratio is suitably large.

4. Properties of CRISP

In this section, we provide an unbiased estimator for CRISP's degrees of freedom. We also derive an analytical expression for the range of λ for which the solution to (4) takes a constant value, $\boldsymbol{m}^* = \left(\frac{1}{n}\mathbf{1}^T\boldsymbol{y}\right)\mathbf{1}$. Lastly, we discuss the role of q and λ in controlling the granularity of CRISP. Throughout this section, we use $\boldsymbol{A}^+$ to denote the Moore-Penrose pseudoinverse of a matrix $\boldsymbol{A}$.

4.1 Degrees of Freedom

Suppose that $\mathrm{Var}(\boldsymbol{y}) = \sigma^2\boldsymbol{I}$, and let $g(\boldsymbol{y}) = \hat{\boldsymbol{y}}$ denote the fit corresponding to some model-fitting procedure g. Then the degrees of freedom of g is defined as $\frac{1}{\sigma^2}\sum_{i=1}^n \mathrm{Cov}(y_i, \hat{y}_i)$ (Hastie and Tibshirani, 1990; Efron, 1986).

The concept of degrees of freedom provides a common framework for comparing the complexities of various models; this is particularly useful when the models under consideration are complex or unrelated. Ye (1998) proposed a computationally-burdensome Monte Carlo approach for estimating the degrees of freedom of a model-fitting procedure. In recent years, unbiased estimators for the degrees of freedom have been derived for the lasso and generalized lasso (Zou et al., 2007; Tibshirani and Taylor, 2012), among other methods. These estimators allow us to characterize a model's complexity, and also can be used in order to develop an approach for tuning parameter selection based on Akaike's information criterion (AIC; Akaike, 1973) or Bayesian information criterion (BIC; Schwarz, 1978).

Problem (3) is equivalent to the problem

$$\underset{\boldsymbol{m}}{\text{minimize}} \quad \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{m}\|_2^2 + \lambda\sum_{i=1}^{q-1}\left[\|\boldsymbol{R}_i\boldsymbol{m}\|_2 + \|\boldsymbol{C}_i\boldsymbol{m}\|_2\right] + \frac{\gamma}{2}\|\boldsymbol{m}\|_2^2 \tag{5}$$

with $\gamma = 0$. In the rest of this section, we take γ to be a small positive constant, which ensures strong convexity and enforces uniqueness of the solution.

We now introduce some notation. First, we define $\mathcal{C}$, the set of difference matrices corresponding to equal neighboring rows or columns in the solution $\boldsymbol{m}^*$ to (5). That is, $\mathcal{C} = \{\boldsymbol{A}_i : \|\boldsymbol{A}_i\boldsymbol{m}^*\|_2 = 0\}$ where $\boldsymbol{A}_1 = \boldsymbol{R}_1, \boldsymbol{A}_2 = \boldsymbol{R}_2, \ldots, \boldsymbol{A}_{q-1} = \boldsymbol{R}_{q-1}, \boldsymbol{A}_q = \boldsymbol{C}_1, \boldsymbol{A}_{q+1} = \boldsymbol{C}_2, \ldots, \boldsymbol{A}_{2q-2} = \boldsymbol{C}_{q-1}$. Then we define $\boldsymbol{A}_*$ to be the submatrix of $\boldsymbol{A}$ obtained by retaining only the rows of $\boldsymbol{A}$ corresponding to matrices $\boldsymbol{A}_i \in \mathcal{C}$. Note that $\boldsymbol{A}_* \in \mathbb{R}^{q|\mathcal{C}|\times q^2}$. We propose to estimate the degrees of freedom of CRISP as

$$\hat{df}_{CRISP} = \text{Tr}\left[\boldsymbol{Q}\left(\boldsymbol{D} + \lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}} S_2(\boldsymbol{A}_i, \boldsymbol{m}^*)\boldsymbol{P} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{P}\boldsymbol{Q}^T\right], \tag{6}$$

where $\boldsymbol{P} = \boldsymbol{I}_{q^2} - \boldsymbol{A}_*^+\boldsymbol{A}_*$, $S_2(\boldsymbol{A}_i, \boldsymbol{m}^*) = \frac{\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2} - \frac{\boldsymbol{A}_i^T\boldsymbol{A}_i\boldsymbol{m}^*\boldsymbol{m}^{*T}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2^3}$, and $\boldsymbol{Q}$ was defined in (3). Recall that $\boldsymbol{M}^*$ will tend to contain row-column blocks of constant value, as shown in Figure 1(d). We define $\boldsymbol{D} = \text{diag}\left(h(m_1^*), \cdots, h(m_{q^2}^*)\right)$, where $h(m_i^*)$ is the ratio of the number of observations in the block of $\boldsymbol{M}^*$ that contains m_i^* to the number of elements of $\boldsymbol{M}^*$ in the block of $\boldsymbol{M}^*$ that contains m_i^*. We use the notation MVN to indicate a multivariate normal distribution.

Proposition 1 *Assume $\boldsymbol{y} \sim MVN(\boldsymbol{\mu}, \sigma^2\boldsymbol{I})$. Then $\hat{df}_{CRISP}$ is an unbiased estimator of the degrees of freedom of CRISP.*

The following corollary indicates that the estimator (6) simplifies substantially when the CRISP solution takes a particular form.

Corollary 2 *Assume $\boldsymbol{y} \sim MVN(\boldsymbol{\mu}, \sigma^2\boldsymbol{I})$. If either all rows or all columns of $\boldsymbol{M}^*$ are equal, then the total number of blocks of $\boldsymbol{M}^*$ is an unbiased estimator of the degrees of freedom.*

In 100 replicate data sets with $y_i \sim N(\mu_i, \sigma^2)$, we compare the mean of (6) to the mean of

$$\frac{1}{\sigma^2}\sum_{i=1}^{n}(\hat{y}_i - \mu_i)(y_i - \mu_i), \tag{7}$$

which provides a Monte Carlo estimate of $\frac{1}{\sigma^2}\sum_{i=1}^{n}\text{Cov}(y_i, \hat{y}_i)$, the true degrees of freedom of CRISP. The results in Figure 6(a) empirically validate Proposition 1, showing that (6) is an unbiased estimator of CRISP's degrees of freedom. Note that the proofs of Proposition 1 and Corollary 2 can be found in Appendices E and F, respectively.

4.2 Range of λ that Yields a Constant Solution

CRISP has a single tuning parameter λ, which we typically will select via cross-validation or a related approach. Here, we derive the minimum value of λ such that $\boldsymbol{m}^* = \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}$, corresponding to a fit in which all elements of $\boldsymbol{m}^*$ are equal.

Lemma 3 *The solution to (4) is constant (i.e., $\boldsymbol{m}^* = \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}$) if and only if*

$$\lambda \geq \max_{1\leq i\leq q-1}\{\|\boldsymbol{d}_{1i}^*\|_2, \|\boldsymbol{d}_{2i}^*\|_2\},$$

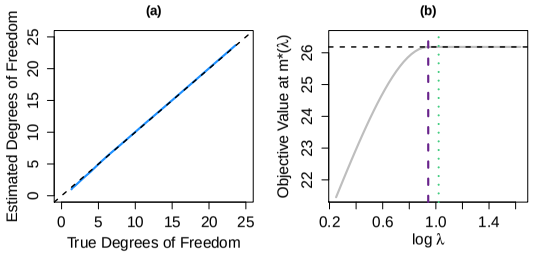


Figure 6: In *(a)*, we compare the degrees of freedom calculated using our estimator (7) (y-axis) from Section 4.1 to the unbiased, Monte Carlo estimator (6) (x-axis). Varying λ gives the solid line, and the dashed line indicates $y = x$. In *(b)*, we plot the value of the objective of (4) at $\boldsymbol{m}^*(\lambda)$, the minimizer of (4) at λ, for a replicate of data as λ varies. We compare two ways of finding a λ large enough such that $\boldsymbol{m}^*(\lambda) = \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}$, which results in the objective shown as - - -. We take $\lambda = \max_{1\leq i\leq q-1}\{\|\boldsymbol{d}_{1i}\|_2, \|\boldsymbol{d}_{2i}\|_2\}$ with either $\boldsymbol{d}$ being the solution to (8) (— —) or $\boldsymbol{d} = (\boldsymbol{A}^T)^+\boldsymbol{Q}^T\left(\boldsymbol{y} - \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}\right)$ (· - - ·). The former (— —) matches the result of Lemma 3 in Section 4.2.

where $\boldsymbol{d}^* = (\boldsymbol{d}_{11}^{*T} \cdots \boldsymbol{d}_{1(q-1)}^{*T}\ \boldsymbol{d}_{21}^{*T} \ldots \boldsymbol{d}_{2(q-1)}^{*T})^T$ is the solution to

$$\underset{\boldsymbol{d}}{\text{minimize}} \quad \max_{1\leq i\leq q-1}\{\|\boldsymbol{d}_{1i}\|_2, \|\boldsymbol{d}_{2i}\|_2\} \quad \textit{subject to} \quad \boldsymbol{Q}^T\left(\boldsymbol{y} - \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}\right) = \boldsymbol{A}^T\boldsymbol{d}. \tag{8}$$

Recall that the matrix $\boldsymbol{Q}$ was defined in (3). Taking $\lambda = \max_{1\leq i\leq q-1}\left\{\|\tilde{\boldsymbol{d}}_{1i}\|_2, \|\tilde{\boldsymbol{d}}_{2i}\|_2\right\}$ for any feasible vector $\tilde{\boldsymbol{d}}$ for (8) will give a value of λ sufficiently large so $\boldsymbol{m}^*$ is constant. For example, we can choose $\tilde{\boldsymbol{d}} = (\boldsymbol{A}^T)^+\boldsymbol{Q}^T\left(\boldsymbol{y} - \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}\right)$. However, choosing λ in accordance with Lemma 3 will give the minimum value of λ such that $\boldsymbol{m}^* = \left(\frac{1}{n}\boldsymbol{1}^T\boldsymbol{y}\right)\boldsymbol{1}$. The optimization problem (8) can be solved using a standard convex solver, such as SDPT3 via `CVX` in `MATLAB` (Grant and Boyd, 2008, 2014). An illustration of Lemma 3 is provided in Figure 6(b).

4.3 Controlling the Granularity of CRISP

Both q and λ control the granularity of the final CRISP model: q controls the size of the grid used to construct $\boldsymbol{M}$, and λ controls the number of blocks in the final fitted CRISP model. For a range of very small λ values, there will be q^2 blocks; for larger λ values, the CRISP solution will have a smaller number of blocks.

Given that q and λ both influence the number of blocks in the final fitted CRISP model, one might wonder whether it is necessary to have both q and λ. We illustrate the value of both q and λ through some simple examples.

4.3.1 Choice of q

In principle, q may be chosen to equal n. This means that each bin of the $q \times q$ grid would contain at most one observation. However, when n is large, choosing $q = n$ can lead to excessive computational time, memory burden, and variance in the fit. Instead, we aim to choose q to be large enough to allow for adequate granularity, but not excessively large. What constitutes adequate granularity will depend on the context of the problem.

In our analyses, we choose to treat q as a fixed parameter that is chosen prior to fitting CRISP. However, if desired, q could be chosen by K-fold cross validation.

4.3.2 Choice of λ

To illustrate the role of λ, consider taking $\lambda = 0$ in (3), and treating q as a tuning parameter rather than a fixed value. When $\lambda = 0$, (3) contains only a sum of squared errors term, so the estimate within each bin is the mean value of the observations in that bin. For bins without any observations, we estimate the corresponding element of $\boldsymbol{M}$ to be the overall mean of $\boldsymbol{y}$.

For the mean models shown in Figure 2, we compare CRISP to (3) with $\lambda = 0$ and q chosen adaptively. We focus on the general findings here, but detailed results are given in Appendix H. When the true mean model is piecewise constant with boundaries that are well-approximated by a grid of bins (as in Scenarios 1–3), CRISP and (3) with $\lambda = 0$ and variable q perform similarly. However, CRISP is clearly superior at estimating the smooth mean model of Scenario 4 (Figure 12), as it is able to borrow information across bins, instead of simply fitting the mean of observations within each bin. CRISP also allows the granularity of the fitted model to vary adaptively over the covariate space, as shown in Figure 13(a) of Appendix H. The blocks of this mean model perfectly align with a grid that has $q = 3$, but the mean model only has 4 blocks. While (3) with $\lambda = 0$ and $q = 3$ fits 9 blocks, CRISP correctly identifies 4 blocks (Figures 13(b) and 13(c) of Appendix H).

5. Connections to Other Methods

In this section, we establish connections between CRISP and two previous proposals.

5.1 Connection to One-Dimensional Fused Lasso

Suppose that for a given value of λ, the CRISP fit involves only one covariate: that is, $\boldsymbol{M}^* = \tilde{\boldsymbol{m}}\mathbf{1}_q^T$ or $\boldsymbol{M}^* = \mathbf{1}_q\tilde{\boldsymbol{m}}^T$ for some $\tilde{\boldsymbol{m}} \in \mathbb{R}^q$. We will now show that in this setting, the CRISP solution can be recovered by solving a one-dimensional fused lasso problem (Tibshirani et al., 2005).

Before presenting Lemma 4, we introduce some notation. Define $\boldsymbol{D} = [\boldsymbol{I}_{(q-1)\times(q-1)}\ \mathbf{0}_{(q-1)\times 1}] - [\mathbf{0}_{(q-1)\times 1}\ \boldsymbol{I}_{(q-1)\times(q-1)}]$ to be the first difference matrix. Define $\tilde{\boldsymbol{y}} \in \mathbb{R}^q$ such that $\tilde{y}_i$ is the mean outcome value of the observations in the ith row of the $q \times q$ grid used to construct $\boldsymbol{M}$. Let n_i denote the number of observations in the ith row of the $q \times q$ grid used to construct $\boldsymbol{M}$. Define $\boldsymbol{W} \in \mathbb{R}^{q\times q}$ to be the diagonal matrix with entries $\sqrt{n_1}, \sqrt{n_2}, \ldots, \sqrt{n_q}$.

Lemma 4 *Suppose that, for some value of λ, the CRISP solution is of the form $\boldsymbol{M}^* = \tilde{\boldsymbol{m}}\mathbf{1}_q^T$ for some $\tilde{\boldsymbol{m}} \in \mathbb{R}^q$. Then $\tilde{\boldsymbol{m}}$ is the solution to the problem*

$$\underset{\tilde{\boldsymbol{m}}\in\mathbb{R}^q}{\text{minimize}} \quad \frac{1}{2}\left\|\boldsymbol{W}(\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{m}})\right\|_2^2 + \lambda\sqrt{q}\left\|\boldsymbol{D}\tilde{\boldsymbol{m}}\right\|_1. \tag{9}$$

If instead $\boldsymbol{M}^* = \mathbf{1}_q\tilde{\boldsymbol{m}}^T$, then a result similar to Lemma 4 holds, with modifications to the definitions of $\boldsymbol{W}$ and $\tilde{\boldsymbol{y}}$.

Equation 9 is a weighted fused lasso problem with response vector $\tilde{\boldsymbol{y}}$ and weights $\sqrt{n_1}, \sqrt{n_2}, \ldots, \sqrt{n_q}$. When $q = n$, (9) simplifies to a standard one-dimensional fused lasso problem.

Corollary 5 *If $q = n$ and $\boldsymbol{M}^* = \tilde{\boldsymbol{m}}\mathbf{1}_n^T$, then $\tilde{\boldsymbol{m}}$ is the solution to the one-dimensional fused lasso problem*

$$\underset{\tilde{\boldsymbol{m}}\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\left\|\boldsymbol{P}\boldsymbol{y} - \tilde{\boldsymbol{m}}\right\|_2^2 + \lambda\sqrt{n}\left\|\boldsymbol{D}\tilde{\boldsymbol{m}}\right\|_1, \tag{10}$$

where $\boldsymbol{P}$ is the permutation matrix that orders the elements of $\boldsymbol{x}_1$ from least to greatest.

If instead $\boldsymbol{M}^* = \mathbf{1}_n\tilde{\boldsymbol{m}}^T$, then Corollary 5 holds with $\boldsymbol{P}$ defined to be the permutation matrix that orders the elements of $\boldsymbol{x}_2$ from least to greatest.

5.2 Connection to Fused Lasso Additive Model

In this subsection, we will establish that CRISP is a generalization of the fused lasso additive model (FLAM) proposal of Petersen et al. (forthcoming). FLAM fits an additive model in which each covariate's fit is estimated to be piecewise constant with adaptively-chosen knots.

For simplicity, assume that $q = n$. Consider a modification of CRISP in which we impose additivity on the mean matrix $\boldsymbol{M}$. That is, we assume $f(x_1, x_2) = \theta_0 + f_1(x_1) + f_2(x_2)$, where θ_0 is an overall mean, and f_1 and f_2 are mean-zero over the training observations. We introduce the n-vectors $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, where $f_1(x_{i1}) = \theta_{1i}$ and $f_2(x_{i2}) = \theta_{2i}$ for all $i = 1, \ldots, n$. Thus the additivity constraint for the (i, j) element of $\boldsymbol{M}$, $M_{(i)(j)}$, can be expressed as

$$M_{(i)(j)} = \theta_0 + \theta_{1i} + \theta_{2j} \text{ for } i = 1, \ldots, n;\ j = 1, \ldots, n \quad \text{with} \quad \mathbf{1}^T\boldsymbol{\theta}_1 = \mathbf{1}^T\boldsymbol{\theta}_2 = 0. \tag{11}$$

Lemma 6 *CRISP (1)–(2) with $q = n$ and with the additional additivity constraint (11) is equivalent to FLAM with $p = 2$, which is the solution to the optimization problem*

$$\begin{aligned} &\underset{\theta_0\in\mathbb{R},\boldsymbol{\theta}_1,\boldsymbol{\theta}_2\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}\left\|\boldsymbol{y} - (\theta_0\mathbf{1} + \boldsymbol{\theta}_1 + \boldsymbol{\theta}_2)\right\|_2^2 + \lambda\left(\left\|\boldsymbol{D}\boldsymbol{P}_1\boldsymbol{\theta}_1\right\|_1 + \left\|\boldsymbol{D}\boldsymbol{P}_2\boldsymbol{\theta}_2\right\|_1\right) \\ &\textit{subject to } \mathbf{1}^T\boldsymbol{\theta}_1 = \mathbf{1}^T\boldsymbol{\theta}_2 = \mathbf{0}, \end{aligned} \tag{12}$$

where $\lambda \geq 0$ is a tuning parameter, $\boldsymbol{P}_j$ is the permutation matrix that orders the elements of $\boldsymbol{x}_j$ from least to greatest, and $\boldsymbol{D} = [\boldsymbol{I}_{(n-1)\times(n-1)}\ \mathbf{0}_{(n-1)\times 1}] - [\mathbf{0}_{(n-1)\times 1}\ \boldsymbol{I}_{(n-1)\times(n-1)}]$ is the first difference matrix.

The proof of Lemma 6 follows from algebraic manipulation.

CRISP (1)–(2) with the additivity constraint (11) is also equivalent to FLAM when the ℓ_2 norms in the penalty (2) are changed to ℓ_1 or ℓ_∞ norms. These alternative penalties are discussed further in Appendix B.

Lemma 6 can be generalized in order to establish that CRISP with $q < n$ is equivalent to a version of FLAM that re-weights the loss function in (12) appropriately.

6. Data Application

We consider predicting median house value on the basis of median income and average occupancy, measured for 20,640 neighborhoods in California. The data set was originally considered in Pace and Barry (1997) and is publicly available from the Carnegie Mellon StatLib data repository (`lib.stat.cmu.edu`).

For this analysis, we focus on predicting median house value for the central area of the covariate space. In particular, we filter the neighborhoods to select those with median incomes and average occupancies that both fall within the central 95% of the covariate distribution, which results in 18,662 neighborhoods to be analyzed. Further details are provided in Appendix I. To illustrate the impact that the size of the data set may have on the preferred analysis approach, we consider five different training set sizes: 100, 500, 1000, 5000, and 11,198 (which corresponds to 60% of the observations). We use the observations not selected for the training set as the test set. For each training set size, we consider 10 different data samples. We compare the performance of CRISP (with $q = 100$) to CART and TPS.

Figure 7 shows that income is positively associated with house value. Occupancy is not strongly associated with house value in low-income neighborhoods. However, among neighborhoods with median incomes exceeding around \$50,000, neighborhoods with mostly single or double occupancy tend to have more expensive homes than those with higher occupancies and the same income. This is perhaps because single people and couples without children have more disposable income to spend on housing than families at the same income level.

In Figure 7, we show estimated mean models from CRISP for two different values of λ. The larger value of λ has slightly worse prediction performance, but has a simple block structure reminiscent of CART. The smaller value of λ gives better prediction performance with a more complex fit structure that resembles the fits from TPS. This illustrates how CRISP's tuning parameter, λ, balances the trade-off between interpretability and prediction performance.

While the fit from CART in Figure 7 is quite interpretable, CART gives highly-variable fits across different splits of the data. This is illustrated in Figure 8. The average variance of predictions from CART across the 10 splits of data is more than three times that of CRISP and TPS. For larger training sets, the variance decreases, though the variability of the CART predictions remains much larger than that of CRISP and TPS. In Figure 8, we also see that CART's performance in terms of test set mean squared error (MSE) is worse than CRISP and TPS, but becomes increasingly similar with larger sample sizes. For example, in Figure 9, we show the results for the largest training set sample considered ($n = 11,198$). We see that all three methods perform very similarly in terms of test set MSE, and provide qualitatively similar estimated mean models. As the available sample size increases, the differences between CRISP, TPS, and CART in terms of prediction performance and interpretability of fits become less pronounced.

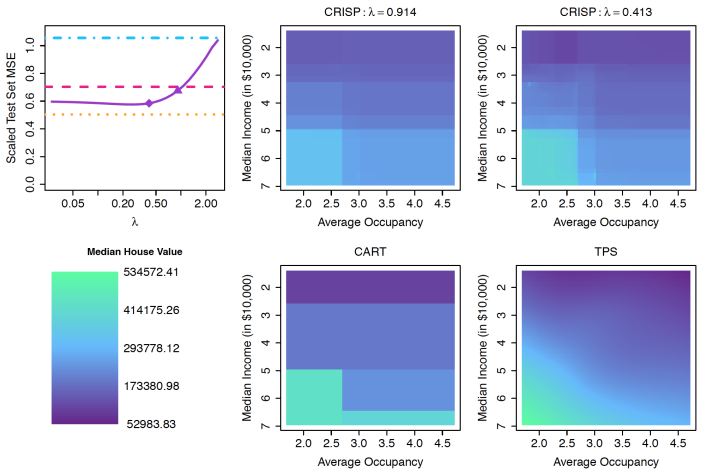


Figure 7: We consider predicting median house value on the basis of median income and average occupancy using a training set of size $n = 100$, as considered in Section 6. We plot the average value for 10 data samples of test set MSE divided by the variance of the training set outcome. We plot this scaled test set MSE versus λ for CRISP (▬), and show the minimum scaled test set MSE achieved by CART (▬ ▬), TPS (▬ ▬), and an intercept-only model (▬ ▬). Estimated mean models for CRISP are shown for a larger value of λ (indicated by ▲) and a smaller value (indicated by ◆). The estimated mean models shown for CART and TPS correspond to the tuning parameter with the minimum test set MSE. The heat scale legend for the median house value is shown.

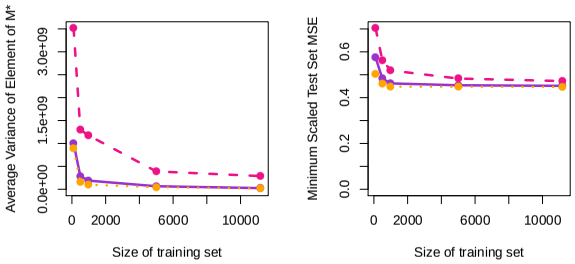


Figure 8: We plot the average variance of predictions and the minimum scaled test set MSE (as defined in Figure 7) as a function of training set sample size for CRISP (▬), CART (– – –), and TPS (- - -) applied to the housing data considered in Section 6.

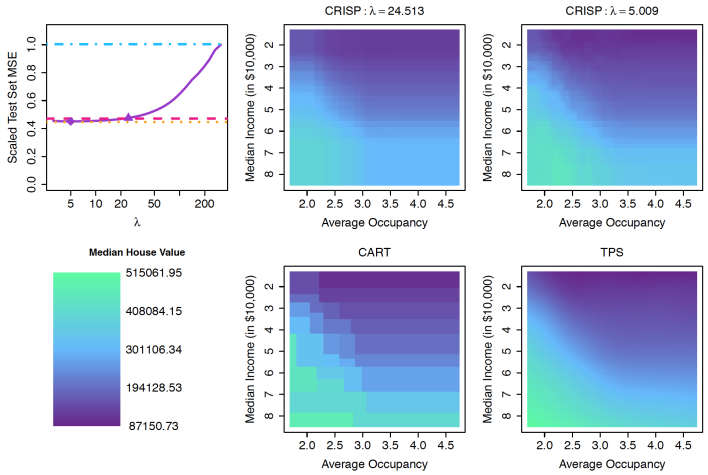


Figure 9: Results using median income and average occupancy as predictors of median house value using a training set of size $n = 11,198$, as considered in Section 6. Details are as in Figure 7.

7. Extension to $p > 2$

We have assumed thus far that $p = 2$. In this case, the estimated mean model for the entire covariate space can be summarized in a single plot, as in Figure 2.

We extend CRISP to the setting of $p > 2$ by constructing an *additive model of bivariate fits*. That is, we estimate the fit for each of the $\frac{p(p-1)}{2}$ *pairs* of features, giving a bivariate fit for each pair of covariates like those obtained in the setting of $p = 2$ and shown in Figure 2. We assume that the mean model is additive in these fits. We restrict the model to pairwise interactions between covariates for a couple of reasons. First, only considering pairwise interactions increases interpretability and reduces model complexity. Our model fit with pairwise interactions can be summarized using $\frac{p(p-1)}{2}$ plots, like those shown in Figure 2. There is no analogous way to easily summarize the model if we were to include higher-order interactions. Second, considering higher-order interactions would cause our model to suffer from the *curse of dimensionality*. That is, as the number of covariates increases, the data in any region of the p-dimensional space will become sparser and sparser: there would be an insufficient density of data throughout the covariate space to reasonably estimate a mean model with higher-order interactions.

We now present the details of our proposal for CRISP with $p > 2$. We consider interactions between each pair of features, $\{(j, j') : 1 \leq j < j' \leq p\}$. For ease of notation, we refer to the elements of this set using the index $k \in (1, \ldots, K)$ where $K = \frac{p(p-1)}{2}$. Recall that for $p = 2$, the mean model for CRISP is $\mathrm{E}[\boldsymbol{y} \mid \boldsymbol{x}_1, \boldsymbol{x}_2] = \boldsymbol{Q}\boldsymbol{m}$, where $\boldsymbol{m} \in \mathbb{R}^{q^2}$ is the vectorized mean matrix and $\boldsymbol{Q}$ selects the elements of $\boldsymbol{m}$ corresponding to the covariate bins of the elements of $\boldsymbol{y}$. Recall that $\boldsymbol{Q}$ is a function of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, though we suppress this to simplify the notation. For $p > 2$, we consider the mean model

$$\mathrm{E}[\boldsymbol{y} \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_p] = m_0\mathbf{1} + \sum_{k=1}^{K} \boldsymbol{Q}_k \boldsymbol{m}_k,$$

where $m_0 \in \mathbb{R}$ is an intercept, $\boldsymbol{m}_k \in \mathbb{R}^{q^2}$ is the vectorized mean matrix for the pair of features indexed by k, and $\boldsymbol{Q}_k \in \mathbb{R}^{n \times q^2}$ selects the elements of $\boldsymbol{m}_k$ corresponding to the covariate bins for the pair of covariates indexed by k. We include the intercept $m_0 \in \mathbb{R}$ in our model, and assume that $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K$ are mean-zero, to ensure identifiability.

When $p > 2$, we extend the CRISP optimization problem (4) as follows:

$$\underset{m_0, \boldsymbol{m}_k, \boldsymbol{z}_k : k=1,\ldots,K}{\text{minimize}} \quad \frac{1}{2} \left\| \boldsymbol{y} - \left(m_0\mathbf{1} + \sum_{k=1}^{K} \boldsymbol{Q}_k \boldsymbol{m}_k \right) \right\|_2^2 + \lambda \sum_{k=1}^{K} \sum_{i=1}^{q-1} \left[\|\boldsymbol{z}_{k,1i}\|_2 + \|\boldsymbol{z}_{k,2i}\|_2 \right] \quad (13)$$
$$\text{subject to } \boldsymbol{A}\boldsymbol{m}_k = \boldsymbol{z}_k, \mathbf{1}^T \boldsymbol{m}_k = 0,$$

where $\boldsymbol{A}$ is as defined in Section 2.2. Thus $\hat{\boldsymbol{y}} = m_0^*\mathbf{1} + \sum_{k=1}^{K} \boldsymbol{Q}_k \boldsymbol{m}_k^*$, where $(m_0^*, \boldsymbol{m}_1^*, \ldots, \boldsymbol{m}_K^*)$ is the solution to (13).

Problem (13) can be solved using block coordinate descent (Tseng, 2001), which gives Algorithm 2. We iterate through the pairs of covariates, and perform a partial minimization (using Algorithm 1) for each $\boldsymbol{m}_k$, while keeping the others fixed. Using an argument similar to that in Section 2.3, the computational complexity of Algorithm 2 is $\mathcal{O}(K(n + q^4))$ for

an initial step and $\mathcal{O}(q^3)$ for each iteration of Step 2(b) of Algorithm 2. In practice, the number of iterations needed to achieve convergence in Step 2(b) of Algorithm 2 is relatively small.

We present a block coordinate descent algorithm, since it is a natural extension of Algorithm 1 to the $p > 2$ setting. However, CRISP with $p \gg 2$ can alternatively be fit using generalized gradient descent, which allows the updates for each bivariate fit to be run in parallel on a cluster.

Algorithm 2 — Block Coordinate Descent for CRISP with $p > 2$ (Equation (13))

1. Initialize $m_0^* = 0$ and $\boldsymbol{m}_k^* = \boldsymbol{0}$ for all $k = 1, \ldots, K$.
2. For $k = 1, \ldots, K, 1, \ldots, K, \ldots$, until convergence of the objective of (13):
   (a) Compute the residual $\boldsymbol{r}_k = \boldsymbol{y} - \left(m_0^* \boldsymbol{1} + \sum_{k' \neq k} \boldsymbol{Q}_{k'} \boldsymbol{m}_{k'}^*\right)$.
   (b) Using Algorithm 1, solve
   $$\underset{\boldsymbol{m}_k, \boldsymbol{z}_k}{\text{minimize}} \quad \frac{1}{2} \|\boldsymbol{r}_k - \boldsymbol{Q}_k \boldsymbol{m}_k\|_2^2 + \lambda \sum_{i=1}^{q-1} \left[\|\boldsymbol{z}_{k,1i}\|_2 + \|\boldsymbol{z}_{k,2i}\|_2\right] \quad \text{subject to } \boldsymbol{A}\boldsymbol{m}_k = \boldsymbol{z}_k.$$
   Let $\boldsymbol{m}_k^*$ denote the solution.
   (c) Compute the intercept, $m_0^* \leftarrow m_0^* + \text{mean}(\boldsymbol{m}_k^*)$, and center, $\boldsymbol{m}_k^* \leftarrow \boldsymbol{m}_k^* - \text{mean}(\boldsymbol{m}_k^*)$.

8. Discussion

We have presented CRISP, a method for fitting interpretable, flexible, and non-additive predictive models. CRISP fits have an easily-interpreted block structure, which is somewhat reminiscent of the fits from CART. But the fits from CRISP result from a non-greedy procedure, and are much less variable than those of CART. In our numerical studies, the prediction performance of CRISP is similar to TPS, and in many cases CRISP provides a simpler and more interpretable fit.

Future work could consider an alternative penalization scheme. Recall that CRISP first divides the covariate space into a $q \times q$ grid of bins. Our proposal only uses the information about the bin into which each of the n observations falls, which is used to construct $\boldsymbol{Q}$ in (4). Thus CRISP only makes use of the rankings of the observations for each covariate, rather than the actual values of the covariates. A modification to (4) could allow us to more heavily penalize the differences between pairs of neighboring rows or columns corresponding to observations with similar values in a given covariate. This modification is not very important when the covariate pairs are distributed uniformly over the covariate space, as in our simulation study in Section 3.

In this paper, we have only considered the setting of $p \ll n$. An extension of CRISP to larger p is left to future work.

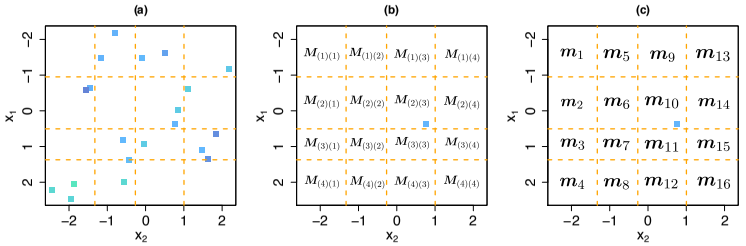


Figure 10: In *(a)*, each of the 20 squares represents an observation (x_1, x_2, y). There are $q^2 = 16$ bins of (x_1, x_2) values, whose boundaries coincide with the quartiles (- - -) of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. In *(b)* and *(c)*, we label the elements of $\boldsymbol{M}$ and $\boldsymbol{m}$, respectively, corresponding to each bin of (x_1, x_2) values. Additionally, in *(b)* and *(c)*, we show $(x_{1i}, x_{2i}) = (0.4, 0.8)$, which is used in Appendix A to describe the construction of $\boldsymbol{Q}$.

Acknowledgments

We thank the associate editor and three referees for helpful comments. D.W. was supported by NIH Grant DP5OD009145, NSF CAREER Award DMS-1252624, and an Alfred P. Sloan Foundation Research Fellowship. N.S. was supported by NIH Grant DP5OD019820.

Appendix A. Notational Details

We first give an intuitive explanation of our vectorization scheme. Recall that each row of $\boldsymbol{Q} \in \mathbb{R}^{n \times q^2}$ contains $q^2 - 1$ elements that equal 0, and a single 1 that extracts an element of $\boldsymbol{m}$ according to the covariate values for that observation. For example, consider the ith row of $\boldsymbol{Q}$ for $(x_{1i}, x_{2i}) = (0.4, 0.8)$ in Figure 10(a). These covariate values fall within the 2nd row and 3rd column of the 4×4 grid, meaning that $\boldsymbol{M}_{(2)(3)}$ provides an estimate for y_i. After vectorizing $\boldsymbol{M}$, $\boldsymbol{M}_{(2)(3)}$ is $\boldsymbol{m}_{10}$, the 10th element of the mean vector. Note that we can convert between the matrix and vector notation by taking the column number minus one multiplied by q and adding the row number (e.g., $(3-1) \times 4 + 2$). The correspondence between $\boldsymbol{M}$ and $\boldsymbol{m}$ is illustrated in Figures 10(b) and 10(c). Thus the ith row of $\boldsymbol{Q}$ would contains all zeros, except a single 1 for the 10th element. Finally, $(\boldsymbol{Q}\boldsymbol{m})_i = m_{10}$.

Before formally defining the function Ω and matrices $\boldsymbol{Q}$, $\boldsymbol{R}_i$ for $i = 1, \ldots, q-1$, and $\boldsymbol{C}_i$ for $i = 1, \ldots, q-1$ introduced in Section 2.2, we define a quantile function. We use quantile$(\cdot)$ to denote the quantile range into which an element falls: quantile$(x_{1i}) = k$ if x_{1i} is between the $\frac{k-1}{q}$- and $\frac{k}{q}$-quantiles of $\boldsymbol{x}_1$. For example, if $n = q = 4$ and $\boldsymbol{x}_1 = (9\ 3\ 5\ 2)^T$, then quantile$(x_{11}) = 4$. Similarly, if $n = 6$, $q = 3$, and $\boldsymbol{x}_1 = (7\ 2\ 3\ 8\ 1\ 5)^T$, then quantile$(x_{16}) = 2$.

We define the function Ω as $\Omega(\boldsymbol{M}, x_{1i}, x_{2i}) = M_{(a)(b)}$ where $a = \text{quantile}(x_{1i})$ and $b = \text{quantile}(x_{2i})$.

We construct $\boldsymbol{Q} \in \mathbb{R}^{n \times q^2}$ such that

$$[\boldsymbol{Q}]_{jk} = \begin{cases} 1 & \text{if } k = \text{quantile}(x_{1j}) + q \times (\text{quantile}(x_{2j}) - 1) \\ 0 & \text{otherwise} \end{cases},$$

$\boldsymbol{R}_i \in \mathbb{R}^{q \times q^2}$ for $i = 1, \ldots, q-1$ such that

$$[\boldsymbol{R}_i]_{jk} = \begin{cases} 1 & \text{if } k = i + q \times (j-1) \\ -1 & \text{if } k = i + 1 + q \times (j-1) \\ 0 & \text{otherwise} \end{cases},$$

and $\boldsymbol{C}_i \in \mathbb{R}^{q \times q^2}$ for $i = 1, \ldots, q-1$ such that

$$[\boldsymbol{C}_i]_{jk} = \begin{cases} 1 & \text{if } k = j + q \times (i-1) \\ -1 & \text{if } k = j + q \times i \\ 0 & \text{otherwise} \end{cases}.$$

Appendix B. Alternative Penalties

A more general formulation of our proposal in (1) is

$$\underset{\boldsymbol{M} \in \mathbb{R}^{q \times q}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{n} \left(y_i - \Omega(\boldsymbol{M}, x_{1i}, x_{2i})\right)^2 + \lambda \sum_{i=1}^{q-1} \left[\left\| \boldsymbol{M}_{i\cdot} - \boldsymbol{M}_{(i+1)\cdot} \right\|_t + \left\| \boldsymbol{M}_{\cdot i} - \boldsymbol{M}_{\cdot(i+1)} \right\|_t \right], \quad (14)$$

which is equivalent to (1) for $t = 2$. One might consider solving (14) for $t = \infty$, which (like $t = 2$) encourages pairs of neighboring rows or columns of $\boldsymbol{M}$ to be identical. We compare the fit for $t = 2$ to that for $t = \infty$ in Figure 11(a)–(b). While $t = \infty$ gives desirable fits similar to $t = 2$, the computational time required is much higher than that for $t = 2$. This is because when adapted to $t = \infty$, Step 2(b) of Algorithm 1 no longer has a closed-form solution (Duchi and Singer, 2009).

We also consider the use of $t = 1$ in (14); this encourages each element of $\boldsymbol{M}$ to equal its four adjacent elements. However, using $t = 1$ gives very poor results: the bins of $\boldsymbol{M}$ containing observations are estimated to be shrunken versions of their observed values, while the bins of $\boldsymbol{M}$ without observations are estimated to be a common value (Figure 11(c)). In a sense, the penalization for $t = 1$ is too local given the data sparsity (e.g., only q of q^2 elements observed when $q = n$).

The results for $t = 1$ improve if an additional penalty is added to the objective function. First, note that (14) can also be written as

$$\underset{\boldsymbol{M} \in \mathbb{R}^{q \times q}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{n} \left(y_i - \Omega(\boldsymbol{M}, x_{1i}, x_{2i})\right)^2 + \lambda \left(\|\boldsymbol{M}^T \boldsymbol{D}^T\|_{t,1} + \|\boldsymbol{M} \boldsymbol{D}^T\|_{t,1} \right), \quad (15)$$

where $\boldsymbol{D} = [\boldsymbol{I}_{(q-1)\times(q-1)} \; \boldsymbol{0}_{(q-1)\times 1}] - [\boldsymbol{0}_{(q-1)\times 1} \; \boldsymbol{I}_{(q-1)\times(q-1)}]$. Motivated by a proposal from van de Geer (2000), we add an additional penalty to (15) with $t = 1$,

$$\underset{\boldsymbol{M} \in \mathbb{R}^{q \times q}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{n} \left(y_i - \Omega(\boldsymbol{M}, x_{1i}, x_{2i})\right)^2 + \lambda \left(\|\boldsymbol{M}^T \boldsymbol{D}^T\|_{1,1} + \|\boldsymbol{M} \boldsymbol{D}^T\|_{1,1} + \|\boldsymbol{D}\boldsymbol{M}\boldsymbol{D}^T\|_{1,1} \right). \quad (16)$$

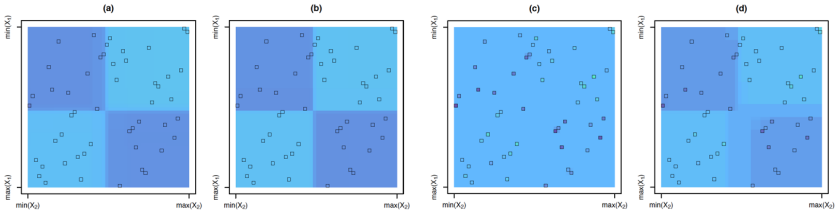


Figure 11: The estimated mean model from solving (14) for *(a)* $t = 2$ (CRISP), *(b)* $t = \infty$, and *(c)* $t = 1$, as well as *(d)* the estimated mean model from solving (16). The methods are described in detail in Appendix B. Note that $q = n$ was used for all methods. Data was generated for $n = 50$ from Scenario 2 (described in Section 3). The locations of the 50 observations are outlined in each plot. The heat scale legend is in Figure 1(e).

The penalty $\|\boldsymbol{D}\boldsymbol{M}\boldsymbol{D}^T\|_{1,1}$ encourages $|M_{(i)(j)} + M_{(i-1)(j-1)} - M_{(i-1)j} - M_{i(j-1)}|$ to equal zero, which results in a block structure as shown in Figure 11(d). While (16) outperforms (14) with $t = 1$, CRISP with $t = 2$ yields better results.

Appendix C. Details of Algorithm 1

C.1 Derivation of Algorithm 1

The scaled augmented Lagrangian of (4) is

$$L_\rho(\boldsymbol{m}, \boldsymbol{z}, \boldsymbol{u}) = \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{m}\|_2^2 + \lambda \sum_{i=1}^{q-1} \left[\|\boldsymbol{z}_{1i}\|_2 + \|\boldsymbol{z}_{2i}\|_2 \right] + \frac{\rho}{2} \sum_{i=1}^{q-1} \left[\|\boldsymbol{R}_i \boldsymbol{m} - \boldsymbol{z}_{1i} + \boldsymbol{u}_{1i}\|_2^2 + \|\boldsymbol{C}_i \boldsymbol{m} - \boldsymbol{z}_{2i} + \boldsymbol{u}_{2i}\|_2^2 \right] \quad (17)$$

where $\boldsymbol{u} = \left((\boldsymbol{u}_{11})^T \ldots (\boldsymbol{u}_{1(q-1)})^T (\boldsymbol{u}_{21})^T \ldots (\boldsymbol{u}_{2(q-1)})^T \right)^T$ is the scaled dual variable. Solving (4) using ADMM relies on initializing estimates $\boldsymbol{m}^{(0)} := \boldsymbol{0}, \boldsymbol{z}^{(0)} := \boldsymbol{0}$, and $\boldsymbol{u}^{(0)} := \boldsymbol{0}$ and then iterating over three steps until convergence. At iteration k, the updates are

Step 1. $\boldsymbol{m}^{(k)} := \underset{\boldsymbol{m}}{\text{argmin}} \quad L_\rho \left(\boldsymbol{m}^{(k-1)}, \boldsymbol{z}^{(k-1)}, \boldsymbol{u}^{(k-1)} \right)$

Step 2. $\boldsymbol{z}^{(k)} := \underset{\boldsymbol{z}}{\text{argmin}} \quad L_\rho \left(\boldsymbol{m}^{(k)}, \boldsymbol{z}^{(k-1)}, \boldsymbol{u}^{(k-1)} \right)$

Step 3. $\boldsymbol{u}_{1i}^{(k)} := \boldsymbol{u}_{1i}^{(k-1)} + \boldsymbol{R}_i \boldsymbol{m}^{(k)} - \boldsymbol{z}_{1i}^{(k)}$ for $i = 1, \ldots, q-1$
$\boldsymbol{u}_{2i}^{(k)} := \boldsymbol{u}_{2i}^{(k-1)} + \boldsymbol{C}_i \boldsymbol{m}^{(k)} - \boldsymbol{z}_{2i}^{(k)}$ for $i = 1, \ldots, q-1$

Note that Step 3 can equivalently be written as $\boldsymbol{u}^{(k)} := \boldsymbol{u}^{(k-1)} + \boldsymbol{A}\boldsymbol{m}^{(k)} - \boldsymbol{z}^{(k)}$. We provide details regarding Steps 1 and 2 below.

Details of Step 1

The optimality condition of (17) for $\boldsymbol{m}$ is

$$\frac{\partial L_\rho}{\partial \boldsymbol{m}} = -\boldsymbol{Q}^T(\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{m}) + \rho \sum_{i=1}^{q-1} \left[\boldsymbol{R}_i^T(\boldsymbol{R}_i\boldsymbol{m} - \boldsymbol{z}_{1i} + \boldsymbol{u}_{1i}) + \boldsymbol{C}_i^T(\boldsymbol{C}_i\boldsymbol{m} - \boldsymbol{z}_{2i} + \boldsymbol{u}_{2i})\right] = \boldsymbol{0}$$

or equivalently, $-\boldsymbol{Q}^T(\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{m}) + \rho\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{m} + \boldsymbol{u} - \boldsymbol{z}) = \boldsymbol{0}$. Therefore the update for Step 1 is $\boldsymbol{m}^{(k)} := \left[\boldsymbol{Q}^T\boldsymbol{Q} + \rho\boldsymbol{A}^T\boldsymbol{A}\right]^{-1}\left[\boldsymbol{Q}^T\boldsymbol{y} + \rho\boldsymbol{A}^T(\boldsymbol{z}^{(k-1)} - \boldsymbol{u}^{(k-1)})\right]$.

Details of Step 2

The *proximal operator* $\mathbf{prox}_{\lambda f}$ of λf is defined by $\mathbf{prox}_{\lambda f}(\boldsymbol{v}) = \underset{\boldsymbol{x}}{\operatorname{argmin}}\left(f(\boldsymbol{x}) + \frac{1}{2\lambda}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2\right)$. The minimization for Step 2 is separable in the $\boldsymbol{z}_{1i}$ and $\boldsymbol{z}_{2i}$ for $i = 1, \ldots, q-1$. The minimization for $\boldsymbol{z}_{1i}$ is

$$\begin{aligned}
\boldsymbol{z}_{1i}^{(k)} &:= \underset{\boldsymbol{z}_{1i}}{\operatorname{argmin}} \left[\lambda\|\boldsymbol{z}_{1i}\|_2 + \frac{\rho}{2}\left\|\boldsymbol{R}_i\boldsymbol{m}^{(k)} - \boldsymbol{z}_{1i} + \boldsymbol{u}_{1i}^{(k-1)}\right\|_2^2\right] \\
&= \mathbf{prox}_{\frac{\lambda}{\rho}\|\cdot\|_2}\left(\boldsymbol{R}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{1i}^{(k-1)}\right) \\
&= \left(\boldsymbol{R}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{1i}^{(k-1)}\right)\left(1 - \frac{\lambda}{\rho\left\|\boldsymbol{R}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{1i}^{(k-1)}\right\|_2}\right)_+ .
\end{aligned}$$

Similarly, the update for $\boldsymbol{z}_{2i}$ is $\boldsymbol{z}_{2i}^{(k)} := \left(\boldsymbol{C}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{2i}^{(k-1)}\right)\left(1 - \frac{\lambda}{\rho\left\|\boldsymbol{C}_i\boldsymbol{m}^{(k)} + \boldsymbol{u}_{2i}^{(k-1)}\right\|_2}\right)_+$.

C.2 Stopping Criterion

We use the stopping criterion for Algorithm 1 suggested in Boyd et al. (2011), stopping when the primal residual $\boldsymbol{r}^{(k)} = \boldsymbol{A}\boldsymbol{m}^{(k)} - \boldsymbol{z}^{(k)}$ and dual residual $\boldsymbol{s}^{(k)} = \rho\boldsymbol{A}^T\left(\boldsymbol{z}^{(k-1)} - \boldsymbol{z}^{(k)}\right)$ are sufficiently small. Specifically, we check if

$$\|\boldsymbol{r}^{(k)}\|_2 \leq \sqrt{2q(q-1)}\epsilon^{abs} + \epsilon^{rel}\max\{\|\boldsymbol{A}\boldsymbol{m}^{(k)}\|_2, \|\boldsymbol{z}^{(k)}\|_2\} \quad \text{and} \quad \|\boldsymbol{s}^{(k)}\|_2 \leq q\epsilon^{abs} + \epsilon^{rel}\|\rho\boldsymbol{A}^T\boldsymbol{u}^{(k)}\|_2$$

with $\epsilon^{abs}, \epsilon^{rel} > 0$. We use $\epsilon^{abs} = 10^{-4}$ and $\epsilon^{rel} = 10^{-2}$ in order to obtain the results presented in Sections 3 and 6.

C.3 Varying Penalty Parameter

We can vary ρ from iteration to iteration in order to achieve better convergence and reduce the dependence of performance on the initially chosen ρ. We adopt the scheme for varying ρ that is reviewed in Boyd et al. (2011). Since we use the scaled dual variable, $\boldsymbol{u}$ must also

be updated in conjunction with the updating of ρ. At the end of each iteration, we apply the updates

$$(\rho^{(k+1)}, \boldsymbol{u}^{(k+1)}) := \begin{cases} (\tau^{incr}\rho^{(k)}, \boldsymbol{u}^{(k)}/\tau^{incr}) & \text{if } \|\boldsymbol{r}^{(k)}\|_2 > \delta\|\boldsymbol{s}^{(k)}\|_2 \\ (\rho^{(k)}/\tau^{decr}, \tau^{decr}\boldsymbol{u}^{(k)}) & \text{if } \|\boldsymbol{s}^{(k)}\|_2 > \delta\|\boldsymbol{r}^{(k)}\|_2 \\ (\rho^{(k)}, \boldsymbol{u}^{(k)}) & \text{otherwise} \end{cases}$$

where $\delta, \tau^{incr}, \tau^{decr} > 1$. We choose $\delta = 10$ and $\tau^{incr} = \tau^{decr} = 2$. Updating ρ keeps the norms of the residuals $\boldsymbol{r}^{(k)}$ and $\boldsymbol{s}^{(k)}$ within a factor of δ of one another. While convergence of ADMM has only been proven for fixed ρ, varying ρ has been shown to work well in practice (Boyd et al., 2011).

C.4 Modification to Provide Sparsity

Inspection of the updates for $\boldsymbol{z}_{1i}^*$ and $\boldsymbol{z}_{2i}^*$ in Algorithm 1 indicates that the ADMM algorithm yields sparsity in $\boldsymbol{z}_{1i}^*$ and $\boldsymbol{z}_{2i}^*$, but not necessarily exact equality of the rows and columns of $\boldsymbol{M}^*$. This is in effect a numerical issue: our algorithm might yield $\boldsymbol{z}_{1i} = \boldsymbol{0}$, but $\|\boldsymbol{M}_{i\cdot}^* - \boldsymbol{M}_{(i+1)\cdot}^*\|_2 = 1 \times 10^{-8}$. To resolve this issue, we first determine the "blocks" of $\boldsymbol{m}^*$ using an initial run of Algorithm 1, and then solve (4) once more with constraints on the rows and columns of $\boldsymbol{M}$ to enforce equality of the appropriate rows and columns. This second optimization is performed simply to yield an estimate of $\boldsymbol{M}$ for which elements are exactly equal within each block.

Appendix D. Details of Simulations in Section 3

The mean models $f(x_1, x_2)$ used to generate data for Scenarios 1–4 in Section 3 are defined as follows. Note that x_1 and x_2 are sampled uniformly from $[-2.5, 2.5]$. We define the indicator function $\mathbf{1}_{\mathcal{A}}(x) = \begin{cases} 1 & \text{if } x \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}$.

Scenario 1: $f(x_1, x_2) = \text{sign}(x_1) \times \mathbf{1}_{[0,\infty)}(x_1 \times x_2)$
Scenario 2: $f(x_1, x_2) = -\text{sign}(x_1 \times x_2)$
Scenario 3: $f(x_1, x_2) = -3 \times \mathbf{1}_{[-2.5,-0.83)}(x_1) \times \mathbf{1}_{[-2.5,-1.25)}(x_2) + \mathbf{1}_{[-2.5,-0.83)}(x_1) \times \mathbf{1}_{[-1.25,2.5]}(x_2) - 2\times\mathbf{1}_{[-0.83,0.83]}(x_1)\times\mathbf{1}_{[-2.5,0)}(x_2) + 2\times\mathbf{1}_{[-0.83,0.83]}(x_1)\times\mathbf{1}_{[0,2.5]}(x_2) - \mathbf{1}_{(0.83,2.5]}(x_1)\times\mathbf{1}_{[-2.5,1.25)}(x_2) + 3 \times \mathbf{1}_{(0.83,2.5]}(x_1) \times \mathbf{1}_{[1.25,2.5]}(x_2)$
Scenario 4: $f(x_1, x_2) = \frac{10}{\left(\frac{x_1-2.5}{3}\right)^2 + \left(\frac{x_2-2.5}{3}\right)^2 + 1} + \frac{10}{\left(\frac{x_1+2.5}{3}\right)^2 + \left(\frac{x_2+2.5}{3}\right)^2 + 1}$

Each of the mean models $f(x_1, x_2)$ defined above is centered and scaled such that $\int_{-2.5}^{2.5}\int_{-2.5}^{2.5} f(x_1, x_2)\, dx_1\, dx_2 = 0$ and $\frac{1}{25}\int_{-2.5}^{2.5}\int_{-2.5}^{2.5} f(x_1, x_2)^2\, dx_1\, dx_2 = 2$.

Appendix E. Proof Sketch of Proposition 1

Proof Using the dual problem of (5) and Lemma 1 of Tibshirani and Taylor (2012), it can be shown that $g : \mathbb{R}^n \to \mathbb{R}^n$ with $\hat{\boldsymbol{y}} = g(\boldsymbol{y}) = (g_1(\boldsymbol{y}), \ldots, g_n(\boldsymbol{y}))^T$ is continuous and almost differentiable. Thus, Stein's lemma implies that $\text{df}(\hat{\boldsymbol{y}}) = \text{E}\left[\text{Tr}\left(\frac{\partial g(\boldsymbol{y})}{\partial \boldsymbol{y}}\right)\right]$. At the optimum

of (5), we have

$$\boldsymbol{Q}^T(\boldsymbol{y}-\boldsymbol{Q}\boldsymbol{m}^*)=\lambda\sum_{i=1}^{q-1}[\boldsymbol{R}_i^T S_1(\boldsymbol{R}_i,\boldsymbol{m}^*)+\boldsymbol{C}_i^T S_1(\boldsymbol{C}_i,\boldsymbol{m}^*)]+\gamma\boldsymbol{m}^*, \tag{18}$$

where $S_1(\boldsymbol{A}_i,\boldsymbol{m}^*)=\begin{cases}\frac{\boldsymbol{A}_i\boldsymbol{m}^*}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2} & \text{if } \|\boldsymbol{A}_i\boldsymbol{m}^*\|_2\neq 0\\ \in\{\boldsymbol{g}:\|\boldsymbol{g}\|_2\leq 1\} & \text{if } \|\boldsymbol{A}_i\boldsymbol{m}^*\|_2=0\end{cases}$.

We define $\mathcal{C}=\{\boldsymbol{A}_i:\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2=0\}$ where $\boldsymbol{A}_1=\boldsymbol{R}_1,\boldsymbol{A}_2=\boldsymbol{R}_2,\ldots,\boldsymbol{A}_{q-1}=\boldsymbol{R}_{q-1},\boldsymbol{A}_q=\boldsymbol{C}_1,\boldsymbol{A}_{q+1}=\boldsymbol{C}_2,\ldots,\boldsymbol{A}_{2q-2}=\boldsymbol{C}_{q-1}$. We define $\boldsymbol{A}_*$ to be the submatrix of $\boldsymbol{A}$ with the rows corresponding to $\boldsymbol{A}_i\notin\mathcal{C}$ removed, and let $\boldsymbol{P}=\boldsymbol{I}_{q^2}-\boldsymbol{A}_*^+\boldsymbol{A}_*$, the projection onto the space orthogonal to the row space of $\boldsymbol{A}_*$. We left-multiply (18) by $\boldsymbol{P}$ to give

$$\boldsymbol{P}\boldsymbol{Q}^T(\boldsymbol{y}-\boldsymbol{Q}\boldsymbol{m}^*)=\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}\frac{\boldsymbol{A}_i^T\boldsymbol{A}_i\boldsymbol{m}^*}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}+\gamma\boldsymbol{P}\boldsymbol{m}^*, \tag{19}$$

since $\boldsymbol{P}\boldsymbol{A}_i^T S_1(\boldsymbol{A}_i,\boldsymbol{m}^*)=\boldsymbol{0}$ if $\boldsymbol{A}_i\in\mathcal{C}$ (i.e., $\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2=0$). Because $\boldsymbol{P}\boldsymbol{m}^*=\boldsymbol{m}^*$, (19) can be rewritten as

$$\boldsymbol{P}\boldsymbol{Q}^T(\boldsymbol{y}-\boldsymbol{Q}\boldsymbol{P}\boldsymbol{m}^*)=\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}\frac{\boldsymbol{A}_i^T\boldsymbol{A}_i\boldsymbol{P}\boldsymbol{m}^*}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}+\gamma\boldsymbol{m}^*. \tag{20}$$

We let $\boldsymbol{D}=\operatorname{diag}\left(h(m_1^*),\cdots,h(m_{q^2}^*)\right)$, where $h(m_i^*)$ is defined to be the ratio of the number of observations in the block of $\boldsymbol{M}^*$ that contains m_i^* to the number of elements of $\boldsymbol{M}^*$ in the block of $\boldsymbol{M}^*$ that contains m_i^*. Note that $\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{P}=\boldsymbol{D}\boldsymbol{P}$. Thus $\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{P}\boldsymbol{m}^*=\boldsymbol{D}\boldsymbol{m}^*$, and (20) is equivalent to

$$\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{y}=\boldsymbol{D}\boldsymbol{m}^*+\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}\frac{\boldsymbol{A}_i^T\boldsymbol{A}_i\boldsymbol{P}\boldsymbol{m}^*}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}+\gamma\boldsymbol{m}^*. \tag{21}$$

We conjecture that there is a neighborhood around almost every $\boldsymbol{y}$ such that the blocks of $\boldsymbol{m}^*$ do not change. That is, $\mathcal{C}$ and $\boldsymbol{P}$ in (21) are constant with respect to $\boldsymbol{y}$, and the derivative of (21) with respect to $\boldsymbol{y}$ is

$$\boldsymbol{P}\boldsymbol{Q}^T=\left(\boldsymbol{D}+\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)\boldsymbol{P}+\gamma\boldsymbol{I}\right)\frac{\partial\boldsymbol{m}^*}{\partial\boldsymbol{y}}, \tag{22}$$

where $S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)=\frac{\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}-\frac{\boldsymbol{A}_i^T\boldsymbol{A}_i\boldsymbol{m}^*\boldsymbol{m}^{*T}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2^3}$. Recall $\hat{\boldsymbol{y}}=\boldsymbol{Q}\boldsymbol{m}^*$, so solving (22) for $\frac{\partial\boldsymbol{m}^*}{\partial\boldsymbol{y}}$ and left-multiplying by $\boldsymbol{Q}$ gives

$$\frac{\partial\hat{\boldsymbol{y}}}{\partial\boldsymbol{y}}=\boldsymbol{Q}\left(\boldsymbol{D}+\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)\boldsymbol{P}+\gamma\boldsymbol{I}\right)^{-1}\boldsymbol{P}\boldsymbol{Q}^T,$$

where $\left(\boldsymbol{D}+\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)\boldsymbol{P}+\gamma\boldsymbol{I}\right)$ is invertible as both $\boldsymbol{D}$ and $\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)\boldsymbol{P}$ are positive semi-definite. Therefore, the degrees of freedom is

$$\mathrm{E}\left[\operatorname{Tr}\left(\boldsymbol{Q}\left(\boldsymbol{D}+\lambda\boldsymbol{P}\sum_{i:\boldsymbol{A}_i\notin\mathcal{C}}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)\boldsymbol{P}+\gamma\boldsymbol{I}\right)^{-1}\boldsymbol{P}\boldsymbol{Q}^T\right)\right].$$

This establishes the unbiasedness of the estimator (6). ■

Appendix F. Proof of Corollary 2

Proof This corollary pertains to the setting in which either all rows of $\boldsymbol{M}^*$ are equal (i.e., $\boldsymbol{R}_i\in\mathcal{C}$ for all i) or all columns of $\boldsymbol{M}^*$ are equal (i.e., $\boldsymbol{C}_i\in\mathcal{C}$ for all i). In this setting, we will show $\boldsymbol{P}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)=\boldsymbol{0}$ for any $\boldsymbol{A}_i\notin\mathcal{C}$ using two facts: (1) $\boldsymbol{A}_i\boldsymbol{m}^*=c_i\boldsymbol{1}_q$ for some $c_i\in\mathbb{R}$ and (2) $\boldsymbol{P}\boldsymbol{A}_i^T=\boldsymbol{v}_i\boldsymbol{1}_q^T$ for some $\boldsymbol{v}_i\in\mathbb{R}^{q^2}$. These facts follow from the assumption that either all rows or all columns of $\boldsymbol{M}^*$ are equal. Consider some $\boldsymbol{A}_i\notin\mathcal{C}$. We have

$$\begin{aligned}\boldsymbol{P}S_2(\boldsymbol{A}_i,\boldsymbol{m}^*)&=\frac{\boldsymbol{P}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}-\frac{\boldsymbol{P}\boldsymbol{A}_i^T\boldsymbol{A}_i\boldsymbol{m}^*\boldsymbol{m}^{*T}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2^3}\\&=\frac{\boldsymbol{P}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}-\frac{(\boldsymbol{v}_i\boldsymbol{1}_q^T)(c_i\boldsymbol{1}_q)(c_i\boldsymbol{1}_q^T)\boldsymbol{A}_i}{c_i^2q\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}\\&=\frac{\boldsymbol{P}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}-\frac{\boldsymbol{v}_i\boldsymbol{1}_q^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}\\&=\frac{\boldsymbol{P}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}-\frac{\boldsymbol{P}\boldsymbol{A}_i^T\boldsymbol{A}_i}{\|\boldsymbol{A}_i\boldsymbol{m}^*\|_2}\\&=\boldsymbol{0}.\end{aligned}$$

Therefore, the estimator (6) with $\gamma=0$ simplifies to $\operatorname{Tr}[\boldsymbol{Q}\boldsymbol{D}^{-1}\boldsymbol{P}\boldsymbol{Q}^T]=\operatorname{Tr}[\boldsymbol{D}^{-1}\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{Q}]$. Recall that $\boldsymbol{D}$ is a diagonal matrix with $D_{ii}=h(m_i^*)=N_i^0/N_i$, where N_i^0 and N_i are the number of observations and the number of elements, respectively, in the block of $\boldsymbol{M}^*$ containing m_i^*. Note that $(\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{Q})_{ii}$ equals n_i^0/N_i, where n_i^0 is the number of observations corresponding to m_i^*. Thus

$$\operatorname{Tr}[\boldsymbol{D}^{-1}\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{Q}]=\sum_{i=1}^{q^2}\frac{(\boldsymbol{P}\boldsymbol{Q}^T\boldsymbol{Q})_{ii}}{D_{ii}}=\sum_{i:m_i^*\text{ observed}}\frac{N_i}{N_i^0}\frac{n_i^0}{N_i}=\sum_{i:m_i^*\text{ observed}}\frac{n_i^0}{N_i^0},$$

which equals the total number of blocks of $\boldsymbol{M}^*$ since the n_i^0's for a block sum to N_i^0. ■

Appendix G. Proof of Lemma 3

Proof If $\boldsymbol{m}^* = \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_{q^2}$ solves (3), then there exist q-vectors $\boldsymbol{d}_{1i}, \boldsymbol{d}_{2i}$ with $\|\boldsymbol{d}_{1i}\|_2 \leq \lambda$ and $\|\boldsymbol{d}_{2i}\|_2 \leq \lambda$ such that

$$\boldsymbol{Q}^T\left(\boldsymbol{y} - \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_q\right) = \sum_{i=1}^{q-1}\left[\boldsymbol{R}_i^T\boldsymbol{d}_{1i} + \boldsymbol{C}_i^T\boldsymbol{d}_{2i}\right], \tag{23}$$

since $\boldsymbol{Q}\mathbf{1}_{q^2} = \mathbf{1}_q$. Let $\boldsymbol{d} = (\boldsymbol{d}_{11}^T \ \cdots \ \boldsymbol{d}_{1(q-1)}^T \ \boldsymbol{d}_{21}^T \ \ldots \ \boldsymbol{d}_{2(q-1)}^T)^T$. Then (23) can be rewritten as

$$\boldsymbol{Q}^T\left(\boldsymbol{y} - \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_q\right) = \boldsymbol{A}^T\boldsymbol{d}. \tag{24}$$

Note that $\boldsymbol{m}^* = \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_{q^2}$ for a certain λ if and only if (24) is satisfied for some $\boldsymbol{d}$ for which $\|\boldsymbol{d}_{1i}\|_2 \leq \lambda, \|\boldsymbol{d}_{2i}\|_2 \leq \lambda$ for $i = 1, \ldots, q-1$. We find the $\boldsymbol{d}^*$ corresponding to the minimum λ for which $\boldsymbol{m}^* = \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_{q^2}$ by solving the convex optimization problem

$$\underset{\boldsymbol{d}}{\text{minimize}} \quad \max_{1\leq i\leq q-1}\left\{\|\boldsymbol{d}_{1i}\|_2, \|\boldsymbol{d}_{2i}\|_2\right\} \quad \text{subject to} \quad \boldsymbol{Q}^T\left(\boldsymbol{y} - \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_q\right) = \boldsymbol{A}^T\boldsymbol{d}.$$

Thus $\boldsymbol{m}^* = \left(\frac{1}{n}\mathbf{1}_n^T\boldsymbol{y}\right)\mathbf{1}_{q^2}$ if and only if $\lambda \geq \max_{1\leq i\leq q-1}\left\{\|\boldsymbol{d}_{1i}^*\|_2, \|\boldsymbol{d}_{2i}^*\|_2\right\}$. ■

Appendix H. Simulations Illustrating Performance of (3) with $\lambda = 0$ and Variable q

We illustrate how (3) with $\lambda = 0$ over a range of q values performs compared to CRISP for a variety of scenarios. We generate data with $n = 100$ by independently sampling each element of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ from a Unif$[-2.5, 2.5]$ distribution, and then taking $\boldsymbol{y} = f(\boldsymbol{x}_1, \boldsymbol{x}_2) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \text{MVN}(\mathbf{0}, \boldsymbol{I}_n)$. The four mean models $f(\boldsymbol{x}_1, \boldsymbol{x}_2)$ we consider are shown in Figure 12. Note that these are the same mean models we consider extensively in Section 3.

For each mean model, we generate 1000 replicates of data and estimate the mean model using (3) with $\lambda = 0$ and various q. We plot the MSE, squared bias, and variance of the mean model estimate as a function of q in Figure 12. In Scenarios 1 and 2, $q = 2$ has the best performance, which is unsurprising given the mean model structure. Using $q = 2$, there will be four bins whose boundaries roughly coincide with the true boundaries of the mean model. As q increases, the bias increases in an oscillating fashion where even values of q give better performance than odd ones. This is because odd values of q will not tend to have bins with boundaries that coincide with the true boundaries the mean model. As q increases, most of the q^2 bins will not have observations in them, and their estimates will be the mean of $\boldsymbol{y}$. Thus the variance decreases as many bins take on the same value, but the squared bias continues to increase. In Scenarios 3 and 4, the minimum MSE occurs at $q = 4$, not $q = 2$ as in Scenarios 1 and 2. This is because the mean models in Scenarios 3 and 4 are more complex and not well-estimated using only 2×2 grid of bins.

We also consider the performance for an additional mean model, shown in Figure 13(a). The same simulation set-up was used as for Scenarios 1–4 above. Though the blocks of the true mean model perfectly align with a grid that has $q = 3$, there are only 4 distinct blocks.

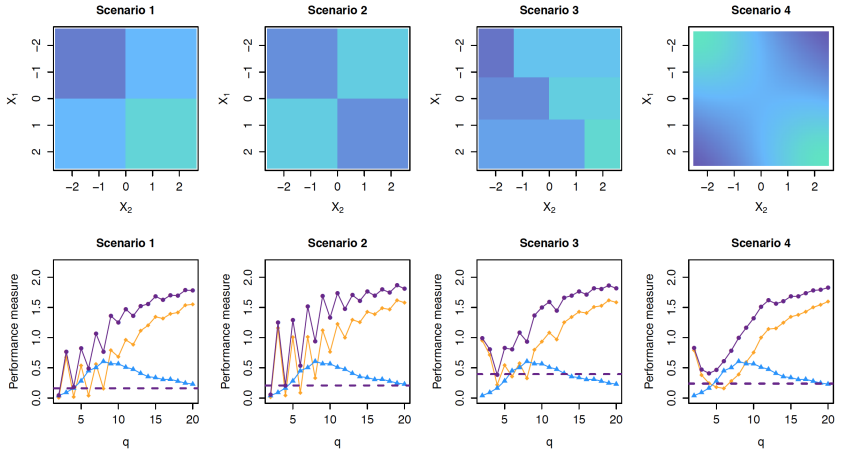


Figure 12: The top row of figures shows the mean models $f(x_1, x_2)$ used to generate data in each of the four scenarios in Appendix H. The bottom row of figures shows the performance of the method of (3) with $\lambda = 0$ as a function of q in terms of MSE (—●—), squared bias (—◆—), and variance (—▲—). The MSE for CRISP with $q = n$ and optimal λ is shown (- - -) for comparison.

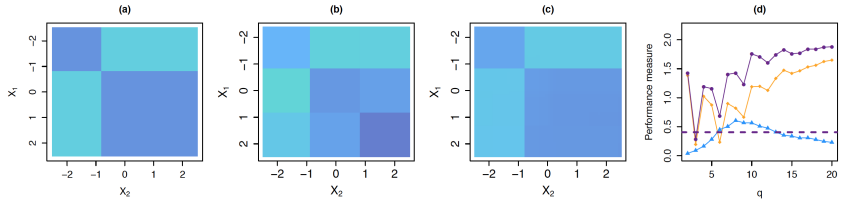


Figure 13: In *(a)*, we plot the mean model $f(x_1, x_2)$ used to generate data for the simulation described in Appendix H. In *(b)*, we show the estimated mean model from the method of (3) with $\lambda = 0$ and $q = 3$. In *(c)*, we show the estimated mean model from CRISP with $q = n$. In *(d)*, we show the performance of the method of (3) with $\lambda = 0$ as a function of q in terms of MSE (—●—), squared bias (—◆—), and variance (—▲—). The MSE for CRISP with $q = n$ and optimal λ is shown (- - -) for comparison.

The method of (3) with $\lambda = 0$ unsurprisingly has the best performance for $q = 3$, which is shown in Figure 13(d). The estimated mean model from using $q = 3$ and $\lambda = 0$ has $q^2 = 9$ blocks, as shown in Figure 13(b), since there is no adaptive shrinking together of blocks. However, CRISP is able to adaptively determine that only 4 blocks are needed, as shown in the estimated mean model in Figure 13(c). This example illustrates how CRISP is able to adaptively determine the amount of granularity over the covariate space. With $\lambda = 0$, the amount of granularity is constant across the covariate space.

Appendix I. Details of Data Application

In Section 6, we analyze housing data with the outcome of median house value and predictors of median income and average occupancy. We plot median income versus average occupancy in Figure 14. Note that 37 neighborhoods had an average occupancy larger than 10 and are omitted from the plot. The mean of average occupancy for these neighborhoods with an average occupancy greater than 10 was 88. In Figure 14, we outline the central 95% of the data in both covariates. That is, the 2.5% and 97.5% quantiles are shown for both covariates. We restrict our analysis to observations that fall in the central 95% of the data for both covariates. Of the original 20,640 neighborhoods, this excludes 1978 observations, leaving 18,662 observations for analysis.

References

Hirotugu Akaike. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, pages 267–281. Akademinai Kiado, 1973.

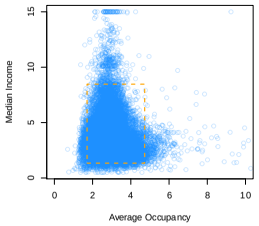


Figure 14: We plot median income versus average occupancy for the housing data considered in Section 6 and described in Appendix I. The rectangle (- - -) identifies observations falling within the central 95% of the data for both covariates.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. *Classification and Regression Trees*. CRC Press, 1984.

John Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *The Journal of Machine Learning Research*, 10:2899–2934, 2009.

Jean Duchon. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive Theory of Functions of Several Variables*, pages 85–100. Springer, 1977.

Bradley Efron. How biased is the apparent error rate of a prediction rule? *Journal of the American Statistical Association*, 81(394):461–470, 1986.

Jerome H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, pages 1–67, 1991.

Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

Trevor J. Hastie and Robert J. Tibshirani. *Generalized Additive Models*, volume 43. CRC Press, 1990.

Seung-Jean Kim, Kwangmoo Koh, Stephen Boyd, and Dimitry Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

Douglas Nychka, Reinhard Furrer, and Stephan Sain. *fields: Tools for spatial data*, 2014. URL `http://CRAN.R-project.org/package=fields`. R package version 7.1.

R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.

Ashley Petersen. *flam: Fits Piecewise Constant Models with Data-Adaptive Knots*, 2014. URL `http://CRAN.R-project.org/package=flam`. R package version 1.0.

Ashley Petersen, Daniela Witten, and Noah Simon. Fused lasso additive model. *Journal of Computational and Graphical Statistics*, forthcoming.

Aaditya Ramdas and Ryan J. Tibshirani. Fast and flexible ADMM algorithms for trend filtering. *Journal of Computational and Graphical Statistics*, forthcoming.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978.

Terry Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning and Regression Trees*, 2014. URL `http://CRAN.R-project.org/package=rpart`. R package version 4.1-8.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

Ryan J. Tibshirani and Jonathan Taylor. Degrees of freedom in lasso problems. *The Annals of Statistics*, 40(2):1198–1232, 2012.

Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

Sara van de Geer. *Empirical Processes in M-estimation*, volume 6. Cambridge University Press, 2000.

Jianming Ye. On measuring and correcting the effects of data mining and model selection. *Journal of the American Statistical Association*, 93(441):120–131, 1998.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Hui Zou, Trevor Hastie, and Robert Tibshirani. On the "degrees of freedom" of the lasso. *The Annals of Statistics*, 35(5):2173–2192, 2007.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 7/15; Revised 5/16; Published 5/16

JCLAL: A Java Framework for Active Learning

Oscar Reyes ogreyesp@gmail.com
Eduardo Pérez eperezp@facinf.uho.edu.cu
Department of Computer Science
University of Holguín
Holguín, Cuba

María del Carmen Rodríguez-Hernández 692383@unizar.es
Department of Computer Science and Systems Engineering
University of Zaragoza
Zaragoza, Spain

Habib M. Fardoun hfardoun@kau.edu.sa
Department of Information Systems
King Abdulaziz University
Jeddah, Saudi Arabia

Sebastián Ventura sventura@uco.es
Department of Computer Science and Numerical Analysis
University of Córdoba
Córdoba, Spain
Department of Information Systems
King Abdulaziz University
Jeddah, Saudi Arabia

Editor: Geoff Holmes

Abstract

Active Learning has become an important area of research owing to the increasing number of real-world problems which contain labelled and unlabelled examples at the same time. JCLAL is a Java Class Library for Active Learning which has an architecture that follows strong principles of object-oriented design. It is easy to use, and it allows the developers to adapt, modify and extend the framework according to their needs. The library offers a variety of active learning methods that have been proposed in the literature. The software is available under the GPL license.

Keywords: active learning, framework, java language, object-oriented design

1. Introduction

In the last decade, the study of problems which contain a small number of labelled examples and a large number of unlabelled examples at the same time have received special attention. Currently, there are two main areas that research the learning of models from labelled and unlabelled data, namely Semi-Supervised Learning and Active Learning (AL). AL is concerned with learning accurate classifiers by choosing which instances will be labelled, reducing the labelling effort and the cost of training an accurate model (Settles, 2012).

Currently, there are several software tools which assist the experimentation process and development of new algorithms in the data mining and machine learning areas, such as Rapid Miner, WEKA, Scikit-learn, Orange and KEEL. However, these tools are focused to Supervised and Unsupervised Learning problems.

Some libraries and independent code that implement AL methods can be found on the Internet, such as Vowpal Wabbit, DUALIST, Active-Learning-Scala, TexNLP and LibAct. The Active-Learning-Scala and LibAct libraries are mainly focused to AL, they implement several AL strategies that have been proposed in the literature. On the other hand, Vowpal Wabbit, DUALIST and TexNLP have been designed for a different purpose, but they also include some AL methods.

To date, and in our opinion, there has been insufficient effort towards the creation of a computational tool mainly focused to AL. In our view, a good computational tool is not only a tool which includes the most relevant AL strategies, but also one that is extensible, user-friendly, interoperable, portable, etc.

The above situation motivated the development of the JCLAL framework. JCLAL is an open source software for researchers and end-users to develop AL methods. It includes the most relevant strategies that have been proposed in single-label and multi-label learning paradigms. It provides the necessary interfaces, classes and methods to develop any AL method.

This paper is arranged as follows: Section 2 provides a general description of the JCLAL framework. The Section 3 presents an example for using the software. Finally, the documentation and the requirements of this software are outlined in Section 4.

2. The JCLAL Framework

JCLAL is inspired by the architecture of JCLEC (Ventura et al., 2007; Cano et al., 2014) which is a framework for evolutionary computation. JCLAL provides a high-level software environment to perform any kind of AL method. It has an architecture that follows strong principles of object-oriented programming, where it is common and easy to reuse code. The main features of the library are the following:

- Generic. Through a flexible class structure, the library provides the possibility of including new AL methods, as well as the ability to adapt, modify or extend the framework according to developer's needs.
- User friendly. The library has several mechanisms that offer a user friendly programming interface. It allows users to execute an experiment through an XML configuration file.
- Portable. The library has been coded in the Java programming language. This ensures its portability between all platforms that implement a Java Virtual Machine.
- Elegant. The use of the XML file format provides a common ground for tools development and to integrate the framework with other systems.

- Open Source. The source code is free and available under the GNU General Public License (GPL). It is hosted at SourceForge, GitHub, OSSRH repository provided by Sonatype, and Maven Central Repository.

JCLAL aims to bring the benefits of machine learning open source software (Sönnenburg et al., 2007) to people working in the area of AL. The library offers several state-of-the-art AL strategies for single-label and multi-label learning paradigms. It uses the WEKA (Hall et al., 2009) and MULAN (Tsoumakas et al., 2011) libraries. WEKA is one of the most popular libraries which has several resources on supervised learning algorithms. On the other hand, MULAN is a Java library which includes several multi-label learning algorithms. For future versions, we hope to provide AL strategies related with multi-instance and multi-label-multi-instance learning paradigms.

Currently, the library provides the following single-label AL strategies: Entropy Sampling, Least Confident and Margin Sampling which belong to the Uncertainty Sampling category. Together with the Vote Entropy and Kullback Leibler Divergence strategies which belong to the Query By Committee category. In the Expected Error Reduction category, the Expected 0/1-loss and Expected Log-Loss strategies are included. One AL strategy belongs to the Variance Reduction family. The Information Density framework is also provided. More information about all of these single-label strategies can be found in (Settles, 2012). On the other hand, the following multi-label AL strategies are provided: Binary Minimum (Brinker, 2006), Max Loss (Li et al., 2004), Mean Max Loss (Li et al., 2004), Maximal Loss Reduction with Maximal Confidence (Yang et al., 2009), Confidence-Minimun-NonWeighted (Esuli and Sebastiani, 2009), Confidence-Average-NonWeighted (Esuli and Sebastiani, 2009), Max-Margin Prediction Uncertainty (Li and Guo, 2013) and Label Cardinality Inconsistency (Li and Guo, 2013).

The Stream-Based Selective Sampling and Pool-Based Sampling scenarios are supported. JCLAL provides the interfaces and abstract classes for implementing batch-mode AL methods and other types of oracle. Furthermore, the library has a simple manner of defining new stopping criteria which may change according to the problem. The library contains a structure which allows a set of listeners to simply define the events of an algorithm. The AL methods can be tested using the following evaluation methods: Hold-Out, k-fold cross validation, 5X2 cross validation and Leave-One Out. A method for actual deployment is also provided.

The library contains a set of utilities, e.g. algorithms for random number generation, sort algorithms, sampling methods and methods to compute, for example, AUC. A plug-in which permits the integration of the library with WEKA's explorer is also provided.

3. Using JCLAL

The library allows the users to execute an experiment through an XML configuration file as well as directly from Java code. A configuration file comprises a series of parameters required to run an algorithm. Below, an example of a configuration file is shown, which we call `MarginSampling.cfg`.

In this example, a 10-fold cross validation evaluation method is used on the data set `ecoli` located in the folder `datasets`. For each fold, 5% of the training set is selected to

construct the labelled set and the rest of the instances form the unlabelled set. A pool-based sampling scenario with the Margin Sampling strategy is used. The Naive Bayes algorithm is used as a base classifier.

```
<experiment>
 <process evaluation-method-type="net.sf.jclal.evaluation.method.kFoldCrossValidation">
  <rand-gen-factory seed="9871234" type="net.sf.jclal.util.random.RanecuFactory"/>
  <file-dataset>datasets/ecoli.arff</file-dataset>
  <stratify>true</stratify>
  <num-folds>10</num-folds>
  <sampling-method type="net.sf.jclal.sampling.unsupervised.Resample">
   <percentage-to-select>5.0</percentage-to-select>
  </sampling-method>
  <algorithm type="net.sf.jclal.activelearning.algorithm.ClassicalALAlgorithm">
   <stop-criterion type="net.sf.jclal.activelearning.stopcriteria.MaxIteration">
      <max-iteration>50</max-iteration>
   </stop-criterion>
   <stop-criterion type="net.sf.jclal.activelearning.stopcriteria.UnlabeledSetEmpty"/>
   <listener type="net.sf.jclal.listener.ClassicalReporterListener">
    <report-title>Margin-Sampling</report-title>
    <report-frequency>1</report-frequency>
    <report-directory>reports/ecoli</report-directory>
    <report-on-file>true</report-on-file>
   </listener>
   <scenario type="net.sf.jclal.activelearning.scenario.PoolBasedSamplingScenario">
    <batch-mode type="net.sf.jclal.activelearning.batchmode.QBestBatchMode">
     <batch-size>1</batch-size>
    </batch-mode>
    <oracle type="net.sf.jclal.activelearning.oracle.SimulatedOracle"/>
    <query-strategy type="net.sf.jclal.activelearning.singlelabel.querystrategy.
    MarginSamplingQueryStrategy">
     <wrapper-classifier type="net.sf.jclal.classifier.WekaClassifier">
       <classifier type="weka.classifiers.bayes.NaiveBayes"/>
     </wrapper-classifier>
    </query-strategy>
   </scenario>
  </algorithm>
 </process>
</experiment>
```

There are several ways to execute an experiment. One way is using the JAR file. For running the experiment just type:

```
java -jar jclal-1.0.jar -cfg "examples/MarginSampling.cfg"
```

After the experiment is run, a summary report which comprises information about the induced classifier and several performance measures is created.

4. Documentation, Requirements and Availability

The library is available under the GNU GPL license. A user manual and developer documentation which describes the software packages, examples, information to include new methods, API reference and running tests, is provided.

The software requires Java 1.7, Apache commons logging 1.1, Apache commons collections 3.2, Apache commons configuration 1.5, Apache commons lang 2.4, JFreeChart 1.0, WEKA 3.7, MULAN 1.4 and JUnit 4.10 (for running tests). There is also a mailing list and a discussion forum for requesting support on using or extending the framework.

Acknowledgments

This research was supported by the Spanish Ministry of Economy and Competitiveness, project TIN-2014-55252-P, and by FEDER funds.

References

K. Brinker. *From Data and Information Analysis to Knowledge Engineering: Proceedings of the 29th Annual Conference of the Gesellschaft für Klassifikation e.V. University of Magdeburg*, chapter On Active Learning in Multi-label Classification, pages 206–213. Springer Berlin Heidelberg, 2006.

A. Cano, J. M. Luna, A. Zafra, and S. Ventura. A classification module for genetic programming algorithms in JCLEC. *Journal of Machine Learning Research*, 1:1–4, 2014.

A. Esuli and F. Sebastiani. *Advances in Information Retrieval: Proceedings of the 31th European Conference on IR Research (ECIR)*, chapter Active Learning Strategies for Multi-Label Text Classification, pages 102–113. Springer Berlin Heidelberg, 2009.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An Update. *SIGKDD explorations*, 11(1):10–18, 2009.

X. Li and Y. Guo. Active learning with multi-label SVM classification. In *Proceedings of the 23th International Joint Conference on Artificial Intelligence*, pages 1479–1485, 2013.

X. Li, L. Wang, and E. Sung. Multi-label SVM active learning for image classification. In *Proceedings of the International Conference on Image Processing (ICIP)*, volume 4, pages 2207–2210. IEEE, 2004.

B. Settles. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool, 1st edition, 2012.

S. Sönnenburg, M. L. Braun, C. S. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K.-R. Müller, F. Pereira, C. E. Rasmussen, G. Rätsch, B. Schölkopf, A. Smola, P. Vincent, J. Weston, and R. C. Williamson. The need for open source software in machine learning. *Journal of Machine Learning Research*, 8:2443–2466, 2007.

G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. MULAN: a java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011.

S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás. JCLEC: a java framework for evolutionary computation. *Soft Computing*, 12:381–392, 2007.

B. Yang, J. T. Sun, T. Wang, and Z. Chen. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 917–926. ACM, 2009.

Journal of Machine Learning Research 17 (2016) 1-37 Submitted 12/13; Revised 9/15; Published 6/16

Integrated Common Sense Learning and Planning in POMDPs

Brendan Juba* BJUBA@WUSTL.EDU
Washington University
1 Brookings Dr.
St. Louis, MO 63130 USA

Editor: John Langford

Abstract

We formulate a new variant of the problem of planning in an unknown environment, for which we can provide algorithms with reasonable theoretical guarantees in spite of large state spaces and time horizons, partial observability, and complex dynamics. In this variant, an agent is given a collection of example traces produced by a reference policy, which may, for example, capture the agent's past behavior. The agent is (only) asked to find policies that are supported by regularities in the dynamics that are observable on these example traces. We describe an efficient algorithm that uses such "common sense" knowledge reflected in the example traces to construct decision tree policies for goal-oriented factored POMDPs. More precisely, our algorithm (provably) succeeds at finding a policy for a given input goal when (1) there is a CNF that is almost always observed satisfied on the traces of the POMDP, capturing a sufficient *approximation* of its dynamics and (2) for a decision tree policy of bounded complexity, there exist small-space resolution proofs that the goal is achieved on each branch using the aforementioned CNF capturing the "common sense rules." Such a CNF always exists for noisy STRIPS domains, for example. Our results thus essentially establish that the possession of a suitable *exploration policy* for collecting the necessary examples is the fundamental obstacle to learning to act in such environments.

Keywords: Partially Observed Markov Decision Process, Decision Tree Policies, PAC-Semantics, Noisy STRIPS, Non-monontonic Reasoning

1. Introduction

A central problem in artificial intelligence, first considered by McCarthy (1959) concerns how to enable a machine to autonomously operate in an environment. The dominant approach to this problem has been to first build a *model* of the environment, and then use the model representation to generate *plans*. While these models were originally built by hand, modern advances across AI have been powered by the substitution of *learning* for hand-crafted knowledge representations, and we will likewise here consider the problem of learning an environment in order to support planning.

Although such separation of concerns is generally considered good engineering practice, such a decomposition of the problem mediated by an explicit intermediate representation may be intractable even when the overall problem has efficient algorithms: Khardon and Roth (1997) famously presented an efficient algorithm for an NP-hard reasoning task over a learned DNF representation, for example. And, *both* learning models of interactive environments *and* planning in them directly is indeed usually infeasible if, e.g., one is learning a model that can express DFAs (Kearns and Valiant, 1994), or one is attempting to plan in a model that captures propositional STRIPS domains with a bounded time horizon (Bylander, 1994; Erol et al., 1995). Although it is feasible to learn and plan in environments that are generated by extremely simple models, the problem becomes much harder when we are trying to utilize incomplete state information, when simple models do not perfectly capture the environment's dynamics, and when the state space is large—indeed, any one of these alone poses a challenge that current work seeks to address, but for most interesting applications, we must confront all three. In this work, we propose an approach to formulating plans using integrated learning and reasoning that provably simultaneously addresses all three challenges.

1.1 Formulation of the Integrated Learning and Planning Problem

Naturally, the task of developing algorithms to address the hard instances of intractable problems is inherently hopeless; as both learning and planning feature many hard instances, we can only hope to guarantee good performance by carefully formulating the theoretical problem to avoid these hopeless instances. Indeed, perhaps the main contribution of this work is the formulation of the integrated learning and planning problem that "factors out" the difficult task of *exploring* the environment: We avoid the exploration problem by assuming that an *exploration policy* has been fixed, and only seeking to plan using actions this reference policy explores with non-negligible probability. Naturally, we will also restrict the policies we consider—this is the analogue of the usual restrictions on the representations we consider in standard machine learning problems. For the concrete families of policies we consider, then, the polynomial-time algorithms we present for learning and planning thus establish that the only inherent difficulty in the task is exploring the environment.

Exploration often captures the essence of the kind of "needle in a haystack" search task that is necessarily slow: Even exploring environments described by very simple rules can force algorithms to suffer a complexity that is linear in the number of environment states, such as "combination lock" environments (Kakade, 2003, Section 8.6). And yet, the formulation of, for example, the standard reinforcement learning task combines the exploration task with the tasks of learning and planning, denying the possibility of any analysis establishing reasonable bounds on the time or number of samples required for most standard environment classes. By shifting our attention to settings where exploration is either solved for us or guaranteed to be possible, we will be able to obtain algorithms of polynomial complexity in the number of attributes describing the states, as opposed to the number of actual states (which is of course exponentially larger). This complexity bound is similar to what is achieved by the relaxation of Kakade and Langford (2002) for fully observed environments with a value function (as opposed to goal satisfaction), but using a different assumption.

We will aim to learn the dynamics (and observation model) of a *factored Partially Observed Markov Decision Process* (POMDP) with partial but noiseless observations. Pre-

*. Work originally performed while the author was affiliated with Harvard University and supported by ONR grant number N000141210358. Preparation of this article was supported by an AFOSR Young Investigator Award.

cisely, the state space we consider will be described by vectors of n Boolean attributes ("fluents" in the planning literature), and the agent's observations will be of the settings of a *subset* of these attributes. In our model, the agent may have access to some arbitrary *background knowledge* about the environment's dynamics. This knowledge is given to the agent as a collection of Boolean formulas over the state attributes that are assumed to be simultaneously true (with probability close to 1) on sequences of states generated by the environment. The agent also has access to a collection of observation histories that have been generated by the (arbitrary) exploration policy. The agent is then provided with a *goal*, represented by another Boolean formula over the attributes of the environment's state. The agent wishes to choose *actions* that guide the environment to a state satisfying this goal formula. We wish to use the background knowledge and example histories generated by the reference exploration policy to inform these choices of actions in the POMDP in order to achieve this new goal. That is, we are only seeking to learn the dynamics of the POMDP from this background knowledge and these example histories to the extent that it enables the agent to reach a goal state.

A naïve approach might directly estimate the dynamics of ("belief") distributions over the 2^n states produced by the model, but notice that the dynamics can require $2^{\Omega(2^n)}$ bits to represent. We note that unlike the full-information task, the history of past observations may provide additional information about the current state of a POMDP, and hence it is important to be able to construct a stateful policy. Following the spirit of McCarthy's approach, we propose to focus on cases where based on "common sense knowledge" that can be learned from the exploration policy's observations of the POMDP, there exists a *proof* that a given goal is satisfied on the actual, underlying trajectory generated by the policy and the POMDP.

Technically, the proofs serve to provide evaluations of the agent's performance under partial information: our "reward function" is given as an input formula that may refer to the unobserved portions of the state, separate from the domain examples. In order to determine that the reward function is satisfied when some information is missing, it is inherently necessary that we establish that none of the missing values could prevent the satisfaction of the goal—each setting of the missing data is either inconsistent with some background knowledge we have, or the goal condition is satisfied on it anyway. In particular, we do not assume that we have any model for the distribution for the missing values; these values are missing from the very data we use for learning. Addressing all of the possible values of such missing information is the essence of theorem-proving, and the existence of a simple proof gives us a potential means to efficiently consider all of these possible values for the missing data. In particular, we will assume that the goal is given as a DNF formula, and we will use treelike resolution proofs of bounded *Strahler numbers* (aka pebble number or clause space, cf. Section 2.2.1). This is a class of proofs that, even for Strahler number two, captures the kind of reasoning needed to verify the correctness of STRIPS plans, for example. Moreover, the reasoning problem for any constant Strahler number can furthermore be solved in polynomial time with the exponent depending on the Strahler number, cf. Kullmann (1999).

1.2 Techniques and Results

We propose here to use algorithms for reasoning over such common sense knowledge learned from partial information, as described by Juba (2013) for most known tractable proof systems, to encapsulate the learning of the POMDP's dynamics in planning. Indeed, we reduce the planning problem to answering such queries, establishing that such solutions to the integrated learning and reasoning problem are sufficient to solve our POMDP planning problem. This is analogous to Kautz and Selman's SATPLAN (Kautz and Selman, 1992), but for stochastic domains, with richer policy representations, and with the domain encoding *learned* from the partially observed traces; this learning is provided by the aforementioned algorithm from prior work by Juba, that stands in for the SAT-solver in this analogy.

Specifically, we will leverage the integrated learning and reasoning algorithm to find *decision tree policies of bounded Strahler number* for achieving the given goal in the POMDP (when they exist). A restriction to the space of decision tree policies was first considered for MDPs by Chapman and Kaelbling (1991) and essentially for POMDPs in the work of McCallum (1995, 1996); decision list policies (an example of *bounded* Strahler number decision tree policies) were considered by Khardon (1999). The restriction of the space of policies to more general bounded Strahler number trees is also natural: it encompasses bounded-fault tolerant policies (Jensen et al., 2004), for example. Such policy-space restrictions serve two relatively well-understood purposes: as with restrictions on the class of representations in learning theory, a restriction on the space of policies enables us to statistically distinguish good policies from bad policies (Kearns et al., 2002), and moreover enables us to design algorithms to *find* policies efficiently (e.g., as in the reduction of Mansour, 1999).

Our reduction adapts the algorithm of Ehrenfeucht and Haussler (1989) for supervised learning of decision tree classifiers of bounded Strahler number to search for policies. (Just like Ehrenfeucht and Haussler (1989), we can also search through all decision trees of a given size in *quasipolynomial* time, i.e., time $2^{\text{poly}\log n}$.) This algorithm proceeds by recursively building a tree below a candidate branch, using the ability to test when a branch is consistent with the input examples, terminating successfully when such a consistent branch is found. The key insight of Ehrenfeucht and Haussler was that a Strahler-number restriction enables the algorithm to control the amount of branching performed during this recursive search, by allowing us to search over the smaller (lower Strahler-number) subtrees first.

In our setting, deciding whether or not a branch is "consistent" means deciding whether or not the goal is achieved on the (example) traces that are consistent with the tests and actions selected on the branch. Such a test is thus naturally accomplished by a query to the integrated learning and reasoning algorithm of Juba (2013). The decision tree search algorithm also must be extended to search over *actions*, as opposed to just branches on settings of the observed attributes, which are all that the original decision trees use. We control this search indirectly, by partitioning the set of example traces according to the mutually exclusive choice of next action, and only passing the corresponding subset of the examples to the recursive call; the amount of work performed by the algorithm can be shown to be linear in the number of examples, so it is easy to bound the total work performed by these branches. We give an analysis of the overall algorithm showing in particular that only polynomially many traces are needed to learn the POMDP well enough to find policies.

This in turn crucially exploits the problem formulation: we only promise to find policies using actions that the reference exploration policy is likely to consider.

As promised, we note that this system finds plans even when the domain's dynamics are only partially, approximately captured by a small CNF formula (and hence, by the premises in a small resolution proof). For example, logical encodings of planning problems typically use "frame axioms" that assert that nothing changes unless it is the effect of an action. In a real world setting, these axioms are not strictly true, but such rules still provide a useful approximation. It is therefore crucial that we can learn to utilize such imperfect logical encodings. We will more generally be able to learn to plan in noisy versions of STRIPS instances (Fikes and Nilsson, 1971) (a standard test domain of interest) which are approximated well but imperfectly by their noiseless versions. Also, as a consequence of our adoption of *PAC-Semantics* (Valiant, 2000a) for our logic, we will also be able to soundly incorporate explicitly specified background knowledge into our algorithms, even when this knowledge may likewise only hold in an approximate sense.

In order to support some more interesting cases of planning under partial information, we also extend the PAC-Semantics slightly to non-monotonic logics using the Well-Founded Semantics (van Gelder et al., 1991) for negation-as-failure. The Well-Founded Semantics for propositional logics is a relatively clean semantics that features polynomial-time reasoning, and allows us to formulate, for example, "generic" frame axioms as explicit background knowledge; that is, frame axioms that do not depend on the unknown dynamics of the environment. These explicit frame axioms in turn allow the agent to reason about attributes of the environment that are unobserved in the traces collected by the exploration policy. The agent can then, for example, plan to achieve goals that refer to portions of the environment that cannot be simultaneously observed.

2. Preliminaries

Before stating the problem we address, we must describe the relevant models of environments, approximate reasoning, policies, and typical experiences.

2.1 Factored POMDPs

Informally, a POMDP is a state-based model of an environment for an agent that evolves probabilistically and only provides the agent with partial information about its states.

Definition 1 (POMDP) *A* Partially Observed Markov Decision Process (POMDP) *is given by collections of distributions on a* state space S *and an* observation space O *as follows: there is an* action set A *such that for each action* $a \in A$ *and* $s \in S$*, there is a distribution* $D_{(a,s)}$ *over* S*, and for each* $s \in S$ *there is a distribution* D_s *over* O*.*

For any fixed sequence of actions $a^{(1)}, a^{(2)}, \ldots$ from A, the distributions $\{D_{(a,s)}\}_{(a,s)\in A\times S}$ give rise to a (non-stationary) Markov process on S: given an initial state $s^{(1)}$, $s^{(2)}$ is drawn from $D_{(a^{(1)},s^{(1)})}$, and generally thereafter, $s^{(i+1)}$ is drawn from $D_{(a^{(i)},s^{(i)})}$. For a sequence of states so generated, the distributions $\{D_s\}_{s\in S}$ now give rise to the POMDP distribution over observations: the ith observation $o^{(i)}$ is drawn from $D_{s^{(i)}}$. We normally think of the agent choosing $a^{(i)}$ on the basis of the history of interaction with the environment somehow,

i.e., with knowledge of $o^{(1)}, \ldots, o^{(i)}$ and $a^{(1)}, \ldots, a^{(i-1)}$. We refer to the agent's strategy for choosing such actions as a *policy*. One normally also fixes some kind of a reward (or loss) function over the states S to quantify how "good" or "bad" an agent's policy for acting in the environment is (we will elaborate on this shortly). For a fixed policy, a sample from the joint distribution over the actions and observations generated by the interaction between the POMDP and the policy is called a *trace* or *history*; a sample from the distribution over the actual underlying states, actions, and observations is a *trajectory*. Naturally, a history can be obtained from a trajectory by dropping the actual states.

We will not use the most general definition of a POMDP; we will instead use the following natural special case featuring most of the key features of a POMDP. We will assume that the state space is *factored*, that is, described by n propositional variables (*"fluents,"* in the usual language of planning), taking $S = \{0,1\}^n$ (we will continue to denote the indices of these propositional variables in S by subscripts, hence our use of superscripts to denote the sequence of states). We choose to use propositional representations because they are sufficient to capture the only cases of first-order representations (to our knowledge) that have tractable learning and inference algorithms—note that Haussler (1989) shows that fitting first-order expressions even in very simple domains is intractable. By contrast, following Valiant (2000a) for example, we could consider relational representations of small arity and a polynomial size universe of objects, and take the atomic formulas obtained by various bindings of our relations over these objects as our propositional variables. But, as the propositional case is surely simpler and suffices for the current work, we will not consider first-order representations further. Thus, in particular, we will fix a (possibly polynomially related to n) horizon bound T, and for each tth step (of a candidate plan) and ith component of the state of the POMDP, we will have a separate propositional variable $s_i^{(t)}$.

The *observations* will then have the form of state vectors with some of their entries masked. More precisely:

Definition 2 (Partial states) *A* partial state ρ *is an element of* $\{0,1,*\}^n$*. We say that a partial state* ρ *is* consistent *with a state* $s \in \{0,1\}^n$ *if whenever* $\rho_i \neq *$*,* $\rho_i = s_i$*.*

The observations will consist of partial states produced by a fixed *masking process* applied to the current state of the POMDP (Michael, 2010):

Definition 3 (Masking process) *A* mask *is a function* $m : \{0,1\}^n \to \{0,1,*\}^n$*, with the property that for any* $s \in \{0,1\}^n$*,* $m(s)$ *is consistent with* s*. A* masking process M *is a mask-valued random variable (i.e., a random function).*

Notice, which attributes are hidden can depend arbitrarily on the underlying state. Masking leads to a kind of perceptual aliasing if the distinguishing attributes in two distinct states are masked; indeed, there may even be propositional variables that the masking process *never* reveals, that have an arbitrary effect on the dynamics (as they correspond to distinct states). In such a case, we cannot learn about the relationship of these missing variables to the larger dynamics of the POMDP using the observations alone. We may have *background knowledge* that mentions these variables though, thereby permitting us to reason about their contents. We shall discuss this further shortly.

In contrast to the most typical set-up, our reward function is *not* fixed with the POMDP, and in particular its value may not be determined by the partially observed example traces. It will be specified to the algorithm by a *goal predicate* G given by a DNF over the propositional state variables. Along these lines, an agent's policy is considered good if it manages to (quickly) reach a state in which G is satisfied (as opposed to demanding a policy that maintains a high average reward over time).

Noise. This model assumes that there is no noise in the observations themselves. Although as given this does not accurately reflect robotics problems (for example), it is a reasonable model for agents that interact with computer systems, for example a web-crawling agent.[1] In such applications, only a portion of the environment's state, such as a single web-page, single directory, etc. is visible at a time, but perception of this state is not at issue; the actions available to the agent then would correspond to the interface elements (e.g., links, buttons, etc.). In any case, it happens that noise in the observations will only affect our approach by possibly leading us to evaluate the goal predicate incorrectly, and therefore leading the policy to terminate prematurely. Otherwise there is not much difference between noise (stochasticity) in the dynamics—which we *do* consider, cf. Section 2.3.2 for an illustration—and noise in the observations.

2.1.1 Typical Actions

Our objective is to enable the agent to utilize knowledge that it may learn from its experiences. This means that the agent reasons about knowledge drawn from its particular experiences, which consist of a fixed collection of traces. We suppose that the agent's histories are generated by some arbitrary policy Π^*. In a fixed sample, we can only hope to evaluate actions that Π^* takes with probability polynomially related to the size of the sample, which are then explored reasonably well; conversely, we can also guarantee that all of the actions that Π^* takes with some minimum probability μ are well-explored in a sample of size polynomially related to $1/\mu$. So, the probability that Π^* explores a sequence of actions approximately characterizes the sequences that are well-represented in a sample of traces of a given size. In particular, since distinct sequences of actions refer to disjoint events, there are at most $1/\mu$ distinct sequences of actions that are taken with probability μ. A lower bound on this probability therefore bounds the number of sequences of actions we have to consider, and so the policy Π^* does all of the work in "exploring" the POMDP.

Conceptually, the policy Π^* will serve as a reference point that defines "typical" actions in the POMDP, and we will only expect the agent to understand the dynamics of the POMDP so far as it concerns traces that have some non-negligible weight under Π^*. Such settings reasonably capture everyday cases where the agent has been taught or has learned relevant sub-policies, or where the relevant plan is merely very simple. Some such guarantee seems to be necessary in order to obtain algorithms of reasonable complexity: Kakade (2003, Section 2.5) shows that even very simple fully-observed environments can be hard to learn, depending linearly on the number of possible states, which may be exponential in the number of fluents, or exponentially on the time horizon.

1. Indeed, the original motivation for considering this model arose from the desire to develop devices and distributed/networked software that featured "universal" (or at least broad) compatibility (Goldreich et al., 2012; Juba, 2011).

We further suppose that the agent's histories using Π^* are given together with the probabilities that Π^* chose its actions on each step in the history. These probabilities, although a somewhat nonstandard addition, are usually easy to generate (or at least estimate) while an algorithm computes a policy, and provide a convenient way to re-use the traces to estimate the quality of alternative policies via importance weighting, an observation due to Precup et al. (2000, 2001) for POMDPs. (The technique is discussed for full-information MDPs by Sutton and Barto, 1998, Chapter 5.)

Formally now, we will only hope for the agent to choose a policy consisting of sequences of actions $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(t)}$ such that Π^* takes the sequence of actions from the initial state of the POMDP with probability at least μ for some $\mu > 0$. For example, if Π^* is a random walk policy and the space of possible actions is small, then every short sequence of actions will be "typical" and we will expect the agent to be able to find any suitable short plan in such a case. But, Π^* may also describe some more sophisticated policy for exploring the relevant portions of the POMDP, or may be simply the time-average of the various policies used by the agent in the past (in which case such an estimator was considered by Shelton, 2001). We remark that the distribution over the initial states of the POMDP could be taken to be a "typical state" initial distribution such as the long-run state distribution under Π^*. Regardless of the details of this policy, we then reach the following definition of a typical sequence of actions:

Definition 4 (μ-typical) *If Π^* produces the sequence of actions $(\alpha^{(1)}, \ldots, \alpha^{(t)})$ with probability at least μ in a POMDP, then we say that the sequence of actions is* μ-typical *for Π^* in the POMDP.*

We stress that algorithms that achieve a sample complexity m for learning and planning in the POMDP under most conventional models can be used as an exploration policy for producing traces in which the (significant) actions of the plan are $1/\mathrm{poly}(m)$-typical. Otherwise, the algorithm would not have an adequate estimate of the quality of the traces to produce a reliable plan.

Another definition that similarly evaluates policies with respect to some fixed (reset) distribution was proposed previously by Kakade and Langford (2002) for learning in fully observed MDPs. The assumption is of course quite different, and their setting attempts to optimize a value function (with respect to this reset distribution) rather than achieving a goal. Nevertheless, much like in Kakade and Langford's work, we will be able to avoid an exponential dependence on the time horizon or number of attributes in the time and sample complexity of our algorithm as a result of our definition. A related assumption for POMDPs was introduced by Bagnell et al. (2004), which assumes that the fixed distribution is *close* to that induced by the target policy in statistical distance.

2.1.2 Knowledge Representations and STRIPS Domains

As we noted earlier, the special case of a factored POMDP that we consider here is naturally captured by a propositional logic in which there is a propositional variable for each ith fluent of the POMDP on each tth step of the trajectory. In this way, we can use propositional logic to reason about the actual, underlying state that the POMDP could be in on each step, and thus cope with our missing observations. In general, we will have a collection of propositional formulas (i.e., formed by the usual Boolean connectives over propositional

variables) capturing basic knowledge about our encoding of the POMDP, or specific to the actual POMDP under consideration. The entire collection is referred to as the *"background knowledge"* or *"knowledge base"* (abbreviated as "KB"). We will generally assume that the formulas in the knowledge base hold simultaneously with probability close to 1 over the distribution of sequences of states generated by the reference policy and POMDP.

It will be helpful for us to distinguish between the partial state as an observation and as a state of knowledge. In particular, looking ahead, our decision-tree policies will produce actions based on the three-valued observation vectors, not the actual state—so for example, it is meaningful for the policy to branch on whether or not an attribute's value is observed. Supposing we denote our observation in the tth step by $o^{(t)}$, we will encode $[o_i^{(t)} = b]$ for $b \in \{0, 1, *\}$ as variables $x_{i,b}^{(t)}$; we will add the clauses $x_{i,0}^{(t)} \vee x_{i,1}^{(t)} \vee x_{i,*}^{(t)}$ and $\neg x_{i_b}^{(t)} \vee \neg x_{i,b'}^{(t)}$ for all i and $b \neq b'$ to the background knowledge thus asserting that x_i takes exactly one of the three values.

When reasoning about plans for the POMDP, we will naturally likewise use a propositional representation: for each possible action α and action taken by the agent $a^{(t)}$, we will have a vector of propositional variables $a_\alpha^{(t)}$ encoding $[a^{(t)} = \alpha]$. Naturally, these propositional variables should be mutually exclusive. Typically, we will add a list of clauses to our background knowledge for every pair of actions $\alpha \neq \alpha'$ asserting that only one is taken, i.e., $\neg a_\alpha^{(t)} \vee \neg a_{\alpha'}^{(t)}$.

STRIPS domains. A class of simple domains that are sufficiently general to provide a variety of excellent examples of propositional domains were first introduced for the STRIPS planner of Fikes and Nilsson (1971). In a (propositionalized) STRIPS domain, the actions are described by lists of literals capturing the action's *preconditions* and *effects*. The intention is that the preconditions must be satisfied at time t in order for the action to be available to the agent. Once the action is taken, the state of the environment is updated so that at time $t + 1$, the literals capturing the effects are all satisfied. All other literals are unchanged. We will take the convention that if the preconditions of an action are not satisfied at time t but the agent tries to "take the action at time t," the action fails—as opposed to being unassertable, as traditionally required; this will be useful in partial information settings where the agent may not be certain whether or not the preconditions hold. Finally, the goals are given by a conjunction of literals.

The encoding of the action rules into clauses is then very natural. Supposing the action α has preconditions $p_1, \ldots, p_r$ and effects $e_1, \ldots, e_s$, for each time t, we have for each effect e_i a clause $[p_1^{(t)} \wedge \cdots \wedge p_r^{(t)} \wedge a_\alpha^{(t)}] \Rightarrow e_i^{(t+1)}$. The convention that the attributes of the state are only changed by actions is encoded by *"frame axioms"* like so. For each literal ℓ over some state of the environment and each action $\alpha_1 \ldots, \alpha_k$ that has an effect that falsifies ℓ, i.e., has some $e_i = \neg\ell$, we have a clause $[\ell^{(t)} \wedge \neg a_{\alpha_1}^{(t)} \wedge \cdots \wedge \neg a_{\alpha_k}^{(t)}] \Rightarrow \ell^{(t+1)}$.

Note that our DNF goals generalize the STRIPS-style "conjunctive goals." Likewise, although the actions are not necessarily captured by STRIPS actions, and the evolution of our environment is generally *not* deterministic (and our approach does *not* require it to be so, cf. Section 2.3.2), such examples are helpful to keep in mind as examples of POMDPs for which the dynamics can be concisely described by a CNF. We stress that in spite of our focus on STRIPS-like examples, our approach will extend to the kind of environments that can be expressed by, e.g., grounded PPDDL (Younes and Littman, 2004), provided

that there exists a sufficiently simple policy (for a given goal) that is "fault-tolerant" or otherwise reasonably reliable overall.

2.1.3 An Example

We can illustrate (STRIPS) rules, POMDPs, and their relationship with an example of a simple domain based on the Gripper problem from the first International Planning Competition (IPC 1998). Informally, the problem captures an environment consisting of two rooms, which may contain several balls. The agent is a robot that can pick up and put down the balls and carry then between rooms. The goal is usually to carry all of the balls to a given room.

The states. We can describe the environment (POMDP) with a vector of Boolean attributes as follows: We will index the rooms by $r \in \{1, 2\}$, and the balls by $b \in B$ (for some other set B). We will let the variables x_r indicate whether the agent is in room r, which are mutually exclusive; the variables $x_{r,b}$ indicate whether ball b is in room r, and for each fixed b the $x_{r,b}$ are mutually exclusive; and the variables x_b indicate whether the agent is holding ball b, and these are also mutually exclusive. The agent's actions are a_r, meaning "go to room r"; a_b, meaning "pick up ball b"; and a_0, meaning "drop."

The rules. The state variables and actions are related by the following clauses, and these are the STRIPS-style rules that will approximate the dynamics of the POMDP. First, if the agent executes action a_r, then the x_r is set to 1 and for $r' \neq r$, $x_{r'}$ is set to 0, that is for each time index t, we have rules $\neg a_r^{(t)} \vee x_r^{(t+1)}$ and $\neg a_r^{(t)} \vee \neg x_{r'}^{(t+1)}$. In the usual language of STRIPS, $x_r^{(t+1)}$ and $\neg x_{r'}^{(t+1)}$ are the *effects* of a_r at t. Furthermore, if the agent is holding ball b, then $x_{b,r}$ is also set to 1 and $x_{b,r'}$ is set to 0. These correspond to clauses $\neg x_b^{(t)} \vee \neg a_r^{(t)} \vee x_{b,r}^{(t+1)}$ and $\neg x_b^{(t)} \vee \neg a_r^{(t)} \vee \neg x_{b,r'}^{(t+1)}$ (for $r' \neq r$). These are *conditional effects* of $a_r^{(t)}$ with the *precondition* $x_b^{(t)}$.[2] Now, if the agent is in room r and ball b is also in room r, then the agent may pick up b with its gripper; when this happens, the agent also drops any other balls it was holding. This is captured by clauses $\neg x_{b,r}^{(t)} \vee \neg x_r^{(t)} \vee \neg a_b^{(t)} \vee x_b^{(t+1)}$ for each r and b, and clauses $\neg a_b^{(t)} \vee \neg x_{b'}^{(t+1)}$ for $b' \neq b$. Here $x_{b,r}^{(t)}$ and $x_r^{(t)}$ are again the preconditions for (conditional effects of) the action (they must jointly hold for *some* room r, or at the end the agent fails to hold b). Finally, if the agent executes a_0, it simply ceases to hold any balls at all, that is, $\neg a_0^{(t)} \vee \neg x_b^{(t+1)}$ for all b. Now, we also require *frame axioms* indicating that the state does not change unless it is the effect of one of these actions as described above. Actually, these are elegantly stated using negation-as-failure, as we describe in Section 4, so we will not list out the classical form of these rules here.

The actual POMDP and validity of the STRIPS-style rules. As we hinted at above, the deterministic STRIPS-style rules will only *approximate* the true environment, which is actually a Markov decision process. In this Markov decision process, we will model an agent whose actions are not flawlessly performed: suppose that the pick-up actions fail 1% of the time, and that when carrying a ball from one room to another, the agent has a 1% chance of dropping the ball in each of the rooms (for a 2% chance overall of ending up

2. Strictly speaking, conditional effects are not supported by STRIPS, so this is a (standard) extension.

in the destination without holding the ball). That is, in states where the pick-up action a_b should have set x_b to 1, the agent only enters such a state with probability .99, and enters a state where x_b is (still) 0 instead with probability .01. Likewise, in states where x_b is 1, x_r is 1, and the agent takes action $a_{r'}$, then with probability .98 the agent enters the state where x_b is still 1, $x_{r,b}$ and x_r are 0, and $x_{r',b}$ and $x_{r'}$ are 1; but, with probability .01, x_b switches to 0 (the agent drops the ball upon arrival in room r'), and with probability .01, not only does x_b become 0, but moreover $x_{r,b}$ is still 1 and $x_{r',b}$ is still 0 (that is, the ball didn't get carried). We could also further envision an environment in which the balls may switch rooms without our agent's intervention, because they were moved by some other agent, for example. We will assume that the other effects (in particular, the room-switching effect of a_r) occur with probability 1, and thus the corresponding rules are actually *"*1*-valid"* in the language of PAC-Semantics that we will discuss next; the rules for a_r described above turn out to be *".*99*-valid"* except for the frame axiom that asserts that x_b should remain 1 (unless a_0 or $a_{b'}$ are executed), which is only *".*98*-valid."*

In a POMDP, the agent does not observe the complete states of these underlying Markov decision processes in general. In our case, partial states are generated by the following simple masking process: when x_r is 1 and $x_{r,b}$ is not 1, the variables $x_{r',b}$ for $r' \neq r$ are masked (always set to $*$ in the partial states appearing in the example traces and the partial states provided to the agent's policy when it is executed). Intuitively, this means that the agent does not "see" the balls in room r', but may know that a ball b is in room r and not r'. In this simple environment, we will assume that all of the other attributes remain unmasked. Therefore, all of the rules that do not mention the $x_{r,b}$ variables are always *"witnessed true"* (i.e., they always have a satisfied literal—we will define this formally later); it turns out that the rules that *do* mention $x_{r,b}$—namely, the rules capturing the effects of a_r and a_b—also each have a satisfied literal and are therefore "witnessed true" with probability .99. Notice, $x_{r,b}$ (in the effects of a_r and $a_{r'}$) is only not "witnessed true" when both $\neg x_r$ and $\neg x_b$ are 0, and in this case, after a_r, $x_{r,b}$ and $\neg x_{r',b}$ are both guaranteed to be witnessed true unless the ball was dropped in room r', which occurs with probability .01. Similarly, in the rules describing the effect of a_b, either $\neg x_r$ or $\neg x_{r,b}$ is witnessed true (if the ball is absent), or else x_b is witnessed true unless the action fails, which occurs with probability .01. Since these STRIPS-style rules are all "witnessed true" with high probability, it will turn out that they are therefore guaranteed to be learnable in pursuit of policy construction, as we will see.

2.2 PAC-Semantics

Valiant (2000a) introduced PAC-Semantics in order to capture the property satisfied by the outputs of PAC-learning algorithms when formulated in a logic. For our purposes, PAC-Semantics will capture the extent to which a propositional formula holds on trajectories sampled according to the POMDP and some policy. We remark that Valiant's original motivation for formulating this semantics for a logic was to provide semantics for the representations created in the neuroidal cognitive model (Valiant, 2000b), and it is further proposed to capture certain kinds of learned "common sense" knowledge in general (Valiant, 2006). This roughly means that the agent possesses a large knowledge base without having been explicitly given most of its contents, as in the sense in which McCarthy (1959) used "common sense" when introducing the problem that we aim to address in this work. But, it also

so happens that this semantics features other hallmarks of "common sense" reasoning as identified in the AI literature, such as defaults and nonmonotonicity under conditioning, cf. Valiant (1995, 2000b) and Roth (1995). (We will not dwell on such aspects further in this work.) We note that systems using learning and logical reasoning based on PAC-Semantics have been successfully built and, for example, demonstrated to improve the accuracy with which a missing word in a sentence can be guessed (Michael and Valiant, 2008).

The key definition, capturing the notion of "approximation" in PAC-learning is:

Definition 5 ($(1-\epsilon)$-valid) *Given a distribution D, we say that a relation R is* $(1-\epsilon)$-valid *w.r.t. D if* $\Pr_{x\in D}[R(x)=1] \geq 1-\epsilon$.

Naturally, our relations will be given by formulas of propositional logic over the literals describing the observations over the n state attributes and the agent's actions for each time step $t = 1, \ldots, T$ over a trace of a policy in the POMDP. The distribution D we consider will generally be the distribution over actual states of the POMDP (and actions of the policy) given by the interaction between some policy and the POMDP, either the reference policy or else some (partial) policy under consideration by the algorithm.

Of course, since learning is impossible when *all* entries of our examples are hidden by a masking process, we must restrict our attention to settings where it is possible to learn something about D. We will consider formulas that can be evaluated in the straightforward way from the partial states with high probability:

Definition 6 (Witnessed and testable CNFs) *We say that a CNF φ is* witnessed to evaluate to true *on a partial state ρ if every clause of φ contains a literal ℓ such that $\ell(\rho) = 1$. If φ is witnessed true with probability at least p on partial states from some distribution over masked states $M(D)$ (usually given by a masking process M on a distribution D), we say that φ is* p-testable *with respect to $M(D)$.*

With respect to the individual clauses of the CNF, it is easy to see that only clauses that are witnessed to evaluate to true are known to be true in an example, and thus only clauses that are testable under D can possibly be learned from random examples.[3]

As we will only be able to learn the rules governing the POMDP's dynamics when they are witnessed true with high probability, it is important to note when rules are witnessed. In particular, consider the clause encoding a STRIPS action rule. In our model, the literals capturing the action are always observed. Thus, the rule is witnessed if either the effect of the action is observed *or* if some unsatisfied precondition is observed when the action fails. Likewise, a frame axiom expressing that the setting of a fluent ℓ persists at time t is witnessed when either an action is taken that changes the fluent ℓ, $\ell^{(t-1)}$ is observed to be false, or when $\ell^{(t)}$ is observed to be true.

2.2.1 Reasoning

We introduced the notions of propositional knowledge representations and PAC-Semantics in order to capture the process of reasoning about missing information. In this work, we

3. NB: these simple rules will only serve as the "base case" for reasoning, since our algorithms will further carry out reasoning based on them. Thus, in some sense, we also will "learn" knowledge that can be derived (easily enough) from these empirically observed rules.

are seeking algorithms for which we can provide theoretical guarantees of their performance. However, propositional reasoning was the original example of a (co-)NP-complete task (Cook, 1971). We will eventually recall a theorem due to Juba (2013) capturing a situation in which such reasoning is tractable. But, this theorem requires that we restrict the family of reasoning problems under consideration.

Although there are a variety of examples of families of reasoning problems for which algorithms exist, for our purposes the most natural family is described in terms of the *resolution* proof system. This is because resolution is one of the simplest propositional proof systems, and it simultaneously captures the capabilities of the most effective algorithms for reasoning in practice, cf. Beame et al. (2004). Recall, resolution is a proof system that operates on clauses. There is a single rule of inference in resolution called *cut* that allows one to derive new clauses: given a pair of clauses, $x \vee y \vee \cdots$ and $\neg x \vee z \vee \cdots$ that respectively contain an attribute x and its negation, cut allows one to derive the clause obained by taking the OR of the rest of the clauses, $(y \vee \cdots) \vee (z \vee \cdots)$.

Resolution is typically used to prove a DNF formula using a proof by contradiction: since the cut rule is sound, if we can derive the unsatisfiable, empty clause from an initial set of clauses, then the initial set must also have been unsatisfiable. If we are given that our background knowledge is true and we wish to prove that a DNF (say representing our goal) is satisfied, then by taking the negation of that DNF, we obtain a CNF via de Morgan's law; by showing that this CNF cannot be satisfied together with the background knowledge, we establish that the original DNF cannot be falsified, so it must be true.

For example, in our running example of the Gripper problem, suppose there are three rooms r, r', and r'', and two balls, b and b'. Suppose our goal for the moment is simply to not have the balls in the same room. This may be encoded by the DNF

$$[\neg x_{r,b} \wedge x_{r,b'}] \vee [\neg x_{r',b} \wedge x_{r',b'}] \vee [\neg x_{r',b} \wedge x_{r'',b'}].$$

Recall that we supposed that we had background knowledge capturing the fact that each ball is in exactly one room—in particular, for each of b and b', exactly one of $x_{r,b}$, $x_{r',b}$ and $x_{r'',b}$ is true. We would encode this by the clauses $x_{r,b} \vee x_{r',b} \vee x_{r'',b}$ (so at least one is true) and $\neg x_{r,b} \vee \neg x_{r',b}$, $\neg x_{r,b} \vee \neg x_{r'',b}$, and $\neg x_{r',b} \vee \neg x_{r'',b}$ (so at most one is true), and similarly for b'. Now, if we are in room r and we know that the ball b is in room r but the ball b' is not, then there is a resolution proof that the goal is satisfied as follows. We are now seeking to refute the negation of the goal,

$$[x_{r,b} \vee \neg x_{r,b'}] \wedge [x_{r',b} \vee \neg x_{r',b'}] \wedge [x_{r',b} \vee \neg x_{r'',b'}].$$

A resolution proof might proceed as follows: since we know $x_{r,b}$, our background knowledge that b is in at most one room, $\neg x_{r,b} \vee \neg x_{r',b}$ and $\neg x_{r,b} \vee \neg x_{r'',b}$ allows us to derive respectively $\neg x_{r',b}$ and $\neg x_{r'',b}$. In the second and third clauses of the negation of the goal, these allow us to derive $\neg x_{r',b'}$ and $\neg x_{r'',b'}$. But now, our background knowledge that b' is in some room, $x_{r,b'} \vee x_{r',b'} \vee x_{r'',b'}$ allows us to derive $x_{r,b'}$. But, we are given that b' is *not* in room r, i.e., $\neg x_{r,b'}$. So we can use this to arrive at the empty clause, a contradiction.

There is a natural graph associated with a proof: the clauses used in the proof appear as nodes of the graph, and for each derived clause in the proof, there is a directed edge from each of the clauses used in the derivation to the result. The sources of this graph are

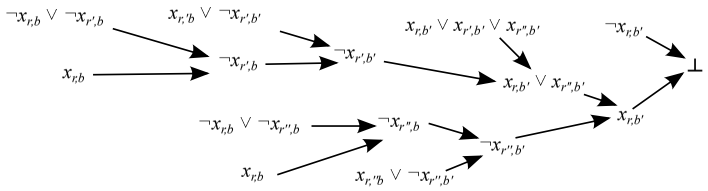


Figure 1: An example (treelike) resolution proof of $[\neg x_{r,b} \wedge x_{r,b'}] \vee [\neg x_{r',b} \wedge x_{r',b'}] \vee [\neg x_{r',b} \wedge x_{r'',b'}]$ from $x_{r,b}$, $\neg x_{r,b'}$, and the background knowledge.

labeled by premises of the proof, and the sink should be labeled by the empty clause. If this graph is a tree (we allow the same premise to label multiple leaves) then the proof is said to be *treelike*. Equivalently, a treelike proof is one that does not re-use intermediate derivations. The proof we just gave is treelike for example, cf. Figure 1. Any proof can be made treelike by simply repeating any intermediate steps; this comes at the price of increasing the number of steps, perhaps by up to a factor of two. Thus, the property of having a *small* treelike resolution proof (in terms of the number of steps) is a significant restriction. This restriction can be exploited to give efficient algorithms that guarantee that they find conclusions whenever such proofs exist. Again, we stress that without *some* such restriction, the problem of propositional reasoning is (co-)NP-complete.

A measure of tree complexity for proofs and policies. The main notion of "complexity" of a tree (either treelike resolution proof or decision tree) that we use is:

Definition 7 (Strahler number) *The* Strahler number *(aka* rank *or* pebble number*) of the nodes of a rooted binary tree are inductively defined as follows:*

- *all leaves have Strahler number one,*
- *nodes with two children of equal Strahler number s have Strahler number $s+1$, and*
- *nodes otherwise have Strahler number equal to the maximum of the Strahler numbers of their children.*

Finally, the Strahler number *of the tree is the Strahler number of its root.*

So, for example, our example treelike resolution proof given in Figure 1 has Strahler number three. Kullmann (1999) was the first to consider resolution proofs of bounded Strahler number, and showed that they can be found in polynomial time (where the polynomial depends on the Strahler number).

A significant property, noted by Ansótegui et al. (2008), is that derivations of Strahler number 2 correspond *precisely* to derivations using the well-known *unit propagation* rule, which in turn naturally simulates *chaining* with, e.g., Horn rules. In particular, our earlier example essentially followed from applications of the unit propagation rule, and the corresponding Strahler-2 derivation appears in Figure 2. (In general the Strahler-2 resolution proof follows the opposite derivation order from the unit propagation or chaining derivation.)

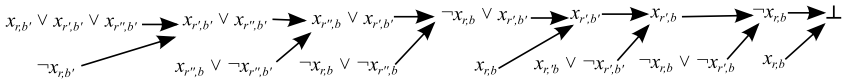


Figure 2: A Strahler-2 resolution proof of $[\neg x_{r,b} \wedge x_{r,b'}] \vee [\neg x_{r',b} \wedge x_{r',b'}] \vee [\neg x_{r',b} \wedge x_{r'',b'}]$ from $x_{r,b}$, $\neg x_{r,b'}$, and the background knowledge.

We also note that the Strahler number of a treelike resolution proof is always one less than the *clause space* of the proof, as defined by Esteban and Torán (2001): that is, the number of clauses that need to be remembered simultaneously to carry out the corresponding derivation. Indeed, if we consider derivations using a constant number of "registers" in which the registers' contents are deleted or overwritten upon their use in a derivation, then we obtain an alternative characterization of treelike proofs with bounded Strahler number.

These characterizations of bounded Strahler number proofs help clarify when such simple derivations exist. For example, plans (given by a set of action literals) in STRIPS domains can be proved to achieve their goals by chaining the action and initial state literals through the relevant clauses, and hence have derivations of Strahler number 2. Thus, it will be helpful to think of this work as giving a family of algorithms, parameterized by the Strahler numbers they use, as fixed Strahler numbers encompass natural syntactic classes of proofs and decision trees. (We will discuss the interpretation of bounded Strahler number decision trees later.)

We also briefly remark that Ansótegui et al. (2008) report experiments indicating that "industrial" instances of SAT that are easy in practice actually have (cf. the connection between modern SAT-solvers and resolution refutations from Beame et al., 2004) resolution (sub-)refutations with Strahler numbers that are significantly lower than those of random formulas of the same size and clause density. They propose that the property of having bounded Strahler number derivations might be what distinguishes instances that are easy in practice from those that are not.

Complex goals. We note that this environment model and class of proofs encompasses goals defined by arbitrary polynomial-size circuits, and not merely DNFs by a standard transformation: given a circuit defining a goal predicate, conceptually, we replace the POMDP we wish to solve with a new POMDP in which the states are extended with new propositional attributes that are always masked for each of the wires of the circuit, and the clausal (SAT) encoding of the constraints imposed by the circuit's gates is included as explicit background knowledge. Reasoning with resolution proofs of Strahler number two or greater will then use the background knowledge to evaluate the goal circuit: That is, once a setting for the inputs to a circuit has been guessed or fixed in some way, the unit propagation rule (which corresponds to a Strahler-2 proof) fills in the remaining, forced values of the circuit, much as it fills out the effects of a plan in an environment described by STRIPS rules. The propositional formula that we give to the algorithm then ultimately

consists only of the propositional variable encoding the output wire of the circuit, which then indeed indicates whether or not the goal is achieved.

2.2.2 Algorithms for Integrated Learning and Reasoning in PAC-Semantics

We now recall some efficient (but incomplete) algorithms for verifying the $(1-\epsilon)$-validity of a query DNF using a sample of partial assignments from our masked background distribution. Specifically, we recall an algorithm of Juba (2013) for verifying the $(1-\epsilon)$-validity of a formula when a Strahler-s treelike resolution proof of the formula from a $(1-\epsilon)$-testable CNF (and the KB, given as a CNF) exists.

Theorem 8 (cf. Theorem 13 of Juba 2013) *Let a KB CNF Φ and DNF query φ be given, and suppose that partial assignments are drawn from a masking process for an underlying distribution D' and are given together with weights w such that for the underlying example x, $w = D(x)/D'(x)$ with $w \leq W$; suppose further that for $\gamma > 0$ either*

1. *There exists some CNF ψ such that ψ is $(1-\epsilon+\gamma)$-testable under $M(D)$ and there is a Strahler-s treelike refutation of $\Phi \wedge \neg\varphi \wedge \psi$ or else*
2. *$[\Phi \Rightarrow \varphi]$ is at most $(1-\epsilon-\gamma)$-valid w.r.t. D*

Then, there is an algorithm running in time $O(\frac{W^2(|\Phi|+|\varphi|)}{\gamma^2} n^{2(s-1)} \log \frac{1}{\delta})$ (where $|\cdot|$ refers to the representation size in bits) that distinguishes these cases with probability $1-\delta$ when given φ, Φ, ϵ, γ, and a sample of $O(\frac{W^2}{\gamma^2} \log \frac{1}{\delta})$ partial assignments.

The cited result of Juba (2013) is for a sample of assignments from D directly without weights. Given that the algorithm simply computes an empirical estimate of an expectation over examples from $M(D)$, by reweighting each example from $M(D')$ by w, we obtain an unbiased estimate over $M(D)$ (at a cost of needing to scale γ by W to obtain our guarantee, yielding the quoted change to the sample complexity and running time).

As suggested earlier, works such as Precup et al. (2001) showed how such weights enable us to use importance weighting to use the traces produced by the exploration policy Π^* to estimate the probability with which a new, candidate policy Π achieves a goal. Since our goal predicate G will be given by a DNF, Theorem 8 allows us to check that a sequence of actions $\alpha^{(1)}, \ldots, \alpha^{(T)}$ results in the satisfaction of the goal predicate G in $s^{(T)}$, and hence test if a sequence of actions leads to the goal being achieved: the formula $[a^{(1)}_{\alpha^{(1)}} \wedge \cdots \wedge a^{(T)}_{\alpha^{(T)}}] \Rightarrow G(s^{(T)})$ is also a DNF, and hence Theorem 8 can be applied. Theorem 8 guarantees that if the algorithm reports that the formula is valid, then with probability $1-\delta$ over the examples, G will be satisfied on $s^{(T)}$ after these actions with probability $1-\epsilon-\gamma$ over the dynamics of the POMDP. At the same time, Theorem 8 guarantees that the algorithm reports that the formula is valid when some appropriate testable CNF exists, under which G can be proved from $a^{(1)}_{\alpha^{(1)}} \wedge \cdots \wedge a^{(T)}_{\alpha^{(T)}}$, e.g., a CNF encoding STRIPS rules for the environment if we are using Strahler-2 treelike resolution as our proof system.

2.3 Decision Tree Policies

The approach as described in the previous section—i.e., guaranteeing that the goal is satisfied given only the sequence of actions—is oblivious to the state of the environment, and hence only captures *"conformant"* planning (Goldman and Boddy, 1996). Such an approach

side-steps issues of partial information when it is possible, but of course such plans with high probability of success are unlikely to exist in most situations. On the other hand, introducing the *entire* sequence of observations (as a conjunction) is problematic as the space of possible observations is large, and under many natural environments it is unlikely that we would ever encounter a consistent sequence of observations even twice. It would then be infeasible to collect a large enough sample to learn the dynamics sufficiently well for such an approach to be effective. A natural alternative that incorporates some knowledge about the state of the environment at a "coarser" level is to use a policy that is computed by a (small) decision tree, as proposed by Chapman and Kaelbling (1991) in the context of fully observed MDPs, and essentially applied to POMDPs in McCallum (1995, 1996).

Definition 9 (Decision tree policy) *A* decision tree policy *is a rooted tree of degree* ≤ 2 *in which*

- *nodes of degree* 2 *are labeled by an attribute and an observed value for the attribute from the set of observable values* $\{0, 1, *\}$*, and one edge to a child node is labeled true, and the other false,*
- *other nodes (of degree zero or one) may be labeled with actions*

with the interpretation that given an observation $o \in \{0, 1, *\}^n$*, the agent takes the action (possibly, "no action") labeling the next such node on the branch starting from the previous action (or the root, if no such action exists) on which the true edge at each intermediate node is taken precisely when the attributes take the values on given node's label.*

If at most T *action nodes appear on any path, then we say that the policy is a* T-horizon decision tree.

A small decision tree policy is "coarser" in the sense that the actions of the policy only depend on the observed state (or lack of observation) of the attributes examined on a branch of the tree, which in general must contain far less than all of the attributes. We note that when the decision tree has some fixed polynomial size (in n), it has a polynomial number of leaves with polynomial length branches—and hence, if on each branch labeled by the literals $\ell_1, \ldots, \ell_k$, the formula $[\ell_1 \wedge \cdots \wedge \ell_k] \Rightarrow G$ is $(1-\epsilon)$-valid, then by a union bound, the policy will achieve G with probability at least $1 - \epsilon'$ for some ϵ' polynomially related to ϵ. Such a guarantee of $(1-\epsilon)$-validity can, in turn, be provided by the algorithm of Theorem 8 if a proof of $[\ell_1 \wedge \cdots \wedge \ell_k] \Rightarrow G$ (from some witnessed CNF) exists for each branch of the tree.

2.3.1 Strahler-s Decision Tree Policies

As hinted at in the previous section, we will also consider the Strahler number as a measure of decision tree complexity, specifying a restriction on the space of policies we consider. For a fixed Strahler number, the trees will indeed have a fixed polynomial bound on their size, and hence we can establish their quality leaf-by-leaf as discussed above. Moreover, a Strahler number bound will enable polynomial-time algorithms to *find* a policy when such "simple," provably good policies exist, and will guarantee the existence of quasipolynomial-time algorithms for general decision trees.

Much as with resolution proofs, the Strahler number of decision tree policies has a natural interpretation as follows. First, observe that "decision trees" of Strahler number 1 (i.e., a single branch) are precisely conformant plans. More generally, a policy of Strahler

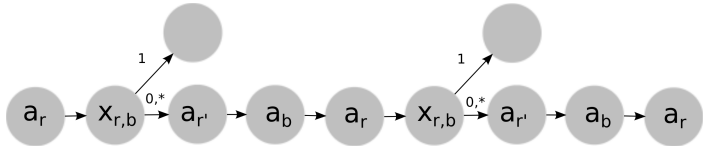


Figure 3: A Strahler-2 decision tree policy for the simple Gripper problem.

number s can be viewed as a decision list over Strahler number $s-1$ policies. That is, we have a hierarchically structured family of policies in which a sth level policy selects from among the members of level $s-1$ by testing a series of literals. (The policy may also take some actions amidst these tests, before selecting a lower-level policy to invoke.) We remark that, by testing the effects of each action after taking it, this captures a class of partial-information s-fault tolerant plans (cf. Jensen et al., 2004), where each fault is signaled by the first literal that did not end up satisfied. We note that Strahler-s policies can capture *all* such plans. We briefly remark that Strahler-s decision trees and the more popular OBDDs (Bryant, 1992) are of incomparable strength, depending on the variable ordering.

Alternatively, we can also characterize these policies in terms of the depth of nesting of the conditional branching: in a Strahler-s policy, at each conditional branch, there must be some sequence of branches such that the policy tests at most $s-1$ literals before terminating. Of course, this means that a policy that always tests k attributes to decide what action(s) to take has Strahler number k. Such a policy is a decision tree of size 2^k, and so may be very complex if k is even moderately large.

2.3.2 Example Decision Tree Policies

Returning to our running example based on the Gripper domain, we now describe a simple Strahler-2 decision tree policy for the goal $x_{r,b}$, i.e., carrying ball b to room r, that only fails on any given branch with probability at most .01. The policy is depicted in Figure 3. It first executes a_r, and branches on $x_{r,b}$; if $x_{r,b} = 1$, the agent terminates the execution (and the goal is satisfied w.p. 1 on this branch). Otherwise, the agent executes $a_{r'}$ for the other room r', and executes a_b. The agent then executes a_r, and branches on $x_{r,b}$. Again, if it is 1, the agent terminates and the goal is satisfied w.p. 1. Otherwise, the agent again executes $a_{r'}$, a_b, and a_r, and finally terminates. Notice, this final branch is only taken with probability $.01 + .01 \cdot .99 = .0199$, and conditioned on it having been taken, it only fails with probability .0199 again, so the overall probability of the branch ending in failure is $(.0199)^2 < .01$ (it is actually less than 0.04%), and it turns out that this is the total failure probability of this policy. The policy can be seen to have Strahler number 2 since at each test, (at least) one branch terminates with no further tests.

We note that a union bound over the failure probabilities of the two STRIPS-style rules only gives .98-validity for the conclusion that the ball should be in room r after the action sequence a_b, a_r, and the actual validity is .9801 since these failures are independent. This is (still) too high for the policy to safely terminate after executing it just once. The more

complex rule

$$[a_{r'}^{(t)} \wedge a_b^{(t+1)} \wedge a_r^{(t+2)} \wedge \neg x_{r,b}^{(t+3)} \wedge a_{r'}^{(t+3)} \wedge a_b^{(t+4)} \wedge a_r^{(t+5)}] \Rightarrow x_{r,b}^{(t+6)}$$

on the other hand, actually not only has validity $1-(.0199)^2$ (it is satisfied if either $x_{r,b}^{(t+3)}$ is 1 or $x_{r,b}^{(t+6)}$ is 1), but is also witnessed true with that probability in our environment (conditioned on that action sequence). Since this same conjunction of literals is asserted on the aforementioned branch of the policy, there is a Strahler-2 (chaining) derivation of the goal, $x_{r,b}$, using this witnessed rule.

Stochastic environments, iteration, and Strahler numbers. It should be evident that even if the failure probabilities were much higher than 1%—say it was p—by repeating the action sequence $a_{r'}, a_b, a_r$ k times, we can drive the probability of failure overall down to p^k; so, we can reach a desired ϵ failure probability in $3\log_p \frac{1}{\epsilon}$ steps (assuming the time horizon T is sufficiently large), and that such a policy still has Strahler number 2: it simply has a longer list of tests. In such a way, even if the environment is highly stochastic and is not well described by STRIPS rules, e.g., a more general PPDDL domain (Younes and Littman, 2004), there may still be rules referring to multiple states (e.g., repetitions of a faulty action) that are witnessed true with high probability and that guarantee that the goal is achieved. We stress that Theorem 8 (and hence our ultimate algorithm) only requires that such a rule exists to (implicitly) make use of it. We do not need to ever explicitly construct the rule, so it does not matter if the rule is complex, and it does not matter if all of the simple (STRIPS) rules are unreliable.

One change to the POMDP that would require a policy with a larger Strahler number would be if the number of rooms increased from 2 to R (since then it does not suffice from testing room r to determine that the ball is in the other room; more tests are needed). But, in this case there is a Strahler-3 decision tree policy: consider a decision list that iterates over the R rooms by invoking $a_{r'}$ for each room r', then branching on $x_{b,r'}$, moving on if $x_{b,r'}$ is not 1, but otherwise executing the Strahler-2 policy above for the room r' where $x_{b,r'}$ was 1. By our decision list characterization of the Strahler number, this is a Strahler-3 decision tree policy. By an essentially similar argument, it achieves the goal with similar reliability.

The role of reasoning. Back in the two-room environment, a more complicated goal is $x_{r,b} \wedge x_{r',b'}$ (where $r \neq r'$ and $b \neq b'$), that is, placing ball b in room r and ball r' in room b'. There is a natural Strahler-3 policy for this goal: the agent executes the Strahler-2 policy for $x_{r,b}$ until it would terminate, it then executes a_0 (dropping b in room r) and invokes the Strahler-2 policy for $x_{r',b'}$. (Since at every branch, at least one branch leads to a Strahler-2 policy, the overall policy is again guaranteed to have Strahler number 3.) It is easily verified that for this policy, we can bound the probability that each branch is taken and leads to failure by $2 \cdot (.0199)^2 - (.0199)^4$ which is less than 0.08%, and the overall probability of the policy failing is likewise less than 0.08%.

The verification of this goal is *not* trivial, however, since $x_{r,b}$ and $x_{r',b'}$ are never simultaneously unmasked. In the absence of any further knowledge about the environment, this would pose a problem. But, by providing the agent with explicit frame axioms as we describe in Section 4, we will see how the agent can still identify a successful policy. As

we noted above, for every branch of the initial policy for r and b, we have a proof that $x_{r,b}$ is satisfied. Now, after a_0 is executed, x_b is set to 0, and then none of the actions taken in the policy for r' and b' would ever falsify $x_{r,b}$. Therefore, the frame axioms for the literal $x_{r,b}$ can be invoked—and these are 1-valid here—and so by chaining $x_{r,b}$ across the subsequent steps, we obtain that $x_{r,b}$ is satisfied at the final step of the plan. Together with the derivation of $x_{r',b'}$ on the branch in question (variously described above for the simple one-ball policy) this allows us to *refute* $\neg x_{r,b} \vee \neg x_{r',b'}$ with a chaining proof, and hence we have a Strahler-2 derivation of $x_{r,b} \wedge x_{r',b'}$.

3. Learning and Planning

Algorithms for simultaneously learning and reasoning in PAC-Semantics (as captured in Theorem 8) turn out to provide a bridge between *learning* dynamics from examples and logic-based approaches to *planning*. Intuitively, Theorem 8 describes an algorithm that can verify when we have found a good branch of a policy by finding a proof (using knowledge learned from example traces) that the branch achieves the goal sufficiently well. Our algorithm is then based on the work of Ehrenfeucht and Haussler (1989) on learning decision trees with low Strahler number: once we can verify that individual branches are suitable, their analysis shows that the entire policy can be found efficiently via an algorithm that recursively searches for a decision tree policy that achieves the goal with high probability below a given, candidate branch.

Theorem 10 (Finding decision tree policies) *There is an algorithm that, when given a time horizon T, Strahler number bound s, $\mu, \gamma, \delta, \epsilon > 0$, KB CNF Φ, goal DNF G and action set A for an n attribute POMDP, runs in time $O(m|\Phi|(nT)^{4(s-1)})$ using at most $m = O(\frac{1}{\mu^2\gamma^2}(nT + \log\frac{|A|}{\mu} + \log\frac{1}{\delta}))$ weighted traces of an arbitrary policy Π^*. (Again, $|\Phi|$ refers to the representation size of Φ in bits, while $|A|$ refers to the cardinality of A.)*

Suppose there is a Strahler-s T-horizon decision tree policy Π taking μ-typical actions with respect to Π^* such that for each branch, there is a Strahler-s treelike resolution proof of "this branch is taken $\Rightarrow G$" from the KB and some CNF ψ that is $(1-\epsilon+\gamma)$-testable over traces of the POMDP with Π. Then with probability $1-\delta$, the algorithm finds a Strahler-s T-horizon decision tree policy that achieves G in the POMDP with probability $1-O((nT)^{s-1}(\epsilon+\gamma))$.

We stress that our focus is primarily on policies and proofs with small Strahler numbers, say $s = 2$, $s = 4$, etc., so that this algorithm is polynomial-time and obtains a polynomially bounded increase of the failure probability.

We require the following combinatorial lemma:

Lemma 11 *1. The number of branches k in a T-horizon decision tree policy over n fluents of Strahler number s with $n \geq s \geq 1$ is $2^{s-1} \leq k \leq (3enT/(s-1))^{(s-1)}$ where e is the base of the natural logarithm.*

2. At most $3^{3nT}\frac{T}{\mu}$ distinct sets of literals and actions label branches of any T-horizon decision tree policy over n fluents with μ-typical actions.

Proof Note that our T-horizon decision tree policy can be written as a decision tree over $3nT$ variables, $x_{i,b}^{(t)}$ for $i = 1, \ldots, n$, $b \in \{0, 1, *\}$, and $t = 1, \ldots, T$; therefore, the first part

Algorithm 1 Find-DT()

PAC-refute(φ, R) decides if $\varphi \wedge \psi$ is at most ϵ-valid given some testable ψ over weighted partial examples in R (taken out of m) (cf. Theorem 8).
Input: DNF goal G, CNF KB Φ, Strahler number s, policy branch Π taking $t-1$ actions, horizon bound T, sample of traces with stepwise weights R
Output: A Strahler-s decision tree or $\perp$
if Π takes more than T actions **then**
  return $\perp$
end if
$R^t \leftarrow \{(\times_{i=1}^{t-1}(o^{(i)}, a^{(i)}) \times o^{(t)}, \Pi_{i=1}^{t-1} w_i) : ((o^{(i)}, a^{(i)}, w_i)_{i=1}^{T}, o^{(T+1)}) \in R, \Pi_{i=1}^{t-1} w_i \leq \frac{4}{\mu}\}$
if PAC-Refute($\neg G^{(t)} \wedge \Phi \wedge \Pi$ taken, R^t) **then**
  return Π.
end if
if $s > 1$ **then**
  for all ℓ s.t. $\ell^{(t)}$ and $\neg\ell^{(t)}$ not in Π **do**
    $\Pi_\ell \leftarrow \Pi$ with a final test for $\ell^{(t)}$
    $\Pi_1 \leftarrow$Find-DT$(G, \Phi, s-1, \Pi_\ell, T, R)$
    if $\Pi_1 \neq \perp$ **then**
      $\Pi_0 \leftarrow$Find-DT$(G, \Phi, s, \Pi_{\neg\ell}, T, R)$
      if $\Pi_0 \neq \perp$ **then**
        return Policy following Π_b if $\ell^{(t)} = b$
      else
        return $\perp$
      end if
    end if
  end for
end if
for all α s.t. $\#\{\rho \in R : a^{(t)} = \alpha\} \geq (\mu/2)m$ **do**
  $\Pi_\alpha \leftarrow \Pi$ with $a^{(t)} = \alpha$
  $R_\alpha \leftarrow \{\rho \in R : a^{(t)} = \alpha\}$
  $\Pi' \leftarrow$Find-DT$(G, \Phi, s, \Pi_\alpha, T, R_\alpha)$
  if $\Pi' \neq \perp$ **then**
    return $a^{(t)} = \alpha$ followed by Π'
  end if
end for
return $\perp$

follows immediately from the first part of Lemma 1 of Ehrenfeucht and Haussler (1989). For the second part, we simply note that there are at most $\frac{1}{\mu}$ possible maximal sequences of μ-typical actions (i.e., that are not prefixes of one another). These sequences have at most T prefixes each, and for each of the $3nT$ variables, the branch may either omit the variable or check that the variable is true or false. The bound is now immediate. ■

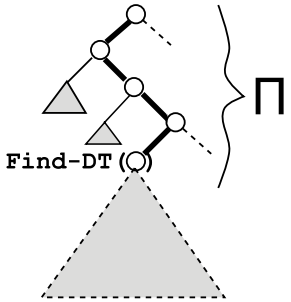


Figure 4: A snapshot of the execution of Algorithm 1: `Find-DT` is invoked at the indicated point at the end of the branch Π. `Find-DT` returns a subtree (shown in dashes) rooted at the end of Π such that the overall policy achieves the input goal G with high probability at all of the leaves of the subtree.

Proof (of Theorem 10) The algorithm, given as Algorithm 1, recursively searches for a sub-policy in which each leaf, with probability at least $(1 - \epsilon - \gamma)$, either achieves G or is not reached (see Figure 4). Intuitively, we are filtering our samples so that we obtain samples from a policy $\tilde{\Pi}^*$ that simulates Π^* until Π^* takes an action such that the action sequence would have overall probability less than $\mu/4$, and then takes a distinct "abort" action instead. Note that t-step traces in which $\tilde{\Pi}^*$ does not abort are precisely those with $\prod_{i=1}^{t} w_i \leq 4/\mu$. At each node, our learning and reasoning algorithm is invoked on the example traces to test if either G is satisfied or that branch is not taken with a $1-\epsilon$ fraction of the m examples (with examples off the branch removed from the sample since they are guaranteed to satisfy our target condition). If so, we have found a suitable leaf.

In $m \geq \frac{8}{\mu^2}(3nT \ln 3 + \ln \frac{4T|A|}{\mu} + \ln \frac{3}{\delta})$ example traces, the fraction of times a sequence of actions appears approximates its probability under $\tilde{\Pi}^*$ to within an additive $\mu/4$ with probability $1 - \delta/(3|A|M)$ where M is the number of distinct events corresponding to taking branches appearing in T-horizon decision tree policies labeled by $\mu/4$-typical actions, as given in part 2 of Lemma 11. In particular, since there are at most $|A|M$ action sequences that are either $\mu/4$-typical or not $\mu/4$-typical and minimally so, this guarantees that with probability $1 - \delta/3$, all of the action sequences which appear in a $\mu/2$-fraction of the traces are at least $\mu/4$-typical under $\tilde{\Pi}^*$, and that the $(3/4)\mu$-typical actions under $\tilde{\Pi}^*$ all appear in at least a $\mu/2$-fraction of the traces. In particular, since all of the μ-typical action sequences of Π^* are at least $(3/4)\mu$-typical under $\tilde{\Pi}^*$, all of these appear.

Soundness. We first note that as a consequence of our filtering the examples to be consistent with the branch Π, the weights we compute for our queries are

$$\prod_{i=1}^{t} w_i = \prod_{i=1}^{t} \frac{I[a^{(i)}_{\alpha^{(i)}} | \Pi \text{ followed for } 1,\ldots,i-1]}{\Pr_{\tilde{\Pi}^*}[a^{(i)}_{\alpha^{(i)}} | \Pi \text{ followed for } 1,\ldots,i-1]} = \frac{\Pr_{\Pi}[\Pi \text{ followed for } 1,\ldots,t]}{\Pr_{\tilde{\Pi}^*}[\Pi \text{ followed for } 1,\ldots,t]}$$

where these weights are bounded by $4/\mu$. Therefore m examples are sufficient to bound the probability of queries being satisfied over traces of Π by an additive γ with probability $1 - \frac{\delta}{3M}$. By a union bound over the M possible queries, every query is answered correctly w.p. $1 - \delta/3$.

As Theorem 8 guarantees that for each branch the formula asserting that either the leaf is not reached or the policy succeeds is $(1-\epsilon-\gamma)$-valid, the probability that the policy fails on each branch is at most $\epsilon+\gamma$. So, when a decision tree policy is returned, a union bound over the $O((nT)^{s-1})$ branches (by part 1 of Lemma 11) yields the claimed performance. So the algorithm only fails by returning a bad policy when some query is answered incorrectly, which occurs w.p. at most $\delta/3$.

Completeness. We already saw that the algorithm considers all sequences of μ-typical actions under Π^* with probability $1 - \delta/3$. Now, if the algorithm does not find a policy in which a branch with lower Strahler number takes no action, then if there is a policy with the given Strahler number bound, it must take *some* action at the next step, and otherwise, (i.e., if a sub-policy was found) there is no harm in asserting the opposite setting of $x^{(t)}_{i,b}$, since adding this literal to the formula for the branches of the target policy only increases the probability of the branch not being taken (and thus improves the success probability for that branch). In particular, the original Strahler-s tree is still a candidate. Theorem 8 also guarantees that for the quoted number of examples, all of the branches of the target policy will be accepted w.p. at least $1 - \delta/3$ as needed. Thus overall the algorithm succeeds at returning a good policy when it exists w.p. $1 - \delta$.

Time complexity. Finally, the work per recursive call of our algorithm involves partitioning the examples and running `PAC-refute` on the sample, which is linear in the sample size $|R|$. So, given $W \cdot m$ work per node with $|R| = m$, the running time bound for the algorithm of $O(W \cdot m(nT)^{2(s-1)})$ is easily established by considering the following recurrence with boundary conditions $f(0, m, s) = 0$, $f(n', m, 0) = W \cdot m$ over $n' = (3n+1)T$, counting the possible remaining literals plus one action for each time step: the bound is a solution to the recurrence

$$f(n', m, s) \leq f(n'-1, m, s) + 2n' f(n'-1, m, s-1) + W \cdot m.$$

The running time of the algorithm also satisfies this recurrence since for such an f linear in m, if we take $m_\alpha = |R_\alpha|$ in Algorithm 1, $\sum_{\alpha \in A} f(n'-1, m_\alpha, s) \leq f(n'-1, m, s)$, and indeed our algorithm only iterates over actions (in which it partitions the example set among the recursive calls) if it does not make a Strahler-s recursive call while searching for a branch over the literals. Thus the running time satisfies the given bound satisfying the recurrence.

We now merely observe that the work by the learning and reasoning algorithm per node is $W \cdot m = O(|\Phi|(nT)^{2(s-1)}m)$, giving the claimed time bound. ■

It follows from Theorem 10 that we can find *general* decision tree policies taking μ-typical actions in quasipolynomial time since if a policy has B branches, Lemma 11 guarantees it has Strahler number at most $s = \log B + 1$.

3.1 An Illustration of the Algorithm

We return once again to our running example of the simple goal in the Gripper domain to illustrate the behavior of our algorithm; recall that in Section 2.3.2 we described a Strahler-2 decision tree policy for the goal $x_{r,b}$ (carrying the ball b to room r in a two-room environment), that had a Strahler-2 proof from a witnessed CNF that the goal was achieved with probability greater than 99% on each branch (it actually had probability greater than 99.96%). The policy itself appeared in Figure 3.

There is actually a conformant (Strahler-1) policy that achieves the goal with similar probability. Indeed, the policy we discussed in Section 2.3.2 does not need to examine the state of the environment. Our analysis of the plan could be applied about as well to the plan that simply applies the sequence of actions "move to room r', try to pick up the ball, move to room r, drop the ball" twice. In this case, when the plan is successful, the goal $x_{r,b}$ is actually witnessed satisfied on the final step (the attribute is unmasked in room r), and hence is "proved" by a trivial, Strahler-1 proof. Let's suppose that this policy involves a μ-typical sequence of actions for the exploration policy Π^*, and first consider what Algorithm 1 does when invoked on Strahler number $s = 1$.

The algorithm first checks to see if the goal, $x_{r,b}$ can be proved satisfied in the initial environment configuration with probability 99% using `PAC-refute`. Let's suppose not. Then, since the algorithm is invoked on Strahler number $s = 1$, it skips the first loop that searches for literals to branch on, and moves directly to a search over actions. It partitions the examples according to the first action taken, invoking a recursive call along a branch taking each possible action that is empirically at least $\mu/2$-typical. Let us consider the recursive call on a branch that starts with the action, "move to room r'" that we know may lead to a successful policy. This branch consists of a literal asserting $a^{(1)}_{r'} = 1$, and the recursive call only receives the subset of traces in which this action was taken. Again, `PAC-refute` should determine that $x^{(2)}_{r,b}$ is not provable on these traces. Now, the algorithm searches over actions at $t = 2$; since we are only considering traces in which $a^{(1)}_{r'} = 1$, strictly fewer actions produce a sequence (following $a_{r'}$ at $t = 1$) that is empirically $\mu/2$-typical; perhaps few of these traces try to move to room r' again, or few traces try to move immediately back to room r. The algorithm then does not make a recursive call for those actions. Since we have supposed that the overall policy is μ-typical, the action "pick up ball b" in particular will still be considered, so we consider the recursive call on this branch that now asserts $a^{(1)}_{r'} = 1$ and $a^{(2)}_{b} = 1$, and only considers the traces satisfying these conditions. On this call, we should still not find that $x^{(3)}_{r,b}$ is provable, and so we will recursively consider the sequences that begin with $a^{(1)}_{r'}$ and $a^{(2)}_{b}$, and are empirically $\mu/2$-typical; a_r should be an option. Now, on this call, `PAC-refute` should find that $x^{(4)}_{r,b}$

is at least 98%-valid. Unfortunately, this is not high enough, so the algorithm continues; but, we know that on the branch $a^{(1)}_{r'}, a^{(2)}_{b}, a^{(3)}_{r}, a^{(4)}_{0}, a^{(5)}_{r'}, a^{(6)}_{b}, a^{(7)}_{r}$, then in more than 99% of the traces, $x^{(8)}_{r,b} = 1$, so this branch will be returned. Of course, there is no guarantee that this policy will be the one that is found. For example, if the sequence of actions that starts with dropping any balls, and then moving to room r, and then continuing with the above sequence is empirically $\mu/2$-typical under Π^* for some reason, then it may be that the algorithm returns a policy $a^{(1)}_{0}, a^{(2)}_{r}, a^{(3)}_{r'}, a^{(4)}_{b}, a^{(5)}_{r}, a^{(6)}_{0}, \ldots$.

It is also instructive to consider the behavior of Algorithm 1 in this example, when invoked with Strahler number $s = 2$. This may find a rather different policy than the one illustrated in Figure 3. As before, we suppose that $x^{(1)}_{r,b}$ is not provable with sufficiently high probability. The $s = 2$ case then searches for tests of a single literal such that Strahler-1 branches suffice to achieve the goal, given that the test is satisfied. Naturally, the goal itself, $x_{r,b}$ is such a test, the leaf of the branch on which $x^{(1)}_{r,b} = 1$ trivially satisfies the goal w.p. 1. So, the algorithm will generally construct such a branch when it considers the literal $x_{r,b}$, and continues recursively searching for a Straher-2 policy on the subtree where $x^{(1)}_{r,b} \in \{0, *\}$. But, on any other literal, we also know that there is a Strahler-1 policy that achieves the goal with probability greater than 99%. For such literals, depending on how many traces are consistent with the sequence of conditions on $x^{(1)}$ that have been imposed thus far, a branch featuring some appropriate Strahler-1 plan will be added. Relatedly, we may also find that some branches are added that use, e.g., the policy we discussed that only achieves the goal with probability 98%, as long as at least 1% of the traces that would end in failure do not reach that branch, so that overall only at most 1% of the traces actually fail on the branch. Finally, some conditions on $x^{(1)}$ may simply be satisfied by fewer than 1% of the traces. Since our algorithm tolerates branches with a 1% failure probability in general, the algorithm will also add such branches to the tree, terminating immediately with leaves.[4] In general, the algorithm continues until either (1) less than 1% of the traces remain consistent with all of the conditions on the traces imposed by the main, Strahler-2 branch, in which case the algorithm will consider the remaining leaf to be adequate or (2) the main, Strahler-2 branch adds a successful sequence of actions, allowing $x^{(t)}_{r,b}$ to be proved to hold with probability 99% (as in the Strahler-1 case described above). Actually, in this example, once the algorithm considers branches for each of the three possible observed settings of any single attribute, every example trace is consistent with one of these three branches, and hence less than 1% (actually, 0%) will remain consistent with the main branch. The algorithm will therefore terminate before moving on to consider a single action on the main, Strahler-2 branch. In any case, depending on the order in which the attributes are considered, the final decision tree policy that is returned may have up to $\sim n$ branches, each either containing some suitable conformant plan or reached by fewer than 1% of the traces (and may not achieve the goal). This possible inclusion of extra branches that are each allowed a 1% failure probability is the reason for the $\sim nT$ increase in the total failure probability in the guarantee provided by Theorem 10.

4. Intuitively, such branches that only lead to failure are undesirable. But, in general sometimes we may need to take a branch with a high conditional failure probability in order to capture a few traces where the policy succeeds.

3.2 Discussion

The complexity of Algorithm 1, as established by Theorem 10 is roughly what one would hope to achieve given the current state of knowledge in learning theory. Our task is very similar to supervised learning in the sense that we are using examples from a *fixed distribution* over trajectories generated by the reference policy Π^* to estimate the probability with which policies achieve the goal. Our sample complexity depends logarithmically on the number of possible branches (that is, linearly in their representation size), and quadratically on $\mu\gamma$. Note that a branch of a decision tree computes a conjunction, i.e., the AND of the literals that lead to that branch; the former quantity is the *VC-dimension* (Vapnik and Chervonenkis, 1971) of the class of conjunctions, which characterizes the number of examples needed to estimate their fit (Vapnik and Chervonenkis, 1971; Blumer et al., 1989; Ehrenfeucht et al., 1989). We are exploiting the fact that the VC-dimension actually characterizes the number of examples needed in order to obtain a *uniform bound* on the estimates of the error for *the entire class of conjunctions*—that is, the set of all possible branches, here. As we have seen (in the algorithm of Ehrenfeucht and Haussler, 1989), one only needs to know which branches have low error in order to identify a good tree. The latter, quadratic dependence on $\mu\gamma$ arises because we are trying to estimate the probability that a sequence of actions (that only is taken with probability μ) fails up to an accuracy of γ; since we are essentially conditioning on an event of probability $\sim \mu$, this means that we are essentially seeking to estimate the probability that the sequence of actions leads to failure up to an accuracy of $\mu\gamma$ in the original distribution. It is well-known that this latter task requires $\sim 1/(\mu\gamma)^2$ samples. In any case, we stress that if we had not separated out the task of exploration, a $\sim |A| \cdot 2^n$ or 2^T lower bound on the sample complexity holds for even very simple environments (Kakade, 2003, Section 2.5).

Likewise, the time complexity for finding a Strahler-s policy is comparable to the time taken by the algorithm of Ehrenfeucht and Haussler (1989) for finding decision tree classifiers, which remains the fastest known algorithm for this task. Our running time is slightly greater since we need to use theorem-proving techniques to determine whether or not a branch of our policies are suitable, and we test each branch individually. It is an interesting question whether or not, for example, some intermediate results can be cached across invocations of our theorem-proving algorithms, possibly leading to a reduced total running time.

The main quantitative downside of this algorithm is the potential $\sim (nT)^{s-1}$ increase of the failure probability of the policy found by the algorithm over the best-possible policy. Unfortunately, this is roughly in accordance with what we hope to achieve with the current state of the art. We again draw an analogy to supervised classification, in which even very simple tasks seem to lead to a similarly large increase in the error in the related "agnostic" noise model. The trouble is that the task of finding representations that optimize the error appears to be intractable under various plausible assumptions, even for very simple classes of representations such as halfspaces (Shalev-Shwartz et al., 2011) or even disjunctions (Kearns et al., 1994; Daniely et al., 2014; Daniely and Shalev-Shwartz, 2014). We still do not know how tight an approximation of the optimal error can be achieved, but an approximation factor that depends polynomially on the number of attributes reflects the current state of the art: consider for example, the relatively simple task of learning a disjunctive classifier (an

OR of the attributes). The best known algorithm for this task in the agnostic noise model suffers an increase of $\sim n^{1/3}$ in its error probability when there are n attributes (Awasthi et al., 2010).

Thus, in practice, this algorithm is only useful in situations when a highly fault-tolerant policy exists. As we noted in Section 2.3.2, such policies may exist, even if the environment is highly stochastic. Our algorithm is relatively tolerant of a long time horizon, and by repeating actions, it may be possible to decrease the failure probability of the overall plan exponentially with T (whereas the dependence of the overhead on T only grows linearly). This depends on repetitions of such actions being "typical" under Π^*, of course, but if Π^* is produced by (for example) solving other, smaller instances of planning problems, such sequences of repeated actions in a stochastic environment may well be typical.

4. Extension to Non-monotonic Reasoning

One weakness of the result of the previous section in learning STRIPS domains is that the frame axioms can only be learned under partial information when they are not needed: in order for them to be witnessed, the corresponding attributes of the POMDP must be observed. At the same time, we cannot supply generic frame axioms to the algorithm since the standard formulation of the frame axioms depends on the preconditions and effects of the actions, which are what we hope the agent to learn in each domain. Generic *non-monotonic* formulations of the frame axioms do exist, however: the clause $[\ell^{(t)} \wedge \sim(\neg\ell^{(t+1)})] \Rightarrow \ell^{(t+1)}$, where $\sim \ell$ roughly means "ℓ cannot be proved," nicely captures the frame axiom whenever we have learned the effects and preconditions of the actions. In this section, we will describe a semantics for $\sim\ell$ that is suitable for these purposes with respect to Strahler-s resolution and can be evaluated in polynomial time; we can then extend the algorithm of Theorem 8 to answer queries containing such literals, and thus we can add these generic frame axioms to the KB when invoking Algorithm 1.

We will use a variant of the *Well-Founded Semantics* (van Gelder et al., 1991) (WFS) for logic programs. More specifically, we will modify the iterated quotient definition of Przymusinski (1994) to obtain, for each s, an analogue of WFS for Strahler-s resolution. In particular, since Strahler-2 resolution corresponds precisely to chaining, our generalization for Strahler-2 resolution coincides with WFS over all (appropriate) directed versions of our original, undirected set of clauses.

Definition 12 (Quotient operator) *For any CNF φ over literals possibly using $\sim$ and any pair of sets of literals (L_+, L_-), the* quotient of φ modulo (L_+, L_-), *denoted $\varphi/(L_+, L_-)$, is the CNF obtained by substituting 1 for occurrences of $\sim\ell$ s.t. $\ell \in L_-$, substituting 0 for occurrences of $\sim\ell$ s.t. $\ell \in L_+$, substituting other occurrences of $\sim\ell$ with occurrences of a corresponding new variable $\tilde{\ell}$, and simplifying the resulting formula by deleting satisfied clauses and falsified literals.*

The intuition behind the quotient is that the sets L_+ and L_- consist of the literals that we know, respectively, can and can't be proved from φ. We will see how to obtain a pair (L_+, L_-) conservatively respecting this intuition for any formula.

Definition 13 (Least partial state) *The* Strahler-s least partial state *LPS_s of a CNF φ is a pair of sets of literals (L_+, L_-) (*not *over the $\tilde{\ell}$ variables) s.t. $\ell \in L_+$ iff ℓ has a Strahler-s proof from φ and $\ell \in L_-$ iff for the CNF φ' in which all occurrences of the $\tilde{\ell}$ variables have been deleted, there is no Strahler-s proof of ℓ from φ'.*

This definition is analogous to a construction of the (unique) "least partial models" for normal logic programs from Przymusinski (1991), in which the name has a more meaningful interpretation in terms of the partial models of logic programs. The name "least" partial state comes from considerations like the following. The pair of sets of literals (L_+, L_-) is a partial state in the sense that all literals $\ell \in L_+$ under a $\sim$ are set to 1, $\ell \in L_-$ under $\sim$ are set to 0, and otherwise $\sim\ell$ is set to 1/2. We suppose we are trying to (pointwise) minimize the truth values of the literals of φ while respecting the Strahler-s conclusions that can be derived from it, possibly given some further settings of the $\sim$ literals. That is, the existence of a Strahler-s derivation of ℓ requires that the ℓ be true in the partial state, and if for *some* substitution of the different occurrences of $\sim$ literals for truth values there exists a Strahler-s derivation of ℓ, then that prevents setting ℓ to false. Then we assign each literal the least possible value subject to these constraints.

We will finally obtain our "well-founded" Strahler-s semantics for negation-as-failure by taking the *least defined* fixed-point of the quotient and least partial state operators: the quotient incorporates the settings for $\sim$ literals that have been derived, and the least partial state operator obtains the consequences for Strahler-s provability. That is, it can be shown (moreover) that literals are only marked as "provable" (placed in L_+) or "unprovable" (placed in L_-) precisely when they are provable or unprovable, respectively, in *every* partial state that is a fixed-point under the composition of the quotient and least partial state operators. Much as in the original conception of the Well-Founded Semantics in van Gelder et al. (1991), this definition also conforms to the desirable intuition that literals are only marked as "provable" or "unprovable" if this can be ultimately derived from φ itself (with no additional knowledge or assumptions about what else is provable), ruling out consistent but circular negation-as-failure settings.

Proposition 14 (Semantics of Strahler-s NAF) *For any CNF φ over literals possibly using $\sim$, the sequence of pairs of sets of literals $(L_+^{(0)}, L_-^{(0)}) = (\varnothing, \varnothing)$, $(L_+^{(i+1)}, L_-^{(i+1)}) = LPS_s(\varphi/(L_+^{(i)}, L_-^{(i)}))$ converges to a pair of sets of literals (L_+^*, L_-^*) s.t. for every literal ℓ (without $\sim$),*

1. *if there is a Strahler-s proof of ℓ from $\varphi/(L_+^*, L_-^*)$, then $\ell \in L_+^*$*
2. *if $\ell \in L_-^*$ then there is no Strahler-s proof of ℓ from $\varphi/(L_+^*, L_-^*)$*

Furthermore, there is an algorithm that computes (L_+^*, L_-^*) given φ in time $O(|\varphi| n^{2s})$

Proof Convergence will follow from the observation that $L_+^{(i)} \subseteq L_+^{(i+1)}$ and $L_-^{(i)} \subseteq L_-^{(i+1)}$. For $i = 0$, the statement is trivial.

To see $L_+^{(i)} \subseteq L_+^{(i+1)}$ for $i \geq 1$, consider the proof of ℓ from $\varphi/(L_+^{(i-1)}, L_-^{(i-1)})$ witnessing $\ell \in L_+^{(i)}$. By induction on the structure of the proof of ℓ, we construct a new proof as follows: if this clause was obtained by a cut rule on one of the $\tilde{\ell}'$ variables and $\ell' \in L_+^{(i)} \cup L_-^{(i)}$, in which case, one of the clauses survives in $\varphi/(L_+^{(i)}, L_-^{(i)})$ with the occurrence of $\tilde{\ell}'$ eliminated, and hence the clause may be obtained by weakening the clause of $\varphi/(L_+^{(i)}, L_-^{(i)})$ from which $\tilde{\ell}'$ was eliminated; otherwise, by our induction hypothesis, we can derive subclauses of the two clauses involved in this final step from $\varphi/(L_+^{(i)}, L_-^{(i)})$. (The base case is trivial.)

As for $L_-^{(i)} \subseteq L_-^{(i+1)}$ for $i \geq 1$, we assume inductively that $L_+^{(i-1)} \subseteq L_+^{(i)}$ and $L_-^{(i-1)} \subseteq L_-^{(i)}$. We note that when $\ell \notin L_-^{(i+1)}$, there is a proof of ℓ from $(\varphi/(L_+^{(i)}, L_-^{(i)}))'$, which is constructed from φ ultimately by deleting clauses that either contain $\sim\ell'$ for $\ell' \in L_-^{(i)}$ or $\neg(\sim\ell')$ for $\ell' \in L_+^{(i)}$, and eliminating the rest of the $\tilde{\ell}'$ literals from the remaining clauses. Thus, $(\varphi/(L_+^{(i)}, L_-^{(i)}))'$ is a subformula of $(\varphi/(L_+^{(i-1)}, L_-^{(i-1)}))'$ since $L_+^{(i-1)} \subseteq L_+^{(i)}$ and $L_-^{(i-1)} \subseteq L_-^{(i)}$ by our inductive hypothesis. Therefore, the same proof establishes $\ell \notin L_-^{(i)}$, completing the inductive step.

Since we add at least one literal on each iteration until convergence, if we compute (L_+^*, L_-^*) using the iterative definition, the algorithm terminates in at most n iterations. Since each iteration can be computed by at most $2n$ applications of the Strahler-s proof search algorithm on a formula of size at most $|\varphi|$ (which runs in time $O(|\varphi| n^{2(s-1)})$, the running time bound follows.

For our fixed-point, we have a pair of sets of literals (L_+^*, L_-^*) such that $(L_+^*, L_-^*) = LPS_s(\varphi/(L_+^*, L_-^*))$, and hence the two claimed properties are immediate from the definition of the Strahler-s least partial state. ■

Naturally, the interpretation of φ is given by the classical propositional formula $\varphi/(L_+^*, L_-^*)$. In particular, we can now extend our PAC-Semantics to incorporate negation-as-failure like so:

Definition 15 *We say that a formula φ (possibly using $\sim$) is* $(1-\epsilon)$-valid *w.r.t. a distribution D and masking process M if, for m drawn from M, x drawn from D, and the formula ψ consisting of the conjunction of literals satisfied on $m(x)$, putting (L_+^*, L_-^*) equal to the fixed point obtained from $\varphi \wedge \psi$, the probability (over m and x) that x satisfies $\varphi/(L_+^*, L_-^*)$ is at least $1-\epsilon$.*

We can answer queries in this extended PAC-Semantics by an easy modification of the algorithm underlying Theorem 8 wherein, for each example, we first compute (L_+^*, L_-^*) for the KB Φ, and then for a query φ, check for a refutation of $\varphi \wedge (\Phi/(L_+^*, L_-^*))$. Such an algorithm simply replaces the existing subroutine in Algorithm 1. For $s \geq 2$, we can then verify that the generic frame axioms $[\ell^{(t)} \wedge \sim(\neg\ell^{(t+1)})] \Rightarrow \ell^{(t+1)}$ are $(1-\epsilon-\nu)$-valid under the Strahler-s NAF semantics in a noisy STRIPS instance with noise rate ν and $(1-\epsilon)$-testable actions. We are therefore free to include them in the KB at a small cost in validity. Crucially, it may also be verified that they allow the values of hidden state attributes to be propagated forward, at least when they are the only clauses in the KB containing the state attributes.

5. Relationship to Other Work

Our ability to learn descriptions of the dynamics that are simple but imperfect already distinguishes our work from some others that attempt to learn explicit descriptions of an environment's rules from examples, e.g., (Otero, 2005; Amir and Chang, 2008). Others attempt to construct a model that accounts for the imperfection (García-Martínez and Borrajo, 2000; Schmill et al., 2000; Pasula et al., 2007; Yoon and Kambhampati, 2007;

Lang and Toussaint, 2009; Mourão et al., 2012), but cannot provide a formal analysis. This lack of theoretical grounding seems inherent due to the difficulties posed by agnostic learning; moreover, the negative results of Michael (2011) speak to the advantages of attempting to answer queries without producing an explicit action model. Work on applying logic programming to learning for planning (Thon et al., 2009) features algorithms with theoretical grounding for several tasks, but similarly encounters the structure learning problem when learning rule sets, and indicates that it (together with dealing with hidden information) remains a frontier problem for that approach. Work on Predictive State Representations by Boots et al. (2011) only establishes consistency, not fast convergence under any clear conditions—some kind of assumption on the condition number of the matrices seems essential, but neither the details nor the significance of this requirement seem to be understood.

The use of a limited class of policies to approximate optimal behavior has a long history in the reinforcement learning literature, see Sutton and Barto (1998, Chapter 8) for a review; in particular, techniques that used neural nets or other kinds of linear approximators to choose a good action are examples of such techniques. Chapman and Kaelbling (1991) seem to be the first to limit their policies to decision trees. The Utile Suffix Memory of McCallum (1995) similarly builds a coarsened approximation of the state space itself by classifying histories in partial information environments. Both of these works viewed the decision trees as dynamically refined estimates of the environments in which (approximately) optimal actions may be selected directly, rather than simply as policy representations to optimize; this latter view emerged only somewhat later (Kearns et al., 2002; Meuleau et al., 1999). We note that McCallum (1995) did not consider factored state and observation spaces, and so Utile Suffix Memory differs from our decision trees in that it does not branch along the propositional factors of the observations as we use here; this was later considered in the U-Tree algorithm of McCallum (1996). In any case, both Utile Suffix Memory and U-Tree are still viewed as classifying a partial trace, indexed by offsets from the current observation (e.g., branching on the observation from t steps prior) by a choice of action. Conceptually, this is more like a reactive policy, but applied to a longer history; by contrast, "states" of our decision tree policies are captured by a node of the tree, so the behavior of the policy is "non-uniform" with respect to time. The representation is still thus not equivalent once complexity measures such as the size come into consideration; our objective is also different, in that we merely seek for our trees to reach a goal state, whereas McCallum's policies are in the usual utility-maximization framework in which one seeks a high discounted utility over time. The algorithm we use thus also ends up being rather different from those proposed by McCallum. Finally, McCallum focuses strictly on empirical evaluation of his strategies, whereas our focus is strictly on a theoretical evaluation.

Our setting is related to a setting considered previously by Khardon (1999), apprenticeship learning (Syed and Schapire, 2010) and reductions to classification (Langford and Zadrozny, 2005): These works aim to learn a policy using example traces, even for sophisticated policy representations. The distinction is that we don't assume that Π^* is our desired policy, just that there exists a good policy that Π^* agrees with at least a μ-fraction of the time. (If the number of actions is small, Π^* could even be a random walk.)

Reinforcement learning techniques such as Kearns et al. (2002) can learn from a poor initial policy such as a random walk. But, these works often face the exploration problem by using repeated experience with the environment model, in particular with an explicitly

provided reward function – i.e., goal – at hand to guide their search for a policy. Largely as a consequence of the inherent difficulties of exploration in general environments, the theoretical analyses provided by such works exhibit an undesirable dependence on either the size of the state space or, in the particular case of Kearns et al., on the time horizon. (We note that a discount factor of $1 - 1/T$ is roughly equivalent to a time horizon of T.) Along somewhat similar lines, work by Fern et al. (2006) and Lazaric et al. (2010) consider learning from a poor initial policy in a full-information model by using a reduction to classification; the full-information setting is quite different in that the history of observations is no longer relevant (and so there is no need to consider a stateful policy like our decision trees). Theoretically, Fern et al. showed how to solve the MDP if the best action at each state is substantially better than the second best, and if the resulting deterministic optimal policy for a fixed horizon is expressed by an efficiently learnable class. Naturally, these assumptions may easily be violated, in particular if the actions in a plan can be reordered. Lazaric et al. extended the analysis of Fern et al., but assume that a policy minimizing the (nonzero, in general) loss can be found somehow. It is not clear what kind of policies Lazaric et al. have in mind, but in Boolean classification, this is essentially the problem of agnostic learning (Kearns et al., 1994); recent evidence suggests that this is intractable for all but the simplest kinds of classifiers (Daniely et al., 2014; Daniely and Shalev-Shwartz, 2014).[5]

Work by Walsh (2010) touches on all of the above areas; he constructs complete relational domain models with a full theoretical analysis, but under the assumption of constant arity expressions and numbers of effects (and/or in an apprenticeship learning setup). The assumptions of bounded arity mean that these relational expressions can be expressed by moderate size propositional representations, so the main trade-off between Walsh's work and ours is that Walsh produces explicit rules unlike us, but at the cost of exponential scaling when these arities and numbers of possible effects is large.

Learning from exercises (Natarajan, 1989; Tadepalli, 2008) is similar, except that the learner is provided sequences of examples that gradually increase in difficulty. The difference with our work is that we don't assume that the current goal can be reduced to subproblems of "lower difficulty" (but nor do we guarantee success when this is the case). Later, Joshi et al. (2010) took a similar approach in which example policies (generated either by simple strategies or by other planning algorithms) were used to obtain a fast model-checking policy search algorithm for relational planning in a fully observed setting. Their setting was thus quite different from ours—we are seeking to cope with missing information, as opposed to gaining a speed-up in a full-information setting.

We noted that the use of importance sampling to enable off-policy learning in POMDPs was first considered by Precup et al. (2000, 2001), building on earlier work on off-policy learning in MDPs (Sutton and Barto, 1998, Chapter 5). Precup et al. used a second-moment bound in their analysis of importance sampling since in general, the likelihood ratios may be large. Indeed, the issue that generally plagues importance sampling is that this variance may be high (Shelton, 2001; Peshkin and Shelton, 2002), and a variety of techniques (e.g., Hachiya et al., 2009, 2011) have been proposed to control the variance in more general settings. This problem actually does not arise in our setting, as a consequence

5. And, *approximate* agnostic learning generally incurs an increase in the loss of a similar magnitude as we obtain in Theorem 10—cf., the best known polynomial time algorithm for learning disjunctive classifiers on n attributes increases the loss by a factor of $n^{1/3}$ (Awasthi et al., 2010).

of our exclusive focus on μ-typical sequences of actions under the sampling distribution, which thus translates into an absolute $1/\mu$ bound on the likelihood ratio. We note that some other works (Uchibe and Doya, 2004; Wawrzyński, 2009) have simply capped the likelihood ratios in an importance sampling computation; here, we achieve such a bound by *disregarding* policies that have a large likelihood ratio, instead of altering it. Our analysis of importance sampling then follows a standard learning theory paradigm, most similar to that of Peshkin and Mukherjee (2001), except that the bound on the likelihood ratios allows us to get away with the use of Hoeffding's inequality rather than Bernstein's inequality (which considers the variance).[6]

6. Future Directions

One natural direction concerns improving Theorem 10: One of the main insights underlying Theorem 8 is that the main barrier to agnostic learning is finding an explicit representation, and yet Algorithm 1 proceeds by constructing an explicit policy. It is conceivable that the $O((nT)^{s-1})$-factor blow-up in the policy failure probability ϵ could be eliminated if we could similarly identify a good action on-line, without going so far as to construct an entire policy.

Our set-up of learning policies that take typical actions with respect to a reference policy naturally suggests a bootstrapping approach to learning complex policies; another immediate direction is then to investigate what can be *proved* learnable by bootstrapping (in contrast to the empirical work of Fern et al., 2006). In a related direction, Ross and Bagnell (2012) essentially showed that a penalty for atypicality similar to what our algorithm suffers can be eliminated in the standard (discounted, fixed cost-function, full-information) MDP setting by iteratively constructing a policy, collecting new data, retraining the policy, and repeating. In this way, the training data is shaped so that the actions of the policy become highly "typical" and well-estimated. It is likely that a similar strategy will also work in our partial information setting, although it may rely on the goal remaining fixed during this process.

More generally, our work side-steps the entire problem of exploration of POMDPs; it can be viewed as finding a policy, given that the POMDP has been sufficiently well explored. So, one might try to address the exploration problem in the context of such planning algorithms by showing that a class of environments can be efficiently explored sufficiently well to permit good policies to be found.

Acknowledgments

I am grateful to Leslie Kaelbling for numerous detailed suggestions and criticisms of this work that improved it immensely. I likewise thank Mithun Chakraborty and my anonymous reviewers for their detailed comments and suggestions. This work was also heavily influenced by conversations with Leslie Valiant.

6. For off-policy evaluation in practice, it may be vastly preferable to apply a Bernstein-like inequality using an empirical estimate of the variance instead of Hoeffding's inequality (Thomas et al., 2015). I thank a reviewer for bringing this to my attention.

References

Eyal Amir and Allen Chang. Learning partially observable deterministic action models. *JAIR*, 33:349–402, 2008.

Carlos Ansótegui, María Luisa Bonet, Jordi Levy, and Felip Manyá. Measuring the hardness of SAT instances. In *Proc. 23rd AAAI*, pages 222–228, 2008.

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Improved guarantees for agnostic learning of disjunctions. In *Proc. 23rd COLT*, 2010.

J. Andrew Bagnell, Sham Kakade, Andrew Ng, and Jeff Schneider. Policy search by dynamic programming. In S. Thrun, L. K. Saul, and B. Schőlkopf, editors, *Advances in Neural Information Processing Systems 16 (NIPS 2003)*, pages 831–838. MIT Press, Cambridge, MA, 2004.

Paul Beame, Henry Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *JAIR*, 22:319–351, 2004.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, 1989.

Byron Boots, Sajid M. Siddiqi, and Geoffrey J. Gordon. Closing the learning-planning loop with predictive state representations. *Int. J. Robotics Res.*, 30(7):954–966, 2011.

Randy E. Bryant. Symbolic Boolean manipulation with ordered binary decision diagrams. *ACM Computing Surveys*, 24(3):293–218, 1992.

Tom Bylander. The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69:165–204, 1994.

David Chapman and Leslie Pack Kaelbling. Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In *Proc. 12th IJCAI*, pages 726–731, 1991.

Stephen A. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd STOC*, pages 151–158, 1971.

Amit Daniely and Shai Shalev-Shwartz. Complexity theoretic limtations on learning DNF's. To appear in *29th COLT*, 2016. Preprint version: arXiv:1404.3378

Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. From average case complexity to improper learning complexity. In *Proc. 46th STOC*, pages 441–448, 2014.

Andrzej Ehrenfeucht and David Haussler. Learning decision trees from random examples. *Inf. Comp.*, 82(3):231–246, 1989.

Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Inf. Comp.*, 82:247–261, 1989.

Kutluhan Erol, Dana S. Nau, and V. S. Subrahmanian. Complexity, decidability and undecidability results for domain-independent planning. *Artificial Intelligence*, 76(1–2):75–88, 1995.

Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Inf. Comp.*, 171(1): 84–97, 2001.

Alan Fern, Sungyook Yoon, and Robert Givan. Approximate policy iteration with a policy language bias: solving relational Markov decision processes. *JAIR*, 25:85–118, 2006.

Richard E. Fikes and Nils J. Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3–4):189–208, 1971.

Ramón García-Martínez and Daniel Borrajo. An integrated approach of learning, planning, and execution. *J. Intelligent and Robotic Sys.*, 29(1):47–78, 2000.

Robert P. Goldman and Mark S. Boddy. Expressive planning and explicit knowledge. In *Proc. 3rd AIPS*, pages 110–117, 1996.

Oded Goldreich, Brendan Juba, and Madhu Sudan. A theory of goal-oriented communication. *J. ACM*, 59(2):8:1–8:65, 2012.

Hirotaka Hachiya, Takayuki Akiyama, Masashi Sugiyama, and Jan Peters. Adaptive importance sampling for value function approximation in off-policy reinforcement learning. *Neural Networks*, 22(10):1399–1410, 2009.

Hirotaka Hachiya, Jan Peters, and Masashi Sugiyama. Reward-weighted regression with sample reuse for direct policy search in reinforcement learning. *Neural Networks*, 23(11): 2798–2832, 2011.

David Haussler. Learning conjunctive concepts in structural domains. *Mach. Learn.*, 4(1): 7–40, 1989.

Rune M. Jensen, Manuela M. Veloso, and Randal E. Bryant. Fault tolerant planning: Toward probabilistic uncertainty models in symbolic non-deterministic planning. In *Proc. 14th ICAPS*, pages 335–344, 2004.

Saket Joshi, Kristian Kersting, and Roni Khardon. Self-taught decision theoretic planning with first order decision diagrams. In *Proc. 20th ICAPS*, pages 89–96, 2010.

Brendan Juba. *Universal Semantic Communication*. Springer, Berlin, 2011.

Brendan Juba. Implicit learning of common sense for reasoning. In *Proc. 23rd IJCAI*, pages 939–946, 2013.

Sham M. Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, University College London, 2003.

Sham M. Kakade and John Langford. Approximately optimal reinforcement learning. In *Proc. 19th ICML*, pages 267–274, 2002.

Henry Kautz and Bart Selman. Planning as satisfiability. In *Proc. 10th ECAI*, pages 359–363, 1992.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *J. ACM*, 41:67–95, 1994.

Michael Kearns, Yishay Mansour, and Andrew Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Mach. Learn.*, 49(2):193–208, 2002.

Michael J. Kearns, Robert E. Schapire, and Linda M. Sellie. Towards efficient agnostic learning. *Mach. Learn.*, 17(2-3):115–141, 1994.

Roni Khardon. Learning to take actions. *Mach. Learn.*, 35(1):57–90, 1999.

Roni Khardon and Dan Roth. Learning to reason. *J. ACM*, 44(5):697–725, 1997.

Oliver Kullmann. Investigating a general hierarchy of polynomially decidable classes of CNF's based on short tree-like resolution proofs. Technical Report TR99-041, ECCC, 1999.

Tobias Lang and Marc Toussaint. Approximate inference for planning in stochastic relational worlds. In *Proc. 26th ICML*, pages 585–592, 2009.

John Langford and Bianca Zadrozny. Relating reinforcement learning performance to classification performance. In *Proc. 22nd ICML*, pages 473–480, 2005.

Alessandro Lazaric, Mohammad Ghavamzadeh, and Rémi Munos. Analysis of classification-based policy iteration algorithms. In *Proc. 27th ICML*, 2010.

Yishay Mansour. Reinforcement learning and mistake bounded algorithms. In *Proc. 12th COLT*, pages 183–192, 1999.

Andrew Kachites McCallum. Learning to use selective attention and short-term memory in sequential tasks. In *From animals to animals 4: proceedings of the fourth international conference on simulation of adaptive behavior*, pages 315–325. MIT Press, Cambridge, MA, 1996.

R. Andrew McCallum. Instance-based utile distinctions for reinforcement learning with hidden state. In *Proc. 12th ICML*, pages 387–395, 1995.

John McCarthy. Programs with common sense. In *Teddington Conf. on the Mechanization of Thought Processes*, pages 756–791, 1959. Available at http://www-formal.stanford.edu/jmc/mcc59.html.

Nicolas Meuleau, Kee-Eung Kim, Leslie Pack Kaelbling, and Anthony R. Cassandra. Solving POMDPs by searching the space of finite policies. In *Proc. 15th UAI*, pages 417–426, 1999.

Loizos Michael. Partial observability and learnability. *Artificial Intelligence*, 174(11):639–669, 2010.

Loizos Michael. Causal learnability. In *Proc. 22nd IJCAI*, pages 1014–1020, 2011.

Loizos Michael and Leslie G. Valiant. A first experimental demonstration of massive knowledge infusion. In *Proc. 11th KR*, pages 378–389, 2008.

Kira Mourão, Luke Zettlemoyer, Ronald P. A. Petrick, and Mark Steedman. Learning STRIPS operators from noisy and incomplete observations. In *Proc. 28th UAI*, pages 614–623, 2012.

Balas K. Natarajan. On learning from exercises. In *Proc. 2nd COLT*, pages 72–87, 1989.

Ramon P. Otero. Induction of the indirect effects of actions by monotonic methods. In *Proc. ILP 2005*, volume 3625 of *LNAI*, pages 279–294. Springer, 2005.

Hanna M. Pasula, Luke S. Zettlemoyer, and Leslie Pack Kaelbling. Learning symbolic models of stochastic domains. *JAIR*, 29:309–352, 2007.

Leonid Peshkin and Sayan Mukherjee. Bounds on sample size for policy evaluation in Markov environments. In *Proc. COLT/EuroCOLT 2001*, volume 2111 of *LNAI*, pages 616–629. Springer, 2001.

Leonid Peshkin and Christopher R. Shelton. Learning from scarce experience. In *Proc. 19th ICML*, pages 498–505, 2002.

Doina Precup, Richard S. Sutton, and Sanjoy Dasgupta. Eligibility traces for off-policy policy evaluation. In *Proc. 17th ICML*, pages 759–766, 2000.

Doina Precup, Richard S. Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *Proc. 18th ICML*, pages 417–424, 2001.

Teodor C. Przymusinski. Stable semantics for disjunctive programs. *New Generation Comput.*, 9:401–424, 1991.

Teodor C. Przymusinski. Well-founded and stationary models of logic programs. *Ann. Mathematics and Artificial Intelligence*, 12(3):141–187, 1994.

Stéphane Ross and J. Andrew Bagnell. Agnostic system identification for model-based reinforcement learning. In *Proc. 29th ICML*, pages 1703–1710, 2012.

Dan Roth. Learning to reason: the non-monotonic case. In *Proc. 14th IJCAI*, volume 2, pages 1178–1184, 1995.

Matthew D. Schmill, Tim Oates, and Paul R. Cohen. Learning planning operators in real-world, partially observable environments. In *Proc. 5th AIPS*, pages 246–253, 2000.

Shai Shalev-Shwartz, Ohad Shamir, and Karthik Sridharan. Learning kernel based half-spaces with the 0-1 loss. *SIAM J. Comput.*, 40(6):1623–1646, 2011.

Christopher Shelton. Policy improvement for POMDPs using normalized importance sampling. In *Proc. 17th UAI*, pages 496–503, 2001.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning*. MIT Press, Cambridge, MA, 1998.

Umar Syed and Robert E. Schapire. A reduction from apprenticeship learning to classification. In *Proc. 23rd NIPS*, pages 2253–2261, 2010.

Prasad Tadepalli. Learning to solve problems from exercises. *Computational Intelligence*, 24(4):257–291, 2008.

Philip S. Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence off-policy evaluation. In *Proc. 29th AAAI*, pages 3000–3006, 2015.

Ingo Thon, Bernd Gutmann, Martijn van Otterlo, Niels Landwehr, and Luc De Raedt. From non-deterministic to probabilistic planning with the help of statistical relational learning. In *ICAPS 2009 - Proc. Workshop on Planning and Learning*, pages 22–30, 2009.

Eiji Uchibe and Kenji Doya. Competitive-cooperative-concurrent reinforcement learning with importance sampling. In *Proc. Intl. Conf. Simulation of Adaptive Behavior*, pages 287–296, 2004.

Leslie G. Valiant. Rationality. In *Proc. 8th COLT*, pages 3–14, 1995.

Leslie G. Valiant. Robust logics. *Artificial Intelligence*, 117:231–253, 2000a.

Leslie G. Valiant. A neuroidal architecture for cognitive computation. *J. ACM*, 47(5): 854–882, 2000b.

Leslie G. Valiant. Knowledge infusion. In *Proc. 21st AAAI*, pages 1546–1551, 2006.

Allen van Gelder, Kenneth A. Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *J. ACM*, 38(3):620–650, 1991.

Vladimr Vapnik and Alexei Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2): 264–280, 1971.

Thomas J. Walsh. *Efficient Learning of Relational Models for Sequential Decision Making*. PhD thesis, Rutgers University, 2010.

Paweł Wawrzyński. Real-time reinforcement learning by sequential actor-critics and experience replay. *Neural Networks*, 22:1484–1497, 2009.

Sungyook Yoon and Subbarao Kambhampati. Towards model-lite planning: A proposal for learning & planning with incomplete domain models. In *Proc. ICAPS Workshop on AI Planning and Learning*, 2007.

Håkan L. S. Younes and Michael L. Littman. PPDDL1.0: An extension to PDDL for expressing planning domains with probabilistic effects. Technical Report CMU-CS-04-167, Carnegie Mellon University, 2004.

Journal of Machine Learning Research 17 (2016) 1-37 Submitted 5/14; Revised 12/15; Published 7/16

Cells in Multidimensional Recurrent Neural Networks

Gundram Leifert gundram.leifert@uni-rostock.de
Tobias Strauß tobias.strauss@uni-rostock.de
Tobias Grüning tobias.gruening@uni-rostock.de
Welf Wustlich welf.wustlich@uni-rostock.de
Roger Labahn roger.labahn@uni-rostock.de
University of Rostock
Institute of Mathematics
18051 Rostock, Germany

Editor: Yoshua Bengio

Keywords: LSTM, MDRNN, CTC, handwriting recognition, neural network

Abstract

The transcription of handwritten text on images is one task in machine learning and one solution to solve it is using multi-dimensional recurrent neural networks (MDRNN) with connectionist temporal classification (CTC). The RNNs can contain special units, the long short-term memory (LSTM) cells. They are able to learn long term dependencies but they get unstable when the dimension is chosen greater than one. We defined some useful and necessary properties for the one-dimensional LSTM cell and extend them in the multi-dimensional case. Thereby we introduce several new cells with better stability. We present a method to design cells using the theory of linear shift invariant systems. The new cells are compared to the LSTM cell on the IFN/ENIT and Rimes database, where we can improve the recognition rate compared to the LSTM cell. So each application where the LSTM cells in MDRNNs are used could be improved by substituting them by the new developed cells.

1. Introduction

Since the last decade, artificial neural networks (NN) became state-of-the-art in many fields of machine learning, for example they can be applied to pattern recognition. Typical NN are feedforward NN (FFNN) or recurrent NN (RNN), whereas the latter contain recurrent connections. When nearby inputs depend on each other, providing these inputs as additional information to the NN can improve its recognition result. FFNNs obtain these dependencies by making this nearby inputs accessible. If RNNs are used, the recurrent connections can be used to learn if the surrounding input is relevant, but these connections result in a vanishing dependency over time. In S. Hochreiter, J. Schmidhuber (1997) the authors develop the long short-term memory (LSTM) which is able to have a long term dependency. This LSTM is extended in A. Graves, S. Fernandez andJ. Schmidhuber (2007) to the multi-dimensional (MD) case and is used in a hierarchical multi-dimensional RNN (MDRNN) which performed best in three competitions at the International Conference on Document Analysis and Recognition (ICDAR) in 2009 without any feature extraction and knowledge of the recognized language model.
In this paper we analyse these MD LSTM regarding the ability to provide long term dependencies in MDRNNs and show that it can easily have an unwanted growing dependency for higher dimensions. We define a more general description of an LSTM—a cell—and change the LSTM architecture which leads to new MD cell types, which also can provide long term dependencies. In two experiments we show that substituting the LSTM in MDRNNs by these cells works well. Due to this we assume that substituting the LSTM cell by the best performing cell, the *LeakyLP cell*, will improve the performance of an MDRNN also in other scenarios. Furthermore the new cell types could also be used for the one-dimensional (1D) case, so using them in a bidirectional RNN with LSTMs (BLSTM) could lead to better recognition rates.
In Section 2 we introduce the reader to the development of the LSTM cells (S. Hochreiter, J. Schmidhuber, 1997) and its extension (F. A. Gers, J. Schmidhuber andF. Cummins, 1999). Based on that in Section 3 we define two properties that probably lead to the good performance of the 1D LSTM cells. Both together guarantee that the cell can have a long term dependency. A third property ensures that gradient cannot explode over time. In Section 4 we show that the MD version of the LSTM is still able to provide long term dependency whereas the gradient can explode easily for dimension greater than 1. In Section 5 we change the architecture of the MD LSTM cell and reduce it to the 1D LSTM cell so that the cell fulfills the two properties for any dimension. Nevertheless the internal cell state can linearly grow over time. This problem is solved in Section 6 using a trainable convex combination of the input and the previous internal cell states. The new cell type can provide long term dependencies and does not suffer from exploding gradients. Motivated by the last sections we introduce a more general way to define MD cells in Section 7. Using the theory of linear shift-invariant systems and their frequency analysis we are able to get a new interpretation of the cells and we create 5 new cell types. To test the performance of the cells in Section 8 we take two data sets from the ICDAR 2009 competitions, where the MDRNNs with LSTM cell won. On these data sets we compare the recognition results of the MDRNNs when we substitute the LSTM cells by the new developed cells. On both data sets, the IFN/ENIT data set and the RIMES data set we can improve the recognition rate using the new developed cells.

2. Previous Work

In this section we briefly want to introduce a recurrent neural network (RNN) and the development of the LSTM cell. In previous literature there are various notation to describe the update equations of RNNs an LSTMs. To unify the notations we will refer to their notation using "$\triangleq$" (F. A. Gers, J. Schmidhuber andF. Cummins, 1999; S. Hochreiter, J. Schmidhuber, 1997; A. Graves andJ. Schmidhuber, 2008). Therefore we concentrate on a simple hierarchical RNN with one input layer with the set of neurons I, one recurrent hidden layer with the set of neurons H and one output layer with the set of neurons O. For each time step $t \in \mathbb{N}$ the layers are updated asynchronously in the order I, H, O. In one specific layer all neurons can be updated synchronously. In the hidden layer for one neuron $c \in H$

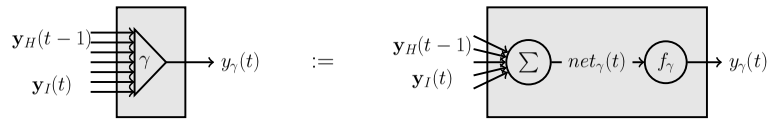


Figure 1: Schematic diagram of a unit: The unit $\gamma \in H$ is a simple neuron with the network's feed forward input $\mathbf{y}_I(t) = \left(y_i(t)\right)_{i\in I}$ and recurrent input $\mathbf{y}_H(t-1) = \left(y_h(t-1)\right)_{h\in H}$ and an output activation $y_\gamma(t)$. *Right:* A unit has an input activation $net_\gamma(t)$, which is a linear combination of the source activations $\mathbf{y}_I(t), \mathbf{y}_H(t-1)$. The output activation $y_\gamma(t)$ is computed by applying the activation function f_γ to the input activation. *Left:* The short notation of a unit.

at time $t \in \mathbb{N}$ we calculate the neuron's *input activation* net_c by

$$\left(a_c(t) \triangleq\right) \quad net_c(t) = \sum_{i\in I} w_{c,i} y_i(t) + \sum_{h\in H} w_{c,h} y_h(t-1). \tag{1}$$

with weights $w_{[\text{target neuron}],[\text{source neuron}]}$. A bias in (1) can be added by extending the set $I := I \cup \{bias\}$ with $y_{bias}(t) = 1 \forall t \in \mathbb{N}$ and hence we will not write the bias in the equations, but we use them in our RNNs in Section 8. The neuron's *output activation* is calculated by

$$\left(y^c(t), b_c(t) \triangleq\right) \quad y_c(t) = f_c\left(net_c(t)\right)$$

with a differentiable sigmoid *activation function* f_c. To make (1) suitable for $t \leq 0$ we define $\forall h \in H, \forall t \in \mathbb{Z} \setminus \mathbb{N} : y_h(t) = 0$. This simple neuron with a linear function of activations as input and one activation function we call *unit* (compare to Figure 1). In (1) the activation of the unit is dependent on the current activations of the layer below and the previous activations of the units from the same layer. When there are no recurrent connections ($\forall c, h \in H : w_{c,h} = 0$), the layer is called feed-forward layer, otherwise recurrent layer.

2.1 The Long Short-Term Memory

A *standard LSTM* cell c has one input with an input activation $y_{c_{in}}(t)$ a set of *gates*, one *internal state* s_c and one output(-activation) y_c ($\triangleq y^c$). The gates are also units and their task is to learn whether a signal should pass the gate or not. They almost always have the logistic activation function $f_{log}(x) := \frac{1}{1+\exp(-x)}$ ($\triangleq f_1(x)$). The input of the standard LSTM cell is calculated from a unit with an odd activation function with a slope of 1 at $x = 0$. We use $f_c(x) = \tanh(x)$ in this paper, another solution could be $f_c(x) = 2\tanh\left(\frac{x}{2}\right)$ (see S. Hochreiter, J. Schmidhuber, 1997). The standard LSTM has two gates: The *input gate* (IG or ι) and the *output gate* (OG or ω). These both gates are calculated like a unit, so that

$$net_\iota(t) = \sum_{i\in I} w_{\iota,i} y_i(t) + \sum_{h\in H} w_{\iota,h} y_h(t-1)$$
$$\left(y^{in_c}(t), b_\iota(t) \triangleq\right) \quad y_\iota(t) = f_{log}\left(net_\iota(t)\right)$$

and

$$net_\omega(t) = \sum_{i\in I} w_{\omega,i} y_i(t) + \sum_{h\in H} w_{\omega,h} y_h(t-1)$$
$$\left(y^{out_c}(t), b_\omega(t) \triangleq\right) \quad y_\omega(t) = f_{log}\left(net_\omega(t)\right).$$

The input of an LSTM is defined like in (1) by

$$\left(net_c(t) \triangleq\right) \quad net_{c_{in}}(t) = \sum_{i\in I} w_{c,i} y_i(t) + \sum_{h\in H} w_{c,h} y_h(t-1),$$
$$\left(g\left(net_c(t)\right), f_2\left(net_c(t)\right) \triangleq\right) \quad y_{c_{in}}(t) = f_c\left(net_{c_{in}}(t)\right).$$

The internal state $s_c(t)$ is calculated by

$$s_c(t) = y_{c_{in}}(t) \cdot y_\iota(t) + s_c(t-1), \tag{2}$$

the output activation $y_c(t)$ of the LSTM is calculated from

$$\left(y^c(t), b_c(t) \triangleq\right) \quad y_c(t) = h_c\left(s_c(t)\right) \cdot y_\omega(t) \tag{3}$$

with $h_c(x) := \tanh(x)$ ($\triangleq f_3(x)$). The LSTM can be interpreted as a kind of memory module where the internal state stores the information. For a given input $y_{c_{in}}(t) \in (-1, 1)$ the IG "decides" if the new input is relevant for the internal state. If so, the input is added to the internal state. The information of the input is now saved in the activation of the internal state. The OG determines whether or not the internal activation should be displayed to the rest of the network. So the information, stored in the LSTM is just "readable" when the OG is active. To sum up, an open IG can be seen as a "write"-operation into the memory and an open OG as a "read"-operation of the memory.

Another way to understand the LSTM is to take a look at the gradient propagated through it. To analyse the LSTM properly, we have to ignore gradients comming from recurrent weights. We define the *truncated gradient* similar to S. Hochreiter, J. Schmidhuber (1997) and F. A. Gers, J. Schmidhuber andF. Cummins (1999).

Definition 1 (truncated gradient) *Let $\gamma \in \{c_{in}, \iota, \omega\}$ be any input or gate unit and $y_c(t-1)$ any previous output activation. The* truncated gradient *differs from the exact gradient only by setting recurrent weighted gradient propagation $\frac{\partial net_\gamma(t)}{\partial y_c(t-1)}$ to zero. We write*

$$\frac{\partial net_\gamma(t)}{\partial y_c(t-1)}\Big(= w_{\gamma,c}\Big) \underset{tr}{=} 0.$$

Now, let E be an arbitrary error which is used to train the RNN and $\frac{\partial E(t)}{\partial y_c(t)}$ the resulting derivative at the output of the LSTM. The OG can eliminate the gradient coming from the output, because

$$\frac{\partial y_c(t)}{\partial s_c(t)} = \underbrace{h_c'\left(s_c(t)\right)}_{\in(0,1]} \cdot \underbrace{y_\omega(t)}_{\in(0,1)},$$

so the OG decides when the gradient should go into the internal state. Especially for $|s_c(t)| \ll 1$ we get

$$\frac{\partial y_c(t)}{\partial s_c(t)} \approx y_\omega(t).$$

The key idea of the LSTMs is that an error that occurs at the internal state neither explode nor vanish over time. Therefore, we take a look at the partial derivative $\frac{\partial s_c(t)}{\partial s_c(t-1)}$, which is also known as error carousel (for more details see S. Hochreiter, J. Schmidhuber, 1997). Using the truncated gradient of Definition 1 for this derivative, we get

$$\begin{aligned}
\frac{\partial s_c(t)}{\partial s_c(t-1)} =& y_{c_{in}}(t) \cdot \frac{\partial y_\iota(t)}{\partial s_c(t-1)} + y_\iota(t) \cdot \frac{\partial y_{c_{in}}(t)}{\partial s_c(t-1)} + 1 \\
=& y_{c_{in}}(t) \cdot \frac{\partial y_\iota(t)}{\partial y_c(t-1)} \frac{\partial y_c(t-1)}{\partial s_c(t-1)} + y_\iota(t) \cdot \frac{\partial y_{c_{in}}(t)}{\partial y_c(t-1)} \frac{\partial y_c(t-1)}{\partial s_c(t-1)} + 1 \\
=& y_{c_{in}}(t) \cdot \frac{\partial y_\iota(t)}{\partial net_\iota(t)} \underbrace{\frac{\partial net_\iota(t)}{\partial y_c(t-1)}}_{\underset{tr}{=}0} \frac{\partial y_c(t-1)}{\partial s_c(t-1)} \\
& + y_\iota(t) \cdot \frac{\partial y_{c_{in}}(t)}{\partial net_{c_{in}}(t)} \underbrace{\frac{\partial net_{c_{in}}(t)}{\partial y_c(t-1)}}_{\underset{tr}{=}0} \frac{\partial y_c(t-1)}{\partial s_c(t-1)} + 1 \\
\Rightarrow \frac{\partial s_c(t)}{\partial s_c(t-1)} \underset{tr}{=}& 1. \qquad (4)
\end{aligned}$$

So, once having a gradient at the internal state we can use the chain rule and get $\forall \tau \in \mathbb{N}$: $\frac{\partial s_c(t)}{\partial s_c(t-\tau)} \underset{tr}{=} 1$. This is called *constant error carousel*.
Like the OG can eliminate the gradient coming from the LSTM output, the IG can do the same with the gradient coming from the internal state, that means it decides when the gradient should be injected to the source activations. This can be seen by taking a look at the partial derivative

$$\frac{\partial s_c(t)}{\partial net_{c_{in}}(t)} = \frac{\partial s_c(t)}{\partial y_{c_{in}}(t)} \frac{\partial y_{c_{in}}(t)}{\partial net_{c_{in}}(t)} = y_\iota(t) f_c'\left(net_{c_{in}}(t)\right).$$

If there is a small input $|net_{c_{in}}(t)| \ll 1$, we get $f_c'(net_{c_{in}}(t)) \approx 1$ and can estimate

$$\frac{\partial s_c(t)}{\partial net_{c_{in}}(t)} \approx y_\iota(t).$$

All in all, this LSTM is able to store information and learn long-term dependencies, but it has one drawback which will be discussed in 2.2.

2.2 Learning to Forget

For long time series the internal state is unbounded (compare with F. A. Gers, J. Schmidhuber andF. Cummins, 1999, 2.1). Assuming a positive or negative input and a non zero

activation of the IG, the absolute activation of the internal state grows over time. Using the weight-space symmetries in a network with at least one hidden layer (Bishop, 2006, 5.1.1) we assume without loss of generality $y_{c_{in}}(t) \geq 0$, so $s_c(t) \xrightarrow{t\to\infty} \infty$. Hence, the activation function h_c saturates and (3) can be simplified to

$$y_c(t) = \underbrace{h_c\left(s_c(t)\right)}_{\to 1} y_\omega(t) \approx y_\omega(t).$$

Thus, for great activations of $s_c(t)$ the whole LSTM works like a unit with a logistic activation function. A similar problem can be observed for the gradient. The gradient coming from the output is multiplied by the activation of the OG and the derivative of h_c. For great values of $s_c(t)$ we get $h_c'\left(s_c(t)\right) \to 0$ and we can estimate the partial derivative

$$\frac{\partial y_c(t)}{\partial s_c(t)} = h_c'\left(\left(s_c(t)\right) \cdot y_\omega(t) \approx 0,\right.$$

which can be interpreted that the OG is not able to propagate back the gradient into the LSTM. Some solutions to solve the linear growing state problem are introduced in F. A. Gers, J. Schmidhuber andF. Cummins (1999). They tried to stabilize the LSTM with a "state decay" by multiplying the internal state in each time step with a value $\in (0, 1)$, which did not improve the performance. Another solution was to add an additional gate, the forget gate (FG or ϕ). The last state $s_c(t-1)$ is multiplied by the activation of the FG before it is added to the current state $s_c(t)$. So we can substitute (2) by

$$s_c(t) = y_{c_{in}}(t) \cdot y_\iota(t) + s_c(t-1) \cdot y_\phi(t),$$

so that the truncated gradient in (4) is changed to

$$\begin{aligned}
\frac{\partial s_c(t)}{\partial s_c(t-1)} &= y_{c_{in}}(t) \cdot \frac{\partial y_\iota(t)}{\partial s_c(t-1)} + y_\iota(t) \cdot \frac{\partial y_{c_{in}}(t)}{\partial s_c(t-1)} + y_\phi(t) \\
&\underset{tr}{=} y_\phi(t)
\end{aligned}$$

and for longer time series we get $\forall \tau \in \mathbb{N}$

$$\frac{\partial s_c(t)}{\partial s_c(t-\tau)} \underset{tr}{=} \prod_{t'=0}^{\tau-1} y_\phi(t-t').$$

Now, the *Extended LSTM* is able to learn to forget its previous state. However, an Extended LSTM is still able to work like an standard LSTM without FG by having an activation $y_\phi(t) \approx 1$. In this paper we denote the Extended LSTM as *LSTM*
Another point of view was introduce in Bengio et al. (1994): To learn long-term dependencies a system must have an architecture to that an input can be saved over long time and does not suffer from the "vanishing gradient" problem. On the other hand the system should avoid an "exploding gradient", which means that a small disturbance has a growing influence over time. In this paper we do not want to solve the problem of vanishing and exploding gradient for a whole system, we want to solve this problem only for one single cell. But we think that it is an necessary condition to provide long time dependencies of a system.

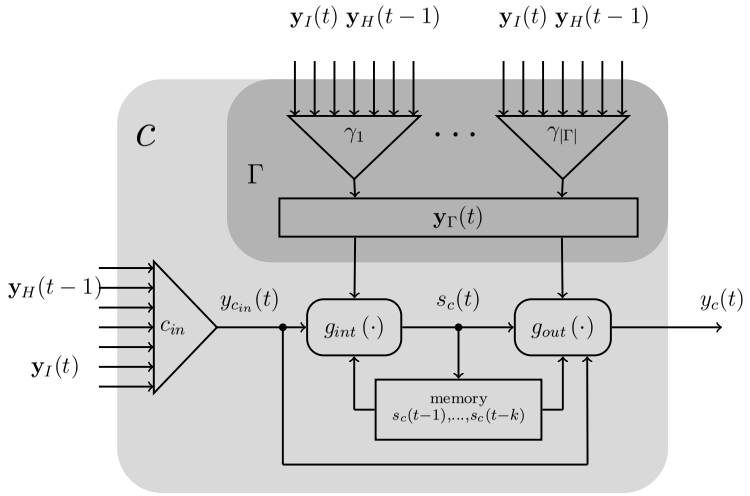


Figure 2: Schematic diagram of a cell: The function g_{int} calculates the internal state $s_c(t)$ from the previous internal states $s_c(t-1),\ldots,s_c(t-k)$ and the cell input $y_{c_{in}}(t)$ using the gate activations $\mathbf{y}_\Gamma(t)$. The function g_{out} calculates the output $y_c(t)$ of the cell from the actual and previous internal states $s_c(t),\ldots,s_c(t-k)$, the cell input $y_{c_{in}}(t)$ also using the gate activations $\mathbf{y}_\Gamma(t)$.

3. Cells and Their Properties

In this section we want to introduce a general cell and figure out properties for these cells which probably lead to the good performance observed by LSTM cells.

Definition 2 (Cell, cf. Fig. 2) *A* cell, c, of order k *consists of*

- *one designated* input *unit, c_{in}, with sigmoid activation function f_c (typically $f_c = \tanh$ unless specified otherwise);*
- *a set Γ (not containing c_{in}) of units called* gates $\gamma_1, \gamma_2, \ldots$ *with sigmoid activation functions f_{γ_i}, $i = 1, \ldots$ (typically logistic $f_{\gamma_i} = f_{log}$ unless specified otherwise);*
- *an arbitrary function, g_{int}, and a cell activation function, g_{out}, mapping into $[-1, 1]$.*

Each unit of $\Gamma \cup \{c_{in}\}$ receives the same set of input activations. The cell update in time step $t \in \mathbb{N}$ is performed in three subsequent phases:

1. *Following the classical update scheme of neurons (see Section 2), all units in $\Gamma \cup \{c_{in}\}$ calculate synchronously their activations, which will be denoted by $\boldsymbol{y}_\Gamma(t) := \left(y_\gamma(t)\right)_{\gamma\in\Gamma}$ and $y_{c_{in}}(t)$. Furthermore, we call $y_{c_{in}}(t)$ the* input activation *of the cell.*
2. *Then, the cell computes it's so-called* internal state
$$s_c(t) := g_{int}\left(\boldsymbol{y}_\Gamma(t), y_{c_{in}}(t), s_c(t-1), \ldots, s_c(t-k)\right) .$$
3. *Finally, the cell computes it's so-called* output activation
$$y_c(t) := g_{out}\left(\boldsymbol{y}_\Gamma(t), y_{c_{in}}(t), s_c(t), s_c(t-1), \ldots, s_c(t-k)\right) .$$

In this paper we concentrate on first order cells ($k = 1$). Now, we use Definition 2 to re-introduce the (Extended) LSTM cell.

Remark 3 (LSTM cell) *An* LSTM cell *is a cell of order* 1 *where $h_c = \tanh$ and*

- $\Gamma = \{\iota, \phi, \omega\}$
- $s_c(t) := g_{int}\left(\boldsymbol{y}_\Gamma(t), y_{c_{in}}(t), s_c(t-1)\right) := y_{c_{in}}(t)y_\iota(t) + s_c(t-1)y_\phi(t)$
- $y_c(t) := g_{out}\left(\boldsymbol{y}_\Gamma(t), s_c(t)\right) := h_c\left(s_c(t)\right) y_\omega(t)$

Properties of cells. Developing the 1D LSTM cells, the main idea is to save *exactly one piece of information* over a long time series and to propagate the gradient back over this long time, so that the system can learn precise storage of this piece of information. In instance a given input $y_{c_{in}}$ (which represent the information) at time t_{in} should be stored into the cell state s_c until the information is required at time t_{out}.
To be able to prove the following properties, we will assume the truncated gradient defined in Definition 1. Nevertheless we will use the full gradient in our Experiments, because it turned out that it works much better. The next two properties of a cell ensure the ability to work as such a memory.

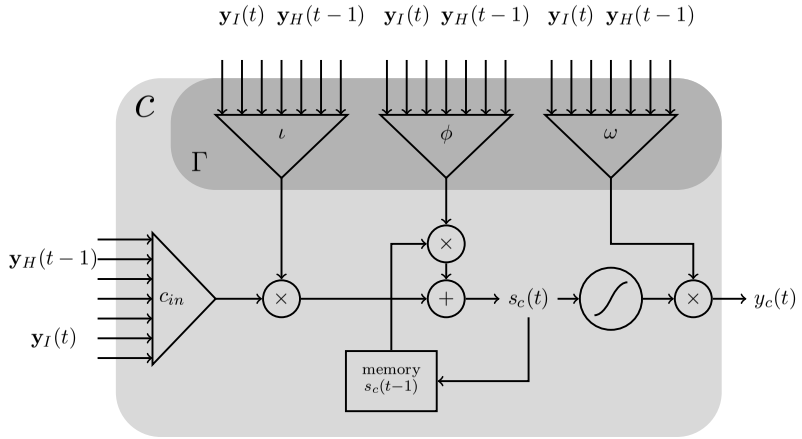


Figure 3: Schematic diagram of a one-dimensional LSTM cell: The input (c_{in}) is multiplied by the IG (ι). The previous state $s_c(t-1)$ is gated by the FG (ϕ) and added to the activation coming from the IG and input. The output of the cell is the squashed internal state (squashed by $h_c(x) = \tanh(x)$) and gated by the OG (ω).

The first property should ensure that an input $y_{c_{in}}$ at time t_{in} can be memorized (the cell input is open) in the internal activation s_c until t_{out} (the cell memorizes) and has a negligibly influence on the internal activation for $t > t_{out}$ (the cell forgets). In addition, the cell is able to prevent influence of other inputs at time steps $t \neq t_{in}$ (the cell input is closed).

Definition 4 (Not vanishing gradient (NVG)) *A cell c allows an NVG* $:\Leftrightarrow$
For arbitrary $t_{in}, t_{out} \in \mathbb{N}, t_{in} \leq t_{out}, \forall \delta > 0$ there exist gate activations $\boldsymbol{y}_\Gamma(t)$ such that for any $t_1, t_2 \in \mathbb{N}$

$$\frac{\partial s_c(t_2)}{\partial y_{c_{in}}(t_1)} \underset{tr}{\in} \begin{cases} [1-\delta, 1] & \text{for} \quad t_1 = t_{in} \text{ and } t_{in} \leq t_2 \leq t_{out} \\ [0, \delta] & \text{otherwise} \end{cases} \tag{5}$$

holds.

The next definition guaranties that at any time $t \in \mathbb{N}$ the gate activations can (the cell output is open) or not (the cell output is closed) distribute the piece of information saved in s_c to the network. This is an important property because the piece of information can be memorized in the cell without presenting it to the network. Note that the decision is just dependent on gate activations at time t and there are no constraints to previous gate activations. In Definition 2 we require $y_c(t) \in [-1, 1]$ whereas $s_c(t) \in \mathbb{R}$. So we cannot have arbitrarily small intervals of the derivative as in (5), but we can ensure two distinct intervals for open and closed cell output. When we take Definition 4 and 5 together, a cell is able to save an input over long term series, can decide at each time step whether or not it is presented to the network and can forget the saved input.

Definition 5 (Controllable output dependency (COD)) *A cell c of order k allows an COD* $:\Leftrightarrow$
There exist $\delta_1, \delta_2 \in (0, 1), \delta_2 < \delta_1$ so that for any time $t \in \mathbb{N}$ there exists a gate vector $\boldsymbol{y}_\Gamma(t)$ leading to open output dependency

$$\frac{\partial y_c(t)}{\partial s_c(t)} \in [\delta_1, 1] \tag{6}$$

and there exists another gate vector $\boldsymbol{y}_\Gamma(t)$ leading to a closed output dependency

$$\frac{\partial y_c(t)}{\partial s_c(t)} \in [0, \delta_2]\,. \tag{7}$$

The third property is a kind of stability criterion. An unwanted case is that a small change (caused by any noisy signal) at time step t_{in} has a growing influence at later time steps. This is equivalent to an exploding gradient over time. Controlling the gradient of the whole system and avoiding him not to explode is a hard problem. But we can at least avoid the exploding gradient in one cell. This should be prohibited for *any* gate activations.

Definition 6 (Not exploding gradient (NEG)) *A cell c has an NEG* $:\Leftrightarrow$
For any time steps $t_{in}, t \in \mathbb{N}, t_{in} < t$ and any gate activations $\boldsymbol{y}_\Gamma(t)$ the truncated gradient in bounded by

$$\frac{\partial s_c(t)}{\partial s_c(t_{in})} \underset{tr}{\in} [0, 1]\,.$$

We think that a cell fulfilling these three properties can work as stable memory. To be able to prove these properties for the LSTM cell we have to considerate the gate activations. In general, the activation function of the gates does not have to be the logistic activation function f_{log}, whereas for this paper we set $\forall\gamma \in \Gamma : f_\gamma := f_{log}$. So the activation of gates can never be exactly 0 or 1, because of a finite input activation $net_\gamma(t)$ to the gate activation function. But a gate can have an activation $y_\gamma(t) \in [1-\varepsilon, 1)$ if it is opened or $y_\gamma(t) \in (0, \varepsilon]$ if it is closed, because for a realistic large input activation $net_\gamma(t) \geq 7$ (low input activation $net_\gamma(t) < -7$) we get an activation within the interval $y_\gamma(t) \in [1-\varepsilon, 1)$ ($y_\gamma(t) \in (0, \varepsilon]$) with $\varepsilon < \frac{1}{1000}$. Handling with these activation intervals we can prove the definitions for the LSTM cell. Now we can prove whether or not the LSTM cell has these properties.

Theorem 7 (Properties of the LSTM cell) *The 1D LSTM cell allows NVG and has an NEG, but does not allow COD.*

Proof see A.1 in appendix. ■

4. Expanding to More Dimensions

In A. Graves, S. Fernandez andJ. Schmidhuber (2007) the 1D LSTM cell is extended to an arbitrary number of dimensions; this is solved by using one FG for each dimension. In many publications using the MD LSTM cell in MDRNNs outperform state-of-the-art recognition systems (for example see A. Graves andJ. Schmidhuber, 2008).
But by expanding the cell to the MD case, the absolute value of the internal state $|s_c|$ can grow faster than linear over time. When $\left|s_c^{\boldsymbol{p}}\right| \to \infty$ and there are peephole connections (for peephole connection details see F. A. Gers, N. Schraudolph andJ. Schmidhuber, 2002), the cells have an output activation of $y_c^{\boldsymbol{p}} \in \{-1, 0, 1\}$: The internal state multiplied by the peephole weight overlays the other activation-weight-products and this leads to an activation of the OG $y_\omega^{\boldsymbol{p}} \in \{0, 1\}$ and a squashed internal state $h_c\left(s_c^{\boldsymbol{p}}\right) \in \{-1, 1\}$. So the output of the cell is $y_\omega^{\boldsymbol{p}} h_c\left(s_c^{\boldsymbol{p}}\right) = y_c^{\boldsymbol{p}} \in \{-1, 0, 1\}$. But also without peephole connections the internal state can grow, which leads to $h_c\left(s_c^{\boldsymbol{p}}\right) \in \{-1, 1\}$ and the cell works like a conventional unit with a logistic activation function $y_c(t) \approx \pm y_\omega(t)$.
Our goal is to transfer the Definitions 4, 5 and 6 defined in Section 3 into the MD case and we will see that the MD LSTM cell has an exploding gradient. In the next sections we will provide alternative cell types, that fulfill two or all of these definitions.
In the 1D case it is clear, that there is just one way to come from date t_1 to date t_2, when $t_1 < t_2$, by incrementing t_1 as long as t_2 is reached. For the MD case the number of paths depends on the number of dimensions and the distance between these two dates. An MD path is defined as follows.

Definition 8 (MD path) *Let $\boldsymbol{p}, \boldsymbol{q} \in \mathbb{N}^D$ be two dates. A $\boldsymbol{p}$-$\boldsymbol{q}$-path π of length $k \geq 0$ is a sequence*

$$\pi := \{\boldsymbol{p} = \boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k = \boldsymbol{q}\}$$

with $\forall i \in \{1, \ldots, k\} \exists! d \in \{1, \ldots, D\} : (\boldsymbol{p}_i)_d^- = \boldsymbol{p}_{i-1}$. Further, let $\pi_i := \boldsymbol{p}_i$.

We can define the distance vector

$$\overrightarrow{\boldsymbol{pq}} := \boldsymbol{q} - \boldsymbol{p} = \begin{pmatrix} \boldsymbol{q}_1 - \boldsymbol{p}_1 \\ \vdots \\ \boldsymbol{q}_D - \boldsymbol{p}_D \end{pmatrix} = \begin{pmatrix} \overrightarrow{\boldsymbol{pq}}_1 \\ \vdots \\ \overrightarrow{\boldsymbol{pq}}_D \end{pmatrix}$$

between the dates $\boldsymbol{p}$ and $\boldsymbol{q}$. When $\overrightarrow{\boldsymbol{pq}}$ has at least one negative component, there exists no $\boldsymbol{p}$-$\boldsymbol{q}$-path. Otherwise there exist exactly

$$\#\{\overrightarrow{\boldsymbol{pq}}\} := \binom{\sum_{i=1}^D \overrightarrow{\boldsymbol{pq}}_i}{\overrightarrow{\boldsymbol{pq}}_1, \ldots, \overrightarrow{\boldsymbol{pq}}_D} = \frac{\left(\sum_{i=1}^D \overrightarrow{\boldsymbol{pq}}_i\right)!}{\prod_{i=1}^D \overrightarrow{\boldsymbol{pq}}_i!}$$

$\boldsymbol{p}$-$\boldsymbol{q}$-paths (compare with the multinomial coefficient). We write $\boldsymbol{p} < \boldsymbol{q}$ when $\#\{\overrightarrow{\boldsymbol{pq}}\} \geq 1$ and $\boldsymbol{p} \leq \boldsymbol{q}$ when $\boldsymbol{p} = \boldsymbol{q} \vee \boldsymbol{p} < \boldsymbol{q}$. Now we can extend the definitions of the 1D case to the MD case, whereas we concentrate on the MD cells of order 1.

Definition 9 (MD cell) *An MD cell, c, of order 1 and dimension D consists of the same parts as a 1D cell of order 1. The cell update in date $\boldsymbol{p} \in \mathbb{N}^D$ is performed in three subsequent phases:*

1. *Following the classical update scheme of neurons (see Section 2), all units in $\Gamma \cup \{c_{in}\}$ synchronously calculate their activations, which will be denoted by $\boldsymbol{y}_\Gamma^{\boldsymbol{p}} = \left(y_\gamma^{\boldsymbol{p}}\right)_{\gamma \in \Gamma}$. Furthermore, we call $y_{c_{in}}^{\boldsymbol{p}}$ the* input activation *of the cell.*

2. *Then, the cell computes it's so-called* internal state

$$s_c^{\boldsymbol{p}} := g_{int}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, y_{c_{in}}^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right).$$

3. *Finally, the cell computes it's so-called* output activation

$$y_c^{\boldsymbol{p}} := g_{out}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, y_{c_{in}}^{\boldsymbol{p}}, s_c^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right).$$

Using this, we can reintroduce the LSTM cell as well as Definition 4, 5 and 6 for the MD case:

Definition 10 (MD LSTM cell) *An MD LSTM cell is a cell of dimension D and order 1 where $h_c = \tanh$ and*

- $\Gamma = \{\iota, (\phi, 1), \ldots, (\phi, D), \omega\}$
- $s_c^{\boldsymbol{p}} = g_{int}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, y_{c_{in}}^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right) = y_\iota^{\boldsymbol{p}} y_{c_{in}}^{\boldsymbol{p}} + \sum_{d=1}^D s_c^{\boldsymbol{p}_d^-} y_{\phi,d}^{\boldsymbol{p}}$
- $y_c^{\boldsymbol{p}} = g_{out}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}}\right) = h_c\left(s_c^{\boldsymbol{p}}\right) y_\omega^{\boldsymbol{p}}$

Definition 11 (MD Not vanishing gradient (NVG)) *An MD cell c allows an NVG* $:\Leftrightarrow$
For arbitrary $\boldsymbol{p}_{in}, \boldsymbol{p}_{out} \in \mathbb{N}^D, \boldsymbol{p}_{in} \leq \boldsymbol{p}_{out}, \forall \delta > 0$ there exist $\forall \boldsymbol{p} \in \mathbb{N}^D$ gate activations $\boldsymbol{y}_\Gamma^{\boldsymbol{p}}$ such that for any $\boldsymbol{p}_1, \boldsymbol{p}_2 \in \mathbb{N}^D$

$$\frac{\partial s_c^{\boldsymbol{p}_2}}{\partial y_{c_{in}}^{\boldsymbol{p}_1}} \underset{tr}{\in} \begin{cases} [1-\delta, 1] & \text{for} \quad \boldsymbol{p}_1 = \boldsymbol{p}_{in} \text{ and } \boldsymbol{p}_{in} \leq \boldsymbol{p}_2 \leq \boldsymbol{p}_{out} \\ [0, \delta] & \text{otherwise} \end{cases} \tag{8}$$

holds.

Definition 12 (MD Controllable output dependency (COD)) *An MD cell c allows an COD* $:\Leftrightarrow$
There exist $\delta_1, \delta_2 \in (0,1), \delta_2 < \delta_1$ so that for any time $t \in \mathbb{N}$ there exists a gate vector $\boldsymbol{y}_\Gamma^{\boldsymbol{p}}$ leading to open output dependency

$$\frac{\partial y_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}}} \in [\delta_1, 1] \tag{9}$$

and there exists another gate vector $\boldsymbol{y}_\Gamma^{\boldsymbol{p}}$ leading to a closed output dependency

$$\frac{\partial y_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}}} \in [0, \delta_2]\,. \tag{10}$$

Definition 13 (MD Not exploding gradient (NEG)) *An MD cell c has an NEG* $:\Leftrightarrow$
For any time steps $\boldsymbol{p}_{in}, \boldsymbol{p} \in \mathbb{N}^D, \boldsymbol{p}_{in} < \boldsymbol{p}$ and any gate activations $\boldsymbol{y}_\Gamma^{\boldsymbol{p}}$ the truncated gradient in bounded by

$$\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} [0, 1]$$

We can now consider these definitions for the MD LSTM cell.

Theorem 14 (NVG of MD LSTM cells) *An MD LSTM cell allows an NVG.*

Proof see A.2 in appendix ■

For arbitrary activations of FGs the partial derivative $\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}}$ can grow over time:

Theorem 15 (NEG of MD LSTM cells) *An MD LSTM cell can have an exploding gradient, when $D \geq 2$.*

Proof see A.3 in appendix. ■

The MD LSTM cell does not allow the COD, because the 1D case is a special case of the MD case.
Our idea for the next section is to change the MD LSTM layout, so that it has an NEG.

5. Reducing the MD LSTM Cell to One Dimension

In the last section, we showed that the MD LSTM cell can have an exploding gradient. We tried different ways to solve this problem. For example we divided the activation of the FG by the number of dimensions. Then the gradient cannot explode over time, but the gradient vanishes along some paths rapidly. Another approach was to give the cells the opportunity to learn to stabilize itself, when the internal state starts diverging. Therefore we add an additional peephole connection between the square value of the previous internal states $\left(s_c^{\boldsymbol{p}_d^-}\right)^2$ and the FGs so that the cell is able to learn that it has to close the FG for large internal states. This also does not make a significant difference. Also forcing the cell to learn to stabilize itself by adding an error

$$Loss_{state} = \varepsilon \left\| s_c^{\boldsymbol{p}} \right\|_p$$

with $p = \{1, 2, 3, 4\}$ and different learning rates ε does not work. So we tried to change the layout of the MD LSTM cell.

5.1 MD LSTM Stable Cell

In Section 3 we realized that 1D LSTM cells work good and the gradient does not explode, but in the MD case it does. Our idea is to combine the previous states $s_c^{\boldsymbol{p}_d^-}$ at date $\boldsymbol{p}$ to one previous state $s_c^{\boldsymbol{p}^-}$ and take the 1D form of the LSTM cell. For this reason we call this cell *LSTM Stable cell*.
Therefore, a function

$$s_c^{\boldsymbol{p}^-} = f\left(s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right)$$

is needed, so that the following two benefits of the 1D LSTM cell remain:

1. The MD LSTM Stable cell has an NEG
2. The MD LSTM Stable cell allows NVG.

The convex combination

$$s_c^{\boldsymbol{p}^-} = f\left(s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right) = \sum_{d=1}^{D} \lambda_d^{\boldsymbol{p}} s_c^{\boldsymbol{p}_d^-}, \forall d = 1, \ldots, D : \lambda_d^{\boldsymbol{p}} \geq 0, \sum_{d=1}^{D} \lambda_d^{\boldsymbol{p}} = 1 \tag{11}$$

of all states satisfies these both points (see Theorems 17 and 18). To calculate these D coefficients we want to use the activation of D gates and we call them lambda gates (LG or λ).

Definition 16 (MD LSTM Stable cell) *An MD LSTM Stable cell is a cell of dimension D and order 1 where $h_c = \tanh$ and*

- $\Gamma = \{\iota, (\lambda, 1), \ldots, (\lambda, D), \phi, \omega\}$

- $s_c^{\boldsymbol{p}^-} = g_{conv}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right) = \sum_{d=1}^{D} s_c^{\boldsymbol{p}_d^-} \frac{y_{\lambda,d}^{\boldsymbol{p}}}{\sum_{d'=1}^{D} y_{\lambda,d'}^{\boldsymbol{p}}}$
- $s_c^{\boldsymbol{p}} = g_{int}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, y_{c_{in}}^{\boldsymbol{p}}, s_c^{\boldsymbol{p}^-}\right) = y_t^{\boldsymbol{p}} y_{c_{in}}^{\boldsymbol{p}} + s_c^{\boldsymbol{p}^-} y_\phi^{\boldsymbol{p}}$
- $y_c^{\boldsymbol{p}} = g_{out}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}}\right) = y_\omega^{\boldsymbol{p}} h_c\left(s_c^{\boldsymbol{p}}\right)$

Using these equations we can test the cell for its properties. The MD LSTM Stable cell does not have the COD, because the 1D LSTM cell also does not have this property. For the other propertiese we get:

Theorem 17 (LTD of MD LSTM Stable cells) *An MD LSTM Stable cell allows NVG.*

Proof See A.4 in appendix. ■

Theorem 18 (NEG of MD LSTM Stable cells) *An MD LSTM Stable cell has an NEG.*

Proof See A.5 in appendix. ■

Reducing the number of gates by one. When $D \geq 2$ an MD LSTM Stable cell has one more gate than a classical MD LSTM (for $D = 1$ the both cells are equivalent). But it is possible to reduce the number of LGs by one. One solution is to choose one dimension $d' \in \{1, \ldots, D\}$ which does not get an LG. Its activation is calculated by

$$y_{\lambda,d'}^{\boldsymbol{p}} = \prod_{d \in \{1,..,D\} \setminus \{d'\}} \left(1 - y_{\lambda,d}^{\boldsymbol{p}}\right).$$

In the special case of $D = 2$ we can choose $d' = 2$ and we get $\sum_{d'=1}^{2} y_{\lambda,d'} = y_{\lambda,1} + (1 - y_{\lambda,1}) = 1$ and the update equation of the internal state can be simplified to

$$s_c^{\boldsymbol{p}} = g_{int}\left(y_t^{\boldsymbol{p}}, y_{\lambda,1}^{\boldsymbol{p}}, y_\phi^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, s_c^{\boldsymbol{p}_2^-}\right) = y_t^{\boldsymbol{p}} y_{c_{in}}^{\boldsymbol{p}} + y_{\lambda,1}^{\boldsymbol{p}} s_c^{\boldsymbol{p}_1^-} + \left(1 - y_{\lambda,1}^{\boldsymbol{p}}\right) s_c^{\boldsymbol{p}_2^-}.$$

6. Bounding the Internal State

In the last sections we discussed the growing of the EC over time and we found a solution to have a NGEC for higher dimensions. Nevertheless it is possible that the internal state grows linearly over time. When we take a look at Definition 10, we see that the partial derivative for $\boldsymbol{p} = \boldsymbol{p}_{out}$ depends on $h_c'\left(s_c^{\boldsymbol{p}}\right)$. So having the inequality

$$\frac{\partial y_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}}} \leq h_c'(s_c^{\boldsymbol{p}}) \quad \text{with} \quad h_c'\left(s_c^{\boldsymbol{p}}\right) \xrightarrow{|s_c^{\boldsymbol{p}}| \to \infty} 0$$

the cell allows NVG defined in Definition 11, but actually we have $\frac{\partial y_c^{\boldsymbol{p}_{out}}}{\partial y_{c_{in}}^{\boldsymbol{p}_{in}}} \xrightarrow{|s_c^{\boldsymbol{p}_{out}}| \to \infty} 0$ for arbitrary gate activations. Again, ideas like state decay, additional peephole connections or additional loss functions like mentioned in Section 4 either do not work or destroy the NVG of the LSTM and LSTM Stable cell. So, our solution is to change the architecture of the MD LSTM Stable cell, so that it fulfills has an NEG and allows NVG and COD. The key idea is to bound the internal state, so that for all inputs $\left|y_{c_{in}}^{\boldsymbol{p}}\right| \leq 1$, $\boldsymbol{p} \in \mathbb{N}^D$ the internal state is bounded by $\left|s_c^{\boldsymbol{p}}\right| \leq 1$.
Note that this is comparable with the well-known Bounded-Input-Bounded-Output-Stability (BIBO-Stability). To create an MD cell that has an NEG, allows NVG and has a bounded internal state, we take the MD LSTM Stable cell proposed in the last section and change its layout. Therefore we calculate the activation of the IG as function of the FG, so that we achieve $\left|s_c^{\boldsymbol{p}}\right| \leq 1$ by choosing $y_t^{\boldsymbol{p}} := 1 - y_\phi^{\boldsymbol{p}}$. So the activation of the FG controls how much leaks from the previous states. The activation of the FG can also be interpreted as switch, if the internal activation, the new activation or a convex combination of these both activations should be stored in the cell. So the s_c can be seen as *time-dependent exponential moving average* of $y_{c_{in}}$.

Definition 19 (MD Leaky cell) *An MD Leaky cell is a cell of dimension D and order 1 where $h_c = \tanh$ and*

- $\Gamma = \{(\lambda, 1), \ldots, (\lambda, D), \phi, \omega\}$
- $s_c^{\boldsymbol{p}^-} = g_{conv}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right) = \sum_{d=1}^{D} s_c^{\boldsymbol{p}_d^-} \frac{y_{\lambda,d}^{\boldsymbol{p}}}{\sum_{d'=1}^{D} y_{\lambda,d'}^{\boldsymbol{p}}}$
- $s_c^{\boldsymbol{p}} = g_{int}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, y_{c_{in}}^{\boldsymbol{p}}, s_c^{\boldsymbol{p}^-}\right) = \left(1 - y_\phi^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + s_c^{\boldsymbol{p}^-} y_\phi^{\boldsymbol{p}}$
- $y_c^{\boldsymbol{p}} = g_{out}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}}\right) = y_\omega^{\boldsymbol{p}} h_c\left(s_c^{\boldsymbol{p}}\right)$

Now we can prove that the resulting cell has all benefits.

Theorem 20 *The MD Leaky cell has an NEG and allows NVG and COD.*

Proof See A.6 in appendix. ■

The MD Leaky cell can have one gate less than the MD LSTM cell and the MD LSTM Stable cell and because of this, the update path requires less computations.

7. General Derivation of Leaky Cells

So far we proposed cells for the MD case, which are able to provide long term memory. But especially in MDRNNs with more than one MD layer it is hard to measure if and how much long term dependencies are used and even if it is useful. Another way to interpret the cell is to consider them as kind of MD feature extractor like "feature maps" in Convolutional Neural Networks (Bengio and LeCun, 1995). Then the aim is to construct an MD cell which is able to generate useful features. Having a hierarchical Neural Network like in Bengio and LeCun (1995) and A. Graves andJ. Schmidhuber (2008) over the hierarchies the number of

features increases with a simultaneously decreasing feature resolution. Features in a layer with low resolution can be seen as low frequency features in comparison to features in a layer with high resolution. So it would be useful to construct a cell as feature extractor which produces a low frequency output in comparison to its input. In appendix B we take a closer look at the theory of linear shift invariant (LSI)-systems and their frequency analysis and analyse a first order LSI-system regarding its free selectable parameters using the $\mathcal{F}$- and $\mathcal{Z}$-transform. There, we derive the *MD LeakyLP cell* (see Definition 21) and 5 additional first order MD cells, which we do not test in Section 8.

Definition 21 (MD LeakyLP cell) *An MD LeakyLP cell is a cell of dimension D and order 1 where $h_c = \tanh$ and*

- $\Gamma = \{(\lambda, 1), \ldots, (\lambda, D), \phi, \omega_0, \omega_1\}$
- $s_c^{\boldsymbol{p}^-} = g_{conv}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}_1^-}, \ldots, s_c^{\boldsymbol{p}_D^-}\right) = \sum_{d=1}^{D} s_c^{\boldsymbol{p}_d^-} \frac{y_{\lambda,d}^{\boldsymbol{p}}}{\sum_{d'=1}^{D} y_{\lambda,d'}^{\boldsymbol{p}}}$
- $s_c^{\boldsymbol{p}} = g_{int}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, y_{c_{in}}^{\boldsymbol{p}}, s_c^{\boldsymbol{p}^-}\right) = \left(1 - y_\phi^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + s_c^{\boldsymbol{p}^-} y_\phi^{\boldsymbol{p}}$
- $y_c^{\boldsymbol{p}} = g_{out}\left(\boldsymbol{y}_\Gamma^{\boldsymbol{p}}, s_c^{\boldsymbol{p}}, s_c^{\boldsymbol{p}^-}\right) = h_c\left(s_c^{\boldsymbol{p}} y_{\omega_0}^{\boldsymbol{p}} + s_c^{\boldsymbol{p}^-} y_{\omega_1}^{\boldsymbol{p}}\right)$

Setting the second OG ($y_{\omega_1}^{\boldsymbol{p}}$) to zero, the LeakyLP cell corresponds to the Leaky cell, hence it fulfills all three properties, but has one more gate, which is as much gates as the LSTM cell.

8. Experiments

RNNs with 1D LSTM cells are well studied. In some experiments the activations of the gates and the internal state are observed and one can see that the cell can really learn, when to "forget" information and when the internal state should be accessible for the network (see F. A. Gers, N. Schraudolph andJ. Schmidhuber, 2002). However, we did not find experiments like these for the MD case and we do not want to transfer these experiments into the MD case. Instead we compare the different cell types with each other in two scenarios where the MD RNNs with LSTM cells perform very well. In both benchmarks the task is to transcribe a handwritten text on an image, so we have a 2D RNN. In this case we compare the cells on the IFN/ENIT (Pechwitz, M. and Maddouri, S. and Märgner, V. and Ellouze, N. and Amiri, H. and others, 2002) and the Rimes database (Augustin, E. and Brodin, J.-M and Carré, M. and Geoffrois, E. and Grosicki, E. and Prêteux, F., 2006). Both tasks are solved with the MD RNN layout described in A. Graves andJ. Schmidhuber (2008) and shown in Figure 4. All networks are trained with *Backpropagation through time (BPTT)* . To compare the different cell types in RNNs with each other we take 10 RNNs with different weight initializations of each cell type and calculate the minimum, the maximum and the median of the best label error rate (LER) on a validation set of these 10 RNNs. In all tables we present these three LERs to compare the cell types.
We think it is more important to have stable cells in the lower MD layers because of two reasons: First, when we have just a few cells in a layer, the saturation of one cell has a

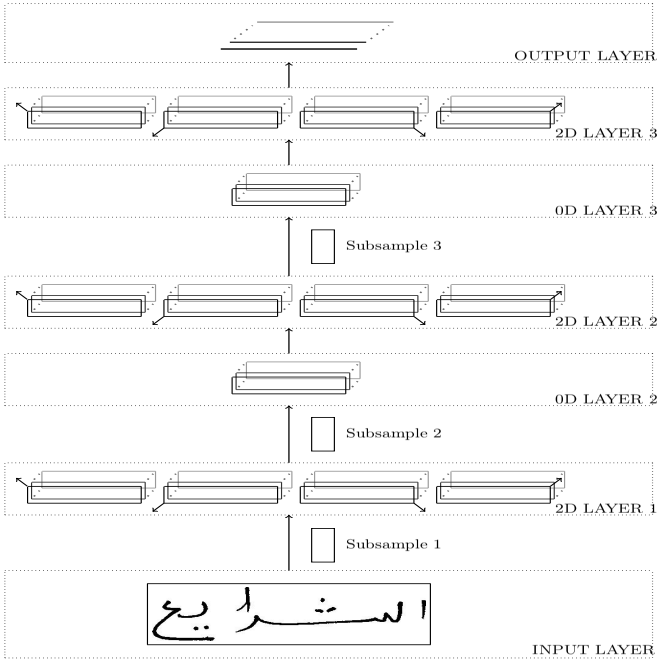


Figure 4: Architecture of the hierarchical MDRNN used for the experiments: It is equivalent to A. Graves andJ. Schmidhuber (2008, Figure 2). A 2D layer contains 2^2 distinct layers (for each combination of scanning direction left/right and up/down one layer). To reduce the number of weights between two 2D layers, a 0D layer is inserted, which contains units with tanh as activation function. They have dimension 0 because they have no recurrent connections. These layers can be seen as feed-forward or convolutional layer. Each 2D layer (or its allocated 0D layer) reduces its size in in x and y dimension using a two-dimensional subsampling. Simultaneously the number of feature maps (z-dimension) increases to have no bottleneck between input and output layer.

greater effect on the performance of the network. Second, in lower layers there are longer time series so having an unstable cell in such a layer, it has time to saturate. So our first experiment compares the recognition results when we substitute the LSTM cells in the lowest layer (which is "2D layer 1" in Figure 4) by the newly developed cells.
In the second experiment we compare the LSTM cell and the LeakyLP cell also in the higher MD layers ("2D layer 2 and 3 in Figure 4), to evaluate if the LeakyLP cell work better also

Celltype	Label-Error-Rate in Percent min	max	median
LSTM	8,58%	14,73%	10,58%
Stable	8,78%	11,75%	9,55%
Leaky	8,87%	10,47%	9,10%
LeakyLP	8,24%	9,40%	8,93%

Table 1: Different cell types in the lowest MD layer

in long time series.
In Bengio (2012, 3.1.1) it is mentioned, that an important hyper parameter for a training is the learning rate, so another experiment is to train all networks with stochastic gradient decent with different learning rates $\delta \in \{1 \cdot 10^{-3}, 5 \cdot 10^{-4}, 2 \cdot 10^{-4}, 1 \cdot 10^{-4}\}$ and compare the best LER according a fixed learning rate.

8.1 The IFN/ENIT Database

This database contains handwritten names of towns and villages of Tunisia. The set is divided into 7 (a-f,s) sets, where 5 (a-e) are available for training and validation (for details see Pechwitz, M. and Maddouri, S. and Märgner, V. and Ellouze, N. and Amiri, H. and others, 2002). With all information we got from A. Graves, we were able to get comparable results to A. Graves andJ. Schmidhuber (2008). Therefor we divide the sets a-e into 30000 training samples and 2493 validation samples. All network are trained 100 epochs with a fixed learning rate $\delta = 1 \cdot 10^{-4}$. The LER is calculated on the validation set.

8.1.1 Different Cells in the Lowest MD Layer

In our first experiment we substitute the LSTM cell in the lowest MD layer. We take some of the cells described in this paper. In Table 1 the results are shown. The first row is the same RNN layout used in A. Graves andJ. Schmidhuber (2008). We can see, that the LeakyLP cell performs the best. Nevertheless the worst RNN with LeakyLP cells in the lowest MD layer performs worth than the best RNN with LSTM cells. So we cannot say, that LeakyLP is always better. But it can be observed that the variance of the RNN performance is very high with LSTM cells in the lowest MD layer. Our interpretation is that LSTM cells have a comparable performance like the LeakyLP cells in the lowest layer, when they do not saturate. Note, that the Leaky cell has one gate less, so they are faster and have less trainable weights.

8.1.2 Different Cells in Other MD Layers

Now we want to compare the best new developed cell—the LeakyLP cell—with the LSTM cell in the other MD layers. So we also substitute the LSTM cell in the upper MD layers. We enumerate the 2D layers like shown in Figure 4. In Table 2 we can see that substituting the LSTM cells only in the lowest or in the both lowest layer perform slightly better. The best results can be achieved when we use LeakyLP cells in 2D layer 1 and LSTM cells in 2D layer 3. Using LSTM in the middle layer seems to work slightly better than using the LeakyLP

Celltype in 2D layer 1	2	3	Label-Error-Rate in Percent min	max	median
LSTM	LSTM	LSTM	8,58%	14,73%	10,58%
LeakyLP	LSTM	LSTM	8,24%	9,40%	8,93%
LeakyLP	LeakyLP	LSTM	8,35%	11,27%	8,91%
LeakyLP	LeakyLP	LeakyLP	8,92%	11,69%	9,74%

Table 2: Different cells in other layers

Celltype	BP-delta	Label-Error-Rate in Percent min	max	median
LSTM	$1 \cdot 10^{-4}$	8,58%	14,73%	10,58%
LSTM	$2 \cdot 10^{-4}$	9,15%	16,86%	10,51%
LSTM	$5 \cdot 10^{-4}$	9,03%	21,77%	11,44%
LSTM	$1 \cdot 10^{-3}$	10,21%	30,20%	11,44%
LeakyLP	$1 \cdot 10^{-4}$	8,92%	11,69%	9,74%
LeakyLP	$2 \cdot 10^{-4}$	8,38%	9,09%	8,81%
LeakyLP	$5 \cdot 10^{-4}$	8,25%	8,95%	8,78%
LeakyLP	$1 \cdot 10^{-3}$	8,29%	9,20%	8,88%
LeakyLP	$2 \cdot 10^{-3}$	8,95%	12,81%	9,55%

Table 3: Performance of cells regarding learning-rate

cells instead. This fits to our intuition mentioned before that the LSTM cells perform better when they do not have a too long time series and when there are enough cells in one layer which do not saturate.

8.1.3 Performance of Cells Regarding Learning-Rate

When we take a look at the update equations and the proofs of the NEG it can be assumed, that the gradient going through the cells is lower for LeakyLP cells in contrast to LSTM cells. So we think the learning rate have to be larger for LeakyLP cells. In Table 3 we compare the networks with either only LSTM or LeakyLP cells. There we can see that the learning rate have to be much higher for the LeakyLP cells. In addition, the RNNs with LeakyLP cells are more robust to the choice of the learning rate.

8.2 The Rimes Database

One task of the Rimes database is the handwritten word recognition (for more details see E. Grosicki, M. Carré, J.-M. Brodin andE. Geoffrois, 2008; E. Grosicki andH. El-Abed, 2011). It contains 59292 images of french single words. It is divided into distinct subsets; a training set of 44196 samples, a validation set of 7542 samples and a test set of 7464 samples. We train the MD RNNs by using the training set for training and calculate the LER over the validation set, so the network is trained on 44196 training samples each epoch. The network used in this section differs only in the subsampling rate between two layers from the network used in A. Graves andJ. Schmidhuber (2008). When there is a subsampling between layers,

Celltype	Label-Error-Rate in Percent min	max	median
LSTM	14,96%	17,63%	16,50%
Stable	14,45%	16,02%	15,11%
Leaky	14,77%	16,39%	15,85%
LeakyLP	14,63%	15,78%	15,30%

Table 4: Different cell types in the lowest MD layer

Celltype in 2D layer 1	2	3	Label-Error-Rate in Percent min	max	median
LSTM	LSTM	LSTM	14,96%	17,63%	16,50%
LeakyLP	LSTM	LSTM	14,63%	15,78%	15,30%
LeakyLP	LeakyLP	LSTM	14,21%	15,57%	14,92%
LeakyLP	LeakyLP	LeakyLP	14,94%	16,18%	15,52%

Table 5: Different cells in other layers

the factors are 3×2 instead of 4×3 or 4×2. The rest of the experiment is the same like described in Section 8.1.

8.2.1 Different Cells in the Lowest MD Layer

In Table 4 we can see that substituting the LSTM in the lowest layer by one of the three cells improves the performance of the network, even the Leaky cell with one gate less.

8.2.2 Different Cells in Other MD Layers

We want to see the effect of the substitution of the LSTM cell by the LeakyLP cell in the upper MD layers. In Table 5 we can see that using LeakyLP cells in both lowest layers perform very well. So we also take this setup to try different learning rates.

Performance of Cells Regarding Learning-Rate. Using different learning rates we can see that the RNN with LeakyLP cells in the both lowest layers and the LSTM cells in the top layer can significantly improve the performance . Even the maximal LER of this RNN works better than the best network with LSTM cells in each layer.

9. Conclusion

In this paper we took a look at the one-dimensional LSTM cell and discussed the benefits of this cell. We found two properties, that probably make these cells so powerful in the one dimensional case. Expanding these properties to the multi dimensional case, we saw that the LSTM does not fulfill one of these properties any more. We solved this problem by changing the architecture of the cell. In addition we presented a more general idea how to create one dimensional or multi dimensional cells. We compare some newly developed cells with the LSTM cell on two data sets and we can improve the performance using the

Celltype	BP-delta	Label-Error-Rate in Percent min	max	median
LSTM	$1 \cdot 10^{-4}$	14,96%	17,63%	16,50%
LSTM	$2 \cdot 10^{-4}$	14,41%	16,88%	15,61%
LSTM	$5 \cdot 10^{-4}$	15,05%	16,27%	15,47%
LeakyLP	$1 \cdot 10^{-4}$	14,94%	16,18%	15,52%
LeakyLP	$5 \cdot 10^{-4}$	12,68%	13,95%	13,57%
LeakyLP in 2D layer 1 & 2	$2 \cdot 10^{-4}$	13,26%	14,04%	13,65%
LeakyLP in 2D layer 1 & 2	$5 \cdot 10^{-4}$	12,08%	13,42%	12,87%

Table 6: Performance of cells regarding learning-rate

new cell types. Due to this we think that substituting the multi-dimensional LSTM cells by the multi-dimensional *LeakyLP cell* could improve the performance of many system working with a multi-dimensional space.

Appendix A. Proofs

A.1 Proof of 7

Proof Let c be a 1D LSTM cell. To get the derivative $\frac{\partial s_c(t_2)}{\partial s_c(t_1)}$ according the truncated gradient between two time steps $t_1, t_2 \in \mathbb{N}$ we have to take a look at g_{int}.

$$\begin{aligned}\frac{\partial s_c(t_2)}{\partial s_c(t_1)} &= \frac{\partial g_{int}\left(\boldsymbol{y}_\Gamma(t_2), y_{c_{in}}(t_2), s_c(t_2-1)\right)}{\partial s_c(t_1)} \\ &= \underbrace{\frac{\partial\left(y_{c_{in}}(t_2) y_\iota(t_2)\right)}{\partial s_c(t_1)}}_{\underset{tr}{=}0} + \frac{\partial s_c(t_2-1)}{\partial s_c(t_1)} y_\phi(t_2) + s_c(t_2-1) \underbrace{\frac{\partial y_\phi(t_2)}{\partial s_c(t_1)}}_{\underset{tr}{=}0} \\ &\underset{tr}{=} \frac{\partial s_c(t_2-1)}{\partial s_c(t_1)} y_\phi(t_2) \\ &\underset{tr}{=} \prod_{t=t_1+1}^{t_2} y_\phi(t)\end{aligned} \tag{12}$$

(13)

In addition, $\forall t \in \mathbb{N}$ we have

$$\frac{\partial s_c(t)}{\partial y_{c_{in}}(t)} = y_\iota(t) \qquad \text{and} \qquad \frac{\partial y_c(t)}{\partial s_c(t)} = h_c'\left(s_c(t)\right) y_\omega(t). \tag{14}$$

We will prove the properties successively.
NEG: For the LSTM cell the FG $f_\phi = f_{log}$ ensures $y_\phi(t) \in (0,1)$, so using these bounds in (13) with

$$\frac{\partial s_c(t)}{\partial s_c(t_{in})} \underset{tr}{=} \prod_{t'=t_{in}+1}^{t} y_\phi(t') \in (0,1)$$

the LSTM cell has an NEG.
NVG: Therefore, we choose

$$y_\iota(t) \in \begin{cases} [1-\varepsilon, 1) & \text{if} \quad t = t_{in} \\ (0, \varepsilon] & \text{otherwise} \end{cases},$$
$$y_\phi(t) \in \begin{cases} [1-\varepsilon, 1) & \text{if} \quad t_{in} < t \leq t_{out} \\ (0, \varepsilon] & \text{otherwise} \end{cases},$$

with a later chosen $\varepsilon > 0$. Let $t_1, t_2 \in \mathbb{N}, t_1 \leq t_2$ be two arbitrary dates, where we want to calculate the gradient $\frac{\partial s_c(t_2)}{\partial y_{c_{in}}(t_1)}$. First, we want to show that the LSTM cell allows NVG for $t_1 = t_{in}$ and $t_{in} \leq t_2 \leq t_{out}$:
We have $y_\iota(t_1) \in [1-\varepsilon, 1)$ and $\forall t = t_{in}+1, \ldots, t_{out} : y_\phi(t) \in [1-\varepsilon, 1)$. Then, we can estimate the derivative from (12) and (14) by

$$\begin{aligned} \frac{\partial s_c(t_2)}{\partial y_{c_{in}}(t_1)} = \frac{\partial s_c(t_2)}{\partial s_c(t_1)} \frac{\partial s_c(t_1)}{\partial y_{c_{in}}(t_1)} &\underset{tr}{=} y_\iota(t_1) \prod_{t=t_1+1}^{t_2} y_\phi(t) \\ &\underset{tr}{\in} \left[(1-\varepsilon) \prod_{t=t_1+1}^{t_2} (1-\varepsilon), 1 \right) \subseteq \left[(1-\varepsilon)^{t_{out}-t_{in}+1}, 1 \right). \end{aligned}$$

To fulfill the equation for NVG we choose ε depending on δ such that

$$\begin{aligned} & 1-\delta \leq (1-\varepsilon)^{t_{out}-t_{in}+1} \\ \Leftrightarrow \quad & \varepsilon \leq 1 - (1-\delta)^{\frac{1}{t_{out}-t_{in}+1}} \end{aligned}$$

holds. Second, we have to show, that the derivative is in $[0, \delta]$, when $t_1 = t_{in}$ and $t_{in} \leq t_2 \leq t_{out}$ is not fulfilled.
In the case of $t_1 \neq t_{in}$ when $\varepsilon \leq \delta$ we can use the NEG which leads to

$$\frac{\partial s_c(t_2)}{\partial y_{c_{in}}(t_1)} = \underbrace{\frac{\partial s_c(t_2)}{\partial s_c(t_1)}}_{\underset{tr}{\in}[0,1]} \underbrace{\frac{\partial s_c(t_1)}{\partial y_{c_{in}}(t_1)}}_{\in(0,\varepsilon]} \subseteq [0, \varepsilon] \subseteq [0, \delta].$$

When $t_1 = t_{in}$ we have two cases: $t_2 < t_{in}$ or $t_2 > t_{out}$. For the case $t_2 < t_{in}$ the derivative is zero ($\subset [0, \delta]$), because the cell is causal. For $t_2 > t_{out}$ we can split the derivative at t_{out} and get

$$\begin{aligned} \frac{\partial s_c(t_2)}{\partial y_{c_{in}}(t_1)} &\underset{tr}{=} \underbrace{y_\iota(t_1) \prod_{t=t_1+1}^{t_{out}} y_\phi(t)}_{\in(0,1)} \prod_{t=t_{out}+1}^{t_2} \underbrace{y_\phi(t)}_{\in(0,\varepsilon]} \\ &\underset{tr}{\in} \left(0, \varepsilon^{t_2-t_{out}}\right] \subset [0, \varepsilon] \subseteq [0, \delta]. \end{aligned}$$

For $\varepsilon \leq \min\left\{\delta, 1-(1-\delta)^{\frac{1}{t_{out}-t_{in}+1}}\right\}$ the LSTM cell allows NVG.
COD: To prove that the LSTM cell has no COD, we show that there are gate activations

such that in Definition 5 we get $\delta_2 > \delta_1$. Therefore, we assume that all gate activations are arbitrary ($y_\gamma(t) \in (0,1)$), closed ($y_\gamma(t) \in (0, \varepsilon]$) or opened ($y_\gamma(t) \in [1-\varepsilon, 1)$) with a later chosen $\varepsilon > 0$. We take a look at the right side of (14). For $s_c(t) = 0$ we get $h_c'(s_c(t)) = 1$. In Definition 5 we have to satisfy $\exists \boldsymbol{y}_\Gamma(t) : \frac{\partial y_c(t)}{\partial s_c(t)} \in [0, \delta_2]$ an choose the OG $y_\omega(t) \in (0, \varepsilon]$ with

$$\varepsilon \leq \delta_2. \tag{15}$$

But then for $t' = 1, \ldots, t-1$ we can choose the IG and FG open with the same ε so that

$$y_\phi(t'), y_\iota(t') \in [1-\varepsilon, 1).$$

When for all time steps $t' = 1, \ldots, t$ there is a positive input $y_{c_{in}}(t') \in [c, 1), c \in (0,1) \subset \mathbb{R}$ and an internal state $s_c(t'-1) < c\frac{(1-\varepsilon)}{\varepsilon}$, the internal state is growing over time, because

$$\begin{aligned} s_c(t') &= y_{c_{in}}(t') y_\iota(t') + s_c(t'-1) y_\phi(t') \\ &\geq c(1-\varepsilon) + s_c(t'-1)(1-\varepsilon) \\ &\geq s_c(t'-1) + c(1-\varepsilon) - s_c(t'-1)\varepsilon \\ &> s_c(t'-1) + c(1-\varepsilon) - c\frac{(1-\varepsilon)}{\varepsilon}\varepsilon \\ &> s_c(t'-1). \end{aligned}$$

For large $s_c(t) \geq c\frac{(1-\varepsilon)}{\varepsilon} \gg 1$ we can estimate

$$\underbrace{\tanh(s_c(t))}_{\approx \exp(-2s_c(t))} \leq \exp(-s_c(t)) \leq \exp\left(-c\frac{(1-\varepsilon)}{\varepsilon}\right).$$

This yields in (14) to the bound

$$\begin{aligned} \left|\frac{\partial y_c(t)}{\partial s_c(t)}\right| &= \left|h_c'(s_c(t))\, y_\omega(t)\right| && (16) \\ &\leq \exp\left(-c\frac{(1-\varepsilon)}{\varepsilon}\right) && (17) \end{aligned}$$

so in Definition 5 we get

$$\delta_1 \leq \exp\left(-c\frac{(1-\varepsilon)}{\varepsilon}\right). \tag{18}$$

But when we combine (15), (18) and the restriction in Definition 5, we have

$$\varepsilon \leq \delta_2 < \delta_1 \leq \exp\left(-c\frac{(1-\varepsilon)}{\varepsilon}\right),$$

but there exist ε, c, such that the inequality is not fulfilled, which is a contradiction.
Summarized, the 1D LSTM cell allows an NVG and has an NEG, but does not allow COD. ■

A.2 Proof of 14

Proof Let c be an MD LSTM cell of dimension D, $\boldsymbol{p},\boldsymbol{p}_1,\boldsymbol{p}_2,\boldsymbol{p}_{in},\boldsymbol{p}_{out} \in \mathbb{N}^D$,$\boldsymbol{p}_{in} \leq \boldsymbol{p}_{out}$ arbitrary dates and $h_c = \tanh$ the sigmoid function. Besides $\varepsilon > 0$ is a later chosen value. In the first step we want to show that there are activations of the forget gates, so that

$$\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} \begin{cases} \left[(1-\varepsilon)^{\|\boldsymbol{p}-\boldsymbol{p}_{in}\|_1}, 1\right] & \text{for} \quad \boldsymbol{p}_{in} \leq \boldsymbol{p} \leq \boldsymbol{p}_{out} \\ [0, D\varepsilon] & \text{otherwise} \end{cases} \tag{19}$$

is fulfilled. The prove is done using induction over $k = \|\boldsymbol{p}-\boldsymbol{p}_{in}\|_1$ with $\boldsymbol{p} \geq \boldsymbol{p}_{in}$. The base $k = 0$ is clear. Let be $k \geq 1$. We define

$$P_{\boldsymbol{p}} := \left\{d \in \{1,\dots,D\} \mid \boldsymbol{p}_d^- \geq \boldsymbol{p}_{in}\right\}$$

the set of dimensions d, in which are $\boldsymbol{p}_{in}$-$\boldsymbol{p}_d^-$-paths. Note, that this set cannot be empty, because $\boldsymbol{p} > \boldsymbol{p}_{in}$ for $k \geq 1$. When we have a dimension $d \in P_{\boldsymbol{p}}$ then $\left\|\boldsymbol{p}_d^- - \boldsymbol{p}_{in}\right\|_1 = k-1$ and we assume

$$\frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} \left[(1-\varepsilon)^{\left\|\boldsymbol{p}_d^- - \boldsymbol{p}_{in}\right\|_1}, 1\right] = \left[(1-\varepsilon)^{k-1}, 1\right]. \tag{20}$$

Then we choose the activations of the FG to be

$$y_{\phi,d}^{\boldsymbol{p}} \in \begin{cases} \left[\frac{1-\varepsilon}{|P_{\boldsymbol{p}}|}, \frac{1}{|P_{\boldsymbol{p}}|}\right) & \text{for} \quad d \in P_{\boldsymbol{p}} \text{ and } \boldsymbol{p}_{in} < \boldsymbol{p} \leq \boldsymbol{p}_{out} \\ [0, \varepsilon] & \text{otherwise} \end{cases}. \tag{21}$$

Then we can estimate the derivative for $\boldsymbol{p}_{in} \leq \boldsymbol{p} \leq \boldsymbol{p}_{out}$ using (20) and (21) to

$$\begin{aligned} \frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{=} \sum_{d \in P_{\boldsymbol{p}}} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} y_{\phi,d}^{\boldsymbol{p}} \in \left[\sum_{d \in P_{\boldsymbol{p}}} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \frac{1-\varepsilon}{|P_{\boldsymbol{p}}|}, \sum_{d \in P_{\boldsymbol{p}}} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \frac{1}{|P_{\boldsymbol{p}}|}\right) \\ \Rightarrow \frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{\in} \left[|P_{\boldsymbol{p}}|\,(1-\varepsilon)^{k-1}\,\frac{1-\varepsilon}{|P_{\boldsymbol{p}}|}, |P_{\boldsymbol{p}}|\,\frac{1}{|P_{\boldsymbol{p}}|}\right) \\ \Leftrightarrow \frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{\in} \left[(1-\varepsilon)^{\|\boldsymbol{p}-\boldsymbol{p}_{in}\|_1}, 1\right), \end{aligned} \tag{22}$$

so (19) is fulfilled for $\boldsymbol{p}_{in} \leq \boldsymbol{p} \leq \boldsymbol{p}_{out}$.
If we have $\boldsymbol{p} < \boldsymbol{p}_{in}$ in (19), the derivative is 0, because we have a causal system.
For $\boldsymbol{p} > \boldsymbol{p}_{out}$ in (19), we choose $\varepsilon \leq \frac{1}{D} \leq \frac{1}{|P_{\boldsymbol{p}}|}$ in (21) to ensure $\forall \boldsymbol{p} \in \mathbb{N}^D$:$\left|\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}}\right| \leq 1$ (see (22)) and we get

$$\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{=} \sum_{d=1,\dots,D} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} y_{\phi,d}^{\boldsymbol{p}} \in \left(0, D\varepsilon \max_{d=1,\dots,D} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}}\right] \subseteq (0, D\varepsilon], \tag{23}$$

and (19) is fulfilled.
In the second step let $\boldsymbol{p}_1 \leq \boldsymbol{p}_2$ be the date, for which we want to calculate the truncated gradient $\frac{\partial s_c^{\boldsymbol{p}_2}}{\partial y_{c_{in}}^{\boldsymbol{p}_1}}$. We choose the IG activation as

$$y_t^{\boldsymbol{p}} \in \begin{cases} [1-\varepsilon, 1) & \text{if} \quad \boldsymbol{p} = \boldsymbol{p}_{in} \\ (0, \varepsilon] & \text{otherwise} \end{cases} \tag{24}$$

and we get $\frac{\partial s_c^{\boldsymbol{p}}}{\partial y_{c_{in}}^{\boldsymbol{p}}} = y_t^{\boldsymbol{p}}$. Using (22), (23) and (24), we can estimate the partial derivative by

$$\begin{aligned} &\frac{\partial s_c^{\boldsymbol{p}_2}}{\partial y_{c_{in}}^{\boldsymbol{p}_1}} = \frac{\partial s_c^{\boldsymbol{p}_2}}{\partial s_c^{\boldsymbol{p}_1}} \frac{\partial s_c^{\boldsymbol{p}_1}}{\partial y_{c_{in}}^{\boldsymbol{p}_1}} \\ \Rightarrow &\frac{\partial s_c^{\boldsymbol{p}_2}}{\partial s_c^{\boldsymbol{p}_1}} \underset{tr}{\in} \begin{cases} \left[(1-\varepsilon)(1-\varepsilon)^{\|\boldsymbol{p}_2-\boldsymbol{p}_{in}\|_1}, 1\right] & \text{for} \quad \boldsymbol{p}_1 = \boldsymbol{p}_{in} \text{ and } \boldsymbol{p}_{in} \leq \boldsymbol{p}_2 \leq \boldsymbol{p}_{out} \\ [0, D\varepsilon] & \text{otherwise} \end{cases}. \end{aligned}$$

and setting

$$\varepsilon := \min\left\{\frac{\delta}{D}, 1-(1-\delta)^{\frac{1}{\|\boldsymbol{p}_{in}-\boldsymbol{p}_{out}\|_1+1}}\right\}$$

the conditions of Definition 11 are fulfilled. ■

A.3 Proof of 15

Proof Let c be an MD cell of dimension D with the internal state s_c and $\boldsymbol{p}_{in},\boldsymbol{p}_k \in \mathbb{N}^D$,$\boldsymbol{p}_{in} \leq \boldsymbol{p}_k$ two dates. Let $\boldsymbol{p}_k$ be a date k steps further in each dimension than a fixed date $\boldsymbol{p}_{in}$. So the distance between them is $\|\boldsymbol{p}_{in}-\boldsymbol{p}_k\|_1 = Dk$. Let Π be the set of all $\boldsymbol{p}_{in}$-$\boldsymbol{p}_k$-paths, then there exist $|\Pi| = \#\{\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\}$ paths (see Definition 8). We assume

$$y_{\phi,d}^{\boldsymbol{p}} \in [\varepsilon, 1-\varepsilon]$$

with $\varepsilon \in (0, 0.5)$ and we can estimate the partial derivative, using the truncated gradient, with

$$\begin{aligned} \frac{\partial s_c^{\boldsymbol{p}_k}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{=} \sum_{\pi \in \Pi} \prod_{i=1}^{k} y_{\phi,d}^{\pi_i} \\ &\underset{tr}{\in} \left[\varepsilon^k \#\{\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\}, (1-\varepsilon)^k \#\{\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\}\right]. \end{aligned}$$

For $D = 1$ we get $|\Pi| = 1$ and the cell has a NGEC. When $D \geq 2$ we can count the number of paths using the Stirling's approximation and we can estimate the number of paths with

$$\#\{\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\} = \frac{\left(\sum_{i=1}^{D}\left(\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\right)_i\right)!}{\prod_{i=1}^{D}\left(\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\right)_i!} = \frac{(Dk)!}{(k!)^D} \xrightarrow{k \gg 1} \frac{\sqrt{2\pi Dk}\left(\frac{Dk}{e}\right)^{Dk}}{\left(\sqrt{2\pi k}\left(\frac{k}{e}\right)^k\right)^D} = \frac{\sqrt{D}D^{Dk}}{\sqrt{2\pi k}^{D-1}}.$$

When we combine it with the FG activations we can estimate the derivative for great k with the Stirling's approximation and get

$$\begin{aligned} \frac{\partial s_c^{\boldsymbol{p}_k}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{\in} \left[\varepsilon^{Dk}\#\{\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\}, (1-\varepsilon)^{Dk}\#\{\overrightarrow{\boldsymbol{p}_{in}\boldsymbol{p}_k}\}\right] \\ &\overset{k \gg 1}{\underset{tr}{\in}} \left[\frac{\sqrt{D}}{\sqrt{2\pi k}^{D-1}}(D\varepsilon)^{Dk}, \frac{\sqrt{D}}{\sqrt{2\pi k}^{D-1}}(D(1-\varepsilon))^{Dk}\right]. \end{aligned} \tag{25}$$

The upper bound of this interval can grow for great k, if $[D\,(1-\varepsilon)] > 1$ and this is the case for $D \geq 2$. So the MD LSTM cell can have an exploding gradient for $D \geq 2$. When the weights to the FGs are initialized with small values, we have $y^{\boldsymbol{p}}_{\phi,d} \approx 0.5$. Then we have an exploding gradient when $D \geq 3$, when the training is starting. In the worst case we have $y^{\boldsymbol{p}}_{\phi,d} \approx 1$ and the derivative in (25) goes for great k to

$$\frac{\partial s_c^{\boldsymbol{p}_k}}{\partial s_c^{\boldsymbol{p}_{in}}} \approx \frac{\sqrt{D}}{\sqrt{2\pi}^{D-1}} k^{\frac{1-D}{2}} (D)^{Dk}.$$

■

A.4 Proof of 17

Proof Let c be an MD LSTM Stable cell of dimension $D \geq 2$ (for $D = 1$ the proof is equivalent to the 1D case of the LSTM cell), $\boldsymbol{p}, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_{in}, \boldsymbol{p}_{out} \in \mathbb{N}^D, \boldsymbol{p}_{in} \leq \boldsymbol{p}_{out}$ arbitrary dates and $h_c = \tanh$ the sigmoid function. Besides $\varepsilon > 0$ is a later chosen value.
In the first step we want to show that there are activations of the forget gates, so that

$$\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} \begin{cases} \left[(1-(D-1)\varepsilon)^{2\|\boldsymbol{p}-\boldsymbol{p}_{in}\|_1}, 1\right] & \text{for} \quad \boldsymbol{p}_{in} \leq \boldsymbol{p} \leq \boldsymbol{p}_{out} \\ [0, \varepsilon] & \text{otherwise} \end{cases} \tag{26}$$

is fulfilled. The prove is done using induction over $k = \|\boldsymbol{p}\_\boldsymbol{p}_{in}\|_1$. The base $k = 0$ is clear. Let be $k \geq 1$. We define

$$P_{\boldsymbol{p}} := \left\{ d \in \{1, \ldots, D\} \mid \boldsymbol{p}_d^- \geq \boldsymbol{p}_{in} \right\}$$

the set of dimensions d, in which are $\boldsymbol{p}_{in}$-$\boldsymbol{p}_d^-$-paths. Note, that this set cannot be empty, because $\boldsymbol{p} > \boldsymbol{p}_{in}$ for $k \geq 1$. When we have a dimension $d \in P_{\boldsymbol{p}}$ then $\left\|\boldsymbol{p}_d^- - \boldsymbol{p}_{in}\right\|_1 = k-1$ and we assume

$$\frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} \left[(1-(D-1)\varepsilon)^{2\left\|\boldsymbol{p}_d^- - \boldsymbol{p}_{in}\right\|_1}, 1\right] = \left[(1-(D-1)\varepsilon)^{2(k-1)}, 1\right]. \tag{27}$$

When we choose the activations of the LGs to be

$$y^{\boldsymbol{p}}_{\lambda,d} \in \begin{cases} [1-\varepsilon, 1) & \text{for} \quad d \in P_{\boldsymbol{p}} \text{ and } \boldsymbol{p}_{in} < \boldsymbol{p} \leq \boldsymbol{p}_{out} \\ (0, \varepsilon] & \text{otherwise} \end{cases},$$

we can estimate $\frac{\sum_{d\in P_{\boldsymbol{p}}} y^{\boldsymbol{p}}_{\lambda,d}}{\sum_{d'=1}^{D} y^{\boldsymbol{p}}_{\lambda,d'}} \in (1-(D-1)\varepsilon, 1]$, because

$$\begin{aligned} 1 \geq \frac{\sum_{d\in P_{\boldsymbol{p}}} y^{\boldsymbol{p}}_{\lambda,d}}{\sum_{d'=1}^{D} y^{\boldsymbol{p}}_{\lambda,d'}} &= \frac{\sum_{d\in P_{\boldsymbol{p}}} y^{\boldsymbol{p}}_{\lambda,d}}{\sum_{d\in P_{\boldsymbol{p}}} y^{\boldsymbol{p}}_{\lambda,d} + \sum_{d\in\{1,..,D\}\setminus P_{\boldsymbol{p}}} y^{\boldsymbol{p}}_{\lambda,d'}} \\ &\geq \frac{|P_{\boldsymbol{p}}|\,(1-\varepsilon)}{|P_{\boldsymbol{p}}|\,(1-\varepsilon) + \underbrace{(D-|P_{\boldsymbol{p}}|)}_{\leq D-1}\varepsilon} \\ &\geq \frac{|P_{\boldsymbol{p}}|\,(1-(D-1)\varepsilon)}{|P_{\boldsymbol{p}}|\,(1-(D-1)\varepsilon) + (D-1)\,\varepsilon} \\ &\geq (1-(D-1)\varepsilon)\frac{|P_{\boldsymbol{p}}|}{|P_{\boldsymbol{p}}| - \varepsilon(D-1)\,(|P_{\boldsymbol{p}}|-1)} \\ &\geq (1-(D-1)\varepsilon). \end{aligned} \tag{28}$$

Setting the FG to

$$y^{\boldsymbol{p}}_{\phi} \in \begin{cases} [1-\varepsilon, 1) & \text{for} \quad \boldsymbol{p}_{in} < \boldsymbol{p} \leq \boldsymbol{p}_{out} \\ (0, \varepsilon] & \text{otherwise} \end{cases} \tag{29}$$

we can estimate the derivative for $\boldsymbol{p}_{in} \leq \boldsymbol{p} \leq \boldsymbol{p}_{out}$ using (27),(28) and (29) to

$$\begin{aligned} \frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{=} y^{\boldsymbol{p}}_{\phi} \left(\sum_{d\in P_{\boldsymbol{p}}} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \frac{y^{\boldsymbol{p}}_{\lambda,d}}{\sum_{d'=1}^{D} y^{\boldsymbol{p}}_{\lambda,d'}} + \underbrace{\sum_{d\in\{1,..,D\}\setminus P_{\boldsymbol{p}}} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \frac{y^{\boldsymbol{p}}_{\lambda,d}}{\sum_{d'=1}^{D} y^{\boldsymbol{p}}_{\lambda,d'}}}_{=0} \right) \\ &\underset{tr}{\in} \left((1-\varepsilon)\,(1-(D-1)\varepsilon)^{2(k-1)}\,(1-(D-1)\varepsilon), 1 \right) \\ \Rightarrow \quad \frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{\in} \left((1-(D-1)\varepsilon)^{2k}, 1 \right) \end{aligned} \tag{30}$$

so (26) is fulfilled for $\boldsymbol{p}_{in} \leq \boldsymbol{p} \leq \boldsymbol{p}_{out}$.
If we have $\boldsymbol{p} < \boldsymbol{p}_{in}$ in (26), the derivative is 0, because we have a causal system.
For $\boldsymbol{p} > \boldsymbol{p}_{out}$ the FG is closed (see (29)), and using the upper bounds of (27) and (28) we get

$$\begin{aligned} \frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} &\underset{tr}{=} y^{\boldsymbol{p}}_{\phi} \left(\sum_{d=1}^{D} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \frac{y^{\boldsymbol{p}}_{\lambda,d}}{\sum_{d'=1}^{D} y^{\boldsymbol{p}}_{\lambda,d'}} \right) \\ &\underset{tr}{\in} (0, \varepsilon] \end{aligned} \tag{31}$$

and (26) is fulfilled.
In the second step let $\boldsymbol{p}_1 \leq \boldsymbol{p}_2$ be the date, for which we want to calculate the truncated gradient $\frac{\partial s_c^{\boldsymbol{p}_2}}{\partial y^{\boldsymbol{p}_1}_{c_{in}}}$. We choose the IG activation as

$$y^{\boldsymbol{p}}_{\iota} \in \begin{cases} [1-\varepsilon, 1) & \text{if} \quad \boldsymbol{p} = \boldsymbol{p}_{in} \\ (0, \varepsilon] & \text{otherwise} \end{cases} \tag{32}$$

and we get $\frac{\partial s_c^{\boldsymbol{p}}}{\partial y_{c_{in}}^{\boldsymbol{p}}} = y_t^{\boldsymbol{p}}$. Using (30), (31) and (32), we can estimate the partial derivative by

$$\frac{\partial s_c^{\boldsymbol{p}_2}}{\partial y_{c_{in}}^{\boldsymbol{p}_1}} = \frac{\partial s_c^{\boldsymbol{p}_2}}{\partial s_c^{\boldsymbol{p}_1}} \frac{\partial s_c^{\boldsymbol{p}_1}}{\partial y_{c_{in}}^{\boldsymbol{p}_1}}$$
$$\Rightarrow \frac{\partial s_c^{\boldsymbol{p}_2}}{\partial s_c^{\boldsymbol{p}_1}} \underset{tr}{\in} \begin{cases} \left[(1-\varepsilon)(1-(D-1)\varepsilon)^{2\|\boldsymbol{p}_2-\boldsymbol{p}_{in}\|_1}, 1\right] & \text{for} \quad \boldsymbol{p}_1 = \boldsymbol{p}_{in} \text{ and } \boldsymbol{p}_{in} \leq \boldsymbol{p}_2 \leq \boldsymbol{p}_{out} \\ [0, \varepsilon] & \text{otherwise} \end{cases}.$$

and setting

$$\varepsilon := \min\left\{\delta, \left(1-(1-\delta)^{\frac{1}{2\|\boldsymbol{p}_{in}-\boldsymbol{p}_{out}\|_1+1}}\right)\frac{1}{D-1}\right\}$$

the conditions of Definition 11 are fulfilled. ■

A.5 Proof of 18

Proof Let c be a MD LSTM Stable cell of dimension D with the internal state s_c and $\boldsymbol{p}_{in}, \boldsymbol{p} \in \mathbb{N}^D, \boldsymbol{p}_{in} \leq \boldsymbol{p}$ two arbitrary dates and $\|\boldsymbol{p}_{in} - \boldsymbol{p}\|_1 = k$. Let all gate activations be arbitrary in $[0,1]$. We show that

$$\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} [0,1] \tag{33}$$

is fulfilled $\forall k \in \mathbb{N}$ using induction over k. For the base case $k=0$ we get $\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} = \frac{\partial s_c^{\boldsymbol{p}_{in}}}{\partial s_c^{\boldsymbol{p}_{in}}} = 1$. Let (33) be fulfilled for $k-1$. That means if $\boldsymbol{p}_d^- \geq \boldsymbol{p}_{in}$ we have $\left\|\boldsymbol{p}_d^- - \boldsymbol{p}_{in}\right\|_1 = k-1$ and this leads to $\frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \underset{tr}{\in} [0,1]$. If $\boldsymbol{p}_d^- \not\geq \boldsymbol{p}_{in}$ then there is no $\boldsymbol{p}_{in}$-$\boldsymbol{p}_d^-$-path and we have $\frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} = 0$ for this dimension. Then we can calculate the derivative

$$0\frac{\partial s_c^{\boldsymbol{p}}}{\partial s_c^{\boldsymbol{p}_{in}}} = y_\phi^{\boldsymbol{p}} \sum_{d=1}^{D} \frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \frac{y_{\lambda,d}^{\boldsymbol{p}}}{\sum_{d'=1}^{D} y_{\lambda,d'}^{\boldsymbol{p}}} \in \left[0, \max_{d \in P_{\boldsymbol{p}}}\left\{\frac{\partial s_c^{\boldsymbol{p}_d^-}}{\partial s_c^{\boldsymbol{p}_{in}}} \underbrace{\frac{y_{\lambda,d}^{\boldsymbol{p}}}{\sum_{d'=1}^{D} y_{\lambda,d'}^{\boldsymbol{p}}}}_{\leq 1}\right\}\right]$$
$$\underset{tr}{\in} [0,1],$$

which gives us the desired interval. ■

A.6 Proof of 20

Proof
NEG: The cell has an NEG, because all gates have the same bounds as the MD Stable cell.
NVG: To prove the NVG, we use the proof of Theorem 17. The difference between the MD Stable cell and the MD Leaky cell is that the activations of the FG and IG are dependent

on each other for the Leaky cell. Let $\boldsymbol{p}_{in}, \boldsymbol{p} \in \mathbb{N}^D, \boldsymbol{p}_{in} \leq \boldsymbol{p}$ be two arbitrary dates like in Theorem 17. The IG has just the a restriction that for $\boldsymbol{p} = \boldsymbol{p}_{in}$ it has to hold $y_t^{\boldsymbol{p}} \in [1-\varepsilon, 1)$. Here, the FG can have an arbitrary activation, so we chose $y_\phi^{\boldsymbol{p}} = 1 - y_t^{\boldsymbol{p}}$. For all $\boldsymbol{p} > \boldsymbol{p}_{in}$ the FG have to be in the ranges, shown in (29), while the IG has no restriction and we choose $y_t^{\boldsymbol{p}} = 1 - y_\phi^{\boldsymbol{p}}$, so the MD Leaky cell has the NVG.
COD: The proof that the MD Leaky cell allows COD can be done by estimating the bounds of $s_c^{\boldsymbol{p}}$. From the update equations of the cell we get

$$\left|s_c^{\boldsymbol{p}^-}\right| \leq \max_{i=1,\ldots,D}\left|s_c^{\boldsymbol{p}_d^-}\right|.$$

Now we can estimate the internal state using the ranges $y_{c_{in}}^{\boldsymbol{p}} \in [-1,1]$, recursion over $\boldsymbol{p}$

$$|s_c^{\boldsymbol{p}}| = \left|\left(1-y_\phi^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + y_\phi^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}\right| \leq \max\left\{\left|y_{c_{in}}^{\boldsymbol{p}}\right|, \left|s_c^{\boldsymbol{p}_1^-}\right|, \ldots, \left|s_c^{\boldsymbol{p}_D^-}\right|\right\} \leq \max_{\boldsymbol{q}<\boldsymbol{p}}\left\{\left|y_{c_{in}}^{\boldsymbol{q}}\right|\right\} \leq 1$$

and get $s_c^{\boldsymbol{p}} \in [-1,1]$. To fulfill the derivatives in Definition 12, for δ_1 we choose $y_\omega^{\boldsymbol{p}} \in [1-\varepsilon, 1)$ and get

$$\delta_1 \leq \min_{s_c^{\boldsymbol{p}}}\left\{h_c'\left(s_c^{\boldsymbol{p}}\right)\right\}(1-\varepsilon) = h_c'(1)(1-\varepsilon). \tag{34}$$

For δ_2 we choose $y_\omega^{\boldsymbol{p}} \in (0, \varepsilon]$ and get

$$\delta_2 \geq \max_{s_c^{\boldsymbol{p}}}\left\{h_c'\left(s_c^{\boldsymbol{p}}\right)\right\}\varepsilon = \varepsilon. \tag{35}$$

To fulfill the derivatives in Definition 12 we use (34), (35) and $h_c'(1) > \frac{1}{3}$ and with

$$\varepsilon \leq \delta_2 < \delta_1 \leq h_c'(1)(1-\varepsilon)$$
$$\Rightarrow \varepsilon \leq \frac{1}{4} < \frac{h_c'(1)}{h_c'(1)+1}$$

the COD is proven. ■

Appendix B. Theory to Create First Order MD Cells

If one wants to take a closer look at the theory of linear shift invariant (LSI)-systems and their frequency analysis and analyse a first order LSI-system regarding its free selectable parameters using the $\mathcal{F}$- and $\mathcal{Z}$-transform, it is highly recommended to be familar with these theories (for a good overview and more details see Poularikas, 2000; Schlichthärle, 2000). Adding the knowledge of reducing the MD case to the 1D case (see Section 5) we create new cell types for the MD case.

B.1 Analysing a First Order LSI-System

The update equations of a first order LSI-system with one input u, one internal state x and one output y can be written as

$$x[n] = h_1(u[n], x[n-1]) = \alpha_0 u[n] + \alpha_1 x[n-1], \tag{36}$$
$$y[n] = h_2(x[n], x[n-1]) = b_0 x[n] + b_1 x[n-1] \tag{37}$$

with the free selectable coefficients $\alpha_0, \alpha_1, b_0, b_1 \in \mathbb{R}$. Let $U(z) = \mathcal{Z}\{u[n]\}$ be the $\mathcal{Z}$-transformed signal of $u[n]$ and $X(z), Y(z)$ respectively. Then we can write the so called *transfer functions*

$$H_1(z) := \frac{X(z)}{U(z)} = \frac{\alpha_0}{1 - \alpha_1 z^{-1}},$$
$$H_2(z) := \frac{Y(z)}{X(z)} = b_0 + b_1 z^{-1},$$
$$H(z) := \frac{Y(z)}{U(z)} = H_2(z) H_1(z).$$

To analyse (36) and (37) according their *frequency response* we use the relationship between the $\mathcal{F}$-transform and the $\mathcal{Z}$-transform:

Remark 22 *Let $u[n] = e^{j\omega n}$ be a harmonic input sequence with the imaginary number $j^2 = -1$ and $H(z) = \frac{Y(z)}{U(z)}$ be a transfer functions of an LSI-system. When the poles of $H(z)$ are inside the circle $|z| = 1$, we can change from $\mathcal{Z}$- to $\mathcal{F}$-transform using the substitution*

$$H(\omega) = H(z)\Big|_{z=e^{j\omega}}$$

with the harmonic sequence $y[n] = H(\omega)u[n] = H(\omega)e^{j\omega n}$ with the same frequency ω but with a different amplitude and a different phase dependent on the frequency ω.

We only want to analyse the amplitude of this harmonic sequence

$$|y[n]| = \left|H(z)e^{j\omega n}\right| = |H(z)| = |H_2(z)|\,|H_1(z)|$$

and do that by analysing both transfer functions $H_1(z)$ and $H_2(z)$ separately.
The amplitude of $H_1(\omega) = H_1(z)|_{z=e^{j\omega}}$ is calculated by

$$|H_1(\omega)| = \frac{|\alpha_0|}{\sqrt{(1 - \alpha_1 \cos(\omega))^2 + \alpha_1^2 \sin^2(\omega)}}.$$

Like mentioned before, in many tasks, the information signal has a low frequency. To have the largest amplitude at $\omega = 0$ we have to choose $\alpha_1 \geq 0$. As mentioned in Remark 22 the poles of $H_1(z) = \frac{\alpha_0}{1-\alpha_1 z^{-1}} = \frac{z\alpha_0}{z-\alpha_1}$ have to be in the circle $|z| = 1$, so we have the additional constraint $|\alpha_1| < 1$. This leads to the bounds $\alpha_1 \in [0, 1)$. But for $\alpha_1 \to 1$ we

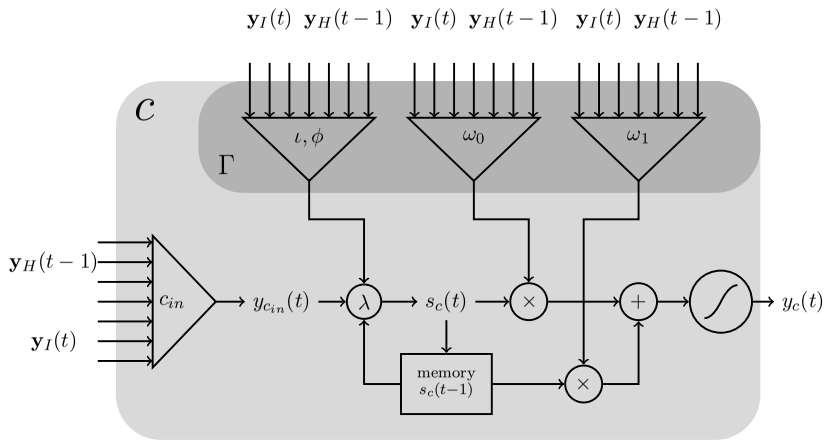


Figure 5: Schematic diagram of a one-dimensional LeakyLP cell: The internal state is a convex combination of the new input c_{in} and the previous state $s_c(t-1)$. The previous state $s_c(t-1)$ and the current state $s_c(t)$ are gated (ω_0 and ω_1) and accumulated afterwards. The output is squashed by tanh into the interval $[-1, 1]$.

have $H_1(0) \to \infty$, so we have to choose α_0 dependent on α_1. We set a maximum gain of $\max_\omega |H_1(\omega)| = |H_1(0)| = 1$, so we get the constraint

$$|\alpha_0| \leq 1 - \alpha_1. \tag{38}$$

In the same way we analyse $H_2(z)$:

$$|H_2(\omega)| = \left|b_0 + b_1 e^{-j\omega}\right| = \sqrt{(b_0 + b_1 \cos(\omega))^2 + b_1^2 \sin^2(\omega)}$$

To get the maximal gain at low frequency the parameters b_0 and b_1 must have the same sign.

B.2 Creating a First Order Cell

With these constraints for the parameters we now can define a new cell type. The parameters $\alpha_0, \alpha_1, b_0, b_1$ should be activations of gates like in LSTM cells. We have to find the right activation functions to fulfill the inequalities above. Using the weight-space symmetries in a network with at least one hidden layer (Bishop, 2006, 5.1.1), without loss of generality we set $\alpha_0, \alpha_1, b_0, b_1 \geq 0$. To fulfill the bounds for H_1, we set α_1 as activation of a gate with activation function f_{log}. So we have $\alpha_1 \in (0, 1)$. This is comparable with the FG in the previous sections. To select the α_0 we choose $\alpha_0 := 1 - \alpha_1$ (see (38)). So the value of α_0 is comparable with the activation of the IG. For H_2 we set both values b_0, b_1 as activations of

a gate with activation function f_{log} which leads to $\max_\omega |H_2(\omega)| = \max\{b_0 + b_1\} = 2$, so the amplitude response is bounded by 2.

With these bounds we can define a cell with a cell input $y^{\mathbf{p}}_{c_{in}} = u[n]$, a previous internal state $s^{\mathbf{p}^-}_c = x[n-1]$, an internal state $s^{\mathbf{p}}_c = x[n]$ and a cell output $y^{\mathbf{p}}_c = y[n]$. We substitute the coefficients by time dependent gate activations

$$\begin{array}{ll} \alpha_0 := 1 - y^{\mathbf{p}}_\phi = y^{\mathbf{p}}_\imath & \text{IG} \\ \alpha_1 := y^{\mathbf{p}}_\phi & \text{FG} \\ b_0 := y^{\mathbf{p}}_{\omega_0} & \text{OG} \\ b_1 := y^{\mathbf{p}}_{\omega_1} & \text{OG of the previous internal state} \end{array}$$

which leads to the transfer functions

$$\begin{aligned} H_1^{y^{\mathbf{p}}_\phi}(z) &= \frac{\alpha_0}{1 - \alpha_1 z^{-1}} = \frac{1 - y^{\mathbf{p}}_\phi}{1 - y^{\mathbf{p}}_\phi z^{-1}}, \\ H_2^{y^{\mathbf{p}}_{\omega_0}; y^{\mathbf{p}}_{\omega_1}}(z) &= b_0 + b_1 z^{-1} = y^{\mathbf{p}}_{\omega_0} + y^{\mathbf{p}}_{\omega_1} z^{-1}, \\ H(z) = H^{y^{\mathbf{p}}_\phi; y^{\mathbf{p}}_{\omega_0}; y^{\mathbf{p}}_{\omega_1}}(z) &= \frac{Y(z)}{U(z)} = \alpha_0 \frac{b_0 + b_1 z^{-1}}{1 - \alpha_1 z^{-1}} = \frac{1 - y^{\mathbf{p}}_\phi}{1 - y^{\mathbf{p}}_\phi z^{-1}} \left(y^{\mathbf{p}}_{\omega_0} + y^{\mathbf{p}}_{\omega_1} z^{-1}\right). \end{aligned} \tag{39}$$

and the update equations

$$\begin{aligned} x[n] =& \alpha_0 u[n] + \alpha_1 x[n-1] && \Leftrightarrow s^{\mathbf{p}}_c = (1 - y^{\mathbf{p}}_\phi) y^{\mathbf{p}}_{c_{in}} + y^{\mathbf{p}}_\phi s^{\mathbf{p}^-}_c, \\ y[n] =& b_0 x[n] + b_1 x[n-1] && \Leftrightarrow y^{\mathbf{p}}_c = y^{\mathbf{p}}_{\omega_0} s^{\mathbf{p}}_c + y^{\mathbf{p}}_{\omega_1} s^{\mathbf{p}^-}_c. \end{aligned} \tag{40}$$

The output of the cell is already bounded in $[-2, 2]$, but to fulfill Definition 9 we change (40) to

$$y^{\mathbf{p}}_c = h_c \left(y^{\mathbf{p}}_{\omega_0} s^{\mathbf{p}}_c + y^{\mathbf{p}}_{\omega_1} s^{\mathbf{p}^-}_c \right) \tag{41}$$

with $h_c = \tanh$ to ensure $y^{\mathbf{p}}_c \in [-1, 1]$. This additional non-linearity is not necessary but leads to a better performance. This new cell type called *MD Leaky lowpass (LeakyLP) cell* is defined in Definition 21. A block diagram of a 1D LeakyLP cell is shown in Figure 5 and different frequency responses in Figure 6.

B.3 General First Order MD Cells

With the theory of this section we can easily create new cell types. In general, a cell has a number of gates $\gamma_1, \gamma_2, \ldots \in \Gamma_c$. For $D = 1$ a previous state $s^{\mathbf{p}^-}_c$ is given directly. Otherwise the previous state is calculated as trainable convex combination of D previous states, like described in Section 5. In Table 7 cell layouts are depicted whereby type A is the cell developed in Section 7 (compare to (39)). For the other types we briefly want to describe the main ideas.

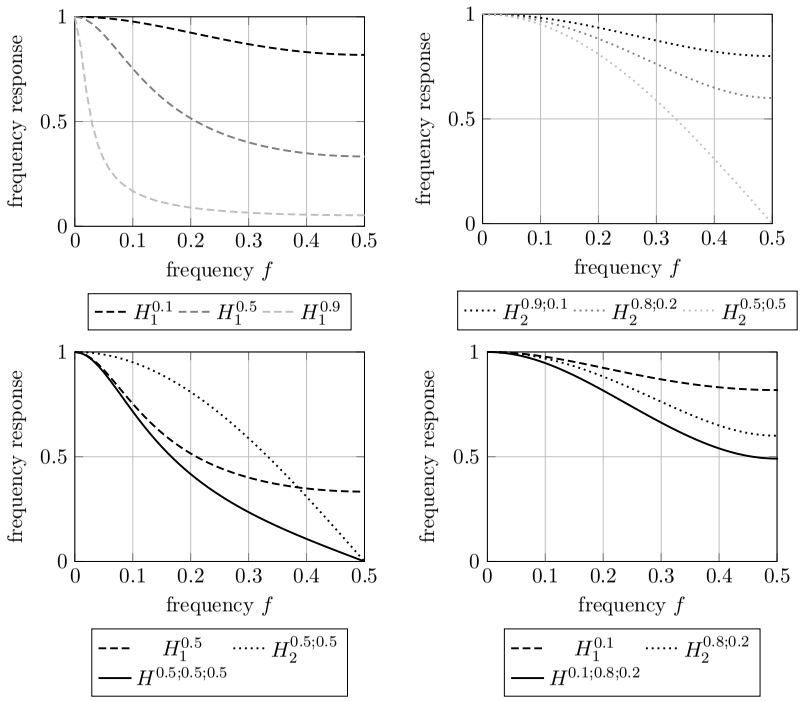


Figure 6: Frequency response of H_1 (dashed), H_2 (dotted) and H (solid) for special parameters. Top-left: The frequency response of an IIR filter is able to block even low frequency signals, but it cannot be zero at $f = 0.5$. Top-right: The frequency response of an FIR filter cannot be lower than the lightgray dotted line, but for $f = 0.5$ it can be zero. Bottom: When these both filters are concatenated, the resulting frequency response can combine the benefits of each filter.

B.3.1 The MD Butterworth Lowpass Filter

The cell of type B is a special case of the LeakyLP cell. When we set $y^{\mathbf{p}}_{\omega_0} = y^{\mathbf{p}}_{\omega_1} = 0.5$ there is a direct relation between the cutoff frequency of a discrete Butterworth lowpass filter and the activation of $y^{\mathbf{p}}_\phi$: Let f_{cutoff} be the frequency, where amplitude response is reduced to

$\frac{1}{\sqrt{2}}$ of the maximal gain. We can calculate f_{cutoff} by

$$f_{\text{cutoff}} = \frac{1}{\pi} \arctan\left(\frac{1-y_\phi^{\boldsymbol{p}}}{1+y_\phi^{\boldsymbol{p}}}\right) \tag{42}$$
$$\Leftrightarrow y_\phi^{\boldsymbol{p}} = \frac{1+\tan(\pi f_{\text{cutoff}})}{1-\tan(\pi f_{\text{cutoff}})}$$

with the bounds $f_{\text{cutoff}} \in (0, 0.5)$ and $y_\phi^{\boldsymbol{p}} \in (-1, 1)$ (for more details see Schlichthärle, 2000, 2.2;6.4.2). For $y_\phi^{\boldsymbol{p}} \in (0, 1)$ we get $f_{\text{cutoff}} \in (0, 0.25)$. In Figure 7 (left) we can see, that even for a negative value of $y_\phi^{\boldsymbol{p}}$ and a highpass characteristic of $H_1(z)$ the impulse response $H(z)$ has a lowpass characteristic.

B.3.2 Adding an Additional State Gate

In B.2 we fulfilled (38) for the MD LeakyLP cell by setting $\alpha_0 := 1 - \alpha_1$, so α_0 is directly connected with α_1. Another solution would be to add an additional value $\gamma \in (0, 1)$ and choose $\alpha_0 := \gamma(1 - \alpha_1)$. So we can extend the MD LeakyLP cell by adding an additional gate γ_4 for the previous state (see type C). Unfortunately this does not lead to a better performance and one more gate has to be calculated.

B.3.3 Another Solution for the Output

The cell of type D is another solution to choose b_0 and b_1 in Section B.2. For the LeakyLP cell we calculate the output as described in (41). Now we set $b_0 = \gamma_2^{\boldsymbol{p}} \gamma_3^{\boldsymbol{p}}$ and $b_1 = \left(1 - \gamma_2^{\boldsymbol{p}}\right) \gamma_3^{\boldsymbol{p}}$, and get

$$y_c^{\boldsymbol{p}} = \gamma_3^{\boldsymbol{p}} \left(\gamma_2^{\boldsymbol{p}} s_c^{\boldsymbol{p}} + \left(1 - \gamma_2^{\boldsymbol{p}}\right) s_c^{\boldsymbol{p}^-}\right).$$

This cell actually works as well as the MD LeakyLP cell and has the same number of gates. In this case we do not need a squashing function h_c, because we already have $y_c^{\boldsymbol{p}} \in [-1, 1]$.

B.3.4 An MD Cell as MD PID-Controller

Type E has a completely different interpretation: In controlling engineering a PID-controller gets an error as input. In our case the gate activations have to decide, if the proportional (P), the integral (I) or the derivative (D) term of the error is important for the output. When $\gamma_1^{\boldsymbol{p}} \approx 0$ we have $y_{c_{in}}^{\boldsymbol{p}} \approx s_c^{\boldsymbol{p}}$ so the internal state is proportional to the input. Then $\gamma_2^{\boldsymbol{p}}$ gates the proportional part (P) of the input. The second gate $\gamma_3^{\boldsymbol{p}}$ gates the difference between the last and the current input, which can be seen as a discrete derivative (D). If $\gamma_1^{\boldsymbol{p}} \approx 1$ the internal state is an exponential moving average of $y_{c_{in}}^{\boldsymbol{p}}$ which is an integral term. So $\gamma_2^{\boldsymbol{p}}$ gates a mainly integral part of the input (I), whereas $\gamma_3^{\boldsymbol{p}}$ gates a mainly proportional part of the input (P). Dependent on $\gamma_1^{\boldsymbol{p}}$ type E can be a PD-controller, a PI-controller or a mix of these both. In Figure 7(right) can be seen the frequency response of this cell for different gate activations.

Type	$g_{int}(\cdot)$	$g_{out}(\cdot)$	$H(z)$ for $h_c(x) = x$
A	$\left(1-\gamma_1^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + \gamma_1^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}$	$h_c\left(\gamma_2^{\boldsymbol{p}} s_c^{\boldsymbol{p}} + \gamma_3^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}\right)$	$\frac{(1-\gamma_1^{\boldsymbol{p}})}{1-\gamma_1^{\boldsymbol{p}} z^{-1}}\left(\gamma_2^{\boldsymbol{p}} + \gamma_3^{\boldsymbol{p}} z^{-1}\right)$
B	$\left(1-\gamma_1^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + \gamma_1^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}$	$\frac{s_c^{\boldsymbol{p}} + s_c^{\boldsymbol{p}^-}}{2}$	$\frac{(1-\gamma_1^{\boldsymbol{p}})}{1-\gamma_1^{\boldsymbol{p}} z^{-1}} \frac{1+z^{-1}}{2}$
C	$\left(1-\gamma_1^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + \gamma_1^{\boldsymbol{p}} \gamma_4^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}$	$h_c\left(\gamma_2^{\boldsymbol{p}} s_c^{\boldsymbol{p}} + \gamma_4^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}\right)$	$\frac{(1-\gamma_1^{\boldsymbol{p}})}{1-\gamma_1^{\boldsymbol{p}} \gamma_4^{\boldsymbol{p}} z^{-1}}\left(\gamma_2^{\boldsymbol{p}} + \gamma_3^{\boldsymbol{p}} z^{-1}\right)$
D	$\left(1-\gamma_1^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + \gamma_1^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}$	$\gamma_3^{\boldsymbol{p}}\left(\gamma_2^{\boldsymbol{p}} s_c^{\boldsymbol{p}} + \left(1-\gamma_2^{\boldsymbol{p}}\right) s_c^{\boldsymbol{p}^-}\right)$	$\frac{(1-\gamma_1^{\boldsymbol{p}})}{1-\gamma_1^{\boldsymbol{p}} z^{-1}} \gamma_3^{\boldsymbol{p}}\left(\gamma_2^{\boldsymbol{p}} + \left(1-\gamma_2^{\boldsymbol{p}}\right) z^{-1}\right)$
E	$\left(1-\gamma_1^{\boldsymbol{p}}\right) y_{c_{in}}^{\boldsymbol{p}} + \gamma_1^{\boldsymbol{p}} s_c^{\boldsymbol{p}^-}$	$h_c\left(\gamma_2^{\boldsymbol{p}} s_c^{\boldsymbol{p}} + \gamma_3^{\boldsymbol{p}}\left(s_c^{\boldsymbol{p}} - s_c^{\boldsymbol{p}^-}\right)\right)$	$\frac{(1-\gamma_1^{\boldsymbol{p}})}{1-\gamma_1^{\boldsymbol{p}} z^{-1}}\left(\gamma_2^{\boldsymbol{p}} + \gamma_3^{\boldsymbol{p}}\left(1-z^{-1}\right)\right)$

Table 7: Update equations and transfer function for different cell layouts. The column $s_c^{\boldsymbol{p}}$ contains the update equations to calculate the internal state and column $y_c^{\boldsymbol{p}}$ contains the update equation for the output. These equations lead to the transfer function $H(z) = H^{\gamma_1^{\boldsymbol{p}}, \gamma_2^{\boldsymbol{p}}, \cdots}(z)$.

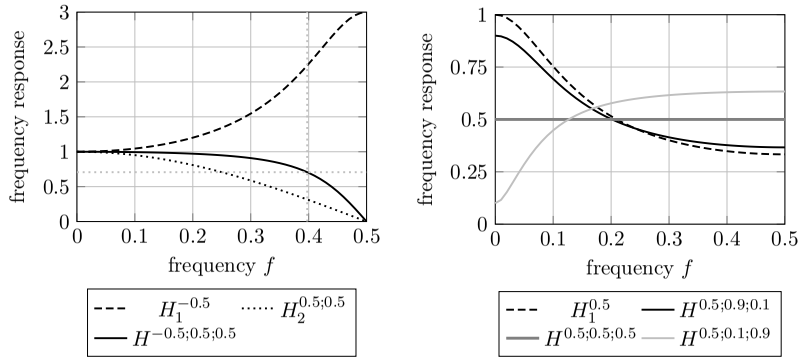


Figure 7: Frequency response of H_1 (dashed), H_2 (dotted) and H (solid) for special layouts and parameters of Table 7. Left (type B): A butterworth lowpass filter with a negative gate activation $\gamma_0^{\boldsymbol{p}} = -0.5$ leads to the cutoff frequency $f_{\text{cutoff}} = \frac{1}{\pi} \arctan\left(\frac{1+0.5}{1-0.5}\right) \approx 0.3976$. Right (type E): Different frequency responses of a PID controller. Having a fixed $\gamma_0^{\boldsymbol{p}} = 0.5$ the frequency response is dependent on the activations of $\gamma_1^{\boldsymbol{p}}$ and $\gamma_2^{\boldsymbol{p}}$ and can have lowpass (black), allpass (gray) and highpass (lightgray) characteristic.

References

A. Graves andJ. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *NIPS*, pages 545–552, 2008.

A. Graves, S. Fernandez andJ. Schmidhuber. Multi-dimensional recurrent neural networks. Technical report, IDSIA-04-07, 2007.

Augustin, E. and Brodin, J.-M and Carré, M. and Geoffrois, E. and Grosicki, E. and Prêteux, F. RIMES evaluation campaign for handwritten mail processing. In *Proc. of the Workshop on Frontiers in Handwriting Recognition*, number 1, 2006.

Y. Bengio. Practical recommendations for gradient-based training of deep architectures. *CoRR*, abs/1206.5533, 2012.

Y. Bengio and Y. LeCun. Convolutional networks for images, speech andtime-series. 1995.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. URL `http://www.iro.umontreal.ca/~lisa/pointeurs/ieeetrnn94.pdf`.

C. M. Bishop. *Pattern Recognition and Mashine Learning*. Springer, 2006.

E. Grosicki andH. El-Abed. ICDAR 2011: French handwriting recognition competition. In *Proc. of the Int. Conf. on Document Analysis and Recognition*, pages 1459–1463, 2011.

E. Grosicki, M. Carré, J.-M. Brodin andE. Geoffrois. RIMES evaluation campaign for handwritten mail processing. In *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition*, 2008.

F. A. Gers, J. Schmidhuber andF. Cummins. Learning to forget: Continual prediction with lstm. Technical report, IDSIA-01-99, 1999.

F. A. Gers, N. Schraudolph andJ. Schmidhuber. Learning precise timing with lstm recurrent networks. *Journal of Machine Learning Research*, 3:115–143, 2002.

Pechwitz, M. and Maddouri, S. and Märgner, V. and Ellouze, N. and Amiri, H. and others. Ifn/enit-database of handwritten arabic words. In *Proc. of CIFED*, volume 2, pages 127–136. Citeseer, 2002.

A. D. Poularikas. *The Transforms and Applications Handbook*. CRC Press, 2. edition edition, 2000.

S. Hochreiter, J. Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

D. Schlichthärle. *Digital Filters*. Springer, 2000.

Journal of Machine Learning Research 17 (2016) 1-37 Submitted 5/14; Revised 11/15; Published 2/16

Learning Taxonomy Adaptation in Large-scale Classification

Rohit Babbar * FIRSTNAME.LASTNAME@TUEBINGEN.MPG.DE
Max-Planck Institute for Intelligent Systems
Tübingen, Germany

Ioannis Partalas * IOANNIS.PARTALAS@VISEO.COM
Viseo Research Center
Grenoble, France

Eric Gaussier
Massih-Reza Amini
Cécile Amblard FIRSTNAME.LASTNAME@IMAG.FR
LIG, Université Grenoble Alpes - CNRS
Grenoble, cedex 9, France, 38041

Editor: Samy Bengio

Abstract

In this paper, we study flat and hierarchical classification strategies in the context of large-scale taxonomies. Addressing the problem from a learning-theoretic point of view, we first propose a multi-class, hierarchical data dependent bound on the generalization error of classifiers deployed in large-scale taxonomies. This bound provides an explanation to several empirical results reported in the literature, related to the performance of flat and hierarchical classifiers. Based on this bound, we also propose a technique for modifying a given taxonomy through pruning, that leads to a lower value of the upper bound as compared to the original taxonomy. We then present another method for hierarchy pruning by studying approximation error of a family of classifiers, and derive from it features used in a meta-classifier to decide which nodes to prune. We finally illustrate the theoretical developments through several experiments conducted on two widely used taxonomies.

Keywords: Large-scale classification, Hierarchical classification, Taxonomy adaptation, Rademacher complexity, Meta-learning

1. Introduction

With the rapid surge of digital data in the form of text and images, the scale of problems being addressed by machine learning practitioners is no longer restricted to the size of training and feature sets, but is also being quantified by the number of target classes. Classification of textual and visual data into a large number of target classes has attained significance particularly in the context of *Big Data*. This is due to the tremendous growth in data from various sources such as social networks, web-directories and digital encyclopedia. Directory Mozilla, DMOZ (`www.dmoz.org`), Wikipedia and Yahoo! Directory (`www.dir.yahoo.com`) are instances of such large scale textual datasets which consist of millions of documents that are distributed among hundreds of thousand target categories. Directory Mozilla, for instance, lists over 5 million websites distributed among close to 1 million categories, and is maintained by close to 100,000 editors. In the more commonly used Wikipedia, which consists of over 30 million pages, documents are typically assigned to multiple categories which are shown at the bottom of each page. The Medical Subject Heading (MESH) [1] hierarchy of the National Library of Medicine is another instance of a large-scale classification system in the domain of life sciences. The target classes in such large-scale scenarios typically have an inherent hierarchical structure among themselves. DMOZ is in the form of a rooted tree where a traversal of path from root-to-leaf depicts transformation of semantics from generalization to specialization. More generally parent-child relationship can exist in the form of directed acyclic graphs, as found in taxonomies such as Wikipedia. The tree and DAG relationship among categories is illustrated for DMOZ and Wikipedia taxonomies in Figure 1.

*. Most of this work was done when the authors were affiliated with LIG

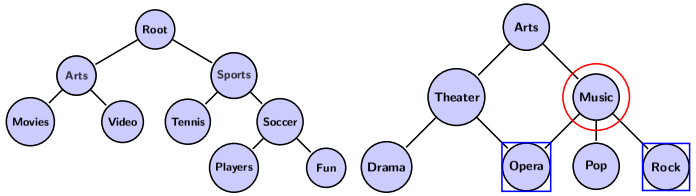


Figure 1: DMOZ and Wikipedia Taxonomies

Due to the sheer scale of the task of classifying data into target categories, there is a definite need to automate the process of classification of websites in DMOZ, encyclopedia pages in Wikipedia and medical abstracts in the MESH hierarchy. However, the scale of the data also poses challenges for the classical techniques that need to be adapted in order to tackle large-scale classification problems. In this context, one can exploit the taxonomy of classes as in the divide-and-conquer paradigm in order to partition the input space. Various classification techniques have been proposed for deploying classifiers in such large-scale scenarios, which differ in the way they exploit the given taxonomy. These can be broadly divided into four main categories :

- *Hierarchical top-down strategy* with independent classification problems at each node,
- Hierarchical top-down strategy on a *simplified hierarchy*, such as by partially flattening the hierarchy,
- Ignoring the hierarchy information altogether and using *flat classification* that is, training one classifier for each target class, and
- Using the taxonomy information for an *appropriate loss-function design* such as by considering the distance between the true and predicted target class label.

In large-scale classification involving tens of thousand of target categories, the goal of a machine learning practitioner is to achieve the best trade-off among the various metrics

1. `https://www.nlm.nih.gov/mesh/`

of interest. Flat and top-down hierarchical classification perform significantly differently on these metrics of interest which primarily include prediction accuracy, computational complexity of training, space complexity of the trained model and complexity of prediction.

1.1 Prediction Accuracy

Our focus in this work is primarily on the prediction accuracy when dealing with large-scale category systems. Hierarchical models for large scale classification, such as top-down Pachinko-machine based methods, suffer from the drawback that they have to make many decisions prior to reaching a final category, which leads to the error propagation phenomenon causing a decrease in accuracy. This is mainly due to the fact that the top level classes in large scale taxonomies are quite general. For example, *Business* and *Shopping* categories in DMOZ (not shown in Figure 1 above) are likely to be confused while classifying a new document. Moreover, since the classification is not recoverable, it leads to the phenomenon of error propagation and hence degrades accuracy at the leaf level. On the other hand, flat classifiers rely on a single decision including all the final categories, a single decision that is however difficult to make as it involves many categories, which are potentially unbalanced. It is thus very difficult to assess which strategy is better and there is no consensus, at the time being, on to which approach, flat or hierarchical, should be preferred on a particular category system. Furthermore, we explore the methods which learn to adapt the given hierarchy of categories such that the resulting hierarchy leads to better classification in the top-down Pachinko-machine based method. We also show that when dealing with large number of power-law distributed categories, taxonomy adaptation by pruning some nodes leads to better classification than building taxonomies from scratch.

1.2 Computational Complexity of Training and Prediction

When dealing with large-scale category systems, flat and hierarchical classification techniques exhibit significant difference in their performance when compared on the basis of computational complexity of training and prediction. In the pioneering work of Liu et al. (2005), an extensive comparison of training time complexity for flat and hierarchical classification has been studied. Using the power-law distribution of documents among categories, the authors analytically and empirically demonstrate that the training time of top-down Pachinko machine classification strategy is orders of magnitude better than that for flat classification.

In terms of complexity of prediction for flat classification when dealing with K target categories, for every test instance one needs to evaluate the inner-product with $O(K)$ weight vectors in . This is much higher than the logarithmic computational complexity of prediction in a top-down Pachinko-machine where only $O(\log(K))$ weight-vectors need to be evaluated. In view of this advantage for tree-based classifiers, there has been a surge in research works on the techniques for automatically building taxonomy of target categories (Bengio et al., 2010; Gao and Koller, 2011; Deng et al., 2011; Agrawal et al., 2013). The focus in these works is to show that by building such tree-based taxonomy of categories, one can reduce the complexity of prediction, while still maintaining good rates for accuracy of prediction.

Since the computational complexity of training and prediction has been studied in the above works, and it has been already shown that top-down methods are more favorable

as compared to flat method, we do not focus on these aspects explicitly in this paper. Furthermore, in our recent work Babbar et al. (2014a), we present a quantitative analysis of the fit to power-law distribution of documents among categories in large-scale category systems, and show that space complexity of top-down classification methods is lower than that of flat methods under conditions that typically hold in practice.

2. Contributions and Related Work

2.1 Contributions

One of the research challenges we address in this work is the study of flat versus hierarchical classification in large-scale taxonomies from a learning-theoretic point of view, and the consistency of the empirical risk minimization principle for the Pachinko-machine based methods. This theoretical analysis naturally leads to a model selection problem. We extend and elaborate further on our recent work Babbar et al. (2013a) to address the problem of choosing between the two strategies. We introduce bounds based on Rademacher complexity for the generalization errors of classifiers deployed in large-scale taxonomies. These bounds explicitly demonstrate the trade-off that both flat and hierarchical classifiers face in large-scale taxonomies.

Even though the given human-built taxonomies such as DMOZ and Yahoo! Directory provide a good starting point to capture the underlying semantics of the target categories, these may not be optimal, especially in the presence of large number of power-law distributed categories. In the second part of our contributions, we then propose a strategy for taxonomy adaptation which modifies the given taxonomy by pruning nodes in the tree to output a new taxonomy which is better suited for the classification problem. In this part, we exploit the generalization error bound developed earlier to build a criterion for Support Vector Machine classifier, so as to choose the nodes to prune in a computationally efficient manner. We also empirically show that adapting a given taxonomy leads to better generalization performance as compared to the original taxonomy and the ones obtained by taxonomy construction methods (Beygelzimer et al., 2009b; Choromanska and Langford, 2014). This difference is particularly magnified in category systems in which categories are power-law distributed. As discussed extensively in the work of Liu et al. (2005), power-law distribution of documents among categories is a common phenomenon in most naturally occurring category systems such as Yahoo! directory and Directory Mozilla.

In the third part, we present a more comprehensive approach for pruning the given hierarchy that is applicable to both discriminative and generative classifiers. Towards this end, we cover the Logistic Regression and Naive Bayes classifiers and present approximation error based bounds for their multi-class versions. Based on these bounds, we then propose a meta-learning strategy for hierarchy pruning applicable for both discriminative and generative classifiers. We perform a three-step procedure towards achieving hierarchy pruning, (i) we make classifier specific theoretical analysis to identify the key features which determine the variation in classification accuracy upon flattening, (ii) based on the features obtained in the step above, we train a meta-classifier on a validation set, and finally, (iii) the meta-classifier when presented with an unseen hierarchy and the corresponding training data modifies it to output the desired hierarchy which leads to better classification accuracy. Contrary to Dekel (2009) that reweighs the edges in a taxonomy through a cost sensitive

loss function to achieve this goal, our simple pruning strategy modifies the taxonomy in an explicit way.

Lastly, we empirically demonstrate and verify the theoretical findings for flat and hierarchical classification on several large-scale taxonomies extracted from DMOZ and the International Patent Classification (IPC) collections. The experimental results are in line with results reported in previous studies, as well as with our theoretical developments. Secondly, we also study the impact of the two methods proposed for taxonomy adaptation by pruning the hierarchy. Lastly, these strategies are also empirically compared against those that build taxonomies from scratch such as LOMTree (Choromanska and Langford, 2014) and FilterTree (Beygelzimer et al., 2009b).

2.2 Related Work

Large-scale classification, involving tens of thousand target categories, has assumed significance in the era of Big data. Many approaches for classification of data in large number of target categories have been proposed in the context of text and image classification. These approaches differ in the manner in which they exploit the semantic relationship among categories. In similar vein, open challenges such as Large-scale Hierarchical Text Classification (LSHTC) (Partalas et al., 2015) and Large Scale Visual Recognition Challenge (ILSVRC)[2] (Russakovsky et al., 2014) have been organized in recent years.

Some of the earlier works on exploiting hierarchy among target classes for the purpose of text classification has been studied by Koller and Sahami (1997) and Dumais and Chen (2000). These techniques use the taxonomy to train independent classifiers at each node in the top-down Pachinko Machine manner. Parameter smoothing for Naive Bayes classifier along the root to leaf path was explored by McCallum et al. (1998). The work by Liu et al. (2005) is one of first studies to apply hierarchical SVM to the scale with over 100,000 categories in Yahoo! directory. More recently, other techniques for large scale hierarchical text classification have been proposed. Prevention of error propagation by applying *Refined Experts* trained on a validation was proposed by Bennett and Nguyen (2009). In this approach, bottom-up information propagation is performed by utilizing the output of the lower level classifiers in order to improve the classification of top-level classifiers. Another approach to control the propagation of error in tree-based classifiers is to explore multiple root-to-leaf paths as in beam-search (Norvig, 1992). In this respect, Fleuret and Geman (2001); Sun et al. (2013) proposed such approaches. However, this increases the computationally complexity of prediction especially in the presence of large-number of target categories and hence these methods may not scale well for tens of thousand target categories. Deep Classification by Xue et al. (2008) proposes hierarchy pruning to first identify a much smaller subset of target classes. Prediction of a test instance is then performed by re-training a Naive Bayes classifier on the subset of target classes identified from the first step.

Using the taxonomy in the design of loss function for maximum-margin based approaches have been proposed by Cai and Hofmann (2004); Dekel et al. (2004), where the degree of penalization in mis-classification depends on the distance between the true and predicted class in the hierarchy tree. Another recent approach by Dekel (2009) which proposes to make the loss function design robust to class-imbalance and arbitrariness problems in taxonomy

2. http://www.image-net.org/challenges/LSVRC/

structure. However, these approaches were applied to the datasets in which the number of categories were limited to a few hundreds. Bayesian modeling of large scale hierarchical classification has been proposed by Gopal et al. (2012) in which hierarchical dependencies between the parent-child nodes are modeled by centering the prior of the child node at the parameter values of its parent. Recursive-regularization based strategy for large-scale classification has been proposed by Gopal and Yang (2013). The approaches presented in the two studies above attempt to solve the problem wherein the number of categories are in the range of tens of thousands. In both these works, the authors employ the intuition that the weight vectors of a parent-child pair of nodes should be close to each other. This can be enforced in the form of a prior in the Bayesian approach (Gopal et al., 2012) and as a regularizer in the recursive regularization approach (Gopal and Yang, 2013). However, on most of the large-scale datasets used in these papers, the accuracy performance (Micro-F1) of the proposed approaches is close to the flat classification scheme for which ready to use packages such as Liblinear are available. Another study related to our work is the one of Narasimhan et al. (2015) that studies the consistency of hierarchical classification algorithms with respect to the tree distance metric on the hierarchy tree of class labels.

Hierarchy simplification by flattening entire layer in the hierarchy has been studied from an empirical view-point by Wang and Lu (2010); Malik (2009). These strategies do not provide any theoretical justification for applying this procedure. Moreover, they offer no clear guidelines regarding which layer in the hierarchy one should flatten. In contrast, our strategy presented in this paper for taxonomy adaptation has the advantage that, (i) it is based on a well-founded theoretical criteria, and (ii) it is applied in a node-specific sense rather than to an entire layer. This strategy is also similar in spirit to the approach presented in our another recent study Babbar et al. (2013b) which is motivated from Perceptron Decision Trees (Bennett et al., 2000). The study by Weinberger and Chapelle (2008) introduces a slightly different simplification of the hierarchy of classes, and it achieves this by an embedding the classes and documents into a common space. Our recent work Babbar et al. (2013b) for hierarchy simplification by pruning nodes in a large-scale taxonomy is similar in spirit to the approach presented in this paper. Semi-supervised approach for hierarchical classification in incomplete hierarchies has been presented in the recent work of Dalvi and Cohen (2014). A post-processing approach for improving rare categories detection in large-scale power-law distributed category systems is discussed in the work by Babbar et al. (2014b).

Apart from accuracy, other important factors while evaluating the classification strategies for large scale classification are training and prediction speed. Learning the hierarchy tree from large number of classes in order to make faster prediction has also attained significance as explored in the recent works by Bengio et al. (2010); Beygelzimer et al. (2009a); Gao and Koller (2011); Choromanska and Langford (2014). The aim in these approaches is to achieve better prediction speed while maintaining the same classification accuracy as flat classification. On the other end of the spectrum are flat classification techniques such as employed by Perronnin et al. (2012) which ignore the hierarchy structure altogether. These strategies are likely to perform well for balanced hierarchies with sufficient training instances per target class and not so well in *truly* large-scale taxonomies which suffer from the problem of rare categories. In this respect, our work is unique in the sense that by performing selective hierarchy pruning we improve accuracy over the fully hierarchical

strategy. Furthermore, since the proposed pruning method maintains the overall hierarchical structure, it enjoys the computational advantages of better training and prediction speed.

The remainder of the paper is organized as follows: In Section 2.2 we review the recently proposed approaches in the context of large-scale hierarchical text classification. We introduce the notations used in Section 3 and then study flat versus hierarchical strategies by studying the generalization error bounds for classification in large-scale taxonomies. Taxonomy adaptation by pruning the hierarchy using the developed generalization error analyses is given in Section 4. Approximation error for multi-class versions of Naive Bayes and Logistic Regression classifiers are presented in Section 5.1 and Section 5.2 respectively, and the meta-learning based hierarchy pruning method is presented in Section 5.3. Section 6 illustrates these developments via experiments and finally, Section 7 concludes this study.

3. Flat versus Hierarchical Classification

In this section, we present the generalization error analysis for top-down hierarchical classification using the notion of Rademacher complexity for measuring the complexity of a function class. This will motivate a criterion for choosing between flat and hierarchical classification for a given category system. More importantly, the criterion will be based on quantities that can computed from the training data.

3.1 A hierarchical Rademacher data-dependent bound

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and let V be a finite set of class labels. We further assume that examples are pairs $(\mathbf{x}, v)$ drawn according to a fixed but unknown distribution $\mathcal{D}$ over $\mathcal{X} \times V$. In the case of hierarchical classification, the hierarchy of classes $\mathcal{H} = (V, E)$ is defined in the form of a rooted tree, with a root $\perp$ and a parent relationship $\pi : V \setminus \{\perp\} \to V$ where $\pi(v)$ is the parent of node $v \in V \setminus \{\perp\}$, and E denotes the set of edges with parent to child orientation. For each node $v \in V \setminus \{\perp\}$, we further define the set of its siblings $\mathfrak{S}(v) = \{v' \in V \setminus \{\perp\}; v \neq v' \wedge \pi(v) = \pi(v')\}$ and its children $\mathfrak{D}(v) = \{v' \in V \setminus \{\perp\}; \pi(v') = v\}$. The nodes at the intermediary levels of the hierarchy define general class labels while the specialized nodes at the leaf level, denoted by $\mathcal{Y} = \{y \in V : \nexists v \in V, (y, v) \in E\} \subset V$, constitute the set of target classes. Finally for each class y in $\mathcal{Y}$ we define the set of its ancestors $\mathfrak{P}(y)$ defined as

$$\mathfrak{P}(y) = \{v_1^y, \ldots, v_{k_y}^y; v_1^y = \pi(y) \wedge \forall l \in \{1, \ldots, k_y - 1\}, v_{l+1}^y = \pi(v_l^y) \wedge \pi(v_{k_y}^y) = \perp\}$$

For classifying an example $\mathbf{x}$, we consider a top-down classifier making decisions at each level of the hierarchy, this process sometimes referred to as the *Pachinko* machine selects the best class at each level of the hierarchy and iteratively proceeds down the hierarchy. In the case of flat classification, the hierarchy $\mathcal{H}$ is ignored, $\mathcal{Y} = V$, and the problem reduces to the classical supervised multi-class classification problem.

Our main result is the following theorem which provides a data-dependent bound on the generalization error of a top-down multi-class hierarchical classifier. We consider here kernel-based hypotheses, with $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a PDS (positive definite symmetric) kernel

and $\Phi : \mathcal{X} \to \mathbb{H}$ its associated feature mapping function, defined as :

$$\mathcal{F}_B = \{f : (\mathbf{x}, v) \in \mathcal{X} \times V \mapsto \langle \Phi(\mathbf{x}), \mathbf{w}_v \rangle \mid \mathbf{W} = (w_1 \ldots, w_{|V|}), ||\mathbf{W}||_{\mathbb{H}} \leq B\}$$

where $\mathbf{W} = (w_1 \ldots, w_{|V|})$ is the matrix formed by the $|V|$ weight vectors defining the kernel-based hypotheses, $\langle ., . \rangle$ denotes the dot product, and $||\mathbf{W}||_{\mathbb{H}} = \left(\sum_{v \in V} ||\mathbf{w}_v||^2\right)^{1/2}$ is the $L^2_{\mathbb{H}}$ group norm of $\mathbf{W}$. We further define the following associated function class:

$$\mathcal{G}_{\mathcal{F}_B} = \{g_f : (\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y} \mapsto \min_{v \in \mathfrak{P}(y)} (f(\mathbf{x}, v) - \max_{v' \in \mathfrak{S}(v)} f(\mathbf{x}, v')) \mid f \in \mathcal{F}_B\}$$

For a given hypothesis $f \in \mathcal{F}_B$, the sign of its associated function $g_f \in \mathcal{G}_{\mathcal{F}_B}$ directly defines a hierarchical classification rule for f as the top-down classification scheme outlined before simply amounts to: *assign* $\mathbf{x}$ *to* y *iff* $g_f(\mathbf{x}, y) > 0$. The learning problem we address is then to find a hypothesis f from $\mathcal{F}_B$ such that the generalization error of $g_f \in \mathcal{G}_{\mathcal{F}_B}$, $\mathcal{E}(g_f) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\mathbb{1}_{g_f(\mathbf{x},y)\leq 0}\right]$, is minimal ($\mathbb{1}_{g_f(\mathbf{x},y)\leq 0}$ is the 0/1 loss, equal to 1 if $g_f(\mathbf{x}, y) \leq 0$ and 0 otherwise).

The following theorem sheds light on the trade-off between flat versus hierarchical classification. The notion of function class capacity used here is the *empirical Rademacher complexity* (Bartlett and Mendelson, 2002; Meir and Zhang, 2003).

Theorem 1 *Let* $\mathcal{S} = ((\mathbf{x}^{(i)}, y^{(i)}))_{i=1}^m$ *be a dataset of* m *examples drawn i.i.d. according to a probability distribution* $\mathcal{D}$ *over* $\mathcal{X} \times \mathcal{Y}$*, and let* $\mathcal{A}$ *be a Lipschitz function with constant* L *dominating the* 0/1 *loss; further let* $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *be a PSD (positive semi-definite) kernel and let* $\Phi : \mathcal{X} \to \mathbb{H}$ *be the associated feature mapping function. Assume that there exists* $R > 0$ *such that* $K(\mathbf{x}, \mathbf{x}) \leq R^2$ *for all* $\mathbf{x} \in \mathcal{X}$*. Then, for all* $1 > \delta > 0$*, with probability at least* $(1 - \delta)$ *the following hierarchical multi-class classification generalization bound holds for all* $g_f \in \mathcal{G}_{\mathcal{F}_B}$ *:*

$$\mathcal{E}(g_f) \leq \frac{1}{m} \sum_{i=1}^{m} \mathcal{A}(g_f(\mathbf{x}^{(i)}, y^{(i)})) + \frac{8BRL}{\sqrt{m}} \sum_{v \in V \setminus \mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)| - 1) + 3\sqrt{\frac{\ln(2/\delta)}{2m}} \quad (1)$$

where $|\mathfrak{D}(v)|$ *denotes the number of children of node* v*.*

Proof Exploiting the fact that $\mathcal{A}$ dominates the 0/1 loss and using the Rademacher data-dependent generalization bound presented in Theorem 4.9 of (Shawe-Taylor and Cristianini, 2004), one has:

$$\begin{aligned} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{1}_{g_f(\mathbf{x},y)\leq 0} - 1\right] &\leq \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\mathcal{A} \circ g_f(\mathbf{x}, y) - 1\right] \\ &\leq \frac{1}{m} \sum_{i=1}^{m} (\mathcal{A}(g_f(\mathbf{x}^{(i)}, y^{(i)})) - 1) + \hat{\mathcal{R}}_m((\mathcal{A} - 1) \circ \mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) + 3\sqrt{\frac{\ln(2/\delta)}{2m}} \end{aligned}$$

where $\hat{\mathcal{R}}_m$ denotes the empirical Rademacher complexity of $(\mathcal{A}-1)\circ\mathcal{G}_{\mathcal{F}_B}$ on $\mathcal{S}$. As $x \mapsto \mathcal{A}(x)$ is a Lipschtiz function with constant L and $(\mathcal{A} - 1)(0) = 0$, we further have:

$$\hat{\mathcal{R}}_m((\mathcal{A} - 1) \circ \mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) \leq 2L\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S})$$

with:

$$\begin{aligned}\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B},\mathcal{S}) &= \mathbb{E}_\sigma\left[\sup_{g_f\in\mathcal{G}_{\mathcal{F}_B}}\left|\frac{2}{m}\sum_{i=1}^m \sigma_i\, g_f(\mathbf{x}^{(i)},y^{(i)})\right|\right]\\ &= \mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}_B}\left|\frac{2}{m}\sum_{i=1}^m \sigma_i \min_{v\in\mathfrak{P}(y^{(i)})}(f(\mathbf{x}^{(i)},v)-\max_{v'\in\mathfrak{S}(v)} f(\mathbf{x}^{(i)},v'))\right|\right]\end{aligned}$$

where σ_is are independent uniform random variables which take value in $\{-1,+1\}$ and are known as Rademacher variables.

Let us define the mapping c from $\mathcal{F}_B\times\mathcal{X}\times\mathcal{Y}$ into $V\times V$ as:

$$\begin{aligned}c(f,\mathbf{x},y)=(v,v') \;\Rightarrow\; & (f(\mathbf{x},v')=\max_{v''\in\mathfrak{S}(v)} f(\mathbf{x},v''))\\ & \wedge\,(f(\mathbf{x},v)-f(\mathbf{x},v')=\min_{u\in\mathfrak{P}(y)}(f(\mathbf{x},u)-\max_{u'\in\mathfrak{S}(u)} f(\mathbf{x},u')))\end{aligned}$$

This definition is similar to the one given by Guermeur (2010) for flat multi-class classification. Then, by construction of c:

$$\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B},\mathcal{S})\le\frac{2}{m}\mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}_B}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\left|\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\sigma_i(f(\mathbf{x}^{(i)},v)-f(\mathbf{x}^{(i)},v'))\right|\right]$$

By definition, $f(\mathbf{x}^{(i)},v)-f(\mathbf{x}^{(i)},v')=\langle\mathbf{w}_v-\mathbf{w}_{v'},\Phi(\mathbf{x}^{(i)})\rangle$ and using Cauchy-Schwartz inequality:

$$\begin{aligned}\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B},\mathcal{S}) &\le \frac{2}{m}\mathbb{E}_\sigma\left[\sup_{\|\mathbf{W}\|_{\mathbb{H}}\le B}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\left|\left\langle\mathbf{w}_v-\mathbf{w}_{v'},\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\sigma_i\Phi(\mathbf{x}^{(i)})\right\rangle\right|\right]\\ &\le \frac{2}{m}\mathbb{E}_\sigma\left[\sup_{\|\mathbf{W}\|_{\mathbb{H}}\le B}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\|\mathbf{w}_v-\mathbf{w}_{v'}\|_{\mathbb{H}}\left\|\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\sigma_i\Phi(\mathbf{x}^{(i)})\right\|_{\mathbb{H}}\right]\\ &\le \frac{4B}{m}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\mathbb{E}_\sigma\left[\left\|\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\sigma_i\Phi(\mathbf{x}^{(i)})\right\|_{\mathbb{H}}\right]\end{aligned}$$

Using Jensen's inequality, and as, $\forall i,j\in\{l|c(f,\mathbf{x}^{(l)},y^{(l)})=(v,v')\}^2, i\neq j, \mathbb{E}_\sigma[\sigma_i\sigma_j]=0$, we get:

$$\begin{aligned}\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B},\mathcal{S}) &\le \frac{4B}{m}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\left(\mathbb{E}_\sigma\left[\left\|\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\sigma_i\Phi(\mathbf{x}^{(i)})\right\|_{\mathbb{H}}^2\right]\right)^{1/2}\\ &= \frac{4B}{m}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\left(\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}\left\|\Phi(\mathbf{x}^{(i)})\right\|_{\mathbb{H}}^2\right)^{1/2}\\ &= \frac{4B}{m}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\left(\sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')}K\left(\mathbf{x}^{(i)},\mathbf{x}^{(i)}\right)\right)^{1/2}\\ &\le \frac{4B}{m}\sum_{(v,v')\in V^2,v'\in\mathfrak{S}(v)}\left(mR^2\right)^{1/2}\\ &= \frac{4BR}{\sqrt{m}}\sum_{v\in V\setminus\mathcal{Y}}|\mathfrak{D}(v)|(|\mathfrak{D}(v)|-1)\end{aligned}$$

Plugging this bound into the first inequality yields the desired result. □

This generalization bound proves the consistency of the ERM principle for the Pachinko-machine based method. Further, for flat multiclass classification, we recover the bounds by Guermeur (2010) by considering a hierarchy containing a root node with as many children as there are categories. Note that the definition of functions in $\mathcal{G}_{\mathcal{F}_B}$ subsumes the definition of the margin function used for the flat multiclass classification problems by Guermeur (2010), and that the factor $8L$ in the complexity term of the bound, instead of 4 by Guermeur (2010), is due to the fact that we are using an L-Lipschitz loss function dominating the 0/1 loss in the empirical Rademacher complexity. Krishnapuram et al. (2005) they provide PAC-Bayes bounds, different to ours, for Bayes Voting classifiers and Gibbs classifier under a PAC-Bayes setup (McAllester, 1998; Seeger, 2003). Lastly, Bartlett et al. (2005) have proposed tighter bounds using local Rademacher complexities. Using such bounds would lead to replace the term involving the complexity of the hierarchy in Theorem 1 by a term involving the fixed point of a sub-root function that upper bounds local Rademacher averages. Such a replacement, if it can lead to tighter bounds under some additional conditions, would however miss the explanation provided below on the behaviors of flat and hierarchical classifiers, an explanation that will be confirmed experimentally.

Flat vs hierarchical classification in large-scale taxonomies. The generalization error is controlled in inequality (1) by a trade-off between the empirical error and the Rademacher complexity of the class of classifiers. The Rademacher complexity term favors hierarchical classifiers over flat ones, as any split of a set of category of size K in p parts $K_1,\cdots,K_p$ ($\sum_{i=1}^p K_i=K$) is such that $\sum_{i=1}^p K_i^2\le K^2$. On the other hand, the empirical error term is likely to favor flat classifiers vs hierarchical ones, as the latter rely on a series of decisions (as many as the length of the path from the root to the chosen category in $\mathcal{Y}$) and are thus more likely to make mistakes. This fact is often referred to as the *propagation error* problem in hierarchical classification.

On the contrary, flat classifiers rely on a single decision and are not prone to this problem (even though the decision to be made is harder). When the classification problem in $\mathcal{Y}$ is highly unbalanced, then the decision that a flat classifier has to make is difficult; hierarchical classifiers still have to make several decisions, but the imbalance problem is less severe on each of them. So, in this case, even though the empirical error of hierarchical classifiers may be higher than the one of flat ones, the difference can be counterbalanced by the Rademacher complexity term, and the bound in Theorem 1 suggests that hierarchical classifiers should be preferred over flat ones.

On the other hand, when the data is well balanced, the Rademacher complexity term may not be sufficient to overcome the difference in empirical errors due to the propagation error in hierarchical classifiers; in this case, Theorem 1 suggests that flat classifiers should be preferred to hierarchical ones. These results have been empirically observed in different studies on classification in large-scale taxonomies and are further discussed in Section 6.

Similarly, one way to improve the accuracy of classifiers deployed in large-scale taxonomies is to modify the taxonomy by pruning (sets of) nodes (Wang and Lu, 2010). By doing so, one is flattening part of the taxonomy and is once again trading-off the two terms in inequality (1): pruning nodes leads to reduce the number of decisions made by the hierarchical classifier while maintaining a reasonable Rademacher complexity. Motivated from the Rademacher-based generalization error bound presented in Theorem 1, we now propose a method for pruning nodes of the given taxonomy. The output of this procedure is a new taxonomy which leads to improvement in classification accuracy when used for top-down classification.

4. Hierarchy Pruning

In this section, we present a strategy aiming at adapting the given hierarchy of classes by pruning some nodes in the hierarchy. An example of node pruning is shown in Figure 2. The rationale and motivation behind adapting the given hierarchy $\mathcal{H} = (V, E)$ to the set of input/output pair $(\mathbf{x}, y)$ is that

- Large-scale taxonomies, such as DMOZ and Yahoo! Directory, are designed with an intent of better user-experience and navigability, and not necessarily for the goal of classification,
- Taxonomy design is subject to certain degree of arbitrariness based on personal choices and preferences of the editors. Therefore, many competing taxonomies may exist, and
- The large-scale nature of such taxonomies poses difficulties in manually designing good taxonomies for classification.

The problem of pruning a hierarchy can be seen as a structure learning problem, where one wants to learn a simplified structure from a given one. The main difficulty in solving this problem is to identify the important features on which to base the decision to prune a node or not. We first present in this section a straightforward strategy that behaves well in practice but is nevertheless computationally expensive, prior to propose a "lighter" strategy based on the previous results. We will, in the next section (Section 5), introduce new theoretical results that will help us identify important features for node pruning, and

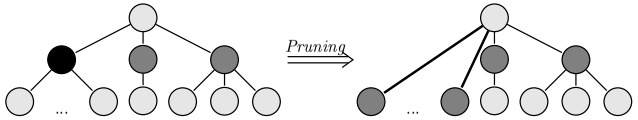


Figure 2: The pruning procedure; the node in black is replaced by its children. In the figure on the left, the gray nodes represent the siblings or sisters of node in black.

on which we will develop a greedy procedure to simplify a given hierarchy. The rationale for a greedy approach here is that optimal pruning would require evaluations of 2^k possible prunings for k siblings, which is infeasible in practice.

4.1 Hierarchy Pruning based on validation estimate

The challenge in the pruning procedure is to identify promising nodes which when pruned lead to improvement in classification accuracy. One of the simplest methods to identify such nodes is by using a validation set to check if pruning a node improves classification accuracy on that set by comparing it with accuracy obtained on the original taxonomy. This can also be interpreted as follows:

$$\text{Whether to prune a node } v? = \begin{cases} \text{Yes} & \text{If classification improves on the validation set} \\ \text{No} & \text{otherwise} \end{cases}$$

Algorithm 1 Hierarchy pruning based on validation estimate

Require: A hierarchy $\mathcal{G}$, Training set $\mathcal{S}$ and a validation set $\mathcal{S}'$
Train SVM classifier at each node of the tree $\mathcal{G}$ using the training set $\mathcal{S}$
Evaluate the accuracy of the classifier-cascade $\mathcal{G}$ on the validation set
for $v \in \mathcal{V}$ **do**
    Prune the node v and replace it by its children
    Re-train SVM classifier at the impacted node of the tree $\mathcal{G}'$
    Evaluate the accuracy of the classifier-cascade $\mathcal{G}'$ on the validation set
    if Cross-validation accuracy is higher on $\mathcal{G}'$ as compared to $\mathcal{G}$ **then**
        Prune the node v
    else
        Do not prune the node v
    end if
end for
return Pruned taxonomy $\mathcal{G}'$

This simple strategy is algorithmically presented in Algorithm 1. In terms of prediction accuracy, this method for pruning works reasonably well, however its main disadvantages are the computational complexity and lack of generalizability to new but somewhat related

taxonomies. In section 6, we also present experimental results obtained by this pruning strategy vis-à-vis other pruning methods presented later in this paper.

Computationally, this method requires (i) a trained cascade of top-down classifiers, (ii) for every pruned node, re-training the parent of the pruned node, and (iii) for every such node, evaluating the top-down performance on the validation set, which involves traversing the root-to-leaf path of classifier evaluation along the taxonomy. Furthermore, the steps (ii) and (iii) are to be repeated for every pruned node. Let C_{td-cas} denotes the computational complexity for training the cascade, C_v denotes the complexity for re-training of the parent node after pruning the node v, and C_{val} denotes the complexity of evaluating the validation set. Let $|V|_p$ denote the number of pruned nodes, then the complexity of the Algorithm 1 is $C_{td-cas} + |V|_p \times (C_v + C_{val})$. Due to the linear dependence on the number of pruned nodes, it becomes computationally expensive to prune a reasonably large number of nodes and check if this would result in improvement in classification accuracy of the top-down cascade. Furthermore, this process does not amount to a learning-based method for pruning and hence needs to be employed from scratch for newer taxonomies, even though these taxonomies may have similar characteristics to those encountered already.

To summarize, though quite simple, the above pruning method has the following disadvantages:

- This is a computationally expensive process to re-train the classifier at the pruned nodes and then test the performance on the validation set. As a result, this may not be applicable for large-scale taxonomies consisting of large number of internal nodes,
- This method does not amount to a learning-based strategy for pruning, and ignores data-dependent information available at the nodes of the taxonomy, and
- This process needs to be repeated for each taxonomy encountered, and information gained from taxonomy cannot be leveraged for newer taxonomies with similar characteristics.

We now turn to another method for taxonomy adaptation by pruning which is based on the generalization error analysis derived in Section 3.1. This method is computationally efficient compared to that presented in Algorithm 1 and only requires a cascade of top-down classifiers. Essentially, the criterion for pruning, which is related to the margin at each node, can be computed while training the top-down cascade. This corresponds to only the first step in Algorithm 1, and rest of the steps of evaluating on validation set are no longer required. Therefore, in terms of computational complexity, the method proposed in the next section has complexity of C_{td-cas}.

4.2 Hierarchy Pruning based on generalization-error

In this section, we present a strategy for pruning which is theoretically well motivated and is based on the generalization error bound for understanding the trade-off for flat and hierarchical classification. In view of the generalization error bound derived in Theorem 1, adapting the given taxonomy of classes aims at achieving a better trade-off between the empirical error and the error attributed to Rademacher complexity. In other words, adapting the given taxonomy $\mathcal{H}$ to the set of input output pairs $(\mathbf{x}, v)$ aims at achieving a lower value

of the bound as compared to that attained by using the original hierarchy. For a node v with parent $\pi(v)$, pruning v and replacing it by its children will increase the number of children of $\pi(v)$ and hence the associated Rademacher complexity but will decrease the empirical error along that path from root to leaf. Therefore, we need to identify those nodes in the taxonomy for which increase in the Rademacher complexity is among the lowest so that a better trade-off between the two error terms is achieved than in the original hierarchy. For this purpose, we turn to the bound on the empirical Rademacher complexity of the function class $\mathcal{G}_{\mathcal{F}_B}$.

In the derivation of Theorem 1, the empirical Rademacher complexity was upper bounded as follows:

$$\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S}) \leq \frac{2}{m}\mathbb{E}_\sigma \left[\sup_{\|\mathbf{W}\|_{\mathbb{H}} \leq B} \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \|\mathbf{w}_v - \mathbf{w}_{v'}\|_{\mathbb{H}} \left\| \sum_{i:c(f,\mathbf{x}^{(i)},y^{(i)})=(v,v')} \sigma_i \Phi(\mathbf{x}^{(i)}) \right\|_{\mathbb{H}} \right] \quad (2)$$

From the above bound, we define a quantity $C(v)$ for each node v

$$C(v) = \sum_{(v,v') \in V^2, v' \in \mathfrak{S}(v)} \|\mathbf{w}_v - \mathbf{w}_{v'}\|_{\mathbb{H}} \quad (3)$$

As one can note, the right hand side of the inequality (2) above provides an upper bound on $\hat{\mathcal{R}}_m(\mathcal{G}_{\mathcal{F}_B}, \mathcal{S})$ and one can meaningfully compare only the sibling nodes since these nodes have the same training set. Thus, the explicit computation of expectation $\mathbb{E}_\sigma(.)$ with respect to the Rademacher random variables and the computation of the inner product in the feature space can be avoided. This motivates the definition of $C(v)$ in equation (3), that can be efficiently and effectively computed from the training data, and that represents a distance of node v to its sibling nodes. It must be noted that $C(v')$, for a set of siblings $\{v'\}$, as computed using equation 3 is different for each node v'.

$C(v)$ is higher when $\mathbf{w}_v$ is larger than $\mathbf{w}_{v'}$ (when measured in terms of L2-distance), for all siblings v' of v, or when $\mathbf{w}_v$ is far away from $\mathbf{w}_{v'}$ (implying that the L2-norm of the difference is large), or both. The first and last cases correspond to unbalanced classes, v being the dominant class. In such cases, pruning v by replacing it by its children leads to a more balanced problem, less prone to classification errors. Furthermore, as children of v are based on the features in v, most of them will likely be far away from the siblings of v, and the pruning, even though increasing the Rademacher complexity term, will decrease the empirical error term and, likely, the generalization error. In the second case, pruning v will lead to children that will again be, very likely, far away from the siblings of v. This pruning thus does not introduce confusion between categories and reduces the problem related to error propagations.

This suggests that an effective pruning algorithm must prune the nodes v in the taxonomy for which $C(v)$ is maximal. In practice, we focus on pruning the nodes in the top-two layers of the taxonomy. This is due to the following reasons:

- The categories in these levels represent generic concepts, such as Entertainment and Sports in Yahoo! Directory, which are typically over-lapping in nature, and
- This is also shown in the plot in Figure 3 for the average confusion of the nodes C_v^{avg} for the different levels for two of the taxonomies used in our experiments. It shows

that the confusion among the top-level nodes is much higher as compared to those in the lower levels.

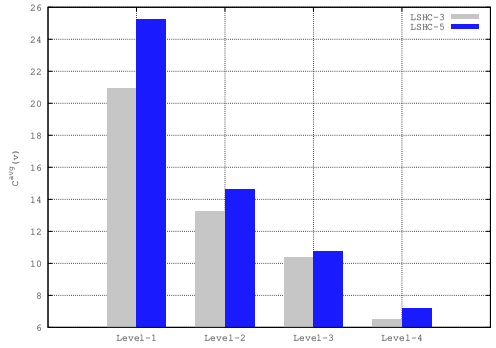


Figure 3: C_v^{avg} **plotted for various levels in the hierarchy Level 1 corresponds to the top-most level.**

Algorithm 2 The proposed method for hierarchy pruning based on Generalization Bound

Require: a hierarchy $\mathcal{G}$, Training set $\mathcal{S}$ consisting of $(\mathbf{x}, y)$ pairs, $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$
  Train SVM classifier at each node of the tree
  $\Delta \leftarrow 0$
  for $v \in \mathcal{V}$ **do**
    Sort its child nodes $v' \in \mathfrak{D}(v)$ in decreasing order of $C(v')$
    Flatten 1st and 2nd ranked child nodes, say v'_1 and v'_2
    $\Delta = C(v'_1) - C(v'_2)$
    $v_{prev} \leftarrow v'_2$ ▷ Set the previous flattened node to v'_2
    for $v' \in \mathcal{V} - \{v'_1, v'_2\}, (v, v') \in \mathcal{E}$ **do**
      if $C(v_{prev}) - C(v') < \Delta$ **then**
        Flatten v'
        $\Delta \leftarrow C(v_{prev}) - C(v')$
        $v_{prev} \leftarrow v'$ ▷ Set the previous flattened node to v'
      else
        break
      end if
    end for
  end for
  return Pruned taxonomy $\mathcal{G}'$

The pruning process as an algorithmic procedure is shown in Algorithm 2, where the variable Δ is used to stop the pruning process in an inner iteration.

The above criterion for pruning the nodes in a large-scale taxonomy is also similar in spirit to the method introduced by Babbar et al. (2013b) which is motivated from the generalization error analysis of Perceptron Decision Trees. As shown in the experiments on large-scale datasets by using SVM and Logistic Regression classifiers, applying this strategy outputs a new taxonomy which leads to better classification accuracy as compared to the original taxonomy.

It may be noted that the pruning procedure adopts a conservative approach to avoid excessive flattening of the taxonomy. It can be modified to prune the nodes more aggressively by scaling the parameter Δ after pruning of every node, and hence allow more nodes to be pruned. However, irrespective of such design choice, this method based on the generalization bound for pruning the hierarchy has two following disadvantages :

- Higher computational complexity since one needs to learn the weight vector $\mathbf{w}_v$ for each node v in the given taxonomy. As a result, the process of identifying these nodes can be computationally expensive for large-scale taxonomies;
- It is restricted only to discriminative classifiers such as Support Vector Machines and Logistic Regression.

Therefore, we next present a meta-learning based pruning strategy for hierarchy pruning which avoids this initial training of the entire taxonomy, and is applicable to both discriminative and generative classifiers.

5. Meta-learning based pruning strategy

In this section, we present a meta-learning based generic pruning strategy which is applicable to both discriminative and generative classifiers. The meta-features for the instances are derived from the analysis of the approximation error for multi-class versions of the two well-known generative and discriminative classifiers: Naive Bayes and Logistic Regression. We then show how this generalization error analysis of the classifier at each node is combined when deployed in a typical top-down cascade of the hierarchy tree. Based on these analyses, we identify the important features that control the variation of the generalization error and determine whether a particular node should be flattened or not. We finally train a meta-classifier based on these meta-features, which predicts whether replacing a node in the hierarchy by its children (Figure 2) will improve the classification accuracy or not.

The remainder of this section is organized as follows:

1. In Section 5.1, we present asymptotic error bounds for Naive Bayes classifiers;
2. Asymptotic error bounds for Multinomial Logistic Regression classifiers are given in Section 5.2;
3. We then develop in Section 5.3:
   (a) A *pruning* bound for both types of classifiers;
   (b) A meta-classifier for pruning nodes of a taxonomy, based on features derived from both asymptotic error and pruning bounds.

Theorem 2 below is recalled by Ng and Jordan (2001). Theorems 3 and 4 provide multi-class versions of the bounds proposed by Ng and Jordan (2001) for the Naive Bayes and Logistic Regression classifiers respectively. Lastly, Theorem 5 provides a hierarchical generalization of these bounds for both classifiers. The features we are using to learn the meta-classifier are derived from Theorem 5.

5.1 Asymptotic approximation error bounds for Naive Bayes

Let us first consider a multinomial, multiclass Naive Bayes classifier in which the predicted class is the one with maximum posterior probability. The parameters of this model are estimated by maximum likelihood and we assume here that Laplace smoothing is used to avoid null probabilities. Our goal here is to derive a generalization error bound for this classifier. To do so, we recall the bound for the binomial version (directly based on the presence/absence of each feature in each document) of the Naive Bayes classifier for two target classes (Theorem 4 of (Ng and Jordan, 2001)).

Theorem 2 *For a two class classification problem in d dimensional feature space with m training examples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$ sampled from distribution $\mathcal{D}$, let h and h_∞ denote the classifiers learned from the training set of finite size m and its asymptotic version respectively. Then, with high probability, the bound on misclassification error of h is given by*

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G\left(O\left(\sqrt{\frac{1}{m}\log d}\right)\right) \tag{4}$$

where $G(\tau)$ represents the probability that the asymptotic classifier predicts correctly and has scores lying in the interval $(-d\tau, d\tau)$.

We extend here this result to the multinomial, multiclass Naive Bayes classifier, for a K class classification problem with $\mathcal{Y} = \{y_1, \ldots y_K\}$. To do so, we first introduce the following lemma, that parallels Lemma 3 of (Ng and Jordan, 2001):

Lemma 1 *$\forall y_k \in \mathcal{Y}$, let $\widehat{P}(y_k)$ be the estimated class probability and $P(y_k)$ its asymptotic version obtained with a training set of infinite size. Similarly, $\forall y_k \in \mathcal{Y}$ and $\forall i$, $1 \leq i \leq d$, let $\widehat{P}(w_i|y_k)$ be the estimated class conditional feature probability and $P(w_i|y_k)$ its asymptotic version (w_i denotes the i^{th} word of the vocabulary). Then, $\forall \epsilon > 0$, with probability at least $(1-\delta)$ we have :*

$$|\widehat{P}(y_k) - P(y_k)| < \epsilon,\ |\widehat{P}(w_i|y_k) - P(w_i|y_k)| < \epsilon$$

with $\delta = K\delta_0 + d\sum_{k=1}^K \delta_k$, where $\delta_0 = 2\exp(-2m\epsilon^2)$ and $\delta_k = 2d\exp(-2d_k\epsilon^2)$. d_k represents the length of class y_k, that is the sum of lengths (in number of occurrences) of all the documents in class k.

The proof of this lemma directly derives from Hoeffding's inequality and the union bound, and is a direct extension of the proof of Lemma 3 given by Ng and Jordan (2001).

Let us now denote the joint log-likelihood of the vector representation of (a document) $\boldsymbol{x}$ in class y_k by $l(\boldsymbol{x}, y_k)$:

$$l(\boldsymbol{x}, y_k) = \log\left[\widehat{P}(y_k)\prod_{i=1}^d \widehat{P}(w_i|y_k)^{x_i}\right] \tag{5}$$

where x_i represents the number of times word w_i appears in $\boldsymbol{x}$. The decision of the Naive Bayes classifier for an instance $\boldsymbol{x}$ is given by:

$$h(\boldsymbol{x}) = \underset{y_k \in \mathcal{Y}}{\operatorname{argmax}}\, l(\boldsymbol{x}, y_k) \tag{6}$$

and the one for its asymptotic version by:

$$h_\infty(\boldsymbol{x}) = \underset{y_k \in \mathcal{Y}}{\operatorname{argmax}}\, l_\infty(\boldsymbol{x}, y_k) \tag{7}$$

Lemma 1 suggests that the predicted and asymptotic log-likelihoods are close to each other, as the quantities they are based on are close to each other. Thus, provided that the asymptotic log-likehoods between the best two classes, for any given $\boldsymbol{x}$, are not too close to each other, the generalization error of the Naive Bayes classifier and the one of its asymptotic version are close to each other. Theorem 3 below states such a relationship, using the following function that measures the confusion between the best two classes for the asymptotic Naive Bayes classifier.

Definition 1 *Let $l^1_\infty(\boldsymbol{x}) = \max_{y_k \in \mathcal{Y}} l_\infty(\boldsymbol{x}, y_k)$ be the best log-likelihood score obtained for $\boldsymbol{x}$ by the asymptotic Naive Bayes classifier, and let $l^2_\infty(\boldsymbol{x}) = \max_{y_k \in \mathcal{Y}\setminus h_\infty(\boldsymbol{x})} l_\infty(\boldsymbol{x}, y_k)$ be the second best log-likelihood score for $\boldsymbol{x}$. We define the confusion of the asymptotic Naive Bayes classifier for a category set $\mathcal{Y}$ as:*

$$G_{\mathcal{Y}}(\tau) = P_{(\boldsymbol{x},y)\sim D}(|l^1_\infty(\boldsymbol{x}) - l^2_\infty(\boldsymbol{x})| < 2\tau)$$

for $\tau > 0$.

We are now in position to formulate a relationship between the generalization error of the multinomial, multiclass Naive Bayes classifier and its asymptotic version.

Theorem 3 *For a K class classification problem in d dimensional feature space with a training set of size m, $\{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^m$, $\boldsymbol{x}^{(i)} \in \mathcal{X}$, $y^{(i)} \in \mathcal{Y}$, sampled from distribution $\mathcal{D}$, let h and h_∞ denote the Naive Bayes classifiers learned from a training set of finite size m and its asymptotic version respectively, and let $\mathcal{E}(h)$ and $\mathcal{E}(h_\infty)$ be their generalization errors. Then, $\forall \epsilon > 0$, one has, with probability at least $(1-\delta_{\mathcal{Y}})$:*

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}(\epsilon) \tag{8}$$

with:

$$\delta_{\mathcal{Y}} = 2K\exp\left(\frac{-2\epsilon^2 m}{C(d+d_{max})^2}\right) + 2d\exp\left(\frac{-2\epsilon^2 d_{min}}{C(d+d_{max})^2}\right)$$

where d_{max} (resp. d_{min}) represents the length (in number of occurrences) of the longest (resp. shortest) class in $\mathcal{Y}$, and C is a constant related to the longest document in $\mathcal{X}$.

Proof Using Lemma 1 and a Taylor expansion of the log function, one gets, $\forall \epsilon > 0$, $\forall \boldsymbol{x} \in \mathcal{X}$, $\forall k \in \mathcal{Y}$:

$$P\left(|l(\boldsymbol{x}, y_k) - l_\infty(\boldsymbol{x}, y_k)| < \sqrt{C}\frac{\epsilon}{\rho_0}\right) > 1 - \delta$$

where δ is the same as in Lemma 1, $\sqrt{C}$ equals to the maximum length of a document and $\rho_0 = \min_{i,k}\{P(y_k), P(w_i|y_k)\}$. The use of Laplace smoothing is important for the quantities $p(w_i|y_k)$, which may be null if word w_i is not observed in class y_k. The Laplace smoother in this case leads to $\rho_0 = \frac{1}{d+d_{max}}$. The log-likelihood functions of the multinomial, multiclass Naive Bayes classifier and the one of its asymptotic version are thus close to each other with high probability. The decision made by the trained Naive Bayes classifier and its asymptotic version on a given x only differ if the distance between the first two classes of the asymptotic classifier is less than two times the distance between the log-likelihood functions of the trained and asymptotic classifiers. Thus, using the union bound, one obtains, with probability at least $(1-\delta)$:

$$\mathcal{E}(h) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}\left(\epsilon\sqrt{C}(d+d_{max})\right)$$

Using a change of variable ($\epsilon' = \epsilon\sqrt{C}(d+d_{max})$) and approximating $\sum_{k=1}^{K}\exp(-2d_k\epsilon^2)$ by $\exp(-2d_{min}\epsilon^2)$, the dominating term in the sum, leads to the desired result. □

5.2 Asymptotic approximation error bounds for Multinomial Logistic Regression

We now propose an asymptotic approximation error bound for a multiclass logistic regression (`MLR`) classifier. We first consider the flat, multiclass case ($V = \mathcal{Y}$), and then show how the bounds can be combined in a typical top-down cascade, leading to the identification of important features that control the variation of these bounds.

Considering a pivot class $y^\star \in \mathcal{Y}$, a `MLR` classifier, with parameters $\boldsymbol{\beta} = \{\beta_0^y, \beta_j^y; y \in \mathcal{Y}\setminus\{y^\star\}, j \in \{1,\ldots,d\}\}$, models the class posterior probabilities via a linear function in $\mathbf{x} = (x_j)_{j=1}^d$ ((see for example Hastie et al., 2001, p. 96)) :

$$\begin{aligned} P(y|\mathbf{x};\boldsymbol{\beta})_{y\neq y^\star} &= \frac{\exp(\beta_0^y + \sum_{j=1}^d \beta_j^y x_j)}{1+\sum_{y'\in\mathcal{Y},y'\neq y^\star}\exp(\beta_0^{y'} + \sum_{j=1}^d \beta_j^{y'} x_j)} \\ P(y^\star|\mathbf{x};\boldsymbol{\beta}) &= \frac{1}{1+\sum_{y'\in\mathcal{Y},y'\neq y^\star}\exp(\beta_0^{y'} + \sum_{j=1}^d \beta_j^{y'} x_j)} \end{aligned}$$

The parameters $\boldsymbol{\beta}$ are usually fit by maximum likelihood over a training set $\mathcal{S}$ of size m (denoted by $\widehat{\boldsymbol{\beta}}_m$ in the following) and the decision rule for this classifier consists in choosing the class with the highest class posterior probability :

$$h_m(\mathbf{x}) = \operatorname*{argmax}_{y\in\mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) \tag{9}$$

The following lemma states to which extent the posterior probabilities with maximum likelihood estimates $\widehat{\boldsymbol{\beta}}_m$ may deviate from their asymptotic values obtained with maximum likelihood estimates when the training size m tends to infinity (denoted by $\widehat{\boldsymbol{\beta}}_\infty$).

Lemma 2 *Let $\mathcal{S}$ be a training set of size m and let $\widehat{\boldsymbol{\beta}}_m$ be the maximum likelihood estimates of the `MLR` classifier over $\mathcal{S}$. Further, let $\widehat{\boldsymbol{\beta}}_\infty$ be the maximum likelihood estimates of parameters of `MLR` when m tends to infinity. For all examples $\mathbf{x}$, let $R > 0$ be the bound such that $\forall y \in \mathcal{Y}\setminus\{y^\star\}, \exp(\beta_0^y + \sum_{j=1}^d \beta_j^y x_j) < \sqrt{R}$; then for all $1 > \delta > 0$, with probability at least $(1-\delta)$ we have:*

$$\forall y \in \mathcal{Y}, \left|P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) - P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)\right| < d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}$$

where $\sigma_0 = \max_{j,y}\sigma_j^y$ and $(\sigma_j^y)_{y,j}$ represent the components of the inverse (diagonal) Fisher information matrix at $\widehat{\boldsymbol{\beta}}_\infty$ and are different from σ_i used in Section 3 wherein these represented Rademacher random variables.

Proof By denoting the sets of parameters $\widehat{\boldsymbol{\beta}}_m = \{\hat{\beta}_j^y; j \in \{0,\ldots,d\}, y \in \mathcal{Y}\setminus\{y^\star\}\}$, and $\widehat{\boldsymbol{\beta}}_\infty = \{\beta_j^y; j \in \{0,\ldots,d\}, y \in \mathcal{Y}\setminus\{y^\star\}\}$, and using the independence assumption and the asymptotic normality of maximum likelihood estimates ((see for example Schervish, 1995, p. 421)), we have, for $0 \leq j \leq d$ and $\forall y \in \mathcal{Y}\setminus\{y^\star\}$: $\sqrt{m}(\widehat{\beta}_j^y - \beta_j^y) \sim N(0, \sigma_j^y)$ where the $(\sigma_j^y)_{y,i}$ represent the components of the inverse (diagonal) Fisher information matrix at $\widehat{\boldsymbol{\beta}}_\infty$. Let $\sigma_0 = \max_{j,y}\sigma_j^y$. Then using Chebyshev's inequality, for $0 \leq j \leq d$ and $\forall y \in \mathcal{Y}\setminus\{y^\star\}$ we have with probability at least $1 - \sigma_0/\epsilon^2$, $|\widehat{\beta}_j^y - \beta_j^y| < \frac{\epsilon}{\sqrt{m}}$. Further $\forall \mathbf{x}$ and $\forall y \in \mathcal{Y}\setminus\{y^\star\}, \exp(\beta_0^y + \sum_{j=1}^d \beta_j^y x_j) < \sqrt{R}$; using a Taylor development of the functions $\exp(x+\epsilon)$ and $(1+x+\epsilon x)^{-1}$ and the union bound, one obtains that, $\forall \epsilon > 0$ and $y \in \mathcal{Y}$ with probability at least $1 - \frac{|\mathcal{Y}|\sigma_0}{\epsilon^2}$: $\left|P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) - P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)\right| < d\sqrt{\frac{R}{m}}\epsilon$. Setting $\frac{|\mathcal{Y}|\sigma_0}{\epsilon^2}$ to δ, and solving for ϵ gives the result. □

Lemma 2 suggests that the predicted and asymptotic posterior probabilities are close to each other, as the quantities they are based on are close to each other. Thus, provided that the asymptotic posterior probabilities between the best two classes, for any given $\mathbf{x}$, are not too close to each other, the generalization error of the `MLR` classifier and the one of its asymptotic version should be similar. Theorem 4 below states such a relationship, using the following function that measures the confusion between the best two classes for the asymptotic `MLR` classifier defined as :

$$h_\infty(\mathbf{x}) = \operatorname*{argmax}_{y\in\mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty) \tag{10}$$

For any given $\mathbf{x} \in \mathcal{X}$, the confusion between the best two classes is defined as follows.

Definition 2 *Let $f_\infty^1(\mathbf{x}) = \max_{y\in\mathcal{Y}} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)$ be the best class posterior probability for $\mathbf{x}$ by the asymptotic `MLR` classifier, and let $f_\infty^2(\mathbf{x}) = \max_{y\in\mathcal{Y}\setminus h_\infty(\mathbf{x})} P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)$ be the second best class posterior probability for $\mathbf{x}$. We define the confusion of the asymptotic `MLR` classifier for a category set $\mathcal{Y}$ as:*

$$G_{\mathcal{Y}}(\tau) = P_{(\mathbf{x},y)\sim\mathcal{D}}(|f_\infty^1(\mathbf{x}) - f_\infty^2(\mathbf{x})| < 2\tau)$$

for a given $\tau > 0$.

The following theorem states a relationship between the generalization error of a trained `MLR` classifier and its asymptotic version.

Theorem 4 *For a multi-class classification problem in d dimensional feature space with a training set of size m, $\{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^{m}$, $\boldsymbol{x}^{(i)} \in \mathcal{X}$, $y^{(i)} \in \mathcal{Y}$, sampled i.i.d. from a probability distribution $\mathcal{D}$, let h_m and h_∞ denote the multiclass logistic regression classifiers learned from a training set of finite size m and its asymptotic version respectively, and let $\mathcal{E}(h_m)$ and $\mathcal{E}(h_\infty)$ be their generalization errors. Then, for all $1 > \delta > 0$, with probability at least $(1-\delta)$ we have:*

$$\mathcal{E}(h_m) \leq \mathcal{E}(h_\infty) + G_{\mathcal{Y}}\left(d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}\right) \tag{11}$$

where $\sqrt{R}$ is a bound on the function $\exp(\beta_0^y + \sum_{j=1}^{d}\beta_j^y x_j)$, $\forall \boldsymbol{x} \in \mathcal{X}$ and $\forall y \in \mathcal{Y}$, and σ_0 is a constant.

Proof The difference $\mathcal{E}(h_m) - \mathcal{E}(h_\infty)$ is bounded by the probability that the asymptotic `MLR` classifier h_∞ correctly classifies an example $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ randomly chosen from $\mathcal{D}$, while h_m misclassifies it. Using Lemma 2, for all $\delta \in (0,1), \forall \mathbf{x} \in \mathcal{X}, \forall y \in \mathcal{Y}$, with probability at least $1-\delta$, we have:

$$\left|P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_m) - P(y|\mathbf{x}, \widehat{\boldsymbol{\beta}}_\infty)\right| < d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}$$

Thus, the decision made by the trained MLR and its asymptotic version on an example $(\mathbf{x}, y)$ differs only if the distance between the two predicted classes of the asymptotic classifier is less than two times the distance between the posterior probabilities obtained with $\widehat{\boldsymbol{\beta}}_m$ and $\widehat{\boldsymbol{\beta}}_\infty$ on that example; and the probability of this is exactly $G_{\mathcal{Y}}\left(d\sqrt{\frac{R|\mathcal{Y}|\sigma_0}{\delta m}}\right)$, which upper-bounds $\mathcal{E}(h_m) - \mathcal{E}(h_\infty)$. □

Note that the quantity σ_0 in Theorem 4 represents the largest value of the inverse (diagonal) Fisher information matrix ((Schervish, 1995)), and thus corresponds to the inverse of the smallest value of the (diagonal) Fisher information matrix, which corresponds to the smallest amount of information one has on the estimation of each parameter $\widehat{\beta}_j^k$. This smallest amount of information is in turn related to the length (in number of occurrences) of the longest (resp. shortest) class in $\mathcal{Y}$ denoted respectively by d_{max} and d_{min} as, the smaller they are, the larger σ_0 is likely to be.

5.3 A learning based node pruning strategy

Let us now consider a hierarchy of classes and a top-down classifier making decisions at each level of the hierarchy. A node-based pruning strategy can be easily derived from the approximation bounds above. Indeed, any node v in the hierarchy $\mathcal{H} = (V, E)$ is associated with three category sets: its sibling categories with the node itself $\mathfrak{S}'(v) = \mathfrak{S}(v) \cup \{v\}$, its children categories, $\mathfrak{D}(v)$, and the union of its siblings and children categories, denoted $\mathfrak{F}(v) = \mathfrak{S}(v) \cup \mathfrak{D}(v)$.

These three sets of categories are the ones involved before and after the pruning of node v. Let us now denote by $h_m^{\mathfrak{S}'_v}$ a classifier learned from a set of sibling categories of node v and the node itself, and by $h_m^{\mathfrak{D}_v}$ a classifier learned from the set of children categories of node v ($h_\infty^{\mathfrak{S}'_v}$ and $h_\infty^{\mathfrak{D}_v}$ respectively denote their asymptotic versions). The following theorem is a direct extension of Theorems 3 and 4 to this setting.

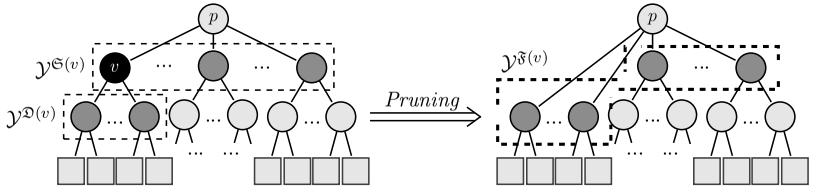


Figure 4: The pruning procedure for a candidate class node u (in black). After replacing the candidate node by its children, the new category set $\mathcal{Y}^{\mathfrak{F}(v)}$ contains the classes from both the daughter and the sister category sets of v.

Theorem 5 *Using notations from both Theorems 3 and 4, $\forall \epsilon > 0$, $v \in V \setminus \mathcal{Y}$, one has:*

$$\mathcal{E}(h_m^{\mathfrak{S}'_v}) + \mathcal{E}(h_m^{\mathfrak{D}_v}) \leq \mathcal{E}(h_\infty^{\mathfrak{S}'_v}) + \mathcal{E}(h_\infty^{\mathfrak{D}_v}) + G_{\mathfrak{S}'(v)}(\epsilon) + G_{\mathfrak{D}(v)}(\epsilon)$$

with probability at least $1 - \left(\frac{Rd^2|\mathfrak{S}'(v)|\sigma_0^{\mathfrak{S}'(v)}}{m_{\mathfrak{S}'(v)}\epsilon^2} + \frac{Rd^2|\mathfrak{D}(v)|\sigma_0^{\mathfrak{D}(v)}}{m_{\mathfrak{D}(v)}\epsilon^2}\right)$ for `MLR` classifiers and with probability at least $1 - \left(\delta_{\mathfrak{S}'(v)} + \delta_{\mathfrak{D}(v)}\right)$ for Naive Bayes classifiers, with $\delta_{\mathcal{Y}}$ defined in Theorem 3.

$\{|\mathcal{Y}^\ell|, m_{\mathcal{Y}^\ell}, \sigma_0^{\mathcal{Y}^\ell}, d_{max}^\ell, d_{min}^\ell; \mathcal{Y}^\ell \in \{\mathfrak{S}'(v), \mathfrak{D}(v)\};\}$ are constants related to the set of categories $\mathcal{Y}^\ell \in \{\mathfrak{S}'(v), \mathfrak{D}(v)\}$ and involved in the respective bounds stated in Theorem 3 and 4. Denoting by $h_m^{\mathfrak{F}_v}$ the classifier trained on the set $\mathfrak{F}(v)$ and by $h_\infty^{\mathfrak{F}_v}$ its asymptotic version, Theorem 5 suggests that one should prune node v if:

$$G_{\mathfrak{F}(v)}(\epsilon) \leq G_{\mathfrak{S}'(v)}(\epsilon) + G_{\mathfrak{D}(v)}(\epsilon) \tag{12a}$$

and, for `MLR` classifiers:

$$\frac{|\mathfrak{F}(v)|\sigma_0^{\mathfrak{F}(v)}}{m_{\mathfrak{F}(v)}} \leq \frac{|\mathfrak{S}'(v)|\sigma_0^{\mathfrak{S}'(v)}}{m_{\mathfrak{S}'(v)}} + \frac{|\mathfrak{D}(v)|\sigma_0^{\mathfrak{D}(v)}}{m_{\mathfrak{D}(v)}} \tag{12b}$$

or, for Naive Bayes classifiers:

$$\delta_{\mathfrak{F}(v)} \leq \delta_{\mathfrak{S}'(v)} + \delta_{\mathfrak{D}(v)} \tag{12c}$$

The above conditions for pruning a node v rely on the union bound and thus are not likely to be exploitable in practice. They nevertheless exhibit the factors that play an important role in assessing whether a particular trained classifier is close or not to its asymptotic version. Following Definitions 1 and 2, $G_{\mathcal{Y}}(\epsilon)$ is of the form:

$$G_{\mathcal{Y}}(\epsilon) = P_{(\mathbf{x},y)\sim\mathcal{D}}(|g_\infty(\mathbf{x}) - g_\infty(\mathbf{x})| < 2\epsilon)$$

where g corresponds to the best log-likelihood for Naive Bayes classifiers or the best class posterior for `MLR` classifiers. As it relies on the unknown distribution $\mathcal{D}$ and asymptotic log-likelihood $g_\infty()$, $G_{\mathcal{Y}}(.)$ cannot be computed directly. It however measures the confusion

Features for node $v \in V$	
1. # of classes in level ℓ, K^ℓ	2. Vocabulary size, d^ℓ
3. # of docs m^ℓ	4. d^ℓ_{min}
5. d^ℓ_{max}	6. $\frac{m^\ell}{(d^\ell+d^\ell_{max})^2}$
7. $\frac{d^\ell_{min}}{(d^\ell+d^\ell_{max})^2}$	8. $cos_{inter}(\ell)$
9. $\mathrm{KL}_{inter}(\ell)$	

Table 1: Features involved in the vector representation of a node $v \in \mathcal{H} = (V, E)$. As $\ell \in \{\mathfrak{F}(v), \mathfrak{D}(v), \mathfrak{S}(v)\}$, we have in total 27 features associated with the different category sets considered for flattening node u.

between categories and can thus be approximated by measures of the similarity between classes. We propose here to estimate this confusion with two simple quantities: the average cosine similarity of all the pairs of classes in $\mathcal{Y}$, and the average symmetric Kullback-Leibler divergences between all the pairs in $\mathcal{Y}$ according to class conditional multinomial distributions. Each node $v \in V$ when deployed with `MLR` and Naive Bayes classifiers can then be characterized by factors shown in Table 1, which are involved in the estimation of inequality (12) above.

The average cosine similarity and the average symmetric KL divergence are calculated as follows for each pair of nodes (u, v) in level l:

$$cos_{inter}(\ell) = \frac{1}{\sum_{k=1}^{K^\ell}\sum_{k'=1k'\neq k}^{K^\ell} m_{u_k^\ell} m_{v_{k'}^\ell}} \sum_{k=1}^{K^\ell}\sum_{k'=1k'\neq k}^{K^\ell} m_{u_k^\ell} m_{v_{k'}^\ell} cos(u_k^\ell, v_{k'}^\ell)$$

$$KL_{inter}(\ell) = \frac{1}{K^\ell(K^\ell-1)} \sum_{k=1}^{K^\ell}\sum_{k'=1k'\neq k}^{K^\ell} KL(Q_{u_k^\ell}||Q_{v_{k'}^\ell}) + KL(Q_{v_{k'}^\ell}||Q_{u_k^\ell}),$$

where $KL(Q_u||Q_v)$ denotes the Kullback-Leibler divergence between the class conditional probability distributions of the features present in the two classes u and v:

$$KL(Q_u||Q_v) = \sum_{w\in u,v} p(w|u) \log \frac{p(w|u)}{p(w|v)}$$

where $p(w|u)$ denotes the probability of word/feature w in class u, taking smoothing into account.

Algorithm 3 presents the process of learning the hierarchy pruning by learning a meta-classifier from the meta-features as mentioned above. The procedure for collecting training data associates a positive (resp. negative) class to a node if the pruning of that node leads to a final performance increase (resp. decrease). A meta-classifier is then trained on these features using a training set from a selected class hierarchy. After the learning phase, the meta-classifier is applied to each node of a new hierarchy of classes so as to identify which

Algorithm 3 The pruning strategy.

```
1:  procedure PRUNE HIERARCHY(a hierarchy H, a meta-classifier C_m)
2:      clist[] ← H.root;                                  ▷ Initialize with root node
3:      while !clist.isEmpty() do
4:          parent ← clist.getNext()
5:          list[] ← Ch(parent);                           ▷ Candidate children to merge
6:          while !list.isEmpty() do
7:              index ← MERGE(parent,list,C_m);            ▷ Index of children to be merged
8:              if index == −1 then
9:                  break;
10:             end if
11:             list.add(Ch(list[index]))                  ▷ Move up the children of node list[index]
12:             list.remove(index);                        ▷ This node has been merged
13:         end while
14:         clist.add(Ch(clist[j]));                       ▷ Adds next level parents
15:     end while
16:     export new hierarchy;
17: end procedure
```

nodes should be pruned. A simple strategy to adopt is then to prune nodes in sequence: starting from the root node, the algorithm checks which children of a given node v should be pruned (lines 6-13) by creating the corresponding meta-instance and feeding the meta-classifier; the child that maximizes the probability of the positive class is then pruned; as the set of categories has changed, we recalculate which children of v can be pruned, prune the best one (as above) and iterate this process till no more children of v can be pruned; we then proceed to the children of v and repeat the process. The function MERGE takes as arguments the parent, the candidate children to be merged as well as the meta-classifier. It returns the index of the child that should be merged. In case where the meta-classifier does not identify any child eligible for merging then the pruning procedure continues with the next parent in the list. The meta-classifier tries to predict whether pruning a specific node will lead to an increase of the performance over 10%. That means that the classifier may identify several candidate nodes and one of them will be selected for pruning.

6. Experimental Analysis

We start our discussion by presenting results on different hierarchical datasets with different characteristics using `MLR` and `SVM` classifiers. The datasets we used in these experiments are two large datasets extracted from the International Patent Classification (**IPC**) dataset[3] and the publicly available DMOZ dataset from the second LSHTC challenge (**LSHTC2**)[4]. Both datasets are multi-class; **IPC** is single-label and **LSHTC2** multi-label with an average of 1.02 categories per class. We created 5 datasets from **LSHTC2** by splitting randomly the first layer nodes (11 in total) of the original hierarchy in disjoint subsets. The classes

3. `http://www.wipo.int/classifications/ipc/en/support/`
4. `http://lshtc.iit.demokritos.gr/`

Dataset	# Tr.	# Test	# Classes	# Feat.	Depth	CR	Error ratio
LSHTC2-1	25,310	6,441	1,789	145,859	6	0.008	1.24
LSHTC2-2	50,558	13,057	4,787	271,557	6	0.003	1.32
LSHTC2-3	38,725	10,102	3,956	145,354	6	0.004	2.65
LSHTC2-4	27,924	7,026	2,544	123,953	6	0.005	1.8
LSHTC2-5	68,367	17,561	7,212	192,259	6	0.002	2.12
IPC	46,324	28,926	451	1,123,497	4	0.02	12.27

Table 2: Datasets used in our experiments along with the properties: number of training examples, test examples, classes and the size of the feature space, the depth of the hierarchy and the complexity ratio of hierarchical over the flat case ($\sum_{v\in V\setminus\mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)|-1)/|\mathcal{Y}|(|\mathcal{Y}|-1)$), the ratio of empirical error for hierarchical and flat models.

for the **IPC** and **LSHTC2** datasets are organized in a hierarchy in which the documents are assigned to the leaf categories only. Table 2 presents the characteristics of the datasets.

CR denotes the complexity ratio between hierarchical and flat classification, given by the Rademacher complexity term in Theorem 1: $\left(\sum_{v\in V\setminus\mathcal{Y}} |\mathfrak{D}(v)|(|\mathfrak{D}(v)|-1)\right) / (|\mathcal{Y}|(|\mathcal{Y}|-1))$; the same constants B, R and L are used in the two cases. As one can note, this complexity ratio always goes in favor of the hierarchical strategy, although it is 2 to 10 times higher on the **IPC** dataset, compared to **LSHTC2-1,2,3,4,5**. On the other hand, the ratio of empirical errors (last column of Table 2) obtained with top-down hierarchical classification over flat classification when using `SVM` with a linear kernel is this time higher than 1, suggesting the opposite conclusion. The error ratio is furthermore really important on **IPC** compared to **LSHTC2-1,2,3,4,5**. The comparison of the complexity and error ratios on all the datasets thus suggests that the flat classification strategy may be preferred on **IPC**, whereas the hierarchical one is more likely to be efficient on the **LSHTC** datasets. This is indeed the case, as is shown below.

To test our simple node pruning strategy, we learned binary classifiers aiming at deciding whether to prune a node, based on the node features described in the previous section. The label associated to each node in this training set is defined as +1 if pruning the node increases the accuracy of the hierarchical classifier by at least 0.1, and -1 if pruning the node decreases the accuracy by more than 0.1. The threshold at 0.1 is used to avoid too much noise in the training set. The meta-classifier is then trained to learn a mapping from the vector representation of a node (based on the above features) and the labels $\{+1; -1\}$. We used the first two datasets of **LSHTC2** to extract the training data while **LSHTC2-3**, **4**, **5** and **IPC** were employed for testing.

The procedure for collecting training data is repeated for the `MNB`, `MLR` and `SVM` classifiers resulting in three meta-datasets of 119 (19 positive and 100 negative), 89 (34 positive and 55 negative) and 94 (32 positive and 62 negative) examples respectively. For the binary classifiers, we used AdaBoost with random forest as a base classifier, setting the number of trees to 20, 50 and 50 for the `MNB`, `MLR` and `SVM` classifiers respectively and leaving the other parameters at their default values. Several values have been tested for the number

of trees ({10, 20, 50, 100 and 200}), the depth of the trees ({unrestricted, 5, 10, 15, 30, 60}), as well as the number of iterations in AdaBoost ({10, 20, 30}). The final values were selected by cross-validation on the training set (**LSHTC2-1** and **LSHTC2-2**) as the ones that maximized accuracy and minimized false-positive rate in order to prevent degradation of accuracy. For example, Table 6 presents the true positive and false positive rates for the two classes for the meta-dataset of `MLR`.

Class	TP rate	FP rate
Prune (+1)	0.737	0.020
Do not prune (-1)	0.980	0.263

Table 3: True positive and false positive rates for the `MLR` meta-daset (119 examples).

We consider three different classifiers which include Multinomial Naive Bayes (`MNB`), Multi-class Logistic Regression (`MLR`) and Support Vector Machine (`SVM`) classifiers. The configurations of the taxonomy that we consider are fully flat classifier (FL), fully hierarchical (FH) top-down *Pachinko* machine, a random pruning (RN), and the two proposed pruning methods which include (i) Bound-based pruning strategy (PR-B) given in Section 4 and (ii) Meta-learning based pruning strategy (PR-M) proposed in Algorithm 3. For the PR-M pruning method, our experimental setup involves two scenarios: (i) The meta-classifier is trained over one kind of DMOZ hierarchy (LSHTC2-1, and LSHTC-2) and tested over another DMOZ hierarchy (LSHTC-3, LSHTC-4 and LSHTC-5), such that both sets have similar characteristics, and secondly, (ii) The meta-classifier is trained over one set of DMOZ hierarchy (LSHTC2-1, and LSHTC-2) and tested over IPC hierarchy which is derived from a different domain (patent classification) such that both sets have much less common characteristics. In this sense, our meta-classifier could learn to learn over one doamin and apply the learnt knowledge to another domain. For the random pruning we restrict the procedure to the first two levels and perform randomly prune 4 nodes (this is the average number of nodes that are pruned in the PR-M and PR-B strategies). We also present results for another pruning strategy proposed in our earlier work (Babbar et al., 2013b) which is based on error analysis of Perceptron Decision Trees (Bennett et al., 2000), and is referred to as PR-P. The results for the naive pruning method based on estimate on a validation set (as described in Section 4.1) are also presented, and referred to as PR-V. For each dataset we perform 5 independent runs for the random pruning and we record the best performance. For `MLR` and `SVM`, we use the LibLinear library (Fan et al., 2008) and use squared hinge-loss with $L2$-regularized versions, setting the penalty parameter C by cross-validation.

6.1 Flat versus Hierarchical classification

The classification error results on the test set of **LSHTC2-3,4,5** and **IPC** are reported in Table 4. On all **LSHTC** datasets flat classification performs worse than the fully hierarchy top-down classification, for all classifiers. These results are in line with complexity and empirical error ratios for `SVM` estimated on different collections and shown in table 2 as well as with the results obtained in Liu et al. (2005); Dumais and Chen (2000) over the same type

	LSHTC2-3			LSHTC2-4			LSHTC2-5			IPC		
	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM
FL	73.0$^{\downarrow\downarrow}$	52.8$^{\downarrow\downarrow}$	53.5$^{\downarrow\downarrow}$	84.9$^{\downarrow\downarrow}$	49.7$^{\downarrow\downarrow}$	50.1$^{\downarrow\downarrow}$	83.9$^{\downarrow\downarrow}$	54.2$^{\downarrow\downarrow}$	54.7$^{\downarrow\downarrow}$	67.2$^{\downarrow\downarrow}$	54.6	**46.6**
RN	61.9$^{\downarrow\downarrow}$	49.3$^{\downarrow\downarrow}$	51.7$^{\downarrow\downarrow}$	70.5$^{\downarrow\downarrow}$	47.8$^{\downarrow\downarrow}$	48.4$^{\downarrow\downarrow}$	69.0$^{\downarrow\downarrow}$	53.2$^{\downarrow\downarrow}$	53.6$^{\downarrow}$	64.3$^{\downarrow\downarrow}$	54.7$^{\downarrow}$	50.2$^{\downarrow\downarrow}$
FH	62.0$^{\downarrow\downarrow}$	48.4$^{\downarrow\downarrow}$	49.8$^{\downarrow\downarrow}$	68.3$^{\downarrow}$	47.3$^{\downarrow\downarrow}$	47.6$^{\downarrow}$	65.6$^{\downarrow}$	52.6$^{\downarrow}$	52.7	64.4$^{\downarrow}$	55.2$^{\downarrow}$	51.3$^{\downarrow\downarrow}$
PR-V	61.7	48.2	49.5	65.9	46.7	46.6	67.7	52.3	52.2	63.7	54.6	50.3
PR-B	-	48.1	49.5	-	**46.6**	**46.5**	-	**52.2**	**52.2**	-	54.5	50.5
PR-M	61.3	**48.0**	**49.3**	65.4	46.9	47.2	67.8	**52.2**	52.3	63.9	**54.4**	50.7
PR-P	-	48.3	49.6	-	**46.6**	46.8	-	52.4	52.3	-	54.5	50.9

Table 4: Error results across all datasets. Bold typeface is used for the best results. Statistical significance (using micro sign test (s-test) as proposed in (Yang and Liu, 1999)) is denoted with $^{\downarrow}$ for p-value<0.05 and with $^{\downarrow\downarrow}$ for p-value<0.01.

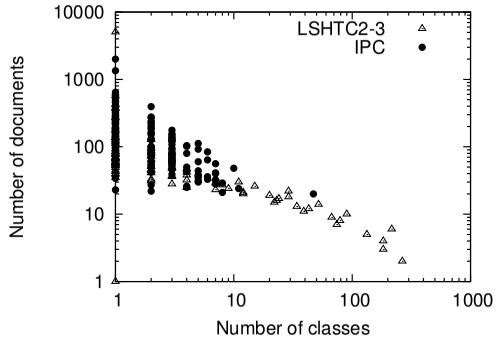


Figure 5: Number of classes (on X-axis) which have the specified number of documents (on Y-axis) for **LSHTC2-3** dataset and **IPC** dataset

of taxonomies. Further, the work by Liu et al. (2005) demonstrated that class hierarchies on **LSHTC** datasets suffer from *rare categories* problem, i.e., 80% of the target categories in such hierarchies have less than 5 documents assigned to them. The value for Macro-F1 measure which weighs all leaf-level classes equally (in contrast to Micro-F1 which weighs each example in the test set equally) is given in Table 5. Macro-F1 measure is particularly interesting when dealing with datasets consisting of rare-categories, which is typically the case in most naturally occurring category systems such as DMOZ and Wikipedia.

As a result, flat methods on such datasets face unbalanced classification problems which results in smaller error ratios; hierarchical classification should be preferred in this case. On the other hand, for hierarchies such as the one of **IPC**, which are relatively well balanced

	LSHTC2-3			LSHTC2-4			LSHTC2-5			IPC		
	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM	MNB	MLR	SVM
FL	17.1$^{\downarrow\downarrow}$	31.1$^{\downarrow\downarrow}$	31.6 $^{\downarrow\downarrow}$	15.1$^{\downarrow\downarrow}$	33.1 $^{\downarrow\downarrow}$	32.9$^{\downarrow\downarrow}$	15.0$^{\downarrow\downarrow}$	29.2$^{\downarrow\downarrow}$	29.1$^{\downarrow\downarrow}$	25.8 $^{\downarrow\downarrow}$	47.9	**51.2**
RN	20.2$^{\downarrow\downarrow}$	32.2$^{\downarrow\downarrow}$	31.9$^{\downarrow}$	19.2$^{\downarrow}$	33.6 $^{\downarrow}$	33.2$^{\downarrow\downarrow}$	18.1$^{\downarrow}$	29.9$^{\downarrow\downarrow}$	29.9$^{\downarrow\downarrow}$	26.1 $^{\downarrow}$	45.2$^{\downarrow\downarrow}$	47.8$^{\downarrow\downarrow}$
FH	22.1$^{\downarrow}$	32.8$^{\downarrow}$	32.2	20.1$^{\downarrow}$	34.1$^{\downarrow}$	33.7$^{\downarrow}$	18.9$^{\downarrow}$	30.5$^{\downarrow}$	30.7	26.2 $^{\downarrow}$	44.2$^{\downarrow}$	46.5$^{\downarrow}$
PR-V	22.3	33.0	32.1	21.2	34.6	34.3	19.4	31.7	31.7	26.4	48.1	48.1
PR-B	-	33.1	32.3	-	34.7	**34.4**	-	**31.8**	**31.9**	-	48.1	47.9
PR-M	22.4	**33.2**	**32.4**	21.2	**34.8**	34.3	19.3	31.7	31.8	26.5	**48.2**	48.7
PR-P	-	33.0	32.2	-	34.4	34.3	-	31.6	31.5	-	48.0	47.6

Table 5: Macro-F1 results across all datasets. Bold typeface is used for the best results. Statistical significance (using macro-level t-test as proposed in (Yang and Liu, 1999)) is denoted with $^{\downarrow}$ for p-value<0.05 and with $^{\downarrow\downarrow}$ for p-value<0.01.

and do not suffer from the rare categories phenomenon, flat classification performs at par or even better than hierarchical classification. The difference in the distribution of data among leaf-level categories for the **LSHTC** datasets and **IPC** dataset is illustrated in Figure 5 on log-log scale. As one can note, in most categories **IPC** have a lot (from tens to few hundreds) of documents which belong to them as denoted by the triangles. On the other hand, **LSHTC2-3** dataset has a lot of classes with a small number (1 or 2) of documents as shown by the high concentration of solid dots near the Y-axis. In this respect, the fit to power-law distribution of documents among categories in large-scale taxonomies has also been studied recently (Babbar et al., 2014a). The relative performance between the flat and top-down approaches on the two kinds of datasets is in agreement with the conclusions obtained in recent studies in which the datasets considered do not have *rare categories* and are more well-balanced (Bengio et al., 2010; Gao and Koller, 2011; Perronnin et al., 2012; Deng et al., 2011).

We would also like to mention that using the top-down classification for mono-label prediction, we obtained an accuracy of 36.6% on the original LSHTC2 dataset. On the other hand, the top approach was a neural network based approach which used the echo-state network, and achieved an accuracy of 38.8%. Unlike the participants in LSHTC whose goal was to maximize the accuracy by using techniques such as ensemble methods and feature engineering, our goal in this work is to theoretical analyze flat versus hierarchical classification and extend the framework for performing taxonomy adaptation. Furthermore, the original LSHTC2 dataset was mildly multi-label, and since our top-down prediction algorithm predicts only a single label, we would loose some performance as compared to algorithms which are aimed towards making multi-label predictions.

6.2 Effect of pruning

The proposed hierarchy pruning strategies aim to adapt the given taxonomy structure for better classification while maintaining the ancestor-descendant relationship between a given pair of nodes. We compare the four pruning strategies, (i) first, based on estimation

on a validation set (PR-V) as given in Section 4.1, (ii) second, based on minimizing the Rademacher-based generalization error bound (PR-B) as given in Section 4.2, (iii) third, based on meta-learning (PR-M) as described in Section 5, and (iv) fourth, based on a method presented by Babbar et al. (2013b) (PR-P) against the random pruning (RN) and fully hierarchical (FH) classification. As shown in Table 4, the proposed pruning strategies lead to statistically significant better results for all three classifiers compared to both the original taxonomy and a randomly pruned one. Since Rademacher-bound based pruning strategy, PR-B, is similar to PR-P proposed by Babbar et al. (2013b), they have almost similar performance on most datasets. A similar result is reported by Wang and Lu (2010) through a pruning of an entire layer of the hierarchy, which can be seen as a generalization, even though empirical in nature, of the pruning strategy retained here. Another interesting approach to modify the original taxonomy is presented by Zhang et al. (2006). In this study, three other elementary modification operations are considered, again with an increase of performance. It is also worth noticing that the pruning strategy based on estimation on a validation set also leads to improvement in classification accuracy. However, as discussed earlier, the method is computationally expensive and unlike the meta-learning based pruning method, this does not amount to a learning algorithm which can be applied to unseen but similar taxonomies.

For `MNB` classifier, one can notice that the proposed pruning method (PR-M) based on meta-learning has the best performance in all datasets achieving significantly better results compared to its rivals. This shows that flattening the hierarchy can boost the performance, even in situations where the fully hierarchical classifier is better than its flat version (this is the case for all the datasets considered for `MNB`). The random pruning achieves slightly better accuracies than FH in **LSHTC2-3** and **IPC** datasets, but is in general in between the performance of the flat classifier and its fully hierarchical version. Statistical significance tests report significant differences in favor of the proposed approach for pruning. We also observe that all hierarchical methods consistently outperform the flat case. This is an expected result as the flat `MNB` classifier suffers from the problem of unbalanced data. The difference between the performance of the flat `MNB` classifier and its hierarchical versions is less marked for the **IPC** dataset.

For `MLR` and `SVM` classifiers, both pruning approaches have better performance in all datasets compared to its rivals, the difference being significant in all cases but with the flat classifier on **IPC**. One can also notice that due to the balanced nature of the **IPC** dataset, the performance of the flat classifier is close to that of hierarchical methods. For the same reason, random pruning is also more effective in the **IPC** dataset as compared to other datasets. Comparing the respective behaviors of the `MLR` and `SVM` against `MNB`, one can note that `MLR` and `SVM` are more robust to variations in the taxonomy as compared to `MNB`. This is reflected in much lesser variation in the accuracy for these classifiers under different configurations of the hierarchy. Lastly, and not surprisingly, the performance of `MLR` and `SVM` are much better than that of `MNB` on all the datasets considered here.

Here we would like to stress that we follow a greedy strategy in both the pruning methods (PR-B and PR-M) which rank the sibling nodes in the order these should be pruned. In case of PR-B, this is measured using the confusion $C(v)$ for each node and for PR-M, this is given by the confidence of the meta-classifier which return this as a probability estimate. This is due to the fact that picking up the optimal subset of nodes to prune from

all possible subsets of children nodes (say K') would require considering all possible $2^{K'}$ subsets. Furthermore, considering all possible orders to pick the nodes one-by-one would require considering all possible permutations of ordering. Both of these have exponential computational complexity in the number of children nodes.

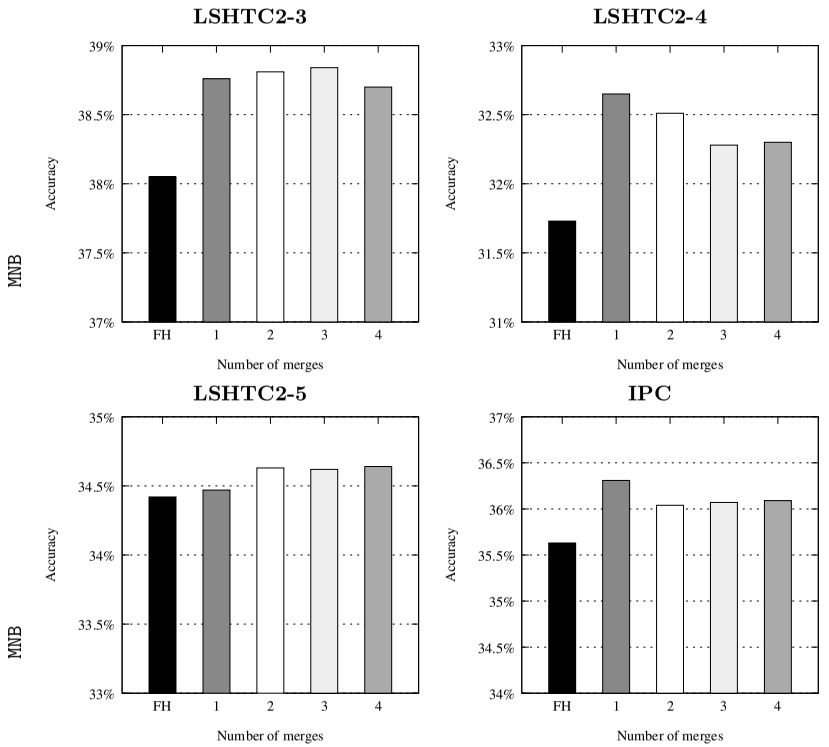


Figure 6: Accuracy performance with respect to the number of pruned nodes for `MNB` on different test sets.

6.3 Effect of number of pruned nodes for meta-learning based pruning strategy

For studying how the performance changes according to the number of pruned nodes, we record the accuracy of the proposed pruning method for 1 to 4 number of prunings. Note that pruning of nodes is done in sequence and is not independent. The results for both

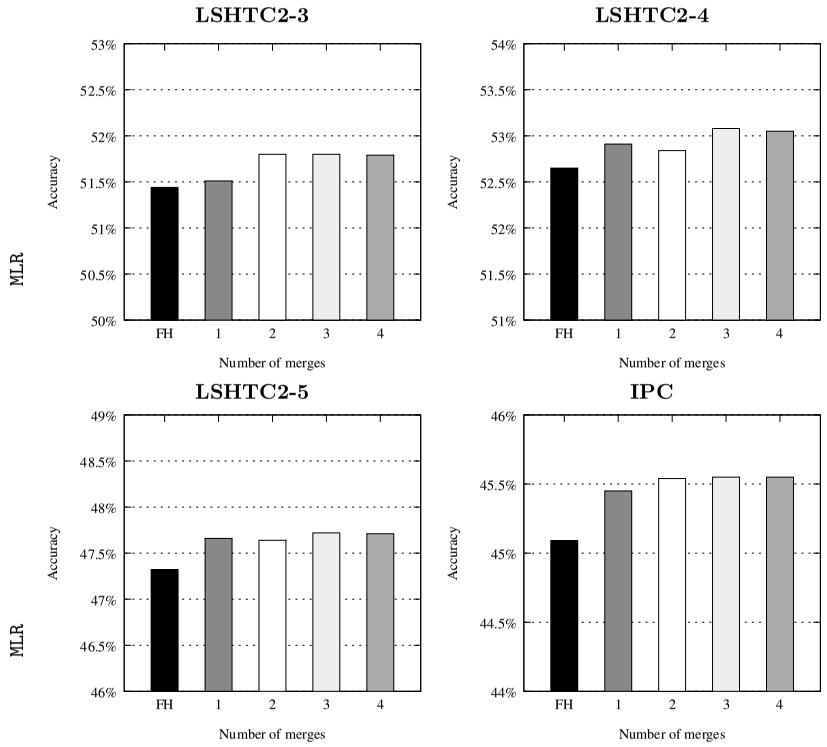


Figure 7: Accuracy performance with respect to the number of pruned nodes for MLR (down) on different test sets.

MNB and MLR are depicted in Figures 6 and 7, with a comparison to the FH method. The comparison with SVM is not explicitly shown as its behavior is similar to MLR classifier.

Interestingly, across all datasets, the proposed method has better performance than FH for all number of pruning nodes for both MNB and MLR. This shows that the proposed method is able to select appropriate nodes in the hierarchy for pruning. Additionally, we note that in the majority of cases the first pruned node provides a higher increase in accuracy than the following nodes. This is an expected behavior as the first prunings are typically performed at the upper level and thus tend to have a higher impact (as they will be used in more in the classification of more documents) than the nodes pruned done at lower levels. We want to stress here the fact that the performance with respect to the number of pruned nodes is affected by several factors, as the accuracy of the meta-classifier, the level of the hierarchy

Dataset	Logistic			Hinge		
	PR-M	LOMTree	FilterTree	PR-M	LOMTree	FilterTree
LSHTC2-3	48.1	60.6	60.3	49.3	60.2	59.8
LSHTC2-4	46.9	72.7	73.6	47.2	72.7	72.6
LSHTC2-5	52.2	72.0	72.7	52.3	70.5	71.0
IPC	54.4	73.9	65.1	50.7	68.8	67.1

Table 6: Comparison of classification errors for the proposed pruning method (PR-M) with approaches that build the hierarchical structure.

where the nodes are pruned and their sequence. For example, in dataset **LSHTC2-4** (Figure 6), there is a drop of performance after the first flattening which we believe is due to false positives provided by the meta-classifier. As shown for MLR in Figure 7 and across all datasets that the behavior of the pruning method is more stable without decrease in the final performance.

6.4 Building Taxonomy

We also compare the proposed method with approaches that construct a hierarchy with logarithmic depth over the labels of the problem. Such methods are extremely useful in cases where there is no hierarchical structured information available. More specifically, we compare with LOMTree (Choromanska and Langford, 2014) and FilterTree (Beygelzimer et al., 2009b), both implemented in the Vowpal Wabbit open source system.[5] LOMTree creates binary trees assigning each time the examples to the two respective children of a parent node. When the algorithm decides to create a leaf node it assigns the target label from $\mathcal{Y}$ that corresponds to the most frequent one amongst the examples reaching that node. On the other hand FilterTree follows a bottom-up partition process. For both methods we experiment with hinge and logistic loss functions using different step sizes ($\{0.15, 0.25, 0.5, 0.75, 1, 2, 4, 8\}$) and up to 32 passes through the data. For the LOMTree we use different settings for final number of non-leaf nodes and for the swap resistance. Finally, for both methods we use bit precision of 30.

Table 6.4 presents the error rate for the pruning algorithm PR-M and the tree approaches LOMTree and FilterTree across all the datasets for logistic and hinge loss functions. In all cases PR-M which is based on the provided hierarchy outperforms the tree construction approaches having a large difference in their errors. The results strongly indicate that in the cases where the hierarchy is provided it is preferable to use in order to perform logarithmic prediction rather than constructing it as it leads to loss of accuracy.

7. Conclusion

We have studied in this paper flat and hierarchical classification strategies from a learning-theoretic view point in the context of large-scale taxonomies, through error generalization

5. http://hunch.net/vw

bounds of multiclass, hierarchical classifiers. The first theorem we have introduced provides an explanation to several empirical results related to the performance of such classifiers. We also introduced two methods to simplify a taxonomy by selectively pruning some of its nodes, (i) by exploiting the bound developed in the first theorem, and (ii) by designing a meta-learning technique which is based on the features derived from the approximation-error based generalization bounds proposed in Sections 5.1 and 5.2. The experimental results reported here (as well as in other papers) are in line with our theoretical developments and justify the pruning strategy adopted.

In addition to theoretically addressing the flat versus top-down classification for large-scale taxonomies, the focus of this work is also on the problem of aligning the taxonomy of classes to the set of input-output pairs. This can be useful in designing better taxonomies for large-scale classification problems. Lastly, this suggests that our theoretical development can also be exploited to grow a hierarchy of classes from a (large) set of categories, as has been done in several studies (like for example Bengio et al. (2010); Choromanska and Langford (2014)). We plan to explore this in future work.

Acknowledgments

This work was supported in part by the ANR project Class-Y, the Mastodons project Garguantua, the LabEx PERSYVAL-Lab ANR-11-LABX-0025 and the European project BioASQ (grant agreement no. 318652). The authors sincerely thank the anonymous reviewers for providing valuable feedback which has led to considerable improvement in the content of this paper.

References

Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pages 13–24. International World Wide Web Conferences Steering Committee, 2013.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. On flat versus hierarchical classification in large-scale taxonomies. In *Advances in Neural Information Processing Systems*, pages 1824–1832, 2013a.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. Maximum-margin framework for training data synchronization in large-scale hierarchical classification. In *Neural Information Processing*, pages 336–343. Springer, 2013b.

Rohit Babbar, Cornelia Metzig, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. On power law distributions in large-scale taxonomies. *ACM SIGKDD Explorations Newsletter*, 16(1):47–56, 2014a.

Rohit Babbar, Ioannis Partalas, Eric Gaussier, and Massih-Reza Amini. Re-ranking approach to classification in large-scale power-law distributed category systems. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1059–1062. ACM, 2014b.

Peter L. Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

Samy Bengio, Jason Weston, and David Grangier. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems 23*, pages 163–171, 2010.

Kristin P Bennett, Nello Cristianini, John Shawe-Taylor, and Donghui Wu. Enlarging the margins in perceptron decision trees. *Machine Learning*, 41(3):295–313, 2000.

Paul N. Bennett and Nam Nguyen. Refined experts: improving classification in large taxonomies. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 11–18, 2009.

Alina Beygelzimer, John Langford, Yuri Lifshits, Gregory Sorkin, and Alexander Strehl. Conditional probability tree estimation analysis and algorithms. In *Proceedings of the Twenty-Fifth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 51–58, Corvallis, Oregon, 2009a. AUAI Press.

Alina Beygelzimer, John Langford, and Pradeep D. Ravikumar. Error-correcting tournaments. In *Algorithmic Learning Theory, 20th International Conference, ALT 2009, Porto, Portugal, October 3-5, 2009. Proceedings*, pages 247–262, 2009b.

Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings* 13^{th} *ACM International Conference on Information and Knowledge Management (CIKM)*, pages 78–87. ACM, 2004.

Anna Choromanska and John Langford. Logarithmic time online multiclass prediction. *CoRR*, abs/1406.1822, 2014. URL `http://arxiv.org/abs/1406.1822`.

Bhavana Dalvi and William W Cohen. Hierarchical semi-supervised classification with incomplete class hierarchies. *In preparation*, 2014.

Ofer Dekel. Distribution-calibrated hierarchical classification. In *Advances in Neural Information Processing Systems 22*, pages 450–458, 2009.

Ofer Dekel, Joseph Keshet, and Yoram Singer. Large margin hierarchical classification. In *Proceedings of the* 21^{st} *International Conference on Machine Learning*, pages 27–35, 2004.

Jia Deng, Sanjeev Satheesh, Alexander C. Berg, and Fei-Fei Li. Fast and balanced: Efficient label tree learning for large scale object recognition. In *Advances in Neural Information Processing Systems 24*, pages 567–575, 2011.

Susan Dumais and Hao Chen. Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263. ACM, 2000.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Francois Fleuret and Donald Geman. Coarse-to-fine face detection. *International Journal of computer vision*, 41(1-2):85–107, 2001.

Tianshi Gao and Daphne Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2072–2079, 2011.

Siddharth Gopal and Yiming Yang. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 257–265. ACM, 2013.

Siddharth Gopal, Yiming Yang, Bing Bai, and Alexandru Niculescu-Mizil. Bayesian models for large-scale hierarchical classification. In *Advances in Neural Information Processing Systems 25*, 2012.

Yann Guermeur. Sample complexity of classifiers taking values in $\mathbb{R}^q$, application to multiclass SVMs. *Communications in Statistics - Theory and Methods*, 39, 2010.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer New York Inc., 2001.

Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, 1997.

Balaji Krishnapuram, Lawrence Carin, Mario AT Figueiredo, and Alexander J Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(6):957–968, 2005.

Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explorations*, 2005.

Hassan Malik. Improving hierarchical SVMs by hierarchy flattening and lazy classification. In *1st Pascal Workshop on Large Scale Hierarchical Classification*, 2009.

David A McAllester. Some pac-bayesian theorems. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 230–234. ACM, 1998.

Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 359–367, 1998.

Ron Meir and Tong Zhang. Generalization error bounds for bayesian mixture algorithms. *The Journal of Machine Learning Research*, 4:839–860, 2003.

Harikrishna Narasimhan, Harish G. Ramaswamy, Aadirupa Saha, and Shivani Agarwal. Consistent multiclass algorithms for complex performance measures. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, pages 2398–2407, 2015.

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems*, pages 841–848, 2001.

Peter Norvig. *Paradigms of artificial intelligence programming: case studies in Common LISP*. Morgan Kaufmann, 1992.

Ioannis Partalas, Aris Kosmopoulos, Nicolas Baskiotis, Thierry Artières, George Paliouras, Éric Gaussier, Ion Androutsopoulos, Massih-Reza Amini, and Patrick Gallinari. LSHTC: A benchmark for large-scale text classification. *CoRR*, abs/1503.08581, 2015.

Florent Perronnin, Zeynep Akata, Zaïd Harchaoui, and Cordelia Schmid. Towards good practice in large-scale learning for image classification. In *Computer Vision and Pattern Recognition*, pages 3482–3489, 2012.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2014.

Mark Schervish. *Theory of Statistics*. Springer Series in Statistics. Springer New York Inc., 1995.

Matthias Seeger. Pac-bayesian generalisation error bounds for gaussian process classification. *The Journal of Machine Learning Research*, 3:233–269, 2003.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521813972.

Min Sun, Wan Huang, and Silvio Savarese. Find the best path: An efficient and accurate classifier for image hierarchies. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 265–272. IEEE, 2013.

Xiaolin Wang and Bao-Liang Lu. Flatten hierarchies for large-scale hierarchical text categorization. In 5^{th} *International Conference on Digital Information Management*, pages 139–144, 2010.

Kilian Q Weinberger and Olivier Chapelle. Large margin taxonomy embedding for document categorization. In *Advances in Neural Information Processing Systems 21*, pages 1737–1744, 2008.

Gui-Rong Xue, Dikan Xing, Qiang Yang, and Yong Yu. Deep classification in large-scale text hierarchies. In *Proceedings of the* 31^{st} *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 619–626, 2008.

Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of the* 22^{nd} *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49. ACM, 1999.

J. Zhang, L. Tang, and H. Liu. Automatically adjusting content taxonomies for hierarchical classification. In *Proceedings of the* 4^{th} *Workshop on Text Mining*, 2006.

Journal of Machine Learning Research 17 (2016) 1-61 Submitted 6/14; Revised 7/15; Published 6/16

How to Center Deep Boltzmann Machines

Jan Melchior Jan.Melchior@ini.rub.de
Asja Fischer Asja.Fischer@ini.rub.de
Laurenz Wiskott Laurenz.Wiskott@ini.rub.de
Institut für Neuroinformatik
Ruhr-Universität Bochum
Bochum, D-44801, Germany

Editor: Yoshua Bengio

Abstract

This work analyzes centered Restricted Boltzmann Machines (RBMs) and centered Deep Boltzmann Machines (DBMs), where centering is done by subtracting offset values from visible and hidden variables. We show analytically that (i) centered and normal Boltzmann Machines (BMs) and thus RBMs and DBMs are different parameterizations of the same model class, such that any normal BM/RBM/DBM can be transformed to an equivalent centered BM/RBM/DBM and *vice versa*, and that this equivalence generalizes to artificial neural networks in general, (ii) the expected performance of centered binary BMs/RBMs/DBMs is invariant under simultaneous flip of data and offsets, for any offset value in the range of zero to one, (iii) centering can be reformulated as a different update rule for normal BMs/RBMs/DBMs, and (iv) using the enhanced gradient is equivalent to setting the offset values to the average over model and data mean. Furthermore, we present numerical simulations suggesting that (i) optimal generative performance is achieved by subtracting mean values from visible as well as hidden variables, (ii) centered binary RBMs/DBMs reach significantly higher log-likelihood values than normal binary RBMs/DBMs, (iii) centering variants whose offsets depend on the model mean, like the enhanced gradient, suffer from severe divergence problems, (iv) learning is stabilized if an exponentially moving average over the batch means is used for the offset values instead of the current batch mean, which also prevents the enhanced gradient from severe divergence, (v) on a similar level of log-likelihood values centered binary RBMs/DBMs have smaller weights and bigger bias parameters than normal binary RBMs/DBMs, (vi) centering leads to an update direction that is closer to the natural gradient, which is extremely efficient for training as we show for small binary RBMs, (vii) centering eliminates the need for greedy layer-wise pre-training of DBMs, which often even deteriorates the results independently of whether centering is used or not, and (ix) centering is also beneficial for auto encoders.

Keywords: centering, restricted Boltzmann machine, deep Boltzmann machine, generative model, artificial neural network, auto encoder, enhanced gradient, natural gradient, stochastic maximum likelihood, contrastive divergence, parallel tempering

1. Introduction

In the last decade Restricted Boltzmann Machines (RBMs) got into the focus of attention because they can be considered as building blocks of deep neural networks (Hinton et al., 2006; Bengio, 2009). RBM training methods are usually based on gradient ascent on the Log-Likelihood (LL) of the model parameters given the training data. Since the gradient is intractable, it is often approximated using Gibbs sampling with only a few steps (Hinton et al., 2006; Tieleman, 2008; Tieleman and Hinton, 2009). Two major problems have been reported when training RBMs.

Firstly, the bias of the gradient approximation introduced by using only a few steps of Gibbs sampling may lead to a divergence of the LL during training (Fischer and Igel, 2010; Schulz et al., 2010; Fischer and Igel, 2011). To overcome the divergence problem Desjardins et al. (2010) and Cho et al. (2010) have proposed to use parallel tempering (Swendsen and Wang, 1986), which is an advanced sampling method that leads to a faster mixing Markov chain and thus to a better approximation of the LL gradient.

Secondly, the learning process is not invariant to the data representation. For example training an RBM on the *MNIST* data set leads to a better model than training it on *1-MNIST* (the data set generated by flipping each bit in *MNIST*). This is due to missing invariance properties of the gradient with respect to these flip transformations and not due to the model's capacity, since an RBM trained on *MNIST* can be transformed in such a way that it models *1-MNIST* with the same LL. Recently, two approaches have been introduced that address this invariance problem. The enhanced gradient (Cho et al., 2011, 2013b) has been designed as an invariant alternative to the true LL gradient of binary RBMs and has been derived by calculating a weighted average over the gradients one gets by applying any possible bit flip combination on the data set. Empirical results suggest that the enhanced gradient leads to more distinct features and thus to better classification results based on the learned hidden representation of the data. Furthermore, in combination with an adaptive learning rate the enhanced gradient leads to more stable training in the sense that good LL values are reached independently of the initial learning rate. Tang and Sutskever (2011) on the other hand have shown empirically that subtracting the data mean from the visible variables leads to a model that can reach similar LL values on the *MNIST* and the *1-MNIST* data set and comparable results to those of the enhanced gradient.[1] Removing the mean from all variables is known as the 'centering trick', which was originally proposed for feed forward neural networks (LeCun et al., 1998; Schraudolph, 1998). It has recently also been applied to the visible and hidden variables of Deep Boltzmann Machines (DBM) (Montavon and Müller, 2012) where it has been shown to lead to a better conditioned optimization problem. Furthermore, the learned features have shown better discriminative properties and centering has improved the generative properties of locally connected DBMs. A related approach applicable to multi-layer perceptrons where the activation functions of the neurons are transformed to have zero mean and zero slope on average has been proposed by Raiko et al. (2012). The authors could show that the gradient under this transformation gets closer to the natural gradient, which is desirable since the natural gradient follows the direction of steepest ascent in the manifold of probability distributions. Furthermore, the natural gradient is independent of the concrete parameterization of the distributions and is thus clearly the update direction of choice (Amari, 1998). Since the exact natural gradient is intractable already for rather small RBMs, Schwehn (2010) and Ollivier et al. (2011) have trained binary RBMs and Desjardins et al. (2013) binary DBMs using approximations of the natural gradient obtained by Markov chain Monte Carlo methods. However, due to

1. Note that changing the model such that the mean of the visible variables is removed is not equivalent to just removing the mean of the data.

the computational overhead the practical relevance of the natural gradient for RBM/DBM training remains questionable. Another approach related to the centering trick is batch normalization, which has recently been proposed by Ioffe and Szegedy (2015) and aims at removing first and second order statistics of the pre synaptic activation in a feed forward network.

In this article[2] we give a unified view on centering, revealing that the methods proposed by Cho et al. (2011), Tang and Sutskever (2011), and Montavon and Müller (2012) can all be considered as different ways of applying the centering trick to RBMs and DBMs. Furthermore, we analyze the properties and performance of different centering variants. In Section 2, we begin with a brief overview over binary RBMs, the standard learning algorithms, the natural gradient of the LL of RBMs, and the basic ideas used to construct the enhanced gradient. In Section 3, we discuss the theoretical properties of centered RBMs, show that centering can be reformulated as a different update rule for normal RBMs, that the enhanced gradient is a particular form of centering, and finally that centering and its properties naturally extend to DBMs and BMs. Furthermore, in Section 4, we show that centering is an alternative parameterization for arbitrary Artificial Neural Networks (ANNs) in general and we discuss how the parameters of centered and normal ANNs should be initialized. Our experimental setups are described in Section 5 before we empirically analyze the performance of centered RBMs with different initializations, offset parameters, sampling methods, and learning rates in Section 6. This empirical analysis includes a comparison of the centered gradient with the natural gradient and extensive experiments on 10 real world data sets. In addition we have also performed experiments with DBMs and Auto encoders (AEs) on the 10 real work data sets. Finally our work is concluded in Section 7.

2. Restricted Boltzmann Machines

An RBM (Smolensky, 1986) is a bipartite undirected graphical model with a set of N visible and M hidden variables taking values $\mathbf{x} = (x_1, ..., x_N)$ and $\mathbf{h} = (h_1, ..., h_M)$, respectively. Since an RBM is a Markov random field, its joint probability distribution is given by a Gibbs distribution

$$p(\mathbf{x}, \mathbf{h}) = \frac{1}{Z} \mathrm{e}^{-E(\mathbf{x},\mathbf{h})} ,$$

with partition function Z and energy $E(\mathbf{x}, \mathbf{h})$. For binary RBMs, $\mathbf{x} \in \{0,1\}^N$, $\mathbf{h} \in \{0,1\}^M$, the energy, which defines the bipartite structure, is given by

$$E(\mathbf{x}, \mathbf{h}) = -\mathbf{x}^T \mathbf{b} - \mathbf{c}^T \mathbf{h} - \mathbf{x}^T \mathbf{W} \mathbf{h} ,$$

where the weight matrix $\mathbf{W}$, the visible bias vector $\mathbf{b}$, and the hidden bias vector $\mathbf{c}$ are the parameters of the model, jointly denoted by $\boldsymbol{\theta}$. The partition function, which sums over all possible visible and hidden states, is given by

$$Z = \sum_{\mathbf{x}} \sum_{\mathbf{h}} \mathrm{e}^{-E(\mathbf{x},\mathbf{h})} .$$

2. Previous versions of this work have been published as an eprint (Melchior et al., 2013) and as part of the PhD thesis by Fischer (2014).

RBM training is usually based on gradient ascent using approximations of the LL gradient

$$\nabla \boldsymbol{\theta} = \frac{\partial \langle \log(p(\mathbf{x}|\boldsymbol{\theta})) \rangle_d}{\partial \boldsymbol{\theta}} = -\left\langle \frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \boldsymbol{\theta}} \right\rangle_d + \left\langle \frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \boldsymbol{\theta}} \right\rangle_m ,$$

where $\langle \cdot \rangle_m$ is the expectation under $p(\mathbf{x}, \mathbf{h})$ and $\langle \cdot \rangle_d$ is the expectation under $p(\mathbf{h}|\mathbf{x}) p_e(\mathbf{x})$ with empirical distribution p_e. We use the notation $\nabla \boldsymbol{\theta}$ for the derivative of the LL with respect to $\boldsymbol{\theta}$ in order to be consistent with the notation used by Cho et al. (2011). For binary RBMs the gradient becomes

$$\begin{aligned} \nabla \mathbf{W} &= \langle \mathbf{x}\mathbf{h}^T \rangle_d - \langle \mathbf{x}\mathbf{h}^T \rangle_m , \\ \nabla \mathbf{b} &= \langle \mathbf{x} \rangle_d - \langle \mathbf{x} \rangle_m , \\ \nabla \mathbf{c} &= \langle \mathbf{h} \rangle_d - \langle \mathbf{h} \rangle_m . \end{aligned}$$

Common RBM training methods approximate $\langle \cdot \rangle_m$ by samples gained by different Markov chain Monte Carlo methods. Sampling k (usually $k = 1$) steps from a Gibbs chain initialized with a data sample yields Contrastive Divergence (CD) (Hinton et al., 2006). In stochastic maximum likelihood (Younes, 1991), in the context of RBMs also known as Persistent Contrastive Divergence (PCD) (Tieleman, 2008), the chain is not reinitialized with a data sample after parameter updates. This has been reported to lead to better gradient approximations if the learning rate is chosen sufficiently small. Fast Persistent Contrastive Divergence (FPCD) (Tieleman and Hinton, 2009) tries to further speed up learning by introducing an additional set of parameters, that is only used for Gibbs sampling during learning. The advanced sampling method Parallel Tempering (PT) (Swendsen and Wang, 1986) introduces additional 'tempered' Gibbs chains corresponding to smoothed versions of $p(\mathbf{x}, \mathbf{h})$. The energy of these distributions is multiplied by $\frac{1}{T}$, where T is referred to as temperature. The higher the temperature of a chain, the 'smoother' the corresponding distribution and the faster the chain mixes. Samples may swap between chains with a probability given by the Metropolis Hastings ratio, which leads to better mixing of the original chain with temperature $T = 1$ (a first theoretical analysis of the mixing rate of PT for sampling in RBMs has been given by Fischer and Igel, 2015). We use PT_c to denote the RBM training algorithm that uses Parallel Tempering with c tempered chains as a sampling method. Usually only one step of Gibbs sampling is performed in each tempered chain before allowing samples to swap, and a deterministic even odd algorithm (Lingenheil et al., 2009) is used as a swapping schedule. PT_c increases the mixing rate and has been reported to achieve better gradient approximations than CD and (F)PCD (Desjardins et al., 2010; Cho et al., 2010) with the drawback of having a higher computational cost.

See the introductory paper of Fischer and Igel (2014) for a recent review of RBMs and their training algorithms.

2.1 Enhanced Gradient

Cho et al. (2011) have proposed a different way to update parameters during training of binary RBMs, which is invariant to the data representation.

When transforming the state $(\mathbf{x}, \mathbf{h})$ of a binary RBM by flipping some of its variables (that is $\tilde{x}_i = 1 - x_i$, and $\tilde{h_j} = 1 - h_j$ for some i, j), yielding a new state $(\tilde{\mathbf{x}}, \tilde{\mathbf{h}})$, one can

transform the parameters $\boldsymbol{\theta}$ of the RBM to $\tilde{\boldsymbol{\theta}}$ such that $E(\mathbf{x}, \mathbf{h}|\boldsymbol{\theta}) = E(\tilde{\mathbf{x}}, \tilde{\mathbf{h}}|\tilde{\boldsymbol{\theta}}) + const$ and thus $p(\mathbf{x}, \mathbf{h}|\boldsymbol{\theta}) = p(\tilde{\mathbf{x}}, \tilde{\mathbf{h}}|\tilde{\boldsymbol{\theta}})$. However, if we update the parameters of the transformed model based on the corresponding LL gradient to $\tilde{\boldsymbol{\theta}}' = \tilde{\boldsymbol{\theta}} + \eta \nabla \tilde{\boldsymbol{\theta}}$ and apply the inverse parameter transformation to $\tilde{\boldsymbol{\theta}}'$, the result differs from $\boldsymbol{\theta}' = \boldsymbol{\theta} + \eta \nabla \boldsymbol{\theta}$. The described procedure of transforming, updating, and transforming back can be regarded as a different way to update $\boldsymbol{\theta}$.

Following this line of thought there exist 2^{N+M} different parameter updates corresponding to the 2^{N+M} possible binary flips of $(\mathbf{x}, \mathbf{h})$. Cho et al. (2011) have proposed the enhanced gradient as a weighted sum of these 2^{N+M} parameter updates, which for their choice of weighting is given by

$$\nabla_e \mathbf{W} = \langle (\mathbf{x} - \langle \mathbf{x} \rangle_d)(\mathbf{h} - \langle \mathbf{h} \rangle_d)^T \rangle_d - \langle (\mathbf{x} - \langle \mathbf{x} \rangle_m)(\mathbf{h} - \langle \mathbf{h} \rangle_m)^T \rangle_m \ , \tag{1}$$

$$\nabla_e \mathbf{b} = \langle \mathbf{x} \rangle_d - \langle \mathbf{x} \rangle_m - \nabla_e \mathbf{W} \frac{1}{2} \left(\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m \right) \ , \tag{2}$$

$$\nabla_e \mathbf{c} = \langle \mathbf{h} \rangle_d - \langle \mathbf{h} \rangle_m - \nabla_e \mathbf{W}^T \frac{1}{2} \left(\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m \right) \ . \tag{3}$$

It has been shown that the enhanced gradient is invariant to arbitrary bit flips of the variables and therefore invariant under the data representation, which has been demonstrated on the *MNIST* and *1-MNIST* data set. Furthermore, the authors have reported more stable training under various settings in terms of the LL estimate and classification accuracy.

2.2 Natural Gradient

Following the direction of steepest ascent in the Euclidean parameter space (as given by the standard gradient) does not necessarily correspond to the direction of steepest ascent in the manifold of probability distributions $\{p(\mathbf{x}|\boldsymbol{\theta}), \boldsymbol{\theta} \in \Theta\}$, which we are actually interested in. To account for the local geometry of the manifold, the Euclidean metric should be replaced by the Fisher information metric defined by $||\boldsymbol{\theta}||_{\mathcal{I}(\boldsymbol{\theta})} = \sqrt{\sum \theta_k \mathcal{I}_{kl}(\boldsymbol{\theta}) \theta_l}$, where $\mathcal{I}(\boldsymbol{\theta})$ is the Fisher information matrix (Amari, 1998). The kl-th entry of the Fisher information matrix for a parameterized distribution $p(\mathbf{x}|\boldsymbol{\theta})$ is given by

$$\boldsymbol{\mathcal{I}}_{kl}(\theta) = \left\langle \left(\frac{\partial \log \left(p(\mathbf{x}|\boldsymbol{\theta}) \right)}{\partial \theta_k} \right) \left(\frac{\partial \log \left(p(\mathbf{x}|\boldsymbol{\theta}) \right)}{\partial \theta_l} \right) \right\rangle_m \ ,$$

where $\langle \cdot \rangle_m$ denotes the expectation under $p(\mathbf{x}|\boldsymbol{\theta})$. The gradient associated with the Fisher metric is called the natural gradient and is given by

$$\nabla_n \boldsymbol{\theta} = \boldsymbol{\mathcal{I}}(\theta)^{-1} \nabla \boldsymbol{\theta} \ .$$

The inverse Fisher information matrix corrects the direction and length of the standard gradient to achieve the largest change of the objective function (here the LL) for an infinitesimal small step $\delta\boldsymbol{\theta}$ from $p(\mathbf{x}|\boldsymbol{\theta})$ to $p(\mathbf{x}|\boldsymbol{\theta}+\boldsymbol{\delta\theta})$ in terms of the Kullback-Leibler divergence (Amari, 1998). This makes the natural gradient independent of the parameterization leading to an invariance to arbitrary coordinate transformations such as flipping variables, which makes it clearly the update direction of choice.

For binary RBMs the entries of the Fisher information matrix (Amari et al., 1992; Desjardins et al., 2013; Ollivier et al., 2011) are given by

$$\begin{aligned}
\boldsymbol{\mathcal{I}}_{w_{ij},w_{uv}}(\theta) = \boldsymbol{\mathcal{I}}_{,w_{uv},w_{ij}}(\theta) &= \langle x_i h_j x_u h_v \rangle_m - \langle x_u h_v \rangle_m \langle x_u h_v \rangle_m \\
&= Cov_m(x_i h_j, x_u h_v) \ , \\
\boldsymbol{\mathcal{I}}_{w_{ij},b_u}(\theta) = \boldsymbol{\mathcal{I}}_{b_u,w_{ij}}(\theta) &= Cov_m(x_i h_j, x_u) \ , \\
\boldsymbol{\mathcal{I}}_{w_{ij},c_v}(\theta) = \boldsymbol{\mathcal{I}}_{c_v,w_{ij}}(\theta) &= Cov_m(x_i h_j, h_v) \ , \\
\boldsymbol{\mathcal{I}}_{b_i,b_u}(\theta) = \boldsymbol{\mathcal{I}}_{b_u,b_i}(\theta) &= Cov_m(x_i, x_u) \ , \\
\boldsymbol{\mathcal{I}}_{c_j,c_v}(\theta) = \boldsymbol{\mathcal{I}}_{c_v,c_j}(\theta) &= Cov_m(h_j, h_v) \ ,
\end{aligned}$$

where $Cov_m(\cdot, \cdot)$ denotes the covariance under the model distribution. Since these expressions involve expectations under the model distribution they are not tractable in general, but can be approximated using MCMC methods (Ollivier et al., 2011; Desjardins et al., 2013). Furthermore, a diagonal approximation of the Fisher information matrix could be used. However, the approximation of the natural gradient for RBMs and DBMs is still computationally very expensive so that the practical usability remains questionable (Schwehn, 2010; Ollivier et al., 2011; Desjardins et al., 2013).

3. Centered Restricted Boltzmann Machines

Inspired by the centering trick proposed by LeCun et al. (1998), Tang and Sutskever (2011) have addressed the problem that RBMs perform differently on the *MNIST* and *1-MNIST* data set by changing the energy of the RBM in a way that the mean of the input data is removed. Montavon and Müller (2012) have extended the idea of centering to the visible and hidden variables of DBMs and have shown that centering improves the conditioning of the underlying optimization problem, leading to models with better discriminative properties for DBMs in general and better generative properties in the case of locally connected DBMs.

Following their line of thought, the energy for a centered binary RBM where the visible and hidden variables are shifted by the offset parameters $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_N)$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M)$, respectively, can be formulated as

$$E(\mathbf{x}, \mathbf{h}) = -(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{b} - \mathbf{c}^T (\mathbf{h} - \boldsymbol{\lambda}) - (\mathbf{x} - \boldsymbol{\mu})^T \mathbf{W} (\mathbf{h} - \boldsymbol{\lambda}) \ . \tag{4}$$

By setting both offsets to zero one retains the normal binary RBM. Setting $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_d$ and $\boldsymbol{\lambda} = \mathbf{0}$ leads to the model introduced by Tang and Sutskever (2011), and by setting $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_d$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_d$ we get a shallow variant of the centered DBM analyzed by Montavon and Müller (2012).

The conditional probabilities for a variable taking the value one are given by

$$p(x_i = 1|\mathbf{h}) = \sigma(\mathbf{w_{i*}} (\mathbf{h} - \boldsymbol{\lambda}) + b_i) \ , \tag{5}$$

$$p(h_j = 1|\mathbf{x}) = \sigma((\mathbf{x} - \boldsymbol{\mu})^T \mathbf{w_{*j}} + c_j) \ , \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathbf{w_{i*}}$ represents the ith row, and $\mathbf{w_{*j}}$ the jth column of the weight matrix $\mathbf{W}$. The LL gradient now takes the form

$$\nabla \mathbf{W} = \langle (\mathbf{x} - \boldsymbol{\mu})(\mathbf{h} - \boldsymbol{\lambda})^T \rangle_d - \langle (\mathbf{x} - \boldsymbol{\mu})(\mathbf{h} - \boldsymbol{\lambda})^T \rangle_m \ , \tag{7}$$

$$\nabla \mathbf{b} = \langle \mathbf{x} - \boldsymbol{\mu} \rangle_d - \langle \mathbf{x} - \boldsymbol{\mu} \rangle_m = \langle \mathbf{x} \rangle_d - \langle \mathbf{x} \rangle_m \ , \tag{8}$$

$$\nabla \mathbf{c} = \langle \mathbf{h} - \boldsymbol{\lambda} \rangle_d - \langle \mathbf{h} - \boldsymbol{\lambda} \rangle_m = \langle \mathbf{h} \rangle_d - \langle \mathbf{h} \rangle_m \ . \tag{9}$$

$\nabla \mathbf{b}$ and $\nabla \mathbf{c}$ are independent of the choice of $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ and thus centering only affects $\nabla \mathbf{W}$. It can be shown (see Appendix A) that the gradient of a centered BM is invariant to flip transformations if a flip of x_i to $1 - x_i$ implies a change of μ_i to $1 - \mu_i$, and in the case of RBMs a flip h_j to $1-h_j$ implies a change of λ_j to $1-\lambda_j$. This holds for $\mu_i = 0.5$, $\lambda_j = 0.5$ but remarkably also for the expectation values over x_i and h_j under any distribution. Moreover, if the offsets are set to the expectation values, centered RBMs get also invariant to shifts

Algorithm 1: Training centered RBMs

Initialize $\mathbf{W}$; /* *i.e.*, $\mathbf{W} \leftarrow \mathcal{N}(0, 0.01)^{N \times M}$ */
Initialize $\boldsymbol{\mu}, \boldsymbol{\lambda}$; /* *i.e.*, $\boldsymbol{\mu} \leftarrow \langle \mathtt{data} \rangle, \boldsymbol{\lambda} \leftarrow \mathbf{0.5}$ */
Initialize $\mathbf{b}, \mathbf{c}$; /* *i.e.*, $\mathbf{b} \leftarrow \sigma^{-1}(\boldsymbol{\mu}), \mathbf{c} \leftarrow \sigma^{-1}(\boldsymbol{\lambda})$ */
Initialize $\eta, \nu_\mu, \nu_\lambda$; /* *i. e.*, $\eta, \nu_\mu, \nu_\lambda \in \{0.001, ..., 0.1\}$ */
repeat
  foreach batch **in** data **do**
    foreach *sample* $\mathbf{x}_d$ **in** batch **do**
      Calculate $\mathbf{h}_d = p(\mathbf{h} = \mathbf{1}|\mathbf{x}_d)$; /* ▷ Eq. (6) */
      Sample $\mathbf{x}_m$ *from RBM* ; /* *i.e.*, PCD-1, ▷ Eqs. (5), (6) */
      Calculate $\mathbf{h}_m = p(\mathbf{h} = \mathbf{1}|\mathbf{x}_m)$; /* ▷ Eq. (6) */
      Store $\mathbf{x}_m, \mathbf{h}_d, \mathbf{h}_m$
    Estimate $\boldsymbol{\mu}_{batch}$; /* *i.e.* $\boldsymbol{\mu}_{batch} \leftarrow \langle \mathbf{x}_d \rangle$ */
    Estimate $\boldsymbol{\lambda}_{batch}$; /* *i.e.* $\boldsymbol{\lambda}_{batch} \leftarrow \langle \mathbf{h}_d \rangle$ */
    /* Transform parameters with respect to the new offsets */
    $\mathbf{b} \leftarrow \mathbf{b} + \nu_\lambda \mathbf{W} (\boldsymbol{\lambda}_{batch} - \boldsymbol{\lambda})$; /* ▷ Eq. (11), $\tilde{\boldsymbol{\lambda}} = (1 - \nu_\lambda)\boldsymbol{\lambda} + \nu_\lambda \boldsymbol{\lambda}_{batch}$ */
    $\mathbf{c} \leftarrow \mathbf{c} + \nu_\mu \mathbf{W}^T (\boldsymbol{\mu}_{batch} - \boldsymbol{\mu})$; /* ▷ Eq. (12), $\tilde{\boldsymbol{\mu}} = (1 - \nu_\mu)\boldsymbol{\mu} + \nu_\mu \boldsymbol{\mu}_{batch}$ */
    /* Update offsets using exp. moving average with sliding factors ν_μ and ν_λ */
    $\boldsymbol{\mu} \leftarrow (1 - \nu_\mu)\boldsymbol{\mu} + \nu_\mu \boldsymbol{\mu}_{batch}$
    $\boldsymbol{\lambda} \leftarrow (1 - \nu_\lambda)\boldsymbol{\lambda} + \nu_\lambda \boldsymbol{\lambda}_{batch}$
    /* Update parameters using the gradient with learning rate η */
    $\nabla \mathbf{W} \leftarrow \langle (\mathbf{x}_d - \boldsymbol{\mu})(\mathbf{h}_d - \boldsymbol{\lambda})^T \rangle - \langle (\mathbf{x}_m - \boldsymbol{\mu})(\mathbf{h}_m - \boldsymbol{\lambda})^T \rangle$; /* ▷ Eq. (7) */
    $\nabla \mathbf{b} \leftarrow \langle \mathbf{x}_d \rangle - \langle \mathbf{x}_m \rangle$; /* ▷ Eq. (8) */
    $\nabla \mathbf{c} \leftarrow \langle \mathbf{h}_d \rangle - \langle \mathbf{h}_m \rangle$; /* ▷ Eq. (9) */
    $\mathbf{W} \leftarrow \mathbf{W} + \eta \nabla \mathbf{W}$
    $\mathbf{b} \leftarrow \mathbf{b} + \eta \nabla \mathbf{b}$
    $\mathbf{c} \leftarrow \mathbf{c} + \eta \nabla \mathbf{c}$
until *stopping criteria is met*;
/* Transform network to a normal binary RBM if desired */
$\mathbf{b} \leftarrow \mathbf{b} - \mathbf{W}\boldsymbol{\lambda}$; /* ▷ Eq. (11) */
$\mathbf{c} \leftarrow \mathbf{c} - \mathbf{W}^T \boldsymbol{\mu}$; /* ▷ Eq. (12) */
$\boldsymbol{\mu} \leftarrow \mathbf{0}$
$\boldsymbol{\lambda} \leftarrow \mathbf{0}$

of variables (see Section 4). Note that these properties of centered RBMs naturally extend to centered BMs and DBMs (see Section 3.2, Appendix A and B).

If we set $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ to the expectation values of the variables, these values may depend on the RBM parameters (think for example about $\langle \mathbf{h} \rangle_d$) and thus they might change during training. Consequently, a learning algorithm for centered RBMs needs to update the offset values to match the expectations under the distribution that has changed through a parameter update. When updating the offsets one needs to transform the RBM parameters such that the modeled probability distribution stays the same. An RBM with offsets $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ can be transformed to an RBM with offsets $\tilde{\boldsymbol{\mu}}$ and $\tilde{\boldsymbol{\lambda}}$ by

$$\tilde{\mathbf{W}} = \mathbf{W} \,, \tag{10}$$

$$\tilde{\mathbf{b}} = \mathbf{b} + \mathbf{W}\left(\tilde{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\right) \,, \tag{11}$$

$$\tilde{\mathbf{c}} = \mathbf{c} + \mathbf{W}^T (\tilde{\boldsymbol{\mu}} - \boldsymbol{\mu}) \,, \tag{12}$$

such that $E(\mathbf{x}, \mathbf{h}|\boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\lambda}) = E(\mathbf{x}, \mathbf{h}|\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\lambda}}) + const$, is guaranteed (see Appendix B for a derivation for a more general BM). Obviously, this can also be used to transform a centered RBM to a normal RBM and *vice versa*, highlighting that centered and normal RBMs are just different parameterizations of the same model class.

If the intractable model mean is used for an offset, it has to be approximated by samples. Furthermore, when $\boldsymbol{\lambda}$ is chosen to be $\langle \mathbf{h} \rangle_d$ or $\langle \mathbf{h} \rangle_m$ or when $\boldsymbol{\mu}$ is chosen to be $\langle \mathbf{x} \rangle_m$ one could approximate the mean values either using the sampled states or the corresponding conditional probabilities. But due to the Rao-Blackwell theorem an estimation based on the probabilities has lower variance and therefore is the approximation of choice.[3]

Algorithm 1 shows pseudo code for training a centered binary RBM, where we use $\langle \cdot \rangle$ to denote the average over samples from the current batch. Thus, for example, we write $\langle \mathbf{x}_d \rangle$ for the average value of data samples $\mathbf{x}_d$ in the current batch, which is used as an approximation for the expectation of $\mathbf{x}$ under the data distribution, that is $\langle \mathbf{x} \rangle_d$. Similarly, $\langle \mathbf{h}_d \rangle = \langle p\left(\mathbf{h} = \mathbf{1}|\mathbf{x}_d\right) \rangle$ approximates $\langle \mathbf{h} \rangle_d$, where $p\left(\mathbf{x} = \mathbf{1}|\mathbf{h}\right)$ denotes the vector containing the elements $p\left(x_i = 1|\mathbf{h}\right)$. Note that in Algorithm 1 the update of the offsets is performed before the gradient is calculated, such that gradient and reparameterization both use the current samples. This is in contrast to the algorithm for centered DBMs proposed by Montavon and Müller (2012), where the update of the offsets and the reparameterization follows after the gradient update. Thus, while the gradient still uses the current samples the reparameterization is based on samples gained from the model of the previous iteration. However, the proposed DBM algorithm smooths the offset estimations by an exponentially moving average over the sample means from many iterations, so that the choice of the sample set used for the offset estimation should be less relevant.

In Algorithm 1 an exponentially moving average for the approximation of $\boldsymbol{\mu}$ is obtained if the corresponding sliding factor ν_μ is set to $0 < \nu_\mu < 1$ or prevented if $\nu_\mu = 1$ and equivalently for $\boldsymbol{\lambda}$ and ν_λ. The effects of using an exponentially moving average are empirically analyzed in Section 6.5.

3. This can be proven analogously to the proof of proposition 1 in the work of Swersky et al. (2010).

3.1 Centered Gradient

We now use the centering trick to derive a centered parameter update, which can replace the gradient during the training of normal RBMs. Similar to the derivation of the enhanced gradient we can transform a normal RBM to a centered RBM, perform a gradient update, and transform the RBM back (see Appendix B for a derivation for the more general BM). This yields the following parameter updates, which we refer to as centered gradient

$$\nabla_c \mathbf{W} = \langle (\mathbf{x}-\boldsymbol{\mu})(\mathbf{h}-\boldsymbol{\lambda})^T \rangle_d - \langle (\mathbf{x}-\boldsymbol{\mu})(\mathbf{h}-\boldsymbol{\lambda})^T \rangle_m \ , \tag{13}$$

$$\nabla_c \mathbf{b} = \langle \mathbf{x} \rangle_d - \langle \mathbf{x} \rangle_m - \nabla_c \mathbf{W} \boldsymbol{\lambda} \ , \tag{14}$$

$$\nabla_c \mathbf{c} = \langle \mathbf{h} \rangle_d - \langle \mathbf{h} \rangle_m - \nabla_c \mathbf{W}^T \boldsymbol{\mu} \ . \tag{15}$$

Note that by setting $\boldsymbol{\mu} = \frac{1}{2}\left(\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m\right)$ and $\boldsymbol{\lambda} = \frac{1}{2}\left(\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m\right)$ the centered gradient becomes equal to the enhanced gradient (see Appendix C). Thus, it becomes clear that the enhanced gradient is a special case of centering. This can also be concluded from the derivation of the enhanced gradient for Gaussian visible variables in (Cho et al., 2013a).

Algorithm 2: Training RBMs using the centered gradient

Initialize $\mathbf{W}$; /* *i.e.* $\mathbf{W} \leftarrow \mathcal{N}(0, 0.01)^{N \times M}$ */
Initialize $\boldsymbol{\mu}, \boldsymbol{\lambda}$; /* *i.e.* $\boldsymbol{\mu} \leftarrow \langle \texttt{data} \rangle, \boldsymbol{\lambda} \leftarrow \mathbf{0.5}$ */
Initialize $\mathbf{b}, \mathbf{c}$; /* *i.e.* $\mathbf{b} \leftarrow \sigma^{-1}(\boldsymbol{\mu}), \mathbf{c} \leftarrow \sigma^{-1}(\boldsymbol{\lambda})$ */
Initialize $\eta, \nu_\mu, \nu_\lambda$; /* *i.e.* $\eta, \nu_\mu, \nu_\lambda \in \{0.001, ..., 0.1\}$ */
repeat
  foreach batch **in** data **do**
    foreach sample $\mathbf{x}_d$ **in** batch **do**
      Calculate $\mathbf{h}_d = p(\mathbf{h} = \mathbf{1} | \mathbf{x}_d)$; /* ▷ Eq. (6) */
      Sample $\mathbf{x}_m$ *from RBM* ; /* *i.e.*, PCD-1, ▷ Eqs. (5), (6) */
      Calculate $\mathbf{h}_m = p(\mathbf{h} = \mathbf{1} | \mathbf{x}_m)$; /* ▷ Eq. (6) */
      Store $\mathbf{x}_m, \mathbf{h}_d, \mathbf{h}_m$
    Estimate $\boldsymbol{\mu}_{batch}$; /* *i.e.* $\boldsymbol{\mu}_{batch} \leftarrow \langle \mathbf{x}_d \rangle$ */
    Estimate $\boldsymbol{\lambda}_{batch}$; /* *i.e.* $\boldsymbol{\lambda}_{batch} \leftarrow \langle \mathbf{h}_d \rangle$ */
    /* Update offsets using exp. moving averages with sliding factors ν_μ and ν_λ */
    $\boldsymbol{\mu} \leftarrow (1 - \nu_\mu)\boldsymbol{\mu} + \nu_\mu \boldsymbol{\mu}_{batch}$
    $\boldsymbol{\lambda} \leftarrow (1 - \nu_\lambda)\boldsymbol{\lambda} + \nu_\lambda \boldsymbol{\lambda}_{batch}$
    /* Update parameters using the centered gradient with learning rate η */
    $\nabla_c \mathbf{W} \leftarrow \langle (\mathbf{x}_d - \boldsymbol{\mu})(\mathbf{h}_d - \boldsymbol{\lambda})^T \rangle - \langle (\mathbf{x}_m - \boldsymbol{\mu})(\mathbf{h}_m - \boldsymbol{\lambda})^T \rangle$; /* ▷ Eq. (13) */
    $\nabla_c \mathbf{b} \leftarrow \langle \mathbf{x}_d \rangle - \langle \mathbf{x}_m \rangle - \nabla_c \mathbf{W} \boldsymbol{\lambda}$; /* ▷ Eq. (14) */
    $\nabla_c \mathbf{c} \leftarrow \langle \mathbf{h}_d \rangle - \langle \mathbf{h}_m \rangle - \nabla_c \mathbf{W}^T \boldsymbol{\mu}$; /* ▷ Eq. (15) */
    $\mathbf{W} \leftarrow \mathbf{W} + \eta \nabla_c \mathbf{W}$
    $\mathbf{b} \leftarrow \mathbf{b} + \eta \nabla_c \mathbf{b}$
    $\mathbf{c} \leftarrow \mathbf{c} + \eta \nabla_c \mathbf{c}$
until *stopping criteria is met*;

The enhanced gradient has been designed such that the weight updates become the difference of the covariances between one visible and one hidden variable under the data and the model distribution. Interestingly, one gets the same weight update for two other choices of offset parameters, either $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_d$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_m$ or $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_m$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_d$. However, these offsets result in different update rules for the bias parameters.

Algorithm 2 shows pseudo code for training a normal binary RBM using the centered gradient, which is equivalent to training a centered binary RBM using Algorithm 1. Both algorithms can easily be extended to RBMs with other types of units, DBMs and BMs.

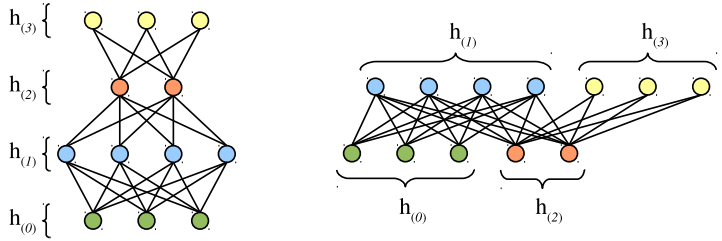
3.2 Centered Deep Boltzmann Machines

A DBM (Salakhutdinov and Hinton, 2009) is a deep undirected graphical model with several hidden layers where successive layers have a bipartite connectivity structure. Therefore, a DBM can be seen as a jointly trained stack of several RBMs and thus as a natural extension of RBMs. A centered binary DBM with L layers $\mathbf{h}_{(0)}, \cdots, \mathbf{h}_{(L)}$ (where $\mathbf{h}_{(0)}$ corresponds to the visible layer) represents a Gibbs distribution with energy

$$E\left(\mathbf{h}_{(0)}, \cdots, \mathbf{h}_{(L)}\right) = -\sum_{l=0}^{L} \left(\mathbf{h}_{(l)} - \boldsymbol{\lambda}_{(l)}\right)^T \mathbf{b}_{(l)} - \sum_{l=0}^{L-1} \left(\mathbf{h}_{(l)} - \boldsymbol{\lambda}_{(l)}\right)^T \mathbf{W}_{(l)} \left(\mathbf{h}_{(l+1)} - \boldsymbol{\lambda}_{(l+1)}\right),$$

where each layer l has a bias $\mathbf{b}_{(l)}$, an offset $\boldsymbol{\lambda}_{(l)}$ and is connected to layer $l+1$ by weight matrix $\mathbf{W}_l$.

The derivations, proofs and algorithms given in this work for RBMs/BMs automatically extend to DBMs since each DBM can be transformed to an RBM with restricted connections and partially unknown input data. This is illustrated for a DBM with four layers in Figure 1. As a consequence of this relation DBMs can essentially be trained in the same way as RBMs but also suffer from the same problems as described before. The major difference in DBM



(a) Deep network version. (b) Shallow network version.

Figure 1: Example for (a) a DBM or an ANN with four layers $h_{(0)}, \cdots, h_{(3)}$ and (b) the equivalent two layer shallow version of the same network with restricted connections and unknown input $h_{(2)}$.

training compared to RBM training is that the expectation under the data distribution in the LL gradient of DBMs gets intractable. Due to the partially missing input data the factorization trick used for the calculation of expectation under the data distribution in RBMs cannot be applied anymore. Instead the term is approximated by running a mean field estimation until convergence (Salakhutdinov and Hinton, 2009), which corresponds to approximating the gradient of the variational lower bound of the LL. Furthermore, it is common to pre-train DBMs in a greedy layer wise fashion using RBMs (Salakhutdinov and Hinton, 2009; Hinton and Salakhutdinov, 2012).

4. Centering in Artificial Neural Networks in General

Removing the mean from visible and hidden units was originally proposed for feed forward neural networks (LeCun et al., 1998; Schraudolph, 1998). When this idea was applied to the visible units of RBMs (Tang and Sutskever, 2011) the model was reparameterized such that the probability distribution defined by the normal and centered RBM stayed the same.

In this section we generalize this concept to show that centering is an alternative parameterization for arbitrary ANN architectures in general, if the network is reparameterized accordingly. Consider the centered artificial neuron model

$$o_j \quad = \quad \phi_j \left(\sum_i w_{ij} \left(a_i - \mu_i \right) + c_j \right), \tag{16}$$

where the output o_j of the jth neuron depends on its activation function ϕ_j, bias term c_j and weights w_{ij} with associated inputs a_i and their corresponding offsets μ_i. Such neurons can be used to construct arbitrary network architectures using undirected, directed and recurrent connections, which can then be optimized with respect to a chosen loss.

Two different ANNs that represent exactly the same functional input-output mapping can be considered as different parameterizations of the same model. Thus, a centered ANN is just a different parameterization of an uncentered ANN if we can show that their functional input-output mappings are the same. This can be guaranteed in general if all corresponding units in a centered and an uncentered ANN have the same mapping from inputs to outputs. If the offset μ_i is changed to $\tilde{\mu}_i = \mu_i + \nabla\mu_i$ then the output of the centered artificial neuron (16) becomes

$$\begin{aligned} \phi_j \left(\sum_i w_{ij} \left(a_i - \tilde{\mu}_i \right) + c_j \right) &= \phi_j \left(\sum_i w_{ij} \left(a_i - (\mu_i + \nabla\mu_i) \right) + c_j \right) \\ &= \phi_j \left(\sum_i w_{ij} \left(a_i - \mu_i \right) + c_j - \sum_i w_{ij} \nabla\mu_i \right), \end{aligned}$$

showing that the unit's output does not change when changing the offset μ_i to $\tilde{\mu}_i$ if the unit's bias parameter c_j is reparameterized to $\tilde{c}_j = c_j + \sum_i w_{ij} \nabla\mu_i$. This generalizes the reparameterization for RBMs given by Equations (10) – (12) to ANNs. Now, by setting μ_i or $\tilde{\mu}_i$ to zero it follows that for each normal ANN there exists a centered ANN and *vice versa* such that the output of each neuron and thus the functional mapping from input to output of the whole network stays the same. This holds independently of the chosen

activation functions, loss function and connection types including directed, undirected and recurrent connections.

Moreover, if we guarantee that a shift of a_i implies a shift of μ_i by the same value (that is, a shift of a_i to $a_i + \delta_i$ implies a shift of μ_i to $\mu_i + \delta_i$) the neuron's output o_j gets invariant to shifts of a_i. This is easy to see since δ_i cancels out in Equation (16) if the same shift is applied to both a_i and μ_i, which holds for example if we set the offsets to the mean values of the corresponding variables since $\langle a_i + \delta_i \rangle = \langle a_i \rangle + \delta_i$.

Note that, the original centering algorithm (LeCun et al., 1998; Schraudolph, 1998) did not reparameterize the network, which can cause instabilities especially if the learning rate is large.

4.1 Auto Encoders

An AE or auto-associator (Rumelhart et al., 1986b) is a type of ANN (and can thus be centered) that has originally been proposed for unsupervised dimensionality reduction. Like RBMs, AEs have also been used for unsupervised feature extraction and greedy layer-wise pre-training of deep neural networks (Bengio et al., 2007), and therefore fit well in this study for analyzing centering in ANNs.

In general, an AE consists of a deterministic encoder $encode(\mathbf{x})$, which maps the input $\mathbf{x} = (x_1, ..., x_N)$ to a hidden representation $\mathbf{h} = (h_1, ..., h_M)$ and a deterministic decoder $decode(\mathbf{h})$, which maps the hidden representation to the reconstructed input representation $\tilde{\mathbf{x}}$. The network is optimized such that the reconstructed input $\tilde{\mathbf{x}}$ gets as close as possible to the original input $\mathbf{x}$ measured by a chosen loss $\mathcal{L}(\mathbf{x}, \tilde{\mathbf{x}})$. Common choices for the loss are the mean squared error $\langle \sum_{i=1}^{N} (x_i - \tilde{x}_i)^2 \rangle_d$ for arbitrary input and the average cross entropy $\langle -\sum_{i=1}^{N} x_i \log \tilde{x}_i + (1 - x_i) \log(1 - \tilde{x}_i) \rangle_d$ for binary data. AEs are usually trained via back-propagation (Kelley, 1960; Rumelhart et al., 1986a) and they can be seen as feed-forward neural networks where the input patterns are also the desired output patterns. We can therefore define a centered AE by centering the encoder and decoder, which for a centered three layer AE corresponds to

$$\begin{aligned} encode(\mathbf{x}) &= \phi^{enc} \left(\mathbf{W}' \left(\mathbf{x} - \boldsymbol{\mu} \right) + \mathbf{c} \right) &= \mathbf{h}, \\ decode(\mathbf{h}) &= \phi^{dec} \left(\mathbf{W} \left(\mathbf{h} - \boldsymbol{\lambda} \right) + \mathbf{b} \right) &= \tilde{\mathbf{x}}, \end{aligned}$$

with encoder matrix $\mathbf{W}'$, decoder matrix $\mathbf{W}$, encoder bias $\mathbf{c}$, decoder bias $\mathbf{b}$, encoder offset $\boldsymbol{\mu}$, decoder offset $\boldsymbol{\lambda}$, encoder activation function ϕ^{enc} and decoder activation function ϕ^{dec}. It is common to assume tied weights, which means that the encoder matrix is just the transpose of the decoder matrix ($\mathbf{W}' = \mathbf{W}^T$). Although this reduces the number of free parameters AEs can still learn trivial representations if the number of hidden units is not chosen to be much smaller than the number of input dimensions or if the network is not regularized. Common regularized variants of AEs are denoising AEs (Vincent et al., 2008), sparse AEs (Olshausen et al., 1996; Poultney et al., 2006; Ng, 2011) and contractive AEs (Bishop, 1995; Rifai et al., 2011).

When choosing the activation functions for the encoder and decoder (that are sigmoid, tangens-hyperbolicus, radial-basis, linear, linear-rectifier, ...), we have to ensure that the encoder activation function is appropriate for the input data (for example a sigmoid cannot represent negative values). It is worth mentioning that when using the sigmoid function

for ϕ^{enc} and ϕ^{dec} and tied weights, the encoder becomes equivalent to Equation (5) and the decoder becomes equivalent to Equation (6). The network structure therefore becomes equivalent to an RBM such that the only difference is the training objective (that is the loss function).

4.2 Initialization of the Model Parameters

It is a common way to initialize the weight matrix of ANNs to small random values to break the symmetry. The bias parameters are often initialized to zero. However, we argue that there exists a more reasonable initialization for the bias parameters.

Hinton (2010) has proposed to initialize the RBM's visible bias parameter b_i to $\ln(p_i/(1-p_i))$, where p_i is the proportion of the data points in which unit i is on (that is $p_i = \langle x_i \rangle_d$). He has stated that if this is not done, the hidden units are used to activate the ith visible unit with a probability of approximately p_i in the early stage of training.

We argue that this initialization is in fact reasonable since it corresponds to the Maximum Likelihood Estimate (MLE) of the visible bias given the data for an RBM with zero weight matrix, given by

$$\mathbf{b}^* = \ln\left(\frac{\langle \mathbf{x} \rangle_d}{\mathbf{1} - \langle \mathbf{x} \rangle_d}\right) = -\ln\left(\frac{\mathbf{1}}{\langle \mathbf{x} \rangle_d} - \mathbf{1}\right) = \sigma^{-1}(\langle \mathbf{x} \rangle_d) \ , \tag{17}$$

where σ^{-1} is the inverse sigmoid function. Note that the MLE of the visible bias for an RBM with zero weights is the same whether the RBM is centered or not. The conditional probability of the visible variables of an RBM with this initialization is then given by $p(\mathbf{x} = \mathbf{1}|\mathbf{h}) = \sigma(\sigma^{-1}(\langle \mathbf{x} \rangle_d)) = \langle \mathbf{x} \rangle_d$, where $p(\mathbf{x} = \mathbf{1}|\mathbf{h})$ denotes the vector containing the elements $p(x_i = 1|\mathbf{h})$. Thus, the mean of the data is initially modeled only by the bias values and the weights are free to model higher order statistics in the beginning of training. For the unknown hidden variables it is reasonable to assume an initial mean of 0.5 so that the MLE of the hidden bias for an RBM with zero weights is given by $\mathbf{c}^* = \sigma^{-1}(0.5) = 0.0$. These considerations still hold approximately if the weights are not zero but initialized to small random values. For bigger initial weight values the model can be initially centered by using the inverse sigmoid of the average hidden activity given the data ($\mathbf{c}^* = \sigma^{-1}(\langle p(\mathbf{h} = \mathbf{1}|\mathbf{x}) \rangle_d)$) for the hidden biases.

Montavon and Müller (2012) have suggested to initialize the bias parameters of centered DBMs to the inverse sigmoid of the initial offset parameters. They have argued that this initialization leads to a good starting point, because it guarantees that the Boltzmann machine is initially centered. Actually, if the initial offsets are set to $\mu_i = \langle x_i \rangle_d$ and $\lambda_j = 0.5$ the initialization suggested by Montavon and Müller (2012) is equal to the initialization to the MLEs as follows from Equation (17).

Note that this initialization is not restricted to RBMs or the sigmoid activation function. Independently of the initial weight matrix we can always set the bias in ANNs to the inverse activation function of the corresponding mean value.

5. Methods

As shown in Section 3 the algorithms described by Cho et al. (2011), Tang and Sutskever (2011) and Montavon and Müller (2012) can all be viewed as different ways of applying

the centering trick. They differ in the choice of the offset parameters and in the way of approximating them, either based on the samples gained from the model in the previous learning step or from the current one, using an exponentially moving average or not. The question arises, how RBMs should be centered to achieve the best performance in terms of the LL. In Section 5 we therefore empirically analyze the different ways of centering and try to achieve a deeper understanding of why centering is beneficial.

For simplicity we introduce the following shorthand notation. The letter d denotes the data mean $\langle \cdot \rangle_d$, m denotes the model mean $\langle \cdot \rangle_m$, a denotes the average of the means $\frac{1}{2}\langle \cdot \rangle_d + \frac{1}{2}\langle \cdot \rangle_m$, and 0 is used if the offsets are set to zero. We indicate the choice of $\boldsymbol{\mu}$ in the first and the choice of $\boldsymbol{\lambda}$ in the second place, for example dm translates to $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_d$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_m$. We add a superscribed b (before) or l (later/after) to denote whether the reparameterization is performed before or after the gradient update. If the sliding factor in Algorithm 1 or 2 is set to a value smaller than one and thus an exponentially moving average is used, a subscript s (sliding) is added. Thus, we represent the variant of Cho

Abbr.	μ	λ	Description
00	$\mathbf{0}$	$\mathbf{0}$	Normal binary RBM (Smolensky, 1986)
$d0$	$\langle \mathbf{x} \rangle_d$	$\mathbf{0}$	Data Normalization RBM (Tang and Sutskever, 2011)
dd_s^l	$\langle \mathbf{x} \rangle_d$	$\langle \mathbf{h} \rangle_d$	Original Centered RBM (Montavon and Müller, 2012) reparam. after gradient update, use of an exp. moving average
aa^b	$0.5\,(\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m)$	$0.5\,(\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m)$	Enhanced gradient RBM (Cho et al., 2011) reparam. before gradient update, no use of an exp. moving average
dd_s^b	$\langle \mathbf{x} \rangle_d$	$\langle \mathbf{h} \rangle_d$	Centering using the data mean, reparam. before gradient update, use of an exp. moving average
mm_s^b	$\langle \mathbf{x} \rangle_m$	$\langle \mathbf{h} \rangle_m$	Centering using the model mean, reparam. before gradient update, use of an exp. moving average
dm_s^b	$\langle \mathbf{x} \rangle_d$	$\langle \mathbf{h} \rangle_m$	Centering using the data mean for the visible and the model mean for hidden units, reparam. before gradient update, use of an exp. moving average

Table 1: Look-up table: Abbreviations for the most frequently used algorithms.

et al. (2011) by aa^b, the one of Montavon and Müller (2012) by dd^l_s, the data normalization of Tang and Sutskever (2011) by *d0*, and the normal binary RBM simply by *00*. Table 1 summarizes the abbreviations most frequently used in this paper.

We begin our analysis with experiments on RBMs, where one layer is small enough to guarantee that the exact LL is still tractable. In a first set of experiments we analyze the four algorithms described above in terms of the evolution of the LL during training. In a second set of experiments we analyze the effect of the initializations described in Section 4.2. We proceed with a comparison of the effects of estimating offset values and reparameterizing the parameters before or after the gradient update. Afterwards we analyze the effects of using an exponentially moving average to approximate the offset values in the different algorithms and of choosing other offset values. We then compare the normal and the centered gradient with the natural gradient. Finally, to verify whether the results scale to more realistic problem sizes we compare RBMs, DBMs and AE on ten large data sets.

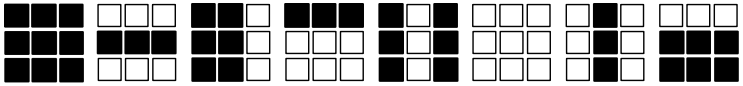
5.1 Benchmark Problems

We consider twelve different benchmark problems in our detailed analysis, see also Figure 2. The ***Bars & Stripes*** (MacKay, 2003) data set consists of patterns of size $D \times D$ that can be generated as follows. First, for each row of a pattern all corresponding pixels are either set to zero or to one with equal probability. Second, with a probability of 0.5 the pattern is rotated by 90 degrees. This leads to $N = 2^{D+1}$ patterns where the completely uniform patterns occur twice as often as the others. The data set is symmetric in terms of the amount of zeros and ones and thus the flipped and unflipped problems are equivalent. An upper bound of the LL is given by $(N-4)\ln\left(\frac{1}{N}\right) + 4\ln\left(\frac{2}{N}\right)$. For our experiments we used $D = 3$ or $D = 2$ (only in Section 6.9) leading to an upper bound of -41.59 and -13.86, respectively.

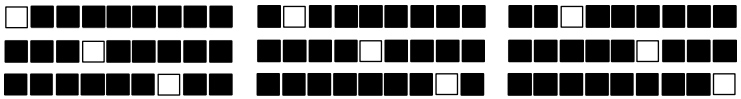
The ***Shifting Bar*** data set is an artificial benchmark problem we have designed to be asymmetric in terms of the amount of zeros and ones in the data. The data set consists of vectors of size N where a set of B successive pixels with cyclic boundary conditions are set to one, the 'bar', and the others are set to zero. More formally the data set can be generated as follows, beginning with a vector where all pixels are set to zero, an index $0 \leq i < N$ is chosen uniformly at random. The pixel with index i and the following $B-1$ pixels, which underly cyclic boundary conditions are set to one (that is the pixels with indices $\{i,\ (i+1) \mod N, \ldots, (i+B-1) \mod N\}$ are set to one). This leads to N different patterns with equal probability, and an upper bound of the LL of $N\ln\left(\frac{1}{N}\right)$, the percentage of ones in the data set is $\frac{B}{N}$. For our experiments we used $N = 9$, $B = 1$ and its flipped version ***Flipped Shifting Bar***, which we get for $N = 9$, $B = 8$, both having an upper LL bound of -19.78.

The ***MNIST*** (LeCun et al., 1998) data set of handwritten digits has become a standard benchmark problem for RBMs. It consists of $60,000$ training and $10,000$ testing examples of gray value handwritten digits of size 28×28. Usually, in a first preprocessing step the values are normalized to lie in $[0, 1]$. Across different studies the normalized values are then either used directly as input, or the data set is binarized using a threshold of 0.5 or by sampling according to the normalized values that are treated as probabilities.[4] In this study

4. In Appendix D we give a an comparison of normal and centered RBMs on the different *MNIST* versions.



(a) 8 out of 16 patterns from the *Bars & Stripes* data set.



(b) All patterns of the *Shifting Bar* data set with $N = 9$ and $B = 1$.

(c) Example patterns from the *MNIST* data set after binarization.

(d) Example patterns from the *Caltech 101 Silhouette* data set.

Figure 2: Some patterns from the different benchmark problems.

we binarized *MNIST* with a threshold of 0.5. The data set contains 13.3% ones, similar to the *Shifting Bar* problem, which for our choice of N and B contains 11.1% ones. We refer to the data set where each bit of *MNIST* is flipped (that is each one is replaced by a zero and *vice versa*) as ***1-MNIST***.

The ***CalTech 101 Silhouettes*** (Marlin et al., 2010) data set consists of 4100 training, 2307 validation, and 2264 testing examples of binary object silhouettes of size 28×28. The data set contains 55.1% ones, and thus (like in the *Bars & Stripes problem*) the amount of zeros and ones is almost the same. The background pixels take the value one, which is in contrast to *MNIST* where the background pixels are set to zero.

In some experiments we also considered the eight ***UCI binary*** (Larochelle et al., 2010; Larochelle and Murray, 2011) data sets from different domains comprising biological, image, text and game-related data. The data sets differ in dimensionality (112 to 500) and size (a few hundred to several thousand examples) and have been separated into training, validation, and test sets. All *UCI binary* data sets contain less ones than zeros, where the percentage of ones lies between 3.9% and 36.8%.

6. Results

For all models in this work the weight matrices were initialized with random values sampled from a Gaussian with zero mean and a standard deviation of 0.01. If not stated otherwise the visible biases, hidden biases, and offsets were initialized as described in Section 4.2.

We began our analysis with experiments on small RBMs where the LL can be calculated exactly, where we used 4 hidden units when modeling *Bars & Stripes* and *Shifting Bar* and 16 hidden units when modeling *MNIST*. For training we used CD-1, PCD-1 and PT_c (with $c = 10$ or $c = 20$) where the c temperatures were distributed uniformly from 0 to 1. For *Bars & Stripes* and *Shifting Bar* full-batch training was performed for 50,000 gradient updates, where the LL was evaluated every 50th gradient update. For modeling *MNIST* mini-batch training with a batch size of 100 was performed for 100 epochs, each consisting of 600 gradient updates and the exact LL was evaluated after each epoch. Note that in order to get an unbiased comparison of the different models, we did not use any additional modifications of the update rule like a momentum term, weight decay or an annealing learning rate.

The tables in this work show the maximum average LL and the corresponding standard deviation reached during training averaged over 25 trials. In addition the final average LL reached at the end of training is given in parenthesis to indicate a potential divergence of the LL. For some computational expensive experiments we used only 10 trials, which will be mentioned explicitly. For reasons of readability, the LL values for the big data sets were normalized to the number of training samples. In order to check if the result of the best method within one row differs significantly from the others we performed pairwise signed Wilcoxon rank-sum tests (with $p = 0.05$). The best results are highlighted in bold. This can be more than one value if the significance test between these values was negative.

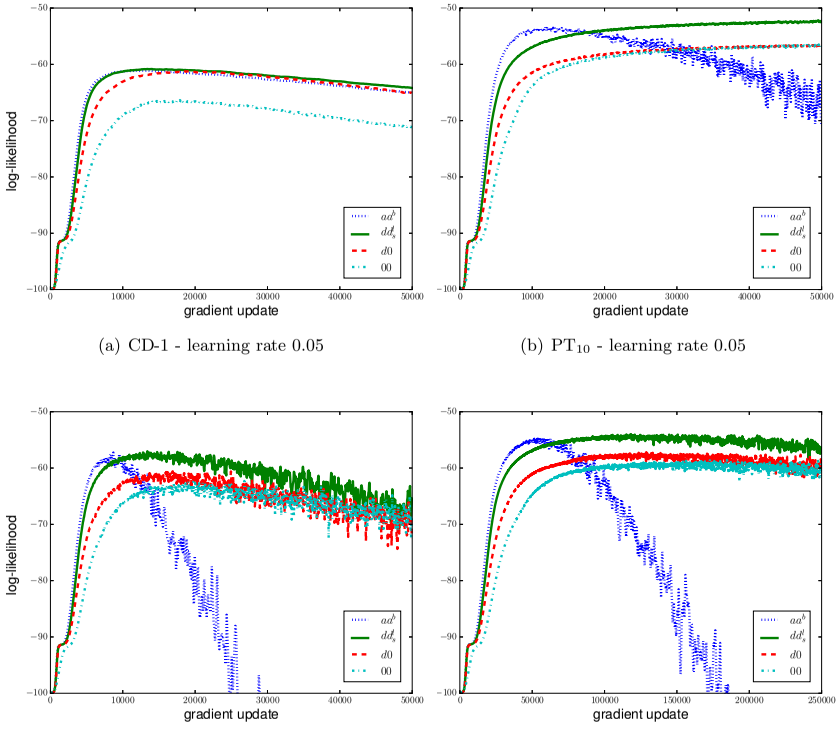
6.1 Comparison of the Standard Methods

The comparison of the learning performance of the previously described algorithms dd_s^l, aa^b, *d0*, and *00* (using their originally proposed initializations) shows that training a centered RBM leads to significantly higher LL values than training a normal binary RBM (see Table 2 for the results on *Bars & Stripes* and *MNIST* and Table 3 for the results on *Shifting Bar* and *Flipped Shifting Bar*).

Figure 3 illustrates on the *Bars & Stripes* data set that centering (aa^b, dd_s^l, and *d0*) leads to faster learning and higher LL values than normal RBMs (*00*). This differs from the observations made for DBMs by Montavon and Müller (2012), who have found a better generative performance of centering only in the case of locally connected DBMs. Furthermore, Figure 3 shows that centering both the visible and the hidden variables (dd_s^l and aa^b) compared to centering only the visible variables (*d0*) accelerates learning and leads to a higher LL. This is different from the observations made for RBMs by Tang and Sutskever (2011), who have stated that centering only the visible variables leads to a performance similar to that of the enhanced gradient. Thus centered RBMs where visible and hidden variables are centered can form more accurate models of the data distribution than normal RBMs and centered RBMs where only the visible variables are centered.

It can also be seen from Figure 3 that all methods show divergence in combination with CD and PCD (as described before by Fischer and Igel, 2010, for normal RBMs), which

is prevented for dd_s^l, *d0*, and *00* when using PT as shown in Figure 3(b). This can be explained by the fact that PT leads to faster mixing Markov chains and thus less biased gradient approximations. The *aa* algorithm however suffers from severe divergence of the LL when PCD or PT is used, which is even worse than with CD. This divergence problem does occur independently of the choice of the learning rate as indicated by the LL values reached at the end of training (given in parentheses) in Table 2 and Table 3, and which can



(a) CD-1 - learning rate 0.05

(b) PT_{10} - learning rate 0.05

(c) PCD-1 - learning rate 0.05

(d) PCD-1 - learning rate 0.01

Figure 3: Evolution of the average LL on the training data for models with 4 hidden units on the *Bars & Stripes* data set for the standard centering methods. (a) CD-1 with a learning rate of $\eta = 0.05$, (b) PT_{10} with a learning rate of $\eta = 0.05$, (c) PCD-1 with a learning rate of $\eta = 0.05$, and (d) PCD-1 with a learning rate of $\eta = 0.01$.

Algorithm-η	aa^b		dd_s^l		$d0$		00	
Bars & Stripes								
CD-1-0.1	**-61.66**	±1.71 (-69.1)	**-61.33**	±1.89 (-69.1)	**-61.73**	±4.19 (-70.9)	-66.53	±3.47 (-78.1)
CD-1-0.05	**-61.09**	±1.89 (-65.0)	**-60.82**	±2.31 (-64.2)	**-61.23**	±3.70 (-65.1)	-66.35	±4.24 (-71.2)
CD-1-0.01	**-61.08**	±1.60 (-61.1)	**-61.31**	±1.55 (-61.3)	-63.30	±3.02 (-63.3)	-68.56	±2.94 (-68.6)
PCD-1-0.1	**-59.05**	±2.07 (-360.6)	**-58.93**	±2.24 (-102.7)	-62.46	±4.65 (-97.3)	-63.28	±5.61 (-84.3)
PCD-1-0.05	**-57.25**	±1.51 (-167.4)	**-57.01**	±1.61 (-65.4)	-60.42	±5.19 (-72.5)	-62.29	±4.84 (-70.6)
PCD-1-0.01	**-54.86**	±0.98 (-55.3)	-56.80	±0.73 (-56.8)	-61.03	±3.74 (-61.0)	-64.61	±2.95 (-64.6)
PT_{10}-0.1	-54.67	±3.43 (-202.5)	**-51.99**	±1.13 (-52.2)	-56.18	±5.32 (-56.7)	-54.89	±3.66 (-55.3)
PT_{10}-0.05	-53.34	±0.99 (-70.7)	**-52.21**	±1.06 (-52.5)	-56.54	±5.82 (-56.6)	-56.46	±4.43 (-56.8)
PT_{10}-0.01	**-53.63**	±1.25 (-53.8)	-56.76	±0.79 (-56.8)	-61.26	±4.58 (-61.3)	-64.71	±3.52 (-64.7)
MNIST								
CD-1-0.1	**-153.01**	±2.32 (-158.0)	**-153.46**	±2.45 (-155.5)	**-153.89**	±2.91 (-155.0)	-170.29	±2.45 (-170.6)
CD-1-0.05	**-153.07**	±1.51 (-156.0)	**-152.78**	±3.57 (-153.8)	**-153.01**	±2.80 (-155.4)	-169.43	±2.95 (-169.7)
CD-1-0.01	**-152.90**	±1.38 (-152.9)	**-152.42**	±2.15 (-152.4)	-153.77	±2.54 (-153.8)	-172.70	±1.86 (-172.8)
PCD-1-0.1	-150.78	±1.95 (-174.9)	**-144.10**	±1.67 (-144.7)	-154.64	±3.14 (-156.6)	-183.11	±8.62 (-183.2)
PCD-1-0.05	-146.83	±1.54 (-163.4)	**-141.93**	±1.25 (-142.9)	-146.86	±2.08 (-147.5)	-185.78	±6.78 (-185.8)
PCD-1-0.01	-143.36	±0.74 (-144.1)	**-140.43**	±0.62 (-140.7)	-141.88	±1.02 (-141.9)	-193.60	±4.56 (-193.6)
PT_{10}-0.01	-259.50	±17.54 (<-999)	**-142.00**	±1.04 (-142.3)	-145.84	±2.49 (-147.4)	-152.38	±3.05 (-152.4)

Table 2: Maximum average LL on the training data for centered and normal RBMs (top) with 4 hidden units on the *Bars & Stripes* data set and (bottom) with 16 hidden units on the *MNIST* data set using different sampling methods and learning rates η. Each LL value is followed by the corresponding standard deviation of the 25 trials and the average LL at the end of training is given in parenthesis.

Algorithm-η	aa^b		dd_s^l		$d0$		00	
Shifting Bar								
CD-1-0.2	-21.04	±1.41 (-21.9)	**-20.42**	±0.85 (-20.4)	-22.15	±1.38 (-22.5)	-22.32	±1.46 (-22.6)
CD-1-0.1	-21.29	±1.18 (-21.5)	**-20.76**	±0.80 (-20.8)	-21.35	±0.95 (-21.4)	-21.56	±0.94 (-21.6)
CD-1-0.05	**-21.16**	±0.84 (-21.2)	-22.74	±0.65 (-22.7)	-26.89	±0.29 (-26.9)	-26.11	±0.40 (-26.1)
PCD-1-0.2	-23.07	±0.78 (-237.2)	**-22.48**	±0.85 (-33.6)	-23.15	±1.01 (-31.9)	-23.34	±0.75 (-31.7)
PCD-1-0.1	**-22.14**	±0.70 (-87.4)	**-21.92**	±0.66 (-24.0)	-22.75	±0.84 (-23.7)	-22.64	±1.15 (-23.3)
PCD-1-0.05	**-21.71**	±0.73 (-26.0)	-22.53	±0.55 (-22.5)	-26.84	±0.36 (-26.8)	-26.06	±0.48 (-26.1)
PT_{10}-0.2	-21.30	±0.81 (-31.9)	**-20.66**	±0.67 (-20.8)	-21.47	±1.13 (-21.6)	-22.12	±1.17 (-22.3)
PT_{10}-0.1	-20.84	±0.62 (-21.5)	**-20.57**	±0.56 (-20.6)	-21.34	±0.90 (-21.4)	-21.21	±0.93 (-21.2)
PT_{10}-0.05	**-20.78**	±0.88 (-20.8)	-22.42	±0.69 (-22.4)	-26.96	±0.30 (-27.0)	-26.18	±0.38 (-26.2)
Flipped Shifting Bar								
CD-1-0.2	**-20.82**	±1.15 (-21.3)	**-20.73**	±1.06 (-20.8)	-22.04	±1.62 (-22.3)	-28.14	±0.27 (-28.2)
CD-1-0.1	**-20.85**	±1.05 (-20.9)	**-20.90**	±0.83 (-20.9)	-21.17	±0.84 (-21.2)	-28.35	±0.04 (-28.4)
CD-1-0.05	**-21.18**	±0.82 (-21.2)	-22.64	±0.65 (-22.6)	-26.85	±0.34 (-26.9)	-28.31	±0.02 (-28.3)
PCD-1-0.2	-22.73	±0.88 (-310.8)	**-22.35**	±0.91 (-29.1)	-23.48	±0.84 (-32.6)	-28.19	±0.28 (-28.3)
PCD-1-0.1	-22.15	±0.71 (-88.3)	**-21.64**	±0.72 (-22.6)	-22.38	±0.88 (-23.2)	-28.35	±0.04 (-28.4)
PCD-1-0.05	**-21.72**	±0.86 (-25.6)	-22.34	±0.60 (-22.3)	-26.90	±0.34 (-26.9)	-28.31	±0.02 (-28.3)
PT_{10}-0.2	-21.13	±0.72 (-32.4)	**-20.57**	±0.63 (-20.6)	-21.14	±0.84 (-21.4)	-28.17	±0.26 (-28.2)
PT_{10}-0.1	-21.03	±0.81 (-21.6)	**-20.73**	±0.74 (-20.8)	-21.26	±0.72 (-21.3)	-28.35	±0.04 (-28.4)
PT_{10}-0.05	**-21.01**	±0.86 (-21.1)	-22.48	±0.73 (-22.5)	-26.87	±0.35 (-26.9)	-28.31	±0.02 (-28.3)

Table 3: Maximum average LL on the training data for centered and normal RBMs with 4 hidden units on (top) the *Shifting Bar* data set and (bottom) the *Flipped Shifting Bar* data set using different sampling methods and learning rates η. Each LL value is followed by the corresponding standard deviation of the 25 trials and the average LL at the end of training is given in parenthesis.

also be seen by comparing Figure 3(c) and Figure 3(d). The divergence occurs earlier and is more severe for bigger learning rates, while for the other algorithms we never observed divergence in combination with PT even for very big learning rates and long training time. The reasons for this divergence are discussed in detail in Section 6.4.

The results in Table 3 also demonstrate the flip invariance of the centered RBMs on the *Shifting Bar* data set empirically. Whereas *00* fails to model the flipped version of the data set correctly, dd_s^l, aa^b, and *d0* have approximately the same performance on the flipped and unflipped data set.

6.2 Initialization

The experiments in this section were done to analyze the effects of different initializations of the parameters as discussed in Section 4.2. First, we trained normal binary RBMs (that is *00*) where the visible bias was initialized to zero or to the inverse sigmoid of the data mean. In both cases the hidden bias was initialized to zero. Table 4 shows the results for normal binary RBMs trained on the *Flipped Shifting Bar* data set, where RBMs with zero initialization failed to learn the distribution accurately. The RBMs using the inverse sigmoid initialization achieved better performance and therefore seem to be less sensitive

Algorithm-η	00 *init zero*			00 *init* σ^{-1}		
Flipped Shifting Bar						
CD-1-0.2	-28.14	±0.27	(-28.2)	**-22.08**	±1.49	(-22.5)
CD-1-0.1	-28.35	±0.04	(-28.4)	**-21.40**	±1.21	(-21.6)
CD-1-0.05	-28.31	±0.02	(-28.3)	**-24.90**	±0.47	(-24.9)
PCD-1-0.2	-28.19	±0.28	(-28.3)	**-25.17**	±1.32	(-42.3)
PCD-1-0.1	-28.35	±0.04	(-28.4)	**-23.71**	±0.98	(-26.7)
PCD-1-0.05	-28.31	±0.02	(-28.3)	**-24.99**	±0.57	(-25.0)
PT_{10}-0.2	-28.17	±0.26	(-28.2)	**-22.83**	±1.41	(-23.5)
PT_{10}-0.1	-28.35	±0.04	(-28.4)	**-21.52**	±0.88	(-21.8)
PT_{10}-0.05	-28.31	±0.02	(-28.3)	**-24.87**	±0.50	(-24.9)
MNIST						
CD-1-0.1	**-170.29**	±2.45	(-170.6)	**-169.62**	±4.45	(-169.8)
CD-1-0.05	**-169.43**	±2.95	(-169.7)	-171.12	±3.05	(-171.2)
CD-1-0.01	-172.70	±1.86	(-172.8)	**-169.97**	±2.02	(-172.3)
PCD-1-0.1	-183.11	±8.62	(-183.2)	**-164.68**	±3.27	(-189.7)
PCD-1-0.05	-185.78	±6.78	(-185.8)	**-156.93**	±4.10	(-158.3)
PCD-1-0.01	-193.60	±4.56	(-193.6)	**-144.28**	±0.93	(-144.3)
PT_{10}-0.01	-152.38	±3.05	(-152.4)	**-148.79**	±2.44	(-150.7)

Table 4: Maximum average LL on the training data for normal RBMs (top) with 4 hidden units on the *Flipped Shifting Bar* data set and (bottom) with 16 hidden units on the *MNIST* data set, where the visible bias is initialized to zero or to the inverse sigmoid of the data mean.

Algorithm-η	dd_s^l *init zero*			dd_s^l *init* σ^{-1}		
Flipped Shifting Bar						
CD-1-0.2	**-20.76**	±0.95	(-20.8)	**-20.73**	±1.06	(-20.8)
CD-1-0.1	**-20.82**	±0.88	(-20.8)	**-20.90**	±0.83	(-20.9)
CD-1-0.05	**-22.98**	±0.73	(-23.0)	**-22.64**	±0.65	(-22.6)
PCD-1-0.2	**-22.32**	±0.82	(-31.8)	**-22.35**	±0.91	(-29.1)
PCD-1-0.1	**-21.75**	±0.60	(-22.6)	**-21.64**	±0.72	(-22.6)
PCD-1-0.05	-22.79	±0.64	(-22.8)	**-22.34**	±0.60	(-22.3)
PT_{10}-0.2	**-20.37**	±0.36	(-20.6)	**-20.57**	±0.63	(-20.6)
PT_{10}-0.1	**-20.74**	±0.75	(-20.8)	**-20.73**	±0.74	(-20.8)
PT_{10}-0.05	-22.95	±0.70	(-22.9)	**-22.48**	±0.73	(-22.5)

Table 5: Maximum average LL on the training data for centered RBMs (dd_s^l) with 4 hidden units on the *Flipped Shifting Bar* data set, where the visible bias is initialized to zero or to the inverse sigmoid of the data mean.

to the 'difficulty' of the data representation. However, the results are not as good as the results of the centered RBMs shown in Table 3. The same observations can be made when training RBMs on the *MNIST* data set (see Table 4 bottom). The RBMs with inverse sigmoid initialization achieved significantly better results than RBMs initialized to zero in the case of PCD and PT, but still worse compared to the centered RBMs. Furthermore, using the inverse sigmoid initialization allows us to achieve similar performance on *MNIST* and *1-MNIST*, while the RBM with zero initialization failed to learn the distribution for *1-MNIST* at all (results not shown). A visual comparison of the filters of normal RBMs trained on *MNIST* and *1-MNIST* has already been given by Cho et al. (2011) and Tang and Sutskever (2011).

We then trained models using the centering versions *dd*, *aa*, and *d0* comparing the initialization suggested in Section 4.2 against the initialization to zero. In most cases the results showed no significant difference in terms of the maximum LL reached in trials with different initializations, or slightly better results were found when using the inverse sigmoid in combination with a small learning rate. This can be explained by the better starting point yielded by this initialization. See Table 5 for the results for dd_s^l on the *Flipped Shifting Bar* data set as an example. We therefore used the inverse sigmoid initialization in the following experiments since it was beneficial for normal RBMs.

6.3 Reparameterization

To investigate the effects of performing the reparameterization before or after the gradient update during training of centered RBMs , we analyzed the learning behavior of dd_s^b (the algorithm suggested here) and dd_s^l (the algorithm suggested by Montavon and Müller, 2012) on all data sets. The results for RBMs trained on the *Bars & Stripes* data set and the *MNIST* data set are given in Table 6 (top). No significant difference between the two versions can be observed and the same result (not shown) was obtained for the *Shifting Bar*

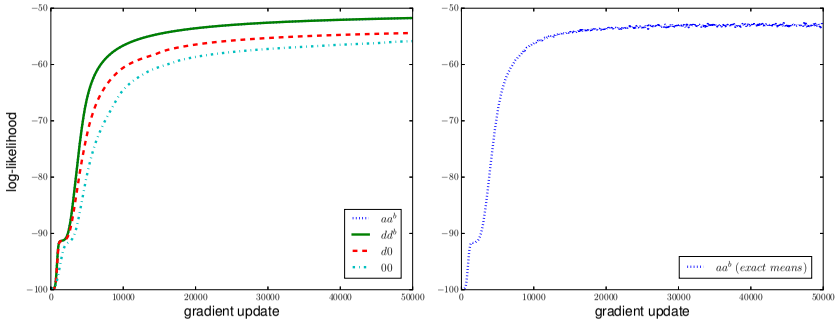
Algorithm-η	dd_s^b			dd_s^l		
Bars & Stripes						
CD-1-0.1	**-61.32**	±1.87	(-69.1)	**-61.33**	±1.89	(-69.1)
CD-1-0.05	**-60.82**	±2.26	(-64.2)	**-60.82**	±2.31	(-64.2)
CD-1-0.01	**-61.31**	±1.55	(-61.3)	**-61.31**	±1.55	(-61.3)
PCD-1-0.1	**-59.08**	±1.53	(-94.5)	**-58.93**	±2.24	(-102.7)
PCD-1-0.05	**-56.97**	±1.74	(-64.3)	**-57.01**	±1.61	(-65.4)
PCD-1-0.01	**-56.88**	±0.75	(-56.9)	**-56.80**	±0.73	(-56.8)
PT_{10}-0.1	**-51.99**	±1.29	(-52.5)	**-51.99**	±1.13	(-52.2)
PT_{10}-0.05	**-52.34**	±1.15	(-52.5)	**-52.21**	±1.06	(-52.5)
PT_{10}-0.01	**-56.76**	±0.77	(-56.8)	**-56.76**	±0.79	(-56.8)
MNIST						
CD-1-0.1	**-153.16**	±2.57	(-154.9)	**-153.46**	±2.45	(-155.5)
CD-1-0.05	**-152.34**	±3.61	(-153.1)	**-152.78**	±3.57	(-153.8)
CD-1-0.01	**-152.23**	±2.03	(-152.2)	**-152.42**	±2.15	(-152.4)
PCD-1-0.1	**-144.12**	±2.12	(-145.2)	**-144.10**	±1.67	(-144.7)
PCD-1-0.05	**-141.88**	±1.28	(-142.2)	**-141.93**	±1.25	(-142.9)
PCD-1-0.01	**-140.40**	±0.66	(-140.7)	**-140.43**	±0.62	(-140.7)
PT_{10}-0.01	**-141.95**	±1.22	(-142.5)	**-142.00**	±1.04	(-142.3)

Table 6: Maximum average LL on the training data for RBMs (top) with 4 hidden units on the *Bars & Stripes* data set and (bottom) with 16 hidden units on the *MNIST* data set, using the reparameterization before (dd_s^b) and after (dd_s^l) the gradient update.

and the *Flipped Shifting Bar* data set. We therefore reparameterize the RBMs before the gradient update in the remainder of this work. This is also motivated by the fact that the enhanced gradient is only equivalent to centering using the average of model and data mean for visible and hidden offsets if the model is reparameterized before it is updated (that is, the enhanced gradient is equivalent to aa^b but not to aa^l).

6.4 Analyzing the Model-Mean-Related Divergence Effect

The severe divergence problem observed when using the enhanced gradient in combination with PCD or PT raises the question whether the problem is induced by setting the offsets to $0.5(\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m)$ and $0.5(\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m)$ or by bad sampling based estimates of gradient and offsets. We therefore trained centered RBMs with 9 visible and 4 hidden units on the *Bars & Stripes* data set using either the exact gradient where only $\langle \mathbf{x} \rangle_m$ and $\langle \mathbf{h} \rangle_m$ were approximated by samples or using PT_{10} estimates of the gradient while $\langle \mathbf{x} \rangle_m$ and $\langle \mathbf{h} \rangle_m$ were calculated exactly. Figure 4(a) shows that if the exact gradient is used but the offsets are estimated, no divergence for aa in combination with PT is observed and the performance of aa and dd become almost equivalent. Interestingly, the divergence is also prevented if the gradient is estimated using PT but the offsets are calculated exactly as shown in Figure 4(b).



(a) Exact gradient with approximated offsets (b) PT_{10} with exact means

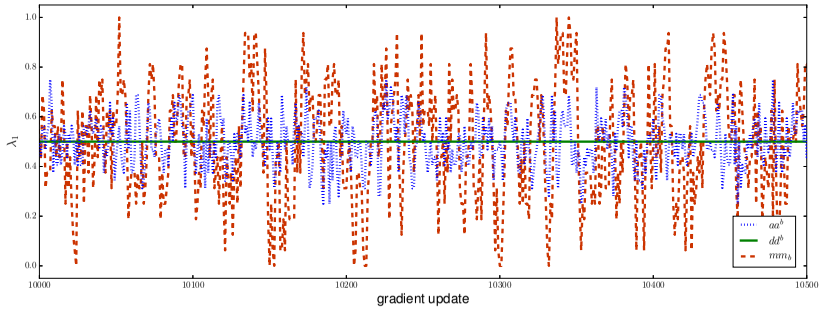
Figure 4: Evolution of the average LL on the training data for RBMs with 4 hidden units on the *Bars & Stripes* data set for the standard centering methods. (a) When the exact gradient is used but the offsets are estimated using PT_{10} and (b) when PT_{10} is used for estimating the gradient but the offsets are calculated exactly. In both cases the learning rate was $\eta = 0.05$.

Thus, the divergence behavior must be induced by approximating both, gradient and offsets. The fact that the mean under the data distribution can always be calculated exactly might explain why we do not observe divergence for dd in combination with PT.

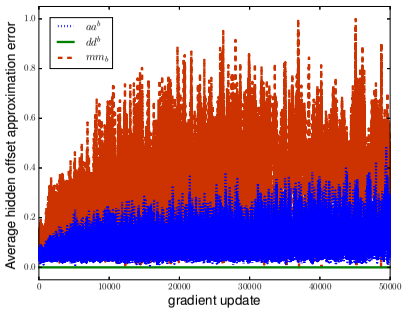
To further deepen the understanding of the divergence effect we investigated the parameter evolution during training of RBMs with different offsets. We observed that the change of the offset values between two gradient updates gets extremely large during training when using the model mean. Figure 5(a) shows in an exemplary manner the evolution of the first hidden offset λ_1 for a single trial, where the approximated offset for dd^b is almost constant while it is rather large for aa^b and even bigger for mm^b (the centering variant that uses the mean of hidden and visible variables under the model distribution as offsets). In each iteration we calculated the exact offsets to estimate the approximation error shown in Figure 5(b). Obviously, there is no approximation error for dd while the error for aa quickly gets large and the error for mm gets even twice as big. In combination with the gradient approximation error this causes the weight matrices for aa and mm to grow extremely big as shown in Figure 5(c). Consistent with the even larger approximation error for the offset estimate of mm, the divergence for mm becomes even worse than that for aa, as shown in Figure 6.

To further confirm that the divergence is not just caused by the additional sampling noise introduced by approximating the offsets, we trained a centered RBM using PT for the gradient approximation while we set the offsets to uniform random values between zero and one in each iteration. The results are shown in Figure 6(a) and demonstrate that even random offset values do not lead to the divergence problems. Thus, the divergence cannot

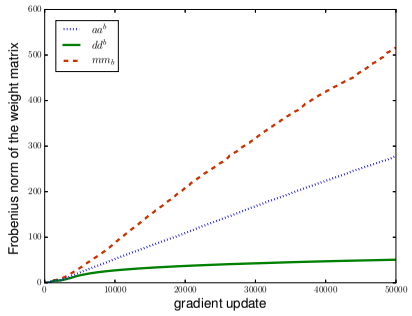
be caused by additional sampling noise but rather by the correlated errors of gradient and offset approximations. To test this hypothesis we investigated mm^b where the samples for offset and gradient approximations were taken from different PT sampling processes. The results are shown in Figure 6(b) where no divergence can be observed anymore. While creating two sets of samples for gradient and offset approximations prevents the LL from diverging it almost doubles the computational cost and can therefore not be considered as a reasonable solution in practice. Moreover, using the model mean as offset still leads to



(a) Close up of the offset evolution for λ_1

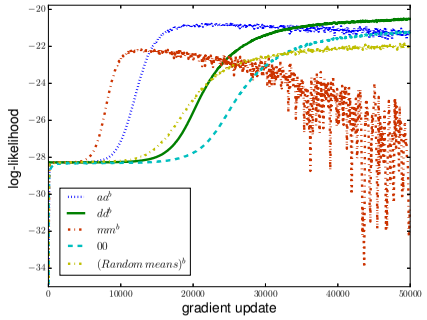


(b) Evolution of the offset approximation error

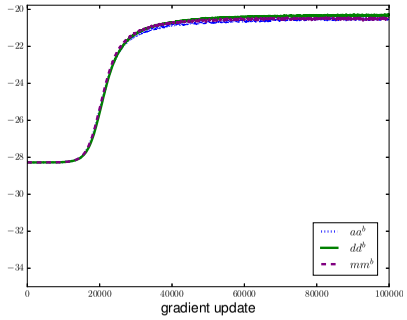


(c) Evolution of the weight norm

Figure 5: Evolution of the offsets and weights of different centered variants for RBMs with 4 hidden units during training on *Bars & Stripes* using PT_{10}. For clearness a single trial is shown, but the experiments were repeated 25 times and all trials showed qualitatively the same results. (a) Close up of the evolution of offset λ_1 over 500 gradient updates. (b) The evolution of the absolute difference between exact and approximated hidden offset averaged over all hidden units. (c) The evolution of the Frobenius norm of the weight matrices.



(a) Random offsets



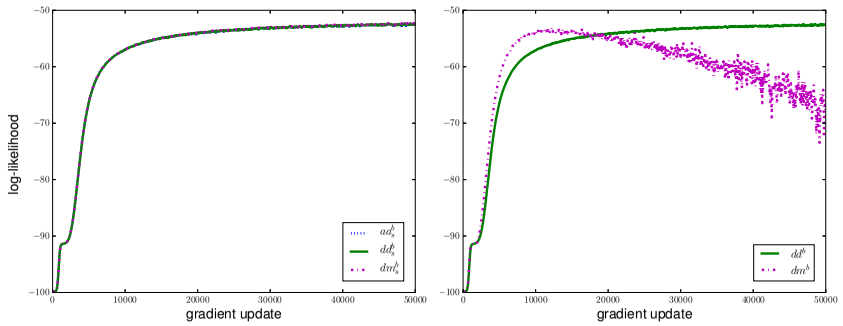
(b) Independent approximations

Figure 6: Evolution of the average LL on the training data for RBMs with 4 hidden units on the *Shifting Bars* data set using PT_{10} with a learning rate of $\eta = 0.1$. (a) Normal RBMs, centered RBMs, and centered RBMs with random offset values (denoted by *Random means*^{b}). (b) Centered RBMs when the samples for the offset approximations come from a different Markov chain than the samples used for the gradient approximation.

slightly worse final LL values than using the mean under the data distribution. This might be explained by the fact that the additional approximation of the model mean introduces noise while the data mean can be estimated exactly. Note, that the divergence also occurs if either only visible or hidden offsets are set to the PT-approximated model mean, which can be seen for example for dm in Figure 7(b).

Interestingly, the observed initially faster learning speed of mm and aa, which can be seen in Figure 6(a), does not occur anymore when offset and gradient approximation are based on different sample sets. This observation can also be made when the exact gradient is used (see Figure 4a). Thus, the initially faster learning speed seems also be caused by the correlated approximations of gradient and offsets and could be explained by the rapidly growing norm of the weight matrix shown in Figure 5(c), which effectively leads to a bigger step size in the parameter updates.

Note, that divergence can either be caused by using the expectation under the model distribution for the offset approximations as described in this section or by the bias of the gradient approximation introduced by using only a few steps of Gibbs sampling (Fischer and Igel, 2010; Schulz et al., 2010; Fischer and Igel, 2011). The latter can occur for all methods when CD or PCD is used for training. Furthermore, the LL can also decrease on the test data due to overfitting to the training data, which is not divergence in the proper sense.



(a) With exponentially moving average (b) Without exponentially moving average

Figure 7: Evolution of the average LL on the training data for RBMs with 4 hidden units on *Bars & Stripes* with different centering variants, using PT_{10} and a learning rate of $\eta = 0.05$. (a) Using an exponentially moving average with a sliding factor of 0.01 (the curves are almost equivalent) and (b) without exponentially moving average (see also Figure 3b for aa^b)

6.5 Usage of an Exponentially Moving Average

The approximation of the mean values of the variables using just a few samples might be too inaccurate or biased by the samples in the current batch. This is in particular the case when the model mean is used as shown in Section 6.4, but also when the data mean is used with a small batch size. Therefore, an exponentially moving average can be used to smooth the approximation of the offsets between parameter updates, which stabilizes the approximations when small batch sizes are used as well as when the model mean is used for the offsets.

We analyzed the impact of using an exponentially moving average with a sliding factor of 0.01 for the estimation of the offset parameters. Figure 7(a) illustrates on the *Bars& Stripes* data set that the learning curves of the different models become almost equivalent and that the divergence problem when using the model mean does not occur anymore when an exponentially moving average is used. However, the maximum LL values reached are the same whether an exponentially moving average is used or not. This can be seen by comparing Figure 7(a) and Figure 7(b) and also by comparing the results in Table 2 and Table 3 with those in Table 7. As discussed in Section 6.4, this problem is caused by the correlation between the approximation error of gradient and offsets. When using an exponential moving average the current offsets contain only a small fraction of the current mean such that the correlation is highly reduced.

In our experiments, *dd* does not suffer from the divergence problem when PT is used for sampling, even without exponentially moving average as can be seen in Figure 7(b) for

example or in the case of mini-batch learning (results not shown). Thus, *dd* seems to be generally more stable than the other centering variants, which is also true for big models as will be shown at the end of Section 6.7.

In the previous experiments an exponentially moving average was used for the approximation of visible and hidden offsets as originally suggested by Montavon and Müller (2012). Note however, that in batch learning when $\langle \mathbf{x} \rangle_d$ is used for the visible offsets, these values stay constant such that an exponentially moving average has no effect. More generally

Algorithm-η	aa_s^b	dd_s^b	dm_s^b
Bars & Stripes			
CD-1-0.1	**-60.90** ±2.14 (-70.5)	**-61.32** ±1.87 (-69.1)	**-61.09** ±2.18 (-68.8)
CD-1-0.05	**-60.25** ±2.62 (-64.2)	**-60.82** ±2.26 (-64.2)	**-60.83** ±2.31 (-64.2)
CD-1-0.01	**-60.87** ±1.31 (-60.9)	**-61.31** ±1.55 (-61.3)	**-61.31** ±1.56 (-61.3)
PCD-1-0.1	**-58.80** ±3.03 (-199.4)	**-59.08** ±1.53 (-94.5)	**-58.53** ±2.42 (-177.3)
PCD-1-0.05	**-57.52** ±2.07 (-101.5)	**-56.97** ±1.74 (-64.3)	**-56.87** ±2.27 (-83.3)
PCD-1-0.01	**-57.31** ±1.34 (-57.3)	-56.88 ±0.75 (-56.9)	**-56.65** ±0.76 (-56.6)
PT_{10}-0.1	**-52.79** ±1.92 (-53.0)	**-51.99** ±1.29 (-52.5)	**-51.88** ±1.02 (-52.5)
PT_{10}-0.05	**-52.80** ±1.62 (-52.9)	**-52.34** ±1.15 (-52.5)	**-52.25** ±1.06 (-52.4)
PT_{10}-0.01	**-57.23** ±1.34 (-57.2)	**-56.76** ±0.77 (-56.8)	**-56.69** ±0.77 (-56.7)
Flipped Shifting Bar			
CD-1-0.2	**-20.79** ±1.11 (-20.8)	**-20.70** ±1.01 (-20.7)	**-20.64** ±1.05 (-20.7)
CD-1-0.1	**-21.07** ±0.91 (-21.1)	**-20.84** ±0.80 (-20.8)	**-21.01** ±0.87 (-21.0)
CD-1-0.05	**-22.52** ±0.61 (-22.5)	**-22.64** ±0.67 (-22.6)	**-22.61** ±0.68 (-22.6)
PCD-1-0.2	**-22.29** ±0.62 (-38.3)	**-22.21** ±0.85 (-34.0)	**-22.44** ±0.71 (-37.8)
PCD-1-0.1	**-21.88** ±0.65 (-23.9)	**-21.66** ±0.73 (-22.6)	**-21.75** ±0.59 (-23.8)
PCD-1-0.05	**-22.50** ±0.59 (-22.5)	**-22.36** ±0.60 (-22.4)	**-22.36** ±0.63 (-22.4)
PT_{10}-0.2	**-20.57** ±0.56 (-20.7)	**-20.47** ±0.57 (-20.6)	**-20.53** ±0.58 (-20.7)
PT_{10}-0.1	**-20.61** ±0.64 (-20.7)	**-20.62** ±0.61 (-20.7)	**-20.62** ±0.60 (-20.7)
PT_{10}-0.05	**-22.37** ±0.65 (-22.4)	**-22.48** ±0.72 (-22.5)	**-22.34** ±0.66 (-22.3)
MNIST			
CD-1-0.1	**-152.81** ±2.34 (-155.5)	**-153.16** ±2.57 (-154.9)	**-153.13** ±2.32 (-155.2)
CD-1-0.05	**-152.01** ±2.04 (-153.2)	**-152.34** ±3.61 (-153.1)	**-152.37** ±3.50 (-153.2)
CD-1-0.01	-153.20 ±1.89 (-153.2)	**-152.23** ±2.03 (-152.2)	**-152.13** ±2.04 (-152.1)
PCD-1-0.1	**-144.07** ±1.54 (-144.5)	**-144.12** ±2.12 (-145.2)	**-143.87** ±1.79 (-145.0)
PCD-1-0.05	**-142.19** ±1.62 (-143.2)	**-141.88** ±1.28 (-142.2)	**-141.98** ±1.96 (-143.2)
PCD-1-0.01	**-140.58** ±0.91 (-140.7)	**-140.40** ±0.66 (-140.7)	**-140.52** ±0.55 (-140.7)
PT_{10}-0.01	-143.68 ±1.53 (-153.0)	**-141.95** ±1.22 (-142.5)	-143.45 ±0.76 (-145.8)

Table 7: Maximum average LL on the training data for RBMs with 4 hidden units on (top) *Bars & Stripes*, and (middle) *Flipped Shifting Bar*, and (bottom) RBMs with 16 hidden units on *MNIST* when using an exponentially moving average with a sliding factor of 0.01.

if the training data and thus $\langle \mathbf{x} \rangle_d$ is known in advance the visible offsets should be fixed to this value independent of whether batch or mini-batch learning is used. However, the use of an exponentially moving average for approximating $\langle \mathbf{x} \rangle_d$ is reasonable if the training data is not known in advance, as well as for the approximation of the mean of the hidden representation $\langle \mathbf{h} \rangle_d$. We therefore used an exponentially moving average for the following experiments on the big data sets.

6.6 Control Experiments: Other Choices for the Offsets

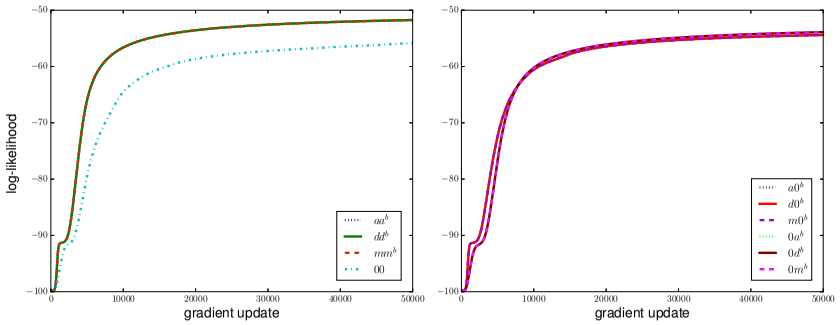
As discussed in Section 3, any offset value between 0 and 1 guarantees the flip invariance property as long as it is flipped simultaneously with the data. An intuitive and constant choice is to set the offsets to 0.5, which has also been proposed by Ollivier et al. (2011) and results in a symmetric variant of the energy of RBMs. This leads to comparable LL values on flipped and unflipped data sets. However, if the data set is unbalanced in the amount of zeros and ones like *MNIST*, we found that the performance is always worse compared to that of a normal RBM on the version of the data set that has fewer ones than zeros. Therefore, fixing the offset values to 0.5 cannot be considered as an alternative for centering using expectation values over the data or model distribution.

In Section 3.1, we mentioned that using either $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_d$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_m$ or $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_m$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_d$ as offsets both lead to the same updates for the weights as the enhanced gradient. Using $\boldsymbol{\mu} = \langle \mathbf{x} \rangle_d$ and $\boldsymbol{\lambda} = \langle \mathbf{h} \rangle_m$ seems reasonable since the data mean is usually known in advance. As mentioned above we refer to centering with this choice of offsets as *dm*. We trained RBMs with dm_s^b using a sliding factor of 0.01. The results are shown in Table 7 and suggest that there is no significant difference between dm_s^b, aa_s^b, and dd_s^b. However, without an exponentially moving average dm^b has the same divergence problems as aa^b when PT_c is used for sampling, as shown in Figure 7(b).

We further tried variants like *mm*, *m*0, 0*d*, *m*0, etc., but did not find better performance than that of *dd*. The variants that subtract an offset from both, visible and hidden variables outperformed or achieved the same performance as the variants that only subtract an offset from either visible or hidden variables. When the model expectation was used without an exponentially moving average either for $\boldsymbol{\mu}$ or $\boldsymbol{\lambda}$, or for both offsets we always observed the divergence when PT_c was used for sampling (results not shown).

Interestingly, if the exact gradient and offsets are used for training, no significant difference can be observed in terms of the LL evolution whether data mean, model mean or the average of both is used for the offsets, as shown in Figure 8. But centering both visible and hidden units still leads to better results than centering only one. Furthermore, the results illustrate that centered RBMs outperform normal binary RBMs also if the exact gradient is used for training both models. This further emphasizes that the worse performance of normal binary RBMs is caused by the properties of the gradient rather than by the gradient approximation.

The control experiments confirm our findings that centering both visible and hidden units is important and that the performance of *dd*, *aa*, and *mm* become similar when an exponentially moving average is used.



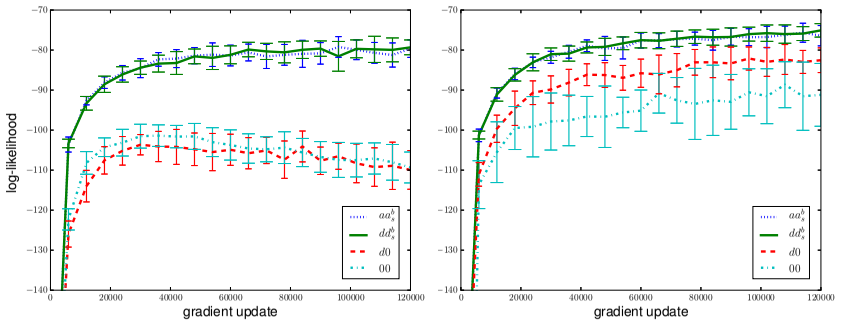
(a) Visible and hidden units centered (b) Visible or hidden units centered

Figure 8: Evolution of the average LL on the training data for RBMs with 4 hidden units on the *Bars & Stripes* data set for the various centering methods when the exact gradient is used with the exact offsets and a learning rate of $\eta = 0.05$. The plots for aa^b, dd^b and mm^b overlay each other and so do $a0^b$, $d0^b$, $m0^b$, and also $0a^b$, $0d^b$ and $0m^b$. In (b) the plots for $a0^b$, $d0^b$, $m0^b$ can be distinguished from the plots for $a0^b$, $d0^b$, $m0^b$ in that they are initially slower and later lower.

6.7 Experiments with Big RBMs

For the experiments in Section 6.1-6.6 we trained small models in order to be able to run many experiments and to evaluate the LL exactly. In this section we want to show that the results we observed on the toy problems and *MNIST* with RBMs having 16 hidden units carry over to more realistic settings. Furthermore, we want to investigate the generalization performance of the different models. In a first set of experiments we therefore trained the models *00*, *d0*, dd_s^b, and aa_s^b with 500 hidden units on *MNIST* and *Caltech*. The weight matrices were initialized with random values sampled from a Gaussian with zero mean and a standard deviation of 0.01, and visible and hidden biases and offsets were initialized as described in Section 4.2. The LL was estimated using Annealed Importance Sampling (AIS), where we used the same setup as described in the analysis of Salakhutdinov and Murray (2008).

Figure 9 shows the evolution of the average LL on the test data of *MNIST* over 25 trials for PCD-1 and PT_{20} for different centering versions. The models were trained for 200 epochs, each consisting of 600 gradient updates with a batch size of 100 and the LL was estimated every 10th epoch using AIS. Both variants dd_s^b and aa_s^b reach significantly higher LL values than *00* and *d0*. Also the standard deviation over the 25 trials indicated by the error bars is smaller for dd_s^b and aa_s^b than for *00* and *d0*, especially when PT_{20} is used for sampling. Furthermore, *00* and *d0* show divergence already after 30000 gradient updates when PCD-1 is used, while no divergence can be observed for dd_s^b and aa_s^b up to 120000



(a) PCD-1 (b) PT_{20}

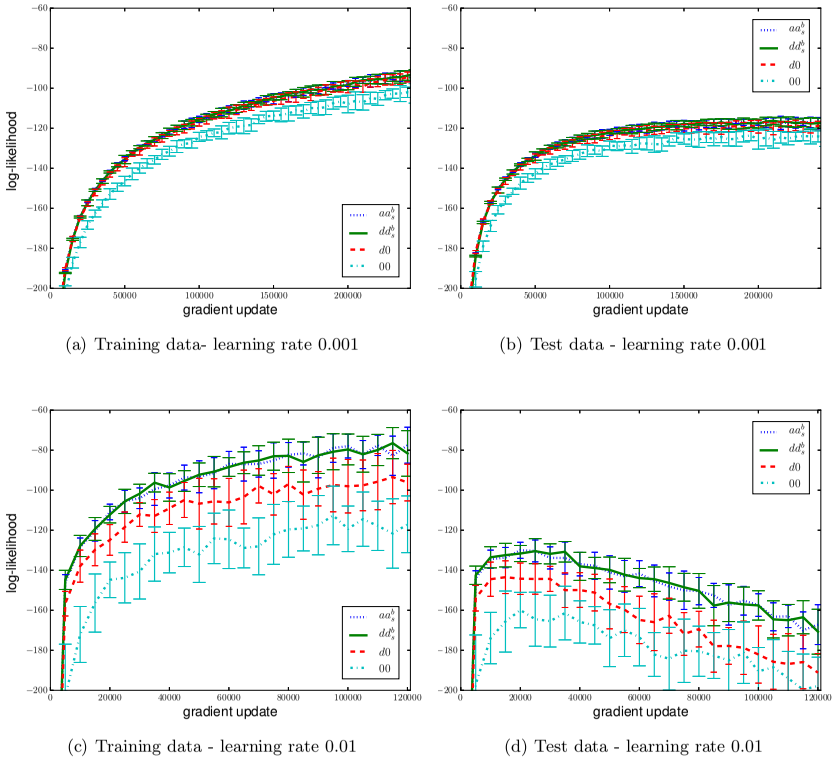
Figure 9: Evolution of the average LL on the test data of *MNIST* during training of different centering variants with 500 hidden units, using a learning rate of $\eta = 0.01$, and a sliding factor of 0.01. (a) When using PCD-1 and (b) when using PT_{20} for training. The error bars indicate the standard deviation of the LL over the 25 trials.

gradient updates. The evolution of the LL on the training data is not shown, since it is almost equivalent to the evolution on the test data. The superiority of centering over normal RBMs can also be observed for the other variants of *MNIST* as shown in Appendix D.

Figure 10 shows the evolution of the average LL on training and test data of the *Caltech* data set over 25 trials for different centering versions using PCD-1 with a batch size of 100 and either a learning rate of 0.001 or 0.01. The LL was estimated every 5000th gradient update using AIS. The results show that dd_s^b, aa_s^b and *d0* reach higher LL values than *00* for both learning rates and on training and test data. While dd_s^b and aa_s^b perform only slightly better than *d0* when a small learning rate is used, the difference becomes more prominent for a big learning rate. Figure 10(c) and (d) show that all models overfit to the training data. Nevertheless, dd_s^b and aa_s^b reach higher LL values on the test data and thus lead to a better generalization. The final average LL on the test data for dd_s^b and aa_s^b shown in Figure 10(b) is around -118. This is consistent with the reported performance of -120.0 for a normal RBM with 500 hidden units trained with PCD-1 (Marlin et al., 2010) and -114.75 when PT sampling is used in combination with the enhanced gradient (Cho et al., 2013b).

In a second set of experiments we extended our analysis to the eight *UCI binary* data sets used by Larochelle et al. (2010). The four different models *00*, *d0*, dd_s^b, and aa_s^b were trained with the same setup as before using PCD-1, a learning rate of 0.01, a batch size of 100 and a total number of 5000 epochs. All experiments were repeated 25 times and we trained either RBMs with 16 hidden units and calculated the LL exactly or RBMs with 200 hidden units using AIS for estimating the LL. Additionally we trained RBMs with 200 hidden variables with a smaller learning rate of 0.001 for 30000 epochs. Due to the long

training time these experiments were repeated only 10 times. The maximum average LL for the test data is shown in Table 8. On seven out of eight data sets dd_s^b or aa_s^b reached the best result independent of whether 16 or 200 hidden units, or a learning rate of 0.01 or 0.001 were used. Whenever aa_s^b reached the highest value it was not significantly different from that of dd_s^b. Note that when training RBMs with 16 hidden units *d0* reached comparable results to



(a) Training data- learning rate 0.001 (b) Test data - learning rate 0.001

(c) Training data - learning rate 0.01 (d) Test data - learning rate 0.01

Figure 10: Evolution of the average LL on *Caltech* data set with different centering variants with 500 hidden units. The results on training and test data for a learning rate of $\eta = 0.001$ are shown in sub-figures (a) and (b), respectively, and for a learning rate of $\eta = 0.01$ in sub-figures (c) and (d), respectively. In both cases a sliding factor of 0.01 and PCD-1 was used. The error bars indicate the standard deviation of the LL over the 25 trials.

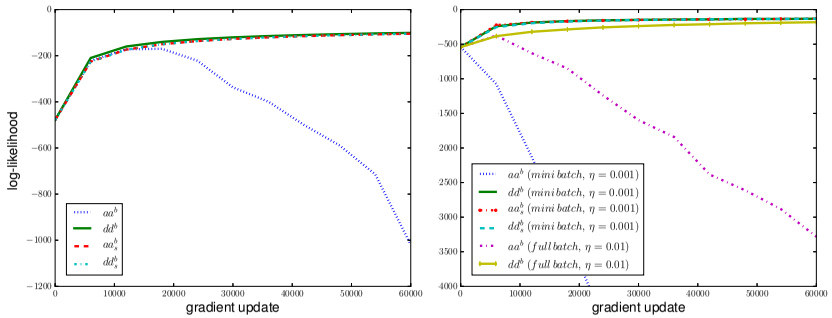
Data set	aa_s^b	dd_s^b	$d0$	00
16 hidden units - learning rate 0.01				
Adult	-18.09 ±0.62 (-18.18)	**-17.70** ±0.25 (-17.74)	-17.90 ±0.21 (-17.97)	-17.94 ±0.22 (-17.98)
CONNECT-4	-20.07 ±0.19 (-20.52)	**-19.89** ±0.21 (-19.90)	-20.14 ±0.24 (-20.16)	-20.59 ±0.29 (-20.74)
DNA	**-96.97** ±0.05 (-97.26)	**-96.97** ±0.04 (-97.25)	-97.01 ±0.04 (-97.21)	-97.03 ±0.05 (-97.19)
Mushroom	-16.83 ±0.57 (-17.11)	**-16.53** ±0.64 (-16.53)	**-16.76** ±0.56 (-17.11)	-17.05 ±0.59 (-17.44)
NIPS	**-276.37** ±0.16 (-279.13)	**-276.38** ±0.16 (-279.11)	-276.53 ±0.21 (-278.97)	-278.04 ±0.27 (-279.95)
OCR	**-45.81** ±0.13 (-45.81)	**-45.82** ±0.12 (-45.83)	**-45.84** ±0.12 (-45.86)	-45.97 ±0.17 (-45.99)
RCV1	-49.57 ±0.04 (-49.57)	-49.58 ±0.05 (-49.58)	-49.59 ±0.05 (-49.59)	**-49.53** ±0.04 (-49.53)
Web	**-29.99** ±0.05 (-29.99)	**-29.98** ±0.05 (-29.98)	**-30.20** ±0.70 (-30.86)	-30.75 ±0.11 (-38.82)
200 hidden units - learning rate 0.01				
Adult	-15.98 ±0.37 (-17.80)	**-15.55** ±0.25 (-16.22)	-16.65 ±0.61 (-18.96)	-16.91 ±0.78 (-19.24)
CONNECT-4	-14.85 ±0.24 (-23.31)	**-14.70** ±0.20 (-17.71)	-16.14 ±0.50 (-23.14)	-17.88 ±0.78 (-30.41)
DNA	-90.15 ±0.09 (-95.17)	**-90.12** ±0.10 (-94.13)	-90.78 ±0.10 (-95.93)	-91.13 ±0.12 (-97.62)
Mushroom	**-15.45** ±1.35 (-20.11)	**-15.61** ±1.23 (-19.93)	-16.28 ±0.88 (-21.59)	-16.42 ±1.61 (-21.90)
NIPS	**-270.81** ±0.05 (-291.82)	**-270.81** ±0.07 (-291.77)	-271.38 ±0.28 (-294.28)	-272.88 ±0.44 (-290.02)
OCR	**-29.75** ±0.50 (-30.18)	**-29.53** ±0.52 (-29.97)	-30.66 ±0.65 (-30.66)	-30.08 ±0.52 (-30.25)
RCV1	-47.13 ±0.12 (-47.13)	-47.14 ±0.12 (-47.16)	-47.14 ±0.10 (-47.19)	**-46.70** ±0.11 (-46.72)
Web	**-28.27** ±0.19 (-28.47)	**-28.27** ±0.19 (-28.58)	-28.50 ±0.51 (-29.16)	-28.35 ±0.14 (-28.71)
200 hidden units - learning rate 0.001				
Adult	-15.51 ±0.17 (-17.14)	-14.90 ±0.07 (-14.90)	-15.27 ±0.22 (-15.48)	-15.16 ±0.21 (-15.39)
CONNECT-4	-13.73 ±0.17 (-15.95)	-13.13 ±0.10 (-13.25)	-13.86 ±0.29 (-14.79)	-14.47 ±0.38 (-16.04)
DNA	-90.17 ±0.07 (-90.17)	-90.15 ±0.07 (-90.15)	-90.45 ±0.09 (-90.45)	-90.73 ±0.13 (-90.73)
Mushroom	-13.44 ±0.75 (-13.96)	-13.17 ±1.15 (-13.94)	-13.47 ±0.73 (-14.51)	-13.68 ±0.77 (-14.77)
NIPS	-270.71 ±0.03 (-313.30)	-270.70 ±0.03 (-313.16)	-270.84 ±0.04 (-315.96)	-271.91 ±0.12 (-309.03)
OCR	-27.92 ±0.34 (-27.92)	-27.70 ±0.15 (-27.75)	-28.24 ±0.21 (-28.34)	-28.04 ±0.39 (-28.29)
RCV1	-47.09 ±0.05 (-47.09)	-47.10 ±0.03 (-47.11)	-46.93 ±0.07 (-46.93)	-46.56 ±0.07 (-46.56)
Web	-28.13 ±0.02 (-28.17)	-28.11 ±0.06 (-28.11)	-28.16 ±0.05 (-28.20)	-28.13 ±0.05 (-28.15)

Table 8: Maximum average LL for the test data of the eight *UCI binary* data sets during training RBMs with (top) 16 hidden units and a learning rate of 0.01, (middle) 200 hidden units and a learning rate of 0.01 and (bottom) 200 hidden units and learning rate of 0.001 (since 10 trials are are not enough to perform a statistical significance test we simply underlined the best result). All models were trained using PCD-1 with a batch size of 100.

Data set	aa_s^b	dd_s^b	$d0$	00
16 hidden units - learning rate 0.01				
Adult	-17.93 ±0.61 (-18.01)	**-17.54** ±0.25 (-17.58)	-17.73 ±0.20 (-17.80)	-17.78 ±0.23 (-17.81)
CONNECT-4	-19.94 ±0.19 (-20.38)	**-19.75** ±0.25 (-19.77)	-20.01 ±0.24 (-20.03)	-20.46 ±0.28 (-20.61)
DNA	**-94.33** ±0.05 (-94.33)	**-94.33** ±0.06 (-94.33)	-94.44 ±0.04 (-94.44)	-94.42 ±0.05 (-94.42)
Mushroom	-16.55 ±0.56 (-16.82)	**-16.25** ±0.64 (-16.25)	**-16.46** ±0.56 (-16.80)	-16.75 ±0.60 (-17.15)
NIPS	**-255.03** ±0.25 (-255.05)	**-255.02** ±0.23 (-255.04)	-255.88 ±0.28 (-255.88)	-258.57 ±0.29 (-258.57)
OCR	**-45.88** ±0.12 (-45.88)	**-45.90** ±0.11 (-45.90)	**-45.90** ±0.12 (-45.93)	-46.03 ±0.16 (-46.05)
RCV1	-49.42 ±0.04 (-49.42)	-49.43 ±0.05 (-49.43)	-49.44 ±0.05 (-49.44)	**-49.38** ±0.04 (-49.38)
Web	**-29.79** ±0.05 (-29.79)	**-29.78** ±0.05 (-29.78)	**-29.97** ±0.70 (-30.62)	-30.64 ±0.11 (-38.61)
200 hidden units - learning rate 0.01				
Adult	-15.19 ±0.62 (-16.29)	**-14.51** ±0.43 (-14.80)	-15.83 ±0.61 (-17.17)	-16.00 ±0.79 (-17.51)
CONNECT-4	-14.39 ±0.24 (-22.50)	**-14.22** ±0.20 (-17.02)	-15.74 ±0.49 (-22.32)	-17.58 ±0.76 (-29.55)
DNA	**-60.19** ±1.20 (-60.69)	**-59.75** ±1.45 (-59.75)	**-60.31** ±1.64 (-60.31)	-61.11 ±1.54 (-61.71)
Mushroom	**-14.90** ±1.34 (-19.14)	**-15.06** ±2.09 (-19.00)	**-15.70** ±2.88 (-20.52)	-15.87 ±1.63 (-20.87)
NIPS	-180.48 ±0.24 (-180.48)	-180.51 ±0.31 (-180.51)	**-180.29** ±0.54 (-180.29)	-184.84 ±0.62 (-184.84)
OCR	**-28.97** ±0.49 (-29.39)	**-28.76** ±0.52 (-29.20)	-29.84 ±0.63 (-29.84)	-29.34 ±0.51 (-29.49)
RCV1	-45.83 ±0.12 (-45.83)	-45.85 ±0.11 (-45.87)	-45.92 ±0.18 (-45.97)	**-45.60** ±0.11 (-45.61)
Web	**-26.22** ±0.67 (-26.22)	-26.28 ±0.20 (-26.32)	-26.57 ±0.66 (-26.75)	-26.34 ±0.38 (-26.39)
200 hidden units - learning rate 0.001				
Adult	-14.61 ±0.52 (-15.96)	-13.85 ±0.06 (-13.85)	-14.07 ±0.22 (-14.17)	-14.03 ±0.21 (-14.11)
CONNECT-4	-13.15 ±0.17 (-15.20)	-12.54 ±0.11 (-12.62)	-13.28 ±0.29 (-14.10)	-13.98 ±0.38 (-15.39)
DNA	-70.22 ±0.08 (-70.22)	-70.24 ±0.08 (-70.24)	-69.66 ±0.06 (-69.66)	-69.40 ±0.08 (-69.40)
Mushroom	-12.76 ±0.75 (-13.21)	-12.43 ±1.18 (-13.21)	-12.69 ±0.72 (-13.68)	-13.05 ±0.76 (-13.92)
NIPS	-148.14 ±0.30 (-148.14)	-148.30 ±0.31 (-148.30)	-147.33 ±0.30 (-147.33)	-150.91 ±0.30 (-150.91)
OCR	-27.06 ±0.33 (-27.06)	-26.90 ±0.14 (-26.95)	-27.41 ±0.20 (-27.52)	-27.28 ±0.39 (-27.53)
RCV1	-45.83 ±0.04 (-45.83)	-45.85 ±0.05 (-45.85)	-45.73 ±0.07 (-45.73)	-45.47 ±0.06 (-45.47)
Web	-26.44 ±0.06 (-26.45)	-26.36 ±0.07 (-26.36)	-26.32 ±0.05 (-26.33)	-26.35 ±0.10 (-26.35)

Table 9: Maximum average LL for the training data on the eight *UCI binary* data sets during training RBMs, with (top) 16 hidden units and a learning rate of 0.01, (middle) 200 hidden units and a learning rate of 0.01 and (bottom) 200 hidden units and a learning rate of 0.001 (since 10 trials are are not enough to perform a statistical significance test we simply underlined the best result). All models were trained using PCD-1 with a batch size of 100

.



(a) MNIST LL on test data

(b) Caltech LL on test data

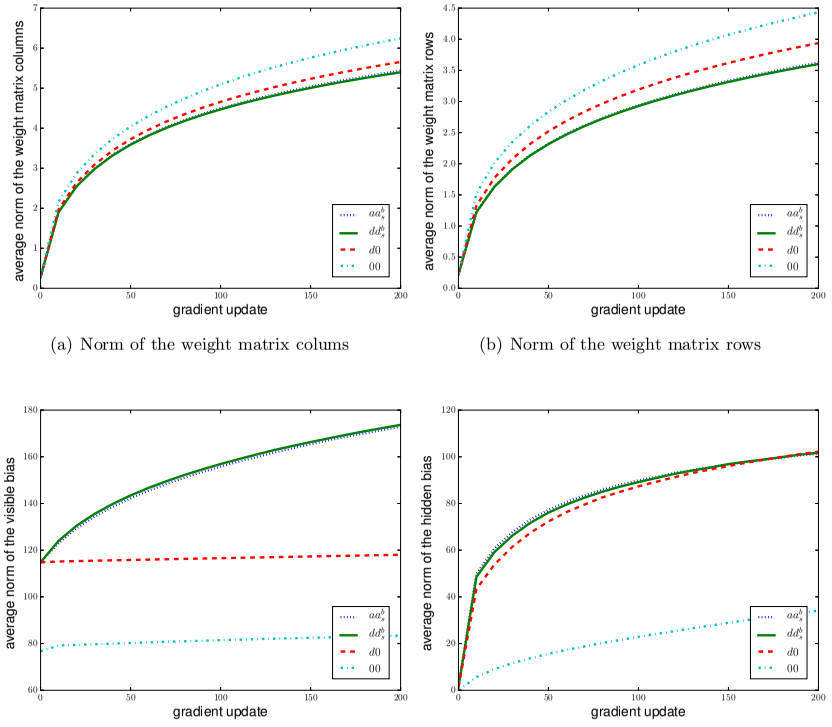
Figure 11: Evolution of the LL of exemplary trials for the different centering variants aa^b, dd^b, aa^b_s and dd^b_s on (a) *MNIST* and (b) *Caltech* during training using PT_{20} with a batch size of 100, 500 hidden units and a learning rate of 0.001. For *Caltech* aa^b and dd^b were also trained in full batch mode with PT_{20} and a learning rate of 0.01.

dd^b_s on some data sets. Only on the *RCV1* data set, 00 lead to better LL values than the centered RBMs for both 16 and 200 hidden units. It seems that the convergence rate on the *RCV1*, *OCR* and *WEB* data set is rather low for all models since the difference between the highest and the final LL values is rather small, indicating that no divergence or overfitting has happened so far. This can also be observed on the training data shown in Table 9. The *DNA* data set and the *NIPS* data set overfitted to the training data as indicated by the fact that the LL only decreased for the test data. In contrast, on the remaining three data sets *ADULT*, *CONNECT-4* and *MUSHROOM* the divergence can be observed on training and test data. Finally note that all eight data sets contain more zeros than ones in the current representation as mentioned in Section 5.1. Thus, the performance of the normal RBM would be even worse on the flipped data sets while for the centering variants it would stay the same (results not shown). Consistent with the experiments on small models, the results from nine of the ten real world data sets clearly support the superiority of centered over normal RBMs, show that centering visible and hidden units in RBMs is important for yielding good models and that averaged over all data sets dd^b_s performs better than aa^b_s.

To emphasize that the divergence problems induced by using the model means as offsets also occurs for big models when no moving average is used, we trained RBMs with 500 hidden units on *MNIST* and *Caltech* using PT_{20} with a learning rate of 0.001 and a batch size of 100. In addition, we trained aa^b and dd^b on *Caltech* using full batch learning and a learning rate of 0.01. Figure 11 shows that aa^b diverges while dd^b and the corresponding centering versions using a moving average, aa^b_s and dd^b_s, show no divergence. The divergence for aa^b even occurs in full batch training as shown in Figure 11(b).

6.8 Investigating the Effect of Centering on the Model Parameters

One explanation why centering works has already been given by Montavon and Müller (2012), who found that centering leads to a better conditioned optimization problem. Furthermore, Cho et al. (2011) have shown that when the enhanced gradient is used for training the update directions for the weights are less correlated than when the standard gradient is used, which allows to learn more meaningful features. From our analysis in Section 3 we know that centered RBMs and normal RBMs belong to the same model class and therefore the reason why centered RBMs outperform normal RBMs can indeed only be due to the optimization procedure. Furthermore, one has to keep in mind that in centered RBMs



(a) Norm of the weight matrix colums

(b) Norm of the weight matrix rows

(c) Norm of the visible bias vector

(d) Norm of the hidden bias vector

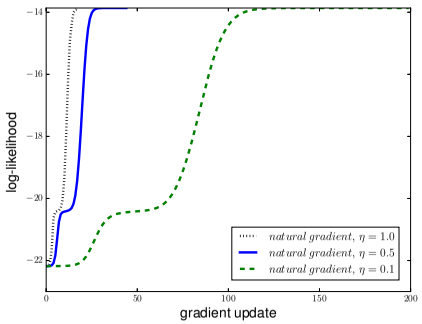
Figure 12: Evolution of the average Euclidean norm of the parameters of the RBMs with 500 hidden units trained on *MNIST*.

the variables mean values are explicitly stored in the corresponding offset parameters, or if the centered gradient is used for training normal RBMs the mean values are transferred to the corresponding bias parameters. This allows the weights to model second and higher order statistics right from the start, which is in contrast to normal binary RBMs where weights usually capture parts of the mean values. To support this statement empirically, we calculated the average weight and bias norms during training of the RBMs with 500 hidden units on *MNIST* using the standard and the centered gradient. The results are shown in Figure 12, where it can be seen that the row and column norms (see Figure 12a and 12b) of the weight matrix for dd_s^b, aa_s^b, and *d0* are consistently smaller than for *00*. At the same time the bias values (see Figure 12c and 12d) for dd_s^b, aa_s^b, and *d0* are much bigger than for *00*, indicating that the weight vectors of *00* model information that could potentially be modeled by the bias values. Interestingly, the curves for all parameters of dd_s^b and aa_s^b show the same logarithmic shape, while for *d0* and *00* the visible bias norm does not change significantly. It seems that the bias values did not adapt properly during training. Comparing, *d0* with dd_s^b and aa_s^b, the weight norms are slightly bigger and the visible bias is much smaller for *d0*, indicating that it is not sufficient to center only the visible variables and that visible and hidden bias influence each other. This dependence between the mean of the hidden variables and the bias of the visible variables can also be seen from Equation (11) where the transformation of the visible bias depends on the offset of the hidden variables.

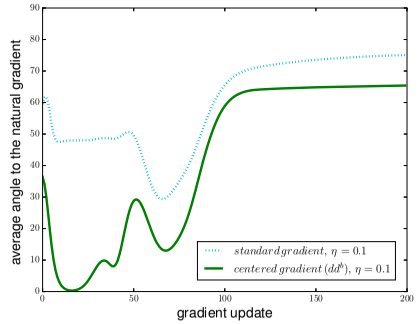
6.9 Comparision with the Natural Gradient

The results in Section 6.7 indicate that one explanation for the better performance of centering and thus the centered gradient compared to the standard gradient is the decoupling of the bias and weight parameters. As described in Section 2.2 the natural gradient is independent of the parameterization of the model distribution. Thus, it is also independent of how the mean information is stored in the parameters and should not suffer from the described bias-weight coupling problem. For the same reason it is also invariant under changes of the representation of the data distribution (for example variable flipping). That is why we expect the direction of the centered gradient to be closer to the direction of the natural gradient than the direction of the standard gradient.

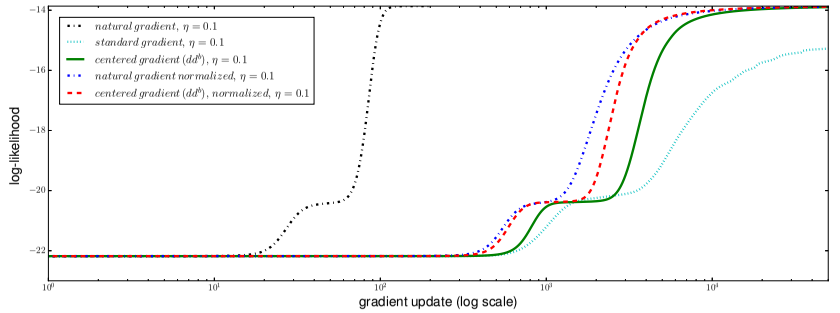
To verify this hypothesis empirically, we trained small RBMs with 4 visible and 4 hidden units using the exact natural gradient on the 2×2 *Bars & Stripes* data set. After each gradient update the different exact gradients were calculated and the angle between the centered and the natural gradient as well as the angle between the standard and the natural gradient were calculated. The results are shown in Figure 13 where Figure 13(a) shows the evolution of the average LL when the exact natural gradient is used for training with different learning rates. Figure 13(b) shows the average angles between the different gradients during training when the natural gradient is used for training with a learning rate of 0.1. The angle between centered and natural gradient is consistently much smaller than the angle between standard and natural gradient. Comparable results can also be observed for the *Shifting Bar* data set and when the standard or centered gradient is used for training (results not shown).



(a) LL evolution of the natural gradient



(b) Angle between standard natural gradient



(c) Comparision of the LL evolution of standard, natural and centered gradient

Figure 13: Comparison of the centered gradient, standard gradient, and natural gradient for RBMs with 4 visible and 4 hidden units trained on *Bars & Stripes* 2×2. (a) The average LL evolution on the training data over 25 trials when the natural gradient is used for training with different learning rates, (b) the average angle over 25 trials between the natural and standard gradient as well as natural and centered gradient when a learning rate of 0.1 is used, and (c) average LL evolution on the training data over 25 trials when either the natural gradient, standard gradient, or centered gradient is used for training.

Note how fast the natural gradient reached a value very close to the theoretical LL upper bound of -13.86 even for a learning rate of 0.1. This verifies empirically the theoretical statement that the natural gradient is clearly the update direction of choice, which should be used if it is tractable. To further emphasize how quickly the natural gradient converges, we compared the average LL evolution for the standard, centered and natural gradient, as shown in Figure 13(c). Although much slower than the natural gradient, the centered gradient reached the theoretical upper bound of the LL. The standard gradient seems to saturate on a much smaller value, showing again the inferiority of the standard gradient even if it is calculated exactly and not only approximated.

To verify that the better performance of natural and centered gradient is not only due to larger gradients resulting in bigger step sizes, we also analyzed the LL evolution for the natural and centered gradient when they are scaled to the norm of the standard gradient before updating the parameters. The results are shown in Figure 13(c). The natural gradient still outperforms the other methods but it becomes significantly slower than when used with its original norm. The reason why the norm of the natural gradient is somehow optimal can be explained by the fact that it ensures constant progress regardless of the curvature of the manifold of probability distributions as explained in (Desjardins et al., 2013). Like a Newton step the Fisher metric results in an automatic local step size adaption such that even a learning rate of 1.0 can be used as illustrated in Figure 13(a).

Interestingly, if the length of the natural gradient and the centered gradient are normalized to the length of the standard gradient and therefore the optimal step size is ignored, the centered gradient becomes almost as fast as the natural gradient. The fact that the normalization of the centered gradient increases the resulting learning speed shows that the norm of the centered gradient is smaller than the norm of the standard gradient. Therefore, the worse performance of the standard gradient does not result from the length but the direction of the gradient.

To conclude, these results support our assumption that the centered gradient is closer to the natural gradient and that it is therefore preferable over the standard gradient (Melchior et al., 2013; Fischer, 2014), which has recently also been confirmed by Grosse and Salakhudinov (2015).

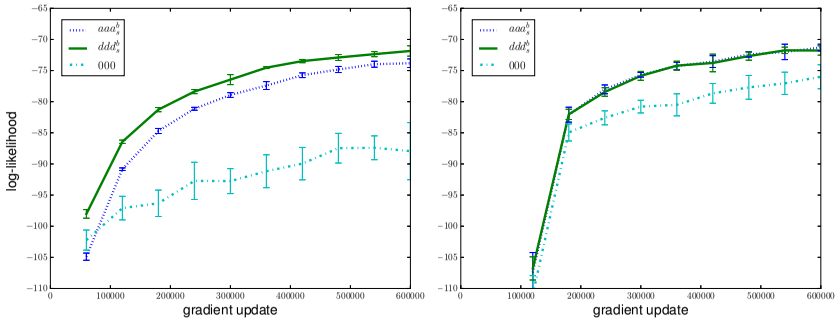
6.10 Experiments with DBMs

When centering was first applied to DBMs by Montavon and Müller (2012) the authors have only seen an improvement via centering for locally connected DBMs. Due to our observations for RBMs and the structural similarity between RBMs and DBMs (a DBM is an RBM with restricted connections and partially unknown input data as discussed in Section 3.2) we expect that the observed benefit of centering also carries over to fully connected DBMs. To verify this assumption and empirically investigate the different centering variants for DBMs we performed extensive experiments on the big data sets listed in Section 5.1.

Training the models and evaluating the lower bound of the LL was performed as originally proposed for normal DBMs by Salakhutdinov and Hinton (2009). The authors have also proposed to pre-train DBMs in a layer-wise fashion based on RBMs (Hinton and Salakhutdinov, 2012). In our experiments we trained all models with and without pre-training to investigate the effect of pre-training in both normal and centered DBMs. For

pre-training we used the same learning rate and the same offset type as in the final DBM models. Note that we keep using the term "average LL" although it is precisely speaking only the lower bound of the average LL, which has been shown to be rather tied (Salakhutdinov and Hinton, 2009). For the estimation of the partition function we again used AIS where we doubled the number of intermediate temperatures compared to the setting in the RBM experiments to 29.000. We continue using the shorthand notations introduced for RBMs also for DBMs with the only difference that we add a third letter to indicate the offset used in the second hidden layer, such that 000 corresponds to the normal binary DBM, and ddd^b_s and aaa^b_s correspond to the centered DBMs using the data mean and the average of data and model mean as offset, respectively. Due to the large number of experiments and the high computational cost (especially for estimating the LL) the experiments were repeated only 10 times and we focused our analysis on normal DBMs (000) and fully centered DBMs (ddd^b_s, aaa^b_s) only.

Again, we begin our analysis with the *MNIST* data set on which we trained normal and centered DBMs with 500 hidden units in the first and 500 units in the second hidden layer. Training was done using PCD-1 with a batch size of 100, a learning rate of 0.001 and in case of centering a sliding factor of 0.01 for the extensive amount of 1.000 epochs (600000

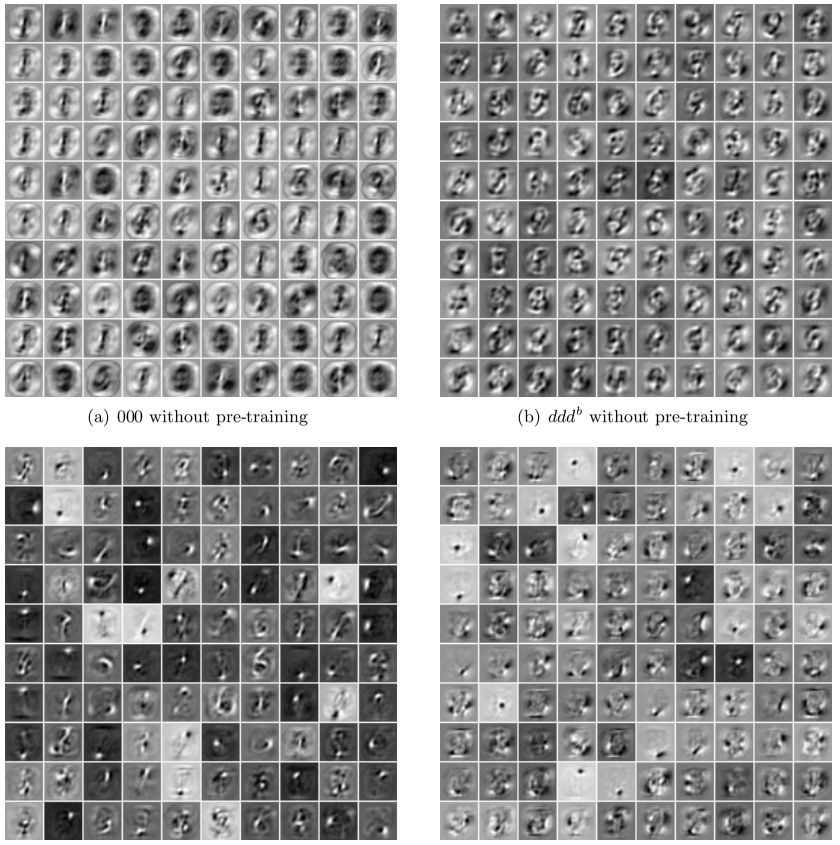


(a) Without pre-training (b) With pre-training

Figure 14: Evolution of the average LL on the test data of the *MNIST* data set for DBMs with 500 hidden units on the first and second layer. The different variants aaa^b_s, ddd^b_s and 000 were either trained (a) without pre-training or (b) with pre-training each DBM layer for 120000 gradient updates (200 epochs). In both cases PCD-1 with a learning rate of $\eta = 0.001$ and for centering a sliding factor of 0.01 was used. The error bars indicate the standard deviation of the average LL over the 10 trials. We skipped evaluating the initial model in (a) and started the LL evaluation in (b) after the 200 epochs of pre-training to roughly account for the computation overhead of pre-training

gradient updates). The evolution of the average LL on the test data without pre-training is shown in Figure 14(a), while the evolution of the average LL on the training data is not shown since it is almost equivalent. Both centered DBMs reach significantly higher LL values with a much smaller standard deviation between the trials than normal DBMs (as indicated by the error bars) and ddd_s^b performs slightly better than aaa_s^b. These findings are different to the observations of Montavon and Müller (2012) who reported an improvement of the model quality in terms of LL through centering only for locally connected DBMs. This might be due to the different training setup (for example a different learning rate, batch size, shorter training time, using ddd_s^l instead of ddd_s^b, or approximation of the data dependent part of the LL gradient by Gibbs sampling instead of optimizing the lower bound of the LL). Figure 14(b) shows the evolution of the average LL on the test data for the same models with pre-training for 120000 gradient updates (200 epochs). The evolution of the average LL on the training data was again almost equivalent and is thus not shown. It can be seen that ddd_s^b has approximately the same performance with and without pre-training but the performance of aaa_s^b improves by pre-training such that a similar LL level as for ddd_s^b is reached. Pre-training allows 000 to reach better LL than without pre-training, however it is still significantly worse compared to the centered DBMs with or without pre-training. By comparing these results with the results for RBMs with 500 hidden units trained on *MNIST* shown in Figure 9(a) we see that all DBMs reach higher LL values than the corresponding RBM models.

The higher layer representations in DBMs highly depend on the data driven lower layer representations. Therefore, we expect to see a qualitative difference between the second layer receptive fields or filters, given by the columns of the weight matrices in centered and normal DBMs. We did not visualize the filters of the first layer since all models showed the well known stroke like structure, which can be seen for RBMs in the review paper by Fischer and Igel (2014) for example. We visualized the receptive fields of the second layer by the linear back-projection of the second layer filters into the input space given by the matrix product of first and second layer weight matrix. The corresponding back projected second layer filters of 000 and ddd_s^b are shown Figure 15(a) and (b), respectively. It can be seen that many second layer filters of 000 are roughly the same and thus highly correlated. Moreover, they seem to represent some kind of mean information. Whereas the filters for ddd_s^b have much more diverse and less correlated structures than the filters of the normal DBM. When pre-training is used the filters of 000 become more diverse and the filters of both 000 and ddd_s^b become more selective as can be seen in Figure 15(c) and (d), respectively. The differences in selectivity of the filters between the different DBM variants can also be seen from the average mean field activation of the variables in the second hidden layer. As shown in Figure 16(a) without pre-training the average activation given the training data is approximately 0.5 for all hidden units of ddd_s^b while for aaa_s^b it is a bit less balanced and for 000 the units tend to be either active or inactive all the time. The results in Figure 16(b) illustrate that the average activities for all models become less balanced when pre-training is used, which also reflects the higher selectivity of the filters as shown in Figure 15(c) and (d). While the second layer hidden activities of ddd_s^b and aaa_s^b stay in a reasonable range, they become extremely selective for 000 where 300 out of 500 units are inactive most of the time. Therefore, the filters, the average activation and the evolution of the LL indicate that normal RBMs have difficulties in making use of the second hidden layer.



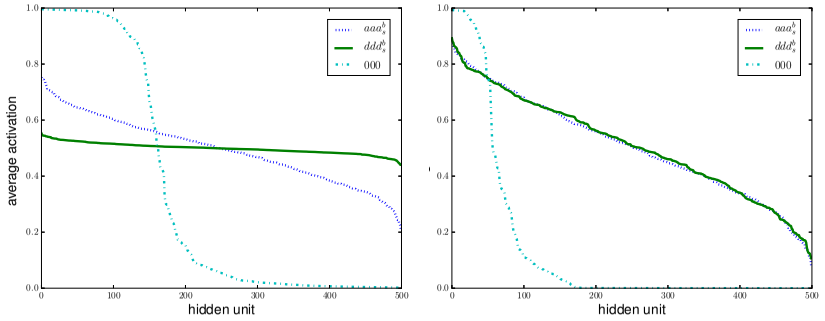
(a) 000 without pre-training

(b) ddd^b without pre-training

(c) 000 with pre-training

(d) ddd^b with pre-training

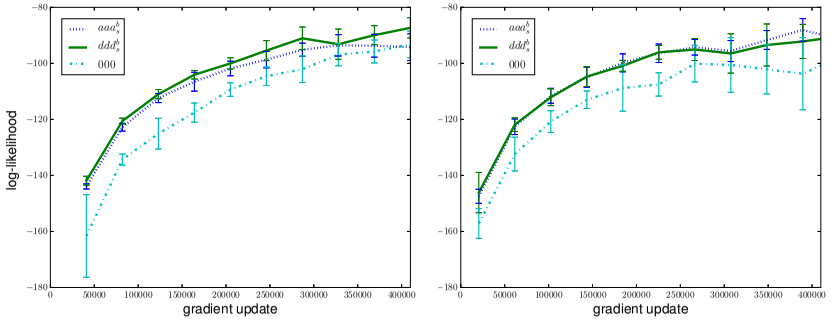
Figure 15: Random selection of 100 linearly projected filters of the second hidden layer for (a) 000 and (b) ddd^b without pre-training and (c) 000 and (d) ddd^b without 200 epochs pre-training. The filters have been normalized independently such that the structure can be seen better.



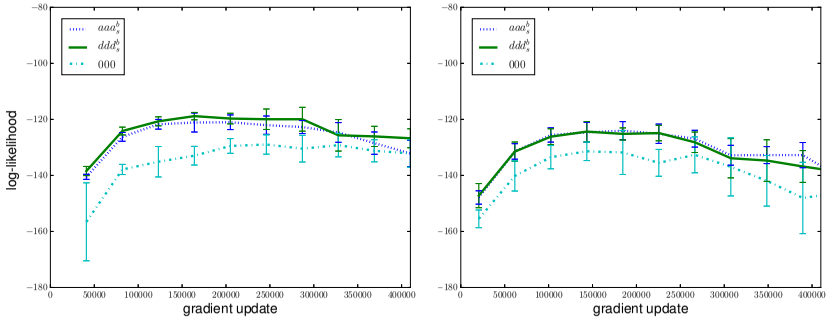
(a) Without pre-training (b) With pre-training

Figure 16: Ordered average mean field activation of the 500 hidden units in the second layer of the different DBMs given the training data (a) without pre-training and (b) with pre-training. Sampling the states or using the conditional probabilities of the variables instead of taking the mean field activation leads to almost equivalent results.

We continue our analysis with experiments on the *Caltech* data set on which we again trained normal and centered DBMs with 500 hidden units on the first and 500 hidden units on the second hidden layer. Training was also done using PCD-1 with a batch size of 100, a learning rate of 0.001 and in case of centering a sliding factor of 0.01. Since the training data has only 41 batches the models were trained for the extensive amount of 10000 epochs (410000 gradient updates). Figure 17 shows the average LL on the test data (c) without and (d) with 500 epochs pre-training. In addition, Figure 17(a) and (b) show the corresponding average LL on the training data demonstrating that all models overfitted to the training data of the *Caltech* data set. The results are consistent with the results on *MNIST*. 000 performs worse than centering on training and test data independently of whether pre-training is used or not. Furthermore, aaa_s^b seems to perform slightly worse than ddd_s^b without pre-training, while the performance becomes equivalent if pre-training is used. But in contrast to the results for *MNIST*, on the *Caltech* data set all methods perform worse with pre-training. This negative effect of pre-training becomes even worse when the number of pre-training epochs is increased. In the case of 2.000 epochs of pre-training for example, ddd_s^b and aaa_s^b still perform better than 000 but the maximal average LL among all models, which was reached by ddd_s^b was only -98.1 for the training data and -141.4 for the test data, compared to -90.4 and -124.0 when 500 epochs of pre-training were used, and -87.3 and -118.8 when no pre-training was used. Without pre-training the LL values are comparable to the results when an RBM with 500 hidden units is trained on *Caltech* as shown in Figure 10, illustrating that in terms of the LL a DBM does not necessarily perform better than an RBM. We also examined the filters and the average hidden activities for the



(a) LL on the training data without pre-training (b) LL on the training data with pre-training



(c) LL on the test data without pre-training (d) LL on the test data with pre-training

Figure 17: Evolution of the average LL on the *Caltech* data set for different DBMs (aaa_s^b, ddd_s^b, and 000) with 500 units in the first and 500 units in second hidden layer. LL on (a) the training data and (c) the test data without pre-training and the LL on (b) the training data and (d) the test data with 500 epochs (20500 gradients updates) of pre-training. The models were trained using PCD-1 with a batch size of 100, a sliding factor of 0.01 and a learning rate of $\eta = 0.001$. The error bars indicate the standard deviation of the LL over the 10 trials. We skipped evaluating the initial model and (a) and (c) start after the 500 epochs of pre-training to roughly account for the computation overhead of pre-training.

training data of *Caltech*. Both led to the same conclusions as the results for *MNIST* and are therefore not shown here.

Finally, we also performed experiments on the eight *UCI binary* data sets described in Section 5.1 using the same training setup as for the corresponding RBM experiments. That is, the DBMs with 200 hidden units on the first and 200 hidden units on the second hidden layer were trained for 5.000 epochs with PCD-1, a batch size of 100, a learning rate of 0.01, and in the case of centering a sliding factor of 0.01. The LL was evaluated every 50th epoch and in the case of pre-training the models were pre-trained for 200 epochs. Table 10 shows the maximum average LL on the test data with and without pre-training. Without pre-training the results are consistent with the findings for RBMs. ddd^b performs better than 000 on all data sets except for RCV1 where 000 performs slightly better. The LL values for the DBMs are comparable but not necessarily better than the corresponding LL values for RBMs, which are shown in Table 8. In the case of the *WEB* data set for example the DBMs even perform worse than the RBMs . When pre-training is used the performance of all models, centered or normal, is worse than the performance of the corresponding DBMs without pre-training. For completeness Table 11 shows the maximum average LL on the training data leading to the same conclusions as the results on test data.

Data set	ddd_s^b			000		
No pre-training						
adult	-15.54	±0.42	(-17.12)	-18.44	±1.15	(-24.92)
connect4	-15.09	±0.39	(-40.83)	-18.15	±1.14	(-43.84)
dna	-89.81	±0.13	(-92.57)	-91.18	±0.17	(-95.12)
mushrooms	-15.24	±0.60	(-19.68)	-17.21	±1.16	(-27.71)
nips	-270.35	±0.09	(-360.59)	-275.43	±1.81	(-360.56)
ocr letters	-30.37	±0.39	(-32.23)	-31.56	±0.93	(-32.74)
rcv1	-46.83	±0.08	(-47.26)	-46.51	±0.56	(-47.88)
web	-30.02	±0.59	(-72.88)	-30.35	±1.19	(-79.36)
With pre-training						
adult	-18.86	±2.74	(-21.43)	-21.64	±1.64	(-40.42)
connect4	-27.38	±1.52	(-32.13)	-41.21	±4.07	(-52.04)
dna	-89.87	±0.11	(-94.03)	-91.06	±0.19	(-97.48)
mushrooms	-24.23	±5.43	(-35.07)	-21.42	±6.32	(-35.82)
nips	-272.92	±0.16	(-404.11)	-276.88	±2.33	(-378.76)
ocr letters	-36.89	±1.44	(-39.76)	-32.25	±1.18	(-35.01)
rcv1	-47.79	±0.84	(-49.30)	-46.90	±0.36	(-48.36)
web	-31.10	±0.14	(-81.93)	-32.43	±1.46	(-47.73)

Table 10: Maximum average LL on test data of the eight *UCI binary* data sets for DBMs with 200 units in the first and second hidden layer. For training (top) without pre-training and (bottom) with 200 epochs of pre-training. PCD-1 with a learning rate of 0.01 and a batch size of 100 was used. (The best result is underlined.)

Data set	ddd_s^b			000		
No pre-training						
adult	-14.65	±0.37	(-15.49)	-17.88	±1.16	(-23.04)
connect4	-14.68	±0.38	(-39.74)	-17.82	±0.63	(-42.78)
dna	-62.00	±1.11	(-62.42)	-62.48	±0.17	(-62.98)
mushrooms	-14.74	±1.54	(-18.61)	-16.62	±1.13	(-26.53)
nips	-107.09	±2.11	(-107.09)	-114.28	±2.81	(-114.28)
ocr letters	-29.15	±0.67	(-30.91)	-30.41	±0.90	(-31.57)
rcv1	-45.75	±0.07	(-46.18)	-45.80	±0.82	(-47.17)
web	-29.37	±0.59	(-69.70)	-29.68	±1.18	(-77.11)
With pre-training						
adult	-17.11	±2.70	(-19.66)	-21.42	±1.63	(-38.21)
connect4	-26.62	±6.66	(-31.14)	-40.53	±4.13	(-51.29)
dna	-59.97	±0.49	(-60.37)	-61.16	±1.57	(-61.72)
mushrooms	-23.89	±5.27	(-34.11)	-20.59	±6.33	(-34.86)
nips	-114.51	±3.76	(-118.29)	-118.90	±3.68	(-120.94)
ocr letters	-35.39	±1.88	(-37.70)	-30.90	±1.14	(-33.21)
rcv1	-46.54	±0.87	(-48.07)	-45.93	±0.36	(-47.39)
web	-30.41	±0.15	(-78.45)	-31.68	±1.45	(-44.42)

Table 11: Maximum average LL on training data on the eight *UCI binary* data sets for DBMs with 200 hidden units on the first and second layer. For training (top) without pre-training and (bottom) with 200 epochs pre-training PCD-1, with a learning rate of 0.01, batch size of 100 was used. (The best result is underlined).

To summarize, the experiments described in this section show that centered DBMs reach higher LL values than normal DBMs. While pre-training leads to more selective filters in general, it is often even harmful for the model performance in terms of the LL.

6.11 Experiments with Auto Encoders

The benefit of centering in feed forward neural networks for supervised tasks has already been shown by Schraudolph (1998). In this section we analyze centering in a special kind of unsupervised feed forward neural networks, namely centered AEs as introduced in Section 4.1. We therefore trained normal and centered three layer AEs on the ten big data sets described in Section 5.1. To avoid trivial solutions we used tied weights and the number of output dimensions was 50 percent of the number of input/data dimensions. Since the data sets are binary we used the sigmoid non-linearity in encoder and decoder and the cross entropy cost function. Training was done using standard back propagation for 5000 epochs without any further modification. As for the RBM and DBM experiments, the weight matrices were initialized with random values sampled from a Gaussian with zero mean and standard deviation 0.01, and the biases and offsets were initialized as described in Section 4.2. The batch size was 100, the sliding factor 0.01 and we used a default learning rate

Data set - Learning rate	dd_s^b			00		
Test data						
mnist - 0.1	50.21472	±0.0256	(5000)	50.24859	±0.0200	(5000)
mnist - 0.5	50.01338	±0.0316	(5000)	56.36068	±0.7578	(22)
caltech - 0.01	44.38403	±0.2257	(2500)	49.21274	±0.2119	(1968)
caltech - 0.1	44.45882	±0.1620	(246)	48.59724	±0.3964	(206)
adult - 0.1	0.38837	±0.0229	(5000)	0.47460	±0.0181	(4676)
adult - 0.5	0.36825	±0.0220	(1526)	0.46086	±0.0155	(884)
connect4 - 0.1	0.03015	±0.0025	(5000)	0.03467	±0.0040	(5000)
connect4 - 0.5	0.02357	±0.0019	(1431)	0.02856	±0.0032	(1349)
dna - 0.01	34.34353	±0.1242	(2161)	34.77299	±0.1948	(2105)
dna - 0.1	34.35117	±0.1245	(216)	34.80547	±0.1823	(210)
mushrooms - 0.1	0.14355	±0.0117	(5000)	0.14226	±0.0081	(5000)
mushrooms - 0.5	0.08555	±0.0167	(3173)	0.09960	±0.0169	(2936)
nips - 0.01	183.21045	±0.6355	(2261)	188.65301	±0.7262	(2107)
nips - 0.1	183.31413	±0.6081	(226)	189.10982	±0.6674	(212)
ocr letters - 0.1	5.41182	±0.1994	(5000)	5.46969	±0.2138	(5000)
ocr letters - 0.5	4.91528	±0.1715	(3703)	5.30343	±0.2737	(1945)
rcv1 - 0.1	12.93456	±0.1562	(5000)	12.55443	±0.3097	(5000)
rcv1 - 0.5	12.32545	±0.3296	(4953)	12.16340	±1.2188	(2946)
web - 0.1	1.07535	±0.0174	(1425)	1.22756	±0.0121	(944)
Training data						
mnist - 0.1	50.03172	±0.0224	(5000)	50.06083	±0.0167	(5000)
mnist - 0.5	49.84428	±0.0253	(5000)	55.89110	±0.7516	(22)
caltech - 0.01	5.71437	±0.0235	(5000)	6.42052	±0.0392	(5000)
caltech - 0.1	0.57507	±0.0030	(5000)	0.62426	±0.0119	(5000)
adult - 0.1	0.04687	±0.0024	(5000)	0.03677	±0.0024	(5000)
adult - 0.5	0.03221	±0.0030	(1526)	0.04053	±0.0051	(894)
connect4 - 0.1	0.01465	±0.0007	(5000)	0.01187	±0.0006	(5000)
connect4 - 0.5	0.01022	±0.0002	(1431)	0.00914	±0.0004	(1349)
dna - 0.01	17.80686	±0.0812	(5000)	18.08306	±0.0665	(5000)
dna - 0.1	11.70726	±0.1018	(5000)	12.08309	±0.0995	(5000)
mushrooms - 0.1	0.06362	±0.0022	(5000)	0.05136	±0.0026	(5000)
mushrooms - 0.5	0.01995	±0.0007	(3173)	0.01768	±0.0013	(2936)
nips - 0.01	21.45218	±0.0614	(5000)	21.94275	±0.0509	(5000)
nips - 0.1	2.05141	±0.0067	(5000)	2.06286	±0.0054	(5000)
ocr letters - 0.1	4.97384	±0.1865	(5000)	5.00803	±0.2078	(5000)
ocr letters - 0.5	4.51704	±0.1642	(3703)	4.88637	±0.2677	(1945)
rcv1 - 0.1	11.95633	±0.1226	(5000)	11.62695	±0.2792	(5000)
rcv1 - 0.5	11.36861	±0.2974	(4953)	11.27664	±1.1136	(2947)
web - 0.1	0.06872	±0.0007	(5000)	0.05554	±0.0003	(5000)

Table 12: Average minimal reached cost value with standard deviation on test and training data of *MNIST*, *Caltech* and the eight*UCI binary* data sets for centered and normal three layer AEs with sigmoid units, cross entropy cost function, and the number of hidden units set to 50 percent of the number of input units (data dimensions). The average number of epochs needed to reach the minimal cost value is given in brackets, and (5000) indicates that convergence was not achieved during training.

of 0.1 for all experiments. Each experiment was repeated 10 times and we calculated the average minimal cost value, the corresponding standard deviation and the average number of epochs needed to reach the minimal cost value. A second set of experiments was performed with a learning rate of 0.5 when the average number of epochs needed to reach the minimal cost value on the test data was close or equal to 5000 epochs, or with a learning rate of 0.01 when the average number of epochs needed to reach the minimal cost value on the test data was less than 500 epochs. The results are given by Table 12, showing that (except for the RCV1 data set) centered AEs perform clearly better in terms of the average minimal cost value on the test data than normal AEs. On the training data normal AEs only perform slightly better on data sets where both models reached very small cost values anyway. We did not show the results for the validation sets since they are almost equivalent to the results for test data.

Interestingly, the result that centering only performs worse on the RCV1 data set is fully consistent with the findings for RBMs and DBMs. We inspected the RCV1 data set and its first and second order statistics but did not find anything conspicuous compared to the other data sets that might have explained why for this particular data set centering is not beneficial. However, learning is much slower for this data set when centering is used, which can also be seen by comparing the results for learning rate 0.1 an 0.5 in Table 12.

7. Conclusion

This work discussed centering in RBMs and DBMs, where centering is done by subtracting offset parameters from visible and hidden variables. Our theoretical analysis has yielded the following results:

1. Centered BMs/RBMs/DBMs and normal BMs/RBMs/DBMs are different parameterizations of the same model class (Section 3), such that any normal BM/RBM/DBM can be transformed to an equivalent centered BM/RBM/DBM an *vice versa*. This equivalence generalizes to ANNs in general, which justifies the use of centering in arbitrary ANNs (Section 4).

2. The LL gradient of centered binary BMs and thus of centered binary RBMs/DBMs is invariant under simultaneous flip of data and offsets, for any offset value in the range of zero to one. This leads to the desired invariance of the LL performance of the model under changes of data representation (Section 3 and Appendix A).

3. Training a centered BM/RBM/DBM can be reformulated as training a normal BM/RBM/DBM with a new parameter update, which we refer to as centered gradient (Section 3.1 and Appendix B).

4. From this new formulation follows that the enhanced gradient is a particular form of centering. That is, the centered gradient becomes equivalent to the enhanced gradient by setting the visible and hidden offsets to the average over model and data mean of the corresponding variable (Section 3.1 and Appendix C).

Our numerical analysis has yielded the following results:

1. Optimal performance of centered binary RBMs/DBMs is achieved when both, visible and hidden variables, are centered and the offsets are set to the expectations of the corresponding variable under data or model distribution (Section 6.1, 6.6 and 6.7).

2. Centered binary RBMs/DBMs reach significantly higher LL values than normal binary RBMs/DBMs (Section 6.7 and 6.10). As an example, centered binary RBMs with 500 hidden units achieved an average LL on the test data of *MNIST* around -75 compared to -88 for normal binary RBMs (Figure 9).

3. Initializing the bias parameters such that the RBM/DBM/AE is initially centered (that is $b_i = \sigma^{-1}(\langle x_i \rangle)$) can already improve the performance of a normal binary RBM. However, this initialization leads to a performance still worse than the performance of a centered binary RBM (Section 6.2) and is therefore no alternative to centering.

4. Using the model expectation (as for the enhanced gradient for example) can lead to a severe divergence of the LL when PCD or PT_c is used for sampling. This is caused by the correlation of offset and gradient approximation (Section 6.4).

5. The divergence can be prevented when an exponentially moving average for the approximations of the offset values is used (Section 6.5), which also stabilizes the training for other centering variants especially when the mini batch size is small.

6. Training centered binary RBMs/DBMs leads to smaller weight norms and larger bias norms compared to normal binary RBMs/DBMs. This supports the hypothesis that when using the standard gradient the mean value is modeled by both weights and biases, while when using the centered gradient the mean values are explicitly modeled by the bias parameters (Section 6.9).

7. The direction of the centered gradient is closer to the natural gradient than that of the standard gradient and the natural gradient is extremely efficient for training RBMs if tractable (Section 6.9).

8. Centered binary DBMs reach higher LL values than normal binary DBMs independently of whether pre-training is used or not. Thus pre-training cannot be considered as an alternative to centering (Section 6.10).

9. While pre-training slightly helps normal binary DBMs on *MNIST* we did not observe an improvement through pre-training for centered binary DBMs. Furthermore, on all data sets other than *MNIST* pre-training led to lower LL values and the results became worse as longer pre-training was performed for normal and centered binary DBMs (Section 6.10).

10. The visual inspection of the learned filters, the average second hidden layer activities and the reached LL values suggest that normal binary DBMs have difficulties in making use of the third and higher layers (Section 6.10).

11. Centering also improves the performance in terms of the optimized loss for AEs, which supports our assumption that centering is beneficial not only for probabilistic models like RBMs and DBMs but also for ANNs in general (Section 6.11).

Based on our results we recommend to center all units in the network using the data mean and to use an exponentially moving average if the mini-batch size is rather small (< 100 for stochastic models and < 10 for deterministic models). Furthermore, we do not recommend to use pre-training for DBMs since it often deteriorates the results.

All results clearly support the superiority of centered RBMs/DBMs and AEs, which we believe will also extend to other models. Future work might focus on centering in BMs and also other probabilistic models such as the neural auto-regressive distribution estimator (Larochelle and Murray, 2011) or in recurrent neural networks such as long short-term memory (Hochreiter and Schmidhuber, 1997). An extension to also normalizing the variance of the units and a comparison to the recently proposed batch normalization (Ioffe and Szegedy, 2015) would also be of interest.

The implementation of the algorithms proposed and analyzed in this work are part of the Python library PyDeep publicly available at `https://github.com/MelJan/PyDeep`.

Acknowledgments

We would like to thank Nan Wang for helpful discussions, Tobias Glasmachers for his suggestions on the natural gradient part, and Alberto Escalante for his valuable corrections. We also like to thank the anonymous reviewers for helpful suggestions. Furthermore, Asja Fischer was supported by the German Federal Ministry of Education and Research within the National Network Computational Neuroscience under grant number 01GQ0951 (Bernstein Fokus Learning behavioral models: From human experiment to technical assistance).

Appendix A. Proof of Invariance for the Centered Gradient

In the following we show that the LL gradient of centered binary BMs and thus the LL gradient for centered binary RBMs and DBMs is invariant to flips of the variables if the corresponding offset parameters flip as well (see Melchior et al., 2013, for a proof specifically for RBMs). Since training a normal binary BM using the centered gradient (see Appendix B) is equivalent to training a centered binary BM, the proof also holds for the centered gradient.

In contrast to an RBM or DBM a BM is a fully connected graphical model and the energy for a centered BM and binary values $\mathbf{x} = (x_1, ..., x_N)$ is given by

$$E(\mathbf{x}) = -\sum_i (x_i - \mu_i)\, b_i - \sum_{i,j>i} (x_i - \mu_i)\, w_{ij}\, (x_j - \mu_j) \quad , \tag{18}$$

and the corresponding LL gradient w.r.t. the parameters w_{ij} and b_i is given by

$$\nabla w_{ij} = \langle (x_i - \mu_i)(x_j - \mu_j) \rangle_d - \langle (x_i - \mu_i)(x_j - \mu_j) \rangle_m \quad , \tag{19}$$

$$\nabla b_i = \langle (x_i - \mu_i) \rangle_d - \langle (x_i - \mu_i) \rangle_m = \langle x_i \rangle_d - \langle x_i \rangle_m \quad . \tag{20}$$

In the following $\boldsymbol{\theta}$ and $\nabla\boldsymbol{\theta}$ is used to jointly denote all parameters w_{ij} and b_i and their derivatives ∇w_{ij} and ∇b_i, respectively .

We begin by formalizing the invariance property for the energy.

Definition 1 *Let there be a binary random variable X_i. The variable X_i' is called 'flipped' or 'flip of X_i' if it takes the values $x_i' = 1 - x_i$ for any given state x_i of X_i.*

Definition 2 *Let there be a centered binary BM with parameters $\boldsymbol{\theta}$, offsets $\boldsymbol{\mu}$, binary random variables $\mathbf{X} = (X_1, \ldots, X_N)$, and energy $E(\mathbf{x})$, and a second centered binary BM with parameters $\tilde{\boldsymbol{\theta}}$, offsets $\tilde{\boldsymbol{\mu}}$, binary random variables $\tilde{\mathbf{X}} = (\tilde{X}_1, \ldots, \tilde{X}_N)$ and energy $\tilde{E}(\tilde{\mathbf{x}})$, where in the latter BM an arbitrary number of variables have been flipped. The energies $E(\mathbf{x})$ and $\tilde{E}(\tilde{\mathbf{x}})$ are called 'flip-invariant' or 'invariant to flips of variables' if $E(\mathbf{x}) = \tilde{E}(\tilde{\mathbf{x}})$ holds.*

Theorem 3 *Let the binary parameter f_i take the value -1 if the corresponding variable X_i, has been flipped and 1 otherwise. The energies $E(\mathbf{x})$ and $\tilde{E}(\tilde{\mathbf{x}})$ are 'flip-invariant' according to Definition 2 if the offsets flip simultaneously with the variables, compactly denoted by*

$$\tilde{x}_i = \frac{1-f_i}{2} + f_i x_i \ , \tag{21}$$

$$\tilde{\mu}_i = \frac{1-f_i}{2} + f_i \mu_i \ , \tag{22}$$

and if the parameters of the models are related in the following way

$$\tilde{w}_{ij} = f_i w_{ij} f_j \ , \tag{23}$$

$$\tilde{b}_i = f_i b_i \ . \tag{24}$$

Proof First note that

$$\begin{aligned} (\tilde{x}_i - \tilde{\mu}_i) &\overset{(21,22)}{=} \left(\frac{1-f_i}{2} + f_i x_i\right) - \left(\frac{1-f_i}{2} + f_i \mu_i\right) \\ &= \frac{1-f_i}{2} + f_i x_i - \frac{1-f_i}{2} - f_i \mu_i \\ &= (x_i - \mu_i) f_i = f_i (x_i - \mu_i) \ , \end{aligned} \tag{25}$$

so that

$$\begin{aligned} \tilde{E}(\tilde{\mathbf{x}}) &\overset{(18)}{=} -\sum_i (\tilde{x}_i - \tilde{\mu}_i)\, \tilde{b}_i - \sum_{i,j>i} (\tilde{x}_i - \tilde{\mu}_i)\, \tilde{w}_{ij}\, (\tilde{x}_j - \tilde{\mu}_j) \\ &\overset{(23,24,25)}{=} -\sum_i (x_i - \mu_i) \underbrace{f_i f_i}_{=1} b_i - \sum_{i,j>i} (x_i - \mu_i) \underbrace{f_i f_i}_{=1} w_{ij} \underbrace{f_j f_j}_{=1} (x_j - \mu_j) \\ &\overset{(18)}{=} E(\mathbf{x}) \ . \end{aligned}$$

■

Definition 4 *Let there be two BMs with 'flip-invariant' energies according to Definition 2. The gradient of the LL ($\nabla\boldsymbol{\theta}$ or $\nabla\tilde{\boldsymbol{\theta}}$ respectively) are called 'flip-invariant' or 'invariant to flips of variables' if $E(\mathbf{x}) = \tilde{E}(\tilde{\mathbf{x}})$ still holds after updating $\boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}}$ to $\boldsymbol{\theta} + \eta\nabla\boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}} + \eta\nabla\tilde{\boldsymbol{\theta}}$, respectively, for any learning rate η.*

We can now state the following theorem for the parameter updates.

Theorem 5 *The gradient of centered binary BMs with flip invariant energies according to Definition 2, is also invariant to flips of arbitrary variables if the corresponding offset parameters flip as well, that is if a flipped variable $\tilde{x}_i = 1 - x_i$ implies $\tilde{\mu}_i = 1 - \mu_i$.*

Proof

$$\begin{aligned} \nabla\tilde{w}_{ij} &\overset{(19)}{=} \langle(\tilde{x}_i - \tilde{\mu}_i)(\tilde{x}_j - \tilde{\mu}_j)\rangle_d - \langle(\tilde{x}_i - \tilde{\mu}_i)(\tilde{x}_j - \tilde{\mu}_j)\rangle_m \\ &\overset{(25)}{=} \langle f_i(x_i - \mu_i)(x_j - \mu_j) f_j\rangle_d - \langle f_i(x_i - \mu_i)(x_j - \mu_j) f_j\rangle_m \\ &\overset{(19)}{=} f_i \nabla w_{ij} f_j \ , \\ \nabla\tilde{b}_i &\overset{(20)}{=} \langle(\tilde{x}_i - \tilde{\mu}_i)\rangle_d - \langle(\tilde{x}_i - \tilde{\mu}_i)\rangle_m \\ &\overset{(25)}{=} \langle f_i(x_i - \mu_i)\rangle_d - \langle f_i(x_i - \mu_i)\rangle_m, \\ &\overset{(20)}{=} f_i \nabla b_i \ . \end{aligned}$$

Comparing these results with Equations (23) – (24) shows that the gradient underlies the same sign changes under variable flips as the parameters. Thus, it holds for the updated parameters $\tilde{\boldsymbol{\theta}} + \eta\nabla\tilde{\boldsymbol{\theta}}$ and $\boldsymbol{\theta} + \eta\nabla\boldsymbol{\theta}$ that

$$\begin{aligned} \tilde{w}_{ij} + \eta\nabla\tilde{w}_{ij} &= f_i\,(w_{ij} + \eta\nabla w_{ij})\, f_j \ , \\ \tilde{b}_i + \eta\nabla\tilde{b}_i &= f_i\,(b_i + \eta\nabla b_i) \ , \end{aligned}$$

showing that $E(\mathbf{x}) = \tilde{E}(\tilde{\mathbf{x}})$ is still guaranteed as follows from Theorem 3 and thus that the gradient of centered RBMs is flip-invariant according to Definition 4. ■

Theorem 3 and Theorem 5 hold for any value from zero to one for μ_i, if it is guaranteed that the offsets flip simultaneously with the corresponding variables. In practice one wants the model to perform equivalently on any flipped version of the data set without knowing which version is presented. This holds if we set the offsets to the expectation value of the corresponding variables under any distribution $p^*(x_i)$, since when $\mu_i = \sum_{x_i} p^*(x_i)\, x_i$, flipping X_i leads to $\tilde{\mu}_i = \sum_{x_i} p^*(x_i)\,(1 - x_i) = 1 - \sum_{x_i} p^*(x_i)\, x_i = 1 - \mu_i$.

Appendix B. Derivation of the Centered Gradient

In the following we show that the gradient of centered BMs can be reformulated as an alternative update for the parameters of a normal binary BM, which we name 'centered gradient'.

We first show that the parameter transformation

$$w_{ij} = \hat{w}_{ij} \ , \tag{26}$$

$$b_i = \hat{b}_i + \sum_{j \neq i} \hat{w}_{ij} \mu_j \ , \tag{27}$$

allows to transform a normal binary BM with energy $\hat{E}(\mathbf{x}) = -\sum_i x_i \hat{b}_i - \sum_{i,j>i} x_i \hat{w}_{ij} x_j$ into a centered binary BM with energy $E(\mathbf{x}) = -\sum_i (x_i - \mu_i)\, b_i - \sum_{i,j>i} (x_i - \mu_i)\, w_{ij}\, (x_j - \mu_j)$

and *vice versa* such that $E(\mathbf{x}) = \hat{E}(\mathbf{x}) + const$ is guaranteed for all $\mathbf{x} \in \{0,1\}^n$ and thus that the modeled distribution stays the same.

$$
\begin{aligned}
E(\mathbf{x}) &= -\sum_i (x_i - \mu_i)\, b_i - \sum_{i,j>i} (x_i - \mu_i)\, w_{ij}\, (x_j - \mu_j) \\
&\overset{(26,27)}{=} -\sum_i (x_i - \mu_i) \left(\hat{b}_i + \sum_{j \neq i} \hat{w}_{ij} \mu_j \right) - \sum_{i,j>i} (x_i - \mu_i)\, \hat{w}_{ij}\, (x_j - \mu_j) \\
&= -\sum_i x_i \hat{b}_i - \sum_i x_i \sum_{j \neq i} \hat{w}_{ij} \mu_j + \sum_i \mu_i \hat{b}_i + \mu_i \sum_{j \neq i} \hat{w}_{ij} \mu_j \\
&\quad - \sum_{i,j>i} x_i \hat{w}_{ij} x_j + \sum_{i,j>i} x_i \hat{w}_{ij} \mu_j + \sum_{i,j>i} \mu_i \hat{w}_{ij} x_j - \sum_{i,j>i} \mu_i \hat{w}_{ij} \mu_j \\
&= \underbrace{-\sum_i x_i \hat{b}_i - \sum_{i,j>i} x_i \hat{w}_{ij} x_j}_{\hat{E}(\mathbf{x})} + \underbrace{\sum_i \mu_i \hat{b}_i + \mu_i \sum_{j \neq i} \hat{w}_{ij} \mu_j - \sum_{i,j>i} \mu_i \hat{w}_{ij} \mu_j}_{const} \\
&\quad - \sum_i x_i \sum_{j \neq i} \hat{w}_{ij} \mu_j + \sum_{i,j>i} x_i \hat{w}_{ij} \mu_j + \sum_{i,j>i} \mu_i \hat{w}_{ij} x_j \\
&= \hat{E}(\mathbf{x}) + const - \sum_i x_i \sum_{j \neq i} \hat{w}_{ij} \mu_j + \sum_{i,j>i} x_i \hat{w}_{ij} \mu_j + \sum_{j,i>j} \mu_j \hat{w}_{ij} x_i \\
&= \hat{E}(\mathbf{x}) + const - \sum_i x_i \sum_{j \neq i} \hat{w}_{ij} \mu_j + \sum_{i,j \neq i} x_i \hat{w}_{ij} \mu_j \\
&= \hat{E}(\mathbf{x}) + const \ .
\end{aligned}
$$

Updating the parameters of the centered BM according to Equations (19) – (20) with a learning rate η leads to an updated set of parameters w'_{ij}, b'_i, given by

$$
\begin{aligned}
w'_{ij} &\overset{(19)}{=} w_{ij} + \eta(\langle (x_i - \mu_i)(x_j - \mu_j) \rangle_d - \langle (x_i - \mu_i)(x_j - \mu_j) \rangle_m) \ , &(28)\\
b'_i &\overset{(20)}{=} b_i + \eta(\langle x_i \rangle_d - \langle x_i \rangle_m) \ . &(29)
\end{aligned}
$$

One can now transform the updated centered BM back to a normal BM by applying the inverse transformation to the updated parameters, which finally leads to the centered gradient.

$$
\begin{aligned}
\hat{w}'_{ij} &\overset{(26)}{=} w'_{ij} \\
&\overset{(26,28)}{=} \hat{w}_{ij} + \eta \underbrace{\left(\langle (x_i - \mu_i)(x_j - \mu_j) \rangle_d - \langle (x_i - \mu_i)(x_j - \mu_j) \rangle_m \right)}_{= \nabla_c \hat{w}_{ij}} \ , &(30)
\end{aligned}
$$

$$
\begin{aligned}
\hat{b}'_i &\overset{(27)}{=} b'_i - \sum_{j \neq i} \hat{w}'_{ij} \mu_j \\
&\overset{(29,30)}{=} b_i + \eta(\langle x_i \rangle_d - \langle x_i \rangle_m) - \sum_{j \neq i} (\hat{w}_{ij} + \eta \nabla_c \hat{w}_{ij}) \mu_j \\
&\overset{(27)}{=} \hat{b}_i + \sum_{j \neq i} \hat{w}_{ij} \mu_j + \eta(\langle x_i \rangle_d - \langle x_i \rangle_m) - \sum_{j \neq i} \hat{w}_{ij} \mu_j - \sum_{j \neq i} \eta \nabla_c \hat{w}_{ij} \mu_j \\
&= \hat{b}_i + \underbrace{\eta(\langle x_i \rangle_d - \langle x_i \rangle_m - \sum_{j \neq i} \nabla_c \hat{w}_{ij} \mu_j)}_{\overset{(14)}{=} \nabla_c \hat{b}_i} \ . &(31)
\end{aligned}
$$

In case of an RBM the visible units are only connected to hidden units and *vice versa* such that the sum in Equation (31) either sums over all visible units or all hidden units leading to the centered gradient for RBMs given by Equations (13) – (15) (see Melchior et al., 2013, for a detailed derivation for RBMs).

Appendix C. Enhanced Gradient as a Special Case of the Centered Gradient

In the following we show that the enhanced gradient can be derived as a special case of the centered gradient. We show the equivalence for RBMs since the enhanced gradient was originally proposed for RBMs. However, the derivations for BMs and DBMs are analogous.

By setting $\boldsymbol{\mu} = \frac{1}{2}(\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m)$ and $\boldsymbol{\lambda} = \frac{1}{2}(\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m)$ we get

$$
\begin{aligned}
\nabla_c \mathbf{W} &\overset{(13)}{=} \langle (\mathbf{x} - \boldsymbol{\mu})(\mathbf{h} - \boldsymbol{\lambda})^T \rangle_d - \langle (\mathbf{x} - \boldsymbol{\mu})(\mathbf{h} - \boldsymbol{\lambda})^T \rangle_m \\
&= \langle \mathbf{x}\mathbf{h}^T \rangle_d - \langle \mathbf{x} \rangle_d \boldsymbol{\lambda}^T - \boldsymbol{\mu} \langle \mathbf{h}^T \rangle_d + \boldsymbol{\mu}\boldsymbol{\lambda}^T - \langle \mathbf{x}\mathbf{h}^T \rangle_m + \langle \mathbf{x} \rangle_m \boldsymbol{\lambda}^T + \boldsymbol{\mu} \langle \mathbf{h}^T \rangle_m - \boldsymbol{\mu}\boldsymbol{\lambda}^T \\
&= \langle \mathbf{x}\mathbf{h}^T \rangle_d - \frac{1}{2} \langle \mathbf{x} \rangle_d (\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m)^T - \frac{1}{2} (\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m) \langle \mathbf{h}^T \rangle_d \\
&\quad - \langle \mathbf{x}\mathbf{h}^T \rangle_m + \frac{1}{2} \langle \mathbf{x} \rangle_m (\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m)^T + \frac{1}{2} (\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m) \langle \mathbf{h}^T \rangle_m \\
&= \langle \mathbf{x}\mathbf{h}^T \rangle_d - \frac{1}{2} \langle \mathbf{x} \rangle_d \langle \mathbf{h}^T \rangle_d - \frac{1}{2} \langle \mathbf{x} \rangle_d \langle \mathbf{h}^T \rangle_m - \frac{1}{2} \langle \mathbf{x} \rangle_d \langle \mathbf{h}^T \rangle_d - \frac{1}{2} \langle \mathbf{x} \rangle_m \langle \mathbf{h}^T \rangle_d \\
&\quad - \langle \mathbf{x}\mathbf{h}^T \rangle_m + \frac{1}{2} \langle \mathbf{x} \rangle_m \langle \mathbf{h}^T \rangle_d + \frac{1}{2} \langle \mathbf{x} \rangle_m \langle \mathbf{h}^T \rangle_m + \frac{1}{2} \langle \mathbf{x} \rangle_d \langle \mathbf{h}^T \rangle_m + \frac{1}{2} \langle \mathbf{x} \rangle_m \langle \mathbf{h}^T \rangle_m \\
&= \langle \mathbf{x}\mathbf{h}^T \rangle_d - \langle \mathbf{x} \rangle_d \langle \mathbf{h}^T \rangle_d - \langle \mathbf{x}\mathbf{h}^T \rangle_m + \langle \mathbf{x} \rangle_m \langle \mathbf{h}^T \rangle_m \\
&= \langle (\mathbf{x} - \langle \mathbf{x} \rangle_d)(\mathbf{h} - \langle \mathbf{h} \rangle_d)^T \rangle_d - \langle (\mathbf{x} - \langle \mathbf{x} \rangle_m)(\mathbf{h} - \langle \mathbf{h} \rangle_m)^T \rangle_m \\
&\overset{(1)}{=} \nabla_e \mathbf{W} \ , &(32)
\end{aligned}
$$

and for the derivatives with respect to the bias parameters follows directly that

$$
\begin{aligned}
\nabla_c \mathbf{b} &\overset{(14,32)}{=} \langle \mathbf{x} \rangle_d - \langle \mathbf{x} \rangle_m - \nabla_e \mathbf{W} \boldsymbol{\lambda} \\
&= \langle \mathbf{x} \rangle_d - \langle \mathbf{x} \rangle_m - \nabla_e \mathbf{W} \frac{1}{2} (\langle \mathbf{h} \rangle_d + \langle \mathbf{h} \rangle_m) \\
&\overset{(2)}{=} \nabla_e \mathbf{b} \ ,
\end{aligned}
$$

$$\begin{aligned}\nabla_c \mathbf{c} &\overset{(15,32)}{=} \langle \mathbf{h} \rangle_d - \langle \mathbf{h} \rangle_m - \nabla_e \mathbf{W}^T \boldsymbol{\mu} \\ &= \langle \mathbf{h} \rangle_d - \langle \mathbf{h} \rangle_m - \nabla_e \mathbf{W}^T \frac{1}{2} \left(\langle \mathbf{x} \rangle_d + \langle \mathbf{x} \rangle_m \right) \\ &\overset{(3)}{=} \nabla_e \mathbf{c} \ .\end{aligned}$$

Appendix D. Comparison of the Different MNIST Variants

In this section we compare the performance of normal and centered RBMs on different variants of the *MNIST* data set (LeCun et al., 1998). The data set is provided at `http://yann.lecun.com/exdb/mnist/` and consists of pixel values in the range of 0 to 255.

Algorithm-η	dd_s^b			00		
MNIST-Probabilities						
CD-1-0.1	**-156.46**	±2.48	(-157.1)	-168.54	±3.99	(-168.5)
CD-1-0.05	**-155.60**	±2.31	(-157.0)	-172.34	±2.53	(-172.4)
CD-1-0.01	**-155.40**	±2.24	(-155.4)	-171.66	±1.85	(-173.3)
PCD-1-0.1	**-147.42**	±1.06	(-148.3)	-166.70	±3.00	(-183.1)
PCD-1-0.05	**-145.42**	±1.24	(-145.7)	-160.54	±4.76	(-162.5)
PCD-1-0.01	**-144.78**	±0.43	(-144.8)	-148.85	±0.99	(-149.0)
PT_{20}-0.01	**-144.92**	±0.57	(-144.9)	-150.97	±2.45	(-151.1)
MNIST-Sampled						
CD-1-0.1	**-155.21**	±1.63	(-157.8)	-169.95	±2.61	(-170.1)
CD-1-0.05	**-154.84**	±2.01	(-156.2)	-170.84	±2.92	(-171.0)
CD-1-0.01	**-154.95**	±1.96	(-155.0)	-171.90	±2.23	(-173.6)
PCD-1-0.1	**-145.03**	±1.13	(-146.0)	-151.04	±3.73	(-152.3)
PCD-1-0.05	**-144.10**	±1.16	(-144.1)	-147.53	±1.61	(-147.9)
PCD-1-0.01	**-143.86**	±0.63	(-143.9)	-146.89	±0.66	(-147.0)
PT_{20}-0.01	**-144.01**	±0.63	(-144.2)	-149.18	±1.51	(-149.2)
MNIST-Threshold						
CD-1-0.1	**-151.94**	±2.38	(-154.0)	-168.96	±4.42	(-169.2)
CD-1-0.05	**-151.84**	±3.72	(-152.5)	-170.39	±3.01	(-170.5)
CD-1-0.01	**-151.53**	±2.17	(-151.5)	-169.15	±2.01	(-171.7)
PCD-1-0.1	**-143.56**	±2.43	(-144.8)	-164.34	±3.30	(-187.2)
PCD-1-0.05	**-141.00**	±1.54	(-142.0)	-156.87	±4.15	(-160.3)
PCD-1-0.01	**-139.61**	±0.91	(-139.8)	-143.70	±0.95	(-143.8)
PT_{20}-0.01	**-140.20**	±1.08	(-140.3)	-146.75	±1.96	(-147.0)

Table 13: Maximum average LL on the test data for centered and normal RBMs with 16 hidden units trained on the three different variants of MNIST averaged over 25 trials. The models were trained for 100 epochs with a batch size of 100 and in the case of dd_s^b a sliding factor of 0.01 was used.

Algorithm-η	dd_s^b			00		
MNIST-Probabilities						
PCD-1-0.01	-100.32	±1.10	(-100.3)	-111.47	±1.76	(-111.5)
PCD-1-0.005	-100.04	±0.98	(-100.0)	-108.76	±1.13	(-109.8)
PT_{20}-0.01	-97.80	±0.40	(-97.8)	-105.93	±3.31	(-106.5)
MNIST-Sampled						
PCD-1-0.01	-92.00	±1.79	(-92.0)	-94.95	±1.19	(-97.8)
PCD-1-0.005	-90.85	±0.61	(-91.4)	-94.29	±1.29	(-94.3)
PT_{20}-0.01	-90.13	±0.74	(-90.1)	-98.35	±2.68	(-99.8)
MNIST-Threshold						
PCD-1-0.01	-78.69	±1.22	(-80.1)	-101.06	±3.37	(-107.1)
PCD-1-0.005	-75.93	±0.69	(-75.9)	-94.72	±2.07	(-97.8)
PT_{20}-0.01	-74.77	±1.08	(-75.2)	-87.79	±5.02	(-87.8)

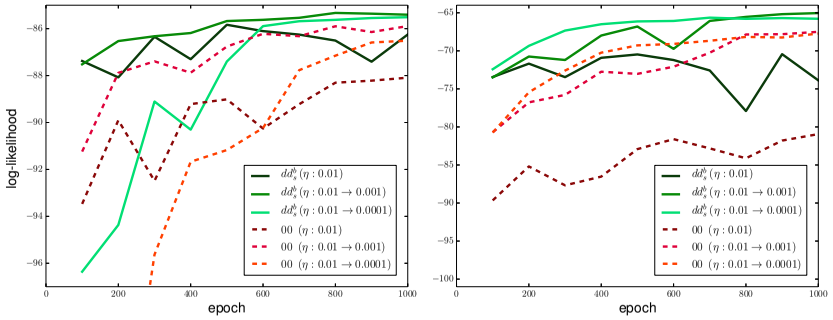
Table 14: Maximum average LL on the test data for centered and normal RBMs with 500 hidden units trained on the three different variants of MNIST averaged over 10 trials. The models were trained for 200 epochs with a batch size of 100 and in the case of dd_s^b a sliding factor of 0.01 was used.

Since the data set is not binary it has to be preprocessed when used with binary RBMs. In a first step the values are usually normalized to lie in $[0, 1]$. In different studies the normalized values are then either used directly as input, or the data set is binarized using a threshold of 0.5 or by sampling according to the normalized values (which are treated as the probabilities for the variables being in state 1). Here, we refer to these three different binarized variants of *MNIST* as *MNIST-Probabilities*, *MNIST-Threshold*, and *MNIST-Sampled*, respectively.

Since it is often not mentioned how the data set is binarized it is hard to compare the LL values reported for *MNIST* experiments across studies. Furthermore, since the results are mainly given for big models the LL is usually approximated using AIS, which easily overestimates the LL if the setup is not chosen properly. In our experience using a reasonable base partition function such as one that is based on the MLE for zero weights (Salakhutdinov and Murray, 2008) is crucial for AIS not to dramatically overestimate the LL for RBMs/DBMs. In this work, we ensured that when given the same pseudo random number generator, our AIS implementation and the *MNIST-Sampled* data set were the same as used by Salakhutdinov and Murray (2008).

As a baseline we trained normal and centered RBMs with 16 hidden units on the three different *MNIST* variants for 100 epochs using the setup described in Section 6. The maximum average of the exact LL values for the test data for dd_s^b and 00 are given in Table 13. Although the maximum reached values are quite different for the different *MNIST* variants, centered RBMs outperformed normal RBMs on all three data sets.

We continued by training normal and centered RBMs with 500 hidden units on the three different data sets for 200 epochs. The setup for training and evaluating the models was



(a) *MNIST-Sampled* (b) *MNIST-Threshold*

Figure 18: Evolution of the LL of single trials on the test data of (a) *MNIST-Sampled* and (b) *MNIST-Threshold* for dd_s^b and 00 with 500 hidden units. The models were trained for 1000 epochs with a weight decay of 0.0002 and a momentum of 0.9 that was reduced to 0.5 after 5 epochs. The learning rate was either fixed to 0.01, decayed from 0.01 to 0.001 or from 0.01 to 0.0001 over 1000 epochs.

the same as described in Section 6. The maximum average LL values are given in Table 14, showing again that the reached LL on the different *MNIST* variants is quite different and that centered RBMs reach better values than normal RBMs on all three data sets.

In our experience the use of weight decay, momentum and an annealing learning rate is crucial to reach a LL that is comparable to the value of -86.34 reported by Salakhutdinov and Murray (2008). We therefore trained centered RBMs (dd_s^b) and normal RBMs (00) using PCD-25 for the extensive amount of 1000 epochs (600000 gradient updates). In addition a weight decay of 0.0002, and a momentum of 0.9 that was set to 0.5 after 5 epochs was used. The learning rate was fixed to 0.01 or either decayed from 0.01 to 0.001 or from 0.01 to 0.0001 over 1000 epochs.

The average LL on the test data of *MNIST-Sampled* for single trials are shown in Figure 18(a). It can be seen that a decaying learning rate is crucial for 00, which only reaches a value of -88.0 without decaying learning rate but -85.9 with a learning rate that decayed from 0.01 to 0.001. dd_s^b reaches a LL value of -86.2 without and -85.5 with a learning rate that decayed from 0.01 to 0.001. Although the difference between normal and centered RBMs become smaller with weight decay, dd_s^b still reaches a higher LL, seems to be less depended on the learning rate schedule and allows faster learning. We performed the same experiments also for the *MNIST-Threshold* data set. The results are shown in Figure 18(b), which show qualitatively the same results. The difference between 00 and dd_s^b is more prominent on the *MNIST-Threshold* where 00 reaches a LL value of -67.5 and dd_s^b reaches -65.0, which can also be seen from Table 14.

References

S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

S. Amari, K. Koji, and N. Hiroshi. Information geometry of Boltzmann machines. *IEEE Transactions on Neural Networks*, 3(2):260–271, 1992.

Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 21(6):1601–1621, 2009.

Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, 19:153, 2007.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

K. Cho, T. Raiko, and A. Ilin. Parallel tempering is efficient for learning restricted Boltzmann machines. In *Proceedings of the International Joint Conference on Neural Networks*, pages 3246–3253. IEEE Press, 2010.

K. Cho, T. Raiko, and A. Ilin. Enhanced gradient and adaptive learning rate for training restricted Boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, pages 105–112. Omnipress, 2011.

K. Cho, T. Raiko, and A. Ilin. Gaussian-Bernoulli deep Boltzmann machine. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–7. IEEE, 2013a.

K. Cho, T. Raiko, and A. Ilin. Enhanced gradient for training restricted Boltzmann machines. *Neural Computation*, 25(3):805–831, 2013b.

G. Desjardins, A. Courville, Y. Bengio, P. Vincent, and O. Dellaleau. Tempered Markov Chain Monte Carlo for training of restricted Boltzmann machines. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 9, pages 145–152, 2010.

G. Desjardins, R. Pascanu, A. Courville, and Y. Bengio. Metric-free natural gradient for joint-training of Boltzmann machines. *Computing Research Repository*, 2013.

A. Fischer. *Training Restricted Boltzmann Machines*. PhD thesis, University of Copenhagen, 2014.

A. Fischer and C. Igel. Empirical analysis of the divergence of Gibbs sampling based learning algorithms for restricted Boltzmann machines. In *Proceedings of the International Conference on Artificial Neural Networks*, volume 6354 of *LNCS*, pages 208–217. Springer-Verlag, 2010.

A. Fischer and C. Igel. Bounding the bias of contrastive divergence learning. *Neural Computation*, 23(3):664–673, 2011.

A. Fischer and C. Igel. Training restricted Boltzmann machines: An introduction. *Pattern Recognition*, 47:25–39, 2014.

A. Fischer and C. Igel. A bound for the convergence rate of parallel tempering for sampling restricted Boltzmann machines. *Theoretical Computer Science*, 2015.

R. Grosse and R. Salakhudinov. Scaling up natural gradient by sparsely factorizing the inverse fisher matrix. In *Proceedings of the International Conference on Machine Learning*, 2015.

G. E. Hinton. A practical guide to training restricted Boltzmann machines. Technical report, Department of Computer Science, University of Toronto, 2010.

G. E. Hinton and R. Salakhutdinov. A better way to pretrain deep Boltzmann machines. In *Advances in Neural Information Processing Systems*, pages 2447–2455. Curran Associates, Inc., 2012.

G. E. Hinton, O. Simon, and T. Yee-Whye. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Computing Research Repository*, abs/1502.03167, 2015. URL `http://arxiv.org/abs/1502.03167`.

H. J. Kelley. Gradient theory of optimal flight paths. *Ars Journal*, 30(10):947–954, 1960.

H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 15, pages 29–37, 2011.

H. Larochelle, Y. Bengio, and J. Turian. Tractable multivariate binary density estimation and the restricted Boltzmann forest. *Neural Computation*, 22:22852307, 2010.

Y. LeCun, L. Bottou, G. Orr, and K. R. Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, Lecture Notes in Computer Science, page 546. Springer Berlin / Heidelberg, 1998.

M. Lingenheil, R. Denschlag, G. Mathias, and P. Tavan. Efficiency of exchange schemes in replica exchange. *Chemical Physics Letters*, 478:80–84, 2009.

D. MacKay. *Information Theory, Inference, and Learning Algorithm*. Cambridge University Press, Cambridge, 2003.

B. Marlin, K. Swersky, B. Chen, and N. de Freitas. Inductive principles for learning restricted Boltzmann machines. In *Artificial Intelligence and Statistics Conference*, 2010.

J. Melchior, A. Fischer, and L. Wiskott. How to center binary restricted Boltzmann machines. *Computing Research Repository*, 2013. URL `http://arxiv.org/abs/1311.1354`.

G. Montavon and K. R. Müller. Deep Boltzmann machines and the centering trick. *Lecture Notes in Computer Science*, 7700:621–637, 2012.

A. Ng. Sparse autoencoder. Technical report, Stanford University, 2011.

Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *Computing Research Repository*, 2011.

B. A. Olshausen et al. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

C. Poultney, S. Chopra, Y. L. Cun, et al. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*, pages 1137–1144, 2006.

T. Raiko, H. Valpola, and Y. LeCun. Deep learning made easier by linear transformations in perceptrons. *Journal of Machine Learning Research*, 22:924–332, 2012.

S. Rifai, P. Vincent, X Muller, X. Glorot, and Y. Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the International Conference on Machine Learning*, pages 833–840, 2011.

D. Rumelhart, G. E. Hinton, and R. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986a.

D. Rumelhart, J. McClelland, and the PDP Research Group. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1. MIT Press, Cambridge, 1986b.

R. Salakhutdinov and G. E. Hinton. Deep Boltzmann machines. In *Artificial Intelligence and Statistics Conference*, volume 5, pages 448–455, Clearwater Beach, Florida, USA, 2009.

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the International Conference on Machine Learning*, pages 872–879, New York, 2008. ACM.

N. Schraudolph. Centering neural network gradient factors. In *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, pages 207–226. Springer Verlag, Berlin, 1998.

H. Schulz, A. Müller, and S. Behnke. Investigating convergence of restricted Boltzmann machine learning. In *Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

B. Schwehn. Using the natural gradient for training restricted Boltzmann machines. Master's thesis, University of Edinburgh, Edinburgh, 2010.

P. Smolensky. *Information Processing in Dynamical Systems: Foundations of Harmony Theory*. MIT Press, Cambridge, MA, USA, 1986.

R. H. Swendsen and J. S. Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607, 1986.

K. Swersky, B. Chen, B. Marlin, and N. de Freitas. A tutorial on stochastic approximation algorithms for training restricted Boltzmann machines and deep belief nets. In *Information Theory and Applications*, 2010.

Y. Tang and I. Sutskever. Data normalization in the learning of restricted Boltzmann machines. Technical report, Department of Computer Science, University of Toronto, 2011.

T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the International Conference on Machine Learning*, pages 1064–1071. ACM, 2008.

T. Tieleman and G. E. Hinton. Using fast weights to improve persistent contrastive divergence. In *Proceedings of the International Conference on Machine Learning*, pages 1033–1040, New York, 2009. ACM.

P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the International Conference on Machine Learning*, pages 1096–1103. ACM, 2008.

L. Younes. Maximum likelihood estimation of Gibbs fields. In *Proceedings of the Joint Conference on Spacial Statistics and Imaging*, Lecture Notes Monograph Series. Institute of Mathematical Statistics, Hayward, California, 1991.

Journal of Machine Learning Research 17 (2016) 1-35 Submitted 9/14; Revised 2/16; Published 2/16

Control Function Instrumental Variable Estimation of Nonlinear Causal Effect Models

Zijian Guo ZIJGUO@WHARTON.UPENN.EDU
Department of Statistics
University of Pennsylvania
Philadelphia,19104, USA

Dylan S. Small DSMALL@WHARTON.UPENN.EDU
Department of Statistics
University of Pennsylvania
Philadelphia,19104, USA

Editor: Isabelle Guyon and Alexander Statnikov

Abstract

The instrumental variable method consistently estimates the effect of a treatment when there is unmeasured confounding and a valid instrumental variable. A valid instrumental variable is a variable that is independent of unmeasured confounders and affects the treatment but does not have a direct effect on the outcome beyond its effect on the treatment. Two commonly used estimators for using an instrumental variable to estimate a treatment effect are the two stage least squares estimator and the control function estimator. For linear causal effect models, these two estimators are equivalent, but for nonlinear causal effect models, the estimators are different. We provide a systematic comparison of these two estimators for nonlinear causal effect models and develop an approach to combing the two estimators that generally performs better than either one alone. We show that the control function estimator is a two stage least squares estimator with an augmented set of instrumental variables. If these augmented instrumental variables are valid, then the control function estimator can be much more efficient than usual two stage least squares without the augmented instrumental variables while if the augmented instrumental variables are not valid, then the control function estimator may be inconsistent while the usual two stage least squares remains consistent. We apply the Hausman test to test whether the augmented instrumental variables are valid and construct a pretest estimator based on this test. The pretest estimator is shown to work well in a simulation study. An application to the effect of exposure to violence on time preference is considered.

Keywords: Causal Inference, Control Function Estimator,Endogenous Variable, Instrumental Variable Method, Two Stage Least Squares Estimator, Pretest Estimator.

1. Introduction

Randomized controlled studies are the gold standard to estimate the treatment effect. Unfortunately, randomized controlled studies are often not feasible because of cost or ethical constraints. When randomized studies are not feasible, observational studies provide an alternative source of data for estimating the treatment effect. Since treatments were not randomly assigned, a major concern in observational studies is the possible presence of unmeasured confounders that affect both the treatment and the outcome. The instrumental variable method is designed to estimate the effects of treatments when there are unmeasured confounders. The method requires a valid instrumental variable which is a variable that (1) is associated with the treatment conditioning on the measured covariates; (2) is independent of the unmeasured confounders conditioning on the measured covariates; (3) has no direct effect on the outcome beyond its effect on the treatment. The definition of an instrumental variable is illustrated in Figure 1, where the arrows represent the causal relationship. The lack of an arrow between the instrumental variable and unmeasured confounding represents assumption (2); The lack of an arrow between the instrumental variable and outcome represents assumption (3). The existence of an arrow between the treatment and instruments represents assumption (1) when the instrumental variable causes the treatment. Note that assumption (1) is also satisfied when the instrumental variable is associated with the treatment, but does not cause the treatment, see Hernan & Robins (2006). See Holland (1988), Angrist et al. (1996), Tan (2006), Cheng et al. (2009), Brookhart et al. (2009), Baiocchi et al. (2014) and Imbens (2014) for good discussions of instrumental variable methods.

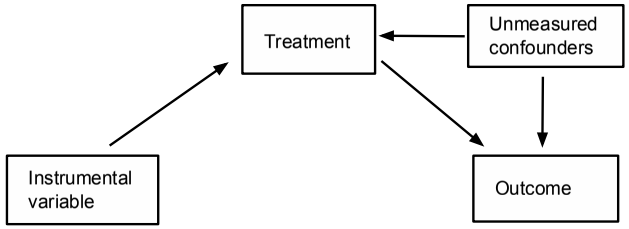


Figure 1: The directed acyclic graph for the relationship between the instrumental variable, treatment, unmeasured confounders and outcome.

Two commonly used methods for using an instrumental variable to estimate the treatment effect are the two stage least squares method and the control function method. Both of these methods are implemented in two stages. In the first stage of the two stage least squares method, we regress the treatment on the instruments and baseline covariates. In the second stage, the outcome is regressed on the predicted value from the first stage and baseline covariates and the coefficients of the second stage regression are taken as the two stage least squares estimates. Compared to the ordinary least squares method, the second stage of the two stage least squares method replaces the treatment with the corresponding predicted value from the first stage. The basic idea of the two stage least squares method is to extract variation in the treatment that is independent of the unmeasured confounders and use this variation to estimate the treatment effect. In contrast to two stage least squares, the control function method uses the instrumental variable to split the unmeasured confounders into two parts: one part that is correlated with the treatment and the other part that is uncorrelated with the treatment. The first stage of the control function method is similar

to that of the two stage least square method. In the second stage of the control function method, the outcome is regressed on the treatment, baseline covariates and the residual of the first stage regression and the coefficients of the second stage regression are taken as the control function estimates. Intuitively, the residual of the first stage regression accounts for the unmeasured confounders. The method is called the control function method because the effect of the unmeasured confounders is controlled for via the inclusion of the control function (residual of the first stage regression) in the second stage regression. The control function method has been developed in econometrics (Heckman (1976); Ahn & Powell (2004); Andrews & Schafgans (1998); Blundell & Powell (2004); Imbens & Wooldridge (2007)) and also in biostatistics and health services research (Nagelkerke et al (2000) and Terza et al. (2008)), where the method has been called the two stage residual inclusion method.

When the treatment has a linear effect on the outcome, then the control function method and two stage least squares produce identical estimates. However, when the treatment has a nonlinear effect on the outcome, then the methods produce different estimates. Settings in which the treatment is thought to have a nonlinear effect on the outcome are common; examples include the effect of the concentration of a drug in a person's body on the body's response (Nedelman et al, 2007), the effect of education on earnings (Card, 1994), the effect of a bank's financial capital on its costs (Hughes & Mester, 1998) and the effect of industry lobbying on government tariff policy (Gawande & Bandyopadhyay , 2000).

Imbens & Wooldridge (2007) conjectured that the control function estimator, while less robust than two stage least squares, might be much more precise because it keeps the treatment variables in the second stage. However, Imbens and Wooldridge note that a systematic comparison had not been done between the two estimators. The goal of this paper is to provide such a systematic comparison of the control function method to two stage least squares for nonlinear causal effect models.

We illuminate the relationship between control function and two stage least squares estimators by showing that the control function estimator is equivalent to a two stage least squares estimator with an augmented set of instrumental variables. When these augmented instrumental variables are valid, the control function method is more efficient than usual two stage least squares. For example, in the setting (1) of Table 1, the control function estimator is considerably more efficient than two stage least squares, sometimes more than 10 times more efficient. However, when the augmented instrumental variables are invalid, the control function method is inconsistent. Thus, the key issue for deciding whether to use the control function estimator vs. usual two stage least squares is whether the augmented instrumental variables are valid. The validity of the augmented instrumental variables can be tested using the Hausman test (Hausman, 1978). We develop a pretest estimator based on this test and show that it performs well in simulation studies, being close to the control function estimator when the augmented instrumental variables are valid and close to two stage least squares when the augmented instrumental variable are invalid. The pretest estimator combines the strengths of the control function and two stage least squares estimators.

Our paper is organized as follows. In section 2, we describe the set up of our model and how to implement the two stage least squares method and the control function method. In section 3, we describe how the control function method is equivalent to two stage least

squares with an augmented set of instrumental variables. In section 4, we describe the Hausman test for the validity of the augmented instrumental variables and formulate a pretest estimator. In section 5 , we present simulation studies comparing the control function method to the usual two stage least squares method. In section 6, we apply our methods to a study of the effect of exposure of violence on time preference. In section 7, we present discussions about more generalized models. In section 8, we conclude the paper. The proof of theorems, more simulation studies and two additional data analysis are presented in the Appendix.

2. Set up of model and description of methods

In this section, we introduce the model and describe the control function and the two stage least squares method. In Section 2.1, the model is introduced; In Section 2.2 and 2.3, we describe the two stage least squares method and the control function method, respectively. In Section 2.4, we discuss the assumptions for the control function method.

2.1 Set up of model

In this paper, our focus is on a model where the outcome variable is a non-linear function of the treatment variable. Let y_1 denote the outcome variable and y_2 denote the treatment variable. Let z_1 denote a vector of measured pre-treatment covariates and z_2 denote a vector of instrumental variables and $z = (z_1, z_2)$. In the following discussion, we assume both z_1 and z_2 are one dimensional. In section 3.4, we will extend the method to allow z_1 and z_2 to be vectors.

For defining the causal effect of the treatment, we use the potential outcome approach (Rubin (1974) and Neyman (1923)). Let $y_1^{(y_2^*)}$ denote the outcome that would be observed if the unit is assigned treatment level $y_2 = y_2^*$. An additive, non-linear causal effect model for the potential outcomes (similar to the linear causal effect model in Holland (1988)) is

$$y_1^{(y_2^*)} = y_1^{(0)} + \beta_1 g_1(y_2^*) + \beta_2 g_2(y_2^*) + \cdots + \beta_k g_k(y_2^*), \tag{1}$$

where $g_1(y_2^*) = y_2^*$ and $g_1, \cdots, g_k$ are linearly independent functions. Since g_1 is linear, $g_2, \cdots, g_k$ are non-linear functions. The causal effect of increasing y_2 from y_2^* to $y_2^* + 1$ is

$$(\beta_1(y_2^* + 1) + \beta_2 g_2(y_2^* + 1) + \cdots + \beta_k g_k(y_2^* + 1)) - (\beta_1 y_2^* + \beta_2 g_2(y_2^*) + \cdots + \beta_k g_k(y_2^*)).$$

We assume that the pretreatment covariate z_1 has a linear effect on potential outcomes, $E\left(y_1^{(0)}|z_1\right) = \beta_0 + \beta_{k+1} z_1$ and denote the residual by $u_1 = y_1^{(0)} - E(y_1^{(0)}|z_1)$. The model for the observed data is

$$y_1 = \beta_0 + \beta_1 g_1(y_2) + \beta_2 g_2(y_2) + \cdots + \beta_k g_k(y_2) + \beta_{k+1} z_1 + u_1, \tag{2}$$

where y_2 is the observed value of the treatment variable and $g_1(y_2) = y_2$. For identifiability, we assume that $g_i(0) = 0$ for $2 \le i \le k$. We also assume that y_2's expectation given z_1 and z_2 is linear in parameters but possibly nonlinear in z_1 and z_2,i.e.,

$$y_2 = \alpha_0 + \alpha_1^T \mathbf{J}(z_1) + \alpha_2^T \mathbf{H}(z_2) + v_2, \tag{3}$$

where $\alpha_2 \neq \mathbf{0}$, z_1 and z_2 are independent of u_1 and v_2, u_1 and v_2 are potentially correlated, and $\mathbf{H} = (z_2, h_2(z_2), \cdots, h_k(z_2))$ is a known vector of linearly independent functions of z_2. $\mathbf{J}$ is a known vector of functions, which can be a nonlinear function, for simplicity of notation, we will assume a linear effect of z_1 henceforth,

$$y_2 = \alpha_0 + \alpha_1 z_1 + \alpha_2^T \mathbf{H}(z_2) + v_2, \tag{4}$$

We say that $I(z_2) = (z_2, h_2(z_2) \cdots, h_k(z_2))$ are valid instrumental variables if $I(z_2)$ satisfies the following two assumptions (Stock (2002), White (1984)(p.8) and Wooldridge (2010)(p.99)):

Assumption 1 (***Relevance***) *The instruments are correlated with the endogenous variables* $W = (g_1(y_2), \cdots, g_k(y_2))$ *given* z_1*, that is,* $E\left(WI(z_2)^T | z_1\right)$ *is of full column rank.*

Assumption 2 (***Exogeneity***) *The instruments are uncorrelated with the error in (2), that is,*

$$E(I(z_2)u_1) = 0.$$

To ensure assumption 1, we need to have at least k valid instrument variables for $g_1(y_2), \cdots, g_k(y_2)$. This requires that the instrument z_2 has at least k different values. Otherwise, we cannot construct k linearly independent functions of z_2 to be instruments for $g_1(y_2), \cdots, g_k(y_2)$.

Remark 1 *In econometric terminology,* y_2 *is called an endogenous variable,* z_1 *are called included exogenous variables and* $I(z_2) = (z_2, h_2(z_2) \cdots, h_k(z_2))$ *is called an excluded exogenous variable. Sometimes* $z_1, z_2, h_2(z_2), \cdots, h_k(z_2)$ *together are referred to as the instrumental variables (i.e., both the included and excluded exogenous variables), but we will just refer to* $I(z_2) = (z_2, h_2(z_2) \cdots, h_k(z_2))$ *(the excluded exogenous variable) as the instruments. Assumption 1 is called the rank condition in the econometric literature while* $r \geq k$ *is called the order condition. As discussed in Wooldridge (2010)(p.99), the order condition is necessary for the rank condition.*

Remark 2 *Sufficient conditions for Assumption 2 to hold are that (i)* I *is independent of the potential outcome* $y_1^{(0)}$ *after controlling for the measured confounders* z_1*, that is* I *is independent of unmeasured confounders and (ii)* I *has no direct effect on* y_1 *and only affects* y_1 *through* y_2*. These conditions are discussed by Holland (1988).*

2.2 Description of two stage least squares method

In the following, we will describe the two stage least squares (2SLS) method briefly. Let $y \sim x_1 + x_2$ denote linear regression of y on x_1, x_2 and an intercept. Following Kelejian (1971), the usual two stage least squares estimator is introduced as Algorithm 1.

We will refer to this as the usual two stage least squares method as we will show in section 3 that the control function method can be viewed as a two stage least squares method with an augmented set of instruments.

Let $L(y|x)$ denote the best linear projection $\gamma^T x$ of y onto x where $\gamma = \arg\min_\tau E\left((y - \tau^T x)^2\right)$. Let $Z = \{1, z_1, z_2, h_2(z_2), \cdots, h_k(z_2)\}$ denote the exogenous variables. The linear projection

Algorithm 1 Two stage least squares estimator (2SLS)

Input: i.i.d observations of pre-treatment covariates z_1, instrumental variables $(z_2, h_2(z_2) \cdots, h_k(z_2))$, the treatment variable y_2 and outcome variable y_1.

Output: The two stage least squares estimator $\widehat{\beta}^{\text{2SLS}}$ of treatment effects $\beta = (\beta_1, \beta_2, \cdots, \beta_k)$ in (2).

1: Implement the regression

$$\begin{aligned} y_2 &\sim z_1 + z_2 + h_2(z_2) + \cdots + h_k(z_2), \\ &\vdots \\ g_k(y_2) &\sim z_1 + z_2 + h_2(z_2) + \cdots + h_k(z_2), \end{aligned} \tag{5}$$

and obtain the corresponding predicted values as $\left(\widehat{y_2}, \widehat{g_2(y_2)}, \cdots, \widehat{g_k(y_2)}\right)$.

2: Implement the regression:

$$y_1 \sim \widehat{y_2} + \cdots + \widehat{g_k(y_2)} + z_1. \tag{6}$$

The two stage least squares estimates are the coefficient estimates on $y_2, g_2(y_2), \cdots, g_k(y_2)$ from (6).

of y_1 onto $1, L(g_1(y_2)|Z), \cdots, L(g_k(y_2)|Z), z_1$ is

$$L(y_1|1, L(g_1(y_2)|Z), \cdots, L(g_k(y_2)|Z), z_1) = \beta_0 + \sum_{i=1}^{k} \beta_i L(g_i(y_2)|Z) + \beta_{k+1} z_1.$$

Thus, if we know $L = \{L(g_1(y_2)|Z), \cdots, L(g_k(y_2)|Z)\}$, then the least squares estimate of y on $1, L(g_1(y_2)|Z), \cdots, L(g_k(y_2)|Z), z_1$ will provide an unbiased estimate of β. Even though L is unknown, we can obtain a consistent estimate of L. Substituting this estimate into the second stage regression provides a consistent estimate of β under regularity conditions. See White (1984)(p21, p32) for the details.

2.3 Description of the control function method

We introduce the control function (CF) method in Algorithm 2.

2.4 Additional assumptions of control function

The consistency of control function estimators requires stronger assumptions than $I(z_2)$ satisfying Assumptions 1 and 2, that is, stronger than $I(z_2)$ being valid instrumental variables. The control function method is consistent under the following additional assumptions in addition to Assumptions 1 and 2 (Imbens & Wooldridge , 2007):

Assumption 3 u_1 *and* v_2 *are independent of* $z = (z_1, z_2)$.

Assumption 4 $E(u_1|v_2) = L(u_1|v_2)$ *where* $L(u_1|v_2) = \rho v_2$ *and* $\rho = \frac{E(u_1 v_2)}{E(v_2^2)}$.

Algorithm 2 Control function estimator (CF)

Input: i.i.d observations of pre-treatment covariates z_1, instrumental variables $(z_2, h_2(z_2)\cdots, h_k(z_2))$, the treatment variable y_2 and outcome variable y_1.

Output: The control function estimator $\widehat{\beta}^{\text{CF}}$ of treatment effects $\beta = (\beta_1, \beta_2, \cdots, \beta_k)$ in (2).

1: Implement the regression

$$y_2 \sim z_1 + z_2 + h_2(z_2) + \cdots + h_k(z_2) \tag{7}$$

and obtain the predicted value $\widehat{y_2}$ and the residual $e_1 = y_2 - \widehat{y_2}$.

2: Implement the regression

$$y_1 \sim y_2 + \cdots + g_k(y_2) + z_1 + e_1. \tag{8}$$

The control function estimates are the coefficient estimates on $y_2, g_2(y_2), \cdots, g_k(y_2)$ from the above regression (8).

Remark 3 *We only assume $z = (z_1, I(z_2))$ is uncorrelated with u_1 in Assumption 2. Two stage least squares does not make assumptions about v_2 but the control function method relies on assumptions about v_2. If (u_1, v_2) follow bivariate normal distribution, Assumption 4 is satisfied automatically.*

Model (29) is an example where $I(z_2) = \left(z_2, z_2^2\right)$ are valid instrumental variables that satisfy Assumptions 1 and 2 but not Assumption 3 and 4. The simulation results in Table 1 show that the control function estimate for model (29) is biased.

Assumptions 3 and 4 are sufficient but not necessary conditions for the control function method to be consistent. To investigate the consistency of the control function method, we can express (2) as

$$y_1 = \beta_0 + \beta_1 g_1(y_2) + \beta_2 g_2(y_2) + \cdots + \beta_k g_k(y_2) + \beta_{k+1} z_1 + \rho v_1 + e, \tag{9}$$

where e is $u_1 - \rho v_2$. Under some regularity conditions (Wooldridge (2010) p.56), one sufficient condition for the control function method to be consistent is that the new error e in (9) is uncorrelated with $g_i(y_2)$:

$$E\left(g_i(y_2)e\right) = 0 \quad \text{for} \quad i = 1, \cdots, k. \tag{10}$$

Assumptions 1,2, 3 and 4 lead to $E\left(f(y_2)e\right) = 0$ for all f, which is stronger than the sufficient condition (10). In Section 4, we develop a test for the consistency of the control function method that can help to decide whether the control function estimator should be used.

3. Relation between control function and two stage least squares

In this section, we compare the control function and the two stage least squares estimator. To better understand the control function method, we will first focus on the case $k = 2$

$$y_1 = \beta_0 + \beta_1 y_2 + \beta_2 g_2(y_2) + \beta_3 z_1 + u_1;$$

$$y_2 = \alpha_0 + \alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 h_2(z_2) + v_2.$$

For example, a common model has $g_2(y_2) = y_2^2$ and $h_2(z_2) = z_2^2$.

3.1 Loss of information for the control function method and comparison to two stage least squares

Two stage least squares regresses y_1 on $\widehat{y_2} = LS\left(y_2|1, z_1, z_2, h_2(z_2)\right)$, $\widehat{g_2(y_2)} = LS\left(g_2(y_2)|1, z_1, z_2, h_2(z_2)\right)$ and z_1 where $LS(A|B)$ denotes the least square estimate of the linear projection of A onto span(B). The control function estimator regresses y_1 on $y_2, g_2(y_2), z_1$ and $e_1 = y_2 - \widehat{y_2}$. Intuitively, it seems that the control function method is better than two stage least squares since it keeps y_2 and $g_2(y_2)$ in the second stage regression rather than losing information by replacing y_2 and $g_2(y_2)$ by projections as two stage least squares does. However, the control function method also loses information as explained in the following theorem.

Theorem 1 *The following regression*

$$y_1 \sim z_1 + \widetilde{y_2} + \widetilde{g_2(y_2)},$$

will have the same estimates of the corresponding coefficients as the second stage regression of the control function method:

$$y_1 \sim z_1 + y_2 + g_2(y_2) + e_1,$$

where e_1 is the residual of the regression $y_2 \sim z_1 + z_2 + h_2(z_2)$ and $\widetilde{y_2}$ is the residual of the regression $y_2 \sim e_1$ and $\widetilde{g_2(y_2)}$ is the residual of the regression $g_2(y_2) \sim e_1$. *By the property of OLS, $\widetilde{y_2}$ differs from $\hat{y}_2$ only by a constant.*

Proof In appendix B. ■

From Theorem 1, the control function method can be viewed as a two stage estimator similar to two stage least squares: in the first stage, the exogenous part of the endogenous variables, $\widetilde{y_2}$ and $\widetilde{g_2(y_2)}$, are obtained by taking residuals from regressing y_2 and $g_2(y_2)$ on the variable e_1 respectively and in the second stage, the endogenous variables in the regression are replaced by the exogenous parts.

Although both the control function method and two stage least squares lose information by projecting $g_2(y_2)$ and y_2 to subspaces, the control function method does preserve more information. Let $(\text{span}\{e_1\})^{\perp}$ denote the orthogonal complement of the space spanned by the vector e_1 and span$\{1, z_1, z_2, h_2(z_2)\}$ denote the space spanned by the vectors $1, z_1, z_2, h_2(z_2)$. The control function method projects $g_2(y_2)$ and y_2 to $(\text{span}\{e_1\})^{\perp}$ while two stage least squares projects $g_2(y_2)$ and y_2 to span$\{1, z_1, z_2, h_2(z_2)\}$, which is a subspace of $(\text{span}\{e_1\})^{\perp}$. We will now show that the control function method is equivalent to two stage least squares with an augmented set of instrumental variables.

3.2 Equivalence of control function to two stage least squares with an augmented set of instrumental variables

In section 3.1, we showed that the first stage projection space of the control function estimator is larger than that of the two stage least squares estimator. One natural question

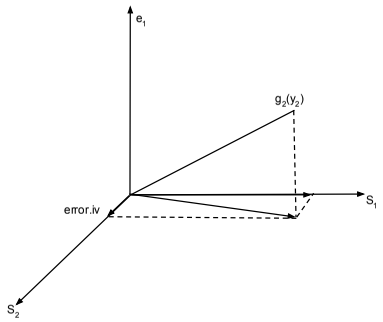


Figure 2: Projection and Decomposition of the control function Estimator. $S_1 = \text{span}\{1, z_1, z_2, h_2(z_2)\}$ and $S_2 = \text{span}\{error.iv\}$ is defined as the orthogonal complement of S_1 in the subspace $\text{span}\{e_1\}^{\perp}$.

is what is the extra part of the control function first stage projection space. The following theorem explains the extra part.

Theorem 2 *The following two regressions give the same estimates of corresponding coefficients*

$$g_2(y_2) \sim z_1 + z_2 + h_2(z_2);$$
$$\widetilde{g_2(y_2)} \sim z_1 + z_2 + h_2(z_2).$$

Proof In appendix B. ■

Theorem 2 implies that regressing the control function first stage estimate of $g_2(y_2)$, $\widetilde{g_2(y_2)}$ on $\{1, z_1, z_2, h_2(z_2)\}$ produces the same predicted value for $g_2(y_2)$ as that of the first stage of two stage least squares. Consequently, the control function method first stage estimate $\widetilde{g_2(y_2)}$ can be decomposed into two orthogonal parts: one part is the same with the two stage least squares first stage estimate and the other part is denoted as $error.iv$. Letting $\gamma_0, \gamma_1, \gamma_2, \gamma_3$ denote the coefficients in $g_2(y_2) \sim z_1 + z_2 + h_2(z_2)$, we define

$$error.iv = \widetilde{g_2(y_2)} - (\gamma_0 + \gamma_1 z_1 + \gamma_2 z_2 + \gamma_3 h_2(z_2)). \tag{11}$$

The decomposition is shown in Figure 2. Let $S_1 = \text{span}\{1, z_1, z_2, h_2(z_2)\}$ and $S_2 = \text{span}\{error.iv\}$. The projection of $g_2(y_2)$ to the space $S_1 \cup S_2$ is the control function first stage predicted value $\widetilde{g_2(y_2)}$ while the projection of $g_2(y_2)$ to the space S_1 is the two stage least squares first stage predicted value $\widehat{g_2(y_2)}$. The projection along the direction of $error.iv$ represents the extra part of the control function projection.

Since $\widetilde{g_2(y_2)}$ is the residual of the regression $g_2(y_2) \sim e_1$, the first stage regression of control function estimation can be written as

$$\begin{aligned} g_2(y_2) &= \gamma_c + \gamma e_1 + \widetilde{g_2(y_2)} \\ &= \gamma_c + \gamma e_1 + \gamma_0 + \gamma_1 z_1 + \gamma_2 z_2 + \gamma_3 h_2(z_2) + error.iv \end{aligned} \tag{12}$$

where the column vector e_1 is orthogonal to the column vectors $z_2, h_2(z_2), error.iv$ and $error.iv$ is orthogonal to z_1, z_2 and $h_2(z_2)$ by the property of OLS. The following theorem shows the equivalence of the control function estimator to the two stage least squares method with an augmented set of instruments.

Theorem 3 *The control function estimator with instruments $z_1, z_2, h_2(z_2)$ in the first stage is equivalent to the two stage least squares estimator with instruments $z_1, z_2, h_2(z_2), error.iv$ in the first stage.*

Proof In appendix B. ■

We now show that if Assumptions 3 and 4 do hold, the extra instrument $error.iv$ is a valid instrumental variable, that is, $error.iv$ satisfies Assumptions 1 and 2. Recall the definition $e = u_1 - \rho v_2$. By Assumptions 3 and 4, $E(g_2(y_2)\, e) = 0$ and the population version of $error.iv$ is

$$g_2(y_2) - L(g_2(y_2) | v_2, z_1, z_2, h_2(z_2))$$

where L means the linear projection. Since v_2 and $z_1, z_2, h_2(z_2)$ are uncorrelated with e,

$$E(error.iv \times e) = E((g_2(y_2) - L(g_2(y_2) | v_2, z_1, , h_2(z_2)))\, e) = 0.$$

3.3 Equivalence of control function with two stage least squares with augmented instruments for general nonlinear model

In the last section, we showed the equivalence of control function with two stage least squares for $k = 2$. In this section, we will show the results hold for $k \geq 3$ and present an algorithm to find the augmented set of instrumental variables for the general model. This will help us understand the control function estimator's properties since two stage least squares' properties are well understood (See Remarks 2 and section 3.5 below).

We will first consider $k = 3$ in our general model (2) and then it will be straightforward to generalize to $k > 3$. The model for $k = 3$ is

$$y_1 = \beta_0 + \beta_1 y_2 + \beta_2 g_2(y_2) + \beta_3 g_3(y_2) + \beta_4 z_1 + u_1.$$

Let $I = \{z_2, h_2(z_2), h_3(z_2)\}$ be valid instrumental variables for $y_2, g_2(y_2), g_3(y_2)$. As in the case of $k = 2$, we can find the extra instrument $error.iv_1$ for $g_2(y_2)$. If we do the same thing for $g_3(y_2)$, we will obtain the extra instrumental variable, $error.iv_2.prime$. However, there exists the problem that the error instrument $error.iv_2.prime$ is not orthogonal to $error.iv_1$. This will make $error.iv_2.prime$ have a non-zero coefficient when we treat $error.iv_2.prime$ as one of the instrumental variables for $g_2(y_2)$. The non-zero coefficient will lead to the estimate of $g_2(y_2)$ being different from what is used in the second stage of control function. To remove the correlation, we do the regression

$$error.iv_2.prime \sim error.iv_1,$$

and treat the residual of this regression as $error.iv_2$.

Theorem 4 *If we regress y_2 on $z_1, z_2, h_2(z_2), h_3(z_2)$ in the first stage, then the control function estimator is equivalent to two stage least squares with the augmented set of instrumental variables $z_1, z_2, h_2(z_2), h_3(z_2), error.iv_1, error.iv_2$.*

Proof In appendix B. ■

Remark 4 *Since the control function method is the same as two stage least squares with an augmented set of instrumental variables, we can calculate the variance of control function estimators with the help of the known variance formula for two stage least squares (Wooldridge (2010), page 102).*

This generalizes to the general $k > 3$ model. The following is an algorithm to identify the augmented set of instrumental variables for which the control function estimator is equivalent to two stage least squares with the augmented set of instrumental variables. Recall that e_1 denotes the residual of the regression $y_2 \sim z_1 + z_2 + h_2(z_2) + h_3(z_2)$ and $\widetilde{g_i(y_2)} = resid(g_i(y_2) \sim e_1)$. Set the control function projection space $S_1 = \text{span}\{z_1, z_2, h_2(z_2), h_3(z_2)\}$ for y_2. Regress $\widetilde{g_2(y_2)}$ on the space S_1 to obtain the residual $error.iv_1$ and set the control function projection space $S_2 = S_1 \oplus \text{span}\{error.iv_1\}$, where $\oplus$ denotes the direct sum of two subspaces. For $g_j(y_2), j = 3, \cdots, k$, regress $\widetilde{g_j(y_2)}$ on S_{j-1} to obtain the extra instrument $error.iv_j$, and set the control function projection space $S_j = S_{j-1} \oplus \text{span}\{error.iv_j\}$. This algorithm will produce the corresponding set of instrumental variables for the control function method. Each nonlinear function of endogenous variables will contribute one extra instrumental variable.

We can generalize Theorem 4 to models with $k > 3$.

Theorem 5 *If we regress y_2 on $z_1, z_2, h_2(z_2), \cdots, h_k(z_2)$ in the first stage, then the control function estimator is equivalent to two stage least squares with the augmented set of instrumental variables $z_1, z_2, h_2(z_2), \cdots, h_k(z_2), error.iv_1, error.iv_2, \cdots, error.iv_{k-1}$.*

The proof of Theorem 4 applies to this generalized theorem.

3.4 General nonlinear model of multidimensional instruments $\mathbf{z}_2$

In this section, we extend our results to allow $\mathbf{z}_1$ and $\mathbf{z}_2$ to be vectors. In the vector case, the first stage model is extended as

$$y_2 = \alpha_0 + \alpha_1^T \mathbf{J}(\mathbf{z}_1) + \alpha_2^T \mathbf{H}(\mathbf{z}_2) + v_2, \tag{13}$$

where $\mathbf{J}$ is a known vector of functions of $\mathbf{z}_1$ and $\mathbf{H} = (\mathbf{z}_2, h_2(\mathbf{z}_2), \cdots, h_k(\mathbf{z}_2))$ is a known vector of linearly independent functions of $\mathbf{z}_2$.

The first stage of two stage least squares is extended as

$$\begin{aligned} y_2 &\sim \mathbf{J}(\mathbf{z}_1) + \mathbf{z}_2 + h_2(\mathbf{z}_2) + \cdots + h_k(\mathbf{z}_2), \\ &\vdots \\ g_k(y_2) &\sim \mathbf{J}(\mathbf{z}_1) + \mathbf{z}_2 + h_2(\mathbf{z}_2) + \cdots + h_k(\mathbf{z}_2), \end{aligned} \tag{14}$$

where the notation $y \sim \mathbf{x}$, where $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_s)$ denotes linear regression of y on $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_s$ and an intercept. From these k first stage regression, we obtain the predicted values of k first stage regressions as $\left(\widehat{y_2}, \widehat{g_2(y_2)}, \cdots, \widehat{g_k(y_2)}\right)$. In the second stage, we do the following regression:

$$y_1 \sim \widehat{y_2} + \cdots + \widehat{g_k(y_2)} + \mathbf{z}_1. \tag{15}$$

The first stage of the control function method is

$$\begin{aligned} &y_2 \sim \mathbf{J}(\mathbf{z}_1) + \mathbf{z}_2 + h_2(\mathbf{z}_2) + \cdots + h_k(\mathbf{z}_2); \\ &e_1 = y_2 - \widehat{y_2}. \end{aligned} \tag{16}$$

In the second stage, we do the following linear regression:

$$y_1 \sim y_2 + \cdots + g_k(y_2) + \mathbf{z}_1 + e_1. \tag{17}$$

The generalization of Theorem 4 and Theorem 5 is as follows:

Theorem 6 *If we regress y_2 on $\mathbf{J}(\mathbf{z}_1), \mathbf{z}_2, h_2(\mathbf{z}_2), \cdots, h_k(\mathbf{z}_2)$ in the first stage, then the control function estimator is equivalent to two stage least squares with the augmented set of instrumental variables $\mathbf{J}(\mathbf{z}_1), \mathbf{z}_2, h_2(\mathbf{z}_2), \cdots, h_k(\mathbf{z}_2), error.iv_1, error.iv_2, \cdots, error.iv_{k-1}$.*

The proof of Theorem 4 applies to this generalized theorem.

3.5 Control function bias formula

To ensure consistency of the control function estimation, the Assumptions 1 and 2 of $z_2, h_2(z_2), \cdots, h_k(z_2)$ being valid instruments are not enough and extra assumptions are needed, such as Assumptions 3 and 4. We will present a formula for the asymptotic bias of the control function method when the extra assumptions of control function are not satisfied. Since we have shown that the control function estimator is a two stage least squares estimator with augmented instrumental variables (Sections 3.2 and 3.3), we could make use of the bias formula of two stage least squares with invalid instrumental variables given in Small (2007).

Let Y_N denote the vector of the outcome variable y_1, W_N denote the matrix whose columns are the endogenous variable $y_2, g_2(y_2), \cdots, g_k(y_2)$, X_N denote the matrix whose columns are the included exogenous variables z_1 and Z_N denote the matrix whose columns are the instrumental variables $z_2, h_2(z_2), \cdots, h_k(z_2)$ for W_N. Let $C_N = [W_N, X_N]$ and $D_N = [Z_N, X_N]$. In the second stage, we will replace C_N by the first stage predicted values $\widehat{C_N} = D_N\left(D_N^T D_N\right)^{-1} D_N^T C_N$. The two stage least squares estimator is $\left(\widehat{C_N}^T \widehat{C_N}\right)^{-1} \widehat{C_N}^T Y_N$. When the instrumental variables are not valid, there exists asymptotic bias for the estimator given in the second stage. For the model

$$y_1 = \beta_0 + \beta_1 g_1(y_2) + \cdots + \beta_k g_k(y_2) + \beta_{k+1} z_1 + u_1 \tag{18}$$

we could write u_1 as

$$u_1 = LP(u_1|Z) + u_1 - LP(u_1|Z) \tag{19}$$

where Z denote the instrumental variables and and $u_1 - LP(u_1|Z)$ is uncorrelated with Z by the property of OLS. Plugging (19) into (18):

$$y_1 = \beta_0 + \beta_1 y_2 + \beta_2 y_2^2 + \beta_3 z_1 + LP(u_1|Z) + u_1 - LP(u_1|Z)$$

$$y_1 - LP(u_1|Z) = \beta_0 + \beta_1 y_2 + \beta_2 y_2^2 + \beta_3 z_1 + u_1 - LP(u_1|Z) \tag{20}$$

For (20), we could make use of the two stage least squares method with the outcome variable $y_1 - LP(u_1|Z)$ to obtain the consistent estimate $\left(\widehat{C_N}^T \widehat{C_N}\right)^{-1} \widehat{C_N}^T \left[Y_N - LP(U_N|Z_N)\right]$, where U_N denotes the vector of error u_1 in the second stage. Hence a consistent estimator of the asymptotic bias from using the control function method is

$$\left(\widehat{C_N}^T \widehat{C_N}\right)^{-1} \widehat{C_N}^T LP(U_N|Z_N) \tag{21}$$

The control function bias formula (21) shows that in order for the control function estimator to be asymptotically unbiased, all augmented instruments are required to be uncorrelated with the error of the second stage. It is possible that the augmented instruments are correlated with the second stage error even when Z_N are valid instrumental variables. See the simulation study in section 5.3 for an example. In Section 4, we develop a test for whether the augmented instruments are correlated with the error of the second stage.

3.6 Comparison between usual two stage least squares and control function

In this section, we compare usual two stage least squares (i.e., without any augmented instruments) to the control function method under the assumptions 1 and 2 of valid instruments and the additional assumptions 3 and 4 of the control function method. For $k = 3$, $X = (y_2, g_2(y_2), g_3(y_2))$ denote the endogenous variables and $Z_1 = (z_1, z_2, h_2(z_2), h_3(z_2))$ denote the instrumental variables. $Z_2 = (error.iv_1, error.iv_2)$ denote the augmented instruments and $\widetilde{X} = (e_1, \delta_0 e_1 + error.iv_1, \delta_1 e_1 + \delta_2 error.iv_1 + error.iv_2)$ denote the residual of regressing X on Z_1. According to White (1984), the asymptotic variance of a two stage least squares estimator will not increase when adding extra valid instruments and will decrease as long as the extra valid instruments are correlated with the endogenous variables given the current instruments. In our setting, adding Z_2 to the list of instrumental variables will decrease the asymptotic variance if Z_2 is correlated with $\widetilde{X}$. Since $E(e_1 \times error.iv) = 0$, we have $E(error.iv_1 \widetilde{X}) = \left(0, E(error.iv_1^2), \delta_2 E(error.iv_1^2)\right)$ and $E(error.iv_2 \widetilde{X}) = \left(0, 0, E(error.iv_1^2)\right)$. Assume

$$g_2(y_2) \text{ is not a linear combination of } \quad 1, e_1, z_1, z_2 \text{ and } h_2(z_2) \quad \text{almost surely,} \tag{22}$$

we have $E(error.iv_1^2) > 0$ and $E(error.iv_1 \widetilde{X}) \neq 0$ and $E(error.iv_2 \widetilde{X}) \neq 0$. Hence, it follows that the control function estimator is strictly better than two stage least squares under the assumptions 1 and 2 of valid instruments, the additional assumptions 3 and 4 of the control function method and the functional assumption (22). Since e_1 is a linear function of y_2, the functional assumption (22) is saying that the nonlinear functional $g_2(y_2)$ cannot be expressed as a linear combination of linear functions of y_2 and the other variables

$1, z_1, z_2, h_2(z_2)$. For example, assuming the first stage model (13) and $g_2(y_2) = y_2^2$ and $h_2(z_2) = z_2^2$, the functional assumption (22) is automatically satisfied.

We have shown that the control function estimator is equivalent to the two stage least squares estimator with an augmented set of instrumental variables and certain assumptions will make the control function estimator consistent and more efficient than the two stage least squares estimator. Just as we can view the control function estimator as one type of two stage least squares estimator, we can view the usual two stage least squares estimator without the augmented instruments as one type of control function estimator. For $k = 3$, if we include e_1, e_2 and e_3, which are the residuals of the regression $y_2 \sim z_1 + z_2 + h_2(z_2) + h_3(z_2)$, $g_2(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2)$ and $g_3(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2)$ respectively, in the second stage regression, this control function estimator is the same as the usual two stage least squares estimator without the augmented instruments.

4. A pretest estimator

We will explain how to test the validity of the augmented instrumental variables the control function method uses for $k = 2$; it is straightforward to extend the test to the more general model. Recall the instruments for the usual two stage least squares estimator: $Z_1 = (1, z_1, z_2, h_2(z_2))$; As we have shown (Sections 3.2 and 3.3), the control function estimator is a two-stage estimator with the instruments $Z = (Z_1, error.iv)$, where $error.iv$ is the augmented instrumental variable. We use $\widehat{\beta}^{\text{2SLS}}$ to denote the usual two stage least squares estimator with instruments Z_1 and use $\widehat{\beta}^{\text{CF}}$ to denote the two stage least squares estimator with the augmented set of instruments Z. Under the null hypothesis that the augmented instrumental variable $error.iv$ is valid, the two estimators are consistent and $\widehat{\beta}^{\text{CF}}$ is efficient. Under the alternative hypothesis that the augmented instrumental variable $error.iv$ is invalid, $\widehat{\beta}^{\text{2SLS}}$ is consistent while $\widehat{\beta}^{\text{CF}}$ is not consistent.

The Hausman test (Hausman (1978)) provides a test for the validity of the augmented instrumental variable $error.iv$. The test statistic $H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}})$ is defined as

$$H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}}) = \left(\widehat{\beta}^{\text{CF}} - \widehat{\beta}^{\text{2SLS}}\right)^T \left[Cov\left(\widehat{\beta}^{\text{2SLS}}\right) - Cov\left(\widehat{\beta}^{\text{CF}}\right)\right]^- \left(\widehat{\beta}^{\text{CF}} - \widehat{\beta}^{\text{2SLS}}\right), \tag{23}$$

where $Cov\left(\widehat{\beta}^{\text{CF}}\right)$ and $Cov\left(\widehat{\beta}^{\text{2SLS}}\right)$ are the covariance matrices of $\widehat{\beta}^{\text{CF}}$ and $\widehat{\beta}^{\text{2SLS}}$ and A^- denote the Moore-Penrose pseudoinverse. Under the null hypothesis where the augmented instrument $error.iv$ is valid, the test statistic $H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}})$ is asymptotically χ_1^2.

Based on the statistic $H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}})$, we introduce a pretest estimator in Algorithm 3, which uses the control function estimator if there is not evidence that it is inconsistent and otherwise uses the usual two stage least squares estimator.

For the rest of the paper, we will use $\alpha = 0.05$ in Algorithm 3. If the p-value of the test statistic $H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}})$ is less than 0.05, then there is evidence that the control function estimator is inconsistent and the usual two stage least squares estimator is used; otherwise, the control function estimator is used.

Algorithm 3 Pretest estimator

Input: i.i.d observations of pre-treatment covariates z_1, instrumental variables $(z_2, h_2(z_2) \cdots, h_k(z_2))$, the treatment variable y_2, outcome variable y_1 and level α.

Output: The pretest estimator of treatment effects $\beta = (\beta_1, \beta_2, \cdots, \beta_k)$ in (2).

1: Implement Algorithm 1 and obtain the estimator $\widehat{\beta}^{\text{2SLS}}$ and the corresponding covariance structure $Cov\left(\widehat{\beta}^{\text{2SLS}}\right)$. Implement Algorithm 2 and obtain the estimator $\widehat{\beta}^{\text{CF}}$ and the corresponding covariance structure $Cov\left(\widehat{\beta}^{\text{CF}}\right)$.

2: Calculate the test statistic $H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}})$ as (23) and define the p-value $p = P\left(\chi^2(1) \geq H(\widehat{\beta}^{\text{CF}}, \widehat{\beta}^{\text{2SLS}})\right)$. The level α pretest estimator is defined as

$$\widehat{\beta}^{\text{Pretest}} = \begin{cases} \widehat{\beta}^{\text{CF}} & \text{if } p > \alpha \\ \widehat{\beta}^{\text{2SLS}} & \text{if } p \leq \alpha \ . \end{cases} \tag{24}$$

5. Simulation study

Our simulation study compares the usual two stage least squares (2SLS) estimator, the control function (CF) estimator and the pretest estimator. We will consider several different model settings where the assumptions of the control function estimator are satisfied, moderately violated (two settings) and drastically violated and we will also investigate the sensitivity of the control function estimator to different joint distributions of errors. The sample size is 10,000 and we implement 10,000 simulations for each setting. We report the winsorized sample mean of the estimators (WMEAN) and the winsorized root mean square error (WRMSE). The non-winsorized sample mean of the estimators (NWEAN) and the non-winsorized root mean square error (NRMSE) are reported in the appendix C. The winsorized statistics are implemented as setting the 95-100 percentile as the 95-th percentile and the 0-5 percentile as the 5-th percentile and using the winsorized data to calculate the mean and root mean square error. The winsorized statistics are less sensitive to outliers (Wilcox & Keselman , 2003). The non-winsorized results summarized in the appendix C have similar patterns as the winsorized results reported in the following discussion. The outcome model that we are considering is

$$y_1 = \beta_0 + \beta_1 z_1 + \beta_2 y_2 + \beta_3 y_2^2 + u_1; \tag{25}$$

In the following subsections, we will generate y_2 according to different models.

5.1 Setting where the assumptions of the control function estimator are satisfied

For the setting where the assumptions of control function estimator are satisfied, the model we consider is:

$$\begin{aligned} y_1 &= 1 + z_1 + 10y_2 + 10y_2^2 + u_1; \\ y_2 &= 1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + v_2; \end{aligned} \tag{26}$$

where $z_1 \sim N(0,1)$ and $z_2 \sim N(0,1)$ and $\begin{pmatrix} u_1 \\ v_2 \end{pmatrix} \sim N\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right]$. Setting (1) of Table 1 presents the absolute value of ratio of bias of the sample winsorized mean to the true value and the ratio of the sample winsorized root mean square error to the sample winsorized root mean square error of the two stage least squares estimator, where the columns under WMEAN present absolute value of the ratio of the bias and the columns under the WRMSE present the ratio of the root mean square error. Since the ratio of the sample winsorized root mean square error is using the sample winsorized root mean square error of the two stage least squares estimator as the basis, we only report the ratio of the ratio of the sample winsorized root mean square error for the control function estimator and the pretest estimator. ERR represents the empirical rejection rate of the null hypothesis that the extra instrumental variable given by the control function method is valid. The ERR is around 0.05 since the assumptions of CF estimator is satisfied in this model. The usual two stage least squares, control function and pretest estimator all have small bias. The WRMSE of control function estimator is consistently smaller than the WRMSE of the usual two stage least squares estimator. For β_2 and β_3, the WRMSE are more than 7 times smaller than that of the usual two stage least squares estimator. The pretest estimator also gains substantially over the two stage least squares estimator.

5.2 Setting where the assumptions of the control function estimator are moderately violated

We consider two models where the assumptions of the control function estimator are moderately violated,

1. y_2 is involved with a cubic function of z_2

$$\begin{aligned} y_1 &= 1 + z_1 + 10y_2 + 10y_2^2 + u_1; \\ y_2 &= \frac{1}{2} \times \frac{\left(1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + z_2^3\right)}{\text{sd}\left(1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + z_2^3\right)} + v_2; \end{aligned} \tag{27}$$

where $\text{sd}\left(1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + z_2^3\right)$ is the standard deviation of$1+\frac{1}{8}z_1+\frac{1}{3}z_2+\frac{1}{8}z_2^2+z_2^3$ and $z_1 \sim N(0,1)$ and $z_2 \sim N(0,1)$ and $\begin{pmatrix} u_1 \\ v_2 \end{pmatrix} \sim N\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right]$.

2. y_2 is involved with an exponential function of z_2.

$$\begin{aligned} y_1 &= 1 + z_1 + 10y_2 + 10y_2^2 + u_1 \\ y_2 &= \frac{1}{2} \times \frac{\left(1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + \exp(z_2)\right)}{\text{sd}\left(1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + \exp(z_2)\right)} + v_2; \end{aligned} \tag{28}$$

where $\text{sd}\left(1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + \exp(z_2)\right)$ is the standard deviation of$1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2+\exp(z_2)$ and $z_1 \sim N(0,1)$ and $z_2 \sim N(0,1)$ and $\begin{pmatrix} u_1 \\ v_2 \end{pmatrix} \sim N\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right]$.

In these simulation settings, since the terms $1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + z_2^3$ or $1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + \exp(z_2)$ involved in the y_2 model tend to be much larger than the error v_2,

we standardize these by their standard deviation. The assumptions of the control function estimator are moderately violated in models (27) and (28) since we use a quadratic model of z_2 to fit the endogenous variable y_2, whose conditional mean has a cubic (exponential) term of z_2.

Setting (2) of Table 1 presents the absolute value of ratio of bias of the sample winsorized mean to the true value of the model (27) and the ratio of the sample winsorized root mean square error to the sample winsorized root mean square error of the two stage least squares estimator. The ratio of bias of two stage least squares, control function and pretest estimator to the true value are small. The empirical rejection rate is around 0.10 and the WRMSE of the control function estimator and the pretest estimator are smaller than WRSME of the two stage least squares estimator.

Setting (3) of Table 1 presents the WMEAN and WRMSE results for the exponential model (28). The ratio of the bias of two stage least squares, control function and pretest estimators to the true value are small. The empirical rejection rate is low and the WRMSE of the control function estimator and the pretest estimator are smaller than WRSME of the two stage least squares estimator. For β_3, the WRMSE of the CF estimator and the pretest estimator are around 5% of WRMSE of the usual two stage least squares estimator. Even though the assumption 3 of the control function method is violated for the models (27) and (28), the CF estimator is still approximately unbiased and more efficient than the 2SLS estimator. For these settings, the pretest estimator generally uses the CF estimator and is considerably more efficient than the 2SLS estimator.

5.3 Setting where the assumptions of the control function estimator are drastically violated

We now consider a setting in which the control function assumptions are drastically violated. The model setting is as follows:

$$\begin{aligned} y_2 &= -\gamma_2 + \gamma_1 z_2 + \gamma_2 z_2^2 + v_2 \\ w &= \delta v_2^2 + N(0,1) \\ y_1 &= \beta_2 y_2 + \beta_3 y_2^2 + \beta_4 w + u_1 \end{aligned} \tag{29}$$

where $z_2 \sim N(0,1)$ and $u_1, v_2 \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right)$ and $\beta_2 = 1, \beta_3 = 0.2, \beta_4 = 1$ and $\gamma_1 = 1, \gamma_2 = 0.2$ and $\delta = 0.5$.

Setting (4) of Table 1 presents the absolute value of ratio of bias of the sample winsorized mean to the true value of the model (29) and the ratio of the sample winsorized root mean square error to the sample winsorized root mean square error of the two stage least squares estimator.

The ratio of bias of the WMEAN of usual two stage least squares and pretest estimators to the true value are close to zero while the ratio of the bias for the control function estimator is far away from zero. The WRMSE of the control function estimator is much larger than that of the usual two stage least squares estimator. The pretest estimator performs well in terms of sample mean and mean square error because the pretest hypothesis testing rejects most of the control function estimators.

	WMEAN			WRMSE	
	2SLS	Control Function	Pretest	Control Function	Pretest
(1)	Satisfied Model (26)(ERR=0.0510)				
β_2	0.001	0.000	0.000	0.139	0.155
β_3	0.000	0.000	0.000	0.070	0.079
(2)	Cubic Model (27)(ERR=0.1073)				
β_2	0.000	0.000	0.000	0.994	0.996
β_3	0.000	0.000	0.000	0.635	0.736
(3)	Exp Model (28) (ERR=0.0515)				
β_2	0.001	0.001	0.000	0.275	0.311
β_3	0.001	0.000	0.000	0.049	0.053
(4)	Drastically Violated Model (29)(ERR=1.0000)				
β_2	0.000	0.128	0.000	6.900	1.000
β_3	0.000	0.546	0.000	10.275	1.000

Table 1: The ratio of the bias of the sample winsorized mean (WMEAN) to the true value and the ratio of the Winsorized RMSE (WRMSE) of the control function(Pretest) estimator to WRMSE of the two stage least squares estimator for Satisfied Model (26), Cubic Model (27) with SD = 0.5, Exponential Model (28) with SD = 0.5 and Drastically Violated Model (29). The Empirical Rejection Rate (ERR) stands for the proportion (out of 10,000 simulations) of rejection of the null hypothesis that the extra instrumental variable given by the control function method is valid.

5.4 Sensitivity of joint distribution of errors

In this section, we investigate how sensitive the control function estimator is to different joint distributions of (u_1, v_2); in particular, we consider settings where Assumption 3 is satisfied but Assumption 4 is not satisfied. Assume that we use a quadratic model in z_2 to fit the first stage and the true first stage model is quadratic in z_2,

$$\begin{aligned} y_1 &= 1 + z_1 + 10y_2 + 10y_2^2 + u_1; \\ y_2 &= 1 + \frac{1}{8}z_1 + \frac{1}{3}z_2 + \frac{1}{8}z_2^2 + v_2; \end{aligned} \tag{30}$$

where $z_1 \sim N(0,1)$ and $z_2 \sim N(0,1)$, but u_1 and v_2 are not bivariate normal. We generate u_1 and v_2 in the following three ways

1. Double exponential distribution: $u_1 = \epsilon_1$ and $v_2 = \frac{1}{2}\epsilon_1 + \frac{\sqrt{3}}{2}\epsilon_2$ where ϵ_1 and ϵ_2 are independent double exponential distribution with mean 0 and variance 1.

2. Bivariate log-normal distribution: $u_1 = \exp(\epsilon_1)$ and $v_2 = \exp(\epsilon_2)$ where $\epsilon_1, \epsilon_2 \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$.

3. Bivariate absolutely normal distribution: $u_1 = |\epsilon_1| - E|\epsilon_1|$ and $v_2 = |\epsilon_2| - E|\epsilon_2|$ where $\epsilon_1, \epsilon_2 \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$.

	WMEAN			WRMSE	
	2SLS	Control Function	Pretest	Control Function	Pretest
(1)	Double exponential distribution (ERR=0.0520)				
β_2	0.001	0.000	0.000	0.154	0.174
β_3	0.000	0.000	0.000	0.099	0.112
(2)	Log normal distribution(ERR=0.0509)				
β_2	0.006	0.006	0.006	0.077	0.079
β_3	0.001	0.001	0.001	0.060	0.059
(3)	Absolute normal distribution(ERR=0.1455)				
β_2	0.000	0.017	0.011	0.964	1.063
β_3	0.000	0.004	0.002	0.968	1.065

Table 2: The ratio of the bias of the sample winsorized mean (WMEAN) to the true value and the ratio of the Winsorized RMSE (WRMSE) of the control function(Pretest) estimator to WRMSE of the two stage least squares estimator for the model (30) and different joint distributions of errors. For this model, the assumption 4 of the control function estimator to be consistent is violated.

For these settings, Assumption 3 is satisfied but Assumption 4 is not satisfied. Table 2 presents the ratio of the bias of the sample winsorized mean (WMEAN) to the true value and the ratio of the Winsorized RMSE (WRMSE) of the control function(Pretest) estimator to WRMSE of the two stage least squares estimator for the model (30) and different joint distributions of errors. For the double exponential distribution and the log normal distribution, the ratio of the bias of the usual two stage least squares, the control function and the pretest estimator are small; the WRMSE of the control function and the pretest estimator are much smaller than that of the usual two stage least squares estimator. For the absolute normal distribution, the ratio of the bias of the control function and the pretest estimator are slightly larger than that of the usual two stage least squares; the WRMSE of the control function estimator is slightly smaller than that of the usual two stage least squares estimator and that of the pretest estimator is slightly larger than that of the usual two stage least squares estimator. In summary, the control function estimator is robust to the violations of Assumption 4 considered in this section.

6. Application to a field experiment in Burundi

In this section, we apply the control function method and usual two stage least squares to estimate the effect of exposure to violence on a person's patience (conflict on time preference). The data set consists of 302 observations from a field experiment in Burundi (Voors et al (2012)). Burundi underwent a civil war from 1993-2005, where the intensity of the conflict varied in different parts of the country. Voors et al (2012) sought to access whether the exposure to violence that a person experienced, as measured by the percentage dead in attacks in the area the person lived (y_{i2}) affected a person's patience (y_{i1}), as measured by a person's discount rate for willingness to receive larger amounts of money in the future compared to smaller amounts of money now. A smaller value of the discount rate y_{i1} means

	two stage least squares	control function
β_2	-1.282	2.101
(SD)	(7.033)	(3.479)
β_3	0.2360	0.017
(SD)	(0.451)	(0.217)

Table 3: Estimates and Standard Error of Coefficients before Endogenous Variables

the person is more patient. The exogenous variables $\mathbf{x}_i$ that are available are whether the respondent is literate, the respondent's age, the respondent's sex, the total land holding per capita, land Gini coefficient, distance to market, conflict over land, ethnic homogeneity, socioeconomic homogeneity, population density and per capita total expenditure. As discussed in Voors et al (2012), it is possible that the exposure to violence (y_{i2}) is endogenous because violence may be targeted in a non-random way that is related to patience of the community; for example, violence may be targeted to extract "economic profit", i.e., steal the assets of others, where communities which are more vulnerable to having their assets stolen may also differ in their patience levels from less vulnerable communities. Due to possibility of endogeneity, the OLS estimator of regressing the outcome on the observed covariates maybe a biased estimate of the effect of exposure to violence on time preference. We follow the discussion in the paper Voors et al (2012) and use the following instrumental variables, distance to Bujumbura (the capital of Burundi) and altitude. Fighting was more intense near Bujumbura (the capital of Burundi) and at higher altitudes. The assumption for distance to Bujumbura and altitude to be valid instruments is that they only affect the distribution of violence and are not associated with the patience level of a community. Voors et al (2012) defend this assumption by discussing that most plausible concern with its validity is that distance to the capital and altitude might be associated with distance to market which might be associated with patience, but that in Burundi, since most farmers operate at a substance level, selling goods to nearby market, there are local markets in all communities, reducing the possibility of geography being correlated with preference. See more detailed discussion in page 956 in Voors et al (2012). We are able to express our model as

$$y_{1i} = \beta_0 + \beta_1^T \mathbf{x}_i + \beta_2 y_{2i} + \beta_3 y_{2i}^2 + u,$$

and

$$y_{2i} = \alpha_0 + \alpha_1 z_{i1} + \alpha_2 z_{i2} + \alpha_3 z_{i1}^2 + \alpha_4 z_{i2}^2 + \alpha_5 z_{i1} \times z_{i2} + \alpha_6^\intercal \mathbf{x}_i + v,$$

where z_{i1} represents the distance to Bujumbura and z_{i2} represents the altitude for the i-th subject.

Table 3 presents the control function estimate and two stage least squares estimate of β_2 and β_3. Rather than focus directly on β_2 and β_3, we focus on a more interpretable quantity, the increase of discount rate when percentage dead in attacks increase from y_2 to $y_2 + 1$, which we denote by $\delta(y_2)$,

$$\delta(y_2) = y_1^{y_2+1} - y_1^{y_2} = \beta_2 + \beta_3 (1 + 2y_2).$$

In Figure 3$(a)/(b)$, we plot usual two stage least squares/control function point estimates and corresponding confidence intervals of $\delta(y_2)$ respectively. The middle curve represents

the point estimate of $\delta(y_2)$, and the higher curve and lower curve represent the upper and lower bound of 95% confidence interval of $\delta(y_2)$ respectively. In Figure 3(c), we compare the variance of $\delta(y_2)$ provided by control function and usual two stage least squares by plotting the ratio:

$$\frac{Var.2SLS\left(\delta(y_2)\right)}{Var.CF\left(\delta(y_2)\right)},$$

where $Var.2SLS\left(\delta(y_2)\right)$ represents the variance of $\delta(y_2)$ given by usual two stage least squares and $Var.CF\left(\delta(y_2)\right)$ represents the variance of $\delta(y_2)$ given by control function. Figure 3(c) illustrates that the ratio of variance is always larger than one and shows that the control function estimate is more efficient than the usual two stage least squares estimate. We apply the Hausman test to this real data and the corresponding pvalue is around 0.579 so that the validity of the extra instrumental variable is not rejected and the pretest estimator is equal to the control function estimator.

Figure 3(d) demonstrates the predicted causal relationship between the discount rate and percentage dead in attacks. All the other covariates are set at the sample mean level and we plug in the estimates of the coefficients by the control function method and the two stage least square method. As illustrated, we see that the exposure to violence (measured by percentage dead in attacks) decreases patience (measured by a smaller discount rate).

We present two additional applications of the control function method to estimate the effect of household income on food demand in Appendix D and the effect of smoking during pregnancy on birthweight on Appendix E.

7. Discussion

The model (2) assumes constant treatment effect. The comparison between the CF estimator and the 2SLS estimator presented in this paper can be extended to more general class of models that allows for heterogeneous treatment effects model, as discussed in section 5 of Small (2007). Without loss of generality, we consider the outcome model to be a quadratic model.

$$y_{1,i} = \beta_0 + \beta_{1,i} y_{2,i} + \beta_{2,i} y_{2,i}^2 + \beta_3 z_{1,i} + u_{1,i}, \quad \text{for} \quad i = 1, \cdots, n, \tag{31}$$

and assume that $z_{2,i}$ and $z_{2,i}^2$ are valid instruments for $y_{2,i}$ and $y_{2,i}^2$. We introduce the following notation $\beta_1 = \mathbb{E}\beta_{1,i}$ and $\beta_2 = \mathbb{E}\beta_{2,i}$ and assume that

$$(\beta_{1,i}, \beta_{2,i}) \quad \text{is independent of} \quad y_{2,i}, z_{1,i}, z_{2,i}, u_{1,i}, \tag{32}$$

which is a special case of assumption A.3 in Wooldridge (1997) by setting $\rho_1 = 0$. As discussed in Wooldridge (1997) and Small (2007), the assumption (32) states that the units do not select their treatment levels ($y_{2,i}$ and $y_{2,i}^2$) based on the gains that they would experience from the treatment ($\beta_{1,i}$ and $\beta_{2,i}$). The heterogenous effect model (31) can be expressed as

$$y_{1,i} = \beta_0 + \beta_1 y_{2,i} + \beta_2 y_{2,i}^2 + \beta_3 z_{1,i} + \left(u_{1,i} + (\beta_{1,i} - \beta_1) y_{2,i} + (\beta_{2,i} - \beta_2) y_{2,i}^2\right), \quad \text{for} \quad i = 1, \cdots, n.$$

The new error $\left(u_{1,i} + (\beta_{1,i} - \beta_1) y_{2,i} + (\beta_{2,i} - \beta_2) y_{2,i}^2\right)$ has the following property

$$\mathbb{E}\left(u_{1,i} + (\beta_{1,i} - \beta_1) y_{2,i} + (\beta_{2,i} - \beta_2) y_{2,i}^2\right) z_{1,i} = \mathbb{E}\left(u_{1,i} z_{1,i}\right) + \mathbb{E}\left(\beta_{1,i} - \beta_1\right) y_{2,i} z_{1,i} + \mathbb{E}\left(\beta_{2,i} - \beta_2\right) y_{2,i}^2 z_{1,i} = 0$$

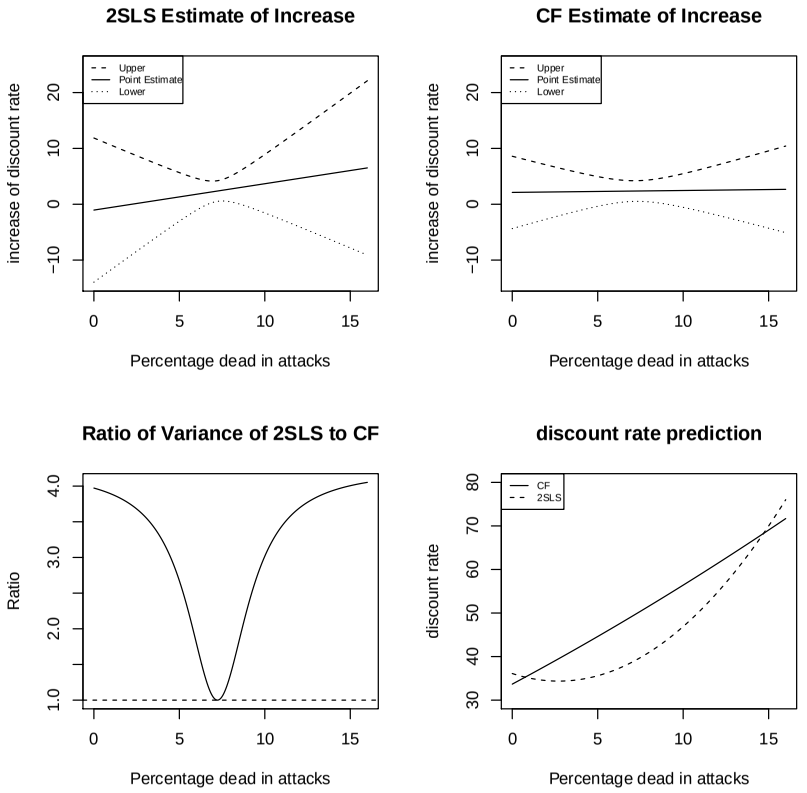


Figure 3: (a) plots the two stage least squares estimate and 95% confidence interval for the increase of discount rate $\delta(y_2)$ with respect to different percentage dead in attacks;(b) plots the control function estimate and 95% confidence interval for the increase of discount rate $\delta(y_2)$ with respect to different percentage dead in attacks;(c) plots the ratio of the variance of two stage least squares to the variance of the control function method with respect to different percentage dead in attacks;(d) plots the discount rate as a function of the percentage dead in attacks with the CF estimate and the 2SLS estimate.

where the first part follows from that $z_{1,i}$ is exogenous and the last two part follows from the assumption (32) . Similarly, we can also show that

$$\mathbb{E}\left(\left(u_{1,i}+\left(\beta_{1,i}-\beta_1\right)y_{2,i}+\left(\beta_{2,i}-\beta_2\right)y_{2,i}^2\right)z_{2,i}\right)=0,$$

and

$$\mathbb{E}\left(\left(u_{1,i}+\left(\beta_{1,i}-\beta_1\right)y_{2,i}+\left(\beta_{2,i}-\beta_2\right)y_{2,i}^2\right)z_{2,i}^2\right)=0.$$

Hence, the instrumental variables are valid for the new error. Our theory for the homogeneous model (2) can be extended to the general (31) under the assumption (32) (it can also be extended under the weaker assumptions than (32) found in Wooldridge (1997)). The pretest estimator provides a consistent estimator and is close to the more efficient estimator between 2SLS and the CF estimator.

We focus on the continuous outcome model in this paper. For a count outcome that follows a log linear model,

$$\log\mathbb{E}\left(y_1|y_2,z\right)=\beta_0+\beta_1y_2+\beta_2z_1+\beta_3u. \tag{33}$$

where

$$y=\alpha_0+\alpha_1z_1+\alpha_2z_2+\delta,$$

and z_1 is the included exogenous variable and z_2 is the instrumental variable and δ is normal variable with zero mean and variance σ^2, Mullahy (1997) shows that the 2SLS estimator is inconsistent while the CF estimator is consistent in this count outcome model. For a binary outcome that follows a logistic regression model, comparison of the CF estimator and the 2SLS estimator for the logistic second stage model can be a further research topic, discussion is found in Cai et al. (2011) and Clarke & Windmeijer (2012).

Imbens & Rubin (1997) formulated the potential outcome model for binary instrumental variable and binary treatment, which can be extended to continuous instrumental variable and treatment as follows: The treatment $y_2=f_c\left(z\right)$ is an increasing function in z given the compliance class c, where $c\in\{1,\cdots,M\}$. We also assume the monotonicity assumption of the compliance class

$$f_{c'}\left(z\right)\leq f_c\left(z\right)\quad\text{for}\quad c'\leq c.$$

The outcome model is

$$y_1|c,z\sim N(f_c(z)+\lambda_c,\sigma^2).$$

As discussed in Imbens & Rubin (1997), the MLE and Bayesian estimator is more efficient than the instrumental variable based method since the instrumental variable method does not take into account the non-negativity of the density function. However, the estimator might be inconsistent if the model is wrong. In this paper, we focus on the semi-parametric model (2), which is robust to the outcome distribution. In the semi-parametric model (2), we show that the CF estimator is equivalent to a 2SLS estimator with an augmented set of instrumental variables. When the extra instrumental variables are valid, we demonstrate by theoretical results and simulation results that the CF estimator is more efficient than the usual 2SLS estimator.

8. Conclusion

This paper compares the control function estimator (Algorithm 2) and the usual two stage least squares estimator (Algorithm 1) for nonlinear causal effect models. Theoretically, we show that the control function estimator is equivalent to a two stage least squares estimator with an augmented set of instrumental variables. When the augmented instrumental variables are valid, the control function method will have a larger set of instrumental variables than the usual two stage least squares estimator and hence the control function estimator will have a smaller variance and root mean square error.

Methodologically, we develop a pretest estimator (Algorithm 3) which tests the validity of the augmented instrumental variables and combines the strength of the control function estimator and the usual two sage least squares estimator. In the simulation study, the pretest estimator provides large gains over two stage least squares in some settings while never being much worse than two stage least squares. Such phenomenon is also observed in the real data analysis.

For practical application, we suggest to use the pretest estimator which means implementing both the two stage least squares estimator and the control function estimator and calculating the Hausman test statistics based on these two estimates. If the Hausman test leads to the rejection of the null, the two stage least squares estimator is more trustworthy; otherwise, we use the control function estimator because it is more efficient and there is no evidence of it being biased.

Appendix A. Consistency of the control function estimator

We now investigate why the control function method works under assumptions 1,2,3 and 4. Define $e = u_1 - \rho v_2$. Since e is the residual of regressing u_1 on v_2, e and v_2 are uncorrelated. By plugging in $u_1 = e + \rho v_2$, we have

$$y_1 = \beta_0 + \beta_1 y_2 + \cdots + \beta_k g_k(y_2) + \beta_{k+1} z_1 + \rho v_2 + e. \tag{34}$$

Since e is uncorrelated with z_1 and v_2, it suffices to show that the new error e is uncorrelated with $g_i(y_2)$. By the law of iterated expectation and the one-to-one correspondence between y_2 and v_2, $E(g_i(y_2)e) = E\left[E(g_i(y_2)e|y_2, z)\right] = E\left[g_i(y_2)E(e|v_2, z)\right]$. By Assumption 3, $E(e|v_2, z) = E(u_1 - \rho v_2|v_2, z) = E(u_1 - \rho v_2|v_2)$. By Assumption 4, $E(u_1 - \rho v_2|v_2) = E(u_1|v_2) - \rho v_2 = 0$. Hence, $E(g_i(y_2)e) = 0$. and the new error e is uncorrelated with $g_i(y_2)$. If we estimate v_2 by its estimate $\hat{v}_2$, the residual of the first stage $y_2 \sim z_1 + z_2 + h_2(z_2) + \cdots + h_k(z_2)$, the consistency of the control function estimate follows if we can verify the regularity conditions in Theorem 1 in Murphy et al (2002), which says that if we obtain the consistent estimates of first stage coefficients, we estimate the second stage coefficients consistently by replacing the residual with its estimate from the first stage. Let Z denote the data matrix where each row is an i.i.d sample of $\{z_1, z_2, h_2(z_2), \cdots, h_k(z_2)\}$, X denote the data matrix where each row is an i.i.d sample of $\{z_1, y_2, g_2(y_2), \cdots, g_k(y_2)\}$ and V_2 denote the data matrix where each row is an i.i.d sample of $\{\hat{v}_2\}$. The regularity conditions for Theorem 1 in Murphy et al (2002) are the following:

(a) $\lim_{n\to\infty} A^T A = Q_0$ where $A = (X, V_2)$ and Q_0 is positive definite matrix.

(b) $v_2 = f(\alpha, Z) = y_2 - (\alpha_0 + \alpha_1 z_1 + \alpha_2 \mathbf{H}(z_2))$ is twice continuously differentiable in α for each Z and $\lim_{n\to\infty} A^T Z = Q_1$.

(c) The first stage estimate $\hat{\alpha}$ is a consistent estimator of α.

We will show that under the following three regularity assumptions, the conditions (a), (b), (c) of Theorem 1 in Murphy et al (2002) are satisfied.

i $\lim_{n\to\infty} A^T A = Q_0$ where $A = (X, V_2)$ and Q_0 is positive definite matrix.

ii $\lim_{n\to\infty} A^T Z = Q_1$.

iii The true model of the first stage is (4).

Assumption i is the same as condition (a). Assumption ii is the second part of condition (b). For the first part of condition (b), since $v_2 = f(\alpha, Z)$ is a linear function of the parameters α, f is twice continuously differentiable. For condition (c), by assuming iii that the true model of the first stage is (4), the first stage regression coefficients $\hat{\alpha}$ of the standard control function estimator (7) are consistent estimates of α.

Appendix B. Proof of the theorems

Lemma 7 *(Adjustment Lemma)*
For the linear model

$$y = \beta_1 x_1 + \beta_2 x_2 + \epsilon,$$

the following two regression give us the same estimate of β_1:

$$y \sim x_1 + x_2,$$

$$y \sim x_{1.2};$$

where $x_{1.2}$ is the residual of the regression $x_1 \sim x_2$.

This lemma is called Adjustment Lemma since we adjust x_1 to x_2 to obtain $x_{1.2}$.
Proof [Proof of Theorem 1] By the adjustment lemma, the coefficients of the regression $y_1 \sim \widetilde{z_1} + \widetilde{y_2} + \widetilde{g_2(y_2)}$ are the same with corresponding coefficients in $y_1 \sim z_1 + y_2 + g_2(y_2) + e_1$ where $\widetilde{x}$ is the residual of the regression $x \sim e_1$. Since z_1 is orthogonal to e_1 by the property of OLS, $\widetilde{z_1} = z_1$. The theorem follows by replacing $\widetilde{z_1}$ with z_1. ■

Proof [Proof of Theorem 2] Let $Z = (1, z_1, z_2, h_2(z_2))$ denote the matrix of observed values of instrumental variables. $y_2^2 = \gamma_c + \gamma e_1 + \widetilde{g_2(y_2)}$ Since e_1 is the residual of the regression

$$y_2 \sim z_1 + z_2 + h_2(z_2),$$

it follows from the property of OLS that $Z^T e_1 = 0_{4\times 1}$. The coefficients given by regression $g_2(y_2) \sim z_1 + z_2 + h_2(z_2)$ are $\left(Z^T Z\right)^{-1} Z^T g_2(y_2)$ and those given by regression $\widetilde{g_2(y_2)} \sim z_1 + z_2 + h_2(z_2)$ are $\left(Z^T Z\right)^{-1} Z^T \widetilde{g_2(y_2)}$. Since

$$\left(Z^T Z\right)^{-1} Z^T g_2(y_2) - \left(Z^T Z\right)^{-1} Z^T \widetilde{g_2(y_2)} = \left(Z^T Z\right)^{-1} Z^T e_1 = 0$$

we come to the conclusion that the regression $g_2(y_2) \sim z_1 + z_2 + h_2(z_2)$ and $\widetilde{g_2(y_2)} \sim z_1 + z_2 + h_2(z_2)$ give the same estimates of coefficients on $1, z_1, z_2, h_2(z_2)$. ■

Proof [Proof of Theorem 3] It is sufficient to show that except for a constant difference,

(1) $\widetilde{y_2}$ is equal to the predicted value of the regression $y_2 \sim z_1 + z_2 + h_2(z_2) + error.iv$.

(2) $\widetilde{g_2(y_2)}$ is equal to the predicted value of the regression $g_2(y_2) \sim z_1 + z_2 + h_2(z_2) + error.iv$.

By the discussion after theorem 2, we have the following decomposition

$$\begin{aligned} g_2(y_2) &= \gamma_c + \gamma e_1 + \widetilde{g_2(y_2)} \\ &= \gamma_c + \gamma e_1 + \gamma_1 z_1 + \gamma_2 z_2 + \gamma_3 h_2(z_2) + error.iv. \end{aligned} \tag{35}$$

where the column vector e_1 is orthogonal to the column vectors $z_2, h_2(z_2)$ and $error.iv$ and $error.iv$ is orthogonal to z_1, z_2 and $h_2(z_2)$ by the property of OLS. By the decomposition (35), we conclude that except for a constant, $\widetilde{g_2(y_2)}$ is equal to the predicted value of the regression $g_2(y_2) \sim z_1 + z_2 + h_2(z_2) + error.iv$. Since $error.iv$ is orthogonal to $z_1, z_2, h_2(z_2)$ and e_1, $error.iv$ is orthogonal to $y_2 = \alpha_0 + \alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 h_2(z_2) + e_1$ and hence the

predicted value of the regression $y_2 \sim z_1 + z_2 + h_2(z_2) + error.iv$ is the same with $\widehat{y_2}$, which is the same with $\widetilde{y_2}$. ■

Proof [Proof of Theorem 4]

It is sufficient to show that except for a constant difference,

(1) $\widetilde{y_2}$ is equal to the predicted value of the regression

$$y_2 \sim z_1 + z_2 + h_2(z_2) + h_3(z_2) + error.iv_1 + error.iv_2.$$

(2) $\widetilde{g_2(y_2)}$ is equal to the predicted value of the regression

$$g_2(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2) + error.iv_1 + error.iv_2.$$

(3) $\widetilde{g_3(y_2)}$ is equal to the predicted value of the regression

$$g_3(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2) + error.iv_1 + error.iv_2.$$

We start with a decomposition of $g_3(y_2)$

$$\begin{aligned} g_3(y_2) &= \gamma_c + \gamma e_1 + \widetilde{g_3(y_2)} \\ &= \gamma_c + \gamma e_1 + \gamma_1 z_1 + \gamma_2 z_2 + \gamma_3 h_2(z_2) + \gamma_4 h_3(z_2) + error.iv_2.prime \\ &= \gamma_c + \gamma e_1 + \gamma_1 z_1 + \gamma_2 z_2 + \gamma_3 h_2(z_2) + \gamma_4 h_3(z_2) + \gamma' error.iv_1 + error.iv_2. \end{aligned} \tag{36}$$

Since e_1 is orthogonal to $error.iv_1$, $error.iv_2$ and $z_1, z_2, h_2(z_2), h_3(z_2)$, the predicted value given by the regression $g_3(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2) + error.iv_1 + error.iv_2$ is the same with $\widetilde{g_3(y_2)}$.

Since $error.iv_2$ is orthogonal to $e_1, z_1, z_2, h_2(z_2), h_3(z_2), error.iv_1$, $error.iv_2$ is orthogonal to $g_2(y_2)$, hence the coefficient of $error.iv_2$ given by the following regression

$$g_2(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2) + error.iv_1 + error.iv_2 \tag{37}$$

is vanishing, which leads to the predicted value of regression (37) is the same with the predicted value of $g_2(y_2) \sim z_1 + z_2 + h_2(z_2) + h_3(z_2) + error.iv_1$ and by the decomposition (35) and same argument in proof of theorem 3, we obtain (2).

Since $error.iv_1$ and $error.iv_2$ are orthogonal to $e_1, z_1, z_2, h_2(z_2), h_3(z_2)$, then $error.iv_1$ and $error.iv_2$ are orthogonal to y_2 and (1) follows.

■

Appendix C. Further results from the simulation study

All the corresponding non-winsorized results are presented in Tables 4 and 5. The observations are similar to the counter part of winsorized results.

	NMEAN			NRMSE	
	2SLS	control function	Pretest	control function	Pretest
(1)	Satisfied Model (26)(ERR=0.0510)				
β_2	0.001	0.000	0.000	0.136	0.543
β_3	0.000	0.000	0.000	0.069	0.535
(2)	Cubic Model (27) (ERR=0.1073)				
β_2	0.000	0.000	0.000	0.993	0.995
β_3	0.000	0.000	0.000	0.619	0.808
(3)	Exp Model (28) (ERR=0.0515)				
β_2	0.002	0.000	0.000	0.236	0.602
β_3	0.001	0.000	0.000	0.040	0.546
(4)	Drastically Violated Model (29)(ERR=1.0000)				
β_2	0.000	0.128	0.000	6.278	1.000
β_3	0.000	0.546	0.000	9.357	1.000

Table 4: The ratio of the bias of the sample non-winsorized mean(NMEAN)to the true value and the ratio of the Non-winsorized RMSE(NRMSE) of the control function(Pretest) estimator to NRMSE of the two stage least squares estimator for Satisfied Model (26), Cubic Model (27) with SD = 0.5, Exponential Model (28) with SD = 0.5 and Drastically Violated Model (29). The Empirical Rejection Rate (ERR) stands for the proportion (out of 10,000 simulations) of rejection of the null hypothesis that the extra instrumental variable given by the control function method is valid.

	WMEAN			WRMSE	
	2SLS	Control Function	Pretest	Control Function	Pretest
(1)	Double exponential distribution(ERR=0.0520)				
β_2	0.001	0.000	0.000	0.150	0.554
β_3	0.000	0.000	0.000	0.096	0.546
(2)	Log normal distribution(ERR=0.0509)				
β_2	0.002	0.006	0.009	0.003	0.310
β_3	0.000	0.001	0.001	0.002	0.299
(3)	Absolute normal distribution(ERR=0.1455)				
β_2	0.000	0.017	0.010	0.881	1.027
β_3	0.000	0.004	0.002	0.883	1.029

Table 5: The ratio of the bias of the sample non-winsorized mean(NMEAN)to the true value and the ratio of the Non-winsorized RMSE(NRMSE) of the control function(Pretest) estimator to NRMSE of the two stage least squares estimator for the model (30) and different joint distributions of errors. For this model, the assumption 4 of the control function estimator to be consistent is violated.

	two stage least squares	control function
β_2	39.993	7.389
(SD)	(22.353)	(3.728)
β_3	-2.397	1.592
(SD)	(2.730)	(0.431)

Table 6: Estimates and Standard Error of Coefficients before Endogenous Variables

Appendix D. Application to demand for food

In this section, we apply the control function method and usual two stage least squares to estimate the effect of income on food demand and compare these two methods. The data set is from Bouis & Haddad (1990) and comes from a survey of farm households in the Bukidnon Province of Philippines. Following Bouis & Haddad (1990), we assume that food expenditure is a quadratic function of log income. Here, y_{1i} is the expenditure on food of the i-th household, y_{2i} is the log of income of the i-th household and x_i represents the included exogenous variables of the i-th household, which consist of mother's education, father's education, mother's age, father's age, mother's nutritional knowledge, corn price, rice price , population density of the municipality, number of household members expressed in adult equivalents, and dummy variables for the round of the survey. Then we are able to express our model as $y_{1i} = \beta_0 + \beta_1 x_i + \beta_2 y_{2i} + \beta_3 y_{2i}^2 + u$. Bouis & Haddad (1990) were concerned that regression of y_1 on y_2 (log income) and x would not provide an unbiased estimate of β because of unmeasured confounding variables. In particular, because farm households make production and consumption decisions simultaneously and there are multiple incomplete markets in the study area, the households' production decisions (which affect their log income y_2) are associated with their preferences according to microeconomic theory (Bardhan & Udry (1999), chap. 2). To solve the problem, Bouis & Haddad (1990) proposed cultivated area per capita as an instrumental variable. Bouis and Haddad's reasoning for why cultivated area per capita is an instrumental variable is that "land availability is assumed to be a constraint in the short run, and therefore exogenous to the household decision making process."

Table 6 presents the control function estimate and two stage least squares estimate of β_2 and β_3. Rather than focus directly on β_2 and β_3, we focus on a more interpretable quantity, the income elasticity of food demand at the mean level of food expenditure. This is the percent change in food expenditure caused by a 1% increase in income for households currently spending at the mean food expenditure level, and we denote it by $\eta(y_2)$. The mean food expenditure of households is 31.14 pesos per capita per week and

$$\eta(y_2) = \frac{100}{31.14}\hat{\beta}_2 \log(1.01) + \hat{\beta}_3 \left(2\log(1.01)y_2 + \log(1.01)^2\right).$$

In Figure 4(a), we plot OLS point estimates and corresponding confidence intervals of $\eta(y_2)$ with respect to different levels of log income y_2. The middle curve represents the point estimate of $\eta(y_2)$, and the higher curve and lower curve represent the upper and lower bound of 95% confidence interval of $\eta(y_2)$ respectively. In Figure 4$(b)/(c)$, we plot usual two stage least squares/control function point estimates and corresponding confidence intervals

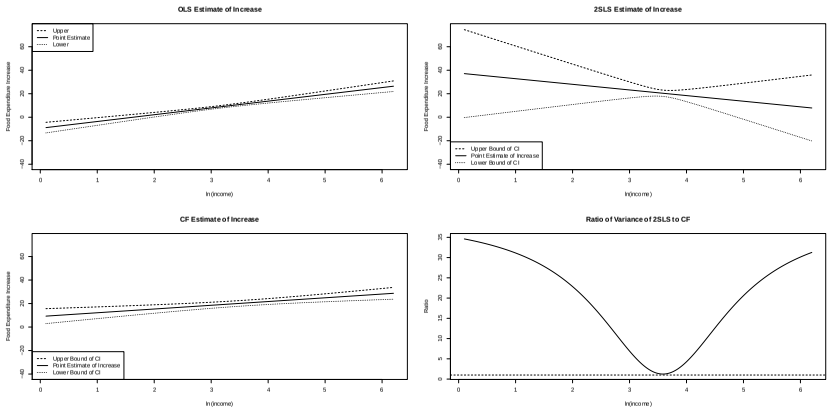


Figure 4: (a) plots the OLS estimate and 95% confidence interval for the income elasticity with respect to different log(income);(b) plots the two stage least squares estimate and 95% confidence interval for the income elasticity with respect to different log(income);(c) plots the control function estimate and 95% confidence interval for the income elasticity with respect to different log(income);(d) plots the ratio of the variance of two stage least squares to the variance of the control function method with respect to different log(income).

of $\eta(y_2)$ respectively. In Figure 4(d), we compare the variance of $\eta(y_2)$ provided by control function and usual two stage least squares by plotting the ratio:

$$\frac{Var.2SLS\left(\eta(y_2)\right)}{Var.CF\left(\eta(y_2)\right)},$$

where $Var.2SLS\left(\eta(y_2)\right)$ represents the variance of $\eta(y_2)$ given by usual two stage least squares and $Var.CF\left(\eta(y_2)\right)$ represents the variance of $\eta(y_2)$ given by control function. Figure 4(b) shows that two stage least squares leads to negative estimate of the non-negative valued Income Elasticity of Food Demand; in contrast, Figure 4(c) shows that the control function estimate of the non-negative valued Income Elasticity of Food Demand is always positive. Figure 4(d) illustrates that the ratio of variance is always larger than one and shows that the control function estimate is more efficient than the usual two stage least squares estimate. We apply the Hausman test to this real data and the corresponding pvalue is around 0.14 so that the validity of the extra instrumental variables is not rejected and the pretest estimator is equal to the control function estimator.

Appendix E. Application to smoking data

In this section, we apply the control function method and usual two stage least squares to estimate the effect of maternal smoking during pregnancy on birth weight. The data set consists of 1388 observations from the Child Health Supplement to the 1998 National Health

	two stage least squares	control function
$\beta_2(\times 10^{-3})$	-12.535	-8.956
$(\mathrm{SD}(\times 10^{-3}))$	(7.463)	(3.040)
$\beta_3(\times 10^{-3})$	0.266	0.121
$(\mathrm{SD}(\times 10^{-3}))$	(0.284)	(0.068)

Table 7: Estimates and Standard Error of Coefficients before Endogenous Variables

Interview Survey. The goal is to estimate the effect of maternal smoking (during pregnancy) on the birth weight. For the i-th observation, the outcome y_{i1} represents the log birth weight, y_{i2} represents the typical number of cigarettes smoked per day during pregnancy and $\mathbf{x}_i$ represents the other exogenous variables: birth order, race and child's sex. As discussed in Wehby et al. (2011), it is possible that mothers who smoke during pregnancy self-select into smoking based on their preferences for health and risk taking and their perceptions of fetal health endowments. These factors, which are typically unobserved in available data samples, can affect the fetal health through other pathways besides smoking. For example, women smoking during pregnancy might adopt other unhealthy behaviors having adverse effects on the fetus (e.g., poorer nutrition or reduced prenatal care). Since some of the confounding variables that are related with both smoking and child health are unobserved, the OLS estimator of regressing the outcome on the observed covariates maybe a biased estimate of the effect of smoking on the birth weight. We follow the discussion in the paper Mullahy (1997) and use the following instrumental variables $\mathbf{z}_i$: maternal schooling, paternal schooling, family income and the per-pack state excise tax on the cigarettes. Then we are able to express our model as

$$y_{1i} = \beta_0 + \beta_1^T \mathbf{x}_i + \beta_2 y_{2i} + \beta_3 y_{2i}^2 + u,$$

and

$$y_{2i} = \alpha^T \mathbf{z}_i + v.$$

Table 7 presents the control function estimate and two stage least squares estimate of β_2 and β_3. Rather than focus directly on β_2 and β_3, we focus on a more interpretable quantity, the increase of log birthweight when the cigarettes increase from y_2 to $y_2 + 1$, which we denote by $\delta(y_2)$,

$$\delta(y_2) = y_1^{y_2+1} - y_1^{y_2} = \beta_2 + \beta_3 (1 + 2y_2).$$

In Figure 5$(a)/(b)$, we plot usual two stage least squares/control function point estimates and corresponding confidence intervals of $\delta(y_2)$ respectively. The middle curve represents the point estimate of $\delta(y_2)$, and the higher curve and lower curve represent the upper and lower bound of 95% confidence interval of $\delta(y_2)$ respectively. In Figure 5(c), we compare the variance of $\delta(y_2)$ provided by control function and usual two stage least squares by plotting the ratio:

$$\frac{Var.2SLS\left(\delta(y_2)\right)}{Var.CF\left(\delta(y_2)\right)},$$

where $Var.2SLS\left(\delta(y_2)\right)$ represents the variance of $\delta(y_2)$ given by usual two stage least squares and $Var.CF\left(\delta(y_2)\right)$ represents the variance of $\delta(y_2)$ given by control function.

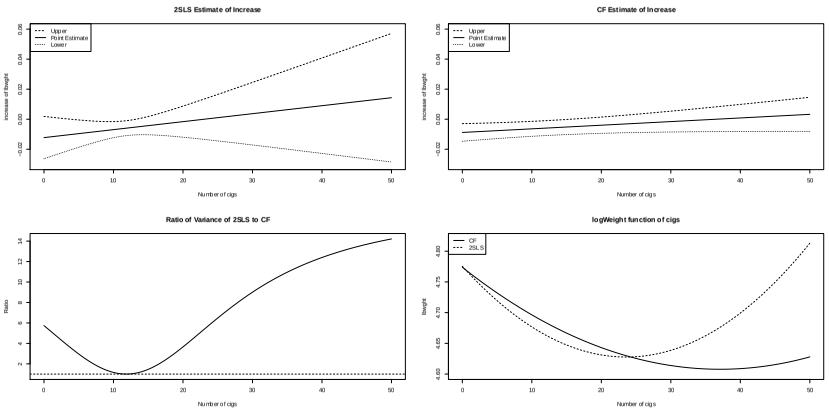


Figure 5: (a) plots the two stage least squares estimate and 95% confidence interval for the increase of lbwght $\delta(y_2)$ with respect to different number of cigarettes;(b) plots the control function estimate and 95% confidence interval for the increase of lbwght $\delta(y_2)$ with respect to different number of cigarettes;(c) plots the ratio of the variance of two stage least squares to the variance of the control function method with respect to different number of cigarettes;(d) plots the lbwght as a function of the number of cigarettes with the CF estimate and the 2SLS estimate.

Figure 5(c) illustrates that the ratio of variance is always larger than one and shows that the control function estimate is more efficient than the usual two stage least squares estimate. We apply the Hausman test to this real data and the corresponding pvalue is around 0.599 so that the validity of the extra instrumental variables is not rejected and the pretest estimator is equal to the control function estimator.

Figure 5(d) demonstrates the predicted causal relationship between the log birthweight and number of cigarettes smoked. All the other covariates are set at the sample mean level and we plug in the estimates of the coefficients by the control function method and the two stage least square method. As illustrated, we see that smoking during pregnancy has a negative effect on birthweight; with the number of cigarettes increasing, the absolute value of the marginal effect is decreasing, that is to say, the effect of additional cigarettes smoked becomes less.

References

Angrist, J. D., Imbens, G. W. & Rubin,D.R.(1996). Identification of causal effects using instrumental variables. *Journal of the American statistical Association*,**91**,444–455.

Ahn, H. & Powell, J. L. (2004). Semiparametric estimation of censored selection models with a nonparametric selection mechanism. *Journal of Econometrics* **58**, 3–29.

Andrews, D.W.K. & Schafgans, M. (1998). Semiparametric estimation of the intercept of a sample selection model. *The Review of Economic Studies* **65**, 497.

Baiocchi, M., Cheng, J. & Small, D.S.(2014). Tutorial in biostatistics: Instrumental variable methods for causal inference *Statistics in Medicine*,**33**, 2297-2340.

Bardhan, P. & Udry, C.(1999). Development microeconomics, *Oxford University Press*,1999.

Blundell, R. & Powell, J. L. (2004). Endogeneity in nonparametric and semiparametric regression models. *Econometric society monographs* **36**, 312–357.

Bouis, H. E. & Haddad,L. J. (1990). Effects of agricultural commercialization on land tenure, household resource allocation, and nutrition in the Philippines. *Intl Food Policy Res Inst.*

Brookhart, A.M., Rassen, A.J. & Schneeweiss, S.(2010). Instrumental variable methods in comparative safety and effectiveness research. *Pharmacoepidemiology and Drug Safety*,**19**, 537 554.

Cai, B., Small, S.D. & Have, T.(2011). Two-stage instrumental variable methods for estimating the causal odds ratio: Analysis of bias *Statistics in medicine*,**30**, 1809-1824.

Card, D. (1993). Using geographic variation in college proximity to estimate the return to schooling. *National Bureau of Economic Research.*

Card, D. (1994). Earnings, schooling, and ability revisited. *National Bureau of Economic Research.*

Cheng, J., Qin, J. & Zhang, B.(2009). Semiparametric estimation and inference for distributional and general treatment effects. *Journal of the Royal Statistical Society: Series B (Methodology)*,**71**, 881–904.

Clarke, P.S., & Windmeijer, F.(2012). Instrumental variable estimators for binary outcomes. *Journal of the American Statistical Association*,**107**, 1638–1652.

Gawande, K. & Bandyopadhyay,U. (2000). Is protection for sale? Evidence on the Grossman-Helpman theory of endogenous protection *Review of Economics and Statistics* **82**, 139–152.

Hausman, J. A. (1978). Specification tests in econometrics. *Econometrica: Journal of the Econometric Society.* **46**,1251–1271.

Heckman, J. J. (1976). The common structure of statistical models of truncation, sample selection and limited dependent variables and a simple estimator for such models. NBER.

Henrich, J., Boyd, R., Bowles, S., Camerer, C., Fehr, E., Gintis, H., McElreath, R. (2001). *In search of homo economicus: behavioral experiments in 15 small-scale societies. American Economic Review* **97**, 73–78.

Hernan, M. A. & Robins, J. M. (2006). Instruments for causal inference: *An Epidemiologist's Dream?. Epidemiology* **17**, 360–372.

Holland, P.W. (1988). Causal inference, path analysis and recursive structural equations models *Sociological Methodology* **18**, 449–484.

Hughes, J. P. & Mester, L. J. (2008). Bank capitalization and cost: evidence of scale economies in risk management and signaling *Review of Economics and Statistics* **80**, 314–325.

Imbens, G. W.(2014). Instrumental Variables: An Econometricians Perspective. *Statistical Science.*

Imbens G. W. & Rubin B.D. (2007). Estimating outcome distributions for compilers in instrumental variables models. *Review of Economic Studies*, 1997, **64**, 555–574.

Imbens G. W. & Wooldridge.M. (2007). Control Function and Related Methods. NBER SummerInstitute.

Kelejian, H. H.(1971). Two-stage least squares and econometric systems linear in parameters but nonlinear in the endogenous variables *J. Am. Statist. Assoc.* **66**, 373–374.

Mullahy, J. (1997). *Instrumental-variable estimation of count data models: Applications to models of cigarette smoking behavior. Review of Economics and Statistics*, **79**, 586–593.

Murphy, K. M & Topel, R. H. (2002). *Estimation and inference in two-step econometric models. Journal of Business & Economic Statistics.*, 88–97.

Nagelkerke, N., Fidler, V., Bernsen, R., & Borgdorff, M. (2000). Estimating treatment effects in randomized clinical trials in the presence of non-compliance *Statistics in Medicine* **19**, 1849–1864.

Nedelman, J.R., Rubin, D.B., & Sheiner, L.B.(2007). Diagnostics for confounding in PK/PD models for oxcarbazepine *Statistics in Medicine* **26**, 290–308.

Neyman, J. (trans. Dabrowska D. and Speed T. P)(1990). On the Application of Probability Theory to Agricultural Experiments *Statistical Science* **5**, 463–480.

Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies.. *Journal of educational Psychology*, **66**, 688-701.

Small, D. S. (2007) Sensitivity analysis for instrumental variables regression with overidentifying restrictions *J. Am. Statist. Assoc.* **102**, 1049–1058.

Stock, J. (2002). Instrumental variables in economics and statistics *International encyclopedia of the social sciences.*

Tan, Z.(2006). Regression and weighting methods for causal inference using instrumental variables *J. Am. Statist. Assoc.* **101**, 1607–1618.

Terza, J. V., Basu, A. & Rathouz, P. J. (2008). Two-stage residual inclusion estimation: addressing endogeneity in health econometric modeling *Journal of Health Economics* **27**, 531–543.

Voors, M. J, Nillesen, E. EM, Verwimp, P., Bulte, E. H, Lensink, R. & Van Soest, D. P. (2012). *Violent conflict and behavior: a field experiment in Burundi. The American Economic Review*, **102**, 941–964.

Wehby, G. L, Fletcher, J. M,Lehrer, S. F,Moreno, L. M,Murray, J. C,Wilcox, A & Lie, R. T.(2011). A genetic instrumental variables analysis of the effects of prenatal smoking on birth weight: evidence from two samples. *Biodemography and social biology* **57**, 3–32.

White, H. (1984). Asymptotic theory for econometricians. Academic Press, New York.

Wilcox, R. R. & Keselman, H. J. (2003). Modern robust data analysis methods: measures of central tendency. *Psychological methods.* **8**, 254.

Wooldridge, J. M. (1997). On two stage least squares estimation of the average treatment effect in a random coefficient model. *Economics Letters*, **56**, 129–133.

Wooldridge, J. M. (2010). Econometric analysis of cross section and panel data. *MIT press.*

Journal of Machine Learning Research 17 (2016) 1-54 Submitted 12/14; Revised 11/15; Published 6/16

Structure Learning in Bayesian Networks of a Moderate Size by Efficient Sampling

Ru He HRHERU@GMAIL.COM
Department of Computer Science and Department of Statistics
Iowa State University
Ames, IA 50011, USA

Jin Tian JTIAN@IASTATE.EDU
Department of Computer Science
Iowa State University
Ames, IA 50011, USA

Huaiqing Wu ISUHWU@IASTATE.EDU
Department of Statistics
Iowa State University
Ames, IA 50011, USA

Editor: Max Chickering

Abstract

We study the Bayesian model averaging approach to learning Bayesian network structures (DAGs) from data. We develop new algorithms including the first algorithm that is able to efficiently sample DAGs of a moderate size (with up to about 25 variables) according to the exact structure posterior. The DAG samples can then be used to construct estimators for the posterior of any feature. We theoretically prove good properties of our estimators and empirically show that our estimators considerably outperform the estimators from the previous state-of-the-art methods.

Keywords: Bayesian model averaging, Bayesian networks, DAG sampling, dynamic programming, order sampling, structure learning

1. Introduction

Bayesian networks are graphical representations of multivariate joint probability distributions and have been widely used in various data-mining tasks for probabilistic inference and causal modeling (Pearl, 2000; Spirtes et al., 2001). The core of a Bayesian network (BN) representation is its Bayesian network structure. A Bayesian network structure is a DAG (directed acyclic graph) whose nodes represent the random variables $X_1, X_2, \cdots, X_n$ in the problem domain and whose edges correspond to the direct probabilistic dependencies. Semantically, a Bayesian network structure G encodes a set of conditional independence assumptions: for each variable (node) X_i, X_i is conditionally independent of its non-descendants given its parents. With the above semantics, a Bayesian network structure provides a compact representation for joint distributions and supports efficient algorithms for answering probabilistic queries. Furthermore, with its semantics, a Bayesian network structure can often provide a deep insight into the problem domain and open the door to the cause-and-effect analysis.

In the last two decades, there have been a large number of research articles focusing on the problem of learning Bayesian network structure(s) from the data. These articles deal with a common real situation where the underlying Bayesian network is typically unknown so that it has to be learned from the observed data. One motivation for the structure learning is to use the learned structure for inference or decision making. For example, we can use the learned model to predict or classify a new instance of data. Another structure-learning motivation, which is more closely related to the semantics of Bayesian network structures, is for discovering the structure of the problem domain. For example, in the context of biological expression data, the discovery of the causal and dependence relation among different genes is often of primary interests. With the semantics of a Bayesian network structure G, the existence of an edge from node X to node Y in G can be interpreted as the fact that variable X directly influences variable Y; the existence of a directed path from node X to node Y can be interpreted as the fact that X eventually influences Y. Furthermore, under certain assumptions (Heckerman et al., 1999; Spirtes et al., 2001), the existence of a directed path from node X to node Y indicates that X causes Y. Thus, with the learned Bayesian network structure, we can answer interesting questions such as whether gene X controls gene Y which in turn controls gene Z by examining whether there is a directed path from node X via node Y to node Z in the learned structure. Just as mentioned by Friedman and Koller (2003), the extraction of these kinds of interesting structural features is often the primary goal in the discovery task.

There are several general approaches to learning BN structures. One approach is to treat learning BN structures as a model-selection problem. This approach defines a scoring criterion that measures how well a BN structure (DAG) fits the data and finds the DAG (or a set of equivalent DAGs) with the optimal score (Silander and Myllymaki, 2006; Jaakkola et al., 2010; Yuan et al., 2011; Malone et al., 2011a,b; Cussens, 2011; Yuan and Malone, 2012; Malone and Yuan, 2013; Cussens and Bartlett, 2013; Yuan and Malone, 2013). (In Bayesian approach, the score of a DAG G is simply the posterior $p(G|D)$ of G given data D.) When the data size is small as compared with the number of variables, however, the posterior $p(G|D)$ often gives significant support to a number of DAGs, and using a single maximum-a-posteriori (MAP) model could lead to unwarranted conclusions (Friedman and Koller, 2003). It is therefore desirable to use the Bayesian model averaging approach by which the posterior probability of any feature of interest is computed by averaging over all the possible DAGs (Heckerman et al., 1999).

Bayesian model averaging is, however, computationally challenging because the number of possible network structures is between $2^{n(n-1)/2}$ and $n!2^{n(n-1)/2}$, super-exponential in the number of variables n. Tractable algorithms have been developed for special cases of averaging over trees (Meila and Jaakkola, 2006) and averaging over DAGs given a node ordering (Dash and Cooper, 2004). Since 2004, dynamic programming (DP) algorithms have been developed for computing exact posterior probabilities of structural features such as edges or subnetworks (Koivisto and Sood, 2004; Koivisto, 2006; Tian and He, 2009). These algorithms have exponential time and space complexity and are capable of handling Bayesian networks of a moderate size with up to around 25 variables (mainly because of their space cost $O(n2^n)$). A big limitation of these algorithms is that they can only compute posteriors of modular features such as an edge but cannot compute non-modular features such as a path ("is there a path from node X to node Y"), a combined path ("is there a path from node X via node Y to node Z" or "is there a path from node X to node Y and no path from node X to node Z"), or a limited-length path ("is there a path of length at most 3 from node X to node Y"). Recently, Parviainen and Koivisto (2011) developed a DP algorithm that can compute the exact posterior probability of a path feature under a certain assumption. (The assump-

tion, called the order-modular assumption, will be discussed in details soon.) This DP algorithm has (even higher) exponential time and space complexity and can only handle a Bayesian network with fewer than 20 variables (mainly because of its space cost $O(3^n)$). Because this DP algorithm can only deal with a path feature, all the other non-modular features (such as a combined path) which would interest various users still cannot be computed by any DP algorithm proposed so far. Note that generally the posterior $p(f|D)$ of a combined feature $f = (f_1, f_2, \ldots, f_J)$ cannot be obtained only from the posterior of each individual feature $p(f_j|D)$ ($j \in \{1, 2, \ldots, J\}$), because the independence among these features does not hold generally. Actually, by comparing $p(f_2|f_1, D)$ with $p(f_2|D)$, a user can know the effect of the feature f_1 upon the feature f_2; but to obtain $p(f_2|f_1, D)$ $(= p(f_1, f_2|D)/p(f_1|D))$, the user typically needs to obtain $p(f_1, f_2|D)$ first. Another limitation of all these DP algorithms is that it is very expensive for them to perform data prediction tasks. They can compute the exact posterior of a new observational data case $p(x|D)$ but the algorithms have to be re-run for each new data case x.

One solution to computing the posterior of an arbitrary non-modular feature is drawing DAG samples $\{G_1, \ldots, G_T\}$ from the posterior $p(G|D)$, which can then be used to approximate the full Bayesian model averaging by estimating the posterior of an arbitrary feature f as $p(f|D)$ $\approx \frac{1}{T}\sum_{i=1}^{T} f(G_i)$, or the posterior predictive distribution as $p(x|D) \approx \frac{1}{T}\sum_{i=1}^{T} p(x|G_i)$. A number of algorithms have been developed for drawing sample DAGs using the bootstrap technique (Friedman et al., 1999) or the Markov chain Monte Carlo (MCMC) techniques (Madigan and York, 1995; Friedman and Koller, 2003; Eaton and Murphy, 2007; Grzegorczyk and Husmeier, 2008; Niinimaki et al., 2011; Niinimaki and Koivisto, 2013). Madigan and York (1995) developed the *Structure MCMC* algorithm that uses the Metropolis-Hastings algorithm in the space of DAGs. Friedman and Koller (2003) developed the *Order MCMC* procedure that operates in the space of orders. The Order MCMC was shown to be able to considerably improve over the Structure MCMC the mixing and convergence of the Markov chain and to outperform the bootstrap approach of Friedman et al. (1999) as well. Eaton and Murphy (2007) developed the *Hybrid MCMC* method (that is, DP+MCMC method) that first runs the DP algorithm of Koivisto (2006) to develop a global proposal distribution and then runs the MCMC phase in the DAG space. Their experiments showed that the Hybrid MCMC converged faster than both the Structure MCMC and the Order MCMC, so that the Hybrid MCMC resulted in more accurate structure-learning performance. An improved MCMC algorithm (often denoted as *REV-MCMC*) traversing in the DAG space with the addition of a new edge-reversal move was developed by Grzegorczyk and Husmeier (2008) and was shown to be superior to the Structure MCMC and nearly as efficient as the Order MCMC in the mixing and convergence. Recently, Niinimaki et al. (2011) proposed the *Partial Order MCMC* method which operates in the space of partial orders. The Partial Order MCMC includes the Order MCMC as its special case (by setting the parameter bucket size b to be 1) and has been shown to be superior to the Order MCMC in terms of the mixing and the structure-learning performance when a more appropriate bucket size $b > 1$ is set. One common drawback of these MCMC algorithms is that there is no guarantee on the quality of the approximation in finite runs. The approach to approximating the full Bayesian model averaging using the K*-best* Bayesian network structures was studied by Tian et al. (2010) and was shown to be competitive with the Hybrid MCMC.

Several of these state-of-the-art algorithms work in the order space, including the exact algorithms (Koivisto and Sood, 2004; Koivisto, 2006; Parviainen and Koivisto, 2011) and the approximate algorithms: the Order MCMC (Friedman and Koller, 2003) and the Partial Order MCMC (Niinimaki et al., 2011). They all assume a special form of the structure prior, termed as the *order-modular* prior (Friedman and Koller, 2003; Koivisto and Sood, 2004), for computational convenience. (Please refer to the beginning of Section 2.1 for the definition of the order-modular prior.) Under the assumption of the order-modular prior, however, the corresponding prior $p(G)$ cannot represent some desirable priors such as a uniform prior over the DAG space [1]; the computed posterior probabilities are biased because a DAG that has a larger number of topological orders will be assigned a larger prior probability. Whether a computed posterior with the bias from the order-modular prior is inferior to its counterpart without such a bias depends on the application scenario and is beyond the scope of this paper. For the detailed discussion about this issue, please see the related papers (Friedman and Koller, 2003; Grzegorczyk and Husmeier, 2008; Parviainen and Koivisto, 2011) [2]. One method that helps the Order MCMC (Friedman and Koller, 2003) to correct this bias was proposed by Ellis and Wong (2008).

In this paper, first we develop a new algorithm that uses the results of the DP algorithm of Koivisto and Sood (2004) to efficiently sample orders according to the *exact* order posterior under the assumption of the order-modular prior. Next, we develop a time-saving strategy for the process of sampling DAGs consistent with given orders. (Such a DAG-sampling process is based on sampling parents for each node as described by Friedman and Koller, 2003 by assuming a bounded node in-degree.) The resulting algorithm (called DDS) is the first algorithm that is able to sample DAGs according to the *exact* DAG posterior with the same order-modular prior assumption. We empirically show that our DDS algorithm is both considerably more accurate and considerably more efficient than the Order MCMC and the Partial Order MCMC when n is moderate so that our DDS algorithm is applicable. Moreover, the estimator based on our DDS algorithm has several desirable properties; for example, unlike the existing MCMC algorithms, the quality of our estimator can be guaranteed by controlling the number of DAGs sampled by our DDS algorithm. The main application of our DDS algorithm is to address the limitation of the exact DP algorithms (Koivisto and Sood, 2004; Koivisto, 2006; Parviainen and Koivisto, 2011) (whose usage is restricted to modular features or path features) in order to estimate the posteriors of various *non-modular* features arbitrarily specified by users. Additionally our DDS algorithm can also be used to efficiently perform data prediction tasks in estimating $p(x|D)$ for a large number of data cases (while the exact DP algorithm has to be re-run for each data case x). Finally, we develop an algorithm (called IW-DDS) to correct the bias (due to the order-modular prior) in the DDS algorithm by extending the idea of Ellis and Wong (2008). We theoretically prove that the estimator based on our IW-DDS has several desirable properties; then we empirically show that our estimator is superior to the estimators based on the Hybrid MCMC method (Eaton and Murphy, 2007) and the K-best algorithm (Tian et al., 2010), two state-of-the-art algorithms that can estimate the posterior of any feature without the order-modular prior assumption. Analogously, our IW-DDS algorithm mainly addresses the limitation of the exact DP algorithm of Tian and He (2009) (whose usage is restricted to modular features) in order to estimate the posteriors of arbitrary *non-modular* features and can additionally be used to efficiently perform data prediction tasks when an application situation prefers to avoid the bias from the order-modular prior.

1. A uniform prior over the DAG space can be represented in another form of the structure prior, termed as the *structure-modular* prior. The definition of the structure-modular prior will be given in Eq. (2) in Section 2.
2. Although a lot of discussions are about the comparison of these two kinds of priors (the order-modular prior and the structure-modular prior), there exist other kinds of priors such as a uniform prior over Markov equivalence classes. Our paper, however, will focus on learning structural features under either the order-modular prior or the structure-modular prior, because these are the two most commonly used priors in applications.

The rest of the paper is organized as follows. In Section 2 we briefly review the Bayesian approach to learning Bayesian networks from data, the related DP algorithms (Koivisto and Sood, 2004; Koivisto, 2006), and the Order MCMC algorithm (Friedman and Koller, 2003). In Section 3 we present our order sampling algorithm, DDS algorithm, and IW-DDS algorithm, and prove good properties of the estimators based on our algorithms. We empirically demonstrate the advantages of our algorithms in Section 4 and conclude the paper in Section 5. Finally, Appendix A provides the proofs of all the conclusions including the propositions, theorems, and corollary referenced in the paper.

2. Bayesian Learning of Bayesian Network Structures

A Bayesian network structure is a DAG G that provides the skeleton for compactly encoding a joint probability distribution over a set $X = \{X_1, \ldots, X_n\}$ of random variables with each node of the DAG representing a variable in X. For convenience we typically work on the index set $V = \{1, \ldots, n\}$ and represent a variable X_i by its index i. We use $X_{Pa_i} \subseteq X$ to represent the parent set of X_i in a DAG G and use $Pa_i \subseteq V$ to represent the corresponding index set. (A pair (i, Pa_i) is often called a family.) Thus, a DAG G can be represented as a vector $(Pa_1, \ldots, Pa_n)$.

Assume that we are given a training data set $D = \{x^1, x^2, \ldots, x^m\}$, where each x^i is a particular instantiation over the set of variables X. We only consider situations where the data are complete, that is, every variable in X is assigned a value. In the Bayesian approach to learning Bayesian networks from the training data D, we compute the posterior probability of a DAG G as

$$p(G|D) = \frac{p(D|G)p(G)}{p(D)} = \frac{p(D|G)p(G)}{\sum_G p(D|G)p(G)}.$$

Assuming global and local parameter independence, and parameter modularity, we can decompose $p(D|G)$ into a product of local marginal likelihoods (often called local scores) as (Cooper and Herskovits, 1992; Heckerman et al., 1995)

$$p(D|G) = \prod_{i=1}^{n} p(X_i|X_{Pa_i} : D) := \prod_{i=1}^{n} score_i(Pa_i : D), \tag{1}$$

where, with appropriate parameter priors, $score_i(Pa_i : D)$ (the local score for a family (i, Pa_i)) has a closed-form solution. In this paper we will assume that these local scores can be computed efficiently from data. The standard assumption for the structure prior $p(G)$ is the *structure-modular* prior (Friedman and Koller, 2003)

$$p(G) = \prod_{i=1}^{n} p_i(Pa_i), \tag{2}$$

where p_i is some nonnegative function over the subsets of $V - \{i\}$.

Combining Eq. (1) and Eq. (2), we have

$$p_{\not\prec}(G, D) = p(D|G)p_{\not\prec}(G) = \prod_{i=1}^{n} score_i(Pa_i : D)p_i(Pa_i). \tag{3}$$

Note that the subscript $\not\prec$ is intentionally added by us to mean that each correspondingly marked probability is the one obtained under the structure-modular prior instead of the order-modular prior. This is different from the probability obtained under the order-modular prior, which will be marked by the subscript $\prec$ for the distinction.

We can compute the posterior probability of any feature of interest by averaging over all the possible DAGs. For example, we are often interested in computing the posteriors of structural features. Let f be a structural feature represented by an indicator function such that $f(G)$ is 1 if the feature is present in G and 0 otherwise. By the full Bayesian model averaging, we have the posterior of f as

$$p(f|D) = \sum_G f(G)p(G|D). \tag{4}$$

Note that $p_{\not\prec}(f|D)$ will be obtained if $p(G|D)$ in Eq. (4) is $p_{\not\prec}(G|D)$; $p_{\prec}(f|D)$ will be obtained if $p(G|D)$ in Eq. (4) is $p_{\prec}(G|D)$. This difference is the key to understanding the bias issue which will be described in details later.

Because directly summing over all the possible DAGs by brute force is generally infeasible for any problem with $n > 6$ using a contemporary computer, one approach to computing the posterior of f is to draw DAG samples $\{G_1, \ldots, G_T\}$ from the posterior $p_{\not\prec}(G|D)$ or $p_{\prec}(G|D)$, which can then be used to estimate the posterior $p_{\not\prec}(f|D)$ or $p_{\prec}(f|D)$ as

$$\hat{p}(f|D) = \frac{1}{T}\sum_{i=1}^{T} f(G_i). \tag{5}$$

2.1 The DP Algorithms

The DP algorithms (Koivisto and Sood, 2004; Koivisto, 2006) work in the order space rather than the DAG space. We define an *order* $\prec$ of variables as a total order (a linear order) on V represented as a vector $(U_1, \ldots, U_n)$, where U_i is the set of predecessors of i in the order $\prec$. To be more clear we may use $U_i^{\prec}$. We say that a DAG $G = (Pa_1, \ldots, Pa_n)$ is consistent with an order $(U_1, \ldots, U_n)$, denoted by $G \subseteq \prec$, if $Pa_i \subseteq U_i$ for each i. If S is a subset of V, we let $\mathcal{L}(S)$ denote the set of linear orders on S. In the following we will largely follow the notation from Koivisto (2006).

The algorithms working in the order space assume the *order-modular* prior defined as follows: if G is consistent with $\prec$, then

$$p(\prec, G) = \prod_{i=1}^{n} q_i(U_i)\rho_i(Pa_i), \tag{6}$$

where each q_i and ρ_i is some function from the subsets of $V - \{i\}$ to the nonnegative real numbers. (If G is not consistent with $\prec$, then $p(\prec, G) = 0$.)

A *modular feature* is defined as

$$f(G) = \prod_{i=1}^{n} f_i(Pa_i),$$

where $f_i(Pa_i)$ is an indicator function returning a $0/1$ value. For example, an edge feature $j \to i$ can be represented by always setting $f_l(Pa_l) = 1$ for each $l \neq i$ and by setting $f_i(Pa_i) = 1$ if and only if $j \in Pa_i$.

With the order-modular prior, we are interested in the posterior $p_{\prec}(f|D) = p_{\prec}(f, D)/p_{\prec}(D)$. Note that $p_{\prec}(f|D)$ can be obtained if the joint probability $p_{\prec}(f, D)$ can be computed, because $p_{\prec}(D) = p_{\prec}(f \equiv 1, D)$ where $f \equiv 1$, meaning that f always equals 1, can be easily achieved by setting each $f_i(Pa_i)$ to be the constant 1. Koivisto and Sood (2004) showed that

$$p(f, \prec, D) = \prod_{i=1}^{n} \alpha_i(U_i^{\prec}), \tag{7}$$

and

$$p_{\prec}(f, D) = \sum_{\prec} \prod_{i=1}^{n} \alpha_i(U_i^{\prec}), \tag{8}$$

where the function α_i is defined for each $i \in V$ and each $S \subseteq V - \{i\}$ as

$$\alpha_i(S) = q_i(S) \sum_{Pa_i \subseteq S} \beta_i(Pa_i),$$

in which the function β_i is defined for each $i \in V$ and each $Pa_i \subseteq V - \{i\}$ as

$$\beta_i(Pa_i) = f_i(Pa_i)\rho_i(Pa_i)score_i(Pa_i : D).$$

Accordingly, the DP algorithm of Koivisto and Sood (2004) consists of the following three steps. The first step computes $\beta_i(Pa_i)$ for each $i \in V$ and each $Pa_i \subseteq V - \{i\}$. The time complexity of this step is $O(n^{k+1}C(m))$ under the assumption of the maximum in-degree k, where n is the number of variables, and $C(m)$ is the cost of computing a single local marginal likelihood $score_i(Pa_i : D)$ for m data instances. The second step computes $\alpha_i(S)$ for each $i \in V$ and each $S \subseteq V - \{i\}$. With the assumed maximum in-degree k, this step takes $O(kn2^n)$ time by using the truncated Möbius transform technique (Koivisto and Sood, 2004), which was extended from the standard fast Möbius transform algorithm (Kennes and Smets, 1991). The third step computes $p_{\prec}(f, D)$ by defining the following function (called forward contribution) for each $S \subseteq V$:

$$L(S) = \sum_{\prec \in \mathcal{L}(S)} \prod_{i \in S} \alpha_i(U_i^{\prec}), \tag{9}$$

where $U_i^{\prec}$ is the set of variables in S ahead of i in the order $\prec \in \mathcal{L}(S)$. It can be shown that for every $S \subseteq V$, $L(S)$ can be computed recursively using the DP technique according to the following equation (Koivisto and Sood, 2004; Koivisto, 2006):

$$L(S) = \sum_{i \in S} \alpha_i(S - \{i\})L(S - \{i\}), \tag{10}$$

starting with $L(\emptyset) = 1$ and ending with $L(V)$. From Eq. (8) and Eq. (9), we have

$$p_{\prec}(f, D) = L(V). \tag{11}$$

The third step takes $O(n2^n)$ time when $L(V)$ is computed using the above DP technique. In summary, the whole DP algorithm of Koivisto and Sood (2004) can compute the posterior of any modular feature (such as an edge feature) in $O(n^{k+1}C(m) + kn2^n)$ time and $O(n2^n)$ space, where its $O(n2^n)$ space cost limits its application to Bayesian networks with up to around 25 variables.

As the extended work of Koivisto and Sood (2004), Koivisto (2006) included the DP algorithm of Koivisto and Sood (2004) as its first three steps and appended two additional steps so that all the $n(n-1)$ edges can be computed in $O(n^{k+1}C(m) + kn2^n)$ time and $O(n2^n)$ space. The foundation of the two additional steps is the introduction of the following function (called backward contribution) for each $T \subseteq V$:

$$R(T) = \sum_{\prec' \in \mathcal{L}(T)} \prod_{i \in T} \alpha_i((V - T) \cup U_i^{\prec'}). \tag{12}$$

Essentially, $R(T)$ represents the contribution of the variables in T when they are the last $|T|$ elements in the unknown linear order on V. Like $L(S)$, $R(T)$ can also be computed recursively using some DP technique. Please refer to the paper of Koivisto (2006) for further details of the two additional steps.

Although the DP algorithms (Koivisto and Sood, 2004; Koivisto, 2006) make significant contributions to the structure learning of Bayesian networks, they have one fundamental limitation: they can only compute the posteriors of modular features. In the next section, we will show how to use the results of the DP algorithm of Koivisto and Sood (2004) to efficiently draw DAG samples, which can then be used to compute the posteriors of arbitrary features.

2.2 Order MCMC

The idea of the Order MCMC is to use the Metropolis-Hastings algorithm to draw order samples $\{\prec_1, \ldots, \prec_{N_o}\}$ that have $p(\prec |D)$ as the invariant distribution, where N_o is the number of sampled orders. For this purpose we need to be able to compute $p(\prec, D)$, which can be obtained from Eq. (7) by setting $f \equiv 1$. Let $\beta_i'(Pa_i)$ denote $\beta_i(Pa_i)$ resulted from setting each $f_i(Pa_i)$ to be the constant 1. Similarly, we define $\alpha_i'(S)$ and $L'(S)$ as the special cases of $\alpha_i(S)$ and $L(S)$ by setting each $f_i(Pa_i)$ to be the constant 1. Then from Eq. (7) and Eq. (11) we have

$$p(\prec, D) = \prod_{i=1}^{n} \alpha_i'(U_i^{\prec}), \tag{13}$$

and

$$p_{\prec}(D) = L'(V). \tag{14}$$

The Order MCMC can estimate the posterior of a modular feature as

$$\hat{p}_{\prec}(f|D) = \frac{1}{N_o} \sum_{i=1}^{N_o} p(f| \prec_i, D). \tag{15}$$

For example, from Propositions 3.1 and 3.2 stated by Friedman and Koller (2003) as well as the definitions of β_i' and α_i', the posterior of a particular choice of parent set $Pa_i \subseteq U_i^{\prec}$ for node i given an order is

$$p((i, Pa_i)| \prec, D) = \frac{\beta_i'(Pa_i)}{\alpha_i'(U_i^{\prec})/q_i(U_i^{\prec})}, \tag{16}$$

and the posterior of the edge feature $j \to i$ given an order is

$$p(j \to i| \prec, D) = 1 - \frac{\alpha'_i(U_i^{\prec} - \{j\})/q_i(U_i^{\prec} - \{j\})}{\alpha'_i(U_i^{\prec})/q_i(U_i^{\prec})}. \quad (17)$$

In order to compute arbitrary non-modular features, we further draw DAG samples after drawing N_o order samples. Given an order, a DAG can be sampled by drawing parents for each node according to Eq. (16). Given DAG samples $\{G_1, \ldots, G_T\}$, we can then estimate any feature posterior $p_{\prec}(f|D)$ using $\hat{p}_{\prec}(f|D)$ shown in Eq. (5).

3. Order Sampling Algorithm and DAG Sampling Algorithms

In this section we present our order sampling algorithm, DDS algorithm, and IW-DDS algorithm. We also prove good properties of the estimators based on our algorithms.

3.1 Order Sampling Algorithm

In this subsection, we show that using the results including $\alpha'_i(S)$ (for each $i \in V$ and each $S \subseteq V - \{i\}$) and $L'(S)$ (for each $S \subseteq V$) computed from the DP algorithm of Koivisto and Sood (2004), we can draw an order sample efficiently by drawing each element in the order one by one. Let an order $\prec$ be represented as $(\sigma_1, \ldots, \sigma_n)$, where σ_i is the ith element in the order.

Proposition 1 *The conditional probability that the kth ($1 \leq k \leq n$) element in the order is σ_k given that the $n-k$ elements after it along the order are $\sigma_{k+1}, \ldots, \sigma_n$ respectively is as follows:*

$$p(\sigma_k|\sigma_{k+1}, \ldots, \sigma_n, D) = \frac{L'(U_{\sigma_k}^{\prec})\alpha'_{\sigma_k}(U_{\sigma_k}^{\prec})}{L'(U_{\sigma_{k+1}}^{\prec})}, \quad (18)$$

where $\sigma_k \in V - \{\sigma_{k+1}, \ldots, \sigma_n\}$, and $U_{\sigma_i}^{\prec} = V - \{\sigma_i, \sigma_{i+1}, \ldots, \sigma_n\}$ so that $U_{\sigma_i}^{\prec}$ denotes the set of predecessors of σ_i in the order $\prec$.
Specifically for $k = n$, we essentially have

$$p(\sigma_n = i|D) = \frac{L'(V - \{i\})\alpha'_i(V - \{i\})}{L'(V)}, \quad (19)$$

where $i \in V$.

Note that all the proofs in this paper are provided in Appendix A.

It is clear that for each $k \in \{1, \ldots, n\}$, $\sum_{i \in U_{\sigma_{k+1}}^{\prec}} p(\sigma_k = i|\sigma_{k+1}, \ldots, \sigma_n, D) = 1$ because of Eq. (10) and $U_{\sigma_k}^{\prec} = U_{\sigma_{k+1}}^{\prec} - \{\sigma_k\}$. Thus, $p(\sigma_k|\sigma_{k+1}, \ldots, \sigma_n, D)$ is a probability mass function (pmf) with k possible σ_k values from $U_{\sigma_{k+1}}^{\prec}$.

Based on Proposition 1, we propose the following order sampling algorithm to sample an order $\prec$:

- Sample σ_n, the last element of the order $\prec$, according to Eq. (19).
- For each k from $n-1$ down to 1: given the sampled $(\sigma_{k+1}, \ldots, \sigma_n)$, sample σ_k, the kth element of the order $\prec$, according to Eq. (18).

Provided that $\alpha'_i(S)$ (for each $i \in V$ and each $S \subseteq V - \{i\}$) and $L'(S)$ (for each $S \subseteq V$) have been computed, sampling an order using the above algorithm takes only $O(n^2)$ time because sampling each element σ_k ($k \in \{1, \ldots, n\}$) in the order takes $O(n)$ time.

The following proposition guarantees the correctness of our order sampling algorithm.

Proposition 2 *An order $\prec$ sampled according to our order sampling algorithm has its pmf equal to the exact posterior $p(\prec|D)$ under the order-modular prior, because*

$$\prod_{k=1}^{n} p(\sigma_k|\sigma_{k+1}, \ldots, \sigma_n, D) = p(\prec|D). \quad (20)$$

The key to our order sampling algorithm is our realization that the results including $\alpha'_i(S)$ and $L'(S)$ computed from the DP algorithm of Koivisto and Sood (2004) are already sufficient to guide the order sampling process. In an abstract point of view, the results computed from the DP algorithm of Koivisto and Sood (2004) are analogous to the answers provided by the #P-oracle stated in Theorem 3.3 of Jerrum et al. (1986). This theorem states that with the aid of a #P-oracle that is always able to provide the exact counting information (the exact number) of accepting configurations from a currently given configuration, a probabilistic Turing machine can serve as a uniform generator so that every accepting configuration will be reached with an equal positive probability. In our situation, instead of providing the exact counting information, the results computed from the DP algorithm of Koivisto and Sood (2004) are able to provide the exact joint probability $p(\sigma_k, \sigma_{k+1}, \ldots, \sigma_n, D)$ for a subsequence $(\sigma_k, \sigma_{k+1}, \ldots, \sigma_n)$ of any order $\prec$ for any $k \in \{1, 2, \ldots, n\}$, which is shown in the proof of Proposition 1 in Appendix A.1. As a result, the order sampling can be efficiently performed based on the definition of the conditional probability distribution.

3.2 DDS Algorithm

After drawing an order sample, we can then easily sample a DAG by drawing parents for each node according to Eq. (16) as described by Friedman and Koller (2003) (assuming a maximum in-degree k). This naturally leads to our algorithm, termed direct DAG sampling (DDS), as follows:

- Step 1: Run the DP algorithm of Koivisto and Sood (2004) with each $f_i(Pa_i)$ set to be the constant 1.
- Step 2 (Order sampling step): Sample N_o orders such that each order $\prec$ is independently sampled according to our order sampling algorithm.
- Step 3 (DAG sampling step): For each sampled order $\prec$, one DAG is independently sampled by drawing a parent set for each node of the DAG according to Eq. (16).

The correctness of our DDS algorithm is guaranteed by the following theorem.

Theorem 3 *The N_o DAGs sampled according to the DDS algorithm are independent and identically distributed (iid) with the pmf equal to the exact posterior $p_{\prec}(G|D)$ under the order-modular prior.*

The time complexity of the DDS algorithm is as follows. Step 1 takes $O(n^{k+1}C(m) + kn2^n)$ time (Koivisto and Sood, 2004), as discussed in Section 2.1. In Step 2, sampling each order takes $O(n^2)$ time. In Step 3, sampling each DAG takes $O(n^{k+1})$ time. Thus, the overall time complexity of our DDS algorithm is $O(n^{k+1}C(m) + kn2^n + n^2N_o + n^{k+1}N_o)$. Because typically we assume $k \geq 1$, the order sampling process (Step 2) does not affect the overall time complexity of the DDS algorithm because of its efficiency.

The time complexity of our DDS algorithm depends on the assumption of the maximum in-degree k. Such an assumption is fairly innocuous, as discussed on page 101 of Friedman and Koller (2003), because DAGs with very large families tend to have low scores. (The maximum-in-degree assumption is also justified in the context of biological expression data on page 270 of Grzegorczyk and Husmeier, 2008.) Accordingly, this assumption has been widely used in the literature (Friedman and Koller, 2003; Koivisto and Sood, 2004; Ellis and Wong, 2008; Grzegorczyk and Husmeier, 2008; Niinimaki et al., 2011; Parviainen and Koivisto, 2011) and the maximum in-degree k has been set to be no greater than 6 in all of their experiments.

Note that the DAG sampling step of the DDS algorithm takes $O(n^{k+1}N_o)$ time. This will dominate the overall running time of the DDS algorithm (even if k is assumed to be 3 or 4), when n is moderate ($n \leq 25$) and the sample size N_o reaches several thousands. Therefore, for the efficiency of our DDS algorithm, we have developed a time-saving strategy for the DAG sampling step, which will be described in details in Section 3.2.1.

Given DAG samples, $\hat{p}_{\prec}(f|D)$, an estimator of the exact posterior of any arbitrary feature f, can be constructed by Eq. (5). Let $C_{n,f}$ denote the time cost of determining the structural feature f in a DAG of n nodes. Then constructing $\hat{p}_{\prec}(f|D)$ takes $O(C_{n,f}N_o)$ time. (For example, $C_{n,f_e} = O(1)$ for an edge feature f_e; $C_{n,f_p} = O(n^2)$ for a path feature f_p.) If we only need order samples, the algorithm consisting of Steps 1 and 2 will be called direct order sampling (DOS). Given order samples, for some modular feature f such as a parent-set feature or an edge feature, $p(f|\ \prec_i, D)$ can be computed by Eq. (16) or (17), and then $p_{\prec}(f|D)$ can be estimated by Eq. (15). (Because computing a parent-set feature or an edge feature by Eq. (16) or (17) takes $O(1)$ time, estimating $p_{\prec}(f|D)$ by Eq. (15) only takes $O(N_o)$ time for these two features.)

As for the space cost of our DDS algorithm, note that Step 1 of our DDS takes $O(n2^n)$ memory space, which limits the application of our DDS to Bayesian networks with up to around 25 variables. Because a total order can be represented as a vector $(U_1, \ldots, U_n)$ and a DAG can be represented as a vector $(Pa_1, \ldots, Pa_n)$, both a total order and a DAG can be represented in $O(n^2)$ space [3]. Therefore, Steps 2 and 3 of our DDS take $O(n^2N_o)$ memory space, and the overall memory requirement of our DDS algorithm is $O(n2^n + n^2N_o)$.

Because of Theorem 3, the estimator $\hat{p}_{\prec}(f|D)$ based on our DDS algorithm has the following desirable properties.

Corollary 4 *For any structural feature f, with respect to the exact posterior $p_{\prec}(f|D)$, the estimator $\hat{p}_{\prec}(f|D)$ based on the N_o DAG samples from the DDS algorithm using Eq. (5) has the following properties:*

(i) $\hat{p}_{\prec}(f|D)$ is an unbiased estimator of $p_{\prec}(f|D)$.

3. When $n \leq 32$, in the vector representation $(Pa_1, \ldots, Pa_n)$ of a DAG, each parent set Pa_i can be represented by a 32-bit integer whose jth bit indicates whether or not node X_j is a parent of node X_i. Thus, a DAG of a moderate size n can be represented by n 32-bit integers. Similarly, when $n \leq 32$, in the vector representation $(U_1, \ldots, U_n)$ of a total order, each U_i can be represented by a 32-bit integer whose jth bit indicates whether or not node X_j is a predecessor of node X_i. Thus, a total order of a moderate size n can also be represented by n 32-bit integers.

(ii) $\hat{p}_{\prec}(f|D)$ converges almost surely to $p_{\prec}(f|D)$.

(iii) If $0 < p_{\prec}(f|D) < 1$, then the random variable

$$\frac{\sqrt{N_o}(\hat{p}_{\prec}(f|D) - p_{\prec}(f|D))}{\sqrt{\hat{p}_{\prec}(f|D)(1 - \hat{p}_{\prec}(f|D))}}$$

has a limiting standard normal distribution.

(iv) For any $\epsilon > 0$ and any $0 < \delta < 1$, if $N_o \geq (\ln(2/\delta))/(2\epsilon^2)$, then $P(|\hat{p}_{\prec}(f|D) - p_{\prec}(f|D)| < \epsilon) \geq 1 - \delta$.

In particular, Corollary 4 (iv), which is essentially from the Hoeffding bound (Hoeffding, 1963; Koller and Friedman, 2009), ensures the probability that the error of the estimator $\hat{p}_{\prec}(f|D)$ from the DDS algorithm is bounded by ϵ to be at least $1-\delta$, as long as the sample size $N_o \geq (\ln(2/\delta))/(2\epsilon^2)$. This property, which the existing MCMC algorithms (Friedman and Koller, 2003; Niinimaki et al., 2011) do not have, can be used to obtain quality guarantee for the estimator from our DDS algorithm.

3.2.1 TIME-SAVING STRATEGY FOR THE DAG SAMPLING STEP OF THE DDS

The running time of the DAG sampling step (Step 3) of the DDS algorithm is $O(n^{k+1}N_o)$, which will dominate the overall running time of the DDS algorithm when n is moderate and the sample size N_o reaches several thousands. Thus, in this sub-subsection we introduce our strategy for effectively reducing the running time of the DAG sampling step so that the efficiency of the overall DDS algorithm can be achieved.

In the DAG sampling step, each sampled order $\prec_i = (\sigma_1, \ldots, \sigma_n)^{\prec_i}$ $(1 \leq i \leq N_o)$ can be represented as $\{(\sigma_1, U_{\sigma_1})^{\prec_i}, \ldots, (\sigma_n, U_{\sigma_n})^{\prec_i}\}$, where U_{σ_j} denotes the set of predecessors of σ_j in the order. For each sampled order $\prec_i$, for each $(\sigma_j, U_{\sigma_j})^{\prec_i}$ $(1 \leq j \leq n)$, we need to sample one Pa_{σ_j} of σ_j (one parent set of σ_j) from a list $\{Pa_{\sigma_j z}\}_z^{\sigma_j}$ including every parent set $Pa_{\sigma_j z} \subseteq U_{\sigma_j}^{\prec_i}$. Let Z_j be the length of such a list. Because $Z_j = \sum_{i=0}^{\min\{k, j-1\}} \binom{j-1}{i} = O(n^k)$, sampling one Pa_{σ_j} for σ_j takes $O(n^k)$ time and sampling one DAG takes $O(n^{k+1})$ time. Note that Z_j is an increasing function of j but in the following we will use the notation Z instead of Z_j for notational convenience when the context is clear. In the following we also will not use the plural for mathematical symbols to avoid notational confusion by ending "s". The reader should assume the correct singular or plural case when reading.

When $N_o > 1$, the overall running time of the DAG sampling step can be reduced as follows. Define $\theta_z^{(\sigma_j, U_{\sigma_j})^{\prec_i}}$ as

$$\begin{aligned}\theta_z^{(\sigma_j, U_{\sigma_j})^{\prec_i}} &= P((\sigma_j, Pa_{\sigma_j z})|\ \prec_i, D) = P((\sigma_j, Pa_{\sigma_j z})|(\sigma_j, U_{\sigma_j})^{\prec_i}, D) \\ &= \beta'_{\sigma_j}(Pa_{\sigma_j z})/[\alpha'_{\sigma_j}(U_{\sigma_j}^{\prec_i})/q_{\sigma_j}(U_{\sigma_j}^{\prec_i})],\end{aligned}$$

for $z \in \{1, \ldots, Z\}$. First, using the common strategy of sampling from a discrete distribution (Koller and Friedman, 2009), for $(\sigma_j, U_{\sigma_j})^{\prec_i}$ we can create $S_I^{(\sigma_j, U_{\sigma_j})^{\prec_i}}$, a sequence of Z probability intervals with the following form:

$$\Big(\big[0, \theta_1^{(\sigma_j,U_{\sigma_j})^{\prec_i}}\big), \big[\theta_1^{(\sigma_j,U_{\sigma_j})^{\prec_i}}, \theta_1^{(\sigma_j,U_{\sigma_j})^{\prec_i}} + \theta_2^{(\sigma_j,U_{\sigma_j})^{\prec_i}}\big), \ldots, \big[\sum_{z=1}^{Z-2} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}, \sum_{z=1}^{Z-1} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}\big), \big[\sum_{z=1}^{Z-1} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}, 1\big)\Big),$$

where the lth interval is $\big[\sum_{z=1}^{l-1} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}, \sum_{z=1}^{l} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}\big)$. Note that $S_I^{(\sigma_j,U_{\sigma_j})^{\prec_i}}$ can be created in time $O(Z)$ and sampling one Pa_{σ_j} for σ_j from a list $\{Pa_{\sigma_j z}\}_z^{\sigma_j}$ can then be achieved using binary search in time $O(\log Z)$ based on $S_I^{(\sigma_j,U_{\sigma_j})^{\prec_i}}$. Then the following observation is the key reason that the running time of the DAG sampling step can be reduced. For two sampled orders $\prec_i$ and $\prec_{i'}$ ($1 \leq i, i' \leq N_o$), even if $\prec_i \neq \prec_{i'}$, it is possible that $(\sigma_j, U_{\sigma_j})^{\prec_i} = (\sigma_j, U_{\sigma_j})^{\prec_{i'}}$ for some $j \in \{1, \ldots, n\}$. This is because for each j, (σ_j, U_{σ_j}) essentially has a multinomial distribution with N_o trials and a set of $n\binom{n-1}{j-1}$ cell probabilities $\{P((\sigma_j, U_{\sigma_j})|D)\}$. Actually, for any j, the following relation holds for each cell probability:

$$P((\sigma_j, U_{\sigma_j})|D) \propto \alpha'_{\sigma_j}(U_{\sigma_j}) L'(U_{\sigma_j}) R'(V - U_{\sigma_j} - \{\sigma_j\}), \tag{21}$$

where $R'(.)$ is the special case of $R(.)$ by setting $f \equiv 1$ and $R(.)$ is defined in Eq. (12). The proof of Eq. (21) is very similar to the derivation shown by Koivisto (2006) and is provided in Appendix A.5 Note that $(\sigma_j, U_{\sigma_j})^{\prec_i} = (\sigma_j, U_{\sigma_j})^{\prec_{i'}}$ implies $S_I^{(\sigma_j,U_{\sigma_j})^{\prec_i}} = S_I^{(\sigma_j,U_{\sigma_j})^{\prec_{i'}}}$. Thus, by storing the created $S_I^{(\sigma_j,U_{\sigma_j})^{\prec_i}}$ in the memory, once $(\sigma_j, U_{\sigma_j})^{\prec_i} = (\sigma_j, U_{\sigma_j})^{\prec_{i'}}$ for $i' > i$, creating $S_I^{(\sigma_j,U_{\sigma_j})^{\prec_{i'}}}$ can be avoided and sampling one Pa_{σ_j} for σ_j takes only $O(\log Z)$ time.

On one hand, our strategy will definitely save the running time for these j such that $n\binom{n-1}{j-1}$ (the number of all the possible values of (σ_j, U_{σ_j})) is smaller than N_o if every created $S_I^{(\sigma_j,U_{\sigma_j})}$ is stored. This is because the running time of sampling Pa_{σ_j} of σ_j is only $O(\log Z)$ in at least $N_o - n\binom{n-1}{j-1}$ samples out of the overall N_o samples. (In the worst case, $S_I^{(\sigma_j,U_{\sigma_j})}$ will be created for each possible (σ_j, U_{σ_j}).) For example, when $j = n$, the number of all the possible values of (σ_j, U_{σ_j}) is only n, and Z_j (the length of the list $\{Pa_{\sigma_j z}\}_z^{\sigma_j}$) achieves its maximum among all the j, so that sampling one Pa_{σ_n} for σ_n takes $O(\log Z_n)$ time in at least $N_o - n$ samples. Accordingly, when $j = n$, the worst-case running time of sampling the N_o $(\sigma_n, Pa_{\sigma_n})$ families is $O(n(Z_n + \log Z_n) + (N_o - n)\log Z_n) = O(nZ_n + N_o \log Z_n)$.

On the other hand, our strategy usually can also save the running time even for these j such that the number of all the possible values of (σ_j, U_{σ_j}) is larger than N_o. This is because the probability mass usually is not uniformly distributed among the set of all the possible values of (σ_j, U_{σ_j}). Once the majority of the probability mass p_Σ is concentrated on r_j values of (σ_j, U_{σ_j}), where r_j is a number smaller than N_o, the probability that only these r_j values of (σ_j, U_{σ_j}) appear in all the N_o order

samples is $(p_\Sigma)^{N_o}$. Accordingly, with the probability $(p_\Sigma)^{N_o}$, the running time of sampling Pa_{σ_j} for σ_j is $O(\log Z_j)$ in at least $N_o - r_j$ samples. As a result, the expected running time of sampling the N_o $(\sigma_j, Pa_{\sigma_j})$ families is below $O([r_j Z_j + N_o \log Z_j](p_\Sigma)^{N_o} + N_o(Z_j + \log Z_j)(1 - (p_\Sigma)^{N_o}))$; the expected running time of sampling the N_o DAGs is below $O(\sum_{j=1}^{n}\{[r_j Z_j + N_o \log Z_j](p_\Sigma)^{N_o} + N_o(Z_j + \log Z_j)(1 - (p_\Sigma)^{N_o})\})$. Typically, when m is not small, the local score $score_i(Pa_i : D)$ will not be uniform at all. Correspondingly, it is likely that the multinomial probability mass function $P((\sigma_j, U_{\sigma_j})|D)$ will concentrate dominant probability mass on a small number of (σ_j, U_{σ_j}) candidates that these $(\sigma_j, Pa_{\sigma_j})$ having large local scores are consistent with. As a result, our time-saving strategy will usually become more effective when m is not small.

Note that we also include the policy of recycling the created $S_I^{(\sigma_j,U_{\sigma_j})}$ for our strategy because it is possible that all the memory in a computer will be exhausted in order to store all the created $S_I^{(\sigma_j,U_{\sigma_j})}$, especially when n is not small but m is small. (The space complexity of storing all the $S_I^{(\sigma_j,U_{\sigma_j})}$ is $O(\sum_{j=1}^{n} n\binom{n-1}{j-1} Z_j) = O(n^{k+1} 2^{n-1})$.) For this paper, we use a simple recycling method as follows. Some upper limit for the total number of the probability intervals (representing $\big[\sum_{z=1}^{l-1} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}, \sum_{z=1}^{l} \theta_z^{(\sigma_j,U_{\sigma_j})^{\prec_i}}\big)$) is pre-specified based on the memory of the computer used. Each time such an upper limit is reached during the DAG sampling step of the DDS, which indicates a large amount of memory has been used to store $S_I^{(\sigma_j,U_{\sigma_j})}$, we recycle the currently stored $S_I^{(\sigma_j,U_{\sigma_j})}$ according to their usage frequencies which serve as the estimates of $P((\sigma_j, U_{\sigma_j})|D)$. The memory for each infrequently used $S_I^{(\sigma_j,U_{\sigma_j})}$ will be reclaimed to ensure that at least a pre-specified number of probability intervals will be recycled from the memory. In addition, in order to have a better use of each created $S_I^{(\sigma_j,U_{\sigma_j})}$ before it possibly gets reclaimed, we sort the N_o sampled orders according to the posterior $p(\prec|D)$ just before executing the DAG sampling step of the DDS. The underlying rationale is that if $p(\prec_i|D)$ is relatively close to $p(\prec_{i'}|D)$, which indicates $p(\prec_i, D)$ is relatively close to $p(\prec_{i'}, D)$ (because $p_\prec(D)$ is a constant), it is likely that $\prec_i$ and $\prec_{i'}$ share some (σ_j, U_{σ_j}) component(s) because of Eq. (13). (The extreme situation is that if $p(\prec_i|D)$ equals $p(\prec_{i'}|D)$, it is very likely that $\prec_i$ equals $\prec_{i'}$ so that $\prec_i$ and $\prec_{i'}$ share every (σ_j, U_{σ_j}).) Thus, $\prec_i$ and $\prec_{i'}$ having similar posteriors tend to be close to each other after the sorting, so that it is likely that the common $S_I^{(\sigma_j,U_{\sigma_j})}$ will be used before the reclamation. Furthermore, as N_o increases, the probability that two orders (out of the N_o sampled orders) share some (σ_j, U_{σ_j}) component(s) increases. Accordingly, after the sorting, the probability of reusing $S_I^{(\sigma_j,U_{\sigma_j})}$ before its reclamation will also increase. As a result, the benefit of our time-saving strategy will typically increase when N_o increases.

The experimental results show that our time-saving strategy for the DAG sampling step of the DDS is very effective. Please see the discussion in Section 4 about $\hat{\mu}(T_{DAG})$ and $\hat{\sigma}(T_{DAG})$, the sample mean and the sample standard deviation of the running time of the DAG sampling step of the DDS, which are reported in Tables 2 and 4.

3.3 IW-DDS Algorithm

In this subsection we present our DAG sampling algorithm under the general structure-modular prior (Eq. 2) by effectively correcting the bias due to the use of the order-modular prior.

As mentioned in Section 1, $p_\prec(f|D)$ has the bias due to the assumption of the order-modular prior. This is essentially because $p_\prec(G|D)$ based on the order-modular prior (Eq. 6) is different from $p_\not\prec(G|D)$ based on the structure-modular prior (Eq. 2).

In fact, with the common setting that $q_i(U_i)$ always equals 1 ($q_i(U_i) \equiv 1$), if $\rho_i(Pa_i)$ in Eq. (6) is set to be always equal to $p_i(Pa_i)$ in Eq. (2) ($\rho_i(Pa_i) \equiv p_i(Pa_i)$), the following relation holds:

$$p_{\prec}(G|D) = \frac{p_{\not\prec}(D)}{p_{\prec}(D)} \cdot \mid \prec_G \mid \cdot p_{\not\prec}(G|D), \tag{22}$$

where $\mid \prec_G \mid$ is the number of orders that G is consistent with. (The proof of Eq. 22 is given in Appendix A.6.) Accordingly,

$$p_{\not\prec}(f|D) = \sum_G f(G) p_{\not\prec}(G|D) = \sum_G f(G) \frac{p_{\prec}(D)}{p_{\not\prec}(D)} \cdot \frac{1}{\mid \prec_G \mid} p_{\prec}(G|D).$$

Note that $p_{\prec}(D)$ can be computed by the DP algorithm of Koivisto and Sood (2004) in $O(n^{k+1}C(m) + kn2^n)$ time, and $p_{\not\prec}(D)$ can be computed by the DP algorithm of Tian and He (2009) in $O(n^{k+1}C(m) + kn2^n + 3^n)$ time. Thus, if $\mid \prec_{G_i} \mid$ is known for each sampled G_i ($i \in \{1, 2, \ldots, N_o\}$), we can use importance sampling to obtain a good estimator

$$\tilde{p}_{\not\prec}(f|D) = \frac{1}{N_o} \sum_{i=1}^{N_o} f(G_i) \frac{p_{\prec}(D)}{p_{\not\prec}(D)} \cdot \frac{1}{\mid \prec_{G_i} \mid}, \tag{23}$$

where each G_i is sampled from our DDS algorithm. Unfortunately, $\mid \prec_{G_i} \mid$ is #P-hard to compute for each G_i (Brightwell and Winkler, 1991); the state-of-the-art DP algorithm proposed by Niinimaki and Koivisto (2013) for computing $\mid \prec_{G_i} \mid$ takes $O(n2^n)$ time. Therefore, in the following we propose an estimator that can be much more efficiently computed than the estimator shown in Eq. (23).

Because $p_{\prec}(f|D)$ has the bias with respect to $p_{\not\prec}(f|D)$, a good estimator $\hat{p}_{\prec}(f|D)$ of $p_{\prec}(f|D)$ typically is not appropriate to be directly used to estimate $p_{\not\prec}(f|D)$. Noticing this problem, Ellis and Wong (2008) proposed to correct this bias for the Order MCMC method as follows: first run the Order MCMC to draw order samples; then for each unique order $\prec$ out of the sampled orders, keep drawing DAGs consistent with $\prec$ (but only keep unique DAGs) until the sum of joint probabilities of these unique DAGs, $\sum_i p(\prec, G_i, D)$, is no less than a pre-specified large proportion (such as 95%) of $p(\prec, D) = \sum_{G \subseteq \prec} p(\prec, G, D)$; finally the resulting union of all the DAG samples is treated as an importance-weighted sample for the structural discovery.

Inspired by the idea of Ellis and Wong (2008), we develop our own bias-correction strategy which is computationally more efficient and can theoretically ensure the resulting estimator to have desirable properties. (Please refer to Section 3.3.1 for detailed discussion.) Our bias-corrected algorithm, termed IW-DDS (importance-weighted DDS), is as follows:

- Step 1 (DDS step): Run the DDS algorithm to draw N_o DAG samples with the setting that $q_i(U_i) \equiv 1$ and $\rho_i(Pa_i) \equiv p_i(Pa_i)$.
- Step 2 (Bias correction step): Make the union set $\mathcal{G}$ of all the sampled DAGs by eliminating the duplicate DAGs.

Given $\mathcal{G}$, $\hat{p}_{\not\prec}(f|D)$, the estimator of the exact posterior of any feature f, can then be constructed as

$$\hat{p}_{\not\prec}(f|D) = \sum_{G \in \mathcal{G}} f(G) \hat{p}_{\not\prec}(G|D), \tag{24}$$

where

$$\hat{p}_{\not\prec}(G|D) = \frac{p_{\not\prec}(G, D)}{\sum_{G \in \mathcal{G}} p_{\not\prec}(G, D)}, \tag{25}$$

and $p_{\not\prec}(G, D)$ is given in Eq. (3).

Because checking the equality of two DAGs takes $O(n^2)$ time [4], with the use of a hash table, both the expected time cost and the space cost of the bias-correction step of the IW-DDS are $O(n^2 N_o)$. Therefore, the expected time cost of our IW-DDS algorithm is $O(n^{k+1}C(m) + kn2^n + n^2 N_o + n^{k+1}N_o)$, and the required memory space of our IW-DDS algorithm is $O(n2^n + n^2 N_o)$.

Note that when each G_i gets sampled, the corresponding joint probability $p_{\not\prec}(G_i, D)$ can be easily computed and stored with G_i. Therefore, just as constructing the estimator from the DDS, constructing the estimator $\hat{p}_{\not\prec}(f|D)$ from the IW-DDS also takes $O(C_{n,f} N_o)$ time, where $C_{n,f}$ denotes the time cost of determining the structural feature f in a DAG of n nodes.

While Ellis and Wong (2008) showed the effectiveness of their method in correcting the bias merely by the experiments, we first theoretically prove that our estimator has desirable properties as follows.

Theorem 5 *For any structural feature f, with respect to the exact posterior $p_{\not\prec}(f|D)$, the estimator $\hat{p}_{\not\prec}(f|D)$ based on the DAG samples from the IW-DDS algorithm using Eq. (24) has the following properties:*

(i) $\hat{p}_{\not\prec}(f|D)$ is an asymptotically unbiased estimator of $p_{\not\prec}(f|D)$.

(ii) $\hat{p}_{\not\prec}(f|D)$ converges almost surely to $p_{\not\prec}(f|D)$.

(iii) The convergence rate of $\hat{p}_{\not\prec}(f|D)$ is $o(a^{N_o})$ for any $0 < a < 1$.

(iv) Define the quantity $\Delta = \sum_{G \in \mathcal{G}} p_{\not\prec}(G|D)$. Then

$$\Delta \cdot \hat{p}_{\not\prec}(f|D) \leq p_{\not\prec}(f|D) \leq \Delta \cdot \hat{p}_{\not\prec}(f|D) + 1 - \Delta. \tag{26}$$

Note that the introduced quantity $\Delta = \sum_{G \in \mathcal{G}} p_{\not\prec}(G, D)/p_{\not\prec}(D)$ and $\Delta \in [0, 1]$ essentially represents the cumulative posterior probability mass of the DAGs in $\mathcal{G}$. Eq. (26) provides a sound interval $[\Delta \cdot \hat{p}_{\not\prec}(f|D), \Delta \cdot \hat{p}_{\not\prec}(f|D) + 1 - \Delta]$ in which $p_{\not\prec}(f|D)$ *must* reside. (The "sound interval" is stronger than the "confidence interval" because there is no probability that $p_{\not\prec}(f|D)$ is outside the sound interval.) The width of the sound interval is $(1 - \Delta)$, where Δ is a nondecreasing function of N_o (because if we increase the original N_o to a larger N_o' and sample additional $N_o' - N_o$ DAGs in the DDS step, the resulting $\mathcal{G}'$ is always a superset of the original $\mathcal{G}$). Thus, in the situations where m (the number of data instances) is not very small, it is possible for Δ to approach 1 by a tractable number N_o of DAG samples so that a desired small-width interval for $p_{\not\prec}(f|D)$ can be obtained. (Please refer to Section 4 for the corresponding experimental results.) Also note that Eq. (26) can be expressed in the following equivalent form:

$$-(1 - \Delta)\hat{p}_{\not\prec}(f|D) \leq p_{\not\prec}(f|D) - \hat{p}_{\not\prec}(f|D) \leq (1 - \Delta)(1 - \hat{p}_{\not\prec}(f|D)), \tag{27}$$

which gives the bound for the estimation error $p_{\not\prec}(f|D) - \hat{p}_{\not\prec}(f|D)$.

4. As described in the footnote in Section 3.2, for the vector representation $(Pa_1, \ldots, Pa_n)$ of a DAG of a size $n \leq 32$, each parent set Pa_i can be represented by a 32-bit integer. Accordingly, for two DAGs of a moderate size n, checking their equality takes at most n comparisons by comparing each integer component in their vector representations.

The competing state-of-the-art algorithms that are also applicable to BNs of a moderate size are the Hybrid MCMC method (Eaton and Murphy, 2007) and the K-best algorithm (Tian et al., 2010). The first competing method, the Hybrid MCMC, includes the DP algorithm of Koivisto (2006) (with time complexity $O(n^{k+1}C(m)+ kn2^n)$ and space complexity $O(n2^n)$) as its first phase and then uses the computed posteriors of all the edges to make the global proposal for its second phase (MCMC phase). When its MCMC phase eventually converges, the Hybrid MCMC will correct the bias coming from the order-prior assumption and provide DAG samples according to the DAG posterior so that the estimator $\hat{p}_{\not\prec}(f|D)$ can be constructed using Eq. (5) for any feature f. The Hybrid MCMC has been empirically shown to converge faster than both the Structure MCMC and the Order MCMC, so that more accurate structure-learning performance can be obtained (Eaton and Murphy, 2007). Note that because the REV-MCMC method (Grzegorczyk and Husmeier, 2008) is shown to be only nearly as efficient as the Order MCMC in the mixing and convergence, the Hybrid MCMC is expected to converge faster than the REV-MCMC method as long as n is moderate so that the Hybrid MCMC is applicable. (But the REV-MCMC method has its own value in learning large BNs since all these methods using some DP technique, including the Hybrid MCMC, the K-best algorithm, and our IW-DDS method, are infeasible for a large n because of the space cost.) One limitation of the Hybrid MCMC is that it cannot obtain the interval for $p_{\not\prec}(f|D)$ as specified by Theorem 5 (iv). Additionally, the convergence rate of the estimator from the Hybrid MCMC is not theoretically provided by its authors.

The second competing method, the K-best algorithm, applies some DP technique to obtain a collection $\mathcal{G}$ of DAGs with the K best scores and then uses these DAGs to construct the estimator $\hat{p}_{\not\prec}(f|D)$ by Eq. (24) and Eq. (25). One advantage of the K-best algorithm is that its estimator also has the property specified by Theorem 5 (iv) so that it can provide the sound interval for $p_{\not\prec}(f|D)$ just as our IW-DDS. However, the K-best algorithm has time complexity $O(n^{k+1}C(m)+ T'(K)n2^{n-1})$ and space complexity $O(Kn2^n)$, where $T'(K)$ is the time spent on the best-first search for K solutions and $T'(K)$ has been shown to be $O(K \log K)$ by Flerova et al. (2012). Thus, the increase in K will dramatically increase the computation cost of the K-best algorithm when n is not small. As a result, to obtain an interval width similar to that from our IW-DDS, much more time and space cost is required for the K-best. In our experiments using an ordinary desktop PC, the computation problem becomes severe for $n \geq 19$ because K can only take some small values (such as no more than 40) before the K-best algorithm exhausts the memory. Accordingly, Δ obtained from the K-best is usually smaller than that from our IW-DDS (so that the interval from the K-best is usually wider) even if K is set to reach the memory limit of a computer. (Please refer to Section 4.2 for detailed discussion.)

3.3.1 COMPARISON BETWEEN OUR BIAS-CORRECTION STRATEGY AND THAT OF ELLIS AND WONG (2008)

Our bias-correction strategy used in the IW-DDS solves the computation problem existing in the idea of Ellis and Wong (2008) and ensures the desirable properties of our estimator $\hat{p}_{\not\prec}(f|D)$ stated in Theorem 5.

Because in the Order MCMC, sampling an order is much more computationally expensive than sampling a DAG given an order, the strategy of Ellis and Wong (2008) emphasizes making the full use of each sampled order $\prec$, that is, keeping drawing DAGs consistent with each sampled $\prec$ until the sum of joint probabilities for the unique sampled DAGs, $\sum_i p(\prec, G_i, D)$, is no less than a large

proportion (such as 95%) of $p(\prec, D)$. Unfortunately, such a strategy has a computational problem when the number of variables n is not small and the number of data instances m is small. Because there are a super-exponential number ($2^{\Theta(kn \log(n))}$) of DAGs (with the maximum in-degree k) consistent with each order (Friedman and Koller, 2003), it is possible that a non-negligible portion of probability mass $p(\prec, D)$ will be distributed almost uniformly to a majority of these consistent DAGs when m is small. Consequently, $N_G^{\prec}$, the required number of DAGs sampled per each sampled order $\prec$, will be extremely large, leading to large computation cost. For sampling $N_G^{\prec}$ DAGs consistent with each sampled order $\prec$, its expected time cost is $O(n^{k+1} + nk \log(n) N_G^{\prec})$ (even if a time-saving strategy as that described in Section 3.2.1 is used) and its memory requirement is $O(n^2 N_G^{\prec})$. If the memory requirement exceeds the memory of the running computer, the hard disk has to be used to temporarily store the sampled DAGs in some way. (We notice that Ellis and Wong, 2008, limited their experiments to the data sets with at most 14 variables.) If we take the data set "Child" (Tsamardinos et al., 2006) with $n = 20$ and $m = 100$ for example, for an order $\prec$ randomly sampled by our order sampling algorithm, our experiment shows that 1×10^7 DAGs (which contain 932,137 unique DAGs) need to be sampled to let the ratio $\sum_i p(\prec, G_i, D) \,/p(\prec, D)$ reach 94.071%; 1.5×10^7 DAGs (which contain 1,204,262 unique DAGs) need to be sampled to let the ratio $\sum_i p(\prec, G_i, D) \,/p(\prec, D)$ reach 94.952%. To solve this problem, based on the efficiency of our order sampling algorithm, our strategy samples only one DAG from each sampled order in the DDS step, so that the large computation cost per each sampled order is avoided for any data set. Meanwhile, unlike the strategy of Ellis and Wong (2008), our strategy does not delete the duplicate order samples. Therefore, if an order $\prec$ gets sampled j (≥ 1) times in the order sampling step, essentially j DAGs will be sampled for such a unique order in the DAG sampling step. Thus, j, the number of occurrences, implicitly serves as an importance indicator for $\prec$ among the orders.

Furthermore, the strategy of Ellis and Wong (2008) cannot guarantee that the sampled DAGs are independent, even if large computation cost is spent in sampling a huge number of DAGs per each sampled order. This is essentially because multiple DAGs sampled from a fixed order according to the strategy of Ellis and Wong (2008) are not independent. For example, given that a DAG G with an edge $X \to Y$ gets sampled from an order $\prec$, which implies that node X precedes node Y in the given order $\prec$, then the conditional probability that any DAG G' with a reverse edge $Y \to X$ gets sampled under the fixed order $\prec$ becomes zero, so that G and G' are not independent. In general, once the number of sampled orders is fixed, even if the number of sampled DAGs per each sampled order keeps increasing, every DAG that is consistent with none of the sampled orders will still have no chance of getting sampled. In contrast, the sampling strategy in our IW-DDS is able to guarantee the property that all the DAGs sampled from the DDS step are independent, which has been stated in Theorem 3. Such a property is a key to ensuring the good properties of our estimator $\hat{p}_{\not\prec}(f|D)$ stated in Theorem 5.

4. Experimental Results

We have implemented our algorithms in a C++ language tool called "BNLearner" and run several experiments to demonstrate its capabilities. (BNLearner is available at `http://www.cs.iastate.edu/~jtian/Software/BNLearner/BNLearner.htm`.) Our tested data sets include ten real data sets from the UCI Machine Learning Repository (Asuncion and Newman, 2007): "Tic-Tac-Toe," (which is also simply called "T-T-T,") "Glass," "Wine," "Housing," "Credit," "Zoo," "Letter," "Tumor," "Vehicle," and "German." Our tested data sets also include three syn-

thetic data sets: the first one is a synthetic data set "Syn15" generated from a gold-standard 15-node Bayesian network built by us; the second one is a synthetic data set "Insur19" generated from a 19-node subnetwork of "Insurance" Bayesian network (Binder et al., 1997); the third one is a synthetic data set "Child" from a 20-node "Child" Bayesian network used by Tsamardinos et al. (2006). All the data sets contain only discrete variables (or are discretized) and have no missing values (or have their missing values filled in). For the four data sets ("Syn15," "Letter," "Insur19," and "Child"), because a large number of data instances are available, we also vary m (the number of instances) to see the corresponding learning performance. (All the data cases are also included in the tool of BNLearner.) All the experiments in this section were run under Linux on one ordinary desktop PC with a 3.0 GHz Intel Pentium processor and 2.0 GB memory if no extra specification is provided. In addition, the maximum in-degree k was assumed to be 5 for all the experiments.

4.1 Experimental Results for the DDS

In this subsection, we compare our DDS algorithm with the Partial Order MCMC method (Niinimaki et al., 2011), the state-of-the-art learning method under the order-modular prior.

The Partial Order MCMC (PO-MCMC) method is implemented in BEANDisco, a C++ language tool provided by Niinimaki et al. (2011). (BEANDisco is available at `http://www.cs.helsinki.fi/u/tzniinim/BEANDisco/`.) The current version of BEANDisco can only estimate the posterior of an edge feature, but as Niinimaki et al. (2011) stated, the PO-MCMC readily enables estimating the posterior of any structural feature by further sampling DAGs consistent with an order.

Because n (the number of the variables) in each investigated data case is moderate, we are able to use REBEL, a C++ language implementation of the DP algorithm of Koivisto (2006), to get the exact posterior of every edge under the assumption of the order-modular prior. (REBEL is available at `http://www.cs.helsinki.fi/u/mkhkoivi/REBEL/`.) Therefore, we can use the criterion of the sum of the absolute differences (SAD) (Eaton and Murphy, 2007) to measure the feature-learning performance for each data case:

$$\text{SAD} = \sum_f |p(f|D) - \hat{p}(f|D)|,$$

where $p(f|D)$ is the exact posterior of the investigated feature f, and $\hat{p}(f|D)$ is the corresponding estimator. In this subsection, SAD is essentially $\sum_{ij} |p_{\prec}(i \to j|D) - \hat{p}_{\prec}(i \to j|D)|$, because the investigated feature is the edge feature $i \to j$ under the order-modular prior. A smaller SAD will indicate a better performance in structure discovery. Note that the criterion SAD is closely related to another criterion MAD (the mean of the absolute differences), because MAD = SAD$/(n(n-1))$. Thus, for each data case the conclusion based on the comparison of SAD values is the same as that based on the comparison of MAD values, because $n(n-1)$ is just a constant for each data case.

For fair comparison, in our algorithms we used the K2 score (Heckerman et al., 1995) and set $q_i(U_i) = 1$ and $\rho_i(Pa_i) = 1/\binom{n-1}{|Pa_i|}$ for each i, U_i, and Pa_i, where $|Pa_i|$ is the size of the set Pa_i, because such a setting is used in both BEANDisco and REBEL.

For the setting of the PO-MCMC, according to the suggestion for the optimal setting from Niinimaki et al. (2011), we set the bucket size b to be 10 for all the data cases except Tic-Tac-Toe. The bucket size b was set to be 9 for the data case Tic-Tac-Toe, because Tic-Tac-Toe has only 10 variables and the setting $b = 10$ will cause the tool BEANDisco to throw a run-time error. We

Name	n	m	PO-MCMC		DOS		DDS	
			$\hat{\mu}$(SAD)	$\hat{\sigma}$(SAD)	$\hat{\mu}$(SAD)	$\hat{\sigma}$(SAD)	$\hat{\mu}$(SAD)	$\hat{\sigma}$(SAD)
T-T-T (Tic-Tac-Toe)	10	958	0.5174	0.1280	0.1350	0.0257	0.1547	0.0378
Glass	11	214	0.0696	0.0249	0.0230	0.0067	0.0529	0.0076
Wine	13	178	0.1616	0.0403	0.0400	0.0097	0.0839	0.0137
Housing	14	506	0.3205	0.1303	0.0650	0.0155	0.1150	0.0117
Credit	16	690	0.4549	0.2495	0.0581	0.0221	0.1071	0.0165
Zoo	17	101	0.6079	0.1809	0.1030	0.0127	0.2756	0.0137
Tumor	18	339	0.6059	0.1849	0.0877	0.0226	0.2050	0.0180
Vehicle	19	846	6.9774	8.7960	0.0200	0.0042	0.0547	0.0096
German	21	1,000	2.8802	2.0191	0.1014	0.0208	0.1298	0.0338
Syn15	15	100	0.9024	0.2258	0.1299	0.0191	0.2622	0.0190
		200	0.6449	0.1569	0.0999	0.0119	0.2228	0.0172
		500	0.3424	0.1214	0.0801	0.0183	0.1116	0.0126
		1,000	0.1558	0.0496	0.0550	0.0094	0.0724	0.0118
		2,000	0.0465	0.0209	0.0350	0.0089	0.0473	0.0071
		5,000	0.0217	0.0144	0.0226	0.0073	0.0247	0.0086
Letter	17	100	0.9530	0.1285	0.1526	0.0181	0.2948	0.0229
		200	0.3854	0.0825	0.0794	0.0210	0.1758	0.0142
		500	0.4369	0.1529	0.0732	0.0115	0.1326	0.0107
		1,000	0.3007	0.1254	0.0561	0.0134	0.0828	0.0171
		2,000	1.3740	0.9177	0.1014	0.0373	0.1386	0.0288
		5,000	0.0669	0.0139	0.0199	0.0037	0.0292	0.0088
Insur19	19	100	0.6150	0.1995	0.0977	0.0202	0.1575	0.0213
		200	0.4428	0.1319	0.0655	0.0126	0.1024	0.0162
		500	0.2757	0.1127	0.0462	0.0132	0.0594	0.0099
		1,000	0.4539	0.3031	0.0362	0.0134	0.0422	0.0134
		2,000	0.0100	0.0073	0.0067	0.0037	0.0079	0.0029
		5,000	0.0110	0.0100	0.0095	0.0046	0.0116	0.0043
Child	20	100	0.4997	0.1153	0.0779	0.0143	0.1772	0.0146
		200	0.1896	0.0528	0.0419	0.0102	0.0982	0.0101
		500	0.2385	0.0702	0.0405	0.0066	0.0816	0.0123
		1,000	0.1079	0.0525	0.0284	0.0069	0.0406	0.0080
		2,000	0.0864	0.0521	0.0216	0.0073	0.0275	0.0083
		5,000	0.0938	0.0539	0.0194	0.0058	0.0246	0.0066

Table 1: Comparison of the PO-MCMC, the DOS, and the DDS in Terms of SAD

ran the first 10,000 iterations for "burn-in," and then took 200 partial-order samples at intervals of 50 iterations. Thus, there were 20,000 iterations in total. (The time cost of each iteration in the PO-MCMC is $O(n^{k+1} + n^2 2^b n/b)$.) In the PO-MCMC, for each sampled partial order P_i, $p(f|D, P_i)$ is obtained by $p(D, f, P_i)/p(D, P_i) = p(D, f, P_i)/p(D, f \equiv 1, P_i)$, where $p(D, f, P_i) = \sum_{\prec \supseteq P_i} \sum_{G \subseteq \prec} f(G)\, p(\prec, G) p(D|G)$. The notation $\sum_{\prec \supseteq P_i}$ means that all the total orders ($\prec$'s) that are linear extensions of the sampled partial order P_i will be included to obtain $p(D, f, P_i)$. For example, for a data set with $n = 20$, because our bucket size $b = 10$, there are $20!/(10!10!) =$ 184,756 total orders that will be included for each sampled partial order P_i. The inclusion of the information of a large number of total orders consistent with each sampled partial order gives great learning power to the PO-MCMC method; such an inclusion can be efficiently computed by the algorithm of Parviainen and Koivisto (2010) with the assumptions of the order-modular prior and the maximum in-degree k. Finally, for the PO-MCMC, the estimated posterior of each edge is computed using $\hat{p}_{\prec}(f|D) = (1/T) \sum_{i=1}^{T} p(f|D, P_i)$.

Because the to-be-learned feature is the edge feature, we can also use our DOS algorithm for the comparison. For both the DOS algorithm and the DDS algorithm, we set N_o = 20,000, that

Name	n	m	PO-MCMC	DOS	DDS	DDS					
			$\hat{\mu}(T_t)$	$\hat{\mu}(T_t)$	$\hat{\mu}(T_t)$	$\hat{\mu}(T_{DP})$	$\hat{\sigma}(T_{DP})$	$\hat{\mu}(T_{ord})$	$\hat{\sigma}(T_{ord})$	$\hat{\mu}(T_{DAG})$	$\hat{\sigma}(T_{DAG})$
T-T-T	10	958	104.30	1.96	1.73	1.42	0.0159	0.24	0.0066	0.06	0.0017
Glass	11	214	222.02	1.52	1.25	0.89	0.0136	0.25	0.0076	0.09	0.0013
Wine	13	178	374.17	2.56	2.53	1.63	0.0121	0.35	0.0039	0.53	0.0024
Housing	14	506	510.65	4.92	4.54	3.88	0.0143	0.40	0.0033	0.23	0.0020
Credit	16	690	962.97	30.90	30.93	29.57	0.2118	0.44	0.0055	0.90	0.0122
Zoo	17	101	1,331.80	13.75	21.90	12.05	0.0837	0.62	0.0039	9.20	0.1156
Tumor	18	339	1,856.03	44.99	60.21	43.33	1.0763	0.72	0.0052	16.12	0.2941
Vehicle	19	846	2,683.91	149.08	149.23	147.35	1.2863	0.65	0.0079	1.20	0.0219
German	21	1,000	4,887.33	333.43	356.17	330.64	1.3304	0.97	0.0087	24.52	0.4441
Syn15	15	100	677.29	4.82	7.13	3.49	0.0824	0.50	0.0021	3.12	0.0337
		200	677.47	5.91	8.91	4.59	0.0473	0.47	0.0044	3.83	0.0258
		500	686.51	8.56	8.44	7.41	0.1911	0.48	0.0100	0.53	0.0050
		1,000	716.31	13.01	12.52	11.78	0.0927	0.48	0.0115	0.24	0.0022
		2,000	731.50	21.70	21.24	20.58	0.5310	0.49	0.0086	0.15	0.0016
		5,000	731.05	48.01	47.26	46.63	0.4110	0.47	0.0031	0.14	0.0008
Letter	17	100	1,322.43	16.35	21.91	14.64	0.2160	0.64	0.0036	6.60	0.0497
		200	1,315.01	19.74	20.46	18.14	0.0809	0.55	0.0026	1.74	0.0234
		500	1,338.33	27.97	28.21	26.35	0.0489	0.51	0.0066	1.32	0.0130
		1,000	1,343.88	39.45	39.03	38.11	0.3011	0.47	0.0051	0.43	0.0089
		2,000	1,358.29	61.75	61.44	60.52	0.5109	0.47	0.0078	0.42	0.0063
		5,000	1,610.37	126.53	126.49	125.67	0.7967	0.52	0.0058	0.27	0.0053
Insur19	19	100	2,616.56	53.39	86.06	51.06	0.2520	0.86	0.0091	34.11	0.9630
		200	2,633.44	62.12	77.44	59.95	0.3025	0.84	0.0083	16.61	0.2278
		500	2,680.85	80.70	85.03	77.89	0.7618	0.79	0.0082	6.32	0.0535
		1,000	2,734.10	106.37	109.39	104.63	1.0958	0.89	0.0122	3.85	0.0296
		2,000	2,915.60	155.05	158.31	154.26	3.7703	0.90	0.0154	3.13	0.0155
		5,000	3,445.84	297.31	300.51	297.41	4.7737	0.96	0.0090	2.11	0.0091
Child	20	100	3,710.49	102.42	181.38	99.71	0.3497	1.07	0.0109	80.56	1.1980
		200	3,717.10	112.68	168.48	110.05	0.1793	1.04	0.0078	57.36	0.5335
		500	3,757.76	136.98	193.11	134.32	0.4652	1.07	0.0111	57.69	0.7276
		1,000	3,799.47	174.18	186.15	171.54	1.9832	1.08	0.0104	13.50	0.9244
		2,000	4,018.03	241.48	254.26	238.37	2.4952	1.15	0.0214	14.71	0.4833
		5,000	4,531.20	443.54	455.30	441.64	4.8785	1.16	0.0068	12.47	0.4113

Table 2: Comparison of the PO-MCMC, the DOS, and the DDS in Terms of Time (in Seconds)

is, 20,000 (total) orders were sampled. Theoretically, we expect that the learning performance of the DOS should be better than that of the DDS because the additional approximation coming from the DAG sampling step is avoided by the DOS. By listing the performance of the DOS, we mainly intend to examine how much the performance of the DDS decreases because of the additional approximation from sampling one DAG per order. Nevertheless, because the DDS but not the DOS is capable of learning non-modular features, the comparison between the PO-MCMC method and the DDS method is our main task.

Table 1 shows the experimental results in terms of SAD for each data case with n variables and m instances, and Table 2 lists the running-time cost corresponding to Table 1. For each of the three methods, we performed 15 independent runs for each data case. The sample mean and the sample standard deviation of the 15 SAD values of each method, denoted by $\hat{\mu}$(SAD) and $\hat{\sigma}$(SAD), are listed in Table 1. Correspondingly, the sample mean of the total running time T_t of each method, denoted by $\hat{\mu}(T_t)$, is shown in Table 2. (Precisely speaking, the reported total running time T_t of the DDS method includes both the time of running the three steps of the DDS and the relatively tiny $O(N_o)$ time cost of computing $\hat{p}_{\prec}(f|D)$ for each edge f using Eq. 5 at the end. Similarly,

the reported total running time T_t of the DOS method also includes the relatively tiny $O(N_o)$ time cost of computing $\hat{p}_{\prec}(f|D)$ for each edge f using Eq. 17 and Eq. 15 at the end.) In addition, the sample mean and the sample standard deviation of the running time of the three steps of the DDS (including the DP step, the order sampling step, and the DAG sampling step), denoted by $\hat{\mu}(T_{DP})$, $\hat{\sigma}(T_{DP})$, $\hat{\mu}(T_{ord})$, $\hat{\sigma}(T_{ord})$, $\hat{\mu}(T_{DAG})$, and $\hat{\sigma}(T_{DAG})$ respectively, are listed in the last six columns of Table 2. Note that we still show $\hat{\mu}(T_{DP})$, the mean running time of the DP step of the DDS in the 15 independent runs, though the DP step is not a random algorithm at all. The running time of the DP step is not exactly the same in each run because of the randomness from uncontrolled factors such as the internal status of the computer. By showing $\hat{\mu}(T_{DP})$, we can clearly see the percentage of the total running time that the DP step typically takes by comparing $\hat{\mu}(T_{DP})$ and $\hat{\mu}(T_t)$.

Tables 1 and 2 clearly illustrate the performance advantage of our DDS method over the PO-MCMC method. The overall time cost of our DDS based on 20,000 DAG samples is much smaller than the corresponding cost of the PO-MCMC method based on 20,000 MCMC iterations in the partial-order space. Using much shorter time, our DDS method has its $\hat{\mu}$(SAD) much smaller than $\hat{\mu}$(SAD) from the PO-MCMC method for 28 out of all the 33 data cases. The five exceptional cases are Glass, Syn15 with $m = 2{,}000$, Syn15 with $m = 5{,}000$, Insur19 with $m = 2{,}000$, and Insur19 with $m = 5{,}000$. (In both Glass and Insur19 with $m = 2{,}000$, $\hat{\mu}$(SAD) using our DDS method is still smaller than that using the PO-MCMC method, but their difference is not large compared with $\hat{\sigma}$(SAD) from the PO-MCMC method.) Furthermore, because both $\hat{\mu}$(SAD) and $\hat{\sigma}$(SAD) are given in Table 1, by the two-sample t test with unequal variances (Casella and Berger, 2002), we can conclude with strong evidence (at the significance level 5×10^{-3}) that the real mean of SAD using our DDS method is smaller than the real mean of SAD using the PO-MCMC method for each of the 28 data cases. For the exceptional data case Glass, the p-value of the t test is 0.012, so that we can conclude at the significance level 0.05 that the real mean of SAD using our DDS method is smaller than that using the PO-MCMC method. For each of the other four exceptions, by the same t test we cannot reject (with the p-value > 0.2) the null hypothesis that there is no difference in the real means of SAD from the two methods. (Corresponding to Table 1, the comparison is also re-demonstrated using boxplots in the supplementary material for all the 33 data cases.) Thus, the advantage of our DDS algorithm over the PO-MCMC method in learning Bayesian networks of a moderate n can be clearly seen, though the value of the PO-MCMC method still remains for larger n for which our DDS algorithm is infeasible.

In terms of the total running time of the DDS algorithm, Table 2 shows that the running time of the DP step always accounts for the largest portion. The running time of the DAG sampling step is less than 81 seconds to get 20,000 DAG samples for all the 33 cases. Though both the order sampling step and the DAG sampling step involve randomness, the variability of their running time is actually small. This can be seen from the ratio of $\hat{\sigma}(T_{ord})$ to $\hat{\mu}(T_{ord})$ (which is always less than 3.04% for all the 33 cases) and the ratio of $\hat{\sigma}(T_{DAG})$ to $\hat{\mu}(T_{DAG})$ (which is always less than 6.85% for all the 33 cases). The ratio of $\hat{\mu}(T_{DAG})$ to $\hat{\mu}(T_{ord})$ ranges from 0.25 to 75.29 across the 33 cases, which is much smaller than the upper bound of the ratio of $O(n^{k+1}N_o)$ to $O(n^2N_o)$. This indicates that our time-saving strategy introduced in Section 3.2.1 can effectively reduce the running time of the DAG sampling step. In addition, the running time of the DAG sampling step often decreases further when m increases, which can be clearly seen from all the four data sets (Syn15, Letter, Insur19, and Child) with different values of m. Take the data set Letter for example, when m increases from 100 to 1,000, the corresponding $\hat{\mu}(T_{DAG})$ decreases from 6.60 to 0.43

second, a 93.48% of decrease. In summary, the effectiveness of our time-saving strategy introduced in Section 3.2.1 has been clearly shown in Table 2.

In addition, Table 1 clearly shows that the learning performance of the DOS is better than that of the DDS, as expected theoretically. Note that $\hat{\mu}$(SAD) from the DOS is significantly smaller than $\hat{\mu}$(SAD) from the DDS for 28 out of the 33 data cases. The five other cases are Tic-Tac-Toe, Syn15 with $m =$ 5,000, and Insur19 with $m =$ 1,000, 2,000, and 5,000. For each of these 28 data cases, by the two-sample t test with unequal variances (Casella and Berger, 2002), we can conclude (at the significance level 0.05) that the real mean of SAD using the DOS is smaller than that using the DDS. This shows that the additional approximation from the DAG sampling step will usually make the DDS perform worse than the DOS in learning modular features. Nevertheless, the DDS but not the DOS has the ability of learning arbitrary non-modular features, which is the main goal of this paper.

Finally, we present the experimental results for the DDS by varying the sample size. We first choose the data case Letter with $m = 500$ as an example. For the DDS, we tried the sample size $N_o =$ 5,000 $\cdot i$, where $i \in \{1, 2, \ldots, 10\}$. For each i, we independently ran the DDS 15 times to get the sample mean and the sample standard deviation of SAD for the (directed) edge features. For the PO-MCMC, with the bucket size $b = 10$, we ran totally 5,000 $\cdot i$ MCMC iterations in the partial-order space, where $i \in \{1, 2, \ldots, 10\}$. For each i, we discarded the first 2,500 $\cdot i$ MCMC iterations for "burn-in" and set the thinning parameter to be 50, so that $50 \cdot i$ partial orders finally got sampled. Again, for each i, we independently ran the PO-MCMC 15 times to get the sample mean and the sample standard deviation of SAD for the edge features.

Figure 1 shows the SAD performance of the two methods with each i in terms of the edge features, where an error bar represents one sample standard deviation $\hat{\sigma}$(SAD) across 15 runs from a method (the PO-MCMC or the DDS) at each i. For example, Figure 1 shows that when $i = 4$, $\hat{\mu}$(SAD) $= 0.1326$ and $\hat{\sigma}$(SAD) $= 0.0107$ from our DDS algorithm; $\hat{\mu}$(SAD) $= 0.4369$ and $\hat{\sigma}$(SAD) $= 0.1529$ from the PO-MCMC method. This exactly matches the results previously shown in Table 1. Correspondingly, Figure 2 shows $\hat{\mu}(T_t)$ (the sample mean of the total running time) of the PO-MCMC and the DDS with each i, where the running time is in seconds. The advantage of our DDS can be clearly seen from Figures 1 and 2. For each $i \in \{1, 2, \ldots, 10\}$, the real mean of SAD from the DDS is significantly smaller than that from the PO-MCMC with the p-value $< 5 \times 10^{-3}$ returned by the two-sample t test (with unequal variances). Meanwhile, the total running time of the DDS is much shorter than that of the PO-MCMC. For example, $\hat{\mu}(T_t)$ of the DDS increases with respect to i and reaches only 29.55 seconds at $i = 10$. This is shorter than 9% of $\hat{\mu}(T_t)$ of the PO-MCMC at $i = 1$, which is 336.09 seconds. Therefore, the learning performance of the DDS with each sample size is significantly better than that of the PO-MCMC for the data case Letter with $m = 500$.

We also performed the experiment with the same experimental settings for the data cases Tic-Tac-Toe, Wine, Child with $m = 500$, and German. Please refer to the supplementary material for the experimental results. (The supplementary material is available at `http://www.cs.iastate.edu/~jtian/Software/BNLearner/BN_Learning_Sampling_Supplement.pdf`.) The same conclusion about the learning performance can be clearly drawn by examining the figures shown in the supplementary material.

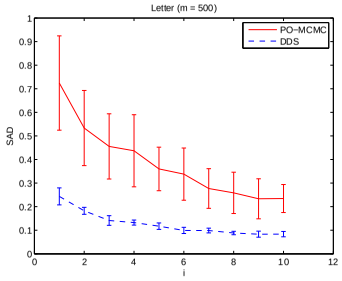


Figure 1: Plot of the SAD Performance of the PO-MCMC and the DDS for Letter ($m = 500$)

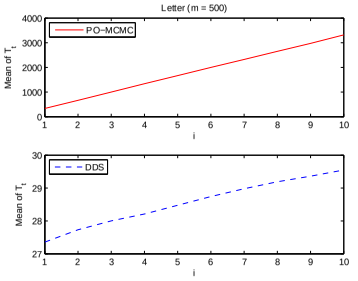


Figure 2: Plot of the Total Running Time of the PO-MCMC and the DDS for Letter ($m = 500$)

4.2 Experimental Results for the IW-DDS

In this subsection, we compare our IW-DDS algorithm with the Hybrid MCMC (that is, DP+MCMC) method (Eaton and Murphy, 2007) and the K-best algorithm (Tian et al., 2010), two state-of-the-art methods that can estimate the posteriors of any features without the order-modular prior assumption. The implementation of the Hybrid MCMC (called BDAGL) and the implementation of the K-best (called KBest) are both made available online by their corresponding authors. (BDAGL is available at `http://www.cs.ubc.ca/~murphyk/Software/BDAGL/`; KBest is available at `http://www.cs.iastate.edu/~jtian/Software/UAI-10/KBest.htm`.)

Because n in each investigated data case is moderate, we are able to use POSTER, a C++ language implementation of the DP algorithm of Tian and He (2009), to get $p_{\not\prec}(i \to j|D)$, the exact posterior of each edge $i \to j$ under the structure-modular prior. (POSTER is available at `http://www.cs.iastate.edu/~jtian/Software/UAI-09/Poster.htm`.) Therefore, we can use the SAD criterion $(\sum_{ij} |p_{\not\prec}(i \to j|D) - \hat{p}_{\not\prec}(i \to j|D)|)$ to measure the performance of these three methods in the structure learning for each data case.

For fair comparison, in our algorithm we used the BDeu score (Heckerman et al., 1995) with equivalent sample size 1 and set $p_i(Pa_i)(\equiv \rho_i(Pa_i)) \equiv 1$, because these settings are also used in POSTER and the implementation of the K-best algorithm.

As for the DP+MCMC, we note that most part of its implementation in BDAGL tool is written in Matlab, whereas both the K-best and the IW-DDS are implemented in C++. In order to make relatively fair comparison in terms of the running time, we used REBEL tool, a C++ implementation of the DP algorithm of Koivisto (2006), to perform the computation in the DP phase of the DP+MCMC; but for fair comparison we changed its scoring criterion into the BDeu score with equivalent sample size 1 and set $q_i(U_i) \equiv 1$ and $\rho_i(Pa_i) \equiv 1$. To perform the computation in the MCMC phase, we used the Matlab implementation of BDAGL [5] and ran it under Windows 7 on

5. The original BDAGL was found to get an out-of-memory error for any data case with more than 19 variables in our experiments. This is because the original BDAGL intends to pre-compute the local scores of all the $n2^{n-1}$ possible families and store them in an array for the later usage in both the DP phase and the MCMC

Name	n	m	DP	DP+MCMC		K-best	IW-DDS		K-best	IW-DDS	
			SAD	$\hat{\mu}$(SAD)	$\hat{\sigma}$(SAD)	SAD	$\hat{\mu}$(SAD)	$\hat{\sigma}$(SAD)	Δ	$\hat{\mu}(\Delta)$	$\hat{\sigma}(\Delta)$
T-T-T	10	958	0.1651	15.0079	2.9877	1.4194	0.0227	0.0102	6.943E-01	9.935E-01	8.636E-04
Glass	11	214	1.5444	0.3587	0.4599	0.0904	0.0381	0.0019	9.780E-01	9.901E-01	6.368E-04
Wine	13	178	1.4786	0.4605	0.2968	0.2011	0.1041	0.0075	9.023E-01	9.670E-01	1.700E-03
Housing	14	506	5.6478	8.0000	3.3408	9.1179	4.8276	0.0624	2.880E-02	1.096E-01	1.200E-03
Credit	16	690	4.0580	5.0261	2.3482	5.1492	2.9148	0.0336	5.010E-02	1.793E-01	1.700E-03
Zoo	17	101	8.2142	32.4189	10.0953	35.1215	19.7025	4.0767	3.815E-08	1.652E-11	1.211E-11
Tumor	18	339	5.1536	17.5104	7.9198	20.5793	10.3139	0.6950	2.940E-05	3.619E-06	3.586E-07
Vehicle	19	846	3.5759	3.8234	0.4011	3.3683	0.4648	0.0194	5.387E-01	9.432E-01	2.600E-03
German	21	1,000	3.7261	5.0207	3.2223	5.5902	0.9891	0.0449	7.800E-02	7.422E-01	7.200E-03
Syn15	15	100	4.9321	12.8705	7.6384	11.8685	10.1341	0.1495	1.604E-04	1.183E-04	4.664E-06
		200	3.2557	4.5090	2.5875	7.5232	4.9079	0.0583	2.700E-03	7.300E-03	1.265E-04
		500	6.9798	5.5466	1.9175	4.4379	4.2965	0.1619	1.526E-01	2.388E-01	4.900E-03
		1,000	1.3000	0.3974	0.3299	0.0848	0.0498	0.0048	9.699E-01	9.843E-01	8.909E-04
		2,000	1.7192	1.8263	1.7095	0.3701	0.1081	0.0147	8.521E-01	9.703E-01	3.300E-03
		5,000	1.9473	0.0304	0.0094	8.89E-04	0.0022	0.0002	9.998E-01	9.994E-01	9.821E-05
Letter	17	100	9.2140	27.1507	4.0940	24.4313	15.8780	0.2764	2.908E-04	1.621E-04	5.336E-06
		200	7.2855	15.1587	3.5615	9.4512	6.7936	0.1191	7.800E-03	1.220E-02	3.341E-04
		500	6.0961	3.4637	4.6789	1.7237	0.6347	0.0119	6.948E-01	8.808E-01	3.000E-03
		1,000	0.6394	0.1761	0.0166	0.0837	0.0766	0.0039	9.834E-01	9.864E-01	8.678E-04
		2,000	2.3913	3.5085	3.1132	2.0976	0.1338	0.0213	6.859E-01	9.756E-01	2.700E-03
		5,000	0.8407	0.1182	0.0442	0.0160	0.0072	0.0005	9.948E-01	9.972E-01	2.077E-04
Insur19	19	100	5.3356	9.4318	3.9576	11.7779	6.5062	0.0891	6.000E-03	2.010E-02	4.767E-04
		200	5.9844	5.5465	2.7295	2.2572	1.4630	0.0557	4.495E-01	7.049E-01	1.120E-02
		500	1.8274	0.3605	0.2287	0.4970	0.1328	0.0105	7.464E-01	9.379E-01	3.000E-03
		1,000	1.7386	0.2186	0.0762	0.7498	0.0623	0.0111	5.866E-01	9.785E-01	2.600E-03
		2,000	1.2737	0.1217	0.0380	0.0174	0.0062	0.0012	9.900E-01	9.976E-01	4.864E-04
		5,000	1.9511	0.1765	0.0594	0.0103	0.0092	0.0010	9.970E-01	9.973E-01	2.086E-04
Child	20	100	6.9783	11.8987	3.1086	11.6189	7.0304	0.0909	7.848E-04	2.700E-03	1.066E-04
		200	3.2826	4.7066	4.3749	5.0729	2.8510	0.0192	4.800E-03	1.506E-01	1.200E-03
		500	2.5580	2.4716	1.3489	1.5304	0.5416	0.0305	3.582E-01	8.222E-01	5.100E-03
		1,000	2.4708	2.6061	2.2909	0.7066	0.1499	0.0198	7.013E-01	9.545E-01	3.400E-03
		2,000	2.3330	1.4286	1.2290	1.5279	0.0662	0.0161	6.509E-01	9.828E-01	2.800E-03
		5,000	2.0365	1.2533	1.7313	0.8783	0.0150	0.0013	8.209E-01	9.940E-01	4.696E-04

Table 3: Comparison of the DP+MCMC, the K-best, and the IW-DDS in Terms of SAD

an ordinary laptop with 2.40 GHz Intel Core i5 CPU and 4.0 GB memory. The MCMC used the pure global proposal (with the local proposal choice $\beta = 0$), because such a setting was reported by Eaton and Murphy (2007) to have the best performance for edge discovery when up to about 190,000 MCMC iterations were performed in their experimental results. We ran totally 190,000 MCMC iterations each time and discarded the first 100,000 iterations as the burn-in period. Then we set the thinning parameter to be 3 to get the final 30,000 DAG samples. As a result, the time statistics of the DP phase (the numbers before the + sign in Table 4) but not the MCMC phase (the numbers after the + sign) can be directly compared with those of the other two methods. For each data case, we performed 20 independent MCMC runs based on the DP outcome from REBEL to get the results.

phase. To solve this out-of-memory problem, we have updated the original Matlab code in BDAGL and provided the BDAGL-New package which is also available at `http://www.cs.iastate.edu/~jtian/Software/BNLearner/BNLearner.htm`. The main update is that, with the assumed maximum in-degree, only the local scores of all the families whose sizes are no more than the assumed maximum in-degree are pre-computed and stored in a hash table. With BDAGL-New, the experiments for all the data cases in this paper can be performed without any error.

Name	n	m	DP+MCMC	K-best	IW-DDS	IW-DDS					
			$\hat{\mu}(T_t)$	T_t	$\hat{\mu}(T_t)$	$\hat{\mu}(T_{DP})$	$\hat{\sigma}(T_{DP})$	$\hat{\mu}(T_{ord})$	$\hat{\sigma}(T_{ord})$	$\hat{\mu}(T_{DAG})$	$\hat{\sigma}(T_{DAG})$
T-T-T	10	958	1.29 + 1,032.40	8.37	3.28	1.27	0.0060	1.57	0.0109	0.44	0.0128
Glass	11	214	1.04 + 1,037.26	18.13	1.72	0.98	0.0210	0.50	0.0054	0.24	0.0018
Wine	13	178	2.44 + 1,127.80	141.20	3.52	2.15	0.0199	0.63	0.0061	0.74	0.0057
Housing	14	506	4.83 + 1,421.00	323.37	5.67	4.21	0.0813	0.63	0.0042	0.52	0.0043
Credit	16	690	33.73 + 1,476.90	2,073.41	35.20	29.30	0.3322	0.81	0.0055	4.94	1.6191
Zoo	17	101	22.33 + 2,107.50	5,531.81	26.16	12.49	0.1853	1.11	0.0048	12.05	0.1898
Tumor	18	339	60.86 + 1,799.99	18,640.09	87.35	39.49	0.4419	1.19	0.0184	46.31	0.2937
Vehicle	19	846	207.27 + 1,886.10	17,126.45	171.75	160.55	0.9310	1.49	0.0175	9.67	0.0665
German	21	1,000	600.19 + 1,849.90	10,981.29	540.61	392.25	2.8656	1.68	0.0207	146.65	1.4290
Syn15	15	100	5.21 + 1,284.00	889.97	9.41	3.56	0.0667	0.81	0.0048	4.80	0.0273
		200	6.28 + 1,286.20	901.23	10.29	4.76	0.2090	0.80	0.0111	4.53	0.0256
		500	9.64 + 1,336.80	899.42	9.59	7.50	0.0821	0.85	0.0051	1.08	0.0128
		1,000	13.69 + 1,364.60	910.76	13.35	12.15	0.7578	0.85	0.0156	0.33	0.0038
		2,000	22.64 + 1,372.10	907.02	21.98	20.81	0.5200	0.85	0.0061	0.30	0.0019
		5,000	54.79 + 1,356.70	932.96	52.07	47.80	1.0887	3.99	0.0235	0.28	0.0023
Letter	17	100	25.53 + 1,572.60	7,639.02	20.27	15.83	0.0601	1.15	0.0062	2.83	0.0292
		200	30.01 + 1,576.80	7,966.25	25.87	20.22	0.1769	1.09	0.0069	4.21	0.0304
		500	39.58 + 1,598.90	8,257.60	31.88	29.84	0.1575	0.95	0.0055	1.01	0.0123
		1,000	52.85 + 1,575.60	8,380.57	44.48	42.93	0.4835	0.98	0.0093	0.56	0.0100
		2,000	77.48 + 1,591.00	7,619.29	69.14	66.34	0.5184	2.01	0.0241	0.78	0.0068
		5,000	157.69 + 1,636.40	8,134.85	136.95	134.94	1.6127	1.36	0.0097	0.65	0.0067
Insur19	19	100	101.47 + 1,828.00	6,745.41	112.28	55.85	0.1540	1.41	0.0117	54.71	0.5438
		200	113.79 + 1,896.10	6,783.76	82.52	68.12	0.4187	1.45	0.0120	12.90	0.1329
		500	137.18 + 1,864.40	6,894.15	98.96	91.44	0.4547	1.43	0.0128	6.07	0.0358
		1,000	168.55 + 1,862.30	6,966.90	133.87	123.42	0.8007	1.44	0.0157	9.00	0.0601
		2,000	226.36 + 1,781.70	7,277.60	185.02	177.66	1.3165	3.30	0.0483	4.06	0.0356
		5,000	380.78 + 1,814.80	7,061.76	336.03	329.78	4.5841	1.89	0.0220	4.34	0.0713
Child	20	100	203.44 + 1,785.10	15,085.86	223.62	106.01	0.3976	1.67	0.0176	115.60	1.3183
		200	215.70 + 1,760.80	14,222.86	225.76	119.82	1.8604	1.69	0.0107	104.09	0.9659
		500	248.17 + 1,818.70	14,016.94	214.91	149.73	0.8344	1.67	0.0183	63.48	0.5783
		1,000	292.40 + 1,817.20	15,504.82	232.11	193.18	1.1041	1.72	0.0135	37.20	0.3176
		2,000	371.12 + 1,841.40	16,109.91	306.42	268.78	2.3209	1.82	0.0165	35.81	0.3680
		5,000	589.99 + 1,846.40	15,372.61	524.63	483.90	6.7403	1.93	0.0160	38.80	0.5411

Table 4: Comparison of the DP+MCMC, the K-best, and the IW-DDS in Terms of Time (in Seconds)

For our IW-DDS, we set $N_o = 30{,}000$. We performed 20 independent runs for each data case to get the results. For the K-best, note that its SAD is fixed because there is no randomness in the computed results. So we only ran it once to get the result. We set K to be 100 for Tic-Tac-Toe, Glass, Wine, Housing, Credit, Zoo, Syn15, and Letter, that is, we got the 100 best DAGs from Tic-Tac-Toe, Glass, Wine, Housing, Credit, Zoo, each of the six cases of Syn15, and each of the six cases of Letter. We set K to be only 80 for Tumor because our experiments showed that for Tumor the K-best program ran out of memory with $K > 80$. Because of the same out-of-memory issue, we set K to be only 40 for Vehicle and Insur19; we set K to be only 20 for Child and 9 for German. The fact that K can only take a value no greater than 40 for $n \geq 19$ in our experiments confirms our claim about the computation problem of the K-best algorithm in terms of its space cost.

Table 3 shows the experimental results in terms of SAD for each data case, and Table 4 shows the running-time cost corresponding to Table 3. The column named DP in Table 3 shows the SAD ($\sum_{ij} |p_{\not\prec}(i \to j|D) - p_{\prec}(i \to j|D)|$), where each edge posterior $p_{\not\prec}(i \to j|D)$ is computed by the exact DP method of Tian and He (2009), and each edge posterior $p_{\prec}(i \to j|D)$ is computed by

the exact DP method of Koivisto (2006). The SAD values reported in this column indicate the bias due to the assumption of the order-modular prior. Next to the DP column, the SAD values of the three methods are listed in Table 3. Because the DP+MCMC method and the IW-DDS method are random, both the sample mean $\hat{\mu}$(SAD) and the sample standard deviation $\hat{\sigma}$(SAD) of the 20 SAD values are shown for these two methods. The outcome of the K-best algorithm is not random, so that only its SAD is shown. Finally Table 3 also shows the cumulative posterior probability mass Δ for both the K-best algorithm and the IW-DDS method. Again, because the IW-DDS method is random, both the sample mean and the sample standard deviation of the 20 Δ values, denoted by $\hat{\mu}(\Delta)$ and $\hat{\sigma}(\Delta)$, are listed for the IW-DDS method in Table 3. As for Table 4, its presentation format is very similar to that of Table 2. Table 4 first shows the sample mean of the total running time T_t (denoted by $\hat{\mu}(T_t)$) of both the DP+MCMC method and the IW-DDS method as well as the total running time T_t of the K-best algorithm. The last six columns of Table 4, similar to the last six columns of Table 2, list the sample mean and the sample standard deviation of the running time of the three steps of the DDS in the IW-DDS algorithm.

Tables 3 and 4 clearly demonstrate the advantage of our method over the other two methods. With much shorter computation time, our method has its $\hat{\mu}$(SAD) less than that from the DP+MCMC for 32 out of the 33 data cases. The only exceptional case is Syn15 with $m = 200$. Furthermore, based on the two-sample t test with unequal variances, we can conclude at the significance level 0.05 that the real mean of SAD using our method is less than that using the DP+MCMC for each of the 31 cases; the two exceptional cases are Syn15 with $m = 100$ and Syn15 with $m = 200$. (For 30 out of these 31 cases, the p-value of the two-sample t test is less than 0.01.) Meanwhile, $\hat{\sigma}$(SAD) using our method is always much smaller than that using the DP+MCMC for each of the 33 cases, which indicates higher stability of the performance of our method. Similarly, with much shorter computation time, our method has its $\hat{\mu}$(SAD) less than the SAD from the K-best for 32 out of the 33 cases. The only exception is Syn15 with m = 5,000. Furthermore, based on the one-sample t test (Casella and Berger, 2002), we can conclude at the significance level 5×10^{-4} that the real mean of SAD using our method is less than the SAD using the K-best for each of these 32 cases. (Corresponding to Table 3, the comparison of the three methods is also re-illustrated using boxplots in the supplementary material for all the 33 data cases.)

There are several other interesting things shown in Tables 3 and 4. In terms of the SAD, for very small m, $\hat{\mu}$(SAD) using the DP+MCMC method is even larger than the SAD from the DP phase (Koivisto, 2006) itself. For example, for the data case Zoo, the SAD from the DP phase is 8.2142, but $\hat{\mu}$(SAD) obtained after the MCMC phase of the DP+MCMC method is 32.4189. Similar situations also occur in Syn15, Letter, Insur19, and Child when $m = 100$. This indicates that for very small m, the MCMC phase of the DP+MCMC method is unable to reduce the bias from the DP method of Koivisto (2006) for all these cases based on 190,000 MCMC iterations. As for the running time, note that $\hat{\mu}(T_{DP})$ of our IW-DDS is always less than the running time of the DP phase of the DP+MCMC method. This is because the DP step of our method uses the DP algorithm of Koivisto and Sood (2004), that is, the first three steps of the DP algorithm of Koivisto (2006); while the DP phase of the DP+MCMC method uses all the five steps of the DP algorithm of Koivisto (2006). In other words, compared with the DP algorithm of Koivisto and Sood (2004), the DP algorithm of Koivisto (2006) includes a larger constant factor hidden in the $O(n^{k+1}C(m) + kn2^n)$ notation though these two DP algorithms have the same time complexity. This difference will make the total running time of our IW-DDS even less than the running time of the DP phase of the DP+MCMC method when the remaining steps of the IW-DDS run faster than the last two steps

of the DP algorithm of Koivisto (2006). For example, for the data case Child with m = 5,000, $\hat{\mu}(T_t)$ of the IW-DDS is 524.63 seconds while the corresponding running time of the DP phase of the DP+MCMC method is 589.99 seconds. Actually, Table 4 shows that there are 21 out of the 33 cases where $\hat{\mu}(T_t)$ of the IW-DDS is less than the running time of the DP phase of the DP+MCMC method. In addition, just as shown in Section 4.1, the effectiveness of our time-saving strategy can also be clearly seen from Table 4. For example, the ratio of $\hat{\mu}(T_{DAG})$ to $\hat{\mu}(T_{ord})$ ranges from 0.07 to 87.29 across the 33 cases, which is much smaller than the upper bound of the ratio of $O(n^{k+1}N_o)$ to $O(n^2N_o)$.

Table 3 also shows the resulting cumulative probability mass Δ from the K-best and our IW-DDS for all the data cases. (Note that Δ is computed by the formula $\Delta = \sum_{G \in \mathcal{G}} p_{\not\perp}(G, D)/p_{\not\perp}(D)$, where $p_{\not\perp}(D)$ is computed using POSTER tool.) In the table, $\hat{\mu}(\Delta)$ from our IW-DDS is greater than Δ from the K-best for 28 out of the 33 data cases. The five exceptional cases are Zoo, Tumor, Syn15 with $m = 100$, Syn15 with m = 5,000, and Letter with $m = 100$. Interestingly, for four out of the five exceptional cases (as well as the other 28 cases), $\hat{\mu}$(SAD) from our IW-DDS is significantly smaller than SAD from the K-best. One possible reason is that the K best DAGs tend to have the same or similar local structures (families (i, Pa_i)'s) that have relatively large local scores, but a large number of DAGs sampled from our IW-DDS include various local structures for each node i. When Δ is far below 1, the inclusion of various local structures seems to be more effective in improving the structure-learning performance.

In addition, Table 3 shows that when m is not very small (such as no smaller than 1,000), Δ from our IW-DDS with N_o = 30,000 can reach a large percentage (such as greater than 90%) in most of our data cases. As a result, we can obtain a sound interval for $p_{\not\perp}(f|D)$ with a small width (such as less than 0.1) for any feature f.

To further demonstrate that our IW-DDS can obtain a large Δ efficiently when m is not very small, we increased N_o from 100,000 to 600,000 with each increment 100,000 to see its performance for the data cases Letter with $m = 500$, Child with $m = 500$, and German. Again, we performed 20 independent runs for each data case to get the results. Figures 3, 5, and 7 show the increase in Δ with respect to the increase in N_o for these three data cases. Correspondingly, Figures 4, 6, and 8 indicate the increase in $\hat{\mu}(T_t)$, $\hat{\mu}(T_{ord})$ and $\hat{\mu}(T_{DAG})$ with respect to the increase in N_o for these three data cases, where the running time is in seconds. These figures clearly show that our IW-DDS can efficiently achieve a large Δ. Take the data case German for example, with the time cost $\hat{\mu}(T_t)$ = 1,493.02 seconds, our IW-DDS can collect N_o = 600,000 DAG samples so that the corresponding mean of Δ can reach 91.74%. Therefore, for any feature f in the data case German, our IW-DDS can provide a sound interval for $p_{\not\perp}(f|D)$ with a width of 0.0826. Note that the K-best can only provide a meaningless sound interval for $p_{\not\perp}(f|D)$ with a huge width of 0.922 because its Δ can only reach 0.078 in the data case German before running out of the memory. Also note that the ratio of $\hat{\mu}(T_{DAG})$ to $\hat{\mu}(T_{ord})$ decreases from 56.45 to 30.95 when N_o increases from 100,000 to 600,000. (The rate of increase of $\hat{\mu}(T_{ord})$ is a constant with respect to N_o, but the rate of increase of $\hat{\mu}(T_{DAG})$ actually decreases as N_o increases.) This confirms our statement in Section 3.2.1 that the benefit from our time-saving strategy will typically increase when N_o increases.

Finally, we present the experimental results for the IW-DDS by varying the sample size. As in Section 4.1, the experiments were performed for the five data cases Tic-Tac-Toe, Wine, Letter with $m = 500$, Child with $m = 500$, and German. For the IW-DDS, we tried the sample size N_o = 5,000 $\cdot i$, where $i \in \{1, 2, \ldots, 10\}$. For each i, we independently ran the IW-DDS 20 times to get the sample mean and the sample standard deviation of SAD for the (directed) edge features. For

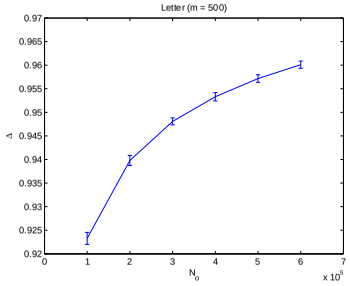


Figure 3: Plot of Δ versus N_o for Letter ($m = 500$)

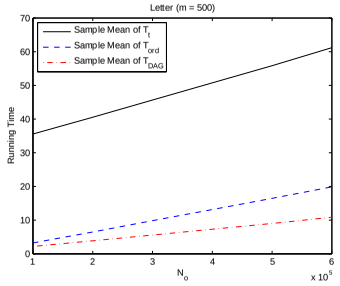


Figure 4: Plot of the Running Time versus N_o for Letter ($m = 500$)

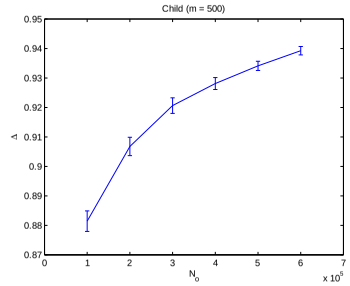


Figure 5: Plot of Δ versus N_o for Child ($m = 500$)

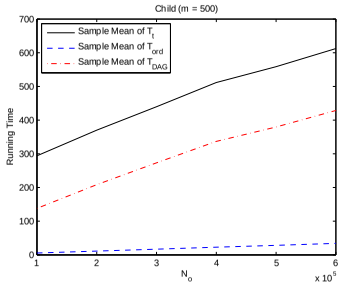


Figure 6: Plot of the Running Time versus N_o for Child ($m = 500$)

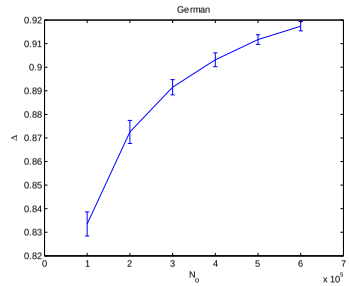


Figure 7: Plot of Δ versus N_o for German

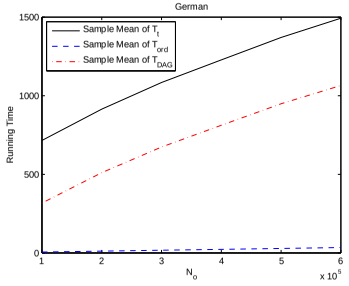


Figure 8: Plot of the Running Time versus N_o for German

the DP+MCMC, we ran totally 50,000 $\cdot i$ MCMC iterations, where $i \in \{1, 2, \ldots, 10\}$. For each i, we discarded the first 25,000 $\cdot i$ MCMC iterations for "burn-in" and set the thinning parameter to be 5, so that 5,000 $\cdot i$ DAGs got sampled. Again, for each i, we independently ran the MCMC 20 times to get the sample mean and the sample standard deviation of SAD for the edge features. As for the K-best, different experimental settings were used for different data cases because of the out-of-memory issue. For the two data cases Tic-Tac-Toe and Wine, we ran the K-best program with $K = 20 \cdot i$, where $i \in \{1, 2, \ldots, 10\}$. (The setting of $K = 20 \cdot i$ guarantees that for these two data cases the running time of the K-best is longer than the running time of the IW-DDS at each i, which will be demonstrated soon.) For the data case Letter with $m = 500$, we only ran the K-best with $K = 162$ because the K-best program will run out of memory when $K > 162$ because of its expensive space cost. The corresponding result of the K-best would be compared with the result of the IW-DDS at $i = 10$ (that is, the IW-DDS with $N_o = 50{,}000$). For the same out-of-memory issue, we only set $K = 20$ for Child with $m = 500$ and set $K = 9$ for German when running the K-best program. Note that because there is no randomness in the outcome of the K-best algorithm, we always ran the K-best program only once to get its fixed SAD for the edge features.

The experimental results of comparing the three methods based on the data case Tic-Tac-Toe are shown in Figures 9 and 10. Figure 9 shows the SAD performance of the three methods with each $i \in \{1, 2, \ldots, 10\}$ in terms of the edge features, where an error bar represents one sample standard deviation $\hat{\sigma}$(SAD) across 20 runs from the DP+MCMC or the IW-DDS at each i. Figure 10 shows $\hat{\mu}(T_t)$ (the sample mean of the total running time) of the DP+MCMC and the IW-DDS as well as T_t (the total running time) of the K-best with each i, where the running time is in seconds. The advantage of our IW-DDS can be clearly seen from Figures 9 and 10. Comparing with the DP+MCMC, for each $i \in \{1, 2, \ldots, 10\}$, the IW-DDS uses shorter running time but has its real mean of SAD significantly smaller than that from the DP+MCMC, with the p-value $< 1 \times 10^{-10}$ from the two-sample t test with unequal variances. Comparing with the K-best, for each $i \in \{1, 2, \ldots, 10\}$, the IW-DDS uses shorter running time but has its real mean of SAD significantly smaller than the SAD from the K-best, with the p-value $< 1 \times 10^{-35}$ from the one-sample t test. Therefore, the learning performance of the IW-DDS is significantly better than that of the other two methods at each i for the data case Tic-Tac-Toe.

The experimental results based on the data case Letter with $m = 500$ are shown in Figures 11 and 12. Just as the description for Figures 9 and 10, Figure 11 shows the SAD performance of the three methods in terms of the edge features, and Figure 12 shows the corresponding time cost of the three methods. The only difference is that in both Figure 11 and Figure 12, the corresponding result of the K-best with $K = 162$ is marked as a star and compared with that of the IW-DDS at $i = 10$. The advantage of our IW-DDS can be clearly seen from Figures 11 and 12. Comparing with the DP+MCMC, for each $i \in \{1, 2, \ldots, 10\}$, the IW-DDS uses much shorter running time but has its real mean of SAD significantly smaller than that from the DP+MCMC, with the p-value < 0.013 from the two-sample t test with unequal variances. ($\hat{\mu}(T_t)$ of the IW-DDS at $i = 10$ is only 33.00 seconds, which is even less than the running time 39.58 seconds of the DP phase of the DP+MCMC method.) Note that $\hat{\sigma}$(SAD) from the DP+MCMC is shown to be very large. The DP+MCMC has its $\hat{\sigma}$(SAD) even larger than its $\hat{\mu}$(SAD) (the sample mean of SAD) when $i \geq 8$, which indicates that the performance of the DP+MCMC is not stable based on the 500,000 MCMC iterations. Comparing with the K-best, the IW-DDS with $N_o = 50{,}000$ uses much shorter running time but has its real mean of SAD significantly smaller than the SAD from the K-best with $K = 162$, because the p-value from the corresponding one-sample t test is less than 1×10^{-35}.

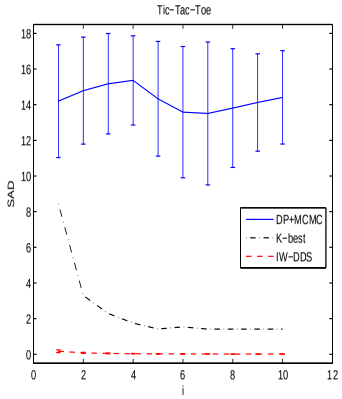


Figure 9: Plot of the SAD Performance of the DP+MCMC, the K-best, and the IW-DDS for Tic-Tac-Toe

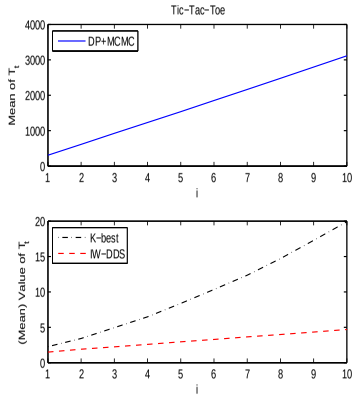


Figure 10: Plot of the Total Running Time of the DP+MCMC, the K-best, and the IW-DDS for Tic-Tac-Toe

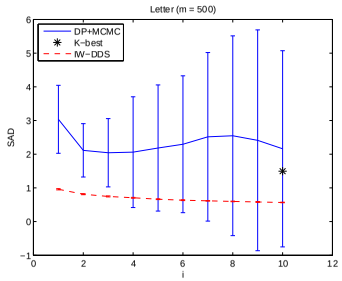


Figure 11: Plot of the SAD Performance of the DP+MCMC, the K-best, and the IW-DDS for Letter ($m = 500$)

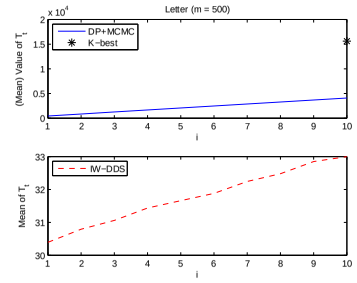


Figure 12: Plot of the Total Running Time of the DP+MCMC, the K-best, and the IW-DDS for Letter ($m = 500$)

Therefore, the learning performance of the IW-DDS is also significantly better than that of the other two methods for the data case Letter with $m = 500$.

The experimental results for the other three data cases Wine, Child with $m = 500$, and German are represented similarly in the supplementary material. The same conclusion about the learning performance can be clearly drawn by examining the figures shown in the supplementary material.

4.3 Learning Performance on Non-modular Features

In Sections 4.1 and 4.2 we did not provide experimental results on the learning performance on non-modular features. We did not do so in Section 4.2 because there is no known method to compute the true/exact posterior probability of any non-modular feature $p_{\not\prec}(f|D)$ except by the brute-force enumeration over all the (super-exponential number of) DAGs so that the quality of the corresponding $\hat{p}_{\not\prec}(f|D)$ learned from any approximate method cannot be precisely measured. We did not do so in Section 4.1 because the current PO-MCMC tool (BEANDisco) only supports the estimation of the posterior of an edge feature so that the comparison of our method and the PO-MCMC can only be made for the edge feature. (Thus, we did not make the comparison for the path feature, which is one particular non-modular feature, though the DP algorithm of Parviainen and Koivisto, 2011 can compute the exact posterior of a path feature $p_{\prec}(f|D)$.) Our idea is that by showing that our algorithms have significantly better performance in computing fundamental structural features (directed edge features), which should be due to the better quality of our DAG samples with respect to the corresponding $p_{\not\prec}(G|D)$ or $p_{\prec}(G|D)$, we expect that our algorithms will also be superior in computing other complicated structural features using the same set of DAG samples.

To verify our expectation, we performed the experiments on the real data set "Iris" (with $n = 5$) from the UCI Machine Learning Repository (Asuncion and Newman, 2007) and the well-studied data set "Coronary" (coronary heart disease) (with $n = 6$) (Edwards, 2000). Because n is small, by enumerating all the DAGs, we were able to compute $p_{\not\prec}(f|D)$, the true posterior probability for any interesting non-modular feature f. For demonstration, we investigated the following five interesting non-modular features. Feature f_1, a directed path feature from node x to node y, denoted by $x \rightsquigarrow y$, represents the situation that variable x eventually influences variable y. Feature f_2, a limited-length directed path feature $x \rightsquigarrow y$ that has its path length no more than 2, represents that variable x can influence variable y via at most one intermediate variable. Feature f_3, a combined path feature $x \rightsquigarrow y \rightsquigarrow z$, can be interpreted as the situation that variable x eventually influences variable y which in turn eventually influences variable z. Feature f_4, a combined path feature $y \leftsquigarrow x \rightsquigarrow z$ with $y \neq z$, means that variable x eventually influences both variable y and variable z. Feature f_5, a combined path feature $y \leftsquigarrow x \not\rightsquigarrow z$ with $x \neq z$, represents that variable x eventually influences variable y but not variable z. Please see Figures 13 to 18 for the example features in the data set Coronary. Then we compared the SAD performance on the (directed) edge feature with the corresponding SAD performance [6] on each feature f_j ($j \in \{1, 2, 3, 4, 5\}$) from the DP+MCMC, the K-best, and the IW-DDS. The experimental results on both data sets show that if the SAD of the IW-DDS is significantly smaller than the SAD of the competing method (the DP+MCMC or the K-best) for the edge feature, then the SAD of the IW-DDS will also be significantly smaller than the SAD of the competing method for each investigated non-modular feature f_j using the same set of DAG samples. Thus, our expectation is supported by the experiments. The detailed experimental results are as follows.

The following is our experimental design for the data set Coronary with $n = 6$ and $m = 1841$. For the IW-DDS, we tried the sample size $N_o = 2{,}500 \cdot i$, where $i \in \{1, 2, \ldots, 20\}$. For each i, we

6. More specifically, SAD = $(\sum_{xy} |p_{\not\prec}(x \rightsquigarrow y|D) - \hat{p}_{\not\prec}(x \rightsquigarrow y|D)|)$ for the path feature $x \rightsquigarrow y$; SAD = $(\sum_{xy} |p_{\not\prec}(x \rightsquigarrow y|D) - \hat{p}_{\not\prec}(x \rightsquigarrow y|D)|)$ for the path feature $x \rightsquigarrow y$ whose length is no more than 2; SAD = $(\sum_{xyz} |p_{\not\prec}(x \rightsquigarrow y \rightsquigarrow z|D) - \hat{p}_{\not\prec}(x \rightsquigarrow y \rightsquigarrow z|D)|)$ for the combined feature $x \rightsquigarrow y \rightsquigarrow z$; SAD = $(\sum_{xyz, y \neq z} |p_{\not\prec}(y \leftsquigarrow x \rightsquigarrow z|D) - \hat{p}_{\not\prec}(y \leftsquigarrow x \rightsquigarrow z|D)|)$ for the combined feature $y \leftsquigarrow x \rightsquigarrow z$ with $y \neq z$; SAD = $(\sum_{xyz, x \neq z} |p_{\not\prec}(y \leftsquigarrow x \not\rightsquigarrow z|D) - \hat{p}_{\not\prec}(y \leftsquigarrow x \not\rightsquigarrow z|D)|)$ for the combined feature $y \leftsquigarrow x \not\rightsquigarrow z$ with $x \neq z$.

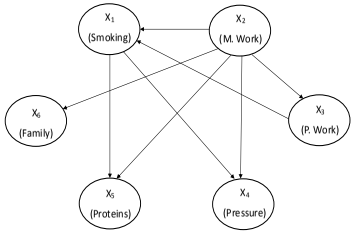


Figure 13: Example Edges in Coronary

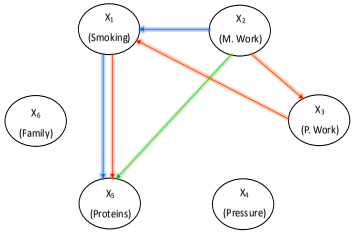


Figure 14: Example f_1 Feature in Coronary

Figure 13 shows every (directed) edge feature e with its true posterior $p_{\not\leftarrow}(e|D) > 0.5$. Figure 14 shows the directed path feature $X_2 \sim\!> X_5$, which has the largest true posterior 0.8348 among all the possible f_1 features. The directed path feature $X_2 \sim\!> X_5$ includes many possible ways going from X_2 to X_5. For clarity, only three possible ways that have every constituent edge with true posterior > 0.5 are shown in Figure 14. These three possible ways are $X_2 \rightarrow X_5$, $X_2 \rightarrow X_1 \rightarrow X_5$, and $X_2 \rightarrow X_3 \rightarrow X_1 \rightarrow X_5$. The figure is best viewed in color.

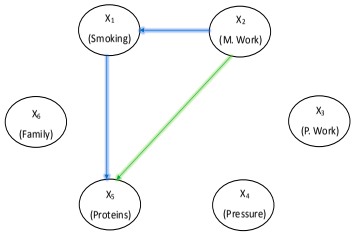


Figure 15: Example f_2 Feature in Coronary

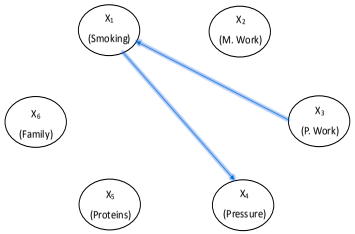


Figure 16: Example f_3 Feature in Coronary

Figure 15 shows the limited-length directed path feature $X_2 \sim\!> X_5$, which has the largest true posterior 0.8348 among all the possible f_2 features. The limited-length directed path feature $X_2 \sim\!> X_5$ includes many possible ways going from X_2 to X_5. For clarity, only two possible ways that have every constituent edge with true posterior > 0.5 are shown in Figure 15. These two possible ways are $X_2 \rightarrow X_5$, and $X_2 \rightarrow X_1 \rightarrow X_5$. Figure 16 shows the combined path feature $X_3 \sim\!> X_1 \sim\!> X_4$, which has the largest true posterior 0.5044 among all the possible f_3 features. The combined path feature $X_3 \sim\!> X_1 \sim\!> X_4$ includes many possible ways going from X_3 via X_1 to X_4. For clarity, only one possible way ($X_3 \rightarrow X_1 \rightarrow X_4$) that has every constituent edge with true posterior > 0.5 is shown in Figure 16.

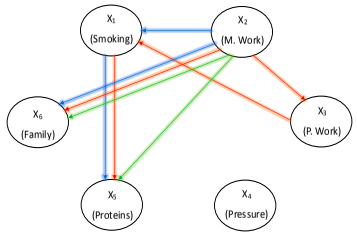


Figure 17: Example f_4 Feature in Coronary

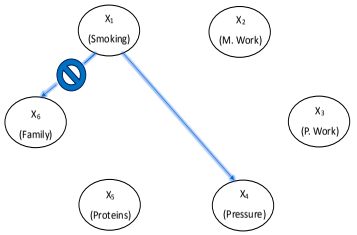


Figure 18: Example f_5 Feature in Coronary

Figure 17 shows the combined path feature $X_5 <\!\sim X_2 \sim\!> X_6$, which has the largest true posterior 0.6020 among all the possible f_4 features. The combined path feature $X_5 <\!\sim X_2 \sim\!> X_6$ includes many possible ways. For clarity, only three possible ways that have every constituent edge with true posterior > 0.5 are shown in Figure 17. These three possible ways are $X_6 \leftarrow X_2 \rightarrow X_5$, $X_6 \leftarrow X_2 \rightarrow X_1 \rightarrow X_5$, and $X_6 \leftarrow X_2 \rightarrow X_3 \rightarrow X_1 \rightarrow X_5$. The figure is best viewed in color. Figure 18 shows the combined path feature $X_4 <\!\sim X_1 \sim\!> X_6$, which has the largest true posterior 0.5139 among all the possible f_5 features. The combined path feature $X_4 <\!\sim X_1 \sim\!> X_6$ excludes any possible way going from X_1 to X_6 but includes many possible ways going from X_1 to X_4. For clarity, only one possible way ($X_1 \rightarrow X_4$) that has every constituent edge with true posterior > 0.5 is shown in Figure 18.

independently ran the IW-DDS 20 times to get the sample mean and the sample standard deviation of SAD for the (directed) edge feature and the five non-modular features (f_1, f_2, f_3, f_4, and f_5). For the DP+MCMC, we ran 25,000 $\cdot i$ MCMC iterations, where $i \in \{1, 2, \ldots, 20\}$. For each i, we discarded the first 12,500 $\cdot i$ MCMC iterations for "burn-in" and set the thinning parameter to be 5, so that 2,500 $\cdot i$ DAGs got sampled. For each i, we independently ran the MCMC 20 times to get the sample mean and the sample standard deviation of SAD for the edge feature, f_1, f_2, f_3, f_4, and f_5. For the K-best, we ran the K-best program with $K = 10 \cdot i$, where $i \in \{1, 2, \ldots, 20\}$. For each i, we ran the K-best just once to get SAD for the edge feature, f_1, f_2, f_3, f_4, and f_5, because there is no randomness in the outcome of the K-best algorithm. [7]

The experimental results for the data set Coronary are demonstrated in Figures 19 to 24. Figure 19 shows the SAD performance of the three methods with each i for the edge feature, where an error bar represents one sample standard deviation across 20 runs for the DP+MCMC or the IW-DDS at each i. Correspondingly, Figures 20 to 24 show the SAD performance of the three methods with each i for the five investigated non-modular features (f_1, f_2, f_3, f_4, and f_5) respectively. From Figure 19 and each of Figures 20 to 24, one can clearly see that if the SAD of the IW-DDS is significantly smaller than the SAD of the competing method (the DP+MCMC or the K-best) for the edge feature, then the SAD of the IW-DDS will also be significantly smaller than the SAD of the competing method for each of the five investigated non-modular features. More

7. The purpose of the experimental setting for the DP+MCMC and the K-best is merely to testify the claimed "if-then" conditional statement: if the SAD performance from the IW-DDS is significantly better than the SAD performance from the competing method for an edge feature, then the SAD performance from the IW-DDS will also be significantly better than that from the competing method for each investigated non-modular feature using the same set of DAG samples.

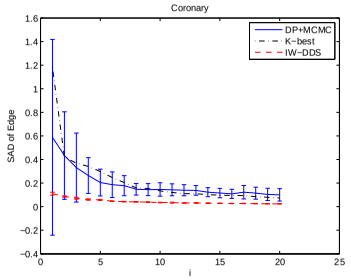


Figure 19: SAD of the Learned Edge Features for Coronary

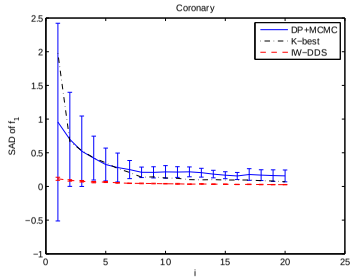


Figure 20: SAD of the Learned f_1 Features for Coronary

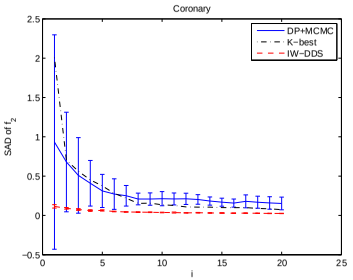


Figure 21: SAD of the Learned f_2 Features for Coronary

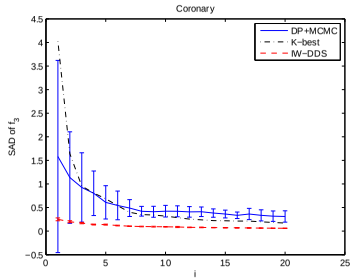


Figure 22: SAD of the Learned f_3 Features for Coronary

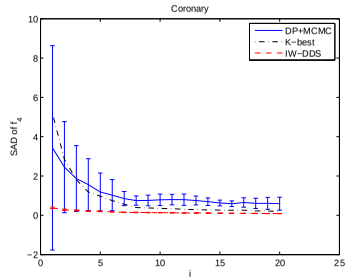


Figure 23: SAD of the Learned f_4 Features for Coronary

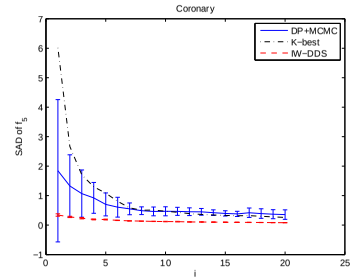


Figure 24: SAD of the Learned f_5 Features for Coronary

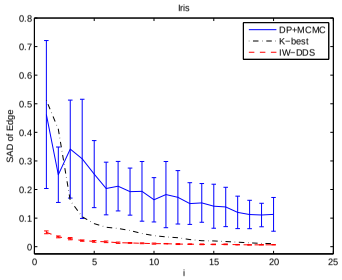


Figure 25: SAD of the Learned Edge Features for Iris

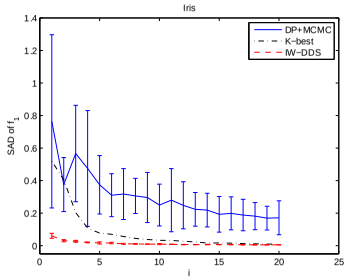


Figure 26: SAD of the Learned f_1 Features for Iris

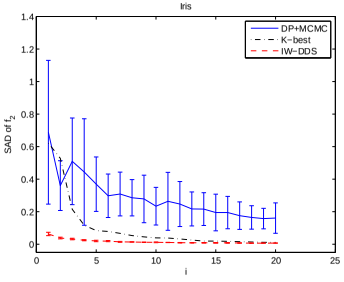


Figure 27: SAD of the Learned f_2 Features for Iris

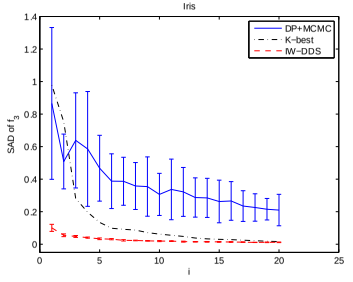


Figure 28: SAD of the Learned f_3 Features for Iris

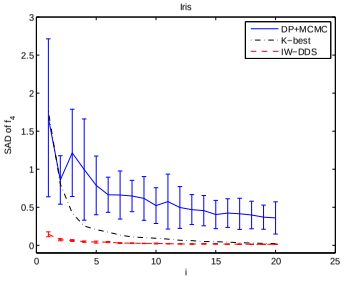


Figure 29: SAD of the Learned f_4 Features for Iris

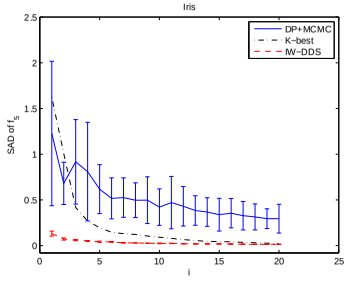


Figure 30: SAD of the Learned f_5 Features for Iris

specifically, comparing with the DP+MCMC, at each $i \in \{1, 2, \dots 20\}$, for the edge feature, the real mean of SAD from the IW-DDS is significantly smaller than that from the DP+MCMC with the p-value < 0.01 from the two-sample t test with unequal variances. Consistently, at each $i \in \{1, 2, \dots 20\}$, for each investigated non-modular feature f_j, the real mean of SAD from the IW-DDS is also significantly smaller than that from the DP+MCMC with the p-value < 0.01 from the same t test. Comparing with the K-best, at each $i \in \{1, 2, \dots 20\}$, for the edge feature, the real mean of SAD from the IW-DDS is significantly smaller than the SAD from the K-best with the p-value $< 1 \times 10^{-20}$ from the one-sample t test. Consistently, at each $i \in \{1, 2, \dots 20\}$, for each investigated non-modular feature f_j, the real mean of SAD from the IW-DDS is also significantly smaller than the SAD from the K-best with the p-value $< 1 \times 10^{-20}$ from the same t test.

We also performed the same kind of experiments for the data set Iris with $n = 5$ and $m = 150$. The results are demonstrated in Figures 25 to 30. Comparing with the DP+MCMC, at each $i \in \{1, 2, \dots 20\}$, just as the comparison result for the edge feature, for each investigated f_j, the real mean of SAD from the IW-DDS is significantly smaller than that from the DP+MCMC with the p-value $< 1 \times 10^{-5}$ from the two-sample t test with unequal variances. Comparing with the K-best, at each $i \in \{1, 2, \dots 20\}$, just as the comparison result for the edge feature, for each investigated f_j, the real mean of SAD from the IW-DDS is also significantly smaller than the SAD from the K-best with the p-value $< 1 \times 10^{-9}$ from the one-sample t test. Thus, a conclusion similar to that from Coronary can be drawn: if the SAD of the IW-DDS is significantly smaller than the SAD of the competing method for the edge feature, then the SAD of the IW-DDS will also be significantly smaller than the SAD of the competing method for each of the five investigated non-modular features.

4.4 Performance Guarantee for the DDS Algorithm

To testify the quality guarantee from Corollary 4 (iv) for the estimator based on the DDS algorithm, we performed experiments based on two data cases (Letter with $m = 100$ and Tic-Tac-Toe), which have relatively large $\hat{\mu}$(SAD) or $\hat{\mu}$(MAD) ($= \hat{\mu}$(SAD) $/(n(n-1))$) from the DDS algorithm shown in Table 1. Based on hypothesis testing, we can conclude with very strong evidence that the performance guarantee for our estimator holds for both data cases. The details of the experiments are as follows.

For the first set of experiments, we choose the data case Letter with $m = 100$, which has the largest $\hat{\mu}$(SAD) ($= 0.2948$) from the DDS among all the 33 data cases shown in Table 1. We first consider the setting of the parameters specified as $\epsilon = 0.02$ and $\delta = 0.05$, which serves as our performance requirement. By setting the DAG sample size $N_o = \lceil (\ln(2/\delta))/(2\epsilon^2) \rceil = 4612$, we intend to show that the estimator $\hat{p}_{\prec}(f|D)$ coming from our DDS has the performance guarantee such that the Hoeffding inequality $P(|\hat{p}_{\prec}(f|D) - p_{\prec}(f|D)| \geq \epsilon) \leq \delta$ holds. Each directed edge feature f is investigated here because the posterior of each edge $p_{\prec}(f|D)$ can be easily obtained by the DP algorithm of Koivisto (2006). For each edge f, we call the event of $|\hat{p}_{\prec}(f|D) - p_{\prec}(f|D)| \geq \epsilon$ as the event of violation (of the pre-specified estimation error bound) in the learning of f. Define the indicator variable W for the event of violation in the learning of f. Thus, W is a Bernoulli random variable with the success probability $p_{vio} = P(|\hat{p}_{\prec}(f|D) - p_{\prec}(f|D)| \geq \epsilon)$. We independently repeat the DDS algorithm (with the same N_o) $R = 400$ times and use the average of W as the estimator $\hat{p}_{vio}$ for each edge. Note that the mean of $\hat{p}_{vio}$ is p_{vio} and the variance of $\hat{p}_{vio}$ is $p_{vio}(1-p_{vio})/R$ because $R\hat{p}_{vio}$ has a binomial distribution with the trial number R and the success probability p_{vio}. Because we expect that p_{vio} will be small so that $p_{vio}(1 - p_{vio})$ will be large as

compared with p_{vio}, we choose large $R = 400$ to make the variance of $\hat{p}_{vio}$ relatively small with respect to the mean of $\hat{p}_{vio}$. Figure 31 shows the histogram of $\hat{p}_{vio}$ for each of all the $n(n-1) = 272$ directed edges. For each of the 272 edges, it can be clearly seen that the corresponding $\hat{p}_{vio}$ is much smaller than $\delta = 0.05$ marked by the vertical bar. For 240 out of the 272 edges, the corresponding $\hat{p}_{vio}$'s are exactly equal to 0. Even for the largest $\hat{p}_{vio} = 0.015$, corresponding to 6 successes among 400 trials, we can use the one-sided hypothesis testing to reject the null hypothesis that $p_{vio} \geq 0.05$ and to conclude that $p_{vio} < 0.05$ with the p-value less than 2×10^{-4}. Therefore, the Hoeffding inequality holds for the learning of each edge in this parameter setting.

Next, we consider another setting of the parameters with $\epsilon = 0.01$ and $\delta = 0.02$, which has a more demanding performance requirement. By setting the DAG sample size $N_o = \lceil (\ln(2/\delta))/(2\epsilon^2) \rceil = 23026$, we want to show that the estimator $\hat{p}_{\prec}(f|D)$ coming from our DDS has the performance guarantee satisfying the Hoeffding inequality. With the same logic, we independently repeat the DDS algorithm (with the same N_o) $R = 1250$ times and use the average of W as the estimator $\hat{p}_{vio}$ for each edge. (Here we choose even larger R because we expect that p_{vio} will be smaller in this parameter setting.) Figure 32 shows the histogram of $\hat{p}_{vio}$ for each of all the 272 directed edges. For each edge, it can be clearly seen that the corresponding $\hat{p}_{vio}$ is much smaller than $\delta = 0.02$. Even for the largest $\hat{p}_{vio} = 0.004$, corresponding to 5 successes among 1250 trials, we can use the one-sided hypothesis testing to reject the null hypothesis that $p_{vio} \geq 0.02$ and to conclude that $p_{vio} < 0.02$ with the p-value less than 2×10^{-6}. Therefore, the Hoeffding inequality also holds in this parameter setting.

For the second set of experiments, we choose the data case Tic-Tac-Toe, which has the largest $\hat{\mu}$(MAD) ($= \hat{\mu}$(SAD) $/(n(n-1)) = 0.1547/90$) from the DDS among all the 33 data cases shown in Table 1. The same kind of experiments are performed for this data case. For the parameter setting with $\epsilon = 0.02$ and $\delta = 0.05$, the corresponding result is shown in Figure 33. For each of the 90 edges, the corresponding $\hat{p}_{vio}$ is clearly much smaller than $\delta = 0.05$. Even for the largest $\hat{p}_{vio} = 0.0125$, corresponding to 5 successes among 400 trials, we can use the one-sided hypothesis testing to conclude that $p_{vio} < 0.05$ with the p-value less than 6×10^{-5}. For the parameter setting with $\epsilon = 0.01$ and $\delta = 0.02$, the corresponding result is shown in Figure 34. For each edge, it can be clearly seen that the corresponding $\hat{p}_{vio}$ is much smaller than $\delta = 0.02$. Even for the largest $\hat{p}_{vio}$ $= 0.0056$, corresponding to 7 successes among 1250 trials, we can use the one-sided hypothesis testing to conclude that $p_{vio} < 0.02$ with the p-value less than 3×10^{-5}. Thus, the Hoeffding inequality also holds in the set of experiments for this data case.

Finally, for the data case Tic-Tac-Toe, we fix $\epsilon = 0.02$ but increase N_o from 2,000 to 10,000 by an increment of 1,000 each time. For each N_o, we plot the Hoeffding bound $2e^{-2N_o\epsilon^2}$ for the probability of violation $p_{vio} = P(|\hat{p}_{\prec}(f|D) - p_{\prec}(f|D)| \geq \epsilon)$ in Figure 35. (Note that the Hoeffding bound decreases at an exponential rate as N_o increases.) Then for each N_o, we also plot both the maximum and the mean of all the $n(n-1)$ $\hat{p}_{vio}$'s in Figure 35. Again $\hat{p}_{vio}$ for each edge is the average of W by independently running the DDS algorithm (with the same N_o) R times. We set $R = \max\{400, \lceil 10/(e^{-2N_o\epsilon^2}) \rceil\}$ and use larger R for larger N_o because we expect that $\hat{p}_{vio}$ will be smaller for larger N_o. From Figure 35, we can clearly see that $\hat{p}^*_{vio}$, the maximum of all the $n(n-1)$ $\hat{p}_{vio}$'s, is always far below the Hoeffding bound for each N_o. Furthermore, for each N_o we can use the one-sided hypothesis testing to reject the null hypothesis that $p_{vio} \geq 2e^{-2N_o\epsilon^2}$ and to conclude that $p_{vio} < 2e^{-2N_o\epsilon^2}$ with the p-value less than 3×10^{-4}. Therefore, the Hoeffding inequality holds for each N_o.

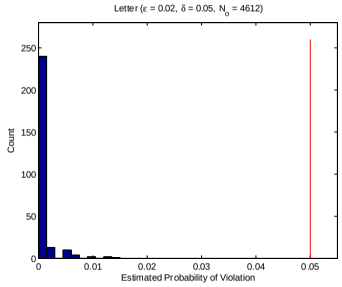


Figure 31: Histogram of Estimated Probabilities of Violation in Edge Learning for Letter ($m = 100$) with $\epsilon = 0.02$, $\delta = 0.05$, and $N_o = 4612$

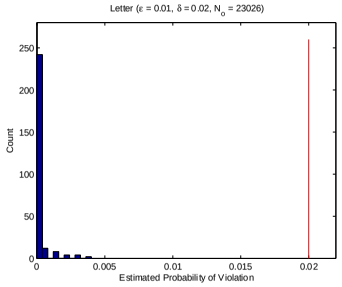


Figure 32: Histogram of Estimated Probabilities of Violation in Edge Learning for Letter ($m = 100$) with $\epsilon = 0.01$, $\delta = 0.02$, and $N_o = 23026$

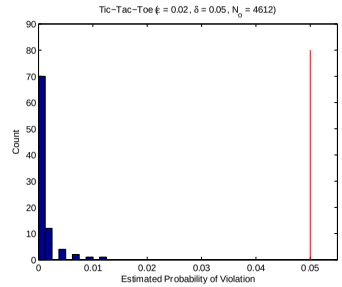


Figure 33: Histogram of Estimated Probabilities of Violation in Edge Learning for Tic-Tac-Toe with $\epsilon = 0.02$, $\delta = 0.05$, and $N_o = 4612$

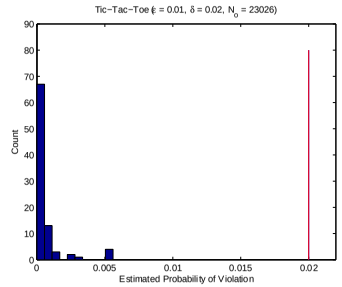


Figure 34: Histogram of Estimated Probabilities of Violation in Edge Learning for Tic-Tac-Toe with $\epsilon = 0.01$, $\delta = 0.02$, and $N_o = 23026$

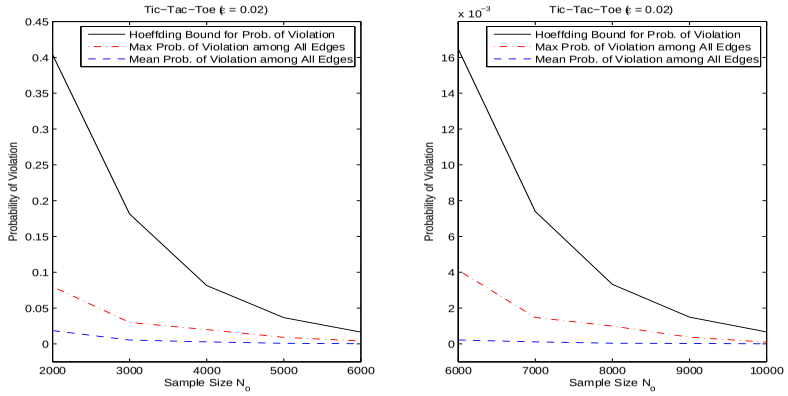


Figure 35: Plot of Probability of Violation versus N_o for Tic-Tac-Toe with $\epsilon = 0.02$

5. Conclusion

We develop new algorithms for efficiently sampling Bayesian network structures (DAGs). The sampled DAGs can then be used to build estimators for the posteriors of any features of interests. Theoretically we show that our estimators have several desirable properties. For example, unlike the existing MCMC algorithms, the estimators based on the DDS algorithm satisfy the Hoeffding bound and therefore enjoy the quality guarantee of the estimation with the given number of samples. Empirically we show that our estimators considerably outperform the previous state-of-the-art estimators with or without assuming the order-modular prior.

Our algorithms are capable of estimating the posteriors of arbitrary (non-modular) features (the DDS under the order-modular prior, and the IW-DDS under the structure-modular prior); while the exact algorithms are available for computing modular features under the order-modular prior with time $O(n^{k+1}C(m)+kn2^n)$ and space $O(n2^n)$ (Koivisto and Sood, 2004; Koivisto, 2006); computing path features under the order-modular prior with time $O(n^{k+1}C(m) + n3^n)$ and space $O(3^n)$ (Parviainen and Koivisto, 2011); and computing modular features under the structure-modular prior with time $O(n^{k+1}C(m) + kn2^n + 3^n)$ and space $O(n2^n)$ (Tian and He, 2009). The bottleneck of our algorithms is their first computation step, the DP algorithm of Koivisto and Sood (2004) whose space cost is $O(n2^n)$. Therefore, the application of our algorithms is limited to the data sets on which the DP algorithm of Koivisto and Sood (2004) is able to run—up to around 25 variables in current desktops, while a parallel implementation of the DP algorithm has been demonstrated on a data set with 33 variables using a cluster including totally 2,048 processors and 8,192 GB memory (Chen et al., 2014).

Appendix A. Proofs of Propositions, Theorems, and Corollary

This appendix provides the proofs of the propositions, theorems, and corollary in the paper.

A.1 Proof of Proposition 1

We first prove a lemma for Proposition 1.

Let an order $\prec$ be represented as $(\sigma_1, \ldots, \sigma_n)$, where σ_i is the ith element in the order.

Lemma 6 *The probability that the last $n-k+1$ elements along the order are $\sigma_k, \sigma_{k+1}, \ldots, \sigma_n$ respectively is given by*

$$p(\sigma_k, \sigma_{k+1}, \ldots, \sigma_n, D) = L'(U_{\sigma_k}^{\prec}) \prod_{i=k}^{n} \alpha'_{\sigma_i}(U_{\sigma_i}^{\prec}),$$

where $U_{\sigma_i}^{\prec} = V - \{\sigma_i, \sigma_{i+1}, \ldots, \sigma_n\}$.

Proof

$$\begin{aligned}
&p(\sigma_k, \sigma_{k+1}, \ldots, \sigma_n, D) \\
&= \sum_{\prec' \in \mathcal{L}(U_{\sigma_k}^{\prec})} p(\prec', \sigma_k, \sigma_{k+1}, \ldots, \sigma_n, D) \\
&= \prod_{i=k}^{n} \alpha'_{\sigma_i}(U_{\sigma_i}^{\prec}) \sum_{\prec' \in \mathcal{L}(U_{\sigma_k}^{\prec})} \prod_{i \in U_{\sigma_k}^{\prec}} \alpha'_i(U_i^{\prec}) \quad \text{(from Eq. 13)} \\
&= L'(U_{\sigma_k}^{\prec}) \prod_{i=k}^{n} \alpha'_{\sigma_i}(U_{\sigma_i}^{\prec}) \quad \text{(from Eq. 9)}.
\end{aligned}$$

■

Proposition 1 can be directly proved by the conclusion of Lemma 6 according to the definition of the conditional probability.

A.2 Proof of Proposition 2

Proof

From Proposition 1 and $L'(U_{\sigma_1}^{\prec} = \emptyset) = 1$, we have

$$\begin{aligned}
&\prod_{k=1}^{n} p(\sigma_k | \sigma_{k+1}, \ldots, \sigma_n, D) \\
&= \frac{\prod_{i=1}^{n} \alpha'_{\sigma_i}(U_{\sigma_i}^{\prec})}{L'(V)} \\
&= \frac{p(\prec, D)}{p_{\prec}(D)} \quad \text{(from Eq. 13 and Eq. 14)} \\
&= p(\prec | D),
\end{aligned}$$

which proves Eq. (20). ■

A.3 Proof of Theorem 3

Proof

First, we show that each DAG G sampled according to our DDS algorithm has its pmf $p_s(G)$ equal to the exact posterior $p_{\prec}(G|D)$ by assuming the order-modular prior.

On one hand, from the following derivation, we can get the exact form for $p_{\prec}(G|D)$:

$$\begin{aligned}
&p_{\prec}(f|D) \\
&= \sum_{\prec} p(\prec | D) p(f | \prec, D) \\
&= \sum_{\prec} p(\prec | D) \sum_{G \subseteq \prec} f(G) p(G | \prec, D) \\
&= \sum_{\prec} \sum_{G \subseteq \prec} f(G) p(\prec, G | D) \\
&= \sum_{G} f(G) \sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec, G | D). \qquad (28)
\end{aligned}$$

Thus, for each possible DAG G_i, by setting $f(G)$ to be the indicator function $I[G = G_i]$ and then relating Eq. (28) with Eq. (4), we know that $p_{\prec}(G_i|D) = \sum_{\prec \text{s.t.} G_i \subseteq \prec} p(\prec, G_i|D)$ for each G_i.

On the other hand, the event that a DAG G gets sampled according to our DDS algorithm occurs if and only if one of the orders that G is consistent with gets sampled in Step 2 of the DDS algorithm and then G gets sampled from the sampled order in Step 3 of the DDS algorithm. Therefore, based on the total probability formula, $p_s(G) = \sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec | D) p(G | \prec, D) = \sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec, G|D)$ $= p_{\prec}(G|D)$.

Second, because N_o orders are sampled independently and each DAG per sampled order is sampled independently, $p_s(G_1, G_2, \ldots, G_{N_o}|D) = \prod_{i=1}^{N_o} [\sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec | D) p(G_i | \prec, D)] = \prod_{i=1}^{N_o}$ $p_{\prec}(G_i|D)$.

Therefore, Theorem 3 is proved.

■

A.4 Proof of Corollary 4

Proof

For each G_i in the DAG set $\{G_1, G_2, \ldots, G_{N_o}\}$ sampled from the DDS algorithm, $f(G_i) \in \{0, 1\}$. Because $G_1, G_2, \ldots, G_{N_o}$ are iid with pmf $p_{\prec}(G|D)$ (by Theorem 3), $f(G_1)$, $f(G_2), \ldots,$ $f(G_{N_o})$ are iid with Bernoulli pmf.

For each G_i, the following is true for $E(f(G_i))$, the expectation of $f(G_i)$:

$$\begin{aligned}
&E(f(G_i)) \\
&= \sum_{G} f(G) p_{\prec}(G|D) \\
&= p_{\prec}(f|D).
\end{aligned}$$

Thus, $f(G_1)$, $f(G_2), \ldots, f(G_{N_o})$ are iid with Bernoulli($p_{\prec}(f|D)$). In other words, $f(G_1)$, $f(G_2), \ldots, f(G_{N_o})$ are independent; and for each G_i, $P(f(G_i) = 1) = p_{\prec}(f|D)$ and $P(f(G_i) = 0) = 1 - p_{\prec}(f|D)$.

(i) Proof that $\hat{p}_\prec(f|D)$ is an unbiased estimator of $p_\prec(f|D)$, that is, $E(\hat{p}_\prec(f|D)) = p_\prec(f|D)$.

$$\begin{aligned}
&E\left(\hat{p}_\prec(f|D)\right)\\
&= E\left(\frac{1}{N_o}\sum_{i=1}^{N_o} f(G_i)\right)\\
&= \frac{1}{N_o}\sum_{i=1}^{N_o} E\left(f(G_i)\right)\\
&= \frac{1}{N_o} N_o p_\prec(f|D)\\
&= p_\prec(f|D).
\end{aligned}$$

(ii) Proof that $\hat{p}_\prec(f|D)$ converges almost surely to $p_\prec(f|D)$.
Because $f(G_1)$, $f(G_2), \ldots, f(G_{N_o})$ are iid with $E(f(G_i)) = p_\prec(f|D) < \infty$ and

$$\hat{p}_\prec(f|D) = \frac{1}{N_o}\sum_{i=1}^{N_o} f(G_i),$$

based on the strong law of large numbers (Theorem 5.5.9 of Casella and Berger, 2002), $\hat{p}_\prec(f|D)$ converges almost surely to $p_\prec(f|D)$ (as $N_o \to \infty$).

Note that, by Theorem 2.5.1 of Athreya and Lahiri (2006), the property that $\hat{p}_\prec(f|D)$ converges almost surely to $p_\prec(f|D)$ implies that $\hat{p}_\prec(f|D)$ converges in probability to $p_\prec(f|D)$, that is, $\hat{p}_\prec(f|D)$ is a consistent estimator of $p_\prec(f|D)$.

(iii) Proof that if $0 < p_\prec(f|D) < 1$, then the random variable

$$\frac{\sqrt{N_o}(\hat{p}_\prec(f|D) - p_\prec(f|D))}{\sqrt{\hat{p}_\prec(f|D)(1-\hat{p}_\prec(f|D))}}$$

has a limiting standard normal distribution, denoted by

$$\frac{\sqrt{N_o}(\hat{p}_\prec(f|D) - p_\prec(f|D))}{\sqrt{\hat{p}_\prec(f|D)(1-\hat{p}_\prec(f|D))}} \longrightarrow^d \mathcal{N}(0,1).$$

Because $f(G_1)$, $f(G_2), \ldots, f(G_{N_o})$ are iid with Bernoulli($p_\prec(f|D)$), for each G_i, the following is true for $\mathrm{Var}(f(G_i))$, the variance of $f(G_i)$:

$$\begin{aligned}
&\mathrm{Var}(f(G_i))\\
&= E(f(G_i))(1 - E(f(G_i)))\\
&= p_\prec(f|D)(1 - p_\prec(f|D))\\
&< \infty.
\end{aligned}$$

Because $0 < p_\prec(f|D) < 1$, $\mathrm{Var}(f(G_i))$ is also strictly greater than 0.
Again, we have already known that $E(f(G_i)) = p_\prec(f|D) < \infty$.
Thus, by the central limit theorem (Theorem 5.5.15 of Casella and Berger, 2002),

$$\frac{\sqrt{N_o}(\hat{p}_\prec(f|D) - E(f(G)))}{\sqrt{\mathrm{Var}(f(G))}} \longrightarrow^d \mathcal{N}(0,1),$$

that is,

$$\frac{\sqrt{N_o}(\hat{p}_\prec(f|D) - p_\prec(f|D))}{\sqrt{p_\prec(f|D)(1-p_\prec(f|D))}} \longrightarrow^d \mathcal{N}(0,1).$$

Because $\hat{p}_\prec(f|D)$ converges in probability to $p_\prec(f|D)$, denoted by $\hat{p}_\prec(f|D) \longrightarrow^p p_\prec(f|D)$, by the continuous mapping theorem (Theorem 5.5.4 of Casella and Berger, 2002),

$$\sqrt{\hat{p}_\prec(f|D)(1-\hat{p}_\prec(f|D))} \longrightarrow^p \sqrt{p_\prec(f|D)(1-p_\prec(f|D))}.$$

Finally, by Slutsky's theorem (Theorem 5.5.17 of Casella and Berger, 2002),

$$\frac{\sqrt{N_o}(\hat{p}_\prec(f|D) - p_\prec(f|D))}{\sqrt{\hat{p}_\prec(f|D)(1-\hat{p}_\prec(f|D))}} \longrightarrow^d \mathcal{N}(0,1).$$

(iv) Proof that for any $\epsilon > 0$, any $0 < \delta < 1$, if $N_o \geq (\ln(2/\delta))/(2\epsilon^2)$, then $P(|\hat{p}_\prec(f|D) - p_\prec(f|D)| < \epsilon) \geq 1 - \delta$.

Because $f(G_1)$, $f(G_2), \ldots, f(G_{N_o})$ are iid with Bernoulli($p_\prec(f|D)$), the Hoeffding bound (Hoeffding, 1963; Koller and Friedman, 2009) holds:

$$P(|\hat{p}_\prec(f|D) - p_\prec(f|D)| \geq \epsilon) \leq 2e^{-2N_o\epsilon^2}.$$

This is equivalent to

$$P(|\hat{p}_\prec(f|D) - p_\prec(f|D)| < \epsilon) \geq 1 - 2e^{-2N_o\epsilon^2} \geq 1 - \delta$$

for $N_o \geq (\ln(2/\delta))/(2\epsilon^2)$.

■

A.5 Proof of Equation (21)

Proof For any $j \in \{1, \ldots, n\}$,

$$\begin{aligned}
&P((\sigma_j, U_{\sigma_j})|D)\\
&\propto P((\sigma_j, U_{\sigma_j}), D)\\
&= \sum_{\substack{(\sigma_1,\ldots,\sigma_{j-1})\\ \in\mathcal{L}(U_{\sigma_j})}} \sum_{\substack{(\sigma_{j+1},\ldots,\sigma_n)\\ \in\mathcal{L}(V-U_{\sigma_j}-\{\sigma_j\})}} P(\sigma_1, \ldots, \sigma_{j-1}, \sigma_j, \sigma_{j+1}, \ldots, \sigma_n, D)\\
&= \sum_{\substack{(\sigma_1,\ldots,\sigma_{j-1})\\ \in\mathcal{L}(U_{\sigma_j})}} \sum_{\substack{(\sigma_{j+1},\ldots,\sigma_n)\\ \in\mathcal{L}(V-U_{\sigma_j}-\{\sigma_j\})}} \prod_{i=1}^{n} \alpha'_{\sigma_i}(U_{\sigma_i})\\
&= \alpha'_{\sigma_j}(U_{\sigma_j}) L'(U_{\sigma_j}) R'(V - U_{\sigma_j} - \{\sigma_j\}).
\end{aligned}$$

■

A.6 Proof of Equation (22)

Proof

$$
\begin{aligned}
& p_{\prec}(G|D) \\
&= \sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec, G|D) \\
&= \frac{1}{p_{\prec}(D)} \sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec, G, D) \\
&= \frac{1}{p_{\prec}(D)} \sum_{\prec \text{s.t.} G \subseteq \prec} p(\prec, G) p(D|G) \\
&= \frac{1}{p_{\prec}(D)} \sum_{\prec \text{s.t.} G \subseteq \prec} \left(\prod_{i=1}^{n} q_i(U_i) \rho_i(Pa_i) \right) p(D|G) \\
&= \frac{1}{p_{\prec}(D)} \sum_{\prec \text{s.t.} G \subseteq \prec} \left(\prod_{i=1}^{n} p_i(Pa_i) \right) p(D|G) \\
&= \frac{1}{p_{\prec}(D)} \sum_{\prec \text{s.t.} G \subseteq \prec} p_{\not\prec}(G) p(D|G) \\
&= \frac{1}{p_{\prec}(D)} \cdot | \prec_G | \cdot p_{\not\prec}(G, D) \\
&= \frac{p_{\not\prec}(D)}{p_{\prec}(D)} \cdot | \prec_G | \cdot p_{\not\prec}(G|D).
\end{aligned}
$$

Because both $p_{\not\prec}(D) > 0$ and $p_{\prec}(D) > 0$, $p_{\prec}(G|D) \propto | \prec_G | \cdot p_{\not\prec}(G|D)$, which also was shown by Ellis and Wong (2008).

■

A.7 Proof of Theorem 5

Proof

Let Ω denote the set of all the DAGs. Define $\mathcal{U}^+ = \{G \in \Omega : p_{\not\prec}(G, D) > 0\}$. $\mathcal{U}^+$ will equal Ω if the user chooses a prior such that $p_{\not\prec}(G) > 0$ for every DAG G (such as a uniform DAG prior $p_{\not\prec}(G) \equiv 1$). If the user, however, has some additional domain knowledge so that he or she sets some prior to exclude some DAGs a priori, $\mathcal{U}^+$ will be a proper subset of Ω. Note that $p_{\not\prec}(f|D) = p_{\not\prec}(f, D)/p_{\not\prec}(D)$, where $p_{\not\prec}(f, D) = \sum_G f(G) p_{\not\prec}(G, D) = \sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D)$, and $p_{\not\prec}(D) = p_{\not\prec}(f \equiv 1, D) = \sum_G p_{\not\prec}(G, D) = \sum_{G \in \mathcal{U}^+} p_{\not\prec}(G, D)$.

Let $I[\cdot]$ denote the indicator function. Rewrite $\hat{p}_{\not\prec}(f|D)$ in Eq. (24) as $\hat{p}_{\not\prec}(f, D)/\hat{p}_{\not\prec}(D)$, where $\hat{p}_{\not\prec}(f, D) = \sum_{G \in \mathcal{G}} f(G) p_{\not\prec}(G, D) = \sum_G f(G)\, p_{\not\prec}(G, D) I[G \in \mathcal{G}] = \sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D) I[G \in \mathcal{G}]$, and $\hat{p}_{\not\prec}(D) = \hat{p}_{\not\prec}(f \equiv 1, D) = \sum_{G \in \mathcal{G}} p_{\not\prec}(G, D) = \sum_G p_{\not\prec}(G, D) I[G \in \mathcal{G}] = \sum_{G \in \mathcal{U}^+} p_{\not\prec}(G, D) I[G \in \mathcal{G}]$.

Note that by Theorem 3, $P(G \in \mathcal{G}) = 1 - (1 - p_{\prec}(G|D))^{N_o}$, where $p_{\prec}(G|D)$ is the exact posterior of G under the order-modular prior assumption. Also note that by Eq. (22), for any $G \in \Omega$, $p_{\not\prec}(G, D) > 0$ implies $p_{\prec}(G|D) > 0$.

(i) Proof that $\hat{p}_{\not\prec}(f|D)$ is an asymptotically unbiased estimator of $p_{\not\prec}(f|D)$, that is,

$$\lim_{N_o \to \infty} E(\hat{p}_{\not\prec}(f|D)) = p_{\not\prec}(f|D). \tag{29}$$

For notational convenience, let γ denote $\hat{p}_{\not\prec}(f, D)$ and τ denote $\hat{p}_{\not\prec}(D)$. Define $g(\gamma, \tau) = \gamma/\tau$ so that $g(\gamma, \tau)$ is $\hat{p}_{\not\prec}(f|D)$.

Note that

$$
\begin{aligned}
& E(\gamma) \\
&= E(\hat{p}_{\not\prec}(f, D)) \\
&= E\left(\sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D) I[G \in \mathcal{G}] \right) \\
&= \sum_{G \in \mathcal{U}^+} E(f(G) p_{\not\prec}(G, D) I[G \in \mathcal{G}]) \\
&= \sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D) P(G \in \mathcal{G}) \\
&= \sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D)(1 - (1 - p_{\prec}(G|D))^{N_o}).
\end{aligned}
$$

Thus,

$$
\begin{aligned}
& \lim_{N_o \to \infty} E(\gamma) \\
&= \lim_{N_o \to \infty} \left(\sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D)(1 - (1 - p_{\prec}(G|D))^{N_o}) \right) \\
&= \sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D) \lim_{N_o \to \infty} (1 - (1 - p_{\prec}(G|D))^{N_o}) \\
&= \sum_{G \in \mathcal{U}^+} f(G) p_{\not\prec}(G, D) \\
&= p_{\not\prec}(f, D).
\end{aligned}
$$

Similarly, by setting $f \equiv 1$, we have

$$
\begin{aligned}
& E(\tau) \\
&= E(\hat{p}_{\not\prec}(D)) \\
&= \sum_{G \in \mathcal{U}^+} p_{\not\prec}(G, D)(1 - (1 - p_{\prec}(G|D))^{N_o}),
\end{aligned}
$$

and

$$\lim_{N_o \to \infty} E(\tau) = p_{\not\prec}(D).$$

Next, by Taylor's theorem (with the Lagrange form of the remainder),

$$\begin{aligned} & g(\gamma,\tau) \\ &= g(E(\gamma),E(\tau)) + \left[\frac{\partial g(\gamma,\tau)}{\partial \gamma}\right]_{\gamma=E(\gamma),\tau=E(\tau)} (\gamma^* - E(\gamma)) \\ &\quad + \left[\frac{\partial g(\gamma,\tau)}{\partial \tau}\right]_{\gamma=E(\gamma),\tau=E(\tau)} (\tau^* - E(\tau)), \end{aligned}$$

where $\gamma^* = E(\gamma) + \theta(\gamma - E(\gamma))$, $\tau^* = E(\tau) + \theta(\tau - E(\tau))$, and θ is a random variable such that $0 < \theta < 1$.

Examine the components separately:

$$g(E(\gamma),E(\tau)) = E(\gamma)/E(\tau);$$

$$\begin{aligned} & \left[\frac{\partial g(\gamma,\tau)}{\partial \gamma}\right]_{\gamma=E(\gamma),\tau=E(\tau)} \\ &= \left[\tau^{-1}\right]_{\gamma=E(\gamma),\tau=E(\tau)} \\ &= (E(\tau))^{-1}; \end{aligned}$$

$$\begin{aligned} & \left[\frac{\partial g(\gamma,\tau)}{\partial \tau}\right]_{\gamma=E(\gamma),\tau=E(\tau)} \\ &= \left[-\gamma(\tau)^{-2}\right]_{\gamma=E(\gamma),\tau=E(\tau)} \\ &= -E(\gamma)(E(\tau))^{-2}. \end{aligned}$$

Because neither $E(\gamma)$ nor $E(\tau)$ is random,

$$\begin{aligned} & E\left(g(\gamma,\tau)\right) \\ &= E(\gamma)/E(\tau) + (E(\tau))^{-1}E(\gamma^* - E(\gamma)) - E(\gamma)(E(\tau))^{-2}E(\tau^* - E(\tau)) \\ &= E(\gamma)/E(\tau) + (E(\tau))^{-1}E(\theta(\gamma - E(\gamma))) - E(\gamma)(E(\tau))^{-2}E(\theta(\tau - E(\tau))). \end{aligned}$$

Consider the limit of each component separately:

$$\begin{aligned} & \lim_{N_o\to\infty} (E(\gamma)/E(\tau)) \\ &= \left(\lim_{N_o\to\infty} E(\gamma)\right) / \left(\lim_{N_o\to\infty} E(\tau)\right) \\ &= p_{\not\propto}(f,D)/p_{\not\propto}(D) \\ &= p_{\not\propto}(f|D); \end{aligned}$$

$$\begin{aligned} & \lim_{N_o\to\infty} (E(\tau))^{-1} \\ &= \left(\lim_{N_o\to\infty} E(\tau)\right)^{-1} \\ &= (p_{\not\propto}(D))^{-1}; \end{aligned}$$

$$\begin{aligned} & \lim_{N_o\to\infty} -E(\gamma)(E(\tau))^{-2} \\ &= -\left(\lim_{N_o\to\infty} E(\gamma)\right)\left(\lim_{N_o\to\infty} E(\tau)\right)^{-2} \\ &= -p_{\not\propto}(f,D)(p_{\not\propto}(D))^{-2}. \end{aligned}$$

Note that both $(p_{\not\propto}(D))^{-1}$ and $-p_{\not\propto}(f,D)(p_{\not\propto}(D))^{-2}$ are constant real numbers.
Finally, we intend to prove the following two equalities:

$$\lim_{N_o\to\infty} E(\theta(\gamma - E(\gamma))) = 0, \tag{30}$$

$$\lim_{N_o\to\infty} E(\theta(\tau - E(\tau))) = 0. \tag{31}$$

Once this is done,

$$\begin{aligned} & \lim_{N_o\to\infty} E\left(g(\gamma,\tau)\right) \\ &= \lim_{N_o\to\infty} (E(\gamma)/E(\tau)) + \lim_{N_o\to\infty} (E(\tau))^{-1}\cdot 0 + \lim_{N_o\to\infty} \left(-E(\gamma)(E(\tau))^{-2}\right)\cdot 0 \\ &= p_{\not\propto}(f|D). \end{aligned}$$

The whole proof of Eq. (29) will then be done.
The proof of Eq. (30) is as follows.
Based on the definition of the limit, proving Eq. (30) is equivalent to proving

$$\lim_{N_o\to\infty} |E(\theta(\gamma - E(\gamma)))| = 0. \tag{32}$$

Because

$$\begin{aligned} & |E(\theta(\gamma - E(\gamma)))| \\ &\le E\left(|\theta(\gamma - E(\gamma))|\right) \\ &= E\left(|\theta|\cdot|\gamma - E(\gamma)|\right) \\ &\le E\left(|\gamma - E(\gamma)|\right) \quad \text{(because } 0 < \theta < 1\text{)}, \end{aligned}$$

and $|E(\theta(\gamma - E(\gamma)))| \ge 0$, to prove Eq. (32), it is sufficient to prove

$$\lim_{N_o\to\infty} E\left(|\gamma - E(\gamma)|\right) = 0. \tag{33}$$

Note that

$$
\begin{aligned}
& E\left(|\gamma - E(\gamma)|\right) \\
&= E\left(\left|\sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)I[G\in\mathcal{G}] \ - \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)P(G\in\mathcal{G})\right|\right) \\
&= E\left(\left|\sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)(I[G\in\mathcal{G}] - P(G\in\mathcal{G}))\right|\right) \\
&\le E\left(\sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)\,|I[G\in\mathcal{G}] - P(G\in\mathcal{G})|\right) \\
&= \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)E\left(|I[G\in\mathcal{G}] - P(G\in\mathcal{G})|\right) \\
&= \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)[(1-P(G\in\mathcal{G}))P(G\in\mathcal{G}) \\
&\qquad + (P(G\in\mathcal{G})-0)(1-P(G\in\mathcal{G}))] \\
&= \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)[2P(G\in\mathcal{G})(1-P(G\in\mathcal{G}))] \\
&= \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)[2(1-(1-p_{\prec}(G|D))^{N_o}) \times (1-p_{\prec}(G|D))^{N_o}].
\end{aligned}
$$

Because for any $G\in\mathcal{U}^+$, $p_{\prec}(G|D)>0$, we have

$$
\begin{aligned}
& \lim_{N_o\to\infty} \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)[2(1-(1-p_{\prec}(G|D))^{N_o}) \times (1-p_{\prec}(G|D))^{N_o}] \\
&= \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D) \lim_{N_o\to\infty} [2(1-(1-p_{\prec}(G|D))^{N_o}) \times (1-p_{\prec}(G|D))^{N_o}] \\
&= \sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)\cdot 0 \\
&= 0.
\end{aligned}
$$

Eq. (33) is proved, and the proof of Eq. (30) is done.

Setting $f\equiv 1$ in Eq. (30) leads to Eq. (31).

Thus, the whole proof of Eq. (29) is done.

(ii) Proof that $\hat{p}_{\not\prec}(f|D)$ converges almost surely to $p_{\not\prec}(f|D)$, denoted by $\hat{p}_{\not\prec}(f|D) \longrightarrow^{a.s.} p_{\not\prec}(f|D)$.

Recall that Ω denotes the set of all the DAGs, that is, $\Omega=\{G_1,G_2,\ldots,G_{W^*}\}$, where W^* is $|\Omega|$, the number of all the DAGs. Note that W^* is a finite positive integer though it is super-exponential in the number of variables n.

Let $\mathcal{F}$ be $\mathcal{P}(\Omega)$, the power set of Ω. Thus, $\mathcal{F}$ is a σ-algebra on Ω (Athreya and Lahiri, 2006). Define for any $A\in\mathcal{F}$, $\mu(A)=\sum_{G_j\in A} p_{\prec}(G_j|D)$. It is well-known that μ is a probability measure on $\mathcal{F}$ so that $(\Omega,\mathcal{F},\mu)$ is a probability space (Athreya and Lahiri, 2006).

For each $i\ge 1$, let $\Omega_i=\Omega$, $\mathcal{F}_i=\mathcal{F}$, and $\mu_i=\mu$. Let $\Omega^\infty=\{(G^{(1)},G^{(2)},\ldots): G^{(i)}\in\Omega_i, i\ge 1\}$. Let a cylinder set $\underset{\sim}{A}=A_1\times A_2\times\cdots\times A_k\times\Omega_{k+1}\times\Omega_{k+2}\times\cdots$, where there exists $1\le k<\infty$ such that $A_i\in\mathcal{F}_i$ for $1\le i\le k$ and $A_i=\Omega_i$ for $i>k$. Let $\mathcal{F}^\infty=\sigma<\{\underset{\sim}{A}: \underset{\sim}{A} \text{ is a cylinder set }\}>$, that is, $\mathcal{F}^\infty$ is a σ-algebra generated by the set of all the $\underset{\sim}{A}$'s. $\mathcal{F}^\infty$ is a σ-algebra on Ω^∞ and is called a product σ-algebra.

Define, for each $(A_1,A_2,\ldots,A_k,\Omega_{k+1},\Omega_{k+2},\ldots)\in\mathcal{F}^\infty$, $\mu^\infty(A_1,A_2,\ldots,A_k,\Omega_{k+1},\Omega_{k+2},\ldots)=\mu_1(A_1)\times\mu_2(A_2)\times\cdots\times\mu_k(A_k)$. By Kolmogorov's consistency theorem, $(\Omega^\infty,\mathcal{F}^\infty,\mu^\infty)$ is a probability space (Athreya and Lahiri, 2006).

Let $\Omega^{\infty,0}=\{(G^{(1)},G^{(2)},\ldots)\in\Omega^\infty:$ there exists $G\in\mathcal{U}^+$ such that for any $i\ge 1, G^{(i)}\ne G\}$. Let $\Omega^{\infty,1}=\Omega^\infty-\Omega^{\infty,0}$. Thus, $\Omega^{\infty,1}=\{(G^{(1)},G^{(2)},\ldots)\in\Omega^\infty:$ for any $G\in\mathcal{U}^+,$ there exists $i\ge 1$ such that $G^{(i)}=G\}$.

Define, for each $\omega^\infty\in\Omega^\infty$,

$$
\hat{p}^{N_o}_{\not\prec}(\omega^\infty)=\frac{\sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)I[G\in\mathcal{G}^{N_o}(\omega^\infty)]}{\sum_{G\in\mathcal{U}^+} p_{\not\prec}(G,D)I[G\in\mathcal{G}^{N_o}(\omega^\infty)]},
$$

where $\mathcal{G}^{N_o}(\omega^\infty)$ is the DAG set that includes the first N_o coordinates of ω^∞. By the definition, we know that $\hat{p}^{N_o}_{\not\prec}(\omega^\infty)=\hat{p}_{\not\prec}(f|D)$.

For each $\omega^\infty\in\Omega^{\infty,1}$, for each $G\in\mathcal{U}^+$, let $N(G,\omega^\infty)$ be the smallest integer such that $G^{(N(G,\omega^\infty))}=G$. Let $N(\omega^\infty)=\max_{G\in\mathcal{U}^+} N(G,\omega^\infty)$. Then for each $N_o\ge N(\omega^\infty)$, for each $G\in\mathcal{U}^+$, $I[G\in\mathcal{G}^{N_o}(\omega^\infty)]=1$.

Accordingly, for each $\omega^\infty\in\Omega^{\infty,1}$, there exists $N(\omega^\infty)$ such that for each $N_o\ge N(\omega^\infty)$,

$$
\begin{aligned}
& \hat{p}^{N_o}_{\not\prec}(\omega^\infty) \\
&= \frac{\sum_{G\in\mathcal{U}^+} f(G)p_{\not\prec}(G,D)}{\sum_{G\in\mathcal{U}^+} p_{\not\prec}(G,D)} \\
&= p_{\not\prec}(f|D).
\end{aligned}
$$

This implies that $\lim_{N_o\to\infty}\hat{p}^{N_o}_{\not\prec}(\omega^\infty)=p_{\not\prec}(f|D)$ for each $\omega^\infty\in\Omega^{\infty,1}$.

Finally, we intend to prove the following equality:

$$
\mu^\infty(\Omega^{\infty,1})=1. \tag{34}
$$

Once this is done, the whole proof that $\hat{p}_{\not\prec}(f|D)\longrightarrow^{a.s.} p_{\not\prec}(f|D)$ is done.

Proving Eq. (34) is equivalent to proving

$$
\mu^\infty(\Omega^{\infty,0})=0. \tag{35}
$$

For any $G\in\Omega$, let $\Omega^{\infty,0,G}=\{(G^{(1)},G^{(2)},\ldots)\in\Omega^\infty:$ for any $i\ge 1, G^{(i)}\ne G\}$. Thus, $\Omega^{\infty,0}=\bigcup_{G\in\mathcal{U}^+}\Omega^{\infty,0,G}$. Accordingly, $\mu^\infty(\Omega^{\infty,0})\le\sum_{G\in\mathcal{U}^+}\mu^\infty(\Omega^{\infty,0,G})$.

For each $j\ge 1$, let $\Omega^{\infty,0,G,j}=\{(G^{(1)},G^{(2)},\ldots)\in\Omega^\infty:$ for any $i\in\{1,\ldots,j\}, G^{(i)}\ne G\}$. Note that $\Omega^{\infty,0,G,1}\supseteq\Omega^{\infty,0,G,2}\supseteq\ldots\supseteq\Omega^{\infty,0,G,j-1}\supseteq\Omega^{\infty,0,G,j}$ for each $j\ge 1$. Thus, $\Omega^{\infty,0,G}=\bigcap_{j=1}^\infty\Omega^{\infty,0,G,j}$.

Finally, for any $G \in \mathcal{U}^+$,

$$\begin{aligned}
&\mu^\infty(\Omega^{\infty,0,G}) \\
&= \mu^\infty(\bigcap_{j=1}^{\infty} \Omega^{\infty,0,G,j}) \\
&= \lim_{j\to\infty} \mu^\infty(\Omega^{\infty,0,G,j}) \\
&= \lim_{j\to\infty} \mu_1(\Omega - \{G\}) \times \mu_2(\Omega - \{G\}) \times \cdots \times \mu_j(\Omega - \{G\}) \\
&= \lim_{j\to\infty} [1 - \mu(\{G\})]^j \\
&= \lim_{j\to\infty} [1 - p_\prec(G|D)]^j \\
&= 0.
\end{aligned}$$

Thus, $\sum_{G\in\mathcal{U}^+} \mu^\infty(\Omega^{\infty,0,G}) = 0$, so that Eq. (35) is proved. The whole proof is done.

Note that, by Theorem 2.5.1 of Athreya and Lahiri (2006), the property that $\hat{p}_\prec(f|D)$ converges almost surely to $p_\prec(f|D)$ implies that $\hat{p}_\prec(f|D)$ converges in probability to $p_\prec(f|D)$, that is, $\hat{p}_\prec(f|D)$ is a consistent estimator of $p_\prec(f|D)$.

(iii) Proof that the convergence rate of $\hat{p}_\prec(f|D)$ is $o(a^{N_o})$ for any $0 < a < 1$.

In the proof of (ii), we have shown that for each $\omega^\infty \in \Omega^{\infty,1}$, there exists $N(\omega^\infty)$ such that for each $N_o \geq N(\omega^\infty)$, $\hat{p}^{N_o}_\prec(\omega^\infty) = p_\prec(f|D)$. This means that for any $0 < a < 1$, $(a^{N_o})^{-1}[\hat{p}^{N_o}_\prec(\omega^\infty) - p_\prec(f|D)] = 0$. Thus, $\lim_{N_o\to\infty}(a^{N_o})^{-1}[\hat{p}^{N_o}_\prec(\omega^\infty) - p_\prec(f|D)] = 0$ so that the proof is done.

(iv) Proof that if the quantity $\Delta = \sum_{G\in\mathcal{G}} p_\prec(G|D)$, then $\Delta \cdot \hat{p}_\prec(f|D) \leq p_\prec(f|D) \leq \Delta \cdot \hat{p}_\prec(f|D) + 1 - \Delta$.

The proof is essentially the same as the proof of Proposition 1 of Tian et al. (2010) which proves Eq. (27), an equivalent form of Eq. (26). The direct proof of Eq. (26) is also provided in the supplementary material.

■

References

Arthur Asuncion and David Newman. UCI machine learning repository, 2007.

Krishna Athreya and Soumendra Lahiri. Measure Theory and Probability Theory. Springer, 2006.

John Binder, Daphne Koller, Stuart Russell, and Keiji Kanazawa. Adaptive probabilistic networks with hidden variables. Machine Learning, 29:213–244, 1997.

Graham Brightwell and Peter Winkler. Counting linear extensions is #P-complete. In Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing, pages 175–181, 1991.

George Casella and Roger Berger. Statistical Inference. Duxbury, 2002.

Yetian Chen, Jin Tian, Olga Nikolova, and Srinivas Aluru. A parallel algorithm for exact Bayesian structure discovery in Bayesian networks. CoRR, abs/1408.1664, 2014. URL http://arxiv.org/abs/1408.1664.

Gregory Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. Machine Learning, 9:309–347, 1992.

James Cussens. Bayesian network learning with cutting planes. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 153–160, 2011.

James Cussens and Mark Bartlett. Advances in Bayesian network learning using integer programming. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 182–191, 2013.

Denver Dash and Gregory Cooper. Model averaging for prediction with discrete Bayesian networks. Journal of Machine Learning Research, 5:1177–1203, 2004.

Daniel Eaton and Kevin Murphy. Bayesian structure learning using dynamic programming and MCMC. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 101–108, 2007.

David Edwards. Introduction to Graphical Modelling. Springer, 2nd edition, 2000.

Byron Ellis and Wing Wong. Learning causal Bayesian network structures from experimental data. Journal of the American Statistical Association, 103:778–789, 2008.

Natalia Flerova, Emma Rollon, and Rina Dechter. Bucket and mini-bucket schemes for m best solutions over graphical models. In Graph Structures for Knowledge Representation and Reasoning, volume 7205, pages 91–118. 2012.

Nir Friedman and Daphne Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. Machine Learning, 50:95–125, 2003.

Nir Friedman, Moises Goldszmidt, and Abraham Wyner. Data analysis with Bayesian networks: A bootstrap approach. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 196–205, 1999.

Marco Grzegorczyk and Dirk Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. Machine Learning, 71:265–305, 2008.

David Heckerman, Dan Geiger, and David Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning, 20:197–243, 1995.

David Heckerman, Christopher Meek, and Gregory Cooper. A Bayesian approach to causal discovery. In Computation, Causation, and Discovery, pages 141–166, 1999.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association, 58:13–30, 1963.

Tommi Jaakkola, David Sontag, Amir Globerson, and Marina Meila. Learning Bayesian network structure using LP relaxations. In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), pages 358–365, 2010.

Mark Jerrum, Leslie Valiant, and Vijay Vazirani. Random generation of combinatorial structures from a uniform distribution. Theoretical Computer Science, 43:169–188, 1986.

Robert Kennes and Philippe Smets. Computational aspects of the Möbius transformation. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 401–416, 1991.

Mikko Koivisto. Advances in exact Bayesian structure discovery in Bayesian networks. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 241–248, 2006.

Mikko Koivisto and Kismat Sood. Exact Bayesian structure discovery in Bayesian networks. Journal of Machine Learning Research, 5:549–573, 2004.

Daphne Koller and Nir Friedman. Probabilistic Graphical Models: Principles and Techniques. The MIT Press, 2009.

David Madigan and Jeremy York. Bayesian graphical models for discrete data. International Statistical Review, 63:215–232, 1995.

Brandon Malone and Changhe Yuan. Evaluating anytime algorithms for learning optimal Bayesian networks. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 381–390, 2013.

Brandon Malone, Changhe Yuan, and Eric Hansen. Memory-efficient dynamic programming for learning optimal Bayesian networks. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pages 1057–1062, 2011a.

Brandon Malone, Changhe Yuan, Eric Hansen, and Susan Bridges. Improving the scalability of optimal Bayesian network learning with external-memory frontier breadth-first branch and bound search. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 479–488, 2011b.

Marina Meila and Tommi Jaakkola. Tractable Bayesian learning of tree belief networks. Statistics and Computing, 16:77–92, 2006.

Teppo Niinimaki and Mikko Koivisto. Annealed importance sampling for structure learning in Bayesian networks. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pages 1579–1585, 2013.

Teppo Niinimaki, Pekka Parviainen, and Mikko Koivisto. Partial order MCMC for structure discovery in Bayesian networks. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 557–564, 2011.

Pekka Parviainen and Mikko Koivisto. Bayesian structure discovery in Bayesian networks with less space. In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), pages 589–596, 2010.

Pekka Parviainen and Mikko Koivisto. Ancestor relations in the presence of unobserved variables. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), pages 581–596, 2011.

Judea Pearl. Causality: Models, Reasoning, and Inference. Cambridge University Press, NY, 2000.

Tomi Silander and Petri Myllymaki. A simple approach for finding the globally optimal Bayesian network structure. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 445–452, 2006.

Peter Spirtes, Clark Glymour, and Richard Scheines. Causation, Prediction, and Search. MIT Press, Cambridge, MA, 2nd edition, 2001.

Jin Tian and Ru He. Computing posterior probabilities of structural features in Bayesian networks. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 538–547, 2009.

Jin Tian, Ru He, and Lavanya Ram. Bayesian model averaging using the k-best Bayesian network structures. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 589–597, 2010.

Ioannis Tsamardinos, Laura Brown, and Constantin Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning, 65:31–78, 2006.

Changhe Yuan and Brandon Malone. An improved admissible heuristic for learning optimal Bayesian networks. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI), pages 924–933, 2012.

Changhe Yuan and Brandon Malone. Learning optimal Bayesian networks: A shortest path perspective. Journal of Artificial Intelligence Research, 48:23–65, 2013.

Changhe Yuan, Brandon Malone, and Xiaojian Wu. Learning optimal Bayesian networks using A* search. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pages 2186–2191, 2011.

Journal of Machine Learning Research 17 (2016) 1-44 Submitted 12/14; Revised 4/15; Published 7/16

Spectral Methods Meet EM: A Provably Optimal Algorithm for Crowdsourcing

Yuchen Zhang YUCZHANG@EECS.BERKELEY.EDU
Department of Electrical Engineering and Computer Science
University of California, Berkeley, Berkeley, CA 94720, USA

Xi Chen XCHEN3@STERN.NYU.EDU
Stern School of Business
New York University, New York, NY 10012, USA

Dengyong Zhou DENGYONG.ZHOU@MICROSOFT.COM
Microsoft Research
1 Microsoft Way, Redmond, WA 98052, USA

Michael I. Jordan JORDAN@STAT.BERKELEY.EDU
Department of Electrical Engineering and Computer Science and Department of Statistics
University of California, Berkeley, Berkeley, CA 94720, USA

Editor: Alexander Ihler

Abstract

Crowdsourcing is a popular paradigm for effectively collecting labels at low cost. The Dawid-Skene estimator has been widely used for inferring the true labels from the noisy labels provided by non-expert crowdsourcing workers. However, since the estimator maximizes a non-convex log-likelihood function, it is hard to theoretically justify its performance. In this paper, we propose a two-stage efficient algorithm for multi-class crowd labeling problems. The first stage uses the spectral method to obtain an initial estimate of parameters. Then the second stage refines the estimation by optimizing the objective function of the Dawid-Skene estimator via the EM algorithm. We show that our algorithm achieves the optimal convergence rate up to a logarithmic factor. We conduct extensive experiments on synthetic and real datasets. Experimental results demonstrate that the proposed algorithm is comparable to the most accurate empirical approach, while outperforming several other recently proposed methods.

Keywords: crowdsourcing, spectral methods, EM, Dawid-Skene model, non-convex optimization, minimax rate

1. Introduction

With the advent of online services such as Amazon Mechanical Turk, crowdsourcing has become an efficient and inexpensive way to collect labels for large-scale data. However, labels collected from the crowd can be of low quality since crowdsourcing workers are often non-experts and sometimes unreliable. As a remedy, most crowdsourcing services resort to labeling redundancy, collecting multiple labels from different workers for each item. Such a strategy raises a fundamental problem in crowdsourcing: how to infer true labels from noisy but redundant worker labels?

For labeling tasks with k different categories, Dawid and Skene (1979) develop a maximum likelihood approach to this problem based on the EM algorithm. They assume that each worker is associated with a $k \times k$ confusion matrix, where the (l, c)-th entry represents the probability that a random chosen item in class l is labeled as class c by the worker. The true labels and worker confusion matrices are jointly estimated by maximizing the likelihood of the observed labels, where the unobserved true labels are treated as latent variables.

Although this EM-based approach has had empirical success (Snow et al., 2008; Raykar et al., 2010; Liu et al., 2012; Zhou et al., 2012; Chen et al., 2013; Zhou et al., 2014), there is as yet no theoretical guarantee for its performance. A recent theoretical study (Gao and Zhou, 2014) shows that the global optimal solutions of the Dawid-Skene estimator can achieve minimax rates of convergence in a simplified scenario, where the labeling task is binary and each worker has a single parameter to represent her labeling accuracy (referred to as the "one-coin" model in what follows). However, since the likelihood function is nonconvex, this guarantee is not operational because the EM algorithm can get trapped in a local optimum. Several alternative approaches have been developed that aim to circumvent the theoretical deficiencies of the EM algorithm, still the context of the one-coin model (Karger et al., 2013, 2014; Ghosh et al., 2011; Dalvi et al., 2013), but, as we survey in Section 2, they either fail to achieve an optimal rate or make restrictive assumptions that can be hard to justify in practice.

We propose a computationally efficient and provably optimal algorithm to simultaneously estimate true labels and worker confusion matrices for multi-class labeling problems. Our approach is a two-stage procedure, in which we first compute an initial estimate of worker confusion matrices using a spectral method, and then in the second stage we turn to the EM algorithm. Under some mild conditions, we show that this two-stage procedure achieves minimax rates of convergence up to a logarithmic factor, even after only one iteration of EM. In particular, given any $\delta \in (0, 1)$, we provide an upper bound on the number of workers and the number of items so that our method can correctly estimate labels for all items with probability at least $1 - \delta$. We also establish a matching lower bound. Further, we provide both upper and lower bounds for estimating the confusion matrix of each worker and show that our algorithm achieves the optimal accuracy.

This work not only provides an optimal algorithm for crowdsourcing but provides new general insight into the method of moments. Empirical studies show that when the spectral method is used as an initialization for the EM algorithm, it outperforms EM with random initialization (Liang, 2013; Chaganty and Liang, 2013). This work provides a concrete way to justify such observations theoretically. It is also known that starting from a root-n consistent estimator obtained by the spectral method, one Newton-Raphson step leads to an asymptotically optimal estimator (Lehmann and Casella, 2003). However, obtaining a root-n consistent estimator and performing a Newton-Raphson step can be demanding computationally. In contrast, our initialization doesn't need to be root-n consistent, thus a small portion of data suffices to initialize. Moreover, performing one iteration of EM is computationally more attractive and numerically more robust than a Newton-Raphson step especially for high-dimensional problems.

The paper is organized as follows. In Section 2, we provide background on crowdsourcing and the method of moments for latent variables models. In Section 3, we describe our crowdsourcing problem. Our provably optimal algorithm is presented in Section 4. Section

5 is devoted to theoretical analysis (with the proofs gathered in the Appendix). In Section 6, we consider the special case of the one-coin model. A simpler algorithm is introduced together with a sharper rate. Numerical results on both synthetic and real datasets are reported in Section 7, followed by our conclusions in Section 8.

2. Related Work

Many methods have been proposed to address the problem of estimating true labels in crowdsourcing (Whitehill et al., 2009; Raykar et al., 2010; Welinder et al., 2010; Ghosh et al., 2011; Liu et al., 2012; Zhou et al., 2012; Dalvi et al., 2013; Karger et al., 2014, 2013; Parisi et al., 2014; Zhou et al., 2014). The methods in Raykar et al. (2010); Ghosh et al. (2011); Karger et al. (2014); Liu et al. (2012); Karger et al. (2013); Dalvi et al. (2013) are based on the generative model proposed by Dawid and Skene (1979). In particular, Ghosh et al. (2011) propose a method based on Singular Value Decomposition (SVD) which addresses binary labeling problems under the one-coin model. The analysis in Ghosh et al. (2011) assumes that the labeling matrix is full, that is, each worker labels all items. To relax this assumption, Dalvi et al. (2013) propose another SVD-based algorithm which explicitly considers the sparsity of the labeling matrix in both algorithm design and theoretical analysis. Karger et al. (2014) propose an iterative algorithm for binary labeling problems under the one-coin model and extended it to multi-class labeling tasks by converting a k-class problem into $k-1$ binary problems (Karger et al., 2013). This line of work assumes that tasks are assigned to workers according to a random regular graph, thus imposes specific constraints on the number of workers and the number of items. In Section 5, we compare our theoretical results with that of existing approaches (Ghosh et al., 2011; Dalvi et al., 2013; Karger et al., 2014, 2013). The methods in Raykar et al. (2010); Liu et al. (2012); Chen et al. (2013) incorporate Bayesian inference into the Dawid-Skene estimator by assuming a prior over confusion matrices. Zhou et al. (2012, 2014) propose a minimax entropy principle for crowdsourcing which leads to an exponential family model parameterized with worker ability and item difficulty. When all items have zero difficulty, the exponential family model reduces to the generative model suggested by Dawid and Skene (1979).

Our method for initializing the EM algorithm in crowdsourcing is inspired by recent work using spectral methods to estimate latent variable models (Anandkumar et al., 2014, 2015, 2012, 2013; Chaganty and Liang, 2013; Zou et al., 2013; Hsu et al., 2012; Jain and Oh, 2014). The basic idea in this line of work is to compute third-order empirical moments from the data and then to estimate parameters by computing a certain orthogonal decomposition of tensor derived from the moments. Given the special symmetric structure of the moments, the tensor factorization can be computed efficiently using the robust tensor power method (Anandkumar et al., 2014). A problem with this approach is that the estimation error can have a poor dependence on the condition number of the second-order moment matrix and thus empirically it sometimes performs worse than EM with multiple random initializations. Our method, by contrast, requires only a rough initialization from the moment of moments; we show that the estimation error does not depend on the condition number (see Theorem 4 (b)).

Recently, Balakrishnan et al. (2016) study the convergence rate of EM algorithm under a good initialization, which belongs to a ball centered at the true parameter. They show that

when the radius of the ball is small enough to satisfy certain gradient stability condition and sample deviation condition, EM has a geometric convergence rate. Although this is an insightful theoretical result, Balakrishnan et al. (2016) fail to provide a practical approach to constructing such an initialization.

Other related work is by Dasgupta and Schulman (2007), who study an EM algorithm for learning mixture of Gaussians under certain initialization. They establish a nearly optimal estimation precision when using a two-round EM algorithm to learn the parameters from a mixture of k well-separated spherical Gaussians in the high-dimensional space. The space dimension d is assumed to be much greater than $\log(k)$. Although the high-level idea is quite similar to ours (i.e., constructing a good initializer and running one or two steps of EM), our work is different from this work in several respects. First, while Dasgupta and Schulman (2007) require the Gaussian means to be well separated, we do not assume the workers' labeling distributions for different true labels to be separated. In fact, for any worker, we allow his/her labeling distributions for two classes to be arbitrarily close or even identical. We only assume that there is a partitioning of the workers into three groups, such that the averaged confusion matrix of each group has full rank. Second, Dasgupta and Schulman (2007) consider the high-dimensional case where the dimension of the parameter space is $d \gg \log(k)$. In crowdsourcing, the labeling distribution lies in a k-dimensional space where k is the number of classes. Typically k is a small integer (below 10). Finally, in terms of analysis technique, Dasgupta and Schulman (2007) heavily rely on Gaussian concentration results. Our work uses a variety of techniques for different theorems. In particular, for Theorem 3, we repeatedly use matrix perturbation and matrix concentration results to establish the sample complexity for the spectral method. For Theorem 4, we create three random events in equation (33) (that holds with high probability), then establish both a prediction error bound and a confusion matrix estimation error bound for the EM algorithm. We use Le Cam's method (Yu, 1997) to establish the minimax lower bound result in Theorem 5.

3. Problem Setting

Throughout this paper, $[a]$ denotes the integer set $\{1, 2, \ldots, a\}$ and $\sigma_b(A)$ denotes the b-th largest singular value of matrix A. Suppose that there are m workers, n items and k classes. The true label y_j of item $j \in [n]$ is assumed to be sampled from a probability distribution $\mathbb{P}[y_j = l] = w_l$ where $\{w_l : l \in [k]\}$ are positive values satisfying $\sum_{l=1}^{k} w_l = 1$. Denote by a vector $z_{ij} \in \mathbb{R}^k$ the label that worker i assigns to item j. When the assigned label is c, we write $z_{ij} = e_c$, where e_c represents the c-th canonical basis vector in $\mathbb{R}^k$ in which the c-th entry is 1 and all other entries are 0. A worker may not label every item. Let π_i indicate the probability that worker i labels a randomly chosen item. If item j is not labeled by worker i, we write $z_{ij} = 0$. Our goal is to estimate the true labels $\{y_j : j \in [n]\}$ from the observed labels $\{z_{ij} : i \in [m], j \in [n]\}$.

For this estimation purpose, we need to make assumptions on the process of generating observed labels. Following the work of Dawid and Skene (1979), we assume that the probability that worker i labels an item in class l as class c is independent of any particular chosen item, that is, it is a constant over $j \in [n]$. Let us denote the constant probability by μ_{ilc}. Let $\mu_{il} = [\mu_{il1}\ \mu_{il2}\ \cdots\ \mu_{ilk}]^T$. The matrix $C_i = [\mu_{i1}\ \mu_{i2}\ \ldots\ \mu_{ik}] \in \mathbb{R}^{k\times k}$ is called the *confusion*

Algorithm 1: Estimating confusion matrices

Input: integer k, observed labels $z_{ij} \in \mathbb{R}^k$ for $i \in [m]$ and $j \in [n]$.

Output: confusion matrix estimates $\widehat{C}_i \in \mathbb{R}^{k\times k}$ for $i \in [m]$.

(1) Partition the workers into three disjoint and non-empty group G_1, G_2 and G_3. Compute the group aggregated labels Z_{gj} by Eq. (1).

(2) For $(a, b, c) \in \{(2, 3, 1), (3, 1, 2), (1, 2, 3)\}$, compute the second and the third order moments $\widehat{M}_2 \in \mathbb{R}^{k\times k}$, $\widehat{M}_3 \in \mathbb{R}^{k\times k\times k}$ by Eq. (2a)-(2d), then compute $\widehat{C}_c^\diamond \in \mathbb{R}^{k\times k}$ and $\widehat{W} \in \mathbb{R}^{k\times k}$ by tensor decomposition:

  (a) Compute whitening matrix $\widehat{Q} \in \mathbb{R}^{k\times k}$ (such that $\widehat{Q}^T \widehat{M}_2 \widehat{Q} = I$) using SVD.

  (b) Compute eigenvalue-eigenvector pairs $\{(\widehat{\alpha}_h, \widehat{v}_h)\}_{h=1}^k$ of the whitened tensor $\widehat{M}_3(\widehat{Q}, \widehat{Q}, \widehat{Q})$ by using the robust tensor power method. Then compute $\widehat{w}_h = \widehat{\alpha}_h^{-2}$ and $\widehat{\mu}_h^\diamond = (\widehat{Q}^T)^{-1}(\widehat{\alpha}_h \widehat{v}_h)$.

  (c) For $l = 1, \ldots, k$, set the l-th column of $\widehat{C}_c^\diamond$ by some $\widehat{\mu}_h^\diamond$ whose l-th coordinate has the greatest component, then set the l-th diagonal entry of $\widehat{W}$ by $\widehat{w}_h$.

(3) Compute $\widehat{C}_i$ by Eq. (5).

matrix of worker i. In the special case of the one-coin model, all the diagonal elements of C_i are equal to a constant while all the off-diagonal elements are equal to another constant such that each column of C_i sums to 1.

4. Our Algorithm

In this section, we present an algorithm to estimate the confusion matrices and true labels. Our algorithm consists of two stages. In the first stage, we compute an initial estimate for the confusion matrices via the method of moments. In the second stage, we perform the standard EM algorithm by taking the result of the Stage 1 as an initialization.

4.1 Stage 1: Estimating confusion matrices

Partitioning the workers into three disjoint and non-empty groups G_1, G_2 and G_3, the outline of this stage is the following: we use the method of moments to estimate the averaged confusion matrices for the three groups, then utilize this intermediate estimate to obtain the confusion matrix of each individual worker. In particular, for $g \in \{1, 2, 3\}$ and $j \in [n]$, we calculate the averaged labeling within each group by

$$Z_{gj} := \frac{1}{|G_g|} \sum_{i\in G_g} z_{ij}. \tag{1}$$

Denoting the aggregated confusion matrix columns by

$$\mu_{gl}^\diamond := \mathbb{E}(Z_{gj}|y_j = l) = \frac{1}{|G_g|} \sum_{i\in G_g} \pi_i \mu_{il},$$

our first step is to estimate $C_g^\diamond := [\mu_{g1}^\diamond, \mu_{g2}^\diamond, \ldots, \mu_{gk}^\diamond]$ and to estimate the distribution of true labels $W := \mathrm{diag}(w_1, w_2, \ldots, w_k)$. The following proposition shows that we can solve for $C_g^\diamond$ and W from the moments of $\{Z_{gj}\}$.

Proposition 1 (Anandkumar et al. (2015)) *Assume that the vectors $\{\mu_{g1}^\diamond, \mu_{g2}^\diamond, \ldots, \mu_{gk}^\diamond\}$ are linearly independent for each $g \in \{1, 2, 3\}$. Let (a, b, c) be a permutation of $\{1, 2, 3\}$. Define*

$$\begin{aligned}
Z'_{aj} &:= \mathbb{E}[Z_{cj} \otimes Z_{bj}] \left(\mathbb{E}[Z_{aj} \otimes Z_{bj}]\right)^{-1} Z_{aj}, \\
Z'_{bj} &:= \mathbb{E}[Z_{cj} \otimes Z_{aj}] \left(\mathbb{E}[Z_{bj} \otimes Z_{aj}]\right)^{-1} Z_{bj}, \\
M_2 &:= \mathbb{E}[Z'_{aj} \otimes Z'_{bj}], \\
M_3 &:= \mathbb{E}[Z'_{aj} \otimes Z'_{bj} \otimes Z_{cj}].
\end{aligned}$$

Then,

$$M_2 = \sum_{l=1}^k w_l \, \mu_{cl}^\diamond \otimes \mu_{cl}^\diamond \quad \textit{and} \quad M_3 = \sum_{l=1}^k w_l \, \mu_{cl}^\diamond \otimes \mu_{cl}^\diamond \otimes \mu_{cl}^\diamond.$$

Since we only have finite samples, the expectations in Proposition 1 must be approximated by empirical moments. In particular, they are computed by averaging over indices $j = 1, 2, \ldots, n$. For each permutation $(a, b, c) \in \{(2, 3, 1), (3, 1, 2), (1, 2, 3)\}$, we compute

$$\widehat{Z}'_{aj} := \Big(\frac{1}{n}\sum_{j=1}^n Z_{cj} \otimes Z_{bj}\Big)\Big(\frac{1}{n}\sum_{j=1}^n Z_{aj} \otimes Z_{bj}\Big)^{-1} Z_{aj}, \tag{2a}$$

$$\widehat{Z}'_{bj} := \Big(\frac{1}{n}\sum_{j=1}^n Z_{cj} \otimes Z_{aj}\Big)\Big(\frac{1}{n}\sum_{j=1}^n Z_{bj} \otimes Z_{aj}\Big)^{-1} Z_{bj}, \tag{2b}$$

$$\widehat{M}_2 := \frac{1}{n}\sum_{j=1}^n \widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj}, \tag{2c}$$

$$\widehat{M}_3 := \frac{1}{n}\sum_{j=1}^n \widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} \otimes Z_{cj}. \tag{2d}$$

The statement of Proposition 1 suggests that we can recover the columns of $C_c^\diamond$ and the diagonal entries of W by operating on the moments $\widehat{M}_2$ and $\widehat{M}_3$. This is implemented by the tensor factorization method in Algorithm 1. In particular, the tensor factorization algorithm returns a set of vectors $\{(\widehat{\mu}_h^\diamond, \widehat{w}_h) : h = 1, \ldots, k\}$, where each $(\widehat{\mu}_h^\diamond, \widehat{w}_h)$ estimates a particular column of $C_c^\diamond$ (for some $\mu_{cl}^\diamond$) and a particular diagonal entry of W (for some w_l).

It is important to note that the tensor factorization algorithm doesn't provide a one-to-one correspondence between the recovered column and the true columns of $C_c^\diamond$. Thus, $\widehat{\mu}_1^\diamond, \ldots, \widehat{\mu}_k^\diamond$ represents an arbitrary permutation of the true columns.

To discover the index correspondence, we take each $\widehat{\mu}_h^\diamond$ and examine its greatest component. We assume that within each group, the probability of assigning a correct label is always greater than the probability of assigning any specific incorrect label. This assumption will be made precise in the next section. As a consequence, if $\widehat{\mu}_h^\diamond$ corresponds to the l-th column of $C_c^\diamond$, then its l-th coordinate is expected to be greater than other coordinates. Thus, we set the l-th column of $\widehat{C}_c^\diamond$ to some vector $\widehat{\mu}_h^\diamond$ whose l-th coordinate has the greatest component (if there are multiple such vectors, then randomly select one of them; if there is no such vector, then randomly select a $\widehat{\mu}_h^\diamond$). Then, we set the l-th diagonal entry of $\widehat{W}$ to the scalar $\widehat{w}_h$ associated with $\widehat{\mu}_h^\diamond$. Note that by iterating over $(a,b,c) \in \{(2,3,1),(3,1,2),(1,2,3)\}$, we obtain $\widehat{C}_c^\diamond$ for $c = 1,2,3$ respectively. There will be three copies of $\widehat{W}$ estimating the same matrix W—we average them for the best accuracy.

In the second step, we estimate each individual confusion matrix C_i. The following proposition shows that we can recover C_i from the moments of $\{z_{ij}\}$.

Proposition 2 *For any $g \in \{1,2,3\}$ and any $i \in G_g$, let $a \in \{1,2,3\}\backslash\{g\}$ be one of the remaining group index. Then*

$$\pi_i C_i W (C_a^\diamond)^T = \mathbb{E}[z_{ij} Z_{aj}^T].$$

Proof First, notice that

$$\mathbb{E}[z_{ij} Z_{aj}^T] = \mathbb{E}\left[\mathbb{E}[z_{ij} Z_{aj}^T | y_j]\right] = \sum_{l=1}^k w_l \mathbb{E}\left[z_{ij} Z_{aj}^T | y_j = l\right]. \tag{3}$$

Since z_{ij} for $1 \le i \le m$ are conditionally independent given y_j, we can write

$$\mathbb{E}\left[z_{ij} Z_{aj}^T | y_j = l\right] = \mathbb{E}\left[z_{ij} | y_j = l\right] \mathbb{E}\left[Z_{aj}^T | y_j = l\right] = (\pi_i \mu_{il})(\mu_{al}^\diamond)^T. \tag{4}$$

Combining (3) and (4) implies the desired result,

$$\mathbb{E}[z_{ij} Z_{aj}^T] = \pi_i \sum_{l=1}^k w_l \mu_{il} (\mu_{al}^\diamond)^T = \pi_i C_i W (C_a^\diamond)^T.$$

■

Proposition 2 suggests a plug-in estimator for C_i. We compute $\widehat{C}_i$ using the empirical approximation of $\mathbb{E}[z_{ij} Z_{aj}^T]$ and using the matrices $\widehat{C}_a^\diamond$, $\widehat{C}_b^\diamond$, $\widehat{W}$ obtained in the first step. Concretely, we calculate

$$\widehat{C}_i := \text{normalize}\left\{\left(\frac{1}{n}\sum_{j=1}^n z_{ij} Z_{aj}^T\right)\left(\widehat{W}(\widehat{C}_a^\diamond)^T\right)^{-1}\right\}, \tag{5}$$

where the normalization operator rescales the matrix columns, making sure that each column sums to 1. The overall procedure for Stage 1 is summarized in Algorithm 1.

4.2 Stage 2: EM algorithm

The second stage is devoted to refining the initial estimate provided by Stage 1. The joint likelihood of true label y_j and observed labels z_{ij}, as a function of confusion matrices μ_i, can be written as

$$L(\mu; y, z) := \prod_{j=1}^n \prod_{i=1}^m \prod_{c=1}^k (\mu_{iy_jc})^{\mathbb{I}(z_{ij}=e_c)}.$$

By assuming a uniform prior over y, we maximize the marginal log-likelihood function

$$\ell(\mu) := \log\left(\sum_{y\in[k]^n} L(\mu; y, z)\right). \tag{6}$$

We refine the initial estimate of Stage 1 by maximizing the objective function (6), which is implemented by the Expectation Maximization (EM) algorithm. The EM algorithm takes as initialization the values $\{\widehat{\mu}_{ilc}\}$ provided as output by Stage 1, and then executes the following E-step and M-step for at least one round.

E-step Calculate the expected value of the log-likelihood function, with respect to the conditional distribution of y given z under the current estimate of μ:

$$Q(\mu) := \mathbb{E}_{y|z,\widehat{\mu}}\left[\log(L(\mu; y, z))\right] = \sum_{j=1}^n \left\{\sum_{l=1}^k \widehat{q}_{jl} \log\left(\prod_{i=1}^m \prod_{c=1}^k (\mu_{ilc})^{\mathbb{I}(z_{ij}=e_c)}\right)\right\},$$

$$\text{where} \quad \widehat{q}_{jl} \leftarrow \frac{\exp\left(\sum_{i=1}^m \sum_{c=1}^k \mathbb{I}(z_{ij}=e_c)\log(\widehat{\mu}_{ilc})\right)}{\sum_{l'=1}^k \exp\left(\sum_{i=1}^m \sum_{c=1}^k \mathbb{I}(z_{ij}=e_c)\log(\widehat{\mu}_{il'c})\right)} \qquad \text{for } j \in [n],\ l \in [k]. \tag{7}$$

M-step Find the estimate $\widehat{\mu}$ that maximizes the function $Q(\mu)$:

$$\widehat{\mu}_{ilc} \leftarrow \frac{\sum_{j=1}^n \widehat{q}_{jl} \mathbb{I}(z_{ij}=e_c)}{\sum_{c'=1}^k \sum_{j=1}^n \widehat{q}_{jl} \mathbb{I}(z_{ij}=e_{c'})} \qquad \text{for } i \in [m],\ l \in [k],\ c \in [k]. \tag{8}$$

In practice, we alternatively execute the updates (7) and (8), for one iteration or until convergence. Each update increases the objective function $\ell(\mu)$. Since $\ell(\mu)$ is not concave, the EM update doesn't guarantee converging to the global maximum. It may converge to distinct local stationary points for different initializations. Nevertheless, as we prove in the next section, it is guaranteed that the EM algorithm will output statistically optimal estimates of true labels and worker confusion matrices if it is initialized by Algorithm 1.

5. Convergence Analysis

To state our main theoretical results, we first need to introduce some notation and assumptions. Let

$$w_{\min} := \min\{w_l\}_{l=1}^k \quad \text{and} \quad \pi_{\min} := \min\{\pi_i\}_{i=1}^m$$

be the smallest portion of true labels and the most extreme sparsity level of workers. Our first assumption assumes that both $w_{\min}$ and $\pi_{\min}$ are strictly positive, that is, every class and every worker contributes to the dataset.

Our second assumption assumes that the confusion matrices for each of the three groups, namely $C_1^\diamond$, $C_2^\diamond$ and $C_3^\diamond$, are nonsingular. As a consequence, if we define matrices S_{ab} and tensors T_{abc} for any $a, b, c \in \{1, 2, 3\}$ as

$$S_{ab} := \sum_{l=1}^{k} w_l \, \mu_{al}^\diamond \otimes \mu_{bl}^\diamond = C_a^\diamond W (C_b^\diamond)^T \quad \text{and} \quad T_{abc} := \sum_{l=1}^{k} w_l \, \mu_{al}^\diamond \otimes \mu_{bl}^\diamond \otimes \mu_{cl}^\diamond,$$

then there will be a positive scalar σ_L such that $\sigma_k(S_{ab}) \geq \sigma_L > 0$.

Our third assumption assumes that within each group, the average probability of assigning a correct label is always higher than the average probability of assigning any incorrect label. To make this statement rigorous, we define a quantity

$$\kappa := \min_{g \in \{1,2,3\}} \min_{l \in [k]} \min_{c \in [k] \setminus \{l\}} \{\mu_{gll}^\diamond - \mu_{glc}^\diamond\}$$

indicating the smallest gap between diagonal entries and non-diagonal entries in the confusion matrix. The assumption requires that κ is strictly positive. Note that this assumption is group-based, thus doesn't assume the accuracy of any individual worker.

Finally, we introduce a quantity that measures the average ability of workers in identifying distinct labels. For two discrete distributions P and Q, let $\mathbb{D}_{\mathrm{KL}}(P, Q) := \sum_i P(i) \log(P(i)/Q(i))$ represent the KL-divergence between P and Q. Since each column of the confusion matrix represents a discrete distribution, we can define the following quantity:

$$\overline{D} = \min_{l \neq l'} \frac{1}{m} \sum_{i=1}^{m} \pi_i \mathbb{D}_{\mathrm{KL}}(\mu_{il}, \mu_{il'}). \tag{9}$$

The quantity $\overline{D}$ lower bounds the averaged KL-divergence between two columns. If $\overline{D}$ is strictly positive, it means that every pair of labels can be distinguished by at least one subset of workers. As the last assumption, we assume that $\overline{D}$ is strictly positive.

The following two theorems characterize the performance of our algorithm. We split the convergence analysis into two parts. Theorem 3 characterizes the performance of Algorithm 1, providing sufficient conditions for achieving an arbitrarily accurate initialization. We provide the proof of Theorem 3 in Appendix A.

Theorem 3 *For any scalar $\delta > 0$ and any scalar ϵ satisfying $\epsilon \leq \min\left\{\frac{36\kappa k}{\pi_{\min} w_{\min} \sigma_L}, 2\right\}$, if the number of items n satisfies*

$$n = \Omega\left(\frac{k^5 \log((k+m)/\delta)}{\epsilon^2 \pi_{\min}^2 w_{\min}^2 \sigma_L^{13}}\right),$$

then the confusion matrices returned by Algorithm 1 are bounded as

$$\|\widehat{C}_i - C_i\|_\infty \leq \epsilon \qquad \text{for all } i \in [m],$$

with probability at least $1 - \delta$. Here, $\|\cdot\|_\infty$ denotes the element-wise ℓ_∞-norm of a matrix.

Theorem 4 characterizes the error rate in Stage 2. It states that when a sufficiently accurate initialization is taken, the updates (7) and (8) refine the estimates $\widehat{\mu}$ and $\widehat{y}$ to the optimal accuracy. See Appendix B for the proof.

Theorem 4 *Assume that $\mu_{ilc} \geq \rho$ holds for all $(i, l, c) \in [m] \times [k]^2$. For any scalar $\delta > 0$, if confusion matrices $\widehat{C}_i$ are initialized in a way such that*

$$\|\widehat{C}_i - C_i\|_\infty \leq \alpha := \min\left\{\frac{\rho}{2}, \frac{\rho\overline{D}}{16}\right\} \qquad \text{for all } i \in [m] \tag{10}$$

and the number of workers m and the number of items n satisfy

$$m = \Omega\left(\frac{\log(1/\rho)\log(kn/\delta)}{\overline{D}}\right) \quad \text{and} \quad n = \Omega\left(\frac{\log(mk/\delta)}{\pi_{\min} w_{\min} \alpha^2}\right),$$

then, for $\widehat{\mu}$ and $\widehat{q}$ obtained by iterating (7) and (8) (for at least one round), with probability at least $1 - \delta$,

(a) Let $\widehat{y}_j = \arg\max_{l \in [k]} \widehat{q}_{jl}$, then $\widehat{y}_j = y_j$ holds for all $j \in [n]$.

(b) $\|\widehat{\mu}_{il} - \mu_{il}\|_2^2 \leq \frac{48\log(8mk/\delta)}{\pi_i w_l n}$ holds for all $(i, l) \in [m] \times [k]$.

In Theorem 4, the assumption that all confusion matrix entries are lower bounded by $\rho > 0$ is somewhat restrictive. For datasets violating this assumption, we enforce positive confusion matrix entries by adding random noise: Given any observed label z_{ij}, we replace it by a random label in $\{1, ..., k\}$ with probability $k\rho$. In this modified model, every entry of the confusion matrix is lower bounded by ρ, so that Theorem 4 holds. The random noise makes the constant $\overline{D}$ smaller than its original value, but the change is minor for small ρ.

To see the consequence of the convergence analysis, we take error rate ϵ in Theorem 3 equal to the constant α defined in Theorem 4. Then we combine the statements of the two theorems. This shows that if we choose the number of workers m and the number of items n such that

$$m = \widetilde{\Omega}\left(\frac{1}{\overline{D}}\right) \quad \text{and} \quad n = \widetilde{\Omega}\left(\frac{k^5}{\pi_{\min}^2 w_{\min}^2 \sigma_L^{13} \min\{\rho^2, (\rho\overline{D})^2\}}\right); \tag{11}$$

that is, if both m and n are lower bounded by a problem-specific constant and logarithmic terms, then with high probability, the predictor $\widehat{y}$ will be perfectly accurate, and the estimator $\widehat{\mu}$ will be bounded as $\|\widehat{\mu}_{il} - \mu_{il}\|_2^2 \leq \widetilde{\mathcal{O}}(1/(\pi_i w_l n))$. To show the optimality of this convergence rate, we present the following minimax lower bounds. See Appendix C for the proof.

Theorem 5 *There are universal constants $c_1 > 0$ and $c_2 > 0$ such that:*

(a) For any $\{\mu_{ilc}\}$, $\{\pi_i\}$ and any number of items n, if the number of workers $m \le 1/(4\overline{D})$, then

$$\inf_{\widehat{y}} \sup_{v\in[k]^n} \mathbb{E}\Big[\sum_{j=1}^{n} \mathbb{I}(\widehat{y}_j \neq y_j)\Big|\{\mu_{ilc}\},\{\pi_i\}, y = v\Big] \ge c_1 n.$$

(b) For any $\{w_l\}$, $\{\pi_i\}$, any worker-item pair (m,n) and any pair of indices $(i,l) \in [m]\times[k]$, we have

$$\inf_{\widehat{\mu}} \sup_{\mu\in\mathbb{R}^{m\times k\times k}} \mathbb{E}\Big[\|\widehat{\mu}_{il} - \mu_{il}\|_2^2\Big|\{w_l\},\{\pi_i\}\Big] \ge c_2 \min\left\{1, \frac{1}{\pi_i w_l n}\right\}.$$

In part (a) of Theorem 5, we see that the number of workers should be at least $1/\overline{D}$, otherwise any predictor will make many mistakes. This lower bound matches our sufficient condition on the number of workers m (see Eq. (11)). In part (b), we see that the best possible estimate for μ_{il} has $1/(\pi_i w_l n)$ mean-squared error. It verifies the optimality of our estimator $\widehat{\mu}_{il}$. It is also worth noting that the constraint on the number of items n (see Eq. (11)) depends on problem-specific constants, which might be improvable. Nevertheless, the constraint scales logarithmically with m and $1/\delta$, thus is easy to satisfy for reasonably large datasets.

5.1 Discussion of theoretical results

In this section, we present a discussion of the foregoing theoretical results. In particular, we compare Theorem 3 and Theorem 4 to existing theoretical results on crowdsourcing and the EM method.

5.1.1 Sparse sampling regime vs. dense sampling regime

In our theoretical analysis, we make the assumption that the minimum labeling frequency $\pi_{\min}$ is bounded away from zero, i.e., $\pi_{\min} = \Omega(1)$. This corresponds to the *dense sampling regime* where the average number of labels for each item should be $\Theta(m)$. According to Theorem 4, to guarantee perfect label prediction with probability $1-\delta$, we require the average number of samples for each item to be $\Omega\left(\frac{\log(1/\delta)\pi_{\min}}{D}\right)$. Two related works in the dense sampling regime include Ghosh et al. (2011) and Gao and Zhou (2014). Ghosh et al. (2011) studied the one-coin model for binary labeling. To attain a δ prediction error, their algorithm requires m and n to scale with $1/\delta^2$, while our algorithm allows m and n to scale with $\log(1/\delta)$. Gao and Zhou (2014) studied the minimax rate of the prediction error and showed that maximizing likelihood is statistically optimal. However, they didn't provide a polynomial-time algorithm to achieve the minimax rate.

Another sampling regime is the *sparse sampling regime*, where the labeling frequency $\pi_{\min}$ goes to zero as the number of items n goes to infinity. In fact, this is a practical regime for large-scale datasets when workers complete only a vanishing fraction of the total tasks. As can be seen in condition (11), our theoretical result doesn't apply to the very sparse regime when $\pi = o(1/\sqrt{n})$. Several work have been devoted to investigate the sparse sampling regime (Karger et al., 2014, 2013; Dalvi et al., 2013). Under the sparse sampling regime, the high-probability recovery of *all* true labels might be impossible. However, it is still of great interest to establish the upper bound on the prediction error as a function of the average number of labels per item. Karger et al. (2014, 2013) show that if worker labels are organized by a random regular bipartite graph, then the number of labels on each item should scale as $\log(1/\delta)$, where δ is the label prediction error. Their analysis assumes that the limit of number of items goes to infinity, or that the number of workers is many times of the number of items. Dalvi et al. (2013) provide algorithms that improve the theoretical guarantee for the one-coin model. Their algorithms succeed without the regular bipartite graph assumption, and without the requirement that the limit of number of items goes to infinity.

Although our theoretical analysis does not fully cover the sparse sampling regime, the algorithm sill applies to the sparse regime and achieves reasonably good performance (see, e.g., the experiment with TREC data in Section 7.2). Empirically, spectral-initialized EM is rather robust for both dense and sparse sampling regimes.[1]

5.1.2 Lower bound on the number of items

In both Theorem 3 and 4, we require lower bounds on the number of times. It is interesting to see whether these bounds can be improved. One idea is to improve those lower bounds using the technique from Balakrishnan et al. (2016) discussed in Section 2. Nevertheless, as we explain below, such improvement is up to a multiplicative factor $1/\alpha$, which doesn't depend on σ_L, $w_{\min}$ and $\pi_{\min}$. We recall that there are two lower bounds on the number of items n in our theoretical results.

1. The lower bound in the condition of Theorem 3 (denoted by n_{spectral}). It relies on the target error ϵ, the minimum singular value σ_L and the minimum prior probabilities $\pi_{\min}, w_{\min}$.

2. The lower bound in the condition of Theorem 4 (denoted by n_{EM}). It establishes the performance guarantee for EM. This lower bound relies on the initialization accuracy α and the minimum prior probabilities $\pi_{\min}, w_{\min}$.

Theorem 4 shows that it is sufficient to set the target error of Theorem 3 equal to $\epsilon := \alpha$. Thus, the constant n_{spectral} also depends on α. Due to the additional dependence on σ_L, the condition on n_{spectral} is more restrictive than that on n_{EM}.

The result of Balakrishnan et al. (2016) provides conditions (e.g., gradient stability condition and certain sample deviation condition) under which the EM method has geometric convergence rate. Assuming that these conditions are weaker than ours in Theorem 4, then instead of requiring the initialization condition $\|\widehat{C}_i - C_i\|_\infty \le \alpha$ for all $i \in [m]$ as in equation (10), we may have another constant $\alpha' > \alpha$, such that $\|C_i - \widehat{C}_i\|_\infty \le \alpha'$ ensures the linear convergence of EM. (It might be difficult to find the largest α' that makes the conditions of Balakrishnan et al. (2016) hold). The best possible value of α' is 1 since any entry of $\widehat{C}_i$ and C_i is bounded by 1. This means that we can potentially improve n_{spectral} by

1. An anonymous reviewer pointed out that if the system designer has the control over how to assign tasks to the workers, our result can be applied to sparse sampling regime via a grouping technique. In particular, one can partition the items into groups of small size and assign different workers to each groups of items so that the each subgroups of items and workers form a dense sub-matrix.

a factor of $1/\alpha$. Note that α is a constant that doesn't depend on σ_L. Thus, this potential improvement doesn't affect the lower bound's dependence on the condition number of the confusion matrix.

The result of Balakrishnan et al. (2016) provides conditions for EM to work but it doesn't show how to initialize EM to satisfy these conditions. Our paper uses the spectral method to do the initialization. As a consequence, the restriction on n_{spectral} is a premise for the spectral method to work. In order to improve the dependence on the condition number, one has to invent a better initialization scheme, which is out of the scope of this paper.

5.1.3 Discussion of simple majority voting estimator

It is also interesting to compare our algorithm with the majority voting estimator, where the true label is simply estimated by a majority vote among workers. Gao and Zhou (2014) show that if there are many spammers and few experts, the majority voting estimator gives almost a random guess. In contrast, our algorithm requires a sufficiently large $m\overline{D}$ to guarantee good performance. Since $m\overline{D}$ is the aggregated KL-divergence, a small number of experts suffices to ensure it is large enough.

6. One-Coin Model

In this section, we consider a simpler crowdsourcing model that is usually referred to as the "one-coin model." For the one-coin model, the confusion matrix C_i is parameterized by a single parameter p_i. More concretely, its entries are defined as

$$\mu_{ilc} = \begin{cases} p_i & \text{if } l = c, \\ \frac{1-p_i}{k-1} & \text{if } l \neq c. \end{cases} \tag{12}$$

In other words, the worker i uses a single coin flip to decide her assignment. No matter what the true label is, the worker has p_i probability to assign the correct label, and has $1 - p_i$ probability to randomly assign an incorrect label. For the one-coin model, it suffices to estimate p_i for every worker i and estimate y_j for every item j. Because of its simplicity, the one-coin model is easier to estimate and enjoys better convergence properties.

To simplify our presentation, we consider the case where $\pi_i \equiv 1$; noting that with proper normalization, the algorithm can be easily adapted to the case where $\pi_i < 1$. The statement of the algorithm relies on the following notation: For every two workers a and b, let the quantity N_{ab} be defined as

$$N_{ab} := \frac{k-1}{k}\left(\frac{\sum_{j=1}^{n}\mathbb{I}(z_{aj} = z_{bj})}{n} - \frac{1}{k}\right).$$

For every worker i, let workers a_i, b_i be defined as

$$(a_i, b_i) = \arg\max_{(a,b)}\{|N_{ab}| : \ a \neq b \neq i\}.$$

The algorithm contains two separate stages. First, we initialize $\widehat{p}_i$ by an estimator based on the method of moments. In contrast with the algorithm for the general model, the estimator

Algorithm 2: Estimating one-coin model

Input: integer k, observed labels $z_{ij} \in \mathbb{R}^k$ for $i \in [m]$ and $j \in [n]$.
Output: Estimator $\widehat{p}_i$ for $i \in [m]$ and $\widehat{y}_j$ for $j \in [n]$.

(1) Initialize $\widehat{p}_i$ by

$$\widehat{p}_i \leftarrow \frac{1}{k} + \text{sign}(N_{ia_1})\sqrt{\frac{N_{ia_i}N_{ib_i}}{N_{a_ib_i}}} \tag{13}$$

(2) If $\frac{1}{m}\sum_{i=1}^{m}\widehat{p}_i \geq \frac{1}{k}$ does not hold, then set $\widehat{p}_i \leftarrow \frac{2}{k} - \widehat{p}_i$ for all $i \in [m]$.

(3) Iteratively execute the following two steps for at least one round:

$$\widehat{q}_{jl} \propto \exp\Big(\sum_{i=1}^{m}\mathbb{I}(z_{ij} = e_l)\log(\widehat{p}_i) + \mathbb{I}(z_{ij} \neq e_l)\log\Big(\frac{1-\widehat{p}_i}{k-1}\Big)\Big) \qquad \text{for } j \in [n],\ l \in [k], \tag{14}$$

$$\widehat{p}_i \leftarrow \frac{1}{n}\sum_{j=1}^{n}\sum_{l=1}^{k}\widehat{q}_{jl}\mathbb{I}(z_{ij} = e_l) \qquad \text{for } i \in [m], \tag{15}$$

where update (14) normalizes $\widehat{q}_{jl}$, making $\sum_{l=1}^{k}\widehat{q}_{jl} = 1$ hold for all $j \in [n]$.

(4) Output $\{\widehat{p}_i\}$ and $\widehat{y}_j := \arg\max_{l\in[k]}\{\widehat{q}_{jl}\}$.

for the one-coin model doesn't need third-order moments. Instead, it only relies on pairwise statistics N_{ab}. Second, an EM algorithm is employed to iteratively maximize the objective function (6). See Algorithm 2 for a detailed description.

To theoretically characterize the performance of Algorithm 2, we need some additional notation. Let κ_i be the i-th largest element in $\{|p_i - 1/k|\}_{i=1}^{m}$. In addition, let $\overline{\kappa} := \frac{1}{m}\sum_{i=1}^{m}(p_i - 1/k)$ be the average gap between all accuracies and $1/k$. We assume that $\overline{\kappa}$ is strictly positive. We follow the definition of $\overline{D}$ in Eq. (9). The following theorem is proved in Appendix D.

Theorem 6 *Assume that $\rho \leq p_i \leq 1 - \rho$ holds for all $i \in [m]$. For any scalar $\delta > 0$, if the number of workers m and the number of items n satisfy*

$$m = \Omega\left(\frac{\log(1/\rho)\log(kn/\delta)}{\overline{D}}\right) \quad \text{and} \quad n = \Omega\left(\frac{\log(mk/\delta)}{\kappa_3^6\min\{\overline{\kappa}^2, \rho^2, (\rho\overline{D})^2\}}\right), \tag{16}$$

then, for $\widehat{p}$ and $\widehat{y}$ returned by Algorithm 2, with probability at least $1 - \delta$,

(a) $\widehat{y}_j = y_j$ holds for all $j \in [n]$;

(b) $|\widehat{p}_i - p_i| \leq 2\sqrt{\frac{3\log(6m/\delta)}{n}}$ holds for all $i \in [m]$.

	Opt-D&S	MV-D&S	Majority Voting	KOS	Ghosh-SVD	EigenRatio
$\pi = 0.2$	**7.64**	7.65	18.85	8.34	12.35	10.49
$\pi = 0.5$	**0.84**	**0.84**	7.97	1.04	4.52	4.52
$\pi = 1.0$	**0.01**	**0.01**	1.57	0.02	0.15	0.15

Table 1: Prediction error (%) on the synthetic dataset. The parameter π indicates the sparsity of data—it is the probability that the worker labels each task.

It is worth contrasting condition (11) with condition (16), namely the sufficient conditions for the general model and for the one-coin model. It turns out that the one-coin model requires much milder conditions on the number of items. In particular, κ_3 will be close to 1 if among all the workers there are three experts giving high-quality answers. As a consequence, the one-coin model is more robust than the general model. By contrasting the convergence rate of $\widehat{\mu}_{il}$ (by Theorem 4) and $\widehat{p}_i$ (by Theorem 6), the convergence rate of $\widehat{p}_i$ does not depend on $\{w_l\}_{l=1}^k$. This is additional evidence that the one-coin model enjoys a better convergence rate because of its simplicity.

7. Experiments

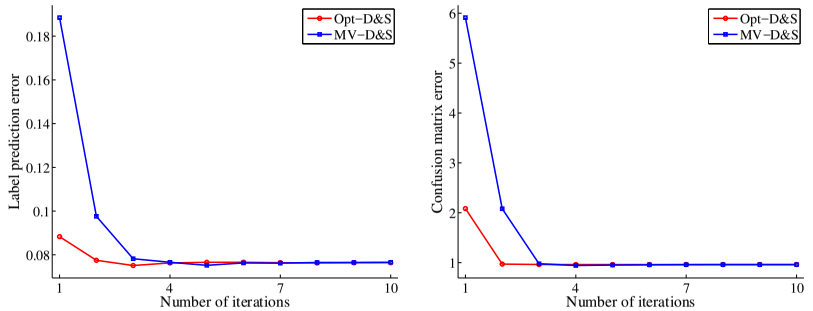
In this section, we report the results of empirical studies comparing the algorithm we propose in Section 4 (referred to as Opt-D&S) with a variety of other methods. We compare to the Dawid & Skene estimator initialized by majority voting (refereed to as MV-D&S), the pure majority voting estimator, the multi-class labeling algorithm proposed by Karger et al. (2013) (referred to as KOS), the SVD-based algorithm proposed by Ghosh et al. (2011) (referred to as Ghost-SVD) and the "Eigenvalues of Ratio" algorithm proposed by Dalvi et al. (2013) (referred to as EigenRatio). The evaluation is made on three synthetic datasets and five real datasets.

7.1 Synthetic data

For synthetic data, we generate $m = 100$ workers and $n = 1000$ binary tasks. The true label of each task is uniformly sampled from $\{1, 2\}$. For each worker, the 2-by-2 confusion matrix is generated as follow: the two diagonal entries are independently and uniformly sampled from the interval $[0.3, 0.9]$, then the non-diagonal entries are determined to make the confusion matrix columns sum to 1. To simulate a sparse dataset, we make each worker label a task with probability π. With the choice $\pi \in \{0.2, 0.5, 1.0\}$, we obtain three different datasets.

We execute every algorithm independently ten times and average the outcomes. For the Opt-D&S algorithm and the MV-D&S estimator, the estimation is outputted after ten EM iterates. For the group partitioning step involved in the Opt-D&S algorithm, the workers are randomly and evenly partitioned into three groups.

The main evaluation metric is the error of predicting the true label of items. The performance of various methods are reported in Table 1. On all sparsity levels, the Opt-D&S algorithm achieves the best accuracy, followed by the MV-D&S estimator. All other



(a) (b)

Figure 1: Comparing the convergence rate of the Opt-D&S algorithm and the MV-D&S estimator on synthetic dataset with $\pi = 0.2$: (a) convergence of the prediction error. (b) convergence of the squared error $\sum_{i=1}^m \|\widehat{C}_i - C_i\|_F^2$ for estimating confusion matrices.

methods are consistently worse. It is not surprising that the Opt-D&S algorithm and the MV-D&S estimator yield similar accuracies, since they optimize the same log-likelihood objective. It is also meaningful to look at the convergence speed of both methods, as they employ distinct initialization strategies. Figure 1 shows that the Opt-D&S algorithm converges faster than the MV-D&S estimator, both in estimating the true labels and in estimating confusion matrices. This can be explained by the general theoretical guarantee associated with Opt-D&S (recall Theorem 3).

The first and the second iteration of the Opt-D&S curve in Figure 1(b) correspond to the confusion matrix estimation error achieved by 1) the spectral method and 2) the spectral method with one iteration of EM, respectively. In Table 2, we provide a more detailed comparison, where the errors are compared with $\pi \in \{0.2, 0.5, 1.0\}$ and at different stages of the iteration. We found that for dense data ($\pi = 1$), the spectral method alone provides a reasonably good estimate on the confusion matrix, but its performance degenerates as the data becomes sparser. For both dense and sparse data, the spectral method with one iteration of EM achieves the nearly optimal error rate, which coincides with our theoretical prediction. In contrast, if the algorithm is initialized by majority voting, then one iteration of EM fails to provide a good estimate. Table 2 shows that its error rate is an order-of-magnitude higher than the spectral method initialized EM. This supports our concern that majority-voting initialization can be far from optimal—a principal motivation for this paper. Nevertheless, both initializations converge to the same error rate after ten iterations. Given the robustness of the EM method, deriving a sufficient and necessary condition under which the majority-voting initialization converges to an optimal solution remains an open problem.

	$\pi = 0.2$	$\pi = 0.5$	$\pi = 1.0$
Spectral Method	2.084	0.583	0.164
Opt-D&S (1st iteration)	0.972	0.352	0.143
Opt-D&S (10th iteration)	0.962	0.343	0.143
MV-D&S (1st iteration)	5.912	6.191	6.618
MV-D&S (10st iteration)	0.962	0.343	0.143

Table 2: Squared error for estimating the confusion matrix. The table compares (1) spectral method; (2) spectral initialization + one iteration of EM; (3) spectral initialization + 10 iterations of EM; (4) majority voting initialization + one iteration of EM; and (5) majority voting initialization + 10 iterations of EM.

Dataset name	# classes	# items	# workers	# worker labels
Bird	2	108	39	4,212
RTE	2	800	164	8,000
TREC	2	19,033	762	88,385
Dog	4	807	52	7,354
Web	5	2,665	177	15,567

Table 3: The summary of datasets used in the real data experiment.

7.2 Real data

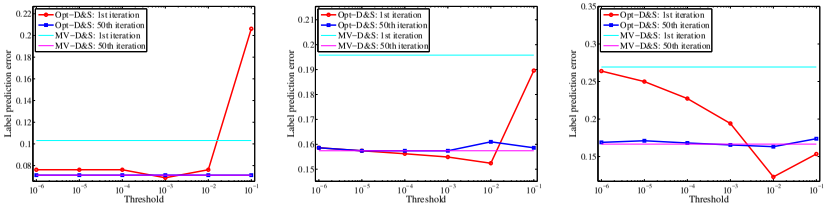
For real data experiments, we compare crowdsourcing algorithms on five datasets: three binary tasks and two multi-class tasks. Binary tasks include labeling bird species (Welinder et al., 2010) (Bird dataset), recognizing textual entailment (Snow et al., 2008) (RTE dataset) and assessing the quality of documents in TREC 2011 crowdsourcing track (Lease and Kazai, 2011) (TREC dataset). Multi-class tasks include labeling the bread of dogs from ImageNet (Deng et al., 2009) (Dog dataset) and judging the relevance of web search results (Zhou et al., 2012) (Web dataset). The statistics for the five datasets are summarized in Table 3. Since the Ghost-SVD algorithm and the EigenRatio algorithm work on binary tasks, they are evaluated on the Bird, RTE and TREC dataset. For the MV-D&S estimator and the Opt-D&S algorithm, we iterate their EM steps until convergence.

Since entries of the confusion matrix are positive, we find it helpful to incorporate this prior knowledge into the initialization stage of the Opt-D&S algorithm. In particular, when estimating the confusion matrix entries by equation (5), we add an extra checking step before the normalization, examining if the matrix components are greater than or equal to a small threshold Δ. For components that are smaller than Δ, they are reset to Δ. The default choice of the thresholding parameter is $\Delta = 10^{-6}$. Later, we will compare the Opt-D&S algorithm with respect to different choices of Δ. It is important to note that this modification doesn't change our theoretical result, since the thresholding step doesn't take effect if the initialization error is bounded by Theorem 3.

Table 4 summarizes the performance of each method. The MV-D&S estimator and the Opt-D&S algorithm consistently outperform the other methods in predicting the true label of items. The KOS algorithm, the Ghost-SVD algorithm and the EigenRatio algorithm yield

	Opt-D&S	MV-D&S	Majority Voting	KOS	Ghosh-SVD	EigenRatio
Bird	**10.09**	11.11	24.07	11.11	27.78	27.78
RTE	**7.12**	**7.12**	10.31	39.75	49.13	9.00
TREC	**29.80**	30.02	34.86	51.96	42.99	43.96
Dog	16.89	**16.66**	19.58	31.72	–	–
Web	15.86	**15.74**	26.93	42.93	–	–

Table 4: Error rate (%) in predicting the true labels on real data.



(a) RTE (b) Dog (c) Web

Figure 2: Comparing the MV-D&S estimator the Opt-D&S algorithm with different thresholding parameter Δ. The predict error is plotted after the 1st EM update and after convergence.

poorer performance, presumably due to the fact that they rely on idealized assumptions that are not met by the real data. In Figure 2, we compare the Opt-D&S algorithm with respect to different thresholding parameters $\Delta \in \{10^{-i}\}_{i=1}^{6}$. We plot results for three datasets (RET, Dog, Web), where the performance of the MV-D&S estimator is equal to or slightly better than that of Opt-D&S. The plot shows that the performance of the Opt-D&S algorithm is stable after convergence. But when only using the spectral method with just one E-step for the label prediction (i.e., the red curve Opt-D&S: 1st iteration in Figure 2), the error rates are more sensitive to the choice of Δ. A proper choice of Δ makes the Opt-D&S algorithm perform better than MV-D&S. The result suggests that a proper spectral initialization with just an E-step is good enough for the purposes of prediction. In practice, the best choice of Δ can be obtained by cross validation.

8. Conclusions

Under the generative model proposed by Dawid and Skene (1979), we propose an optimal algorithm for inferring true labels in the multi-class crowd labeling setting. Our approach utilizes the method of moments to construct an initial estimator for the EM algorithm. We proved that our method achieves the optimal rate with only one iteration of the EM algorithm.

To the best of our knowledge, this work provides the first instance of provable convergence for a latent variable model in which EM is initialized with the method of moments.

One-step EM initialized by the method of moments not only leads to better estimation error in terms of the dependence on the condition number of the second-order moment matrix but it also computationally more attractive than the standard one-step estimator obtained via a Newton-Raphson step. It is interesting to explore whether a properly initialized one-step EM algorithm can achieve the optimal rate for other latent variable models such as latent Dirichlet allocation or other mixed membership models.

Acknowledgments

We would like to thank anonymous reviewers and the associate editor for their constructive comments on improving the quality of the paper. Xi Chen would like to thank the support from Google Faculty Research Awards. This research was supported in part by DHS Award HSHQDC-16-3-00083, NSF CISE Expeditions Award CCF-1139158, DOE Award SN10040 DE-SC0012463, and DARPA XData Award FA8750-12-2-0331, and gifts from Amazon Web Services, Google, IBM, SAP, The Thomas and Stacey Siebel Foundation, Apple Inc., Arimo, Blue Goji, Bosch, Cisco, Cray, Cloudera, Ericsson, Facebook, Fujitsu, HP, Huawei, Intel, Microsoft, Pivotal, Samsung, Schlumberger, Splunk, State Farm and VMware.

Appendix A. Proof of Theorem 3

If $a \neq b$, it is easy to verify that $S_{ab} = C_a^\diamond W (C_b^\diamond)^T = \mathbb{E}[Z_{aj} \otimes Z_{bj}]$. Furthermore, we can upper bound the spectral norm of S_{ab}, namely

$$\|S_{ab}\|_{\text{op}} \leq \sum_{l=1}^{k} w_l \, \|\mu_{al}^\diamond\|_2 \, \|\mu_{bl}^\diamond\|_2 \leq \sum_{l=1}^{k} w_l \, \|\mu_{al}^\diamond\|_1 \, \|\mu_{bl}^\diamond\|_1 \leq 1.$$

For the same reason, it can be shown that $\|T_{abc}\|_{\text{op}} \leq 1$.

Our proof strategy is briefly described as follows: we upper bound the estimation error for computing empirical moments (2a)-(2d) in Lemma 7, and upper bound the estimation error for tensor decomposition in Lemma 8. Then, we combine both lemmas to upper bound the error of formula (5).

Lemma 7 *Given a permutation (a, b, c) of $(1, 2, 3)$, for any scalar $\epsilon \leq \sigma_L/2$, the second and the third moments $\widehat{M}_2$ and $\widehat{M}_3$ computed by equation (2c) and (2d) are bounded as*

$$\max\{\|\widehat{M}_2 - M_2\|_{\text{op}}, \|\widehat{M}_3 - M_3\|_{\text{op}}\} \leq 31\epsilon/\sigma_L^3 \tag{17}$$

with probability at least $1 - \delta$, where $\delta = 6\exp(-(\sqrt{n}\epsilon - 1)^2) + k\exp(-(\sqrt{n/k}\epsilon - 1)^2)$.

Lemma 8 *Suppose that (a, b, c) is permutation of $(1, 2, 3)$. For any scalar $\epsilon \leq \kappa/2$, if the empirical moments $\widehat{M}_2$ and $\widehat{M}_3$ satisfy*

$$\max\{\|\widehat{M}_2 - M_2\|_{\text{op}}, \|\widehat{M}_3 - M_3\|_{\text{op}}\} \leq \epsilon H \tag{18}$$

$$\text{for} \quad H := \min\left\{\frac{1}{2}, \frac{2\sigma_L^{3/2}}{15k(24\sigma_L^{-1} + 2\sqrt{2})}, \frac{\sigma_L^{3/2}}{4\sqrt{3/2}\sigma_L^{1/2} + 8k(24/\sigma_L + 2\sqrt{2})}\right\}$$

then the estimates $\widehat{C}_c^\diamond$ and $\widehat{W}$ are bounded as

$$\|\widehat{C}_c^\diamond - C_c^\diamond\|_{\text{op}} \leq \sqrt{k}\epsilon \qquad \text{and} \qquad \|\widehat{W} - W\|_{\text{op}} \leq \epsilon.$$

with probability at least $1 - \delta$, where δ is defined in Lemma 7.

Combining Lemma 7, Lemma 8, if we choose a scalar ϵ_1 satisfying

$$\epsilon_1 \leq \min\{\kappa/2, \pi_{\min} w_{\min} \sigma_L/(36k)\}, \tag{19}$$

then the estimates $\widehat{C}_g^\diamond$ (for $g = 1, 2, 3$) and $\widehat{W}$ satisfy

$$\|\widehat{C}_g^\diamond - C_g^\diamond\|_{\text{op}} \leq \sqrt{k}\epsilon_1 \qquad \text{and} \qquad \|\widehat{W} - W\|_{\text{op}} \leq \epsilon_1. \tag{20}$$

with probability at least $1 - 6\delta$, where

$$\delta = (6 + k)\exp\Big(-(\sqrt{n/k}\epsilon_1 H \sigma_L^3/31 - 1)^2\Big).$$

To be more precise, we obtain the bound (20) by plugging $\epsilon := \epsilon_1 H\sigma_L^3/31$ into Lemma 7, then plugging $\epsilon := \epsilon_1$ into Lemma 8. The high probability statement is obtained by applying a union bound.

Assuming inequality (20), for any $a \in \{1,2,3\}$, since $\|C_a^\diamond\|_{\text{op}} \le \sqrt{k}$, $\|\widehat{C}_a^\diamond - C_a^\diamond\|_{\text{op}} \le \sqrt{k}\epsilon_1$ and $\|W\|_{\text{op}} \le 1$, $\left\|\widehat{W} - W\right\|_{\text{op}} \le \epsilon_1$, Lemma 18 (the preconditions are satisfied by inequality (19)) implies that

$$\left\|\widehat{W}\widehat{C}_a^\diamond - WC_a^\diamond\right\|_{\text{op}} \le 4\sqrt{k}\epsilon_1,$$

Since condition (19) implies

$$\|\widehat{W}\widehat{C}_a^\diamond - WC_a^\diamond\|_{\text{op}} \le 4\sqrt{k}\epsilon_1 \le \sqrt{w_{\min}\sigma_L}/2 \le \sigma_k(WC_a^\diamond)/2$$

Lemma 17 yields that

$$\left\|\left(\widehat{W}\widehat{C}_a^\diamond\right)^{-1} - (WC_a^\diamond)^{-1}\right\|_{\text{op}} \le \frac{8\sqrt{k}\epsilon_1}{w_{\min}\sigma_L}.$$

By Lemma 19, for any $i \in [m]$, the concentration bound

$$\left\|\frac{1}{n}\sum_{j=1}^n z_{ij}Z_{aj}^T - \mathbb{E}[z_{ij}Z_{aj}^T]\right\|_{\text{op}} \le \epsilon_1$$

holds with probability at least $1 - m\exp(-(\sqrt{n}\epsilon_1 - 1)^2)$. Combining the above two inequalities with Proposition 2, then applying Lemma 18 with preconditions

$$\left\|(WC_a^\diamond)^{-1}\right\|_{\text{op}} \le \frac{1}{w_{\min}\sigma_L} \quad \text{and} \quad \left\|\mathbb{E}\left[z_{ij}Z_{aj}^T\right]\right\|_{\text{op}} \le 1,$$

we have

$$\Bigg\| \underbrace{\left(\frac{1}{n}\sum_{j=1}^n z_{ij}Z_{aj}^T\right)\left(\widehat{W}\widehat{C}_a^\diamond\right)^{-1}}_{\widehat{G}} - \pi_i C_i \Bigg\|_{\text{op}} \le \frac{18\sqrt{k}\epsilon_1}{w_{\min}\sigma_L}. \tag{21}$$

Let $\widehat{G} \in \mathbb{R}^{k\times k}$ be the first term on the left hand side of inequality (21). Each column of $\widehat{G}$, denoted by $\widehat{G}_l$, is an estimate of $\pi_i\mu_{il}$. The ℓ_2-norm estimation error is bounded by $\frac{18\sqrt{k}\epsilon_1}{w_{\min}\sigma_L}$. Hence, we have

$$\|\widehat{G}_l - \pi_i\mu_{il}\|_1 \le \sqrt{k}\|\widehat{G}_l - \pi_i\mu_{il}\|_2 \le \sqrt{k}\|\widehat{G} - \pi_i C_i\|_{\text{op}} \le \frac{18k\epsilon_1}{w_{\min}\sigma_L}, \tag{22}$$

and consequently, using the fact that $\sum_{c=1}^k \mu_{ilc} = 1$, we have

$$\begin{aligned}\left\|\text{normalize}(\widehat{G}_l) - \mu_{il}\right\|_2 &= \left\|\frac{\widehat{G}_l}{\pi_i + \sum_{c=1}^k\left(\widehat{G}_{lc} - \pi_i\mu_{ilc}\right)} - \mu_{il}\right\|_2 \\ &\le \frac{\|\widehat{G}_l - \pi_i\mu_{il}\|_2 + \|\widehat{G}_l - \pi_i\mu_{il}\|_1\|\mu_{il}\|_2}{\pi_i - \|\widehat{G}_l - \pi_i\mu_{il}\|_1} \\ &\le \frac{72k\epsilon_1}{\pi_{\min}w_{\min}\sigma_L}\end{aligned} \tag{23}$$

where the last step combines inequalities (21), (22) with the bound $\frac{18k\epsilon_1}{w_{\min}\sigma_L} \le \pi_i/2$ from condition (19), and uses the fact that $\|\mu_{il}\|_2 \le 1$.

Note that inequality (23) holds with probability at least

$$1 - (36+6k)\exp\Big(-(\sqrt{n/k}\epsilon_1 H\sigma_L^3/31 - 1)^2\Big) - m\exp(-(\sqrt{n}\epsilon_1 - 1)^2).$$

It can be verified that $H \ge \frac{\sigma_L^{5/2}}{230k}$. Thus, the above expression is lower bounded by

$$1 - (36+6k+m)\exp\Big(-\Big(\frac{\sqrt{n}\epsilon_1\sigma_L^{11/2}}{31\times 230\cdot k^{3/2}} - 1\Big)^2\Big),$$

If we represent this probability in the form of $1-\delta$, then

$$\epsilon_1 = \frac{31\times 230\cdot k^{3/2}}{\sqrt{n}\sigma_L^{11/2}}\left(1 + \sqrt{\log((36+6k+m)/\delta)}\right). \tag{24}$$

Combining condition (19) and inequality (23), we find that to make $\|\widehat{C} - C\|_\infty$ bounded by ϵ, it is sufficient to choose ϵ_1 such that

$$\epsilon_1 \le \min\left\{\frac{\epsilon\pi_{\min}w_{\min}\sigma_L}{72k}, \frac{\kappa}{2}, \frac{\pi_{\min}w_{\min}\sigma_L}{36k}\right\}$$

This condition can be further simplified to

$$\epsilon_1 \le \frac{\epsilon\pi_{\min}w_{\min}\sigma_L}{72k} \tag{25}$$

for small ϵ, that is $\epsilon \le \min\left\{\frac{36\kappa k}{\pi_{\min}w_{\min}\sigma_L}, 2\right\}$. According to equation (24), the condition (25) will be satisfied if

$$\sqrt{n} \ge \frac{72\times 31\times 230\cdot k^{5/2}}{\epsilon\pi_{\min}w_{\min}\sigma_L^{13/2}}\left(1 + \sqrt{\log((36+6k+m)/\delta)}\right).$$

Squaring both sides of the inequality completes the proof.

A.1 Proof of Lemma 7

Throughout the proof, we assume that the following concentration bound holds: for any distinct indices $(a', b') \in \{1, 2, 3\}$, we have

$$\left\| \frac{1}{n} \sum_{j=1}^{n} Z_{a'j} \otimes Z_{b'j} - \mathbb{E}[Z_{a'j} \otimes Z_{b'j}] \right\|_{\text{op}} \leq \epsilon. \tag{26}$$

By Lemma 19 and the union bound, this event happens with probability at least $1 - 6\exp(-(\sqrt{n}\epsilon - 1)^2)$. By the assumption that $\epsilon \leq \sigma_L/2 \leq \sigma_k(S_{ab})/2$ and Lemma 17, we have

$$\left\| \frac{1}{n} \sum_{j=1}^{n} Z_{cj} \otimes Z_{bj} - \mathbb{E}[Z_{cj} \otimes Z_{bj}] \right\|_{\text{op}} \leq \epsilon \quad \text{and}$$

$$\left\| \left(\frac{1}{n} \sum_{j=1}^{n} Z_{aj} \otimes Z_{bj} \right)^{-1} - (\mathbb{E}[Z_{aj} \otimes Z_{bj}])^{-1} \right\|_{\text{op}} \leq \frac{2\epsilon}{\sigma_k^2(S_{ab})}$$

Under the preconditions

$$\|\mathbb{E}[Z_{cj} \otimes Z_{bj}]\|_{\text{op}} \leq 1 \quad \text{and} \quad \left\|(\mathbb{E}[Z_{aj} \otimes Z_{bj}])^{-1}\right\|_{\text{op}} \leq \frac{1}{\sigma_k(S_{ab})},$$

Lemma 18 implies that

$$\begin{aligned} &\left\| \left(\frac{1}{n} \sum_{j=1}^{n} Z_{cj} \otimes Z_{bj} \right) \left(\frac{1}{n} \sum_{j=1}^{n} Z_{aj} \otimes Z_{bj} \right)^{-1} - \mathbb{E}[Z_{cj} \otimes Z_{bj}](\mathbb{E}[Z_{aj} \otimes Z_{bj}])^{-1} \right\|_{\text{op}} \\ &\quad \leq 2 \left(\frac{\epsilon}{\sigma_k(S_{ab})} + \frac{2\epsilon}{\sigma_k^2(S_{ab})} \right) \leq 6\epsilon/\sigma_L^2 \end{aligned} \tag{27}$$

and for the same reason, we have

$$\left\| \left(\frac{1}{n} \sum_{j=1}^{n} Z_{cj} \otimes Z_{aj} \right) \left(\frac{1}{n} \sum_{j=1}^{n} Z_{bj} \otimes Z_{aj} \right)^{-1} - \mathbb{E}[Z_{cj} \otimes Z_{aj}](\mathbb{E}[Z_{bj} \otimes Z_{aj}])^{-1} \right\|_{\text{op}} \leq 6\epsilon/\sigma_L^2 \tag{28}$$

Now, let matrices F_2 and F_3 be defined as

$$F_2 := \mathbb{E}[Z_{cj} \otimes Z_{bj}](\mathbb{E}[Z_{aj} \otimes Z_{bj}])^{-1},$$
$$F_3 := \mathbb{E}[Z_{cj} \otimes Z_{aj}](\mathbb{E}[Z_{bj} \otimes Z_{aj}])^{-1},$$

and let the matrix on the left hand side of inequalities (27) and (28) be denoted by Δ_2 and Δ_3, we have

$$\begin{aligned} \left\| \widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right\|_{\text{op}} &= \left\| \left(F_2 + \Delta_2 \right) \left(Z_{aj} \otimes Z_{bj} \right) \left(F_3 + \Delta_3 \right)^T - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right\|_{\text{op}} \\ &\leq \|Z_{aj} \otimes Z_{bj}\|_{\text{op}} \left(\|\Delta_2\|_{\text{op}} \|F_3 + \Delta_2\|_{\text{op}} + \|F_2\|_{\text{op}} \|\Delta_3\|_{\text{op}} \right) \leq 30\epsilon \|Z_{aj} \otimes Z_{bj}\|_{\text{op}} / \sigma_L^3. \end{aligned}$$

where the last steps uses inequality (27), (28) and the fact that $\max\{\|F_2\|_{\text{op}}, \|F_3\|_{\text{op}}\} \leq 1/\sigma_L$ and

$$\|F_3 + \Delta_2\|_{\text{op}} \leq \|F_3\|_{\text{op}} + \|\Delta_2\|_{\text{op}} \leq 1/\sigma_L + 6\epsilon/\sigma_L^2 \leq 4/\sigma_L.$$

To upper bound the norm $\|Z_{aj} \otimes Z_{bj}\|_{\text{op}}$, notice that

$$\|Z_{aj} \otimes Z_{bj}\|_{\text{op}} \leq \|Z_{aj}\|_2 \|Z_{bj}\|_2 \leq \|Z_{aj}\|_1 \|Z_{bj}\|_1 \leq 1.$$

Consequently, we have

$$\left\| \widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right\|_{\text{op}} \leq 30\epsilon/\sigma_L^3. \tag{29}$$

For the rest of the proof, we use inequality (29) to bound $\widehat{M}_2$ and $\widehat{M}_3$. For the second moment, we have

$$\begin{aligned} \left\| \widehat{M}_2 - M_2 \right\|_{\text{op}} &\leq \frac{1}{n} \sum_{j=1}^{n} \left\| \widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right\|_{\text{op}} + \left\| F_2 \left(\frac{1}{n} \sum_{j=1}^{n} Z_{aj} \otimes Z_{bj} \right) F_3^T - M_2 \right\|_{\text{op}} \\ &\leq 30\epsilon/\sigma_L^3 + \left\| F_2 \left(\frac{1}{n} \sum_{j=1}^{n} Z_{aj} \otimes Z_{bj} - \mathbb{E}[Z_{aj} \otimes Z_{bj}] \right) F_3^T \right\|_{\text{op}} \\ &\leq 30\epsilon/\sigma_L^3 + \epsilon/\sigma_L^2 \leq 31\epsilon/\sigma_L^3. \end{aligned}$$

For the third moment, we have

$$\begin{aligned} \widehat{M}_3 - M_3 = &\frac{1}{n} \sum_{j=1}^{n} \left(\widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right) \otimes Z_{cj} \\ &+ \left(\frac{1}{n} \sum_{j=1}^{n} F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \otimes Z_{cj} - \mathbb{E}\left[F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \otimes Z_{cj} \right] \right). \end{aligned} \tag{30}$$

We examine the right hand side of equation (30). The first term is bounded as

$$\begin{aligned} \left\| \left(\widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right) \otimes Z_{cj} \right\|_{\text{op}} &\leq \left\| \widehat{Z}'_{aj} \otimes \widehat{Z}'_{bj} - F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \right\|_{\text{op}} \|Z_{cj}\|_2 \\ &\leq 30\epsilon/\sigma_L^3. \end{aligned} \tag{31}$$

For the second term, since $\|F_2 Z_{aj}\|_2 \leq 1/\sigma_L$, $\|F_3 Z_{bj}\|_2 \leq 1/\sigma_L$ and $\|Z_{cj}\|_2 \leq 1$, Lemma 19 implies that

$$\left\| \frac{1}{n} \sum_{j=1}^{n} F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \otimes Z_{cj} - \mathbb{E}\left[F_2 \left(Z_{aj} \otimes Z_{bj} \right) F_3^T \otimes Z_{cj} \right] \right\|_{\text{op}} \leq \epsilon/\sigma_L^2 \tag{32}$$

with probability at least $1 - k\exp(-(\sqrt{n/k}\epsilon - 1)^2)$. Combining inequalities (31) and (32), we have

$$\left\| \widehat{M}_3 - M_3 \right\|_{\text{op}} \leq 30\epsilon/\sigma_L^3 + \epsilon/\sigma_L^2 \leq 31\epsilon/\sigma_L^3.$$

Applying a union bound to all high-probability events completes the proof.

A.2 Proof of Lemma 8

Chaganty and Liang (2013) (Lemma 4) prove that when condition (18) holds, the tensor decomposition method of Algorithm 1 outputs $\{\widehat{\mu}_h^\diamond, \widehat{w}_h\}_{h=1}^k$, such that with probability at least $1-\delta$, a permutation π satisfies

$$\|\widehat{\mu}_h^\diamond - \mu_{c\pi(h)}^\diamond\|_2 \le \epsilon \qquad \text{and} \qquad \left\|\widehat{w}_h - w_{\pi(h)}\right\|_\infty \le \epsilon.$$

Note that the constant H in Lemma 8 is obtained by plugging upper bounds $\|M_2\|_{\text{op}} \le 1$ and $\|M_3\|_{\text{op}} \le 1$ into Lemma 4 of Chaganty and Liang (2013).

The $\pi(h)$-th component of $\mu_{c\pi(h)}^\diamond$ is greater than other components of $\mu_{c\pi(h)}^\diamond$, by a margin of κ. Assuming $\epsilon \le \kappa/2$, the greatest component of $\widehat{\mu}_h^\diamond$ is its $\pi(h)$-th component. Thus, Algorithm 1 is able to correctly estimate the $\pi(h)$-th column of $\widehat{C}_c^\diamond$ by the vector $\widehat{\mu}_h^\diamond$. Consequently, for every column of $\widehat{C}_c^\diamond$, the ℓ_2-norm error is bounded by ϵ. Thus, the spectral-norm error of $\widehat{C}_c^\diamond$ is bounded by $\sqrt{k}\epsilon$. Since W is a diagonal matrix and $\left\|\widehat{w}_h - w_{\pi(h)}\right\|_\infty \le \epsilon$, we have $\|\widehat{W} - W\|_{\text{op}} \le \epsilon$.

Appendix B. Proof of Theorem 4

We define three random events that will be shown holding with high probability:

$$\mathcal{E}_1: \ \sum_{i=1}^m \sum_{c=1}^k \mathbb{I}(z_{ij}=e_c)\log(\mu_{iy_jc}/\mu_{ilc}) \ge m\overline{D}/2 \qquad \text{for all } j\in[n] \text{ and } l\in[k]\backslash\{y_j\}.$$

$$\mathcal{E}_2: \ \left|\sum_{j=1}^n \mathbb{I}(y_j=l)\mathbb{I}(z_{ij}=e_c) - nw_l\pi_i\mu_{ilc}\right| \le nt_{ilc} \qquad \text{for all } (i,l,c)\in[m]\times[k]^2.$$

$$\mathcal{E}_3: \ \left|\sum_{j=1}^n \mathbb{I}(y_j=l)\mathbb{I}(z_{ij}\ne 0) - nw_l\pi_i\right| \le \frac{nt_{ilc}}{\mu_{ilc}} \qquad \text{for all } (i,l,c)\in[m]\times[k]^2. \tag{33}$$

where $t_{ilc} > 0$ are scalars to be specified later. We define $t_{\min}$ to be the smallest element among $\{t_{ilc}\}$. Assuming that $\mathcal{E}_1 \cap \mathcal{E}_2$ holds, the following lemma shows that performing updates (7) and (8) attains the desired level of accuracy. See Section B.1 for the proof.

Lemma 9 *Assume that $\mathcal{E}_1\cap\mathcal{E}_2$ holds. Also assume that $\mu_{ilc} \ge \rho$ for all $(i,l,c)\in[m]\times[k]^2$. If $\widehat{C}$ is initialized such that inequality (10) holds, and scalars t_{ilc} satisfy*

$$2\exp\Big(-m\overline{D}/4 + \log(k)\Big) \le t_{ilc} \le \pi_{\min}w_{\min}\min\left\{\frac{\rho}{8}, \frac{\rho\overline{D}}{64}\right\}. \tag{34}$$

Then by alternating updates (7) and (8) for at least one round, the estimates $\widehat{C}$ and $\widehat{q}$ are bounded as

$$|\widehat{\mu}_{ilc} - \mu_{ilc}| \le 4t_{ilc}/(\pi_i w_l). \qquad \text{for all } i\in[m],\ l\in[k],\ c\in[k].$$

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j=l)|\} \le \exp\big(-m\overline{D}/4 + \log(k)\big) \qquad \text{for all } j\in[n].$$

Next, we characterize the probability that events $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$ hold. For measuring $\mathbb{P}[\mathcal{E}_1]$, we define auxiliary variable $s_i := \sum_{c=1}^k \mathbb{I}(z_{ij}=e_c)\log(\mu_{iy_jc}/\mu_{ilc})$. It is straightforward to see that $s_1, s_2, \ldots, s_m$ are mutually independent on any value of y_j, and each s_i belongs to the interval $[0, \log(1/\rho)]$. it is easy to verify that

$$\mathbb{E}\left[\sum_{i=1}^m s_i \Big| y_i\right] = \sum_{i=1}^m \pi_i \mathbb{D}_{\text{KL}}\left(\mu_{iy_j}, \mu_{il}\right).$$

We denote the right hand side of the above equation by D. The following lemma shows that the second moment of s_i is bounded by the KL-divergence between labels.

Lemma 10 *Conditioning on any value of y_j, we have*

$$\mathbb{E}[s_i^2|y_i] \le \frac{2\log(1/\rho)}{1-\rho}\pi_i\mathbb{D}_{\text{KL}}\left(\mu_{iy_j}, \mu_{il}\right).$$

According to Lemma 10, the aggregated second moment of s_i is bounded by

$$\mathbb{E}\left[\sum_{i=1}^m s_i^2 \Big| y_i\right] \le \frac{2\log(1/\rho)}{1-\rho}\sum_{i=1}^m \pi_i\mathbb{D}_{\text{KL}}\left(\mu_{iy_jc}, \mu_{ilc}\right) = \frac{2\log(1/\rho)}{1-\rho}D$$

Thus, applying the Bernstein inequality, we have

$$\mathbb{P}\Big[\sum_{i=1} s_i \ge D/2 | y_i\Big] \ge 1 - \exp\left(-\frac{\frac{1}{2}(D/2)^2}{\frac{2\log(1/\rho)}{1-\rho}D + \frac{1}{3}(2\log(1/\rho))(D/2)}\right),$$

Since $\rho \le 1/2$ and $D \ge m\overline{D}$, combining the above inequality with the union bound, we have

$$\mathbb{P}[\mathcal{E}_1] \ge 1 - kn\exp\left(-\frac{m\overline{D}}{33\log(1/\rho)}\right). \tag{35}$$

For measuring $\mathbb{P}[\mathcal{E}_2]$, we observe that $\sum_{j=1}^n \mathbb{I}(y_j=l)\mathbb{I}(z_{ij}=e_c)$ is the sum of n i.i.d. Bernoulli random variables with mean $p := \pi_i w_l \mu_{ilc}$. Since $t_{ilc} \le \pi_{\min}w_{\min}\rho/8 \le p$, applying the Chernoff bound implies

$$\mathbb{P}\left[\left|\sum_{j=1}^n \mathbb{I}(y_j=l)\mathbb{I}(z_{ij}=e_c) - np\right| \ge nt_{ilc}\right] \le 2\exp(-nt_{ilc}^2/(3p)) = 2\exp\left(-\frac{nt_{ilc}^2}{3\pi_i w_l\mu_{ilc}}\right),$$

For measuring $\mathbb{P}[\mathcal{E}_3]$, note that $\sum_{j=1}^n \mathbb{I}(y_j=l)\mathbb{I}(z_{ij}\ne e_c)$ is the sum of n i.i.d. Bernoulli random variables with mean $q := \pi_i w_l$. Since $\frac{t_{ilc}}{\mu_{ilc}} \le \frac{\pi_{\min}w_{\min}\rho/8}{\rho} \le q$, using a Chernoff bound yields

$$\mathbb{P}\left[\left|\sum_{j=1}^n \mathbb{I}(y_j=l)\mathbb{I}(z_{ij}\ne 0) - nq\right| \ge n\frac{t_{ilc}}{\mu_{ilc}}\right] \le 2\exp(-\frac{nt_{ilc}^2}{3q\mu_{ilc}^2}) \le 2\exp\left(-\frac{nt_{ilc}^2}{3\pi_i w_l\mu_{ilc}}\right),$$

Summarizing the probability bounds on $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$, we conclude that $\mathcal{E}_1\cap\mathcal{E}_2\cap\mathcal{E}_3$ holds with probability at least

$$1 - kn\exp\left(-\frac{m\overline{D}}{33\log(1/\rho)}\right) - \sum_{i=1}^m\sum_{l=1}^k 4\exp\left(-\frac{nt_{ilc}^2}{3\pi_i w_l\mu_{ilc}}\right). \tag{36}$$

Proof of Part (a) According to Lemma 9, for $\widehat{y}_j = y_j$ being true, it sufficient to have $\exp(-m\overline{D}/4 + \log(k)) < 1/2$, or equivalently

$$m > 4\log(2k)/\overline{D}. \tag{37}$$

To ensure that this bound holds with probability at least $1-\delta$, expression (36) needs to be lower bounded by δ. It is achieved if we have

$$m \geq \frac{33\log(1/\rho)\log(2kn/\delta)}{\overline{D}} \quad \text{and} \quad n \geq \frac{3\pi_i w_l \mu_{ilc}\log(8mk/\delta)}{t_{ilc}^2} \tag{38}$$

If we choose

$$t_{ilc} := \sqrt{\frac{3\pi_i w_l \mu_{ilc}\log(8mk/\delta)}{n}}. \tag{39}$$

then the second part of condition (38) is guaranteed. To ensure that t_{ilc} satisfies condition (34). We need to have

$$\sqrt{\frac{3\pi_i w_l \mu_{ilc}\log(8mk/\delta)}{n}} \geq 2\exp\Big(-m\overline{D}/4 + \log(k)\Big) \quad \text{and}$$
$$\sqrt{\frac{3\pi_i w_l \mu_{ilc}\log(8mk/\delta)}{n}} \leq \pi_{\min} w_{\min}\alpha/4.$$

The above two conditions require that m and n satisfy

$$m \geq \frac{4\log(2k\sqrt{n/(3\pi_{\min}w_{\min}\rho\log(8mk/\delta))})}{\overline{D}} \tag{40}$$
$$n \geq \frac{48\log(8mk/\delta)}{\pi_{\min}w_{\min}\alpha^2} \tag{41}$$

The four conditions (37), (38), (40) and (41) are simultaneously satisfied if we have

$$m \geq \frac{33\log(1/\rho)\log(2kn/\delta)}{\overline{D}} \quad \text{and}$$
$$n \geq \frac{48\log(8mk/\delta)}{\pi_{\min}w_{\min}\alpha^2}.$$

Under this setup, $\widehat{y}_j = y_j$ holds for all $j \in [n]$ with probability at least $1-\delta$.

Proof of Part (b) If t_{ilc} is set by equation (39), combining Lemma 9 with this assignment, we have

$$(\widehat{\mu}_{ilc} - \mu_{ilc})^2 \leq \frac{48\mu_{ilc}\log(8mk/\delta)}{\pi_i w_l n}$$

with probability at least $1-\delta$. Summing both sides of the inequality over $c = 1, 2, \ldots, k$ completes the proof.

B.1 Proof of Lemma 9

To prove Lemma 9, we study the consequences of update (7) and update (8). We prove two important lemmas, which show that both updates provide good estimates if they are properly initialized.

Lemma 11 *Assume that event $\mathcal{E}_1$ holds. If μ and its estimate $\widehat{\mu}$ satisfy*

$$\mu_{ilc} \geq \rho \quad \text{and} \quad |\widehat{\mu}_{ilc} - \mu_{ilc}| \leq \delta_1 \qquad \text{for all } i \in [m],\ l \in [k],\ c \in [k], \tag{42}$$

and $\widehat{q}$ is updated by formula (7), then $\widehat{q}$ is bounded as:

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j = l)|\} \leq \exp\left(-m\left(\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1}\right) + \log(k)\right) \qquad \text{for all } j \in [n]. \tag{43}$$

Proof

For an arbitrary index $l \neq y_j$, we consider the quantity

$$A_l := \sum_{i=1}^{m}\sum_{c=1}^{k}\mathbb{I}(z_{ij} = e_c)\log(\widehat{\mu}_{iy_jc}/\widehat{\mu}_{ilc})$$

By the assumption that $\mathcal{E}_1$ and inequality (42) holds, we obtain that

$$A_l = \sum_{i=1}^{m}\sum_{c=1}^{k}\mathbb{I}(z_{ij} = e_c)\log(\mu_{iy_jc}/\mu_{ilc}) + \sum_{i=1}^{m}\sum_{c=1}^{k}\mathbb{I}(z_{ij} = e_c)\left[\log\left(\frac{\widehat{\mu}_{iy_jc}}{\mu_{iy_jc}}\right) - \log\left(\frac{\widehat{\mu}_{ilc}}{\mu_{ilc}}\right)\right]$$
$$\geq \left(\sum_{i=1}^{m}\frac{\pi_i\mathbb{D}_{\mathrm{KL}}\left(\mu_{iy_j}, \mu_{il}\right)}{2}\right) - 2m\log\left(\frac{\rho}{\rho - \delta_1}\right) \geq m\left(\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1}\right). \tag{44}$$

Thus, for every index $l \neq y_j$, combining formula (7) and inequality (44) implies that

$$\widehat{q}_{jl} \leq \frac{1}{\exp(A_l)} \leq \exp\left(-m\left(\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1}\right)\right).$$

Consequently, we have

$$\widehat{q}_{jy_j} \geq 1 - \sum_{l\neq y_j}\widehat{q}_{jl} \geq 1 - k\exp\left(-m\left(\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1}\right)\right).$$

Combining the above two inequalities completes the proof. ■

Lemma 12 *Assume that event $\mathcal{E}_2$ holds. If $\widehat{q}$ satisfies*

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j = l)|\} \leq \delta_2 \qquad \text{for all } j \in [n], \tag{45}$$

and $\widehat{\mu}$ is updated by formula (8), then $\widehat{\mu}$ is bounded as:

$$|\widehat{\mu}_{ilc} - \mu_{ilc}| \leq \frac{2nt_{ilc} + 2n\delta_2}{(7/8)n\pi_i w_l - n\delta_2}. \qquad \text{for all } i \in [m],\ l \in [k],\ c \in [k]. \tag{46}$$

Proof By formula (8), we can write $\widehat{\mu}_{ilc} = A/B$, where

$$A := \sum_{j=1}^{n} \widehat{q}_{jl}\mathbb{I}(z_{ij} = e_c) \quad \text{and} \quad B := \sum_{c'=1}^{k}\sum_{j=1}^{n} \widehat{q}_{jl}\mathbb{I}(z_{ij} = e_{c'}) = \sum_{j=1}^{n} \widehat{q}_{jl}\mathbb{I}(z_{ij} \neq 0).$$

Combining this definition with the assumption that event $\mathcal{E}_2$ and inequality (45) hold, we find that

$$\begin{aligned}|A - n\pi_i w_l \mu_{ilc}| &\leq \left|\sum_{j=1}^{n} \mathbb{I}(q_{jl} = y_j)\mathbb{I}(z_{ij} = e_c) - n\pi_i w_l \mu_{ilc}\right| + \left|\sum_{j=1}^{n} \widehat{q}_{jl}\mathbb{I}(z_{ij} = e_c) - \sum_{j=1}^{n} \mathbb{I}(q_{jl} = y_j)\mathbb{I}(z_{ij} = e_c)\right| \\ &\leq nt_{ilc} + n\delta_2.\end{aligned}$$

Similarly, using the assumption that event $\mathcal{E}_3$ and inequality (45) hold, we have

$$\begin{aligned}|B - n\pi_i w_l| &\leq \left|\sum_{j=1}^{n} \mathbb{I}(q_{jl} = y_j)\mathbb{I}(z_{ij} \neq 0) - n\pi_i w_l\right| + \left|\sum_{j=1}^{n} \widehat{q}_{jl}\mathbb{I}(z_{ij} \neq 0) - \sum_{j=1}^{n} \mathbb{I}(q_{jl} = y_j)\mathbb{I}(z_{ij} \neq 0)\right| \\ &\leq \frac{nt_{ilc}}{\mu_{ilc}} + n\delta_2.\end{aligned}$$

Combining the bound for A and B, we obtain that

$$\begin{aligned}|\widehat{\mu}_{ilc} - \mu_{ilc}| &= \left|\frac{n\pi_i w_l \mu_{ilc} + (A - n\pi_i w_l \mu_{ilc})}{n\pi_i w_l + (B - n\pi_i w_l)} - \mu_{ilc}\right| = \left|\frac{(A - n\pi_i w_l \mu_{ilc}) - \mu_{ilc}(B - n\pi_i w_l)}{n\pi_i w_l + (B - n\pi_i w_l)}\right| \\ &\leq \frac{2nt_{ilc} + 2n\delta_2}{n\pi_i w_l - n(t_{ilc}/\mu_{ilc}) - n\delta_2}.\end{aligned}$$

Condition (34) implies

$$\frac{t_{ilc}}{\mu_{ilc}} \leq \frac{t_{ilc}}{\rho} \leq \frac{\pi_{\min} w_{\min}\rho}{8\rho} = \frac{\pi_{\min} w_{\min}}{8},$$

lower bounding the denominator. Plugging in this bound completes the proof. ■

To proceed with the proof, we assign specific values to δ_1 and δ_2. Let

$$\delta_1 := \min\left\{\frac{\rho}{2}, \frac{\rho\overline{D}}{16}\right\} \quad \text{and} \quad \delta_2 := t_{\min}/2. \tag{47}$$

We claim that at any step in the update, the preconditions (42) and (45) always hold.

We prove the claim by induction. Before the iteration begins, $\widehat{\mu}$ is initialized such that the accuracy bound (10) holds. Thus, condition (42) is satisfied at the beginning. We assume by induction that condition (42) is satisfied at time $1, 2, \ldots, \tau - 1$ and condition (45) is satisfied at time $2, 3, \ldots, \tau - 1$. At time τ, either update (7) or update (8) is performed. If update (7) is performed, then by the inductive hypothesis, condition (42) holds before the update. Thus, Lemma 11 implies that

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j = l)|\} \leq \exp\left(-m\left(\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1}\right) + \log(k)\right).$$

The assignment (47) implies $\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1} \geq \frac{\overline{D}}{4}$, which yields that

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j = l)|\} \leq \exp(-m\overline{D}/4 + \log(k)) \leq t_{\min}/2 = \delta_2,$$

where the last inequality follows from condition (34). It suggests that condition (45) holds after the update.

On the other hand, we assume that update (8) is performed at time τ. Since update (8) follows update (7), we have $\tau \geq 2$. By the inductive hypothesis, condition (45) holds before the update, so Lemma 12 implies

$$|\widehat{\mu}_{ilc} - \mu_{ilc}| \leq \frac{2nt_{ilc} + 2n\delta_2}{(7/8)n\pi_i w_l - n\delta_2} = \frac{2nt_{ilc} + nt_{\min}}{(7/8)n\pi_i w_l - nt_{\min}/2} \leq \frac{3nt_{ilc}}{(7/8)n\pi_i w_l - nt_{\min}/2},$$

where the last step follows since $t_{\min} \leq t_{ilc}$. Noticing $\rho \leq 1$, condition (34) implies that $t_{\min} \leq \pi_{\min} w_{\min}/8$. Thus, the right hand side of the above inequality is bounded by $4t_{ilc}/(\pi_i w_l)$. Using condition (34) again, we find

$$\frac{4t_{ilc}}{\pi_i w_l} \leq \frac{4t_{ilc}}{\pi_{\min} w_{\min}} \leq \min\left\{\frac{\rho}{2}, \frac{\rho\overline{D}}{16}\right\} = \delta_1,$$

which verifies that condition (42) holds after the update. This completes the induction.

Since preconditions (42) and (45) hold for any time $\tau \geq 2$, Lemma 11 and Lemma 12 implies that the concentration bounds (43) and (46) always hold. These two concentration bounds establish the lemma's conclusion.

B.2 Proof of Lemma 10

By the definition of s_i, we have

$$\mathbb{E}[s_i^2] = \pi_i \sum_{c=1}^{k} \mu_{iy_jc}(\log(\mu_{iy_jc}/\mu_{ilc}))^2 = \pi_i \sum_{c=1}^{k} \mu_{iy_jc}(\log(\mu_{ilc}/\mu_{iy_jc}))^2$$

We claim that for any $x \geq \rho$ and $\rho < 1$, the following inequality holds:

$$\log^2(x) \leq \frac{2\log(1/\rho)}{1 - \rho}(x - 1 - \log(x)). \tag{48}$$

We defer the proof of inequality (48), focusing on its consequence. Let $x := \mu_{ilc}/\mu_{iy_jc}$, then inequality (48) yields that

$$\mathbb{E}[s_i^2] \leq \frac{2\log(1/\rho)}{1 - \rho}\pi_i\left(\sum_{c=1}^{k} \mu_{ilc} - \mu_{iy_jc} - \mu_{iy_jc}\log(\mu_{ilc}/\mu_{iy_jc})\right) = \frac{2\log(1/\rho)}{1 - \rho}\pi_i \mathbb{D}_{\mathrm{KL}}\left(\mu_{iy_j}, \mu_{il}\right).$$

It remains to prove the claim (48). Let $f(x) := \log^2(x) - \frac{2\log(1/\rho)}{1-\rho}(x - 1 - \log(x))$. It suffices to show that $f(x) \leq 0$ for $x \geq \rho$. First, we have $f(1) = 0$ and

$$f'(x) = \frac{2(\log(x) - \frac{\log(1/\rho)}{1-\rho}(x - 1))}{x}.$$

For any $x > 1$, we have

$$\log(x) < x - 1 \leq \frac{\log(1/\rho)}{1-\rho}(x-1)$$

where the last inequality holds since $\log(1/\rho) \geq 1 - \rho$. Hence, we have $f'(x) < 0$ and consequently $f(x) < 0$ for $x > 1$.

For any $\rho \leq x < 1$, notice that $\log(x) - \frac{\log(1/\rho)}{1-\rho}(x-1)$ is a concave function of x, and equals zero at two points $x = 1$ and $x = \rho$. Thus, $f'(x) \geq 0$ at any point $x \in [\rho, 1)$, which implies $f(x) \leq 0$.

Appendix C. Proof of Theorem 5

In this section we prove Theorem 5. The proof separates into two parts.

C.1 Proof of Part (a)

Throughout the proof, probabilities are implicitly conditioning on $\{\pi_i\}$ and $\{\mu_{ilc}\}$. We assume that (l, l') are the pair of labels such that

$$\overline{D} = \frac{1}{m}\sum_{i=1}^{m} \pi_i \mathbb{D}_{\mathrm{KL}}\left(\mu_{il}, \mu_{il'}\right).$$

Let $\mathbb{Q}$ be a uniform distribution over the set $\{l, l'\}^n$. For any predictor $\widehat{y}$, we have

$$\begin{aligned}\max_{v\in[k]^n} \mathbb{E}\Big[\sum_{j=1}^{n} \mathbb{I}(\widehat{y}_j \neq y_j)\Big| y = v\Big] &\geq \sum_{v\in\{l,l'\}^n} \mathbb{Q}(v)\, \mathbb{E}\Big[\sum_{j=1}^{n} \mathbb{I}(\widehat{y}_j \neq y_j)\Big| y = v\Big] \\ &= \sum_{j=1}^{n}\sum_{v\in\{l,l'\}^n} \mathbb{Q}(v)\, \mathbb{E}\Big[\mathbb{I}(\widehat{y}_j \neq y_j)\Big| y = v\Big]. \end{aligned} \tag{49}$$

Thus, it is sufficient to lower bound the right hand side of inequality (49).

For the rest of the proof, we lower bound the quantity $\sum_{y\in\{l,l'\}^n} \mathbb{Q}(v)\, \mathbb{E}[\mathbb{I}(\widehat{y}_j \neq y_j)|y]$ for every item j. Let $Z := \{z_{ij} : i \in [m],\ j \in [n]\}$ be the set of all observations. We define two probability measures $\mathbb{P}_0$ and $\mathbb{P}_1$, such that $\mathbb{P}_0$ is the measure of Z conditioning on $y_j = l$, while $\mathbb{P}_1$ is the measure of Z conditioning on $y_j = l'$. By applying Le Cam's method (Yu, 1997) and Pinsker's inequality, we have

$$\begin{aligned}\sum_{v\in\{l,l'\}^n} \mathbb{Q}(v)\, \mathbb{E}\Big[\mathbb{I}(\widehat{y}_j \neq y_j)\Big| y = v\Big] &= \mathbb{Q}(y_j = l)\mathbb{P}_0(\widehat{y}_j \neq l) + \mathbb{Q}(y_j = l')\mathbb{P}_1(\widehat{y}_j \neq l') \\ &\geq \frac{1}{2} - \frac{1}{2}\|\mathbb{P}_0 - \mathbb{P}_1\|_{\mathrm{TV}} \\ &\geq \frac{1}{2} - \frac{1}{4}\sqrt{\mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0, \mathbb{P}_1\right)}. \end{aligned} \tag{50}$$

The remaining arguments upper bound the KL-divergence between $\mathbb{P}_0$ and $\mathbb{P}_1$. Conditioning on y_j, the set of random variables $Z_j := \{z_{ij} : i \in [m]\}$ are independent of $Z\backslash Z_j$ for both

$\mathbb{P}_0$ and $\mathbb{P}_1$. Letting the distribution of X with respect to probability measure $\mathbb{P}$ be denoted by $\mathbb{P}(X)$, we have

$$\mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0, \mathbb{P}_1\right) = \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_j), \mathbb{P}_1(Z_j)\right) + \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z\backslash Z_j), \mathbb{P}_1(Z\backslash Z_j)\right) = \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_j), \mathbb{P}_1(Z_j)\right), \tag{51}$$

where the last step follows since $\mathbb{P}_0(Z\backslash Z_j) = \mathbb{P}_1(Z\backslash Z_j)$. Next, we observe that $z_{1j}, z_{2j}, \ldots, z_{mj}$ are mutually independent given y_j, which implies

$$\begin{aligned}\mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_j), \mathbb{P}_1(Z_j)\right) &= \sum_{i=1}^{m} \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(z_{ij}), \mathbb{P}_1(z_{ij})\right) \\ &= \sum_{i=1}^{m}\left[(1-\pi_i)\log\left(\frac{1-\pi_i}{1-\pi_i}\right) + \sum_{c=1}^{k}\pi_i\mu_{ilc}\log\left(\frac{\pi_i\mu_{ilc}}{\pi_i\mu_{il'c}}\right)\right] \\ &= \sum_{i=1}^{m}\sum_{c=1}^{k}\pi_i\mathbb{D}_{\mathrm{KL}}\left(\mu_{ilc}, \mu_{il'c}\right) = m\overline{D}. \end{aligned} \tag{52}$$

Combining inequality (50) with equations (51) and (52), we have

$$\sum_{v\in\{l,l'\}^n} \mathbb{Q}(v)\, \mathbb{E}\Big[\mathbb{I}(\widehat{y}_j \neq y_j)\Big| y = v\Big] \geq \frac{1}{2} - \frac{1}{4}\sqrt{m\overline{D}}.$$

Thus, if $m \leq 1/(4\overline{D})$, then the above inequality is lower bounded by $3/8$. Plugging this lower bound into inequality (49) completes the proof.

C.2 Proof of Part (b)

Throughout the proof, probabilities are implicitly conditioning on $\{\pi_i\}$ and $\{w_l\}$. We define two vectors

$$u_0 := \left(\frac{1}{2}, \frac{1}{2}, 0, \ldots, 0\right)^T \in \mathbb{R}^k \qquad \text{and} \qquad u_1 := \left(\frac{1}{2}+\delta, \frac{1}{2}-\delta, 0, \ldots, 0\right)^T \in \mathbb{R}^k,$$

where $\delta \leq 1/4$ is a scalar to be specified. Consider a m-by-k random matrix V whose entries are uniformly sampled from $\{0, 1\}$. We define a random tensor $u_V \in \mathbb{R}^{m\times k\times k}$, such that $(u_V)_{il} := u_{V_{il}}$ for all $(i, l) \in [m] \times [k]$. Givan an estimator $\widehat{\mu}$ and a pair of indices $(\bar{i}, \bar{l})$, we have

$$\sup_{\mu\in\mathbb{R}^{m\times k\times k}} \mathbb{E}\Big[\|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2^2\Big] \geq \sum_{v\in[k]^n} \mathbb{P}(y = v)\left(\sum_V \mathbb{P}(V)\, \mathbb{E}\Big[\|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2^2\Big|\mu = u_V, y = v\Big]\right). \tag{53}$$

For the rest of the proof, we lower bound the term $\sum_V \mathbb{P}(V)\, \mathbb{E}[\|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2^2|\mu = u_V, y = v]$ for every $v \in [k]^n$. Let $\widehat{V}$ be an estimator defined as

$$\widehat{V} = \begin{cases} 0 & \text{if } \|\widehat{\mu}_{\bar{i}\bar{l}} - u_0\|_2 \leq \|\widehat{\mu}_{\bar{i}\bar{l}} - u_1\|_2. \\ 1 & \text{otherwise.} \end{cases}$$

If $\mu = u_V$, then $\widehat{V} \neq V_{\bar{i}\bar{l}} \Rightarrow \|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2 \geq \frac{\sqrt{2}}{2}\delta$. Consequently, we have

$$\sum_V \mathbb{P}(V)\, \mathbb{E}[\|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2^2 | \mu = u_V, y = v] \geq \frac{\delta^2}{2} \mathrm{P}[\widehat{V} \neq V_{\bar{i}\bar{l}} | y = v]. \tag{54}$$

Let $Z := \{z_{ij} : i \in [m],\ j \in [n]\}$ be the set of all observations. We define two probability measures $\mathbb{P}_0$ and $\mathbb{P}_1$, such that $\mathbb{P}_0$ is the measure of Z conditioning on $y = v$ and $\mu_{\bar{i}\bar{l}} = u_0$, and $\mathbb{P}_1$ is the measure of Z conditioning on $y = v$ and $\mu_{\bar{i}\bar{l}} = u_1$. For any other pair of indices $(i, l) \neq (\bar{i}, \bar{l})$, $\mu_{il} = u_{V_{il}}$ for both $\mathbb{P}_0$ and $\mathbb{P}_1$. By this definition, the distribution of Z conditioning on $y = v$ and $\mu = u_V$ is a mixture of distributions $\mathbb{Q} := \frac{1}{2}\mathbb{P}_0 + \frac{1}{2}\mathbb{P}_1$. By applying Le Cam's method (Yu, 1997) and Pinsker's inequality, we have

$$\begin{aligned} \mathrm{P}[\widehat{V} \neq V_{\bar{i}\bar{l}} | y = v] &\geq \frac{1}{2} - \frac{1}{2}\|\mathbb{P}_0 - \mathbb{P}_1\|_{\mathrm{TV}} \\ &\geq \frac{1}{2} - \frac{1}{4}\sqrt{\mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0, \mathbb{P}_1\right)}. \end{aligned} \tag{55}$$

Conditioning on $y = v$, the set of random variables $Z_i := \{z_{ij} : j \in [n]\}$ are mutually independent for both $\mathbb{P}_0$ and $\mathbb{P}_1$. Letting the distribution of X with respect to probability measure $\mathbb{P}$ be denoted by $\mathbb{P}(X)$, we have

$$\mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0, \mathbb{P}_1\right) = \sum_{i=1}^m \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_i), \mathbb{P}_1(Z_i)\right) = \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_{\bar{i}}), \mathbb{P}_1(Z_{\bar{i}})\right), \tag{56}$$

where the last step follows since $\mathbb{P}_0(Z_i) = \mathbb{P}_1(Z_i)$ for all $i \neq \bar{i}$. Next, we let $J := \{j : v_j = \bar{l}\}$ and define a set of random variables $Z_{iJ} := \{z_{ij} : j \in J\}$. It is straightforward to see that Z_{iJ} is independent of $Z_i \backslash Z_{iJ}$ for both $\mathbb{P}_0$ and $\mathbb{P}_1$. Hence, we have

$$\begin{aligned} \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_{\bar{i}}), \mathbb{P}_1(Z_{\bar{i}})\right) &= \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_{\bar{i}J}), \mathbb{P}_1(Z_{\bar{i}J})\right) + \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_{\bar{i}} \backslash Z_{\bar{i}J}), \mathbb{P}_1(Z_{\bar{i}} \backslash Z_{\bar{i}J})\right) \\ &= \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_{\bar{i}J}), \mathbb{P}_1(Z_{\bar{i}J})\right), \end{aligned} \tag{57}$$

where the last step follows since $\mathbb{P}_0(Z_{\bar{i}} \backslash Z_{\bar{i}J}) = \mathbb{P}_1(Z_{\bar{i}} \backslash Z_{\bar{i}J})$. Finally, since $\mu_{\bar{i}\bar{l}}$ is explicitly given in both $\mathbb{P}_0$ and $\mathbb{P}_1$, the random variables contained in $Z_{\bar{i}J}$ are mutually independent. Consequently, we have

$$\begin{aligned} \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(Z_{\bar{i}J}), \mathbb{P}_1(Z_{\bar{i}J})\right) &= \sum_{j \in J} \mathbb{D}_{\mathrm{KL}}\left(\mathbb{P}_0(z_{\bar{i}j}), \mathbb{P}_1(z_{\bar{i}j})\right) = |J|\ \pi_{\bar{i}}\, \frac{1}{2} \log\left(\frac{1}{1 - 4\delta^2}\right) \\ &\leq \frac{5}{2}|J|\ \pi_{\bar{i}}\delta^2. \end{aligned} \tag{58}$$

Here, we have used the fact that $\log(1/(1 - 4x^2)) \leq 5x^2$ holds for any $x \in [0, 1/4]$.

Combining the lower bound (55) with upper bounds (56), (57) and (58), we find

$$\mathbb{P}[\widehat{V}_{il} \neq V_{il} | y = v] \geq \frac{3}{8}\mathbb{I}\left(\frac{5}{2}|J|\ \pi_{\bar{i}}\delta^2 \leq \frac{1}{4}\right).$$

Plugging the above lower bound into inequalities (53) and (54) implies that

$$\sup_{\mu \in \mathbb{R}^{m \times k \times k}} \mathbb{E}\left[\|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2^2\right] \geq \frac{3\delta^2}{16}\mathbb{P}\left[|\{j : y_j = \bar{l}\}| \leq \frac{1}{10\pi_{\bar{i}}\delta^2}\right].$$

Note than $|\{j : y_j = \bar{l}\}| \sim \mathrm{Binomial}(n, w_{\bar{l}})$. Thus, if we set

$$\delta^2 := \min\left\{\frac{1}{16}, \frac{1}{10\pi_{\bar{i}}w_{\bar{l}}n}\right\},$$

then $\frac{1}{10\pi_{\bar{i}}\delta^2}$ is greater than or equal to the median of $|\{j : y_j = \bar{l}\}|$, and consequently,

$$\sup_{\mu \in \mathbb{R}^{m \times k \times k}} \mathbb{E}\left[\|\widehat{\mu}_{\bar{i}\bar{l}} - \mu_{\bar{i}\bar{l}}\|_2^2\right] \geq \min\left\{\frac{3}{512}, \frac{3}{320\pi_{\bar{i}}w_{\bar{l}}n}\right\},$$

which establishes the theorem.

Appendix D. Proof of Theorem 6

Our proof strategy is briefly described as follow: We first upper bound the error of Step (1)-(2) in Algorithm 2. This upper bound is presented as lemma 13. Then, we analyze the performance of Step (3), taking the guarantee obtained from the previous two steps.

Lemma 13 *Assume that $\kappa_3 > 0$. Let $\widehat{p}_i$ be initialized by Step (1)-(2). For any scalar $0 < t < \frac{\overline{\kappa}\kappa_3^3}{18}$, the upper bound*

$$\max_{i \in [m]}\{|\widehat{p}_i - p_i|\} \leq \frac{18t}{\kappa_3^3} \tag{59}$$

holds with probability at least $1 - m^2\exp(-nt^2/2)$.

The rest of the proof upper bounds the error of Step (3). The proof follows very similar steps as in the proof of Theorem 4. We first define two events that will be shown holding with high probability.

$$\mathcal{E}_1: \ \sum_{i=1}^m \sum_{c=1}^k \mathbb{I}(z_{ij} = e_c)\log(\mu_{iy_jc}/\mu_{ilc}) \geq m\overline{D}/2 \qquad \text{for all } j \in [n] \text{ and } l \in [k] \backslash \{y_j\}.$$

$$\mathcal{E}_2: \ \left|\sum_{j=1}^n \mathbb{I}(z_{ij} = e_{y_j}) - np_i\right| \leq nt_i \qquad \text{for all } i \in [m].$$

Lemma 14 *Assume that $\mathcal{E}_1 \cap \mathcal{E}_2$ holds. Also assume that $\rho \leq p_i \leq 1 - \rho$ for all $i \in [m]$. If $\widehat{p}$ is initialized such that*

$$|\widehat{p}_i - p_i| \leq \alpha := \min\left\{\frac{\overline{\kappa}}{2}, \frac{\rho}{2}, \frac{\rho\overline{D}}{16}\right\} \qquad \textit{for all } i \in [m] \tag{60}$$

and scalars t_i satisfy

$$\exp\Big(-m\overline{D}/4+\log(k)\Big)\le t_i\le \min\left\{\frac{\rho}{4},\frac{\rho\overline{D}}{32}\right\} \tag{61}$$

Then the estimates $\widehat{p}$ and $\widehat{q}$ obtained by alternating updates (14) and (15) satisfy:

$$|\widehat{p}_i-p_i|\le 2t_i. \qquad \text{for all } i\in[m].$$
$$\max_{l\in[k]}\{|\widehat{q}_{jl}-\mathbb{I}(y_j=l)|\}\le \exp\big(-m\overline{D}/4+\log(k)\big) \qquad \text{for all } j\in[n].$$

As in the proof of Theorem 4, we can lower bound the probability of the event $\mathcal{E}_1\cap\mathcal{E}_2$ by applying Bernstein's inequality and the Chernoff bound. In particular, the following bound holds:

$$\mathbb{P}[\mathcal{E}_1\cap\mathcal{E}_2]\ge 1-kn\exp\left(-\frac{m\overline{D}}{33\log(1/\rho)}\right)-\sum_{i=1}^{m}2\exp\left(-\frac{nt_i^2}{3p_i}\right). \tag{62}$$

The proof of inequality (62) precisely follows the proof of Theorem 4.

Proof of upper bounds (a) and (b) in Theorem 6 To apply Lemma 14, we need to ensure that condition (60) holds. If we assign $t:=\alpha\kappa_3^3/18$ in Lemma 13, then condition (60) holds with probability at least $1-m^2\exp(-n\alpha^2\kappa_3^6/648)$. To ensure that this event holds with probability at least $1-\delta/3$, we need to have

$$n\ge\frac{648\log(3m^2/\delta)}{\alpha^2\kappa_3^6}. \tag{63}$$

By Lemma 14, for $\widehat{y}_j=y_j$ being true, it suffices to have

$$m>4\log(2k)/\overline{D} \tag{64}$$

To ensure that $\mathcal{E}_1\cap\mathcal{E}_2$ holds with probability at least $1-2\delta/3$, expression (62) needs to be lower bounded by $1-2\delta/3$. It is achieved by

$$m\ge\frac{33\log(1/\rho)\log(3kn/\delta)}{\overline{D}}\quad\text{and}\quad n\ge\frac{3p_i\log(6m/\delta)}{t_i^2} \tag{65}$$

If we choose

$$t_i:=\sqrt{\frac{3\log(6m/\delta)}{n}}. \tag{66}$$

then the second part of condition (65) is guaranteed. To ensure that t_{ilc} satisfies condition (61). We need to have

$$\sqrt{\frac{3\log(6m/\delta)}{n}}\ge\exp\Big(-m\overline{D}/4+\log(k)\Big)\quad\text{and}$$
$$\sqrt{\frac{3\log(6m/\delta)}{n}}\le\alpha/2.$$

The above two conditions requires that m and n satisfy

$$m\ge\frac{4\log(k\sqrt{n/(3\log(6m/\delta))})}{\overline{D}} \tag{67}$$
$$n\ge\frac{12\log(6m/\delta)}{\alpha^2}. \tag{68}$$

The five conditions (63), (64), (65), (67) and (68) are simultaneously satisfied if we have

$$m\ge\frac{33\log(1/\rho)\log(3kn/\delta)}{\overline{D}}\quad\text{and}$$
$$n\ge\frac{648\log(3m^2/\delta)}{\alpha^2\kappa_3^6}.$$

Under this setup, $\widehat{y}_j=y_j$ holds for all $j\in[n]$ with probability at least $1-\delta$. Combining equation (66) with Lemma 14, the bound

$$|\widehat{p}_i-p_i|\le 2\sqrt{\frac{3\log(6m/\delta)}{n}}$$

holds with probability at least $1-\delta$.

D.1 Proof of Lemma 13

We claim that after initializing $\widehat{p}$ via formula (13), it satisfies

$$\min\left\{\max_{i\in[m]}\{|\widehat{p}_i-p_i|\},\max_{i\in[m]}\{|\widehat{p}_i-(2/k-p_i)|\}\right\}\le\frac{18t}{\kappa_3^3} \tag{69}$$

with probability at least $1-m^2\exp(-nt^2/2)$. Assuming inequality (69), it is straightforward to see that this bound is preserved by the algorithm's step (2). In addition, step (2) ensures that $\frac{1}{m}\sum_{i=1}^m\widehat{p}_i\ge\frac{1}{k}$, which implies

$$\begin{aligned}\max_{i\in[m]}\{|\widehat{p}_i-(2/k-p_i)|\}&\ge\left|\frac{1}{m}\sum_{i=1}^m\widehat{p}_i-\left(\frac{2}{k}-\frac{1}{m}\sum_{i=1}^m p_i\right)\right|\\&\ge\frac{1}{k}-\left(\frac{1}{k}-\overline{\kappa}\right)=\overline{\kappa}>\frac{18t}{\kappa_3^3}.\end{aligned} \tag{70}$$

Combining inequalities (69) and (70) establishes the lemma.

We turn to prove claim (69). For any worker a and worker b, it is obvious that $\mathbb{I}(z_{aj}=z_{bj})$ are independent random variables for $j=1,2,\ldots,n$. Since

$$\mathbb{E}[\mathbb{I}(z_{aj}=z_{bj})]=p_ap_b+(k-1)\,\frac{1-p_a}{k-1}\,\frac{1-p_b}{k-1}=\frac{k}{k-1}\,(p_a-1/k)(p_b-1/k)+\frac{1}{k}$$

and $\frac{k-1}{k}(\mathbb{I}(z_{aj}=z_{bj})-\frac{1}{k})$ belongs to the interval $[-1,1]$, applying Hoeffding's inequality implies that

$$\mathbb{P}(|N_{ab}-(p_a-1/k)(p_b-1/k)|\le t)\ge 1-\exp(-nt^2/2)\quad\text{for any } t>0.$$

By applying the union bound, the inequality

$$|N_{ab} - (p_a - 1/k)(p_b - 1/k)| \le t \tag{71}$$

holds for all $(a, b) \in [m]^2$ with probability at least $1 - m^2 \exp(-nt^2/2)$. For the rest of the proof, we assume that this high-probability event holds.

Given an arbitrary index i, we take indices (a_i, b_i) such that

$$(a_i, b_i) = \arg\max_{(a,b)} \{|N_{ab}| : a \neq b \neq i\}. \tag{72}$$

We consider another two indices (a^*, b^*) such that $|p_{a^*} - 1/k|$ and $|p_{b^*} - 1/k|$ are the two greatest elements in $\{|p_a - 1/k| : a \in [m]\backslash\{i\}\}$. Let $\beta_i := p_i - 1/k$ be a shorthand notation, then inequality (71) and equation (72) yields that

$$|\beta_{a_i}\beta_{b_i}| \ge |N_{a_ib_i}| - t \ge |N_{a^*b^*}| - t \ge |\beta_{a^*}\beta_{b^*}| - 2t \ge |\beta_{a^*}\beta_{b^*}|/2, \tag{73}$$

where the last step follows since $2t \le \kappa_3^2/2 \le |\beta_{a^*}\beta_{b^*}|/2$. Note that $|\beta_{b_i}| \le |\beta_{a^*}|$ (since $|\beta_{a^*}|$ is the largest entry by its definition), inequality (73) implies that $|\beta_{a_i}| \ge \frac{|\beta_{b^*}\beta_{a^*}|}{2|\beta_{b_i}|} \ge \frac{|\beta_{b^*}|}{2} \ge \kappa_3/2$. By the same argument, we obtain $|\beta_{b_i}| \ge |\beta_{b^*}|/2 \ge \kappa_3/2$. To upper bound the estimation error, we write $|N_{ia_i}|$, $|N_{ib_i}|$, $|N_{a_ib_i}|$ in the form of

$$\begin{aligned} |N_{ia_i}| &= |\beta_i\beta_{a_i}| + \delta_1 \\ |N_{ib_i}| &= |\beta_i\beta_{b_i}| + \delta_2 \\ |N_{a_ib_i}| &= |\beta_{a_i}\beta_{b_i}| + \delta_3, \end{aligned}$$

where $|\delta_1|, |\delta_2|, |\delta_3| \le t$. Firstly, notice that $N_{ia_i}, N_{ib_i} \in [-1, 1]$, thus,

$$\left|\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}} - \sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|\beta_{a_i}\beta_{b_i}|}}\right| \le \left|\frac{1}{\sqrt{|N_{a_ib_i}|}} - \frac{1}{\sqrt{|\beta_{a_i}\beta_{b_i}|}}\right| \le \frac{t}{2(|\beta_{a_i}\beta_{b_i}| - t)^{3/2}} \le \frac{t}{(\kappa_3^2/4)^{3/2}}, \tag{74}$$

where the last step relies on the inequality $|\beta_{a_i}\beta_{b_i}| - t \ge \kappa_3^2/4$ obtained by inequality (73). Secondly, we upper bound the difference between $\sqrt{|N_{ia_i}N_{ib_i}|}$ and $\sqrt{|\beta_i^2\beta_{a_i}\beta_{b_i}|}$. If $|\beta_i| \le t$, using the fact that $|\beta_{a_i}|, |\beta_{b_i}| \le 1$, we have

$$\left|\sqrt{|N_{ia_i}N_{ib_i}|} - \sqrt{|\beta_i^2\beta_{a_i}\beta_{b_i}|}\right| \le \sqrt{|N_{ia_i}N_{ib_i}|} + \sqrt{|\beta_i^2\beta_{a_i}\beta_{b_i}|} \le \sqrt{4t^2} + \sqrt{t^2} \le 3t.$$

If $|\beta_i| > t$, using the fact that $|\beta_{a_i}|, |\beta_{b_i}| \in [\kappa_3/2, 1]$ and $|\beta_{a_i}\beta_{b_i}| \ge \kappa_3^2/2$, we have

$$\begin{aligned} \left|\sqrt{|N_{ia_i}N_{ib_i}|} - \sqrt{|\beta_i^2\beta_{a_i}\beta_{b_i}|}\right| &\le \frac{|\beta_i\beta_{b_i}\delta_1| + |\beta_i\beta_{a_i}\delta_2| + |\delta_1\delta_2|}{\sqrt{|\beta_i^2\beta_{a_i}\beta_{b_i}|}} \\ &\le \frac{|\delta_1|}{\sqrt{|\beta_{a_i}/\beta_{b_i}|}} + \frac{|\delta_2|}{\sqrt{|\beta_{b_i}/\beta_{a_i}|}} + \frac{|\delta_1\delta_2|}{t\sqrt{|\beta_{a_i}\beta_{b_i}|}} \\ &\le 3\sqrt{2}t/\kappa_3. \end{aligned}$$

Combining the above two upper bounds implies

$$\left|\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|\beta_{a_i}\beta_{b_i}|}} - \sqrt{\frac{|\beta_i^2\beta_{a_i}\beta_{b_i}|}{|\beta_{a_i}\beta_{b_i}|}}\right| = \frac{\left|\sqrt{|N_{ia_i}N_{ib_i}|} - \sqrt{|\beta_i^2\beta_{a_i}\beta_{b_i}|}\right|}{\sqrt{|\beta_{a_i}\beta_{b_i}|}} \le \frac{6t}{\kappa_3^2}. \tag{75}$$

Combining inequalities (74) and (75), we obtain

$$\left|\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}} - |\beta_i|\right| \le \frac{14t}{\kappa_3^3}. \tag{76}$$

Finally, we turn to analyzing the sign of N_{ia_1}. According to inequality (71), we have

$$N_{ia_1} = \beta_i\beta_{a_1} + \delta_4,$$

where $|\delta_4| \le t$. Following the same argument for β_{a_i} and β_{b_i}, it was shown that $|\beta_{a_1}| \ge \kappa_3/2$. We combine inequality (76) with a case study of $\mathrm{sign}(N_{ia_1})$ to complete the proof. Let

$$\widehat{p}_i := \frac{1}{k} + \mathrm{sign}(N_{ia_1})\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}}.$$

If $\mathrm{sign}(N_{ia_1}) \neq \mathrm{sign}(\beta_i\beta_{a_1})$, then $|\beta_i\beta_{a_1}| \le |\delta_4| \le t$. Thus, $|\beta_i| \le t/|\beta_{a_i}| \le 2t/\kappa_3$, and consequently,

$$\begin{aligned} \max\{|\widehat{p}_i - p_i|, |\widehat{p}_i - (2/k - p_i)|\} &\le \left|\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}} + |p_i - 1/k|\right| \\ &\le \left|\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}} - |p_i - 1/k|\right| + 2|p_i - 1/k| \le \frac{18t}{\kappa_3^3} \end{aligned} \tag{77}$$

Otherwise, we have $\mathrm{sign}(N_{ia_1}) = \mathrm{sign}(\beta_i\beta_{a_1})$ and consequently $\mathrm{sign}(\beta_i) = \mathrm{sign}(N_{ia_1})\mathrm{sign}(\beta_{a_1})$. If $\mathrm{sign}(\beta_{a_1}) = 1$, then $\mathrm{sign}(\beta_i) = \mathrm{sign}(N_{ia_1})$, which yields that

$$|\widehat{p}_i - p_i| = \left|\mathrm{sign}(N_{ia_1})\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}} - \mathrm{sign}(\beta_i)|\beta_i|\right| \le \frac{14t}{\kappa_3^3}. \tag{78}$$

If $\mathrm{sign}(\beta_{a_1}) = -1$, then $\mathrm{sign}(\beta_i) = -\mathrm{sign}(N_{ia_1})$, which yields that

$$|\widehat{p}_i - (2/k - p_i)| = \left|\mathrm{sign}(N_{ia_1})\sqrt{\frac{|N_{ia_i}N_{ib_i}|}{|N_{a_ib_i}|}} + \mathrm{sign}(\beta_i)|\beta_i|\right| \le \frac{14t}{\kappa_3^3}. \tag{79}$$

Combining inequalities (77), (78) and (79), we find that

$$\min\left\{\max_{i\in[m]}\{|\widehat{p}_i - p_i|\}, \max_{i\in[m]}\{|\widehat{p}_i - (2/k - p_i)|\}\right\} \le \frac{18t}{\kappa_3^3}.$$

which establishes claim (69).

D.2 Proof of Lemma 14

The proof follows the argument in the proof of Lemma 9. We present two lemmas upper bounding the error of update (14) and update (15), assuming proper initialization.

Lemma 15 *Assume that event $\mathcal{E}_1$ holds. If p and its estimate $\widehat{p}$ satisfies*

$$\rho \le p_i \le 1-\rho \quad \text{and} \quad |\widehat{p}_i - p_i| \le \delta_1 \qquad \text{for all } i \in [m], \tag{80}$$

and $\widehat{q}$ is updated by formula (14), then $\widehat{q}$ is bounded as:

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j = l)|\} \le \exp\left(-m\left(\frac{\overline{D}}{2} - \frac{2\delta_1}{\rho - \delta_1}\right) + \log(k)\right) \qquad \text{for all } j \in [n]. \tag{81}$$

Proof Following the proof of Lemma 11, the lemma is established since both $|\log(\widehat{p}_i/p_i)|$ and $|\log((1-\widehat{p}_i)/(1-p_i))|$ are bounded by $\log(\rho/(\rho-\delta_1))$. ■

Lemma 16 *Assume that event $\mathcal{E}_2$ holds. If $\widehat{q}$ satisfies*

$$\max_{l\in[k]}\{|\widehat{q}_{jl} - \mathbb{I}(y_j = l)|\} \le \delta_2 \qquad \text{for all } j \in [n], \tag{82}$$

and $\widehat{p}$ is updated by formula (15), then $\widehat{p}$ is bounded as:

$$|\widehat{p}_i - p_i| \le t_i + \delta_2. \qquad \text{for all } i \in [m]. \tag{83}$$

Proof By formula (15), we have

$$\widehat{p}_i - p_i = \frac{1}{n}\left(\sum_{j=1}^{n}\mathbb{I}(z_{ij} = e_{y_i}) - np_i\right) + \frac{1}{n}\sum_{j=1}^{n}\sum_{l=1}^{k}(\widehat{q}_{il} - \mathbb{I}(y_j = l))\mathbb{I}(z_{ij} = e_l).$$

Combining inequality (82) with the inequality implied by event $\mathcal{E}_2$ completes the proof. ■

Following the steps in the proof of Lemma 9, we assign specific values to δ_1 and δ_2. Let

$$\delta_1 := \min\left\{\frac{\rho}{2}, \frac{\rho\overline{D}}{16}\right\} \quad \text{and} \quad \delta_2 := \min_{i\in[m]}\{t_i\}.$$

By the same inductive argument for proving Lemma 9, we can show that the upper bounds (81) and (83) always hold after the first iteration. Plugging the assignments of δ_1 and δ_2 into upper bounds (81) and (83) completes the proof.

Appendix E. Basic Lemmas

In this section, we prove some standard lemmas that we use for proving technical results.

Lemma 17 (Matrix Inversion) *Let $A, E \in \mathbb{R}^{k\times k}$ be given, where A is invertible and E satisfies that $\|E\|_{\text{op}} \le \sigma_k(A)/2$. Then*

$$\|(A+E)^{-1} - A^{-1}\|_{\text{op}} \le \frac{2\|E\|_{\text{op}}}{\sigma_k^2(A)}.$$

Proof A little bit of algebra reveals that

$$(A+E)^{-1} - A^{-1} = (A+E)^{-1}EA^{-1}.$$

Thus, we have

$$\|(A+E)^{-1} - A^{-1}\|_{\text{op}} \le \frac{\|E\|_{\text{op}}}{\sigma_k(A)\sigma_k(A+E)}.$$

We can lower bound the eigenvalues of $A+E$ by $\sigma_k(A)$ and $\|E\|_{\text{op}}$. More concretely, since

$$\|(A+E)\theta\|_2 \ge \|A\theta\|_2 - \|E\theta\|_2 \ge \sigma_k(A) - \|E\|_{\text{op}}$$

holds for any $\|\theta\|_2 = 1$, we have $\sigma_k(A+E) \ge \sigma_k(A) - \|E\|_{\text{op}}$. By the assumption that $\|E\|_{\text{op}} \le \sigma_k(A)/2$, we have $\sigma_k(A+E) \ge \sigma_k(A)/2$. Then the desired bound follows. ■

Lemma 18 (Matrix Multiplication) *Let $A_i, E_i \in \mathbb{R}^{k\times k}$ be given for $i = 1, \ldots, n$, where the matrix A_i and the perturbation matrix E_i satisfy $\|A_i\|_{\text{op}} \le K_i$, $\|E_i\|_{\text{op}} \le K_i$. Then*

$$\left\|\prod_{i=1}^{n}(A_i + E_i) - \prod_{i=1}^{n}A_i\right\|_{\text{op}} \le 2^{n-1}\left(\sum_{i=1}^{n}\frac{\|E_i\|_{\text{op}}}{K_i}\right)\prod_{i=1}^{n}K_i.$$

Proof By the triangle inequality, we have

$$\begin{aligned}
\left\|\prod_{i=1}^{n}(A_i + E_i) - \prod_{i=1}^{n}A_i\right\|_{\text{op}} &= \left\|\sum_{i=1}^{n}\left(\prod_{j=1}^{i-1}A_j\right)\left(\prod_{k=i+1}^{n}(A_k + E_k)\right)E_i\right\|_{\text{op}} \\
&\le \sum_{i=1}^{n}\|E_i\|_{\text{op}}\left(\prod_{j=1}^{i-1}\|A_j\|_{\text{op}}\right)\left(\prod_{k=i+1}^{n}\|A_k + E_k\|_{\text{op}}\right) \\
&\le \sum_{i=1}^{n}2^{n-i}\frac{\|E_i\|_{\text{op}}}{K_i}\prod_{i=1}^{n}K_i \\
&= 2^{n-1}\left(\sum_{i=1}^{n}\frac{\|E_i\|_{\text{op}}}{K_i}\right)\prod_{i=1}^{n}K_i
\end{aligned}$$

which completes the proof. ■

Lemma 19 (Matrix and Tensor Concentration) *Let $\{X_j\}_{j=1}^n$, $\{Y_j\}_{j=1}^n$ and $\{Z_j\}_{j=1}^n$ be i.i.d. samples from some distribution over $\mathbb{R}^k$ with bounded support ($\|X\|_2 \le 1$, $\|Y\|_2 \le 1$ and $\|Z\|_2 \le 1$ with probability 1). Then with probability at least $1-\delta$,*

$$\left\| \frac{1}{n}\sum_{j=1}^n X_j \otimes Y_j - \mathbb{E}[X_1 \otimes Y_1] \right\|_F \le \frac{1+\sqrt{\log(1/\delta)}}{\sqrt{n}}. \tag{84}$$

$$\left\| \frac{1}{n}\sum_{j=1}^n X_j \otimes Y_j \otimes Z_j - \mathbb{E}[X_1 \otimes Y_1 \otimes Z_1] \right\|_F \le \frac{1+\sqrt{\log(k/\delta)}}{\sqrt{n/k}}. \tag{85}$$

Proof Inequality (84) is proved in Lemma D.1 of Anandkumar et al. (2015). To prove inequality (85), we note that for any tensor $T \in \mathbb{R}^{k\times k\times k}$, we can define k-by-k matrices $T_1, \ldots, T_k$ such that $(T_i)_{jk} := T_{ijk}$. As a result, we have $\|T\|_F^2 = \sum_{i=1}^k \|T_i\|_F^2$. If we set T to be the tensor on the left hand side of inequality (85), then

$$T_i = \frac{1}{n}\sum_{j=1}^n (Z_j^{(i)}\ X_j) \otimes Y_j - \mathbb{E}[(Z_j^{(i)}\ X_1) \otimes Y_1]$$

By applying the result of inequality (84), we find that with probability at least $1-k\delta'$, we have

$$\left\| \frac{1}{n}\sum_{j=1}^n X_j \otimes Y_j \otimes Z_j - \mathbb{E}[X_1 \otimes Y_1 \otimes Z_1] \right\|_F^2 \le k\left(\frac{1+\sqrt{\log(1/\delta')}}{\sqrt{n}}\right)^2.$$

Setting $\delta' = \delta/k$ completes the proof. ■

References

Anima Anandkumar, Daniel Hsu, and Sham M. Kakade. A method of moments for mixture models and hidden Markov models. In *Proceedings of the Annual Conference on Learning Theory*, 2012.

Anima Anandkumar, Rong Ge, Daniel Hsu, and Sham M. Kakade. A tensor spectral approach to learning mixed membership community models. In *Proceedings of the Annual Conference on Learning Theory*, 2013.

Anima Anandkumar, Rong Ge, Daniel Hsu, Sham M. Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

Animashree Anandkumar, Dean P. Foster, Daniel Hsu, Sham M. Kakade, and Yi-Kai Liu. A spectral algorithm for latent Dirichlet allocation. *Algorithmica*, 72(1):193–214, 2015.

Sivaraman Balakrishnan, Martin J. Wainwright, and Bin Yu. Statistical guarantees for the EM algorithm: From population to sample-based analysis. *Annals of Statisics (to appear)*, 2016.

Arun Tejasvi Chaganty and Percy Liang. Spectral experts for estimating mixtures of linear regressions. In *Proceedings of the International Conference on Machine Learning*, 2013.

Xi Chen, Qihang Lin, and Dengyong Zhou. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In *Proceedings of the International Conference on Machine Learning*, 2013.

Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. Aggregating crowdsourced binary ratings. In *Proceedings of the World Wide Web Conference*, 2013.

Sanjoy Dasgupta and Leonard Schulman. A probabilistic analysis of EM for mixtures of separated, spherical Gaussians. *Journal of Machine Learning Research*, 8:203–226, 2007.

Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society, Series C*, 28 (1):20–28, 1979.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Chao Gao and Dengyong Zhou. Minimax optimal convergence rates for estimating ground truth from crowdsourced labels. *arXiv preprint arXiv:1310.5764*, 2014.

Arpita Ghosh, Satyen Kale, and Preston McAfee. Who moderates the moderators? Crowdsourcing abuse detection in user-generated content. In *Proceedings of the ACM Conference on Electronic Commerce*, 2011.

Daniel Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

Prateek Jain and Sewoong Oh. Learning mixtures of discrete product distributions using spectral decompositions. In *Proceedings of the Conference on Learning Theory*, 2014.

David R. Karger, Sewoong Oh, and Devavrat Shah. Efficient crowdsourcing for multi-class labeling. In *Proceedings of the ACM SIGMETRICS*, 2013.

David R. Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research*, 62(1):1–24, 2014.

Matthew Lease and Gabriella Kazai. Overview of the TREC 2011 crowdsourcing track. In *Proceedings of TREC 2011*, 2011.

Erich L. Lehmann and George Casella. *Theory of Point Estimation*. Springer, 2nd edition, 2003.

Percy Liang. Partial information from spectral methods. NIPS Spectral Learning Workshop, 2013.

Qiang Liu, Jian Peng, and Alexander T Ihler. Variational inference for crowdsourcing. In *Advances in Neural Information Processing Systems*, 2012.

Fabio Parisi, Francesco Strino, Boaz Nadler, and Yuval Klugera. Ranking and combining multiple predictors without labeled data. *Proceedings of the National Academy of Sciences*, 111(4):1253–1258, 2014.

Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11: 1297–1322, 2010.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2008.

Peter Welinder, Steve Branson, Serge Belongie, and Pietro Perona. The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems*, 2010.

Jacob Whitehill, Ting F. Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems*, 2009.

Bin Yu. Assouad, Fano, and Le Cam. In *Festschrift for Lucien Le Cam*, pages 423–435. Springer, 1997.

Dengyong Zhou, John C. Platt, Sumit Basu, and Yi Mao. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems*, 2012.

Dengyong Zhou, Qiang Liu, John C. Platt, and Christopher Meek. Aggregating ordinal labels from crowds by minimax conditional entropy. In *Proceedings of the International Conference on Machine Learning*, 2014.

James Zou, Daniel Hsu, David Parkes, and Ryan Adams. Contrastive learning using spectral methods. In *Advances in Neural Information Processing Systems*, 2013.

Journal of Machine Learning Research 17 (2016) 1-38 Submitted 12/14; Revised 5/16; Published 6/16

Bayesian Leave-One-Out Cross-Validation Approximations for Gaussian Latent Variable Models

Aki Vehtari aki.vehtari@aalto.fi
Tommi Mononen
Ville Tolvanen
Tuomas Sivula
Helsinki Institute of Information Technology HIIT,
Department of Computer Science, Aalto University
P.O.Box 15400, 00076 Aalto, Finland

Ole Winther owi@imm.dtu.dk
Technical University of Denmark
DK-2800 Lyngby, Denmark

Editor: Kevin Murphy

Abstract

The future predictive performance of a Bayesian model can be estimated using Bayesian cross-validation. In this article, we consider Gaussian latent variable models where the integration over the latent values is approximated using the Laplace method or expectation propagation (EP). We study the properties of several Bayesian leave-one-out (LOO) cross-validation approximations that in most cases can be computed with a small additional cost after forming the posterior approximation given the full data. Our main objective is to assess the *accuracy of the approximative LOO cross-validation estimators*. That is, for each method (Laplace and EP) we compare the approximate fast computation with the exact brute force LOO computation. Secondarily, we evaluate the accuracy of the Laplace and EP approximations themselves against a ground truth established through extensive Markov chain Monte Carlo simulation. Our empirical results show that the approach based upon a Gaussian approximation to the LOO marginal distribution (the so-called cavity distribution) gives the most accurate and reliable results among the fast methods.

Keywords: predictive performance, leave-one-out cross-validation, Gaussian latent variable model, Laplace approximation, expectation propagation

1. Introduction

Bayesian cross-validation can be used to assess predictive performance. Vehtari and Ojanen (2012) provide an extensive review of theory and methods in Bayesian predictive performance assessment including decision theoretical assumptions made in Bayesian cross-validation. Gelman et al. (2014) provide further details on theoretical properties of leave-one-out cross-validation and information criteria, and Vehtari et al. (2016) provide practical fast computation in the case of Monte Carlo posterior inference. In this article, we present the properties of several Bayesian leave-one-out (LOO) cross-validation approximations for *Gaussian latent variable models* (GLVM) with factorizing likelihoods. Integration over the latent variables is performed with either the Laplace method or expectation propagation (EP). We show that for these methods leave-one-out cross-validation can be computed accurately with zero or a negligible additional cost after forming the full data posterior approximation.

Global (Gaussian) and factorizing variational approximations for latent variable inference are not considered in this paper. They have the same order computational complexity as Laplace and EP but with a larger pre-factor on the dominating $\mathcal{O}(n^3)$ term, where n is the number of observations (Nickisch and Rasmussen, 2008). EP may be expected to be the most accurate method (e.g. Nickisch and Rasmussen, 2008; Vanhatalo and Vehtari, 2010; Jylänki et al., 2011; Riihimäki et al., 2013) and Laplace to have the smallest computational overhead. So EP and Laplace may be considered the methods of choice for accuracy and speed, respectively. We expect that our overall results and conclusions for Laplace and EP carry over to Gaussian variational. For non-GLVM models such as generalized linear and deep generative models, the (factorized) Gaussian variational approximations scale to large data sets (Challis and Barber, 2013; Ranganath et al., 2014; Kingma and Welling, 2014; Rezende et al., 2014). It is of interest to derive approximate LOO estimators for these models, but that is outside the scope of this paper.

We consider a prediction problem with an explanatory variable x and an outcome variable y. The same notation is used interchangeably for scalar and vector-valued quantities. The observed data are denoted by $D = \{(x_i, y_i)\}_{i=1}^n$ and future observations by $(\tilde{x}, \tilde{y})$. We focus on GLVMs, where the observation model $p(y_i|f_i, \phi)$ depends on a local latent value f_i and possibly on some global parameters ϕ, such as the scale of the measurement error process. Latent values $f = (f_1, \ldots, f_n)$ have a joint Gaussian prior $p(f|x, \theta)$ which depends on covariates x and hyperparameters θ (e.g., covariance function parameters for a Gaussian process). The posterior of the latent f is then

$$p(f|D, \theta, \phi) \propto p(f|x, \theta) \prod_{i=1}^{n} p(y_i|f_i, \phi). \tag{1}$$

As a specific example we use Gaussian process (GP) models (reviewed, e.g., by Rasmussen and Williams, 2006), but the methods are applicable also for other GLVMs which have the same factorizing form (e.g. Gaussian Markov random field models used in the R-INLA software (Lindgren and Rue, 2015)). Some of the presented methods are applicable more generally, requiring only a factorizing likelihood with terms $p(y_i|f_i, \phi)$ and a method to integrate over the marginal posteriors $p(f_i|D, \theta, \phi)$. The results presented in this paper can be generalized to the cases where a likelihood term depends upon more than one latent variable (e.g. Tolvanen et al., 2014) or the latent value prior is non-Gaussian (e.g. Seeger, 2008; Hernández-Lobato et al., 2008; Hernáandez-Lobato et al., 2010). For clarity we restrict our treatment to the case of one likelihood term with one latent value.

We are interested in assessing the predictive performance of our models to report this to application experts or to perform model selection. For simplicity, in this paper we use only the logarithmic score, but the methods can be also be used with application specific utilities such as classification error. Logarithmic score is the standard scoring rule in Bayesian cross-validation (see Geisser and Eddy (1979)) and it has desirable properties for scientific inference (Bernardo and Smith, 1994; Gneiting and Raftery, 2007).

The predictive distribution for a future observation $\tilde{y}$ given future covariate values $\tilde{x}$ is

$$p(\tilde{y}|\tilde{x}, D) = \int p(\tilde{y}|\tilde{f}, \phi)p(\tilde{f}|\tilde{x}, D, \theta)p(\phi, \theta|D)d\tilde{f}d\phi d\theta. \quad (2)$$

The expected predictive performance using the log score and unknown true distribution of the future observation $p_t(\tilde{x}, \tilde{y})$ is

$$\int p_t(\tilde{x}, \tilde{y}) \log p(\tilde{y}|\tilde{x}, D)d\tilde{x}d\tilde{y}. \quad (3)$$

This expectation can be approximated by re-using the observations and computing the leave-one-out Bayesian cross-validation estimate

$$\text{LOO} = \frac{1}{n}\sum_{i=1}^{n} \log p(y_i|x_i, D_{-i}), \quad (4)$$

where D_{-i} is all other observations except (x_i, y_i). Here we consider only cases with random $\tilde{x}$ from the same distribution as x. See Vehtari and Ojanen (2012) for discussion of fixed, shifted, deterministic, or constrained $\tilde{x}$.

In addition to estimating the expected log predictive density, it may be interesting to look at a single value, $\log p(y_i|x_i, D_{-i})$. These terms, also called conditional predictive ordinates (CPO_i), may reveal observations which are highly influential or not well explained by the model (see, e.g., Gelfand, 1996). The probability integral transform (PIT) values $F(y_i|x_i, D_{-i})$, where F is the predictive CDF, can be used to assess the calibration of the predictions (see, e.g., Gneiting et al., 2007).

The straightforward brute force implementation of leave-one-out cross-validation requires recomputing the posterior distribution n times. Often leave-one-out cross-validation is replaced with k-fold cross-validation requiring only k recomputations of the posterior, with k usually 10 or less. Although k-fold-CV is robust and would often be computationally feasible, there are several fast approximations for computing LOO with a negligible additional computational cost after forming the posterior with the full data.

Several studies have shown that the Laplace method and EP perform well (compared to the gold standard Markov chain Monte Carlo inference) for GLVMs with many log-concave likelihoods (e.g. Rue et al., 2009; Vanhatalo et al., 2010; Martins et al., 2013; Riihimäki and Vehtari, 2014). EP has also been shown to be close to Markov chain Monte Carlo inference for classification models (log-concave likelihood, but potentially highly skewed posterior) and non-log-concave likelihoods (e.g. Nickisch and Rasmussen, 2008; Vanhatalo and Vehtari, 2010; Jylänki et al., 2011; Cseke and Heskes, 2011; Riihimäki et al., 2013; Vanhatalo et al., 2013; Tolvanen et al., 2014). In this paper we also consider the accuracy of approximative LOO with standard Markov chain Monte Carlo inference for LOO as our benchmark.

In practical data analysis work, it is useful to start with fast approximations and step by step check whether a computationally more expensive approach can improve the predictive accuracy. We propose the following three step approach:

1. Find the MAP estimate $(\hat{\phi}, \hat{\theta})$ using the Laplace method to approximately integrate over the latent values f.

2. Using $(\hat{\phi}, \hat{\theta})$ obtained in the previous step, use EP to integrate over the latent values and check whether the predictive performance improves substantially compared to using the Laplace method (we may also re-estimate $\hat{\phi}$ and $\hat{\theta}$).

3. Integrate over ϕ and θ and check whether integration over the parameters improves predictive performance.

Details of the computations involved are given in Sections 2 and 3. Based on these steps we can continue with the model that has the best predictive performance or the one that makes predictions fastest, or both. Often we also need to re-estimate models when data are updated or additional covariates become available, and then again a fast and accurate posterior approximation is useful. To follow the above approach, we need accurate predictive performance estimates for the Laplace method and EP.

The main contributions of this paper are:

- A unified presentation and thorough empirical comparison of methods for approximate LOO for Gaussian latent variable models with both log-concave and non-log-concave likelihoods and MAP and hierarchical approaches for handling hyperparameter inference (Section 3).

- The main conclusion from the empirical investigation (Section 4) is the observed superior accuracy/complexity tradeoff of Gaussian latent cavity distribution based LOO estimators. Although there are more accurate non-Gaussian approximations of the marginal posteriors, their use does not translate into substantial improvements in terms of LOO cross-validation accuracy and also introduces considerable instability. Using the widely applicable information criterion (WAIC) in the computation does not provide any benefits.

- The Laplace Gaussian cavity distribution (LA-LOO) (Section 3.5), although mentioned by Cseke and Heskes (2011), has not been used previously for LOO estimation. LOO consistency of LA-LOO using linear response theory is proved (Appendix A).

- Truncated weights quadrature integration (Section 3.7) inspired by truncated importance sampling is a novel way to stabilize the quadrature used in some LOO computations.

2. Gaussian Latent Variable Models

In this section, we briefly review the notation and methods for Gaussian latent variable models used in the rest of the article. We focus on Gaussian processes (see, e.g., Rasmussen and Williams, 2006), but most of the discussion also holds for other factorizing GLVMs. We consider models with a Gaussian prior $p(f|x, \theta)$ on latent values $f = (f_1, \ldots, f_n)$ and factorizing likelihood

$$p(f|D, \theta, \phi) = \frac{1}{Z}\prod_{i=1}^{n} p(y_i|f_i, \phi)p(f|X, \theta), \quad (5)$$

where Z is a normalization factor and equal to the marginal likelihood $p(y|X,\theta,\phi) = \int \prod_{i=1}^{n} p(y_i|f_i,\phi)p(f|X,\theta)df$. For example, in the Gaussian process framework the multivariate Gaussian prior on latent values is $p(f|x,\theta) = \mathrm{N}(f|\mu_0, K)$, where μ_0 is the prior mean and K is a covariance matrix constructed by a covariance function $K_{i,j} = k(x_i, x_j; \theta)$, which characterizes the correlation between two points. In this paper, we assume that the prior mean μ_0 is zero, but the results generalize to nonzero prior means as well.

2.1 Gaussian Observation Model

With a Gaussian observation model,

$$p(y_i|f_i,\sigma^2) = \mathrm{N}(y_i|f_i,\sigma^2), \tag{6}$$

where $\phi = \sigma^2$ is the noise variance, the conditional posterior of the latent variables is a multivariate Gaussian

$$\begin{aligned} &p(f|D,\theta,\phi) = \mathrm{N}(f|\mu,\Sigma),\\ &\quad\text{where}\quad \mu = K(K+\sigma^2 I)^{-1}y\\ &\quad\text{and}\quad \Sigma = (K^{-1}+\sigma^{-2}I)^{-1} = K - K(K+\sigma^2 I)^{-1}K. \end{aligned} \tag{7}$$

The marginal posterior is simply $p(f_i|D,\theta,\sigma^2) = \mathrm{N}(\mu_i, \Sigma_{ii})$ and the marginal likelihood $p(y|X,\theta,\sigma^2)$ can be computed analytically using properties of the multivariate Gaussian (see, e.g., Rasmussen and Williams, 2006).

2.2 Non-Gaussian Observation Model

In the case of a non-Gaussian likelihood, the conditional posterior $p(f|D,\theta,\phi)$ needs to be approximated. In this paper, we focus on expectation propagation (EP) and the Laplace method (LA), which form a multivariate Gaussian approximation of the joint latent posterior

$$q(f|D,\theta,\phi) = \frac{1}{Z}p(f|X,\theta)\prod_{i=1}^{n}\tilde{t}_i(f_i), \tag{8}$$

where the $\tilde{t}_i$ are (unnormalized) Gaussian approximations of the likelihood contributions. We use q to denote approximative joint and marginal distributions in general, or the specific approximation used in each case can be inferred from the context.

2.3 Expectation Propagation

Expectation propagation (Opper and Winther, 2000; Minka, 2001) approximates independent non-Gaussian likelihood terms by unnormalized Gaussian form site approximations (aka pseudo-observations),

$$p(y_i|f_i) \simeq \tilde{t}_i(f_i|\tilde{Z}_i,\tilde{\mu}_i,\tilde{\Sigma}_i) = \tilde{Z}_i\,\mathrm{N}(f_i|\tilde{\mu}_i,\tilde{\Sigma}_i), \tag{9}$$

where $\tilde{Z}_i = \int p(y_i|f_i)\,\mathrm{N}(f_i|\tilde{\mu}_i,\tilde{\Sigma}_i)df_i$, and $\tilde{\mu}_i$ and $\tilde{\Sigma}_i$ are the parameters of the site approximations, or *site parameters*. The joint latent posterior approximation is then

$$\begin{aligned} p(f|D,\phi,\theta) &= \frac{1}{Z}p(f|X,\theta)\prod_i p(y_i|f_i,\phi)\\ &\approx \frac{1}{Z_{\mathrm{EP}}}p(f|X,\theta)\prod_i \tilde{t}_i(f_i) = q(f|D,\phi,\theta), \end{aligned} \tag{10}$$

where Z is the normalization constant or the marginal likelihood, Z_{EP} is the EP approximation to the marginal likelihood and $q(f|D)$ is a multivariate Gaussian posterior approximation.

EP updates the site approximations by iteratively improving accuracy of the marginals. To update the ith site approximation, it is first removed from the marginal approximation to form a cavity distribution,

$$q_{-i}(f_i) \propto q(f_i|D)/\tilde{t}_i(f_i), \tag{11}$$

where the marginal $q(f_i|D)$ is obtained analytically using properties of the multivariate Gaussian.

The cavity distribution is combined with the original likelihood term to form a more accurate marginal distribution called the tilted distribution:

$$q_{-i}(f_i)p(y_i|f_i,\phi). \tag{12}$$

Minimization of Kullback-Leibler divergence from the tilted distribution to the marginal approximation corresponds to matching the moments of the distributions. Hence for Gaussian approximation, the zeroth, first and second moments of this tilted distribution are computed, for example, using one-dimensional numerical integration. The site parameters are updated so that moments of the marginal approximation $q(f_i|D)$ match the moments of the tilted distribution $q_{-i}(f_i)p(y_i|f_i,\phi)$. The new $q(f)$ can be computed after a single site approximation has been updated (sequential EP) or after all the site approximations have been updated (parallel EP).

2.4 Laplace Approximation

The Laplace approximation is constructed from the second-order Taylor expansion of $\log p(f|y,\theta,\phi)$ around the mode $\hat{f}$, which gives a Gaussian approximation to the conditional posterior,

$$q(f|D,\theta,\phi) = \mathrm{N}(f|\hat{f},\hat{\Sigma}) \approx p(f|D,\theta,\phi), \tag{13}$$

where $\hat{\Sigma} = (K^{-1}+\tilde{\Sigma}^{-1})^{-1}$ is the inverse of the Hessian at the mode with $\tilde{\Sigma}$ being a diagonal matrix with elements (e.g., Rasmussen and Williams, 2006; Gelman et al., 2013),

$$\tilde{\Sigma}_i = -\frac{1}{\nabla_i\nabla_i \log p(y_i|f_i,\phi)|_{f_i=\hat{f}_i}}. \tag{14}$$

From this joint Gaussian approximation we can analytically compute an approximation of the marginal posterior $p(f_i|D,\theta,\phi)$ and the marginal likelihood $p(y|x,\theta,\phi)$. The Laplace approximation can also be written as

$$q(f|D,\theta,\phi) = \frac{1}{Z} p(f|X,\theta) \prod_{i=1}^{n} \tilde{t}_i(f_i), \tag{15}$$

where $\tilde{t}_i(f_i)$ are Gaussian terms $\mathrm{N}(f_i|\tilde{\mu}_i, \tilde{\Sigma}_i)$ with

$$\tilde{\mu}_i = \hat{f} + \tilde{\Sigma}_i \nabla_i \log p(y_i|f_i,\phi)|_{f_i=\hat{f}_i}. \tag{16}$$

2.5 Marginal Posterior Approximations

Many leave-one-out approximation methods require explicit computation of full posterior marginal approximations. We thus review alternative Gaussian and non-Gaussian approximations of the marginal posteriors $p(f_i|D,\theta,\phi)$ following the article by Cseke and Heskes (2011). The exact joint posterior can be written as (dropping θ, ϕ and D for brevity)

$$p(f) \propto q(f) \prod_i \epsilon_i(f_i) \quad \text{with} \quad \epsilon_i(f_i) = p(y_i|f_i,\phi)/\tilde{t}_i(f_i), \tag{17}$$

where $\epsilon_i(f_i)$ is the ratio of the exact likelihood and the site term approximating the likelihood. By integrating over the other latent variables, the marginal posterior can be written as

$$p(f_i) \propto q(f_i)\epsilon_i(f_i) \underbrace{\int q(f_{-i}|f_i) \prod_{j\neq i} \epsilon_j(f_j) df_{-i}}_{c_i(f_i)}, \tag{18}$$

where f_{-i} represents all other latent variables except f_i. Local methods use $\epsilon_i(f_i)$ which depends locally only on f_i. Global methods additionally use an approximation of $c_i(f_i)$ which depends globally on all latent variables. Next we briefly review different marginal posterior approximations of this exact marginal (see Table 1 for a summary).

2.5.1 Gaussian Approximations

The simplest approximation is to use the Gaussian marginals $q(f_i)$, which are easily obtained from the joint Gaussian obtained by the Laplace approximation or expectation propagation; we call these LA-G and EP-G. By denoting the mean and variance of the pseudo observations (defined by the site terms) by $\tilde{\mu}_i$ and $\tilde{\sigma}_i^2$ respectively, the joint approximation has the same form as in the Gaussian case:

$$\begin{aligned} &q(f|D,\theta,\phi) = \mathrm{N}(\mu,\Sigma) \\ &\quad \text{with} \quad \mu = \Sigma\tilde{\Sigma}^{-1}\tilde{\mu}, \quad \text{and} \quad \Sigma = (K^{-1} + \tilde{\Sigma}^{-1})^{-1}, \end{aligned} \tag{19}$$

where $\tilde{\Sigma}$ is diagonal matrix with $\tilde{\Sigma}_{ii} = \tilde{\sigma}^2$. Then the marginal is simply $q(f_i) = \mathrm{N}(\mu_i, \Sigma_{ii})$.

Method	Improvement	Explanation
LA-G	-	Gaussian marginal $q(f_i)$ from the joint distribution
LA-L	local	tilted distribution $q(f_i)\tilde{t}_i(f_i)^{-1}p(y_i\|f_i,\phi)$
LA-TK	global	$q(f_i)\tilde{t}_i(f_i)^{-1}p(y_i\|f_i,\phi)c_i(f_i)$, where $c_i(f_i)$ is approximated using the Laplace approximation
LA-CM/CM2/FACT	global	$q(f_i)\tilde{t}_i(f_i)^{-1}p(y_i\|f_i,\phi)c_i(f_i)$, where $c_i(f_i)$ is approximated using the Laplace approximation with simplifications
EP-G	-	Gaussian marginal $q(f_i)$ from the joint distribution
EP-L	local	tilted distribution $q_{-i}(f_i)p(y_i\|f_i,\phi)$, where $q_{-i}(f_i)$ is obtained as a part of EP method
EP-FULL	global	$q_{-i}(f_i)p(y_i\|f_i,\phi)c_i(f_i)$, where $c_i(f_i)$ is approximated using EP
EP-1STEP/FACT	global	$q_{-i}(f_i)p(y_i\|f_i,\phi)c_i(f_i)$, where $c_i(f_i)$ is approximated using EP with simplifications

Table 1: Summary of the methods for computing approximate marginal posteriors. In global methods $c_i(f_i) = \int q(f_{-i}|f_i)\prod_{j\neq i}\epsilon_j(f_j)df_{-i}$ is a multivariate integral and $\epsilon_j(f_j) = p(y_j|f_j,\phi)/\tilde{t}_j(f_j)$.

2.5.2 Non-Gaussian Approximations Using a Local Correction

The simplest improvement to Gaussian marginals is to include the local term $\epsilon_i(f_i)$, and assume that the global term $c_i(f_i) \approx 1$. For EP the result is the tilted distribution $q(f_i)\epsilon_i(f_i) = q_{-i}(f_i)p(y_i|f_i,\phi)$ which is obtained as a part of the EP algorithm (Opper and Winther, 2000). As only the local terms are used to compute the improvement, Cseke and Heskes (2011) refer to it as the local improvement and denote the locally improved EP marginal as EP-L.

For the Laplace approximation, Cseke and Heskes (2011) propose a similar local improvement LA-L which can be written as $q(f_i)\tilde{t}_i(f_i)^{-1}p(y_i|f_i,\phi)$, where the site approximation $\tilde{t}_i(f_i)$ is based on the second order approximation of $\log p(y_i|f_i,\phi)$ (see Section 2.4). In Section 3.5, we propose an alternative way to compute the equivalent marginal improvement using a tilted distribution $q_{-i}(f_i)p(y_i|f_i,\phi)$, where the cavity distribution $q_{-i}(f_i)$ is based on a leave-one-out formula derived using linear response theory (Appendix A). The local methods EP-L and LA-L can improve the marginal posterior approximation only at the observed x, and the marginal posterior at new $\tilde{x}$ is the usual Gaussian predictive distribution.

2.5.3 Non-Gaussian Approximations Using a Global Correction

Global approximations also take into account the global term $c_i(f_i)$ by approximating the multidimensional integral in Equation (18), again using Laplace or EP. To obtain an approximation for the marginal distribution, the integral $c_i(f_i)$ has to be evaluated with several f_i values and the computations can be time consuming unless some simplifications are

used. Global methods can be used to obtain an improved non-Gaussian posterior marginal approximation also at the not yet observed $\tilde{x}$.

Using the Laplace approximation to evaluate $c_i(f_i)$ corresponds to an approach proposed by Tierney and Kadane (1986), and so we label the marginal improvement as LA-TK. Rue et al. (2009) proposed an approach that can be seen as a compromise between the computationally intensive LA-TK and the local approximation LA-L. Instead of finding the mode for each f_i, they evaluate the Taylor expansion around the conditional mean obtained from the joint approximation $q(f)$. The method is referred to as LA-CM. Cseke and Heskes (2011) propose the improvement LA-CM2 which adds a correction to take into account that the Taylor expansion is not done at the mode. To further reduce the computational effort, Rue et al. (2009) propose additional approximations with performance somewhere between LA-CM and LA-L. Rue et al. (2009) also discuss computationally efficient schemes for selecting values of f_i and interpolation or parametric model fitting to estimate the marginal density for other values of f_i. Cseke and Heskes (2011) propose similar approaches for EP, with EP-FULL corresponding to LA-TK, and EP-1STEP corresponding to LA-CM/LA-CM2. Cseke and Heskes (2011) also propose EP-FACT and LA-FACT which use factorized approximation to speed up the computation of the normalization terms.

The local improvements EP-L and LA-L are obtained practically for free and all global approximations are significantly slower. See Appendix B for the computational complexities of the global approximations. Based on the results by Cseke and Heskes (2011), EP-L is inferior to global approximations, but the difference is often small, and LA-L is often worse than the global approximations. Also based on the results by Cseke and Heskes (2011) and our own experiments, we chose to use LA-CM2 and EP-FACT as the global corrections in the experiments.

2.6 Integration Over the Parameters

To marginalize out the parameters θ and ϕ from the previously mentioned conditional posteriors, we can use the exact or approximated marginal likelihood $p(y|x,\theta,\phi)$ to form the marginal posterior for the parameters

$$p(\theta,\phi|D) \propto p(y|X,\theta,\phi)p(\theta,\phi), \tag{20}$$

and use numerical integration to integrate over θ and ϕ. Commonly used methods include various Monte Carlo algorithms (see list of references in Vanhatalo et al., 2013) as well as deterministic procedures, such as the central composite design (CCD) method by Rue et al. (2009). Using stochastic or deterministic samples, the marginal posterior can be approximated as

$$p(f|\mathcal{D}) \approx \sum_{s=1}^{S} p(f|\mathcal{D},\phi^s,\theta^s)w^s, \tag{21}$$

where w^s is a weight for the sample (ϕ^s,θ^s).

If the marginal posterior distribution $p(\theta,\phi|D)$ is narrow, which can happen if n is large and the dimensionality of (θ,ϕ) is small, then the effect of the integration over the parameters may be negligible and we can use Type II MAP, that is, choose $(\hat{\phi},\hat{\theta}) = \arg\max_{\phi,\theta} p(\phi,\theta|D)$.

Method	Based on
IS-LOO	importance sampling / importance weighting, Section 3.6
Q-LOO	quadrature integration, Section 3.7
TQ-LOO	truncated quadrature integration, Section 3.7
LA-LOO	same as Q-LOO with LA-L, Section 3.5
EP-LOO	same as Q-LOO with EP-L, obtained as byproduct of EP, Section 3.4
WAIC_G	matches the first two terms of the Taylor series expansion of LOO, Section 3.8
WAIC_V	matches the first three terms of the Taylor series expansion of LOO, Section 3.8

Table 2: Summary of the leave-one-out (LOO) cross-validation approximations reviewed.

3. Leave-One-Out Cross-Validation Approximations

We start by reviewing the generic exact LOO equations, which are then used to provide a unifying view of the different approximations in the subsequent sections. We first review some special cases and then more generic approximations. The LOO approximations and their abbreviations are listed in Table 2. The computational complexities of the LOO approximations have been collected in Appendix B.

3.1 LOO from the Full Posterior

Consider the case where we have not yet seen the ith observation. Then using Bayes' rule we can add information from the ith observation:

$$p(f_i|D) = \frac{p(y_i|f_i)p(f_i|x_i,D_{-i})}{p(y_i|x_i,D_{-i})}, \tag{22}$$

again dropping ϕ and θ for brevity. Correspondingly we can remove the effect of the ith observation from the full posterior:

$$p(f_i|x_i,D_{-i}) = \frac{p(f_i|D)p(y_i|x_i,D_{-i})}{p(y_i|f_i)} \tag{23}$$

If we now integrate both sides over f_i and rearrange the terms we get

$$p(y_i|x_i,D_{-i}) = 1/\int \frac{p(f_i|D)}{p(y_i|f_i)} df_i. \tag{24}$$

In theory this gives the exact LOO result, but in practice we usually need to approximate $p(f_i|D)$ and the integral over f_i. In the following sections we first discuss the hierarchical approach, then the analytic, Monte Carlo, quadrature, WAIC, and Taylor series approaches for computing the conditional version of Equation (24). We then consider how the different marginal posterior approximations affect the result.

In some cases, we can compute $p(f_i|x_i,D_{-i})$ exactly or approximate it efficiently and then we can compute the LOO predictive density for y_i,

$$p(y_i|x_i,D_{-i}) = \int p(y_i|f_i)p(f_i|x_i,D_{-i})df_i. \tag{25}$$

Or, if we are interested in the predictive distribution for a new observation $\tilde{y}_i$, we can compute

$$p(\tilde{y}_i|x_i, D_{-i}) = \int p(\tilde{y}_i|f_i)p(f_i|x_i, D_{-i})df_i, \tag{26}$$

which is evaluated with different values of $\tilde{y}_i$ as it is not fixed like y_i.

3.2 Hierarchical Approximations

Instead of approximating the leave-one-out predictive density $p(y_i|x_i, D_{-i})$ directly, for hierarchical models such as GLVMs it is often easier to first compute the leave-one-out predictive density conditional on the parameters $p(y_i|x_i, D_{-i}, \theta, \phi)$, then compute the leave-one-out posteriors for the parameters $p(\theta, \phi|D_{-i})$ and combine the results

$$p(y_i|x_i, D_{-i}) = \int p(y_i|x_i, D_{-i}, \theta, \phi)p(\theta, \phi|D_{-i})d\theta d\phi. \tag{27}$$

Sometimes the leave-one-out posterior of the hyperparameters is close to the full posterior, that is, $p(\theta, \phi|D_{-i}) \approx p(\theta, \phi|D)$. The joint leave-one-out posterior can be then approximated as

$$p(f_i|x_i, D_{-i}) \approx \int p(f_i|x_i, D_{-i}, \theta, \phi)p(\theta, \phi|D)d\theta d\phi \tag{28}$$

(see, e.g., Marshall and Spiegelhalter, 2003). This approximation is a reasonable alternative if removing (x_i, y_i) has only a small impact on $p(\theta, \phi|D)$ but a larger impact on $p(f_i|D, \phi, \theta)$. Furthermore, if the posterior $p(\theta, \phi|D)$ is narrow, a Type II MAP point estimate of the parameters $\hat{\phi}, \hat{\theta}$ may produce similar predictions as integrating over the parameters,

$$p(f_i|x_i, D_{-i}) \approx p(f_i|x_i, D_{-i}, \hat{\theta}, \hat{\phi}). \tag{29}$$

3.3 LOO with Gaussian Likelihood

If both $p(y_i|f_i, \phi)$ and $p(f|\theta)$ are Gaussian, then we can compute $p(f_i|x_i, D_{-i})$ analytically. Starting from the marginal posterior we can remove the contribution of the ith factor in the likelihood:

$$\begin{aligned} p(f_i|x_i, D_{-i}, \theta, \phi) &\propto \frac{p(f_i|D, \theta)}{p(y_i|f_i, \phi)} \\ &= \mathrm{N}(f_i|\mu_{-i}, v_{-i}), \end{aligned} \tag{30}$$

where

$$\begin{aligned} \mu_{-i} &= v_{-i}(\Sigma_{ii}^{-1}\mu_i - \sigma^{-2}y_i) \\ v_{-i} &= \left(\Sigma_{ii}^{-1} - \sigma^{-2}\right)^{-1}. \end{aligned} \tag{31}$$

These equations correspond to the cavity distribution equations in EP.

Sundararajan and Keerthi (2001) derived the leave-one-out predictive distribution $p(y_i|x_i, D_{-i})$ directly from the joint posterior using prediction equations and properties of

partitioned matrices. This gives a numerically alternative but mathematically equivalent way to compute the leave-one-out posterior mean and variance:

$$\begin{aligned} \mu_{-i} &= y_i - \bar{c}_{ii}^{-1}g_i \\ v_{-i} &= \bar{c}_{ii}^{-1} - \sigma^2, \end{aligned} \tag{32}$$

where

$$\begin{aligned} g_i &= \left[(K + \sigma^2 I)^{-1}y\right]_i \\ \bar{c}_{ii} &= \left[(K + \sigma^2 I)^{-1}\right]_{ii}. \end{aligned} \tag{33}$$

Sundararajan and Keerthi (2001) also provided the equation for the LOO log predictive density

$$\log p(y_i|x_i, D_{-i}, \theta, \phi) = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log \bar{c}_{ii} - \frac{1}{2}\frac{g_i^2}{\bar{c}_{ii}}. \tag{34}$$

Instead of integrating over the parameters, Sundararajan and Keerthi (2001) used the result (and its gradient) to find a point estimate for the parameters maximizing the LOO log predictive density.

3.4 LOO with Expectation Propagation

In EP, the leave-one-out marginal posterior of the latent variable is computed explicitly as a part of the algorithm. The cavity distribution (11) is formed from the marginal posterior approximation by removing the site approximation (pseudo observation) using (31) and can be used to approximate the LOO posterior

$$p(f_i|x_i, D_{-i}, \theta, \phi) \approx q_{-i}(f_i). \tag{35}$$

The approximation for the LOO predictive density

$$p(y_i|x_i, D_{-i}, \theta, \phi) \approx \int p(y_i|f_i)q_{-i}(f_i)df_i \tag{36}$$

is the same as the zeroth moment of the tilted distribution. Hence we obtain an approximation for $p(f_i|x_i, D_{-i}, \theta, \phi)$ and $p(y_i|x_i, D_{-i}, \theta, \phi)$ as a free by-product of the EP algorithm. We denote this approach as EP-LOO. For certain likelihoods (36) can be computed analytically, but otherwise quadrature methods with a controllable error tolerance are usually used.

The EP algorithm uses all observations when converging to its fixed point and thus the cavity distribution $q_{-i}(f_i)$ technically depends on the observation y_i. Opper and Winther (2000) showed using linear response theory that the cavity distribution is up to first order leave-one-out consistent. Opper and Winther (2000) also showed experimentally in one case that the cavity distribution approximation is accurate. Cseke and Heskes (2011) did not consider LOO, but compared visually the tilted distribution marginal approximation EP-L to many global marginal posterior improvements. Based on these results, EP-L has some error on the shape of the marginal approximation if there is a strong prior correlation, but even then the zeroth moment — the LOO predictive density — is accurate. Our experiments provide much more evidence of the excellent accuracy of the EP-LOO approximation.

3.5 LOO with Laplace Approximation

Using linear response theory, which was used by Opper and Winther (2000) to prove LOO consistency of EP, we also prove the LOO consistency of Laplace approximation (derivation in Appendix A). Hence, we obtain a good approximation for $p(f_i|x_i, D_{-i}, \theta, \phi)$ also as a free by-product of the Laplace method. Linear response theory can be used to derive two alternative ways to compute the cavity distribution $q_{-i}(f_i)$.

The Laplace approximation can be written in terms of the Gaussian prior times the product of (unnormalized) Gaussian form site approximations. Cseke and Heskes (2011) define the LA-L marginal approximation as $q(f_i)\tilde{t}_i(f_i)^{-1}p(y_i|f_i, \phi)$, from which the cavity distribution, that is the leave-one-out distribution, follows as $q_{-i}(f_i) = q(f_i)\tilde{t}_i(f_i)^{-1}$. It can be computed using (31). We refer to this approach as LA-LOO. The LOO predictive density can be obtained by numerical integration

$$p(y_i|x_i, D_{-i}, \theta, \phi) \approx \int q_{-i}(f_i)p(y_i|f_i, \phi)df_i. \tag{37}$$

An alternative way to compute the Laplace LOO derived using linear response theory is

$$p(f_i|x_i, D_{-i}, \theta, \phi) \approx \mathrm{N}(f_i|\hat{f}_i - v_{-i}\hat{g}_i, v_{-i}), \tag{38}$$

where $\hat{f}$ is the posterior mode, $\hat{g}_i = \nabla_i \log p(y_i|f_i)|_{f_i=\hat{f}_i}$ is the derivative of the log likelihood at the mode, and

$$v_{-i} = \left(\frac{1}{\Sigma_{ii}} - \frac{1}{\tilde{\Sigma}_i}\right)^{-1}. \tag{39}$$

If we consider having pseudo observations with means $\hat{f}_i$ and variances $1/\hat{h}_i$, then these resemble the exact LOO equations for a Gaussian likelihood given in Section 3.3.

3.6 Importance Sampling and Weighting

A generic approach not restricted to GLVMs is based on obtaining Monte Carlo samples $(f_i^s, \phi^s, \theta^s)$ from the full posterior $p(f_i, \phi, \theta|D)$ and approximating (24) as

$$p(y_i|x_i, D_{-i}) \approx \frac{1}{\frac{1}{S}\sum_{s=1}^{S}\frac{1}{p(y_i|f_i^s,\phi^s)}}, \tag{40}$$

where θ^s drops out since y_i is independent of θ^s given f_i^s and ϕ^s. This approach was first proposed by Gelfand et al. (1992) (see also, Gelfand, 1996) and it corresponds to importance sampling (IS) where the full posterior is used as the proposal distribution. We refer to this approach as IS-LOO.

A more general importance sampling form is

$$p(\tilde{y}_i|x_i, D_{-i}) \approx \frac{\sum_{s=1}^{S} p(\tilde{y}_i|f_i^s, \phi^s)w_i^s}{\sum_{s=1}^{S} w_i^s}, \tag{41}$$

where w_i^s are importance weights and

$$w_i^s = \frac{p(f_i^s|x_i, D_{-i})}{p(f_i^s|D)} \propto \frac{1}{p(y_i|f_i^s, \phi^s)}. \tag{42}$$

This form shows the importance weights explicitly and allows the computation of other leave-one-out quantities like the LOO predictive distribution. If the predictive density $p(\tilde{y}_i|f_i^s, \phi^s)$ is evaluated with the observed value $\tilde{y}_i = y_i$, Equation (41) reduces to (40).

The approximation (40) has the form of the harmonic mean, which is notoriously unstable (see, e.g., Newton & Raftery 1994). However the leave-one-out version is not as unstable as the harmonic mean estimator of the marginal likelihood, which uses the harmonic mean of $\prod_{i=1}^{n} p(y_i|f_i^s, \phi^s)$ and corresponds to using the joint posterior as the importance sampling proposal distribution for the joint prior.

3.6.1 Integrated Importance Sampling

For the Gaussian observation model, Vehtari (2001) and Vehtari and Lampinen (2002) used exact computation for $p(y_i|x_i, D_{-i}, \theta, \phi)$ and importance sampling only for $p(\theta, \phi|D_{-i})$. The integrated importance weights are then

$$w_i^s \propto \frac{1}{p(y_i|x_i, D_{-i}, \theta^s, \phi^s)}, \tag{43}$$

and the LOO predictive density is

$$p(y_i|x_i, D_{-i}) \approx \frac{1}{\sum_{s=1}^{S}\frac{1}{p(y_i|x_i,D_{-i},\theta^s,\phi^s)}}. \tag{44}$$

The same marginalization approach can be used in the case of non-Gaussian observation models. Held et al. (2010) used the Laplace approximation, marginal improvements, and numerical integration to obtain an approximation for $p(y_i|x_i, D_{-i}, \theta^s, \phi^s)$ (see more in Section 3.7). Vanhatalo et al. (2013) use EP and the Laplace method for the marginalisation in the GPstuff toolbox. Li et al. (2014) considered generic latent variable models using Monte Carlo inference, and propose to marginalise f_i by obtaining additional Monte Carlo samples from the posterior $p(f_i|x_i, D_{-i}, \theta, \phi)$. Li et al. (2014) also proposed the name integrated IS and provided useful results illustrating the benefits of the marginalization. As we are focusing on EP and Laplace approximations for the latent inference, in our experiments we use IS only for hyperparameters.

3.6.2 The Variance of Importance Weights

The variance of the estimate (40) depends on the variance of the importance weights. The full posterior marginal $p(f_i^s|D)$ is likely to be narrower and have thinner tails than the leave-one-out distribution $p(f_i^s|x_i, D_{-i})$. This may cause the importance weights to have high or infinite variance (Peruggia, 1997; Epifani et al., 2008) as rare samples from the low density region in the tails of $p(f_i^s|D)$ may have very large weights.

If the variance of the importance weights is finite, the central limit theorem holds (Epifani et al., 2008), and the corresponding estimates converge quickly. If the variance of the raw importance ratios is infinite but the mean exists, the generalized central limit theorem for stable distributions holds, and the convergence of the estimate is slower (Vehtari and Gelman, 2015).

If the variance of the weights is finite, then an effective sample size estimate can be estimated as

$$S_{\text{eff}} = 1/\sum_{s=1}^{S}(\tilde{w}^s)^2, \tag{45}$$

where $\tilde{w}^s$ are normalized weights (with a sum equal to one) (Kong et al., 1994). This estimate is noisy if the variance is large, but with smaller variances it provides an easily interpretable estimate of the efficiency of the importance sampling.

3.6.3 Pareto Smoothed Importance Sampling

Vehtari et al. (2016) propose to use Pareto smoothed importance sampling (PSIS) by Vehtari and Gelman (2015) for diagnostics and to regularize the importance weights in IS-LOO. Pareto smoothed importance sampling uses the empirical Bayes estimate of Zhang and Stephens (2009) to fit a generalized Pareto distribution to the tail. By examining the estimated shape parameter $\hat{k}$ of the Pareto distribution, we are able to obtain sample based estimates of the existence of the moments (Koopman et al., 2009). When the tail of the weight distribution is long, a direct use of importance sampling is sensitive to one or few largest values. To stabilize the estimates, Vehtari et al. (2016) propose to replace the M largest weights by the expected values of the order statistics of the fitted generalized Pareto distribution. Vehtari et al. (2016) also apply optional truncation for very large weights following truncated importance sampling by Ionides (2008) to guarantee finite and reduced variance of the estimate in all cases. Even if the raw importance weights do not have finite variance, the PSIS-LOO estimate still has a finite variance, although at the cost of additional bias. Vehtari et al. (2016) demonstrate that this bias is likely to be small when the estimated Pareto shape parameter $\hat{k} < 0.7$.

3.6.4 Importance Weighting of Deterministic Points

Importance weighting can also be used with deterministic evaluation points (ϕ^s, θ^s) obtained from, for example, grid or CCD by re-weighting the weights w^s in (21); see Held et al. (2010) and Vanhatalo et al. (2013). As the deterministic points are usually used in the low dimensional case and the evaluation points are not far in the tails, the variance of the observed weights is usually smaller than with Monte Carlo. If the full posterior $p(\theta, \phi|D)$ is a poor fit to each LOO posterior $p(\theta, \phi|D_{-i})$, then the problem remains that the tails are not well approximated and LOO is biased towards the hierarchical approximation (28) that uses the full posterior of the parameters $p(\theta, \phi|D)$.

In the ideal case, the CCD evaluation points except the modal point would have equal weights. The CCD approach adjusts these weights based on the actual density and importance weighting will further adjust them, making it possible that a small number of evaluation points have large weights. Although the CCD evaluation points have been chosen deterministically, we can diagnose the reliability of CCD by investigating the distribution of the weights. If there is a small number of CCD points, we examine the effective sample size, and in cases where the number of points exceed 280 (which happens when there are more than 11 parameters), we also estimate the Pareto shape parameter $\hat{k}$.

3.7 Quadrature LOO

Held et al. (2010) proposed to use numerical integration to approximate

$$p(y_i|x_i, D_{-i}, \theta, \phi) \approx 1/\int \frac{q(f_i|D, \theta, \phi)}{p(y_i|f_i, \phi)} df_i. \tag{46}$$

We call this quadrature LOO (Q-LOO), as one-dimensional numerical integration methods are usually called quadrature. Given exact $p(f_i|D, \theta, \phi)$ and accurate quadrature, this would provide an accurate result (e.g., if the true posterior is Gaussian, quadrature should give a result similar to the analytic solution apart from numerical inaccuracies). However, some error will be introduced when the latent posterior is approximated with $q(f_i|D, \theta, \phi)$. The numerical integration of the ratio expression may also be numerically unstable if the tail of the likelihood term $p(y_i|f_i, \phi)$ decays faster than the tail of the approximation $q(f_i|D, \theta, \phi)$. For example, the probit likelihood, which has a tail that goes as $\exp(-f^2/2)/f$, will be numerically unstable if $q(f_i|D, \theta, \phi)$ is Gaussian with a variance below one.

Held et al. (2010) tested the Gaussian marginal approximation (LA-G) and two non-Gaussian improved marginal approximations (LA-CM and simplified LA-CM, see Section 2.5). All had problems with the tails, although less so with the more accurate approximations. Held et al. proposed to rerun the failed LOO cases with actual removal of the data. As Held et al. had 13 to 56 failures in their experiments, the proposed approach would make LOO relatively expensive. In our experiments with Gaussian marginal approximations LA-G/EP-G, we also had several severe failures with some data sets. However with the non-Gaussian approximations LA-CM2/EP-FACT, we did not observe severe failures (see Section 4).

If we use marginal approximations EP-L or LA-L based on the tilted distribution $q_{-i}(f_i)p(y_i|f_i, \phi)$ (see Table 1), we can see that the tail problem vanishes. Inserting the normalized tilted distribution from (46), the equation reduces to

$$p(y_i|x_i, D_{-i}, \theta, \phi) \approx \int q_{-i}(f_i)p(y_i|f_i, \phi)df_i, \tag{47}$$

which is the EP-LOO or LA-LOO predictive density estimate depending on which approximation is used.

3.7.1 Alternative Form for Quadrature LOO

We also present an alternative form of (46), which gives additional insight about the numerical stability when the global marginal improvements are used. As discussed in Section 2.5, we can write the marginal approximation with a global improvement as

$$\frac{Z_q}{Z_p} q(f_i)\tilde{t}(f_i)^{-1} p(y_i|f_i, \phi)c_i(f_i), \tag{48}$$

where $c_i(f_i)$ is a global correction term (see Eq. (18)). Replacing $q(f_i)\tilde{t}(f_i)^{-1}$ with the cavity distribution from EP-L or LA-L gives

$$\frac{Z_q}{Z_p} q_{-i}(f_i)p(y_i|f_i, \phi)c_i(f_i), \tag{49}$$

which we can insert into (46) to obtain

$$p(y_i|x_i, D_{-i}, \theta, \phi) \approx \frac{\int p(y_i|f_i, \phi) q_{-i}(f_i) c_i(f_i) df_i}{\int q_{-i}(f_i) c_i(f_i) df_i}. \tag{50}$$

Here $q_{-i}(f_i)c_i(f_i)$ is a global corrected leave-one-out posterior, and we can see that the stability will depend on $c_i(f_i)$. The correction term $c_i(f_i)$ may have increasing tails, which is usually not a problem in $q_{-i}(f_i)p(y_i|f_i, \phi)c_i(f_i)$, but may be a problem in $q_{-i}(f_i)c_i(f_i)$. In addition, the evaluation of $c_i(f_i)$ at a small number of points and using interpolation for the quadrature (as proposed by Rue et al., 2009) is sometimes unstable, which may increase the instability of $\int q_{-i}(f_i)c_i(f_i)df_i$. Depending on the details of the computation, (46) and (50) can produce the same result up to numerical accuracy, if the relevant terms cancel out numerically in Equation (46). This happens in our implementation with global marginal posterior improvements, and thus in Section 4 we do not report the results separately for (46) and (50).

Held et al. (2010) and Vanhatalo et al. (2013) use quadrature LOO in a hierarchical approximation, where the parameter level is handled using importance weighting (Section 3.6). Our experiments also use this approach. Alternatively, we could approximate by integrating over the parameters in the marginal and likelihood separately and approximate LOO as

$$p(y_i|x_i, D_{-i}) \approx 1/\int \frac{q(f_i|D)}{p(y_i|f_i, D)} df_i. \tag{51}$$

If the integration over θ and ϕ is made using Monte Carlo or deterministic sampling (e.g. CCD), then this is equivalent to using quadrature for conditional terms and importance weighting of the parameter samples.

3.7.2 Truncated Weights Quadrature

As the quadrature approach may also be applied beyond simple GLVMs, we propose an approach for stabilizing the general form. Inspired by truncated importance sampling by Ionides (2008), we propose a modification of the quadrature approach, which makes it more robust to approximation errors in tails:

$$p(y_i|x_i, D_{-i}, \theta, \phi) \approx \frac{\int p(y_i|f_i, \phi)\ p(f_i|D, \theta, \phi)\tilde{w}(f_i) df_i}{\int p(f_i|D, \theta, \phi)\tilde{w}(f_i) df_i}, \tag{52}$$

where

$$\tilde{w}(f_i) = \frac{1}{\max(p(y_i|f_i, \phi), c)}, \tag{53}$$

and c is a small positive constant. When $c = 0$, we get the original equation. When c is larger than the maximum value of $p(y_i|f_i, \phi)$, we get the posterior predictive density $p(y_i|D)$.

With larger values of $p(y_i|f_i, \phi)$ and c we avoid the possibility that the ratio explodes. In easy cases, where the numerator gets close to zero before c is used, we get a negligible bias. In difficult cases, we have a bias towards the full posterior predictive density.

In truncated importance sampling, the truncation level is based on the average raw weight size and the number of samples (see details in Ionides, 2008). Following this idea we choose

$$c^{-1} = c_0^{-1} \int_a^b \frac{p(f_i|D, \theta, \phi)}{p(y_i|f_i, \phi)} df_i.$$

By limiting the integral to interval (a, b), we avoid tail problems while capturing information about the average level of the weights. Based on experiments not reported here, we choose $c_0 = 10^{-4}$ and the interval (a, b) to extend 6 standard deviations from the mode of the marginal posterior in each direction. A case-specific c_0 could further improve results, but a fixed c_0 already shows the usefulness of the truncation. We refer to truncated weights quadrature LOO by TQ-LOO. In the experiments we show that TQ-LOO can provide more stable results than Q-LOO.

3.8 Widely Applicable Information Criterion

Watanabe (2010a,b) showed that the widely applicable information criterion (WAIC) is asymptotically equivalent to Bayesian LOO. Watanabe (2010a,b) provided two forms for WAIC, which we refer to as WAIC_G and WAIC_V following Vehtari and Ojanen (2012). WAIC was originally defined on the scale of mean negative log density, but for better cohesion within this paper we use the scale of mean log density. In the following discussion we drop the dependence on ϕ and θ and return to this point towards the end of the section. Both WAIC forms consist of the mean training log predictive density $\frac{1}{n}\sum_{i=1}^n \log p(y_i|D)$ and a second term to correct for its optimistic bias. These correction terms may be interpreted as the complexity of the model or the effective number of parameters in the model, but the interpretation does not always seem to be clear.

The correction term in WAIC_G is based on the difference between the training utility and Gibbs utility ($\frac{1}{n}\sum_{i=1}^n \int \log p(y_i|f_i)p(f_i|D)df_i$) giving

$$\text{WAIC}_G = \frac{1}{n}\sum_{i=1}^n \log p(y_i|D) - 2\sum_{i=1}^n \left[\log E_{f_i|D}[p(y_i|f_i)] - E_{f_i|D}[\log p(y_i|f_i)] \right], \tag{54}$$

where the Gibbs utility differs from the mean training log predictive density by the changed order of the logarithm and the expectation over the posterior.

The correction term in WAIC_V is based on the functional variance which describes the fluctuation of the posterior distribution:

$$\text{WAIC}_V = \frac{1}{n}\sum_{i=1}^n \log p(y_i|D) - \frac{1}{n}\sum_{i=1}^n \text{Var}_{f_i|D}[\log p(y_i|f_i)]. \tag{55}$$

Both of these criteria are easy to compute using Monte Carlo samples from the joint posterior $p(f|D)$, or marginal posterior approximation of $p(f_i|D)$ and quadrature integration.

Watanabe (2010b) used a Taylor series expansion to prove the asymptotic equivalence to Bayesian LOO with error term $O_p(n^{-2})$. To examine this relation we write the LOO log predictive density using condensed notation for (24)

$$-\frac{1}{n}\sum_{i=1}^n \log E_{f_i|D}[p(y_i|f_i)^{-1}]. \tag{56}$$

By defining a generating function of functional cumulants,

$$F(\alpha) = \frac{1}{n}\sum_{i=1}^{n} \log E_{f_i|D}[p(y_i|f_i)^\alpha], \tag{57}$$

and applying a Taylor expansion of $F(\alpha)$ around 0 with $\alpha = -1$, we obtain an expansion of the leave-one-out predictive density:

$$\text{LOO} = F'(0) - \frac{1}{2}F''(0) + \frac{1}{6}F^{(3)}(0) - \sum_{i=4}^{\infty} \frac{(-1)^i F^{(i)}(0)}{i!}. \tag{58}$$

From the definition of $F(\alpha)$ we get

$$\begin{aligned} F(0) &= 0 \\ F(1) &= \frac{1}{n}\sum_{i=1}^{n} \log E_{f_i|D}[p(y_i|f_i)] \\ F'(0) &= \frac{1}{n}\sum_{i=1}^{n} E_{f_i|D}[\log p(y_i|f_i)] \\ F''(0) &= \frac{1}{n}\sum_{i=1}^{n} \text{Var}_{f_i|D}[\log p(y_i|f_i)]. \end{aligned} \tag{59}$$

Furthermore, the expansion for the mean training log predictive density is

$$F(1) = F'(0) + \frac{1}{2}F''(0) + \frac{1}{6}F^{(3)}(0) + \sum_{i=4}^{\infty} \frac{F^{(i)}(0)}{i!}, \tag{60}$$

the expansion for WAIC_G is

$$\begin{aligned} \text{WAIC}_G(n) &= F(1) - 2[F(1) - F'(0)] = -F(1) + 2F'(0) \\ &= F'(0) - \frac{1}{2}F''(0) - \frac{1}{6}F^{(3)}(0) - \sum_{i=4}^{\infty} \frac{F^{(i)}(0)}{i!}, \end{aligned} \tag{61}$$

and the expansion for WAIC_V is

$$\begin{aligned} \text{WAIC}_V(n) &= F(1) - F''(0) \\ &= F'(0) - \frac{1}{2}F''(0) + \frac{1}{6}F^{(3)}(0) + \sum_{i=4}^{\infty} \frac{F^{(i)}(0)}{i!}. \end{aligned} \tag{62}$$

The first two terms of the expansion of WAIC_G and the first three terms of the expansion of WAIC_V match with the expansion of LOO. Based on the expansion we may assume that WAIC_V is the more accurate approximation for LOO.

Watanabe (2010b) shows that the error of WAIC_V is $O_p(n^{-2})$ and argues that asymptotically further terms have negligible contribution. However, the error can be significant in the case of finite n and weak prior information, as shown by Gelman et al. (2014), and for

hierarchical models, as demonstrated by Vehtari et al. (2016). For example, with Gaussian processes, if x_i is far from all other x_j, then f_i has a low correlation with any other f_j and the effective number of observations affecting the posterior of f_i is close to 1. In such cases, the higher order terms of the expansion are significant. The higher order terms of WAIC_V match the higher order terms of the mean training log predictive density and thus WAIC_V will be biased towards that. This is also evident from our experiments (see Section 4). It is not as clear what happens with WAIC_G, but experimentally the behavior is similar but with higher variance than with WAIC_V. The performance of both WAICs clearly also depend on the accuracy of the marginal approximation $q(f_i|D)$.

Instead of WAIC, we could directly compute a desired number of terms from the series expansion of LOO. In theory, we could approximate the exact result with a desired accuracy if enough higher order functional cumulants exist. This does not always work (e.g., if the posterior is Cauchy and the observation model is Gaussian), but it is true with a Gaussian prior on latent variables and a log-concave likelihood (An, 1998). In practice, the accuracy is limited by the computational precision of the higher cumulants, which is limited by the number of Monte Carlo samples or by the distributional approximation $q(f_i|D)$. If the cumulants are computed using $q(f_i|D)$ and quadrature, then the approximation based on Taylor series expansion converges eventually to Q-LOO (within numerical accuracy).

In the above equations we had dropped dependency on ϕ and θ. Like in other LOO-CV approximations, the parameter level can be handled using importance weighting. Alternatively we can handle the parameter level in full WAIC style by computing the cumulants of the marginal posteriors, where ϕ and θ have been integrated out, and using these cumulants to compute WAIC.

WAIC is related to the deviance information criterion (DIC). We do not review DIC here and instead refer to Gelman et al. (2014) for the reasons we prefer WAIC to DIC. Indeed, in our experiments not reported here, DIC had larger error than WAIC.

4. Results

Using several real data sets we present results illustrating the properties of the reviewed LOO-CV approximations. Table 3 lists the basic properties of four classification data sets (Ripley, Australian, Ionosphere, Sonar), one survival data set with censoring (Leukemia), and one data set for a Student's t regression (Boston). All data sets are available from the internet. Several classification data sets were selected as the posterior is likely to be skewed and there are often differences in performance between Laplace approximation and expectation propagation. The classification data sets have different numbers of covariates so we can investigate to what degree this affects the accuracy of the LOO-CV approximations. The leukemia survival data set was selected as we often analyze survival data with censoring. The Boston data set for a regression with a Student's t observation model was selected to illustrate the performance in the case of a non-log-concave likelihood, which may produce multimodal latent posterior. Similar results were obtained with other data sets not reported here.

For all data sets we fit Gaussian processes with constant, linear, and squared exponential covariance functions. When using the squared exponential covariance function, we use a separate length scale for each covariate except with the Ionosphere and Sonar data sets,

Data set	n	d	$\#(\phi, \theta)$	observation model
Ripley	250	2	5	probit
Australian	690	14	17	probit
Ionosphere	351	33	4	probit
Sonar	208	60	4	probit
Leukemia	1043	4	7	log-logistic with censoring
Boston	506	13	17	Student's t

Table 3: Summary of data sets and models in our examples.

where we use one common length scale. For the classification data sets we use a Bernoulli observation model with probit link. For the Leukemia data set we use a log-logistic model with censoring (as in Gelman et al., 2013, p. 511). For the Boston data set we use a Student's t observation model with $\nu = 4$ degrees of freedom. A fixed ν was chosen as the Laplace approximation (Vanhatalo et al., 2009) had occasional problems when integrating over an unknown ν. Robust-EP by Jylänki et al. (2011) works well also with ν unknown. All the experiments were done using GPstuff toolbox[1] (Vanhatalo et al., 2013). The Laplace method is implemented as described in Vanhatalo et al. (2010). The Laplace-EM method for Student's t model is implemented as described in Vanhatalo et al. (2009). Parallel EP for other data sets than Boston and parallel robust-EP for Student's t models are implemented as described in Jylänki et al. (2011). CCD is implemented as described in Vanhatalo et al. (2010). Markov chain Monte Carlo (MCMC) sampling is based on elliptical slice sampling for latent values (Murray et al., 2010) and surrogate slice sampling (Murray and Adams, 2010) for jointly sampling latent values and hyperparameters. The practical speed comparisons of the posterior and LOO approximation methods are shown in Appendix C.

Although in the review we described the estimation of the expected performance $\mathrm{LOO} = \frac{1}{n}\sum_{i=1}^{n} \log p(y_i|x_i, D_{-i})$, below we report $n \times \mathrm{LOO}$. For these data sets this puts the approximation errors for all sets on the same scale. This scale has two other interpretations. First, the difference between the sum training log predictive density and $n \times \mathrm{LOO}$ can be interpreted sometimes as the effective number of parameters measuring the model complexity (Vehtari and Ojanen, 2012; Gelman et al., 2014). Second, the significance of the difference between two models can be approximately calibrated if $n \times \mathrm{LOO}$ is interpreted as a pseudo log Bayes factor and if a similar calibration scale is used as for the Bayes factor (Vehtari and Ojanen, 2012). As a rule of thumb, based upon asymptotic theory and experience we would like the approximation error for nLOO to be smaller than 1. See the additional discussion of using Bayesian cross-validation in model selection in Vehtari and Lampinen (2002) and Vehtari and Ojanen (2012). We let $\mathrm{LOO}_i \equiv \log p(y_i|x_i, D_{-i})$ and $\widehat{\mathrm{LOO}}_i$ be the corresponding approximate quantity. In the tables we report a bias and deviation of individual terms as

$$\mathrm{Bias} = \sum_{i=1}^{n} (\widehat{\mathrm{LOO}}_i - \mathrm{LOO}_i) \tag{63}$$

$$\mathrm{Std}^2 = \sum_{i=1}^{n} (\widehat{\mathrm{LOO}}_i - \mathrm{LOO}_i - \mathrm{Bias})^2. \tag{64}$$

1. GPstuff is available at http://research.cs.aalto.fi/pml/software/gpstuff/

The acronyms used in the following are MCMC=Markov chain Monte Carlo, CCD=central composite design, MAP=Type II maximum a posteriori, PSIS=Pareto smoothed importance sampling, and those listed in Tables 1 and 2.

4.1 Exact LOO Comparison to MCMC

The ground truth exact LOO results were obtained by brute force computation of each $p(y_i|x_i, D_{-i})$ separately by leaving out the ith observation. We do that for each method: Laplace, EP and MCMC. MCMC serves as the golden standard for the posterior inference to which we compare Laplace and EP. We show results separately for estimating the predictive performance with and without a global correction (CM2/FACT). As discussed in Section 2.5, only the global corrections produce non-Gaussian predictive distributions for the latent variable $\tilde{f}$ at a new point $\tilde{x}$. Our main interest is in approximating $p(y_i|x_i, D_{-i})$, but we also show exact LOO results for the conditional $p(y_i|x_i, D_{-i}, \phi, \theta)$ with fixed parameters θ, ϕ, which were obtained by optimizing the marginal posterior $p(\theta, \phi|D)$ (type II MAP). In this case, LOO-CV is unbiased only conditionally as it does not take into account the effect of the fitting of the parameters θ, ϕ. However, it is useful to first evaluate the accuracy of approximations for $p(y_i|x_i, D_{-i}, \theta, \phi)$, as these can be used with integrated importance sampling (see Section 3.6) for hierarchical computation of $p(y_i|x_i, D_{-i})$.

The first part of Table 4 shows the exact LOO results with hyperparameters fixed to Laplace Type II MAP. LA has similar performance to MCMC for all data sets except Ionosphere and Sonar, for which LA is significantly inferior. LA-CM2 is able to improve the predictive performance for the Sonar data set to be similar with MCMC, and for the Ionosphere, the performance is even better than for MCMC.

The second part of Table 4 shows the exact LOO results with hyperparameters fixed to EP Type II MAP. EP has similar performance to MCMC for all data sets and EP-FACT is not able improve the performance. The small differences between MCMC results conditional on either LA-MAP or EP-MAP fixed hyperparameters are due to differences in the marginal likelihood approximations of LA and EP leading to different MAP estimates. However, this difference between LA-MAP and EP-MAP results is less interesting than differences with full integration.

The third part of Table 4 shows the exact LOO results with hyperparameters integrated with MCMC or CCD. LA+CCD is as good as MCMC for the Ripley, Australian and Leukemia data sets. LA-CM2+CCD improves the predictive performance for Ionosphere and Sonar. The performance of LA-CM2+CCD for Sonar is even better than MCMC and EP(-FACT)+CCD. LA-CM2+CCD failed to produce an answer in about 9% of leave-one-out rounds (the LA-CM2 method failing with some hyperparameter values) and thus no result is shown. EP is as good as MCMC for all data sets other than Boston and EP+FACT is not able to improve the performance at all.

Overall, when integrating over the hyperparameters, the difference between the predictive performance of LA and EP is small except for the Ionosphere and Sonar data sets. LA(-CM2)+CCD and EP(-FACT)+CCD have significantly worse predictive performance than MCMC for the Student's t regression with the Boston data. Since LA(-CM2) and EP(-FACT) performed as well as MCMC with fixed hyperparameters, the worse performance of CCD is due to error in the approximation of the marginal likelihood (see Jylänki et al., 2011) and

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
θ, ϕ fixed to LA-MAP						
MCMC	**-68**	**-217**	**-54**	**-68**	**-1627**	**-1097**
LA	-70 (0.6)	**-220** (2.8)	-72 (2.4)	-77 (1.7)	**-1626** (0.3)	**-1098** (3.0)
LA-CM2	**-68** (0.1)	**-217** (0.6)	-49 (0.7)	**-67** (0.9)	**-1626** (0.2)	**-1098** (2.8)
θ, ϕ fixed to EP-MAP						
MCMC	**-69**	**-211**	**-54**	**-64**	**-1626**	**-1098**
EP	**-68** (0.1)	**-211** (0.5)	**-54** (0.3)	**-64** (0.2)	**-1627** (0.3)	**-1095** (3.3)
EP-FACT	**-68** (0.1)	**-211** (0.4)	**-54** (0.3)	**-64** (0.2)	**-1627** (0.3)	**-1094** (3.2)
θ, ϕ integrated						
MCMC	**-70**	**-228**	**-56**	**-66**	**-1631**	**-1063**
LA+CCD	**-71** (0.5)	**-230** (2.7)	-74 (2.9)	-79 (1.4)	**-1631** (0.5)	-1116 (6.3)
LA-CM2+CCD	**-69** (0.2)	**-228** (1.2)	-51 (1.5)	-69 (1.6)	**-1631** (0.5)	NA (NA)
EP+CCD	**-70** (0.2)	**-226** (3.0)	**-57** (0.5)	**-65** (0.3)	**-1631** (0.5)	-1113 (5.1)
EP-FACT+CCD	**-70** (0.2)	**-226** (3.1)	**-57** (0.5)	**-65** (0.3)	**-1631** (0.5)	-1113 (5.1)

Table 4: Exact LOO (with brute force computation) using MCMC, Laplace (LA), Laplace with CM2 marginal corrections (LA-CM2), EP or EP with FACT marginal corrections (EP-FACT) for the latent values f, and fixed hyperparameters ϕ, θ (type II MAP) or integration over the hyperparameters with MCMC or CCD. The values in the parentheses are standard deviations of the pairwise differences from the corresponding MCMC result. Bolded values are not significantly different from the best accuracy in the corresponding category. NA indicates failed computation.

full MCMC is able to find better hyperparameters during the joint sampling of the latent values and hyperparameters.

4.2 Approximate LOO Comparison to Exact LOO – Fixed Hyperparameters

As discussed in Section 3.2, we compute LOO densities $p(y_i|x_i, D_{-i})$ hierarchically by first computing the conditional LOO densities $p(y_i|x_i, D_{-i}, \theta, \phi)$. As the accuracy of the full LOO densities depends crucially on the conditional LOO densities, we first analyze the LOO approximations conditional on fixed hyperparameters. The ground truth in this section are the LA, LA-CM2, EP, and EP-FACT results shown in Table 4.

Table 5 shows results when *the ground truth is exact LOO with fixed parameters and Laplace approximation without a global correction* (LA in Table 4). LA-LOO gives the best accuracy for all data sets by a significant margin. Quadrature LOO with Gaussian approximation of the latent marginals (Q-LOO-LA-G) produces bad results for the classification data sets and sometimes completely fails. The posterior marginals in the case of the Leukemia model are so close to Gaussian that Q-LOO-LA-G also provides a useful result. Truncated quadrature (TQ-LA-LOO-G) is more stable, but it cannot fix the whole problem. Using more accurate marginal approximation improves WAICs. WAIC_V with the LA-L marginal approximation gives useful results for the two simplest data sets.

Table 6 shows results when *the ground truth is exact LOO with fixed parameters and expectation propagation without a global correction* (EP in Table 4). EP-LOO gives the best accuracy for all data sets by a significant margin. Other results are similar to the Laplace case, that is, all methods except EP-LOO fail badly for several data sets. Only the Ripley

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
LA-LOO	**0.01** (0.02)	**0.1** (0.04)	**-0.2** (0.05)	**-0.2** (0.03)	**-0.0** (0.001)	**-2.7** (1.0)
Q-LOO-LA-G	-1379 (431)	NA (NA)	NA (NA)	-6732 (747)	0.38 (0.02)	79 (6)
TQ-LOO-LA-G	-1.2 (0.4)	-10 (1)	-5.6 (2.4)	-22 (3)	2.0 (0.3)	87 (6)
WAIC_G-LA-G	-1.5 (1.1)	11 (2)	-81 (10)	-11 (3)	1.2 (0.05)	101 (7)
WAIC_V-LA-G	-8.5 (6.6)	-9.4 (5.4)	-616 (91)	-75 (11)	0.40 (0.02)	81 (7)
WAIC_G-LA-L	0.8 (0.2)	21 (2)	23 (2)	26 (2)	0.8 (0.04)	54 (4)
WAIC_V-LA-L	0.3 (0.1)	6.9 (0.8)	16 (2)	17 (2)	0.02 (0.002)	15 (3)

Table 5: Bias and standard deviation when the ground truth is exact LOO with Laplace and fixed full posterior MAP hyperparameters (LA in Table 4). Bolded values have significantly smaller absolute value than the values from the other methods for the same data set. NA indicates that computation failed.

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
EP-LOO	**0.2** (0.1)	**1.6** (0.5)	**0.3** (0.4)	**-0.5** (0.1)	**-0.0** (0.003)	**-1.1** (0.9)
Q-LOO-EP-G	-352 (171)	NA (NA)	NA (NA)	NA (NA)	0.02 (0.003)	33 (3)
TQ-LOO-EP-G	-0.2 (0.2)	14 (8)	20 (4)	NA (NA)	1.7 (0.4)	44 (4)
WAIC_G-EP-G	0.7 (0.2)	59 (8)	**0.5** (3)	-42 (4)	0.8 (0.04)	76 (5)
WAIC_V-EP-G	-0.2 (0.4)	-4.3 (7)	-94 (11)	-804 (64)	0.03 (0.004)	37 (3)
WAIC_G-EP-L	0.7 (0.2)	81 (8)	23 (3)	48 (4)	0.8 (0.04)	81 (5)
WAIC_V-EP-L	0.4 (0.1)	54 (6)	17 (2)	42 (4)	0.02 (0.003)	26 (3)

Table 6: Bias and standard deviation when the ground truth is exact LOO with EP and fixed full posterior MAP hyperparameters (EP in Table 4). Bolded values have significantly smaller absolute values than the values from the other methods for the same data set. NA indicates that computation failed.

and Leukemia data sets are easy enough for most of the methods to produce useful accuracy.

Table 7 shows results when *the ground truth is exact LOO with fixed parameters and Laplace approximation with LA-CM2 global correction* (LA-CM2 in Table 4). Quadrature LOO with LA-CM2 approximation of the latent marginals (Q-LOO-LA-CM2) has the best accuracy for all data sets except for Boston, but the accuracy is satisfactory only for the Ripley and Leukemia data sets. Here LA-LOO has a negative bias as the global correction LA-CM2 can improve the marginal approximation and therefore also the expected performance estimated with exact LOO. The results for truncated quadrature (TQ-LOO-LA-CM2) are not reported in the table as with adaptive truncation it produced the same results as quadrature LOO (Q-LOO-LA-CM2). WAIC_V performs better than WAIC_G, but worse than Q-LOO-LA-CM2.

Table 8 shows results when *the ground truth is exact LOO with fixed parameters and expectation propagation with EP-FACT global correction* (EP-FACT in Table 4). EP-LOO provides significantly better accuracy for the Sonar and Leukemia data sets than the other methods. EP-LOO also gives the best accuracy for the other data sets, but not significantly better than quadrature with EP-FACT approximation of the latent marginals (Q-LOO-EP-FACT). In addition, for the Ripley data set all methods except WAIC_G provide good

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
LA-LOO	-1.4 (0.6)	-3.3 (3.3)	-23 (3)	-11 (2)	**0.00** (0.1)	**-2.6** (2.2)
Q-LOO-LA-CM2	**0.3** (0.1)	**3.1** (0.5)	**9.0** (1.8)	**7.4** (0.9)	**0.01** (0.0004)	11 (2)
WAIC_G-LA-CM2	1.0 (0.2)	25 (3)	16 (3)	27 (3)	0.8 (0.04)	61 (4)
WAIC_V-LA-CM2	0.5 (0.1)	11 (2)	13 (2)	20 (3)	0.02 (0.002)	22 (3)

Table 7: Bias and standard deviation when the ground truth is exact LOO with Laplace-CM2 and fixed full posterior MAP hyperparameters (LA+CM2 in Table 4). Bolded values have significantly smaller absolute values than the values from the other methods for the same data set.

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
EP-LOO	**0.13** (0.08)	**1.8** (0.6)	**0.1** (0.5)	**-0.64** (0.08)	**-0.0** (0.002)	**-1.4** (0.7)
Q-LOO-EP-FACT	**0.15** (0.04)	**3.2** (0.8)	**0.9** (0.3)	1.1 (0.3)	4.3 (1.3)	4.8 (1.1)
WAIC_G-EP-FACT	0.86 (0.22)	82 (8)	24 (3)	48 (4)	5.1 (1.3)	81 (5)
WAIC_V-EP-FACT	**0.31** (0.10)	54 (6)	17 (2)	42 (4)	4.4 (1.3)	27 (3)

Table 8: Bias and standard deviation when the ground truth is exact LOO with EP-FACT and fixed full posterior MAP hyperparameters (EP+FACT in Table 4). Bolded values have significantly smaller absolute values than the values from the other methods for the same data set.

results. The EP-LOO using the EP-L tilted distribution approximation is already good and the global correction does not change the result much. Small errors in the quadrature integration cumulate and Q-LOO-EP-FACT produces slightly worse results than EP-LOO.

4.3 LOO and WAIC with Varying Model Flexibility

Above we saw that the methods other than LA-LOO and EP-LOO had more difficulties with most of the data sets and especially with data sets with a large number of covariates. Figures 1–4 illustrate how the flexibility of the Gaussian process models affects the performance of the approximations. We took the models with MAP parameter values and re-ran the models and LOO tests, varying the length scales for all data sets except Boston (see later). With a smaller length scale, the GPs are more flexible and more non-linear. With a larger length scale GPs approach the linear model. We measure the flexibility by the difference between the mean training log predictive density and LOO, which can be interpreted as the degree to which the model has fit to the data or the relative effective number of parameters (p_{eff}/n). When the length scale gets smaller, there will be more such f_is that have a low correlation with any other f_j. In this case the full marginal posterior and LOO marginal posterior are likely to be more different and most LOO approximations become less accurate. This phenomenon will also occur more easily in the case of many covariates, because more data points will tend to be located at the corners of the data. Figures 1–4 show that LA-LOO and EP-LOO work well with different flexibilities. All the other methods have difficulties when the model flexibility increases and the marginal distribution and the cavity distribution are more different. If we look at the accuracy for each i, the methods other than LA-LOO and EP-LOO start to fail when the estimated $p_{\text{eff},i}$ is larger than 10%–20%. As a quick overall

rule of thumb, methods other than LA-LOO and EP-LOO start to fail when the relative effective number of parameters (p_{eff}/n) is larger than 2%–5%.

Figures 1–4 also show for Boston data how the degrees of freedom ν in the Student's t observation model affects the accuracy. When ν increases, the observation model is closer to Gaussian and the latent posterior is more likely to be unimodal. Although the latent posterior is easier to approximate with a Gaussian when ν is large, the posterior is less robust to influential observations ("outliers") and the error made by the methods other than LA-LOO and EP-LOO increases.

4.4 Approximate LOO Comparison to Exact LOO – Hierarchical Model

Next we examine the accuracy of hierarchical LOO approximation of $p(y_i|x_i, D_{-i})$ (see Section 3.2), where the conditional LOO densities $p(y_i|x_i, D_{-i}, \theta, \phi)$ are approximated with LA-LOO or EP-LOO, which we found performed best for conditional densities (see previous section).

Table 9 shows the results when *the ground truth is exact LOO with CCD used to integrate over the parameter posterior and the Laplace method is used to integrate over the latent values* (LA+CCD in Table 4). The Laplace approximation combined with type II MAP parameter estimates or CCD integration but no importance weighting has an error size related to the number of hyperparameters (θ, ϕ). The unweighted CCD or MAP gives a small error only if the number of parameters (θ, ϕ) is small. Importance weighting of CCD works well for all data sets except Australian and Boston. These data sets have more parameters (17) than the others (4-8), making the inference more difficult. The minimum relative effective sample sizes (Ripley=60%, Australian=16%, Ionosphere=59%, Sonar=70%, Leukemia=36%, Boston=0.3%) correctly indicate that importance weighting for Australian and Boston data sets is unreliable.

Table 10 shows the corresponding results when *the ground truth is exact LOO with CCD used to integrate over the parameter posterior and expectation propagation used to integrate over the latent values* (EP+CCD in Table 4). EP with the unweighted CCD or MAP gives a small error only if the number of parameters (θ, ϕ) is small. Importance weighting of CCD works well for all data sets except Australian and Boston. Again the minimum relative effective sample sizes (Ripley=60%, Australian=12%, Ionosphere=36%, Sonar=65%, Leukemia=35%, Boston=9%) correctly indicate that importance weighting for Australian and Boston is unreliable.

As CCD integration provided good results for exact LOO (Table 4), the larger errors of CCD+IS for the Australian and Boston data is not due to CCD itself failing, but importance weighting failing. As an additional check we sampled the hyperparameters with MCMC (6000 samples with slice sampling) and computed Pareto smoothed importance sampling estimates (MCMC+PSIS) shown also in Tables 9 and 10. Due to larger number of samples, the errors are slightly reduced, but still for the Australian and Boston data sets the errors are larger. The PSIS diagnostics (maximum of Pareto shape parameters $\hat{k}$ for Laplace: Ripley=0.4, Australian=1.2, Ionosphere=0.4, Sonar=0.4, Leukemia=0.2, Boston=1.3; for EP: Ripley=0.3, Australian=1.6, Ionosphere=0.3, Sonar=0.3, Leukemia=0.2, Boston=0.7) correctly indicate the problematic cases ($\hat{k} > 0.7$).

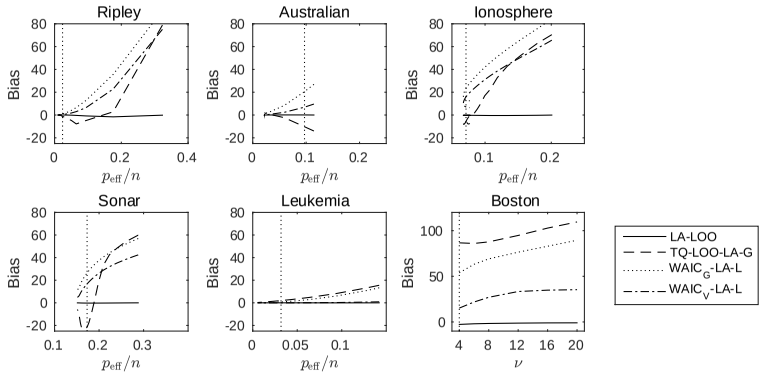


Figure 1: Bias when the ground truth is exact LOO with Laplace (LA in Table4) and varying flexibility of the model, or degrees of freedom in the Student's t model for the Boston data. Model flexibility was varied by rescaling the length scale(s) in the GP model. Model flexibility is measured by the relative effective number of parameters p_{eff}/n. The flexibility of the MAP model is shown with a vertical dashed line. For the Student's t model the vertical dashed line is at $\nu = 4$.

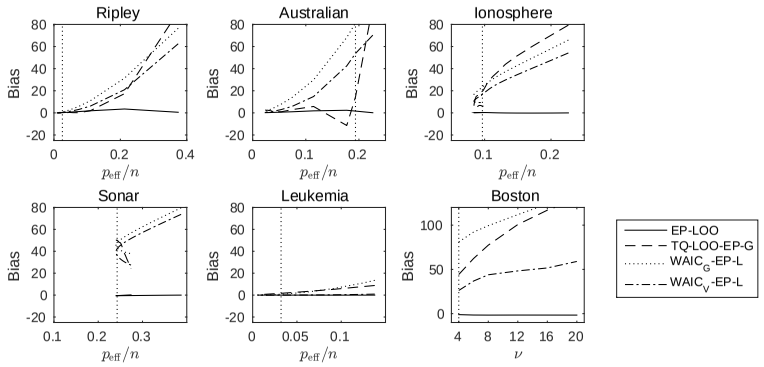


Figure 2: Bias when the ground truth is exact LOO with EP (EP in Table4) and varying flexibility of the model, or degrees of freedom in the Student's t model for the Boston data. Model flexibility was varied by rescaling the length scale(s) in the GP model. Model flexibility is measured by the relative effective number of parameters p_{eff}/n. The flexibility of the MAP model is shown with a vertical dashed line. For the Student's t the vertical dashed line is at $\nu = 4$.

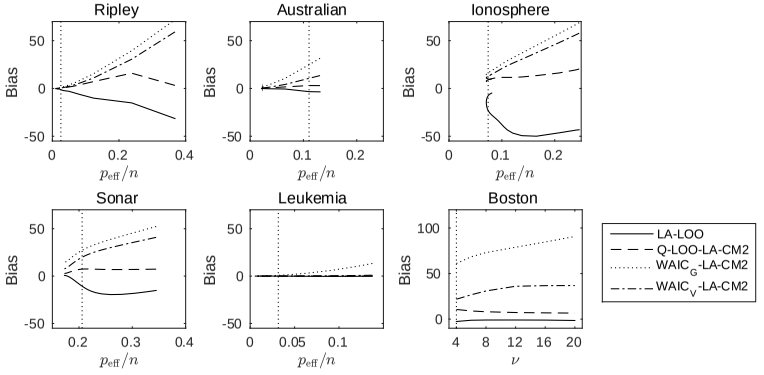


Figure 3: Bias when the ground truth is exact LOO with Laplace-CM2 (LA-CM2 in Table4) and varying flexibility of the model, or degrees of freedom in the Student's t model for the Boston data. Model flexibility was varied by rescaling the length scale(s) in the GP model. Model flexibility is measured by the relative effective number of parameters p_{eff}/n. The flexibility of the MAP model is shown with a vertical dashed line. For the Student's t the vertical dashed line is at $\nu = 4$.

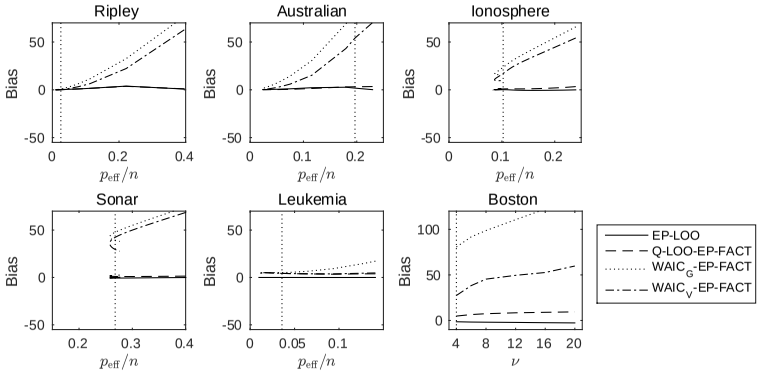


Figure 4: Bias when the ground truth is exact LOO with EP-FACT (EP-FACT in Table4) and varying flexibility of the model, or degrees of freedom in the Student's t model for the Boston data. Model flexibility was varied by rescaling the length scale(s) in the GP model. Model flexibility is measured by the relative effective number of parameters p_{eff}/n. The flexibility of the MAP model is shown with a vertical dashed line. For the Student's t the vertical dashed line is at $\nu = 4$.

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
LA-LOO+MCMC+PSIS	*0.08 (0.15)*	*-1.1 (2.3)*	*0.2 (0.1)*	*0.1 (0.2)*	*-0.04 (0.19)*	*-6.1 (2.4)*
LA-LOO+CCD+IS	**0.18** (0.10)	**3.4** (0.4)	**-0.1** (0.1)	**-0.13** (0.06)	**0.56** (0.05)	**-5.2** (5.0)
LA-LOO+CCD	0.8 (0.2)	7.2 (0.9)	0.6 (0.2)	0.5 (0.2)	4.8 (0.2)	17 (3)
LA-LOO+MAP	1.0 (0.2)	9.2 (1.8)	1.3 (0.2)	1.3 (0.3)	4.9 (0.6)	15 (3)

Table 9: Bias and standard deviation when the ground truth is exact LOO with Laplace and CCD (LA+CCD in Table 4). Bolded values have significantly smaller absolute error than the values from the other methods for the same data set.

Method	Ripley	Australian	Ionosphere	Sonar	Leukemia	Boston
EP-LOO+MCMC+PSIS	*0.38 (0.17)*	*-2.4 (3.4)*	*0.8 (0.5)*	*-0.23 (0.22)*	*-0.16 (0.23)*	*-0.9 (1.0)*
EP-LOO+CCD+IS	**0.42** (0.14)	**7.3** (1.4)	**0.8** (0.6)	**-0.24** (0.14)	**0.49** (0.04)	**2.2** (1.0)
EP-LOO+CCD	1.3 (0.4)	15 (2)	2.8 (1.3)	0.6 (0.3)	4.8 (0.2)	20 (2)
EP-LOO+MAP	1.4 (0.3)	17 (2)	2.8 (0.7)	0.9 (0.3)	4.9 (0.6)	17 (2)

Table 10: Bias and standard deviation when the ground truth is exact LOO with EP and CCD (EP+CCD in Table 4). Bolded values have significantly smaller absolute error than the values from the other methods for the same data set.

If the minimum relative effective sample size or PSIS diagnostics warn about potential problems, depending on the application it may be necessary to run, for example, k-fold cross-validation.

5. Discussion

We have shown that LA-LOO and EP-LOO provide fast and accurate conditional LOO results when the predictions at new points are made using the Gaussian latent value distribution. If the predictions at new points are made using non-Gaussian distributions obtained from the global correction, then quadrature LOO gives useful results, but it would be faster and more accurate to just use EP without the global correction. Both Laplace-LOO and EP-LOO can be combined with importance sampling or importance weighted CCD to get fast and accurate full Bayesian leave-one-out cross-validation results.

If other methods than LA-LOO or EP-LOO are used, we propose the following rule of thumb for diagnostics: The methods other than LA-LOO and EP-LOO start to fail when the relative effective number of parameters (p_{eff}/n) is larger than 2%–5%.

Here we have considered fully factorizing likelihoods, but the methods can be extended for use with likelihoods with grouped factorization, such as in multi-class classification, multi-output regression, and some hierarchical models with lowest level grouping. We assume that the accuracy using Laplace-LOO and EP-LOO would also be good in these cases.

In this paper, we have concentrated on how well exact LOO can be estimated with fast approximations. LOO is useful for estimating the predictive performance of of a model or in model comparison, but it should not be used to select a single model among a large number of models due to a selection induced bias as demonstrated by Piironen and Vehtari (2016).

Acknowledgments

We thank Jonah Gabry, Andrew Gelman, Alan Saul, Arno Solin, the editor, and two anonymous reviewers for helpful comments. We acknowledge the computational resources provided by Aalto Science-IT project.

A. Linear Response Laplace Leave-One-Out

Using linear response theory, used by Opper and Winther (2000) to prove LOO consistency of EP, we here derive approximative Laplace leave-one-out equations.

The idea is to express the posterior mode solution for the LOO problem in terms of the solution for the full problem. The computationally cheap solution can be obtained by making the assumption that the difference between these two solutions is small such that their difference may be treated as a second order Taylor expansion. We will give two different derivations of the result stated in Section 3.5; One is based on a second order expansion of the log likelihood and the second on a classical linear response argument.

In the expansion approach we make the approximation that when example i is removed we can treat the change in the mode for the remaining variables to second order. The log prior is already quadratic so it is only the non-linearity in the log likelihood terms $j \neq i$ that we expand to second order:

$$\log p(y_j|f_j,\phi) \approx \log p(y_j|\hat{f}_j,\phi) + (f_j - \hat{f}_j)\nabla_j \log p(y_j|f_j,\phi)|_{f_j=\hat{f}_j} - \frac{(f_j-\hat{f}_j)^2}{2\tilde{\Sigma}_j} , \qquad (65)$$

where $\tilde{\Sigma}_j$ is defined in Equation (14). We collect the first and second order contributions of the expansion to give the Gaussian type leave i out factors for the likelihood terms $j \neq i$. We recognize that *these approximate factors coincide with those introduced in the full Laplace approximation* in Equation (8). We can now write the approximate leave one out posterior as

$$q(f|D_{-i},\theta,\phi) \propto \prod_{j\neq i} \tilde{t}_j(f_j) p(f|X,\theta) \qquad (66)$$

and the marginal as

$$\begin{aligned} p(f_i|D_{-i},\theta,\phi) \approx q_{-i}(f_i) &\propto \frac{1}{\tilde{t}_i(f_i)} \int \prod_j \tilde{t}_j(f_j) p(f|X,\theta) df_{-i} \\ &\propto \frac{1}{\tilde{t}_i(f_i)} \int \mathrm{N}(f|\hat{f},\hat{\Sigma}) df_{-i} = \frac{\mathrm{N}(f_i|\hat{f}_i,\hat{\Sigma}_{ii})}{\tilde{t}_i(f_i)} . \end{aligned} \qquad (67)$$

This result shows that in a self-consistent second order approximation, where we take into account both the explicit removal of likelihood term i and the implicit effect on the remaining variables, the leave one out posterior is obtained simply by dividing by the Gaussian factor for i. Finally we complete the square and obtain the result in Equation (38).

Next we show how the same result can be obtained by a linear response argument. The equation for the mode is

$$K^{-1}\hat{f} = \hat{g} , \qquad (68)$$

where $\hat{g} = \nabla \log p(y|\hat{f}, \phi)$ is the vector of derivatives of the terms in the log likelihood (depending non-linearly on $\hat{f}$). Because this defines an equation for the mode, we only need to make a variation to first order in this case to recover the result we obtained above. When we remove likelihood term i the change in the mode can be written as

$$K^{-1}\delta\hat{f} = \delta\hat{g} \tag{69}$$

with the change in $\hat{g}$ to first order

$$\delta\hat{g} \approx -\tilde{\Sigma}^{-1}\delta\hat{f} + e_i\tilde{\Sigma}_i^{-1}\delta\hat{f}_i - e_i\hat{g}_i, \tag{70}$$

where we have used $\frac{\partial\hat{g}_i}{\partial\hat{f}_i} = -\hat{h}_i = -\tilde{\Sigma}_i^{-1}$ and e_i is a unit vector in the ith direction. The first two terms on the right hand side are the indirect change of the equation due to the removal of term i and the last is the direct contribution. We can now solve the linearized equation with respect to $\delta\hat{f}$ using the definition of the Laplace covariance $\Sigma = (K^{-1} + \tilde{\Sigma}^{-1})^{-1}$

$$\delta\hat{f} = \Sigma e_i(\tilde{\Sigma}_i^{-1}\delta\hat{f}_i - \hat{g}_i). \tag{71}$$

Specializing to $\delta\hat{f}_i$ we get

$$\delta\hat{f}_i = \Sigma_{ii}(\tilde{\Sigma}_i^{-1}\delta\hat{f}_i - \hat{g}_i) \tag{72}$$

which can be solved with respect to $\delta\hat{f}_i$ to give $\delta\hat{f}_i = -v_{-i}\hat{g}_i$. This is in agreement with the change in the mode equation (38) we found above. The variance term can be derived with a related linear response argument (Opper and Winther, 2000).

B. Computational Complexities

We summarize here the computational complexities of different methods in the paper. We first summarize the computational complexities of the Laplace method, expectation propagation, and marginal approximations used to obtain the full data posterior and its marginals. Then we summarize the additional computational complexities of the LOO methods. The related practical speed comparison results are shown in Appendix C.

The computational complexity for both the Laplace method and EP for GLVMs is dominated by matrix computations related to the covariance or precision matrix. We denote this basic cost as c_{basic}. For GLVMs with a full rank dense covariance matrix (such as Gaussian processes used in Section 4), c_{basic} scales with n^3. For reduced rank approximations in Gaussian processes such as FITC (Quiñonero-Candela and Rasmussen, 2005), c_{basic} scales with m^2n, where $m \ll n$ is the reduced rank (affecting the flexibility of GP). For sparse precision (in Gaussian Markov random field models (see, e.g., Rue et al., 2009)) or covariance matrices (in compact support covariance function GPs (see, e.g., Vanhatalo and Vehtari, 2008)), c_{basic} scales with n^2_{nonzeros}, where $n < n_{\text{nonzeros}} < n^2/2$ is the number of non-zeros in the precision, covariance, or Cholesky matrix (see more detailed analysis of sparse GLVMs in Cseke and Heskes, 2011).

For fixed ϕ and θ, the computation of the conditional posterior and marginal likelihoods scales for the Laplace method with $n^{\text{Newton}}_{\text{steps}} \times c_{\text{basic}}$ and for EP with $n^{\text{EP}}_{\text{steps}} \times (c_{\text{basic}} + n \times n_{\text{quad}})$, where n_{quad} is the number of potential quadrature evaluations to compute moments (for a probit classification model the moments can be computed in closed form).

Method	Additional computational complexity
EP-LOO	0
LA-LOO	$n \times n_{\text{quad}}$
(T)Q-LOO-LA/EP-G	$n \times n_{\text{quad}}$
WAIC$_{G/V}$-LA/EP-G/L	$n \times n_{\text{quad}}$
(T)Q-LOO-EP-FACT	$n^2 \times n_{\text{quad},1} \times n_{\text{quad},2}$
WAIC$_{G/V}$-EP-FACT	$n^2 \times n_{\text{quad},1} \times n_{\text{quad},2}$
(T)Q-LOO-LA-CM2	$n \times c_{\text{basic}} \times n_{\text{quad}}$
WAIC$_{G/V}$-LA-CM2	$n \times c_{\text{basic}} \times n_{\text{quad}}$
Exact brute force LOO EP	$n \times (n^{\text{EP}}_{\text{steps}} \times (c_{\text{basic}} + n \times n_{\text{quad}}))$
Exact brute force LOO Laplace	$n \times (n^{\text{Newton}}_{\text{steps}} \times c_{\text{basic}} + n_{\text{quad}})$

Table 11: Additional computational complexity of LOO methods for fixed θ and ϕ after obtaining the full posterior approximation with the Laplace method or EP.

After the last step of the Newton or EP algorithm, the additional computational complexities for different LOO methods are shown in Table 11. EP-LOO has zero additional complexity as the LOO log predictive density is computed as part of the algorithm. LA-LOO and methods using Gaussian marginals require n quadrature integrals to obtain log predictive densities and thus have negligible additional complexity. EP-FACT and LA-CM2 based methods have significantly larger additional complexity. The additional complexity of EP-FACT based methods scale with $n^2 \times n_{\text{quad},1} \times n_{\text{quad},2}$, where $n_{\text{quad},1}$ and $n_{\text{quad},2}$ refer to two different quadratures in the method. The additional complexity of LA-CM based methods scale with $n \times c_{\text{basic}} \times n_{\text{quad}} > n^3 \times n_{\text{quad}}$, which can be more than the complexity for the conditional posterior.

The computational complexity for the Type II MAP solution is the computational complexity of forming the conditional posterior given θ and ϕ times the number of marginal posterior evaluations in optimisation. The additional computation to obtain LOO after Type II MAP is the computation of LOO with fixed θ and ϕ.

The computational complexity for integration over the marginal posterior of θ and ϕ is the computational complexity of forming the conditional posterior given θ and ϕ times the number of marginal posterior evaluations in the (deterministic or stochastic) algorithm forming the posterior approximation. The additional computation for LOO requires the computation of LOO with fixed θ and ϕ for each point in the final marginal posterior approximation and computation of importance weights which has a negligible additional cost.

C. Practical Speed Comparison

To further give some idea of the practical speed differences between the different algorithm implementations we show examples of computation times for computing the marginal likelihood and LOO given fixed θ and ϕ. The speed comparisons were run with a laptop (Intel Core i5-4300U CPU @ 1.90GHz x 4 + 8GB memory). As one of the reviewers was interested in comparison to global Gaussian variational method, we have included it in this speed comparison, verifying the previous results that it is much slower than EP (Nickisch and

				GPstuff		GPML		
Data set	n	d	lik	LA	EP[1]	LA	EP[2]	KL
Ripley	250	2	probit	0.02	0.04	0.06	0.71	155
Australian	690	14	probit	0.13	0.40	0.26	10	704
Ionosphere	351	33	probit	0.05	0.13	0.08	1.7	516
Sonar	208	60	probit	0.03	0.04	0.06	0.47	233
Leukemia	1043	4	log-logistic w. cens.	0.18	3.5	NA[3]	NA[3]	NA[3]
Boston	506	13	Student's t	1.1[4]	1.1	NA[5]	NA[6]	39[7]

Table 12: Time (in seconds) to compute the posterior and marginal likelihood with fixed hyperparameters. [1]GPstuff uses parallel EP (van Gerven et al., 2009) except for Student's t parallel robust-EP (Jylänki et al., 2011) is used. [2]GPML uses random order sequential EP (Rasmussen and Williams, 2006). [3]GPML does not have log-logistic model with censoring. [4]For Student's t Laplace-EM method (Vanhatalo et al., 2009) was used. [5]The GPML's Laplace inference algorithm did run without errors, but the results were really bad (difference in LOO-LPD 1.4e4). [6]GPML does not support EP for non-log-concave likelihoods. [7]For Student's t the performance of global Gaussian variational (KL) method was much worse than the performance of Laplace-EM and EP (difference in LOO-LPD 147)

Rasmussen, 2008). As GPstuff does not have the global Gaussian variational approximation, we use the KL method in GPML toolbox (Rasmussen and Nickisch, 2010) to compute that. To take into account potential general speed differences between GPstuff and GPML we also timed GPML Laplace and EP methods. Computations were timed several times so that caching of the previous computations were not used.

Table 12 shows the time to compute the latent posterior and marginal likelihood with fixed hyperparameters. In optimization (or gradient based MCMC), the computation of gradients would have additional computational cost. When hyperparameters are integrated out, the approximative computation time is these multiplied by the number of unique parameter values evaluated when obtaining the marginal posterior samples (there are additional overheads and potential speed-ups). For probit, where the moments required in EP can be computed analytically, GPstuff-EP is about 1.5–5 times slower than GPstuff-LA. For log-logistic with censoring GPstuff-EP is about 18 times slower due to slow quadrature based moment computations (which could be made faster). For the Student's t model GPstuff-LA and GPstuff-EP have similar performance, as the robust Laplace-EM method by Vanhatalo et al. (2009) is slower than basic Laplace approximation. GPstuff-LA and GPML-LA have quite similar speed, GPstuff being slightly faster. GPstuff-EP is 10-25 times faster than GPML-EP, which is probably due to using parallel updates and better vectorization allowed by parallel updates. GPstuff has robust-EP implementation which also works for non-log-concave likelihoods such as Student's t. Although KL has the same $O(n^3)$ computational scaling as EP, its computational overhead makes GPML-KL 70-500 times slower than GPML-EP.

Table 13 shows time to compute the LOO for fixed parameters after the full posterior has been computed (see Table 12). When hyperparameters are integrated out, the approximative computation time is these multiplied by the number of parameter samples approximating

Data set	LA-LOO	EP-LOO	Q-LOO-LA-CM2	Q-LOO-EP-FACT	Exact LOO Laplace	Exact LOO EP
Ripley	0.01	0.005	30	3.7	6.3	13
Australian	0.11	0.005	672	15	90	323
Ionosphere	0.03	0.005	91	6.1	19	47
Sonar	0.02	0.005	19	3.0	7.2	12
Leukemia	0.89	0.005	2547	11876	198	3762
Boston	0.47	0.005	237	7.5	583	587

Table 13: Time (in seconds) to compute LOO for fixed parameters after the full posterior has been computed. Here Q-LOO computations also include the time to compute the marginal corrections (LA-CM2 and EP-FACT).

the marginal posterior. There is no added computational cost of going from EP to EP-LOO and the time is spent retrieving the stored result. The computational cost of LA-LOO is computing cavity distributions and one quadrature. Here Q-LOO computations also include the time to compute the marginal corrections (LA-CM2 and EP-FACT), which make them much slower.

References

Mark Yuying An. Logconcavity versus logconvexity: a complete characterization. *Journal of Economic Theory*, 80(2):350–369, 1998.

José M. Bernardo and Adrian F. M. Smith. *Bayesian Theory*. John Wiley & Sons, 1994.

Edward Challis and David Barber. Gaussian Kullback-Leibler approximate inference. *Journal of Machine Learning Research*, 14:2239–2286, 2013.

Botond Cseke and Tom Heskes. Approximate marginals in latent Gaussian models. *Journal of Machine Learning Research*, 12:417–454, 2 2011. ISSN 1532-4435.

Ilenia Epifani, Steven N MacEachern, and Mario Peruggia. Case-deletion importance sampling estimators: Central limit theorems and related results. *Electronic Journal of Statistics*, 2:774–806, 2008.

Seymour Geisser and William F. Eddy. A predictive approach to model selection. *Journal of the American Statistical Association*, 74(365):153–160, March 1979.

Alan E. Gelfand. Model determination using sampling-based methods. In W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors, *Markov Chain Monte Carlo in Practice*, pages 145–162. Chapman & Hall, 1996.

Alan E. Gelfand, D. K. Dey, and H. Chang. Model determination using predictive distributions with implementation via sampling-based methods (with discussion). In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 4*, pages 147–167. Oxford University Press, 1992.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, third edition, 2013.

Andrew Gelman, Jessica Hwang, and Aki Vehtari. Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, 24(6):997–1016, 2014.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of American Statistical Association*, 102:359–379, 2007.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.

Leonhard Held, Birgit Schrödle, and Håvard Rue. Posterior and cross-validatory predictive checks: A comparison of MCMC and INLA. In Thomas Kneib and Gerhard Tutz, editors, *Statistical Modelling and Regression Structures*, pages 91–110. Springer, 2010.

Daniel Hernáandez-Lobato, José M. Hernández-Lobato, and Alberto Suárez. Expectation propagation for microarray data classification. *Pattern Recognition Letters*, 31(12):1618–1626, 2010.

José M. Hernández-Lobato, Tjeerd Dijkstra, and Tom Heskes. Regulator discovery from gene expression time series of malaria parasites: a hierarchical approach. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 649–656, 2008.

Edward L. Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17(2):295–311, 2008.

Pasi Jylänki, Jarno Vanhatalo, and Aki Vehtari. Robust Gaussian process regression with a Student-t likelihood. *Journal of Machine Learning Research*, 12:3227–3257, 2011.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *The International Conference on Learning Representations (ICLR)*, 2014.

Augustine Kong, Jun S. Liu, and Wing Hung Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

Siem Jan Koopman, Neil Shephard, and Drew Creal. Testing the assumptions behind importance sampling. *Journal of Econometrics*, 149(1):2–11, 2009.

Longhai Li, Shi Qiu, Bei Zhang, and Cindy X. Feng. Approximating cross-validatory predictive evaluation in Bayesian latent variables models with integrated IS and WAIC. *arXiv preprint arXiv:1404.2918*, 2014.

Finn Lindgren and Håvard Rue. Bayesian spatial modelling with R-INLA. *Journal of Statistical Software*, 63(19), 2015.

E. C. Marshall and D. J. Spiegelhalter. Approximate cross-validatory predictive checks in disease mapping models. *Statistics in Medicine*, 22(10):1649–1660, 2003.

Thiago G Martins, Daniel Simpson, Finn Lindgren, and Håvard Rue. Bayesian computing with INLA: new features. *Computational Statistics & Data Analysis*, 67:68–83, 2013.

Thomas Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

Iain Murray and Ryan Prescott Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In J. Lafferty, C. K. I. Williams, R. Zemel, J. Shawe-Taylor, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1732–1740, 2010.

Iain Murray, Ryan Prescott Adams, and David J.C. MacKay. Elliptical slice sampling. *JMLR Workshop and Conference Proceedings: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 9:541–548, 2010.

Hannes Nickisch and Carl Edward Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9:2035–2078, October 2008.

Manfred Opper and Ole Winther. Gaussian processes for classification: Mean-field algorithms. *Neural Computation*, 12(11):2655–2684, 2000.

Mario Peruggia. On the variability of case-deletion importance sampling weights in the Bayesian linear model. *Journal of the American Statistical Association*, 92(437):199–207, 1997.

Juho Piironen and Aki Vehtari. Comparison of Bayesian predictive methods for model selection. *Statistics and Computing*, 2016. Online first: DOI:10.1007/s11222-016-9649-y.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(3): 1939–1959, 2005.

Rajesh Ranganath, Sean Gerrish, and David M Blei. Black box variational inference. *JMLR Workshop and Conference Proceedings: Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, 33:814–822, 2014.

Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (GPML) toolbox. *The Journal of Machine Learning Research*, 11:3011–3015, 2010.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *JMLR Workshop and Conference Proceedings: Proceedings of The 31st International Conference on Machine Learning*, 32: 1278–1286, 2014.

Jaakko Riihimäki and Aki Vehtari. Laplace approximation for logistic Gaussian process density estimation and regression. *Bayesian analysis*, 9(2):425–448, 2014.

Jaakko Riihimäki, Pasi Jylänki, and Aki Vehtari. Nested expectation propagation for Gaussian process classification with a multinomial probit likelihood. *Journal of Machine Learning Research*, 14:75–109, 2013.

Håvard Rue, Sara Martino, and Nicolas Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal statistical Society B*, 71(2):1–35, 2009.

Matthias Seeger. Bayesian inference and optimal design for the sparse linear model. *Journal of Machine Learning Research*, 9:759–813, 2008.

S. Sundararajan and S. S. Keerthi. Predictive approaches for choosing hyperparameters in Gaussian processes. *Neural Computation*, 13(5):1103–1118, May 2001.

Luke Tierney and Joseph B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

Ville Tolvanen, Pasi Jylänki, and Aki Vehtari. Expectation propagation for nonstationary heteroscedastic Gaussian process regression. In *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2014. DOI:10.1109/MLSP.2014.6958906.

Marcel van Gerven, Botond Cseke, Robert Oostenveld, and Tom Heskes. Bayesian source localization with the multivariate Laplace prior. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 1901–1909, 2009.

Jarno Vanhatalo and Aki Vehtari. Modelling local and global phenomena with sparse Gaussian processes. In David A. McAllester and Petri Myllymäki, editors, *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pages 571–578, 2008.

Jarno Vanhatalo and Aki Vehtari. Speeding up the binary Gaussian process classification. In Peter Grünwald and Peter Spirtes, editors, *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 1–9, 2010.

Jarno Vanhatalo, Pasi Jylänki, and Aki Vehtari. Gaussian process regression with Student-t likelihood. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1910–1918, 2009.

Jarno Vanhatalo, Ville Pietiläinen, and Aki Vehtari. Approximate inference for disease mapping with sparse Gaussian processes. *Statistics in Medicine*, 29(15):1580–1607, 2010.

Jarno Vanhatalo, Jaakko Riihimäki, Jouni Hartikainen, Pasi Jylänki, Ville Tolvanen, and Aki Vehtari. GPstuff: Bayesian modeling with Gaussian processes. *Journal of Machine Learning Research*, 14:1175–1179, 2013.

Aki Vehtari. *Bayesian Model Assessment and Selection Using Expected Utilities*. PhD thesis, Helsinki University of Technology, 2001.

Aki Vehtari and Andrew Gelman. Pareto smoothed importance sampling. *arXiv preprint arXiv:1507.02646*, 2015.

Aki Vehtari and Jouko Lampinen. Bayesian model assessment and comparison using cross-validation predictive densities. *Neural Computation*, 14(10):2439–2468, 2002.

Aki Vehtari and Janne Ojanen. A survey of Bayesian predictive methods for model assessment, selection and comparison. *Statistics Surveys*, 6:142–228, 2012.

Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *arXiv preprint arXiv:1507.04544*, 2016.

Sumio Watanabe. Equations of states in singular statistical estimation. *Neural Networks*, 23 (1):20–34, 2010a.

Sumio Watanabe. Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of Machine Learning Research*, 11:3571–3594, 2010b.

Jin Zhang and Michael A Stephens. A new and efficient estimation method for the generalized Pareto distribution. *Technometrics*, 51(3):316–325, 2009.

Journal of Machine Learning Research 17 (2016) 1-32 Submitted 2/15; Revised 4/16; Published 8/16

ϵ-PAL: An Active Learning Approach to the Multi-Objective Optimization Problem

Marcela Zuluaga ZULUAGA@INF.ETHZ.CH
Department of Computer Science
ETH Zurich
Zurich, Switzerland

Andreas Krause ANDREAS.KRAUSE@INF.ETHZ.CH
Department of Computer Science
ETH Zurich
Zurich, Switzerland

Markus Püschel PUESCHEL@INF.ETHZ.CH
Department of Computer Science
ETH Zurich
Zurich, Switzerland

Editor: Kevin Murphy

Abstract

In many fields one encounters the challenge of identifying out of a pool of possible designs those that simultaneously optimize multiple objectives. In many applications an exhaustive search for the Pareto-optimal set is infeasible. To address this challenge, we propose the ϵ-*Pareto Active Learning* (ϵ-PAL) algorithm which adaptively samples the design space to predict a set of Pareto-optimal solutions that cover the true Pareto front of the design space with some granularity regulated by a parameter ϵ. Key features of ϵ-PAL include (1) modeling the objectives as draws from a Gaussian process distribution to capture structure and accommodate noisy evaluation; (2) a method to carefully choose the next design to evaluate to maximize progress; and (3) the ability to control prediction accuracy and sampling cost. We provide theoretical bounds on ϵ-PAL's sampling cost required to achieve a desired accuracy. Further, we perform an experimental evaluation on three real-world data sets that demonstrate ϵ-PAL's effectiveness; in comparison to the state-of-the-art active learning algorithm PAL, ϵ-PAL reduces the amount of computations and the number of samples from the design space required to meet the user's desired level of accuracy. In addition, we show that ϵ-PAL improves significantly over a state-of-the-art multi-objective optimization method, saving in most cases 30% to 70% evaluations to achieve the same accuracy.

Keywords: multi-objective optimization, active learning, pareto optimality, Bayesian optimization, design space exploration

1. Introduction

A fundamental challenge in many problems in engineering and other domains is to find the right balance amongst several objectives. As a concrete example, in hardware design, one often has to choose between different candidate designs that trade multiple objectives

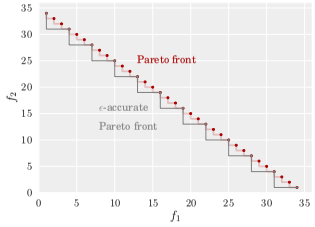


Figure 1: Assume that two objective functions are to be maximized simultaneously. This figure shows an example of the true Pareto front of a design space contrasted with an ϵ-accurate Pareto front, with $\epsilon = (3, 3)$.

such as energy consumption, throughput, or chip area. Usually there is not a single design that excels in all objectives, and therefore one may be interested in identifying all (Pareto-)optimal designs. Furthermore, often in these domains, evaluating the objective functions is expensive and noisy. In hardware design, for example, synthesis of only one design can take hours or even days. The fundamental problem addressed in this paper is how to predict a Pareto-optimal set at low cost, i.e., by evaluating as few designs as possible.

In this paper we propose a solution for finite design spaces that we call the ϵ-*Pareto Active Learning* (ϵ-PAL) algorithm. The parameter ϵ allows users to control the accuracy of the prediction produced by the algorithm. This accuracy is defined in terms of the density or granularity of the estimated Pareto front that is returned. Decreasing granularity means the algorithm can discard more points along its execution. Fewer Pareto points are returned, but spread to offer a wide range of trade-offs in the objective space.

More specifically, the granularity is controlled by a vector ϵ, containing one value per objective, which is given in the same units as the corresponding objective function. It specifies that for the user a difference of ϵ in the objective space is negligible. Accordingly, an ϵ-accurate Pareto front might have fewer points than the true Pareto front. Fig. 1 visualizes this idea for two objective functions that are to be maximized simultaneously. The true Pareto-front is shown in red, and an ϵ-accurate Pareto front, which contains fewer points, is shown in grey. While the true Pareto front of a design space is unique, there can be many ϵ-accurate Pareto fronts; finding one of them efficiently is our goal.

ϵ-PAL has several key features. In the spirit of Bayesian optimization, it captures domain knowledge about the regularity of the design space by using Gaussian process (GP) models to predict objective values for designs that have not been evaluated yet. Further, it uses the predictive uncertainty associated with these nonparametric models in order to guide the iterative sampling. Specifically, ϵ-PAL's sampling strategy aims to maximize progress on designs that are likely to be Pareto-optimal. ϵ-PAL iteratively discards points that are either redundant or suboptimal, and it terminates when no more points can be removed in order to guarantee that, with high probability, the remaining points define an ϵ-accurate Pareto set of the given design space.

A main contribution of this paper is the theoretical performance analysis of ϵ-PAL, that provides bounds on the sampling cost required to achieve a desired accuracy. These bounds depend on the parameter ϵ and on the characteristics of the covariance function used for the GP models. Finally, we carry out an extensive empirical evaluation, where we demonstrate ϵ-PAL's effectiveness on several real-world multi-objective optimization problems. Two cases are from different applications in the domain of hardware design, in which it is very expensive to run low level synthesis to obtain the exact cost and performance of a single design (Zuluaga et al., 2012b; Almer et al., 2011). The third application is from software optimization, where different compilation settings are evaluated for performance and memory footprint size (Siegmund et al., 2012).

We compare the performance of ϵ-PAL against PAL, a predecessor of ϵ-PAL proposed by Zuluaga et al. (2013), and with another state-of-the-art multi-objective optimization method called ParEGO, proposed by Knowles (2006). Across all data sets and almost all desired accuracies, ϵ-PAL outperforms ParEGO, requiring in most cases 30% to 70% less function evaluations. In comparison to PAL, our experiments show that ϵ-PAL reduces the runtime by one to two orders of magnitude, while also reducing the number of evaluations and offering more flexibility in the desired error.

1.1 Main Contributions

Our main contributions are summarized as follows.

- We propose ϵ-PAL, an active learning approach towards identifying a set of optimal solutions in a given multi-objective optimization problem in which function evaluations are expensive. This includes sampling strategy and stopping criteria. ϵ-PAL allows users to specify the desired level of granularity through the parameter ϵ, and guarantees that with high probability an ϵ-accurate Pareto front is returned.
- We theoretically analyze ϵ-PAL when the design space is a finite set, and the objective functions satisfy regularity assumptions as specified via a positive definite kernel. We provide bounds on the number of iterations required by the algorithm to achieve a desired target accuracy.
- In comparison to the state-of-the-art algorithm PAL, ϵ-PAL reduces the asymptotic complexity of the number of computations performed in each iteration, making it more suitable for larger design spaces.
- We perform an extensive experimental evaluation to demonstrate ϵ-PAL's effectiveness and superiority over PAL and ParEGO on three real-world multi-objective optimization problems.

1.2 Organization

In Section 2, we introduce background concepts such as multi-objective optimization, Pareto optimality, ϵ-accurate Pareto optimality, and Gaussian processes. In addition, we formally define the problem tackled in this paper. Section 3 presents the proposed algorithm ϵ-PAL, explains the stages that take place during its execution, and analyzes run time. In

Section 4, we present the theoretical analysis of the algorithm and provide upper bounds for the number of samples needed before ϵ-PAL terminates, when the target functions have bounded RKHS norm. Section 5 discusses some implementation issues that might arise in practice when using our algorithm. Section 6 reviews related work in the areas of multi-objective optimization, Bayesian optimization and evolutionary algorithms. Section 7 shows the effectiveness of ϵ-PAL in comparison with state-of-the-art alternative algorithms, using three data sets obtained from real applications. In the Appendix, we prove our main Theorem, which is stated in Section 4.

2. Background and Problem Statement

In this section, we review multi-objective optimization and Pareto optimality, and introduce the terminology and notation used in the rest of the paper. At the end of the section, we formally define the problem addresses by ϵ-PAL.

2.1 Multi-Objective Optimization

We consider a multi-objective optimization problem over a finite set E (called the *design space*). This means that we wish to simultaneously optimize m objective functions $f_1, \ldots, f_m : E \to \mathbb{R}$. In our analysis, we assume that $f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ are to be maximized, however our results straightforwardly generalize to a minimization or a combined minimization/maximization problem. We use the notation $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))$ to refer to the vector of all objectives evaluated on the input $\boldsymbol{x}$. The *objective space* is the image $\boldsymbol{f}(E) \subset \mathbb{R}^m$.[1]

2.2 Pareto Optimality

The goal in multi-objective optimization is to identify[2] the *Pareto set* of E. Formally, we consider the canonical partial order in $\mathbb{R}^m$: $\boldsymbol{y} \succeq \boldsymbol{y}'$ iff $y_i \geq y'_i$, $1 \leq i \leq m$, and define the induced relation on E: $\boldsymbol{x} \succeq \boldsymbol{x}'$ iff $\boldsymbol{f}(\boldsymbol{x}) \succeq \boldsymbol{f}(\boldsymbol{x}')$. We say that $\boldsymbol{x}$ dominates $\boldsymbol{x}'$ in this case. Note that $\succeq$ on E is not a partial order, but only a preorder, since it lacks antisymmetry (i.e., two different designs can have the same objectives).

Definition 1 (Pareto set) *The Pareto set $\Pi(\boldsymbol{f}(E)))$ in the objective space $\boldsymbol{f}(E) \subset \mathbb{R}^m$ is, as usual, the set of maximal points. Further, we call any set $\Pi(E) \subseteq E$ a Pareto set of E if it satisfies*

$$\boldsymbol{f}(\Pi(E)) = \Pi(\boldsymbol{f}(E))).$$

In words, $\Pi(E)$ is any set of designs that yields all optimal objectives. It is not unique since $\succeq$ on E is not antisymmetric.

1. Scalars and functions that return scalars are written unbolded; tuples and vectors are boldfaced.
2. Note that one may also approach multi-objective optimization via scalarization, see, e.g., Roijers et al. (2013). In our approach, we focus on retrieving an approximate Pareto-frontier, so that we do not have to commit to a particular family of scalarization functions.

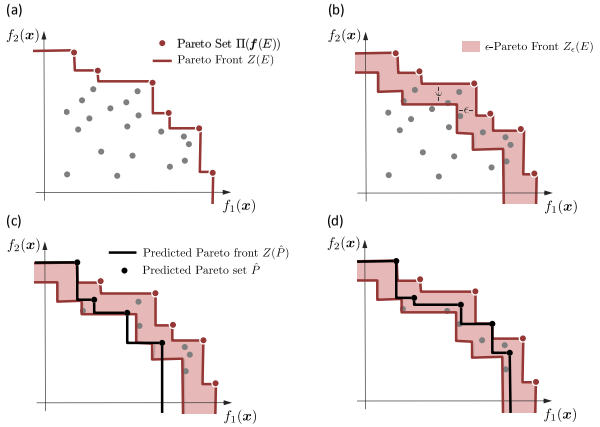


Figure 2: (a) Example of Pareto set and Pareto front for $m = 2$. (b) Example of an $\boldsymbol{\epsilon}$-Pareto front for $m = 2$. (c) Example of a predicted Pareto set that is not ϵ-accurate. (d) Example of a predicted Pareto set that is ϵ-accurate.

Definition 2 (Pareto front) *We define the Pareto front $Z(E)$ as the set of points in $\mathbb{R}^m$ that constitutes the surface of the space dominated by the Pareto set $\Pi(\boldsymbol{f}(E))$. Formally,*

$$Z(E) = \partial\{\boldsymbol{y} \in \mathbb{R}^m : \text{ there is an } \boldsymbol{x} \in E \text{ with } \boldsymbol{f}(\boldsymbol{x}) \succeq \boldsymbol{y})\}. \tag{1}$$

Hereby, the operator ∂ applied to a set Y denotes the boundary of Y.

Fig. 2(a) visualizes the concepts of Pareto set and Pareto front for $m = 2$.

2.3 $\boldsymbol{\epsilon}$-Pareto Optimality

We now relax the relation $\succeq$ in $\mathbb{R}^m$ and E for the purpose of our algorithm by adding a small tolerance. Consider a vector $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_d)$, with $\epsilon_i \geq 0$, $1 \leq i \leq m$. We say that a point $\boldsymbol{y} \in \mathbb{R}^m$ $\boldsymbol{\epsilon}$-dominates $\boldsymbol{y}'$, written as $\boldsymbol{y} \succeq_\epsilon \boldsymbol{y}'$, if $\boldsymbol{y} + \boldsymbol{\epsilon} \succeq \boldsymbol{y}'$. As before, we pull the relation to E: $\boldsymbol{x} \succeq_\epsilon \boldsymbol{x}'$ iff $\boldsymbol{f}(\boldsymbol{x}) \succeq_\epsilon \boldsymbol{f}(\boldsymbol{x}')$. Note that this relation is neither a partial order nor a preorder: antiysmmetry and transitivity do not hold.

We use this relation next to define an appropriate notion of $\boldsymbol{\epsilon}$-accurate Pareto set. To do so we first need two auxiliary definitions.

Definition 3 ($\boldsymbol{\epsilon}$-Pareto front) *We define the $\boldsymbol{\epsilon}$-Pareto front $Z_\epsilon(E)$ of E as the set of points between $Z(E)$ and $Z(E) - \boldsymbol{\epsilon}$, including the boundaries. Formally,*

$$Z_\epsilon(E) = \{\boldsymbol{y} \in \mathbb{R}^m : \boldsymbol{y}' \succeq \boldsymbol{y} \text{ for some } \boldsymbol{y}' \in Z(E) \text{ and } \boldsymbol{y} \succeq_\epsilon \boldsymbol{y}'' \text{ for some } \boldsymbol{y}'' \in Z(E)\}$$

Fig. 2(b) shows the $\boldsymbol{\epsilon}$-Pareto front associated with Fig. 2(a).

Definition 4 ($\boldsymbol{\epsilon}$-Pareto front covering) *We say that a nonempty set $C \subseteq Z_\epsilon(E)$ covers $Z_\epsilon(E)$, if for every point $\boldsymbol{y} \in Z_\epsilon(E)$ there is at least one point $\boldsymbol{y}' \in C$ s.t. $\boldsymbol{y}' \succeq_\epsilon \boldsymbol{y}$.*

In words, Definition 4 requires that the Pareto front spanned by $\Pi(C)$ is contained in $Z_\epsilon(E)$.

Definition 5 ($\boldsymbol{\epsilon}$-Accurate Pareto set) *We call a set of points $\Pi_\epsilon(E) \subseteq E$ an $\boldsymbol{\epsilon}$-accurate Pareto set of E, if $\boldsymbol{f}(\Pi_\epsilon(E))$ covers $Z_\epsilon(E)$.*

In words, for an $\boldsymbol{\epsilon}$-accurate Pareto set, the associated front is contained in the $\boldsymbol{\epsilon}$-Pareto front $Z_\epsilon(E)$ and it contains no suboptimal designs. As examples, the set with objectives $\hat{P}$ in Fig. 2(c) does not satisfy this property, but in Fig. 2(d) it does. Note that the $\boldsymbol{\epsilon}$-accurate Pareto set may have fewer points than the actual Pareto set.

An $\boldsymbol{\epsilon}$-accurate Pareto set $\Pi_\epsilon(E)$ is a natural approximate substitute of the original set E. Having access to it, one can rest assured that for any point on the Pareto front $Z(E)$ associated with E, there is some element $\boldsymbol{x} \in \Pi_\epsilon(E)$ which is at most ϵ worse according to all of the objectives. The high level goal of our algorithm $\boldsymbol{\epsilon}$-PAL is to find, with few evaluations $\boldsymbol{f}(\boldsymbol{x})$, a small $\boldsymbol{\epsilon}$-accurate Pareto set of E.

2.4 Gaussian Processes (GP)

$\boldsymbol{\epsilon}$-PAL models $\boldsymbol{f}$ as a draw from an m-variate Gaussian process (GP) distribution. A GP distribution over a real function $f(\boldsymbol{x})$ is fully specified by its mean function $\mu(\boldsymbol{x})$ and its covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ (Rasmussen and Williams, 2006). The *kernel* or covariance function k captures regularity in the form of the correlation of the marginal distributions $f(\boldsymbol{x})$ and $f(\boldsymbol{x}')$.

In our multi-objective setting, we model each objective function $f_i(\boldsymbol{x})$ as a draw from an independent[3] GP distribution.

On every iteration t in our algorithm we choose a design $\boldsymbol{x}_t$ to evaluate, which yields a noisy sample[4] $y_{t,i} = f_i(\boldsymbol{x}_t) + \nu_{t,i}$; after T iterations we have a vector $\boldsymbol{y}_{T,i} = (y_{1,i}, \ldots, y_{T,i})$. Assuming $\nu_{t,i} \sim N(0, \sigma^2)$ (i.i.d. Gaussian noise), the posterior distribution of f_i is a Gaussian process with mean $\mu_{T,i}(\boldsymbol{x})$, covariance $k_{T,i}(\boldsymbol{x}, \boldsymbol{x}')$, and variance $\sigma^2_{T,i}(\boldsymbol{x})$:

$$\mu_{T,i}(\boldsymbol{x}) = \boldsymbol{k}_{T,i}(\boldsymbol{x})^T (\boldsymbol{K}_{T,i} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y}_{T,i}, \tag{2}$$

$$k_{T,i}(\boldsymbol{x}, \boldsymbol{x}') = k_i(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_{T,i}(\boldsymbol{x})^T (\boldsymbol{K}_{T,i} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{T,i}(\boldsymbol{x}'), \tag{3}$$

$$\sigma^2_{T,i}(\boldsymbol{x}) = k_{T,i}(\boldsymbol{x}, \boldsymbol{x}), \tag{4}$$

where $\boldsymbol{x}, \boldsymbol{x}' \in E$, $\boldsymbol{k}_{T,i}(\boldsymbol{x}) = (k_i(\boldsymbol{x}, \boldsymbol{x}_t))_{1 \leq t \leq T}$ and $\boldsymbol{K}_{T,i} = (k_i(\boldsymbol{x}_j, \boldsymbol{x}_\ell))_{1 \leq j, \ell \leq T}$. Note that this posterior distribution captures our uncertainty about $\boldsymbol{f}(\boldsymbol{x})$ for all points $\boldsymbol{x} \in E$.

2.5 Reproducing Kernel Hilbert Spaces (RKHS)

Using Gaussian processes to model the target functions f_i assumes that we know the prior from which they have been generated. This is rarely the case in practice. Therefore, for

3. Note that dependence amongst the outputs could be captured as well; e.g., Bonilla et al. (2008).
4. We use the term "sampling" to refer to evaluating a design via a noisy measurement.

our theoretical analysis we take a more agnostic approach in which we assume that f_i are arbitrary functions from the RKHS associated with kernel k.

The RKHS $\mathcal{H}_k(E)$ is a Hilbert space consisting of functions f on the domain E, endowed with an inner product $\langle \cdot, \cdot \rangle_k$ that satisfies the following properties with respect to a positive definite function k:

- For every $\boldsymbol{x} \in E$, $k(\boldsymbol{x}, \boldsymbol{x}')$ as a function of $\boldsymbol{x}'$ belongs to $\mathcal{H}_k(E)$.
- The reproducing property holds for k, i.e., $\langle f, k(\boldsymbol{x}, \cdot) \rangle_k = f(\boldsymbol{x})$.

The smoothness of the functions $f \in \mathcal{H}_k(E)$ with respect to k is encoded by the norm $||f||_k = \sqrt{\langle f, f \rangle_k}$. Functions with low norm are usually relatively smooth. In our case, as E is a finite set, any f is guaranteed to have bounded norm, i.e., $||f||_k < \infty$, as long as the kernel is universal (such as the Gaussian kernel).

2.6 Problem Statement

Let E be a finite set with a positive definite kernel. We wish to simultaneously optimize m objective functions $f_1, \ldots, f_m : E \to \mathbb{R}$, considering that evaluating $\boldsymbol{f}(\boldsymbol{x})$ for any $\boldsymbol{x} \in E$ is expensive. We wish to identify an $\boldsymbol{\epsilon}$-accurate Pareto set $\Pi_\epsilon(E) \subseteq E$ while minimizing the number of evaluations $\boldsymbol{f}(\boldsymbol{x})$.

In the following, we develop an active learning algorithm that iteratively and adaptively selects a sequence of designs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ to be evaluated, and that uses these evaluations along with the model's predictive uncertainty to predict an $\boldsymbol{\epsilon}$-accurate Pareto set of E. This iterative algorithm terminates when, with high probability, an $\boldsymbol{\epsilon}$-accurate Pareto set of E has been found, and therefore no more evaluations are needed. In addition, we theoretically analyze our algorithm and provide a bound on the number of evaluations required by the algorithm to generate an $\boldsymbol{\epsilon}$-accurate prediction.

3. $\boldsymbol{\epsilon}$-PAL Algorithm

In this section we describe our algorithm: $\boldsymbol{\epsilon}$-*Pareto Active Learning* ($\boldsymbol{\epsilon}$-PAL).

3.1 Overview

Our approach to predicting an $\boldsymbol{\epsilon}$-accurate Pareto set of E trains GP models on a small subset of E. The models predict the objective functions f_i, $1 \leq i \leq m$, allowing us to make statistical inferences about the Pareto-optimality of every point in E. The true value of $\boldsymbol{f}(\boldsymbol{x})$ is approximated by the models as $\hat{\boldsymbol{f}}(\boldsymbol{x}) = \boldsymbol{\mu}(\boldsymbol{x}) = (\mu_i(\boldsymbol{x}))_{1 \leqslant i \leqslant m}$. Additionally $\boldsymbol{\sigma}(\boldsymbol{x}) = (\sigma_i(\boldsymbol{x}))_{1 \leqslant i \leqslant m}$ is interpreted as the uncertainty of this prediction. We capture this uncertainty through the hyper-rectangle[5]

$$Q_{\boldsymbol{\mu},\boldsymbol{\sigma},\beta}(\boldsymbol{x}) = \{\boldsymbol{y} : \boldsymbol{\mu}(\boldsymbol{x}) - \beta^{1/2}\boldsymbol{\sigma}(\boldsymbol{x}) \preceq \boldsymbol{y} \preceq \boldsymbol{\mu}(\boldsymbol{x}) + \beta^{1/2}\boldsymbol{\sigma}(\boldsymbol{x})\}, \tag{5}$$

where β is a scaling parameter to be chosen later.

The goal of the algorithm is to return a set $\hat{P} \subseteq E$ such that, with high probability, $\hat{P}$ is an $\boldsymbol{\epsilon}$-accurate Pareto set of E. $\hat{P}$ may contain points that are not sampled yet, but

5. conservatively bounding the ellipsoid with radii σ_i.

probabilistically, it is guaranteed to cover $Z_\epsilon(E)$, i.e., with high probability, all points in E are $\boldsymbol{\epsilon}$-dominated by a point in $\hat{P}$.

$\boldsymbol{\epsilon}$-PAL iterates over four stages: modeling, discarding, $\boldsymbol{\epsilon}$-Pareto front covering, and sampling. It stops when all remaining points have been either sampled, discarded or determined to belong to $Z_\epsilon(E)$ with high probability. The following sections explain each of these stages, and Alg. 1 presents the corresponding pseudocode.

The algorithm maintains two working sets, indexed by the iteration number t, starting at 0:

- U_t: undecided points, $U_0 = E$,
- P_t: points predicted to be members of an $\boldsymbol{\epsilon}$-accurate Pareto set of E, $P_0 = \emptyset$.

On the first iteration ($t = 0$), all points in E are copied to a set U_0. Subsequently, on any iteration t, a point to be sampled is chosen from U_t or P_t. Points in U_t that with high probability are $\boldsymbol{\epsilon}$-dominated by another point are discarded, i.e., removed from U_t, and points that are determined to belong to $Z_\epsilon(E)$ are moved from U_t to set P_t. The algorithm terminates at some iteration T when U_T is empty. Then $\hat{P} = P_T$ is returned. $\boldsymbol{\epsilon}$-PAL guarantees that with high probability $\hat{P}$ is an $\boldsymbol{\epsilon}$-accurate Pareto set of E.

3.2 Modeling

$\boldsymbol{\epsilon}$-PAL uses Gaussian process inference to predict the mean vector $\boldsymbol{\mu}_t(\boldsymbol{x})$ and the standard deviation vector $\boldsymbol{\sigma}_t(\boldsymbol{x})$ of any point $\boldsymbol{x} \in E$ that has not been discarded yet. This prediction is generated based on the noisy samples obtained. Each point $\boldsymbol{x} \in P_t \cup U_t$ is then assigned its *uncertainty region*, which is the hyperrectangle

$$R_t(\boldsymbol{x}) = R_{t-1}(\boldsymbol{x}) \cap Q_{\boldsymbol{\mu}_t,\boldsymbol{\sigma}_t,\beta_{t+1}}(\boldsymbol{x}), \tag{6}$$

where β_{t+1} is a positive value that defines how large this region is in proportion to $\boldsymbol{\sigma}_t$. In Section 4 we will suggest a value for this parameter. The iterative intersection ensures that all uncertainty regions are non-increasing with t. Intuitively, for a point $\boldsymbol{x} \in E$ that has not been evaluated yet, we can say that with high probability $\boldsymbol{f}(\boldsymbol{x}) \in R_t(\boldsymbol{x})$.

Within $R_t(\boldsymbol{x})$, the pessimistic and optimistic outcomes are $\min(R_t(\boldsymbol{x}))$ and $\max(R_t(\boldsymbol{x}))$, respectively, both taken in the partial order $\preceq$ and unique. Fig. 3(a) shows this with an example. Notice that points that have been evaluated also have an uncertainty region, since we only have access to noisy samples on those points.

3.3 Discarding

The goal of this stage is to discard points in U_t that with high probability are $\boldsymbol{\epsilon}$-dominated by another point in E, while making sure that at least one of such dominating points ends up in P_T.

Under uncertainty, a point $\boldsymbol{x}$ is $\boldsymbol{\epsilon}$-dominated by another point $\boldsymbol{x}'$, with high probability, if the pessimistic outcome of $\boldsymbol{x}'$ ($\min(R_t(\boldsymbol{x}'))$) $\boldsymbol{\epsilon}$-dominates the optimistic outcome of $\boldsymbol{x}$ ($\max(R_t(\boldsymbol{x}))$):

$$\max(R_t(\boldsymbol{x})) \preceq_\epsilon \min(R_t(\boldsymbol{x}')).$$

In this case, $\boldsymbol{x}$ can be discarded, i.e., removed from U_t.

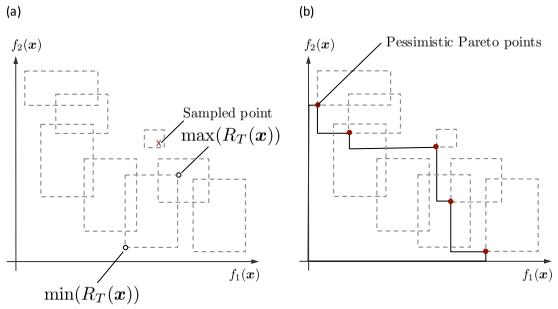


Figure 3: (a) Example of $\min(R_t(\boldsymbol{x}))$ and $\max(R_t(\boldsymbol{x}))$ for $m = 2$. (b) Example of a set of pessimistic Pareto points for $m = 2$.

If this relationship does not hold, either the uncertainty regions are still too large to draw any conclusions, or $\boldsymbol{x}$ and $\boldsymbol{x}'$ are not comparable.

To check if a point is $\boldsymbol{\epsilon}$-dominated by another one in E, $\boldsymbol{\epsilon}$-PAL compares its relationship with all *pessimistic Pareto points*.

Definition 6 (Pessimistic Pareto set) *For a subset $D \subseteq E$, we define $p_{pess}(D)$, or the* pessimistic Pareto set *of D, as the set of points $\boldsymbol{x} \in D$ for which there is no other point $\boldsymbol{x}' \in D$ such that*

$$\min(R_t(\boldsymbol{x})) \preceq \min(R_t(\boldsymbol{x}')).$$

Fig. 3(b) illustrates this situation.

We discard any point $\boldsymbol{x} \in U_t \setminus p_{\text{pess}}(P_t \cup U_t)$ if $\boldsymbol{x} \preceq_\epsilon \boldsymbol{x}'$ for some $\boldsymbol{x}' \in p_{\text{pess}}(P_t \cup U_t)$. This ensures that no element in $p_{\text{pess}}(S_t \cup P_t \cup U_t)$ is discarded and therefore, for every point that is discarded there will always be a point in the working sets that $\boldsymbol{\epsilon}$-dominates it.

A point in $p_{\text{pess}}(P_t \cup U_t)$ is only discarded if it is $\boldsymbol{\epsilon}$-dominated by any point in P_t, since all points in P_t are already guaranteed to belong to the returned set $\hat{P}$. Points in P_t are never discarded.

3.4 $\boldsymbol{\epsilon}$-Pareto Front Covering

The goal of $\boldsymbol{\epsilon}$-PAL is to empty U_t as fast as possible, i.e., with the least amount of points sampled and with the least amount of points in P_t needed to cover the $\boldsymbol{\epsilon}$-Pareto front of E. As previously stated, a point is moved to P_t if it can be determined that with high probability it belongs to $Z_\epsilon(E)$.

Therefore, care must be taken not to sample when all points in U_t can be either discarded or moved to P_t. If there is at least one point in U_t that cannot be discarded or moved to P_t, we proceed with sampling.

Definition 7 ($\boldsymbol{x}$ belongs to $\boldsymbol{Z_\epsilon(E)}$ with high probability) *We can determine that with high probability $\boldsymbol{x} \in E$ belongs to $Z_\epsilon(E)$ if there is no other point $\boldsymbol{x}' \in P_t \cup U_t$ such that*

$$\max(R_t(\boldsymbol{x}')) \succeq_\epsilon \min(R_t(\boldsymbol{x})).$$

In this stage we attempt to update the set P_t, to make progress in covering the $\boldsymbol{\epsilon}$-Pareto front of E, until we find a point in U_t that cannot be moved to P_t. After a point is moved to P_t, we check what points in U_t are $\boldsymbol{\epsilon}$-dominated by it, and therefore can be discarded. Details on this procedure can be found in Alg. 1.

3.5 Sampling

As long as $U_t \neq \emptyset$, on each iteration, a new point $\boldsymbol{x}_t$ is selected for sampling with the following selection rule. Each point $\boldsymbol{x} \in U_t \cup P_t$ is assigned a value

$$w_t(\boldsymbol{x}) = \max_{\boldsymbol{y}, \boldsymbol{y}' \in R_t(\boldsymbol{x})} ||\boldsymbol{y} - \boldsymbol{y}'||_2,$$

which is the diameter of its uncertainty region $R_t(\boldsymbol{x})$. Amongst these points, the one with the largest $w_t(\boldsymbol{x})$ (i.e., most uncertain) is chosen as the next sample $\boldsymbol{x}_t$ to be evaluated. We refer to $w_t(\boldsymbol{x}_t)$ as $\overline{w}_t$. Note that our approach does not simply pick the most uncertain point over the whole space E, but only over the set $U_t \cup P_t$, i.e., the points that (in a statistically plausible manner) might be useful in constructing an $\boldsymbol{\epsilon}$-accurate Pareto set. This is similar to exploration–exploitation tradeoffs commonly encountered in Bayesian optimization.

3.6 Stopping Criteria

The iterations terminate at $t = T$ when $U_T = \emptyset$, i.e., all points have been either moved to P_T or have been discarded. The predicted set $\hat{P} = P_T$ is guaranteed to be an $\boldsymbol{\epsilon}$-accurate Pareto set of E. Termination occurs at the latest when $\overline{w}_t \leq \epsilon$. In this case, dominated points can be discarded in one pass.

3.7 Analysis of Execution Time

In this section, we review the most time consuming subroutines of $\boldsymbol{\epsilon}$-PAL and analyze their computational cost, which depends on the number of points in the working sets: $n_t = |U_t| + |P_t|$. While $n_0 = |E|$ in the first iteration, discarded points are removed from the working sets and thus the number of computations is reduced as t increases. As it is not possible to make conclusions in how n_t decreases with t, we assume the upper bound $n = |E|$. In practice, the number of computations is much lower for most iterations of the algorithm. On the other hand, we assume that the number m of objective functions is a small constant.

In the modeling stage, $\boldsymbol{\epsilon}$-PAL updates Gaussian process models and evaluates the prediction μ and the uncertainty σ for every point. Let s be the number of points that have been sampled, the complexity of this step is $O(s^3 + ns^2)$ (Rasmussen and Williams, 2006). Note that typically $s \ll n$.

In the discarding stage, $\boldsymbol{\epsilon}$-PAL first finds the Pareto pessimistic points of a set. The problem of efficiently determining the Pareto points (or maxima) of a set of vectors is

Algorithm 1 The $\boldsymbol{\epsilon}$-PAL algorithm

Input: design space E ($|E| = n$); GP prior $\mu_{0,i}, \sigma_0, k_i$ for all $1 \leq i \leq n$; $\boldsymbol{\epsilon}$; β_t for $t \in \mathbb{N}$
Output: predicted Pareto set $\hat{P}$

$P_0 = \emptyset$ {predicted set}, $U_0 = E$ {undecided set}
$R_0(\boldsymbol{x}) = \mathbb{R}^n$ for all $\boldsymbol{x} \in E$, $t = 1$
repeat
  $P_t = P_{t-1}$, $U_t = U_{t-1}$
  ——— *Modeling* ———
  Obtain $\boldsymbol{\mu}_t(\boldsymbol{x})$ and $\boldsymbol{\sigma}_t(\boldsymbol{x})$ for all $\boldsymbol{x} \in P_t \cup U_t$
  $R_t(\boldsymbol{x}) = R_{t-1}(\boldsymbol{x}) \cap Q_{\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t, \beta_{t+1}}(\boldsymbol{x})$ for all $\boldsymbol{x} \in P_t \cup U_t$
  ——— *Discard* ———
  $p_{\text{pess}}(P_t)$ = Pareto pessimistic set of P_t
  discard points $\boldsymbol{x}$ in U_t that are $\boldsymbol{\epsilon}$-dominated by some point $\boldsymbol{x}'$ in $p_{\text{pess}}(P_t)$, i.e., where $\max(R_t(\boldsymbol{x})) \preceq_\epsilon \min(R_t(\boldsymbol{x}'))$ (See Alg. 2 for $m = 2$)
  $p_{\text{pess}}(P_t \cup U_t)$ = Pareto pessimistic set of P_t and U_t
  discard points $\boldsymbol{x}$ in $U_t \setminus p_{\text{pess}}(P_t \cup U_t)$ that are $\boldsymbol{\epsilon}$-dominated by some point $\boldsymbol{x}'$ in $p_{\text{pess}}(P_t \cup U_t)$, i.e., where $\max(R_t(\boldsymbol{x})) \preceq_\epsilon \min(R_t(\boldsymbol{x}'))$. (See Alg. 2 for $m = 2$)
  ——— *$\boldsymbol{\epsilon}$-Pareto Front Covering* ———
  repeat
    Choose $\boldsymbol{x}' = \arg\max_{x \in U_t} \{w_t(\boldsymbol{x})\}$
    if for all $\boldsymbol{x} \in P_t \cup U_t \setminus \{\boldsymbol{x}'\}$ it holds that $\max(R_t(\boldsymbol{x})) \preceq_\epsilon \min(R_t(\boldsymbol{x}'))$ **then**
      $P_t = P_t \cup \{\boldsymbol{x}'\}$, $U_t = U_t \setminus \{\boldsymbol{x}'\}$
    else
      break
    end if
  until $U_t = \emptyset$
  ——— *Sampling* ———
  Choose $\boldsymbol{x}_t = \arg\max_{x \in U_t \cup P_t} \{w_t(\boldsymbol{x})\}$
  Sample $\boldsymbol{y}_t(\boldsymbol{x}_t) = \boldsymbol{f}(\boldsymbol{x}_t) + \boldsymbol{\nu}_t$
  $t = t + 1$
until $U_t = \emptyset$
return $\hat{P} = P_t$

Algorithm 2 Discard points in U that are $\boldsymbol{\epsilon}$-dominated by p_{pess} for $m = 2$.

Input: set U; Pareto pessimistic set p_{pess}; $\boldsymbol{\epsilon}$
Output: updated set U

$U' = \{(\boldsymbol{x}, \min(R_t(\boldsymbol{x})) + \boldsymbol{\epsilon}) : \boldsymbol{x} \in p_{\text{pess}}\} \cup \{(\boldsymbol{x}, \max(R_t(\boldsymbol{x}))) : \boldsymbol{x} \in U \setminus p_{\text{pess}}\}$ {U' is a set of $(\boldsymbol{x}, (\hat{f}_1, \hat{f}_2))$ pairs}
$sortU$ = sort U' with respect to $\hat{f}_1$ in ascending order
$currentMax = -\infty$
for $(\boldsymbol{x}, (\hat{f}_1, \hat{f}_2))$ in $sortU$ **do**
  if $\boldsymbol{x} \in p_{\text{pess}}$ **then**
    $currentMax = \hat{f}_2$
  else
    if $\hat{f}_2 \leq currentMax$ **then**
      $U = U \setminus \{\boldsymbol{x}\}$ {discard $\boldsymbol{x}$}
    end if
  end if
end for

addressed by Kung et al. (1975). For $m = 2$, their algorithm starts by ordering the list of vectors in the first dimension f_1. Then, the list is traversed in ascending order while discarding the vectors for which the second dimension f_2 is smaller than the greatest value

of f_2 seen so far. This procedure exhibits computational complexity of $O(n \log n)$ for $m = 2$. The same computational complexity is achieved by a similar implementation for $m = 3$. For $m > 3$, Kung et al. (1975) presents a divide and conquer algorithm with worst case asymptotic complexity of $O(n(\log n)^{m-2}) + O(n \log n)$.

In the discarding stage, $\boldsymbol{\epsilon}$-PAL also discards points in a set U that are $\boldsymbol{\epsilon}$-dominated by a set p_{pess}. This means that, if done naively, every point $\max(\boldsymbol{x})$ with $\boldsymbol{x} \in U$ must be compared with every $\min(\boldsymbol{x}') + \boldsymbol{\epsilon}$ with $\boldsymbol{x}' \in p_{\text{pess}}$. However, this can be done by adapting the above mentioned implementations to compute the Pareto points of a set of vectors to substantially reduce the runtime of the algorithm. Alg. 2 shows the pseudocode for this adaptation. A joined set U' is created from points $\max(\boldsymbol{x})$ with $\boldsymbol{x} \in U$ and $\min(\boldsymbol{x}') + \boldsymbol{\epsilon}$ with $\boldsymbol{x}' \in p_{\text{pess}}$. This joint list is ordered with respect to the first dimension f_1. Then, the list is traversed in ascending order, but the maximum value of f_2 is only updated by a point that belongs to p_{pess}, and only points that belong to U can be discarded. For $m \geq 3$ the adaptation of the algorithm by Kung et al. (1975) is analogous.

In the $\boldsymbol{\epsilon}$-Pareto front covering stage, $\boldsymbol{\epsilon}$-PAL checks if a point $\boldsymbol{x}$ can be guaranteed to belong to $Z_\epsilon(E)$ according to Definition 7. This is done by comparing $\boldsymbol{x}$ with the rest of the points $\boldsymbol{x}'$ in the working sets, until a case in which $\min(R_t(\boldsymbol{x}')) + \boldsymbol{\epsilon} \preceq \max(R_t(\boldsymbol{x}))$ is found. The number of computations for this step is $O(n)$; in practice, however, it is much smaller than n for most iterations. This operation might be done several times per iteration, until no more points in U_t can be moved to P_t or $U_t = \emptyset$. The number of times is assumed to be much smaller than n.

The computational complexity of an iteration is therefore dependent on m and s. For $m = 2$ and $m = 3$ it is $O(ns^2 + s^3 + n \log n)$, and for $m > 3$ it is $O(n(\log n)^{m-2}) + O(ns^2 + s^3 + n \log n)$.

4. Correctness and Sample Complexity

So far we have left the parameter β_t undefined. In this section, we will choose a β_t that guarantees the correctness of the algorithm, and the number of iterations required to meet our bounds.

Of key importance in the convergence analysis is the effect of the regularity imposed by the kernel function k. In our analysis, this effect is quantified by the *maximum information gain* associated with the GP prior. Formally, we consider the information gain

$$I(\boldsymbol{y}_1 \dots \boldsymbol{y}_T; \boldsymbol{f}) = H(\boldsymbol{f}) - H(\boldsymbol{f} \mid \boldsymbol{y}_1 \dots \boldsymbol{y}_T),$$

i.e, the reduction of uncertainty on $\boldsymbol{f}$ caused by (noisy) observations of $\boldsymbol{f}$ on the T first sampled points. The crucial quantity governing the convergence rate is

$$\gamma_T = \max_{\boldsymbol{y}_1 \cdots \boldsymbol{y}_T} I(\boldsymbol{y}_1 \dots \boldsymbol{y}_T; \boldsymbol{f}),$$

i.e., the maximal reduction of uncertainty achievable by sampling T points. Intuitively, if the kernel k imposes strong regularity (smoothness) on $\boldsymbol{f}$, few samples suffice to gather much information about $\boldsymbol{f}$, and as a consequence γ_T grows sub-linearly (exhibits a strong diminishing returns effect). In contrast, if k imposes little regularity (e.g., is close to diagonal), γ_T grows almost linearly with T. Srinivas et al. (2010, 2012) established γ_T as

key quantity in bounding the regret in single-objective GP optimization. Here, we show that this quantity more broadly governs convergence in the much more general problem of predicting the Pareto-optimal set in multi-criterion optimization.

We obtain bounds on the number of iterations required when assuming that the target functions f_i, $0 \leq i \leq m$, lie in the RKHS $\mathcal{H}_k(E)$ corresponding to kernel $k(\boldsymbol{x}, \boldsymbol{x}')$, and that the noise ν_t is an arbitrary martingale difference sequence which has zero mean and is uniformly bounded by σ. Furthermore, in order to obtain the bounds, it is necessary to specify an upper bound[6] B on all $||f_i||_k$.

The following theorem constitutes our main theoretical result.

Theorem 1 *Assume that the true functions f_i lie in the RKHS $\mathcal{H}_k(E)$ corresponding to kernel $k(\boldsymbol{x}, \boldsymbol{x}')$, and that the noise ν_t has zero mean conditioned on the history and is bounded by σ almost surely. Let $\delta \in (0,1)$ and $||f_i||_k^2 \leq B$. Running $\boldsymbol{\epsilon}$-PAL with $\beta_t = 2B^2 + 300\gamma_t \log^3(m|E|t/\delta)$, prior $GP(0, k(\boldsymbol{x}, \boldsymbol{x}'))$ and noise $N(0, \sigma^2)$, the following holds with probability $1-\delta$.*

An $\boldsymbol{\epsilon}$-accurate Pareto set can be obtained after at most T iterations, where T is the smallest number satisfying

$$\sqrt{\frac{C_1 \beta_T \gamma_T}{T}} \geq \epsilon. \tag{7}$$

Here, $C_1 = 8/\log(1+\sigma^{-2})$, $\epsilon = ||\boldsymbol{\epsilon}||_\infty$, and γ_T depends on the type of kernel used.

This means that, with high probability, $\overline{w}_t$ is bounded by $O^*((B\sqrt{\gamma_T}+\gamma_T)/\sqrt{T})$. Note that in practice one might want to, in addition to *identifying* an $\boldsymbol{\epsilon}$-accurate Pareto front, seek to be sufficiently confident such that for all predicted Pareto points (i.e., points $\mathbf{x}$ in P_T in Algorithm 1) it holds that $w_T(\mathbf{x}) \leq \epsilon$. This requirement can be added to the stopping condition in Line 26 of Algorithm 1, and the same sample complexity of Theorem 1 holds.

4.1 Proof Outline

The above theorem implies that by specifying δ and a target accuracy $\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon}$-PAL automatically stops when the target error is achieved with confidence $1-\delta$. Additionally, the theorem bounds the number of iterations T required to obtain this result.

Our strategy for the proof consists of four parts. First, Lemma 1 shows that $|f_i(\boldsymbol{x}) - \mu_{t-1,i}(x)| \leq \beta_t^{1/2}\sigma_{t-1,i}(\boldsymbol{x})$ for $1 \leq i \leq m$, $t \geq 1$, and for all $\boldsymbol{x} \in E$. Then, we analyze how $\overline{w}_t$ decreases with t. Subsequently, we relate $\boldsymbol{\epsilon}$ and $\overline{w}_t$ to ensure the termination of the algorithm. The last two parts are analyzed in Section 4.2. Finally, Lemma 8 supports the accuracy of the $\boldsymbol{\epsilon}$-PAL, given that the proper β_t value is used on every iteration t.

4.2 Reduction in Uncertainty

The first step of the proof is to show that with probability at least $1-\delta$, $\boldsymbol{f}(\boldsymbol{x})$ lies for all $\boldsymbol{x} \in E$ within the uncertainty region (see (5) and (6)):

$$R_t(\boldsymbol{x}) = Q_{\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0, \beta_1}(\boldsymbol{x}) \cap \cdots \cap Q_{\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t, \beta_{t+1}}(\boldsymbol{x}),$$

6. While in practice this might not be possible, a standard guess-and-doubling approach can be applied.

which is achieved by choosing $\beta_t = 2B^2 + 300\gamma_t \log^3(m|E|t/\delta)$.

Srinivas et al. (2010) showed that information gain can be expressed in terms of the predicted variances. Similarly, we show how the cumulative $\sum_{k=1}^{t} \overline{w}_k$ can also be expressed in terms of maximum information gain γ_t. Since the sampling rules used by $\boldsymbol{\epsilon}$-PAL guarantee that $\overline{w}_t$ decreases with t, we get the following bound for $\overline{w}_t$. With probability $\geq 1-\delta$,

$$\overline{w}_t \leq \sqrt{\frac{C_1 \beta_t \gamma_t}{t}} \text{ for all } t \geq 1, \tag{8}$$

where $C_1 = 8/\log(1+\sigma^{-2})$ and β_t is as before.

The proof is supported by Lemmas 1 to 5 found in the appendix of this paper. Key challenges and differences in comparison to Srinivas et al. (2010) include (1) dealing with multiple objectives; (2) the use of a different sampling criterion; and (3) incorporating the monotonic classification scheme.

We also show that, with probability $1-\delta$, the algorithm terminates with no further sampling at iteration T if

$$\overline{w}_T \leq \epsilon, \tag{9}$$

where $\epsilon = ||\boldsymbol{\epsilon}||_\infty = \max(\epsilon_i)_{1 \leq i \leq m}$. This is proved in Lemma 7.

4.3 Explicit Bounds for the Squared Exponential Kernel

Theorem 1 holds for general covariance functions $k(\boldsymbol{x}, \boldsymbol{x}')$. Srinivas et al. (2010) derived bounds for γ_T depending on the choice of kernel. These can be used to specialize Theorem 1.

We illustrate this using the squared exponential kernel as example, i.e., $k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(l^{-2}\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2\right)$ for some $l > 0$.

Let $E \subset \mathbb{R}^d$ be compact and convex, $d \in \mathbb{N}$. According to Srinivas et al. (2010), for $m = 1$, there exists a constant K such that

$$\gamma_t \leq K \log^{d+1} t \text{ for all } t > 1.$$

For $m > 1$, since we assume i.i.d. GPs, we thus get

$$\gamma_t \leq Km \log^{d+1} t$$

and hence the following corollary to Theorem 1.

Corollary 1 *Let k_i be the squared exponential kernel used by $\boldsymbol{\epsilon}$-PAL, and assume f_i lies in the RKHS with its norm bounded by $||f_i||_k^2 \leq B$, for all $1 \leq i \leq m$. When choosing $\delta \in (0,1)$, a target accuracy specified by $\boldsymbol{\epsilon}$, the following holds with probability $1-\delta$. $\boldsymbol{\epsilon}$-PAL terminates after at most T iterations, where T is the smallest number satisfying*

$$\frac{\sqrt{16B^2K^2m\log^{d+1}T + 2400K^2m^2\log^{2(d+1)}T\log^3(m|E|T/\delta)}}{\sqrt{T\log(1+\sigma^{-2})}} \geq \epsilon.$$

These results suggests that, under both scenarios, ϵ increases as T decreases in the following manner: Asymptotically, for any $\rho > 0$, as well as fixed m, n and d, we have $T = O(\frac{1}{\epsilon^{2+\rho}})$.

5. Discussion and Implementation Details

We now discuss some of the limitations, choices made, and other aspects that arose when implementing and using our $\boldsymbol{\epsilon}$-PAL algorithm.

5.1 Sampling Strategy

After clearly non-competitive designs are removed in the discarding phase, our sampling strategy (Section 3.5) chooses the one with the highest uncertainty. Other choices would be possible, e.g, chosing the one that maximizes the expected gain in hypervolume. Our choice aims at reducing time to termination, which is achieved by reducing uncertainty in the model as fast as possible.

5.2 Parameterization

For practical usage, two parameters, namely $\boldsymbol{\epsilon}$ and δ, need to be specified. These parameters relate to the desired level of accuracy of the prediction. Our theoretical bounds are likely to be loose in practice; thus, it may be useful to choose more "aggressive" values than recommended by the theory. The choice of δ impacts the value of β_t and therefore the convergence rate of the algorithm, since the latter scales the uncertainty regions $R_t(\boldsymbol{x})$. Since the analysis is conservative, scaling down β_t, possibly to be constant, is a viable option.

In contrast, the choice of $\boldsymbol{\epsilon}$ should pose no problem. One only may consider scaling the objective functions so that all ϵ_i components of $\boldsymbol{\epsilon}$ have comparable values.

5.3 Kernel Hyper-Parameters

So far, we have assumed that the kernel function is given. Usually, its parameters need to be chosen. Therefore, prior to running $\boldsymbol{\epsilon}$-PAL, it may be practical to randomly sample a small fraction of the design space and to optimize the parameters (e.g., by maximizing the marginal likelihood). One may also consider maintaining uncertainty in the hyper-parameters by placing and updating priors on them. These can then be marginalized to obtain suitable hyper-rectangles to capture the uncertainty about the designs. This extension poses new challenges in computational efficiency (i.e., using approximate Bayesian inference), and we defer it to future work.

5.4 Scalability

The computational complexity of our algorithm depends on our ability to perform efficient inference in Gaussian processes. In general, this complexity grows cubically with the number of samples (function evaluations), which can pose a challenge in large design spaces. This is a general issue in Bayesian optimization, and not specific to our approach. Fortunately, there are many techniques available for scaling up Gaussian process inference that our approach can immediately benefit from (see, e.g., Rasmussen and Williams (2006)).

5.5 Continuous vs. discrete domains

In our approach, we have focused on finite, discrete domains E, which often naturally arise in design-space exploration problems as those considered in our experiments. For Lipschitz-continuous objectives (which are commonly assumed in Bayesian optimization), it is be possible to handle continuous compact domains E via standard covering arguments. Given the allowed tolerances $\boldsymbol{\epsilon}$ and Lipschitz constants, one can construct a discretization $\hat{E}$ such that for any point $\boldsymbol{x} \in E$ in the domain, there is one in the discretization $\boldsymbol{x}' \in \hat{E}$ such that $\boldsymbol{f}(\boldsymbol{x}') \succeq_\epsilon \boldsymbol{f}(\boldsymbol{x})$. It can be seen that any $\boldsymbol{\epsilon}$-accurate Pareto set of $\hat{E}$ is a $2\boldsymbol{\epsilon}$-accurate Pareto set of E. This discretization-based approach becomes impractical with higher dimensions. Developing a variant of our approach that does not require an explicit discretization is an interesting direction for future work.

5.6 Current Implementation

For our experimental evaluation, we implemented a prototype that is restricted to $m = 2$ objectives. The code is available at Zuluaga et al. (2015).

6. Related Work

We now discuss different lines of related work.

6.1 Evolutionary Algorithms

One class of approaches uses evolutionary algorithms to approximate the Pareto frontier via a population of evaluated designs that is iteratively evolved (Künzli et al., 2005; Coello et al., 2006; Zitzler et al., 2002). Most of these approaches do not use models for the objectives, and consequently cannot make predictions about unevaluated designs. As a consequence, a large number of evaluations are typically needed for convergence with reasonable accuracy. To overcome this challenge, model-based (or "response surface") approaches approximate the objectives by models, which are fast to evaluate. A concrete example is the algorithm presented by Emmerich et al. (2006), which uses Gaussian random fields metamodels to predict the objective functions. These models are used to screen each generation of the population and to extract promising individuals. On the other hand, Laumanns and Ocenasek (2002) and Buche et al. (2005) investigate the use of Gaussian models to maintain and improve the diversity of the population during the offspring-generation phase.

We will compare against ParEGO (Knowles, 2006) (explained in Section 6.6 below), which improves on the above work by combining the evolutionary approach with optimization to reduce the number of evaluations needed.

6.2 Scalarization to the Single-Objective Setting

An alternative approach to multi-objective optimization problems is the reduction to a single-objective problem (for which a wealth of methods are available). This is commonly done via scalarization, for example by considering convex combinations of the objective functions (Boyd and Vandenberghe, 2004). Some evolutionary algorithms use this approach to tackle multiple objectives. For example, the approach presented by Zhang and Li (2007)

optimizes simultaneously several single-objective subproblems that are created using different scalarization criteria. The diversity amongst the subproblems will lead to diversity in the approximated Pareto front.

A number of global optimization algorithms are also conditioned to the multi-objective setting by using scalarization. As a concrete example, Zhang et al. (2010) and Knowles (2006) extend the single-objective efficient global optimization (EGO) approach of Jones et al. (1998) by decomposing the optimization problem into several single-objective subproblems. A GP model is built for every subproblem, and sample candidates are selected based on their expected improvement.

A major disadvantage of the scalarization approach is that without further assumptions (e.g., convexity) on the objectives, not all Pareto-optimal solutions can be recovered (Boyd and Vandenberghe, 2004). Therefore, we avoid scalarization in our approach.

6.3 Heuristics-based Methods

Instead of weighted combinations, numerous domain-specific heuristics have been proposed that aim at identifying Pareto-optimal solutions. These approaches typically combine search algorithms to suit the nature of the problem (Deng et al., 2008; Palermo et al., 2009; Zuluaga et al., 2012a) and defy theoretical analysis to provide bounds on the sampling cost. With this work we aim at creating a method that generalizes across a large range of applications and target scenarios and that is analyzable, i.e., comes with theoretical guarantees.

6.4 Single-Objective Bayesian Optimization and Active Learning

In the single-objective setting, there has been much work on active learning, in particular for classification (see, e.g., Settles (2010) for an overview). For optimization, model-based approaches are used to address settings where the objective is noisy and expensive to evaluate. In particular in Bayesian optimization (see the survey by Brochu et al. (2010)), the objective is modeled as a draw from a stochastic process (often a Gaussian process), as originally proposed by Mockus et al. (1978). The advantage of this approach is the flexibility in encoding prior assumptions (e.g., via choice of the kernel and likelihood functions), as well as the ability to guide sampling: several different (usually greedy) heuristic criteria have been proposed to pick the next sample based on the predictive uncertainty of the Bayesian model. A common example is the EGO approach of Jones et al. (1998), which uses the expected improvement. In recent years, there have been considerable theoretical advances in Bayesian optimization. Several analyses focus on the case of deterministic (i.e., noise-free) observations. Vazquez and Bect (2010) prove that under some conditions the EGO approach produces a dense sequence of samples in the domain, i.e., asymptotically getting arbitrarily close to the optimum. Bull (2011) go further in providing convergence rates for this setting. Srinivas et al. (2010) analyzed the GP-UCB criterion (Cox and John, 1997), and proved global convergence guarantees and rates for Bayesian optimization allowing noisy observations. We build on their results to establish guarantees about our ϵ-PAL algorithm in the multi-objective setting. de Freitas et al. (2012) improve the results of Srinivas et al. (2010) using a UCB-like algorithm under deterministic observations to obtain exponential regret bounds.

6.5 Multi-Objective Bayesian Optimization and Active Learning

The algorithm PAL by Zuluaga et al. (2013), is an earlier version of ϵ-PAL. It also requires a parameter ϵ that allows the user to set different levels of prediction accuracy. PAL attempts to classify points as Pareto optimal or not Pareto-optimal until all points have been classified; it uses ϵ to ease this classification. However, it fails to generate an ϵ-accurate Pareto set. It only uses ϵ to stop the algorithm in different stages of training, and as a result, less accuracy is achieved when the value of ϵ is increased.

There are two main advantages of ϵ-PAL over PAL. The more effective use of ϵ is one of them. This not only allows it to generate an ϵ-accurate Pareto set, but it also reduces the runtime of the algorithm, as it attempts to remove redundancy and discards points more efficiently along the execution of the algorithm. In addition, the convergence bounds are also defined in terms of ϵ, which is intuitive. In contrast, PAL uses the hypervolume error, a concept that is more difficult to reason about.

The other advantage is the asymptotic improvement in the number of computations required per iteration, which is independent to the improvements just mentioned above. As analyzed in Section 3.7, for problems with two or three objective functions, ϵ-PAL requires $O(n \log n)$ computations on every iteration, whereas PAL requires $O(n^2 \log n)$. For problems with more than two objective functions, ϵ-PAL requires $O(n(\log n)^{m-2}) + O(n \log n)$ computations on every iteration, whereas PAL requires $O(n^2(\log n)^{m-2}) + O(n \log n)$. Here, n is the number of elements in the design space. This is very important since it allows ϵ-PAL to handle larger design spaces.

There have been recent approaches that simplify to some extent the PAL algorithm. Campigotto et al. (2014) proposes an active learning algorithm that also uses Gaussian process modeling to predict the objective function. It iteratively samples the most uncertain point in the design space until a threshold in information gain is reached. On the other hand, Steponavice et al. (2014) propose an algorithm that probabilistically classifies points as Pareto-optimal or not. This probability is used to provide flexibility on the accuracy of the prediction. This algorithm does not use Gaussian process modeling with the goal of improving runtime. Neither of these two approaches provides an approximation to the Pareto front with some granularity as ϵ-PAL does. Also, they do not provide any theoretical support on the convergence of the algorithm.

Multi-objective Bayesian optimization is also related to several other variants of Bayesian optimization studied in the literature. For example, there has been work on contextual (Krause and Ong, 2011), multi-task (Swersky et al., 2013) and collaborative (Bardenet et al., 2013) Bayesian optimization. These approaches can be viewed as carrying out Bayesian optimization for multiple related objectives, exploiting dependencies between them in order to share statistical strength. While also considering multiple objectives, the goal is to find separate optimizers for each of them, rather than identifying a Pareto frontier. Lastly Multi-objective Bayesian optimization is also related to Bayesian optimization under unknown constraints (Gelbart et al., 2014), since one way to obtain Pareto-optimal solutions is to optimize one objective under constraints on the others. The approach of (Gelbart et al., 2014) however only aims to identify a (single) optimal feasible solution as opposed to an entire Pareto frontier.

6.6 Multi-Objective Global Optimization and Evolutionary Algorithms

Global optimization algorithms have been paired with evolutionary algorithms to find the solution that maximizes an acquisition function, for example expected improvement. The best amongst these appears to be ParEGO (Knowles, 2006), which also uses GP models of the objective functions. On every iteration, the problem is scalarized using a different weight vector, and the solution that maximizes the expected improvement, based on the current single-objective function, is chosen for evaluation. An evolutionary algorithm performs the search using the GP models to asses the expected improvement of each solution. A similar approach, presented by Zhang et al. (2010), generates several populations per iteration to take advantage of a parallel implementation. As a result, several points are sampled on every iteration. We will compare our approach against ParEGO, as it also generates serialized function evaluations; this method might be preferred when the objective functions are expensive to evaluate.

7. Experiments

In this section we evaluate $\boldsymbol{\epsilon}$-PAL on three real world data sets obtained from different applications in computer science and engineering. We assess the prediction error versus the number of evaluations required to obtain it, for different settings of the accuracy parameter $\boldsymbol{\epsilon}$. Further, we compare with PAL (Zuluaga et al. (2013)) and ParEGO (Knowles, 2006), a state-of-the-art multi-objective optimization method based on evolutionary algorithms, using an implementation provided by the authors and adapted to run with our data sets. ParEGO also uses GP modeling to aid convergence.

Before presenting the results we introduce the data sets, and explain the experimental setup.

7.1 Data Sets

The first data set, called SNW, is taken from Zuluaga et al. (2012b). The design space consists of 206 different hardware implementations of a sorting network for 256 inputs. Each design is characterized by $d = 3$ parameters. The objectives are area and throughput when synthesized for a field-programmable gate array (FPGA) platform. This synthesis is very costly and can take up to many hours for large designs.

The second data set, called NoC, is taken from Almer et al. (2011). The design space consists of 259 different implementations of a tree-based network-on-chip, targeting application-specific circuits (ASICs) and multi-processor system-on-chip designs. Each design is defined by $d = 4$ parameters. The objectives are energy and runtime for the synthesized designs run on the Coremark benchmark workload. Again, the evaluation of each design is very costly.

The third data set, called SW-LLVM, is taken from Siegmund et al. (2012). The design space consists of 1023 different compiler settings for the LLVM compiler framework. Each setting is specified by $d = 11$ binary flags. The objectives are performance and memory footprint for a given suite of software programs when compiled with these settings. Note that the Pareto-optimal set consists of two points only.

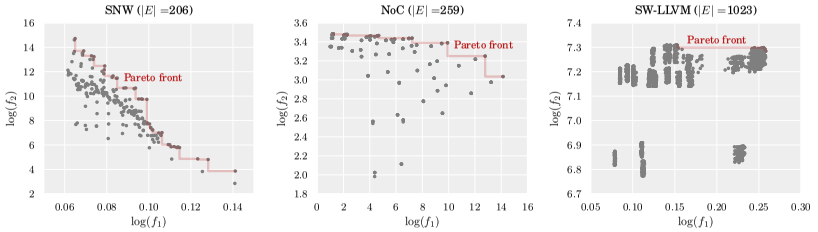


Figure 4: Objective space of the input sets use in our experiments.

Data set	d	m	$\lvert E\rvert$
SNW	3	2	206
NoC	4	2	259
SW-LLVM	11	2	1023

Table 1: Data sets used in our experiments.

The main characteristics of the data sets are summarized in Table 1; note that in all cases $m = 2$. To obtain the ground truth, we completely evaluated all data sets to determine P in each case. This was very costly, most notably, it took 20 days for NoC alone. The evaluations are plotted in log scale in Fig. 4; the Pareto fronts are emphasized. All the data is normalized such that all objectives are to be maximized.

7.2 Experimental Setup

Our implementation of $\boldsymbol{\epsilon}$-PAL uses the Gaussian Process Regression and Classification Toolbox for Matlab (Rasmussen and Nickisch, 2013). In our experiments we used the squared exponential covariance function with automatic relevance determination. The standard deviation of the noise ν is fixed to 0.1. We use

$$\beta_t^{1/2} = \frac{1}{3}\sqrt{2\log(m|E|\pi^2 t^2/(6\delta))},$$

which is the value suggested in Zuluaga et al. (2013) scaled by 1/3. This factor is empirical and used because the theoretical analysis is too conservative. In Srinivas et al. (2010) 1/5 is used.

All of the experiments were repeated 200 times and the median of the outcomes is shown in the plots. Additionally, several values of $\boldsymbol{\epsilon} = (\epsilon_i)_{1\leq i\leq 2}$ were evaluated, where ϵ_i is proportional to the range $r_i = \max_{\boldsymbol{x}\in E}\{f_i(\boldsymbol{x})\} - \min_{\boldsymbol{x}\in E}\{f_i(\boldsymbol{x})\}$. We start with $\epsilon_i = 0.01\times r_i$ and increase it up to $\epsilon_i = 0.3\times r_i$. When we say $\epsilon = 1\%$, we mean that ϵ_i is 1% of the range r_i.

Prior to running $\boldsymbol{\epsilon}$-PAL, a subset of the design space was evaluated in order to initialize the training set and optimize the kernel parameters used by the model. For SNW and NoC,

15 points were chosen uniformly at random. For LLVM-SW, as the dimensionality of the design space is much larger, 30 points were taken.

All of the experiments were run on a machine equipped with two 6-Core Intel Xeon X5680 (3.06GHz) and 144GB (1333MHz) of memory.

7.3 Prediction Error

After running $\boldsymbol{\epsilon}$-PAL with any of the data sets, a prediction $\hat{P} \subset E$ of the true Pareto set is returned. Some of the points in $\hat{P}$ are then already evaluated, i.e., $\boldsymbol{f}$ is already known; for the others the exact value of $\boldsymbol{f}$ was not necessary to generate the prediction. For such points, the objective functions are evaluated after running $\boldsymbol{\epsilon}$-PAL to determine their values and thus all of $\boldsymbol{f}(\hat{P})$. We compare the prediction $\hat{P}$ with the true Pareto set P, where $\boldsymbol{f}(P)$ is obtained through exhaustive evaluation. The comparison uses a percentage prediction error $e(P, \hat{P})$ based on the concept of $\boldsymbol{\epsilon}$-dominance. It is the average maximum distance found between a point in $\boldsymbol{f}(P)$ and the closest point in $\boldsymbol{f}(\hat{P})$.

When considering two objective functions, i.e., $m = 2$, $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))$. In this case, the prediction error is calculated as

$$e(P, \hat{P}) = \text{avg}_{x \in P} \text{min}_{x' \in \hat{P}} \text{max}_{1 \leq i \leq 2} \frac{(f_i(\boldsymbol{x}) - f_i(\boldsymbol{x}')) \cdot 100}{r_i}.$$

The distance is expressed as a percentage of the range r_i, in order to compare the results between the two dimensions and amongst the different data sets.

7.4 Quality of Pareto Front Prediction

The first row of Fig. 5 shows the $\boldsymbol{\epsilon}$-accurate Pareto front $Z(\hat{P})$ obtained with $\boldsymbol{\epsilon}$-PAL for three different configurations of $\boldsymbol{\epsilon}$, using the first data set. The red shaded line is the true Pareto front found by exhaustive evaluation. The gray points are the ones that have been either sampled during the execution of $\boldsymbol{\epsilon}$-PAL or selected to be part of the $\boldsymbol{\epsilon}$-Pareto front. We can see that, as intended, with larger values of $\boldsymbol{\epsilon}$, a less accurate Pareto front is obtained, but also fewer points are evaluated. The rest of the points are discarded by $\boldsymbol{\epsilon}$-PAL during its execution.

The second row of Fig. 5 shows equivalent plots obtained with PAL. PAL attempts to classify points as Pareto optimal or not Pareto-optimal until all points have been classified; it uses $\boldsymbol{\epsilon}$ to ease this classification. Therefore, as $\boldsymbol{\epsilon}$ increases, more points are selected as Pareto optimal. The gray points in the plots correspond to solutions that are either evaluated during the execution of PAL or classified as Pareto optimal. It is clear that PAL does not generate an $\boldsymbol{\epsilon}$-accurate Pareto set, but simply uses a parameter $\boldsymbol{\epsilon}$ to make the algorithm terminate at different stages. The prediction can be as precise as the models for f_i generated with the samples gathered by the termination of the algorithm. For small values of $\boldsymbol{\epsilon}$, the resulting Pareto front is close to the true Pareto front because more samples have been taken and a more accurate classification can be made. On the other hand, for large values of $\boldsymbol{\epsilon}$, this happens because most points are classified as Pareto optimal, even though they are far from the true Pareto front. In all cases, the returned Pareto front attempts to be as close as possible to the true Pareto front.

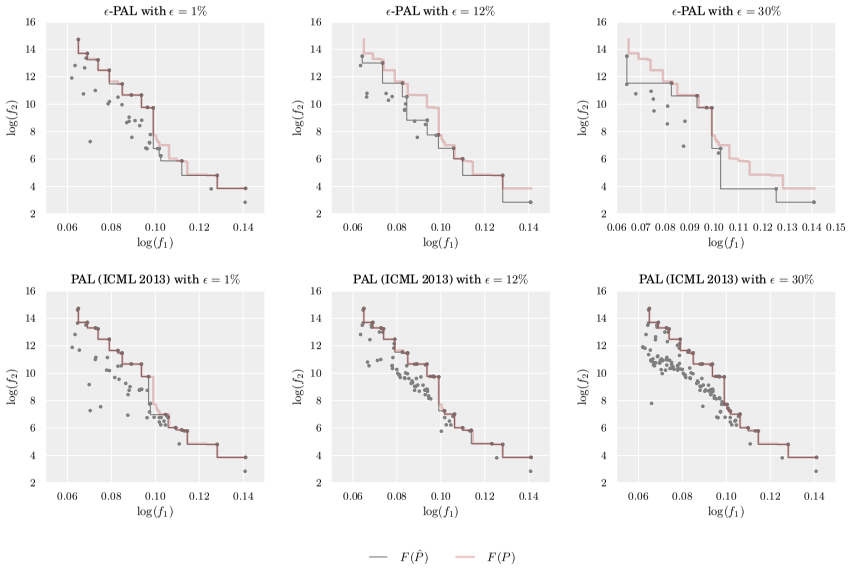


Figure 5: The first row shows the $\boldsymbol{\epsilon}$-accurate Pareto fronts (black line) of the SNW design space obtained with $\boldsymbol{\epsilon}$-PAL for $\epsilon \in \{1\%, 12\%, 30\%\}$. The second row shows equivalent plots for PAL in which the prediction $\hat{P}$ is composed of the sampled points and the points classified as Pareto optimal at the termination of the algorithm. The red shaded line in each plots is the true Pareto front of the design space.

Fig. 6 shows a set of experiments in which we do both, exploring the effect of choosing $\boldsymbol{\epsilon}$ and comparing against PAL and ParEGO (Knowles, 2006). Every plot in Fig. 6 corresponds to one data set in Table 1. In each case, the x-axis shows the number of evaluations (sampling cost) of $\boldsymbol{f}$. For $\boldsymbol{\epsilon}$-PAL and PAL this is T (the total number of iterations) plus the evaluations of designs in $\hat{P}$ that have not been evaluated yet while running the algorithm. On the y-axis, we show the average percentage error $e(P, \hat{P})$ as defined above. We measure the error at each iteration, and plot $(t, e(P, \hat{P}))$. As expected, the error in all cases decreases with increasing numbers of evaluations.

$\boldsymbol{\epsilon}$-PAL provides a wide range of accuracy-cost trade-offs as an effect of choosing different $\boldsymbol{\epsilon}$ configurations. When $\epsilon = 30\%$, only a few iterations are made by the algorithm, returning an error of less than 7% with less than 30 function evaluations in all cases. On the other hand, for $\epsilon = 1\%$, $\boldsymbol{\epsilon}$-PAL yields an error of less than 0.7% with less than 50 function evaluations in all cases.

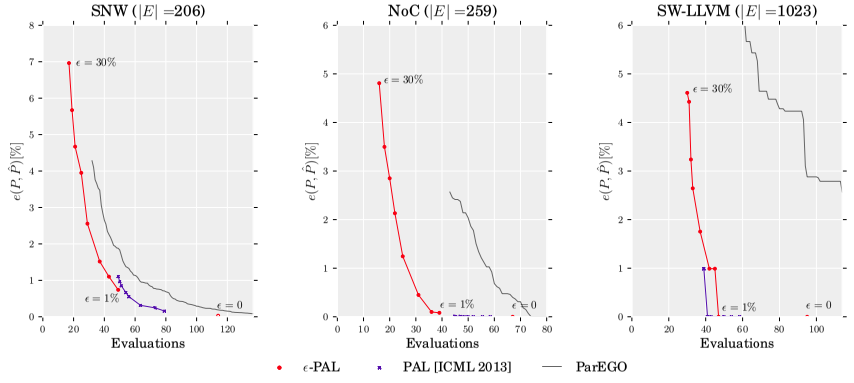


Figure 6: Prediction error $e(P, \hat{P})$ vs. number of evaluations. For $\boldsymbol{\epsilon}$-PAL and PAL the values $\epsilon \in \{30\%, 20\%, 16\%, 12\%, 8\%, 4\%, 2\%, 1\%\}$ are used. The corresponding points are consecutive in the lines for $\boldsymbol{\epsilon}$-PAL and PAL.

The measured error $e(P, \hat{P})$ is comparable with ϵ as they both measure how far the predicted Pareto front is from the true Pareto front. ϵ measures the desired accuracy, and $e(P, \hat{P})$ measures what is obtained. We can then say that $\boldsymbol{\epsilon}$-PAL meets the expectations on the prediction for all $\boldsymbol{\epsilon}$ configurations and in all cases.

The plots in Fig. 6 also show a point in which $\boldsymbol{\epsilon}$-PAL achieves an error of 0. This is obtained with $\epsilon = 0$, $\delta = 0.05$ and β_t having the exact value suggested by Theorem 1. Under these configurations, $\boldsymbol{\epsilon}$-PAL yields a perfect prediction with less than 115 evaluations in all cases.

PAL (purple line) generates a more constrained set of trade-offs using the same $\boldsymbol{\epsilon}$ configurations that were used for $\boldsymbol{\epsilon}$-PAL. The actual $\hat{P}$ reported in (Zuluaga et al., 2013) corresponds to the Pareto-optimal points amongst all pairs $(\mu_1(\boldsymbol{x}), \mu_2(\boldsymbol{x}))$ for $\boldsymbol{x} \in E$. On the other hand, ParEGO (gray line) does not provide a methodology to stop the algorithm in different stages of accuracy and cost. We then plot a continuous line that shows the error obtained by the set S_t of sampled points on every iteration of the algorithm t, i.e., we plot points $(|S_t|, e(P, S_t))$. ParEGO uses a heuristic to find the number of samples of the starting population depending on the characteristics of the design space. Hence the line always starts with a certain minimum number of evaluations.

We observe that $\boldsymbol{\epsilon}$-PAL and PAL in all cases significantly improve over ParEGO. In particular, the gains on SW-LLVM were considerable. Overall, the number of evaluations was reduced by 1/3 to 2/3 in all cases when comparing against ParEGO. $\boldsymbol{\epsilon}$-PAL overall compares favorably to PAL except for SW-LLVM, where PAL wins for a small range in the error/performance tradeoff space.

As explained above, $\boldsymbol{\epsilon}$-PAL returns an $\boldsymbol{\epsilon}$-accurate Pareto set and PAL returns a more dense Pareto set, therefore, the prediction error in PAL is always fairly small and in most of the cases it requires more function evaluations. In the case of SW-LLVM, there are some trade-offs obtained with $\boldsymbol{\epsilon}$-PAL that yield slightly higher errors than those obtained with PAL. This is because $\boldsymbol{\epsilon}$-PAL eliminates solutions along its execution and as the corresponding design space has only one Pareto point, when this is removed, the error obtained increases considerably. However the error is within the expected range, i.e., below that indicated by $\boldsymbol{\epsilon}$. Additionally, if the design space is not too large, after the termination of the algorithm, all points that were discarded can be again predicted by the model to find a more accurate Pareto set similar to what PAL returns.

In conclusion, $\boldsymbol{\epsilon}$-PAL in most cases produces better trade-offs than PAL and ParEGO. In comparison to PAL, it uses more effectively the fact that the users, by setting $\boldsymbol{\epsilon}$, are expressing their desire to generate only $\boldsymbol{\epsilon}$-accurate Pareto fronts in exchange for a smaller number of function evaluations. This means that it can offer a wider range of trade-offs between accuracy and number of evaluations. Moreover, the resulting Pareto fronts contain a variety of trade-offs between the underlying multi-objective design space, that are spread evenly along the range of possibilities.

7.5 Execution Time

A drawback of PAL is the high number of computations required per iteration, which constrains it to small design spaces. As explained in Section 6, $\boldsymbol{\epsilon}$-PAL reduces the number of computations per iteration by a factor of n while achieving better results than PAL. In this section, we compare the runtime of executing PAL and $\boldsymbol{\epsilon}$-PAL with our three data sets. Another advantage of $\boldsymbol{\epsilon}$-PAL that helps improving its computational efficiency is that it uses $\boldsymbol{\epsilon}$ to discard more points from the design space as its value increases. In PAL on the other hand, as $\boldsymbol{\epsilon}$ increases, the number of points classified as Pareto optimal increases, and thus those points require computations on every iteration until the algorithm terminates.

The first plot in Fig. 7 shows runtime speedups, obtained with the three data sets and all the values of $\boldsymbol{\epsilon}$ considered before. We measured for each value of $\boldsymbol{\epsilon}$ the runtime for $\boldsymbol{\epsilon}$-PAL and PAL and divide them to obtain the speedups. Here we assume that the cost of evaluating $\boldsymbol{f}$ is 0. The plot shows that, as expected, the speedups improve as n increases. Therefore, LLVM-SW, which has the largest design space of all data sets, obtains the largest speedups, from 45× with $\epsilon = 1\%$ up to 420× with $\epsilon = 30\%$. Better speedups are obtained with larger values of $\boldsymbol{\epsilon}$ since more points can be discarded during the first iterations.

The second plot in Fig. 7 shows equivalent results but this time assuming that evaluating $\boldsymbol{f}$ takes 30 minutes. In this case, the speedups are smaller, going up to almost 3× for $\epsilon = 30\%$. This is because the data sets are relatively small and thus the gains in runtime are offset by the high cost of evaluating $\boldsymbol{f}$. In other words, the speedups that we observed here are speedups on function evaluations and are related to the improvements that can be visualized in Fig. 6. These improvements come from the fact that $\boldsymbol{\epsilon}$-PAL requires less function evaluations than PAL to achieve an error equivalent to the requested $\boldsymbol{\epsilon}$. As a result, the denser the true Pareto front of the design space, the better the speedups obtained by using $\boldsymbol{\epsilon}$-PAL. For this reason, LLVM-SW does not generate important speedups on function evaluations, since the true Pareto front is composed of only one point. The best speedups

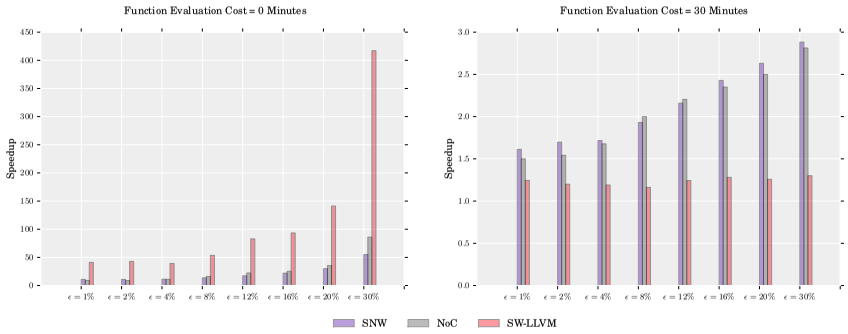


Figure 7: Speedups obtained by using ϵ-PAL against using PAL. The plot of the left shows the speedups on the runtime of ϵ-PAL, assuming that function evaluations are free. The plot on the right shows speedups assuming that one function evaluation takes 30 minutes.

are obtained with SNW, followed by NoC, tha same order as the densities of their true Pareto curves.

8. Conclusions

In this paper, we proposed a novel algorithm for efficiently localizing an ϵ-accurate Pareto-frontier in multi-objective optimization. In the spirit of Bayesian optimization, it uses Gaussian processes to exploit prior information about the objective functions' regularity to predict the objective functions, to guide the sampling process, and to make inferences about the optimality of designs without directly evaluating them. We presented an extensive theoretical analysis including bounds for the number of samples required to achieve the desired target accuracy.

The algorithm described in this paper presents major improvements from its predecessor PAL. First, it returns an ϵ-accurate Pareto front instead of a dense approximation of the true Pareto front. It is often the case that a small difference in the objective function, conveyed to the algorithm with the parameter ϵ, does not affect the quality of a design significantly. Thus, it is preferable to reduce the number of evaluations made and to obtain a set of close-to-optimal solutions well distributed in the objective space.

A second improvement over PAL is that ϵ-PAL reduces the asymptotic runtime, which facilitates the exploration of larger design spaces, in which perhaps the function evaluation is less expensive.

Finally, we demonstrated the effectiveness of our approach on three case studies obtained from real engineering applications. Our results show that we offer better cost-performance trade-offs in comparison to ParEGO and PAL. Across all data sets and almost all desired accuracies, ϵ-PAL outperforms ParEGO, requiring in most cases 30% to 70% less function

evaluations. Moreover, we showed that our parameterization strategy provides a wide range of cost-performance trade-offs. In comparison to PAL, our experiments show that ϵ-PAL reduces the amount of computations by up to 420×, and the number of samples from the design space required to meet the user's desired level of accuracy by up to 2.9×.

Acknowledgments. We would like to thank the authors of (Almer et al., 2011) and (Siegmund et al., 2012) for providing the data for our experiments. This research was supported in part by SNSF grants 200021_137971, 200021_137528, DARPA MSEE FA8650-11-1-7156 and ERC StG 307036.

Appendix A. Theoretical Analysis and Asymptotic Bounds

This section provides the proofs of Theorem 1, which follows from Lemmas 1 to 8.

Lemma 1 *Assume that the target functions f_i, $0 \leq i \leq m$, are bounded on their RKHS norm $||f_i||_k$. Given $\delta \in (0,1)$ and $\beta_t = 2||f_i||_k^2 + 300\gamma_t \log^3(m|E|t/\delta)$, the following holds with probability $\geq 1-\delta$:*

$$\begin{aligned} |f_i(\boldsymbol{x}) - \mu_{t-1,i}(x)| &\leq \beta_t^{1/2}\sigma_{t-1,i}(\boldsymbol{x}) \\ \text{for all } 1 \leq i \leq m,\ \boldsymbol{x} \in E,\ &\text{for all } t \geq 1. \end{aligned} \tag{10}$$

In other words, with probability $\geq 1-\delta$:

$$\boldsymbol{f}(\boldsymbol{x}) \in R_t(\boldsymbol{x}) \text{ for all } \boldsymbol{x} \in E, \text{ for all } t \geq 1$$

Proof According to Theorem 6 in (Srinivas et al., 2012), the following inequality holds:

$$Pr\left\{|f_i(\boldsymbol{x}) - \mu_{t-1,i}(\boldsymbol{x})| > \beta_t^{1/2}\sigma_{t-1,i}(\boldsymbol{x})\right\} \leq \delta',$$

with $\beta_t = 2||f_i||_k^2 + 300\gamma_t \log^3(t/\delta')$. Applying the union bound for $i, t \in \mathbb{N}$, we obtain that the following holds with probability $\geq 1 - m|E|\delta'$:

$$\begin{aligned} |f_i(\boldsymbol{x}) - \mu_{t-1,i}(\boldsymbol{x})| &\leq \beta_t^{1/2}\sigma_{t-1,i}(\boldsymbol{x}) \\ \text{for all } 1 \leq i \leq m,\ &\text{for all } \boldsymbol{x} \in E, \end{aligned} \tag{11}$$

with $\beta_t = 2||f_i||_k^2 + 300\gamma_t \log^3(t/\delta')$. The lemma holds by choosing $\delta = m|E|\delta'$. ■

Lemma 2 *If $m = 1$ and $\boldsymbol{f}_T = (\boldsymbol{f}(\boldsymbol{x}_t))_{1 \leq t \leq T}$, then*

$$I(\boldsymbol{y}_T; \boldsymbol{f}_T) = \frac{1}{2}\sum_{t=1}^{T} \log(1 + \sigma^{-2}\sigma_{t-1}^2(\boldsymbol{x}_t)).$$

This is directly taken from Lemma 5.3 in (Srinivas et al., 2010). Hereby $I(\boldsymbol{y}_T; \boldsymbol{f}_T) = H(\boldsymbol{f}_T) - H(\boldsymbol{f}_T \mid (\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_T, \boldsymbol{y}_T))$ is the mutual information between the function values $\boldsymbol{f}_T$ and the noisy observations $\boldsymbol{y}_T = \boldsymbol{f}_T + \nu_T$ where where $\nu_T \sim N(\boldsymbol{0}, \sigma^2\boldsymbol{I})$. This relation holds no matter in what sequence the samples $\boldsymbol{x}_t$ are picked by $\boldsymbol{\epsilon}$-PAL, and no matter what values $\boldsymbol{y}_t$ are being observed. Latter statement is true since for Gaussian processes the posterior variance over $\boldsymbol{f}$ only depends on where it is evaluated, i.e., $\boldsymbol{x}_T$, not on the actual values $\boldsymbol{y}_T$ observed (Srinivas et al., 2010).

Lemma 3 *Given $\delta \in (0,1)$ and $\beta_t = 2\log(m|E|\pi_t/\delta)$, the following holds:*

$$\sum_{t=1}^{T} \overline{w}_t^2 \leq \beta_T C_1 I(\boldsymbol{y}_T; \boldsymbol{f}_T) \leq C_1 \beta_T \gamma_T \text{ for all } T \geq 1,$$

where $C_1 = 8/\log(1+\sigma^{-2})$.

Proof One of the rectangles of which $R_t(\boldsymbol{x}_t)$ is the intersection has a diagonal length of $2\beta_t^{1/2}\|\boldsymbol{\sigma}_{t-1}(\boldsymbol{x}_t)\|_2$: as a consequence,

$$\overline{w}_t^2 \leq 4\beta_t \|\boldsymbol{\sigma}_{t-1}(\boldsymbol{x}_t)\|_2^2.$$

As β_t is increasing, we have that

$$\begin{aligned} \overline{w}_t^2 &\leq 4\beta_T\sigma^2 \sum_{i=1}^{m} \sigma^{-2}\sigma_{t-1,i}^2(\boldsymbol{x}_t) \\ &\leq 4\beta_T\sigma^2 C_2 \sum_{i=1}^{m} \log(1 + \sigma^{-2}\sigma_{t-1,i}^2(\boldsymbol{x}_t)) \end{aligned}$$

with $C_2 = \sigma^{-2}/\log(1+\sigma^{-2}) \geq 1$, since $s^2 \leq C_2 \log(1+s^2)$ for $0 \leq s \leq \sigma^{-2}$, and $\sigma^{-2}\sigma_{t-1,i}^2(\boldsymbol{x}_t) \leq \sigma^{-2}k_i(\boldsymbol{x}_t, \boldsymbol{x}_t) \leq \sigma^{-2}$.

Using $C_1 = 8\sigma^2 C_2$ and Lemma 2 we have that

$$\begin{aligned} \sum_{t=1}^{T} \overline{w}_t^2 &\leq \beta_T C_1 \sum_{i=1}^{m} I(\boldsymbol{y}_T; f_{T,i}) \\ &\leq \beta_T C_1 I(\boldsymbol{y}_T; \boldsymbol{f}_T) \end{aligned}$$

■

Lemma 4 *Given $\delta \in (0,1)$ and $\beta_t = 2\log(m|E|\pi_t/\delta)$, the following holds with probability $\geq 1-\delta$:*

$$\sum_{t=1}^{T} \overline{w}_t \leq \sqrt{C_1 T \beta_T \gamma_T} \text{ for all } T \geq 1$$

Proof This follows from Lemma 3, since $(\sum_t^T \overline{w}_t)^2 \leq T\sum_{t=1}^T \overline{w}_t^2$ by the Cauchy-Schwarz inequality. ■

Lemma 5 *Running $\boldsymbol{\epsilon}$-PAL with a monotonic classification, it holds that $\overline{w}_t$ decreases with t.*

Proof As a direct consequence of the sample selection rule, $w_{t-1}(\boldsymbol{x}_t) \leq \overline{w}_{t-1}$. On the other hand, $w_t(\boldsymbol{x}) \leq w_{t-1}(\boldsymbol{x})$ and thus, $\overline{w}_t \leq w_{t-1}(\boldsymbol{x}_t)$. The lemma follows. ■

Lemma 6 *Running $\boldsymbol{\epsilon}$-PAL with $\delta \in (0,1)$ and $\beta_t = 2\log(m|E|\pi_t/\delta)$, the following holds:*

$$Pr\left\{\overline{w}_T \leq \sqrt{\frac{C_1\beta_T\gamma_T}{T}} \text{ for all } T \geq 1\right\} \geq 1-\delta, \tag{12}$$

where $C_1 = 8/\log(1+\sigma^{-2})$ and $\pi_t = \pi^2 t^2/6$.

Proof This is derived from Lemmas 4 and 5, since $\sum_{t=1}^{T} \overline{w}_t/T \geq \overline{w}_T$. ■

Lemma 7 *If when running ϵ-PAL, $\overline{w}_t \leq ||\boldsymbol{\epsilon}||_\infty$ holds at iteration t, then the algorithm terminates without further sampling.*

Proof Since $\overline{w}_t$ is un upper bound of $||\max(R_t(\boldsymbol{x})) - \min(R_t(\boldsymbol{x}))||_2$, it is enough to show that the statement is true if $\boldsymbol{w} = \max(R_t(\boldsymbol{x}_i)) - \min(R_t(\boldsymbol{x}_i)) = (\overline{w}_t^{(1)}, \ldots, \overline{w}_t^{(m)})$ for all $\boldsymbol{x}_i \in U_t \cup P_t$. Therefore, $\overline{w}_t \leq ||\boldsymbol{\epsilon}||_\infty$ implies $\boldsymbol{w} \preceq \boldsymbol{\epsilon}$.

We show that if $\boldsymbol{w} \preceq \boldsymbol{\epsilon}$, in the same iteration of $\boldsymbol{\epsilon}$-PAL all points in U_t will be either moved to P_t or discarded.

If a point $\boldsymbol{x}$ does not belong to $Z_\epsilon(E)$, then there is a point $\boldsymbol{x}'$ such that

$$\max(R_t(\boldsymbol{x}')) \succeq \min(R_t(\boldsymbol{x})) + \boldsymbol{\epsilon}.$$

Using $\max(R_t(\boldsymbol{x}')) = \min(R_t(\boldsymbol{x}')) + \boldsymbol{w}$, we can rewrite

$$\min(R_t(\boldsymbol{x})) + \boldsymbol{\epsilon} \preceq \min(R_t(\boldsymbol{x}')) + \boldsymbol{w}.$$

Since $\boldsymbol{w} \preceq \boldsymbol{\epsilon}$, this is equivalent to

$$\min(R_t(\boldsymbol{x})) + \boldsymbol{w} \preceq \min(R_t(\boldsymbol{x}')) + \boldsymbol{\epsilon}.$$

It follows that

$$\max(R_t(\boldsymbol{x})) \preceq \min(R_t(\boldsymbol{x}')) + \boldsymbol{\epsilon},$$

which is the condition that ensures that $\boldsymbol{x} \preceq_\epsilon \boldsymbol{x}'$. This means that there is a point $\boldsymbol{x}' \in U_t \cup P_t$ that dominates $\boldsymbol{x}$, and therefore $\boldsymbol{x}$ will be discarded. ■

Lemma 8 *If ϵ-PAL is run until its termination with some $\boldsymbol{\epsilon}$ and confidence parameter δ, then with probability at least $1-\delta$ it holds that $\Pi(\hat{P})$ is an $\boldsymbol{\epsilon}$-accurate Pareto set of E.*

Proof According to Definition 5, an $\boldsymbol{\epsilon}$-accurate Pareto set of E requires every point of E to have an $\boldsymbol{\epsilon}$-dominator in $\hat{P}$. In the discarding stage of $\boldsymbol{\epsilon}$-PAL, points in U_t are discarded if they are $\boldsymbol{\epsilon}$-dominated by a pessimistic Pareto point. If a point is $\boldsymbol{\epsilon}$-dominated in E, it is $\boldsymbol{\epsilon}$-dominated by a point in the pessimistic Pareto set of E. Therefore, the discarding stage of $\boldsymbol{\epsilon}$-PAL ensures that there will be always be a point in the remaining sets that $\boldsymbol{\epsilon}$-dominates the points that are discarded. Pessimistic Pareto points are only discarded if they are $\boldsymbol{\epsilon}$-dominated by any point in P_t. Hence, all discarding decisions are "safe", in the sense that for every discarded point, there is always at least one $\boldsymbol{\epsilon}$-dominating point remaining. They are also "complete", i.e., only points that belong to $Z_\epsilon(E)$ are moved to P_t. Hence, at termination, since U_t is empty, all remaining points P_t form an $\boldsymbol{\epsilon}$-accurate Pareto set of E. ■

References

O. Almer, N. Topham, and B. Franke. A learning-based approach to the automated design of MPSoC networks. *Architecture of Computing Systems (ARCS)*, pages 243–258, 2011.

R. Bardenet, M. Brendel, B. Kégl, and M. Sebag. Collaborative hyperparameter tuning. In *Intl. Conference on Machine Learning (ICML)*, pages 199–207, 2013.

E. Bonilla, K. M. A. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pages 153–160, 2008.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Arxiv preprint arXiv:1012.2599*, 2010.

D. Buche, N. N. Schraudolph, and P. Koumoutsakos. Accelerating evolutionary algorithms with Gaussian process fitness function models. *IEEE Trans. on Systems, Man, and Cybernetics*, 35:183–194, 2005.

A. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, 2011.

P. Campigotto, A. Passerini, and R Battiti. Active learning of Pareto fronts. *IEEE Trans. on Neural Networks and Learning Systems*, 25:506–519, 2014.

C. Coello, G. B. Lamont, and D. Veldhuizen. *Evolutionary algorithms for solving multi-objective problems*. Springer, 2006.

D. D. Cox and S. John. SDO: A statistical method for global optimization. *Multidisciplinary Design Optimization: State of the Art*, pages 315–329, 1997.

N. de Freitas, A. Smola, and M. Zoghi. Exponential regret bounds for Gaussian process bandits with deterministic observations. In *Intl. Conference on Machine Learning (ICML)*, pages 1743–1750, 2012.

L. Deng, K. Sobti, and C. Chakrabarti. Accurate models for estimating area and power of FPGA implementations. In *Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1417–1420, 2008.

M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Trans. on Evolutionary Computation*, 10(4):421–439, 2006.

M. A. Gelbart, J. Snoek, and R. P. Adams. Bayesian optimization with unknown constraints. In *Uncertainty In Artificial Intelligence (UAI)*, 2014.

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.

J. Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Trans. on Evolutionary Computation*, 10(1):50 – 66, 2006.

A. Krause and C. S. Ong. Contextual Gaussian process bandit optimization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2447–2455, 2011.

H. T. Kung, F. Luccio, and F. P. Preparata. On finding the maxima of a set of vectors. *Journal of the ACM*, 22:469–476, 1975.

S. Künzli, L. Thiele, and E. Zitzler. Modular design space exploration framework for embedded systems. *Computers & Digital Techniques*, 152(2):183–192, 2005.

M. Laumanns and J. Ocenasek. Bayesian optimization algorithms for multi-objective optimization. In *Conference on Parallel Problem Solving from Nature*, pages 298–307. Springer, 2002.

J. Mockus, V. Tiesis, and A. Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

G. Palermo, C. Silvano, and V. Zaccaria. ReSPIR: A response surface-based Pareto iterative refinement for application-specific design space exploration. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 28:1816 –1829, 2009.

C. E. Rasmussen and H. Nickisch. Gaussian process regression and classification toolbox Version 3.2 for Matlab 7.x, 2013.

C. E Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

Burr Settles. Active Learning Literature Survey. Technical Report 1648, University of Wisconsin-Madison, 2010.

N. Siegmund, S.S. Kolesnikov, C. Kastner, S. Apel, D. Batory, M. Rosenmuller, and G. Saake. Predicting performance via automated feature-interaction detection. In *Intl. Conference on Software Engineering (ICSE)*, pages 167–177, 2012.

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Intl. Conference on Machine Learning (ICML)*, pages 1015–1022, 2010.

N. Srinivas, A. Krause, S.M. Kakade, and M. Seeger. Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. *IEEE Trans. on Information Theory*, 58(5):3250–3265, 2012.

I. Steponavice, R. J. Hyndman, Smith-Miles K., and L. Villanova. Efficient identification of the Pareto optimal set. In *Learning and Intelligent Optimization Conference (LION)*, volume 8426 of *Lecture Note of Computer Science*, pages 341–352, 2014.

K. Swersky, J. Snoek, and R. P. Adams. Multi-task Bayesian optimization. In *Advances in Neural Information Processing Systems*, pages 2004–2012, 2013.

E. Vazquez and J. Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and inference*, 140(11):3088–3095, 2010.

Q. Zhang and H. Li. MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. on Evolutionary Computations*, 11:712–731, 2007.

Q. Zhang, L. Wudong, E. Tsang, and B. Virginas. Expensive multiobjective optimization by MOEA/D with Gaussian process model. *IEEE Trans. on Evolutionary Computation*, 14(3):456 –474, 2010.

E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimisation, and Control*, pages 95–100, 2002.

M. Zuluaga, A. Krause, Milder P. A., and M. Püschel. "Smart" design space sampling to predict Pareto-optimal solutions. In *Languages, Compilers, Tools and Theory for Embedded Systems (LCTES)*, pages 119–128, 2012a.

M. Zuluaga, P. Milder, and M. Püschel. Computer generation of streaming sorting networks. In *Design Automation Conference (DAC)*, pages 1245–1253, 2012b.

M. Zuluaga, A. Krause, G. Sergent, and M. Püschel. Active learning for multi-objective optimization. In *Intl. Conference on Machine Learning (ICML)*, pages 462–470, 2013.

M. Zuluaga, A. Krause, and M. Püschel. e-PAL source code, 2015. URL `http://www.spiral.net/software/pal.html`.

Journal of Machine Learning Research 17 (2016) 1-41 Submitted 4/15; Revised 1/16; Published 4/16

Trend Filtering on Graphs

Yu-Xiang Wang[1,2] YUXIANGW@CS.CMU.EDU
James Sharpnack[3] JSHARPNA@UCDAVIS.EDU
Alexander J. Smola[1,4] ALEX@SMOLA.ORG
Ryan J. Tibshirani[1,2] RYANTIBS@STAT.CMU.EDU
[1] *Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA 15213*
[2] *Department of Statistics, Carnegie Mellon University, Pittsburgh, PA 15213*
[3] *Mathematics Department, University of California at San Diego, La Jolla, CA 10280*
[4] *Marianas Labs, Pittsburgh, PA 15213*

Editor: Andreas Krause

Abstract

We introduce a family of adaptive estimators on graphs, based on penalizing the ℓ_1 norm of discrete graph differences. This generalizes the idea of trend filtering (Kim et al., 2009; Tibshirani, 2014), used for univariate nonparametric regression, to graphs. Analogous to the univariate case, graph trend filtering exhibits a level of local adaptivity unmatched by the usual ℓ_2-based graph smoothers. It is also defined by a convex minimization problem that is readily solved (e.g., by fast ADMM or Newton algorithms). We demonstrate the merits of graph trend filtering through both examples and theory.

Keywords: *trend filtering, graph smoothing, total variation denoising, fused lasso, local adaptivity*

1. Introduction

Nonparametric regression has a rich history in statistics, carrying well over 50 years of associated literature. The goal of this paper is to port a successful idea in univariate nonparametric regression, trend filtering (Steidl et al., 2006; Kim et al., 2009; Tibshirani, 2014; Wang et al., 2014), to the setting of estimation on graphs. The proposed estimator, graph trend filtering, shares three key properties of trend filtering in the univariate setting.

1. **Local adaptivity:** graph trend filtering can adapt to inhomogeneity in the level of smoothness of an observed signal across nodes. This stands in contrast to the usual ℓ_2-based methods, e.g., Laplacian regularization (Smola and Kondor, 2003), which enforce smoothness globally with a much heavier hand, and tends to yield estimates that are either smooth or else wiggly throughout.
2. **Computational efficiency:** graph trend filtering is defined by a regularized least squares problem, in which the penalty term is nonsmooth, but convex and structured enough to permit efficient large-scale computation.
3. **Analysis regularization:** the graph trend filtering problem directly penalizes (possibly higher order) differences in the fitted signal across nodes. Therefore graph trend filtering falls into what is called the *analysis* framework for defining estimators. Alternatively, in the *synthesis* framework, we would first construct a suitable basis over the graph, and then regress the observed signal over this basis; e.g., Shuman et al. (2013) survey a number of such approaches using wavelets; likewise, kernel methods regularize in terms of the eigenfunctions of the graph Laplacian (Kondor and Lafferty, 2002). An advantage of analysis regularization is that it easily yields complex extensions of the basic estimator by mixing penalties.

As a motivating example, consider a denoising problem on 402 census tracts of Allegheny County, PA, arranged into a graph with 402 vertices and 2382 edges obtained by connecting spatially adjacent tracts. To illustrate the adaptive property of graph trend filtering we generated an artificial signal with inhomogeneous smoothness across the nodes, and two sharp peaks near the center of the graph, as can be seen in the top left panel of Figure 1. (The signal was formed using a mixture of five Gaussians, in the underlying spatial coordinates.) We drew noisy observations around this signal, shown in the top right panel of the figure, and we fit graph trend filtering, graph Laplacian smoothing, and wavelet smoothing to these observations. Graph trend filtering is to be defined in Section 2 (here we used $k = 2$, quadratic order); the latter two, recall, are defined by the optimization problems

$$\min_{\beta \in \mathbb{R}^n} \|y - \beta\|_2^2 + \lambda \beta^\top L \beta \quad \text{(Laplacian smoothing)},$$
$$\min_{\theta \in \mathbb{R}^n} \frac{1}{2}\|y - W\theta\|_2^2 + \lambda \|\theta\|_1 \quad \text{(wavelet smoothing)},$$

where $y \in \mathbb{R}^n$ the vector of observations measured over the $n = 402$ nodes in the graph, $L \in \mathbb{R}^{n \times n}$ is the graph Laplacian matrix, and $W \in \mathbb{R}^{n \times n}$ is a wavelet basis built over the graph. The wavelet smoothing problem displayed above is really an oversimplified representation of the class of wavelets methods, since it only encapsulates estimators that employ an orthogonal wavelet basis W (and soft-threshold the wavelet coefficients). For the present experiment, we constructed W according to the spanning tree wavelet design of Sharpnack et al. (2013a); we found this construction performed best among the graph wavelet designs we considered for the data at hand. For completeness, the results from alternative wavelet designs are given in the Appendix.

Graph trend filtering, Laplacian smoothing, and wavelet smoothing each have their own regularization parameters λ, and these parameters are not generally on the same scale. Therefore, in our comparisons we use effective degrees of freedom (df) as a common measure for the complexities of the fitted models. The top right panel of Figure 1 shows the graph trend filtering estimate with 68 df. We see that it adaptively fits the sharp peaks in the center of the graph, and smooths out the surrounding regions appropriately. The graph Laplacian estimate with 68 df (bottom left), substantially oversmooths the high peaks in the center, while at 132 df (bottom middle), it begins to detect the high peaks in the center, but undersmooths neighboring regions. Wavelet smoothing performs quite poorly across all df values—it appears to be most affected by the level of noise in the observations.

As a more quantitative assessment, Figure 2 shows the mean squared errors between the estimates and the true underlying signal. The differences in performance here are analogous to the univariate case, when comparing trend filtering to smoothing splines (Tibshirani, 2014). At smaller df values, Laplacian smoothing, due to its global considerations, fails to adapt to local differences across nodes. Trend filtering performs much better at low df values, and yet it matches Laplacian smoothing when both are sufficiently complex, i.e., in the overfitting regime. This demonstrates that the local flexibility of trend filtering estimates is a key attribute.

Here is an outline for the rest of this article. Section 2 defines graph trend filtering and gives underlying motivation and intuition. Section 3 covers basic properties and extensions of the graph

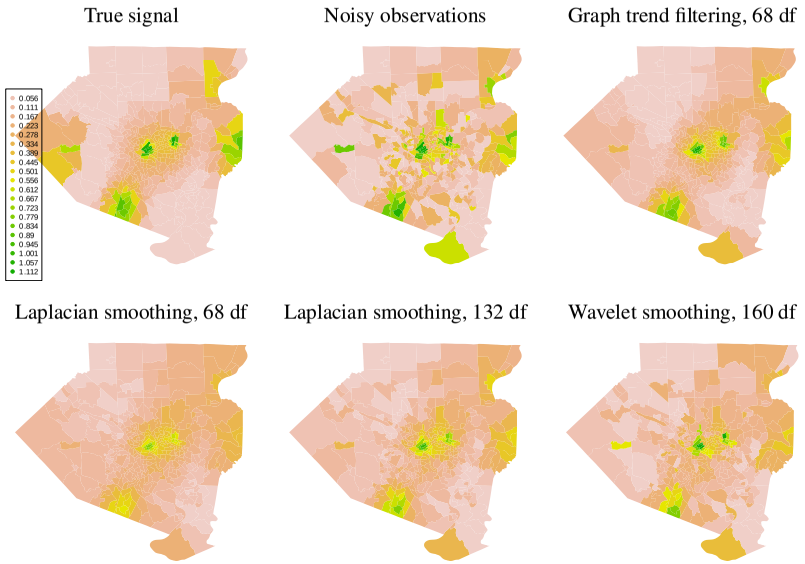


Figure 1: Color maps for the Allegheny County example.

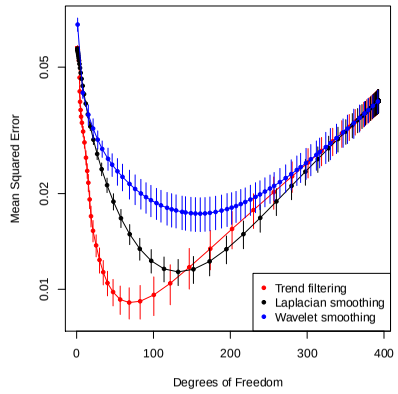


Figure 2: Mean squared errors for the Allegheny County example. Results were averaged over 10 simulations; the bars denote ± 1 standard errors.

trend filtering estimator. Section 4 examines computational approaches, and Section 5 looks at a number of both real and simulated data examples. Section 6 presents asymptotic error bounds for graph trend filtering. Section 7 concludes with a discussion. As for notation, we write X_A to extract the rows of a matrix $X \in \mathbb{R}^{m \times n}$ that correspond to a subset $A \subseteq \{1, \dots m\}$, and X_{-A} to extract the complementary rows. We use a similar convention for vectors: x_A and x_{-A} denote the components of a vector $x \in \mathbb{R}^m$ that correspond to the set A and its complement, respectively. We write $\mathrm{row}(X)$ and $\mathrm{null}(X)$ for the row and null spaces of X, respectively, and $X^\dagger$ for the pseudoinverse of X, with $X^\dagger = (X^\top X)^\dagger X^\top$ when X is rectangular.

2. Trend Filtering on Graphs

In this section, we motivate and formally define graph trend filtering.

2.1 Review: Univariate Trend Filtering

We begin by reviewing trend filtering in the univariate setting, where discrete difference operators play a central role. Suppose that we observe $y = (y_1, \dots y_n) \in \mathbb{R}^n$ across input locations $x = (x_1, \dots x_n) \in \mathbb{R}^n$; for simplicity, suppose that the inputs are evenly spaced, say, $x = (1, \dots n)$. Given an integer $k \geq 0$, the kth order trend filtering estimate $\hat{\beta} = (\hat{\beta}_1, \dots \hat{\beta}_n)$ is defined as

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \ \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|D^{(k+1)}\beta\|_1, \tag{1}$$

where $\lambda \geq 0$ is a tuning parameter, and $D^{(k+1)}$ is the discrete difference operator of order $k+1$. When $k = 0$, problem (1) employs the first difference operator,

$$D^{(1)} = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \vdots & & \ddots & \ddots & \\ 0 & 0 & \dots & -1 & 1 \end{bmatrix}. \tag{2}$$

Therefore $\|D^{(1)}\beta\|_1 = \sum_{i=1}^{n-1} |\beta_{i+1} - \beta_i|$, and the 0th order trend filtering estimate in (1) reduces to the 1-dimensional fused lasso estimator (Tibshirani et al., 2005), also called 1-dimensional total variation denoising (Rudin et al., 1992). For $k \geq 1$ the operator $D^{(k+1)}$ is defined recursively by

$$D^{(k+1)} = D^{(1)} D^{(k)}, \tag{3}$$

with $D^{(1)}$ above denoting the $(n-k-1) \times (n-k)$ version of the first difference operator in (2). In words, $D^{(k+1)}$ is given by taking first differences of kth differences. The interpretation is hence that problem (1) penalizes the changes in the kth discrete differences of the fitted trend. The estimated components $\hat{\beta}_1, \dots \hat{\beta}_n$ exhibit the form of a kth order piecewise polynomial function, evaluated over the input locations $x_1, \dots x_n$. This can be formally verified (Tibshirani, 2014; Wang et al., 2014) by examining a continuous-time analog of (1).

2.2 Trend Filtering over Graphs

Let $G = (V, E)$ be an graph, with vertices $V = \{1, \dots n\}$ and undirected edges $E = \{e_1, \dots e_m\}$, and suppose that we observe $y = (y_1, \dots y_n) \in \mathbb{R}^n$ over the nodes. Following the univariate

definition in (1), we define the kth order *graph trend filtering* (GTF) estimate $\hat{\beta} = (\hat{\beta}_1, \dots \hat{\beta}_n)$ by

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|\Delta^{(k+1)}\beta\|_1. \tag{4}$$

In broad terms, this problem (like univariate trend filtering) is a type of generalized lasso problem (Tibshirani and Taylor, 2011), in which the penalty matrix $\Delta^{(k+1)}$ is a suitably defined *graph difference operator*, of order $k+1$. In fact, the novelty in our proposal lies entirely within the definition of this operator.

When $k = 0$, we define first order graph difference operator $\Delta^{(1)}$ in such a way it yields the graph-equivalent of a penalty on local differences:

$$\|\Delta^{(1)}\beta\|_1 = \sum_{(i,j) \in E} |\beta_i - \beta_j|.$$

so that the penalty term in (4) sums the absolute differences across connected nodes in G. To achieve this, we let $\Delta^{(1)} \in \{-1, 0, 1\}^{m \times n}$ be the oriented incidence matrix of the graph G, containing one row for each edge in the graph; specifically, if $e_\ell = (i, j)$, then $\Delta^{(1)}$ has ℓth row

$$\Delta^{(1)}_\ell = (0, \dots \underset{\uparrow}{-1}_i, \dots \underset{\uparrow}{1}_j, \dots 0), \tag{5}$$

where the orientations of signs are arbitrary. Like trend filtering in the 1d setting, the 0th order graph trend filtering estimate coincides with the fused lasso (total variation regularized) estimate over G (Hoefling, 2010; Tibshirani and Taylor, 2011; Sharpnack et al., 2012).

For $k \geq 1$, we use a recursion to define the higher order graph difference operators, in a manner similar to the univariate case. The recursion alternates in multiplying by the first difference operator $\Delta^{(1)}$ and its transpose (taking into account that this matrix not square):

$$\Delta^{(k+1)} = \begin{cases} (\Delta^{(1)})^\top \Delta^{(k)} = L^{\frac{k+1}{2}} & \text{for odd } k \\ \Delta^{(1)} \Delta^{(k)} = D L^{\frac{k}{2}} & \text{for even } k. \end{cases} \tag{6}$$

Above, we abbreviated the oriented incidence matrix $\Delta^{(1)}$ by D of G, and exploited the fact that $L = D^\top D$ is one representation for the graph Laplacian matrix. Note that $\Delta^{(k+1)} \in \mathbb{R}^{n \times n}$ for odd k, and $\Delta^{(k+1)} \in \mathbb{R}^{m \times n}$ for even k.

An important point is that our defined graph difference operators (5), (6) reduce to the univariate ones (2), (3) in the case of a chain graph (in which $V = \{1, \dots n\}$ and $E = \{(i, i+1) : i = 1, \dots n-1\}$), modulo boundary terms. That is, when k is even, if one removes the first $k/2$ rows and last $k/2$ rows of $\Delta^{(k+1)}$ for the chain graph, then one recovers $D^{(k+1)}$; when k is odd, if one removes the first and last $(k+1)/2$ rows of $\Delta^{(k+1)}$ for the chain graph, then one recovers $D^{(k+1)}$. Further intuition for our graph difference operators is given next.

2.3 Piecewise Polynomials over Graphs

We give some insight for our definition of graph difference operators (5), (6), based on the idea of piecewise polynomials over graphs. In the univariate case, as described in Section 2.1, sparsity of β under the difference operator $D^{(k+1)}$ implies a specific kth order piecewise polynomial structure for

the components of β (Tibshirani, 2014; Wang et al., 2014). Since the components of β correspond to (real-valued) input locations $x = (x_1, \dots x_n)$, the interpretation of a piecewise polynomial here is unambiguous. But for a graph, one might ask: does sparsity of $\Delta^{(k+1)}\beta$ mean that the components of β are piecewise polynomial? And what does the latter even mean, as the components of β are defined over the nodes? To address these questions, we intuitively *define* a piecewise polynomial over a graph, and show that it implies sparsity under our constructed graph difference operators.

- **Piecewise constant ($k = 0$):** we say that a signal β is piecewise constant over a graph G if many of the differences $\beta_i - \beta_j$ are zero across edges $(i, j) \in E$ in G. Note that this is exactly the property associated with sparsity of $\Delta^{(1)}\beta$, since $\Delta^{(1)} = D$, the oriented incidence matrix of G.

- **Piecewise linear ($k = 1$):** we say that a signal β has a piecewise linear structure over G if β satisfies

  $$\beta_i - \frac{1}{n_i} \sum_{(i,j) \in E} \beta_j = 0,$$

  for many nodes $i \in V$, where n_i is the number of nodes adjacent to i. In words, we are requiring that the signal components can be linearly interpolated from its neighboring values at many nodes in the graph. This is quite a natural notion of (piecewise) linearity: requiring that β_i be equal to the average of its neighboring values would enforce linearity at β_i under an appropriate embedding of the points in Euclidean space. Again, this is precisely the same as requiring $\Delta^{(2)}\beta$ to be sparse, since $\Delta^{(2)} = L$, the graph Laplacian.

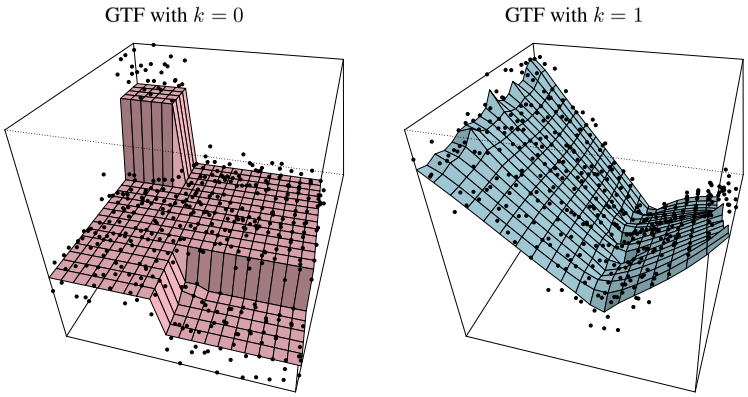
- **Piecewise polynomial ($k \geq 2$):** We say that β has a piecewise quadratic structure over G if the first differences $\alpha_i - \alpha_j$ of the second differences $\alpha = \Delta^{(2)}\beta$ are mostly zero, over edges $(i, j) \in E$. Likewise, β has a piecewise cubic structure over G if the second differences $\alpha_i - \frac{1}{n_i}\sum_{(i,j) \in E} \alpha_j$ of the second differences $\alpha = \Delta^{(2)}\beta$ are mostly zero, over nodes $i \in V$. This argument extends, alternating between leading first and second differences for even and odd k. Sparsity of $\Delta^{(k+1)}\beta$ in either case exactly corresponds to many of these differences being zero, by construction.

In Figure 3, we illustrate the graph trend filtering estimator on a 2d grid graph of dimension 20×20, i.e., a grid graph with 400 nodes and 740 edges. For each of the cases $k = 0, 1, 2$, we generated synthetic measurements over the grid, and computed a GTF estimate of the corresponding order. We chose the 2d grid setting so that the piecewise polynomial nature of GTF estimates could be visualized. Below each plot, the utilized graph trend filtering penalty is displayed in more explicit detail.

2.4 ℓ_1 versus ℓ_2 Regularization

It is instructive to compare the graph trend filtering estimator, as defined in (4), (5), (6) to Laplacian smoothing (Smola and Kondor, 2003). Standard Laplacian smoothing uses the same least squares loss as in (4), but replaces the penalty term with $\beta^\top L \beta$. A natural generalization would be to allow for a power of the Laplacian matrix L, and define kth order graph Laplacian smoothing according to

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \|y - \beta\|_2^2 + \lambda \beta^\top L^{k+1} \beta. \tag{7}$$



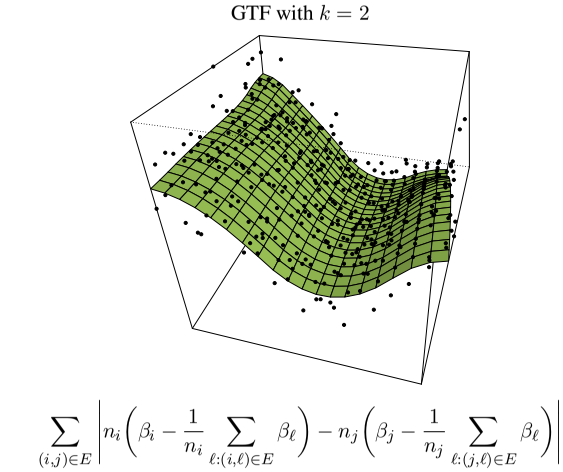


Figure 3: Graph trend filtering estimates of orders $k = 0, 1, 2$ on a 2d grid. The utilized ℓ_1 graph difference penalties are shown in elementwise detail below each plot (first, second, and third order graph differences).

The above penalty term can be written as $\|L^{(k+1)/2}\beta\|_2^2$ for odd k, and $\|DL^{k/2}\beta\|_2^2$ for even k; i.e., this penalty is exactly $\|\Delta^{(k+1)}\beta\|_2^2$ for the graph difference operator $\Delta^{(k+1)}$ defined previously.

As we can see, the critical difference between graph Laplacian smoothing (7) and graph trend filtering (4) lies in the choice of penalty norm: ℓ_2 in the former, and ℓ_1 in the latter. The effect of the ℓ_1 penalty is that the GTF program can set many (higher order) graph differences to zero exactly, and leave others at large nonzero values; i.e., the GTF estimate can simultaneously be smooth in some parts of the graph and wiggly in others. On the other hand, due to the (squared) ℓ_2 penalty, the graph Laplacian smoother cannot set any graph differences to zero exactly, and roughly speaking, must choose between making all graph differences small or large. The relevant analogy here is the comparison between the lasso and ridge regression, or univariate trend filtering and smoothing splines (Tibshirani, 2014), and the suggestion is that GTF can adapt to the proper local degree of smoothness, while Laplacian smoothing cannot. This is strongly supported by the examples given throughout this paper.

2.5 Related Work

Some authors from the signal processing community, e.g., Bredies et al. (2010); Setzer et al. (2011), have studied total generalized variation (TGV), a higher order variant of total variation regularization. Moreover, several discrete versions of these operators have been proposed. They are often similar to the construction that we have. However, the focus of these works is mostly on how well a discrete functional approximates its continuous counterpart. This is quite different from our concern, as a signal on a graph (say a social network) may not have any meaningful continuous-space embedding at all. In addition, we are not aware of any study on the statistical properties of these regularizers. In fact, our theoretical analysis in Section 6 may be extended to these methods too.

3. Properties and Extensions

We first study the structure of graph trend filtering estimates, then discuss interpretations and extensions.

3.1 Basic Structure and Degrees of Freedom

We describe the basic structure of graph trend filtering estimates and present an unbiased estimate for their degrees of freedom. Let the tuning parameter λ be arbitrary but fixed. By virtue of the ℓ_1 penalty in (4), the solution $\hat{\beta}$ satisfies $\text{supp}(\Delta^{(k+1)}\hat{\beta}) = A$ for some active set A (typically A is smaller when λ is larger). Trivially, we can reexpress this as $\Delta^{(k+1)}_{-A}\hat{\beta} = 0$, or $\hat{\beta} \in \text{null}(\Delta^{(k+1)}_{-A})$. Therefore, the basic structure of GTF estimates is revealed by analyzing the null space of the sub-operator $\Delta^{(k+1)}_{-A}$.

Lemma 1 *Assume without a loss of generality that G is connected (otherwise the results apply to each connected component of G). Let D, L be the oriented incidence matrix and Laplacian matrix of G. For even k, let $A \subseteq \{1, \dots m\}$, and let G_{-A} denote the subgraph induced by removing the edges indexed by A (i.e., removing edges e_ℓ, $\ell \in A$). Let $C_1, \dots C_s$ be the connected components of G_{-A}. Then*

$$\text{null}(\Delta^{(k+1)}_{-A}) = \text{span}\{\mathbb{1}\} + (L^\dagger)^{\frac{k}{2}}\text{span}\{\mathbb{1}_{C_1}, \dots \mathbb{1}_{C_s}\},$$

where $\mathbb{1} = (1, \dots 1) \in \mathbb{R}^n$*, and* $\mathbb{1}_{C_1}, \dots \mathbb{1}_{C_s} \in \mathbb{R}^n$ *are the indicator vectors over connected components. For odd* k*, let* $A \subseteq \{1, \dots n\}$*. Then*

$$\mathrm{null}(\Delta_{-A}^{(k+1)}) = \mathrm{span}\{\mathbb{1}\} + \{(L^\dagger)^{\frac{k+1}{2}} v : v_{-A} = 0\}.$$

The proof of Lemma 1 appears in the Appendix. The lemma is useful for a few reasons. First, as motivated above, it describes the coarse structure of GTF solutions. When $k = 0$, we can see (as $(L^\dagger)^{0/2} = I$) that $\hat{\beta}$ will indeed be piecewise constant over groups of nodes $C_1, \dots C_s$ of G. For $k = 2, 4, \dots$, this structure is smoothed by multiplying such piecewise constant levels by $(L^\dagger)^{k/2}$. Meanwhile, for $k = 1, 3 \dots$, the structure of the GTF estimate is based on assigning nonzero values to a subset A of nodes, and then smoothing through multiplication by $(L^\dagger)^{(k+1)/2}$. Both of these smoothing operations, which depend on $L^\dagger$, have interesting interpretations in terms of to the electrical network perspective for graphs. This is developed in the next subsection.

A second use of Lemma 1 is that it leads to a simple expression for the degrees of freedom, i.e., the effective number of parameters, of the GTF estimate $\hat{\beta}$. From results on generalized lasso problems (Tibshirani and Taylor, 2011, 2012), we have $\mathrm{df}(\hat{\beta}) = \mathbb{E}[\mathrm{nullity}(\Delta_{-A}^{(k+1)})]$, with A denoting the support of $\Delta^{(k+1)}\hat{\beta}$, and $\mathrm{nullity}(X)$ the dimension of the null space of a matrix X. Applying Lemma 1 then gives the following.

Lemma 2 *Assume that* G *is connected. Let* $\hat{\beta}$ *denote the GTF estimate at a fixed but arbitrary value of* λ*. Under the normal error model* $y \sim \mathcal{N}(\beta_0, \sigma^2 I)$*, the GTF estimate* $\hat{\beta}$ *has degrees of freedom given by*

$$\mathrm{df}(\hat{\beta}) = \begin{cases} \mathbb{E}\left[\max\{|A|, 1\}\right] & \textit{odd } k, \\ \mathbb{E}\left[\textit{number of connected components of } G_{-A}\right] & \textit{even } k. \end{cases}$$

Here $A = \mathrm{supp}(\Delta^{(k+1)}\hat{\beta})$ *denotes the active set of* $\hat{\beta}$*.*

As a result of Lemma 2, we can form simple unbiased estimate for $\mathrm{df}(\hat{\beta})$; for k odd, this is $\max\{|A|, 1\}$, and for k even, this is the number of connected components of G_{-A}, where A is the support of $\Delta^{(k+1)}\hat{\beta}$. When reporting degrees of freedom for graph trend filtering (as in the example in the introduction), we use these unbiased estimates.

3.2 Electrical Network Interpretation

Lemma 1 reveals a mathematical structure for GTF estimates $\hat{\beta}$, which satisfy $\hat{\beta} \in \mathrm{null}(\Delta_{-A}^{(k+1)})$ for some set A. It is interesting to interpret the results using the electrical network perspective for graphs (Vishnoi, 2012). In this perspective, we imagine replacing each edge in the graph with a resistor of value 1. If $u \in \mathbb{R}^n$ describes how much current is going in at each node in the graph, then $v = Lu$ describes the induced voltage at each node. Provided that $\mathbb{1}^\top c = 0$, which means that the total accumulation of current in the network is 0, we can solve for the current values from the voltage values: $u = L^\dagger v$.

The odd case in Lemma 1 asserts that

$$\mathrm{null}(\Delta_{-A}^{(k+1)}) = \mathrm{span}\{\mathbb{1}\} + \{(L^\dagger)^{\frac{k+1}{2}} v : v_{-A} = 0\}.$$

For $k = 1$, this says that GTF estimates are formed by assigning a sparse number of nodes in the graph a nonzero voltage v, then solving for the induced current $L^\dagger v$ (and shifting this entire current

vector by a constant amount). For $k = 3$, we assign a sparse number of nodes a nonzero voltage, solve for the induced current, and then *repeat this*: we relabel the induced current as input voltages to the nodes, and compute the new induced current. This process is again iterated for $k = 5, 7, \dots$.

The even case in Lemma 1 asserts that

$$\mathrm{null}(\Delta_{-A}^{(k+1)}) = \mathrm{span}\{\mathbb{1}\} + (L^\dagger)^{\frac{k}{2}} \mathrm{span}\{\mathbb{1}_{C_1}, \dots \mathbb{1}_{C_s}\}.$$

For $k = 2$, this result says that GTF estimates are given by choosing a partition $C_1, \dots C_s$ of the nodes, and assigning a constant input voltage to each element of the partition. We then solve for the induced current (and potentially shift this by an overall constant amount). The process is iterated for $k = 4, 6, \dots$ by relabeling the induced current as input voltage.

The comparison between the structure of estimates for $k = 2$ and $k = 3$ is informative: in a sense, the above tells us that 2nd order GTF estimates will be *smoother* than 3rd order estimates, as a sparse input voltage vector need not induce a current that is piecewise constant over nodes in the graph. For example, an input voltage vector that has only a few nodes with very large nonzero values will induce a current that is peaked around these nodes, but not piecewise constant.

3.3 Extensions

Several extensions of the proposed graph trend filtering model are possible. Trend filtering over a weighted graph, for example, could be performed by using a properly weighted version of the edge incidence matrix in (5), and carrying forward the same recursion in (6) for the higher order difference operators. As another example, the Gaussian regression loss in (4) could be changed to another suitable likelihood-derived losses in order to accommodate a different data type for y, say, logistic regression loss for binary data, or Poisson regression loss for count data.

In Section 5.2, we explore a modest extension of GTF, where we add a strongly convex prior term to the criterion in (4) to assist in performing graph-based imputation from partially observed data over the nodes. In Section 5.3, we investigate a modification of the proposed regularization scheme, where we add a pure ℓ_1 penalty on β in (4), hence forming a sparse variant of GTF. Other potentially interesting penalty extensions include: mixing graph difference penalties of various orders, and tying together several denoising tasks with a group penalty. Extensions such as these are easily built, recall, as a result of the analysis framework used by the GTF program, wherein the estimate defined through direct regularization via an analyzing operator, the ℓ_1-based graph difference penalty $\|\Delta^{(k+1)}\beta\|_1$.

4. Computation

Graph trend filtering is defined by a convex optimization problem (4). In principle this means that, at least for small or moderately sized problems, its solutions can be reliably computed using a variety of standard algorithms. In order to handle larger scale problems, we describe two specialized algorithms that improve on generic procedures by taking advantage of the structure of $\Delta^{(k+1)}$.

4.1 A Fast ADMM Algorithm

We reparametrize (4) by introducing auxiliary variables, so that we can apply ADMM. For even k, we use a special transformation that is critical for fast computation (following Ramdas and Tibshirani (2015) in univariate trend filtering); for odd k, this is not possible. The reparametrizations for

even and odd k are

$$\min_{\beta, z \in \mathbb{R}^n} \frac{1}{2}\|y-\beta\|_2^2 + \lambda\|Dz\|_1 \;\text{ s.t. }\; z = L^{\frac{k}{2}} x,$$
$$\min_{\beta, z \in \mathbb{R}^n} \frac{1}{2}\|y-\beta\|_2^2 + \lambda\|z\|_1 \;\text{ s.t. }\; z = L^{\frac{k+1}{2}} x,$$

respectively. Recall that D is the oriented incidence matrix and L is the graph Laplacian. The augmented Lagrangian is

$$\frac{1}{2}\|y-\beta\|_2^2 + \lambda\|Sz\|_1 + \frac{\rho}{2}\|z - L^q\beta + u\|_2^2 - \frac{\rho}{2}\|u\|_2^2,$$

where $S = D$ or $S = I$ depending on whether k is even or odd, and likewise $q = k/2$ or $q = (k+1)/2$. ADMM then proceeds by iteratively minimizing the augmented Lagrangian over β, minimizing over z, and performing a dual update over u. The β and z updates are of the form, for some b,

$$\beta \leftarrow (I + \rho L^{2q})^{-1} b, \tag{8}$$
$$z \leftarrow \operatorname*{argmin}_{x \in \mathbb{R}^n} \frac{1}{2}\|b-x\|_2^2 + \frac{\lambda}{\rho}\|Sx\|_1, \tag{9}$$

The linear system in (8) is well-conditioned, sparse, and can be solved efficiently using the preconditioned conjugate gradient method. This involves only multiplication with Laplacian matrices. For a small enough choices of $\rho > 0$ (the augmented Lagrangian parameter), the system in (8) is diagonally dominant, special Laplacian/SDD solvers can be applied, which run in almost linear time (Spielman and Teng, 2004; Koutis et al., 2011; Kelner et al., 2013).

For $S = I$, the update in (9) is simply given by soft-thresholding, and for $S = D$, it is given by graph TV denoising, i.e., given by solving a graph fused lasso problem. Note that this subproblem has the exact structure of the graph trend filtering problem (4) with $k = 0$. A direct approach for graph TV denoising is available based on parametric max-flow (Chambolle and Darbon, 2009), and this algorithm is empirically much faster than its worst-case complexity (Boykov and Kolmogorov, 2004). In the special case that the underlying graph is a grid, a promising alternative method employs proximal stacking techniques (Barbero and Sra, 2014).

4.2 A Fast Newton Method

As an alternative to ADMM, a projected Newton-type method (Bertsekas, 1982; Barbero and Sra, 2011) can be used to solve (4) via its dual problem:

$$\hat{v} = \operatorname*{argmin}_{v \in \mathbb{R}^r} \|y - (\Delta^{(k+1)})^\top v\|_2^2 \;\text{ s.t. }\; \|v\|_\infty \le \lambda.$$

The solution of (4) is then given by $\hat{\beta} = y - (\Delta^{(k+1)})^\top \hat{v}$. (For univariate trend filtering, Kim et al. (2009) adopt a similar strategy, but instead use an interior point method.) The projected Newton method performs updates using a reduced Hessian, so abbreviating $\Delta = \Delta^{(k+1)}$, each iteration boils down to

$$v \leftarrow a + (\Delta_I^\top)^\dagger b, \tag{10}$$

for some a, b and set of indices I. The linear system in (10) is always sparse, but conditioning becomes an issue as k grows (note that the same problem does not occur in (8) because of the addition of the identity matrix I). We have found empirically that a preconditioned conjugate gradient method works quite well for (10) for $k = 1$, but struggles for larger k.

4.3 Computation Summary

In our experience, the following algorithms work well for the various order k of graph trend filtering. We remark that orders $k = 0, 1, 2$ are of most practical interest (and solutions of polynomial order $k \ge 3$ are less likely to be sought in practice).[1]

Order	Algorithm
$k = 0$	Parametric max-flow
$k = 1$	Projected Newton method
$k = 2, 4, \ldots$	ADMM with parametric max-flow
$k = 3, 5, \ldots$	ADMM with soft-thresholding

Figure 4 compares performances of the described algorithms on a moderately large simulated example, using a 2d grid graph. We see that when $k = 1$, the projected Newton method converges faster than ADMM (superlinear versus at best linear convergence). When $k = 2$, the story is reversed, as the projected Newton iterations quickly become stagnant, and the ADMM enjoys better convergence.

5. Examples

In this section, we present a variety of examples of running graph trend filtering on real graphs.

5.1 Trend Filtering over the Facebook Graph

In the Introduction, we examined the denoising power of graph trend filtering in a spatial setting. Here we examine the behavior of graph trend filtering on a nonplanar graph: the Facebook graph from the Stanford Network Analysis Project (`http://snap.stanford.edu`). This is composed of 4039 nodes representing Facebook users, and 88,234 edges representing friendships, collected from real survey participants; the graph has one connected component, but the observed degree sequence is very mixed, ranging from 1 to 1045 (refer to McAuley and Leskovec (2012) for more details).

We generated synthetic measurements over the Facebook nodes (users) based on three different ground truth models, so that we can precisely evaluate and compare the estimation accuracy of GTF, Laplacian smoothing, and wavelet smoothing. For the latter, we again used the spanning tree wavelet design of Sharpnack et al. (2013a), because it performed among the best of wavelets designs in all data settings considered here. Results from other wavelet designs are presented in

1. Loosely speaking, each order $k = 0, 1, 2$ provides solutions that exhibit a different class of structure: $k = 0$ gives piecewise constant solutions, $k = 1$ gives piecewise linear, and $k = 2$ gives piecewise smooth. All orders $k \ge 3$ continue to give piecewise smooth fits, with less and less transparent differences (the practical differences between piecewise quadratic versus piecewise linear fits is greater than piecewise cubic versus piecewise quadratic, etc.). Since the conditioning of the graph trend filtering operator $\Delta^{(k+1)}$ worsens as k increases, which makes computation more difficult, it makes most practical sense to simply choose $k = 2$ whenever a piecewise smooth fit is desired.

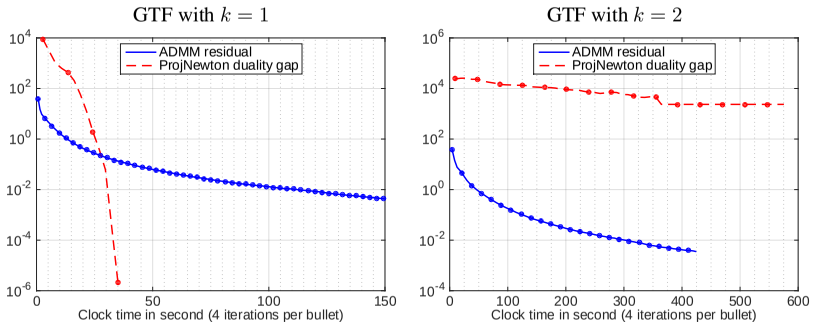


Figure 4: Convergence plots for projected Newton method and ADMM for solving GTF with $k = 1$ and $k = 2$. The algorithms are all run on a 2d grid graph (an 512×512 image) with 262,144 nodes and 523,264 edges. For projected Newton, we plot the duality gap across iterations; for ADMM, we plot the average of the primal and dual residuals (which also serves as a valid suboptimality bound in the ADMM framework).

the Appendix. The three ground truth models represent very different scenarios for the underlying signal x, each one favorable to different estimation methods. These are:

1. **Dense Poisson equation:** we solved the Poisson equation $Lx = b$ for x, where b is arbitrary and dense (its entries were i.i.d. normal draws).
2. **Sparse Poisson equation:** we solved the Poisson equation $Lx = b$ for x, where b is sparse and has 30 nonzero entries (again i.i.d. normal draws).
3. **Inhomogeneous random walk:** we ran a set of decaying random walks at different starter nodes in the graph, and recorded in x the total number of visits at each node. Specifically, we chose 10 nodes as starter nodes, and assigned each starter node a decay probability uniformly at random between 0 and 1 (this is the probability that the walk terminates at each step instead of travelling to a neighboring node). At each starter node, we also sent out a varying number of random walks, chosen uniformly between 0 and 1000.

In each case, the synthetic measurements were formed by adding noise to x. We note that model 1 is designed to be favorable for Laplace smoothing; model 2 is designed to be favorable for GTF; and in the inhomogeneity in model 3 is designed to be challenging for Laplacian smoothing, and favorable for the more adaptive GTF and wavelet methods.

Figure 5 shows the performance of the three estimation methods, over a wide range of noise levels in the synthetic measurements; performance here is measured by the best achieved mean squared error, allowing each method to be tuned optimally at each noise level. The summary: GTF estimates are (expectedly) superior when the Laplacian-based sparsity pattern is in effect (model 2), but are nonetheless highly competitive in both other settings—the dense case, in which Laplacian smoothing thrives, and the inhomogeneous random walk case, in which wavelets thrive.

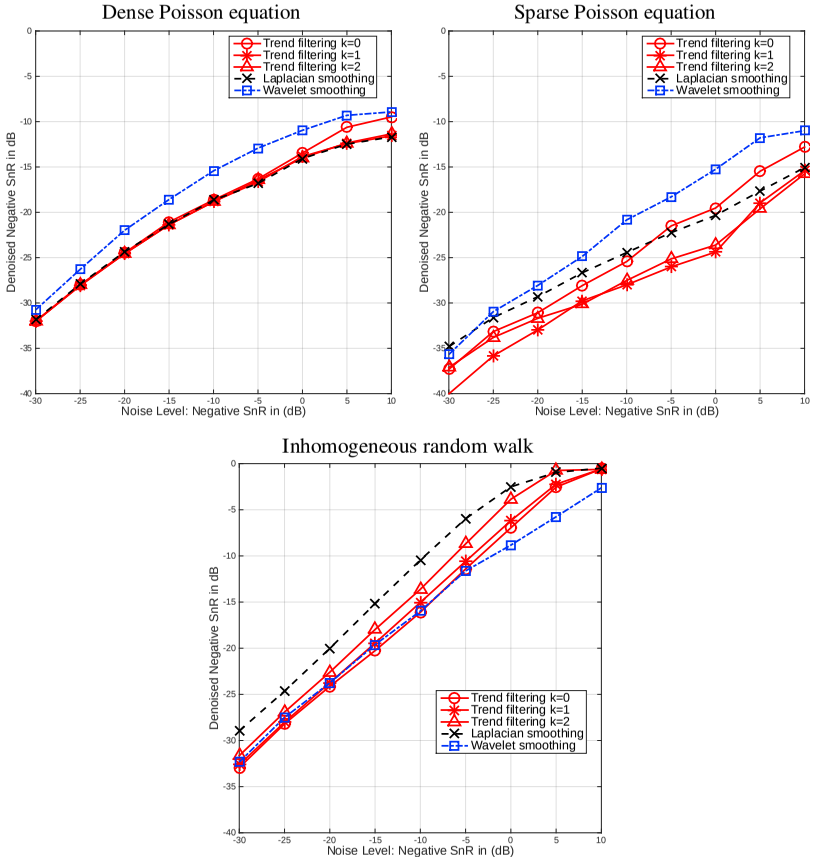


Figure 5: Performance of GTF and others for three generative models on the Facebook graph. The x-axis shows the negative SnR: $10 \log_{10}(n\sigma^2/\|x\|_2^2)$, where $n = 4039$, x is the underlying signal, and σ^2 is the noise variance. Hence the noise level is increasing from left to right. The y-axis shows the denoised negative SnR: $10 \log_{10}(\text{MSE}/\|x\|_2^2)$, where MSE denotes mean squared error, so the achieved MSE is increasing from bottom to top.

5.2 Graph-Based Transductive Learning over UCI Data

Graph trend filtering can used for graph-based transductive learning, as motivated by the work of Talukdar and Crammer (2009); Talukdar and Pereira (2010), who rely on Laplacian regularization. Consider a semi-supervised learning setting, where we are given only a small number of seed labels over nodes of a graph, and the goal is to impute the labels on the remaining nodes. Write $O \subseteq \{1, \dots n\}$ for the set of observed nodes, and assume that each observed label falls into $\{1, \dots K\}$. Then we can define the modified absorption problem under graph trend filtering regularization (MAD-GTF) by

$$\hat{B} = \underset{B \in \mathbb{R}^{n \times K}}{\operatorname{argmin}} \sum_{j=1}^{K} \sum_{i \in O} (Y_{ij} - B_{ij})^2 + \lambda \sum_{j=1}^{K} \|\Delta^{(k+1)} B_j\|_1 + \epsilon \sum_{j=1}^{K} \|R_j - B_j\|_2^2. \tag{11}$$

The matrix $Y \in \mathbb{R}^{n \times K}$ is an indicator matrix: each observed row $i \in O$ is described by $Y_{ij} = 1$ if class j is observed and $Y_{ij} = 0$ otherwise. The matrix $B \in \mathbb{R}^{n \times K}$ contains fitted probabilities, with B_{ij} giving the probability that node i is of class j. We write B_j for its jth column, and hence the middle term in the above criterion encourages each set of class probabilities to behave smoothly over the graph. The last term in the above criterion ties the fitted probabilities to some given prior weights $R \in \mathbb{R}^{n \times K}$. In principle ϵ could act as a second tuning parameter, but for simplicity we take ϵ to be small and fixed, with any $\epsilon > 0$ guaranteeing that the criterion in (11) is strictly convex, and thus has a unique solution $\hat{B}$. The entries of $\hat{B}$ need not be probabilites in any strict sense, but we can still interpret them as relative probabilities, and imputation can be performed by assigning each unobserved node $i \notin O$ a class label j such that $\hat{B}_{ij}$ is largest.

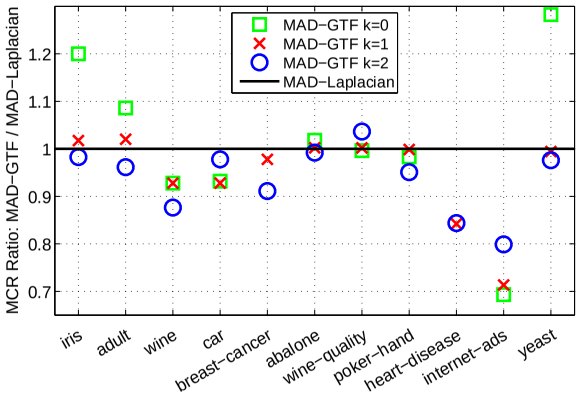


Figure 6: Ratio of the misclassification rate of MAD-GTF to MAD-Laplacian, for graph-based imputation, on the 11 most popular UCI classification data sets.

Our specification of MAD-GTF only deviates from the MAD proposal of Talukdar and Crammer (2009) in that these authors used the Laplacian regularization term $\sum_{j=1}^{K} B_j^\top L B_j$, in place of ℓ_1-based graph difference regularizer in (11). If the underlying class probabilities are thought

	iris	adult	wine	car	breast	abalone	wine-qual.	poker	heart	ads	yeast
# of classes	3	2	3	4	2	29	6	10	2	2	10
# of samples	150	32,561	178	1,728	569	4,177	1,599	3,000	303	3,279	1,484
Laplacian	0.085	0.270	0.060	0.316	0.064	0.872	0.712	0.814	0.208	0.306	0.566
GTF, $k = 0$	0.102	0.293	0.055	0.294	0.500	0.888	0.709	0.801	0.472	0.212	0.726
p-value	0.254	0.648	0.406	0.091	0.000	0.090	0.953	0.732	0.000	0.006	0.000
GTF, $k = 1$	0.087	0.275	0.055	0.293	0.063	0.874	0.713	0.813	0.175	0.218	0.563
p-value	0.443	0.413	0.025	0.012	0.498	0.699	0.920	0.801	0.134	0.054	0.636
GTF, $k = 2$	0.084	0.259	0.052	0.309	0.059	0.865	0.738	0.774	0.175	0.244	0.552
p-value	0.798	0.482	0.024	0.523	0.073	0.144	0.479	0.138	0.301	0.212	0.100

Table 1: Misclassification rates of MAD-Laplacian and MAD-GTF for imputation in the UCI data sets. We also compute p-values over the 10 repetitions for each data set (10 draws of nodes to serve as seed labels) via paired t-tests. Cases where MAD-GTF achieves significantly better misclassification rate, at the 0.1 level, are highlighted in green; cases where MAD-GTF achieves a significantly worse miclassification rate, at the 0.1 level, are highlighted in red.

to have heterogeneous smoothness over the graph, then replacing the Laplacian regularizer with the GTF-designed one might lead to better performance. As a broad comparison of the two methods, we ran them on the 11 most popular classification data sets from the UCI Machine Learning repository (`http://archive.ics.uci.edu/ml/`).[2] For each data set, we constructed a 5-nearest-neighbor graph based on the Euclidean distance between provided features, and randomly selected 5 seeds per class to serve as the observed class labels. Then we set $\epsilon = 0.01$, used prior weights $R_{ij} = 1/K$ for all i and j, and chose the tuning parameter λ over a wide grid of values to represent the best achievable performance by each method, on each experiment. Figure 6 and Table 1 summarize the misclassification rates from imputation using MAD-Laplacian and MAD-GTF, averaged over 10 repetitions of the randomly selected seed labels. We see that MAD-GTF with $k = 0$ (basically a graph partition akin to MRF-based graph cut, using an Ising model) does not seem to work as well as the smoother alternatives. Importantly, MAD-GTF with $k = 1$ and $k = 2$ both perform at least as well, and sometimes better, than MAD-Laplacian on each one of the UCI data sets. Recall that these data sets were selected entirely based on their popularity, and not at all on the belief that they might represent favorable scenarios for GTF (i.e., not on the prospect that they might exhibit some heterogeneity in the distribution of class labels over their respective graphs). Therefore, the fact that MAD-GTF nonetheless performs competitively in such a broad range of experiments is convincing evidence for the utility of the GTF regularizer.

5.3 Event Detection with NYC Taxi Trips Data

We illustrate a sparse variant of our proposed regularizers, given by adding a pure ℓ_1 penalty to the coefficients in (4), as in

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2} \|y - \beta\|_2^2 + \lambda_1 \|\Delta^{(k+1)} \beta\|_1 + \lambda_2 \|\beta\|_1. \tag{12}$$

We call this *sparse graph trend filtering*, now with two tuning parameters $\lambda_1, \lambda_2 \geq 0$. Under the proper tuning, the sparse GTF estimate will be zero at many nodes in the graph, and will otherwise

2. We used all data sets here, except the "forest-fires" data set, which is a regression problem. Also, we zero-filled the missing data in "internet-ads" data set and used a random one third of the data in the "poker" data set.

deviate smoothly from zero. This can be useful in situations where the observed signal represents a difference between two smooth processes that are mostly similar, but exhibit (perhaps significant) differences over a few regions of the graph. Here we apply it to the problem of detecting events based on abnormalities in the number of taxi trips at different locations of New York city. This data set was kindly provided by authors of Doraiswamy et al. (2014), who obtained the data from NYC Taxi & Limosine Commission.[3] Specifically, we consider the graph to be the road network of Manhattan, which contains 3874 nodes (junctions) and 7070 edges (sections of roads that connect two junctions). For measurements over the nodes, we used the number of taxi pickups and dropoffs over a particular time period of interest: 12:00–2:00 pm on June 26, 2011, corresponding to the Gay Pride parade. As pickups and dropoffs do not generically occur at road junctions, we used interpolation to form counts over the graph nodes. A baseline seasonal average was calculated by considering data from the same time block 12:00–2:00 pm on the same day of the week across the nearest eight weeks. Thus the measurements y were then taken to be the difference between the counts observed during the Gay Pride parade and the seasonal averages.

Note that the nonzero node estimates from sparse GTF applied to y, after proper tuning, mark events of interest, because they convey substantial differences between the observed and expected taxi counts. According to descriptions in the news, we know that the Gay Pride parade was a giant march down at noon from 36th St. and Fifth Ave. all the way to Christopher St. in Greenwich Village, and traffic was blocked over the entire route for two hours (meaning no pickups and dropoffs could occur). We therefore hand-labeled this route as a crude "ground truth" for the event of interest, illustrated in the left panel of Figure 7.

In the bottom two panels of Figure 7, we compare sparse GTF with $k = 0$ (i.e., the sparse graph fused lasso) and a sparse variant of Laplacian smoothing, obtained by replacing the first regularization term in (12) by $\beta^\top L\beta$. For a qualitative visual comparison, the smoothing parameter λ_1 was chosen so that both methods have 200 degrees of freedom (without any sparsity imposed). The sparsity parameter was then set as $\lambda_2 = 0.2$. Similar to what we have seen already, GTF is able to better localize its estimates around strong inhomogenous spikes in the measurements, and is able to better capture the event of interest. The result of sparse Laplacian smoothing is far from localized around the ground truth event, and displays many nonzero node estimates throughout distant regions of the graph. If we were to decrease its flexibility (increase the smoothing parameter λ_1 in its problem formulation), then the sparse Laplacian output would display more smoothness over the graph, but the node estimates around the ground truth region would also be grossly shrunken.

6. Estimation Error Bounds

In this section, we assume that $y \sim \mathcal{N}(\beta_0, \sigma^2 I)$, and study asymptotic error rates for graph trend filtering. (The assumption of a normal error model could be relaxed, but is used for simplicity). Our analysis actually focuses more broadly on the generalized lasso problem

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \ \frac{1}{2}\|y - \beta\|_2^2 + \lambda\|\Delta\beta\|_1, \tag{13}$$

3. These authors also considered event detection, but their topological definition of an "event" is very different from what we considered here, and hence our results not directly comparable.

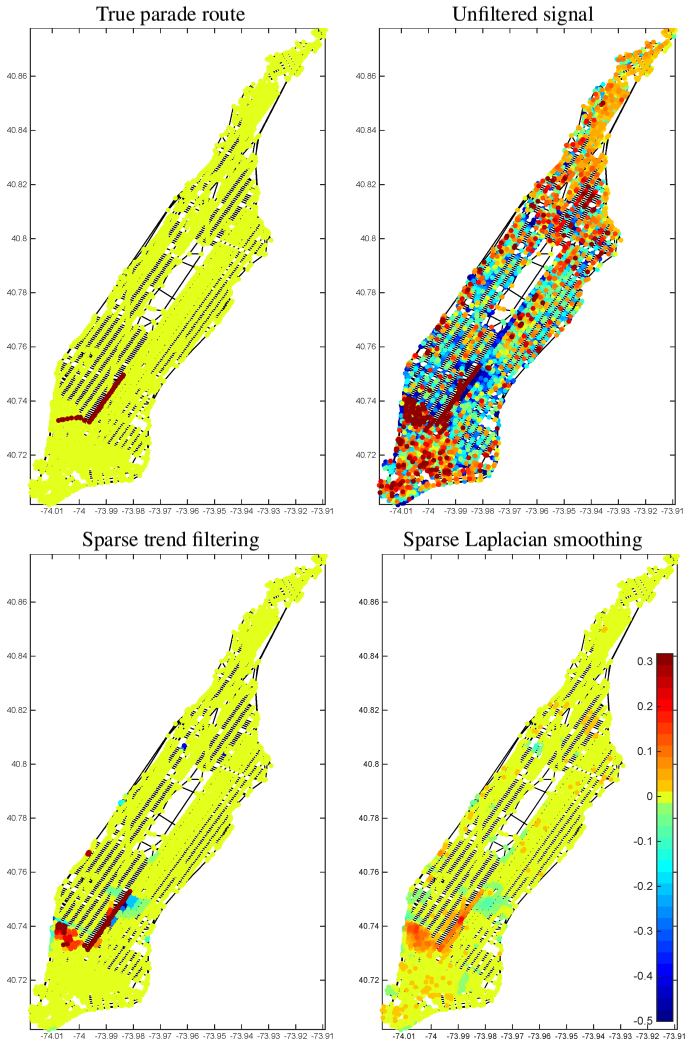


Figure 7: Comparison of sparse GTF and sparse Laplacian smoothing. We can see qualitatively that sparse GTF delivers better event detection with fewer false positives (zoomed-in, the sparse Laplacian plot shows a scattering of many non-zero colors).

where $\Delta \in \mathbb{R}^{r \times n}$ is an arbitrary linear operator, and r denotes its number of rows. Throughout, we specialize the derived results to the graph difference operator $\Delta = \Delta^{(k+1)}$, to obtain concrete statements about GTF over particular graphs. All proofs are deferred to the Appendix.

6.1 Basic Error Bounds

Using similar arguments to the basic inequality for the lasso (Buhlmann and van de Geer, 2011), we have the following preliminary bound.

Theorem 3 *Let M denote the maximum ℓ_2 norm of the columns of $\Delta^\dagger$. Then for a tuning parameter value $\lambda = \Theta(M\sqrt{\log r})$, the generalized lasso estimate $\hat{\beta}$ in (13) has average squared error*

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\mathrm{nullity}(\Delta)}{n} + \frac{M\sqrt{\log r}}{n} \cdot \|\Delta\beta_0\|_1\right).$$

Recall that $\mathrm{nullity}(\Delta)$ denotes the dimension of the null space of Δ. For the GTF operator $\Delta^{(k+1)}$ of any order k, note that $\mathrm{nullity}(\Delta^{(k+1)})$ is the number of connected components in the underlying graph.

When both $\|\Delta\beta_0\|_1 = O(1)$ and $\mathrm{nullity}(\Delta) = O(1)$, Theorem 3 says that the estimate $\hat{\beta}$ converges in average squared error at the rate $M\sqrt{\log r}/n$, in probability. This theorem is quite general, as it applies to any linear operator Δ, and one might therefore think that it cannot yield fast rates. Still, as we show next, it does imply consistency for graph trend filtering in certain cases.

Corollary 4 *Consider the trend filtering estimator $\hat{\beta}$ of order k, and the choice of the tuning parameter λ as in Theorem 3. Then:*

1. *for univariate trend filtering (i.e., essentially GTF on a chain graph),*

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\sqrt{\frac{\log n}{n}} \cdot n^k \|D^{(k+1)}\beta_0\|_1\right);$$

2. *for GTF on an Erdos-Renyi random graph, with edge probability p, and expected degree $d = np \geq 1$,*

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\sqrt{\log(nd)}}{nd^{\frac{k+1}{2}}} \cdot \|\Delta^{(k+1)}\beta_0\|_1\right);$$

3. *for GTF on a Ramanujan d-regular graph, and $d \geq 1$,*

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\sqrt{\log(nd)}}{nd^{\frac{k+1}{2}}} \cdot \|\Delta^{(k+1)}\beta_0\|_1\right).$$

Cases 2 and 3 of Corollary 4 stem from the simple inequality $M \leq \|\Delta^\dagger\|_2$, the largest singular value of $\Delta^\dagger$. When $\Delta = \Delta^{(k+1)}$, the GTF operator of order $k+1$, we have

$$\|(\Delta^{(k+1)})^\dagger\|_2 \leq 1/\lambda_{\min}(L)^{(k+1)/2},$$

where $\lambda_{\min}(L)$ is the smallest nonzero eigenvalue of the Laplacian L (also known as the Fiedler eigenvalue (Fiedler, 1973)). In general, $\lambda_{\min}(L)$ can be very small, leading to loose error bounds,

but for the particular graphs in question, it is well-controlled. When $\|\Delta^{(k+1)}\beta_0\|_1$ is bounded, cases 2 and 3 of the corollary show that the average squared error of GTF converges at the rate $\sqrt{\log(nd)}/(nd^{(k+1)/2})$. As k increases, this rate is stronger, but so is the assumption that $\|\Delta^{(k+1)}\beta_0\|_1$ is bounded.

Case 1 in Corollary 4 covers univariate trend filtering (which, recall, is basically the same as GTF over a chain graph; the only differences between the two are boundary terms in the construction of the difference operators). The result in case 1 is based on direct calculation of M, using specific facts that are known about the univariate difference operators. It is natural in the univariate setting to assume that $n^k\|D^{(k+1)}\beta_0\|_1$ is bounded (this is the scaling that would link β_0 to the evaluations of a piecewise polynomial function f_0 over $[0,1]$, with $\mathrm{TV}(f_0^{(k)})$ bounded). Under such an assumption, the above corollary yields a convergence rate of $\sqrt{\log n/n}$ for univariate trend filtering, which is not tight. A more refined analysis shows the univariate trend filtering estimator to converge at the minimax optimal rate $n^{-(2k+2)/(2k+3)}$, proved in Tibshirani (2014) by using a connection between univariate trend filtering and locally adaptive regression splines, and relying on sharp entropy-based rates for locally adaptive regression splines from Mammen and van de Geer (1997). We note that in a pure graph-centric setting, the latter strategy is not generally applicable, as the notion of a spline function does not obviously extend to the nodes of an arbitrary graph structure.

In the next subsections, we develop more advanced strategies for deriving fast GTF error rates, based on incoherence, and entropy. These can provide substantial improvements over the basic error bound established in this subsection, but are only applicable to certain graph models. Fortunately, this includes common graphs of interest, such as regular grids. To verify the sharpness of these alternative strategies, we will show that they can be used to recover optimal rates of convergence for trend filtering in the 1d setting.

6.2 Strong Error Bounds Based on Incoherence

A key step in the proof of Theorem 3 argues, roughly speaking, that

$$\epsilon^\top \Delta^\dagger \Delta x \leq \|(\Delta^\dagger)^\top \epsilon\|_\infty \|\Delta x\|_1 = O_{\mathbb{P}}(M\sqrt{\log r}\|\Delta x\|_1), \tag{14}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. The second bound holds by a standard result on maxima of Gaussians (recall that M is largest ℓ_2 norm of the columns of $\Delta^\dagger$). The first bound above uses Holder's inequality; note that this applies to any ϵ, Δ, i.e., it does not use any information about the distribution of ϵ, or the properties of Δ. The next lemma reveals a potential advantage that can be gained from replacing the bound (14), stemming from Holder's inequality, with a "linearized" bound.

Lemma 5 *Denote $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, and assume that*

$$\max_{x \in \mathcal{S}_\Delta(1)} \frac{\epsilon^\top x - A}{\|x\|_2} = O_{\mathbb{P}}(B), \tag{15}$$

where $\mathcal{S}_\Delta(1) = \{x \in \mathrm{row}(\Delta) : \|\Delta x\|_1 \leq 1\}$. With $\lambda = \Theta(A)$, the generalized lasso estimate $\hat{\beta}$ satisfies

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\mathrm{nullity}(\Delta)}{n} + \frac{B^2}{n} + \frac{A}{n} \cdot \|\Delta\beta_0\|_1\right).$$

The inequality in (15) is referred to as a "linearized" bound because it implies that for $x \in \mathcal{S}_\Delta(1)$,

$$\epsilon^\top x = O_{\mathbb{P}}(A + B\|x\|_2),$$

and the right-hand side is a linear function of $\|x\|_2$. Indeed, for $A = M\sqrt{2\log r}$ and $B = 0$, this encompasses the bound in (14) as a special case, and the result of Lemma 5 reduces to that of Theorem 3. But the result in Lemma 5 can be much stronger, if A, B can be adjusted so that A is smaller than $M\sqrt{2\log r}$, and B is also small. Such an arrangement is possible for certain operators Δ; e.g., it is possible under an incoherence-type assumption on Δ.

Theorem 6 *Let $q = \mathrm{rank}(\Delta)$, and let $\xi_1 \leq \ldots \leq \xi_q$ denote the singular values of Δ, in increasing order. Also let $u_1, \ldots u_q$ be the corresponding left singular vectors. Assume that these vectors are incoherent:*

$$\|u_i\|_\infty \leq \mu/\sqrt{n}, \quad i = 1, \ldots q,$$

for some constant $\mu \geq 1$. For $i_0 \in \{1, \ldots q\}$, let

$$\lambda = \Theta\left(\mu\sqrt{\frac{\log r}{n}\sum_{i=i_0+1}^{q}\frac{1}{\xi_i^2}}\right).$$

Then the generalized lasso estimate $\hat{\beta}$ has average squared error

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\mathrm{nullity}(\Delta)}{n} + \frac{i_0}{n} + \frac{\mu}{n}\sqrt{\frac{\log r}{n}\sum_{i=i_0+1}^{q}\frac{1}{\xi_i^2}} \cdot \|\Delta\beta_0\|_1\right).$$

Theorem 6 is proved by leveraging the linearized bound (15), which holds under the incoherence condition on the singular vectors of Δ. Compared to the basic result in Theorem 3, the result in Theorem 6 is clearly stronger as it allows us to replace M—which can grow like the reciprocal of the minimum nonzero singular value of Δ—with something akin to the average reciprocal of larger singular values. But it does, of course, also make stronger assumptions (incoherence). It is interesting to note that the functional in the theorem, $\sum_{i=i_0+1}^{q}\xi_i^{-2}$, was also determined to play a leading role in error bounds for a graph Fourier based scan statistic in the hypothesis testing framework (Sharpnack et al., 2013b).

Applying the above theorem to the GTF estimator requires knowledge of the singular vectors of $\Delta = \Delta^{(k+1)}$, the $(k+1)$st order graph difference operator. The validity of an incoherence assumption on these singular vectors depend on the graph G in question. When k is odd, these singular vectors are eigenvectors of the Laplacian L; when k is even, they are left singular vectors of the edge incidence matrix D. Loosely speaking, these vectors will be incoherent when neighborhoods of different vertices look roughly the same. Most social networks will have this property for the bulk of their vertices (i.e., with the exception of a small number of high degree vertices). Grid graphs also have this property. First, we consider trend filtering over a 1d grid, i.e., a chain (which, recall, is essentially equivalent to univariate trend filtering).

Corollary 7 *Consider the GTF estimator $\hat{\beta}$ of order k, over a chain graph, i.e., a 1d grid graph. Letting*

$$\lambda = \Theta\left(n^{\frac{2k+1}{2k+3}}(\log n)^{\frac{1}{2k+3}}\|\Delta^{(k+1)}\beta_0\|_1^{-\frac{2k+1}{2k+3}}\right),$$

the estimator $\hat{\beta}$ (here, essentially, the univariate trend filtering estimator) satisfies

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(n^{-\frac{2k+2}{2k+3}}(\log n)^{\frac{1}{2k+3}} \cdot \left(n^k\|\Delta^{(k+1)}\beta_0\|_1\right)^{\frac{2}{2k+3}}\right).$$

We note that the above corollary essentially recovers the optimal rate of convergence for the univariate trend filtering estimator, for all orders k. (To be precise, it studies GTF on a chain graph instead, but this is basically the same problem.) When $n^k\|\Delta^{(k+1)}\beta_0\|_1$ is assumed to be bounded, a natural assumption in the univariate setting, the corollary shows the estimator to converge at the rate $n^{-(2k+2)/(2k+3)}(\log n)^{1/(2k+3)}$. Ignoring the log factor, this matches the minimax optimal rate as established in Tibshirani (2014); Wang et al. (2014). Importantly, the proof of Corollary 7, unlike that used in previous works, is free from any dependence on univariate spline functions; it is completely graph-theoretic, and only uses on the incoherence properties of the 1d grid graph. The strength of this approach is its wider applicability, which we demonstrate by moving up to 2d grids.

Corollary 8 *Consider the GTF estimator $\hat{\beta}$ of order k, over a 2d grid graph, of size $\sqrt{n} \times \sqrt{n}$. Letting*

$$\lambda = \Theta\left(n^{\frac{2k+1}{2k+5}}(\log n)^{\frac{1}{2k+5}}\|\Delta^{(k+1)}\beta_0\|_1^{-\frac{2k+1}{2k+5}}\right),$$

the estimator $\hat{\beta}$ satisfies

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(n^{-\frac{2k+4}{2k+5}}(\log n)^{\frac{1}{2k+5}} \cdot \left(n^{\frac{k}{2}}\|\Delta^{(k+1)}\beta_0\|_1\right)^{\frac{4}{2k+5}}\right).$$

The 2d result in Corollary 8 is written in a form that mimics the 1d result in Corollary 7, as we claim that the analog of boundedness of $n^k\|\Delta^{(k+1)}\beta_0\|_1$ in 1d is boundedness of $n^{k/2}\|\Delta^{(k+1)}\beta_0\|_1$ in 2d.[4] Thus, under the appropriate boundedness condition, the 2d rate shows improvement over the 1d rate, which makes sense, since regularization here is being enforced in a richer manner. It is worthwhile highlighting the result for $k = 0$ in particular: this says that, when the sum of absolute discrete differences $\|\Delta^{(1)}\beta_0\|_1$ is bounded over a 2d grid, the 2d fused lasso (i.e., 2d total variation denoising) has error rate $n^{-4/5}$. This is faster than the $n^{-2/3}$ rate for the 1d fused lasso, when the sum of absolute differences $\|D^{(1)}\beta_0\|_1$ is bounded. Rates for higher dimensional grid graphs (for all k) follow from analogous arguments, but we omit the details.

6.3 Strong Error Bounds Based on Entropy

A different "fractional" bound on the Gaussian contrast $\epsilon^\top x$, over $x \in \mathcal{S}_\Delta(1)$, provides an alternate route to deriving sharper rates. This style of bound is inspired by the seminal work of van de Geer (1990).

Lemma 9 *Denote $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, and assume that for a constant $w < 2$,*

$$\max_{x \in \mathcal{S}_\Delta(1)} \frac{\epsilon^\top x}{\|x\|_2^{1-w/2}} = O_{\mathbb{P}}(K), \tag{16}$$

where recall $\mathcal{S}_\Delta(1) = \{x \in \mathrm{row}(\Delta) : \|\Delta x\|_1 \leq 1\}$. Then with

$$\lambda = \Theta\left(K^{\frac{2}{1+w/2}} \cdot \|\Delta\beta_0\|_1^{-\frac{1-w/2}{1+w/2}}\right),$$

4. This is because $1/\sqrt{n}$ is the distance between adjacent 2d grid points, when viewed as a 2d lattice over $[0,1]^2$.

the generalized lasso estimate $\hat{\beta}$ satisfies

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\text{nullity}(\Delta)}{n} + \frac{K^{\frac{2}{1+w/2}}}{n} \cdot \|\Delta\beta_0\|_1^{\frac{w}{1+w/2}}\right).$$

The main motivation for bounds of the form (16) is that they follow from entropy bounds on the set $\mathcal{S}_\Delta(1)$. Recall that for a set S, the covering number $N(\delta, S, \|\cdot\|)$ is the fewest number of balls of radius δ that cover S, with respect to the norm $\|\cdot\|$. The log covering or entropy number is $\log N(\delta, S, \|\cdot\|)$. In the next result, we make the connection between between entropy and fractional bounds precise; this follows closely from Lemma 3.5 of van de Geer (1990).

Theorem 10 *Suppose that there exist a constant $w < 2$ such that for n large enough,*

$$\log N(\delta, \mathcal{S}_\Delta(1), \|\cdot\|_2) \leq E\Big(\frac{\sqrt{n}}{\delta}\Big)^w, \tag{17}$$

for $\delta > 0$, where E can depend on n. Then the fractional bound in (16) *holds with $K = \sqrt{E}n^{w/4}$, and as a result, for*

$$\lambda = \Theta\left(E^{\frac{1}{1+w/2}} n^{\frac{w/2}{1+w/2}} \|\Delta\beta_0\|_1^{-\frac{1-w/2}{1+w/2}}\right),$$

the generalized lasso estimate $\hat{\beta}$ has average squared error

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\text{nullity}(\Delta)}{n} + E^{\frac{1}{1+w/2}} n^{-\frac{1}{1+w/2}} \cdot \|\Delta\beta_0\|_1^{\frac{w}{1+w/2}}\right).$$

To make use of the result in Theorem 10, we must obtain an entropy bound as in (17), on the set $\mathcal{S}_\Delta(1)$. The literature on entropy numbers is rich, and there are various methods for computing entropy bounds, any of which can be used for these purposes as long as the bounds fit the form of (17), as required by the theorem. For bounding the entropy of a set like $\mathcal{S}_\Delta(1)$, two common techniques are to use a characterization of the spectral decay of $\Delta^\dagger$, or an analysis of the correlations between columns of $\Delta^\dagger$. For a nice review of such strategies and their applications, we refer the reader to Section 6 of van de Geer and Lederer (2013) and Section 14.12 of Buhlmann and van de Geer (2011). We do not pursue either of these two strategies in the current paper. We instead consider a third, somewhat more transparent strategy, based on a covering number bound of the columns of $\Delta^\dagger$.

Lemma 11 *Let $g_1, \ldots g_r$ denote the "atoms" associated with the operator Δ, i.e., the columns of $\Delta^\dagger$, and let $\mathcal{G} = \{\pm g_i : i = 1, \ldots r\}$ denote the symmetrized set of atoms. Suppose that there exists constants ζ, C_0 with the following property: for each $j = 1, \ldots 2r$, there is an arrangement of j balls having radius at most*

$$C_0 \sqrt{n} j^{-1/\zeta},$$

with respect to the norm $\|\cdot\|_2$, that covers $\mathcal{G}$. Then the entropy bound in (17) *is met with $w = 2\zeta/(2+\zeta)$ and $E = O(1)$. Therefore, the generalized lasso estimate $\hat{\beta}$, with*

$$\lambda = \Theta\left(n^{\frac{\zeta}{2+2\zeta}} \|\Delta\beta_0\|_1^{-\frac{1}{1+\zeta}}\right),$$

satisfies

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(\frac{\text{nullity}(\Delta)}{n} + n^{-\frac{2+\zeta}{2+2\zeta}} \cdot \|\Delta\beta_0\|_1^{\frac{\zeta}{1+\zeta}}\right).$$

The entropy-based results in this subsection (Lemma 9, Theorem 10, and Lemma 11) may appear more complex than those involving incoherence in the previous subsection (Lemma 5 and Theorem 6). Indeed, the same can be said of their proofs, which can be found in the Appendix. But after all this entropy machinery has all been established, it can actually be remarkably easy to use, say, Lemma 11 to produce sharp results. We conclude by giving an example.

Corollary 12 *Consider the 1d fused lasso, i.e., the GTF estimator with $k = 0$ over a chain graph. In this case, we have $\Delta = D^{(1)}$, the univariate difference operator, and the symmetrized set $\mathcal{G}$ of atoms can be covered by j balls with radius at most $\sqrt{2n/j}$, for $j = 1, \ldots 2(n-1)$. Hence, with $\lambda = \Theta(n^{1/3}\|D^{(1)}\beta_0\|_1^{-1/3})$, the 1d fused lasso estimate $\hat{\beta}$ satisfies*

$$\frac{\|\hat{\beta} - \beta_0\|_2^2}{n} = O_{\mathbb{P}}\left(n^{-2/3} \cdot \|D^{(1)}\beta_0\|_1^{2/3}\right).$$

This corollary rederives the optimal convergence rate of $n^{-2/3}$ for the univariate fused lasso, assuming boundedness of $\|D^{(1)}\beta_0\|_1$, as has been already shown in Mammen and van de Geer (1997); Tibshirani (2014). Like Corollary 7 (but unlike previous works), its proof does not rely on any special facts about 1d functions of bounded variation. It only uses a covering number bound on the columns of the operator $(D^{(1)})^+$, a strategy that, in principle, extends to many other settings (graphs). It is worth emphasizing just how simple this covering number construction is, compared to the incoherence-based arguments that lead to the same result; we invite the curious reader to compare the proofs of Corollaries 7 and 12.

7. Discussion

In this work, we proposed graph trend filtering as a useful alternative to Laplacian and wavelet smoothers on graphs. This is analogous to the usefulness of univariate trend filtering in nonparametric regression, as an alternative to smoothing splines and wavelets (Tibshirani, 2014). We have documented empirical evidence for the superior local adaptivity of the ℓ_1-based GTF over the ℓ_2-based graph Laplacian smoother, and the superior robustness of GTF over wavelet smoothing in high-noise scenarios. Our theoretical analysis provides a basis for a deeper understanding of the estimation properties of GTF. More precise theoretical characterizations involving entropy will be the topic of future work, as will comparisons between the error rates achieved by GTF and other common estimators, such as Laplacian smoothing. These extensions, and many others, are well within reach.

ACKNOWLEDGMENTS

The authors would like to thank Harish Doraiswamy, Nivan Ferreira, Theodoros Damoulas and Claudio Silva for sharing the pre-processed NYC taxi data, Jeff Irion and Naoki Saito for their help with the implementation of the graph wavelet algorithms, as well as the associate editor and anonymous reviewers for the valuable feedback.

YW was supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office. JS was supported by NSF Grant DMS-1223137. AS was supported by a Google Faculty Research Grant. RT was supported by NSF Grant DMS-1309174.

Appendix A. Additional Analysis from Alternative Wavelet Designs

We provide detailed comparisons to a few recently proposed wavelet approaches for graph smoothing.

A.1 Allegheny County Example

In addition to considering the wavelet design of Sharpnack et al. (2013a) for the Allegheny County example, we also considered designs of Coiman and Maggioni (2006)—a classic method that builds diffusion wavelets on a graph, and Irion (2015)—a more recent graph wavelet construction. In contrast to Sharpnack et al. (2013a), which produces a single signal-independent orthogonal basis for a graph, both Coiman and Maggioni (2006); Irion (2015) build wavelet packets from a given graph structure. A wavelet packet is an overcomplete basis indexed by a hierarchical data structure that can be used to generate an exponential number of orthogonal bases. This construction is computationally expensive as it typically involves computing eigendecompositions of large matrices. Once the wavelet packet has been constructed, for each input signal that one observes over the graph in question, one runs a "best basis" algorithm to choose a particular orthogonal basis from the wavelet packet by optimizing a particular cost function of the eventual wavelet coefficients. This is based on a message-passing-like dynamic programming algorithm, and can be quite efficient. Lastly, the denoising procedure is defined as usual (e.g., as in Donoho and Johnstone (1995)), namely, one performs the basis transformation, soft-thresholds (or hard-thresholds) the coefficients, and then reconstructs the denoised signal.

In our experiments, we used the wavelet implementations released by the authors of Coiman and Maggioni (2006); Irion (2015) with their default settings. In particular, the former implementation of Coiman and Maggioni (2006) builds wavelets from a diffusion operator constructed from the adjacency matrix of a graph, and the cost function for the best basis is defined by the ℓ_1 norm of the wavelet coefficients. The latter implementation of Irion (2015) uses a more exhaustive search, building wavelet packets through a hierarchical partitioning and eigentransform of three different Laplacian matrices and a fourth generalized Haar-Walsh transform (GHWT), then choosing the best basis from all four collections by optimizing a meta cost function of the ℓ_p norm of wavelet coefficients over $p \in \{0.1, 0.2, \dots 2\}$. This is the "cumulative relative error" defined in equation (7.5) of Irion (2015).

In the left panel of Figure 8, we plot the mean squared errors for these new wavelet methods over the same 10 simulations from the Allegheny County example in Figure 2 of Section A.1. The middle and right panels of the figure show the denoised signals from the new methods fit to the data in Figure 1, at their optimal degrees of freedom (df) values (in terms of the achieved MSE). We can see that the spanning tree wavelet design of Sharpnack et al. (2013a) is the best performer among the three candidate wavelet designs. In a rough sense, the construction of Irion (2015) seems to perform similarly to that of Sharpnack et al. (2013a), in that the MSE is best for larger df values (corresponding to more nonzero wavelet coefficients, i.e., complex fitted models), whereas the construction of Coiman and Maggioni (2006) performs best for smaller df values (fewer nonzero wavelet coefficients, i.e., simpler fitted models).

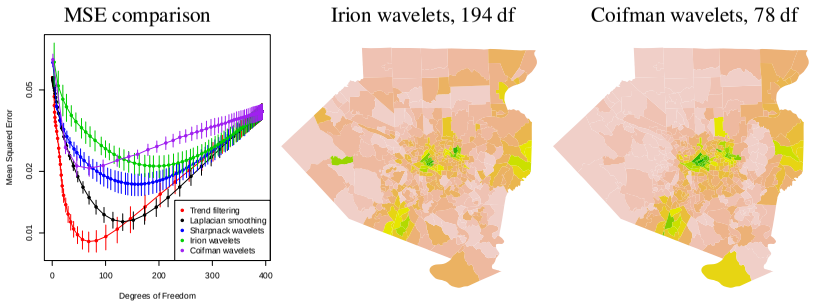


Figure 8: Additional wavelet analysis of the Allegheny County example.

A.2 Facebook Graph Example

Again, we consider the designs of Coiman and Maggioni (2006); Irion (2015) for the Facebook graph example of Section 5.1. Due to practical reasons, we had to change some of the default settings in the implementations provided by the authors of these wavelet methods; in the wavelet implementation of Coiman and Maggioni (2006), we took the power of the diffusion operator to be 1 instead of 4 (since the latter choice threw an error in the provided code); and in the wavelet implementation of Irion (2015), we used another "best basis" algorithm that only searches within the basis collection from the GHWT eigendecomposition, as the original algorithm was too slow due to the larger scale considered in this example. (In most examples in Irion (2015), the chosen bases come from the GHWT eigendecomposition.) We view these changes as minor, because when the same changes were applied to the methods of Coiman and Maggioni (2006); Irion (2015) on the smaller Allegheny County example, there are no obvious differences in the results.

Figure 9 shows the results for the two new wavelet methods on the Facebook graph simulation, using the same setup as in Figure 5. Once again, we find that the spanning tree wavelets of Sharpnack et al. (2013a) perform better or on par with the other two wavelet methods across essentially all scenarios.

Appendix B. Proofs of Theoretical Results

Here we present proofs of our theoretical results presented in Sections 3 and 6.

B.1 Proof of Lemma 1

For even k, we have $\Delta^{(k+1)} = DL^{k/2}$, so if A denotes a subset of edges, then $\Delta_{-A}^{(k+1)} = D_{-A}L^{k/2}$. Recall that for a connected graph, $\mathrm{null}(L) = \mathrm{span}\{\mathbb{1}\}$, and the same is true for any power of L. This means that we can write

$$\mathrm{null}(\Delta^{(k+1)}) = \mathrm{span}\{\mathbb{1}\} + \mathrm{span}\{\mathbb{1}\}^{\perp} \cap \{u : DL^{\frac{k}{2}}u = 0\}.$$

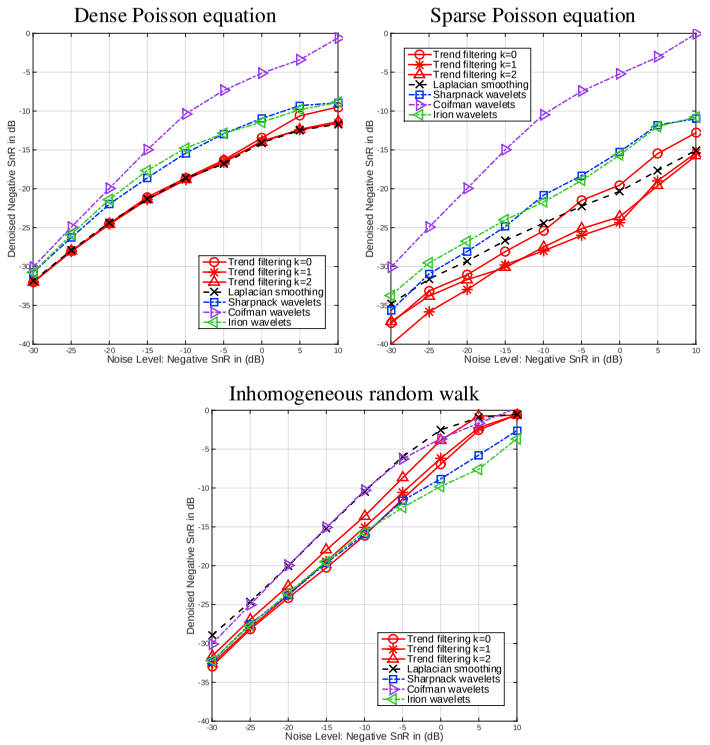


Figure 9: Additional wavelet analysis of the Facebook graph example.

Note that if $\mathbb{1}^\top u = 0$, then $v = L^{\frac{k}{2}} u \iff u = (L^\dagger)^{\frac{k}{2}} u$. Moreover, if G_{-A} has connected components $C_1, \ldots C_s$, then $\mathrm{null}(D_{-A}) = \mathrm{span}\{\mathbb{1}_{C_1}, \ldots \mathbb{1}_{C_s}\}$. Putting these statements together proves the result for even k. For k odd, the arguments are similar.

B.2 Proof of Theorem 3

By assumption we can write

$$y = \beta_0 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2 I).$$

Denote $R = \mathrm{row}(\Delta)$, the row space of Δ, and $R^\perp = \mathrm{null}(\Delta)$, the null space of Δ. Also let P_R be the projection onto R, and $P_{R^\perp}$ the projection onto $R^\perp$. Consider

$$\hat{\beta} = \operatorname*{argmin}_{\beta \in \mathbb{R}^n} \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|\Delta \beta\|_1,$$
$$\tilde{\beta} = \operatorname*{argmin}_{\beta \in \mathbb{R}^n} \frac{1}{2}\|P_R y - \beta\|_2^2 + \lambda \|\Delta \beta\|_1.$$

The first quantity $\hat{\beta} \in \mathbb{R}^n$ is the estimate of interest, the second one $\tilde{\beta} \in R$ is easier to analyze. Note that

$$\hat{\beta} = P_{R^\perp} y + \tilde{\beta},$$

and write $\|x\|_R = \|P_R x\|_2$, $\|x\|_{R^\perp} = \|P_{R^\perp} x\|_2$. Then

$$\|\hat{\beta} - \beta_0\|_2^2 = \|\epsilon\|_{R^\perp}^2 + \|\tilde{\beta} - \beta_0\|_R^2.$$

The first term is on the order $\dim(R^\perp) = \mathrm{nullity}(\Delta)$, and it suffices to bound the second term. Now we establish a basic inequality for $\tilde{\beta}$. By optimality of $\tilde{\beta}$, we have

$$\frac{1}{2}\|y - \tilde{\beta}\|_R^2 + \lambda \|\Delta \tilde{\beta}\|_1 \leq \frac{1}{2}\|y - \beta_0\|_R^2 + \lambda \|\Delta \beta_0\|_1,$$

and after rearranging terms,

$$\|\tilde{\beta} - \beta_0\|_R^2 \leq 2\epsilon^\top P_R(\tilde{\beta} - \beta_0) + 2\lambda \|\Delta \beta_0\|_1 - 2\lambda \|\Delta \tilde{\beta}\|_1. \tag{18}$$

This is our basic inequality. In the first term above, we use $P_R = \Delta^\dagger \Delta$, and apply Holder's inequality:

$$\epsilon^\top \Delta^\dagger \Delta(\tilde{\beta} - \beta_0) \leq \|(\Delta^\dagger)^\top \epsilon\|_\infty \|\Delta(\tilde{\beta} - \beta_0)\|_1. \tag{19}$$

If $\lambda \geq \|(\Delta^\dagger)^\top \epsilon\|_\infty$, then from (18), (19), and the triangle inequality, we see that

$$\|\tilde{\beta} - \beta_0\|_R^2 \leq 4\lambda \|\Delta \beta_0\|_1.$$

Well, $\|(\Delta^\dagger)^\top \epsilon\|_\infty = O_{\mathbb{P}}(M\sqrt{\log r})$ by a standard result on the maximum of Gaussians (derived using the union bound, and Mills' bound on the Gaussian tail), where recall M is the maximum ℓ_2 norm of the columns of $\Delta^\dagger$. Thus with $\lambda = \Theta(M\sqrt{\log r})$, we have from the above that

$$\|\tilde{\beta} - \beta_0\|_R^2 = O_{\mathbb{P}}\big(M\sqrt{\log r}\|\Delta \beta_0\|_1\big),$$

as desired.

B.3 Proof of Corollary 4

Case 1. When $\hat{\beta}$ is the univariate trend filtering estimator of order k, we are considering a penalty matrix $\Delta = D^{(k+1)}$, the univariate difference operator of order $k+1$. Note that $D^{(k+1)} \in \mathbb{R}^{(n-k-1)\times n}$, and its null space has constant dimension $k+1$. We show in Lemma 13 of Appendix B.4 that $(D^{(k+1)})^\dagger = P_R H_2^{(k)}/k!$, where $R = \mathrm{row}(D^{(k+1)})$, and $H_2^{(k)} \in \mathbb{R}^{n\times(n-k-1)}$ contains the last $n-k-1$ columns of the order k falling factorial basis matrix (Wang et al., 2014), evaluated

over the input points $x_1 = 1, \dots x_n = n$. The largest column norm of $P_R H_2^{(k)}/k!$ is on the order of $n^{k+1/2}$, which proves the result.

Cases 2 and 3. When G is the Ramanujan d-regular graph, the number of edges in the graph is $O(nd)$. The operator $\Delta = \Delta^{(k+1)}$ has number of rows $r = n$ when k is odd and $r = O(nd)$ when k is even; overall this is $O(nd)$. The dimension of the null space of Δ is constant (it is in fact 1, since the graph is connected). When G is the Erdos-Renyi random graph, the same bounds apply to the number of rows and the dimension of the null space, except that the bounds become probabilistic ones.

Now we apply the crude inequality, with e_i, $i = 1, \dots r$ denoting the standard basis vectors,

$$M = \max_{i=1,\dots r} \Delta^\dagger e_i \leq \max_{\|x\|_2 \leq 1} \Delta^\dagger x = \|\Delta^\dagger\|_2,$$

the right-hand side being the maximum singular value of $\Delta^\dagger$. As $\Delta = \Delta^{(k+1)}$, the graph difference operator of order $k+1$, we claim that

$$\|\Delta^\dagger\|_2 \leq 1/\lambda_{\min}(L)^{\frac{k+1}{2}}, \tag{20}$$

where $\lambda_{\min}(L)$ denotes the smallest nonzero eigenvalue of the graph Laplacian L. To see this, note first that $\|\Delta^\dagger\|_2 = 1/\sigma_{\min}(\Delta)$, where the denominator is the smallest nonzero singular value of Δ. Now for odd k, we have $\Delta^{(k+1)} = L^{(k+1)/2}$, and the claim follows as

$$\sigma_{\min}(L^{\frac{k+1}{2}}) = \min_{x \in R: \|x\|_2 \leq 1} \|L^{\frac{k+1}{2}} x\|_2 \geq \big(\sigma_{\min}(L)\big)^{\frac{k+1}{2}},$$

and $\sigma_{\min}(L) = \lambda_{\min}(L)$, since L is symmetric. Above, R denotes the row space of L (the space orthogonal to the vector $\mathbb{1}$ of all 1s). For even k, we have $\Delta^{(k+1)} = DL^{k/2}$, and again

$$\sigma_{\min}(DL^{\frac{k}{2}}) = \min_{x \in R: \|x\|_2 \leq 1} \|DL^{\frac{k+1}{2}} x\|_2 \geq \sigma_{\min}(D)\big(\sigma_{\min}(L)\big)^{\frac{k}{2}},$$

where $\sigma_{\min}(D) = \sqrt{\lambda_{\min}(L)}$, since $D^\top D = L$. This verifies the claim.

Having established (20), it suffices to lower bound $\lambda_{\min}(L)$ for the two graphs in question. Indeed, for both graphs, we have the lower bound

$$\lambda_{\min}(L) = \Omega(d - \sqrt{d}).$$

e.g., see Lubotzky et al. (1988); Marcus et al. (2014) for the Ramanujan graph and Feige and Ofek (2005); Chung and Radcliffe (2011) for the Erdos-Renyi graph. This completes the proof.

B.4 Calculation of $(D^{(k+1)})^\dagger$

Lemma 13 *The $(k+1)$st order discrete difference operator has pseudoinverse*

$$(D^{(k+1)})^\dagger = P_R H_2^{(k)}/k!,$$

where we denote $R = \mathrm{row}(D^{(k+1)})$, and $H_2^{(k)} \in \mathbb{R}^{n \times (n-k-1)}$ the last $n-k-1$ columns of the kth order falling factorial basis matrix.

Proof We abbreviate $D = D^{(k+1)}$, and consider the linear system

$$DD^\top x = Db \tag{21}$$

in x, where $b \in \mathbb{R}^n$ is arbitrary. We seek an expression for $x = (DD^\top)^{-1} D^\top = (D^\dagger)^\top b$, and this will tell us the form of $D^\dagger$. Define

$$\tilde{D} = \begin{bmatrix} C \\ D \end{bmatrix} \in \mathbb{R}^{n \times n},$$

where $C \in \mathbb{R}^{(k+1) \times n}$ is the matrix that collects the first row of each lower order difference operator, defined in Lemma 2 of Wang et al. (2014). From this same lemma, we know that

$$\tilde{D}^{-1} = H/k!,$$

where $H = H^{(k)}$ is falling factorial basis matrix of order k, evaluated over $x_1, \dots x_n$. With this in mind, consider the expanded linear system

$$\begin{bmatrix} CC^\top & CD^\top \\ DC^\top & DD^\top \end{bmatrix} \begin{bmatrix} w \\ x \end{bmatrix} = \begin{bmatrix} a \\ Db \end{bmatrix}. \tag{22}$$

The second equation reads

$$DC^\top w + DD^\top x = Db,$$

and so if we can choose a in (22) so that at the solution we have $w = 0$, then x is the solution in (21). The first equation in (22) reads

$$CC^\top w + CD^\top x = a,$$

i.e.,

$$w = (CC^\top)^{-1}(a - CD^\top x).$$

That is, we want to choose

$$a = CD^\top x = CD^\top (DD^\top)^{-1} Db = CP_R b,$$

where P_R is the projection onto row space of D. Thus we can reexpress (22) as

$$\tilde{D}\tilde{D}^\top \begin{bmatrix} w \\ x \end{bmatrix} = \begin{bmatrix} CP_R b \\ Db \end{bmatrix} = \tilde{D} P_R b$$

and, using $\tilde{D}^{-1} = H/k!$,

$$\begin{bmatrix} w \\ x \end{bmatrix} = H^\top P_R b/k!.$$

Finally, writing H_2 for the last $n-k-1$ columns of H, we have $x = H_2^\top P_R b/k!$, as desired. ■

Remark. The above proof did not rely on the input points $x_1, \dots x_n$; indeed, the result holds true for any sequence of inputs used to define the discrete difference matrix and falling factorial basis matrix.

B.5 Proof of Lemma 5

We follow the proof of Theorem 3, up until the application of Holder's inequality in (19). In place of this step, we use the linearized bound in (15), which we claim implies that

$$\epsilon^\top P_R(\tilde{\beta} - \beta_0) \leq \tilde{B}\|\tilde{\beta} - \beta_0\|_R + A\|\Delta(\tilde{\beta} - \beta_0)\|_1,$$

where $\tilde{B} = O_{\mathbb{P}}(B)$. This simply follows from applying (15) to $x = P_R(\tilde{\beta} - \beta_0)/\|\Delta(\tilde{\beta} - \beta_0)\|_1$, which is easily seen to be an element of $\mathcal{S}_\Delta(1)$. Hence we can take take $\lambda = \Theta(A)$, and argue as in the proof of Theorem 3 to arrive at

$$\|\tilde{\beta} - \beta_0\|_R^2 \leq \tilde{B}\|\tilde{\beta} - \beta_0\|_R + \tilde{A}\|\Delta\beta_0\|_1,$$

where $\tilde{A} = O_{\mathbb{P}}(A)$. Note that the above is a quadratic inequality of the form $ax^2 - bx - c \leq 0$ with $x = \|\tilde{\beta} - \beta_0\|_R$. As $a > 0$, the larger of its two roots serves as a bound for x, i.e., $x \leq (b + \sqrt{b^2 + 4ac})/(2a) \leq b/a + \sqrt{c/a}$, or $x^2 \leq 2b^2/a^2 + 2c/a$, which means that

$$\|\tilde{\beta} - \beta_0\|_R^2 \leq 2\tilde{B}^2 + 2\tilde{A}\|\Delta\beta_0\|_1 = O_{\mathbb{P}}\big(B^2 + A\|\Delta\beta_0\|_1\big),$$

completing the proof.

B.6 Proof of Theorem 6

For an index $i_0 \in \{1, \dots q\}$, let

$$C = \mu\sqrt{\frac{2\log 2r}{n}\sum_{i=i_0+1}^{q}\frac{1}{\xi_i^2}}.$$

We will show that

$$\max_{x \in \mathcal{S}_\Delta(1)} \frac{\epsilon^\top x - 1.001\sigma C}{\|x\|_2} = O_{\mathbb{P}}(\sqrt{i_0}).$$

Invoking Lemma 5 with $A = 1.001\sigma C$ and $b = \sqrt{i_0}$ would then give the result.

Henceforth we denote $[i] = \{1, \dots i\}$. Recall that $q = \mathrm{rank}(\Delta)$. Let the singular value decomposition of Δ be

$$\Delta = U\Sigma V^\top,$$

where $U \in \mathbb{R}^{r \times q}$, $V \in \mathbb{R}^{n \times q}$ are orthogonal, and $\Sigma \in \mathbb{R}^{q \times q}$ has diagonal elements $(\Sigma)_{ii} = \xi_i > 0$ for $i \in [q]$. First, let us establish that

$$\Delta^\dagger = V\Sigma^{-1}U^\top.$$

Consider an arbitrary point $x = P_R z \in \mathcal{S}_\Delta(1)$. Denote the projection $P_{[i_0]} = V_{[i_0]}V_{[i_0]}^\top$ where $V_{[i_0]}$ contains the first i_0 right singular vectors. We can decompose

$$\epsilon^\top P_R z = \epsilon^\top P_{[i_0]} P_R z + \epsilon^\top (I - P_{[i_0]}) P_R z.$$

The first term can be bounded by

$$\epsilon^\top P_{[i_0]} P_R z \leq \|P_{[i_0]}\epsilon\|_2 \|z\|_R = O_{\mathbb{P}}(\sqrt{i_0}\|z\|_R),$$

using the fact that $\|P_{[i_0]}\epsilon\|_2^2 \overset{d}{=} \sum_{i=1}^{i_0}\epsilon_i^2$. We can bound the second term by

$$\epsilon^\top (I - P_{[i_0]}) P_R z = \epsilon^\top (I - P_{[i_0]})\Delta^\dagger \Delta z \leq \|(\Delta^\dagger)^\top (I - P_{[i_0]})\epsilon\|_\infty,$$

using $P_R = \Delta^\dagger\Delta$, Holder's inequality, and the fact that $\|\Delta z\|_1 \leq 1$. Define $g_j = (I - P_{[i_0]})\Delta^\dagger e_j$ for $j \in [r]$ with e_j the jth canonical basis vector. So,

$$\|g_j\|_2^2 = \|[\,0 \;\; V_{[n]\setminus[i_0]}\,] \cdot \Sigma^{-1}U^\top e_j\|_2^2 \leq \frac{\mu^2}{n}\sum_{i=i_0+1}^{q}\frac{1}{\xi_i^2},$$

by rotational invariance of $\|\cdot\|_2$ and the incoherence assumption on the columns of U. By a standard result on maxima of Gaussians,

$$\|(\Delta^\dagger)^\top (I - P_{[i_0]})\epsilon\|_\infty = \max_{j \in [r]} |g_j^\top \epsilon| \leq 1.001\sigma\sqrt{2\log(2r)\frac{\mu^2}{n}\sum_{i=i_0+1}^{q}\frac{1}{\xi_i^2}} = 1.001\sigma C,$$

with probability approaching 1. Putting these two terms together completes the proof, as we have shown that

$$\frac{\epsilon^\top P_R z - 1.001\sigma C}{\|z\|_R} = O_{\mathbb{P}}(\sqrt{i_0}),$$

with the probability bound on the right-hand side not depending on z.

B.7 Proof of Corollary 7

We focus on the k odd and k even cases separately.

Case for k odd. When k is odd, we have $\Delta = \Delta^{(k+1)} = L^{(k+1)/2}$, where L the graph Laplacian of a chain graph (i.e., 1d grid graph), to be perfectly explicit,

$$L = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix}.$$

In numerical methods for differential equations, this matrix L is called the finite difference operator for the 1d Laplace equation with Neumann boundary conditions (e.g., Conte and de Boor, 1980; Godunov and Ryabenkii, 1987), and is known to have eigenvalues and eigenvectors

$$\xi_i = 4\sin^2\Big(\frac{\pi(i-1)}{2n}\Big), \quad \text{for } i = 1, \dots n,$$

$$u_{ij} = \begin{cases} \frac{1}{\sqrt{n}} & \text{if } i = 1 \\ \sqrt{\frac{2}{n}}\cos\Big(\frac{\pi(i-1)(j-1/2)}{n}\Big) & \text{otherwise} \end{cases}, \quad \text{for } i, j = 1, \dots n.$$

Therefore, the eigenvectors of L are incoherent with constant $\mu = \sqrt{2}$. This of course implies the same of $L^{(k+1)/2}$, which shares the eigenvectors of L. Meanwhile, the eigenvalues of $L^{(k+1)/2}$ are just given by raising those of L to the power of $(k+1)/2$, and for $i_0 \in \{1, \dots n\}$, we compute the partial sum of their squared reciprocals, as in

$$\frac{1}{n} \sum_{i=i_0+1}^{n} \frac{1}{\xi_i^{k+1}} = \frac{1}{n} \sum_{i=i_0+1}^{n} \frac{1}{4^{k+1} \sin^{2k+2}(\pi(i-1)/(2n))} \leq \int_{(i_0-1)/n}^{(n-2)/n} \frac{1}{4^{k+1} \sin^{2k+2}(\pi x/2)} dx,$$

where we have used the fact that the right-endpoint Riemann sum, for a monotone nonincreasing function, is an underestimate of its integral. Continuing on, the above integral can be bounded by

$$\frac{1}{4^{k+1} \sin^{2k}(\pi i_0/(2n))} \int_{(i_0-1)/n}^{1} \frac{1}{\sin^2(\pi x/2)} dx = \frac{2 \cot(\pi i_0/(2n))}{4^{k+1} \pi \sin^{2k}(\pi i_0/(2n))} \leq \frac{1}{4^{k+1}\pi} \left(\frac{2n}{\pi i_0}\right)^{2k+1},$$

the last step using a Taylor expansion around 0. Hence to choose a tight a bound as possible in Theorem 6, we seek to balance i_0 with $\sqrt{(n/i_0)^{2k+1} \log n} \cdot \|\Delta^{(k+1)} \beta_0\|_1$. This is accomplished by choosing

$$i_0 = n^{\frac{2k+1}{2k+3}} (\log n)^{\frac{1}{2k+3}} \|\Delta^{(k+1)} \beta_0\|_1^{\frac{2}{2k+3}},$$

and applying Theorem 6 gives the result for k odd.

Case for k even. When k is even, we instead have $\Delta = \Delta^{(k+1)} = DL^{k/2}$, where D is the edge incidence matrix of a 1d chain, and $L = D^\top D$. It is clear that the left singular vectors of $DL^{k/2}$ are simply the left singular vectors of D, or equivalently, the eigenvectors of $DD^\top$. To be explicit,

$$DD^\top = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 2 \end{bmatrix},$$

which is called the finite difference operator associated with the 1d Laplace equation under Dirichlet boundary conditions in numerical methods (e.g., Conte and de Boor, 1980; Godunov and Ryabenkii, 1987), and is known to have eigenvectors

$$u_{ij} = \sqrt{\frac{2}{n}} \sin\left(\frac{\pi i j}{n}\right), \quad \text{for } i, j = 1, \dots n-1.$$

It is evident that these vectors are incoherent, with constant $\mu = \sqrt{2}$. Furthermore, the singular values of $DL^{k/2}$ are exactly the eigenvalues of L raised to the power of $(k+1)/2$, and the remainder of the proof goes through as in the k odd case.

B.8 Proof of Corollary 8

Again we treat the k odd and even cases separately.

Case for k odd. As k is odd, the GTF operator is $\Delta = \Delta^{(k+1)} = L^{(k+1)/2}$, where the L is the Laplacian matrix of a 2d grid graph. Writing $L_{1\text{d}} \in \mathbb{R}^{\ell \times \ell}$ for the Laplacian matrix over a 1d grid of size $\ell = \sqrt{n}$ (and $I \in \mathbb{R}^{\ell \times \ell}$ for the identity matrix), we note that

$$L = I \otimes L_{1\text{d}} + L_{1\text{d}} \otimes I,$$

i.e., the 2d grid Laplacian L is the Kronecker sum of the 1d grid Laplacian $L_{1\text{d}}$, so its eigenvectors are given by all pairwise Kronecker products of eigenvectors of $L_{1\text{d}}$, of the form $u_i \otimes u_j$. Moreover, it is not hard to see that each $u_i \otimes u_j$ has unit norm (since u_i, u_j do) and $\|u_i \otimes u_j\|_\infty \leq 2/\sqrt{n}$. This allows us to conclude that the eigenvectors of L obey the incoherence property with $\mu = 2$.

The eigenvalues of L are given by all pairwise sums of eigenvalues in the 1d case. Indexing by 2d grid coordinates, we may write these as

$$\xi_{j_1, j_2} = 4 \sin^2\left(\frac{\pi(j_1 - 1)}{2\ell}\right) + 4 \sin^2\left(\frac{\pi(j_2 - 1)}{2\ell}\right), \quad \text{for } j_1, j_2 = 1, \dots \ell.$$

Eigenvalues of $L^{(k+1)/2}$ are just given by raising the above to the power of $(k+1)/2$, and for $j_0 \in \{1, \dots \ell\}$, we let $i_0 = j_0^2$, and compute the sum

$$\frac{1}{n} \sum_{\max\{j_1 j_2\} > j_0} \frac{1}{\xi_{j_1, j_2}^{k+1}} \leq \frac{2}{n} \sum_{j_1 = j_0+1}^{\ell} \sum_{j_2=1}^{\ell} \frac{1}{\xi_{j_1, j_2}^{k+1}} \leq \frac{2}{\ell} \sum_{j_1 = j_0+1}^{\ell} \frac{1}{4^{k+1} \sin^{2k+2}(\pi(j_1 - 1)/(2\ell))}.$$

Just as we argued in the 1d case (for k odd), the above is bounded by

$$\frac{2}{4^{k+1}\pi} \left(\frac{2\ell}{\pi j_0}\right)^{2k+1},$$

and thus we seek to balance $i_0 = j_0^2$ with $\sqrt{(\ell/j_0)^{2k+1} \log n} \cdot \|\Delta^{(k+1)} \beta_0\|_1$. This yields

$$j_0 = \ell^{\frac{2k+1}{2k+5}} (\log n)^{\frac{1}{2k+5}} \|\Delta^{(k+1)} \beta_0\|_1^{\frac{2}{2k+5}},$$

i.e.,

$$i_0 = n^{\frac{2k+1}{2k+5}} (\log n)^{\frac{2}{2k+5}} \|\Delta^{(k+1)} \beta_0\|_1^{\frac{4}{2k+5}},$$

and applying Theorem 6 gives the result for k odd.

Case for k even. For k even, we have the GTF operator being $\Delta = \Delta^{(k+1)} = DL^{k/2}$, where D is the edge incidence matrix of a 2d grid, and $L = D^\top D$. It will be helpful to write

$$D = \begin{bmatrix} I \otimes D_{1\text{d}} \\ D_{1\text{d}} \otimes I \end{bmatrix},$$

where $D_{1\text{d}} \in \mathbb{R}^{(\ell-1) \times \ell}$ is the difference operator for a 1d grid of size $\ell = \sqrt{n}$ (and $I \in \mathbb{R}^{\ell \times \ell}$ is the identity matrix). It suffices to check the incoherence of the left singular vectors of $DL^{k/2}$, since the eigenvalues of $DL^{k/2}$ are those of L raised to the power of $(k+1)/2$, and so the rest of the proof then follows precisely as in the case when k is odd. The left singular vectors of $DL^{k/2}$ are the same as the left singular vectors of D, which are the eigenvectors of $DD^\top$. Observe that

$$DD^\top = \begin{bmatrix} I \otimes D_{1\text{d}} D_{1\text{d}}^\top & D_{1\text{d}}^\top \otimes D_{1\text{d}} \\ D_{1\text{d}} \otimes D_{1\text{d}}^\top & D_{1\text{d}} D_{1\text{d}}^\top \otimes I \end{bmatrix}.$$

Let u_i, $i = 1, \dots \ell - 1$ be the eigenvectors of $D_{1\mathrm{d}} D_{1\mathrm{d}}^\top$, corresponding to eigenvalues λ_i, $i = 1, \dots \ell - 1$. Define $v_i = D_{1\mathrm{d}}^\top u_i / \sqrt{\lambda_i}$, $i = 1, \dots \ell - 1$, and $e = \mathbb{1}/\sqrt{\ell}$, where $\mathbb{1} = (1, \dots 1) \in \mathbb{R}^\ell$ is the vector of all 1s. A straightforward calculation verifies that

$$
\begin{aligned}
DD^\top \begin{bmatrix} v_i \otimes u_i \\ u_i \otimes v_i \end{bmatrix} &= 2\lambda_i \begin{bmatrix} v_i \otimes u_i \\ u_i \otimes v_i \end{bmatrix}, \quad \text{for } i = 1, \dots \ell - 1, \\
DD^\top \begin{bmatrix} e \otimes u_i \\ 0 \end{bmatrix} &= \lambda_i \begin{bmatrix} e \otimes u_i \\ 0 \end{bmatrix}, \quad \text{for } i = 1, \dots \ell - 1, \\
DD^\top \begin{bmatrix} 0 \\ u_i \otimes e \end{bmatrix} &= \lambda_i \begin{bmatrix} 0 \\ u_i \otimes e \end{bmatrix}, \quad \text{for } i = 1, \dots \ell - 1.
\end{aligned}
$$

Hence we have derived $3(\ell - 1)$ eigenvectors of $DD^\top$. Note that the vectors v_i, $i = 1, \dots \ell - 1$ are actually the eigenvectors of $L_{1\mathrm{d}} = D_{1\mathrm{d}}^\top D_{1\mathrm{d}}$ (corresponding to the $\ell - 1$ nonzero eigenvalues), and from our work in the 1d case, recall, both v_i, $i = 1, \dots \ell - 1$ (studied for k odd) and u_i, $i = 1, \dots \ell - 1$ (studied for k even) are unit vectors satisfying the incoherence property with $\mu = \sqrt{2}$. This means that the above eigenvectors are all unit norm, and are also incoherent, with constant $\mu = 2$.

There are $(\ell - 1)(\ell - 2)$ more eigenvectors of $DD^\top$, as the rank of $DD^\top$ is $n - 1 = \ell^2 - 1$. A somewhat longer but still straightforward calculation verifies that

$$
\begin{aligned}
DD^\top \begin{bmatrix} v_i \otimes u_j + v_j \otimes u_i \\ \sqrt{\frac{\lambda_i}{\lambda_j}} u_i \otimes v_j + \sqrt{\frac{\lambda_j}{\lambda_i}} u_j \otimes v_i \end{bmatrix} &= (\lambda_i + \lambda_j) \begin{bmatrix} v_i \otimes u_j + v_j \otimes u_i \\ \sqrt{\frac{\lambda_i}{\lambda_j}} u_i \otimes v_j + \sqrt{\frac{\lambda_j}{\lambda_i}} u_j \otimes v_i \end{bmatrix}, \text{ for } i < j, \\
DD^\top \begin{bmatrix} \sqrt{\frac{\lambda_j}{\lambda_i}} v_i \otimes u_j + \sqrt{\frac{\lambda_i}{\lambda_j}} v_j \otimes u_i \\ u_i \otimes v_j + u_j \otimes v_i \end{bmatrix} &= (\lambda_i + \lambda_j) \begin{bmatrix} \sqrt{\frac{\lambda_j}{\lambda_i}} v_i \otimes u_j + \sqrt{\frac{\lambda_i}{\lambda_j}} v_j \otimes u_i \\ u_i \otimes v_j + u_j \otimes v_i \end{bmatrix}, \text{ for } i < j.
\end{aligned}
$$

Modulo the appropriate normalization, we have derived the remaining $(\ell - 1)(\ell - 2)$ eigenvectors of $DD^\top$. It remains to check their incoherence, once we have normalized them (to have unit norm). As the eigenvectors in the first and second expressions above are simply (block) rearrangements of each other, it does not matter which form we study; consider, say, those in the second expression, and fix $i < j$. The entrywise absolute maximum of the eigenvector in question is at most $\sqrt{\lambda_j/\lambda_i}(4/\sqrt{n})$. Thus it suffices show that the normalization constant for this eigenvector is on the order of $\sqrt{\lambda_j/\lambda_i}$. Observe that

$$
\left\| \begin{bmatrix} \sqrt{\frac{\lambda_j}{\lambda_i}} v_i \otimes u_j + \sqrt{\frac{\lambda_i}{\lambda_j}} v_j \otimes u_i \\ u_i \otimes v_j + u_j \otimes v_i \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} \sqrt{\frac{\lambda_j}{\lambda_i}} v_i \otimes u_j \\ u_i \otimes v_j \end{bmatrix} \right\|_2^2 + \left\| \begin{bmatrix} \sqrt{\frac{\lambda_i}{\lambda_j}} v_j \otimes u_i \\ u_j \otimes v_i \end{bmatrix} \right\|_2^2.
$$

Here the cross-term is $(v_i^\top \otimes u_j^\top)(v_j \otimes u_i) = (v_i^\top v_j)(u_i^\top u_j) = 0$, as $v_i^\top v_j = 0$ and $u_i^\top u_j = 0$. This means that the normalization constant lies within $[\sqrt{\lambda_j/\lambda_i + 2}, \sqrt{2\lambda_j/\lambda_i + 2}]$. In particular, the lower bound shows that the incoherence property holds with $\mu = 4$. This completes the proof.

B.9 Proof of Lemma 9

As before, we follow the proof of Theorem 3 up until the application of Holder's inequality in (19), but we use the fractional bound in (16) instead. We claim that this implies

$$
\epsilon^\top P_R(\tilde{\beta} - \beta_0) \leq \tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} (\|\Delta\tilde{\beta}\|_1 + \|\Delta\beta_0\|_1)^{w/2},
$$

where $\tilde{K} = O_{\mathbb{P}}(K)$. This is verified by noting that $x = P_R(\tilde{\beta} - \beta_0)/(\|\Delta\tilde{\beta}\|_1 + \|\Delta\beta_0\|_1) \in \mathcal{S}_\Delta(1)$, applying (16) to x, and then rearranging. Therefore, as in the proof of Theorem 3, we have

$$
\|\tilde{\beta} - \beta_0\|_R^2 \leq 2\tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} (\|\Delta\tilde{\beta}\|_1 + \|\Delta\beta_0\|_1)^{w/2} + 2\lambda(\|\Delta\beta_0\|_1 - \|\Delta\tilde{\beta}\|_1), \tag{23}
$$

We now set

$$
\lambda = \Theta\left(K^{\frac{2}{1+w/2}} \|\Delta\beta_0\|_1^{-\frac{1-w/2}{1+w/2}} \right),
$$

and in the spirit of van de Geer (1990); Mammen and van de Geer (1997), we proceed to argue in cases.

Case 1. Suppose that $\frac{1}{2}\|\Delta\tilde{\beta}\|_1 \geq \|\Delta\beta_0\|_1$. Then we see that (23) implies

$$
0 \leq \|\tilde{\beta} - \beta_0\|_R^2 \leq \tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} \left(\frac{3}{2}\right)^{w/2} \|\Delta\tilde{\beta}\|_1^{w/2} - \lambda\|\Delta\tilde{\beta}\|_1, \tag{24}
$$

so that

$$
\lambda\|\Delta\tilde{\beta}\|_1 \leq \tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} \|\Delta\tilde{\beta}\|_1^{w/2},
$$

where for simplicity have absorbed a constant factor $2(3/2)^{w/2}$ into $\tilde{K}$ (since this does not change the fact that $\tilde{K} = O_{\mathbb{P}}(K)$), and thus

$$
\|\Delta\tilde{\beta}\|_1 \leq \left(\frac{\tilde{K}}{\lambda}\right)^{\frac{1}{1-w/2}} \|\tilde{\beta} - \beta_0\|_R.
$$

Plugging this back into (24) gives

$$
\|\tilde{\beta} - \beta_0\|_R^2 \leq \tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} \left(\frac{\tilde{K}}{\lambda}\right)^{\frac{w/2}{1-w/2}} \|\tilde{\beta} - \beta_0\|_R^{w/2},
$$

or

$$
\|\tilde{\beta} - \beta_0\|_R \leq \tilde{K}^{\frac{1}{1+w/2}} \left(\frac{1}{\lambda}\right)^{\frac{w/2}{1-w/2}} = O_{\mathbb{P}}\left(K^{\frac{1}{1+w/2}} \|\Delta\beta_0\|_1^{\frac{w/2}{1+w/2}} \right),
$$

as desired.

Case 2. Suppose that $\frac{1}{2}\|\Delta\tilde{\beta}\|_1 \leq \|\Delta\beta_0\|_1$. Then from (23),

$$
\|\tilde{\beta} - \beta_0\|_R^2 \leq \underbrace{2\lambda\|\Delta\beta_0\|_1}_{a} + \underbrace{2\tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} 3^{w/2} \|\Delta\beta_0\|_1^{w/2}}_{b},
$$

and hence either $\|\tilde{\beta} - \beta_0\|_R^2 \leq 2a$, or $\|\tilde{\beta} - \beta_0\|_R^2 \leq 2b$, and $a \leq b$. The first subcase is straightforward and leads to

$$
\|\tilde{\beta} - \beta_0\|_R \leq 2\sqrt{\lambda\|\Delta\beta_0\|_1} = O_{\mathbb{P}}\left(K^{\frac{1}{1+w/2}} \|\Delta\beta_0\|_1^{\frac{w/2}{1+w/2}} \right),
$$

as desired. In the second subcase, we have by assumption

$$
\|\tilde{\beta} - \beta_0\|_R^2 \leq 2\tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} \|\Delta\beta_0\|_1^{w/2}, \tag{25}
$$

$$
2\lambda\|\Delta\beta_0\|_1 \leq \tilde{K} \|\tilde{\beta} - \beta_0\|_R^{1-w/2} \|\Delta\beta_0\|_1^{w/2}, \tag{26}
$$

where again we have absorbed a constant factor $2(3^{w/2})$ into $\tilde{K}$. Working from (26), we derive

$$\|\Delta\beta_0\|_1 \leq \Big(\frac{\tilde{K}}{2\lambda}\Big)^{\frac{1}{1-w/2}} \|\tilde{\beta} - \beta_0\|_R,$$

and plugging this back into (25), we see

$$\|\tilde{\beta} - \beta_0\|_R^2 \leq 2\tilde{K}\|\tilde{\beta} - \beta_0\|_R^{1-w/2} \Big(\frac{\tilde{K}}{2\lambda}\Big)^{\frac{w/2}{1-w/2}} \|\tilde{\beta} - \beta_0\|_R^{w/2},$$

and finally

$$\|\tilde{\beta} - \beta_0\|_R \leq 2\tilde{K}^{\frac{1}{1+w/2}} \Big(\frac{1}{\lambda}\Big)^{\frac{w/2}{1-w/2}} = O_{\mathbb{P}}\left(K^{\frac{1}{1+w/2}} \|\Delta\beta_0\|_1^{\frac{w/2}{1+w/2}}\right).$$

This completes the second case, and the proof.

B.10 Proof of Theorem 10

The proof follows closely from Lemma 3.5 of van de Geer (1990). However, this author uses a different problem scaling than ours, so some care must be taken in applying the lemma. First we abbreviate $\mathcal{S} = \mathcal{S}_\Delta(1)$, and define $\tilde{\mathcal{S}} = \mathcal{S} \cdot \sqrt{n}/M$, where recall M is the maximum column norm of $\Delta^\dagger$. Now it is not hard to check that

$$\mathcal{S} = \{x \in \mathrm{row}(\Delta) : \|\Delta x\|_1 \leq 1\} = \Delta^\dagger\{\alpha \in \mathrm{col}(\Delta) : \|\alpha\|_1 \leq 1\},$$

so that $\max_{x\in\mathcal{S}} \|x\|_2 \leq M$, and $\max_{x\in\tilde{\mathcal{S}}} \|x\|_2 \leq \sqrt{n}$. This is important because Lemma 3.5 of van de Geer (1990) concerns a form of "local" entropy that allows for deviations on the order of $\sqrt{n}$ in the norm $\|\cdot\|_2$, or equivalently, constant order in the scaled metric $\|\cdot\|_n = \|\cdot\|_2/\sqrt{n}$. Hence, the entropy bound in (17) translates into

$$\log N(\delta, \tilde{\mathcal{S}}, \|\cdot\|_2) \leq E\Big(\frac{\sqrt{n}}{M}\Big)^w \Big(\frac{\sqrt{n}}{\delta}\Big)^w,$$

that is,

$$\log N(\delta, \tilde{\mathcal{S}}, \|\cdot\|_n) \leq E\Big(\frac{\sqrt{n}}{M}\Big)^w \delta^{-w}.$$

Now we apply Lemma 3.5 of van de Geer (1990): in the scaled metric used by this author,

$$\max_{x\in\tilde{\mathcal{S}}} \frac{\epsilon^\top x}{\sqrt{n}\|x\|_n^{1-w/2}} = O_{\mathbb{P}}\left(\sqrt{E}\Big(\frac{\sqrt{n}}{M}\Big)^{w/2}\right),$$

that is,

$$\max_{x\in\tilde{\mathcal{S}}} \frac{\epsilon^\top x}{\|x\|_2^{1-w/2}} = O_{\mathbb{P}}\left(\sqrt{E}(\sqrt{n})^{w/2}\Big(\frac{\sqrt{n}}{M}\Big)^{w/2}\right),$$

and finally,

$$\max_{x\in\mathcal{S}} \frac{\epsilon^\top x}{\|x\|_2^{1-w/2}} = O_{\mathbb{P}}\left(\sqrt{E}(\sqrt{n})^{w/2}\right),$$

as desired.

B.11 Proof of Corollary 11

For each $j = 1, \dots 2r$, if $\mathcal{G}$ is covered by j balls having radius at most $C_0\sqrt{n}j^{-1/\zeta}$, with respect to the norm $\|\cdot\|_2$, then it is covered by j balls having radius at most $C_0 j^{-1/\zeta}$, with respect to the scaled norm $\|\cdot\|_n = \|\cdot\|_2/\sqrt{n}$. By Theorem 1 of Carl (1997), this implies that for each $j = 1, 2, 3, \dots$, the convex hull $\mathrm{conv}(\mathcal{G})$ is covered by 2^j balls having radius at most $C_0' j^{-(1/2+1/\zeta)}$, with respect to $\|\cdot\|_n$, for another constant C_0'. Converting this back to an entropy bound in our original metric, and noting that $\mathrm{conv}(\mathcal{G}) = \mathcal{S}_\Delta(1)$, we have

$$\log(\delta, \mathcal{S}_\Delta(1), \|\cdot\|_2) \leq C_0''\Big(\frac{\sqrt{n}}{\delta}\Big)^{\frac{1}{1/2+1/\zeta}},$$

for a constant C_0'', as needed. This proves the lemma.

B.12 Proof of Corollary 12

According to Lemma 13, we know that $(D^{(1)})^\dagger = P_{\mathbb{1}}^\perp H$, where H is an $n \times (n-1)$ lower triangular matrix with $H_{ij} = 1$ if $i > j$ and 0 otherwise, and $P_{\mathbb{1}}^\perp$ is the projection map orthogonal to the all 1s vector. Thus $g_i = P_{\mathbb{1}}^\perp h_i$, $i = 1, \dots n-1$, with $h_1, \dots h_{n-1}$ denoting the columns of H. It is immediately apparent that

$$\|g_i - g_\ell\|_2 \leq \|h_i - h_\ell\|_2 \leq \sqrt{i-\ell},$$

for all $i > \ell$. Now, given $2j$ balls at our disposal, consider centering the first j balls at

$$g_d, g_{2d}, \dots g_{jd},$$

where $d = \lfloor n/j \rfloor$. Also let these balls have radius $\sqrt{n/j}$. By construction, then, we see that

$$\|g_1 - g_d\|_2 \leq \sqrt{n/j},\ \|g_d - g_{2d}\|_2 \leq \sqrt{n/j},\ \dots \|g_{jd} - g_{n-1}\|_2 \leq \sqrt{n/j},$$

which means that we have covered $g_1, \dots g_{n-1}$ with j balls of radius $\sqrt{n/j}$.

We can cover $-g_1, \dots, -g_{n-1}$ with the remaining j balls analogously. Therefore, we have shown that $2j$ balls require a radius of $\sqrt{n/j}$, or in other words, j balls require a radius of $\sqrt{2n/j}$.

References

Alvaro Barbero and Suvrit Sra. Fast Newton-type methods for total variation regularization. In *International Conference on Machine Learning (ICML-11)*, volume 28, pages 313–320, 2011.

Alvaro Barbero and Suvrit Sra. Modular proximal optimization for multidimensional total-variation regularization. arXiv: 1411.0589, 2014.

Dimitri P Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2):221–246, 1982.

Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

Kristian Bredies, Karl Kunisch, and Thomas Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010.

Peter Buhlmann and Sara van de Geer. *Statistics for High-Dimensional Data.* Springer, Berlin, 2011.

Bernd Carl. Metric entropy of convex hulls in Hilbert spaces. *Bulletin of the London Mathematical Society*, 29(04):452–458, 1997.

Antonin Chambolle and Jerome Darbon. On total variation minimization and surface evolution using parametric maximum flows. *International Journal of Computer Vision*, 84(3):288–307, 2009.

Fan Chung and Mary Radcliffe. On the spectra of general random graphs. *The Electronic Journal of Combinatorics*, 18(1), 2011.

Ronald Coiman and Mauro Maggioni. Diffusion wavelets. *Applied and Computational Harmonic Analysis*, 21(2):53–94, 2006.

Samuel Conte and Carl de Boor. *Elementary Numerical Analysis: An Algorithmic Approach.* McGraw-Hill, New York, 1980. International Series in Pure and Applied Mathematics.

David L. Donoho and Iain Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90(432):1200–1224, 1995.

Harish Doraiswamy, Nivan Ferreira, Theodoros Damoulas, Juliana Freire, and Claudio Silva. Using topological analysis to support event-guided exploration in urban data. *Visualization and Computer Graphics, IEEE Transactions on*, PP(99), 2014.

Uriel Feige and Eran Ofek. Spectral techniques applied to sparse random graphs. *Random Structures & Algorithms*, 27(2):251–275, 2005.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98): 298–305, 1973.

Sergei K. Godunov and Viktor S. Ryabenkii. *Difference Schemes: An Introduction to the Underlying Theory*. Elsevier, Amsterdam, 1987. Number 19 in Studies in Mathematics and Its Applications.

Holger Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4):984–1006, 2010.

Jeff Irion. *Multiscale Transformations for Signals on Graphs: Methods and Applications*. PhD thesis, Department of Mathematical Sciences, University of California at Davis, 2015.

Jonathan Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving SDD systems in nearly-linear time. *Proceedings of the ACM Annual Symposium on Theory of Computing (STOC-13)*, 45:911–920, 2013.

Seung-Jean Kim, Kwangmoo Koh, Stephen Boyd, and Dimitry Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

Risi Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete structures. In *International Conference on Machine Learning (ICML-02)*, pages 315–322, San Francisco, CA, 2002. Morgan Kaufmann.

Ioannis Koutis, Gary Miller, and Richard Peng. A nearly-$m \log n$ time solver for SDD linear systems. *Proceedings of the IEEE Annual Symposium on Foundations of Computer Science (FOCS-11)*, 52:590–598, 2011.

Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3): 261–277, 1988.

Enno Mammen and Sara van de Geer. Locally adaptive regression splines. *Annals of Statistics*, 25 (1):387–413, 1997.

Adam W Marcus, Daniel A Spielman, and Nikhil Srivastava. Ramanujan graphs and the solution of the Kadison-Singer problem. arXiv: 1408.4421, 2014.

Julian McAuley and Jure Leskovec. Learning to discover social circles in ego networks. *Advances in Neural Information Processing Systems*, 25, 2012.

Aaditya Ramdas and Ryan J Tibshirani. Fast and flexible admm algorithms for trend filtering. *Journal of Computational and Graphical Statistics*, 2015.

Leonid I. Rudin, Stanley Osher, and Emad Faterni. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60:259–268, 1992.

Simon Setzer, Gabriel Steidl, and Tanja Teuber. Infimal convolution regularizations with discrete l1-type functionals. *Communications in Mathematical Science*, 9(3):797–827, 2011.

James Sharpnack, Alessandro Rinaldo, and Aarti Singh. Sparsistency via the edge lasso. *International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, 15:1028–1036, 2012.

James Sharpnack, Aarti Singh, and Akshay Krishnamurthy. Detecting activations over graphs using spanning tree wavelet bases. In *International Conference on Artificial Intelligence and Statistics (AISTATS-13)*, volume 16, pages 536–544, 2013a.

James Sharpnack, Aarti Singh, and Alessandro Rinaldo. Changepoint detection over graphs with the spectral scan statistic. In *International Conference on Artificial Intelligence and Statistics (AISTATS-13)*, volume 16, pages 545–553, 2013b.

David Shuman, Sunil Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

Alexander Smola and Risi Kondor. Kernels and regularization on graphs. In Bernhard Scholkopf and Manfred Warmuth, editors, *Learning Theory and Kernel Machines*, volume 2777 of *Lecture Notes in Computer Science*, pages 144–158. Springer, Berlin, 2003.

Daniel Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. *Proceedings of the ACM Annual Symposium on Theory of Computing (STOC-04)*, 36:81–90, 2004.

Gabriel Steidl, Stephan Didas, and Julia Neumann. Splines in higher order TV regularization. *International Journal of Computer Vision*, 70(3):214–255, 2006.

Partha Pratim Talukdar and Koby Crammer. New regularized algorithms for transductive learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 442–457. Springer, 2009.

Partha Pratim Talukdar and Fernando Pereira. Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1473–1481. Association for Computational Linguistics, 2010.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B*, 67(1):91–108, 2005.

Ryan J. Tibshirani. Adaptive piecewise polynomial estimation via trend filtering. *Annals of Statistics*, 42(1):285–323, 2014.

Ryan J. Tibshirani and Jonathan Taylor. The solution path of the generalized lasso. *Annals of Statistics*, 39(3):1335–1371, 2011.

Ryan J. Tibshirani and Jonathan Taylor. Degrees of freedom in lasso problems. *Annals of Statistics*, 40(2):1198–1232, 2012.

Sara van de Geer. Estimating a regression function. *Annals of Statistics*, 18(2):907—924, 1990.

Sara van de Geer and Johannes Lederer. The lasso, correlated design, and improved oracle inequalities. *IMS Collecions*, 9:303–316, 2013.

Nisheeth Vishnoi. $Lx = b$: Laplacian solvers and their algorithmic applications. *Foundations and Trends in Theoretical Computer Science*, 8(1–2):1–141, 2012.

Yu-Xiang Wang, Alex Smola, and Ryan J. Tibshirani. The falling factorial basis and its statistical properties. In *International Conference on Machine Learning (ICML-14)*, volume 31, 2014.

Journal of Machine Learning Research 17 (2016) 1-37 Submitted 4/15; Revised 11/15; Published 6/16

Multi-Task Learning for Straggler Avoiding Predictive Job Scheduling

Neeraja J. Yadwadkar neerajay@eecs.berkeley.edu
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Bharath Hariharan bharath2@eecs.berkeley.edu
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Joseph E. Gonzalez jegonzal@eecs.berkeley.edu
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Randy Katz randy@cs.berkeley.edu
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

Parallel processing frameworks (Dean and Ghemawat, 2004) accelerate jobs by breaking them into tasks that execute in parallel. However, slow running or *straggler* tasks can run up to 8 times slower than the median task on a production cluster (Ananthanarayanan et al., 2013), leading to delayed job completion and inefficient use of resources. Existing straggler mitigation techniques wait to detect stragglers and then relaunch them, delaying straggler detection and wasting resources. We built Wrangler (Yadwadkar et al., 2014), a system that predicts when stragglers are going to occur and makes scheduling decisions to avoid such situations. To capture node and workload variability, Wrangler built separate models for every node and workload, requiring the time-consuming collection of substantial training data. In this paper, we propose multi-task learning formulations that share information between the various models, allowing us to use less training data and bring training time down from 4 hours to 40 minutes. Unlike naive multi-task learning formulations, our formulations capture the shared structure in our data, improving generalization performance on limited data. Finally, we extend these formulations using group sparsity inducing norms to automatically discover the similarities between tasks and improve interpretability.

1. Introduction

Distributed processing frameworks, such as (Dean and Ghemawat, 2004; Isard et al., 2007; Li, 2009), split a data intensive computation job into multiple smaller tasks, which are then executed in parallel on commodity clusters to achieve faster job completion. A natural consequence of such a parallel execution model is that the slow-running tasks, commonly called *stragglers*, potentially delay overall job completion. In a recent study, Ananthanarayanan et al. (2013) show that straggler tasks are on an average 6 to 8 times slower than the median task of the corresponding job, despite existing mitigation techniques. Suri and Vassilvitskii (2011) describe the intensity of straggler problem as the "curse of the last reducer", causing the last 1% of the computation to take much longer to finish execution. Mitigating stragglers thus remains an important problem. In this paper, our focus is on machine learning approaches for predicting and avoiding these stragglers.

Existing approaches to straggler mitigation, whether reactive or proactive, fall short in multiple ways. Commonly used reactive approaches such as speculative execution (Dean and Ghemawat, 2004), act after the tasks have already slowed down. Proactive approaches that launch redundant copies of a task in a hope that at least one of them will finish in a timely manner (Ananthanarayanan et al., 2013), waste resources. It is hard to, *a priori*, identify which factors, and configurations lead to stragglers, motivating a data driven machine learning approach to straggler prediction. Bortnikov et al. (2012) explored predictive models for predicting slowdown for tasks. But they learn a single model for a cluster of nodes, ignoring the variability caused due to the heterogeneity across nodes and across jobs from different workloads. Moreover, they built models but did not show if these models could indeed improve job completion times, which is what matters ultimately. Gupta et al. (2013) explored learning based job scheduling by matching node capabilities to job requirements. However, their approach is not designed to deal with stragglers effectively because it does not model the time-varying state of the nodes (for instance, how busy they are). Moreover, the effectiveness of these methods in reducing job-completion times in real world clusters hasn't been shown. To this end, we built Wrangler (Yadwadkar et al., 2014), a system that builds predictive models of straggler behavior and incorporates this model in the scheduler, improving the 99^{th} percentile job completion time by up to 61% as compared to speculative execution for real world production level workloads[1] from Facebook and Cloudera's customers.

Proactive model based approaches including Wrangler, have previously treated each workload and even compute node as a separate straggler estimation task with independent models. The decision to model each workload and node independently is motivated by the observation that the resource contention patterns that cause stragglers can vary from node to node and workload to workload. For a detailed analysis about causes of stragglers across nodes and workloads, see Section 4.2.2. of Wrangler, Yadwadkar et al. (2014). To address such heterogeneity in the nodes and changing workload patterns, training separate models for each workload being run on each node, is considered necessary. However, such independent models pose two critical challenges: (1) each new node and workload requires new training data which can take hours to collect, delaying the application of model based scheduling, and (2) clusters with many nodes may only have limited data for a given workload on each node leading to lower quality models.

These shortcomings can be addressed if each classifier is able to leverage information gleaned at other nodes and from other workloads. For instance, when there is not enough

1. Clusters are used for different purposes, and statistics such as the kinds of jobs submitted, their resource requirements and the frequency at which they are submitted vary depending upon the usage. We call one such distribution of jobs a workload. Section 3.2 explains the notion of workloads in further detail.

data at a node for a workload, we can gain from the data collected at that node while it was executing other workloads, or from other nodes running the same workload. Such information sharing falls in the ambit of multi-task learning (MTL), where the learner is embedded in an environment of related tasks, and the learner's aim is to leverage similarities between the tasks to improve performance of all tasks.

In this paper, we propose an MTL formulation for learning predictors that are more accurate and generalize better than existing straggler predicting models. Basic MTL formulations often assume only limited correlation structure between learning tasks. However, our application has an additional overlapping group structure: models built on the same node for different workloads can be grouped together, as can be the models corresponding to the same workload running on different nodes. We propose a new formulation that leverages this overlapping group structure, and show that doing so significantly reduces the number of parameters and improves generalization to tasks with very little data.

In applications such as ours, high accuracy is not the only objective: we also want to be able to gain some insight into what is causing these stragglers. Such insight can aid in debugging root causes of stragglers and system performance degradation. Automatically learned classifiers can be opaque and hard to interpret. To this end, we propose group sparsity inducing mixed-norm regularization on top of our basic MTL formulation to automatically reveal the group structure that exists in the data, and hopefully provide insights into how straggler behavior is related across nodes and workloads. We also explore sparsity-inducing formulations based on feature selection that attempt to reveal the characteristics of straggler behavior.

For our experimental evaluation, we use a collection of real-world workloads from Facebook and Cloudera's customers. We evaluate not only the prediction accuracy but also the improvement in job completion times, which is the metric that directly impacts the end user. We show that our formulation to predict stragglers allows us to reduce job completion times by up to 59% over the previous state-of-the-art learning base system, Wrangler (Yadwadkar et al. (2014)). This large reduction arises from a 7 point increase in prediction accuracy. Further, we can get equal or better accuracy than Wrangler (Yadwadkar et al. (2014)) using a sixth of the training data, thus bringing the training time down from 4 hours to about 40 minutes. We also provide empirical evidence that compared to naive MTL formulations, our formulation indeed generalizes to new tasks, i.e, it is significantly better at predicting straggler behavior for nodes for which we have not collected enough data.

Finally, while our initial motivation and experimental validation focus on the straggler avoidance problem, our learning formulations are general and can be applied to other systems that train node or workload dependent classifiers (Gupta et al., 2013; Delimitrou and Kozyrakis, 2014). For instance, ThroughputScheduler (Gupta et al., 2013) uses such classifiers to allot resources to tasks, and can benefit from such multitask reasoning. We leave these extensions to future work.

To summarize, our key contributions are:

1. An MTL formulation of the straggler estimation problem that allows us to use less data and reduce parameters, improving generalization (Section 4.1).

2. A mixed-norm group sparsity inducing formulation, that automatically detects group structure (Section 4.3) facilitating interpretability.

3. An efficient optimization technique based on a reduction to the standard SVM (Cortes and Vapnik (1995)) formulation (Section 4.2).

4. A comprehensive evaluation as part of a complete system for predicting and avoiding stragglers in real world production cluster traces (Section 5). We show that, compared to existing approaches, our formulations

   (a) avoid stragglers better with 7% improved prediction accuracy, improving job completion times significantly with up to 57.8% improvement in the 99^{th} percentile, and reducing net resource usage by up to 40%, and

   (b) can work even with a sixth of the training data and thus a much shorter training period, reducing the training data collection time significantly from 4 hours to 40 minutes.

In what follows, we first give some background on stragglers in Section 2. We then describe Wrangler and discuss its shortcomings in Section 3. In Section 4, we describe our multi-task learning formulations. In Section 5, we empirically evaluate our formulations on real world production level traces from Facebook and Cloudera's customers. We end with a discussion of related work.

2. Background and Motivation

The growing popularity of Internet-based applications, in addition to reduced costs of storage media, has resulted in generation of data at an unprecedented scale. Analytics on such huge datasets has become a driving factor for business. Parallel processing on commodity clusters has emerged as the de-facto way of dealing with the scale and complexity of such data-intensive applications. Dean and Ghemawat (2004) originally proposed the MapReduce framework at Google to process enormous amounts of data. MapReduce is highly scalable to large clusters of inexpensive commodity computers. Hadoop (White, 2009), an open source implementation of MapReduce, has been widely adopted by industry over the last decade.

A data intensive application is submitted to a cluster of commodity computers as a MapReduce *job*. To accelerate completion, MapReduce divides a job into multiple *tasks*. A cluster scheduler assigns these to machines (nodes), where they are executed in parallel. A job finishes when all its tasks have finished execution. A key benefit of such frameworks is that they automatically handle failures (which are more likely to occur on a cluster of commodity computers) without needing extra efforts from the programmer. Two basic modes of failures are the failure of a node and the failure of a task. If a node crashes, MapReduce re-runs all the tasks it was executing on a different node. If a task fails, MapReduce automatically re-launches it.

However, a tricky situation arises when a node is available, but is performing poorly. This causes the tasks scheduled on that node to execute slower than other tasks of the same job scheduled on other nodes in the cluster. Since a job finishes execution only when all its

tasks have finished execution, such slow-running tasks, called *stragglers*, extend the job's completion time. This, in turn, leads to increased user costs.

Stragglers abound in the real world. According to Ananthanarayanan et al. (2013), if stragglers did not exist in real-world production clusters, the average job completion times would have been improved by 47%, 29% and 36% in the Facebook, Bing and Yahoo traces respectively. We observed that 22-28% of the total tasks are stragglers in a replay[2] of Facebook and Cloudera's customers' Hadoop production cluster traces. Thus, stragglers are a major hurdle in achieving faster job completions.

It is challenging to deal with stragglers (Dean and Ghemawat, 2004; Zaharia et al., 2008; Ananthanarayanan et al., 2010). Dean and Ghemawat (2004) mentioned that stragglers could arise due to various reasons, such as competition for resources, problematic hardware, and mis-configurations. Ananthanarayanan et al. (2010) report that consistently slow nodes are rarely the reason behind stragglers; the majority are caused by transient node behavior. The reason is simple: cluster schedulers assign multiple tasks, belonging to the same or different jobs, to be executed simultaneously on a node. Depending on the node's characteristics and current load, as well as the characteristics of the workloads, these jobs may result in counterproductive resource contention patterns. Moreover, this gives rise to dynamically changing task execution environments, causing large variations in task execution times. We cannot simply look at the initial configuration of the node and decide if it will cause stragglers. We must track the continuously changing state of the node (its memory usage, cpu usage etc.) and use that to predict straggler behavior.

Above and beyond this temporal variability is the variability across nodes and workloads. If the scheduler assigns memory hungry tasks on a memory-constrained node, stragglers could occur due to contention for memory. While for another node that has slower disk, straggler behavior will primarily depend on how many I/O-intensive tasks are being run. To predict straggler behavior, we need to consider both the current state of each node and tailor the decision to the node type (its resource capabilities) and particular kind of tasks being executed (their resource requirements). In our experiments, we find that models that do not take this into account do about 6-7% worse than models that do (see Table 5 in Section 5.3).

Due to these difficulties in understanding the causes behind stragglers, and the challenges involved in predicting stragglers, initial attempts at mitigating stragglers have been *reactive* (Dean and Ghemawat, 2004; Zaharia et al., 2008; Ananthanarayanan et al., 2010). Dean and Ghemawat (2004) suggested *speculative execution* as a mitigation mechanism for stragglers. This is a reactive scheme that is dominantly used on production clusters including those at Facebook and Microsoft Bing (Ananthanarayanan et al., 2014). It operates in two steps: (1) wait-and-speculate if a task is executing slower than other tasks of the same job, and (2) replicate or spawn multiple redundant copies of such tasks hoping a copy will reach completion before the original. As soon as one of the copies or the original task finishes execution, the rest are killed.

The benefits of Speculative execution, in terms of improved job completion times at the cost of resources consumed, are unclear. Due to the wait-and-speculate step, this scheme is inefficient in time, leading to a delay before speculation begins. Also, due to the second

2. Please see Chen et al. (2012) for details on the faithful replay of production level traces.

Trace	% of speculatively executed tasks that were killed
Facebook 2009 (FB2009)	77.9%
Facebook 2010 (FB2010)	88.6%
Cloudera's Customer b (CC_b)	74.4%
Cloudera's Customer e (CC_e)	48.8%

Table 1: Majority of the speculatively executed tasks eventually get killed since the original task finishes execution before them.

step that replicates tasks, such mechanisms lead to increased resource consumption without necessarily gaining performance benefits. Moreover, we observed that the original task, that was marked as a straggler in the wait-and-speculate step, often finishes execution before any of the redundant copies launched in the second step. Thus, a significant number of such redundant copies end up getting killed, resulting in wastage of resources and perhaps additional unnecessary contention. Table 1 shows that about 48% to 88% of the speculatively executed copies were killed in our replay of Facebook and Cloudera's customers' production cluster traces. LATE (Zaharia et al., 2008) improves over speculative execution using a notion of progress scores, but still results in resource wastage due to replication.

Since reactive mechanisms fall short of efficiently mitigating stragglers, some proactive approaches have been proposed. Dolly (Ananthanarayanan et al., 2013) is a cloning mechanism that avoids the wait-and-speculate phase of speculative execution and immediately launches redundant copies of tasks belonging to small jobs. However, being replication-based, it also incurs resource overhead.

Machine learning has shown promise in dealing with the challenges of estimating task completion times and predicting stragglers in parallel processing environments (Bortnikov et al., 2012; Gupta et al., 2013). We built Wrangler (Yadwadkar et al., 2014) (see Section 3), a system that learns to predict nodes that might create stragglers and uses these predictions as hints to the scheduler so as to avoid creating stragglers by rejecting bad placement decisions. Thus, being proactive, Wrangler is time efficient. Also, by smarter scheduling, we avoid replication of straggler tasks. Thus, Wrangler is also efficient in terms of reducing the resources consumed. Wrangler was the first complete system that demonstrated the utility of learning methods in reducing job-completion times in real world clusters.

In the following sections, we review Wrangler, describe its architecture, discuss its limitations and avenues for improvements. Then in Section 4, we present our multi-task learning based approach that improves upon Wrangler by resolving its limitations.

3. Wrangler

Wrangler is a system that predicts stragglers based on cluster resource usage counters and uses these predictions to inform scheduling decisions. Wrangler achieves this by adding two main components to the scheduling system of data intensive computational frameworks: (i) model-builder, and (ii) predictive scheduler. Figure 1 summarizes Wrangler's architecture. The model-builder learns to predict stragglers using cluster resource usage counters. We explain this component in more detail in the following Sections 3.1, 3.2, and 3.3. Then, every time a task is scheduled to run on a particular node, Wrangler makes a prediction

using these models, for whether the task will become a straggler. It then uses this prediction to modify the scheduling decisions if straggler behavior is predicted. Section 3.4 provides a brief explanation of the predictive scheduler.

3.1 Features and labels

To predict whether scheduling a task at a particular node will lead to straggler behavior, Wrangler uses the resource usage counters at the node. Dean and Ghemawat (2004) mentioned that stragglers could arise due to various reasons such as competition for CPU, memory, local disk, network bandwidth. Zaharia et al. (2008) further suggest that stragglers could be caused due to faulty hardware and misconfiguration. Ananthanarayanan et al. (2010) report that the dynamically changing resource contention patterns on an underlying node could give rise to stragglers. Based on these findings, we collected the performance counters for CPU, memory, disk, network, and other operating system level counters describing the degree of concurrency before launching a task on a node. The counters we collected span multiple broad categories as follows:

1. *CPU utilization:* CPU idle time, system and user time and speed of the CPU, etc.
2. *Network utilization:* Number of bytes sent and received, statistics of remote read and write, statistics of RPCs, etc.
3. *Disk utilization:* The local read and write statistics from the datanodes, amount of free space, etc.
4. *Memory utilization:* Amount of virtual, physical memory available, amount of buffer space, cache space, shared memory space available, etc.
5. *System-level features:* Number of threads in different states (waiting, running, terminated, blocked, etc.), memory statistics at the system level.

In total, we collect 107 distinct features characterizing the state of the machine. See Section 5.1 for details of our dataset.

Features: Multiple tasks from jobs of each workload may be run on each node. Therefore to simplify notation, we index the execution of a particular task by i and define $S_{n,l}$ as the set of tasks corresponding to workload l executed on node n. *Before* executing task $i \in S_{n,l}$ corresponding to workload l on node n we collect the resource usage counters described above on node n to form the feature vector $x_i \in \mathbb{R}^{107}$. For each feature described above we subtract the minimum across the entire dataset and rescale so that it lies between 0 and 1 for the entire dataset.

Labels: After running task i we measure the normalized task duration $nd(i)$ which is the ratio of task execution time to the amount of work done (bytes read/written) by task i. From the normalized duration, we determine whether a task has *straggled* using the following definition:

Definition 1 *A task i of a job J is called a straggler if*

$$nd(i) > \beta \times \underset{\forall j \in J}{\text{median}}\{nd(j)\} \tag{1}$$

where $nd(i)$ is the normalized duration of task i computed as the ratio of task execution time to the amount of work done (bytes read/written) by task i.

In this paper, as in Wrangler, we set β to 1.3. Given the definition of a straggler, for task i we define $y_i \in \{0, 1\}$ as a binary label indicating whether the corresponding task i ended up being a straggler relative to other tasks in the same job.

3.2 Prediction Task

While the resource usage counters do track the time-varying state of the node, they do not model the variability across nodes, or the properties of the particular task we are executing. To deal with the variability of nodes, Wrangler builds a separate predictor for each node.

Modeling the variability across tasks is harder since it would require understanding the code that the task is executing. Instead, Wrangler uses the notion of "workloads", which we define next. Companies such as Facebook, Google, use compute clusters for various computational purposes. The specific pattern of execution of jobs on these clusters is called a *workload*. These workloads are specified using various statistics, such as submission times of multiple jobs, number of their constituent tasks along with their input data sizes, shuffle sizes, and output sizes. Wrangler assumes that all tasks in a particular workload have similar properties in terms of resource requirements etc., and it captures the variability across workloads by building separate predictors for each workload. Thus Wrangler builds separate predictors for each node and for every workload.

Putting it all together, we can state the binary classification problem for predicting straggler tasks more formally as follows. A datapoint in our setting corresponds to a task i of job J from a particular workload l that is executed on a node n in our cluster. Before running task i we collect the features x_i which characterize the state of node n. After running task i we then measure the normalized duration and determine whether the task straggled with respect to other tasks of job J (see Definition 1). Our goal is to learn a function:

$$f_{n,l} : x \longrightarrow y,$$

for each node n and workload l that maps the current characteristics $x \in \mathbb{R}^d$ of that node (e.g., current CPU utilization) to the binary variable $y \in \{0, 1\}$ indicating whether that task will straggle (i.e., run longer than β times the median runtime).

3.3 Training

For any workload l, Wrangler first collects data and ground truth labels for training and validation. In particular, for every task i launched on node n, Wrangler records both the resource usage counters x_i and the label y_i which indicates if the task straggles or not. Since there is a separate predictor for each node and workload, Wrangler produces separate datasets for each node and workload. Let $S_{n,l}$ be the set of tasks of jobs corresponding to workload l, executed on node n. Thus, we record the dataset for node n, workload l as:

$$D_{n,l} = \{(x_i, y_i) : i \in S_{n,l}\}.$$

Then Wrangler divides each dataset into a training set and test set temporally, i.e, the first few jobs constitute the train set and the rest form the validation. The predictors are

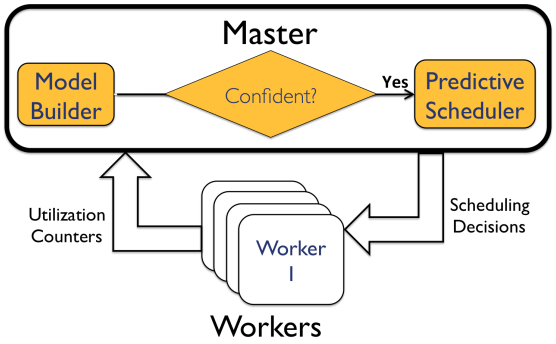


Figure 1: Architecture of *Wrangler*.

then trained on these datasets. Stragglers, by definition are fewer than the non-straggler tasks. Therefore for training, Wrangler statistically oversampled the class of stragglers to avoid the skew in the strengths of the two classes. This gets the two classes represented equally.

After this initial training phase, the classifiers are frozen, and the trained classifiers are incorporated into the job scheduler as described above. They are then tested on the next $T_{test} = 10$ hours by measuring the impact of these classifiers on job completion times. Further details about the train and test splits are in Section 5.

3.4 Model-aware scheduling

Job scheduling in Hadoop is handled by a master, which controls the workers. The master assigns tasks to the worker nodes. The assignments depend upon the number of available slots as well as data-locality. Wrangler modifies this scheduler to incorporate its predictions. Before launching a task i of a job coming from a workload l, on a node n, the scheduler collects the node's resource usage counters x_i and runs the classifier $f_{n,l}$. If the classifier predicts that the task will be a straggler with high enough "confidence" (Wrangler uses SVMs with scaling by Platt (1999) to produce a confidence), the scheduler does not assign the task to that node. It is later assigned to a node that is not predicted to create a straggler. See Yadwadkar et al. (2014) for details on the algorithm and implementation.

3.5 Shortcomings of Wrangler and avenues for improvements

Due to the heterogeneity of nodes in a cluster, the model builder trains a separate classifier for each node. Note that to build a training set per node, every node should have executed sufficient number of tasks. Wrangler takes a few hours (approximately 2-4 hours, depending on the workload) for this process. Additionally, because each workload might be different, these models are retrained for every new workload. Thus, for every new workload that is executed on the cluster, there is a 2-4 hour model building period. In typical large

production clusters with tens of thousands of nodes, it might be a long time before a node collects enough data to train a classifier. (In Table 5 we show that the prediction accuracy of Wrangler rapidly degrades as the amount of training data is reduced). In some cases, workloads may only be run a few times in total, limiting the ability of systems like Wrangler to make meaningful predictions.

Alternatively, large clusters with locality aware scheduling can lead to poor sample coverage. Recall that, in our case, each task of a workload executed on a node amounts to a training data point. The placement of input data on nodes in a cluster is managed by the underlying distributed file system (Ghemawat et al., 2003). To achieve locality for faster reading of input data, sophisticated locality-aware schedulers (Zaharia et al., 2008, 2010) try to assign tasks to nodes already having the appropriate data. Based on the popularity of the data, number of tasks assigned to a node could vary. Hence, we may not get uniform number of training data points, i.e., tasks executed, across all the nodes in a cluster. There could be other reasons behind skewed assignment of tasks to nodes (Kwon et al., 2012): even when every map task has the same amount of data, a task may take longer depending on the code path it takes based on the data it processes. Hence, the node slots will be busy due to such long running tasks. This could lead to some nodes executing fewer tasks than others.

These observations suggest that our modeling framework needs to be robust to limited and potentially skewed data. Thus, there is a need for straggler prediction models (1) that can be trained using minimum available data, and (2) that generalize to unseen nodes or workloads. In the following section, we describe our multi-task learning based approach with these goals for avoiding stragglers. We evaluate it using real world production level traces from Facebook and Cloudera's customers, and compare the gains with Wrangler.

4. Multi-task learning for straggler avoidance

As described in the previous section, Wrangler builds separate models for each workload and for every node. Thus, every {node, workload} tuple is a separate learning problem. However, learning problems corresponding to different workloads executed on the same node clearly have something in common, as do learning tasks corresponding to different nodes executing the same workload. We want to use this shared structure between the learning problems to reduce data collection time.

Concretely, a task executing on a node will be a straggler because of a combination of factors. Some of these factors involve the properties of the node where the task is executing (for instance, the node may be memory-constrained) and some others involve particular requirements that the tasks might have in terms of resources (for instance, the task may require a lot of memory). These are workload-related factors. When collecting data for a new workload executing on a given node, one must be able to use information about the workload collected while it executed on other nodes, and information collected about the node collected while it executed other workloads.

We turn to multi-task learning to leverage this shared structure. In the terminology of multi-task learning, each {node, workload} pair forms a separate learning-task[3] and these learning problems have a shared structure between them. However, unlike typical MTL formulations, our learning-tasks are not simply correlated with each other; they share a specific structure, clustering along node- or workload-dependent axes. In what follows, we first describe a general MTL formulation that can capture such learning-task grouping. We then detail how we apply this formulation to our application of straggler avoidance.

4.1 Partitioning tasks into groups

Suppose there are T learning tasks, with the training set for the t-th learning-task denoted by $D_t = \{(\mathbf{x}_{it}, y_{it}) : i = 1, \ldots, k_t\}$, with $\mathbf{x}_{it} \in \mathbb{R}^d$. We begin with the formulation proposed by Evgeniou and Pontil (2004). Evgeniou, et al. proposed a basic hierarchical regression formulation with the linear model, $\mathbf{w}_t$, for learning-task t as:

$$\mathbf{w}_t = \mathbf{w}_0 + \mathbf{v}_t \tag{2}$$

The intuition here is that $\mathbf{w}_0$ captures properties common to all learning-tasks, while $\mathbf{v}_t$ captures how the individual learning-tasks deviate from $\mathbf{w}_0$.

We then frame learning in the context of empirical loss minimization. Given the variability in node behavior and the need for robust predictions we adopt the hinge-loss and apply l_2 regularization to both the base w_0 and learning-task specific weights v_t:

$$\min_{\mathbf{w}_0, \mathbf{v}_t, b} \lambda_0 \|\mathbf{w}_0\|^2 + \frac{\lambda_1}{T} \sum_{t=1}^{T} \|\mathbf{v}_t\|^2 + \sum_{t=1}^{T} \sum_{i=1}^{k_t} L_{Hinge}\left(y_{it}, (\mathbf{w}_0 + \mathbf{v}_t)^T \mathbf{x}_{it} + b\right) \tag{3}$$

where $L_{Hinge}(y_{it}, s_{it}) = \max(0, 1 - y_{it} s_{it})$ is the hinge loss.

In the above formulation, all learning-tasks are treated equivalently. However, as discussed above, in our application some learning-tasks naturally cluster together. Suppose that the learning-tasks cluster into G non-overlapping *groups*, with the t-th learning-task belonging to the $g(t)$-th group. Note that while we derive our formulations for non-overlapping groups, which is true in our application, the modification for overlapping groups is trivial. Using the same intuition as Equation 2, we can write the classifier $\mathbf{w}_t$ as:

$$\mathbf{w}_t = \mathbf{w}_0 + \mathbf{v}_t + \mathbf{w}_{g(t)} \tag{4}$$

In general, there may be more than one way of dividing our learning-tasks into groups. In our application, one may split learning-tasks into groups based on workload or based on nodes. We call one particular way of dividing learning-tasks into groups a *partition*. The p-th partition has G_p groups, and the learning-task t belongs to the $g_p(t)$ group under this partition. Now, we also have a separate set of weight vectors for each partition p, and the weight vector of the g-th group of the p-th partition is denoted by $\mathbf{w}_{p,g}$. Then, we can write the classifier $\mathbf{w}_t$ as:

$$\mathbf{w}_t = \mathbf{w}_0 + \mathbf{v}_t + \sum_{p=1}^{P} \mathbf{w}_{p,g_p(t)} \tag{5}$$

3. The machine learning notion of a "task" as a learning problem differs from the cloud computing notion of a "task" as a part of a job that is run in parallel. The intended meaning should be clear from the context.

Finally, note that $\mathbf{w}_0$ and $\mathbf{v}_t$ can also be seen as weight vectors corresponding to trivial partitions: $\mathbf{w}_0$ corresponds to the partition where all learning-tasks belong to a single group, and $\mathbf{v}_t$ corresponds to the partition where each learning-task is its own group. Thus, we can include $\mathbf{w}_0$ and $\mathbf{v}_t$ in our partitions and write Equation 5 as:

$$\mathbf{w}_t = \sum_{p=1}^{P} \mathbf{w}_{p,g_p(t)} \tag{6}$$

Intuitively, at test time, we get the classifier for the t-th learning-task by summing weight vectors corresponding to each group to which t belongs.
As in Equation 3, the learning problem involves minimizing the sum of l_2 regularizers on each of the weight vectors and the hinge loss:

$$\min_{\mathbf{w}_{p,g}, b} \sum_{p=1}^{P} \sum_{g=1}^{G_p} \lambda_{p,g} \|\mathbf{w}_{p,g}\|^2 + \sum_{t=1}^{T} \sum_{i=1}^{k_t} L_{Hinge}\left(y_{it}, \left(\sum_{p=1}^{P} \mathbf{w}_{p,g_p(t)}\right)^T \mathbf{x}_{it} + b\right) \tag{7}$$

Here, the regularizer coefficient $\lambda_{p,g} = \frac{\lambda_p \#(p,g)}{T}$, where $\#(p,g)$ denotes the number of learning-tasks assigned to the g-th group of the p-th partitioning. The scaling factor $\frac{\lambda_p \#(p,g)}{T}$ interpolates smoothly between λ_0, when all learning-tasks belong to a single group, and $\frac{\lambda_1}{T}$, when each learning-task is its own group. Lowering λ_p for a particular partition will reduce the penalty on the weights of the p-th partition and thus cause the model to rely more on the p-th partitioning. For the base partition $p = 0$, setting $\lambda_p = 0$ would thus favor as much parameter sharing as feasible.

4.2 Reduction to a standard SVM

One advantage of the formulation we use is that it can be reduced to a standard SVM (Cortes and Vapnik, 1995), allowing the usage of off-the-shelf SVM solvers. Below, we show how this reduction can be achieved. Given λ, for every group g of every partition p, define:

$$\tilde{\mathbf{w}}_{p,g} = \sqrt{\frac{\lambda_{p,g}}{\lambda}} \mathbf{w}_{p,g} \tag{8}$$

Now concatenate these vectors into one large weight vector $\tilde{\mathbf{w}}$:

$$\tilde{\mathbf{w}} = [\tilde{\mathbf{w}}_{1,1}^T, \ldots, \tilde{\mathbf{w}}_{p,g}^T, \ldots, \tilde{\mathbf{w}}_{P,G_P}^T]^T \tag{9}$$

Then, it can be seen that $\lambda \|\tilde{\mathbf{w}}\|^2 = \sum_{p=1}^{P} \sum_{g=1}^{G_p} \lambda_{p,g} \|\mathbf{w}_{p,g}\|^2$. Thus, with this change of variables, the regularizer in our optimization problem resembles a standard SVM. Next, we transform the data points $\mathbf{x}_{it}$ into $\phi(\mathbf{x}_{it})$ such that we can replace the scoring function with $\tilde{\mathbf{w}}^T \phi(\mathbf{x}_{it})$. This transformation is as follows. Again, define:

$$\phi_{p,g}(\mathbf{x}_{it}) = \delta_{g_p(t),g} \sqrt{\frac{\lambda}{\lambda_{p,g}}} \mathbf{x}_{it} \tag{10}$$

Here, $\delta_{g_p(t),g}$ is a kronecker delta, which is 1 if $g_p(t) = g$ (i.e. , if the learning-task t belongs to group g in the p-th partitioning) and 0 otherwise. Our feature transformation is then the concatenation of all these vectors:

$$\phi(\mathbf{x}) = [\phi_{1,1}(\mathbf{x})^T, \ldots, \phi_{p,g}(\mathbf{x})^T, \ldots, \phi_{P,G_P}(\mathbf{x})^T]^T \tag{11}$$

It is easy to see that:

$$\tilde{\mathbf{w}}^T \phi(\mathbf{x}_{it}) = \left(\sum_{p=1}^{P} \mathbf{w}_{p,g_p(t)} \right)^T \mathbf{x}_{it} \tag{12}$$

Intuitively, $\tilde{\mathbf{w}}$ concatenates all our parameters with their appropriate scalings into one long weight vector, with one block for every group of every partitioning. $\phi(\mathbf{x}_{it})$ transforms a data point into an equally long feature vector, by placing scaled copies of $\mathbf{x}_{it}$ in the appropriate blocks and zeros everywhere else.

With these transformations, we can now write our learning problem as :

$$\min_{\tilde{\mathbf{w}},b} \lambda \|\tilde{\mathbf{w}}\|^2 + \sum_{t=1}^{T} \sum_{i=1}^{k_t} L_{Hinge}(y_{it}, \tilde{\mathbf{w}}^T \phi(\mathbf{x}_{it}) + b) \tag{13}$$

which corresponds to a standard SVM. In practice, we use this transformation and change of variables, both at train time and at test time.

4.3 Automatically selecting groups or partitions

In many real world applications including our own, good classification accuracy is not an end in itself. Model interpretability is of critical importance. A large powerful classifier working on tons of features may do a very good job of classification, but will not provide any insight to an engineer trying to identify the root causes of a problem. It will also be difficult to debug such a classifier if and when it fails. In the absence of sufficient training data, large models may also overfit. Thus, interpretability and simplicity of the model are important goals. In the context of the formulation above, this means that we want to keep only a minimal set of groups/partitions while maintaining high classification accuracy.

We can use mixed l_1 and l_2 norms to induce a sparse selection of groups or partitions (Bach et al., 2011). Briefly, suppose we are given a long weight vector $\mathbf{w}$ divided into M blocks, with the m-th block denoted by $\mathbf{w}_m$. Then the squared mixed l_1 and l_2 norm is:

$$\Omega(\mathbf{w}) = \left(\sum_{m=1}^{M} \|\mathbf{w}_m\|_2 \right)^2 \tag{14}$$

This can be considered as an l_1 norm on a vector with elements $\|\mathbf{w}_m\|_2$. When $\Omega(\cdot)$ is used as a regularizer, the l_1 norm will force some elements of this vector to be set to 0, which in turn would force the corresponding *block* of weights $\mathbf{w}_m$ to be set to 0. It thus encourages entire blocks of the weight vector to be set to 0, a sparsity pattern that is often called "group sparsity".

In our context, our weight vector $\tilde{\mathbf{w}}$ is made up of $\tilde{\mathbf{w}}_{p,g}$. We consider two alternatives:

1. We can put all the weight vectors corresponding to a single partition in the same block. Then, the regularizer becomes:

$$\Omega_{ps}(\tilde{\mathbf{w}}) = \left(\sum_{p=1}^{P} \|\tilde{\mathbf{w}}_p\|_2 \right)^2 \tag{15}$$

   where $\tilde{\mathbf{w}}_p = [\tilde{\mathbf{w}}_{p,1}^T, \ldots, \tilde{\mathbf{w}}_{p,G_p}^T]^T$ concatenates all weight vectors corresponding to the p-th partition. This regularizer will thus tend to kill entire partitions. In other words, it will uncover notions of grouping, or learning-task similarity, that are the most important.

2. We can put the weight vector of each group of each partition in separate blocks. Then the regularizer becomes:

$$\Omega_{gs}(\tilde{\mathbf{w}}) = \left(\sum_{p=1}^{P} \sum_{g=1}^{G} \|\tilde{\mathbf{w}}_{p,g}\|_2 \right)^2 \tag{16}$$

   This regularizer will tend to select a small set of groups which are needed to get a good performance.

4.4 Automatically selecting features

Mixed norms can also be used to select a sparse set of feature blocks that are most useful for classification. This is again useful for interpretability. In our application, features are resource counters at a node. Some of these features are related to memory, others to cpu usage, etc. If our model is able to predict straggler behavior solely on the basis of memory usage counters, for instance, then it might indicate that the cluster or the workload is memory constrained.

In our multi-task setting, such feature selection can also interact with the different groups and partitions of learning-tasks. Suppose each feature vector $\mathbf{x}$ is made up of blocks corresponding to different kinds of features, denoted by $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots \mathbf{x}^{(B)}$. We can similarly divide each weight vector $\tilde{\mathbf{w}}_{p,g}$ into blocks denoted by $\tilde{\mathbf{w}}_{p,g}^{(1)}$, $\tilde{\mathbf{w}}_{p,g}^{(2)}$ and so on. Then we have two alternatives.

1. One can concatenate corresponding feature blocks from all the weight vectors to get $\tilde{\mathbf{w}}^{(1)}, \ldots, \tilde{\mathbf{w}}^{(B)}$. Then a mixed l_1 and l_2 regularizer using these weight blocks can be written as:

$$\Omega_{fs1}(\tilde{\mathbf{w}}) = \left(\sum_{b=1}^{B} \|\tilde{\mathbf{w}}^{(b)}\|_2 \right)^2 \tag{17}$$

   Such a regularizer will encourage the model to select a sparse set of feature blocks on which to base its decision, setting *all* weights corresponding to all the other feature blocks to 0.

2. An alternative would be to let each group vector choose its own sparse set of feature blocks. This can be achieved with the following regularizer:

$$\Omega_{fs2}(\tilde{\mathbf{w}}) = \left(\sum_{p,g} \sum_{b=1}^{B} \|\tilde{\mathbf{w}}_{p,g}^{(b)}\|_2 \right)^2 \tag{18}$$

4.5 Kernelizing the formulation

We have till now described our formulation in primal space. However, kernelizing this formulation is simple. First, note that if we use a simple squared l_2 regularizer, then our formulation is equivalent to a standard SVM in a transformed feature space. This transformed feature space corresponds to a new kernel:

$$\begin{aligned} K_{new}(\mathbf{x}_{it}, \mathbf{x}_{ju}) &= \langle \phi(\mathbf{x}_{it}), \phi(\mathbf{x}_{ju}) \rangle \\ &= \sum_{p,g} \langle \phi_{p,g}(\mathbf{x}_{it}), \phi_{p,g}(\mathbf{x}_{ju}) \rangle \\ &= \sum_{p,g} \delta_{g_p(t),g} \delta_{g_p(u),g} \frac{\lambda}{\lambda_{p,g}} K(\mathbf{x}_{it}, \mathbf{x}_{jt}) \end{aligned} \tag{19}$$

Thus, the transformed kernel corresponds to the linear combination of a set of kernels, one for each group. Each group kernel is zero unless both data points belong to the group, in which case it equals a scaled version of the kernel in the original feature space.

When using the mixed-norm regularizers, the derivation is a bit more involved. We first note that (Aflalo et al., 2011):

$$\left(\sum_{m=1}^{M} \|\mathbf{w}_m\|_2 \right)^2 = \min_{0 \le \eta \le 1, \|\eta\|_1 \le 1} \sum_{m=1}^{M} \frac{\|\mathbf{w}_m\|_2^2}{\eta_m} \tag{20}$$

Using this transformation leads to the following primal objective:

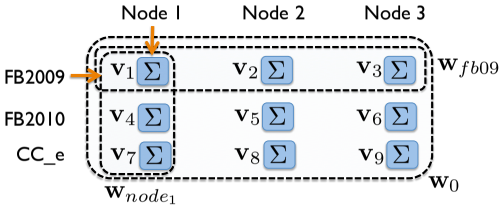
$$\min_{\mathbf{w}, b, 0 \le \eta \le 1, \|\eta\|_1 \le 1} \sum_{m=1}^{M} \frac{\|\mathbf{w}_m\|_2^2}{\eta_m} + \sum_{i,t} L(x_{it}, y_{it}, \mathbf{w}) \tag{21}$$

where $L(x_{it}, y_{it}, \mathbf{w})$ is the hinge loss. This corresponds to a 1-norm multiple kernel learning formulation (Kloft et al., 2011) where each block of feature weights corresponds to a separate kernel. We refer the reader to Kloft et al. (2011) for a description of the dual of this formulation.

We note that these kernelized versions are very similar to the ones derived in (Widmer et al., 2010; Blanchard et al., 2011). However, the group and partition structure and the application domain are unique to ours.

4.6 Application to straggler avoidance

We apply this formulation to straggler avoidance as follows. Suppose there are N nodes and L workloads. Then there are $N \times L$ learning-tasks, and Wrangler trains as many models, one for each {node, workload} tuple. For our proposal, we consider four different partitions:



$$\mathbf{w}_1 = \mathbf{w}_0 + \mathbf{w}_{node_1} + \mathbf{w}_{fb09} + \mathbf{v}_1$$

Figure 2: In our context of straggler avoidance, the learning-tasks naturally cluster into various groups in multiple partitions. When a particular learning- task, for example, node 1 and workload FB2009 ($\mathbf{v}_1$), has limited training data available, we learn its weight vector, $\mathbf{w}_1$, by adding the weight vectors of groups it belongs to from different partitions.

1. A single group consisting of all nodes and workloads. This gives us the single weight vector $\mathbf{w}_0$.

2. One group for each node, consisting of all L learning-tasks belonging to that node. This gives us one weight vector for each node $\mathbf{w}_n, n = 1, \ldots, N$, that captures the heterogeneity of nodes.

3. One group for each workload, consisting of all N learning-tasks belonging to that workload. This gives us one weight vector for each workload $\mathbf{w}_l, l = 1, \ldots, L$, that captures peculiarities of particular workloads.

4. Each {node, workload} tuple as its own group. Since there are $N \times L$ such pairs, we get $N \times L$ weight vectors, which we denote as $\mathbf{v}_t$, following the notation considered in Evgeniou and Pontil (2004).

Thus, if we use all four partitions, the weight vector $\mathbf{w}_t$ for a given workload l_t and a given node n_t is:

$$\mathbf{w}_t = \mathbf{w}_0 + \mathbf{w}_{n_t} + \mathbf{w}_{l_t} + \mathbf{v}_t \tag{22}$$

Figure 2 shows an example. The learning problem for the FB2009[4] workload running on node 1 belongs to one group from each of the four partitions mentioned above: (1) the global partition, denoted by the weight vector, $\mathbf{w}_0$, (2) the group corresponding to node 1 from the node-wise partition, denoted by the weight vector $\mathbf{w}_{node_1}$, (3) the group corresponding to the FB2009 workload from the workload-wise partition, denoted by the weight vector $\mathbf{w}_{fb09}$, and (4) the group containing just this individual learning problem, denoted by the weight vector $\mathbf{v}_1$. Thus, we can learn the weight vector $\mathbf{w}_1$ as:

$$\mathbf{w}_1 = \mathbf{w}_0 + \mathbf{w}_{node_1} + \mathbf{w}_{fb09} + \mathbf{v}_1 \tag{23}$$

The corresponding training problem is then:

4. See Section 5.1 for details about the workloads we use.

$$\min_{\mathbf{w},b} \lambda_0\|\mathbf{w}_0\|^2 + \frac{\nu}{N}\sum_{n=1}^{N}\|\mathbf{w}_n\|^2 + \frac{\omega}{L}\sum_{l=1}^{L}\|\mathbf{w}_l\|^2 + \frac{\tau}{T}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2 + \sum_{t=1}^{T}\sum_{i=1}^{k_t} L_{Hinge}\left(y_{it}, (\mathbf{w}_0 + \mathbf{w}_{n_t} + \mathbf{w}_{l_t} + \mathbf{v}_t)^T\mathbf{x}_{it} + b\right) \quad (24)$$

where λ_0, ν, ω, τ are hyperparameters. We ran an initial grid search on a validation set to fix these hyperparameters and found that prediction accuracy was not very sensitive to these settings: several settings gave very close to optimal results. We used $\lambda_0 = \nu = \omega = \tau = 1$, which was one of the highest performing settings, for all our experiments. Appendix A provides the details of this grid search experiment along with the sensitivity of these hyperparameters to a set of values. As described in Section 4.2, we work in a transformed space in which the above problem reduces to a standard SVM. In this space the weight vector is

$$\tilde{\mathbf{w}} = [\tilde{\mathbf{w}}_0^T, \tilde{\mathbf{w}}_{n_1}^T, \ldots, \tilde{\mathbf{w}}_{n_N}^T, \tilde{\mathbf{w}}_{l_1}^T, \ldots, \tilde{\mathbf{w}}_{l_L}^T, \tilde{\mathbf{v}}_{t_1}^T, \ldots, \tilde{\mathbf{v}}_{t_T}^T]^T \quad (25)$$

This decomposition will change depending on which partitions we use.

As described in the previous section, we can also use mixed l_1 and l_2 norms to automatically select groups or partitions. To select partitions, we combine all the node-related weight vectors $\tilde{\mathbf{w}}_{n_1}, \ldots, \tilde{\mathbf{w}}_{n_N}$ into one long vector $\tilde{\mathbf{w}}_N$, all workload vectors $\tilde{\mathbf{w}}_{l_1}, \ldots, \tilde{\mathbf{w}}_{l_L}$ into $\tilde{\mathbf{w}}_L$ and all $\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_T$ into $\tilde{\mathbf{v}}_T$. We then solve the following optimization:

$$\min_{\tilde{\mathbf{w}},b} \left(\|\tilde{\mathbf{w}}_0\|_2 + \|\tilde{\mathbf{w}}_N\|_2 + \|\tilde{\mathbf{w}}_L\|_2 + \|\tilde{\mathbf{v}}_T\|_2\right)^2 + C\sum_{i,t} L_{Hinge}\left(y_{it}, \tilde{\mathbf{w}}^T\phi(\mathbf{x}_{it}) + b\right) \quad (26)$$

This model will learn which notion of grouping is most important. For instance if $\tilde{\mathbf{w}}_L$ is set to 0, then we might conclude that straggler behaviour doesn't depend that much on the particular workload being executed.

To select individual groups, we solve the optimization problem:

$$\min_{\tilde{\mathbf{w}},b,\xi\geq 0} \left(\|\tilde{\mathbf{w}}_0\|_2 + \sum_{n=1}^{N}\|\tilde{\mathbf{w}}_n\|_2 + \sum_{l=1}^{L}\|\tilde{\mathbf{w}}_l\|_2 + \sum_{t=1}^{T}\|\tilde{\mathbf{v}}_t\|_2\right)^2 + C\sum_{i,t} L_{Hinge}\left(y_{it}, \tilde{\mathbf{w}}^T\phi(\mathbf{x}_{it}) + b\right) \quad (27)$$

This formulation can set some individual node or workload models to 0. This would mean that straggler behavior on some nodes or workloads can be predicted by generic models, but others require more node-specific or workload-specific reasoning.

We can also use mixed norms for feature selection. The features in our feature vector correspond to resource usage counters, and we divide them into 5 categories, as explained

in Section 3.1: counters based on cpu, those based on network, those based on disk, those based on memory, and other system-level counters. Then each $\tilde{\mathbf{w}}_n$, $\tilde{\mathbf{w}}_l$ and $\tilde{\mathbf{v}}_t$ gets similarly split up. As described before, we can either let each model choose its own set of features using this regularizer:

$$\begin{aligned}\Omega(\tilde{\mathbf{w}}) \quad &= (\|\tilde{\mathbf{w}}_0^{mem}\|_2 + \|\tilde{\mathbf{w}}_0^{disk}\|_2 + \|\tilde{\mathbf{w}}_0^{cpu}\|_2 + \|\tilde{\mathbf{w}}_0^{network}\|_2 + \|\tilde{\mathbf{w}}_0^{system}\|_2 + \\ &+ \textstyle\sum_{n=1}^{N}[\|\tilde{\mathbf{w}}_n^{mem}\|_2 + \|\tilde{\mathbf{w}}_n^{disk}\|_2 + \|\tilde{\mathbf{w}}_n^{cpu}\|_2 + \|\tilde{\mathbf{w}}_n^{network}\|_2 + \|\tilde{\mathbf{w}}_n^{system}\|_2] \\ &+ \textstyle\sum_{l=1}^{L}[\|\tilde{\mathbf{w}}_l^{mem}\|_2 + \|\tilde{\mathbf{w}}_l^{disk}\|_2 + \|\tilde{\mathbf{w}}_l^{cpu}\|_2 + \|\tilde{\mathbf{w}}_l^{network}\|_2 + \|\tilde{\mathbf{w}}_l^{system}\|_2] \\ &+ \textstyle\sum_{t=1}^{T}[\|\tilde{\mathbf{v}}_t^{mem}\|_2 + \|\tilde{\mathbf{v}}_t^{disk}\|_2 + \|\tilde{\mathbf{v}}_t^{cpu}\|_2 + \|\tilde{\mathbf{v}}_t^{network}\|_2 + \|\tilde{\mathbf{v}}_t^{system}\|_2])^2 \quad (28)\end{aligned}$$

or choose a single set of features globally using this regularizer:

$$\Omega(\tilde{\mathbf{w}}) = \left(\|\tilde{\mathbf{w}}^{mem}\|_2 + \|\tilde{\mathbf{w}}^{cpu}\|_2 + \|\tilde{\mathbf{w}}^{disk}\|_2 + \|\tilde{\mathbf{w}}^{network}\|_2 + \|\tilde{\mathbf{w}}^{system}\|_2\right)^2 \quad (29)$$

where $\tilde{\mathbf{w}}^{mem}$ concatenates all memory related weights from all models,

$$\tilde{\mathbf{w}}^{mem} = [\tilde{\mathbf{w}}_0^{memT}, \tilde{\mathbf{w}}_{n_1}^{memT}, \ldots, \tilde{\mathbf{w}}_{n_N}^{memT}, \tilde{\mathbf{w}}_{l_1}^{memT}, \ldots, \tilde{\mathbf{w}}_{l_L}^{memT}]^T \quad (30)$$

$\tilde{\mathbf{w}}^{cpu}$, $\tilde{\mathbf{w}}^{disk}$ etc. are defined similarly.

4.7 Exploring the relationships between the weight vectors

Before getting into the experiments, we can get some insights on what our formulation will learn by looking at the KKT conditions. Equation 24 can equivalently be written as:

$$\begin{aligned}\min_{\mathbf{w},b,\xi} \lambda_0\|\mathbf{w}_0\|^2 + \frac{\nu}{N}\sum_{n=1}^{N}\|\mathbf{w}_n\|^2 + \frac{\omega}{L}\sum_{l=1}^{L}\|\mathbf{w}_l\|^2 \\ + \frac{\tau}{T}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2 + \sum_{t=1}^{T}\sum_{i=1}^{k_t}\xi_{it} \quad (31) \\ \text{s.t. } y_{it}\left((\mathbf{w}_0 + \mathbf{w}_{n_t} + \mathbf{w}_{l_t} + \mathbf{v}_t)^T\mathbf{x}_{it} + b\right) \geq 1 - \xi_{it} \ \ \forall i,t \\ \xi_{it} \geq 0 \ \ \forall i,t\end{aligned}$$

The Lagrangian of the formulation in Equation 31 is:

$$\begin{aligned}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\gamma}) = \lambda_0\|\mathbf{w}_0\|^2 + \frac{\nu}{N}\sum_{n=1}^{N}\|\mathbf{w}_n\|^2 + \frac{\omega}{L}\sum_{l=1}^{L}\|\mathbf{w}_l\|^2 \\ + \frac{\tau}{T}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2 + \sum_{t=1}^{T}\sum_{i=1}^{k_t}\xi_{it} - \sum_{t=1}^{T}\sum_{i=1}^{k_t}\gamma_{it}\xi_{it} \quad (32) \\ + \sum_{t=1}^{T}\sum_{i=1}^{k_t}\alpha_{it}\left(1 - \xi_{it} - y_{it}\left(\mathbf{w}_t^T\mathbf{x}_{it} + b\right)\right)\end{aligned}$$

Taking derivatives w.r.t the primal variables and setting to 0 gives us relationships between $\mathbf{w}_0, \mathbf{v}_t, \mathbf{w}_n$ and $\mathbf{w}_l$:

$$\lambda_0 \mathbf{w}_0^* = \frac{\tau}{T} \sum_t \mathbf{v}_t^* \tag{33}$$

$$\nu \mathbf{w}_n^* = \frac{\tau}{T/N} \sum_{t:n_t=n} \mathbf{v}_t^* \tag{34}$$

$$\omega \mathbf{w}_l^* = \frac{\tau}{T/L} \sum_{t:l_t=l} \mathbf{v}_t^* \tag{35}$$

$$\lambda_0 \mathbf{w}_0^* = \frac{\nu}{N} \sum_n \mathbf{w}_n^* \tag{36}$$

$$\lambda_0 \mathbf{w}_0^* = \frac{\omega}{L} \sum_l \mathbf{w}_l^* \tag{37}$$

Evgeniou and Pontil (2004) also obtain Equation 33 in their formulation, but the other relationships are specific to ours. These relationships imply that these variables shouldn't be considered independent. $\mathbf{w}_n$, $\mathbf{w}_l$ and $\mathbf{w}_0$ are scaled means of the $\mathbf{v}_t$'s of the group they capture.

4.8 Generalizing to unseen nodes and workloads

Consider what happens when we remove the partition corresponding to individual {node, workload} tuples, i.e., $\mathbf{v}_t$, from our formulations. We now do not have any parameters specific to a node-workload combination, but can still capture both node- and workload-dependent properties of the learning problem. Such formulations are thus similar to factorized models where the node and workload dependent factors are grouped into separate blocks. We end up with only $(N + L)d$ parameters, whereas a formulation like that of (Evgeniou and Pontil, 2004; Evgeniou et al., 2005; Jacob et al., 2009) will still have NLd parameters (here d is the input dimensionality). Thus, we can *reduce* the number of parameters while still capturing the essential properties of the learning problems.

In addition, since we no longer have a separate weight vector for each {node, workload} tuple, we can generalize to node-workload pairs that are completely unseen at train time: the classifier for such an unseen combination t will simply be $\mathbf{w}_0 + \mathbf{w}_{n_t} + \mathbf{w}_{l_t}$. We thus explicitly use knowledge gleaned from prior workloads run on this node (through $\mathbf{w}_{n_t}$) and other nodes running this workload (through $\mathbf{w}_{l_t}$). This is especially useful in our application where there may be a large number of nodes and workloads. In such cases, collecting data for each node-workload pair will be time consuming, and generalizing to unseen combinations will be a significant advantage.

In most of our experiments, therefore, we remove the partition corresponding to individual {node, workload} tuples. We explicitly evaluate how well we generalize by doing so in Section 5.4.

5. Empirical Evaluation

In this section, we describe our dataset, provide variants to our proposed formulation and then evaluate them using the following metrics: first, classification accuracy when there is

Trace	#Machines	Length	Date	#Jobs
FB2009	600	6 month	2009	1129193
FB2010	3000	1.5 months	2010	1169184
CC_b	300	9 days	2011	22974
CC_e	100	9 days	2011	10790
Total	≈ 4000	≈ 8.5 months	-	2332141

Table 2: Dataset. *FB*: Facebook, *CC*: Cloudera Customer.

sufficient data and also when sufficient data is not available, and second, improvement in overall job completion times, and third, reduction in resources consumed.

5.1 Datasets

The set of real-world workloads considered in this paper are collected from the production compute clusters at Facebook and Cloudera's customers, which we denote as $FB2009$, $FB2010$, CC_b and CC_e. Table 2 provides details about these workloads in terms of the number of machines in the actual clusters, the length and date of data capture, total number of jobs in those workloads. Chen, et al., explain the data in further details in (Chen et al., 2012). Together, the dataset consists of traces from over about 4000 machines captured over almost eight months. For faithfully replaying these real-world production traces on our 20 node EC2 cluster, we used a statistical workload replay tool, SWIM (Chen et al., 2011) that synthesizes a workload with representative job submission rates and patterns, shuffle/input data size and output/shuffle data ratios (see Chen et al. (2011) for details of replay methodology). SWIM scales the workload to the number of nodes in the experimental cluster.

For each workload, we need data with ground-truth labels for training and validating our models. We collect this data by running tasks from the workload as described above and recording the resource usage counters x_i at a node at the time a task i is launched, and the ground truth label y_i by checking if it ends up becoming a straggler. Then we divide this dataset temporally into a training set and a validation set. In other words, the first few tasks that were executed form the train set and the rest of the tasks form the validation set. In the experiments below, we vary the percentage of data that is used for training, and compute the prediction accuracy on the validation set. We train our final model using two-thirds of this dataset and proceed to evaluate it on our ultimate metric, i.e., job completion times.

To measure job completion times, we then incorporate the trained models into the job scheduler (as in Wrangler). We then run the replay for the workload again, but with a fresh set of tasks. These fresh set of tasks form our test set, and this test set is only used to measure job completion times. Table 3 shows the sizes of the datasets.

Each data point is represented by a 107 dimensional feature vector comprising the node's resource usage counters at the time of launching a task on it. We optimize all our formulations using Liblinear (Fan et al., 2008) for the l_2 regularized variants and using the algorithm proposed by Aflalo et al. (2011) (modified to work in the primal) for the mixed norm variants.

Workload	No of tasks (Training+Validation)	No of tasks Test
$FB2009$	4885	13632
$FB2010$	3843	38158
CC_b	5991	30203
CC_e	39014	94550

Table 3: Number of tasks we use for each workload in the train+val and test sets.

Below, we describe (1) how we use different MTL formulations and prediction accuracy achieved by these formulations, (2) how we learn a classifier for previously unseen node and/or workload and the prediction accuracy it achieves, (3) the improvement in overall job completion times achieved by our formulation and Wrangler over speculative execution, and (4) reduction in resources consumed using our formulation compared to Wrangler.

5.2 Variants of proposed formulation

We consider several variants of the general formulation described in Section 4. Using a simple squared l_2 regularizer, we first consider $\mathbf{w}_0$, $\mathbf{w}_n$ and $\mathbf{w}_l$, individually:

- f_0: In this formulation, we consider only the global partition in which all learning problems belong to a single group. This corresponds to removing $\mathbf{v}_t$, $\mathbf{w}_n$ and $\mathbf{w}_l$. This formulation thus learns a single global weight vector, $\mathbf{w}_0$, for all the nodes and all the workloads.

- f_n: Here we consider only the partition based on nodes. This corresponds to only learning a $\mathbf{w}_n$, that is, one model for each node. This model learns to predict stragglers based on a node's resource usage counters across different workloads, but it cannot capture any properties that are specific to a particular workload.

- f_l: Here we consider only the partition based on workloads. This means we only learn $\mathbf{w}_l$, i.e., a workload dependent model across nodes executing a particular workload. This model learns to predict stragglers based on the resource usage pattern caused due to a workload across nodes, but ignores the characteristics of a specific node.

The above three formulations either discard the node information, the workload information, or both. We now consider multi-task variants that capture both, node and workload properties:

- $f_{0,t}$: This is the formulation proposed by Evgeniou and Pontil (2004), and corresponds to using the global partition where all learning-tasks belong to one group, and the partition where each learning-task is its own group. This learns $\mathbf{w}_0$ and $\mathbf{v}_t$. Note that this formulation still has to learn on the order of NLd different parameters, and has to collect enough data to learn a separate weight vector for each {node, workload} combination.

- $f_{0,t,l}$: This formulation extends the formulation in $f_{0,t}$ by additionally adding the partition based on workloads. It learns $\mathbf{w}_0, \mathbf{w}_l$ and $\mathbf{v}_t$.

- $f_{0,n,l}$: We remove the partition corresponding to individual {node, workload} tuples, removing $\mathbf{v}_t$ entirely and only learning $\mathbf{w}_0$, $\mathbf{w}_l$ and $\mathbf{w}_n$. As described in Section 4.8, this formulation reduces the total number of parameters to $(N + L)d$ and can also generalize to unseen {node, workload} tuples.

For all these formulations, the hyperparameters λ_0, ν, ω and τ were set to 1 wherever applicable. We found this setting to be close to optimal in our initial cross-validation experiments (see Appendix A).

In addition to these, we also consider sparsity-inducing formulations for automatically selecting partitions or groups or blocks of features, as explained in Sections 4.3, 4.4, and 4.6.

- f_{ps} : We use the global, node-based and workload-based formulations thus removing $\mathbf{v}_t$ entirely as in $f_{0,n,l}$ and use mixed l_1 and l_2 norms to automatically select the useful partitions from among these. This model will learn the notion of grouping that is most important. To select partitions, we combine all the node-related weight vectors $\tilde{\mathbf{w}}_{n_1}, \ldots, \tilde{\mathbf{w}}_{n_N}$ into one long vector $\tilde{\mathbf{w}}_N$ and all workload vectors into $\tilde{\mathbf{w}}_L$ as shown below:

$$\tilde{\mathbf{w}} = [\tilde{\mathbf{w}}_0^T, \underbrace{\tilde{\mathbf{w}}_{n_1}^T, \ldots, \tilde{\mathbf{w}}_{n_N}^T}_{\tilde{\mathbf{w}}_N^T}, \underbrace{\tilde{\mathbf{w}}_{l_1}^T, \ldots, \tilde{\mathbf{w}}_{l_L}^T}_{\tilde{\mathbf{w}}_L^T}]^T \tag{38}$$

  Thus our weight vector $\tilde{\mathbf{w}}$ is split up into blocks corresponding to node, workload and global models.

- f_{gs} : This is the formulation where we use mixed l_1 and l_2 norms to automatically select groups within a partition. Again, we only consider the global, node-based and workload-based formulations. This formulation can set individual node or workload models to zero, unlike f_{ps}, that can set a complete partition, i.e. in our case, a combined model of all the nodes or all the workloads to zero. This would mean that predicting straggler behavior on some nodes or workloads does not need reasoning that is specific to those nodes or workloads; instead a generic model would work.

Finally, we also try the following two mixed norms formulations for feature selection. As before, both these formulations remove the partition corresponding to individual {node, workload} tuples, i.e., remove $\mathbf{v}_t$.

- f_{fs1} : As explained in Equation 28, we divide our features into five categories corresponding to cpu, memory, disk, network and other system-level counters. Then each weight vector gets similarly split up into these categories. This formulation learns which categories of features are more important for some nodes or workloads than others.

- f_{fs2} : This formulation, as given in Equation 29, selects a category of features across all the weight vectors of nodes and workloads. This formulation can learn if stragglers in a cluster are caused due to contention for a specific resource.

5.3 Prediction accuracy

In this section, we evaluate the formulations described in the previous section for their straggler prediction accuracy. In the following subsection 5.3.1, we evaluate formulations that use l_2 regularizers, viz., f_0, f_n, f_l, $f_{0,n,l}$, $f_{0,t}$, and $f_{0,t,l}$. Then, in Section 5.3.2, we evaluate the mixed norm formulations (a) that automatically selects partitions, f_{ps}, (b) that automatically selects groups, f_{gs}, and then the formulations (c) f_{fs1}, and (d) f_{fs2} that can automatically select features. We list our formulations with a brief description in Table 4.

5.3.1 Formulations with l_2 regularizers

We aim at learning to predict stragglers using as small amount of data as feasible, as this means shorter data capture time. Note that stragglers are fewer than non-stragglers, so we oversample from the stragglers' class to represent the two classes equally in both, the training and validation sets[5]. Table 5 shows the percentage accuracy of predicting stragglers with varying amount of training data. We observe that:

- With very small amounts of data, all MTL variants outperform Wrangler. In fact, all of f_0 to $f_{0,t,l}$ need only one sixth of the training data to achieve the same or better accuracy.
- It is important to capture both node- and workload-dependent aspects of the problem: $f_{0,n,l}$, $f_{0,t}$ and $f_{0,t,l}$ consistently outperform f_0, f_n and f_l.
- $f_{0,t}$ and $f_{0,t,l}$ perform up to 7 percentage points better than Wrangler with the same amount of training data, with $f_{0,n,l}$ not far behind.

For a better visualization, Figure 3 shows the comparison of prediction accuracy of these formulations, in terms of percentage true positives and percentage false positives when 50% of total data is available.

Next, we evaluate and discuss the sparsity-inducing formulations, f_{ps} (Equation 26), f_{gs} (Equation 27), f_{fs1} (Equation 28), and f_{fs2} (Equation 29).

5.3.2 Formulations with mixed l_1 and l_2 norms

Automatically selecting partitions or groups: In this Section, we evaluate f_{ps} and f_{gs}. Table 6 shows the prediction accuracy of these formulations compared to f_0, f_n, f_l and $f_{0,n,l}$. f_{ps} and f_{gs} show comparable prediction accuracy to $f_{0,n,l}$. We also found interesting sparsity patterns in the learnt weight vectors.

- f_{ps} : Recall that f_{ps} attempts to set the weights of entire partitions to 0. In our case we have a global partition, a node-based partition and a workload-based partition. We observed that only $\tilde{\mathbf{w}}_0$ is zero in the resulting weight vector learned. This means that given node-specific and workload-specific factors, the global factors that are common across all the nodes and workloads do not contribute to the prediction. In other words, similar accuracy could be achieved without using $\tilde{\mathbf{w}}_0$. However, both node- and workload-dependent weights are necessary.

5. An alternative to statistical oversampling would be to use class-sensitive miss-classification penalties.

Formulation	Description
f_0	uses a single, global weight vector
f_n	uses only node-specific weights
f_l	uses only workload-specific weights
$f_{0,n,l}$	uses global, node- and workload-specific weights
$f_{0,t}$	uses global weights and weights specific to {node, workload} tuples
$f_{0,t,l}$	uses global and workload-specific weights and weights specific to {node, workload} tuples
f_{ps}	selects partitions automatically
f_{gs}	selects groups automatically
f_{fs1}	selects feature-blocks automatically for individual groups
f_{fs2}	selects feature-blocks automatically across all the groups

Table 4: Brief description of all our formulations.

%Training Data	Wrangler Yadwadkar et al. (2014)	f_0	f_n	f_l	$f_{0,n,l}$	$f_{0,t}$	$f_{0,t,l}$
1	Insufficient data	66.9	63.5	66.5	65.5	63.7	66.2
2	Insufficient Data	67.1	63.3	67.7	67.5	64.3	67.7
5	Insufficient Data	67.5	68.1	69.1	69.8	69.6	69.1
10	63.9	67.8	70.9	69.4	72.3	73.1	72.9
20	67.2	68.0	72.6	70.1	72.9	74.7	74.8
30	68.5	68.5	73.2	70.3	74.1	75.9	75.8
40	69.7	68.2	73.9	70.5	74.3	76.4	76.4
50	70.1	68.5	74.1	70.4	75.3	77.1	77.2

Table 5: Prediction accuracies (in %) of various MTL formulations for straggler prediction with varying amount of training data. See Section 5.3.1 for details.

- f_{gs} : In f_{gs}, we encourage individual nodes-specific or individual workloads-specific weight vectors separately to be set to zero. We observed that some of the nodes' weight vectors and some of the workload-specific weight vectors were zero, indicating that in some cases we do not need node or workload specific reasoning. (One can use the learnt sparsity pattern and attempt to correlate it with some node and workload characteristics; however we have not explored this in this paper.)

We also note that our mixed-norm formulations automatically learn a grouping that achieves comparable accuracy with lesser total number of groups, and thus fewer parameters.

Automatically selecting features: In this section, we evaluate the remaining formulations f_{fs1} and f_{fs2}. These two formulations group sets of features based on resources. We divide the features in five different categories viz., features measuring (1) CPU utilization, (2) memory utilization, (3) network usage, (5) disk utilization, (6) other system level performance counters. We evaluate their straggler prediction accuracy and then discuss their interpretability in terms of understanding the causes behind stragglers.

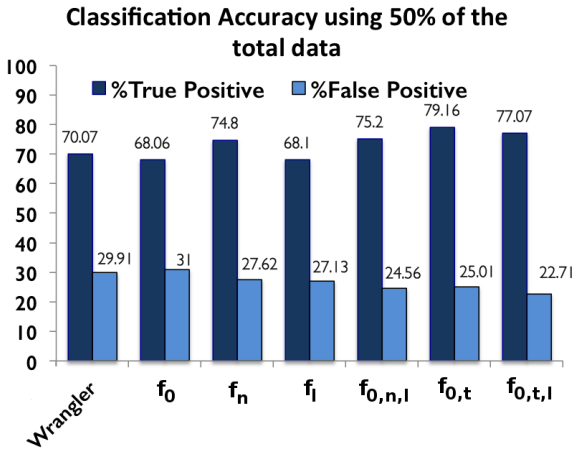


Figure 3: Classification accuracy of various MTL formulations as compared to Wrangler using 50% of the total data. This plot shows the percentage of true positives and the percentage of false positives in each of the cases. These quantities are computed as: % True Positive = (fraction of stragglers predicted correctly as stragglers) $\times$ 100, and % False Positive = (fraction of non-stragglers predicted incorrectly to be stragglers) $\times$ 100.

Formulation	Straggler Prediction Accuracy
f_0	68.5
f_n	74.1
f_l	70.4
$f_{0,n,l}$	75.3
f_{ps}	74.4
f_{gs}	73.8

Table 6: Straggler prediction accuracies (in %) using the four mixed-norm formulations f_{ps} and f_{gs} compared with formulations that use l_2 regularizers. Note that f_{ps} and f_{gs} perform with comparable accuracy with $f_{0,n,l}$, however, use lesser number of groups and parameters, resulting in simpler models.

Table 7 shows the percentage prediction accuracies of these formulations on our test set. Note that these formulations show comparable prediction accuracy. Next, we discuss the impact of f_{fs1} and f_{fs2} on understanding the straggler behavior.

- f_{fs1}: Because this formulation divides each group weight vector further into blocks based on the kind of features, it can potentially provide fine-grained insight into what kinds of features are most important for each group weight vector. Indeed, we found that some node models assign zero weight to features from the network category, while others assign a zero weight to the disk category. However, no global patterns emerge.

Formulation	Straggler Prediction Accuracy (in %)
f_{fs1}	74.9
f_{fs2}	74.9

Table 7: Straggler prediction accuracies (in %) using f_{fs1} and f_{fs2} that encourage sparsity across bocks of features.

FB2009		FB2010		CC_b		CC_e	
$f_{0,n,l}$	$f_{0,t}$	$f_{0,n,l}$	$f_{0,t}$	$f_{0,n,l}$	$f_{0,t}$	$f_{0,n,l}$	$f_{0,t}$
73.1	45.3	46.7	48.3	50.2	49.4	52.8	68.2
56.2	57.5	57.3	58.7	61.0	53.5	64.4	48.9
63.9	55.5	50.0	48.8	59.4	53.4	48.9	65.1
63.2	47.7	60.6	57.4	55.7	49.5	47.3	73.9
50.7	42.4	51.4	56.2	50.8	44.6	71.2	59.9

Table 8: Straggler Prediction accuracies (in %) of $f_{0,n,l}$ and $f_{0,t}$ on test data from an unseen node-workload pair. See Section 5.4 for details.

This reinforces our belief that the causes of stragglers vary quite a bit from node to node or workload to workload.

- f_{fs2}: This formulation considers these feature categories across the various nodes and workloads and provides a way of knowing if there are certain dominating factors causing stragglers in a cluster. However, we observed that none of the feature categories had zero weights in the weight vector learned for our dataset. Again, this means that there is no single, easily discoverable reason for straggler behavior, and provides evidence to the claim made by Ananthanarayanan et al. (2013, 2014) that the causes behind stragglers are hard to figure out.

5.4 Prediction accuracy for a {node, workload} tuple with insufficient data

One of our goals in this work is to reduce the amount of training data required to get a straggler prediction model up and running. When a new workload begins to execute on a node, we want to learn a model as quickly as possible. Recall that (Section 5.3) with enough training data available we found that $f_{0,n,l}$, $f_{0,t}$ and $f_{0,t,l}$ seem to perform similarly, with $f_{0,n,l}$ performing slightly worse. However, as mentioned in Section 4.8, formulation $f_{0,n,l}$ has fewer parameters and, because it has no weight vector specific to a particular {node, workload} tuple, can generalize to new {node, workload} tuples unseen at train time. This is in contrast to $f_{0,t}$ which has to fall back on $\mathbf{w}_0$ in such a situation, and thus may not generalize as well. In this section, we see if this is indeed true.

We trained classifiers based on $f_{0,n,l}$ and $f_{0,t}$ leaving out 95% of the data of one node-workload pair every time. We then test the models on the left-out data. Table 8 shows the percentage classification accuracy from 20 such runs. We note the following:

- For 13 out of 20 classification experiments, $f_{0,n,l}$ performs better than $f_{0,t}$. For 10 out of these 13 cases, the difference in performance is more than 5 percentage points.
- For workloads $FB2009$ and CC_b, we see $f_{0,n,l}$ performs better consistently.

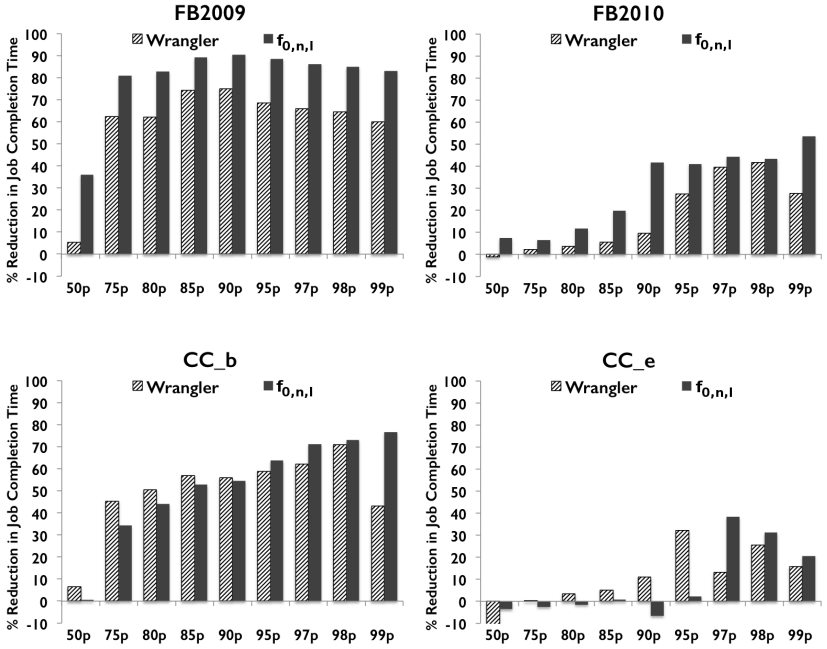


Figure 4: Improvement in the overall job completion times achieved by $f_{0,n,l}$ and Wrangler over speculative execution.

- $f_{0,n,l}$ sometimes performs worse, but in only 3 of these cases is it significantly worse (worse by more than 5 percentage points). All 3 of these instances are in case of the CC_e workload. In general, for this workload, we also notice a huge variance in the numbers obtained across multiple nodes. See Yadwadkar et al. (2014), for a discussion of some of the issues in this workload.

This shows that $f_{0,n,l}$ works better in real-world settings where one cannot expect enough data for all node-workload pairs. Therefore, we evaluate $f_{0,n,l}$ in our next experiment (Section 5.5) to see if it improves job completion times.

5.5 Improvement in overall job completion time

We now evaluate our formulation, $f_{0,n,l}$, using the second metric, improvement in the overall job completion times over speculative execution. We compare these improvements to that achieved by Wrangler (Figure 4). Improvement at the 99^{th} percentile is a strong indicator of the effectiveness of straggler mitigation techniques. We see that $f_{0,n,l}$ significantly

Workload	% Reduction in total task-seconds (MTL with $f_{0,n,l}$)	(Wrangler)
FB-2009	73.33	55.09
FB-2010	8.9	24.77
CC_b	64.12	40.15
CC_e	13.04	8.24

Table 9: Resource utilization with $f_{0,n,l}$ and with *Wrangler* over speculative execution, in terms of total task execution times (in seconds) across all the jobs. $f_{0,n,l}$ reduces resources consumed over Wrangler for $FB2009$, CC_b and CC_e.

improves over Wrangler, reflecting the improvements in prediction accuracy. At the 99^{th} percentile, we improve Wrangler's job completion times by $57.8\%, 35.8\%, 58.9\%$ and 5.7% for $FB2009, FB2010, CC\_b$ and CC_e respectively. Note that Wrangler is already a strong baseline. Hence, the improvement in job completion times on top of the improvement achieved by Wrangler is significant.

5.6 Reduction in resources consumed

When a job is launched on a cluster, it will be broken into small tasks and these tasks will be run in a distributed fashion. Thus, to calculate the resources used, we can sum the resources used by all the tasks. As in Yadwadkar et al. (2014), we use the time taken by each task as a measure of the resources used by the task. Note that, because these tasks will likely be executing in parallel, the total time taken by the tasks will be much larger than the time taken for the whole job to finish, which is what job completion time measures (shown in Figure 4). Ideally, straggler prediction will prevent tasks from becoming stragglers. Fewer stragglers means fewer tasks that need to be replicated by straggler mitigation mechanisms (like speculative execution) and thus lower resource consumption. Thus, improved straggler prediction should also reduce the total task-seconds i.e., resources consumed.

Table 9 compares the percentage reduction in resources consumed in terms of total task-seconds achieved by $f_{0,n,l}$ and Wrangler over speculative execution. We see that the improved predictions of $f_{0,n,l}$ reduce resource consumption significantly more than Wrangler for 3 out of 4 workloads, thus supporting our intuitions. In particular, for $FB2009$ and CC_b, $f_{0,n,l}$ reduces Wrangler's resource consumption by about 40%, while for CC_e the reduction is about 5%.

6. Related Work on Multi-task Learning

The idea that multiple learning problems might be related and can gain from each other dates back to Thrun (1996) and Caruana (1993). They pointed out that humans do not learn a new task from scratch but instead reuse knowledge gleaned from other learning tasks. This notion was formalized by, among others, Baxter (2000) and Ando and Zhang (2005), who quantified this gain. Much of this early work relied on neural networks as a means of learning these shared representations. However, contemporary work has also focused on SVMs and kernel machines.

Our work is an extension of the work of Evgeniou and Pontil (2004), who proposed an additive model for MTL that decomposes classifiers into a shared component and a task-

specific component. In later work, Evgeniou et al. (2005) propose an MTL framework that uses a general quadratic form as a regularizer. They show that if the tasks can be grouped into clusters, they can use a regularizer that encourages all the weight vectors of the group to be closer to each other. Jacob et al. (2009) extend this formulation when the group structure is not known a priori. Xue et al. (2007) infer the group structure using a Bayesian approach. The approach of Widmer et al. (2010) is similar to ours and groups tasks into "meta-tasks", and tries to automatically figure out the meta-tasks required to get good performance. The formulation we propose is also designed to handle group structure, but allows us to dispense with task-specific classifiers entirely, reducing the number of parameters drastically. This allows us to handle tasks that have very little training data by transferring parameters learnt on other tasks. Our formulation shares this property with that of Blanchard et al. (2011), and indeed our basic formulation can be written down in the kernel-based framework they describe for learning to generalize onto a completely unseen task. Other ways of controlling parameters include learning a distance metric (Parameswaran and Weinberger, 2010), and using low rank regularizers (Pong et al., 2010).

Our setting is an example of multilinear multitask learning where each learning problem is indexed by two indices: the node and the workload. Previous work on this subfield of multitask learning has typically used low-rank regularizers on the weight matrix represented as tensors (Romera-Paredes et al., 2013; Wimalawarne et al., 2014). It is also possible to define a similarity between tasks based on how many indices they share (Signoretto et al., 2014). Our formulation captures some of the same intuitions, but has the added advantage of simplicity and ease of implementation.

Using mixed norms for inducing sparsity has a rich history. Donoho (2006) showed that minimizing the l_1 norm recovers sparse solutions when solving linear systems. When used as a regularizer, the l_1 norm learns sparse models, where most weights are 0. The most well known of such sparse formulations is the lasso (Tibshirani, 1996), which uses the l_1 norm to select features in regression. Yuan and Lin (2006) extend the lasso to group lasso, where they use a mixed l_1 and l_2 norm to select a sparse set of *groups* of features. Bach (2008) study the theoretical properties of group lasso. Since these initial papers, mixed norms have found use in a variety of applications. For instance , Quattoni et al. (2008) use a mixed l_∞ and l_1 norm for feature selection. Such mixed norms also show up in the literature on kernel learning, where they are used to select a sparse set of kernels (Varma and Ray, 2007; Kloft et al., 2011) or a sparse set of groups of kernels (Aflalo et al., 2011). Bach et al. (2011) provides an accessible review of mixed norms and their optimization, and we direct the interested reader to that article for more details.

7. Conclusion

Through this work, we have shown the utility of multitask learning in solving the real-world problem of avoiding stragglers in distributed data processing. We have presented a novel MTL formulation that captures the structure of our learning-tasks and reduces job completion times by up to 59% over prior work (Yadwadkar et al., 2014). This reduction comes from a 7 percentage point increase in prediction accuracy. Our formulation can achieve better accuracy with only a sixth of the training data and can generalize better than other MTL approaches for learning-tasks with little or no data. We have also presented extensions

to our formulation using group sparsity inducing mixed norms that automatically discover the structure of our learning tasks and make the final model more interpretable. Finally, we note that, although we use straggler avoidance as the motivation, our formulation is more generally applicable, especially for other prediction problems in distributed computing frameworks, such as resource allocation (Gupta et al., 2013; Delimitrou and Kozyrakis, 2014).

Appendix A. Cross-validating hyperparameter settings

Our formulation reduces the number of hyperparameters to just one per partitioning, which makes it much easier to cross-validate to set their values. In particular, our formulation in the context of straggler avoidance, Equation (24), has four hyperparameters: λ_0, ν, ω, τ. To tune these parameters we used a simple grid search with cross-validation (results are shown in Tables 10, 11, 12, and 13). In general we found that the model formulation is relatively robust to the choice of hyperparameters so long as they are within the correct order of magnitude.

References

Jonathan Aflalo, Aharon Ben-Tal, Chiranjib Bhattacharyya, Jagarlapudi Saketha Nath, and Sankaran Raman. Variable sparsity kernel learning. *Journal of Machine Learning Research*, 12, 2011.

Ganesh Ananthanarayanan, Srikanth Kandula, Albert Greenberg, Ion Stoica, Yi Lu, Bikas Saha, and Edward Harris. Reining in the outliers in map-reduce clusters using mantri. In *OSDI*, 2010.

Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, and Ion Stoica. Effective straggler mitigation: Attack of the clones. In *NSDI*, 2013.

Ganesh Ananthanarayanan, Michael Chien-Chun Hung, Xiaoqi Ren, Ion Stoica, Adam Wierman, and Minlan Yu. Grass: Trimming stragglers in approximation analytics. In *NSDI*, 2014.

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6, 2005.

Francis Bach, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, et al. Convex optimization with sparsity-inducing norms. *Optimization for Machine Learning*, pages 19–53, 2011.

Francis R Bach. Consistency of the group lasso and multiple kernel learning. *The Journal of Machine Learning Research*, 9:1179–1225, 2008.

Jonathan Baxter. A model of inductive bias learning. *JAIR*, 12, 2000.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *NIPS*, 2011.

λ_0	ν	ω	τ	Accuracy (%)
1	1	1	1	75.38
1	1	1	10	75.57
1	1	1	100	75.24
1	1	1	1000	74.39
1	1	10	1	74.93
1	1	10	10	75.51
1	1	10	100	75.05
1	1	10	1000	74.38
1	1	100	1	74.84
1	1	100	10	74.84
1	1	100	100	73.95
1	1	100	1000	73.04
1	1	1000	1	73.03
1	1	1000	10	72.60
1	1	1000	100	72.11
1	1	1000	1000	71.17
1	10	1	1	74.95
1	10	1	10	75.17
1	10	1	100	74.34
1	10	1	1000	74.05
1	10	10	1	75.42
1	10	10	10	75.05
1	10	10	100	74.69
1	10	10	1000	75.00
1	10	100	1	74.84
1	10	100	10	74.77
1	10	100	100	74.42
1	10	100	1000	73.53
1	10	1000	1	72.88
1	10	1000	10	72.23
1	10	1000	100	72.07
1	10	1000	1000	71.16

λ_0	ν	ω	τ	Accuracy (%)
1	100	1	1	75.39
1	100	1	10	74.95
1	100	1	100	74.42
1	100	1	1000	74.00
1	100	10	1	75.58
1	100	10	10	75.14
1	100	10	100	74.92
1	100	10	1000	73.90
1	100	100	1	74.87
1	100	100	10	74.72
1	100	100	100	74.09
1	100	100	1000	72.95
1	100	1000	1	73.08
1	100	1000	10	72.15
1	100	1000	100	72.47
1	100	1000	1000	70.72
1	1000	1	1	75.47
1	1000	1	10	75.37
1	1000	1	100	74.73
1	1000	1	1000	74.49
1	1000	10	1	75.78
1	1000	10	10	74.73
1	1000	10	100	74.87
1	1000	10	1000	74.40
1	1000	100	1	74.97
1	1000	100	10	75.06
1	1000	100	100	73.68
1	1000	100	1000	72.51
1	1000	1000	1	72.29
1	1000	1000	10	72.02
1	1000	1000	100	71.94
1	1000	1000	1000	70.34

Table 10: Tuning the hyperparameters λ_0, ν, ω and τ using grid search.

λ_0	ν	ω	τ	Accuracy (%)
10	1	1	1	65.93
10	1	1	10	67.67
10	1	1	100	63.99
10	1	1	1000	64.67
10	1	10	1	69.97
10	1	10	10	74.92
10	1	10	100	75.13
10	1	10	1000	74.52
10	1	100	1	61.19
10	1	100	10	75.85
10	1	100	100	75.44
10	1	100	1000	75.30
10	1	1000	1	68.35
10	1	1000	10	74.20
10	1	1000	100	74.45
10	1	1000	1000	73.90
10	10	1	1	64.50
10	10	1	10	66.59
10	10	1	100	64.90
10	10	1	1000	61.98
10	10	10	1	69.95
10	10	10	10	75.61
10	10	10	100	75.02
10	10	10	1000	74.63
10	10	100	1	73.26
10	10	100	10	75.19
10	10	100	100	74.77
10	10	100	1000	74.68
10	10	1000	1	58.03
10	10	1000	10	74.83
10	10	1000	100	74.95
10	10	1000	1000	73.83

λ_0	ν	ω	τ	Accuracy (%)
10	100	1	1	66.44
10	100	1	10	64.54
10	100	1	100	64.98
10	100	1	1000	62.97
10	100	10	1	68.87
10	100	10	10	75.84
10	100	10	100	75.09
10	100	10	1000	75.41
10	100	100	1	69.45
10	100	100	10	75.41
10	100	100	100	75.74
10	100	100	1000	75.30
10	100	1000	1	72.66
10	100	1000	10	75.19
10	100	1000	100	74.30
10	100	1000	1000	74.03
10	1000	1	1	67.33
10	1000	1	10	63.73
10	1000	1	100	61.84
10	1000	1	1000	60.44
10	1000	10	1	67.40
10	1000	10	10	75.60
10	1000	10	100	75.26
10	1000	10	1000	75.30
10	1000	100	1	72.45
10	1000	100	10	75.44
10	1000	100	100	75.30
10	1000	100	1000	75.03
10	1000	1000	1	66.71
10	1000	1000	10	75.00
10	1000	1000	100	75.07
10	1000	1000	1000	74.42

Table 11: Tuning the hyperparameters λ_0, ν, ω and τ using grid search.

λ_0	ν	ω	τ	Accuracy (%)
100	1	1	1	67.36
100	1	1	10	66.39
100	1	1	100	63.79
100	1	1	1000	65.24
100	1	10	1	65.46
100	1	10	10	67.84
100	1	10	100	64.63
100	1	10	1000	60.97
100	1	100	1	72.27
100	1	100	10	56.05
100	1	100	100	66.44
100	1	100	1000	70.18
100	1	1000	1	52.14
100	1	1000	10	71.03
100	1	1000	100	65.30
100	1	1000	1000	73.16
100	10	1	1	65.87
100	10	1	10	65.94
100	10	1	100	67.53
100	10	1	1000	64.05
100	10	10	1	69.18
100	10	10	10	62.42
100	10	10	100	62.87
100	10	10	1000	63.63
100	10	100	1	56.48
100	10	100	10	66.43
100	10	100	100	75.18
100	10	100	1000	75.57
100	10	1000	1	53.74
100	10	1000	10	69.14
100	10	1000	100	75.5
100	10	1000	1000	75.24

λ_0	ν	ω	τ	Accuracy (%)
100	100	1	1	64.43
100	100	1	10	65.58
100	100	1	100	65.06
100	100	1	1000	63.42
100	100	10	1	57.59
100	100	10	10	65.14
100	100	10	100	67.08
100	100	10	1000	68.03
100	100	100	1	72.68
100	100	100	10	65.14
100	100	100	100	75.01
100	100	100	1000	75.26
100	100	1000	1	64.70
100	100	1000	10	55.27
100	100	1000	100	75.24
100	100	1000	1000	74.79
100	1000	1	1	65.79
100	1000	1	10	65.15
100	1000	1	100	60.47
100	1000	1	1000	61.25
100	1000	10	1	67.20
100	1000	10	10	63.50
100	1000	10	100	66.72
100	1000	10	1000	61.48
100	1000	100	1	67.65
100	1000	100	10	66.88
100	1000	100	100	75.71
100	1000	100	1000	75.40
100	1000	1000	1	57.90
100	1000	1000	10	69.47
100	1000	1000	100	75.91
100	1000	1000	1000	75.49

Table 12: Tuning the hyperparameters λ_0, ν, ω and τ using grid search.

λ_0	ν	ω	τ	Accuracy (%)
1000	1	1	1	64.42
1000	1	1	10	62.57
1000	1	1	100	65.63
1000	1	1	1000	62.00
1000	1	10	1	64.62
1000	1	10	10	65.87
1000	1	10	100	65.36
1000	1	10	1000	65.51
1000	1	100	1	71.64
1000	1	100	10	65.88
1000	1	100	100	61.27
1000	1	100	1000	71.41
1000	1	1000	1	66.75
1000	1	1000	10	74.07
1000	1	1000	100	54.60
1000	1	1000	1000	71.88
1000	10	1	1	63.93
1000	10	1	10	63.35
1000	10	1	100	63.68
1000	10	1	1000	64.02
1000	10	10	1	71.02
1000	10	10	10	65.37
1000	10	10	100	66.78
1000	10	10	1000	64.47
1000	10	100	1	56.41
1000	10	100	10	60.10
1000	10	100	100	68.24
1000	10	100	1000	61.10
1000	10	1000	1	55.50
1000	10	1000	10	66.23
1000	10	1000	100	65.71
1000	10	1000	1000	72.19

λ_0	ν	ω	τ	Accuracy (%)
1000	100	1	1	63.52
1000	100	1	10	66.93
1000	100	1	100	65.36
1000	100	1	1000	64.37
1000	100	10	1	55.54
1000	100	10	10	62.08
1000	100	10	100	64.24
1000	100	10	1000	62.66
1000	100	100	1	63.72
1000	100	100	10	69.01
1000	100	100	100	61.42
1000	100	100	1000	64.31
1000	100	1000	1	60.21
1000	100	1000	10	71.92
1000	100	1000	100	67.46
1000	100	1000	1000	75.28
1000	1000	1	1	63.09
1000	1000	1	10	68.08
1000	1000	1	100	63.58
1000	1000	1	1000	59.18
1000	1000	10	1	69.81
1000	1000	10	10	68.01
1000	1000	10	100	67.49
1000	1000	10	1000	66.57
1000	1000	100	1	59.24
1000	1000	100	10	61.69
1000	1000	100	100	66.04
1000	1000	100	1000	61.54
1000	1000	1000	1	71.65
1000	1000	1000	10	58.39
1000	1000	1000	100	71.65
1000	1000	1000	1000	75.59

Table 13: Tuning the hyperparameters λ_0, ν, ω and τ using grid search.

Edward Bortnikov, Ari Frank, Eshcar Hillel, and Sriram Rao. Predicting execution bottlenecks in map-reduce clusters. In *HotCloud*, 2012.

Rich Caruana. Multitask learning: A knowledge-based source of inductive bias. In *ICML*, 1993.

Yanpei Chen, Archana Ganapathi, Rean Griffith, and Randy Katz. The case for evaluating mapreduce performance using workload suites. In *Proceedings of the 2011 IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*, 2011.

Yanpei Chen, Sara Alspaugh, and Randy H. Katz. Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads. *PVLDB*, 5(12), 2012.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3), 1995.

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *OSDI*, 2004.

Christina Delimitrou and Christos Kozyrakis. Quasar: Resource-efficient and qos-aware cluster management. In *ASPLOS*, 2014.

David L Donoho. For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution. *Communications on pure and applied mathematics*, 59(6), 2006.

Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *KDD*, 2004.

Theodoros Evgeniou, Charles A Micchelli, Massimiliano Pontil, and John Shawe-Taylor. Learning multiple tasks with kernel methods. *JMLR*, 6(4), 2005.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9, 2008.

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. In *SOSP*, 2003.

Shekhar Gupta, Christian Fritz, Bob Price, Roger Hoover, Johan Dekleer, and Cees Witteveen. Throughputscheduler: Learning to schedule on heterogeneous hadoop clusters. In *ICAC*, 2013.

Michael Isard, Mihai Budiu, Yuan Yu, Andrew Birrell, and Dennis Fetterly. Dryad: Distributed data-parallel programs from sequential building blocks. In *EuroSys*, 2007.

Laurent Jacob, Jean philippe Vert, and Francis R. Bach. Clustered multi-task learning: A convex formulation. In *NIPS*. 2009.

Marius Kloft, Ulf Brefeld, Sören Sonnenburg, and Alexander Zien. lp-norm multiple kernel learning. *J. Mach. Learn. Res.*, 12:953–997, July 2011. ISSN 1532-4435.

YongChul Kwon, Magdalena Balazinska, Bill Howe, and Jerome Rolia. Skewtune: Mitigating skew in mapreduce applications. In *SIGMOD*, 2012.

Henry Li. *Introducing Windows Azure*. Apress, Berkely, CA, USA, 2009. ISBN 143022469X, 9781430224693.

Shibin Parameswaran and Kilian Q. Weinberger. Large margin multi-task metric learning. In *NIPS*. 2010.

John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3), 1999.

Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye. Trace norm regularization: reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 20 (6), 2010.

Ariadna Quattoni, Michael Collins, and Trevor Darrell. Transfer learning for image classification with sparse prototype representations. In *CVPR*, 2008.

Bernardino Romera-Paredes, Hane Aung, Nadia Bianchi-Berthouze, and Massimiliano Pontil. Multilinear multitask learning. In *ICML*, 2013.

M Signoretto, R Langone, M Pontil, and J Suykens. Graph based regularization for multilinear multitask learning. 2014.

Siddharth Suri and Sergei Vassilvitskii. Counting triangles and the curse of the last reducer. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 607–614, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0632-4. doi: 10.1145/1963405.1963491. URL `http://doi.acm.org/10.1145/1963405.1963491`.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *NIPS*, 1996.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *ICCV*, October 2007.

Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition, 2009. ISBN 0596521979, 9780596521974.

Christian Widmer, Yasemin Altun, and Nora C. Toussaint. Multitask multiple kernel learning (mt-mkl), 2010.

Kishan Wimalawarne, Masashi Sugiyama, and Ryota Tomioka. Multitask learning meets tensor factorization: task imputation via convex optimization. In *NIPS*, 2014.

Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-task learning for classification with dirichlet process priors. *JMLR*, 8, 2007.

Neeraja J. Yadwadkar, Ganesh Ananthanarayanan, and Randy Katz. Wrangler: Predictable and faster jobs using fewer resources. In *Proceedings of the ACM Symposium on Cloud Computing*, SOCC '14, pages 26:1–26:14, New York, NY, USA, 2014. ACM.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Matei Zaharia, Andy Konwinski, Anthony D. Joseph, Randy H. Katz, and Ion Stoica. Improving mapreduce performance in heterogeneous environments. In *OSDI*, 2008.

Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In *Eurosys*, 2010.

Journal of Machine Learning Research 17 (2016) 1-31 Submitted 5/15; Revised 4/16; Published 6/16

Interleaved Text/Image Deep Mining on a Large-Scale Radiology Database for Automated Image Interpretation

Hoo-Chang Shin HOOCHANG.SHIN@NIH.GOV
Le Lu LE.LU@NIH.GOV
Lauren Kim LAUREN.KIM2@NIH.GOV
Ari Seff ARI.SEFF@NIH.GOV
Jianhua Yao JYAO@CC.NIH.GOV
Ronald M. Summers RMS@NIH.GOV
Imaging Biomarkers and Computer-Aided Diagnosis Laboratory
Radiology and Imaging Sciences
National Institutes of Health Clinical Center
Bethesda, MD 20892-1182, USA

Editor: Benjamin M. Marlin, C. David Page, and Suchi Saria

Abstract

Despite tremendous progress in computer vision, there has not been an attempt to apply machine learning on very large-scale medical image databases. We present an interleaved text/image deep learning system to extract and mine the semantic interactions of radiology images and reports from a national research hospital's Picture Archiving and Communication System. With natural language processing, we mine a collection of ∼216K representative two-dimensional images selected by clinicians for diagnostic reference and match the images with their descriptions in an automated manner. We then employ a weakly supervised approach using all of our available data to build models for generating approximate interpretations of patient images. Finally, we demonstrate a more strictly supervised approach to detect the presence and absence of a number of frequent disease types, providing more specific interpretations of patient scans. A relatively small amount of data is used for this part, due to the challenge in gathering quality labels from large raw text data. Our work shows the feasibility of large-scale learning and prediction in electronic patient records available in most modern clinical institutions. It also demonstrates the trade-offs to consider in designing machine learning systems for analyzing large medical data.

Keywords: Deep learning, Convolutional Neural Networks, Topic Models, Natural Language Processing, Medical Imaging

1. Introduction

The ImageNet Large Scale Visual Recognition Challenge by Deng et al. (2009) provides more than one million labeled images of 1,000 object categories. The accessibility of a huge amount of well-annotated image data in computer vision rekindled deep convolutional neural networks (CNNs) as the premier learning tool to solve the visual object class recognition tasks, as shown by Krizhevsky et al. (2012); Simonyan and Zisserman (2015); Szegedy et al. (2015). Deep CNNs can perform significantly better than traditional shallow learning methods but usually require much more training data as was shown by Krizhevsky et al. (2012);

Russakovsky et al. (2015). In the medical domain, however, there are no similar large-scale labeled image data sets available. On the other hand, large collections of radiology images and reports are stored in many modern hospitals' Picture Archiving and Communication Systems (PACS). The invaluable semantic diagnostic knowledge inhabiting the mapping between hundreds of thousands of clinician-created high-quality text reports and linked image volumes remains largely unexplored. One of our primary goals is to extract and associate radiology images with clinically semantic labels via interleaved text/image data mining and deep learning on a large-scale PACS database (∼780K imaging examinations). To the best of our knowledge, this is the first reported work performing automated mining and prediction on a hospital PACS database at a very large scale.

The Radiology reports are text documents describing patient history, symptoms, image observations and impressions written by board-certified radiologists. However, the reports do not contain specific image labels to be trained by a machine learning algorithm. Building the ImageNet database (Deng et al., 2009) was mainly a manual process: harvesting images returned from Google image search engine according to the WordNet (Miller, 1995) ontology hierarchy and pruning falsely tagged images using crowd-sourcing such as Amazon Mechanical Turk (AMT). This does not meet our data collection and labeling needs due to the demanding difficulties of medical annotation tasks and the need for data privacy. Thus, we first propose to mine categorical semantic labels using a non-parametric topic modeling method—latent Dirichlet Allocation (LDA) by Blei et al. (2003)—to provide a semantic interpretation of a patient image in three levels. While this provides a first-level interpretation of a patient image, labeling based on categorization can be nonspecific. To alleviate the issue of non-specificity, we further mine specific disease words in the reports mentioning the images. Feed-forward CNNs were then used to train and predict the presence/absence of the specific disease categories.

Our work has been inspired by the works of Deng et al. (2009); Russakovsky et al. (2015) building very large-scale image databases and the works establishing semantic connections of texts and images by Kulkarni et al. (2013). Please note that there has not yet been much comparable development on large-scale medical imaging interpretation. Kulkarni et al. (2013) have spearheaded the efforts of learning the semantic connections between image contents and the sentences describing them, such as image captions. Detecting objects of interest, attributes and prepositions and applying contextual regularization with a conditional random field (CRF) is a feasible approach as shown by Kulkarni et al. (2013), and many useful tools for image annotation using it are available in computer vision.

In this work, both deep feed-forward CNNs of Krizhevsky et al. (2012); Simonyan and Zisserman (2015) and word-embedding networks of Mikolov et al. (2013a,b) are used to model image and text. Also, the CNN parameters pre-trained on ImageNet are used to initialize CNNs for medical image analysis. We show the benefit of this transfer learning and domain adaptation in Section 4.2. The fact that deep learning requires no hand-crafted image features is very desirable since significant adaptation would be needed to apply conventional image features, (for example, HOG, SIFT) to medical images. The large-scale data sets of extracted representative images (referred to as "key images" in this paper) and their categorization, vector labels, and describing sentences can be harnessed to alleviate deep learning's "data-hungry" challenge in the medical domain.

1.1 Related Work

The ImageCLEF medical image annotation tasks of 2005-2007 by Deselaers and Ney (2008) have 9,000 training and 1,000 test images, converted to 32×32 pixel thumbnails with 57 labels. Local image descriptors and intensity histograms are used as a bag-of-features approach for this scene-recognition-like problem. However, the data set is limited to radiographs (for example, chest and bone x-rays), and it is difficult to detect any disease from 32×32 size images. Unsupervised LDA-based matching from lung disease words (for example, fibrosis, emphysema) to two-dimensional image blocks from axial CT chest scans is studied by Carrivick et al. (2005) where data were collected from a relatively small number (24) of patients. The works of Barnard et al. (2003); Blei and Jordan (2003) using generative models of combining words and images under a very limited word/image vocabulary has also motivated this study.

Socher et al. (2013); Frome et al. (2013) first map words into vector space using recurrent neural networks and then project images into the label-associated word-vector embeddings by minimizing the L_2 (Socher et al., 2013) or hinge rank losses (Frome et al., 2013) between the visual and label manifolds. The language model is trained on the texts of Wikipedia and tested on label-associated images from the CIFAR (Krizhevsky and Hinton, 2009; Socher et al., 2013) and ImageNet data sets (Deng et al., 2009; Frome et al., 2013). Image-to-language correspondence was learned from the ImageNet data set and reasonably high quality image description data sets (Pascal1K (Rashtchian et al., 2010), Flickr8K (Hodosh et al., 2013), Flickr30K (Young et al., 2014), MS-COCO Lin et al. (2014)) by Karpathy et al. (2014); Vinyals et al. (2015); Donahue et al. (2015); Xu et al. (2015); Mao et al. (2015), where such caption data sets are not available in the medical domain.

The tasks of mining and labeling images from a data set of blog posts with user photos and related texts and retrieving them with query words were demonstrated in Kim et al. (2015b,a, 2014). Similarly, a noisy image-text data set consisting of product photos (such as bags, clothing and shoes) and their associated text description (Berg et al., 2010) was used to demonstrate image retrieval with text queries and image description generation. Nonetheless, they all require pre-trained models either from the large ImageNet data set or a large text data set (for example, word representations trained on Wikipedia or Reuters news data sets (Turian et al., 2010)). Still there exists no such large data set of images and texts in the medical domain.

Graphical models have been employed to predict image attributes by Lampert et al. (2014); Scheirer et al. (2012), or to describe images by Kulkarni et al. (2013) using manually annotated data sets. Automatic label mining on large, unlabeled data sets is presented by Ordonez et al. (2011); Jaderberg et al. (2014), however, the variety of the label-space is limited to image text annotations. In this work, we demonstrate the automatic generation of descriptive attributes of patient images as well as the detection of frequent disease types with associated confidences. A large data set of patient images and radiologist text reports from a hospital is used for the demonstration, and we highlight the key issues to consider when analyzing large-scale medical data with minimal annotation.

total number of		# words in documents		# image modalities	
# documents	∼780k	`mean`	131.13	CT	∼169k
# images	∼216k	`std`	95.72	MR	∼46k
# words	∼1 billion	`max`	1502	PET	67
# vocabulary	∼29k	`min`	2	others	34

Table 1: Some statistics of the data set. "Others" include computed radiography, and ultrasound.

2. Data

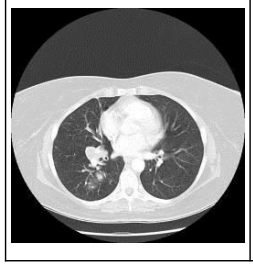
To gain the most comprehensive interpretation of diagnostic semantics, we use all available radiology reports of around 780,000 imaging examinations, stored in the PACS of National Institutes of Health Clinical Center since the year 2000. Around 216,000 two-dimensional representative image slices referred by doctors are studied here, instead of using all three-dimensional image volumes. Within three-dimensional patient scans, most of the imaging information represented are normal anatomy, therefore they are often not the focus of the radiology reports. The two-dimensional "key images" referenced by radiologists manually during radiology report writing provide a visual reference to pathologies or other notable findings (Figure 1). Therefore, the two-dimensional key images are more correlated with the diagnostic semantics in the reports than the whole three-dimensional scans, but not all reports have referenced key images (215,786 images from about 61,845 unique patients). Table 1 provides some statistics of the extracted database, and Table 2 shows examples of the most frequently occurring words in the radiology reports collected. Leveraging our deep learning models exploited in this paper will make it possible to automatically select key images from three-dimensional patient scans to avoid mis-referencing.

Finding and extracting key images from radiology reports is done by natural language processing (NLP), that is, finding a sentence mentioning a referenced image. For example, "*There may be mild fat stranding of the right parapharyngeal soft tissues (series 1001, image 32)*" is listed in Figure 1. The NLP steps are sentence tokenization, word/number matching and stemming, and rule-based information extraction (for example, translating "image 1013-78" to "images 1013-1078"). A total of ∼187K images are retrieved and matched this way, whereas the rest of ∼28K key images were extracted according to their reference accession numbers in PACS. The image-text matching is accurate as we use exact annotations from the sentences in reports in retrieving the images, however, it is possible we missed some image-text pairs due to limitations in our NLP pipelines. We do not evaluate the recall-rate of our method in this study, but it can be considered as a future work. The software package of Bird et al. (2009) is used for the basic NLP pipelines.

3. Document Topic Learning with Latent Dirichlet Allocation

It is difficult to annotate the ∼216K images and the sentences referring to them. Unlike the images of ImageNet (Deng et al., 2009; Russakovsky et al., 2015) which often have a dominant object appearing in the center, our key images are mostly CT and MRI slices showing

0001 REPORT : REASON FOR EXAM (Entered by ordering clinician into CRIS): hx of head and neck cancer. needs scan CT of the nasopharynx.
HISTORY: Head and neck cancer.
TECHNIQUE: Contiguous 2.5 mm axial images of the nasopharynx were performed without IV contrast. COMPARISON: xx/xx/xxxx.
FINDINGS: No soft tissue masses are seen within the soft tissues of the neck. The parotid and submandibular glands are predominantly fatty-replaced. Soft tissues of the Naso, oropharynx are unremarkable. There may be mild fat stranding of the right parapharyngeal soft tissues (series 1001, image 32). No abnormal masses are seen at that site. No bulky lymphadenopathy is seen. There is a fusiform aneurysm of the basilar artery as previously described. It appears to the mildly increased in size and currently measures 2.0 cm in transverse dimensions and previously measured 1.8 cm. It measured 1.5 cm in transverse dimensions on xx/xx/xxxx. Atherosclerotic calcifications are also seen within the carotid arteries bilaterally. There is near-complete opacification of the maxillary sinuses bilaterally. This has increased predominantly within the left maxillary sinus and mildly within the right maxillary sinus. The ethmoidal air cells are clear. Sphenoidal and frontal sinuses are clear. Degenerative changes of the cervical spine are noted.
IMPRESSIONS: 1. No soft tissue masses however, mild right parapharyngeal fat stranding is seen it may be postoperative or post radiation in nature. 2. Basilar artery aneurysm that has gradually increased in size when compared to prior examinations. 3. Atherosclerotic disease of the coronary arteries bilaterally..



0001 Report: CHEST, ABDOMEN, PELVIS CT: Multidetector helical (5 mm, quad) images following, and abdomen images prior to vascular contrast infusion (45 s delay, 2 cc/s, 130 cc Isovue) obtained without apparent complication. History: renal cell pt on Medarex protocol here for end of course evaluation.
CHEST: Multiple right, and at least one left lung masses minimally-moderately increasing since xx/xx/xxxx, compatible with metastases despite moderate decrease in at least one right mid-lung mass (e.g. series 4 image 30). Minimal pretracheal and subcarinal adenopathy increasing. Spine osteophytes. Enlargement thyroid on right side, and thyroid heterogeneity unchanged, possibly due to goiter. No evidence of pleural or pericardial effusion, axilla or left hilum adenopathy.
ABDOMEN, PELVIS: Few right and left liver foci, left periaortic and left adrenal fossa, and right sacrum mass and lytic lesion (series 3 image 88-95) increasing minimally, compatible with metastases. Scattered lumbar vertebra and bilateral ilium foci (e.g. series 3 image 55, 60, 80, 84-7, 96) foci possibly due to bone metastases. Uterine fundus focus (series 3 image 95) increasing in density since xx/xx/xxxx, possibly due to fibroid, metastasis. No evidence of splenomegaly, hydronephrosis, gallbladder calcification, or bulky mesenteric adenopathy.

Figure 1: Two examples of radiology reports and the referenced "key images" (providing a visual reference to pathologies or other notable findings).

right	937k	images	312k	contrast	260k	unremarkable	195k
left	870k	seen	299k	axial	253k	lower	195k
impression	421k	mass	296k	lung	243k	upper	192k
evidence	352k	normal	278k	bone	219k	lesion	180k
findings	340k	small	275k	chest	208k	lobe	174k
CT	312k	noted	263k	MRI	204k	pleural	172k

Table 2: Examples of the most frequently occurring words in the radiology report documents.

several organs usually with pathologies. There is a high amount of intrinsic ambiguity in defining and assigning a semantic label set to images, even for experienced clinicians. We therefore propose to mine image categorization labels using the non-parametric topic-modeling algorithm of Blei et al. (2003) on the ∼780K radiology text reports in PACS. Our

hypothesis is that the large collection of radiology reports statistically defines the categories meaningful for topic-mining and visual correspondence learning for these topics.

Latent Dirichlet Allocation (LDA) was originally proposed by Blei et al. (2003) to find latent topics for a collection of text documents such as newspaper articles. There are some other popular methods for document topic modeling, such as Probabilistic Latent Semantic Analysis (pLSA) by Hofmann (1999) and Non-negative Matrix Factorization (NMF) by Lee and Seung (1999). In a study done by Stevens et al. (2012) LDA showed the most favorable results overall in human evaluations of the generated topics compared to other popular methods. Furthermore, pLSA can be regarded as a special case of LDA (Girolami and Kabán, 2003) and NMF as a semi-equivalent model of pLSA (Gaussier and Goutte, 2005; Ding et al., 2006).

LDA offers a hierarchy of extracted topics and the number of topics can be chosen by evaluating each model's *perplexity score* (Equation 1), which is a common way to measure how well a probabilistic model generalizes by evaluating the log-likelihood of the model on a held-out validation set. For an unseen document set D_{val}, the perplexity score is defined as in Equation 1, where M is the number of documents in the validation set, $\mathbf{w}_d$ the words in the unseen document d, N_d the number of words in document d, with $\mathbf{\Phi}$ the topic matrix, and α the hyper-parameter for topic distribution of the documents.

$$perplexity(D_{val}) = \exp\left\{-\frac{\sum_{d=1}^{M}\log p(\mathbf{w}_d|\mathbf{\Phi},\alpha)}{\sum_{d=1}^{M}N_d}\right\} \quad (1)$$

A lower perplexity score generally implies a better fit of the model for a given document set (Blei et al., 2003).

Based on the perplexity score evaluated on 80% of the total documents used for training and 20% used for validation, the number of topics chosen is 80 for the document-level model using perplexity scores for model selection (Figure 2). Although the document distribution in the topic space is approximately balanced, the distribution of image counts for the topics is more unbalanced (Figure 3). Specifically, topic #77 (non-primary metastasis spreading across a variety of body parts) contains nearly half of the ∼216K key images. To address this data bias, sub-topics are obtained for each of the first document-level topics, resulting in 800 topics, where the number of the sub-topics is also chosen based on the average perplexity scores evaluated on each document-level topic. Lastly, to compare the method of using the whole report with using only the sentence directly describing the key images for latent topic mining, sentence-level LDA topics are obtained based on three sentences only: the sentence mentioning the key-image (Figure 1) and its adjacent sentences as proximal context. The perplexity scores keep decreasing with an increasing number of topics; we choose the topic count to be 1000 as the rate of the perplexity score decrease is very small beyond that point (Figure 2).

We observe that LDA-generated image categorization labels are valid, demonstrating good semantic coherence among clinician observers. Some examples of document-level topics with their corresponding images and topic key words are shown in Figure 4. All reports and sentences referring to the images have associated topics, and images are sampled from the sentences belonging to the multi-level topics. The lists of key words and sampled images per topic label are subjected to a board-certified radiologist's review and validation.

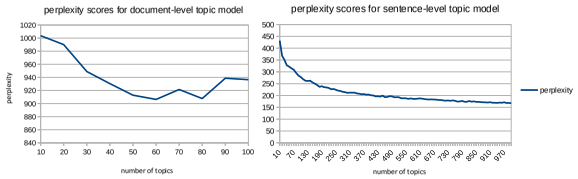


Figure 2: Perplexity scores for document-/sentence- level topic models. Number of topics with low perplexity score is selected as the optimal (80 for document-level, 1000 for sentence-level).

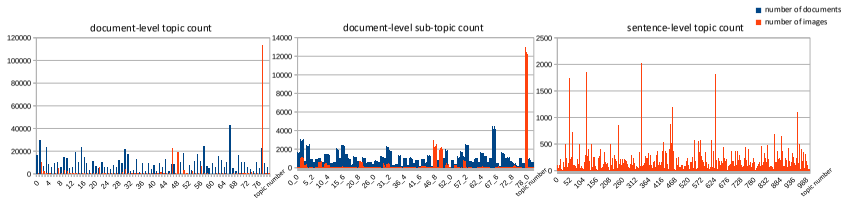


Figure 3: Distribution of documents and images for document-level topic, document-level sub-topic, and sentence-level topic. Sixth sub-topic (second-level topic) of (first-level) document topic 41 is noted as 40_5.

There are 73 low-level concepts, for example, pathology examination of certain body regions and organs: topic #47 - sinus diseases; #2 - lesions of solid abdominal organs, primarily kidney; #10 - pulmonary diseases; #13 - brain MRI; #19 - renal diseases on mixed imaging modalities; #36 - brain tumors. There are 7 mid- to high-level concepts, such as: topic #77 - non-primary metastasis spreading across a variety of body parts; topic #79 - cases with high diagnosis uncertainty or equivocation; #72 - indeterminate lesions; #74 - instrumentation artifacts limiting interpretation. Low-level topic images tend to be visually more coherent than the higher-level topic images.

High-level topics may be analogous to the high-level visual concepts in natural images as was studied by Kiapour et al. (2014); Ordonez and Berg (2014). About half of the key images are associated with topic #77, implying that the clinicians' image referencing behavior patterns heavily focuses on metastatic patients. Sub-topics of document-level topic #77 are sub-categories of metastatic disease, for example: #77-0 - abdominal mass; #77-2 - bulky tumor; #77-4 - multifocal metastatic disease; #77-9 - liver tumor. Meanwhile, some of the sub-topics of document-level #77 do not seem very focused. Many of the sentence-level topics have valid semantics too, for example, 'renal imaging', 'musculoskeletal imaging',

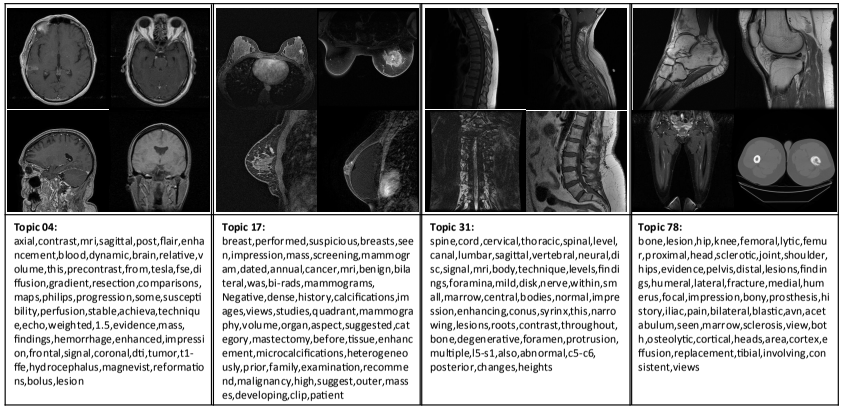


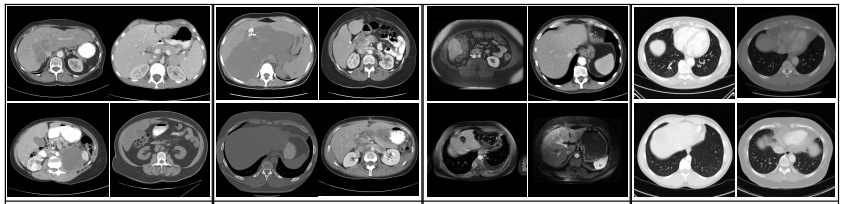
Figure 4: Examples of LDA generated document-level topics with corresponding images and key words. Topic #4 is MRI of brain tumor; topic #17: breast imaging; topic #31: degenerative spine disc disease; and topic #78: bone metastases. These are verified by a radiologist.

'chest port catheter', 'chest imaging with disease or pathology', and 'degenerative disease in bone'.

We also obtained LDA topics on the reports having associated images only, resulting in 20 topics according to perplexity score. However, these did not add any more meaningful semantics in addition to the already obtained topics in three levels, so that we do not include the topics. For more details and the image-topic associations, refer to Figures 4, 5, and the supplementary material. Even though LDA labels are computed with text information only, we next investigate the plausibility of mapping images to the topic labels of different levels via deep CNN models.

4. Image to Document Topic Mapping with Deep Convolutional Neural Networks

For each level of topics discussed in Section 3, we train deep CNNs to map the images into document categories using the Caffe framework of Jia et al. (2014). We split our whole key image data set as follows: 85% used as the training data set, 5% as the validation, and 10% as the test data set. If a topic has too few images to be divided into training/validation/test for deep CNN learning, then that topic is neglected for the CNN training. These cases are normally the topics of rare imaging protocols, for example: topic #5 - Abdominal ultrasound; topic #28& #49 - DEXA scans of different usages. In total, 60 topics were used for the document-level topic mapping, 385 for the document-level sub-topic mapping,



Topic 77-0:	Topic 77-2:	Topic 77-5:	Topic 77-9:
kidney,images,abdomen,e.g,prior,mass,pancreas,following,cysts,adrenal,liver,foci,renal,contrast,approximate,including,focus,cyst,bilateral,masses,size,enhancing,for,also,given,possibly,mid,2.5,vascular,without,due,nephrectomy,please,1.5,from,few,multiphase,subcentimeter,least,comparison,patient,dual-phase,length,apparent,complication,obtained,upper,study,lower,vhl	bulky,pelvis,bone,gross,since,liver,abdomen,calcification,vascular,study,lung,mass,isovue,dfov,without,contrast,administration,impression,metastasis,chest,for,images,mesenteric,axilla,following,hilum,cc/s,helical,multidetector,ascites,enteric,reason,apparent,complication,pleural,splenomegaly,pericardial,hydronephrosis,delay,effusion,mediastinum,obtained,300,spine,gallbladder,report,130,retroperitoneal,spleen,e.g	images,axial,t1-weighted,without,prior,liver,following,t2-weighted,tesla,fat-suppressed,multiple,sequences,e.g,characteristic,obtained,1.5,foci,fat,abdomen,for,prolonged,coronal,including,relaxation,hydronephrosis,mri,magnevist,splenomegaly,complication,apparent,vascular,pleural,impression,report,effusion,contrast,reason,study,mass,administration,since,focus,multiphase,definite,echo,defect,gross,filling,ascites,into	lung,chest,pleural,images,bilateral,minimal,effusion,lower,obtained,pericardial,multidetector,helical,axilla,study,report,mass,infiltrate,for,scarring,since,bulky,and/or,clinical,splenomegaly,dfov,cavity,e.g,impression,decreasing,infiltrates,focal,mediastinum,disease,atelectasis,hydronephrosis,small,reason,upper,untoward,history,probable,appearing,calcification,lobe,8-channel,supine,scattered,prone,bone,intervals
Document-level Topic 77: compatible,adenopathy,series,unchanged,image,evidence,images,e.g,pelvis,lung,since,abdomen,vascular,minimal,foci,bulky,mass,calcification,bone,chest,contrast,liver,effusion,pleural,obtained,gross,following,without,splenomegaly,axilla,hydronephrosis,metastasis,bilateral,pericardial,increasing,helical,multidetector,apparent,complication,hilum,due,spine,gallbladder,administration,mesenteric,fat,dfov,cc/s,appearing,delay			

Figure 5: Examples of some sub-topics of document-level topic #77, with corresponding images and topic key-words. The key-words and the images for the document-level topic (#77) indicates metastatic disease. The key-words for topic #77 are: [abdomen,pelvis,chest,contrast,performed,oral,was,present,masses,stable,intravenous,adenopathy, liver,retroperitoneal,comparison,administration,scans,130,small,parenchymal,mediastinal,dated, after,which,evidence,were,pulmonary,made,adrenal,prior,pelvic,without,cysts,spleen,mass,disease, multiple,isovue-300,obtained,areas,consistent,nodules,changes,pleural,lesions,following,abdominal, that,hilar,axillary].

and 717 for the sentence-level mapping. Systematic diagrams showing how each level of semantic topics are learned, assigned to images, and trained to map from images to topics are shown in Figure 6.

4.1 Implementation

All our CNN network settings are similar or the same as the ImageNet Challenge "AlexNet" (Krizhevsky et al., 2012), and "VGG-16 & 19" (Simonyan and Zisserman, 2015) models. For "AlexNet" we use the Caffe reference network of Jia et al. (2014), which is a slight modification to the "AlexNet" by Krizhevsky et al. (2012). The AlexNet model by Krizhevsky et al. (2012) has about 60 million parameters (650,000 neurons) and consists of five convolutional layers (1st, 2nd and 5th followed by max-pooling layers), and three fully-connected (FC) layers with a final classification layer. The VGG variations of CNN models by Simonyan and Zisserman (2015) are significantly deeper by having 16-19 convolutional layers and 133-144 million parameters. The top-1 error rates on ImageNet data set of these models are AlexNet: 15.3% (Krizhevsky et al., 2012); VGG-16: 7.4%; and VGG-19: 7.3% (Simonyan and Zisserman, 2015), respectively.

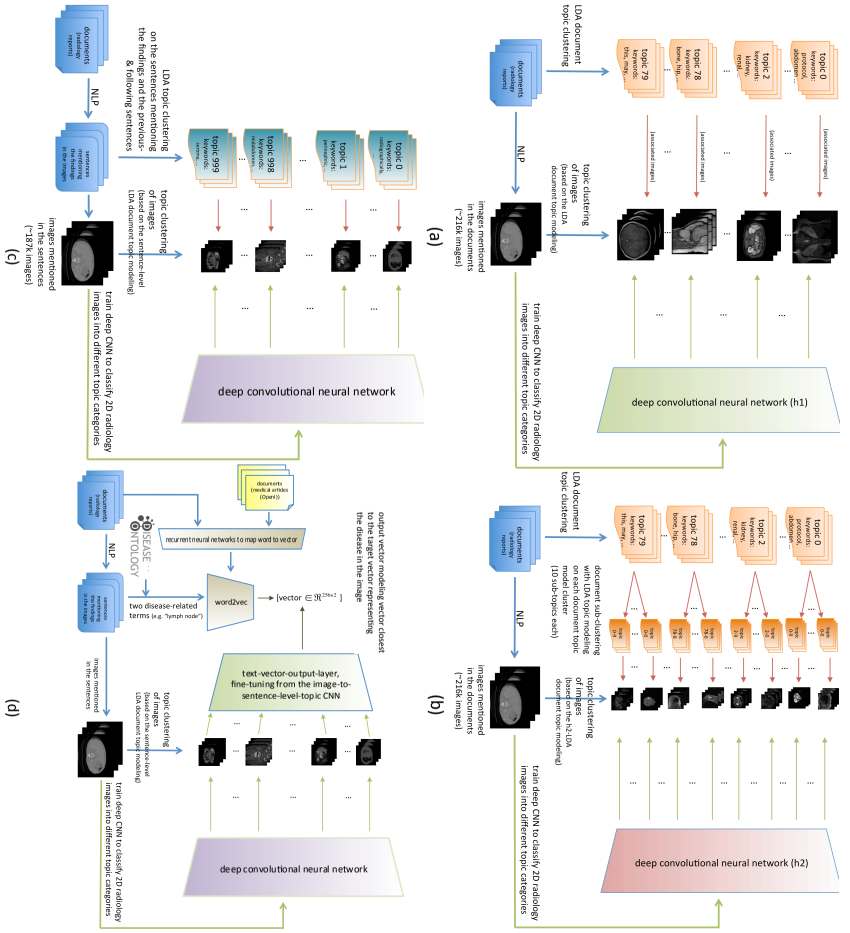


Figure 6: Systematic diagrams for training CNNs to learn to classify images into (a) document-level topics (b) document-level sub-topics, (c) sentence-level topics. A systematic diagram for image-to-word model (in Section 5.3) is shown in (d).

For the image to topic mapping, we change the numbers of output nodes in the last softmax classification layer, that is, 60, 385 and 717 for the document-level, document-level

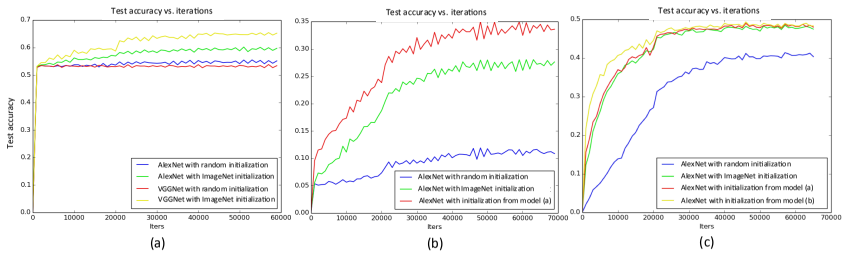


Figure 7: Traces of classification accuracies during training, showing the benefits of using ImageNet data set as pre-training for this task with medical images and improvements of fine-tuning from CNN neural networks of similar tasks (for example, from document-level (h1) CNN model to document-level sub-topic (h2) CNN model). (a) Image-to-document-level-topic (h1) classification, (b) image-to-document-level-sub-topic (h2) classification, and (c) image-to-sentence-level-topic (h3) classification.

sub-topics, and sentence-level respectively. The networks for first-level semantic labels are fine-tuned from the pre-trained ImageNet models, where the networks for the lower-level semantic labels are fine-tuned from the models of the higher-level semantic labels.

4.2 Transfer Learning and Domain Adaptation

We find that transfer learning from the ImageNet pre-trained CNN parameters on natural images to our medical image modalities (mostly CT, MRI) significantly helps the image classification performance. Additionally, transfer learning from a CNN trained for a more related task (for example, from CNN trained on the image-to-document-level-topic models to train CNN for the image-to-document-level-sub-topic model) is found to be more effective than from a CNN trained for a less related task (for example, from CNN trained on ImageNet to train CNN for image-to-document-level-sub-topic model). Examples of classification accuracy traces during training using CNNs from random initialization, transfer learning from CNN trained on ImageNet and transfer learning from higher level image-to-topic model to lower level image-to-topic models are shown in Figure 7. Similar findings that deep CNN features can be generalized across different image modalities have been reported by Gupta et al. (2014, 2013) but are empirically verified with only much smaller data sets than ours. Our key image data set is about one-fifth the size of ImageNet (Russakovsky et al., 2015) and is the largest annotated medical image data set to date.

From Figure 7 we can see that: (1) CNN testing accuracy quickly increases from $\sim$0% to 50+% in roughly 1600 iterations due to the unbalanced data distribution among classes in document-level; (2) A more complex, deeper CNN model (VGG-Net) performs better than the model which already is a good benchmark (AlexNet), but only when starting from a good initialization (that is, pre-training via ImageNet models); (3) Fine-tuning from a more closely related task CNN model is even better than fine-tuning from less related task model (alexnet_tp80_h2_start_tp80h1 > alexnet_tp80_h2_start_imagenet).

With these findings, we train our CNN models with transfer-learning by default for the remaining parts of our study. All the CNN layers except the newly modified ones are initialized with the weights of a previously trained related model and trained with a new task with a low learning rate of 0.001. The modified layers with a new number of classes are initialized randomly, and their learning rates are set with a higher learning rate of 0.01. All the key images are re-sampled to a spatial resolution of 256×256 pixels. Then we follow the approach of Simonyan and Zisserman (2015) to crop the input images from 256×256 to 227×227 for training.

4.3 Classification Results and Discussion

We would expect that the level of difficulties for learning and classifying the images into the LDA-induced topics will be different for each semantic level. Low-level semantic classes can have key images of axial/sagittal/coronal slices with position variations and across MRI/CT modalities. Mid- to high-level concepts all demonstrate much larger within-class variations in their visual appearance since they are diseases occurring within different organs and are only coherent at high-level semantics. Table 3 provides the top-1 and top-5 testing in classification accuracies for each level of topic models using AlexNet (Krizhevsky et al., 2012), and VGG-16&19 Simonyan and Zisserman (2015) based deep CNN models.

All top-5 accuracy scores are significantly higher than top-1 values, for example, increasing from 0.658 to 0.946 using VGG-19, or 0.607 to 0.929 via AlexNet in document-level. This indicates that the classification errors or fusions are not uniformly distributed among other false classes. Latent "blocky subspace of classes" may exist in our discovered label space, where several topic classes form a tightly correlated subgroup. The confusion matrices in Figure 8 verify this finding.

It is shown that the deeper models (VGG-16&19) perform consistently better than the shallower 8-layer model (AlexNet) in classification accuracy, especially for document-level sub-topics. While the images of some topic categories and some body parts are easily distinguishable as shown in Figure 4, the visual differences in abdominal parts are rather subtle as in Figure 5. Distinguishing the subtleties and high-level concept categories in the images could benefit from a more complex model so that the model can handle these subtleties.

It is also noticeable that VGG-16&19 models require significantly more computational resource and time to train than the shallower model. Table 4 shows the memory consumption and time required to train the CNN models for the image-to-sentence-level-topic model with up to 70,000 iterations using the NVidia Tesla K40 GPU. However, comparing VGG-16 and VGG-19, three additional convolutional layers seem to have contributed to raise the top-5 accuracies by a small amount ($\sim$2%), which is coherent with the results reported by Simonyan and Zisserman (2015) for object recognition task on the ImageNet data set.

Compared with the ImageNet 2014 results, top-1 error rates are moderately higher (34% versus 30%) and top-5 test errors ($6\% - 8\%$) are comparable. In summary, our quantitative results are very encouraging, but there also exist some uncertainties in annotations because labels stem from an unsupervised learning algorithm. Multi-level semantic concepts

	AlexNet 8-layers		VGG 16-layers		VGG 19-layers	
	top-1	top-5	top-1	top-5	top-1	top-5
document-level	0.61	0.93	0.66	0.93	0.66	0.95
document-level-h2	0.33	0.56	0.54	0.70	0.54	0.70
sentence-level	0.48	0.56	0.50	0.56	0.50	0.58

Table 3: Top-1, top-5 test classification accuracies for image to document-level topics, document-level sub-topics (document-level-h2) and sentence-level topics, using AlexNet (Krizhevsky et al., 2012), and VGG-16&19 (Simonyan and Zisserman, 2015) deep CNN models.

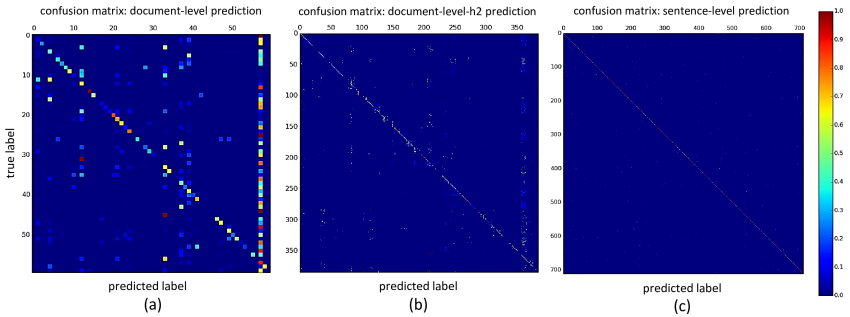


Figure 8: Confusion matrices of (a) document-level topic, (b) document-level sub-topic (document-level-h2), and (c) sentence- level classification Simonyan and Zisserman (2015) ((b) and (c) can be viewed best in electronic version of this document).

show good image learnability by deep CNN models which shed light on the feasibility of automatically parsing very large-scale radiology image databases.

5. Generating Image-to-Text Description

The image-to-topic mapping in Section 4 is a promising first step towards large-scale automated medical image interpretation. However, generating image descriptions as in Lampert et al. (2014); Scheirer et al. (2012); Kulkarni et al. (2013) will be more readily interpretable and descriptive.In addition, key words in the topics can help to understand the content of a given image with more semantic meaning. We therefore propose to generate relevant key-word text descriptions similarly to Kulkarni et al. (2013), using deep language/image CNN models.

	AlexNet 8-layers	VGG 16-layers	VGG 19-layers
time	4 hours 35 mins	3 days 2 hours	4 days 40 mins
memory	~1.4 GBytes	~10 GBytes	~11 GBytes

Table 4: Training times for the CNN models used to reach 70,000 iterations, and their memory consumption, using Caffe framework (Jia et al., 2014) on NVidia Tesla K40 GPU.

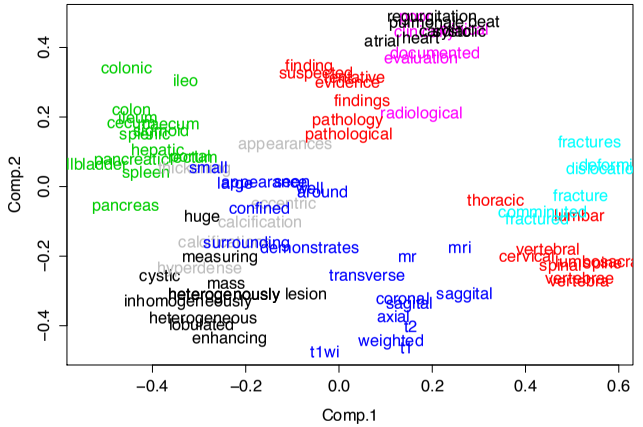


Figure 9: Example words embedded in the vector space using word-to-vector modeling (https://code.google.com/p/word2vec/) visualized on a two-dimensional space, showing (clinical) words with similar meanings are located nearby in the vector space (colors are used to highlight these in visualization).

5.1 Word-to-Vector Modeling and Removing Word-Level Ambiguity

In radiology reports, there exist many recurring word morphisms in text identification, for example, [mr, mri, t1-/t2-weighted] (natural language expressions for imaging modalities of magnetic resonance imaging (MRI)), [cyst, cystic, cysts], [tumor, tumour, tumors, metastasis, metastatic], etc. We train the word-to-vector embedding model of Mikolov et al. (2013c,b,a) to address this word-level labeling space ambiguity while also transforming the words to vectors. A total of 1.2 billion words from our radiology reports as well as from biomedical research articles obtained from OpenI (ope: `http://openi.nlm.nih.gov`) are used. Words with similar meaning are mapped or projected to closer locations in the vector space than dissimilar ones. An example visualization of the word vectors on a two-dimensional space using principal component analysis is shown in Figure 9.

~1.2 billion words with OpenI		*~1.2 billion words with OpenI*		*~1.2 billion words with OpenI*		*~1.2 billion words with OpenI*	
"cyst"		**"heart"**		**"brain"**		**"liver"**	
cysts	0.799191	cardiac	0.672690	hemisphere	0.684149	hepatic	0.764163
hydatid	0.734686	respiratory	0.644453	hemispheric	0.668626	spleen	0.683242
cystic	0.701855	beat	0.642630	cerebellum	0.663902	cirrhotic	0.664428
unilocular	0.654273	pressure	0.558879	whole	0.661564	cirrhosis	0.664262
tailgut	0.639764	murmur	0.551323	regions	0.647632	hcc	0.656473
nonparasitic	0.621647	systolic	0.548490	mri	0.646674	portal	0.610437
epidermoid	0.604492	pericardial	0.538957	structural	0.638171	hepatocellular	0.603930
lipoma	0.588372	dobutamine	0.537429	neuroanatomical	0.636563	parenchyma	0.597169
cheesy	0.586947	intracardiac	0.533799	crinion	0.626951	splenic	0.579957
multiloculated	0.584199	great	0.532735	in	0.626707	hepatomegaly	0.573687
pearly	0.583126	rate	0.531352	parasaggital	0.618392	tumor	0.571135
multilocular	0.582670	beats	0.524729	illustration	0.610440	abdomen	0.559092
lesion	0.579009	atrial	0.524052	striatal	0.609282	hepatectomy	0.556156
tgdc	0.578533	tachycardia	0.521093	brains	0.607442	bclc	0.546798
multiseptate	0.575851	minute	0.520249	behavioral	0.606803	subcapsular	0.542745
~1 billion words reports only		*~1 billion words reports only*		*~1 billion words reports only*		*~1 billion words reports only*	
"cyst"		**"heart"**		**"brain"**		**"liver"**	
cysts	0.768382	lungs	0.526600	t1	0.615066	spleen	0.759884
septated	0.586067	mediastinum	0.517008	mri	0.595027	gallbladder	0.648075
polyp	0.583761	consolidating	0.486605	sagittal	0.580841	hepatomegaly	0.642022
simple	0.534717	pa	0.449816	flair	0.565445	gallstones	0.611837
septation	0.500951	chest	0.433362	t2	0.555053	pancreas	0.608356
parapelvic	0.500877	infiltrates	0.428404	axial	0.554040	gallstone	0.606063
incidental	0.500760	hyperinflated	0.413326	spgr	0.520954	steatosis	0.601081
small	0.487211	cardiomegaly	0.410785	weighted	0.502047	dome	0.594812
cystic	0.477632	hyperlucent	0.400836	technique	0.487768	portal	0.570008
pole	0.471933	pectus	0.396142	astrocytoma	0.480527	ascites	0.551869
multiseptated	0.469851	great	0.395712	gbm	0.476956	hepatosplenomegaly	0.540501
polyps	0.464380	ectatic	0.394560	gradient	0.476593	hepatic	0.537453
exophytic	0.459088	shifted	0.389205	oligodendroglioma	0.465892	cirrhosis	0.530389
hyperdense	0.457558	ray	0.389091	postcontrast	0.463686	fatty	0.522134
mucous	0.448427	infiltrate	0.387224	3d	0.458123	kidneys	0.515252

Figure 10: Word-to-vector models trained on a collection of biomedical research articles (from OpenI ope) and radiology reports, and radiology reports only. Search words (with quotes) and their closest words in vector-space cosine similarity (higher the better) are listed in a descending order.

A skip-gram model of Mikolov et al. (2013a,b) is employed with the mapping vector dimension of $\mathbb{R}^{256\times1}$ per word, trained using the *hierarchical softmax* cost function, the sliding-window size of 10 and frequent words sub-sampled in the frequency of 0.01. It is found that combining an additional, more diverse set of related documents such as OpenI biomedical research articles, is helpful for the model to learn a better vector representation while keeping all the hyper-parameters the same. Similar findings on unsupervised feature learning models, that robust features can be learned from a slightly noisy and diverse set of input, were reported by Vincent et al. (2010, 2008); Shin et al. (2013). Some examples of query words and their corresponding closest words with respect to cosine similarity for the word-to-vector models (Mikolov et al., 2013c), which are trained on radiology reports only (total of ∼1 billion words) and with additional OpenI articles (total of 1.2 billion words), are shown in Figure 10.

#words/sentence	mean	median	std	max	min
reports-wide	11.70	9	8.97	1014	1
image references	23.22	19	16.99	221	4
image references, no stopwords no digits	13.46	11	9.94	143	2
image references, disease terms only	5.17	4	2.52	25	1

Table 5: Some statistics about number of words per sentence—across the radiology reports (reports-wide), across the sentences identifying the key images and its two adjacent ones (image references) and these not counting stop-words and digits as well as counting disease related words only.

5.2 Image-to-Description Relation Mining and Matching

The sentence referring to a key image and its adjacent sentences may contain a variety of words, but we are mostly interested in the disease-related terms which are highly correlated to diagnostic semantics. To obtain only the disease-related terms, we exploit the human disease terms and their synonyms from the Disease-Ontology (DO; Schriml et al. (2012)), a collection of 8,707 unique disease-related terms. While the sentences referring to an image and their adjacent sentences have 50.08 words on average, the number of disease-related terms in the three consecutive sentences is 5.17 on average with a standard deviation of 2.5. Therefore, we chose to use bi-grams for the image descriptions, to achieve a good trade-off between the medium level complexity without neglecting too many text-image pairs. Some statistics about the number of words in the documents are shown in Table 5.

Bi-gram disease terms are extracted so that we can train a deep CNN model in Section 5.3 to predict the vector-/word- level image representation of $\mathbb{R}^{256\times2}$. If multiple bi-grams can be extracted per image from the sentence referring the image and the two adjacent ones, the image is trained as many times as the number of different bi-grams with different target vectors ($\mathbb{R}^{256\times2}$). If a disease term cannot form a bi-gram, then the term is ignored, where the process is illustrated in Figure 11. This is a challenging *weakly annotated learning* problem using referring sentences for labels. The bi-grams of DO disease-related terms in the vector representation of $\mathbb{R}^{256\times2}$ are somewhat analogous to the work of Kulkarni et al. (2013) detecting multiple objects of interest and describing their spatial configurations in the image caption. A deep regression CNN model is employed here, to map an image to a continuous output word-vector space from an image. The resulting bi-gram vector can be matched against a reference disease-related vocabulary in the word-vector space using cosine similarity.

5.3 Image-to-Words Deep CNN Regression

To learn the image-to-text representation, we map the images to the vectors of word sequences describing the image. This can be formulated as a regression CNN, replacing the

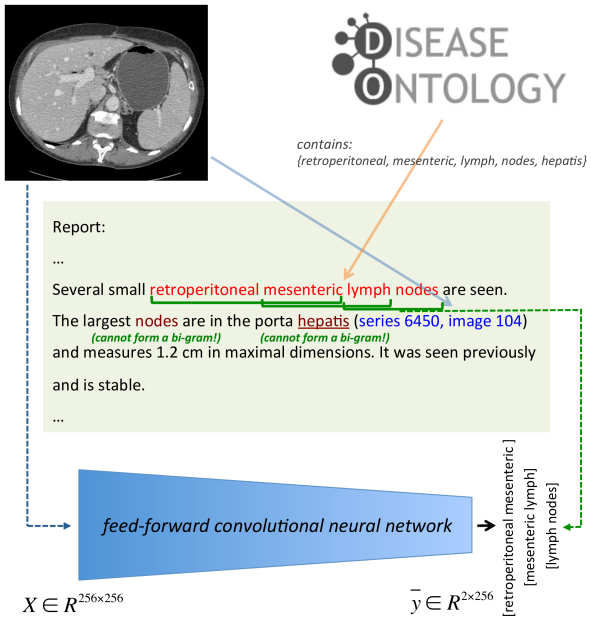


Figure 11: Illustration of how word sequences are learned for an image. Bi-grams are selected from the image's reference sentences containing disease-related terms from the disease ontology (DO; Schriml et al. (2012)). Each bi-gram is converted to a vector of $\mathbf{Z} \in \mathbb{R}^{256\times2}$ to learn from an image. Image input vectors as $\{\mathbf{X} \in \mathbb{R}^{256\times256}\}$ are learned through a CNN by minimizing the cross-entropy loss between the target vector and output vector. The words "nodes" and "hepatis" in the second line are DO terms but are ignored since they can not form a bi-gram. The DO logo is reproduced with permission from `http://disease-ontology.org`.

softmax cost in Section 4 with the cross-entropy cost function for the last output layer of VGG-19 CNN model (Simonyan and Zisserman, 2015):

$$E = -\frac{1}{n}\sum_{n=1}^{N}[g(\mathbf{z})_n \hat{g}(\bar{\mathbf{z}}_n) + (1 - g(\mathbf{z}_n))\log(1 - g(\hat{\mathbf{z}}_n))], \tag{2}$$

where $\mathbf{z}_n$ or $\hat{\mathbf{z}}_n$ is any uni-element of the target word vectors $\mathbf{Z}_n$ or optimized output vectors $\hat{\mathbf{Z}}_n$, $g(x)$ is the sigmoid function ($g(x) = 1/(1 + e^{-x})$), and n is the number of samples in the database.

We adopt the CNN model of Simonyan and Zisserman (2015) for the image-to-text representation since it works consistently better than the other relatively simpler model of Krizhevsky et al. (2012) in our image-to-topic mapping tasks. We fine-tune the parameters of the CNNs for predicting the topic-level labels in Section 4 with the modified cost function, to model the image-to-text representation instead of classifying images into categories. The newly modified output layer has 512 nodes for bi-grams as 256 nodes for each word in a bi-gram.

5.4 Key-Word Generation from Images and Discussion

For any key image in testing, first, we predict its topics at three levels (document-level, document-level sub-topics, sentence-level) using the three deep CNN models of Simonyan and Zisserman (2015) in Section 4. Based on each word's probability of appearing in the LDA document topic, the fifty key-words with highest probability are mapped into the word-to-vector space of multivariate variables in $\mathbb{R}^{256\times1}$ (Section 5.1). Then, the image is mapped to a $\mathbb{R}^{256\times2}$ output vector using the bi-gram CNN model in Section 5.3. Lastly, we match each of the 50 topic key-word vectors of $\mathbb{R}^{256\times1}$ against the first and second half of the $\mathbb{R}^{256\times2}$ output vector using cosine similarity. The closest key-words at three levels of topics (with the highest cosine similarity against either of the bi-gram words) are kept per image.

The rate of predicted disease-related words matching the actual words in the report sentences of test set (recall-at-K, K=1 (R@1 score)) is 0.56. Two examples of key-word generation are shown in Figure 12, with three key-words from three categorization levels per image. We only report R@1 score on disease-related words compared to the previous works of Karpathy et al. (2014); Frome et al. (2013), where they report from R@1 up to R@20 on the entire image caption words (for example, R@1=0.16 on Flickr30K data set by Karpathy et al. (2014)). As we use NLP to parse and extract image-describing sentences from the radiology reports, our ground-truth image-to-text associations are much noisier than the caption data set used by Frome et al. (2013); Karpathy et al. (2014). Also for that reason, our generated image-to-text associations are not as exact as the generated descriptions by Frome et al. (2013); Karpathy et al. (2014).

5.4.1 Discussion

Generating key-words for images by CNN regression shows good feasibility for automated interpretation of patient images. The generated key-words describe what to expect from the given image, although sometimes unrelated words can be generated too. Finding and understanding the relations between the generated words will be the next step to explore, for example via more thorough text mining using sophisticated NLP parsing as by Li et al. (2011) and combining them with the specific frequent disease prediction in the next section.

6. Predicting Presence or Absence of Frequent Disease Types

While the key-words generation in Section 5 can aid the interpretation of a patient scan, the generated key-words, for example, "spine", "lung", are not very specific to a disease in an image. Nonetheless, one of the ultimate goals for large-scale radiology image/text analysis

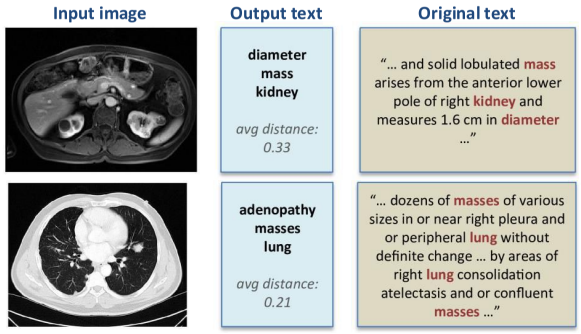


Figure 12: Examples of text key-word generation results, and average cosine distances between the generated words from the disease-related words in the original texts. The word "diameter" appears in the original radiology report of the first image, but not much can be derived by the word only. The rate of predicted disease-related words matching the actual words in the report sentences (recall-at-K, K=1 (R@1 score)) on test set is 0.56.

would be to automatically diagnose disease from a patient scan. In order to achieve the goal of automated disease detection, we add an additional pipeline of mining disease words rather than disease-related words using radiology semantics and predicting these in an image using CNNs with softmax cost-function.

6.1 Mining Presence/Absence of Frequent Disease Terms

The disease names in Disease Ontology (DO) contain not only disease terms but also non-disease terms describing a disease. Some examples of disease names in DO containing non-disease terms are "occlusion of gallbladder" (DOID: 9714), "acute diarrhea" (DOID: 0050140), "strawberry gallbladder" (DOID: 10254), and "exocrine pancreatic insufficiency" (DOID: 13316). Nonetheless, it is rare that "occlusion of gallbladder" or "exocrine pancreatic insufficiency" is described in radiology reports exactly that way, making it difficult to mine specific disease terms with presence or absence.

The Unified Medical Language System (UMLS) of Lindberg et al. (1993); Humphreys et al. (1998) integrates and distributes key terminology, classification and coding standards, and associated resources to promote the creation of more effective and inter-operable biomedical information systems and services, including electronic health records. It is a compendium of many controlled vocabularies in the biomedical sciences, created in 1986 and maintained by the National Library of Medicine.

The Metathesaurus (Schuyler et al., 1993) forms the base of the UMLS and comprises over 1 million biomedical concepts and 5 million concept names, where all of them are

collected from over 100 incorporated controlled vocabularies and classification systems. The Metathesaurus is organized by concept, where each concept has specific attributes defining its meaning and is linked to the corresponding concept names. The Metathesaurus has 133 semantic types that provide a consistent categorization of all concepts represented in it. Among the 133 semantic types we chose to focus on "T033: finding" and "T047: disease or syndrome", as they seemed most relevant to be disease specific. Examples of some other semantic types we do not focus on this study are: "T017: anatomical structure", "T074: medical device", and "T184: sign or symptom".

RadLex (Langlotz, 2006) is a unified language to organize and retrieve radiology imaging reports and medical records. While the Metathesaurus has a vast resource of biomedical concepts, we also use RadLex to confine our disease-term-mining more specifically to radiology related terms. The mined words are one-word terms appearing in the "T033: finding" and "T047: disease or syndrome" of the UMLS Metathesaurus appearing also in RadLex (RadLex is not a subset of Metathesaurus).

We are not only interested in disease terms associated with an image, but also whether the disease mentioned is present or absent. After detecting semantic terms of "T033: finding" and "T047: disease or syndrome", we use the assertion/negation detection algorithm of Chapman et al. (2001, 2013) to detect presence and absence of disease terms. The algorithm of Chapman et al. (2001, 2013) locates trigger terms which can indicate a clinical condition as negated or possible and determines which text falls within the scope of the trigger terms. The number of occurrences "T033: finding" and "T047: disease or syndrome" detected as assertion or negations in radiology reports are shown in Figure 13.

While the assertion/negation detection of "T047: disease or syndrome" seemed specific enough, the detection of "T033: finding" was not. For example, it seemed difficult to derive any specific disease information from 43,219 occurrences of possible "unchanged" and 422 occurrences of negated "unchanged". Some other similar examples are: 10,236 occurrences of possible "finding" and 1,129 occurrences of negated "finding"; 3,781 occurrences of possible "t2" (an MRI image modality) and 661 occurrences of negated "t2". We therefore decided to focus on "T047: disease or syndrome" terms only, and further ignored the terms which occurred less than 10 times in the radiology reports. The total number of "T047: disease or syndrome" terms for detecting their presence are 59, and the total number of the terms for detecting their absence are 18.

6.2 Predicting Disease in Images using CNN

Similarly to the object detection task in the ImageNet challenge, we match and detect disease terms found in the sentences of radiology reports referring to an image using a CNN and softmax cost function.

In addition to assigning disease terms to images, we also assign negated disease terms as the absence of the diseases in the images. The total number of labels is 77 (59 present, 18 absent). If more than one disease term is mentioned for an image, we simply assign the terms multiple times for an image. Some statistics on the number of assertion/negation occurrences per image are shown in Table 6.

As we found in Section 4.2 that transfer learning from the most related model is helpful, we fine-tune the image-to-topic CNN model for the disease prediction model. For this task,

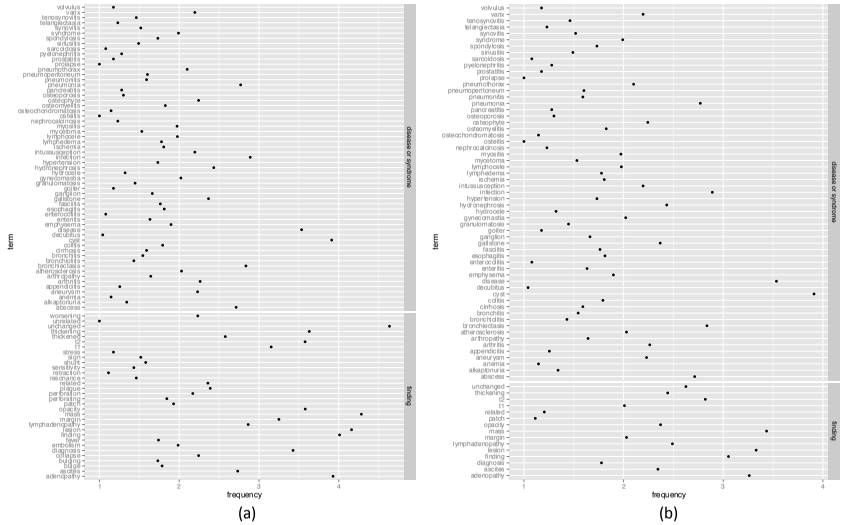


Figure 13: Number of occurrences (frequencies) of semantic terms "T033: finding" and "T047: disease or syndrome" in UMLS Metathesaurus and also appearing in RadLex, detected as (a) assertion and (b) negation in the radiology reports. Frequencies are shown in $\log_{10}$ scale.

# images		per image mean/std		# assertions per image		# negations per image	
total matching	18291	# assertions mean	1.05	1/image	16133	1/image	1581
total not matching	197495	# negations mean	1.05	2/image	613	2/image	84
with assertions	16827	# assertions std	0.23	3/image	81	3/image	0
with negations	1665	# negations std	0.22	4/image	0	4/image	0

Table 6: Some statistics of images-to-disease presence/absence label matching.

we fine-tune from the image to sentence-level-topic (h3) model in Section 4, as the image-to-sentence-level-topic seems to be most closely related to the image-to-disease-specific-terms model. Similarly to Section 4, 85% of image-label pairs are used for training, 5% for validation, and 10% for testing.

6.3 Prediction Result and Discussion

With the CNN trained to model image to disease presence/absence prediction, the top-1 test accuracy achieved is 0.71, and top-5 accuracy is 0.88. We combine this with the

previous image-to-topic mapping and key-word generation (Section 5.4) to generate the final output for comprehensive image interpretation. Some examples of test cases where top-1 probability output matches the originally assigned disease labels are shown in Figure 14. It is noticeable that specific disease words are detected with high probability when there is one disease word per image, but with relatively lower top-1 probability for one disease word and other words within the top-5 probabilities (Figure 14 (b)—" ... infection abscess").

We also observe that automatic label assignment to images can sometimes be challenging. In Figure 14 (d) "cyst" is assigned as the correct label based on the original statement "... possibly due to cyst ...", but it would be unclear whether cyst will be present in the image (and the cyst is not visibly apparent). It applies similarly to Figure 14 (e) where the presence of "osteophyte" is not clear from the referring sentence but is assigned as the correct label (and osteophyte is not visibly apparent on the image). In Figure 14 (f) "no cyst" is labeled and predicted correctly, but it is not obvious what to derive from this prediction that indicates an absence of a disease versus a presence.

Some examples of test cases where top-1 probability does not match the originally assigned labels are shown in Figure 15. Four ((a),(c),(e),(f)) of the six examples, however, contain the originally assigned label in the top-5 probability predictions, which is coherent with the relatively high (88%) top-5 prediction accuracy.

Here again, Figure 15 (a) is automatically labeled as "cyst", but the cyst is not clearly visible on the image where the original statement "... too small to definitely characterize cyst ..." supports this. The example of Figure 15 (b) shows a failed case of assertion/negation algorithm, where "cyst" is detected as negated based on the statement "... small cyst". Nonetheless, true label ("cyst") is detected as its top-1 probability. For Figure 15 (c) "cyst" is predicted where the true label assigned was "abscess"; however cyst and abscess are sometimes visibly similar. Similarly to Figure 14 (d), it is unclear whether we should expect to find emphysema in the image from the statement such as " ... possibly due to emphysema" (and emphysema is not visibly present). Therefore, it would be challenging to correctly interpret such statement for label assignment. Figure 15 (e) shows a disease which can be bronchiectasis, but it is also unclear from the image. However, bronchiectasis is predicted with the second highest probability. Bronchiectasis is visible in Figure 15 (f), and it was predicted with second highest probability too.

6.3.1 Discussion

Automated mining of disease-specific terms with semantics enables us to predict disease more specifically with promising results. However, compared to image-to-topic modeling in Section 4 where image labeling was based on topic modeling and loose coupling of image-to-keyword pairs, by matching the images to more specific disease words we lose about 90% of the images for the analysis due to nonspecific original statements. The proportion of the cases where radiologists indicate a disease as strongly positive or negative is often much less than the cases where they describe a finding rather vaguely. Mining and assigning the semantic label "T033: finding" will yield more images for specific disease-label pairs. However, it is probably less specific to model an image with a generic term as "mass" (which is a more vague indication of a specific disease such as "cyst" or "tumor") and detecting

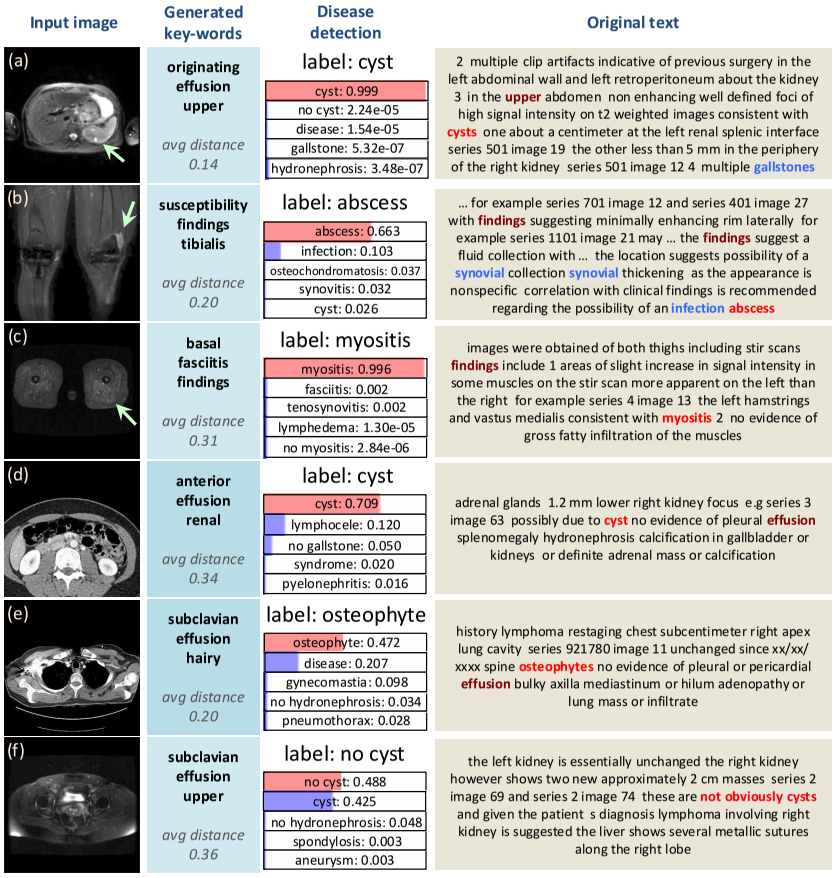


Figure 14: Some examples of final outputs for automated image interpretation, where top-1 probability matches the originally assigned label. Generated key-words appearing in the original text in radiology reports mentioning the image are shown in bold brown, specific disease words assigned as label mentioned in the reports are shown in bold red, and disease words predicted with top-5 probability in the reports are shown in bold blue. The probability assigned to the originally assigned label is shown with a red bar, and the other top-5 probabilities are shown with blue bars. Disease region identified in an image is pointed by arrow.

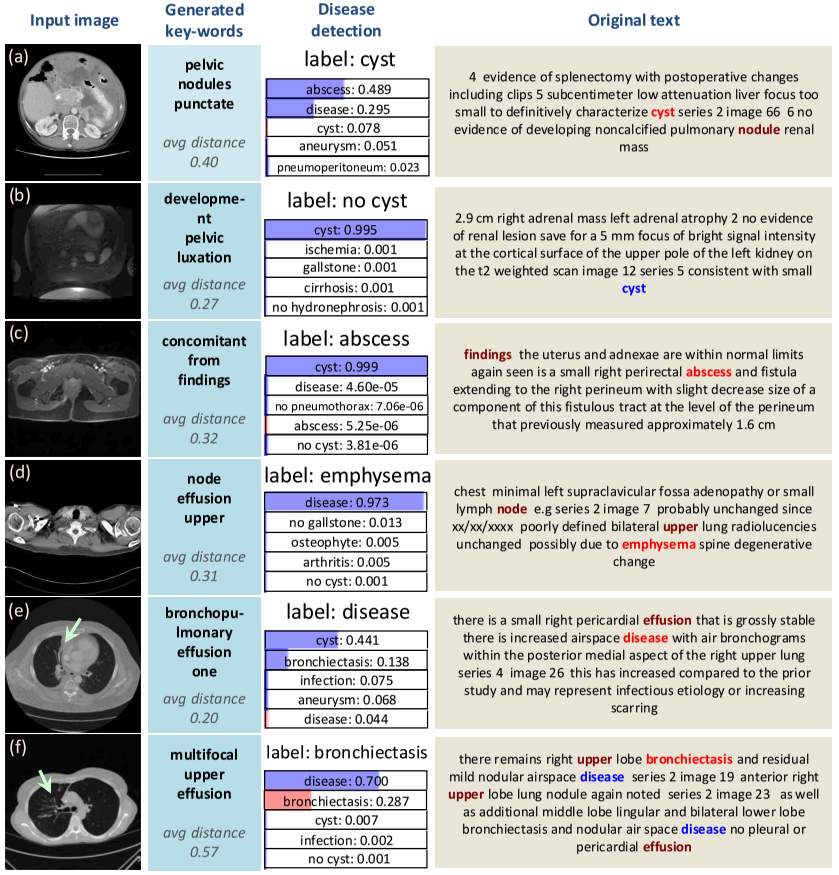


Figure 15: Some examples of final outputs for automated image interpretation where top-1 probability does not match the originally assigned label. One of the top-5 probabilities match the originally assigned labels in the examples of images (a), (c), (d), and (f). None of the top-5 probabilities match the originally assigned labels in the examples of image (b) and (d). However, label assignment of second row example is incorrect, as a failed case of assertion/negation detection algorithm used. Nonetheless, the CNN predicted "true" label correctly ("cyst").

it than modeling and detecting an image with a more specific term as "cyst" (similarly to "finding" or "unchanged").

It is a compromise between whether to go for big data and loose labels or to go for smaller data and more accurate labels. The key-word generation from the rather loose labeling scheme enables us to use most of the available 216K images. While the generated key-words can help understand the contents of the image, sometimes they are not specific and can also be irrelevant. More specific mining and assignment of specific disease labels to images could provide more accurate and precise disease prediction; however, only about 10% of the total images are made available by this scheme. Another alternative is to obtain annotation by radiologists to be even more specific, but the amount of data available will be even smaller due to the time and cost limitations.

Consequently, utilizing bigger data will enable us to make a more generalizable model, but labeling will become more challenging as the amount of data gets bigger and becomes more heterogeneous. The compromise between the amount of data and the quality of labels seems to be a recurring dilemma in the majority of automated mining in big data applications. More advanced NLP techniques and comprehensive analysis of hospital discharge summaries, progress notes, and patient histories might address the need to obtain more specific information relating to an image even when the original image descriptions are not very specific.

7. Conclusion

It has been unclear how to extend the significant success in image classification using deep convolutional neural networks from computer vision to medical imaging. What are the clinically relevant image labels to be defined, how to annotate the huge amount of medical images required by deep learning models, and to what extent and scale the deep CNN architecture is generalizable to medical image analysis are open questions.

In this paper, we present an interleaved text/image deep mining system to extract the semantic interactions of radiology reports and diagnostic key images at a very large, unprecedented scale in the medical domain. Images are classified into hierarchies of topics according to their associated documents, and a neural language model is learned to assign disease terms to predict the image interpretation. However, by generating the "attributes" of patient images, the generated descriptions are not disease-specific, whereas one of the primary goals for medical image analysis is to automatically diagnose diseases. In order to address this issue, we mine and match frequent disease types using disease ontology and semantics, and demonstrate prediction of the presence/absence of disease with probability outputs. Yet, only about 10% of the entire data set could be used for this study due to the challenge of more precisely matching the disease words with semantics. This raises interesting questions regarding the trade-offs in designing a machine learning system analyzing large medical data.

To the best of our knowledge, this is the first study performing a large-scale image/text analysis on a hospital picture archiving and communication system database. Our database is the largest one ever reported and is highly representative of the huge collection of radiology diagnostic semantics over the last decade. Exploring effective deep learning models on this database opens new ways to parse and understand large-scale radiology image data sets.

We hope that this study will inspire and encourage other institutions in mining other large unannotated clinical databases, to achieve the goal of establishing a central training resource and performance benchmark for large-scale medical image research, similar to the ImageNet of Deng et al. (2009) for computer vision.

Acknowledgments

This work was supported in part by the Intramural Research Program of the National Institutes of Health Clinical Center, and in part by a grant from the KRIBB Research Initiative Program (Korean Biomedical Scientist Fellowship Program), Korea Research Institute of Bioscience and Biotechnology, Republic of Korea. This study utilized the high-performance computational capabilities of the Biowulf Linux cluster at the National Institutes of Health, Bethesda, MD (http://biowulf.nih.gov). We thank NVIDIA for the K40 GPU donation.

References

Openi - an open access biomedical image search engine. `http://openi.nlm.nih.gov`. Lister Hill National Center for Biomedical Communications, U.S. National Library of Medicine.

Kobus Barnard, Pinar Duygulu, David Forsyth, Nando de Freitas, David M Blei, and Michael I Jordan. Matching words and pictures. *Journal of machine learning research*, 3 (Feb):1107–1135, 2003.

Tamara L Berg, Alexander C Berg, and Jonathan Shih. Automatic attribute discovery and characterization from noisy web data. In *European Conference on Computer Vision*, pages 663–676. Springer, 2010.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python*. O'Reilly Media, Inc., 2009.

David M Blei and Michael I Jordan. Modeling annotated data. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 127–134. ACM, 2003.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3:993–1022, 2003.

Luke Carrivick, Sanjay Prabhu, Paul Goddard, and Jonathan Rossiter. Unsupervised learning in radiology using novel latent variable models. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 854–859. IEEE, 2005.

Wendy W Chapman, Will Bridewell, Paul Hanbury, Gregory F Cooper, and Bruce G Buchanan. A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of biomedical informatics*, 34(5):301–310, 2001.

Wendy W Chapman, Dieter Hilert, Sumithra Velupillai, Maria Kvist, Maria Skeppstedt, Brian E Chapman, Michael Conway, Melissa Tharp, Danielle L Mowery, and Louise

Deleger. Extending the negex lexicon for multiple languages. *Studies in health technology and informatics*, 192:677, 2013.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

Thomas Deselaers and Hermann Ney. Deformations, patches, and discriminative models for automatic annotation of medical radiographs. *Pattern Recognition Letters*, 29(15): 2003–2010, 2008.

Chris Ding, Tao Li, and Wei Peng. Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence chi-square statistic, and a hybrid method. In *Proceedings of the national conference on artificial intelligence*, volume 21, page 342. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013.

Eric Gaussier and Cyril Goutte. Relation between plsa and nmf and implications. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 601–602. ACM, 2005.

Mark Girolami and Ata Kabán. On an equivalence between plsi and lda. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 433–434. ACM, 2003.

Ashish Gupta, Murat Ayhan, and Anthony Maida. Natural image bases to represent neuroimaging data. In *Proceedings of the International Conference on Machine Learning*, 2013.

Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014.

Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, pages 853–899, 2013.

Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

Betsy L Humphreys, Donald AB Lindberg, Harold M Schoolman, and G Octo Barnett. The unified medical language system an informatics research collaboration. *Journal of the American Medical Informatics Association*, 5(1):1–11, 1998.

Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. In *European conference on computer vision*, pages 512–528. Springer, 2014.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

Andrej Karpathy, Armand Joulin, and Fei Fei F Li. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in Neural Information Processing Systems*, pages 1889–1897, 2014.

M Hadi Kiapour, Kota Yamaguchi, Alexander C Berg, and Tamara L Berg. Hipster wars: Discovering elements of fashion styles. In *European conference on computer vision*, pages 472–488. Springer, 2014.

Gunhee Kim, Leonid Sigal, and Eric P Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 4225–4232. IEEE, 2014.

Gunhee Kim, Seungwhan Moon, and Leonid Sigal. Joint photo stream and blog post summarization and exploration. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015a.

Gunhee Kim, Seungwhan Moon, and Leonid Sigal. Ranking and retrieval of image sequences from multiple paragraph queries. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015b.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

G. Kulkarni, V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, A. Berg, and T. Berg. Babytalk: Understanding and generating simple image descriptions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(12):2891–2903, 2013.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2014.

Curtis P Langlotz. Radlex: A new method for indexing online educational materials 1. *Radiographics*, 26(6):1595–1597, 2006.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

S. Li, G. Kulkarni, T. Berg, A. Berg, and Y. Choi. Composing simple image descriptions using web-scale n-grams. In *ACM CoNLL*, pages 220–228, 2011.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

Donald A Lindberg, Betsy L Humphreys, and Alexa T McCray. The unified medical language system. *Methods of information in Medicine*, 32(4):281–291, 1993.

Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013b.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751. Citeseer, 2013c.

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38 (11):39–41, 1995.

Vicente Ordonez and Tamara L Berg. Learning high-level judgments of urban perception. In *European Conference on Computer Vision*, pages 494–510. Springer, 2014.

Vicente Ordonez, Girish Kulkarni, and Tamara L Berg. Im2text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems*, pages 1143–1151, 2011.

Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 139–147. Association for Computational Linguistics, 2010.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3): 211–252, 2015.

Walter J Scheirer, Neeraj Kumar, Peter N Belhumeur, and Terrance E Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *Computer Vision and Pattern Recognition, 2012 IEEE Conference on*, pages 2933–2940. IEEE, 2012.

Lynn Marie Schriml, Cesar Arze, Suvarna Nadendla, Yu-Wei Wayne Chang, Mark Mazaitis, Victor Felix, Gang Feng, and Warren Alden Kibbe. Disease ontology: a backbone for disease semantic integration. *Nucleic acids research*, 40(D1):D940–D946, 2012.

Peri L Schuyler, William T Hole, Mark S Tuttle, and David D Sherertz. The umls metathesaurus: representing different views of biomedical concepts. *Bulletin of the Medical Library Association*, 81(2):217, 1993.

Hoo-Chang Shin, Matthew R Orton, David J Collins, Simon J Doran, and Martin O Leach. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1930–1943, 2013.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *Advances in Neural Information Processing Systems*, pages 935–943, 2013.

Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. Exploring topic coherence over many models and many topics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 952–961. Association for Computational Linguistics, 2012.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *Proceedings of the 31st International Conference on Machine Learning*, 2015.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Journal of Machine Learning Research 17 (2016) 1-30 Submitted 5/15; Revised 2/16; Published 7/16

Distribution-Matching Embedding for Visual Domain Adaptation

Mahsa Baktashmotlagh M.BAKTASHMOTLAGH@QUT.EDU.AU
Queensland University of Technology
Brisbane, Australia

Mehrtash Harandi MEHRTASH.HARANDI@NICTA.COM.AU
Australian National University & NICTA[*]
Canberra, Australia

Mathieu Salzmann MATHIEU.SALZMANN@EPFL.CH
CVLab, EPFL[*]
Lausanne, Switzerland

Editor: Urun Dogan, Marius Kloft, Francesco Orabona, and Tatiana Tommasi

Abstract

Domain-invariant representations are key to addressing the domain shift problem where the training and test examples follow different distributions. Existing techniques that have attempted to match the distributions of the source and target domains typically compare these distributions in the original feature space. This space, however, may not be directly suitable for such a comparison, since some of the features may have been distorted by the domain shift, or may be domain specific. In this paper, we introduce a Distribution-Matching Embedding approach: An unsupervised domain adaptation method that overcomes this issue by mapping the data to a latent space where the distance between the empirical distributions of the source and target examples is minimized. In other words, we seek to extract the information that is invariant across the source and target data. In particular, we study two different distances to compare the source and target distributions: the Maximum Mean Discrepancy and the Hellinger distance. Furthermore, we show that our approach allows us to learn either a linear embedding, or a nonlinear one. We demonstrate the benefits of our approach on the tasks of visual object recognition, text categorization, and WiFi localization.

Keywords: Domain Adaptation, Maximum Mean Discrepancy, Hellinger Distance, Distribution Matching, Domain Invariant Representations

1. Introduction

As evidenced by the recent surge of interest in domain adaptation (Saenko et al., 2010; Jain and Learned-Miller, 2011; Gong et al., 2012; Gopalan et al., 2011), domain shift is a fundamental problem for visual recognition. This problem typically occurs when the training and test images are acquired with different cameras, or in very different conditions (e.g., commercial website versus home environment, images taken under different illuminations). As a consequence, the training (source) and test (target) samples follow different distributions. As demonstrated in, e.g., (Saenko et al., 2010; Gopalan et al., 2011; Gong et al., 2012, 2013), failing to model this distribution shift in the hope that the image features will be robust enough often yields poor recognition accuracy. While labeling sufficiently many images from the target domain to train a discriminative classifier specific to this domain could alleviate this problem, it typically is prohibitively time-consuming and impractical in realistic scenarios. Domain adaptation therefore seeks to prevent this by explicitly modeling the domain shift.

Existing domain adaptation methods can be divided into two categories: Semi-supervised approaches that assume that a small number of labeled examples from the target domain are available during training, and unsupervised approaches that do not require any labels from the target domain. In the former category, modifications of existing classifiers have been proposed to exploit the availability of labeled and unlabeled data from the target domain (Daumé III and Marcu, 2006; Duan et al., 2009b; Bergamo and Torresani, 2010; Tommasi and Caputo, 2013). Co-regularization and adaptive regularization of similar classifiers was also introduced to utilize unlabeled target data during training (Daumé III et al., 2010; Rückert and Kloft, 2011). Multi-Model knowledge transfer was proposed to select and weigh prior knowledge coming from different categories (Jie et al., 2011; Tommasi et al., 2014, 2010). For visual recognition, metric learning (Saenko et al., 2010) and transformation learning (Kulis et al., 2011) were shown to be effective at making use of the labeled target examples. Furthermore, semi-supervised methods have also been employed to tackle the case where multiple source domains are available (Duan et al., 2009a; Hoffman et al., 2012). While semi-supervised methods are often effective, in many applications, labeled target examples are not available and cannot easily be acquired.

By contrast, unsupervised domain adaptation approaches rely on purely unsupervised target data (Xing et al., 2007; Bruzzone and Marconcini, 2010; Chen et al., 2011; Kuzborskij and Orabona, 2013). In particular, two types of methods have proven quite successful at the task of visual object recognition: Subspace-based approaches and sample re-weighting techniques. Subspace-based approaches (Blitzer et al., 2011; Gong et al., 2012; Gopalan et al., 2011) typically model each domain with a subspace, and attempt to relate the source and target representations via intermediate subspaces. While these methods have proven effective in practice, they suffer from the fact that they do not explicitly try to match the probability distributions of the source and target data. Therefore, they may easily yield sub-optimal representations for classification. By contrast, sample selection, or re-weighting, approaches (Huang et al., 2006; Gretton et al., 2009; Gong et al., 2013) explicitly attempt to match the source and target distributions by finding the most appropriate source examples for the target data. However, these methods fail to account for the fact that the image features themselves may have been distorted by the domain shift, and that some of these features may be specific to one domain and thus irrelevant for classification in the other one.

In light of the above discussion, we propose to tackle the problem of domain shift by discovering the information that is invariant across the source and target domains. To this end, we introduce a Distribution-Matching Embedding (DME) approach, which aims to learn a latent space where the source and target distributions are similar. Learning such a projection allows us to account for the potential distortions induced by the domain shift, as well as for the presence of domain-specific image features. Furthermore, since the distributions of the source and target data in the latent space are similar, we expect a classifier trained on the source samples to perform well on the target domain.

More specifically, here, we study two different distances to compare the source and target distributions in the latent space. First, we make use of the Maximum Mean Discrepancy (MMD) (Gretton et al., 2012a), which compares the means of two empirical distributions in a reproducing kernel Hilbert space. While the MMD is endowed with nice properties (Gretton et al., 2012a), it does not truly consider the geometry of the space of probability distributions. From information geometry,

*. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the ARC through the ICT Centre of Excellence program. This research was performed while M. Salzmann was affiliated with the Australian National University and NICTA.

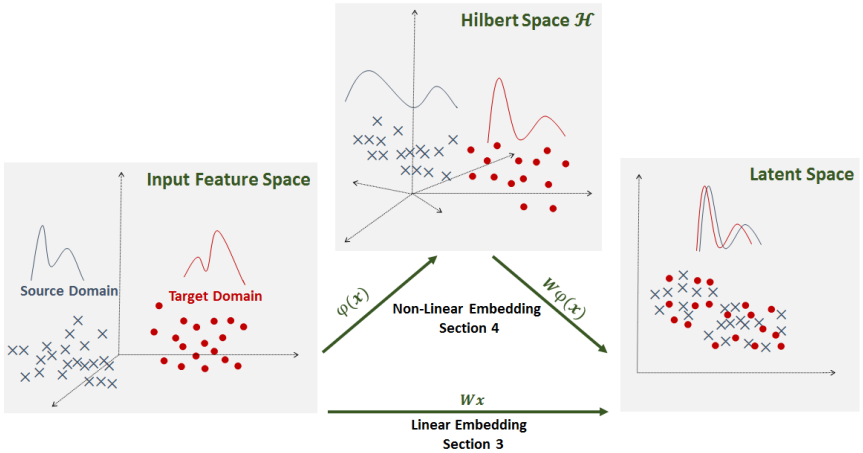


Figure 1: **Illustration of our approach.** Our goal is to learn a latent space, via either a linear mapping or a nonlinear one, such that the source and target distributions in this latent space are as similar as possible.

we know that probability distributions lie on a Riemannian manifold named the statistical manifold. In computer vision, it has been consistently demonstrated that exploiting the Riemannian metric of the manifold to compare manifold-values entities, such as covariance descriptors (Tuzel et al., 2008), linear subspaces (Harandi et al., 2013), or rotation matrices (Hartley et al., 2013), was beneficial for the task at hand. Therefore, we follow a similar intuition here and make use of the Riemannian metric on the statistical manifold as a second distance measure to compare the source and target distributions. Since the true Riemmannian metric, i.e., the Fisher-Rao metric, is difficult to use with general, non-parametric distributions, such as those obtained by kernel density estimation, we propose to rely on the Hellinger distance, which we show to be closely related to the Fisher-Rao Riemannian metric.

Given these two distances, we first introduce algorithms to learn a linear mapping to a low-dimensional latent space where the source and target distributions are similar. By exploiting the Riesz representer theorem (Schölkopf and Smola, 2002), we then show that, for both distances, our approach also allows us to learn a nonlinear embedding to a distribution-matching latent space. In the linear and the nonlinear scenarios, learning our Distribution-Matching Embeddings can then be formulated as an optimization problem on a Grassmann manifold. This lets us utilize Grassmannian geometry to effectively obtain our latent representations. Fig. 1 illustrates our approach, both for linear and nonlinear mappings. In essence, our approach consists of two main components: (i) a mapping to a latent space, which in the nonlinear case is achieved via a mapping to a high-dimensional Hilbert space; and (ii) a distance to compare the source and target distributions in the latent space. While, here, we rely on either the MMD or the Hellinger distance, our approach is general and could potentially be extended to other distance measures.

In short, we introduce the idea of finding a distribution-matching representation of the source and target data, and propose several effective algorithms to learn such a representation. We demonstrate the benefits of our approach on the tasks of visual object recognition, text categorization and Wifi localization using standard domain adaptation data sets. This article is an extended version of our ICCV 2013 (Baktashmotlagh et al., 2013) and CVPR 2014 (Baktashmotlagh et al., 2014) papers. Compared to the conference papers, it contains additional details about the linear formulations, as well as new results. Furthermore, it introduces the nonlinear formulations of our previous methods.

2. Preliminaries

In this section, we provide the background theory and groundwork for the techniques described in the following sections. In particular, we discuss the idea of Maximum Mean Discrepancy and review the derivation of the Hellinger distance on statistical manifolds, as well as study its relationship with the Fisher-Rao Riemannian metric. We then introduce some notions of Grassmannian geometry, as well as discuss the conjugate gradient (CG) algorithm that will be used in our optimization process.

2.1 Maximum Mean Discrepancy (MMD)

In this work, we are interested in measuring the distance between two probability distributions s and t. Rather than restricting these distributions to take a specific parametric form, we opt for a non-parametric approach to compare s and t. Non-parametric representations are very well-suited to visual data, which typically exhibits complex probability distributions in high-dimensional spaces.

To this end, here, we employ the Maximum Mean Discrepancy (Gretton et al., 2012a). The MMD is an effective non-parametric criterion that compares the distributions of two sets of data by mapping the data to RKHS. Given two distributions s and t, the MMD between s and t is defined as

$$D'(\mathfrak{F}, s, t) = \sup_{f \in \mathfrak{F}} \left(E_{\boldsymbol{x}^s \sim s}[f(\boldsymbol{x}^s)] - E_{\boldsymbol{x}^t \sim t}[f(\boldsymbol{x}^t)] \right) \ ,$$

where $E_{\boldsymbol{x} \sim s}[\cdot]$ is the expectation under distribution s. By defining $\mathfrak{F}$ as the set of functions in the unit ball in a universal RKHS $\mathcal{H}$, it was shown that $D'(\mathfrak{F}, s, t) = 0$ if and only if $s = t$ (Gretton et al., 2012a).

Let $\boldsymbol{X_s} = \{\boldsymbol{x}_1^s, \cdots, \boldsymbol{x}_n^s\}$ and $\boldsymbol{X_t} = \{\boldsymbol{x}_1^t, \cdots, \boldsymbol{x}_m^t\}$ be two sets of observations drawn i.i.d. from s and t, respectively. An empirical estimate of the MMD can be computed as

$$\begin{aligned} \hat{D}_M(\tilde{\boldsymbol{X}}_s, \tilde{\boldsymbol{X}}_t) &= \left\| \frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{x}_i^s) - \frac{1}{m} \sum_{j=1}^{m} \phi(\boldsymbol{x}_j^t) \right\|_{\mathcal{H}} \\ &= \left(\sum_{i,j=1}^{n} \frac{k(\boldsymbol{x}_i^s, \boldsymbol{x}_j^s)}{n^2} + \sum_{i,j=1}^{m} \frac{k(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)}{m^2} - 2 \sum_{i,j=1}^{n,m} \frac{k(\boldsymbol{x}_i^s, \boldsymbol{x}_j^t)}{nm} \right)^{\frac{1}{2}} , \end{aligned}$$

where $\phi(\cdot)$ is the mapping to the RKHS $\mathcal{H}$, and $k(\cdot, \cdot) = \langle \phi(\cdot), \phi(\cdot) \rangle$ is the universal kernel associated with this mapping. In short, the MMD between the distributions of two sets of observations is equivalent to the distance between the sample means in a high-dimensional feature space.

2.2 Hellinger Distance on Statistical Manifolds

In this section, we review some concepts of Riemannian geometry on statistical manifolds. In particular, we focus on the derivation of the Hellinger distance, which will be used in our algorithms.

Statistical manifolds are Riemannian manifolds whose elements are probability distributions. Loosely speaking, given a non-empty set $\mathcal{X}$ and a family of probability density functions $p(\boldsymbol{x}|\boldsymbol{\theta})$ parametrized by $\boldsymbol{\theta}$ on $\mathcal{X}$, the space $\mathcal{M} = \{p(\boldsymbol{x}|\boldsymbol{\theta})|\boldsymbol{\theta} \in \mathbb{R}^d\}$ forms a Riemannian manifold. The Fisher-Rao Riemannian metric on $\mathcal{M}$ is a function of $\boldsymbol{\theta}$ and induces geodesics, i.e., curves with minimum length on $\mathcal{M}$.

While the Fisher-Rao metric can be computed for specific parametric distributions, such as a Gaussian or a Gaussian mixture (Peter and Rangarajan, 2006), for other parametric forms, it does not even have a closed form solution. More importantly, in general, the parametrization of the PDFs of the data at hand is unknown, and choosing a specific distribution may not reflect the reality. Unfortunately, the Fisher-Rao metric is ill-suited to handle non-parametric distributions, which are of interest for our purpose. Therefore, several studies have opted for approximations of the Fisher-Rao metric. For instance, (Srivastava et al., 2007) proposed to map distributions to the hyper-sphere and use geodesics on this different type of manifold. By contrast, here, we make use of another class of approximations relying on f-divergences, which can be expressed as

$$D_f(s\|t) = \int f(\frac{s(\boldsymbol{x})}{t(\boldsymbol{x})})t(\boldsymbol{x})d\boldsymbol{x} \ .$$

The (squared) Hellinger distance is a special case of f-divergences, obtained by taking $f(y) = (\sqrt{y}-1)^2$. The (squared) Hellinger distance can thus be written as

$$D_H^2(s\|t) = \int \left(\sqrt{s(\boldsymbol{x})} - \sqrt{t(\boldsymbol{x})}\right)^2 d\boldsymbol{x} \ , \tag{1}$$

which is symmetric, satisfies the triangle inequality and is bounded from above by 2.

More importantly, in the following theorem, we show an interesting relationship between the Hellinger distance and the Fisher-Rao Riemannian metric.

Theorem 1 *The length of any curve γ is the same under the Fisher-Rao metric D_{FR} and the Hellinger distance D_H up to a scale of 2.*

Proof We start with the definition of intrinsic metric and curve length. Without any assumption on differentiability, let (M, d) be a metric space. A curve in M is a continuous function $\gamma : [0, 1] \to M$ and joins the starting point $\gamma(0) = p$ to the end point $\gamma(1) = q$. Our proof then relies on two theorems from (Hartley et al., 2013) stated below. To state and exploit these two theorems, we first need the following two definitions coming from (Hartley et al., 2013):

Definition 2 *The length of a curve γ is the supremum of $\ell(\gamma; \alpha_i)$ over all possible partitions α_i such that $0 = \alpha_0 < \alpha_1 < ... < \alpha_n = 1$, where $\ell(\gamma; \alpha_i) = \sum_i d(\gamma(\alpha_i), \gamma(\alpha_{i-1}))$.*

Definition 3 *The intrinsic metric $\hat{\delta}(x, y)$ between two points x and y on a metric space M is defined as the infimum of the lengths of all paths from x to y.*

Theorem 4 *(Hartley et al., 2013) If the intrinsic metrics induced by two metrics d_1 and d_2 are identical to scale ξ, then the length of any given curve is the same under both metrics up to ξ.*

Proof We refer the reader to (Hartley et al., 2013) for the proof of this theorem. ■

Theorem 5 *(Hartley et al., 2013) If $d_1(s, t)$ and $d_2(s, t)$ are two metrics defined on a space M such that*

$$\lim_{d_1(s,t)\to 0} \frac{d_2(s,t)}{d_1(s,t)} = 1 \tag{2}$$

uniformly (with respect to s and t), then their intrinsic metrics are identical.

Proof We refer the reader to (Hartley et al., 2013) for the proof of this theorem. ■

In (Kass and Vos, 2011), it was shown that $\lim_{s\to t} D_H(s\|t) = 0.5 D_{FR}(s\|t)$. The asymptotic behavior of the Hellinger distance and the Fisher-Rao metric can be expressed as $D_H(s, t) = 0.5 * D_{FR}(s, t) + O(D_{FR}(s, t)^3)$ as $s \to t$. This guarantees uniform convergence since the higher order terms are bounded and vanish rapidly independently of the path between s and t. It therefore directly follows from Theorems 5 and 4 that the length of a curve under D_H and D_{FR} is the same up to a scale of 2, which concludes the proof. ■

2.2.1 EMPIRICAL ESTIMATE OF THE HELLINGER DISTANCE

In a practical scenario, our goal is to compute the Hellinger distance between the distributions s and t when discrete observations are provided. In other words, we are interested in estimating Eq. 1 given n samples $\{\boldsymbol{x}_i^s\}$ drawn from s and m samples $\{\boldsymbol{x}_i^t\}$ drawn from t.

To have a symmetric and bounded estimate of the Hellinger distance with respect to a single density, we begin by defining $T(x) = \frac{s(x)}{s(x)+t(x)}$. The Hellinger distance can then be defined in terms of $T(x)$ as

$$\begin{aligned} D_H = & \int (\sqrt{s(x)} - \sqrt{t(x)})^2 dx \\ = & \int \left(\sqrt{\frac{s(x)}{s(x)+t(x)}} - \sqrt{\frac{t(x)}{s(x)+t(x)}}\right)^2 (s(x)+t(x))\, dx \\ = & \int (\sqrt{T(x)} - \sqrt{1-T(x)})^2 s(x) dx + \int (\sqrt{T(x)} - \sqrt{1-T(x)})^2 t(x) dx \ . \end{aligned} \tag{3}$$

Since the two terms in Eq. 3 are expectations, and following the strong law of large numbers, given our two sets of samples $\{\boldsymbol{x}_i^s\}$ and $\{\boldsymbol{x}_i^t\}$, an empirical estimate of the Hellinger distance can be obtained as (Carter, 2009)

$$\hat{D}_H^2 = \frac{1}{n}\sum_{i=1}^{n} \left(\sqrt{\hat{T}(\boldsymbol{x}_i^s)} - \sqrt{1-\hat{T}(\boldsymbol{x}_i^s)}\right)^2 + \frac{1}{m}\sum_{i=1}^{m} \left(\sqrt{\hat{T}(\boldsymbol{x}_i^t)} - \sqrt{1-\hat{T}(\boldsymbol{x}_i^t)}\right)^2 \ , \tag{4}$$

where $\hat{T}(\boldsymbol{x}) = \hat{s}(\boldsymbol{x})/\big(\hat{s}(\boldsymbol{x}) + \hat{t}(\boldsymbol{x})\big)$, with $\hat{s}(\boldsymbol{x})$ and $\hat{t}(\boldsymbol{x})$ the empirical estimates of $s(\boldsymbol{x})$ and $t(\boldsymbol{x})$, respectively. Importantly, this numerical approximation respects some of the properties of the true Hellinger distance (Carter, 2009). In particular, it is symmetric and bounded from above by 2.

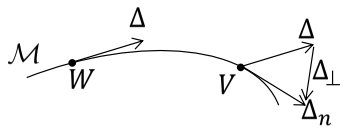


Figure 2: Parallel transport of a tangent vector Δ from a point $\boldsymbol{W}$ to point $\boldsymbol{V}$ on the manifold.

2.3 Grassmann Manifolds

The Grassmann manifold $\mathcal{G}(d, D)$ consists of the set of all linear d-dimensional subspaces of $\mathbb{R}^D$. In particular, for $\boldsymbol{W} \in \mathcal{G}(d, D)$, this lets us handle constraints of the form $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$. As will be shown in Section 3, our mappings to latent space involve nonlinear optimization on the Grassmann manifold. Below, we therefore review some useful notions of differential geometry.

In differential geometry, the shortest path between two points on a manifold is a curve called a *geodesic*. The *tangent space* at a point on a manifold is a vector space that consists of the tangent vectors of all possible curves passing through this point. *Parallel transport* is the action of transferring a tangent vector between two points on a manifold. As illustrated in Fig. 2, unlike in flat spaces, this cannot be achieved by simple translation, but requires subtracting a normal component at the end point (Edelman et al., 1998).

On a Grassmann manifold, the above-mentioned operations have efficient numerical forms and can thus be used to perform optimization on the manifold. In particular, we make use of a conjugate gradient (CG) algorithm on the Grassmann manifold (Edelman et al., 1998). CG techniques are popular nonlinear optimization methods with fast convergence rates. These methods iteratively optimize the objective function in linearly independent directions called conjugate directions (Ruszczynski, 2006). CG on a Grassmann manifold can be summarized by the following steps:

(i) Compute the gradient $\nabla f_{\boldsymbol{W}}$ of the objective function f on the manifold at the current estimate $\boldsymbol{W}$ as

$$\nabla f_{\boldsymbol{W}} = \partial f_{\boldsymbol{W}} - \boldsymbol{W}\boldsymbol{W}^T\partial f_{\boldsymbol{W}} \ , \tag{5}$$

with $\partial f_{\boldsymbol{W}}$ the matrix of usual partial derivatives.

(ii) Determine the search direction $\boldsymbol{H}$ by parallel transporting the previous search direction and combining it with $\nabla f_{\boldsymbol{W}}$.

(iii) Perform a line search along the geodesic at $\boldsymbol{W}$ in the direction $\boldsymbol{H}$.

These steps are repeated until convergence to a local minimum, or until a maximum number of iterations is reached.

Note that, while we rely on a conjugate gradient method, other optimization strategies have been studied on Grassmann manifolds, such as (i) Stochastic-gradient flow, where a stochastic component is added to the gradient to construct a stochastic gradient process such that the solution converges to a global optimum in the limit; (ii) Acceptance-rejection methods, where the stochastic gradient part provides candidates to update the estimate, that are accepted/rejected according to a probability density function (Metropolis-Hastings type acceptance-rejection step); and (iii) Simulated annealing, where instead of sampling from a probability distribution, an annealing procedure is applied to find the optimal points of the function f (Srivastava and Liu, 2005). A complete study of these Grassmannian optimization strategies goes beyond the scope of this paper.

3. Distribution-Matching Embedding (DME)

In this section, we introduce our approach to unsupervised domain adaptation, which relies on mapping the data to a low-dimensional latent space such that the distance between the source and target distributions is minimized. Intuitively, with such a latent representation, a classifier trained on the source domain should perform equally well on the target domain. In particular, here, we make use of a linear mapping of the form

$$\boldsymbol{y} = \boldsymbol{W}^T\boldsymbol{x} \ , \tag{6}$$

where $\boldsymbol{x} \in \mathbb{R}^D$ is the original data (e.g., image features), $\boldsymbol{y} \in \mathbb{R}^d$ is the resulting low-dimensional representation, and $\boldsymbol{W} \in \mathbb{R}^{D\times d}$ is the parameter matrix that we seek to learn. Furthermore, we enforce orthogonality constraints on $\boldsymbol{W}$, such that

$$\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I} \ . \tag{7}$$

These constraints typically avoid degeneracies, such as having all samples collapsing to the origin, and have proven effective in many dimensionality reduction methods, such as Principal Component Analysis (PCA) and Canonical Correlation Analysis (CCA).

In the remainder of this section, we denote by $s(\boldsymbol{x})$ and $t(\boldsymbol{x})$ the probability density functions of the source samples $\boldsymbol{X_s} = [\boldsymbol{x}_1^s, \cdots, \boldsymbol{x}_n^s]$ and target samples $\boldsymbol{X_t} = [\boldsymbol{x}_1^t, \cdots, \boldsymbol{x}_m^t]$, respectively, where each $\boldsymbol{x}_i^* \in \mathbb{R}^D$. We first derive our MMD-based algorithm (DME-MMD), and then discuss our formulation based on the Hellinger distance (DME-H).

3.1 DME with the MMD (DME-MMD)

To derive our first approach to learning a distribution-matching representation, we make use of the MMD to measure the distance between the source and target distributions. Following the derivations provided in Section 2, and by making use of the linear mapping defined in Eq. 6, the MMD in the latent space can be expressed as

$$\hat{D}_M(\boldsymbol{W}^T\boldsymbol{X_s}, \boldsymbol{W}^T\boldsymbol{X_t}) = \left\| \frac{1}{n}\sum_{i=1}^{n}\phi(\boldsymbol{W}^T\boldsymbol{x}_i^s) - \frac{1}{m}\sum_{j=1}^{m}\phi(\boldsymbol{W}^T\boldsymbol{x}_j^t) \right\|_{\mathcal{H}} , \tag{8}$$

with $\phi(\cdot)$ the mapping from $\mathbb{R}^D$ to the high-dimensional RKHS $\mathcal{H}$. Note that, here, $\boldsymbol{W}$ appears inside $\phi(\cdot)$ in order to measure the MMD of the projected samples.

Using the MMD, in conjunction with the constraints described in Eq. 7, learning $\boldsymbol{W}$ can be expressed as the optimization problem

$$\begin{aligned} \boldsymbol{W}^* \quad &= \quad \underset{\boldsymbol{W}}{\operatorname{argmin}} \ D^2(\boldsymbol{W}^T\boldsymbol{X_s}, \boldsymbol{W}^T\boldsymbol{X_t}) \\ &\text{s.t.} \quad \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I} \ . \end{aligned} \tag{9}$$

As shown in Section 2, the MMD can be expressed in terms of a kernel function $k(\cdot, \cdot)$. Here, we first propose to exploit the Gaussian kernel function, which is known to be universal (Steinwart,

2002). This lets us rewrite our objective function as

$$\begin{aligned}\hat{D}^2_M(\boldsymbol{W}^T\boldsymbol{X_s},\boldsymbol{W}^T\boldsymbol{X_t}) = &\frac{1}{n^2}\sum_{i,j=1}^{n}\exp\left(-\frac{(\boldsymbol{x}_i^s-\boldsymbol{x}_j^s)^T\boldsymbol{W}\boldsymbol{W}^T(\boldsymbol{x}_i^s-\boldsymbol{x}_j^s)}{\sigma}\right)\\ &+\frac{1}{m^2}\sum_{i,j=1}^{m}\exp\left(-\frac{(\boldsymbol{x}_i^t-\boldsymbol{x}_j^t)^T\boldsymbol{W}\boldsymbol{W}^T(\boldsymbol{x}_i^t-\boldsymbol{x}_j^t)}{\sigma}\right)\\ &-\frac{2}{mn}\sum_{i,j=1}^{n,m}\exp\left(-\frac{(\boldsymbol{x}_i^s-\boldsymbol{x}_j^t)^T\boldsymbol{W}\boldsymbol{W}^T(\boldsymbol{x}_i^s-\boldsymbol{x}_j^t)}{\sigma}\right)\ ,\end{aligned} \tag{10}$$

where, in practice, we take σ to be the median squared distance between all the source examples.

Since the Gaussian kernel satisfies the universality condition of the MMD, it is a natural choice for our approach. However, it was shown that, in practice, choices of non-universal kernels may be more appropriate to measure the MMD (Borgwardt et al., 2006). In particular, the more general class of characteristic kernels can also be employed. This class incorporates all strictly positive definite kernels, such as the well-known polynomial kernel. Therefore, here, we also consider the polynomial kernel of degree two. The fact that this kernel yields a distribution distance that only compares the first and second moment of the two distributions (Gretton et al., 2012a) will be shown to have little impact on our experimental results, thus showing the robustness of our approach to the choice of kernel.

Replacing the Gaussian kernel with this polynomial kernel in our objective function yields

$$\begin{aligned}\hat{D}^2_M(\boldsymbol{W}^T\boldsymbol{X_s},\boldsymbol{W}^T\boldsymbol{X_t}) = &\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(1+\boldsymbol{x}_i^{sT}\boldsymbol{W}\boldsymbol{W}^T\boldsymbol{x}_j^s)^2\\ &+\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}(1+\boldsymbol{x}_i^{tT}\boldsymbol{W}\boldsymbol{W}^T\boldsymbol{x}_j^t)^2\\ &-\frac{2}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m}(1+\boldsymbol{x}_i^{sT}\boldsymbol{W}\boldsymbol{W}^T\boldsymbol{x}_j^t)^2.\end{aligned} \tag{11}$$

The two variants of the MMD introduced in Eqs. 10 and 11 can be computed efficiently in matrix form as

$$\hat{D}^2_M(\boldsymbol{W}^T\boldsymbol{X_s},\boldsymbol{W}^T\boldsymbol{X_t}) = Tr(\boldsymbol{K_W}\boldsymbol{L})\ , \tag{12}$$

where

$$\begin{aligned}\boldsymbol{K_W} &= \begin{bmatrix}\boldsymbol{K}_{s,s} & \boldsymbol{K}_{s,t}\\ \boldsymbol{K}_{t,s} & \boldsymbol{K}_{t,t}\end{bmatrix}\in\mathbb{R}^{(n+m)\times(n+m)}\ ,\ \text{and}\\ L_{ij} &= \begin{cases}1/n^2 & i,j\in\mathcal{S}\\ 1/m^2 & i,j\in\mathcal{T}\\ -1/(nm) & \text{otherwise}\end{cases}\ ,\end{aligned}$$

with $\mathcal{S}$ and $\mathcal{T}$ the sets of source and target indices, respectively. Each element in $\boldsymbol{K_W}$ is computed using the kernel function (either Gaussian, or polynomial), and thus depends on $\boldsymbol{W}$. Note that,

with both kernels, $\boldsymbol{K_W}$ can be computed efficiently in matrix form (i.e., without looping over its elements). This yields the optimization problem

$$\begin{aligned}\boldsymbol{W}^* &= \underset{\boldsymbol{W}}{\text{argmin}}\ Tr(\boldsymbol{K_W}\boldsymbol{L})\\ &\text{s.t. } \boldsymbol{W}^T\boldsymbol{W}=\boldsymbol{I}\ ,\end{aligned} \tag{13}$$

which is a nonlinear constrained problem. Due to the constraints, this problem can be solved either on the Stiefel manifold, or on the Grassmann manifold. The main difference between these two manifolds lies in the fact that, on the Grassmannian, two subspaces that are identical up to a d-dimensional rotation are identified as the same point on the manifold. In other words, a point on the Grassmann manifold is an equivalence class. It can easily be verified that, with our two kernels, a rotation of $\boldsymbol{W}$ would yield exactly the same objective function value. Therefore, our problem can be solved on the Grassmann manifold. The details of the optimization scheme and the resulting algorithm will be discussed in Section 3.3.

3.2 DME with the Hellinger Distance (DME-H)

While the MMD has nice properties (Gretton et al., 2012a), it does not truly consider the geometry of the space of probability distributions. Furthermore, according to (Gretton et al., 2012b), non-optimal choices of kernel and kernel parameters can lead to poor estimates of the distance between two distributions. This therefore motivates the use of the Hellinger distance instead of the MMD, since, as shown in Section 2.2, the Hellinger distance is related to the geodesic distance on the statistical manifold.

Given the linear mapping in Eq. 6 and the definition of the empirical estimate of the Hellinger distance in Eq. 4, we can express the (squared) distance between the source and target distributions as

$$\begin{aligned}\hat{D}^2_H(\boldsymbol{W}^T\boldsymbol{X_s},\boldsymbol{W}^T\boldsymbol{X_t}) &= \frac{1}{n}\sum_{i=1}^{n}\left(\sqrt{\hat{T}(\boldsymbol{W}^T\boldsymbol{x}_i^s)}-\sqrt{1-\hat{T}(\boldsymbol{W}^T\boldsymbol{x}_i^s)}\right)^2\\ &+ \frac{1}{m}\sum_{i=1}^{m}\left(\sqrt{\hat{T}(\boldsymbol{W}^T\boldsymbol{x}_i^t)}-\sqrt{1-\hat{T}(\boldsymbol{W}^T\boldsymbol{x}_i^t)}\right)^2\ .\end{aligned} \tag{14}$$

This distance depends on the function $\hat{T}(\boldsymbol{W}^T\boldsymbol{x})$, which, as mentioned in Section 2.2.1, is derived from the empirical estimates of the source and target distributions, $\hat{s}(\boldsymbol{W}^T\boldsymbol{x})$ and $\hat{t}(\boldsymbol{W}^T\boldsymbol{x})$, respectively.

In this work, we make use of kernel density estimation (KDE) with a Gaussian kernel to model these distributions. This lets us write

$$\hat{s}(\boldsymbol{W}^T\boldsymbol{x}) = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{\sqrt{|2\pi\boldsymbol{H}_s|}}\exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x}_j^s)^T\boldsymbol{W}\boldsymbol{H}_s^{-1}\boldsymbol{W}^T(\boldsymbol{x}-\boldsymbol{x}_j^s)}{2}\right), \tag{15}$$

where $\boldsymbol{H}_s$ is a diagonal matrix. In practice, we take $\boldsymbol{H}_s=\sigma_s\boldsymbol{I}$, where σ_s is computed using the maximal smoothing principle (Terrell, 1990) and kept constant. A similar estimate $\hat{t}(\boldsymbol{W}^T\boldsymbol{x})$ can be obtained from the m projected target samples $\{\boldsymbol{W}^T\boldsymbol{x}_j^t\}$. As such, we can write $\hat{T}(\boldsymbol{W}^T\boldsymbol{x})$ as:

$$\hat{T}(\boldsymbol{W}^T\boldsymbol{x}) = \frac{\frac{1}{n}\sum_{j=1}^{n}k(\boldsymbol{W}^T\boldsymbol{x},\boldsymbol{W}^T\boldsymbol{x}_j^s)}{\frac{1}{n}\sum_{j=1}^{n}k(\boldsymbol{W}^T\boldsymbol{x},\boldsymbol{W}^T\boldsymbol{x}_j^s)+\frac{1}{m}\sum_{j=1}^{m}k(\boldsymbol{W}^T\boldsymbol{x},\boldsymbol{W}^T\boldsymbol{x}_j^t)}\ , \tag{16}$$

where $k(\cdot,\cdot)$ is the Gaussian kernel function.

Finding a mapping that minimizes the Hellinger distance between the source and target distributions can then be expressed as

$$\begin{aligned} \boldsymbol{W}^* &= \min_{\boldsymbol{W}} \hat{D}_H^2(\boldsymbol{W}^T\boldsymbol{X_s}, \boldsymbol{W}^T\boldsymbol{X_t}) \\ \text{s.t.} \quad & \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}\ . \end{aligned} \tag{17}$$

As with the MMD, this is a nonlinear, constrained optimization problem, which, because of the form of the constraints, can be modeled as an optimization problem on either the Stiefel manifold or the Grassmannian. As before, it can easily be verified that the objective function will be unaffected by a rotation of $\boldsymbol{W}$. Therefore, this corresponds to a problem on the Grassmann manifold.

3.3 Learning the Mapping

As mentioned in the previous two sections, both DME-MMD and DME-H correspond to optimization problems on the Grassmann manifold. Optimization on Grassmann manifolds has proven effective at avoiding bad local minima (Absil et al., 2008). More precisely, manifold optimization methods often have better convergence behavior than iterative projection methods, which can be crucial with a nonlinear objective function (Absil et al., 2008).

By expressing $\boldsymbol{W}$ as a point on a Grassmann manifold, we can rewrite our constrained optimization problems as unconstrained problems on the manifold $\mathcal{G}(d,D)$. More specifically, we can generally express Problems (13) and (17) as

$$\boldsymbol{W^*} = \operatorname*{argmin}_{\boldsymbol{W}\in\mathcal{G}(d,D)} f(\boldsymbol{W})\ , \tag{18}$$

where $f(\boldsymbol{W})$ represents either the MMD-based objective function, or the Hellinger-based one.

While the optimization problem above has become unconstrained, it remains nonlinear. To effectively address this, we make use of a conjugate gradient method on the manifold. Recall from Section 2.3 that CG on a Grassmann manifold involves **(i)** computing the gradient on the manifold $\nabla f_{\boldsymbol{W}}$, **(ii)** estimating the search direction $\boldsymbol{H}$, and **(iii)** performing a line search along a geodesic. Our general approach to learning $\boldsymbol{W}$ can then be summarized by Algorithm 1, where we denote by $\tau(\Delta, \boldsymbol{W}, \boldsymbol{V})$ the parallel transport of tangent vector Δ from $\boldsymbol{W}$ to $\boldsymbol{V}$. In practice, we initialize $\boldsymbol{W}$ to the truncated identity matrix. We observed that learning $\boldsymbol{W}$ typically converges in only a few iterations.

Note that Eq. 5 shows that the gradient on the manifold depends on the partial derivatives of the objective function w.r.t. $\boldsymbol{W}$, i.e., $\partial f/\partial \boldsymbol{W}$. These derivatives depend on the specific form of the objective function, and are thus different for DME-MMD and DME-H.

For DME-MMD, the general form of $\partial f/\partial \boldsymbol{W}$ can be expressed as

$$\frac{\partial f}{\partial \boldsymbol{W}} = \sum_{i,j=1}^{n} \frac{\boldsymbol{G}_{ss}(i,j)}{n^2} + \sum_{i,j=1}^{m} \frac{\boldsymbol{G}_{tt}(i,j)}{m^2} - 2\sum_{i,j=1}^{n,m} \frac{\boldsymbol{G}_{st}(i,j)}{mn}\ ,$$

where $\boldsymbol{G}_{ss}(\cdot,\cdot)$, $\boldsymbol{G}_{tt}(\cdot,\cdot)$ and $\boldsymbol{G}_{st}(\cdot,\cdot)$ are matrices of size $D\times d$. With the definition of the MMD in Eq. 10 based on the Gaussian kernel $k_G(\cdot,\cdot)$, the matrix, e.g., $\boldsymbol{G}_{ss}(i,j)$ has the form

$$\boldsymbol{G}_{ss}(i,j) = -\frac{2}{\sigma} k_G(\boldsymbol{x}_s^i, \boldsymbol{x}_s^j)(\boldsymbol{x}_s^i - \boldsymbol{x}_s^j)(\boldsymbol{x}_s^i - \boldsymbol{x}_s^j)^T\boldsymbol{W}\ ,$$

Algorithm 1 : Learning on a Grassmann Manifold

Input:
$\boldsymbol{X_s}$: the source examples
$\boldsymbol{X_t}$: the target examples
d: the dimensionality of the subspace

Output:
$\boldsymbol{W}^* \in \mathbb{R}^{D\times d}$, such that $\boldsymbol{W}^{*T}\boldsymbol{W}^* = \boldsymbol{I}$

$\boldsymbol{W}_{prev} \leftarrow \boldsymbol{I}_{D\times d}$ (i.e., truncated identity matrix)
$\boldsymbol{W}_{cur} \leftarrow \boldsymbol{W}_{prev}$
$\boldsymbol{H}_{prev} \leftarrow \boldsymbol{0}$
repeat
  $\boldsymbol{D}_{cur} \leftarrow \nabla f_{\boldsymbol{W}_i}$
  $\boldsymbol{H}_{cur} \leftarrow -\boldsymbol{D}_{cur} + \eta\tau(\boldsymbol{H}_{prev}, \boldsymbol{W}_{prev}, \boldsymbol{W}_{cur})$
  Line search to find $\boldsymbol{W}^*$ that minimizes $f(\boldsymbol{W})$ along the geodesic at $\boldsymbol{W}_{cur}$ in the direction $\boldsymbol{H}_{cur}$
  $\boldsymbol{H}_{prev} \leftarrow \boldsymbol{H}_{cur}$
  $\boldsymbol{W}_{prev} \leftarrow \boldsymbol{W}_{cur}$
  $\boldsymbol{W}_{cur} \leftarrow \boldsymbol{W}^*$
until convergence

and similarly for $\boldsymbol{G}_{tt}(\cdot,\cdot)$ and $\boldsymbol{G}_{st}(\cdot,\cdot)$. With the MMD of Eq. 11 based on the degree 2 polynomial kernel $k_P(\cdot,\cdot)$, $\boldsymbol{G}_{ss}(i,j)$ becomes

$$\boldsymbol{G}_{ss}(i,j) = 2k_P(\boldsymbol{x}_s^i, \boldsymbol{x}_s^j)(\boldsymbol{x}_s^i\boldsymbol{x}_s^{jT} + \boldsymbol{x}_s^j\boldsymbol{x}_s^{iT})\boldsymbol{W}\ ,$$

and similarly for $\boldsymbol{G}_{tt}(\cdot,\cdot)$ and $\boldsymbol{G}_{st}(\cdot,\cdot)$. Similarly to the objective function itself, these derivatives can be efficiently computed in matrix form.

For DME-H, the derivatives can be written as

$$\begin{aligned} \frac{\partial f}{\partial \boldsymbol{W}} &= \tfrac{1}{n}\textstyle\sum_i^n \left\{ \sqrt{\frac{2T\left(\boldsymbol{W}^T\boldsymbol{x}_i^s\right)-1}{T\left(\boldsymbol{W}^T\boldsymbol{x}_i^s\right)\left(1-T\left(\boldsymbol{W}^T\boldsymbol{x}_i^s\right)\right)}}\frac{\partial T\left(\boldsymbol{W}^T\boldsymbol{x}_i^s\right)}{\partial \boldsymbol{W}} \right\} \\ &\quad + \tfrac{1}{m}\textstyle\sum_i^m \left\{ \sqrt{\frac{2T\left(\boldsymbol{W}^T\boldsymbol{x}_i^t\right)-1}{T\left(\boldsymbol{W}^T\boldsymbol{x}_i^t\right)\left(1-T\left(\boldsymbol{W}^T\boldsymbol{x}_i^t\right)\right)}}\frac{\partial T\left(\boldsymbol{W}^T\boldsymbol{x}_i^t\right)}{\partial \boldsymbol{W}} \right\}, \end{aligned} \tag{19}$$

where

$$\begin{aligned} \frac{\partial T(\boldsymbol{W}^T\boldsymbol{x})}{\partial \boldsymbol{W}} &= \frac{\partial}{\partial \boldsymbol{W}}\frac{s(\boldsymbol{W}^T\boldsymbol{x})}{s(\boldsymbol{W}^T\boldsymbol{x}) + t(\boldsymbol{W}^T\boldsymbol{x})} \\ &= \frac{1}{\left(s(\boldsymbol{W}^T\boldsymbol{x}) + t(\boldsymbol{W}^T\boldsymbol{x})\right)^2}\left(t(\boldsymbol{W}^T\boldsymbol{x})\frac{\partial s(\boldsymbol{W}^T\boldsymbol{x})}{\partial \boldsymbol{W}} - s(\boldsymbol{W}^T\boldsymbol{x})\frac{\partial t(\boldsymbol{W}^T\boldsymbol{x})}{\partial \boldsymbol{W}}\right). \end{aligned} \tag{20}$$

This lets us learn a linear mapping to a low-dimensional subspace that minimizes either the MMD or the Hellinger distance between the source and target data in a completely unsupervised

manner. A classifier can then be learned from the labeled source samples projected to this latent space, and directly applied to the projected target samples.

4. Nonlinear DME (NL-DME)

The Distribution-Matching Embedding methods introduced in the previous section make use of a linear mapping. As such, they have limited power to represent complex transformations between the source and target domains. To overcome this limitation, we introduce a nonlinear version of DME, which, as is often the case with nonlinear embedding techniques, boils down to applying a linear method after mapping the data to a high-dimensional Reproducing Kernel Hilbert Space.

More specifically, let $\varphi : \mathbb{R}^D \rightarrow \tilde{\mathcal{H}}$ be the function mapping an input vector $\boldsymbol{x}$ to a high-dimensional RKHS. Our goal is to learn an embedding of the form

$$\boldsymbol{y} = \boldsymbol{W}^T\varphi(\boldsymbol{x}) \; . \tag{21}$$

Since, in theory, $\tilde{\mathcal{H}}$ can be infinite-dimensional, learning the matrix $\boldsymbol{W}$ directly is not practical. Therefore, here, we make use of the Riesz representer theorem (Schölkopf and Smola, 2002; Canu and Smola, 2006) to express $\boldsymbol{W}$ as a linear combination of the examples in $\tilde{\mathcal{H}}$. In other words, we write

$$\boldsymbol{W} = \varphi(\boldsymbol{X_{s+t}})\boldsymbol{\alpha} \; , \tag{22}$$

where $\varphi(\boldsymbol{X_{s+t}})$ is the matrix containing all samples (i.e., source and target) mapped to $\tilde{\mathcal{H}}$, and $\boldsymbol{\alpha} \in \mathbb{R}^{(n+m)\times d}$ corresponds to the new parameters of the mapping.

4.1 NL-DME with the MMD (NL-DME-MMD)

Given the definition of the mapping above, we can write the Gaussian-kernel-based MMD as

$$\begin{aligned}
\hat{D}^2_M(\boldsymbol{W}^T\varphi(\boldsymbol{X_s}), \boldsymbol{W}^T\varphi(\boldsymbol{X_t})) &= \left\| \frac{1}{n}\sum_{i=1}^{n}\phi(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(\boldsymbol{x}_i^s)) - \frac{1}{m}\sum_{j=1}^{m}\phi(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(\boldsymbol{x}_j^t)) \right\|^2_{\mathcal{H}} \\
&= \frac{1}{n^2}\sum_{i,j=1}^{n}\exp\left(-\frac{(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_i^s) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^s))^T\boldsymbol{\alpha}\boldsymbol{\alpha}^T(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_i^s) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^s))}{\sigma}\right) \\
&+ \frac{1}{m^2}\sum_{i,j=1}^{m}\exp\left(-\frac{(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_i^t) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^t))^T\boldsymbol{\alpha}\boldsymbol{\alpha}^T(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_i^t) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^t))}{\sigma}\right) \\
&- \frac{2}{mn}\sum_{i,j=1}^{n,m}\exp\left(-\frac{(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_i^s) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^t))^T\boldsymbol{\alpha}\boldsymbol{\alpha}^T(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_i^s) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^t))}{\sigma}\right) \; ,
\end{aligned} \tag{23}$$

where $\tilde{k}(\cdot,\cdot)$ is the kernel function corresponding to the mapping to $\tilde{\mathcal{H}}$. Note that the MMD then only depends on kernel values and not on the high-dimensional representation $\varphi(\boldsymbol{x})$. It can easily be verified that this remains true when expressing the MMD with the degree 2 polynomial kernel, as in Eq. 11.

Learning a nonlinear distribution-matching embedding with the MMD can then be expressed as the optimization problem

$$\begin{aligned}
\boldsymbol{\alpha}^* &= \underset{\boldsymbol{\alpha}}{\text{argmin}} \; \hat{D}^2_M(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(\boldsymbol{X_s}), \boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(\boldsymbol{X_t})) \\
&\text{s.t.} \quad \boldsymbol{\alpha}^T\boldsymbol{K}\boldsymbol{\alpha} = \boldsymbol{I} \; ,
\end{aligned} \tag{24}$$

As in the linear case, the objective function can be computed efficiently in matrix form, thus yielding the optimization problem

$$\begin{aligned}
\boldsymbol{\beta}^* &= \underset{\boldsymbol{\beta}}{\text{argmin}} \; Tr\left(\boldsymbol{K'}_{\boldsymbol{\beta}}\boldsymbol{L}\right) \\
&\text{s.t.} \; \boldsymbol{\beta}^T\boldsymbol{\beta} = \boldsymbol{I} \; ,
\end{aligned} \tag{25}$$

where $\boldsymbol{L}$ is defined as in (12), $\boldsymbol{K'}_{\boldsymbol{\beta}}$ can be written as

$$\boldsymbol{K'}_{\boldsymbol{\beta}} = \begin{bmatrix} \boldsymbol{K}(\tilde{\boldsymbol{K}}_{s,s+t}, \tilde{\boldsymbol{K}}_{s,s+t}) & \boldsymbol{K}(\tilde{\boldsymbol{K}}_{s,s+t}, \tilde{\boldsymbol{K}}_{t,s+t}) \\ \boldsymbol{K}(\tilde{\boldsymbol{K}}_{t,s+t}, \tilde{\boldsymbol{K}}_{s,s+t}) & \boldsymbol{K}(\tilde{\boldsymbol{K}}_{t,s+t}, \tilde{\boldsymbol{K}}_{t,s+t}) \end{bmatrix} \; ,$$

and we defined a new variable $\boldsymbol{\beta} = \boldsymbol{K}^{1/2}\boldsymbol{\alpha}$. This new variable $\boldsymbol{\beta}$ can be represented as a point on a Grassmann manifold. Therefore, it allows us to make use of the same conjugate gradient method on the manifold as before. The original variable $\boldsymbol{\alpha}$ can then be obtained as $\boldsymbol{\alpha} = \boldsymbol{K}^{-1/2}\boldsymbol{\beta}$.

4.2 NL-DME with the Hellinger Distance (NL-DME-H)

Similarly, from the definition of our nonlinear mapping, the Hellinger distance can be written as

$$\begin{aligned}
\hat{D}^2_H(\boldsymbol{W}^T\varphi(\boldsymbol{X_s}), \boldsymbol{W}^T\varphi(\boldsymbol{X_t})) &= \\
&\frac{1}{n}\sum_{i=1}^{n}\left(\sqrt{\hat{T}(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(x_i^s))} - \sqrt{1-\hat{T}(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(x_i^s))}\right)^2 \\
&+ \frac{1}{m}\sum_{i=1}^{m}\left(\sqrt{\hat{T}(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(x_i^t))} - \sqrt{1-\hat{T}(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(x_i^t))}\right)^2 \; .
\end{aligned} \tag{26}$$

Using KDE, the distribution of the source data in the latent space can then be expressed as

$$\hat{s}(\boldsymbol{W}^T\varphi(\boldsymbol{x})) = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{\sqrt{|2\pi\boldsymbol{H}|}}\exp\left(-\frac{(\tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^s))^T\boldsymbol{\alpha}\boldsymbol{H}^{-1}\boldsymbol{\alpha}^T(\tilde{k}(\boldsymbol{X_{s+t}},x) - \tilde{k}(\boldsymbol{X_{s+t}},\boldsymbol{x}_j^s))}{2}\right) \; , \tag{27}$$

and similarly for the target distribution. Note that, here again, this only depends on kernel values.

This lets us write NL-DME-H as the optimization problem

$$\begin{aligned}
\boldsymbol{\alpha}^* &= \min_{\boldsymbol{\alpha}} \; \hat{D}^2_H(\boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(\boldsymbol{X_s}), \boldsymbol{\alpha}^T\varphi(\boldsymbol{X_{s+t}})^T\varphi(\boldsymbol{X_t})) \\
&\text{s.t.} \quad \boldsymbol{\alpha}^T\boldsymbol{K}\boldsymbol{\alpha} = \boldsymbol{I} \; .
\end{aligned} \tag{28}$$

As with the MMD, this problem can be re-written in terms of a new variable $\boldsymbol{\beta} = \boldsymbol{K}^{1/2}\boldsymbol{\alpha}$, and solved using a conjugate gradient method on the Grassmann manifold.

5. Related Work

We now discuss the domain adaptation methods that are most related to our approach. In particular, we focus on sample selection, or re-weighting, techniques and subspace-based methods.

Similarly to our approach, sample selection methods focus on comparing the distributions of the source and target data. In particular, in (Huang et al., 2006; Gretton et al., 2009), the source examples are re-weighted so as to minimize the MMD between the source and target distributions. More recently, an approach to selecting landmarks among the source examples based on the MMD was introduced (Gong et al., 2013). This sample selection approach was shown to be very effective, especially for the task of visual object recognition, to the point that it outperforms state-of-the-art semi-supervised approaches. Despite their success, it is important to note that sample re-weighting and selection methods compare the source and target distributions directly in the original feature space. More precisely, these techniques place a weight outside the mapping to Hilbert space $\phi(\cdot)$ performed in the MMD. Unfortunately, the original feature space may not be well-suited to compare the distributions, since the features may have been distorted by the domain shift, and since some of the features may only be relevant to one specific domain. By contrast, in this work, we compare the source and target distributions in a low-dimensional latent space where these effects are removed, or reduced.

Several techniques have also proposed to rely on subspaces to address the problem of domain adaptation. A popular approach in this class of methods differs significantly from our work in that, instead of learning a projection of the data, it seeks to directly represent the data with multiple subspaces (Blitzer et al., 2011; Gopalan et al., 2011; Gong et al., 2012). In particular, in (Blitzer et al., 2011), coupled subspaces are learned using Canonical Correlation Analysis (CCA). Rather than limiting the representation to one source and one target subspaces, several techniques exploit intermediate subspaces, which link the source data to the target data. This idea was originally introduced in (Gopalan et al., 2011), where the subspaces were modeled as points on a Grassmann manifold, and intermediate subspaces were obtained by sampling points along the geodesic between the source and target subspaces. This method was extended in (Gong et al., 2012), which showed that all intermediate subspaces could be taken into account by integrating along the geodesic. While this formulation nicely characterizes the change between the source and target data, it is not clear why all the subspaces along this path should yield meaningful representations. More importantly, these subspace-based methods do not explicitly exploit the statistical properties of the data.

By contrast, and similarly to our goal, a few methods have proposed to learn linear transformations of the data by considering the distributions of the different domains (Pan et al., 2011; Muandet et al., 2013). In particular, Transfer Component Analysis (TCA) (Pan et al., 2011) makes use of an MMD-based criterion to learn a subspace. However, in TCA, the linear transformation is applied outside the mapping to Hilbert space $\phi(\cdot)$ performed by the MMD. In other words, the distance between the sample means is measured in a lower-dimensional space rather than in RKHS, which somewhat contradicts the intuition behind the use of kernels. Domain-Invariant Component Analysis (DICA) (Muandet et al., 2013) is closely related to TCA in the sense that it is a kernel-based optimization algorithm that learns an invariant transformation of the data by minimizing the dissimilarity across domains. Moreover, it preserves the functional relationship between input and output variables based on the assumption that the functional relationship is stable or varies smoothly across domains.

Importantly, while the above-mentioned approaches have indeed also followed the idea of comparing distributions, they are all confined to using the MMD. In our experiments, we will show that in many cases the Hellinger distance yields better performance. Furthermore, to the best of our knowledge, no existing method has proposed to learn a nonlinear transformation of the data to account for the domain shift. As evidenced by our results, this again can yield to significant gains in accuracy.

6. Experiments

We evaluated our approach on the tasks of visual object recognition, cross-domain text categorization, and cross-domain WiFi localization, and compare its performance against the state-of-the art methods in each task.

In all our experiments, we used the subspace disagreement measure of (Gong et al., 2012) to automatically determine the dimensionality of the projection matrix $\boldsymbol{W}$. This method can be summarized as follows: We extracted PCA subspaces for the source data, the target data, and their combination. Intuitively, the similarity of the source and target domains should be directly proportional to the distance between these three subspaces on the Grassmann manifold. Therefore, the dimensionality d is taken as the one minimizing the sum of the minimum correlation distances (Hamm and Lee, 2008) between the source and combination subspaces and the target and combination subspaces, respectively. The same dimensionality was used for all dimensionality-reduction-based methods, which makes the comparison fair, since the dimensionality was not tuned for our approach either.

For recognition, we employed either a kernel SVM classifier with a degree 2 polynomial kernel, or a linear SVM classifier. These classifiers were trained on the projected source samples. The same types of classifiers were trained for all the baselines that do not inherently include a classifier (i.e., all the baselines except SVMA and DAM). The only parameter of such classifiers is the regularizer weight C. For each method, we tested with $C \in \{10^{-5}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$, and report the best result. Note that the parameters for SVMA, DAM and KMM were chosen using the code provided by the authors of these methods.

As mentioned earlier, the two hyperparameters of our approach were set as follows: The bandwidth of the Gaussian RBF kernel used in MMD was taken as the median distance computed over all pairwise data points; the value σ_s, such that $\boldsymbol{H}_s = \sigma_s \boldsymbol{I}$ in Eq. 15, was computed using the maximal smoothing principle (Terrell, 1990) .

6.1 Visual Object Recognition

We first evaluated our approach on the task of visual object recognition using the benchmark domain adaptation data set introduced in (Saenko et al., 2010). This data set contains images from four different domains: Amazon, DSLR, Webcam, and Caltech. The Amazon domain consists of images acquired in a highly-controlled environment with studio lighting conditions. These images capture the large intra-class variations of 31 classes, but typically show the objects only from one canonical viewpoint. The DSLR domain consists of high resolution images of 31 categories that were taken with a digital SLR camera in a home environment under natural lighting. The Webcam images were acquired in a similar environment as the DSLR ones, but have much lower resolution and contain significant noise, as well as color and white balance artifacts. The last domain, Caltech (Griffin et al., 2007), consists of images of 256 object classes downloaded from Google images. Following (Gong

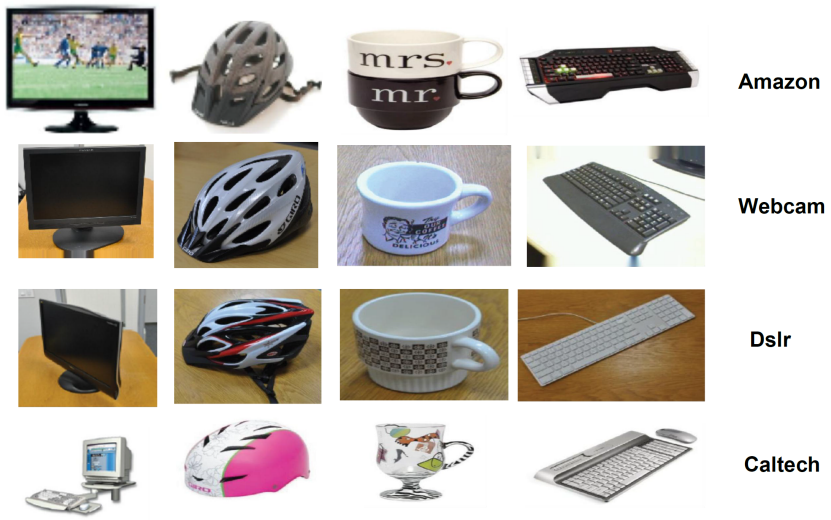


Figure 3: Sample images from the object categories ***monitor***, ***helmet***, ***mug*** and ***keyboard*** *in the four domains* ***Amazon***, ***Webcam***, ***DSLR***, *and* ***Caltech***.

et al., 2012), we used the 10 object classes common to all four data sets. This yields 2533 images in total, with 8 to 151 images per category and per domain. Fig. 3 depicts sample images from the four domains.

For our evaluation, we used two different types of image features. First, we employed the features provided by (Gong et al., 2012), which were obtained using the protocol described in (Saenko et al., 2010). More specifically, all images were converted to grayscale and resized to have the same width. Local scale-invariant interest points were detected by the SURF detector (Bay et al., 2006), and a 64-dimensional rotation invariant SURF descriptor was extracted from the image patch around each interest point. A codebook of size 800 was then generated from a subset of the Amazon data set using k-means clustering on the SURF descriptors. The final feature vector for each image is the normalized histogram of visual words obtained from this codebook. As a second feature type, we used the deep learning features of (Donahue et al., 2014) which have shown promising results for object recognition. Specifically, as visual features, we used the outputs derived from the activation of the 6th, 7th, and 8th layers of a deep convolutional network (CNN) with weights trained on the ImageNet data set (Deng et al., 2009), leading to 4096-dimensional $DeCAF_6$ and $DeCAF_7$ features, as well as 1000-dimensional $DeCAF_8$ features (Tommasi and Tuytelaars, 2014).

We used the conventional evaluation protocol introduced in (Saenko et al., 2010), which consists of splitting the data into multiple partitions. For each source/target pair, we report the average recognition accuracy and standard deviation over the 20 partitions provided with GFK[1]. With this protocol, we evaluated all possible combinations of source and target domains. For all the methods based on dimensionality reduction, we used the dimensionalities provided in the GFK code (i.e.,

1. www-scf.usc.edu/~boqinggo/domainadaptation.html

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	38.7 ± 1.6	36.7 ± 2.3	37.2 ± 2.8	44.3 ± 2.4	41.1 ± 3.9	39.9 ± 3.2
SVMA (Duan et al., 2012)	34.77 ± 1.43	34.14 ± 3.70	32.47 ± 2.85	39.13 ± 2.02	34.52 ± 3.54	32.88 ± 2.27
DAM (Duan et al., 2012)	34.92 ± 1.46	34.27 ± 3.58	32.54 ± 2.72	39.20 ± 2.07	34.65 ± 3.53	33.05 ± 2.27
GFK (Gong et al., 2012)	37.2 ± 1.9	37.7 ± 3.3	38.9 ± 3.1	46.6 ± 2.7	36.8 ± 1.8	37.2 ± 4.0
TCA (Pan et al., 2011)	40 ± 1.3	39.1 ± 1.5	40.1 ± 1.2	46.7 ± 1.1	41.4 ± 1.2	36.2 ± 1.0
SA (Fernando et al., 2013)	41 ± 1.8	41.2 ± 5.0	42 ± 3.5	48.2 ± 3.1	50.3 ± 4.2	46.5 ± 4.9
KMM (Huang et al., 2006)	40.7 ± 2.1	39.8 ± 1.9	39.0 ± 3.7	48.6 ± 2.8	46.6 ± 3.3	42.2 ± 4.3
DME-MMD	43.3 ± 1.4	42.8 ± 2.5	46.7 ± 2.7	50 ± 3.2	49 ± 2.9	47.6 ± 3.5
DME-MMD (Poly)	43.1 ± 1.3	41.3 ± 2.7	45.6 ± 2.4	50.6 ± 2.9	47.8 ± 3.1	46.1 ± 3.1
DME-H	44.5 ± 1.7	43.2 ± 0.9	48.6 ± 2.3	51.9 ± 1.4	52.5 ± 2.9	47.3 ± 4.6
NL-DME-MMD	43.1 ± 1.9	44.4 ± 3.8	45.4 ± 4.0	50.4 ± 2.3	50.9 ± 3.1	49.6 ± 3.3
NL-DME-H	44.3 ± 1.3	47.2 ± 2.1	47.7 ± 2.8	52.1 ± 3.2	52.6 ± 2.8	48.8 ± 3.0

Table 1: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using SURF features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 2.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	33.6 ± 1.7	31.1 ± 0.9	75.2 ± 2.6	36.9 ± 1.2	33.4 ± 1.1	80.2 ± 2.5	44
SVMA (Duan et al., 2012)	33.43 ± 1.24	31.40 ± 0.87	74.44 ± 2.21	36.63 ± 1.08	33.52 ± 0.77	74.97 ± 2.65	41.1
DAM (Duan et al., 2012)	33.50 ± 1.29	31.52 ± 0.88	74.68 ± 2.14	34.73 ± 1.14	31.18 ± 1.25	68.34 ± 3.16	40.2
GFK (Gong et al., 2012)	37.7 ± 1.8	33.3 ± 1.3	79.9 ± 2.8	41.5 ± 1.8	34.5 ± 0.9	76.7 ± 1.4	44.8
TCA (Pan et al., 2011)	39.6 ± 1.2	34 ± 1.1	80.4 ± 2.6	40.2 ± 1.1	33.7 ± 1.1	77.5 ± 2.5	42.8
SA (Fernando et al., 2013)	41.1 ± 1.6	35.4 ± 1.8	84.4 ± 2.4	38.2 ± 1.4	33.3 ± 1.2	83.3 ± 1.6	48.7
KMM (Huang et al., 2006)	38 ± 1.8	34.3 ± 1.2	82.0 ± 1.7	39.0 ± 1.2	35.3 ± 1.0	86.8 ± 2.0	47.7
DME-MMD	40.5 ± 1	39 ± 0.5	86.7 ± 1.2	42.5 ± 1.5	37 ± 0.9	86.4 ± 1.8	50.9
DME-MMD (Poly)	40.8 ± 0.9	39.1 ± 0.6	87.1 ± 1.0	41.3 ± 1.3	36.8 ± 0.9	85.8 ± 2.2	50.4
DME-H	39.1 ± 0.6	38.9 ± 0.4	88.6 ± 1.0	44.1 ± 0.8	39.9 ± 0.7	89.3 ± 0.5	52.3
NL-DME-MMD	40.1 ± 2.2	37.6 ± 0.6	87.3 ± 1.0	41.02 ± 1.3	36.7 ± 2.4	86.7 ± 2.2	51.1
NL-DME-H	41.6 ± 1.3	36.4 ± 0.4	87.5 ± 1.1	44.3 ± 1.1	38 ± 0.7	88.8 ± 1.6	**52.5**

Table 2: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using SURF features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, our approach clearly outperforms the baselines, with best performance for NL-DME-H.

W-D: 10, D-A: 20, W-A: 10, C-W: 20, C-D: 10, C-A: 20. Note that the dimensionality for X-Y is the same as for Y-X).

For the kernel SVM classifier, we report the results of our algorithms and several baselines on all source and target pairs in Tables 1 and 2 for the SURF features, in Tables 3 and 4 for the DeCAF6 features, in Tables 5 and 6 for the DeCAF7 features, and in Tables 7 and 8 for the DeCAF8 features. Similar results using a linear SVM classifier are provided in Tables 9 to 16. In the last column of every other table, we report the average accuracy over all the source-target pairs. Note that, with SURF features, all our algorithms clearly outperform the baselines, with the best average accuracy

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	81.6 ± 1.4	82.6 ± 3.4	74.6 ± 3.3	89.8 ± 1.5	84.3 ± 2.6	77.8 ± 1.7
SVMA (Duan et al., 2012)	83.54	81.72	74.58	91.00	83.89	76.61
DAM (Duan et al., 2012)	84.73	82.48	78.14	91.8	84.59	79.39
GFK (Gong et al., 2012)	84.8 ± 1.0	89.3 ± 2.4	84.6 ± 2.1	90.9 ± 1.0	87.1 ± 1.5	85.2 ± 1.5
TCA(Pan et al., 2011)	82.9 ± 0.9	89 ± 2.0	77.8 ± 3.9	89.9 ± 1.7	87.8 ± 2.1	82.9 ± 2.0
SA (Fernando et al., 2013)	86.1 ± 0.9	80.1 ± 1.0	75.3 ± 0.8	91.5 ± 0.7	85.6 ± 4.1	85.1 ± 0.9
KMM (Huang et al., 2006)	83.7 ± 1.4	86.7 ± 2.8	75.4 ± 4.6	90.4 ± 1.3	85.03 ± 3.1	78 ± 3.3
DME-MMD	84.3 ± 1.0	89.2 ± 3.7	81.5 ± 4.3	90.3 ± 0.6	87.3 ± 2.3	84.7 ± 3.4
DME-H	85.1 ± 1.4	89.5 ± 2.5	80.9 ± 3.4	90.9 ± 1.1	88.1 ± 2.1	83.2 ± 2.7
NL-DME-MMD	85.3 ± 1.0	89.3 ± 3.0	79.8 ± 3.1	91.4 ± 0.7	87.9 ± 2.3	82.2 ± 2.7
NL-DME-H	85.3 ± 1.1	91.1 ± 2.1	83.8 ± 3.6	91.6 ± 0.9	86.8 ± 3.3	85.2 ± 2.9

Table 3: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf6 features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 4.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	79.2 ± 2.3	73.4 ± 2.0	95.6 ± 1.1	75.3 ± 1.5	69.5 ± 1.1	99.4 ± 0.6	81.9
SVMA (Duan et al., 2012)	85.37	78.14	96.71	74.36	70.58	96.6	82.7
DAM (Duan et al., 2012)	87.88	81.27	96.31	76.6	74.32	93.8	84.2
GFK (Gong et al., 2012)	84.2 ± 2.3	77.5 ± 2.0	96.4 ± 1.1	85.4 ± 1.7	77.1 ± 0.5	99.5 ± 0.3	86.8
TCA (Pan et al., 2011)	84.1 ± 1.6	77.7 ± 1.9	95.9 ± 0.8	83.8 ± 1.0	76.5 ± 0.9	98.6 ± 0.9	85.6
SA (Fernando et al., 2013)	90.1 ± 0.9	83.9 ± 1.6	96.8 ± 1.6	85.0 ± 3.3	78.7 ± 2.8	99.3 ± 0.7	86.5
KMM (Huang et al., 2006)	84.3 ± 2.4	77.4 ± 1.1	96.2 ± 1.8	75.5 ± 3.2	72.8 ± 1.9	97.9 ± 0.9	83.6
DME-MMD	82.9 ± 2.9	77.5 ± 2.7	96.4 ± 1.2	82.1 ± 1.9	78.6 ± 1.4	98.8 ± 0.3	86.2
DME-H	84.5 ± 2.5	79.6 ± 1.8	97 ± 0.9	83.9 ± 1.1	77.9 ± 1.4	99.7 ± 0.4	86.7
NL-DME-MMD	86.4 ± 2.2	76.01 ± 2.9	97.7 ± 1.3	84.3 ± 1.4	77.3 ± 1.5	98.6 ± 0.7	86.4
NL-DME-H	86.3 ± 2.6	82.2 ± 2.6	98.1 ± 1.4	86.1 ± 1.6	78.1 ± 1.6	99.3 ± 1.0	**87.9**

Table 4: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf6 features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, NL-DME-H still yields the best accuracy.

achieved by NL-DME-H. To the best of our knowledge, this represents the state-of-the-art result on this data. With deep learning features, our algorithms yield accuracies that are on par with the state-of-the-art results. Note that, among our algorithms, the best results are achieved using Decaf7 features with NL-DME-H. In Fig. 4, we illustrate the behavior of our algorithms on the *mug* class. As shown in the bottom-right panel, even humans would have a hard time to correctly label some of the misclassified examples.

We then further compare the results of our approach and the previous baselines with two recent deep learning DA methods, Deep Domain Confusion (DDC) (Tzeng et al., 2014) and Reverse Gradient (RG) (Ganin and Lempitsky, 2015). Table 17 was computed from the Office-Caltech data set with 10 classes, as in the previous experiments. In this setting, only the results of DDC for 6 pairs

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	84.2 ± 0.8	87.9 ± 1.8	77.5 ± 1.5	89.9 ± 1.2	84.9 ± 3.5	78.9 ± 2.6
SVMA (Duan et al., 2012)	84.5 ± 1.3	84.9 ± 3.2	74.8 ± 4.41	91.8 ± 0.70	83.5 ± 2.3	78.5 ± 4.0
DAM (Duan et al., 2012)	85.5 ± 1.2	84.0 ± 5.0	77.4 ± 4.55	92.2 ± 0.6	83.5 ± 2.7	81.1 ± 3.9
GFK (Gong et al., 2012)	85.8 ± 0.5	91 ± 2.2	83.7 ± 2.4	91.3 ± 0.9	88.3 ± 1.7	85.7 ± 2.4
SA (Fernando et al., 2013)	86.4 ± 0.8	91.4 ± 3.2	87.8 ± 2.5	92.2 ± 0.7	89.0 ± 3.2	88.9 ± 2.1
TCA(Pan et al., 2011)	84.6 ± 0.8	89.6 ± 2.8	82.3 ± 4.2	89.8 ± 1.2	87.3 ± 3.5	83.7 ± 3.6
KMM(Huang et al., 2006)	85.7 ± 0.7	86.8 ± 1.4	76.5 ± 1.6	91.3 ± 0.6	85.3 ± 2.8	79.8 ± 4.1
DME-MMD	85.4 ± 0.7	91.1 ± 1.3	81.5 ± 1.6	91.4 ± 0.4	88.03 ± 1.9	82.6 ± 2.3
DME-H	85.6 ± 0.9	89.3 ± 2.2	80 ± 2.2	91.8 ± 0.6	86.4 ± 2.7	83.6 ± 3.01
NL-DME-MMD	85.7 ± 0.6	89.2 ± 2.5	83.6 ± 2.3	91.6 ± 0.3	88.9 ± 2.7	85.6 ± 2.3
NL-DME-H	86.7 ± 0.9	89.4 ± 2.0	79.7 ± 2.5	91.8 ± 0.5	89.6 ± 2.3	83.8 ± 2.04

Table 5: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf7 features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 6.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	84.0 ± 1.9	77.9 ± 1.1	96.6 ± 1.6	82.9 ± 1.1	74.4 ± 0.7	99.1 ± 0.6	84.8
SVMA (Duan et al., 2012)	87.6 ± 1.1	80.8 ± 0.9	96.0 ± 1.5	81.0 ± 1.6	77.0 ± 0.7	98.4 ± 1.0	84.9
DAM (Duan et al., 2012)	90.1 ± 1.1	83.1 ± 1.2	95.3 ± 1.3	81.7 ± 3.4	77.8 ± 2.4	95.6 ± 2.2	85.6
GFK (Gong et al., 2012)	85.9 ± 1.9	80.7 ± 1.3	96.6 ± 1.3	89.2 ± 1.0	78.8 ± 0.6	99.6 ± 0.5	88.1
SA (Fernando et al., 2013)	89.8 ± 0.7	83.7 ± 1.8	97.1 ± 0.8	88.3 ± 2.2	83.4 ± 0.7	99.6 ± 0.3	**89.8**
TCA(Pan et al., 2011)	86.0 ± 1.6	80.5 ± 1.1	95.6 ± 1.8	90.1 ± 0.8	78.7 ± 1.0	98.3 ± 0.7	87.2
KMM(Huang et al., 2006)	87.9 ± 1.2	81.5 ± 1.1	96.7 ± 1.2	83.9 ± 1.9	77.8 ± 1.2	98.4 ± 0.8	86.0
DME-MMD	85.6 ± 1.8	80.8 ± 2.8	96.5 ± 0.7	83.6 ± 2.2	77.3 ± 2.5	99.4 ± 0.4	86.9
DME-H	88.3 ± 1.6	78.9 ± 2.1	97 ± 1.7	85.1 ± 2.3	78.9 ± 1.6	99.3 ± 0.4	87.01
NL-DME-MMD	86.2 ± 1.1	82.5 ± 0.9	96.7 ± 1.3	89 ± 0.8	78.3 ± 0.6	99.6 ± 0.3	88.1
NL-DME-H	89.5 ± 0.8	82.8 ± 2.3	97.3 ± 1	90.1 ± 3.1	79.8 ± 2.0	99.2 ± 0.7	88.3

Table 6: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf7 features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, NL-DME-H still yields competitive accuracy.

were available. Note that our approach yields slightly better accuracies than DDC. In Table 18, we compare our results with both deep learning baselines using the 31 classes of the original Office data set, and for the 3 pairs that were reported in (Ganin and Lempitsky, 2015). Note that, here, while our approach is among the top performer non-deep-learning methods, the two works that jointly learn the features and perform domain adaptation tend to perform better. This suggests an interesting avenue for future research by incorporating our Hellinger-based metric within a deep learning framework.

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	75.3 ± 1.4	83.5 ± 2.2	75.6 ± 2.1	77.6 ± 1.3	81.3 ± 1.3	73.1 ± 1.4
SVMA (Duan et al., 2012)	76.3 ± 0.8	85.1 ± 1.1	75.8 ± 2.3	79.5 ± 0.6	82.1 ± 1.3	73.3 ± 3.1
DAM (Duan et al., 2012)	77.1 ± 1.1	85.4 ± 1.2	77.8 ± 2.5	79.9 ± 0.6	84.2 ± 1.0	75.4 ± 2.7
GFK (Gong et al., 2012)	76.1 ± 2.0	86.4 ± 1.9	78.9 ± 2.4	79.4 ± 0.9	82.4 ± 1.1	76.3 ± 2.2
SA (Fernando et al., 2013)	77.2 ± 1.4	84.6 ± 1.2	78.8 ± 1.5	78.9 ± 1.2	85.1 ± 1.4	73.7 ± 2.2
TCA(Pan et al., 2011)	75.9 ± 1.6	86.7 ± 1.7	75.7 ± 1.5	79.7 ± 0.8	82.5 ± 1.0	75.6 ± 2.3
KMM(Huang et al., 2006)	77.3 ± 1.1	83.1 ± 1.6	75.8 ± 2.2	79.9 ± 1.0	81.4 ± 2.1	77.3 ± 1.7
DME-MMD	75.6 ± 0.8	86.2 ± 2.1	75.4 ± 1.9	78.9 ± 1.0	81.8 ± 1.3	73.5 ± 1.5
DME-H	77.5 ± 1.0	87.6 ± 1.9	75.8 ± 2.8	79.4 ± 0.8	83.5 ± 1.0	74.5 ± 3.1
NL-DME-MMD	76.5 ± 1.2	86.9 ± 2.6	78.2 ± 2.5	80.3 ± 1.2	81.9 ± 1.7	77.9 ± 1.5
NL-DME-H	78.9 ± 1.2	87.4 ± 1.7	79.3 ± 2.4	81 ± 0.9	82.4 ± 0.8	77.3 ± 1.6

Table 7: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf8 features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 8.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	80.3 ± 0.9	73.6 ± 0.8	81.4 ± 1.6	84.1 ± 0.5	78.3 ± 1.2	93.3 ± 0.8	79.7
SVMA (Duan et al., 2012)	81.8 ± 2.1	75.0 ± 1.1	83.1 ± 1.0	83.5 ± 0.4	77.4 ± 0.6	93.5 ± 1.0	80.5
DAM (Duan et al., 2012)	82.1 ± 2.1	75.7 ± 1.1	83.5 ± 0.8	82.0 ± 1.7	78.8 ± 1.8	90.2 ± 1.5	81.1
GFK (Gong et al., 2012)	81.7 ± 0.6	74.9 ± 1.4	82.3 ± 1.5	83.1 ± 0.4	80.2 ± 1.6	93.6 ± 1.1	81.3
SA (Fernando et al., 2013)	82.4 ± 0.8	77.9 ± 1.7	83.4 ± 1.3	82.3 ± 0.5	81.0 ± 1.6	90.7 ± 2.1	81.3
TCA(Pan et al., 2011)	79.7 ± 2.4	74.4 ± 1.7	81.1 ± 1.9	82.9 ± 0.6	78.9 ± 1.0	93.6 ± 1.4	80.6
KMM(Huang et al., 2006)	79.5 ± 2.0	74.5 ± 1.8	82.3 ± 1.9	84.3 ± 0.6	78.8 ± 1.1	93.4 ± 0.9	80.6
DME-MMD	82.6 ± 0.5	74.5 ± 1.7	80.2 ± 1.5	83.3 ± 0.7	78.4 ± 1.0	95.2 ± 1.1	80.5
DME-H	80.7 ± 1.5	75.4 ± 1.1	82.8 ± 1.3	83.6 ± 1.2	79.6 ± 1.2	95.3 ± 0.5	81.3
NL-DME-MMD	83.3 ± 0.9	75.6 ± 1.5	81.4 ± 1.0	82.8 ± 0.5	78.3 ± 1.3	93.7 ± 1.5	81.4
NL-DME-H	82.9 ± 0.9	76.7 ± 1.7	84.9 ± 1.6	83.2 ± 0.9	79.9 ± 1.3	94.6 ± 0.8	**82.4**

Table 8: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf8 features with Kernel SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, NL-DME-H still yields the best accuracy.

6.2 Cross-domain Text Categorization

As a second type of experiment, we made use of the 20 Newsgroups data set, which has become popular in the machine learning community to evaluate methods tackling problems such as text classification and text clustering. The 20 Newsgroups data set is a collection of 18,774 newsgroup documents organized in a hierarchical structure of six main categories and 20 subcategories, each corresponding to a different topic. Some of the newsgroups (from the same category) are very closely related to each other (e.g., comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g., misc.forsale / soc.religion.christian), making this data set well-suited to evaluate cross-domain learning algorithms.

For this experiment, we used the protocol of (Duan et al., 2012): The four largest main categories (comp, rec, sci, and talk) were chosen for evaluation. Specifically, for each main category, the largest

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	38.3 ± 1.8	35.9 ± 2.6	38.1 ± 2.4	44.7 ± 2.1	40.7 ± 3.3	37.4 ± 1.9
SVMA (Duan et al., 2012)	34.77 ± 1.43	34.14 ± 3.70	32.47 ± 2.85	39.13 ± 2.02	34.52 ± 3.54	32.88 ± 2.27
DAM (Duan et al., 2012)	34.92 ± 1.46	34.27 ± 3.58	32.54 ± 2.72	39.20 ± 2.07	34.65 ± 3.53	33.05 ± 2.27
GFK (Gong et al., 2012)	41.1 ± 1.2	40.5 ± 4.3	42.5 ± 4.7	47.9 ± 3.5	47.8 ± 2.0	44.4 ± 3.8
TCA (Pan et al., 2011)	37 ± 2.5	36.6 ± 3.2	33.2 ± 4.4	42.9 ± 2.7	39.5 ± 3.2	36.03 ± 4.3
SA (Fernando et al., 2013)	40.6 ± 1.7	39.2 ± 2.6	37.6 ± 4.5	49.4 ± 3.1	48.7 ± 3.6	43.4 ± 2.9
KMM (Huang et al., 2006)	38.3 ± 1.8	36 ± 2.6	38.1 ± 2.4	44.7 ± 2.1	40.7 ± 3.3	37.4 ± 1.9
DME-MMD	43.1 ± 1.9	39.9 ± 4.0	41.7 ± 3.2	45.9 ± 3.1	43.7 ± 3.8	45.7 ± 3.6
DME-H	42.7 ± 1.9	40.3 ± 3.8	42 ± 2.7	46.7 ± 2.4	44.1 ± 2.6	45.2 ± 3.1
NL-DME-MMD	42.1 ± 1.3	41.5 ± 3.5	40.8 ± 3.8	48.5 ± 2.3	45 ± 4.5	47.3 ± 4.5
NL-DME-H	41.5 ± 1.6	41.7 ± 3.4	40.5 ± 3.0	49.4 ± 3.4	46.1 ± 3.8	47.1 ± 3.8

Table 9: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using SURF features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 10.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	33.8 ± 2.0	31.4 ± 1.3	75.5 ± 2.2	36.6 ± 1.1	32 ± 1.1	80.5 ± 1.4	43.7
SVMA (Duan et al., 2012)	33.4 ± 1.2	31.4 ± 0.9	74.4 ± 2.2	36.6 ± 1.1	33.5 ± 0.8	74.9 ± 2.7	41.1
DAM (Duan et al., 2012)	33.5 ± 1.3	31.5 ± 0.9	74.6 ± 2.1	34.7 ± 1.1	31.1 ± 1.3	68.3 ± 3.2	40.2
GFK (Gong et al., 2012)	36.9 ± 2.9	34.3 ± 1.7	77.3 ± 2.2	41 ± 1.7	34.3 ± 1.1	75.8 ± 2.8	47
TCA (Pan et al., 2011)	34.4 ± 1.9	33.2 ± 1.7	76.4 ± 2.3	38.3 ± 2.2	33.8 ± 1.6	57.8 ± 5.2	41.6
SA (Fernando et al., 2013)	37.7 ± 2.2	33.9 ± 1.3	74.8 ± 4.5	39.4 ± 1.2	35.3 ± 1.6	76.6 ± 3.9	46.4
KMM (Huang et al., 2006)	33.8 ± 2.0	31.4 ± 1.3	75.5 ± 2.2	36.6 ± 1.1	32.1 ± 1.1	80.5 ± 1.4	43.8
DME-MMD	38.1 ± 2.7	32.8 ± 1.1	74.9 ± 1.9	42.3 ± 2.8	35.2 ± 1.2	74.5 ± 2.1	46.4
DME-H	37.4 ± 1.9	34.6 ± 1.7	74.3 ± 1.4	41.3 ± 0.9	35 ± 1.4	73.9 ± 2.2	46.5
NL-DME-MMD	36.9 ± 2.2	32.5 ± 1.3	76.8 ± 3.1	40.4 ± 1.9	35.7 ± 0.7	79.5 ± 3.2	47.2
NL-DME-H	38.5 ± 1.6	34.8 ± 1.7	73.6 ± 2.0	42.1 ± 1.1	35.8 ± 0.8	78.4 ± 3.0	**47.5**

Table 10: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using SURF features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, our approach outperforms the baselines, with best performance for NL-DME-H.

subcategory was selected as the target domain. We considered the largest category "comp" as the positive class and one of the three other categories as the negative class for each setting. Table 19 provides detailed information about the three settings. We used word-frequency features to represent each document. To construct the training set, we used 1000 randomly selected samples (evenly distributed positive and negative samples) from the source domain, and repeated this procedure 5 times. For each such partition, our mappings were then learned using this training data and the unlabeled samples from the target domain.

In Table 20, we report the mean recognition accuracies of our algorithms and of state-of-the-art baselines. For all the baselines, we employed a kernel SVM classifier with a degree 2 polynomial kernel, we tested with the regularizer weight $C \in \{10^{-5}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$, and report the best result. For all the baselines based on dimensionality reduction, we set the dimen-

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	83.4 ± 0.8	85.4 ± 3.0	77.1 ± 2.5	90.5 ± 1.2	84.2 ± 2.9	77.4 ± 2.8
SVMA (Duan et al., 2012)	83.54	81.72	74.58	91.00	83.89	76.61
DAM (Duan et al., 2012)	84.73	82.48	78.14	91.8	84.59	79.39
GFK (Gong et al., 2012)	85.2 ± 0.9	86.8 ± 1.0	80.8 ± 1.8	91.3 ± 1.2	84.7 ± 1.7	82.8 ± 2.4
TCA(Pan et al., 2011)	85.5 ± 1.2	87.4 ± 3.6	80.6 ± 1.8	91.2 ± 1.2	84.9 ± 2.1	81.0 ± 1.9
SA (Fernando et al., 2013)	84.9 ± 0.7	87.0 ± 3.9	79.6 ± 5.7	91.6 ± 1.0	86.6 ± 2.7	84 ± 1.8
KMM (Huang et al., 2006)	83.6 ± 0.7	85.2 ± 3.0	75.7 ± 2.9	90.5 ± 1.2	84.5 ± 3.1	77.0 ± 2.2
DME-MMD	84.3 ± 0.8	83.6 ± 3.7	76.4 ± 1.5	88 ± 1.9	84.4 ± 3.4	77.4 ± 6.4
DME-H	84.7 ± 0.8	86.1 ± 1.1	77.6 ± 2.5	91.3 ± 1.7	85.2 ± 3.3	77.8 ± 3.3
NL-DME-MMD	84.8 ± 0.9	86.2 ± 0.8	78 ± 5.2	91.5 ± 1.0	85.9 ± 2.3	78.6 ± 3.9
NL-DME-H	83.6 ± 1.2	83.4 ± 4.4	77.03 ± 2.7	90.4 ± 1.6	85 ± 2.5	81.3 ± 3.6

Table 11: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf6 features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 12.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	85.5 ± 2.4	77.1 ± 2.0	94.1 ± 1.7	76.6 ± 2.8	70.5 ± 1.2	98.9 ± 0.8	83.3
SVMA (Duan et al., 2012)	85.37	78.14	96.71	74.36	70.58	96.6	82.7
DAM (Duan et al., 2012)	87.88	81.27	96.31	76.6	74.32	93.8	84.2
GFK (Gong et al., 2012)	88.2 ± 1.2	80.7 ± 1.9	97.4 ± 1.5	88 ± 0.9	79.5 ± 1.4	98.9 ± 0.8	87
TCA (Pan et al., 2011)	89.9 ± 0.9	82.7 ± 2.1	95.9 ± 1.5	86.7 ± 1.5	77.8 ± 2.8	99.2 ± 0.4	86.9
SA (Fernando et al., 2013)	91.2 ± 0.4	84.2 ± 1.2	97.4 ± 1.2	89.6 ± 0.6	79.9 ± 1.1	99.5 ± 0.4	**87.9**
KMM (Huang et al., 2006)	85.6 ± 1.5	77.4 ± 2.3	97.1 ± 1.7	76.0 ± 2.5	72.2 ± 1.4	98.4 ± 1.0	83.6
DME-MMD	88.2 ± 1.1	85.2 ± 2.1	95.5 ± 0.9	87.6 ± 1.8	74.9 ± 1.4	98.8 ± 0.8	85.4
DME-H	86.4 ± 1.7	85.6 ± 0.6	95.1 ± 1.3	87.3 ± 1.3	73.5 ± 0.9	98 ± 1.0	85.7
NL-DME-MMD	86.8 ± 1.5	84.2 ± 1.1	97.2 ± 1.1	86.8 ± 2.4	77.2 ± 1.5	99.4 ± 0.7	86.3
NL-DME-H	91.7 ± 0.8	85 ± 1.3	96.1 ± 1.1	90.6 ± 2.2	82.5 ± 1.7	99.0 ± 0.9	87.1

Table 12: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf6 features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, NL-DME-H still yields competitive accuracy.

sionalities based on the Subspace Disagreement Measure (SDM) (Gong et al., 2012)[2] (i.e., comp vs. rec: 10, comp vs. sci: 27, comp vs. talk: 47). Similarly to the visual recognition task, our algorithms achieve state-of-the-art results. Again, NL-DME-H yields the best average accuracy of all methods.

6.3 Cross-domain WiFi Localization

To evaluate our approach on a different domain adaptation task, we used the WiFi data set published in the 2007 IEEE ICDM Contest for domain adaptation (Yang et al., 2008). The goal here is to estimate the location of mobile devices based on the received signal strength (RSS) values from

2. The code can be downloaded from: `http://users.cecs.anu.edu.au/~basura/DA_SA/getGFKDim.m`

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	85.4 ± 0.8	87.3 ± 1.6	76.8 ± 3.2	90.9 ± 0.9	86.1 ± 3.2	77.5 ± 4.3
SVMA (Duan et al., 2012)	84.5 ± 1.3	84.9 ± 3.2	74.8 ± 4.41	91.8 ± 0.70	83.5 ± 2.3	78.5 ± 4.0
DAM (Duan et al., 2012)	85.5 ± 1.2	84.0 ± 5.0	77.4 ± 4.55	92.2 ± 0.6	83.5 ± 2.7	81.1 ± 3.9
GFK (Gong et al., 2012)	86.2 ± 0.8	89.6 ± 2.1	81.1 ± 2.5	91.2 ± 0.8	89.5 ± 2.3	84.6 ± 1.8
SA (Fernando et al., 2013)	86.3 ± 0.7	90.2 ± 3.1	87.9 ± 2.2	91.4 ± 0.6	88.4 ± 4.3	85.8 ± 3.1
TCA(Pan et al., 2011)	86.5 ± 0.8	90.1 ± 3.0	80.5 ± 3.0	91.7 ± 0.5	86.3 ± 3.8	84.6 ± 2.9
KMM(Huang et al., 2006)	85.5 ± 0.7	87.1 ± 2.4	76.5 ± 4.1	91.4 ± 0.7	86.3 ± 2.7	78.2 ± 4.9
DME-MMD	85.6 ± 0.8	89.2 ± 2.9	78.1 ± 3.2	91.1 ± 0.8	88.2 ± 2.3	82.5 ± 2.9
DME-H	86.1 ± 1.1	88.5 ± 2.0	78.9 ± 4.3	92.2 ± 0.5	88.6 ± 2.9	82 ± 3.4
NL-DME-MMD	86.2 ± 0.9	88.3 ± 3.2	79.9 ± 5.3	91.7 ± 0.53	87.5 ± 2.8	82.5 ± 2.1
NL-DME-H	85.3 ± 1.0	89.4 ± 3.8	83.3 ± 3.5	91.3 ± 0.8	88.4 ± 2.9	83 ± 0.4

Table 13: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf7 features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 14.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	88.3 ± 1.2	80.4 ± 1.0	97.3 ± 1.2	82.1 ± 1.1	76.1 ± 0.6	95.2 ± 0.9	85.2
SVMA (Duan et al., 2012)	87.6 ± 1.1	80.8 ± 0.9	96.0 ± 1.5	81.0 ± 1.6	77.0 ± 0.7	98.4 ± 1.0	84.9
DAM (Duan et al., 2012)	90.1 ± 1.1	83.1 ± 1.2	95.3 ± 1.3	81.7 ± 3.4	77.8 ± 2.4	95.6 ± 2.2	85.6
GFK (Gong et al., 2012)	89.4 ± 0.9	81.6 ± 1.1	97.3 ± 1.0	89 ± 0.9	83.6 ± 1.1	99.0 ± 0.5	88.5
SA (Fernando et al., 2013)	90 ± 0.4	82.9 ± 1.2	97.0 ± 0.9	88.2 ± 1.7	82.7 ± 0.8	98.7 ± 1.6	**89.1**
TCA(Pan et al., 2011)	91.7 ± 0.4	84 ± 0.9	97 ± 0.9	90.6 ± 1.6	81.1 ± 0.8	98.6 ± 0.9	88.5
KMM(Huang et al., 2006)	87.9 ± 1.4	80.3 ± 0.8	97 ± 1.4	82.4 ± 1.9	77.5 ± 1.4	98.5 ± 0.9	85.7
DME-MMD	88.5 ± 1.1	83.1 ± 2.6	96.6 ± 1.6	86.1 ± 3.1	79.3 ± 0.9	98.3 ± 0.9	87.2
DME-H	89 ± 1.1	82.4 ± 2.4	96.8 ± 1.0	89.7 ± 3.1	79.5 ± 2.0	98.1 ± 1.0	87.7
NL-DME-MMD	88.5 ± 1.3	84.8 ± 0.9	97.4 ± 0.8	89.5 ± 0.7	78.7 ± 1.0	98.9 ± 0.4	87.8
NL-DME-H	91.3 ± 1.0	85.8 ± 1.0	96.8 ± 1.5	90.9 ± 1.3	79.6 ± 1.7	98.2 ± 1.0	88.6

Table 14: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf7 features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. As shown by the average accuracy over all pairs in the last columns, NL-DME-H still yields competitive accuracy.

different access points. The different domains represent two different time periods during which the collected RSS values may have different distributions. The data set contains 621 labeled examples collected during time period A (i.e., the source) and 3128 unlabeled examples collected during time period B (i.e., the target). We followed the transductive setting of Pan et al. (2011), which uses all the samples from the source and 400 random samples from the target.

In this case, we report the mean Average Error Distance (AED) over 10 random selections of target samples. The AED is computed as $AED = \frac{\sum_i l(\boldsymbol{x}_i) - y_i}{N}$, where $\boldsymbol{x}_i$ is a vector of RSS values, $l(\boldsymbol{x}_i)$ is the predicted location and y_i the corresponding ground-truth location. Note that, here, all results were obtained with a nearest-neighbor classifier to follow the procedure of (Pan et al., 2011). Fig. 5 depicts the accuracy as a function of the dimensionality of the learned subspace for several

Method	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$
NO ADAPT-SVM	73 ± 1.4	83.9 ± 1.9	71.5 ± 2.6	77 ± 0.5	80.7 ± 1.8	71.8 ± 1.3
SVMA (Duan et al., 2012)	76.3 ± 0.8	85.1 ± 1.1	75.8 ± 2.3	79.5 ± 0.6	82.1 ± 1.3	73.3 ± 3.1
DAM (Duan et al., 2012)	77.1 ± 1.1	85.4 ± 1.2	77.8 ± 2.5	79.9 ± 0.6	84.2 ± 1.0	75.4 ± 2.7
GFK (Gong et al., 2012)	73.9 ± 1.3	85.2 ± 2.1	70.1 ± 1.7	79.2 ± 0.7	80.2 ± 1.9	70.5 ± 2.0
SA (Fernando et al., 2013)	75.6 ± 1	83.7 ± 1.1	72.8 ± 2.3	78.6 ± 1.0	78.5 ± 1.4	69.2 ± 2.3
TCA(Pan et al., 2011)	72.7 ± 1.5	85.3 ± 1.2	71.7 ± 1.4	79.1 ± 1.0	80.7 ± 1.6	71.3 ± 1.7
KMM(Huang et al., 2006)	75.1 ± 1.8	83.1 ± 1.5	74.3 ± 2.2	78.6 ± 1.7	81 ± 1.8	74.2 ± 2.7
DME-MMD	76.4 ± 1.6	85.4 ± 2.8	76.6 ± 3.4	78.5 ± 0.7	79.9 ± 2.1	71.9 ± 2.4
DME-H	75.3 ± 1.3	83.8 ± 1.9	74.4 ± 1.2	78.9 ± 1.2	78.3 ± 1.9	73.6 ± 1.9
NL-DME-MMD	74.9 ± 1.3	86 ± 2.1	78 ± 1.7	79.4 ± 1.4	81.2 ± 2.0	73.4 ± 1.9
NL-DME-H	74.4 ± 1.2	84.4 ± 1.6	75.2 ± 3.0	79.6 ± 1.3	78.9 ± 4.3	74 ± 2.5

Table 15: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf8 features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**. The remaining pairs and the average accuracy over all pairs are shown in Table 16.

Method	$D \to A$	$D \to C$	$D \to W$	$W \to A$	$W \to C$	$W \to D$	Avg.
NO ADAPT-SVM	80.6 ± 0.9	72.7 ± 1.1	71.4 ± 1.7	81.4 ± 1.3	76.1 ± 0.6	91.8 ± 0.9	77.6
SVMA (Duan et al., 2012)	81.8 ± 2.1	75.0 ± 1.1	83.1 ± 1.0	83.5 ± 0.4	77.4 ± 0.6	93.5 ± 1.0	80.5
DAM (Duan et al., 2012)	82.1 ± 2.1	75.7 ± 1.1	83.5 ± 0.8	82.0 ± 1.7	78.8 ± 1.8	90.2 ± 1.5	**81.1**
GFK (Gong et al., 2012)	82.5 ± 0.7	73.7 ± 1.7	78.1 ± 1.8	81.9 ± 1.3	80.6 ± 1.1	91.2 ± 1.5	78.9
SA (Fernando et al., 2013)	82.4 ± 1.1	74.9 ± 1.7	76.3 ± 2.3	81.2 ± 1.3	82.7 ± 0.8	86.9 ± 3.3	78.5
TCA(Pan et al., 2011)	80.2 ± 0.9	69.4 ± 1.2	72.8 ± 1.0	81.2 ± 1.1	83.1 ± 0.8	89.9 ± 1.9	78.1
KMM(Huang et al., 2006)	80.8 ± 1.6	75.5 ± 2.5	81.5 ± 2.8	83.5 ± 1.6	77.5 ± 1.4	89.5 ± 1.7	79.5
DME-MMD	82.7 ± 0.9	72.6 ± 2.5	78.2 ± 1.4	81.8 ± 1.3	76.8 ± 1.5	88.2 ± 1.5	79
DME-H	82.2 ± 1.4	74.1 ± 1.2	79 ± 1.1	81.8 ± 0.9	78 ± 1.1	87.8 ± 2.6	78.9
NL-DME-MMD	81.9 ± 0.7	73.7 ± 1.3	78.8 ± 2.2	82.8 ± 1.3	77.2 ± 1.0	94.1 ± 1.7	80.1
NL-DME-H	82.6 ± 1.1	74.8 ± 1.5	79.6 ± 1.9	81.3 ± 3.4	77.5 ± 1.0	90.1 ± 2.4	79.3

Table 16: Recognition accuracies on the remaining 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf8 features with Linear SVM. C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**.

subspace-based methods. As before, NL-DME-H yields the best average accuracy of all methods. Note that all our algorithms are quite robust to the choice of subspace dimension.

7. Conclusion and Future Work

In this paper, we have introduced an approach to unsupervised domain adaptation that focuses on extracting a domain-invariant representation of the source and target data. To this end, we have proposed to match the source and target distributions in a low-dimensional latent space, rather than in the original feature space. In particular, we have studied two different metrics to compare the distributions and have introduced linear and nonlinear techniques to map the data to latent spaces. Our experiments have evidenced the importance of exploiting distribution invariance for domain adaptation by revealing that our algorithms yield state-of-the-art results on several problems, with best

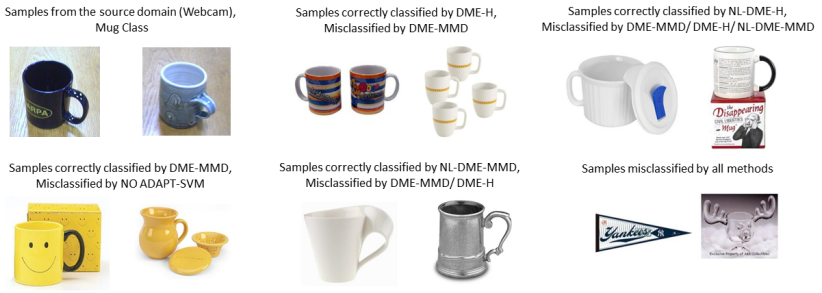


Figure 4: Misclassification examples for our algorithms when using Decaf7 features: Webcam as source, Amazon as target.

Method	$A \to C$	$C \to A$	$C \to D$	$C \to W$	$D \to C$	$W \to D$	Avg.
NO ADAPT-SVM	85.4 ± 0.8	90.9 ± 0.9	86.1 ± 3.2	77.5 ± 4.3	80.4 ± 1.0	76.1 ± 0.6	82.7
SVMA (Duan et al., 2012)	84.5 ± 1.3	91.8 ± 0.70	83.5 ± 2.3	78.5 ± 4.0	80.8 ± 0.9	77.4 ± 0.6	82.8
DAM (Duan et al., 2012)	85.5 ± 1.2	92.2 ± 0.6	83.5 ± 2.7	81.1 ± 3.9	83.1 ± 1.2	78.8 ± 1.8	84.0
GFK (Gong et al., 2012)	86.2 ± 0.8	91.2 ± 0.8	89.5 ± 2.3	84.6 ± 1.8	81.6 ± 1.1	80.6 ± 1.1	85.6
SA (Fernando et al., 2013)	86.3 ± 0.7	91.4 ± 0.6	88.4 ± 4.3	85.8 ± 3.1	82.9 ± 1.2	82.7 ± 0.8	86.3
TCA(Pan et al., 2011)	86.5 ± 0.8	91.7 ± 0.5	86.3 ± 3.8	84.6 ± 2.9	84 ± 0.9	83.1 ± 0.8	86.0
KMM(Huang et al., 2006)	85.5 ± 0.7	91.4 ± 0.7	86.3 ± 2.7	78.2 ± 4.9	80.3 ± 0.8	77.5 ± 1.4	83.2
DDC(Tzeng et al., 2014)	84.3 ± 0.5	91.3 ± 0.3	89.1 ± 0.3	85.5 ± 0.3	80.5 ± 0.2	76.9 ± 0.4	84.6
DME-MMD	85.6 ± 0.8	91.1 ± 0.8	88.2 ± 2.3	82.5 ± 2.9	83.1 ± 2.6	76.8 ± 1.5	84.6
DME-H	86.1 ± 1.1	92.2 ± 0.5	88.6 ± 2.9	82 ± 3.4	82.4 ± 2.4	78 ± 1.1	84.8
NL-DME-MMD	86.2 ± 0.9	91.7 ± 0.53	87.5 ± 2.8	82.5 ± 2.1	84.8 ± 0.9	77.2 ± 1.0	84.9
NL-DME-H	85.3 ± 1.0	91.3 ± 0.8	88.4 ± 2.9	83 ± 0.4	85.8 ± 1.0	77.5 ± 1.0	85.2

Table 17: Recognition accuracies on 6 pairs of source/target domains using the evaluation protocol of (Saenko et al., 2010) and using Decaf7 features (Comparison with Domain Deep Confusion). C**: Caltech,** A**: Amazon,** W**: Webcam,** D**: DSLR**.

performance achieved by NL-DME-H. A current limitation of our approach is the non-convexity of the optimization problems. Although, in practice, optimization on the Grassmann manifold has proven well-behaved, we intend to study if the use of other characteristic kernels in conjunction with different optimization strategies, such as the convex-concave procedure, could yield theoretical convergence guarantees within our formalism. Finally, we also plan to study the performance of other metrics, such as the KL-divergence and the Bhattacharyya distance, to compare the source and target distributions.

References

P. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

Method	$A \to D$	$D \to W$	$W \to D$
NO ADAPT-SVM	50.5 ± 2.5	87.1 ± 1.9	93.7 ± 0.8
SVMA (Duan et al., 2012)	56.7 ± 2.6	88.3 ± 1.5	95.2 ± 0.8
DAM (Duan et al., 2012)	54.9 ± 3.2	86.9 ± 1.4	84.2 ± 1.4
GFK (Gong et al., 2012)	51.2 ± 3.1	85.2 ± 2.3	93.2 ± 1.2
TCA (Pan et al., 2011)	50.1 ± 3.1	82.7 ± 1.9	91.6 ± 2.4
KMM (Huang et al., 2006)	43.7 ± 2.8	83.9 ± 2.3	93.3 ± 1.2
SA (Fernando et al., 2013)	51.8 ± 3.1	88.4 ± 1.8	96.5 ± 0.8
RG(Ganin and Lempitsky, 2015)	67.3 ± 1.7	94.0 ± 0.8	93.7 ± 1.0
DDC(Tzeng et al., 2014)	59.4 ± 0.8	92.5 ± 0.3	91.7 ± 0.8
DME-MMD	53.2 ± 2.3	86.3 ± 1.8	93.7 ± 1.5
DME-H	51.6 ± 2.0	87.4 ± 1.3	92.9 ± 1.3
NL-DME-MMD	53.5 ± 3.1	87.8 ± 1.2	95.6 ± 2.4
NL-DME-H	52.2 ± 2.4	88.3 ± 1.9	94.7 ± 2.1

Table 18: Recognition accuracies on the 3 pairs of source/target domains on Office31 data set using the evaluation protocol of (Saenko et al., 2010) and using Decaf7 features with Linear SVM. A**: Amazon,** W**: Webcam,** D**: DSLR**.

Setting	**Source Domain**	**Target Domain**
comp vs. rec	comp.windows.x & rec.sport.hockey	comp.sys.ibm.pc.hardware & rec.motorcycles
comp vs. sci	comp.windows.x & sci.crypt	comp.sys.ibm.pc.hardware & sci.med
comp vs. talk	comp.windows.x & talk.politics.mideast	comp.sys.ibm.pc.hardware & talk.politics.guns

Table 19: Experimental setting for the 20 Newsgroups data set.

Method	comp vs. rec	comp vs. sci	comp vs. talk	Avg.
NO ADAPT-SVM	79.9 ± 2.4	70.2 ± 1.6	80.6 ± 1.1	76.9
DTMKL (Duan et al., 2012)	87.8 ± 2.1	74.5 ± 1.1	$\mathbf{92.4 \pm 0.9}$	84.9
TCA (Pan et al., 2011)	85.5 ± 1.7	75.9 ± 1.0	91.9 ± 0.7	84.4
SA (Fernando et al., 2013)	80.4 ± 2.8	71.2 ± 2.1	90.1 ± 1.1	80.6
DME-MMD	87.2 ± 1.7	77 ± 1.0	91.7 ± 0.7	85.3
DME-H	87.1 ± 1.4	74.2 ± 1.9	91.6 ± 0.7	84.3
NL-DME-MMD	88.3 ± 1.8	77.3 ± 1.0	89.8 ± 0.9	85.2
NL-DME-H	$\mathbf{88.7} \pm 1.6$	$\mathbf{79.0 \pm 2.7}$	89.3 ± 0.8	**85.7**

Table 20: Recognition accuracies for the 3 source/target pairs on the 20 Newsgroups data set.

M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *Int'l Conf. on Computer Vision (ICCV)*, pages 769–776. IEEE, 2013.

M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann. Domain adaptation on the statistical manifold. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2481–2488. IEEE, 2014.

H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *Euro. Conf. on Computer Vision (ECCV)*, pages 404–417. Springer, 2006.

A. Bergamo and L. Torresani. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *Advances in Neural Information Processing Systems (NIPS)*, pages 181–189, 2010.

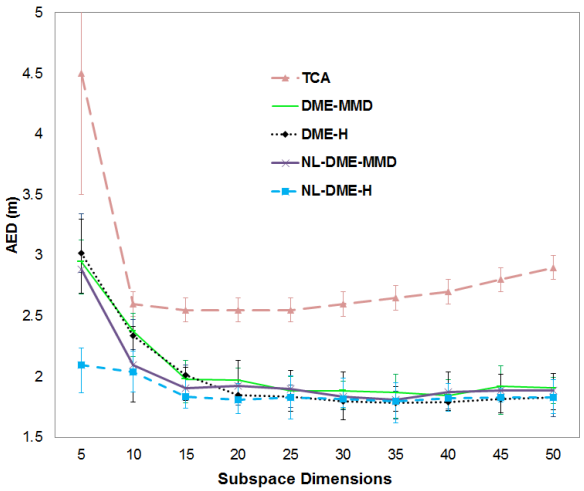


Figure 5: Comparison of the our algorithms with TCA on the task of WiFi localization. Note that NL-DME-H yields the best results, and that, in general, our algorithms are robust to the choice of subspace dimension.

J. Blitzer, D. Foster, and S. Kakade. Domain adaptation with coupled subspaces. *Journal of Machine Learning Research (JMLR)*, pages 173–181, 2011.

K. Borgwardt, A. Gretton, M. Rasch, H. Kriegel, B. Schoelkopf, and A. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Journal of Bioinformatics (BIO)*, 22(14):49–57, 2006.

L. Bruzzone and M. Marconcini. Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *The IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(5):770–787, 2010.

S. Canu and A. Smola. Kernel methods and the exponential family. *Neurocomputing*, 69(7):714–720, 2006.

K. Carter. *Dimensionality reduction on statistical manifolds*. ProQuest, 2009.

M. Chen, K. Weinberger, and J. Blitzer. Co-training for domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2456–2464, 2011.

H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research (JAIR)*, 26:101–126, 2006.

H. Daumé III, A. Kumar, and A. Saha. Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 478–486, 2010.

J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE, 2009.

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Int'l Conf. on Machine Learning (ICML)*, pages 647–655, 2014.

L. Duan, I. Tsang, D. Xu, and T. Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *Int'l Conf. on Machine Learning (ICML)*, pages 289–296. ACM, 2009a.

L. Duan, I. Tsang, D. Xu, and S. Maybank. Domain transfer svm for video concept detection. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1375–1381. IEEE, 2009b.

L. Duan, I. W Tsang, and D. Xu. Domain transfer multiple kernel learning. *The IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):465–479, 2012.

A. Edelman, T. Arias, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Mathematical Analysis (SIAM)*, 20(2):303–353, 1998.

B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Int'l Conf. on Computer Vision (ICCV)*, pages 2960–2967, 2013.

Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Int'l Conf. on Machine Learning (ICML)*, pages 1180–1189, 2015.

B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073. IEEE, 2012.

B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *Int'l Conf. on Machine Learning (ICML)*, pages 222–230, 2013.

R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Int'l Conf. on Computer Vision (ICCV)*, pages 999–1006. IEEE, 2011.

A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf. Covariate shift by kernel mean matching. *Journal of Royal. Statistical Society (JRSS)*, 3(4):5, 2009.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research (JMLR)*, 13(1):723–773, 2012a.

A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B. Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1205–1213, 2012b.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report, Calif. Inst. of Tech., 2007.

J. Hamm and D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Int'l Conf. on Machine Learning (ICML)*, pages 376–383. ACM, 2008.

M. Harandi, C. Sanderson, S. Shirazi, and B. Lovell. Kernel analysis on grassmann manifolds for action recognition. *Pattern Recognition Letters (PRL)*, 34(15):1906–1915, 2013.

R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *Int'l Journal of Computer Vision (IJCV)*, 103(3):267–305, 2013.

J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *Euro. Conf. on Computer Vision (ECCV)*, pages 702–715. Springer, 2012.

J. Huang, A. J Smola, A. Gretton, K. Borgwardt, and B. Scholkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems (NIPS)*, pages 601–608, 2006.

V. Jain and E. Learned-Miller. Online domain adaptation of a pre-trained cascade of classifiers. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 577–584. IEEE, 2011.

L. Jie, T. Tommasi, and B. Caputo. Multiclass transfer learning from unconstrained priors. In *Int'l Conf. on Computer Vision (ICCV)*, pages 1863–1870. IEEE, 2011.

Robert E Kass and Paul W Vos. *Geometrical foundations of asymptotic inference*. Wiley. com, 2011.

B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1785–1792. IEEE, 2011.

I. Kuzborskij and F. Orabona. Stability and hypothesis transfer learning. In *Int'l Conf. on Machine Learning (ICML)*, pages 942–950, 2013.

K. Muandet, D. Balduzzi, and B. Schölkopf. Domain generalization via invariant feature representation. In *Int'l Conf. on Machine Learning (ICML)*, pages 10–18, 2013.

S. Pan, I. Tsang, J. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *The IEEE Transactions on Neural Networks (TNN)*, 22(2):199–210, 2011.

A. Peter and A. Rangarajan. Shape analysis using the fisher-rao riemannian metric: Unifying shape representation and deformation. In *Biomedical Imaging (ISBI)*, pages 1164–1167. IEEE, 2006.

U. Rückert and M. Kloft. Transfer learning with adaptive regularizers. In *Euro. Conf. on Machine Learning (ECML)*, pages 65–80. 2011.

A. Ruszczynski. *Nonlinear optimization*. Princeton University press, 2006.

K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Euro. Conf. on Computer Vision (ECCV)*, pages 213–226. Springer, 2010.

B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT press, 2002.

A. Srivastava and X. Liu. Tools for application-driven linear dimension reduction. *Neurocomputing*, 67:136–160, 2005.

A. Srivastava, I. Jermyn, and S. Joshi. Riemannian analysis of probability density functions with applications in vision. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research (JMLR)*, 2:67–93, 2002.

G. Terrell. The maximal smoothing principle in density estimation. *J. of the American Statistical Association*, 85(410): 470–477, 1990.

T. Tommasi and B. Caputo. Frustratingly easy nbnn domain adaptation. In *Int'l Conf. on Computer Vision (ICCV)*, pages 897–904, 2013.

T. Tommasi and T. Tuytelaars. A testbed for cross-dataset analysis. In *Euro. Conf. on Computer Vision (ECCV Workshops)*, pages 18–31, 2014.

T. Tommasi, F. Orabona, and B. Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3081–3088. IEEE, 2010.

T. Tommasi, F. Orabona, and B. Caputo. Learning categories from few examples with multi model knowledge transfer. *The IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(5):928–941, 2014.

O. Tuzel, F. Porikli, and P. Meer. Pedestrian detection via classification on riemannian manifolds. *The IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(10):1713–1727, 2008.

E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. In *CoRR, abs/1412.3474*, 2014.

D. Xing, W. Dai, G. Xue, and Y. Yu. Bridged refinement for transfer learning. In *Euro. Conf. on Machine Learning (ECML)*, pages 324–335. Springer, 2007.

Q. Yang, J. Pan, and V. Zheng. Estimating location using wi-fi. *IEEE Intelligent Systems*, 23(1):8–13, 2008.

Journal of Machine Learning Research 17 (2016) 1-47 Submitted 5/15; Revised 1/16; Published 7/16

Monotonic Calibrated Interpolated Look-Up Tables

Maya Gupta MAYAGUPTA@GOOGLE.COM
Andrew Cotter ACOTTER@GOOGLE.COM
Jan Pfeifer JANPF@GOOGLE.COM
Konstantin Voevodski KVODSKI@GOOGLE.COM
Kevin Canini CANINI@GOOGLE.COM
Alexander Mangylov AMANGY@GOOGLE.COM
Wojciech Moczydlowski WOJTEKM@GOOGLE.COM
Alexander van Esbroeck ALEXVE@GOOGLE.COM
Google
1600 Amphitheatre Pkwy
Mountain View, CA 94301, USA

Editor: Saharon Rosset

Abstract

Real-world machine learning applications may have requirements beyond accuracy, such as fast evaluation times and interpretability. In particular, guaranteed monotonicity of the learned function with respect to some of the inputs can be critical for user confidence. We propose meeting these goals for low-dimensional machine learning problems by learning flexible, monotonic functions using calibrated interpolated look-up tables. We extend the structural risk minimization framework of lattice regression to monotonic functions by adding linear inequality constraints. In addition, we propose jointly learning interpretable calibrations of each feature to normalize continuous features and handle categorical or missing data, at the cost of making the objective non-convex. We address large-scale learning through parallelization, mini-batching, and random sampling of additive regularizer terms. Case studies on real-world problems with up to sixteen features and up to hundreds of millions of training samples demonstrate the proposed monotonic functions can achieve state-of-the-art accuracy in practice while providing greater transparency to users.

Keywords: interpretability, interpolation, look-up tables, monotonicity

1. Introduction

Many challenging issues arise when making machine learning useful in practice. Evaluation of the trained model may need to be fast. Features may be categorical, missing, or poorly calibrated. A blackbox model may be unacceptable: users may require guarantees that the function will behave sensibly for all samples, and prefer functions that are easier to understand and debug. In this paper we address these practical issues, without trading-off for accuracy.

We have found that a key interpretability issue in practice is whether the learned model can be guaranteed to be *monotonic* with respect to some features. For example, suppose the goal is to estimate the value of a used car, and one of the features is the number of km it has been driven. If all the other feature values are held fixed, we expect the value of the used car to never increase as the number of km driven increases. But a model learned from a small set of noisy samples may not, in fact, respect this prior knowledge.

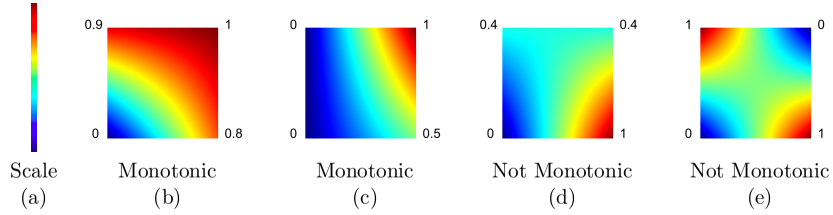


Figure 1: Example 2×2 interpolated look-up table functions over a unit square, with color scale shown in (a). Each function is defined by a 2×2 lattice with four parameters, which are the values of the function in the four corners (shown). The function is linearly interpolated from its parameters (see Figure 2 for a pictorial description of linear interpolation). The function in (b) is strictly monotonically increasing in both features, which can be verified by checking that each upper parameter is larger than the parameter below it, and that each parameter on the right is larger than the parameter to its left. The function in (c) is strictly monotonically increasing in the first feature, and monotonically increasing in the second feature (but not strictly so since the parameters on the left are both zero). The function in (d) is monotonically increasing in the first feature (one verifies this by noting that $1 \geq 0$ and $0.4 \geq 0.4$), but non-monotonic in the second feature: on the left side the function increases from $0 \rightarrow 0.4$, but on the right side the function decreases from $1 \rightarrow 0.4$. The function in (e) is a saddle function interpolating an exclusive-OR, and is non-monotonic in both features.

In this paper, we propose learning monotonic, efficient, and flexible functions by constraining and calibrating interpolated look-up tables in a structural risk minimization framework. Learning monotonic functions is difficult, and previously published work has only been illustrated on small problems (see Table 1). Our experimental results demonstrate learning flexible, guaranteed monotonic functions on more features and data than prior work, and that these functions achieve state-of-the-art performance on real-world problems.

The parameters of an interpolated look-up table are simply values of the function, regularly spaced in the input space, and these values are interpolated to compute $f(x)$ for any x. See Figures 1 and 2 for examples of 2×2 and 2×3 look-up tables and the functions produced by interpolating them. Each parameter has a clear meaning: it is the value of the function for a particular input, for a set of inputs on a regular grid. These parameters can be individually read and checked to understand the learned function's behavior.

Interpolating look-up tables is a classic strategy for representing low-dimensional functions. For example, backs of old textbooks have pages of look-up tables for one-dimensional functions like $sin(x)$, and interpolating look-up tables is standardized by the ICC Profile for the three and four dimensional nonlinear transformations needed to color manage printers (Sharma and Bala, 2002). In this paper we interpolate look-up tables defined over much

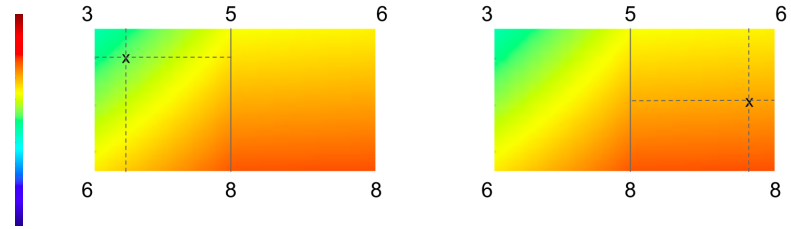


Figure 2: Figure illustrates a 3×2 lattice function and multilinear interpolation, with color scale given by (a) and parameters as shown. The lattice function shown is continuous everywhere, and differentiable everywhere except at the boundary between lattice cells, which is the vertical edge joining the middle parameters 5 and 8. As shown in (b), to evaluate $f(x)$, any x that falls in the left cell of the lattice is linearly interpolated from the parameters at the vertices of the left cell, here 6, 3, 5 and 8. Linear interpolation is linear not in x but in the lattice parameters, that is $f(x)$ is a weighted combination of the parameter values 6, 3, 5, and 8. The weights on the parameters are the areas of the four boxes formed by the dotted lines drawn orthogonally through x, with each parameter weighted by the area of the box farthest from it, so that as x moves closer to a parameter the weight on that parameter gets bigger. Because the dotted lines partition a unit square cell, the sum of these linear interpolation weights is always one. As shown in (c), samples like the marked x that fall in the right cell of the lattice are interpolated from that cell's parameters: 8, 5, 6 and 8.

larger feature spaces. Using an efficient linear interpolation method we refer to as *simplex interpolation*, the interpolation of a D-dimensional look-up table can be computed in $O(D \log D)$ time. For example, we found that interpolating a look-up table defined over $D = 20$ features takes only 2 microseconds on a standard CPU. The number of parameters in the look-up table scales as 2^D, which limits the size of D, but still enables us to learn higher-dimensional flexible monotonic functions than ever before.

Estimating the parameters of an interpolated look-up table using structural risk minimization was proposed by Garcia and Gupta (2009) and called *lattice regression*. Lattice regression can be viewed as a kernel method that uses the explicit nonlinear feature transformation formed by mapping an input $x \in [0, 1]^D$ to a vector of linear interpolation weights $\phi(x) \in \Delta^{2^D}$ over the 2^D vertices of the look-up table cell that contains x, where Δ denotes the standard simplex. Then the function is linear in these transformed features: $f(x) = \theta^T \phi(x)$. We will refer to the look-up table parameters θ as the *lattice*, and to the interpolated look-up table $f(x)$ as the *lattice function*. Earlier work in lattice regression focused on learning highly nonlinear functions over 2 to 4 features with fine-grained lattices, such as a $17 \times 17 \times 17$ lattice for modeling a color printer or super-resolution of spherical

images (Garcia et al., 2010, 2012). In this paper, we apply lattice regression to more generic machine learning problems with $D = 5$ to 16 features, and show that 2^D lattices work well for many real-world classification and ranking problems, especially when paired with jointly trained one-dimensional pre-processing functions.

We begin with a survey of related work in machine learning of interpretable and monotonic functions. Then we review lattice regression in Section 3. The main contribution is learning monotonic lattices in Section 4. We discuss efficient linear interpolation in Section 5. We propose an interpretable torsion lattice regularizer in Section 6. We propose jointly learning one-dimensional calibration functions in Section 7, and consider two strategies for supervised handling of missing data for lattice regression in Section 8. In Section 9, we consider strategies for speeding up training and handling large-scale problems and large-scale constraint-handling. A series of case studies in Section 10 experimentally explore the paper's proposals, and demonstrate that monotonic lattice regression achieves similar accuracy as a random forest, and that monotonicity is a common issue that arises in many different applications. The paper ends with conclusions and open questions in Section 11.

2. Related Work

We give a brief overview of related work in interpretable machine learning, then survey related work in learning monotonic functions.

2.1 Related Work in Interpretable Machine Learning

Two key themes of the prior work on interpretable machine learning are *(i)* interpretable function classes, and *(ii)* preferring simpler functions within a function class.

2.1.1 Interpretable Function Classes

The function classes of decision trees and rules are generally regarded as relatively interpretable. Naïve Bayes classifiers can be interpreted in terms of weights of evidence (Good, 1965; Spiegelhalter and Knill-Jones, 1984). Similarly, linear models form an interpretable function class in that the parameters dictate the relative importance of each feature. Linear approaches can be generalized to sum nonlinear components, as in generalized additive models (Hastie and Tibshirani, 1990) and some kernel methods, while still retaining some of their interpretable aspects.

The function class of interpolated look-up tables is interpretable in that the function's parameters are the look-up table values, and so are semantically meaningful: they are simply examples of the function's output, regularly spaced in the domain. Given two look-up tables with the same structure and the same features, one can analyze how their functions differ by analyzing how the look-up table parameters differ. Analyzing which parameters change by how much can help answer questions like "If I add training examples and re-train, what changes about the model?"

2.1.2 Prefer Simpler Functions

Another body of work focuses on choosing simpler functions within a function class, optimizing an objective of the form: minimize empirical error and maximize simplicity, where

simplicity is usually defined as some manifestation of Occam's Razor or variant of Kolmogorov complexity. For example, Ishibuchi and Nojima (2007) minimize the number of fuzzy rules in a rule set, Osei-Bryson (2007) prunes a decision tree for interpretability, Rätsch et al. (2006) finds a sparse convex combination of kernels for a multi-kernel support vector machine, and Nock (2002) prefers smaller committees of ensemble classifiers. Similarly, Garcia et al. (2009) measure the interpretability of rule-based classifiers in terms of the number of rules and number of features used. More generally, this category of interpretability includes model selection criteria like the Bayesian information criterion and Akaike information criterion (Hastie et al., 2001), sparsity regularizers like sparse linear regression models, and feature selection methods. Other approaches to simplicity may include simplified structure in graphical models or neural nets, such as the structured neural nets of Strannegaard (2012).

While sparsity-based approaches to interpretability can provide regularization that reduces over-fitting and hence increases accuracy, it has also been noted that such strategies may create a trade-off between interpretability and accuracy (Casillas et al., 2002; Nock, 2002; Yu and Xiao, 2012; Shukla and Tripathi, 2012). We hypothesize this occurs when the assumed simpler structure is a poor model of the true function.

Monotonicity is another way to choose a semantically simpler function to increase interpretability (and regularize). Our case studies in Section 10 illustrate that when applied to problems where monotonicity is warranted true, we do not see a trade-off with accuracy.

2.2 Related Work in Monotonic Functions

A function $f(x)$ is monotonically increasing with respect to feature d if $f(x_i) \geq f(x_j)$ for any two feature vectors $x_i, x_j \in \mathbb{R}^D$ where $x_i[d] \geq x_j[d]$ and $x_i[m] = x_j[m]$ for $m \neq d$.

A number of approaches have been proposed for enforcing and encouraging monotonicity in machine learning. The computational complexity of these algorthims tends to be high, and most methods scale poorly in the number of features D and samples n, as summarized in Table 1.

We detail the related work in the following sections organized by the type of machine learning, but these methods could instead be organized by strategy, which mostly falls into one of four categories:

1. Constrain a more flexible function class to be monotonic, such as linear functions with positive coefficients, or a sigmoidal neural network with positive weights.
2. Post-process by pruning or reducing monotonicity violations after training.
3. Penalize monotonicity violations by pairs of samples or sample derivatives when training.
4. Re-label samples to be monotonic before training.

2.2.1 Monotonic Linear and Polynomial Functions

Linear functions can be easily constrained to be monotonic in certain inputs by requiring the corresponding slope coefficients to be non-negative, but linear functions are not sufficiently flexible for many problems. Polynomial functions (equivalently, linear functions with pre-defined crosses of features) can also be easily forced to be monotonic by requiring all coefficients to be positive. However, this is only a sufficient and not necessary condition: there are monotonic polynomials whose coefficients are not all positive. For example, consider the simple case of second degree multilinear polynomials defined over the unit square $f : [0, 1]^2 \to \mathbb{R}$ such that:

$$f(x) = a_0 + a_1 x[0] + a_2 x[1] + a_3 x[0]x[1]. \tag{1}$$

Restricting the function to a bounded domain the domain $x \in [0, 1]^2$ and forcing the derivative to be positive over that domain, one sees that the complete set of monotonic functions of the form (1) on the unit square is described by four linear inequalities:

$$\begin{array}{ll} a_1 > 0 & a_2 > 0 \\ a_1 + a_3 > 0 & a_2 + a_3 > 0. \end{array}$$

The general problem of checking whether a particular choice of polynomial coefficients produces a monotonic function requires checking whether the polynomial's derivative (also a polynomial) is positive everywhere, which is equivalent to checking if the derivative has any real roots, which can be computationally challenging (see, for example, Sturm's theorem for details).

Functions of the form (1) can be equivalently expressed as a 2×2 lattice interpolated with multilinear interpolation, but as we will show in Section 4, with this alternate parameterization it is easier to check and enforce the complete set of monotonic functions.

2.2.2 Monotonic Splines

In this paper we extend lattice regression, which is a spline method with fixed knots on a regular grid and a linear kernel (Garcia et al., 2012), to be monotonic. There have been a number of proposals to learn monotonic one-dimensional splines. For example, building on Ramsay (1998), Shively et al. (2009) parameterize the set of all smooth and strictly monotonic one-dimensional functions using an integrated exponential form $f(x) = a + \int_0^x e^{b+u(t)}dt$, and showed better performance than the monotone functions estimators of Neelon and Dunson (2004) and Holmes and Heard (2003) for smooth functions. In other related spline work, Villalobos and Wahba (1987) considered smoothing splines with linear inequality constraints, but did not address monotonicity.

2.2.3 Monotonic Decision Trees and Forests:

Stumps and forests of stumps are easily constrained to be monotonic. However, for deeper or broader trees, all pairs of leaves must be checked to verify monotonicity (Potharst and Feelders, 2002b). Non-monotonic trees can be pruned to be monotonic using various strategies that iteratively reduce the non-monotonic branches (Ben-David, 1992; Potharst and Feelders, 2002b). Monontonicity can also be encouraged during tree construction by penalizing the splitting criterion to reduce the number of non-monotonic leaves a split would create (Ben-David, 1995). Potharst and Feelders (2002a) achieved completely flexible monotonic trees using a strategy akin to bogosort (Gruber et al., 2007): train many trees on different random subsets of the training samples, then select one that is monotonic.

	Method	Monotonicity Strategy	Guaranteed Monotonic?	Max D	Max n
Archer and Wang (1993)	neural net	constrain function	yes	2	50
Wang (1994)	neural net	constrain function	yes	1	150
Mukarjee and Stern (1994)	kernel estimate	post-process	yes	2	2447
Ben-David (1995)	tree	penalize splits	yes	8	125
Sill and Abu-Mostafa (1997)	neural net	penalize pairs	no	6	550
Sill (1998)	neural net	constrain function	yes	10	196
Kay and Ungar (2000)	neural net	constrain function	yes	1	100
Potharst and Feelders (2002a)	tree	randomize	yes	8	60
Potharst and Feelders (2002b)	tree	prune	yes	11	1090
Spouge et al. (2003)	isotonic regression	constrain	yes	2	100,000
Duivesteijn and Feelders (2008)	k-NN	re-label samples	no	12	768
Lauer and Bloch (2008)	svm	sample derivatives	no	none	none
Dugas et al. (2000, 2009)	neural net	constrain function	yes	4	3434
Shively et al. (2009)	spline	constrain function	yes	1	100
Kotlowski and Slowinski (2009)	rule-based	re-label samples	yes	11	1728
Daniels and Velikova (2010)	neural net	constrain function	yes	6	174
Riihimäki and Vehtari (2010)	Gaussian process	sample derivatives	no	7	1222
Qu and Hu (2011)	neural net	derivatives / constrain	yes	1	30
Neumann et al. (2013)	neural net	sample derivatives	no	3	625

Table 1: Some related work in learning monotonic functions. Many of these methods *guarantee* a monotonic solution, but some only *encourage* monotonicity. The last two columns gives the largest number of features D and the largest number of samples n used in any of the experiments in that paper (generally not the same experiment).

2.2.4 Monotonic Support Vector Machines

With a linear kernel, it may be easy to check and enforce monotonicity of support vector machines, but for nonlinear kernels it will be more challenging. Lauer and Bloch (2008) encouraged support vector machines to be more monotonic by constraining the derivative of the function at the training samples. Riihimäki and Vehtari (2010) used the same strategy to encourage more monotonic Gaussian processes.

2.2.5 Monotonic Neural Networks

In perhaps the earliest work on monotonic neural networks, Archer and Wang (1993) adaptively down-weighted samples during training whose gradient updates would violate monotonicity, to produce a positive weighted neural net. Other researchers explicitly proposed constraining the weights to be positive in a single hidden-layer neural network with the sigmoid or other monotonic nonlinear transformation (Wang, 1994; Kay and Ungar, 2000; Dugas et al., 2000, 2009; Minin et al., 2010). Dugas et al. (2009) showed with simulations of four features and 400 training samples that both bias and variance were reduced by enforcing monotonicity. However, Daniels and Velikova (2010) showed this approach requires D hidden layers to arbitrarily approximate any D-dimensional monotonic function. In addition to a general proof, they provide a simple and realistic example of a two-dimensional monotonic function that cannot be fit with one hidden layer and positive weights.

Abu-Mostafa (1993) and Sill and Abu-Mostafa (1997) proposed regularizing a function to be more monotonic by penalizing squared deviations in monotonicity for virtual pairs of input samples that are added for this purpose. Unfortunately, it generally does not guarantee monotonicity everywhere, only with respect to those virtual input pairs. (And in fact, to guarantee monotonicity for the sampled pairs, an exact penalty function would be needed with a sufficiently large regularization parameter to ensure the regularization was equivalent to a constraint).

Lauer and Bloch (2008), Riihimäki and Vehtari (2010), and Neumann et al. (2013) encouraged *extreme learning machines* to be more monotonic by constraining the derivative of the function to be positive for a set of sampled points.

Qu and Hu (2011) did a small-scale comparison of encouraging monotonicity by constraining input pairs to be monotonic, versus encouraging monotonic neural nets by constraining the function's derivatives at a subset of samples (analogous to Lauer and Bloch (2008)), versus using a sigmoidal function with positive weights. They concluded the positive-weight sigmoidal function is best.

Sill (1998) proposed a guaranteed monotonic neural network with two hidden layers by requiring the first linear layer's weights to be positive, using hidden nodes that take the maximum of groups of first layer variables, and a second hidden layer that takes the minimum of the maxima. The resulting surface is piecewise linear, and as such can represent any continuous differentiable function arbitrarily well. The resulting objective function is not strictly convex, but the authors propose training such monotonic networks using gradient descent where samples are associated with one active hyperplane at each iteration. Daniels and Velikova (2010) generalized this approach to handle the "partially monotonic" case that the function is only monotonic with respect to some features.

2.2.6 Isotonic Regression and Monotonic Nearest Neighbors

Isotonic regression re-labels the input samples with values that are monotonic and close to the original labels. These monotonically re-labeled samples can then be used, for example, to define a monotonic piecewise constant or piecewise linear surface. This is an old approach; see Barlow et al. (1972) for an early survey. Isotonic regression can be solved in $O(n)$ time if monotonicity implies a total ordering of the n samples. But for usual multi-dimensional machine learning problems, monotonicity implies only a partial order, and solving the n-parameter quadratic program is generally $O(n^4)$, and $O(n^3)$ for two-dimensional samples (Spouge et al., 2003). Also problematic for large n is the $O(n)$ evaluation time for new samples.

Mukarjee and Stern (1994) proposed a suboptimal monotonic kernel regression that is computationally easier to train than isotonic regression. It computes a standard kernel estimate, then locally upper and lower bounds it to enforce monotonicity.

The *isotonic separation* method of Chandrasekaran et al. (2005) is like the work of Abu-Mostafa (1993) in that it penalizes violations of monotonicity by pairs of training samples. Like isotonic regression, the output is a re-labeling of the original samples, the solution is at least $O(n^3)$ in the general case, and evaluation time is $O(n)$.

Ben-David et al. (1989); Ben-David (1992) constructed a monotonic rule-based classifier by sequentially adding training examples (each of which defines a rule) that do not violate monotonicity restrictions.

Duivesteijn and Feelders (2008) proposed re-labeling samples before applying nearest neighbors based on a monotonicity violation graph with the training examples at the vertices. Coupled with a proposed modified version of k-NN, they can enforce monotonic outputs. Similar pre-processing of samples can be used to encourage any subsequently trained classifier to be more monotonic (Feelders, 2010).

Similarly, Kotlowski and Slowinski (2009) try to solve the isotonic regression problem to re-label the dataset to be monotonic, then fit a monotonic ensemble of rules to the re-labeled data, requiring zero training error. They showed overall better performance than the ordinal learning model of Ben-David et al. (1989) and isotonic separation (Chandrasekaran et al., 2005).

3. Review of Lattice Regression

Before proposing *monotonic* lattice regression, we review lattice regression (Garcia and Gupta, 2009; Garcia et al., 2012). Key notation is listed in Table 2.

Let $M_d \in \mathbb{N}$ be a hyperparameter specifying the number of vertices in the look-up table (that is, lattice) for the dth feature. Then the lattice is a regular grid of $M \triangleq M_1 \times M_2 \times \ldots M_D$ parameters placed at natural numbers so that the lattice spans the hyper-rectangle $\mathcal{M} \triangleq [0, M_1 - 1] \times [0, M_2 - 1] \times \ldots [0, M_D - 1]$. See Figure 1 for examples of 2×2 lattices, and Figure 2 for an example 3×2 lattice. For machine learning problems we find $M_d = 2$ for all d to often work well in practice, as detailed in the case studies in Section 10. For image processing applications with only two to four features, much larger values of M_d were needed (Garcia et al., 2012).

D	number of features
n	number of samples
$M_d \in \mathbb{N}$	number of vertices in the lattice along the dth feature
$M \in \mathbb{N}$	total number of vertices in the lattice: $M = \prod_d M_d$
$\mathcal{M}$	hyper-rectangular span of the lattice: $[0, M_1 - 1] \times \ldots \times [0, M_D - 1]$
x_i	ith training sample with D components. Domain depends on section.
$y_i \in \mathbb{R}$	ith training sample label
$x[d]$	dth component of feature vector x
$\phi(x) \in [0,1]^M$	linear interpolation weights for x
$\theta \in \mathbb{R}^M$	lattice values (parameters)
$v_j \in \{0,1\}^D$	jth vertex of a 2^D lattice

Table 2: Key Notation

The feature values are assumed to be bounded and linearly scaled to fit the lattice, so that the dth feature vector value $x[d]$ lies in $[0, M_d - 1]$. (We propose learning non-linear scalings of features jointly with the lattice parameters in Section 7.)

Lattice regression is a kernel method that maps $x \in \mathcal{M}$ to a transformed feature vector $\phi(x) \in [0,1]^M$. The values of $\phi(x)$ are the interpolation weights for x for the 2^D indices corresponding to the 2^D vertices of the hypercube surrounding x; for all other indices, $\phi(x) = 0$.

The function $f(x)$ is linear in $\phi(x)$ such that $f(x) = \theta^T \phi(x)$. That is, the function parameters θ each correspond to a vertex in the lattice, and $f(x)$ linearly interpolates the θ for the lattice cell containing x.

Before reviewing the lattice regression objective for learning the parameters θ, we review standard multilinear interpolation to define $\phi(x)$.

3.1 Multilinear Interpolation

Multilinear interpolation is the multi-dimensional generalization of the familiar bilinear interpolation that is commonly used to up-sample images. See Figure 2 for a pictorial explanation.

For notational simplicity, we assume a 2^D lattice such that $x \in [0,1]^D$. For multi-cell lattices, the same math and logic is applied to the lattice cell containing the x. Denote the kth component of $\phi(x)$ as $\phi_k(x)$. Let $v_k \in \{0,1\}^D$ be the kth vertex of the unit hypercube. The multilinear interpolation weight on the vertex v_k is

$$\phi_k(x) = \prod_{d=0}^{D-1} x[d]^{v_k[d]} (1 - x[d])^{1 - v_k[d]}. \tag{2}$$

Note the exponents in (2) are v_k and $1 - v_k[d]$, which either equal 0 and 1, or equal 1 and 0, so these exponents act like selectors that multiply in either $x[d]$ or $1 - x[d]$ for each

dimension d. Equivalently, one can write

$$\phi_k(x) = \prod_{i=0}^{D-1} \left((1 - \texttt{bit}[i,k]) \, (1 - x[i]) + \texttt{bit}[i,k] x[i] \right), \tag{3}$$

where $\texttt{bit}[i,k] \in \{0,1\}$ denotes the ith bit of vertex v_k, and can be computed $\texttt{bit}[i,k] = (k \gg i) \,\&1$ using bitwise arithmetic.

The resulting $f(x) = \theta^T \phi(x)$ is a multilinear polynomial over each lattice cell. For example, a 2×2 lattice interpolated with multilinear interpolation gives:

$$f(x) = \theta[0](1-x[0])(1-x[1]) + \theta[1]x[0](1-x[1]) + \theta[2](1-x[0])x[1] + \theta[3]x[0]x[1]. \tag{4}$$

Expanding (4), one sees it is a different parameterization of the multilinear function given in (1), where the parameter vectors are related by a linear matrix transform: $a = T\theta$ for $T \in \mathbb{R}^{4\times 4}$. But the θ parameterization has the advantage that each parameter is the function value for a feature vector at the vertex of the lattice (see Figure 1), and as we show in Section 4, makes it easier to learn the complete set of monotonic functions.

The linear interpolation is applied per lattice cell. At lattice cell boundaries the resulting function is continuous, but not differentiable. The overall function is piecewise polynomial, and hence a spline, and can be equivalently formulated using a linear basis function. Higher-order basis functions like the popular cubic spline will lead to smoother and potentially slightly more accurate functions (Garcia et al., 2012). However, higher-order basis functions destroy the interpretable localized effect of the parameters, and increase the computational complexity.

The multilinear interpolation weights are just one type of linear interpolation. In general, linear interpolation weights are defined as solutions to the system of $D+1$ equations:

$$\sum_{k=0}^{2^D} \phi_k(x) v_k = x \text{ and } \sum_{k=0}^{2^D} \phi_k(x) = 1. \tag{5}$$

This system of equations is under-determined and has many solutions for an x in the convex hull of a lattice cell. The multilinear interpolation weights given in (2) are the maximum entropy solution to (5) (Gupta et al., 2006), and thus have good noise averaging and smoothness properties compared to other solutions. We discuss a more efficient linear interpolation in Sec. 5.2.

3.2 The Lattice Regression Objective

Consider the standard supervised machine learning set-up of a training set of randomly sampled pairs $\{(x_i, y_i)\}$ pairs, where $x_i \in \mathcal{M}$ and $y_i \in \mathbb{R}$, for $i = 1, \ldots, n$. Historically, people created look-up tables by first fitting a function $h(x)$ to the $\{x_i, y_i\}$ using a regression algorithm such as a neural net or local linear regression, and then evaluating $h(x)$ on a regular grid to produce the look-up table values (Sharma and Bala, 2002). However, even if they fit the function to minimize empirical risk on the training samples, they did not minimize the *actual* empirical risk because these approaches did not take into account that the trained look-up table would be interpolated at run-time, and this interpolation changes the error on the training samples.

Garcia and Gupta (2009) proposed directly optimizing the look-up table parameters θ to minimize the actual empirical error between the training labels and the interpolated look-up table:

$$\arg\min_{\theta \in \mathbb{R}^M} \sum_{i=1}^{n} \ell(y_i, \theta^T \phi(x_i)) + R(\theta), \tag{6}$$

where ℓ is a loss function such as squared error, $\phi(x_i) \in [0,1]^M$ is the vector of linear interpolation weights over the lattice for training sample x_i (detailed in Section 3.1 and Sec. 5.2), $f(x_i) = \theta^T \phi(x_i)$ is the linear interpolation of x_i from the look-up table parameters θ, and $R(\theta)$ is a regularizer on the lattice parameters. In general, we assume the loss ℓ and regularizer R are convex functions of θ so that solving (6) is a convex optimization. Prior work focused on squared error loss, and used graph regularizers $R(\theta)$ of the form $\theta^T K \theta$ for some PSD matrix K, in which case (6) has a closed-form solution which can be computed with sparse matrix inversions (Garcia and Gupta, 2009; Garcia et al., 2010, 2012).

4. Monotonic Lattices

In this section we propose constraining lattice regression to learn monotonic functions.

4.1 Monotonicity Constraints For a Lattice

In general, simply checking whether a nonlinear function is monotonic can be quite difficult (see the related work in Section 2.2). But for a linearly interpolated look-up table, checking for monotonicity is relatively easy: if the lattice values increase in a given direction, then the function increases in that direction. See Figure 1 for examples. Specifically, one must check that $\theta_s > \theta_r$ for each pair of adjacent look-up table parameters θ_r and θ_s. If all features are specified to be monotonic for a 2^D lattice, this results in $D2^{D-1}$ pairwise linear inequality constraints to check.

These same pairwise linear inequality constraints can be imposed when learning the parameters θ to ensure a monotonic function is learned. The following result establishes these constraints are sufficient and necessary for a 2^D lattice to be monotonically increasing in the dth feature (the result extends trivially to larger lattices):

Lemma 1 (Monotonicity Constraints) *Let $f(x) = \theta^T \phi(x)$ for $x \in [0,1]^D$ and $\phi(x)$ given in (2). The partial derivative $\partial f(x)/\partial x[d] > 0$ for fixed d and any x iff $\theta_{k'} > \theta_k$ for all k, k' such that $v_k[d] = 0$, $v_{k'}[d] = 1$ and $v_k[m] = v_{k'}[m]$ for all $m \neq d$.*

Proof First we show the constraints are necessary to ensure monotonicity. Consider the function values $f(v_k)$ and $f(v_{k'})$ for some adjacent pair of vertices $v_k, v_{k'}$ that differ only in the dth feature. For $f(v_k)$ and $f(v_{k'})$, all of the interpolation weight falls on θ_k or $\theta_{k'}$ respectively, such that $f(v_k) = \theta_k$ and $f(v_{k'}) = \theta_{k'}$. So $\theta_{k'} > \theta_k$ is necessary for $\partial f(x)/\partial x[d] > 0$ everywhere.

Next we show the constraints are sufficient to ensure monotonicity. Pair the terms in the interpolation $f(x) = \theta^T \phi(x)$ corresponding to adjacent parameters $\theta_k, \theta_{k'}$ so that for

each k, k' it holds that $v_k[d] = 0, v_{k'}[d] = 1, v_k[m] = v_{k'}[m]$ for $m \neq d$:

$$\begin{aligned} f(x) &= \sum_{k,k'} \theta_k \phi_k(x) + \theta_{k'} \phi_{k'}(x), \text{ then expand } \phi_k(x) \text{ and } \phi_{k'}(x) \text{ using (2) :} \\ &= \sum_{k,k'} \alpha_k \left(\theta_k x[d]^{v_k[d]} (1 - x[d])^{(1-v_k[d])} + \theta_{k'} x[d]^{v_{k'}[d]} (1 - x[d])^{(1-v_{k'}[d])} \right), \\ &\quad \text{where } \alpha_k \text{ is the product of the } m \neq d \text{ terms in (2) that are the same for } k \text{ and } k', \\ &= \sum_{k,k'} \alpha_k \left(\theta_k (1 - x[d]) + \theta_{k'} x[d] \right) \text{ by the definition of } v_k \text{ and } v_{k'}. \end{aligned} \tag{7}$$

The partial derivative of (7) is $\frac{\partial f(x)}{\partial x[d]} = \sum_{k,k'} \alpha_k (\theta_{k'} - \theta_k)$. Because each $\alpha_k \in [0, 1]$, it is sufficient that $\theta_{k'} > \theta_k$ for each k, k' pair to guarantee this partial is positive for all x. ■

4.2 Monotonic Lattice Regression Objective

We relax strict monotonicity to monotonicity by allowing equality in the adjacent parameter constraints (for an example, see the second function from the left in Figure 1). Then the set of pairwise constraints can be expressed as $A\theta \leq 0$ for the appropriate sparse matrix A with one 1 and -1 per row of A, and one row per constraint. Each feature can independently be left unconstrained, or constrained to be either monotonically increasing or decreasing by the specification of A.

Thus the proposed monotonic lattice regression objective is convex with linear inequality constraints:

$$\arg\min_{\theta} \sum_{i=1}^{n} \ell(y_i, \theta^T \phi(x_i)) + R(\theta), \text{ s.t. } A\theta \leq b. \tag{8}$$

Additional linear constraints can be included in $A\theta \leq b$ to also constrain the fitted function in other practical ways, such as $f(x) \in [0, 1]$ or $f(x) \geq 0$.

The approach extends to the standard learning to rank from pairs problem (Liu, 2011), where the training data is pairs of samples x_i^+ and x_i^- and the goal is to learn a function such that $f(x_i^+) \geq f(x_i^-)$ for as many pairs as possible. For this case, the monotonic lattice regression objective is:

$$\arg\min_{\theta} \sum_{i=1}^{n} \ell(1, \theta^T \phi(x_i^+) - \theta^T \phi(x_i^-)) + R(\theta), \text{ s.t. } A\theta \leq b. \tag{9}$$

The loss functions in (6), (8) and (9) all have the same form, for example, squared loss $\ell(y, z) = (y - z)^2$, hinge loss $\ell(y, z) = \max(0, 1 - yz)$, or logistic loss $\ell(y, z) = \log(1 + \exp(y - z))$.

5. Faster Linear Interpolation

Interpolating a look-up table has long been considered an efficient way to specify and evaluate a low-dimensional non-linear function (Sharma and Bala, 2002; Garcia et al., 2012).

But computing linear interpolation weights with (3) requires $O(D)$ operations for each of the 2^D interpolation weights, for a total cost of $O(D2^D)$ to evaluate a sample. In Section 5.1, we show the multilinear interpolation weights of (3) can be computed in $O(2^D)$ operations. Then, in Section 5.2, we review and analyze a different linear interpolation that we refer to as *simplex* interpolation that takes only $O(D \log D)$ operations.

5.1 Fast Multilinear Interpolation

Much of the computation in (3) can be shared between the different weights. In Algorithm 1 we give a dynamic programming solution that loops D times, where the dth loop takes 2^d time, so in total there are $\sum_{d=0}^{D-1} 2^d = O(2^D)$ operations.

Algorithm 1 Computes the multilinear interpolation weights and corresponding vertex indices for a unit lattice cell $[0, 1]^D$ and an $x \in [0, 1]^D$. Let the lattice parameters be indexed such that $s_d = 2^d$ is the difference in the indices of the parameters corresponding to any two vertices that are adjacent in the dth dimension, for example, for the 2×2 lattice, order the vertices [0 0], [1 0], [0 1], [1 1] and index the corresponding lattice parameters in that order.

```
CalculateMultilinearInterpolationWeightsAndParameterIndices(x)
1  Initialize indices[] = [0], weights[] = [1]
2  For d = 0 to D − 1:
3      For k = 0 to 2^d − 1:
4          Append s_d + indices[k] to indices
5          Append x[d] × weights[k] to weights
6          Update weights[k] = (1 − (x[d])) × weights[k]
7  Return indices and weights
```

The following lemma establishes the correctness of Algorithm 1.

Lemma 2 (Fast Multilinear Interpolation) *Under its assumptions, Algorithm 1 returns the indices of the 2^D parameters corresponding to the vertices of the lattice cell containing x:*

$$\texttt{indices}[k] = \sum_{d=0}^{D-1} \left(\lfloor x[d] \rfloor + \texttt{bit}_i(k) \right) s_d, \text{ for } k = 1, 2, \ldots, 2^D \tag{10}$$

and the corresponding 2^D multilinear interpolation weights given by (3).

Proof At the end of the D'th iteration over the dimension in Algorithm 1:

$$\begin{aligned} \texttt{size}\,(\texttt{indices}) = \texttt{size}\,(\texttt{weights}) &= 2^{D'+1} \\ \texttt{indices}[k] &= \sum_{d=0}^{D'} \left(\lfloor x[d] \rfloor + \texttt{bit}_d(k) \right) s_d \\ \texttt{weights}[k] &= \prod_{d=0}^{D'} \left((1 - \texttt{bit}_d(k)) \left(1 - (x[d] - \lfloor x_d \rfloor)\right) + \texttt{bit}_d(k)(x[d] - \lfloor x_d \rfloor) \right). \end{aligned}$$

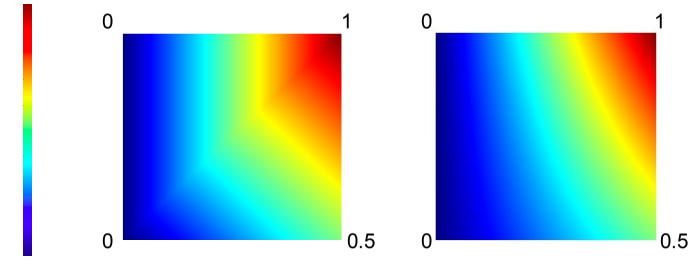


Figure 3: Illustration of two different linear interpolations of the same 2×2 look-up table with parameters $0, 0, 0.5$ and 1. The simplex interpolation splits the unit square into two simplices (the upper and lower triangle) and interpolates within each. The function is continuous because the points along the diagonal are interpolated from only the two corner vertices, and the interpolated function is linear over each simplex. Both interpolations produce monotonic functions over both features.

The above holds for the $D' = 1$ case by the initialization and inspection of the loop. It is straightforward to verify that if the above hold for D', then they also hold for $D' + 1$. Then by induction it holds for $D' = D - 1$, as claimed. ■

5.2 Simplex Linear Interpolation

For speed, we propose using a more efficient linear interpolation for lattice regression that linearly interpolates each x from only $D+1$ of the 2^D surrounding vertices. Many different linear interpolation strategies have been proposed to interpolate look-up tables using only a subset of the 2^D vertices (for a review, see Kang (1997)). However, most such strategies suffer from being too computationally expensive to determine the subset of vertices needed to interpolate a given x. The wonder of *simplex interpolation* is that it takes only $O(D \log D)$ operations to determine the $D+1$ vertices needed to interpolate any given x, and then only $O(D)$ operations to interpolate the identified $D+1$ vertices. An illustrative comparison of simplex and multilinear interpolation is given in Figure 3 for the same lattice parameters.

Simplex interpolation was proposed in the color management literature by Kasson et al. (1993), and independently later by Rovatti et al. (1998). Simplex interpolation is also known as the *Lovasz extension* in submodular optimization, where it is used to extend a function defined on the vertices of a unit hypercube to be defined on its interior (Bach, 2013).

After reviewing how simplex interpolation works, we show in Section 5.2.3 that it requires the same constraints for monotonicity as multilinear interpolation, and then we discuss how its rotational dependence impacts its use for machine learning in Section 5.2.4. We give example runtime comparisons in Section 10.7.

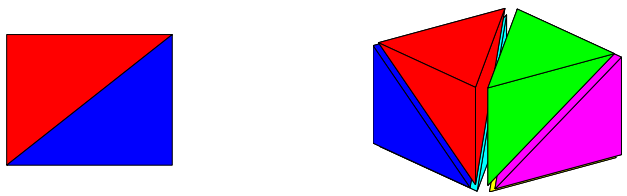


Figure 4: Simplex interpolation decomposes the D-dimensional unit hypercube into $D!$ simplices. **Left:** For the unit square, there are two simplices, one is defined by the three vertices [0 0], [0 1], and [1 1], and the other is defined by the three vertices [0 0], [1 0], and [1 1]. **Right:** For the unit cube there are $3! = 6$ simplices, each defined by four vertices. The first has vertices: [0 0 0], [0 0 1], [0 1 1] , [1 1 1]. The second has vertices: [0 0 0], [0 0 1], [1 0 1], [1 1 1]. And so on. All six simplices have vertices [0 0 0] and [1 1 1], and thus share the diagonal between those two vertices.

5.2.1 Partitioning of the Unit Hypercube Into Simplices

Simplex interpolation implicitly partitions the hypercube into the set of D! congruent simplices that satisfy the following: each simplex includes the all 0's vertex, one vertex is all zeros but has a single 1, one vertex is all zeros but has two 1's, and so on, ending with one vertex that is all 1's, for a total of $D+1$ vertices in each simplex. Figure 4 shows the partitioning for the $D = 2$ and $D = 3$ unit hypercubes.

This decomposition can also be described by the hyperplanes $x_k = x_r$ for $1 \le k \le r \le D$ (Schimdt and Simon, 2007). Knop (1973) discussed this decomposition as a special case of Eulerian partitioning of the hypercube, and Mead (1979) showed this is the smallest possible equivolume decomposition of the unit hypercube.

5.2.2 Simplex Interpolation

Given $x \in [0,1]^D$, the $D+1$ vertices that specify the simplex that contains x can be computed in $O(D \log D)$ operations by sorting the D values of the feature vector x, and then the dth simplex vertex has ones in the first d sorted components of x. For example, if x =[.8 .2 .3], the $D+1$ vertices of its simplex are [0 0 0], [1 0 0], [1 0 1], [1 1 1].

Let V be the $D+1$ by D matrix whose dth row is the dth vertex of the simplex containing x. Then the simplex interpolation weights $\psi(x)$ must satisfy the linear interpolation equations given in (5) such that $\begin{bmatrix} V^T \\ \mathbf{1}^T \end{bmatrix} \psi(x) = \begin{bmatrix} x \\ 1 \end{bmatrix}$. Thus $\psi(x) = \begin{bmatrix} V^T \\ \mathbf{1}^T \end{bmatrix}^{-1} \begin{bmatrix} x \\ 1 \end{bmatrix}$, where because of the highly structured nature of the simplex decomposition the required inverse always exists, and has a simple form such that $\psi(x)$ is the vector of differences of sequential sorted components of x. For example, for a $2 \times 2 \times 2$ lattice, and an x such that $x[0] > x[1] > x[2]$, the simplex interpolation weights $\psi(x) = [1 - x[0], x[0] - x[1], x[1] - x[2], x[2]]$ on the vertices

$[0,0,0],[1,0,0],[1,1,0],[1,1,1]$, respectively. The general formula is detailed in Algorithm 2; for more mathematical details see Rovatti et al. (1998).

Algorithm 2 Computes the simplex interpolation weights and corresponding vertex indices for a unit lattice cell $[0,1]^D$ and an $x \in [0,1]^D$. Let the lattice parameters be indexed such that $s_d = 2^d$ is the difference in the indices of the parameters corresponding to any two vertices that are adjacent in the dth dimension, for example, for the 2×2 lattice, order the vertices [0 0], [1 0], [0 1], [1 1] and index the corresponding lattice parameters in that order.

`CalculateSimplexInterpolationWeightsAndParameterIndices`(x)
1. Compute the sorted order π of the components of x such that $x[\pi[k]]$ is the kth largest value of x,
2. that is, $x[\pi[1]]$ is the largest value of x, etc.
3. Initialize `index` $= 0$, $z = 1$, `indices`$[] = []$, `weights`$[] = []$
4. For $d = 0$ to $D-1$:
5. Append `index` to `indices`
6. Append $z - x[\pi[d]]$ to `weights`
7. Update `index` $=$ `index` $+ s_{\pi[d]}$
8. Update $z = x[\pi[d]]$
9. Append `index` to `indices`
10. Append z to `weights`
11. Return `indices` and `weights`

5.2.3 Simplex Interpolation and Monotonicity

We show that the same linear inequality constraints that guarantee monotonicity for multilinear interpolation also guarantee monotonicity with simplex interpolation:

Lemma 3 (Monotonic Constraints with Simplex Interpolation) *Let $f(x) = \theta^T \phi(x)$ for $\phi(x)$ given in Algorithm 2. The partial derivative $\partial f(x)/\partial x[d] > 0$ iff $\theta_{k'} > \theta_k$ for all k, k' such that $v_k[d] = 0$, $v_{k'}[d] = 1$, and $v_k[m] = v_{k'}[m]$ for all $m \neq d$.*

Proof Simplex interpolation linearly interpolates from $D+1$ vertices at a time, and thus the resulting function is linear over each simplex. Thus to prove that the function is monotonic everywhere, we need only to show that each locally linear function is monotonically increasing in dimension d, and that the function is continuous everywhere. Each simplex only has one pair of vertices v_k and $v_{k'}$ that differ in dimension d such that $v_k[d] = 0$, $v_{k'}[d] = 1$, and $v_k[m] = v_{k'}[m]$ for all $m \neq d$. In addition, we can verify that for the linear function over this simplex, $\partial f(x)/\partial x[d] = \theta_{k'} - \theta_k$, where θ_k and $\theta_{k'}$ are the parameters corresponding to these vertices. Therefore if $\theta_{k'} > \theta_k$, then the linear function over that simplex must be increasing with respect to d. Conversely, if it does not hold that $\theta_{k'} > \theta_k$, then the linear function over that simplex must have non-positive slope with respect to d. Further, $f(x)$ is continuous for all x, because any x on a boundary between simplices only has nonzero interpolation weight on the vertices defining that boundary. In conclusion, the function is piecewise monotonic and continuous, and thus monotonic everywhere. ■

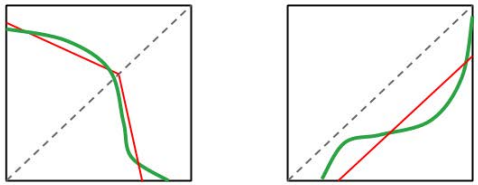


Figure 5: Illustrates rotational dependence of simplex interpolation for a 2×2 lattice and its impact on a binary classification problem. Green thick line denotes the true decision boundary of a binary classification problem. Red thin lines denote the piecewise linear decision boundary fit by lattice regression using simplex interpolation. Dotted gray line separates the two simplices; the function is locally linear over each simplex. **Left:** The true decision boundary (green) crosses the two simplices. The simplex decision boundary (red) has two linear pieces and can fit the green boundary well. **Right:** The same green boundary has been rotated ninety degrees, and now lies entirely in one simplex. The simplex decision boundary (in red) is linear within each simplex, and hence has less flexibility to fit the true green decision boundary.

5.2.4 Using Simplex Interpolation for Machine Learning

Simplex interpolation produces a locally linear continuous function made-up of $D!$ hyperplanes oriented around the main diagonal axis of the unit hypercube. Compared to multilinear interpolation, simplex interpolation is not as smooth (though continuous), and it is rotationally-dependent.

For low-dimensional regression problems using a look-up table with many cells, performance of the two interpolation methods has been found to be similar, particularly if one is using a fine-grained lattice with many cells. For example, in a comparison by Sun and Zhou (2012) for the three-dimensional regression problem of color managing an LCD monitor, multilinear interpolation of a $9 \times 9 \times 9$ look-up table (also called trilinear interpolation in the special case of three-dimensions) produced around 1% worse average error than simplex interpolation, but the maximum error with multilinear interpolation was only 60% of the maximum simplex interpolation error. Another study by Kang (1995) using simulations concluded that the interpolation errors of these methods was "about the same."

However, when using a coarser lattice like 2^D, as we have found useful in practice for machine learning, the rotational dependence of simplex interpolation can cause problems because the flexibility of the interpolated function $f(x)$ differs in different parts of the feature space. Figure 5 illustrates this for a binary classifier on two features.

To address the rotational dependence, we recommend using prior knowledge to define the features positively or negatively in a way that aligns the simplices' shared diagonal axis along the assumed slope of $f(x)$. If there are monotonicity constraints, this is done by specifying each feature so that it is monotonically increasing, rather than monotonically

decreasing. For binary classification, features should be specified so that the feature vector for the most prototypical example of the negative class is the all-zeros feature vector, and the feature vector for the most prototypical example of a positive class is the all-ones feature vector. This should put the decision boundary as orthogonal to the shared diagonal axis as possible, providing the interpolated function the most flexibility to model that decision boundary. In addition, for low-dimensional problems, using a finer-grained lattice will produce more flexibility overall, so that the flexibility within each lattice cell is less of an issue.

Following these guidelines, we surprisingly and consistently find that simplex interpolation of 2^D lattices is roughly as accurate as multilinear interpolation, and much faster for $D \geq 8$. This is demonstrated in the case studies of Section 10 (runtime comparisons given in Section 10.7).

6. Regularizing the Lattice Regression To Be More Linear

We propose a new regularizer that takes advantage of the lattice structure and encourages the fitted function to be more linear by penalizing differences in parallel edges:

$$R_{\text{torsion}}(\theta) = \sum_{d=1}^{D} \sum_{\substack{\tilde{d}=1 \\ \tilde{d}\neq d}}^{D} \sum_{\substack{r,s,t,u \text{ such that} \\ v_r \text{ and } v_s \text{ adjacent in dimension } d, \\ v_t \text{ and } v_u \text{ adjacent in dimension } d, \\ v_r \text{ and } v_t \text{ adjacent in dimension } \tilde{d}}} ((\theta_r - \theta_s) - (\theta_t - \theta_u))^2. \qquad (11)$$

This regularizer penalizes how much the lattice function twists from side-to-side, and hence we refer to this as the *torsion* regularizer. The larger the weight on the torsion regularizer in the objective function, the more linear the lattice function will be over each 2^D lattice cell.

Figure 6 illustrates the torsion regularizer and compares it to previously proposed lattice regularizers, the standard graph Laplacian (Garcia and Gupta, 2009) and graph Hessian (Garcia et al., 2012). As shown in the figure, for multi-cell lattices, the torsion and graph Hessian regularizers make the function more linear in different ways, and may both be needed to closely approximate a linear function. Like the graph Laplacian and graph Hessian regularizers, the proposed torsion regularizer is convex but not strictly convex, and can be expressed in quadratic form as $\theta^T K \theta$, where K is a positive semidefinite matrix.

7. Jointly Learning Feature Calibrations

One can learn arbitrary bounded functions with a sufficiently fine-grained lattice, but increasing the number of lattice vertices M_d for the dth feature multiplicatively grows the total number of parameters $M = \prod_d M_d$. However, we find in practice that if the features are first each transformed appropriately, then many problems require only a 2^D lattice to capture the feature interactions. For example, a feature that measures distance might be better specified as log of the distance. Instead of relying on a user to determine how to best transform each feature, we automate this feature pre-processing by augmenting our function class with D one-dimensional transformations $c_d(x[d])$ that we learn jointly with the lattice, as shown in Figure 7.

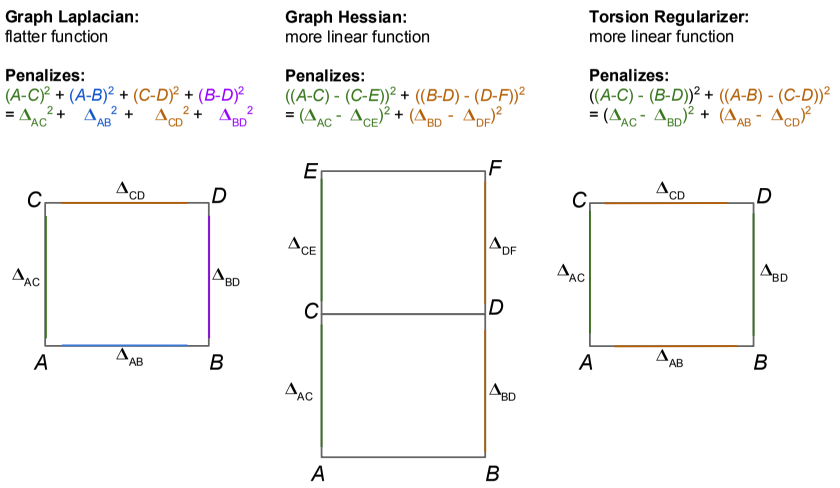


Figure 6: Comparison of lattice regularizers. The lattice parameters are denoted by A, B, C, D, E, and F. The deltas indicate the differences between adjacent parameters along each edge, and thus each delta is the slope of the function along its edge. Each color corresponds to a different additive term in a regularizer. The graph Laplacian regularizer *(left)* minimizes the sum of the squared slopes, producing a flatter function. The graph Hessian regularizer *(middle)* minimizes the change in slope in each direction of a multi-cell lattice, keeping the function from bending too much between lattice cells. The proposed torsion regularizer *(right)* minimizes the change in slope between sides of the lattice, for each direction, minimizing the twisting of the function.

7.1 Calibrating Continuous Features

We calibrate each continuous feature with a one-dimensional monotonic piecewise linear function, as illustrated in Figure 8. Our approach is similar to the work of Howard and Jebara (2007), which jointly learns monotonic piecewise linear one-dimensional transformations and a linear function.

This joint estimation makes the objective non-convex, discussed further in Section 9.3. To simplify estimating the parameters, we treat the number of changepoints C_d for the dth feature as a hyperparameter, and fix the C_d changepoint locations (also called knots) at equally-spaced quantiles of the feature values. The changepoint values are then optimized jointly with the lattice parameters, detailed in Section 9.3.

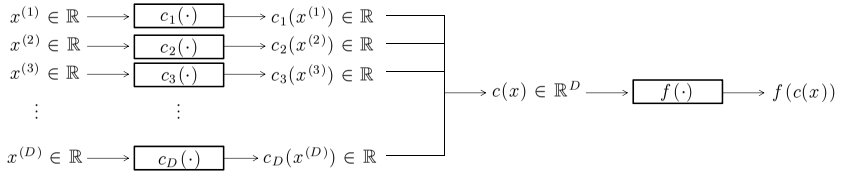
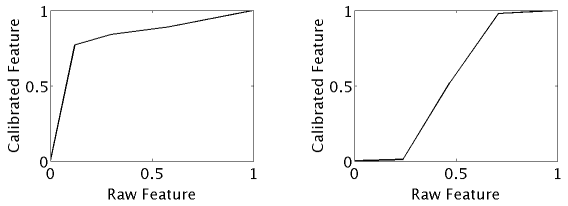


Figure 7: Block diagram showing one-dimensional calibration functions $\{c_d(\cdot)\}$ to pre-process each feature before the lattice $f(\cdot)$ fuses the features together nonlinearly.



Distance Calibration Address Similarity Calibration

Figure 8: Learned one-dimensional piecewise linear calibration functions for a distance and address-similarity feature for the business-matching case study in Section 10.2. **Left:** The raw distance is measured in meters, and its calibration has a log-like effect. **Right:** The raw address feature is calibrated with a sigmoid-like transformation.

7.2 Calibrating Categorical Features

If the dth feature is categorical, we propose using a calibration function $c_d(\cdot)$ to map each category to a real value in $[0, M_d - 1]$. That is, let the set of possible categories for the dth feature be denoted $\mathcal{G}_d$, then $c_d : \mathcal{G}_d \to [0, M_d - 1]$, adding $|\mathcal{G}_d|$ additional parameters to the model. Figure 9 shows an example lattice with a categorical country feature that has been calibrated to lie on $[0, 2]$. If prior knowledge is given about the ordering of the original discrete values or categories, then partial or full pairwise constraints can be added on the mapped values to respect the known ordering information. These can be expressed as additional sparse linear constraints on pairs of parameters.

8. Calibrating Missing Data and Using Missing Data Vertices

We propose two supervised approaches to handle missing values in the training or test set.

First, one can do a supervised imputation of missing data values by calibrating a missing data value for each feature. This is the same approach proposed for calibrating categorical

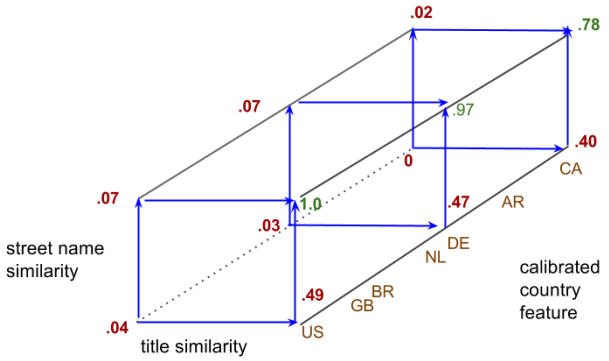


Figure 9: A $2 \times 2 \times 3$ lattice illustrating calibrating a categorical feature. In this example, each sample is a pair of business listings, and the problem is to classify whether the two listings are about the same business, based on the similarity of their street names, titles, and the country. A score $f(x)$ is interpolated from the parameters corresponding to the vertices of the $2 \times 2 \times 2$ lattice cell in which x lies, then thresholded at 0.5. The red parameter values are below the matching threshold of 0.50, and the green parameters are above the matching threshold. The blue arrows denote that the lattice was constrained to be monotonically increasing in the street name similarity and the title similarity. In this toy example, we only show the calibrated values for a few countries: US maps to 0, Great Britain maps to .3, Brazil to .4, Netherlands to .9, Germany to 1, Argentina to 1.5, and Canada to 2. One can interpret this lattice as modeling three classifiers, sliced along the country vertices: a classifier for country = 0, one for country = 1, and one for country = 2. Samples from Argentina (AR) are interpolated equally from the parameters where country = 1 and country = 2. Samples from Great Britain, and Netherlands are interpolated from the two classifiers specified at country = 0 and 1, with Netherlands putting the most weight on the classifier where country = 1. The lattice parameters can be interpreted as showing that both the street name and title features are stronger positive evidence in the US than in Canada.

values in Section 7.2: learn the numeric value in $[0, M_d - 1]$ to impute if the dth feature is missing that minimizes the structural risk minimization obejctive. In this approach, missing data is handled by a calibration function $c_d(\cdot)$, and like the other calibration function parameters. Other researchers have also considered joint training of classifiers and imputations for missing data, for example van Esbroeck et al. (2014) and Liao et al. (2007).

Second, a more flexible option is to give missing data its own *missing data vertices* in the lattice, as shown in Figure 10. This is similar to a decision tree handling a missing data value by splitting a node on whether that feature is missing. For example, the non-

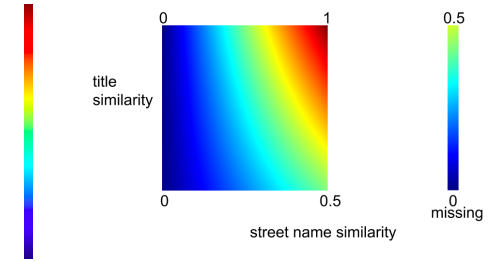


Figure 10: Illustration of handling missing data by assigning missing data to its own slice of vertices in the lattice. In this example, one feature is a title similarity and is always given, and the other feature is a street name similarity that can be missing. The lattice has $3 \times 2 = 6$ parameters, with the parameter values shown. For example, given a feature vector x with title similarity 0.5 and missing street name similarity, the two parameters corresponding to the missing street name slice of the lattice would be interpolated with equal weights, producing the output $f(x) = 0.25$.

missing feature values can be scaled to $[0, M_d - 2]$, and if the data is missing is it mapped to $M_d - 1$. This increases the number of parameters but gives the model the flexibility to handle missing data differently than non-missing data. For example, missing the street number in a business description may correlate with lower quality information for all the features.

To regularize the lattice parameters corresponding to missing data vertices, we apply the graph regularizers detailed in Section 6. These could be use to tie any of the parameters to the missing data parameters. In our experiments, for the purposes of graph regularization, we treat the missing data vertices as though they were adjacent to the minimum and maximum vertices of that feature in the lattice.

With either of these two proposed strategies, linear inequalities can be added on the appropriate parameters (the calibrator parameters in the first proposal, or the missing data vertex parameters in the second proposal) to ensure that the function value for missing data is bounded by the minimum and maximum function values, that is, that missing $x[d]$ never produces a smaller $f(x)$ than $x[d] = 0$, nor a larger $f(x)$ than $x[d] = M_d$.

9. Large-Scale Training

For convex loss functions $\ell(\theta)$ and convex regularizers $R(\theta)$, any solver for convex problems with linear inequality constraints can be used to optimize the lattice parameters θ in (8). However, for large n and for even relatively small D, training the proposed calibrated monotonic lattices is challenging due to the number of linear inequality constraints, the number

of terms in the graph-regularizers, and the non-convexity created by using calibration functions.

In this section we discuss various standard and new strategies we found useful in practice: our use of stochastic gradient descent (SGD), stochastic handling of the regularizers, parallelizing-and-averaging for distributed training, handling the large number of constraints in the context of SGD, and finally some details on how we optimize the non-convex problem of training the calibrator functions and the lattice parameters. Throughout this section, we assume the standard setting of (8); the generalization to the pairwise ranking problem of (9) is straightforward.

9.1 SGD and Reducing Variance of the Subgradients

To scale to a large number of samples n, we used SGD for all our experiments. For each SGD iteration t, a labeled training sample (x_i, y_i) is sampled uniformly from the set of training sample pairs. One finds the corresponding subgradient of (8), and takes a tiny step in its negative gradient direction. (The resulting parameters may then violate the constraints, which we discuss in Section 9.4.)

A straightforward SGD implementation for (8) would use the subgradient:

$$\Delta = \nabla_\theta \ell\left(\theta^T \phi\left(x_i\right), y_i\right) + \nabla_\theta R\left(\theta\right), \tag{12}$$

where the ∇_θ operator finds an arbitrary subgradient of its argument w.r.t. θ. Ideally, these subgradients should be cheap-to-compute, so each iteration is fast. The computational cost is dominated by computing the regularizer, if using any of the graph regularizers discussed in Section 6.

Because the training example (x_i, y_i) in (12) is randomly sampled, the above subgradient is a realization of a stochastic subgradient whose expectation is equal to the true gradient. The number of iterations needed for the SGD to converge depends on the squared Euclidean norms of the stochastic subgradients (Nemirovski et al., 2009), with larger norms resulting in slower convergence. The expected squared norm of the stochastic subgradient can be decomposed into the sum of two terms: the squared expected subgradient magnitude, and the variance. We can do little about the expected magnitude, but we can improve the trade-off between the computational cost of each subgradient and the variance of the stochastic subgradients. In the next two sub-sections, we describe two such strategies.

9.1.1 Mini-Batching

We reduce the variance of the stochastic subgradient's loss term by mini-batching over multiple random samples (Dekel et al., 2012). Let $\mathcal{S}_\ell$ denote a set of k_ℓ training indices sampled uniformly with replacement from $1, \ldots, n$, then the mini-batched subgradient is:

$$\Delta = \frac{1}{k_\ell} \sum_{i \in \mathcal{S}_\ell} \nabla_\theta \ell\left(\theta^T \phi\left(x_i\right), y_i\right) + \nabla_\theta R\left(\theta\right). \tag{13}$$

This simultaneously reduces the variance and increases the computational cost of the loss term by a factor of k_ℓ. For sufficiently small k_ℓ, this is a net win because differentiating the regularizer is the dominant computational term.

9.1.2 Stochastic Subgradients for Regularizers

We propose to reduce the computational cost of each SGD iteration by randomly sampling the additive terms of the regularizer, for regularizers that can be expressed as a sum of terms: $R(\theta) = \sum_{j=1}^{m} r_j(\theta)$. For example, for a 2^D lattice, each calculation of the graph Laplacian regularizer subgradient sums over $m = D2^{D-1}$ terms, and the graph torsion regularizer subgradient sums over $m = D(D-1)2^{D-3}$ terms.

Let $\mathcal{S}_R$ denote a set of k_R indices sampled uniformly with replacement from $1, l \ldots, m$, then define the subgradient:

$$\Delta = \frac{1}{k_\ell} \sum_{i \in \mathcal{S}_\ell} \nabla_\theta \ell\left(\theta^T \phi\left(X_i\right), Y_i\right) + \frac{m}{k_R} \sum_{j \in \mathcal{S}_R} \nabla_\theta r_j\left(\theta\right). \tag{14}$$

While this makes the subgradient's regularizer term stochastic, and hence increases the subgradient variance, we find that good choices of k_ℓ and k_R in (14) can produce a useful tradeoff between the computational cost of computing each subgradient and the number of SGD iterations needed for acceptable converge. For example, in one real-world application using torsion regularization, the choice of $k_R = 1024$ and $k_\ell = 1$ led to a 150× speed-up in training and produced statistically indistinguishable accuracy on a held-out test set.

9.2 Parallelizing and Averaging

For a large number of training samples n, one can split the n training samples into K sets, then independently and in-parallel train a lattice on each of the K sets. Once trained, the vector lattice parameters for the K lattices can simply be averaged. This parallelize-and-average approach was investigated for large-scale training of linear models by Mann et al. (2009). Their results showed similar accuracies to distributed gradient descent, but 1000× less network traffic and reduced wall-clock time for large datasets. In our implementation of the parallelize-and-average approach we do multiple syncs: averaging the lattices, then sending out the averaged lattice to parallelized workers to keep improving with further training. We illustrate the performance and speed-up of this simple parallelize-and-average for learning monotonic lattices in Section 10.6 and Section 10.7. A more complicated implementation of this strategy would use the alternating direction method of multipliers with a consensus constraint (Boyd et al., 2010), but that requires an additional regularization towards a local copy of the most recent consensus parameters.

Note that if calibration functions are used, they must be held fixed during the parallelization of the lattice training, as it does not make mathematical sense to average differently calibrated lattices.

9.3 Jointly Optimizing Lattice and Calibration Functions

To learn a *calibrated* monotonic lattice, we jointly optimize the calibration functions and the lattice parameters. Let x denote a feature vector with D components, each of which is either a continuous or categorical value (discrete features can be modeled either as continuous features or categorical as the user sees fit). Let $c_d(x[d]; \alpha^{(d)})$ be a calibration function that acts on the dth component of x and has parameters $\alpha^{(d)}$.

If the dth feature is continuous, we assume it has a bounded domain such that $x[d] \in [l_d, u_d]$ for finite $l_d, u_d \in \mathbb{R}$. Then the dth calibration function $c_d(x[d]; \alpha^{(d)})$ is a monotonic

piecewise linear transform with fixed knots at l_d, u_d, and the $C_d - 2$ equally-spaced quantiles of dth feature over the training set. Let the first and last knots of the piecewise linear function map to the lattice bounds 0 and $M_d - 1$ respectively (as shown in Figure 8), that is, if $C_d = 2$ then $c_d(x[d]; \alpha^{(d)})$ simply linearly scales the raw range $[l_d, u_d]$ to the lattice domain $[0, M_d - 1]$ and there are no parameters $\alpha^{(d)}$. For $C_d > 2$, the parameters $\alpha^{(d)} \in [0, M_d - 1]^{C_d - 2}$ are the $C_d - 2$ output values of the piecewise linear function for the middle $C_d - 2$ knots.

If the dth feature is categorical with finite category set $\mathcal{G}_d$ such that $x[d] \in \mathcal{G}_d$, then the dth calibration function maps the categories to the lattice span such that $c_d(x[d]; \alpha^{(d)}) : \mathcal{G}_d \rightarrow [0, M_d - 1]$ and the parameters are the $|\mathcal{G}_d|$ categorical mappings such that $c_d(x[d]; \alpha^{(d)}) = \alpha^{(d)}[k]$ if $x[d]$ belongs to category k and $\alpha^{(d)} \in [0, M_d - 1]^{|\mathcal{G}_d|}$.

Let $c(x; \alpha)$ denote the vector function with dth component function $c_d(x[d]; \alpha^{(d)})$, and note $c(x; \alpha)$ maps a feature vector x to the domain $\mathcal{M}$ of the lattice function. Use e_d to denote the standard unit basis vector that is one for the dth component and zero elsewhere with length D, then one can write:

$$c(x; \alpha) = \sum_{d=1}^{D} e_d c_d(e_d^T x; \alpha^{(d)}), \tag{15}$$

Then the proposed calibrated monotonic lattice regression objective expands the monotonic lattice regression objective (8) to:

$$\arg\min_{\theta, \alpha} \sum_{i=1}^{n} \ell(y_i, \theta^T \phi(c(x_i, \alpha)) + R(\theta) \text{ s.t. } A\theta \leq b \text{ and } \tilde{A}\alpha \leq \tilde{b},$$

where each row of A specifies a monotonicity constraint for a pair of adjacent lattice parameters (as before), and each row of $\tilde{A}$ similarly specifies a monotonicity constraint for a pair of adjacent calibration parameters for one of the piecewise linear calibration functions.

This turns the convex optimization problem (8) into a non-convex problem that is marginally convex in the lattice parameters θ for fixed α, but not necessarily convex with respect to α even if θ is fixed. Despite the non-convexity of the objective, in our experiments we found sensible and effective solutions by using projected SGD, updating θ and α with the appropriate stochastic subgradient for each x_i. Calculate the subgradient w.r.t. θ holding α constant, essentially the same as before. Calculate the subgradient w.r.t α by holding θ constant and using the chain rule:

$$\frac{\partial \theta^T \phi(c(x_i, \alpha))}{\partial \alpha^{(d)}} = \frac{\partial \theta^T \phi(c(x_i, \alpha))}{\partial c(x_i, \alpha)} \frac{\partial c(x_i, \alpha)}{\partial \alpha^{(d)}}. \tag{16}$$

If the dth feature is categorical, the partial derivative is 1 for the calibration mapping parameter corresponding to the category of $x_i[d]$ and zero otherwise:

$$\frac{\partial c(x_i, \alpha)}{\partial \alpha^{(d)}[k]} = 1 \text{ if } x_i[d] \text{ is the } k\text{th category and 0 otherwise.} \tag{17}$$

If the dth feature is continuous, then the parameters $\alpha^{(j)}[d]$ are the values of the calibration function at the knots of the piecewise linear function. If $x_i[d]$ lies between the kth

and $(k+1)$th knots at (fixed) positions β_k and β_{k+1}, then

$$\frac{\partial c(x_i, \alpha)}{\partial \alpha^{(d)}[k]} = \frac{(\beta_{k+1} - x_i[d])}{(\beta_{k+1} - \beta_k)}$$
$$\frac{\partial c(x_i, \alpha)}{\partial \alpha^{(d)}[k+1]} = \frac{(x_i[d] - \beta_k)}{(\beta_{k+1} - \beta_k)},$$

and the partial derviative is zero for all other components of $\alpha^{(d)}$. After taking an SGD step that updates $\alpha^{(d)}[k]$ and $\alpha^{(d)}[k+1]$, the $\alpha^{(d)}$ may violate the monotonicity constraints that ensure a monotonic calibration function, which can be fixed with a projection onto the constraints (see Section 9.4 for details).

A standard strategy with nonconvex gradient descent is to try multiple random initializations of the parameters. We did not explore this avenue; instead we simply try to initialize sensibly. Each lattice parameter is initialized to be the sum of its monotonically increasing components (multiply by -1 for any monotonically decreasing components) so that the lattice initialization respects the monotonicity constraints and is a linear function. The piecewise linear calibration functions are initialized to scale linearly to $[0, M_d - 1]$. The categorical calibration parameters are ordered by their mean label, then spaced uniformly on $[0, M_d - 1]$ in that order.

9.4 Large-Scale Projection Handling

Standard projected stochastic gradient descent projects the parameters onto the constraints after each stochastic gradient update. Given the extremely large number of linear inequality constraints needed to enforce monotonicity for even small D, we found a full projection each iteration impractical and un-necessary. We avoid the full projection at each iterate by using one of two strategies.

9.4.1 Suboptimal Projections

We found that modifying the SGD update to approximate the projection worked well. Specifically, for each new stochastic subgradient $\eta\Delta$, we create a set of active constraints initialized to $\emptyset$, and, starting from the last parameter values, move along the portion of $\eta\Delta$ that is orthogonal to the current active set until we encounter a constraint, add this constraint to the active set, and then continue until the update $\eta\Delta$ is exhausted or it is not possible to move orthogonal to the current active set. At all times, the parameters satisfy the constraints. It can be particularly fast because it is possible to exploit the sparsity of the monotonicity constraints (each of which depends on only two parameters) and the sparsity of Δ (when using simplex interpolation) to optimize the implementation.

But, this strategy is sub-optimal because we do not remove any constraints from the active set during each iteration, and thus parameters can "get stuck" at a corner of the feasible set, as illustrated in Figure 11. In practice, we found such problems resolve themselves because the stochasticity of the subsequent stochastic gradients eventually jiggles the parameters free. Experimentally, we found this suboptimal strategy to be very effective and to produce statistically similar objective function values and test accuracies more optimal approaches. All of the experimental results reported in this paper used this strategy. See Section 10.7 for example runtimes.

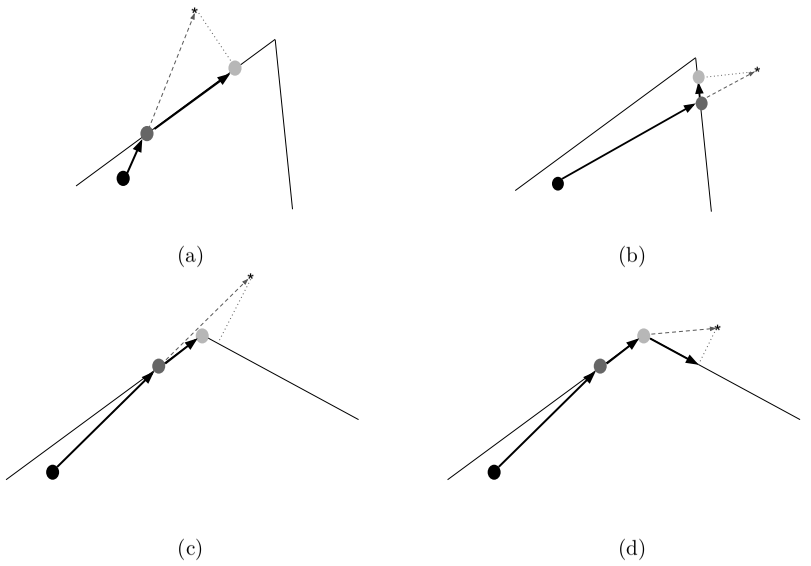


Figure 11: Four examples of the suboptimal projection stochastic gradient descent step described in Section 9.4. In each case, the constraints are marked by thin solid lines, the black dot represents the parameters at the end of the last SGD iteration, and the new full stochastic gradient descent update is marked by the dashed line, ending in a star. The optimal projection of the star onto the constraints is marked by the dotted line. The stochastic gradient is followed until it hits a constraint, and then the component of the remaining gradient orthogonal to the active constraint is applied. The update ends at the light gray dot. In cases (a) and (b), the resulting light dot is the optimal projection of the star onto the constraints. But in case (c), first one constraint is hit, and then another constraint is hit, and the update gets stuck at the corner of the feasible set without being able to apply all of the stochastic gradient. The resulting light gray dot is *not* the projection of the star onto the constraints, hence the projection for this iteration is suboptimal. However, it is likely that a future stochastic gradient will jiggle the optimization loose, as pictured in (d), producing an update that is again an optimal projection of the latest stochastic gradient.

9.4.2 Stochastic Constraints with LightTouch

An optimal approach we compared with for handling large-scale constraints is called *LightTouch* (Cotter et al., 2015). At each iteration, LightTouch does not project onto any constraints, but rather moves the constraints into the objective, and applies a random sub-

set of constraints each iteration as stochastic gradient updates to the parameters, where the distribution over the constraints is learned as the optimization proceeds to focus on constraints that are more likely to be active. This replaces the per-iteration projections with cheap gradient updates. Intermediate solutions may not satisfy all the constraints, but one full projection is performed at the very end to ensure final satisfaction of the constraints. Experimentally, we found LightTouch generally converged faster (see Cotter et al. (2015) for its theoretical convergence rate), while producing similar experimental results to the above approximate projected SGD. LightTouch does require a more complicated implementation to effectively learn the distribution over the constraints.

9.4.3 Adapting Stepsizes with Adagrad

One can generally improve the speed of SGD with adagrad (Duchi et al., 2011), even for nonconvex problems (Gupta et al., 2014). Adagrad decays the step-size adaptively for each parameter, so that parameters updated more often or with larger magnitude gradients have a smaller step size. We found adagrad did speed up convergence slightly, but required a complicated implementation to correctly handle the constraints because the projections must be with respect to the adagrad norm rather than the Euclidean norm. We experimented with approximating the adagrad norm projection with the Euclidean projection, but found this approximation resulted in poor convergence. The experimental results did not make use of adagrad.

10. Case Studies

We present a series of experimental case studies on real world problems to demonstrate different aspects of the proposed methods, followed by some example runtimes for interpolation and training in Section 10.7, and some observations about the practical value of imposing monotonicity in Section 10.8.

Previous datasets used to evaluate monotonic algorithms have been small, both in the number of samples and the number of dimensions, as detailed in Table 1. In order to produce statistically significant experimental results, and to better demonstrate the practical need for monotonicity constraints, we use real-world case studies with relatively large datasets, and for which the application engineers have confirmed that they expect or want the learned function to be monotonic with respect to some subset of features. The datasets used are detailed in Table 3, and include datasets with eight thousand to 400 million samples, and nine to sixteen features, most of which are constrained to be monotonic.

The case studies demonstrate that for problems where the monotonicity assumption is warranted, the proposed calibrated monotonic lattice regression produces similar accuracy to random forests. Random forests is an unconstrained method that consistently provides competitive results on benchmark datasets, compared to many other types of machine learning methods (Fernandez-Delgado et al., 2014)).

Because any bounded function can be expressed using a sufficiently fine-grained interpolation look-up table, we expect that with appropriate use of regularizers, monotonic lattice regression will perform similarly to other guaranteed monotonic methods that use a flexible function class and are appropriately regularized, such as monotonic neural nets (see 2.2.5). However, of guaranteed monotonic methods, the only monotonic strategy that has

Dataset	# Training Samples	# Test Samples	# Features	# Lattice Parameters
Business Matching	8,000	4,000	9	1728
Ad–Query Matching	235,996	58,224	5	32
Rendering Classifier	20,000	2,500	16	65,536
Fusing Pipelines	1.6 million	390k	12	24,576
Video Ranking	400 million	25 million	12	531,441

Table 3: Summary of datasets used in the case studies.

been demonstrated to scale to the number of training samples and the number of features treated in our case studies is linear regression with non-negative coefficients (see Table 1).

10.1 General Experimental Details

We used ten-fold cross-validation on each training set to choose hyperparameters, including: whether to use graph Laplacian regularization or torsion regularization, how much regularization (in powers of ten), whether to calibrate missing data or use a missing data vertex, the number of change-points if feature calibration was used from the choices: $\{2, 3, 5, 10, 20, 50\}$, and the number of vertices for each feature was started at 2 and increased by 1 as long as cross-validation accuracy increased. The step size was tuned using ten-fold cross-validation and choices were powers of 10; it was usually chosen to be one of $\{.01, .1, 1\}$. If calibration functions were used, a hyperparameter was used to scale the step size for the calibration function gradients compared to the lattice function gradients; this calibration step size scale was also chosen using ten-fold cross-validation and powers of 10, and was usually chosen to be one of $\{.01, .1, 1, 10\}$. Multilinear interpolation was used unless it is noted that simplex interpolation was used. The loss function was squared error, unless noted that logistic loss was used.

Comparisons were made to random forests (Breiman, 2001), and to linear models, with either the logistic loss (logistic regression) or squared error loss (linear regression), and a ridge regularizer on the linear coefficients, with any categorical or missing features converted to Boolean features. All comparisons were trained on the same training set, hyperparameters were tuned using cross-validation, and tested on the same test set. Statistical significance was measured using a binomial statistical significance test with a p-value of .05 on the test samples rated differently by two models.

10.2 Case Study: Business Entity Resolution

In this case study, we compare the relative impact of several of our proposed extensions to lattice regression. The business entity resolution problem is to determine if two business descriptions refer to the same real-world business. This problem is also treated by Dalvi et al. (2014), where they focus on defining a good title similarity. Here, we consider only the problem of fusing different similarities (such as a title similarity and phone similarity) into one score that predicts whether a pair of businesses are the same business. The learned function is required to be monotonically increasing in seven attribute similarities, such as

the similarity between the two business titles and the similarity between the street names. There are two other features with no monotonicity constraints, such as the geographic region, which takes on one of 14 categorical values. Each sample is derived from a pair of business descriptions, and a label provided by an expert human rater indicating whether that pair of business descriptions describe the same real-world business. We measure accuracy in terms of whether a predicted label matches the ground truth label, but in actual usage, the learned function is also used to rank multiple matches that pass the decision threshold, and thus a strictly monotonic function is preferred to a piecewise constant function. The training and test sets, detailed in Table 3, were randomly split from the complete labeled set. Most of the samples were drawn using active sampling, so most of the samples are difficult to classify correctly.

Table 4 reports results. The linear model performed poorly, because there are many important high-order interactions between the features. For example, the pair of businesses might describe two pizza places at the same location, one of which recently closed, and the other recently opened. In this case, location-based features will be strongly positive, but the classifier must be sensitive to low title similarity to determine the businesses are different. On the other hand, high title similarity is not sufficient to classify the pair as the same, for example, two Starbucks cafes across the street from each other in downtown London.

The lattice regression model was first optimized using cross-validation, and then we made the series of minor changes (with all else held constant) listed in Table 4 to illustrate the impact of these changes on accuracy. First, removing the monotonicity constraints resulted in a statistically significant drop in accuracy of half a percent. Thus it appears the monotonicity constraints are successfully regularizing given the small amount of training data and the known high Bayes error in some parts of the feature space. Lattice regression without the monotonicity constraints performed similarly to random forests (and not statistically significantly better), as expected due to the similar modeling abilities of the methods.

The cross-validated lattice was $3 \times 3 \times 3 \times 2^6$, where the first three features used a missing data vertex (so the non-missing data is interpolated from a 2^9 lattice). Calibrating the missing values for those three features instead of using missing data vertices statistically significantly dropped the accuracy from 81.9% to 80.7%. (However, if one subsamples the training set down to 3000 samples, then the less flexible option of calibrating the missing values works better than using missing data vertices.)

The cross-validated calibration used five changepoints for two of the four continuous features, and no calibration for the two other continuous features. Figure 8 shows the calibrations learned in the optimized lattice regression. Removing the continuous signal calibration resulted in a statistically significant drop in accuracy.

Another important proposal of this paper is calibrating categorical features to real-valued features. For this problem, this is applied to a feature specifying which of 14 possible geographical categories the businesses are in. Removing this geographic feature statistically significantly reduced the accuracy by half a percent.

The amount of torsion regularization was cross-validated to be 10^{-4}. Changing to graph Laplacian and re-optimizing the amount of regularization decreased accuracy slightly, but not statistically significantly so. This is consistent with what we often find: torsion is

	Test Set Accuracy	Monotonic Guarantee?
Linear Model	66.6%	yes
Random Forest	81.2%	no
Lattice Regression, Optimized	81.9%	yes
... Remove Monotonicity Constraints	81.4%	no
... Calibrate All Missing Data	80.7%	yes
... Remove Calibration	81.1%	yes
... Remove the Geographic Feature	81.4%	yes
... Change to Graph Laplacian	81.7%	yes
... Change to Simplex Interpolation	81.6%	yes

Table 4: Comparison on a business entity resolution problem.

often slightly better, but often not statistically significantly so, than the graph Laplacian regularizer.

Changing the multilinear interpolation to simplex interpolation (see Section 5.2) dropped the accuracy slightly, but not statistically significantly. For some problems we even see simplex interpolation provide slightly better results, but generally the accuracy difference between simplex and multilinear interpolation is negligible.

10.3 Case Study: Scoring Ad–Query Pairs

In this case study, we demonstrate the potential of the calibration functions. The goal is to score how well an ad matches a web search query, based on five different features that each measure a different notion of a good match. The score is required to be monotonic with respect to all five features. The labels are binary, so this is trained and tested as a classification problem. The train and test sets were independently and identically distributed, and are detailed in Table 3.

Results are shown in Table 5. The cross-validated lattice size was $2 \times 2 \times 2 \times 2 \times 2$, and the calibration functions each used 5 changepoints. Removing the calibration functions and re-cross-validating the lattice size resulted in a larger lattice sized $4 \times 4 \times 4 \times 4 \times 4$, and slightly worse (but not statistically significantly worse) accuracy. In total, the uncalibrated lattice model used 1024 parameters, whereas the calibrated lattice model used only 57 parameters. We hypothesize that the smaller calibrated lattice will be more robust to feature noise and drift in the test sample distribution than the larger uncalibrated lattice model. In general, we find that the one-dimensional calibration functions are a very efficient way to capture the flexibility needed, and that in conjunction with good one-dimensional calibrations, only coarse-grained (e.g. 2^D) lattices are needed.

Both with and without calibration functions, the lattice regression models were statistically significantly better than the linear model. The random forest performed well, but was not statistically significantly better than the lattice regression.

A boosted stumps model was also trained for this problem. See Fig. 12 for a comparison of two-dimensional slices of the boosted stumps and lattice functions. The boosted stumps' test set accuracy was relatively low at 75.4%. In practice, the goal of this problem is to have

	Test Set Accuracy	Monotonic Guarantee?
Linear Model	77.2%	yes
Random Forests	78.8%	no
Lattice Regression	78.7%	yes
... Remove Continuous Signal Calibration	78.4%	yes

Table 5: Comparison on an ad-query scoring problem.

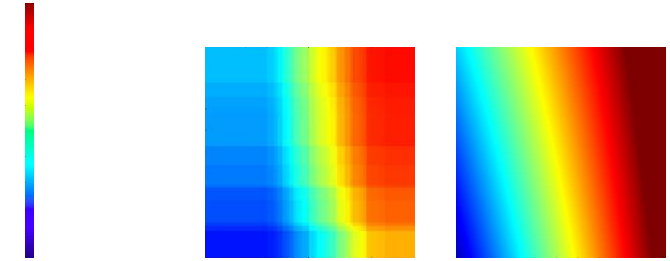


Figure 12: Slices of the learned ad-query matching functions for boosted stumps and a $2 \times 2 \times 2 \times 2 \times 2$ lattice regression, plotted as a function of two of the five features, with median values chosen for the other three features. The boosted stumps required hundreds of stumps to approximate the function, and the resulting function is piecewise constant, creating frequent ties when ranking a large number of ads for a given query, despite a priori knowledge that the output should be strictly monotonic in each of the features.

a score useful for ranking candidates as well as determining if they are a sufficiently good match. Even with many trees, this model produces many ties due its piecewise-constant surface. In addition, the live experiments with the boosted stumps showed that the output was problematically sensitive to feature noise, which would cause samples near the boundary of two piecewise constant surfaces to experience fluctuating scores.

10.4 Case Study: Rendering Classifier

This case study demonstrates training a flexible function (using a lattice) that is monotonic with respect to fifteen features. The goal is to score whether a particular display element should be rendered on a webpage. The score is required to be monotonic in fifteen of the features, and there is a sixteenth Boolean feature that is not constrained. The training and test sets (detailed in Table 3) consisted almost entirely of samples known to be difficult to correctly classify (hence the rather low accuracies).

We used a fixed 2^{16} lattice size, a fixed 5 changepoints per feature for the six continuous signals (the other ten signals were Boolean), and no graph regularization, so no hyperpa-

	Test Set Accuracy	Monotonic Guarantee?
Linear Model	54.6%	yes
Random Forest	61.3%	no
Lattice Regression	63.0%	yes

Table 6: Comparison on a rendering classifier.

rameters were optimized for this case study. Simplex interpolation was used for speed. A single training loop through the 20,000 training samples took around five minutes on a Xeon-type Intel desktop using a single-threaded C++ implementation with sparse vectors, with the training time dominated by the constraint handling. Training in total took around five hours.

Results in Table 6 show substantial gains over the linear model, while still producing a monotonic, smooth function. The lattice regression was also statistically significantly better than random forests, we hypothesize due to the regularization provided by the monotonicity constraints which is important in this case due to the difficulty of the problem on the given examples and the relatively small number of training samples.

10.5 Case Study: Fusing Pipelines

While this paper focuses on learning monotonic functions, we believe it is also the first paper to propose applying lattice regression to classification problems, rather than only regression problems. With that in mind, we include this case study demonstrating that lattice regression *without constraints* also performs similarly to random forests on a real-world large-scale multi-class problem.

The goal in this case study is to fuse the predictions from two pipelines, each of which makes a prediction about the likelihood of seven user categories based on a different set of high-dimensional features. Because each pipeline's probability estimates sum to one, only the first six probability estimates from each pipeline are needed as features to the fusion, for a total of twelve features. The training and test set were split by time, with the older 1.6 million samples used for training, and the newest 390,000 samples used as a test set.

The lattice was trained with a multi-class logistic loss, and used simplex interpolation for speed. The cross-validated model was a 2^{12} lattice for six of the output classes (with the probability of the seventh class being subtracted from one) and no calibration functions, resulting in a total of $2^{12} \times 6 = 24,576$ parameters.

The results are reported in Table 7. Even though Pipeline 2 alone is 6.5% more accurate than Pipeline 1 alone, the test set accuracy can be increased by fusing the estimates from both pipelines, with a small improvement in accuracy by lattice regression over the random forest classifier, logistic regression, or simply averaging the two pipeline estimates.

10.6 Case Study: Video Ranking and Large-Scale Learning

This case study demonstrates large-scale training of a large monotonic lattice and learning from ranked pairs. The goal is to learn a function to rank videos a user might like to watch,

	Test Set Accuracy Gain on top of Pipeline 1 Accuracy
Pipeline 2 Only	6.5%
Average the Two Pipeline Estimates	7.4%
Fuse with Linear Model	8.5%
Fuse with Random Forest	9.3%
Fuse with Lattice Regression	9.7%

Table 7: Comparison on fusing user category prediction pipelines.

based on the video they have just watched. Experiments were performed on anonymized data from YouTube.

Each feature vector x_i is a vector of features about a pair of videos, $x_i = h(v_j, v_k)$, where v_j is the watched video, v_k is a candidate video to watch next, and h is a function that takes a pair of videos and outputs a twelve-dimensional feature vector x_i. For example, a feature might be the number of times that video v_j and video v_k were watched in the same session.

Each of the twelve features was specified to be positively correlated with users viewing preference, and thus we constrained the model to be monotonically increasing with respect to each. Of course, human preference is complicated and these monotonicity constraints cannot fully model human judgement. For example, knowing that a video that has been watched many times is generally a very good indicator that it is good to suggest, and yet a very popular video at some point will flare out and become less popular.

Monotonicity constraints can also be useful to enforce secondary objectives. For example, all other features equal, one might prefer to serve fresher videos. While users in the long-run want to see fresh videos, they may preferentially click on familiar videos, thus click data may not capture this desire. This secondary goal can be enforced by constraining the learned function to be monotonic in a feature that measures video freshness. This achieves a multi-objective function without overly-complicating or distorting the training label definition.

There are billions of videos in YouTube, and thus many many pairs of watched-and-candidate videos to score and re-score as the underlying feature values change over time. Thus it is important the learned ranking functions to be cheap to evaluate, and so we use simplex interpolation for its evaluation speed; see Section 10.7 for comparison of evaluation speeds.

We trained to minimize the ranked pairs objective from (9), such that the learned function f is trained for the goal of minimizing pairwise ranking errors,

$$f(h(v_j, v_k^+)) > f(h(v_j, v_k^-)),$$

for each training event consisting of a watched video v_j, and a pair of candidate videos v_k^+ and v_k^- where there is information that a user who has just watched video v_j prefers to watch v_k^+ next over v_k^-.

10.6.1 Which Pairs of Candidate Videos?

A key question is which sample pairs of candidate videos v_k^+ and v_k^- should be used as the preferred and unpreferred videos for a given watched video v_j. We used anonymized click data from YouTube's current video-suggestion system. For each watched video v_j, if a user clicked a suggested video in the second position or below, then we took the clicked video as the preferred video v_k^+, and the video suggested right above the clicked video as the unpreferred video v_j^-. We call this choice of v_k^+ and v_k^- a *bottom-clicked pair*. This choice is consistent with the findings of Joachims et al. (2005), whose eye-tracking experiments on webpage search results showed that users on average look at least at one result above the clicked result, and that these pairs of preferred/unpreferred samples correlated strongly with explicit relevance judgements. Also, using bottom-clicked pairs removes the *trust bias* that users know they are being presented with a ranked list and prefer samples that are ranked-higher (Joachims et al., 2005). In a set of preliminary experiments, we also tried training using either a randomly sampled video as v_k^-, or the video just after the clicked video, and then tested on bottom-clicked pairs. Those results showed test accuracy on bottom-clicked pairs was up to 1% more accurate if the training set only included the bottom-clicked pairs, even though that meant fewer training pairs.

An additional goal (and one that is common in commercial large-scale machine learning systems for various practical reasons) is for the learned ranking function to be as similar to the current ranking function as possible. That is, we wish to minimize changes to the current scoring if they do not improve accuracy; such accuracy-neutral changes are referred to as *churn*. To reduce churn, we added in additional pairs that reflect the decisions of the current ranking function. Each of these pairs also takes the clicked video as the preferred v_k^+, but sets the unpreferred video v_k^- to be the video that the current system ranked ten candidates lower than the clicked video. The dataset is a 50-50 mix of these churn-reducing pairs and bottom-clicked pairs.

10.6.2 More Experimental Details

The dataset was randomly split into mutually exclusive training, test, and validation sets of size 400 million, 25 million, and 25 million pairs, respectively. To ensure privacy, the dataset only contained the feature vector, and no information identifying the video or user. The disadvantage of that is the train, test and validation sets are likely to have some samples from the same videos and same users. However, in total the datasets capture millions of unique users and unique watched videos.

We used a fixed 3^{12} lattice, for a total of 531,441 parameters. The pre-processing functions were fixed in this case, so no calibration functions were learned. We compared training on increasingly-larger randomly-sampled subsets of the 400 million training set (see Figure 13 for training set sizes). We compared training on a single worker to the parallelize-and-average strategy explained in Section 9.2. Parallel results were parallelized over 100 workers. The stepsize was chosen independently for each training set based on accuracy on the validation set.

We report results with and without monotonicity constraints. For the unconstrained results, each training (single or parallel) touched each sample in the training set once. For the monotonic results (single or parallelized), each sample was touched ten times, and

minibatching was used with a minibatch size of 32 stochastic gradients. Logistic loss was used.

10.6.3 Results

Figure 13 compares test set accuracy for single and parallelized training for different amounts of training data, with and without monotonicity constraints. For each dataset, the single and parallel training saw the same total number of training samples and were allowed the same total number of stochastic gradient updates.

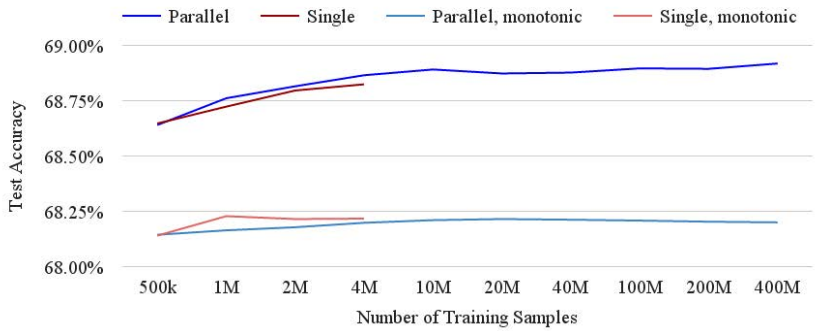


Figure 13: Comparison of training with a single worker versus 100 workers in parallel, as a function of training set size.

On the click data test set, not using monotonicity constraints (the dark lines) is about .5% better at pairwise accuracy than if we constrain the function to be monotonic. However, in live experiments that required ranking all videos (not only ones that had been top-ranked in the past, and hence included in the click data sets), models trained with monotonicity constraints showed better performance on the actual measures of user-engagement (as opposed to the training metric of pairwise accuracy). This discrepancy appears to be due to the biased sampling of the click data, as the click-data has a biased distribution over the feature space compared to the distribution of all videos which must get ranked in practice. The biased distribution of the click data appears to cause parameters in sparser regions of the feature space to be non-monotonic in an effort to increase the flexibility (and accuracy) of the function in the denser regions, thus increasing the accuracy on the click data. Enforcing monotonicity helps address this sampling bias problem by not allowing the training to ignore the accuracy in sparser regions that are important in practice to accurately rank all videos.

Even though there are 500k parameters to train, the click-data accuracy is already very good with only 500k training samples, and test accuracy increases only slightly when trained on 400 million samples compared to 10 million samples. This is largely because the click-

data samples are densely clustered in the feature space, and with simplex interpolation, only a small fraction of the 500k parameters control the function over the dense part of the feature space.

The darker lines of Figure 13 show the parallelization versus single-machine results *without* monotonicity constraints. Unconstrained, the parallelized runs appear to perform slightly better to the single-machine training given the same number of training samples (and the same total number of gradient updates). We hypothesize this slight improvement is due to some noise-averaging across the 100 parallelized trained lattices. The lighter lines of Figure 13 show the parallelization versus single-machine results *with* monotonicity constraints. Trained on 500k pairs, the parallelized training and single-machine monotonic training produce the same test accuracy. However, as the training set size increases, the parallelized training takes more data to achieve the same accuracy as the single-machine training. We believe this is because averaging the 100 monotonic lattices is a convex combination of lattices likely on the edge of the monotonicity constraint set, producing an average lattice in the interior of the constraint set, that is, the averaged lattice is over-constrained.

10.7 Run Times

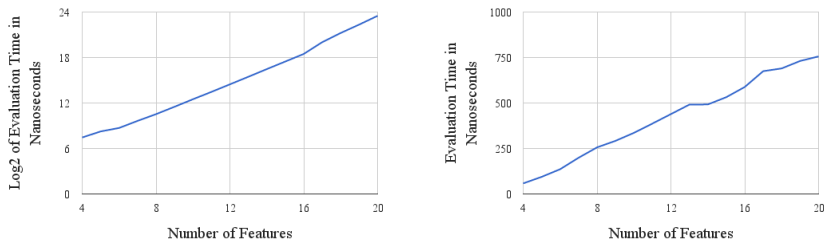
We give some timing examples for the different interpolations and for training.

Figure 14 shows average evaluation times for multilinear and simplex interpolation of one sample from a 2^D lattice for $D = 4$ to $D = 20$ using a single-threaded 3.5GHz Intel Ivy Bridge processor. Note the multilinear evaluation times are reported on a log-scale, and on a log scale the evaluation time increases roughly linearly in D, matching the theoretical $O(2^D)$ complexity given in Section 5.1. The simplex evaluation times scale roughly linearly with D, consistent with the theoretical $O(D \log D)$ complexity. For $D = 6$ features, simplex interpolation is already three times faster than multilinear. With $D = 20$ features, the simplex interpolation is still only 750 nanoseconds, but the multilinear interpolation is about $15,000$ times slower, at around 12 milliseconds.

Training times are difficult to report in an accurate or meaningful way due to the high-variance of running on a large, shared, distributed cluster. Here is one example: with every feature constrained to be monotonic, a single worker training one loop of a 2^{12} lattice on 4 million samples usually takes around 15 minutes, whereas with 100 parallelized workers one loop through 400 million samples (4 million samples for each worker) usually takes around 20 minutes. Large step-sizes can take much longer than smaller stepsizes, because larger updates tend to violate more monotonicity constraints and thus require more expensive projections. Minibatching is particularly effective at speeding up training because the averaged batch of stochastic gradients reduces the number of monotonicity violations and the need for projections. Without monotonicity constraints, training is generally $10\times$ to $1000\times$ faster, depending on how non-monotonic the data is.

10.8 Interpretability in Practice

It is difficult to quantify interpretability, but we summarize our observations from working with around 50 different users on around a dozen different real-world machine learning problems where there are a relatively small number of semantically meaningful features.



(a) Multilinear Interpolation

(b) Simplex Interpolation

Figure 14: Average evaluation time to interpolate a sample from a 2^D lattice. Figure (a) shows the multilinear interpolation time on a $\log_2$ scale in nanoseconds. Figure (b) shows the much faster simplex interpolation time in nanoseconds. Simplex interpolation is $10\times$ faster than multilinear for $D = 9$ features, about $100\times$ faster for $D = 13$ features, and over $1,000\times$ faster for $D = 17$ features.

First, we do find that being able to summarize a model as being a positive or negative function with respect to each input feature does help users feel that they understand and can predict the model's behavior better than a comparable unconstrained model. In particular, while aggregate measures like accuracy, precision, or recall over a test set provide summary statistics over that particular test set, we find that users in some cases do worry about the *unknown unknowns* of using a machine learning model, and that adding monotonicity constraints gives these users more confidence that the model can be trusted not to behave unreasonably for any examples. And this confidence is well-founded: as discussed in the video ranking case study in Section 10.6, monotonicity constraints do in practice guard against potentially strange behavior of highly nonlinear functions in rarer parts of the feature space.

We have also found that monotonicity constraints make debugging highly nonlinear models easier. We find that one particularly useful debugging tool is sensitivity plots like the one shown in Figure 15, which show how $f(x)$ relates to each feature value of x, for a particular sample x. Monotonicity constraints make these sensitivity plots monotonic, which we find makes it easier to identify problems with signals and training data.

Apart from the issue of monotonicity, we expected that using one-dimensional calibration functions and interpolated look-up tables would produce parameters that were interpretable. These expectations were half-right. We do find it helpful and common for users to check and analyze the signals' calibration functions, and that the calibrations provide useful information to users about what the model has learned, and helps identify unexpected behavior or problems with features. But, we do not find that users examine the *lattice* parameters directly very often, and that the readability of the lattice parameters becomes generally less useful as D increases. Users are more likely to utilize other analytics, such as how correlated the output is with each calibrated feature. While this information does not

control for between-feature correlations (so a feature might be highly correlated with the output but not needed if it correlates with another feature), users can conclude the model behaves a lot like highly-correlated features, and this aids model understanding. Low correlation between a calibrated feature and the output either indicates an unimportant feature, or, if the feature is known to be important (because dropping it hurts accuracy), that it plays an important role interacting or conditioning other features.

To understand feature interactions, again we find that rather than analyze the lattice parameters explicitly, users generally prefer to analyze two-dimensional visualizations of the model over pairs of features (averaged or sliced over the other features), or to examine examples to check their hypotheses about expected higher-order interactions.

Another common problem is to understand how two similar models are different (for example, one might have been trained with more training data, or one might use an additional feature). For this purpose, we find it is again rare that users want to directly analyze differences in model parameters except for very tiny models, preferring instead to look at examples that are scored differently by the two models, and analyzing these example samples in their raw form (for example, looking at the videos that have been promoted or demoted by a new model, in conjunction with the features used by the model).

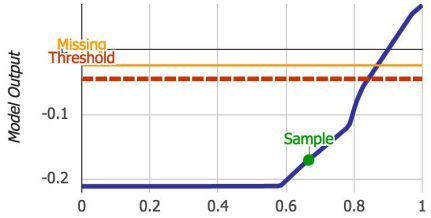


Figure 15: Illustration of a sensitivity plot for a calibrated monotonic lattice with respect to one of the D features. The blue line shows how the output $f(x)$ (y-axis) of a calibrated monotonic lattice changes if only this one feature value of x (x-axis) is changed, but all other components of x are kept fixed. The green dot marks the current input and output. The yellow line shows the model output if the first feature is missing. The red dotted line shows the binary classification threshold. This plot is piecewise linear because the calibrator function is piecewise linear and the simplex interpolation is also piecewise linear, and monotonically increasing because the function was constrained to be monotonic with respect to this feature. In this example, one sees that to change the classification decision without changing any other features, this feature would have to be increased from its current value of 0.67 to at least 0.84 at which point it would cross the red line marking the decision threshold.

11. Discussion and Some Open Questions

We have proposed using constrained interpolated look-up tables to effectively learn flexible, monotonic functions for low-dimensional machine learning problems of classification, ranking, and regression. We addressed a number of practical issues, including interpretability, evaluation speed, automated pre-processing of features, missing data, and categorical features. Experimental results show state-of-the-art performance on the largest training sets and largest number of features published for monotonic methods.

Practical experience has shown us that being able to check and ensure monotonicity helps users trust the model, and leads to models that generalize better. For us, the monotonicity constraints have come from engineers who believe the output should be monotonic in the feature. In the absence of clear prior information about monotonicity, it may be tempting to use the direction of a linear fit to specify a monotonic direction and then use monotonicity as a regularizer. Magdon-Ismail and Sill (2008) point out that using the linear regression coefficients for this purpose can be misleading if features are correlated and not jointly Gaussian.

For classifiers, requiring the entire function to be monotonic is a stronger requirement than needed to simply guarantee that the decision boundary (and hence classifier) is monotonic. It is an open question how to enforce only the thresholded function to be monotonic, and whether that would be more useful in practice.

One surprise was that for practical machine learning problems like those of Section 10, we found a simple 2^D lattice is often sufficient to capture the interactions of D features, especially if we jointly optimized D one-dimensional feature calibration functions. When we began this work, we expected to have to use much more fine-grained lattices with many vertices in each feature, or perhaps irregular lattices to achieve state-of-the-art accuracy. In fact, calibration functions help approximately linearize each feature with respect to the label, making a 2^D lattice sufficiently flexible for most of the real-world problems we have encountered.

For some cases, a 2^D lattice is too flexible. We reduced lattice flexibility with new regularizers: monotonicity, and the torsion regularizer that encourages a more linear model. While good for interpretability and accuracy, these regularization strategies do not reduce the model size.

For a large number of features D, the exponential model size of a 2^D lattice is a memory issue. On a single machine, training and evaluating with a few million parameters is viable, but this still limits this approach to not much more than $D = 20$ features. An open question is how such large models could be sparsified, and if useful sparsification approaches could also provide additional useful regularization.

A second surprise was that simplex interpolation provides similar accuracy to multilinear interpolation. The rotational dependence of simplex interpolation seemed at first troubling, but the proposed approach of aligning the shared axis of the simplices with the main increasing axis of the function appears to solve this problem in practice. The geometry of the simplices at first seemed odd in that it produces a locally linear surface over elongated simplices. However, this partitioning turns out to work well because it provides a very flexible piecewise linear decision boundary. Lastly, we found that the theoretical

$O(D \log D)$ computational complexity does result in practice in orders of magnitude faster interpolation than multilinear interpolation as D increases.

A common practical issue in machine learning is handling categorical data. We proposed to learn a mapping from mutually exclusive categories to feature values, jointly with the other model parameters. We found categorical-mapping to be interpretable, flexible, and accurate. The proposed categorical mapping can be viewed as learning a one-dimensional embedding of the categories. Though we generally only needed two vertices in the lattice for continuous features, for categorical features we often find it helpful to use more vertices (a finer-grained lattice) for more flexibility. Some preliminary experiments learning two-dimensional embeddings of categories (that is, mapping one category to $[0, 1]^2$) showed promise, but we found this required more careful initialization and handling of the increased non-convexity.

Learning the monotonic lattice is a convex problem, but composing the lattice and the one-dimensional calibration functions creates a non-convex objective. We used only one initialization of the lattice and calibrators for all our experiments, but tuned the stepsize of the stochastic gradient descent separately for the set of lattice parameters and the set of calibration parameters. In some cases we saw a substantial sensitivity of the accuracy to the initial SGD stepsizes. We hypothesize that this is caused by some interplay of the relative stepsizes and the relative size of the local optima.

We employed a number of strategies to speed up training. One of the biggest speed-ups comes from randomly sampling the additive terms of the graph regularizers, analogous to the random sampling of the additive terms of the empirical loss that SGD uses. We showed that a parallelize-and-average strategy works for training the lattices. The largest computational bottleneck remains the projections onto the monotonicity constraints. Mini-batching the samples reduces the number of projections and provides speed-ups, but a faster approach to optimization given possibly hundreds of thousands of constraints would be valuable.

Lastly, this work leaves open a number of theoretical questions for the function class of interpolated look-up tables, for example how monotonicity constraints theoretically affect convergence speed.

12. Acknowledgments

We thank Sugato Basu, David Cardoze, James Chen, Emmanuel Christophe, Brendan Collins, Mahdi Milani Fard, James Muller, Biswanath Panda, and Alex Vodomerov for help with experiments and helpful discussions.

References

Y. S. Abu-Mostafa. A method for learning from hints. In *Advances in Neural Information Processing Systems*, pages 73–80, 1993.

N. P. Archer and S. Wang. Application of the back propagation neural network algorithm with monotonicity constraints for two-group classification problems. *Decision Sciences*, 24(1):60–75, 1993.

F. Bach. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2), 2013.

R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk. *Statistical inference under order restrictions; the theory and application of isotonic regression*. Wiley, New York, USA, 1972.

A. Ben-David. Automatic generation of symbolic multiattribute ordinal knowledge based DSS: methodology and applications. *Decision Sciences*, pages 1357–1372, 1992.

A. Ben-David. Monotonicity maintenance in information-theoretic machine learning algorithms. *Machine Learning*, 21:35–50, 1995.

A. Ben-David, L. Sterling, and Y. H. Pao. Learning and classification of monotonic ordinal concepts. *Computational Intelligence*, 5(1):45–49, 1989.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1), 2010.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

J. Casillas, O. Cordon, F. Herrera, and L. Magdalena (Eds.). Trade-off between accuracy and interpretability in fuzzy rule-based modelling. *Physica-Verlag*, 2002.

R. Chandrasekaran, Y. U. Ryu, V. S. Jacob, and S. Hong. Isotonic separation. *INFORMS Journal on Computing*, 17(4):462–474, 2005.

A. Cotter, M. R. Gupta, and J. Pfeifer. A Light Touch for Heavily Constrained SGD. *arXiv preprint*, 2015. URL `http://arxiv.org/abs/1512.04960`.

N. Dalvi, M. Olteanu, M. Raghavan, and P. Bohannon. Deduplicating a places database. *Proc. ACM WWW Conf.*, 2014.

H. Daniels and M. Velikova. Monotone and partially monotone neural networks. *IEEE Trans. Neural Networks*, 21(6):906–917, 2010.

O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal distributed online prediction using mini-batches. *Journal Machine Learning Research*, 13(1):165–202, January 2012.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal Machine Learning Research*, 12:2121–2159, 2011.

C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia. Incorporating second-order functional knowledge for better option pricing. In *Advances in Neural Information Processing Systems (NIPS)*, 2000.

C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia. Incorporating functional knowledge in neural networks. *Journal Machine Learning Research*, 2009.

W. Duivesteijn and A. Feelders. Nearest neighbour classification with monotonicity constraints. *Proc. European Conf. Machine Learning*, pages 301–316, 2008.

A. Feelders. Monotone relabeling in ordinal classification. *Proc. IEEE Conf. Data Mining*, pages 803–808, 2010.

M. Fernandez-Delgado, E. Cernadas, S. Barro, and D. Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal Machine Learning Research*, 2014.

E. K. Garcia and M. R. Gupta. Lattice regression. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

E. K. Garcia, S. Feldman, M. R. Gupta, and S. Srivastava. Completely lazy learning. *IEEE Trans. Knowledge and Data Engineering*, 22(9):1274–1285, Sept. 2010.

E. K. Garcia, R. Arora, and M. R. Gupta. Optimized regression for efficient function evaluation. *IEEE Trans. Image Processing*, 21(9):4128–4140, Sept. 2012.

S. Garcia, A. Fernandez, J. Luengo, and F. Herrera. A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Computing*, 13:959–977, 2009.

I. J. Good. *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. MIT Press, 1965.

H. Gruber, M. Holzer, and O. Ruepp. Sorting the slow way: an analysis of perversely awful randomized sorting algorithms. In *Fun with Algorithms*, pages 183–197. Springer, 2007.

M. Gupta, S. Bengio, and J. Weston. Training highly multiclass classifiers. *Journal Machine Learning Research*, 2014.

M. R. Gupta, R. M. Gray, and R. A. Olshen. Nonparametric supervised learning by linear interpolation with maximum entropy. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(5):766–781, 2006.

T. Hastie and R. Tibshirani. *Generalized Additive Models*. Chapman Hall, New York, 1990.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, 2001.

C. C. Holmes and N. A. Heard. Generalized monotonic regression using random change points. *Statistics in Medicine*, 22:623–638, 2003.

A. Howard and T. Jebara. Learning monotonic transformations for classification. In *Advances in Neural Information Processing Systems*, 2007.

H. Ishibuchi and Y. Nojima. Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning. *International Journal of Approximate Reasoning*, 44:4–31, 2007.

T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. *Proc. SIGIR*, 2005.

H. R. Kang. Comparison of three-dimensional interpolation techniques by simulations. *SPIE Vol. 2414*, 1995.

H. R. Kang. *Color Technology for Electronic Imaging Devices*. SPIE Press, USA, 1997.

J. Kasson, W. Plouffe, and S. Nin. A tetrahedral interpolation technique for color space conversion. *SPIE Vol. 1909*, 1993.

H. Kay and L. H. Ungar. Estimating monotonic functions and their bounds. *AIChE Journal*, 46(12):2426–2434, 2000.

R. E. Knop. A note on hypercube partitions. *Journal of Combinatorial Theory, Ser. A*, 15 (3):338–342, 1973.

W. Kotlowski and R. Slowinski. Rule learning with monotonicity constraints. In *Proceedings International Conference on Machine Learning*, 2009.

F. Lauer and G. Bloch. Incorporating prior knowledge in support vector regression. *Machine Learning*, 70(1):89–118, 2008.

X. Liao, H. Li, and L. Carin. Quadratically gated mixture of experts for incomplete data classification. *Proc. ICML*, 2007.

T.-Y. Liu. *Learning to Rank for Information Retrieval*. Springer, 2011.

Malik Magdon-Ismail and J. Sill. A linear fit gets the correct monotonicity directions. *Machine Learning*, pages 21–43, 2008.

G. Mann, R. McDonald, M. Mohri, N. Silberman, and D. D. Walker. Efficient large-scale distributed training of conditional maximum entropy models. *Advances in Neural Information Processing Systems (NIPS)*, 2009.

D. G. Mead. Dissection of the hypercube into simplexes. *Proc. Amer. Math. Soc.*, 76: 302–304, 1979.

A. Minin, M. Velikova, B. Lang, and H. Daniels. Comparison of universal approximators incorporating partial monotonicity by structure. *Neural Networks*, 23(4):471–475, 2010.

H. Mukarjee and S. Stern. Feasible nonparametric estimation of multiargument monotone functions. *Journal of the American Statistical Association*, 89(425):77–80, 1994.

B. Neelon and D. B. Dunson. Bayesian isotonic regression and trend analysis. *Biometrics*, 60:398–406, 2004.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, January 2009.

K. Neumann, M. Rolf, and J. J. Steil. Reliable integration of continuous constraints into extreme learning machines. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 21(supp02):35–50, 2013.

R. Nock. Inducing interpretable voting classifiers without trading accuracy for simplicity: Theoretical results, approximation algorithms, and experiments. *Journal Artificial Intelligence Research*, 17:137–170, 2002.

K.-M. Osei-Bryson. Post-pruning in decision tree induction using multiple performance measures. *Computers and Operations Research*, 34:3331–3345, 2007.

R. Potharst and A. J. Feelders. Classification trees for problems with monotonicity constraints. *ACM SIGKDD Explorations*, pages 1–10, 2002a.

R. Potharst and A. J. Feelders. Pruning for monotone classification trees. *Springer Lecture Notes on Computer Science*, 2810:1–12, 2002b.

Y.-J. Qu and B.-G. Hu. Generalized constraint neural network regression model subject to linear priors. *IEEE Trans. on Neural Networks*, 22(11):2447–2459, 2011.

J. O. Ramsay. Estimating smooth monotone functions. *Journal of the Royal Statistical Society, Series B*, 60:365–375, 1998.

G. Rätsch, S. Sonnenburg, and C. Schäfer. Learning interpretable SVMs for biological sequence classification. *BMC Bioinformatics*, 7, 2006.

J. Riihimäki and A. Vehtari. Gaussian processes with monotonicity information. In *International Conference on Artificial Intelligence and Statistics*, pages 645–652, 2010.

R. Rovatti, M. Borgatti, and R. Guerrieri. A geometric approach to maximum-speed n-dimensional continuous linear interpolation in rectangular grids. *IEEE Trans. on Computers*, 47(8):894–899, 1998.

F. Schimdt and R. Simon. Some geometric probability problems involving the Eulerian numbers. *Electronic Journal of Combinatorics*, 4(2), 2007.

G. Sharma and R. Bala. *Digital Color Imaging Handbook*. CRC Press, New York, 2002.

T. S. Shively, T. W. Sager, and S. G. Walker. A Bayesian approach to non-parametric monotone function estimation. *Journal of the Royal Statistical Society, Series B*, 71(1): 159–175, 2009.

P. K. Shukla and S. P. Tripathi. A review on the interpretability-accuracy trade-off in evolutionary multi-objective fuzzy systems (EMOFS). *Information*, 2012.

J. Sill. Monotonic networks. *Advances in Neural Information Processing Systems (NIPS)*, 1998.

J. Sill and Y. S. Abu-Mostafa. Monotonicity hints. *Advances in Neural Information Processing Systems (NIPS)*, pages 634–640, 1997.

D. J. Spiegelhalter and R. P. Knill-Jones. Statistical and knowledge-based approaches to clinical decision support systems, with an application in gastroenterology. *Journal of the Royal Statistical Society A*, 147:35–77, 1984.

J. Spouge, H. Wan, and W. J. Wilbur. Least squares isotonic regression in two dimensions. *Journal of Optimization Theory and Applications*, 117(3):585–605, 2003.

C. Strannegaard. Transparent neural networks. *Proc. Artificial General Intelligence*, 2012.

B. Sun and S. Zhou. Study on the 3D interpolation models used in color conversion. *IACSIT Intl. Journal Engineering and Technology*, 4(1), 2012.

A. van Esbroeck, S. Singh, I. Rubinfeld, and Z. Syed. Evaluating trauma patients: Addressing missing covariates with joint optimization. *Proc. AAAI*, 2014.

M. Villalobos and G. Wahba. Inequality-constrained multivariate smoothing splines with application to the estimation of posterior probabilities. *Journal of the American Statistical Association*, 82(397):239–248, 1987.

S. Wang. A neural network method of density estimation for univariate unimodal data. *Neural Computing & Applications*, 2(3):160–167, 1994.

L. Yu and J. Xiao. Trade-off between accuracy and interpretability: experience-oriented fuzzy modeling via reduced-set vectors. *Computers and Mathematics with Applications*, 57:885–895, 2012.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 7/15; Revised 3/16; Published 7/16

Are Random Forests Truly the Best Classifiers?

Michael Wainberg m.wainberg@utoronto.ca
Department of Electrical and Computer Engineering
University of Toronto, Toronto, ON M5S 3G4, Canada;
Deep Genomics, Toronto, ON M5G 1L7, Canada

Babak Alipanahi babak@psi.toronto.edu
Department of Electrical and Computer Engineering
University of Toronto, Toronto, ON M5S 3G4, Canada

Brendan J. Frey frey@psi.toronto.edu
Department of Electrical and Computer Engineering
University of Toronto, Toronto, ON M5S 3G4, Canada;
Deep Genomics, Toronto, ON M5G 1L7, Canada

Editor: Nando de Freitas

Abstract

The JMLR study *Do we need hundreds of classifiers to solve real world classification problems?* benchmarks 179 classifiers in 17 families on 121 data sets from the UCI repository and claims that "the random forest is clearly the best family of classifier". In this response, we show that the study's results are biased by the lack of a held-out test set and the exclusion of trials with errors. Further, the study's own statistical tests indicate that random forests do not have significantly higher percent accuracy than support vector machines and neural networks, calling into question the conclusion that random forests are the best classifiers.

Keywords: classification, benchmarking, random forests, support vector machines, neural networks

1. Errors in Results

The authors state that they used the following training procedure for 102 of the 121 benchmarked data sets, which were not already divided into training and test sets:

> "One training and one test set are generated randomly (each with 50% of the available patterns) [...]. This couple of sets is used only for parameter tuning (in those classifiers which have tunable parameters), selecting the parameter values which provide the best accuracy on the test set. [...] Then, using the selected values for the tunable parameters, a 4-fold cross validation is developed using the whole available data. [...] The test results is the average over the 4 test sets."

This procedure is incorrect because **it does not use a held-out test set**. Since half the test examples are also used as a validation set for hyperparameter tuning, selecting hyperparameters which maximize accuracy on the validation set will by definition inflate performance on half the test examples. This error has likely inflated the reported performance of at least some of the benchmarked classifiers with tunable hyperparameters (based on the descriptions in the original paper, these include `rrlda_R`, `sda_t`, `PenalizedLDA_t`, `sparseLDA_R`, `fda_t`, `mda_t`, `pda_t`, `rda_R`, `hdda_R`, `rbf_m`, `rbf_t`, `rbfDDA_t`, `mlp_m`, `mlp_C`, `mlp_t`, `avNNet_t`, `mlpWeightDecay_t`, `nnet_t`, `pcaNNet_t`, `pnn_m`, `elm_m`, `elm_kernel_m`, `lvq_R`, `lvq_t`, `dkp_C`, `svm_C`, `svmlight_C`, `LibSVM_w`, `svmRadial_t`, `svmRadialCost_t`, `svmLinear_t`, `svmPoly_t`, `lssvmRadial_t`, `rpart_t`, `rpart2_t`, `ctree_t`, `ctree2_t`, `JRip_t`, `C5.0_t`, `MultiBoostAB_LibSVM_w`, `Bagging_LibSVM_w`, `rf_t`, `RRF_t`, `cforest_t`, `parRF_t`, `RRFglobal_t`, `MultiScheme_w`, `knn_R`, `knn_t`, `IBk_w`, `pls_t`, `spls_R`, `multinom_t`, `CVParameterSelection_w`, `gaussprRadial_t` and possibly others). The authors justify the procedure by stating that:

> "We must emphasize that, since parameter tuning and testing use different data sets, the final result can not be biased by parameter optimization, because the set of parameter values selected in the tuning stage is not necessarily the best on the test set."

While hyperparameter tuning and testing do technically use different sets of examples, since one is a subset of the other, the two sets must be disjoint to avoid bias.

Further, the authors did not follow the stated procedure: the test evaluation does not use cross-validation since the 4 test sets[1] are not independent. For instance, trains, the smallest benchmark with 10 examples, has two examples in each test set, with the following indices: $\{2,9\}$, $\{4,9\}$, $\{4,9\}$, and $\{4,7\}$. This means that examples 4 and 9 are given three times the weight of examples 2 and 7 when calculating the test accuracy, and the other six examples are ignored. This negates the purpose of cross-validation, which is to give equal importance to every example.

The results are also biased by the exclusion of trials with errors. If a classifier is unable to run a particular benchmark, that benchmark is excluded when calculating the classifier's mean percent accuracy—but it is not excluded for classifiers that ran it successfully. Effectively, each classifier is evaluated on a slightly different set of benchmarks, so the mean percent accuracies are not directly comparable. We re-evaluated the mean percent accuracy of the top 8 classifiers on only the benchmarks successfully run by all 8, and found that a neural network, `elm_kernel_matlab`, was competitive with random forests (Table 1), even having the highest mean accuracy (albeit by a very small, insignificant, margin). Still, this neural network also had the highest number of failures (which did not count toward mean accuracy).

2. Flawed Conclusions

The conclusion that "The random forest is clearly the best family of classifiers" is flawed. The paper gives three arguments for why random forests are the best family: "The eight random forest classifiers are included among the 25 best classifiers having all of them low ranks", "The family RF has the lowest minimum rank (32.9) and mean (46.7), and also a narrow interval (up to 60.5), which means that all the RF classifiers work very well", and "3 out of [the] 5" best classifiers are random forests.

The problem with the first two arguments is that the notion of a "best family" is not well defined, and is sensitive to the choice of classifiers included in each family. For instance,

1. Partitions are available at http://persoal.citius.usc.es/manuel.fernandez.delgado/papers/jmlr/data.tar.gz.

Rank	Classifier	Mean % accuracy		# failed
		Original	Corrected	
1	elm_kernel_matlab	81.96	81.84	7
2	rf_caret	82.30	81.78	1
3	rforest_R	81.93	81.58	1
4	parRF_caret	81.96	81.32	0
5	svm_C	81.81	81.30	2
6	svmRadialCost_caret	81.43	81.17	6
7	svmPoly_caret	81.19	81.06	6
8	svmRadial_caret	81.04	80.82	6

Table 1: Mean percent accuracy of the top 8 classifiers reported in Fernández-Delgado et al. ("Original") and when re-evaluated on only the benchmarks successfully run by all 8 ("Corrected"). The number of benchmarks which failed to run for each classifier is also shown. A neural network, elm_kernel_m, has the highest mean percent accuracy, followed by three random forest models and then four support vector machine models.

the worst-performing neural network is the direct parallel perceptron (Fernandez-Delgado et al. (2011)), developed by the authors of the original paper and designed to favour extreme computational efficiency over accuracy. Similarly, the worst rule-based classifier, ZeroR_w, is a baseline that always predicts the majority class. Mean accuracy conflates these classifiers with others in the same family that are designed to favour accuracy. Also, larger families will tend to have greater variance, with a lower minimum rank and a higher maximum rank.

The argument that "3 out of [the] 5" best classifiers are random forests is also questionable. The three best random forest classifiers are actually a single classifier (randomForest in R) with different wrappers (parRF_t is parallelized; parRF_t and rf_t use caret) and settings for mtry, the number of features in each tree (parRF_t uses a grid search of 2 to 8 in steps of 2; rf_t searches from 2 to 29 in steps of 3, and rforest_R sets mtry = $\sqrt{\#features}$), so it would be more correct (but still not fully correct) to say that one out of the three best classifiers is a random forest and two are support vector machines.

Most importantly, the results do not show that the best random forests perform any better than the best support vector machines and neural networks. The authors conducted paired t-tests showing that the differences in accuracy between the top-ranked random forest parRF_t and the other top eight models, including support vector machines, neural networks and other random forests, are not statistically significant (left panel of Fig. 4 in Fernández-Delgado et al.). This calls into question the conclusion that random forests are the best classifiers.

The use of a paired t-test is itself incorrect, since the test assumes that the difference between the two distributions under comparison is normally distributed, yet none of the differences between parRF_t and the other top 10 classifiers are normally distributed according to a χ^2 goodness-of-fit test for normality (Table 2). Also, before calculating the p values in Fig. 4, missing values were imputed by setting accuracies for benchmarks with errors to 82%, the mean accuracy of parRF_t across all benchmarks. This biases the results because 82% is an average over many benchmarks, some of which are so simple that every classifier achieves

Classifier	Normality p
elm_kernel_matlab	1e-29
rf_caret	2e-43
rforest_R	1e-36
svm_C	2e-15
svmRadialCost_caret	7e-13
svmPoly_caret	5e-09
svmRadial_caret	4e-14
avNNet_caret	6e-10
C5.0_caret	1e-21

Table 2: p values that the differences between parRF_t and each of the other top 10 classifiers, across all benchmarks where both classifiers ran without errors, are not normally distributed. All p values are significant, indicating that none of the differences are normally distributed.

Classifier	Paired t-test p, errors set to 82% (Fig. 4 in original paper)	Paired t-test p, excluding errors	Wilcoxon signed-rank p, excluding errors
elm_kernel_matlab	1	0.5	0.4
rf_caret	0.4	0.3	1
rforest_R	1	0.9	0.1
svm_C	0.8	1	0.8
svmRadialCost_caret	0.5	0.6	0.6
svmPoly_caret	0.3	0.4	0.4
svmRadial_caret	0.1	0.2	0.1
avNNet_caret	0.03	0.03	0.02
C5.0_caret	0.001	0.001	0.0001

Table 3: Using a Wilcoxon signed-rank test instead of a paired t-test, and removing missing values rather than incorrectly imputing them, results in the same conclusions about the significance of the differences between parRF_t and each of the other top 10 classifiers.

100% accuracy and others of which are so difficult that most classifiers achieves below 30% accuracy, so it is unreasonable to assume that a classifier would achieve 82% accuracy on every failed benchmark. A hypothetical classifier that failed to run every benchmark would be considered as good as parRF_t according to this imputation method! However, using the Wilcoxon signed-rank test, which does not make the same assumption of normality, and removing benchmarks where either classifier gave an error rather than imputing to 82%, results in the same conclusions about significance (Table 3).

To support the conclusion that parRF_t is better than the second-ranked classifier svm_C, even though the paired t-test showed them to be statistically indistinguishable, Fernández-Delgado et al. states that:

> "Although `parRF_t` is better than `svm_C` in 56 of 121 data sets, worse than `svm_C` in 55 sets, and equal in 10 sets, [...] `svm_C` is never much better than `parRF_t`: when `svm_C` outperforms `parRF_t`, the difference is small, but when `parRF_t` outperforms `svm_C`, the difference is higher [...]. In fact, calculating for each data set the difference between the accuracies of `parRF_t` and `svm_C`, the sum of positive differences (parRF is better) is 193.8, while the negative ones (`svm_C` better) sum [to] 139.8."

`parRF_t` and `svm_C` are only equal for 8 benchmarks, not 10: the two benchmarks where `svm_C` gave an error are erroneously counted towards this total. Also, the quoted figures are for `rf_t`, not `parRF_t`: the true figures, based on the online results, are 220.4 and 219.6. The above statement does not imply that the difference between `parRF_t` and `svm_C` is significant, since it is equivalent to saying that the difference in mean percent accuracy between the two classifiers, across the 119 benchmarks where both ran without errors, is $\frac{220.4-219.6}{119 \text{ benchmarks}} \approx 0.007$, a difference that the paired t-test already showed to be insignificant ($p = 0.834$ in Fig. 4 of the original paper, or $p = 0.8$ using a Wilcoxon signed-rank test). (Note that 0.007 does not exactly equal $82.0 - 81.8$, the difference between the reported mean percent accuracies of `parRF_t` and `svm_C`, because these accuracies are biased by the exclusion of trials with errors as discussed in Section 1.) Notably, the difference between `rf_t` and `svm_C` (193.8 vs 139.8), while larger, is also insignificant (Wilcoxon signed-rank $p = 0.8$).

3. Availability of Code

Code to reproduce the three tables and calculate the sum of positive and negative accuracy differences between pairs of classifiers is available as a supplement to this paper.

Acknowledgments

We thank Andrew Delong and Michael Leung for useful discussions. Funding was provided by the Natural Sciences and Engineering Research Council of Canada's John C. Polanyi Award and grants from the Canadian Institutes of Health Research and the Ontario Genomics Institute. MW was supported by a Rogers Scholarship. BA was supported by a joint Autism Research Training and NeuroDevNet Fellowship. BJF is a Fellow of the Canadian Institute for Advanced Research.

References

Manuel Fernandez-Delgado, Jorge Ribeiro, Eva Cernadas, and Senén Barro Ameneiro. Direct parallel perceptrons (DPPs): fast analytical calculation of the parallel perceptrons weights with margin control for classification tasks. *Neural Networks, IEEE Transactions on*, 22 (11):1837–1848, 2011.

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

Journal of Machine Learning Research 17 (2016) 1-43 Submitted 7/15; Revised 11/15; Published 7/16

Minimax Adaptive Estimation of Nonparametric Hidden Markov Models

Yohann De Castro YOHANN.DECASTRO@MATH.U-PSUD.FR

Élisabeth Gassiat ELISABETH.GASSIAT@MATH.U-PSUD.FR

Claire Lacour CLAIRE.LACOUR@MATH.U-PSUD.FR
Laboratoire de Mathématiques d'Orsay,
Univ. Paris-Sud, CNRS, Université Paris-Saclay,
91405 Orsay, France.

Editor: David Dunson

Abstract

We consider stationary hidden Markov models with finite state space and nonparametric modeling of the emission distributions. It has remained unknown until very recently that such models are identifiable. In this paper, we propose a new penalized least-squares estimator for the emission distributions which is statistically optimal and practically tractable. We prove a non asymptotic oracle inequality for our nonparametric estimator of the emission distributions. A consequence is that this new estimator is rate minimax adaptive up to a logarithmic term. Our methodology is based on projections of the emission distributions onto nested subspaces of increasing complexity. The popular spectral estimators are unable to achieve the optimal rate but may be used as initial points in our procedure. Simulations are given that show the improvement obtained when applying the least-squares minimization consecutively to the spectral estimation.

Keywords: nonparametric estimation, hidden Markov models, minimax adaptive estimation, oracle inequality, penalized least-squares.

1. Introduction

1.1 Context and Motivations

Finite state space hidden Markov models (HMMs for short) are widely used to model data evolving in time and coming from heterogeneous populations. They seem to be reliable tools to model practical situations in a variety of applications such as economics, genomics, signal processing and image analysis, ecology, environment, speech recognition, to name but a few. From a statistical view point, finite state space HMMs are stochastic processes $(X_j, Y_j)_{j\geq 1}$ where $(X_j)_{j\geq 1}$ is a Markov chain with finite state space and conditionally on $(X_j)_{j\geq 1}$ the Y_j's are independent with a distribution depending only on X_j. The observations are $Y_{1:N} = (Y_1, \ldots, Y_N)$ and the associated states $X_{1:N} = (X_1, \ldots, X_N)$ are unobserved. The parameters of the model are the initial distribution, the transition matrix of the hidden chain, and the emission distributions of the observations, that is the probability distributions of the Y_j's conditionally to $X_j = x$ for all possible x's. In this paper we shall consider stationary ergodic HMMs so that the initial distribution is the stationary distribution of the (ergodic) hidden Markov chain.

Until very recently, asymptotic performances of estimators were proved only in the parametric setting (that is, with finitely many unknown parameters). Though, nonparametric methods for HMMs have been considered in applied papers, but with no theoretical guarantees, see for instance Couvreur and Couvreur (2000) for voice activity detection, Lambert et al. (2003) for climate state identification, Lefèvre (2003) for automatic speech recognition, Shang and Chan (2009) for facial expression recognition, Volant et al. (2014) for methylation comparison of proteins, Yau et al. (2011) for copy number variants identification in DNA analysis.

The preliminary obstacle to obtain theoretical results on general finite state space nonparametric HMMs was to understand when such models are indeed identifiable. Marginal distributions of finitely many observations are finite mixtures of products of the emission distributions. It is clear that identifiability can not be obtained based on the marginal distribution of only one observation. It is needed, and it is enough, to consider the marginal distribution of at least three consecutive observations to get identifiability, see Gassiat et al. (2016), following Allman et al. (2009) and Hsu et al. (2012).

1.2 Contribution

The aim of our paper is to propose a new approach to estimate nonparametric HMMs with a statistically optimal and practically tractable method. We obtain this way nonparametric estimators of the emission distributions that achieve the minimax rate of estimation in an adaptive setting.

Our perspective is based on estimating the projections of the emission laws onto nested subspaces of increasing complexity. Our analysis encompasses any family of nested subspaces of Hilbert spaces and works with a large variety of models. In this framework one could think to use the spectral estimators as proposed by Hsu et al. (2012); Anandkumar et al. (2012) in the parametric framework, by extending them to the nonparametric framework. But a careful analysis of the tradeoff between sampling size and approximation complexity shows that they do not lead to rate optimal estimators of the emission densities, see De Castro et al. (2015) for a formal statement and proof. This can be easily understood. Indeed, the spectral estimators of the emission densities are computed as functions of the empirical estimator of the marginal distribution of three consecutive observations on $\mathcal{Y}^3$ (where $\mathcal{Y}$ is the observation space), for which, roughly speaking, when $\mathcal{Y}$ is a subset of $\mathbb{R}$, the optimal rate is $N^{-s/(2s+3)}$, N being the number of observations and s the smoothness of the emission densities. Thus the rate obtained this way for the emission densities is also $N^{-s/(2s+3)}$. But since those emission densities describe one dimensional random variables on $\mathcal{Y}$, one could hope to be able to obtain the sharper rate $N^{-s/(2s+1)}$. This is the rate we obtain, up to a $\log N$ term, with our new method. Let us explain how it works.

Using the HMM modeling, and using sieves for the emission densities on $\mathcal{Y}$, we propose a penalized least squares estimator in the model selection framework. We prove an oracle inequality for the L_2-risk of the estimator of the density of (Y_1, Y_2, Y_3), see Theorem 4. Since the complexity of the model is that given by the sieves for the emission densities, this leads, up to a $\log N$ term, to the adaptive minimax rate computed as for the density of only

one observation Y_1 though we estimate the density of (Y_1, Y_2, Y_3). Roughly speaking, when the observations are one dimensional, that is when $\mathcal{Y}$ is a subset of $\mathbb{R}$, the obtained rate for the density of (Y_1, Y_2, Y_3) is of order $N^{-s/(2s+1)}$ up to a $\log N$ term, N being the number of observations and s the smoothness of the emission densities.

The key point is then to be able to go back to the emission densities. This is the cornerstone of our main result. We prove in Theorem 6 that, under the assumption **[HD]** defined in Section 4.2, the quadratic risk for the density of (Y_1, Y_2, Y_3) is lower bounded by some positive constant multiplied by the quadratic risk for the emission densities. This technical assumption is generically satisfied in the sense that it holds for all possible emission densities for which the L_2-norms and Hilbert dot products do not lie on a particular algebraic surface with coefficients depending on the transition matrix of the hidden chain. Moreover, we prove that, when the number of hidden states equals two, this assumption is always verified when the two emission densities are distinct, see Lemma 5.

Our methodology requires that we have a preliminary estimator of the transition matrix. To get such an estimator, it is possible to use spectral methods. Thus our approach is the following. First, get a preliminary estimator of the initial distribution and the transition matrix of the hidden chain. Second, apply penalized least squares estimation on the density of three consecutive observations, using HMM modeling, model selection on the emission densities, and initial distribution and stationary matrix of the hidden chain set at the estimated value. This gives emission density estimators which have minimax adaptive rate, as our main result states, see Theorem 7. A simplified version of this theorem can be given as follows.

Theorem 1 *Assume $(Y_j)_{j\geq 1}$ is a hidden Markov model on $\mathbb{R}$, with latent Markov chain $(X_j)_{j\geq 1}$ with K possible values and true transition matrix $\mathbf{Q}^\star$. Denote $f_k^\star$ the density of Y_n given $X_n = k$, for $k = 1, \dots, K$. Assume the true transition matrix $\mathbf{Q}^\star$ is full rank and the true emission densities $f_k^\star$, $k = 1, \dots, K$ are linearly independent, with smoothness s. Assume that* **[HD]** *holds true. Then, up to label switching, for N the number of observations large enough, the estimators $\hat{\mathbf{Q}}$, $\hat{f}_k$, $k = 1, \dots, K$ built in Section 3 and 5 satisfy*

$$\mathbb{E}\left[\|\mathbf{Q}^\star - \hat{\mathbf{Q}}\|^2\right] = O\Big(\frac{\log N}{N}\Big) \quad \text{and} \quad \mathbb{E}\left[\|f_k^\star - \hat{f}_k\|_2^2\right] = O\Big(\big[\frac{\log N}{N}\big]^{\frac{s}{2s+1}}\Big),\ k = 1, \dots, K.$$

Moreover, since the family of sieves we consider is that given by finite dimensional spaces described by an orthonormal basis, we are able to use the spectral estimators of the coefficients of the densities as initial points in the least squares minimization. This is important since, in the HMM framework, least squares minimization does not have an explicit solution and may lead to several local minima. However, since the spectral estimates are proved to be consistent, we may be confident that their use as initial point is enough. Simulations indeed confirm this point.

To conclude we claim that our results support a powerful new approach to estimate, for the first time, nonparametric HMMs with a statistically optimal and practically tractable method.

1.3 Related Works

The papers Allman et al. (2009), Hsu et al. (2012) and Anandkumar et al. (2012) paved the way to obtain identifiability under reasonable assumptions. In Anandkumar et al. (2012)

the authors point out a structural link between multivariate mixtures with conditionally independent observations and finite state space HMMs. In Hsu et al. (2012) the authors propose a spectral method to estimate all parameters for finite state space HMMs (with finitely many observations), under the assumption that the transition matrix of the hidden chain is non singular, and that the (finitely valued) emission distributions are linearly independent. Extension to emission distributions on any space, under the linear independence assumptions (and keeping the assumption of non singularity of the transition matrix), allowed to prove the general identifiability result for finite state space HMMs, see Gassiat et al. (2016), where also model selection likelihood methods and nonparametric kernel methods are proposed to get nonparametric estimators. Let us notice also Vernet (2015) that proves theoretical consistency of the posterior in nonparametric Bayesian methods for finite state space HMMs with adequate assumptions. Later, Alexandrovich et al. (2016) obtained identifiability when the emission distributions are all distinct (not necessarily linearly independent) and still when the transition matrix of the hidden chain is full rank. In the nonparametric multivariate mixture model, Song et al. (2014) prove that any linear functional of the emission distributions may be estimated with parametric rate of convergence in the context of reproducing kernel Hilbert spaces. The latter uses spectral methods, not the same but similar to the ones proposed in Hsu et al. (2012) and Anandkumar et al. (2012).

Recent papers that contain theoretical results on different kinds of nonparametric HMMs are Gassiat and Rousseau (2016), where the emitted distributions are translated versions of each other, and Dumont and Le Corff (2017) in which the authors consider regression models with hidden regressor variables that can be Markovian on a continuous state space. Parallel to our work, the article Bonhomme et al. (2016) introduces a non-adaptive spectral method to estimate hidden parameters in latent-structure models.

1.4 Outline of the paper

In Section 2, we set the notations, the model we tudy, and the assumptions we consider. We then state an identifiability lemma (see Lemma 3) that will be useful for our estimation method. In Sections 3 and 4 we give our main results. We explain the penalized least-squares estimation method in Section 3, and we prove in Section 4 that, when the transition matrix is irreducible and aperiodic, when the emission distributions are linearly independent and the penalty is adequately chosen, then, under a technical assumption, the penalized least squares estimator is asymptotically minimax adaptive up to a $\log N$ term, see Theorem 7 and Corollary 10. For this, we first prove an oracle inequality for the estimation of the density of (Y_1, Y_2, Y_3), see Theorem 4, then we prove the key result relating the risk of the density of (Y_1, Y_2, Y_3) to that of the emission densities, see Theorem 6. The latter holds under a technical assumption which we prove to be always verified in case $K = 2$, see Lemma 5. Finally, we need the performances of the spectral estimator of the transition matrix and of the stationary distribution which are given in Section 5, see Theorem 11, proved in De Castro et al. (2015). We finally present simulations in Section 6 to illustrate our theoretical results. Those simulations show in particular the improvement obtained when applying the least-squares minimization consecutively to the spectral estimation. Detailed proofs are given in Section 8.

2. Notations and Assumptions

2.1 Nonparametric Hidden Markov Model

Let K, D be positive integers and let $\mathcal{L}^D$ be the Lebesgue measure on $\mathbb{R}^D$. Denote by $\mathcal{X}$ the set $\{1,\dots,K\}$ of hidden states, $\mathcal{Y} \subset \mathbb{R}^D$ the observation space, and Δ_K the space of probability measures on $\mathcal{X}$ identified to the $(K-1)$-dimensional simplex. Let $(X_n)_{n\geq 1}$ be a Markov chain on $\mathcal{X}$ with $K \times K$ transition matrix $\mathbf{Q}^\star$ and initial distribution $\pi^\star \in \Delta_K$. Let $(Y_n)_{n\geq 1}$ be a sequence of observed random variables on $\mathcal{Y}$. Assume that, conditional on $(X_n)_{n\geq 1}$, the observations $(Y_n)_{n\geq 1}$ are independent and, for all $n \in \mathbb{N}$, the distribution of Y_n depends only on X_n. Denote by $\mu_k^\star$ the conditional law of Y_n conditional on $\{X_n = k\}$, and assume that $\mu_k^\star$ has density $f_k^\star$ with respect to the measure $\mathcal{L}^D$ on $\mathcal{Y}$:

$$\forall k \in \mathcal{X}\,, \quad \mathrm{d}\mu_k^\star = f_k^\star \mathrm{d}\mathcal{L}^D\,.$$

Denote by $\mathfrak{F}^\star := \{f_1^\star,\dots,f_K^\star\}$ the set of emission densities with respect to the Lebesgue measure. Then, for any integer n, the distribution of $(Y_1,\dots,Y_n)$ has density with respect to $(\mathcal{L}^D)^{\otimes n}$

$$\sum_{k_1,\dots,k_n=1}^{K} \pi^\star(k_1)\mathbf{Q}^\star(k_1,k_2)\dots\mathbf{Q}^\star(k_{n-1},k_n) f_{k_1}^\star(y_1)\dots f_{k_n}^\star(y_n).$$

We shall denote $g^\star$ the density of (Y_1,Y_2,Y_3).

In this paper we shall address two observations schemes. We shall consider N i.i.d. samples $(Y_1^{(s)},Y_2^{(s)},Y_3^{(s)})_{s=1}^N$ of three consecutive observations **(Scenario A)** or consecutive observations of the same chain **(Scenario B)**:

$$\forall s \in \{1,\dots,N\}, \quad (Y_1^{(s)},Y_2^{(s)},Y_3^{(s)}) := (Y_s,Y_{s+1},Y_{s+2})\,.$$

2.2 Projections of the population joint laws

Denote by $(\mathbf{L}^2(\mathcal{Y},\mathcal{L}^D),\|\cdot\|_2)$ the Hilbert space of square integrable functions on $\mathcal{Y}$ with respect to the Lebesgue measure $\mathcal{L}^D$ equipped with the usual inner product $\langle\cdot,\cdot\rangle$ on $\mathbf{L}^2(\mathcal{Y},\mathcal{L}^D)$. Assume $\mathfrak{F}^\star \subset \mathbf{L}^2(\mathcal{Y},\mathcal{L}^D)$.

Let $(M_r)_{r\geq 1}$ be an increasing sequence of integers, and let $(\mathfrak{P}_{M_r})_{r\geq 1}$ be a sequence of nested subspaces with dimension M_r such that their union is dense in $\mathbf{L}^2(\mathcal{Y},\mathcal{L}^D)$. Let $\Phi_{M_r} := \{\varphi_1,\dots,\varphi_{M_r}\}$ be an orthonormal basis of $\mathfrak{P}_{M_r}$. Recall that for all $f \in \mathbf{L}^2(\mathcal{Y},\mathcal{L}^D)$,

$$\lim_{r\to\infty} \sum_{m=1}^{M_r} \langle f,\varphi_m\rangle\varphi_m = f\,, \tag{1}$$

in $\mathbf{L}^2(\mathcal{Y},\mathcal{L}^D)$. Note that changing M_r may change all functions φ_m, $1 \leq m \leq M_r$ in the basis Φ_{M_r}, which we shall not indicate in the notation for sake of readability. Also, we drop the dependence on r and write M instead of M_r. Define the projection of the emission laws onto $\mathfrak{P}_M$ by

$$\forall k \in \mathcal{X}, \quad f_{M,k}^\star := \sum_{m=1}^{M} \langle f_k^\star,\varphi_m\rangle\varphi_m\,.$$

We shall write $\mathbf{f}_M^\star := (f_{M,1}^\star,\dots,f_{M,K}^\star)$ and $\mathbf{f}^\star := (f_1^\star,\dots,f_K^\star)$ throughout this paper.

Remark 2 *One can consider the following standard examples:*

(Spline) *The space of piecewise polynomials of degree bounded by d_r based on the regular partition with p_r^D regular pieces on $\mathcal{Y} = [0,1]^D$. It holds that $M_r = (d_r+1)^D p_r^D$.*

(Trig.) *The space of real trigonometric polynomials on $\mathcal{Y} = [0,1]^D$ with degree less than r. It holds that $M_r = (2r+1)^D$.*

(Wav.) *A wavelet basis Φ_{M_r} of scale r on $\mathcal{Y} = [0,1]^D$, see Meyer (1992). In this case, it holds that $M_r = 2^{(r+1)D}$.*

2.3 Assumptions

We shall use the following assumptions on the hidden chain.

[H1] *The transition matrix $\mathbf{Q}^\star$ has full rank,*

[H2] *The Markov chain $(X_n)_{n\geq 1}$ is irreducible and aperiodic,*

[H3] *The initial distribution $\pi^\star = (\pi_1^\star,\dots,\pi_K^\star)$ is the stationary distribution.*

Notice that under **[H1]**, **[H2]** and **[H3]**, one has for all $k \in \mathcal{X}$, $\pi_k^\star \geq \pi_{\min}^\star > 0$. We shall use the following assumption on the emission densities.

[H4] *The family of emission densities $\mathfrak{F}^\star := \{f_1^\star,\dots,f_K^\star\}$ is linearly independent.*

Those assumptions appear in spectral methods, see for instance Hsu et al. (2012); Anandkumar et al. (2012), and in identifiability issues, see for instance Allman et al. (2009); Gassiat et al. (2016).

2.4 Identifiability Lemma

For any $\mathbf{f} = (f_1,\dots,f_K) \in (\mathbf{L}^2(\mathcal{Y},\mathcal{L}^D))^K$ and any transition matrix $\mathbf{Q}$, denote by $g^{\mathbf{Q},\mathbf{f}} : \mathcal{Y}^3 \to \mathbb{R}$ the function given by

$$g^{\mathbf{Q},\mathbf{f}}(y_1,y_2,y_3) = \sum_{k_1,k_2,k_3=1}^{K} \pi(k_1)\mathbf{Q}(k_1,k_2)\mathbf{Q}(k_2,k_3) f_{k_1}(y_1) f_{k_2}(y_2) f_{k_3}(y_3), \tag{2}$$

where π is the stationary distribution of $\mathbf{Q}$. When $\mathbf{Q} = \mathbf{Q}^\star$ and $\mathbf{f} = \mathbf{f}^\star$, we get $g^{\mathbf{Q}^\star,\mathbf{f}^\star} = g^\star$. When $f_1,\dots,f_K$ are probability densities on $\mathcal{Y}$, $g^{\mathbf{Q},\mathbf{f}}$ is the probability distribution of three consecutive observations of a stationary HMM. We now state a lemma that gathers all what we need about identifiability.

For any transition matrix $\mathbf{Q}$, let $T_{\mathbf{Q}}$ be the set of permutations τ such that for all i and j, $\mathbf{Q}(\tau(i),\tau(j)) = \mathbf{Q}(i,j)$. The permutations in $T_{\mathbf{Q}}$ describe how the states of the Markov chain may be permuted without changing the distribution of the whole chain: for any τ in $T_{\mathbf{Q}}$, $(\tau(X_n))_{n\geq 1}$ has the same distribution as $(X_n)_{n\geq 1}$. Since the hidden chain is not observed, if the emission distributions are permuted using τ, we get the same HMM. In other words, if $\mathbf{f}^\tau = (f_{\tau(1)},\dots,f_{\tau(K)})$, then $g^{\mathbf{Q},\mathbf{f}^\tau} = g^{\mathbf{Q},\mathbf{f}}$. Since identifiability up to permutation of the hidden states is obtained from the marginal distribution of three consecutive observations, we get the following lemma whose detailed proof is given in Section 8.1.

Lemma 3 *Assume that* $\mathbf{Q}$ *is a transition matrix for which* [**H1**] *and* [**H2**] *hold. Assume that* [**H4**] *holds. Then for any* $\mathbf{h} \in (\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D))^K$,

$$g^{\mathbf{Q},\mathbf{f}^\star+\mathbf{h}} = g^{\mathbf{Q},\mathbf{f}^\star} \iff \exists \tau \in T_{\mathbf{Q}} \text{ such that } h_j = f^\star_{\tau(j)} - f^\star_j,\ j = 1,\dots,K.$$

In particular, if $T_{\mathbf{Q}}$ *reduces to the identity permutation,* $g^{\mathbf{Q},\mathbf{f}^\star+\mathbf{h}} = g^{\mathbf{Q},\mathbf{f}^\star} \iff \mathbf{h} = (0,\dots,0)$.

3. The Penalized Least-Squares Estimator

In this section we shall estimate the emission densities using the so-called penalized least squares method. Here, the least squares adjustment is made on the density $g^\star$ of (Y_1, Y_2, Y_3). Starting from the operator $\Gamma : \ t \mapsto \|t - g^\star\|_2^2 - \|g^\star\|_2^2 = \|t\|_2^2 - 2\int t g^\star$ which is minimal for the target $g^\star$, we introduce the corresponding empirical contrast γ_N. Namely, for any $t \in \mathbf{L}^2(\mathcal{Y}^3, \mathcal{L}^{D^{\otimes 3}})$, set

$$\gamma_N(t) = \|t\|_2^2 - \frac{2}{N}\sum_{s=1}^N t(Z_s),$$

with $Z_s := (Y_1^{(s)}, Y_2^{(s)}, Y_3^{(s)})$ (**Scenario A**) or $Z_s := (Y_s, Y_{s+1}, Y_{s+2})$ (**Scenario B**). As N tends to infinity, $\gamma_N(t) - \gamma_N(g^\star)$ converges almost surely to $\|t - g^\star\|_2^2$, thus the name least squares contrast function. A natural estimator is then a function t such that $\gamma_N(t)$ is minimal over a judicious approximation space which is a set of functions of form $g^{\mathbf{Q},\mathbf{f}}$, $\mathbf{Q}$ a transition matrix and $\mathbf{f} \in \mathcal{F}^K$, for $\mathcal{F}$ a subset of $\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)$. We thus define a whole collection of estimates $\hat{g}_M$, each M indexing an approximation subspace (also called model). Considering (2) we shall introduce a collection of model of functions by projection of possible $\mathbf{f}$'s on the subspaces $(\mathfrak{P}_M)_M$. Thus, for any irreducible transition matrix $\mathbf{Q}$ with stationary distribution π, we define $\mathcal{S}(\mathbf{Q}, M)$ as the set of functions $g^{\mathbf{Q},\mathbf{f}}$ such that $\mathbf{f} \in \mathcal{F}^K$ and, for each $k = 1,\dots,K$, there exists $(a_{m,k})_{1\le m\le M} \in \mathbb{R}^M$ such that

$$f_k = \sum_{m=1}^M a_{m,k}\varphi_m.$$

We now assume that we have in hand an estimator $\hat{\mathbf{Q}}$ of $\mathbf{Q}^\star$. For instance, one can use a spectral estimator, we recall such a construction in Section 5. Then, $(\mathcal{S}(\hat{\mathbf{Q}}, M))_M$ is the collection of models we use for the least squares minimization. For any M, define $\hat{g}_M$ as a minimizer of $\gamma_N(t)$ for $t \in \mathcal{S}(\hat{\mathbf{Q}}, M)$. Then $\hat{g}_M$ can be written as $\hat{g}_M = g^{\hat{\mathbf{Q}},\hat{\mathbf{f}}_M}$ with $\hat{\mathbf{f}}_M \in \mathcal{F}^K$ and $\hat{f}_{M,k} = \sum_{m=1}^M \hat{a}_{m,k}\varphi_m$ $(k = 1,\dots,K)$ for some $(\hat{a}_{m,k})_{1\le m\le M} \in \mathbb{R}^M$, $k = 1,\dots,K$. It then remains to select the best model, that is to choose M which minimizes $\|\hat{g}_M - g^\star\|_2^2 - \|g^\star\|_2^2$. This quantity is close to $\gamma_N(\hat{g}_M)$, but we need to take into account the deviations of the process $\Gamma - \gamma_N$. Then we rather minimize $\gamma_N(\hat{g}_M) + \mathrm{pen}(N, M)$ where $\mathrm{pen}(N, M)$ is a penalty term to be specified. Our final estimator will be a penalized least squares estimator. For this purpose we choose a penalty function $\mathrm{pen}(N, M)$ and we let

$$\hat{M} = \arg\min_{M=1,\dots,N} \{\gamma_N(\hat{g}_M) + \mathrm{pen}(N, M)\}.$$

Notice that, with N observations, we consider N subspaces as candidates for model selection. Then the estimator of $g^\star$ is $\hat{g} = \hat{g}_{\hat{M}}$, and the estimator of $\mathbf{f}^\star$ is $\hat{\mathbf{f}} := \hat{\mathbf{f}}_{\hat{M}}$ so that $\hat{g} = g^{\hat{\mathbf{Q}},\hat{\mathbf{f}}}$.

The least squares estimator does not have an explicit form such as in usual nonparametric estimation, so that one has to use numerical minimization algorithms. As initial point of the minimization algorithm, we shall use the spectral estimator, see Section 6 for more details. Since the spectral estimator is consistent, see De Castro et al. (2015), the algorithm does not suffer from initialization problems.

4. Adaptive Estimation of the Emission Distributions

4.1 Oracle Inequality for the Estimation of $g^\star$

We now fix a subset $\mathcal{F}$ of $\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)$, and we shall use the following assumption:

[**HF**] $\mathcal{F}$ *is a closed subset of* $\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)$ *such that: for any* $f \in \mathcal{F}$, $\int f d\mathcal{L}^D = 1$, $\|f\|_2 \le C_{\mathcal{F},2}$ *and* $\|f\|_\infty \le C_{\mathcal{F},\infty}$ *for some fixed positive* $C_{\mathcal{F},2}$ *and* $C_{\mathcal{F},\infty}$.

Our first main result is an oracle inequality for the estimation of $g^\star$ which is stated below and proved in Section 8.2. We denote by $\mathfrak{S}_K$ the set of permutations of $\{1,\dots,K\}$. When a is a vector, $\|a\|_2$ denotes its Euclidian norm, and when A is a matrix, $\|A\|_F$ denotes its Frobenius norm.

Theorem 4 *Assume* [**H1**]-[**H4**] *and* [**HF**]. *Assume also* $\mathbf{f}^\star \in \mathcal{F}^K$, *and for all* M, $\mathbf{f}^\star_M \in \mathcal{F}^K$. *Then, there exists positive constants* N_0, $\rho^\star$ *and* $A_1^\star$ (*depending on* $C_{\mathcal{F},2}$ *and* $C_{\mathcal{F},\infty}$ (**Scenario B**) *or on* $\mathbf{Q}^\star$, $C_{\mathcal{F},2}$ *and* $C_{\mathcal{F},\infty}$ (**Scenario A**)) *such that, if*

$$\mathrm{pen}(N, M) \ge \rho^\star \frac{M \log N}{N}$$

then for all $x > 0$, *for all* $N \ge N_0$, *one has with probability* $1 - (e-1)^{-1}e^{-x}$, *for any permutation* $\tau \in \mathfrak{S}_K$,

$$\begin{aligned}\|\hat{g} - g^\star\|_2^2 \ \le\ & 6 \inf_M \left\{\|g^\star - g^{\mathbf{Q}^\star,\mathbf{f}^\star_M}\|_2^2 + \mathrm{pen}(N, M)\right\} + A_1^\star \frac{x}{N} \\ & + 18 C_{\mathcal{F},2}^6 (2\|\mathbf{Q}^\star - \mathbb{P}_\tau \hat{\mathbf{Q}}_N \mathbb{P}_\tau^\top\|_F^2 + \|\pi^\star - \mathbb{P}_\tau \hat{\pi}\|_2^2).\end{aligned}$$

Here, $\mathbb{P}_\tau$ *is the permutation matrix associated to* τ.

The important fact in this oracle inequality is that the minimal possible penalty is of order M/N (up to logarithmic terms) and not M^3/N as is usually the case when estimating a joint density of three random variables, so that we get a minimax rate adaptive estimator of the joint density $g^\star$.

4.2 Main Result

The problem is now to deduce from Theorem 4 a result on $\|f_k^\star - \hat{f}_k\|_2^2$, $k = 1,\dots,K$. This is the cornerstone of our work: we prove that, under a technical assumption on the parameters of the unknown HMM, a direct lower bound links $\|\hat{g} - g^\star\|_2^2$ to $\sum_{k=1}^K \|f_1^\star - \hat{f}_k\|_2^2$, up to some positive constant. Let us now describe the assumption and comment on its genericity.

For any $\mathbf{f} \in \mathcal{F}^K$, define $G(\mathbf{f})$ the $K \times K$ matrix with coefficients $G(\mathbf{f})_{i,j} = \langle f_i, f_j\rangle$, $i, j = 1,\dots,K$. Notice that under the assumption [**H4**], $G(\mathbf{f}^\star)$ is positive definite. Let $\mathbf{Q}$

be a transition matrix verifying **[H1]**-**[H2]** and let A_Q be the diagonal matrix having the stationary distribution π of $\mathbf{Q}$ on the diagonal. We shall now define a quadratic form with coefficients depending on $\mathbf{Q}$ and $G(\mathbf{f})$. If U is a $K \times K$ matrix such that $U\mathbf{1}_K = 0$,

$$\begin{aligned}\mathcal{D} := \sum_{i,j=1}^{K} \Big\{ & (\mathbf{Q}^T A_Q U G(\mathbf{f}) U^T A_Q \mathbf{Q})_{i,j} (G(\mathbf{f}))_{i,j} (\mathbf{Q} G(\mathbf{f}) \mathbf{Q}^T)_{i,j} \\ & + (\mathbf{Q}^T A_Q G(\mathbf{f}) A_Q \mathbf{Q})_{i,j} (UG(\mathbf{f})U^T)_{i,j} (\mathbf{Q} G(\mathbf{f}) \mathbf{Q}^T)_{i,j} \\ & + (\mathbf{Q}^T A_Q G(\mathbf{f}) A_Q \mathbf{Q})_{i,j} (G(\mathbf{f}))_{i,j} (\mathbf{Q} U G(\mathbf{f}) U^T \mathbf{Q}^T)_{i,j} \Big\} \\ +2 \sum_{i,j} \Big\{ & (\mathbf{Q}^T A_Q U G(\mathbf{f}) A_Q \mathbf{Q})_{i,j} (UG(\mathbf{f}))_{j,i} (\mathbf{Q} G(\mathbf{f}) \mathbf{Q}^T)_{i,j} \\ & + (\mathbf{Q}^T A_Q U G(\mathbf{f}) A_Q \mathbf{Q})_{i,j} (\mathbf{Q} U G(\mathbf{f}) \mathbf{Q}^T)_{j,i} (G(\mathbf{f}))_{i,j} \\ & + (UG(\mathbf{f}))_{i,j} (\mathbf{Q} U G(\mathbf{f}) \mathbf{Q}^T)_{j,i} (\mathbf{Q}^T A_Q G(\mathbf{f}) A_Q \mathbf{Q})_{i,j} \Big\}\end{aligned}$$

defines a semidefinite positive quadratic form $\mathcal{D}$ in the coefficients $U_{i,j}$, $i = 1, \dots, K$, $j = 1, \dots, K-1$. The determinant of this quadratic form is a polynomial in the coefficients of the matrices $\mathbf{Q}$, A_Q and $G(\mathbf{f})$. Since the coefficients of A_Q are rational functions of the coefficients of the matrix $\mathbf{Q}$, this determinant is also a rational function of the coefficients of the matrices $\mathbf{Q}$ and $G(\mathbf{f})$. Define $H(\mathbf{Q}, G(\mathbf{f}))$ the numerator of the determinant. Then $H(\mathbf{Q}, G(\mathbf{f}))$ is a polynomial in the coefficients of the matrices $\mathbf{Q}$ and $G(\mathbf{f})$. Our assumption will be:

$$\textbf{[HD]} \quad H(\mathbf{Q}^\star, G(\mathbf{f}^\star)) \neq 0.$$

Since H is a polynomial function of $Q^\star_{i,j}$, $i = 1, \dots, K$, $j = 1, \dots, K-1$, and $\langle f_i^\star, f_j^\star \rangle$, $i, j = 1, \dots, K$, the assumption **[HD]** is generically satisfied. We have been able to prove that **[HD]** always holds in the case $K = 2$. We were only able to prove this result by direct computation, it is given in Section 8.4.

Lemma 5 *Assume $K = 2$. Then for all $\mathbf{Q}^\star$ and $\mathbf{f}^\star$ such that* **[H1]**-**[H4]** *hold, one has $H(\mathbf{Q}^\star, G(\mathbf{f}^\star)) > 0$.*

Notice now that, when **[HD]** and **[H1]**-**[H3]** hold, it is possible to define a compact neighborhood $\mathcal{V}$ of $\mathbf{Q}^\star$ such that, for all $\mathbf{Q} \in \mathcal{V}$, $H(\mathbf{Q}, G(\mathbf{f}^\star)) \neq 0$, **[H1]**-**[H3]** hold for $\mathbf{Q}$ and $T_{\mathbf{Q}} \subset T_{\mathbf{Q}^\star}$.

For any $\mathbf{h} \in \left(\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)\right)^K$, define $\|\mathbf{h}\|^2_{\mathbf{Q}} := \min_{\tau \in T_{\mathbf{Q}}} \{\sum_{k=1}^K \|h_k + f_k^\star - f^\star_{\tau(k)}\|_2^2\}$. Denote $\|\mathbf{h}\|_2^2 := \{\sum_{k=1}^K \|h_k\|_2^2\}$. We may now state the theorem which is the cornerstone of our main result.

Theorem 6 *Assume* **[H1]**-**[H4]** *and* **[HD]**. *Let $\mathcal{K}$ be a closed bounded subset of $\left(\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)\right)^K$ such that if $\mathbf{h} \in \mathcal{K}$, then $\int h_i d\mathcal{L}^D = 0$, $i = 1, \dots, K$. Let $\mathcal{V}$ be a compact neighborhood of $\mathbf{Q}^\star$ such that, for all $\mathbf{Q} \in \mathcal{V}$, $H(\mathbf{Q}, G(\mathbf{f}^\star)) \neq 0$,* **[H1]**-**[H3]** *holds for $\mathbf{Q}$ and $T_{\mathbf{Q}} \subset T_{\mathbf{Q}^\star}$. Then there exists a positive constant $c(\mathcal{K}, \mathcal{V}, \mathfrak{F}^\star)$ such that*

$$\forall \mathbf{h} \in \mathcal{K}, \ \forall \mathbf{Q} \in \mathcal{V}, \quad \|g^{\mathbf{Q}, \mathbf{f}^\star + \mathbf{h}} - g^{\mathbf{Q}, \mathbf{f}^\star}\|_2 \geq c(\mathcal{K}, \mathcal{V}, \mathfrak{F}^\star) \|\mathbf{h}\|_{\mathbf{Q}^\star}.$$

This theorem is proved in Section 8.3.

We are now ready to prove our main result on the penalized least squares estimator of the emission densities. The following theorem gives an oracle inequality for the estimators of the emission distributions provided the penalty is adequately chosen. It is proved in Section 8.5.

Theorem 7 (Adaptive estimation) *Assume* **[H1]**-**[H4]**, **[HF]** *and* **[HD]**. *Assume also that for all M, $\mathbf{f}^\star_M \in \mathcal{F}^K$. Let $\mathcal{V}$ be a compact neighborhood of $\mathbf{Q}^\star$ such that, for all $\mathbf{Q} \in \mathcal{V}$, $H(\mathbf{Q}, G(\mathbf{f}^\star)) \neq 0$ and* **[H1]**-**[H3]** *holds for $\mathbf{Q}$. Then, there exists a positive constant $A^\star$ (depending on $\mathcal{V}$, $\mathbf{f}^\star$, $C_{\mathcal{F},2}$ and $C_{\mathcal{F},\infty}$) and positive constants N_0 and $\rho^\star$ (depending on $C_{\mathcal{F},2}$ and $C_{\mathcal{F},\infty}$ (**Scenario A**) or on $\mathbf{Q}^\star$, $C_{\mathcal{F},2}$ and $C_{\mathcal{F},\infty}$ (**Scenario B**)) such that, if*

$$\mathrm{pen}(N, M) \geq \rho^\star \frac{M \log N}{N}$$

then for all $x > 0$, for all $N \geq N_0$, for any permutation $\tau_N \in \mathfrak{S}_K$, with probability larger than $1 - (e-1)^{-1} e^{-x} - \mathbb{P}\left(\mathbb{P}_{\tau_N} \hat{\mathbf{Q}} \mathbb{P}^T_{\tau_N} \notin \mathcal{V}\right)$, there exists $\tau \in T_{\mathbf{Q}^\star}$ such that

$$\begin{aligned}\sum_{k=1}^K \|f^\star_{\tau(k)} - \hat{f}_{\tau_N(k)}\|_2^2 \leq A^\star \Big[\inf_M \Big\{ \sum_{k=1}^K \|f_k^\star - f^\star_{M,k}\|_2^2 + \mathrm{pen}(N, M) \Big\} \\ + \|\mathbf{Q}^\star - \mathbb{P}_{\tau_N} \hat{\mathbf{Q}} \mathbb{P}^\top_{\tau_N}\|_F^2 + \|\pi^\star - \mathbb{P}_{\tau_N} \hat{\pi}\|_2^2 + \frac{x}{N} \Big].\end{aligned}$$

Remark 8 *As usual in HMM or mixture model, it is only possible to estimate the model up to label switching of the hidden states, this is the meaning of the permutation τ_N.*

Remark 9 *An important consequence of the theorem is that a right choice of the penalty leads to a rate minimax adaptive estimator up to a $\log N$ term, see Corollary 10 below. For this purpose, one has to choose an estimator $\hat{\mathbf{Q}}$ of $\mathbf{Q}^\star$ which is, up to label switching, consistent with controlled rate. One possible choice is a spectral estimator.*

To apply Theorem 7 one has to choose an estimator $\hat{\mathbf{Q}}$ with controlled behavior, to be able to evaluate the probability of the event $\{\mathbb{P}_{\tau_N} \hat{\mathbf{Q}} \mathbb{P}_{\tau_N} \in \mathcal{V}\}$ and the rate of convergence of $\mathbb{P}_{\tau_N} \hat{\mathbf{Q}} \mathbb{P}_{\tau_N}$ and $\mathbb{P}_{\tau_N} \hat{\pi}$. One possibility is to use the spectral estimator described in Section 5. To get the following result (proved in Section 8.6), we propose to use the spectral estimator with, for each N, the dimension M_N chosen such that $\eta_3(\Phi_{M_N}) = O\big((\log N)^{1/4}\big)$, see Section 5 for a definition of η_3.

Corollary 10 *With this choice of $\hat{\mathbf{Q}}$, under the assumptions of Theorem 7 , there exists a sequence of permutations $\tau_N \in \mathfrak{S}_K$ such that as N tends to infinity,*

$$\mathbb{E}\left[\sum_{k=1}^K \|f_k^\star - \hat{f}_{\tau_N(k)}\|_2^2\right] = O\left(\inf_{M'} \left\{ \sum_{k=1}^K \|f_k^\star - f^\star_{M',k}\|_2^2 + \mathrm{pen}(N, M') \right\} + \frac{\log N}{N}\right).$$

Thus, choosing $\mathrm{pen}(N, M) = \rho M \log N / N$ for a large ρ leads to the minimax asymptotic rate of convergence up to a power of $\log N$. Indeed, standard results in approximation theory (see DeVore and Lorentz (1993) for instance) show that one can upper bound the approximation error $\|f_k^\star - f^\star_{M,k}\|_2$ by $\mathcal{O}(M^{-\frac{s}{D}})$ where $s > 0$ denotes a regularity parameter.

Then the trade-off is obtained for $M^{\frac{1}{D}} \sim (N/\log N)^{\frac{1}{2s+D}}$, which leads to the quasi-optimal rate $(N/\log N)^{-\frac{s}{2s+D}}$ for the nonparametric estimation when the minimal smoothness of the emission densities is s. Notice that the algorithm automatically selects the best M leading to this rate.

To implement the estimator, it remains to choose a value for ρ in the penalty. The calibration of this parameter is a classical issue and could be the subject of a full paper. In practice one can use the slope heuristic as in Baudry et al. (2012).

5. Nonparametric Spectral Method

This section is devoted to a short description of the nonparametric spectral method for sake of completeness: we describe the algorithm, and give the results we need to support the use of spectral estimators to initialize our algorithm. A detailed study of the nonparametric spectral method is given in De Castro et al. (2015).

The following procedure (see Algorithm 1) describes a tractable approach to estimate the transition matrix in a way that can be used for the penalized least squares estimator of the emission densities, and also for the estimation of the projections of the emission densities that may be used to initialize the least squares algorithm. The procedure is based on recent developments in parametric estimation of HMMs. For each fixed M, we estimate the projection of the emission distributions on the basis Φ_M using the spectral method proposed in Anandkumar et al. (2012). As the authors of the latter paper explain, this allows further to estimate the transition matrix (we use a modified version of their estimator), and we set the estimator of the stationary distribution as the stationary distribution of the estimator of the transition matrix. The computation of those estimators is particularly simple: it is based on one SVD, some matrix inversions and one diagonalization. One can prove, with overwhelming probability, all matrix inversions and the diagonalization can be done rightfully, see De Castro et al. (2015). In the following, when A is a $(p \times q)$ matrix with $p \geq q$, $A^\top$ denotes the transpose matrix of A, $A(k,l)$ its (k,l)th entry, $A(.,l)$ its lth column and $A(k,.)$ its kth line. When v is a vector of size p, we denote by $\mathfrak{Diag}[v]$ the diagonal matrix with diagonal entries v_i and, by abuse of notation, $\mathfrak{Diag}[v] = \mathfrak{Diag}[v^\top]$.

We now state a result which allows to derive the asymptotic properties of the spectral estimators. Let us define:

$$\eta_3^2(\Phi_M) := \sup_{y,y' \in \mathcal{Y}^3} \sum_{a,b,c=1}^{M} (\varphi_a(y_1)\varphi_b(y_2)\varphi_c(y_3) - \varphi_a(y_1')\varphi_b(y_2')\varphi_c(y_3'))^2.$$

Note that in the examples **(Spline)**, **(Trig.)** and **(Wav.)** we have

$$\eta_3(\Phi_M) \leq C_\eta M^{\frac{3}{2}}$$

where $C_\eta > 0$ is a constant. The following theorem is proved in De Castro et al. (2015). Its statement concerns **(Scenario B)** (same chain sampling) and the interested reader may consult De Castro et al. (2015) for its statement under **(Scenario A)**.

Algorithm 1: Nonparametric spectral estimation of HMMs

Data: An observed chain $(Y_1, \ldots, Y_N)$ and a number of hidden states K.
Result: Spectral estimators $\hat{\pi}$, $\hat{\mathbf{Q}}$ and $(\hat{f}_{M,k})_{k \in \mathcal{X}}$.

[Step 1] Consider the following empirical estimators: For any a, b, c in $\{1, \ldots, M\}$,
$\hat{\mathbf{L}}_M(a) := \frac{1}{N}\sum_{s=1}^N \varphi_a(Y_1^{(s)})$, $\hat{\mathbf{M}}_M(a,b,c) := \frac{1}{N}\sum_{s=1}^N \varphi_a(Y_1^{(s)})\varphi_b(Y_2^{(s)})\varphi_c(Y_3^{(s)})$,
$\hat{\mathbf{N}}_M(a,b) := \frac{1}{N}\sum_{s=1}^N \varphi_a(Y_1^{(s)})\varphi_b(Y_2^{(s)})$, $\hat{\mathbf{P}}_M(a,c) := \frac{1}{N}\sum_{s=1}^N \varphi_a(Y_1^{(s)})\varphi_c(Y_3^{(s)})$.

[Step 2] Let $\hat{\mathbf{U}}$ be the $M \times K$ matrix of orthonormal right singular vectors of $\hat{\mathbf{P}}_M$ corresponding to its top K singular values.

[Step 3] Form the matrices for all $b \in \{1, \ldots, M\}$,
$\hat{\mathbf{B}}(b) := (\hat{\mathbf{U}}^\top \hat{\mathbf{P}}_M \hat{\mathbf{U}})^{-1} \hat{\mathbf{U}}^\top \hat{\mathbf{M}}_M(.,b,.)\hat{\mathbf{U}}$.

[Step 4] Set Θ a $(K \times K)$ random unitary matrix uniformly drawn and form the matrices for all $k \in \{1, \ldots, K\}$, $\hat{\mathbf{C}}(k) := \sum_{b=1}^M (\hat{\mathbf{U}}\Theta)(b,k)\hat{\mathbf{B}}(b)$.

[Step 5] Compute $\hat{\mathbf{R}}$ a $(K \times K)$ unit Euclidean norm columns matrix that diagonalizes the matrix $\hat{\mathbf{C}}(1)$: $\hat{\mathbf{R}}^{-1}\hat{\mathbf{C}}(1)\hat{\mathbf{R}} = \mathfrak{Diag}[(\hat{\Lambda}(1,1), \ldots, \hat{\Lambda}(1,K))]$.

[Step 6] Set for all $k, k' \in \mathcal{X}$, $\hat{\Lambda}(k,k') := (\hat{\mathbf{R}}^{-1}\hat{\mathbf{C}}(k)\hat{\mathbf{R}})(k',k')$ and $\hat{\mathbf{O}}_M := \hat{\mathbf{U}}\Theta\hat{\Lambda}$.

[Step 7] Consider the emission laws estimator $\tilde{\mathbf{f}} := (\tilde{f}_{M,k})_{k \in \mathcal{X}}$ defined by for all $k \in \mathcal{X}$, $\tilde{f}_{M,k} := \sum_{m=1}^M \hat{\mathbf{O}}_M(m,k)\varphi_m$.

[Step 8] Set $\tilde{\pi} := \left(\hat{\mathbf{U}}^\top \hat{\mathbf{O}}_M\right)^{-1} \hat{\mathbf{U}}^\top \hat{\mathbf{L}}_M$.

[Step 9] Consider the transition matrix estimator:

$$\hat{\mathbf{Q}} := \Pi_{\mathrm{TM}}\Big((\hat{\mathbf{U}}^\top \hat{\mathbf{O}}_M \mathfrak{Diag}[\tilde{\pi}])^{-1} \hat{\mathbf{U}}^\top \hat{\mathbf{N}}_M \hat{\mathbf{U}} (\hat{\mathbf{O}}_M^\top \hat{\mathbf{U}})^{-1} \Big),$$

where Π_{TM} denotes the projection onto the convex set of transition matrices, and define $\hat{\pi}$ as the stationary distribution of $\hat{\mathbf{Q}}$.

Theorem 11 (Spectral estimators) *Assume that* **[H1]**-**[H4]** *hold. Then, there exist positive constant numbers $M_{\mathfrak{F}^\star}$, $x(\mathbf{Q}^\star)$, $\mathcal{C}(\mathbf{Q}^\star, \mathfrak{F}^\star)$ and $\mathbf{N}(\mathbf{Q}^\star, \mathfrak{F}^\star)$ such that the following holds. For any $x \geq x(\mathbf{Q}^\star)$, for any $\delta \in (0,1)$, for any $M \geq M_{\mathfrak{F}^\star}$, there exists a permutation $\tau_M \in \mathfrak{S}_K$ such that the spectral method estimators $\hat{f}_{M,k}$, $\hat{\pi}$ and $\hat{\mathbf{Q}}$ satisfy: For any $N \geq \mathbf{N}(\mathbf{Q}^\star, \mathfrak{F}^\star)\eta_3(\Phi_M)^2 x(-\log\delta)/\delta^2$, with probability greater than $1 - 2\delta - 4e^{-x}$,*

$$\|f_{M,k}^\star - \hat{f}_{M,\tau_M(k)}\|_2 \leq \mathcal{C}(\mathbf{Q}^\star, \mathfrak{F}^\star)\frac{\sqrt{-\log\delta}}{\delta}\frac{\eta_3(\Phi_M)}{\sqrt{N}}\sqrt{x}\,,$$

$$\|\pi^\star - \mathbb{P}_{\tau_M}\hat{\pi}\|_2 \leq \mathcal{C}(\mathbf{Q}^\star, \mathfrak{F}^\star)\frac{\sqrt{-\log\delta}}{\delta}\frac{\eta_3(\Phi_M)}{\sqrt{N}}\sqrt{x}\,,$$

$$\|\mathbf{Q}^\star - \mathbb{P}_{\tau_M}\hat{\mathbf{Q}}\mathbb{P}_{\tau_M}^\top\| \leq \mathcal{C}(\mathbf{Q}^\star, \mathfrak{F}^\star)\frac{\sqrt{-\log\delta}}{\delta}\frac{\eta_3(\Phi_M)}{\sqrt{N}}\sqrt{x}\,.$$

6. Numerical experiments

6.1 General description

In this section we present the numerical performances of our method. We recall that the experimenter knows nothing about the underlying hidden Markov model but the number of hidden states K. In this set of experiments, we consider the regular histogram basis or the trigonometric basis for estimating emission laws given by beta laws from a single chain observation of length $N = 50{,}000$.

Our procedure is based on the computation of the empirical least squares estimators $\hat{g}_M$ defined as minimizers of the empirical contrast γ_N on the space $\mathcal{S}(\hat{\mathbf{Q}}, M)$ where $\hat{\mathbf{Q}}$ is an estimator of the transition matrix (for instance the spectral estimation of the transition matrix). Since the function γ_N is non-convex, we use a second order approach estimating a positive definite matrix (using a covariance matrix) within an iterative procedure called CMAES for Covariance Matrix Adaptation Evolution Strategy, see Hansen (2006). Using this latter algorithm, we search for the minimum of γ_N with starting point the spectral estimation of the emission laws.

Then, we estimate the size of the model thanks to

$$\hat{M}(\rho) \in \arg \min_{M=1,\dots,M_{\max}} \left\{ \gamma_N(\hat{g}_M) + \rho \frac{M \log N}{N} \right\} , \tag{3}$$

where the penalty term ρ has to be tuned and the maximum size of the model $M_{\max}$ can be set by the experimenter in a data-driven procedure.

Indeed, we shall apply the slope heuristic to adjust the penalty term and to choose $M_{\max}$. As presented in Baudry et al. (2012), the minimum contrast function $M \mapsto \gamma_N(\hat{g}_M)$ should have a linear behavior for large values of M. The experimenter has to consider $M_{\max}$ large enough in order to observe this linear stabilization, as depicted in Figure 2. The slope of the linear interpolation is then $(\hat{\rho}/2) \log N/N$ (recall that the sample size N is fixed here) where $\hat{\rho}$ is the slope heuristic choice on how ρ should be tuned. Another procedure (theoretically equivalent) consists in plotting the function $\rho \mapsto \hat{M}(\rho)$ which is a non-increasing piecewise constant function. The estimated $\hat{\rho}$ is such that the largest drop (called "dimension jump") of this function occurs at point $\hat{\rho}/2$. We illustrate this procedure in Figure 3 where one can clearly point the jump and deduce the size $\hat{M}$.

To summarize, our procedure reads as follows.

1. For all $M \leq M_{\max}$, compute the spectral estimations $(\hat{\mathbf{Q}}, \hat{\pi})$ of the transition matrix and its stationary distribution and the spectral estimation $\tilde{\mathbf{f}}$ of the emission laws. This is straightforward using the procedure described by **[Step1-9]** in Section 5.

2. For all $M \leq M_{\max}$, compute a minimum $\hat{g}_M$ of the empirical contrast function γ_N using "Covariance Matrix Adaptation Evolution Strategy", see Hansen (2006). Use the estimation $\tilde{\mathbf{f}}$ of the spectral method as a starting point of CMAES.

3. Tune the penalty term using the slope heuristic procedure and select $\hat{M}$.

4. Return the emission laws of the solution of point (2) for $M = \hat{M}$.

Note that the size M of the projection space for the spectral estimator has been set as the one chosen by the slope heuristic for the empirical least squares estimators.

All the codes of the numerical experiments are available at `https://mycore.core-cloud.net/public.php?service=files&t=44459ccb178a3240cfb8712f27a28d75`. We shall indicate that the slope heuristic has been done using CAPUSHE, the Matlab graphical user interface presented in Baudry et al. (2012).

6.2 Complexity

A crucial step of our method lies in computing the empirical least squares estimators $\hat{g}_M$. One may struggle to compute $\hat{g}_M$ since the function γ_N is non-convex. It follows that an acceptable procedure must start from a good approximation of $\hat{g}_M$. This is done by the spectral method. Observe that the key leitmotiv throughout this paper is a two steps estimation procedure that starts by the spectral estimator. This latter has rate of convergence of the order of $N^{-s/(2s+3)}$ and seems to be a good candidate to initialize an iterative scheme that will converge towards $\hat{g}_M$. It follows that the main consuming operations in our algorithm are the following steps.

- The computation of the tensor $\hat{\mathbf{M}}_M$ of the empirical law of three consecutive observations where we use three loops of size M and one loop of size N so the complexity is $\mathcal{O}(NM^3)$,
- The singular value decomposition of $\hat{\mathbf{P}}_M$ in the spectral method (complexity: $\mathcal{O}(M^3)$),
- The computation of the minimum of the empirical contrast function: cost of one evaluation of the empirical contrast function $\mathcal{O}(K^3M^3) = \mathcal{O}(M^3)$ times the number $f(M, K)$ of evaluations while minimizing the empirical contrast. Recall that we start from the spectral estimator solution to get the minimum so a constant number of evaluation is enough in practice, say `stopeval` $=1e4$ using CMAES.

We have to compute the minimal contrast value for all models of size $M = 1, \dots, M_{\max}$ where $M_{\max}$ has to be chosen so that one can apply the slope heuristic. We deduce that the overall complexity of our algorithm is $\mathcal{O}((f(M_{\max}, K)K^3 \vee N)M_{\max}^4)$ where $f(M_{\max}, K)$ is the number of evaluations of γ_N while minimizing the empirical contrast. Since we use the spectral estimator as a starting point of the minimization of the empirical contrast, we believe that $f(M_{\max}, K)$ can be considered as constant, say $1e4$. Note that the upper bound $M_{\max}$ has to be large enough in order to observe a linear stabilization of $M \mapsto \hat{g}_M$, see Baudry et al. (2012) for instance. Moreover, recall that the trade-off between the approximation bias and the penalty term (accounting for the standard error of the empirical law) is obtained for $M \sim (N/\log N)^{\frac{D}{2s+D}}$ where $s > 0$ denotes the minimal smoothness parameter of the emission laws. In order to properly apply the slope heuristic, it is enough to consider models with this order of magnitude, so that $M_{\max} = \mathcal{O}((N/\log N)^{\frac{D}{2s+D}})$. It follows that the overall complexity of our procedure can be expressed in terms of the minimal smoothness parameter s of the emission laws as

$$\text{Complexity} = \mathcal{O}(N^{1+\frac{4D}{2s+D}}) ,$$

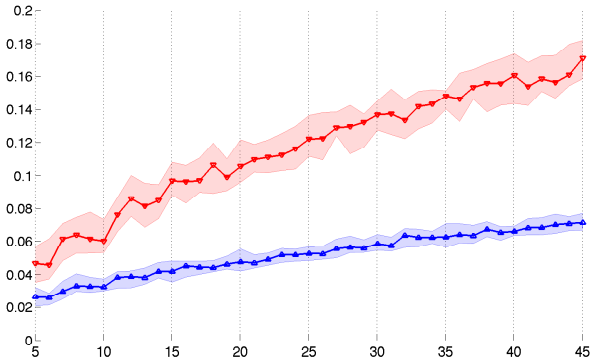


Figure 1: Variance comparison of the spectral and empirical least squares estimators. The upper curve (in red) present the performance (median value of the variance over 40 iterations) of the spectral method while the lower curve (in blue) the performance of the empirical least squares estimator. For each curve, we have plotted a shaded box plot representing the first and third quartiles.

as soon as $K = \mathcal{O}(N^{1/3})$ which is a reasonable assumption. Nevertheless, this theoretical bound is unknown for the practitioner since it involves the unknown minimal smoothness parameter $s > 0$. For chains of length $\mathcal{O}(1e5)$, we have witnessed that one can afford a maximal model size $M_{\max} \leq 50$ and this allows to consider problems where typical sizes of M ranges between 1 and 50. All numerical experiments of this paper fall in this frame.

6.3 Comparison of the Variances

The quadratic loss can be expressed as a variance term and a bias term as follows

$$\forall 1 \leq k \leq K,\ \forall M \geq 0, \quad \|f_k^\star - \hat{f}_k\|_2^2 = \|\hat{f}_k - f_{M,k}^\star\|_2^2 + \|f_k^\star - f_{M,k}^\star\|_2^2$$

where $f_{M,k}^\star$ is the orthogonal projection of $f_k^\star$ on $\mathfrak{P}_M$ and $\hat{f}_k$ is any estimator such that $\hat{f}_k$ belongs to $\mathfrak{P}_M$. Note that the bias term $\|f_k^\star - f_{M,k}^\star\|_2$ does not depend on the estimator $\hat{f}_k$. Hence, the variance term

$$\text{Variance}_M(\hat{f}) := \min_{\tau \in \mathfrak{S}_K} \max_{1 \leq k \leq K} \|\hat{f}_k - f_{M,\tau(k)}^\star\|_2^2 ,$$

accounts for the performances of the estimator $\hat{f}_k$.

As depicted in Figure 1, we have compared, for each M, the variance terms obtained by the spectral method and the empirical least squares method over 40 iterations on chains of length $N = 5e4$. We have considered $K = 2$ hidden states whose emission variables are distributed with respect to beta laws of parameters $(2, 5)$ and $(4, 2)$. This numerical experiment consolidates the idea that the least squares method significantly improves upon

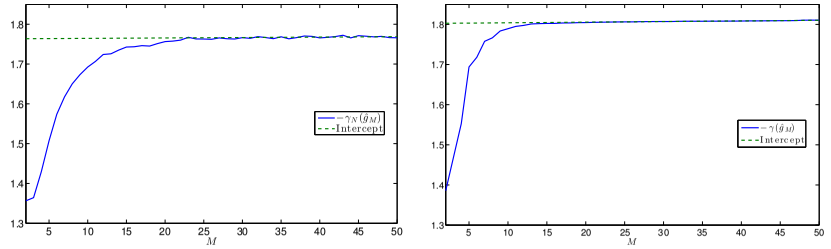


Figure 2: Slope heuristic to choose M: the experimenter may observe a linear stabilization of the empirical contrast γ_N for estimating beta emission laws of parameters $(2, 5)$ and $(4, 2)$. We have $K = 2$ hidden states and $N = 5e4$ samples along a single chain. On the left panel we have used the trigonometric basis as approximation space, the stabilization occurs on the points $M = 30$ to $M = 50$ and the interpolation of the slope leads to $\hat{M} = 23$. On the right panel we have considered the trigonometric basis, the stabilization occurs on the points $M = 20$ to $M = 50$ and it leads to $\hat{M} = 21$.

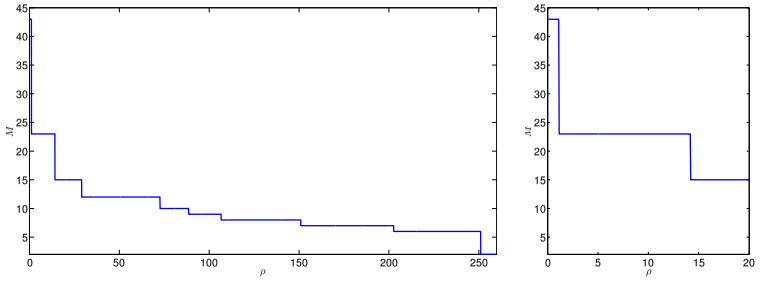


Figure 3: Slope heuristic to choose M: the experimenter observes the largest drop of the function $\rho \mapsto \hat{M}(\rho)$ at 1.1 so that $\hat{\rho} = 2.2$ and $\hat{M} = 23$. We have $K = 2$ hidden states and a single chain of length $N = 5e4$. We have used the histogram basis as approximation space.

the spectral method. Indeed, even for small values of M, one may see in Figure 1 that the variance term is divided by a constant factor.

6.4 Histogram Basis and Trigonometric Basis as Approximation Spaces

An illustrative example of our method can be given using the histogram basis (regular basis with M bins) or the trigonometric basis. In the following experiments, we have $K = 2$

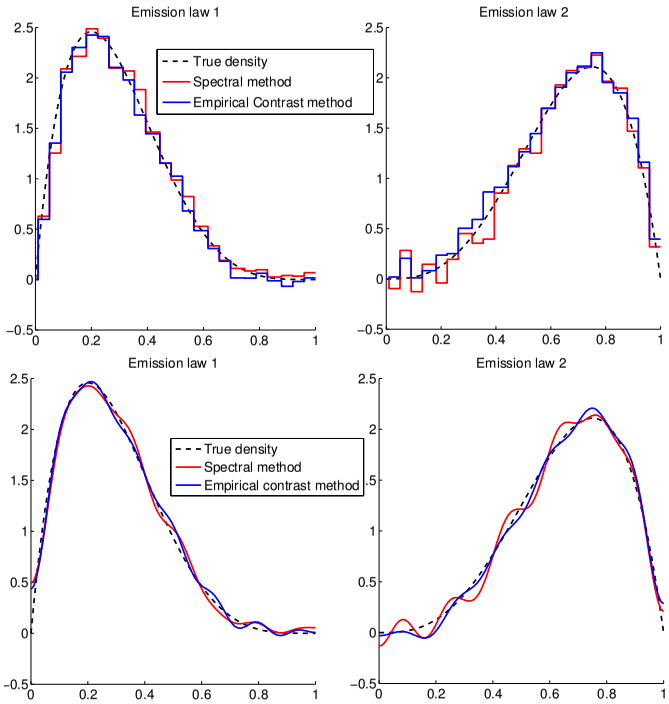


Figure 4: Estimators of the emissions densities (beta laws of parameters $(2,5)$ and $(4,2)$) from the observation of a single chain of length $N = 5e4$. On the top panels, we have used the histogram basis ($\hat{M} = 23$). On the bottom panels, we have considered the trigonometric basis ($\hat{M} = 21$).

hidden states and emission laws given by beta laws of parameters $(2,5)$ and $(4,2)$. Recall we observe a single chain of length $N = 5e4$.

We begin with the computation of the minimum contrast function $M \mapsto \gamma(\hat{g}_M)$, as depicted in Figure 2. Observe that the slope of this function unquestionably stabilizes at a critical value refer to as $\hat{\rho}/2$ in both the histogram and the trigonometric case. This leads to an adaptive choice of $\hat{M} = 23$ for the histogram basis and $\hat{M} = 21$ for the trigonometric basis, see Figures 2 and 3.

Furthermore, one can see on Figure 4 that our method also qualitatively improves upon the spectral method in both the histogram and the trigonometric case.

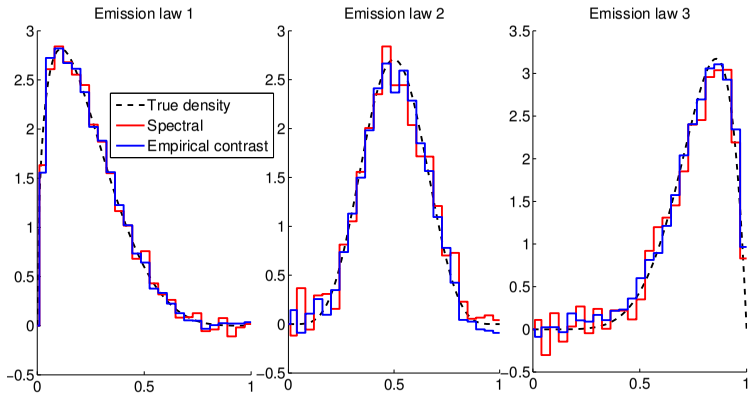


Figure 5: Estimation of three densities given by beta laws of parameters $(1.5,5)$, $(6,6)$ and $(7,2)$ from a single chain of length $N = 5e4$. We have used the histogram basis and we have found $\hat{M} = 25$ using the slope heuristic.

6.5 Three States

Our method can be performed for $K > 2$ as illustrated in Figure 5. In this example $K = 3$, the sample size is $N = 5e4$ and the emission laws are three beta distributions with parameters $(1.5,5)$, $(6,6)$ and $(7,2)$. Note that the number of hidden states K does not really impact on the complexity of the algorithm as we have seen in Section 6.2.

In this example, we were able to observe a linear stabilization of the minimum contrast function. The slope heuristic procedure led to an adaptive choice $\hat{M} = 25$.

7. Discussion

We have proposed a penalized least squares method to estimate the emission densities of the hidden chain when the transition matrix of the hidden chain is full rank and the emission probability distributions are linearly independent. The algorithm may be initialized using spectral estimators. The obtained estimators are adaptive rate optimal up to a log factor, where adaptivity is upon the family of emission densities. The results hold under an assumption on the parameter that holds generically. We have proved that this assumption is always verified when there are two hidden states. We did not find a general argument to prove that the assumption always holds when $K > 2$, and a natural question is to ask if, when the number of hidden states is $K > 2$, this assumption is also always verified.
It is proved in Alexandrovich et al. (2016) that identifiability holds as soon as $f_1^\star, \ldots, f_K^\star$ are distinct densities. The identifiability is obtained in that case using the marginal distribution of dimension $2K+1$, that is the marginal distribution of $Y_1, \ldots, Y_{2K+1}$. Thus, to get consistent estimators, one needs to use the joint distribution of $2K+1$ consecutive observations.

Though linear independence is generically satisfied, one may wonder what happens when emission densities are not far to be linearly dependent. Simulations in Lehéricy (2015) show that estimation becomes harder. In those practical situations where estimation becomes difficult, it is observed that the Gram matrix of $f_1^\star, \dots, f_K^\star$ has an eigenvalue close to 0. On the theoretical side, the proof of Theorem 6 uses the linear independence of the emission densities by using that Gram matrices are positive. An interesting problem would be to investigate if it is possible to estimate the emission densities with the classical adaptive rate for density estimation when the emission densities are linearly dependent (though all distinct). It is possible using model selection to get the classical rate for the estimation of the density of $2K+1$ consecutive observations, but it does not seem obvious to see whether this rate can be transferred to the estimators of the emission densities. This is the subject of further work, see Lehéricy (2016).
Another question arising from our work is whether it is possible to adapt to different smoothnesses of the emission densities.

8. Proofs

8.1 Proof of lemma 3

In Hsu et al. (2012) it is proved that when **[H1]**, **[H2]**, **[H3]** hold and when the rank of the matrix $\mathbf{O}_M := (\langle \varphi_m, f_k^\star \rangle_{1\le m\le M, 1\le k\le K}$ is K, the knowledge of the tensor $\mathbf{M}_M$ given by $\mathbf{M}_M(a,b,c) = \mathbb{E}(\varphi_a(Y_1)\varphi_b(Y_2)\varphi_c(Y_3))$ for all a,b,c in $\{1,\dots,M\}$ allows to recover $\mathbf{O}_M$ and $\mathbf{Q}$ up to relabelling of the hidden states. Thus, when **[H1]**, **[H2]**, **[H3]** and **[H4]** hold, the knowledge of $g^{\mathbf{Q},\mathbf{f}^\star}$ is equivalent to the knowledge of the sequence $(\mathbf{M}_M)_M$, which allows to recover $\mathbf{Q}$ and the sequence $(\mathbf{O}_M)_M$, up to relabelling of the hidden states, which allows to recover $\mathbf{f}^\star = (f_1^\star, \dots, f_K^\star)$ up to relabelling of the hidden states, thanks to (1). See also Gassiat et al. (2016).

8.2 Proof of Theorem 4

Throughout the proof N is fixed, and we write γ (instead of γ_N) for the contrast function.

8.2.1 Beginning of the proof: algebraic manipulations

Let us fix some M and some permutation τ. Using the definitions of $\hat{g}_M$ and $\hat{M}$, we can write

$$\gamma(\hat{g}_{\hat{M}}) + \mathrm{pen}(N,\hat{M}) \le \gamma(\hat{g}_M) + \mathrm{pen}(N,M) \le \gamma(g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}}) + \mathrm{pen}(N,M)\,,$$

where $\mathbf{f}^\star_{M,\tau^{-1}} = (f^\star_{M,\tau^{-1}(1)}, \dots, f^\star_{M,\tau^{-1}(K)})$ (here we use that $\mathbf{f}^\star_{M,\tau^{-1}} \in \mathcal{F}^K$). But we can compute for all functions t_1, t_2,

$$\gamma(t_1) - \gamma(t_2) = \|t_1 - g^\star\|_2^2 - \|t_2 - g^\star\|_2^2 - 2\nu(t_1 - t_2)\,,$$

where ν is the centered empirical process

$$\nu(t) = \frac{1}{N}\sum_{s=1}^N t(Y_1^{(s)}, Y_2^{(s)}, Y_3^{(s)}) - \int t g^\star\,.$$

This gives

$$\|\hat{g}_{\hat{M}} - g^\star\|_2^2 \le \|g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}} - g^\star\|_2^2 + 2\nu(\hat{g}_{\hat{M}} - g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}}) + \mathrm{pen}(N,M) - \mathrm{pen}(N,\hat{M}) \qquad (4)$$

Now, we denote by $B_M = \|g^{\mathbf{Q}^\star,\mathbf{f}^\star_M} - g^\star\|_2^2$ a bias term and we notice that $g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}} = g^{\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top,\mathbf{f}^\star_M}$. Then

$$\begin{aligned}\|g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}} - g^\star\|_2^2 &\le 2\|g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}} - g^{\mathbf{Q}^\star,\mathbf{f}^\star_M}\|_2^2 + 2\|g^{\mathbf{Q}^\star,\mathbf{f}^\star_M} - g^\star\|_2^2 \\ &\le 2\|g^{\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top,\mathbf{f}^\star_M} - g^{\mathbf{Q}^\star,\mathbf{f}^\star_M}\|_2^2 + 2B_M\,.\end{aligned}$$

But, using Schwarz inequality, $\|g^{\mathbf{Q}_1,\mathbf{f}^\star_M} - g^{\mathbf{Q}_2,\mathbf{f}^\star_M}\|_2^2$ can be bounded by

$$\begin{aligned}&\sum_{m_1,m_2,m_3=1}^M \Big| \sum_{k_1,k_2,k_3=1}^K (\pi_1(k_1)\mathbf{Q}_1(k_1,k_2)\mathbf{Q}_1(k_2,k_3) - \pi_2(k_1)\mathbf{Q}_2(k_1,k_2)\mathbf{Q}_2(k_2,k_3)) \\ &\qquad \langle f^\star_{k_1}, \varphi_{m_1}\rangle\langle f^\star_{k_2}, \varphi_{m_2}\rangle\langle f^\star_{k_3}, \varphi_{m_3}\rangle\Big|^2 \\ &\le \left(\sum_{k_1,k_2,k_3=1}^K (\pi_1(k_1)\mathbf{Q}_1(k_1,k_2)\mathbf{Q}_1(k_2,k_3) - \pi_2(k_1)\mathbf{Q}_2(k_1,k_2)\mathbf{Q}_2(k_2,k_3))^2\right) \\ &\quad \sum_{m_1,m_2,m_3=1}^M \sum_{k_1,k_2,k_3=1}^K \Big|\langle f^\star_{k_1}, \varphi_{m_1}\rangle\langle f^\star_{k_2}, \varphi_{m_2}\rangle\langle f^\star_{k_3}, \varphi_{m_3}\rangle\Big|^2 \\ &\le 3K^3 C^6_{\mathcal{F},2}\left(\|\pi_1 - \pi_2\|_2^2 + 2\|\mathbf{Q}_1 - \mathbf{Q}_2\|_F^2\right) \qquad (5)\end{aligned}$$

so that

$$\|g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}} - g^\star\|_2^2 \le 6K^3 C^6_{\mathcal{F},2}\left(\|\mathbb{P}_\tau\hat{\pi} - \pi^\star\|_2^2 + 2\|\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top - \mathbf{Q}^\star\|_F^2\right) + 2B_M\,.$$

Next we set $S_M = \cup_{\mathbf{Q}}\mathcal{S}(\mathbf{Q},M)$ and

$$Z_M = \sup_{t\in S_M}\left[\frac{|\nu(t-g^\star)|}{\|t-g^\star\|_2^2 + x_M^2}\right]$$

for x_M to be determined later. Then

$$\begin{aligned}\nu(\hat{g}_{\hat{M}} - g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}}) &= \nu(\hat{g}_{\hat{M}} - g^\star) + \nu(g^\star - g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}}) \\ &\le Z_{\hat{M}}(\|\hat{g}_{\hat{M}} - g^\star\|_2^2 + x_{\hat{M}}^2) + Z_M(\|g^{\hat{\mathbf{Q}},\mathbf{f}^\star_{M,\tau^{-1}}} - g^\star\|_2^2 + x_M^2).\end{aligned}$$

Denoting by $R_{\hat{M}} = \|\hat{g}_{\hat{M}} - g^\star\|_2^2$ the squared risk, (4) becomes

$$\begin{aligned} R_{\hat{M}} &\leq 6K^3C_{\mathcal{F},2}^6\left(\|\mathbb{P}_\tau\hat{\pi} - \pi^\star\|_2^2 + 2\|\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top - \mathbf{Q}^\star\|_F^2\right) + 2B_M + 2Z_{\hat{M}}(R_{\hat{M}} + x_{\hat{M}}^2) \\ &\quad + 2Z_M\left(6K^3C_{\mathcal{F},2}^6\left(\|\mathbb{P}_\tau\hat{\pi} - \pi^\star\|_2^2 + 2\|\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top - \mathbf{Q}^\star\|_F^2\right) + 2B_M + x_M^2\right) \\ &\quad + 2\mathrm{pen}(N,M) - \mathrm{pen}(N,\hat{M}) - \mathrm{pen}(N,M)\,, \\ R_{\hat{M}}(1 - 2Z_{\hat{M}}) &\leq (2 + 4Z_M)B_M + 2\mathrm{pen}(N,M) \\ &\quad + (1 + 2Z_M)6K^3C_{\mathcal{F},2}^6\left(\|\mathbb{P}_\tau\hat{\pi} - \pi^\star\|_2^2 + 2\|\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top - \mathbf{Q}^\star\|_F^2\right) \\ &\quad + 2\sup_{M'}(2Z_{M'}x_{M'}^2 - \mathrm{pen}(N,M'))\,. \end{aligned}$$

To conclude it is then sufficient to establish that, with probability larger than $1 - (e-1)^{-1}e^{-x}$, it holds

$$\sup_{M'} Z_{M'} \leq \frac{1}{4} \quad \text{and} \quad \sup_{M'}(2Z_{M'}x_{M'}^2 - \mathrm{pen}(N,M')) \leq A\frac{x}{N}\,,$$

with A a constant depending only on $\mathbf{Q}^\star$ and $\mathbf{f}^\star$ and not on N, M, x. Thus we will have, for any M, with probability larger than $1 - (e-1)^{-1}e^{-x}$,

$$\begin{aligned} \frac{1}{2}R_{\hat{M}} &\leq 3B_M + 2\mathrm{pen}(N,M) + 2A\frac{x}{N} \\ &\quad + 9C_{\mathcal{F},2}^6\left(\|\mathbb{P}_\tau\hat{\pi} - \pi^\star\|_2^2 + 2\|\mathbb{P}_\tau\hat{\mathbf{Q}}\mathbb{P}_\tau^\top - \mathbf{Q}^\star\|_F^2\right) \end{aligned}$$

which is the announced result.

The heart of the proof is then the study of Z_M. We introduce u_M a projection of $g^\star$ on S_M and we split Z_M into two terms: $Z_M \leq 4Z_{M,1} + Z_{M,2}$ with

$$\begin{cases} Z_{M,1} = \sup_{t\in S_M}\left[\dfrac{|\nu(t-u_M)|}{\|t-u_M\|_2^2 + 4x_M^2}\right] \\ Z_{M,2} = \dfrac{|\nu(u_M - g^\star)|}{\|u_M - g^\star\|_2^2 + x_M^2} \end{cases}$$

Indeed u_M verifies: for all $t \in S_M$,

$$\|u_M - g^\star\|_2 \leq \|t - g^\star\|_2 \quad \text{and} \quad \|u_M - t\|_2 \leq 2\|t - g^\star\|_2\,.$$

8.2.2 Deviation inequality for $Z_{M,2}$

Bernstein's inequality (24) for HMMs (see Appendix A) gives, with probability larger than $1 - e^{-z}$:

$$|\nu(u_M - g^\star)| \leq 2\sqrt{2c^\star\|u_M - g^\star\|_2^2\|g^\star\|_\infty\frac{z}{N}} + 2\sqrt{2}c^\star\|u_M - g^\star\|_\infty\frac{z}{N}\,.$$

Then, using $a^2 + b^2 \geq 2ab$, with probability larger than $1 - e^{-z}$:

$$\frac{|\nu(u_M - g^\star)|}{\|u_M - g^\star\|_2^2 + x_M^2} \leq 2\sqrt{2c^\star\|g^\star\|_\infty}\frac{1}{2x_M}\sqrt{\frac{z}{N}} + 2\sqrt{2}c^\star\frac{\|u_M\|_\infty + \|g^\star\|_\infty}{x_M^2}\frac{z}{N}\,.$$

But any function t in S_M can be written

$$t = \sum_{k_1,k_2,k_3=1}^K \pi(k_1)\mathbf{Q}(k_1,k_2)\mathbf{Q}(k_2,k_3)f_{k_1}\otimes f_{k_2}\otimes f_{k_3}\,,$$

with $f_k \in \mathcal{F}$ for $k = 1,\dots,K$, so that $\sup_{t\in S_M}\|t\|_\infty \leq C_{\mathcal{F},\infty}^3$. Then, with probability larger than $1 - e^{-z_M - z}$

$$Z_{M,2} \leq \sqrt{2c^\star\|g^\star\|_\infty}\sqrt{\frac{z_M + z}{x_M^2 N}} + 4\sqrt{2}c^\star C_{\mathcal{F},\infty}^3\frac{z_M + z}{x_M^2 N}\,. \tag{6}$$

8.2.3 Deviation inequality for $Z_{M,1}$

We shall first study the term $\sup_{t\in B_\sigma}|\nu(t-u_M)|$ where

$$B_\sigma = \{t \in S_M, \|t - u_M\|_2 \leq \sigma\}.$$

Remark that, for all $t \in \mathcal{S}(\mathbf{Q}, M)$,

$$\|t\|_2^2 \leq \sum_{k_1,k_2,k_3=1}^K \pi^2(k_1)\mathbf{Q}^2(k_1,k_2)\mathbf{Q}^2(k_2,k_3)\sum_{k_1,k_2,k_3=1}^K C_{\mathcal{F},2}^2C_{\mathcal{F},2}^2C_{\mathcal{F},2}^2 \leq K^3C_{\mathcal{F},2}^6.$$

Then, if $t \in B_\sigma$, $\|t - u_M\|_2 \leq \sigma \wedge 2K^{3/2}C_{\mathcal{F},2}^3$. Notice also that for all $t \in S_M$, $\|t - u_M\|_\infty \leq 2C_{\mathcal{F},\infty}^3$. Now Proposition 13 in Appendix A (applied to a countable dense set in B_σ) gives that for any measurable set A such that $\mathbb{P}(A) > 0$,

$$\mathbb{E}^A(\sup_{t\in B_\sigma}|\nu(t-u_M)|) \leq C^\star\left[\frac{E}{N} + \sigma\sqrt{\frac{1}{N}\log\left(\frac{1}{\mathbb{P}(A)}\right)} + \frac{2C_{\mathcal{F},\infty}^3}{N}\log\left(\frac{1}{\mathbb{P}(A)}\right)\right],$$

and

$$E = \sqrt{N}\int_0^\sigma\sqrt{H(u)\wedge N}du + (2C_{\mathcal{F},\infty}^3 + 2K^{3/2}C_{\mathcal{F},2}^3)H(\sigma)\,.$$

Here, for any integrable random variable Z, $E^A[Z]$ denotes $E[Z\mathbb{1}_A]/\mathbb{P}(A)$.

We shall compute E later and find σ_M and φ such that

$$\forall\sigma \geq \sigma_M \qquad E \leq (1 + 2C_{\mathcal{F},\infty}^3 + 2K^{3/2}C_{\mathcal{F},2}^3)\varphi(\sigma)\sqrt{N}. \tag{7}$$

(see Section 8.2.4). We then use Lemma 4.23 in Massart (2007) to write (for $x_M \geq \sigma_M$)

$$\mathbb{E}^A\left(\sup_{t\in S_M}\left[\frac{|\nu(t-u_M)|}{\|t-u_M\|_2^2 + 4x_M^2}\right]\right) \leq \frac{C^\star}{x_M^2}\left[C\frac{\varphi(2x_M)}{\sqrt{N}} + 2x_M\sqrt{\frac{1}{N}\log\left(\frac{1}{\mathbb{P}(A)}\right)} + \frac{2C_{\mathcal{F},\infty}^3}{N}\log\left(\frac{1}{\mathbb{P}(A)}\right)\right]$$

Finally, Lemma 2.4 in Massart (2007) ensures that, with probability $1 - e^{-z_M - z}$:

$$Z_{M,1} = \sup_{t\in S_M}\left[\frac{|\nu(t-u_M)|}{\|t-u_M\|_2^2 + 4x_M^2}\right] \leq C^\star\left[C\frac{\varphi(2x_M)}{x_M^2\sqrt{N}} + 2\sqrt{\frac{z_M + z}{x_M^2 N}} + 2C_{\mathcal{F},\infty}^3\frac{z_M + z}{x_M^2 N}\right]. \tag{8}$$

8.2.4 Computation of the entropy and function φ

The definition of H given in Proposition 13 shows that $H(\delta)$ is bounded by the classical bracketing entropy for $\mathbf{L}^2$ distance at point $\delta/C^3_{\mathcal{F},\infty}$ (where $C^3_{\mathcal{F},\infty}$ bounds the sup norm of $g^\star$): $H(\delta) \leq H(\delta/C^3_{\mathcal{F},\infty}, S_M, \mathbf{L}^2)$. We denote by $N(u, S, \mathbf{L}^2) = e^{H(u,S,\mathbf{L}^2)}$ the minimal number of brackets of radius u to cover S. Recall that when t_1 and t_2 are real valued functions, the bracket $[t_1, t_2]$ is the set of real valued functions t such that $t_1(\cdot) \leq t(\cdot) \leq t_2(\cdot)$, and the radius of the bracket is $\|t_2 - t_1\|_2$. Now, observe that $S_M = \cup_{\mathbf{Q}} \mathcal{S}(\mathbf{Q}, M)$ is a set of mixtures of parametric functions. Denoting $\mathbf{k} = (k_1, k_2, k_3)$, S_M is included in

$$\left\{ \sum_{\mathbf{k}\in\{1,\dots,K\}^3} \mu(\mathbf{k}) f_{k_1} \otimes f_{k_2} \otimes f_{k_3},\ \mu \geq 0,\ \sum_{\mathbf{k}\in\{1,\dots,K\}^3} \mu(\mathbf{k}) = 1, \right.$$
$$\left. f_{k_i} \in \mathcal{F} \cap \mathrm{Span}(\varphi_1, \dots, \varphi_M),\ i = 1, 2, 3 \right\} .$$

Set

$$\mathcal{A} = \{ f_1 \otimes f_2 \otimes f_3, f_i \in \mathcal{F} \cap \mathrm{Span}(\varphi_1, \dots, \varphi_M),\ i = 1, 2, 3 \}.$$

Then following the proof in Appendix A of Bontemps and Toussile (2013), we can prove

$$N(\varepsilon, S_M, \mathbf{L}^2) \leq \left(\frac{C_1}{\varepsilon}\right)^{K^3-1} \left[N\left(\frac{\varepsilon}{3}, \mathcal{A}, \mathbf{L}^2\right) \right]^{K^3} . \tag{9}$$

where C_1 depends on K and $C_{\mathcal{F},2}$. Denote $\mathcal{B} = \mathcal{F} \cap \mathrm{Span}(\varphi_1, \dots, \varphi_M)$. Let $a = (a_m)_{1\leq m\leq M} \in \mathbb{R}^M$ and $b = (b_m)_{1\leq m\leq M} \in \mathbb{R}^M$ such that $a_m < b_m$, $m = 1, \dots, M$. For each $m = 1, \dots, M$ and $y \in \mathcal{Y}$, let

$$u_m(y) = \begin{cases} a_m & \text{if } \varphi_m(y) \geq 0 \\ b_m & \text{otherwise} \end{cases}$$
$$v_m(y) = a_m + b_m - u_m(y).$$

Then, if $(c_m)_{1\leq m\leq M} \in \mathbb{R}^M$ is such that for all $m = 1, \dots, M$, $a_m \leq c_m \leq b_m$, then

$$U^1_{a,b}(y) := \sum_{m=1}^M u_m(y)\varphi_m(y) \leq \sum_{m=1}^M c_m\varphi_m(y) \leq \sum_{m=1}^M v_m(y)\varphi_m(y) = U^2_{a,b}(y).$$

Moreover,

$$\|U^2_{a,b} - U^1_{a,b}\|_2^2 = \| \sum_{m=1}^M |b_m - a_m|.|\varphi_m| \|_2^2$$
$$\leq M\|b - a\|_2^2$$

using Cauchy-Schwarz inequality. Thus, one may cover $\mathcal{B}$ with brackets of form $[U^1_{a,b}, U^2_{a,b}]$. Also, for $i = 1, 2$,

$$\|U^i_{a,b}\|_2^2 \leq \| \sum_{m_i=1}^M |b_m + a_m|.|\varphi_m| \|_2^2$$
$$\leq 2M(\|a\|_2^2 + \|b\|_2^2).$$

If now for some a^i, b^i in $\mathbb{R}^M$, $f_i \in [U^1_{a^i,b^i}, U^2_{a^i,b^i}]$, $i = 1, 2, 3$, then

$$f_1 \otimes f_2 \otimes f_3 \in [V, W]$$

with

$$V = \min\{ U^{i_1}_{a^1,b^1} U^{i_2}_{a^2,b^2} U^{i_3}_{a^3,b^3},\ i_1, i_2, i_3 \in \{1, 2\} \}$$

and

$$W = \max\{ U^{i_1}_{a^1,b^1} U^{i_2}_{a^2,b^2} U^{i_3}_{a^3,b^3},\ i_1, i_2, i_3 \in \{1, 2\} \},$$

pointwise. Moreover, one can see that

$$\begin{aligned} |W - V| &\leq \left|U^2_{a^1,b^1} - U^1_{a^1,b^1}\right| \max_{j_1,j_2\in\{1,2\}} \left|U^{j_1}_{a^2,b^2}\right| . \left|U^{j_2}_{a^3,b^3}\right| \\ &\quad + \left|U^2_{a^2,b^2} - U^1_{a^2,b^2}\right| \max_{j_1,j_2\in\{1,2\}} \left|U^{j_1}_{a^1,b^1}\right| . \left|U^{j_2}_{a^3,b^3}\right| \\ &\quad + \left|U^2_{a^3,b^3} - U^1_{a^3,b^3}\right| \max_{j_1,j_2\in\{1,2\}} \left|U^{j_1}_{a^1,b^1}\right| . \left|U^{j_2}_{a^2,b^2}\right| \\ &\leq \sum_{i=1}^3 \left|U^2_{a^i,b^i} - U^1_{a^i,b^i}\right| \prod_{j\neq i} \left(\left|U^1_{a^j,b^j}\right| + \left|U^2_{a^j,b^j}\right| \right) \end{aligned}$$

so that

$$\begin{aligned} \|W - V\|_2^2 &\leq 12 \sum_{i=1}^3 \left\|U^2_{a^i,b^i} - U^1_{a^i,b^i}\right\|_2^2 \prod_{j\neq i} \left(\left\|U^1_{a^j,b^j}\right\|_2^2 + \left\|U^2_{a^j,b^j}\right\|_2^2 \right) \\ &\leq 48M^3 \sum_{i=1}^3 \|b^i - a^i\|_2^2 \prod_{j\neq i} \left(\|a^j\|_2^2 + \|b^j\|_2^2 \right) \\ &\leq 192M^3 C^4_{\mathcal{F},2} \sum_{i=1}^3 \|b^i - a^i\|_2^2. \end{aligned}$$

Thus one may cover $\mathcal{A}$ by covering the ball of radius $C_{\mathcal{F},2}$ in $\mathbb{R}^M$ with hypercubes $[a, b]$, for which $\|a\|_2$, $\|b\|_2$ are less than $C_{\mathcal{F},2}$. To get a bracket with radius u, it is enough that $\|b^i - a^i\|_2^2 \leq u^2/(576M^3C^4_{\mathcal{F},2})$, $i = 1, 2, 3$. We finally obtain that

$$N\left(u, \mathcal{A}, \mathbf{L}^2\right) \leq \left(\frac{48\sqrt{3}M^{3/2}C^3_{\mathcal{F},2}}{u} \right)^{3M} . \tag{10}$$

We deduce from (9) and (10) that

$$N(u, S_M, \mathbf{L}^2) \leq \left(\frac{C_1}{u}\right)^{K^3-1} \left(\frac{48\sqrt{3}M^{3/2}C^3_{\mathcal{F},2}}{u} \right)^{3MK^3} ,$$

and then

$$H(u, S_M, \mathbf{L}^2) \leq (K^3 - 1) \log(\frac{C_1}{u}) + 3MK^3 \log\left(\frac{C_2 M^{3/2}}{u} \right) ,$$

with C_2 depending on K and $C_{\mathcal{F},2}$. To conclude we use that $\int_0^\sigma \sqrt{\log\left(\frac{1}{x}\right)}dx \leq \sigma(\sqrt{\pi} + \sqrt{\log\left(\frac{1}{\sigma}\right)})$, see Baudry et al. (2012). Finally we can write for $\sigma \leq M^{3/2}$:

$$\int_0^\sigma \sqrt{H(u)}du \leq C_3\sqrt{M}\sigma\left(1+\sqrt{\log\left(\frac{M^{3/2}}{\sigma}\right)}\right),$$

where C_3 depends on K, $C_{\mathcal{F},2}$ and $C_{\mathcal{F},\infty}$. Set

$$\varphi(x) = C_3\sqrt{M}x\left(1+\sqrt{\log\left(\frac{M^{3/2}}{x}\right)}\right)$$

The function φ is increasing on $]0, M^{3/2}]$, and $\varphi(x)/x$ is decreasing. Moreover $\varphi(\sigma) \geq \int_0^\sigma \sqrt{H(u)}du$ and $\varphi^2(\sigma) \geq \sigma^2 H(\sigma)$.

8.2.5 End of the proof, choice of parameters

As soon as $N \geq C_3^2/M^2 := N_0$, we may define σ_M as the solution of equation $\varphi(x) = \sqrt{N}x^2$. Then, for all $\sigma \geq \sigma_M$,

$$H(\sigma) \leq \frac{\varphi(\sigma)^2}{\sigma^2} \leq \frac{\varphi(\sigma)}{\sigma}\sigma\sqrt{N}.$$

This yields, for all $\sigma \geq \sigma_M$,

$$E \leq (1 + 2C_{\mathcal{F},\infty}^3 + 2K^{3/2}C_{\mathcal{F},2}^3)\varphi(\sigma)\sqrt{N},$$

which was required in (7).

Moreover $\frac{\varphi(2x_M)}{x_M\sqrt{N}} \leq 2\sigma_M$ as soon as $x_M \geq \sigma_M$. Combining (8) and (6), we obtain, with probability $1 - e^{-z_M - z}$:

$$Z_M \leq C^{\star\star}\left[\frac{\sigma_M}{x_M} + \sqrt{\frac{z_M+z}{x_M^2 N}} + \frac{z_M+z}{x_M^2 N}\right],$$

where $C^{\star\star}$ depends on K, $C_{\mathcal{F},2}$, $C_{\mathcal{F},\infty}$, $\mathbf{Q}^\star$. Now let us choose $x_M = \theta^{-1}\sqrt{\sigma_M^2 + \frac{z_M+z}{N}}$ with θ such that $2\theta + \theta^2 \leq (C^{\star\star})^{-1}/4$. This choice entails: $x_M \geq \theta^{-1}\sigma_M$ and $x_M^2 \geq \theta^{-2}\frac{z_M+z}{N}$. Then with probability $1 - e^{-z_M - z}$:

$$Z_M \leq C^{\star\star}(\theta + \theta + \theta^2).$$

We now choose $z_M = M$ which implies $\sum_{M\geq 1} e^{-z_M} = (e-1)^{-1}$. Then, with probability $1 - (e-1)^{-1}e^{-z}$,

$$\forall M \qquad Z_M \leq C^{\star\star}(2\theta + \theta^2) \leq \frac{1}{4},$$

and for all M,

$$\begin{aligned} Z_M x_M^2 &\leq C^{\star\star}\left[\sigma_M x_M + x_M\sqrt{\frac{z_M+z}{N}} + \frac{z_M+z}{N}\right] \\ &\leq C^{\star\star}\theta^{-1}\left(\sigma_M + \sqrt{\frac{z_M+z}{N}}\right)^2 + C^{\star\star}\frac{z_M+z}{N}. \end{aligned}$$

Then, with probability $1 - (e-1)^{-1}e^{-z}$, for all M,

$$Z_M x_M^2 - C^{\star\star}\left(2\theta^{-1}\sigma_M^2 + (2\theta^{-1}+1)\frac{M}{N}\right) \leq C^{\star\star}(2\theta^{-1}+1)\frac{z}{N}.$$

Then the result is proved as soon as

$$\mathrm{pen}(N,M) \geq 2C^{\star\star}\left(2\theta^{-1}\sigma_M^2 + (2\theta^{-1}+1)\frac{M}{N}\right). \tag{11}$$

It remains to get an upper bound for σ_M. Recall that σ_M is defined as the solution of equation $C_3\sqrt{M}x(1+\sqrt{\log\left(\frac{M^3}{x}\right)}) = \sqrt{N}x^2$. Then we obtain that for some C_4

$$\sigma_M \leq C_4\sqrt{\frac{M}{N}}(1+\sqrt{\log(N)}),$$

and (11) holds as soon as

$$\mathrm{pen}(N,M) \geq \rho^\star\frac{M\log(N)}{N}$$

for some constant $\rho^\star$ depending on $C_{\mathcal{F},2}$ and $C_{\mathcal{F},\infty}$ (**Scenario A**) or on $\mathbf{Q}^\star$, $C_{\mathcal{F},2}$ and $C_{\mathcal{F},\infty}$ (**Scenario B**).

8.3 Proof of Theorem 6

For any $\mathbf{h} \in \mathcal{K}^K$ and $\mathbf{Q} \in \mathcal{V}$, denote $N(\mathbf{Q},\mathbf{h}) = \|g^{\mathbf{Q},\mathbf{f}^\star+\mathbf{h}} - g^{\mathbf{Q},\mathbf{f}^\star}\|_2^2$. What we want to prove is that

$$c := c(\mathcal{K},\mathcal{V},\mathfrak{F}^\star)^2 := \inf_{\mathbf{Q}\in\mathcal{V},\mathbf{h}\in\mathcal{K}^K,\|\mathbf{h}\|_2\neq 0}\frac{N(\mathbf{Q},\mathbf{h})}{\|\mathbf{h}\|_{\mathbf{Q}}^2} > 0.$$

One can compute:

$$\begin{aligned} N(\mathbf{Q},\mathbf{h}) = &\sum_{k_1,k_2,k_3,k_1',k_2',k_3'=1}^{K}\pi(k_1)\mathbf{Q}(k_1,k_2)\mathbf{Q}(k_2,k_3)\pi(k_1')\mathbf{Q}(k_1',k_2')\mathbf{Q}(k_2',k_3') \\ &\left(\prod_{i=1}^3\langle f_{k_i}^\star + h_{k_i}, f_{k_i'}^\star + h_{k_i'}\rangle + \prod_{i=1}^3\langle f_{k_i}^\star, f_{k_i'}^\star\rangle - \prod_{i=1}^3\langle f_{k_i}^\star + h_{k_i}, f_{k_i'}^\star\rangle - \prod_{i=1}^3\langle f_{k_i}^\star, f_{k_i'}^\star + h_{k_i'}\rangle\right). \end{aligned}$$

Let $\mathbf{u} = (u_1,\ldots,u_K)$ be such that u_i, $i = 1,\ldots,K$, is the orthogonal projection of h_i on the subspace of $\mathbf{L}^2(\mathcal{Y},\mathcal{L}^D)$ spanned by $f_1^\star\ldots,f_K^\star$. Then

$$N(\mathbf{Q},\mathbf{h}) = N(\mathbf{Q},\mathbf{u}) + M(\mathbf{Q},\mathbf{u},\mathbf{h}-\mathbf{u}) \tag{12}$$

where, for any $\mathbf{a} = (a_1, \ldots, a_K) \in \mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)^K$,

$$M(\mathbf{Q}, \mathbf{u}, \mathbf{a}) = \sum_{k_1,k_2,k_3,k_1',k_2',k_3'=1}^{K} \pi(k_1)\mathbf{Q}(k_1,k_2)\mathbf{Q}(k_2,k_3)\pi(k_1')\mathbf{Q}(k_1',k_2')\mathbf{Q}(k_2',k_3')$$
$$\left(\prod_{i=1}^{3}\langle a_{k_i}, a_{k_i'}\rangle + \sum_{i=1}^{3}\langle a_{k_i}, a_{k_i'}\rangle \prod_{j\neq i}\langle f^\star_{k_j} + u_{k_j}, f^\star_{k_j'} + u_{k_j'}\rangle + \sum_{i=1}^{3}\langle f^\star_{k_i} + u_{k_i}, f^\star_{k_i'} + u_{k_i'}\rangle \prod_{j\neq i}\langle a_{k_j}, a_{k_j'}\rangle\right).$$

Let $A = \mathfrak{Diag}[\pi]$ with π the stationary distribution of $\mathbf{Q}$. Then $M(\mathbf{Q}, \mathbf{u}, \mathbf{a})$ may be rewritten as:

$$\begin{aligned}
M(\mathbf{Q}, \mathbf{u}, \mathbf{a}) \quad = \sum_{i,j=1}^{K} \quad & \langle (\mathbf{Q}^T A\mathbf{a})_i, (\mathbf{Q}^T A\mathbf{a})_j\rangle\langle a_i, a_j\rangle\langle (\mathbf{Q}\mathbf{a})_i, (\mathbf{Q}\mathbf{a})_j\rangle \\
& + \langle (\mathbf{Q}^T A\mathbf{a})_i, (\mathbf{Q}^T A\mathbf{a})_j\rangle\langle (\mathbf{f}^\star + \mathbf{u}_i), (\mathbf{f}^\star + \mathbf{u})_j\rangle\langle (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_j\rangle \\
& + \langle (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_j\rangle\langle a_i, a_j\rangle\langle (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_j\rangle \\
& + \langle (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_j\rangle\langle (\mathbf{f}^\star + \mathbf{u})_i, (\mathbf{f}^\star + \mathbf{u})_j\rangle\langle (\mathbf{Q}\mathbf{a})_i, (\mathbf{Q}\mathbf{a})_j\rangle \\
& + \langle (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_j\rangle\langle a_i, a_j\rangle\langle (\mathbf{Q}\mathbf{a})_i, (\mathbf{Q}\mathbf{a})_j\rangle \\
& + \langle (\mathbf{Q}^T A\mathbf{a})_i, (\mathbf{Q}^T A\mathbf{a})_j\rangle\langle (\mathbf{f}^\star + \mathbf{u})_i, (\mathbf{f}^\star + \mathbf{u})_j\rangle\langle (\mathbf{Q}\mathbf{a})_i, (\mathbf{Q}\mathbf{a})_j\rangle \\
& + \langle (\mathbf{Q}^T A\mathbf{a})_i, (\mathbf{Q}^T A\mathbf{a})_j\rangle\langle a_i, a_j\rangle\langle (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_j\rangle.
\end{aligned}$$

All terms in this sum are non negative. Let us prove it for the first one, the argument for the others is similar. Define V the $K \times K$ matrix given by

$$V_{i,j} = \langle (\mathbf{Q}^T A\mathbf{a})_i, (\mathbf{Q}^T A\mathbf{a})_j\rangle\langle (\mathbf{Q}\mathbf{a})_i, (\mathbf{Q}\mathbf{a})_j\rangle, \quad i,j = 1, \ldots, K.$$

V is the Hadamard product of two Gram matrices which are non negative, thus V is itself non negative by the Schur product Theorem, see Schur (1911), and

$$\sum_{i,j=1}^{K} V_{i,j}\langle a_i, a_j\rangle = \int \mathbf{a}(y)^T V \mathbf{a}(y) dy \geq 0.$$

Thus we have that $M(\mathbf{Q}, \mathbf{u}, \mathbf{a})$ is lower bounded by one term of the sum so that

$$M(\mathbf{Q}, \mathbf{u}, \mathbf{a}) \geq \sum_{i,j=1}^{K} \langle (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))_j\rangle\langle a_i, a_j\rangle\langle (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_i, (\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))_j\rangle.$$

The minimal eigenvalue of the Hadamard product of two non negative matrices is lower bounded by the product of the minimal eigenvalues of each matrix, and we get

$$M(\mathbf{Q}, \mathbf{u}, \mathbf{a}) \geq \left(\min_{i=1,\ldots,K} \lambda_i(\mathbf{Q}^T A(\mathbf{f}^\star + \mathbf{u}))\right)\left(\min_{i=1,\ldots,K} \lambda_i(\mathbf{Q}(\mathbf{f}^\star + \mathbf{u}))\right)\|\mathbf{a}\|_2^2 \qquad (13)$$

where $\|\mathbf{a}\|_2^2 = \sum_{k=1}^{k} \|a_k\|_2^2$, and where, if $\mathbf{h} \in \mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)^K$, $\lambda_1(\mathbf{h})$, $\ldots$, $\lambda_K(\mathbf{h})$ are the (non negative) eigenvalues of the Gram matrix of $h_1, \ldots, h_K$.

Let now $(\mathbf{Q}_n, \mathbf{h}_n)_n$ be a sequence in $\mathcal{V} \times \mathcal{K}$ such that $c = \lim_n \frac{N(\mathbf{Q}_n, \mathbf{h}_n)}{\|\mathbf{h}_n\|^2_{\mathbf{Q}^\star}}$. Let $\mathbf{u}_n$ be the vector of the orthogonal projections of the coordinate functions of $\mathbf{h}_n$ on the subspace of $\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D)$ spanned by $f_1^\star \ldots, f_K^\star$. Notice that

$$\|\mathbf{h}_n\|^2_{\mathbf{Q}^\star} = \|\mathbf{u}_n\|^2_{\mathbf{Q}^\star} + \|\mathbf{h}_n - \mathbf{u}_n\|_2^2.$$

Let $C_{\mathcal{K},2}$ be the upper bound of the norm of elements of $\mathcal{K}$. We have, for any $n \geq 1$,

$$\|\mathbf{h}_n\|^2_{\mathbf{Q}^\star} \leq K(C_{\mathcal{K},2} + 2C_{\mathcal{F},2})^2$$

so that for any $n \geq 1$,

$$\|\mathbf{u}_n\|^2_{\mathbf{Q}^\star} \leq K(C_{\mathcal{K},2} + 2C_{\mathcal{F},2})^2.$$

Since $(\mathbf{Q}_n, \mathbf{u}_n)_n$ is a bounded sequence in a finite dimensional space it has a limit point $(\mathbf{Q}, \mathbf{u})$. Now, using (12) and the non negativity of $M(\mathbf{Q}_n, \mathbf{u}_n, \mathbf{h}_n - \mathbf{u}_n)$, we get on the converging subsequence

$$c \geq \lim_{n\to+\infty} \frac{N(\mathbf{Q}_n, \mathbf{u}_n)}{K(C_{\mathcal{K},2} + 2C_{\mathcal{F},2})^2} = \frac{N(\mathbf{Q}, \mathbf{u})}{K(C_{\mathcal{K},2} + 2C_{\mathcal{F},2})^2}.$$

Since $\mathbf{Q} \in \mathcal{V}$, $T_{\mathbf{Q}} \subset T_{\mathbf{Q}^\star}$ so that $\|\mathbf{u}\|_{\mathbf{Q}} \geq \|\mathbf{u}\|_{\mathbf{Q}^\star}$. Thus if $\|\mathbf{u}\|_{\mathbf{Q}^\star} \neq 0$, $\|\mathbf{u}\|_{\mathbf{Q}} \neq 0$, and using Lemma 3, $N(\mathbf{Q}, \mathbf{u}) \neq 0$ so that $c > 0$ in this case.

Consider now the situation where $\|\mathbf{u}\|_{\mathbf{Q}^\star} = 0$. Since $\lim_{n\to+\infty} \|\mathbf{u}_n\|_{\mathbf{Q}^\star} = 0$, there exists n_1 and $\tau \in T_{\mathbf{Q}^\star}$ such that for all $n \geq n_1$, one has $\|\mathbf{u}_n\|^2_{\mathbf{Q}^\star} = \sum_{k=1}^{K} \|u_{n,k} + f_k^\star - f_{\tau(k)}^\star\|_2^2$, and it is possible to exchange the states in the transition matrix using τ so that we just have to consider the situation where $\|\mathbf{u}_n\|^2_{\mathbf{Q}^\star} = \|\mathbf{u}_n\|_2^2$ for large enough n.
Eigenvalues of Gram matrices of functions are continuous in the functions so that using (12) and (13) we get

$$c \geq \lim_{n\to+\infty} \frac{N(\mathbf{Q}_n, \mathbf{u}_n)}{\|\mathbf{u}_n\|_2^2 + \|\mathbf{h}_n - \mathbf{u}_n\|_2^2} + \left(\min_{i=1,\ldots,K} \lambda_i(\mathbf{Q}^T A\mathbf{f}^\star)\right)\left(\min_{i=1,\ldots,K} \lambda_i(\mathbf{Q}\mathbf{f}^\star)\right) \liminf_{n\to+\infty} \frac{\|\mathbf{h}_n - \mathbf{u}_n\|_2^2}{\|\mathbf{u}_n\|_2^2 + \|\mathbf{h}_n - \mathbf{u}_n\|_2^2}.$$

Under assumptions **[H1]** and **[H4]**, $\mathbf{Q}^T A\mathbf{f}^\star$ is a vector of linearly independent functions and $\mathbf{Q}\mathbf{f}^\star$ also, so that

$$\left(\min_{i=1,\ldots,K} \lambda_i(\mathbf{Q}^T A\mathbf{f}^\star)\right)\left(\min_{i=1,\ldots,K} \lambda_i(\mathbf{Q}\mathbf{f}^\star)\right) > 0.$$

Thus, if $\liminf_{n\to+\infty} \frac{\|\mathbf{h}_n - \mathbf{u}_n\|_2^2}{\|\mathbf{u}_n\|_2^2 + \|\mathbf{h}_n - \mathbf{u}_n\|_2^2} > 0$ we obtain $c > 0$.
If now it holds that $\liminf_{n\to+\infty} \frac{\|\mathbf{h}_n - \mathbf{u}_n\|_2^2}{\|\mathbf{u}_n\|_2^2 + \|\mathbf{h}_n - \mathbf{u}_n\|_2^2} = 0$. On a subsequence it holds that $\|\mathbf{h}_n - \mathbf{u}_n\|_2 = o(\|\mathbf{u}_n\|_2)$ and we have

$$c \geq \lim_{n\to+\infty} \frac{N(\mathbf{Q}_n, \mathbf{u}_n)}{\|\mathbf{u}_n\|_2^2} \frac{\|\mathbf{u}_n\|_2^2}{\|\mathbf{u}_n\|_2^2 + \|\mathbf{h}_n - \mathbf{u}_n\|_2^2} = \lim_{n\to+\infty} \frac{N(\mathbf{Q}_n, \mathbf{u}_n)}{\|\mathbf{u}_n\|_2^2} \qquad (14)$$

with $(\mathbf{u}_n)_n$ a sequence of vectors of functions in the finite dimensional space spanned by $f_1^\star, \ldots, f_K^\star$. Writing

$$\frac{(\mathbf{u}_n)_i(y)}{\|\mathbf{u}_n\|_2} = \frac{\|\mathbf{h}_n\|_2}{\|\mathbf{u}_n\|_2}\frac{(\mathbf{u}_n)_i(y)}{\|\mathbf{h}_n\|_2} = \frac{\|\mathbf{h}_n\|_2}{\|\mathbf{u}_n\|_2}\Big[\frac{(\mathbf{h}_n-\mathbf{u}_n)_i+(\mathbf{u}_n)_i(y)}{\|\mathbf{h}_n\|_2} - \frac{(\mathbf{h}_n-\mathbf{u}_n)_i}{\|\mathbf{h}_n\|_2}\Big],$$

it follows that for $i = 1, \ldots, K$,

$$\lim_{n\to+\infty}\int \frac{(\mathbf{u}_n)_i(y)}{\|\mathbf{u}_n\|_2}dy = 0 \tag{15}$$

since for all n and all $i = 1, \ldots, K$, $\int \frac{(\mathbf{h}_n-\mathbf{u}_n)_i+(\mathbf{u}_n)_i(y)}{\|\mathbf{h}_n\|_2}dy = 0$, and it holds that $\frac{\|\mathbf{h}_n\|_2}{\|\mathbf{u}_n\|_2} \to 1$ and $\|\mathbf{h}_n - \mathbf{u}_n\|_2/\|\mathbf{u}_n\|_2 = o(1)$.
Let us return to general considerations on the function $N(\cdot,\cdot)$. As it may be seen from its formula, $N(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}})$ is polynomial in the variables $\tilde{\mathbf{Q}}_{i,j}$, $\langle f_i^\star, f_j^\star\rangle$, $\langle \tilde{h}_i, f_j^\star\rangle$, $\langle \tilde{h}_i, \tilde{h}_j\rangle$, $i, j = 1, \ldots, K$. Let $D(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}})$ denote the part of $N(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}})$ which is homogeneous of degree 2 with respect to the variable $\tilde{\mathbf{h}}$, that is

$$\begin{aligned} D(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}}) \;=\; & \sum_{k_1,k_2,k_3,k_1',k_2',k_3'=1}^{K} \tilde{\pi}(k_1)\tilde{\mathbf{Q}}(k_1,k_2)\tilde{\mathbf{Q}}(k_2,k_3)\tilde{\pi}(k_1')\tilde{\mathbf{Q}}(k_1',k_2')\tilde{\mathbf{Q}}(k_2',k_3') \\ & \Big(\sum_{i=1}^{3}\langle \tilde{h}_{k_i}, \tilde{h}_{k_i'}\rangle \prod_{j\neq i}\langle f_{k_j}^\star, f_{k_j'}^\star\rangle + 2\sum_{i=1}^{3}\langle f_{k_i}^\star, f_{k_i'}^\star\rangle \prod_{j\neq j'\neq i}\langle f_{k_j}^\star, \tilde{h}_{k_j'}\rangle\langle \tilde{h}_{k_{j'}}, f_{k_{j'}'}^\star\rangle\Big). \end{aligned} \tag{16}$$

One gets

$$N(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}}) = D(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}}) + O\left(\|\tilde{\mathbf{h}}\|_2^3\right)$$

where the $O(\cdot)$ depends only on $\mathbf{f}^\star$. Let us first notice that $D(\cdot,\cdot)$ is always non negative. Indeed, since for all $\tilde{\mathbf{Q}} \in \mathcal{V}$ and all $\tilde{\mathbf{h}} \in (\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D))^K$ one has $N(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}}) \geq 0$, it holds

$$\forall \tilde{\mathbf{Q}} \in \mathcal{V},\ \forall \tilde{\mathbf{h}} \in (\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D))^K,\ \frac{D(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}})}{\|\tilde{\mathbf{h}}\|_2^2} + O(\|\tilde{\mathbf{h}}\|_2) \geq 0,$$

so that, since for all $\lambda \in \mathbb{R}$, $D(\tilde{\mathbf{Q}}, \lambda\tilde{\mathbf{h}}) = \lambda^2 D(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}})$,

$$\forall \tilde{\mathbf{Q}} \in \mathcal{V},\ \forall \tilde{\mathbf{h}} \in (\mathbf{L}^2(\mathcal{Y}, \mathcal{L}^D))^K,\ D(\tilde{\mathbf{Q}}, \tilde{\mathbf{h}}) \geq 0. \tag{17}$$

Then we obtain from (14)

$$c \geq \liminf_{n\to+\infty} D(\mathbf{Q}_n, \frac{\mathbf{u_n}}{\|\mathbf{u_n}\|_2}).$$

Let $\mathbf{b} = (b_1, \ldots, b_K)$ be a limit point of the sequence $(\frac{\mathbf{u_n}}{\|\mathbf{u_n}\|_2})_n$. We then have

$$c \geq D(\mathbf{Q}, \mathbf{b}).$$

Now, using (15) we get that

$$\int b_k d\mathcal{L}^D = 0,\ k = 1, \ldots, K.$$

Thus there exists a $K \times K$ matrix U such that $\mathbf{b}^T = U(\mathbf{f}^\star)^T$ and $U\mathbf{1}_k = 0$, and equation (16) leads to

$$\begin{aligned} & D(\mathbf{Q}, \mathbf{b}) = \\ & \sum_{i,j}\Big\{\left(\mathbf{Q}^T AUG^\star U^T A\mathbf{Q}\right)_{i,j}(G^\star)_{i,j}\left(\mathbf{Q}G^\star\mathbf{Q}^T\right)_{i,j} + \left(\mathbf{Q}^T AG^\star A\mathbf{Q}\right)_{i,j}\left(UG^\star U^T\right)_{i,j}\left(\mathbf{Q}G^\star\mathbf{Q}^T\right)_{i,j} \\ & + \left(\mathbf{Q}^T AG^\star A\mathbf{Q}\right)_{i,j}(G^\star)_{i,j}\left(\mathbf{Q}UG^\star U^T\mathbf{Q}^T\right)_{i,j}\Big\} + 2\sum_{i,j}\Big\{\left(\mathbf{Q}^T AUG^\star A\mathbf{Q}\right)_{i,j}(UG^\star)_{j,i}\left(\mathbf{Q}G^\star\mathbf{Q}^T\right)_{i,j} \\ & + \left(\mathbf{Q}^T AUG^\star A\mathbf{Q}\right)_{i,j}\left(\mathbf{Q}UG^\star\mathbf{Q}^T\right)_{j,i}(G^\star)_{i,j} + (UG^\star)_{i,j}\left(\mathbf{Q}UG^\star\mathbf{Q}^T\right)_{j,i}\left(\mathbf{Q}^T AG^\star A\mathbf{Q}\right)_{i,j}\Big\}. \end{aligned}$$

with $G^\star$ the $K \times K$ Gram matrix such that $(G^\star)_{i,j} = \langle f_i^\star, f_j^\star\rangle$, $i = 1, \ldots, K$.
This is the quadratic form $\mathcal{D}$ in $U_{i,j}$, $i = 1, \ldots, K$, $j = 1, \ldots, K-1$ defined in Section 4.2. This quadratic form is non negative, and as soon as it is positive, we get that $c > 0$. But the quadratic form $\mathcal{D}$ is positive as soon as its determinant is non zero, that is if and only if $H(\mathbf{Q}, G(\mathbf{f}^\star)) \neq 0$.

8.4 Proof of Lemma 5

Here we specialize to the situation where $K = 2$. In such a case, $\mathbf{f}^\star = (f_1^\star, f_2^\star)$, and

$$\mathbf{Q}^\star = \begin{pmatrix} 1-p^\star & p^\star \\ q^\star & 1-q^\star \end{pmatrix}$$

for some $p^\star, q^\star$ in $[0,1]$ for which $0 < p^\star < 1,\ 0 < q^\star < 1,\ p^\star \neq 1-q^\star$. Now

$$U = \begin{pmatrix} \alpha & -\alpha \\ \beta & -\beta \end{pmatrix}$$

for some real numbers α and β, and brute force computation gives $D(\mathbf{Q}, \mathbf{b}) = D_{1,1}\alpha^2 + 2D_{1,2}\alpha\beta + D_{2,2}\beta^2$ with, denoting $p = \mathbf{Q}(1,2)$ and $q = \mathbf{Q}(2,1)$:

$$\begin{aligned} \frac{(p+q)^2 D_{1,1}}{q^2} =& 2(1-p)^2\|f_1^\star - f_2^\star\|^2\|f_1^\star\|^2\|(1-p)f_1^\star + pf_2^\star\|^2 + \|(1-p)f_1^\star + pf_2^\star\|^4\|f_1^\star - f_2^\star\|^2 \\ & + 4p(1-p)\left(\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle\right)\langle f_1^\star, f_2^\star\rangle\|f_1^\star - f_2^\star\|^2 \\ & + 2p^2\|f_1^\star - f_2^\star\|^2\|f_2^\star\|^2\|qf_1^\star + (1-q)f_2^\star\|^2 \\ & + 2(1-p)^2\left(\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\right)^2\|f_1^\star\|^2 \\ & + 2p^2\left(\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\right)^2\|f_2^\star\|^2 \\ & + 4p(1-p)\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_2^\star\rangle \\ & + 4(1-p)\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_1^\star - f_2^\star\rangle\|(1-p)f_1^\star + pf_2^\star\|^2 \\ & + 4p\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_2^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle, \end{aligned}$$

$$
\begin{aligned}
\frac{(p+q)^2 D_{2,2}}{p^2} =& 2q^2\|f_1^\star - f_2^\star\|^2\|f_1^\star\|^2\|(1-p)f_1^\star + pf_2^\star\|^2 + \|qf_1^\star + (1-q)f_2^\star\|^4\|f_1^\star - f_2^\star\|^2 \\
&+ 4(1-q)q\left(\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle\right)\langle f_1^\star, f_2^\star\rangle\|f_1^\star - f_2^\star\|^2 \\
&+ 2(1-q)^2\|f_1^\star - f_2^\star\|^2\|f_2^\star\|^2\|qf_1^\star + (1-q)f_2^\star\|^2 \\
&+ 2q^2\left(\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\right)^2\|f_1^\star\|^2 \\
&+ 2(1-q)^2\left(\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\right)^2\|f_2\|^2 \\
&+ 4q(1-q)\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_2^\star\rangle \\
&+ 4q\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle \\
&+ 4(1-q)\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle f_2^\star, f_1^\star - f_2^\star\rangle\|qf_1^\star + (1-q)f_2^\star\|^2,
\end{aligned}
$$

and:

$$
\begin{aligned}
\frac{(p+q)^2 D_{1,2}}{pq} =& 2(1-p)q\|f_1^\star - f_2^\star\|^2\|f_1^\star\|^2\|(1-p)f_1^\star + pf_2^\star\|^2 \\
&+ 2[pq + (1-p)(1-q)]\left(\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle\right)\langle f_1^\star, f_2^\star\rangle\|f_1^\star - f_2^\star\|^2 \\
&+ \left(\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle\right)^2\|f_1^\star - f_2^\star\|^2 \\
&+ 2p(1-q)\|f_1^\star - f_2^\star\|^2\|f_2^\star\|^2\|qf_1^\star + (1-q)f_2^\star\|^2 \\
&+ 2q(1-p)\left(\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\right)^2\|f_1^\star\|^2 \\
&+ 2p(1-q)\left(\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\right)^2\|f_2^\star\|^2 \\
&+ 2pq\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, f_1 - f_2^\star\rangle\langle f_1^\star, f_2^\star\rangle \\
&+ 2(1-p)(1-q)\langle qf_1^\star + (1-q)f_2^\star, f_1 - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_2^\star\rangle \\
&+ q\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_1^\star - f_2^\star\rangle\|(1-p)f_1^\star + pf_2^\star\|^2 \\
&+ 2(1-p)\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle f_1^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle \\
&+ 2(1-q)\langle (1-p)f_1^\star + pf_2^\star, f_1^\star - f_2^\star\rangle\langle f_2^\star, f_1^\star - f_2^\star\rangle\langle (1-p)f_1^\star + pf_2^\star, qf_1^\star + (1-q)f_2^\star\rangle \\
&+ 2p\langle qf_1^\star + (1-q)f_2^\star, f_1^\star - f_2^\star\rangle\langle f_2^\star, f_1^\star - f_2^\star\rangle\|qf_1^\star + (1-q)f_2^\star\|^2.
\end{aligned}
$$

We have:

$$H(\mathbf{Q}, G(\mathbf{f}^\star)) = D_{1,1}D_{2,2} - D_{1,2}^2.$$

We shall now write $H(\mathbf{Q}, G(\mathbf{f}^\star))$ using

$$n_1 = \|f_1^\star\|_2,\ n_2 = \|f_2^\star\|_2,\ a = \frac{\langle f_1^\star, f_2^\star\rangle}{\|f_1^\star\|_2\|f_2^\star\|_2},$$

for which the range is $[1,\infty[^2\times[0,1[$. Doing so, we obtain a polynomial P_1 in the variables n_1, n_2, a, p and q.
First observe that, by symmetry,

$$P_1(n_1, n_2, a, p, q) = P_1(n_2, n_1, a, q, p),$$

so that it is sufficient to prove that the polynomial P_1 is positive on the domain

$$1 \le n_2 \le n_1, \tag{18}$$

and $0 \le a < 1$ and $0 < p \ne q < 1$.
Furthermore, consider the change of variable

$$q = 1 - p + d$$

then we have a polynomial P_2 in the variables n_1, n_2, a, p and d which factorizes with

$$\frac{p^2(1-a^2)d^2n_1^2n_2^2(1+d-p)^2}{(1+d)^4}.$$

Dividing by this factor, one gets a polynomial P_3 which is homogeneous of degree 8 in n_1 and n_2, so that one may set $n_1 = 1$ and keep $b = n_2 \in]0,1]$ (observe that we have used (18) to reduce the problem to the domain $n_2/n_1 \le 1$) and obtain a polynomial P_4 in the variables b, a, p and d. It remains to prove that P_4 is positive on $\mathcal{D}_4 = \{b \in]0,1], a \in [0,1[, p \in]0,1[, d \in]p-1, 0[\cup]0, p[\}$.
Consider now the following change of variables

$$b = \frac{1}{1+x^2}, \quad a = \frac{y^2}{1+y^2}, \quad p = \frac{z^2}{1+z^2}, \quad \text{and} \quad d = \frac{(tz)^2 - 1}{(1+t^2)(1+z^2)},$$

mapping $(x,y,z,t) \in \mathbb{R}^4$ onto $(b,a,p,d) \in \mathcal{D}_5 = \{b \in]0,1], a \in [0,1[, p \in [0,1[, d \in]p-1, p[\}$ which contains $\mathcal{D}_4$. This change of variables maps P_4 onto a rational fraction with positive denominator, namely

$$(1+t^2)^4(1+y^2)^4(1+z^2)^4(1+x^2)^8$$

So it remains to prove that its numerator P_5, which is polynomial, is positive on $\mathbb{R}^4$. An expression of P_5 can be found in Appendix B. Observe that P_5 is polynomial in x^2, y^2, z^2 and t^2 and there are only three monomials with negative coefficients. These monomials can be expressed as sum of squares using others monomials, namely:

- $-18x^{12}t^2 + 27x^{12} + 1979x^{12}t^4 = 18x^{12} + 9(x^6 - x^6t^2)^2 + 1970x^{12}t^4$,
- $-108x^{10}t^2 + 1970x^{12}t^4 + 495x^8 = 439x^8 + 56(x^4 - x^6t^2)^2 + 1914x^{12}t^4 + 4t^2x^{10}$,
- and $-114x^8t^2 + 972x^4 + 1914x^{12}t^4 = 915x^4 + 57(x^2 - x^6t^2)^2 + 1857x^{12}t^4$.

Thus P_5 is equal to 144 more a sum of squares, hence it is positive. This proves that $H(\mathbf{Q}, G(\mathbf{f}^\star))$ is always positive.

8.5 Proof of Theorem 7

Let $\mathcal{K} = \{\mathbf{h} = \mathbf{f} - \mathbf{f}^\star, \mathbf{f} \in \mathcal{F}^K\}$. Using Theorem 4 we get that for all $x > 0$, for all $N \ge N_0$, with probability $1 - (e-1)^{-1}e^{-x}$, one has for any permutation τ_N,

$$
\begin{aligned}
\|\hat{g} - g^\star\|_2^2 \le\ & 6\inf_M\left\{\|g^\star - g^{\mathbf{Q}^\star, \mathbf{f}_M^\star}\|_2^2 + \mathrm{pen}(N,M)\right\} + A_1^\star\frac{x}{N} \\
& + 18C_{\mathcal{F},2}^6\left(\|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}_N\mathbb{P}_{\tau_N}^\top\|_F^2 + \|\pi^\star - \mathbb{P}_{\tau_N}\hat{\pi}\|_2^2\right).
\end{aligned} \tag{19}
$$

Notice that writing

$$\hat{g}\,(y_1,y_2,y_3) = \sum_{k_1,k_2,k_3=1}^{K} (\mathbb{P}_{\tau_N}\hat{\pi})(k_1)(\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top)(k_1,k_2)(\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top)(k_2,k_3) \times \hat{f}_{\tau_N(k_1)}(y_1)\hat{f}_{\tau_N(k_2)}(y_2)\hat{f}_{\tau_N(k_3)}(y_3)\,,$$

and applying Theorem 6 we get that, on the event $\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top \in \mathcal{V}$, there exists $\tau \in T_{\mathbf{Q}^\star}$ such that

$$\sum_{k=1}^{K} \|f^\star_{\tau(k)} - \hat{f}_{\tau_N(k)}\|_2^2 \leq \frac{1}{c(\mathcal{K},\mathcal{V},\mathfrak{F}^\star)^2}\|\hat{g} - g^{\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top,\mathbf{f}^\star}\|_2^2. \tag{20}$$

Now by the triangular inequality

$$\|\hat{g} - g^{\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top,\mathbf{f}^\star}\|_2 \leq \|\hat{g} - g^\star\|_2 + \|g^{\mathbf{Q}^\star,\mathbf{f}^\star} - g^{\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top,\mathbf{f}^\star}\|_2. \tag{21}$$

Similarly to (5), we have

$$\|g^{\mathbf{Q}^\star,\mathbf{f}^\star} - g^{\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top,\mathbf{f}^\star}\|_2^2 \leq 3K^3C_{\mathcal{F},2}^6\left[\|\pi^\star - \mathbb{P}_{\tau_N}\hat{\pi}\|_2^2 + 2\|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top\|_F^2\right]. \tag{22}$$

In the same way,

$$\left(g^\star - g^{\mathbf{Q}^\star,\mathbf{f}^\star_M}\right)(y_1,y_2,y_3) = \sum_{k_1,k_2,k_3=1}^{K} \pi^\star(k_1)\mathbf{Q}^\star(k_1,k_2)\mathbf{Q}^\star(k_2,k_3)\left(f^\star_{k_1}(y_1)f^\star_{k_2}(y_2)f^\star_{k_3}(y_3) - f^\star_{M,k_1}(y_1)f^\star_{M,k_2}(y_2)f^\star_{M,k_3}(y_3)\right)$$

so that

$$\|g^\star - g^{\mathbf{Q}^\star,\mathbf{f}^\star_M}\|_2^2 \leq 3K^3C_{\mathcal{F},2}^4 \max\{\|f^\star_k - f^\star_{M,k}\|_2^2,\ k=1,\dots,K\}.$$

Thus collecting (19), (20), (21), (22) and with an appropriate choice of $A^\star$ we get Theorem 7.

8.6 Proof of Corollary 10

We shall apply Theorem 11 where, for each N, we define δ_N such that $(-\log\delta_N)/\delta_N^2 := (\log N)^{1/2}$. Notice first that δ_N goes to 0 and that M_N tends to infinity as N tends to infinity, so that for large enough N, $M_N \geq M_{\mathfrak{F}^\star}$. By denoting τ_N the τ_{M_N} given by Theorem 11 we get that for all $x \geq x(\mathbf{Q}^\star)$, for all $N \geq \mathbf{N}(\mathbf{Q}^\star,\mathfrak{F}^\star)x\log N$, with probability $1-[4+(e-1)^{-1}]e^{-x} - 2\delta_N$,

$$\|\pi^\star - \mathbb{P}_{\tau_N}\hat{\pi}\|_2 \leq \mathcal{C}(\mathbf{Q}^\star,\mathfrak{F}^\star)\sqrt{\frac{\log N}{N}}\sqrt{x}$$

and

$$\|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top\| \leq \mathcal{C}(\mathbf{Q}^\star,\mathfrak{F}^\star)\sqrt{\frac{\log N}{N}}\sqrt{x}\,.$$

We first obtain that

$$\limsup_{N\to+\infty}\mathbb{E}\left[\frac{N}{\log N}\|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^T\|^2\right] \leq \mathcal{C}(\mathbf{Q}^\star,\mathfrak{F}^\star)^2\int_0^{+\infty}\limsup_{N\to+\infty}\mathbb{P}\left(\frac{\sqrt{N}}{\mathcal{C}(\mathbf{Q}^\star,\mathfrak{F}^\star)\sqrt{\log N}}\|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^T\| \geq \sqrt{x}\right)dx \leq \mathcal{C}(\mathbf{Q}^\star,\mathfrak{F}^\star)^2x(\mathbf{Q}^\star) + \mathcal{C}(\mathbf{Q}^\star,\mathfrak{F}^\star)^2\int_{x(\mathbf{Q}^\star)}^{+\infty}[4+(e-1)^{-1}]e^{-x}dx < +\infty$$

so that

$$\mathbb{E}\left[\|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^T\|^2\right] = O\left(\frac{\log N}{N}\right).$$

Similarly, one has $\mathbb{E}\left[\|\pi^\star - \mathbb{P}_{\tau_N}\hat{\pi}\|^2\right] = O\left(\frac{\log N}{N}\right)$. We also obtain, by taking $x = N/(\log N)^{1/4}$, that

$$\limsup_{N\to+\infty}\mathbb{P}\left(\mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^T \notin \mathcal{V}\right) = 0,$$

so that, using Theorem 7, we get for some $\tau \in T_{\mathbf{Q}^\star}$,

$$\limsup_{N\to+\infty}\mathbb{P}\left(\frac{N}{A^\star}\left[\sum_{k=1}^{K}\|f^\star_k - \hat{f}_{\tau^{-1}\circ\tau_N(k)}\|_2^2 - \inf_M\left\{\sum_{k=1}^{K}\|f^\star_k - f^\star_{M,k}\|_2^2 + \mathrm{pen}(N,M)\right\} - \|\mathbf{Q}^\star - \mathbb{P}_{\tau_N}\hat{\mathbf{Q}}\mathbb{P}_{\tau_N}^\top\|_F^2 - \|\pi^\star - \mathbb{P}_{\tau_N}\hat{\pi}\|_2^2 + \frac{x}{N}\right] \geq x\right) \leq (e-1)^{-1}e^{-x}.$$

Thus, by integration and the previous results, Corollary 10 follows.

Acknowledgments

We would like to thank N. Curien and D. Henrion for their help in the computation of P_5. We are deeply grateful to V. Magron for pointing us that P_5 can be explicitly expressed as a Sum-Of-Squares. Thanks are due to Luc Lehericy for a careful reading of the manuscript. We are grateful to anonymous referees for their careful reading of this work.

Appendix A. Concentration Inequalities

We first recall results that hold both for **(Scenario A)** (where we consider N i.i.d. samples $(Y_1^{(s)}, Y_2^{(s)}, Y_3^{(s)})_{s=1}^N$ of three consecutive observations) and for **(Scenario B)** (where we consider consecutive observations of the same chain).

The following proposition is the classical Bernstein's inequality for **(Scenario A)** and is proved in Paulin (2015), Theorem 2.4, for **(Scenario B)**.

Proposition 12 *Let t be a real valued and measurable bounded function on $\mathcal{Y}^3$. Let $V = \mathbb{E}[t^2(Z_1)]$. There exists a positive constant $c^\star$ depending only on $\mathbf{Q}^\star$ such that for all $0 \leq \lambda \leq 1/(2\sqrt{2}c^\star\|t\|_\infty)$:*

$$\log \mathbb{E}\exp\left[\lambda \sum_{s=1}^N (t(Z_s) - \mathbb{E}t(Z_s))\right] \leq \frac{2Nc^\star V\lambda^2}{1 - 2\sqrt{2}c^\star\|t\|_\infty\lambda} \tag{23}$$

so that for all $x \geq 0$,

$$\mathbb{P}\left(\sum_{s=1}^N (t(Z_s) - \mathbb{E}t(Z_s)) \geq 2\sqrt{2Nc^\star Vx} + 2\sqrt{2}c^\star\|t\|_\infty x\right) \leq e^{-x}. \tag{24}$$

We now state a deviation inequality, which comes from Massart (2007) Theorem 6.8 and Corollary 6.9 for **(Scenario A)**. For **(Scenario B)** the proof of the following proposition follows *mutatis mutandis* from the proof of Theorem 6.8 (and then Corollary 6.9) in Massart (2007) the early first step being equation (23). Recall that when t_1 and t_2 are real valued functions, the bracket $[t_1, t_2]$ is the set of real valued functions t such that $t_1(\cdot) \leq t(\cdot) \leq t_2(\cdot)$. For any measurable set A such that $\mathbb{P}(A) > 0$, and any integrable random variable Z, denote $E^A[Z] = E[Z\mathbb{1}_A]/\mathbb{P}(A)$.

Proposition 13 *Let $\mathcal{T}$ be some countable class of real valued and measurable functions on $\mathcal{Y}^3$. Assume that there exists some positive numbers σ and b such that for all $t \in \mathcal{T}$, $\|t\|_\infty \leq b$ and $\mathbb{E}[t^2(Z_1)] \leq \sigma^2$.*
Assume furthermore that for any positive number δ, there exists some finite set B_δ of brackets covering $\mathcal{F}$ such that for any bracket $[t_1, t_2] \in B_\delta$, $\|t_1 - t_2\|_\infty \leq b$ and $\mathbb{E}[(t_1 - t_2)^2(Z_1)] \leq \delta^2$. Let $e^{H(\delta)}$ denote the minimal cardinality of such a covering. Then, there exists a positive constant $C^\star$ depending only on $\mathbf{Q}^\star$ such that: for any measurable set A,

$$\mathbb{E}^A\left(\sup_{t\in\mathcal{T}} \sum_{s=1}^N (t(Z_s) - \mathbb{E}t(Z_s))\right) \leq C^\star\left[E + \sigma\sqrt{N\log\left(\frac{1}{\mathbb{P}(A)}\right)} + b\log\left(\frac{1}{\mathbb{P}(A)}\right)\right]$$

and for all positive number x

$$\mathbb{P}\left(\sup_{t\in\mathcal{T}} \sum_{s=1}^N (t(Z_s) - \mathbb{E}t(Z_s)) \geq C^\star[E + \sigma\sqrt{Nx} + bx]\right) \leq \exp(-x),$$

where

$$E = \sqrt{N}\int_0^\sigma \sqrt{H(u) \wedge N}du + (b + \sigma)H(\sigma).$$

Appendix B. Expression of Polynomial P_5

Computer assisted computations (available at `https://mycore.core-cloud.net/public.php?service=files&t=db7b8c1a2bcbcca157dcda5ecab84374`) give that:

$P_5 =$

```
144 - 114 t^2 x^8 - 108 t^2 x^10  - 18 t^2 x^12 +
192 t^2 + 128 t^4 + 256 t^6 + 176 t^8 + 576 x^2 + 624 t^2 x^2 +
672 t^4 x^2 + 1776 t^6 x^2 + 1152 t^8 x^2 + 972 x^4 + 720 t^2 x^4 +
1884 t^4 x^4 + 5496 t^6 x^4 + 3360 t^8 x^4 + 900 x^6 + 264 t^2 x^6 +
3556 t^4 x^6 + 9920 t^6 x^6 + 5728 t^8 x^6 + 495 x^8 +
4551 t^4 x^8 + 11424 t^6 x^8 + 6264 t^8 x^8 + 162 x^10 +
3810 t^4 x^10 + 8592 t^6 x^10 + 4512 t^8 x^10 +
27 x^12 + 1979 t^4 x^12 + 4120 t^6 x^12 +
2096 t^8 x^12 + 576 t^4 x^14 + 1152 t^6 x^14 + 576 t^8 x^14 +
72 t^4 x^16 + 144 t^6 x^16 + 72 t^8 x^16 + 144 y^2 + 480 t^2 y^2 +
784 t^4 y^2 + 704 t^6 y^2 + 256 t^8 y^2 + 576 x^2 y^2 +
2064 t^2 x^2 y^2 + 4192 t^4 x^2 y^2 + 4496 t^6 x^2 y^2 +
1792 t^8 x^2 y^2 + 1080 x^4 y^2 + 4104 t^2 x^4 y^2 +
10760 t^4 x^4 y^2 + 13528 t^6 x^4 y^2 + 5792 t^8 x^4 y^2 +
1224 x^6 y^2 + 5016 t^2 x^6 y^2 + 17592 t^4 x^6 y^2 +
25032 t^6 x^6 y^2 + 11232 t^8 x^6 y^2 + 900 x^8 y^2 +
4224 t^2 x^8 y^2 + 19924 t^4 x^8 y^2 + 30776 t^6 x^8 y^2 +
14176 t^8 x^8 y^2 + 432 x^10 y^2 + 2520 t^2 x^10 y^2 +
15584 t^4 x^10 y^2 + 25336 t^6 x^10 y^2 + 11840 t^8 x^10 y^2 +
108 x^12 y^2 + 936 t^2 x^12 y^2 + 7916 t^4 x^12 y^2 +
13456 t^6 x^12 y^2 + 6368 t^8 x^12 y^2 + 144 t^2 x^14 y^2 +
2304 t^4 x^14 y^2 + 4176 t^6 x^14 y^2 + 2016 t^8 x^14 y^2 +
288 t^4 x^16 y^2 + 576 t^6 x^16 y^2 + 288 t^8 x^16 y^2 + 144 y^4 +
480 t^2 y^4 + 624 t^4 y^4 + 384 t^6 y^4 + 96 t^8 y^4 + 576 x^2 y^4 +
2208 t^2 x^2 y^4 + 3392 t^4 x^2 y^4 + 2464 t^6 x^2 y^4 +
704 t^8 x^2 y^4 + 1188 x^4 y^4 + 5256 t^2 x^4 y^4 +
9636 t^4 x^4 y^4 + 8256 t^6 x^4 y^4 + 2688 t^8 x^4 y^4 +
1548 x^6 y^4 + 8112 t^2 x^6 y^4 + 18076 t^4 x^6 y^4 +
18008 t^6 x^6 y^4 + 6496 t^8 x^6 y^4 + 1359 x^8 y^4 +
8598 t^2 x^8 y^4 + 23375 t^4 x^8 y^4 + 26392 t^6 x^8 y^4 +
10256 t^8 x^8 y^4 + 810 x^10 y^4 + 6156 t^2 x^10 y^4 +
20442 t^4 x^10 y^4 + 25656 t^6 x^10 y^4 + 10560 t^8 x^10 y^4 +
243 x^12 y^4 + 2574 t^2 x^12 y^4 + 11299 t^4 x^12 y^4 +
15848 t^6 x^12 y^4 + 6880 t^8 x^12 y^4 + 432 t^2 x^14 y^4 +
3456 t^4 x^14 y^4 + 5616 t^6 x^14 y^4 + 2592 t^8 x^14 y^4 +
432 t^4 x^16 y^4 + 864 t^6 x^16 y^4 + 432 t^8 x^16 y^4 +
216 x^4 y^6 + 720 t^2 x^4 y^6 + 952 t^4 x^4 y^6 + 608 t^6 x^4 y^6 +
160 t^8 x^4 y^6 + 648 x^6 y^6 + 2592 t^2 x^6 y^6 +
4168 t^4 x^6 y^6 + 3152 t^6 x^6 y^6 + 928 t^8 x^6 y^6 +
918 x^8 y^6 + 4428 t^2 x^8 y^6 + 8502 t^4 x^8 y^6 +
7392 t^6 x^8 y^6 + 2400 t^8 x^8 y^6 + 756 x^10 y^6 +
4392 t^2 x^10 y^6 + 10036 t^4 x^10 y^6 + 9920 t^6 x^10 y^6 +
3520 t^8 x^10 y^6 + 270 x^12 y^6 + 2268 t^2 x^12 y^6 +
6766 t^4 x^12 y^6 + 7808 t^6 x^12 y^6 + 3040 t^8 x^12 y^6 +
432 t^2 x^14 y^6 + 2304 t^4 x^14 y^6 + 3312 t^6 x^14 y^6 +
```

```
1440 t^8 x^14 y^6 + 288 t^4 x^16 y^6 + 576 t^6 x^16 y^6 +
288 t^8 x^16 y^6 + 108 x^8 y^8 + 360 t^2 x^8 y^8 + 468 t^4 x^8 y^8 +
288 t^6 x^8 y^8 + 72 t^8 x^8 y^8 + 216 x^10 y^8 + 864 t^2 x^10 y^8 +
1368 t^4 x^10 y^8 + 1008 t^6 x^10 y^8 + 288 t^8 x^10 y^8 +
108 x^12 y^8 + 648 t^2 x^12 y^8 + 1404 t^4 x^12 y^8 +
1296 t^6 x^12 y^8 + 432 t^8 x^12 y^8 + 144 t^2 x^14 y^8 +
576 t^4 x^14 y^8 + 720 t^6 x^14 y^8 + 288 t^8 x^14 y^8 +
72 t^4 x^16 y^8 + 144 t^6 x^16 y^8 + 72 t^8 x^16 y^8 + 192 z^2 +
416 t^2 z^2 + 288 t^4 z^2 + 320 t^6 z^2 + 256 t^8 z^2 +
912 x^2 z^2 + 1664 t^2 x^2 z^2 + 1248 t^4 x^2 z^2 +
2304 t^6 x^2 z^2 + 1808 t^8 x^2 z^2 + 1728 x^4 z^2 +
2520 t^2 x^4 z^2 + 2776 t^4 x^4 z^2 + 7624 t^6 x^4 z^2 +
5640 t^8 x^4 z^2 + 1704 x^6 z^2 + 1736 t^2 x^6 z^2 +
4664 t^4 x^6 z^2 + 14808 t^6 x^6 z^2 + 10176 t^8 x^6 z^2 +
966 x^8 z^2 + 494 t^2 x^8 z^2 + 6098 t^4 x^8 z^2 +
18218 t^6 x^8 z^2 + 11648 t^8 x^8 z^2 + 324 x^10 z^2 +
36 t^2 x^10 z^2 + 5468 t^4 x^10 z^2 + 14444 t^6 x^10 z^2 +
8688 t^8 x^10 z^2 + 54 x^12 z^2 + 6 t^2 x^12 z^2 +
3002 t^4 x^12 z^2 + 7186 t^6 x^12 z^2 + 4136 t^8 x^12 z^2 +
896 t^4 x^14 z^2 + 2048 t^6 x^14 z^2 + 1152 t^8 x^14 z^2 +
112 t^4 x^16 z^2 + 256 t^6 x^16 z^2 + 144 t^8 x^16 z^2 +
480 y^2 z^2 + 1312 t^2 y^2 z^2 + 1888 t^4 y^2 z^2 +
1760 t^6 y^2 z^2 + 704 t^8 y^2 z^2 + 1776 x^2 y^2 z^2 +
5248 t^2 x^2 y^2 z^2 + 9504 t^4 x^2 y^2 z^2 +
10624 t^6 x^2 y^2 z^2 + 4592 t^8 x^2 y^2 z^2 + 3096 x^4 y^2 z^2 +
9904 t^2 x^4 y^2 z^2 + 23104 t^4 x^4 y^2 z^2 +
30288 t^6 x^4 y^2 z^2 + 13992 t^8 x^4 y^2 z^2 + 3144 x^6 y^2 z^2 +
11344 t^2 x^6 y^2 z^2 + 35712 t^4 x^6 y^2 z^2 +
53424 t^6 x^6 y^2 z^2 + 25912 t^8 x^6 y^2 z^2 + 2064 x^8 y^2 z^2 +
9016 t^2 x^8 y^2 z^2 + 38552 t^4 x^8 y^2 z^2 +
63192 t^6 x^8 y^2 z^2 + 31592 t^8 x^8 y^2 z^2 + 936 x^10 y^2 z^2 +
5248 t^2 x^10 y^2 z^2 + 29072 t^4 x^10 y^2 z^2 +
50464 t^6 x^10 y^2 z^2 + 25704 t^8 x^10 y^2 z^2 + 216 x^12 y^2 z^2 +
1872 t^2 x^12 y^2 z^2 + 14192 t^4 x^12 y^2 z^2 +
26056 t^6 x^12 y^2 z^2 + 13520 t^8 x^12 y^2 z^2 +
264 t^2 x^14 y^2 z^2 + 3896 t^4 x^14 y^2 z^2 +
7808 t^6 x^14 y^2 z^2 + 4176 t^8 x^14 y^2 z^2 +
448 t^4 x^16 y^2 z^2 + 1024 t^6 x^16 y^2 z^2 +
576 t^8 x^16 y^2 z^2 + 480 y^4 z^2 + 1632 t^2 y^4 z^2 +
2208 t^4 y^4 z^2 + 1440 t^6 y^4 z^2 + 384 t^8 y^4 z^2 +
1632 x^2 y^4 z^2 + 6528 t^2 x^2 y^4 z^2 + 10688 t^4 x^2 y^4 z^2 +
8320 t^6 x^2 y^4 z^2 + 2528 t^8 x^2 y^4 z^2 + 3240 x^4 y^4 z^2 +
14280 t^2 x^4 y^4 z^2 + 27448 t^4 x^4 y^4 z^2 +
25048 t^6 x^4 y^4 z^2 + 8640 t^8 x^4 y^4 z^2 + 3936 x^6 y^4 z^2 +
19992 t^2 x^6 y^4 z^2 + 46552 t^4 x^6 y^4 z^2 +
49352 t^6 x^6 y^4 z^2 + 18856 t^8 x^6 y^4 z^2 + 3198 x^8 y^4 z^2 +
19518 t^2 x^8 y^4 z^2 + 55218 t^4 x^8 y^4 z^2 +
66170 t^6 x^8 y^4 z^2 + 27272 t^8 x^8 y^4 z^2 + 1836 x^10 y^4 z^2 +
13332 t^2 x^10 y^4 z^2 + 44988 t^4 x^10 y^4 z^2 +
59580 t^6 x^10 y^4 z^2 + 26088 t^8 x^10 y^4 z^2 + 486 x^12 y^4 z^2 +
5214 t^2 x^12 y^4 z^2 + 22994 t^4 x^12 y^4 z^2 +
```



```
34194 t^6 x^12 y^4 z^2 + 15928 t^8 x^12 y^4 z^2 +
792 t^2 x^14 y^4 z^2 + 6312 t^4 x^14 y^4 z^2 +
11136 t^6 x^14 y^4 z^2 + 5616 t^8 x^14 y^4 z^2 +
672 t^4 x^16 y^4 z^2 + 1536 t^6 x^16 y^4 z^2 +
864 t^8 x^16 y^4 z^2 + 720 x^4 y^6 z^2 + 2480 t^2 x^4 y^6 z^2 +
3472 t^4 x^4 y^6 z^2 + 2384 t^6 x^4 y^6 z^2 + 672 t^8 x^4 y^6 z^2 +
1728 x^6 y^6 z^2 + 7440 t^2 x^6 y^6 z^2 + 13072 t^4 x^6 y^6 z^2 +
10736 t^6 x^6 y^6 z^2 + 3376 t^8 x^6 y^6 z^2 + 2268 x^8 y^6 z^2 +
11484 t^2 x^8 y^6 z^2 + 23812 t^4 x^8 y^6 z^2 +
22276 t^6 x^8 y^6 z^2 + 7680 t^8 x^8 y^6 z^2 + 1800 x^10 y^6 z^2 +
10568 t^2 x^10 y^6 z^2 + 25560 t^4 x^10 y^6 z^2 +
26872 t^6 x^10 y^6 z^2 + 10080 t^8 x^10 y^6 z^2 + 540 x^12 y^6 z^2 +
4836 t^2 x^12 y^6 z^2 + 15420 t^4 x^12 y^6 z^2 +
18964 t^6 x^12 y^6 z^2 + 7840 t^8 x^12 y^6 z^2 +
792 t^2 x^14 y^6 z^2 + 4520 t^4 x^14 y^6 z^2 +
7040 t^6 x^14 y^6 z^2 + 3312 t^8 x^14 y^6 z^2 +
448 t^4 x^16 y^6 z^2 + 1024 t^6 x^16 y^6 z^2 +
576 t^8 x^16 y^6 z^2 + 360 x^8 y^8 z^2 + 1224 t^2 x^8 y^8 z^2 +
1656 t^4 x^8 y^8 z^2 + 1080 t^6 x^8 y^8 z^2 + 288 t^8 x^8 y^8 z^2 +
576 x^10 y^8 z^2 + 2448 t^2 x^10 y^8 z^2 + 4176 t^4 x^10 y^8 z^2 +
3312 t^6 x^10 y^8 z^2 + 1008 t^8 x^10 y^8 z^2 + 216 x^12 y^8 z^2 +
1488 t^2 x^12 y^8 z^2 + 3616 t^4 x^12 y^8 z^2 +
3640 t^6 x^12 y^8 z^2 + 1296 t^8 x^12 y^8 z^2 +
264 t^2 x^14 y^8 z^2 + 1208 t^4 x^14 y^8 z^2 +
1664 t^6 x^14 y^8 z^2 + 720 t^8 x^14 y^8 z^2 +
112 t^4 x^16 y^8 z^2 + 256 t^6 x^16 y^8 z^2 + 144 t^8 x^16 y^8 z^2 +
128 z^4 + 288 t^2 z^4 + 352 t^4 z^4 + 384 t^6 z^4 + 256 t^8 z^4 +
352 x^2 z^4 + 1056 t^2 x^2 z^4 + 1408 t^4 x^2 z^4 +
1952 t^6 x^2 z^4 + 1504 t^8 x^2 z^4 + 764 x^4 z^4 +
2104 t^2 x^4 z^4 + 2616 t^4 x^4 z^4 + 5016 t^6 x^4 z^4 +
4252 t^8 x^4 z^4 + 804 x^6 z^4 + 1912 t^2 x^6 z^4 +
2920 t^4 x^6 z^4 + 8536 t^6 x^6 z^4 + 7364 t^8 x^6 z^4 +
471 x^8 z^4 + 898 t^2 x^8 z^4 + 2694 t^4 x^8 z^4 +
10058 t^6 x^8 z^4 + 8335 t^8 x^8 z^4 + 162 x^10 z^4 +
252 t^2 x^10 z^4 + 2164 t^4 x^10 z^4 + 7980 t^6 x^10 z^4 +
6226 t^8 x^10 z^4 + 27 x^12 z^4 + 42 t^2 x^12 z^4 +
1182 t^4 x^12 z^4 + 4018 t^6 x^12 z^4 + 2979 t^8 x^12 z^4 +
352 t^4 x^14 z^4 + 1152 t^6 x^14 z^4 + 832 t^8 x^14 z^4 +
44 t^4 x^16 z^4 + 144 t^6 x^16 z^4 + 104 t^8 x^16 z^4 +
784 y^2 z^4 + 1888 t^2 y^2 z^4 + 2208 t^4 y^2 z^4 +
1888 t^6 y^2 z^4 + 784 t^8 y^2 z^4 + 2080 x^2 y^2 z^4 +
5600 t^2 x^2 y^2 z^4 + 8832 t^4 x^2 y^2 z^4 + 9952 t^6 x^2 y^2 z^4 +
4640 t^8 x^2 y^2 z^4 + 3368 x^4 y^2 z^4 + 9440 t^2 x^4 y^2 z^4 +
18928 t^4 x^4 y^2 z^4 + 25952 t^6 x^4 y^2 z^4 +
13224 t^8 x^4 y^2 z^4 + 2840 x^6 y^2 z^4 + 9056 t^2 x^6 y^2 z^4 +
25872 t^4 x^6 y^2 z^4 + 42464 t^6 x^6 y^2 z^4 +
23192 t^8 x^6 y^2 z^4 + 1524 x^8 y^2 z^4 + 6072 t^2 x^8 y^2 z^4 +
25016 t^4 x^8 y^2 z^4 + 46792 t^6 x^8 y^2 z^4 +
26900 t^8 x^8 y^2 z^4 + 576 x^10 y^2 z^4 + 3184 t^2 x^10 y^2 z^4 +
17216 t^4 x^10 y^2 z^4 + 35024 t^6 x^10 y^2 z^4 +
20928 t^8 x^10 y^2 z^4 + 108 x^12 y^2 z^4 + 1008 t^2 x^12 y^2 z^4 +
```

```
7584 t^4 x^12 y^2 z^4 + 16968 t^6 x^12 y^2 z^4 +
10572 t^8 x^12 y^2 z^4 + 120 t^2 x^14 y^2 z^4 +
1816 t^4 x^14 y^2 z^4 + 4736 t^6 x^14 y^2 z^4 +
3136 t^8 x^14 y^2 z^4 + 176 t^4 x^16 y^2 z^4 +
576 t^6 x^16 y^2 z^4 + 416 t^8 x^16 y^2 z^4 + 624 y^4 z^4 +
2208 t^2 y^4 z^4 + 3168 t^4 y^4 z^4 + 2208 t^6 y^4 z^4 +
624 t^8 y^4 z^4 + 1600 x^2 y^4 z^4 + 6976 t^2 x^2 y^4 z^4 +
12672 t^4 x^2 y^4 z^4 + 10816 t^6 x^2 y^4 z^4 +
3520 t^8 x^2 y^4 z^4 + 3364 x^4 y^4 z^4 + 14456 t^2 x^4 y^4 z^4 +
29416 t^4 x^4 y^4 z^4 + 29016 t^6 x^4 y^4 z^4 +
10692 t^8 x^4 y^4 z^4 + 3452 x^6 y^4 z^4 + 17336 t^2 x^6 y^4 z^4 +
43896 t^4 x^6 y^4 z^4 + 51032 t^6 x^6 y^4 z^4 +
21020 t^8 x^6 y^4 z^4 + 2495 x^8 y^4 z^4 + 14658 t^2 x^8 y^4 z^4 +
45814 t^4 x^8 y^4 z^4 + 61162 t^6 x^8 y^4 z^4 +
27607 t^8 x^8 y^4 z^4 + 1242 x^10 y^4 z^4 + 8892 t^2 x^10 y^4 z^4 +
33252 t^4 x^10 y^4 z^4 + 49644 t^6 x^10 y^4 z^4 +
24234 t^8 x^10 y^4 z^4 + 243 x^12 y^4 z^4 + 2914 t^2 x^12 y^4 z^4 +
14758 t^4 x^12 y^4 z^4 + 25538 t^6 x^12 y^4 z^4 +
13643 t^8 x^12 y^4 z^4 + 360 t^2 x^14 y^4 z^4 +
3336 t^4 x^14 y^4 z^4 + 7296 t^6 x^14 y^4 z^4 +
4416 t^8 x^14 y^4 z^4 + 264 t^4 x^16 y^4 z^4 +
864 t^6 x^16 y^4 z^4 + 624 t^8 x^16 y^4 z^4 + 952 x^4 y^6 z^4 +
3472 t^2 x^4 y^6 z^4 + 5232 t^4 x^4 y^6 z^4 + 3856 t^6 x^4 y^6 z^4 +
1144 t^8 x^4 y^6 z^4 + 1544 x^6 y^6 z^4 + 7760 t^2 x^6 y^6 z^4 +
15696 t^4 x^6 y^6 z^4 + 14288 t^6 x^6 y^6 z^4 +
4808 t^8 x^6 y^6 z^4 + 1942 x^8 y^6 z^4 + 10532 t^2 x^8 y^6 z^4 +
24556 t^4 x^8 y^6 z^4 + 25380 t^6 x^8 y^6 z^4 +
9414 t^8 x^8 y^6 z^4 + 1332 x^10 y^6 z^4 + 8408 t^2 x^10 y^6 z^4 +
22952 t^4 x^10 y^6 z^4 + 26776 t^6 x^10 y^6 z^4 +
10900 t^8 x^10 y^6 z^4 + 270 x^12 y^6 z^4 + 2972 t^2 x^12 y^6 z^4 +
11492 t^4 x^12 y^6 z^4 + 16244 t^6 x^12 y^6 z^4 +
7486 t^8 x^12 y^6 z^4 + 360 t^2 x^14 y^6 z^4 +
2632 t^4 x^14 y^6 z^4 + 4992 t^6 x^14 y^6 z^4 +
2752 t^8 x^14 y^6 z^4 + 176 t^4 x^16 y^6 z^4 +
576 t^6 x^16 y^6 z^4 + 416 t^8 x^16 y^6 z^4 + 468 x^8 y^8 z^4 +
1656 t^2 x^8 y^8 z^4 + 2376 t^4 x^8 y^8 z^4 + 1656 t^6 x^8 y^8 z^4 +
468 t^8 x^8 y^8 z^4 + 504 x^10 y^8 z^4 + 2448 t^2 x^10 y^8 z^4 +
4752 t^4 x^10 y^8 z^4 + 4176 t^6 x^10 y^8 z^4 +
1368 t^8 x^10 y^8 z^4 + 108 x^12 y^8 z^4 + 1024 t^2 x^12 y^8 z^4 +
3136 t^4 x^12 y^8 z^4 + 3656 t^6 x^12 y^8 z^4 +
1436 t^8 x^12 y^8 z^4 + 120 t^2 x^14 y^8 z^4 +
760 t^4 x^14 y^8 z^4 + 1280 t^6 x^14 y^8 z^4 +
640 t^8 x^14 y^8 z^4 + 44 t^4 x^16 y^8 z^4 + 144 t^6 x^16 y^8 z^4 +
104 t^8 x^16 y^8 z^4 + 256 z^6 + 320 t^2 z^6 + 384 t^4 z^6 +
352 t^6 z^6 + 160 t^8 z^6 + 272 x^2 z^6 + 256 t^2 x^2 z^6 +
1120 t^4 x^2 z^6 + 1408 t^6 x^2 z^6 + 784 t^8 x^2 z^6 +
232 x^4 z^6 + 456 t^2 x^4 z^6 + 2104 t^4 x^4 z^6 +
2712 t^6 x^4 z^6 + 1856 t^8 x^4 z^6 + 96 x^6 z^6 + 472 t^2 x^6 z^6 +
2072 t^4 x^6 z^6 + 3208 t^6 x^6 z^6 + 2792 t^8 x^6 z^6 +
24 x^8 z^6 + 298 t^2 x^8 z^6 + 1178 t^4 x^8 z^6 + 2686 t^6 x^8 z^6 +
2870 t^8 x^8 z^6 + 108 t^2 x^10 z^6 + 396 t^4 x^10 z^6 +
```



```
1668 t^6 x^10 z^6 + 2020 t^8 x^10 z^6 + 18 t^2 x^12 z^6 +
66 t^4 x^12 z^6 + 726 t^6 x^12 z^6 + 934 t^8 x^12 z^6 +
192 t^6 x^14 z^6 + 256 t^8 x^14 z^6 + 24 t^6 x^16 z^6 +
32 t^8 x^16 z^6 + 704 y^2 z^6 + 1760 t^2 y^2 z^6 +
1888 t^4 y^2 z^6 + 1312 t^6 y^2 z^6 + 480 t^8 y^2 z^6 +
1136 x^2 y^2 z^6 + 3456 t^2 x^2 y^2 z^6 + 5152 t^4 x^2 y^2 z^6 +
5248 t^6 x^2 y^2 z^6 + 2416 t^8 x^2 y^2 z^6 + 1768 x^4 y^2 z^6 +
5200 t^2 x^4 y^2 z^6 + 9152 t^4 x^4 y^2 z^6 +
11696 t^6 x^4 y^2 z^6 + 6232 t^8 x^4 y^2 z^6 + 1144 x^6 y^2 z^6 +
3760 t^2 x^6 y^2 z^6 + 9984 t^4 x^6 y^2 z^6 +
16720 t^6 x^6 y^2 z^6 + 10120 t^8 x^6 y^2 z^6 + 456 x^8 y^2 z^6 +
1752 t^2 x^8 y^2 z^6 + 7592 t^4 x^8 y^2 z^6 +
16024 t^6 x^8 y^2 z^6 + 10880 t^8 x^8 y^2 z^6 + 72 x^10 y^2 z^6 +
544 t^2 x^10 y^2 z^6 + 3952 t^4 x^10 y^2 z^6 +
10304 t^6 x^10 y^2 z^6 + 7848 t^8 x^10 y^2 z^6 +
72 t^2 x^12 y^2 z^6 + 1160 t^4 x^12 y^2 z^6 +
4192 t^6 x^12 y^2 z^6 + 3680 t^8 x^12 y^2 z^6 +
128 t^4 x^14 y^2 z^6 + 952 t^6 x^14 y^2 z^6 +
1016 t^8 x^14 y^2 z^6 + 96 t^6 x^16 y^2 z^6 + 128 t^8 x^16 y^2 z^6 +
384 y^4 z^6 + 1440 t^2 y^4 z^6 + 2208 t^4 y^4 z^6 +
1632 t^6 y^4 z^6 + 480 t^8 y^4 z^6 + 608 x^2 y^4 z^6 +
3200 t^2 x^2 y^4 z^6 + 6848 t^4 x^2 y^4 z^6 + 6528 t^6 x^2 y^4 z^6 +
2272 t^8 x^2 y^4 z^6 + 1760 x^4 y^4 z^6 + 7128 t^2 x^4 y^4 z^6 +
15128 t^4 x^4 y^4 z^6 + 16008 t^6 x^4 y^4 z^6 +
6248 t^8 x^4 y^4 z^6 + 1288 x^6 y^4 z^6 + 6856 t^2 x^6 y^4 z^6 +
19576 t^4 x^6 y^4 z^6 + 25176 t^6 x^6 y^4 z^6 +
11168 t^8 x^6 y^4 z^6 + 832 x^8 y^4 z^6 + 4730 t^2 x^8 y^4 z^6 +
17242 t^4 x^8 y^4 z^6 + 26382 t^6 x^8 y^4 z^6 +
13230 t^8 x^8 y^4 z^6 + 216 x^10 y^4 z^6 + 1980 t^2 x^10 y^4 z^6 +
10092 t^4 x^10 y^4 z^6 + 18420 t^6 x^10 y^4 z^6 +
10476 t^8 x^10 y^4 z^6 + 274 t^2 x^12 y^4 z^6 +
3186 t^4 x^12 y^4 z^6 + 7806 t^6 x^12 y^4 z^6 +
5278 t^8 x^12 y^4 z^6 + 384 t^4 x^14 y^4 z^6 +
1704 t^6 x^14 y^4 z^6 + 1512 t^8 x^14 y^4 z^6 +
144 t^6 x^16 y^4 z^6 + 192 t^8 x^16 y^4 z^6 + 608 x^4 y^6 z^6 +
2384 t^2 x^4 y^6 z^6 + 3856 t^4 x^4 y^6 z^6 + 2992 t^6 x^4 y^6 z^6 +
912 t^8 x^4 y^6 z^6 + 496 x^6 y^6 z^6 + 3568 t^2 x^6 y^6 z^6 +
8848 t^4 x^6 y^6 z^6 + 8976 t^6 x^6 y^6 z^6 + 3200 t^8 x^6 y^6 z^6 +
752 x^8 y^6 z^6 + 4356 t^2 x^8 y^6 z^6 + 11780 t^4 x^8 y^6 z^6 +
13596 t^6 x^8 y^6 z^6 + 5420 t^8 x^8 y^6 z^6 + 288 x^10 y^6 z^6 +
2552 t^2 x^10 y^6 z^6 + 8984 t^4 x^10 y^6 z^6 +
12232 t^6 x^10 y^6 z^6 + 5512 t^8 x^10 y^6 z^6 +
404 t^2 x^12 y^6 z^6 + 3156 t^4 x^12 y^6 z^6 +
5940 t^6 x^12 y^6 z^6 + 3252 t^8 x^12 y^6 z^6 +
384 t^4 x^14 y^6 z^6 + 1320 t^6 x^14 y^6 z^6 +
1000 t^8 x^14 y^6 z^6 + 96 t^6 x^16 y^6 z^6 + 128 t^8 x^16 y^6 z^6 +
288 x^8 y^8 z^6 + 1080 t^2 x^8 y^8 z^6 + 1656 t^4 x^8 y^8 z^6 +
1224 t^6 x^8 y^8 z^6 + 360 t^8 x^8 y^8 z^6 + 144 x^10 y^8 z^6 +
1008 t^2 x^10 y^8 z^6 + 2448 t^4 x^10 y^8 z^6 +
2448 t^6 x^10 y^8 z^6 + 864 t^8 x^10 y^8 z^6 +
184 t^2 x^12 y^8 z^6 + 1064 t^4 x^12 y^8 z^6 +
```

```
1600 t^6 x^12 y^8 z^6 + 720 t^8 x^12 y^8 z^6 +
128 t^4 x^14 y^8 z^6 + 376 t^6 x^14 y^8 z^6 + 248 t^8 x^14 y^8 z^6 +
24 t^6 x^16 y^8 z^6 + 32 t^8 x^16 y^8 z^6 + 176 z^8 + 256 t^2 z^8 +
256 t^4 z^8 + 160 t^6 z^8 + 48 t^8 z^8 + 256 x^2 z^8 +
240 t^2 x^2 z^8 + 544 t^4 x^2 z^8 + 496 t^6 x^2 z^8 +
192 t^8 x^2 z^8 + 224 x^4 z^8 + 152 t^2 x^4 z^8 + 892 t^4 x^4 z^8 +
848 t^6 x^4 z^8 + 396 t^8 x^4 z^8 + 96 x^6 z^8 + 32 t^2 x^6 z^8 +
900 t^4 x^6 z^8 + 840 t^6 x^6 z^8 + 516 t^8 x^6 z^8 + 24 x^8 z^8 +
8 t^2 x^8 z^8 + 575 t^4 x^8 z^8 + 510 t^6 x^8 z^8 +
463 t^8 x^8 z^8 + 210 t^4 x^10 z^8 + 180 t^6 x^10 z^8 +
290 t^8 x^10 z^8 + 35 t^4 x^12 z^8 + 30 t^6 x^12 z^8 +
123 t^8 x^12 z^8 + 32 t^8 x^14 z^8 + 4 t^8 x^16 z^8 + 256 y^2 z^8 +
704 t^2 y^2 z^8 + 784 t^4 y^2 z^8 + 480 t^6 y^2 z^8 +
144 t^8 y^2 z^8 + 256 x^2 y^2 z^8 + 1040 t^2 x^2 y^2 z^8 +
1632 t^4 x^2 y^2 z^8 + 1424 t^6 x^2 y^2 z^8 + 576 t^8 x^2 y^2 z^8 +
416 x^4 y^2 z^8 + 1560 t^2 x^4 y^2 z^8 + 2696 t^4 x^4 y^2 z^8 +
2760 t^6 x^4 y^2 z^8 + 1336 t^8 x^4 y^2 z^8 + 224 x^6 y^2 z^8 +
1032 t^2 x^6 y^2 z^8 + 2616 t^4 x^6 y^2 z^8 + 3416 t^6 x^6 y^2 z^8 +
1992 t^8 x^6 y^2 z^8 + 96 x^8 y^2 z^8 + 472 t^2 x^8 y^2 z^8 +
1780 t^4 x^8 y^2 z^8 + 2800 t^6 x^8 y^2 z^8 + 1972 t^8 x^8 y^2 z^8 +
88 t^2 x^10 y^2 z^8 + 736 t^4 x^10 y^2 z^8 + 1432 t^6 x^10 y^2 z^8 +
1296 t^8 x^10 y^2 z^8 + 140 t^4 x^12 y^2 z^8 +
400 t^6 x^12 y^2 z^8 + 548 t^8 x^12 y^2 z^8 + 40 t^6 x^14 y^2 z^8 +
136 t^8 x^14 y^2 z^8 + 16 t^8 x^16 y^2 z^8 + 96 y^4 z^8 +
384 t^2 y^4 z^8 + 624 t^4 y^4 z^8 + 480 t^6 y^4 z^8 +
144 t^8 y^4 z^8 + 64 x^2 y^4 z^8 + 544 t^2 x^2 y^4 z^8 +
1472 t^4 x^2 y^4 z^8 + 1568 t^6 x^2 y^4 z^8 + 576 t^8 x^2 y^4 z^8 +
448 x^4 y^4 z^8 + 16t96 t^2 x^4 y^4 z^8 + 3524 t^4 x^4 y^4 z^8 +
3784 t^6 x^4 y^4 z^8 + 1508 t^8 x^4 y^4 z^8 + 224 x^6 y^4 z^8 +
1400 t^2 x^6 y^4 z^8 + 4156 t^4 x^6 y^4 z^8 + 5488 t^6 x^6 y^4 z^8 +
2508 t^8 x^6 y^4 z^8 + 176 x^8 y^4 z^8 + 992 t^2 x^8 y^4 z^8 +
3367 t^4 x^8 y^4 z^8 + 5190 t^6 x^8 y^4 z^8 + 2735 t^8 x^8 y^4 z^8 +
264 t^2 x^10 y^4 z^8 + 1578 t^4 x^10 y^4 z^8 +
3084 t^6 x^10 y^4 z^8 + 1962 t^8 x^10 y^4 z^8 +
315 t^4 x^12 y^4 z^8 + 998 t^6 x^12 y^4 z^8 + 875 t^8 x^12 y^4 z^8 +
120 t^6 x^14 y^4 z^8 + 216 t^8 x^14 y^4 z^8 + 24 t^8 x^16 y^4 z^8 +
160 x^4 y^6 z^8 + 672 t^2 x^4 y^6 z^8 + 1144 t^4 x^4 y^6 z^8 +
912 t^6 x^4 y^6 z^8 + 280 t^8 x^4 y^6 z^8 + 32 x^6 y^6 z^8 +
656 t^2 x^6 y^6 z^8 + 2056 t^4 x^6 y^6 z^8 + 2272 t^6 x^6 y^6 z^8 +
840 t^8 x^6 y^6 z^8 + 160 x^8 y^6 z^8 + 880 t^2 x^8 y^6 z^8 +
2534 t^4 x^8 y^6 z^8 + 3100 t^6 x^8 y^6 z^8 + 1286 t^8 x^8 y^6 z^8 +
320 t^2 x^10 y^6 z^8 + 1556 t^4 x^10 y^6 z^8 +
2408 t^6 x^10 y^6 z^8 + 1172 t^8 x^10 y^6 z^8 +
350 t^4 x^12 y^6 z^8 + 916 t^6 x^12 y^6 z^8 + 598 t^8 x^12 y^6 z^8 +
120 t^6 x^14 y^6 z^8 + 152 t^8 x^14 y^6 z^8 + 16 t^8 x^16 y^6 z^8 +
72 x^8 y^8 z^8 + 288 t^2 x^8 y^8 z^8 + 468 t^4 x^8 y^8 z^8 +
360 t^6 x^8 y^8 z^8 + 108 t^8 x^8 y^8 z^8 + 144 t^2 x^10 y^8 z^8 +
504 t^4 x^10 y^8 z^8 + 576 t^6 x^10 y^8 z^8 + 216 t^8 x^10 y^8 z^8 +
140 t^4 x^12 y^8 z^8 + 288 t^6 x^12 y^8 z^8 + 148 t^8 x^12 y^8 z^8 +
40 t^6 x^14 y^8 z^8 + 40 t^8 x^14 y^8 z^8 + 4 t^8 x^16 y^8 z^8
```


References

Grigory Alexandrovich, Hajo Holzmann, and Anna Leister. Nonparametric identification and maximum likelihood estimation for hidden markov models. *Biometrika*, 103(2):423–434, 2016.

Elizabeth S Allman, Catherine Matias, and John A Rhodes. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, pages 3099–3132, 2009.

Animashree Anandkumar, Daniel Hsu, and Sham M Kakade. A method of moments for mixture models and hidden markov models. In *25th Annual Conference on Learning Theory*, volume 23 of *JMLR: Workshop and Conference Proceedings*, pages 33.1–33.34, 2012.

Jean-Patrick Baudry, Cathy Maugis, and Bertrand Michel. Slope heuristics: overview and implementation. *Statistics and Computing*, 22(2):455–470, 2012.

Stéphane Bonhomme, Koen Jochmans, and Jean-Marc Robin. Estimating multivariate latent-structure models. *The Annals of Statistics*, 44(2):540–563, 2016.

Dominique Bontemps and Wilson Toussile. Clustering and variable selection for categorical multivariate data. *Electronic Journal of Statistics*, 7:2344–2371, 2013.

Laurent Couvreur and Christophe Couvreur. Wavelet based non-parametric HMMs: theory and methods. In *ICASSP '00 Proceedings*, pages 604–607, 2000.

Yohann De Castro, Elisabeth Gassiat, and Sylvain Le Corff. Consistent estimation of the filtering and marginal smoothing distributions in nonparametric hidden markov models. *arXiv preprint arXiv:1507.06510*, 2015.

Ronald A DeVore and George G Lorentz. *Constructive approximation*, volume 303. Springer Science & Business Media, 1993.

Thierry Dumont and Sylvain Le Corff. Nonparametric regression on hidden phi-mixing variables: identifiability and consistency of a pseudo-likelihood based estimation procedure. *Bernoulli*, to appear, 2017.

Élisabeth Gassiat and Judith Rousseau. Nonparametric finite translation hidden markov models and extensions. *Bernoulli*, 22(1):193–212, 2016.

Élisabeth Gassiat, Alice Cleynen, and Stéphane Robin. Inference in finite state space non parametric hidden markov models and applications. *Statistics and Computing*, 26(1-2): 61–71, 2016.

Nikolaus Hansen. The cma evolution strategy: a comparing review. In *Towards a new evolutionary computation*, pages 75–102. Springer, 2006.

Daniel Hsu, Sham M Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

Martin F Lambert, Julian P Whiting, and Andrew V Metcalfe. A non-parametric hidden markov model for climate state identification. *Hydrology and Earth System Sciences Discussions*, 7(5):652–667, 2003.

Fabrice Lefèvre. Non-parametric probability estimation for hmm-based automatic speech recognition. *Computer Speech & Language*, 17(2):113–136, 2003.

Luc Lehéricy. Estimation adaptative non paramétrique pour les modèles à chaîne de Markov cachée. Mémoire de M2, Orsay, 2015.

Luc Lehéricy. Order estimation for non-parametric hidden markov models. *arXiv preprint arXiv:1606.00622*, 2016.

Pascal Massart. *Concentration inequalities and model selection*, volume 1896 of *Lecture Notes in Mathematics*. Springer, Berlin, 2007. ISBN 978-3-540-48497-4; 3-540-48497-3. Lectures from the 33rd Summer School on Probability Theory held in Saint-Flour, July 6–23, 2003, With a foreword by Jean Picard.

Yves Meyer. *Wavelets and operators*, volume 37 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 1992. ISBN 0-521-42000-8; 0-521-45869-2. Translated from the 1990 French original by D. H. Salinger.

Daniel Paulin. Concentration inequalities for markov chains by marton couplings and spectral methods. *Electronic Journal of Probability*, 20, 2015.

Issai Schur. Bemerkungen zur Theorie der beschränkten Bilinearformen mit unendlich vielen Veränderlichen. *J. Reine Angew. Math.*, 140:1–28, 1911. ISSN 0075-4102. doi: 10.1515/crll.1911.140.1. URL `http://dx.doi.org/10.1515/crll.1911.140.1`.

Lifeng Shang and Kwok-Ping Chan. Nonparametric discriminant hmm and application to facial expression recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2090–2096. IEEE, 2009.

Le Song, Animashree Anandkumar, Bo Dai, and Bo Xie. Nonparametric estimation of multi-view latent variable models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *JMLR: Workshop and Conference Proceedings*, 2014.

Élodie Vernet. Posterior consistency for nonparametric hidden markov models with finite state space. *Electronic Journal of Statistics*, 9:717–752, 2015.

Stevenn Volant, Caroline Bérard, Marie-Laure Martin-Magniette, and Stéphane Robin. Hidden markov models with mixtures as emission distributions. *Statistics and Computing*, 24(4):493–504, 2014.

Christopher Yau, Omiros Papaspiliopoulos, Gareth O Roberts, and Christopher Holmes. Bayesian non-parametric hidden markov models with applications in genomics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(1):37–57, 2011.

Journal of Machine Learning Research 17 (2016) 1-30 Submitted 8/15; Revised 6/16; Published 7/16

Decrypting "Cryptogenic" Epilepsy: Semi-supervised Hierarchical Conditional Random Fields For Detecting Cortical Lesions In MRI-Negative Patients

Bilal Ahmed bahmed01@cs.tufts.edu
Department of Computer Science
Tufts University
Medford, MA 02155, USA

Thomas Thesen thomas.thesen@med.nyu.edu
Comprehensive Epilepsy Center, Department of Neurology
School Of Medicine, New York University
New York, NY 10016, USA

Karen E. Blackmon karen.blackmon@nyumc.edu
Comprehensive Epilepsy Center, Department of Neurology
School Of Medicine, New York University
New York, NY 10016, USA

Ruben Kuzniecky ruben.kuzniecky@nyumc.edu
Comprehensive Epilepsy Center, Department of Neurology
School Of Medicine, New York University
New York, NY 10016, USA

Orrin Devinsky od4@med.nyu.edu
Comprehensive Epilepsy Center, Department of Neurology
School Of Medicine, New York University
New York, NY 10016, USA

Carla E. Brodley c.brodley@neu.edu
College of Computer and Information Science
Northeastern University
Boston, MA 02115, USA

Editor: Benjamin M. Marlin, C. David Page, and Suchi Saria

Abstract

Focal cortical dysplasia (FCD) is the most common cause of pediatric epilepsy and the third most common cause in adults with treatment-resistant epilepsy. Surgical resection of the lesion is the most effective treatment to stop seizures. Technical advances in MRI have revolutionized the diagnosis of FCD, leading to high success rates for resective surgery. However, 45% of histologically confirmed FCD patients have normal MRIs (*MRI-negative*). Without a visible lesion, the success rate of surgery drops from 66% to 29%. In this work, we cast the problem of detecting potential FCD lesions using MRI scans of MRI-negative patients in an image segmentation framework based on hierarchical conditional random fields (HCRF). We use surface based morphometry to model the cortical surface as a two-dimensional surface which is then segmented at multiple scales to extract superpixels of different sizes. Each superpixel is assigned an outlier score by comparing it to a control population. The lesion is detected by fusing the outlier probabilities across multiple scales using a tree-structured HCRF. The proposed method achieves a higher detection rate, with superior recall and precision on a sample of twenty MRI-negative FCD patients as compared to a baseline across four morphological features and their combinations.

Keywords: Conditional Random Fields, LOF, Focal Cortical Dysplasia, Epilepsy

1. Introduction

Epilepsy is a common neurological disorder, affecting approximately 1% of the world's population (Hauser and Hesdorffer, 1990). It is characterized by profound abnormal neural activity during seizures and inter-ictal periods. Uncontrolled epilepsy can have harmful effects on the brain and has increased risk of injuries and sudden death (Bernasconi et al., 2011). About one third of epilepsy patients suffer from treatment-resistant epilepsy (TRE); their seizures cannot be managed by medicine (Kwan and Brodie, 2000). Cortical malformations, particularly focal cortical dysplasia (FCD) is recognized as the most common source of pediatric epilepsy and the third most common source in adults with TRE (Kuzniecky and Barkovich, 2001; Lerner et al., 2009).

Early detection and subsequent surgical removal of the FCD lesion is the most effective treatment to stop seizures and is often the last hope for TRE patients. A growing number of studies demonstrate that surgery is highly effective for FCD patients (Benbadis et al., 2003). However, the surgical procedure remains underutilized. One of the main reasons for this is that about 45% of histologically-verified FCD patients have normal MRIs (Wang et al., 2013). The chance of success for the surgical resection based on a visually detected lesion (*MRI-positive*) is 66%, which drops to 29% when the MRI is read as normal (*MRI-negative*) (Bell et al., 2009). Therefore patients who lack an MRI-visible lesion are less likely to be referred to specialized epilepsy centers by neurologists (Benbadis et al., 2003) likely because epilepsy specialists are reluctant to operate without a well-defined lesion (Tllez-Zenteno et al., 2005).

In this paper, we present an automated method for detecting FCD lesions using surface-based morphometry (SBM) (Dale et al., 1999) to extract the cortical surface from the T1-weighted structural MRI scans. The key advantage of using SBM is that the cortical surfaces of different individuals can be registered to an average surface allowing for point-wise comparisons between different individuals. The proposed method first applies a multiscale segmentation of the cortical surface and then combines the results via a hierarchical conditional random field (HCRF) (Reynolds and Murphy, 2007). Because we do not have accurately labeled training data, we cast the problem as a semi-supervised outlier detection task and thus we extend the HRCF framework proposed in (Reynolds and Murphy, 2007) to perform semi-supervised outlier detection. The resulting outlier regions (lesions), sorted by their probability and surface area, are shown to a team of radiologists and neurosurgeons who can combine the MRI lesion-detection results with other information such as the pattern of seizure onset and the patient's intracranial EEG (iEEG) data to determine the final candidate resection zone.

HCRFs have been effectively applied in computer vision for semantic image labeling (Plath et al., 2009), figure-ground segmentation (Reynolds and Murphy, 2007) and object detection (Awasthi et al., 2007). To train the HCRF, previous applications required either

that each pixel in the image have a label or that a bounding box around the object(s) of interest is provided. Consequently, the accuracy of the labels directly impacts the final performance of the HCRF. In this paper, we propose an extension of the HCRF-based image segmentation and object detection framework for problems in which pixel-level labels are not available. For our task, we are given the MRIs of patients and of normal controls, but information about which pixels form a lesion in the patient MRIs is either missing or highly noisy (Ahmed et al., 2015). Thus, we have a global label for each image indicating either that the image contains no lesions (healthy control) or that it contains one or more lesions (FCD patient). In Section 2 we explain in more detail why we cannot use the lesions/resection zones of previously treated patients as labels during training.

In a preliminary version of this work (Ahmed et al., 2014), we applied the proposed HCRF framework to both MRI-positive and MRI-negative patients using only cortical thickness as the feature of interest. Our preliminary findings showed that HCRF was able to achieve a higher detection rate with significantly higher precision and recall as compared to a recently reported semi-supervised approach (Thesen et al., 2011) and a human expert.[1] The current work extends that body of work by evaluating the proposed method on three new morphological features. In addition we investigate different mechanisms of combining the detections produced by the different features to achieve higher detection rates. In our experiments we observed that it is not straightforward to combine the detections of individual features, because some of the features are noisier than others i.e., have a higher false positive rate than others. We show that this issue can be overcome by either using feature selection or changing the cluster ranking criterion. We provide extensive results on a larger set of patients[2] as compared to our previous work, showing that feature selection can be used to achieve a high detection rate by appropriately tuning the ranking criterion.

The remainder of this paper is organized as follows. In Section 2 we briefly introduce SBM, the different morphological features that can be extracted within the SBM framework, and describe the current state of the art in lesion detection using SBM and its shortcomings. In Section 3 we detail how we pose the lesion detection task within the object detection/segmentation framework, where the saliency of the target object is defined based on its "outlier-ness". We present different methods for combining the results of using four different morphological features in our proposed lesion detection framework. Section 4, provides the details of our ranking and evaluation methodology and it also provides an empirical comparison of our approach and a baseline approach across different morphological features and their combinations.

This work makes a significant contribution toward the detection of FCD lesions. Our empirical evaluation demonstrates that the proposed method was able to detect abnormal regions within the resection zones in a higher number of MRI-negative patients as compared to a baseline approach across four morphological features and their combinations. Not only was our method able to achieve a higher detection rate, it did so while achieving significantly higher precision and recall. Indeed, our 75% detection rate on the MRI-Negative patients

1. We omit the results of the blind comparison to a human expert in this paper. Interested readers can refer to (Ahmed et al., 2014).
2. The current sample has twenty patients, fourteen of which are retained from the previous study. In Section 4 we exlain that the twenty patients comprise the *entire* set of MRI-Negative patients successfully treated via surgical resection over a three year period at NYU's Comprehensive Epilepsy Center.

in our evauation dataset (compared to a human expert detection rate of 0%), suggests that this method can be used as an effective tool in the pre-surgical evaluation of TRE patients who are likely to undergo surgical resection; our method has started to be incorporated in the weekly meetings of radiologists and neurosurgeons to help identify the seizure onset zones in MRI-Negative patients. Thus, this work has the potential to increase the number of patients who are referred to resective surgery, and ultimately who are seizure free after surgery.

Our contribution to machine learning in addition to making progress on a challenging and important application is a new method for using HCRFs for binary object detection/segmentation for which only image captions are available. The proposed method can be generalized for other data modalities such as time series data, where it can be used for time series segmentation. A caveat to this contribution is that the individual data (images/time series) must be able to be accurately aligned such that a one-to-one correspondence can be made between them.

2. Surface Based Morphometry

Surface based morphometry (SBM) provides the means to characterize and analyze the human brain by explicitly modeling the cortex using a suitable geometric model (Dale et al., 1999). The cortical surface represents the outer layer of the brain modeled as a folded two-dimensional surface in three-dimensional space. T1-weighted structural MRI scans are used to extract the cortical surface by delineating the boundary between the gray and white matter (Dale et al., 1999). This process is referred to as surface reconstruction, and involves a number of reconstruction and segmentation steps aimed at locating the gray/white matter boundary up to submillimeter accuracy (Fischl et al., 1999a). The reconstructed surface is represented as a triangulated mesh and at each vertex different morphological features are estimated to characterize the cortex. It should be noted that the spatial resolution of the reconstructed surface is different from that of the MRI volume.

This section introduces surface based morphometry. We first describe the different features that can be used to characterize the cortex that have been used for detecting FCD lesions. We also provide the related work where machine learning methods have been used in conjunction with SBM to detect FCD lesions.

2.1 Morphological Features

In this work, we use four morphological features to characterize the cortex:

1. *Cortical thickness:* represents the thickness of the cortex which is defined as the distance between the gray/white matter boundary and the outermost surface of the gray matter (pial surface). It is calculated at each vertex using an average of two measurements (Fischl and Dale, 2000): (a) the shortest distance from the white matter surface to the pial surface; and (b) the shortest distance from the pial surface at each point to the white matter surface.

2. *Gray/white-matter contrast (GWC):* represents the degree of blurring at the gray/white-matter boundary. GWC is estimated by calculating the non-normalized T1 image intensity contrast at 0.5mm above and below the gray/white boundary with trilinear

interpolation of the images. The range of GWC values lies in $[-1, 0]$, with values near zero indicating a higher degree of blurring of the gray/white boundary.

3. *Sulcal depth:* characterizes the folded structure of the cortex. It is estimated by calculating the dot product of the movement vectors with the surface normal (Fischl et al., 1999a), and results in the calculation of the depth/height of each point above the average surface. The values of sulcal depth lie in the range $[-2, 2]$ with lower values indicating a location in the sulcus whereas higher values indicate a location on the gyral crown.

4. *Curvature:* Curvature is measured as $\frac{1}{r}$, where r is the radius of an inscribed circle and mean curvature represents the average of two principal curvatures with a unit of 1/mm (Pienaar et al., 2008). Mean curvature quantifies the sharpness of cortical folding at the gyral crown or within the sulcus, and can be used to assess the folding of small secondary and tertiary folds in the cortical surface.

Thesen et al. (2011) nominate cortical thickness along with GWC as the two most informative features for detecting FCD lesions in MRI-positive patients, using a vertex-based detection scheme. Similarly, Hong et al. (2014) identify GWC, cortical thickness and sulcal depth as being more sensitive to detecting FCD lesions in MRI-negative patients. However, they also note that most of the false positives detected using their proposed approach were largely caused by sulcal depth. The technical details of both these works are provided in the next subsection. In this study we use the four features described above, and also analyze different mechanisms to combine their detections in order to achieve higher sensitivity.

After surface reconstruction, different morphological transforms can be applied to register the cortical surface to a standard surface also known as a group-atlas. Registration is achieved by aligning specific sulcal and gyral patterns across the *reconstructed* cortical surfaces allowing for a more precise comparison of individual cortical structures across subjects (Fischl et al., 1999b). SBM has been used successfully for analyzing and detecting neurological abnormalities in various neurological disorders such as Schizophrenia (Rimol et al., 2012), Autism (Nordahl et al., 2007), and Epilepsy (Thesen et al., 2011; Hong et al., 2014).

2.2 Current Methods for Detecting FCD using Surface Based Morphometry

In this section we first define the related work with regards to automated techniques of FCD lesion detection. We then discuss the critical limitations of existing approaches. Most of the techniques reported thus far in literature deal either with MRI-positive patients (Besson et al., 2008; Thesen et al., 2011) or patients who were initially deemed MRI-negative during their preliminary radiological screening, but later their lesions were found to visible on MRI (Hong et al., 2014). As opposed to these studies, we evaluate our proposed methods on a sample of *pure* MRI-negative patients whose lesions are not visible on their MRI.

SBM has been used in conjunction with supervised machine learning techniques to identify lesions in FCD patients. Besson et al. (2008) use texture, GWC and a number of morphological features including cortical thickness to represent each vertex on the reconstructed cortical surface of MRI-positive patients. They then train a neural network to classify each

vertex as being normal or lesional. Hong et al. (2014) developed a two-stage Fisher linear discriminant analysis (LDA) (Bishop, 2006) classifier to detect FCD lesions in patients who were radiologically classified as MRI-negative. Initially they train a vertex-level classifier that classifies each vertex on the reconstructed cortical surface as being lesional or non-lesional for both controls and patients. These detections are further refined using a second LDA classifier that is trained to distinguish between actual FCD lesions (detections made inside the manually traced resection zones of patients) and spurious lesional detections made on controls. The lesions for all the patients included in the study were ultimately found to be visible and were manually traced by an expert using texture-based maps. Thesen et al. (2011) use a semi-supervised uni-variate z-score based thresholding approach on registered SBM data of MRI-positive patients to classify each vertex as being lesional or normal, using cortical thickness, GWC, curvature, sulcal depth and Jacobian-distortion, individually. They nominate cortical thickness along with GWC as being the most informative features for FCD lesion detection in MRI-positive patients.

These studies classify individual vertices of the cortical surface as lesional or normal, using labeled training data from MRI-positive patients and controls. There are four crucial issues that these studies fail to address:

(1) The goal of resective surgery is to remove the entire lesion. If any part of the lesion is left behind, the outcome will not be successful. This introduces label noise, because the expert-marked lesion can contain normal vertices; the margin around the lesion is marked in a "generous" manner so as to increase the chances of capturing the entire lesion. In our previous work (Ahmed et al., 2015) we used a stratified logistic regression classifier to detect lesions in MRI-negative patients. By manually reducing the resection masks for MRI-negative patients to correct for label noise we were able to achieve a detection rate of 58%, as opposed to 12% when the original resection masks were used as the ground truth.

(2) These studies assume that individual vertices are i.i.d., completely ignoring the spatial correlation that exists between neighboring vertices. It has been shown in other domains such as object detection and segmentation in natural images, that modeling spatial correlations leads to superior performance (Reynolds and Murphy, 2007; Plath et al., 2009).

(3) Vertex-based classification methods typically employ a post-processing method to reduce the false positive rate. In this strategy a portion of the vertices labeled lesional by the classifier are re-labeled as normal. This can be done by training a second-level classifier to classify the detected clusters as lesional or non-lesional (Besson et al., 2008; Hong et al., 2014). Similarly, different heuristics can also be used such as the surface area of the detected clusters (Thesen et al., 2011). Discarding any detected region based on its size or surface area can result in discarding the actual lesion or part of the lesion, because FCD can be located in any part of the cortex, is highly variable in size, and may occur in multiple lobes (Blumcke et al., 2011).

(4) Results are evaluated on MRI-positive patients, but the real challenge is to find lesions in MRI-negative patients.

Our proposed method is designed to explicitly address these issues. First, we model lesion detection as an outlier detection problem. The assumption is that a lesional region is an outlier in feature space when compared to the same region across a control population. This view eliminates the use of noisy class labels. Second, instead of classifying individual vertices we classify segmented patches of the cortex. The patches are obtained

using unsupervised segmentation of the flattened cortex that isolates regions of homogeneous feature values. As the size of the FCD lesions has a wide range, using a single scale to isolate the lesion may not be effective. To minimize the chances of missing the lesion, we employ a multiscale strategy where the segmentation is carried out at different scales of varying granularity. The interplay between the patches obtained in this scale hierarchy is modeled as a tree structured HCRF, rooted at the most crude scale and having leaves at the finest scale. We fully exploit the spatial dependencies in the data by classifying image patches rather than by individual vertices, and furthermore larger spatial interactions are accomodated by explicitly modeling the dependencies using HCRF between image patches at different scales. Third, we define a ranking criterion which takes into account both the size and probability of a cortical region (cluster) that is labeled as being lesional. This ranking approach eliminates the need to post process the results, and provides a natural way of presenting the results to a radiologist to function as a focus of attention mechanism. Finally, we evaluate our approach on MRI-negative patients whose resections contained the primary FCD lesion, confirmed by a histological exam on the resected tissue. Since the proposed method is semi-supervised we use the z-score based approach (Thesen et al., 2011) as the baseline from which to contrast the performance of the HCRF method. Furthermore, the patients and healthy controls used in this work were treated and scanned at the NYU Comprehensive Epilepsy Center, where the z-score based method is currently part of the pre-surgical evaluation protocol of epilepsy patients. The next section describes the details of the HCRF construction and inference.

3. Hierarchical Conditional Random Fields

Hierarchical Conditional Random Fields (HCRFs) provide a suitable framework for supervised image segmentation (Reynolds and Murphy, 2007), object detection and semantic image labeling (Plath et al., 2009). In the original HCRF framework proposed for figure-ground segmentation (Reynolds and Murphy, 2007), an image is first segmented into a number of patches at different scales. Each patch is then classified as being part of the background or foreground, using a suitable binary classifier based on image features such as texture, scale-invariant feature transform (SIFT) (Lowe, 1999), etc. Exploiting the fact that the labels assigned to overlapping patches between different scales should agree, an HCRF (a tree-structured conditional random field) is constructed to model these inter-scale interactions. The image is thus modeled as a forest, where the root node for each tree corresponds to a patch obtained at the coarsest scale, while the leaves reside at the finest scale. The joint probability of all the patch labels is estimated by running inference on the HCRFs. The image is segmented by thresholding the final probabilities at the leaves. Plath et al. (2009) extend this framework to work with more than two classes. Mutliclass image labeling using HCRFs is also done in (Awasthi et al., 2007), where instead of obtaining image patches using segmentation, the authors impose a grid structure on the image at different scales and model the HCRF as a quad-tree structure. These multiscale methods are highly sensitive to the accuracy of pixel-level labels. For example in (Reynolds and Murphy, 2007) the bounding boxes around the region of interest (ROI) in training images were manually refined to eliminate extraneous pixels and this resulted in a significant increase in accuracy. In this section we first provide an overview of the different steps involved in the

construction and inference of an HCRF for lesion detection. Subsequent sections provide the necessary technical and processing details involved at each step.

3.1 Adapting HCRFs for Lesion Detection

For FCD lesion detection, we have training data from MRI-negative patients who have undergone surgical resection and are seizure-free. The resected coritcal region, can be used to obtain vertex-level labels which can be used to train a classifier. However, as explained previously these labels tend to be highly noisy and using them to train a classifier will result in noisy predictions (Ahmed et al., 2015). To ameliorate this problem we extend the HCRF framework proposed in (Reynolds and Murphy, 2007) to perform outlier detection on registered image data. In contrast to the approaches mentioned previously, we cannot utilize vertex-level labels. Our proposed method works in a semi-supervised manner, where only global labels are available i.e., whether the cortical surface belongs to a healthy control or a patient. Thus, we define an FCD lesion as a region of the brain which is considered an outlier when compared to the same region across a population of normal controls.

The construction of the HCRF for FCD lesion detection involves the following steps (Ahmed et al., 2014):

1. Segment the cortex at multiple scales, to obtain image patches of varying sizes.
2. Assign an outlier score to each image patch by comparing it to the same cortical region across the control population. This one-to-one comparison is made possible by registering all the control's and patient's cortical surface to the same average surface.
3. Construct multiple HCRFs, one for each image patch obtained at the coarsest scale.
4. Run inference on the HCRFs to calculate the posterior probability at each node. The final lesion is detected by thresholding the posterior at the leaves.

We start by describing our approach to segmentation.

3.2 Segmentation

The functional organization of the cortex is two-dimensional, e.g., the functional mapping of the primary visual areas (Van Essen et al., 1998). Therefore, as an initial simplification we have chosen to work with the flattened cortex because it will simplify the segmentation procedure and will allow us to use already well-established image segmentation techniques. Using SBM, the cortex is modeled as a two-dimensional surface, which on average contains approximately 0.15 million vertices. Even though it is possible to flatten the entire cortex, it's segmentation and subsqeuent inference on the resulting HCRFs would require significant computational resources. Thus, to reduce the processing overload we have chosen to subdivide the lesion detection task into smaller regions of the cortical surface as defined by a standard neuroanatomical atlas, which outlines cortical regions based on their morpho-functional properties (Fischl et al., 2002). These regions are also known as *parcellations*. Instead of segmenting the entire cortical surface at once, we isolate these parcellations one at a time and flatten them individually to obtain a standard two-dimensional image, which we then segment at multiple scales. Any morphological feature (e.g., cortical thickness,

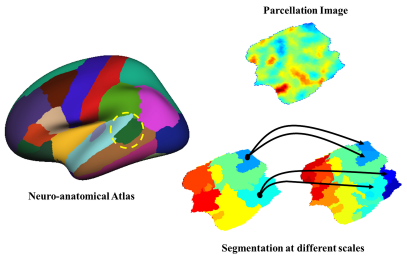


Figure 1: Constructing an HCRF using a standard neuroanatomical atlas (*left*), and a parcellation image (*top-right*). Any morphological feature can be used to represent the image (the image is created using cortical thickness). At the bottom we have image patches obtained at two different scales using Quickshift. Each image patch on the coarser scale (*bottom-left*) becomes a root having children at the adjacent finer scale (*bottom-right*).

curvature, etc.), can be used to represent the intensity values in the resulting image. Figure 1 illustrates the overall HCRF construction process for a parcellation.

We use quick shift (Vedaldi and Soatto, 2008b) for unsupervised segmentation. One of the main advantages of using quick shift is that the number and size of segments need not be specified. Additionally, quick shift does not penalize for boundary regions, and produces a diverse set of segments having different shapes and sizes. It should be noted that any segmentation method can be used, as long as it has the ability to segment the image at different scales.

The standard quick shift algorithm is a fast mode seeking algorithm similar to mean shift (Comaniciu and Meer, 2002). It performs a hierarchical segmentation of the image, where the sub-trees represent image segments. It has two parameters namely the size of the Gaussian kernel (σ) used by a Parzen window density estimator, and the maximum distance (Δ) between two pixels permitted while remaining part of the same segment. We vary the scale parameter σ to change the average size of segments, and set Δ to be a multiple of σ (Vedaldi and Fulkerson, 2008a). Thus, higher values of σ produce larger segments. By using different combinations of these parameters, we construct the scale-hierarchy which is the basic building block of the HCRF, as explained next.

3.3 Multiscale HCRFs

Once the multiscale segmentation is complete for a particular subject, we obtain a set of patches at different scales. Let I_p^k be the p^{th} patch obtained at the k^{th} scale. We can collect the corresponding patches from all controls and then estimate a label $y \in \{0,1\}$ for I_p^k, where $y = 1$ indicates that I_p^k is an outlier. This label cannot be considered independent from the labels of other patches that overlap with I_p^k at other scales.

We model the joint prediction of these mutually dependent labels of all the patches using a tree structured HCRF. Let I_p^{k+1} be an image patch at level $k+1$, it has a parent I_q^k at the immediately coarser level k, such that I_q^k has maximal overlap with I_p^{k+1} (Reynolds and Murphy, 2007). We find the index q as follows:

$$q := \arg\max_q \frac{|I_p^{k+1} \cap I_q^k|}{|I_q^k|} \tag{1}$$

Each patch at the coarsest scale is the root of a tree having leaves at the finest scale. Therefore, the parcellation image is represented by a forest, where each tree is modeled as an HCRF, as shown in Figure 1.

CRFs model the joint conditional probability distribution of all the patch labels $y = (y_1, \ldots, y_n)$ in the tree based on the values of the input morphological feature (x). Generally, this can be written as:

$$p(y|x,\theta) = \frac{1}{Z(x,\theta)} \prod_i \phi(y_i|x,\theta) \prod_{i,\pi(i)} \psi(y_i, y_{\pi(i)}) \tag{2}$$

where, $\pi(.)$ represents the parent patch, and $Z(x,\theta)$ is the normalization constant also called the partition function. $\phi(.)$ is called the node potential and represents the local evidence for the label y_i based on the observed data x. The edge potentials that model the coupling between adjacent labels are represented by $\psi(.)$. As the graph is a tree we can efficiently calculate $Z(x,\theta)$ and the posterior probabilities of the patch labels at all scales using standard belief propagation (Pearl, 1988).

When labeled training data is available the node and edge potentials are parameterized, and the parameters are learned jointly (see (Sutton and McCallum, 2010) for details). For our application, because the labels are noisy and we have chosen to work in an unsupervised manner, we set the node and edge potentials separately, which we describe next. Similar strategies for parameter estimation in HCRFs have been used for figure-ground segmentation (Reynolds and Murphy, 2007) and for object detection (Plath et al., 2009) in natural images.

3.3.1 Node Potentials

The node potential is modeled to reflect our belief about the abnormality of an individual image patch. Most of the available outlier detection mechanisms produce outlier scores that are poorly calibrated i.e., the range of the outlier score is dependant on the dataset (Schubert et al., 2012). This makes it difficult to compare the outlier scores between datasets produced by the same method. Popular outlier detection methods such as local outlier factor (LOF) (Breunig et al., 2000) and local correlation integral (LOCI) (Papadimitriou et al., 2003) suffer from the same problem. In our case we would like to work with an outlier detection method that produces standardized scores that carry the same semantics at each scale and can be compared between different scales. This is an important design choice because running inference on non-standardized scores, which are not comparable between different scales, will produce meaningless results. To overcome this, we have chosen to work with local outlier probabilities (LoOP) (Kriegel et al., 2009), a standardized version of LOF that

produces standardized scores within the range $[0, 1]$ which can be treated as the probability that a data point is an outlier.

LoOP assumes that each data instance x has a context set $S \subseteq D$, and the set of distances between x and $s \in S$ has a Gaussian distribution (Kriegel et al., 2009). The standard deviation of these distances $\sigma(x, S)$ combined with a significance factor λ produces the *probabilistic set distance of x to S* (Kriegel et al., 2009) defined as:

$$pdist(\lambda, x, S) := \lambda \cdot \sigma(x, S) \tag{3}$$

where S is determined using a k-nearest neighbor query. The parameter λ defines the sensitivity of the final probability estimates. It denotes that any instance that deviates more than λ times the standard deviation would be considered an outlier. Its values are analogous to the empirical confidence levels defined for the standard normal distribution (Kriegel et al., 2009). The probabilistic local outlier factor for x can then be calculated in a manner similar to LOF:

$$PLOF_{\lambda,S}(x) := \frac{pdist(\lambda, x, S)}{E_{s \in S}[pdist(\lambda, s, S(s))]} - 1 \tag{4}$$

PLOF values of greater than zero indicate that the given instance may be an outlier. In order to convert a PLOF value into a probability estimate it can be assumed that they are distributed around 0 with a standard deviation calculated as $\sqrt{E[(PLOF)^2]}$. The final probability can then be calculated as:

$$LoOP_S(x) := \max\left\{0, \mathrm{erf}\left(\frac{PLOF_{\lambda,S}(x)}{\lambda\sqrt{2E[(PLOF)^2]}}\right)\right\} \tag{5}$$

where, erf(.) is the Gauss error function (Andrews, 1992).

3.3.2 Edge Potentials

Each edge in the HCRF represents the dependency between the "parent" image patch at scale t and the "child" patch at scale $t + 1$. We set the edge potential to reflect the visual similarity between the two patches, using the chi-squared distance between the histograms of scale invariant feature transform (SIFT) features (Lowe, 1999) of the parent and child patches. Thus, the labels of image patches that bear close visual similarity to each other in the scale hierarchy are more strongly coupled than those with lower similarity. This heuristic is similar to one chosen by Reynolds and Murphy (2007).

To estimate the histograms of the SIFT features for each image, we initially learn a codebook of m codewords using the control data. For each control image in the subset we flatten and isolate the parcellation, and then calculate a SIFT feature vector at each pixel. These vectors are then clustered into m clusters using *k-means* clustering. Each feature has its own range of values and defines separate morphological properties of the cortex (see Section 2.1), we learn a separate codebook for each parcellation/feature combination. The edge potential between two adjacent nodes in the tree is then calculated as (Reynolds and Murphy, 2007; Plath et al., 2009):

$$\psi(y_i, y_j) = \begin{bmatrix} e^{\gamma.\eta_{ij}} & e^{-\gamma.\eta_{ij}} \\ e^{-\gamma.\eta_{ij}} & e^{\gamma.\eta_{ij}} \end{bmatrix} \tag{6}$$

where, γ is a free parameter that represents the strength of coupling between adjacent levels in the CRF and $\eta_{ij} = e^{-\chi^2(x_i, x_j)}$. x_l represents the normalized histogram of SIFT features for the l^{th} patch in the HCRF, and $\chi^2(.,.)$ is the chi-squared distance between two normalized histograms each having n bins and defined as:

$$\chi^2(P, Q) = \frac{1}{2} \cdot \sum_{i=1}^{n} \frac{(P_i - Q_i)^2}{(P_i + Q_i)} \tag{7}$$

where, P and Q are normalized histograms.

3.4 Lesion Detection

For each subject, we calculate the posterior probabilities at each node of the HCRF for every parcellation by running belief propagation (Pearl, 1988). The final detection is obtained by thresholding the posterior beliefs at the leaves of each HCRF (Reynolds and Murphy, 2007; Plath et al., 2009). Different strategies for thresholding can be used, such as defining a single threshold across all subjects, or calculating a threshold for each subject individually. In this work we calculate an adaptive threshold for each patient separately. This decision is based on the observations that 1) FCD lesions can manifest differently for different individuals, and 2) the morphological features vary with different demographic factors such as gender and age. For example cortical thickness is correlated with the age of the patient (Salat et al., 2004). To this end, we sort the posterior probabilities and define the threshold as the lowest probability among the top K probability estimates. In practice the value of K can be left as a free parameter that the user can vary to see the different regions which are deemed lesional with varying levels of confidence. Thus, the radiologist has a knob to turn which shows more/fewer possible candidate lesions. This is a desirable feature, because the detection scheme presented here is designed to be a part of the comprehensive pre-surgical evaluation protocol that includes MRI, Positron Emission Tomography (PET), scalp EEG and iEEG. The final resection target is determined by combining evidence from all evaluations. Therefore, the ability to generate multiple cortical maps delineating possible lesions at different confidence levels provides a richer set of evidence which in turn increases the probability of capturing the actual lesion.

4. Empirical Evaluation

Our data consists of MRI-negative patients who have undergone resective surgery and for whom their resected tissue was histologically verified to contain FCD. Each patient who undergoes surgery is assigned an "Engel" class. An Engel class of 1 represents complete seizure freedom while an Engel class of 4 represents no improvement. We selected only patients with an Engel class outcome of 1 for our experiments in order to verify that the region resected was indeed the primary lesion and that no additional epileptogenic lesions were present in other parts of the brain. This resulted in a dataset with twenty MRI-negative patients, whose information is provided in Table-1. This may appear to be a small dataset, but few patients proceed to surgery when no visible lesion is found on their MRI, and of those that do, less than a third experience complete seizure freedom (Bell et al., 2009). These twenty patients represent *all* MRI-negative patients who underwent surgery

Patient	Location	Age	Sex	Seizure Onset Age	Seizure Frequency	Engel Class
NY46	R Temporal	41	M	3	52	1
NY67	R Temporal	27	M	13	1825	1
NY149	R Frontal	32	F	11	1460	1
NY159	R Parietal	21	F	8	2190	1
NY226	L Temporal	40	F	5	8	1
NY255	R Temporal	20	F	15	48	1
NY294	R Temporal	51	F	1	12	1
NY315	L Occipital	47	F	9	12	1
NY322	R Frontal, Insular & Temporal	24	F	9	12	1
NY338	R Temporal	30	M	19	120	1
NY343	R Temporal	32	M	21	1825	1
NY351	L Temporal	30	M	12	12	1
NY371	R Temporal	17	M	17	365	1
NY375	R Temporal	16	F	2	54	1
NY394	R Temporal	27	M	19	72	1
NY404	R Temporal	51	F	45	6	1
NY441	L Temporal	41	M	31	72	1
NY451	R Inferior Parietal	25	M	9	912	1
NY455	L Temporal	61	M	19	12	1
NY486	L Temporal	29	F	27	96	1
Mean		**33.10**		**14.60**	**458.25**	1

Table 1: Demographic and seizure-related information for the MRI-negative patients.

at New York University comprehensive epilepsy treatment center during the past three years, and were classified post-surgically as Engel class 1. Developing automated lesion mechanisms for MRI-negative patients is an active area of research and our sample size is consistent with the existing work in the domain (Besson et al. (2008); Thesen et al. (2011); Hong et al. (2014)). However, in contrast to our evaluation, most of these studies evaluate their proposed detection schemes on MRI-positive patients i.e., patients whose lesion was visible on the MRI during the initial evaluation or was found visually at a later stage. It is important to note that our sample consists of "pure" MRI-Negative patients, and therefore our results target that patient population where there is an actual need of an automated lesion detection scheme and where such a scheme can have a positive impact impact on the outcome of resective surgery.

4.1 Imaging

Imaging for both the subjects and the controls was performed on a Siemens Allegra 3T scanner. Image acquisitions included a conventional 3-plane localizer and a T1-weighted volume pulse sequence (TE=3.25 ms, TR =2530 ms, TI =1100 ms, flip angle =7 deg field of view (FOV) = 256 mm, matrix = 256×256, vertex size =$1 \times 1 \times 1.3$ mm, scan time: 8:07 min). Acquisition parameters were optimized for increased gray/white matter image

contrast. The T1-weighted image was re-oriented into a common space, roughly similar to alignment based on the AC-PC line. Images were corrected for nonlinear warping caused by no-uniform fields created by the gradient coils. In this study we have a total of 115 controls, 55 males (33.7 ± 12.5 years) and 60 females (32.0 ± 11.5 years). It should be noted that all the patients were scanned on the same scanner, and the data used here is based on these research scans, which is different from their original clinical scans, as most of them were referred from external epilepsy centers.

Resection Tracing: For all patients, the post-operative T1-weighted image (with the resection area removed) was rigid-body coregistered to the (intact) pre-operative T1-weighted image. The brain resection area was manually traced on the post-surgical MRI scan by a trained technician blinded to patient diagnosis and reviewed by a board-certified neurologist. The manual masks in the MRI volume were subsequently projected onto the cortical surface by assigning each MRI voxel to the nearest surface vertex. Because the surface has sub-voxel resolution, a morphological closing operation was used to fill in any unlabeled vertices.

4.2 Data Pre-processing and Parameter Selection

After the surface has been reconstructed using the freesurfer software[3] we used the Desikan-Killiany atlas (Desikan et al., 2006) to isolate the different parcellations. It should be noted that any suitable neuroanatomical atlas can be used to subdivide the cortical surface. Each parcellation is flattened to obtain a standard 2-d image, where the intensity of each pixel can be represented by any one of the four morphological features.

The values of the different parameters such as the segmentation scales, number of nearest neighbors in calculating the outlier probabilities, etc., depend on various factors, such as the size of the control population, the distribution of ages across the control cohort and the gender of the subject. We therefore present these parameters as actual free parameters that can be varied over a pre-set range of values to get different detection results. Whether an image patch is an outlier depends on the set of controls used to learn the "normal" model. Most morphological features vary with different demographic factors such as age, gender, handedness, etc. Ideally, we could choose a customized set of controls for each patient, but currently we do not have enough controls to customize for age and other factors, but we do select controls based on the patient's gender.

To select the parameters for the various aspects of our method, we used a validation set consisting of two MRI-positive and two MRI-negative patients, which are distinct from the patients used to evaluate our method. We used all 115 controls to learn a separate codebook of SIFT features for every parcellation/feature combination. Dense SIFT features were calculated at each pixel. We tested vocabulary sizes of 50, 100 and 500 and selected a vocabulary size of 50 as it resulted in higher recall and precision on the validation set. This codebook was used subsequently to estimate the histograms of SIFT features at each pixel location for all patient parcellation images in the test set.

Each parcellation image was segmented at three different scales using quick shift. We used $\sigma = \{2, 3, 4\}$ and Δ was set to 5σ. These values were chosen such that the smallest lesion in our validation set is over-segmented i.e., there are multiple segments that contain

3. Available at http://surfer.nmr.mgh.harvard.edu/

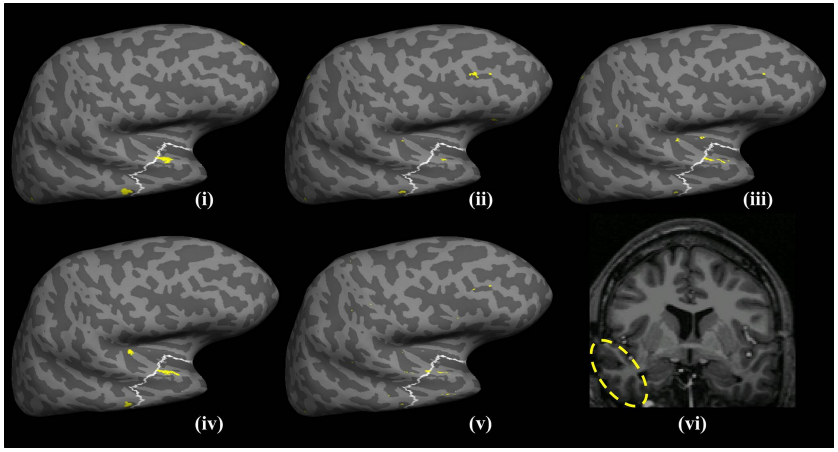


Figure 2: Detection results for NY67 using cortical thickness shown on an inflated model of the lateral cortical surface. The resected region is delineated as the white circled region and the detection results are shown as filled yellow regions. It can be seen that the lesion is detected at individual scales (i) and (iii) prior to combining the outlier probabilities using HCRF. However, at the second scale (ii) a large cluster is detected outside the resection. When these findings are combined using the HCRF as shown in (iv) the largest detected cluser is within the resection zone while the false detection in (ii) is suppressed. (v) shows the detection made by the the z-score based approach. The results are shown for the most stringent (first) threshold without any post-processing. (vi) shows the lesion highlighted on a T1 MRI slice.

the lesional area. This increases the probability that a patch can be entirely formed from lesional vertices, rather than having patches that partially overlap with the lesion, which would be harder to detect as outliers. Based on these settings, the validation set resulted an average of 4255 ± 107 HCRF models per patient, with 19292 ± 373 leaves at the finest scale using cortical thickness. Although the size of the validation set seems small as far as the number of patients are concerned, we conjecture that the resulting number of HCRF models and number of instances are adequate for setting the model parameters.

Finally, before performing outlier detection, we apply a standard dimension reduction technique on each patch using principal component analysis (PCA). Note that the PCA is done using only the control data. We retained the top m principal components that accounted for 95% of the variance in data. Based on results for the validation set (carried out independently for each feature), the parameters for outlier detection were set to $k = 10$ in LoOP and γ (c.f. equation (6)) was set to 50.

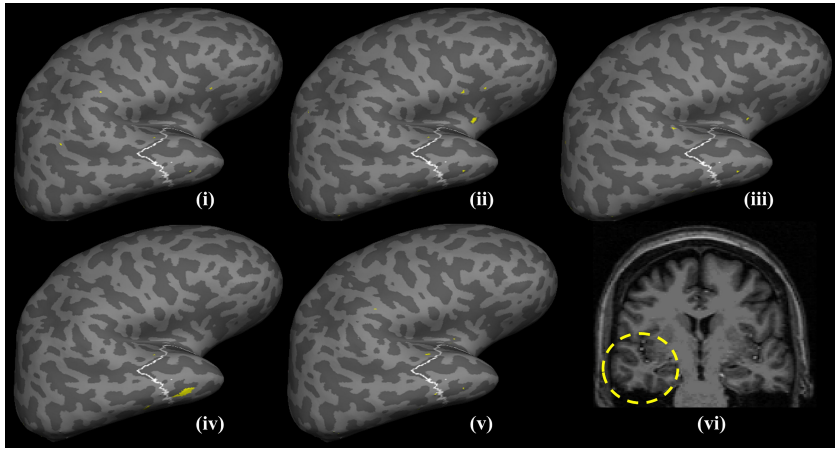


Figure 3: Detection results for NY294 based on curvature. The white outlined area represents the region that was resected, while the filled yellow patches represent the detected clusters at the first detection threshold for both the HCRF and the z-score based method. The detected clusters at the individual scales are shown in (i)-(iii). It can be seen that very small (almost negligible) clusters are detected that overlap with the resected region. However, after running belief propagation (iv) a large cluster is detected within the resection zone while the outliers are eliminated. (v) shows the results for z-score based method while (vi) shows the lesion highlighted on a T1 MRI slice.

4.3 Evaluation Methodology

The final detection for each subject is determined by thresholding the posterior probabilities at the leaves of the CRF, which represent the segments obtained at the finest scale. We determine the detection thresholds by dividing the last percentile of the final outlier probabilities into ten equal parts. The first threshold corresponds to the lowest probability in the highest 0.1% scores and so on. For the results presented in this section we determine five such thresholds to get five different possible detections. Because, this is an adaptive mechanism, it has a possible drawback that it always detects something even when the probabilities are very small. Thus we set 1×10^{-4} as the minimum probability, such that no threshold is calculated below this value. This limiting value was selected based on the observation that any threshold calculated below this value resulted in more than 80% of the cortex being labeled as lesional for the patients in the validation set.

We have chosen to evaluate and contrast the performance of the detection techniques in an information retrieval framework. We first calculate the clusters by thresholding the posterior probability at a given threshold. All the detected clusters are then ranked based

on the following score function:

$$score(c\,;\alpha) = \alpha s(c) + (1-\alpha)o(c) \;\; ; \;\; 0 \leq \alpha \leq 1 \tag{8}$$

where c is a cluster detected at a pre-defined threshold, $s(.) \in [0,1]$ is the relative surface area of the cluster calculated as the ratio between the surface area of c and the total surface area labeled as lesional. $o(.) \in [0,1]$ is a scoring function that represents the degree of "outlier-ness" of the cluster. For the HCRF, we model $o(.)$ as the average of the outlier probabilities calculated at each vertex that is part of the cluster. α is a tradeoff parameter such that $\alpha = 1$ defines a ranking that is based solely on cluster-size, while $\alpha = 0$ ranks the clusters based only on their average probability of being lesional. Intermediate values of α define a ranking in which a smaller cluster detected at a stringent threshold is ranked higher than a larger cluster detected at a more lenient threshold and vice versa. In the ideal case clusters having a higher rank should be within the lesion/resection zone of the patient.

We compare the results of our proposed technique against a recently reported univariate technique (Thesen et al., 2011) to detect FCD lesions using SBM. In this baseline approach all control and subject surfaces are registered to the average surface. After registration, it calculates the z-scores at each vertex for the subjects, which are then thresholded to obtain the detection results. We calculate the z-score based on gender matched controls instead of using all the controls. To facilitate comparison we calculate multiple thresholds in the exact same manner as outlined above for HCRF, and rank the clusters at each threshold based on Equation 8. We omit the last step of the baseline method, which post processes the detections to eliminate "small" clusters (Thesen et al., 2011). We have chosen this technique as the baseline method because, i) it is a semi-supervised approach and does not require accurate vertex-level labels, and ii) it has been part of the pre-surgical evaluation at the NYU comprehensive epilepsy treatment center where the patients included in our evaluation were treated. We use the following measures to evaluate our proposed method.

4.3.1 Detection Rate

Detection rate is defined as the number of patients for whom one or more detected clusters overlap with the resected area. Usually a post-processing step is applied to the raw detections before estimating the detection rate. Hong et al. (2014) train a classifier to distinguish between clusters detected in the resection/lesional area and the extra-lesional clusters using the training data, this classifier is then applied to the clusters detected on the test subject before estimating the detection rate. Similarly, in Thesen et al. (2011) all clusters below a pre-set size threshold are discarded, and a successful detection results if one or more of the remaining cluster overlap with the lesional area. Discarding any detected cluster based on its size increases the risk of discarding subtle lesions. Instead of discarding detected clusters, we use cluster ranking to estimate the detection rate. To this end, we calculate five thresholds based on the outlier probabilities for the HRCF method, and similarly for the z-score method. After ranking the detected clusters based on (Equation 8), at each threshold we consider a subject to be correctly detected if a cluster amongst the top n (where n is relatively small as compared to the total number of detected clusters) completely or partially overlaps with the lesion/resection. This produces more conservative estimates of the detection rate as compared to approaches that do not use cluster ranking.

4.3.2 Precision and Recall

In order to compare the quality of detections, we calculated the precision and recall for both HCRF and the z-score based method. To this end, we consider all detected clusters at each threshold. We define recall as the ratio of the total surface area of all the clusters that overlap with the resection zone to the surface area of the resection zone. Similarly, we define precision as the ratio of the surface areas of clusters overlapping with the resection zone to the sum of the surface area of all the detected clusters.

Accurately calculating the false postive rate for the proposed detection scheme is challenging for several reasons. A patient can have abnormalities outside the lesion/resection zone which may not be epileptogenic. For example, abnormal cortical thinning remote from the epileptogenic onset region has been observed in focal epilepsy (McDonald et al., 2008; Lin et al., 2007) and attributed to the destructive impact of chronic seizures on brain structure rather than from malformations during cortical development. This might result in elevated extra-lesional false positives when detecting structural malformations characterized by abnormal cortical thickness. In our previous work (Ahmed et al. (2014)) we compared our detections on MRI-positive patients with an expert neuroradiologist. In 50% of the cases, the expert identified abnormal regions that coincided with detections outside the resection that were classified as false positives by our evaluation methodology that used the resection zone as the ground truth. This problem becomes more challenging for MRI-negative patients whose structural abnormalities are not visible on their MRI. In order to circumvent the presence of false negatives in our labeled data that would result in elevated estimates of the false positive rate, we use precision to evaluate the efficacy of our proposed scheme. Furthermore, based on the existence of structural abnormalities outside the resection zone (*false negatives*) the precision estimates provided here should be treated as lower bounds.

4.4 Results

In our experiments we first evaluate the HCRF framework independently for each of the four morphological features: cortical thickness, gray/white-matter contrast, curvature and sulcal depth. In the next set of experiments we analyze different mechanisms of combining the detections from individual features. Recall, that the ranking function (Equation 8) has a direct impact on the detection rate and by setting the tradeoff parameter we can assign more weight to either cluster size or the average cluster outlier probability. To facilitate comparison between the proposed method and the baseline we initially set the tradeoff parameter α to 1, so that all clusters are ranked based only on their surface area.

Figure 4(a) shows the comparison of the detection rates for MRI-negative patients when cortical thickness is used to represent the cortex. HCRF performs better than the z-score baseline across all the five thresholds, for the top five detections. HCRF detects the lesion in 14 (70%) patients, while the baseline detects only 11 (55%) subjects when considering the top ten largest clusters. HCRF is also able to achieve higher recall and precision as shown in Figures 4(b)-4(c). The difference between the recall values of the proposed method (1.1140 ± 0.5654) and the baseline (0.8035 ± 0.4745) was significant at $t(9) = 7.9927,\ p < 0.001$. Similarly, the differences in precision for HCRF (10.4710 ± 1.0248) and the baseline (9.0608 ± 0.5577) were found to be significant at $t(9) = 6.1161,\ p < 0.001$ using a paired

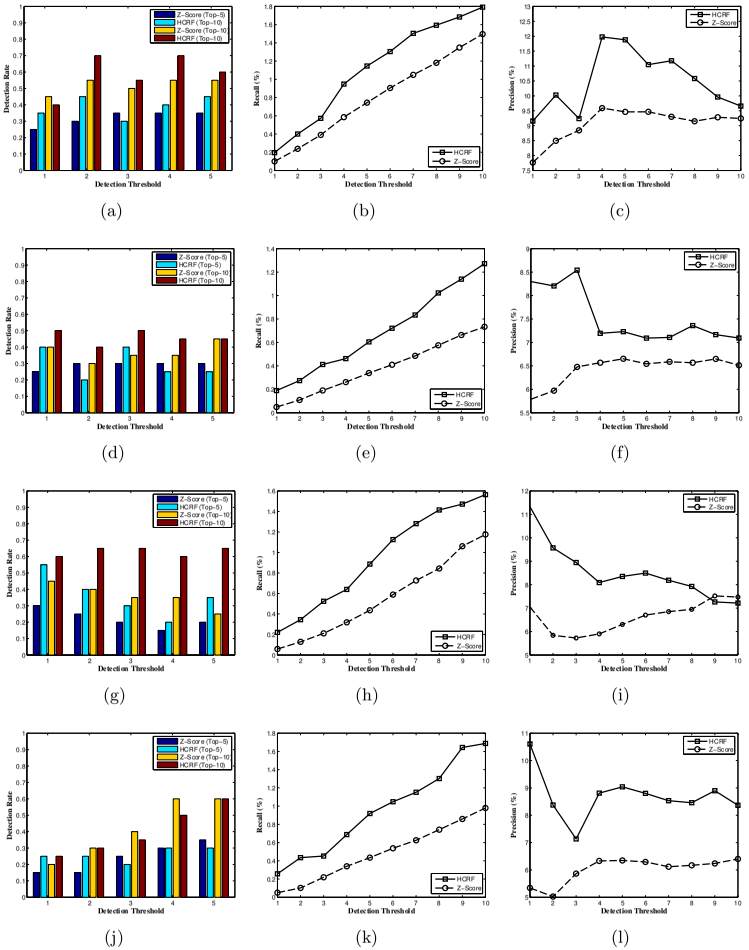


Figure 4: Comparison of detection rates, precision and recall between then HCRF based approach and the baseline method using thickness (a)-(c), GWC (d)-(f), curvature (g)-(i) and sulcal depth (j)-(l). Here, $\alpha = 1$ so that larger clusters are ranked higher (refer to Equation 8).

t-test. Figure-2 provides an example of the detected clusters using HCRF and the baseline for a patient.

Using GWC, HCRF is able to detect abnormal clusters within the resection zones of ten (50%) patients as opposed to the baseline that detects only nine (45%), as shown in Figure 4(d). Figure 4(e) shows the recall for HCRF method (0.6931 ± 0.3702) that is significantly higher ($t(9) = 7.1317,\ p < 0.001$) than the recall of the baseline method (0.3815 ± 0.2334). Figure 4(f) compares the precision of the HCRF and baseline using GWC. The differences in the precision values for HCRF (7.5286 ± 0.5769) and the baseline (6.4313 ± 0.2987) were found to be significant at $t(9) = 4.2350, p = 0.0022$ using a paired t-test. Although, using GWC HCRF is able to outperform the baseline, the resulting detection rate is worse than HCRF with cortical thickness.

Figure 4(g) shows the comparison of the detection rates using curvature to represent the cortex. HCRF dominates the z-score baseline across all the five thresholds, for both top five and top ten detections. HCRF detects abnormal clusters within the resection zones of 13 (65%) patients, while the baseline detects only 9 (45%) subjects when the top ten largest clusters are considered. Figures 4(h)-4(i) show that HCRF is able to achieve higher recall and precision, respectively. The difference between the recall values of the proposed method (0.9473 ± 0.4927) and the baseline (0.5549 ± 0.3903) was significant at $t(9) = 8.825,\ p < 0.001$. Similarly, the differences in precision for HCRF (8.5373 ± 1.2063) and the baseline (6.6313 ± 0.6597) were found to be significant at $t(9) = 3.9135,\ p < 0.0035$ using a paired t-test. Figure 3 shows the resulting detections from both the HCRF and the baseline when curvature is used to characterize the cortex for an MRI-negative patient.

When sulcal depth is used to represent the cortex, both the HCRF and the baseline method achieve the same detection rate. Both approaches are able to detect abnormal clusters that overlap with the resections of 12 (60%) patients (Figure 4(j)). However, as Figures 4(k)-4(l) show, HCRF is able to achieve higher recall and precision values. The difference between the recall values for HCRF (0.9585 ± 0.5013) and the baseline (0.4891 ± 0.3165) was significant at $t(9) = 7.7730,\ p < 0.001$. Similarly, the differences in precision for HCRF (8.7008 ± 0.8541) and the baseline (6.0124 ± 0.4679) were found to be significant at $t(9) = 8.0983,\ p < 0.001$ using a paired t-test.

Using individual features, HCRF is able to achieve a maximum detection rate of 70% while the baseline has a maximum detection rate of 60%, when top ten largest clusters are considered. For the baseline sulcal depth and cortical thickness achieve higher detection rates as compared to GWC and curvature. Cortical thickness outperforms all other features based on its average precision and recall. For the HCRF method sulcal-depth and curvature achieve identical performance with GWC ranking the lowest.

An important consideration is the degree of consensus between the individual features with respect to the detected patients. If there is some degree of disagreement between the features, then combining their detection can potentially increase the overall detection rate. Considering top ten largest clusters, two patients were not detected by any of the four features. Both cortical thickness and curvature detect a combined total of 16 patients, differing on one patient each. On the other hand all except a single patient detected by GWC and sulcal depth were also detected by either thickness or curvature. Based on these results if we combine the output probabilities of all four features, and then use the same thresholding and ranking technique we should be able to achieve a detection rate that is higher than the detection rate of the individual features. We investigate the combination of all four features in the next section.

4.5 Combining Features

In this section we explore the question of whether the HCRF based method will achieve a higher detection rate if the detections of the individual features are combined. As a first strategy, we can simply aggregate the posterior probabilities as obtained by the application of HCRF to each individual feature. Because every feature defines its own segmentation of a given parcellation image, it is not possible to directly aggregate the probabilities obtained at the leaves of the HCRF. To solve this issue we map the posterior probabilities obtained at the leaves of the HCRF, back to the cortical surface for each feature and then define a combination rule at every vertex. We use two basic aggregation rules, in the first we average the probabilities across the four features, and in the second each vertex is assigned a probability that is calculated as the maximum of the four individual probabilities. Similarly, for comparison to the baseline method we use the same techniques to calculate a single z-score estimate at each vertex for the baseline.

Aggregation based on averaging is similar to majority vote rule. In this strategy, vertices for whom most of the features have a high probability of being abnormal will be considered abnormal in the final detection. This has the effect of lowering estimation errors leading to a lower false positive rate by smoothing the outlier probabilities at each vertex. The second strategy that uses the maximum across the probabilities would label a vertex as lesional even if one of the features assigns it a higher outlier probability. This would lead to a higher detection rate along with a high number of false positives.

4.5.1 Performance Comparison to Baseline

Figure 5 shows the results of applying the two aggregation strategies in conjunction with the HCRF and baseline methods. In particular, Figure-5(a) shows the detection rates when the probabilities are averaged across features. It can be seen that the baseline performs better than the HCRF at the early thresholds, however the HCRF is able to produce better results as the threshold becomes more lenient. Considering the top ten largest clusters, HCRF is able to achieve a detection rate of 60% which is slightly higher than the baseline that achieves a detection rate of 55%. The recall and precision for the HCRF method are significantly higher than the baseline as shown in Figures 5(b)-5(c). Using a paired t-test, the difference in the recall values of the HCRF (1.0206 ± 0.5797) and the baseline (0.7046 ± 0.4410) was significant at $t(9) = 7.1317,\ p < 0.001$, and the difference in the precision values of the HCRF (9.2946 ± 0.9708) and the baseline (8.1422 ± 0.5595) was significant at $t(9) = 4.2350, p = 0.0022$.

When the posterior probability at each vertex is calculated as the maximum across the four features, HCRF achieves higher detection rates as shown in Figure 5(d). HCRF detects abnormal clusters within the resection zones of 13 (65%) patients, while the baseline detects only 10 (50%) subjects when top ten largest clusters are considered. HCRF achieves higher recall ($t(9) = 8.825,\ p < 0.001$) and precision ($t(9) = 3.9135,\ p < 0.0035$), as shown in Figures 5(e)-5(f), respectively.

4.5.2 Performance Comparison to Individual Features

A comparison of the detection rate of both aggregation strategies with the detection rates of the individual features is shown in Figure 5(g). We can see that both perform worse than

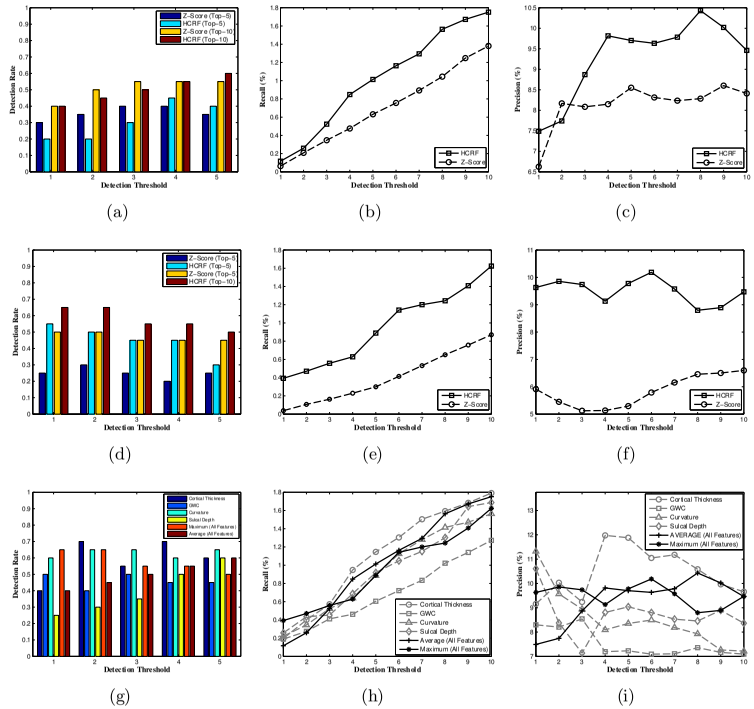


Figure 5: Comparison of detection rates, precision and recall between then HCRF based approach and the z-score based baseline method when the detection scores are averaged across features (a)-(c), and when the final output score is computed as the maximum across features (d)-(f). (g) contrasts the detection rate of both aggregation strategies with that of the individual features when the top ten largest clusters are considered and (h)-(i) provide the same comparison for recall and precision. Note that $\alpha = 1$ such that larger clusters are ranked higher (refer to Equation 8).

cortical thickness, and achieve a maximum detection rate of 65% when the top ten largest clusters are considered. The same detection rate is also achieved by curvature. Similarly, based on recall and precision values we can see that both the combination strategies fail to outperform any of the individual features with the exception of GWC.

Both the averaging and maximum strategies achieved lower recall and precision than cortical thickness (Figures 5(h) and 5(i), respectively). In addition to cortical thickness, the maximum strategy achieved lower average recall than both curvature and sulcal-depth. On

the other hand, for the averaging technique the differences in recall and precision were not significant when compared to curvature and sulcal depth. For the sake of brevity we omit the results of the paired t-tests that were used to establish these pairwise comparisons.

One reason for the failure of the combined strategies is that each feature has its own idiosyncrasies, which when not accounted for will introduce noise in the ranking/detection process. As an example, consider sulcal depth and curvature. Both features when used within the HCRF framework, achieve similar precision and recall but different detection rates. This shows that although, sulcal depth detects clusters within the resection zones of patients, it detects larger clusters outside the resection zone. If the detections of sulcal depth and curvature are combined then the noisy clusters detected by sulcal depth will cause a drop in the overall detection rate. There are two possible solutions: 1) select only informative features and discard the ones that are noisy, and 2) tune the tradeoff parameter (α) in the ranking function (Equation 8) such that the ranks of smaller clusters that are highly abnormal remain resilient to the presence of larger noisy clusters. It should be noted that changing the ranking function will have no effect on the overall precision and recall, because cluster ranks only influence the detection rate.

To explore option 1, feature selection, we selected cortical thickness and curvature because of their higher detection rates, precision and recall. We employ the same aggregation strategies as before, namely averaging and maximum. In Figure 6 we observe that when using only thickness and curvature, both the averaging and maximum strategies produce higher detection rates, precision and recall than the baseline (Figures 6(a)-6(f)).

More interestingly, when compared to individual features, the combination of curvature and cortical thickness is able to achieve significantly higher precision and recall, with the exception of cortical thickness (the average precision and recall is higher but the differences are not statistically significant), as shown in Figures 6(h)-6(i), respectively. However, as Figure 6(g) shows the detection rate although higher than when all four features are aggregated does not exceed that achieved by thickness alone. However these results confirm that when clusters are ranked based only on their size, both GWC and sulcal depth produce noisy detections, that can lower overall detection rate. Our finding that sulcal depth produces noisy or large extra-lesional clusters is also corroborated by Hong et al. (2014).

Next, we explore the effects of tuning the size/probability tradeoff parameter α, on the detection rate of both individual and combined strategies.

4.5.3 Ranking Criterion and the Detection Rate

Thus far we have fixed the ranking criterion to be the size of the detected cluster. However, as defined in Equation 8, we can tune the tradeoff parameter such that the cluster ranking criterion pays attention to both the size and the average outlier probability of the cluster. To ascertain how α influences the performance of HCRF framework, we varied α over its entire range of values and determined the detection rate for individual features, the combination of all features, and the combination of the top two features.

For a given input feature (or a combination of features) we divided the range of $\alpha \in [0, 1]$ uniformly into twenty 21 points. At each resulting value of α we re-estimated the ranking of the clusters. The detection rate corresponding to each value of α was determined by taking the maximum number of patients detected using the top ten ranked clusters, across

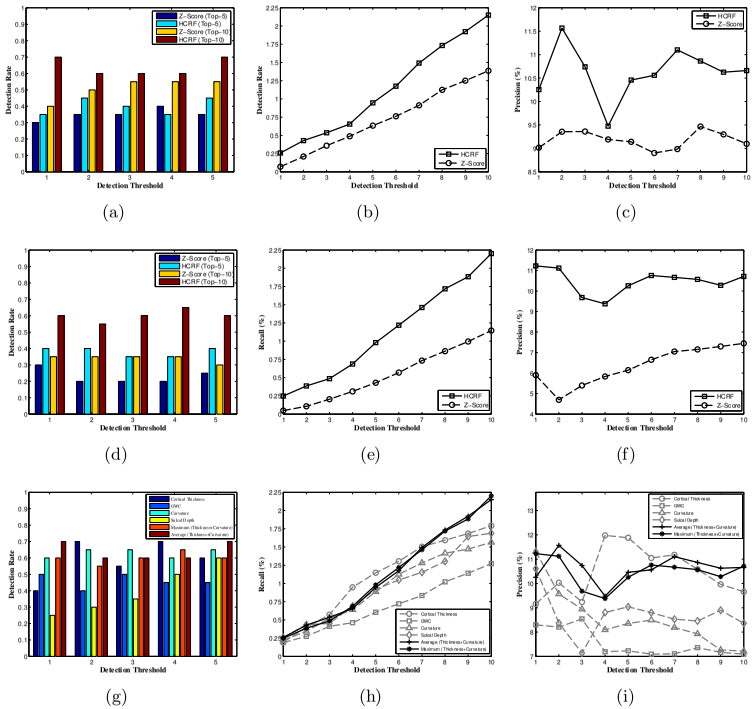


Figure 6: Comparison of detection rates, precision and recall between then HCRF based approach and the z-score based baseline method when the detection scores are averaged across thickness and curvature (a)-(c), and when the final output score is computed as the maximum between the two features (d)-(f). (g) contrasts the detection rate of both averaging and maximum with that of the individual features when the top ten largest clusters are considered and (h)-(i) compare the recall and precision. Note that, $\alpha = 1$ so that larger clusters are ranked higher (refer to Equation 8).

the first five thresholds. Figure 7(a) shows the detection rates of each individual feature for different values of α. Both cortical thickness and sulcal depth achieve their maximum detection rates when $\alpha = 1$, while curvature does so for intermediate values of $\alpha = 50,\ 75$. On the other hand the detection rate of GWC drops as α increases. Thus, every feature has its own idiosyncratic dependency on α which should be taken into account when combining the outputs from multiple features, especially because the goal is to improve the overall detection rate.

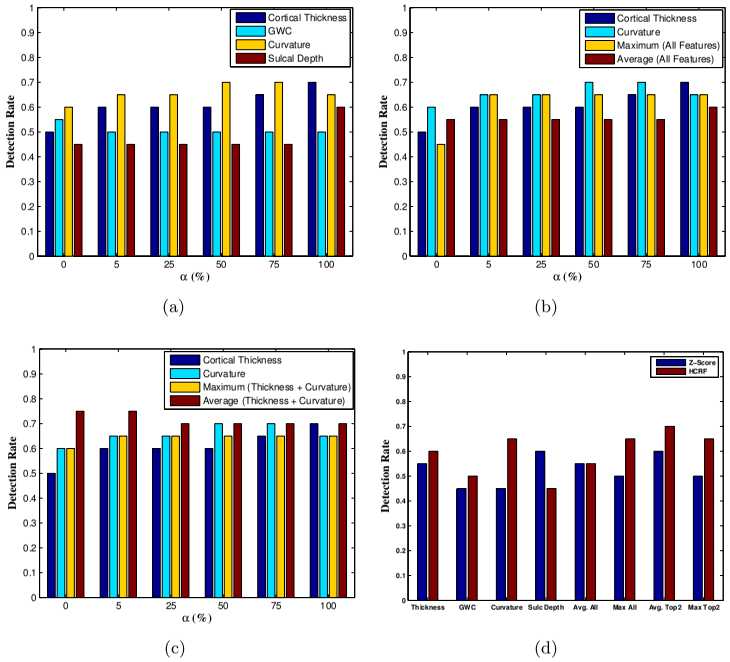


Figure 7: Effect of α on the detection rates of (a) individual features, (b) combination of all four morphological features and (c) combination of the top two ranked features. In (b) and (c) we have omitted the detection rates of sulcal depth and GWC to improve the clarity of the plot. (d) compares the overall median detection rate of HCRF with the baseline method using different input features and their combinations across the entire range of α.

Figures 7(b)-7(c) compare the influence of alpha on the detection rates of the two combination strategies (averaging and maximum) when all the features are used and when only cortical thickness and curvature are used, respectively. It can be observed that the highest detection rate, 75%, results from using an averaging technique to combine the posterior probabilities of cortical thickness and curvature (Figure 7(c)).

In order to contrast the performance of the HCRF based method based on different input settings across the entire range of α, we used a a two-sided Wilcoxon signed-rank test (Japkowicz and Shah, 2011). Figure 7(d) compares the median detection rate of HCRF and the z-score based method for different input features (and their combinations). For each individual feature HCRF achieves a higher detection rate as compared to the baseline, except

in the case of sulcal-depth, where the baseline is able to outperform HCRF. Similarly, when we combine cortical thickness and curvature to define the final detection, HCRF dominates the baseline, achieving the highest detection rate 75% using an averaging technique to combine the posterior probabilities of the two input features. However, when the same averaging technique is used to combine the results of all four features, both HCRF and the baseline perform comparably (albeit worse than using only two features) and the difference in their performance across the different values of α is not statistically significant.

5. Discussion

Any method of automated detection of FCD lesions is meant to augment the standard comprehensive clinical evaluation protocol for epilepsy surgery candidates. This standard protocol typically involves a neurological exam, scalp electroencephalography (EEG), neuropsychological exam, positron emission tomography (PET), and magnetic resonance imaging (MRI). Due to the common occurrence of widespread network abnormalities in focal epilepsy, each of these methods has a high rate of false positives. Thus, convergence of evidence from multiple sources is critical to determining the region(s) with the highest likelihood of hosting the seizure onset zone. In this work, we addressed this challenging task of detecting FCD lesions in a semi-supervised image segmentation framework. To this end, we developed a novel semi-supervised image segmentation method based on hierarchical conditional random fields (HCRF). We evaluated the proposed method on four morphological features, and also investigated different mechanisms of combining the outcomes of these input features.

In an empirical evaluation that involved 20 histologically verified MRI-negative patients, who had undergone resective surgery and were subsequently seizure-free, our proposed method was able to achieve higher detection rates using four morphological features as compared to a baseline method. Furthermore, when the detections based on these features were combined, HCRF was still able to detect abnormal clusters within the resection zone of a higher number of patients as compared to the selected baseline. Not only did the proposed method have a high detection rate, it also achieved significantly higher precision and recall across all features and their combinations.

Furthermore, in this work we establish that each of the four morphological features, namely cortical thickness, GWC, curvature and sulcal depth exhibit different behavior for different settings of the cluster ranking criterion and some of them produce noisier detections as compared to others. These two observations show that any method that aims at combining the detections from different features should consider feature specific properties such as the false positive rate and adjust the ranking criterion to achieve a high detection rate.

Because, identifying the abnormal region in cryptogenic epilepsy is a multifaceted procedure that is based on a confluence of evidence from multiple sources; the high detection rate of our proposed method will have a deeper impact in the application domain by enhancing the sensitivity of the patient evaluation methodology. Indeed, our 75% detection rate on the MRI-Negative patients in our evauation dataset (compared to a human expert detection rate of 0%), suggests that this method can be used as an effective tool in the presurgical evaluation of TRE patients who are likely to undergo surgical resection. Currently,

HCRF results have started being incorporated into the weekly meeting of radiologists and neurosurgeons to help identify the seizure onset zones for MRI-negative patients who may be candidates for resective surgery at the New York Universityś Comprehensive Epilepsy Center.

As part of the pre-surgical protocol, all patients undergo an intra-cranial EEG (iEEG) exam in which invasive subdural electrodes are placed directly on the cortex to record electrical activity. As a future research direction, we plan to use the results of the iEEG exam to augment the resection zones such that they provide "soft" labels that can be used to jointly learn the parameters of the node and edge potentials in the HCRF. Another avenue of future research is to further expand the feature set to include other features such as diffusivity, connectivity, etc.

Acknowledgments

We thank the reviewers for their helpful comments. We would also like to thank John Brumm, Omar Khan, Hugh Wang and Christine Chao for help with data processing. This research was funded by the Epilepsy Foundation and FACES - Finding A Cure for Epilepsy and Seizures.

References

B. Ahmed, C. E. Brodley, K. E. Blackmon, R. Kuzniecky, G. Barash, C. Carlson, B. T. Quinn, W. Doyle, J. French, O. Devinsky, and T. Thesen. Cortical feature analysis and machine learning improves detection of mri-negative focal cortical dysplasia. *Epilepsy & Behavior*, 48:21 – 28, 2015.

Bilal Ahmed, Thomas Thesen, Karen Blackmon, Yijun Zhao, Orrin Devinsky, Ruben Kuzniecky, and Carla Brodley. Hierarchical conditional random fields for outlier detection: An application to detecting epileptogenic cortical malformations. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1080–1088, 2014.

L.C. Andrews. *Special Functions of Mathematics for Engineers*. SPIE Optical Engineering Press, 1992. ISBN 9780819426161.

P. Awasthi, A. Gagrani, and B. Ravindran. Image modelling using tree structured conditional random fields. In *IJCAI*, pages 2060–2065, 2007.

M.L. Bell, S. Rao, E.L. So, et al. Epilepsy surgery outcomes in temporal lobe epilepsy with a normal MRI. *Epilepsia*, 50(9):2053–2060, 2009.

S. R. Benbadis, L. Heriaud, W. O. Tatum IV, and F. L. Vale. Epilepsy surgery, delays and referral patternsare all your epilepsy patients controlled? *Seizure*, 12(3):167 – 170, 2003.

A. Bernasconi, N. Bernasconi, B.C. Bernhardt, and D. Schrader. Advances in mri for 'cryptogenic' epilepsies. *Nat Rev Neurol.*, 7(2):99–108, 2011.

P. Besson, N. Bernasconi, O. Colliot, et al. Surface-based texture and morphological analysis detects subtle cortical dysplasia. In *MICCAI*, pages 645–652, 2008.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006. ISBN 0387310738.

I. Blumcke, M. Thom, E. Aronica, et al. The clinicopathologic spectrum of focal cortical dysplasias: A consensus classification proposed by an ad hoc task force of the ILAE diagnostic methods commission. *Epilepsia*, 52(1):158–174, 2011.

M. Breunig, Hans-Peter Kriegel, R.T. Ng, and J. Sander. LOF: Identifying Density-Based Local Outliers. In *ACM SIGMOD ICMD*, pages 93–104. ACM, 2000.

D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002.

A.M. Dale, B. Fischl, and M.I. Sereno. Cortical surface-based analysis: I. segmentation and surface reconstruction. *NeuroImage*, 9(2):179–194, 1999.

R.S. Desikan, F. Sgonne, B. Fischl, et al. An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest. *NeuroImage*, 31(3):968–980, 2006.

B. Fischl and A. M. Dale. Measuring the thickness of the human cerebral cortex from magnetic resonance images. *Proceedings of the National Academy of Sciences*, 97(20): 11050–11055, 2000.

B. Fischl, M.I. Sereno, and A.M. Dale. Cortical surface-based analysis: II: Inflation, flattening, and a surface-based coordinate system. *NeuroImage*, 9(2):195–207, 1999a.

B. Fischl, D.H. Salat, E. Busa, et al. Whole brain segmentation: Automated labeling of neuroanatomical structures in the human brain. *Neuron*, 33(3):341–355, 2002.

Bruce Fischl, Martin I. Sereno, Roger B.H. Tootell, and Anders M. Dale. High-resolution intersubject averaging and a coordinate system for the cortical surface. *Human Brain Mapping*, 8(4):272–284, 1999b.

W. A. Hauser and D. C. Hesdorffer. *Epilepsy: frequency, causes and consequences*. Epilepsy Foundation of America, 1990.

S. J. Hong, H. Kim, D. Schrader, N. Bernasconi, B. C. Bernhardt, and A. Bernasconi. Automated detection of cortical dysplasia type II in MRI-negative epilepsy. *Neurology*, 83(1):48–55, 2014.

Nathalie Japkowicz and Mohak Shah. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, New York, NY, USA, 2011. ISBN 0521196000, 9780521196000.

Hans-Peter Kriegel, P. Kröger, E. Schubert, and A. Zimek. LoOP: Local Outlier Probabilities. In *ACM CIKM*, pages 1649–1652, 2009.

Ruben I. Kuzniecky and A.James Barkovich. Malformations of cortical development and epilepsy. *Brain and Development*, 23(1):2 – 11, 2001.

P. Kwan and M. J. Brodie. Early identification of refractory epilepsy. *New England Journal Of Medicine*, 342(5):314–319, 2000.

J. T. Lerner et al. Assessment and surgical outcomes for mild type I and severe type II cortical dysplasia: a critical review and the UCLA experience. *Epilepsia*, 50(6):1310–1335, 2009.

J.J. Lin, N. Salamon, A.D. Lee, et al. Reduced neocortical thickness and complexity mapped in mesial temporal lobe epilepsy with hippocampal sclerosis. *Cereb. Cortex*, 17(9):2007–2018, 2007.

D.G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

C.R. McDonald, D. J. H. Jr, M. E. Ahmadi, et al. Regional neocortical thinning in mesial temporal lobe epilepsy. *Epilepsia*, 49(5):794–803, 2008.

C.W. Nordahl, D. Dierker, I. Mostafavi, et al. Cortical folding abnormalities in autism revealed by surface-based morphometry. *J Neurosci.*, 27(43):11725–11735, 2007.

S. Papadimitriou, H. Kitagawa, P.B. Gibbons, and C. Faloutsos. LOCI: fast outlier detection using the local correlation integral. In *ICDE*, pages 315–326, 2003.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988. ISBN 1558604790.

R. Pienaar, B. Fischl, V. Caviness, N. Makris, and P. E. Grant. A methodology for analyzing curvature in the developing brain from preterm to adult. *International Journal of Imaging Systems and Technology*, 18(1):42–68, 2008.

N. Plath, M. Toussaint, and S. Nakajima. Multi-class image segmentation using conditional random fields and global classification. In *ICML*, pages 817–824, 2009.

J. Reynolds and K. Murphy. Figure-ground segmentation using a hierarchical conditional random field. In *CRV*, pages 175–182, 2007.

L.M. Rimol, R. Nesvg, D.J. Hagler Jr., et al. Cortical volume, surface area, and thickness in schizophrenia and bipolar disorder. *Biological Psychiatry*, 71(6):552–560, 2012.

D. H. Salat, R. L. Buckner, A.Z. Snyder, et al. Thinning of the cerebral cortex in aging. *Cerebral Cortex*, 14(7):721–730, 2004.

E. Schubert, R. Wojdanowski, A. Zimek, and Hans-Peter Kriegel. On evaluation of outlier rankings and outlier scores. In *SDM*, pages 1047–1058, 2012.

C. Sutton and A. McCallum. An Introduction to Conditional Random Fields, 2010. eprint arXiv:1011.4088.

T. Thesen, B.T. Quinn, C. Carlson, et al. Detection of epileptogenic cortical malformations with surface-based MRI morphometry. *PLoS ONE*, 6(2):e16430, 2011.

J. F. Tllez-Zenteno, R. Dhar, and S. Wiebe. Long-term seizure outcomes following epilepsy surgery: a systematic review and meta-analysis. *Brain*, 128(5):1188–1198, 2005.

D. C. Van Essen, H. A. Drury, S. Joshi, and M. I. Miller. Functional and structural mapping of human cerebral cortex: Solutions are in the surfaces. *Proceedings of the National Academy of Sciences*, 95(3):788–795, 1998.

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008a.

A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *ECCV*, pages 705–718, 2008b.

Z. I. Wang, A. V. Alexopoulos, S. E. Jones, Z. Jaisani, I. M. Najm, and R. A. Prayson. The pathology of magnetic-resonance-imaging-negative epilepsy. *Mod Pathol*, 26(8):1051–1058, 2013.

Journal of Machine Learning Research 17 (2016) 1-23 Submitted 11/15; Revised 6/16; Published 7/16

Fused Lasso Approach in Regression Coefficients Clustering – Learning Parameter Heterogeneity in Data Integration

Lu Tang LUTANG@UMICH.EDU
Peter X.K. Song PXSONG@UMICH.EDU
Department of Biostatistics
University of Michigan
Ann Arbor, MI 48109, USA

Editor: Hui Zou

Abstract

As data sets of related studies become more easily accessible, combining data sets of similar studies is often undertaken in practice to achieve a larger sample size and higher power. A major challenge arising from data integration pertains to data heterogeneity in terms of study population, study design, or study coordination. Ignoring such heterogeneity in data analysis may result in biased estimation and misleading inference. Traditional techniques of remedy to data heterogeneity include the use of interactions and random effects, which are inferior to achieving desirable statistical power or providing a meaningful interpretation, especially when a large number of smaller data sets are combined. In this paper, we propose a regularized fusion method that allows us to identify and merge inter-study homogeneous parameter clusters in regression analysis, without the use of hypothesis testing approach. Using the fused lasso, we establish a computationally efficient procedure to deal with large-scale integrated data. Incorporating the estimated parameter ordering in the fused lasso facilitates computing speed with no loss of statistical power. We conduct extensive simulation studies and provide an application example to demonstrate the performance of the new method with a comparison to the conventional methods.

Keywords: Fused lasso, Data integration, Extended BIC, Generalized Linear Models

1. Introduction

Combining data sets collected from multiple studies is undertaken routinely in practice to achieve a larger sample size and higher statistical power. Such information integration is commonly seen in biomedical research, for example, the study of genetics or rare diseases where data repositories are available. The motivation of this paper arises from the consideration of data heterogeneity during data integration. Although data integration has different meanings, in here, we consider the concatenation of data sets of similar studies over different subjects, where the number of integrated data sets can be very large.

Inter-study heterogeneity can result from the differences in study environment, population, design and protocols (Leek and Storey, 2007; Sutton and Higgins, 2008; Liu et al., 2015). Data heterogeneity is likely attributed to population parameter heterogeneity, where the association of interest can differ across different study populations from which data sets are collected. Examples include multi-center clinical trials when participant data from different sites are combined (Shekelle et al., 2003) and genetics studies when genomic data

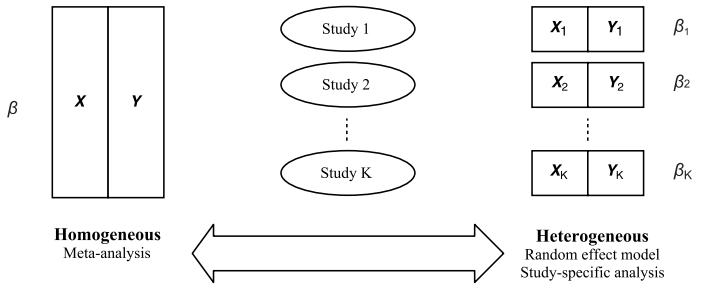


Figure 1: Homogeneous assumption (left) versus heterogeneous assumption (right).

from multiple similar studies are combined (Lohmueller et al., 2003; Sullivan et al., 2000). Discrepancies in treatment effect or trait-gene association may arise due to the differences in facilities, practices and patient characteristics across studies, albeit the adjustment of confounding (Leek and Storey, 2007). The parameter heterogeneity introduced in data integration compromises the power of the larger sample size and may even lead to biased results and misleading scientific conclusions. Thus, counterintuitively, the model obtained from the combined studies may not serve as a proper prediction model for each individual study in the case of heterogeneous study populations.

Traditional treatments of parameter heterogeneity are not optimal. Meta-analysis methods such as combining summary statistics (Glass, 1976), estimating functions (Hansen, 1982; Qin and Lawless, 1994) or p-values functions (Xie et al., 2012) are built upon the assumption of complete parameter homogeneity, as shown in the left panel of Figure 1. This assumption is hardly valid in practice. When individual participant data from multiple data sets are available, a retreat to the classical meta-analysis methods is necessary, because in this case assessing the assumption of inter-study homogeneity becomes possible. The two most common approaches to handling parameter heterogeneity include (i) specifying study-specific effects by including interaction terms between study indicator and covariates (e.g., Lin et al. (1998)), and (ii) utilizing random covariate effects by allowing variations across studies as random variables (e.g., DerSimonian and Kacker (2007)). Both approaches essentially assume fully heterogeneous covariate effects, namely, each study having its own set of regression coefficients, as shown in the right panel of Figure 1.

When study-specific effects are of interest, the interaction-based formulation may lead to over-parameterization, which impairs statistical power. The most straightforward way to reduce the number of parameters is to identify clusters of homogeneous parameters through exhaustive tests for the differences between every pair of study-specific coefficients. However, when the number of data sets is large, the use of hypothesis testing to determine parameter clusters becomes untrackable in addition to the multiple-testing problem. One may draw different or even conflicting conclusions due to different orders of hypotheses performed.

In reality, covariate effects from multiple studies are likely to form groups, a scenario falling in between the complete heterogeneity and the complete homogeneity. This leads

to the following two essential yet related analytic tasks: (i) to assess the inter-study heterogeneity, so to determine an appropriate form of parsimonious parameterization in model specification; and (ii) to identify and merge groups of homogeneous parameters for better statistical power for parameter estimation and inference based on a more parsimonious model. Along the idea of lasso shrinkage estimator (Tibshirani, 1996), fused lasso methods (Tibshirani et al., 2005; Friedman et al., 2007; Yang et al., 2012) have been introduced to achieve covariate grouping, where covariate adjacencies are naturally defined by a metric of time, location or network structure. In our problem of data integration, there does not exist a natural metric to define the ordering of regression coefficients from different studies. Shen and Huang (2010) proposed the grouping pursuit via penalization of all pairwise coefficient differences in a single study, where covariate orderings are not considered. To reduce the computational burden in the all-pairs based regularization, Wang et al. (2016) and Ke et al. (2015) used the initial coefficient estimates to establish certain ordering and then to define parameter adjacencies. However, most of these studies have been entirely focusing on a single cohort of subjects from a single study. For example, Shin et al. (2016) proposed to fuse regression coefficients of different loss functions obtained from a single study, such as coefficients from different quantile regression models. Limited publication of fusion learning and grouping pursuit has been available in the literature, except Wang et al. (2016), to assess the differences and similarities among regression coefficients across multiple studies in the scenario of data integration.

In this paper, we propose an agglomerative clustering method for regression coefficients in the context of data integration, named as the *Fused Lasso Approach in Regression Coefficients Clustering (FLARCC)*. FLARCC is proposed to identify heterogeneity patterns of regression coefficients across studies (or data sets) and to provide estimates of all regression coefficients simultaneously. It is interesting to draw a connection between our method and Pan et al. (2013) where they consider a classic clustering problem of individual responses by pairwise coefficient fusion via penalized regression. Their method aims at clustering subjects, while our method focuses on clustering regression coefficients across multiple data sets, and these two methods coincide only in a special case where each study is composed of only one subject. FLARCC achieves clustering of study-specific effects by penalizing the ℓ_1-norm differences of adjacent coefficients, with adjacency defined by the estimated ranks. Our method extends the bCARDS method in Ke et al. (2015) from one study to multiple studies as well as from the linear model to the generalized linear models, and focuses on simultaneous clustering of regression coefficients of individual covariates from multiple studies in data integration. An R package `metafuse` is created as part of our methodology development to perform the proposed integrated data analysis which can be downloaded from the Comprehensive R Archive Network (web link `https://cran.r-project.org/web/packages/metafuse`).

In the proposed method, tuning parameter is used to determine the clustering pattern of coefficients across data sets. Specifically, let λ be the tuning parameter of regularization. If $\lambda = 0$ (i.e., no penalty), FLARCC becomes a method under the setting of complete heterogeneity, so that study-specific regression coefficients for each covariate are assumed different across data sets. If λ is large enough that all differences of regression coefficients are shrunk to zero, FLARCC reduces to a homogeneous model in that a common regression coefficient for each covariate is assumed for all studies. In light of the hierarchical clustering

scheme, these two extreme cases above correspond to the start and end of an agglomerative clustering, respectively; however, the reality is believed to reside in between. Analogous to dendrograms in the hierarchical clustering, we propose a new tree-type graphic display, named as *fusogram*, which presents tree-based coefficient clusters according to solution paths obtained from FLARCC. The selection of optimal λ pertains to pruning of clustering trees, which can be based on certain model selection criterion. We use the extended Bayesian information criterion (EBIC) proposed by Chen and Chen (2008) as our model selection criterion and show that EBIC exhibits better performance than BIC when the number of studies (or data sets) is large. In addition, we propose a scaling strategy to "harmonize" solution paths by covariate-wise adaptive weights to allow flexible tuning, which further improves the clustering performance.

The rest of this paper is organized as follows. Section 2 describes FLARCC in detail under the generalized linear models (GLM) framework. Section 3 presents the theoretical properties of the proposed method (with technical proofs presented in the Appendix). Section 4 discusses the interpretation and selection of the tuning parameter. In Section 5, we use simulation studies to evaluate the performance of our method. A real data analysis is given in Section 6 with interpretation of coefficient estimates and illustration of *fusograms*. Discussion and concluding remarks are in Section 7.

2. Method of Parameter Fusion

In this section, we present the method and algorithm of FLARCC.

2.1 Notations and Method

We start by introducing necessary notations. Throughout this paper, i, j and k are used to index subject, covariate and study, respectively. For instance, $X_{j,k}^{(i)}$ denotes the measurement of the jth covariate from the ith individual from study k, and $Y_k^{(i)}$ is the measurement of a response variable from the ith individual from study k. The total number of studies is denoted as K and the number of covariates involved is p. The sample size for study k is n_k, $k = 1, \ldots, K$, and the combined sample size is $N = \sum_{k=1}^{K} n_k$. The collection of all coefficients (covariates-wise) is denoted as $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1,\bullet}^\top, \boldsymbol{\beta}_{2,\bullet}^\top, \ldots, \boldsymbol{\beta}_{p,\bullet}^\top)^\top$ with $\boldsymbol{\beta}_{j,\bullet} = (\beta_{j,1}, \ldots, \beta_{j,K})^\top$ for $j = 1, \ldots, p$. An indicator vector $\boldsymbol{c} = (c_1, \ldots, c_p)^\top$ is used to flag heterogeneous covariates, namely if the jth covariate is treated as heterogeneous (i.e., all different coefficients across K studies) then $c_j = 1$ and as homogeneous (a common coefficient across K studies) otherwise. Thus $c_j = 0$ for some $j \in \{1, \ldots, p\}$ implies that coefficient vector $\boldsymbol{\beta}_{j,\bullet}$ reduces to a common scalar parameter β_j for all K studies.

For illustration, let us consider a simple scenario of $\boldsymbol{c} = (1, 1, 0, \ldots, 0)^\top$, in which the first two covariates are set as heterogeneous and the remaining $p-2$ covariates are set as homogeneous. The resulting coefficient vector is $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1,\bullet}^\top, \boldsymbol{\beta}_{2,\bullet}^\top, \beta_3, \ldots, \beta_p)^\top$. Then the corresponding design matrix $\boldsymbol{X}$ can be written as

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{1,1} & & & \boldsymbol{X}_{2,1} & & & \boldsymbol{X}_{3,1} & \cdots & \boldsymbol{X}_{p,1} \\ & \ddots & & & \ddots & & \vdots & \ddots & \vdots \\ & & \boldsymbol{X}_{1,K} & & & \boldsymbol{X}_{2,K} & \boldsymbol{X}_{3,K} & \cdots & \boldsymbol{X}_{p,K} \end{pmatrix}_{N \times (2K+p-2)}$$

where $\boldsymbol{X}_{j,k} = (X_{j,k}^{(1)}, \ldots, X_{j,k}^{(n)})^\top$, $j = 1, \ldots, p$, $k = 1, \ldots, K$. The specification of $\boldsymbol{c}$ is can be dependent on the study interest. For example, in a multi-center clinical trial where we believe that the differences between the services provided across centers are non-negligible, but the study participants are similar, we can specify the clinic-related variables (e.g., treatment and cost) to be heterogeneous and the patient-related variables (e.g., age and gender) to be homogeneous. In addition, the specification of $\boldsymbol{c}$ can be dependent on preliminary marginal analysis of the homogeneousness of each variable, such as tests for random effects. When the homogeneousness of a covariate is unclear, we suggest specifying it as heterogeneous rather than homogeneous.

Under the assumption that both within-study and between-study samples are independent, for any $\boldsymbol{c} = (c_1, \ldots, c_p)^\top$ with $c_j \in \{0, 1\}$, $j = 1, \ldots, p$, the initial estimate of $\boldsymbol{\beta}$, which gives the starting level of clustering (i.e., $\lambda = 0$), can be consistently estimated by the maximum likelihood estimator

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{(K \times p)}}{\operatorname{argmax}} \frac{1}{K} \sum_{k=1}^{K} \frac{1}{n_k} \log L_k(\boldsymbol{\beta}), \tag{1}$$

where $L_k(\boldsymbol{\beta}) = \prod_{i=1}^{n_k} L_k^{(i)}(\boldsymbol{\beta})$, $k = 1, \ldots, K$ are the study-specific likelihoods from the given GLMs. For the purpose of parameter grouping and fusion, we propose the regularized maximum likelihood estimation for $\boldsymbol{\beta}$ by minimizing the following objective function:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^{(K \times p)}} \left(-\frac{1}{K} \sum_{k=1}^{K} \frac{1}{n_k} \log L_k(\boldsymbol{\beta}) + P(\boldsymbol{\beta}) \right), \tag{2}$$

where $P(\boldsymbol{\beta})$ is a penalty function of certain form. Here we adopt weighting $\frac{1}{n_k}$ to balance the contribution from each study so to avoid the dominance of large studies. Other types of weighting schemes may be considered to serve for different purposes, such as the inverse of estimated variances of initial estimates, which helps to achieve better estimation precision.

To achieve parameter fusion, Shen and Huang (2010) proposed the grouping pursuit algorithm, which specifies the sum of ℓ_1-norm differences of all study-specific coefficient pairs among individual heterogeneous coefficient vectors $\boldsymbol{\beta}_{j,\cdot}$, where $c_j = 1$, as the penalty:

$$P_\lambda(\boldsymbol{\beta}) = \lambda \sum_{j=1}^{p} c_j \sum_{k=1}^{K-1} \sum_{k'>k}^{K} |\beta_{j,k} - \beta_{j,k'}|,$$

with $\lambda \geq 0$. In this penalty, there are $\binom{K}{2}$ terms of pairwise differences for each heterogeneous covariate and the total number of terms increases by an order of $O(K^2)$, given p fixed. This penalty contains many redundant constraints and imposes great computational challenges as pointed out in Shen and Huang (2010) and Ke et al. (2015).

Following arguments in Wang et al. (2016) and Ke et al. (2015), we develop the method of FLARCC by a simplified penalty function that uses the information on the ordering of coefficients. For the jth covariate, let $\boldsymbol{U}_j = (U_{j,1}, \ldots, U_{j,K})^\top$ be the ranking with no ties of $\boldsymbol{\beta}_{j,\cdot} = (\beta_{j,1}, \ldots, \beta_{j,K})^\top$, from the smallest to the largest. Specifically, $U_{j,k} = \sum_{k'=1}^{K} \mathbf{1}\{\beta_{j,k'} \leq \beta_{j,k}\}$ if there are no ties in $\boldsymbol{\beta}_{j,\cdot}$; otherwise, the ties in $\boldsymbol{U}_j$ are resolved by the first-occurrence-wins

rule according to k to ensure rank uniqueness. Then, the fusion penalty in FLARCC with parameter orderings $\boldsymbol{U}_j$, $j = 1, \ldots, p$, takes the form:

$$P_\lambda(\boldsymbol{\beta}) = \lambda \sum_{j=1}^{p} c_j \nu_j \sum_{k=1}^{K-1} \sum_{k'>k}^{K} \mu_{j,k,k'} \mathbf{1}\{|U_{j,k} - U_{j,k'}| = 1\} |\beta_{j,k} - \beta_{j,k'}|, \tag{3}$$

where the constraints occur effectively only on adjacent ordered pairs. Clearly, the penalty in (3) only involves $K - 1$ terms for each case of $c_j = 1$, which is of an order $O(K)$, given p fixed. The ν_j's and $\mu_{j,k,k'}$'s in (3) are weights. Following Zou (2006), we choose adaptive weights $\hat{\mu}_{j,k,k'} = 1/|\hat{\beta}_{j,k} - \hat{\beta}_{j,k'}|^r$, $r > 0$, so that parameters with smaller difference will be penalized more than those with larger differences. Similarly, for a group of parameters $\boldsymbol{\beta}_{j,\cdot} = (\beta_{j,1}, \ldots, \beta_{j,K})^\top$, ν_j is an adaptive weight to characterize the degree of heterogeneousness of $\boldsymbol{\beta}_{j,\cdot}$. Specifically, in this paper we let $\hat{\nu}_j = 1/|\hat{\beta}_{j,(K)} - \hat{\beta}_{j,(1)}|^s$, the inverse of the range of the estimates, with $s \geq 0$; when a covariate is homogeneous, the differences of study-specific coefficients will be penalized more than those that are heterogeneous. In this way, we can "harmonize" solution paths so to greatly improve the performance by a single tuning parameter. We compare $s = 0$ and $s = 1$ in the simulation experiments and show in Section 5 that the introduction of such group-wise weights ν_j, $j = 1, \ldots, p$, gives rise to improvement on the performance of identifying homogeneous covariates when K and p are large.

A sparse version of FLARCC can also be achieved by including the traditional lasso penalty in (3) for covariate selection. In order to minimize the interference between fusion and sparsity penalties, we only encourage sparsity for the coefficient closest to zero in each $\boldsymbol{\beta}_{j,\cdot} = (\beta_{j,1}, \ldots, \beta_{j,K})^\top$, for $j = 1, \ldots, p$. Similar to the definition of $\boldsymbol{U}_j$, let $\boldsymbol{V}_j = (V_{j,1}, \ldots, V_{j,K})^\top$ be the ranking with no ties, from the smallest to the largest, of the absolute values of $\boldsymbol{\beta}_{j,\cdot}$, i.e., $(|\beta_{j,1}|, \ldots, |\beta_{j,K}|)^\top$. First we calculate $\boldsymbol{V}_j$ by $V_{j,k} = \sum_{k'=1}^{K} \mathbf{1}\{|\beta_{j,k'}| \leq |\beta_{j,k}|\}$, then we resolve the ties in $\boldsymbol{V}_j$ by the first-occurrence-wins rule according to k. Thus we can extend (3) to achieve variable selection by the following penalty function:

$$\begin{aligned} P_{\lambda,\alpha}(\boldsymbol{\beta}) = \lambda \sum_{j=1}^{p} c_j \nu_j \sum_{k=1}^{K-1} \sum_{k'>k}^{K} \mu_{j,k,k'} \mathbf{1}\{|U_{j,k} - U_{j,k'}| = 1\} |\beta_{j,k} - \beta_{j,k'}| \\ + \alpha\lambda \sum_{j=1}^{p} \sum_{k=1}^{K} \mu_{j,k} \mathbf{1}\{V_{j,k} = 1\} |\beta_{j,k}|, \end{aligned} \tag{4}$$

where $\alpha \geq 0$ is another tuning parameter that controls the relative ratio between fusion and sparsity penalties, and $\hat{\mu}_{j,k} = 1/|\hat{\beta}_{j,k}|^r$. The sparsity penalty, although only enforced on the smallest coefficient in absolute value of $\boldsymbol{\beta}_{j,\cdot}$, is capable of shrinking a group of coefficients to zero when combined with the fusion penalty.

In practice, the weights (ν_j, $\mu_{j,k,k'}$ and $\mu_{j,k}$) and the parameter orderings ($\boldsymbol{U}_j$ and $\boldsymbol{V}_j$) are unknown, for $j = 1, \ldots, p$. We replace them with their estimates based on root-n consistent estimates $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_{1,\cdot}^\top, \ldots, \hat{\boldsymbol{\beta}}_{p,\cdot}^\top)^\top$, such as those from (1). In the simulation experiments and the real data application of this paper, we set $r = 1$ in $\hat{\mu}_{j,k,k'}$ and $\hat{\mu}_{j,k}$.

2.2 Algorithm

Optimization problem (2) with $P(\boldsymbol{\beta}) = P_{\lambda,\alpha}(\boldsymbol{\beta})$ given in (4) can be carried out by a lasso regression through suitable reparameterization. Let the ordered coefficients of $\boldsymbol{\beta}_{j,\bullet}$ in an ascending order based on ranking $\boldsymbol{U}_j$ be $(\beta_{j,(1)}, \ldots, \beta_{j,(K)})^\top$, $j = 1, \ldots, p$. For the jth covariate, consider a set of transformed parameters $\boldsymbol{\theta}_{j,\bullet} = (\theta_{j,1}, \ldots, \theta_{j,K})^\top$ defined by

$$\begin{aligned} \theta_{j,1} &= \beta_{j,k}, && \text{for } k \text{ s.t. } V_{j,k} = 1; \\ \theta_{j,k} &= \beta_{j,(k)} - \beta_{j,(k-1)}, && \text{for } k = 2, \ldots, K. \end{aligned}$$

Then the $P_{\lambda,\alpha}(\boldsymbol{\beta})$ in (4) can be rewritten as

$$P_{\lambda,\alpha}(\boldsymbol{\theta}) = \lambda \sum_{j=1}^{p} \sum_{k=1}^{K} \omega_{j,k} |\theta_{j,k}|, \tag{5}$$

where

$$\hat{\omega}_{j,k} = \begin{cases} \alpha \frac{1}{|\hat{\theta}_{j,1}|^r}, & \text{if } k = 1 \\ c_j \frac{1}{|\sum_{k'=2}^{K} \hat{\theta}_{j,k'}|^s} \frac{1}{|\hat{\theta}_{j,k}|^r}, & \text{if } k = 2, \ldots, K, \end{cases} \tag{6}$$

for $j = 1, \ldots, p$. Since no ties are allowed in the parameter ordering of FLARCC, one-to-one transformation exists between $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1,\bullet}^\top, \boldsymbol{\beta}_{2,\bullet}^\top, \ldots, \boldsymbol{\beta}_{p,\bullet}^\top)^\top$ and $\boldsymbol{\theta} = (\boldsymbol{\theta}_{1,\bullet}^\top, \boldsymbol{\theta}_{2,\bullet}^\top, \ldots, \boldsymbol{\theta}_{p,\bullet}^\top)^\top$ by suitable sorting matrix $\boldsymbol{S}$ and reparameterization matrix $\boldsymbol{R}$; that is, $\boldsymbol{\theta} = \boldsymbol{RS\beta}$ and $\boldsymbol{\beta} = (\boldsymbol{RS})^{-1}\boldsymbol{\theta}$ with both $\boldsymbol{S}$ and $\boldsymbol{R}$ being full-rank square matrices. Thus, a solution to the fused lasso problem can be obtained equivalently by solving a routine lasso problem with respect to coefficient vector $\boldsymbol{\theta}$ and a transformed design matrix $\boldsymbol{X}(\boldsymbol{RS})^{-1}$. As aforementioned, the estimated parameter ordering is used to construct $\boldsymbol{S}$. It is obvious that the constraint in (5) is convex, thus FLARCC does not suffer from multiple local minimal issue. The optimization is done using R package `glmnet` (version 2.0-2) (Friedman et al., 2010), which accommodates GLMs with Gaussian, binomial and Poisson distributions.

3. Large-sample Properties

First we present the oracle property of our method when the parameter ordering is known, then we prove that the same large-sample properties are preserved when consistently estimated parameter ordering is used. Here we assume K is fixed. Theorems will be stated under the setting of all coefficients being heterogeneous, i.e., $\boldsymbol{c} = (1, \ldots, 1)^\top$. The large-sample theories for other specification of $\boldsymbol{c}$ can be established as a special case.

Denote the true parameter values as $\boldsymbol{\beta}^*$ and $\boldsymbol{\theta}^*$. Let the collection of true parameter orderings of all covariates and their absolute values be $\boldsymbol{W} = \{\boldsymbol{U}_j, \boldsymbol{V}_j\}_{j=1}^p$, and the estimated orderings based on the root-n consistent estimator $\hat{\boldsymbol{\beta}}$ from (1) as $\hat{\boldsymbol{W}} = \{\hat{\boldsymbol{U}}_j, \hat{\boldsymbol{V}}_j\}_{j=1}^p$. Denote the FLARCC estimator of $\boldsymbol{\theta}^*$ as $\hat{\boldsymbol{\theta}}^{\boldsymbol{W}}$ when $\boldsymbol{W}$ is known, and $\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}}$ when the estimated parameter ordering $\hat{\boldsymbol{W}}$ is used. Let $\mathcal{A} = \bigcup_{j=1}^p \{\mathcal{A}_j\}$ be the index set of nonzero values in $\boldsymbol{\theta}^*$, where $\mathcal{A}_j = \{(j,k) : \theta_{j,k}^* \neq 0\}$, and $\mathcal{A}^c$ be the complement of $\mathcal{A}$. Thus, $\boldsymbol{\theta}^*$ can be partitioned into two subsets, the true-zero set $\boldsymbol{\theta}_{\mathcal{A}^c}^*$ and the nonzero set $\boldsymbol{\theta}_{\mathcal{A}}^*$. Similarly, let $\hat{\mathcal{A}}^{\boldsymbol{W}}$ and $\hat{\mathcal{A}}^{\hat{\boldsymbol{W}}}$ be the index sets of nonzero elements in $\hat{\boldsymbol{\theta}}^{\boldsymbol{W}}$ and $\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}}$, respectively. Let $n = \min_{1 \le k \le K} n_k$, $N = \sum_{k=1}^K n_k$, and $\lambda_N = N\lambda$.

Theorem 1 *Suppose that tuning parameter λ_N satisfies $\lambda_N/\sqrt{N} \to 0$ and $\lambda_N N^{(r-1)/2} \to \infty$. Then under some mild regularity conditions (see Appendix A), the FLARCC estimator $\hat{\boldsymbol{\theta}}^{\boldsymbol{W}}$ based on the true parameter ordering $\boldsymbol{W}$ satisfies*

(i) (Selection Consistency) $\lim_n P(\hat{\mathcal{A}}^{\boldsymbol{W}} = \mathcal{A}) = 1$;

(ii) (Asymptotic Normality) $\sqrt{N}(\hat{\boldsymbol{\theta}}_{\mathcal{A}}^{\boldsymbol{W}} - \boldsymbol{\theta}_{\mathcal{A}}^*) \xrightarrow{d} \mathcal{N}(0, \boldsymbol{I}_{11}^{-1})$ *as $n \to \infty$, where $\boldsymbol{I}_{11}$ is the submatrix of Fisher information matrix $\boldsymbol{I}$ corresponding to set $\mathcal{A}$.*

Theorem 1 states that when the coefficient orderings $\boldsymbol{W}$ of $\boldsymbol{\beta}$ is known, under mild regularity conditions, the FLARCC estimator $\hat{\boldsymbol{\theta}}^{\boldsymbol{W}}$ enjoys selection consistency and asymptotic normality. The proof of Theorem 1 follows Zou (2006) and is given in Appendix A. Now we present Theorem 3, which states that the same properties of Theorem 1 hold for $\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}}$, the FLARCC estimator of $\boldsymbol{\theta}^*$ based on the estimated parameter ordering $\hat{\boldsymbol{W}}$. In effect, Theorem 3 is a consequence of the following lemma.

Lemma 2 *If $\hat{\boldsymbol{\beta}}$ is a root-n consistent estimator of $\boldsymbol{\beta}$, then* $\lim_n P(\hat{\boldsymbol{U}}_j = \boldsymbol{U}_j) = 1$ *and* $\lim_n P(\hat{\boldsymbol{V}}_j = \boldsymbol{V}_j) = 1$ *for $j = 1, \ldots, p$.*

The proof of Lemma 2 is given in Appendix A. Lemma 2 implies that the parameter ordering can be consistently estimated. Using Lemma 2, we are able to extend the properties of $\hat{\boldsymbol{\theta}}^{\boldsymbol{W}}$ in Theorem 1 to the proposed FLARCC estimator $\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}}$.

Theorem 3 *Suppose that $\lambda_N/\sqrt{N} \to 0$ and $\lambda_N N^{(r-1)/2} \to \infty$. Let the estimated parameter ordering $\hat{\boldsymbol{W}}$ be the ranks from a root-n initial consistent estimator $\hat{\boldsymbol{\beta}}$. Under the same regularity conditions of Theorem 1, the FLARCC estimator $\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}}$ satisfies*

(i) (Selection Consistency) $\lim_n P(\hat{\mathcal{A}}^{\hat{\boldsymbol{W}}} = \mathcal{A}) = 1$;

(ii) (Asymptotic Normality) $\sqrt{N}(\hat{\boldsymbol{\theta}}_{\mathcal{A}}^{\hat{\boldsymbol{W}}} - \boldsymbol{\theta}_{\mathcal{A}}^*) \xrightarrow{d} \mathcal{N}(0, \boldsymbol{I}_{11}^{-1})$ *as $n \to \infty$, where $\boldsymbol{I}_{11}$ is the submatrix of Fisher information matrix $\boldsymbol{I}$ corresponding to set $\mathcal{A}$.*

The proof of Theorem 3 is given in Appendix A. The asymptotic normality for $\hat{\boldsymbol{\beta}}$ can also be derived by a simple linear transformation.

4. Tuning Parameter

In this section, we provide interpretation of the tuning parameter λ and discuss the selection criteria used for selecting λ.

4.1 Interpretation of ν_j's

Intuitively speaking, the study-specific coefficients of a homogeneous covariate tend to be fused at a small λ value, say λ_1, but the fusion of a heterogeneous covariate requires another λ value, λ_2, assuming $\lambda_2 > \lambda_1$. The region to draw correct clustering conclusion is $[\lambda_1, \lambda_2]$, that is, any λ within this region will produce the correct clustering result. However, when the number of covariates p is large, the region that λ can take value from to ensure the correct clustering of all p coefficient vectors simultaneously becomes narrower and may even

be empty. For example, when $\lambda_2 < \lambda_1$ in the above case, no single λ is able to correctly cluster both sets of parameters. The introduction of ν_j's in (4) creates larger separation between homogeneous and heterogeneous groups, so that the range for λ to identify the correct clustering pattern for all covariates is better established than the case with $s = 0$, namely no use of weighting ν_j's. When the number of covariates p is large, ν_j plays a more important role in harmonizing solution paths across covariates, and the performance will be greatly improved by simultaneous tuning via a single λ.

4.2 Model Selection

In the current literature, the tuning parameter λ may be selected by multiple model selection criteria, such as Bayesian information criterion (BIC) (Schwarz, 1978) and generalized cross-validation (GCV) (Golub et al., 1979). In this paper, we consider the widely used BIC and its modification, extended BIC, i.e., EBIC (Chen and Chen, 2008; Gao and Song, 2010), which has showed the benefit of achieving sparse solutions.

Following the derivation of BIC for weighted likelihoods in Lumley and Scott (2015), the conventional BIC for FLARCC is defined as follows:

$$BIC_\lambda = -2\sum_{k=1}^{K} \frac{\bar{n}}{n_k} \log L_k(\hat{\boldsymbol{\beta}}(\lambda)) + \mathrm{df}(\hat{\boldsymbol{\beta}}(\lambda)) \log(N), \tag{7}$$

where $\bar{n} = N/K$ is the average sample size per study, $L_k(\boldsymbol{\beta})$ is the study-specific likelihood, $\hat{\boldsymbol{\beta}}(\lambda)$ is the estimation of $\boldsymbol{\beta}$ at tuning parameter value λ, and $\mathrm{df}(\hat{\boldsymbol{\beta}}(\lambda)) = \sum_{j=1}^{p} \mathrm{df}(\hat{\boldsymbol{\beta}}_{j,\bullet}(\lambda))$ is the total number of distinct parameters in $\hat{\boldsymbol{\beta}}(\lambda)$. The study-specific log-likelihoods for three most common models are listed below:

$$\text{Normal: } \log L_k(\hat{\boldsymbol{\beta}}(\lambda)) \propto -\frac{n_k}{2} \log \left\{ \sum_{i=1}^{n_k} \left(Y_k^{(i)} - \boldsymbol{X}_k^{(i)\top} \hat{\boldsymbol{\beta}}(\lambda) \right)^2 / n_k \right\};$$

$$\text{Logistic: } \log L_k(\hat{\boldsymbol{\beta}}(\lambda)) \propto \sum_{i=1}^{n_k} \left\{ Y_k^{(i)} \boldsymbol{X}_k^{(i)\top} \hat{\boldsymbol{\beta}}(\lambda) - \log \left(1 + e^{\boldsymbol{X}_k^{(i)\top} \hat{\boldsymbol{\beta}}(\lambda)} \right) \right\};$$

$$\text{Poisson: } \log L_k(\hat{\boldsymbol{\beta}}(\lambda)) \propto \sum_{i=1}^{n_k} \left\{ Y_k^{(i)} \boldsymbol{X}_k^{(i)\top} \hat{\boldsymbol{\beta}}(\lambda) - e^{\boldsymbol{X}_k^{(i)\top} \hat{\boldsymbol{\beta}}(\lambda)} \right\}.$$

To improve the BIC by further controlling model size and encouraging sparer models, we adapt the EBIC for FLARCC, which takes the following form:

$$EBIC_\lambda = -2\sum_{k=1}^{K} \frac{\bar{n}}{n_k} \log L_k(\hat{\boldsymbol{\beta}}(\lambda)) + \mathrm{df}(\hat{\boldsymbol{\beta}}(\lambda)) \log(N) + 2\gamma \log \sum_{j=1}^{p} \binom{K}{\mathrm{df}(\hat{\boldsymbol{\beta}}_{j,\bullet}(\lambda))}, \tag{8}$$

where $\gamma \in [0, 1]$ is a tuning parameter that is typically fixed at 1 as done in our numerical experiments. Note that EBIC reduces to BIC when $\gamma = 0$. The last term in (8) encourages a sparser solution in comparison to the conventional BIC. Simulation studies in Section 5 provide numerical evidence to elucidate the difference between BIC and EBIC in terms of their performance on achieving sparsity.

In a view of hierarchical clustering, the solution path of each covariate can be thought of as a hierarchical clustering tree. For the jth covariate, $\lambda = 0$ corresponds to the bottom of the clustering tree; and $\lambda = \lambda_{Fuse,j}$, the smallest λ value to achieve complete parameter fusion, corresponds to the top of the clustering tree. The completely heterogeneous model corresponds to the position on the solution path at $\lambda = 0$ and the completely homogeneous model corresponds to the model at $\lambda = \lambda_{Fuse} := \max_{1 \le j \le p} \lambda_{Fuse,j}$.

5. Simulation Studies

This section presents results from two simulation experiments. The first simulation compares the performance of FLARCC under different GLM regression models. The second simulation is a more complicated scenario with large K and more non-important covariates, where covariate selection is also of interest.

5.1 Simulation Experiment 1

The first simulation study aims to assess the performance of our method for different GLM regression models. For this, we consider combining data sets from $K = 10$ different studies with, for simplicity, equal sample size $n_1 = \cdots = n_{10} = 100$. Data are simulated from the following mean regression model:

$$h\{E(Y_k^{(i)})\} = \beta_{1,k} X_{1,k}^{(i)} + \beta_{2,k} X_{2,k}^{(i)} + \beta_{3,k} X_{3,k}^{(i)},\ i = 1, \ldots, 100,\ k = 1, \ldots, 10,$$

where the true coefficient vectors have the following clustering structures:

$$\begin{aligned}
\boldsymbol{\beta}_{1,\bullet} &= (\underbrace{0, \ldots, 0}_{10})^\top; \\
\boldsymbol{\beta}_{2,\bullet} &= (\underbrace{0, \ldots, 0}_{5}, \underbrace{1, \ldots, 1}_{5})^\top; \\
\boldsymbol{\beta}_{3,\bullet} &= (\underbrace{-1, \ldots, -1}_{3}, \underbrace{0, \ldots, 0}_{4}, \underbrace{1, \ldots, 1}_{3})^\top.
\end{aligned}$$

The true values in $\boldsymbol{\beta}_2$ and $\boldsymbol{\beta}_3$ are heterogeneous, while the true values in $\boldsymbol{\beta}_1$ are homogeneous across studies. The three covariates are correlated with exchangeable correlation of 0.3 and marginally distributed according to the standard normal distributions, $\mathcal{N}(0, 1)$. Three types of GLM regression models are considered: linear model for continuous normal outcomes (with errors simulated from $\mathcal{N}(0, 1)$), logistic model for binary outcomes and Poisson model for count outcomes.

To evaluate the performance of FLARCC to correctly detect patterns of all covariates, we assume all covariates are heterogeneous across studies with no prior knowledge on clustering structure of any covariate. Intercept is fitted and assumed to be homogeneous. No sparsity penalty is applied on the covariates (i.e., $\alpha = 0$) in this simulation experiment. Coefficients of all three covariates are fused simultaneously, and the optimal tuning parameter λ_{opt} is selected by EBIC. We report sensitivity and specificity as metrics of the performance of FLARCC to identify similar and distinct coefficient pairs. Sensitivity measures the proportion of equal coefficient pairs that are correctly identified. Similarly, specificity measures

Method ($\tilde{\lambda}_{opt}$)	$\boldsymbol{\beta}$	$\hat{\boldsymbol{\beta}}$ size	$\tilde{\lambda}_{Fuse,j}$	Sensitivity	Specificity	MSE when $\lambda =$ λ_{opt}	λ_{Fuse}	0
Linear: continuous response								
$s=0$ (0.154)	$\boldsymbol{\beta}_1$	1.067	0.111	0.974	–	0.001	0.002	0.012
	$\boldsymbol{\beta}_2$	2.075	1.275	0.982	1.000	0.003	0.253	0.012
	$\boldsymbol{\beta}_3$	3.081	1.368	0.982	1.000	0.004	0.603	0.012
$s=1$ (0.349)	$\boldsymbol{\beta}_1$	1.006	0.080	0.998	–	0.001	0.002	0.012
	$\boldsymbol{\beta}_2$	2.058	1.584	0.986	1.000	0.003	0.253	0.012
	$\boldsymbol{\beta}_3$	3.123	1.972	0.974	1.000	0.004	0.603	0.012
Logistic: binary response								
$s=0$ (0.066)	$\boldsymbol{\beta}_1$	1.270	0.064	0.898	–	0.010	0.005	0.070
	$\boldsymbol{\beta}_2$	2.572	0.318	0.819	0.963	0.047	0.268	0.087
	$\boldsymbol{\beta}_3$	3.682	0.437	0.784	0.964	0.069	0.607	0.091
$s=1$ (0.112)	$\boldsymbol{\beta}_1$	1.075	0.050	0.972	–	0.007	0.005	0.069
	$\boldsymbol{\beta}_2$	2.478	0.414	0.837	0.952	0.052	0.268	0.088
	$\boldsymbol{\beta}_3$	3.912	0.711	0.749	0.971	0.064	0.607	0.091
Poisson: count response								
$s=0$ (0.187)	$\boldsymbol{\beta}_1$	1.087	0.129	0.976	–	0.001	0.005	0.008
	$\boldsymbol{\beta}_2$	2.084	1.751	0.984	1.000	0.001	0.271	0.008
	$\boldsymbol{\beta}_3$	3.076	1.885	0.986	1.000	0.002	0.659	0.008
$s=1$ (0.433)	$\boldsymbol{\beta}_1$	1.047	0.087	0.992	–	0.001	0.005	0.008
	$\boldsymbol{\beta}_2$	2.088	2.060	0.984	1.000	0.002	0.271	0.008
	$\boldsymbol{\beta}_3$	3.111	2.536	0.978	1.000	0.002	0.657	0.008

Table 1: Results of simulation experiment 1 for FLARCC when scaling weight parameter $s = 0$ and $s = 1$ with λ selected by EBIC, for the linear, logistic and Poisson models. Tuning parameters are reported in log scale, i.e., $\tilde{\lambda} = \log_{10}(\lambda+1)$. Results are summarized from 1,000 replications.

the proportion of unequal coefficients pairs that are correctly identified; however, specificity is not defined for homogeneous covariates which have no unequal coefficient pairs. In addition, we calculate the mean squared error (MSE) for each $\hat{\boldsymbol{\beta}}_{j,\bullet}$ across all K studies, defined as $\text{MSE}_j = \sum_{k=1}^{K}(\hat{\beta}_{j,k} - \beta_{j,k})^2/K, j = 1, \ldots, p$, and compare with the MSE of each estimate based on homogeneous model ($\lambda = \lambda_{Fuse}$) and heterogeneous model ($\lambda = 0$).

Table 1 shows the results of simulation experiment 1 from 1,000 simulation replicates. The MSE of all estimated covariates based on FLARCC ($\lambda = \lambda_{opt}$) are consistently and significantly smaller than those based on the homogeneous ($\lambda = \lambda_{Fuse}$) and heterogeneous ($\lambda = 0$) models, regardless of the model type. FLARCC performs very well in the linear and Poisson regressions in terms of identifying the correct clustering, with the sensitivity and specificity both above 95% for all covariates (specificity is not reported for $\boldsymbol{\beta}_1$ since there is no unequal pair within $\boldsymbol{\beta}_{1,\bullet}$). Sensitivity and specificity of FLARCC drop in the logistic regression, especially as the level of heterogeneity increases. One reason for the reduced performance of FLARCC in the logistic regression is that the estimated variances of

regression coefficients in the logistic model are larger than in the linear and Poisson models, given the same coefficient setting. Therefore, the estimated parameter ordering for which our method is based on may be less accurate. For the logistic regression, increasing sample sizes is one of the possible ways to improve the performance. The performance difference between scaling weight parameter $s = 0$ and $s = 1$ in (4) is small in this case because of the relatively small number of covariates $p = 3$. Additionally, since K is small in this case, the optimal λ selected by BIC and EBIC are very close, thus we only display results based on EBIC. As p and K become larger, FLARCC will increasingly benefit from the additional weights ν_j (i.e., $s = 1$) and EBIC, as will be shown in Section 5.2. A sensitivity analysis to investigate how the initial ordering affect the performance of FLARCC is conducted, with results shown in Appendix B. We show that when the initial parameter ordering is slightly distorted, our method still achieves satisfactory performance.

5.2 Simulation Experiment 2

The second simulation study aims to evaluate the performance of FLARCC in a more challenging setting. More specifically, we consider data sets from $K = 100$ studies, each with a sample size 100, totaling 10,000 subject-level observations. Comparing to the previous setting, we increase the number of covariates and reduce the gaps between heterogeneous coefficients. For each study, we simulate data from the following linear regression model:

$$E(Y_k^{(i)}) = \sum_{j=1}^{8} \beta_{j,k} X_{j,k}^{(i)}, \ i = 1, \ldots, 100, \ k = 1, \ldots, 100.$$

The signals are set sparse; only the first four covariates with coefficient vectors, $\boldsymbol{\beta}_1$ to $\boldsymbol{\beta}_4$, are influential to Y with the true clustered effect patterns given as follows:

$$\begin{aligned}
\boldsymbol{\beta}_{1,\bullet} &= (\underbrace{0, \ldots, 0}_{50}, \underbrace{0.5, \ldots, 0.5}_{50})^\top, \\
\boldsymbol{\beta}_{2,\bullet} &= (\underbrace{-0.5, \ldots, -0.5}_{30}, \underbrace{0, \ldots, 0}_{40}, \underbrace{0.5, \ldots, 0.5}_{30})^\top, \\
\boldsymbol{\beta}_{3,\bullet} &= (\underbrace{-0.5, \ldots, -0.5}_{25}, \underbrace{0, \ldots, 0}_{25}, \underbrace{0.5, \ldots, 0.5}_{25}, \underbrace{1, \ldots, 1}_{25})^\top, \\
\boldsymbol{\beta}_{4,\bullet} &= (\underbrace{-1, \ldots, -1}_{20}, \underbrace{-0.5, \ldots, -0.5}_{20}, \underbrace{0, \ldots, 0}_{20}, \underbrace{0.5, \ldots, 0.5}_{20}, \underbrace{1, \ldots, 1}_{20})^\top,
\end{aligned}$$

whereas $\boldsymbol{\beta}_5$ to $\boldsymbol{\beta}_8$ are all zero, i.e., $\boldsymbol{\beta}_{j,\bullet} = (0, \ldots, 0)^\top$, for $j = 5, 6, 7, 8$. All covariates are equally correlated with an exchangeable correlation of 0.3 and marginally distributed according to $\mathcal{N}(0, 1)$. We set $\boldsymbol{\beta}_1$ to $\boldsymbol{\beta}_8$ as being heterogeneous from the start and fuse all of them simultaneously. We apply the additional sparsity penalty to all covariates by setting $\alpha = 1$. The intercept is assumed to be homogeneous in the analysis.

Since K is large, we also present results from individual covariate K-means clustering. This is a two-step method where we first estimate regression coefficients within each study, and then separately for each covariate, we perform the K-means clustering on the estimated study-specific coefficients of each covariate. The number of clusters is selected by the generalized cross-validation criterion $\sum_{k=1}^{K}(\hat{\beta}_k - \hat{\beta}_{c(k)})^2/(K - \text{GDF})^2$, with $\hat{\beta}_{c(k)}$ being the

Method ($\tilde{\lambda}_{opt}$)	$\boldsymbol{\beta}$	$\hat{\boldsymbol{\beta}}$ size	$\tilde{\lambda}_{Fuse,j}$	Sensitivity	Specificity	Sparsity	MSE when $\lambda =$ λ_{opt}	λ_{Fuse}	0
$s=0$ BIC (0.143)	$\boldsymbol{\beta}_1$	8.115	1.517	0.373	0.995	0.199	0.006	0.063	0.014
	$\boldsymbol{\beta}_2$	10.689	1.551	0.401	0.996	0.166	0.008	0.150	0.014
	$\boldsymbol{\beta}_3$	13.107	1.962	0.443	0.997	0.113	0.008	0.313	0.014
	$\boldsymbol{\beta}_4$	15.178	1.984	0.461	0.997	0.091	0.009	0.500	0.014
	$\boldsymbol{\beta}_5$	4.818	0.301	0.322	–	0.338	0.003	0.000	0.014
	$\boldsymbol{\beta}_6$	4.860	0.305	0.322	–	0.339	0.003	0.000	0.014
	$\boldsymbol{\beta}_7$	4.860	0.302	0.321	–	0.330	0.003	0.000	0.014
	$\boldsymbol{\beta}_8$	4.820	0.301	0.319	–	0.336	0.003	0.000	0.014
$s=0$ EBIC (0.159)	$\boldsymbol{\beta}_1$	7.538	1.509	0.417	0.994	0.224	0.006	0.063	0.014
	$\boldsymbol{\beta}_2$	9.975	1.546	0.441	0.995	0.182	0.007	0.150	0.014
	$\boldsymbol{\beta}_3$	12.212	1.953	0.483	0.996	0.124	0.008	0.313	0.014
	$\boldsymbol{\beta}_4$	14.096	1.978	0.503	0.997	0.101	0.008	0.500	0.014
	$\boldsymbol{\beta}_5$	4.388	0.298	0.377	–	0.394	0.003	0.000	0.014
	$\boldsymbol{\beta}_6$	4.413	0.303	0.379	–	0.397	0.003	0.000	0.014
	$\boldsymbol{\beta}_7$	4.408	0.299	0.374	–	0.385	0.003	0.000	0.014
	$\boldsymbol{\beta}_8$	4.385	0.298	0.374	–	0.392	0.003	0.000	0.014
$s=1$ EBIC (0.492)	$\boldsymbol{\beta}_1$	3.563	1.589	0.800	0.981	0.422	0.006	0.063	0.014
	$\boldsymbol{\beta}_2$	6.111	1.810	0.708	0.989	0.291	0.007	0.150	0.014
	$\boldsymbol{\beta}_3$	8.843	2.388	0.667	0.994	0.168	0.007	0.313	0.014
	$\boldsymbol{\beta}_4$	11.358	2.521	0.635	0.995	0.128	0.008	0.500	0.014
	$\boldsymbol{\beta}_5$	1.329	0.275	0.928	–	0.932	0.000	0.000	0.014
	$\boldsymbol{\beta}_6$	1.321	0.280	0.933	–	0.937	0.000	0.000	0.014
	$\boldsymbol{\beta}_7$	1.311	0.274	0.934	–	0.935	0.000	0.000	0.014
	$\boldsymbol{\beta}_8$	1.321	0.273	0.929	–	0.936	0.000	0.000	0.014
							MSE from K-means		
K-means GCV (GDF)	$\boldsymbol{\beta}_1$	7.196	–	0.753	0.971	0.000			0.008
	$\boldsymbol{\beta}_2$	11.236	–	0.671	0.983	0.000			0.009
	$\boldsymbol{\beta}_3$	13.721	–	0.639	0.984	0.000			0.011
	$\boldsymbol{\beta}_4$	18.308	–	0.527	0.985	0.000			0.014
	$\boldsymbol{\beta}_5$	6.415	–	0.759	–	0.000			0.004
	$\boldsymbol{\beta}_6$	5.271	–	0.769	–	0.000			0.004
	$\boldsymbol{\beta}_7$	5.629	–	0.767	–	0.000			0.004
	$\boldsymbol{\beta}_8$	5.080	–	0.794	–	0.000			0.004

Table 2: Result of simulation experiment 2 under the linear model. Scaling weight parameter is set at $s = 0$ and $s = 1$. Tuning parameters are reported in log scale, i.e., $\tilde{\lambda} = \log_{10}(\lambda+1)$. Sparsity denotes the proportion of zero in estimation. Results are summarized from 1,000 replications.

cluster center of $\hat{\beta}_k$ and GDF is the generalized degrees of freedom estimated according to Ye (1998), where purturbations are generated independently from $\mathcal{N}(0, 0.01)$. The cluster centroids are then used as the estimates of the group-level parameters.

Table 2 summarizes the simulation results for linear model where the errors are generated independently from $\mathcal{N}(0, 1)$. Similar to simulation 1, FLARCC gives the smallest MSE for heterogeneous covariates, $\boldsymbol{\beta}_1$ to $\boldsymbol{\beta}_4$, among all three models, and has comparable MSE as the homogeneous model for homogeneous covariates, $\boldsymbol{\beta}_5$ to $\boldsymbol{\beta}_8$. More interestingly, when K is large, BIC does not provide satisfactory model selection, erring on the lack of parsimony, while EBIC encourages stronger fusion and improves the ability to detect equal coefficient pairs in all eight covariates, regardless of their levels of heterogeneity. In addition, EBIC improves the sparsity detection among both the important and nonimportant covariates. It is interesting to note that the choice between BIC and EBIC does not alter solution paths, but only model selection. FLARCC with scaling weight parameter $s = 1$ has the best clustering performance among all compared methods. The difference between the choices of $s = 0$ and $s = 1$ is substantial in simulation 2, in contrast to the results from simulation 1. This indicates that the covariate-specific weights for heterogeneity $\{\nu_j\}_{j=1}^p$ are very effective to improve the performance of the proposed fusion learning, especially when K and p are large. Sensitivity and specificity of the two-step K-means clustering method are higher than those of FLARCC with $s = 0$, but lower than those of FLARCC with $s = 1$. The two-step K-means has larger MSE than FLARCC because it does not consider the correlation between covariates. More importantly, the K-means clustering is a model-free method, so the results obtained from this method cannot be plugged in back to the model for prediction. As suggested from the empirical results of both simulation experiments, EBIC tends to provide better model selection for FLARCC than the conventional BIC.

6. Application: Clustering of Cohort Effects

In this data analysis example, we like to demonstrate the use of our method to derive clusters of cohort effects. Here we consider the Panel Study of Income Dynamics (PSID), which is a household survey study following thousands of families across different states in the US. PSID collects information of employment, income, health, and so on. In this data analysis, we focus on the association of household income with body mass index (BMI) on school-aged children between age of 11 and 19, adjusted for age, gender and birth weight. Data of 1880 children were gathered from four census regions (1-Northeast, 2-Midwest, 3-South and 4-West), as defined by U.S. Census Bureau (2015). All variables are standardized before model fitting. We are interested in investigating if regional heterogeneity exists and if the effects of interest differ across regions with region-dependent patterns.

Table 3 shows the results of coefficient estimates obtained from three different models: (A) homogeneous model ($\lambda = \lambda_{Fuse}$), coefficients estimated by combining data sets from four regions, (B) heterogeneous model ($\lambda = 0$), coefficients estimated separately by region-specific data, and (C) FLARCC ($\lambda = \lambda_{EBIC}$). Model A suggests that age and birth weight are positively associated with BMI for the subjects, but income was negatively associated with BMI. The estimates from Model B suggest that heterogeneous coefficient patterns exist among these associations since conclusions differ between regions. Model C appears more sensible when regression coefficients are heterogeneous across these regions. Since K and p are small in this data application, we apply FLARCC with $s = 0$ on the PSID data, assuming effects of income, age, gender and birth weight are heterogeneous across regions, and set sparsity parameter $\alpha = 1$ for variable selection.

Region	n	Intercept $\boldsymbol{\beta}_0$	Age $\boldsymbol{\beta}_1$	Sex $\boldsymbol{\beta}_2$	Birth Wt. $\boldsymbol{\beta}_3$	Income $\boldsymbol{\beta}_4$
(A) Homogeneous model – combine all regions						
All regions	1880	0.000	0.206	0.016	0.063	-0.096
(B) Heterogeneous model – region specific estimates						
1-Northeast	239	-0.133	0.228	-0.079	-0.003	0.004
2-Midwest	493	-0.054	0.229	0.017	0.124	-0.132
3-South	805	0.128	0.158	0.095	0.068	-0.071
4-West	343	-0.155	0.236	-0.083	0.057	-0.074
(C) Fused model using FLARCC						
1-Northeast	239	-0.093	0.201	-0.036	0.000	0.000
2-Midwest	493	-0.093	0.201	0.000	0.021	-0.047
3-South	805	0.075	0.201	0.000	0.021	-0.047
4-West	343	-0.093	0.201	-0.036	0.021	-0.047

Table 3: Coefficient estimates of the homogeneous model ($\lambda = \lambda_{Fuse}$), the heterogeneous model ($\lambda = 0$) and the fused model using FLARCC with λ selected by EBIC, respectively.

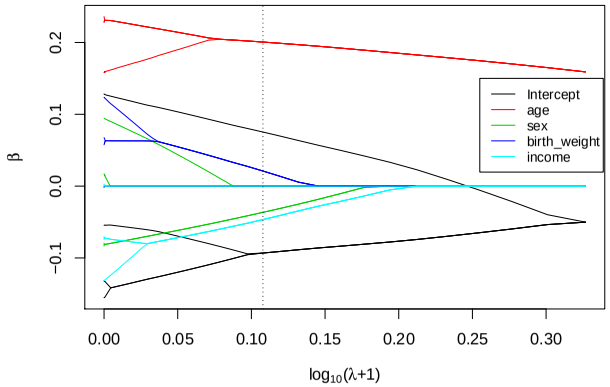


Figure 2: FLARCC solution paths of all covariates over the transformed tuning parameter $\tilde{\lambda} = \log_{10}(\lambda+1)$, with $s = 0$. The vertical dotted line denotes the optimal tuning parameter value $\tilde{\lambda}_{EBIC}$.

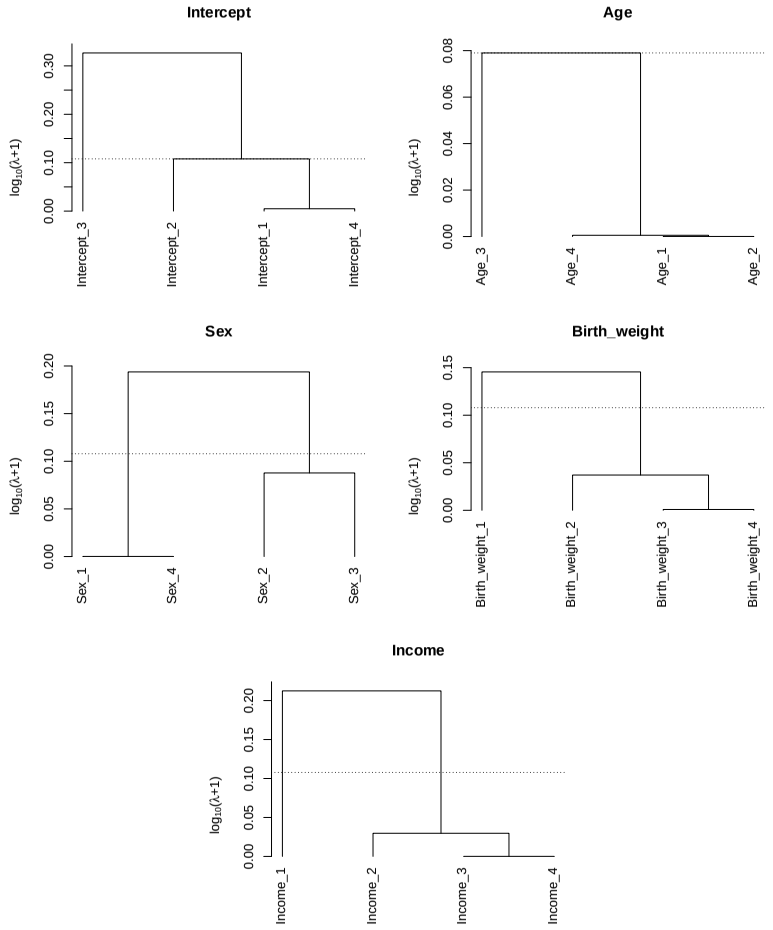


Figure 3: *Fusograms* of all covariates based on FLARCC solution paths. The horizontal dotted lines denote the optimal regression coefficient clustering determined by EBIC.

Based on the results from FLARCC, the estimated mean of standardized BMI in the South is 0.168 higher (or 0.97 higher in original scale of BMI) than that of the other three regions, which share the same mean. The effects of age are consistent across four regions. The effects of gender are classified into two clusters. The mean of standardized BMI of females is 0.036 lower (or 0.42 lower in original scale) than that of males in the Northeast and the West, but males and females have the same mean BMI in the Midwest and the South. Standardized BMI increases by 0.021 for every standard deviation increase of birth weight (or BMI increases by 0.19 for every unit increase of birth weight) in all regions except the Northeast. Similarly, standardized BMI decreases by 0.047 for every standard deviation increase of log income (or BMI decreases by 0.27 for every unit increase of income) in all regions except the Northeast where BMI is not affected by income. The leave-one-out mean squared prediction errors for model A, B and C are 0.953, 0.945 and 0.950, respectively. The differences between the prediction errors are small because of the relatively small effect sizes of the heterogeneous covariates identified by FLARCC, i.e., sex, birth weight and income. The most significant covariate, age, is homogeneous thus it does not differentiate the prediction power among the three models. Solution paths and *fusograms* of all covariates are shown in Figure 2 and Figure 3, respectively, for illustration. In summary, FLARCC ensures parsimony where necessary to maximize the prediction power of the final model; and it provides more informative interpretation and better visualization than the other two traditional models.

7. Concluding Remarks

The proposed method brings a new perspective to model fitting when combining multiple data sets from different sources is of primary interest. As data volumes and data sources grow fast, more and more opportunities and demands emerge in practice to borrow strengths of combined data sets. In such case, traditional methods are challenged by the complex data structures and do not provide desirable treatments and meaningful interpretations to data heterogeneity, especially when the number of data sets is very large. FLARCC allows the flexibility to explore the heterogeneity pattern of parameters among large number of data sets by tuning the shrinkage parameter.

When K and p are small, weights $\{\nu_j\}_{j=1}^p$ do not contribute to much difference in terms of clustering and estimation. However, since only one tuning parameter is used to regularize the fusion of all covariates, when both K and p are large, we suggest letting $s > 0$ to allow covariate-specific weights adapting to the heterogeneousness of coefficients from individual covariates to achieve better results. In addition, the estimation consistency of rank estimator is a critical component needed to determine adjacent pairs. The current consistency is established under the case of K being fixed, and the validity of its property is unknown when K increases along the total sample size.

FLARCC can be applied to various scientific problems, such as the detection of outlying studies by singling out outlying coefficients; it can also be applied to the clustering of patient trajectories by viewing the time series data of patients as individual studies. Essentially, all study that are interested in the group-specific effects may be analyzed from the perspective of parameter fusion using the proposed method.

Acknowledgments

We thank Drs. Fei Wang and Ling Zhou for helpful discussion. We are grateful to the action editor and two anonymous reviewers for their constructive comments that have led to an improvement of this paper. This research is supported by an NIH grant R01 ES024732.

Appendix A. Theorem Proofs

Proof of Theorem 1: The proof of Theorem 1 closely follows arguments given in Zou (2006). Without loss of generality, we assume $n_1 = \cdots = n_K = n$ and $N = Kn$. As K is fixed, $n \to \infty$ implies $N \to \infty$ in the same order. We assume the following regularity conditions:

(i) The Fisher information matrix is finite and positive definite,

$$\boldsymbol{I}(\boldsymbol{\theta}^*) = E\left[\phi''\left(\boldsymbol{X}^\top\boldsymbol{\theta}^*\right)\boldsymbol{X}\boldsymbol{X}^\top\right].$$

Here, $\boldsymbol{\theta}^*_{(Kp\times 1)}$ is the true parameters, $\boldsymbol{X}_{(N\times Kp)}$ is the design matrix corresponding to $\boldsymbol{\theta}$ and ϕ is the link function (i.e., $\phi' = h^{-1}$) defined in the following optimization problem

$$\hat{\boldsymbol{\theta}}^{\boldsymbol{W}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\left\{-\frac{1}{K}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i=1}^{n_k}\left(Y_k^{(i)}\boldsymbol{X}_k^{(i)\top}\boldsymbol{\theta}(\lambda) - \phi\left(\boldsymbol{X}_k^{(i)\top}\boldsymbol{\theta}(\lambda)\right)\right) + P_{\lambda,\alpha}(\boldsymbol{\theta})\right\}$$

with $P_{\lambda,\alpha}(\boldsymbol{\theta})$ as defined in (4), and $\hat{\boldsymbol{\theta}}^{\boldsymbol{W}}$ is the estimator with true ordering $\boldsymbol{W}$ given.

(ii) There is a sufficiently large open set $\mathcal{O}$ that contains $\boldsymbol{\theta}^*$ such that $\forall\boldsymbol{\theta}\in\mathcal{O}$,

$$|\phi'''(\boldsymbol{X}^\top\boldsymbol{\theta})| \le M(\boldsymbol{X}^\top) < \infty, \text{ and}$$
$$E\left[M(\boldsymbol{X})|x_j x_k x_l|\right] < \infty$$

for a suitable function M and all $1 \le j,k,l \le Kp$.

First we prove asymptotic normality. For $\forall s \ge 0$ and $r > 0$, let $\boldsymbol{\theta} = \boldsymbol{\theta}^* + \boldsymbol{u}/\sqrt{N}$. Define

$$\Gamma_N(\boldsymbol{u}) = -\sum_{k=1}^{K}\sum_{i=1}^{n}\left(Y_k^{(i)}\boldsymbol{X}_k^{(i)\top}\left(\boldsymbol{\theta}^* + \frac{\boldsymbol{u}}{\sqrt{N}}\right) - \phi\left(\boldsymbol{X}_k^{(i)\top}\left(\boldsymbol{\theta}^* + \frac{\boldsymbol{u}}{\sqrt{N}}\right)\right)\right)$$
$$+\lambda_N\sum_{j=1}^{p}\sum_{k=1}^{K}\hat{\omega}_{j,k}\left|\theta^*_{j,k} + \frac{u_{j,k}}{\sqrt{N}}\right|$$

where $\hat{\omega}_{j,k}$ is specified in (6). Let $\hat{\boldsymbol{u}}^{(N)} = \arg\min_{\boldsymbol{u}}\Gamma_N(\boldsymbol{u})$; then $\hat{\boldsymbol{u}}^{(N)} = \sqrt{N}(\hat{\boldsymbol{\theta}}^{\boldsymbol{W}} - \boldsymbol{\theta}^*)$. By Taylor expansion, we have $\Gamma_N(\boldsymbol{u}) - \Gamma_N(\boldsymbol{0}) = H^{(N)}(\boldsymbol{u})$, where

$$H^{(N)}(\boldsymbol{u}) \equiv A_1^{(N)} + A_2^{(N)} + A_3^{(N)} + A_4^{(N)},$$

with

$$A_1^{(N)} = -\sum_{k=1}^{K}\sum_{i=1}^{n}\left[Y_k^{(i)} - \phi'(\boldsymbol{X}_k^{(i)\top}\boldsymbol{\theta}^*)\right]\frac{\boldsymbol{X}_k^{(i)\top}\boldsymbol{u}}{\sqrt{N}},$$
$$A_2^{(N)} = \sum_{k=1}^{K}\sum_{i=1}^{n}\frac{1}{2}\phi''(\boldsymbol{X}_k^{(i)\top}\boldsymbol{\theta}^*)\boldsymbol{u}^\top\frac{\boldsymbol{X}_k^{(i)}\boldsymbol{X}_k^{(i)\top}}{\sqrt{N}}\boldsymbol{u},$$
$$A_3^{(N)} = \frac{\lambda_N}{\sqrt{N}}\sum_{j=1}^{p}\sum_{k=1}^{K}\hat{\omega}_{j,k}\sqrt{N}\left(\left|\theta_{j,k}^* + \frac{u_{j,k}}{\sqrt{N}}\right| - \left|\theta_{j,k}^*\right|\right),$$
$$\text{and } A_4^{(N)} = N^{-3/2}\sum_{k=1}^{K}\sum_{i=1}^{n}\frac{1}{6}\phi'''\left(\boldsymbol{X}_k^{(i)\top}\tilde{\boldsymbol{\theta}}_*\right)\left(\boldsymbol{X}_k^{(i)\top}\boldsymbol{u}\right)^3,$$

where $\tilde{\boldsymbol{\theta}}_*$ is between $\boldsymbol{\theta}^*$ and $\boldsymbol{\theta}^* + \boldsymbol{u}/\sqrt{N}$. The asymptotic limits of $A_1^{(N)}$, $A_2^{(N)}$ and $A_4^{(N)}$ is exactly the same as those in the proof of Theorem 4 in Zou (2006). It suffice to show that $A_3^{(N)}$ has the same asymptotic limit. If $\theta_{j,k}^* \neq 0$, $\hat{\omega}_{j,1} \to_p \alpha|\theta_{j,1}^*|^{-r}$, $\hat{\omega}_{j,k} \to_p |\sum_{k'=2}^{K}\theta_{j,k'}^*|^{-s}|\theta_{j,k}^*|^{-r}$ for $k = 2,\ldots,K$, and $\sqrt{N}\left(\left|\theta_{j,k}^* + \frac{u_{j,k}}{\sqrt{N}}\right| - \left|\theta_{j,k}^*\right|\right) \to u_{j,k}\operatorname{sgn}(\theta_{j,k}^*)$. Thus by Slutsky's theorem, $A_3^{(N)} \to 0$. If $\theta_{j,k}^* = 0$, for $k = 1$, since $\sqrt{N}\hat{\theta}_{j,1} = O_p(1)$, $\frac{\lambda_N}{\sqrt{N}}N^{r-2}\alpha(|\sqrt{N}\hat{\theta}_{j,1}|)^{-r} \to \infty$; for $k = 2,\ldots,K$, if $\sum_{k'=2}^{K}\theta_{j,k'}^* = 0$ (i.e., homogeneous), $\sqrt{N}\sum_{k'=2}^{K}\hat{\theta}_{j,k'} = O_p(1)$, thus $\frac{\lambda_N}{\sqrt{N}}N^{\frac{s+r}{2}}(|\sqrt{N}\sum_{k'=2}^{K}\hat{\theta}_{j,k'}|)^{-s}(|\sqrt{N}\hat{\theta}_{j,k}|)^{-r} \to \infty$; similarly, if $\sum_{k'=2}^{K}\theta_{j,k'}^* \neq 0$ (i.e., heterogeneous), $\sum_{k'=2}^{K}\hat{\theta}_{j,k'} \to_p \sum_{k'=2}^{K}\theta_{j,k'}^*$, $\frac{\lambda_N}{\sqrt{N}}\hat{\omega}_{j,k} \to \infty$ still holds. And since $\sqrt{N}\left(\left|\theta_{j,k}^* + \frac{u_{j,k}}{\sqrt{N}}\right| - \left|\theta_{j,k}^*\right|\right) \to |u_{j,k}|$, we have the following result summary:

$$\frac{\lambda_N}{\sqrt{N}}\hat{\omega}_{j,k}\sqrt{N}\left(\left|\theta_{j,k}^* + \frac{u_{j,k}}{\sqrt{N}}\right| - \left|\theta_{j,k}^*\right|\right) \to_p \begin{cases} 0 & \text{if } \theta_{j,k}^* \neq 0 \\ 0 & \text{if } \theta_{j,k}^* = 0 \text{ and } u_{j,k} = 0 \\ \infty & \text{if } \theta_{j,k}^* = 0 \text{ and } u_{j,k} \neq 0. \end{cases}$$

Following same arguments in Zou (2006)'s proof of Theorem 4, we have $\hat{\boldsymbol{u}}_{\mathcal{A}}^{(N)} \to_d \mathcal{N}(0, \boldsymbol{I}_{11}^{-1})$ and $\hat{\boldsymbol{u}}_{\mathcal{A}^c}^{(N)} \to_d \boldsymbol{0}$. The proof of the consistency part is similar and thus omitted. ■

Proof of Lemma 2: The estimated ordering $\hat{\boldsymbol{U}}_j$ of $\boldsymbol{\beta}_{j,\bullet}^*$ is only determined by the differences between distinct parameter groups within $\boldsymbol{\beta}_{j,\bullet}^*$. First note that for any $0 < \epsilon < 1$, if two parameters $\beta_{j,k}^*$ and $\beta_{j,k'}^*$ are in the same parameter group (i.e., $\beta_{j,k}^* = \beta_{j,k'}^*$), assigning arbitrary ordering between them will not affect the estimated ordering of the parameters between groups, because the ordering within the same parameter group is exchangeable. On the other hand, when two parameters $\beta_{j,k}^*$ and $\beta_{j,k'}^*$ are from different parameter groups,

without loss of generality, let $\beta_{j,k}^* > \beta_{j,k'}^*$, the probability of estimating a wrong ordering

$$\begin{aligned} P\left(\mathbf{1}\{\hat{\beta}_{j,k'} \geq \hat{\beta}_{j,k}\} > \epsilon\right) &= P\left(\hat{\beta}_{j,k'} \geq \hat{\beta}_{j,k}\right) \\ &= P\left(\hat{\beta}_{j,k'} - \hat{\beta}_{j,k} + \beta_{j,k}^* - \beta_{j,k'}^* \geq \beta_{j,k}^* - \beta_{j,k'}^*\right) \\ &\leq P\left(|\hat{\beta}_{j,k'} - \beta_{j,k'}^*| + |\hat{\beta}_{j,k} - \beta_{j,k}^*| > 0\right) \\ &= 1 - P\left(\hat{\beta}_{j,k'} = \beta_{j,k'}^*\right)P\left(\hat{\beta}_{j,k} = \beta_{j,k}^*\right) \to 0 \end{aligned}$$

as $n \to \infty$ since $\hat{\beta}_{j,k'}$ and $\hat{\beta}_{j,k}$ are independent and consistent estimators. Similarly, the consistency of the estimated ordering $\hat{\boldsymbol{V}}_j$ of the absolute values in vector $\boldsymbol{\beta}_{j,\bullet}^*$ can be derived by taking the square of the absolute values and following the same argument as for $\hat{\boldsymbol{U}}_j$. ■

Proof of Theorem 3: Here we assume the same regularity condition as in Theorem 1. To complete this proof, we first define the event $\mathcal{W}$ when the orderings of all p covariates are correctly assigned as

$$\mathcal{W} = \bigcap_{j=1}^{p}\left(\{\hat{\boldsymbol{U}}_j = \boldsymbol{U}_j\} \cap \{\hat{\boldsymbol{V}}_j = \boldsymbol{V}_j\}\right).$$

Let $\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}}$ be $\hat{\boldsymbol{\theta}}_{\mathcal{W}}$ when $\mathcal{W}$ occurs; otherwise, denote it as $\hat{\boldsymbol{\theta}}_{\mathcal{W}^c}$. Then, the estimator can be rewritten as

$$\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}} = \hat{\boldsymbol{\theta}}_{\mathcal{W}}\mathbf{1}\{\mathcal{W}\} + \hat{\boldsymbol{\theta}}_{\mathcal{W}^c}\mathbf{1}\{\mathcal{W}^c\}$$

and therefore

$$\sqrt{N}\left(\hat{\boldsymbol{\theta}}^{\hat{\boldsymbol{W}}} - \boldsymbol{\theta}^*\right) = \sqrt{N}\left(\hat{\boldsymbol{\theta}}_{\mathcal{W}} - \boldsymbol{\theta}^*\right)\mathbf{1}\{\mathcal{W}\} + \sqrt{N}\left(\hat{\boldsymbol{\theta}}_{\mathcal{W}^c} - \boldsymbol{\theta}^*\right)\mathbf{1}\{\mathcal{W}^c\}. \tag{9}$$

By Theorem 1, we have $\sqrt{N}\left(\hat{\boldsymbol{\theta}}_{\mathcal{W}} - \boldsymbol{\theta}^*\right) = O(1)$ and $\sqrt{N}\left(\hat{\boldsymbol{\theta}}_{\mathcal{W}^c} - \boldsymbol{\theta}^*\right) = O(1)$ as $n \to \infty$. By Lemma 2, we have $P(\mathcal{W}) \to 1$ and $P(\mathcal{W}^c) \to 0$ as $n \to \infty$. Therefore, by Slutsky's Theorem, (9) converge to the same distribution as $\sqrt{N}\left(\hat{\boldsymbol{\theta}}_{\mathcal{W}} - \boldsymbol{\theta}^*\right)$. Similarly, by results from Theorem 1 and Lemma 2, we have selection consistency

$$P(\hat{\mathcal{A}}^{\hat{\boldsymbol{W}}} = \mathcal{A}) = P(\hat{\mathcal{A}}^{\hat{\boldsymbol{W}}} = \mathcal{A}|\mathcal{W})P(\mathcal{W}) + P(\hat{\mathcal{A}}^{\hat{\boldsymbol{W}}} = \mathcal{A}|\mathcal{W}^c)P(\mathcal{W}^c) \to 1$$

as $n \to \infty$. This completes the proof of the Theorem 3. ■

Appendix B. Performance with Distorted Parameter Ordering

Under the same setting as simulation experiment 1 in Section 5.1 with $\alpha = 0$ and $s = 0$, we conduct a sensitivity analysis to evaluate the performance of FLARCC when parameter ordering is incorrectly specified. Specifically, we report results of sensitivity, specificity and MSE for the linear regression model when the coefficient ordering is determined from the initial estimate with distortion through an added disturbance ϵ, $\hat{\boldsymbol{\beta}} + \epsilon$, where $\hat{\boldsymbol{\beta}}$ from (1) and $\epsilon \sim \mathcal{N}(0, v^2)$. As v^2 increases, the percent of order switching in initial estimates increases. Sensitivity, specificity and MSE in relation to the percentage of wrongly ordered

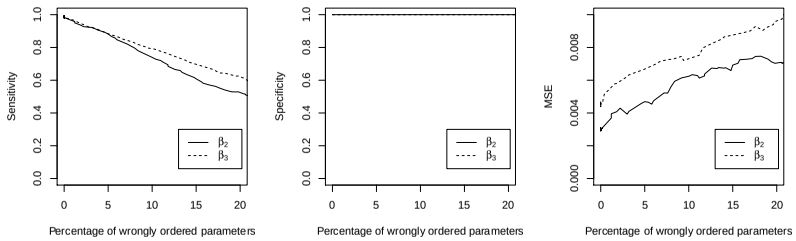


Figure 4: Clustering sensitivity and mean squared error of two heterogeneous slope parameters $\boldsymbol{\beta}_2$ and $\boldsymbol{\beta}_3$ based on FLARCC with λ selected by EBIC, as the percent of distorted ordering increases. Results are summarized from 100 replications.

parameters are displayed in Figure 4 for the two heterogeneous effects $\boldsymbol{\beta}_2$ and $\boldsymbol{\beta}_3$, and the homogeneous parameter $\boldsymbol{\beta}_1$ is not included in the comparison because of no effect from the distortion on its performance. As the percentage of wrongly ordered parameters increases, as expected, sensitivity becomes lower and MSE becomes larger. However, specificity remains unaffected. When the distortion of ordering is mild ($\leq 10\%$), the performance of FLARCC appears satisfactory in this simulation setting.

References

Jiahua Chen and Zehua Chen. Extended bayesian information criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771, 2008.

Rebecca DerSimonian and Raghu Kacker. Random-effects model for meta-analysis of clinical trials: an update. *Contemporary Clinical Trials*, 28(2):105–114, 2007.

Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010.

Xin Gao and Peter X-K Song. Composite likelihood bayesian information criteria for model selection in high-dimensional data. *Journal of the American Statistical Association*, 105 (492):1531–1540, 2010.

Gene V Glass. Primary, secondary, and meta-analysis of research. *Educational Researcher*, 5(10):3–8, 1976.

Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

Lars Peter Hansen. Large sample properties of generalized method of moments estimators. *Econometrica*, 50(4):1029–1054, 1982.

Zheng Tracy Ke, Jianqing Fan, and Yichao Wu. Homogeneity pursuit. *Journal of the American Statistical Association*, 110(509):175–194, 2015.

Jeffrey T Leek and John D Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genetics*, 3(9):1724–1735, 2007.

Hung-Mo Lin, H Myron Kauffman, Maureen A McBride, Darcy B Davies, John D Rosendale, Carol M Smith, Erick B Edwards, O Patrick Daily, James Kirklin, Charles F Shield, and Lawrence G Hunsicker. Center-specific graft and patient survival rates: 1997 united network for organ sharing (unos) report. *Journal of the American Medical Association*, 280(13):1153–1160, 1998.

Dungang Liu, Regina Y Liu, and Minge Xie. Multivariate meta-analysis of heterogeneous studies using only summary statistics: efficiency and robustness. *Journal of the American Statistical Association*, 110(509):326–340, 2015.

Kirk E Lohmueller, Celeste L Pearce, Malcolm Pike, Eric S Lander, and Joel N Hirschhorn. Meta-analysis of genetic association studies supports a contribution of common variants to susceptibility to common disease. *Nature Genetics*, 33(2):177–182, 2003.

Thomas Lumley and Alastair Scott. AIC and BIC for modeling with complex survey data. *Journal of Survey Statistics and Methodology*, 3(1):1–18, 2015.

Wei Pan, Xiaotong Shen, and Binghui Liu. Cluster analysis: unsupervised learning via supervised learning with a non-convex penalty. *The Journal of Machine Learning Research*, 14(1):1865–1889, 2013.

Jin Qin and Jerry Lawless. Empirical likelihood and general estimating equations. *The Annals of Statistics*, 22(1):300–325, 1994.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978.

Paul G Shekelle, Mary L Hardy, Sally C Morton, Margaret Maglione, Walter A Mojica, Marika J Suttorp, Shannon L Rhodes, Lara Jungvig, and James Gagné. Efficacy and safety of ephedra and ephedrine for weight loss and athletic performance: a meta-analysis. *Journal of the American Medical Association*, 289(12):1537–1545, 2003.

Xiaotong Shen and Hsin-Cheng Huang. Grouping pursuit through a regularization solution surface. *Journal of the American Statistical Association*, 105(490):727–739, 2010.

Sunyoung Shin, Jason Fine, and Yufeng Liu. Adaptive estimation with partially overlapping models. *Statistica Sinica*, 26(1):235–253, 2016.

Patrick F Sullivan, Michael C Neale, and Kenneth S Kendler. Genetic epidemiology of major depression: review and meta-analysis. *American Journal of Psychiatry*, 157(10): 1552–1562, 2000.

Alexander J Sutton and Julian Higgins. Recent developments in meta-analysis. *Statistics in Medicine*, 27(5):625–650, 2008.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58(1):267–288, 1996.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

U.S. Census Bureau. Regions and divisions. `http://www.census.gov/econ/census/help/geography/regions_and_divisions.html`, 2015. [Online; accessed 10-31-2015].

Fei Wang, Lu Wang, and Peter X-K Song. Fused lasso with the adaptation of parameter ordering in combining multiple studies with repeated measurements. *Biometrics*, 2016. doi: 10.1111/biom.12496. Advance online publication.

Minge Xie, Kesar Singh, and William E Strawderman. Confidence distributions and a unifying framework for meta-analysis. *Journal of the American Statistical Association*, 106(493):320–333, 2012.

Sen Yang, Lei Yuan, Ying-Cheng Lai, Xiaotong Shen, Peter Wonka, and Jieping Ye. Feature grouping and selection over an undirected graph. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 922–930. ACM, 2012.

Jianming Ye. On measuring and correcting the effects of data mining and model selection. *Journal of the American Statistical Association*, 93(441):120–131, 1998.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

Journal of Machine Learning Research 17 (2016) 1-5 Submitted 12/15; Revised 5/16; Published 6/16

The LRP Toolbox for Artificial Neural Networks

Sebastian Lapuschkin sebastian.lapuschkin@hhi.fraunhofer.de
Fraunhofer Heinrich Hertz Institute, Video Coding and Analytics
10587 Berlin, Germany

Alexander Binder alexander_binder@sutd.edu.sg
Singapore University of Technology, ISTD
Singapore 487372, Singapore

Grégoire Montavon gregoire.montavon@tu-berlin.de
Berlin Institute of Technology, Machine Learning Group
10623 Berlin, Germany

Klaus-Robert Müller klaus-robert.mueller@tu-berlin.de
Berlin Institute of Technology, Machine Learning Group
10623 Berlin, Germany
Korea University, Department of Brain and Cognitive Engineering
Seoul 02841, Korea

Wojciech Samek wojciech.samek@hhi.fraunhofer.de
Fraunhofer Heinrich Hertz Institute, Video Coding and Analytics
10587 Berlin, Germany

Editor: Geoff Holmes

Abstract

The Layer-wise Relevance Propagation (LRP) algorithm explains a classifier's prediction specific to a given data point by attributing *relevance scores* to important components of the input by using the topology of the learned model itself. With the LRP Toolbox we provide platform-agnostic implementations for explaining the predictions of pre-trained state of the art Caffe networks and stand-alone implementations for fully connected Neural Network models. The implementations for Matlab and python shall serve as a playing field to familiarize oneself with the LRP algorithm and are implemented with readability and transparency in mind. Models and data can be imported and exported using raw text formats, Matlab's `.mat` files and the `.npy` format for numpy or plain text.

Keywords: layer-wise relevance propagation, explaining classifiers, deep learning, artificial neural networks, computer vision

1. Introduction

Classification of images has become a key ingredient in many computer vision applications, with nonlinear methods such as Deep Neural Networks (DNNs) being the gold standard in the fields of vision (Krizhevsky et al., 2012; Ciresan et al., 2012b; Szegedy et al., 2014; Ciresan et al., 2012a), natural language processing (Collobert et al., 2011; Socher et al., 2013), speech recognition (Yu and Deng, 2014) or physics (Montavon et al., 2013); see also (Montavon et al., 2012). Although performing incredibly well, they lack transparency. In the sciences, however, understanding a learning machine in order to gain scientific insight on the problem analysed is often as important as record prediction performance. This is also one of the reasons for the popularity of linear modeling as interpretation is straight forward. In this sense DNNs so far held a disadvantage in practice over simpler but interpretable models. Especially DNNs act as black boxes due to their multilayer nonlinear structure. However, interest in gaining insight into the decision process of nonlinear methods has peaked recently. Several works have been using sensitivity maps (Baehrens et al., 2010; Rasmussen et al., 2012) for visualization of classifier predictions which were based on using partial derivatives at the prediction point $\boldsymbol{x}$.

For DNNs promising approaches for opening the black box are deconvolution networks (Zeiler and Fergus, 2014) highlighting activated input patterns for object detected during the forward pass of a convolutional neural network, saliency maps (Simonyan et al., 2013) visualizing local sensitivities at the input point and LRP (Bach et al., 2015) producing explanatory input patterns that indicate evidence for or against a prediction target of choice in given data. The latter – for which this toolbox is provided and (Samek et al., 2015) have measured and verified meaningfulnes of the computed heatmaps – takes a pre-trained model and an evaluation data point and explains the classifier's decision, e.g. LRP decomposes the decision function such that for each input dimension d a *relevance score* $R_d^{(1)}$ is computed, which indicates that the state of x_d speaks for the presence of the prediction target if $R_d^{(1)} > 0$ and against it if $R_d^{(1)} < 0$. By allowing both signs for $R_d^{(1)}$ also class background information can be modelled in a natural manner.

To this end, the model output $f(\boldsymbol{x})$ is backwards-propagated through the model to the input layer as *relevance* R, by taking into account the model's reaction to the input $\boldsymbol{x}$ in all its intermediate representations at all computational layers l. Due to the relevance conservation principle as a constraint to LRP (Bach et al., 2015), no relevance is lost or gained in between layers of computation, e.g.

$$\sum_i R_i^{(l)} = \sum_j R_j^{(l+1)} \text{ and thus } \sum_d R_d^{(1)} = f(\boldsymbol{x}).$$

Note that to this end, the propagated relevances are always normalized. The relevance values $R_d^{(1)}$ can then be visualized as a heatmap, providing valuable insight to the classifier's decision process. For a theoretical view on LRP we refer the reader to Montavon et al. (2015)

2. Capabilities of the LRP Toolbox for Artificial Neural Networks

The LRP Toolbox provides platform-independant stand-alone implementations of the LRP algorithm for python and Matlab, as well as adapted `.cpp` modules to support LRP for the Caffe deep learning framework (Jia et al., 2014) and operates on pre-trained neural network models. The functionality of the framework and a demo thereof is accessible via command line or an IDE of choice for Matlab, python and Caffe (C++). The Caffe version is based on the caffe-master branched on 3rd October 2015. The latest official release is available from `http://www.heatmapping.org`.

We provide three implementations due to the different strengths and advantages of the three implementations. Caffe is an established toolbox for neural networks, written in C++, and has the big advantage that it comes with many pretrained neural networks in the Caffe Model Zoo. The Caffe version works out of the box with the BVLC reference, the

GoogLeNet model and the VGG CNN models from Chatfield et al. (2014). On the other hand, many scientists are unfamiliar with programming C++ interfaces, thus creating a barrier for experimentation when one is interested in modifying and trying out different LRP implementations. Note that LRP is a principle which permits multiple solutions. The python and Matlab implementations allow an easier access and for easy modification of LRP rules, in particular because python and Matlab are popular among machine learners and are taught in many undergrad university curricula. Furthermore the python and Matlab implementations require less external libraries which is practical in restricted setups where the user does not have full control over the installed packages. These trade-offs lead us to the choice of implementing LRP in three ways.

2.1 Platforms and Requirements

The python and Matlab implementations are available for systems running Linux, Windows and OSX due to the platform agnostic nature of python and Matlab. Caffe runs on Linux and OSX, however Caffe ports for Windows became available very recently on github (e.g. `https://github.com/happynear/caffe-windows`). Known and successfully tested minimum package requirements are available in the toolbox manual.

2.2 User Interface

The example implementations of the LRP pipeline can be executed and modified with the provided files `matlab/lrp_demo.m` and `python/lrp_demo.py` by navigating to the respective folders and executing the scripts using the chosen language's interpreter, e.g. for unix-based systems

```
cd <toolbox_location>/matlab
matlab -nodesktop -r lrp_demo
```

or

```
cd <toolbox_location>/python
python lrp_demo.py
```

The C++-based Caffe implementation of LRP uses a configuration file with its settings being explained in the manual. It takes as further input a text file which contains in each line the path to one image and an integer denoting the class for which the heatmap is to be computed. The third input is a path prefix.

```
cd <toolbox_location>/caffe-master-lrp/demonstrator
./lrp_demo ./config_sequential.txt ./testfilelist.txt ./
```

will then compute heatmaps for the images listed in `./testfilelist.txt`.

2.3 Documentation and Examples

The user manual guides through the installation and use of the toolbox. It also explains the LRP algorithm in detail and instructs on how to compute and interpret the results.

2.4 Developer Access, Licensing and Availability

The source code is provided under FreeBSD (2-Clause) License and is available in a separate archive for each released version. The latest official release of the toolbox code is available from `http://heatmapping.org`. More recent and work-in-progress versions can be found at `https://github.com/sebastian-lapuschkin/lrp_toolbox`.

3. Conclusion

The presented LRP package provides a simple and easy to use implementation that allows a user to explore LRP in Matlab and python or run more involved applications within the Caffe framework. We would like to emphasize that the user can readily take an existing network to apply the LRP procedure. In other words also DNNs from past projects could be retrospectively explained by LRP. Furthermore note that the usage of LRP is not limited to DNNs, in fact, as shown in (Bach et al., 2015) also kernel methods and other learning machines, e.g. using Bag of Words representations can yield heatmaps through LRP. Future work will use the LRP framework and software package in the sciences and for general applications that require explanation.

Acknowledgments

The work was funded in part by the Federal Ministry for Education and Research (BMBF) under Grant 01IS14013A-E and Grant 01GQ1115, as well as by the Deutsche Forschungsgesellschaft (DFG) under Grant MU 987/19-1 and the Brain Korea 21 Plus Program by the Korean Government. Correspondence to KRM and WS.

References

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140, 2015.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *JMLR*, 11: 1803–1831, 2010.

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR*, pages 3642–3649. IEEE, 2012a.

Dan C. Ciresan, Alessandro Giusti, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *NIPS*, pages 2852–2860, 2012b.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537, 2011.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors. *Neural networks: Tricks of the trade, reloaded*, volume 7700 of *LNCS*. Springer, 2nd edition, 2012.

Grégoire Montavon, Matthias Rupp, Vivekanand Gobre, Alvaro Vazquez-Mayagoitia, Katja Hansen, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *NJP*, 15 (9):095003, 2013.

Grégoire Montavon, Sebastian Bach, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *CoRR*, abs/1512.02479, 2015. URL `http://arxiv.org/abs/1512.02479`.

Peter M Rasmussen, Tanya Schmah, Kristoffer H Madsen, Torben E Lund, Stephen C Strother, and Lars K Hansen. Visualization of nonlinear classification models in neuroimaging. In *BIOSIGNALS*, 2012.

Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Bach, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *CoRR*, abs/1509.06321, 2015. URL `http://arxiv.org/abs/1509.06321`.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

Dong Yu and Li Deng. *Automatic speech recognition - a deep learning approach*. Springer, October 2014. URL `http://research.microsoft.com/apps/pubs/default.aspx?id=230891`.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014. doi: 10.1007/978-3-319-10590-1_53.

Journal of Machine Learning Research 17 (2016) 1-21 Submitted 1/16; Revised 5/16; Published 6/16

Equivalence of Graphical Lasso and Thresholding for Sparse Graphs

Somayeh Sojoudi, SOJOUDI@BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

Editor: Michael Mahoney

Abstract

This paper is concerned with the problem of finding a sparse graph capturing the conditional dependence between the entries of a Gaussian random vector, where the only available information is a sample correlation matrix. A popular approach to address this problem is the graphical lasso technique, which employs a sparsity-promoting regularization term. This paper derives a simple condition under which the computationally-expensive graphical lasso behaves the same as the simple heuristic method of thresholding. This condition depends only on the solution of graphical lasso and makes no direct use of the sample correlation matrix or the regularization coefficient. It is proved that this condition is always satisfied if the solution of graphical lasso is close to its first-order Taylor approximation or equivalently the regularization term is relatively large. This condition is tested on several random problems, and it is shown that graphical lasso and the thresholding method lead to highly similar results in the case where a sparse graph is sought. We also conduct two case studies on brain connectivity networks of twenty subjects based on fMRI data and the topology identification of electrical circuits to support the findings of this work on the similarity of graphical lasso and thresholding.

Keywords: Graphical Lasso, Graphical Models, Sparse Graphs, Brain Connectivity Networks, Electrical Circuits

1. Introduction

In recent years, there has been a growing interest in developing techniques for estimating sparse undirected graphical models (Banerjee et al., 2008; Bruckstein et al., 2009; Chandrasekaran et al., 2010; Goldstein and Osher, 2009; Jalali et al., 2011; Schmidt et al., 2007). Finding sparse solutions have become essential to many applications, including signal processing, pattern recognition, and data mining. Many applications use L_1-regularized models such as the Lasso (Tibshirani, 1996). L_1 regularization aims to find sparse solutions, which is especially useful for high-dimensional problems with a large number of features (Bühlmann and Van De Geer, 2011; Fan and Lv, 2010; Meinshausen and Yu, 2009; Richtárik and Takáč, 2012; Wright et al., 2009; Zhang and Huang, 2008). Although Lasso-type algorithms are shown to be effective in recovering sparse solutions, they are computationally expensive for large-scale problems. In this work, we derive a simple condition under which the computationally-expensive graphical lasso behaves the same as the simple heuristic method of thresholding.

Consider a random vector $\mathbf{x} = (x_1, x_2, ..., x_n)$ with a multivariate normal distribution. Let $\Sigma_* \in \mathbb{S}^n_+$ denote the correlation matrix associated with the vector $\mathbf{x}$. The inverse of the correlation matrix can be used to determine the conditional independence between the random variables $x_1, x_2, ..., x_n$. In particular, if the $(i,j)^{\text{th}}$ entry of Σ_*^{-1} is zero, then x_i and x_j are conditionally independent. The graph $\mathcal{G}(\Sigma_*^{-1})$ (i.e., the sparsity graph of Σ_*^{-1}) represents a graphical model capturing the conditional independence between the elements of $\mathbf{x}$. Assume that $\mathcal{G}(\Sigma_*^{-1})$ is a sparse graph. Finding this graph is nontrivial in practice because the exact correlation matrix Σ_* is rarely known. More precisely, $\mathcal{G}(\Sigma_*^{-1})$ should be constructed from a given sample correlation matrix as opposed to Σ_*. Let Σ denote an arbitrary $n \times n$ positive semidefinite matrix, which is provided as an estimate of Σ_*. In this paper, we do not impose any assumption on the error $\|\Sigma - \Sigma_*\|$. Consider the convex optimization problem

$$\underset{S \in \mathbb{S}^n_+}{\text{minimize}} \quad -\log\det(S) + \text{trace}(\Sigma S) \tag{1}$$

where $\mathbb{S}^n_+$ denotes the set of $n \times n$ positive semidefinite matrices. It is easy to verify that the optimal solution of the above problem is $S^{\text{opt}} = \Sigma^{-1}$. Hence, S^{opt} aims to estimate Σ_*^{-1}. On the other hand, although the inverse of Σ_* is assumed to be a sparse graph, a small perturbation of Σ_* would make its inverse a dense graph. This implies that the sparsity graph of S^{opt}, denoted as $\mathcal{G}(S^{\text{opt}})$, may not resemble the graphical model $\mathcal{G}(\Sigma_*^{-1})$ in general. Hence, the optimization problem (1) needs to be modified to indirectly enforce some sparsity on its solution. To this end, consider the problem

$$\underset{S \in \mathbb{S}^n_+}{\text{minimize}} \quad -\log\det(S) + \text{trace}(\Sigma S) + \lambda\|S\|_1 \tag{2}$$

where $\lambda \in \mathbb{R}_+$ is a regularization parameter and $\|S\|_1$ is defined as $\sum_{i=1}^n \sum_{j=1}^n |S_{ij}|$. This problem is referred to as *graphical lasso* (Banerjee et al., 2008; Friedman et al., 2008; Yuan and Lin, 2007). Intuitively, the penalty term $\lambda\|S\|_1$ with a large λ aims to decrease the off-diagonal entries of S in magnitude and enforce most of them to be zero. Henceforth, the notation S^{opt} is used to denote a solution of the graphical lasso instead of the unregularized optimization problem (1). There is a large body of literature suggesting that $\mathcal{G}(S^{\text{opt}})$ is a good estimate of the graphical model $\mathcal{G}(\Sigma_*^{-1})$ for a suitable choice of λ (Banerjee et al., 2008; Danaher et al., 2014; Friedman et al., 2008; Krämer et al., 2009; Liu et al., 2010; Witten et al., 2011; Yuan and Lin, 2007). Note that although graphical lasso is motivated by multivariate normal random variables, its application is beyond this class of random variables and it applies to all problems for which a *sparse* inverse correlation matrix is sought.

Suppose that it is known *a priori* that the true graph $\mathcal{G}(\Sigma_*^{-1})$ has k edges, for some given number k. Assume that the nonzero entries of the upper triangular part of Σ (excluding its diagonal) have different magnitudes (this assumption is satisfied both generically and under an infinitesimal perturbation of the nonzero entries of Σ). Two heuristic methods for finding an approximation of $\mathcal{G}(\Sigma_*^{-1})$ are as follows:

- *Graphical Lasso:* We solve the optimization problem (2) repeatedly for different values of λ until a solution S^{opt} with exactly $2k$ nonzero off-diagonal entries are found. Note

that λ can be updated in the optimization problem using the bisection technique (Theorem 3 in Fattahi and Lavaei (2016) guarantees the existence of an appropriate interval for λ under generic conditions).

- *Thresholding:* Without solving any optimization problem, we simply identify those $2k$ entries of Σ that have the largest magnitudes among all off-diagonal entries of Σ. We then replace the remaining $n^2 - n - 2k$ off-diagonal entries of Σ with zero and denote the resulting matrix as Σ_k. Note that Σ and Σ_k have the same diagonal. Finally, we consider the sparsity graph of Σ_k, namely $\mathcal{G}(\Sigma_k)$, as an estimate of $\mathcal{G}\big(\Sigma_*^{-1}\big)$.

The connection between graphical lasso and the thresholding technique is not well understood and there are only a few studies on this subject. The work Mazumder and Hastie (2012) has recently shown that if the sample covariance matrix is thresholded at λ and its corresponding graph is decomposed into connected components, then the vertex-partition induced by these components is equal to the one induced by the connected components of the estimated graph obtained from graphical lasso for the same λ. The paper Guillot and Rajaratnam (2011) obtains graph conditions that are required for preserving the positive definiteness of the sample correlation matrix after thresholding.

This paper is focused on the investigation of the connection between graphical lasso and thresholding. First, we derive a condition under which the heuristic thresholding method performs very similarly to the computationally-heavy graphical lasso. We then argue that this condition is satisfied as long as λ is large enough. Moreover, we demonstrate in numerical examples that graphical lasso and thresholding lead to the same approximate graph for $\mathcal{G}\big(\Sigma_*^{-1}\big)$. Note that although the condition provided here depends on the solution of graphical lasso, it can be systematically expressed in terms of the sample correlation matrix Σ for certain types of graphs (see the technical report Sojoudi (2016) for the derivation of such conditions for acyclic graphs).

Recently, there has been a significant interest in studying the human brain functional connectivity networks using functional MRI (fMRI) data. Functional connectivity is measured as the temporal coherence or correlation between the activities of disjoint brain areas, where the direct statistical dependence between every two brain regions in the functional network can be obtained using partial correlation. In most fMRI studies, computing partial correlations is a daunting challenge due to the limitation on the number samples available from fMRI scans. Graphical lasso has become popular in the literature for the identification of the direct correlations between the activities of different parts of the brain using a small number of samples (Huang et al., 2010). In this work, we apply graphical lasso and thresholding to the resting-state fMRI data collected from twenty subjects and observe a high degree of similarity between the outcomes of these two techniques for each individual subject. Note that the matrix Σ is not invertible for the fMRI study conducted here. More precisely, this work makes no assumption on the invertibility of the sample correlation matrix Σ (although the true correlation matrix Σ_* needs to be invertible).

In Sojoudi and Doyle (2014), we have developed a new method for generating synthetic data based on sparse electrical circuit models, where certain nodes of each circuit are connected to one another through resistors and capacitors that are subject to thermal noise (to create stochasticity). The main property of this model is that the connectivity of the circuit model can be obtained from the sparsity pattern of the inverse covariance matrix associated with the nodal voltages. In other words, the sparsity pattern of the inverse covariance matrix has the same structure as the adjacency matrix of the circuit network. In this work, we use the above circuit model to verify the high similarity between graphical lasso and the thresholding technique for electrical networks.

This paper is organized as follows. The main results are presented in Section 2. Simulations on random systems are provided in Section 3. Two case studies on fMRI data and electrical circuits are conducted in Sections 4 and 5, respectively. Some concluding remarks are drawn in Section 6.

1.1 Notations and Definitions

Notations: $\mathbb{R}$, $\mathbb{S}^n$, and $\mathbb{S}^n_+$ denote the sets of real numbers, $n \times n$ (real) symmetric matrices, and $n \times n$ positive semidefinite matrices, respectively. $\text{trace}\{M\}$ and $\log\det\{M\}$ denote the trace and the logarithm of the determinant of a matrix M. The (i,j) entry of M is shown as M_{ij}. The notations $|x|$ and $\|M\|_1$ represent the absolute value of a scalar x and the sum of absolute values of a matrix M, respectively. The symbol $\text{sign}(\cdot)$ denotes the sign function (note that $\text{sign}(0) = 0$). The standard basis vectors in $\mathbb{R}^n$ are shown as $e_1, e_2, ..., e_n$. The optimal value of a matrix variable M is denoted as M^{opt}.

Definition 1 *Given a symmetric matrix $S \in \mathbb{S}^n$, the support (sparsity) graph of S is defined as a graph with the vertex set $\mathcal{V} := \{1, 2, ..., n\}$ and the edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ such that $(i,j) \in \mathcal{V}$ if and only if $S_{ij} \neq 0$, for every two different vertices $i, j \in \mathcal{V}$. The support graph of S captures the sparsity of the matrix S and is denoted as $\mathcal{G}(S)$.*

Definition 2 *Given two graphs $\mathcal{G}$ and $\mathcal{G}'$ with the same vertex set, define $\mathcal{G} - \mathcal{G}'$ as a graph obtained from $\mathcal{G}$ by removing the common edges of $\mathcal{G}$ and $\mathcal{G}'$.*

2. Main Results

In this section, we study the connection between thresholding and graphical lasso. To simplify the presentation, we assume that the nonzero entries of the upper triangular part of Σ (excluding its diagonal) have different magnitudes. Assume also that the number of such nonzero entries is greater than or equal to k (in order to guarantee that the thresholding method is able to obtain a graph with k edges). For notational convenience, we denote the $(i,j)^{\text{th}}$ entry of the matrix $(S^{\text{opt}})^{-1}$ as $(S^{\text{opt}})^{-1}_{ij}$ throughout this work.

2.1 Optimality Conditions

Consider the convex optimization problem (2) together with an optimal solution S^{opt}. First, we aim to obtain necessary and sufficient optimality conditions for graphical lasso.

Lemma 3 *S^{opt} is an optimal solution of graphical lasso if and only if the conditions*

$$(S^{opt})^{-1}_{ij} = \Sigma_{ij} + \lambda \qquad \text{if} \quad i = j \tag{3a}$$
$$(S^{opt})^{-1}_{ij} = \Sigma_{ij} + \lambda \times \text{sign}(S^{opt}_{ij}) \qquad \text{if} \quad S^{opt}_{ij} \neq 0 \tag{3b}$$
$$(S^{opt})^{-1}_{ij} \leq \Sigma_{ij} + \lambda \qquad \text{if} \quad S^{opt}_{ij} = 0 \tag{3c}$$
$$(S^{opt})^{-1}_{ij} \geq \Sigma_{ij} - \lambda \qquad \text{if} \quad S^{opt}_{ij} = 0 \tag{3d}$$

are satisfied for all indices $i, j \in \{1, ..., n\}$.

Proof Due to the convexity of graphical lasso, a locally optimal solution of this problem is a global solution and, therefore, a local perturbation analysis can be used to prove the lemma. To this end, notice that $-\log(0) = +\infty$, $\text{trace}(\Sigma S) \geq 0$, and $\|S\|_1$ is finite only when all entries of S are finite. It follows from these properties that S^{opt} has bounded entries and a nonzero determinant, i.e., $S^{\text{opt}} \succ 0$ (note that a zero determinant for S^{opt} makes the objective function of graphical lasso equal to $+\infty$). This means that $S^{\text{opt}}(\varepsilon; i, j)$ defined as $S^{\text{opt}} + \varepsilon(e_i e_j^T + e_j e_i^T)$ is a feasible solution of the optimization problem (2) for small values of ε (note that the matrix $e_i e_j^T + e_j e_i^T$ is sign indefinite). On the other hand, for every $i, j \in \{1, ..., n\}$, one can write:

$$\begin{aligned} &\big[-\log\det(S^{\text{opt}}) + \text{trace}(\Sigma S^{\text{opt}}) + \lambda\|S^{\text{opt}}\|_1\big] \\ &- \big[-\log\det(S^{\text{opt}}(\varepsilon; i, j)) + \text{trace}(\Sigma S^{\text{opt}}(\varepsilon; i, j)) + \lambda\|S^{\text{opt}}(\varepsilon; i, j)\|_1\big] \\ &\qquad = 2\left((S^{\text{opt}})^{-1}_{ij} - \Sigma_{ij}\right)\varepsilon + 2\left(|S^{\text{opt}}_{ij}| - |S^{\text{opt}}_{ij} + \varepsilon|\right)\lambda + \tau\varepsilon^2 + O(\varepsilon^3) \end{aligned} \tag{4}$$

where τ is a positive number due to the strict convexity of the objective of the optimization problem (2). The above equation is derived based on the Taylor series expansion of $\log\det(\cdot)$ and the fact that the derivative of $\log\det(S)$ with respect to S is equal to S^{-1}. Now, recall that S^{opt} is an optimal solution of (2), whereas $S^{\text{opt}} + \varepsilon(e_i e_j^T + e_j e_i^T)$ is only a feasible solution. Hence, the left side of the equality (4) must always be non-positive for all sufficiently small values of ε. A simple analysis of this equation leads to the conditions provided in (3). ■

Lemma 3 offers a set of necessary and sufficient conditions for the matrix S^{opt} to be an optimal solution of graphical lasso. These optimality conditions can be summarized as:

- The diagonal of $(S^{\text{opt}})^{-1}$ is obtained from that of Σ after a shift by the number λ.
- Each off-diagonal element (i, j) of the matrix $(S^{\text{opt}})^{-1}$ is in the interval $[\Sigma_{ij} - \lambda, \Sigma_{ij} + \lambda]$, and is located at one of the endpoints of this interval if (i, j) is an edge of the graphical model $\mathcal{G}(S^{\text{opt}})$.

2.2 First Condition for Equivalence

In this part, we derive a condition under which graphical lasso and thresholding result in the same estimate graphical model for the random vector $\mathbf{x}$.

Definition 4 *A symmetric matrix $\hat{S}$ is said to be equivalent to S^{opt} if $\hat{S}$ can be obtained from S^{opt} through two operations: (i) permutation of its off-diagonal entries, and (ii) flipping the sign of some of its off-diagonal entries. We use the notation $\hat{S} \sim S^{opt}$ to show this equivalence.*

Theorem 5 *Let k denote the number of edges of the graph $\mathcal{G}(S^{opt})$. Consider the optimization problems*

$$\underset{S \in \mathbb{S}^n}{\text{minimize}} \quad \text{trace}(\Sigma S) + \lambda\|S\|_1 \quad \text{subject to} \quad \text{S} \sim \text{S}^{opt} \tag{5}$$

and

$$\underset{S \in \mathbb{S}^n_+}{\text{minimize}} \quad -\log\det(S) + \text{trace}(\Sigma S) + \lambda\|S\|_1 \quad \text{subject to} \quad \text{S} \sim \text{S}^{opt} \tag{6}$$

If these two problems possess the same solution, then S^{opt} and Σ_k will have the same support graph.

Proof Let $\hat{S}^{\text{opt}}$ denote an optimal solution of the optimization problem (5). Our first goal is to show that $\mathcal{G}(\hat{S}^{\text{opt}}) = \mathcal{G}(\Sigma_k)$. To this end, notice that every feasible solution S of (5) satisfies the equality $\|S\|_1 = \|S^{\text{opt}}\|_1$ due to Definition 4. This implies that the additive term $\lambda\|S\|_1$ can be eliminated from the objective function of the optimization problem (5). On the other hand, the first part of the objective function can be expressed as $\text{trace}(\Sigma S) = \sum_{i,j=1}^{n} (\Sigma_{ij} S_{ij})$. By investigating this sum, it is straightforward to show that the optimization problem (5) has a unique solution $\hat{S}^{\text{opt}}$ that can be obtained as follows:

- First, we focus on the upper triangular part of Σ (excluding the diagonal) and identify those k entries with the largest absolute values, which are denoted as $\Sigma_{i_1 j_1}, \Sigma_{i_2 j_2}, ...,$ $\Sigma_{i_k j_k}$ such that
$$|\Sigma_{i_1 j_1}| > |\Sigma_{i_2 j_2}| > ... > |\Sigma_{i_k j_k}| \tag{7}$$
(note that the nonzero entries of the upper triangular part of Σ have different magnitudes, by assumption).
- Second, we repeat the above procedure on the matrix S^{opt} and identify those k entries with the greatest absolute values, which are denoted as $(p_1, q_1), ..., (p_k, q_k)$ such that
$$|S^{\text{opt}}_{p_1 q_1}| \geq |S^{\text{opt}}_{p_2 q_2}| \geq ... \geq |S^{\text{opt}}_{p_k q_k}| \tag{8}$$
- For every $l \in \{1, ..., k\}$, the $(i_l, j_l)^{\text{th}}$ entry of $\hat{S}^{\text{opt}}$ is equal to -sign$(\Sigma_{i_l j_l})|S^{\text{opt}}_{p_l q_l}|$.

Now, it is easy to verify that

$$\mathcal{G}(\hat{S}^{\text{opt}}) = \mathcal{G}(\Sigma_k) \tag{9}$$

On the other hand, since the objective of the optimization problem (2) is strictly convex, $\mathcal{S}^{\text{opt}}$ is a unique solution of this problem. Hence, due to the fact the feasible set of the problem (6) is contained in that of (2), the optimization problem (6) has the unique solution $\mathcal{S}^{\text{opt}}$. Therefore, the two solutions $\hat{S}^{\text{opt}}$ and $\mathcal{S}^{\text{opt}}$ of the problem (6) are identical. Now, the proof follows from the relation (9). ■

As verified by the author, the condition given in Theorem 5 is satisfied for many numerical examples, leading to the equivalence of thresholding and graphical lasso. The main intuition behind the satisfaction of the above condition is as follows:

- Consider a small number k (or a large number λ) for which the matrix S^{opt} is highly sparse.
- Due to Lemma 3, the diagonal entries of S^{opt} would be relatively much larger than the nonzero off-diagonal entries of S^{opt}.
- Hence, the permutation of the (small) off-diagonal entries of the positive semidefinite S^{opt} would not make the matrix sign indefinite and also has a negligible effect on the $\log\det$ of the matrix.
- Under such circumstances, the condition derived in Theorem 5 would be satisfied (note that (5) is obtained from (6) by dropping the sign-definite condition and the $\log\det$ term).

To strengthen the above argument, an easy-to-check condition will be provided next to guarantee the equivalence of thresholding and graphical lasso.

2.3 Second Condition for Equivalence

Consider the solution S^{opt}. The objective of this part is to derive a condition of equivalence that depends only on the entries of S^{opt}, without using λ or Σ explicitly. Recall that this equivalence does not requite that the matrices S^{opt} and Σ_k be identical (which is unlikely to occur in practice), and is only concerned with the sparsity patterns of these matrices.

Theorem 6 *Let k denote the number of edges of the graph $\mathcal{G}(S^{opt})$. Assume that the inequalities*

$$\text{sign}\left(S^{opt}_{ij}\right) \times \text{sign}\left((S^{opt})^{-1}_{ij}\right) \leq 0 \tag{10a}$$

$$|(S^{opt})^{-1}_{ij}| \geq |(S^{opt})^{-1}_{pq}| \tag{10b}$$

hold for every two pairs (i,j) and (p,q) satisfying

$$(i,j) \in \mathcal{G}(S^{opt}) \tag{11a}$$

$$(p,q) \in \mathcal{G}((S^{opt})^{-1}) - \mathcal{G}(S^{opt}) \tag{11b}$$

Then, graphical lasso and thresholding produce the same graph, i.e., $\mathcal{G}(S^{opt}) = \mathcal{G}(\Sigma_k)$.

Proof Consider two arbitrary pairs (i,j) and (p,q) satisfying (11). It follows from (10) and Lemma 3 that

$$|\Sigma_{ij}| \geq \lambda \tag{12}$$

and that

$$|\Sigma_{ij}| - \lambda \geq |(S^{\text{opt}})^{-1}_{pq}| \tag{13}$$

Similar to the proof of Theorem 5, let the entries of the upper triangular part of Σ_k be ordered as

$$|\Sigma_{i_1 j_1}| > ... > |\Sigma_{i_k j_k}| > |\Sigma_{i_{k+1} j_{k+1}}| \geq \cdots \geq |\Sigma_{i_m j_m}| \tag{14}$$

where $m = \frac{n^2-n}{2}$. To prove the theorem by contradiction, assume that $\mathcal{G}(S^{\text{opt}}) \neq \mathcal{G}(\Sigma_k)$. Recall that $\mathcal{G}(\Sigma_k)$ is a graph with n vertices and the edges $(i_1,j_1),(i_2,j_2),...,(i_k,j_k)$. Since $\mathcal{G}(S^{\text{opt}})$ has exactly k edges, the above assumption implies that there exist two numbers s and r such that

$$s \in \{1,...,k\} \qquad \text{and} \qquad (i_s,j_s) \notin \mathcal{G}(S^{\text{opt}}) \tag{15a}$$

$$r \in \{s+1,...,m\} \qquad \text{and} \qquad (i_r,j_r) \in \mathcal{G}(S^{\text{opt}}) \tag{15b}$$

Therefore, it follows from (13) that

$$|(S^{\text{opt}})^{-1}_{i_s j_s}| \leq |\Sigma_{i_r j_r}| - \lambda < |\Sigma_{i_s j_s}| - \lambda \tag{16}$$

However, the inequality

$$|(S^{\text{opt}})^{-1}_{i_s j_s}| < |\Sigma_{i_s j_s}| - \lambda \tag{17}$$

is in contradiction with the relation

$$(S^{\text{opt}})^{-1}_{i_s j_s} \in [\Sigma_{i_s j_s} - \lambda, \Sigma_{i_2 j_s} + \lambda] \tag{18}$$

that is given in (3). This completes the proof. ■

Theorem 6 states that graphical lasso and thresholding are equivalent if two conditions are satisfied:

- *Condition 1:* The sign of every nonzero off-diagonal entry of S^{opt} is different from that of its corresponding entry in the inverse of S^{opt}.
- *Condition 2:* If a zero off-diagonal entry of S^{opt} takes a nonzero value in the inverse of S^{opt}, then its magnitude is not larger than the magnitude of any off-diagonal element of $(S^{\text{opt}})^{-1}$ corresponding to a nonzero entry of S^{opt}.

Note that the above conditions only depend on S^{opt} and are not directly related to Σ. To better understand these conditions, we decompose S^{opt} as

$$S^{\text{opt}} = D^{\text{opt}} + O^{\text{opt}} \tag{19}$$

where D^{opt} is a diagonal matrix and O^{opt} has a zero diagonal. If the norm of $(D^{\text{opt}})^{-1}O^{\text{opt}}$ is less than 1, one can write

$$(S^{\text{opt}})^{-1} = \left(\sum_{t=0}^{\infty}\left(-(D^{\text{opt}})^{-1}O^{\text{opt}}\right)^t\right)(D^{\text{opt}})^{-1} \tag{20}$$

In general, we have

$$(S^{\text{opt}})^{-1} = (D^{\text{opt}})^{-1} - (D^{\text{opt}})^{-1}O^{\text{opt}}(D^{\text{opt}})^{-1} + h.o.t. \tag{21}$$

where $h.o.t$ stands for *higher order terms* in the Taylor series expansion if the norm of $(D^{\rm opt})^{-1}O^{\rm opt}$ is less than 1, and otherwise is equal to $(S^{\rm opt})^{-1}O^{\rm opt}(D^{\rm opt})^{-1}O^{\rm opt}(D^{\rm opt})^{-1}$ (as a general formula). We refer to

$$E^{\rm opt} := (D^{\rm opt})^{-1} - (D^{\rm opt})^{-1}O^{\rm opt}(D^{\rm opt})^{-1} \tag{22}$$

as the first-order approximation of $(S^{\rm opt})^{-1}$. Note that if λ is relatively large, it is expected that $O^{\rm opt}$ will be small compared to $D^{\rm opt}$, which will lead to small higher order terms.

Theorem 7 *The condition* (10) *given in Theorem 6 to guarantee the equivalence of graphical lasso and thresholding is satisfied if $(S^{opt})^{-1}$ is replaced by its first-order approximation E^{opt} in the condition.*

Proof Equation (22) yields that

$$E^{\rm opt}_{ij} = \begin{cases} (D^{\rm opt})^{-1}_{ii} & \text{if} \quad i=j \\ -(D^{\rm opt})^{-1}_{ii}O^{\rm opt}_{ij}(D^{\rm opt})^{-1}_{jj} & \text{if} \quad (i,j)\in\mathcal{G}(S^{\rm opt}) \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

for every $i,j\in\{1,...,n\}$. Note that $D^{\rm opt}>0$ due to the positive definiteness of $S^{\rm opt}$. Hence, given a pair $(i,j)\in\mathcal{G}(S^{\rm opt})$, one can write:

$$\text{sign}\left(S^{\rm opt}_{ij}\right)\text{sign}\left(E^{\rm opt}_{ij}\right) = -\text{sign}\left(O^{\rm opt}_{ij}\right)^2(D^{\rm opt})^{-1}_{ii}(D^{\rm opt})^{-1}_{jj} \leq 0 \tag{24}$$

Moreover, given arbitrary pairs (i,j) and (p,q) satisfying (11), it follows from (23) that $E^{\rm opt}_{pq}=0$. Therefore,

$$|E^{\rm opt}_{ij}| \geq |E^{\rm opt}_{pq}| \tag{25}$$

The proof is completed by combining (24) and (25). ■

Before further simplifying the conditions of Theorem 6 based on Theorem 7, it is desirable to offer a more general condition measuring the "similarity" of graphical lasso and thresholding (as opposed to their "equivalence").

Definition 8 *Define $\mathbb{I}_k$ as the set of indices (locations) of those k entries of the upper triangular part of $(\mathcal{S}^{opt})^{-1}$ that have the largest magnitudes.*

Note that if multiple entries of $(\mathcal{S}^{\rm opt})^{-1}$ have the same value, then $\mathbb{I}_k$ may not be uniquely defined. In that case, $\mathbb{I}_k$ can be considered as any of the sets satisfying the properties given in Definition 8.

Theorem 9 *Let k denote the number of edges of the graph $\mathcal{G}(S^{opt})$, and h be the number of indices (i,j)'s in the set $\mathbb{I}_k$ for which the relation*

$$\text{sign}\left(S^{opt}_{ij}\right)\times\text{sign}\left((S^{opt})^{-1}_{ij}\right) < 0 \tag{26}$$

holds. Then, the graphs $\mathcal{G}(\Sigma_k)$ and $\mathcal{G}(\mathcal{S}^{opt})$ have at least h edges in common. In particular, graphical lasso and thresholding are equivalent if $h=k$.

Proof The proof of Theorem 6 can be adopted to prove this theorem. The details are omitted for brevity. ■

Theorem 9 states that the graph $\mathcal{G}(\mathcal{S}^{\rm opt})$ (obtained using graphical lasso) and the graph $\mathcal{G}(\Sigma_k)$ (obtained using thresholding) share at least h edges. Hence, this theorem provides a simple mechanism to check the similarity between graphical lasso and thresholding through a simple test on $\mathcal{S}^{\rm opt}$. This mechanism will be further studied below.

Definition 10 *Given a matrix $H\in\mathbb{S}^n$ and a positive integer $t\in\{1,2,...,k\}$, define $f(H,t)$ as the magnitude of the t^{th} largest entry (in magnitude) of the upper triangular part of H (excluding its diagonal). For example, $f(H,1)$ is equal to the absolute value of an off-diagonal entry of H with the largest magnitude.*

We define $\Delta E^{\rm opt}$ as the difference between the matrix $(S^{\rm opt})^{-1}$ and its first-order approximation $E^{\rm opt}$.

Theorem 11 *Given a positive integer $t\in\{1,2,...,k\}$, the graphs $G(\Sigma_k)$ and $G(S^{opt})$ have at least t edges in common if*

$$2\ f(\Delta E^{opt},1) < f(E^{opt},t) \tag{27}$$

In particular, graphical lasso and thresholding lead to the same approximate graph if the above inequality is satisfied for $t=k$.

Proof The proof follows from Theorems 7 and 9. The main idea will be sketched for $t=k$ below. Consider arbitrary pairs (i,j) and (p,q) satisfying (11). It follows from (27) that

$$|E^{\rm opt}_{ij}| \geq f(E^{\rm opt},k) \geq 2f(\Delta E^{\rm opt},1) \geq\ 2|\Delta E^{\rm opt}_{ij}| \tag{28}$$

Similarly,

$$|E^{\rm opt}_{ij}| \geq f(E^{\rm opt},k) \geq 2f(\Delta E^{\rm opt},1) \geq\ 2|\Delta E^{\rm opt}_{pq}| \tag{29}$$

On the other hand, $(S^{\rm opt})^{-1}_{ij} = E^{\rm opt}_{ij} + \Delta E^{\rm opt}_{ij}$ and $\text{sign}(S^{\rm opt}_{ij}) = \text{sign}(O^{\rm opt}_{ij}) = -\text{sign}(E^{\rm opt}_{ij})$. The above relations together with (23) imply the condition (10). This completes the proof. ■

Roughly speaking, $f(\Delta E^{\rm opt},1)$ is small for a relatively large number λ, and $f(E^{\rm opt},t)$ would stay away from zero due to Lemma 3. Theorem 11 explains that the relationship between $E^{\rm opt}$ and $\Delta E^{\rm opt}$ determines the degree of similarity between graphical lasso and thresholding. Note that Theorem 11 is more conservative than Theorem 9, but its condition is more insightful.

Note that the definition of graphical lasso in (2) is based on the regularization term $\lambda\|S\|_1$. Consider a second version of graphical lasso where only the off-diagonal entries of S are penalized in the regularization term. This is realized by replacing the term $\|S\|_1$ with $2\sum_{i=1}^n\sum_{j=i+1}^n|S_{ij}|$. The optimality conditions given in Lemma 3 for graphical lasso are valid for the second version of graphical lasso after dropping λ from the right side of (3a). As a result, the original and second version of graphical lasso are very similar, and the only

$$\Sigma = \begin{bmatrix} 1 & 0.5448 & 0.4980 & 0.2045 & -0.2818 & -0.1452 \\ 0.5448 & 1 & -0.1327 & -0.0604 & -0.6860 & -0.0457 \\ 0.4980 & -0.1327 & 1 & 0.1283 & -0.1859 & -0.5174 \\ 0.2045 & -0.0604 & 0.1283 & 1 & 0.4019 & 0.6238 \\ -0.2818 & -0.6860 & -0.1859 & 0.4019 & 1 & 0.5139 \\ -0.1452 & -0.0457 & -0.5174 & 0.6238 & 0.5139 & 1 \end{bmatrix} \tag{30}$$

$$\mathcal{S}^{\text{opt}} = \begin{bmatrix} 0.6934 & -0.0453 & -0.0229 & 0 & 0 & 0 \\ -0.0453 & 0.7114 & 0 & 0 & 0.1153 & 0 \\ -0.0229 & 0 & 0.6919 & 0 & 0 & 0.0321 \\ 0 & 0 & 0 & 0.6997 & 0 & -0.0839 \\ 0 & 0.1153 & 0 & 0 & 0.7098 & -0.0305 \\ 0 & 0 & 0.0321 & -0.0839 & -0.0305 & 0.7025 \end{bmatrix} \tag{31}$$

$$(\mathcal{S}^{\text{opt}})^{-1} = \begin{bmatrix} 1.4500 & 0.0949 & 0.0480 & -0.0003 & -0.0155 & -0.0029 \\ 0.0949 & 1.4500 & 0.0036 & -0.0013 & -0.2360 & -0.0106 \\ 0.0480 & 0.0036 & 1.4500 & -0.0081 & -0.0035 & -0.0674 \\ -0.0003 & -0.0013 & -0.0081 & 1.4500 & 0.0077 & 0.1738 \\ -0.0155 & -0.2360 & -0.0035 & 0.0077 & 1.4500 & 0.0640 \\ -0.0029 & -0.0106 & -0.0674 & 0.1738 & 0.0640 & 1.4500 \end{bmatrix} \tag{32}$$

difference is in the diagonal of $(S^{\text{opt}})^{-1}$. It is easy to verify that the theorems developed in this work are all valid for the second version of graphical lasso as well. However, note that in order to obtain a graph with k edges, the value of the regularization term λ may not be the same for the original and the second version of graphical lasso.

3. Numerical Examples

Example 1: Consider Σ as the randomly generated matrix given in (30). The solution $\mathcal{S}^{\text{opt}}$ of graphical lasso with $\lambda = 0.45$ is provided in (31) (this value of λ guarantees that $\mathcal{G}(\mathcal{S}^{\text{opt}})$ will have n edges). It can be deduced from this solution that the graph $\mathcal{G}(\mathcal{S}^{\text{opt}})$ consists of the vertex set $\mathcal{V} := \{1, 2, ..., 6\}$ and the edge set $\mathcal{E} := \{(1,2), (1,3), (2,5), (3,6), (4,6), (5,6)\}$. On the other hand, it follows from a simple inspection of the matrix Σ that $\mathcal{E}$ coincides with the index set of the 6 largest absolute values of the upper triangular part of Σ. Hence, graphical lasso and thresholding are equivalent in this example, meaning that $\mathcal{G}(\mathcal{S}^{\text{opt}}) = \mathcal{G}(\Sigma_6)$.

It is desirable to find out whether the simple conditions proposed in Theorem 9 can be used to detect this equivalence. Recall that these conditions are expressed in terms of the matrix $\mathcal{S}^{\text{opt}}$, without using λ and Σ explicitly. To check these conditions, the matrix $(\mathcal{S}^{\text{opt}})^{-1}$ can be obtained as the one given in (32). Since the upper triangular part of $\mathcal{S}^{\text{opt}}$ has 6 nonzero entries, the index set $\mathbb{I}_6$ needs to be found. After the identification of the

$$\Sigma = \begin{bmatrix} 1 & 0.9342 & 0.8156 & 0.8609 & 0.6994 & 0.8457 \\ 0.9342 & 1 & 0.7110 & 0.7736 & 0.7283 & 0.8532 \\ 0.8156 & 0.7110 & 1 & 0.8593 & 0.8905 & 0.7958 \\ 0.8609 & 0.7736 & 0.8593 & 1 & 0.7876 & 0.7793 \\ 0.6994 & 0.7283 & 0.8905 & 0.7876 & 1 & 0.7040 \\ 0.8457 & 0.8532 & 0.7958 & 0.7793 & 0.7040 & 1 \end{bmatrix} \tag{35}$$

$$\Sigma_5 = \begin{bmatrix} 1 & 0.9342 & 0 & 0.8609 & 0 & 0 \\ 0.9342 & 1 & 0 & 0 & 0 & 0.8532 \\ 0 & 0 & 1 & 0.8593 & 0.8905 & 0 \\ 0.8609 & 0 & 0.8593 & 1 & 0 & 0 \\ 0 & 0 & 0.8905 & 0 & 1 & 0 \\ 0 & 0.8532 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{36}$$

$$\mathcal{S}^{\text{opt}} = \begin{bmatrix} 0.5417 & -0.0247 & 0 & -0.0032 & 0 & 0 \\ -0.0247 & 0.5417 & 0 & 0 & 0 & -0.0009 \\ 0 & 0 & 0.5408 & -0.0027 & -0.0118 & 0 \\ -0.0032 & 0 & -0.0027 & 0.5406 & 0 & 0 \\ 0 & 0 & -0.0118 & 0 & 0.5408 & 0 \\ 0 & -0.0009 & 0 & 0 & 0 & 0.5405 \end{bmatrix} \tag{37}$$

$$(\mathcal{S}^{\text{opt}})^{-1} = \begin{bmatrix} 1.8500 & 0.0842 & 0.0001 & 0.0109 & 0.0000 & 0.0002 \\ 0.0842 & 1.8500 & 0.0000 & 0.0005 & 0.0000 & 0.0032 \\ 0.0001 & 0.0000 & 1.8500 & 0.0093 & 0.0405 & 0.0000 \\ 0.0109 & 0.0005 & 0.0093 & 1.8500 & 0.0002 & 0.0000 \\ 0.0000 & 0.0000 & 0.0405 & 0.0002 & 1.8500 & 0.0000 \\ 0.0002 & 0.0032 & 0.0000 & 0.0000 & 0.0000 & 1.8500 \end{bmatrix} \tag{38}$$

largest absolute values of the matrix $(\mathcal{S}^{\text{opt}})^{-1}$, it turns out that

$$\mathbb{I}_6 = \{(1,2), (1,3), (2,5), (3,6), (4,6), (5,6)\} \tag{33}$$

In addition, the relation

$$\text{sign}\left(S^{\text{opt}}_{ij}\right) \times \text{sign}\left((S^{\text{opt}})^{-1}_{ij}\right) < 0, \quad \forall (i,j) \in \mathbb{I}_6 \tag{34}$$

holds. Therefore, it follows from Theorem 9 that graphical lasso and thresholding lead to the same result.

Example 2: Consider the randomly generated matrix in (35). Thresholding this matrix at the level of $k = 5$ yields the solution given in (36). On the other hand, solving the graphical

lasso for $\lambda = 0.85$ leads to the solution $\mathcal{S}^{\text{opt}}$ provided in (37), with the inverse given in (38) (this value of λ guarantees that $\mathcal{G}(\mathcal{S}^{\text{opt}})$ will have $n-1$ edges). By analyzing the matrix $\mathcal{S}^{\text{opt}}$ and its inverse, it can be verified that the conditions provided in Theorem 9 are satisfied. Hence, thresholding and graphical lasso are equivalent. An interesting observation is that the matrix Σ has two close entries 0.8457 and 0.8532 such that only one of them is removed through thresholding, but graphical lasso recognizes this fact and selects the entry with a slightly higher value. In other words, even in the case where the matrix Σ have entries with similar values (which makes it hard to decide what entries should be included in the graphical model), graphical lasso may still behave the same as thresholding.

Example 3: We construct a matrix NN^T, where the entries of $N \in \mathbb{R}^{30\times 30}$ are chosen at random according to some probably distribution. Define Σ as a matrix obtained from NN^T through a normalization to make its diagonal entries all equal to 1. We order the entries of Σ according to (14) and consider λ as $\frac{|\Sigma_{i_{30}j_{30}}|+|\Sigma_{i_{29}j_{29}}|}{2}$ (i.e., the average of the 29^{th} and 30^{th} largest absolute values of the upper triangular part of Σ). Two experiments will be concluded below.

Experiment I: By choosing every entry of the matrix N from a normal probability distribution, we generated 100 random matrices Σ's. In Figure 1(a), the number of edges of $\mathcal{G}(\mathcal{S}^{\text{opt}})$ is shown for each of the 100 trials (the trials are reordered to make the curve increasing). In Figure 1(b), the number of edges of the difference graph $\mathcal{G}(\mathcal{S}^{\text{opt}}) - \mathcal{G}(\Sigma_k)$ is depicted, where k is considered as the number of edges of $\mathcal{G}(\mathcal{S}^{\text{opt}})$. It can be seen that graphical lasso and thresholding are equivalent in more than 75 trials and are different by at most 2 edges in the remaining trials (note that the orderings of the trials for Figures 1(a) and 1(b) are different to ensure that each curve changes smoothly). This supports the claim of the paper that graphical lasso and thresholding would behave highly similarly. As opposed to computing the graph $\mathcal{G}(\mathcal{S}^{\text{opt}}) - \mathcal{G}(\Sigma_k)$ directly, Theorem 9 offers a simple insightful condition to find a subset of the common edges of $\mathcal{G}(\mathcal{S}^{\text{opt}})$ and $\mathcal{G}(\Sigma_k)$. This simple condition is tested on the 100 trials and the results are summarized in Figure 1(c). This figure shows the percentage of the common edges of $\mathcal{G}(\mathcal{S}^{\text{opt}})$ and $\mathcal{G}(\Sigma_k)$ that are detected by Theorem 9. It can be observed that the condition provided in the paper is able to detect a similarity degree on the order of 80% for more than 90% of the trials.

Experiment II: This study is the same as the previous experiment with the only difference that every entry of the matrix N was chosen from the interval $[0, 1]$ with a uniform probability distribution. The results are shown in Figures 1(d), (e) and (f). It can be seen that graphical lasso and thresholding are similar with the probability of at least 95%.

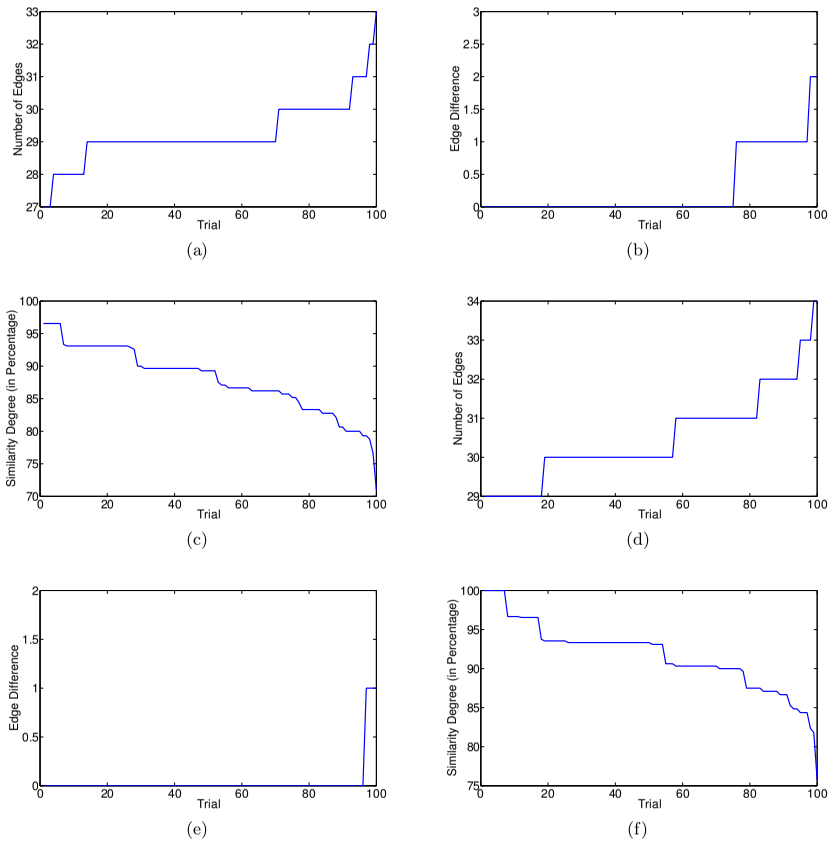


Figure 1: Figures (a), (b) and (c) show the number of edges of $\mathcal{G}(\mathcal{S}^{\text{opt}})$, the number of edges of $\mathcal{G}(\mathcal{S}^{\text{opt}}) - \mathcal{G}(\Sigma_k)$, and the similarity degree of thresholding and graphical lasso detected via Theorem 9 for Experiment I. Figures (d), (e) and (f) show the number of edges of $\mathcal{G}(\mathcal{S}^{\text{opt}})$, the number of edges of $\mathcal{G}(\mathcal{S}^{\text{opt}}) - \mathcal{G}(\Sigma_k)$, and the similarity degree of thresholding and graphical lasso detected via Theorem 9 for Experiment II.

4. Case Study on Brain Networks

Brain functional connectivity is defined as the statistical dependencies between the activities of disjoint brain regions. A brain functional network can be represented by a set of nodes and edges, where each node corresponds to a brain region and each edge shows the presence of correlated activities (namely, a nonzero partial correlation) between two disjoint regions. We apply graphical lasso and thresholding techniques to the fMRI data collected from twenty subjects to obtain their associated brain functional networks. These fMRI data sets are borrowed form Vértes et al. (2012). Each data set includes 134 samples of the low frequency oscillations, taken at 140 cortical brain regions in the right hemisphere. The goal is to find the brain functional connectivity network of each subject that specifies the

direct interactions (or conditional dependence/independence) between the activities of these cortical regions.

Using the aforementioned time series data, a 140×140 sample correlation matrix can be computed for each subject. Note that the number of samples is smaller than the number of variables and, therefore, the sample correlation matrix is not invertible. As a result, it is not possible to take the inverse of the sample correlation matrix for estimating a partial correlation matrix. The brain network being sought has $n = 140$ nodes (brain regions). In an effort to find a subgraph of this network as closely as possible to a spanning tree, we choose the regularization parameter λ in the graphical lasso algorithm and the level of thresholding in such a way that they both lead to graphs with $n - 1 = 139$ edges. As illustrated in Figure 2, the similarity degree between the outcomes of these two techniques is above 90% for all twenty subjects. The outcomes of graphical lasso and thresholding obtained for 4 of these subjects are given in Figure 3 for illustration. Note that these graphs are not spanning trees as indented, which imply that graphical lasso and thresholding are able to obtain graphs with $n - 1$ edges but they inevitably have cycles.

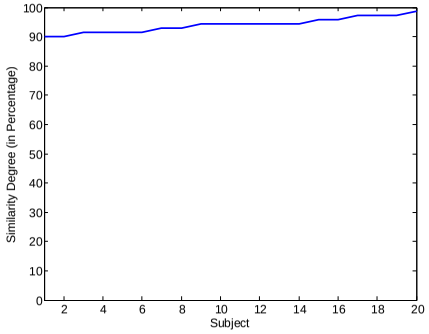


Figure 2: The similarity degree of thresholding and graphical lasso for obtaining brain networks of 20 subjects.

5. Case Study on Electrical Circuits

Consider an arbitrary resistor-capacitor (RC) circuit with n nodes, where certain nodes are connected to each other or the ground via resistor-capacitor elements. Figure 4(a) illustrates an RC circuit. The connectivity of each circuit can be represented by a graph, as demonstrated in Figure 4(b). Assume that the physical structure of the circuit is unknown and only the nodal voltages are available for measurement. It is desirable to find the structural connectivity of the circuit from the measured signals. Given a time instance t, let $V(t)$ denote the vector of the voltages for nodes $1, ..., n$ at time t. Assume that the circuit elements are subject to white thermal noise, namely Johnson-Nyquist noise, and that the conductance and capacitance in each resistor-capacitor element are identical. Let

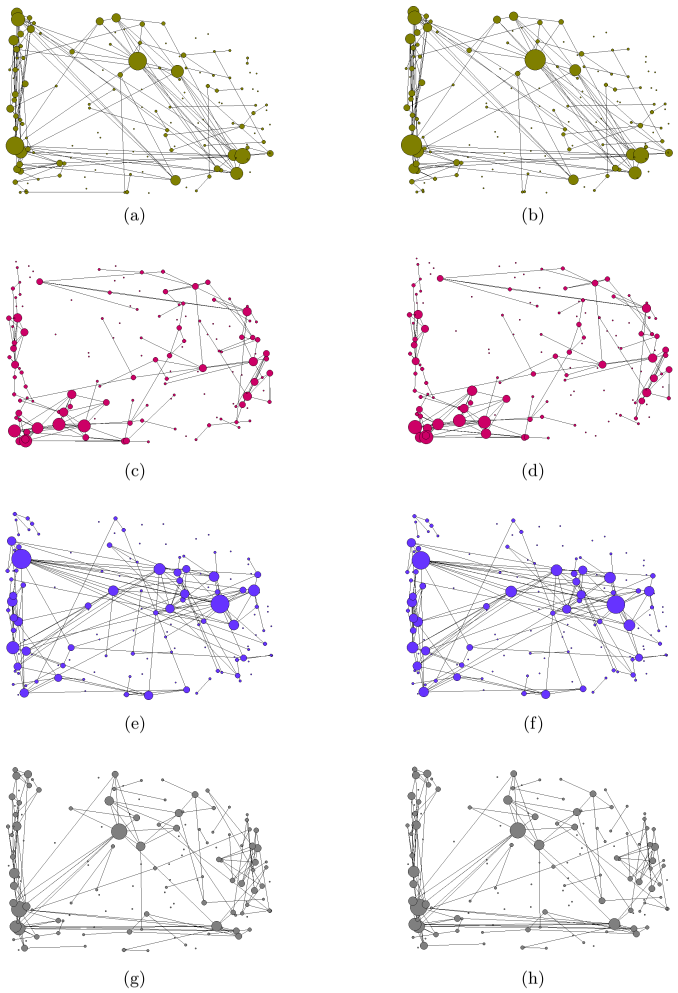
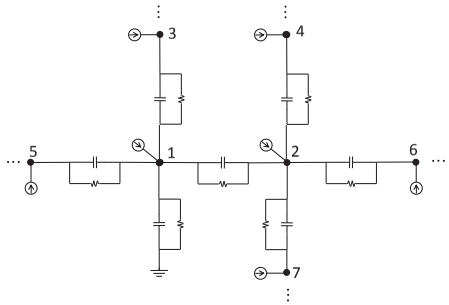


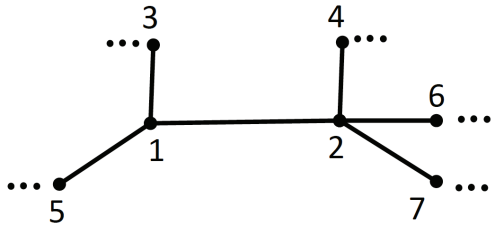
Figure 3: Figures (a) and (b) show the brain networks of one subject obtained from graphical lasso and thresholding, respectively. Similarly, figures (c)-(d), (e)-(f) and (g)-(h) show the networks obtained from graphical lasso and thresholding for three other subjects. The size of each node in these networks reflects its degree.

Σ_* denote the "steady-state" covariance of the voltage measurements. It can be shown that $\Sigma_* = C^{-1}$, where C is the $n \times n$ capacitance matrix of the circuit (Sojoudi and Doyle, 2014). Note that the sparsity pattern of C is consistent with the topology of the circuit. Therefore, the inverse covariance matrix Σ_*^{-1} and the partial correlation matrix both have the same sparsity structure as the circuit. In other words, the partial correlation matrix unveils the physical connectivity of the circuit. Assuming that the circuit under study has a sparse structure, it can be concluded that

- Σ_* is generically a dense matrix, due to being the inverse of the sparse matrix C.
- Σ_*^{-1} is sparse and its sparsity pattern conforms with the circuit topology.



(a)



(b)

Figure 4: a) An RC network with node 1 connected to the ground via a resistor and a capacitor, b) a graph representation of the RC network connectivity.

The above physical model illustrates the fact that the topology of a system may have been encoded in the partial correlation matrix. To recover the circuit topology from the voltage vector $V(t)$, one can sample the vector $V(t)$ at different times $t_1, t_2, ..., t_r$ and construct a sample covariance matrix as

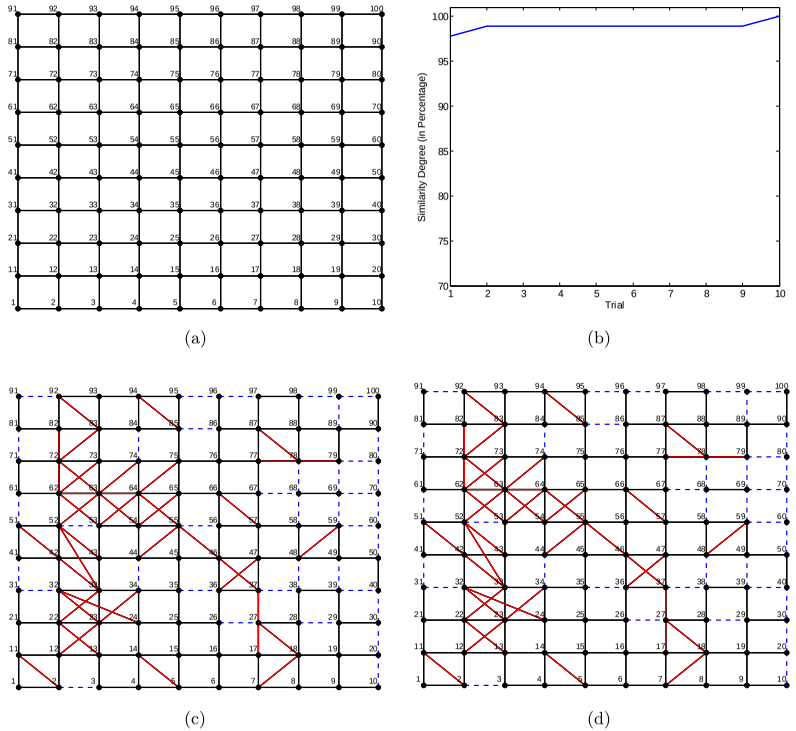
$$\Sigma = \frac{1}{r} \sum_{i=1}^{r} V_i(t_i) V_i(t_i)^T \tag{39}$$

where r denotes the number of samples. Note that Σ converges to the population covariance Σ_* as $r \to \infty$. When r is finite, two possible scenarios arise: i) Σ is invertible but the inverse matrix needs to be thresholded to some level due to the error $\Sigma_* - \Sigma$, ii) Σ is not invertible and therefore alternative methods are needed to estimate the inverse matrix.

The above circuit model can be used to study the relationship between the thresholding and graphical lasso techniques. As an example, consider a mesh circuit with $n = 100$ nodes that are connected to one another through 180 links. The graphical model of this grid circuit is depicted in Figure 5(a). With no loss of generality, assume that $C_{ij} = -1$ for all $(i, j) \in \mathcal{E}$. Furthermore, suppose that nodes 2, 9, 22, 61 and 70 of the circuit are connected to the ground through parallel RC circuits with values equal to 0.1. For $r = 99$ and 10 different trials, we have calculated the sample covariance matrices and applied the thresholding technique and graphical lasso to these matrices in order to recover networks with 180 edges. Note that since the number of samples is less than the number of variables, the sample partial correlations cannot be obtained through matrix inversion. The degree of similarity between graphical lasso and thresholding for these 10 trials are given in Figure 5(b). Figures 5(c) and 5(d) show the networks obtained from thresholding and graphical lasso for one of the trials. False positives are marked in red and false negatives are colored in blue. These two graphs have 178 edges in common (out of 180 edges), which indicates a high degree of similarity between the outcomes of the two techniques under study (to observe some of the few differences, one can inspect the existence of the edges $(64, 75)$ and $(55, 56)$ in these two graphs). We have repeated the above experiment on many circuit models beyond mesh networks and observed a very similar result.

6. Conclusions

The objective of this paper is to study the problem of finding a sparse conditional dependence graph associated with a multivariate random vector, where the only available information is a sample correlation matrix. A commonly used technique for this problem is a convex program, named graphical lasso, where the objective function of this optimization problem has a sparsity-promoting penalty term. In this work, a simple condition is derived under which the graph obtained from graphical lasso coincides with the one obtained by simply thresholding the sample correlation matrix. This condition depends only on the solution of graphical lasso, and makes no use of the regularization coefficient or the sample correlation matrix. The focus of the paper is on the case where the regularization term is high enough to search for a sparse graph. A theoretical result is developed to support that graphical lasso and thresholding behave similarly in this regime, and this statement is verified in several random simulations as well as two case studies on brain connectivity networks and electrical circuits.



(a) (b)

(c) (d)

Figure 5: a) The mesh RC circuit studied in Section 5, b) the similarity degree of thresholding and graphical lasso for the mesh network over 10 trails, c) the network obtained from thresholding the sample correlation matrix for one trial, d) the network obtained from graphical lasso for the same trial used for Figure (c).

References

Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *The Journal of Machine Learning Research*, 9:485–516, 2008.

Alfred M Bruckstein, David L Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1):34–81, 2009.

Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

Venkat Chandrasekaran, Pablo Parrilo, Alan S Willsky, et al. Latent variable graphical model selection via convex optimization. In *48th Annual Allerton Conference on Communication, Control, and Computing*, pages 1610–1613, 2010.

Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397, 2014.

Jianqing Fan and Jinchi Lv. A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 20(1):101, 2010.

Salar Fattahi and Javad Lavaei. On the convexity of optimal decentralized control problem and sparsity path. `http://www.ieor.berkeley.edu/~lavaei/SODC_2016.pdf`, 2016.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Tom Goldstein and Stanley Osher. The Split Bregman method for L1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

Dominique Guillot and Bala Rajaratnam. Retaining positive definiteness in thresholded matrices. `http://arxiv.org/abs/1108.3325`, 2011.

Shuai Huang, Jing Li, Liang Sun, Jieping Ye, Adam Fleisher, Teresa Wu, Kewei Chen, Eric Reiman, Alzheimer's Disease NeuroImaging Initiative, et al. Learning brain connectivity of alzheimer's disease by sparse inverse covariance estimation. *NeuroImage*, 50(3):935–949, 2010.

Ali Jalali, Pradeep D Ravikumar, Vishvas Vasuki, and Sujay Sanghavi. On learning discrete graphical models using group-sparse regularization. In *International Conference on Artificial Intelligence and Statistics*, pages 378–387, 2011.

Nicole Krämer, Juliane Schäfer, and Anne-Laure Boulesteix. Regularized estimation of large-scale gene association networks using graphical Gaussian models. *BMC bioinformatics*, 10(1):384, 2009.

Han Liu, Kathryn Roeder, and Larry Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. In *Advances in Neural Information Processing Systems*, pages 1432–1440, 2010.

Rahul Mazumder and Trevor Hastie. Exact covariance thresholding into connected components for large-scale graphical lasso. *The Journal of Machine Learning Research*, 13(1): 781–794, 2012.

Nicolai Meinshausen and Bin Yu. Lasso-type recovery of sparse representations for high-dimensional data. *The Annals of Statistics*, pages 246–270, 2009.

Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, pages 1–52, 2012.

Mark Schmidt, Alexandru Niculescu-Mizil, Kevin Murphy, et al. Learning graphical model structure using L1-regularization paths. In *AAAI*, volume 7, pages 1278–1283, 2007.

Somayeh Sojoudi. Graphical lasso and thresholding: Conditions for equivalence. *http://www.somayehsojoudi.com/GL_Versus_TH2.pdf*, 2016.

Somayeh Sojoudi and John Doyle. Study of the brain functional network using synthetic data. In *52nd Annual Allerton Conference on Communication, Control, and Computing*, pages 350–357, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Petra E Vértes, Aaron F Alexander-Bloch, Nitin Gogtay, Jay N Giedd, Judith L Rapoport, and Edward T Bullmore. Simple models of human brain functional networks. *Proceedings of the National Academy of Sciences*, 109(15):5868–5873, 2012.

Daniela M Witten, Jerome H Friedman, and Noah Simon. New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, 20(4): 892–900, 2011.

John Wright, Allen Yang, Arvind Ganesh, Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

Ming Yuan and Yi Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

Cun-Hui Zhang and Jian Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *The Annals of Statistics*, pages 1567–1594, 2008.

Journal of Machine Learning Research 17 (2016) 1-13 Submitted 2/16; Published 7/16

A Network That Learns Strassen Multiplication

Veit Elser VE10@CORNELL.EDU
Department of Physics
Cornell
Ithaca, NY 14853-2501, USA

Editor: Nando de Freitas

Abstract

We study neural networks whose only non-linear components are multipliers, to test a new training rule in a context where the precise representation of data is paramount. These networks are challenged to discover the rules of matrix multiplication, given many examples. By limiting the number of multipliers, the network is forced to discover the Strassen multiplication rules. This is the mathematical equivalent of finding low rank decompositions of the $n \times n$ matrix multiplication tensor, M_n. We train these networks with the conservative learning rule, which makes minimal changes to the weights so as to give the correct output for each input at the time the input-output pair is received. Conservative learning needs a few thousand examples to find the rank 7 decomposition of M_2, and 10^5 for the rank 23 decomposition of M_3 (the lowest known). High precision is critical, especially for M_3, to discriminate between true decompositions and "border approximations".

Keywords: sum-product networks, Strassen multiplication, tensor decomposition

1. Introduction

Questions related to the Strassen algorithm (1969) for matrix multiplication provide an opportunity for testing neural networks in the interesting setting where the representation of knowledge is both exact and poorly understood (an open mathematical problem). In this short contribution we define Strassen multiplication (SM) for non-experts, construct a neural network that has the capacity to implement SM, and introduce a simple protocol for training the network called conservative learning. We also present numerical experiments that are unique in the degree of precision required to establish results.

2. What is Strassen Multiplication?

In 1969 Volker Strassen published a paper that showed, among other things, that a pair of 2×2 matrices could be multiplied with just seven scalar multiplications, one less than the eight required in the standard algorithm. Strassen's trick becomes practical on very large matrices where it can be applied recursively on blocks. Finding the fewest number of scalar multiplications for general $n \times n$ matrices is the subject of intensive research.

A glance at the network that implements Strassen multiplication (SM) in Figure 1 quickly establishes two things that make this application of machine learning unique:

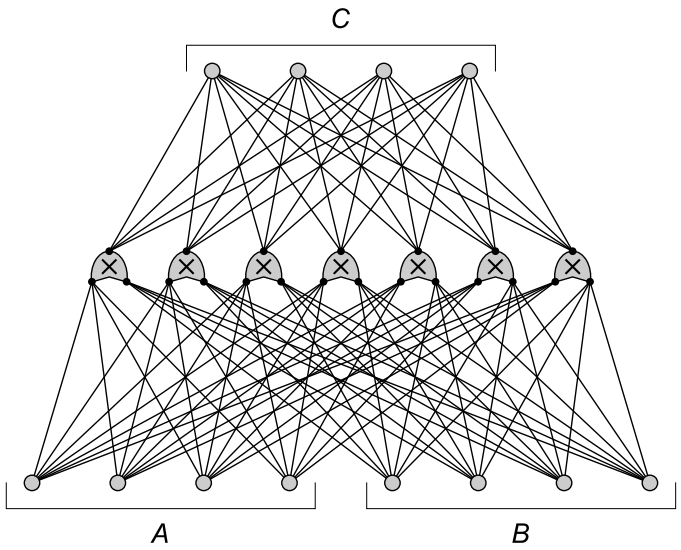


Figure 1: A network for multiplying 2×2 matrices that uses only seven multipliers. Multiplication by a constant ("weight") occurs at each of the lines connecting input and output registers (for the matrices A, B and C) to the multipliers in the middle layer of the network.

- The architecture of the network and the operations performed by its components are *a perfect match to the mathematical problem*, as opposed to being just a well motivated platform for general machine learning applications.
- The design of the network is deliberately naive and lives up to the standard that the machine should have the capacity *to learn something that we don't already know*. Our network, in fact, does not even know how numbers are arranged to form a matrix!

Figure 1 is an engineer's representation of SM. The elements of the 2×2 matrices A and B are read in from two registers of numbers $a(i)$ and $b(i)$, $i = 1, \ldots, 4$ on the left and right. Those from A are 'pooled' into the left inputs of the seven multipliers in the middle layer of the network, those from B into the right inputs. Pooling is the most general linear-homogeneous operation. Denoting the inputs to the multipliers $a^*(j)$ and $b^*(j)$, $j = 1, \ldots, 7$, the pooling equations are

$$a^*(j) = \sum_{i=1}^{4} W_a(j,i)a(i) \qquad b^*(j) = \sum_{i=1}^{4} W_b(j,i)b(i). \tag{1}$$

The 7×4 sets of real numbers W_a and W_b are called *weights*, though they are not required to be non-negative. Multiplication by a weight is not counted as one of the multiplies in SM. Unlike the multiplications performed by the seven multipliers, weight multiplication is a linear operation in the inputs to the network. The challenge of SM is to find a set of weights that work for all conceivable matrices we wish to multiply. We will see that this set is not unique, and there exist transformations from one set to another that works just as well. In the most practical set found by Strassen (1969), all the nonzero weights are just ± 1, so the weight-multiplications are really just additions and subtractions.

The output end of the network works in the same way as the input end: the outputs of the seven multipliers are pooled into the four output registers of the C matrix, $c(i)$, $i = 1, \ldots, 4$, again in the most general way with a set of weights:

$$c(i) = \sum_{j=1}^{7} W_c(i,j)\, a^*(j) b^*(j). \tag{2}$$

Strassen's discovery can now be expressed in purely engineering terms. That is, there exists a fixed set of weights W_a, W_b and W_c such that the four numbers in the product of any pair of matrices can be computed with the 7-multiplier network of Figure 1 (*i.e.* the 8-multiplier network implicit in the standard algorithm is sub-optimal).

3. Mathematical Digression: Tensor Rank

Combining (1) and (2) we obtain a curious statement about matrix multiplication, now for $n \times n$ matrices:

$$c(i) = \sum_{k=1}^{n^2} \sum_{l=1}^{n^2} M_n(i,k,l)\, a(k) b(l)$$

$$M_n(i,k,l) = \sum_{j=1}^{r} W_c(i,j)\, W_a(j,k)\, W_b(j,l).$$

The universal three-index set of numbers M_n is a tensor of order three, the $(n \times n)$-*matrix multiplication tensor*. We see that M_n can be written as a sum of r products of three 1-tensors, the atoms of all tensors. The minimum r in the decomposition into products of 1-tensors is called the rank of the tensor. Unlike the problem of determining the rank of a matrix (a 2-tensor), determining the ranks of higher order tensors such as M_n is computationally difficult (Håstad, 1990). While we know $\mathrm{rank}\,(M_2) = 7$ (Winograd, 1971), already for the next case we currently only have bounds (Bläser, 2003; Laderman, 1976): $19 \leq \mathrm{rank}\,(M_3) \leq 23$.

A general transformation of the weights that gives another correct network (decomposition of M_n) can be inferred from the fact that the transformed matrices

$$\tilde{A} = UAV^{-1} \qquad \tilde{B} = VBW^{-1} \qquad \tilde{C} = UCW^{-1}, \tag{3}$$

for arbitrary invertible (U, V, W), satisfy the matrix product property whenever A, B and C do ($\tilde{A}\tilde{B} = \tilde{C}$). But feeding $\tilde{A}$ and $\tilde{B}$ into the network, and seeing the correct output $\tilde{C}$, is

equivalent to feeding in A and B and seeing C in the output after the three sets of weights have been appropriately transformed by the linear relations (3).

When we test our network for "matrix multiplication fidelity" we would like to do better than spot check its accuracy on sample instances. To assess how well it will perform on *any* instance of matrix multiplication we compute ϵ, where

$$\epsilon^2 = \frac{1}{(n^2)^3} \sum_{i\,k\,l} \left(M_n(i,k,l) - \sum_{j=1}^{r} W_c(i,j)\, W_a(j,k)\, W_b(j,l) \right)^2 \tag{4}$$

is the mean-square error in how our weights are decomposing the true M_n.

4. Conservative Learning

The amazing thing about the network in Figure 1 is not that, given the special Strassen weights, it manages to produce the product $C = AB$ with only seven multiplies. What is truly remarkable is the fact that the network can learn a correct set of weights just by being shown enough examples of correctly multiplied matrices.

A huge industry has grown up around the problem of training networks. Most methods start with some initial weights and adjust these to minimize the discrepancy between the correct output and the actual output of the network. There are many schemes for minimization, though most are local and based on gradients of some measure of discrepancy ("loss") with respect to the weights. We will use a somewhat different approach for the Strassen network called conservative learning.

Conservative learning is not the oxymoron it would seem, but a combination of two very reasonable principles:

- When presented with a training item for which the network gives the wrong output, change the weights so that at least this item produces the correct output.
- Make the smallest possible change to the weights when learning each new item.

By making the smallest possible change, when obsessing over the most recent item, we stand a better chance of not corrupting the accumulated knowledge derived from all previous items. To add mathematical support to this statement we analyze a very simple network.

Consider a network that implements a linear transformation from a set of inputs $x \in \mathbb{R}^m$ to a set of outputs $y \in \mathbb{R}^n$. We use matrix notation in what follows. The transformation to be learned corresponds to a matrix of weights $W^\star$, and training samples are pairs (x, y) where $y = W^\star x$. Suppose W is the current matrix and $(x, W^\star x)$ a new training item. As our network is linear, we choose to train only with normalized inputs, $x^T x = 1$. We wish to find the smallest change Δ such that

$$(W + \Delta)x = W^\star x, \tag{5}$$

where small is defined by the Frobenius norm $\|\Delta\| = \mathrm{Tr}\,\Delta\Delta^T$. We may decompose Δ as a sum of terms

$$\Delta = \Delta_\parallel x^T + \Delta_\perp,$$

where the first defines its action on the subspace generated by x, and the second gives its action on the orthogonal complement. Since $\Delta_\perp x = 0$, from (5) we infer

$$\Delta_\parallel = (W^\star - W)x.$$

Moreover,

$$\begin{aligned} \operatorname{Tr}\Delta\Delta^T &= \operatorname{Tr}(\Delta_\parallel x^T + \Delta_\perp)(x\Delta_\parallel^T + \Delta_\perp^T) \\ &= \operatorname{Tr}\Delta_\parallel\Delta_\parallel^T + \operatorname{Tr}\Delta_\perp\Delta_\perp^T, \end{aligned}$$

shows that minimizing the norm of Δ requires $\Delta_\perp = 0$, since the second term is a sum of squares. The conservative learning rule is therefore:

$$W \to W' = W + (W^\star - W)xx^T.$$

The current weights are incremented by a rank 1 matrix comprising the transposed input vector x^T as one factor and the output discrepancy $(W^\star - W)x$ as the other. After the change, the weights W' are correct on the item $(x, W^\star x)$.

To better understand how we are doing relative to the corpus of all possible items, we make note of the following relationship between old and new weights:

$$\begin{aligned} W' - W^\star &= (W - W^\star)(1 - xx^T) \\ &= P_x^\perp(W - W^\star). \end{aligned} \tag{6}$$

The operator $P_x^\perp$ projects rows to the orthogonal complement of x. From (6) we infer the norm inequality

$$\|W' - W^\star\| \le \|W - W^\star\|,$$

where we get equality only in the rare case that x is orthogonal to all the rows of $W - W^\star$. This inequality proves that conservatism is sound: by minimally accommodating each new item there is monotone improvement in our approximation of the transformation $(W^\star)$ itself.

Conservative learning has the nice feature that there are no parameters that have to be tuned, such as the step size in gradient descent. This continues to be true when the principle is applied to general networks. However, multiple layers and nonlinear components make the problem of finding the exact weight modifications intractable. Fortunately, there is a systematic procedure for keeping track of the order of smallness of the changes in the general network so that an approximate learning rule can be written down. The update procedure for the weights in the multilayer setting has similarities with the back-propagation rules in gradient-based learning, but there are also some differences. Not surprisingly, proving convergence is also out of reach. Our experiments with the Strassen network show spectacular convergence, even without imposing a small parameter to keep changes in check.

A derivation of the conservative learning rules for the Strassen network is given in the Appendix. This network has most of the features that need to be addressed in general networks and therefore serves as a good vehicle for explaining the method.

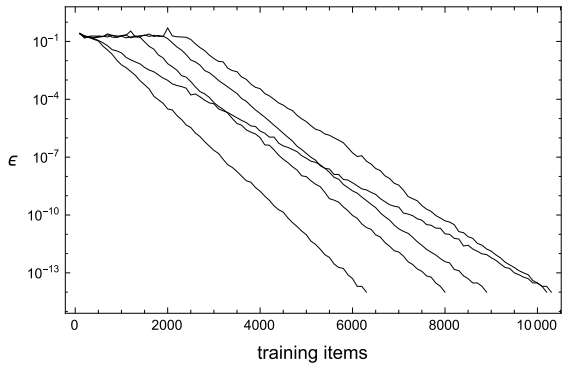


Figure 2: Root-mean-square error ϵ in the rank 7 decomposition of the 2×2 matrix multiplication tensor by the network in Figure 1. Shown are results for five runs of the network, each starting with random weights and trained on streams of random instances of correctly multiplied matrices.

5. Conservative Learning with the Strassen Network

There are two sources of randomness when training a network: the distribution of the initial weights and the distribution of the training items. We did not explore this dimension in our experiments with the Strassen network. The weights were initialized with independent uniform samples from a symmetric interval about the origin. Clearly the scale of this interval is not arbitrary, since the tensor M_n being decomposed has a definite scale. A scale for the weights is also implied by the error objective (4). Conservative learning proved not to be very sensitive to the scale of the initial weights, with only extreme limits (small and large scales) showing performance degradation. For simplicity we therefore chose to always initialize with samples drawn from $[-1, 1]$.

Adding an interesting bias to the distribution of training matrices is also something we did not try. Our input matrices were simply constructed from uniform samples of $[-1, 1]$ in each element of A and B. After rescaling to conform with our normalization convention $\operatorname{Tr} A^T A = \operatorname{Tr} B^T B = 1$, these together with $C = AB$ were handed to the network for training.

It astonished us how quickly the network discovers Strassen's trick for 2×2 matrices, when the weights are updated by the conservative learning rules. Figure 2 shows the decomposition error ϵ converging linearly, after an initial training set of only a few thousand. The five runs shown have slightly different convergence rates, a variability that can only be attributed to the initial weights and the early training items (before the onset of linear convergence). We did not compare with other learning rules, all of which, unlike conservative

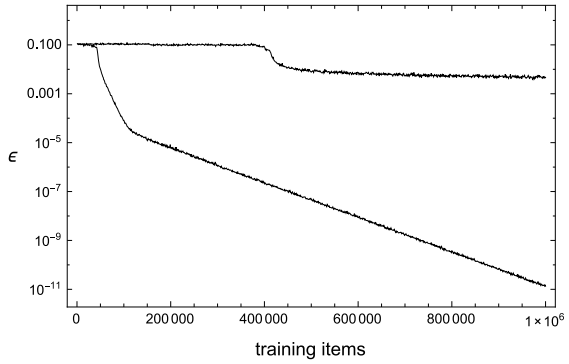


Figure 3: Root-mean-square error ϵ, in rank 23 decompositions of the 3×3 matrix multiplication tensor, on two runs. The absence of convincing linear convergence in one of the runs is accompanied (Figure 4) by a slow growth in the maximum weight magnitude.

learning, come with parameters and batch protocols that we were not prepared to manage with competence.

When we remove one multiplier from the middle layer of the network in Figure 1, that is, attempt to find a rank 6 decomposition of M_2, the error ϵ fluctuates indefinitely. This of course is what we expect, since there is a proof that M_2 has rank 7 (Winograd, 1971). For 3×3 matrices the network exhibits a new kind of behavior. Since it is known that rank $(M_3) \leq 23$ (Laderman, 1976), convergent behavior is possible in principle when we have 23 multipliers. We find that this indeed happens in about 64% of all runs, an example of which is the lower plot in Figure 3. In the remaining runs ϵ decays much more slowly; an example of this behavior is the higher plot in Figure 3.

The phenomenon of there being two qualitatively different asymptotic states of the network, when decomposing M_3, is consistent with the *border rank* property. This property, peculiar to tensors of order three and higher, refers to the topological closure properties of the space of tensor decompositions. When this space is not closed, the abstract closure defines tensor decompositions on the "border" of the space that have lower rank. In more concrete terms, it gives rise to approximate decompositions of lower rank, where the approximation as measured by ϵ can be arbitrarily good when the weights are allowed to diverge without limit.

We see evidence of "border" behavior when we compare the evolution of the maximum weight magnitude, $\max(|W_a|, |W_b|, |W_c|)$, for the two runs in Figure 3. These are plotted in Figure 4. In contrast to the linearly convergent run, where the maximum weight saturates, the run with slow convergence has a corresponding slow growth in the maximum weight. An interpretation of our results, consistent with what is known about tensor decompositions of

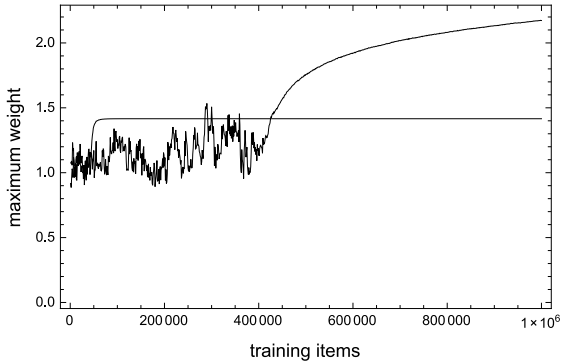


Figure 4: The maximum weight magnitudes, $\max(|W_a|, |W_b|, |W_c|)$, for the two training runs shown in Figure 3.

M_3, is that the weights in the slowly converging runs are rank 23 border approximations to true rank 24 or higher decompositions. Running the network with 22 multipliers we only observe the slow convergence/growing weight behavior. While this does not prove the true rank is 23, it is consistent with M_3 having rank 22 border approximations (Schönhage, 1981). That slow convergence was never observed in our experiments with M_2 is in agreement with the known fact (Landsberg, 2006) that this tensor does not have lower rank border approximations.

Conservative learning with the Strassen network for 3×3 matrices is summarized in Figure 5. This gives the distribution of the final decomposition error ϵ for 10^3 runs, each limited to 10^8 training items and terminated when the error dropped below 10^{-14}. Runs with linear convergence, such as the one in Figure 3, typically required far fewer than 10^8 training items. The large peak at the low end of the distribution in Figure 5, about 64% of all runs, therefore gives the probability that the network finds a true rank 23 decomposition. As explained above, we believe the network is slowly converging to border approximations in the other runs.

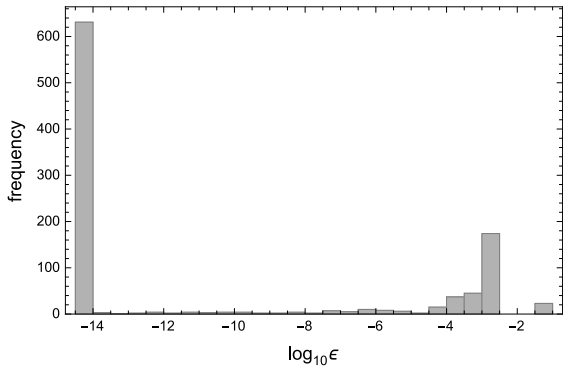


Figure 5: Distribution of the final decomposition error of M_3 in 10^3 runs, each with 10^8 training items. Runs were terminated when ϵ dropped below 10^{-14}.

6. Some History and Concluding Remarks

Not surprisingly, given the naturalness of the principle, a number of prior applications of conservative learning came to light after submission of the manuscript. Our definition is practically indistinguishable from the "minimal disturbance principle" that Bernard Widrow and coworkers (1988) applied to multi-layer networks with two-valued (sign) activation functions. The following is a direct quote from their paper:

> The idea is to adapt the network to properly respond to the newest input pattern while minimally disturbing the responses already trained in for the previous input patterns. Unless this principle is practiced, it is difficult for the network to simultaneously store all of the required pattern responses.

Due to the discontinuous activation functions in their network, the implementation of conservative learning (the minimal disturbance principle) posed a greater challenge than the non-linearity in the Strassen network. Combinations of flips of signs (activations) had to be exhaustively tested, for each training item, to discover the minimal modification of the weights. We were able to avoid a combinatorial explosion in the Strassen network because differentiability gave a set of equations for the minimum whose linearization could be solved for a unique, approximate solution.

While nearly forgotten by the neural network community, minimal disturbance has stood as a major design principle for adaptive filters (Widrow and Hoff, 1960), where it is known as the least mean squares (LMS) algorithm. With the development of efficient optimization algorithms such as ADMM for implementation, the application of the principle to neural networks should now be feasible, and offers an interesting training alternative to stochastic gradient descent.

Acknowledgments

I thank Cris Moore for instigating this study, Jonathan Yedidia for suggesting the conservative learning method, and Alex Alemi for keeping things competitive. Junping Shao discovered the related earlier work by Widrow. The Simons Foundation and ADI Lyric Labs provided financial support.

Appendix A. Conservative Learning Rules for the Strassen Network

The starting point for deriving the conservative learning rules for the Strassen network (Fig. 1) is the Lagrangian function

$$\mathcal{L} = \frac{1}{2}\,\mathrm{Tr}\,(\Delta_a^T\Delta_a) + \frac{1}{2}\,\mathrm{Tr}\,(\Delta_b^T\Delta_b) + \frac{1}{2}\,\mathrm{Tr}\,(\Delta_c^T\Delta_c) + \alpha^T\,(a^* - (W_a+\Delta_a)a) + \beta^T\,(b^* - (W_b+\Delta_b)b) + \gamma^T\,(c - (W_c+\Delta_c)a^*\circ b^*)\,.$$

To streamline the derivation we use matrix notation. The new training item is the triplet of vectors (a, b, c) corresponding to the unrolled matrices $(A, B, C = AB)$. We should think of (a, b) as inputs to the network and c as the output. Prior to this item the network has weight matrices W_a and W_b that map the inputs to the pairs of inputs of the multipliers in the middle layer, (a^*, b^*). The outputs of the multipliers is the vector $c^* = a^* \circ b^*$, where $\circ$ denotes componentwise multiplication. If the network weights are correct even for the new item, then mapping c^* with the weight matrix W_c should match the output vector c. In general, the three weight matrices have to be changed by $(\Delta_a, \Delta_b, \Delta_c)$ for this to be true, and the first three terms of $\mathcal{L}$ are the Frobenius norm objective on these changes to keep them small. The last three terms are constraints imposed via three vectors of Lagrange multipliers, (α, β, γ). These insure that with the changed weights the inputs/outputs of the network match the inputs/outputs of the multipliers in the middle layer.

Given the current weights (W_a, W_b, W_c) and the new training item (a, b, c), our task is then to find a stationary point of Lagrangian $\mathcal{L}$ for the variables $(\Delta_a, \Delta_b, \Delta_c)$, (a^*, b^*), and (α, β, γ). In the derivation below we assume that the inputs are normalized as $a^T a = b^T b = 1$.

Stationarity with respect to Δ_a and Δ_b imply

$$\Delta_a = \alpha\, a^T \qquad \Delta_b = \beta\, b^T. \tag{7}$$

Comparing with the update rule for the linear network in Section 4 prompts us to interpret α and β as discrepancies associated with the multiplier inputs. Multiplying (7) on the right respectively by a and b we also have

$$\Delta_a\, a = \alpha \qquad \Delta_b\, b = \beta. \tag{8}$$

These equations and ones to follow are consistent with the Lagrange multipliers vanishing proportionately with the changes in the weights.

Before we impose stationarity with respect to the other variables, we define a set of approximate vectors associated with implementing the network using the current (unchanged)

weights. This mode of evaluating the nodes (vectors) in the network is called a forward pass.

$$\tilde{a}^* = W_a\, a \qquad \tilde{b}^* = W_b\, b. \tag{9}$$

$$\tilde{c} = W_c\, \tilde{a}^* \circ \tilde{b}^*. \tag{10}$$

Imposing stationarity with respect to (α, β) and comparing the resulting equations with (9) and (8), we obtain

$$a^* = (W_a + \Delta_a)a \qquad b^* = (W_b + \Delta_b)b,$$

$$a^* - \tilde{a}^* = \Delta_a\, a = \alpha \qquad b^* - \tilde{b}^* = \Delta_b\, b = \beta.$$

This shows that the discrepancies represented by α and β are the differences between their true values (after the conservative-learning update) and their forward-pass values.

Learning in some sense starts at the output layer, where the $\tilde{c}$ of the forward pass is compared with the output c of the training item. To obtain the conservative learning rule for this we start by imposing stationarity of $\mathcal{L}$ with respect to Δ_c:

$$\Delta_c = \gamma\, (a^* \circ b^*)^T. \tag{11}$$

We now make the first of a series of approximations. Extending to the output layer the property of the middle layer, that the pairs (Δ_a, α) and (Δ_b, β) have the same order of smallness, we expect (Δ_c, γ) also to vanish proportionately. A good approximation of (11) is then to replace a^* and b^* by their forward pass values (the error being higher order),

$$\Delta_c \approx \gamma\, (\tilde{a}^* \circ \tilde{b}^*)^T = \gamma\, \tilde{c}^{*T}, \tag{12}$$

where we have expressed the result in terms of the forward pass value of the multiplier outputs, $\tilde{c}^*$. Now imposing stationarity of $\mathcal{L}$ with respect to γ,

$$c = (W_c + \Delta_c)\, a^* \circ b^*,$$

and comparing with (10) we obtain

$$\begin{aligned} c - \tilde{c} &\approx \Delta_c\, \tilde{c}^* + W_c \left((a^* - \tilde{a}^*) \circ \tilde{b}^* + \tilde{a}^* \circ (b^* - \tilde{b}^*) \right) \\ &\approx \gamma\, \tilde{c}^{*T} \tilde{c}^* + W_c\, (\alpha \circ \tilde{b}^* + \tilde{a}^* \circ \beta), \end{aligned} \tag{13}$$

where we have neglected second order terms and used (7) and (12).

By imposing stationarity with respect to the two remaining sets of variables (a^*, b^*) we will obtain equations that relate (α, β) to γ, enabling us to cast the equation for the output discrepancy (13) just in terms of the unknown γ:

$$\alpha = b^* \circ (W_c^T + \Delta_c^T)\gamma \qquad \beta = a^* \circ (W_c^T + \Delta_c^T)\gamma,$$

$$\alpha \approx b^* \circ (W_c^T \gamma) \qquad \beta \approx a^* \circ (W_c^T \gamma). \tag{14}$$

In (14) we again discarded second order terms. Substituting α and β from (14) into (13) we arrive at the equation that begins the process of updating the weights:

$$c - \tilde{c} \approx (\tilde{c}^{*T} \tilde{c}^*)\, \gamma + W_c\, (\tilde{a}^* \circ \tilde{a}^* + \tilde{b}^* \circ \tilde{b}^*)\, W_c^T\, \gamma. \tag{15}$$

Equation (15) relates the discrepancy, between the true (training) c and the $\tilde{c}$ of the forward pass, to the Lagrange multipliers γ. Were we to neglect the off-diagonal terms in this linear matrix equation and determine γ by

$$\gamma \leftarrow \frac{1}{\tilde{c}^{*T} \tilde{c}^*} (c - \tilde{c}), \tag{16}$$

then the process of learning item (a, b, c) would be completely analogous to the usual "back-propagation" scheme. After forward-propagating the inputs (a, b) with W_a, W_b and W_c to determine the discrepancy $c - \tilde{c}$, the γ given by (16) is back-propagated (14) by W_c^T to get α and β. The three Lagrange multipliers then determine the weight changes by (7) and (12).

Lacking an argument for discarding the off-diagonal terms in (15), we need to look for methods to solve this more complex linear equation. The off-diagonal terms correspond to one-level of back-propagation followed by forward-propagation. An iterative solution of the linear equation would thus involve multiple backward-forward propagations between just the final two layers of the network. This is not as intimidating as it might seem for two reasons. First, when using the conjugate gradient (CG) method, the number of backward-forward iterations is bounded by the number of components of the vector γ. Second, given a reasonable initial solution-estimate, CG usually requires only few iterations in practice.

Since our derivation of the conservative learning rules has been based on the premise that the discrepancies $(a^* - \tilde{a}^*, b^* - \tilde{b}^*, c - \tilde{c})$ are small, we are keeping with this premise when we take as our initial CG solution-estimate $\gamma = 0$. As a better motivated alternative to (16) we apply the fewest (non-trivial) number of CG iterations — a single one — to this solution-estimate. At this single-iteration level of CG the approximate solution has a simple interpretation. Let

$$G = (\tilde{c}^{*T} \tilde{c}^*) + W_c\, (\tilde{a}^* \circ \tilde{a}^* + \tilde{b}^* \circ \tilde{b}^*)\, W_c^T \tag{17}$$

be the symmetric positive definite matrix in our linear equation

$$\delta = c - \tilde{c} = G\gamma.$$

The single-iteration CG solution has the form $\gamma_1 = \lambda\delta$, where λ is a scalar multiplier and determined by projecting the equation on δ:

$$\delta^T(\delta - G(\lambda\delta)) = 0.$$

The conservative learning alternative to (16) is therefore

$$\gamma \leftarrow \gamma_1 = \frac{(c - \tilde{c})^T (c - \tilde{c})}{(c - \tilde{c})^T G (c - \tilde{c})} (c - \tilde{c}). \tag{18}$$

Not surprisingly this reduces to (16) when the off-diagonal terms in G are dropped.

We conclude this Appendix with a short summary of the conservative learning rules for updating the network weights when given a new training item (a, b, c).

1. Forward propagate (a, b) using (9) and (10) to get $(\tilde{a}^*, \tilde{b}^*, \tilde{c}^* = \tilde{a}^* \circ \tilde{b}^*)$ and $\tilde{c}$.
2. From the output discrepancy $c - \tilde{c}$ compute γ using (18). Computing the scalar multiple of $c - \tilde{c}$ in this expression requires a single backward-forward propagation by the matrix G in (17).
3. The update Δ_c of W_c is the rank 1 matrix (12) constructed from γ and $\tilde{c}^*$.
4. Backward propagate γ by (14) to obtain α and β.
5. The updates (Δ_a, Δ_b) of (W_a, W_b) are given by the rank 1 matrices (7) constructed from (α, β) and (a, b).

References

M. Bläser. On the Complexity of the Multiplication of Matrices of Small Formats. *J. Complexity*, 19:43–60, 2003.

J. Håstad. Tensor Rank is NP-Complete. *J. Algorithms*, 11:644–654, 1990.

J.D. Laderman. A Noncommutative Algorithm for Multiplying 3×3 Matrices Using 23 Multiplications. *Bull. Amer. Math. Soc.*, 82:126–128, 1976.

J.M. Landsberg. The Border Rank of the Multiplication of 2×2 Matrices is Seven. *J. Amer. Math. Soc.*, 19:447–459, 2006.

A. Schönhage. Partial and Total Matrix Multiplication. *SIAM J. Comput.*, 10:434–455, 1981.

V. Strassen. Gaussian Elimination is Not Optimal. *Numer. Math.*, 13:354–356, 1969.

B. Widrow and M.E. Hoff. Adaptive Switching Circuits. *IRE WESCON Conv. Rec.*, 4: 96–104, 1960.

B. Widrow, R.G. Winter, and R.A. Baxter. Layered Neural Nets for Pattern Recognition. *IEEE Trans. Sig. Process.*, 36:1109–1118, 1988.

S. Winograd. On the Multiplication of 2×2 Matrices. *Linear Algebra Appl.*, 4:381–388, 1971.

Journal of Machine Learning Research 17 (2016) 1-65 Submitted 2/14; Revised 2/15; Published 4/16

Revisiting the Nyström Method for Improved Large-scale Machine Learning

Alex Gittens GITTENS@ICSI.BERKELEY.EDU
Michael W. Mahoney MMAHONEY@STAT.BERKELEY.EDU
International Computer Science Institute and Department of Statistics
University of California, Berkeley
Berkeley, CA

Editor: Mehryar Mohri

Abstract

We reconsider randomized algorithms for the low-rank approximation of symmetric positive semi-definite (SPSD) matrices such as Laplacian and kernel matrices that arise in data analysis and machine learning applications. Our main results consist of an empirical evaluation of the performance quality and running time of sampling and projection methods on a diverse suite of SPSD matrices. Our results highlight complementary aspects of sampling versus projection methods; they characterize the effects of common data preprocessing steps on the performance of these algorithms; and they point to important differences between uniform sampling and nonuniform sampling methods based on leverage scores. In addition, our empirical results illustrate that existing theory is so weak that it does not provide even a qualitative guide to practice. Thus, we complement our empirical results with a suite of worst-case theoretical bounds for both random sampling and random projection methods. These bounds are qualitatively superior to existing bounds—*e.g.*, improved additive-error bounds for spectral and Frobenius norm error and relative-error bounds for trace norm error—and they point to future directions to make these algorithms useful in even larger-scale machine learning applications.

Keywords: Nyström approximation, low-rank approximation, kernel methods, randomized algorithms, numerical linear algebra

1. Introduction

We reconsider randomized algorithms for the low-rank approximation of symmetric positive semi-definite (SPSD) matrices such as Laplacian and kernel matrices that arise in data analysis and machine learning applications. Our goal is to obtain an improved understanding, both empirically and theoretically, of the complementary strengths of sampling versus projection methods on realistic data. Our main results consist of an empirical evaluation of the performance quality and running time of sampling and projection methods on a diverse suite of dense and sparse SPSD matrices drawn both from machine learning as well as more general data analysis applications. These results are not intended to be comprehensive but instead to be illustrative of how randomized algorithms for the low-rank approximation of SPSD matrices behave in a broad range of realistic machine learning and data analysis applications.

Our empirical results point to several directions that are not explained well by existing theory. (For example, that the results are much better than existing worst-case theory would suggest, and that sampling with respect to the statistical leverage scores leads to results that are complementary to those achieved by projection-based methods.) Thus, we complement our empirical results with a suite of worst-case theoretical bounds for both random sampling and random projection methods. These bounds are qualitatively superior to existing bounds—*e.g.*, improved additive-error bounds for spectral and Frobenius norm error and relative-error bounds for trace norm error. By considering random sampling and random projection algorithms on an equal footing, we identify within our analysis deterministic structural properties of the input data and sampling/projection methods that are responsible for high-quality low-rank approximation.

In more detail, our main contributions are fourfold.

- First, we provide an empirical illustration of the complementary strengths and weaknesses of data-independent random projection methods and data-dependent random sampling methods when applied to SPSD matrices. We do so for a diverse class of SPSD matrices drawn from machine learning and data analysis applications, and we consider reconstruction error with respect to the spectral, Frobenius, and trace norms. Depending on the parameter settings, the matrix norm of interest, the data set under consideration, etc., one or the other method might be preferable. In addition, we illustrate how these empirical properties can often be understood in terms of the structural nonuniformities of the input data that are of independent interest.
- Second, we consider the running time of high-quality sampling and projection algorithms. For random sampling algorithms, the computational bottleneck is typically the exact or approximate computation of the importance sampling distribution with respect to which one samples; and for random projection methods, the computational bottleneck is often the implementation of the random projection. By exploiting and extending recent work on "fast" random projections and related recent work on "fast" approximation of the statistical leverage scores, we illustrate that high-quality leverage-based random sampling and high-quality random projection algorithms have comparable running times. Although both are slower than simple (and in general much lower-quality) uniform sampling, both can be implemented more quickly than a naïve computation of an orthogonal basis for the top part of the spectrum.
- Third, our main technical contribution is a set of deterministic structural results that hold for any "sketching matrix" applied to an SPSD matrix. We call these "deterministic structural results" since there is no randomness involved in their statement or analysis and since they depend on structural properties of the input data matrix and the way the sketching matrix interacts with the input data. In particular, they highlight the importance of the statistical leverage scores, which have proven important in other applications of random sampling and random projection algorithms.
- Fourth, our main algorithmic contribution is to show that when the low-rank sketching matrix represents certain random projection or random sampling operations, then we obtain worst-case quality-of-approximation bounds that hold with high probability. These bounds are qualitatively better than existing bounds and they illustrate

how high-quality random sampling algorithms and high-quality random projection algorithms can be treated from a unified perspective.

A novel aspect of our work is that we adopt a unified approach to these low-rank approximation questions—unified in the sense that we consider both sampling and projection algorithms on an equal footing, and that we illustrate how the structural nonuniformities responsible for high-quality low-rank approximation in worst-case analysis also have important empirical consequences in a diverse class of SPSD matrices. By identifying deterministic structural conditions responsible for high-quality low-rank approximation of SPSD matrices, we highlight complementary aspects of sampling and projection methods; and by illustrating the empirical consequences of structural nonuniformities, we provide theory that is a much closer guide to practice than has been provided by prior work. We note also that our deterministic structural results could be used to check, in an *a posteriori* manner, the quality of a sketching method for which one cannot establish an *a priori* bound.

Our analysis is timely for several reasons. First, in spite of the empirical successes of Nyström-based and other randomized low-rank methods, existing theory for the Nyström method is quite modest. For example, existing worst-case bounds such as those of Drineas and Mahoney (2005) are very weak, especially compared with existing bounds for least-squares regression and general low-rank matrix approximation problems (Drineas et al., 2008, 2010; Mahoney, 2011).[1] Moreover, many other worst-case bounds make very strong assumptions about the coherence properties of the input data (Kumar et al., 2012; Gittens, 2012). Second, there have been conflicting views in the literature about the usefulness of uniform sampling versus nonuniform sampling based on the empirical statistical leverage scores of the data in realistic data analysis and machine learning applications. For example, some work has concluded that the statistical leverage scores of realistic data matrices are fairly uniform, meaning that the coherence is small and thus uniform sampling is appropriate (Williams and Seeger, 2001; Kumar et al., 2012); while other work has demonstrated that leverage scores are often very nonuniform in ways that render uniform sampling inappropriate and that can be essential to highlight properties of downstream interest (Paschou et al., 2007; Mahoney and Drineas, 2009). Third, in recent years several high-quality numerical implementations of randomized matrix algorithms for least-squares and low-rank approximation problems have been developed (Avron et al., 2010; Meng et al., 2014; Woolfe et al., 2008; Rokhlin et al., 2009; Martinsson et al., 2011). These have been developed from a "scientific computing" perspective, where condition numbers, spectral norms, etc. are of greater interest (Mahoney, 2012), and where relatively strong homogeneity assumptions can be made about the input data. In many "data analytics" applications, the questions one asks are very different, and the input data are much less well-structured. Thus, we expect that some of our results will help guide the development of algorithms and implementations that are more appropriate for large-scale analytics applications.

In the next section, Section 2, we start by presenting some notation, preliminaries, and related prior work. Then, in Section 3 we present our main empirical results; and in Section 4 we present our main theoretical results. We conclude in Section 5 with a brief discussion of our results in a broader context.

1. This statement may at first surprise the reader, since an SPSD matrix is an example of a general matrix, and one might suppose that the existing theory for general matrices could be applied to SPSD matrices. While this is true, these existing methods for general matrices do not in general respect the symmetry or positive semi-definiteness of the input.

2. Notation, Preliminaries, and Related Prior Work

In this section, we introduce the notation used throughout the paper, and we address several preliminary considerations, including reviewing related prior work.

2.1 Notation

Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be an arbitrary SPSD matrix with eigenvalue decomposition $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T$, where we partition $\mathbf{U}$ and $\mathbf{\Sigma}$ as

$$\mathbf{U} = \begin{pmatrix}\mathbf{U}_1 & \mathbf{U}_2\end{pmatrix} \text{ and } \mathbf{\Sigma} = \begin{pmatrix}\mathbf{\Sigma}_1 & \\ & \mathbf{\Sigma}_2\end{pmatrix}. \tag{1}$$

Here, $\mathbf{U}_1$ has k columns and spans the top k-dimensional eigenspace of $\mathbf{A}$, and $\mathbf{\Sigma}_1 \in \mathbb{R}^{k\times k}$ is full-rank.[2] We denote the eigenvalues of $\mathbf{A}$ with $\lambda_1(\mathbf{A}) \geq \ldots \geq \lambda_n(\mathbf{A})$.

Given $\mathbf{A}$ and a rank parameter k, the *statistical leverage scores of* $\mathbf{A}$ *relative to the best rank-k approximation to* $\mathbf{A}$ equal the squared Euclidean norms of the rows of the $n \times k$ matrix $\mathbf{U}_1$:

$$\ell_j = \|(\mathbf{U}_1)_j\|^2. \tag{2}$$

The leverage scores provide a more refined notion of the structural nonuniformities of $\mathbf{A}$ than does the notion of *coherence*, $\mu = \frac{n}{k}\max_{i\in\{1,\ldots,n\}} \ell_i$, which equals (up to scale) the largest leverage score; and they have been used historically in regression diagnostics to identify particularly influential or outlying data points. Less obviously, the statistical leverage scores play a crucial role in recent work on randomized matrix algorithms: they define the key structural nonuniformity that must be dealt with in order to obtain high-quality low-rank and least-squares approximation of general matrices via random sampling and random projection methods (Mahoney, 2011). Although Equation (2) defines them with respect to a particular basis, the statistical leverage scores equal the diagonal elements of the projection matrix onto the span of that basis, and thus they can be computed from any basis spanning the same space. Moreover, they can be approximated more quickly than the time required to compute that basis with a truncated SVD or a QR decomposition (Drineas et al., 2012).

We denote by $\mathbf{S}$ an arbitrary $n \times \ell$ "sketching" matrix that, when post-multiplying a matrix $\mathbf{A}$, maps points from $\mathbb{R}^n$ to $\mathbb{R}^\ell$. We are most interested in the case where $\mathbf{S}$ is a random matrix that represents a random sampling process or a random projection process, but we do not impose this as a restriction unless explicitly stated. We let

$$\mathbf{\Omega}_1 = \mathbf{U}_1^T\mathbf{S} \quad \text{and } \mathbf{\Omega}_2 = \mathbf{U}_2^T\mathbf{S} \tag{3}$$

denote the projection of $\mathbf{S}$ onto the top and bottom eigenspaces of $\mathbf{A}$, respectively.

2. Variants of our results hold trivially if the rank of $\mathbf{A}$ is k or less, so we focus on this more general case here.

Recall that, by keeping just the top k singular vectors, the matrix $\mathbf{A}_k := \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{U}_1^T$ is the best rank-k approximation to $\mathbf{A}$, when measured with respect to any unitarily-invariant matrix norm, *e.g.*, the spectral, Frobenius, or trace norm. For a vector $\mathbf{x} \in \mathbb{R}^n$, let $\|\mathbf{x}\|_\xi$, for $\xi = 1, 2, \infty$, denote the 1-norm, the Euclidean norm, and the ∞-norm, respectively, and let $\mathrm{Diag}(\mathbf{A})$ denote the vector consisting of the diagonal entries of the matrix $\mathbf{A}$. Then, $\|\mathbf{A}\|_2 = \|\mathrm{Diag}(\mathbf{\Sigma})\|_\infty$ denotes the *spectral norm* of $\mathbf{A}$; $\|\mathbf{A}\|_\mathrm{F} = \|\mathrm{Diag}(\mathbf{\Sigma})\|_2$ denotes the *Frobenius norm* of $\mathbf{A}$; and $\|\mathbf{A}\|_\star = \|\mathrm{Diag}(\mathbf{\Sigma})\|_1$ denotes the *trace norm* (or nuclear norm) of $\mathbf{A}$. Clearly,

$$\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_\mathrm{F} \leq \|\mathbf{A}\|_\star \leq \sqrt{n}\,\|\mathbf{A}\|_\mathrm{F} \leq n\,\|\mathbf{A}\|_2\,.$$

We quantify the quality of our algorithms by the "additional error" (above and beyond that incurred by the best rank-k approximation to $\mathbf{A}$). In the theory of algorithms, bounds of the form provided by (16) below are known as *additive-error bounds*, the reason being that the additional error is an additive factor of the form ϵ times a size scale that is larger than the "base error" incurred by the best rank-k approximation. In this case, the goal is to minimize the "size scale" of the additional error. Bounds of this form are very different and in general weaker than when the additional error enters as a multiplicative factor, such as when the error bounds are of the form $\|\mathbf{A} - \tilde{\mathbf{A}}\| \leq f(n, k, \eta)\|\mathbf{A} - \mathbf{A}_k\|$, where $f(\cdot)$ is some function and η represents other parameters of the problem. These latter bounds are of greatest interest when $f = 1 + \epsilon$, for an error parameter ϵ, as in (18) and (19) below. These *relative-error bounds*, in which the size scale of the additional error equals that of the base error, provide a *much* stronger notion of approximation than additive-error bounds.

2.2 Preliminaries

In many machine learning and data analysis applications, one is interested in symmetric positive semi-definite (SPSD) matrices, *e.g.*, kernel matrices and Laplacian matrices. One common column-sampling-based approach to low-rank approximation of SPSD matrices is the so-called Nyström method (Williams and Seeger, 2001; Drineas and Mahoney, 2005; Kumar et al., 2012). The Nyström method—both randomized and deterministic variants—has proven useful in applications where the kernel matrices are reasonably well-approximated by low-rank matrices; and it has been applied to Gaussian process regression, spectral clustering and image segmentation, manifold learning, and a range of other common machine learning tasks (Williams and Seeger, 2001; Williams et al., 2002; Fowlkes et al., 2004; Talwalkar et al., 2008; Zhang and Kwok, 2010; Kumar et al., 2012). The simplest Nyström-based procedure selects columns from the original data set uniformly at random and then uses those columns to construct a low-rank SPSD approximation. Although this procedure can be effective in practice for certain input matrices, two extensions (both of which are more expensive) can substantially improve the performance, *e.g.*, lead to lower reconstruction error for a fixed number of column samples, both in theory and in practice. The first extension is to sample columns with a judiciously-chosen nonuniform importance sampling distribution; and the second extension is to randomly mix (or combine linearly) columns before sampling them. For the random sampling algorithms, an important question is what importance sampling distribution should be used to construct the sample; while for the random projection algorithms, an important question is how to implement the random projections. In either case, appropriate consideration should be paid to questions such as

whether the data are sparse or dense, how the eigenvalue spectrum decays, the nonuniformity properties of eigenvectors, *e.g.*, as quantified by the statistical leverage scores, whether one is interested in reconstructing the matrix or performing a downstream machine learning task, and so on.

The following sketching model subsumes both of these classes of methods.

- *SPSD Sketching Model.* Let $\mathbf{A}$ be an $n \times n$ positive semi-definite matrix, and let $\mathbf{S}$ be a matrix of size $n \times \ell$, where $\ell \ll n$. Take

$$\mathbf{C} = \mathbf{AS} \quad \text{and } \mathbf{W} = \mathbf{S}^T\mathbf{AS}.$$

  Then $\mathbf{CW}^\dagger\mathbf{C}^T$ is a low-rank approximation to $\mathbf{A}$ with rank at most ℓ.

We should note that the SPSD Sketching Model, formulated in this way, is *not* guaranteed to be numerically stable: if $\mathbf{W}$ is ill-conditioned, then instabilities may arise in forming the product $\mathbf{CW}^\dagger\mathbf{C}^T$. For simplicity in our presentation, we do not describe the generalizations of our results that could be obtained for the various algorithmic tweaks that have been considered to address this potential issue (Drineas et al., 2008; Mahoney and Drineas, 2009; Chiu and Demanet, 2013).

The choice of distribution for the sketching matrix $\mathbf{S}$ leads to different classes of low-rank approximations. For example, if $\mathbf{S}$ represents the process of column sampling, either uniformly or according to a nonuniform importance sampling distribution, then we refer to the resulting approximation as a Nyström extension; if $\mathbf{S}$ consists of random linear combinations of most or all of the columns of $\mathbf{A}$, then we refer to the resulting approximation as a projection-based SPSD approximation. In this paper, we focus on Nyström extensions and projection-based SPSD approximations that fit the above SPSD Sketching Model. In particular, we do not consider adaptive schemes, which iteratively select columns to progressively decrease the approximation error. While these methods often perform well in practice (Belabbas and Wolfe, 2009b,a; Farahat et al., 2011; Kumar et al., 2012), rigorous analyses of them are hard to come by—interested readers are referred to the discussion in (Farahat et al., 2011; Kumar et al., 2012).

2.3 The Power Method

One can obtain the optimal rank-k approximation to $\mathbf{A}$ by forming an SPSD sketch where the sketching matrix $\mathbf{S}$ is an orthonormal basis for the range of $\mathbf{A}_k$, because with such a choice,

$$\mathbf{CW}^\dagger\mathbf{C}^T = \mathbf{AS}(\mathbf{S}^T\mathbf{AS})^\dagger\mathbf{S}^T\mathbf{A} = \mathbf{A}(\mathbf{SS}^T\mathbf{ASS}^T)^\dagger\mathbf{A} = \mathbf{A}(\mathbf{P}_{\mathbf{A}_k}\mathbf{AP}_{\mathbf{A}_k})^\dagger\mathbf{A} = \mathbf{AA}_k^\dagger\mathbf{A} = \mathbf{A}_k.$$

Of course, one cannot quickly obtain such a basis; this motivates considering sketching matrices $\mathbf{S}_q$ obtained using the power method: that is, taking $\mathbf{S}_q = \mathbf{A}^q\mathbf{S}_0$ where q is a positive integer and $\mathbf{S}_0 \in \mathbb{R}^{n\times\ell}$ with $l \geq k$. As $q \to \infty$, assuming $\mathbf{U}_1^T\mathbf{S}_0$ has full row-rank, the matrices $\mathbf{S}_q$ increasingly capture the dominant k-dimensional eigenspaces of $\mathbf{A}$ (see Golub and Van Loan, 1996, Chapter 8), so one can reasonably expect that the sketching matrix $\mathbf{S}_q$ produces SPSD sketches of $\mathbf{A}$ with lower additional error.

SPSD sketches produced using q iterations of the power method have lower error than sketches produced without using the power method, but are roughly q times more costly to

produce. Thus, the power method is most applicable when $\mathbf{A}$ is such that one can compute the product $\mathbf{A}^q\mathbf{S}_0$ fast. We consider the empirical performance of sketches produced using the power method in Section 3, and we consider the theoretical performance in Section 4.

2.4 Related Prior Work

Motivated by large-scale data analysis and machine learning applications, recent theoretical and empirical work has focused on "sketching" methods such as random sampling and random projection algorithms. A large part of the recent body of this work on randomized matrix algorithms has been summarized in the recent monograph by Mahoney (2011) and the recent review article by Halko et al. (2011). Here, we note that, on the empirical side, both random projection methods (*e.g.*, Bingham and Mannila, 2001; Fradkin and Madigan, 2003; Venkatasubramanian and Wang, 2011; Banerjee et al., 2012) and random sampling methods (*e.g.*, Paschou et al., 2007; Mahoney and Drineas, 2009) have been used in applications for clustering and classification of general data matrices; and that some of this work has highlighted the importance of the statistical leverage scores that we use in this paper (Paschou et al., 2007; Mahoney and Drineas, 2009; Mahoney, 2011; Yip et al., 2014). In parallel, so-called Nyström-based methods have also been used in machine learning applications. Originally used by Williams and Seeger to solve regression and classification problems involving Gaussian processes when the SPSD matrix $\mathbf{A}$ is well-approximated by a low-rank matrix (Williams and Seeger, 2001; Williams et al., 2002), the Nyström extension has been used in a large body of subsequent work. For example, applications of the Nyström method to large-scale machine learning problems include the work of Talwalkar et al. (2008); Kumar et al. (2009a,c); Mackey et al. (2011b) and Zhang et al. (2008); Li et al. (2010); Zhang and Kwok (2010), and applications in statistics and signal processing include the work of Parker et al. (2005); Belabbas and Wolfe (2007a,b); Spendley and Wolfe (2008); Belabbas and Wolfe (2008, 2009b,a).

Much of this work has focused on new proposals for selecting columns (*e.g.*, Zhang et al., 2008; Zhang and Kwok, 2009; Liu et al., 2010; Arcolano and Wolfe, 2010; Li et al., 2010) and/or coupling the method with downstream applications (*e.g.*, Bach and Jordan, 2005; Cortes et al., 2010; Jin et al., 2013; Homrighausen and McDonald, 2011; Machart et al., 2011; Bach, 2013). The most detailed results are provided by Kumar et al. (2012) as well as the conference papers on which it is based (Kumar et al., 2009a,b,c). Interestingly, they observe that uniform sampling performs quite well, suggesting that in the data they considered the leverage scores are quite uniform, which also motivated the related works of Talwalkar and Rostamizadeh (2010); Mohri and Talwalkar (2011). This is in contrast with applications in genetics (Paschou et al., 2007), term-document analysis (Mahoney and Drineas, 2009), and astronomy (Yip et al., 2014), where the statistical leverage scores were seen to be very nonuniform in ways of interest to the downstream scientist; we return to this issue in Section 3.

On the theoretical side, much of the work has followed that of Drineas and Mahoney (2005), who provided the first rigorous bounds for the Nyström extension of a general SPSD matrix. They show that when $\Omega(k\epsilon^{-4}\ln\delta^{-1})$ columns are sampled with an importance sampling distribution that is proportional to the square of the diagonal entries of $\mathbf{A}$, then

$$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_\xi \leq \|\mathbf{A}-\mathbf{A}_k\|_\xi + \epsilon\sum_{k=1}^n(\mathbf{A})_{ii}^2 \tag{4}$$

holds with probability $1-\delta$, where $\xi = 2, F$ represents the Frobenius or spectral norm. (Actually, they prove a stronger result of the form given in Equation (4), except with $\mathbf{W}^\dagger$ replaced with $\mathbf{W}_k^\dagger$, where $\mathbf{W}_k$ represents the best rank-k approximation to $\mathbf{W}$ (Drineas and Mahoney, 2005).) Subsequently, Kumar, Mohri, and Talwalkar show that if $\mu k\ln(k/\delta))$ columns are sampled uniformly at random with replacement from an $\mathbf{A}$ that has *exactly* rank k, then one achieves exact recovery, *i.e.*, $\mathbf{A}=\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T$, with high probability (Kumar et al., 2009a). Gittens (2012) extends this to the case where $\mathbf{A}$ is only approximately low-rank. In particular, he shows that if $\ell=\Omega(\mu k\ln k)$ columns are sampled uniformly at random (either with or without replacement), then

$$\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2 \leq \|\mathbf{A}-\mathbf{A}_k\|_2\left(1+\frac{2n}{\ell}\right) \tag{5}$$

with probability exceeding $1-\delta$ and

$$\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2 \leq \|\mathbf{A}-\mathbf{A}_k\|_2 + \frac{2}{\delta}\cdot\|\mathbf{A}-\mathbf{A}_k\|_\star \tag{6}$$

with probability exceeding $1-2\delta$.

We have described these prior theoretical bounds in detail to emphasize how strong, relative to the prior work, our new bounds are. For example, Equation (4) provides an additive-error approximation with a very large scale; the bounds of Kumar, Mohri, and Talwalkar require a sampling complexity that depends on the coherence of the input matrix (Kumar et al., 2009a), which means that unless the coherence is very low one needs to sample essentially all the rows and columns in order to reconstruct the matrix; Equation (5) provides a bound where the additive scale depends on n; and Equation (6) provides a spectral norm bound where the scale of the additional error is the (much larger) trace norm. Table 1 compares the bounds on the approximation errors of SPSD sketches derived in this work to those available in the literature. We note further that Wang and Zhang recently established lower-bounds on the worst-case relative spectral and trace norm errors of uniform Nyström extensions (Wang and Zhang, 2013). Our Lemma 8 provides matching upper bounds, showing the optimality of these estimates.

A related stream of research concerns projection-based low-rank approximations of general (*i.e.*, non-SPSD) matrices (Halko et al., 2011; Mahoney, 2011). Such approximations are formed by first constructing an approximate basis for the top left invariant subspace of $\mathbf{A}$, and then restricting $\mathbf{A}$ to this space. Algorithmically, one constructs $\mathbf{Y}=\mathbf{A}\mathbf{S}$, where $\mathbf{S}$ is a sketching matrix, then takes $\mathbf{Q}$ to be a basis obtained from the QR decomposition of $\mathbf{Y}$, and then forms the low-rank approximation $\mathbf{Q}\mathbf{Q}^T\mathbf{A}$. The survey paper Halko et al. (2011) proposes two schemes for the approximation of SPSD matrices that fit within this paradigm: $\mathbf{Q}(\mathbf{Q}^T\mathbf{A}\mathbf{Q})\mathbf{Q}^T$ and $(\mathbf{A}\mathbf{Q})(\mathbf{Q}^T\mathbf{A}\mathbf{Q})^\dagger(\mathbf{Q}^T\mathbf{A})$. The first scheme—for which Halko et al. (2011) provides quite sharp error bounds when $\mathbf{S}$ is a matrix of i.i.d. standard Gaussian random variables—has the salutary property of being numerically stable. In Wang and Zhang (2013), the authors show that using the first scheme with an adaptively sampled $\mathbf{S}$ results in approximations with expected Frobenius error within a factor of $1+\epsilon$ of the optimal rank-k approximation error when $\mathrm{O}(k/\epsilon^2)$ columns are sampled.

Halko et al. (2011) does not provide any theoretical guarantees for the second scheme, but observes that this latter scheme produces noticeably more accurate approximations in

Source	ℓ	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_2$	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_F$	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_\star$
Prior works				
Drineas and Mahoney (2005)	$\Omega(\epsilon^{-4}k)$	$\mathrm{opt}_2+\epsilon\sum_{i=}^n A_{ii}^2$	$\mathrm{opt}_F+\epsilon\sum_{i=1}^n A_{ii}^2$	–
Belabbas and Wolfe (2009b)	$\Omega(1)$	–	–	$\mathrm{O}\left(\frac{n-\ell}{n}\right)\|\mathbf{A}\|_\star$
Talwalkar and Rostamizadeh (2010)	$\Omega(\mu_r r\ln r)$	0	0	0
Kumar et al. (2012)	$\Omega(1)$	$\mathrm{opt}_2+\frac{n}{\sqrt{\ell}}\|\mathbf{A}\|_2$	$\mathrm{opt}_F+n(\frac{k}{\ell})^{1/4}\|\mathbf{A}\|_2$	–
This work				
Lemma 8, uniform column sampling	$\Omega\left(\frac{\mu_k k\ln k}{(1-\epsilon)^2}\right)$	$\mathrm{opt}_2(1+\frac{n}{\epsilon\ell})$	$\mathrm{opt}_F+\epsilon^{-1}\mathrm{opt}_\star$	$\mathrm{opt}_\star(1+\epsilon^{-1})$
Lemma 5 leverage-based column sampling	$\Omega\left(\frac{k\ln(k/\beta)}{\beta\epsilon^2}\right)$	$\mathrm{opt}_2+\epsilon^2\mathrm{opt}_\star$	$\mathrm{opt}_F+\epsilon\mathrm{opt}_\star$	$(1+\epsilon^2)\mathrm{opt}_\star$
Lemma 6, Fourier-based projection	$\Omega(\epsilon^{-1}k\ln n)$	$(1+\frac{1}{1-\sqrt{\epsilon}})\mathrm{opt}_2+\frac{\epsilon\mathrm{opt}_\star}{(1-\sqrt{\epsilon})k}$	$\mathrm{opt}_F+\sqrt{\epsilon}\mathrm{opt}_\star$	$(1+\epsilon)\mathrm{opt}_\star$
Lemma 7, Gaussian-based projection	$\Omega(k\epsilon^{-1})$	$(1+\epsilon^2)\mathrm{opt}_2+\frac{\epsilon}{k}\mathrm{opt}_\star$	$\mathrm{opt}_F+\epsilon\mathrm{opt}_\star$	$(1+\epsilon^2)\mathrm{opt}_\star$

Table 1: Comparison of our bounds on the approximation errors of several types of SPSD sketches with those provided in prior works. Only the asymptotically largest terms (as $\epsilon\to 0$) are displayed and constants are omitted, for simplicity. Here, $\epsilon\in(0,1)$, opt_ξ is the smallest ξ-norm error possible when approximating $\mathbf{A}$ with a rank-k matrix ($k\geq\ln n$), $r=\mathrm{rank}(\mathbf{A})$, ℓ is the number of column samples sufficient for the stated bounds to hold, k is a target rank, and μ_s is the coherence of $\mathbf{A}$ relative to the best rank-s approximation to $\mathbf{A}$. The parameter $\beta\in(0,1]$ allows for the possibility of sampling using β-approximate leverage scores (see Section 4.2.1) rather than the exact leverage scores. With the exception of (Drineas and Mahoney, 2005), which samples columns with probability proportional to their Euclidean norms, and our novel leverage-based Nyström bound, these bounds are for sampling columns or linear combinations of columns uniformly at random. All bounds hold with constant probability.

practice. In Section 3, we show this second scheme is an instantiation of the power method (as described in Section 2.3) with $q=1$. Accordingly, the deterministic and stochastic error bounds provided in Section 4 provide theoretical guarantees for this SPSD sketch.

Table 2: The min/mean/max ratios of the errors of several non-rank-restricted SPSD sketches to the optimal rank-k approximation error for several of the matrices considered in Table 4. Here k is the target rank and ℓ is the number of column samples used to form the SPSD sketches. The min/mean/max ratios were computed using 30 trials for each combination of ℓ and sketching method.

Enron, $k=60$

	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_2/\|\mathbf{A}-\mathbf{A}_k\|_2$		
	$\ell=k+8$	$\ell=k\ln n$	$\ell=k\ln n$
Nyström	1.386/1.386/1.386	1.386/1.386/1.386	1.386/1.386/1.386
SRFT sketch	1.378/1.379/1.381	1.357/1.360/1.364	1.310/1.317/1.323
Gaussian sketch	1.378/1.380/1.381	1.357/1.360/1.364	1.314/1.318/1.323
Leverage sketch	1.321/1.381/1.386	1.039/1.188/1.386	1.039/1.042/1.113
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_F/\|\mathbf{A}-\mathbf{A}_k\|_F$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.004/1.004/1.004	0.993/0.994/0.994	0.972/0.972/0.973
SRFT sketch	1.004/1.004/1.004	0.994/0.994/0.994	0.972/0.972/0.972
Gaussian sketch	1.004/1.004/1.004	0.994/0.994/0.994	0.972/0.972/0.972
Leverage sketch	1.002/1.002/1.003	0.994/0.995/0.996	0.988/0.989/0.989
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_\star/\|\mathbf{A}-\mathbf{A}_k\|_\star$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.002/1.002/1.003	0.984/0.984/0.984	0.943/0.944/0.944
SRFT sketch	1.002/1.002/1.002	0.984/0.984/0.984	0.944/0.944/0.944
Gaussian sketch	1.002/1.002/1.002	0.984/0.984/0.984	0.944/0.944/0.944
Leverage sketch	1.002/1.002/1.003	0.990/0.991/0.992	0.977/0.978/0.980

AbaloneD, $\sigma=.15$, $k=20$

	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_2/\|\mathbf{A}-\mathbf{A}_k\|_2$		
	$\ell=k+8$	$\ell=k\ln n$	$\ell=k\ln n$
Nyström	2.168/2.455/2.569	2.022/2.381/2.569	1.823/2.204/2.567
SRFT sketch	2.329/2.416/2.489	2.146/2.249/2.338	1.741/1.840/1.918
Gaussian sketch	2.347/2.409/2.484	2.161/2.254/2.361	1.723/1.822/1.951
Leverage sketch	1.508/1.859/2.377	1.152/1.417/2.036	0.774/0.908/1.091
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_F/\|\mathbf{A}-\mathbf{A}_k\|_F$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.078/1.090/1.098	1.061/1.078/1.091	1.026/1.040/1.054
SRFT sketch	1.088/1.089/1.090	1.074/1.075/1.077	1.034/1.035/1.037
Gaussian sketch	1.087/1.089/1.091	1.073/1.075/1.077	1.033/1.035/1.036
Leverage sketch	1.028/1.040/1.059	0.998/1.006/1.020	0.959/0.963/0.968
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_\star/\|\mathbf{A}-\mathbf{A}_k\|_\star$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.022/1.024/1.026	1.010/1.014/1.016	0.977/0.980/0.983
SRFT sketch	1.024/1.024/1.024	1.014/1.014/1.014	0.980/0.980/0.981
Gaussian sketch	1.024/1.024/1.024	1.014/1.014/1.014	0.980/0.980/0.981
Leverage sketch	1.009/1.012/1.016	0.994/0.997/1.000	0.965/0.968/0.971

Protein, $k=10$

	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_2/\|\mathbf{A}-\mathbf{A}_k\|_2$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.570/2.104/2.197	1.496/2.100/2.196	1.023/1.350/2.050
SRFT sketch	1.835/1.956/2.039	1.686/1.894/2.009	1.187/1.283/1.458
Gaussian sketch	1.812/1.956/2.058	1.653/1.894/2.007	1.187/1.283/1.458
Leverage sketch	1.345/1.644/2.166	1.198/1.498/2.160	0.942/0.994/1.073
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_F/\|\mathbf{A}-\mathbf{A}_k\|_F$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.041/1.054/1.065	1.023/1.042/1.054	0.867/0.877/0.894
SRFT sketch	1.049/1.054/1.058	1.032/1.037/1.043	0.873/0.877/0.880
Gaussian sketch	1.049/1.054/1.060	1.032/1.039/1.043	0.874/0.878/0.883
Leverage sketch	1.027/1.036/1.054	1.011/1.018/1.034	0.862/0.868/0.875
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_\star/\|\mathbf{A}-\mathbf{A}_k\|_\star$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.011/1.014/1.018	0.988/0.994/0.998	0.760/0.764/0.770
SRFT sketch	1.013/1.015/1.016	0.990/0.993/0.995	0.762/0.764/0.766
Gaussian sketch	1.013/1.015/1.017	0.991/0.993/0.994	0.762/0.765/0.767
Leverage sketch	1.004/1.008/1.014	0.982/0.985/0.991	0.758/0.765/0.771

WineS, $\sigma=1$, $k=20$

	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_2/\|\mathbf{A}-\mathbf{A}_k\|_2$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.989/2.001/2.002	1.987/1.998/2.002	1.739/1.978/2.002
SRFT sketch	1.910/1.938/1.966	1.840/1.873/1.905	1.624/1.669/1.709
Gaussian sketch	1.903/1.942/1.966	1.839/1.873/1.910	1.619/1.670/1.707
Leverage sketch	1.242/1.762/1.995	1.000/1.317/1.987	1.000/1.000/1.005
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_F/\|\mathbf{A}-\mathbf{A}_k\|_F$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.036/1.040/1.043	1.028/1.034/1.038	0.998/1.009/1.018
SRFT sketch	1.038/1.039/1.039	1.029/1.030/1.030	1.000/1.000/1.001
Gaussian sketch	1.038/1.039/1.039	1.029/1.030/1.030	1.000/1.000/1.001
Leverage sketch	1.004/1.011/1.018	0.996/1.000/1.005	0.994/0.995/0.997
	$\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_\star/\|\mathbf{A}-\mathbf{A}_k\|_\star$		
	$\ell=k+8$	$\ell=k\ln k$	$\ell=k\ln n$
Nyström	1.013/1.015/1.016	1.002/1.005/1.007	0.965/0.970/0.976
SRFT sketch	1.014/1.014/1.015	1.004/1.004/1.004	0.970/0.970/0.970
Gaussian sketch	1.014/1.014/1.015	1.004/1.004/1.004	0.970/0.970/0.970
Leverage sketch	1.002/1.005/1.009	0.997/0.999/1.002	0.995/0.996/0.997

source, sketch	pred./obs. spectral error	pred./obs. Frobenius error	pred./obs. trace error
Enron, $k = 60$			
Drineas and Mahoney (2005) nonuniform column sampling	3041.0	66.2	–
Belabbas and Wolfe (2009b) uniform column sampling	–	–	2.0
Kumar et al. (2012) uniform column sampling	331.2	77.7	–
Lemma 5 leverage-based	1287.0	20.5	1.2
Lemma 6 Fourier-based	102.1	42.0	1.6
Lemma 7 Gaussian-based	20.1	7.6	1.4
Lemma 8 uniform column sampling	9.4	285.1	9.5
Protein, $k = 10$			
Drineas and Mahoney (2005), nonuniform column sampling	125.2	18.6	–
Belabbas and Wolfe (2009b), uniform column sampling	–	–	3.6
Kumar et al. (2012), uniform column sampling	35.1	20.5	–
Lemma 5, leverage-based	42.4	6.2	2.0
Lemma 6, Fourier-based	155.0	20.4	3.1
Lemma 7, Gaussian-based	5.7	5.6	2.2
Lemma 8, uniform column sampling	90.0	63.4	14.3
AbaloneD, $\sigma = .15, k = 20$			
Drineas and Mahoney (2005), nonuniform column sampling	360.8	42.5	–
Belabbas and Wolfe (2009b), uniform column sampling	–	–	2.0
Kumar et al. (2012), uniform column sampling	62.0	45.7	–
Lemma 5, leverage-based	235.4	14.1	1.3
Lemma 6, Fourier-based	70.1	36.0	1.7
Lemma 7, Gaussian-based	8.7	8.3	1.3
Lemma 8, uniform column sampling	13.2	166.2	9.0
WineS, $\sigma = 1, k = 20$			
Drineas and Mahoney (2005), nonuniform column sampling	408.4	41.1	–
Belabbas and Wolfe (2009b), uniform column sampling	–	–	2.1
Kumar et al. (2012), uniform column sampling	70.3	44.3	–
Lemma 5, leverage-based	244.6	12.9	1.2
Lemma 6, Fourier-based	94.8	36.0	1.7
Lemma 7, Gaussian-based	11.4	8.1	1.4
Lemma 8, uniform column sampling	13.2	162.2	9.1

Table 3: Comparison of the empirically observed approximation errors to the guarantees provided in this and other works, for several data sets. Each approximation was formed using $\ell = 6k \ln k$ samples. To evaluate the error guarantees, $\delta = 1/2$ was taken and all constants present in the statements of the bounds were replaced with ones. The observed errors were taken to be the average errors over 30 runs of the approximation algorithms. The data sets, described in Section 3.1, are representative of several classes of matrices prevalent in machine learning applications.

2.5 An Overview of Our Bounds

Our bounds in Table 1 (established as Lemmas 5–8 in Section 4.2) exhibit a common structure: for the spectral and Frobenius norms, we see that the additional error is on a larger scale than the optimal error, and the trace norm bounds all guarantee relative error approximations. This follows from the fact, as detailed in Section 4.1, that low-rank approximations that conform to the SPSD sketching model can be understood as forming column-sample/projection-based approximations to the *square root* of $\mathbf{A}$, and thus squaring this approximation yields the resulting approximation to $\mathbf{A}$. The squaring process unavoidably results in potentially large additional errors in the case of the spectral and Frobenius norms— whether or not the additional errors are large in practice depends upon the properties of the matrix and the form of stochasticity used in the sampling process. For instance, from our bounds it is clear that Gaussian-based SPSD sketches are expected to have lower additional error in the spectral norm than any of the other sketches considered.

From Table 1, we also see, in the case of uniform Nyström extensions, a necessary dependence on the coherence of the input matrix since columns are sampled uniformly at random. However, we also see that the scales of the additional error of the Frobenius and trace norm bounds are substantially improved over those in prior results. The large additional error in the spectral norm error bound is necessary in the worse case (Gittens, 2012). Lemmas 5, 6 and 7 in Section 4.2—which respectively address leverage-based, Fourier-based, and Gaussian-based SPSD sketches—show that spectral norm additive-error bounds with additional error on a substantially smaller scale can be obtained if one first mixes the columns before sampling from $\mathbf{A}$ or one samples from a judicious nonuniform distribution over the columns.

Table 2 compares the minimum, mean, and maximum approximation errors of several SPSD sketches of four matrices (described in Section 3.1) to the optimal rank-k approximation errors. We consider three regimes for ℓ, the number of column samples used to construct the sketch: $\ell = \mathrm{O}(k)$, $\ell = \mathrm{O}(k \ln k)$, and $\ell = \mathrm{O}(k \ln n)$. These matrices exhibit a diverse range of properties: *e.g.*, Enron is sparse and has a slowly decaying spectrum, while Protein is dense and has a rapidly decaying spectrum. Yet we notice that the sketches perform quite well on each of these matrices. In particular, when $\ell = \mathrm{O}(k \ln n)$, the average errors of the sketches are within $1 + \epsilon$ of the optimal rank-k approximation errors, where $\epsilon \in [0, 1]$. Also note that the leverage-based sketches consistently have lower average errors (in all of the three norms considered) than all other sketches. Likewise, the uniform Nyström extensions usually have larger average errors than the other sketches. These two sketches represent opposite extremes: uniform Nyström extensions (constructed using uniform column sampling) are constructed using no knowledge about the matrix, while leverage-based sketches use an importance sampling distribution derived from the SVD of the matrix to determine which columns to use in the construction of the sketch.

Table 3 illustrates the gap between the theoretical results currently available in the literature and what is observed in practice: it depicts the ratio between the error bounds in Table 1 and the average errors observed over 30 runs of the SPSD approximation algorithms (the error bound from (Talwalkar and Rostamizadeh, 2010) is not considered in the table, as it does not apply at the number of samples ℓ used in the experiments). Several trends can be identified; among them, we note that the bounds provided in this paper for Gaussian-based

sketches come quite close to capturing the errors seen in practice, and the Frobenius and trace norm error guarantees of the leverage-based and Fourier-based sketches tend to more closely reflect the empirical behavior than the error guarantees provided in prior work for Nyström sketches. Overall, the trace norm error bounds are quite accurate. On the other hand, prior bounds are sometimes more informative in the case of the spectral norm (with the notable exception of the Gaussian sketches). Several important points can be gleaned from these observations. First, the accuracy of the Gaussian error bounds suggests that the main theoretical contribution of this work, the deterministic structural results given as Theorems 2 through 4, captures the underlying behavior of the SPSD sketching process. This supports our belief that this work provides a foundation for truly informative error bounds. Given that this is the case, it is clear that the analysis of the stochastic elements of the SPSD sketching process is much sharper in the Gaussian case than in the leverage-score, Fourier, and uniform Nyström cases. We expect that, at least in the case of leverage and Fourier-based sketches, the stochastic analysis can and will be sharpened to produce error guarantees almost as informative as the ones we have provided for Gaussian-based sketches.

3. Empirical Aspects of SPSD Low-rank Approximation

In this section, we present our main empirical results, which consist of evaluating sampling and projection algorithms applied to a diverse set of SPSD matrices. The bulk of our empirical evaluation considers two random projection procedures and two random sampling procedures for the sketching matrix $\mathbf{S}$: for random projections, we consider using SRFTs (Subsampled Randomized Fourier Transforms) as well as uniformly sampling from Gaussian mixtures of the columns; and for random sampling, we consider sampling columns uniformly at random as well as sampling columns according to a nonuniform importance sampling distribution that depends on the empirical statistical leverage scores. In the latter case of leverage score-based sampling, we also consider the use of both the (naïve and expensive) exact algorithm as well as a (recently-developed fast) approximation algorithm. Section 3.1 starts with a brief description of the data sets we consider; Section 3.2 describes the details of our SPSD sketching algorithms; Section 3.3 summarizes our experimental results to help guide in the selection of sketching methods; in Section 3.4, we present our main results on reconstruction quality for the random sampling and random projection methods; and, in Section 3.5, we discuss running time issues, and we present our main results for running time and reconstruction quality for both exact and approximate versions of leverage-based sampling.

We emphasize that we don't intend these results to be "comprehensive" but instead to be "illustrative" case-studies—that are representative of a much wider range of applications than have been considered previously. In particular, we would like to illustrate the tradeoffs between these methods in different realistic applications in order, *e.g.*, to provide directions for future work. In addition to clarifying some of these issues, our empirical evaluation also illustrates ways in which existing theory is insufficient to explain the success of sampling and projection methods. This motivates our improvements to existing theory that we describe in Section 4.

All of our computations were conducted using 64-bit MATLAB R2012a under Ubuntu on a 2.6–GHz quad-core Intel i7 machine with 6Gb of RAM. To allow for accurate timing

comparisons, all computations were carried out in a single thread. When applied to an $n \times n$ SPSD matrix $\mathbf{A}$, our implementation of the SRFT requires $\mathrm{O}(n^2 \ln n)$ operations, as it applies MATLAB's `fft` to the entire matrix $\mathbf{A}$ and *then* it samples ℓ columns from the resulting matrix. A more rigorous implementation of the SRFT algorithm could reduce this running time to $\mathrm{O}(n^2 \ln \ell)$, but due to the complexities involved in optimizing pruned FFT codes, we did not pursue this avenue.

3.1 Data Sets

Table 4 provides summary statistics for the data sets used in our empirical evaluation. We consider four classes of matrices commonly encountered in machine learning and data analysis applications: normalized Laplacians of very sparse graphs drawn from "informatics graph" applications; dense matrices corresponding to Linear Kernels from machine learning applications; dense matrices constructed from a Gaussian Radial Basis Function Kernel (RBFK); and sparse RBFK matrices constructed using Gaussian radial basis functions, truncated to be nonzero only for nearest neighbors. This collection of data sets represents a wide range of data sets with very different (sparsity, spectral, leverage score, etc.) properties that have been of interest recently not only in machine learning but in data analysis more generally.

To understand better the Laplacian data, recall that, given an undirected graph with weighted adjacency matrix $\mathbf{W}$, its normalized graph Laplacian is

$$\mathbf{A} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2},$$

where $\mathbf{D}$ is the diagonal matrix of weighted degrees of the nodes of the graph, *i.e.*, $D_{ii} = \sum_{j \neq i} W_{ij}$.

The remaining data sets are kernel matrices associated with data drawn from a variety of application areas. Recall that, given given points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and a function $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, the $n \times n$ matrix with elements

$$A_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

is called the kernel matrix of κ with respect to $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Appropriate choices of κ ensure that $\mathbf{A}$ is positive semidefinite. When this is the case, the entries A_{ij} can be interpreted as measuring, in a sense determined by the choice of κ, the similarity of points i and j. Specifically, if $\mathbf{A}$ is SPSD, then κ determines a so-called *feature map* $\Phi_\kappa : \mathbb{R}^d \to \mathbb{R}^n$ such that

$$A_{ij} = \langle \Phi_\kappa(\mathbf{x}_i), \Phi_\kappa(\mathbf{x}_j) \rangle$$

measures the similarity (correlation) of $\mathbf{x}_i$ and $\mathbf{x}_j$ in feature space (Schölkopf and Smola, 2001).

When κ is the usual Euclidean inner-product, so that

$$A_{ij} = \langle \mathbf{x}_i, \mathbf{x}_k \rangle,$$

$\mathbf{A}$ is called a Linear Kernel matrix. Gaussian RBFK matrices, defined by

$$A^\sigma_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\sigma^2}\right),$$

Name	Description	n	d	%nnz
Laplacian Kernels				
HEP	arXiv High Energy Physics collaboration graph	9877	NA	0.06
GR	arXiv General Relativity collaboration graph	5242	NA	0.12
Enron	subgraph of the Enron email graph	10000	NA	0.22
Gnutella	Gnutella peer to peer network on Aug. 6, 2002	8717	NA	0.09
Linear Kernels				
Dexter	bag of words	2000	20000	83.8
Protein	derived feature matrix for *S. cerevisiae*	6621	357	99.7
SNPs	DNA microarray data from cancer patients	5520	43	100
Gisette	images of handwritten digits	6000	5000	100
Dense RBF Kernels				
AbaloneD	physical measurements of abalones	4177	8	100
WineD	chemical measurements of wine	4898	12	100
Sparse RBF Kernels				
AbaloneS	physical measurements of abalones	4177	8	82.9/48.1
WineS	chemical measurements of wine	4898	12	11.1/88.0

Table 4: The data sets used in our empirical evaluation (Leskovec et al., 2007; Klimt and Yang, 2004; Guyon et al., 2005; Gustafson et al., 2006; Nielsen et al., 2002; Corke, 1996; Asuncion and Newman, 2012). Here, n is the number of data points, d is the number of features in the input space before kernelization, and %nnz is the percentage of nonzero entries in the matrix. For Laplacian "kernels," n is the number of nodes in the graph (and thus there is no d since the graph is "given" rather than "constructed"). The %nnz for the Sparse RBF Kernels depends on the σ parameter; see Table 5.

correspond to the similarity measure $\kappa(\mathbf{x},\mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|_2^2/\sigma^2)$. Here σ, a nonnegative number, defines the scale of the kernel. Informally, σ defines the "size scale" over which pairs of points $\mathbf{x}_i$ and $\mathbf{x}_j$ "see" each other. Typically σ is determined by a global cross-validation criterion, as $\mathbf{A}^\sigma$ is generated for some specific machine learning task; and, thus, one may have no *a priori* knowledge of the behavior of the spectrum or leverage scores of $\mathbf{A}^\sigma$ as σ is varied. Accordingly, we consider Gaussian RBFK matrices with different values of σ.

Finally, given the same data points, $\mathbf{x}_1,\ldots,\mathbf{x}_n$, one can construct sparse Gaussian RBFK matrices

$$A_{ij}^{(\sigma,\nu,C)} = \left[\left(1-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|_2}{C}\right)^\nu\right]^+ \cdot \exp\left(\frac{-\|\mathbf{x}_i-\mathbf{x}_j\|_2^2}{\sigma^2}\right),$$

where $[x]^+ = \max\{0,x\}$. When ν is larger than $(d+1)/2$, this kernel matrix is positive semidefinite (Genton, 2002). Increasing ν shrinks the magnitudes of the off-diagonal entries of the matrix toward zero. As the cutoff point C decreases the matrix becomes more sparse; in particular, $C \to 0$ ensures that $\mathbf{A}^{(\sigma,\nu,C)} \to \mathbf{I}$. On the other hand, $C \to \infty$ ensures that

Name	%nnz	$\lceil \frac{\|\mathbf{A}\|_F^2}{\|\mathbf{A}\|_2^2} \rceil$	k	$\frac{\lambda_{k+1}}{\lambda_k}$	$100\frac{\|\mathbf{A}-\mathbf{A}_k\|_F}{\|\mathbf{A}\|_F}$	$100\frac{\|\mathbf{A}-\mathbf{A}_k\|_\star}{\|\mathbf{A}\|_\star}$	kth-largest leverage score scaled by n/k
HEP	0.06	3078	20	0.998	7.8	0.4	128.8
HEP	0.06	3078	60	0.998	13.2	1.1	41.9
GR	0.12	1679	20	0.999	10.5	0.74	71.6
GR	0.12	1679	60	1	17.9	2.16	25.3
Enron	0.22	2588	20	0.997	7.77	0.352	245.8
Enron	0.22	2588	60	0.999	12.0	0.94	49.6
Gnutella	0.09	2757	20	1	8.1	0.41	166.2
Gnutella	0.09	2757	60	0.999	13.7	1.20	49.4
Dexter	83.8	176	8	0.963	14.5	.934	16.6
Protein	99.7	24	10	0.987	42.6	7.66	5.45
SNPs	100	3	5	0.928	85.5	37.6	2.64
Gisette	100	4	12	0.90	90.1	14.6	2.46
AbaloneD (dense, $\sigma = .15$)	100	41	20	0.992	42.1	3.21	18.11
AbaloneD (dense, $\sigma = 1$)	100	4	20	0.935	97.8	59	2.44
WineD (dense, $\sigma = 1$)	100	31	20	0.99	43.1	3.89	26.2
WineD (dense, $\sigma = 2.1$)	100	3	20	0.936	94.8	31.2	2.29
AbaloneS (sparse, $\sigma = .15$)	82.9	400	20	0.989	15.4	1.06	48.4
AbaloneS (sparse, $\sigma = 1$)	48.1	5	20	0.982	90.6	21.8	3.57
WineS (sparse, $\sigma = 1$)	11.1	116	20	0.995	29.5	2.29	49.0
WineS (sparse, $\sigma = 2.1$)	88.0	39	20	0.992	41.6	3.53	24.1

Table 5: Summary statistics for the data sets from Table 4 that we used in our empirical evaluation.

$\mathbf{A}^{(\sigma,\nu,C)}$ approaches the (dense) Gaussian RBFK matrix $\mathbf{A}^\sigma$. For simplicity, in our empirical evaluations, we fix $\nu = \lceil (d+1)/2 \rceil$ and $C = 3\sigma$, and we vary σ.

To illustrate the diverse range of properties exhibited by these four classes of data sets, consider Table 5. Several observations are particularly relevant to our discussion below.

- All of the Laplacian Kernels drawn from informatics graph applications are extremely sparse in terms of number of nonzeros, and they all tend to have very slow spectral decay, as illustrated both by the quantity $\lceil \|\mathbf{A}\|_F^2 / \|\mathbf{A}\|_2^2 \rceil$ (this is the *stable rank*, which is a numerically stable (under)estimate of the rank of $\mathbf{A}$) as well as by the relatively small fraction of the Frobenius norm that is captured by the best rank-k approximation to $\mathbf{A}$.

- Both the Linear Kernels and the Dense RBF Kernels are much denser and are much more well-approximated by moderately to very low-rank matrices. In addition, both the Linear Kernels and the Dense RBF Kernels have statistical leverage scores that are much more uniform—there are several ways to illustrate this, none of them perfect. Here, we illustrate this by considering the k^{th} largest leverage score, scaled by the factor n/k (if $\mathbf{A}$ were exactly rank k, this would be the coherence of $\mathbf{A}$). For the Linear Kernels and the Dense RBF Kernels, this quantity is typically one to two orders of magnitude smaller than for the Laplacian Kernels.

- For the Dense RBF Kernels, we consider two values of the σ parameter, again chosen (somewhat) arbitrarily. For both AbaloneD and WineD, we see that decreasing σ from 1 to 0.15, *i.e.*, letting data points "see" fewer nearby points, has two important effects: first, it results in matrices that are much *less* well-approximated by low-rank matrices; and second, it results in matrices that have *much* more heterogeneous leverage scores.

- For the Sparse RBF Kernels, there are a range of sparsities, ranging from above the sparsity of the sparsest Linear Kernel, but all are denser than the Laplacian Kernels. Changing the σ parameter has the same effect (although it is even more pronounced) for Sparse RBF Kernels as it has for Dense RBF Kernels. In addition, "sparsifying" a Dense RBF Kernel also has the effect of making the matrix less well approximated by a low-rank matrix and of making the leverage scores more nonuniform.

As we see below, when we consider the RBF Kernels as the width parameter and sparsity are varied, we observe a range of intermediate cases between the extremes of the ("nice") Linear Kernels and the (very "non-nice") Laplacian Kernels.

3.2 SPSD Sketching Algorithms

The sketching matrix $\mathbf{S}$ may be selected in a variety of ways. For sampling-based sketches, the sketching matrix $\mathbf{S}$ contains exactly one nonzero in each column, corresponding to a single sample from the columns of $\mathbf{A}$. For projection-based sketches, $\mathbf{S}$ is dense, and mixes the columns of $\mathbf{A}$ before sampling from the resulting matrix.

In more detail, we consider two types of sampling-based SPSD sketches (*i.e.* Nyström extensions): those constructed by sampling columns uniformly at random with replacement, and those constructed by sampling columns from a distribution based upon the leverage scores of the matrix filtered through the optimal rank-k approximation of the matrix. In the case of column sampling, the sketching matrix $\mathbf{S}$ is simply the first ℓ columns of a matrix that was chosen uniformly at random from the set of all permutation matrices.

In the case of leverage-based sampling, $\mathbf{S}$ has a more complicated distribution. Recall that the leverage scores relative to the best rank-k approximation to $\mathbf{A}$ are the squared Euclidean norms of the rows of the $n \times k$ matrix $\mathbf{U}_1$:

$$\ell_j = \|(\mathbf{U}_1)_j\|^2.$$

It follows from the orthonormality of $\mathbf{U}_1$ that $\sum_j (\ell_j/k) = 1$, and the leverage scores can thus be interpreted as a probability distribution over the columns of $\mathbf{A}$. To construct a sketching matrix corresponding to sampling from this distribution, we first select the columns to be used by sampling with replacement from this distribution. Then, $\mathbf{S}$ is constructed as $\mathbf{S} = \mathbf{RD}$ where $\mathbf{R} \in \mathbb{R}^{n\times\ell}$ is a column selection matrix that samples columns of $\mathbf{A}$ from the given distribution—*i.e.*, $\mathbf{R}_{ij} = 1$ iff the ith column of $\mathbf{A}$ is the jth column selected—and $\mathbf{D}$ is a diagonal rescaling matrix satisfying $\mathbf{D}_{jj} = \frac{1}{\sqrt{\ell p_i}}$ iff $\mathbf{R}_{ij} = 1$. Here, $p_i = \ell_i/k$ is the probability of choosing the ith column of $\mathbf{A}$. It is often expensive to compute the leverage scores exactly; in Section 3.5, we consider the performance of sketches based on several leverage score approximation algorithms.

The two projection-based sketches we consider use Gaussians and the real Fourier transform. In the former case, $\mathbf{S}$ is a matrix of i.i.d. $\mathcal{N}(0,1)$ random variables. In the latter case,

$\mathbf{S}$ is a *subsampled randomized Fourier transform* (SRFT) matrix; that is, $\mathbf{S} = \sqrt{\frac{n}{\ell}}\mathbf{DFR}$, where $\mathbf{D}$ is a diagonal matrix of Rademacher random variables, $\mathbf{F}$ is the real Fourier transform matrix, and $\mathbf{R}$ restricts to ℓ columns.

For conciseness, we do not present results for sampling-based sketches where rows are selected with probability proportional to their row norms. This form of sampling can be similar to leverage-score sampling for sparse graphs with highly connected vertices (Mahoney and Drineas, 2009), and in cases where the matrix has been preprocessed to have uniform row lengths, reduces to uniform sampling.

In the figures, we refer to sketches constructed by selecting columns uniformly at random with the label 'unif', leverage score-based sketches with 'lev', Gaussian sketches with 'gaussian', and Fourier sketches with 'srft'.

3.3 Guidelines for Selecting Sketching Schemes

In the remainder of this section of the paper, we provide empirical evaluations of the sampling and projection-based sketching schemes just described, with an eye towards identifying the aspects of the datasets that affect the relative performance of the sketching schemes. However our experiments also provide some practical guidelines for selecting a particular sketching scheme.

- Despite the theoretical result that the worst-case spectral error in using Nyström sketches obtained via uniform column-samples can be much worse than that of using projection or leverage-based sketches, on the corpus of data sets we considered, such sketches perform within a small multiple of the error of more computationally expensive leverage-based and projection-based sketches. For data sets with more nonuniform leverage score properties, random projections and leverage-based sampling will do better (Ma et al., 2014).

- In the case where parsimony of the sketch is of primary concern, *i.e.* where the primary concern is to maintain $\ell \approx k$, leverage sketches are an attractive option. In particular, when an RBF kernel with small bandwidth is used, or the data set is sparse, leverage-based sketches often provide higher accuracy than projection or uniform-sampling based sketches.

- The norm in which the error is measured should be taked into consideration when selecting the sketching algorithm. In particular, sketches which use power iterations are most useful when the error is measured in the spectral norm, and in this case, projection-based sketches (in particular, *prolonged* sketches– see Section 3.6) noticeably outperform uniform sampling-based sketches.

3.4 Reconstruction Accuracy of Sampling and Projection Algorithms

Here, we describe the performances of the SPSD sketches described in Section 3.2—column sampling uniformly at random without replacement, column sampling according to the nonuniform leverage score probabilities, and sampling using Gaussian and SRFT mixtures of the columns—in terms of reconstruction accuracy for the data sets described in Section 3.1. We describe general observations we have made about each class of matrices in

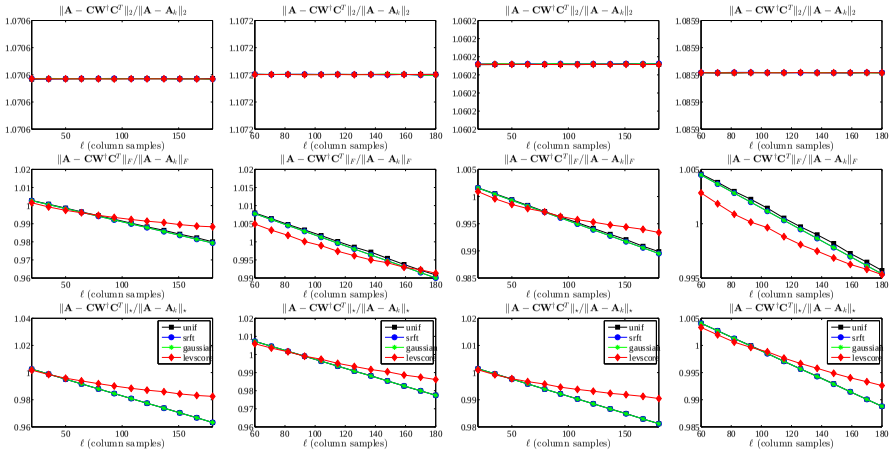
turn, and then we summarize our observations. We consider only the use of exact leverage scores here, and we postpone until Section 3.5 a discussion of running time issues and similar reconstruction results when approximate leverage scores are used for the importance sampling distribution. The relative errors

$$\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\right\|_{\xi} / \left\|\mathbf{A}-\mathbf{A}_{k}\right\|_{\xi} \tag{7}$$

are plotted, with each point in the figures of this section representing the average errors observed over 30 trials.

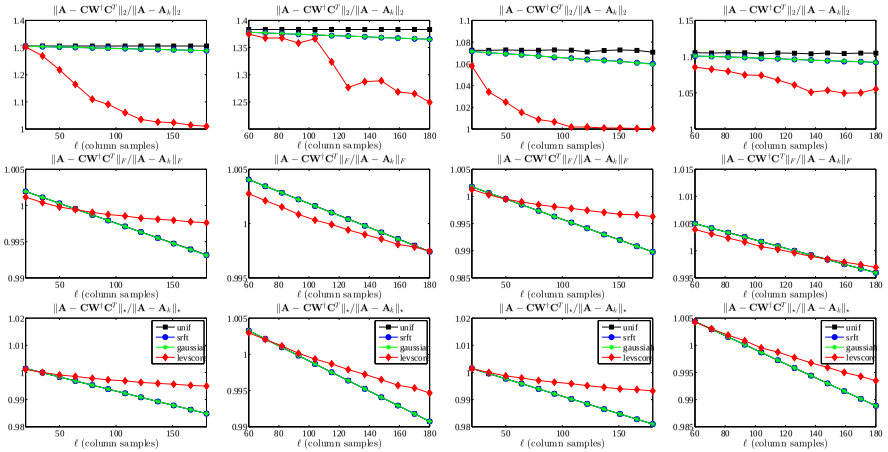
3.4.1 Graph Laplacians

Figure 1 and Figure 2 show the reconstruction error results for sampling and projection methods applied to several normalized graph Laplacians. The former shows GR and HEP, each for two values of the rank parameter, and the latter shows Enron and Gnutella, again each for two values of the rank parameter. Both figures show the spectral, Frobenius, and trace norm approximation errors, as a function of the number of column samples ℓ, relative to the error of the optimal rank-k approximation of $\mathbf{A}$.



(a) GR, $k = 20$ (b) GR, $k = 60$ (c) HEP, $k = 20$ (d) HEP, $k = 60$

Figure 1: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several SPSD sketches, as a function of the number of column samples ℓ, for the GR and HEP Laplacian data sets, with two choices of the rank parameter k.



(a) Enron, $k = 20$ (b) Enron, $k = 60$ (c) Gnutella, $k = 20$ (d) Gnutella, $k = 60$

Figure 2: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several SPSD sketches, as a function of the number of column samples ℓ, for the Enron and Gnutella Laplacian data sets, with two choices of the rank parameter k.

These and subsequent figures contain a lot of information, some of which is peculiar to the given data sets and some of which is more general. In light of subsequent discussion, several observations are worth making about the results presented in these two figures.

- All of the SPSD sketches provide quite accurate approximations—relative to the best possible approximation factor for that norm, and relative to bounds provided by existing theory, as reviewed in Section 2.4—even with only k column samples (or in the case of the Gaussian and SRFT mixtures, with only k linear combinations of columns). Upon examination, this is partly due to the extreme sparsity and extremely slow spectral decay of these data sets which means, as shown in Table 4, that only a small fraction of the (spectral or Frobenius or trace) mass is captured by the optimal rank 20 or 60 approximation. Thus, although an SPSD sketch constructed from 20 or 60 vectors also only captures a small portion of the mass of the matrix, the relative error is small, since the scale of the residual error is large.

- The scale of the Y axes is different between different figures and subfigures. This is to highlight properties within a given plot, but it can hide several things. In particular, note that the scale for the spectral norm is generally larger than for the Frobenius norm, which is generally larger than for the trace norm, consistent with the size of those norms; and that the scale is larger for higher-rank approximations, *e.g.* compare

GR $k = 20$ with GR $k = 60$. This is also consistent with the larger amount of mass captured by higher-rank approximations.

- For $\ell > k$, the errors tend to decrease (or at least not increase, as for GR and HEP the spectral norm error is flat as a function of ℓ), which is intuitive.

- The X axes ranges from k to $9k$ for the $k = 20$ plots and from k to $3k$ for the $k = 60$ plots. As a practical matter, choosing ℓ between k and (say) $2k$ or $3k$ is probably of greatest interest. In this regime, there is an interesting tradeoff: for moderately large values of ℓ in this regime, the error for leverage-based sampling is moderately better than for uniform sampling or random projections, while if one chooses ℓ to be much larger then the improvements from leverage-based sampling saturate and the uniform sampling and random projection methods are better. This is most obvious in the Frobenius norm plots, although it is also seen in the trace norm plots, and it suggests that some combination of leverage-based sampling and uniform sampling might be best.

- The behavior of the approximations with respect to the spectral norm is quite different from the behavior in the Frobenius and trace norms. In the latter, as the number of samples ℓ increases, the errors tend to decrease; while for the former, the errors tend to be much flatter as a function of increasing ℓ for at least the Gaussian, SRFT, and uniformly sampled sketches.

All in all, there seems to be quite complicated behavior for low-rank sketches for these Laplacian data sets. Several of these observations can also be made for subsequent figures; but in some other cases the (very sparse and not very low rank) structural properties of the data are primarily responsible.

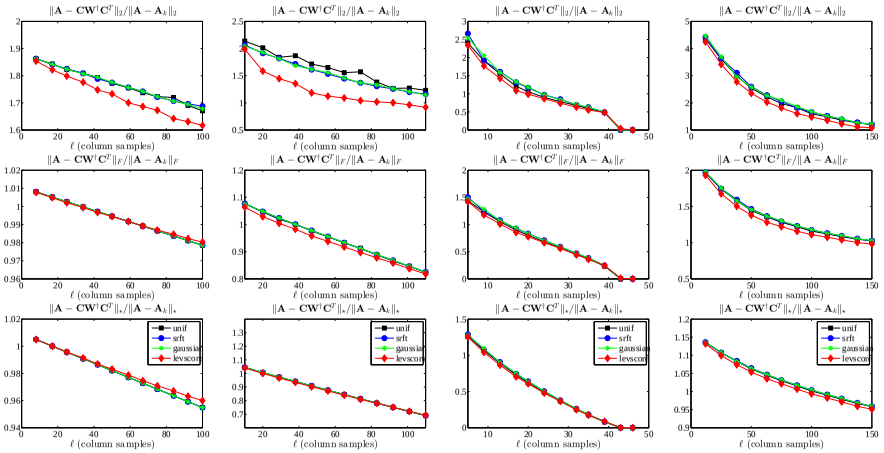
3.4.2 Linear Kernels

Figure 3 shows the reconstruction error results for sampling and projection methods applied to several Linear Kernels. The data sets (Dexter, Protein, SNPs, and Gisette) are all quite low-rank and have fairly uniform leverage scores. Several observations are worth making about the results presented in this figure.

- All of the methods perform quite similarly: all have errors that decrease smoothly with increasing ℓ, and in this case there is little advantage to using methods other than uniform sampling (since they perform similarly and are more expensive). Also, since the ranks are so low and the leverage scores are so uniform, the leverage score sketch is no longer significantly distinguished by its tendency to saturate quickly.

- The scale of the Y axes is much larger than for the Laplacian data sets, mostly since the matrices are much more well-approximated by low-rank matrices, although the scale decreases as one goes from spectral to Frobenius to trace reconstruction error, as before.

These linear kernels (and also to some extent the dense RBF kernels below that have larger σ parameter) are examples of relatively "nice" machine learning data sets that are similar



(a) Dexter, $k = 8$ (b) Protein, $k = 10$ (c) SNPs, $k = 5$ (d) Gisette, $k = 12$

Figure 3: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several SPSD sketches, as a function of the number of column samples ℓ, for the Linear Kernel data sets.

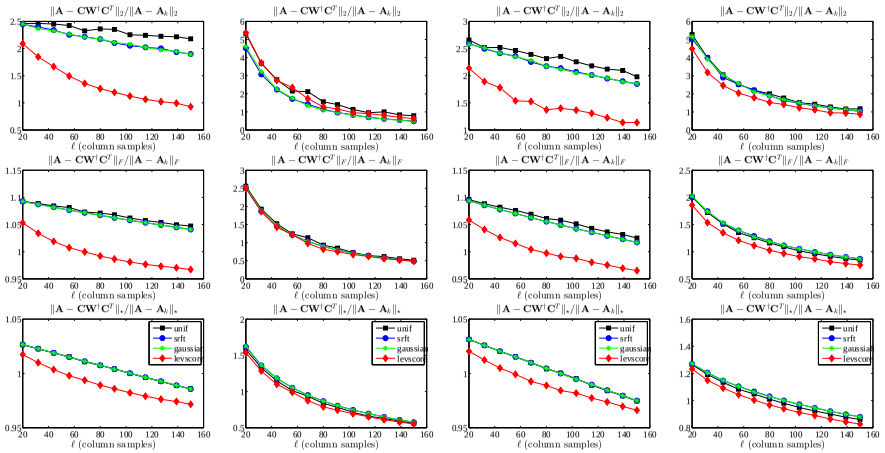
to matrices where uniform sampling has been shown to perform well previously (Talwalkar et al., 2008; Kumar et al., 2009a,c, 2012); for these matrices our empirical results agree with these prior works.

3.4.3 Dense and Sparse RBF Kernels

Figure 4 and Figure 5 present the reconstruction error results for sampling and projection methods applied to several dense RBF and sparse RBF kernels. Several observations are worth making about the results presented in these figures.

- All of the methods have errors that decrease with increasing ℓ, but for larger values of σ and for denser data, the decrease is somewhat more regular, and the four methods tend to perform similarly. For larger values of σ and sparser data, leverage score sampling is somewhat better. This parallels what we observed with the Linear Kernels, except that here the leverage score sampling is somewhat better for all values of ℓ.

- For smaller values of σ, leverage score sampling tends to be much better than uniform sampling and projection-based methods. For sparse data, however, this effect saturates; and we again observe (especially when σ is smaller in AbaloneS and WineS) the tradeoff we observed previously with the Laplacian data—leverage score sampling is better when ℓ is moderately larger than k, while uniform sampling and random projections are better when ℓ is much larger than k.



(a) AbaloneD, $\sigma = .15, k = 20$ (b) AbaloneD, $\sigma = 1, k = 20$ (c) WineD, $\sigma = 1, k = 20$ (d) WineD, $\sigma = 2.1, k = 20$

Figure 4: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several SPSD sketches, as a function of the number of column samples ℓ, for several dense RBF data sets.

Recall from Table 5 that for smaller values of σ and for sparser kernels, the SPSD matrices are less well-approximated by low-rank matrices, and they have more heterogeneous leverage scores. Thus, they are more similar to the Laplacian data than the Linear Kernel data; this suggests (as we have observed) that leverage score sampling should perform relatively better than uniform column sampling and projection-based schemes when in these two cases.

3.4.4 Summary of Comparison of Sampling and Projection Algorithms

Before proceeding, there are several summary observations that we can make about sampling versus projection methods for the data sets we have considered.

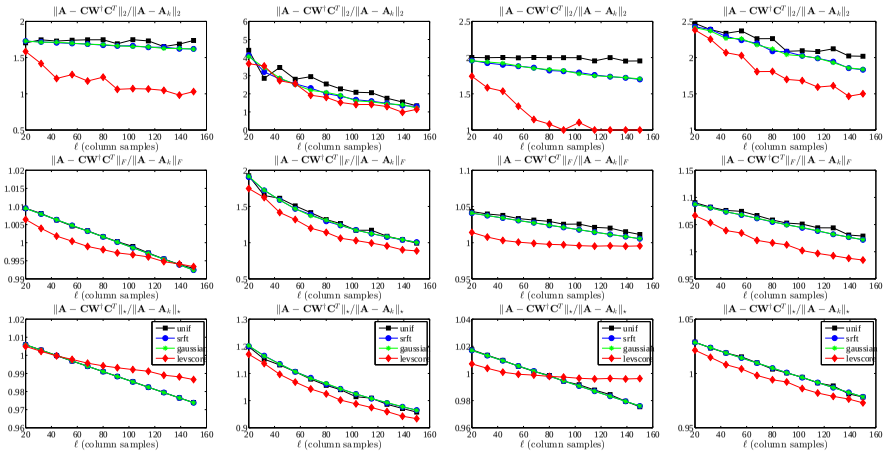
- Linear Kernels and to a lesser extent Dense RBF Kernels with larger σ parameter have relatively low rank and relatively uniform leverage scores, and in these cases uniform sampling does quite well. These data sets correspond most closely with those that have been studied previously in the machine learning literature, and for these data sets our results are in agreement with that prior work.

- Sparsifying RBF Kernels and/or choosing a smaller σ parameter tends to make these kernels less well-approximated by low-rank matrices and to have more heterogeneous leverage scores. In general, these two properties need not be directly related—the spectrum is a property of eigenvalues, while the leverage scores are determined by the eigenvectors—but for the data we examined they are related, in that matrices with more slowly decaying spectra also often have more heterogeneous leverage scores.

- For Dense RBF Kernels with smaller σ and Sparse RBF Kernels, leverage score sampling tends to do much better than other methods. Interestingly, the Sparse RBF Kernels have many properties of very sparse Laplacian Kernels corresponding to relatively-unstructured informatics graphs, an observation which should be of interest for researchers who construct sparse graphs from data using, *e.g.*, "locally linear" methods, to try to reconstruct hypothesized low-dimensional manifolds.

- Reconstruction quality under leverage score sampling saturates, as a function of choosing more samples ℓ. As a consequence, there can be a tradeoff between leverage score sampling or other methods being better, depending on the values of ℓ that are chosen.

In general, *all* of the sampling and projection methods we considered perform *much* better on the SPSD matrices we considered than previous worst-case bounds (*e.g.*, (Drineas and Mahoney, 2005; Kumar et al., 2012; Gittens, 2012)) would suggest. Specifically, even the worst results correspond to single-digit approximation factors in relative scale. This observation is intriguing, because the motivation of leverage score sampling (recall that in this context random projections should be viewed as performing uniform random sampling in a



(a) AbaloneS, $\sigma = .15, k = 20$ (b) AbaloneS, $\sigma = 1, k = 20$ (c) WineS, $\sigma = 1, k = 20$ (d) WineS, $\sigma = 2.1, k = 20$

Figure 5: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several SPSD sketches, as a function of the number of column samples ℓ, for several sparse RBF data sets.

randomly-rotated basis where the leverage scores have been approximately uniformized (Mahoney, 2011)) is very much tied to the Frobenius norm, and so there is no *a priori* reason to expect its good performance to extend to the spectral or trace norms. Motivated by this, we revisit the question of proving improved worst-case theoretical bounds in Section 4.

Before describing these improved theoretical results, however, we address in Section 3.5 running time questions. After all, a naïve implementation of sampling with exact leverage scores is slower than other methods (and much slower than uniform sampling). As shown below, by using the recently-developed approximation algorithm of Drineas et al. (2012), not only does this approximation algorithm run in time comparable with random projections (for certain parameter settings), it also leads to approximations that soften the strong bias that the exact leverage scores provide toward the best rank-k approximation to the matrix, thereby leading to improved reconstruction results in many cases.

3.5 Reconstruction Accuracy of Leverage Score Approximation Algorithms

A naïve view might assume that computing probabilities that permit leverage-based sampling requires an $O(n^3)$ computation of the full SVD, or at least the full computation of a partial SVD, and thus that it would be much more expensive than recently-developed random projection methods. Indeed, an "exact" computation of the leverage scores with a truncated SVD takes roughly $\mathrm{O}(n^2k)$ time. Recent work, however, has shown that relative-error approximations to all the statistical leverage scores can be computed more quickly than this exact algorithm (Drineas et al., 2012). Here, we implement and evaluate a version of this algorithm. We evaluate it both in terms of running time and in terms of reconstruction quality on the diverse suite of real data matrices we considered above. This is the first work to provide an empirical evaluation of an implementation of the leverage score approximation algorithms of Drineas et al. (2012), illustrating empirically the tradeoffs between cost and efficiency in a practical setting.

3.5.1 Description of the Fast Approximation Algorithm of Drineas et al. (2012)

Algorithm 1 (which originally appeared as Algorithm 1 in Drineas et al. (2012)) takes as input an arbitrary $n \times d$ matrix $\mathbf{A}$, where $n \gg d$, and it returns as output a $1 \pm \epsilon$ approximation to *all* of the statistical leverage scores of the input matrix. The original algorithm of Drineas et al. (2012) uses a subsampled Hadamard transform and requires r_1 to be somewhat larger than what we state in Algorithm 1. That an SRFT with a smaller value of r_1 can be used instead is a consequence of the fact that (Drineas et al., 2012, Lemma 3) is also satisfied by an SRFT matrix with the given r_1; this is established in (Tropp, 2011; Boutsidis and Gittens, 2013).

The running time of this algorithm, given in the caption of the algorithm, is roughly $O(nd \ln d)$ when $d = \Omega(\ln n)$. Thus Algorithm 1 generates relative-error approximations to the leverage scores of a tall and skinny matrix $\mathbf{A}$ in time $o(nd^2)$, rather than the $\Omega(nd^2)$ time that would be required to compute a QR decomposition or a thin SVD of the $n \times d$ matrix $\mathbf{A}$. The basic idea behind Algorithm 1 is as follows. If we had a QR decomposition of $\mathbf{A}$, then we could postmultiply $\mathbf{A}$ by the inverse of the "R" matrix to obtain an orthogonal matrix spanning the column space of $\mathbf{A}$; and from this $n \times d$ orthogonal matrix, we could read off

Input: $\mathbf{A} \in \mathbb{R}^{n \times d}$ (with SVD $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$), error parameter $\epsilon \in (0, 1/2]$.

Output: $\tilde{\ell}_i, i = 1, \ldots, n$, approximations to the leverage scores of $\mathbf{A}$.

1. Let $\mathbf{\Pi}_1 \in \mathbb{R}^{r_1 \times n}$ be an SRFT with
$$r_1 = \Omega(\epsilon^{-2}(\sqrt{d} + \sqrt{\ln n})^2 \ln d)$$
2. Compute $\mathbf{\Pi}_1\mathbf{A} \in \mathbb{R}^{r_1 \times d}$ and its QR factorization $\mathbf{\Pi}_1\mathbf{A} = \mathbf{QR}$.
3. Let $\mathbf{\Pi}_2 \in \mathbb{R}^{d \times r_2}$ be a matrix of i.i.d. standard Gaussian random variables, where
$$r_2 = \Omega\left(\epsilon^{-2} \ln n\right).$$
4. Construct the product $\mathbf{\Omega} = \mathbf{A}\mathbf{R}^{-1}\mathbf{\Pi}_2$.
5. For $i = 1, \ldots, n$ compute $\tilde{\ell}_i = \left\|\Omega_{(i)}\right\|_2^2$.

Algorithm 1: Algorithm (Drineas et al., 2012, Algorithm 1) for approximating the leverage scores ℓ_i of an $n \times d$ matrix $\mathbf{A}$, where $n \gg d$, to within a multiplicative factor of $1 \pm \epsilon$. The running time of the algorithm is $\mathrm{O}(nd \ln(\sqrt{d} + \sqrt{\ln n}) + nd\epsilon^{-2} \ln n + d^2\epsilon^{-2}(\sqrt{d} + \sqrt{\ln n})^2 \ln d)$.

the leverage scores from the Euclidean norms of the rows. Of course, computing the QR decomposition would require $O(nd^2)$ time. To get around this, Algorithm 1 premultiplies $\mathbf{A}$ by a structured random projection $\mathbf{\Pi}_1$, computes a QR decomposition of $\mathbf{\Pi}_1\mathbf{A}$, and postmultiplies $\mathbf{A}$ by $\mathbf{R}^{-1}$, *i.e.*, the inverse of the "R" matrix from the QR decomposition of $\mathbf{\Pi}_1\mathbf{A}$. Since $\mathbf{\Pi}_1$ is an SRFT, premultiplying by it takes roughly $O(nd \ln d)$ time. In addition, note that $\mathbf{\Pi}_1\mathbf{A}$ needs to be post multiplied by a second random projection in order to compute all of the leverage scores in the allotted time; see (Drineas et al., 2012) for details. This algorithm is simpler than the algorithm in which we are primarily interested that is applicable to square SPSD matrices, but we start with it since it illustrates the basic ideas of how our main algorithm works and since our main algorithm calls it as a subroutine. We note, however, that this algorithm is directly useful for approximating the leverage scores of Linear Kernel matrices $\mathbf{A} = \mathbf{X}\mathbf{X}^T$, when $\mathbf{X}$ is a tall and skinny matrix.

Consider, next, Algorithm 2 (which originally appeared as Algorithm 4 in (Drineas et al., 2012)), which takes as input an *arbitrary* $n \times d$ matrix $\mathbf{A}$ and a rank parameter k, and returns as output a $1 \pm \epsilon$ approximation to *all* of the statistical leverage scores (relative to the best rank-k approximation) of the input. An important technical point is that the problem of computing the leverage scores of a matrix relative to a low-dimensional space is ill-posed, essentially because the spectral gap between the k^{th} and the $(k+1)^{st}$ eigenvalues can be small, and thus Algorithm 2 actually computes approximations to the leverage scores of a matrix that is near to $\mathbf{A}$ in the spectral norm (or the Frobenius norm if $q = 0$). See (Drineas et al., 2012) for details. Basically, this algorithm uses Gaussian sampling to find

Input: $\mathbf{A} \in \mathbb{R}^{n\times d}$, a rank parameter k, and an error parameter $\epsilon \in (0, 1/2]$.

Output: $\hat{\ell}_i, i = 1, \ldots, n$, approximations to the leverage scores of $\mathbf{A}$ filtered through its dominant dimension-k subspace.

1. Construct $\mathbf{\Pi} \in \mathbb{R}^{d\times 2k}$ with i.i.d. standard Gaussian entries.
2. Compute $\mathbf{B} = \left(\mathbf{A}\mathbf{A}^T\right)^q \mathbf{A}\mathbf{\Pi} \in \mathbb{R}^{n\times 2k}$ with
$$q \geq \left\lceil \frac{\ln\left(1 + \sqrt{\frac{k}{k-1}} + \mathrm{e}\sqrt{\frac{2}{k}}\sqrt{\min\{n,d\} - k}\right)}{2\ln\left(1+\epsilon/10\right) - 1/2} \right\rceil .$$
3. Approximate the leverage scores of $\mathbf{B}$ by calling Algorithm 1 with inputs $\mathbf{B}$ and ϵ; let $\hat{\ell}_i$ for $i = 1, \ldots, n$ be the outputs of Algorithm 1.

Algorithm 2: Algorithm (Drineas et al., 2012, Algorithm 4) for approximating the leverage scores (relative to the best rank-k approximation to $\mathbf{A}$) of a general $n \times d$ matrix $\mathbf{A}$ with those of a matrix that is close by in the spectral norm (or the Frobenius norm if $q = 0$). This algorithm runs in time $\mathrm{O}(ndkq) + T_1$, where T_1 is the running time of Algorithm 1.

a matrix close to $\mathbf{A}$ in the Frobenius norm or spectral norm, and then it approximates the leverage scores of this matrix by using Algorithm 1 on the smaller, very rectangular matrix $\mathbf{B}$. When $\mathbf{A}$ is square, as in our applications, Algorithm 2 is typically more costly than direct computation of the leverage scores, at least for dense matrices (but it does have the advantage that the number of iterations is bounded, independent of properties of the matrix, which is not true for typical iterative methods to compute low-rank approximations).

Of greater practical interest is Algorithm 3, which is a modification of Algorithm 2 in which the Gaussian random projection is replaced with an SRFT. That is, Algorithm 3 uses an SRFT projection to find a matrix close by to $\mathbf{A}$ in the Frobenius norm or spectral norm (depending on the value of q), and then it exactly computes the leverage scores of this matrix. This improves the running time to $\mathrm{O}(n^2\ln(\sqrt{k}+\sqrt{\ln n}) + n^2(\sqrt{k}+\sqrt{\ln n})^2\ln(k)q + n(\sqrt{k}+\sqrt{\ln n})^4\ln^2(k))$, which is $o(n^2k)$ when $q = 0$. Thus an important point for Algorithm 3 (as well as for Algorithm 2) is the parameter q which describes the number of iterations. For $q = 0$ iterations, we get an inexpensive Frobenius norm approximation; while for higher q, we get better spectral norm approximations that are more expensive.[3] This flexibility is of interest, as one may want to approximate the actual leverage scores accurately or one may simply want to find crude approximations useful for obtaining SPSD sketches with low reconstruction error.

3. Observe that since $\mathbf{A}$ is rectangular in Algorithms 2 and 3, we approximate the leverage scores of $\mathbf{A}$ with those of $\mathbf{B} = (\mathbf{A}\mathbf{A}^T)^q\mathbf{A}\mathbf{\Pi}$; in particular the case $q = 0$ corresponds to taking $\mathbf{B} = \mathbf{A}\mathbf{\Pi}$. By way of contrast, when we use the power method to construct sketches of an SPSD matrix, we take $\mathbf{C} = \mathbf{A}^q\mathbf{S}$, so the case $q = 1$ corresponds to $\mathbf{C} = \mathbf{A}\mathbf{S}$.

Input: $\mathbf{A} \in \mathbb{R}^{n\times d}$, a rank parameter k, and an iteration parameter q.

Output: $\hat{\ell}_i, i \in= 1, \ldots, n$, approximations to the leverage scores of $\mathbf{A}$ filtered through its dominant dimension-k subspace.

1. Construct an SRHT matrix $\mathbf{\Pi} \in \mathbb{R}^{d\times r}$, where
$$r \geq \left\lceil 36\epsilon^{-2}[\sqrt{k} + \sqrt{8\ln(kd)}]^2\ln(k) \right\rceil .$$
2. Compute $\mathbf{B} = \left(\mathbf{A}\mathbf{A}^T\right)^q \mathbf{A}\mathbf{\Pi} \in \mathbb{R}^{n\times r}$, where $q \geq 0$ is an integer.
3. Return the exact leverage scores of $\mathbf{B}$.

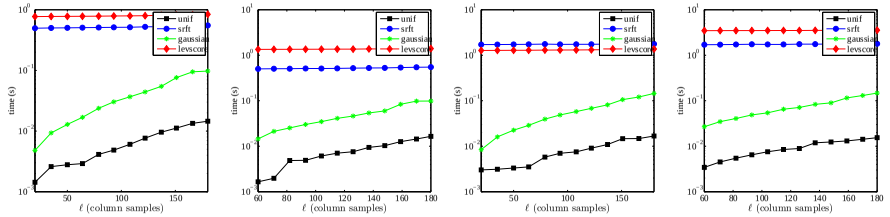
Algorithm 3: Algorithm for approximating the leverage scores (relative to the best rank-k approximation to $\mathbf{A}$) of a general $n \times d$ matrix $\mathbf{A}$ with those of a matrix that is close by in the spectral norm. This is a modified version of Algorithm 2, in which the random projection is implemented with an SRFT rather than a Gaussian random matrix, and where the number of "iterations" q is prespecified. This algorithm runs in time $\mathrm{O}(nd\ln r + ndrq + nr^2)$ since $\mathbf{A}\mathbf{\Pi}$ can be computed in time $\mathrm{O}(nd\ln r)$.

Finally, note that although choosing the number of iterations q as we did in Algorithm 2 is convenient for worst-case analysis, as a practical implementational matter it is easier either to choose q based on spectral gap information revealed during the running of the algorithm or to prespecify q to be a small integer, *e.g.*, 2 or 3, before the algorithm runs. Both of these have an interpretation of accelerating the rate of decay of the spectrum with a power iteration, but they behave somewhat differently due to the different stopping conditions. Below, we consider both variants.

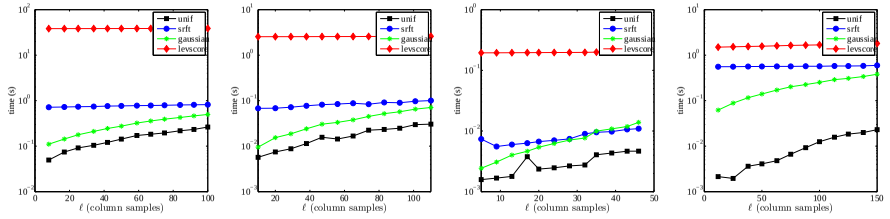
3.5.2 Running Time Comparisons

Here, we describe the performances of the various random sampling and random projection low-rank sketches considered in Section 3.4 in terms of their running time, where the method that involves using the leverage scores to construct the importance sampling distribution is implemented both by computing the leverage scores "exactly" by calling a truncated SVD, as a black box, as well as computing them approximately by using one of several versions of Algorithm 3. Our running time results are presented in Figure 6 and Figure 7.

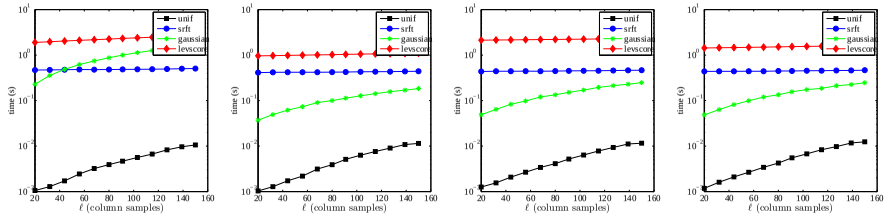
We start with the results described in Figure 6, which shows the running times, as a function of ℓ, for the low-rank approximations described in Section 3.4: *i.e.*, for column sampling uniformly at random without replacement; for column sampling according to the exact nonuniform leverage score probabilities; and for sketching using Gaussian and SRFT mixtures of the columns. Several observations are worth making about the results presented in this figure.



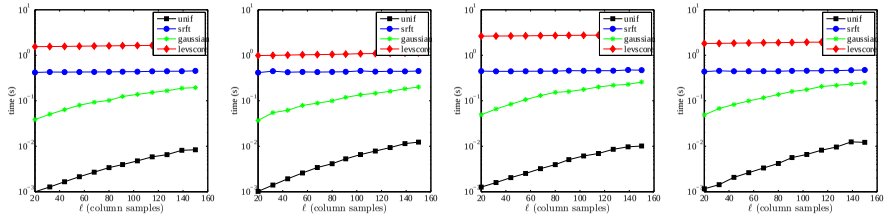
(a) GR, $k = 20$ (b) GR, $k = 60$ (c) HEP, $k = 20$ (d) HEP, $k = 60$



(e) Dexter, $k = 8$ (f) Protein, $k = 10$ (g) SNPs, $k = 5$ (h) Gisette, $k = 12$

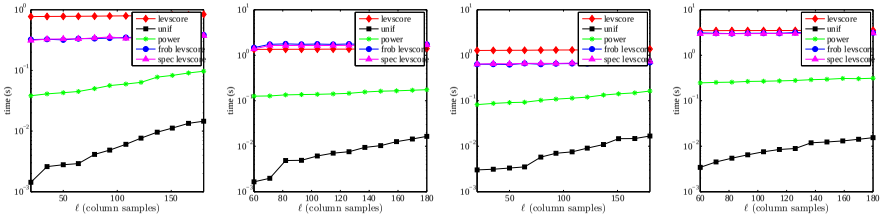


(i) AbaloneD, $\sigma = .15, k = 20$ (j) AbaloneD, $\sigma = 1, k = 20$ (k) WineD, $\sigma = 1, k = 20$ (l) WineD, $\sigma = 2.1, k = 20$

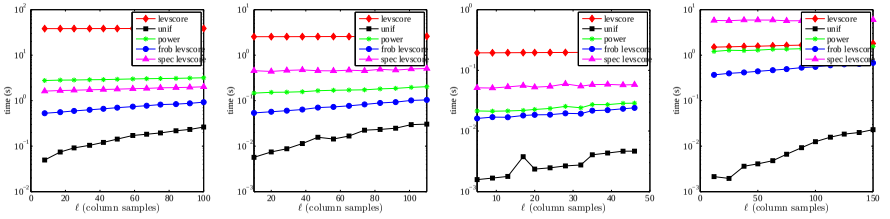


(m) AbaloneS, $\sigma = .15, k = 20$ (n) AbaloneS, $\sigma = 1, k = 20$ (o) WineS, $\sigma = 1, k = 20$ (p) WineS, $\sigma = 2.1, k = 20$

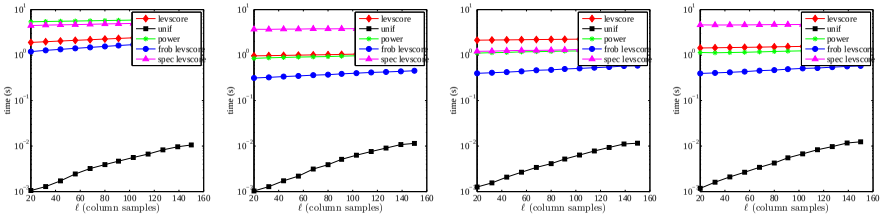
Figure 6: The times required to compute SPSD sketches, as a function of the number of column samples ℓ for several data sets and two choices of the rank parameter k.



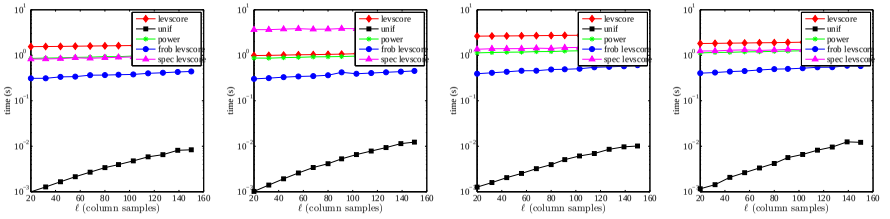
(a) GR, $k = 20$ (b) GR, $k = 60$ (c) HEP, $k = 20$ (d) HEP, $k = 60$



(e) Dexter, $k = 8$ (f) Protein, $k = 10$ (g) SNPs, $k = 5$ (h) Gisette, $k = 12$



(i) AbaloneD, $\sigma = .15, k = 20$ (j) AbaloneD, $\sigma = 1, k = 20$ (k) WineD, $\sigma = 1, k = 20$ (l) WineD, $\sigma = 2.1, k = 20$



(m) AbaloneS, $\sigma = .15, k = 20$ (n) AbaloneS, $\sigma = 1, k = 20$ (o) WineS, $\sigma = 1, k = 20$ (p) WineS, $\sigma = 2.1, k = 20$

Figure 7: The times required to compute approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ for several data sets.

- Uniform sampling is always less expensive and typically much less expensive than the other methods, while (with one minor exception) sampling according to the *exact* leverage scores is always the most expensive method.

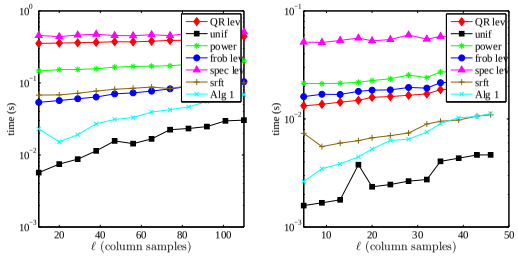
- For most matrices, using the SRFT is nearly as expensive as exact leverage score sampling. This is most true for the very sparse graph Laplacian Kernels, largely since the SRFT does not respect sparsity. The main exception to this is for the dense and relatively well-behaved Linear Kernels, where especially for large values of ℓ the SRFT is quite fast and usually not too much more expensive than uniform sampling.

- The "fast Fourier" methods underlying the SRFT can take advantage of the structure of the Linear Kernels to yield algorithms that are similar to Gaussian projections and much better than exact leverage score computation. Note that the reason that SRFT is worse than Gaussians here is that the matrices we are considering are *not* extremely large, and we are not considering very large values of the rank parameter. Extending in both those directions leads to Gaussian projections being slower than SRFT, as the trends in the figures clearly indicate.

- Gaussian projections are not too much slower than uniform sampling for the extremely sparse Laplacian Kernels—this is due to the sparsity of the Laplacian Kernels, since Gaussian projections can take advantage of fast matrix-vector multiplies, while the SRFT-based scheme cannot—but this advantage is lost for the (denser) Sparse RBF Kernels, to the extent that there is little running time improvement relative to the Dense RBF Kernels. In addition, Gaussian projections are relatively slower, when compared to the SRFT and uniform sampling, for the Dense RBF Kernels than for the Linear Kernels, although both of those data sets are maximally dense.

We next turn to the results described in Figure 7, which shows the running times, as a function of ℓ, for several variants of approximate leverage-based sampling. For ease of comparison, the timings for uniform sampling ("unif") and exact leverage score sampling ("levscore") are depicted in Figure 7 using the same shading as used in Figure 6. In addition to these two baselines, Figure 7 shows running time results for the following three variants of approximate leverage score sampling: "frob levscore" (which is Algorithm 3 with $q = 0$ and $r = 2k$); "spec levscore" (Algorithm 3 with $q = 4$ and $r = 2k$); and "power". The "power" scheme is a version of Algorithm 3 where $r = k$ and q is determined by monitoring the convergence of the leverage scores of $\mathbf{A}^{2q+1}\mathbf{\Pi}$ and terminating when the change in the leverage scores between iterations, as measured in the infinity norm, is smaller than 10^{-2}. This is simply a version of subspace iteration with a convergence criterion appropriate for the task at hand. Since "frob levscore" requires one application of an SRFT, its timing results are depicted using the same shade as the SRFT timing results in Figure 6. (There are no other correspondences between the shadings in the two figures.) Several observations are worth making about the results presented in this figure.

- These approximate leverage score-based algorithms can be orders of magnitude faster than exact leverage score computation; but, especially for "spec levscore" when q is not prespecified to be 2 or 3, they can even be somewhat slower. Exactly which is



(a) Protein, $k = 10$ (b) SNPs, $k = 5$

Figure 8: The running time of SPSD sketches computed using Algorithm 1 compared with that of other approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ for two Linear Kernel datasets. The parameters in Algorithm 1 were taken to be $r_1 = \epsilon^{-2}\ln(d\delta^{-1})(\sqrt{d} + \sqrt{\ln(n\delta^{-1})})^2$ and $r_2 = \epsilon^{-2}(\ln n + \ln \delta^{-1})$ with $\epsilon = 1$ and $\delta = 1/10$.

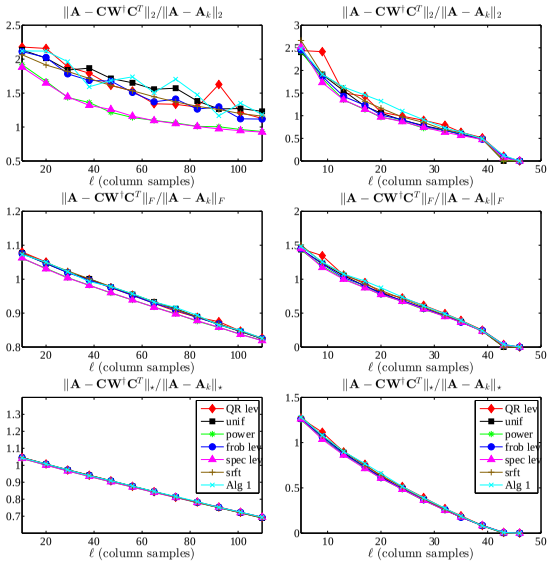
the case depends upon the properties of the matrix and the parameters used in the approximation algorithm, including especially the number of power iterations.

- The "frob levscore" approximation method has running time comparable to the running time of the SRFT, which is expected, given that the computation of the SRFT is the theoretical bottleneck for the running time of the "frob levscore" algorithm. In particular, for larger values of ℓ for Linear Kernels, "frob levscore" is not much slower than uniform sampling.

- The "spec levscore" and "power" approximations with $q > 0$ are more expensive than the $q = 0$ "frob lev" approximation, which is a result of the relatively-expensive matrix-matrix multiplication. For the Linear Kernels, both are much better than the exact leverage score computation, and for most other data at least "power" is somewhat less expensive than the exact leverage score computation. For example, this is particularly true for the Laplacian Kernels.

Recall that the cost associated with these SPSD sketches is two-fold: first, the cost to construct the sample—by sampling columns uniformly at random, by computing a nonuniform importance sampling distribution, or by performing a random projection to uniformize the leverage scores; and second, the cost to construct the low-rank approximation from the sample. For uniform sampling, the latter step dominates the cost, while for more sophisticated methods the former step typically dominates the cost. The approximate leverage score sampling methods are still sufficiently expensive that the cost of computing the sampling probabilities still dominates the cost to construct the low-rank approximation.

Finally, Algorithm 1 can be used to approximate quickly the leverage scores of matrices of the form $\mathbf{A} = \mathbf{X}\mathbf{X}^T$, when $\mathbf{X} \in \mathbb{R}^{n\times d}$ is a rectangular matrix of sufficent aspect ratio, and



(a) Protein, $k = 10$ (b) SNPs, $k = 5$

Figure 9: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of SPSD sketches computed using Algorithm 1 compared with those of other approximate leverage score-based sketching schemes, as a function of the number of column samples ℓ, for two Linear Kernel data sets. The parameters in Algorithm 1 were taken to be $r_1 = \epsilon^{-2}\ln(d\delta^{-1})(\sqrt{d}+\sqrt{\ln(n\delta^{-1})})^2$ and $r_2 = \epsilon^{-2}(\ln n + \ln\delta^{-1})$ with $\epsilon = 1$ and $\delta = 1/10$.

in such cases it is faster than Algorithm 3. Specifically, for the first dimensional reduction step in Algorithm 1 to be beneficial (*i.e.*, to ensure $r_1 < n$), the condition $n = \Omega(d\ln d)$ is necessary; for the second dimensional reduction step to be beneficial (*i.e.*, to ensure $r_2 < d$), the condition $d = \Omega(\ln n)$ must be satisfied. Figure 8 summarizes our main results for the run time of Algorithm 1 applied to rectangular matrices with $n \gg d$. Among other things, Figure 8 illustrates, using the Linear Kernel datasets Protein and SNPs (which satisfy these constraints), two points.

- Most importantly, the running time of Algorithm 1 on these rectangular matrices is faster than performing a QR decomposition on $\mathbf{A}$ and is comparable to applying a SRFT to $\mathbf{A}$. This is expected, since the running time bottleneck for Algorithm 1 is the application of the SRFT.
- In addition, the running time of Algorithm 1 is significantly faster than the other approximate leverage score algorithms. This too is expected, since these other algorithms are applied to $\mathbf{A}$ and ignore the rectangular structure of $\mathbf{X}$.

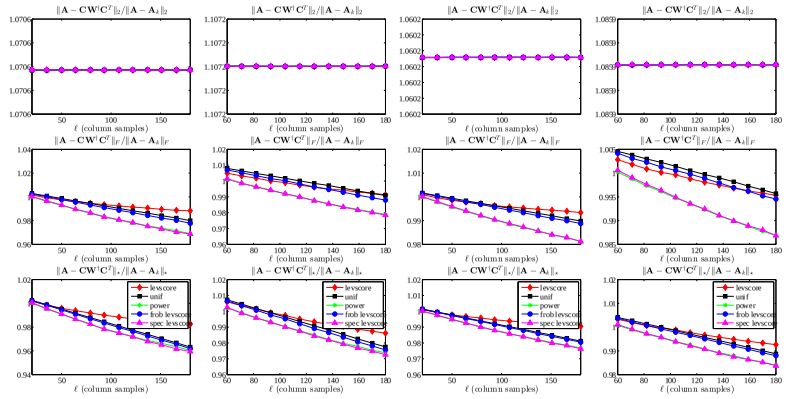
Figure 9 shows that these improved running time gains for Algorithm 1 can come at the cost of a slight loss in the reconstruction accuracy (relative to the exact computation of the leverage scores) of the low-rank approximations; the accuracy of the other approximate leverage score algorithms is discussed in the following subsection.

3.5.3 Reconstruction Accuracy Results

Here, we describe the performances of the various low-rank approximations that use approximate leverage scores in terms of reconstruction accuracy for the data sets described in Section 3.1. The results are presented in Figure 10 through Figure 14. The setup for these results parallels that for the low-rank approximation results described in Section 3.4, and these figures parallel Figure 1 through Figure 5. To provide a baseline for the comparison, we also plot the previous reconstruction errors for sampling with the exact leverage scores as well as the uniform column sampling sketch. Several observations are worth making about the results presented in these figures.

For Laplacian Kernels, "frob levscore" is only slightly better than uniform sampling, while "power" and "spec levscore" are substantially better than uniform sampling; all of those methods also lead to even *better* reconstruction results than using the exact leverage scores (suggesting that some form of implicit regularization is taking place): the reconstruction quality is higher for a given ℓ and, also, using approximate leverage scores does not lead to the saturation effect observed when using the exact leverage scores. For the Linear Kernels, all the methods perform similarly. For both the dense and the sparse RBF data sets, the approximate leverage score algorithms tend to parallel the exact leverage score algorithm, and they are not substantially better. In particular, both "power" and "spec levscore" tend to saturate when the exact method saturates, but in those cases "frob levscore" tends not to saturate.

Note that the difference between different approximate leverage score algorithms often corresponds to a difference in the spectral gaps of the corresponding matrices. From Table 5, if we fix k and use the approximate leverage scores filtered through rank k to form a Nyström approximation to $\mathbf{A}$, the accuracy of that approximation has a strong dependence on the



(a) GR, $k = 20$ (b) GR, $k = 60$ (c) HEP, $k = 20$ (d) HEP, $k = 60$

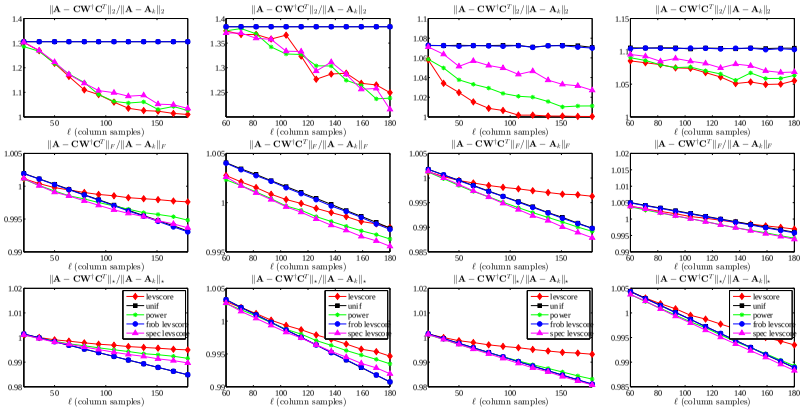
Figure 10: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ, for the GR and HEP Laplacian data sets, with two choices of the rank parameter k.

spectral gap of $\mathbf{A}$ at rank k, as measured by $\frac{\lambda_k}{\lambda_{k+1}}$. In general, the larger the spectral gap, the more accurate the approximation. This phenomena can also be understood in terms of the convergence of the approximate leverage scores: the approximation algorithms (Algorithm 2 and Algorithm 3) are essentially truncated versions of the subspace iteration method for computing the top k eigenvectors of $\mathbf{A}$. It is a classical result that the spectral gap determines the rate of convergence of the subspace iteration process to the desired eigenvectors: the larger it is, the fewer iterations of the process are required to get accurate approximations of the top eigenvectors. It follows immediately that the larger the spectral gap, the more accurate the approximate leverage scores generated by these approximation algorithms are. Our empirical results illustrate the complexities and subtle consequences of these properties in realistic machine learning applications of even modestly-large size.

3.5.4 Summary of Leverage Score Approximation Algorithms

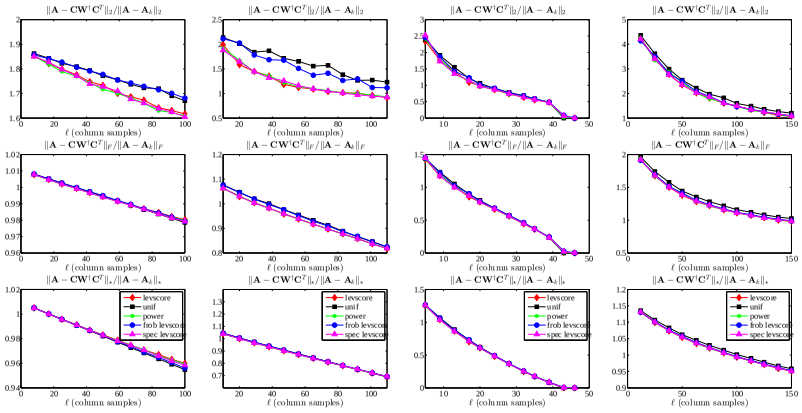
Before proceeding, there are several summary observations that we can make about the running time and reconstruction quality of approximate leverage score sampling algorithms for the data sets we have considered.

- The running time of computing the exact leverage scores is generally much worse than that of uniform sampling and both SRFT-based and Gaussian-based random projection methods.



(a) Enron, $k = 20$ (b) Enron, $k = 60$ (c) Gnutella, $k = 20$ (d) Gnutella, $k = 60$

Figure 11: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ, for the Enron and Gnutella Laplacian data sets, with two choices of the rank parameter k.



(a) Dexter, $k = 8$ (b) Protein, $k = 10$ (c) SNPs, $k = 5$ (d) Gisette, $k = 12$

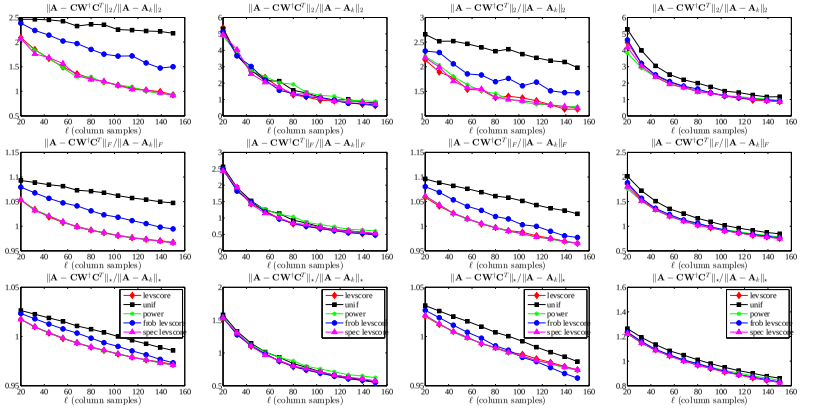
Figure 12: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ, for the Linear Kernel data sets.

- The running time of computing approximations to the leverage scores can, with appropriate choice of parameters, be much faster than the exact computation of the leverage scores; and, especially for "frob levscore," can be comparable to the running time of the random projection (SRFT or Gaussian) used in the leverage score approximation algorithm. For the methods that involve $q > 0$ iterations to compute stronger approximations to the leverage scores, the running time can vary considerably depending on details of the stopping condition.

- The leverage scores computed by the "frob levscore" procedure are typically very different than the "exact" leverage scores, but they are leverage scores for a low-rank space that is near the best rank-k approximation to the matrix. This is often sufficient for good low-rank approximation.

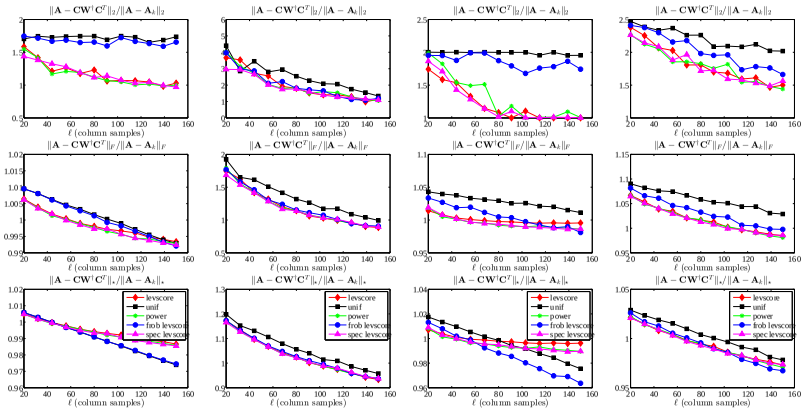
- The approximate leverage scores computed from "power" and "spec levscore" approach those of the exact leverage scores, as q is increased; and they obtain reconstruction accuracy that is no worse, and in many cases is better, than that obtained by the exact leverage scores. This suggests that, by not fitting exactly to the empirical statistical leverage scores, we are observing a form of implicit regularization.

- The running time of Algorithm 1, when applied to "tall" matrices for which $n \gg d$, is faster than the running time of performing a QR decomposition of the matrix $\mathbf{A}$; and it is comparable to the running time of applying a random projection to $\mathbf{A}$ (which is the computational bottleneck of applying Algorithm 1). Thus, in particular, one could



(a) AbaloneD, $\sigma = .15, k = 20$ (b) AbaloneD, $\sigma = 1, k = 20$ (c) WineD, $\sigma = 1, k = 20$ (d) WineD, $\sigma = 2.1, k = 20$

Figure 13: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ, for several dense RBF data sets.



(a) AbaloneS, $\sigma = .15, k = 20$ (b) AbaloneS, $\sigma = 1, k = 20$ (c) WineS, $\sigma = 1, k = 20$ (d) WineS, $\sigma = 2.1, k = 20$

Figure 14: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several approximate leverage score-based SPSD sketches, as a function of the number of column samples ℓ, for several sparse RBF data sets.

use this algorithm to compute approximations to the leverage scores to obtain a sketch that provides a relative-error approximation to a least-squares problem involving $\mathbf{A}$ (Drineas et al., 2008, 2010; Mahoney, 2011); or one could use the sketch thereby obtained as a preconditioner to an iterative method to solve the least-squares problem, in a manner analogous to how Blendenpik or LSRN do so with a random projection (Avron et al., 2010; Meng et al., 2014).

Previous work has showed that one can implement random projection algorithms to provide low-rank approximations with error comparable to that of the SVD in less time than state-of-the art Krylov solvers and other "exact" numerical methods (Halko et al., 2011; Mahoney, 2011). Our empirical results show that these random projection algorithms can be used in two complementary ways to approximate SPSD matrices of interest in machine learning: first, they can be used directly to compute a projection-based low-rank approximation; and second, they can be used to compute approximations to the leverage scores, which can be used to compute a sampling-based low-rank approximation. With the right choice of parameters, the two complementary approaches have roughly comparable running times, and neither one dominates the other in terms of reconstruction accuracy.

3.6 Projection-based Sketches

Finally, for completeness, we consider the performance of the two projection-based SPSD sketches proposed by Halko et al. (2011), and we show how they perform when compared with the sketches we have considered. Recall that the idea of these sketches is to construct low-rank approximations by forming an approximate basis $\mathbf{Q}$ for the top eigenspace of $\mathbf{A}$ and then restricting $\mathbf{A}$ to that eigenspace. In more detail, given a sketching matrix $\mathbf{S}$, form the matrix $\mathbf{Y} = \mathbf{AS}$ and take the QR decomposition of $\mathbf{Y}$ to obtain $\mathbf{Q}$, a matrix with orthonormal columns. The first sketch, which we eponymously refer to as the *pinched* sketch, is simply $\mathbf{A}$ pinched to the space spanned by $\mathbf{Q}$:

$$\mathbf{Q}(\mathbf{Q}^T\mathbf{AQ})\mathbf{Q}^T. \tag{8}$$

The second sketch, which we refer to as the *prolonged* sketch, is

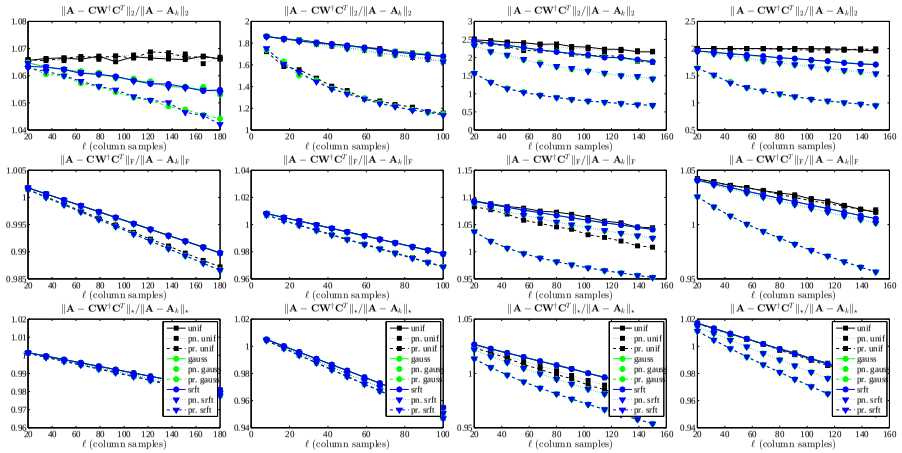
$$\mathbf{AQ}(\mathbf{Q}^T\mathbf{AQ})^\dagger\mathbf{Q}^T\mathbf{A}. \tag{9}$$

It is clear that the prolonged sketch can be constructed using our SPSD Sketching Model by taking $\mathbf{Q}$ as the sketching matrix. In fact, a stronger statement can be made. As stated in Lemma 1 below, it is the case, for any sketching matrix $\mathbf{X}$, that when $\mathbf{C} = \mathbf{AX}$ and $\mathbf{W} = \mathbf{X}^T\mathbf{AX}$,

$$\mathbf{CW}^\dagger\mathbf{C}^T = \mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{1/2}\mathbf{X}}\mathbf{A}^{1/2}.$$

By considering the sketching matrix $\mathbf{X} = \mathbf{A}^1\mathbf{S}$, we see that in fact the prolonged sketch is exactly the sketch obtained by applying the power method with $q = 1$:

$$\begin{aligned}\mathbf{AQ}(\mathbf{Q}^T\mathbf{AQ})^\dagger\mathbf{Q}^T\mathbf{A} &= \mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{1/2}\mathbf{Q}}\mathbf{A}^{1/2} \\ &= \mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{1/2}(\mathbf{AS})}\mathbf{A}^{1/2} \\ &= \mathbf{A}^2\mathbf{S}(\mathbf{S}^T\mathbf{A}^3\mathbf{S})^\dagger\mathbf{S}^T\mathbf{A}^2 \\ &= \mathbf{AX}(\mathbf{X}^T\mathbf{AX})^\dagger\mathbf{X}^T\mathbf{A}.\end{aligned}$$



(a) Gnutella, $k = 20$ (b) Dexter, $k = 8$ (c) AbaloneD, $\sigma = .15, k = 20$ (d) WineS, $\sigma = 1, k = 20$

Figure 15: The spectral, Frobenius, and trace norm errors (top to bottom, respectively, in each subfigure) of several SPSD sketches, including the pinched and prolonged sketches, as a function of the number of column samples ℓ, for several datasets. Pinched and prolonged sketches, respectively indicated by "pn." and "pr.", are defined in Equations (8) and (9).

It follows that the bounds we provide in Section 4 on the performance of sketches obtained using the power method pertain also to prolonged sketches.

In Figure 15, we compare the empirical performances of several of the SPSD sketches considered earlier with their pinched and prolonged variants. Specifically, we plot the errors of pinched and prolonged sketches for several choices of sketching matrices—corresponding to uniform column sampling, gaussian column mixtures, and SRFT-based column mixtures—along with the errors of non-pinched, non-prolonged sketches constructed using the same choices of $\mathbf{S}$. In the interest of brevity, we provide results only for several of the datasets listed in Table 4.

Some trends are clear from Figure 15.

- In the spectral norm, the prolonged sketches are considerably more accurate than the pinched and standard sketches for all the datasets considered. Without exception, the prolonged Gaussian and SRFT column-mixture sketches are the most accurate in the spectral norm, of all the sketches considered. Only in the case of the Dexter Linear Kernel is the prolonged uniformly column-sampled sketch nearly as accurate in the spectral norm as the prolonged Gaussian and SRFT sketches. To a lesser extent, the prolonged sketches are also more accurate in the Frobenius and trace norms than the other sketches considered. The increased Frobenius and trace norm accuracy is particularly notable for the two RBF Kernel datasets; again, the prolonged Gaussian and SRFT sketches are considerably more accurate than the prolonged uniformly column-sampled sketches.

- After the prolonged sketches, the pinched Gaussian and SRFT column-mixture sketches exhibit the least spectral, Frobenius, and trace norm errors. Again, however, we see that the pinched uniformly column-sampled sketches are considerably less accurate than the pinched Gaussian and SRFT column-mixture sketches. Particularly in the spectral and Frobenius norms, the pinched uniformly column-sampled sketches are not any more accurate than the basic uniformly column-sampled sketches.

From these considerations, it seems evident that the benefits of pinched and prolonged sketches are most prominent when the spectral norm is the error metric, or when the dataset is an RBF Kernel. In particular, pinched and prolonged sketches are not significantly more accurate (than the sketches considered in the previous subsections) in the Frobenius and trace norms for any of the datasets considered.

It is also evident from Figure 15 that the pinched sketches often have a much slighter increase in accuracy over the basic sketches than do the prolonged sketches. To understand why the pinched sketches are less accurate than the prolonged sketches, observe that the pinched sketches satisfy

$$\begin{aligned}\mathbf{Q}(\mathbf{Q}^T\mathbf{A}\mathbf{Q})\mathbf{Q}^T &= \mathbf{P}_{\mathbf{AS}}\mathbf{A}\mathbf{P}_{\mathbf{AS}} \\ &= (\mathbf{P}_{\mathbf{AS}}\mathbf{A}^{1/2})(\mathbf{A}^{1/2}\mathbf{P}_{\mathbf{AS}}),\end{aligned}$$

while, as noted above, the prolonged sketches can be written in the form

$$\mathbf{A}\mathbf{Q}(\mathbf{Q}^T\mathbf{A}\mathbf{Q})^{\dagger}\mathbf{Q}^T\mathbf{A} = (\mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{3/2}\mathbf{S}})(\mathbf{P}_{\mathbf{A}^{3/2}\mathbf{S}}\mathbf{A}^{1/2}).$$

Thus, pinched and prolonged sketches approximate the square root of $\mathbf{A}$ by projecting, respectively, onto the ranges of $\mathbf{AS}$ and $\mathbf{A}^{3/2}\mathbf{S}$. The spectral decay present in $\mathbf{A}$ is increased when $\mathbf{A}$ is raised to a power larger than one; consequently, the range of $\mathbf{A}^{3/2}\mathbf{S}$ is more biased towards the top k-dimensional invariant subspace of $\mathbf{A}$ than is the range of $\mathbf{AS}$. It follows that the approximate square root used to construct the prolonged sketches more accurately captures the top k-dimensional subspace of $\mathbf{A}$ than does that used to construct the pinched sketches.

4. Theoretical Aspects of SPSD Low-rank Approximation

In this section, we present our main theoretical results, which consist of a suite of bounds on the quality of low-rank approximation under several different sketching methods. As mentioned above, these were motivated by our empirical observation that *all* of the sampling and projection methods we considered perform *much* better on the SPSD matrices we considered than previous worst-case bounds (*e.g.*, Drineas and Mahoney, 2005; Kumar et al., 2012; Gittens, 2012) would suggest. We start in Section 4.1 with deterministic structural conditions for the spectral, Frobenius, and trace norms. In Section 4.2, we use these results to provide our bounds for several random sampling and random projection procedures.

4.1 Deterministic Error Bounds for Low-rank SPSD Approximation

In this section, we present three theorems that provide error bounds for the spectral, Frobenius, and trace norm approximation errors under the SPSD Sketching Model of Section 2.2. These bounds hold for *any*, *e.g.*, deterministic or randomized, sketching matrix $\mathbf{S}$. Thus, *e.g.*, one could use them to check, in an *a posteriori* manner, the quality of a sketching method for which one cannot establish an *a priori* bound. Rather than doing this, we use these results (in Section 4.2 below) to derive *a priori* bounds for when the sketching operation consists of common random sampling and random projection algorithms. We note that the bounds can be interpreted geometrically in terms of the angles between the subspace spanned by the sampling matrix $\mathbf{S}$ and the dominant eigenspaces of $\mathbf{A}$; we refer the interested reader to the technical report (Gittens and Mahoney, 2013) for details.

Our results are based on the fact that approximations which satisfy our SPSD Sketching Model can be written in terms of a projection onto a subspace of the range of the square root of the matrix being approximated. The following fact appears in the proof of (Gittens, 2012, Proposition 1).

Lemma 1 *Let $\mathbf{A}$ be an SPSD matrix and $\mathbf{S}$ be a conformal sketching matrix. Then when $\mathbf{C} = \mathbf{AS}$ and $\mathbf{W} = \mathbf{S}^T\mathbf{AS}$, the corresponding low-rank SPSD approximation satisfies*

$$\mathbf{CW}^\dagger\mathbf{C}^T = \mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{1/2}\mathbf{S}}\mathbf{A}^{1/2}.$$

4.1.1 Spectral Norm Bounds

We start with a bound on the spectral norm of the residual error. Although this result is trivial to prove given prior work, it highlights several properties that we use in the analysis of our subsequent results.

Theorem 2 *Let $\mathbf{A}$ be an $n \times n$ SPSD matrix with eigenvalue decomposition partitioned as in Equation (1), $\mathbf{S}$ be a sketching matrix of size $n \times \ell$, q be a positive integer, and $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ be as defined in Equation (3). Then when $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ and $\mathbf{W} = \mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$, the corresponding low-rank SPSD approximation satisfies*

$$\left\|\mathbf{A} - \mathbf{CW}^\dagger\mathbf{C}^T\right\|_2 \leq \|\mathbf{\Sigma}_2\|_2 + \left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^{2/(2q-1)},$$

assuming $\mathbf{\Omega}_1$ has full row rank.

Proof Apply Lemma 1 with the sampling matrix $\mathbf{S}' = \mathbf{A}^{q-1}\mathbf{S}$ (where, recall, $q \geq 1$) to see that

$$\mathbf{CW}^\dagger\mathbf{C}^T = \mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}}\mathbf{A}^{1/2}.$$

It follows that

$$\left\|\mathbf{A} - \mathbf{CW}^\dagger\mathbf{C}^T\right\|_2 = \left\|\mathbf{A}^{1/2}\left(\mathbf{I} - \mathbf{P}_{\left(\mathbf{A}^{1/2}\right)^{2q-1}\mathbf{S}}\right)\mathbf{A}^{1/2}\right\|_2^2. \tag{10}$$

Next, recall that $\mathbf{\Omega}_i = \mathbf{U}_i^T\mathbf{S}$ and that $\mathbf{A}^{1/2}$ has eigenvalue decomposition $\mathbf{A}^{1/2} = \mathbf{U}\mathbf{\Sigma}^{1/2}\mathbf{U}^T$, where

$$\mathbf{U} = \begin{pmatrix}\mathbf{U}_1 & \mathbf{U}_2\end{pmatrix} \text{ and } \mathbf{\Sigma}^{1/2} = \begin{pmatrix}\mathbf{\Sigma}_1^{1/2} & \\ & \mathbf{\Sigma}_2^{1/2}\end{pmatrix}.$$

It can be shown (see Halko et al., 2011, Theorems 9.1 and 9.2) that, because $\mathbf{\Omega}_1$ has full row rank,

$$\left\|\mathbf{A}^{1/2}\left(\mathbf{I} - \mathbf{P}_{\left(\mathbf{A}^{1/2}\right)^{2q-1}\mathbf{S}}\right)\mathbf{A}^{1/2}\right\|_2^2 \leq \left(\left\|\left(\mathbf{\Sigma}_2^{1/2}\right)^{2q-1}\right\|_2^2 + \left\|\left(\mathbf{\Sigma}_2^{1/2}\right)^{2q-1}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\right)^{1/(2q-1)}. \tag{11}$$

Equations (10) and (11) imply that

$$\begin{aligned}\left\|\mathbf{A} - \mathbf{CW}^\dagger\mathbf{C}^T\right\|_2 &\leq \left(\left\|\mathbf{\Sigma}_2^{q-1/2}\right\|_2^2 + \left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\right)^{1/(2q-1)} \\ &\leq \|\mathbf{\Sigma}_2\|_2 + \left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^{2/(2q-1)}\end{aligned}$$

The latter inequality follows from the fact that the $2q-1$ radical function is subadditive when $q \geq 1$ and the identity $\left\|\mathbf{\Sigma}_2^{q-1/2}\right\|_2^2 = \|\mathbf{\Sigma}_2\|_2^{2q-1}$. This establishes the stated bound. ■

Remark. The assumption that $\mathbf{\Omega}_1$ has full row rank is very non-trivial. It is, however, satisfied by our algorithms below. See Section 4.1.4 for more details on this point.

Remark. The proof of Theorem 2 proceeds in two steps. The first step relates low-rank approximation of an SPSD matrix $\mathbf{A}$ under the SPSD Sketching Model of Section 2.2 to column sketching (*e.g.*, sampling or projecting) from the square-root of $\mathbf{A}$. A weaker relation of this type was used by Drineas and Mahoney (2005), but the stronger form that we use here in Equation (10) was first proved in (Gittens, 2012). The second step is to use a deterministic structural result that holds for sampling/projecting from an arbitrary matrix. The structural bound of the form of Equation (11) was originally proven for $q = 1$ by Boutsidis et al. (2009), who applied it to the Column Subset Selection Problem. The bound was subsequently improved by Halko et al. (2011), who applied it to a random projection algorithm and extended it to apply when $q > 1$. Although the analyses of our next two results are more complicated, they follow the same high-level two-step approach.

4.1.2 Frobenius Norm Bounds

Next, we state and prove the following bound on the Frobenius norm of the residual error. The proof parallels that for the spectral norm bound, in that we divide it into two analogous parts, but the analysis is somewhat more complex.

The multiplicative eigengap $\gamma = \lambda_{k+1}(\mathbf{A})/\lambda_k(\mathbf{A})$ that appears in the statement of this theorem predicts the effect of using the power method when constructing sketches. Specifically, the additional errors of sketches constructed using $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ are at least a factor of γ^{q-1} times smaller than those constructed using $\mathbf{C} = \mathbf{AS}$.

Theorem 3 *Let* $\mathbf{A}$ *be an* $n \times n$ *SPSD matrix with eigenvalue decomposition partitioned as in Equation* (1)*,* $\mathbf{S}$ *be a sketching matrix of size* $n \times \ell$*,* q *be a positive integer,* $\mathbf{\Omega}_1$ *and* $\mathbf{\Omega}_2$ *be as defined in Equation* (3)*, and define*

$$\gamma = \frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}.$$

Then when $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ *and* $\mathbf{W} = \mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$*, the corresponding low-rank SPSD approximation satisfies*

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F} \leq \|\mathbf{\Sigma}_2\|_\mathrm{F} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2 \cdot \left(\sqrt{2\,\mathrm{Tr}\,(\mathbf{\Sigma}_2)} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}\right),$$

assuming $\mathbf{\Omega}_1$ *has full row rank.*

Proof Apply Lemma 1 with the sampling matrix $\mathbf{S}' = \mathbf{A}^{q-1}\mathbf{S}$ to see that

$$\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T = \mathbf{A}^{1/2}\mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}}\mathbf{A}^{1/2}.$$

It follows that

$$E := \left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F} = \left\|\mathbf{A}^{1/2}\left(\mathbf{I} - \mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}}\right)\mathbf{A}^{1/2}\right\|_\mathrm{F}.$$

To bound this quantity, we first use the unitary invariance of the Frobenius norm and the fact that

$$\mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}} = \mathbf{U}\mathbf{P}_{\mathbf{\Sigma}^{q-1/2}\mathbf{U}^T\mathbf{S}}\mathbf{U}^T$$

to obtain

$$E^2 = \left\|\mathbf{A}^{1/2}\left(\mathbf{I} - \mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}}\right)\mathbf{A}^{1/2}\right\|_\mathrm{F}^2 = \left\|\mathbf{\Sigma}^{1/2}\left(\mathbf{I} - \mathbf{P}_{\mathbf{\Sigma}^{q-1/2}\mathbf{U}^T\mathbf{S}}\right)\mathbf{\Sigma}^{1/2}\right\|_\mathrm{F}^2.$$

Then we take

$$\mathbf{Z} = \mathbf{\Sigma}^{q-1/2}\mathbf{U}^T\mathbf{S}\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-(q-1/2)} = \begin{pmatrix}\mathbf{I}\\ \mathbf{F}\end{pmatrix}, \tag{12}$$

where $\mathbf{I} \in \mathbb{R}^{k\times k}$ and $\mathbf{F} \in \mathbb{R}^{n-k\times k}$ is given by $\mathbf{F} = \mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-(q-1/2)}$. The latter equality in Equation (12) holds because of our assumption that $\mathbf{\Omega}_1$ has full row rank. Since the range of $\mathbf{Z}$ is contained in the range of $\mathbf{\Sigma}^{q-1/2}\mathbf{U}^T\mathbf{S}$,

$$E^2 \leq \left\|\mathbf{\Sigma}^{1/2}(\mathbf{I} - \mathbf{P}_\mathbf{Z})\mathbf{\Sigma}^{1/2}\right\|_\mathrm{F}^2.$$

By construction, $\mathbf{Z}$ has full column rank, thus $\mathbf{Z}(\mathbf{Z}^T\mathbf{Z})^{-1/2}$ is an orthonormal basis for the span of $\mathbf{Z}$, and

$$\begin{aligned}\mathbf{I} - \mathbf{P}_\mathbf{Z} &= \mathbf{I} - \mathbf{Z}(\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T = \mathbf{I} - \begin{pmatrix}\mathbf{I}\\ \mathbf{F}\end{pmatrix}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\begin{pmatrix}\mathbf{I} & \mathbf{F}^T\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{I} - (\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1} & -(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\\ -\mathbf{F}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1} & \mathbf{I} - \mathbf{F}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\end{pmatrix}.\end{aligned} \tag{13}$$

This implies that

$$\begin{aligned}E^2 &\leq \left\|\mathbf{\Sigma}^{1/2}\begin{pmatrix}\mathbf{I} - (\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1} & -(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\\ -\mathbf{F}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1} & \mathbf{I} - \mathbf{F}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\end{pmatrix}\mathbf{\Sigma}^{1/2}\right\|_\mathrm{F}^2\\ &= \left\|\mathbf{\Sigma}_1^{1/2}\left(\mathbf{I} - (\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\right)\mathbf{\Sigma}_1^{1/2}\right\|_\mathrm{F}^2 + 2\left\|\mathbf{\Sigma}_1^{1/2}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{\Sigma}_2^{1/2}\right\|_\mathrm{F}^2\\ &\quad + \left\|\mathbf{\Sigma}_2^{1/2}\left(\mathbf{I} - \mathbf{F}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\right)\mathbf{\Sigma}_2^{1/2}\right\|_\mathrm{F}^2\\ &:= T_1 + T_2 + T_3.\end{aligned} \tag{14}$$

Next, we provide bounds for T_1, T_2, and T_3. Using the fact that $\mathbf{0} \preceq \mathbf{I} - \mathbf{F}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T \preceq \mathbf{I}$, we can bound T_3 with

$$T_3 \leq \|\mathbf{\Sigma}_2\|_\mathrm{F}^2.$$

Likewise, the fact that $\mathbf{I} - (\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1} \preceq \mathbf{F}^T\mathbf{F}$ (easily seen with an SVD) implies that we can bound T_1 as

$$\begin{aligned}T_1 &\leq \left\|\mathbf{\Sigma}_1^{1/2}\mathbf{F}^T\mathbf{F}\mathbf{\Sigma}_1^{1/2}\right\|_\mathrm{F}^2 \leq \left\|\mathbf{F}\mathbf{\Sigma}_1^{1/2}\right\|_2^2\left\|\mathbf{F}\mathbf{\Sigma}_1^{1/2}\right\|_\mathrm{F}^2\\ &= \left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-(q-1)}\right\|_2^2\left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-(q-1)}\right\|_\mathrm{F}^2\\ &\leq \left\|\mathbf{\Sigma}_2^{q-1}\right\|_2^4\left\|\mathbf{\Sigma}_1^{-(q-1)}\right\|_2^4\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}^2\\ &= \left(\|\mathbf{\Sigma}_2\|_2\left\|\mathbf{\Sigma}_1^{-1}\right\|_2\right)^{4(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}^2\\ &= \left(\frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}\right)^{4(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}^2.\end{aligned}$$

We proceed to bound T_2 by using the estimate

$$T_2 \leq 2\left\|\mathbf{\Sigma}_1^{1/2}(\mathbf{I} + \mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\right\|_\mathrm{F}^2. \tag{15}$$

To develop the term involving a spectral norm, observe that for any SPSD matrix $\mathbf{M}$ with eigenvalue decomposition $\mathbf{M} = \mathbf{V}\mathbf{D}\mathbf{V}^T$,

$$\begin{aligned}(\mathbf{I} + \mathbf{M})^{-1}\mathbf{M}(\mathbf{I} + \mathbf{M})^{-1} &= (\mathbf{V}\mathbf{V}^T + \mathbf{V}\mathbf{D}\mathbf{V}^T)^{-1}\mathbf{V}\mathbf{D}\mathbf{V}^T(\mathbf{V}\mathbf{V}^T + \mathbf{V}\mathbf{D}\mathbf{V}^T)^{-1}\\ &= \mathbf{V}(\mathbf{I} + \mathbf{D})^{-1}\mathbf{D}(\mathbf{I} + \mathbf{D})^{-1}\mathbf{V}^T\\ &\preceq \mathbf{V}\mathbf{D}\mathbf{V}^T = \mathbf{M}.\end{aligned}$$

It follows that

$$\begin{aligned}\left\|\mathbf{\Sigma}_1^{1/2}(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\right\|_2^2 &= \left\|\mathbf{\Sigma}_1^{1/2}(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{F}(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1}\mathbf{\Sigma}_1^{1/2}\right\|_2 \\ &\leq \left\|\mathbf{\Sigma}_1^{1/2}\mathbf{F}^T\mathbf{F}\mathbf{\Sigma}_1^{1/2}\right\|_2 = \left\|\mathbf{F}\mathbf{\Sigma}_1^{1/2}\right\|_2^2 \\ &= \left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-(q-1)}\right\|_2^2 \\ &\leq \left\|\mathbf{\Sigma}_2^{q-1}\right\|_2^2\left\|\mathbf{\Sigma}_1^{-(q-1)}\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2 \\ &= \left(\frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}\right)^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2.\end{aligned}$$

Using this estimate in Equation (15), we conclude that

$$T_2 \leq 2\left(\frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}\right)^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\right\|_{\mathrm{F}}^2.$$

Combining our estimates for T_1, T_2, and T_3 with Equation (14) gives

$$\begin{aligned}E^2 = \left\|\mathbf{A}^{1/2}\left(\mathbf{I}-\mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}}\right)\mathbf{A}^{1/2}\right\|_{\mathrm{F}}^2 &\leq \|\mathbf{\Sigma}_2\|_{\mathrm{F}}^2 \\ + \left(\frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}\right)^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2 &\cdot\left(2\left\|\mathbf{\Sigma}_2^{1/2}\right\|_{\mathrm{F}}^2 + \left(\frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}\right)^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2\right).\end{aligned}$$

The claimed bound follows by identifying γ and applying the subadditivity of the square-root function:

$$E \leq \|\mathbf{\Sigma}_2\|_{\mathrm{F}} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2\cdot\left(\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}\right).$$

■

Remark. The quality of approximation guarantee provided by Theorem 3 depends on the quantities $\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2$ and $\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}$; these quantities reflect the extent to which the sketching matrix is aligned with the eigenspaces of $\mathbf{A}$. The dependence on γ captures the facts that the power method is effective only when there is spectral decay, and that larger gaps between the k and $k+1$ eigenvalues lead to smaller errors when the power method is used.

Remark. To obtain a greater understanding of the additional error term in Theorem 3, assume that $\mathbf{S}$ is a particularly effective sketching matrix, so that $\left\|\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2 = \mathrm{O}(1)$. Then

$$\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2 = \mathrm{O}\left(\|\mathbf{\Sigma}_2\|_2^{1/2}\right) \quad \text{and} \quad \sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)} + \left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}} = \mathrm{O}\left(\|\mathbf{\Sigma}_2\|_\star^{1/2}\right),$$

and the theorem guarantees that the additional error is on the order of $\sqrt{\|\mathbf{\Sigma}_2\|_2\|\mathbf{\Sigma}_2\|_\star}$. This is an upper bound on the optimal Frobenius error:

$$\|\mathbf{\Sigma}_2\|_{\mathrm{F}} \leq \sqrt{\|\mathbf{\Sigma}_2\|_2\|\mathbf{\Sigma}_2\|_\star}.$$

We see, in particular, that if the residual spectrum is flat, i.e. $\lambda_{k+1}(\mathbf{A}) = \cdots = \lambda_n(\mathbf{A})$, then equality holds and the additional error is on the scale of the optimal error.

4.1.3 Trace Norm Bounds

Finally, we state and prove the following bound on the trace norm of the residual error. The proof method is analogous to that for the spectral and Frobenius norm bounds.

As in the case of the Frobenius norm error, we see that the multiplicative eigengap $\gamma = \lambda_{k+1}(\mathbf{A})/\lambda_k(\mathbf{A})$ predicts the effect of using the power method when constructing sketches.

Theorem 4 *Let $\mathbf{A}$ be an $n\times n$ SPSD matrix with eigenvalue decomposition partitioned as in Equation (1), $\mathbf{S}$ be a sketching matrix of size $n\times\ell$, q be a positive integer, $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ be as defined in Equation (3), and define*

$$\gamma = \frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}.$$

Then when $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ and $\mathbf{W} = \mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$, the corresponding low-rank SPSD approximation satisfies

$$\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\star \leq \operatorname{Tr}(\mathbf{\Sigma}_2) + \gamma^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2,$$

assuming $\mathbf{\Omega}_1$ has full row rank.

Proof Since $\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T = \mathbf{A}^{1/2}(\mathbf{I}-\mathbf{P}_{\mathbf{A}^{q-1/2}\mathbf{S}})\mathbf{A}^{1/2} \succeq \mathbf{0}$, its trace norm simplifies to its trace. Thus

$$\begin{aligned}\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\star &= \operatorname{Tr}\left(\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right) = \operatorname{Tr}\left(\mathbf{\Sigma}^{1/2}\left(\mathbf{I}-\mathbf{P}_{\mathbf{\Sigma}^{q-1/2}\mathbf{S}}\right)\mathbf{\Sigma}^{1/2}\right) \\ &\leq \operatorname{Tr}\left(\mathbf{\Sigma}^{1/2}(\mathbf{I}-\mathbf{P}_{\mathbf{Z}})\mathbf{\Sigma}^{1/2}\right),\end{aligned}$$

where $\mathbf{Z} = \begin{pmatrix}\mathbf{I}\\ \mathbf{F}\end{pmatrix}$ is defined in Equation (12). The expression for $\mathbf{I}-\mathbf{P}_{\mathbf{Z}}$ given in Equation (13) implies that

$$\begin{aligned}&\operatorname{Tr}\left(\mathbf{\Sigma}^{1/2}(\mathbf{I}-\mathbf{P}_{\mathbf{Z}})\mathbf{\Sigma}^{1/2}\right) \\ &\qquad = \operatorname{Tr}\left(\mathbf{\Sigma}_1^{1/2}(\mathbf{I}-(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1})\mathbf{\Sigma}_1^{1/2}\right) + \operatorname{Tr}\left(\mathbf{\Sigma}_2^{1/2}(\mathbf{I}-\mathbf{F}(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T)\mathbf{\Sigma}_2^{1/2}\right).\end{aligned}$$

Recall the estimate $\mathbf{I}-(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1} \preceq \mathbf{F}^T\mathbf{F}$ and the basic estimate $\mathbf{I}-\mathbf{F}(\mathbf{I}+\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T \preceq \mathbf{I}$. Together these imply that

$$\begin{aligned}\operatorname{Tr}\left(\mathbf{\Sigma}^{1/2}(\mathbf{I}-\mathbf{P}_{\mathbf{Z}})\mathbf{\Sigma}^{1/2}\right) &\leq \operatorname{Tr}\left(\mathbf{\Sigma}_1^{1/2}\mathbf{F}^T\mathbf{F}\mathbf{\Sigma}_1^{1/2}\right) + \operatorname{Tr}(\mathbf{\Sigma}_2) \\ &= \operatorname{Tr}(\mathbf{\Sigma}_2) + \left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\mathbf{\Sigma}_1^{-(q-1)}\right\|_{\mathrm{F}}^2 \\ &\leq \operatorname{Tr}(\mathbf{\Sigma}_2) + \left\|\mathbf{\Sigma}_2^{q-1}\right\|_2^2\left\|\mathbf{\Sigma}_1^{-(q-1)}\right\|_2^2\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2 \\ &= \operatorname{Tr}(\mathbf{\Sigma}_2) + \gamma^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2.\end{aligned}$$

The first equality follows from substituting the definition of $\mathbf{F}$ and identifying the squared Frobenius norm. The last equality follows from identifying γ. We have established the claimed bound.

■

Remark. Since the identity $\|\mathbf{X}\|_{\mathrm{F}}^2 = \left\|\mathbf{X}\mathbf{X}^T\right\|_\star$ holds for any matrix $\mathbf{X}$, the squared Frobenius norm term present in the deterministic error bound for the trace norm error is on the scale of $\|\mathbf{\Sigma}_2\|_\star$ when $\left\|\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2$ is O(1).

4.1.4 Additional Remarks on Our Deterministic Structural Results

Before applying these deterministic structural results in particular randomized algorithmic settings, we pause to make several additional remarks about these three theorems.

First, for some randomized sampling schemes, it may be difficult to obtain a sharp bound on $\left\|\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\xi$ for $\xi = 2, F$. In these situations, the bounds on the excess error supplied by Theorems 2, 3, and 4 may be quite pessimistic. On the other hand, since $\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T = \mathbf{A}^{1/2}(\mathbf{I} - \mathbf{P}_{\left(\mathbf{A}^{1/2}\right)^{2q-1}\mathbf{S}})\mathbf{A}^{1/2}$, it follows that $\mathbf{0} \preceq \mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T \preceq \mathbf{A}$. This implies that the errors of *any* approximation generated used the SPSD Sketching Model, deterministic or randomized, satisfy at least the crude bound $\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\xi \leq \|\mathbf{A}\|_\xi$.

Second, we emphasize that these theorems are deterministic structural results that bound the additional error (beyond that of the optimal rank-k approximation) of low-rank approximations which follow our SPSD sketching model. That is, there is no randomness in their statement or analysis. In particular, these bounds hold for deterministic as well as randomized sketching matrices $\mathbf{S}$. In the latter case, the randomness enters only through $\mathbf{S}$, and one needs to show that the condition that $\mathbf{\Omega}_1$ has full row rank is satisfied with high probability; conditioned on this, the quality of the bound is determined by terms that depend on how the sketching matrix interacts with the subspace structure of the matrix $\mathbf{A}$.

In particular, we remind the reader that (although it is beyond the scope of this paper to explore this point in detail) these deterministic structural results could be used to check, in an *a posteriori* manner, the quality of a sketching method for which one cannot establish an *a priori* bound.

Third, we also emphasize that the assumption that $\mathbf{\Omega}_1$ has full row rank (equivalently, that $\tan(\mathbf{S}, \mathbf{U}_1) < \infty$) is very non-trivial; and that it is false, in worst-case at least and for non-trivial parameter values, for common sketching methods such as uniform sampling. To see that some version of leverage-based sampling is needed to ensure this condition, recall that $\mathbf{U}_1^T\mathbf{U}_1 = \mathbf{I}$ and thus that $\mathbf{\Omega}_1\mathbf{\Omega}_1^T = \mathbf{U}_1^T\mathbf{S}\mathbf{S}^T\mathbf{U}_1$ can be viewed as approximating $\mathbf{I}$ with a small number of rank-1 components of $\mathbf{U}_1^T\mathbf{U}_1$. The condition that $\mathbf{\Omega}_1$ has full row rank is equivalent to $\left\|\mathbf{U}_1^T\mathbf{U}_1 - \mathbf{U}_1^T\mathbf{S}\mathbf{S}^T\mathbf{U}_1\right\|_2 < 1$. Work on approximating the product of matrices by random sampling shows that to obtain non-trivial bounds one must sample with respect to the norm of the rank-1 components (Drineas et al., 2006), which here (since we are approximating the product of two orthogonal matrices) equal the statistical leverage scores. From this perspective, random projections satisfy this condition since (informally) they rotate to a random basis where the leverage scores of the rotated matrix are approximately uniform and thus where uniform sampling is appropriate (Drineas et al., 2010; Mahoney, 2011).

Finally, as observed recently in Bach (2013), methods that use knowledge of a matrix square root Φ (*i.e.*, a Φ such that $\mathbf{A} = \Phi\Phi^T$) typically lead to $\Omega(n^2)$ complexity. An important feature of our approach is that we only use the matrix square root implicitly—that is, inside the analysis, and not in the statement of the algorithm—and thus we do *not* incur any such cost.

4.2 Stochastic Error Bounds for Low-rank SPSD Approximation

In this section, we apply the three theorems from Section 4.1 to bound the reconstruction errors for several random sampling and random projection methods that conform to our SPSD Sketching Model. In particular, we consider two variants of random sampling and two variants of random projections: sampling columns according to an importance sampling distribution that depends on the statistical leverage scores (in Section 4.2.1); randomly projecting by using subsampled randomized Fourier transformations (in Section 4.2.2); randomly projecting by uniformly sampling from Gaussian mixtures of the columns (in Section 4.2.3); and, finally, sampling columns uniformly at random (in Section 4.2.4).

The results are presented for the general case of SPSD sketches constructed using the power method, i.e., sketches constructed using $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ for a positive integer q. The additive errors of these sketches decrease proportionally to the number of iterations q, where the constant of proportionality is given by the multiplicative eigengap $\gamma = \lambda_{k+1}(\mathbf{A})/\lambda_k(\mathbf{A})$. Accordingly, the bounds involve the terms γ^{q-1} and $\gamma^{2(q-1)}$. The bounds simplify considerably when $q = 1$ (*i.e.*, when there are no additional iterations) or $\gamma = 1$ (*i.e.*, when there is no eigengap). In either of these cases, the terms γ^{q-1} and $\gamma^{2(q-1)}$ all become the constant 1.

Before establishing these results, we pause here to provide a brief review of running time issues, some of which were addressed empirically in Section 3. The computational bottleneck for random sampling algorithms (except for uniform sampling that we address in Section 4.2.4, which is trivial to implement) is often the exact or approximate computation of the importance sampling distribution with respect to which one samples; and the computational bottleneck for random projection methods is often the implementation of the random projection. For example, if the sketching matrix $\mathbf{S}$ is a random projection constructed as an $n \times \ell$ matrix of i.i.d. Gaussian random variables, as we use in Section 4.2.3, then the running time of dense data in RAM is not substantially faster than computing $\mathbf{U}_1$, while the running time can be much faster for certain sparse matrices or for computation in parallel or distributed environments. Alternately, if the sketching matrix $\mathbf{S}$ is a Fourier-based projection, as we use in Section 4.2.2, then the running time for data stored in RAM is typically $O(n^2 \ln k)$, as opposed to the $O(n^2 k)$ time that would be needed to compute $\mathbf{U}_1$. These running times depend sensitively on the size of the data and the model of data access; see Mahoney (2011); Halko et al. (2011) for detailed discussions of these issues.

In particular, for random sampling algorithms that use a leverage-based importance sampling distribution, as we use in Section 4.2.1, it is often said that the running time is no faster than that of computing $\mathbf{U}_1$. (This $O(n^2 k)$ running time claim is simply the running time of the naïve algorithm that computes $\mathbf{U}_1$ "exactly," *e.g.*, with a variant of the QR decomposition, and then reads off the Euclidean norms of the rows.) However, the randomized algorithm of Drineas et al. (2012) that computes relative-error approximations to *all* of the statistical leverage in a time that is qualitatively faster—in worst-case theory

and, by using existing high-quality randomized numerical code (Avron et al., 2010; Meng et al., 2014; Halko et al., 2011), in practice—gets around this bottleneck, as was shown in Section 3. The computational bottleneck for the algorithms of Drineas et al. (2012) is that of applying a random projection, and thus the running time for leverage-based Nyström extension is that of applying a ("fast" Fourier-based or "slow" Gaussian-based, as appropriate) random projection to $\mathbf{A}$ (Drineas et al., 2012). See Section 3 or (Avron et al., 2010; Meng et al., 2014; Halko et al., 2011) for additional details.

4.2.1 Sampling with Leverage-based Importance Sampling Probabilities

Here, the columns of $\mathbf{A}$ are sampled with replacement according to a nonuniform probability distribution determined by the (exact or approximate) statistical leverage scores of $\mathbf{A}$ relative to the best rank-k approximation to $\mathbf{A}$, which in turn depend on nonuniformity properties of the top k-dimensional eigenspace of $\mathbf{A}$. To add flexibility (*e.g.*, in case the scores are computed only approximately with the fast algorithm of Drineas et al. (2012)), we formulate the following lemma in terms of any probability distribution that is β-close to the leverage score distribution. In particular, consider any probability distribution satisfying

$$p_j \geq \frac{\beta}{k} \left\|(\mathbf{U}_1)_j\right\|_2^2 \quad \text{and} \quad \sum_{j=1}^n p_j = 1,$$

where $\beta \in (0,1]$. Given these (β-approximate) leverage-based probabilities, the sketching matrix is $\mathbf{S} = \mathbf{RD}$ where $\mathbf{R} \in \mathbb{R}^{n\times\ell}$ is a column selection matrix that samples columns of $\mathbf{A}$ from the given distribution—*i.e.*, $\mathbf{R}_{ij} = 1$ iff the ith column of $\mathbf{A}$ is the jth column selected—and $\mathbf{D}$ is a diagonal rescaling matrix satisfying $\mathbf{D}_{jj} = \frac{1}{\sqrt{\ell p_i}}$ iff $\mathbf{R}_{ij} = 1$. For this case, we can prove the following.

Lemma 5 *Let $\mathbf{A}$ be an $n \times n$ SPSD matrix, q be a positive integer, and $\mathbf{S}$ be a sampling matrix of size $n \times \ell$ corresponding to a leverage-based probability distribution derived from the top k-dimensional eigenspace of $\mathbf{A}$, satisfying*

$$p_j \geq \frac{\beta}{k} \left\|(\mathbf{U}_1)_j\right\|_2^2 \quad \textit{and} \quad \sum_{j=1}^n p_j = 1$$

for some $\beta \in (0,1]$. Fix a failure probability $\delta \in (0,1]$ and approximation factor $\epsilon \in (0,1]$, and let

$$\gamma = \frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}.$$

If $\ell \geq 3200(\beta\epsilon^2)^{-1}k\ln(4k/(\beta\delta))$, then, when $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ and $\mathbf{W} = \mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$, the corresponding low-rank SPSD approximation satisfies

$$\left\|\mathbf{A} - \mathbf{CW}^\dagger\mathbf{C}^T\right\|_2 \leq \|\mathbf{A} - \mathbf{A}_k\|_2 + \left(\epsilon^2 \left\|(\mathbf{A} - \mathbf{A}_k)^{2q-1}\right\|_\star\right)^{1/(2q-1)}, \tag{16}$$

$$\left\|\mathbf{A} - \mathbf{CW}^\dagger\mathbf{C}^T\right\|_{\mathrm{F}} \leq \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} + \left(\sqrt{2}\epsilon\gamma^{q-1} + \epsilon^2\gamma^{2(q-1)}\right)\|\mathbf{A} - \mathbf{A}_k\|_\star, \textit{ and} \tag{17}$$

$$\left\|\mathbf{A} - \mathbf{CW}^\dagger\mathbf{C}^T\right\|_\star \leq (1 + \gamma^{2(q-1)}\epsilon^2)\|\mathbf{A} - \mathbf{A}_k\|_\star, \tag{18}$$

simultaneously with probability at least $1 - 6\delta - 0.6$.

Proof In (Mackey et al., 2011a, proof of Proposition 22) it is shown that if ℓ satisfies the given bound and the samples are drawn from an approximate subspace probability distribution, then for any SPSD diagonal matrix $\mathbf{D}$,

$$\left\|\mathbf{D}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_{\mathrm{F}} \leq \epsilon\|\mathbf{D}\|_{\mathrm{F}}$$

with probability at least $1 - 2\delta - 0.2$. Thus, the estimates

$$\left\|\boldsymbol{\Sigma}_2^{1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_{\mathrm{F}} \leq \epsilon\left\|\boldsymbol{\Sigma}_2^{1/2}\right\|_{\mathrm{F}} = \epsilon\sqrt{\mathrm{Tr}\left(\boldsymbol{\Sigma}_2\right)} = \epsilon\sqrt{\|\mathbf{A} - \mathbf{A}_k\|_\star},$$

and

$$\begin{aligned}
\left(\left\|\boldsymbol{\Sigma}_2^{q-1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_2\right)^{2/(2q-1)} &\leq \left(\left\|\boldsymbol{\Sigma}_2^{p-1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_{\mathrm{F}}\right)^{2/(2q-1)} \\
&\leq \left(\epsilon^2\left\|\boldsymbol{\Sigma}_2^{q-1/2}\right\|_{\mathrm{F}}^2\right)^{1/(2q-1)} \\
&= \left(\epsilon^2\,\mathrm{Tr}\left(\boldsymbol{\Sigma}_2^{2q-1}\right)\right)^{1/(2q-1)} \\
&= \left(\epsilon^2\left\|(\mathbf{A} - \mathbf{A}_k)^{2q-1}\right\|_\star\right)^{1/(2q-1)}
\end{aligned}$$

each hold, individually, with probability at least $1 - 2\delta - 0.2$. In particular, taking $q = 1$, we see that

$$\left\|\boldsymbol{\Sigma}_2^{1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_2 \leq \epsilon\sqrt{\|\mathbf{A} - \mathbf{A}_k\|_\star}$$

with the same probability.

These three estimates used in Theorems 2, 3, and 4 yield the bounds given in the statement of the theorem. ■

Remark. The additive scale factors for the spectral and Frobenius norm bounds are much improved relative to the prior results of Drineas and Mahoney (2005). At root, this is since the leverage score importance sampling probabilities highlight structural properties of the data (*e.g.*, how to satisfy the condition in Theorems 2, 3, and 4 that $\boldsymbol{\Omega}_1$ has full row rank) in a more refined way than the importance sampling probabilities of Drineas and Mahoney (2005).

Remark. These improvements come at additional computational expense, but we remind the reader that leverage-based sampling probabilities of the form used by Lemma 5 can be computed faster than the time needed to compute the basis $\mathbf{U}_1$ (Drineas et al., 2012). The computational bottleneck of the algorithm of Drineas et al. (2012) is the time required to perform a random projection on the input matrix.

Remark. Not surprisingly, constant factors such as 3200 (as well as other similarly large factors below) and a failure probability bounded away from zero are artifacts of the analysis; the empirical behavior of this sampling method is much better. This has been observed previously (Drineas et al., 2008; Mahoney and Drineas, 2009).

4.2.2 Random Projections with Subsampled Randomized Fourier Transforms

Here, the columns of $\mathbf{A}$ are randomly mixed using a unitary matrix before the columns are sampled. In particular, $\mathbf{S} = \sqrt{\frac{n}{\ell}}\mathbf{DTR}$, where $\mathbf{D}$ is a diagonal matrix of Rademacher random variables, $\mathbf{T}$ is a highly incoherent unitary matrix, and $\mathbf{R}$ restricts to ℓ columns. For concreteness, and because it has an associated fast transform, we consider the case where $\mathbf{T}$ is the normalized Fourier transform of size $n \times n$. For this case, we can prove the following.

Lemma 6 *Let $\mathbf{A}$ be an $n \times n$ SPSD matrix, q be a positive integer, and $\mathbf{S} = \sqrt{\frac{n}{\ell}}\mathbf{DFR}$ be a sampling matrix of size $n \times \ell$, where $\mathbf{D}$ is a diagonal matrix of Rademacher random variables, $\mathbf{F}$ is a normalized Fourier matrix of size $n \times n$, and $\mathbf{R}$ restricts to ℓ columns. Fix a failure probability $\delta \in (0,1)$, approximation factor $\epsilon \in (0,1)$, and assume that $k \geq 4$. Define*

$$\gamma = \frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}.$$

If $\ell \geq 24\epsilon^{-1}[\sqrt{k} + \sqrt{8\ln(8n/\delta)}]^2 \ln(8k/\delta)$, then, when $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ and $\mathbf{W} = \mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$, the corresponding low-rank SPSD approximation satisfies

$$\begin{aligned}
\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2 &\leq \left[1 + \left(\frac{1}{1-\sqrt{\epsilon}} \cdot \left(5 + \frac{16\ln(n/\delta)^2}{\ell}\right)\right)^{1/(2q-1)}\right] \cdot \|\mathbf{A} - \mathbf{A}_k\|_2 \\
&\quad + \left(\frac{2\ln(n/\delta)}{(1-\sqrt{\epsilon})\ell}\right)^{1/(2q-1)} \left\|(\mathbf{A} - \mathbf{A}_k)^{2q-1}\right\|_\star^{1/(2q-1)}, \qquad (19) \\
\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F} &\leq \|\mathbf{A} - \mathbf{A}_k\|_\mathrm{F} + \left(7\gamma^{q-1}\sqrt{\epsilon} + 22\gamma^{2q-2}\epsilon\right)\|\mathbf{A} - \mathbf{A}_k\|_\star, \text{ and} \\
\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\star &\leq (1 + 22\epsilon\gamma^{2(q-1)})\|\mathbf{A} - \mathbf{A}_k\|_\star
\end{aligned}$$

simultaneously with probability at least $1 - 2\delta$.

Proof Let $\mathbf{M} = \mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger$ denote the matrix referenced in Theorems 2 and 4. In (Boutsidis and Gittens, 2013, proof of Theorem 4), it is shown that for the stated choice of $\mathbf{S}$ and number of samples ℓ,

$$\begin{aligned}
\|\mathbf{M}\|_2^2 &\leq \frac{1}{1-\sqrt{\epsilon}} \cdot \left(5\|\mathbf{\Sigma}_2\|_2^{2q-1} + \frac{\ln(n/\delta)}{\ell}\left(\left\|\mathbf{\Sigma}_2^{q-1/2}\right\|_\mathrm{F} + \sqrt{8\ln(n/\delta)}\left\|\mathbf{\Sigma}_2^{q-1/2}\right\|_2\right)^2\right) \\
&= \frac{1}{1-\sqrt{\epsilon}} \cdot \left(5\|\mathbf{\Sigma}\|_2^{2q-1} + \frac{\ln(n/\delta)}{\ell}\left(\left\|\mathbf{\Sigma}_2^{2q-1}\right\|_\star^{1/2} + \sqrt{8\ln(n/\delta)}\|\mathbf{\Sigma}_2\|_2^{q-1/2}\right)^2\right) \\
&\leq \frac{1}{1-\sqrt{\epsilon}} \cdot \left(\left(5 + \frac{16\ln(n/\delta)^2}{\ell}\right)\|\mathbf{\Sigma}_2\|_2^{2q-1} + \frac{2\ln(n/\delta)}{\ell}\left\|\mathbf{\Sigma}_2^{2q-1}\right\|_\star\right)
\end{aligned}$$

and

$$\|\mathbf{M}\|_\mathrm{F} \leq \sqrt{22\epsilon}\left\|\mathbf{\Sigma}_2^{1/2}\right\|_\mathrm{F} = \sqrt{22\epsilon\|\mathbf{\Sigma}_2\|_\star}$$

each hold, individually, with probability at least $1 - \delta$. These estimates used in Theorems 2 and 4 yield the stated bounds for the spectral and trace norm errors.

The Frobenius norm bound follows from the same estimates and a simplification of the bound stated in Theorem 3:

$$\begin{aligned}
\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F} &\leq \|\mathbf{\Sigma}_2\|_\mathrm{F} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2\left(\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}\right) \\
&\leq \|\mathbf{\Sigma}_2\|_\mathrm{F} + \gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)} + \gamma^{2(q-1)}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}^2 \\
&\leq \|\mathbf{\Sigma}_2\|_\mathrm{F} + \left(\gamma^{q-1}\sqrt{44\epsilon} + 22\gamma^{2q-2}\epsilon\right)\|\mathbf{\Sigma}_2\|_\star.
\end{aligned}$$

We note that a direct application of Theorem 3 gives a potentially tighter, but more unwieldy, bound. ■

Remark. Suppressing the dependence on δ and ϵ, the spectral norm bound ensures that when $q = 1$, $k = \Omega(\ln n)$ and $\ell = \Omega(k\ln k)$, then

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2 = \mathrm{O}\left(\frac{\ln n}{\ln k}\|\mathbf{A} - \mathbf{A}_k\|_2 + \frac{1}{\ln k}\|\mathbf{A} - \mathbf{A}_k\|_\star\right).$$

This should be compared to the guarantee established in Lemma 7 below for Gaussian-based SPSD sketches constructed using the same number of measurements:

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2 = \mathrm{O}\left(\|\mathbf{A} - \mathbf{A}_k\|_2 + \frac{1}{k\ln k}\|\mathbf{A} - \mathbf{A}_k\|_\star\right).$$

Lemma 6 guarantees that errors on this order can be achieved if one increases the number of samples by a logarithm factor in the dimension: specifically, such a bound is achieved when $k = \Omega(\ln n)$ and $\ell = \Omega(k\ln k\ln n)$. The difference between the number of samples necessary for Fourier-based sketches and Gaussian-based sketches is reflective of the differing natures of the random projections: the geometry of any k-dimensional subspace is preserved under projection onto the span of $\ell = \mathrm{O}(k)$ Gaussian random vectors (Halko et al., 2011), but the sharpest analysis available suggests that to preserve the geometry of such a subspace under projection onto the span of ℓ SRFT vectors, ℓ must satisfy $\ell = \Omega(\max\{k, \ln n\}\ln k)$ (Tropp, 2011). We note, however, that in practice the Fourier-based and Gaussian-based SPSD sketches have similar reconstruction errors.

Remark. The structure of the Frobenius and trace norm bounds for the Fourier-based projection are identical to the structure of the corresponding bounds from Lemma 5 for leverage-based sampling (and the bounds could be made identical with appropriate choice of parameters). This is not surprising since (informally) Fourier-based (and other) random projections rotate to a random basis where the leverage scores are approximately uniform and thus where uniform sampling is appropriate (Mahoney, 2011). The disparity of the spectral norm bounds suggests that leverage-based SPSD sketches should be expected to be more accurate in the spectral norm than Fourier-based sketches; the empirical results of Section 3.4 support this interpretation. The running times of the Fourier-based and the leverage-based algorithms are the same, to leading order, if the algorithm of Drineas et al. (2012) (which uses the same transform $\mathbf{S} = \sqrt{\frac{n}{\ell}}\mathbf{DTR}$) is used to approximate the leverage scores.

4.2.3 Random Projections with i.i.d. Gaussian Random Matrices

Here, the columns of $\mathbf{A}$ are randomly mixed using Gaussian random variables before sampling. Thus, the entries of the sampling matrix $\mathbf{S} \in \mathbb{R}^{n\times\ell}$ are i.i.d. standard Gaussian random variables.

Lemma 7 *Let $\mathbf{A}$ be an $n \times n$ SPSD matrix, q be a positive integer, $\mathbf{S} \in \mathbb{R}^{n\times\ell}$ be a matrix of i.i.d standard Gaussians, and define*

$$\gamma = \frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}.$$

If $\ell \geq 2\epsilon^{-2}k\ln k$ where $\epsilon \in (0,1)$ and $k > 4$, then, when $\mathbf{C} = \mathbf{A}^q\mathbf{S}$ and $\mathbf{W} = \mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$, the corresponding low-rank SPSD approximation satisfies

$$\begin{aligned}
\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2 &\leq \left(1+\left(89\frac{\epsilon^2}{\ln k}+874\frac{\epsilon^2}{k}\right)^{1/(2q-1)}\right)\|\mathbf{A}-\mathbf{A}_k\|_2 \\
&\quad+\left(219\frac{\epsilon^2}{k\ln k}\right)^{1/(2q-1)}\cdot\|\mathbf{A}-\mathbf{A}_k\|_\star, \\
\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_{\mathrm{F}} &\leq \|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}+\left[\gamma^{q-1}\epsilon\left(\frac{42}{\sqrt{k}}+\frac{14}{\sqrt{\ln k}}\right)\right. \\
&\quad\left.+\gamma^{2q-2}\epsilon^2\left(\frac{45}{\ln k}+\frac{140}{\sqrt{k\ln k}}+\frac{219}{k\sqrt{\ln k}}\right)\right]\sqrt{\|\mathbf{A}-\mathbf{A}_k\|_2\,\|\mathbf{A}-\mathbf{A}_k\|_\star} \\
&\quad+\left(21\gamma^{q-1}\frac{\epsilon}{\sqrt{k\ln k}}+70\gamma^{2q-2}\frac{\epsilon^2}{\sqrt{k}\ln k}\right)\|\mathbf{A}-\mathbf{A}_k\|_\star \\
&\quad+\gamma^{2q-2}\epsilon^2\left(\frac{140}{\sqrt{k\ln k}}+\frac{437}{k}\right)\|\mathbf{A}-\mathbf{A}_k\|_2, \text{ and} \\
\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\star &\leq \left(1+45\frac{\gamma^{2q-2}\epsilon^2}{\ln k}\right)\|\mathbf{A}-\mathbf{A}_k\|_\star+437\frac{\gamma^{2q-2}\epsilon^2}{k}\|\mathbf{A}-\mathbf{A}_k\|_2
\end{aligned}$$

simultaneously with probability at least $1-2k^{-1}-4k^{-k/\epsilon^2}$.

Remark. The way we have parameterized these bounds for Gaussian-based projections makes explicit the dependence on various parameters, but hides the structural simplicity of these bounds. In particular, note that the Frobenius norm approximation error is upper bounded by a term that depends on the Frobenius norm error of the optimal low-rank approximant and a term that depends on the trace norm error of the optimal low-rank approximant; and that, similarly, the trace norm approximation error is upper bounded by a multiplicative factor that can be set to $1+\epsilon$ with an appropriate choice of parameters.

Proof As before, this result is established by bounding the quantities involved in Theorems 2, 3, and 4. The following deviation bounds, established in (Halko et al., 2011, Section 10), are useful in that regard: if $\mathbf{D}$ is a diagonal matrix, $\ell = k+p$ with $p > 4$ and $u, t \geq 1$,

then with our choice of $\mathbf{S}$,

$$\begin{aligned}
\mathbb{P}\left\{\left\|\mathbf{D}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_2 > \|\mathbf{D}\|_2\left(\sqrt{\frac{3k}{p+1}}\cdot t+\frac{\mathrm{e}\sqrt{\ell}}{p+1}\cdot tu\right)+\|\mathbf{D}\|_{\mathrm{F}}\frac{\mathrm{e}\sqrt{\ell}}{p+1}\cdot t\right\} &\leq 2t^{-p}+\mathrm{e}^{-u^2/2}, \text{ and} \\
\mathbb{P}\left\{\left\|\mathbf{D}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_{\mathrm{F}} > \|\mathbf{D}\|_{\mathrm{F}}\sqrt{\frac{3k}{p+1}}\cdot t+\|\mathbf{D}\|_2\frac{\mathrm{e}\sqrt{\ell}}{p+1}\cdot tu\right\} &\leq 2t^{-p}+\mathrm{e}^{-u^2/2}.
\end{aligned} \tag{20}$$

Write $\ell = k+p$. Since $\ell \geq 2\epsilon^{-2}k\ln k$, we have that $p \geq \epsilon^{-2}k\ln k$. Accordingly, the following estimates hold:

$$\begin{aligned}
\sqrt{\frac{3k}{p+1}} &\leq \sqrt{\frac{3k}{p}} \leq \sqrt{\frac{3}{\ln k}}\epsilon \\
\frac{\sqrt{\ell}}{p+1} &\leq \frac{\sqrt{k+p}}{p} \leq \sqrt{\frac{\epsilon^4}{k\ln^2 k}+\frac{\epsilon^2}{k\ln k}} < \sqrt{\frac{2}{k\ln k}}\epsilon.
\end{aligned}$$

Use these estimates and take $t = \mathrm{e}$ and $u = \sqrt{2\ln k}$ in (20) to obtain that

$$\begin{aligned}
\left\|\boldsymbol{\Sigma}_2^{q-1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_2^2 &\leq \left[\epsilon\left(\mathrm{e}\sqrt{\frac{3}{\ln k}}+2\mathrm{e}^2\sqrt{\frac{1}{k}}\right)\cdot\left\|\boldsymbol{\Sigma}_2^{q-1/2}\right\|_2+\epsilon\mathrm{e}^2\sqrt{\frac{2}{k\ln k}}\cdot\left\|\boldsymbol{\Sigma}_2^{q-1/2}\right\|_{\mathrm{F}}\right]^2 \\
&\leq 2\epsilon^2\left(\mathrm{e}\sqrt{\frac{3}{\ln k}}+2\mathrm{e}^2\sqrt{\frac{1}{k}}\right)^2\cdot\|\boldsymbol{\Sigma}_2\|_2^{2q-1}+\frac{4\epsilon^2\mathrm{e}^4}{k\ln k}\cdot\left\|\boldsymbol{\Sigma}_2^{q-1/2}\right\|_{\mathrm{F}}^2 \\
&\leq \left(\frac{12\mathrm{e}^2}{\ln k}+\frac{16\mathrm{e}^4}{k}\right)\epsilon^2\cdot\|\boldsymbol{\Sigma}_2\|_2^{2q-1}+\frac{4\epsilon^2\mathrm{e}^4}{k\ln k}\cdot\left\|\boldsymbol{\Sigma}_2^{2q-1}\right\|_\star
\end{aligned}$$

with probability at least $1-k^{-1}-2k^{-k/\epsilon^2}$ and

$$\begin{aligned}
\left\|\boldsymbol{\Sigma}_2^{1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_{\mathrm{F}} &\leq \sqrt{\frac{3}{\ln k}}\epsilon\mathrm{e}\cdot\left\|\boldsymbol{\Sigma}_2^{1/2}\right\|_{\mathrm{F}}+\frac{2\mathrm{e}^2}{\sqrt{k}}\epsilon\cdot\left\|\boldsymbol{\Sigma}_2^{1/2}\right\|_2 \\
&= \epsilon\mathrm{e}\sqrt{\frac{3}{\ln k}\|\boldsymbol{\Sigma}_2\|_\star}+\frac{2\mathrm{e}^2}{\sqrt{k}}\epsilon\cdot\|\boldsymbol{\Sigma}_2\|_2^{1/2}
\end{aligned}$$

with the same probability. Likewise,

$$\begin{aligned}
\left\|\boldsymbol{\Sigma}_2^{1/2}\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2 &\leq \left(\epsilon\mathrm{e}\sqrt{\frac{3}{\ln k}\|\boldsymbol{\Sigma}_2\|_\star}+\frac{2\mathrm{e}^2}{\sqrt{k}}\epsilon\cdot\|\boldsymbol{\Sigma}_2\|_2^{1/2}\right)^2 \\
&\leq \frac{6}{\ln k}\epsilon^2\mathrm{e}^2\cdot\|\boldsymbol{\Sigma}_2\|_\star+\frac{8\mathrm{e}^4}{k}\epsilon^2\cdot\|\boldsymbol{\Sigma}_2\|_2
\end{aligned}$$

with the same probability.

These estimates used in Theorems 2 and 4 yield the stated spectral and trace norm bounds. To obtain the corresponding Frobenius norm bound, define the quantities

$$\begin{aligned}
G_1 &= \left(\frac{12\mathrm{e}^2}{\ln k}+\frac{16\mathrm{e}^4}{k}\right)\epsilon^2 & G_3 &= 3\mathrm{e}^2\frac{\epsilon^2}{\ln k} \\
G_2 &= 4\mathrm{e}^4\frac{\epsilon^2}{k\ln k} & G_4 &= 4\mathrm{e}^4\frac{\epsilon^2}{k}
\end{aligned}$$

By Theorem 3 and our estimates for $\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2$ and $\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}$,

$$\begin{aligned}\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F} &\leq \|\mathbf{\Sigma}_2\|_\mathrm{F}+\gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2\cdot\left(\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)}+\gamma^{q-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}\right)\\ &\leq \|\mathbf{\Sigma}_2\|_\mathrm{F}+\gamma^{q-1}(G_1\|\mathbf{\Sigma}_2\|_2+G_2\|\mathbf{\Sigma}_2\|_\star)^{1/2}\times\\ &\quad\left(\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)}+\gamma^{q-1}\sqrt{G_3\|\mathbf{\Sigma}_2\|_\star}+\gamma^{q-1}\sqrt{G_4\|\mathbf{\Sigma}_2\|_2}\right)\\ &\leq \|\mathbf{\Sigma}_2\|_\mathrm{F}+\left(\gamma^{q-1}\sqrt{2G_1}+\gamma^{2q-2}(\sqrt{G_1G_3}+\sqrt{G_2G_4})\right)\cdot\sqrt{\|\mathbf{\Sigma}_2\|_2\|\mathbf{\Sigma}_2\|_\star}\\ &\quad+\left(\gamma^{q-1}\sqrt{2G_2}+\gamma^{2q-2}\sqrt{G_2G_3}\right)\cdot\|\mathbf{\Sigma}_2\|_\star\\ &\quad+\gamma^{2q-2}\sqrt{G_1G_4}\,\|\mathbf{\Sigma}_2\|_2.\end{aligned}\tag{21}$$

The following estimates hold for the coefficients in this inequality:

$$\begin{aligned}\sqrt{2G_1}&\leq\left(\frac{42}{\sqrt{k}}+\frac{14}{\sqrt{\ln k}}\right)\epsilon & \sqrt{G_1G_3}&\leq\left(\frac{45}{\ln k}+\frac{140}{\sqrt{k\ln k}}\right)\epsilon^2\\ \sqrt{G_2G_4}&\leq\frac{219}{k\sqrt{\ln k}}\epsilon^2 & \sqrt{2G_2}&\leq 21\frac{\epsilon}{\sqrt{k\ln k}}\\ \sqrt{G_2G_3}&\leq 70\frac{\epsilon^2}{\sqrt{k}\ln k} & \sqrt{G_1G_4}&\leq\left(\frac{140}{\sqrt{k\ln k}}+\frac{437}{k}\right)\epsilon^2.\end{aligned}$$

The Frobenius norm bound follows from using these estimates in Equation (21) and grouping terms appropriately:

$$\begin{aligned}\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F} &\leq \|\mathbf{\Sigma}_2\|_\mathrm{F}+\left[\gamma^{q-1}\epsilon\left(\frac{42}{\sqrt{k}}+\frac{14}{\sqrt{\ln k}}\right)\right.\\ &\qquad\left.+\gamma^{2q-2}\epsilon^2\left(\frac{45}{\ln k}+\frac{140}{\sqrt{k\ln k}}+\frac{219}{k\sqrt{\ln k}}\right)\right]\sqrt{\|\mathbf{\Sigma}_2\|_2\|\mathbf{\Sigma}_2\|_\star}\\ &\quad+\left(21\gamma^{q-1}\frac{\epsilon}{\sqrt{k\ln k}}+70\gamma^{2q-2}\frac{\epsilon^2}{\sqrt{k}\ln k}\right)\cdot\|\mathbf{\Sigma}_2\|_\star\\ &\quad+\gamma^{2q-2}\epsilon^2\left(\frac{140}{\sqrt{k\ln k}}+\frac{437}{k}\right)\|\mathbf{\Sigma}_2\|_2.\end{aligned}$$

■

4.2.4 Sampling Columns Uniformly at Random

Here, the columns of $\mathbf{A}$ are sampled uniformly at random (with or without replacement). Such uniformly-at-random column sampling only makes sense when the leverage scores of the top k-dimensional invariant subspace of the matrix are sufficiently uniform that no column is significantly more informative than the others. For this case, we can prove the following.

Lemma 8 *Let $\mathbf{A}$ be an $n\times n$ SPSD matrix, q be a positive integer, and $\mathbf{S}$ be a sampling matrix of size $n\times\ell$ corresponding to sampling the columns of $\mathbf{A}$ uniformly at random (with*

or without replacement). Let μ denote the coherence of the top k-dimensional eigenspace of $\mathbf{A}$ and fix a failure probability $\delta\in(0,1)$ and accuracy factor $\epsilon\in(0,1)$. Define

$$\gamma=\frac{\lambda_{k+1}(\mathbf{A})}{\lambda_k(\mathbf{A})}.$$

If $\ell\geq 2\mu\epsilon^{-2}k\ln(k/\delta)$, then, when $\mathbf{C}=\mathbf{A}^q\mathbf{S}$ and $\mathbf{W}=\mathbf{S}^T\mathbf{A}^{2q-1}\mathbf{S}$, the corresponding low-rank SPSD approximation satisfies

$$\begin{aligned}&\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_2\leq\left(1+\left(\frac{n}{(1-\epsilon)\ell}\right)^{1/(2q-1)}\right)\|\mathbf{A}-\mathbf{A}_k\|_2,\\ &\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\mathrm{F}\leq\|\mathbf{A}-\mathbf{A}_k\|_\mathrm{F}+\left(\gamma^{q-1}\frac{\sqrt{2}}{\delta\sqrt{1-\epsilon}}+\frac{\gamma^{2q-2}}{(1-\epsilon)\delta^2}\right)\|\mathbf{A}-\mathbf{A}_k\|_\star,\ \textit{and}\\ &\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_\star\leq\left(1+\frac{\gamma^{2q-2}}{\delta^2(1-\epsilon)}\right)\|\mathbf{A}-\mathbf{A}_k\|_\star,\end{aligned}$$

simultaneously with probability at least $1-3\delta$.

Proof In (Gittens, 2012), it is shown that

$$\left\|\mathbf{\Omega}_1^\dagger\right\|_2^2\leq\frac{n}{(1-\epsilon)\ell}$$

with probability at least $1-\delta$ when ℓ satisfies the stated bound. Observe that $\|\mathbf{\Omega}_2\|_2\leq\|\mathbf{U}_2\|_2\|\mathbf{S}\|_2\leq 1$, so that

$$\left\|\mathbf{\Sigma}_2^{q-1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2^2\leq\left\|\mathbf{\Sigma}_2^{q-1/2}\right\|_2^2\left\|\mathbf{\Omega}_1^\dagger\right\|_2^2\leq\|\mathbf{\Sigma}_2\|_2^{2q-1}\frac{n}{(1-\epsilon)\ell}$$

with probability at least $1-\delta$. Also,

$$\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_\mathrm{F}\leq\sqrt{\frac{n}{(1-\epsilon)\ell}}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\right\|_\mathrm{F}\tag{22}$$

with at least the same probability. Observe that since $\mathbf{S}$ selects ℓ columns uniformly at random,

$$\mathbb{E}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\right\|_\mathrm{F}^2=\mathbb{E}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{U}_2^T\mathbf{S}\right\|_\mathrm{F}^2=\sum_{i=1}^{\ell}\mathbb{E}\|\mathbf{x}_i\|^2,$$

where the summands $\mathbf{x}_i$ are distributed uniformly at random over the columns of $\mathbf{\Sigma}_2^{1/2}\mathbf{U}_2^T$. Regardless of whether $\mathbf{S}$ selects the columns with replacement or without replacement, the summands all have the same expectation:

$$\mathbb{E}\|\mathbf{x}_i\|^2=\frac{1}{n}\sum_{j=1}^{n}\|(\mathbf{\Sigma}_2^{1/2}\mathbf{U}_2^T)^j)\|^2=\frac{1}{n}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{U}_2^T\right\|_\mathrm{F}^2=\frac{1}{n}\left\|\mathbf{\Sigma}_2^{1/2}\right\|_\mathrm{F}^2=\frac{1}{n}\|\mathbf{\Sigma}_2\|_\star.$$

Consequently,

$$\mathbb{E}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\right\|_\mathrm{F}^2=\frac{\ell}{n}\|\mathbf{\Sigma}_2\|_\star,$$

so by Jensen's inequality

$$\mathbb{E}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\right\|_{\mathrm{F}} \leq \left(\mathbb{E}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\right\|_{\mathrm{F}}^2\right)^{1/2} = \sqrt{\frac{\ell}{n}\left\|\mathbf{\Sigma}_2\right\|_\star}.$$

Now applying Markov's inequality to (22), we see that

$$\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}} \leq \frac{1}{\delta}\sqrt{\frac{1}{(1-\epsilon)}\left\|\mathbf{\Sigma}_2\right\|_\star}$$

with probability at least $1-2\delta$. Thus, we also know that

$$\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2 \leq \frac{1}{(1-\epsilon)\delta^2}\left\|\mathbf{\Sigma}_2\right\|_\star,$$

also with probability at least $1-2\delta$. These estimates used in Theorems 2 and 4 yield the stated spectral and trace norm bounds.

To obtain the Frobenius norm bound, observe that Theorem 3 implies

$$\begin{aligned}\left\|\mathbf{A}-\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\right\|_{\mathrm{F}} &\leq \left\|\mathbf{\Sigma}_2\right\|_{\mathrm{F}} + \gamma^{p-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_2\left(\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)}+\gamma^{p-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}\right)\\ &\leq \left\|\mathbf{\Sigma}_2\right\|_{\mathrm{F}} + +\gamma^{p-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}\left(\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)}+\gamma^{p-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}\right)\\ &\leq \left\|\mathbf{\Sigma}_2\right\|_{\mathrm{F}} + \gamma^{2p-2}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2 + \gamma^{p-1}\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}\sqrt{2\operatorname{Tr}(\mathbf{\Sigma}_2)}.\end{aligned}$$

Now substitute our estimate for $\left\|\mathbf{\Sigma}_2^{1/2}\mathbf{\Omega}_2\mathbf{\Omega}_1^\dagger\right\|_{\mathrm{F}}^2$ to obtain the stated Frobenius norm bound. ■

Remark. As with previous bounds for uniform sampling, (*e.g.*, Kumar et al., 2012; Gittens, 2012), these results for uniform sampling are much weaker than our bounds from the previous subsections, since the sampling complexity depends on the coherence of the input matrix. When the matrix has small coherence, however, these bounds are similar to the bounds derived from the leverage-based sampling probabilities. Recall that, by the algorithm of Drineas et al. (2012), the coherence of an arbitrary input matrix can be computed in roughly the time it takes to perform a random projection on the input matrix.

5. Discussion and Conclusion

We have presented a unified approach to a large class of low-rank approximations of Laplacian and kernel matrices that arise in machine learning and data analysis applications. In doing so, we have provided qualitatively-improved worst-case theory and clarified the performance of these algorithms in practical settings. Our theoretical and empirical results suggest several obvious directions for future work.

In general, our empirical evaluation demonstrates that obtaining moderately high-quality low-rank approximations, as measured by minimizing the reconstruction error, depends in complicated ways on the spectral decay, the leverage score structure, the eigenvalue gaps in

relevant parts of the spectrum, etc. (Ironically, our empirical evaluation also demonstrates that *all* the sketches considered are reasonably-effective at approximating both sparse and dense, and both low-rank and high-rank matrices which arise in practice. That is, with only roughly $\mathrm{O}(k)$ measurements, the spectral, Frobenius, and trace approximation errors stay within a small multiplicative factor of around 3 of the optimal rank-k approximation errors. The reason for this is that matrices for which uniform sampling is least appropriate tend to be those which are least well-approximated by low-rank matrices, meaning that the residual error is much larger.) Thus, *e.g.*, depending on whether one is interested in ℓ being slightly larger or much larger than k, leverage-based sampling or a random projection might be most appropriate; and, more generally, an ensemble-based method that draws complementary strengths from each of these methods might be best.

In addition, we should note that, in situations where one is concerned with the quality of approximation of the actual eigenspaces, one desires both a small spectral norm error (because by the Davis–Kahan $\sin\Theta$ theorem and similar perturbation results, this would imply that the range space of the sketch effectively captures the top k-dimensional eigenspace of $\mathbf{A}$) as well as to use as few samples as possible (because one prefers to approximate the top k-dimension eigenspace of $\mathbf{A}$ with as close to a k-dimensional subspace as possible). Our results suggest that the leverage score probabilities supply the best sampling scheme for balancing these two competing objectives.

More generally, although our empirical evaluation consists of *random* sampling and *random* projection algorithms, our theoretical analysis clearly decouples the randomness in the algorithm from the structural heterogenities in the Euclidean vector space that are responsible for the poor performance of uniform sampling algorithms. Thus, if those structural conditions can be satisfied with a deterministic algorithm, an iterative algorithm, or any other method, then one can certify (after running the algorithm) that good approximation guarantees hold for particular input matrices in less time than is required for general matrices. Moreover, this structural decomposition suggests greedy heuristics—*e.g.*, greedily keep some number of columns according to approximate statistical leverage scores and "residualize." In our experience, a procedure of this form often performs quite well in practice, although theoretical guarantees tend to be much weaker; and thus we expect that, when coupled with our results, such procedures will perform quite well in practice in many medium-scale and large-scale machine learning applications.

Acknowledgments

AG would like to acknowledge the support, under the auspice of Joel Tropp, of ONR awards N00014-08-1-0883 and N00014-11-1-0025, AFOSR award FA9550-09-1-0643, and a Sloan Fellowship; and MM would like to acknowledge a grant from the Defense Advanced Research Projects Agency.

References

N. Arcolano and P. J. Wolfe. Nyström Approximation of Wishart Matrices. In *Proceedings of the 2010 IEEE International Conference on Acoustics Speech and Signal Processing*,

pages 3606–3609, 2010.

A. Asuncion and D. J. Newman. UCI Machine Learning Repository, November 2012. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

H. Avron, P. Maymounkov, and S. Toledo. Blendenpik: Supercharging LAPACK's Least-squares Solver. *SIAM Journal on Scientific Computing*, 32:1217–1236, 2010.

F. Bach. Sharp Analysis of Low-rank Kernel Matrix Approximations. In *Proceedings of the International Conference on Learning Theory (COLT)*, 2013.

F.R. Bach and M.I. Jordan. Predictive Low-rank Decomposition for Kernel Methods. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 33–40, 2005.

A. Banerjee, D. Dunson, and S. Tokdar. Efficient Gaussian Process Regression for Large Data Sets. *Biometrika*, 100:75–89, 2012.

M.-A. Belabbas and P. J. Wolfe. Fast Low-Rank Approximation for Covariance Matrices. In *Second IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pages 293–296, 2007a.

M.-A. Belabbas and P. J. Wolfe. On Sparse Representations of Linear Operators and the Approximation of Matrix Products. In *Proceedings of the 42nd Annual Conference on Information Sciences and Systems*, pages 258–263, 2008.

M.-A. Belabbas and P. J. Wolfe. On Landmark Selection and Sampling in High-dimensional Data Analysis. *Philosophical Transactions of the Royal Society, Series A*, 367:4295–4312, 2009a.

M.-A. Belabbas and P. J. Wolfe. Spectral Methods in Machine Learning and New Strategies for Very Large Datasets. *Proc. Natl. Acad. Sci. USA*, 106:369–374, 2009b.

M.-A. Belabbas and P.J. Wolfe. On the Approximation of Matrix Products and Positive Definite Matrices. Technical report, 2007b. Preprint: arXiv:0707.4448 (2007).

E. Bingham and H. Mannila. Random Projection in Dimensionality Reduction: Applications to Image and Text Data. In *Proceedings of the 7th Annual ACM SIGKDD Conference*, pages 245–250, 2001.

C. Boutsidis and A. Gittens. Improved Matrix Algorithms Via the Subsampled Randomized Hadamard Transform. *SIAM Journal of Matrix Analysis and Its Applications*, 34:1301–1340, 2013.

C. Boutsidis, M.W. Mahoney, and P. Drineas. An Improved Approximation Algorithm for the Column Subset Selection Problem. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 968–977, 2009.

J. Chiu and L. Demanet. Sublinear Randomized Algorithms for Skeleton Decompositions. *SIAM Journal on Matrix Analysis and Its Applications*, 34:1361–1383, 2013.

P. I. Corke. A Robotics Toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3:24–32, 1996.

C. Cortes, M. Mohri, and A. Talwalkar. On the Impact of Kernel Approximation on Learning Accuracy. In *Proceedings of the 13th International Workshop on Artificial Intelligence and Statistics*, 2010.

P. Drineas and M.W. Mahoney. On the Nyström Method for Approximating a Gram Matrix for Improved Kernel-Based Learning. *Journal of Machine Learning Research*, 6: 2153–2175, 2005.

P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo Algorithms for Matrices I: Approximating Matrix Multiplication. *SIAM Journal on Computing*, 36:132–157, 2006.

P. Drineas, M.W. Mahoney, and S. Muthukrishnan. Relative-error CUR Matrix Decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30:844–881, 2008.

P. Drineas, M.W. Mahoney, S. Muthukrishnan, and T. Sarlós. Faster Least Squares Approximation. *Numerische Mathematik*, 117(2):219–249, 2010.

P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff. Fast Approximation of Matrix Coherence and Statistical Leverage. *Journal of Machine Learning Research*, 13:3475–3506, 2012.

A. K. Farahat, A. Ghodsi, and M. S. Kamel. A Novel Greedy Algorithm for Nyström Approximation. In *Proceedings of the 14th International Workshop on Artificial Intelligence and Statistics*, 2011.

C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral Grouping Using the Nyström Method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

D. Fradkin and D. Madigan. Experiments with Random Projections for Machine Learning. In *Proceedings of the 9th Annual ACM SIGKDD Conference*, pages 517–522, 2003.

M. Genton. Classes of Kernels for Machine Learning: a Statistics Perspective. *J. Mach. Learn. Res.*, 2:299–312, 2002.

A. Gittens. The Spectral Norm Error of the Naïve Nyström Extension. Technical report, California Institute of Technology, 2012. Preprint: arXiv:1110.5305 (2011).

A. Gittens and M. W. Mahoney. Revisiting the Nyström Method for Improved Large-scale Machine Learning. Technical report, 2013. Tech Report: arXiv:1303.1849.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

A. M. Gustafson, E. S. Snitkin, S. C. J. Parker, C. DeLisi, and S. Kasif. Towards the Identification of Essential Genes Using Targeted Genome Sequencing and Comparative Analysis. *BMC Genomics*, 7:265, 2006.

I. Guyon, S. R. Gunn, A. Ben-Hur, and G. Dror. Result Analysis of the NIPS 2003 Feature Selection Challenge. In *Advances in Neural Information Processing Systems 17*. MIT Press, 2005.

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2):217–288, 2011.

D. Homrighausen and D. J. McDonald. Spectral Approximations in Machine Learning. Technical report, 2011. Preprint: arXiv:1107.4340 (2011).

R. Jin, T. Yang, M. Mahdavi, Y.-F. Li, and Z.-H. Zhou. Improved Bound for the Nyström's Method and its Application to Kernel Classification. *IEEE Information Theory*, 59: 6939–6949, 2013. Preprint: arXiv:1111.2262 (2011).

B. Klimt and Y. Yang. The Enron Corpus: a New Dataset for Email Classification Research. In *Proceedings of the 15th European Conference on Machine Learning*, pages 217–226, 2004.

S. Kumar, M. Mohri, and A. Talwalkar. On Sampling-based Approximate Spectral Decomposition. In *Proceedings of the 26th International Conference on Machine Learning*, pages 553–560, 2009a.

S. Kumar, M. Mohri, and A. Talwalkar. Ensemble Nyström Method. In *Annual Advances in Neural Information Processing Systems 22: Proceedings of the 2009 Conference*, 2009b.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling Techniques for the Nyström Method. In *Proceedings of the 12th Tenth International Workshop on Artificial Intelligence and Statistics*, pages 304–311, 2009c.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling Methods for the Nyström Method. *Journal of Machine Learning Research*, 13:981–1006, 2012.

J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph Evolution: Densification and Shrinking Diameters. *ACM Transactions on Knowledge Discovery from Data*, 1, 2007.

M. Li, J.T. Kwok, and B.-L. Lu. Making Large-Scale Nyström Approximation Possible. In *Proceedings of the 27th International Conference on Machine Learning*, pages 631–638, 2010.

S. Liu, J. Zhang, and K. Sun. Learning Low-rank Kernel Matrices with Column-based Methods. *Communications in Statistics—Simulation and Computation*, 39(7):1485–1498, 2010.

P. Ma, M. W. Mahoney, and B. Yu. A Statistical Perspective on Algorithmic Leveraging. In *Proceedings of the 31th International Conference on Machine Learning*, 2014.

P. Machart, T. Peel, S. Anthoine, L. Ralaivola, and H. Glotin. Stochastic Low-Rank Kernel Learning for Regression. In *Proceedings of the 28th International Conference on Machine Learning*, pages 969–976, 2011.

L. Mackey, A. Talwalkar, and M. I. Jordan. Divide-and-conquer Matrix Factorization. In *Annual Advances in Neural Information Processing Systems 24: Proceedings of the 2011 Conference*, 2011a.

L. Mackey, A. Talwalkar, and M. I. Jordan. Divide-and-conquer Matrix Factorization. In *NIPS*, 2011b. Preprint: arXiv:1107.0789 (2011).

M. W. Mahoney. *Randomized Algorithms for Matrices and Data*. Foundations and Trends in Machine Learning. NOW Publishers, Boston, 2011. Also available at: arXiv:1104.5557.

M. W. Mahoney. Algorithmic and Statistical Perspectives on Large-Scale Data Analysis. In U. Naumann and O. Schenk, editors, *Combinatorial Scientific Computing*, Chapman & Hall/CRC Computational Science. CRC Press, 2012.

M.W. Mahoney and P. Drineas. CUR Matrix Decompositions for Improved Data Analysis. *Proc. Natl. Acad. Sci. USA*, 106:697–702, 2009.

P.-G. Martinsson, V. Rokhlin, and M. Tygert. A Randomized Algorithm for the Decomposition of Matrices. *Applied and Computational Harmonic Analysis*, 30:47–68, 2011.

X. Meng, M. A. Saunders, and M. W. Mahoney. LSRN: a Parallel Iterative Solver for Strongly Over- or Under-Determined Systems. *SIAM Journal on Scientific Computing*, 36(2):C95–C118, 2014.

M. Mohri and A. Talwalkar. Can Matrix Coherence be Efficiently and Accurately Estimated? In *Proceedings of the 14th International Workshop on Artificial Intelligence and Statistics*, 2011.

T. O. Nielsen, R. B. West, S. C. Linn, O. Alter, M. A. Knowling, J. X. O'Connell, S. Zhu, M. Fero, G. Sherlock, J. R. Pollack, P. O. Brown, D. Botstein, and M. van de Rijn. Molecular Characterisation of Soft Tissue Tumours: a Gene Expression Study. *The Lancet*, 359:1301–1307, 2002.

P. Parker, P. J. Wolfe, and V. Tarok. A Signal Processing Application of Randomized Low-rank Approximations. In *Proceedings of the 13th IEEE Workshop on Statistical Signal Processing*, pages 345–350, 2005.

P. Paschou, E. Ziv, E.G. Burchard, S. Choudhry, W. Rodriguez-Cintron, M.W. Mahoney, and P. Drineas. PCA-Correlated SNPs for Structure Identification in Worldwide Human Populations. *PLoS Genetics*, 3:1672–1686, 2007.

V. Rokhlin, A. Szlam, and M. Tygert. A Randomized Algorithm for Principal Component Analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

D. N. Spendley and P. J. Wolfe. Adaptive Beamforming Using Fast Low-rank Covariance Matrix Approximations. In *Proceedings of the IEEE Radar Conference*, pages 1–5, 2008.

A. Talwalkar and A. Rostamizadeh. Matrix Coherence and the Nyström Method. In *Proceedings of the 26th Conference in Uncertainty in Artificial Intelligence*, 2010.

A. Talwalkar, S. Kumar, and H. Rowley. Large-scale Manifold Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

J. A. Tropp. Improved Analysis of the Subsampled Randomized Hadamard Transform. *Adv. Adapt. Data Anal.*, 3(1-2):115–126, 2011.

S. Venkatasubramanian and Q. Wang. The Johnson–Lindenstrauss Transform: An Empirical Study. In *ALENEX11: Workshop on Algorithms Engineering and Experimentation*, pages 164–173, 2011.

S. Wang and Z. Zhang. Improving CUR Matrix Decomposition and Nyström Approximation via Adaptive Sampling. *Journal of Machine Learning Research*, 14:2549–2589, 2013. Preprint: arXiv:1303.4207 (2013).

C.K.I. Williams and M. Seeger. Using the Nyström Method to Speed Up Kernel Machines. In *Annual Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, pages 682–688, 2001.

C.K.I. Williams, C.E. Rasmussen, A. Schwaighofer, and V. Tresp. Observations on the Nyström Method for Gaussian Process Prediction. Technical report, University of Edinburgh, 2002.

F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert. A Fast Randomized Algorithm for the Approximation of Matrices. *Applied and Computational Harmonic Analysis*, 25(3): 335–366, 2008.

C.-W. Yip, M. W. Mahoney, A. S. Szalay, I. Csabai, T. Budavári, R. F. G. Wyse, and L. Dobos. Objective Identification of Informative Wavelength Regions in Galaxy Spectra. *The Astronomical Journal*, 147(5):110, 2014.

K. Zhang and J. T. Kwok. Density-weighted Nyström Method for Computing Large Kernel Eigensystems. *Neural Computation*, 21(1):121–146, 2009.

K. Zhang and J. T. Kwok. Clustered Nyström Method for Large Scale Manifold Learning and Dimension Reduction. *IEEE Transactions on Neural Networks*, 21(10):1576–1587, 2010.

K. Zhang, I.W. Tsang, and J.T. Kwok. Improved Nyström Low-rank Approximation and Error Analysis. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1232–1239, 2008.

Journal of Machine Learning Research 17 (2016) 1-20 Submitted 8/14; Revised 7/15; Published 4/16

Improving Structure MCMC for Bayesian Networks through Markov Blanket Resampling

Chengwei Su CHENGWEI.SU.TH@DARTMOUTH.EDU
Mark E. Borsuk* MARK.BORSUK@DARTMOUTH.EDU
Thayer School of Engineering
Dartmouth College
Hanover, NH 03755-8000, USA
**Corresponding Author:*
Phone number: 1-603-646-9944
Fax number: 1-603-646-2277

Editor: Max Chickering

Abstract

Algorithms for inferring the structure of Bayesian networks from data have become an increasingly popular method for uncovering the direct and indirect influences among variables in complex systems. A Bayesian approach to structure learning uses posterior probabilities to quantify the strength with which the data and prior knowledge jointly support each possible graph feature. Existing Markov Chain Monte Carlo (MCMC) algorithms for estimating these posterior probabilities are slow in mixing and convergence, especially for large networks. We present a novel Markov blanket resampling (MBR) scheme that intermittently reconstructs the Markov blanket of nodes, thus allowing the sampler to more effectively traverse low-probability regions between local maxima. As we can derive the complementary forward and backward directions of the MBR proposal distribution, the Metropolis-Hastings algorithm can be used to account for any asymmetries in these proposals. Experiments across a range of network sizes show that the MBR scheme outperforms other state-of-the-art algorithms, both in terms of learning performance and convergence rate. In particular, MBR achieves better learning performance than the other algorithms when the number of observations is relatively small and faster convergence when the number of variables in the network is large.

Keywords: probabilistic graphical models, directed acyclic graph, Bayesian inference, Markov chain Monte Carlo

1. Introduction

A Bayesian network (BN) is a compact graphical representation of a multivariate joint probability distribution of variables. A BN is represented in the form of a directed acyclic graph (DAG), with nodes representing variables and directed edges representing probabilistic dependencies. An important feature of a BN is that each variable represented by a node is understood to be conditionally independent of the set of all its predecessors in the DAG, given the states of its parents (Neapolitan, 2004). In other words, the absence of a directly connecting edge between any two nodes implies that the two corresponding variables are independent given the states of the variables represented by intermediate nodes along a directed path. Thus, direct dependence between variables can be graphically distinguished from simple correlation mediated by other variables.

Algorithms have been developed to attempt to uncover the BN structures consistent with data from complex systems, with the goal of better understanding the direct and indirect mechanisms of action (Daly et al., 2011). This is particularly the case in systems biology, where diverse data on genes, gene expression, environmental exposure, and disease might be used to reveal the mechanistic links between genotype and phenotype (Su et al., 2013). Early attempts to learn BN structures from data focused on identifying the single best-fitting model. However, when the amount of data is relatively small compared to the number of variables, there are likely to be many different structures that fit the available data nearly equally well. The single best-fitting model is often an arbitrary result of the particular states of the variables that happen to have been observed. Further, presentation of this single best model provides no indication of the relative confidence one can have in its particular structural features (i.e., presence or absence of edges).

A Bayesian approach to structure learning allows for a quantification of the strength with which the data and any available prior knowledge jointly support each possible graph structure, in the form of posterior probabilities. Unfortunately, the number of possible DAGs grows super-exponentially with the number of variables being represented, making a full comparison of Bayesian posterior probabilities associated with alternative structures intractable. The Markov Chain Monte Carlo (MCMC) method as applied to graphical structures provides a numerical solution to this problem. In the original version of this algorithm, proposed by Madigan and York (1995), each transition in the Markov Chain consists of essentially a single edge change to the current graph. While this performs relatively well in small domains with 5-15 variables, it is rather slow in mixing and convergence with a larger number of variables. As the size of the structural search space grows, the chain is prone to getting trapped in local high probability regions separated by regions of lower probability regions. As a result, the samples obtained may not be representative of the true posterior distribution.

Friedman and Koller (2003) proposed a variation on the MCMC algorithm that explores the space of topological node orders rather than exact structures. The space of node orders is much smaller than the space of structures and is also likely to be smoother, allowing for better mixing. Thus, order MCMC is generally observed to converge more reliably to the same posterior probability estimates. More recently, Niinimki et al. (2012)introduced an algorithm based on partial, rather than linear, orders to create a still smaller and smoother posterior sampling space. Given a sample of (linear or partial) node orders, it is then straightforward to obtain a sample of graphs consistent with each order. However, as graphs can be consistent with more than one order, the effective prior probability over graphs involves marginalization over orders. This order-modular prior means that it is difficult to explicitly specify priors over graphs (Grzegorczyk and Husmeier, 2008) and that commonly-used order priors, such as the uniform, introduce bias by generating misrepresentative graph priors. Several variants of the order MCMC algorithm have been developed to address this prior bias (Koivisto and Sood, 2004; Eaton and Murphy, 2007; Ellis and Wong, 2008; Niinimki and Koivisto, 2013). Unfortunately, these approaches all incur substantial computational costs, making them impractical for learning the structure of networks with more than about 20 nodes. Recently Masegosa and Moral (2013) proposed a skeleton-based

approach, in which samples are restricted in the DAG space constrained by a given BN skeleton. This method scales to larger networks but performs poorly for small sample sizes.

In the present paper, we seek to improve the poor convergence of the structure MCMC algorithm without substantially increasing the computational costs. One way to do this is to introduce transitions that take larger steps in the structural search space, thus allowing the sampler to more effectively traverse low-probability regions when moving between local maxima and ridges (Koller and Friedman, 2009). With this intent, Grzegorczyk and Husmeier (2008) propose a new edge reversal move which selects an edge from the current MCMC iteration, reverses the direction of this edge, and then resamples new parents for the two nodes at either end of this edge. The idea is to implement a substantial modification to the current DAG that is customized to the new direction of the edge being reversed. The experimental results of Grzegorczyk and Husmeier (2008) indicate that this approach is an improvement over traditional structure MCMC with regard to convergence and mixing and is as computationally efficient as order MCMC.

We see an opportunity to take the approach of Grzegorczyk and Husmeier (2008) a step further by intermittently resampling the Markov blanket of randomly selected nodes. The Markov blanket of a node contains its parents, children, and the other parents of its children. These nodes comprise a module of local relations that shield a node from the rest of the network. These local relations are sometimes too sturdy to modify through small changes, causing the Markov Chain to linger in a region of the structural space for an unnecessarily long time. Our premise is that, by reconstructing the Markov blanket of a node, our search could more readily escape from local maxima, thus substantially improving the mixing of the Markov chain. For ease of comparison, we employ the same notation as Grzegorczyk and Husmeier (2008) and report similar diagnostic results.

2. BN Structure Learning Using MCMC

There are many ways to learn a BN from data. Since the size of search space of BNs is super exponential in the number of variables, it is impractical to do an exhaustive search. Many approaches therefore employ a score-based heuristic search algorithm, which starts with a random or seeded DAG and then proceeds by adding, reversing or deleting one edge at a time to obtain new DAGs. For each DAG, a score is calculated that indicates how well the DAG fits the data. The Bayesian approach to inference specifies that the consistency of a graph with data D is calculated as:

$$P(G \mid D) = \frac{P(D \mid G) \cdot P(G)}{\sum_{G^* \in \Omega} P(D \mid G^*) \cdot P(G^*)}, \tag{1}$$

where $P(G|D)$ is the posterior probability of a particular graph, $P(D|G)$ is the likelihood of the data given that graph and P(G) is the prior probability over the space of all possible graphs ($G \in \Omega$). Given the independence assumptions employed by a BN, and a noninformative graph prior (Grzegorczyk and Husmeier, 2008), the Bayesian score of a DAG, $P(G|D)$, can be factored into a product of local scores for each node X_n given its parents π_n as:

$$P(G \mid D) = \frac{1}{Z_N} \prod_{n=1}^{N} exp(\Psi[(X_n, \pi_n \mid D]), \tag{2}$$

where Z_N is a normalization term, N is the number of nodes, and $\Psi[X_n, \pi_n D]$ are the local scores defined by the probability model chosen for the likelihood $P(D|G)$. The multinomial distribution with a Dirichlet prior is often adopted, as it leads to a closed form expression for the local scores (Cooper and Herskovits, 1992).

The structure MCMC algorithm of Madigan and York (1995) seeks to generate a set of samples of possible structures with relative frequencies that correspond to the Bayesian posterior distribution. These samples can then be used, for example, to estimate the posterior probabilities of particular features of interest (marginalizing over the various structures). Implementations of the structure MCMC algorithm typically employ a Metropolis-Hastings sampler (Giudici and Castelo, 2003). At each step of the Markov chain, a candidate graph $G^{'}$ is proposed that will replace the current graph G with a probability equal to $\mu = min\{1, R(G^{'} \mid G)\}$ where

$$R(G^{'} \mid G) = \frac{P(G^{'} \mid D)}{P(G \mid D)} \cdot \frac{Q(G \mid G^{'})}{Q(G^{'} \mid G)}. \tag{3}$$

$Q(G^{'} \mid G)$ is the proposal probability for a move from G to $G^{'}$ and $Q(G|G')$ is the probability of a reverse move from $G^{'}$ to G. The ratio of these probabilities is referred to as the Hastings factor and serves to correct for any asymmetries that might be present in the proposal moves.

In the conventional structure MCMC algorithm, each step represents a move to a neighboring DAG, which is different in one edge from the current DAG. When Q is a uniform probability over the current structure's neighbors, the Hastings factor is equal to the ratio of number of neighbors for the reverse and forward moves, and the Markov chain is guaranteed to have a stationary distribution equal to the posterior distribution $P(G \mid D)$. Unfortunately, the space of DAGs is super exponential in the number of variables and the posterior landscape is characterized by ridges (representing equally well fitting graphs) separated by valleys. This makes it difficult for the chain to explore the space, making it slow to mix.

Grzegorczyk and Husmeier's (2008) 'new edge reversal'(REV) move replaces the simple single edge addition, removal, or reversal of Madigan and York's algorithm with a more substantial structural change. First, an edge in the current graph G is selected at random for reversal. The nodes at either end of this edge are then orphaned by removing all incoming edges. Next, a new parent set is sampled for each of these edges which contains the reversal of the originally selected edge and does not introduce any directed cycles to the graph. This then becomes the proposed graph $G^{'}$. Grzegorczyk and Husmeier specify the acceptance probability of the REV move, including the appropriate Hastings factor, thus guaranteeing convergence to the correct posterior distribution under ergodicity. As ergodicity is not assured, they intersperse their REV move with traditional structure MCMC moves in their final algorithm

3. Markov Blanket Resampling

Given the success of REV at improving mixing and convergence without added significant computational burden, we believe that there may be value to introducing more extreme steps into the structure MCMC algorithm. In particular, the traditional sampler may rarely leave regions of the structure space that have a similar posterior probability score. Graphs belonging to the same equivalence class, for example, comprise equiprobable ridges.

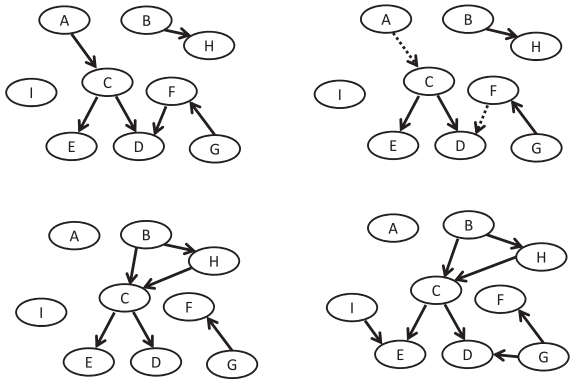


Figure 1: Illustration of the Markov Blanket Resampling (MBR) move. In this example, node C is selected as the target node (top left). Its Markov blanket includes nodes A, D, E, and F. First, all edges pointing to itself and to its children from other nodes are removed (top right). New nodes are next randomly selected as its new parents, in this case nodes B and H (bottom left). Finally, new additional parents are selected for its children, in this case G for D and I for E (bottom right).

Single edge reversals only lead to a new equivalence class when they involve a V-structure. REV improves on this by orphaning two nodes as an intermediate step, but in most cases the chain can easily revert to the same equivalence class within an iteration or two. Thus, even with REV, low probability valleys between ridges may not be readily traversed. As the Markov blanket of a node represents all the variables that can directly influence the state of that node, its entire reconstruction represents the most substantial change that can be made with respect to any single node. Thus, we believe that intermittent resampling of the Markov blanket could further improve the mixing and convergence of a Markov chain.

3.1 Overview

The idea of our Markov Blanket Resampling (MBR) move is shown in Figure 1. We start with the current DAG (upper left). First, a node is selected at random (node C, in this example). All edges pointing into this node are deleted, as well as edges pointing into its children from nodes other than itself (upper right, dotted edges), thus disconnecting much of the node's Markov blanket. New parents for this target node are then sampled disallowing any of its previous parents (bottom left), as are new additional parents, without any restrictions, for all the target node's children (bottom right), both under the condition of not creating any cycles.

3.2 Mathematical Details

Step 1: A node X_i is selected from a DAG, G, according to a uniform distribution over all nodes N in G. Orphaning this node and removing edges pointing into its children from nodes other than X_i creates a new DAG, G_0. (If X_i has no children, then only Step 2 is implemented and the products involving X_j in eq. (9) are set equal to 1.)
Step 2: A new parent set $\pi_i^{'}$ is selected for X_i, disallowing its previous parents π_i, to obtain new DAG, $G_0^{X_i \leftarrow \pi_i^{'}}$. Mathematically, therefore, $\pi_i^{'}$ conforms to the following distribution:

$$\Gamma(\pi_i^{'} \mid G_0, \pi_i) = \frac{exp(\Psi[X_i, \pi_i^{'} \mid D]) \cdot H(G_0^{X_i \leftarrow \pi_i^{'}}) \cdot I(\pi_i^{'}, \pi_i)}{Z^*(X_i \mid G_0, \pi_i)}, \tag{4}$$

where $H(G) = 0$ if the graph G contains a directed cycle and 1 otherwise, $I(\pi_i^{'}, \pi) = 0$ if $\pi \cap \pi_i^{'} \neq \phi$ and 1 otherwise, and $Z^*(X_i|G_0)$ represents the sum of local scores over all allowable parent sets π of X_i

$$Z^*(X_i \mid G_0, \pi_i) := \sum_{\substack{\pi: H(G_0^{X_i \leftarrow \pi}) = 1 \\ \pi_i \cap \pi_i^{'} \neq \phi}} exp(\Psi[X_i, \pi \mid D]). \tag{5}$$

This yields graph G_1.
Step 3: New parent sets $\pi^{'j}_i$ are sampled for each of X_i's children $X_i^j \in \gamma_i = \{X_i^1, ..., X_i^J\}$ that include node X_i and conform to the following distribution:

$$\Gamma(\pi_i^{'} \mid G_j) = \frac{exp(\Psi[X_i^j, \pi^{'j}_i \mid D]) \cdot H(G_j^{X_i^j \leftarrow \pi^{'j}_i}) \cdot J(\pi^{'j}_i, X_i)}{Z(X_i^j \mid G_j, X_i)}, \tag{6}$$

where G_j is the graph that exists prior to sampling new parents for node X_i^j, $J(\pi^{'j}_i, X_i) = 1$ if $X_i \in \pi^{'j}_i$ and 0 otherwise, and $Z(X_i^j|G_j, X_i)$ represents the sum of local scores over all allowable parent sets of X_i^j which contain X_i:

$$Z(X_i^j \mid G_j, X_i) := \sum_{\substack{\pi: H(G_j^{X_i^j \leftarrow \pi}) = 1 \\ X_i \in \pi}} exp(\Psi[X_i^j, \pi \mid D]). \tag{7}$$

It is important to note that sampling of new parent sets for each X_i^j occurs in a specified order that is randomized for each MBR move (see Appendix), so that if another child of X_i (i.e., a 'sibling' of X_i^j) becomes a parent of X_i^j in graph G_{j+1}, then X_i^j is not an allowable parent of that sibling when its new parents are subsequently sampled. This can be considered as a special case of disallowing directed cycles when sampling new parent sets for each X_i^j.

Thus, after these three steps, the DAG $G^{'}$ has been proposed by the MBR move, with overall proposal probability $Q(G^{'} \mid G)$ given by:

$$Q(G^{'} \mid G) = \frac{1}{N} \cdot \frac{exp(\Psi[X_i, \pi_i^{'} \mid D]) \cdot H(G_0^{X_i \leftarrow \pi_i^{'}}) \cdot I(\pi_i^{'}, \pi_i)}{Z^*(X_i \mid G_0, \pi_i)} \cdot \prod_{j=1}^{J} \frac{exp(\Psi[X_i^j, \pi^{'j}_i \mid D]) \cdot H(G_j^{X_i^j \leftarrow \pi^{'j}_i}) \cdot J(\pi^{'j}_i, Xi)}{Z(X_i^j \mid G_j, X_i)}. \tag{8}$$

To compute the acceptance probability of the Metropolis-Hastings algorithm, the probability of performing the complementary reverse move from G' to G, $Q(G \mid G')$, must be articulated. This involves another MBR move for node X_i in which the node is first orphaned and all edges pointing into its children from nodes other than itself are removed to yield DAG, $G_0^{'}$. In step 2, the original parent set π of X_i is selected resulting in graph $G_1^{'}$, while in step 3 the original parent sets of each of X_i's children are selected in graphs $G_j^{'}$ (accounting for node order) thus getting back to the original DAG G.

The numerators of the proposal distributions are cancelled out by the corresponding terms of the posterior ratio (Grzegorczyk and Husmeier 2008), as are all local scores corresponding to unaffected nodes. Therefore, the critical term in the acceptance probability can be written simply as:

$$R(G^{'} \mid G) = \frac{P(G^{'} \mid D)}{P(G \mid D)} \cdot \frac{Q(G \mid G^{'})}{Q(G^{'} \mid G)} = \frac{Z^*(X_i \mid G_0, \pi_i)}{Z^*(X_i \mid G_0^{'}, \pi_i^{'})} \cdot \frac{\prod_{j=1}^{J} Z(X_i^j \mid G_j, X_i)}{\prod_{j=1}^{J} Z(X_i^j \mid G_j^{'}, X_i)}. \tag{9}$$

An outline of the MBR algorithm is given in the Appendix.

3.3 Implementation Details

As with the order MCMC of Friedman and Koller (2003) and REV-enhanced structure MCMC of Grzegorczyk and Husmeier (2008), computational efficiency can be obtained when implementing the MBR algorithm by pre-computing and storing the scores associated with all potential parent sets of each node. Further efficiency can be gained for large networks by only considering as potential parents of each node those which have the highest local scores when considered as sole parents. However, as this is an approximate technique, we do not employ it in the examples presented here. We do, however, employ a fan-in restriction in which a maximum of three parents are allowed for any one node. When we sample for new parents for a node, we test all combinations of at most three. This approximation is used in most applications to reduce computational complexity and improve convergence.

To preserve acyclicity when sampling new parent sets for a node, we can simply eliminate invalid parent sets from the list of potential parent sets. In other words, if a parent set contains a node that is a descendant of the target node or its children, this parent set is removed from the current list of candidate parent sets. This method can also be used to define the scores used in computing the function Z^*. Finally, as the DAGs resulting from the first steps of both the forward and reverse MBR moves, G_0 and $G_0^{'}$, are the same, the identification of descendants only has to occur once during each MBR iteration.

In our applications we implement the MBR move probabilistically in the MCMC algorithm, with a fixed probability of p_m=1/15. The traditional structure move consisting of

the addition, deletion or reversal of a single edge is thus performed with probability p_s=1 p_m. These are the same probabilities as suggested by Grzegorczyk and Husmeier (2008) for their REV move. We experimented with different probabilities but found that 1/15 yields desirable results in terms of prediction quality and convergence rate. More frequent implementations lead to a decline in efficiency in sampling the highest probability regions as the chain seems to jump too often to low probability regions in the structure space, while a lower frequency reduces the mixing of the chain in efficiency incurred by using MBR. Since both the REV and MBR moves scores are pre-computed, MBR does not incur extra running time relative to REV, a fact we confirmed by through time trials using our selected data sets.

4. Experimental Testing

In this section, we test a version of the structure MCMC sampler enhanced by our MBR move against the standard structure sampler and one enhanced by the REV move of Grzegorczyk and Husmeier (2008). We find that the MBR-structure sampler performs better than the other two in terms of reliable convergence and learning performance, especially for relatively small sample sizes. We also find that it converges much faster for large and very large networks. These conclusions are drawn using data simulated from four models, all with more than 30 nodes: ALARM, a BN designed for patient monitoring consisting of 37 nodes and 46 edges (Beinlich et al., 1989); Hailfinder, a BN with 56 nodes and 66 edges developed to forecast severe weather (Abramson et al., 1996) HEPAR II, a BN built for diagnosis of liver disorders consisting of 70 nodes and 123 edges (Onisko, 2003), and GENS2, a tool for simulating genetic epidemiological datasets containing 100 exposure, gene, and disease variables (Pinelli et al., 2012). The ALARM data were used by Grzegorczyk and Husmeier (2008) to demonstrate that the REV-structure sampler performed much better than the standard MCMC sampler and equally well as the order sampler of Friedman and Koller (2003). We employ the other three datasets to compare performance for even larger networks.

Our algorithms were implemented in Matlab on a 128 GB machine with an Intel Xeon 2.00GHz processor. For the ALARM network with a simulated data set of m=1,000, precomputation of scores required 1017.6 sec, and implementation of 10,000 MCMC iterations required 382.6 sec for the standard structure algorithm, 401.5 sec for REV, and 407.3 sec for MBR. Given the 1/15 frequency of implementation, these times imply that the REV and MBR moves require approximately twice as much computation time as the conventional structure moves. This is consistent with the fact that the average number of children per node being considered in each iteration of MBR for ALARM was found to be 1.60.

4.1 Convergence Reliability

To compare the reliability of convergence of the three structure samplers, we generated two MCMC runs using datasets of various sizes (m=50, 100, 250, 500, 1000) simulated from ALARM: one initialized with an empty DAG and another initialized with a DAG identified as highest scoring through an independent run of the MCMC algorithm. We then compared the estimates of the posterior probabilities of the directed edges obtained from the two runs

using scatter plots. Points lying along the 1:1 line imply that the two runs agree, indicating reliable convergence.

We used the same run settings as Grzegorczyk and Husmeier (2008): for the standard structure sampler the burn-in was set to 500,000 iterations and 1000 DAGs were then collected by storing every 1000th iteration. For both the REV and MBR moves, the burn-in length was set to 312,500 and 1000 DAGS were collected by storing every 625th iteration. This difference was employed to ensure approximately equal computational costs across the three samplers, under the conservative assumption that the computational costs of the REV and MBR moves are 10-times greater than those of standard structure iterations and that the probability of these moves in any iteration was 1/15 (Grzegorczyk and Husmeier, 2008).

Scatter plots of the posterior probabilities of directed edges estimated from the two independent runs show that the MBR and REV structure samplers converge approximately equally reliably, even for small sample sizes (Figure 2). There is a high correspondence between the posterior probability estimates for the differently initialized runs. Under the traditional structure sampler, however, the posterior probabilities of some edge features depend on the initialization as indicated by off-diagonal points, even for very large samples, implying that this algorithm tends to get stuck at local maxima. Differences across algorithms also imply imperfect convergence.

4.2 Convergence Rate

Having established that, when added to the traditional MCMC structure sampler, the MBR move improves convergence reliability to the same degree as the REV move, we next compared the convergence rate for networks over a range of sizes: ALARM (37 nodes), HEPAR II (70 nodes), GENS2 (100 nodes). From each network, we generated sets of 1000 simulated observations and applied each of the MCMC algorithms described in the previous section, starting with an empty DAG. Trace plots reveal the speed and reliability of convergence.

For all three networks, the MBR-enhanced algorithm improves the convergence rate substantially over the REV-enhanced version (Figure 3). The highest scoring structures are found in approximately half the number of iterations of REV and in one tenth the number of the traditional structure sampler. In fact, it appears that both the REV and the traditional sampler may be getting stuck at local maxima for the largest networks, while the MBR sampler quickly converges.

4.3 Learning Performance

When the true graph network underlying a dataset is known, we can evaluate the performance of a structure learning algorithm using the concept of AUROC values. If we take the directed edges that have an estimated posterior probability greater than some threshold ϵ to be positive assertions of the existence of that feature, then a comparison against the true network will yield the number of true positive (TP), false positive (FP) and false negative (FN) assertions corresponding to that value of ϵ. The corresponding sensitivity, or true positive rate, TPR=TP/(TP+FN) and the complement of the specificity, or false positive rate, FPR=FP/(TN+FP) can then be calculated, and a plot of these two metrics for varying values of ϵ yields the Receiver Operator Characteristic (ROC) curve. Integrating the ROC

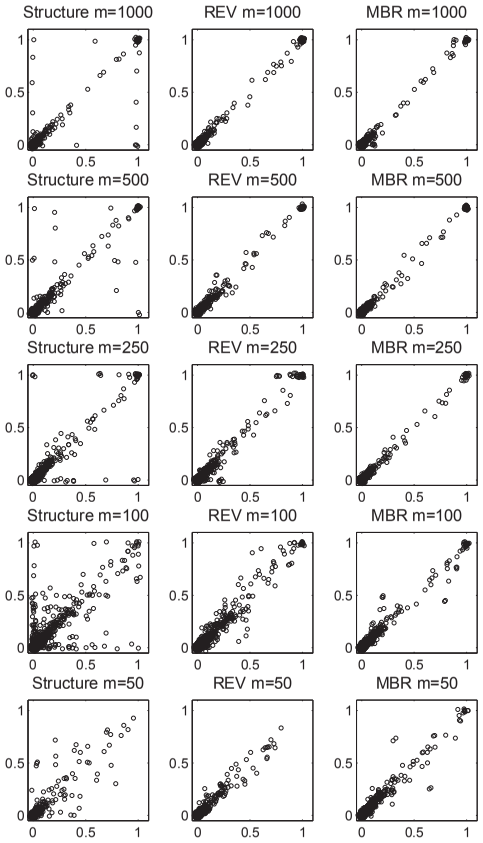


Figure 2: Scatter plots of posterior probability estimates for directed edges in the ALARM network, inferred from simulated datasets of various sizes. In each plot, the x-axis represents the posterior probabilities estimated from an MCMC run initialized with an empty DAG, and the y-axis represents the probability estimates obtained from an MCMC initialized with a high scoring DAG found from an independent run of the MCMC algorithm. When points lie along the diagonal, the two runs have results that agree, indicating reliable convergence. Values clustering near 0 and 1 for larger sample sizes indicate a reduction in inference uncertainty. Left column: traditional structure sampler; center column: REV-enhanced sampler; right column: MBR-enhanced sampler.

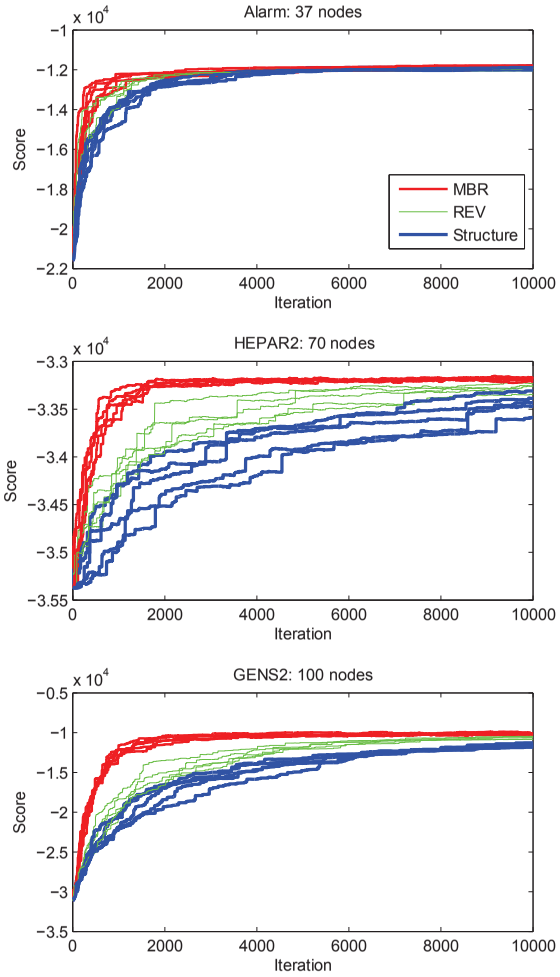


Figure 3: Trace plots of the first 10,000 iterations of five MCMC runs from each algorithm (MBR, REV, Structure) on data sets simulated from networks ranging in size. All runs involved 1000 simulated observations.

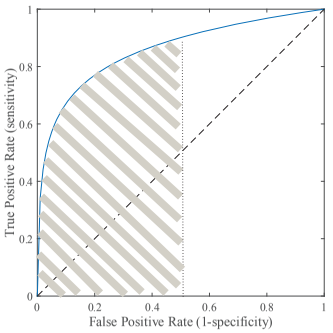


Figure 4: Hypothetical ROC curve. The hatched area represents the $AUROC_{0.5}$ corresponding to an upper limit of ϵ=0.5.

curve from 0 to an upper value of FPR=ϵ then gives the Area under the Receiver Operator Characteristic ($AUROC_\epsilon$) curve as a summary of learning performance, with high values indicating better performance (Figure 4). $AUROC_\epsilon$ values corresponding to low levels of the upper limit are of particular interest, as we are generally concerned with limiting the false positive rate.

For the ALARM network, we calculated $AUROC_\epsilon$ values for graphs learned from datasets of size m = 50, 100, 250, 500, and 1000 observations using each of the three MCMC algorithms and each of the two initializations, as described above. $AUROC_\epsilon$ values were calculated for ϵ = 1.0, 0.1, 0.05, and 0.01. Plots of these values (Figure 5) show that, as expected, learning performance tends to increase with the number of observations. The three algorithms perform nearly comparably at sample sizes of m=250 and greater, with the structure MCMC being somewhat less consistent between initializations, supporting the results shown in Fig. 2. At smaller sample sizes and lower values of ϵ, the traditional structure sampler performs poorly, the REV-enhanced version performs somewhat better, and the MBR-enhanced version performs the best of the three algorithms. Additionally, performance differs very little between the two independent initializations.

To assess learning performance on a different and somewhat larger network, we made comparable calculations with Hailfinder, using simulated datasets of size m = 500, 750, 1000, 1250, and 1500 observations (Figure 6). The traditional structure MCMC again shows increased learning performance with increasing number of observations. Yet, while the REV and MBR-enhanced versions show comparable performance with large datasets, they both yield even better performance with small datasets, as measured by $AUROC_1$, $AUROC_{0.01}$, and $AUROC_{0.05}$ values. This occurs because the Hailfinder network was originally constructed based, in part, on expert judgment. Therefore, the original network is not necessarily the best scoring one. For small sample sizes, the algorithms each find networks that fit the data better (i.e., more concisely) than the original network. Others working with similar-sized networks employing expert judgment or hidden nodes have found similar

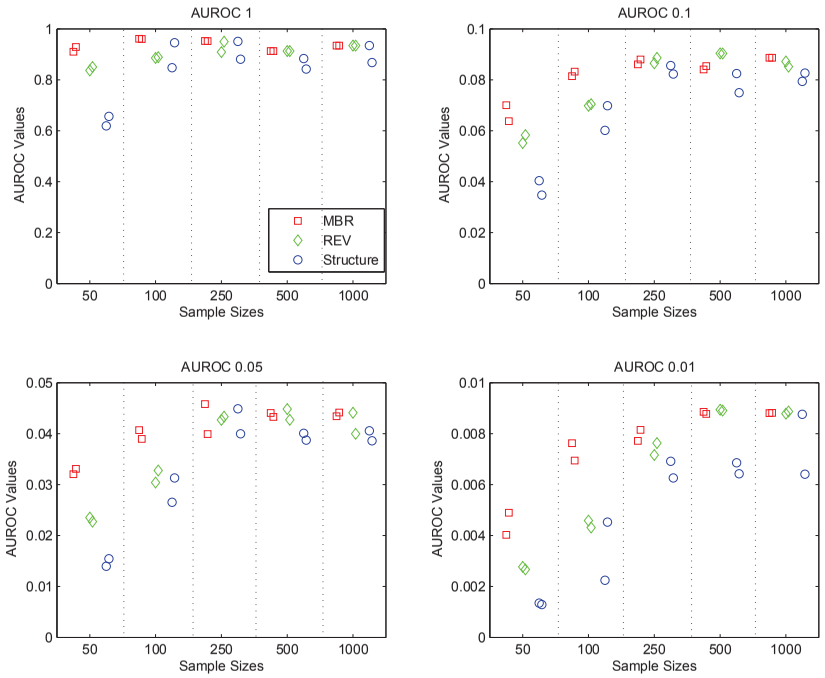


Figure 5: Learning performance of the three MCMC samplers as applied to simulated data from the ALARM network. Each panel shows $AUROC_\epsilon$ values corresponding to a different upper limit on the inverse specificity ($\epsilon = 1, 0.1, 0.05, 0.01$). Within each panel, results are shown for each sampler with a different symbol, as applied to varying numbers of simulated observations (m = 50, 100, 250, 500, 1000) across the x-axis. For each sampler, there are two values, corresponding to runs initialized with an empty DAG (slightly left) and a high scoring DAG found from an independent run of the MCMC algorithm (slightly right).

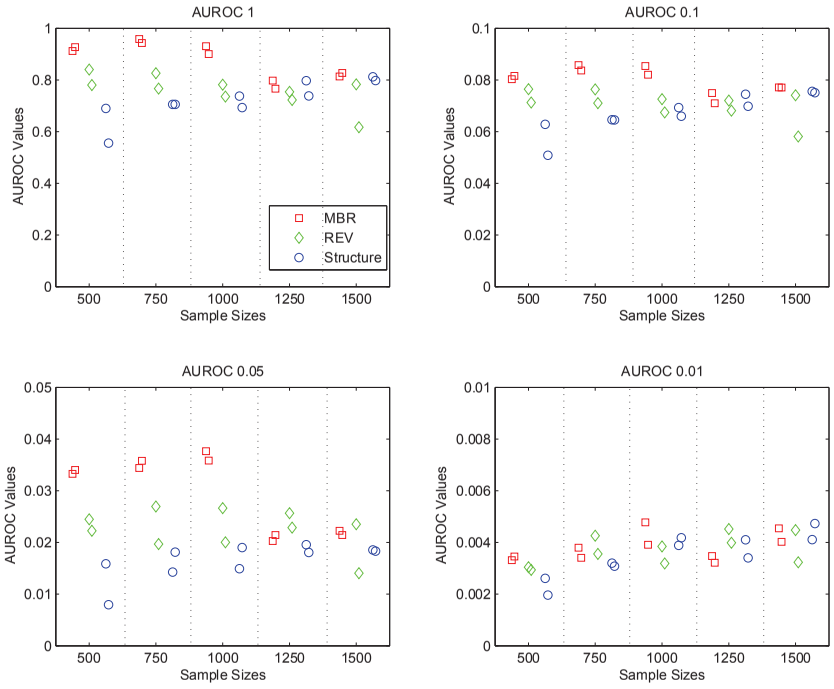


Figure 6: Learning performance of the three MCMC samplers as applied to simulated data from the Hailfinder network. Each panel shows $AUROC_\epsilon$ values corresponding to a different upper limit on the inverse specificity ($\epsilon = 1, 0.1, 0.05, 0.01$). Within each panel, results are shown for each sampler with a different symbol, as applied to varying numbers of simulated observations (m = 50, 100, 250, 500, 1000) across the x-axis. For each sampler, there are two values, corresponding to runs initialized with an empty DAG (slightly left) and a high scoring DAG found from an independent run of the MCMC algorithm (slightly right).

AUROC	1	0.1	0.05	0.01
Test Statistic	3.52	7.71	6.09	7.78
P-value	0.0065	2.96e-05	1.80e-04	2.74e-05

Table 1: Paired t-test results comparing MBR and REV as applied to the Alarm Data

behavior (Masegosa and Moral, 2013). The MBR version appears to perform somewhat better than the REV version at the smallest sample sizes with the possible exception of $\epsilon = 0.01$.

To rigorously compare the learning performance of the three algorithms, we employed a statistical hypothesis testing procedure. First, we produced 10 independent data samples from the Alarm network and generated 10 corresponding MCMC runs for each algorithm starting with an empty structure. $AUROC_\epsilon$ values were calculated from these 10 runs for each algorithm. We then performed paired-sample t-tests in Matlab with null hypothesis that the differences in $AUROC_\epsilon$ values between each pair of algorithms have mean equal to zero (assuming a normal distribution with unknown variance), with a two-sided alternative hypothesis. A paired test was used because we were able to calculate $AUROC_\epsilon$ values for all three algorithms using the same random data sample. Our hypothesis test was implemented for a data sample size of 100.

Results (Figure 7) show that MBR gives consistently greater $AUROC_\epsilon$ values across all four values of and that the within algorithm variance is relatively small compared to the between-algorithm variance, suggesting that the differences are statistically significant. Indeed, results shown in Table 1 indicate that our MBR algorithm statistically outperforms the other two algorithms for the Alarm data of 100 observations. Q-Q plots (not shown) confirm the approximate normality of the $AUROC_\epsilon$ value differences.

5. Conclusions

We have presented a novel MCMC sampling scheme to improve the mixing and convergence of the traditional MCMC algorithm used for probabilistically inferring the structure of BNs from observational data. The idea is to occasionally introduce major moves in the structural search space, thus allowing the sampler to more effectively traverse low-probability regions between local maxima and ridges. Our moves are more extreme than the new edge reversal (REV) move of Grzegorczyk and Husmeier (2008) in that we propose resampling much of the Markov blanket of nodes. As the Markov blanket contains all the variables that can directly influence the state of a node, its resampling represents a substantial change. Yet, as we can derive the complementary forward and backward moves of MBR, the Metropolis-Hastings algorithm can be used to account for any asymmetries that might be present in these proposal moves.

Our experiments across a range of network sizes show that the Markov Blanket Resampling (MBR) scheme outperforms the state-of-the-art new edge reversal (REV) scheme of Grzegorczyk and Husmeier (2008), both in terms of learning performance and convergence rate. In particular, MBR achieves better learning performance than the other algorithms when the number of observations is relatively small and faster convergence when the number

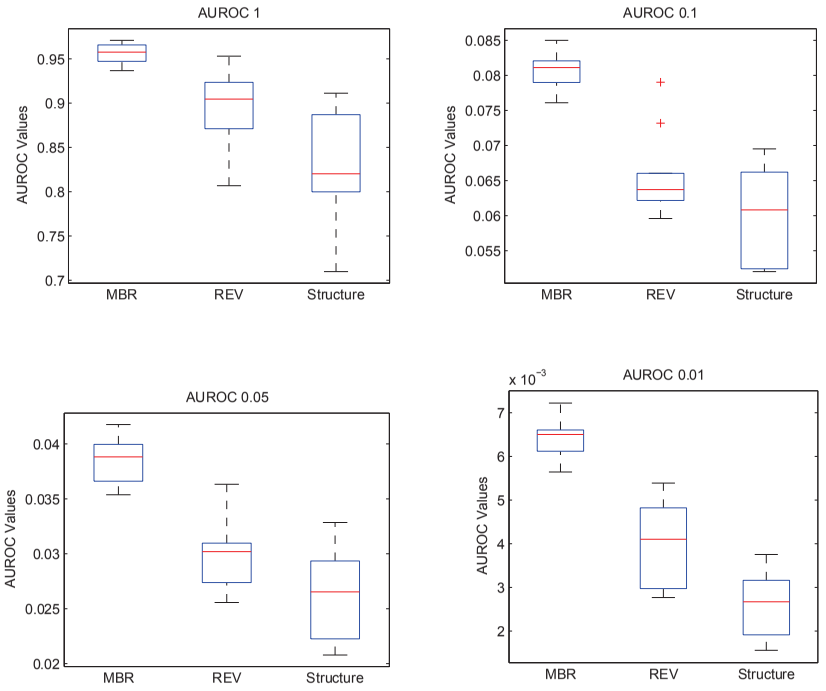


Figure 7: Boxplots of the $AUROC_\epsilon$ values from the three MCMC algorithms as applied to 10 independent sets of simulated data of sample size 100 from the Alarm network. Each panel shows values of $AUROC_\epsilon$ corresponding to a different upper limit on the inverse specificity ($\epsilon = 1, 0.1, 0.05, 0.01$). Boxes indicate the middle 50% (interquartile range, IQR) of the $AUROC_\epsilon$ values for each algorithm, central lines indicate median values, vertical whiskers extend out to the furthest value within 1.5*IQR of the boxes, and crosses indicate outlying values.

of variables in the network is large. As many problems in systems biology are characterized by a large number of variables and a relative paucity of observations (e.g. genome wide association studies), we believe our MBR algorithm will be especially useful in this field. Further, there typically exists prior knowledge (on gene-gene and gene-disease interactions for example) and this prior knowledge can be readily used to improve the performance of the structure-based MCMC algorithms (Imoto et al., 2004; Gao and Wang, 2011; Su et al., 2014), including our MBR scheme. This is not necessarily the case for other algorithms developed to address poor mixing and convergence, such as the order sampler of Friedman and Koller (2003), as discussed in the introduction. Thus, we believe a promising avenue of further research will be to improve methods for constructing informative structure priors from published data, experimental results, and expert opinion.

Acknowledgments

Research supported by grants from the National Center for Research Resources (5P20RR024474-02) and the National Institute of General Medical Sciences (8 P20 GM103534-02) from the National Institutes of Health. We thank the four anonymous reviewers for their thoughtful reviews and constructive suggestions.

Appendix A. Markov Blanket Resampling (MBR) Algorithm

For all domain variables $\{X_1, ..., X_N\}$, pre-compute and store lists of scores of all valid parent sets. We constrain the maximum number of parents a node can have to three in all our tests.

Given the current DAG, perform an MBR move with probability p_m, otherwise perform a traditional single-edge move. If an MBR move is to be performed, then proceed as follows:

- Randomly select a node X_i in current graph G, with probability N^{-1}.
- Store the current parent set π_i of node X_i.
- Delete all edges pointing into X_i and into its children X_i^j with the exception of X_i itself, thus obtaining G_0.
- Find and store the set $D(X_i|G_0)$ of all of X_i's descendant nodes in the current graph G_0.
- In the list of pre-computed scores of possible parent sets of node X_i, mark those that contain a node from the set $D(X_i|G_0)$. Also mark those that contain its parents $_i$ in G_0.
- Sample a new parent set $\pi_i^{'}$ from the unmarked parents sets of X_i according to equation (4). Store the sum of all unmarked scores as the value of $Z^*(X_i|G_0, \pi_i)$.
- Add the new parent set $\pi_i^{'}$ to G_0 to obtain DAG G_1.
- For j from 1 to J (with J being the number of X_i's children), in a specified order that is randomized for each MBR move:

  - Find and store the set $D(X_i^j|G_j)$ of all of X_i^j's descendant nodes in the current graph G_j.
  - In the list of pre-computed scores of possible parent sets of node X_i^j, mark those that contain a node from the set $D(X_i^j|G_j)$. Also mark those that do not contain X_i.
  - Sample a new parent set $\pi_i^{'j}$ from the unmarked parents sets of X_i^j according to equation (6).
  - Add the new parent set $\pi_i^{'j}$ to G_j to obtain G_{j+1}
- The final graph is the DAG $G^{'}$ being proposed by the MBR move.
- To compute the terms of the complementary inverse move required for calculating the acceptance probability, proceed as follows:
  - Calculate the joint probability of again selecting X_i as N^{-1}.
  - Store the current parent set π_i of node X_i in G'.
  - Delete all edges in G' pointing into X_i and into its children X_j with the exception of X_i itself, thus obtaining $G_0^{'}$.
  - In the list of pre-computed scores of possible parent sets of X_i, mark those that contain a node in $D(X_i|G_0)$ (which is equivalent to $D(X_i|G_0^{'})$, but has already been computed). Also mark those that contain its parents $\pi^{'}\_i$ in $G^{'}$. Store the sum of the unmarked scores as the value of $Z^*(X_i|G_0^{'}, \pi_i^{'})$.
  - Adding the original parent set π_i to $G_0^{'}$ yields DAG $G_1^{'}$ (which is equivalent to G_1).
  - For j from 1 to J:
    - Find and store the set $D(X_i^j|G_j^{'})$ of all of X_i^j's descendant nodes in the current graph $G_j^{'}$.
    - In the list of pre-computed scores of possible parent sets of node X_i^j, mark those that contain a node from the set $D(X_i^j|G_j^{'})$. Also mark those that do not contain X_i.
    - Store the sum of the unmarked scores as $Z(X_i^j|G_j^{'}, X_i)$.
    - Adding the original parent set π_i^j to $G_j^{'}$ yields G'_{j+1}.
- Calculate the overall acceptance probability of the move from G to $G^{'}$ according to equation (9) using the stored values.
- If the move is accepted replace the current DAG G with $G^{'}$, otherwise keep the current DAG.

References

B. Abramson, J. Brown, W. Edwards, A. Murphy, and R. L. Winkler. A Bayesian system for forecasting severe weather. *International Journal of Forecasting*, 12(1):57–71, 1996.

I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The Alarm monitoring system: A case study with two probabilistic inference techniques for belief networks. *Lecture Notes in Medical Informatics*, 38:247–256, 1989.

G. F. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

R. Daly, Q. Shen, and S.Aitken. Learning Bayesian networks: Approaches and issues. *The Knowledge Engineering Review*, 26(2):99–157, 2011.

D. Eaton and K. Murphy. Bayesian structure learning using dynamic programming and MCMC. *Proceedings of the Twenty-Third Conference on Uncertainty in Artifical Intelligence*, 2007.

B. Ellis and W. H. Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103(482):778–789, 2008.

N. Friedman and D. Koller. Being Bayesian about network structure. a Bayesian approach to structure discovery in bayesian networks. *Journal of the American Statistical Association*, 50(1-2):95–125, 2003.

S. Gao and X. Wang. Quantitative utilization of prior biological knowledge in the Bayesian network modeling of gene expression data. *BMC Bioinformatics*, 12(1):359, 2011.

P. Giudici and R. Castelo. Improving Markov Chain Monte Carlo model search for data mining. *Machine Learning*, 50(1-2):127–158, 2003.

M. Grzegorczyk and D. Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2-3):265–305, 2008.

S. Imoto, T. Higuchi, T. Goto, K. Tashiro, S. Kuhara, and S. Miyano. Combining microarrays and biological knowledge for estimating gene networks via Bayesian networks. *Journal of Bioinformatics and Computational Biology*, 2(1):77–98, 2004.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *The Journal of Machine Learning Research*, 5:549–573, 2004.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

D. Madigan and J. York. Bayesian graphical models for discrete data. international statistical review. *Revue Internationale de Statistique*, pages 215–232, 1995.

A. R. Masegosa and S. Moral. New skeleton-based approaches for Bayesian structure learning of Bayesian networks. *Applied Soft Computing*, 13(2):1110–1120, 2013.

R. E. Neapolitan. *Learning Bayesian Networks*. Pearson Prentice Hall Upper Saddle River, 2004.

T. Niinimki and M. Koivisto. Annealed importance sampling for structure learning in Bayesian networks. *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

T. Niinimki, P. Parviainen, and M. Koivisto. Partial order MCMC for structure discovery in Bayesian networks. *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, 2012.

A. Onisko. *Probabilistic Causal Models in Medicine: Application to Diagnosis in Liver Disorders*. 2003.

M. Pinelli, G. Scala, R. Amato, S. Cocozza, and G. Miele. Simulating gene-gene and gene-environment interactions in complex diseases: Gene-Environment Interaction Simulator 2. *BMC Bioinformatics*, 13(1):132, 2012.

C. Su, A. Andrew, M. R. Karagas, and M. E. Borsuk. Using Bayesian networks to discover relations between genes, environment, and disease. *BioData Mining*, 6(1):6, 2013.

C. Su, A. Andrew, M. R. Karagas, and M. E. Borsuk. Incorporating prior expert knowledge in learning Bayesian networks from genetic epidemiological data. *Computational Intelligence in Bioinformatics and Computational Biology*, 2014.

Journal of Machine Learning Research 17 (2016) 1-34 Submitted 9/14; Published 4/16

Volumetric Spanners: An Efficient Exploration Basis for Learning

Elad Hazan EHAZAN@CS.PRINCETON.EDU
Princeton University, Department of Computer Science
Princeton, NJ 08540, USA

Zohar Karnin ZKARNIN@YMAIL.COM
Yahoo Haifa Labs
Matam, Haifa 31905, Israel

Editor: Alexander Rakhlin

Abstract

Numerous learning problems that contain exploration, such as experiment design, multi-arm bandits, online routing, search result aggregation and many more, have been studied extensively in isolation. In this paper we consider a generic and efficiently computable method for action space exploration based on convex geometry.

We define a novel geometric notion of an exploration mechanism with low variance called volumetric spanners, and give efficient algorithms to construct such spanners. We describe applications of this mechanism to the problem of optimal experiment design and the general framework for decision making under uncertainty of bandit linear optimization. For the latter we give efficient and near-optimal regret algorithm over general convex sets. Previously such results were known only for specific convex sets, or under special conditions such as the existence of an efficient self-concordant barrier for the underlying set. [1]

Keywords: barycentric spanner, volumetric spanner, linear bandits, hard margin linear regression

1. Introduction

A fundamental challenge in machine learning is environment exploration. A prominent example is the famed multi-armed bandit (MAB) problem, in which a decision maker iteratively chooses an action from a set of available actions and receives a payoff, without observing the payoff of all other actions she could have taken. The MAB problem displays an exploration-exploitation tradeoff, in which the decision maker trades exploring the action space vs. exploiting the knowledge already obtained to pick the best arm. The exploration challenge arises in structured bandit problems such as online routing and rank aggregation: how to choose the most informative path in a graph, or the most informative ranking?

Another example in which environment exploration is crucial is experiment design, and more generally the setting of active learning. In this setting it is important to correctly identify the most informative experiments/queries so as to efficiently construct a solution.

1. This paper is the full version of a merger of two extended abstracts (Hazan et al., 2014) and (Karnin and Hazan, 2014).

Exploration is hardly summarized by picking an action uniformly at random. Indeed, sophisticated techniques from various areas of optimization, statistics and convex geometry have been applied to designing ever better exploration algorithms. To mention a few: (Awerbuch and Kleinberg, 2008) devise the notion of *barycentric spanners*, and use this construction to give the first low-regret algorithms for complex decision problems such as online routing. (Abernethy et al., 2012) use self-concordant barriers to build an efficient exploration strategy for convex sets in Euclidean space. (Bubeck et al., 2012) apply tools from convex geometry, namely the John ellipsoid to construct optimal-regret algorithms for bandit linear optimization, albeit not always efficiently.

In this paper we consider a generic approach to exploration, and quantify what efficient exploration with *low variance* requires in general. Given a set in Euclidean space, a low-variance exploration basis is a subset with the following property: given noisy estimates of a linear function over the basis, one can construct an estimate for the linear function over the entire set without increasing the variance of the estimates.

By definition, such low variance exploration bases are immediately applicable to noisy linear regression: given a low-variance exploration basis, it suffices to learn the function values only over the basis in order to interpolate the value of the underlying linear regressor over the entire decision set. This fact can be used for active learning as well as for the exploration component of a bandit linear optimization algorithm.

Henceforth we define a novel construction for a low variance exploration basis called **volumetric spanners** and give efficient algorithms to construct them. We further investigate the convex geometry implications of our construction, and define the notion of a **minimal volumetric ellipsoid** of a convex body. We give structural theorems on the existence and properties of these ellipsoids, as well as constructive algorithms to compute them in several cases.

We complement our findings with two applications to machine learning. The first application is to the problem of experiment design, in which we give an efficient algorithm for hard-margin active linear regression with optimal bounds. Next, we advance a well-studied open problem that has exploration as its core difficulty: an efficient and near-optimal regret algorithm for bandit linear optimization (BLO). We expect that volumetric spanners and volumetric ellipsoids can be useful elsewhere in experiment design and active learning.

1.1 Informal statement of results

Experiment design: In the statistical field called *optimal design of experiments*, or just *optimal design* (Atkinson and Donev, 1992; Wu, 1978) , a statistician is faced with the task of choosing experiments to perform from a given pool, with the goal of producing the optimal result within the budget constraint.

Formally, consider a pool of possible experiments denoted $x_1, ..., x_n \in \mathbb{R}^d$. The goal of the designer is to choose a distribution over the pool of experiments, such that experiments chosen according to this distribution produce a hypothesis $\hat{\mathbf{w}}$ that is as close as possible to the true linear function behind the data. The distance between the hypothesis and true linear function can be measured in different ways, each corresponding to a different *optimality criteria*. The common property of the criteria is that they all minimize the variance of the hypothesis. Since the variance is not a scalar but a $d \times d$ matrix, the different

criteria differ by the fact that each one minimizes a different function $\Phi : \mathcal{R}^{d\times d} \to \mathcal{R}$ over the covariance matrix. Common criteria are the A-, D-, and E-optimality criteria. D-optimality, minimizes the determinant of the covariance matrix, and thus minimizes the volume of the confidence region. In A-optimality the trace of the covariance matrix, i.e., the total variance of the parameter estimates, is minimized. E-optimality minimizes the maximum eigenvalue of the covariance matrix, and thus minimizes the size of the major axis of the confidence region.

The above criteria do not directly characterize the quality of predictions on test data. A common criterion that directly takes the test data into account is that of G-optimality. Here the goal is to minimize the maximum variance of the predicted values. In other words, by denoting $\mathrm{Var}_S(x_i)$ the variance of the prediction of x_i after querying the points of S, the goal in G-optimality is to minimize $\max_i \mathrm{Var}_S(x_i)$. G-optimality and D-optimality are closely related in the sense that an exact solution to one is the solution to the other, see for example (Spruill and Studden, 1979).

In this paper we solve a problem closely related to the G-optimality criteria. Given a pool of data points $\mathcal{K} \in \mathcal{R}^d$, say representing a pool of patients, we aim to solve an active regression problem finding w.h.p a regressor minimizing the *worst-case error*, while minimizing the number of (noisy) queries to the regressor. The formal definition of the problem is given in Section 6. This problem differs from classic linear regression results as there, the *mean square error* is bounded. The difference between the described problem and solving the optimal design problem with the G-optimality criteria is two-fold. First, we do not aim to minimize the variance but to obtain a high probability bound. Second, in optimal design the quality of the distribution is measured when the budget tends to infinity. Specifically, notice that for a distribution over the possible experiments, rather than a deterministic subset of them, the corresponding covariance matrix is random. The discussed minimizations are done over the expected covariance matrix, where the expectation is taken over the subset of chosen experiments. When the budget tends to infinity the actual covariance matrix is close w.h.p to its expected counterpart. We call this the *infinite budget setting*. Ours is a finite budget setting where one does not aim to provide a distribution over the possible experiments but a deterministic subset of them of a fixed size.

For the finite budget setting various relaxations have been considered in the statistical literature, usually without an approximation guarantee. Our method differs from previous works of this spirit by: First, we do not impose a hard-budget constraint of experiments, but rather bound the number of experiments as a function of the desired approximation guarantee. Second, we obtain a computationally efficient algorithm with provable optimality results. Finally, as an added bonus our solution has the property of choosing very few data points to explore, potentially much less than the budget. A motivating example for this property is the medical experiment design. Here a data point is a human subject and it is more realistic to have few volunteers being thoroughly tested on as opposed to performing few tests over many volunteers. Our setting is arguably more natural for the medical-patient-experiment motivating example; in general there are numerous examples where the budget of experiments is not fixed but rather the tolerable error. Of equal importance is the fact that our setting allows to derive efficient algorithms with rigorous theoretical guarantees.

A related and recently popular model is called random design (Hsu et al., 2012; Audibert and Catoni, 2010; Györfi et al., 2006; Audibert and Catoni, 2011). In this setting the designer is given a set of measurements $\{x_i, y_i | i \in [n]\}$ for $x_i \in \mathbb{R}^d$ drawn from an unknown distribution $\mathcal{D}$. The goal is to predict as well as the best linear predictor measured according to the mean square error, i.e., minimize

$$\mathop{\mathbf{E}}_{(x,y)\in\mathcal{D}} \left[(x^\top w - y)^2 - (x^\top w^* - y)^2 \right]$$

where w^* is the optimal linear regressor. Various other performance metrics have been considered in the referenced papers, i.e., measuring the norm of the regressor vs. the optimal regressor in a norm proportional to the covariance matrix. However, in this setting an expected error is the criterion vs. our criterion of worst-case, or a high confidence bound on the error[2], which is more suitable for some experiment design settings.

Active learning: The most well-studied setting in active learning is pool-based active learning (McCallum and Nigam, 1998), in which the learner has access to a pool of examples, and can iteratively query labels of particular examples of her choice. Compared to passive learning, in which labelled examples are drawn from a fixed unknown distribution, known active learning algorithms can attain a certain generalization error guarantee albeit observing exponentially fewer labelled examples, e.g., (Cohn et al., 1994; Dasgupta et al., 2009; Hanneke, 2007; Balcan et al., 2009), under certain assumptions such as special hypothesis classes, realizability or large-margin. Active learning with noise is a much less studied topic: (Balcan et al., 2009) give an exponential improvement over passive learning of linear threshold functions, but under the condition that the noise is smaller than the desired accuracy. Real valued active learning with a soft-margin criteria was addressed in (Ganti and Gray, 2012). The reader is referred to (Dasgupta and Langford, 2009) for a more detailed survey of active learning literature.

Bandit Linear Optimization Bandit linear optimization (BLO) is a fundamental problem in decision making under uncertainty that efficiently captures structured action sets. The canonical example is that of online routing in graphs: a decision maker iteratively chooses a path in a given graph from source to destination, the adversary chooses lengths of the edges of the graph, and the decision maker receives as feedback the length of the path she chose but no other information (Awerbuch and Kleinberg, 2008). Her goal over many iterations is to attain an average travel time as short as that of the best fixed shortest path in the graph.

This decision problem is readily modeled in the "experts" framework, albeit with efficiency issues: the number of possible paths is potentially exponential in the graph representation. The BLO framework gives an efficient model for capturing such structured decision problems: iteratively a decision maker chooses a point in a convex set and receives as a payoff an adversarially chosen linear cost function. In the particular case of online routing, the decision set is taken to be the s-t-flow polytope, which captures the convex hull of all source-destination shortest paths in a given graph, and has a succinct representation

2. Our results though stated as a worst case error can be generalized to a high probability solution in the random design scenario.

with polynomially many constraints and low dimensionality. The linear cost function corresponds to a weight function on the graphs edges, where the length of a path is defined as the sum of weights of its edges.

The BLO framework captures many other structured problems efficiently, e.g., learning permutations, rankings and other examples (Abernethy et al., 2012). As such, it has been the focus of much research in the past few years. The reader is referred to the recent survey of (Bubeck and Cesa-Bianchi, 2012a) for more details on algorithmic results for BLO. Let us remark that certain online bandit problems do not immediately fall into the convex BLO model that we address, such as combinatorial bandits studied in (Cesa-Bianchi and Lugosi, 2012).

In this paper we contribute to the large literature on the BLO model by giving the first polynomial-time and near optimal-regret algorithm for BLO over general convex decision sets; see Section 7 for a formal statement. Previously efficient algorithms, with non-optimal-regret, were known over convex sets that admit an efficient self-concordant barrier (Abernethy et al., 2012), and optimal-regret algorithms were known over general sets (Bubeck et al., 2012) but these algorithms were not computationally efficient. Our result, based on volumetric spanners, is able to attain the best of both worlds.

1.2 Volumetric Ellipsoids and Spanners

We now describe the main convex geometric concepts we introduce and use for low variance exploration. To do so we first review some basic notions from convex geometry.

Let $\mathbb{R}^d$ be the d-dimensional vector space over the reals. Given a set of vectors $S = \{v_1, \ldots, v_t\} \subset \mathbb{R}^d$, we denote by $\mathcal{E}(S)$ the ellipsoid defined by S:

$$\mathcal{E}(S) = \left\{ \sum_{i \in S} \alpha_i v_i \ : \ \sum_i \alpha_i^2 \leq 1 \right\}.$$

By abuse of notation, we also say that $\mathcal{E}(S)$ is *supported* on the set S.

Ellipsoids play an important role in convex geometry and specific ellipsoids associated with a convex body have been used in previous works in machine learning for designing good exploration bases for convex sets $\mathcal{K} \subseteq \mathbb{R}^d$. For example, the notion of *barycentric spanners* which were introduced in the seminal work of Awerbuch and Kleinberg (Awerbuch and Kleinberg, 2008) corresponds to looking at the ellipsoid of maximum volume supported by exactly d points from [3] $\mathcal{K}$. Barycentric Spanners have since been used as an exploration basis in several works: Online bandit linear optimization (Dani et al., 2007), A high probability counterpart of online bandit linear optimization (Bartlett et al., 2008), repeated decision making of approximable functions (Kakade et al., 2009) and a stochastic version of bandit linear optimization (Dani et al., 2008). Another example is the work of Bubeck et al. (Bubeck et al., 2012) which looks at the minimum volume enclosing ellipsoid (MVEE) also known as the John ellipsoid (see Section 2 for more background on this fundamental object from convex geometry).

As will be clear soon, our definition of a *minimal volumetric ellipsoid* enjoys the best properties of the examples above enabling us to get more efficient algorithms. Similar to

3. While the definition of (Awerbuch and Kleinberg, 2008) is not phrased as such, their analysis shows the existence of barycentric spanners by looking at the maximum volume ellipsoid.

barycentric spanners, it is supported by a small (linear) set of points of $\mathcal{K}$. Simultaneously and unlike the barycentric counterpart, the volumetric ellipsoid contains the body $\mathcal{K}$, a property shared with the John ellipsoid.

Definition 1 (Volumetric Ellipsoids) *Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a set in Euclidean space. For $S \subseteq \mathcal{K}$, we say that $\mathcal{E}(S)$ is a* volumetric ellipsoid *for $\mathcal{K}$ if it contains $\mathcal{K}$. We say that $\mathcal{E}(S)$ is a* minimal volumetric ellipsoid *if it is a containing ellipsoid defined by a set of minimal cardinality*

$$S \in \arg\min_{S' \in \mathcal{S}(\mathcal{K})} |S'|, \qquad \mathcal{S}(\mathcal{K}) = \left\{ S' \mid S' \subseteq \mathcal{K} \subseteq \mathcal{E}(S') \right\}$$

We say that $|S|$ is the order *of the minimal volumetric ellipsoid or of* [4] *$\mathcal{K}$ denoted* $\mathbf{order}(\mathcal{K})$.

We discuss various geometric properties of volumetric ellipsoids later. For now, we focus on their utility in designing efficient exploration bases. To make this concrete and to simplify some terminology later on (and also to draw an analogy to barycentric spanners), we introduce the notion of *volumetric spanners*. Informally, these correspond to sets S that span all points in a given set with coefficients having Euclidean norm at most one. Formally:

Definition 2 *Let $\mathcal{K} \subseteq \mathbb{R}^d$ and let $S \subseteq \mathcal{K}$. We say that S is a* volumetric spanner *for $\mathcal{K}$ if $\mathcal{K} \subseteq \mathcal{E}(S)$.*

Clearly, a set $\mathcal{K}$ has a volumetric spanner of cardinality t if and only if $\mathbf{order}(\mathcal{K}) \leq t$.

Our goal in this work is to bound the order of arbitrary sets. A priori, it is not even clear if there is a universal bound (depending only on the dimension and not on the set) on the order S for compact sets $\mathcal{K}$. However, barycentric spanners and the John ellipsoid show that the order of any compact set in $\mathcal{R}^d$ is at most $O(d^2)$. Our main structural result in convex geometry gives a nearly optimal linear bound on the order of all sets.

Theorem 3 (Main) *Any compact set $\mathcal{K} \subseteq \mathbb{R}^d$ admits a volumetric ellipsoid of order at most $12d$. Further, if $\mathcal{K} = \{v_1, \ldots, v_n\}$ is a discrete set, then a volumetric ellipsoid for $\mathcal{K}$ of order at most $12d$ can be constructed in $\mathrm{poly}(n, d)$ time.*

This structural result is achieved by sparsifying (via the methods given in Batson et al. 2012) the John contact points of $\mathcal{K}$. We emphasize the last part of the above theorem giving an algorithm for finding volumetric spanners of small size; this could be useful in using our structural results for algorithmic purposes. We also give a different algorithmic construction for the discrete case (a set of n vectors) in Section 5. While being sub-optimal by logarithmic factors (gives an ellipsoid of order $O(d(\log d)(\log n))$ this alternate construction has the advantage of being simpler and more efficient to compute.

4. We note that our definition allows for multi-sets, meaning that S may contain the same vector more than once.

1.3 Approximate Volumetric Spanners

Theorem 3 shows the existence of good volumetric spanners and also gives an efficient algorithm for finding such a spanner in the discrete case, i.e., when $\mathcal{K}$ is finite and given explicitly. However, the existence proof uses the John ellipsoid in a fundamental way and it is not known how to compute (even approximately) the John ellipsoid efficiently for the case of general convex bodies. For such computationally difficult cases, we introduce a natural relaxation of volumetric ellipsoids which can be computed efficiently for a bigger class of bodies and is similarly useful. The relaxation comes from requiring that the ellipsoid of small support contain all but an ε fraction of the points in $\mathcal{K}$ (under some distribution). In addition, we also require that the measure of points decays exponentially fast w.r.t their $\mathcal{E}(S)$-norm (see precise definition in next section); this property gives us tighter control on the set of points not contained in the ellipsoid. When discussing a measure over the points of a body the most natural one is the uniform distribution over the body. However, it makes sense to consider other measures as well and our approximation results in fact hold for a wide class of distributions.

Definition 4 *Let $S \subseteq \mathcal{R}^d$ be a set of vectors and let V be the matrix whose columns are the vectors of S. We define the semi-norm*

$$\|x\|_{\mathcal{E}(S)} = \sqrt{x^\top (VV^\top)^{-1} x}\ ,$$

where $(VV^\top)^{-1}$ is the Moore-Penrose pseudo-inverse of $VV^\top$. Let $\mathcal{K}$ be a convex set in $\mathcal{R}^d$, p a distribution over it, and let $\varepsilon > 0$. A (p,ε)-exp-volumetric spanner *of $\mathcal{K}$ is a set $S \subseteq \mathcal{K}$ such that for any $\theta > 1$*

$$\Pr_{x\sim p}[\|x\|_{\mathcal{E}(S)} \geq \theta] \leq \varepsilon^{-\theta}.$$

To understand the intuition behind the above definition, notice that for any point x in $\mathcal{E}(S)$ we have $\|x\|_{\mathcal{E}(S)} \leq 1$. Hence, if S is such that $\mathcal{K} \subseteq \mathcal{E}(S)$ we have that S is a *(p,ε)-exp-volumetric spanner* of $\mathcal{K}$ for $\varepsilon = 0$ and any p. It follows that the above provides an approximate volumetric spanner; in what follows we show that this particular type of approximation can be computed efficeintly for log-concave p and is sufficient in certain cases.

Theorem 5 *Let $\mathcal{K}$ be a convex set in $\mathcal{R}^d$ and p a log-concave distribution over it. By sampling $O(d+\log^2(1/\varepsilon))$ i.i.d. points from p one obtains, w.p. at least $1-\exp\left(-\sqrt{\max\{\log(1/\varepsilon), d\}}\right)$, a (p,ε)-exp-volumetric spanner for $\mathcal{K}$.*

1.4 Structure of the paper

In the next section we list the preliminaries and known results from measure concentration, convex geometry and online learning that we need. In Section 3 we show the existence of small size volumetric spanners. In Sections 4 and 5 we give efficient constructions of volumetric spanners for continuous and discrete sets, respectively. We describe the application to experiment design in Section 6. We then proceed to describe the application of our geometric results to bandit linear optimization in Section 7.

2. Preliminaries

We now describe several preliminary results we need from convex geometry and linear algebra. We start with some notation:

- A matrix $A \in \mathbb{R}^{d\times d}$ is positive semi-definite (PSD) when for all $x \in \mathcal{R}^d$ it holds that $x^\top Ax \geq 0$. Alternatively, when all of its eigenvalues are non-negative. We say that $A \succeq B$ if $A - B$ is PSD.
- Given an ellipsoid $\mathcal{E}(S) = \{\sum_i \alpha_i v_i : \sum_i \alpha_i^2 \leq 1\}$, we shall use the notation $\|x\|_{\mathcal{E}(S)} \triangleq \sqrt{x^\top (VV^\top)^{-1} x}$ to denote the (Minkowski) semi-norm defined by the ellipsoid, where V is the matrix with the vectors v_i's as columns. Notice that $\mathcal{E}(S)$ is symmetric and convex hence it defines a norm.
- Throughout, we denote by I_d the $d \times d$ identity matrix.

We next describe properties of the John ellipsoid which plays an important role in our proofs.

2.1 The Fritz John Ellipsoid

Let $\mathcal{K} \subseteq \mathbb{R}^n$ be an arbitrary convex body. Then, the **John ellipsoid** of $\mathcal{K}$ is the minimum volume ellipsoid containing $\mathcal{K}$. This ellipsoid is unique and its properties have been the subject of important study in convex geometry since the seminal work of John (John, 1948) (see Ball 1997 and Henk 2012 for historic information).

Suppose that we have linearly transformed $\mathcal{K}$ such that its minimum volume enclosing ellipsoid (MVEE) is the unit sphere; in convex geometric terms, $\mathcal{K}$ is in *John's position*. The celebrated decomposition theorem by (John, 1948) gives a characterization of when a body is in John's position and will play an important role in our constructions (the version here is from Ball 1997).

Below we consider only symmetric convex sets, i.e., sets that admit the following property: if $x \in \mathcal{K}$ then also $-x \in \mathcal{K}$. The sets encountered in machine learning applications are most always symmetric, since estimating a linear function on a point x is equivalent to estimating it on its polar $-x$, and negating the outcome.

Theorem 6 (Ball 1997) *Let $\mathcal{K} \in \mathcal{R}^d$ be a symmetric set such that its MVEE is the unit sphere. Then there exist $m \leq d(d+1)/2 - 1$* contact points *of $\mathcal{K}$ and the sphere $u_1, \ldots, u_m$ and non-negative weights $c_1, \ldots, c_m$ such that $\sum_i c_i u_i = 0$ and $\sum c_i u_i u_i^T = I_d$.*

The contact points of a convex body have been extensively studied in convex geometry and they also make for an appealing exploration basis in our context. Indeed, (Bubeck et al., 2012) use these contact points to attain an optimal-regret algorithm for BLO. Unfortunately we know of no efficient algorithm to compute, or even approximate, the John ellipsoid for a general convex set, thus the results by (Bubeck et al., 2012) do not yield efficient algorithms for BLO.

For our construction of volumetric spanners we need to compute the MVEE of a discrete symmetric set, which can be done efficiently. We make use of the following (folklore) result:

Theorem 7 (folklore, see e.g., Khachiyan 1996; Damla Ahipasaoglu et al. 2008) *Let $\mathcal{K} \subseteq \mathcal{R}^d$ be a set of n points. It is possible to compute an ε-approximate MVEE for $\mathcal{K}$ (an enclosing ellipsoid of volume at most $(1+\varepsilon)$ that of the MVEE) that is also supported on a subset of $\mathcal{K}$ in time $O(n^{3.5}\log\frac{1}{\varepsilon})$.*

The run-time above is attainable via the ellipsoid method or path-following interior point methods (see references in theorem statement). An approximation algorithm rather than an exact one is necessary in a real-valued computation model, and the logarithmic dependence on the approximation guarantee is as good as one can hope for in general.

The above theorem allows us to efficiently compute a linear transformation such that the MVEE of $\mathcal{K}$ is essentially the unit sphere. We can then use linear programming to compute an approximate decomposition like in John's theorem as follows.

Theorem 8 *Let $\{x_1,\ldots,x_n\} = \mathcal{K} \subseteq \mathcal{R}^d$ be a set of n points and assume that:*

1. *$\mathcal{K}$ is symmetric.*
2. *The John Ellipsoid of $\mathcal{K}$ is the unit sphere.*

Then it is possible, in $O((\sqrt{n}+d)n^3)$ time, to compute non-negative weights $c_1,\ldots,c_n$ such that (1) $\sum_i c_i x_i = 0$ and (2) $\sum_{i=1}^n c_i x_i x_i^\top = I_d$.

Proof

Denote the MVEE of $\mathcal{K}$ by $\mathcal{E}$ and let V be its corresponding $d\times d$ matrix, meaning V is such that $\|y\|_{\mathcal{E}}^2 = y^\top V^{-1} y \le 1$ for all $y\in\mathcal{K}$. By our assumptions $I_d = V$.

As $\mathcal{K}$ is symmetric and its MVEE is the unit ball, according to Theorem 6, there exist $m \le d(d+1)/2-1$ contact points $u_1,\ldots,u_m$ of $\mathcal{K}$ with the unit ball and a vector $c'\in\mathcal{R}^m$ such that $c'\ge 0$, $\sum c'_i = d$ and $\sum c'_i u_i u_i^T = I_d$. It follows that the following LP has a feasible solution: Find $c\in\mathcal{R}^n$ such that $c\ge 0$, $\sum c_i \le d$ and $\sum c_i u_i u_i^T = I_d$. The described LP has $O(n+d^2)$ constraints and n variables. It can thus be solved in time $O(d+\sqrt{n})n^3)$ via interior point methods. ■

We next state the results from probability theory that we need.

2.2 Distributions and Measure Concentration

For a set $\mathcal{K}$, let $x\sim\mathcal{K}$ denote a uniformly random vector from $\mathcal{K}$.

Definition 9 *A distribution over $\mathcal{R}^d$ is log-concave if its probability distribution function (pdf) p is such that for all $x,y\in\mathcal{R}^d$ and $\lambda\in[0,1]$,*

$$p(\lambda x + (1-\lambda)y) \ge p(x)^\lambda p(y)^{1-\lambda}$$

Two log-concave distributions of interest to us are (1) the uniform distribution over a convex body and (2) a distribution over a convex body where $p(x)\propto\exp(L^\top x)$, where L is some vector in $\mathcal{R}^d$. The following result shows that given oracle access to the pdf of a log-concave distribution we can sample from it efficiently. An oracle to a pdf accepts as input a point $x\in\mathcal{R}^d$ and returns the value $p(x)$.

Lemma 10 (Lovász and Vempala 2007, Theorems 2.1 and 2.2) *Let p be a log-concave distribution over $\mathbb{R}^d$ and let $\delta>0$. Then, given oracle access to p, i.e., and oracle computing its pdf for any point in $\mathbb{R}^d$, there is an algorithm which approximately samples from p such that:*

1. *The total variation distance between the produced distribution and the distribution defined by p is no more than δ. That is, the difference between the probabilities of any event in the produced and actual distribution is bounded by δ.*
2. *The algorithm requires a pre-processing time of $\tilde{O}(d^5)$.*
3. *After pre-processing, each sample can be produced in time $\tilde{O}(d^4/\delta^4)$, or amortized time of $\tilde{O}(d^3/\delta^4)$ if more than d samples are needed.*

Definition 11 (Isotropic position) *A random variable x is said to be in isotropic position*
(or isotropic) if

$$\mathbf{E}[x] = 0, \qquad \mathbf{E}[xx^\top] = I_d.$$

A set $\mathcal{K}\subseteq\mathbb{R}^d$ is said to be in isotropic position if $x\sim\mathcal{K}$ is isotropic. Similarly, a distribution p is isotropic if $x\sim p$ is isotropic.

Henceforth we use several results regarding the concentration of log-concave isotropic random vectors. In these results we use the matrix operator norm (i.e., spectral norm) defined as $\|A\| = \sup_{x\neq 0}\frac{\|Ax\|}{\|x\|}$. We use slight modification where the center of the distribution is not necessarily in the origin. For completeness we present the proof of the modified theorems in Appendix A

Theorem 12 (Theorem 4.1 in Adamczak et al. 2010) *Let p be a log-concave distribution over $\mathcal{R}^d$ in isotropic position. There is a constant C such that for all $t,\delta>0$, the following holds for $n = \frac{Ct^4 d\log^2(t/\delta)}{\delta^2}$. For independent random vectors $x_1,\ldots,x_n\sim p$, with probability at least $1-\exp(-t\sqrt{d})$,*

$$\left\|\frac{1}{n}\sum_{i=1}^n x_i x_i^\top - I_d\right\| \le \delta.$$

Corollary 13 *Let p be a log-concave distribution over $\mathcal{R}^d$ and $x\sim p$. Assume that x is such that $\mathbf{E}[xx^T] = I_d$. Then, there is a constant C such that for all $t\ge 1,\delta>0$, the following holds for $n = \frac{Ct^4 d\log^2(t/\delta)}{\delta^2}$. For independent random vectors $x_1,\ldots,x_n\sim p$, with probability at least $1-\exp(-t\sqrt{d})$,*

$$\left\|\frac{1}{n}\sum_{i=1}^n x_i x_i^\top - I_d\right\| \le \delta.$$

Theorem 14 (Theorem 1.1 in Guédon and Milman 2011) *There exist constants c,C such that the following holds. Let p be a log-concave distribution over $\mathcal{R}^d$ in isotropic position and let $x\sim p$. Then, for all $\theta\ge 0$,*

$$Pr\left[\left|\|x\| - \sqrt{d}\right| > \theta\sqrt{d}\right] \le C\exp(-c\sqrt{d}\cdot\min(\theta^3,\theta)).$$

Corollary 15 *Let p be a log-concave distribution over $\mathcal{R}^n$ and let $x \sim p$. Assume that $\mathbf{E}[xx^T] = I_d$. Then for some universal C, c it holds for any $\theta \geq 3$ that*

$$\Pr\left[\|x\| > \theta\sqrt{d}\right] \leq C\exp\left(-c\theta\sqrt{d}\right)$$

The following theorem provides a concentration bound for random vectors originating from an arbitrary distribution.

Theorem 16 (Rudelson 1999) *Let X be a vector-valued random variable over $\mathbb{R}^d$ with $\mathbf{E}[XX^\top] = \Sigma$ and $\|\Sigma^{-1/2}X\|^2 \leq R$. Then, for independent samples $X_1, \ldots, X_M$ from X, and $M \geq CR\log(R/\varepsilon)/\epsilon^2$ the following holds with probability at least $1/2$:*

$$\left\|\frac{1}{M}\sum_{i=1}^{M} X_i X_i^\top - \Sigma\right\| \leq \epsilon\|\Sigma\|.$$

Finally, we also make use of barycentric spanners in our application to BLO and we briefly describe them next.

2.3 Barycentric Spanners

Definition 17 (Awerbuch and Kleinberg 2008) *A barycentric spanner of $\mathcal{K} \subseteq \mathcal{R}^d$ is a set of d points $S = \{u_1, \ldots, u_d\} \subseteq \mathcal{K}$ such that any point in $\mathcal{K}$ may be expressed as a linear combination of the elements of S using coefficients in $[-1, 1]$. For $C > 1$, S is a C-approximate barycentric spanner of $\mathcal{K}$ if any point in $\mathcal{K}$ may be expressed as a linear combination of the elements of S using coefficients in $[-C, C]$*

In (Awerbuch and Kleinberg, 2008) it is shown that any compact set has a barycentric spanner. Moreover, they show that given an oracle with the ability to solve linear optimization problems over $\mathcal{K}$, an approximate barycentric spanner can be efficiently obtained. In the following sections we will use this constructive result.

Theorem 18 (Proposition 2.5 in Awerbuch and Kleinberg 2008) *Let $\mathcal{K}$ be a compact set in $\mathcal{R}^d$ that is not contained in any proper linear subspace. Given an oracle for optimizing linear functions over $\mathcal{K}$, for any $C > 1$, it is possible to compute a C-approximate barycentric spanner for $\mathcal{K}$, using $O(d^2 \log_C(d))$ calls to the optimization oracle.*

3. Existence of Volumetric Ellipsoids and Spanners

In this section we show the existence of low order volumetric ellipsoids proving our main structural result, Theorem 3. Before we do so, we first state a few simple properties of volumetric ellipsoids (recall the Definition of **order** from Definition 1):

- The definition of **order** is linear invariant: for any invertible linear transformation $T : \mathbb{R}^d \to \mathbb{R}^d$ and $K \subseteq \mathbb{R}^d$, $\mathbf{order}(\mathcal{K}) = \mathbf{order}(T\mathcal{K})$.

  Proof Let $S \subseteq \mathcal{K}$ be such that $\mathcal{K} \subseteq \mathcal{E}(S)$. Then, clearly $T\mathcal{K} \subseteq \mathcal{E}(TS)$. Thus, $\mathbf{order}(T\mathcal{K}) \leq \mathbf{order}(\mathcal{K})$. The same argument applied to T^{-1} and $T\mathcal{K}$ shows that

Figure 1: In $\mathbb{R}^2$ the order of the volumetric ellipsoid of the equilateral triangle centered at the origin is at least 3. If the vertices are $[0, 1], [-\frac{\sqrt{3}}{2}, -\frac{1}{2}], [\frac{\sqrt{3}}{2}, -\frac{1}{2}]$, then the eigenpoles of the ellipsoid of the bottom two vertices are $[0, \frac{2}{3}], [2, 0]$. The second figure shows one possibility for a volumetric ellipsoid by adding $\frac{3}{4}$ of the first vertex to the previous ellipsoid. This shows the ellipsoid to be non-unique, as it can be rotated three ways.

  $\mathbf{order}(\mathcal{K}) \leq \mathbf{order}(T\mathcal{K})$. ■

- The minimal volumetric ellipsoid is not unique in general; see example in Figure 1. Further, it is in general different from the John ellipsoid.

- For non-degenerate bodies $\mathcal{K}$, their order is naturally lower bounded by d, and there are examples in which it is strictly larger than d (e.g., Figure 1).

In the proof Theorem 3 we require a modification of a result by (Batson et al., 2012) providing a method to sparsity a distribution over vectors while approximately maintaining its covariance matrix. We show that this technique can be applied over the distribution derived from John's decomposition of $\mathcal{K}$ in order to obtain a small set from which we construct a volumetric spanner. We begin by presenting the result given in (Batson et al., 2012), then its modification, and then proceed to the proof of Theorem 3.

Theorem 19 (Theorem 3.1 of Batson et al. 2012) *Let $v_1, \ldots, v_m$ be vectors in $\mathcal{R}^d$ and let $c > 1$. Assume that $\sum_i v_i v_i^\top = I_d$. There exist scalars $s_i \geq 0$ for $i \in [m]$ with $|\{i | s_i > 0\}| \leq cd$ such that*

$$I_d \preceq \sum_i s_i v_i v_i^\top \preceq \frac{c + 1 + 2\sqrt{c}}{c + 1 - 2\sqrt{c}} I_d$$

Furthermore, these scalars can be found in time $O(cd^3m)$.

Lemma 20 *Let $u_1, \ldots, u_m$ be unit vectors and let $p \in \Delta(m)$ be a distribution over $[m]$ such that $d\sum_{i=1}^m p_i u_i u_i^\top = I_d$. Then, there exists a (possibly multi) set $S \subseteq \{u_1, \ldots, u_m\}$ such that $\sum_{v \in S} vv^\top \succeq I_d$ and $|S| \leq 12d$. Moreover, such a set can be computed in time $O(md^3)$.*

Proof Let $v_i = \sqrt{p_i}u_i$. We fix c as some constant whose value will be determined later. Clearly for these vectors v_i it holds that $d\sum_i v_i v_i^\top = I_d$. It follows from the above theorem that there exist some scalars $s_i \geq 0$ for which at most cd are non-zeros and

$$I_d \preceq d\sum_i s_i v_i v_i^\top \preceq \frac{c+1+2\sqrt{c}}{c+1-2\sqrt{c}} I_d \tag{1}$$

Our set S will be composed by taking each u_i $\lceil ds_i p_i \rceil$ many times. Plugging in equation (1) shows that indeed

$$\sum_{w\in S} ww^\top \succeq I_d$$

and it remains to bound the size of S i.e., $\sum \lceil ds_i p_i \rceil$. By taking the trace of the expression and dividing by d we get that

$$\sum_i s_i \text{Trace}(v_i v_i^\top) \preceq \frac{c+1+2\sqrt{c}}{c+1-2\sqrt{c}}$$

Plugging in the expressions for v_i along with u_i being unit vectors (hence $\text{Trace}(u_i u_i^\top) = 1$) lead to

$$\sum_i s_i p_i \preceq \frac{c+1+2\sqrt{c}}{c+1-2\sqrt{c}}$$

It follows that

$$\sum \lceil ds_i p_i \rceil \leq d\frac{c+1+2\sqrt{c}}{c+1-2\sqrt{c}} + |\{i|s_i \geq 0\}| \leq d\left(c + \frac{c+1+2\sqrt{c}}{c+1-2\sqrt{c}}\right)$$

By optimizing c we get $\sum \lceil ds_i p_i \rceil \leq 12d$, and the lemma is proved. ■

Proof of Theorem 3 Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a compact set. Without loss of generality assume that $\mathcal{K}$ is symmetric and contains the origin; we can do so as in the following we only look at outer products of the form $vv^\top$ for vectors $v \in \mathcal{K}$. Further, as $\textbf{order}(\mathcal{K})$ is invariant under linear transformations, we can compute, as detailed in Theorem 7 the MVEE[5] of $\mathcal{K}$ and transform the space into one where this ellipsoid is the Euclidean unit sphere. That is, move $\mathcal{K}$ into John's position, at $\text{poly}(n, d)$ time.

According to Theorem 8 it is then possible to compute a distribution p over $\mathcal{K}$ with

$$d\sum_{x\in\mathcal{K}} p_x xx^\top = I_d$$

Lemma 20 provides a way to compute a (multi-)set S of size at most $12d$ with

$$\sum_{v\in S} vv^\top \succeq I_d$$

5. Notice that the Theorem provides an approximation of $1+\varepsilon$ of the MVEE where the running time scales as $\log(1/\varepsilon)$. In what follows it is easy to see that the precision of all equalities is affected up to a multiplicative factor of $1 \pm \varepsilon$ by this issue. This eventually translates into a set of size $12(1+\varepsilon)d$ rather than $12d$. We omit these technical details for a more readable proof.

Since $\mathcal{K}$ is contained in the unit ball we get that S is a volumetric spanner for $\mathcal{K}$ as required. The total running time required to find s includes the computation of the transformation into John's position, John's decomposition, and its sparsification. The running time amounts to $O(n^{3.5} + dn^3 + nd^3)$. ■

4. Approximate Volumetric Spanners

In this section we provide a construction for (p, ε)-exp-volumetric spanner (as in Definition 4), proving Theorem 5. We start by providing a more technical definition of a spanner. Note that unlike previous definitions, the following is not impervious to linear operators and will only be used to aid our construction.

Definition 21 *A β-relative-spanner is a discrete subset $S \subseteq \mathcal{K}$ such that for all $x \in \mathcal{K}$, $\|x\|^2_{\mathcal{E}(S)} \leq \beta\|x\|^2$.*

A first step is a spectral characterization of relative spanners:

Lemma 22 *Let $S = \{v_1, ..., v_T\} \subseteq \mathcal{K}$ span $\mathcal{K}$ and be such that*

$$W = \sum_{i=1}^T v_i v_i^\top \succeq \frac{1}{\beta} I_d$$

Then S is a β-relative-spanner.

Proof Let $V \in \mathbb{R}^{d\times T}$ be a matrix whose columns are the vectors of S. As $VV^\top = W \succeq \frac{1}{\beta} I_d$ we have that

$$\beta I_d \succeq (VV^\top)^{-1}$$

It follows that

$$\|x\|_{\mathcal{E}(S)} = x^\top (VV^\top)^{-1} x \leq \beta\|x\|^2$$

as required. ■

Proof [Proof of Theorem 5] We analyze the algorithm of sampling i.i.d points according to p, previously defined within Theorem 5, assuming the vectors are sampled exactly according to the log-concave distribution. The result involving an approximate sample, which is necessary for implementing the algorithm in the general case, is an immediate application of Lemma 10 and Corollary 13.

Our analysis of the algorithm is for $T = C(d + \log^2(1/\varepsilon))$ samples, where C is some sufficiently large constant. Assume first that $\mathbf{E}_{x\sim p}[xx^\top] = I_d$. Let $W = \sum_{i=1}^T u_i u_i^\top$. Then, for $C > 0$ large enough, by Corollary 13, $\|\frac{1}{T}W - I_d\| \leq 1/2$ w.p. at least $1 - \exp(-\sqrt{d})$. Therefore, S spans $\mathbb{R}^d$ and

$$\frac{1}{T}W \succeq I_d - \frac{1}{2}I_d = \frac{1}{2}I_d$$

Thus according to Lemma 22, S is a $(2/T)$-relative spanner. Consider the case where $\Sigma = \mathbf{E}_{x\sim p}[xx^\top]$ is not necessarily the identity. By the above analysis we get that

$$\Sigma^{-1/2}S = \{\Sigma^{-1/2}u_1, \ldots, \Sigma^{-1/2}u_T\}$$

form a $(2/T)$-relative spanner for $\Sigma^{-1/2}\mathcal{K}$. This is since the r.v defined as $\Sigma^{-1/2}x$ where $x \sim p$ is log-concave. The latter along with Corollary 15 implies that for any $\theta \geq 1$,

$$\Pr_{x\sim p}\left[\|\Sigma^{-1/2}x\| \geq 3\theta\sqrt{d}\right] \leq c_1 \exp\left(-c_2\theta\sqrt{d}\right) \tag{2}$$

for some universal constants $c_1, c_2 > 0$. It follows that for our set S and any $\theta \geq 1$,

$$\begin{aligned}
\Pr_{x\sim p}\left[\|x\|_{\mathcal{E}(S)} > \theta\right] &= \Pr_{x\sim p}\left[\|\Sigma^{-1/2}x\|_{\mathcal{E}(\Sigma^{-1/2}S)} > \theta\right] && \|x\|_{\mathcal{E}(S)} = \\
&&& \|\Sigma^{-1/2}x\|_{\mathcal{E}(\Sigma^{-1/2}S)} \\
&\leq \Pr_{x\sim p}\left[\|\Sigma^{-1/2}x\| > \theta\sqrt{T/2}\right] && \Sigma^{-1/2}S \text{ is a} \\
&&& 2/T\text{-relative-spanner} \\
&= \Pr_{x\sim p}\left[\|\Sigma^{-1/2}x\| > 3\theta\sqrt{\tfrac{dC}{18}}\cdot\sqrt{1+\tfrac{\log^2(1/\varepsilon)}{d}}\right] && T = C(d+\log^2(1/\varepsilon)) \\
&\leq c_1\exp\left(-c_2\theta\sqrt{d}\sqrt{\tfrac{C}{18}}\cdot\sqrt{1+\tfrac{\log^2(1/\varepsilon)}{d}}\right) && \text{Equation (2), } C \geq 18 \\
&\leq \exp\left(-\theta\sqrt{d+\log^2(1/\varepsilon)}\right) && C \text{ sufficiently large} \\
&\leq \varepsilon^{-\theta}
\end{aligned}$$

■

In our application of volumetric spanners to BLO, we also need the following relaxation of volumetric spanners where we allow ourselves the flexibility to scale the ellipsoid:

Definition 23 *A ρ-ratio-volumetric spanner S of $\mathcal{K}$ is a subset $S \subseteq \mathcal{K}$ such that for all $x \in \mathcal{K}$, $\|x\|_{\mathcal{E}(S)} \leq \rho$.*

One example for such an approximate spanner with $\rho = \sqrt{d}$ is a barycentric spanner (Definition 17). In fact, it is easy to see that a C-approximate barycentric spanner is a $C\sqrt{d}$-ratio-volumetric spanner . The following is immediate from Theorem 18.

Corollary 24 *Let $\mathcal{K}$ be a compact set in $\mathcal{R}^d$ that is not contained in any proper linear subspace. Given an oracle for optimizing linear functions over $\mathcal{K}$, for any $C > 1$, it is possible to compute a $C\sqrt{d}$-ratio-volumetric spanner S of $\mathcal{K}$ of cardinality $|S| = d$, using $O(d^2\log_C(d))$ calls to the optimization oracle.*

5. Fast Volumetric Spanners for Discrete Sets

In this section we describe a different algorithm that constructs volumetric spanners for discrete sets. The order of the spanners we construct here is suboptimal (in particular, there is a dependence on the size of the set $\mathcal{K}$ which we didn't have before). However, the algorithm is particularly simple and efficient to implement (takes time linear in the size of the set).

Algorithm 1 Fast Volumetric Spanner construction

1: Input $\mathcal{K} = \{x_1, ..., x_n\} \subseteq \mathbb{R}^d$, $C \in \mathbb{R}$.
2: Set $T = \emptyset$
3: **while** $|K| > Cd\log d$ **do**
4: Compute $\Sigma = \sum_i x_i x_i^\top$ and let $u_i = \Sigma^{-1/2}x_i$, $p_i = 1/2n + \|u_i\|^2/2d$.
5: Set $S \leftarrow \emptyset$.
6: **for** $i = 1, ..., Cd\log d$ **do**
7: sample with replacement from $[n]$ according to $p_1, \ldots, p_n$: $S \leftarrow S \cup \{i\}$ w.p. p_i
8: **end for**
9: **if** $|\{i\ ,\ \|x_i\|_{\mathcal{E}(S)} \leq 1\}| < \frac{n}{2}$ **then**
10: Set $S \leftarrow \emptyset$ and GOTO step (6).
11: **end if**
12: Set $T \leftarrow T \cup S$
13: Set the remainder as $\mathcal{K} \leftarrow \{x_i \text{ s.t. } \|x_i\|_{\mathcal{E}(S)} > 1\}$
14: **end while**
15: **return** $T \leftarrow T \cup \mathcal{K}$

Theorem 25 *Given a set of vectors $\mathcal{K} = \{x_1, \ldots, x_n\} \in \mathbb{R}^d$, Algorithm 1 outputs a volumetric spanner of size $O((d\log d)(\log n))$ and has an expected running time of $O(nd^2)$.*

Proof Consider a single iteration of the algorithm with input $v_1, \ldots, v_n \in \mathbb{R}^d$. We claim that the random set S obtained in Step 7 satisfies the following condition with constant probability:

$$\Pr_{x\in\mathcal{K}}\left[\|x\|_{\mathcal{E}(S)} \leq 1\right] \geq 1/2 \tag{3}$$

Suppose the above statement is true. Then, the lemma follows easily as it implies that for the next iteration there are fewer than $n/2$ vectors. Hence, after $(\log n)$ recursive calls we will have a volumetric spanner. The total size of the set will be $O((d\log d)(\log n))$. To see the time complexity, consider a single run of the algorithm. The most computationally intensive steps are computing Σ and $\Sigma^{-1/2}$ which take time $O(nd^2)$ and $O(d^3)$ respectively. We also need to compute $(\sum_{v\in S} vv^\top)^{-1}$ (to compute the $\mathcal{E}(S)$ norm) which takes time $O(d^3\log d)$, and compute the $\mathcal{E}(S)$ norm of all the vectors which requires $O(nd^2)$. As $n = \Omega(d\log(d))$, it follows that a single iteration runs of a total expected time of $O(nd^2)$. Since the size of n is split in half between iterations, the claim follows.

We now prove that Equation 3 holds with constant probability

$$x_j^\top\left(\sum_{v\in S} vv^\top\right)^{-1} x_j = u_j^\top\left(\sum_{v\in S'} vv^\top\right)^{-1} u_j. \tag{4}$$

where $S' = \{\Sigma^{-1/2}v | v \in S\}$ is the (linearly) shifted version of S. Therefore, it suffices to show that with sufficiently high probability, the right hand side of the above equation is bounded by 1 for at least $n/2$ indices $j \in [n]$.

Note that $p_i = 1/2n + \|u_i\|^2/2d$ form a probability distribution: $\sum_i p_i = 1/2 + (\sum_i \|u_i\|^2)/2d = 1$. Let $X \in \mathbb{R}^d$ be a random variable with $X = u_i/\sqrt{p_i}$ with probability p_i for $i \in [n]$. Then, $\mathbf{E}[XX^\top] = I_d$. Further, for any $i \in [n]$

$$\|u_i\|^2/p_i \leq 2d.$$

Therefore, by Theorem 16, if we take $M = Cd(\log d)$ samples $X_1, \ldots, X_M$ for C sufficiently large, then with probability of at least $1/2$, it holds that

$$\sum_{i=1}^{M} X_i X_i^\top \succeq (M/2) I_d.$$

Let $T \subseteq [n]$ be the multiset corresponding to the indices of the sampled vectors $X_1, \ldots, X_M$. The above inequality implies that

$$\sum_{i \in T} \frac{1}{p_i} u_i u_i^\top \succeq (M/2) I_d.$$

Now,

$$\sum_{v \in S'} vv^\top \succeq (\min_i p_i) \sum_{v \in S'} \frac{1}{p_i} vv^\top \succeq (\min_i p_i)(M/2) I_d \succeq (M/4n) I_d.$$

Therefore,

$$\begin{aligned}
\sum_{i=1}^{n} u_i^\top \left(\sum_{v \in S'} vv^\top \right)^{-1} u_i &= \sum_{i=1}^{n} \mathrm{Tr}\left(\left(\sum_{v \in S'} vv^\top \right)^{-1} \left(u_i u_i^\top \right) \right) \\
&= \mathrm{Tr}\left(\left(\sum_{v \in S'} vv^\top \right)^{-1} \left(\sum_{i=1}^{n} u_i u_i^\top \right) \right) \\
&= \mathrm{Tr}\left(\left(\sum_{v \in S'} vv^\top \right)^{-1} \right) \\
&\leq \frac{4nd}{M} \leq \frac{4n}{C \log d} \leq \frac{n}{2 \log d},
\end{aligned}$$

for C sufficiently large. Therefore, by Markov's inequality and Equation 4, it follows that Equation 3 holds with high probability. The theorem now follows. ■

6. Experiment design using volumetric spanners

The active linear regression (ALR) problem is formally defined as follows. The input is a pool of n data points $\mathcal{K} = \{x_1, \ldots, x_n\} \subseteq \mathcal{R}^d$, a tolerable error $\varepsilon > 0$ and a confidence

parameter δ. A *query* to a point x returns an unbiased noisy estimate of $\langle w^*, x \rangle$ for some unknown vector $w^* \in \mathcal{R}^d$, with variance bounded by 1. Our objective is to actively choose points to query, and based on these queries, obtain a vector $w \in \mathcal{R}^d$ that with probability at least $1 - \delta$ has a *max-error* of

$$\max_{x \in \mathcal{K}} |\langle w^*, x \rangle - \langle w, x \rangle| < \varepsilon$$

We aim to minimize the *query complexity* of the solution, which is the mean number of queries required by the process. To avoid tedious definitions and details we assume that

1. all of the data points $\{x_i\}$ and w^* are bounded in the Euclidean unit sphere
2. the noise is bounded in absolute value by one with probability one.

Our results can be extended to remove the first assumption, simply by allowing an additional parameter which is the radius of the minimal enclosing Euclidian ball.

It is also possible to remove the second assumption, although this requires more robust estimators, e.g., the median-of-means estimator[6].

6.1 A lower bound for passive linear regression

In this section we provide an example for a set $\mathcal{K}$ where the passive learning algorithm must use $\Omega(\frac{n}{\varepsilon^2})$ observations to obtain a regressor w with additive error of at most ε on all of the data points. We start by formally defining the passive setup. Here, a query returns a random point x chosen uniformly from the set $\mathcal{K}$ and an unbiased noisy measurement of $\langle w^*, x \rangle$, with variance of at most 1. As before we assume that all points, including w^* are contained in the ℓ_2 unit sphere.

The set $\mathcal{K}$ in our example is defined in the following manner. Let $Y \subseteq \mathcal{R}^d$ be an arbitrary set of size $n - 1$ such that for all $x \in Y$, $\langle x, e_1 \rangle = 0$. Let $\mathcal{K} = Y \cup \{e_1\}$. Here, e_1 is the first vector in the standard basis for $\mathcal{R}^d$.

Theorem 26 *Any algorithm in the passive setting achieving an additive error of at most ε in all of the data points of $\mathcal{K}$ whose success probability is $1 - \delta$ requires $\Omega(\log(1/\delta)n/\varepsilon^2)$ queries.*

The theorem is an immediate corollary of the following lemma.

Lemma 27 *For $\mathcal{K}$ defined above, any policy distinguishing between the case where $\langle w^*, e_1 \rangle = -\varepsilon$ and $\langle w^*, e_1 \rangle = \varepsilon$ with probability larger than $1 - \delta$ must use $\Omega(n \log(1/\delta)/\varepsilon^2)$ queries.*

Proof We begin by mentioning that a query of a point x where $\langle x, e_1 \rangle = 0$ provides no information to the sign of $\langle w^*, e_1 \rangle$, hence does not help distinguish between the two hypotheses. The following lemma provides a lower bound to the number of queries at point e_1 required to estimate the $\langle w^*, e_1 \rangle$ up to a sufficiently small additive error and with sufficient confidence. It is a folklore lemma in statistics and appears e.g., in (Mannor and Tsitsiklis, 2004) in a much more general form.

6. The median-of-means estimator is defined as follows: Given m queries to a single dimensional distribution, the estimator of its mean is not taken to be the average of the queries; the m queries are instead split into $\log(1/\delta)$ equal sized buckets, each bucket has its average computed and the estimator is the median of these averages.

Lemma 28 (Theorem 1 of Mannor and Tsitsiklis 2004) *Let $\mathcal{D}$ be a distribution over $[-1,1]$. Let $\varepsilon > 0$ be such that for $X \sim \mathcal{D}$, $|\mathbf{E}[X]| \geq \varepsilon$. Let T be the expected number of queries required by any algorithm that queries i.i.d copies of $X \sim \mathcal{D}$ until being able to distinguish, with probability at least $1-\delta$ between the cases $\mathbf{E}[X] \leq -\varepsilon$ and $\mathbf{E}[X] \geq \varepsilon$. Then for universal constants $\varepsilon_0 > 0$, $\delta_0 > 0$, c_1, c_2 it holds that if $\varepsilon < \varepsilon_0$ and $\delta < \delta_0$ then $T \geq \frac{c_1 \log(c_2/\delta)}{\varepsilon^2}$.*

It follows that the expected number of queries needed in order to distinguish between the two hypotheses with probability $\geq 1-\delta$ is at least $\frac{c_1 n \log(c_2/\delta)}{\varepsilon^2}$, as the probability of observing a query to the inner product with e_1 is $1/n$. ■

6.2 Our ALR solution

In this section prove the following.

Theorem 29 *There exists an algorithm to the ALR problem with success probability of at least*
$1-\delta$ and with the following properties:

1. *The algorithm requires a preprocessing stage for building a volumetric spanner over $\mathcal{K}$*
2. *It's running time (after preprocessing) is $\tilde{O}\left(\frac{nd\log(1/\delta)}{\varepsilon^2}\right)$*
3. *It's query complexity is at most $O\left(\frac{d\log(n)\log(1/\delta)}{\varepsilon^2}\right)$.*

The intuition behind the algorithm is the following. We begin with a preprocessing stage of computing a volumetric spanner S for the set of points $\mathcal{K}$. Given this spanner we can implement a procedure that outputs, for all of the points of $\mathcal{K}$ simultaneously, an unbiased estimator of $\langle w^*, x\rangle$ with variance of at most $|S|$. To demonstrate the usefulness of this estimator, consider averaging $|S|\log(n/\delta)/\varepsilon^2$ i.i.d outputs of S. Standard concentration bound show that w.p at least $1-\delta$ the estimates of all points in $\mathcal{K}$ are correct up to an additive error of ε. Rather than computing a noisy output for w^* on the points and recovering a hypothesis w from that we use a technique by (Clarkson et al., 2012) that given an oracle for a function over a set of data points constructs a hypothesis w using a small number of queries to the oracle.

6.2.1 Constructing a low variance estimator

The main component of our method is a black box providing a noisy estimate of $\langle x, w^*\rangle$ to all of the data point of $\mathcal{K}$ simultaneously. The intuition behind the algorithm we describe next, is that given sufficiently many queries to the noisy estimator, a union bound argument can ensure an accurate estimate in all of the data points simultaneously.

We begin with the description of this black box providing the estimates. The main tool used for this 'all-point-estimator' is a volumetric spanner for the set $\mathcal{K}$. Algorithm 2 provides the formal description of how a volumetric spanner can be used to obtain these estimates.

The following lemma provides the analysis of Algorithm 2.

Algorithm 2 Sample($\mathcal{K}$)

1: Input: set $\mathcal{K} = \{x_1, ..., x_n\}$, Volumetric spanner for $\mathcal{K}$ denoted S, and measurement oracle that given x returns an unbiased estimator $\widehat{\langle x, w^*\rangle}$ with variance at most one for some fixed w^*.
2: Choose a point $v_j \in S$ uniformly at random, query its inner product $\hat{\ell} = \widehat{\langle v_j, w^*\rangle}$
3: let V be the $d \times |S|$ matrix whose columns are the elements of S, and let $V^\dagger \in \mathcal{R}^{|S|\times d}$ be its Moore-Penrose pseudo inverse.
4: For $x \in \mathcal{K}$, let $\alpha_x = V^\dagger x$ and let $\hat{\ell}_x \leftarrow (\alpha_x)_j \hat{\ell} \cdot |S|$
5: **return** estimates $\{\hat{\ell}_x\}_{x\in\mathcal{K}}$

Lemma 30 *Algorithm 2 queries a single point from $\mathcal{K}$. Its estimates have the properties of (1) being unbiased and (2) have a variance of at most $12d$. More formally, we have*

$$\forall x \in \mathcal{K}\ .\ \mathbf{E}[\hat{\ell}_x] = \langle x, w^*\rangle\ ,\ \mathbf{Var}(\hat{\ell}_x) \leq |S| \leq 12d$$

Proof

$$\begin{aligned}\mathbf{E}\left[\hat{\ell}_x\right] &= \textstyle\sum_{j\in S} \Pr[v_j] \cdot (\alpha_x)_j \,\mathbf{E}\,\widehat{\langle v_j, w^*\rangle} \cdot |S| \\ &= \textstyle\sum_{j\in S} (\alpha_x)_j \,\mathbf{E}\,\widehat{\langle v_j, w^*\rangle} \\ &= (V^\dagger x)^T V^T w^* = \langle x, w^*\rangle\end{aligned}$$

For the variance, recall that $x \in \mathcal{K}$ and S is a volumetric spanner of $\mathcal{K}$ indicates that $\|\alpha_x\|_2 \leq 1$:

$$\begin{aligned}\mathbf{E}[\hat{\ell}_x^2] &= \textstyle\sum_{j\in S} \Pr[v_j] \cdot (\alpha_x)_j^2 \,\mathbf{E}\left[\widehat{\langle v_j, w^*\rangle}^2\right] \cdot |S|^2 \\ &\leq |S| \textstyle\sum_{j\in S} (\alpha_x)_j^2 \leq |S|\end{aligned}$$

By Theorem 3 we can efficiently construct volumetric spanners of size $|S| = 12d$. ■

6.2.2 Algorithm and its analysis

In this section we present an algorithm for the ALR problem, following the primal-dual paradigm as in (Clarkson et al., 2012), and specifically their meta algorithm 3. This latter meta-algorithm can be used to solve any convex constrained feasibility problem, of which our ALR problem is a special case. The idea is to apply a low regret algorithm to a distribution over the constraints. The distribution over the constraints is changed according to a multiplicative update rule. The specific meta-algorithm we apply uses random estimates of the constraints that enable faster running time. We proceed to spell out the details.

To avoid extraneous notions we will assume henceforth w.l.o.g that $\mathcal{K}$ is symmetric meaning that $x \in \mathcal{K}$ iff $-x \in \mathcal{K}$. This is without loss of generality since an unbiased estimator for $\langle -x, w^*\rangle$ is obtained by negating the estimator for $\langle x, w^*\rangle$.

We write the ALR problem as the following mathematical program:

$$\min_{\|w\|\leq 1} g(w) \quad s.t. \qquad g(w) = \max_{x\in\mathcal{K}} c_x(w) \tag{5}$$
$$c_x(w) = \langle x, w\rangle - \langle x, w^*\rangle$$

Note that by definition $g(w^*) = 0$, which is the optimal solution to the problem as $\mathcal{K}$ is symmetric. In addition, an ε approximate solution to the ALR instance, assuming $\|w^*\| \leq 1$, corresponds to a vector $\hat{w}$ with $g(\hat{w}) \leq \varepsilon$.

Algorithm 3 Primal-Dual Algorithm for ALR

Input: T
Let $w_1 \leftarrow 0$, $q_0 \leftarrow \mathbf{1}_n$, $\eta \leftarrow \frac{1}{100}\sqrt{\frac{\log(n)}{T}}$.
for $t = 1$ to T **do**
  Query **Sample**($\mathcal{K}$) $12d$ times to obtain unit-variance zero-mean estimators $\tilde{a}_t(i)$ for all constraints c_i's:

$$\tilde{a}_t(i) \stackrel{\text{def}}{=} \langle x_i, w_t\rangle - \widehat{\langle x_i, w^*\rangle}$$

  where $\widehat{\langle x_i, w^*\rangle}$ is the estimate of $\langle x_i, w^*\rangle$ given by **Sample**($\mathcal{K}$).
  for $i \in [n]$ **do**
    $a_t(i) \leftarrow \text{clip}(\tilde{a}_t(i), 1/\eta)$, where

$$\text{clip}(\alpha, \beta) = \begin{cases} \min\{\alpha, |\beta|\} & \alpha \geq 0 \\ \max\{-\alpha, -|\beta|\} & \alpha < 0 \end{cases}$$

    $q_t(i) \leftarrow q_{t-1}(i)(1 - \eta a_t(i) + \eta^2 a_t(i)^2)$
  end for
  Choose $i_t \in [n]$ at random with $\Pr[i_t = i] \propto q_t(i)$
  $w_t \leftarrow w_{t-1} - \frac{1}{\sqrt{t}}\nabla_w c_{i_t}$, where $\nabla_w c_{i_t} = x_{i_t}$
end for
return $\bar{w} = \frac{1}{T}\sum_t w_t$

The following theorem bounds the performance of Algorithm 3. It immediately follows from Lemma 4.1 in (Clarkson et al., 2012).

Theorem 31 *Algorithm 3 runs in time $\tilde{O}(\frac{dn}{\varepsilon^2})$ and requires $O(\frac{d\log n}{\varepsilon^2})$ queries to the procedure* **Sample**($\mathcal{K}$)*. It returns, with probability of at least $\frac{1}{2}$, a vector w such that $\max_{x\in\mathcal{K}}\langle w - w^*, x\rangle \leq \varepsilon$.*

Proof

Notice that Algorithm 3 is an instantiation of Alg 3 from (Clarkson et al., 2012) applied to mathematical Program 5 with the following arguments:

1. The primal decision set $\{\|w\| \leq 1\}$ and (linear) cost functions $c_x(w)$, admits an iterative low regret algorithm, namely online gradient descent, with expected regret $\mathbf{E}[R(T)] \leq 2\sqrt{T}$. This follows since the norms of x, w (for all $x \in \mathcal{K}$) are bounded by one. See e.g., Theorem 1 by (Zinkevich, 2003).

   The expression $T_\epsilon(LRA)$ (here LRA stands for Low Regret Algorithm, and not the Active Linear Regression problem we are addressing) refers to the number T such that the average regret of online gradient descent is at most ϵ, in our case, this is $\frac{4}{\epsilon^2}$.

2. Meta-algorithm 3 in (Clarkson et al., 2012) assumes an oracle to a procedure **Sample**($\mathcal{K}$) that returns a vector of length $|\mathcal{K}|$ whose entries are unbiased estimators of $\langle x, w^*\rangle$, for all $x \in \mathcal{K}$ whose variance is upper bounded by 1. Recall that such a procedure, with variance $12d$ rather than 1, was given in Section 6.2.1. By averaging $12d$ such samples we obtain unit-variance estimates.

Thus, Lemma 4.1 (Clarkson et al., 2012) implies that the algorithm returns w.p. $\frac{1}{2}$ an ϵ-approximate solution in number of iterations bounded by

$$\max\{T_\epsilon(LRA), \frac{\log n}{\epsilon^2}\} \leq \frac{\log n}{\epsilon^2}$$

Each iteration involves elementary operations that can be implemented in time $\tilde{O}(nd)$ and $O(d)$ queries to **Sample**.

■

6.2.3 Validation and high probability algorithm

Algorithm 3 provides a method to obtain an approximated solution to the ALR problem with probability $1/2$. We now describe a method to amplify the success probability to $1-\delta$. The idea is to use a validation procedure and repeat the algorithm multiple times. It is easy to see that with a validation process, repeating Algorithm 3 for $O(\log(1/\delta))$ many times will increase the probability of success to $1 - \delta$.

We now describe a validation procedure that is in itself random, in the sense that it may err but its error probability is manageable. Algorithm 4 is given as input a hypothesis w and a ALR problem. It verifies, w.h.p., that w is an ε-approximate solution to the ALR problem accurate.

Algorithm 4 Verification

Input: Volumetric spanner S, parameters $\varepsilon, \delta > 0$, hypothesis $w \in \mathcal{R}^d$.
run **Sample**($\mathcal{K}$) $T = 2\ln(2n/\delta)|S|/\varepsilon^2$ times and obtain for each data point in $\mathcal{K}$, T i.i.d samples of $\langle w^*, x\rangle$
for each $x \in \mathcal{K}$ let $\tilde{f}(x)$ be the average of the above T samples.
declare w as accurate iff for all x, $|\langle w, x\rangle - \tilde{f}(x)| < 2\varepsilon$

Lemma 32 *Algorithm 4 has the following properties:*

- *It requires $O(\log(n/\delta)|S|/\varepsilon^2)$ queries to the oracle* **Sample**($\mathcal{K}$)

- *If the worst-case error of w is bounded by ε then w.p. at least $1-\delta$ it will be declared as accurate*
- *If the worst-case error of w is larger than 3ε then w.p. at least $1-\delta$ it will be declared as inaccurate.*

Proof We recall that the process **Sample**($\mathcal{K}$) returns unbiased estimates of $\langle w^*, x\rangle$ for all of the data points where the estimates are bounded in absolute value by $|S|$. Fix some point $x \in \mathcal{K}$. Chernoff's inequality dictates that

$$\Pr\left[\left|\langle w^*, x\rangle - \tilde{f}(x)\right| > \varepsilon\right] \leq 2\exp\left(-\frac{\varepsilon^2}{2|S|}\cdot\frac{2\ln(2n/\delta)|S|}{\varepsilon^2}\right) = \delta/n$$

Hence, w.p at least $1-\delta$ it holds for all $x \in \mathcal{K}$ simultaneously that $\left|\langle w^*, x\rangle - \tilde{f}(x)\right| < \varepsilon$. The claim immediately follows by using the triangle inequality in order to bound $\left|\langle w, x\rangle - \tilde{f}(x)\right|$. ■

Corollary 33 *There exists an algorithm that runs in time $\tilde{O}\left(\frac{dn\log\frac{1}{\delta}}{\varepsilon^2}\right)$ and returns, with probability of at least $1-\delta$, a vector w such that $\max_{x\in\mathcal{K}}\langle w - w^*, x\rangle \leq \varepsilon$.*

Proof Given the parameter δ, we run $\log(1/2\delta)$ independent copies of Algorithm 3 with parameter $\varepsilon' = \varepsilon/3$. Each such copy will produce a hypothesis w. We check for each such hypothesis w whether it is $\varepsilon' = \varepsilon/3$ accurate using Algorithm 4, with success probability $1-\delta/2\log(1/2\delta)$. With probability $1/2\delta$, all of the occurrences of the validation procedures will not err. Also, with probability at least $1-2\delta$ at least one hypothesis will be sufficiently accurate. Hence, by union bound we have with probability at least $1-\delta$ that at least one hypothesis will be declared accurate and any of the hypotheses declared accurate will be at least $3\varepsilon' = \varepsilon$ accurate. This concludes the quality of the output of the algorithm. The running time analysis is trivial. ■

7. Bandit Linear Optimization

Recall the problem of Bandit Linear Optimization (BLO): iteratively at each time sequence t, the environment chooses a loss vector L_t that is not revealed to the player. The player chooses a vector $x_t \in \mathcal{K}$ where $\mathcal{K} \subseteq \mathbb{R}^d$ is convex, and once she commits to her choice, the loss $\ell_t = x_t^\top L_t$ is revealed. The objective is to minimize the loss and specifically, the regret, defined as the strategy's loss minus the loss of the best fixed strategy of choosing some $x^* \in \mathcal{K}$ for all t. We henceforth assume that the loss vectors L_t's are chosen from the polar of $\mathcal{K}$, meaning from $\{L : |L^\top x| \leq 1\ \forall x \in \mathcal{K}\}$. In particular this means that the losses are bounded in absolute value, although a different choice of assumption (i.e., ℓ_∞ bound on the losses) can yield different regret bounds, see discussion in (Audibert et al., 2011).

The problem of BLO is a natural generalization of the classical Multi-Armed Bandit problem and extremely useful for efficiently modeling decision making under partial feedback for structured problems. As such the research literature is rich with algorithms and insights

into this fundamental problem (see surveys Bubeck and Cesa-Bianchi 2012b and Hazan 2014). In this section we focus on the first efficient and optimal-regret algorithm, and thus immediately jump to Algorithm 5. We make the following assumptions over the decision set $\mathcal{K}$:

1. The set $\mathcal{K}$ is equipped with a membership oracle. This implies via the results by (Lovász and Vempala, 2007) (Lemma 10) that there exists an efficient algorithm for sampling from a given log-concave distribution over $\mathcal{K}$. Via the discussion in previous sections, this also implies that we can construct approximate (both types of approximations, see Definitions 23 and 4) volumetric spanners efficiently over $\mathcal{K}$.
2. The losses are bounded in absolute values by 1. That is, the loss functions are always chosen (by an oblivious adversary) from a convex set $\mathcal{Z}$ such that $\mathcal{K}$ is contained in its polar, i.e., $\forall L \in \mathcal{Z}, x \in \mathcal{K}$, $|L^\top x| \leq 1$. This implies that the set $\mathcal{K}$ admits for any $\varepsilon > 0$ an ε-net, w.r.t the norm defined by $\mathcal{Z}$, whose size we denote by $|K|_\varepsilon \leq (\varepsilon/2)^{-d}$.

For Algorithm 5 we prove the optimal regret of Theorem 34.

Remark: notice that to obtain a (p,ε)-exp-volumetric spanner for a log-concave distribution p over a body $\mathcal{K}$ we simply choose sufficiently many i.i.d samples from p. Since in Algorithm 5 p_t is always log-concave, it follows that S'_t consists of i.i.d samples from p_t, meaning that if we would not have required S''_t, the exploration and exploration strategies would be the same! Since we still require the set S''_t, there exists a need for a separate exploration strategy. Interestingly, the $2\sqrt{d}$-ratio-volumetric spanner is obtained by taking a barycentric spanner, which is the exploration strategy given in (Dani et al., 2007).

Algorithm 5 GeometricHedge with Volumetric Spanners Exploration

$\mathcal{K}$, parameters γ, η, horizon T.
$p_1(x)$ uniform distribution over $\mathcal{K}$.
for $t = 1$ to T **do**
  Let S'_t be a $(p_t, \exp(-(4\sqrt{d} + \log(2T))))$-exp-volumetric spanner of $\mathcal{K}$.
  Let S''_t be a $2\sqrt{d}$-ratio-volumetric spanner of $\mathcal{K}$
  Set S_t as the union of S'_t, S''_t.
  $\hat{p}_t(x) = (1-\gamma)p_t(x) + \frac{\gamma}{|S_t|}1_{x\in S_t}$
  sample x_t according to $\hat{p}_t$ (via the tools described in Lemma 10)
  observe loss $\ell_t \triangleq L_t^\top x_t$
  Let $C_t \triangleq \mathbf{E}_{x\sim\hat{p}_t}[xx^\top]$
  $\hat{L}_t \triangleq \ell_t C_t^{-1} x_t$
  $p_{t+1}(x) \propto p_t(x)e^{-\eta\hat{L}_t^\top x}$
end for

Theorem 34 *Under the assumptions stated above, and let $s = \max_t |S_t|$, $\eta = \sqrt{\frac{\log|\mathcal{K}|_{1/T}}{dT}}$ and let $\gamma = s\sqrt{\frac{\log(|\mathcal{K}|_{1/T})}{dT}}$. Algorithm 5 given parameters γ, η suffers a regret bounded by*

$$O\left((s+d)\sqrt{\frac{T\log|\mathcal{K}|_{1/T}}{d}}\right)$$

We note that while the size $\log(|\mathcal{K}|_{1/T})$ can be bounded by $d\log(T)$, in certain scenarios such as s-t paths in graphs it is possible to obtain sharper upper bounds that immediately imply better regret via Theorem 34.

Corollary 35 *There exist an efficient algorithm for BLO for any convex set $\mathcal{K}$ with regret of*

$$O\left(\sqrt{dT\log|\mathcal{K}|_{1/T}}\right) = O\left(d\sqrt{T\log(T)}\right)$$

Proof The spanner in Step 4 of the algorithm does not have to be explicitly constructed. According to Theorem 5, to obtain such as spanner it suffices to sample sufficiently many points from the distribution p_t, hence this portion of the exploration strategy is identical to the exploitation strategy.

According to Corollary 24, a $2\sqrt{d}$-ratio-volumetric spanner of size d can be efficiently constructed, given a linear optimization oracle which in turn can be efficiently implemented by the membership oracle for $\mathcal{K}$. Hence, it follows that for the purpose of the analysis, $s = d$ and the bound follows. ■

To prove the theorem we follow the general methodology used in analyzing the performance of the geometric hedge algorithm. The major deviation from standard technique is the following sub-exponential tail bound, which we use to replace the the standard absolute bound for $|\hat{L}_t x|$. After giving its proof and a few auxiliary lemmas, we give the proof of the main theorem.

Lemma 36 *Let $x \sim p_t$, $x_t \sim \hat{p}_t$ and let $\hat{L}_t$ be defined according to x_t. It holds, for any $\theta > 1$ that*

$$\Pr\left[|\hat{L}_t^\top x| > \frac{\theta s}{\gamma}\right] \le \exp(-2\theta)/T$$

Proof

$$\begin{aligned}
\Pr\left[|\hat{L}_t^\top x| > \frac{\theta s}{\gamma}\right] &\le \Pr\left[\|x\|_{\mathcal{E}(S_t)} \cdot \|x_t\|_{\mathcal{E}(S_t)} \ge \theta\right] && \text{Lemma 37}\\
&\le \Pr\left[\|x\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta} \ \bigvee \ \|x_t\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta}\right]\\
&\le \Pr\left[\|x\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta}\right] + \Pr\left[\|x_t\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta}\right]\\
&\le 2\Pr\left[\|x\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta}\right]
\end{aligned}$$

To justify the last inequality notice that $x \sim p_t$ and $x_t \sim \hat{p}_t$ where $\hat{p}_t$ is a convex sum of p_t and a distribution q_t for which $\Pr_{y\sim q_t}\left[\|y\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta} > 1\right] = 0$. Before we continue recall that we can assume that $\sqrt{\theta} \le 2\sqrt{d}$, since S_t'' is a $2\sqrt{d}$-ratio-volumetric spanner .

$$\begin{aligned}
\Pr\left[|\hat{L}_t^\top x| > \frac{\theta s}{\gamma}\right] &\le 2\Pr\left[\|x\|_{\mathcal{E}(S_t)} \ge \sqrt{\theta}\right]\\
&\le 2\exp(-\sqrt{\theta}(4\sqrt{d} + \log 2T)) && \text{property of exp-volumetric spanner}\\
&\le \tfrac{1}{T}\exp(-2\sqrt{\theta\cdot 4d})\\
&\le \tfrac{1}{T}\exp(-2\theta) && \text{since } \theta \le 4d
\end{aligned}$$

■

Lemma 37 *For all $x \in \mathcal{K}$ it holds that $|\hat{L}_t^\top x| \le \frac{|S_t|\|x\|_{\mathcal{E}(S_t)}\|x_t\|_{\mathcal{E}(S_t)}}{\gamma}$.*

Proof Let $x \in \mathcal{K}$. Denote by V_t the matrix whose columns are the elements of S_t and recall that $\|y\|^2_{\mathcal{E}(S_t)} = y^\top(V_tV_t^\top)^{-1}y$. Since $C_t \triangleq \mathbf{E}_{x\sim\hat{p}_t}[xx^\top]$, it holds that

$$C_t \succeq \frac{\gamma}{|S_t|}\sum_{v\in S_t} vv^\top = \frac{\gamma}{|S_t|}V_tV_t^\top$$

since both matrices are full rank, it holds that

$$C_t^{-1} \preceq \frac{|S_t|}{\gamma}(V_tV_t^\top)^{-1}$$

Notice that due to the Cauchy-Schwartz inequality,

$$|x^\top\hat{L}_t| = |\ell_t|\cdot|x^\top C_t^{-1}x_t| \le |\ell_t|\cdot\|x^\top C_t^{-1/2}\|\cdot\|C_t^{-1/2}x_t\|$$

The matrix $C_t^{-1/2}$ is defined as C_t is positive definite. Now,

$$\|x^\top C_t^{-1/2}\|^2 = x^\top C_t^{-1}x \le x^\top\frac{|S_t|}{\gamma}(V_tV_t^\top)^{-1}x = \frac{|S_t|}{\gamma}\|x\|^2_{\mathcal{E}(S_t)}$$

Since the analog can be said for $\|C_t^{-1/2}x_t\|$ (as $x_t \in \mathcal{K}$), it follows that

$$|x^\top\hat{L}_t| \le |\ell_t|\frac{|S_t|\|x\|_{\mathcal{E}(S_t)}\|x_t\|_{\mathcal{E}(S_t)}}{\gamma} \le \frac{|S_t|\|x\|_{\mathcal{E}(S_t)}\|x_t\|_{\mathcal{E}(S_t)}}{\gamma}$$

The last inequality is since we assume the rewards are in $[-1, 1]$. ■

Implementation for general convex bodies. In the case where the set $\mathcal{K}$ is a general convex body, the analysis must include the fact that we can only approximately sample a log-concave distribution over $\mathcal{K}$. As the main focus of our work is to prove a polynomial solution we present only a simple analysis yielding a running time polynomial in the dimension d and horizon T. It is likely that a more thorough analysis can substantially reduce the running time.

Corollary 38 *In the general case where an approximate sampling is required, Algorithm 5 can be implemented with a running time of $\tilde{O}(d^5 + d^3T^6)$ per iteration.*

Proof

Fix an error parameter δ, and let us run Algorithm 5 with the approximate samplers p'_t guaranteed by Theorem 10. Then, in each use of the sampler we are replacing the true distribution we should be using p_t, with a distribution p'_t such that statistical distance between p_t, p'_t is at most δ. Let us now analyze the error incurred by this approximation by bounding

the loss from the first round onwards. Suppose the algorithm ran with the approximate sampler in the first round but the exact sampler in each round afterwards. Then, as the statistical distance between the distributions is at most δ and the loss in each round is bounded by 1 and there are T rounds, the net difference in expected regret between using p_1 and p'_1 will be at most $\delta \cdot T$. Similarly, if we ran the algorithm with $p'_1, \ldots, p'_{i-1}, p'_i, p_{i+1}, \cdots, p_T$ as opposed to[7] $p'_1, \ldots, p'_{i-1}, p_i, p_{i+1}, \cdots, p_T$ (we are changing the i'th distribution from exact to approximate), the net difference in expected regret would be at most $\delta \cdot T$. Therefore, the total additional loss we may incur for using the approximate oracles is at most $T \cdot (\delta T) = \delta T^2$. Thus, if we take $\delta = \Delta/T^2$, where Δ is the regret bound from Theorem 34, we get a regret bound of 2Δ. The required value of δ is bounded by $T^{-1.5}$. Applying Theorem 10 leads to a running time of $\tilde{O}(d^5 + d^3 T^6)$ per iteration. ■

7.1 Proof of Theorem 34

We continue the analysis of the Geometric Hedge algorithm similarly to (Dani et al., 2007; Bubeck et al., 2012), under certain assumptions over the exploration strategy. For convenience we will assume that the set of possible arms $\mathcal{K}$ is finite. This assumption holds w.l.o.g since if $\mathcal{K}$ is infinite, a $\sqrt{1/T}$-net of it can be considered as described earlier (this will have no effect on the computational complexity of our algorithm, but a mere technical convenience in the proof below).

Before proving the theorem we will require three technical lemmas. In the first we show that $\hat{L}_t$ is an unbiased estimator of L_t. In the second, we bound a proxy of its variance. In the third, we bound a proxy of the expected value of its exponent.

Lemma 39 *In each t, $\hat{L}_t$ is an unbiased estimator of L_t*

Proof

$$\hat{L}_t = \ell_t C_t^{-1} x_t = (L_t^\top x_t) C_t^{-1} x_t = C_t^{-1}(x_t x_t^\top) L_t$$

Hence,

$$\mathop{\mathbf{E}}_{x_t \sim p_t}[\hat{L}_t] = C_t^{-1} \mathop{\mathbf{E}}_{x_t \sim p_t}[x_t x_t^\top] L_t = C_t^{-1} C_t L_t = L_t$$

■

Lemma 40 *Let $t \in [T]$, $x \sim p_t$ and $x_t \sim \hat{p}_t$. It holds that $\mathbf{E}[(\hat{L}_t^\top x)^2] \le d/(1-\gamma) \le 2d$*

Proof For convenience, denote by q_t the uniform distribution over S_t - the exploration strategy at round t. First notice that for any $x \in \mathcal{K}$,

$$\begin{aligned}\mathop{\mathbf{E}}_{x_t \sim \hat{p}_t}[(\hat{L}_t^\top x)^2] &= x^\top \mathbf{E}_{x_t \sim \hat{p}_t}[\hat{L}_t \hat{L}_t^\top] x = x^\top \mathbf{E}_{x_t \sim \hat{p}_t}[\ell_t^2 C_t^{-1} x_t x_t^\top C_t^{-1}] x \\ &= \ell_t^2 x^\top C_t^{-1} \mathbf{E}_{x_t \sim \hat{p}_t}[x_t x_t^\top] C_t^{-1} x = \ell_t^2 x^\top C_t^{-1} x \\ &\le x^\top C_t^{-1} x \end{aligned} \tag{6}$$

7. Here, p_j's are interpreted as the distribution given by the algorithm based on the distribution from previous round and p'_j is the approximate oracle for this distribution p_j.

Next,

$$\mathop{\mathbf{E}}_{x \sim \hat{p}_t}[x^\top C_t^{-1} x] = \mathop{\mathbf{E}}_{x \sim \hat{p}_t}[C_t^{-1} \bullet x x^\top] = C_t^{-1} \bullet \mathop{\mathbf{E}}_{x \sim \hat{p}_t}[x x^\top] = C_t^{-1} \bullet C_t = \mathbf{Tr}(I_d) = d$$

Where we used linearity of expectation and denote $A \bullet B = \mathbf{Tr}(AB)$. Since C_t^{-1} is positive semi definite,

$$(1-\gamma) \mathop{\mathbf{E}}_{x \sim p_t}[x^\top C_t^{-1} x] \le (1-\gamma) \mathop{\mathbf{E}}_{x \sim p_t}[x^\top C_t^{-1} x] + \gamma \mathop{\mathbf{E}}_{x \sim q_t}[x^\top C_t^{-1} x] = \mathop{\mathbf{E}}_{x \sim \hat{p}_t}[x^\top C_t^{-1} x] = d \tag{7}$$

The lemma follows from combining Equations 6 and 7. ■

Lemma 41 *Denote by $\mathbf{1}_\phi$ the random variable taking a value of 1 if event ϕ occurred and 0 otherwise. Let $t \in [T]$, $x_t \sim \hat{p}_t$ and $x \sim p_t$. For $\hat{L}_t$ defined by x_t it holds that*

$$\mathbf{E}\left[\exp(-\eta \hat{L}_t^\top x) \mathbf{1}_{-\eta \hat{L}_t^\top x > 1}\right] \le \frac{2}{T}$$

Proof Let f, F be the pdf and cdf of the random variable $Y = -\eta \hat{L}_t^\top x$ correspondingly. From Lemma 36 and the fact that $1/\eta = s/\gamma$ ($s = \max_t |S_t|$) we have that for any $\theta \ge 1$,

$$1 - F(\theta) \le \frac{1}{T} e^{-2\theta}$$

and we'd like to prove that under this condition,

$$\mathbf{E}[e^Y \mathbf{1}_{Y>1}] = \int_{\theta=1}^{\infty} e^\theta f(\theta) d\theta \le \frac{2}{T}$$

which follows from the definition of the cdf and pdf:

$$\begin{aligned}\mathbf{E}[e^Y \mathbf{1}_{Y>1}] &= \textstyle\int_{\theta=1}^{\infty} e^\theta f(\theta) d\theta \\ &= \textstyle\sum_{k=1}^{\infty} \int_{\theta=k}^{k+1} e^\theta f(\theta) d\theta \\ &\le \textstyle\sum_{k=1}^{\infty} e^{k+1} \int_{\theta=k}^{k+1} f(\theta) d\theta \\ &\le \textstyle\sum_{k=1}^{\infty} e^{k+1} (F(k+1) - F(k)) \\ &\le \textstyle\sum_{k=1}^{\infty} e^{k+1} (1 - F(k)) \\ &\le \textstyle\sum_{k=1}^{\infty} e^{k+1} \cdot \frac{1}{T} e^{-2k} && \text{Lemma 36} \\ &= \textstyle\frac{e}{T} \sum_{k=1}^{\infty} e^{-k} = \frac{e}{T} \cdot \frac{e^{-1}}{1-e^{-1}} \le \frac{2}{T}\end{aligned}$$

■

Proof [Proof of Theorem 34] For convenience we define within this proof for $x \in \mathcal{K}$, $\hat{\ell}_{1:t-1}(x) \triangleq \sum_{i=1}^{t-1} \hat{L}_i^\top x$ and let $\hat{\ell}_t(x) \triangleq \hat{L}_t^\top x$. Let $W_t = \sum_{x \in \mathcal{K}} \exp(-\eta \hat{\ell}_{1:t-1}(x))$. For all

$t \in [T]$:

$$\begin{aligned}
\mathbf{E}\left[\frac{W_{t+1}}{W_t}\right] &= \mathbf{E}\left[\sum_{x\in\mathcal{K}} \frac{\exp(-\eta\hat{\ell}_{1:t-1}(x))\exp(-\eta\hat{\ell}_t(x))}{W_t}\right] \\
&= \mathbf{E}_{x_t\sim\hat{p}_t}\left[\sum_{x\in\mathcal{K}} p_t(x)\exp(-\eta\hat{\ell}_t(x))\right] \\
&= \mathbf{E}_{x_t\sim\hat{p}_t, x\sim p_t}[\exp(-\eta\hat{\ell}_t(x))] \leq \\
&\leq 1 - \eta\,\mathbf{E}[\hat{L}_t^\top x] + \eta^2\,\mathbf{E}[(\hat{L}_t^\top x)^2] + \mathbf{E}\left[\exp(-\eta\hat{L}_t^\top x)\mathbf{1}_{-\eta\hat{L}_t^\top x>1}\right]
\end{aligned}$$

using the inequality $\exp(y) \leq 1 + y + y^2 + \exp(y)\cdot\mathbf{1}_{y>1}$

$$\leq 1 - \eta\,\mathbf{E}[\hat{L}_t^\top x] + \eta^2\,\mathbf{E}[(\hat{L}_t^\top x)^2] + \tfrac{2}{T} \qquad \text{Lemma 41}$$

Since $\hat{L}_t$ is an unbiased estimator of L_t (Lemma 39) and according to Lemma 40, $\mathbf{E}[(\hat{L}_t^\top x)^2] \leq 2d$, we get:

$$\mathbf{E}\left[\frac{W_{t+1}}{W_t}\right] \leq 1 - \eta L_t^\top \mathop{\mathbf{E}}_{x\sim p_t}[x] + 2\eta^2 d + \frac{2}{T} \tag{8}$$

We now use Jensen's inequality:

$$\begin{aligned}
\mathbf{E}[\log(W_T)] - \mathbf{E}[\log(W_1)] &= \mathbf{E}[\log(W_T/W_1)] \\
&= \textstyle\sum_{t=1}^{T-1} \mathbf{E}[\log(W_{t+1}/W_t)] \\
&\leq \textstyle\sum_{t=1}^{T-1} \log(\mathbf{E}[W_{t+1}/W_t]) && \text{Jensen} \\
&\leq \textstyle\sum_{t=1}^{T-1} \log\left(1 - \eta L_t^\top \mathbf{E}_{x\sim p_t}[x] + 2\eta^2 d + \frac{2}{T}\right) && (8) \\
&\leq \textstyle\sum_{t=1}^{T-1} -\eta L_t^\top \mathbf{E}_{x\sim p_t}[x] + 2\eta^2 d + \frac{2}{T} && \ln(1+y) \leq y \\
& && \text{for all } y > -1 \\
&\leq 2 + 2\eta^2 T d - \eta \textstyle\sum_t \mathbf{E}_{x\sim p_t}[L_t^\top x]
\end{aligned}$$

Now, since $\log(W_1) = \log(|\mathcal{K}|)$ and $W_T \geq \exp(-\eta\hat{\ell}_{1:T}(x^*))$ for any $x^* \in \mathcal{K}$, by shifting sides of the above it holds for any $x^* \in \mathcal{K}$ that

$$\sum_t \mathop{\mathbf{E}}_{x\sim p_t}[L_t^\top x] - \sum_t L_t^\top x^* \leq \sum_t \mathop{\mathbf{E}}_{x\sim p_t}[L_t^\top x] + \mathbf{E}[\log W_T] \leq \frac{\log(|\mathcal{K}|)+2}{\eta} + 2\eta T d$$

Finally, by noticing that

$$\sum_t \mathop{\mathbf{E}}_{x\sim \hat{p}_t}[L_t^\top x] - \sum_t \mathop{\mathbf{E}}_{x\sim p_t}[L_t^\top x] \leq \gamma T$$

we obtain a bound of

$$\mathbf{E}[\text{Regret}] = \mathbf{E}[\sum_t L_t^\top x_t] - \sum_t L_t^\top x^* = \sum_t \mathop{\mathbf{E}}_{x\sim\hat{p}_t}[L_t^\top x] - \text{Loss}(x^*) \leq \frac{\log(|\mathcal{K}|)+2}{\eta} + 2\eta T d + \gamma T$$

on the expected regret. By plugging in the values of η, γ we get the bound of

$$O\left((s+d)\sqrt{\frac{T\log(|\mathcal{K}|)}{d}}\right)$$

as required. ■

8. Conclusion and Open Question

We have described a geometric mechanism for exploration in machine learning problems and its application to experiment design as well as bandit linear optimization.

The following question in high-dimensional geometry remains open: what is the worst-case order of a given set in $\mathbb{R}^d$ (cardinality of its minimal volumetric ellipsoid, as per Definition 1)? A gap remains between our lower bound of $d+1$ and upper bound of $12d$.

Acknowledgments

We would like to thank Raghu Meka for helpful discussions in areas related to high dimensional geometry that were key to the results of this paper.

Elad Hazan gratefully acknowledges support from the European Research Council, in the form of a Marie Curie fellowship and an ERC Starting grant project SUBLRN.

Appendix A. Concentration bounds for non centered isotropic log concave distributions

We begin by proving an auxiliary lemma used in the proof of Corollary 13.

Lemma 42 *Let $\delta > 0$, $t \geq 1$, let d be a positive integer and let $n = \frac{Ct^4 d\log^2(t/\delta)}{\delta^2}$ for some sufficiently large universal constant C. Let $y_1, \ldots, y_n$ be i.i.d d-dimensional vectors from an isotropic log-concave distribution. Then*

$$\Pr\left[\left\|\frac{1}{n}\sum y_i\right\| > \delta\right] \leq \exp(-t\sqrt{d})$$

Proof For convenience let $S_n = \frac{1}{\sqrt{n}}\sum_{i=1}^n y_i$. Since the y's are independent, S_n is also log-concave distributed. Notice that $\mathbf{E}[S_n] = 0$ and $\mathbf{E}[S_n S_n^T] = \frac{1}{n}\sum \mathbf{E}[y_i y_i^T] = I_d$ hence S_n is isotropic. Now,

$$\begin{aligned}
\Pr\left[\left\|\frac{1}{n}\sum y_i\right\| > \delta\right] &= \Pr\left[\|S_n\| > \sqrt{n}\delta\right] \\
&= \Pr\left[\|S_n\| > \sqrt{d}\cdot\sqrt{Ct^4\log^2(t/\delta)}\right] \\
&\leq \Pr\left[\|S_n\| > \sqrt{d} + \sqrt{d}\cdot\tfrac{1}{2}t\sqrt{C}\right]
\end{aligned}$$

The last inequality holds for $t \geq 1$ and $C \geq 4$. It now follows from Theorem 14 that

$$\Pr\left[\left\|\frac{1}{n}\sum y_i\right\| > \delta\right] \leq c_1 \exp(-c_2 t\sqrt{Cd})$$

where c_1, c_2 are some universal constants. Since $t\sqrt{d} \geq 1$, setting $C \geq (\frac{1+\log(c_1)}{c_2})^2$ proves the claim. ■

Proof [Proof of Corollary 13] Let $a = \mathbf{E}[x]$ and let $\tilde{a} = \frac{1}{n}\sum x_i$. Notice that

$$\mathbf{E}[(x-a)(x-a)^T] = \mathbf{E}[xx^T] - \mathbf{E}[x]a^T - a\,\mathbf{E}[x] + aa^T = I_d - aa^T$$

is a PSD matrix hence $\|a\| \leq 1$. Consider the following equality.

$$\frac{1}{n}\sum_{i=1}^{n}(x_i-a)(x_i-a)^\top = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^\top - a\tilde{a}^\top - \tilde{a}a^\top + aa^\top$$

According to Lemma 42, w.p. at least $1-\exp(-t\sqrt{d})$,

$$\|\tilde{a} - a\| \leq \delta$$

in which case, since $\|a\| \leq 1$ and according to the triangle inequality,

$$\left\|\frac{1}{n}\sum_{i=1}^{n} x_i x_i^\top - I_d\right\| \leq \left\|\frac{1}{n}\sum_{i=1}^{n}(x_i-a)(x_i-a)^\top - (I_d - aa^\top)\right\| + 2\delta$$

According to Theorem 12, w.p. at least $1-\exp(-t\sqrt{d})$

$$\left\|\frac{1}{n}\sum_{i=1}^{n}(x_i-a)(x_i-a)^\top - (I_d - aa^\top)\right\| \leq \delta$$

and the corollary follows. ■

Proof [Proof of Corollary 15] Let $\mathbf{E}[x] = a$. Consider the r.v $y = x - a$. It holds that $\mathbf{E}[y] = 0$ and $\mathbf{E}[yy^T] = I_d - aa^T$. Notice that we can derive that

$$\|a\| \leq 1 \tag{9}$$

As $\mathbf{E}[yy^T]$ is a PSD matrix. Also, it is easy to verify that y is log-concave distributed. We now consider the r.v[8] $z = (I_d - aa^t)^{-1/2}y$. It is easy to verify that the distribution of z is also log-concave and isotropic. It follows, from Theorem 14 that for any $\theta \geq 2$

$$\Pr\left[\|z\| > \theta\sqrt{d}\right] \leq \Pr\left[\|z\| - \sqrt{d} > \frac{1}{2}\theta\sqrt{d}\right] \leq C'\exp(-c\theta\sqrt{d})$$

By using Equation 9 we get that for $\theta > 3$

$$\Pr\left[\|x\| > \theta\sqrt{d}\right] \leq \Pr\left[\|y\| > \theta\sqrt{d} - 1\right] \leq \Pr\left[\|z\| > (\theta - 1/\sqrt{d})\sqrt{d}\right] \leq C'\exp(c' - c'\theta\sqrt{d}).$$

The last inequality holds since $\theta - 1/\sqrt{d} \geq 2$. ■

8. if $I_d - aa^T$ is not of full rank then y is in fact supported in an affine subspace of rank $d-1$ and we can continue the analysis there.

References

J.D. Abernethy, E. Hazan, and A. Rakhlin. Interior-point methods for full-information and bandit online learning. *IEEE Transactions on Information Theory*, 58(7):4164–4175, 2012.

Radoslaw Adamczak, Alexander E. Litvak, Alain Pajor, and Nicole Tomczak-Jaegermann. Quantitative estimates of the convergence of the empirical covariance matrix in log-concave ensembles. *Journal of American Mathematical Society*, 23:535–561, 2010.

A.A.C. Atkinson and A.A.N. Donev. *Optimum Experimental Designs*. Oxford science publications. OXFORD University Press, 1992. ISBN 9780198522546. URL `http://books.google.co.il/books?id=cmmOA_-M7S0C`.

Jean-Yves Audibert and Olivier Catoni. Linear regression through pac-bayesian truncation. *arXiv preprint arXiv:1010.0072*, 2010.

Jean-Yves Audibert and Olivier Catoni. Robust linear least squares regression. *The Annals of Statistics*, pages 2766–2794, 2011.

Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. Minimax policies for combinatorial prediction games. In Sham M. Kakade and Ulrike von Luxburg, editors, *COLT*, volume 19 of *Journal of Machine Learning Research - Proceedings Track*, pages 107–132. JMLR.org, 2011.

Baruch Awerbuch and Robert Kleinberg. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.

Maria-Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. *Journal of Computer and System Sciences*, 75(1):78–89, January 2009. ISSN 0022-0000. doi: 10.1016/j.jcss.2008.07.003. URL `http://dx.doi.org/10.1016/j.jcss.2008.07.003`.

Keith Ball. An elementary introduction to modern convex geometry. In *Flavors of Geometry*, pages 1–58. Univ. Press, 1997.

Peter L. Bartlett, Varsha Dani, Thomas P. Hayes, Sham Kakade, Alexander Rakhlin, and Ambuj Tewari. High-probability regret bounds for bandit online linear optimization. In *Proceedings of The 21st Conference on Learning Theory (COLT)*, pages 335–342, 2008.

Joshua Batson, Daniel A Spielman, and Nikhil Srivastava. Twice-ramanujan sparsifiers. *SIAM Journal on Computing*, 41(6):1704–1721, 2012.

S. Bubeck and N. Cesa-Bianchi. *Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems*, volume 5 of *Foundations and Trends in Machine Learning*. NOW, 2012a.

Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5:1–122, 2012b.

Sébastien Bubeck, Nicolò Cesa-Bianchi, and Sham M. Kakade. Towards minimax policies for online linear optimization with bandit feedback. *Journal of Machine Learning Research - Proceedings Track*, 23:41.1–41.14, 2012.

Nicolò Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.

Kenneth L. Clarkson, Elad Hazan, and David P. Woodruff. Sublinear optimization for machine learning. *Journal of the ACM (JACM)*, 59(5):23:1–23:49, November 2012.

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.

S. Damla Ahipasaoglu, Peng Sun, and Michael J. Todd. Linear convergence of a modified frank–wolfe algorithm for computing minimum-volume enclosing ellipsoids. *Optimization Methods and Software*, 23(1):5–19, 2008.

Varsha Dani, Sham M Kakade, and Thomas P Hayes. The price of bandit information for online optimization. In *Advances in Neural Information Processing Systems*, pages 345–352, 2007.

Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In *Proceedings of The 21st Conference on Learning Theory (COLT)*, pages 355–366, 2008.

Sanjoy Dasgupta and John Langford. Active learning tutorial. *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009. URL `http://hunch.net/~active_learning/`.

Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. *Journal of Machine Learning Research*, 10:281–299, 2009.

Ravi Ganti and Alexander G. Gray. Upal: Unbiased pool based active learning. In *AISTATS*, volume 22 of *Journal of Machine Learning Research - Proceedings Track*, pages 422–431. JMLR.org, 2012.

Olivier Guédon and Emanuel Milman. Interpolating thin-shell and sharp large-deviation estimates for lsotropic log-concave measures. *Geometric and Functional Analysis*, 21(5): 1043–1068, 2011. ISSN 1016-443X.

László Györfi, Michael Kohler, Adam Krzyzak, and Harro Walk. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

Steve Hanneke. Teaching dimension and the complexity of active learning. In *Proceedings of The 20th Conference on Learning Theory (COLT)*, pages 66–81. Springer-Verlag, 2007.

Elad Hazan. Introduction to online convex optimization. *Manuscript*, 2014. URL `http://ocobook.cs.princeton.edu/OCObook.pdf`.

Elad Hazan, Zohar Karnin, and Raghu Meka. Volumetric spanners: an efficient exploration basis for learning. In Maria-Florina Balcan and Csaba Szepesvári, editors, *COLT*, volume 35 of *Journal of Machine Learning Research - Proceedings Track*, pages 408–422. JMLR.org, 2014.

Martin Henk. Löwner-John ellipsoids. *Documenta Mathematica*, pages 95–106, 2012.

Daniel Hsu, Sham M. Kakade, and Tong Zhang. Random design analysis of ridge regression. *Journal of Machine Learning Research - Proceedings Track*, 23:9.1–9.24, 2012.

F. John. Extremum Problems with Inequalities as Subsidiary Conditions. In K. O. Friedrichs, O. E. Neugebauer, and J. J. Stoker, editors, *Studies and Essays: Courant Anniversary Volume*, pages 187–204. Wiley-Interscience, New York, 1948.

Sham M Kakade, Adam Tauman Kalai, and Katrina Ligett. Playing games with approximation algorithms. *SIAM Journal on Computing*, 39(3):1088–1106, 2009.

Zohar Karnin and Elad Hazan. Hard-margin active linear regression. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 883–891, 2014.

Leonid G Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21(2):307–320, 1996.

László Lovász and Santosh Vempala. The geometry of logconcave functions and sampling algorithms. *Random Structures & Algorithms*, 30(3):307–358, 2007.

Shie Mannor and John N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5:623–648, 2004.

Andrew McCallum and Kamal Nigam. Employing em and pool-based active learning for text classification. In *Proceedings of the 15th International Conference on Machine Learning (ICML)*, pages 350–358, 1998.

M. Rudelson. Random vectors in the isotropic position. *Journal of Functional Analysis*, 164(1):60–72, 1999.

MC Spruill and WJ Studden. A kiefer-wolfowitz theorem in a stochastic process setting. *The Annals of Statistics*, 7(6):1329–1332, 1979.

Chien-Fu Wu. Some algorithmic aspects of the theory of optimal designs. *The Annals of Statistics*, 6(6):1286–1301, 1978.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 928–936, 2003.

Journal of Machine Learning Research 17 (2016) 1-38 Submitted 12/14; Revised 8/15; Published 4/16

Quasi-Monte Carlo Feature Maps for Shift-Invariant Kernels *

Haim Avron† HAIMAV@POST.TAU.AC.IL
School of Mathematical Sciences
Tel Aviv University
Tel Aviv, 69978, Israel

Vikas Sindhwani† SINDHWANI@GOOGLE.COM
Google Research
New York, NY 10011, USA

Jiyan Yang† JIYAN@STANFORD.EDU
Institute for Computational and Mathematical Engineering
Stanford University
Stanford, CA 94305, USA

Michael W. Mahoney MMAHONEY@STAT.BERKELEY.EDU
International Computer Science Institute and Department of Statistics
University of California at Berkeley
Berkeley, CA 94720, USA

Editor: Nando de Freitas

Abstract

We consider the problem of improving the efficiency of randomized Fourier feature maps to accelerate training and testing speed of kernel methods on large data sets. These approximate feature maps arise as Monte Carlo approximations to integral representations of shift-invariant kernel functions (e.g., Gaussian kernel). In this paper, we propose to use *Quasi-Monte Carlo* (QMC) approximations instead, where the relevant integrands are evaluated on a low-discrepancy sequence of points as opposed to random point sets as in the Monte Carlo approach. We derive a new discrepancy measure called *box discrepancy* based on theoretical characterizations of the integration error with respect to a given sequence. We then propose to learn QMC sequences adapted to our setting based on explicit box discrepancy minimization. Our theoretical analyses are complemented with empirical results that demonstrate the effectiveness of classical and adaptive QMC techniques for this problem.

1. Introduction

Kernel methods (Schölkopf and Smola, 2002; Wahba, 1990; Cucker and Smale, 2001) offer a comprehensive suite of mathematically well-founded non-parametric modeling techniques for a wide range of problems in machine learning. These include nonlinear classification, regression, clustering, semi-supervised learning (Belkin et al., 2006), time-series analysis (Parzen, 1970), sequence modeling (Song et al., 2010), dynamical systems (Boots et al., 2013), hypothesis testing (Harchaoui et al., 2013), causal modeling (Zhang et al., 2011) and many more.

∗. A short version of this paper has been presented in ICML 2014.
†. Equal contributors

The central object of kernel methods is a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined on an input domain $\mathcal{X} \subset \mathbb{R}^d$ [1]. The kernel k is (non-uniquely) associated with an embedding of the input space into a high-dimensional Hilbert space $\mathcal{H}$ (with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$) via a feature map, $\Psi : \mathcal{X} \to \mathcal{H}$, such that

$$k(\mathbf{x}, \mathbf{z}) = \langle \Psi(\mathbf{x}), \Psi(\mathbf{z}) \rangle_{\mathcal{H}} .$$

Standard regularized linear statistical models in $\mathcal{H}$ then provide non-linear inference with respect to the original input representation. The algorithmic basis of such constructions are classical Representer Theorems (Wahba, 1990; Schölkopf and Smola, 2002) that guarantee finite-dimensional solutions of associated optimization problems, even if $\mathcal{H}$ is infinite-dimensional.

However, there is a steep price of these elegant generalizations in terms of scalability. Consider, for example, least squares regression given n data points $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ and assume that $n \gg d$. The complexity of linear regression training using standard least squares solvers is $O(nd^2)$, with $O(nd)$ memory requirements, and $O(d)$ prediction speed on a test point. Its kernel-based nonlinear counterpart, however, requires solving a linear system involving the Gram matrix of the kernel function (defined by $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$). In general, this incurs $O(n^3 + n^2 d)$ complexity for training, $O(n^2)$ memory requirements, and $O(nd)$ prediction time for a single test point – none of which are particularly appealing in "big data" settings. Similar conclusions apply to other algorithms such as Kernel PCA.

This is rather unfortunate, since non-parametric models, such as the ones produced by kernel methods, are particularly appealing in a big data settings as they can adapt to the full complexity of the underlying domain, as uncovered by increasing data set sizes. It is well-known that imposing strong structural constraints upfront for the purpose of allowing an efficient solution (in the above example: a linear hypothesis space) often limits, both theoretically and empirically, the potential to deliver value on large amounts of data. Thus, as big data becomes pervasive across a number of application domains, it has become necessary to be able to develop highly scalable algorithms for kernel methods.

Recent years have seen intensive research on improving the scalability of kernel methods; we review some recent progress in the next section. In this paper, we revisit one of the most successful techniques, namely the randomized construction of a family of low-dimensional approximate feature maps proposed by Rahimi and Recht (2008). These randomized feature maps, $\hat{\Psi} : \mathcal{X} \to \mathbb{C}^s$, provide low-distortion approximations for (complex-valued) kernel functions $k : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$:

$$k(\mathbf{x}, \mathbf{z}) \approx \langle \hat{\Psi}(\mathbf{x}), \hat{\Psi}(\mathbf{z}) \rangle_{\mathbb{C}^s} \tag{1}$$

where $\mathbb{C}^s$ denotes the space of s-dimensional complex numbers with the inner product, $\langle \alpha, \beta \rangle_{\mathbb{C}^s} = \sum_{i=1}^s \alpha_i \beta_i^*$, with z^* denoting the conjugate of the complex number z (though Rahimi and Recht (2008) also define real-valued feature maps for real-valued kernels, our technical exposition is simplified by adopting the generality of complex-valued features). The mapping $\hat{\Psi}(\cdot)$ is now applied to each of the data points, to obtain a randomized feature representation of the data. We then apply a simple linear method to these random features. That is, if our data is $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ we learn on $\{(\mathbf{z}_i, y_i)\}_{i=1}^n$ where $\mathbf{z}_i = \hat{\Psi}(\mathbf{x}_i)$. As long as s is sufficiently smaller than n, this leads to more scalable solutions, e.g., for regression we get back to $O(ns^2)$ training and $O(sd)$ prediction time, with $O(ns)$ memory requirements. This technique is immensely successful, and has been used in

1. In fact, $\mathcal{X}$ can be a rather general set. However, in this paper it is restricted to being a subset of $\mathbb{R}^d$.

recent years to obtain state-of-the-art accuracies for some classical data sets (Huang et al., 2014; Dai et al., 2014; Sindhwani and Avron, 2014; Lu et al., 2014).

The starting point of Rahimi and Recht (2008) is a celebrated result that characterizes the class of positive definite functions:

Definition 1 *A function $g : \mathbb{R}^d \mapsto \mathbb{C}$ is a* positive definite function *if for any set of m points, $\mathbf{x}_1 \dots \mathbf{x}_m \in \mathbb{R}^d$, the $m \times m$ matrix $\mathbf{A}$ defined by $\mathbf{A}_{ij} = g(\mathbf{x}_i - \mathbf{x}_j)$ is positive semi-definite.*

Theorem 2 (Bochner (1933)) *A complex-valued function $g : \mathbb{R}^d \mapsto \mathbb{C}$ is positive definite if and only if it is the Fourier Transform of a finite non-negative Borel measure μ on $\mathbb{R}^d$, i.e.,*

$$g(\mathbf{x}) = \hat{\mu}(\mathbf{x}) = \int_{\mathbb{R}^d} e^{-i\mathbf{x}^T\mathbf{w}} d\mu(\mathbf{w}), \quad \forall \mathbf{x} \in \mathbb{R}^d \ .$$

Without loss of generality, we assume henceforth that $\mu(\cdot)$ is a probability measure with associated probability density function $p(\cdot)$.

A kernel function $k : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{C}$ on $\mathbb{R}^d$ is called *shift-invariant* if $k(\mathbf{x}, \mathbf{z}) = g(\mathbf{x} - \mathbf{z})$, for some positive definite function $g : \mathbb{R}^d \mapsto \mathbb{C}$. Bochner's theorem implies that a scaled shift-invariant kernel can therefore be put into one-to-one correspondence with a density $p(\cdot)$ such that,

$$k(\mathbf{x}, \mathbf{z}) = g(\mathbf{x} - \mathbf{z}) = \int_{\mathbb{R}^d} e^{-i(\mathbf{x}-\mathbf{z})^T\mathbf{w}} p(\mathbf{w}) d\mathbf{w} \ . \tag{2}$$

For the most notable member of the shift-invariant family of kernels – the Gaussian kernel:

$$k(\mathbf{x}, \mathbf{z}) = e^{-\frac{\|\mathbf{x}-\mathbf{z}\|_2^2}{2\sigma^2}},$$

the associated density is again Gaussian $\mathcal{N}(0, \sigma^{-2}\mathbf{I}_d)$.

The integral representation of the kernel (2) may be approximated as follows:

$$\begin{aligned} k(\mathbf{x}, \mathbf{z}) &= \int_{\mathbb{R}^d} e^{-i(\mathbf{x}-\mathbf{z})^T\mathbf{w}} p(\mathbf{w}) d\mathbf{w} \\ &\approx \frac{1}{s} \sum_{j=1}^{s} e^{-i(\mathbf{x}-\mathbf{z})^T\mathbf{w}_s} \\ &= \langle \hat{\Psi}_S(\mathbf{x}), \hat{\Psi}_S(\mathbf{z}) \rangle_{\mathbb{C}^s} \ , \end{aligned}$$

through the feature map,

$$\hat{\Psi}_S(\mathbf{x}) = \frac{1}{\sqrt{s}} \left[e^{-i\mathbf{x}^T\mathbf{w}_1} \dots e^{-i\mathbf{x}^T\mathbf{w}_s} \right] \in \mathbb{C}^s \ . \tag{3}$$

The subscript S denotes dependence of the feature map on the sequence $S = \{\mathbf{w}_1, \dots, \mathbf{w}_s\}$.

The goal of this work is to improve the convergence behavior of this approximation, so that a smaller s can be used to get the same quality of approximation to the kernel function. This is motivated by recent work that showed that in order to obtain state-of-the-art accuracy on some important data sets, a very large number of random features is needed (Huang et al., 2014; Sindhwani and Avron, 2014).

Our point of departure from the work of Rahimi and Recht (2008) is the simple observation that when $\mathbf{w}_1, \dots, \mathbf{w}_s$ are are drawn from the distribution defined by the density function $p(\cdot)$, the approximation in (3) may be viewed as a standard *Monte Carlo* (MC) approximation to the integral representation of the kernel. Instead of using plain MC approximation, we propose to use the low-discrepancy properties of *Quasi-Monte Carlo* (QMC) sequences to reduce the integration error in approximations of the form (3). A self-contained overview of Quasi-Monte Carlo techniques for high-dimensional integration problems is provided in Section 2. In Section 3, we describe how QMC techniques apply to our setting.

We then proceed to apply an average case theoretical analysis of the integration error for any given sequence S (Section 4). This bound motivates an optimization problem over the sequence S whose minimizer provides *adaptive QMC* sequences fine tuned to our kernels (Section 5).

Finally, empirical results (Section 6) clearly demonstrate the superiority of QMC techniques over the MC feature maps (Rahimi and Recht, 2008), the correctness of our theoretical analysis and the potential value of adaptive QMC techniques for large-scale kernel methods.

2. Preliminaries

In this section we give the notation that will be used throughout the paper, a summary of related work and an overview of the Quasi-Monte Carlo method.

2.1 Notation

We use i both for subscript and for denoting $\sqrt{-1}$, relying on the context to distinguish between the two. We use $y, z, \dots$ to denote scalars. We use $\mathbf{w}, \mathbf{t}, \mathbf{x} \dots$ to denote vectors, and use w_i to denote the i-th coordinate of vectors $\mathbf{w}$. Furthermore, in a sequence of vectors, we use $\mathbf{w}_i$ to denote the i-th element of the sequence and use w_{ij} to denote the j-th coordinate of vector $\mathbf{w}_i$. Given $\mathbf{x}_1, \dots, \mathbf{x}_n$, the Gram matrix is defined as $\mathbf{K} \in \mathbb{R}^{n \times n}$ where $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j = 1, \dots, n$. We denote the error function by $\mathrm{erf}(\cdot)$, i.e., $\mathrm{erf}(z) = \int_0^z e^{-z^2} dz$ for $z \in \mathbb{C}$; see Weideman (1994) and Mori (1983) for more details.

In "*MC sequence*" we mean points drawn randomly either from the unit cube or certain distribution that will be clear from the text. For "*QMC sequence*" we mean a deterministic sequence designed to reduce the integration error. Typically, it will be a low-discrepancy sequence on the unit cube.

It is also useful to recall the definition of Reproducing Kernel Hilbert Space (RKHS).

Definition 3 (Reproducing Kernel Hilbert Space (Berlinet and Thomas-Agnan, 2004)) *A* reproducing kernel Hilbert space (RKHS) *is a Hilbert Space $\mathcal{H} : \mathcal{X} \to \mathbb{C}$ that possesses a reproducing kernel, i.e., a function $h : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$ for which the following hold for all $\mathbf{x} \in \mathcal{X}$ and $f \in \mathcal{H}$:*

- $h(\mathbf{x}, \cdot) \in \mathcal{H}$
- $\langle f, h(\mathbf{x}, \cdot) \rangle_{\mathcal{H}} = f(\mathbf{x})$ *(*Reproducing Property*)*

Equivalently, RKHSs are Hilbert spaces with bounded, continuous evaluation functionals. Informally, they are Hilbert spaces with the nice property that if two functions $f, g \in \mathcal{H}$ are close in the sense of the distance derived from the norm in $\mathcal{H}$ (i.e., $\|f - g\|_{\mathcal{H}}$ is small), then their values $f(\mathbf{x}), g(\mathbf{x})$ are also close for all $\mathbf{x} \in \mathcal{X}$; in other words, the norm controls the pointwise behavior of functions in $\mathcal{H}$ (Berlinet and Thomas-Agnan, 2004).

2.2 Related Work

In this section we discuss related work on scalable kernel methods. Relevant work on QMC methods is discussed in the next subsection.

Scalability has long been identified as a key challenge associated with deploying kernel methods in practice. One dominant line of work constructs low-rank approximations of the Gram matrix, either using data-oblivious randomized feature maps to approximate the kernel function, or using sampling techniques such as the classical Nyström method (Williams and Seeger, 2001). In its vanilla version, the latter approach - Nyström method - samples points from the data set, computes the columns of the Gram matrix that corresponds to the sampled data points, and uses this partial computation of the Gram matrix to construct an approximation to the entire Gram matrix. More elaborate techniques exist, both randomized and deterministic; see Gittens and Mahoney (2013) for a thorough treatment.

More relevant to our work is the randomized feature mapping approach. Pioneered by the seminal paper of Rahimi and Recht (2008), the core idea is to construct, for a given kernel on a data domain $\mathcal{X}$, a transformation $\hat{\Psi} : \mathcal{X} \to \mathbb{C}^s$ such that $k(\mathbf{x}, \mathbf{z}) \approx \langle \hat{\Psi}(\mathbf{x}), \hat{\Psi}(\mathbf{z}) \rangle_{\mathbb{C}^s}$. Invoking Bochner's theorem, a classical result in harmonic analysis, Rahimi and Recht show how to construct a randomized feature map for shift-invariant kernels, i.e., kernels that can be written $k(\mathbf{x}, \mathbf{z}) = g(\mathbf{x} - \mathbf{y})$ for some positive definite function $g(\cdot)$.

Subsequently, there has been considerable effort given to extending this technique to other classes of kernels. Li et al. (2010) use Bochner's theorem to provide random features to the wider class of group-invariant kernels. Maji and Berg (2009) suggested random features for the intersection kernel $k(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{d} \min(x_i, z_i)$. Vedaldi and Zisserman (2012) developed feature maps for γ-homogeneous kernels. Sreekanth et al. (2010) developed feature maps for generalized RBF kernels $k(\mathbf{x}, \mathbf{z}) = g(D(\mathbf{x}, \mathbf{z})^2)$ where $g(\cdot)$ is a positive definite function, and $D(\cdot, \cdot)$ is a distance metric. Kar and Karnick (2012) suggested feature maps for dot-product kernels. The feature maps are based on the Maclaurin expansion, which is guaranteed to be non-negative due to a classical result of Schoenberg (1942). Pham and Pagh (2013) suggested feature maps for the polynomial kernels. Their construction leverages known techniques from sketching theory. It can also be shown that their feature map is an *oblivious subspace embedding*, and this observation provides stronger theoretical guarantees than point-wise error bounds prevalent in the feature map literature (Avron et al., 2014). By invoking a variant of Bochner's theorem that replaces the Fourier transform with the Laplace transform, Yang et al. (2014) obtained randomized feature maps for semigroup kernels on histograms. We note that while the original feature maps suggested by Rahimi and Recht were randomized, some of the aforementioned maps are deterministic.

Our work is more in-line with recent efforts on scaling up the random features, so that learning and prediction can be done faster. Le et al. (2013) return to the original construction of Rahimi and Recht (2008), and devise a clever distribution of random samples $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_s$ that is structured so that the generation of random features can be done much faster. They showed that only a very limited concession in term of convergence rate is made. Hamid et al. (2014), working on the polynomial kernel, suggest first generating a very large amount of random features, and then applying them a low-distortion embedding based the Fast Johnson-Lindenstruass Transform, so the make the final size of the mapped vector rather small. In contrast, our work tries to design $\mathbf{w}_1, \ldots, \mathbf{w}_s$ so that less features will be necessary to get the same quality of kernel approximation.

Several other scalable approaches for large-scale kernel methods have been suggested over the years, starting from approaches such as chunking and decomposition methods proposed in the early days of SVM optimization literature. Raykar and Duraiswami (2007) use an improved fast Gauss transform for large scale Gaussian Process regression. There are also approaches that are more specific to the objective function at hand, e.g., Keerthi et al. (2006) builds a kernel expansion greedily to optimize the SVM objective function. Another well known approach is the Core Vector Machines (Tsang et al., 2005) which draws on approximation algorithms from computational geometry to scale up a class of kernel methods that can be reformulated in terms of the minimum enclosing ball problem.

For a broader discussion of these methods, and others, see Bottou et al. (2007).

2.3 Quasi-Monte Carlo Techniques: an Overview

In this section we provide a self-contained overview of Quasi-Monte Carlo (QMC) techniques. For brevity, we restrict our discussion to background that is necessary for understanding subsequent sections. We refer the interested reader to the excellent reviews by Caflisch (1998) and Dick et al. (2013), and the recent book Leobacher and Pillichschammer (2014) for a much more detailed exposition.

Consider the task of computing an approximation of the following integral

$$I_d[f] = \int_{[0,1]^d} f(\mathbf{x}) d\mathbf{x} \,. \tag{4}$$

One can observe that if $\mathbf{x}$ is a random vector uniformly distributed over $[0,1]^d$ then $I_d[f] = \mathbb{E}[f(\mathbf{x})]$. An empirical approximation to the expected value can be computed by drawing a random point set $S = \{\mathbf{w}_1, \ldots, \mathbf{w}_s\}$ independently from $[0,1]^d$, and computing:

$$I_S[f] = \frac{1}{s} \sum_{\mathbf{w} \in S} f(\mathbf{w}) \,.$$

This is the Monte Carlo (MC) method.

Define the integration error with respect to the point set S as,

$$\epsilon_S[f] = |I_d(f) - I_S(f)| \,.$$

When S is drawn randomly, the Central Limit Theorem asserts that if $s = |S|$ is large enough then $\epsilon_S[f] \approx \sigma[f] s^{-1/2} \boldsymbol{\nu}$ where $\boldsymbol{\nu}$ is a standard normal random variable, and $\sigma[f]$ is the square-root of the variance of f; that is,

$$\sigma^2[f] = \int_{[0,1]^d} (f(\mathbf{x}) - I_d(f))^2 \, d\mathbf{x} \,.$$

In other words, the root mean square error of the Monte Carlo method is,

$$\left(\mathbb{E}_S\left[\epsilon_S[f]^2\right]\right)^{1/2} \approx \sigma[f] s^{-1/2}. \tag{5}$$

Therefore, the Monte Carlo method converges at a rate of $O(s^{-1/2})$.

The aim of QMC methods is to improve the convergence rate by using a deterministic *low-discrepancy sequence* to construct S, instead of randomly sampling points. The underlying intuition

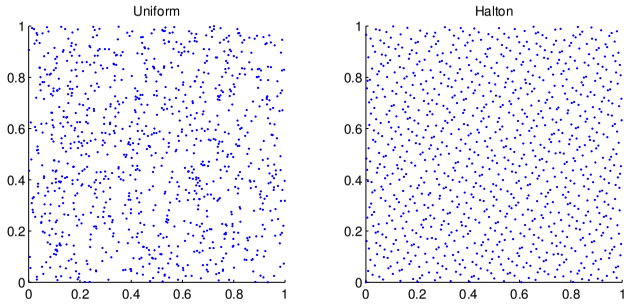


Figure 1: Comparison of MC and QMC sequences.

is illustrated in Figure 1, where we plot a set of 1000 two-dimensional random points (left graph), and a set of 1000 two-dimensional points from a quasi-random sequence (Halton sequence; right graph). In the random sequence we see that there is an undesired clustering of points, and as a consequence empty spaces. Clusters add little to the approximation of the integral in those regions, while in the empty spaces the integrand is not sampled. This lack of uniformity is due to the fact that Monte Carlo samples are independent of each other. By carefully designing a sequence of correlated points to avoid such clustering effects, QMC attempts to avoid this phenomena, and thus provide faster convergence to the integral.

The theoretical apparatus for designing such sequences are inequalities of the form

$$\epsilon_S(f) \leq D(S)V(f) \,,$$

in which $V(f)$ is a measure of the variation or difficulty of integrating $f(\cdot)$ and $D(S)$ is a sequence-dependent term that typically measures the *discrepancy*, or degree of deviation from uniformity, of the sequence S. For example, the expected Monte Carlo integration error decouples into a variance term, and $s^{-1/2}$ as in (5).

A prototypical inequality of this sort is the following remarkable and classical result:

Theorem 4 (Koksma-Hlawka inequality) *For any function f with bounded variation, and sequence $S = \{\mathbf{w}_1, \ldots, \mathbf{w}_s\}$, the integration error is bounded above as follows,*

$$\epsilon_S[f] \leq D^\star(S)V_{HK}[f] \,,$$

where V_{HK} is the Hardy-Krause variation of f *(see Niederreiter (1992)), which is defined in terms of the following partial derivatives,*

$$V_{HK}[f] = \sum_{I \subset [d], I \neq \emptyset} \int_{[0,1]^{|I|}} \left| \frac{\partial f}{\partial \mathbf{u}_I}\Big|_{u_j=1, j \notin I} \right| d\mathbf{u}_I \,, \tag{6}$$

and $D^\star$ is the star discrepancy *defined by*

$$D^\star(S) = \sup_{\mathbf{x} \in [0,1]^d} |\mathrm{disr}_S(\mathbf{x})| \,,$$

where disr_S *is the* local discrepancy function

$$\mathrm{disr}_S(\mathbf{x}) = \mathrm{Vol}(J_\mathbf{x}) - \frac{|\{i \,:\, \mathbf{w}_i \in J_\mathbf{x}\}|}{s}$$

with $J_\mathbf{x} = [0, x_1) \times [0, x_2) \times \cdots \times [0, x_d)$ *with* $\mathrm{Vol}(J_\mathbf{x}) = \prod_{j=1}^d x_j$.

Given $\mathbf{x}$, the second term in $\mathrm{disr}_S(\mathbf{x})$ is an estimate of the volume of $J_\mathbf{x}$, which will be accurate if the points in S are uniform enough. $D^\star(S)$ measures the maximum difference between the actual volume of the subregion $J_\mathbf{x}$ and its estimate for all $\mathbf{x}$ in $[0, 1]^d$.

An infinite sequence $\mathbf{w}_1, \mathbf{w}_2, \ldots$ is defined to be a *low-discrepancy sequence* if, as a function of s, $D^\star(\{\mathbf{w}_1, \ldots, \mathbf{w}_s\}) = O((\log s)^d/s)$. Several constructions are know to be low-discrepancy sequences. One notable example is the *Halton sequences*, which are defined as follows. Let $p_1, \ldots, p_d$ be the first d prime numbers. The Halton sequence $\mathbf{w}_1, \mathbf{w}_2, \ldots$ of dimension d is defined by

$$\mathbf{w}_i = (\phi_{p_1}(i), \ldots, \phi_{p_d}(i))$$

where for integers $i \geq 0$ and $b \geq 2$ we have

$$\phi_b(i) = \sum_{a=1}^{\infty} i_a b^{-a}$$

in which $i_0, i_1, \cdots \in \{0, 1, \ldots, b-1\}$ is given by the unique decomposition

$$i = \sum_{a=1}^{\infty} i_a b^{a-1} \,.$$

It is outside the scope of this paper to describe all these constructions in detail. However we mention that in addition to the Halton sequences, other notable members are *Sobol' sequences*, *Faure sequences*, *Niederreiter sequences*, and more (see Dick et al. (2013), Section 2). We also mention that it is conjectured that the $O((\log s)^d/s)$ rate for star discrepancy decay is optimal.

The classical QMC theory, which is based on the Koksma-Hlawka inequality and low discrepancy sequences, thus achieves a convergence rate of $O((\log s)^d/s)$. While this is asymptotically superior to $O(s^{-1/2})$ for a fixed d, it requires s to be exponential in d for the improvement to manifest. As such, in the past QMC methods were dismissed as unsuitable for very high-dimensional integration.

However, several authors noticed that QMC methods perform better than MC even for very high-dimensional integration (Sloan and Wozniakowski, 1998; Dick et al., 2013).[2] Contemporary QMC literature explains and expands on these empirical observations, by leveraging the structure of the space in which the integrand function lives, to derive more refined bounds and discrepancy measures, even when classical measures of variation such as (6) are unbounded. This literature has evolved along at least two directions: one, where worst-case analysis is provided under the assumption that the integrands live in a Reproducing Kernel Hilbert Space (RKHS) of sufficiently smooth and well-behaved functions (see Dick et al. (2013), Section 3) and second, where the analysis is done in terms of *average-case* error, under an assumed probability distribution over the integrands, instead of worst-case error (Wozniakowski, 1991; Traub and Wozniakowski, 1994). We refrain from more details, as these are essentially the paths that the analysis in Section 4 follows for our specific setting.

2. Also see: "On the unreasonable effectiveness of QMC", I.H. Sloan `https://mcqmc.mimuw.edu.pl/Presentations/sloan.pdf`

Algorithm 1 Quasi-Random Fourier Features

Input: Shift-invariant kernel k, size s.

Output: Feature map $\hat{\Psi}(\mathbf{x}) : \mathbb{R}^d \mapsto \mathbb{C}^s$.

1: Find p, the inverse Fourier transform of k.
2: Generate a low discrepancy sequence $\mathbf{t}_1, \ldots, \mathbf{t}_s$.
3: Transform the sequence: $\mathbf{w}_i = \Phi^{-1}(\mathbf{t}_i)$ by (7).
4: Set $\hat{\Psi}(\mathbf{x}) = \sqrt{\frac{1}{s}}\left[e^{-i\mathbf{x}^T\mathbf{w}_1}, \ldots, e^{-i\mathbf{x}^T\mathbf{w}_s}\right]$.

3. QMC Feature Maps: Our Algorithm

We assume that the density function in (2) can be written as $p(\mathbf{x}) = \prod_{j=1}^d p_j(x_j)$, where $p_j(\cdot)$ is a univariate density function. The density functions associated to many shift-invariant kernels, e.g., Gaussian, Laplacian and Cauchy, admits such a form.

The QMC method is generally applicable to integrals over a unit cube. So typically integrals of the form (2) are handled by first generating a low discrepancy sequence $\mathbf{t}_1, \ldots, \mathbf{t}_s \in [0,1]^d$, and transforming it into a sequence $\mathbf{w}_1, \ldots, \mathbf{w}_s$ in $\mathbb{R}^d$, instead of drawing the elements of the sequence from $p(\cdot)$ as in the MC method.

To convert (2) to an integral over the unit cube, a simple change of variables suffices. For $\mathbf{t} \in \mathbb{R}^d$, define

$$\Phi^{-1}(\mathbf{t}) = \left(\Phi_1^{-1}(t_1), \ldots, \Phi_d^{-1}(t_d)\right) \in \mathbb{R}^d\,, \tag{7}$$

where $\Phi_j(\cdot)$ is the cumulative distribution function (CDF) of $p_j(\cdot)$, for $j = 1, \ldots, d$. By setting $\mathbf{w} = \Phi^{-1}(\mathbf{t})$, then (2) can be equivalently written as

$$\int_{\mathbb{R}^d} e^{-i(\mathbf{x}-\mathbf{z})^T\mathbf{w}} p(\mathbf{w}) d\mathbf{w} = \int_{[0,1]^d} e^{-i(\mathbf{x}-\mathbf{z})^T\Phi^{-1}(\mathbf{t})} d\mathbf{t}\,.$$

Thus, a low discrepancy sequence $\mathbf{t}_1, \ldots, \mathbf{t}_s \in [0,1]^d$ can be transformed using $\mathbf{w}_i = \Phi^{-1}(\mathbf{t}_i)$, which is then plugged into (3) to yield the QMC feature map. This simple procedure is summarized in Algorithm 1. QMC feature maps are analyzed in the next section.

4. Theoretical Analysis and Average Case Error Bounds

The proofs for assertions made in this section and the next can be found in the Appendix.

The goal of this section is to develop a framework for analyzing the approximation quality of the QMC feature maps described in the previous section (Algorithm 1). We need to develop such a framework since the classical Koksma-Hlawka inequality cannot be applied to our setting, as the following proposition shows:

Proposition 5 *For any $p(\mathbf{x}) = \prod_{j=1}^d p_j(x_j)$, where $p_j(\cdot)$ is a univariate density function, let*

$$\Phi^{-1}(\mathbf{t}) = \left(\Phi_1^{-1}(t_1), \ldots, \Phi_d^{-1}(t_d)\right)\,.$$

For a fixed $\mathbf{u} \in \mathbb{R}^d$, consider $f_{\mathbf{u}}(\mathbf{t}) = e^{-i\mathbf{u}^T\Phi^{-1}(\mathbf{t})}$, $\mathbf{t} \in [0,1]^d$. The Hardy-Krause variation of $f_{\mathbf{u}}(\cdot)$ is unbounded. That is, one of the integrals in the sum (6) *is unbounded.*

Our framework is based on a new discrepancy measure, *box discrepancy*, that characterizes integration error for the set of integrals defined with respect to the underlying data domain. Throughout this section we use the convention that $S = \{\mathbf{w}_1, \ldots, \mathbf{w}_s\}$, and the notation $\bar{\mathcal{X}} = \{\mathbf{x} - \mathbf{z} \mid \mathbf{x}, \mathbf{z} \in \mathcal{X}\}$.

Given a probability density function $p(\cdot)$ and S, we define the integration error $\epsilon_{S,p}[f]$ of a function $f(\cdot)$ with respect to $p(\cdot)$ and the s samples as,

$$\epsilon_{S,p}[f] = \left| \int_{\mathbb{R}^d} f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} - \frac{1}{s}\sum_{i=1}^s f(\mathbf{w}_i) \right|\,.$$

We are interested in characterizing the behavior of $\epsilon_{S,p}[f]$ on $f \in \mathcal{F}_{\bar{\mathcal{X}}}$ where

$$\mathcal{F}_{\bar{\mathcal{X}}} = \left\{ f_{\mathbf{u}}(\mathbf{x}) = e^{-i\mathbf{u}^T\mathbf{x}},\ \mathbf{u} \in \bar{\mathcal{X}} \right\}\,.$$

As is common in modern QMC analysis (Leobacher and Pillichshammer, 2014; Dick et al., 2013), our analysis is based on setting up a Reproducing Kernel Hilbert Space of "nice" functions that is related to integrands that we are interested in, and using properties of the RKHS to derive bounds on the integration error. In particular, the integration error of integrands in an RKHS can be bounded using the following proposition.

Proposition 6 (Integration Error in an RKHS) *Let $\mathcal{H}$ be an RKHS with kernel $h(\cdot,\cdot)$. Assume that $\kappa = \sup_{\mathbf{x}\in\mathbb{R}^d} h(\mathbf{x},\mathbf{x}) < \infty$. Then, for all $f \in \mathcal{H}$ we have,*

$$\epsilon_{S,p}[f] \le \|f\|_{\mathcal{H}} D_{h,p}(S)\,, \tag{8}$$

where

$$\begin{aligned} D_{h,p}(S)^2 &= \left\| \int_{\mathbb{R}^d} h(\omega,\cdot)p(\omega)d\omega - \frac{1}{s}\sum_{l=1}^s h(\mathbf{w}_l,\cdot) \right\|_{\mathcal{H}}^2 \\ &= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\omega,\phi)p(\omega)p(\phi)d\omega d\phi - \frac{2}{s}\sum_{l=1}^s \int_{\mathbb{R}^d} h(\mathbf{w}_l,\omega)p(\omega)d\omega \\ &\quad + \frac{1}{s^2}\sum_{l=1}^s\sum_{j=1}^s h(\mathbf{w}_l,\mathbf{w}_j)\,. \end{aligned} \tag{9}$$

Remark 7 *In the theory of RKHS embeddings of probability distributions (Smola et al., 2007; Sriperumbudur et al., 2010), the function*

$$\mu_{h,p}(\mathbf{x}) = \int_{\mathbb{R}^d} h(\omega,\mathbf{x})p(\omega)d\omega$$

is known as the kernel mean *embedding of $p(\cdot)$. The function*

$$\hat{\mu}_{h,p,S}(\mathbf{x}) = \frac{1}{s}\sum_{l=1}^s h(\mathbf{w}_l,\mathbf{x})$$

is then the empirical mean map.

The RKHS we use is as follows. For a vector $\mathbf{b} \in \mathbb{R}^d$, let us define $\square\mathbf{b} = \{\mathbf{u} \in \mathbb{R}^d \mid |u_j| \leq b_j\}$. Let

$$\mathcal{F}_{\square\mathbf{b}} = \left\{ f_{\mathbf{u}}(\mathbf{x}) = e^{-i\mathbf{u}^T\mathbf{x}},\ \mathbf{u} \in \square\mathbf{b} \right\} ,$$

and consider the space of functions that admit an integral representation over $\mathcal{F}_{\square\mathbf{b}}$ of the form

$$f(\mathbf{x}) = \int_{\mathbf{u}\in\square\mathbf{b}} \hat{f}(\mathbf{u}) e^{-i\mathbf{u}^T\mathbf{x}} d\mathbf{u} \text{ where } \hat{f}(\mathbf{u}) \in L_2(\square\mathbf{b}) . \tag{10}$$

This space is associated with bandlimited functions, i.e., functions with compactly-supported inverse Fourier transforms, which are of fundamental importance in the Shannon-Nyquist sampling theory. Under a natural choice of inner product, these spaces are called *Paley-Wiener spaces* and they constitute an RKHS (Berlinet and Thomas-Agnan, 2004; Yao, 1967; Peloso, 2011).

Proposition 8 (Kernel of Paley-Wiener RKHS) *By $PW_{\mathbf{b}}$, denote the space of functions which admit the representation in* (10)*, with the inner product $\langle f, g\rangle_{PW_{\mathbf{b}}} = (2\pi)^{2d}\langle \hat{f}, \hat{g}\rangle_{L_2(\square\mathbf{b})}$. $PW_{\mathbf{b}}$ is an RKHS with kernel function,*

$$\operatorname{sinc}_{\mathbf{b}}(\mathbf{u}, \mathbf{v}) = \pi^{-d} \prod_{j=1}^{d} \frac{\sin\left(b_j(u_j - v_j)\right)}{u_j - v_j} .$$

For notational convenience, in the above we define $\sin(b\cdot 0)/0$ to be b. Furthermore, $\langle f, g\rangle_{PW_{\mathbf{b}}} = \langle f, g\rangle_{L_2(\square\mathbf{b})}$.

If $b_j = \sup_{\mathbf{u}\in\bar{\mathcal{X}}} |u_j|$ then $\bar{\mathcal{X}} \subset \square\mathbf{b}$, so $F_{\bar{\mathcal{X}}} \subset F_{\square\mathbf{b}}$. Since we wish to bound the integration error on functions in $F_{\bar{\mathcal{X}}}$, it suffices to bound the integration error on $\mathcal{F}_{\square\mathbf{b}}$. Unfortunately, while $\mathcal{F}_{\square\mathbf{b}}$ defines $PW_{\mathbf{b}}$, the functions in it, being not square integrable, are *not* members of $PW_{\mathbf{b}}$, so analyzing the integration error in $PW_{\mathbf{b}}$ do not directly apply to them. However, damped approximations of $f_{\mathbf{u}}(\cdot)$ of the form $\tilde{f}_{\mathbf{u}}(\mathbf{x}) = e^{-i\mathbf{u}^T\mathbf{x}} \operatorname{sinc}(T\mathbf{x})$ are members of $PW_{\mathbf{b}}$ with $\|\tilde{f}\|_{PW_{\mathbf{b}}} = \frac{1}{\sqrt{T}}$. Hence, we expect the analysis of the integration error in $PW_{\mathbf{b}}$ to provide provide a discrepancy measure for integrating functions in $\mathcal{F}_{\square\mathbf{b}}$.

For $PW_{\mathbf{b}}$ the discrepancy measure $D_{h,S}$ in Proposition 6 can be written explicitly.

Theorem 9 (Discrepancy in $PW_{\mathbf{b}}$) *Suppose that $p(\cdot)$ is a probability density function, and that we can write $p(\mathbf{x}) = \prod_{j=1}^{d} p_j(x_j)$ where each $p_j(\cdot)$ is a univariate probability density function as well. Let $\varphi_j(\cdot)$ be the characteristic function associated with $p_j(\cdot)$. Then,*

$$\begin{aligned} D_{\operatorname{sinc}_{\mathbf{b}},p}(S)^2 \;=\; & \pi^{-d} \prod_{j=1}^{d} \int_{-b_j}^{b_j} |\varphi_j(\beta)|^2 d\beta - \\ & \frac{2(2\pi)^{-d}}{s} \sum_{l=1}^{s} \prod_{j=1}^{d} \int_{-b_j}^{b_j} \varphi_j(\beta) e^{iw_{lj}\beta} d\beta + \\ & \frac{1}{s^2} \sum_{l=1}^{s}\sum_{j=1}^{s} \operatorname{sinc}_{\mathbf{b}}(\mathbf{w}_l, \mathbf{w}_j) . \end{aligned} \tag{11}$$

This naturally leads to the definition of the *box discrepancy*, analogous to the star discrepancy described in Theorem 4.

Definition 10 (Box Discrepancy) *The box discrepancy of a sequence S with respect to $p(\cdot)$ is defined as,*

$$D_p^{\square\mathbf{b}}(S) = D_{\operatorname{sinc}_{\mathbf{b}},p}(S) .$$

For notational convenience, we generally omit the $\mathbf{b}$ from $D_p^{\square\mathbf{b}}(S)$ as long as it is clear from the context.

The worse-case integration error bound for Paley-Wiener spaces is stated in the following as a corollary of Proposition 6. As explained earlier, this result not yet apply to functions in $\mathcal{F}_{\square\mathbf{b}}$ because these functions are not part of $PW_{\mathbf{b}}$. Nevertheless, we state it here for completeness.

Corollary 11 (Integration Error in $PW_{\mathbf{b}}$) *For $f \in PW_{\mathbf{b}}$ we have*

$$\epsilon_{S,p}[f] \leq \|f\|_{PW_{\mathbf{b}}} D_p^{\square}(S) .$$

Our main result shows that the expected square error of an integrand drawn from a uniform distribution over $\mathcal{F}_{\square\mathbf{b}}$ is proportional to the square discrepancy measure $D_p^{\square}(S)$. This result is in the spirit of similar average case analysis in the QMC literature (Wozniakowski, 1991; Traub and Wozniakowski, 1994).

Theorem 12 (Average Case Error) *Let $\mathcal{U}(\mathcal{F}_{\square\mathbf{b}})$ denote the uniform distribution on $\mathcal{F}_{\square\mathbf{b}}$. That is, $f \sim \mathcal{U}(\mathcal{F}_{\square\mathbf{b}})$ denotes $f = f_{\mathbf{u}}$ where $f_{\mathbf{u}}(\mathbf{x}) = e^{-i\mathbf{u}^T\mathbf{x}}$ and $\mathbf{u}$ is randomly drawn from a uniform distribution on $\square\mathbf{b}$. We have,*

$$\mathbb{E}_{f\sim\mathcal{U}(\mathcal{F}_{\square\mathbf{b}})}\left[\epsilon_{S,p}[f]^2\right] = \frac{\pi^d}{\prod_{j=1}^{d} b_j} D_p^{\square}(S)^2 .$$

We now give an explicit formula for $D_p^{\square}(S)$ for the case that $p(\cdot)$ is the density function of the multivariate Gaussian distribution with zero mean and independent components. This is an important special case since this is the density function that is relevant for the Gaussian kernel.

Corollary 13 (Discrepancy for Gaussian Distribution) *Let $p(\cdot)$ be the d-dimensional multivariate Gaussian density function with zero mean and covariance matrix equal to $\operatorname{diag}(\sigma_1^{-2}, \ldots, \sigma_d^{-2})$. We have,*

$$\begin{aligned} D_p^{\square}(S)^2 \;=\; & \frac{1}{s^2} \sum_{l=1}^{s}\sum_{j=1}^{s} \operatorname{sinc}_{\mathbf{b}}(\mathbf{w}_l, \mathbf{w}_j) - \\ & \frac{2}{s} \sum_{l=1}^{s} \prod_{j=1}^{d} c_{lj} \operatorname{Re}\left(\operatorname{erf}\left(\frac{b_j}{\sigma_j\sqrt{2}} - i\frac{\sigma_j w_{lj}}{\sqrt{2}} \right) \right) + \\ & + \prod_{j=1}^{d} \frac{\sigma_j}{2\sqrt{\pi}} \operatorname{erf}\left(\frac{b_j}{\sigma_j}\right) , \end{aligned} \tag{12}$$

where

$$c_{lj} = \left(\frac{\sigma_j}{\sqrt{2\pi}}\right) e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} .$$

Intuitively, the box discrepancy of the Gaussian kernel can be interpreted as follows. The function $\mathrm{sinc}(x) = \sin(x)/x$ achieves its maximum at $x = 0$ and minimizes at discrete values of x decaying to 0 as $|x|$ goes to ∞. Hence the first term in (12) tends to be minimized when the pairwise distance between $\mathbf{w}_j$ are sufficiently separated. Due to the shape of cumulative distribution function of Gaussian distribution, the values of $\mathbf{t}_j = \Phi(\mathbf{w}_j)$ $(j = 1, \ldots, s)$ are driven to be close to the boundary of the unit cube. As for second term, the original expression is - $\frac{2}{s}\sum_{l=1}^{s}\int_{\mathbb{R}^d} h(\mathbf{w}_l, \omega)p(\omega)d\omega$. This term encourages the sequence $\{\mathbf{w}_l\}$ to mimic samples from $p(\omega)$. Since $p(\omega)$ concentrates its mass around $\omega = 0$, the $\mathbf{w}_j$ also concentrates around 0 to maximize the integral and therefore the values of $\mathbf{t}_j = \Phi(\mathbf{w}_j)$ $(j = 1, \ldots, s)$ are driven closer to the center of the unit cube. Sequences with low box discrepancy therefore optimize a tradeoff between these competing terms.

Two other shift-invariant kernel that have been mentioned in the machine learning literature is the Laplacian kernel (Rahimi and Recht, 2008) and Matern kernel (Le et al., 2013). The distribution associated with the Laplacian kernel can be written as a product $p(\mathbf{x}) = \prod_{j=1}^{d} p_j(x_j)$, where $p_j(\cdot)$ is density associated with the Cauchy distribution. The characteristic function is simple ($\phi_j(\beta) = e^{-|\beta|/\sigma_j}$) so analytic formulas like (12) can be derived. The distribution associated with the Matern kernel, on the other hand, is the multivariate t-distribution, which cannot be written as a product $p(\mathbf{x}) = \prod_{j=1}^{d} p_j(x_j)$, so the presented theory does not apply to it.

Discrepancy of Monte-Carlo Sequences.

We now derive an expression for the expected discrepancy of Monte-Carlo sequences, and show that it decays as $O(s^{-1/2})$. This is useful since via an averaging argument we are guaranteed that there exists sets for which the discrepancy behaves $O(s^{-1/2})$.

Corollary 14 *Suppose $\mathbf{t}_1, \ldots, \mathbf{t}_s$ are chosen uniformly from $[0,1]^d$. Let $\mathbf{w}_i = \Phi^{-1}(\mathbf{t}_i)$, for $i = 1, \ldots, s$. Assume that $\kappa = \sup_{\mathbf{x}\in\mathbb{R}^d} h(\mathbf{x}, \mathbf{x}) < \infty$. Then*

$$\mathbb{E}\left[D_{h,p}(S)^2\right] = \frac{1}{s}\int_{\mathbb{R}^d} h(\omega,\omega)p(\omega)d\omega - \frac{1}{s}\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\omega,\phi)p(\omega)p(\phi)d\omega d\phi\ .$$

Again, we can derive specific formulas for the Gaussian density. The following is straightforward from Corollary 14. We omit the proof.

Corollary 15 *Let $p(\cdot)$ be the d-dimensional multivariate Gaussian density function with zero mean and covariance matrix equal to $\mathrm{diag}(\sigma_1^{-2}, \ldots, \sigma_d^{-2})$. Suppose $\mathbf{t}_1, \ldots, \mathbf{t}_s$ are chosen uniformly from $[0,1]^d$. Let $\mathbf{w}_i = \Phi^{-1}(\mathbf{t}_i)$, for $i = 1, \ldots, s$. Then,*

$$\mathbb{E}\left[D_p^{\square}(S)^2\right] = \frac{1}{s}\left(\pi^{-d}\prod_{j=1}^{d} b_j - \prod_{j=1}^{d}\frac{\sigma_j}{2\sqrt{\pi}}\,\mathrm{erf}\left(\frac{b_j}{\sigma_j}\right)\right). \tag{13}$$

5. Learning Adaptive QMC Sequences

For simplicity, in this section we assume that $p(\cdot)$ is the density function of Gaussian distribution with zero mean. We also omit the subscript p from $D_p^{\square}$. Similar analysis and equations can be derived for other density functions.

Error characterization via discrepancy measures like (12) is typically used in the QMC literature to prescribe sequences whose discrepancy behaves favorably. It is clear that for the box discrepancy,

a meticulous design is needed for a high quality sequence and we leave this to future work. Instead, in this work, we use the fact that unlike the star discrepancy (4), the box discrepancy is a smooth function of the sequence with a closed-form formula. This allows us to both evaluate various candidate sequences, and select the one with the lowest discrepancy, as well as to *adaptively learn* a QMC sequence that is specialized for our problem setting via numerical optimization. The basis is the following proposition, which gives an expression for the gradient of $D^{\square}(S)$.

Proposition 16 (Gradient of Box Discrepancy) *Define the following scalar functions and variables,*

$$\begin{aligned}
\mathrm{sinc}'(z) &= \frac{\cos(z)}{z} - \frac{\sin(z)}{z^2},\quad \mathrm{sinc}'_b(z) = \frac{b}{\pi}\mathrm{sinc}'(bz)\,;\\
c_j &= \left(\frac{\sigma_j}{\sqrt{2\pi}}\right), j = 1, \ldots, d\,;\\
g_j(x) &= c_j e^{-\frac{\sigma_j^2}{2}x^2}\,\mathrm{Re}\left(\mathrm{erf}\left[\frac{b_j}{\sigma_j\sqrt{2}} - i\frac{\sigma_j x}{\sqrt{2}}\right]\right)\,;\\
g'_j(x) &= -\sigma_j^2 x g_j(x) + \sqrt{\frac{2}{\pi}}c_j\sigma_j e^{-\frac{b_j^2}{2\sigma_j^2}}\sin(b_j x)\,.
\end{aligned}$$

In the above ,we define $\mathrm{sinc}'(0)$ to be 0. Then, the elements of the gradient vector of $D^{\square}$ are given by,

$$\begin{aligned}
\frac{\partial D^{\square}}{\partial w_{lj}} &= \frac{2}{s^2}\sum_{\substack{m=1\\ m\neq l}}^{s}\left(b_j\,\mathrm{sinc}'_{b_j}(w_{lj}, w_{mj})\prod_{q\neq j}\mathrm{sinc}_{b_q}(w_{lq}, w_{mq})\right) -\\
&\quad \frac{2}{s}g'_j(w_{lj})\left(\prod_{q\neq j} g_q(w_{lq})\right)\,.
\end{aligned} \tag{14}$$

We explore two possible approaches for finding sequences based on optimizing the box discrepancy, namely *global optimization* and *greedy optimization*. The latter is closely connected to *herding* algorithms (Welling, 2009).

5.1 Global Adaptive Sequences

The task is posed in terms minimization of the box discrepancy function (12) over the space of sequences of s vectors in $\mathbb{R}^d$:

$$S^* = \arg\min_{S=(\mathbf{w}_1\ldots\mathbf{w}_s)\in\mathbb{R}^{ds}} D^{\square}(S)\,.$$

The gradient can be plugged into any first order numerical solver for non-convex optimization. We use non-linear conjugate gradient in our experiments (Section 6.2).

The above learning mechanism can be extended in various directions. For example, QMC sequences for n-point rank-one Lattice Rules (Dick et al., 2013) are integral fractions of a lattice defined by a single generating vector $\mathbf{v}$. This generating vector may be learnt via local minimization of the box discrepancy.

5.2 Greedy Adaptive Sequences

Starting with $S_0 = \emptyset$, for $t \geq 1$, let $S_t = \{\mathbf{w}_1, \ldots, \mathbf{w}_t\}$. At step $t+1$, we solve the following optimization problem,

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} D^{\square}(S_t \cup \{\mathbf{w}\}) \,. \tag{15}$$

Set $S_{t+1} = S_t \cup \{w_{t+1}\}$ and repeat the above procedure. The gradient of the above objective is also given in (14). Again, we use non-linear conjugate gradient in our experiments (Section 6.2).

The greedy adaptive procedure is closely related to the herding algorithm, recently presented by Welling (2009). Applying the herding algorithm to $PW_{\mathbf{b}}$ and $p(\cdot)$, and using our notation, the points $\mathbf{w}_1, \mathbf{w}_2, \ldots$ are generated using the following iteration

$$\begin{aligned} \mathbf{w}_{t+1} &\in \arg\max_{\mathbf{w}\in\mathbb{R}^d} \langle \mathbf{z}_t(\cdot), h(\mathbf{w},\cdot)\rangle_{PW_{\mathbf{b}}} \\ \mathbf{z}_{t+1}(\mathbf{x}) &\equiv \mathbf{z}_t(\mathbf{x}) + \mu_{h,p}(x) - h(\mathbf{w},\mathbf{x}) \,. \end{aligned}$$

In the above, $\mathbf{z}_0, \mathbf{z}_1, \ldots$ is a series of functions in $PW_{\mathbf{b}}$. The literature is not specific on the initial value of $\mathbf{z}_0$, with both $\mathbf{z}_0 = 0$ and $\mathbf{z}_0 = \mu_{h,p}$ suggested. Either way, it is always the case that $\mathbf{z}_t = \mathbf{z}_0 + t(\mu_{h,p} - \hat{\mu}_{h,p,S_t})$ where $S_t = \{\mathbf{w}_1, \ldots, \mathbf{w}_t\}$.

Chen et al. (2010) showed that under some additional assumptions, the herding algorithm, when applied to a RKHS $\mathcal{H}$, greedily minimizes $\|\mu_{h,p} - \hat{\mu}_{h,p,S_t}\|^2_{\mathcal{H}}$, which, recall, is equal to $D_{h,p}(S_t)$. Thus, under certain assumptions, herding and (15) are equivalent. Chen et al. (2010) also showed that under certain restrictions on the RKHS, herding will reduce the discrepancy in a ratio of $O(1/t)$. However, it is unclear whether those restrictions hold for $PW_{\mathbf{b}}$ and $p(\cdot)$. Indeed, Bach et al. (2012) recently shown that these restrictions never hold for infinite-dimensional RKHS, as long as the domain is compact. This result does not immediately apply to our case since $\mathbb{R}^d$ is not compact.

5.3 Weighted Sequences

Classically, Monte-Carlo and Quasi-Monte Carlo approximations of integrals are unweighted, or more precisely, have a uniform weights. However, it is quite plausible to weight the approximations, i.e. approximate $I_d[f] = \int_{[0,1]^d} f(\mathbf{x})d\mathbf{x}$ using

$$I_{S,\Xi}[f] = \sum_{i=1}^{n} \xi_i f(\mathbf{w}_i) \,, \tag{16}$$

where $\Xi = \{\xi_1, \ldots, \xi_s\} \subset \mathbb{R}$ is a set of weights. This lead to the feature map

$$\hat{\Psi}_S(\mathbf{x}) = \left[\sqrt{\xi_1}e^{-i\mathbf{x}^T\mathbf{w}_1} \ldots \sqrt{\xi_s}e^{-i\mathbf{x}^T\mathbf{w}_s}\right] \,.$$

This construction requires $\xi_i \geq 0$ for $i = 1, \ldots, s$, although we note that (16) itself does not preclude negative weights. We do not require the weights to be normalized, that is it is possible that $\sum_{i=1}^{s} \xi_i \neq 1$.

One can easily generalize the result of the previous section to derive the following discrepancy measure that takes into consideration the weights

$$\begin{aligned} D_p^{\square\mathbf{b}}(S,\Xi)^2 &= \pi^{-d}\prod_{j=1}^{d}\int_{-b_j}^{b_j} |\varphi_j(\beta)|^2 d\beta - \\ &\quad 2(2\pi)^{-d}\sum_{l=1}^{s}\xi_l\prod_{j=1}^{d}\int_{-b_j}^{b_j}\varphi_j(\beta)e^{iw_{lj}\beta}d\beta + \\ &\quad \sum_{l=1}^{s}\sum_{j=1}^{s}\xi_l\xi_j \operatorname{sinc}_{\mathbf{b}}(\mathbf{w}_l,\mathbf{w}_j) \,. \end{aligned}$$

Using this discrepancy measure, global adaptive and greedy adaptive sequences of points and weights can be found.

However, we note that if we fix the points, then optimizing just the weights is a simple convex optimization problem. The box discrepancy can be written as

$$D_p^{\square\mathbf{b}}(S,\Xi)^2 = \pi^{-d}\prod_{j=1}^{d}\int_{-b_j}^{b_j}|\varphi_j(\beta)|^2 d\beta - 2\mathbf{v}^T\xi + \xi^T\mathbf{H}\xi \,,$$

where $\xi \in \mathbb{R}^s$ has entry i equal to ξ_i, $\mathbf{v} \in \mathbb{R}^s$ and $\mathbf{H} \in \mathbb{R}^{s\times s}$ are defined by

$$\mathbf{H}_{ij} = \operatorname{sinc}_{\mathbf{b}}(\mathbf{w}_l,\mathbf{w}_j)$$

$$\mathbf{v}_i = (2\pi)^{-d}\prod_{j=1}^{d}\int_{-b_j}^{b_j}\varphi_j(\beta)e^{iw_{lj}\beta}d\beta \,.$$

The ξ that minimizes $D_p^{\square\mathbf{b}}(S,\Xi)^2$ is equal to $\mathbf{H}^{-1}\mathbf{v}$, but there is no guarantee that $\xi_i \geq 0$ for all i. We need to explicitly impose these conditions. Thus, the optimal weights can be found by solving the following convex optimization problem

$$\Xi^* = \arg\min_{\xi\in\mathbb{R}^s} \xi^T\mathbf{H}\xi - 2\mathbf{v}^T\xi \quad \text{s.t. } \xi \geq 0 \,. \tag{17}$$

Selecting the weights in such a way is closely connected to the so-called *Bayesian Monte Carlo* (BMC) method, originally suggested by Ghahramani and Rasmussen (2003). In BMC, a Bayesian approach is utilized in which the function is a assumed to be random with a prior that is a Gaussian Process. Combining with the observations, a posterior is obtained, which naturally leads to the selection of weights. Huszár and Duvenaud (2012) subsequently pointed out the connection between this approach and the herding algorithm discussed earlier.

We remark that as long as all the weights are positive, the hypothesis space of functions induced by the feature map (that is, $\{g_{\mathbf{w}}(\mathbf{x}) = \hat{\Psi}_S^T(\mathbf{x})\mathbf{w},\ \mathbf{w} \in \mathbb{R}^s\}$) will not change in terms of the set of functions in it. However, the norms will be affected (that is, the norm of a function in that set also depends on the weights), which in turn affects the regularization.

6. Experiments

In this section we report experiments with both classical QMC sequences and adaptive sequences learnt from box discrepancy minimization.

6.1 Experiments With Classical QMC Sequences

We examine the behavior of classical low-discrepancy sequences when compared to random Fourier features (i.e., MC). We consider four sequences: Halton, Sobol', Lattice Rules, and Digital Nets. For Halton and Sobol', we use the implementation available in MATLAB.[3] For Lattice Rules and Digital Nets, we use publicly available implementations.[4] For all four low-discrepancy sequences, we use scrambling and shifting techniques recommended in the QMC literature (see Dick et al. (2013) for details). For Sobol', Lattice Rules and Digital Nets, scrambling introduces randomization and hence variance. For Halton sequence, scrambling is deterministic, and there is no variance. The generation of these sequences is extremely fast, and quite negligible when compared to the time for any reasonable downstream use. For example, for `census` data set with size 18,000 by 119, if we choose the number of random features $s = 2000$, the running time for performing kernel ridge regression model is more than 2 minutes, while the time of generating the QMC sequences is only around 0.2 seconds (Digital Nets sequence takes longer, but not much longer) and that of MC sequence is around 0.01 seconds. Therefore, we do not report running times as these are essentially the same across methods.

In all experiments, we work with a Gaussian kernel. For learning, we use regularized least square classification on the feature mapped data set, which can be thought of as a form of approximate kernel ridge regression. For each data set, we performed 5-fold cross-validation when using random Fourier features (MC sequence) to set the bandwidth σ, and then used the same σ for all other sequences.

6.1.1 QUALITY OF KERNEL APPROXIMATION

In our setting, the most natural and fundamental metric for comparison is the quality of approximation of the Gram matrix. We examine how close $\tilde{\mathbf{K}}$ (defined by $\tilde{\mathbf{K}}_{ij} = \tilde{k}(\mathbf{x}_i, \mathbf{x}_j)$ where $\tilde{k}(\cdot,\cdot) = \langle \hat{\Psi}_S(\cdot), \hat{\Psi}_S(\cdot) \rangle$ is the kernel approximation) is to the Gram matrix $\mathbf{K}$ of the exact kernel.

We examine four data sets: `cpu` (6554 examples, 21 dimensions), `census` (a subset chosen randomly with 5,000 examples, 119 dimensions), `USPST` (1,506 examples, 250 dimensions after PCA) and `MNIST` (a subset chosen randomly with 5,000 examples, 250 dimensions after PCA). The reason we do subsampling on large data sets is to be able to compute the full exact Gram matrix for comparison purposes. The reason we use dimensionality reduction on `MNIST` is that the maximum dimension supported by the Lattice Rules implementation we use is 250.

To measure the quality of approximation we use both $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 / \|\mathbf{K}\|_2$ and $\|\mathbf{K} - \tilde{\mathbf{K}}\|_F / \|\mathbf{K}\|_F$. The plots are shown in Figure 2.

We can clearly see that except Sobol' sequences classical low-discrepancy sequences consistently produce better approximations to the Gram matrix than the approximations produced using MC sequences. Among the four classical QMC sequences, the Digital Nets, Lattice Rules and Halton sequences yield much lower error. Similar results were observed for other data sets (not reported here). Although using scrambled variants of QMC sequences may incur some variance, the variance is quite small compared to that of the MC random features.

Scrambled (whether deterministic or randomized) QMC sequences tend to yield higher accuracies than non-scambled QMC sequences. In Figure 3, we show the ratio between the relative errors

3. http://www.mathworks.com/help/stats/quasi-random-numbers.html
4. http://people.cs.kuleuven.be/ dirk.nuyens/qmc-generators/

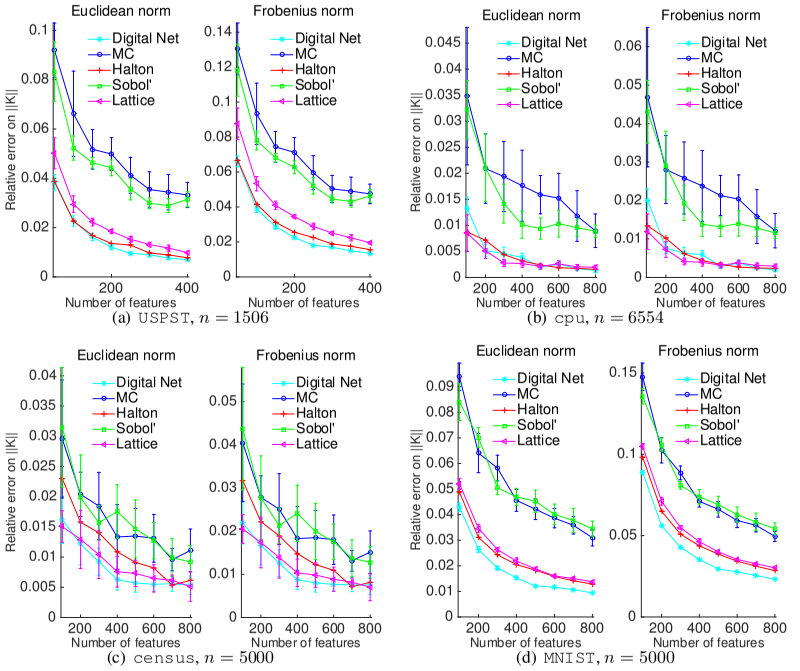
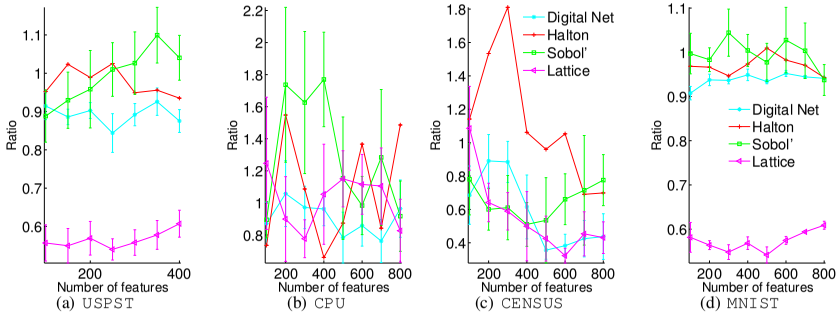


Figure 2: Relative error on approximating the Gram matrix measured in Euclidean norm and Frobenius norm, i.e., $\|\mathbf{K} - \tilde{\mathbf{K}}\|_2 / \|\mathbf{K}\|_2$ and $\|\mathbf{K} - \tilde{\mathbf{K}}\|_F / \|\mathbf{K}\|_F$, for various s. For each kind of random feature and s, 10 independent trials are executed, and the mean and standard deviation are plotted.

achieved by using both scrambled and non-scrambled QMC sequences. As can be seen, scrambled QMC sequences provide more accurate approximations in most cases as the ratio value tends to be less than one. In particular, scrambled Lattice sequence outperforms the non-scrambled one across all the cases for larger values of s. Therefore, in the rest of the experiments we use scrambled sequences.

6.1.2 GENERALIZATION ERROR

We consider two regression data sets, `cpu` and `census`, and use (approximate) kernel ridge regression to build a regression model. The ridge parameter is set by the optimal value we obtain via 5-fold cross-validation on the training set by using the MC sequence. Table 1 summarizes the results.



(a) USPST (b) CPU (c) CENSUS (d) MNIST

Figure 3: Ratio between relative errors on approximating the Gram matrix using both the scrambled and non-scambled version of the same QMC sequence for various s. The lower the ratio value is, the more accurate the scrambled QMC approximation is. For each kind of QMC sequences and s, 10 independent trails are executed, and the mean and standard deviation are plotted.

	s	HALTON	SOBOL'	LATTICE	DIGIT	MC
CPU	100	**0.0367**	0.0383	0.0374	0.0376	0.0383
		(0)	(0.0015)	(0.0010)	(0.0010)	(0.0013)
	500	**0.0339**	0.0344	0.0348	0.0343	0.0349
		(0)	(0.0005)	(0.0007)	(0.0005)	(0.0009)
	1000	**0.0334**	0.0339	0.0337	0.0335	0.0338
		(0)	(0.0007)	(0.0004)	(0.0003)	(0.0005)
CENSUS	400	**0.0529**	0.0747	0.0801	0.0755	0.0791
		(0)	(0.0138)	(0.0206)	(0.0080)	(0.0180)
	1200	**0.0553**	0.0588	0.0694	0.0587	0.0670
		(0)	(0.0080)	(0.0188)	(0.0067)	(0.0078)
	1800	**0.0498**	0.0613	0.0608	0.0583	0.0600
		(0)	(0.0084)	(0.0129)	(0.0100)	(0.0113)

Table 1: Regression error, i.e., $\|\hat{\mathbf{y}} - \mathbf{y}\|_2/\|\mathbf{y}\|_2$ where $\hat{\mathbf{y}}$ is the predicted value and $\mathbf{y}$ is the ground truth. For each kind of random feature and s, 10 independent trials are executed, and the mean and standard deviation are listed.

As we see, for cpu, all the sequences behave similarly, with the Halton sequence yielding the lowest test error. For census, the advantage of using Halton sequence is significant (almost 20% reduction in generalization error) followed by Digital Nets and Sobol'. In addition, MC sequence tends to generate higher variance across all the sampling size. Overall, QMC sequences, especially Halton, outperform MC sequences on these data sets.

When performed on classification data sets by using the same learning model, with a moderate range of s, e.g., less than 2000, the QMC sequences do not yield accuracy improvements over the MC sequence with the same consistency as in the regression case. The connection between kernel approximation and the performance in downstream applications is outside the scope of the current paper. Worth mentioning in this regard, is the recent work by Bach (2013), which analyses the connection between Nyström approximations of the Gram matrix, and the regression error, and the work of El Alaoui and Mahoney (2014) on kernel methods with statistical guarantees.

6.1.3 BEHAVIOR OF BOX DISCREPANCY

Next, we examine if $D^{\square}$ is predictive of the quality of approximation. We compute the normalized square box discrepancy values (i.e., $\pi^d(\prod_{j=1}^d b_j)^{-1}D^{\square}(S)^2$) as well as Gram matrix approximation error for the different sequences with different sample sizes s. The expected normalized square box discrepancy values for MC are computed using (13).

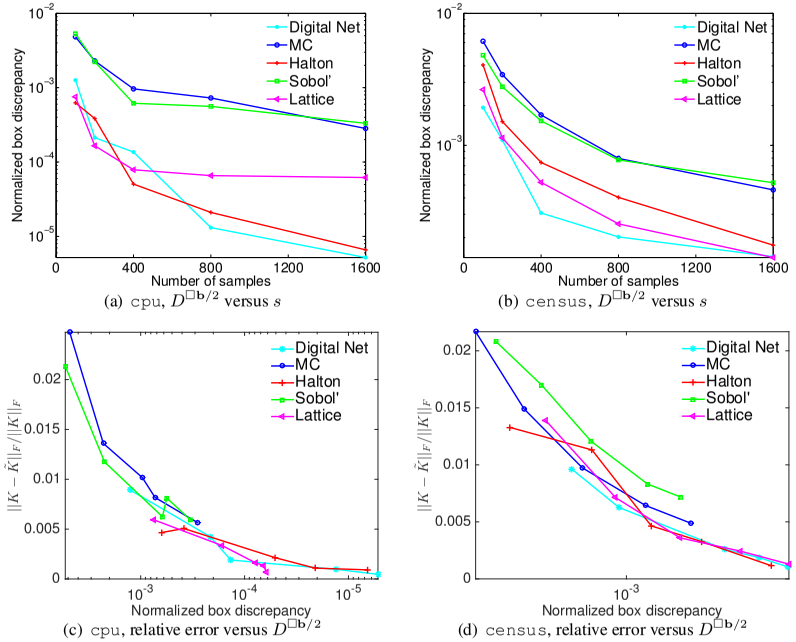
Our experiments revealed that using the full $\square\mathbf{b}$ does not yield box discrepancy values that are very useful. Either the values were not predictive of the kernel approximation, or they tended to stay constant. Recall, that while the bounding box $\square\mathbf{b}$ is set based on observed ranges of feature values in the data set, the actual distribution of points $\tilde{\mathcal{X}}$ encountered inside that box might be far from uniform. This lead us to consider the discrepancy measure when measured on the central part of the bounding box (i.e., $\square\mathbf{b}/2$ instead of $\square\mathbf{b}$), which is equal to the integration error averaged over that part of the bounding box. Presumably, points from $\tilde{\mathcal{X}}$ concentrate in that region, and they may be more relevant for downstream predictive task.

The results are shown in Figure 4. In the top graphs we can see, as expected, increasing number of features in the sequence leads to a lower box discrepancy value. In the bottom graphs, which compare $\|\mathbf{K} - \tilde{\mathbf{K}}\|_F/\|\mathbf{K}\|_F$ to $D^{\square\mathbf{b}/2}$, we can see a strong correlation between the quality of approximation and the discrepancy value.

6.2 Experiments With Adaptive QMC Sequences

The goal of this subsection is to provide a proof-of-concept for learning adaptive QMC sequences, using the three schemes described in Section 5. We demonstrate that QMC sequences can be improved to produce better approximation to the Gram matrix, and that can sometimes lead to improved generalization error.

Note that the running time of learning the adaptive sequences is less relevant in our experimental setting for the following reasons. Given the values of s, d, $\mathbf{b}$ and σ the optimization of a sequence needs only to be done once. There is some flexibility in these parameters: d can be adjusted by adding zero features or by doing PCA on the input; one can use longer or shorter sequences; and the data can be forced to a fit a particular bounding box using (possibly non-equal) scaling of the features (this, in turn, affects the choice of the σ) . Since designing adaptive QMC sequences is data-independent with applicability to a variety of downstream applications of kernel methods, it is quite conceivable to generate many point sets in advance and to use them for many learning tasks.



(a) cpu, $D^{\square \mathbf{b}/2}$ versus s (b) census, $D^{\square \mathbf{b}/2}$ versus s

(c) cpu, relative error versus $D^{\square \mathbf{b}/2}$ (d) census, relative error versus $D^{\square \mathbf{b}/2}$

Figure 4: Discrepancy values ($D^{\square \mathbf{b}/2}$) for the different sequences on cpu and census. We measure the discrepancy on the central part of the bounding box (we use $\square \mathbf{b}/2$ instead of $\square \mathbf{b}$ as the domain in the box discrepancy).

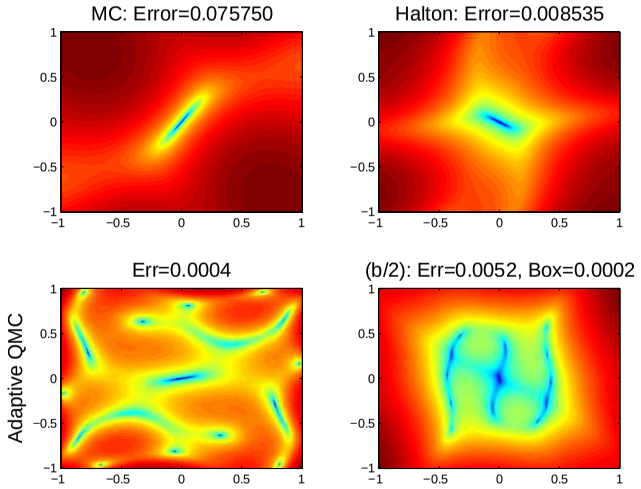


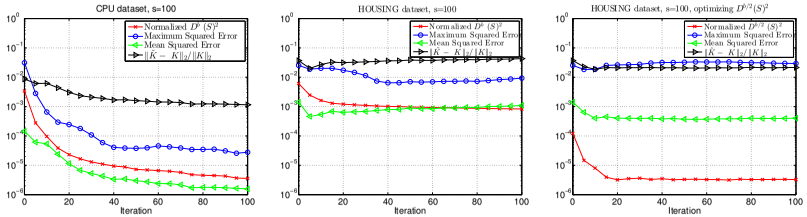
Figure 5: Integration error.

Furthermore, the total size of the sequences ($s \times d$) is independent of the number of examples n, which is the dominant term in large scale learning settings.

We name the three sequences as *Global Adaptive*, *Greedy Adaptive* and *Weighted* respectively. For *Global Adaptive*, the Halton sequence is used as the initial setting of the optimization variables S. For *Greedy Adaptive*, when optimizing for $\mathbf{w}_t$, the t-th point in the Halton sequence is used as the initial point. In both cases, we use non-linear conjugate gradient to perform numerical optimization. For *Weighted*, the initial features are generated using the Halton sequence and we optimize for the weights. We used CVX (Grant and Boyd, 2014, 2008) to compute the sequence (solve (17)).

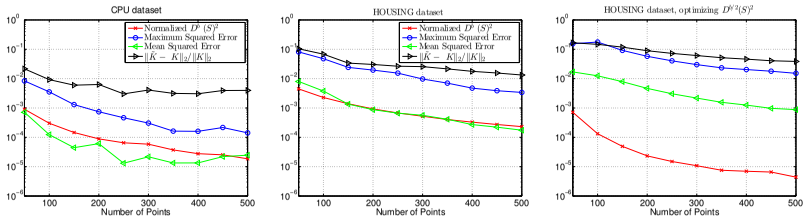
6.2.1 INTEGRAL APPROXIMATION

We begin by examining the integration error over the unit square by using three different sequences, namely, MC, Halton and global adaptive QMC sequences. The integral is of the form $\int_{[0,1]^2} e^{-i\mathbf{u}^T\mathbf{t}} d\mathbf{t}$ where $\mathbf{u}$ spans the unit square. The error is plotted in Figure 5. We see that MC sequences concentrate most of the error reduction near the origin. The Halton sequence gives significant improvement expanding the region of low integration error. Global adaptive QMC sequences give another order of magnitude improvement in integration error which is now diffused over the entire unit square; the estimation of such sequences is "aware" of the full integration region. In fact, by controlling the box size (see plot labeled b/2), adaptive sequences can be made to focus in a specified sub-box which can help with generalization if the actual data distribution is better represented by this sub-box.



(a) **(b)** **(c)**

Figure 6: Examining the behavior of learning *Global Adaptive* sequences. Various metrics on the Gram matrix approximation are plotted.



(a) **(b)** **(c)**

Figure 7: Examining the behavior of learning *Greedy Adaptive* sequences. Various metrics on the Gram matrix approximation are plotted.

6.2.2 QUALITY OF KERNEL APPROXIMATION

In Figure 6 and Figure 7 we examine how various metrics (discrepancy, maximum squared error, mean squared error, norm of the error) on the Gram matrix approximation evolve during the optimization process for both adaptive sequences. Since learning the adaptive sequences on data set with low dimensional features is more affordable, the experiment is performed on two such data sets, namely, `cpu` and `housing`.

For *Global Adaptive*, we fixed $s = 100$ and examine how the performance evolves as the number of iterations grows. In Figure 6 (a) we examine the behavior on `cpu`. We see that all metrics go down as the iteration progresses. This supports our hypothesis that by optimizing the box discrepancy we can improve the approximation of the Gram matrix. Figure 6 (b), which examines the same metrics on the scaled version of the `housing` data set, has some interesting behaviors. Initially all metrics go down, but eventually all the metrics except the box-discrepancy start to go up; the box-discrepancy continues to go down. One plausible explanation is that the integrands are not uniformly distributed in the bounding box, and that by optimizing the expectation over the entire box we start to overfit it, thereby increasing the error in those regions of the box where integrands actually concentrate. One possible way to handle this is to optimize closer to the center of the box

(e.g., on $\square\mathbf{b}/2$), under the assumption that integrands concentrate there. In Figure 6 (c) we try this on the `housing` data set. We see that now the mean error and the norm error are much improved, which supports the interpretation above. But the maximum error eventually goes up. This is quite reasonable as the outer parts of the bounding box are harder to approximate, so the maximum error is likely to originate from there. Subsequently, we stop the adaptive learning of the QMC sequences early, to avoid the actual error from going up due to averaging.

For *Greedy Adaptive*, we examine its behavior as the number of points increases. In Figure 7 (a) and (b), as expected, as the number of points in the sequence increases, the box discrepancy goes down. This is also translated to non-monotonic decrease in the other metrics of Gram matrix approximation. However, unlike the global case, we see in Figure 7 (c), when the points are generated by optimizing on a smaller box $\square\mathbf{b}/2$, the resulting metrics become higher for a fixed number of points. Although the *Greedy Adaptive* sequence can be computed faster than the adaptive sequence, potentially it might need a large number of points to achieve certain low magnitude of discrepancy. Hence, as shown in the plots, when the number of points is below 500, the quality of the optimization is not good enough to provide a good approximation the Gram matrix. For example, one can check when the number of points is 100, the discrepancy value of the *Greedy Adaptive* sequence is higher than that of the *Global Adaptive* sequence with more than 10 iterations.

		$D^{\square\mathbf{b}}$				$D^{\square\mathbf{b}/4}$			
	s	HALTON	GLOBAL$_{\mathbf{b}}$	GREEDY$_{\mathbf{b}}$	WEIGHTED$_{\mathbf{b}}$	HALTON	GLOBAL$_{\mathbf{b}/4}$	GREEDY$_{\mathbf{b}/4}$	WEIGHTED$_{\mathbf{b}/4}$
CPU	100	3.41E-3	1.29E-6	3.02E-4	7.84E-5	9.44E-5	5.57E-8	2.62E-5	1.67E-8
	300	8.09E-4	5.14E-6	5.85E-5	1.45E-6	2.57E-5	1.06E-7	3.08E-6	2.93E-9
	500	2.39E-4	2.83E-6	1.86E-5	3.39E-7	7.91E-6	2.62E-8	1.04E-6	2.43E-9
CENSUS	400	2.61E-3	9.32E-4	7.47E-4	8.83E-4	5.73E-4	2.79E-5	2.20E-5	2.45E-5
	800	1.21E-3	5.02E-4	3.33E-4	4.91E-4	2.21E-4	1.12E-5	8.04E-6	5.46E-6
	1200	8.27E-4	3.41E-4	2.06E-4	3.39E-4	1.39E-4	8.15E-6	4.23E-6	2.29E-6
	1800	5.31E-4	2.17E-4	1.27E-4	2.31E-4	3.79E-5	5.59E-6	2.63E-6	8.37E-7
	2200	4.33E-4	1.73E-4	1.01E-4	1.87E-4	2.34E-5	3.35E-6	1.95E-6	4.93E-7

Table 2: Discrepancy values, measured on the full bounding box and its central part, i.e., $D^{\square\mathbf{b}}$ and $D^{\square\mathbf{b}/4}$.

Table 2 also shows the discrepancy values of various sequences on `cpu` and `census`. Using adaptive sequences improves the discrepancy values by orders-of-magnitude. We note that a significant reduction in terms of discrepancy values can be achieved using only weights, sometimes yielding discrepancy values that are better than the hard-to-compute global or greedy sequences.

6.2.3 GENERALIZATION ERROR

We use the three algorithms for learning adaptive sequences as described in the previous subsections, and use them for doing approximate kernel ridge regression. The ridge parameter is set by the value which is near-optimal for both sequences in 5-fold cross-validation on the training set. Table 3 summarizes the results.

For both `cpu` and `census`, at least one of the adaptive sequences sequences can yield lower test error for each sampling size (since the test error is already low, around 3% or 5%, such improvement in accuracy is not trivial). For `cpu`, greedy approach seems to give slightly better results. When $s = 500$ or even larger (not reported here), the performance of the sequences are very close. For

	s	HALTON	GLOBAL$_{\mathbf{b}}$	GLOBAL$_{\mathbf{b}/4}$	GREEDY$_{\mathbf{b}}$	GREEDY$_{\mathbf{b}/4}$	WEIGHTED$_{\mathbf{b}}$	WEIGHTED$_{\mathbf{b}/4}$
CPU	100	0.0304	0.0315	**0.0296**	0.0307	**0.0296**	0.0366	0.0305
	300	0.0303	0.0278	0.0293	0.0274	**0.0269**	0.0290	0.0302
	500	0.0348	0.0347	0.0348	0.0328	**0.0291**	0.0342	0.0347
CENSUS	400	0.0529	0.1034	0.0997	0.0598	0.0655	**0.0512**	0.0926
	800	0.0545	0.0702	0.0581	0.0522	0.0501	**0.0476**	0.0487
	1200	0.0553	0.0639	**0.0481**	0.0525	0.0498	0.0496	0.0501
	1800	0.0498	0.0568	**0.0476**	0.0685	0.0548	0.0498	0.0491
	2200	0.0519	**0.0487**	0.0515	0.0694	0.0504	0.0529	0.0499

Table 3: Regression error, i.e., $\|\hat{\mathbf{y}} - \mathbf{y}\|_2/\|\mathbf{y}\|_2$ where $\hat{\mathbf{y}}$ is the predicted value and $\mathbf{y}$ is the ground truth.

census, the weighted sequence yields the lowest generalization error when $s = 400, 800$. Afterwards we can see global adaptive sequence outperforms the rest of the sequences, even though it has better discrepancy values. In some cases, adaptive sequences sometimes produce errors that are bigger than the unoptimized sequences.

In most cases, the adaptive sequence on the central part of the bounding box outperforms the adaptive sequence on the entire box. This is likely due to the non-uniformity phenomena discussed earlier.

7. Conclusion and Future Work

Recent work on applying kernel methods to very large data sets, has shown their ability to achieve state-of-the-art accuracies that sometimes match those attained by Deep Neural Networks (DNN) (Huang et al., 2014). Key to these results is the ability to apply kernel method to such data sets. The random features approach, originally due to Rahimi and Recht (2008), as emerged as a key technology for scaling up kernel methods (Sindhwani and Avron, 2014).

Close examination of those empirical results reveals that to achieve state-of-the-art accuracies, a very large number of random features was needed. For example, on TIMIT, a classical speech recognition data set, over 200,000 random features were used in order to match DNN performance (Huang et al., 2014). It is clear that improving the efficiency of random features can have a significant impact on our ability to scale up kernel methods, and potentially get even higher accuracies.

This paper is the first to exploit high-dimensional approximate integration techniques from the QMC literature in this context, with promising empirical results backed by rigorous theoretical analyses. Avenues for future work include incorporating stronger data-dependence in the estimation of adaptive sequences and analyzing how resulting Gram matrix approximations translate into downstream performance improvements for a variety of large-scale learning tasks.

Acknowledgements

The authors would like to thank Josef Dick for useful pointers to literature about improvement of the QMC sequences; Ha Quang Minh for several discussions on Paley-Wiener spaces and RKHS theory; the anonymous ICML reviewers for pointing out the connection to herding and other helpful comments; the anonymous JMLR reviewers for suggesting weighted sequences and other helpful

comments. This research was supported by the XDATA program of the Defense Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323. This work was done while H. Avron and V. Sindhwani were research staff members and J. Yang was a summer intern at IBM Research.

Appendix A. Technical Details

In this section we give detailed proofs of the assertions made in Section 4 and 5.

A.1 Proof of Proposition 5

Recall, for any $\mathbf{t} \in \mathbb{R}^d$, for $\Phi^{-1}(\mathbf{t})$, we mean $\left(\Phi_1^{-1}(t_1), \ldots, \Phi_d^{-1}(t_d)\right) \in \mathbb{R}^d$, where $\Phi_j(\cdot)$ is the CDF of $p_j(\cdot)$.

From $f_{\mathbf{u}}(\mathbf{t}) = e^{-i\mathbf{u}^T\Phi^{-1}(\mathbf{t})}$, for any $j = 1, \ldots, d$, we have

$$\frac{\partial f(\mathbf{t})}{\partial \mathbf{t}_j} = (-i)\frac{u_j}{p_j(\Phi_j^{-1}(t_j))}e^{-i\mathbf{u}^T\Phi_j^{-1}(\mathbf{t})} .$$

Thus,

$$\frac{\partial^d f(\mathbf{t})}{\partial t_1 \cdots \partial t_d} = \prod_{j=1}^{d}\left((-i)\frac{u_j}{p_j(\Phi_j^{-1}(t_j))}\right)e^{-i\mathbf{u}^T\Phi_j^{-1}(\mathbf{t})} .$$

In (6), when $I = [d]$,

$$\begin{aligned}
\int_{[0,1]^{|I|}}\left|\frac{\partial f}{\partial \mathbf{u}_I}\right|d\mathbf{t}_I &= \int_{[0,1]^d}\left|\frac{\partial^d f(\mathbf{t})}{\partial t_1 \cdots \partial t_d}\right|dt_1 \cdots dt_d \\
&= \int_{[0,1]^d}\left|\prod_{j=1}^{d}\left((-i)\frac{u_j}{p_j(\Phi_j^{-1}(t_j))}\right)e^{-i\mathbf{u}^T\Phi_j^{-1}(\mathbf{t})}\right|dt_1 \cdots dt_d \\
&= \int_{[0,1]^d}\prod_{j=1}^{d}\left|\frac{u_j}{p_j(\Phi_j^{-1}(t_j))}\right|dt_1 \cdots dt_d \\
&= \prod_{j=1}^{d}\left(\int_{[0,1]}\left|\frac{u_j}{p_j(\Phi_j^{-1}(t_j))}\right|dt_j\right) . \qquad (18)
\end{aligned}$$

With a change of variable, $\Phi_j(t_j) = v_j$, for $j = 1, \ldots, d$, (18) becomes

$$\prod_{j=1}^{d}\left(\int_{[0,1]}\left|\left(\frac{u_j}{p_j(\Phi_j^{-1}(t_j))}\right)\right|dt_j\right) = \prod_{j=1}^{d}\left(\int_{\mathbb{R}}|u_j|dv_j\right) = \infty .$$

As this is a term in (6), we know that $V_{HK}[f_{\mathbf{u}}(\mathbf{t})]$ is unbounded. ■

A.2 Proof of Proposition 6

We need the following lemmas, across which we share some notation.

Lemma 17 *Assuming that $\kappa = \sup_{\mathbf{x}\in\mathbb{R}^d} h(\mathbf{x},\mathbf{x}) < \infty$, if $f \in \mathcal{H}$, where $\mathcal{H}$ is an RKHS with kernel $h(\cdot,\cdot)$, the integral $\int_{\mathbb{R}^d} f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$ is finite.*

Proof For notational convenience, we note that

$$\int_{\mathbb{R}^d} f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = \mathbb{E}\left[f(X)\right],$$

where $\mathbb{E}[\cdot]$ denotes expectation and X is a random variable distributed according to the probability density $p(\cdot)$ on $\mathbb{R}^d$.

Now consider a linear functional T that maps f to $\mathbb{E}[f(X)]$, i.e.,

$$T[f] = \mathbb{E}\left[f(X)\right]. \tag{19}$$

The linear functional T is a bounded linear functional on the RKHS $\mathcal{H}$. To see this:

$$\begin{aligned}
|\mathbb{E}\left[f(X)\right]| &\leq \mathbb{E}\left[|f(X)|\right] \quad \text{(Jensen's Inequality)}\\
&= \mathbb{E}\left[|\langle f, h(X,\cdot)\rangle_{\mathcal{H}}|\right] \quad \text{(Reproducing Property)}\\
&\leq \|f\|_{\mathcal{H}}\mathbb{E}\left[\|h(X,\cdot)\|_{\mathcal{H}}\right] \quad \text{(Cauchy-Schwartz)}\\
&\leq \|f\|_{\mathcal{H}}\mathbb{E}\left[\sqrt{h(X,X)}\right] = \|f\|_{\mathcal{H}}\sqrt{\kappa} < \infty .
\end{aligned}$$

This shows that the integral $\int_{\mathbb{R}^d} f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$ exists. ■

Lemma 18 *The mean $\mu_{h,p}(\mathbf{u}) = \int_{\mathbb{R}^d} h(\mathbf{u},\mathbf{x})p(\mathbf{x})d\mathbf{x}$ is in $\mathcal{H}$. In addition, for any $f \in \mathcal{H}$,*

$$\mathbb{E}\left[f(X)\right] = \int_{\mathbb{R}^d} f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = \langle f, \mu_{h,p}\rangle_{\mathcal{H}} . \tag{20}$$

Proof From the Riesz Representation Theorem, every bounded linear functional on $\mathcal{H}$ admits an inner product representation. Therefore, for T defined in (19), there exists $\mu_{h,p} \in \mathcal{H}$ such that,

$$T[f] = \mathbb{E}\left[f(X)\right] = \langle f, \mu_{h,p}\rangle_{\mathcal{H}} .$$

Therefore we have, $\langle f, \mu_{h,p}\rangle_{\mathcal{H}} = \int f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$ for all $f \in \mathcal{H}$. For any $\mathbf{z}$, choosing $f(\cdot) = h(\mathbf{z},\cdot)$, where $h(\cdot,\cdot)$ is the kernel associated with $\mathcal{H}$, and invoking the reproducing property we see that,

$$\mu_{h,p}(\mathbf{z}) = \langle h(\mathbf{z},\cdot), \mu_{h,p}\rangle_{\mathcal{H}} = \int_{\mathbb{R}^d} h(\mathbf{z},\mathbf{x})p(\mathbf{x})d\mathbf{x} .$$

■

The proof of Proposition 6 follows from the existence Lemmas above, and the following steps.

$$\begin{aligned}
\epsilon_{S,p}[f] &= \left|\int_{\mathbb{R}^d} f(\mathbf{x})p(\mathbf{x})d\mathbf{x} - \frac{1}{s}\sum_{l=1}^{s} f(\mathbf{w}_l)\right|\\
&= \left|\langle f, \mu_{h,p}\rangle_{\mathcal{H}} - \frac{1}{s}\sum_{l=1}^{s}\langle f, h(\mathbf{w}_l,\cdot)\rangle_{\mathcal{H}}\right|\\
&= \left|\langle f, \mu_{h,p} - \frac{1}{s}\sum_{l=1}^{s} h(\mathbf{w}_l,\cdot)\rangle_{\mathcal{H}}\right|\\
&\leq \|f\|_{\mathcal{H}}\left\|\mu_{h,p} - \frac{1}{s}\sum_{l=1}^{s} h(\mathbf{w}_l,\cdot)\right\|_{\mathcal{H}} = \|f\|_{\mathcal{H}} D_{h,p}(S) ,
\end{aligned}$$

where $D_{h,p}(S)$ is given as follows,

$$\begin{aligned}
D_{h,p}(S)^2 &= \left\|\mu_{h,p} - \frac{1}{s}\sum_{l=1}^{s} h(\mathbf{w}_l,\cdot)\right\|_{\mathcal{H}}^2\\
&= \langle\mu_{h,p},\mu_{h,p}\rangle_{\mathcal{H}} - \frac{2}{s}\sum_{l=1}^{s}\langle\mu_{h,p}, h(\mathbf{w}_l,\cdot)\rangle_{\mathcal{H}} + \frac{1}{s^2}\sum_{l=1}^{s}\sum_{j=1}^{s}\langle h(\mathbf{w}_l,\cdot), h(\mathbf{w}_j,\cdot)\rangle_{\mathcal{H}}\\
&= \mathbb{E}\left[\mu_{h,p}(X)\right] - \frac{2}{s}\sum_{l=1}^{s}\mathbb{E}\left[h(\mathbf{w}_l,\cdot)\right] + \frac{1}{s^2}\sum_{l=1}^{s}\sum_{j=1}^{s} h(\mathbf{w}_l,\mathbf{w}_j)\\
&= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\omega,\phi)p(\omega)p(\phi)d\omega d\phi - \frac{2}{s}\sum_{l=1}^{s}\int_{\mathbb{R}^d} h(\mathbf{w}_l,\omega)p(\omega)d\omega\\
&\quad + \frac{1}{s^2}\sum_{l=1}^{s}\sum_{j=1}^{s} h(\mathbf{w}_l,\mathbf{w}_j) .
\end{aligned}$$

■

A.3 Proof of Theorem 9

We apply (9) to the particular case of $h = \mathrm{sinc}_{\mathbf{b}}$. We have

$$\begin{aligned}
\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\omega,\phi)p(\omega)p(\phi)d\omega d\phi &= \pi^{-d}\int_{\mathbb{R}^d}\int_{\mathbb{R}^d}\prod_{j=1}^{d}\frac{\sin(b_j(\omega_j-\phi_j))}{\omega_j-\phi_j}p_j(\omega_j)p_j(\phi_j)d\omega d\phi\\
&= \pi^{-d}\prod_{j=1}^{d}\int_{\mathbb{R}}\int_{\mathbb{R}}\frac{\sin(b_j(\omega_j-\phi_j))}{\omega_j-\phi_j}p_j(\omega_j)p_j(\phi_j)d\omega_j d\phi_j ,
\end{aligned}$$

and

$$\begin{aligned}
\sum_{l=1}^{s}\int_{\mathbb{R}^d} h(w_l,\omega)p(\omega)d\omega &= \pi^{-d}\sum_{l=1}^{s}\int_{\mathbb{R}^d}\prod_{j=1}^{d}\frac{\sin(b_j(w_{lj}-\omega_j))}{w_{lj}-\omega_j}p_j(\omega_j)d\omega\\
&= \pi^{-d}\sum_{l=1}^{s}\prod_{j=1}^{d}\int_{\mathbb{R}^d}\frac{\sin(b_j(w_{lj}-\omega_j))}{w_{lj}-\omega_j}p_j(\omega_j)d\omega_j .
\end{aligned}$$

So we can consider each coordinate on its own.

Fix j. We have

$$\begin{aligned}\int_{\mathbb{R}} \frac{\sin(b_j x)}{x} p_j(x)dx &= \int_{\mathbb{R}}\int_0^{b_j} \cos(\beta x)p_j(x)d\beta dx \\ &= \frac{1}{2}\int_{-b_j}^{b_j}\int_{\mathbb{R}} e^{i\beta x}p(x)dxd\beta \\ &= \frac{1}{2}\int_{-b_j}^{b_j}\varphi_j(\beta)d\beta\ .\end{aligned}$$

The interchange in the second line is allowed since the $p_j(x)$ makes the function integrable (with respect to x).

Now fix $w \in \mathbb{R}$ as well. Let $h_j(x,y) = \sin(b_j(x-y))/\pi(x-y)$. We have

$$\begin{aligned}\int_{\mathbb{R}} h_j(\omega, w)p_j(\omega)d\omega &= \pi^{-1}\int_{\mathbb{R}} \frac{\sin(b_j(\omega-w))}{\omega-w}p_j(\omega)d\omega \\ &= \pi^{-1}\int_{\mathbb{R}} \frac{\sin(b_j x)}{x}p_j(x+w)dx \\ &= (2\pi)^{-1}\int_{-b_j}^{b_j}\varphi_j(\beta)e^{iw\beta}d\beta\ ,\end{aligned}$$

where the last equality follows from first noticing that the characteristic function associated with the density function $x \mapsto p_j(x+w)$ is $\beta \mapsto \varphi(\beta)e^{iw\beta}$, and then applying the previous inequality.

We also have,

$$\begin{aligned}\int_{\mathbb{R}}\int_{\mathbb{R}} \frac{\sin(b_j(x-y))}{x-y}p_j(x)p_j(y)dxdy &= \int_{\mathbb{R}}\int_{\mathbb{R}}\int_0^{b_j}\cos(\beta(x-y))p_j(x)p_j(y)d\beta dxdy \\ &= \frac{1}{2}\int_{\mathbb{R}}\int_{\mathbb{R}}\int_{-b_j}^{b_j} e^{i\beta(x-y)}p_j(x)p_j(y)d\beta dxdy \\ &= \frac{1}{2}\int_{-b_j}^{b_j}\int_{\mathbb{R}}\int_{\mathbb{R}} e^{i\beta(x-y)}p_j(x)p_j(y)dxdyd\beta \\ &= \frac{1}{2}\int_{-b_j}^{b_j}\left(\int_{\mathbb{R}} e^{i\beta x}p_j(x)dx\right)\left(\int_{\mathbb{R}} e^{-i\beta y}p_j(y)dy\right)d\beta \\ &= \frac{1}{2}\int_{-b_j}^{b_j}\varphi_j(\beta)\varphi_j(\beta)^* d\beta \\ &= \frac{1}{2}\int_{-b_j}^{b_j}|\varphi_j(\beta)|^2 d\beta\ .\end{aligned}$$

The interchange at the third line is allowed because of $p_j(x)p_j(y)$. In the last line we use the fact that the $\varphi_j(\cdot)$ is Hermitian. ■

A.4 Proof of Theorem 12

Let $b > 0$ be a scalar, and let $u \in [-b, b]$ and $z \in \mathbb{R}$. We have,

$$\begin{aligned}\int_{-\infty}^{\infty} e^{-iux}\frac{\sin(b(x-z))}{\pi(x-z)}dx &= e^{-iuz}\int_{-\infty}^{\infty} e^{-i2\pi\frac{u}{2b}y}\frac{\sin(\pi y)}{\pi y}dy \\ &= e^{-iuz}\,\mathrm{rect}(u/2b) \\ &= e^{-iuz}\ .\end{aligned}$$

In the above, rect is the function that is 1 on $[-1/2, 1/2]$ and zero elsewhere.

The last equality implies that for every $\mathbf{u} \in \square\mathbf{b}$ and every $\mathbf{x} \in \mathbb{R}^d$ we have

$$f_{\mathbf{u}}(\mathbf{x}) = \int_{\mathbb{R}^d} f_{\mathbf{u}}(\mathbf{y})\,\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{x})d\mathbf{y}\ .$$

We now have for every $\mathbf{u} \in \square\mathbf{b}$,

$$\begin{aligned}\epsilon_{S,p}[f_{\mathbf{u}}] &= \left|\int_{\mathbb{R}^d} f_{\mathbf{u}}(\mathbf{x})p(\mathbf{x})d\mathbf{x} - \frac{1}{s}\sum_{i=1}^{s} f(\mathbf{w}_i)\right| \\ &= \left|\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} f_{\mathbf{u}}(\mathbf{y})\,\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{x})d\mathbf{y}p(\mathbf{x})d\mathbf{x} - \frac{1}{s}\sum_{i=1}^{s}\int_{\mathbb{R}^d} f_{\mathbf{u}}(\mathbf{y})\,\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{w}_i)d\mathbf{y}\right| \\ &= \left|\int_{\mathbb{R}^d} f_{\mathbf{u}}(\mathbf{y})\left[\int_{\mathbb{R}^d}\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{x})p(\mathbf{x})d\mathbf{x} - \frac{1}{s}\sum_{i=1}^{s}\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{w}_i)\right]d\mathbf{y}\right|\ .\end{aligned}$$

Let us denote

$$r_S(\mathbf{y}) = \int_{\mathbb{R}^d}\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{x})p(\mathbf{x})d\mathbf{x} - \frac{1}{s}\sum_{i=1}^{s}\mathrm{sinc}_{\mathbf{b}}(\mathbf{y},\mathbf{w}_i)\ .$$

So,

$$\epsilon_{S,p}[f_{\mathbf{u}}] = \left|\int_{\mathbb{R}^d} f_{\mathbf{u}}(\mathbf{y})r_S(\mathbf{y})d\mathbf{y}\right|\ .$$

The function $r_S(\cdot)$ is square-integrable, so it has a Fourier transform $\hat{r}_S(\cdot)$. The above formula is exactly the value of $\hat{r}_S(\mathbf{u})$. That is,

$$\epsilon_{S,p}[f_{\mathbf{u}}] = |\hat{r}_S(\mathbf{u})|\ .$$

Now,

$$\begin{aligned}
\mathbb{E}_{f\sim\mathcal{U}(\mathcal{F}_{\square\mathbf{b}})}\left[\epsilon_{S,p}[f]^2\right] &= \mathbb{E}_{\mathbf{u}\sim\mathcal{U}(\square\mathbf{b})}\left[\epsilon_{S,p}[f_{\mathbf{u}}]^2\right] \\
&= \int_{\mathbf{u}\in\square\mathbf{b}} |\hat{r}_S(\mathbf{u})|^2 \left(\prod_{j=1}^d 2b_j\right)^{-1} d\mathbf{u} \\
&= \left(\prod_{j=1}^d 2b_j\right)^{-1} \|\hat{r}_S\|_{L2}^2 \\
&= \frac{(2\pi)^d}{\prod_{j=1}^d 2b_j} \|r_S\|_{PW_{\mathbf{b}}}^2 \\
&= \frac{\pi^d}{\prod_{j=1}^d b_j} D_p^{\square}(S)^2 \,.
\end{aligned}$$

The equality before the last follows from Plancherel formula and the equality of the norm in $PW_{\mathbf{b}}$ to the $L2$-norm. The last equality follows from the fact that r_S is exactly the expression used in the proof of Proposition 6 to derive $D_p^{\square}$. ■

A.5 Proof of Corollary 13

In this case, $p(\mathbf{x}) = \prod_{j=1}^d p_j(x_j)$ where $p_j(\cdot)$ is the density function of $\mathcal{N}(0, 1/\sigma_j)$. The characteristic function associated with $p_j(\cdot)$ is $\varphi_j(\beta) = e^{-\frac{\beta^2}{2\sigma_j^2}}$. We apply (11) directly.

For the first term, since

$$\begin{aligned}
\int_0^{b_j} |\varphi_j(\beta)|^2 d\beta &= \int_0^{b_j} e^{-\frac{\beta^2}{\sigma_j^2}} d\beta \\
&= \sigma_j \int_0^{b_j/\sigma_j} e^{-y^2} dy \\
&= \frac{\sigma_j\sqrt{\pi}}{2} \operatorname{erf}\left(\frac{b_j}{\sigma_j}\right) ,
\end{aligned}$$

we have

$$\pi^{-d} \prod_{j=1}^d \int_0^{b_j} |\varphi_j(\beta)|^2 d\beta = \prod_{j=1}^d \frac{\sigma_j}{2\sqrt{\pi}} \operatorname{erf}\left(\frac{b_j}{\sigma_j}\right) . \tag{21}$$

For the second term, since

$$\begin{aligned}
\int_{-b_j}^{b_j} \varphi_j(\beta) e^{iw_{lj}\beta} d\beta &= \int_{-b_j}^{b_j} e^{-\frac{\beta^2}{2\sigma^2} + iw_{lj}\beta} d\beta \\
&= e^{-\frac{\sigma_j w_{lj}}{2}} \int_{-b_j}^{b_j} e^{-\left(\frac{\beta}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right)^2} d\beta \\
&= \sqrt{2}\sigma_j e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} \int_{-\frac{b_j}{\sqrt{2}\sigma_j}}^{\frac{b_j}{\sqrt{2}\sigma_j}} e^{-\left(y - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right)^2} dy \\
&= \sqrt{2}\sigma_j e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} \int_{-\frac{b_j}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}}^{\frac{b_j}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}} e^{-z^2} dz \\
&= \frac{\sqrt{\pi}\sigma_j}{\sqrt{2}} e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} \left(\operatorname{erf}\left(-\frac{b_j}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right) - \operatorname{erf}\left(\frac{b_j}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right)\right) \\
&= \sqrt{2\pi}\sigma_j e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} \operatorname{Re}\left(\operatorname{erf}\left(-\frac{b_j}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right)\right) ,
\end{aligned}$$

we have

$$\frac{2}{s}(2\pi)^{-d} \sum_{l=1}^s \prod_{j=1}^d \int_{-b_j}^{b_j} \varphi_j(\beta) e^{i\mathbf{w}_{lj}\beta} d\beta = \frac{2}{s} \sum_{l=1}^s \prod_{j=1}^d \frac{\sigma_j}{\sqrt{2\pi}} e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} \operatorname{Re}\left(\operatorname{erf}\left(-\frac{b_j}{\sqrt{2}\sigma_j} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right)\right) . \tag{22}$$

Combining (21), (22) and (11), (12) follows. ■

A.6 Proof of Corollary 14

The proof is similar to the proof of Theorem 3.6 of Dick et al. (2013). Notice that since $\sup_{\mathbf{x}\in\mathbb{R}^d} h(\mathbf{x},\mathbf{x}) < \infty$, we have $\int_{\mathbb{R}^d} h(\mathbf{x},\mathbf{x})p(\mathbf{x})d\mathbf{x} < \infty$. From Lemma 18 we know that $\int_{\mathbb{R}^d} h(\cdot,\mathbf{y})p(\mathbf{y})d\mathbf{y} \in \mathcal{H}$, hence from Lemma 17, we have $\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\mathbf{x},\mathbf{y})p(\mathbf{x})p(\mathbf{y})d\mathbf{x}d\mathbf{y} < \infty$.

By (9), we have

$$\begin{aligned}
D_{h,p}(S)^2 &= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\omega,\phi)p(\omega)p(\phi)d\omega d\phi \\
&\quad -\frac{2}{s}\sum_{l=1}^s \int_{\mathbb{R}^d} h(\mathbf{w}_l,\omega)p(\omega)d\omega \\
&\quad +\frac{1}{s^2}\sum_{l=1}^s h(\mathbf{w}_l,\mathbf{w}_l) + \frac{1}{s^2}\sum_{l,j=1,l\neq j}^s h(\mathbf{w}_l,\mathbf{w}_j) \,.
\end{aligned}$$

Then,

$$\begin{aligned}
\mathbb{E}\left[D_{h,p}(S)^2\right] = \int_{[0,1]^d}\cdots\int_{[0,1]^d} \Bigg(&\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} h(\omega,\phi)p(\omega)p(\phi)d\omega d\phi - \frac{2}{s}\sum_{l=1}^s \int_{\mathbb{R}^d} h(\Phi^{-1}(\mathbf{t}_l),\omega)p(\omega)d\omega \\
&+\frac{1}{s^2}\sum_{l=1}^s h(\Phi^{-1}(\mathbf{t}_l),\Phi^{-1}(\mathbf{t}_l)) + \frac{1}{s^2}\sum_{l,j=1,l\neq j}^s h(\Phi^{-1}(\mathbf{t}_l),\Phi^{-1}(\mathbf{t}_j))\Bigg) d\mathbf{t}_1\cdots d\mathbf{t}_s \,.
\end{aligned}$$

Obviously, the first is a constant which is independent to $\mathbf{t}_1, \ldots, \mathbf{t}_s$. Since all the terms are finite, we can interchange the integral and the sum among rest terms. In the second term, for each l, the only dependence on $\mathbf{t}_1, \ldots, \mathbf{t}_s$ is $\mathbf{t}_l$, hence all the other $\mathbf{t}_j$ can be integrated out. That is,

$$\begin{aligned}\int_{[0,1]^d} \cdots \int_{[0,1]^d} \int_{\mathbb{R}^d} h(\Phi^{-1}(\mathbf{t}_l), \omega) p(\omega) d\omega d\mathbf{t}_1 \cdots d\mathbf{t}_s &= \int_{[0,1]^d} \int_{\mathbb{R}^d} h(\Phi^{-1}(\mathbf{t}_l), \omega) p(\omega) d\omega d\mathbf{t}_l \\ &= \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} h(\phi, \omega) p(\phi) p(\omega) d\phi d\omega.\end{aligned}$$

Above, the last equality comes from a change of variable, i.e., $\mathbf{t}_l = (\Phi_1(\phi_1), \ldots, \Phi_d(\phi_d))$.

Similar operations can be done for the third and fourth term. Combining all of these, we have the following,

$$\begin{aligned}\mathbb{E}\left[D_{h,p}(S)^2\right] &= \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} h(\omega, \phi) p(\omega) p(\phi) d\omega d\phi - 2 \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} h(\omega, \phi) p(\omega) p(\phi) d\omega d\phi \\ &\quad + \frac{1}{s} \int_{\mathbb{R}^d} h(\omega, \omega) p(\omega) d\omega + \frac{s-1}{s} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} h(\omega, \phi) p(\omega) p(\phi) d\omega d\phi \\ &= \frac{1}{s} \int_{\mathbb{R}^d} h(\omega, \omega) p(\omega) d\omega - \frac{1}{s} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} h(\omega, \phi) p(\omega) p(\phi) d\omega d\phi \,.\end{aligned}$$

■

A.7 Proof of Proposition 16

Before we compute the derivative, we prove two auxiliary lemmas.

Lemma 19 *Let $\mathbf{x} \in \mathbb{R}^d$ be a variable and $\mathbf{z} \in \mathbb{R}^d$ be fixed vector. Then,*

$$\frac{\partial \operatorname{sinc}_{\mathbf{b}}(\mathbf{x}, \mathbf{z})}{\partial x_j} = b_j \operatorname{sinc}'_{b_j}(x_j, z_j) \prod_{q \neq j} \operatorname{sinc}_{b_q}(x_q, z_q) \,. \tag{23}$$

We omit the proof as it is a simple computation that follows from the definition of $\operatorname{sinc}_{\mathbf{b}}$.

Lemma 20 *The derivative of the scalar function $f(x) = \operatorname{Re}\left[e^{-ax^2} \operatorname{erf}(c + idx)\right]$, for real scalars a, c, d is given by,*

$$\frac{\partial f}{\partial x} = -2axe^{-ax^2} \operatorname{Re}\left[\operatorname{erf}(c + idx)\right] + \frac{2d}{\sqrt{\pi}} e^{-ax^2} e^{d^2x^2 - c^2} \sin(2cdx) \,.$$

Proof Since

$$\begin{aligned}f(x) &= \frac{1}{2}\left(e^{-ax^2} \operatorname{erf}(c + idx) + \left(e^{-ax^2} \operatorname{erf}(c + idx)\right)^*\right) \\ &= \frac{1}{2}\left(e^{-ax^2} \operatorname{erf}(c + idx) + e^{-ax^2} \operatorname{erf}(c - idx)\right) \,,\end{aligned} \tag{24}$$

it suffices to compute the the derivative $g(x) = e^{-ax^2} \operatorname{erf}(c + idx)$.

Let $k(x) = \operatorname{erf}(c + idx)$. We have

$$g'(x) = -2axe^{-ax^2} k(x) + e^{-ax^2} k'(x) \,. \tag{25}$$

Since

$$\begin{aligned}k(x) &= \operatorname{erf}(c + idx) \\ &= \frac{2}{\sqrt{\pi}} \int_0^{c+idx} e^{-z^2} dz \\ &= \frac{2}{\sqrt{\pi}} \left(\int_0^c e^{-z^2} dz + \int_c^{c+idx} e^{-z^2} dz\right) \\ &= \frac{2}{\sqrt{\pi}} \left(\int_0^c e^{-y^2} dy + (id) \int_0^x e^{-(c+idt)^2} dt\right) \,,\end{aligned} \tag{26}$$

we have

$$k'(x) = \frac{2}{\sqrt{\pi}} e^{-(c+idx)^2} = \frac{2d}{\sqrt{\pi}} e^{d^2x^2 - c^2} (\sin(2cdx) + i\cos(2cdx)) \,. \tag{27}$$

We now have

$$\begin{aligned}f'(x) &= \frac{1}{2}\left(g'(x) + (g^*(x))'\right) \\ &= \frac{1}{2}\left(g'(x) + (g'(x))^*\right) \\ &= \frac{1}{2}\left(-2axe^{-ax^2}(k(x) + k^*(x)) + e^{-ax^2}(k'(x) + (k'(x))^*)\right) \\ &= \frac{1}{2}\left(-4axe^{-ax^2} \operatorname{Re}\left[\operatorname{erf}(c + idx)\right] + e^{-ax^2} \frac{4d}{\sqrt{\pi}} e^{d^2x^2 - c^2} \sin(2cdx)\right) \\ &= -2axe^{-ax^2} \operatorname{Re}\left[\operatorname{erf}(c + idx)\right] + \frac{2d}{\sqrt{\pi}} e^{-ax^2} e^{d^2x^2 - c^2} \sin(2cdx) \,.\end{aligned} \tag{28}$$

■

Proof [Proof of Proposition 16] For the first term in (12), that is $\frac{1}{s^2} \sum_{m=1}^{s} \sum_{r=1}^{s} \operatorname{sinc}_{\mathbf{b}}(\mathbf{w}_m, \mathbf{w}_r)$, to compute the partial derivative of w_{lj}, we only have to consider when at least m or r is equal to l. If $m = j = l$, by definition, the corresponding term in the summation is one. Hence, we only have to consider the case when $m \neq r$. By symmetry, it is equivalent to compute the partial derivative of the following function $\frac{2}{s^2} \sum_{m=1, m\neq l}^{s} \operatorname{sinc}_{\mathbf{b}}(\mathbf{w}_l, \mathbf{w}_m)$. Applying Lemma 19, we get the first term in (14).

Next, for the last term in (12), we only have consider the term associated with one in the summation and the term associated with j in the product. Since $\left(\frac{\sigma_j}{\sqrt{2\pi}}\right) e^{-\frac{\sigma_j^2 w_{lj}^2}{2}} \operatorname{Re}\left(\operatorname{erf}\left(\frac{b_j}{\sigma_j\sqrt{2}} - i\frac{\sigma_j w_{lj}}{\sqrt{2}}\right)\right)$ satisfies the formulation in Lemma 20, we can simply apply Lemma 20 and get its derivative with respect to w_{lj}.

Equation (14) follows by combining these terms. ■

References

H. Avron, H. Nguyen, and D. Woodruff. Subspace embeddings for the polynomial kernel. In *Advances in Neural Information Processing Systems (NIPS)*. 2014.

F. Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory (COLT)*, 2013.

F. Bach, S. Lacoste-Julien, and G. Obozinski. On the equivalence between herding and conditional gradient algorithms. In *International Conference in Machine Learning (ICML)*, 2012.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434, 2006.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, 2004.

S. Bochner. Monotone funktionen, Stieltjes integrale und harmonische analyse. *Math. Ann.*, 108: 378–410, 1933.

B. Boots, A. Gretton, and G. J. Gordon. Hilbert space embeddings of predictive state representations. In *Conference Uncertainty in Artificial Intelligence (UAI)*, 2013.

L. Bottou, O. Chapelle, D. DeCoste, and J. Weston (Editors). *Large-scale Kernel Machines*. MIT Press, 2007.

R. E. Caflisch. Monte Carlo and Quasi-Monte Carlo methods. *Acta Numerica*, 7:1–49, 1 1998.

Y. Chen, M. Welling, and A. Smola. Super-samples from kernel herding. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bull. Amer. Math. Soc.*, 39: 1–49, 2001.

B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems (NIPS)*. 2014.

J. Dick, F. Y. Kuo, and I. H. Sloan. High-dimensional integration: The Quasi-Monte Carlo way. *Acta Numerica*, 22:133–288, 2013.

A. El Alaoui and M. W. Mahoney. Fast Randomized Kernel Methods With Statistical Guarantees. *ArXiv e-prints*, November 2014.

Z. Ghahramani and C. E. Rasmussen. Bayesian Monte Carlo. In *Advances in Neural Information Processing Systems (NIPS)*. 2003.

A. Gittens and M. W. Mahoney. Revisiting the Nyström method for improved large-scale machine learning. In *International Conference on Machine Learning (ICML)*, 2013. To appear in the Journal of Machine Learning Research.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008.

R. Hamid, A. Gittens, Y. Xiao, and D. DeCoste. Compact random feature maps. In *International Conference on Machine Learning (ICML)*, 2014.

Z. Harchaoui, F. Bach, O. Cappe, and E. Moulines. Kernel-based methods for hypothesis testing: A unified view. *IEEE Signal Processing Magazine*, 30(4):87–97, July 2013.

P. Huang, H. Avron, T. Sainath, V. Sindhwani, and B. Ramabhadran. Kernel methods match Deep Neural Networks on TIMIT. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

F. Huszár and D. Duvenaud. Optimally-weighted herding is Bayesian quadrature. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2012.

P. Kar and H. Karnick. Random feature maps for dot product kernels. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

S. S. Keerthi, O. Chapelle, and D. DeCoste. Building support vector machines with reduced classifier complexity. *J. Mach. Learn. Res.*, 7:1493–1515, 2006.

Q. Le, T. Sarlós, and A. Smola. Fastfood – Approximating kernel expansions in loglinear time. In *International Conference on Machine Learning (ICML)*, 2013.

G. Leobacher and F. Pillichschammer. *Introduction to Quasi-Monte Carlo Integration and Applications*. Springer International Publishing, 2014.

F. Li, C. Ionescu, and C. Sminchisescu. Random Fourier approximations for skewed multiplicative histogram kernels. *Pattern Recognition*, 6376:262–271, 2010.

Z. Lu, A. May, K. Liu, A. Bagheri Garakani, D. Guo, A. Bellet, L. Fan, M. Collins, B. Kingsbury, M. Picheny, and F. Sha. How to scale up kernel methods to be as good as deep neural net. *ArXiv e-prints*, November 2014.

S. Maji and A. C. Berg. Max-margin additive classifiers for detection. In *International Conference on Computer Vision (ICCV)*, 2009.

M. Mori. A method for evaluation of the error function of real and complex variable with high relative accuracy. *Publ. RIMS, Kyoto Univ.*, 19:1081–1094, 1983.

H. Niederreiter. *Random number generation and Quasi-Monte Carlo methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

E. Parzen. Statistical inference on time series by RKHS methods. In *Biennial Seminar Canadian Mathematical Congress on Time Series and Stochastic Processes: convexity and combinatorics*, 1970.

M. M. Peloso. Classical spaces of holomorphic functions. Technical report, Universit‘ di Milano, 2011.

N. Pham and R. Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2013.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*. 2008.

V. C. Raykar and R. Duraiswami. Fast large scale gaussian process regression using approximate matrix-vector products, 2007.

I. J. Schoenberg. Positive definite functions on spheres. *Duke Mathematical Journal*, 9(1):96–108, 03 1942.

B. Schölkopf and A. Smola, editors. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, 2002.

V. Sindhwani and H. Avron. High-performance kernel machines with implicit distributed optimization and randomization. In *JSM Proceedings, Tradeoffs in Big Data Modeling - Section on Statistical Computing*, 2014. To appear in Technometrics.

I. H. Sloan and H. Wozniakowski. When are Quasi-Monte Carlo algorithms efficient for high dimensional integrals. *Journal of Complexity*, 14(1):1–33, 1998.

A. Smola, A. Gretton, L. Song, and B. Schlkopf. A Hilbert space embedding for distributions. In *Algorithmic Learning Theory*, volume 4754 of *Lecture Notes in Computer Science*, pages 13–31. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-75224-0.

L. Song, B. Boots, S. Siddiqi, G. Gordon, and A. Smola. Hilbert space embeddings of Hidden Markov Models. In *International Conference in Machine Learning (ICML)*, 2010.

V. Sreekanth, A. Vedaldi, C. V. Jawahar, and A. Zisserman. Generalized RBF feature maps for efficient detection. In *British Machine Vision Conference (BMVC)*, 2010.

B. Sriperumbudur, A. Gretton, K. Fukumizu, B. Scholkopf, and G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *J. Mach. Learn. Res.*, 11:1517–1561, 2010.

J. F. Traub and H. Wozniakowski. Breaking intractability. *Scientific American*, pages 102–107, 1994.

I. W. Tsang, J. T. Kwok, and P. Cheung. Core vector machines: Fast svm training on very large data sets. *J. Mach. Learn. Res.*, 6:363–392, December 2005.

A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):480–492, March 2012.

G. Wahba, editor. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1990.

J. A. C. Weideman. Computation of the complex error function. *SIAM Journal of Numerical Analysis*, 31(5):1497–1518, 10 1994.

M. Welling. Herding dynamical weights to learn. In *International Conference on Machine Learning (ICML)*, 2009.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*. 2001.

H. Wozniakowski. Average case complexity of multivariate integration. *Bull. Amer. Math. Soc.*, 24: 185–194, 1991.

J. Yang, V. Sindhwani, Q. Fan, H. Avron, and M. Mahoney. Random Laplace feature maps for semigroup kernels on histograms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

K. Yao. Applications of Reproducing Kernel Hilbert Spaces - bandlimited signal models. *Inform. Control*, 11:429–444, 1967.

K. Zhang, J. Peters, D. Janzing, and B. Scholkopf. Kernel based conditional independence test and application in causal discovery. In *Confernece on Uncertainty in Artificial Intelligence (UAI)*, 2011.